

Database and Application Migration UGO

FAQs

Issue	01
Date	2024-03-08



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

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

1 Product Consulting	1
1.1 Instance Freezing, Release, and Deletion	1
1.2 Which Schemas in Source Databases Are Ignored for Migration?	2
1.3 What Are the Differences Between Syntax Conversion and Migration & Verification in a Migration Project?	3
1.4 What Are the Database Schema Changes After an Oracle Database Is Migrated to the Target Database?	3
1.5 Why Cannot I Use Some Functions?	3
1.6 What Is the Function of the dsc_ora_ext Schema Generated After Migration to the Target Database?	4
1.7 What Should I Do If Data Collection Fails or Is Slow Due to the Small Values of Certain Oracle SGA Parameters?	4
1.8 What Should I Do If Data Collection Fails and a Message SNAPSHOT TOO OLD Is Displayed?	4
1.9 What Is the Mapping Between the GaussDB Versions Displayed on the UGO Console and Self-built GaussDB Versions?	5
2 Database Connections	6
2.1 What Should I Do If I Cannot Connect to the Source Oracle Database During Database Evaluation Project Creation?	6
2.2 What Should I Do If I Failed To Connect to the Source Database as User sys ?	6
2.3 How Do I Create GaussDB Databases Compatible with Source Databases?	6
2.4 What Should I Do If My Database Fails to be Connected?	8
3 Evaluation Project	10
3.1 How Do I Select a Connection Method?	10
3.2 How Does UGO Collect Data from Source Databases?	10
3.3 How Does UGO Check Database Permissions?	11
3.3.1 Oracle as the Source Database Type	11
3.3.2 MySQL as the Source Database Type	12
3.3.3 GoldenDB as the Source Database Type	13
3.3.4 Microsoft SQL Server as the Source Database Type	14
3.3.5 PostgreSQL as the Source Database Type	14
3.4 What Are the Possible Causes for an Object Collection Failure?	15
3.5 In GaussDB, How Do I Configure a Search Path If A Statement Without Schema Name Fails To Be Executed?	15
3.6 What Is Native Supported, UGO Supported, Supported with Risk, or Unsupported Objects?	16

3.7 What Is the Relationship Between Migration Risk (Top 10 risk SQL) and Risky SQL Summary?.....	16
3.8 What Are Reconstruction Statistics and How Are Reconstruction Points Measured?.....	16
3.9 What Should I Do If An Object Collection Error (Closed Connection) Is Displayed During Evaluation Project Creation?.....	17
3.10 What Should I Do If "ErrorCode=4036" Is Displayed During Evaluation Project Creation?.....	18
3.11 What Should I Do If "ErrorCode=17002" Is Displayed During Evaluation Project Creation?.....	19
4 Migration Project.....	21
4.1 Why Is There No Available Evaluation Project During Migration Project Creation?.....	21
4.2 When Should I Use the Conversion Config Function?.....	21
4.3 What Should I Do If SQL Modifications Failed to Be Saved During Object Correction?.....	21
4.4 Why Is a Name Error Reported During Database Migration?.....	22
4.5 What Should I Do If Migration & Verification Failed to be Performed?.....	23
4.6 How Do I Solve ERROR: syntax error at or near "@"?.....	23
4.7 What Is the Impact of the Migration on the Source Database?.....	24
4.8 What Are the Impacts of GUC Parameter Settings of GaussDB on Migration?.....	25
4.8.1 Setting GUC Parameters for GaussDB When the Source Database Type Is Oracle.....	25
4.8.2 Setting GUC Parameters for GaussDB When the Source Database Type Is MySQL.....	32
4.9 Why Is the Number of Indexes Different After Oracle Data is Migrated to GaussDB?.....	35
4.10 What Should I Do If There Are ctid, xc_node_id, and tableoid Columns in GaussDB?.....	37
4.11 Why Is Data Inconsistent When SELECT Statements Without ORDER BY Are Used for Query in Distributed GaussDB?.....	38
5 Change History.....	40

1 Product Consulting

1.1 Instance Freezing, Release, and Deletion

Why Are My UGO Resources Released?

If your subscriptions have expired but not been renewed, or you are in arrears due to insufficient balance, your resources enter a grace period. If you still do not complete the payment or renewal after the grace period expires, you will enter a retention period. During the retention period, the resources are not available. If you still do not renew them or top up your account after the retention period ends, your resources will be released and your data stored will be deleted. For details, see [Service Suspension and Resource Release](#).

Why Are My UGO Resources Frozen?

Your UGO resources may be frozen for a variety of reasons. The most common reason is that you are in arrears.

How Do I Unfreeze My Resources?

Frozen due to arrears: You can renew your resources or top up your account.

The frozen resources can be renewed, released, or deleted.

What Impacts Does Instance Freezing, Unfreezing or Release Have on My Services?

- After the UGO resources are frozen:
 - You can only view existing projects, and are not allowed to modify, delete, create projects or convert SQL statements.
 - You can manually delete UGO.
- After your resources are unfrozen, you can perform operations such as creation.
- After your resources are released, UGO will be deleted.

How Do I Renew My Resources?

After UGO expires, you can renew it on the [Renewals](#) page. For details, see [Renewal Management](#).

Can My Resources Be Recovered After Being Released?

Deleted resources cannot be recovered.

1.2 Which Schemas in Source Databases Are Ignored for Migration?

Oracle as the Source Database Type

UGO does not migrate the following schemas:

```
'APEX_050000', 'AUDSYS', 'ADAMS', 'ANONYMOUS',  
'AURORA$ORB$UNAUTHENTICATED', 'AWR_STAGE', 'APEX_030200', 'APEX_040200',  
'APEX_PUBLIC_USER', 'APPQOSSYS', 'BI', 'BLAKE', 'CLARK', 'CSMIG',  
'CTXSYS', 'DBSNMP', 'DIP', 'DMSYS',  
'DSSYS', 'DEMO', 'DVSYS', 'DVF',  
'DBSFUSER', 'EXFSYS', 'FLOWS_FILES',  
'GGSYS', 'GSMADMIN_INTERNAL', 'GSMCATUSER', 'GSMUSER', 'HR',  
'IX', 'JONES', 'LBACSYS', 'MDDATA', 'MDSYS', 'MGMT_VIEW',  
'OE', 'OLAPSYS', 'ORACLE_OCM', 'ORDDATA', 'ORDPLUGINS',  
'ORDSYS', 'OUTLN', 'OWBSYS', 'OWBSYS_AUDIT', 'OJVMSYS',  
'PERFSTAT', 'PM', 'REMOTE_SCHEDULER_AGENT', 'SCOTT', 'SH',  
'SI_INFORMTN_SCHEMA', 'SPATIAL_CSW_ADMIN_USR',  
'SPATIAL_WFS_ADMIN_USR', 'SYS', 'SYSMAN', 'SPATIAL_CSW_ADMIN_USR', 'SYSBACKUP',  
'SYSKM', 'SYSDBG', 'SYSRAC', 'SYS$UMF', 'SYSTEM',  
'TRACESVR', 'TSMYSYS', 'WMSYS', 'XDB', 'XS$NULL', 'GSMROOTUSER'
```

MySQL as the Source Database Type

- information_schema
- mysql
- performance_schema
- sys

PostgreSQL as the Source Database Type

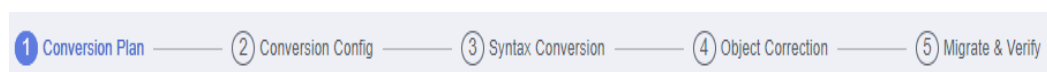
- information_schema
- pg_catalog
- pg_temp_1
- pg_toast
- pg_toast_temp_1
- public

Microsoft SQL Server as the Source Database Type

- guest
- INFORMATION_SCHEMA

- sys
- db_owner
- db_accessadmin
- db_securityadmin
- db_ddladmin
- db_backupoperator
- db_datareader
- db_datawriter
- db_denydatareader
- db_denydatawriter

1.3 What Are the Differences Between Syntax Conversion and Migration & Verification in a Migration Project?



Syntax conversion is the conversion of SQL scripts of source database objects into SQL scripts of the target database objects. The SQL scripts are not executed in the target database.

Migration & Verification is the sending of converted SQL statements to the target database for execution.

1.4 What Are the Database Schema Changes After an Oracle Database Is Migrated to the Target Database?

- The Oracle schemas are converted into the PostgreSQL schemas.
- The Oracle schemas are converted into the MySQL databases.

1.5 Why Cannot I Use Some Functions?

- You do not have the required permissions.
For details about common UGO operations and corresponding actions, see [Permissions Management](#).
For details about how to apply for permissions, see [Creating a User and Granting Permissions](#).
- Your account is frozen or restricted.
You can only view existing projects, and are not allowed to modify, delete, create projects or convert SQL statements.
In this case, you can click **Submit a service Ticket** as prompted. Alternatively, contact the customer service by choosing [Service Tickets > Create Service Ticket](#) in the upper right corner of the console.

1.6 What Is the Function of the dsc_ora_ext Schema Generated After Migration to the Target Database?

dsc_ora_ext is a custom schema compiled by UGO to enable the target database to implement certain functions in the Oracle source database.

If the source database is MySQL, the custom schema is **dsc_mys_ext**.

For example, UGO creates custom target database functions that provide the same functions as Oracle system functions, and it converts Oracle system function calls to custom function calls during migration.

1.7 What Should I Do If Data Collection Fails or Is Slow Due to the Small Values of Certain Oracle SGA Parameters?

If data collection fails or is slow because certain parameters in the Oracle system global area (SGA) are set to small values, run the following command:

show parameter SGA;

Command output:

NAME	TYPE	VALUE
sga_max_size	big integer	796M
sga_target	big integer	0

Contact the database administrator to increase the values of the SGA parameters.

ALTER SYSTEM SET sga_max_size=5G scope=spfile;

ALTER SYSTEM SET sga_target=5G scope=spfile;

NOTE

The preceding parameter values are for reference only. If their value is too large, more resources are occupied. Contact the database administrator to set the parameters to appropriate values.

For details about the parameters, see the [Oracle official documentation](#).

1.8 What Should I Do If Data Collection Fails and a Message SNAPSHOT TOO OLD Is Displayed?

If the collection fails or the error message "ORA-01555: snapshot too old" is displayed, check the UNDO parameters of the source Oracle database. Run the following command:

show parameter undo;

Command output:

```
NAME TYPE VALUE
undo_management string AUTO
undo_retention integer 28800
```

Contact the database administrator to increase the UNDO_RETENTION value.

ALTER SYSTEM SET UNDO_RETENTION =*N*;

Replace *N* with an appropriate value.

1.9 What Is the Mapping Between the GaussDB Versions Displayed on the UGO Console and Self-built GaussDB Versions?

Log in to your self-built GaussDB database and run the following command to query the version:

select version ();

Table 1-1 describes the mapping between self-built GaussDB database versions and the GaussDB versions displayed on the UGO console.

Table 1-1 Version mapping

Self-built GaussDB Version	GaussDB Version on the UGO Console
V500R002C10	GaussDB 2.7 Enterprise Edition
503.0.x	GaussDB 3.1 Enterprise Edition
503.1.x	GaussDB 3.2 Enterprise Edition
503.2.x	GaussDB 3.3 Enterprise Edition
505.0	GaussDB 8.0 Enterprise Edition
505.1	GaussDB 8.100 Enterprise Edition

2 Database Connections

2.1 What Should I Do If I Cannot Connect to the Source Oracle Database During Database Evaluation Project Creation?

Check:

- Whether the network between the source database and UGO is connected. Currently, UGO can connect to the source database through a public network.
- Whether the network is stable (including the packet loss rate and delay). If the network quality of the source database is poor, the evaluation project may fail.
- Whether UGO is allowed by the source database firewall settings.
- Whether the database connection information is correct.

2.2 What Should I Do If I Failed To Connect to the Source Database as User sys?

User **sys** is the super administrator of Oracle databases. Generally, the Oracle database restricts the remote login of user **sys**. You are advised to connect to the source database as other database users.

2.3 How Do I Create GaussDB Databases Compatible with Source Databases?

Oracle as the Source Database Type

Step 1 Log in to your GaussDB instance as a user who has the permission to create databases.

Step 2 Create a GaussDB database compatible with Oracle schemas.

Primary/Standby:

```
create database databasename dbcompatibility = 'A';
```

Distributed:

```
create database databasename dbcompatibility = 'ORA';
```

Step 3 Check whether the database is created.

```
select * from pg_database where datname = 'databasename';
```

- Primary/standby: If the value of **datcompatibility** is **A**, the database is created.
- Distributed: If the value of **datcompatibility** is **ORA**, the database is created.

----End

MySQL as the Source Database Type

Step 1 Log in to your GaussDB instance as a user who has the permission to create databases.

Step 2 Create a GaussDB database compatible with MySQL schemas.

Primary/Standby:

```
create database databasename dbcompatibility = 'B';
```

Distributed:

```
create database databasename dbcompatibility = 'MySQL';
```

Step 3 Check whether the database is created.

```
select * from pg_database where datname = 'databasename';
```

- Primary/standby: If the value of **datcompatibility** is **B**, the database is created.
- Distributed: If the value of **datcompatibility** is **MySQL**, the database is created.

----End

PostgreSQL as the Source Database Type

Step 1 Log in to your GaussDB instance as a user who has the permission to create databases.

Step 2 Create a GaussDB database compatible with Oracle schemas.

NOTE

The Oracle compatibility mode of the GaussDB database supports many syntax features. Currently, UGO analyzes syntax conversion solutions based on this compatibility mode. Therefore, you are advised to set the Oracle compatibility mode to increase the syntax conversion success rate.

Primary/Standby:

```
create database databasename dbcompatibility = 'A';
```

Distributed:

```
create database databasename dbcompatibility = 'ORA';
```

Step 3 Check whether the database is created.

```
select * from pg_database where datname = 'databasename';
```

- Primary/standby: If the value of **datcompatibility** is **A**, the database is created.
- Distributed: If the value of **datcompatibility** is **ORA**, the database is created.

----End

SQL Server as the Source Database Type

Step 1 Log in to your GaussDB instance as a user who has the permission to create databases.

Step 2 Run the following command to create a GaussDB database in Oracle compatibility mode:

NOTE

The Oracle compatibility mode of the GaussDB database supports many syntax features. Currently, UGO analyzes syntax conversion solutions based on this compatibility mode. Therefore, you are advised to set the Oracle compatibility mode to increase the syntax conversion success rate.

Primary/Standby:

```
create database databasename dbcompatibility = 'A' ;
```

Distributed:

```
create database databasename dbcompatibility = 'ORA';
```

Step 3 Check whether the database is created.

```
select * from pg_database where datname = 'databasename';
```

- Primary/standby: If the value of **datcompatibility** is **A**, the database is created.
- Distributed: If the value of **datcompatibility** is **ORA**, the database is created.

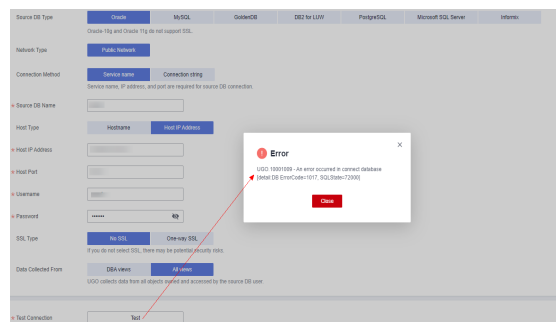
----End

2.4 What Should I Do If My Database Fails to be Connected?

Symptom

When UGO connected to a database, an error message was displayed, indicating that the connection failed.

Figure 2-1 Error message



Possible Causes

1. The database connection information is incorrect.
2. The database user does not have permissions required to connect to the database.
3. The network between the database and UGO is disconnected.
4. The database firewall restricts UGO access to the database.
5. There are too many connections to the database.

Solution

1. Correct the database connection information.
2. Grant the connection permissions to the user.
3. Ensure that the network between the database and UGO is connected.
4. Allow UGO to access the database.
5. Adjust the maximum number of connections for the database.

3 Evaluation Project

3.1 How Do I Select a Connection Method?

- **Service Name:** You need to specify the source database name, host IP address or host name, username and other parameters. UGO constructs a connection string based on these parameters.
- **Connection String:** JDBC URL, which must comply with the format specifications of the source database. You also need to specify database name and host name or IP address.

3.2 How Does UGO Collect Data from Source Databases?

Oracle as the Source Database Type

Unless otherwise specified, DDL information about Oracle objects is obtained using the **DBMS_METADATA.get_ddl system** package function.

The basic information about an object is obtained from system views, which consists of **DBA views** and **All views**. The following uses the **All views** as an example.

INDEX: During index collection, all constraint information is queried from the **ALL_INDEXES** view, filter out the indexes that are created based on primary keys or unique key constraints. (If an index is created before a constraint is created and the constraint name is the same as the index name, the index will not be collected.)

PostgreSQL as the Source Database Type

UGO uses the following method to collect each object of the source database.

- **SCHEMA:** Run **CREATE SCHEMA *schema name*;**
- **TABLE:** Query the **pg_catalog.pg_attribute**, **pg_catalog.pg_class**, **pg_catalog.pg_namespace**, and **pg_inherits** system views to obtain the table

name, field name, field type, and table partition information, and run the **CREATE TABLE** statement. Information about generated columns, table constraints, and foreign key constraints is not collected.

- **INDEX:** Query the **pg_indexes** view to obtain the indexdef field and run the **CREATE INDEX** statement.
- **VIEW:** Query the **pg_views** view to obtain the definition field and run the **CREATE VIEW** statement.
- **FUNCTION/PROCEDURE:** Use **pg_get_functiondef** to obtain the **CREATE FUNCTION** and **CREATE PROCEDURE** statements.

Note that **pg_get_functiondef** cannot process aggregate functions. User-defined aggregate functions cannot be collected.

- **TRIGGER:** Use **pg_get_functiondef** to obtain the trigger function definition, use **pg_get_triggerdef** to obtain the **CREATE TRIGGER** statement, and run the **CREATE TRIGGER** statement.

There are no specific methods for object DDL collection in PostgreSQL. The DDLs of some objects are obtained by querying metadata. As a result, the collected DDL information may be inconsistent with the DDL information in the source database, or even the information may be missing.

If the collected DDLs cannot meet your requirements, you can use **pg_dump** or a third-party database client tool to obtain object DDLs, and then [convert the SQL statements](#).

3.3 How Does UGO Check Database Permissions?

3.3.1 Oracle as the Source Database Type

Check Item	Description	Suggestion
DBMS_MET ADATA	Permission to retrieve metadata from the Oracle database dictionary. This permission is used to obtain the DDL of schema objects.	DBA permission needs to be granted. 1. Create a user. CREATE USER user IDENTIFIED BY password; 2. Grant the login permission to the user. GRANT CONNECT TO user; 3. Grant the DBA permission to the user. GRANT DBA TO user;
Dynamic View	Permission to access various dynamic performance views. This permission is used to obtain basic database information.	DBA permission needs to be granted.

Check Item	Description	Suggestion
Schema Objects	Permission to check schema objects to be evaluated. At least one object needs to be evaluated.	-
DBA	DBA permission required for subsequent operations.	DBA permission needs to be granted.

3.3.2 MySQL as the Source Database Type

Check Item	Description	Suggestion
mysql database	By default, when UGO connects to the mysql database, the user needs to be granted permission to access the mysql database.	Create a user. CREATE USER db-user IDENTIFIEDBY passwd; GRANT SELECT ON mysql.* TO db-user;
Process	Permission to view all tables in information_schema.	Run the following statement: GRANT PROCESS ON *.* TO db-user;
Object collection	Permission to collect objects	Run the following statements: GRANT SELECT ON schema-name.* TO db-user; GRANT SHOW VIEW ON schema-name.* TO db-user; GRANT TRIGGER ON schema-name.* TO db-user; NOTE To collect global objects, replace <schema-name>.* with *.*.

Check Item	Description	Suggestion
Stored procedure and function	In MySQL 8.0.20 and later versions, if there are stored procedures and functions in the source database, grant the user SHOW_ROUTINE permission.	Run the following statement: GRANT SHOW_ROUTINE ON *.* TO db-user ;

3.3.3 GoldenDB as the Source Database Type

Check Item	Description	Suggestion
GoldenDB system table mysql.user	Permission to collect information about USER and ROLE objects.	1. Create a user. CREATE USER db-user IDENTIFIEDBY passwd; GRANT SELECT ON mysql.user TO db-user;
Process	Permission to view all tables in information_schema .	Run the following statement: GRANT PROCESS ON *.* TO db-user;
Object collection	Permission to collect objects	Run the following statement: GRANT SELECT ON schema-name.* TO db-user; GRANT SHOW VIEW ON schema-name.* TO db-user; GRANT TRIGGER ON schema-name.* TO db-user; NOTE <ul style="list-style-type: none">• To collect global objects, replace <schema-name>.* with *.*• By default, the user created on any GoldenDB CN node can connect to all databases. To collect database objects, the user needs to be granted to the SELECT permission.

Check Item	Description	Suggestion
Stored procedure and function	In MySQL 8.0.20 and later versions, if there are stored procedures and functions in the source database, grant the user SHOW_ROUTINE permission.	Run the following statement: GRANT SHOW_ROUTINE ON *.* TO db-user ;

3.3.4 Microsoft SQL Server as the Source Database Type

Table 3-1 Check items for source database Microsoft SQL Server

Check Item	Description	Suggestion
DDL query	Permission to obtain database objects.	Grant the permission to obtain database objects to the user. Run the following statement: GRANT SELECT ON OBJECT ;
Schema objects	Permission to check schema objects to be evaluated.	At least one schema object is selected, or the evaluation task cannot be created. NOTE The number of objects in system schemas of SQL Server is not checked.
View Definition	Permission to view definition.	The user without the permission cannot query meta information. Grant the View Definition permission to the user. Run the following statement: GRANT VIEW DEFINITION ON DATABASE :: <database> TO <user> ;

3.3.5 PostgreSQL as the Source Database Type

Check Item	Description	Suggestion
Schema Objects	Permission to check schema objects to be evaluated. At least one object needs to be evaluated.	-

3.4 What Are the Possible Causes for an Object Collection Failure?

Object collection can fail for various reasons. Some of the possible reasons are as follows:

- The network to the source database is unavailable.
In this case, restore the network connection between UGO and the source database and re-create an evaluation project.
- The target database could not be verified.
If a namespace is specified in the table creation statement, check whether the namespace exists in the target database. If no, create it.
If a view or function fails to be created, check whether it depends on the target database table. If yes, ensure that the table is successfully created.

3.5 In GaussDB, How Do I Configure a Search Path If A Statement Without Schema Name Fails To Be Executed?

You can execute the following command in GaussDB to see if the **aa** table exists.

```
create schema sch1;  
  
create table sch1.aa(col int);  
  
select * from aa;
```

If it does not exist, the actual error message is as follows: -ERROR: The table **aa** does not exist.

```
LINE 1: select * from aa;
```

```
^
```

```
SQL state: 42P01
```

```
Character: 15
```

This is because there is no **sch1** in the search path.

Run following statement to add **sch1** to the search path:

```
set search_path = "$user",public,sch1;
```

Run the SELECT statement.

```
select * from aa;
```

In this case, no error occurs.

 NOTE

To obtain the current search_path, run the following statement:
`show search_path;`

3.6 What Is Native Supported, UGO Supported, Supported with Risk, or Unsupported Objects?

- **Native Supported:** The object syntaxes of source and target databases are compatible.
- **UGO Supported:** UGO can convert the object syntax of source database to be compatible with the target database.
- **Supported with Risk:** UGO can convert the object syntax of source database to be compatible with the target database, but there may be some performance or functional differences after conversion.
- **Unsupported Objects:** UGO cannot convert the object syntax of source database to be compatible with the target database.

Supported objects include **Native Supported**, **UGO Supported**, and **Supported with Risk** objects.

3.7 What Is the Relationship Between Migration Risk (Top 10 risk SQL) and Risky SQL Summary?

There is no relationship between migration risk and risky SQL summary.

Migration Risks (Top 10 risk SQL): describes the 10 slow SQL statements using up the most CPU and memory resources on the source database over the past 7 days.

Risky SQL Summary: describes migration risks from the perspective of the target database. You need to pay attention to some functions that are not directly supported by the target database.

Risks are classified into functional and performance risks.

- **Functional risks:** risks that affect the database functions. For example, data types (such as value ranges) that are not directly supported by the target database can be summarized in terms of data_type_mismatch, table_def_mismatch and sequence_limitation.
- **Performance risks:** risks that affect the database performance. For example, GaussDB does not support partition intervals, which can be summarized in terms of distribution and partitioning.

3.8 What Are Reconstruction Statistics and How Are Reconstruction Points Measured?

Each clause or keyword in the source syntax can be considered a reconstruction point in the migration.

Successful reconstruction points are collected based on native-supported objects, UGO-supported objects, and supported objects with risks.

Failed reconstruction points are collected based on UGO-unsupported objects.

3.9 What Should I Do If An Object Collection Error (Closed Connection) Is Displayed During Evaluation Project Creation?

Symptom

When an evaluation project is being created, **Project Status** becomes **Stopped**. **Object Collection Error**. After you click **Object Collection Error**, **Closed Connection** is displayed in the **Error** column.

Figure 3-1 Error display

▲ - Object Collection Error

Schema Name	Object Type	Object Name	Error
CDATA10	TABLE	BASE_SERVICE_EVENT_T	Closed Connection
CDATA10	TABLE	BASE_SITEMAP_T	Closed Connection
CDATA10	TABLE	BASE_TABLE_BACKUP_IGNORE_T	Closed Connection

Causes

The collection time for UGO to collect DDL information from the source database was set to 60 seconds. If the size of the database objects to be collected are too large or no data is returned within 60 seconds due to poor database performance or network connection, there may be an object collection error.

Solution

Method 1: Manually submit the SQL statements of the objects that fail to be collected.

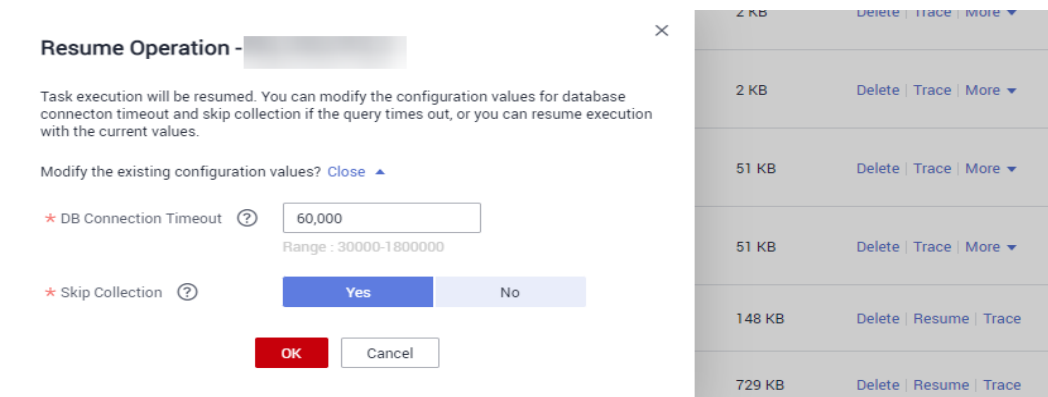
- Step 1** Log in to the UGO console.
- Step 2** In the navigation pane on the left, choose **Schema Migration > DB Evaluation**.
- Step 3** Click the project name to go to the **Source DB Analysis** page.
- Step 4** Click **View Object Details** in the **Object Statistics** area and select a failed schema.
- Step 5** Click **Edit SQL** in the **Operation** column and manually import the SQL statement, and save it.
- Step 6** Locate the desired project on the project list page and click **Resume** in the **Operation** column.

----End

Method 2: Change the collection time.

Step 1 Locate the desired project and click **Resume** in the **Operation** column.

Figure 3-2 Resuming an evaluation task



Step 2 Change the value of **DB Connection Timeout**.

Step 3 Click **OK**.

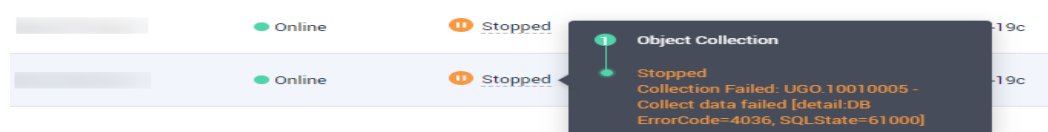
----End

3.10 What Should I Do If "ErrorCode=4036" Is Displayed During Evaluation Project Creation?

Symptom

When an evaluation project is being created, **Project Status** becomes **Stopped**. After you move the mouse to **Stopped**, the error message "ErrorCode=4036" is displayed.

Figure 3-3 Error message



Causes

The value of **pga_aggregate_limit** is inappropriate and there is an error **pga-memory-used-by-the-instance-exceeds-pga-aggregate-limit** in the database.

Solution

Step 1 Log in to the source Oracle database as the **oracle** user.

Step 2 Run the following command to query the value of **pga_aggregate_limit**.

show parameter pga_aggregate_limit;

Step 3 Run the following command to change the value of **pga_aggregate_limit**.

```
alter system set pga_aggregate_limit = 0;
```

Step 4 Log in to the UGO console, locate the desired project, and click **Resume** in the **Operation** column.

----End

3.11 What Should I Do If "ErrorCode=17002" Is Displayed During Evaluation Project Creation?

Symptom

When an evaluation project is being created, **Project Status** becomes **Stopped**. After you move the mouse to **Stopped**, the error message "ErrorCode=17002" is displayed.

Causes

During peak hours or when multiple database evaluation tasks are being created at the same time, the number of database connections will reach the upper limit. As a result, UGO cannot obtain valid connections from the source database and several projects are in the **Stopped** status.

Solution

Step 1 Log in to the source Oracle database as the **oracle** user.

Step 2 Query the maximum number of database connections.

```
show parameter processes;
```

Step 3 Create **spfile**.

```
create spfile from pfile;
```

Step 4 Restart the source database.

```
shutdown immediate;
```

```
startup
```

Step 5 Run the following command to change the maximum number of connections for the source database:

```
alter system set processes = 2000 scope = spfile;
```

Step 6 Restart the source database and check whether the maximum number of connections is changed successfully.

```
shutdown immediate;
```

```
startup
```

```
show parameter processes;
```

Step 7 Log in to the UGO console, locate the desired project, and click **Resume** in the **Operation** column.

----End

4 Migration Project

4.1 Why Is There No Available Evaluation Project During Migration Project Creation?

The evaluation project creation is not yet complete.

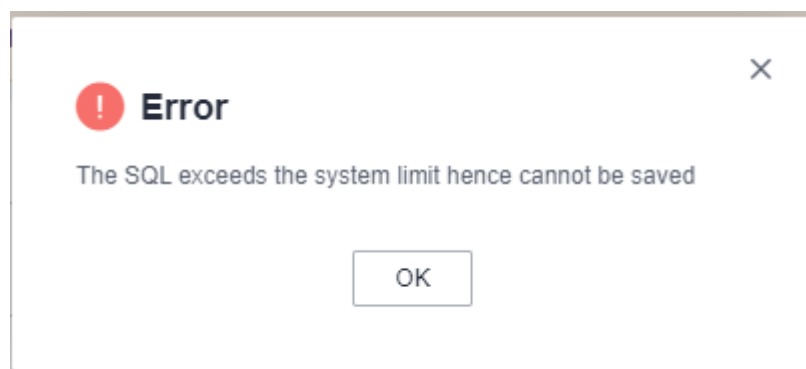
Before migration project creation, ensure that there is at least one evaluation where the target database has been confirmed.

4.2 When Should I Use the Conversion Config Function?

Before the migration, if multiple migration solutions are available in the source database application system, you can click **Conversion Config** to select the optimal configuration solution (based on the source database system and your requirements).

4.3 What Should I Do If SQL Modifications Failed to Be Saved During Object Correction?

After you modify a SQL statement on the object correction page and save the modifications, an error message shown in the following figure may be displayed:

Figure 4-1 Error message

Reason: The statement length exceeds the upper limit of 1 MB.

Suggestion: Adjust the statement length.

4.4 Why Is a Name Error Reported During Database Migration?

Symptom

When DB2 for LUW is used as the source database and GaussDB is used as the target database, an error similar to the following picture is reported during object correction.

```
1 CREATE OR REPLACE TRIGGER TEST.FE2022051000290_EXTRATRIG4
2 BEFORE DELETE ON TEST.emp_t
3 REFERENCING OLD AS oobj
4 FOR EACH ROW
5 BEGIN
6     DECLARE REVSTR VARCHAR(16) DEFAULT 'ABC';
7     DECLARE "REV STR", "REV STR " BOOLEAN DEFAULT FALSE; -- (2)
8
9     DECLARE LEN INT;
10    INSERT INTO emp_bk
11    VALUES ( oobj.empno, oobj.ename, oobj.deptno );
12
13 ;
```

```
1 CREATE OR REPLACE TRIGGER TEST.FE2022051000290_EXTRATRIG4
2 BEFORE DELETE ON TEST.emp_t
3 REFERENCING OLD AS oobj
4 FOR EACH ROW
5 BEGIN
6     DECLARE REVSTR VARCHAR(16) DEFAULT 'ABC';
7     DECLARE "REV STR", "REV STR " BOOLEAN DEFAULT FALSE; -- (2)
8
9     DECLARE LEN INT;
10    INSERT INTO emp_bk
11    VALUES ( oobj.empno, oobj.ename, oobj.deptno );
12
13 ;
```

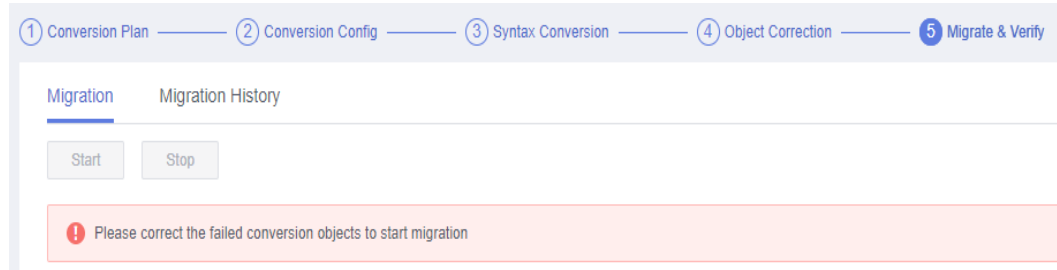
Causes

DB2 for LUW supports variables in DECLARE statements with same name followed by a different number of spaces, UGO does not support this scenario for conversion to GaussDB.

Solution

Manually delete redundant spaces during object correction.

4.5 What Should I Do If Migration & Verification Failed to be Performed?



You need to modify or ignore the objects that were not be migrated.

4.6 How Do I Solve ERROR: syntax error at or near "@"?

Symptom

In primary/standby GaussDB 3.2, 3.3, and 8.0 enterprise edition, after you enter `set @a = 1;`, the following error information is displayed:

Figure 4-2 Error message

```
=> set @a = 1;
ERROR: syntax error at or near "@"
LINE 1: set @a = 1;
           ^
```

Causes

The `enable_set_variables` parameter is not configured.

Solution

Method 1: Configuring the value of `enable_set_variables` and applying the new value in a specified database

Step 1 Connect to the database as the `root` user.

Step 2 Run the following command to configure the `enable_set_variables` parameter:

```
alter database databasename set b_format_behavior_compat_options =
'enable_set_variables';
```

----End

Method 2: Configuring the value of `enable_set_variables` and applying the new value in a specified instance

Step 1 Connect to the instance as the **root** user.

Step 2 Run the following command to switch to the instance user:

```
su - omm
```

 **NOTE**

In the preceding command, **omm** indicates the instance user. Change it based on the site requirements.

Step 3 Run the following command to set the instance parameters:

```
gs_guc reload -Z datanode -N all -I all -c  
"b_format_behavior_compat_options='enable_set_variables';  
----End
```

4.7 What Is the Impact of the Migration on the Source Database?

- When a UGO evaluation task connects to the source database to collect objects, the execution time of the select or alter statement in the source Oracle database increases by about 0.1s.
- The following table describes the impact on the source database performance.

Table 4-1 Impact on the source database performance

Source Database Type	Configuration	CPU Usage	Memory Usage	I/O Usage	Query Statement Execution Time
Oracle database without application load	vCPUs: 48 Memory: 188 GB Storage: 511 GB	0.04% -> 17.03%	1.5% -> 2.56%	0.1% -> 2.8%	N/A
Oracle database with application load simulated in TPC-H	vCPUs: 48 Memory: 188 GB Storage: 511 GB	0.04% -> 19.63%	1.52% -> 2.54%	0.1% -> 10.2%	Increased by 5%

 NOTE

- Source: Lab test data. The specific impact depends on the actual situation.
- The TPC-H is a well-known benchmark used by many database vendors. It consists of a suite of business oriented ad-hoc queries and concurrent data modifications. The queries and the data populating the database have been chosen to have broad industry-wide relevance.

4.8 What Are the Impacts of GUC Parameter Settings of GaussDB on Migration?

4.8.1 Setting GUC Parameters for GaussDB When the Source Database Type Is Oracle

When migrating data from Oracle to GaussDB, you can set the GUC parameters of GaussDB to enhance the compatibility. For details about the GUC parameters involved in the migration and the impact scope, see [Table 4-2](#).

 CAUTION

GaussDB provides many operation parameters. Configurations of these parameters affect the behavior of the database system. Before changing these parameters, learn the impact of these parameters on the database. Otherwise, unexpected results may occur. For details, see [GUC Parameter Usage](#).

Table 4-2 GUC parameters for migrating data from Oracle to GaussDB

Configur ation item (Compat ible with A Mode)	Compatibili ty Configurati on Item	Compatibility Configuration Item Name	Supp orted Datab ase	Enab le	Description
behavior _compat _options	display_leadi ng_zero	Specifies how floating point numbers are displayed. 0 before the decimal point is displayed	Prima ry/ Stand by and distrib uted	Yes	Primary/ Standby: GUC Parameters . Distributed: Parameter Details
	end_month_ calculate	Specifies the calculation logic of the add_months function.	Prima ry/ Stand by and distrib uted	Yes	

Configuration item (Compatible with A Mode)	Compatibility Configuration Item	Compatibility Configuration Item Name	Supported Database	Enable	Description
	bind_procedure_searchpath	Specifies the search path of the database object in a stored procedure for which no schema name is specified.	Primary/Stand by and distributed	Yes	
	correct_to_number	Specifies the compatibility of the to_number() result.	Primary/Stand by and distributed	Yes	
	unbind_divide_bound	Specifies the range check on the result of integer division.	Primary/Stand by and distributed	Yes	
	convert_string_digit_to_numeric	Determines whether to convert numeric constants of the character string type to those of the numeric type before these two types are compared.	Primary/Stand by and distributed	Yes	
	return_null_string	Specifies how to display the empty result (empty string ") of the lpad() and rpad() functions.	Primary/Stand by and distributed	Yes	

Configuration item (Compatible with A Mode)	Compatibility Configuration Item	Compatibility Configuration Item Name	Supported Database	Enable	Description
	merge_update_multi	When MERGE INTO ... WHEN MATCHED THEN UPDATE and INSERT ... ON DUPLICATE KEY UPDATE are used, controls the UPDATE behavior if a piece of target data in the target table conflicts with multiple pieces of source data.	Primary/Standby and distributed	Yes	
	plstmt_implicit_savepoint	Determines whether the execution of an UPDATE statement in a stored procedure has an independent subtransaction.	Primary/Standby and distributed	Yes	
	hide_tailing_zero	Controls configuration item for numeric display.	Primary/Standby and distributed	Yes	
	rownum_type_compat	Specifies the ROWNUM type.	Primary/Standby	Yes	
	aformat_null_test	Specifies the logic for checking whether the row type is not null.	Primary/Standby and distributed	Yes	

Configuration item (Compatible with A Mode)	Compatibility Configuration Item	Compatibility Configuration Item Name	Supported Database	Enable	Description
	aformat_regexp_match	Determines the matching behavior of regular expression functions.	Primary/Standby and distributed	Yes	
	compat_cursor	Determines the compatibility behavior of implicit cursor states.	Primary/Standby	Yes	
	proc_outparam_override	Determines the overloading of output parameters of a stored procedure.	Primary/Standby and distributed	Yes	
	proc_implicit_for_loop_variable	Controls the behavior of the FOR_LOOP query statement in a stored procedure.	Primary/Standby	Yes	
	allow_procedure_compile_check	Controls the compilation check of the SELECT and OPEN CURSOR statements in a stored procedure.	Primary/Standby	Yes	
	plsql_security_definer	Determines whether the definer permission is used by default when a stored procedure is created.	Primary/Standby and distributed	Yes	

Configur ation item (Compat ible with A Mode)	Compatibili ty Configurati on Item	Compatibility Configuration Item Name	Supp orted Datab ase	Enab le	Description
	plpgsql_depe ndency	Determines whether a function, stored procedure, or package containing undefined objects can be created.	Prima ry/ Stand by	Yes	
plsql_co mpile_ch eck_optio ns	plsql_express ion_check	If the plsql_expression_c heck parameter is enabled, the plpgsql_dependenc y parameter must also be enabled.	503.1 prima ry/ stand by	Yes	
a_format _version	10C	Specifies database platform compatibility configuration items.	Prima ry/ Stand by and distrib uted	Yes	
a_format _dev_vers ion	S1	Specifies the compatible minor version of the database platform.	503.0 prima ry/ Stand by and distrib uted	Yes	
	S2	Specifies the compatible minor version of the database platform.	503.1 prima ry/ Stand by and distrib uted	Yes	

Configuration item (Compatible with A Mode)	Compatibility Configuration Item	Compatibility Configuration Item Name	Supported Database	Enable	Description
	S3	Specifies the compatible minor version of the database platform.	503.2 primary/standby and distributed	Yes	
sql_beta_feature	a_style_coerce	Affects the decode function.	Primary/Standby and distributed	Yes	

 NOTE

a_format_dev_version is set to **s1** for GaussDB 3.1 enterprise edition, **s2** for GaussDB 3.2 enterprise edition, and **s3** for GaussDB 3.3 enterprise edition. For details about the version numbers, see [What Is the Mapping Between the GaussDB Versions Displayed on the UGO Console and Self-built GaussDB Versions?](#)

Parameter Settings

- For details about how to set cloud-based GaussDB GUC parameters, see [Setting Parameters](#) in the primary/standby deployment and [Setting Parameters](#) in the distributed deployment.
- For details about how to set on-premises GaussDB GUC parameters, see [Setting Parameters](#) in the primary/standby deployment and [Setting Parameters](#) in the distributed deployment.

Procedure

- Primary/Standby

Step 1 Connect to the GaussDB database and switch to the GaussDB installation user.

su - omm

 NOTE

In the preceding command, **omm** is an example. Change it to the actual installation user.

Step 2 Run the following command to configure the GUC parameters:

```
gs_guc reload -Z coordinator -Z datanode -N all -I all -c
"behavior_compat_options='bind_procedure_searchpath,truncate_numeric_tail
_zero,plsql_security_definer,proc_outparam_override,aformat_null_test,rownu
m_type_compat,allow_procedure_compile_check,proc_implicit_for_loop_variab
le,plstmt_implicit_savepoint,end_month_calculate,disable_rewrite_nesttable,pl
pgsql_dependency,display_leading_zero,correct_to_number,unbind_divide_bou
nd,convert_string_digit_to_numeric,hide_tailing_zero,return_null_string,aform
at_regexp_match,compat_cursor,enable_funcname_with_argsname,tableof_ele
m_constraints,merge_update_multi";
```

```
gs_guc reload -Z coordinator -Z datanode -N all -I all -c
"plsql_compile_check_options='plsql_expression_check";
```

NOTE

It is applied for GaussDB 3.1 enterprise edition and later versions.

```
gs_guc reload -Z coordinator -Z datanode -N all -I all -c
"a_format_version='10c";
```

```
gs_guc reload -Z coordinator -Z datanode -N all -I all -c
"a_format_dev_version='s4";
```

NOTE

a_format_dev_version is set to **s1** for GaussDB 3.1 enterprise edition, **s2** for GaussDB 3.2 enterprise edition, **s3** for GaussDB 3.1 enterprise edition, and **s4** for GaussDB 8.0 enterprise edition. For details about the versions, see [What Is the Mapping Between the GaussDB Versions Displayed on the UGO Console and Self-built GaussDB Versions?](#).

----End

- Distributed

Step 1 Connect to the GaussDB database and switch to the GaussDB installation user.

```
su - omm
```

NOTE

In the preceding command, **omm** is an example. Change it to the actual installation user.

Step 2 Run the following command to configure the GUC parameters:

```
gs_guc reload -Z coordinator -Z datanode -N all -I all -c
"behavior_compat_options='bind_procedure_searchpath,truncate_numeric_tail
_zero,plsql_security_definer,proc_outparam_override,aformat_null_test,plstmt_
implicit_savepoint,end_month_calculate,disable_rewrite_nesttable,display_lea
ding_zero,correct_to_number,unbind_divide_bound,convert_string_digit_to_nu
meric,hide_tailing_zero,return_null_string,aformat_regexp_match,enable_func
name_with_argsname,tableof_elem_constraints,merge_update_multi";
```

```
gs_guc reload -Z coordinator -Z datanode -N all -I all -c
"a_format_version='10c";
```

```
gs_guc reload -Z coordinator -Z datanode -N all -I all -c
"a_format_dev_version='s4";
```

 NOTE

s1 is used for 503.0, s2 is used for 503.1, s3 is used for 503.2, and s4 is used for 505.0. For details about the versions, see [What Is the Mapping Between the GaussDB Versions Displayed on the UGO Console and Self-built GaussDB Versions?](#)

----End

4.8.2 Setting GUC Parameters for GaussDB When the Source Database Type Is MySQL

When migrating data from MySQL to GaussDB, you can set the GUC parameters of GaussDB to enhance the compatibility. For details about the GUC parameters involved in the migration and the impact scope, see [Table 4-3](#).

 CAUTION

GaussDB provides many operation parameters. Configurations of these parameters affect the behavior of the database system. Before changing these parameters, learn the impact of these parameters on the database. Otherwise, unexpected results may occur.

Table 4-3 GUC parameters for migrating data from MySQL to GaussDB

Configurati on item (Compatibl e with B Mode)	Compatibility Configuration Item	Compatibility Configuration Item Name	Supported Database	Enabl e
b_format_behavior_compat_options	enable_set_variables	Takes effect only for primary/standby session variables.	Primary/Standby	Yes
b_format_version	5.7	Specifies database platform compatibility configuration items.	505.0 primary/standby and distributed	Yes
b_format_dev_version	S1	Specifies the compatible minor version of the database platform.	505.0 primary/standby and distributed	Yes
	S2	Specifies the compatible minor version of the database platform.	505.1 primary/standby and distributed	Yes

- The S1 compatibility configuration item affects the following content:

- NEW() function, last_day() function, date_sub(date, INTERVAL expr unit), datediff(expr1, expr2), day()/dayofmonth(), and dayname()
- dayofweek(), dayofyear(), extract(unit FROM date), from_days(days), from_unixtime(unix_timestamp[,format]), get_format({DATE | TIME | DATETIME | TIMESTAMP}, {'EUR' | 'USA' | 'JIS' | 'ISO' | 'INTERNAL'}), hour(), makedate(year,dayofyear), microsecond(), minute(), month(), monthname(), period_add(period, month_number)
- TIMESTAMPDIFF, yearweek(date[, mode]), year(), weekofyear(date), weekday(), week(date[, mode]), utc_timestamp(), utc_time(), period_diff(p1,p2), second()
- quarter(), str_to_date(str, format), subdate(expr, days), time_format(time, format), ifnull(expr1, expr2), database(), current_date, current_time, current_timestamp
- round(arg1, arg2), localtime([precision]), localtimestamp, dbtimezone, timenow(), numtodsinterval(num, interval_unit), numtoyminterval(num, interval_unit)
- new_time(date, timezone1,timezone2), sysdate([precision]), add_months(d,n), months_between(d1, d2), convert_tz(dt, from_tz, to_tz)
- adddate(date, INTERVAL expr unit), date_format(date, format)
- The S2 compatibility configuration item affects:
 - Compatibility behaviors controlled by s1.
 - The following functions:
 - When data is imported to the **AUTO_INCREMENT** column or the **BATCH INSERT** execution plan is inserted, if **0**, **NULL**, and determined values are used together, the auto-increment count is updated immediately when the determined value is inserted, the subsequent **0** or **NULL** increases automatically based on the determined value.
 - The like operator does not report an error when an escape character is at the end of the matching string.
 - The priorities of the sorting rules for character sets and character orders are changed.
 - The following syntax:
 - The CREATE TABLE table_name LIKE source_table syntax is supported.
 - The syntaxes CREATE TABLE table_name LIKE source_table and CREATE TABLE table_name (LIKE source_table) cannot specify **INCLUDING** and **EXCLUDING** options. By default, **INCLUDING ALL** is specified.
 - The LOAD DATA syntax is supported. Some syntax functions in gs_loader that are consistent with the LOAD DATA syntax will change.
 - The collate clause can be specified by set names.

- The syntax for changing table names, such as ALTER TABLE and RENAME TABLE, is involved. For example, if the character string corresponding to the new table name starts with **#MySQL50#** and is followed by other characters, **#MySQL50#** will be ignored.

Procedure

Primary/Standby

Step 1 Connect to the GaussDB database and switch to the GaussDB installation user.

su - omm

NOTE

In the preceding command, **omm** is an example. Change it to the actual installation user.

Step 2 Run the following commands to configure the GUC parameters:

```
gs_guc reload -Z coordinator -Z datanode -N all -I all -c  
"b_format_behavior_compat_options='enable_set_variables'";
```

```
gs_guc reload -Z coordinator -Z datanode -N all -I all -c  
"b_format_version='5.7'";
```

```
gs_guc reload -Z coordinator -Z datanode -N all -I all -c  
"b_format_dev_version='s1'";
```

NOTE

Set **b_format_dev_version** to **s1** for GaussDB 8.0 Enterprise Edition and **s2** for GaussDB 8.100 Enterprise Edition. For details about the version numbers, see [What Is the Mapping Between the GaussDB Versions Displayed on the UGO Console and Self-built GaussDB Versions?](#)

----End

Distributed

Step 1 Connect to the GaussDB database and switch to the GaussDB installation user.

su - omm

NOTE

In the preceding command, **omm** is an example. Change it to the actual installation user.

Step 2 Run the following commands to configure the GUC parameters:

```
gs_guc reload -Z coordinator -Z datanode -N all -I all -c "b_format_version='5.7'";
```

```
gs_guc reload -Z coordinator -Z datanode -N all -I all -c  
"b_format_dev_version='s1'";
```

 NOTE

Set **b_format_dev_version** to **s1** for GaussDB 8.0 Enterprise Edition and **s2** for GaussDB 8.100 Enterprise Edition. For details about the version numbers, see [What Is the Mapping Between the GaussDB Versions Displayed on the UGO Console and Self-built GaussDB Versions?](#)

----End

4.9 Why Is the Number of Indexes Different After Oracle Data is Migrated to GaussDB?

Symptom

After data is migrated from Oracle to GaussDB, the total number of indexes is different in the source and target databases.

Causes

During UGO collection, the primary key and unique constraints, and indexes generated by the system are filtered out.

Solution

Connect to the GaussDB database and run the following SQL statements to query the number of indexes:

1. Query the number of indexes displayed on the UGO evaluation page:

```
SELECT count(*)
FROM (select owner, index_name, status from ALL_INDEXES WHERE OWNER in ('schema_name')) i l
LEFT OUTER JOIN
( WITH
cons_ind AS -- Primary key and unique key constraints
(SELECT constraint_name
FROM ALL_CONSTRAINTS
WHERE owner in ('schema_name') AND constraint_type IN ('P', 'U') AND constraint_name =
index_name),

cons_col AS --Column information of the constraints
(SELECT ci.constraint_name, cc.position, cc.column_name
FROM cons_ind ci, ALL_CONS_COLUMNS cc
WHERE cc.owner in ('schema_name') AND cc.constraint_name = ci.constraint_name),

ind_col AS -- Column information of the indexes
(SELECT ic.index_name, ic.column_position, ic.column_name
FROM cons_ind ci, ALL_IND_COLUMNS ic
WHERE ic.index_owner in ('schema_name') AND ic.index_name = ci.constraint_name),

not_match_ind AS -- The column information of the constraints does not match that of the indexes.
(SELECT cc.constraint_name, ic.index_name
FROM cons_col cc FULL OUTER JOIN ind_col ic
ON cc.constraint_name = ic.index_name
AND cc.position = ic.column_position
AND cc.column_name = ic.column_name
WHERE cc.constraint_name IS NULL OR ic.index_name IS NULL )

SELECT ci.constraint_name
FROM cons_ind ci
LEFT OUTER JOIN ( SELECT constraint_name FROM not_match_ind
WHERE constraint_name IS NOT NULL
```

```
UNION
SELECT index_name FROM not_match_ind
WHERE index_name IS NOT NULL ) nm
ON ci.constraint_name = nm.constraint_name
WHERE nm.constraint_name IS NULL ) fil_cons_not
ON il.index_name = fil_cons_not.constraint_name
WHERE fil_cons_not.CONSTRAINT_NAME is null;
```

2. Query the total number of primary key and unique constraints.

```
SELECT count(*)
FROM ALL_CONSTRAINTS
WHERE owner in ('schema_name') AND constraint_type IN ('P', 'U') AND constraint_name =
index_name;
```

3. Query the number of objects generated by the system

```
select owner,OBJECT_NAME, object_type as objectType, count(*) as count
from dba_objects
where object_type in ('INDEX')
and owner in ('schema_name')
and OBJECT_NAME not in( SELECT index_name
FROM (select owner, index_name, status from ALL_INDEXES WHERE OWNER in ('schema_name') ) il
LEFT OUTER JOIN
( WITH
cons_ind AS -- Primary key and unique key constraints
(SELECT constraint_name
FROM ALL_CONSTRAINTS
WHERE owner in ('schema_name') AND constraint_type IN ('P', 'U') AND constraint_name =
index_name),
cons_col AS --Column information of constraints
(SELECT ci.constraint_name, cc.position, cc.column_name
FROM cons_ind ci, ALL_CONS_COLUMNS cc
WHERE cc.owner in ('schema_name') AND cc.constraint_name = ci.constraint_name),
ind_col AS -- Column information of the indexes
(SELECT ic.index_name, ic.column_position, ic.column_name
FROM cons_ind ci, ALL_IND_COLUMNS ic
WHERE ic.index_owner in ('schema_name') AND ic.index_name = ci.constraint_name),
not_match_ind AS -- The column information of the constraints does not match that of the indexes.
(SELECT cc.constraint_name, ic.index_name
FROM cons_col cc FULL OUTER JOIN ind_col ic
ON cc.constraint_name = ic.index_name
AND cc.position = ic.column_position
AND cc.column_name = ic.column_name
WHERE cc.constraint_name IS NULL OR ic.index_name IS NULL )
SELECT ci.constraint_name
FROM cons_ind ci
LEFT OUTER JOIN ( SELECT constraint_name FROM not_match_ind
WHERE constraint_name IS NOT NULL
UNION
SELECT index_name FROM not_match_ind
WHERE index_name IS NOT NULL ) nm
ON ci.constraint_name = nm.constraint_name
WHERE nm.constraint_name IS NULL ) fil_cons_not
ON il.index_name = fil_cons_not.constraint_name
WHERE fil_cons_not.CONSTRAINT_NAME is null)
and OBJECT_NAME not in (SELECT constraint_name
FROM ALL_CONSTRAINTS
WHERE owner in ('schema_name') AND constraint_type IN ('P', 'U') AND constraint_name =
index_name)
group by owner, object_type,OBJECT_NAME;
```

NOTE

- The total number of indexes queried in the source Oracle database is equal to the sum of the preceding three parts.
- *schema_name* indicates the name of the schemas to be migrated. Replace it with the actual schema name.

4.10 What Should I Do If There Are **ctid**, **xc_node_id**, and **tableoid** Columns in GaussDB?

Symptom

To migrate a heterogeneous database to GaussDB, you need to manually rename the **ctid**, **xc_node_id**, **tableoid**, and **ctrd** columns.

```
ugo=>
ugo=> create table test_row(tableoid int);
ERROR: column name "tableoid" conflicts with a system column name
ugo=>
ugo=> create table test_row1(xc_node_id int);
ERROR: column name "xc_node_id" conflicts with a system column name
ugo=>
ugo=>
ugo=> create table test_row2(ctid int);
ERROR: column name "ctid" conflicts with a system column name
ugo=>
```

Causes

There are the system columns **ctid**, **xc_node_id**, **tableoid**, and **ctrd** both in GaussDB and the source database. You need to manually rename these columns when migrating data to GaussDB.

Solution

Step 1 Run the following command to connect to the GaussDB database:

```
gsql -U username -d database -p 4000 -W pwd
```

NOTE

In the preceding command, *username* indicates the database username, *database* indicates the database name, and *pwd* indicates the password of the username.

Step 2 Run the following SQL statement to rename the columns **ctid**, **xc_node_id**, **tableoid**, and **ctrd**:

```
create table test(CTRD, int);
create table test(XC_NODE_ID,int);
create table test(TABLEOID,int);
create table test(CTRD,int);
```

----End

4.11 Why Is Data Inconsistent When SELECT Statements Without ORDER BY Are Used for Query in Distributed GaussDB?

Symptom

Figure 4-3 There is no ORDER BY in SELECT statements.

```
shiyang202418=>
shiyang202418=> select * from test_keep;
name | id | fatherid
-----+---+-----
C    | 3 |      1
E    | 4 |      2
E    | 4 |      2
A    | 2 |      3
D    | 5 |      1
B    | 2 |      1
(6 rows)

shiyang202418=>
shiyang202418=> select * from test_keep limit 1;
name | id | fatherid
-----+---+-----
A    | 2 |      3
(1 row)

shiyang202418=> select * from test_keep limit 1;
name | id | fatherid
-----+---+-----
C    | 3 |      1
(1 row)

shiyang202418=> select * from test_keep limit 1;
name | id | fatherid
-----+---+-----
C    | 3 |      1
(1 row)

shiyang202418=> select * from test_keep limit 1;
name | id | fatherid
-----+---+-----
C    | 3 |      1
(1 row)

shiyang202418=> select * from test_keep limit 1;
name | id | fatherid
-----+---+-----
C    | 3 |      1
(1 row)

shiyang202418=> select * from test_keep limit 1;
name | id | fatherid
-----+---+-----
B    | 2 |      1
(1 row)

shiyang202418=> select * from test_keep limit 1;
name | id | fatherid
-----+---+-----
C    | 3 |      1
(1 row)
```

Causes

In distributed GaussDB, data is evenly stored on each DN based on distribution keys. If ORDER BY is not specified in a SELECT query, data is randomly read from DNs. Therefore, data consistency cannot be ensured.

Solution

Use ORDER BY in SQL statements for query. The column followed by ORDER BY is the distribution key.

Figure 4-4 SELECT statements with ORDER BY

```
shiyen202418=> \d+ test_keep
Table "public.test_keep"
-----
Column | Type          | Modifiers | Storage | Stats target | Description
-----+-----+-----+-----+-----+-----
name   | character varying(10) |          |          |               |
id      | integer        |          |          |               |
fatherid | numeric(38,0)   |          |          |               |
Has OIDs: no
Distribute By: HASH(name)
Location Nodes: ALL DATANODES
Options: orientation=row, compression=no

shiyen202418=>
shiyen202418=>
shiyen202418=> select * from test_keep order by name limit 1;
name | id | fatherid
-----+---+-----
A    | 2 |      3
(1 row)

shiyen202418=> select * from test_keep order by name limit 1;
name | id | fatherid
-----+---+-----
A    | 2 |      3
(1 row)

shiyen202418=> select * from test_keep order by name limit 1;
name | id | fatherid
-----+---+-----
A    | 2 |      3
(1 row)

shiyen202418=> select * from test_keep order by name limit 1;
name | id | fatherid
-----+---+-----
A    | 2 |      3
(1 row)

shiyen202418=> select * from test_keep order by name limit 1;
name | id | fatherid
-----+---+-----
A    | 2 |      3
(1 row)

shiyen202418=> select * from test_keep order by name limit 1;
name | id | fatherid
-----+---+-----
A    | 2 |      3
(1 row)
```

5 Change History

Released On	Description
2024-03-01	Added the target database GaussDB 8.100 in What Is the Mapping Between the GaussDB Versions Displayed on the UGO Console and Self-built GaussDB Versions? Added How Does UGO Check Database Permissions? Added Setting GUC Parameters for GaussDB When the Source Database Type Is MySQL.
2023-12-30	Deleted the target databases GaussDB 1.4 and 2.0 in What Is the Mapping Between the GaussDB Versions Displayed on the UGO Console and Self-built GaussDB Versions? Added Setting GUC Parameters for GaussDB When the Source Database Type Is Oracle. Added Why Is the Number of Indexes Different After Oracle Data Is Migrated to GaussDB?
2023-11-30	Added the schemas that are ignored when Microsoft SQL Server as the source database in Which Schemas in Source Databases Are Ignored for Migration?
2023-09-30	Added the mapping between GaussDB 3.3 Enterprise Edition and kernel versions in What Is the Mapping Between the GaussDB Versions Displayed on the UGO Console and Self-built GaussDB Versions?
2023-08-30	Added the mapping between GaussDB 8.0 Enterprise Edition and kernel versions in What Is the Mapping Between the GaussDB Versions Displayed on the UGO Console and Self-built GaussDB Versions?
2023-04-30	Added How Does UGO Collect Data from Source Databases?

Released On	Description
2023-03-30	<ul style="list-style-type: none">Added schemas that are ignored during the migration when the source database types are MySQL, DB2 for LUW, and PostgreSQL in Which Schemas in Source Databases Are Ignored for Migration?Added the custom schema name when the source database type is MySQL in What Is the Function of the dsc_ora_ext Schema Generated After Migration to the Target Database?Added the mapping between GaussDB 3.2 Enterprise Edition and kernel versions in What Is the Mapping Between the GaussDB Versions Displayed on the UGO Console and Self-built GaussDB Versions?Added the description about how to create GaussDB databases that are compatible with DB2 for LUW and PostgreSQL databases in How Do I Create GaussDB Databases Compatible with Source Databases?Added How Do I Solve ERROR: syntax error at or near "@"?
2023-01-30	Added: <ul style="list-style-type: none">How Do I Create GaussDB Databases Compatible with Source Databases?What Is the Mapping Between the GaussDB Versions Displayed on the UGO Console and Self-built GaussDB Versions?What Should I Do If An Object Collection Error (Closed Connection) Is Displayed During Evaluation Project Creation?What Should I Do If "ErrorCode=4036" Is Displayed During Evaluation Project Creation?What Should I Do If "ErrorCode=17002" Is Displayed During Evaluation Project Creation?
2022-12-30	Updated the description of What Is Native Supported, UGO Supported, Supported with Risk, or Unsupported Objects?
2022-10-30	Added Why Cannot I Use Some Functions?
2022-06-30	Added Why Is a Name Error Reported During Database Migration?
2022-05-30	This issue is the first official release.