# ApsaraDB HybridDB for PostgreSQL

## Quick Start

# Quick Start

# Overview

ApsaraDB for HybridDB is a distributed cloud database composed of multiple groups and provides MPP (Massively Parallel Processing) data warehousing service. ApsaraDB for HybridDB is developed based on the Greenplum Open Source Database program and incorporates some in-depth extensions by Alibaba Cloud. It is compatible with the Greenplum ecology and supports features including OSS storage, JSON data type, and HyperLogLog approximating analysis. For details about HybridDB features and limitations, see Features and limitations.

You need to create an instance and complete some basic settings to use HybridDB, such as configuring a whitelist, an account and the network, connecting to the database and importing the data.

# Create an instance

You can purchase or create HybridDB instances via the following two methods:

> You can purchase an instance directly on the ApsaraDB for HybridDB Purchase Page of Alibaba Cloud official website.
>
> You can create an instance on the HybridDB Database Management Console.

This article will elaborate the detailed steps for creating a HybridDB instance via the HybridDB console to facilitate your operations for adding or removing instances using the console.

## Billing method

At present, ApsaraDB for HybridDB only supports the “pay-as-you-go” model.

## Prerequisites

You have registered an Alibaba Cloud account. If you haven’ t signed up, go to Alibaba Cloud Official Website to sign up.

## Operation procedures

Log on to the HybridDB management console of Alibaba Cloud.

Click **Create Instance** at the top right corner of the page.

Select the instance configuration. Details of various configuration items are as follows:

Region and zone: for selection of the region and zone, see here.

Engine: type of the database, such as MySQL, PostgreSQL.

Node type: the unit of computing resources. Different node types have different storage spaces and computing capabilities. For node type details, see HybridDB Instance types.

Node: the number of purchased “nodes”, minimum two. Increasing the number of nodes can result in a linear performance increase.

Purchase number: the number of purchased instances, used for purchasing instances in batch.

Select the instance configuration and purchase number, and then click **Buy Now**.

Click **Activate** to confirm to activate the instance.

Go to the “Instance List” page of HybridDB Database Management Console to view the newly created instance.

# Set up instances

# Modify a white list

You must modify the white list before starting an instance. Add the IP address or IP segment of the databases you want to access to the white list to ensure security and stability of databases.

## Background

There are three scenarios for HybridDB database access as below:

Access HybridDB databases from the Internet.

Access HybridDB databases from the Intranet. Ensure that the network types of HybridDB and ECS are consistent.

Access HybridDB databases from the Intranet and Internet at the same time. Ensure that the network types of HybridDB and ECS are consistent, and make sure to set the access mode to **Safe Connection Mode**.

**Note:**

- To set the network type, see Set Network Type.

## Operation procedures

Log on to HybridDB management console.

Select the region where the target instance is located.

Click the ID of the instance to go to the **Basic Information** page of the instance.

In the menu bar of the instance, select **Data Security** to go to the **Data Security** page.

In the **White List Settings** tab, click **Modify** after the default white list group to go to the **Modify White List Group** page.

**Note:** You can also click **Clear** after the default white list group to delete the white list from the default group, and then click **Add White List Group** to create a custom group.

Delete the default white list 127.0.0.1 from in-group white list and then fill in a custom white list. Parameters are described as follows:

**Group name**: The group name contains 2 to 32 characters which consist of lowercase letters, numbers or underscores(_). The group name must start with a lowercase letter and end with a letter or number. The default group cannot be modified or deleted.

**In-group white list**: Fill in the IP address or IP segment that can access the database. IP addresses or IP segments are separated by commas(,).

The white list can contain IP addresses (for example, 10.10.10.1) or IP segments (for example, 10.10.10.0/24, which indicates any IP address in the format of 10.10.10.X can access the database).

% or 0.0.0.0/0 indicates any IP address is allowed to access the database.

> **Note:** This configuration will greatly reduce database security, and thus is not recommended unless necessary.

After an instance is created, the local loopback IP address 127.0.0.1 is set as the default white list, and the external IP addresses are prohibited from accessing this instance.

**Load Intranet IP address of ECS**: Click it to display the ECS belonging to the same account. You can add the ECS to the white list.

Click **OK** to add a white list.

# Subsequent operations

Correct use of the white list can provide more advanced access security protection for HybridDB. For this reason, we recommend you maintain the white list on a regular basis.

During the subsequent operations, you can click **Modify** after the group name to modify an existing group, or click **Delete** to delete an existing custom group.

# Set the account

This document introduces how to create an account and reset the password in the HybridDB instance.

## Create an account

Prior to using HybridDB, you need to create an account in the HybridDB instance.

### Operation procedure

Log on to HybridDB management console.

Select the region where the target instance is located.

Click the ID of the instance to go to the **Basic Information** page of the instance.

Select **Manage Accounts** in the instance menu bar to go to the **Manage Accounts** page.

Click **Create an Initial Account** to go to the **Create an Account** page.

Fill in the database account and password, and click **OK**.

Database account: 2 to 16 characters, and should contain lowercase letters, numbers or underscores. It must begin with a letter and end with a letter or a number, for example, user4example*.

Password: 8 to 32 characters. It should contain three types of characters, consisting of uppercase letters, lowercase letters, numbers or special characters.

Confirm the password: Enter the password again.

## Reset the password

During the usage of HybridDB, if you forget the password of the database account, you can reset the password in the HybridDB management console.

**Note:** We recommend you change the password on a regular basis for data security

considerations.

## Operation procedure

Log on to HybridDB management console.

Select the region where the target instance is located.

Click the **Manage** button in the action bar of the target instance to go to the **Basic Information** page of the instance.

Select **Manage Accounts** in the instance menu bar to go to the **Manage Accounts** page.

Click **Reset Password** next to the account to be managed to go to the **Reset Account Password** page.

Enter and confirm the new password and click **OK**.

> **Note:** The password is comprised of 8 to 32 characters. It should contain three types of characters, consisting of uppercase letters, lowercase letters, numbers or special characters. A password that has been previously used is not allowed.

# Set network type

Alibaba Cloud ApsaraDB supports two network types: classic network and Virtual Private Cloud (VPC). By default, HybridDB uses the classic network. If you want to use VPC, ensure that the HybridDB instance and the VPC are in the same region. This article mainly describes the differences between the two network types and how to configure the settings.

## Background

On the Alibaba Cloud platform, the classic network and VPC have the following differences:

Classic network: The cloud service on the classic network is not isolated, and unauthorized access can only be blocked by the white list policy of the cloud service.

Virtual Private Cloud (VPC): VPC helps you to build an isolated network environment on the Alibaba Cloud. You can customize the routing table, IP address range and gateway in the VPC. You can also combine your self-built machine room and cloud resources on the Alibaba Cloud VPC into a virtual machine room through a leased line or VPN to migrate applications to the cloud seamlessly.

## Operation procedure

Create a VPC in the same region with the target HybridDB instance. For detailed steps, see Create a VPC.

Log on to HybridDB management console.

Select the region where the target instance is located.

Click the ID of the instance to go to the **Basic Information** page of the instance.

Select **Database Connection** in the instance menu bar to go to the **Data Connection** page.

Click **Switch to VPC** to go to the **Switch to VPC** option page.

Select a VPC and virtual switch, and click **OK**.

**Note:** After the network is switched to VPC, the original Intranet address will be changed from a classic network address to a VPC address. ECS on the classic network won’ t be able to access the HybridDB instance in the VPC and the original Internet address remains unchanged.

# Connect to HybridDB database

Greenplum Database is developed based on PostgreSQL 8.2 branch and is fully compatible with the message protocols of PostgreSQL 8.2. HybridDB is also based on this version. Therefore, in theory, HybridDB users can directly use tools that support the message protocols of PostgreSQL 8.2 , such as libpq, JDBC, ODBC, Psycopg 2 and pgAdmin III.

HybridDB provides the download link for binary psql program of Redhat platform. The official website of Greenplum also provides an installer containing JDBC, ODBC and libpq to facilitate the

installation and use.

# psql

Psql is a common tool of Greenplum. It provides a variety of commands and its binary file is located in the BIN directory after Greenplum installation. The steps for using the tool are shown as below:

Connect via any method below:

Catenated string

```
 psql "host=yourgpdbaddress.gpdb.rds.aliyuncs.com port=3568 dbname=postgres
user=gpdbaccount password=gpdbpassword"

postgres=> select version();
version
-----------------------------------------------------------------------------------------------
PostgreSQL 8.3devel (Greenplum Database 4.3.99.00 build dev) compiled on Jun 26 2016
23:44:59
(1 row)
```

Specified parameter

```
psql  -h yourgpdbaddress.gpdb.rds.aliyuncs.com -p 3568 -d postgres -U gpdbaccount
```

Enter the password to enter the psql shell interface.

```
 postgres=> select version();
version
-----------------------------------------------------------------------------------------------
PostgreSQL 8.3devel (Greenplum Database 4.3.99.00 build dev) compiled on Jun 26 2016 23:44:59
(1 row)
```

Parameter descriptions:

-h: specifies the host address.

-p: specifies the port number.

-d: specifies the database (the default database is postgres).

-U: specifies the connected user.

You can view more options through ‘psql —help’. You can execute ‘\?’ to view more supported psql commands.

## Reference

For more usage of Greenplum psql, see “psql”.

You can also use the PostgreSQL psql command, but please note the difference in usage details. For details, see “PostgreSQL 8.3.23 Documentation — psql”.

## pgAdmin III

pgAdmin III is a GUI client of PostgreSQL, which can be used to directly connect to HybridDB. For details, see here.

You can download pgAdmin III 1.6.3 from the official website of PostgreSQL. pgAdmin III 1.6.3 supports a variety of platforms, such as Windows, MacOS and Linux. For other GUI clients, see here.

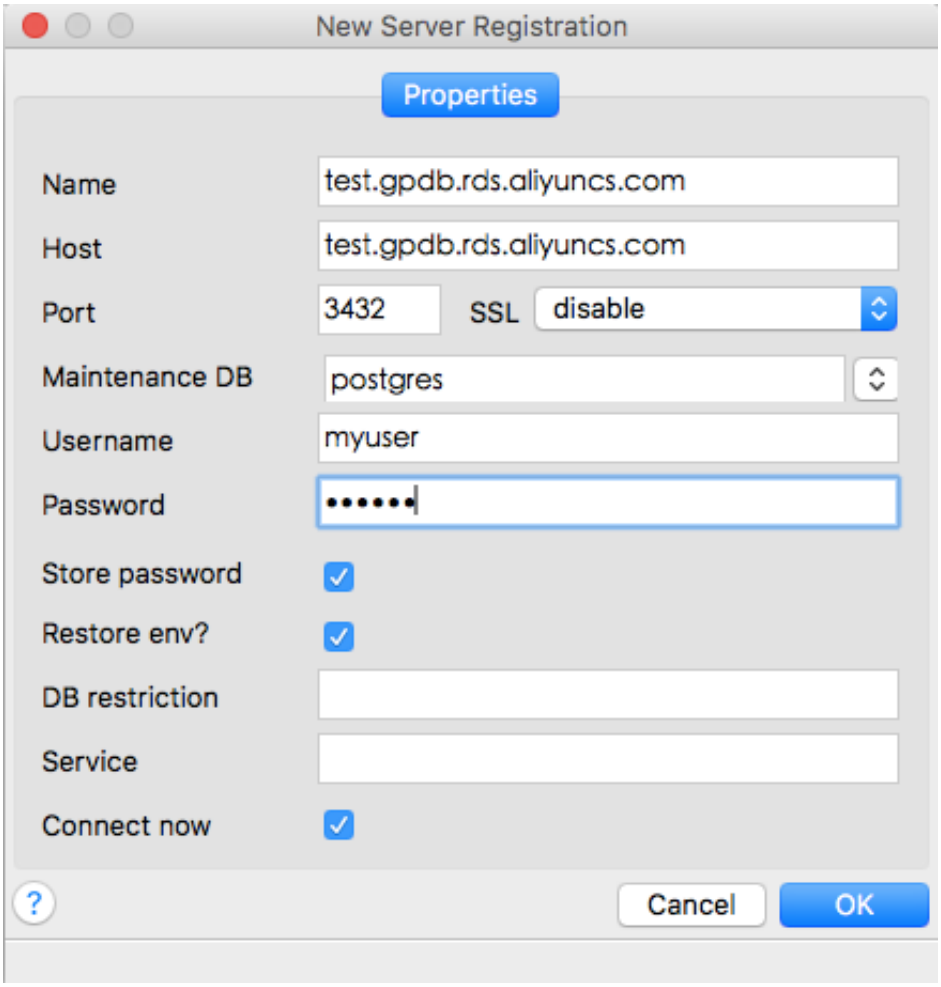
**Note:** HybridDB is compatible with PostgreSQL 8.2. Therefore, you need to use pgAdmin III 1.6.3 or earlier versions to connect to HybridDB. pgAdmin 4 does not support either.

### Operation procedures

Download and install pgAdmin III 1.6.3 or an earlier version.

Select **File** > **Add Server** to enter the **New Server Registration** page.

Fill in the information as the following picture:




Click **OK** to connect to HybridDB.

# JDBC

You can adopt PostgreSQL JDBC. The download method is as follows:

Click [here] (**https://jdbc.postgresql.org/** "here") to download the official PostgreSQL JDBC and add the downloaded JDBC to the environment variable.

You can also adopt the tool package provided by the Greenplum official website. For details, see "**Greenplum Database 4.3 Connectivity Tools for UNIX**".

Code case:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
```

```
import java.sql.SQLException;
import java.sql.Statement;

public class gp_conn {

public static void main(String[] args) {
try {
Class.forName("org.postgresql.Driver");
Connection db =
DriverManager.getConnection("jdbc:postgresql://mygpdbpub.gpdb.rds.aliyuncs.com:3568/postgres","mygpdb","my
gpdb");

Statement st = db.createStatement();
ResultSet rs = st.executeQuery("select * from gp_segment_configuration;");
while (rs.next()) {
System.out.print(rs.getString(1));
System.out.print(" | ");
System.out.print(rs.getString(2));
System.out.print(" | ");
System.out.print(rs.getString(3));
System.out.print(" | ");
System.out.print(rs.getString(4));
System.out.print(" | ");
System.out.print(rs.getString(5));
System.out.print(" | ");
System.out.print(rs.getString(6));
System.out.print(" | ");
System.out.print(rs.getString(7));
System.out.print(" | ");
System.out.print(rs.getString(8));
System.out.print(" | ");
System.out.print(rs.getString(9));
System.out.print(" | ");
System.out.print(rs.getString(10));
System.out.print(" | ");
System.out.println(rs.getString(11));
}
rs.close();
st.close();
} catch (ClassNotFoundException e) {
e.printStackTrace();
} catch (SQLException e) {
e.printStackTrace();
}
}
}
```

For detailed documentation, see “The PostgreSQL JDBC Interface”.

# Python

Python adopts the Psycopg 2 library to connect to Greenplum and PostgreSQL. The steps for using the tool are shown as below:

Install Psycopg 2. In CentOS, there are three installation methods:

Method 1: execute the following command:

```
yum -y install python-psycopg2
```

Method 2: execute the following command:

```
pip install psycopg2
```

Install from source code:

```
 yum install -y postgresql-devel*

wget http://initd.org/psycopg/tarballs/PSYCOPG-2-6/psycopg2-2.6.tar.gz
tar xf psycopg2-2.6.tar.gz
cd psycopg2-2.6
python setup.py build
sudo python setup.py install
```

After the installation, set the PYTHONPATH and you can then reference it, such as:

```
 import psycopg2

sql = 'select * from gp_segment_configuration;'

conn = psycopg2.connect(database='gpdb', user='mygpdb', password='mygpdb',
host='mygpdbpub.gpdb.rds.aliyuncs.com', port=3568)

conn.autocommit = True
cursor = conn.cursor()
cursor.execute(sql)
rows = cursor.fetchall()

for row in rows:
print row

conn.commit()
conn.close()
```

You will get a result similar to the below:

```
(1, -1, 'p', 'p', 's', 'u', 3022, '192.168.2.158', '192.168.2.158', None, None)
(6, -1, 'm', 'm', 's', 'u', 3019, '192.168.2.47', '192.168.2.47', None, None)
(2, 0, 'p', 'p', 's', 'u', 3025, '192.168.2.148', '192.168.2.148', 3525, None)
(4, 0, 'm', 'm', 's', 'u', 3024, '192.168.2.158', '192.168.2.158', 3524, None)
```

```
(3, 1, 'p', 'p', 's', 'u', 3023, '192.168.2.158', '192.168.2.158', 3523, None)
(5, 1, 'm', 'm', 's', 'u', 3026, '192.168.2.148', '192.168.2.148', 3526, None)
```

# libpq

Libpq is the C language interface of the PostgreSQL database. You can access the PostgreSQL database in a C program through libpq for database operations. After Greenplum or PostgreSQL has been installed, you can find its static and dynamic libraries under the lib directory.

For related cases, see [here] (http://www.postgresql.org/docs/8.3/static/libpq-example.html "here"). No further citations will be made.

For libpq details, see "[PostgreSQL 9.4.10 Documentation - Chapter 31. libpq - C Library] (http://www.postgresql.org/docs/9.4/static/libpq.html "PostgreSQL 9.4.10 Documentation - Chapter 31. libpq - C Library")".

# ODBC

PostgreSQL ODBC is an open-source version based on the LGPL (GNU Lesser General Public License) protocol. You can download it from the PostgreSQL official website.

## Operation procedures

Install the driver.

```
yum install -y unixODBC.x86_64
yum install -y postgresql-odbc.x86_64
```

Check the driver's configuration.

```
cat /etc/odbcinst.ini
# Example driver definitions

# Driver from the postgresql-odbc package
# Setup from the unixODBC package
[PostgreSQL]
Description = ODBC for PostgreSQL
Driver = /usr/lib/psqlodbcw.so
Setup = /usr/lib/libodbcpsqlS.so
Driver64 = /usr/lib64/psqlodbcw.so
Setup64 = /usr/lib64/libodbcpsqlS.so
FileUsage = 1

# Driver from the mysql-connector-odbc package
```

```
# Setup from the unixODBC package
[MySQL]
Description = ODBC for MySQL
Driver = /usr/lib/libmyodbc5.so
Setup = /usr/lib/libodbcmyS.so
Driver64 = /usr/lib64/libmyodbc5.so
Setup64 = /usr/lib64/libodbcmyS.so
FileUsage = 1
```

Configure DSN as the following code. Change **** in the code to the actual connection information.

```
[mygpdb]
Description = Test to gp
Driver = PostgreSQL
Database = ****
Servername = ****.gpdb.rds.aliyuncs.com
UserName = ****
Password = ****
Port = ****
ReadOnly = 0
```

Test connectivity.

```
echo "select count(*) from pg_class" | isql mygpdb
+---------------------------------------+
| Connected! |
| |
| sql-statement |
| help [tablename] |
| quit |
| |
+---------------------------------------+
SQL> select count(*) from pg_class
+----------------------+
| count |
+----------------------+
| 388 |
+----------------------+
SQLRowCount returns 1
1 rows fetched
```

ODBC has connected to the instance. Connect the application to ODBC. For details, see here and psqlODBC HOWTO - C#.

# Reference

Pivotal Greenplum Official Documentation

PostgreSQL psql ODBC

PostgreSQL ODBC Compilation

Greenplum ODBC Download

Greenplum JDBC Download

# Import data

# Import data to OSS in parallel

HybridDB supports importing data to or exporting data from OSS in parallel through OSS external tables (that is, the gpossext feature), and supports compressing the OSS external table files through gzip to cut storage space and cost.

## Create an extension for OSS external tables (oss_ext)

Before using OSS external tables, you need to create an oss_ext extension first (Each database needs to create an oss_ext separately). The commands for creating and deleting oss_ext are shown as follows:

```
CREATE EXTENSION IF NOT EXISTS oss_ext;

DROP EXTENSION IF EXISTS oss_ext;
```

## Import data in parallel data

Execute the following operations to import data:

Scatter and store the data evenly among multiple OSS files. The number of files is preferably an integral multiple of the number of HybridDB data nodes (number of

Segments).

Create a READABLE external table in HybridDB.

Execute the following command to import data in parallel.

```
INSERT INTO <target table> SELECT * FROM <external table>
```

# Export data in parallel

Execute the following operations to export data:

Create a WRITABLE external table in HybridDB.

Execute the following command to export data to OSS in parallel.

```
INSERT INTO <external table> SELECT * FROM <source table>
```

# Create syntax for OSS external tables

Execute the following command to create syntax for OSS external tables:

```
CREATE [READABLE] EXTERNAL TABLE tablename
( columnname datatype [, ...] | LIKE othertable )
LOCATION ('ossprotocol')
FORMAT '...'
[ ENCODING 'encoding' ]
[ [LOG ERRORS [INTO error_table]] SEGMENT REJECT LIMIT count
[ROWS | PERCENT] ]

CREATE WRITABLE EXTERNAL TABLE table_name
( column_name data_type [, ...] | LIKE other_table )
LOCATION ('ossprotocol')
FORMAT '...'
[ ENCODING 'encoding' ]
[ DISTRIBUTED BY (column, [ ... ] ) | DISTRIBUTED RANDOMLY ]

ossprotocol:
oss://oss_endpoint prefix=prefix_name
id=userossid key=userosskey bucket=ossbucket compressiontype=[none|gzip] async=[true|false]

ossprotocol:
oss://oss_endpoint dir=[folder/[folder/]...]/file_name
id=userossid key=userosskey bucket=ossbucket compressiontype=[none|gzip] async=[true|false]
```

```
ossprotocol:
oss://oss_endpoint filepath=[folder/[folder/]...]/file_name
id=userossid key=userosskey bucket=ossbucket compressiontype=[none|gzip] async=[true|false]
```

# Parameters description

## Common parameters

Protocol and endpoint: The format is "protocol name://oss_endpoint." The protocol name is oss, and oss_endpoint is the domain name of the corresponding OSS region.

> **Note:** If the access request is from an Alibaba Cloud host, you should use the Intranet domain name (that is, with "internal" in the domain name) to avoid incurring public data usage.

id: OSS account ID.

key: OSS account key.

bucket: Specifies the bucket where the data file is located. It should be pre-created through OSS.

prefix: Specifies the prefix of the corresponding path name of the data file. The prefix does not support regular expressions and is only a matching prefix. In addition, it is mutually exclusive with filepath and dir. So you cannot specify them at the same time.

If you create a READABLE external table for data import, all the OSS files with this prefix will be imported.

If you have specified prefix=test/filename, all the files below will be imported:

test/filename

test/filenamexxx

test/filename/aa

test/filenameyyy/aa

test/filenameyyy/bb/aa

If you have specified prefix=test/filename/, only the following files will be imported (other files listed above won't be imported):

- test/filename/aa

If you create a WRITABLE external table for data export, a unique file name will be generated automatically based on the prefix to name the exported file.

**Note:** There will be more than one exported file. Every data node will export one or more files. The exported file name format is 'prefix_tablename_uuid.x'. To be specific, the uuid is the generated int64 integer value (time stamp in microsecond), and x is the node ID. HybridDB supports multiple export operations with the same external table. The exported files from every export operation are differentiated by the UUID, and the exported files in the same export operation share the same UUID.

dir: The path of virtual folders in OSS. It is mutually exclusive with the prefix and filepath.

The folder path should end with "/", such as "test/mydir/."

If you use this parameter to create an external table during data importing, all the files under the specified virtual directory will be imported, excluding its subdirectories and files under the subdirectories. Unlike filepath, the dir directory has no naming requirements for files under it.

If you use this parameter to create an external table during data exporting, all the data will be exported to the multiple files under this directory. The output file names follow a format of 'filename.x'. To be specific, the x is a number but may be discontinuous.

filepath: The file name that contains a path in OSS. It is mutually exclusive with the prefix and dir. You can only specify it at the creation of a READABLE external table (that is, only usable during data import).

The file name contains the file path, but does not contain the bucket name.

The file naming rule must follow 'filename' or 'filename.x' during data import.

'x' is required to start from 1 and be continuous. For example, if you want to specify "filepath = filename" and the OSS contains the following files, the imported files will include filename, filename.1 and filename.2. Because filename.3 does not exist, filename.4 won't be imported.

```
filename
filename.1
filename.2
filename.4,
```

## Import mode parameters

async: Whether to enable asynchronous mode to load data or not.

Enable the worker thread to load data from OSS to accelerate the import performance. The default import mode is asynchronous mode.

Asynchronous mode is enabled by default. You can use 'async=false' or 'async=f' parameters to disable it.

Asynchronous mode will consume more hardware resources than the normal mode.

compressiontype: The compression type of the imported files.

If specified to none (default value), it indicates that the imported files are not compressed.

If specified to gzip, it indicates that the imported format is gzip. Currently only the gzip compression format is supported.

## Export mode parameters

oss_flush_block_size: The buffer size for a single data flushed to OSS, 32 MB by default. The value ranges from 1 MB to 128 MB.

oss_file_max_size: The maximum size of the file written to OSS. Once this limit is exceeded, the export will switch to another file to continue data writing. The value is 1,024 MB by default and ranges from 8 MB to 4,000 MB.

In addition, you should pay attention to the following items for the export mode:

WRITABLE is the key word of the external table in the export mode. It should be explicitly specified when you create an external table.

Currently the export mode only supports prefix and dir parameter modes, and does not support filepath.

The DISTRIBUTED BY clause in the export mode enables the data node (Segment) to write data to OSS by the specified distribution key.

## Other general parameters

There are also the following fault-tolerant parameters for the import and export modes:

oss_connect_timeout: Sets the connection timeout. The unit is second and the default value is 10 seconds.

oss_dns_cache_timeout: Sets the DNS timeout. The unit is second and the default value is 60 seconds.

oss_speed_limit: Controls the minimum tolerable rate. The default value is 1,024, that is 1 K.

oss_speed_time: Controls the maximum tolerable time. The default value is 15 seconds.

If you apply the default values for the above parameters, and the transmission speed is slower than 1 K for 15 consecutive seconds, timeout will be triggered. For details, see OSS SDK Error Handling.

All other parameters are compatible with the legacy syntax of Greenplum EXTERNAL TABLE. For specific syntax explanations, see Greenplum External Table Syntax Official Documentation. Such parameters mainly include:

FORMAT: The supported file formats, including text and csv.

ENCODING: The encoding format of the data in the file, such as UTF-8.

LOG ERRORS: Specifies that the clause can ignore erroneous data during the import and write the data into error_table. You can also specify the error reporting threshold using the count parameter.

# Examples

```
# Create an external table for OSS import
create readable external table ossexample
(date text, time text, open float, high float,
low float, volume int)
location('oss://oss-cn-hangzhou.aliyuncs.com
prefix=osstest/example id=XXX
key=XXX bucket=testbucket' compressiontype=gzip)
FORMAT 'csv' (QUOTE '''' DELIMITER E'\t')
ENCODING 'utf8'
LOG ERRORS INTO my_error_rows SEGMENT REJECT LIMIT 5;
create readable external table ossexample
(date text, time text, open float, high float,
low float, volume int)
location('oss://oss-cn-hangzhou.aliyuncs.com
dir=osstest/ id=XXX
key=XXX bucket=testbucket')
FORMAT 'csv'
LOG ERRORS SEGMENT REJECT LIMIT 5;
create readable external table ossexample
(date text, time text, open float, high float,
low float, volume int)
location('oss://oss-cn-hangzhou.aliyuncs.com
filepath=osstest/example.csv id=XXX
key=XXX bucket=testbucket')
FORMAT 'csv'
LOG ERRORS SEGMENT REJECT LIMIT 5;

# Create an external table for OSS export
create WRITABLE external table ossexample_exp
(date text, time text, open float, high float,
low float, volume int)
location('oss://oss-cn-hangzhou.aliyuncs.com
prefix=osstest/exp/outfromhdb id=XXX
key=XXX bucket=testbucket') FORMAT 'csv'
DISTRIBUTED BY (date);
create WRITABLE external table ossexample_exp
(date text, time text, open float, high float,
low float, volume int)
location('oss://oss-cn-hangzhou.aliyuncs.com
dir=osstest/exp/ id=XXX
key=XXX bucket=testbucket') FORMAT 'csv'
DISTRIBUTED BY (date);

# Create a heap table to load data
create table example
(date text, time text, open float,
high float, low float, volume int)
DISTRIBUTED BY (date);

# Load data from ossexample to example in parallel
insert into example select * from ossexample;
```

```
# Export data from example to OSS in parallel
insert into ossexample_exp select * from example;

# We can see from the execution plan below that every segment will participate in the work.
# They pull data from the OSS in parallel and then use the Redistribute Motion execution node
to distribute the hashed data to corresponding segments. Data-receiving segments store the data to the database
through the Insert execution node.
explain insert into example select * from ossexample;
QUERY PLAN
------------------------------------------------------------------------------------------------
Insert (slice0; segments: 4) (rows=250000 width=92)
-> Redistribute Motion 4:4 (slice1; segments: 4) (cost=0.00..11000.00 rows=250000 width=92)
Hash Key: ossexample.date
-> External Scan on ossexample (cost=0.00..11000.00 rows=250000 width=92)
(4 rows)

# We can see from the query plan below that the segment exports local data directly to the OSS without data
redistribution.
explain insert into ossexample_exp select * from example;
QUERY PLAN
------------------------------------------------------------
Insert (slice0; segments: 3) (rows=1 width=92)
-> Seq Scan on example (cost=0.00..0.00 rows=1 width=92)
(2 rows)
```

# Attentions

Apart from the location-related parameters, the rest part of the syntax for creating and using external tables is consistent with that of Greenplum.

The data importing performance is related with the HybridDB cluster resources (CPU, IO, memory, network and so on) and OSS. We recommend you use column stores and compression when creating a table to enjoy optimal importing performance. For example, you can specify the clause "WITH (APPENDONLY=true, ORIENTATION=column, COMPRESSTYPE=zlib, COMPRESSLEVEL=5, BLOCKSIZE=1048576)." For details, see Greenplum Database Tabulation Syntax Official Documentation.

The ossendpoint region should match the HybridDB region to ensure the data importing performance. We recommend you configure OSS and HybridDB in the same region to enjoy the optimal performance. For related information, see OSS Endpoint Information.

# SDK error handling

In the case of import or export operation errors, the error log may display the following information:

code: The HTTP status code of the erroneous request.

error_code: The error code of OSS.

error_msg: The error message of OSS.

req_id: The UUID of the request. If you cannot solve a problem, use the req_id to ask OSS development engineers for help.

For details, see OSS API Error Response. Timeout-related errors can be handled using oss_ext related parameters.

## FAQs

The import is too slow. See the import performance descriptions in the “Attentions” section above.

## Reference

OSS Endpoint Information

OSS Help Page

OSS SDK Error Handling

OSS API Error Response

Greenplum Database External Table Syntax Official Documentation

Greenplum Database Tabulation Syntax Official Documentation

# Import data from MySQL

## mysql2pgsql

The mysql2pgsql tool supports migrating tables in MySQL to HybridDB, Greenplum Database, PostgreSQL or PPAS without storing the data separately. The principle of the tool is connecting to the source MySQL database and the target database at the same time, querying and retrieving the data to be exported in the MySQL database, and then importing the data to the target database using the

COPY command. The tool supports multithreaded import (every worker thread is in charge of importing a part of database tables).

# Parameter configuration

Modify the “my.cfg” configuration file, and configure the source and target database connection information.

The connection information of the source MySQL database is as follows:

**Note:** You need to have the read permission to all user tables in the source MySQL database connection information.

```
[src.mysql]
host = "192.168.1.1"
port = "3306"
user = "test"
password = "test"
db = "test"
encodingdir = "share"
encoding = "utf8"
```

The connection information of the target PostgreSQL database (including PostgreSQL, PPAS and HybridDB) is as follows:

**Note:** You need to have the write permission on the target table in the target PostgreSQL database.

```
[desc.pgsql]
connect_string = "host=192.168.1.1 dbname=test port=5888 user=test password=pgsql"
```

# The usage of mysql2pgsql

The usage of mysql2pgsql is illustrated as follows:

```
./mysql2pgsql -l <tables_list_file> -d -j <number of threads>
```

Parameter descriptions:

-l: Optional parameter, used to specify a text file that contains tables to be synchronized. If

this parameter is not specified, all the tables in the database specified in the configuration file will be synchronized. <tables_list_file>is a file name. The file contains the table set to be synchronized and query conditions on the table. An example of the content format is shown below:

```
table1 : select * from table_big where column1 < '2016-08-05'
table2 :
table3
table4: select column1, column2 from tableX where column1 != 10
table5: select * from table_big where column1 >= '2016-08-05'
```

-d: Optional parameter, indicating to only generate the tabulation DDL statement of the target table without performing actual data synchronization.

-j: Optional parameter, specifying the number of threads used for data synchronization. If this parameter is not specified, five threads will be used concurrently.

# Typical usage

## Full-database migration

The full-database migration operation are shown as below:

Get the DDL statements of the corresponding table on the target end through the command below.

```
./mysql2pgsql -d
```

Create a table on the target based on these DDL statements with the distribution key information added.

Execute the following command to synchronize all tables:

```
./mysql2pgsql
```

This command will migrate the data from all MySQL tables in the database specified in the configuration file to the target. Five threads are used during the process (the default thread number is five) to read and import the data from all tables involved.

## Partial table migration

Edit a new file “tab_list.txt” and insert the following content:

```
t1
t2 : select * from t2 where c1 > 138888
```

Execute the following command to synchronize the specified t1 and t2 tables (Note: For the t2 table, only the data that complies with the c1 > 138888 condition is migrated):

```
./mysql2pgsql -l tab_list.txt
```

## Download the binary installer of mysql2pgsql

Click here to download.

## Instructions of mysql2pgsql source code compilation

To view the source code compilation instructions, click here.

# Import data from PostgreSQL

The pgsql2pgsql tool supports migrating the tables in HybridDB, Greenplum Database, PostgreSQL or PPAS to HybridDB, Greenplum Database, PostgreSQL, or PPAS without storing the data separately.

## Features supported by pgsql2pgsql

pgsql2pgsql supports the following features:

Full data migration from PostgreSQL, PPAS, Greenplum Database or HybridDB to PostgreSQL, PPAS, Greenplum Database, or HybridDB.

Full migration and incremental migration from PostgreSQL or PPAS (9.4 or later version) to PostgreSQL, or PPAS.

## Parameters configuration

Modify the “my.cfg” configuration file, and configure the source and target database connection

information.

The connection information of the source PostgreSQL database is shown below:

**Note:** The user should be preferably the corresponding database owner in the source PostgreSQL database connection information.

```
[src.pgsql]
connect_string = "host=192.168.1.1 dbname=test port=5888 user=test password=pgsql"
```

The connection information of the local temporary PostgreSQL database is shown below:

```
[local.pgsql]
connect_string = "host=192.168.1.1 dbname=test port=5888 user=test2 password=pgsql"
```

The connection information of the target PostgreSQL database is shown below:

**Note:** You need to have the write permission on the target table of the target PostgreSQL database.

```
[desc.pgsql]
connect_string = "host=192.168.1.1 dbname=test port=5888 user=test3 password=pgsql"
```

**Note:**

If you want to perform incremental data synchronization, the connected source database should have the permission to create replication slot.

PostgreSQL 9.4 and later versions support logic flow replication, so it supports the incremental migration if PostgreSQL serves as the data source. The kernel will support logic flow replication only after you enable the following kernel parameters.

wal_level = logical

max_wal_senders = 6

max_replication_slots = 6

# The usage of pgsql2pgsql

## Full-database migration

Execute the following command to perform a full-database migration:

```
./pgsql2pgsql
```

The migration program will migrate the table data of all the users in the corresponding PostgreSQL database by default to PostgreSQL.

## Status information query

Connect to the local temporary database and you will be able to view the status information in a single migration process. The information is put in the db_sync_status table, including the start and end time of the full migration, the start time of the incremental migration and the data situation of incremental synchronization.

# Download the binary installer of mysql2pgsql

Click here to download.

# Instructions of mysql2pgsql source code compilation

To view the source code compilation instructions, click here.

# Import data using the COPY command

You can directly execute the \COPY command to import local text file data to HybridDB. The premise is that the local text file must be formatted, such as a file adopting commas(,), colons(:) or special symbols as the separator.

**Note:**

Parallel writing of massive data is not available because the \COPY command performs serial data writing through the master node. If you require parallel writing of massive data, you should use the OSS-based data importing method instead.

The \COPY command is an operation instruction of PostgreSQL. If you use the database instruction COPY rather than \COPY, you should note that only STDIN is supported and file is not supported. Because the “root user” does not have the superuser permission to perform operations on file files.

Reference of the \COPY command is as follows:

```
\COPY table [(column [, ...])] FROM {'file' | STDIN}
[ [WITH]
[OIDS]
[HEADER]
[DELIMITER [ AS ] 'delimiter']
[NULL [ AS ] 'null string']
[ESCAPE [ AS ] 'escape' | 'OFF']
[NEWLINE [ AS ] 'LF' | 'CR' | 'CRLF']
[CSV [QUOTE [ AS ] 'quote']
[FORCE NOT NULL column [, ...]]
[FILL MISSING FIELDS]
[[LOG ERRORS [INTO error_table] [KEEP]
SEGMENT REJECT LIMIT count [ROWS | PERCENT] ]

\COPY {table [(column [, ...])] | (query)} TO {'file' | STDOUT}
[ [WITH]
[OIDS]
[HEADER]
[DELIMITER [ AS ] 'delimiter']
[NULL [ AS ] 'null string']
[ESCAPE [ AS ] 'escape' | 'OFF']
[CSV [QUOTE [ AS ] 'quote']
[FORCE QUOTE column [, ...]] ]
[IGNORE EXTERNAL PARTITIONS ]
```

**Note:**

HybridDB also supports the execution of COPY statements using JDBC which encapsulates the CopyIn method. For detailed usage, see Interface CopyIn.

For the usage of COPY command, see COPY.

# Client tools

The interface protocol of HybridDB is compatible with the Greenplum Database community edition and PostgreSQL 8.3. You can connect to HybridDB on a Greenplum or PostgreSQL client.

# GUI client tools

For HybridDB, you can directly use the client tools supporting Greenplum, such as SQL Workbench, Navicat Premium, Navicat For PostgreSQL, pgadmin III (1.6.3) and so on.

# Command line client psql

## RHEL 6/7 and CentOS 6/7 platforms

For RHEL 6/7 (Red Hat Enterprise Linux) and CentOS 6/7 platforms, you can download the tool from the address below and unzip the package to use it.

For RHEL 6 or CentOS 6 platforms, click hybriddb_client_package_el6.

For RHEL 7 or CentOS 7 platforms, click hybriddb_client_package_el7.

## Other Linux platforms

The compilation methods for client tools applicable to other Linux platforms are as listed below:

To get the source code, the following two methods are available:

Get the git directory directly (you should install the git tool first).

```
git clone https://github.com/greenplum-db/gpdb.git
cd gpdb
git checkout 5d870156
```

Download the code directly.

```
wget https://github.com/greenplum-db/gpdb/archive/5d87015609abd330c68a5402c1267fc86cbc9e1f.zip
unzip 5d87015609abd330c68a5402c1267fc86cbc9e1f.zip
cd gpdb-5d87015609abd330c68a5402c1267fc86cbc9e1f
```

To compile, the steps are listed as below (the GCC among other compilation tools is required):

```
./configure
make -j32
make install
```

Use psql and pg_dump. The paths of the two tools are listed as follows:

```
psql:
/usr/local/pgsql/bin/psql


pg_dump:
/usr/local/pgsql/bin/pg_dump
```

## Windows and other platforms

For the client tools for Windows and other platforms, go to the Pivotal website to download HybridDB Client.