

**Data Replication Service**

# Troubleshooting

**Issue** 01  
**Date** 2024-02-28



**Copyright © Huawei Technologies Co., Ltd. 2024. All rights reserved.**

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

## **Trademarks and Permissions**



HUAWEI and other Huawei trademarks are trademarks of Huawei Technologies Co., Ltd.

All other trademarks and trade names mentioned in this document are the property of their respective holders.

## **Notice**

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

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

## **Huawei Technologies Co., Ltd.**

Address: Huawei Industrial Base  
Bantian, Longgang  
Shenzhen 518129  
People's Republic of China

Website: <https://www.huawei.com>

Email: [support@huawei.com](mailto:support@huawei.com)

# Security Declaration

## Vulnerability

Huawei's regulations on product vulnerability management are subject to the *Vul. Response Process*. For details about this process, visit the following web page:

<https://www.huawei.com/en/psirt/vul-response-process>

For vulnerability information, enterprise customers can visit the following web page:

<https://securitybulletin.huawei.com/enterprise/en/security-advisory>

---

# Contents

---

<b>1 Solutions to Failed Check Items.....</b>	<b>1</b>
1.1 Disk Space.....	1
1.1.1 Checking Whether the Destination Database Has Sufficient Storage Space.....	1
1.1.2 Checking Whether the Destination Server Has Sufficient Storage Space.....	2
1.2 Database Parameters.....	2
1.2.1 Checking Whether the Source Database Binlog Is Enabled.....	2
1.2.2 Checking Whether the Source Database Binlog Is Row-Based.....	3
1.2.3 Checking Whether the expire_logs_days Value in the Source Database Is Correct.....	3
1.2.4 Checking Whether the Source and Destination Database Character Sets Are Consistent.....	4
1.2.5 Checking Whether the Source Database server_id Meets the Incremental Migration Requirements .....	5
1.2.6 Checking Whether the Source and Destination Database Table Names Are Consistent in Case Sensitivity.....	6
1.2.7 Checking Whether the Source Database Contains Object Names with Non-ASCII Characters.....	7
1.2.8 Checking Whether the time_zone Values of the Source and Destination Databases Are the Same.....	8
1.2.9 Checking Whether the collation_server Values of the Source and Destination Databases Are the Same.....	9
1.2.10 Checking Whether the SERVER_UUID Values of the Source and Destination Databases Are the Same.....	9
1.2.11 Checking Whether the SERVER_ID Values of the Source and Destination Databases Are Different .....	10
1.2.12 Checking Whether the Source Database Contains Invalid sql_mode Values.....	10
1.2.13 Checking Whether the sql_mode Values of the Source and Destination Databases Are the Same..	10
1.2.14 Checking Whether the sql_mode Value in the Destination Database Is Not no_engine.....	11
1.2.15 Checking Whether the innodb_strict_mode Values of the Source and Destination Databases Are the Same.....	11
1.2.16 Checking Whether the max_wal_senders Value of the Source Database Is Correctly Configured....	12
1.2.17 Checking Whether the WAL_LEVEL Value in the Source Database Is Correct.....	13
1.2.18 Checking Whether the MAX_REPLICATION_SLOTS Value in the Source Database Is Correct.....	14
1.2.19 Checking Whether the Source Database Is on Standby.....	14
1.2.20 Checking Whether the log_slave_updates Value in the Source Database Is Correct.....	15
1.2.21 Checking Whether the BLOCK_SIZE Value of the Source Database Is the Same as That of the Destination Database.....	16
1.2.22 Checking Whether the binlog_row_image Value is FULL.....	16
1.2.23 Checking Whether the Transaction Isolation Levels are Consistent.....	17

1.2.24 Checking Whether the lc_monetary Values of the Source and Destination Databases Are the Same	18
1.2.25 Checking Whether the Source Database Contains Trigger Names with Non-ASCII Characters	18
1.2.26 Checking Whether the Source Database Collections Contain More Than 10 Indexes	19
1.2.27 Checking Whether the Source Database Collections Contain TTL Indexes	19
1.2.28 Checking Whether log_bin_trust_function_creators Is Set to On in Both the Source and Destination Databases	20
1.2.29 Checking Whether GTID Is Enabled for the Source Database	21
1.2.30 Checking Whether GTID Is Enabled for the Destination Database	21
1.2.31 Checking Whether log_bin_trust_function_creators Is Set to On in the Destination Database	22
1.2.32 Checking Whether the Values in the Source Oracle Database Are Out of the Ranges of the MySQL Database	22
1.2.33 Checking Whether the max_allowed_packet Value of the Destination Database Is too Small	23
1.2.34 Checking Whether the Source Database User Has the Permission to Parse Logs	24
1.2.35 Checking Whether the ExpireLogsDays Value Is 0	25
1.2.36 Checking Whether the Source Database Character Set Is Supported	25
1.2.37 Checking Whether the Length Table and Field Names Is Not Supported	26
1.2.38 Checking Whether the Databases and Tables Exist	26
1.2.39 Checking Whether the Supplemental Log Level of the Source Database Meets Requirements	27
1.2.40 Checking Whether the Length of the Source Database Object Names Exceeds the Limit	28
1.2.41 Checking Whether session_replication_role of the Destination Database Is correctly Set	29
1.2.42 Checking the Database Compatibility Type	29
1.2.43 Checking Whether the Collation of the Destination Database Is Correct	30
1.2.44 Checking Whether the Index Name Is Supported	31
1.2.45 Checking Whether Tables Structures Are Consistent	31
1.2.46 Checking Whether Existing Data Meets the Constraints	32
1.2.47 Checking the Additional Column of the Destination Database	33
1.2.48 Checking Whether Implicit Primary Key Check Is Enabled for the Primary and Standby Databases	34
1.2.49 Checking Whether the Source Table Replication Attribute Is Correct	34
1.2.50 Checking Whether the Source Database Is on Standby	35
1.2.51 Checking Whether the Specified Replication Slot Exists in the Source Database	36
1.2.52 Checking Whether the MongoDB Instance Type Matches the Migration Mode	36
1.2.53 Checking the Physical Standby Database	37
1.2.54 Checking Whether the Case Sensitivity of the Destination Database Is Configured	37
1.2.55 Checking Whether CDC is Enabled for Tables in the Source Database	38
1.2.56 Checking Whether the CDC Retention Period in the Source Database Is Long Enough	38
1.2.57 Checking Whether the Source and Destination Databases Have Different Computer Names	39
1.2.58 Checking Whether the Length of the Source Database Name Exceeds 64	40
1.2.59 Synchronization Object Name Check	40
1.2.60 Checking Whether the Source Database Contains Disabled Clustered Indexes	41
1.2.61 Checking Whether the Source Database Is Empty	41
1.2.62 Checking Whether the Source Database Uses the Full Recovery Model	42

1.2.63 Checking the Synchronization Objects.....	43
1.2.64 Checking Whether the Source Database Tables Contain Data Types Not Supported for Migration .....	44
1.2.65 Checking Whether the SQL Server Agent of the Source Database Is Enabled.....	45
1.2.66 Checking Whether the Values of group_concat_max_len Are Consistent.....	45
1.2.67 Checking Whether the table structures of the source database are consistent.....	46
1.2.68 Checking Whether the Character Sets Are Compatible.....	46
1.2.69 Whether There Are XA Transactions That Have Not Been Submitted for a Long Time in the Source Database.....	47
1.2.70 Whether the Selected Objects Exist in the Destination Database.....	47
1.2.71 Destination Database Same Name Check.....	49
1.2.72 Whether the Destination Database User (Schema) and Table Exist.....	50
1.2.73 Whether the Maximum Number of Indexed Columns Has Been Reached.....	50
1.2.74 Checking the Length of the Index Column in the Source Database.....	51
1.2.75 Whether the Table Structures (Including Primary Key Indexes and the Number of Columns) of the Source Oracle Database and Destination Database Middleware Are Aligned.....	52
1.2.76 Whether Synchronization Objects Exist in the Destination Database.....	52
1.2.77 Whether the Source Database Contains Encrypted Objects.....	53
1.2.78 Checking Whether the Source Database Contains Unsupported Table Field Types.....	53
1.2.79 Checking Replication Attribute of Primary Key Columns.....	54
1.2.80 Whether There Are Tables Containing Fields of the longtext or longblob Type in the Synchronization Object.....	55
1.2.81 Checking Database Mapping Objects.....	56
1.2.82 Whether the Source Database Is the Standby Database of a GaussDB(for MySQL) Instance.....	57
1.2.83 Checking Whether Type Names Mapped to the Destination Database Are Valid.....	57
1.2.84 Checking Whether the Source and Destination Database Character Sets Are Consistent.....	58
1.2.85 Checking Whether Data Replication Is Enabled for the Source Database.....	59
1.2.86 Checking Whether the Maximum Sequence Number of the Source Database is Less Than That of the Destination Database.....	60
1.2.87 Checking Whether Interval Partitioned Tables Are Included in the Source Database.....	60
1.2.88 Oracle Account Check in the Source Database.....	61
1.2.89 Checking the Number of DNs in the Source Database.....	61
1.2.90 Whether the Selected Objects Exist in the Destination Database.....	62
1.2.91 Whether There Are Foreign Keys Containing Unsupported Reference Operations in the Source Database.....	63
1.2.92 Whether the Selected Table Contains Delay Constraints.....	63
1.2.93 Whether the Source Database Tables Contain Primary Keys.....	64
1.2.94 Whether Foreign Keys Are Disabled or Tables to Be Synchronized Have Foreign Keys in the Destination Database.....	64
1.2.95 Whether There Are Composite Hash Indexes in the Source Collection.....	65
1.2.96 Whether There Are Composite Hash Shard Keys in the Source Collection.....	66
1.2.97 Whether the Source Table Structure Contains Newline Characters.....	66
1.2.98 Whether There Are Tables Containing Fields of the bytea or text Type in the Synchronization Object.....	67

1.2.99 Whether the Source Table Structure Contains Virtual Columns.....	67
1.2.100 Whether the max_allowed_packet Value of the Source Database Is Too Small.....	68
1.2.101 Whether the Supplemental Log Level in the Source Database Is Correct.....	68
1.2.102 Whether Kafka Topics Have Been Created.....	69
1.2.103 Whether the Source Database Kernel Encoding Supports Data Replication.....	69
1.2.104 block_encryption_mode Consistency Check.....	70
1.2.105 Character Type and Sorting Rule Check in the Destination Database.....	70
1.2.106 Column Name Check in the Source Database.....	71
1.2.107 Whether the Destination Schemas and Table Objects Are Consistent.....	71
1.2.108 Source Encrypted Table Check.....	72
1.2.109 Checking Whether the Source Table Replication Attribute Is Correct.....	73
1.2.110 Partition Table Check on the Source Database.....	73
1.3 Source DB Instance Statuses.....	74
1.3.1 Checking Whether the Source DB Instance Is Available.....	74
1.3.2 Checking Whether the Source and Destination Databases Are of the Same Type.....	74
1.3.3 Checking Whether the ChangeStream API of the source DB instance is available.....	75
1.4 Destination DB Instance Statuses.....	75
1.4.1 Checking Whether the Destination DB Instance Is Available.....	75
1.4.2 Checking Whether the Destination Database Is Involved in Another Migration Task.....	76
1.4.3 Checking Whether the Destination Database Has a Read Replica.....	76
1.4.4 Checking Whether the Destination Database Is Read-Only.....	77
1.4.5 Checking Whether the Extensions Are Supported.....	77
1.4.6 Checking Whether Destination Contains the Configured Database.....	78
1.4.7 Checking Whether the Destination DB Instance Is Available.....	79
1.4.8 Checking Whether the Destination Database Is Empty.....	80
1.5 Database User Permissions.....	80
1.5.1 Whether the Source Database User Has Sufficient Permissions.....	80
1.5.2 Checking Whether the Destination Database User Has Sufficient Permissions.....	81
1.5.3 Checking Whether the Destination Database Account Has Required Permissions to Migrate Definer .....	82
1.6 Database Versions.....	82
1.6.1 Checking Whether the Source Database Version Is Supported.....	83
1.6.2 Checking Whether the Destination Database Version Is Supported.....	83
1.6.3 Checking Whether the Migration Is from an Earlier Database Version to the Same or a Later Version .....	83
1.7 Networks.....	83
1.7.1 Checking Whether the Source Database Is Connected.....	84
1.7.2 Checking Whether the Destination Database Is Connected.....	85
1.7.3 Checking Whether the Destination Database Can Connect to the Source Database.....	87
1.8 Database Objects.....	87
1.8.1 Checking Whether the Source Database Contains a MyISAM Table.....	87
1.8.2 Checking Whether the Source Database Contains Unsupported Table Field Types.....	88

1.8.3 Checking Whether the Source Database Contains the Functions or Stored Procedures that the Source Database User Is Not Authorized to Migrate.....	89
1.8.4 Checking Whether Objects with the Same Names Exist in the Source Database.....	89
1.8.5 Whether the Source Database Contains Unlogged Tables.....	90
1.8.6 Checking Whether the Names of Views to Be Migrated Are the Same.....	90
1.8.7 Checking Whether the _Id Fields in the Collection of the Source Database Have Indexes.....	91
1.8.8 Checking Whether the Index Length of the Source Database Exceeds the Limit.....	92
1.8.9 Checking Whether the Source Database Tables Use Storage Engines Not Supported by the Destination Database.....	93
1.8.10 Checking Whether the Database Names Mapped to the Destination DB Instance Contain Unsupported Characters.....	93
1.8.11 Checking Whether the Source Database Tables Contain Primary Keys.....	94
1.8.12 Checking Whether the Source Database Contains Triggers or Events.....	96
1.8.13 Checking Whether the Source Database Referenced Roles Pass the Check.....	96
1.8.14 Checking Whether the Source Database Referenced Accounts Pass the Check.....	97
1.8.15 Checking Whether the Source Database Contains Schemas or Users Named cdc.....	98
1.8.16 Checking Whether Associated Objects Are Selected.....	98
1.8.17 Checking Whether the Specified Objects Exist In the Destination Database.....	99
1.8.18 Checking Whether the Source Table Contains Column Types that Cannot Be Used as Distribution Keys.....	99
1.8.19 Checking Whether the Source Table Contains Unsupported Table Field Types.....	100
1.9 Database Configuration Items.....	100
1.9.1 Checking Whether the Source Database Name Is Valid.....	100
1.9.2 Checking Whether the Source Database Table Name Is Valid.....	102
1.9.3 Checking Whether the Source Database View Name Is Valid.....	103
1.9.4 Checking Whether the Source Database Collection Name Is Valid.....	104
1.9.5 Checking Whether the Shard Key Can Be Obtained from the Source Database.....	104
1.9.6 Checking Whether the Source Database Schema Name Is Valid.....	105
1.9.7 Checking Whether the Maximum Number of Chunks in the Destination Database Is Sufficient.....	106
1.9.8 Checking Whether Archive Logs Are Enabled on the Source Oracle Database.....	106
1.9.9 Checking Whether Supplemental Logging Is Correctly Enabled on the Source Database.....	107
1.10 Conflicts.....	107
1.10.1 Checking Whether the Names of the Source and Destination Databases Are the Same.....	107
1.10.2 Checking Whether the Same View Names Exist in Both the Source and Destination Databases...	111
1.10.3 Checking Whether the Destination Database Contains a Non-Empty Collection with the Same Name As the Source Database.....	112
1.10.4 Checking Whether Destination Database Contains the Same Table Names As the Synchronization Objects.....	113
1.10.5 Checking Whether the Destination Database Contains Objects with the Same Name As Those in the Source Database.....	114
1.10.6 Checking Whether Collections in Both the Source and Destination Databases Are Not Capped...	117
1.11 SSL Connections.....	118
1.11.1 Checking Whether the SSL Connection Is Correctly Configured.....	118
1.11.2 Checking Whether the SSL Connection Is Enabled for the Source Database.....	118



1.11.3 Checking Whether the SSL Certificate of the Source Database Exists.....	119
1.11.4 Checking Whether the SSL Certificate of the Destination Database Exists.....	120
1.11.5 Checking Whether Both the Source and Destination Databases Use SSL.....	120
1.12 Object Dependencies.....	121
1.12.1 Checking Whether the Objects Referenced by Views Are Selected for Migration.....	121
1.12.2 Checking Whether Referenced Tables Are Selected for Migration.....	122
1.13 Source Database Information.....	122
1.13.1 Checking Whether the Shards and Mongos Are in the Same Cluster.....	123
1.13.2 Checking Whether the Balancers of the Source Database Is Enabled.....	123
1.13.3 Checking Whether the Source and Destination Database Types Match.....	124
<b>2 Failure Cases.....</b>	<b>125</b>
2.1 Case Overview.....	125
2.2 Real-Time Migration from MongoDB to DDS.....	132
2.2.1 Full Migration Error: Prematurely reached end of stream.....	133
2.2.2 Full Migration Error: not authorized on *** to execute command {***}.....	133
2.2.3 Full Migration Error: GC overhead limit exceeded.....	134
2.2.4 Full Migration Error: The background process is unavailable. Maybe it has been killed manually or by the operating system. Please restart the task if possible or wait for restarting by itself within 5 minutes.".....	134
2.2.5 Full Migration Error: Timed out after 60000 ms while waiting to connect.....	135
2.2.6 Full or Incremental Migration Error: Timed out after 60000 ms while waiting to connect.....	135
2.2.7 Full or Incremental Migration Error: Invalid BSON field name ***.....	136
2.2.8 Incremental Migration Error: Timed out after 60000 ms while waiting for a server that matches com.mongodb.client.internal.....	136
2.2.9 Incremental Migration Error: Command failed with error *** (***):***. The full response is {***}".....	137
2.3 Real-Time Migration and Synchronization from MySQL to MySQL.....	137
2.3.1 Full Phase Error: Table *** doesn't exist.....	137
2.3.2 Full Phase Error: The background process is unavailable.....	138
2.3.3 Full Phase Error: Communications link failure The last packet sent successfully to the server was 0 milliseconds ago. The driver has not received any packets from the server.....	138
2.3.4 Full Phase Error: Error writing file *** (errno: 28 - No space left on device).....	139
2.3.5 Full Phase Error: The MySQL server is running with the --super-read-only option so it cannot execute this statement.....	139
2.3.6 Full Phase Error: The table *** is full.....	140
2.3.7 Full Phase Error: Unknown column *** in 'field list'.....	141
2.3.8 Full Phase Error: Lock wait timeout exceeded; try restarting transaction.....	141
2.3.9 Full Phase Error: Java heap space.....	142
2.3.10 Full Phase Error: Table *** already exists.....	142
2.3.11 Full Phase Error: temp table: *** not exist.....	142
2.3.12 Full Phase Error: failed to create new session.....	143
2.3.13 Full Phase Error: load table: *** failed.....	143
2.3.14 Full Phase Error: extract table structure failed!.....	144
2.3.15 Full Phase Error: read table=*** failed.....	144

2.3.16 Full Phase Error: CANNOT UPDATE USER WITH NULL PASSWORD.....	145
2.3.17 Full Phase Error: Access denied for user *** to database *** .....	145
2.3.18 Full Phase Error: The MySQL server is running with the --super-read-only option so it cannot execute this statement.....	146
2.3.19 Full Phase Error: Temporary file write failure.....	146
2.3.20 Full Phase Error: Incorrect prefix key; the used key part isn't a string, the used length is longer than the key part, or the storage engine doesn't support unique prefix keys.....	147
2.3.21 Full Phase Error: Unknown database ***.....	147
2.3.22 Full Phase Error: Access denied; you need (at least one of) the SUPER privilege(s) for this operation.....	148
2.3.23 Full Phase Error: retry structures failed events and Table *** doesn't exist.....	149
2.3.24 Full Phase Error: shard table=*** failed.....	149
2.3.25 Full Phase Error: error when split table shard occur!.....	149
2.3.26 Full Phase Error: Column name 'AUTO_PK_ROW_ID' is reserved.....	150
2.3.27 Full Phase Error: transfer account failed, can not find password from src DB.....	150
2.3.28 Full Phase Error: Failed to add the foreign key constraint '****' to system tables.....	151
2.3.29 Full Phase Error: Too many keys specified; max 64 keys allowed.....	152
2.3.30 Full Phase Error: Unknown collation: 'utf8mb4_0900_ai_ci'.....	152
2.3.31 Full or Incremental Phase Error: Access denied for user ***.....	153
2.3.32 Full or Incremental Phase Error: binlog is not existed.....	153
2.3.33 Full or Incremental Phase Error: database log download failed.....	154
2.3.34 Full or Incremental Phase Error: Can not read response from server.....	154
2.3.35 Full or Incremental Phase Error: Communications link failure.....	154
2.3.36 Full or Incremental Phase Error: EOF Packet received, master disconnected.....	155
2.3.37 Full or Incremental Phase Error: Extract db create sql failed.....	155
2.3.38 Full or Incremental Phase Error: load database structure failed in source database.....	156
2.3.39 Full or Incremental Phase Error: load table: *** failed.....	156
2.3.40 Full or Incremental Phase Error: Reached end of input stream.....	157
2.3.41 Full or Incremental Phase Error: Read timed out.....	157
2.3.42 Full or Incremental Phase Error: The background process is unavailable.....	158
2.3.43 Full or Incremental Phase Error: Duplicate entry *** for key 'PRIMARY'.....	158
2.3.44 Full or Incremental Phase Error: cause by: Index: ***, Size: *** .....	159
2.3.45 Full or Incremental Phase Error: The offset and file name between src and parser is inconsistency .....	160
2.3.46 Full or Incremental Phase Error: core process is not healthy or crashed.....	161
2.3.47 Full or Incremental Phase Error: table info of table `****` from metadata miss.....	161
2.3.48 Full or Incremental Phase Error: binlog parse fail, data dictionary may be not complete!.....	162
2.3.49 Full or Incremental Phase Error: table *** record field size for insert/delete dml.....	162
2.3.50 Full or Incremental Phase Error: service *** failed, cause by: Unable to connect to DBMS: ***.....	162
2.3.51 Full or Incremental Phase Error: The binlog fetch connection may be interrupted.....	163
2.3.52 Full or Incremental Phase Error: Received error packet: errno - 1047, solstate - HY000 errmsg = Unknown command.....	163
2.3.53 Incremental Phase Error: not equals to target db column count.....	164
2.3.54 Incremental Phase Error: The MySQL server is running with the --super-read-only option.....	164

2.3.55 Incremental Phase Error: you need (at least one of) the SUPER privilege(s) for this operation.....	165
2.3.56 Incremental Phase Error: Can't DROP ***; check that column/key exists.....	165
2.3.57 Incremental Phase Error: Can't find file: *** (errno: 2 - No such file or directory).....	165
2.3.58 Incremental Phase Error: Data truncation: Data too long for column.....	166
2.3.59 Incremental Phase Error: Failed to read file header from.....	166
2.3.60 Incremental Phase Error: Lock wait timeout exceeded.....	167
2.3.61 Incremental Phase Error: Must seek before attempting to read next event.....	167
2.3.62 Incremental Phase Error: Table *** already exists.....	168
2.3.63 Incremental Phase Error: Table *** doesn't exist.....	168
2.3.64 Incremental Phase Error: Table *** not found in database.....	169
2.3.65 Incremental Phase Error: source has more columns than target.....	169
2.3.66 Incremental Phase Error: Unknown storage engine.....	170
2.3.67 Incremental Phase Error: Unknown table.....	170
2.3.68 Incremental Phase Error: You have an error in your SQL syntax.....	171
2.3.69 Incremental Phase Error: not illegal for mariaDb gtid position.....	171
2.3.70 Incremental Phase Error: without PK execute failed.....	171
2.3.71 Incremental Phase Error: Deadlock found when trying to get lock.....	172
2.3.72 Incremental Phase Error: current serverUUID not equals to this session.....	172
2.3.73 Incremental Phase Error: Slave has more GTIDs than the master has, using the master's SERVER_UUID.....	173
2.3.74 Incremental Phase Error: Operation not allowed when innodb_force_recovery > 0.....	173
2.3.75 Incremental Phase Error: filter data in config condition filter error.....	174
2.4 Real-Time Migration and Synchronization from MySQL to GaussDB(for MySQL).....	174
2.4.1 Full or Incremental Phase Error: Illegal mix of collations (utf8mb4_0900_ai_ci,IMPLICIT) and (utf8mb4_general_ci,IMPLICIT) for operation.....	174
2.5 Real-Time Synchronization from MySQL to GaussDB(DWS).....	175
2.5.1 Full Synchronization Error: Table *** not found in database.....	175
2.5.2 Full Synchronization Error: column 'database_table' of relation *** does not exist.....	176
2.5.3 Full Synchronization Error: value too long for type character varying.....	176
2.5.4 Full Synchronization Error: int1 has not implemented.....	177
2.5.5 Full Synchronization Error: column name 'tid' conflicts with a system column name.....	177
2.5.6 Full Synchronization Error: date/time field value out of range.....	178
2.5.7 Full or Incremental Synchronization Error: service LOGMANAGER failed.....	178
2.5.8 Full or Incremental Synchronization Error: service CAPTURER failed.....	179
2.5.9 Full or Incremental Synchronization Error: ERROR: pooler.....	179
2.5.10 Full or Incremental Phase Error: service *** failed, cause by: Unable to connect to DBMS: *** .....	180
2.5.11 Full or Incremental Phase Error: The binlog fetch connection may be interrupted.....	180
2.5.12 Incremental Synchronization Error: dn_***: column *** contains null values.....	181
2.5.13 Incremental Synchronization Error: source has more columns than target.....	181
2.5.14 Incremental Synchronization Errors: Connection to *.*:98:8000 refused. Check that the hostname and port are correct and that the postmaster is accepting TCP/IP connections.....	182
2.5.15 Incremental Synchronization Error: Table *** not found in target database.....	182
2.5.16 Incremental Synchronization Error: in a read-only transaction.....	183

2.5.17 Incremental Synchronization Error: relation *** does not exist.....	183
2.5.18 Incremental Synchronization Error: *** doesn't in the target table.....	184
2.5.19 Incremental Synchronization Error: syntax error at or near.....	184
2.5.20 Incremental Synchronization Error: schema *** does not exist.....	185
2.5.21 Incremental Synchronization Error: Check whether dws supports the DDL.....	185
2.5.22 Incremental Synchronization Error: PL/pgSQL function *** line *** at SQL statement.....	186
2.6 Real-Time Synchronization from MySQL to CSS/ES.....	186
2.6.1 Incremental Synchronization Error: write table *** failed: null.....	186
2.7 Real-Time Synchronization from PostgreSQL to PostgreSQL.....	187
2.7.1 Full Synchronization Error: function *** does not exist.....	187
2.7.2 Full Synchronization Error: relation *** does not exist.....	188
2.7.3 Full Synchronization Error: GC overhead limit exceeded.....	188
2.7.4 Full Synchronization Error: Java heap space.....	189
2.7.5 Full Synchronization Error: column *** of relation *** does not exist.....	189
2.7.6 Full Synchronization Error: column *** does not exist.....	190
2.7.7 Full Synchronization Error: type 'hstore' does not exist.....	190
2.7.8 Full Synchronization Error: type 'geometry' does not exist.....	191
2.7.9 Full Synchronization Error: Check that the hostname and port are correct and that the postmaster is accepting TCP/IP connections.....	191
2.7.10 Full Synchronization Error: invalid locale name.....	192
2.7.11 Full Synchronization Error: password must not equal user name.....	192
2.7.12 Full Synchronization Error: permission denied for schema ***.....	193
2.7.13 Full or Incremental Phase Error: service *** failed, cause by: Unable to connect to DBMS: ***.....	193
2.7.14 Full or Incremental Phase Error: Initialize logical replication stream failed, the source database may have a long transaction.....	194
2.7.15 Full or Incremental Phase Error: memory required is *** MB, maintenance_work_mem is *** MB..	194
2.7.16 Full or Incremental Phase Error: temporary file size exceeds temp_file_limit.....	195
2.7.17 Incremental Synchronization Error: Table *** not found in target database.....	196
2.7.18 Incremental Synchronization Error: remaining connection slots are reserved.....	197
2.7.19 Incremental Synchronization Error: PL/pgSQL function *** line *** at SQL statement.....	197
2.8 Real-Time Synchronization with Oracle Serving as the Source.....	198
2.8.1 Full Synchronization Error: has date/datetime: *** which is outside of dest allowed range.....	198
2.8.2 Full or Incremental Phase Error: Got minus one from a read call.....	198
2.8.3 Incremental Synchronization Error: Source supplemental log level is PK/UI. Missing column data at delete+insert on ***".....	199
2.8.4 Incremental Synchronization Error: timeout when get next file log, maybe has been deleted, please check it.....	200
2.8.5 Incremental Synchronization Error: Failed to construct kafka producer.....	201
2.8.6 Incremental Synchronization Error: Topic *** not present in metadata after 300000 ms.....	201
2.9 Real-Time DR with MySQL Serving as the Source.....	201
2.9.1 DR Error: A dml without pk write target db fail.....	202
2.10 Backup Migration.....	203
2.10.1 Backup Migration Failed Because Backup Files Cannot Be Found.....	203

2.10.2 Backup Migration Failed Because a Backup Database Cannot Be Found in the Backup Files.....	204
2.10.3 Backup Migration Failed Because the Database with the Same Name Already Exists.....	205
2.10.4 Backup Migration Failed Because an Incremental Backup File Is Used.....	205
2.10.5 Backup Migration Failed Because an Log Backup File Is Used.....	206
2.10.6 Backup Migration Failed Because the Backup File Verification Failed.....	206
2.10.7 Backup Migration Failed Because of Insufficient Space.....	206
2.10.8 Backup Migration Failed Because Database Names Are Not Specified.....	207
2.10.9 Backup Migration Failed Because a Full Backup File Is Used.....	207
2.10.10 Backup Migration Failed Because the LSNs of Incremental Backup Files Are Inconsecutive.....	208
2.10.11 Backup Migration Failed Because the Number of Databases to Be Restored Exceeds the Destination Database Threshold.....	208
2.11 Workload Replay.....	208
2.11.1 Parsing Failed, and a Message Is Displayed Indicating That the OBS Connection Failed.....	209
2.12 Data-Level Comparison.....	209
2.12.1 Data-Level Comparison Error: service SDV failed! cause by: the size of records in one shard[ *** ] of target database, exceeds the max size 200000.....	210

# 1 Solutions to Failed Check Items

---

## 1.1 Disk Space

### 1.1.1 Checking Whether the Destination Database Has Sufficient Storage Space

In the migration phase, DRS uses the row-level parallel migration mode to ensure migration performance and transmission stability. If the source database data is compact, the destination database will use more storage space than the source database after DRS migration. DRS checks the available storage space of the destination database during the pre-check. If the storage space is insufficient, the migration may fail.

#### Failure Cause

If the storage space of the destination database is insufficient, it is recommended that the size of the destination database disk be set to the smaller value of the following two values:

1. 1.5 or 2.5 times the size of the data to be migrated in the source database.
2. The size of the data to be migrated in the source database plus 200 GB.

The available storage space of the destination database is subject to the information displayed on the page.

#### Handling Suggestion

Scale up the storage of the destination database or clear data in the destination database.

For details about how to scale up an RDS for MySQL DB instance, see [Scaling Up Storage Space](#).

If you choose to clear data in the destination database, the storage usage decreases within 2 to 3 minutes.

## 1.1.2 Checking Whether the Destination Server Has Sufficient Storage Space

**Table 1-1** Checking whether the destination server has sufficient storage space

<b>Check Item</b>	Whether the destination server has sufficient storage space
<b>Description</b>	If the destination server's storage space is insufficient, the migration will fail.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The amount of data in the source database is greater than the remaining storage space of the destination server. Handling suggestion: Modify the synchronization object.

## 1.2 Database Parameters

### 1.2.1 Checking Whether the Source Database Binlog Is Enabled

Check whether Binlog is enabled for the source database. During an incremental MySQL migration, Binlog of the source database must be enabled.

#### Failure Cause

Binlog is not enabled for the source database.

#### Handling Suggestion

- If the source is an on-premises MySQL database, perform the following operations to enable Binlog.

- a. Run the following command to check whether Binlog is enabled:  
`show variables like "log_bin";`

```
mysql> show variables like "log_bin";
***** 1. row *****
Variable_name: log_bin
Value: OFF
1 row in set (0.01 sec)
```

- b. If Binlog is disabled, add `log-bin = mysql-bin` followed by `[mysqld]` in the MySQL configuration file `my.cnf` or `my.ini`.

```
[mysqld]
log-bin = mysql-bin
```

- c. Restart the database.

```
mysql> show variables like "log_bin"\G;
***** 1. row *****
Variable_name: log_bin
Value: ON
1 row in set (0.00 sec)
```

- If the source is an RDS for MySQL DB instance, Binlog is enabled by default and no additional configuration is required. **binlog\_format** is set to **row**.
- If the source is a TiDB database, add **enable=true** followed by **[binlog]** in the database configuration file and restart the database to apply the settings.

## 1.2.2 Checking Whether the Source Database Binlog Is Row-Based

Check whether the source database binlog format is correct. The binlog of the source database must be enabled and the row-based format must be used during incremental MySQL migration.

### Failure Cause

The source database binlog is not row-based.

### Handling Suggestion

- If the source database is an on-premises MySQL database, perform the following operations to change the binlog format of the source database:
  - Method 1: You can modify the **my.cnf** or **my.ini** configuration file and restart the database.  
`binlog_format=row`
  - Method 2: Stop all service connections.  
`set global binlog_format='ROW'`  
Modify the **my.cnf** or **my.ini** configuration file.  
`binlog_format=row`

In the **ROW** format, the log growth rate increases, which may occupy more disk space.

- If the source database is an RDS for MySQL DB instance, change the value of **binlog\_format** of the source database to **row** by following the instructions provided in [Modifying Parameters of an RDS for MySQL Instance](#). Restart the database to apply the change.

## 1.2.3 Checking Whether the expire\_logs\_days Value in the Source Database Is Correct

During MySQL migration, you can set **expire\_logs\_days** to change the binlog retention period. Set **expire\_logs\_day** to a proper value to ensure that the binlog does not expire before data transfer resumes. This ensures that services can be recovered after interruption.

### Failure Cause

The **expire\_logs\_days** parameter of the source database is set to **0**.



## Handling Suggestion

- If the source database is an RDS for MySQL DB instance, set **expire\_logs\_days** to a proper value by following the instructions provided in [Setting a Local Retention Period for RDS for MySQL Binlogs](#).
- If the source database is an on-premises MySQL database, perform the following steps:
  - a. Log in to the server where the MySQL source database is located.
  - b. Run the following command to check the configured binlog retention period:

```
show variables like 'expire_logs_days';
```

Or

```
show variables like 'binlog_expire_logs_seconds';
```
  - c. Manually modify the **my.cnf** configuration file and set the binlog retention period. The following uses three days as an example.

```
expire_logs_days=3
```

Or

```
binlog_expire_logs_seconds=259200;
```
  - d. After the modification, restart the source database during a non-service period.

## 1.2.4 Checking Whether the Source and Destination Database Character Sets Are Consistent

Check whether the source and destination database character sets are consistent. If the character set of the source database is different from that of the destination database, some data may be garbled or inconsistent.

### Failure Cause

The character sets of the source and destination databases are inconsistent.

### Handling Suggestion

Change the character set of the source or destination database.

- If a MySQL database is used, perform the following operations:
  - If the database is a self-managed database, run commands to change the character set.
    - i. Run the following command to check the character set of the database:

```
SHOW VARIABLES LIKE "character_set_server"\G;
```

```
mysql> show variables like "character_set_server"\G;
***** 1. row *****
Variable_name: character_set_server
Value: utf8
1 row in set (0.00 sec)
```
    - ii. Modify the character set of the source database server.

```
SET character_set_server='utf8';
```

```
mysql> set character_set_server='utf8';  
Query OK, 0 rows affected (0.00 sec)
```

- If the database is an RDS for MySQL DB instance, modify the **character\_set\_server** parameter.
- If a PostgreSQL database is used, perform the following operations:
  - If the database is a self-managed database, run commands to change the character set.
    - i. Run the following command to check the character set of the database:  

```
show server_encoding;
```
    - ii. Modify the character set of the source database server.  

```
set server_encoding='utf8';
```
  - If the database is an RDS for PostgreSQL DB instance, modify the **server\_encoding** parameter.

For details about how to modify character set parameters for other types of databases, see the usage guide of the corresponding database.

## 1.2.5 Checking Whether the Source Database **server\_id** Meets the Incremental Migration Requirements

During an incremental MySQL migration, the source database **server\_id** must meet the following requirements:

- If the source database version is MySQL 5.6 or earlier, the value of **server\_id** ranges from 2 to 4294967296.
- If the source database version is MySQL 5.7 or later, the value of **server\_id** ranges from 1 to 4294967296.

### Failure Cause

The **server\_id** value of the source database does not meet requirements.

### Handling Suggestion

**Step 1** Log in to the server where the MySQL source database is located.

**Step 2** Run the following SQL statement to check the value of **server\_id**:

```
show variables like '%server_id%';
```

**Step 3** If the value of **server\_id** does not meet requirements, run the following command to change the value of **server\_id**:

```
set global server_id=n
```

The value **n** indicates the source database **server\_id**. If the source database version is MySQL 5.6, the value **n** ranges from 2 to 4294967296. Otherwise, the value **n** ranges from 1 to 4294967296.

**Step 4** After the modification, perform the pre-check again.

----End

## 1.2.6 Checking Whether the Source and Destination Database Table Names Are Consistent in Case Sensitivity

### MySQL Migration

**Table 1-2** Checking whether the source and destination database table names are consistent in case sensitivity

<b>Check Item</b>	Whether the source and destination database table names are consistent in case sensitivity
<b>Description</b>	Check whether the source and destination database names and table names are consistent in case sensitivity.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: This item cannot be checked because the source database fails to be connected. Handling suggestion: Check whether the source database is connected.
	Failure cause: Insufficient user permissions Handling suggestion: Check whether the database user permissions meet the migration requirements.
	Failure cause: The <b>lower_case_table_names</b> values in the source and destination databases must be the same. Handling suggestion: <ul style="list-style-type: none"><li>• If you are migrating data out of the cloud, change the values of <b>lower_case_table_names</b> in the source and destination databases to the same. You are advised to change the parameter value in an empty database. For example, if the destination RDS DB instance is empty, run the following example command to change the <b>lower_case_table_names</b> value to the same as that in the source database: Sample command: <pre>set global lower_case_table_names=n;</pre> In the preceding command, <b>n</b> indicates the parameter value of the source database. After the modification, restart the database for the modification to take effect.</li><li>• If you are migrating data out of the cloud, perform the following operations: If the destination database is a self-built database, modify the <b>lower_case_table_names</b> parameter of the destination database. Add <b>lower_case_table_names=n</b> under the <b>[mysqld]</b> tag in the MySQL configuration file <b>my.cnf</b>. <b>n</b> indicates the value of parameter <b>same lower_case_table_names</b> of the source database. The database must be restarted to make the change take effect. If the destination database is a cloud database, check whether the <b>lower_case_table_names</b> parameter can be modified. If not, contact Huawei technical support.</li></ul>

	Failure cause: The <b>lower_case_table_names</b> parameter value of the destination database is different from that of the source database, and the source database contains uppercase database and table names. Handling suggestion: Rectify the fault by referring to <a href="#">FAQs</a> .
	Failure cause: The database is unavailable. Handling suggestion: Contact Huawei technical support.

## MySQL->MySQL, MySQL->Gauss(for MySQL) and Gauss(for MySQL) to MySQL Synchronization

**Table 1-3** Checking whether the source and destination database table names are consistent in case sensitivity

<b>Check Item</b>	Whether the source and destination database table names are consistent in case sensitivity
<b>Description</b>	The destination database is case insensitive. The names of the mapped databases and tables contain uppercase letters. If the destination database is case-insensitive, all uppercase letters are converted to lowercase letters for storage.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The destination database is case insensitive. All uppercase letters are converted to lowercase letters for storage. The names of the mapped databases and tables contain uppercase letters. Handling suggestion: If the mapping relationship is correct, change the database and table names that contain uppercase letters to lowercase letters.

### 1.2.7 Checking Whether the Source Database Contains Object Names with Non-ASCII Characters

#### MySQL

**Table 1-4** Checking whether the source database contains object names with non-ASCII characters

<b>Check Item</b>	Whether the source database contains object names with non-ASCII characters
<b>Description</b>	If the source database contains object names with non-ASCII characters, the migration will fail.

<b>Failure Cause and Handling Suggestion</b>	Failure cause: The source database cannot contain object names with non-ASCII characters. Handling suggestion: In the source database, change the object names containing non-ASCII characters.
--	--

## 1.2.8 Checking Whether the `time_zone` Values of the Source and Destination Databases Are the Same

If the `time_zone` values of the source and destination databases are different, the migration may fail.

### Failure Cause

The `time_zone` or `system_time_zone` values of the source and destination databases must be the same.

### Handling Suggestion

Change the value of `time_zone` (`timezone`) or `system_time_zone` of the source database to be the same as that of the destination database.

- To change the value of `time_zone` in the MySQL database, perform the following steps:
  - If the database is a self-managed database, run commands to change the time zone.
    - i. Run the following command to check the time zone of the database:  

```
SHOW VARIABLES LIKE "%time_zone%";
```
    - ii. Run the following command to change the time zone:  

```
SET time_zone = 'Timezone';
```
  - If the database is an RDS for MySQL DB instance, change the time zone by following the instructions provided in [How Can I Change the Time Zone?](#)
- To change the value of `time_zone` in the Oracle database, perform the following steps:
  - a. Run the following statement to query the value of `time_zone` in the database:  

```
SELECT DBTIMEZONE FROM DUAL;
```
  - b. Run the following statement to change the value of `time_zone` in the database:  

```
ALTER DATABASE SET TIME_ZONE='Time zone';
```

Example of changing the time zone to GMT+8:

```
ALTER DATABASE SET TIME_ZONE='+08:00';
```
  - c. Restart the database after changing the value of `time_zone`.  

```
SQL> shutdown immediate  
SQL> startup
```
- To change the value of `time_zone` in the DDM database, perform the following steps:

Log in to the DDM console and change the time zone.

For details about how to change the time zone for other types of databases, see the usage guide of the corresponding database.

## 1.2.9 Checking Whether the `collation_server` Values of the Source and Destination Databases Are the Same

If the `collation_server` values of the source and destination databases are different, the migration may fail.

### Failure Cause

The `collation_server` values of the source and destination databases must be the same.

### Handling Suggestion

Change the value of `collation_server` of the source database to be the same as that of the destination database.

- If the database is a self-managed MySQL database, run commands to change the value.
  - a. Run the following command to check the value:

```
SHOW VARIABLES LIKE "collation_server";
```
  - b. Run the following command to change the value:

```
SET collation_server='utf8_unicode_ci';
```
- If the database is an RDS for MySQL DB instance, change the value of the `collation_server` parameter by referring to [Modifying Parameters of an RDS for MySQL Instance](#).

For details about how to change the value of `collation_server` for other types of databases, see the usage guide of the corresponding database.

## 1.2.10 Checking Whether the `SERVER_UUID` Values of the Source and Destination Databases Are the Same

### MySQL Migration

**Table 1-5** Checking whether the `SERVER_UUID` values of the source and destination databases are the same

<b>Check Item</b>	Whether the <code>SERVER_UUID</code> values of the source and destination databases are the same
<b>Description</b>	If the <code>SERVER_UUID</code> values of the source and destination databases are the same, the migration fails.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The <code>SERVER_UUID</code> values of the source and destination databases must be different. Handling suggestion: Check that the source and destination databases are not the same MySQL database.

## 1.2.11 Checking Whether the SERVER\_ID Values of the Source and Destination Databases Are Different

### MySQL

**Table 1-6** Checking whether the SERVER\_ID values of the source and destination databases are different

<b>Check Item</b>	Whether the <b>SERVER_ID</b> values of the source and destination databases are different
<b>Description</b>	Check whether the <b>SERVER_ID</b> values of the source and destination databases are different. If they are the same, the migration fails.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The <b>SERVER_ID</b> values of the source and destination databases must be different. Handling suggestion: Change <b>SERVER_ID</b> of the source and destination databases to different values.

## 1.2.12 Checking Whether the Source Database Contains Invalid sql\_mode Values

### MySQL

**Table 1-7** Checking whether the source database contains invalid sql\_mode values

<b>Check Item</b>	Whether the source database contains invalid <b>sql_mode</b> values
<b>Description</b>	If the source database contains invalid <b>sql_mode</b> values, the migration will fail.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The <b>sql_mode</b> value of the source database cannot be <b>no_engine_substitution</b> . Handling suggestion: Change <b>sql_mode</b> of the source database to a proper value.

## 1.2.13 Checking Whether the sql\_mode Values of the Source and Destination Databases Are the Same

Check whether the **sql\_mode** values of the source and destination databases are the same. If they are different, the migration may fail. You are advised to change them to the same value.

## Failure Cause

The **sql\_mode** values of the source and destination databases must be the same.

## Handling Suggestion

Change the value of **sql\_mode** of the source database to be the same as that of the destination database.

- If the database is a self-managed MySQL database, run commands to change the value.  
SET sql\_mode='New value';
- If the database is an RDS for MySQL DB instance, change the value of the **sql\_mode** parameter by referring to [Modifying Parameters of an RDS for MySQL Instance](#).

If MyISAM tables are to be migrated, the **sql\_mode** values in the destination database cannot contain **no\_engine\_substitution**.

For details about how to change the value of **sql\_mode** for other types of databases, see the usage guide of the corresponding database.

## 1.2.14 Checking Whether the sql\_mode Value in the Destination Database Is Not no\_engine

### MySQL Migration and Synchronization

**Table 1-8** Checking whether the sql\_mode value in the destination database is not no\_engine

<b>Check Item</b>	Whether the sql_mode value in the destination database is not no_engine
<b>Description</b>	If the MyISAM tables are included in the migration objects, the <b>sql_mode</b> value in the destination database cannot be <b>no_engine_substitution</b> . Otherwise, the migration fails.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The <b>sql_mode</b> value in the destination database is <b>no_engine_substitution</b> . Handling suggestion: In the destination database, set <b>sql_mode</b> to a value other than no_engine_substitution. For details, see <a href="#">Modifying Parameters</a> in the <i>Relational Database Service User Guide</i> .

## 1.2.15 Checking Whether the innodb\_strict\_mode Values of the Source and Destination Databases Are the Same

Check whether the **innodb\_strict\_mode** values of the source and destination databases are the same. If they are different, the migration may fail. You are advised to change them to the same value.



## Failure Cause

The **innodb\_strict\_mode** values of the source and destination databases must be the same.

## Handling Suggestion

Change the value of **innodb\_strict\_mode** of the source database to be the same as that of the destination database.

- If the database is a self-managed MySQL database, run commands to change the value.
  - a. Run the following command to check the value:  

```
SHOW VARIABLES LIKE "innodb_strict_mode";
```
  - b. Run the following command to change the value:  

```
SET GLOBAL innodb_strict_mode = <value>;
```

To disable this option, set **<value>** to **0**. To enable this option, set **<value>** to **1**.
- If the database is an RDS for MySQL DB instance, change the value of the **innodb\_strict\_mode** parameter by referring to [Modifying Parameters of an RDS for MySQL Instance](#).

For details about how to change the value of **innodb\_strict\_mode** for other types of databases, see the usage guide of the corresponding database.

## 1.2.16 Checking Whether the max\_wal\_senders Value of the Source Database Is Correctly Configured

### PostgreSQL Synchronization

**Table 1-9** Checking whether the max\_wal\_senders value of the source database is correctly configured

<b>Check Item</b>	Whether the <b>max_wal_senders</b> value of the source database is correctly configured
<b>Description</b>	The <b>max_wal_senders</b> value of the source database must be greater than the number of used replication slots. Otherwise, the synchronization may fail.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The <b>max_wal_senders</b> value of the source database is less than or equal to the number of used replication slots. Handling suggestion: Set <b>max_wal_senders</b> to a value greater than the number of used replication slots and restart the database to apply the changes. Run the following command to query the number of used replication slots in the current database: <pre>select count(1) from pg_replication_slots;</pre>

## 1.2.17 Checking Whether the WAL\_LEVEL Value in the Source Database Is Correct

### PostgreSQL Synchronization

**Table 1-10** Checking whether the WAL\_LEVEL value in the source database is correct

<b>Check Item</b>	Whether the WAL_LEVEL value in the source database is correct
<b>Description</b>	Check whether <b>wal_level</b> of the source database is set to <b>logical</b> . If the value is not <b>logical</b> , the incremental logs of the source database cannot be logically decoded. As a result, incremental synchronization cannot be performed.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The <b>wal_level</b> value in the source database is incorrect. Handling suggestion: Change the <b>wal_level</b> value of the source database to <b>logical</b> . For details about how to modify the parameter for self-built databases, see: <ul style="list-style-type: none"><li>• Run <b>alter system set wal_level = logical</b> in the source database as a super user and restart the database to apply the changes.</li><li>• Alternatively, modify the <b>postgresql.conf</b> configuration file, set <b>wal_level</b> to <b>logical</b>, and restart the database to apply the changes.</li></ul>
	Failure cause: The source database version is not supported. Handling suggestion: Ensure that the source database version is supported by DRS. Supported source database versions include PostgreSQL 9.4, 9.5, 9.6, 10, 11, 12, and 13.
	Failure cause: The destination database version is not supported. Handling suggestion: Ensure that the destination database version is supported by DRS. The destination database supports the following major versions: RDS for PostgreSQL 9.5, 9.6, 10, 11, 12, and 13. If the source database is RDS for PostgreSQL Enhanced Edition, the destination database supports only RDS for PostgreSQL Enhanced Edition.

## 1.2.18 Checking Whether the MAX\_REPLICATION\_SLOTS Value in the Source Database Is Correct

### PostgreSQL Synchronization

**Table 1-11** Checking whether the MAX\_REPLICATION\_SLOTS value in the source database is correct

<b>Check Item</b>	Whether the <b>MAX_REPLICATION_SLOTS</b> value in the source database is correct
<b>Description</b>	The <b>max_replication_slots</b> value of the source database must be greater than the number of used replication slots. Otherwise, the synchronization may fail.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The <b>max_replication_slots</b> value of the source database is less than or equal to the number of used replication slots. Handling suggestion: Set <b>max_replication_slots</b> to a value greater than the number of used replication slots and restart the database to apply the changes. Run the following command to query the number of used replication slots in the current database: <pre>select count(1) from pg_replication_slots;</pre>
	Failure cause: Insufficient user permissions Handling suggestion: Check whether the database user permissions meet the synchronization requirements.

## 1.2.19 Checking Whether the Source Database Is on Standby

### PostgreSQL Synchronization

**Table 1-12** Checking whether the source database is on standby

<b>Check Item</b>	Whether the source database is on standby
<b>Description</b>	For a full+incremental synchronization task, the source database cannot be a standby database. Otherwise, incremental synchronization cannot be performed. For a full synchronization task, the source database can be a standby database, but <b>hot_standby_feedback</b> must be set to <b>on</b> . Otherwise, the synchronization may fail.

<b>Failure Cause and Handling Suggestion</b>	Failure cause: In a real-time full+incremental synchronization task, the source database cannot be a standby database. Otherwise, incremental synchronization cannot be performed. Handling suggestion: Configure the source database as the primary database.
	Failure cause: For a full synchronization task, the source database is a standby database, and <b>hot_standby_feedback</b> is set to <b>off</b> . Handling suggestion: Configure the source database as the primary database, or set <b>hot_standby_feedback</b> of the source database to <b>on</b> . <ul style="list-style-type: none"><li>• Change the source database to the primary database.</li><li>• Alternatively, change the <b>hot_standby_feedback</b> value of the source database to <b>on</b> before starting full synchronization. After the full synchronization is complete, change the value of this parameter to <b>off</b>.</li></ul>

## 1.2.20 Checking Whether the `log_slave_updates` Value in the Source Database Is Correct

To ensure that DRS can obtain all binlogs during MySQL migration, enable the `log_slave_updates` parameter.

### Failure Cause

The `log_slave_updates` parameter of the source database is set to **OFF**.

### Handling Suggestion

**Step 1** Log in to the server where the MySQL source database is located.

**Step 2** Run the following SQL statement to check whether the value of `log_slave_updates` is **ON**:

```
show variables like '%log_slave_updates%';
```

**Step 3** Add the following content followed by **[mysqld]** in the MySQL configuration file `my.cnf`:

```
log_slave_updates=1
```

**Step 4** After the modification, restart the source database during a non-service period.

----End

## 1.2.21 Checking Whether the **BLOCK\_SIZE** Value of the Source Database Is the Same as That of the Destination Database

### PostgreSQL Synchronization

**Table 1-13** Checking whether the **BLOCK\_SIZE** value of the source database is the same as that of the destination database

<b>Check Item</b>	Whether the <b>BLOCK_SIZE</b> value of the source database is the same as that of the destination database
<b>Description</b>	The <b>BLOCK_SIZE</b> value of the destination database must be greater than or equal to that of the source database. Otherwise, the synchronization may fail.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The <b>BLOCK_SIZE</b> value of the destination database is less than that of the source database. Handling suggestion: <ul style="list-style-type: none"><li>• Use the destination database whose <b>BLOCK_SIZE</b> value is greater than or equal to that of the source database.</li><li>• Use the source database whose <b>BLOCK_SIZE</b> value is less than or equal to the value of destination database <b>BLOCK_SIZE</b>.</li></ul>

## 1.2.22 Checking Whether the **binlog\_row\_image** Value is **FULL**

### MySQL

**Table 1-14** Checking whether the **binlog\_row\_image** value is **FULL**

<b>Check Item</b>	Whether the <b>binlog_row_image</b> value is <b>FULL</b>
<b>Description</b>	If the <b>binlog_row_image</b> value of the source database is not <b>FULL</b> , the migration will fail.

<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The <b>binlog_row_image</b> value of the source database is not <b>FULL</b>.</p> <p>Handling suggestion:</p> <ul style="list-style-type: none"><li>• If the source database is an RDS DB instance on the cloud, change <b>binlog_row_image</b> to <b>FULL</b> on the RDS console, and then restart the source database.</li><li>• If the source database is an on-premises database, perform the following steps:<ol style="list-style-type: none"><li>1. Log in to the server where the MySQL source database is located.</li><li>2. Manually change the value of <b>binlog_row_image</b> in the <b>my.cnf</b> configuration file to <b>FULL</b> and save the file.<pre>binlog_row_image=full</pre></li><li>3. To ensure a successful task, restart the source database during off-peak hours.</li></ol></li></ul>
--	---

## 1.2.23 Checking Whether the Transaction Isolation Levels are Consistent

### MySQL

**Table 1-15** Checking whether the transaction isolation levels are consistent

<b>Check Item</b>	Whether the transaction isolation levels are consistent
<b>Description</b>	Check whether the transaction isolation levels of the source and destination databases are the same.
<b>Failure Cause and Handling Suggestion</b>	<p>If you are migrating data to the cloud, perform the following operations:</p> <p><b>Failure cause:</b> The transaction isolation levels of the source and destination databases are different.</p> <p><b>Handling suggestion:</b> Change the isolation level (<b>tx_isolation</b> or <b>transaction_isolation</b>) of the destination database to be the same as that of the source database.</p>

## 1.2.24 Checking Whether the lc\_monetary Values of the Source and Destination Databases Are the Same

### PostgreSQL Synchronization

**Table 1-16** Checking whether the lc\_monetary values of the source and destination databases are the same

<b>Check Item</b>	Whether the <b>lc_monetary</b> values of the source and destination databases are the same
<b>Description</b>	Check whether the <b>lc_monetary</b> values of the source and destination databases are the same. If they are inconsistent, the synchronization fails.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: This item cannot be checked because the source database failed to be connected. Handling suggestion: Check whether the source database is connected.
	Failure cause: This item cannot be checked because the destination database failed to be connected. Handling suggestion: Check whether the destination database is connected.
	Failure cause: The <b>lc_monetary</b> values of the source and destination databases must be the same. Handling suggestion: Check whether the <b>lc_monetary</b> values of the source and destination databases meet the synchronization requirements.
	Failure cause: Insufficient user permissions Handling suggestion: Check whether the database user permissions meet the synchronization requirements.

## 1.2.25 Checking Whether the Source Database Contains Trigger Names with Non-ASCII Characters

### MySQL

**Table 1-17** Checking whether the source database contains trigger names with non-ASCII characters

<b>Check Item</b>	Whether the source database contains trigger names with non-ASCII characters
<b>Description</b>	If the source database contains non-ASCII characters, the migration will fail.

<b>Item to Be Confirmed and Handling Suggestion</b>	<p>Item to be confirmed: The source database cannot contain view names with non-ASCII characters.</p> <p>Handling suggestion: To solve this problem, perform the following steps:</p> <p>Method 1:</p> <p>Click <b>Previous</b> to return to the <b>Select Migration Type</b> page. Select a customized object and do not select the trigger name that contains non-ASCII characters.</p> <p>Method 2: Change the trigger name.</p>
---	---

## 1.2.26 Checking Whether the Source Database Collections Contain More Than 10 Indexes

### MongoDB Migration

**Table 1-18** Checking whether the source database collections contain more than 10 indexes

<b>Check Item</b>	Whether the source database collections contain more than 10 indexes
<b>Description</b>	If the number of indexes in the source database exceeds 10, the migration duration is affected.
<b>Failure Cause and Handling Suggestion</b>	<p>Alarm cause: The source database has collections containing more than 10 indexes, which are migrated slowly.</p> <p>Handling suggestion: The number of indexes affects the migration duration. Check whether all indexes need to be migrated. If the index does not need to be migrated, delete the index before starting the migration.</p> <p>Run the following command to delete the index:  <b>run the <code>db.Collection name.dropIndex(Index name)</code></b></p>

## 1.2.27 Checking Whether the Source Database Collections Contain TTL Indexes

### MongoDB Migration

**Table 1-19** Checking whether the source database collections contain TTL indexes

<b>Check Item</b>	Whether the source database collections contain TTL indexes
-------------------	---



<b>Description</b>	Due to inconsistency of time zones and clocks between source and destination databases, migrating TTL indexes will cause data inconsistency.
<b>Failure Cause and Handling Suggestion</b>	<p>Item to be confirmed: Due to inconsistency of time zones and clocks between source and destination databases, migrating TTL indexes will cause data inconsistency.</p> <p>Handling suggestion: If data consistency is required, delete TTL indexes. Alternatively, do not migrate the collections containing TTL indexes.</p> <p>Run the following command to delete the index:  <code>run the db.Collection name.dropIndex(Index name)</code></p>

## 1.2.28 Checking Whether log\_bin\_trust\_function\_creators Is Set to On in Both the Source and Destination Databases

### MySQL

**Table 1-20** Checking whether log\_bin\_trust\_function\_creators is set to on in both the source and destination databases

<b>Check Item</b>	Whether log_bin_trust_function_creators is set to on in both the source and destination databases
<b>Description</b>	During the out-of-cloud migration from MySQL to MySQL, the <b>log_bin_trust_function_creators</b> value of the source database must be the same as that of the destination database. If the source database supports user-defined functions (UDFs) but the destination database does not, change the <b>log_bin_trust_function_creators=off</b> parameter of the destination database to <b>log_bin_trust_function_creators=on</b> . If the parameters of the source and destination are different, the migration may fail.
<b>Item to Be Confirmed and Handling Suggestion</b>	<p>Item to be confirmed: The destination database does not support custom functions.</p> <p>Handling suggestions: In the <b>my.cnf</b> file of the destination database, check whether <b>log_bin_trust_function_creators=on</b> exists. If it does not exist, add <b>log_bin_trust_function_creators=on</b> and restart the database for the modification to take effect.</p>

## 1.2.29 Checking Whether GTID Is Enabled for the Source Database

During data migration, GTID must be enabled for the source database. If GTID is disabled for the source database and a primary/standby switchover is performed on the source database, the task may fail.

### Failure Cause

GTID is disabled on the source database.

### Handling Suggestion

- If the source database is an RDS for MySQL DB instance, GTID is enabled by default and cannot be disabled. If GTID is disabled, contact RDS for MySQL O&M personnel.
- If the source database version is MySQL 5.5, GTID cannot be set, and DRS cannot be used for synchronization and DR tasks. Create a migration task or contact O&M personnel.
- If the source database version is MySQL 5.6 and later, set the following parameters to enable GTID in the database configuration file. Then, restart the database for the modifications to take effect.

Parameters to be configured:

```
gtid_mode = on  
log_slave_updates = true  
enforce_gtid_consistency = on
```

## 1.2.30 Checking Whether GTID Is Enabled for the Destination Database

### MySQL Disaster Recovery

**Table 1-21** Checking whether GTID is enabled for the destination database

<b>Check Item</b>	Whether GTID is enabled for the destination database
<b>Description</b>	During disaster recovery, GTID should be enabled for the destination database. Otherwise, the migration fails.

<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: GTID is disabled for the destination database.</p> <p>Handling suggestion: Check that the destination database binlog is enabled. Modify the parameter settings configuration file as follows to enable the destination database GTID, and then restart the database</p> <p>Sample command:</p> <pre>gtid_mode = on log_slave_updates = true enforce_gtid_consistency = on</pre> <p>Sample command:</p> <pre>log-bin = mysql-bin binlog_gtid_simple_recovery = on</pre>
--	---

### 1.2.31 Checking Whether log\_bin\_trust\_function\_creators Is Set to On in the Destination Database

#### MySQL

**Table 1-22** Checking whether log\_bin\_trust\_function\_creators is set to on in the destination database

<b>Check Item</b>	Whether <b>log_bin_trust_function_creators</b> is set to <b>on</b> in the destination database
<b>Description</b>	During the migration from RDS for MySQL to MySQL out of the cloud, the destination database does not support custom functions.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The destination database does not support custom functions.</p> <p>Handling suggestions: In the <b>my.cnf</b> file of the destination database, check whether <b>log_bin_trust_function_creators=on</b> exists. If it does not exist, add <b>log_bin_trust_function_creators=on</b> and restart the database for the modification to take effect.</p>

### 1.2.32 Checking Whether the Values in the Source Oracle Database Are Out of the Ranges of the MySQL Database

#### Oracle Migration

**Table 1-23** Checking whether the values in the source Oracle database are out of the ranges of the MySQL database

<b>Check Item</b>	Whether the values in the source Oracle database are out of the ranges of the MySQL database
-------------------	--

<b>Description</b>	The values of the following data types in the source Oracle database are out of the ranges of the MySQL database, causing the migration failure.
<b>Failure Cause and Handling Suggestion</b>	<b>Alarm Information</b> <ol style="list-style-type: none"><li>1. Ensure that the primary key or unique key column cannot contain values of character string data types when you map the MySQL data types to the character data types in Oracle. Otherwise, data inconsistency or migration failure may occur.</li><li>2. Due to differences between Oracle and MySQL databases, the migration will fail if the values of the following data types in the source Oracle database are out of the ranges of MySQL database: number, intl, float, double, date, and timestamp.</li></ol> <p>Handling suggestion: Contact technical support.</p>

## 1.2.33 Checking Whether the max\_allowed\_packet Value of the Destination Database Is too Small

### MySQL Migration

**Table 1-24** Checking whether the max\_allowed\_packet value of the destination database is too small

<b>Check Item</b>	Whether the max_allowed_packet value of the destination database is too small
<b>Description</b>	A large amount of data cannot be written to the destination database during the migration because the max_allowed_packet value is smaller than 100 MB. As a result, the full migration failed.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The <b>max_allowed_packet</b> value of the destination database is too small, which may cause data fails to be written during the migration.  Handling suggestions: Set the <b>max_allowed_packet</b> value greater than 100 MB

## MariaDB Synchronization

**Table 1-25** Checking whether the `max_allowed_packet` value of the destination database is too small

<b>Check Item</b>	Whether the <code>max_allowed_packet</code> value of the destination database is too small
<b>Description</b>	A large amount of data cannot be written to the destination database during the migration because the <code>max_allowed_packet</code> value is smaller than 100 MB. As a result, the full migration failed.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The <b>max_allowed_packet</b> value of the destination database is too small, which may cause data fails to be written during the migration.</p> <p>Handling suggestions: Set the <b>max_allowed_packet</b> value greater than 100 MB</p>

### 1.2.34 Checking Whether the Source Database User Has the Permission to Parse Logs

#### Oracle -> MySQL Migration

**Table 1-26** Checking whether the source database user has the permission to parse logs

<b>Check Item</b>	Whether the source database user has the permission to parse logs
<b>Description</b>	If the source database user does not have the log parsing permission, the incremental migration will fail.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The source database user does not have the EXECUTE_CATALOG_ROLE role.</p> <p>Handling suggestion: Assign the required role to the user and perform the check again. Run the <b>GRANT EXECUTE_CATALOG_ROLE TO <i>UserName</i></b> command to assign the role.</p>
<b>Suggestion</b>	<p>Failure cause: The source database user does not have the log parsing permission.</p> <p>Handling suggestion: Assign the required role to the user and perform the check again. Run the <b>GRANT LOGMINING TO <i>UserName</i></b> command to grant the permission.</p>

## 1.2.35 Checking Whether the ExpireLogsDays Value Is 0

### MySQL Synchronization and Disaster Recovery

Table 1-27 Checking whether the expirelogsdays value is 0

<b>Check Item</b>	Whether the <b>expirelogsdays</b> value is 0
<b>Description</b>	If the <b>expire_logs_days</b> value of the source database is set to <b>0</b> , the migration may fail.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: If <b>expire_logs_days</b> is set to <b>0</b> in the source database, operations such as startup and flush logs will trigger binlog clearance and result in a migration failure. Handling suggestion: Set the binlog storage duration by running the following command on the client: <b>set global expire_logs_days=value</b> ( <i>value</i> indicates an integer greater than 0.)

## 1.2.36 Checking Whether the Source Database Character Set Is Supported

### Oracle Synchronization

Table 1-28 Checking whether the source database character set is supported

<b>Check Item</b>	Whether the source database character set is supported
<b>Description</b>	If the character set of the source database is not supported, data synchronization may fail. A migration task from Oracle only supports the following character sets: ZHS16GBK, AL32UTF8, UTF8, US7ASCII, WE8MSWIN1252, WE8ISO8859P1, WE8ISO8859P2, WE8ISO8859P4, WE8ISO8859P5, WE8ISO8859P7, WE8ISO8859P9, WE8ISO8859P13, WE8ISO8859P15.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The source database character set is not supported. Handling suggestion: Go back to the connection test page and select a source database with supported character sets or change the character set of the source database to AL32UTF8.

## 1.2.37 Checking Whether the Length Table and Field Names Is Not Supported

DDM -> Oracle Synchronization and MySQL -> Oracle Synchronization

**Table 1-29** Checking whether the length of table and field names is supported

<b>Check Item</b>	Whether the length of table and field names is supported
<b>Description</b>	The source database table name and field name cannot exceed 30 characters. Otherwise, the synchronization fails.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: There are tables or fields whose name lengths exceed 30 characters in the source database. Handling suggestion: Change the lengths of the table or field names fewer than 30 characters. Alternatively, deselect these tables from the synchronization objects.

## 1.2.38 Checking Whether the Databases and Tables Exist

All Scenarios

**Table 1-30** Checking whether the databases and tables exist

<b>Check Item</b>	Whether the databases and tables exist
<b>Description</b>	There are databases and tables in the uploaded file that do not exist in the source database. The synchronization fails.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: Objects imported from files do not exist in the source database. Handling suggestion: Remove these objects that do not exist and import the file again.

## Synchronization from PostgreSQL to GaussDB(DWS) and from PostgreSQL to GaussDB

**Table 1-31** Checking whether the databases and tables exist

<b>Check Item</b>	Whether the databases and tables exist
<b>Description</b>	The selected tables contain identifier columns, but the destination database does not support identifier columns. As a result, data synchronization fails.
<b>Failure Cause and Handling Suggestion</b>	<b>Failure cause:</b> The selected tables contain identifier columns. <b>Handling suggestion:</b> Deselect tables containing identifier columns.

### 1.2.39 Checking Whether the Supplemental Log Level of the Source Database Meets Requirements

#### Oracle Synchronization

**Table 1-32** Checking whether the supplemental log level of the source database meets requirements

<b>Check Item</b>	Whether the supplemental log level of the source database meets requirements
<b>Description</b>	The supplemental log level of the source Oracle database does not meet requirements. The synchronization fails.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The supplemental logging level of the source Oracle database does not meet requirements. Handling suggestion: Perform any of the following operations in the source database: <ul style="list-style-type: none"><li>• Enable all-level (database-level) supplemental logging: <b>alter database add supplemental log data (all) columns</b></li><li>• Enable minimal-level supplemental logging: alter database add supplemental log data. Then run the following command to enable all-level (table-level) supplemental logging for each to-be-synchronized table: <b>alter table TABLE_NAME add supplemental log data(all) columns</b></li></ul>



## 1.2.40 Checking Whether the Length of the Source Database Object Names Exceeds the Limit

### MySQL to PostgreSQL and MySQL to GaussDB(DWS) Synchronization

**Table 1-33** Checking whether the length of the source database object names exceeds the limit

<b>Check Item</b>	Whether the length of the source database object names exceeds the limit
<b>Description</b>	The destination database object name can contain a maximum of 63 characters. The length of source object names mapped to the destination ones exceeds the upper limit.
<b>Item to Be Confirmed and Handling Suggestion</b>	Failure cause: The destination database object name can contain a maximum of 63 characters. The length of source object names mapped to the destination ones exceeds the upper limit. Handling suggestion: <ol style="list-style-type: none"><li>1. Deselect the objects whose name length exceeds the limit.</li><li>2. If the objects are tables, change the table names to meet the mapping conditions.</li></ol>

### DB2 for LUW to GaussDB and DB2 for LUW to GaussDB(DWS) Synchronization

**Table 1-34** Whether the length of the source database object names exceeds the limit

<b>Check Item</b>	Whether the length of the source database object names exceeds the limit
<b>Description</b>	The destination database object name can contain a maximum of 63 characters. The length of source object names mapped to the destination ones exceeds the upper limit.
<b>Item to Be Confirmed and Handling Suggestion</b>	Failure cause: The destination database object name can contain a maximum of 63 characters. The length of source object names mapped to the destination ones exceeds the upper limit. Handling suggestion: <ol style="list-style-type: none"><li>1. Deselect the objects whose name length exceeds the limit.</li><li>2. If the objects are tables, change the table names to meet the mapping conditions.</li></ol>

## 1.2.41 Checking Whether session\_replication\_role of the Destination Database Is correctly Set

### PostgreSQL Synchronization

**Table 1-35** Checking whether the session\_replication\_role value of the destination database is correctly set

<b>Check Item</b>	Whether the <b>session_replication_role</b> value of the destination database is correctly set.
<b>Description</b>	The <b>session_replication_role</b> parameter of the destination database is not set to <b>replica</b> . Data synchronization may fail when the synchronized table has associated foreign key constraints or triggers.
<b>Item to Be Confirmed and Handling Suggestion</b>	Item to be confirmed: The <b>session_replication_role</b> parameter of the destination database is not set to <b>replica</b> . Handling suggestion: Before starting the synchronization task, set <b>session_replication_role</b> of the destination database to <b>replica</b> . After the synchronization is complete, change the value of this parameter to <b>origin</b> . If the destination database is an RDS instance, you can modify the parameter on the RDS console.

## 1.2.42 Checking the Database Compatibility Type

### MySQL -> GaussDB(DWS) Synchronization

**Table 1-36** Checking the database compatibility type

<b>Check Item</b>	Whether the database compatibility type is supported
<b>Description</b>	The migration of tables without primary keys and empty strings does not support the Oracle (ORA) compatibility mode of GaussDB(DWS).
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The migration of tables without primary keys does not support the ORA compatibility mode of GaussDB(DWS). Handling suggestion: Use the MySQL and Teradata compatibility mode of GaussDB(DWS) or add primary keys to tables that do not have primary keys. Run the following statement to add a primary key to the table: CREATE DATABASE mysql_compatible_db DBCOMPATIBILITY 'MYSQL'; Or: CREATE DATABASE td_compatible_db DBCOMPATIBILITY 'TD';

	<p><b>Failure cause:</b> The migration of empty strings does not support the ORA compatibility mode of GaussDB(DWS).</p> <p><b>Handling suggestion:</b> You are advised to use the MySQL or Teradata compatibility mode.</p>
--	--

## MySQL -> GaussDB Synchronization

**Table 1-37** Checking the database compatibility type

<b>Check Item</b>	Whether the database compatibility type is supported
<b>Description</b>	The destination database is incompatible with MySQL.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The destination database is incompatible with MySQL.</p> <p>Handling suggestion: Use a destination instance that is compatible with MySQL.</p> <p>If the destination instance is a distributed instance, run the following statement to create a compatible database: CREATE DATABASE mysql_compatible_db DBCOMPATIBILITY 'MYSQL';</p> <p>If the destination instance is a primary/standby instance, run the following statement to create a compatible database: CREATE DATABASE mysql_compatible_db DBCOMPATIBILITY 'B';</p>

### 1.2.43 Checking Whether the Collation of the Destination Database Is Correct

#### Oracle -> MySQL, Oracle -> GaussDB(for MySQL), and Oracle -> DDM Synchronization

**Table 1-38** Checking whether the collation of the destination database is correct

<b>Check Item</b>	Whether the collation of the destination database is correct
<b>Description</b>	The primary key or unique key in the destination database contains a collation ended in _ci. The collation ending in _ci is case insensitive, so an error indicating duplicate keys may be reported during synchronization and cause the synchronization to fail.

<b>Failure Cause and Handling Suggestion</b>	Failure cause: The destination database collation is not supported. Handling suggestion: Change the destination database collation to a case-sensitive collation (not ending with _ci).
--	--

## 1.2.44 Checking Whether the Index Name Is Supported

### MySQL -> CSS/ES Synchronization

**Table 1-39** Checking whether the index name is supported

<b>Check Item</b>	Whether the index name is supported
<b>Description</b>	Check whether the index name complies with the specifications. The index name rules are as follows: <ul style="list-style-type: none"> <li>• Uppercase letters are not allowed.</li> <li>• Cannot contain /*?"&lt;,&gt; # and spaces.</li> <li>• Can contain colons (:).</li> <li>• Cannot start with a hyphen (-), underscore (_), or plus sign (+).</li> <li>• Cannot contain periods (.) or ellipsis (...).</li> <li>• Can contain up to 255 characters.</li> </ul>
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The index name is not supported. Handling suggestion: In the migration settings, select another table for synchronization or change the index name for table mapping.

## 1.2.45 Checking Whether Tables Structures Are Consistent

### MySQL -> CSS/ES Synchronization

**Table 1-40** Checking whether tables structures are consistent

<b>Check Item</b>	Whether tables structures are consistent
-------------------	--

<b>Description</b>	Check whether the table structures at the source end and destination are consistent.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The source table columns are not a subset of the destination indexes.</p> <p>Handling suggestion: Return to the <b>Set Synchronization Task</b> page, select the tables that meet the requirements. Alternatively, create indexes required in the destination table.</p> <p>Statement for querying the destination database:                  GET /&lt;index&gt;/_mapping?include_type_name</p>

## MySQL -> GaussDB(DWS) Synchronization

**Table 1-41** Checking whether tables structures are consistent

<b>Check Item</b>	Whether tables structures are consistent
<b>Description</b>	Check whether the table structures at the source end and destination are consistent.
<b>Failure Cause and Handling Suggestion</b>	<p><b>Failure cause:</b> The destination table does not contain all columns of the source table or the not-null constraints of the columns are inconsistent.</p> <p>The possible causes are as follows:</p> <ol style="list-style-type: none"> <li>1. The number of columns in the destination table is less than that in the source table.</li> <li>2. If the source and destination databases are different, the column names in the source table are converted to lowercase letters for comparison with those in the destination table.</li> <li>3. Column names contain spaces or special characters.</li> </ol> <p>Handling suggestion: Return to the <b>Set Synchronization Task</b> page, select the tables that meet the requirements. Alternatively, modify columns required in the destination table. Ensure that the source table columns are a subset of the destination columns.</p>

## 1.2.46 Checking Whether Existing Data Meets the Constraints

### Oracle Synchronization

**Table 1-42** Checking whether existing data meets the constraints

<b>Check Item</b>	Whether existing data meets the constraints
-------------------	---

<b>Description</b>	The constraint type of the source database table does not check whether existing data meets constraints. Some data in the source table may not meet the constraints, and the destination database may not support the constraint types. As a result, data transmission fails or data is lost because the data written to the destination database fails the constraint check.
<b>Item to Be Confirmed and Handling Suggestion</b>	Item to be confirmed: Some check item constraints are not met. Handling suggestion: Ensure that all data in the corresponding table meets the current constraints. If any data does not meet the constraints, consider not configuring constraints in the destination (contact DRS online support to skip the table structure consistency check), or process the source data to ensure that the data meets the constraints.

## 1.2.47 Checking the Additional Column of the Destination Database

### MySQL to GaussDB(for MySQL) Synchronization

**Table 1-43** Checking the additional column of the destination database

<b>Check Item</b>	Additional column check
<b>Description</b>	Check whether the additional column is added to the additional column list of the destination database. If no additional column is added, the incremental synchronization task fails.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: Some columns are missing in the additional column processing table in the destination database. Handling suggestion: Add the missing additional columns to the destination database tables. Reference statement: <code>ALTER TABLE `database`.`table` ADD COLUMN `column_name` `column_definition`;</code>

## 1.2.48 Checking Whether Implicit Primary Key Check Is Enabled for the Primary and Standby Databases

### MySQL Disaster Recovery

**Table 1-44** Checking whether implicit primary key check is enabled for the primary and standby databases

<b>Check Item</b>	Whether implicit primary key check is enabled for the primary and standby databases
<b>Description</b>	Check whether <b>create_default_primary_key</b> is enabled for the source or destination database.
<b>Item to Be Confirmed and Handling Suggestion</b>	<p>Item to be confirmed: The <b>create_default_primary_key</b> parameter of the source or destination database is enabled. As a result, the primary/standby switchover may fail or data may be inconsistent. You are advised to disable <b>create_default_primary_key</b>. If this parameter is disabled, the task may still fail after a primary/standby switchover. The possible cause is that a table without a primary key is created when this parameter is enabled.</p> <p>Handling suggestion: This parameter was discarded, but in some DB instances of earlier versions, this parameter may be enabled. As a result, an error may occur during the primary/standby switchover, which may cause the task to fail or data inconsistency, you are advised to disable this parameter or ensure that the source database does not contain tables that do not have primary keys.</p>

## 1.2.49 Checking Whether the Source Table Replication Attribute Is Correct

### GaussDB->GaussDB, GaussDB->kafka, PostgreSQL->PostgreSQL, and PostgreSQL->Kafka Synchronization

**Table 1-45** Checking whether the source table replication attribute is correct

<b>Check Item</b>	Whether the source table replication attribute is correct
<b>Description</b>	The replica identity attribute of the source database table must be <b>FULL</b> .

<b>Item to Be Confirmed and Handling Suggestion</b>	<p>Item to be confirmed: The selected source database table contains the primary key column, but the replication attribute is not FULL.</p> <p>Handling suggestion: Change the replication attribute of the preceding tables to FULL. Run the following statement in the source database:  <code>alter table table_name replica identity full;</code></p>
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The source database contains tables whose replication attribute is not Full. As a result, incremental synchronization may fail.</p> <p>Handling suggestion: Change the replication attribute of the table to FULL. Run the following statement in the source database:  <code>alter table table_name replica identity full;</code></p>

## 1.2.50 Checking Whether the Source Database Is on Standby

### PostgreSQL->PostgreSQL and PostgreSQL->GaussDB Synchronization

**Table 1-46** Checking whether the source database is on standby

<b>Check Item</b>	Whether the source database is on standby
<b>Description</b>	If the source database is a standby database, set <b>hot_standby_feedback</b> in the source database to <b>on</b> .
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The source database is a standby database and the value of <b>hot_standby_feedback</b> in the source database is <b>off</b>. As a result, full synchronization may fail.</p> <p>Handling suggestion: Before full synchronization, set the <b>hot_standby_feedback</b> parameter of the source database to <b>on</b>. After the full synchronization is complete, change the parameter to the original value.</p>



## 1.2.51 Checking Whether the Specified Replication Slot Exists in the Source Database

### PostgreSQL->Kafka and GaussDB->Kafka Synchronization

**Table 1-47** Checking whether the specified replication slot exists in the source database

<b>Check Item</b>	Whether the specified replication slot exists in the source database
<b>Description</b>	The replication slot with the specified name is automatically created after the task is started and cannot be the same as an existing replication slot in the source database.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The specified replication slot already exists in source database. Handling suggestion: <ul style="list-style-type: none"><li>• Delete the replication slot from the source database.</li><li>• Alternatively, change the replication slot name to a name that does not exist in the source replication slots.</li></ul>

## 1.2.52 Checking Whether the MongoDB Instance Type Matches the Migration Mode

### MongoDB Migration

**Table 1-48** Checking whether the MongoDB instance type matches the migration mode

<b>Check Item</b>	Whether the MongoDB instance type matches the migration mode
<b>Description</b>	Check whether the MongoDB instance type matches the migration mode. If not, the migration fails.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: When a DRS task is created, the source DB instance type is set to <b>Cluster</b> , but the source database is not a cluster. Handling suggestion: If the source DB instance type is set to <b>Cluster</b> , ensure that the source database is a cluster database.

## 1.2.53 Checking the Physical Standby Database

### Oracle Synchronization

Table 1-49 Physical standby database check

<b>Check Item</b>	Physical standby database check
<b>Description</b>	When the source Oracle database is in the incremental phase, check whether the source database is a physical standby database.
<b>Item to Be Confirmed and Handling Suggestion</b>	Item to be confirmed: <ol style="list-style-type: none"><li>1. The physical standby database does not generate logs. It replicates them from the primary database. Check whether supplemental logging of the primary database meets the incremental synchronization requirements. Handling suggestion: Check <a href="#">Oracle supplemental logging</a>.</li><li>2. The physical standby database does not generate logs, resulting in synchronization task delay. You can shorten the interval for archiving logs from the primary database to the physical standby database. However, extremely low values can result in a large number of logs, so you are advised to synchronize data from the logical standby database. Run the following statement on the primary database to specify the log archive interval: <pre>alter system set archive_lag_target=seconds;</pre></li></ol>
	Item to be confirmed: The source database is a physical standby database, where data of the LOB type cannot be parsed. Handling suggestion: Change the Oracle startup mode and restart the Oracle database.

## 1.2.54 Checking Whether the Case Sensitivity of the Destination Database Is Configured

### Synchronization from TiDB to GaussDB(for MySQL)

Table 1-50 Case sensitivity check for the destination database

<b>Check Item</b>	Case sensitivity check for the destination database
<b>Description</b>	Destination database parameter <b>lower_case_table_names</b> check

<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The value of destination database <b>lower_case_table_names</b> is <b>1</b>, and the names of the selected databases or tables contain uppercase and lowercase letters.</p> <p>Handling suggestion: If the destination database is a self-built database, add <b>lower_case_table_names=0</b> under <b>[mysqld]</b> in the MySQL configuration file <b>my.cnf</b> and then restart the destination database.</p> <p>If the destination database is a cloud database, check whether the <b>lower_case_table_names</b> parameter can be modified. If it cannot be modified, contact customer service. If the destination database is an RDS database, you can modify the parameter on the RDS console.</p>
--	--

## 1.2.55 Checking Whether CDC is Enabled for Tables in the Source Database

### Microsoft SQL Server as the Source in Synchronization

Table 1-51 Whether CDC is enabled for tables in the source database

<b>Check Item</b>	Whether CDC is enabled for tables in the source database
<b>Description</b>	Whether CDC is enabled for tables in the source database
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The CDC function is not enabled for the table to be synchronized in the source database.</p> <p>Handling suggestion: Enable CDC for the preceding table in the source database by following the instructions provided in the <a href="#">official Microsoft SQL Server documentation</a>.</p>

## 1.2.56 Checking Whether the CDC Retention Period in the Source Database Is Long Enough

### Microsoft SQL Server as the Source in Synchronization

Table 1-52 Whether the CDC retention period in the source database is long enough

<b>Check Item</b>	Whether the CDC retention period in the source database is long enough
-------------------	--

<b>Description</b>	Check whether the CDC retention period in the source database is long enough.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The retention period of CDC data in the source database is less than one day. As a result, incremental synchronization is abnormal.</p> <p>Handling suggestion: Change the CDC retention period to 1440 minutes (one day) or longer. The recommended value is 4320 minutes (three days).</p> <p>For details, see the following statements:</p> <pre>EXECUTE sys.sp_cdc_change_job     @job_type = N'cleanup',     @retention = 4320;</pre>

## 1.2.57 Checking Whether the Source and Destination Databases Have Different Computer Names

### Microsoft SQL Server as the Source in Synchronization

**Table 1-53** Checking whether the source and destination databases have different computer names

<b>Check Item</b>	Whether the source and destination databases have different computer names
<b>Description</b>	The source and destination databases cannot have the same computer name.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: This item cannot be checked because the source database fails to be connected.</p> <p>Handling suggestion: Check whether the source database is connected.</p>
	<p>Failure cause: This item cannot be checked because the destination database failed to be connected.</p> <p>Handling suggestion: Check whether the destination database is connected.</p>
	<p>Failure cause: The source and destination databases cannot have the same computer name.</p> <p>Handling suggestion: Change the computer name of the source database and restart the computer for the modification to take effect.</p>

## 1.2.58 Checking Whether the Length of the Source Database Name Exceeds 64

### Microsoft SQL Server as the Source in Synchronization

Table 1-54 Checking whether the length of the source database name exceeds 64

<b>Check Item</b>	Whether the length of the source database name exceeds 64
<b>Description</b>	Check whether source database name contains more than 64 characters. If yes, the migration fails.
<b>Item to Be Confirmed and Handling Suggestion</b>	<p>Failure cause: The source database name cannot contain more than 64 characters.</p> <p>Handling suggestion: Change the length of the source database names.</p>

## 1.2.59 Synchronization Object Name Check

### Microsoft SQL Server as the Source in Synchronization

Table 1-55 Synchronization object name check

<b>Check Item</b>	Synchronization object name check
<b>Description</b>	If the source database contains database names, schema names, or table names that do not meet requirements, the migration will fail.
<b>Item to Be Confirmed and Handling Suggestion</b>	<p><b>Failure cause:</b> The source database contains database names, schema names, or table names that do not meet requirements. The database names, schema names, and table names in the source database can contain only letters, underscores (_), hyphens (-), and digits.</p> <p><b>Handling suggestion:</b> Change the object names that do not meet requirements.</p>

## 1.2.60 Checking Whether the Source Database Contains Disabled Clustered Indexes

### Microsoft SQL Server as the Source in Synchronization

**Table 1-56** Checking whether the source database contains disabled clustered indexes

<b>Check Item</b>	Whether the source database contains disabled clustered indexes
<b>Description</b>	If the source database contains disabled clustered indexes, the migration fails.
<b>Item to Be Confirmed and Handling Suggestion</b>	Item to be confirmed: The source database contains disabled clustered indexes. Handling suggestion: Run the following command to enable the clustered indexes: <code>ALTER INDEX [<i>Index name</i>] ON [<i>Table name</i>] REBUILD</code>

## 1.2.61 Checking Whether the Source Database Is Empty

### Microsoft SQL Server as the Source in Synchronization

**Table 1-57** Checking whether the source database is empty

<b>Check Item</b>	Whether the source database is empty
<b>Description</b>	The source database cannot be empty.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: This item cannot be checked because the source database failed to be connected. Handling suggestion: Check whether the source database is connected.
	Failure cause: The source database cannot be empty. Handling suggestion: Create a non-empty database on the source database.

## MongoDB Migration

**Table 1-58** Checking whether the source database is empty

<b>Check Item</b>	Whether the source database is empty
<b>Description</b>	The source database cannot be empty.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: This item cannot be checked because the source database failed to be connected. Handling suggestion: Check whether the source database is connected.
	Failure cause: The source database cannot be empty. Handling suggestion: Create a non-empty database on the source database.

## 1.2.62 Checking Whether the Source Database Uses the Full Recovery Model

### Microsoft SQL Server as the Source in Synchronization

**Table 1-59** Checking whether the source database uses the full recovery model

<b>Check Item</b>	Whether the source database uses the full recovery model
<b>Description</b>	Check whether the source database uses the full recovery model.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: A database does not use the full recovery model in the source database. Handling suggestion: Run the following SQL statements on each database that does not use the full recovery model: <pre>USE [master] GO ALTER DATABASE [<i>Database name</i>] SET RECOVERY FULL WITH NO_WAIT GO</pre>

## 1.2.63 Checking the Synchronization Objects

### Microsoft SQL Server as the Source in Synchronization

Table 1-60 Checking synchronization objects

<b>Check Item</b>	Checking synchronization objects
<b>Description</b>	If the selected synchronization object does not exist in the source database, the check fails.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The objects to be synchronized do not exist in the source database. Handling suggestion: Reselect the objects to be synchronized.
	Failure cause: More than 1,000 tables are configured to be synchronized in a single task. Handling suggestion: Deselect tables that do not need to be synchronized or split the task into multiple synchronization tasks for execution.

### Oracle Serving as the Source in Synchronization

Table 1-61 Checking synchronization objects

<b>Check Item</b>	Checking synchronization objects
<b>Description</b>	Check whether the objects selected from the source database meet requirements.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The objects to be synchronized do not exist in the source database. Handling suggestion: Deselect the objects to be synchronized.
	<b>Failure cause:</b> Due to the restrictions of Oracle Logminer, a schema name, table name, or column name of the selected table in the source database in the incremental synchronization phase cannot exceed 30 characters. <b>Handling suggestion:</b> Modify the schema name, table name, and column name that do not meet requirements to ensure that the length does not exceed 30 characters.



## MySQL > PostgreSQL

**Table 1-62** Checking synchronization objects

<b>Check Item</b>	Checking synchronization objects
<b>Description</b>	In the pre-check phase, the source database is disconnected when the synchronization object check is performed. As a result, the migration fails.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The source database is unavailable. Handling suggestions: Check that the source database is connected and try again later.

### 1.2.64 Checking Whether the Source Database Tables Contain Data Types Not Supported for Migration

#### Microsoft SQL Server as the Source in Synchronization

**Table 1-63** Checking whether the source database tables contain data types not supported for migration

<b>Check Item</b>	Whether the source database tables contain data types not supported for migration
<b>Description</b>	The source database tables cannot contain the <b>SQL_VARIANT</b> , <b>GEOMETRY</b> , and <b>GEOGRAPHY</b> data types. Otherwise, the synchronization task fails.
<b>Item to Be Confirmed and Handling Suggestion</b>	Failure cause: The source database tables contain data types that are not supported. Handling suggestions: Check whether the tables to be migrated in the source database meet the synchronization requirements.

## 1.2.65 Checking Whether the SQL Server Agent of the Source Database Is Enabled

### Microsoft SQL Server as the Source in Synchronization

**Table 1-64** Check whether the SQL Server Agent of the source database is enabled

<b>Check Item</b>	Whether the SQL Server Agent of the source database is enabled
<b>Description</b>	If the SQL Server Agent of the source database is not enabled, the migration will fail.
<b>Item to Be Confirmed and Handling Suggestion</b>	Failure cause: The SQL Server Agent of the source database is not enabled. Handling suggestions: Enable the SQL Server Agent for the source database.

## 1.2.66 Checking Whether the Values of `group_concat_max_len` Are Consistent

If the values of `group_concat_max_len` in the source and destination databases are different, the queried fields may be truncated. Change the parameter values to the same.

### Failure Cause

The `group_concat_max_len` values of the source and destination databases must be the same.

### Handling Suggestion

Change the value of `group_concat_max_len` of the source database to be the same as that of the destination database.

- If the database is a self-managed MySQL database, run commands to change the value.
  - a. Run the following command to check the value:  

```
SHOW VARIABLES LIKE "group_concat_max_len";
```
  - b. Run the following command to change the value:  

```
SET SESSION group_concat_max_len = <New maximum length>
```
- If the database is an RDS for MySQL DB instance, change the value of the `group_concat_max_len` parameter by referring to [Modifying Parameters of an RDS for MySQL Instance](#).

For details about how to change the value of **group\_concat\_max\_len** for other types of databases, see the usage guide of the corresponding database.

## 1.2.67 Checking Whether the table structures of the source database are consistent

### Synchronization from Oracle to GaussDB(DWS)

**Table 1-65** Checking whether the table structures of the source database are consistent

<b>Check Item</b>	Whether the table structures of the source database are consistent
<b>Description</b>	Synchronization failed because the source database table structures are inconsistent.
<b>Failure Cause and Handling Suggestion</b>	Details: A many-to-one synchronization cannot be performed because the structures of the tables in the source database are inconsistent. Handling suggestion: Modify the source table structures to ensure that they are consistent.

## 1.2.68 Checking Whether the Character Sets Are Compatible

### Oracle Synchronization

**Table 1-66** Character set compatibility check

<b>Check Item</b>	Character set compatibility check
<b>Description</b>	The character set of the destination database is incompatible with that of the source database.
<b>Item to Be Confirmed and Handling Suggestion</b>	Item to be confirmed: The character set of the destination database is incompatible with that of the source database. Handling suggestion: Change the character set of the destination database to be the same as that of the source database.

## 1.2.69 Whether There Are XA Transactions That Have Not Been Submitted for a Long Time in the Source Database

### DDM as the Source

**Table 1-67** Whether there are XA transactions that have not been submitted for a long time in the source database

<b>Check Item</b>	Whether there are XA transactions that have not been submitted for a long time in the source database
<b>Description</b>	There are XA transactions that have not been submitted for a long time in the source database. As a result, the data of these XA transactions may be missing.
<b>Item to Be Confirmed and Handling Suggestion</b>	Item to be confirmed: If there are XA transactions that have not been submitted for a long time in the source database. As a result, the data of these XA transactions may be missing. Handling suggestion: Check whether these XA transactions are correctly submitted.

## 1.2.70 Whether the Selected Objects Exist in the Destination Database

### Oracle -> GaussDB Synchronization

**Table 1-68** Whether the selected objects exist in the destination database

<b>Check Item</b>	Whether the selected objects exist in the destination database
<b>Description</b>	Check whether the destination database objects meet the synchronization requirements.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The selected schema does not exist in the destination database. Handling suggestion: Create the schema in the destination database. Statement for creating a schema: <code>CREATE SCHEMA schema_name;</code>

	<p><b>Failure cause:</b> Source database names are the same except for letter cases.</p> <p><b>Handling suggestion:</b> Change the table name or return to the object selection page and deselect the tables with the same name.</p> <p><b>Statement for changing the table name in the Oracle database:</b>                  ALTER TABLE old_table_name RENAME TO new_table_name;</p>
	<p><b>Failure cause:</b> The selected table does not exist in the destination database or the table structure is inconsistent with that in the source database.</p> <p><b>Handling suggestion:</b> Create a table in the destination database and ensure that the table structure is the same as that in the source database. <b>Statement for creating a table:</b>                  CREATE TABLE table_name (column_name data_type);</p>
	<p><b>Failure cause:</b> The destination table contains data, which may cause data conflict or inconsistency.</p> <p><b>Handling suggestion:</b> Run the following SQL statement to delete data from the table:                  TRUNCATE TABLE table_name;</p>
	<p><b>Failure cause:</b> The source and destination databases cannot contain same materialized view names.</p> <p><b>Handling suggestion:</b> Data operations cannot be directly performed on materialized views. Select base tables for data synchronization.</p>

## DB2 for LUW -> GaussDB Synchronization

**Table 1-69** Whether the selected objects exist in the destination database

<b>Check Item</b>	Whether the selected objects exist in the destination database
<b>Description</b>	Check whether the destination database objects meet the synchronization requirements.
<b>Failure Cause and Handling Suggestion</b>	<p><b>Failure cause:</b> The destination database table does not exist and cannot be synchronized.</p> <p><b>Handling suggestion:</b> If you do not synchronize the table structure, create the table to be synchronized in the destination database in advance. <b>Statement for creating a table in the destination database:</b>                  CREATE TABLE table_name (column_name data_type);</p>
	<p><b>Failure cause:</b> Source database names are the same except for letter cases.</p> <p><b>Handling suggestion:</b> Change the table name or return to the object selection page and deselect the tables with the same name.</p> <p><b>Statement for changing a table name in the DB2 for LUW database:</b>                  RENAME TABLE old_table_name to new_table_name;</p>

## 1.2.71 Destination Database Same Name Check

### MySQL to Oracle Synchronization

**Table 1-70** Destination database same name check

<b>Check Item</b>	Destination database same name check
<b>Description</b>	<p>For synchronization from MySQL to Oracle, check whether there is a database table with the same name in the destination database.</p> <ul style="list-style-type: none"> <li>• If <b>Table structure</b> is selected, the destination database cannot contain tables whose names are the same as the source tables to be synchronized.</li> <li>• If <b>Table structure</b> is not selected, the destination database must have tables that match the source tables, and the table structure must be the same as the selected source table structures.</li> </ul>
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The table structure synchronization is selected, and there are tables to be synchronized in the destination database.</p> <p>Handling suggestion: To synchronize table structures, delete existing tables from the destination database.</p>
	<p>Failure cause: The table structure synchronization is not selected, and there are no tables to be synchronized in the destination database.</p> <p>Handling suggestion: Select <b>Table structure</b> in the previous step or create the corresponding tables in the destination database.</p>
	<p>Failure cause: The source table columns are not a subset of the destination columns.</p> <p>Handling suggestion: Return to the <b>Set Synchronization Task</b> page, select the tables that meet the requirements. Alternatively, create columns required in the destination table.</p>
	<p>Failure cause: Some databases cannot be synchronized because the databases with the same names do not exist in the destination databases. For synchronization tasks, you need to create the corresponding database (user) in the destination database in advance.</p> <p>Handling suggestion: Create these databases or users in the destination database or do not synchronize these databases. Statement for creating a user:</p> <pre>CREATE USER user_name IDENTIFIED BY password;</pre>

## 1.2.72 Whether the Destination Database User (Schema) and Table Exist

### GaussDB -> Oracle Synchronization

**Table 1-71** Whether the destination database user (schema) and table exist

<b>Check Item</b>	Whether the destination database user (schema) and table exist
<b>Description</b>	To synchronize data from GaussDB to the Oracle database, you need to create the corresponding user (schema) and table in the destination database in advance.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The destination database user corresponding to the selected schema does not exist.</p> <p>Handling suggestion: Create a user with the same source database schema name in the destination database. Statement for creating a user:  <code>CREATE USER user_name IDENTIFIED BY password;</code></p>
	<p>Failure cause: Source database names are the same except for letter cases.</p> <p>Handling suggestion: Change the table name or return to the object selection page and deselect the tables with the same name. Statement for changing the table name:  <code>ALTER TABLE old_table_name RENAME TO new_table_name;</code></p>

## 1.2.73 Whether the Maximum Number of Indexed Columns Has Been Reached

### DB2 for LUW -> GaussDB Synchronization

**Table 1-72** Whether the maximum number of indexed columns has been reached

<b>Check Item</b>	Whether the maximum number of indexed columns has been reached
<b>Description</b>	The number of indexed columns to be migrated in the source database cannot exceed 32.

<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The number of indexed columns to be migrated in the source database cannot exceed 32.</p> <p>Handling suggestion: Check the number of indexed columns in the source table and modify the indexes.</p> <p>Statement for viewing indexed columns:                  SELECT T1.INDNAME ,T1.COLNAMES FROM SYSCAT.INDEXES AS T1 JOIN SYSCAT.INDEXCOLUSE AS T2 ON T1.INDNAME= T2.INDNAME WHERE T1.TABNAME='table_name' AND T1. INDNAME= 'index_name';</p> <p>Statement for deleting an index:                  DROP INDEX index_name;</p> <p>Statement for creating an index:                  CREATE INDEX index_name ON table_name(col1,col2);</p>
--	---

## 1.2.74 Checking the Length of the Index Column in the Source Database

### Oracle to MySQL Synchronization

**Table 1-73** Checking the length of the index column in the source database

<b>Check Item</b>	Checking the length of the index column in the source database
<b>Description</b>	Check whether the length of index column in the source database exceeds the limit.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The source database contains a data table with more than 64 indexes.</p> <p>Handling suggestion: Do not synchronize indexes or delete some indexes so that the number of indexes in a single table in the source database does not exceed 64.</p> <p>Statement for deleting an index:                  DROP INDEX index_name;</p> <p>Failure cause: There are indexes in the source database exceed the column length limit of the destination database.</p> <p>Handling suggestion: Deselect the table or change the index length.</p> <p>Statement for deleting an index:                  DROP INDEX index_name;</p> <p>Statement for creating an index:                  CREATE INDEX index_name ON table_name(col1,col2);</p>



## 1.2.75 Whether the Table Structures (Including Primary Key Indexes and the Number of Columns) of the Source Oracle Database and Destination Database Middleware Are Aligned

### Oracle to DDM Synchronization

**Table 1-74** Whether the table structures (including primary key indexes and the number of columns) of the source Oracle database and destination database middleware are aligned

<b>Check Item</b>	Whether the table structures (including primary key indexes and the number of columns) of the source Oracle database and destination database middleware are aligned
<b>Description</b>	Check whether the table structures (including primary key indexes and the number of columns) of the source Oracle database and destination DDM database are aligned.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The redundant columns (columns that do not exist in the source database) of the destination database cannot contain not-null constraints. The not-null constraints will cause the migration to fail.</p> <p>Handling suggestion:</p> <ol style="list-style-type: none"> <li>1. Check the not-null constraints on the redundant columns in the destination database. DESC [table_name];</li> <li>2. Modify the non-null constraint on the redundant columns. Alter Table table_name Modify column_name NULL;</li> </ol>
	<p>Failure cause: Source database names are the same except for letter cases.</p> <p>Handling suggestion: Change the table name or return to the object selection page and deselect the tables with the same name. Statement for changing the table name: ALTER TABLE old_table_name RENAME TO new_table_name;</p>

## 1.2.76 Whether Synchronization Objects Exist in the Destination Database

### GaussDB -> MySQL Synchronization

**Table 1-75** Whether synchronization objects exist in the destination database

<b>Check Item</b>	Whether synchronization objects exist in the destination database
<b>Description</b>	Check whether synchronization objects exist in the destination database.

<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: Source database names are the same except for the letter case.</p> <p>Handling suggestion: Change the table name or deselect the tables with the same name on the object selection page. Statement for changing the table name:                  ALTER TABLE old_table_name RENAME TO new_table_name;</p>
--	---

## 1.2.77 Whether the Source Database Contains Encrypted Objects

### Microsoft SQL Server as the Source

**Table 1-76** Whether the source database contains encrypted objects

<b>Check Item</b>	Whether the source database contains encrypted objects
<b>Description</b>	Check whether the source database contains encrypted objects.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The source database contains encrypted objects.</p> <p>Handling suggestion: Go back to the object selection page and select database objects that are not encrypted.</p>

## 1.2.78 Checking Whether the Source Database Contains Unsupported Table Field Types

### Oracle Synchronization

**Table 1-77** Checking whether the source database contains unsupported table field types

<b>Check Item</b>	Whether the source database contains unsupported table field types
<b>Description</b>	Check whether the source database contains unsupported table field types.

<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The source database contains unsupported table field types. The following table field types are supported: VARCHAR, VARCHAR2, NVARCHAR2, NUMBER, FLOAT, LONG, DATE, BINARY_FLOAT, BINARY_DOUBLE, RAW, LONG RAW, CHAR, NCHAR, CLOB, NCLOB, BLOB, ROWID, TIMESTAMP, TIMESTAMP WITH TIME ZONE and TIMESTAMP WITH LOCAL TIME ZONE.</p> <p>Handling suggestion: Select other tables that can be synchronized.</p>
--	--

## MySQL -> GaussDB(DWS)

**Table 1-78** Checking whether the source database contains unsupported table field types

<b>Check Item</b>	Whether the source database contains unsupported table field types
<b>Description</b>	Check whether the source database contains unsupported table field types.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The source database contains unsupported table field types. The following table field types are not supported: xml, geometry, point, lineString, polygon, geometrycollection, multipoint, multilinestring and multipolygon.</p> <p>Handling suggestion: Delete the columns containing the unsupported field types. Alternatively, do not synchronize the tables containing the unsupported table field types.</p>

## 1.2.79 Checking Replication Attribute of Primary Key Columns

### PostgreSQL as the Source

**Table 1-79** Replication attribute check of primary key columns

<b>Check Item</b>	Replication attribute check of primary key columns
<b>Description</b>	During a full+incremental synchronization or an incremental synchronization task, the replication attribute of primary key columns in the source database table is checked.

<b>Item to Be Confirmed and Handling Suggestion</b>	<p>Item to be confirmed: All primary key columns of the tables to be synchronized are columns whose storage attribute is plain, and the replication attribute of the tables is neither full nor default. Incremental synchronization may fail.</p> <p>Handling suggestion: Run the following SQL statement to change the replication attribute of the tables to default: alter table schema.table replica identity default;</p>
	<p>Item to be confirmed: The primary key columns of the tables to be synchronized contain columns whose storage attribute is not plain, and the replication attribute of the tables is neither full nor default. There is a high probability that incremental synchronization will fail.</p> <p>Handling suggestion: Run the following SQL statement to change the replication attribute of the tables to full: (If the replication attribute is changed to default, incremental synchronization may still fail.) alter table schema.table replica identity full;</p>
	<p>Item to be confirmed: The primary key columns of the tables to be synchronized contain columns whose storage attribute is not plain, and the replication attribute of the tables is not full. Incremental synchronization may fail.</p> <p>Handling suggestion: Run the following SQL statement to change the replication attribute of the tables to full: alter table schema.table replica identity full;</p>

## 1.2.80 Whether There Are Tables Containing Fields of the longtext or longblob Type in the Synchronization Object

### MySQL as the Source

**Table 1-80** Whether there are tables containing fields of the longtext or longblob type in the synchronization object

<b>Check Item</b>	Whether there are tables containing fields of the longtext or longblob type in the synchronization object
<b>Description</b>	If there are tables containing fields of the longtext or longblob type in the synchronization object, DRS tasks with small specifications may fail.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: There are tables containing fields of the longtext or longblob type in the synchronization object.</p> <p>Handling suggestion: If tables containing fields of the longtext or longblob type exist in the synchronization object, create a DRS task with large specifications.</p>

## MariaDB Synchronization

**Table 1-81** Checking whether there are tables containing fields of the longtext or longblob type in the synchronization object

<b>Check Item</b>	Whether there are tables containing fields of the longtext or longblob type in the synchronization object
<b>Description</b>	If there are tables containing fields of the longtext or longblob type in the synchronization object, DRS tasks with small specifications may fail.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: There are tables containing fields of the longtext or longblob type in the synchronization object. Handling suggestion: If tables containing fields of the longtext or longblob type exist in the synchronization object, create a DRS task with large specifications.

### 1.2.81 Checking Database Mapping Objects

#### MySQL to MySQL, MySQL to Gauss(for MySQL) and Gauss(for MySQL) to MySQL Synchronization

**Table 1-82** Checking database mapping objects

<b>Check Item</b>	Whether the database mapping objects are supported
<b>Description</b>	Whether the database mapping objects are supported
<b>Item to Be Confirmed and Handling Suggestion</b>	Item to be confirmed: The source database contains objects that cannot be synchronized. Handling suggestion: After database mapping, the source database contains data types that cannot be synchronized, including functions, stored procedures, and views. All the preceding objects cannot be synchronized during database mapping. Check whether the objects to be synchronized meet service requirements.

## 1.2.82 Whether the Source Database Is the Standby Database of a GaussDB(for MySQL) Instance

If the source database is a standby GaussDB(for MySQL) database and there is no binlogs, the incremental migration will fail.

### Failure Cause

The source database is a standby database and does not contain binlogs.

### Handling Suggestion

Use a primary GaussDB(for MySQL) database as the source database, and perform the pre-check again.

## 1.2.83 Checking Whether Type Names Mapped to the Destination Database Are Valid

### MySQL to CSS/ES and GaussDB(for MySQL) to CSS/ES Synchronization

**Table 1-83** Checking whether type names mapped to the destination database are valid

<b>Check Item</b>	Whether type names mapped to the destination database are valid
<b>Description</b>	The type mapped to the destination end must meet the following requirements: <ul style="list-style-type: none"><li>• If the destination version is 5.x, an index can support multiple types.</li><li>• If the destination version is 6.x or later, one index supports only one type.</li></ul>
<b>Item to Be Confirmed and Handling Suggestion</b>	Failure cause: In Elasticsearch 6.x, one index supports only one type. The recommended type name is <b>_doc</b> (In this way, APIs are compatible with 7.x or later). In Elasticsearch 7.x, <b>include_type_name</b> is set to <b>false</b> , and the index API formats are PUT <code>/_{index}/_doc/{id}</code> and POST <code>_{index}/_doc</code> . Handling suggestion: Return to the page for setting the synchronization task, edit index names and type names of the selected synchronization objects to ensure that one index supports only one type.

## 1.2.84 Checking Whether the Source and Destination Database Character Sets Are Consistent

### MySQL->MySQL/DDM Migration

**Table 1-84** Checking whether the source and destination database character sets are consistent

<b>Check Item</b>	Whether the source and destination database character sets are consistent
<b>Description</b>	Checking whether the source and destination database character sets are consistent
<b>Item to Be Confirmed and Handling Suggestion</b>	Item to be confirmed: The source database supports character sets of a later version. Handling suggestion: The source database supports character sets of a later version. Check whether the source database uses a character set of a later version and whether the destination database supports the character set.

### DDM -> MySQL/DDM Synchronization

**Table 1-85** Checking whether the source and destination database character sets are consistent

<b>Check Item</b>	Whether the source and destination database character sets are consistent
<b>Description</b>	Checking whether the source and destination database character sets are consistent
<b>Item to Be Confirmed and Handling Suggestion</b>	Item to be confirmed: The source database supports character sets of a later version. Handling suggestion: The source database supports character sets of a later version. Check whether the source database uses a character set of a later version and whether the destination database supports the character set.

## MariaDB Synchronization

**Table 1-86** Checking whether the source and destination database character sets are consistent

<b>Check Item</b>	Whether the source and destination database character sets are consistent
<b>Description</b>	Checking whether the source and destination database character sets are consistent
<b>Item to Be Confirmed and Handling Suggestion</b>	Item to be confirmed: The source database supports character sets of a later version. Handling suggestion: The source database supports character sets of a later version. Check whether the source database uses a character set of a later version and whether the destination database supports the character set.

### 1.2.85 Checking Whether Data Replication Is Enabled for the Source Database

#### Redis Migration

**Table 1-87** Checking whether data replication is enabled for the source database

<b>Check Item</b>	Whether data replication is enabled for the source database
<b>Description</b>	Checking whether data replication is enabled for the source database
<b>Failure Cause and Handling Suggestion</b>	Failure cause: Data replication is not enabled on the source database. Handling suggestion: Connect to the source database and set <b>enable-replication</b> to <b>1</b> to enable data replication.



## 1.2.86 Checking Whether the Maximum Sequence Number of the Source Database is Less Than That of the Destination Database

### Redis Migration

**Table 1-88** Checking whether the maximum sequence number of the source database is less than that of the destination database

<b>Check Item</b>	Whether the maximum sequence number of the source database is less than that of the destination database
<b>Description</b>	Checking whether the maximum sequence number of the source database is less than that of the destination database
<b>Item to Be Confirmed and Handling Suggestion</b>	Item to Be Confirmed: The maximum sequence number of the source database to be migrated is greater than that of the destination database. If incremental data is generated in the database with the maximum sequence number, the migration task will fail. Handling suggestion: Increase the maximum sequence number of the destination database.

## 1.2.87 Checking Whether Interval Partitioned Tables Are Included in the Source Database

### GaussDB Serving as the Source in Synchronization

**Table 1-89** Checking whether interval partitioned tables are included in the source database

<b>Check Item</b>	Whether interval partitioned tables are included in the source database
<b>Description</b>	Checking whether interval partitioned tables are included in the source database
<b>Failure Cause and Handling Suggestion</b>	Failure cause: Interval partitioned tables cannot be synchronized. Handling suggestion: Go back to the <b>Synchronization Object</b> area and deselect the interval partitioned tables.

## 1.2.88 Oracle Account Check in the Source Database

### Oracle Synchronization

**Table 1-90** Oracle account check in the source database

<b>Check Item</b>	Oracle account check in the source database
<b>Description</b>	Check the source database account when Oracle is the source database in the incremental synchronization.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The source database account is an Oracle account, but not a user account. Incremental synchronization cannot be performed. Handling suggestion: Use a user account instead of the Oracle account.

## 1.2.89 Checking the Number of DNs in the Source Database

### Self-built Distributed GaussDB Serving as the Source in Synchronization

**Table 1-91** Checking the number of DNs in the source database

<b>Check Item</b>	Checking the Number of DNs in the Source Database
<b>Description</b>	Check whether the number of DNs entered by the user is the same as that in the source database.
<b>Item to Be Confirmed and Handling Suggestion</b>	Item to be confirmed: The number of DNs entered by the user is different from that in the source database. Handling suggestion: If the number of DNs entered by the user is different from that in the source database, there may be data inconsistencies. You must confirm the risky item before you continue.

## 1.2.90 Whether the Selected Objects Exist in the Destination Database

### Oracle -> GaussDB Synchronization

**Table 1-92** Whether the selected objects exist in the destination database

<b>Check Item</b>	Whether the selected objects exist in the destination database
<b>Description</b>	Check whether the objects in the source database are consistent with those in the destination database.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: Object names will be converted to lowercase letters after being synchronized to the destination database. To avoid synchronization failures, ensure that the selected source database tables do not contain columns with the same name but different letter cases.</p> <p>Handling suggestion: Delete columns with the same name but different letter cases from the source database tables, or change the names of columns with the same name but different letter cases in the source database tables.</p>
	<p>Item to be confirmed: The selected table does not exist in the destination database or the table structure is inconsistent with that in the source database.</p> <p>Handling suggestion: If the selected table does not exist in the destination database, create the table in the destination database and ensure that the table structure is the same as that in the source database. Statement for creating a table:  <code>CREATE TABLE table_name (column_name data_type);</code></p> <p>If the table structure is inconsistent with that in the source database, create missing columns in the destination database table, convert the names of columns with the same name but different letter cases in the destination database table to lowercase letters, or delete redundant columns from the source database table.</p>
<b>Item to Be Confirmed and Handling Suggestion</b>	<p>Item to be confirmed: The constraints on the destination database are inconsistent with those on the source database. The synchronization may fail due to inconsistent constraints.</p> <p>Handling suggestion: Ensure that the constraints on the destination database are consistent with those on the source database, or confirm that inconsistent constraints do not adversely affect data migration.</p>

## 1.2.91 Whether There Are Foreign Keys Containing Unsupported Reference Operations in the Source Database

### MySQL and GaussDB(for MySQL) Serving as the Source in Full+Incremental or Incremental Migration and Synchronization, MySQL and GaussDB(for MySQL) Serving as the Source in DR

**Table 1-93** Whether there are foreign keys containing unsupported reference operations in the source database

<b>Check Item</b>	Whether there are foreign keys containing unsupported reference operations in the source database
<b>Description</b>	In a synchronization object, there are foreign keys that contain reference operations such as CASCADE, SET NULL, and SET DEFAULT. These operations will cause the update or deletion of rows in parent tables and affect records in child tables. Also, operations related to child tables are not recorded in binlogs. The DRS cannot synchronize data, and data in child tables is inconsistent.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: In a synchronization object, there are foreign keys that contain reference operations such as CASCADE, SET NULL, and SET DEFAULT. These operations will cause the update or deletion of rows in parent tables and affect records in child tables. Also, operations related to child tables are not recorded in binlogs. The DRS cannot synchronize data, and data in child tables is inconsistent.</p> <p>Handling suggestion: Delete foreign keys that contain reference operations such as CASCADE, SET NULL, and SET DEFAULT from child tables, or do not synchronize these child tables.</p> <p>Reference statement for deleting a foreign key: ALTER TABLE table_name DROP FOREIGN KEY foreign_key_name</p>

## 1.2.92 Whether the Selected Table Contains Delay Constraints

### PostgreSQL Serving as the Source in Synchronization

**Table 1-94** Whether the selected table contains delay constraints

<b>Check Item</b>	Whether the selected table contains delay constraints
<b>Description</b>	Tables that contain delay constraints may fail to be synchronized.

<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: Tables that contain delay constraints may fail to be synchronized.</p> <p>Handling suggestion: Remove the delay constraints.</p> <ul style="list-style-type: none"> <li>SQL statement for deleting a constraint:                      alter table Schema_name.Table_name drop CONSTRAINT Constraint_name</li> <li>SQL statement for adding a constraint:                      alter table Schema_name.Table_name add CONSTRAINT Constraint_name                      Constraint_type (Field list) NOT DEFERRABLE</li> </ul>
--	---

## 1.2.93 Whether the Source Database Tables Contain Primary Keys

### MySQL as the Source

**Table 1-95** Checking whether the source database tables contain primary keys

<b>Check Item</b>	Whether the source database tables contain primary keys
<b>Description</b>	The tables to be synchronized in the source database do not contain primary keys.
<b>Item to Be Confirmed and Handling Suggestion</b>	<p><b>Item to be confirmed:</b> The tables to be synchronized in the source database do not contain primary keys.</p> <p><b>Handling suggestion:</b> Create primary keys for the tables as the performance of a table without a primary key is lower than that of a table with a primary key.</p>

## 1.2.94 Whether Foreign Keys Are Disabled or Tables to Be Synchronized Have Foreign Keys in the Destination Database

### Oracle -> GaussDB Synchronization

**Table 1-96** Checking whether foreign keys are disabled or tables to be synchronized have foreign keys in the destination database

<b>Check Item</b>	Whether foreign keys are disabled or tables to be synchronized have foreign keys in the destination database
-------------------	--

<b>Descri ption</b>	Check whether foreign keys are disabled or tables to be synchronized have foreign keys in the destination database.
<b>Failur e Cause and Handl ing Sugge stion</b>	<b>Failure cause:</b> Tables to be synchronized have foreign keys in the destination database and cannot be synchronized. <b>Handling suggestion:</b> Delete the foreign key, disable the trigger, or change the value of <b>session_replication_role</b> to <b>replica</b> .

## 1.2.95 Whether There Are Composite Hash Indexes in the Source Collection

### Migration and Synchronization from MongoDB to DDS

**Table 1-97** Checking whether there are composite hash indexes in the source collection

<b>Check Item</b>	Whether there are composite hash indexes in the source collection
<b>Descri ption</b>	There are collections containing composite hash indexes in the source database.
<b>Failur e Cause and Handl ing Sugge stion</b>	<b>Failure cause:</b> The selected collections contain composite hash indexes. <b>Handling suggestion:</b> Deselect the preceding collections or create non-composite hash indexes in the source database.

## 1.2.96 Whether There Are Composite Hash Shard Keys in the Source Collection

### Migration and Synchronization from MongoDB to DDS

**Table 1-98** Checking whether there are composite hash shard keys in the source collection

<b>Check Item</b>	Whether there are composite hash shard keys in the source collection
<b>Description</b>	There are collections containing composite hash shard keys in the source database.
<b>Failure Cause and Handling Suggestion</b>	<b>Failure cause:</b> The selected collections contain composite hash shard keys. <b>Handling suggestion:</b> Deselect the preceding collections.

## 1.2.97 Whether the Source Table Structure Contains Newline Characters

The source databases, tables, columns, indexes, or constraints contain newline characters, which may cause service problems.

### Failure Cause

The source databases, tables, columns, indexes, or constraints contain newline characters.

### Handling Suggestion

The source databases, tables, columns, indexes, or constraints contain newline characters, which may cause service problems. Modify the objects in the source database and perform the pre-check again.

## 1.2.98 Whether There Are Tables Containing Fields of the bytea or text Type in the Synchronization Object

### PostgreSQL Serving as the Source in Synchronization

**Table 1-99** Checking whether there are tables containing fields of the bytea or text type in the synchronization object

<b>Check Item</b>	Whether there are tables containing fields of the bytea or text type in the synchronization object
<b>Description</b>	Fields of the bytea or text type may result in out of memory (OOM) during synchronization.
<b>Item to Be Confirmed and Handling Suggestion</b>	<p><b>Potential problem:</b> If there are tables containing fields of the bytea or text type in the synchronization object, fields of the bytea or text type may result in task OOM during synchronization.</p> <p><b>Handling suggestion:</b> If there are tables containing fields of the bytea or text type in the synchronization object, create a DRS task with large specifications for synchronization.</p>

## 1.2.99 Whether the Source Table Structure Contains Virtual Columns

### Oracle Serving as the Source in Synchronization

**Table 1-100** Checking whether the source table structure contains virtual columns

<b>Check Item</b>	Whether the source table structure contains virtual columns
<b>Description</b>	The source database contains virtual columns, but data in virtual columns cannot be migrated. As a result, the migrated data is incomplete.
<b>Item to Be Confirmed and Handling Suggestion</b>	<p><b>Item to be confirmed:</b> The source database contains virtual columns, but data in virtual columns cannot be migrated. As a result, the migrated data is incomplete.</p> <p><b>Handling suggestion:</b> After the pre-check, create a table structure that contains virtual columns in the destination database.</p>



## 1.2.100 Whether the max\_allowed\_packet Value of the Source Database Is Too Small

### MySQL and GaussDB(for MySQL) Serving as the Source

**Table 1-101** Checking whether the max\_allowed\_packet value of the source database is too small

<b>Check Item</b>	Whether the max_allowed_packet value of the source database is too small
<b>Description</b>	If the max_allowed_packet value of the source database is too small, data migration may fail.
<b>Item to Be Confirmed and Handling Suggestion</b>	<b>Potential problem:</b> If there is a lot of data to be migrated or there are too many fields to be migrated, and the max_allowed_packet value of the source database is too small, then the migration task may fail. <b>Handling suggestion:</b> Change the max_allowed_packet value of the source database to a value greater than 16777216.

## 1.2.101 Whether the Supplemental Log Level in the Source Database Is Correct

### Oracle to Kafka Synchronization

**Table 1-102** Checking whether the supplemental log level in the source database is correct

<b>Check Item</b>	Whether the supplemental log level in the source database is correct
<b>Description</b>	If all data is required for the synchronization object, all-level supplemental logging must be enabled.
<b>Failure Cause and Handling Suggestion</b>	<b>Failure cause:</b> All data is required for the synchronization object, but all-level supplemental logging is not enabled. <b>Handling suggestion:</b> Enable all-level supplemental logging. Commands for enabling all-level supplemental logging: Database level: alter database add supplemental log data (all) columns Table level: alter table <i>Schema_name.Table_name</i> add supplemental log data(all) columns

## 1.2.102 Whether Kafka Topics Have Been Created

### GaussDB to Kafka Synchronization

**Table 1-103** Checking whether Kafka topics have been created

<b>Check Item</b>	Whether Kafka topics have been created
<b>Description</b>	If you do not create topics in Kafka, DRS automatically creates them during incremental synchronization.
<b>Item to Be Confirmed and Handling Suggestion</b>	<b>Item to be confirmed:</b> If topics were not created in Kafka, DRS automatically creates them during incremental synchronization. <b>Handling suggestion:</b> Create the required topics in Kafka.

## 1.2.103 Whether the Source Database Kernel Encoding Supports Data Replication

### Out-of-Cloud GeminiDB Redis Migration

**Table 1-104** Checking whether the source database kernel encoding supports data replication

<b>Check Item</b>	Whether the source database kernel encoding supports data replication
<b>Description</b>	Check whether the source database kernel encoding supports data replication.
<b>Failure Cause and Handling Suggestion</b>	<b>Failure cause:</b> The source database kernel encoding does not support data replication. <b>Handling suggestion:</b> Contact GeminiDB Redis personnel.

## 1.2.104 block\_encryption\_mode Consistency Check

MySQL -> MySQL and MySQL -> GaussDB(for MySQL) Migration, Synchronization, and DR GaussDB(for MySQL) -> GaussDB(for MySQL) Synchronization and DR, GaussDB(for MySQL) -> MySQL and DDM->MySQL Synchronization, DDM-> DDM DR

**Table 1-105** Checking whether the block\_encryption\_mode values of the source and destination databases are the same

<b>Check Item</b>	block_encryption_mode consistency check
<b>Description</b>	The <b>block_encryption_mode</b> values of the source and destination databases must be the same. Otherwise, the destination database will be unavailable after DRS completes data migration.
<b>Failure Cause and Handling Suggestion</b>	<b>Failure cause:</b> The <b>block_encryption_mode</b> values of the source and destination databases must be the same. <b>Handling suggestion:</b> Change the value of <b>block_encryption_mode</b> in the destination database to be the same as that in the source database.

## 1.2.105 Character Type and Sorting Rule Check in the Destination Database

PostgreSQL -> PostgreSQL Synchronization

**Table 1-106** Checking the character type and sorting rule in the destination database

<b>Check Item</b>	Character type and sorting rule check in the destination database
<b>Description</b>	Check whether the destination database supports the value of <b>lc_ctype</b> or <b>lc_collate</b> in the database to be synchronized.

<b>Item to Be Confirmed and Handling Suggestion</b>	<p><b>Item to be confirmed:</b> The destination database does not support the value of <b>lc_ctype</b> or <b>lc_collate</b> in the database to be synchronized.</p> <p><b>Handling suggestion:</b> Check whether the parameter <b>lc_ctype</b> or <b>lc_collate</b> can be set to the default value when a database is created in the destination database during full synchronization. The value of <b>lc_collate</b> affects the sorting rule of character strings, and the value of <b>lc_ctype</b> affects character type and conversion.</p>
---	---

## 1.2.106 Column Name Check in the Source Database

### Microsoft SQL Server as the Source in Synchronization

Table 1-107 Column name check in the source database

<b>Check Item</b>	Column name check in the source database
<b>Description</b>	Check whether the column names in the source database contain special characters.
<b>Failure Cause and Handling Suggestion</b>	<p><b>Failure cause:</b> The source database contains column names that do not meet requirements. The column names cannot contain the following special characters: []?</p> <p><b>Handling suggestion:</b> Ensure that column names meet requirements.</p>

## 1.2.107 Whether the Destination Schemas and Table Objects Are Consistent

### Synchronization from GaussDB to GaussDB(DWS)

Table 1-108 Whether the destination schemas and table objects are consistent

<b>Check Item</b>	Whether the destination schemas and table objects are consistent
<b>Description</b>	Check whether the table structure objects in the destination database are the same as those in the source database.

<b>Failure Cause and Handling Suggestion</b>	<p><b>Failure cause:</b> The selected table does not exist in the destination database or the table structure is inconsistent with that in the source database.</p> <p><b>Handling suggestion:</b> If the selected table does not exist in the destination database, create a table in the destination database and ensure that the table structure in the destination database is the same as that in the source database.</p> <p>Statement for creating a table:                  CREATE TABLE table_name(Column_name data_type);</p> <p>If the table structure is inconsistent with that in the source database, create missing columns in the destination database table, convert the names of columns with the same name but different letter cases in the destination database table to lowercase letters, or delete redundant columns from the source database table.</p>
	<p><b>Item to be confirmed:</b> Primary key columns of the source tables are inconsistent with those of the destination tables. If the primary key columns are inconsistent, the synchronization may fail or data may be inconsistent.</p> <p><b>Handling suggestion:</b> Change primary key columns of the destination tables to be consistent with those of the source tables.</p> <p>Reference command:                  alter table table_name add constraint pkey name primary key(c1, c2);</p> <p><b>table_name</b> indicates the table name, <b>pkey_name</b> indicates the primary key name, and <b>c1</b> and <b>c2</b> indicate the primary key columns.</p>

## 1.2.108 Source Encrypted Table Check

### MySQL and GaussDB(for MySQL) Serving as the Source

Table 1-109 Source encrypted table check

<b>Check Item</b>	Source encrypted table check
<b>Description</b>	Check whether there are encrypted tables in the source database.
<b>Failure Cause and Handling Suggestion</b>	<p><b>Item to be confirmed:</b> The source database contains encrypted tables. Check whether the destination database supports encrypted tables. If the destination database does not support these tables, the task may fail.</p> <p><b>Handling suggestion:</b> Check whether the destination database supports these tables. If not, deselect them from the synchronization objects.</p>

## 1.2.109 Checking Whether the Source Table Replication Attribute Is Correct

### Synchronization from PostgreSQL to GaussDB and from PostgreSQL to GaussDB(DWS)

Table 1-110 Checking whether the source table replication attribute is correct

<b>Check Item</b>	Whether the source table replication attribute is correct
<b>Description</b>	Check the replication attribute of the table in the source database.
<b>Failure Cause and Handling Suggestion</b>	<p><b>Item to be confirmed:</b> The source database table contains the primary key column, but the replication attribute is not FULL. When the source table data is updated, if the replication attribute of the table is not FULL, the source database logs may not record the old values of all columns, causing data loss.</p> <p><b>Handling suggestion:</b> Run the following statements to change the replication attribute of the preceding tables to FULL:</p> <pre>alter table sch1.t varchar replica identity full; alter table sch1.t char replica identity full;</pre>

## 1.2.110 Partition Table Check on the Source Database

### Synchronization from PostgreSQL to GaussDB and from PostgreSQL to GaussDB(DWS)

Table 1-111 Partition table check on the source database

<b>Check Item</b>	Partition table check on the source database
<b>Description</b>	Check whether the source database has partition tables.
<b>Failure Cause and Handling Suggestion</b>	<p><b>Item to be confirmed:</b> If the source database contains partition tables and source partitions are modified or are deleted during data synchronization, partition data may fail to be synchronized or the synchronization may fail.</p> <p><b>Handling suggestion:</b> The source partition must be a newly-created partition, and the partition name must be unique. This partition can be deleted from the source database only after all data is synchronized to the destination database.</p>

## 1.3 Source DB Instance Statuses

### 1.3.1 Checking Whether the Source DB Instance Is Available

In the pre-check phase, DRS checks the status of the source DB instance.

#### Failure Cause

The source DB instance is unavailable.

#### Handling Suggestion

If the source DB instance is abnormal and cannot be accessed by DRS, wait until the DB instance becomes available and perform the pre-check again.

### 1.3.2 Checking Whether the Source and Destination Databases Are of the Same Type

#### MongoDB Migration

**Table 1-112** Checking whether the source and destination databases are of the same type

<b>Check Item</b>	Whether the source and destination databases are of the same type
<b>Description</b>	Check whether the source and destination databases are of the same type. If they are of different types, the migration fails.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The destination database is a cluster but the source database is a replica set. Handling suggestion: Change the type of the source or destination database.
	Failure cause: The destination database is a replica set but the source database is a cluster. Handling suggestion: Change the type of the source or destination database.

## 1.3.3 Checking Whether the ChangeStream API of the source DB instance is available

### MongoDB-to-DDS Migration

**Table 1-113** Checking whether the ChangeStream API of the source DB instance is available

<b>Check Item</b>	Whether the ChangeStream API of the source DB instance is available
<b>Description</b>	Check whether the ChangeStream API of the source database is available.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The source database cannot use the ChangeStream API. Handling suggestion: <ol style="list-style-type: none"><li>1. Check whether the source database version is MongoDB 4.0 or later.</li><li>2. Check whether the WiredTiger storage engine is enabled for the source database. If not, you are advised to create a DRS task and select the oplog mode. Run the following command (on a shard node): <code>db.serverStatus().storageEngine.name;</code></li></ol>
	Failure cause: The source database cannot use the ChangeStream API. Handling suggestion: Check whether the source database version is DDS 4.0 or later. If not, upgrade the source database to DDS 4.0 or later.

## 1.4 Destination DB Instance Statuses

### 1.4.1 Checking Whether the Destination DB Instance Is Available

In the pre-check phase, DRS checks the status of the destination DB instance.

#### Failure Cause

- The destination DB instance is unavailable.
- The destination DB instance is read-only.

#### Handling Suggestion

- If the destination DB instance is abnormal and cannot be accessed by DRS, wait until the DB instance becomes available and perform the pre-check again.
- If the destination DB instance is read-only and cannot be written, replace the destination database and perform the pre-check again.



## 1.4.2 Checking Whether the Destination Database Is Involved in Another Migration Task

### MySQL

**Table 1-114** Checking whether the destination database is involved in another migration task

<b>Check Item</b>	Whether the destination database is involved in another migration task
<b>Description</b>	Check whether the destination database is being used in another migration task. If more than one migration task uses the same destination database, the migration may fail.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The destination RDS DB instance is being used in another migration task. Handling suggestion: Wait for the migration task to complete. You can also stop or delete an unused migration task.

## 1.4.3 Checking Whether the Destination Database Has a Read Replica

### MySQL

**Table 1-115** Checking whether the destination database has a read replica

<b>Check Item</b>	Whether the destination database has a read replica
<b>Description</b>	Check whether the destination database has read replicas. If the destination database has read replicas, the incremental migration may fail.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: In an incremental migration, the destination database cannot have read replicas. Handling suggestion: Delete the read replicas from the destination database. After the migration is complete, create read replicas.

## 1.4.4 Checking Whether the Destination Database Is Read-Only

### MySQL Migration, Synchronization, and Backward DR

Table 1-116 Checking whether the destination database is read-only

<b>Check Item</b>	Whether the destination database is read-only
<b>Description</b>	The destination database is read-only, and data cannot be written to the destination database.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The destination database is read-only. Handling suggestion: Run the following commands to change the destination database to read/write and then, restart the database. Sample commands: set global read_only=0; set global super_read_only=0;

## 1.4.5 Checking Whether the Extensions Are Supported

### PostgreSQL Synchronization

Table 1-117 Checking whether the extensions are supported

<b>Check Item</b>	Whether the extensions are supported
<b>Description</b>	Check whether the source database has plug-ins that are not installed on the destination database.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: Extensions installed in the source database are not supported in the destination database. Handling suggestion: <ul style="list-style-type: none"><li>If the source database services do not depend on those extensions, run the following statement to delete the extensions. Replace <i>plugin_name</i> with the name of the extension to be deleted. drop extension plugin_name;</li><li>Alternatively, use a destination database that supports these extensions.</li></ul>

	<p>Failure cause: The source database has extensions that contain tables as members.</p> <p>Handling suggestion: Check whether the source database extensions contain metadata generated after the extensions are created. If yes, use the dedicated syntax of the extension to rebuild the metadata after the migration is complete.</p>
	<p>Failure cause: The destination database user does not have the permission to create extensions.</p> <p>Handling suggestion: Grant the permission to the user in the destination database as user root. Run the following SQL statements (replace <i>username</i> with the destination database username):</p> <pre>alter user username inherit; grant root to username;</pre>
	<p>Failure cause: The extension version supported by the destination database is earlier than that installed in the source database.</p> <p>Handling suggestion: Use the destination database that supports extensions of a later version (not earlier than the source database extension version) and create a synchronization task again.</p>

## 1.4.6 Checking Whether Destination Contains the Configured Database

### MySQL > PostgreSQL

**Table 1-118** Checking whether destination contains the configured database

<b>Check Item</b>	Whether the destination contains the configured database.
<b>Description</b>	Databases and schemas cannot be migrated. You need to manually create databases and schemas on the destination database. Otherwise, the migration will fail.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: Databases cannot be migrated from MySQL to PostgreSQL.</p> <p>Handling suggestion: In the destination database, manually create databases and schemas with the same names as those of the source database.</p>
	<p>Failure cause: The objects to be synchronized already exist in the destination database.</p> <p>Handling suggestions: Delete the tables to be synchronized from the destination database or select the tables that do not exist in the destination database for synchronization.</p>

## MySQL -> GaussDB Synchronization

**Table 1-119** Checking whether destination contains the configured database

Check Item	Whether the destination contains the configured database.
Description	In the MySQL to GaussDB synchronization scenario, the mapped database must exist in the destination database. Otherwise, the synchronization fails.
Failure Cause and Handling Suggestion	Failure cause: The destination database does not contain the configured database. Handling suggestion: Before the synchronization, manually create a mapped database in the destination database.

## GaussDB->GaussDB Synchronization

**Table 1-120** Checking whether destination contains the configured database

Check Item	Whether the destination contains the configured database.
Description	In the GaussDB to GaussDB synchronization scenario, the mapped database must exist in the destination database. Otherwise, the synchronization fails.
Failure Cause and Handling Suggestion	Failure cause: The configured database does not exist in the destination. Handling suggestion: Before the synchronization, manually create a configured database in the destination.

## 1.4.7 Checking Whether the Destination DB Instance Is Available

**Table 1-121** Checking whether the destination DB instance is available

Check Item	Whether the destination DB instance is available
Description	Check whether the primary instance and read replicas are available in the destination database. If not, the migration fails.

<b>Failure Cause and Handling Suggestion</b>	Failure cause: The destination DB instance is not available. Handling suggestion: Repair the destination DB instance.
	Failure cause: Read replicas in the destination database are abnormal. Handling suggestion: Repair the abnormal read replicas in the destination database.
	Failure cause: The RDS service is abnormal. Try again later. Handling suggestion: Try again later.

## 1.4.8 Checking Whether the Destination Database Is Empty

### MySQL

**Table 1-122** Checking whether the destination database is empty

<b>Check Item</b>	Whether the destination database is empty
<b>Description</b>	Check whether the destination database is empty. If the destination database is not empty, disaster recovery will fail.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The destination database is not empty. Handling suggestion: Delete all the user-created databases from the destination database.

## 1.5 Database User Permissions

### 1.5.1 Whether the Source Database User Has Sufficient Permissions

Check whether the source database user has sufficient permissions. The source database user permissions required in the full and incremental phases vary depending on the DB engine.

#### Failure Cause

The source database user does not have sufficient permissions.

#### Handling Suggestion

When you use DRS to migrate or synchronize data, the source database user must have required permissions. Tasks of different DB engines and modes require different account permissions. DRS automatically checks the database account permissions in the pre-check phase and provides handling suggestions.

Take the MySQL migration as an example. The source database user permissions are as follows:

- Full migration:  
SELECT, SHOW VIEW, and EVENT  
Reference statement: **GRANT** SELECT, SHOW VIEW, EVENT **ON** \*.\* **TO** 'user1';
- Full+incremental migration:  
SELECT, SHOW VIEW, EVENT, LOCK TABLES, REPLICATION SLAVE, and REPLICATION CLIENT

REPLICATION SLAVE and REPLICATION CLIENT are global permissions and must be enabled separately. The reference statement is as follows:

**GRANT** REPLICATION SLAVE, REPLICATION CLIENT **ON** \*.\* **TO** 'user1';

SELECT, SHOW VIEW, EVENT, and LOCK TABLES are non-global permissions. The reference statement is as follows:

**GRANT** SELECT, SHOW VIEW, EVENT, LOCK TABLES, **ON** [Database to be migrated].\* **TO** 'user1';

## Related Documents

- [Which MySQL Permissions Are Required for DRS?](#)
- [How Do I Set an Independent Oracle Account That Has the Least Privilege and Uses DRS?](#)

## 1.5.2 Checking Whether the Destination Database User Has Sufficient Permissions

Check whether the destination database user has sufficient permissions. The destination database user permissions required in the full and incremental phases vary depending on the DB engine.

### Failure Cause

The destination database user does not have sufficient permissions.

### Handling Suggestion

When you use DRS to migrate or synchronize data, the destination database user must have required permissions. Tasks of different DB engines require different account permissions. DRS automatically checks the database account permissions in the pre-check phase and provides handling suggestions.

Take the MySQL migration as an example. The destination database user permissions are as follows:

SELECT, CREATE, ALTER, DROP, DELETE, INSERT, UPDATE, INDEX, EVENT, CREATE VIEW, CREATE ROUTINE, TRIGGER, REFERENCES, and WITH GRANT OPTION. If the destination database version is in the range 8.0.14 to 8.0.18, the SESSION\_VARIABLES\_ADMIN permission is required.

Reference statement: **GRANT**SELECT, CREATE, ALTER, DROP, DELETE, INSERT, UPDATE, INDEX, EVENT, CREATE VIEW, CREATE ROUTINE, TRIGGER, REFERENCES ON [*Databases to be migrated*].\* **TO** 'user1' **WITH GRANT OPTION**;

## Related Documents

- [Which MySQL Permissions Are Required for DRS?](#)
- [How Do I Set an Independent Oracle Account That Has the Least Privilege and Uses DRS?](#)

## 1.5.3 Checking Whether the Destination Database Account Has Required Permissions to Migrate Definer

### MySQL Migration

**Table 1-123** Checking Whether the Destination Database Account Has Required Permissions to Migrate Definer

<b>Check Item</b>	Checking whether the destination database account has required permissions to migrate Definer
<b>Description</b>	To migrate Definers to the cloud, the source database user must have the all privileges permission.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The permission of the specified destination database account is insufficient.</p> <p>Handling suggestions: Migrate all Definers to the specified destination database account. Alternatively, do not migrate Definers to the specified destination database account and run the following command to grant the destination database account the all privileges permission.</p> <p>Example command:  <code>grant all privileges on *.* to 'user' @' host'</code></p>
	<p>Failure cause: The specified source database user does not have required permissions.</p> <p>Handling suggestion:</p> <ol style="list-style-type: none"> <li>1. When configuring the destination database, select <b>Migrate Definer to User</b> to ensure that Definers of all the objects are under the specified user.</li> <li>2. Retain the Definer settings and run the following command to grant the all privileges permission to the source database user.</li> </ol> <p>Example command:  <code>grant all privileges on *.* to 'user' @' host'</code></p>

## 1.6 Database Versions

## 1.6.1 Checking Whether the Source Database Version Is Supported

Check whether the source database version is supported. Different DB engines have different supported versions. For details, see [Supported Databases](#).

### Failure Cause

The source database version does not meet the migration requirements.

### Handling Suggestion

Select a source database version that meets requirements.

## 1.6.2 Checking Whether the Destination Database Version Is Supported

Check whether the destination database version is supported. Different DB engines have different supported versions. For details, see [Supported Databases](#).

### Failure Cause

The destination database version does not meet the migration requirements.

### Handling Suggestion

Select a destination database version that meets requirements.

## 1.6.3 Checking Whether the Migration Is from an Earlier Database Version to the Same or a Later Version

For homogeneous migration, DRS checks whether data is migrated from an earlier version to a later version or the same version. The database of a later version has new features. If the destination database does not have such features, data migration may fail.

### Failure Cause

The destination database version is earlier than the source database version.

### Handling Suggestion

Select a source or destination database version that meets requirements, or ensure that the source database does not use new features provided by a later version. Otherwise, the migration may fail.

## 1.7 Networks



## 1.7.1 Checking Whether the Source Database Is Connected

In the pre-check phase, DRS checks the connectivity and accuracy of the source database IP address, port, username, and password.

### Failure Cause

- The username or password is incorrect.
- The port cannot be accessed.
- The database account does not allow remote connection.
- The connection fails.

### Handling Suggestion

- Check whether the username and password entered during the DRS connection test are correct. Enter the correct database username and password and perform the pre-check again.
- If the port entered during the connection test cannot be accessed, check whether the port exists. If the port is correct, check whether the firewall is enabled.
- If the source database is PostgreSQL, and the database configuration file **pg\_hba.conf** does not contain the database account configuration, grant the remote connection permission for the account.

Add the following to **pg\_hba.conf**, and restart the database for the modification to take effect:

```
host all xxx(dbuser) 0.0.0.0/0 method
```

After the task is complete, delete this record and restart the database again.

- Before data migration, ensure that the network has been prepared well and security rules have been configured. If the connection fails, perform the following operations to check whether the network configuration is correct:
  - Public network
    - i. Ensure that public accessibility is enabled for the database.
    - ii. Ensure that the security rules of the database are correctly configured.

You need to add the EIP of the DRS instance to the whitelist of the database to ensure that the DRS instance can access the database. The IP address displayed on the **Configure Source and Destination Databases** page is the EIP of the DRS instance.
    - iii. Ensure that the firewall settings are correct.

The firewall of the data center must allow access from the EIP of the DRS instance so that the DRS instance can access the database.

Inbound access: Allows access from the EIP of the DRS instance to the database listening port.

Outbound access: Allows data transmission from the database listening port to the EIP of the DRS instance.
  - VPC
    - i. Ensure that the database security group is correctly configured.

View inbound rules to allow traffic from the private IP address of the DRS instance to the database listening port. The IP address displayed on the **Configure Source and Destination Databases** page is the private IP address of the DRS instance.

- ii. Ensure that the database network ACL is correctly configured.  
By default, a VPC does not have a network ACL. If you have configured a network ACL, add an inbound rule.
- VPN or Direct Connect
  - i. Ensure that the database VPN or Direct Connect is correctly configured.
  - ii. Ensure that the security rules of the database are correctly configured.

You need to add the private IP address of the DRS instance to the whitelist of the database to ensure that the DRS instance can access the database. The IP address displayed on the **Configure Source and Destination Databases** page is the private IP address of the DRS instance.

For more details about network settings, see [Overview of Preparations](#).

## 1.7.2 Checking Whether the Destination Database Is Connected

In the pre-check phase, DRS checks the connectivity and accuracy of the destination database IP address, port, username, and password.

### Failure Cause

- The username or password is incorrect.
- The port cannot be accessed.
- The database account does not allow remote connection.
- Kafka brokers are unavailable.
- The connection fails.

### Handling Suggestion

- Check whether the username and password entered during the DRS connection test are correct. Enter the correct database username and password and perform the pre-check again.
- If the port entered during the connection test cannot be accessed, check whether the port exists. If the port is correct, check whether the firewall is enabled.
- If the destination database is PostgreSQL, and the database configuration file **pg\_hba.conf** does not contain the database account configuration, grant the remote connection permission for the account.

Add the following to **pg\_hba.conf**, and restart the database for the modification to take effect:

```
host all xxx(dbuser) 0.0.0.0/0 method
```

After the task is complete, delete this record and restart the database again.

- If the destination database is Kafka, the possible causes are as follows:
  - Check whether Kafka brokers are normal.
  - Check whether security authentication is enabled on Kafka. If security authentication is enabled, select the corresponding security connection mode. For details, see [Kafka Authentication](#).
- Before data migration, ensure that the network has been prepared well and security rules have been configured. If the connection fails, perform the following operations to check whether the network configuration is correct:
  - Public network
    - i. Ensure that public accessibility is enabled for the database.
    - ii. Ensure that the security rules of the database are correctly configured.

You need to add the EIP of the DRS instance to the whitelist of the database to ensure that the DRS instance can access the database. The IP address displayed on the **Configure Source and Destination Databases** page is the EIP of the DRS instance.
    - iii. Ensure that the firewall settings are correct.

The firewall of the data center must allow access from the EIP of the DRS instance so that the DRS instance can access the database.

Inbound access: Allows access from the EIP of the DRS instance to the database listening port.

Outbound access: Allows data transmission from the database listening port to the EIP of the DRS instance.
  - VPC
    - i. Ensure that the database security group is correctly configured.

View inbound rules to allow traffic from the private IP address of the DRS instance to the database listening port. The IP address displayed on the **Configure Source and Destination Databases** page is the private IP address of the DRS instance.
    - ii. Ensure that the database network ACL is correctly configured.

By default, a VPC does not have a network ACL. If you have configured a network ACL, add an inbound rule.
  - VPN or Direct Connect
    - i. Ensure that the database VPN or Direct Connect is correctly configured.
    - ii. Ensure that the security rules of the database are correctly configured.

You need to add the private IP address of the DRS instance to the whitelist of the database to ensure that the DRS instance can access the database. The IP address displayed on the **Configure Source and Destination Databases** page is the private IP address of the DRS instance.

For more details about network settings, see [Overview of Preparations](#).

## 1.7.3 Checking Whether the Destination Database Can Connect to the Source Database

### MySQL Migration and Synchronization

**Table 1-124** Checking whether the destination database can connect to the source database

<b>Check Item</b>	Whether the destination database can connect to the source database
<b>Description</b>	Check whether the destination database can connect to the source database.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The destination database fails to connect to the source database. Handling suggestion: See network preparations in the Data Replication Service Best Practices.

## 1.8 Database Objects

### 1.8.1 Checking Whether the Source Database Contains a MyISAM Table

#### MySQL

**Table 1-125** Checking whether the source database contains a MyISAM table

<b>Check Item</b>	Whether the source database contains a MyISAM table
<b>Description</b>	If the source database contains a MyISAM table, the migration will fail.
<b>Item to Be Confirmed and Handling Suggestion</b>	Item to be confirmed: The source database contains MyISAM tables that are not supported by the destination database, which may cause the migration to fail. Handling suggestion: Convert the tables in the source database to InnoDB tables and try again. Alternatively, contact technical support.

## 1.8.2 Checking Whether the Source Database Contains Unsupported Table Field Types

### Oracle Migration and Synchronization

**Table 1-126** Checking whether the source database contains unsupported table field types

<b>Check Item</b>	Whether the source database contains unsupported table field types
<b>Description</b>	<p>Unsupported table field types are as follows: BFILE, XMLType, SDO_GEOMETRY, TIMESTAMP (x), INTERVAL DAY TO SECOND (x), and UROWID. The source database contains unsupported table field types, resulting in migration failures.</p> <p><b>NOTE</b></p> <ul style="list-style-type: none"> <li>• x indicates the precision. TIMESTAMP and INTERVAL DAY TO SECOND do not support the column type with precision between 7 to 9.</li> <li>• Due to internal restrictions, DRS cannot filter out special fields during data processing.</li> </ul>
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The source database contains unsupported table field types.</p> <p>Handling suggestion: Delete the columns containing the unsupported field types. Alternatively, do not migrate the tables containing the unsupported table field types.</p>

### MySQL -> PostgreSQL Synchronization

**Table 1-127** Checking whether the source database contains unsupported table field types

<b>Check Item</b>	Whether the source database contains unsupported table field types
<b>Description</b>	The following table fields types are not supported: geometry, point, lineString, polygon, geometrycollection, multipoint, multilinestring, and multipolygon. The source database contains unsupported table field types, resulting in migration failures.

<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The source database contains unsupported table field types.</p> <p>Handling suggestion: Delete the columns containing the unsupported field types. Alternatively, do not migrate the tables containing the unsupported table field types.</p>
--	---

### 1.8.3 Checking Whether the Source Database Contains the Functions or Stored Procedures that the Source Database User Is Not Authorized to Migrate

#### MySQL

**Table 1-128** Checking whether the source database contains the functions or stored procedures that the source database user is not authorized to migrate

<b>Check Item</b>	Whether the source database contains the functions or stored procedures that the source database user is not authorized to migrate.
<b>Description</b>	The source database contains the functions or stored procedures that the source database user is not authorized to migrate.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The source database user does not have the permission to migrate functions and stored procedures.</p> <p>Handling suggestion: Ensure that the source database user has the highest-level right.</p>

### 1.8.4 Checking Whether Objects with the Same Names Exist in the Source Database

#### MySQL -> PostgreSQL Synchronization

**Table 1-129** Checking whether objects with the same names exist in the source database

<b>Check Item</b>	Whether objects with the same names exist in the source database
-------------------	--

<b>Description</b>	Failure cause: The source databases selected are not mapped to the same database, or tables with same names exist in the selected databases.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: PostgreSQL does not support synchronization of multiple databases, or objects have the same names in the source databases. Handling suggestion: Select one database for migration, or map the selected databases to the same database and ensure that the object in the databases have unique names.

## 1.8.5 Whether the Source Database Contains Unlogged Tables

### PostgreSQL as the Source

**Table 1-130** Whether the source database contains unlogged tables

<b>Check Item</b>	Whether the source database contains unlogged tables
<b>Description</b>	Check whether the source database contains unlogged tables. If yes, the synchronization fails.
<b>Item to Be Confirmed and Handling Suggestion</b>	Item to be confirmed: The source database contains unlogged tables and modifications to these tables are not recorded in logs. As a result, incremental data of unlogged tables cannot be synchronized. Handling suggestions: Check whether incremental data of the unlogged tables needs to be synchronized. If yes, run the following command to change the unlogged tables to logged: ALTER TABLE TABLE_NAME SET LOGGED."

## 1.8.6 Checking Whether the Names of Views to Be Migrated Are the Same

### Oracle Migration and Synchronization

**Table 1-131** Checking whether the source database meets integrity constraints

<b>Check Item</b>	Whether the source database meets constraint integrity
<b>Description</b>	Check the constraint integrity of the source database. If the check result does not meet the migration requirements, the migration fails.

<b>Item to Be Confirmed and Handling Suggestion</b>	Failure cause: Tables to be migrated are dependent but the referenced tables are not to be migrated. Handling suggestion: Select the referenced tables in the migration object list.
	Failure cause: Views that cannot be migrated are selected in the migration object list. Handling suggestions: Check all objects which the views reference and select them for migration.
	Failure cause: The names of the views to be migrated are the same except for letter cases. Handling suggestions: Change the view names or do not migrate these views.
	Failure cause: Source database constraint names are the same except for letter cases. Handling suggestions: Change the constraint names or do not migrate the constraints whose names are the same except for letter cases.

## 1.8.7 Checking Whether the `_id` Fields in the Collection of the Source Database Have Indexes

### MongoDB Migration

**Table 1-132** Checking whether the `_id` fields in the collections of the source database have indexes

<b>Check Item</b>	Whether the <code>_id</code> fields in the collections of the source database have indexes
<b>Description</b>	Check whether the <code>_id</code> fields in the collections of the source database have indexes. If not, the migration fails.
<b>Item to Be Confirmed and Handling Suggestion</b>	Failure cause: The <code>_id</code> fields in the collections of the source database must have indexes. Handling suggestion: Run the <code>db.CollectionName.ensureIndex({_id: 1})</code> command to add an index. If the index fails to be added and the system displays a message indicating that the <code>_id</code> value already exists, the collection cannot be migrated.



## 1.8.8 Checking Whether the Index Length of the Source Database Exceeds the Limit

### Oracle -> MySQL/Oracle -> GaussDB(for MySQL) Migration and Synchronization

**Table 1-133** Checking whether the index length of the source database exceeds the limit

<b>Check Item</b>	Whether the index length of the source database exceeds the limit
<b>Description</b>	The migration fails because the index length of the source database exceeds the column length limit of the destination database.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: There are indexes in the source database exceed the column length limit of the destination database. Handling suggestion: 1. Delete the tables from the migration object. 2. Modify the index length.

## MySQL Migration, Synchronization, and Disaster Recovery

**Table 1-134** Checking whether the index length of the source database exceeds the column length limit

<b>Check Item</b>	Whether the index length of the source database exceeds the limit
<b>Description</b>	The migration fails because the index length of the source database exceeds the column length limit of the destination database.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: There are indexes in the source database exceed the column length limit of the destination database. Handling suggestion: Set <b>innodb_large_prefix</b> of the destination database to <b>ON</b> , or return to the previous step and select the table to be migrated again.

## 1.8.9 Checking Whether the Source Database Tables Use Storage Engines Not Supported by the Destination Database

Check whether the source database tables use storage engines not supported by the destination database. If yes, the migration fails.

### Failure Cause

The source database tables use the storage engines that are not supported by the destination database.

### Handling Suggestion

- Step 1** Go back to the object selection page.
- Step 2** Deselect the tables that use the storage engines not supported by the destination database.
- Step 3** Click **Next** to perform the pre-check again.

----End

## 1.8.10 Checking Whether the Database Names Mapped to the Destination DB Instance Contain Unsupported Characters

### MySQL

**Table 1-135** Checking whether the database names mapped to the destination database contain unsupported characters.

<b>Check Item</b>	Whether the database names mapped to the destination DB instance contain unsupported characters.
<b>Description</b>	The following characters are not supported in the database names mapped to the destination DB instance: .<>\'
<b>Item to Be Confirmed and Handling Suggestion</b>	Failure cause: The database names mapped to the destination DB instance contain unsupported characters. Handling suggestion: Go back to the object selection page and change the source database names to be mapped to the destination DB instance.

## 1.8.11 Checking Whether the Source Database Tables Contain Primary Keys

### MySQL Migration and Disaster Recovery

**Table 1-136** Checking whether the source database tables contain primary keys

<b>Check Item</b>	Whether the source database tables contain primary keys
<b>Description</b>	If tables to be migrated in the source database do not contain primary keys, the migration may fail.
<b>Item to Be Confirmed and Handling Suggestion</b>	Item to be confirmed: The tables to be migrated in the source database do not contain primary keys. Handling suggestion: Create a primary key for the table. If the table does not have a primary key to uniquely identify every row and the network connection is unstable, the data in the destination database may be inconsistent with that in the source database.

### MySQL Synchronization

**Table 1-137** Checking whether the source database tables contain primary keys

<b>Check Item</b>	Whether the source database tables contain primary keys
<b>Description</b>	If tables to be synchronized in the source database do not contain primary keys, the synchronization may fail.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The tables to be synchronized in the source database do not contain primary keys. Handling suggestion: Create primary keys for the tables to improve performance. Failure cause: In a many-to-one task, tables with no primary key that have the same name as those in the destination database are not allowed in the source database. Handling suggestion: Modify tables without primary keys, delete tables with no primary key from the destination database, or do not migrate tables without primary keys.

<b>Item to Be Confirmed and Handling Suggestion</b>	<p>Item to be confirmed: The tables to be synchronized in the source database do not contain primary keys.</p> <p>Handling suggestion: Create a primary key for the table. If the table does not have a primary key to uniquely identify every row and the network connection is unstable, the data in the destination database may be inconsistent with that in the source database.</p>
---	---

## Synchronization from Microsoft SQL Server to GaussDB(DWS)

**Table 1-138** Whether the source database tables contain primary keys

<b>Check Item</b>	Whether the source database tables contain primary keys
<b>Description</b>	If the source database contains tables that do not have primary keys, a small amount of data may be inconsistent during synchronization.
<b>Item to Be Confirmed and Handling Suggestion</b>	<p>Item to Be Confirmed: The table without a primary key lacks a unique identifier for rows. When the network is unstable, you may need to retry the task several times, or data inconsistency may occur.</p> <p>Handling suggestion: Add a primary key to a table, or do not synchronize the table that does not have a primary key.</p> <p>Statement for adding a primary key:  <code>ALTER TABLE table_name ADD CONSTRAINT constraint-name PRIMARY KEY (column_name);</code></p>

## Oracle Synchronization

**Table 1-139** Checking whether the source database tables contain primary keys

<b>Check Item</b>	Whether the source database tables contain primary keys
<b>Description</b>	If the source database contains tables that do not have primary keys, a small amount of data may be inconsistent during synchronization.

<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The table without a primary key lacks a unique identifier for rows. When the network is unstable, you may need to retry the task several times, or data inconsistency may occur.</p> <p>Handling suggestion: Add a primary key to a table, or do not synchronize the table that does not have a primary key.</p> <p>Statement for adding a primary key:  <code>ALTER TABLE table_name ADD CONSTRAINT constraint-name PRIMARY KEY (column_name);</code></p> <p>To synchronize tables with primary keys, do not perform the ALTER TABLE MOVE, SPLIT/MERGE, FLASHBACK, or ALTER TABLE SHRINK SPACE operation during full synchronization. Otherwise, data duplication may increase.</p>
--	--

## 1.8.12 Checking Whether the Source Database Contains Triggers or Events

### MySQL Migration

**Table 1-140** Checking whether the source database contains triggers or events

<b>Check Item</b>	Whether the source database contains triggers or events
<b>Description</b>	To prevent unexpected operations on the destination database automatically triggered by triggers or events, this task starts the trigger or event migration only after you stop the task. If you close or disconnect the source database connection during the task running, triggers or events are not migrated.
<b>Item to Be Confirmed and Handling Suggestion</b>	<p>Item to be confirmed: The source database contains triggers or events.</p> <p>Handling suggestion: Stop the task first and then disconnect the network to ensure the completeness of the migration.</p>

## 1.8.13 Checking Whether the Source Database Referenced Roles Pass the Check

### MongoDB Migration

**Table 1-141** Checking whether the source database referenced roles pass the check

<b>Check Item</b>	Whether the source database referenced roles pass the check
-------------------	---

<b>Description</b>	If the roles referenced by accounts to be migrated are not migrated to the destination database, the migration may fail.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The databases referenced by source database roles do not exist in the destination database, and are not displayed in the object selection list. Handling suggestion: Select the databases referenced by roles for migration or do not migrate the roles.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The roles referenced by source database roles do not exist in the destination database, and are not displayed in the role selection list. Handling suggestion: Select the referenced roles or do not migrate the roles that fail the check.

## 1.8.14 Checking Whether the Source Database Referenced Accounts Pass the Check

### MongoDB Migration

**Table 1-142** Checking whether the source database referenced accounts pass the check

<b>Check Item</b>	Whether the source database referenced accounts pass the check
<b>Description</b>	If the roles referenced by accounts to be migrated are not migrated to the destination database, the migration may fail.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The databases referenced by source database account do not exist in the destination database, and are not displayed in the object selection list. Handling suggestion: Select the roles referenced by accounts for migration or do not migrate the accounts.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The databases referenced by source database account do not exist in the destination database, and are not displayed in the object selection list. Handling suggestion: Select the databases referenced by accounts for migration or do not migrate the accounts.

## 1.8.15 Checking Whether the Source Database Contains Schemas or Users Named cdc

### Microsoft SQL Server as the Source

**Table 1-143** Checking whether the source database contains schemas or users named cdc

<b>Check Item</b>	Whether the source database contains schemas or users named cdc
<b>Description</b>	In full+incremental mode, the source database contains a schema or user named cdc.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The source database contains a schema or user named cdc. Handling suggestion: Go back to the object selection page and deselect the schema and user named cdc.

## 1.8.16 Checking Whether Associated Objects Are Selected

### PostgreSQL Serving as the Source in Synchronization

**Table 1-144** Checking whether associated objects are selected

<b>Check Item</b>	Whether the associated objects are selected
<b>Description</b>	The associated objects must be selected for migration. Otherwise, the migration may fail.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: Tables referenced by the foreign key in the table to be migrated are not selected for migration. Handling suggestion: Select the associated objects.
	Failure cause: The selected objects contain views associated with some tables or views that are not selected for migration. Handling suggestion: Select the associated objects.
	Failure cause: The tables associated with the child tables to be migrated are not selected for migration. Handling suggestion: Select the associated objects.

## 1.8.17 Checking Whether the Specified Objects Exist In the Destination Database

### PostgreSQL to RDS PostgreSQL Migration and Synchronization, and PostgreSQL to GaussDB(DWS) Synchronization

**Table 1-145** Checking whether the specified migration objects exist in the destination database.

<b>Check Item</b>	Whether the specified migration objects exist in the destination database.
<b>Description</b>	The objects with the same names as the source objects cannot exist in the destination database. Otherwise, the migration may fail.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The objects to be synchronized exist in the destination database. Handling suggestion: Delete the objects with the same names as the source objects and perform the verification again.

## 1.8.18 Checking Whether the Source Table Contains Column Types that Cannot Be Used as Distribution Keys

### PostgreSQL to GaussDB(DWS) Synchronization

**Table 1-146** Checking whether the source table contains column types that cannot be used as distribution keys

<b>Check Item</b>	Whether the source table contains column types that cannot be used as distribution keys
<b>Description</b>	The source table cannot contain column types used as distribution keys. Otherwise, the synchronization may fail.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The source table contains column types that cannot be used as distribution keys. Handling suggestion: Select another table that can be migrated.



## 1.8.19 Checking Whether the Source Table Contains Unsupported Table Field Types

### PostgreSQL to GaussDB(DWS) Synchronization

**Table 1-147** Checking whether the source table contains unsupported table field types

<b>Check Item</b>	Whether the source table contains unsupported table field types
<b>Description</b>	The source tables cannot contain unsupported field types. Otherwise, the synchronization fails.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The source database table contains unsupported table field types. Handling suggestion: Select another table that can be migrated.

## 1.9 Database Configuration Items

### 1.9.1 Checking Whether the Source Database Name Is Valid

#### MySQL Migration

**Table 1-148** Checking whether the source database name is valid

<b>Check Item</b>	Whether the source database name is valid
<b>Description</b>	The source database name cannot contain invalid characters. It must contain 1 to 64 characters, including only lowercase letters, digits, hyphens (-), and underscores (_). If the source database name contains any invalid character, the migration fails.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: This item cannot be checked because the source database fails to be connected. Handling suggestion: Check whether the source database is connected.

	<p>Failure cause: The source database name cannot contain invalid characters. It must contain 1 to 64 characters, including only lowercase letters, digits, hyphens (-), and underscores (_).</p> <p>Handling suggestion: Change the source database names that contain unsupported characters or go back to the previous page and select the databases that do not contain unsupported characters.</p>
--	---

## MongoDB Migration

**Table 1-149** Checking whether the source database name is valid

<b>Check Item</b>	Whether the source database name is valid
<b>Description</b>	<p>If the source database name contains invalid characters, the migration fails.</p> <p>Failure cause: The source database names cannot contain the following special characters: .&lt;&gt;'</p> <p>Handling suggestion: Change the source database names that contain unsupported characters or go back to the previous page and select the databases that do not contain unsupported characters.</p>

## Oracle Migration

**Table 1-150** Checking whether the source database name is valid

<b>Check Item</b>	Whether the source database name is valid
<b>Description</b>	<p>The source database names cannot contain non-ASCII characters or special characters .&gt;&lt;\ ,?!"</p> <p>If the source database name contains invalid characters, the migration fails.</p> <p>Failure cause: The source database names contain unsupported characters.</p> <p>Handling suggestion: Change the source database names that contain unsupported characters or go back to the previous page and select the databases that do not contain unsupported characters.</p>

## 1.9.2 Checking Whether the Source Database Table Name Is Valid

### MySQL Migration

**Table 1-151** Checking whether the source database table name is valid

<b>Check Item</b>	Whether the source database table name is valid
<b>Description</b>	If the source database table name contains invalid character, the synchronization task fails.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The source database table names contain unsupported characters, non-ASCII characters, or the following characters: &gt;&lt;/\</p> <p>Handling suggestion: To solve this problem, perform the following steps:</p> <p>Click <b>Previous</b> to return to the <b>Select Migration Type</b> page. Select a customized object and do not select the table that contains unsupported characters.</p> <p>Method 2: Change the table name.</p>

### PostgreSQL Migration

**Table 1-152** Checking whether the source database table name is valid

<b>Check Item</b>	Whether the source database table name is valid
<b>Description</b>	The source database table name cannot contain single quotation marks ('), double quotation marks ("), and periods (.). Ensure that the source database table name does not contain invalid characters. Otherwise, the synchronization task fails.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The source database names contain unsupported characters.</p> <p>Handling suggestion: Run the following command to change the source database names that contain unsupported characters:</p> <pre>alter table old_name rename to new_name;</pre>

## Oracle Migration

**Table 1-153** Checking whether the source database table name is valid

<b>Check Item</b>	Whether the source database table name is valid
<b>Description</b>	The source database names cannot contain non-ASCII characters or special characters .><`\ ,?!" If the table name or view name of the source database contains any invalid character, the migration fails.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The table name or view name of the source database contains unsupported characters.</p> <p>Handling suggestion:</p> <ul style="list-style-type: none"><li>• Run the following command to change the table name that contains unsupported characters: alter table old_name rename to new_name;</li><li>• Run the following command to change the view name that contains unsupported characters: rename old_view_name to new_view_name;</li></ul>

### 1.9.3 Checking Whether the Source Database View Name Is Valid

#### MySQL

**Table 1-154** Checking whether the source database contains view names with non-ASCII characters

<b>Check Item</b>	Whether the source database contains view names with non-ASCII characters
<b>Description</b>	If the source database contains non-ASCII characters, the migration will fail.
<b>Item to Be Confirmed and Handling Suggestion</b>	<p>Failure cause: The source database view names contain unsupported characters, non-ASCII characters, or the following characters: &gt;&lt;\/</p> <p>Handling suggestion: To solve this problem, perform the following steps:</p> <p>Method 1:</p> <p>Click <b>Previous</b> to return to the <b>Select Migration Type</b> page. Select a customized object and do not select the view name that contains unsupported characters.</p> <p>Method 2: Change the view name.</p>

## 1.9.4 Checking Whether the Source Database Collection Name Is Valid

### MongoDB Migration

**Table 1-155** Checking whether the source database collection name is valid

<b>Check Item</b>	Whether the source database collection name is valid
<b>Description</b>	The source database collection names cannot contain slashes (/) or backslashes (\).
	Failure cause: The source database collection names cannot contain the following characters: ,<> Handling suggestion: Modify the source database collection names that contain invalid characters or go back to the <b>Select Migration Type</b> page and select the collections that do not contain invalid characters.

## 1.9.5 Checking Whether the Shard Key Can Be Obtained from the Source Database

### MongoDB Migration

**Table 1-156** Checking whether the shard keys can be obtained from the source database

<b>Check Item</b>	Whether the shard keys can be obtained from the source database
<b>Description</b>	Check whether the destination database user permissions meet the migration requirements. If the permissions are insufficient, the migration will fail.
<b>Failure Cause and Handling Suggestion</b>	Item to be confirmed: The source database is a replica set but the shard keys have not been configured in the destination database. Handling suggestion: If the destination database cannot obtain the shard keys of the source database, the data in the source database will be migrated to the primary shard node of the sharded cluster in the destination database. To fully utilize the read/write performance, storage capability, and high availability of the cluster, see <a href="#">FAQs</a> .

	<p>Item to be confirmed: The source database type is unknown, and the shard keys have not been configured in the destination database.</p> <p>Handling suggestion: If the destination database cannot obtain the shard keys of the source database, the data in the source database will be migrated to the primary shard node of the sharded cluster in the destination database. To fully utilize the read/write performance, storage capability, and high availability of the cluster, see <a href="#">FAQs</a>.</p>
	<p>Item to be confirmed: The source database contains collections that do not have shard keys configured.</p> <p>Handling suggestion: If the destination database cannot obtain the shard keys of the source database, the data in the source database will be migrated to the primary shard node of the sharded cluster in the destination database. To fully utilize the read/write performance, storage capability, and high availability of the cluster, see <a href="#">FAQs</a>.</p>

## 1.9.6 Checking Whether the Source Database Schema Name Is Valid

### PostgreSQL

**Table 1-157** Checking whether the source database schema name is valid

Check Item	Whether the source database schema name is valid
<b>Description</b>	The source database schema name cannot contain single quotation marks ('), double quotation marks ("), and periods (.). Ensure that the source database schema name does not contain invalid characters. Otherwise, the synchronization task fails.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The source database schema names contain unsupported characters.</p> <p>Handling suggestion: Run the following command to change the source database schema names that contain unsupported characters:</p> <pre>alter schema old_name rename to new_name;</pre>

## 1.9.7 Checking Whether the Maximum Number of Chunks in the Destination Database Is Sufficient

### MongoDB Migration

**Table 1-158** Checking whether the maximum number of chunks in the destination database is sufficient

<b>Check Item</b>	Whether the maximum number of chunks in the destination database is sufficient
<b>Description</b>	The maximum number of chunks in the destination database is insufficient to support sharding and splitting of the source database. If the maximum number of chunks is reached, chunks are not split and the write performance is negatively affected.
<b>Failure Cause and Handling Suggestion</b>	Item to be confirmed: The maximum number of chunks in the destination database is insufficient to support sharding and splitting of the source database. If the maximum number of chunks is reached, chunks are not split and the write performance is negatively affected. Handling suggestion: Select a destination DB instance of higher specifications.

## 1.9.8 Checking Whether Archive Logs Are Enabled on the Source Oracle Database

### Oracle -> MySQL Migration and Synchronization

**Table 1-159** Checking whether archive logs are enabled on the source Oracle database

<b>Check Item</b>	Whether archive logs are enabled on the source Oracle database
<b>Description</b>	During incremental migration from Oracle to MySQL, archive logs must be enabled on the source Oracle database.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: Archive logs are not enabled on the source Oracle database. Handling suggestion: Run the <b>alter database archivelog</b> command to enable archive logs on the source Oracle database.

## 1.9.9 Checking Whether Supplemental Logging Is Correctly Enabled on the Source Database

### Oracle Serving as the Source in Synchronization

**Table 1-160** Checking whether supplemental logging is correctly enabled on the source database

<b>Check Item</b>	Checking whether supplemental logging is correctly enabled on the source database
<b>Description</b>	The supplemental logging level of the source Oracle database does not meet requirements. The synchronization fails.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: A supplemental logging alarm for the source database is generated. Enabling primary key or unique key logging at the database level for the source database may lead to the loss of column data when data is updated.</p> <p>Handling suggestion: Enable all supplemental logging at the database or table level.</p> <ul style="list-style-type: none"><li>• Statement for enabling all supplemental logging: alter database add supplemental log data (all) columns;</li><li>• Statement for viewing all supplemental logging: select supplemental_log_data_all as allLog from v\$database;</li></ul>

## 1.10 Conflicts

### 1.10.1 Checking Whether the Names of the Source and Destination Databases Are the Same

#### MySQL Migration

**Table 1-161** Checking whether the names of the source and destination databases are the same

<b>Check Item</b>	Whether the names of the source and destination databases are the same
<b>Description</b>	Check whether the names of the source and destination databases are the same.



<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: This item cannot be checked because the source database fails to be connected.</p> <p>Handling suggestion: Check whether the source database is connected.</p>
	<p>Failure cause: This item cannot be checked because the destination database fails to be connected.</p> <p>Handling suggestion: Check whether the destination database is connected.</p>
	<p>Failure cause: Insufficient user permissions</p> <p>Handling suggestion: Check whether the database user permissions meet the migration requirements.</p>
	<p>Handling suggestion:</p> <ul style="list-style-type: none"> <li>• If you are migrating data to the cloud, determine whether to delete the databases with the same names as the source databases or specify a new destination DB instance based on site requirements.</li> <li>• If you are migrating data out of the cloud, determine whether to use the original destination database or specify a new destination DB instance based on site requirements.</li> </ul>
	<p>Failure cause: During an incremental migration, the source and destination databases cannot have the same names.</p> <p>Handling suggestion: Determine whether to retain these databases in the destination RDS DB instance or specify another destination RDS DB instance.</p>

## Migration from Redis to GeminiDB Redis

**Table 1-162** Checking whether the names of the source and destination databases are the same

<b>Check Item</b>	Whether the names of the source and destination databases are the same
<b>Description</b>	Check whether the names of the source and destination databases are the same.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The destination instance cannot contain databases with the same names as those in the source.</p> <p>Handling suggestion: Determine whether to retain these databases in the destination instance or specify another destination instance.</p>

## Oracle -> MySQL/Oracle -> GaussDB(for MySQL) Synchronization

**Table 1-163** Checking Whether the Names of the Source and Destination Databases Are the Same

<b>Check Item</b>	Whether the names of the source and destination databases are the same
<b>Description</b>	The mapping names of the tables to be synchronized in the source databases are the same as the names of the destination database tables.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: Source database names are the same except for letter cases.</p> <p>Handling suggestion: Change the table name or return to the object selection page and deselect the tables with the same name. Statement for changing the table name:                  ALTER TABLE old_table_name RENAME TO new_table_name;</p>
	<p>Failure cause: The destination database contains a table to be synchronized. The table name is the same as the mapping name of the table to be synchronized in the source database.</p> <p>Handling suggestion: Delete the destination database table. Statement for deleting a table:                  DROP TABLE table_name;</p>

## Oracle -> PostgreSQL Synchronization

**Table 1-164** Checking whether the names of the source and destination databases are the same

<b>Check Item</b>	Whether the names of the source and destination databases are the same
<b>Description</b>	The migration fails when the names of the source and destination databases are different.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The source and destination databases must have the same names, except that the destination database must use only lowercase letters.</p> <p>Handling suggestion: Create a database with name in lowercase letters in the destination database.</p>
	<p>Failure cause: The names of the objects to be migrated are the same except for letter cases.</p> <p>Handling suggestion: Select only one database among the databases whose names are the same except for letter cases.</p>

	<p>Failure cause: The names of the tables to be migrated are the same except for letter cases.</p> <p>Handling suggestion: Change table names or do not migrate the tables with the same names.</p>
	<p>Failure cause: The names of the tables to be migrated are the same as those in the destination database and use only lowercase letters.</p> <p>Handling suggestion: Change table names or do not migrate the tables with the same names.</p>
	<p>Failure cause: The destination database contains a table to be synchronized. The table name is the same as the mapping name of the table to be synchronized in the source database.</p> <p>Handling suggestion: Delete the destination database table. Statement for deleting a table: DROP TABLE table_name;</p>

## PostgreSQL > PostgreSQL Synchronization

**Table 1-165** Checking whether the names of the source and destination databases are the same

<b>Check Item</b>	Whether the names of the source and destination databases are the same
<b>Description</b>	Check whether the source and destination databases have the same names to prevent existing databases from being overwritten. If the source and destination databases have the same name, the migration cannot be performed.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The destination database names cannot be the same.</p> <p>Handling suggestion: Change the destination database names to prevent the database from being overwritten.</p>

## PostgreSQL to GaussDB(DWS) Synchronization

**Table 1-166** Checking whether the names of the source and destination databases are the same

<b>Check Item</b>	Whether the names of the source and destination databases are the same
-------------------	--

<b>Description</b>	Check whether the source and destination databases have the same names to prevent existing databases from being overwritten. If the source and destination databases have the same name, the migration cannot be performed.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The destination database names cannot be the same. Handling suggestion: Change the destination database names to prevent the database from being overwritten.

## DDM -> Oracle Synchronization

**Table 1-167** Checking whether the names of the source and destination databases are the same

<b>Check Item</b>	Whether the names of the source and destination databases are the same
<b>Description</b>	To synchronize data from DDM to Oracle, you need to create the corresponding database (user) in the destination database in advance. Otherwise, the synchronization fails.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: Some databases cannot be synchronized because the databases with the same names do not exist in the destination databases. Handling suggestion: Create these databases or users in the destination database or do not migrate these databases. Statement for creating a user: CREATE USER user_name IDENTIFIED BY password;

## 1.10.2 Checking Whether the Same View Names Exist in Both the Source and Destination Databases

### Migration from MongoDB to DDS

**Table 1-168** Checking whether the same view names exist in both the source and destination databases

<b>Check Item</b>	Whether the same view names exist in both the source and destination databases
-------------------	--

<b>Description</b>	Check whether the source and destination databases have the same view names to prevent existing views from being overwritten. If view of the same name exists in the destination database, the migration cannot be performed.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The same view names exist in both the source and destination databases.</p> <p>Handling suggestions: Delete the destination database views that have the same names as those in the source database. Alternatively, do not migrate the views with the same names.</p>

### 1.10.3 Checking Whether the Destination Database Contains a Non-Empty Collection with the Same Name As the Source Database

#### MongoDB Migration

**Table 1-169** Checking whether the destination database contains a non-empty collection with the same name as the source database

<b>Check Item</b>	Whether the destination database contains a non-empty collection with the same name as the source database
<b>Description</b>	Check whether the source and destination databases have the same non-empty collections to prevent existing databases from being overwritten. If the source and destination databases have the same collections, the migration cannot be performed.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: This item cannot be checked because the source database failed to be connected.                      Handling suggestion: Check whether the source database is connected.</p> <p>Failure cause: This item cannot be checked because the destination database failed to be connected.                      Handling suggestion: Check whether the destination database is connected.</p> <p>Failure cause: The same non-empty collections exist in both the source and destination databases.                      Handling suggestion: Determine whether to delete the same non-empty collections from the destination DB instance or specify a new destination DB instance.</p>

## 1.10.4 Checking Whether Destination Database Contains the Same Table Names As the Synchronization Objects

### MySQL Synchronization

**Table 1-170** Checking whether destination database contains the same table names as the synchronization objects

<b>Check Item</b>	Whether destination database contains the same table names as the synchronization objects. (table name conflicts)
<b>Description</b>	<p>The destination database contains objects with the same name as those in the source database. If table of the same name exists in the destination database, the migration cannot be performed.</p> <p>Exceptions: If the names and structures of tables in the source and destination databases are the same, the system determines that no conflict occurs.</p>
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The source and destination database tables cannot have the same names.</p> <p>Handling suggestions: Check whether the tables with the same names need to be retained. If yes, select another object for synchronization. If no, delete the tables with the same names.</p>
	<p>Failure cause: The destination database contains the same table names as those of the synchronization objects.</p> <p>Handling suggestions: Check whether the tables with the same names need to be retained. If yes, select another object for synchronization. If no, delete the tables with the same names.</p>
	<p>Failure cause: Both the source and destination databases are RDS DB instance and do not have mapped databases.</p> <p>Handling suggestion: Create mappings for the databases that are not mapped.</p>
	<p>Item to be confirmed: The destination database contains tables whose SRIDs are different from those in the source database.</p> <p>Handling suggestion: Check whether the geographic coordinate system used by the destination database table meets requirements. If the attribute of the geographic coordinate system has been specified at the source end, modify the structure of the destination database table to be the same as that at the source end.</p>

## MariaDB Synchronization

**Table 1-171** Whether destination database contains the same table names as those of the synchronization objects.

<b>Check Item</b>	Whether destination database contains the same table names as the synchronization objects. (table name conflicts)
<b>Description</b>	The destination database contains objects with the same name as those in the source database. If table of the same name exists in the destination database, the migration cannot be performed. Exceptions: If the names and structures of tables in the source and destination databases are the same, the system determines that no conflict occurs.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The source and destination database tables cannot have the same names. Handling suggestions: Check whether the tables with the same names need to be retained. If yes, select another object for synchronization. If no, delete the tables with the same names.
	Failure cause: The destination database contains the same table names as those of the synchronization objects. Handling suggestions: Check whether the tables with the same names need to be retained. If yes, select another object for synchronization. If no, delete the tables with the same names.
	Failure cause: Both the source and destination databases are RDS DB instance and do not have mapped databases. Handling suggestion: Create mappings for the databases that are not mapped.

### 1.10.5 Checking Whether the Destination Database Contains Objects with the Same Name As Those in the Source Database

#### MySQL -> PostgreSQL Synchronization

**Table 1-172** Checking whether the destination database contains objects with the same name as those in the source database

<b>Check Item</b>	Whether destination database contains objects with the same name as those in the source database
<b>Description</b>	The destination database contains objects with the same name as those in the source database. If a table of the same name exists in the destination database, the migration cannot be performed.

<b>Failure Cause and Handling Suggestion</b>	Failure cause: The source and destination database tables cannot have the same names. Handling suggestions: Check whether the tables with the same names need to be retained. If yes, select another object for synchronization. If no, delete the tables with the same names.
	Failure cause: The destination database contains the same table names as those of the synchronization objects. Handling suggestions: Check whether the tables with the same names need to be retained. If yes, select another object for synchronization. If no, delete the tables with the same names.

## Synchronization from Oracle to GaussDB(DWS)

**Table 1-173** Checking Whether the Destination Database Contains Objects with the Same Name As Those in the Source Database

<b>Check Item</b>	Whether the destination database contains objects with the same name as those in the source database
<b>Description</b>	The destination database contains tables with the same name as those in the source database.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The destination database table does not exist and cannot be synchronized. Handling suggestion: If you do not synchronize the table structure, create the table to be synchronized in the destination database in advance or synchronize the table structure. Statement for creating a table in the destination database: CREATE TABLE table_name (column_name data_type);

## Oracle to GaussDB(DWS)/Oracle to PostgreSQL Synchronization

**Table 1-174** Checking whether the destination database contains objects with the same name as those in the source database

<b>Check Item</b>	Whether the destination database contains objects with the same name as those in the source database
<b>Description</b>	The destination database contains objects with the same name as those in the source database.



<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The destination database contains the data and indexes of the table to be synchronized.</p> <p>Handling suggestion: Delete data and indexes from the destination database table. Otherwise, data inconsistency may occur.</p> <ul style="list-style-type: none"> <li>Statement for clearing data in a table:  <code>TRUNCATE TABLE table_name1;</code></li> <li>Statement for deleting an index:  <code>DROP INDEX index_name ;</code></li> </ul>
	<p>Failure cause: The source database contains encrypted objects.</p> <p>Handling suggestion: Go back to the object selection page and select database objects that are not encrypted.</p>
	<p>Failure cause: Source database names are the same except for letter cases.</p> <p>Handling suggestion: Change the table name or return to the object selection page and deselect the tables with the same name. Statement for changing the table name:  <code>ALTER TABLE old_table_name RENAME TO new_table_name;</code></p>

## Microsoft SQL Server as the Source in Synchronization

**Table 1-175** Checking whether the destination database contains objects with the same name as those in the source database

<b>Check Item</b>	Whether the destination database contains objects with the same name as those in the source database
<b>Description</b>	<p>The destination database contains objects with the same name as those in the source database.</p> <ul style="list-style-type: none"> <li>If <b>Table structure</b> is selected, the destination database cannot contain objects with the same name as those in the source database.</li> <li>If <b>Table structure</b> is not selected, create the corresponding table structure in the destination database in advance.</li> </ul>
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The destination database table does not exist and cannot be synchronized.</p> <p>Handling suggestion: If you do not synchronize the table structure, create the table to be synchronized in the destination database in advance or synchronize the table structure.</p> <p>Statement for creating a table in the destination database:  <code>CREATE TABLE table_name (column_name data_type);</code></p>
	<p>Failure cause: The table to be synchronized has been mapped to the destination database.</p> <p>Handling suggestion: Return to the <b>Set Synchronization Task</b> page, select the tables that meet the requirements. Alternatively, change the mapping names of the tables to be synchronized.</p>

	<p>Failure cause: The table to be synchronized has not been mapped to the destination database.</p> <p>Handling suggestion: Select <b>Table structure</b> to create a database table, or create the corresponding table structure in the destination database. If the table structure was not missing, check whether the name of the mapped table is correct.</p>
--	---

## Table-Level Synchronization from PostgreSQL to PostgreSQL

**Table 1-176** Checking whether destination database contains objects with the same name as those in the source database

<b>Check Item</b>	Whether destination database contains objects with the same name as those in the source database
<b>Description</b>	The destination database contains objects with the same name as those in the source database. If the same object names exist, the migration cannot be performed.
<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: The destination database contains objects with the same name as those in the source database.</p> <p>Handling suggestions: Check whether the objects with the same names need to be retained. If yes, select another object for migration. If no, delete the objects with the same names.</p>

## 1.10.6 Checking Whether Collections in Both the Source and Destination Databases Are Not Capped

### MongoDB Migration

**Table 1-177** Checking whether collections in both the source and destination databases are not capped

<b>Check Item</b>	Whether collections in both the source and destination databases are not capped
<b>Description</b>	Check whether collections in both the source and destination databases are not capped. If not, the migration fails.

<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: the destination database has a collection whose name is the same as that of the capped collection of the source database.</p> <p>Handling suggestion: To ensure data consistency, you can delete the collection in the destination database with the same name as the capped collection in the source database. Alternatively, you can choose not to migrate the capped collection that will contradict with that in the destination database.</p>
	<p>Failure cause: The collections to be migrated are capped collections and already exist in the destination database.</p> <p>Handling suggestion: To ensure data consistency, you can delete the collection in the destination database with the same name as the capped collection in the source database. Alternatively, you can choose not to migrate the capped collection that will contradict with that in the destination database.</p>

## 1.11 SSL Connections

### 1.11.1 Checking Whether the SSL Connection Is Correctly Configured

Check whether the SSL connection is correctly configured for the source or destination database. If SSL connection is enabled for the database, the database must be connected in SSL mode.

#### Failure Cause

SSL is enabled for the database. The database must be connected in SSL mode, but no certificate is uploaded.

#### Handling Suggestion

- On the **Configure Source and Destination Databases** page, enable the SSL connection and upload the certificate.
- Disable the SSL connection.

### 1.11.2 Checking Whether the SSL Connection Is Enabled for the Source Database

#### PostgreSQL

**Table 1-178** Checking whether the SSL connection is enabled for the source database

<b>Check Item</b>	Whether the SSL connection is enabled for the source database
-------------------	---

<b>Description</b>	Check whether the SSL connection is enabled for the source database.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The source database SSL connection is disabled. Handling suggestion: In the <b>postgresql.conf</b> file, set <b>ssl_ca_file</b> to the directory of an SSL root CA certificate and set <b>ssl</b> to <b>on</b> to enable the SSL connection. Then, restart the database for the modifications to take effect.

### 1.11.3 Checking Whether the SSL Certificate of the Source Database Exists

#### MySQL > MySQL

**Table 1-179** Checking whether the SSL certificate of the source database exists

<b>Check Item</b>	Whether the SSL certificate of the source database exists
<b>Description</b>	Check whether the SSL certificate type of the source database is correct during MySQL to MySQL synchronization. Otherwise, the synchronization fails.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The source database uses SSL to encrypt connections but the SSL certificate does not exist. Handling suggestion: On the <b>Configure Source and Destination Databases</b> page, enable SSL connection for the source database and upload an encryption certificate that contains only one beginning tag <b>BEGIN CERTIFICATE</b> and one end tag <b>END CERTIFICATE</b> .  Failure cause: The SSL certificate type of the source database is not supported. Handling suggestion: On the <b>Configure Source and Destination Databases</b> page, enable SSL connection for the source database and upload an encryption certificate that contains only one beginning tag <b>BEGIN CERTIFICATE</b> and one end tag <b>END CERTIFICATE</b> .

## 1.11.4 Checking Whether the SSL Certificate of the Destination Database Exists

### MySQL

**Table 1-180** Checking whether the SSL certificate of the destination database exists

<b>Check Item</b>	Whether the SSL certificate of the destination database exists
<b>Description</b>	Check whether the SSL certificate type of the destination database is correct during migration. Otherwise, the migration fails.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The SSL certificate of the destination database does not exist. Handling suggestion: On the <b>Configure Source and Destination Databases</b> page, enable SSL connection for the destination database and upload an encryption certificate that contains only one beginning tag <b>BEGIN CERTIFICATE</b> and one end tag <b>END CERTIFICATE</b> .
	Failure cause: The SSL certificate type of the destination database is not supported. Handling suggestion: On the <b>Configure Source and Destination Databases</b> page, enable SSL connection for the destination database and upload an encryption certificate that contains only one beginning tag <b>BEGIN CERTIFICATE</b> and one end tag <b>END CERTIFICATE</b> .

## 1.11.5 Checking Whether Both the Source and Destination Databases Use SSL

### MongoDB Migration

**Table 1-181** Checking whether both the source and destination databases use SSL to encrypt connections

<b>Check Item</b>	Whether both the source and destination databases use SSL to encrypt connections
<b>Description</b>	Check whether both the source and destination databases use SSL to encrypt connections.

<b>Failure Cause and Handling Suggestion</b>	<p>Failure cause: Both the source and destination databases use SSL to encrypt connections.</p> <p>Handling suggestion: When migrating data to the cloud, disable SSL on the destination database. When migrating data out of the cloud, disable SSL on the source database. To disable SSL, go to the DDS basic information page.</p>
--	--

## 1.12 Object Dependencies

### 1.12.1 Checking Whether the Objects Referenced by Views Are Selected for Migration

#### Migration from MongoDB to DDS

**Table 1-182** Checking whether the objects referenced by views are selected for migration

<b>Check Item</b>	Whether the objects referenced by views are selected for migration
<b>Description</b>	The views and referenced objects should be migrated together. Otherwise, the migration fails.
<b>Item to Be Confirmed and Handling Suggestion</b>	<p>Failure cause: The views to be migrated have dependencies on the objects that are not to be migrated.</p> <p>Handling suggestion: Select the referenced objects in the migration object list. Alternatively, do not migrate the dependent views.</p>

## 1.12.2 Checking Whether Referenced Tables Are Selected for Migration

### MySQL Migration and Synchronization

**Table 1-183** Checking whether the tables referenced by the foreign key in the table to be migrated are selected for migration.

<b>Check Item</b>	Whether the tables referenced by the foreign key in the table to be migrated are selected for migration.
<b>Description</b>	The tables referenced by the foreign key in the table to be migrated are not selected for migration.
<b>Item to Be Confirmed and Handling Suggestion</b>	Failure cause: Tables referenced by the foreign key in the table to be migrated are not selected for migration. Handling suggestion: Select the referenced tables.

### MariaDB Synchronization

**Table 1-184** Checking whether the tables referenced by the foreign key in the table to be migrated are selected for migration.

<b>Check Item</b>	Whether the tables referenced by the foreign key in the table to be migrated are selected for migration.
<b>Description</b>	The tables referenced by the foreign key in the table to be migrated are not selected for migration.
<b>Item to Be Confirmed and Handling Suggestion</b>	Failure cause: Tables referenced by the foreign key in the table to be migrated are not selected for migration. Handling suggestion: Select the referenced tables.

## 1.13 Source Database Information

## 1.13.1 Checking Whether the Shards and Mongos Are in the Same Cluster

### MongoDB Migration

**Table 1-185** Checking whether the shards and mongos are in the same cluster

<b>Check Item</b>	Whether the shards and mongos are in the same cluster
<b>Description</b>	If the shards and mongos are not in the same cluster, the migration will fail.
<b>Item to Be Confirmed and Handling Suggestion</b>	Failure cause: The shards are in different clusters from mongos. Handling suggestion: On the connection test page, enter the shards which are in the same cluster as the mongos.

## 1.13.2 Checking Whether the Balancers of the Source Database Is Enabled

### MongoDB Migration

**Table 1-186** Checking whether the balancers of the source database is enabled

<b>Check Item</b>	Whether the balancers of the source database are enabled
<b>Description</b>	If the source database contains the collections whose balancers are enabled, the migration will fail.
<b>Item to Be Confirmed and Handling Suggestion</b>	Failure cause: Balancers are enabled for the collections in the source database. Handling suggestions: Disable balancers for the collections.



## 1.13.3 Checking Whether the Source and Destination Database Types Match

### MongoDB Migration

**Table 1-187** Checking whether the source and destination database types match

<b>Check Item</b>	Whether the source and destination database types match
<b>Description</b>	If the source database type does not match the destination database type, the migration task will fail.
<b>Failure Cause and Handling Suggestion</b>	Failure cause: The source and destination database types do not match. Handling suggestion: If the source DB instance type is cluster, ensure that the object type corresponding to the input IP address and port of the source database cluster is mongos, the source shard database type is replica set, and the destination database type is cluster.

# 2 Failure Cases

## 2.1 Case Overview

Table 2-1 Overview

Data Flow	Related Documents
Real-Time Migration from MongoDB to DDS	Full Migration Error: Prematurely reached end of stream
	Full Migration Error: not authorized on *** to execute command {***}
	Full Migration Error: GC overhead limit exceeded
	Full Migration Error: The background process is unavailable. Maybe it has been killed manually or by the operating system. Please restart the task if possible or wait for restarting by itself within 5 minutes."
	Full Migration Error: Timed out after 60000 ms while waiting to connect
	Full or Incremental Migration Error: Timed out after 60000 ms while waiting to connect
	Full or Incremental Migration Error: Invalid BSON field name ***
	Incremental Migration Error: Timed out after 60000 ms while waiting for a server that matches com.mongodb.client.internal
	Incremental Migration Error: Command failed with error *** (**):***. The full response is {***}"

Data Flow	Related Documents
Real-Time Migration and Synchronization from MySQL to MySQL	Full Phase Error: Table *** doesn't exist
	Full Phase Error: The background process is unavailable
	Full Phase Error: Communications link failure The last packet sent successfully to the server was 0 milliseconds ago. The driver has not received any packets from the server.
	Full Phase Error: Error writing file *** (errno: 28 - No space left on device)
	Full Phase Error: The MySQL server is running with the --super-read-only option so it cannot execute this statement
	Full Phase Error: The table *** is full
	Full Phase Error: Unknown column *** in 'field list'
	Full Phase Error: Lock wait timeout exceeded; try restarting transaction
	Full Phase Error: Java heap space
	Full Phase Error: Table *** already exists
	Full Phase Error: temp table: *** not exist
	Full Phase Error: failed to create new session
	Full Phase Error: load table: *** failed
	Full Phase Error: extract table structure failed!
	Full Phase Error: read table=*** failed
	Full Phase Error: CANNOT UPDATE USER WITH NULL PASSWORD
	Full Phase Error: Access denied for user *** to database ***
	Full Phase Error: The MySQL server is running with the --super-read-only option so it cannot execute this statement
	Full Phase Error: Temporary file write failure.
	Full Phase Error: Incorrect prefix key; the used key part isn't a string, the used length is longer than the key part, or the storage engine doesn't support unique prefix keys
Full Phase Error: Unknown database ***	
Full Phase Error: Access denied; you need (at least one of) the SUPER privilege(s) for this operation	
Full Phase Error: retry structures failed events and Table *** doesn't exist	

Data Flow	Related Documents
	Full Phase Error: shard table=*** failed
	Full Phase Error: error when split table shard occur!
	Full Phase Error: Column name 'AUTO_PK_ROW_ID' is reserved.
	Full Phase Error: transfer account failed, can not find password from src DB
	Full Phase Error: Failed to add the foreign key constraint '****' to system tables
	Full Phase Error: Too many keys specified; max 64 keys allowed
	Full Phase Error: Unknown collation: 'utf8mb4_0900_ai_ci'
	Full or Incremental Phase Error: Access denied for user ***
	Full or Incremental Phase Error: binlog is not existed
	Full or Incremental Phase Error: database log download failed
	Full or Incremental Phase Error: Can not read response from server
	Full or Incremental Phase Error: Communications link failure
	Full or Incremental Phase Error: EOF Packet received, master disconnected
	Full or Incremental Phase Error: Extract db create sql failed
	Full or Incremental Phase Error: load database structure failed in source database
	Full or Incremental Phase Error: load table: *** failed
	Full or Incremental Phase Error: Reached end of input stream
	Full or Incremental Phase Error: Read timed out
	Full or Incremental Phase Error: The background process is unavailable
	Full or Incremental Phase Error: Duplicate entry *** for key 'PRIMARY'
	Full or Incremental Phase Error: cause by: Index: ***, Size: ***
	Full or Incremental Phase Error: The offset and file name between src and parser is inconsistency
	Full or Incremental Phase Error: core process is not healthy or crashed

Data Flow	Related Documents
	Full or Incremental Phase Error: table info of table `***` from metadata miss
	Full or Incremental Phase Error: binlog parse fail, data dictionary may be not complete!
	Full or Incremental Phase Error: table *** record field size for insert/delete dml
	Full or Incremental Phase Error: service *** failed, cause by: Unable to connect to DBMS: ***
	Full or Incremental Phase Error: The binlog fetch connection may be interrupted
	Full or Incremental Phase Error: Received error packet: errno - 1047, solstate - HY000 errmsg = Unknown command
	Incremental Phase Error: not equals to target db column count
	Incremental Phase Error: The MySQL server is running with the --super-read-only option
	Incremental Phase Error: you need (at least one of) the SUPER privilege(s) for this operation
	Incremental Phase Error: Can't DROP ***; check that column/key exists
	Incremental Phase Error: Can't find file: *** (errno: 2 - No such file or directory)
	Incremental Phase Error: Data truncation: Data too long for column
	Incremental Phase Error: Failed to read file header from
	Incremental Phase Error: Lock wait timeout exceeded
	Incremental Phase Error: Must seek before attempting to read next event
	Incremental Phase Error: Table *** already exists
	Incremental Phase Error: Table *** doesn't exist
	Incremental Phase Error: Table *** not found in database
	Incremental Phase Error: source has more columns than target
	Incremental Phase Error: Unknown storage engine
	Incremental Phase Error: Unknown table
	Incremental Phase Error: You have an error in your SQL syntax

Data Flow	Related Documents
	<p><a href="#">Incremental Phase Error: not illegal for mariaDb gtid position</a></p> <p><a href="#">Incremental Phase Error: without PK execute failed</a></p> <p><a href="#">Incremental Phase Error: Deadlock found when trying to get lock</a></p> <p><a href="#">Incremental Phase Error: current serverUUID not equals to this session</a></p> <p><a href="#">Incremental Phase Error: Slave has more GTIDs than the master has, using the master's SERVER_UUID.</a></p> <p><a href="#">Incremental Phase Error: Operation not allowed when innodb_force_recovery &gt; 0</a></p> <p><a href="#">Incremental Phase Error: filter data in config condition filter error</a></p>
<p><a href="#">Real-Time Migration and Synchronization from MySQL to GaussDB(for MySQL)</a></p>	<p><a href="#">Full or Incremental Phase Error: Illegal mix of collations (utf8mb4_0900_ai_ci,IMPLICIT) and (utf8mb4_general_ci,IMPLICIT) for operation</a></p>
<p><a href="#">Real-Time Synchronization from MySQL to GaussDB(DWS)</a></p>	<p><a href="#">Full Synchronization Error: Table *** not found in database</a></p> <p><a href="#">Full Synchronization Error: column 'database_table' of relation *** does not exist</a></p> <p><a href="#">Full Synchronization Error: value too long for type character varying</a></p> <p><a href="#">Full Synchronization Error: int1 has not implemented</a></p> <p><a href="#">Full Synchronization Error: column name 'tid' conflicts with a system column name</a></p> <p><a href="#">Full Synchronization Error: date/time field value out of range</a></p> <p><a href="#">Full or Incremental Synchronization Error: service LOGMANAGER failed</a></p> <p><a href="#">Full or Incremental Synchronization Error: service CAPTURER failed</a></p> <p><a href="#">Full or Incremental Synchronization Error: ERROR: pooler</a></p>

Data Flow	Related Documents
	Full or Incremental Phase Error: service *** failed, cause by: Unable to connect to DBMS: ***
	Full or Incremental Phase Error: The binlog fetch connection may be interrupted
	Incremental Synchronization Error: dn_***: column *** contains null values
	Incremental Synchronization Error: source has more columns than target
	Incremental Synchronization Errors: Connection to *.*.*.98:8000 refused. Check that the hostname and port are correct and that the postmaster is accepting TCP/IP connections
	Incremental Synchronization Error: Table *** not found in target database
	Incremental Synchronization Error: in a read-only transaction
	Incremental Synchronization Error: relation *** does not exist
	Incremental Synchronization Error: *** doesn't in the target table
	Incremental Synchronization Error: syntax error at or near
	Incremental Synchronization Error: schema *** does not exist
	Incremental Synchronization Error: Check whether dws supports the DDL
	Incremental Synchronization Error: PL/pgSQL function *** line *** at SQL statement
Real-Time Synchronization from MySQL to CSS/ES	Incremental Synchronization Error: write table *** failed: null
Real-Time Synchronization from PostgreS QL to PostgreS QL	Full Synchronization Error: function *** does not exist
	Full Synchronization Error: relation *** does not exist
	Full Synchronization Error: GC overhead limit exceeded
	Full Synchronization Error: Java heap space
	Full Synchronization Error: column *** of relation *** does not exist

Data Flow	Related Documents
	Full Synchronization Error: column *** does not exist
	Full Synchronization Error: type 'hstore' does not exist
	Full Synchronization Error: type 'geometry' does not exist
	Full Synchronization Error: Check that the hostname and port are correct and that the postmaster is accepting TCP/IP connections
	Full Synchronization Error: invalid locale name
	Full Synchronization Error: password must not equal user name
	Full Synchronization Error: permission denied for schema ***
	Full or Incremental Phase Error: service *** failed, cause by: Unable to connect to DBMS: ***
	Full or Incremental Phase Error: Initialize logical replication stream failed, the source database may have a long transaction
	Full or Incremental Phase Error: memory required is *** MB, maintenance_work_mem is *** MB
	Full or Incremental Phase Error: temporary file size exceeds temp_file_limit
	Incremental Synchronization Error: Table *** not found in target database
	Incremental Synchronization Error: remaining connection slots are reserved
	Incremental Synchronization Error: PL/pgSQL function *** line *** at SQL statement
Real-Time Synchronization with Oracle Serving as the Source	Full Synchronization Error: has date/datetime: *** which is outside of dest allowed range
	Full or Incremental Phase Error: Got minus one from a read call
	Incremental Synchronization Error: Source supplemental log level is PK/UI. Missing column data at delete+insert on ****
	Incremental Synchronization Error: timeout when get next file log, maybe has been deleted, please check it.
	Incremental Synchronization Error: Failed to construct kafka producer.
	Incremental Synchronization Error: Topic *** not present in metadata after 300000 ms



Data Flow	Related Documents
Real-Time DR with MySQL Serving as the Source	DR Error: A dml without pk write target db fail
Backup Migration	Backup Migration Failed Because Backup Files Cannot Be Found
	Backup Migration Failed Because a Backup Database Cannot Be Found in the Backup Files
	Backup Migration Failed Because the Database with the Same Name Already Exists
	Backup Migration Failed Because an Incremental Backup File Is Used
	Backup Migration Failed Because an Log Backup File Is Used
	Backup Migration Failed Because the Backup File Verification Failed
	Backup Migration Failed Because of Insufficient Space
	Backup Migration Failed Because Database Names Are Not Specified
	Backup Migration Failed Because a Full Backup File Is Used
	Backup Migration Failed Because the LSNs of Incremental Backup Files Are Inconsecutive
Backup Migration Failed Because the Number of Databases to Be Restored Exceeds the Destination Database Threshold	
Workload Replay	Parsing Failed, and a Message Is Displayed Indicating That the OBS Connection Failed
Data-Level Comparison	Data-Level Comparison Error: service SDV failed! cause by: the size of records in one shard[ *** ] of target database, exceeds the max size 200000

## 2.2 Real-Time Migration from MongoDB to DDS

## 2.2.1 Full Migration Error: Prematurely reached end of stream

### Scenarios

During real-time MongoDB-to-DDS migration, an error is reported during full migration. The log information is as follows: service DATAMOVE failed, cause by: Prematurely reached end of stream.

### Possible Causes

The number of connections to the source or destination database is insufficient. Check the maximum number of connections to the source or destination database and the number of used connections. Generally, the number of connections used by DRS is about 10 on the source database and 20 on the destination database.

### Solution

- Step 1** Adjust the number of connections to the database.
- For DDS, query and adjust the number of connections to a database by referring to [DDS User Guide](#).
  - For MongoDB, adjust the number of connections to a database by referring to the official document.
- Step 2** After the adjustment is complete, click **Resume** in the **Operation** column to resume the task.
- Step 3** If the fault persists, choose [Service Tickets > Create Service Ticket](#) in the upper right corner of the management console and contact DRS customer service.

----End

## 2.2.2 Full Migration Error: not authorized on \*\*\* to execute command {\*\*\*}

### Scenarios

During real-time migration from MongoDB to DDS, an error is reported during full migration. The log information is as follows: service DATAMOVE failed, cause by: apply event=[type=table\_data, batch\_index\_in\_shard=1, table\_schema=\*\*\*, table\_name=\*\*\*, record\_num=2720] occur error, msg=Command failed with error 13 (Unauthorized): 'not authorized on \*\*\* to execute command {\*\*\*}.

### Possible Causes

The migration account used by DRS does not have the write permission on the destination database.

### Solution

- Step 1** Grant the destination database write permission to the DRS migration account. For details, see the MongoDB official documents or DDS user guide.

**Step 2** After the adjustment is complete, click **Resume** in the **Operation** column to resume the task.

----End

## 2.2.3 Full Migration Error: GC overhead limit exceeded

### Scenarios

During real-time migration from MongoDB to DDS, an error is reported during full migration. The log information is as follows: service DATAMOVE failed, cause by: GC overhead limit exceeded.

### Possible Causes

- The size of a single data record in the source database is too large.
- The replication instance specifications are too small.

### Solution

**Step 1** Check whether the task is normal.

- If the task is normal, this error is recorded in the log and no further action is required.
- If the task is abnormal, go to **Step 2**.

**Step 2** In the upper right corner of the console, choose **Service Tickets > Create Service Ticket** and contact customer service.

----End

**2.2.4 Full Migration Error: The background process is unavailable. Maybe it has been killed manually or by the operating system. Please restart the task if possible or wait for restarting by itself within 5 minutes."**

### Scenarios

During real-time migration from MongoDB to DDS, an error is reported during full migration. The log information is as follows: service DATAMOVE failed, cause by: The background process is unavailable. Maybe it has been killed manually or by the operating system. Please restart the task if possible or wait for restarting by itself within 5 minutes.

### Possible Causes

- The synchronization process is abnormal.

### Solution

No further operation is required. The DRS daemon process automatically resumes the task. The migration and synchronization services are not affected, and data is

resumed from the breakpoint. If the task is abnormal, click **Resume** in the **Operation** column to resume the task.

## 2.2.5 Full Migration Error: Timed out after 60000 ms while waiting to connect

### Scenarios

During real-time migration from MongoDB to DDS, an error is reported during full migration. The log information is as follows: service DATAMOVE failed, cause by: [reason]:Failed to connect to database due to network, check the network between the DRS and the database or try again later.[message]:apply event=[type=table\_data, batch\_index\_in\_shard=144, table\_schema=\*\*\*, table\_name=\*\*\*, record\_num=8510] occur error, msg=Timed out after 60000 ms while waiting to connect. Client view of cluster state is {type=UNKNOWN, servers=[{\*\*\* type=UNKNOWN, state=CONNECTING, exception={com.mongodb.MongoSocketOpenException: Exception opening socket}, caused by {java.net.ConnectException: Connection refused (Connection refused)}}].

### Possible Causes

- The network is unstable. As a result, the connection times out when data is written to the destination database.
- The destination database is busy. As a result, the connection times out.

### Solution

1. Check whether the destination database is running properly.
2. Check whether packet loss or retransmission occurs on the network between the DRS replication instance and the destination database.
3. Contact Huawei engineers to change the default timeout interval.

## 2.2.6 Full or Incremental Migration Error: Timed out after 60000 ms while waiting to connect

### Scenarios

During real-time migration from MongoDB to DDS, an error is reported during full or incremental migration. The log information is as follows: service LOGMANAGER failed, cause by: [reason]:Failed to connect to database due to network, check the network between the DRS and the database or try again later.[message]:Timed out after 60000 ms while waiting to connect. Client view of cluster state is {type=UNKNOWN, servers=[{\*\*\*, type=UNKNOWN, state=CONNECTING, exception={com.mongodb.MongoSocketOpenException: Exception opening socket}, caused by {java.net.ConnectException: Connection refused (Connection refused)}}]

### Possible Causes

- The network is unstable. As a result, the connection to the source database times out.

- The source database is busy. As a result, the connection times out.

## Solution

1. Check whether the source database is running properly.
2. Check whether packet loss or retransmission occurs on the network between the DRS replication instance and the source database.
3. Contact Huawei engineers to change the default timeout interval.

## 2.2.7 Full or Incremental Migration Error: Invalid BSON field name \*\*\*

### Scenarios

During real-time migration from MongoDB to DDS, an error is reported during full or incremental migration. The log information is as follows: service DATAMOVE failed, cause by: apply event=[type=table\_data, batch\_index\_in\_shard={\*\*\*}, table\_schema={\*\*\*}, table\_name={\*\*\*}, record\_num={\*\*\*}] occur error, msg=Invalid BSON field name {\*\*\*}

### Possible Causes

The field contains invalid characters, such as periods (.) and dollar signs (\$).

### Solution

Check and remove invalid symbols in the source database. In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.

## 2.2.8 Incremental Migration Error: Timed out after 60000 ms while waiting for a server that matches com.mongodb.client.internal

### Scenarios

During real-time migration from MongoDB to DDS, an error is reported during incremental migration. The log information is as follows: service INCREMENT failed, cause by: [reason]:Failed to connect to database due to network, check the network between the DRS and the database or try again later.[message]:Timed out after 60000 ms while waiting for a server that matches com.mongodb.client.internal.MongoClientDelegate\$1@27105e1a. Client view of cluster state is {type=REPLICA\_SET, servers=[{address=\*\*\*, type=UNKNOWN, state=CONNECTING, exception={com.mongodb.MongoSocketOpenException: Exception opening socket}, caused by {java.net.ConnectException: Connection timed out (Connection timed out)}}, {address=\*\*\*, type=UNKNOWN, state=CONNECTING, exception={com.mongodb.MongoSocketOpenException: Exception opening socket}, caused by {java.net.NoRouteToHostException: Exception opening socket}}, {address=\*\*\*, type=UNKNOWN, state=CONNECTING, exception={com.mongodb.MongoSocketOpenException: Exception opening socket}, caused by {java.net.NoRouteToHostException: Exception opening socket}}]}

## Possible Causes

- The network is unstable. As a result, the connection times out when data is written to the destination database.
- The destination database is busy. As a result, the connection times out.

## Solution

1. Check whether the destination database is running properly.
2. Check whether packet loss or retransmission occurs on the network between the DRS replication instance and the destination database.
3. Contact Huawei engineers to change the default timeout interval.

### 2.2.9 Incremental Migration Error: Command failed with error **\*\*\* (\*\*\*):\*\*\***. The full response is **{\*\*\*}**"

#### Scenarios

During real-time migration from MongoDB to DDS, an error is reported during incremental migration. The log information is as follows: service INCREMENT failed, cause by: [reason]:The database returns an error.[message]:Command failed with error **\*\*\* (\*\*\*):\*\*\***. The full response is **{\*\*\*}**.

#### Possible Causes

The destination database returns an error. Common error codes are as follows:

- Error 91: The destination database service is abnormal.
- Error 133: The destination database shard is abnormal.
- Error 10107: The primary node of the destination database is abnormal.

#### Solution

Contact destination database engineers to locate and rectify the fault.

## 2.3 Real-Time Migration and Synchronization from MySQL to MySQL

### 2.3.1 Full Phase Error: Table **\*\*\*** doesn't exist

#### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: Table '\*\*\*' doesn't exist.

## Possible Causes

During the full phase, DDL statements are executed in the source database to delete tables.

## Solution

### Solution 1

During the full migration and synchronization phases, the DELETE command cannot be performed. For details about how to recreate a task, see [From MySQL to MySQL](#).

### Solution 2

Create a table with the same structure as the deleted table in the source database. In the task list on the **Online Migration Management** page, locate the task and click **Resume** in the **Operation** column.

## 2.3.2 Full Phase Error: The background process is unavailable

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: The background process is unavailable. Maybe it has been killed manually or by the operating system. Please restart the task if possible or wait for restarting by itself within 5 minutes.

### Possible Causes

During the full migration or synchronization phase, the DRS process is terminated unexpectedly.

### Solution

No further operation is required. The DRS daemon process automatically resumes the task. The migration and synchronization services are not affected, and data is resumed from the breakpoint. If the task is abnormal, click **Resume** in the **Operation** column to resume the task.

## 2.3.3 Full Phase Error: Communications link failure The last packet sent successfully to the server was 0 milliseconds ago. The driver has not received any packets from the server.

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: Unable to connect to DBMS: url=jdbc:mysql://\*\*\* user=root, Caused by: Communications link failure The last packet sent successfully to the server was 0 milliseconds ago. The driver has not received any packets from the server.

## Possible Causes

The connection to the source or destination database fails to be established.

## Solution

1. Check whether the source or destination database is running properly.
2. Check whether the network connection between the DRS instance and the source or destination database is normal.
3. Check whether the whitelist of the source or destination database allows access from DRS instance IP addresses.

## 2.3.4 Full Phase Error: Error writing file \*\*\* (errno: 28 - No space left on device)

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: apply data of table=` %s`.` %s` failed: Error writing file '\*\*\*' (errno: 28 - No space left on device).

## Possible Causes

The destination database storage space is insufficient. As a result, data fails to be written to the destination database.

## Solution

- Step 1** Adjust the storage space of the destination database.
- If RDS for MySQL is used, see [RDS for MySQL Performance Tuning](#) or contact RDS customer service to adjust the destination database storage space.
  - If an on-premises MySQL database or a MySQL database built on another cloud is used, contact database O&M engineers to check and adjust the destination database storage space.
- Step 2** After the adjustment is complete, click **Resume** in the **Operation** column to resume the task.

----End

## 2.3.5 Full Phase Error: The MySQL server is running with the --super-read-only option so it cannot execute this statement

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: apply data of table=` %s`.` %s` failed: The MySQL server is running with the --super-read-only option so it cannot execute this statement.



## Possible Causes

The destination database is read-only. The possible cause is that the space of the destination database is insufficient.

## Solution

**Step 1** Adjust the storage space of the destination database and restore the destination database to the Read/Write state.

- If RDS for MySQL is used, see [RDS for MySQL Performance Tuning](#) or contact RDS customer service to adjust the destination database storage space.
- If an on-premises MySQL database or a MySQL database built on another cloud is used, contact database O&M engineers to check and adjust the destination database storage space.

**Step 2** After the adjustment is complete, click **Resume** in the **Operation** column to resume the task.

----End

## 2.3.6 Full Phase Error: The table \*\*\* is full

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: apply event=[type=table\_data, batch\_index\_in\_shard=1, table\_schema=%s, table\_name=%s, record\_num=%s,] occur error, msg=apply data of table=`%s`.`%s` failed: The table \*\*\* is full.

### Possible Causes

The destination database storage space is insufficient. As a result, data fails to be written to the destination database.

### Solution

**Step 1** Adjust the storage space of the destination database.

- If RDS for MySQL is used, see [RDS for MySQL Performance Tuning](#) or contact RDS customer service to adjust the destination database storage space.
- If an on-premises MySQL database or a MySQL database built on another cloud is used, contact database O&M engineers to check and adjust the destination database storage space.

**Step 2** After the adjustment is complete, click **Resume** in the **Operation** column to resume the task.

----End

## 2.3.7 Full Phase Error: Unknown column \*\*\* in 'field list'

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: apply event=[type=table\_data, batch\_index\_in\_shard=1, table\_schema= %s, table\_name= %s, record\_num=5] occur error, msg=apply data of table=` %s`.` %s` failed: Unknown column ' %s' in'field list'.

### Possible Causes

The table structures of the source and destination databases are inconsistent. The possible cause is that DDL is executed on the columns of the destination database table during full synchronization or the table consistency check is skipped during pre-check.

### Solution

- Step 1** Contact the O&M personnel to change the table structure of the destination database to be the same as that of the source database.
- Step 2** After the change is complete, click **Resume** in the **Operation** column to resume the task.

----End

## 2.3.8 Full Phase Error: Lock wait timeout exceeded; try restarting transaction

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: apply event=[type=table\_data, batch\_index\_in\_shard=\*\*\*, table\_schema= %s, table\_name= %s, record\_num=\*\*\*] occur error, msg=apply data of table=` %s`.` %s` failed: Lock wait timeout exceeded; try restarting transaction.

### Possible Causes

- The service connection of the destination database holds the lock for a long time.
- The performance of the destination database is insufficient or the load is heavy, and the execution is slow.

### Solution

- Step 1** Contact the O&M personnel to check the lock usage, slow SQL statements, or load status of the destination database.

**Step 2** After the destination database is restored, click **Resume** in the **Operation** column to resume the task.

----End

## 2.3.9 Full Phase Error: Java heap space

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: apply event=[type=table\_data, batch\_index\_in\_shard=\*\*\*, table\_schema= %s, table\_name= %s, record\_num=\*\*\*] occur error, msg=apply data of table=` %s ` ` %s ` failed: Java heap space.

### Possible Causes

The size of a single record exceeds 50 MB.

### Solution

Contact Huawei technical support engineers.

## 2.3.10 Full Phase Error: Table \*\*\* already exists

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: apply event=[type=table\_rename\_or\_copy, index=null, schema\_name= %s, object\_name= %s] occur error, msg=rename table %s. %s error: Table '%s ' already exists.

### Possible Causes

The table without a primary key already exists in the destination database.

### Solution

**Step 1** Contact the O&M engineers to delete the tables that do not have primary keys from the destination database.

**Step 2** In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.

----End

## 2.3.11 Full Phase Error: temp table: \*\*\* not exist

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: apply

```
event=[type=table_rename_or_copy, index=null, schema_name= %s, object_name= %s] occur error, msg=temp table: %s. %s not exist
```

## Possible Causes

An exception occurred when a table without a primary key is being migrated.

## Solution

Contact Huawei technical support engineers.

## 2.3.12 Full Phase Error: failed to create new session

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: com.continuent.tungsten.replicator.ReplicatorException: Unable to connect to DBMS: url=jdbc:mysql://\*\*\* user=\*\*\*, Caused by: failed to create new session.

### Possible Causes

The connection to the source or destination database fails to be established.

### Solution

1. Check whether the source or destination database is running properly.
2. Check whether the network connection between the DRS instance and the source or destination database is normal.
3. Check whether the whitelist of the source or destination database allows access from the IP address of the DRS instance.

## 2.3.13 Full Phase Error: load table: \*\*\* failed

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: load table: `%s`.`%s` failed.

### Possible Causes

The table structure of the source database fails to be loaded. The possible cause is that the user deletes tables during synchronization or the source database user does not have required permissions.

### Solution

1. Contact the O&M engineers to check whether the table structure of the source database is normal. The common commands are as follows:

```
SELECT * FROM `%s`.`%s` LIMIT 1  
SHOW CREATE TABLE `%s`.`%s`
```

2. Contact the source database administrator to check whether the source database and tables have been deleted. If they were deleted, recreate the task by referring to [Precautions](#).
3. Check whether the migration account has the SHOW CREATE TABLE permission on the source database tables. If the account does not have the permission, grant the permission to the source database migration account by referring to [Precautions](#). Then, in the task list, click **Resume** in the **Operation** column to resume the task.

## 2.3.14 Full Phase Error: extract table structure failed!

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by:extract table structure failed! Table is %s. message is %s.

### Possible Causes

The table structure of the source database fails to be loaded. The possible cause is that the user deletes tables during synchronization or the source database user does not have required permissions.

### Solution

1. Contact the O&M engineers to check whether the table structure of the source database is normal. The common commands are as follows:

```
SELECT * FROM `%s`.`%s` LIMIT 1  
SHOW CREATE TABLE `%s`.`%s`
```
2. Contact the source database administrator to check whether the source database and tables have been deleted. If they were deleted, recreate the task by referring to [Precautions](#).
3. Check whether the migration account has the SHOW CREATE TABLE permission on the source database tables. If the account does not have the permission, grant the permission to the source database migration account by referring to [Precautions](#). Then, in the task list, click **Resume** in the **Operation** column to resume the task.

## 2.3.15 Full Phase Error: read table=\*\*\* failed

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: read table=` %s`.` %s` failed.

### Possible Causes

Failed to read table data from the source database due to poor source database performance or unstable network connection.

## Solution

- Step 1** Contact Huawei technical support to adjust the timeout interval for accessing the source database.
  - Step 2** After the adjustment is complete, click **Resume** in the **Operation** column to resume the task.
- End

## 2.3.16 Full Phase Error: CANNOT UPDATE USER WITH NULL PASSWORD

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: retry structures failed events=the fail structures are [type=account, index=0, schema\_name=mysql, object\_name='\*\*\*']reason:[CANNOT UPDATE USER WITH NULL PASSWORD].

### Possible Causes

The source database account password is empty.

### Solution

- Step 1** Contact the O&M engineers to add a password for the account that reports the error in the source database.
  - Step 2** In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.
- End

## 2.3.17 Full Phase Error: Access denied for user \*\*\* to database \*\*\*

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: retry structures failed events=the fail structures are [type=account, index=0, schema\_name=mysql, object\_name='\*\*\*']reason:[Access denied for user '\*\*\*' to database '\*\*\*']

### Possible Causes

The DRS migration account does not have sufficient permissions on the destination database.

### Solution

- Step 1** Contact the O&M engineers to add the schema permission to the migration account in the destination database.

**Step 2** In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.

----End

## 2.3.18 Full Phase Error: The MySQL server is running with the --super-read-only option so it cannot execute this statement

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: retry structures failed events=the fail structures are [type=constraint\_data, index=0, schema\_name= %s, object\_name= %s]reason:[The MySQL server is running with the --super-read-only option so it cannot execute this statement]

### Possible Causes

When DRS migrates indexes, the destination database is in the read-only state. The possible cause is that the space of the destination database is insufficient.

### Solution

**Step 1** Contact the O&M engineers to check the status of the destination database and rectify the database fault.

**Step 2** After the destination database is restored, click **Resume** in the **Operation** column to resume the task.

----End

## 2.3.19 Full Phase Error: Temporary file write failure.

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: retry structures failed events=the fail structures are [type=constraint\_data, index=0, schema\_name= %s, object\_name= %s]reason:[Temporary file write failure.]

### Possible Causes

The temporary space of the destination database is insufficient when DRS migrates indexes.

### Solution

**Step 1** Adjust the temporary space of the destination database.

- If an RDS for MySQL instance is used, perform the following operations to adjust the temporary space:
  - a. Scale up the space by referring to [RDS for MySQL Performance Tuning](#).

- b. Check whether the temporary space increases.
    - If yes, go to [Step 2](#).
    - If no, in the upper right corner of the management console, choose [Service Tickets > Create Service Tickets](#) and contact RDS customer service to adjust the temporary space of the destination database.
  - If an on-premises MySQL database or a MySQL database built on another cloud is used, contact database O&M engineers to check and adjust the destination database temporary space.
- Step 2** After the adjustment is complete, click **Resume** in the **Operation** column to resume the task.
- Step 3** If the fault persists, choose [Service Tickets > Create Service Ticket](#) in the upper right corner of the management console and contact DRS customer service.

----End

## 2.3.20 Full Phase Error: Incorrect prefix key; the used key part isn't a string, the used length is longer than the key part, or the storage engine doesn't support unique prefix keys

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: retry structures failed events=the fail structures are [type=constraint\_data, index=106, schema\_name= %s, object\_name= %s]reason:[Incorrect prefix key; the used key part isn't a string, the used length is longer than the key part, or the storage engine doesn't support unique prefix keys]

### Possible Causes

Table structures in the source and destination databases are inconsistent.

### Solution

- Step 1** Contact the O&M engineers to change the table structure of the destination database to be the same as that of the source database.
- Step 2** In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.

----End

## 2.3.21 Full Phase Error: Unknown database \*\*\*

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: retry structures failed events=the fail structures are [type=\*\*\*, index=\*\*\*, schema\_name=\*\*\*, object\_name=\*\*\*]reason:[Unknown database '\*\*\*']



## Possible Causes

- The database corresponding to the destination database structure does not exist.
- The database in which objects to be migrated or synchronized reside is not in the object selection list.

## Solution

- Step 1** Check whether the database specified in the error information exists in the destination database.
- If no, manually create a specified database in the destination database and ensure that the structure of the database is the same as that of the source database. Then, in the task list, locate the row that contains the target task and click **Resume** in the **Operation** column to submit the task again.
  - If yes, go to [Step 2](#).
- Step 2** Check whether the database specified in the error information exists in the source database.
- If yes, select the database again.
  - If no, the database in which objects reside may have been deleted or DRS does not have the permission to read the database. In this case, objects cannot be migrated. Re-create the task and do not select the objects that reside in the deleted database.

----End

## 2.3.22 Full Phase Error: Access denied; you need (at least one of) the SUPER privilege(s) for this operation

### Scenarios

During a full migration, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: retry structures failed events=the fail structures are [type=function, index=2, schema\_name= %s, object\_name= %s]reason:[Access denied; you need (at least one of) the SUPER privilege(s) for this operation]

### Possible Causes

The user retained the definer for migration, but the migration account does not have the super permission.

### Solution

Grant the super permission to the destination database user and submit the task again. For details, see [RDS FAQs](#). Alternatively, you can choose not to migrate definers when recreating a task.

## 2.3.23 Full Phase Error: retry structures failed events and Table \*\*\* doesn't exist

### Scenarios

During a full migration, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: retry structures failed events=the fail structures are [type=view, index=2, schema\_name= %s, object\_name= %s]reason: [Table ' %s. %s' doesn't exist]

### Possible Causes

The user retained the definer for migration, but the definer is abnormal or does not exist.

### Solution

Recreate a task and do not migrate definers.

## 2.3.24 Full Phase Error: shard table=\*\*\* failed

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: shard table=`%s`.`%s` failed

### Possible Causes

The source database performance is insufficient or the network is unstable. As a result, the source database sharding times out.

### Solution

**Step 1** Check whether the task is normal.

- If the task is normal, this error is recorded in the log and no further action is required.
- If the task is abnormal, go to [Step 2](#).

**Step 2** In the upper right corner of the console, choose [Service Tickets > Create Service Ticket](#) and contact DRS customer service.

----End

## 2.3.25 Full Phase Error: error when split table shard occur!

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: error when split table shard occur! Table is %s .Error code is %s.

## Possible Causes

The source database performance is insufficient or the network is unstable. As a result, the source database sharding times out.

## Solution

**Step 1** Check whether the task is normal.

- If the task is normal, this error is recorded in the log and no further action is required.
- If the task is abnormal, go to [Step 2](#).

**Step 2** In the upper right corner of the console, choose [Service Tickets > Create Service Ticket](#) and contact DRS customer service.

----End

## 2.3.26 Full Phase Error: Column name 'AUTO\_PK\_ROW\_ID' is reserved.

### Scenarios

An error is reported during full migration or synchronization, and the following log information is as follows: service LOGMANAGER failed, cause by: create table \*\*\* error. Column name 'AUTO\_PK\_ROW\_ID' is reserved. Operation 'CREATE' is not permitted.

## Possible Causes

The **AUTO\_PK\_ROW\_ID** column name is a reserved column name for the RDS for MySQL database and cannot be created by users.

## Solution

- Check the tables whose column names contain **AUTO\_PK\_ROW\_ID** in the source database, change the column names, and resume the task.
- Create a task again and do not select the tables whose column names contain **AUTO\_PK\_ROW\_ID**.

## 2.3.27 Full Phase Error: transfer account failed, can not find password from src DB

### Scenarios

During a full migration, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: transfer account failed, can not find password from src DB.

## Possible Causes

The RDS security policy does not allow the user password to be empty. However, if the source database is an on-premises MySQL database, the user password can be empty.

## Solution

**Step 1** Run the following SQL statement in the source database to query users whose passwords are empty:

- MySQL 5.7 or later versions:  

```
SELECT USER,HOST,authentication_string FROM MYSQL.user WHERE authentication_string IS NULL OR authentication_string="";
```
- MySQL 5.6 and earlier versions  

```
SELECT USER,HOST,`password` FROM MYSQL.user WHERE `password` IS NULL OR `password`="";
```

**Step 2** Run the following SQL statement to delete the users whose passwords are empty from the source database or set passwords for the users:

- Delete a user whose password is empty.  

```
DROP USER ***@***;
```
- Set a password for a user.  

```
ALTER USER ***@*** IDENTIFIED BY ***;
```

----End

## 2.3.28 Full Phase Error: Failed to add the foreign key constraint '\*\*\*' to system tables

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: Failed to add the foreign key constraint '\*\*\*' to system tables

### Possible Causes

During the full phase, the destination database has foreign key constraints with the same name.

### Solution

1. Run the following SQL statement to delete or rename the foreign key constraints with the same name in the destination database:  

```
select * from information_schema.REFERENTIAL_CONSTRAINTS where CONSTRAINT_NAME = "foreign_key_name";
```
2. In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.

## 2.3.29 Full Phase Error: Too many keys specified; max 64 keys allowed

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: retry structures failed events=the fail structures are [type=constraint\_data, index=0, schema\_name=DB, object\_name=TABLE]reason:[Too many keys specified; max 64 keys allowed]

### Possible Causes

A maximum of 64 secondary indexes can be created for a single MySQL table. The total number of new and existing indexes in the destination database exceeds 64.

### Solution

1. After manually creating the required indexes in the destination database, contact DRS O&M personnel to skip the migration of secondary indexes in the table.
2. In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.

## 2.3.30 Full Phase Error: Unknown collation: 'utf8mb4\_0900\_ai\_ci'

### Scenarios

During a full migration or synchronization, an error is reported, and the log information is as follows: retry structures failed events=the fail structures are [type=table\_structure, index=0, schema\_name=m825, object\_name=t1]reason: [Unknown collation: 'utf8mb4\_0900\_ai\_ci']

### Possible Causes

The source database version is later than the destination database version, or the source database is of a special type and supports the utf8mb4\_0900\_ai\_ci collation, but the destination MySQL database does not support this collation. The DRS task fails to synchronize the table structure because the destination database does not support the collation.

### Solution

- Step 1** Manually create a table structure in the destination database and modify the collation rule.
- Step 2** Create a DRS task again, deselect the table structure for synchronization, and perform full synchronization again.

----End

## 2.3.31 Full or Incremental Phase Error: Access denied for user

\*\*\*

### Scenarios

During a full or incremental migration or synchronization, an error is reported. The log information is as follows: service %s failed, cause by: Unable to connect to DBMS: url=\*\*\*?useUnicode=true&allowLoadLocalInfile=false&characterEncoding=UTF-8&connectTimeout=5000&useSSL=false&allowPublicKeyRetrieval=true&verifyServerCertificate=false&serverTimezone=UTC user=%s, Caused by: Access denied for user %s

### Possible Causes

The connection to the source or destination database fails to be established.

### Solution

1. Check whether the source or destination database is running properly.
2. Check whether the password for connecting to the source or destination database is correct.
3. Check whether the network connection between the DRS instance and the source or destination database is normal.
4. Check whether the IP address of the DRS instance is allowed to access the source or destination database.

## 2.3.32 Full or Incremental Phase Error: binlog is not existed

### Scenarios

During a full or incremental migration or synchronization, an error is reported, and the log information is as follows: service LOGMANAGER failed, cause by: binlog is not existed

### Possible Causes

The binlog files in the source database were deleted. DRS cannot obtain logs from the source database.

### Solution

Recreate a DRS task.

## 2.3.33 Full or Incremental Phase Error: database log download failed

### Scenarios

During a full or incremental migration or synchronization, an error is reported, and the log information is as follows: service LOGMANAGER failed, cause by: database log download failed, error code is %s.

### Possible Causes

The binlog files in the source database were deleted. DRS cannot obtain logs from the source database.

### Solution

Recreate a DRS task.

## 2.3.34 Full or Incremental Phase Error: Can not read response from server

### Scenarios

During a full or incremental migration or synchronization, an error is reported, and the log information is as follows: service LOGMANAGER failed, cause by: Can not read response from server. Expected to read 4 bytes, read 0 bytes before connection was unexpectedly lost.

### Possible Causes

- The network bandwidth between the DRS replication instance and the source database is too small or unstable.
- The source database is overloaded.

### Solution

Contact the source database O&M personnel to check the source database load and check whether packet loss occurs on the network between the source database and the replication instance.

## 2.3.35 Full or Incremental Phase Error: Communications link failure

### Scenarios

During a full or incremental migration or synchronization, an error is reported, and the log information is as follows: service %s failed, cause by: Communications link failure The last packet sent successfully to the server was 0 milliseconds ago. The driver has not received any packets from the server.

## Possible Causes

The connection to the source or destination database fails to be established.

## Solution

1. Check whether the source or destination database is running properly.
2. Check whether the network connection between the DRS instance and the source or destination database is normal.
3. Check whether the IP address of the DRS instance is allowed to access the source or destination database.

## 2.3.36 Full or Incremental Phase Error: EOF Packet received, master disconnected

### Scenarios

During a full or incremental migration or synchronization, an error is reported, and the log information is as follows: service LOGMANAGER failed, cause by: EOF Packet received, master disconnected

### Possible Causes

- The source database is abnormal.
- The binlog file is abnormal.

### Solution

**Step 1** Contact the source database O&M personnel to check whether the source database is running properly.

**Step 2** After the source database is restored, click **Resume** in the **Operation** column to resume the task.

----End

## 2.3.37 Full or Incremental Phase Error: Extract db create sql failed

### Scenarios

During a full or increment migration or synchronization, an error is reported, and the log information is as follows: service LOGMANAGER failed, cause by: Extract db create sql failed, dbName = %s

### Possible Causes

- During full migration, the user deleted databases from the source.
- The source database user does not have the permission to perform operations on the source database.



## Solution

- Contact the source database administrator to check whether the source database has been deleted. If the source database was deleted, recreate the task by referring to the related section in the product documentation.
- Check whether the source database user has the SHOW CREATE TABLE permission on the source database table. If the user does not have the permission, grant the permission to the user and retry the DRS task.

## 2.3.38 Full or Incremental Phase Error: load database structure failed in source database

### Scenarios

During a full or incremental migration or synchronization, an error is reported, and the log information is as follows: service LOGMANAGER failed, cause by: load database structure failed in source database, The failed %s is: type=%s, schema\_name=%s, object\_name=%s, errorcode=%s, message=%s"

### Possible Causes

- During full migration, the user deleted databases from the source.
- The source database user does not have the permission to perform operations on the source database.

### Solution

- Contact the source database administrator to check whether the source database has been deleted. If the source database was deleted, recreate the task by referring to the related section in the product documentation.
- Check whether the source database user has the SHOW CREATE TABLE permission on the source database table. If the user does not have the permission, grant the permission to the user and retry the DRS task.

## 2.3.39 Full or Incremental Phase Error: load table: \*\*\* failed

### Scenarios

During a full or incremental migration or synchronization, an error is reported, and the log information is as follows: service LOGMANAGER failed, cause by: load table: ` %s`.` %s` failed

### Possible Causes

- During full migration, the user deleted databases or tables from the source.
- The source database user does not have the permission to perform operations on the source database or table.

### Solution

1. Contact the source database administrator to check whether the databases and tables in the source database have been deleted. If they were deleted, recreate the task by referring to [Precautions](#).

2. Check whether the migration account has the SHOW CREATE TABLE permission on the source database tables. If the account does not have the permission, grant the permission to the source database migration account by referring to [Precautions](#). Then, in the task list, click **Resume** in the **Operation** column to resume the task.

## 2.3.40 Full or Incremental Phase Error: Reached end of input stream

### Scenarios

During a full or incremental migration or synchronization, an error is reported, and the log information is as follows: service LOGMANAGER failed, cause by: Reached end of input stream

### Possible Causes

- The source database is abnormal.
- The binlog file is abnormal.

### Solution

- Step 1** Contact the source database administrator to check whether the source database is running properly.
- Step 2** After the source database is restored, click **Resume** in the **Operation** column to resume the task.

----End

## 2.3.41 Full or Incremental Phase Error: Read timed out

### Scenarios

During a full or incremental migration or synchronization, an error is reported, and the log information is as follows: service LOGMANAGER failed, cause by: Read timed out

### Possible Causes

The possible causes are as follows:

- Failed to connect to the source database.

### Solution

1. Check whether the source database is running properly.
2. Check whether the network connection between the DRS instance and the source database is normal.

## 2.3.42 Full or Incremental Phase Error: The background process is unavailable

### Scenarios

During a full or incremental migration or synchronization, an error is reported, and the log information is as follows: service %s failed, cause by: The background process is unavailable. Maybe it has been killed manually or by the operating system. Please restart the task if possible or wait for restarting by itself within 5 minutes.

### Possible Causes

During the migration, the DRS process stops unexpectedly.

### Solution

No further operation is required. The DRS daemon process automatically resumes the task. The migration and synchronization services are not affected, and data is resumed from the breakpoint. If the task is abnormal, click **Resume** in the **Operation** column to resume the task.

## 2.3.43 Full or Incremental Phase Error: Duplicate entry \*\*\* for key 'PRIMARY'

### Scenarios

During a full or incremental migration or synchronization, an error is reported, and the log information is as follows: service CAPTURER failed, cause by: Duplicate entry '120' for key 'PRIMARY'.

### Possible Causes

- **binlog\_format** in the source database is not set to **ROW**.
- The **binlog\_format** setting of the source database does not take effect immediately.

### Solution

**Step 1** Log in to the source database using the MySQL official client or other tools.

**Step 2** Run the following command for setting global parameters in the source database.  

```
set global binlog_format = ROW;
```

**Step 3** Run the following command on the source database and check whether the preceding operation is successful:  

```
select @@global.binlog_format;
```

**Step 4** You can use either of the following methods to ensure that the modified binlog format of the source database takes effect immediately:

#### Method 1

1. Select a non-service period to disconnect all service connections on the current database.
  - a. Run the following command to query all service threads (excluding all binlog dump threads and current threads) in the current database:

```
show processlist;
```
  - b. Stop all the service threads queried in the previous step.

**NOTE**

Do not create or start a migration task before the preceding operations are complete. Otherwise, data may be inconsistent.

2. To prevent the binlog format of the source database from becoming invalid due to database restart, add or modify the **binlog\_format** parameter in the startup configuration file (**my.ini** or **my.cnf**) of the source database and save the modification.

```
binlog_format=ROW
```

**Method 2**

1. To prevent the binlog format of the source database from becoming invalid due to database restart, add or modify the **binlog\_format** parameter in the startup configuration file (**my.ini** or **my.cnf**) of the source database and save the modification.

```
binlog_format=ROW
```
2. Ensure that the **binlog\_format** parameter is successfully added or modified. Then, restart the source database at a non-service period.

----End

## 2.3.44 Full or Incremental Phase Error: cause by: Index: \*\*\*, Size: \*\*\*

### Scenarios

During a full or incremental migration or synchronization, an error is reported, and the log information is as follows: service CAPTURER failed, cause by: Index: 8, Size: 7

### Possible Causes

- **binlog\_format** in the source database is not set to **ROW**.
- The **binlog\_format** setting of the source database does not take effect immediately.

### Solution

- Step 1** Log in to the source database using the MySQL official client or other tools.
- Step 2** Run the following command for setting global parameters in the source database.

```
set global binlog_format = ROW;
```
- Step 3** Run the following command on the source database and check whether the preceding operation is successful:

```
select @@global.binlog_format;
```

**Step 4** You can use either of the following methods to ensure that the modified binlog format of the source database takes effect immediately:

#### Method 1

1. Select a non-service period to disconnect all service connections on the current database.
  - a. Run the following command to query all service threads (excluding all binlog dump threads and current threads) in the current database:  
`show processlist;`
  - b. Stop all the service threads queried in the previous step.

#### NOTE

Do not create or start a migration task before the preceding operations are complete. Otherwise, data may be inconsistent.

2. To prevent the binlog format of the source database from becoming invalid due to database restart, add or modify the **binlog\_format** parameter in the startup configuration file (**my.ini** or **my.cnf**) of the source database and save the modification.  
`binlog_format=ROW`

#### Method 2

1. To prevent the binlog format of the source database from becoming invalid due to database restart, add or modify the **binlog\_format** parameter in the startup configuration file (**my.ini** or **my.cnf**) of the source database and save the modification.  
`binlog_format=ROW`
2. Ensure that the **binlog\_format** parameter is successfully added or modified. Then, restart the source database at a non-service period.

----End

## 2.3.45 Full or Incremental Phase Error: The offset and file name between src and parser is inconsistency

### Scenarios

During a full or incremental migration or synchronization, an error is reported, and the log information is as follows: service CAPTURER failed, cause by: The offset and file name between src and parser is inconsistency

### Possible Causes

- The source database is abnormal.
- The binlog file is abnormal.

### Solution

**Step 1** Contact the source database administrator to check whether the source database is running properly.

**Step 2** After the source database is restored, click **Resume** in the **Operation** column to resume the task.

----End

## 2.3.46 Full or Incremental Phase Error: core process is not healthy or crashed

### Scenarios

During a full or incremental migration or synchronization, an error is reported, and the log information is as follows: service CAPTURER failed, cause by: core process is not healthy or crashed

### Possible Causes

During the migration, the DRS process stops unexpectedly.

### Solution

No further operation is required. The DRS daemon process automatically resumes the task. The migration and synchronization services are not affected, and data is resumed from the breakpoint. If the task is abnormal, click **Resume** in the **Operation** column to resume the task.

## 2.3.47 Full or Incremental Phase Error: table info of table `\*\*\*` from metadata miss

### Scenarios

During a full or incremental migration or synchronization, an error is reported, and the log information is as follows: service CAPTURER failed, cause by: table info of table `%s`.`%s` from metadata miss

### Possible Causes

The table may fail to be created due to DDL syntax incompatibility.

### Solution

**Step 1** Check whether the task is normal.

- If the task is normal, this error is recorded in the log and no further action is required.
- If the task is abnormal, go to [Step 2](#).

**Step 2** In the upper right corner of the console, choose [Service Tickets > Create Service Ticket](#) and contact customer service.

----End

## 2.3.48 Full or Incremental Phase Error: binlog parse fail, data dictionary may be not complete!

### Scenarios

During a full or incremental migration or synchronization, an error is reported, and the log information is as follows: service CAPTURER failed, cause by: tbinlog parse fail, data dictionary may be not complete! tableName: %s, databaseName:%s

### Possible Causes

The table may fail to be created due to DDL syntax incompatibility.

### Solution

Contact Huawei technical support engineers.

## 2.3.49 Full or Incremental Phase Error: table \*\*\* record field size for insert/delete dml

### Scenarios

During a full or incremental migration or synchronization, an error is reported, and the log information is as follows: service CAPTURER failed, cause by: table[%s.%s]record field size for insert/delete dml=%s, the column size in dictionary=%s

### Possible Causes

Full image is not enabled for the source database binlog.

### Solution

For details, see [How Do I Set binlog\\_row\\_image=FULL to Take Effect Immediately?](#)

## 2.3.50 Full or Incremental Phase Error: service \*\*\* failed, cause by: Unable to connect to DBMS: \*\*\*

### Scenarios

During a full or incremental migration or synchronization, an error is reported, and the log information is as follows: service \*\*\* failed, cause by: Unable to connect to DBMS: \*\*\*

### Possible Causes

The connection to the source or destination database fails to be established.

## Solution

1. Check whether the source or destination database is running properly.
2. Check whether the network connection between the DRS instance and the source or destination database is normal.
3. Check whether the IP address of the DRS instance is allowed to access the source or destination database.

### 2.3.51 Full or Incremental Phase Error: The binlog fetch connection may be interrupted

#### Scenarios

During a full or incremental migration or synchronization, an error is reported, and the log information is as follows: The binlog fetch connection may be interrupted

#### Possible Causes

DRS is disconnected from the source database to obtain binlogs. The possible cause is that the source database status has changed or the network is abnormal.

#### Solution

1. Check whether the source database is running properly.
2. Check whether the network connection between the DRS instance and the source database is normal.
3. In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.

### 2.3.52 Full or Incremental Phase Error: Received error packet: errno - 1047, solstate - HY000 errmsg = Unknown command

#### Scenarios

During a full or incremental migration or synchronization, an error is reported, and the log information is as follows: service LOGMANAGER failed, cause by: Received error packet: errno - 1047, solstate - HY000 errmsg = Unknown command

#### Possible Causes

DRS failed to obtain binlogs. The source database may be a MySQL proxy node whose binlogs cannot be obtained.

#### Solution

1. Edit the DRS task and replace the source database with a node whose binlogs can be obtained.
2. In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.



## 2.3.53 Incremental Phase Error: not equals to target db column count

### Scenarios

During an incremental migration or synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: write table %s. %s failed: table= %s. %s has field list size=[\*\*\*] not equals to target db column count= %s

### Possible Causes

DDL is executed on the destination database table, causing the table structure in the destination database to be inconsistent with that in the source database.

### Solution

- Step 1** Contact the destination database O&M engineers to change the table structure of the destination database to be the same as that of the source database.
- Step 2** In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.

----End

## 2.3.54 Incremental Phase Error: The MySQL server is running with the --super-read-only option

### Scenarios

During an incremental migration or synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: write table %s. %s failed: record tid: %s,seqno: %s with PK applied failed in table %s. %s, The MySQL server is running with the --super-read-only option so it cannot execute this statement

### Possible Causes

The destination database is in the read-only state. Generally, the destination database storage is insufficient.

### Solution

- Step 1** Contact the O&M engineers to check the running status and disk space of the destination database.
- Step 2** After the destination database is restored, click **Resume** in the **Operation** column to resume the task.

----End

## 2.3.55 Incremental Phase Error: you need (at least one of) the SUPER privilege(s) for this operation

### Scenarios

During an incremental migration, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: Access denied; you need (at least one of) the SUPER privilege(s) for this operation; sql is: CREATE DEFINER= %s

### Possible Causes

The user retained the definer for migration, but the definer is abnormal or does not exist.

### Solution

Recreate a task and do not migrate definers.

## 2.3.56 Incremental Phase Error: Can't DROP \*\*\*; check that column/key exists

### Scenarios

During an incremental migration or synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: Can't DROP ' %s'; check that column/key exists; sql is %s

### Possible Causes

DDL is executed on the destination database table, causing the table structure in the destination database to be inconsistent with that in the source database.

### Solution

**Step 1** Contact the destination database O&M engineers to change the table structure of the destination database to be the same as that of the source database.

**Step 2** In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.

----End

## 2.3.57 Incremental Phase Error: Can't find file: \*\*\* (errno: 2 - No such file or directory)

### Scenarios

During an incremental migration or synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: Can't find file: ' %s' (errno: 2 - No such file or directory); sql is: %s

## Possible Causes

The destination database table file is damaged.

## Solution

- Step 1** Contact the destination database O&M engineers to check whether the corresponding table exists and is normal.
  - Step 2** In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.
- End

## 2.3.58 Incremental Phase Error: Data truncation: Data too long for column

### Scenarios

During an incremental migration or synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: Data truncation: Data too long for column ' %s' at row %s; sql is: %s

## Possible Causes

The DDL statement fails to be executed because data is too long.

## Solution

- Step 1** Contact the destination database O&M engineers to check the structure of the table where the synchronization error is reported, adjust the length of the column where the error is reported, and adjust the column data type in the destination database.
  - Step 2** In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.
- End

## 2.3.59 Incremental Phase Error: Failed to read file header from

### Scenarios

During an incremental migration or synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: Failed to read file header from thl.data.0000000011

## Possible Causes

The format of the DRS data file is damaged.

## Solution

- Step 1** Check whether the task is normal.
- If the task is normal, this error is recorded in the log and no further action is required.
  - If the task is abnormal, go to [Step 2](#).
- Step 2** In the upper right corner of the console, choose [Service Tickets > Create Service Ticket](#) and contact customer service.
- End

### 2.3.60 Incremental Phase Error: Lock wait timeout exceeded

#### Scenarios

During an incremental migration or synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: Lock wait timeout exceeded; try restarting transaction

#### Possible Causes

The lock wait times out when the destination database is accessed.

#### Solution

- Step 1** Contact destination database O&M engineers to check the status and load of the destination database.
- Step 2** In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.
- End

### 2.3.61 Incremental Phase Error: Must seek before attempting to read next event

#### Scenarios

During an incremental migration or synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: Extract THL file fail! Must seek before attempting to read next event

#### Possible Causes

The task is interrupted for a long time. Historical DRS data files were deleted and the task cannot be continued.

#### Solution

Contact the user to re-create the task.

## 2.3.62 Incremental Phase Error: Table \*\*\* already exists

### Scenarios

During an incremental migration or synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: Table ' %s' already exists

### Possible Causes

A table has been created in the destination database. As a result, an error is reported when the DDL statement for creating a table in the source database is executed.

### Solution

**Step 1** Contact destination database O&M engineers to delete the corresponding table from the destination database.

**Step 2** In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.

----End

## 2.3.63 Incremental Phase Error: Table \*\*\* doesn't exist

### Scenarios

During an incremental migration or synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: Table ' %s' doesn't exist; sql is: create %s like matches

### Possible Causes

Tables are deleted from the destination database. As a result, the synchronization statement reports an error.

### Solution

**Step 1** Contact destination database O&M engineers to create a table in the destination database based on the table structure of the source database.

**Step 2** After the table is created, click **Resume** in the **Operation** column to resume the task.

----End

## 2.3.64 Incremental Phase Error: Table \*\*\* not found in database

### Scenarios

During an incremental migration or synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: Table %s not found in database

### Possible Causes

The possible causes are as follows:

- Tables are deleted from the destination database. As a result, the synchronization statement reports an error.
- **Incremental DDLs** is not selected for **Synchronize** during task creation. After a table is created in the source database, DDL statements are filtered out. As a result, an error is reported during synchronization.

### Solution

Contact the destination database O&M engineers to create a table in the destination database based on the table structure of the source database. After the table is created, click **Resume** in the **Operation** column to resume the task.

## 2.3.65 Incremental Phase Error: source has more columns than target

### Scenarios

During an incremental migration or synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: Check table structure consistency fail! Table %s in source has more columns than target

### Possible Causes

The table structure is modified in the destination database. As a result, the synchronization statement reports an error.

### Solution

**Step 1** Contact destination database O&M engineers to change the table structure of the destination database to be the same as that of the source database.

**Step 2** After the change is complete, click **Resume** in the **Operation** column to resume the task.

----End

## 2.3.66 Incremental Phase Error: Unknown storage engine

### Scenarios

During an incremental migration or synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: Unknown storage engine 'FEDERATED'; sql is: %s

### Possible Causes

When the DDL table creation statement of the source database is replayed in destination database, the source DB engine is not supported by the destination database.

### Solution

- Step 1** Contact the user to create a table that supports the destination DB engine in the source database.
- Step 2** Contact Huawei technical support to skip the DDL statement that reports the error.
- Step 3** In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.

----End

## 2.3.67 Incremental Phase Error: Unknown table

### Scenarios

During an incremental migration or synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: Unknown table '%s'; sql is %s

### Possible Causes

The table does not exist in the destination database.

### Solution

- Step 1** Contact the user to create the table in the destination database based on the table structure of the source database.
- Step 2** In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.

----End

## 2.3.68 Incremental Phase Error: You have an error in your SQL syntax

### Scenarios

During an incremental migration or synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'START TRANSACTION' at line 39

### Possible Causes

During the pre-check, the system skips version check. The syntax of the later version fails to be executed in the earlier version.

### Solution

- Step 1** Contact the user to modify the statement based on the destination database syntax and run the statement in the destination database.
- Step 2** Contact Huawei engineers to skip this error.
- Step 3** In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.

----End

## 2.3.69 Incremental Phase Error: not illegal for mariaDb gtid position

### Scenarios

During an incremental migration or synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: %s not illegal for mariaDb gtid position

### Possible Causes

The gtid mode is changed during task creation.

### Solution

Contact the user to recreate the task.

## 2.3.70 Incremental Phase Error: without PK execute failed

### Scenarios

During an incremental migration or synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: record tid: %s, seqno: %s without PK execute failed in table %s



## Possible Causes

A conflict occurs during data synchronization for tables that do not have primary keys.

## Solution

**Step 1** Check whether the task is normal.

- If the task is normal, this error is recorded in the log and no further action is required.
- If the task is abnormal, go to [Step 2](#).

**Step 2** In the upper right corner of the console, choose [Service Tickets > Create Service Ticket](#) and contact customer service.

----End

## 2.3.71 Incremental Phase Error: Deadlock found when trying to get lock

### Scenarios

During an incremental migration or synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: record tid: %s,seqno: %s with PK applied failed in table %s, Deadlock found when trying to get lock; try restarting transaction

## Possible Causes

A deadlock occurs in the destination database.

## Solution

**Step 1** Contact destination database O&M engineers to check the status and load of the destination database.

**Step 2** In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.

----End

## 2.3.72 Incremental Phase Error: current serverUUID not equals to this session

### Scenarios

During an incremental migration or synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: write table %s failed: current serverUUID not equals to this session

## Possible Causes

The destination database had a switchover.

## Solution

**Step 1** Contact destination database O&M engineers to check the destination database status.

**Step 2** In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.

----End

## 2.3.73 Incremental Phase Error: Slave has more GTIDs than the master has, using the master's SERVER\_UUID.

### Scenarios

During an incremental migration or synchronization, an error is reported, and the log information is as follows: service LOGMANAGER failed, cause by: Received error packet: errno = 1236, sqlstate = HY000 errmsg = Slave has more GTIDs than the master has, using the master's SERVER\_UUID. This may indicate that the end of the binary log was truncated or that the last binary log file was lost, e.g., after a power or disk failure when sync\_binlog != 1. The master may or may not have rolled back transactions that were already replicated to the slave. Suggest to replicate any transactions that master has rolled back from slave to master, and/or commit empty transactions on master to account for transactions that have been.

### Possible Causes

The source database position is rolled back, or the source database position is reset by running the **reset master** command.

### Solution

In the task list, locate the target task and click **Reset** in the **Operation** column to reset the task. Alternatively, create a DRS task again.

## 2.3.74 Incremental Phase Error: Operation not allowed when innodb\_force\_recovery > 0

### Scenarios

During an incremental migration or synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: write table \*\*\* failed: Operation not allowed when innodb\_force\_recovery > 0.

## Possible Causes

The destination DB instance is abnormal. When the system variable **innodb\_force\_recovery** is set to be greater than **0** in the destination database, the INSERT, UPDATE, and DELETE operations are disabled in the destination database.

## Solution

- Step 1** Contact destination database O&M engineers to check the destination database status.
- Step 2** After the destination database is restored, click **Resume** in the **Operation** column to resume the task.

----End

## 2.3.75 Incremental Phase Error: filter data in config condition filter error

### Scenarios

During an incremental migration or synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: filter data in config condition filter error!

### Possible Causes

The data filtering conditions of the DRS synchronization task are incorrectly configured. As a result, the incremental data fails to be filtered.

### Solution

Filtering rules cannot be modified for tables that have been synchronized. Create a synchronization task again.

## 2.4 Real-Time Migration and Synchronization from MySQL to GaussDB(for MySQL)

### 2.4.1 Full or Incremental Phase Error: Illegal mix of collations (utf8mb4\_0900\_ai\_ci,IMPLICIT) and (utf8mb4\_general\_ci,IMPLICIT) for operation

### Scenarios

During a full or incremental migration or synchronization, an error is reported, and the log information is as follows: Illegal mix of collations (utf8mb4\_0900\_ai\_ci,IMPLICIT) and (utf8mb4\_general\_ci,IMPLICIT) for operation

## Possible Causes

The sorting rule of the source MySQL 5.\* character set utf8mb4 is utf8mb4\_general\_ci, and that of the destination GaussDB(for MySQL) character set utf8mb4 is utf8mb4\_0900\_ai\_ci. An error is reported, indicating that the sorting rules are inconsistent.

## Solution

- Solution 1
  - a. Run the SQL statement in the destination database to change the character set sorting rule of the corresponding column to utf8mb4\_0900\_ai\_ci. For example, to change the character set sorting rule of column **c1** in table **test\_collation\_1** to utf8mb4\_0900\_ai\_ci, run the following command:

```
ALTER TABLE test_collation_1 MODIFY COLUMN c1 VARCHAR(16) COLLATE utf8mb4_0900_ai_ci;
```
  - b. In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.
- Solution 2
  - a. Delete all columns containing collate utf8mb4\_general\_ci from the source database table.
  - b. In the task list, locate the target task and click **Reset** in the **Operation** column to reset the task. Alternatively, create a DRS task again.
- Solution 3
  - a. Run the SQL statement in the destination database to change the character set sorting rule of the destination database to utf8mb4\_0900\_ai\_ci.

```
SET GLOBAL default_collation_for_utf8mb4='utf8mb4_general_ci';
```
  - b. In the task list, locate the target task and click **Reset** in the **Operation** column to reset the task. Alternatively, create a DRS task again.

## 2.5 Real-Time Synchronization from MySQL to GaussDB(DWS)

### 2.5.1 Full Synchronization Error: Table \*\*\* not found in database

#### Scenarios

During full synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: apply event=[type=table\_data, batch\_index\_in\_shard=1, table\_schema= %s, table\_name= %s, record\_num= %s] occur error, msg=Table %s. %s not found in database. Unable to generate a valid statement.

## Possible Causes

The destination database table is not created. As a result, the synchronization statement reports an error.

## Solution

**Step 1** Contact destination database O&M engineers to create the corresponding table in the destination database.

**Step 2** After the table is created, click **Resume** in the **Operation** column to resume the task.

----End

## 2.5.2 Full Synchronization Error: column 'database\_table' of relation \*\*\* does not exist

### Scenarios

During full synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: apply event=[type=table\_data, batch\_index\_in\_shard= %s, table\_schema= %s, table\_name= %s, record\_num= %s] occur error, msg=apply to %s failed: ERROR: column 'database\_table' of relation ' %s' does not exist.

## Possible Causes

No additional column is added when a table is created in the destination database.

## Solution

**Step 1** Contact destination database O&M engineers to add additional columns to the destination database table.

**Step 2** In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.

----End

## 2.5.3 Full Synchronization Error: value too long for type character varying

### Scenarios

During full synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: apply event=[type=table\_data, batch\_index\_in\_shard= %s, table\_schema= %s, table\_name= %s, record\_num= %s] occur error, msg=apply to %s failed: ERROR: value too long for type character varying( %s) Where: COPY %s, line 1, column remarks.

## Possible Causes

The length of the varchar column in the user-defined table in the destination database is insufficient. MySQL and GaussDB(DWS) define the length of the varchar data type differently. MySQL defines the length of the varchar data type as the number of characters, while GaussDB(DWS) defines the length of the varchar data type as the number of bytes.

## Solution

**Step 1** Contact destination database O&M engineers to increase the field precision in the destination database table.

**Step 2** In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.

----End

## 2.5.4 Full Synchronization Error: int1 has not implemented

### Scenarios

During full synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: apply event=[type=table\_data, batch\_index\_in\_shard= %s, table\_schema= %s, table\_name= %s, record\_num= %s] occur error, msg=apply to %s failed: column: %s, invalid value: %s, int1 has not implemented, type is %s.

### Possible Causes

User-defined table fields in the destination database do not support int1.

### Solution

**Step 1** Contact destination database O&M engineers to create a table in the destination database based on the data flow mappings.

**Step 2** After the table is created, click **Resume** in the **Operation** column to resume the task.

----End

## 2.5.5 Full Synchronization Error: column name 'tid' conflicts with a system column name

### Scenarios

During full synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: retry structures failed events=the fail structures are [type=table\_structure, index=%s, schema\_name=%s, object\_name=%s]reason:[ERROR: column name 'tid' conflicts with a system column name]

## Possible Causes

tid is a reserved character of GaussDB(DWS) and exists in the source database table.

## Solution

**Step 1** This scenario is not supported due to the restrictions of the destination database. Change the column names of the source database table.

**Step 2** After the change is complete, click **Resume** in the **Operation** column to resume the task.

----End

## 2.5.6 Full Synchronization Error: date/time field value out of range

### Scenarios

During full synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: retry structures failed events=the fail structures are [type=table\_structure, index= %s, schema\_name= %s, object\_name= %s]reason:[ERROR: date/time field value out of range: '0000-00-00 00:00:00'].

### Possible Causes

GaussDB(DWS) does not support 0000-00-00 00:00:00.

### Solution

**Step 1** This scenario is not supported due to the restrictions of the destination database. Change the column names of the source database table.

**Step 2** After the change is complete, click **Resume** in the **Operation** column to resume the task.

----End

## 2.5.7 Full or Incremental Synchronization Error: service LOGMANAGER failed

### Scenarios

During full or incremental synchronization, an error is reported, and the log information is as follows: service LOGMANAGER failed, \*\*\*.

### Solution

For synchronization from MySQL to GaussDB(DWS), the cause of the LOGMANAGER error is the same as that of the MySQL to MySQL synchronization. You can rectify the fault based on the error keyword.

- **Full Phase Error: Column name 'AUTO\_PK\_ROW\_ID' is reserved.**
- **Full or Incremental Phase Error: database log download failed**
- **Full or Incremental Phase Error: Can not read response from server**
- **Full or Incremental Phase Error: EOF Packet received, master disconnected**
- **Full or Incremental Phase Error: load database structure failed in source database**
- **Full or Incremental Phase Error: Reached end of input stream**
- **Full or Incremental Phase Error: Read timed out**

## 2.5.8 Full or Incremental Synchronization Error: service CAPTURER failed

### Scenarios

During full or incremental synchronization, an error is reported, and the log information is as follows: service CAPTURER failed, \*\*\*.

### Solution

For synchronization from MySQL to GaussDB(DWS), the cause of the CAPTURER error is the same as that of the MySQL to MySQL synchronization. You can rectify the fault based on the error keyword.

- **Full or Incremental Phase Error: Duplicate entry \*\*\* for key 'PRIMARY'**
- **Full or Incremental Phase Error: cause by: Index: \*\*\*, Size: \*\*\***
- **Full or Incremental Phase Error: The offset and file name between src and parser is inconsistency**
- **Full or Incremental Phase Error: core process is not healthy or crashed**
- **Full or Incremental Phase Error: table info of table `\*\*\*` from metadata miss**
- **Full or Incremental Phase Error: table \*\*\* record field size for insert/delete dml**

## 2.5.9 Full or Incremental Synchronization Error: ERROR: pooler

### Scenarios

During full or incremental synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: %s failed:tid: %s, sqno: %s, ERROR: pooler: %s.

### Possible Causes

The GaussDB(DWS) database returns an error.



## Solution

Contact the destination GaussDB(DWS) database O&M engineers to rectify the fault.

### 2.5.10 Full or Incremental Phase Error: service \*\*\* failed, cause by: Unable to connect to DBMS: \*\*\*

#### Scenarios

During a full or incremental synchronization, an error is reported, and the log information is as follows: service \*\*\* failed, cause by: Unable to connect to DBMS: \*\*\*

#### Possible Causes

The connection to the source or destination database fails to be established.

#### Solution

1. Check whether the source or destination database is running properly.
2. Check whether the network connection between the DRS instance and the source or destination database is normal.
3. Check whether the IP address of the DRS instance is allowed to access the source or destination database.

### 2.5.11 Full or Incremental Phase Error: The binlog fetch connection may be interrupted

#### Scenarios

During a full or incremental synchronization, an error is reported, and the log information is as follows: The binlog fetch connection may be interrupted

#### Possible Causes

DRS is disconnected from the source database to obtain binlogs. The possible cause is that the source database status has changed or the network is abnormal.

#### Solution

1. Check whether the source database is running properly.
2. Check whether the network connection between the DRS instance and the source database is normal.
3. In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.

## 2.5.12 Incremental Synchronization Error: dn\_\*\*\*: column \*\*\* contains null values

### Scenarios

During incremental synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: table %s execute the ddl failed ERROR: dn\_ %s\_ %s: column ' %s' contains null values.

### Possible Causes

The destination GaussDB(DWS) database has the NOT NULL constraint, so the default value needs to be set.

### Solution

Modify the DDL statement and manually run it in the destination database. Then contact Huawei engineers to skip the synchronization of this DDL statement.

## 2.5.13 Incremental Synchronization Error: source has more columns than target

### Scenarios

During incremental synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: Check table structure consistency fail! Table %s in source has more columns than target, the columns is= [%s, %s]

### Possible Causes

- The filter conditions of DDL statements for adding columns in the source database are not synchronized to the destination database.
- As a result, columns are deleted in the destination database.

### Solution

**Step 1** Supplement the missing columns in the destination database based on the table structure of the source database.

**Step 2** In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.

----End

## 2.5.14 Incremental Synchronization Errors: Connection to \*.\*.98:8000 refused. Check that the hostname and port are correct and that the postmaster is accepting TCP/IP connections

### Scenarios

During incremental synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: Unable to connect to DBMS: url=jdbc:dws://\*.\*.115:8000,\*.\*.98:8000,\*.\*.133:8000/tanzhou\_prod?client\_encoding=UTF-8&loadBalanceHosts=true&targetServerType=any&rewriteBatchedInserts=true&socketTimeout=600&connectTimeout=300&binaryTransfer=true&ssl=false&sslmode=prefer user=dbadmin, Caused by: Connection to \*.\*.98:8000 refused. Check that the hostname and port are correct and that the postmaster is accepting TCP/IP connections

### Possible Causes

The connection to the source or destination database fails to be established.

### Solution

1. Check whether the source or destination database is running properly.
2. Check whether the network connection between the DRS replication instance and the source or destination database is normal.
3. Check whether the IP address of the DRS instance is allowed to access the source or destination database.

## 2.5.15 Incremental Synchronization Error: Table \*\*\* not found in target database

### Scenarios

During incremental synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: Check table structure consistency fail! Table %s not found in target database

### Possible Causes

- The filter conditions of DDL statements for creating tables in the source database are not synchronized to the destination database.
- As a result, tables are deleted from the destination database.
- Some DDL statements (such as rename) are filtered out during online DDL.

### Solution

- For the first two causes, create a table in the destination database based on the table structure in the source database.
- If some DDLs are filtered out, rename the temporary table to the correct table name in the destination database.

## 2.5.16 Incremental Synchronization Error: in a read-only transaction

### Scenarios

During incremental synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: insert %s failed:tid: %s, sqno: %s, ERROR: dn\_ %s\_ %s: cannot execute INSERT in a read-only transaction.

### Possible Causes

The destination database is in read-only mode. The possible cause is that the disk space is full.

### Solution

- Step 1** Contact GaussDB (DWS) O&M personnel to restore the destination database.
- Step 2** In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.

----End

## 2.5.17 Incremental Synchronization Error: relation \*\*\* does not exist

### Scenarios

During incremental synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: insert %s failed:tid: %s, sqno: %s, ERROR: relation ' %s' does not exist.

### Possible Causes

- The filter conditions of the DDL statement for creating tables in the source database are not executed in the destination database during synchronization. As a result, the tables do not exist in the destination database.
- The tables are deleted from the destination database.
- Filtering conditions are set for DDL statements during synchronization. Some DDL statements (such as rename) are filtered out. As a result, the destination database table does not exist.

### Solution

- For the first two causes, create a table in the destination database based on the table structure in the source database.
- If some DDL statements are filtered out, rename the temporary table to the correct table name in the destination database.

## 2.5.18 Incremental Synchronization Error: \*\*\* doesn't in the target table

### Scenarios

During incremental synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: insert %s failed: %s doesn't in the target table: %s.

### Possible Causes

- The filter conditions of DDL statements for adding columns in the source database are not synchronized to the destination database.
- Columns are deleted in the destination database.

### Solution

**Step 1** Supplement the missing columns in the destination database based on the table structure of the source database.

**Step 2** In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.

----End

## 2.5.19 Incremental Synchronization Error: syntax error at or near

### Scenarios

During incremental synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: Check whether dws supports the DDL. For support, please contact dws Services. DDL: CREATE TABLE \*\*\* like \*\*\*. failed by:ERROR: syntax error at or near '%s'

### Possible Causes

The DDL statement conversion in the source database does not comply with the syntax of the destination database.

### Solution

DRS does not convert the syntax for incremental DDL synchronization from MySQL to GaussDB(DWS). The DDL executed in the MySQL database will be executed in GaussDB(DWS). Contact Huawei technical support.

## 2.5.20 Incremental Synchronization Error: schema \*\*\* does not exist

### Scenarios

During incremental synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: update %s failed: ERROR: schema ' %s' does not exist.

### Possible Causes

- The filter conditions of DDL statements for adding tables in the source database are not synchronized to the destination database.
- Schemas are deleted from the destination database.

### Solution

**Step 1** Create schemas in the destination database based on the source database structure.

**Step 2** In the task list, locate the target task and click **Resume** in the **Operation** column to resume the task.

----End

## 2.5.21 Incremental Synchronization Error: Check whether dws supports the DDL

### Scenarios

During incremental synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: Check whether dws supports the DDL. For support, please contact dws Services. DDL:alter table gltest01.t\_pk4 add column a3 int after c1. failed by:ERROR: FIRST/AFTER is not yet supported.

### Possible Causes

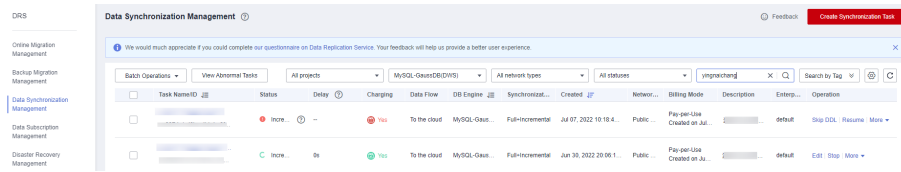
The destination GaussDB(DWS) instance does not support a specified DDL statement.

### Solution

**Step 1** Contact GaussDB(DWS) technical support to execute this statement with the same semantics in the destination database and the statement must comply with the syntax of the destination database.

**Step 2** After the statement is executed on the destination database, click **Skip DDL** in the **Operation** column to skip the error.

Figure 2-1 Skip DDL



----End

## 2.5.22 Incremental Synchronization Error: PL/pgSQL function \*\*\* line \*\*\* at SQL statement

### Scenarios

During incremental synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: \*\*\* failed:tid: \*\*\*, sqno: \*\*\*, ERROR: dn\_\*\*\*\_\*\*\*: concurrent update under Stream mode is not yet supported here: SQL statement \*\*\* PL/pgSQL function \*\*\* line \*\*\* at SQL statement.

### Possible Causes

A trigger exists in the destination database. After data is written to DRS, the trigger automatically starts to execute some SQL operations. However, an error is reported during the execution.

### Solution

Disable or delete the trigger and wait until the DRS task restores.

## 2.6 Real-Time Synchronization from MySQL to CSS/ES

### 2.6.1 Incremental Synchronization Error: write table \*\*\* failed: null

### Scenarios

During incremental synchronization, an error is reported, and the log information is as follows: service INCREMENT failed cause by: write table \*\*\* failed: null

### Possible Causes

During incremental synchronization, the SSL connection is disabled for the destination database. As a result, the DRS task fails.

### Solution

**Step 1** Enable the SSL connection for the destination database.

**Step 2** In the task list, locate the target synchronization task and click **Resume** in the **Operation** column.

----End

## 2.7 Real-Time Synchronization from PostgreSQL to PostgreSQL

### 2.7.1 Full Synchronization Error: function \*\*\* does not exist

#### Scenarios

During full synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: apply event=[type=table\_structure, index=%s, schema\_name=%s, object\_name=%s] occur error, msg=ERROR: function \*\*\* does not exist Hint: No function matches the given name and argument types. You might need to add explicit type casts.

#### Possible Causes

Functions on which the table structure depends are not created in the destination database in advance. In table-level synchronization from PostgreSQL to PostgreSQL, functions and plugin objects cannot be synchronized. Therefore, you need to manually create functions on which the table structure depends in the destination database.

#### NOTE

You can log in to the corresponding database in the destination RDS for PostgreSQL and run the following SQL statement to check whether there is the function. In the command, *f\_name* indicates the function name.

```
select n.nspname,p.proname,pg_get_functiondef(p.oid) as funcdef from pg_proc p left join pg_namespace n on p.pronamespace=n.oid where proname = 'f_name';
```

#### Solution

The missing function may belong to a plugin or is a user-defined function. Perform the following steps to check the source of the function in the source database, create the corresponding plugin or function in the destination database, and retry the DRS task.

**Step 1** Log in to the source database and run the following SQL statement to query the plugin to which the function belongs (*f\_name* indicates the function name):

```
select extname, nspname, proname,pg_get_function_arguments(c.oid) as funcargs from pg_extension e join pg_depend d on (d.refobjid=e.oid) join pg_proc c on (d.objid=c.oid) join pg_namespace n on c.pronamespace=n.oid where proname = 'f_name';
```

- If a query result is displayed, the function belongs to a plugin. The **extname** field in the query result indicates the plugin name. Go to [Step 2](#).
- If no query result is displayed, the function does not belong to any plugin and is a user-defined function. Go to [Step 3](#).



**Step 2** If the function belongs to a plugin, click **Plugins** on the destination RDS for PostgreSQL management page and install the plugin.

**Step 3** If the function is user-defined function, create the same function in the destination database as that in the source database. For details about the function definition statement, see the execution result of the following SQL statement in the source database. *f\_name* indicates the function name.

```
select n.nspname,p.proname,pg_get_functiondef(p.oid) as funcdef from pg_proc p left join pg_namespace n on p.pronamespace=n.oid where proname ='f_name';
```

**Step 4** Retry the DRS task.

----End

## 2.7.2 Full Synchronization Error: relation \*\*\* does not exist

### Scenarios

During full synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: ERROR: relation '%s' does not exist Position: 15

### Possible Causes

During table-level synchronization, objects with dependencies are not synchronized. For example, the source database contains tables A and B and table A depends on table B, but only table A is synchronized.

### Solution

**Step 1** Clear data in the destination database.

**Step 2** Create a synchronization task again and select the objects to be synchronized and all dependent objects.

**Step 3** Start the synchronization task.

----End

## 2.7.3 Full Synchronization Error: GC overhead limit exceeded

### Scenarios

During full synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: GC overhead limit exceeded.

### Possible Causes

Too many large objects exist in the service. As a result, the memory usage of the synchronization task exceeds the threshold.

### Solution

Contact Huawei technical support.

## 2.7.4 Full Synchronization Error: Java heap space

### Scenarios

During full synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: Java heap space.

### Possible Causes

A large number of fields exist in the service. As a result, the memory usage of the synchronization task exceeds the threshold.

### Solution

- Step 1** Check whether the task is normal.
- If the task is normal, this error is recorded in the log and no further action is required.
  - If the task is abnormal, go to [Step 2](#).
- Step 2** In the upper right corner of the console, choose [Service Tickets > Create Service Ticket](#) and contact customer service.

----End

## 2.7.5 Full Synchronization Error: column \*\*\* of relation \*\*\* does not exist

### Scenarios

During full synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: apply event=[type=table\_data, batch\_index\_in\_shard= %s, table\_schema= %s, table\_name= %s, record\_num= %s] occur error, msg=apply table %s data failed: %s: ERROR: column '%s' of relation '%s' does not exist Position: 1043 Call getNextException to see other errors in the batch.

### Possible Causes

- During the full synchronization, DDL operations are executed in the destination database. As a result, the table structure in the destination database is inconsistent with that in the source database.
- During the full synchronization, DDL operations are executed in the source database. As a result, the table structure in the destination database is inconsistent with that in the source database.

Contact the customer to confirm whether they executed DDL operations.

### Solution

Create a synchronization task again. During the full synchronization, ensure that no DDL operation is executed on the source database and no data is written to

the destination database. Otherwise, data may be inconsistent or the synchronization may fail.

## 2.7.6 Full Synchronization Error: column \*\*\* does not exist

### Scenarios

During full synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: retry structures failed events=the fail structures are [type=function, index=0, schema\_name= %s, object\_name=' %s']reason:[ERROR: column ' %s' does not exist Position: %s].

### Possible Causes

- During the full synchronization, DDL operations are executed in the destination database. As a result, the table structure in the destination database is inconsistent with that in the source database.
- During the full synchronization, DDL operations are executed in the source database. As a result, the table structure in the destination database is inconsistent with that in the source database.

Contact the customer to confirm whether they executed DDL operations.

### Solution

Create a synchronization task again. During the full synchronization, ensure that no DDL operation is executed on the source database and no data is written to the destination database. Otherwise, data may be inconsistent or the synchronization may fail.

## 2.7.7 Full Synchronization Error: type 'hstore' does not exist

### Scenarios

During full synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: retry structures failed events=the fail structures are [type=operator, index=2, schema\_name=public, object\_name=?]reason:[ERROR: type 'hstore' does not exist].

### Possible Causes

The hstore plug-in is not installed on the destination database.

#### NOTE

Run the following SQL statement in the destination RDS PostgreSQL database:

```
select * from pg_extension where extname = 'hstore';
```

### Solution

Extensions are not synchronized. Before synchronization, install the corresponding extension in the destination database. Perform the following steps to install the extension and retry the DRS task:

**Step 1** Log in to the destination RDS PostgreSQL database as the **root** user.

**Step 2** Run the following SQL statements to install hstore:

```
create extension "hstore";
```

**Step 3** Retry the DRS task.

----End

## 2.7.8 Full Synchronization Error: type 'geometry' does not exist

### Scenarios

During full synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: retry structures failed events=the fail structures are [type=operator, index=2, schema\_name=public, object\_name=?]reason:[ERROR: type 'geometry' does not exist].

### Possible Causes

The postgis plug-in is not installed on the destination database.

#### NOTE

Run the following SQL statement in the destination RDS for PostgreSQL database:

```
select * from pg_extension where extname = 'postgis';
```

### Solution

Extensions are not synchronized. Before synchronization, install the corresponding extension in the destination database. Perform the following steps to install the extension and retry the DRS task:

**Step 1** Log in to the destination RDS for PostgreSQL database as the **root** user.

**Step 2** Run the following SQL statements to install postgis:

```
create extension "postgis";
```

**Step 3** Retry the DRS task.

----End

## 2.7.9 Full Synchronization Error: Check that the hostname and port are correct and that the postmaster is accepting TCP/IP connections

### Scenarios

During full synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: source engine postgresql client initialize failed, detail: Unable to connect to DBMS: url= %s/position3.0? client\_encoding=UTF-8&ssl=false&sslmode=prefer user= %s, Caused by:

Connection to %s refused. Check that the hostname and port are correct and that the postmaster is accepting TCP/IP connections.

## Possible Causes

The connection to the source or destination database fails to be established.

## Solution

Perform the following operations:

1. Check whether the source or destination database is running properly.
2. Check whether the DRS instance IP address is allowed by the listening port of the source or destination database.
  - For DRS tasks performed over a public network, the source database must allow access from the DRS instance EIP, and the destination database must allow access from the private IP address of the DRS instance.
  - For DRS tasks performed in a VPC, VPN, or Direct Connect network, both the source and destination databases must allow access from the private IP addresses of DRS instance.

## 2.7.10 Full Synchronization Error: invalid locale name

### Scenarios

During full synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: target engine postgresql client initialize failed, detail: Unable to connect to DBMS: url= %s/position3.0? client\_encoding=UTF-8&autosave=always&stringtype=unspecified&ssl=false&sslmode=prefer user= %s, Caused by: ERROR: invalid locale name:'Chinese (Simplified)\_China.936'.

### Possible Causes

The source database region type is not supported by the destination database.

### Solution

Contact the customer to check whether the region type can be changed to another one (UTF-8 by default). The region type may affect the sorting rules of different languages. If the encoding format can be changed to UTF-8, contact Huawei technical support.

## 2.7.11 Full Synchronization Error: password must not equal user name

### Scenarios

During full synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: retry structures failed events=the fail

structures are [type=account, index=0, schema\_name=dummy, object\_name=%s]reason:[ERROR: password must not equal user name].

## Possible Causes

For RDS for PostgreSQL synchronization, the password cannot be the same as the username. If the password is the same as the username in the source database, an error will be reported when data is synchronized to the destination database.

## Solution

Manually create the user in the destination database and click **Resume** on the DRS task management page to continue the synchronization.

## 2.7.12 Full Synchronization Error: permission denied for schema \*\*\*

### Scenarios

During full synchronization, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: retry structures failed events=the fail structures are [type=table\_structure, index=0, schema\_name=%s, object\_name=%s]reason:[ERROR: permission denied for schema %s]

### Possible Causes

The destination database user does not have the create permission on the schema.

### Solution

**Step 1** Run the following SQL statement in the destination database to grant the create permission on the schema to which the table owner belongs:

```
grant create on schema <schema_name> to <table_owner_in_source>;
```

**Step 2** On the **Data Synchronization Management** page, click **Resume** to resume the synchronization task.

----End

## 2.7.13 Full or Incremental Phase Error: service \*\*\* failed, cause by: Unable to connect to DBMS: \*\*\*

### Scenarios

During a full or incremental synchronization, an error is reported, and the log information is as follows: service \*\*\* failed, cause by: Unable to connect to DBMS: \*\*\*

### Possible Causes

Failed to connect to the source or destination database.

## Solution

1. Check whether the source or destination database is running properly.
2. Check whether the network connection between the DRS instance and the source or destination database is normal.
3. Check whether the IP address of the DRS instance is allowed to access the source or destination database.

## 2.7.14 Full or Incremental Phase Error: Initialize logical replication stream failed, the source database may have a long transaction

### Scenarios

During a full or incremental synchronization, an error is reported, and the log information is as follows: service LOGMANAGER failed, cause by: Initialize logical replication stream failed, the source database may have a long transaction: \*\*\*

### Possible Causes

A logical replication slot fails to be created in the source database.

### Solution

**Step 1** Check whether the number of replication slots in the source database reaches the upper limit. If yes, delete replication slots that are no longer used from the source database or increase the value of **max\_replication\_slots** and restart the source database.

- Run the following command to query the number of logical replication slots:  
`select count(1) from pg_replication_slots;`
- Run the following command to query the maximum number of logical replication slots:  
`select setting as number from pg_settings where name = 'max_replication_slots';`

**Step 2** Check whether the source database has long transactions that are not submitted. If yes, slot creation times out. As a result, the task fails.

- Run the following command to query a transaction status:  
`select pid, datname, state, backend_xid, xact_start, (now() - xact_start) as cost from pg_stat_activity where backend_xid is not null order by xact_start;`
- Run the following command to stop a long transaction:  
`select pg_terminate_backend(pid);`

----End

## 2.7.15 Full or Incremental Phase Error: memory required is \*\*\* MB, maintenance\_work\_mem is \*\*\* MB

### Scenarios

During a full or incremental synchronization, an error is reported, and the log information is as follows: retry structures failed events=the fail structures are

[type=index, index=\*\*\*, schema\_name=\*\*\*, object\_name=\*\*\*]reason:[ERROR: memory required is \*\*\* MB, maintenance\_work\_mem is \*\*\* MB]

## Possible Causes

When an index is created in the destination database, the required memory is greater than the value of **maintenance\_work\_mem** configured for the database.

## Solution

- Step 1** Change the value of **maintenance\_work\_mem** in the destination database RDS for PostgreSQL to an appropriate value. For details, see [Modifying RDS for PostgreSQL Instance Parameters](#).
- Step 2** Restart the database to apply the change. Then, On the **Data Synchronization Management** page, locate the target task and click **Resume** in the **Operation** column to resume the task.

----End

## 2.7.16 Full or Incremental Phase Error: temporary file size exceeds temp\_file\_limit

### Scenarios

During a full or incremental synchronization, an error is reported, and the log information is as follows: retry structures failed events=the fail structures are [type=index, index=0, schema\_name=fossbot, object\_name=scan\_mr5\_file\_union]reason:[ERROR: temporary file size exceeds temp\_file\_limit (20000000kB)]

## Possible Causes

The size of the temporary table generated during SQL execution exceeds the upper limit of the temporary tablespace in the system.

## Solution

- Step 1** Increase the value of the **temp\_file\_limit** parameter in the destination database by referring to [Modifying RDS for PostgreSQL Instance Parameters](#)
- Step 2** Restart the database to apply the change. Then, On the **Data Synchronization Management** page, locate the target task and click **Resume** in the **Operation** column. After the synchronization task is complete, change the value to the original value. Otherwise, the DB instance disk may be full due to large temporary tablespace.

----End



## 2.7.17 Incremental Synchronization Error: Table \*\*\* not found in target database

### Scenarios

During incremental synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: Check table structure consistency fail! Table %s not found in target database

### Possible Causes

- The user did not select to synchronize DDL, so the CREATE TABLE statement is not synchronized to the destination database.
- The user selected to synchronize DDL, but the source database uses the DDL statement that is not supported by DRS to create a table.
- The table is deleted from the destination database.

### Solution

- Method 1: Create a task again and comply with the following DRS usage rules:
  - If you select to synchronize DDL, do not execute DDL statements that are not supported by DRS in the source database.
  - If you do not synchronize DDL, do not execute DDL statements in the source database, or execute DDL statements in the destination database before executing the same DDL statements in the source database.
  - During full and incremental synchronization, do not write data to the destination database. Otherwise, data may be inconsistent or the synchronization may fail.
- Method 2: Perform the following operations to restore the DRS task:
  - **Possible cause:** The user did not select to synchronize DDL, so the CREATE TABLE statement in the source database is not synchronized to the destination database.  
**Solution:** Create a table in the destination database based on the table structure of the source database and retry the DRS task.
  - **Possible cause:** The user selected to synchronize DDL, but the source database uses the DDL statement that is not supported by DRS to create a table.  
**Solution:** Create a table in the destination database based on the table structure of the source database and retry the DRS task.
  - **Possible cause:** The table is deleted from the destination database.  
**Solution:** Re-create the table in the destination database based on the structure of the deleted table and retry the DRS task.

#### NOTE

If both the table and the data in the table are deleted, re-creating the table may lead to data inconsistency or cause the task to fail again.

## 2.7.18 Incremental Synchronization Error: remaining connection slots are reserved

### Scenarios

During incremental synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: Unable to connect to DBMS: url= %s user= %s, Caused by: FATAL: remaining connection slots are reserved for non-replication superuser connections.

### Possible Causes

The number of user connections to the destination database reaches the upper limit. As a result, the connection to the destination database fails to be established.

#### NOTE

Log in to the destination RDS PostgreSQL database and run the following SQL statement:

- View max\_connections.  

```
show max_connections;
```
- Check the current number of connections.  

```
select count(*) from pg_stat_activity;
```

### Solution

On the destination RDS PostgreSQL database console, change the value of **max\_connections** to a larger value and make it take effect. Each DRS task requires about 100 connections.

## 2.7.19 Incremental Synchronization Error: PL/pgSQL function \*\*\* line \*\*\* at SQL statement

### Scenarios

During incremental synchronization, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: insert %s failed:tid: %s, sqno: %s, ERROR: %s' PL/pgSQL function %s line %s at SQL statement.

### Possible Causes

The destination database **session\_replication\_role** is not set to **replica**, and the destination database trigger is not disabled.

#### NOTE

You can log in to the destination database RDS PostgreSQL and run the following SQL statement to check the value of **session\_replication\_role**:

```
show session_replication_role;
```

## Solution

**Step 1** On the RDS PostgreSQL console, change the value of `session_replication_role` to `replica` and apply the changes.

**Step 2** Retry the DRS task.

----End

## 2.8 Real-Time Synchronization with Oracle Serving as the Source

### 2.8.1 Full Synchronization Error: has date/datetime: \*\*\* which is outside of dest allowed range

#### Scenarios

During a full synchronization from Oracle to MySQL, an error is reported, and the log information is as follows: service DATAMOVE failed, cause by: `java.lang.InterruptedExcep`tion: Database: `***`, Table: `***`, Column: `***` has date/datetime: `*** 0:0:0` which is outside of dest allowed range.

#### Possible Causes

Oracle and MySQL heterogeneous databases support different time types. You can run the following SQL statement in the Oracle database to view data:

```
select to_char(column_name, 'YYYY-MM-DD') from table_name;
```

#### Solution

Contact Huawei technical support. After the fault is rectified, DRS writes data based on the following rules:

- If the destination database stores DATE data, 0000-01-01 00:00:00 is written.
- If the destination database stores TIMESTAMP data, 1970-01-01 00:00:01 is written.

### 2.8.2 Full or Incremental Phase Error: Got minus one from a read call

#### Scenarios

During a full or incremental synchronization with Oracle serving as the source, an error is reported, and the log information is as follows: service LOGMANAGER failed, cause by: Unable to connect to DBMS: `url=jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=***.***.***.***)(PORT=1521)))(CONNECT_DATA=*) user=*`, Caused by: IO Error: Got minus one from a read call.

## Possible Causes

- The source database server rejects access from the IP address of the DRS task.
- The source database connection information has changed.
- The number of connections to the source database has reached the upper limit.

## Solution

**Step 1** Modify the `sqlnet.ora` file in `$ORACLE_HOME/network/admin` to allow the IP address of the DRS task to access the source database.

- If a whitelist is used, `TCP.INVITED_NODES` must contain the DRS task IP address.
- If a blacklist is used, `TCP.EXCLUDED_NODES` cannot contain the DRS task IP address.

**Step 2** Check whether the source database information (such as the IP address, port number, or service name/sid) is modified. If the source database information is modified, perform the following operations:

- Restore the modified source database information. The DRS task will automatically retry to continue the synchronization task.
- Create a synchronization task again.

**Step 3** Run the following commands to check whether the number of connections to the source database has reached the upper limit.

- Run the following command to check the current number of connections:  
`select count(*) from v$sqlprocess;`
- Run the following command to check the maximum number of connections:  
`select value from v$parameter where name = 'processes';`

If the number of connections to the source database has reached the upper limit, run the following command to change the maximum number of connections allowed by the database:

```
alter system set processes = 300 scope = spfile;
```

Restart the database for the modification to take effect.

----End

## 2.8.3 Incremental Synchronization Error: Source supplemental log level is PK/UI. Missing column data at delete+insert on

\*\*\*"

### Scenarios

During an incremental synchronization from Oracle to PostgreSQL, GaussDB(DWS), or GaussDB, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: Source supplemental log level is PK/UI. Missing column data at delete+insert on \*\*\*

## Possible Causes

The supplemental log level of the source Oracle database is PK/UI. The **update** operation of the source database is not hit in the workload replay of the destination database. DRS converts the update operation to **delete** and **insert** operations by default. During the **insert** operation, the log does not contain data of other columns. As a result, an error is reported.

## Solution

Change the supplemental log level of the source database to **ALL**. Then, in the task list, click **Reset** in the **Operation** column to submit the task again.

## 2.8.4 Incremental Synchronization Error: timeout when get next file log, maybe has been deleted, please check it.

### Scenarios

During an incremental synchronization with Oracle serving as the source, an error is reported, and the log information is as follows: service CAPTURE failed, cause by: get next Oracle log file error. The next file is: 1.log, errorcode = 'code': '01300', 'name': 'LOGS\_NOT\_EXIST', 'retry': false, 'reset': false, 'level': 3, message = timeout when get next file log, maybe has been deleted, please check it.

### Possible Causes

1. The source Oracle database is a physical standby database. The logs of the source database where the incremental startup position is located are not archived. As a result, DRS cannot obtain the logs.
2. There are problems about nodes and logs in the source database. As a result, DRS fails to obtain logs and reports an error.
3. The network is unstable, affecting the speed of obtaining logs from the source database. As a result, reading logs times out.

### Solution

**Step 1** After a DRS task is started, wait for about 10 minutes, click the task name, and check whether the error log is displayed on the **Synchronization Logs** page.

- If no, DRS has obtained logs.
- If yes, go to **Step 2**.

**Step 2** If an error is reported for the LOGMANAGER process on the **Synchronization Logs** page, contact Huawei technical support.

----End

## 2.8.5 Incremental Synchronization Error: Failed to construct kafka producer.

### Scenarios

During an incremental synchronization from Oracle to Kafka, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: Failed to construct kafka producer.

### Possible Causes

If the destination Kafka instance is deployed across multiple AZs and an AZ fails, the preceding error may be reported when the Kafka client produces or consumes messages.

### Solution

**Step 1** Check and restore the Kafka partitioning.

**Step 2** In the task list, locate the target task and click **Reset** in the **Operation** column to submit the task again.

----End

## 2.8.6 Incremental Synchronization Error: Topic \*\*\* not present in metadata after 300000 ms

### Scenarios

During an incremental synchronization from Oracle to Kafka, an error is reported, and the log information is as follows: service INCREMENT failed, cause by: Topic \*\*\* not present in metadata after 300000 ms

### Possible Causes

If the destination Kafka instance is deployed across multiple AZs and an AZ fails, the preceding error may be reported when the Kafka client produces or consumes messages.

### Solution

**Step 1** Check and restore the Kafka partitioning.

**Step 2** In the task list, locate the target task and click **Reset** in the **Operation** column to submit the task again.

----End

## 2.9 Real-Time DR with MySQL Serving as the Source

## 2.9.1 DR Error: A dml without pk write target db fail

### Scenarios

During a DR task with MySQL serving as the source, an error is reported, and the log information is as follows: A dml without pk write target db fail

### Possible Causes

- If a table does not have a primary key to uniquely identify every row and the network connection is unstable, data written to the table without a primary key may be inconsistent with that in the source database.
- The source is RDS for MySQL of an earlier version (5-5.7.23). Tables that have no primary key contain hidden primary keys in the source database. As a result, the DRS task reports an error indicating that the update or delete operation is not hit.

### Solution

- If the table does not have a primary key, create a primary key for the table and create a DRS DR task again.
- If the source is RDS for MySQL of an earlier version (5-5.7.23) and there are hidden primary keys in the tables having no primary key, perform the following steps:

- a. Use an account with the process permission to run the following SQL statement at the source end to query table information. In the statement, *database/table* indicates the database name and table name of a table without a primary key. If the table is a partition table, use the **like** statement.

```
select * from information_schema.INNODB_SYS_TABLES where name = 'database/  
table';
```

```
mysql> select * from information_schema.INNODB_SYS_TABLES where name = 'test1/nopk1';  
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+  
| TABLE_ID | NAME      | FLAG | N_COLS | SPACE | FILE_FORMAT | ROW_FORMAT | ZIP_PAGE_SIZE | SPACE_TYPE |  
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+  
| 44 | test1/nopk1 | 33 | 7 | 29 | Barracuda | Dynamic | 0 | Single |  
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+  
1 row in set (0.00 sec)
```

- b. Run the following SQL statement to query the column information of the table without a primary key based on *TABLE\_ID* obtained in **a**:  
select \* from information\_schema.INNODB\_SYS\_COLUMNS where TABLE\_ID = 44;

```
mysql> select * from information_schema.INNODB_SYS_COLUMNS where TABLE_ID = 44;  
+-----+-----+-----+-----+-----+-----+  
| TABLE_ID | NAME      | POS | MTYPE | PRTYPE | LEN |  
+-----+-----+-----+-----+-----+-----+  
| 44 | name1 | 0 | 12 | 2949135 | 128 |  
| 44 | name3 | 1 | 12 | 2949135 | 128 |  
| 44 | AUTO_PK_ROW_ID | 2 | 6 | 1288 | 8 |  
| 44 | name4 | 3 | 12 | 2949135 | 128 |  
+-----+-----+-----+-----+-----+-----+  
4 rows in set (0.00 sec)
```

- c. According to the query result, the third column whose **POS** is **2** is the hidden auto-increment primary key column. If the hidden primary key column in binlog is not the last column, DRS synchronization will fail.
- d. Log in to the RDS console and **upgrade the minor kernel version** or contact RDS customer service to upgrade the version.
- e. Create a DRS task again.

## 2.10 Backup Migration

### 2.10.1 Backup Migration Failed Because Backup Files Cannot Be Found

#### Scenarios

When you migrate full backups from self-built OBS buckets to clouds, the following error message is displayed: restore:null.

#### Possible Causes

The possible causes are as follows:

- Backup files are deleted after you submit a backup migration task.
- When you upload backup files to a self-built OBS bucket, you select **Archive** for **Storage Class**. OBS archive storage offers cloud storage for rarely accessed data. An archive file uploaded for the first time is in the **Not restored** status. As a result, a Microsoft SQL Server DB instance cannot download the file.

#### Solutions

Based on the previous analysis, solutions are provided as follows:

##### Solution 1

If the migration fails because you delete the backup files, you can upload the deleted backup files again to a self-built OBS bucket and select **Standard** for **Storage Class**. For details, see the [Uploading a File or Folder](#) section in the *Object Storage Service Console Operation Guide*.

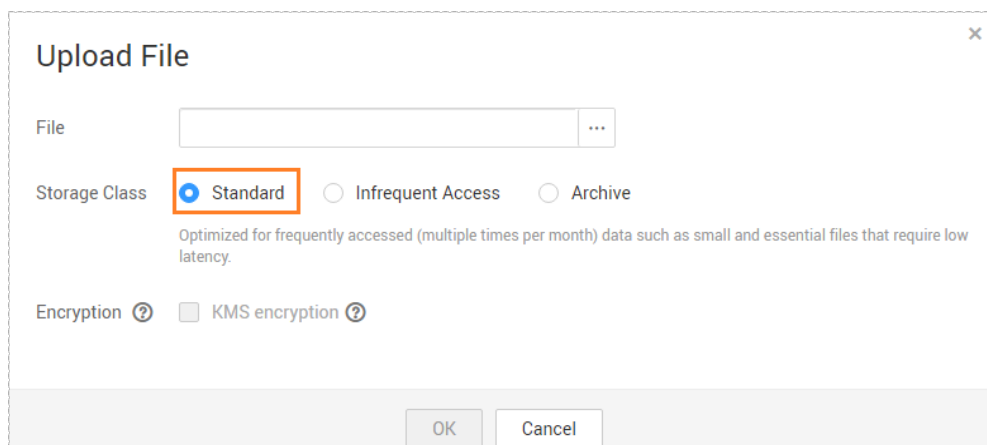
##### Solution 2

- If the migration failed because the storage class of your backup files is **Archive**, perform the following steps. If the size of backup files is small, upload the backup files again to an OBS bucket and select **Standard** for **Storage Class**.

For details, see the [Uploading a File or Folder](#) section in the *Object Storage Service Console Operation Guide*.



Figure 2-2 Uploading a file



- If the backup files are large in size, log in to the OBS console and click the bucket to which the backup files are uploaded. On the displayed page, choose **Objects** in the navigation pane on the left. On the **Objects** page, select the object to be restored and click **Restore** above the file list. After the status of the backup files becomes **Restored**, submit an offline migration task again. For details, see the [Restoring an Archive File on OBS](#) section in the *Object Storage Service Console Operation Guide*.

## 2.10.2 Backup Migration Failed Because a Backup Database Cannot Be Found in the Backup Files

### Scenarios

When you migrate full backups from self-built OBS buckets to clouds, the system displays an error message indicating that the migration failed because the source database cannot be found in the backup files.

Figure 2-3 Backup migration

Name/ID	Status	DB Engine	Created	Completed	Description	Operation
<a href="#">[Link]</a>	Successful	Microsoft SQL Se...	07/11/2018 15:52:23 GMT+0...	07/11/2018 15:54:24 GMT+0...		Delete
<a href="#">[Link]</a>	Failed	Microsoft SQL Se...	07/11/2018 15:01:02 GMT+0...	07/11/2018 15:03:34 GMT+0...	RESTORE: EpointBid_jingjia is not in backup file	Delete
<a href="#">[Link]</a>	Success		07/11/2018 12:03:51 GMT+0...	07/11/2018 12:06:22 GMT+0...		Delete

### Possible Cause

The name of a .bak backup file uploaded to a self-built OBS bucket is too long.

### Solution

Based on the previous analysis, a solution is provided as follows:

- Step 1** Check the name of the backup file to be uploaded to an OBS bucket by referring to [Preparing Backup Files](#) in the *Backup Migration*.
- Step 2** Change the name of the backup file in the local database and upload the file to a self-built OBS bucket again.

----End

## 2.10.3 Backup Migration Failed Because the Database with the Same Name Already Exists

### Scenarios

When you migrate full backup data to the cloud, the following error message is displayed: The restore database already exists in the destination DB instance

### Possible Causes

To ensure data security, RDS for SQL Server does not support migrating databases with the same name to the cloud.

### Solution

If you want to overwrite the data in the existing database, back up the existing data and delete the database with the same name. Alternatively, set **Overwrite Data** to **Yes** when creating a backup migration task, and then migrate the data again.

## 2.10.4 Backup Migration Failed Because an Incremental Backup File Is Used

### Scenarios

When you migrate full backup data to the cloud, the following error message is displayed: In full mode, incremental file restoration is not supported. To restore incremental files, perform full restoration first.

### Possible Causes

The selected backup file is an incremental backup file instead of a full backup file. Only full backup files can be migrated to the cloud at a time. Differential backup is not supported.

### Solution

Incremental files cannot be used for full data restoration. To restore incremental files, perform full restoration first or use full backup files to migrate data.

## 2.10.5 Backup Migration Failed Because an Log Backup File Is Used

### Scenarios

When you migrate full backup data to the cloud, the following error message is displayed: Target database has been restored,can not restore for transaction log

### Possible Causes

The backup file selected during task creation is a log backup file instead of a full backup file. Only full backup files can be migrated to the cloud at a time. Log backup is not supported.

### Solution

Log files cannot be used for full data restoration. Select a full backup file for data migration.

## 2.10.6 Backup Migration Failed Because the Backup File Verification Failed

### Scenarios

When you create a backup migration task, the following error message is displayed: Failed to obtain the restoration file information

### Possible Causes

The backup file is damaged or incomplete, and the backup file verification fails.

### Solution

Select a complete full backup file and perform the migration again.

## 2.10.7 Backup Migration Failed Because of Insufficient Space

### Scenarios

When you create a backup migration task, a message is displayed indicating that the space is insufficient. The following error information may be displayed:

1. The disk space of the target database is insufficient.
2. The disk space of the destination database must be 1.5 times larger than the size of the backup file.
3. The disk space of the destination database is insufficient. Check whether the backup is compressed.

## Possible Causes

- The remaining space of the destination database must be greater than 1.5 times the size of the backup file.
- The backup file is compressed. As a result, the storage space of the destination database is insufficient.

## Solution

Scale up the storage space by referring to [Scaling up Storage Space](#), or contact RDS customer service to change the destination database space and perform the migration again.

## 2.10.8 Backup Migration Failed Because Database Names Are Not Specified

### Scenarios

If you choose to restore some databases, the following error message is displayed:  
If you choose to restore a partial database, specify the database name.

### Possible Causes

If the restoration file contains multiple databases and you select some databases for restoration on the GUI, the names of the databases to be restored are not specified.

### Solution

If you select some databases for restoration, specify the names of the databases to be restored and perform the migration again.

## 2.10.9 Backup Migration Failed Because a Full Backup File Is Used

### Scenarios

When you migrate incremental backup data to the cloud, the following error message is displayed: Full files cannot be restored in incremental mode.

### Possible Causes

During an incremental backup, after the full backup file is restored, only transaction log backup files can be used. If you select a full backup file again, this error is reported.

### Solution

Full files cannot be used for incremental data restoration. Select a backup file and a backup mode based on site requirements.

## 2.10.10 Backup Migration Failed Because the LSNs of Incremental Backup Files Are Inconsecutive

### Scenarios

When you migrate incremental backup data to the cloud, the following error message is displayed: In incremental restoration mode, the incremental .bak file is not continuous with the previous full restoration file.

### Possible Causes

In a SQL Server database, the LSN of the differential backup or log backup must continuously follow the LSN of the backup file restored last time. Otherwise, this error is reported.

### Solution

Select the corresponding LSN backup file for incremental backup based on the backup time sequence. Ensure that the LSN of the selected backup file can continuously follow the LSN of the last restoration file.

## 2.10.11 Backup Migration Failed Because the Number of Databases to Be Restored Exceeds the Destination Database Threshold

### Scenarios

When you migrate backup data to the cloud, the following error message is displayed: The number of the recovery database exceeds the threshold of the target database.

### Possible Causes

The number of databases to be restored exceeds the threshold of the destination database.

### Solution

Select another RDS for SQL Server DB instance as the destination database, or delete unnecessary databases from the destination database and then perform the migration.

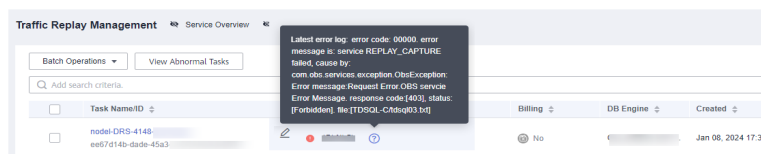
## 2.11 Workload Replay

## 2.11.1 Parsing Failed, and a Message Is Displayed Indicating That the OBS Connection Failed

### Scenarios

When you create a to-the-cloud workload replay task and obtain traffic files in the OBS bucket using an AK/SK, traffic files failed to be parsed and a message is displayed indicating that the OBS connection failed.

Figure 2-4 Parsing failed



### Possible Causes

The possible causes are as follows:

1. The AK, SK, bucket name, or endpoint is incorrect.
2. You do not have the permission to read files in the OBS bucket.

### Solution

Based on the previous analysis, a solution is provided as follows:

**Step 1** Click the task name. The **Basic Information** page is displayed.

**Step 2** In the **Connection Information** area, check whether the AK, SK, bucket name, and endpoint information is correct.

If a **temporary AK/SK** is used, you also need to check the permissions and validity period of the temporary AK/SK and security token.

- If the connection information is correct, go to **Step 3**.
- If the connection information is incorrect, click **Pause** in the **Operation** column of the task. After the task is paused, click **Edit** in the **Operation** column. On the **Configure Source and Destination Databases** page, enter information about the source database and task settings, and start the task.

**Step 3** Check whether you have the permission to read files in the OBS bucket. For details about how to grant users the permission to read files in OBS buckets, see **OBS Permissions Management**.

----End

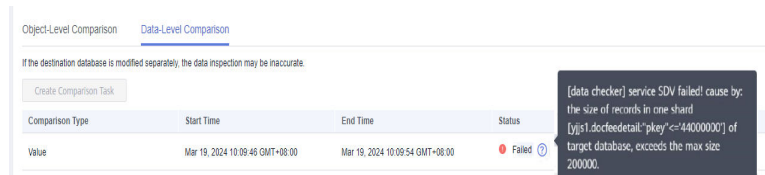
## 2.12 Data-Level Comparison

## 2.12.1 Data-Level Comparison Error: service SDV failed! cause by: the size of records in one shard[ \*\*\* ] of target database, exceeds the max size 200000

### Scenarios

During a value comparison task, an error is reported, and the log information is as follows: service SDV failed! cause by: the size of records in one shard[ \*\*\*.\*\*\* ] of target database, exceeds the max size 200000

Figure 2-5 Comparison failed



The screenshot shows a web interface for 'Data-Level Comparison'. At the top, there are tabs for 'Object-Level Comparison' and 'Data-Level Comparison'. Below the tabs, a note states: 'If the destination database is modified separately, the data inspection may be inaccurate.' There is a 'Create Comparison Task' button. A table below shows the task details:

Comparison Type	Start Time	End Time	Status
Value	Mar 19, 2024 10:09:49 GMT+08:00	Mar 19, 2024 10:09:54 GMT+08:00	Failed

An error message box is overlaid on the table, containing the text: '(data checker) service SDV failed! cause by: the size of records in one shard [type1.docfeedetail:"pkkey"<=44000000] of target database, exceeds the max size 200000.'

### Possible Causes

The data model of the table to be compared is special. As a result, automated sharding cannot be performed for the comparison task, and the task fails.

### Solution

Create a value comparison task again and deselect the table for which the error is reported.

- Step 1** Click the task name. The **Basic Information** page is displayed.
- Step 2** Take a synchronization task as an example. Choose **Synchronization Comparison**.
- Step 3** Click the **Data-Level Comparison** tab and click **Create Comparison Task**.
- Step 4** Deselect the table where the error is reported and click **OK** to submit the comparison task again.
- Step 5** If the fault persists, choose [Service Tickets > Create Service Ticket](#) in the upper right corner of the management console and contact DRS customer service.

----End