Distributed Database Middleware

Best Practices

Issue	01
Date	2022-11-10

HUAWEI TECHNOLOGIES CO., LTD.

Trademarks and Permissions

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.

Contents

1 Overview

This document is based on Huawei Cloud DDM practices and will guide you to complete relevant settings and buy a DDM instance that meets your service requirements.

Table 1-1 DDM best practices

Service	Reference	Description
DDM	**Formulating Sharding Rules**	Selecting a sharding key and algorithm when creating a sharded table
	Determining the Number of Shards in a Schema	Determining the number of shards when creating a sharded schema
	Using Broadcast and Unsharded Tables	Broadcast and unsharded tables
	Transaction Models	DDM transaction models
	SQL Standards	SQL Standards
	Migrating an Entire RDS Database to DDM	Migrating an entire RDS database to a DDM database
	Migrating an Entire MyCat Database to DDM	Migrating an entire MyCat database to DDM
	Accessing DDM Using a JDBC Connection Pool	Accessing DDM using the JDBC connection pool to perform data operations
	Logging In to a DDM Instance Using Navicat	Obtaining an EIP and use this EIP to connect to a DDM instance through the Navicat client
	Migrating Data from RDS for MySQL to DDM Using DRS	Migrating data from RDS for MySQL to DDM through DRS

Service	Reference	Description
	Sharding Database and Table Data of an RDS for MySQL Instance	Sharding database and table data of existing RDS for MySQL instances using DDM

2 Formulating Sharding Rules

When creating a logical table and selecting **Sharded Table** for **Table Type**, you need to specify a sharding algorithm and a sharding key.

If a relationship exists between entities on different tables, formulate the same sharding rule for these tables, and select the associated table fields as the sharding keys respectively so that associated data in different tables is stored in the same shard to avoid cross-shard JOIN operations. For example, use the customer ID as the sharding key when creating sharded tables for storing customer information, orders, or order details.

Table 2-1 Sharding keys and algorithms

Sharding Algorithm	Hash		Range	
Sharding Key	Table field	Table field +date function	Table field	Table field +date function
Description	Data is evenly distributed to shards by table field.	Data is evenly distributed to shards by table field and date function. The table field must be **date**, **datetime**, or **timestamp**.	Data is distributed to a specific shard based on the rules defined in algorithm metadata.	Data is distributed to shards by table field and date function based on the rules defined in algorithm metadata. The table field must be **date**, **datetime**, or **timestamp**.

Application Scenarios	Scenarios requiring even data distribution, for example, banking applications where logical entities are customers. In this case, use the table field corresponding to customers (for example, customer account numbers) as the sharding key.	Scenarios requiring data to be split by time (year, month, day, week, or their combinations) , for example, gaming applications. For these applications, use the table field (for instance, player registration time) corresponding to players as the sharding key. Sharding by day, month, or year helps you easily collect and query operation statistics of players for a specified day or month, and helps game vendors conduct big data analysis.	Scenarios with a large number of range operations, for instance, e-commerce applications. If a service scenario is focused on promotional activities and logical entities are activity dates, use the table field corresponding to activity dates (for example, activity name and date range) as the sharding key. This helps you collect statistics about the sales volume for a specified cycle.	Scenarios involving many different types of complicated information. For example, for log analysis, you can select the time field as the sharding key and then shard data using the date function. To make it easier to clear and dump logs, select the range algorithm and convert the time field value into "year" using the date function so that logs are stored in shards by year. For details, see examples in the following passages.

Selecting a Sharding Algorithm

A sharding algorithm partitions data from logical tables to multiple shards. DDM supports hash and range algorithms.

- Hash

  Hash evenly distributes data across shards.

  Select this algorithm if operators = and **IN** need to be frequently used in SQL queries.

- Range

  Range stores records in tables based on the range specified in algorithm metadata.

Select this algorithm if operators greater (>), less (<), and BETWEEN ... AND ... need to be frequently used in SQL queries.

⚠ CAUTION

If the sharding algorithm is a range algorithm and a DATE function and the sharding key field indicates the creation time, hotspot issues may occur when data is imported to the database. As a result, the advantages of multiple MySQL databases cannot be fully utilized.

Select an appropriate algorithm based on your service requirements to improve efficiency.

Selecting a Sharding Key

A sharding key is a table field used to generate a route during horizontal partitioning of logical tables. After specifying a table field, you can select a date function or manually enter *date function (field name)*. The table field must be **date**, **datetime**, or **timestamp**. Select a date function if data needs to be redistributed by year, month, day, week, or some combinations thereof.

DDM calculates routes based on the sharding key and sharding algorithm, horizontally partitions data in sharded tables, and then redistributes it to shards.

Note that when you select a sharding key and a sharding algorithm:

- Ensure that data is evenly distributed to each shard as much as possible.
- Select the most frequently used field or the most important query condition as the sharding key.
- Prioritize the primary key as the sharding key to keep query the fastest.

Service Scenarios with a Clear Entity

A sharded table generally contains tens of millions of data records. It is extremely important to select an appropriate sharding key and a sharding algorithm. If a logical entity is identified and most database operations are performed on data of that entity, select the table field corresponding to the entity as a sharding key for horizontal partitioning.

Logical entities depend on actual applications. The following scenarios each include a clear logical entity.

1. For customer-related applications of banks, the service logical entities are customers. In this case, use the table field corresponding to customers (for example, customer numbers) as the sharding key. Service scenarios of some systems are based on bank cards or accounts. In such cases, select the bank card or account as the sharding key.
2. For e-commerce applications, if service scenarios are based on products, the service logical entity is products. In this case, use the table field corresponding to products (for example, product code) as the sharding key.
3. Game applications mainly focus on player data, and the service logical entity is players. In this case, use the table field corresponding to players (for example, player ID) as the sharding key.

The following is an example SQL statement for creating tables for bank services:

```
CREATE TABLE PERSONALACCOUNT(
    ACCOUNT VARCHAR(20) NOT NULL PRIMARY KEY,
    NAME VARCHAR(60) NOT NULL,
    TYPE VARCHAR(10) NOT NULL,
    AVAILABLEBALANCE DECIMAL(18, 2) NOT NULL,
    STATUS CHAR(1) NOT NULL,
    CARDNO VARCHAR(24) NOT NULL,
    CUSTOMID VARCHAR(15) NOT NULL
) ENGINE = INNODB DEFAULT CHARSET = UTF8
dbpartition by hash(ACCOUNT);
```

Service Scenarios Without a Clear Entity

If you cannot identify a suitable entity for your service scenario, select the table field that can provide even data distribution as the sharding key.

For example, the log system may contain a wide range of data records. In this case, you can select the time field as the sharding key.

When the time field is selected as the sharding key, you can specify a date function to partition data.

To make it easier to clear and dump logs, select the range algorithm and convert the time field value into "month" using the date function so that logs are stored in shards by month.

Example SQL statement for creating a table:

```
CREATE TABLE LOG(
    LOGTIME DATETIME NOT NULL,
    LOGSOURCESYSTEM VARCHAR(100),
    LOGDETAIL VARCHAR(10000)
)
dbpartition by range(month(LOGTIME)) {
    1 - 2 = 0,
    3 - 4 = 1,
    5 - 6 = 2,
    7 - 8 = 3,
    9 - 10 = 4,
    11 - 12 = 5,
    default = 0
};
```

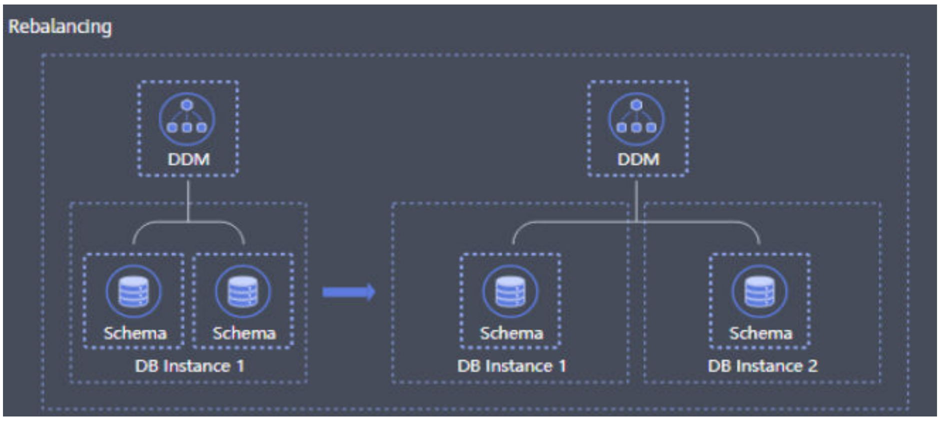
3 Determining the Number of Shards in a Schema

DDM instances support horizontal sharding of tables. After a DDM instance is created, you can create a schema in this instance and specify the shards per MySQL instance and then select appropriate MySQL instances to associate with the schema during the creation. Before creating a schema, you need to first evaluate the expected data volume and determine the shards in a schema. A shard refers to a database in a MySQL instance. The total number of shards in a schema is equal to that of databases in all MySQL instances associated with the schema. There are two schema types:

- Unsharded, a schema corresponds only to one MySQL instance, and only one shard is created for that instance.
- Sharded, a single schema corresponds to multiple MySQL instances. You can specify 1 to 64 shards for each MySQL instance. If more than 64 shards are required, contact DDM technical support.

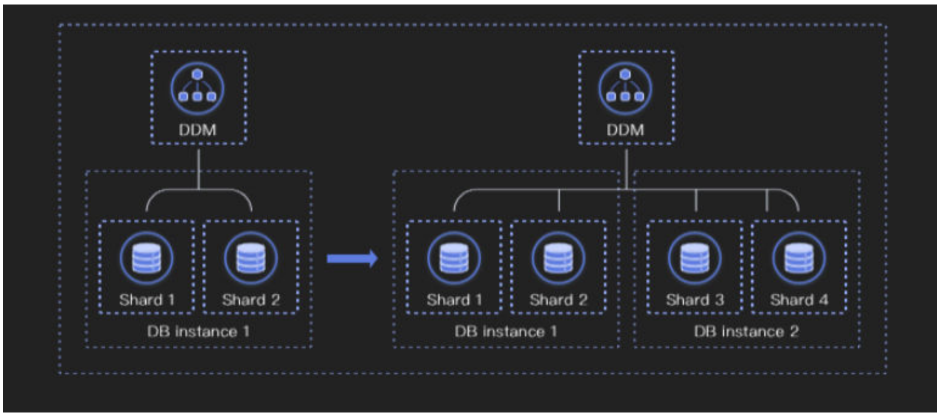
Rebalancing means that database shards are rebalanced between original and new instances.

Figure 3-1 Rebalancing shards



Configuring shards: You can increase shards in a schema based on service requirements. A maximum of 64 shards can be distributed to a single data node. If more than 64 shards are required, contact DDM technical support. DDM distributes shards to all associated data nodes as much as possible. Data will be redistributed as long as the total number of shards changes.

Figure 3-2 Configuring shards



4 Using Broadcast and Unsharded Tables

An unsharded table is one whose data is stored in a single default shard. A broadcast table is a table storing the same data in all shards to improve JOIN efficiency.

Unsharded Table

The DDM console does not allow creation of unsharded tables. To create unsharded tables, you can log in to your DDM instance using a MySQL client or an application.

If a table contains fewer than 10 million data records and has no requests for JOIN operations with other sharded tables for query, set the table type to **Unsharded Table** to store data in the default shard.

The following is an example SQL statement for creating an unsharded table:

```
CREATE TABLE single(
id int NOT NULL AUTO_INCREMENT COMMENT 'Primary key ID',
name varchar(128),
PRIMARY KEY(id)
);
```

Broadcast Table

In business databases, some tables have a small amount of data and are not frequently updated, but they are often used for JOIN operations.

To join such tables and sharded tables, DDM provides broadcast tables, which have the following features:

- The broadcast table stores the same data in each shard. Data insertion, update, and deletion in broadcast tables are executed in each shard in real time.
- Queries of broadcast tables are only executed in one shard.
- Any tables can JOIN with broadcast tables.
- Before using any broadcast hint, ensure that there are available tables.

Example:

When the order management system of an e-commerce enterprise needs to query and collect statistics on order data in Guangdong province (China), JOIN query is performed for a regional table and an order flow table. The order data volume is large and the order flow table needs to be sharded. In this scenario, you can set the regional table as a broadcast table to avoid cross-database JOIN.

The following is an example SQL statement for creating a broadcast table:

```
CREATE TABLE broadcast_tbl (
id int NOT NULL AUTO_INCREMENT COMMENT 'Primary key ID',
name varchar(128),
PRIMARY KEY(id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
BROADCAST;
```

NOTE

Operate broadcast tables or perform full table scanning operations during off-peak hours, such as executing SQL statements that do not contain sharding conditions. Otherwise, an error may occur, indicating that the number of backend RDS connections is insufficient.

5 Transaction Models

Tables in each DDM instance are usually sharded, so data in the tables may be distributed across different database shards in multiple RDS instances. In DDM, one transaction to add, delete, update, or query a logical table is possibly executed on different database shards in multiple RDS instances, so a series of operations on data tables in one shard of an RDS instance can be deemed as a local transaction. Therefore, one DDM transaction is actually a distributed transaction that consists of local transactions on multiple RDS instances. These local transactions either all succeed, or all fail.

Implementation of Distributed Transactions in DDM

DDM executes 2PC distributed transactions based on the MySQL XA protocol, and these transactions can guarantee strong write consistency. For more information about the consistency, see the MySQL official documentation.

In a distributed system, each participant knows whether their operation succeeds or fails, but cannot know the status of the operation on other nodes. If a transaction encompasses multiple nodes, to guarantee atomicity and consistency of a transaction, a coordinator is introduced to control operation results of all participants so that all of the transaction's changes either take effect or do not. The DDM node acts as a coordinator in the distributed transaction, and RDS instances are participants.

The two-phase commit protocol breaks a database commit into two phases:

1. **Prepare phase**: Participants notify the coordinator of the operation result. If the participants give a response that they are prepared, they must reserve the required resources before the coordinator makes a decision.
2. **Commit phase**: After receiving a notification from the participants, the coordinator sends a notification to the participants again and determines whether the participant needs to commit or roll back the operation based on their response.

For example:

Four people A, B, C, and D want to have a dinner party, so they need to determine the time. Now assume that A is the coordinator and B, C, and D are participants.

Prepare phase:

1. A sends a text message to B, C, and D, asking them whether they are available on Tuesday noon.
2. D replies yes.
3. B replies yes.
4. C does not reply for a long time. The time is not determined, so A, B, C, and D cannot continue their dinner party.
5. C replies yes or no.

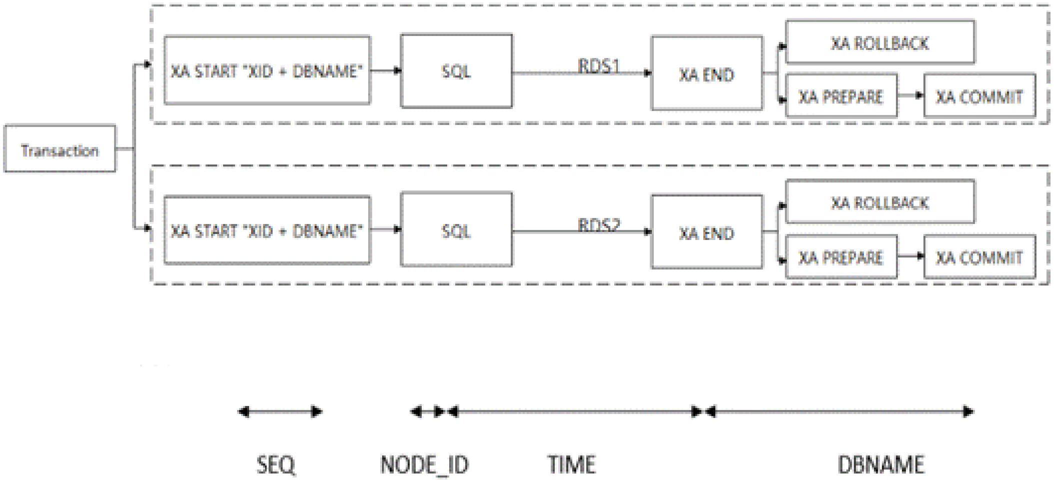
Commit phase:

1. Coordinator A sends the collected results (Tuesday dinner party) to B, C, and D. What the result is and when does A feed it back to the other three depends on the time when C replies.
2. B received.
3. C received.
4. D received.
5. If any of them does not receive the result, A continues to send it until all of them receive the result.

2-Phase Commit

When DDM processes a transaction, applications run BEGIN/COMMIT as they do on a common transaction, without worrying about whether there are distributed transactions at the underlying layer. DDM automatically processes distributed transactions using the two-phase commit (2PC) protocol. If a transaction involves only one data shard, DDM will process it using the one-phase commit protocol. The one-phase commit protocol will not be described in detail here.

Figure 5-1 Flowchart for executing a distributed transaction



XID is the global unique serial number of each distributed transaction. It consists of the transaction serial number, node ID, and timestamp. The XID and the corresponding shard name together constitute the XA_ID that initiates a distributed transaction on a physical database.

Suggestions for Using Transactions

- DDM supports distributed transactions. If all SQL statements in a transaction use the same sharding key, DDM can process the transaction as a single-shard transaction to achieve optimal performance.
- DDM uses the 2PC protocol to process distributed transactions. RDS deadlock detection does not take effect for distributed transactions across RDS instances. If a lock wait times out, there may be a cross-shard deadlock. In this case, check your business model.

6 SQL Standards

INSERT

- **Syntax rules**
  - Do not use INSERT to add data entries one by one. INSERT INTO VALUES (),()..() is recommended.
  - The MySQL JDBC driver ignores the executeBatch() statement by default, splits SQL statements that should be executed in batches into single statements, and sends them to the MySQL database one by one. This greatly deteriorates database performance. To execute SQL statements in batches, you have to set parameter **rewriteBatchedStatements** to **true** and ensure that the SQL JDBC driver is of version 5.1.13 or later. The driver executes SQL statements in batches only when **rewriteBatchedStatements** is set to **true**. This configuration is valid for INSERT, UPDATE, and DELETE operations.

    ⚠ CAUTION

    After you set **rewriteBatchedStatements** to **true**, set a proper value for **batch size** to control the number of INSERT, UPDATE, and DELETE operations. If the **batch size** value is too large, the performance may deteriorate. Unless otherwise specified, the value is not larger than **1000**.

  - Do not set the sharding key value to a function, expression, or sub-query. A constant is recommended.
  - Do not set a common key value to a sub-query. A constant, function, or expression is recommended.
- **Batch data import**

  LOAD DATA LOCAL INFILE is recommended for importing a large volume of data in batches.

  ⚠ CAUTION

  You only need to enable a session, and DDM will automatically imports data.

- **Data migration**

  Use mysqldump to export SQL files and import them by running the MySQL source command.

- **Field auto_increment**
  - DDM can create a unique global sequence of numbers using the AUTO-INCREMENT attribute.
  - If field **auto-increment** is used, do not assign its value in the VALUES clause. Otherwise, a primary key conflict may occur. If a value has been assigned in the VALUES clause, you can change it using ALTER SEQUENCE.
  - Do not set the auto-increment step to 1 to ensure stable performance. The default step is **1000**.

UPDATE and DELETE

- **Common updates**
  - Configure a sharding field in the WHERE condition when you perform an update or delete operation.
  - If the sharding field cannot be configured, reduce concurrency and control data entries involved in updates or deletion operations. You can use SELECT to search for the data that you want to update or delete, determine the data scope with DOUBLE CHECK, and finally perform an update or deletion.
- **Sharding field update**
  - When you update a sharding field, ensure that there is a maximum of 10,000 data entries in the target table. If there are more than 10,000 data entries, create a table with the required sharding field or update the original sharding field by dividing one update operation into multiple small equivalent operations.
  - Update sharding fields during off-peak hours.
- **Association**

  Do not perform an update or delete operation on multiple tables at the same time.

- **Subquery and LIMIT**

  Do not use subqueries in an UPDATE or DELETE statement. Do not use LIMIT or ORDER BY LIMIT in UPDATE or DELETE statements.

SELECT

- **ORDER BY LIMIT function**
  - When you use statement ORDER BY LIMIT, count or ORDER BY OFFSET, count, do not assign a large value for **offset**.
  - If error **Temp table limit exceeded** is returned, a temporary table containing intermediate data is generated and its data entries exceed the upper limit. Contact DDM technical support for SQL tuning.
- **GROUP BY function**
  - Configure only field **group by** in statement SELECT_LIST.

  - Do not use clause ORDER BY in aggregate function **group_concat** that cannot be pushed.
  - Ensure that there are no more than three DISTINCT or GROUP BY fields.
  - Do not perform a GROUP BY operation after a JOIN or subquery operation.
  - Do not use COUNT(DISTINCT) or SUM(DISTINCT).
  - If error **Temp table limit exceeded** is returned, a temporary table containing intermediate data is generated during aggregation and its data entries exceed the upper limit. Contact DDM technical support for SQL tuning.
- **JOIN function**
  - In a SELECT statement, specify a sharding field for each table or a broadcast table in the JOIN condition. Alternatively, use INNER/LEFT JOIN or RIGHT JOIN and ensure that the driving table is the smaller one.
  - Do not join two large tables directly.
  - Do not use OUTER JOIN in the JOIN ON condition.
  - If error **Temp table limit exceeded** is returned, a temporary table containing intermediate data is generated during the JOIN operation and its data entries exceed the upper limit. Contact DDM technical support for SQL tuning.
  - Do not join more than five tables directly.
  - Do not enable transactions when you perform a JOIN query.
  - Do not perform a JOIN query in a transaction. Enabling transactions affects DDM's selection of the most efficient JOIN algorithm.

⚠ CAUTION

The size of tables depends on the volume of data selected using the WHERE condition.

- **Subqueries**
  - Do not use a subquery or its JOIN condition in an OR expression.
  - Do not use scalar subqueries containing LIMIT, for example, SELECT (SELECT x FROM t2 WHERE t2.id= t.id LIMIT 1),a,b FROM t.
  - If subqueries and primary table are routed to the same shard, add **/*+db=xxx*/** before your SQL statement to improve routing accuracy.
  - Do not use JOIN clauses in subqueries.
  - Do not use nested subqueries.
  - Do not perform an operation by comparing ROW expression with a subquery, for example, SELECT * FROM t WHERE (a,b,c)=(SELECT x,y,z FROM t2 WHERE ...).
  - Do not use more than two subqueries in SELECT_LIST.

DDL

- **Execution of DDL statements**

Perform DDL operations on existing tables during off-peak hours.

- **Number of shards**

  Before creating a sharded table, you can estimate the total volume of data to determine table shards. Do not configure more shards than required. The number of table shards is not as large as possible.

- **High-risk DDL statements**

  Carefully check the SQL statement when you perform a high-risk DDL operation, for example, DROP TABLE and TRUNCATE TABLE.

- **Rectification of DDL failures**

  If an error occurs during execution of a DDL statement, execute CHECK TABLE *name* to verify the structure of each table shard, locate the failed table shard, and rectify the fault. For example, if ALTER TABLE fails, add **/*+allow_alter_rerun=true/** before the statement, enable the POWER operation, and execute the statement again till an output is returned, indicating structures of all tables are consistent.

- **DDL execution error caused by MDL locks**
  - **Background**: Before a DDL statement is executed, DDM checks whether there are MDL locks held for tables in the associated RDS database to ensure DDL availability. If there are MDL locks, DDL reports an error and exits.

    ```
    metadata lock exists, one of MDL is [%s],DDL operation can not proceed, please use 'show
    metadata lock' to check current mdl, and use 'kill physical threadId@host:port' to clean it
    ```

  - **Possible issues**: If there are slow SQL statements that have been executed for several minutes, the possible cause may be MDL locks. In this case, DDL statements cannot be executed.
  - **Solution 1**: Set a larger value for **ddl_precheck_mdl_threshold_time** on the DDM console, for example, set it to **30** minutes (1800 seconds).

    NOTE

    ddl_precheck_mdl_threshold_time indicates the maximum duration in seconds for which a DDL statement holds an MDL lock. DDL reports an error only when the lock duration exceeds the threshold. The default value is **120** seconds.

  - **Solution 2**: Execute SHOW METADATA LOCK to check whether DDL execution is blocked by the MDL lock held for a slow transaction. If yes, execute **kill physical threadId@host:port** to disable the underlying slow transaction. Then use hint **/*+allow_alter_rerun=true*/** and CHECK TABLE to complete execution of DDL statements.

    NOTE

    threadId indicates the thread ID of the underlying RDS instance node. **host** and **port** indicate the IP address and port of the RDS instance node.

- **Long execution time of DDL statements**

  If DDL execution is suspended for a long time during off-peak hours, enable another session and execute XA RECOVER to check whether there are slow transactions. If yes, contact technical support.

7 Migrating an Entire RDS Database to DDM

Scenario

This section describes how to migrate an entire RDS instance (the old RDS instance) to an unsharded schema of a DDM instance to split read and write requests in DDM.

NOTE

- Services may be interrupted during migration. The duration of the interruption depends on the amount of data to be migrated and on network conditions.
- Data migration is complicated and is recommended during off-peak hours. This guide is for reference only. The actual migration scheme you should use depends on your specific service scenarios, the amount of data to be migrated, and how much downtime can be tolerated.
- If a large amount of data is involved, contact DDM technical engineers by submitting a service ticket or through after-sales services. Fully rehearse the migration before migrating data.

Preparations before Migration

- Prepare an ECS that can access the old RDS instance, the target DDM instance, and the RDS instance associated with the target DDM instance.
  a. Ensure that the old RDS instance, the target DDM instance, and the RDS instance associated with the target DDM instance are in the same VPC for network connectivity.
  b. Configure the same security group for the above-mentioned instances. If they belong to different security groups, enable the required ports.
  c. Install an official MySQL client on the ECS. Version 5.6 or 5.7 is recommended.
     - Red Hat Enterprise Linux: **yum install mysql mysql-devel**
     - Debian Linux: **apt install mysql-client-5.7 mysql-client-core-5.7**
  d. Ensure that there is enough disk space and memory on the ECS to store and compare dump files.

- Prepare a DDM instance that has been associated with an RDS instance, and configure the required DDM account and DDM schema for it.
- If the target schema is sharded, create a logical table with the same structure as data tables in the old RDS instance on the DDM console.

Constraints

- To ensure data integrity, stop services in the old RDS instance before data migration.
- In this scenario, associating the old RDS instance with DDM does not mean data association. You need to export data from the old RDS instance and then import it into the DDM instance.
- The version of the RDS instance associated with the target DDM instance must be consistent with the MySQL version of the old RDS instance.

Exporting Data from the Old RDS Instance

Step 1 Log in to the target ECS.

Step 2 Run the following command to export structure data. Set parameters in italics to actual values. For details about the parameters, see **Table 7-1**.

```
mysqldump -h {DB_ADDRESS} -P {DB_PORT} -u {DB_USER} -p --skip-lock-tables --add-locks=false --set-gtid-purged=OFF --no-data {DB_NAME} > {mysql_schema.sql}
```

Table 7-1 Parameter description

Parameter	Description	Remarks
DB_ADDRESS	Connection address of the database whose data is to be exported	Mandatory
DB_PORT	Listening port of the database	Mandatory
DB_USER	Database user	Mandatory
DB_NAME	Database name	Mandatory
mysql_schema.sql	Name of the table structure file	The file name varies depending on the table whose structure is exported. You are advised to name the file in the format of *schema name*_**schema** to prevent data from being overwritten, for example, **mysql_schema.sql**.
mysql_data.sql	Name of the database data file	N/A

Step 3 Run the following command to export data of the entire database. Set the parameters in italics. For details, see **Table 7-1**.

```
mysqldump -h {DB_ADDRESS} -P {DB_PORT} -u {DB_USER} -p --hex-blob --complete-insert --
skip-lock-tables --skip-tz-utc --skip-add-locks --set-gtid-purged=OFF --no-create-info
{DB_NAME} > {mysql_data.sql}
```

Step 4 Run the following command on the ECS to view the SQL files exported in **Step 2** and **Step 3**.

```
ls -l
```

----End

Importing Data into DDM

Step 1 Run the following command on the ECS to import the structure file into DDM:

```
mysql -f -h {DDM_ADDRESS} -P {DDM_PORT} -u {DDM_USER} -p {DB_NAME} <
{mysql_schema.sql}
Enter password: **********
```

Table 7-2 Parameter description

Parameter	Description	Remarks
DDM_ADDRESS	Connection address of the DDM instance into which data is to be imported	View the floating IP address and port number on the **Basic Information** tab on the DDM console.
DDM_PORT	Listening port of the DDM instance into which data is to be imported	
DDM_USER	User accessing the DDM instance	Account for creating a schema. The account must have both read and write permissions.
DB_NAME	Name of the schema into which data is to be imported	N/A
mysql_schema.sql	Name of the structure file to be imported	Name of the file exported in **Step 2**
mysql_data.sql	Name of the entire DB data file to be imported	Name of the file exported in **Step 3**

Step 2 Run the following command on the ECS to import the data file into DDM:

```
mysql -f -h {DDM_ADDRESS} -P {DDM_PORT} -u {DDM_USER} -p {DB_NAME} <
{mysql_data.sql}
Enter password: **********
```

----End

8 Migrating an Entire MyCat Database to DDM

Scenario

This section describes how to migrate an entire MyCat database to DDM.

NOTE

- Services may be interrupted during migration. The duration of the interruption depends on the amount of data to be migrated and on network conditions.
- Data migration is complicated and is recommended during off-peak hours. This guide is for reference only. The actual migration scheme you should use depends on your specific service scenarios, the amount of data to be migrated, and how much downtime can be tolerated.
- You can only access DDM through an ECS. Therefore, you must export databases as files to the ECS, and then import data in the files into DDM from the ECS.

Preparations for Migration

- Prepare an ECS that can access the MyCat database, the target DDM instance, and the RDS DB instance associated with the target DDM instance.
  a. Ensure that the MyCat database, the target DDM instance, and the associated RDS DB instance are in the same VPC for stable network connectivity.
  b. Configure the same security group for the ECS where the MyCat database is deployed, the target DDM instance, and the associated RDS DB instance. If they belong to different security groups, configure their security group rules to allow them to access each other.
  c. Install an official MySQL client on the ECS. Version 5.6 or 5.7 is recommended.
     - Red Hat Linux: **yum install mysql mysql-devel**
     - Debian Linux: **apt install mysql-client-5.7 mysql-client-core-5.7**
  d. Ensure that there is enough disk space and memory on the ECS to store and compare dump files.
- Prepare a DDM instance that has been associated with an RDS DB instance, and configure DDM accounts and schemas for it as required.

- Use MyCat 1.6 as an example in this section.

Migration Scheme

Table types in MyCat and DDM are different from each other and migration schemes vary by table type, as described in **Table 8-1**.

Table 8-1 Migration schemes

MyCat Table Type	DDM Table Type	Migration Scheme
Unsharded	Unsharded	1. Export **structure data** and **table data** from MyCat. 2. Connect to the RDS DB instance associated with the target DDM instance and **import data from unsharded tables to the target DDM instance**.
Sharded: hash sharding (including by date)	Sharded: hash sharding (including by date function)	1. Export all table structure data from the MyCat database. 2. Create a table with the same structure as the exported table on the DDM console. 3. **Export data from the entire MyCat database to DDM**. 4. **Connect to DDM to import all database data**.
Sharded: range sharding (including by date)	Sharded: range sharding (including by date function)	
Broadcast	Broadcast	

Constraints

- MyCat services need to be stopped before data migration to ensure data integrity.
- Associating the MyCat database with a DDM instance for data association is not supported. You must export data from the MyCat database and then import the data into the DDM instance.
- The version of the associated RDS DB instance associated must be consistent with that of the MyCat database.

Exporting Table Structure Data from the MyCat Database

Step 1 Log in to the target ECS.

Step 2 Run the following command to export table structure data in the MyCat database. Configure the parameters in italics. For details about the parameters, see **Table 8-2**.

```
mysqldump -h {DB_ADDRESS} -P {DB_PORT} -u {DB_USER} -p  --skip-lock-tables --add-locks=false --set-gtid-purged=OFF --no-data --order-by-primary {DB_NAME} > {mysql_schema.sql}
Enter password: **********
```

Table 8-2 Parameter description

Parameter	Description	Remarks
DB_ADDRESS	Connection address of the database whose data is to be exported	Mandatory
DB_PORT	Listening port of the database	Mandatory
DB_USER	Database user	Mandatory
DB_NAME	Database name	Mandatory
mysql_schema.sql	Name of the table structure file	The file varies depending on the table whose structure is exported. You are advised to name the file in the format of *schema name*_**schema** to prevent data from being overwritten, for example, **mysql_schema.sql**.
mysql_data.sql	Name of the database data file	N/A

----End

Exporting Data from the Entire MyCat Database

Step 1 Log in to the ECS.

Step 2 Run the following command to export data of the MyCat database. Configure the parameters in italics. For details about the parameters, see **Table 8-2**.

```
mysqldump -h {DB_ADDRESS} -P {DB_PORT} -u {DB_USER} -p  --hex-blob --complete-insert --skip-lock-tables --add-locks=false --set-gtid-purged=OFF --quick --no-create-info --order-by-primary {DB_NAME} > {mysql_data.sql}
Enter password: **********
```

Step 3 Run the following command on the ECS to view the exported SQL file.

```
ls -l
```

----End

Importing Data from Unsharded Tables into the Target DDM Instance

On the ECS, use a MySQL client to connect to the associated RDS DB instance. Run the following commands to import the table structure and database files.

For unsharded or common tables:

```
mysql -f -h {RDS_ADDRESS} -P {RDS_PORT} -u {RDS_USER} -p {DB_NAME} <
{mysql_table_schema.sql}
Enter password: **********
mysql -f -h {RDS_ADDRESS} -P {RDS_PORT} -u {RDS_USER} -p {DB_NAME} <
{mysq_table_data.sql}
Enter password: **********
```

- **RDS_ADDRESS** indicates the address of an RDS DB instance into which data is to be imported.
- **RDS_PORT** indicates the port number of an RDS DB instance.
- **RDS_USER** indicates the username of an RDS DB instance.
- **DB_NAME** indicates the name of an RDS database. If data of unsharded tables is to be imported, **DB_NAME** indicates the name of the first shard of RDS.
- **mysql_table_schema.sql** indicates the name of a table structure file to be imported.
- **mysq_table_data.sql** indicates the name of a table data file to be imported.

Importing Data from Sharded or Broadcast Tables into the Target DDM Instance

On the ECS, use a MySQL client to connect to DDM and run the following command to import all database data into DDM.

```
mysql -f -h {DDM_ADDRESS} -P {DDM_PORT} -u {DDM_USER} -p {DB_NAME} <
{mysql_data.sql}
Enter password: **********
```

Table 8-3 Parameter description

Parameter	Description	Remarks
DDM_ADDRESS	Connection address of the DDM instance into which data is to be imported	View the connection address and port number on the **Basic Information** tab on the DDM console.
DDM_PORT	Listening port of the DDM instance into which the data will be imported	
DDM_USER	User accessing the DDM instance	Account for creating a schema. The account must have both read and write permissions.
DB_NAME	Name of the schema that the data will be imported into	N/A

Parameter	Description	Remarks
mysql_data.sql	Name of the entire database file to be imported	Name of the file exported in **Step 2**

9 Accessing DDM Using a JDBC Connection Pool

Scenario

When a connection pool is used, the system stores database connections in memory during initialization. Then, when you send a request to access a database, the system directly selects an available connection in the connection pool. If the connection is not required, the system returns the connection to the pool for another request to access. Connection establishment and disconnection are both managed by the connection pool itself. In addition, you can set connection pool parameters to control the number of initial connections in the connection pool, upper and lower limits of connections, maximum number of times that each connection can be used, and maximum idle time for each connection. If you prefer, you can also use your own system to monitor the number and usage of database connections.

This section describes how to access DDM using the JDBC connection pool to perform data operations. For Java programs, **HikariCP** is recommended.

- Java 8: Version 3.3.1 is recommended.
- Java 7: Version 2.4.13 is recommended.
- Parameter **useCursorFetch** cannot be enabled when you connect to a DDM instance using a JDBC connection.

Procedure

Step 1 Configure Maven.

- Java 8

```
<dependency>
    <groupId>com.zaxxer</groupId>
    <artifactId>HikariCP</artifactId>
    <version>3.3.1</version>
</dependency>
```

- Java 7

```
<dependency>
    <groupId>com.zaxxer</groupId>
    <artifactId>HikariCP-java7</artifactId>
    <version>2.4.13</version>
</dependency>
```

Step 2 Create a table.

Table 9-1 Parameters for creating a table

Table Name	Field	Type	Primary Key
account	account_number	bigint	Yes
	account_type	varchar (45)	No
	account_name	varchar (50)	No

Step 3 Connect to a DDM instance.

1. Set the number of connections by configuring parameters in the JDBC URL and HikariCP parameters.
2. Insert a data record.

Example:

```
package com.huawei.ddm.examples;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.sql.DataSource;
import com.zaxxer.hikari.HikariConfig;
import com.zaxxer.hikari.HikariDataSource;

public class HikariCPDemo {
    private static DataSource datasource;
    private static DataSource getDataSource() {
        if (datasource == null) {
            HikariConfig config = new HikariConfig();

            // Configure parameters in the JDBC URL:
            config.setJdbcUrl("jdbc:mysql:loadbalance://192.168.0.10:5066,192.168.0.11:5066/db_name?
loadBalanceAutoCommitStatementThreshold=5&loadBalanceHostRemovalGracePeriod=15000&loadBal
anceBlacklistTimeout=60000&loadBalancePingTimeout=5000&retriesAllDown=10&connectTimeout=10
000");
            /*
            // Or configure parameters in the JDBC URL as follows:
            config.addDataSourceProperty("loadBalanceAutoCommitStatementThreshold",5);
            config.addDataSourceProperty("loadBalanceHostRemovalGracePeriod", 15000);
            config.addDataSourceProperty("loadBalanceBlacklistTimeout", 60000);
            config.addDataSourceProperty("loadBalancePingTimeout", 5000);
            config.addDataSourceProperty("retriesAllDown", 10);
            config.addDataSourceProperty("connectTimeout", 10000);
            */
            config.setUsername("username");
            config.setPassword("password");
            config.setMaximumPoolSize(10);
            config.setAutoCommit(true);

            // Configure HikariCP parameters.
            config.addDataSourceProperty("cachePrepStmts", true);
            config.addDataSourceProperty("prepStmtCacheSize", 250);
            config.addDataSourceProperty("prepStmtCacheSqlLimit", 2048);
            config.addDataSourceProperty("minimumIdle", 5);
            config.addDataSourceProperty("maximumPoolSize", 10);
            config.addDataSourceProperty("idleTimeout", 30000);

            datasource = new HikariDataSource(config);
        }
        return datasource;
    }

    public static void main(String[] args) {
        Connection connection = null;
        PreparedStatement pstmt = null;
        ResultSet resultSet = null;
        try {
            DataSource dataSource = getDataSource();
            connection = dataSource.getConnection();
            System.out.println("The Connection Object is of Class: " + connection.getClass());
            // Start a transaction.
            connection.setAutoCommit(false);

            // Insert test data.
            String insertSql = "insert into account(account_number, account_type, account_name) 
values(?, ?, ?);";
            PreparedStatement insertStmt = connection.prepareStatement(insertSql);
            insertStmt.setLong (1, 1L);
            insertStmt.setString (2, "manager");
            insertStmt.setString (3, "demotest01");
            insertStmt.executeUpdate();
            connection.commit ();

            // Query data.
            pstmt = connection.prepareStatement("SELECT * FROM account");
            resultSet = pstmt.executeQuery();
            while (resultSet.next()) {
                String accountNumber = resultSet.getString("account_number");
                String accountType = resultSet.getString("account_type");
                String accountName = resultSet.getString("account_name");
                System.out.println(accountNumber + "," + accountType + "," + accountName);
            }
        } catch (Exception e) {
            try {
                if (null != connection) {
                    connection.rollback();
                }
            } catch (SQLException e1) {
                e1.printStackTrace();
            }
            e.printStackTrace();
        }
    }
}
```

----End

10 Logging In to a DDM Instance Using Navicat

Scenario

This section describes how to obtain an EIP and use this EIP to connect to a DDM instance through a Navicat client.

Using Navicat to Connect to a DDM Instance

Step 1 Log in to the DDM console, locate the target DDM instance, and click its name.

Step 2 In the **Instance Information** area, click **Bind**, select an EIP, and bind it with the DDM instance.

Step 3 In the left pane, click the VPC icon and choose **Access Control** > **Security Groups**.

Step 4 On the **Security Group** page, locate the target security group and click **Manage Rule** in the **Operation** column. On the displayed page, click **Add Rule**. Configure the security group rule and click **OK**.

NOTE

- After binding an EIP to your DDM instance, set strict inbound and outbound rules for the security group to enhance database security.

Step 5 Open Navicat and click **Connection**. In the displayed dialog box, enter the host IP address (EIP), username, and password (DDM account and password).

Step 6 Click **Test Connection**. If a message is returned indicating that the connection is successful, click **OK**. The connection will be succeeded 1 to 2 minutes later. If the connection fails, the failure cause is displayed. Modify the required information and try again.

----End

NOTE

Using Navicat to connect to a DDM instance is similar to using other visualized MySQL tools such as MySQL Workbench. The procedure of using other visualized MySQL tools to connect to a DDM instance is omitted here.

11 Migrating Data from RDS for MySQL to DDM Using DRS

11.1 Overview

Scenarios

This practice describes how to migrate data from RDS for MySQL to DDM in a different region using Data Replication Service (DRS), including how to create an RDS for MySQL instance and a DDM instance on Huawei Cloud, and how to migrate data over the VPN network.

Prerequisites

- You have a Huawei Cloud account.
- Your account balance is not below zero.

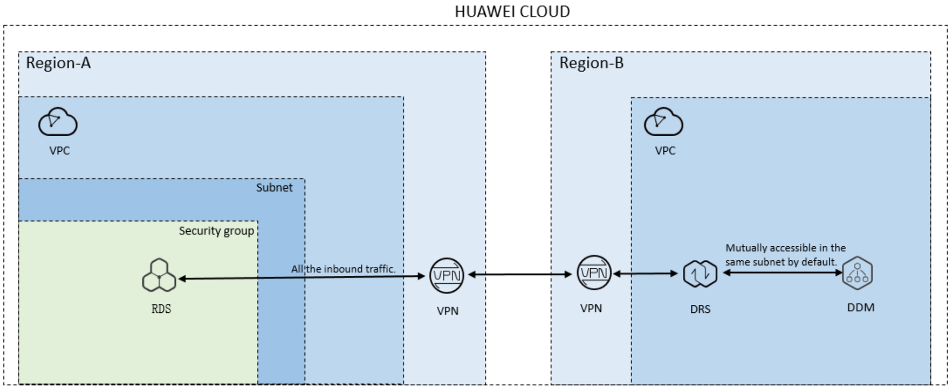
Service List

- Virtual Private Cloud (VPC)
- Virtual Private Network (VPN)
- RDS
- Distributed Database Middleware (DDM)
- Data Replication Service (DRS)
- Data Admin Service (DAS)

Deployment Architecture

In this example, the source is a Huawei Cloud RDS for MySQL instance and the destination is a DDM instance in a different region. Data is migrated from the source to the destination over a VPN. **Figure 11-1** shows the overall deployment architecture.

Figure 11-1 VPN-based deployment architecture



Constraints

- Before data migration, you need to stop your workloads to ensure data integrity.
- The new RDS for MySQL instance must be of the same version as the source RDS for MySQL instance.

Notes on Usage

- The resource planning is for demonstration only. Adjust it as needed.
- The end-to-end test data is for reference only.

11.2 Resource and Cost Planning

Table 11-1 Resource planning

Category	Subcategory	Plan	Remarks
Source VPC	VPC name	vpc-DRSsrc	Specify a name that is easy to identify.
	Region	AP-Singapore	For low network latency and quick resource access, select the region nearest to you.
	AZ	AZ2	-
	Subnet CIDR	10.0.0.0/24	Select a subnet with sufficient network resources.
	Subnet name	subnet-drs01	Specify a name that is easy to identify.
Source RDS instance	RDS instance name	rds-mysql-src	Specify a name that is easy to identify.

Category	Subcategory	Plan	Remarks
	Region	AP-Singapore	For low network latency and quick resource access, select the region nearest to you.
	DB engine version	MySQL 5.7	-
	Instance type	Single	A single instance is used in this example. To improve service reliability, select a primary/standby instance.
	Storage type	Cloud SSD	-
	AZ	AZ2	A single instance is used in this example. To improve service reliability, create a primary/standby RDS instance and then deploy them in different AZs.
	Instance class	General-purpose 4 vCPUs | 8 GB	-
Source VPN	Gateway name	vpngw-src	Specify a name that is easy to identify.
	Region	AP-Singapore	For low network latency and quick resource access, select the region nearest to you.
	VPC	vpc-DRSsrc	The VPC must be the same as the VPC of the source RDS instance.
	VPN connection name	vpn-src01	Specify a name that is easy to identify.
	Local subnet	subnet-drs01	The local subnet is the VPC subnet of the source RDS instance.
	Remote gateway	123.60.251.207	Peer VPN gateway. The value is the gateway address of the destination VPN. After the destination VPN is created, the gateway address is obtained.

Category	Subcategory	Plan	Remarks
	Remote subnet	172.16.0.0/24	Subnet of the peer VPN. The value is the subnet of the destination VPN, which is the subnet of the VPC where the destination DDM instance is deployed.
Destination VPC	VPC name	vpc-DRStar	Specify a name that is easy to identify.
	Region	CN-Hong Kong	For low network latency and quick resource access, select the region nearest to you.
	AZ	AZ1	-
	Subnet CIDR	172.16.0.0/24	Select a subnet with sufficient network resources.
	Subnet name	subnet-drs02	Specify a name that is easy to identify.
Destination DDM instance	DDM instance name	ddm-drs-tar	Specify a name that is easy to identify.
	AZ	AZ1	You can select one or more AZs. Ensure that the destination DDM instance is deployed across different AZs to improve service reliability.
	Node class	General-enhanced 8 vCPUs | 16 GB	-
	Nodes	1	A single node cannot provide high availability. Ensure that the destination DDM instance has at least two nodes.
RDS instance associated with the DDM instance	RDS instance name	rds-ddm01	Specify a name that is easy to identify.
	Region	CN-Hong Kong	For low network latency and quick resource access, select the region nearest to you.
	DB engine version	MySQL 5.7	-

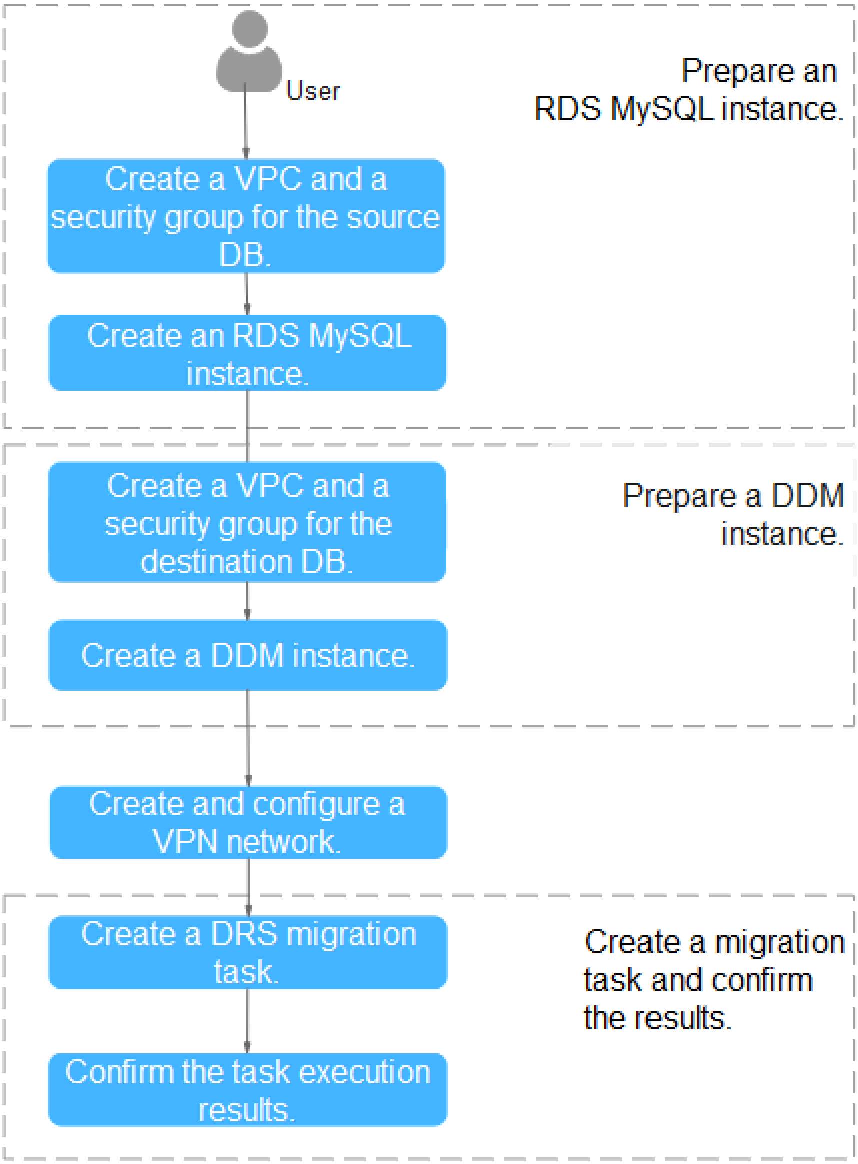
Category	Subcategory	Plan	Remarks
	Instance type	Single	A single instance is used in this example. To improve service reliability, select a primary/standby RDS instance.
	Storage type	Cloud SSD	-
	AZ	AZ1	A single instance is used in this example. To improve service reliability, create a primary/standby RDS instance and then deploy them in different AZs.
	Instance class	General-purpose 4 vCPUs | 8 GB	-
Destination VPN	Gateway name	vpngw-tar	Specify a name that is easy to identify.
	Region	CN-Hong Kong	For low network latency and quick resource access, select the region nearest to you.
	VPC	vpc-DRStar	The value is the VPC of the destination DDM instance.
	VPN connection name	vpn-tar01	Specify a name that is easy to identify.
	Local subnet	subnet-drs02	The value is the VPC subnet of the destination DDM instance.
	Remote gateway	123.60.236.84	Peer VPN gateway. In this example, this parameter is set to the gateway address of the source VPN. After the source VPN is created, the gateway address is obtained.
	Remote subnet	10.0.0.0/24	Subnet of the peer VPN. The value is the subnet of the source VPN, which is the subnet of the VPC where the source RDS for MySQL instance is deployed.
DRS migration task	Task name	DRS-MySQLToDDM	Specify a name that is easy to identify.

Category	Subcategory	Plan	Remarks
	Source DB engine	MySQL	In this example, the source is an RDS for MySQL instance on Huawei Cloud.
	Destination DB engine	DDM	In this example, the destination is a DDM instance.
	Network type	VPN network	In this example, VPNs are used.

11.3 Operation Process

Figure 11-2 shows the process of creating an RDS for MySQL instance and synchronizing data from the RDS instance to DDM.

Figure 11-2 Flowchart


User
Prepare an RDS MySQL instance.
Create a VPC and a security group for the source DB.
Create an RDS MySQL instance.
Create a VPC and a security group for the destination DB.
Prepare a DDM instance.
Create a DDM instance.
Create and configure a VPN network.
Create a DRS migration task.
Create a migration task and confirm the results.
Confirm the task execution results.

11.4 Preparing for the Source RDS for MySQL Instance

11.4.1 Creating a VPC and Security Group

Create a VPC and security group to prepare for creating an RDS for MySQL instance as the source.

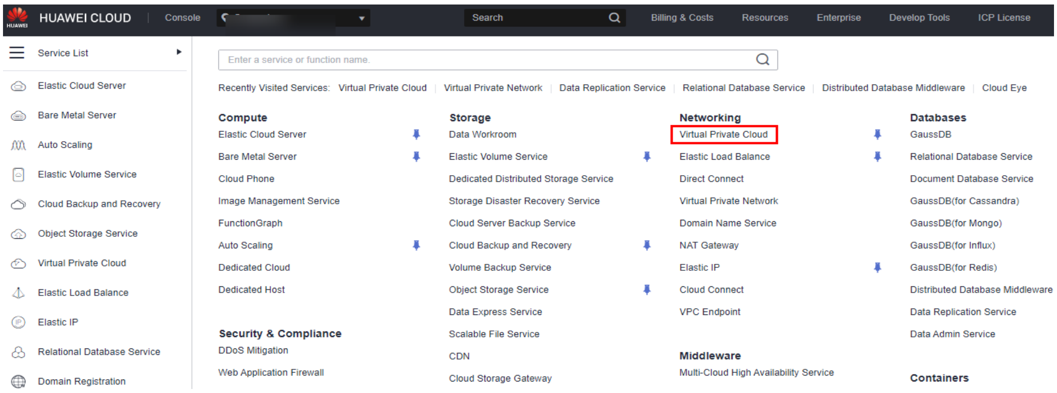
Creating a VPC

Step 1 Log in to the **management console**.

Step 2 Click 📍 in the upper left corner and select AP-Singapore.

Step 3 Click the service list icon on the left and choose **Networking** > **Virtual Private Cloud**.

The VPC console is displayed.



HUAWEI CLOUD | Console | Search | Billing & Costs | Resources | Enterprise | Develop Tools | ICP License

Service List

- Elastic Cloud Server
- Bare Metal Server
- Auto Scaling
- Elastic Volume Service
- Cloud Backup and Recovery
- Object Storage Service
- Virtual Private Cloud
- Elastic Load Balance
- Elastic IP
- Relational Database Service
- Domain Registration

Enter a service or function name.

Recently Visited Services: Virtual Private Cloud | Virtual Private Network | Data Replication Service | Relational Database Service | Distributed Database Middleware | Cloud Eye

Compute
Elastic Cloud Server
Bare Metal Server
Cloud Phone
Image Management Service
FunctionGraph
Auto Scaling
Dedicated Cloud
Dedicated Host

Security & Compliance
DDoS Mitigation
Web Application Firewall

Storage
Data Workroom
Elastic Volume Service
Dedicated Distributed Storage Service
Storage Disaster Recovery Service
Cloud Server Backup Service
Cloud Backup and Recovery
Volume Backup Service
Object Storage Service
Data Express Service
Scalable File Service
CDN
Cloud Storage Gateway

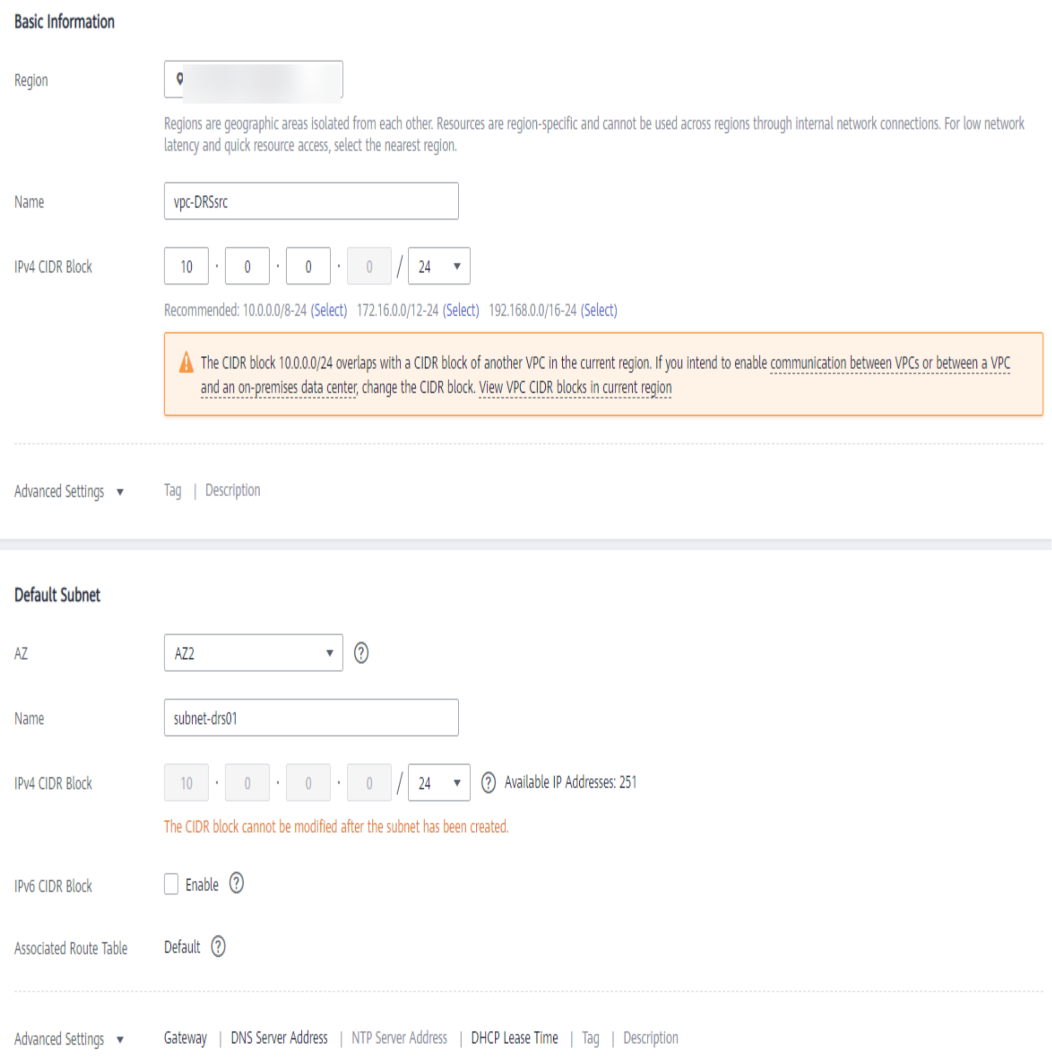
Networking
Virtual Private Cloud
Elastic Load Balance
Direct Connect
Virtual Private Network
Domain Name Service
NAT Gateway
Elastic IP
Cloud Connect
VPC Endpoint

Middleware
Multi-Cloud High Availability Service

Databases
GaussDB
Relational Database Service
Document Database Service
GaussDB(for Cassandra)
GaussDB(for Mongo)
GaussDB(for Influx)
GaussDB(for Redis)
Distributed Database Middleware
Data Replication Service
Data Admin Service

Containers

Step 4 Click **Create VPC**.



Basic Information

Region

Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region.

Name: vpc-DRSsrc

IPv4 CIDR Block: 10 . 0 . 0 . 0 / 24

Recommended: 10.0.0.0/8-24 (Select) 172.16.0.0/12-24 (Select) 192.168.0.0/16-24 (Select)

The CIDR block 10.0.0.0/24 overlaps with a CIDR block of another VPC in the current region. If you intend to enable communication between VPCs or between a VPC and an on-premises data center, change the CIDR block. View VPC CIDR blocks in current region

Advanced Settings ▾ Tag | Description

Default Subnet

AZ: AZ2

Name: subnet-drs01

IPv4 CIDR Block: 10 . 0 . 0 . 0 / 24 Available IP Addresses: 251

The CIDR block cannot be modified after the subnet has been created.

IPv6 CIDR Block: □ Enable

Associated Route Table: Default

Advanced Settings ▾ Gateway | DNS Server Address | NTP Server Address | DHCP Lease Time | Tag | Description

Step 5 Configure parameters as needed and click **Create Now**.

Step 6 Return to the VPC list and check whether the VPC is created.

When its status becomes available, the VPC is created.

----End

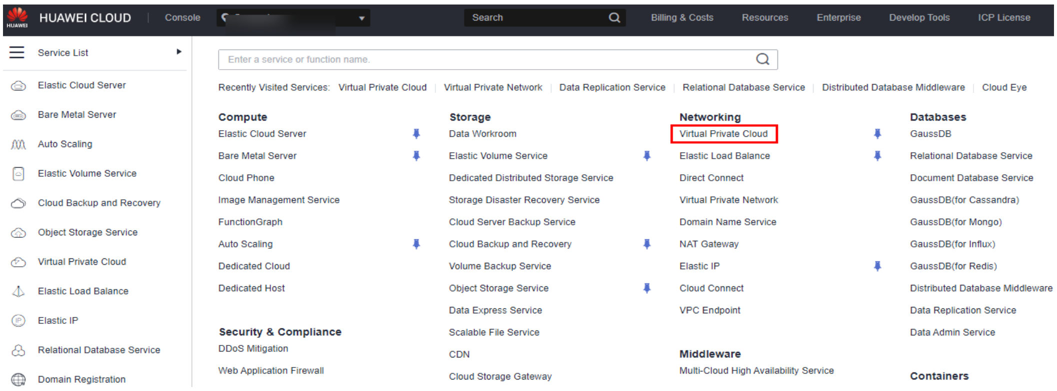
Creating a Security Group

Step 1 Log in to the **management console**.

Step 2 Click in the upper left corner and select AP-Singapore.

Step 3 Click the service list icon on the left and choose **Networking** > **Virtual Private Cloud**.

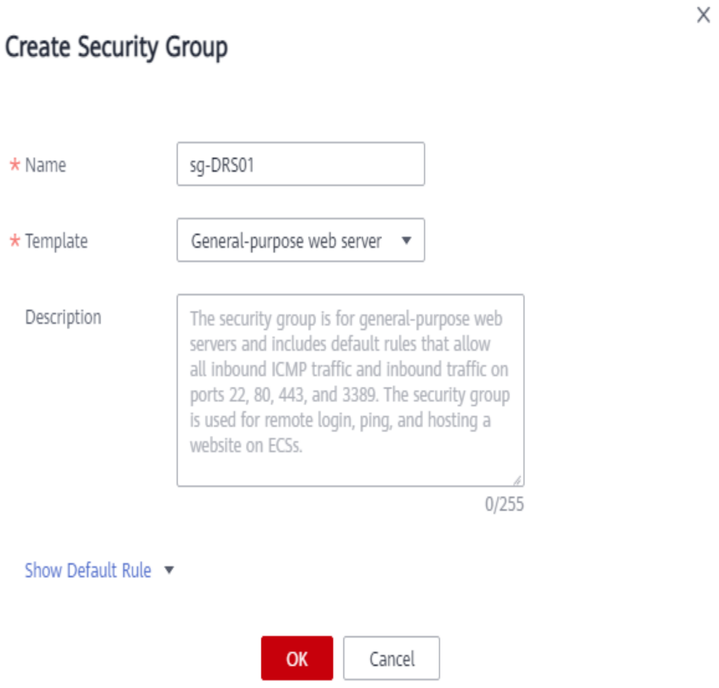
The VPC console is displayed.



Step 4 In the navigation pane, choose **Access Control** > **Security Groups**.

Step 5 Click **Create Security Group**.

Step 6 Specify a security group name and other information.



Create Security Group

* Name: sg-DRS01

* Template: General-purpose web server

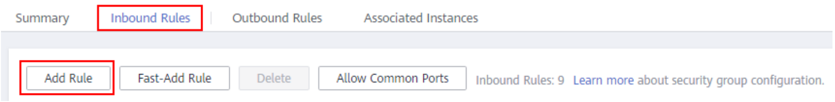
Description: The security group is for general-purpose web servers and includes default rules that allow all inbound ICMP traffic and inbound traffic on ports 22, 80, 443, and 3389. The security group is used for remote login, ping, and hosting a website on ECSs.

0/255

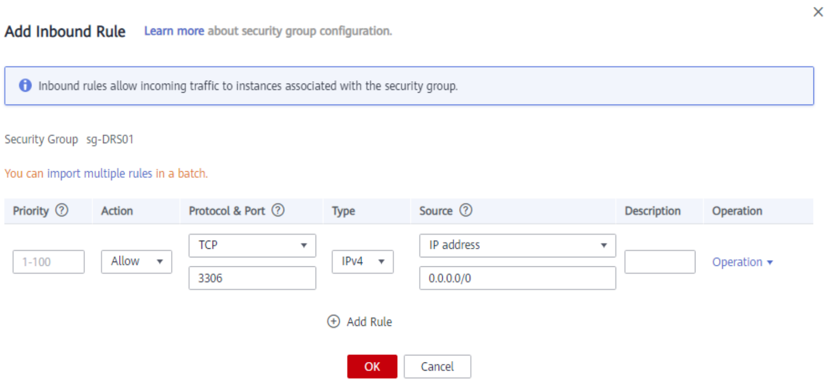
Step 7 Click **OK**.

Step 8 Return to the security group list and click the security group name (**sg-DRS01** in this example).

Step 9 Click the **Inbound Rules** tab, and then click **Add Rule**.



Step 10 Configure an inbound rule to allow access from database port **3306**.



Add Inbound Rule Learn more about security group configuration.

Inbound rules allow incoming traffic to instances associated with the security group.

Security Group sg-DRS01

You can import multiple rules in a batch.

Priority	Action	Protocol & Port	Type	Source	Description	Operation
1-100	Allow	TCP 3306	IPv4	IP address 0.0.0.0/0		Operation

----End

11.4.2 Creating an RDS for MySQL Instance

Create an RDS for MySQL instance and test data.

Creating an RDS for MySQL Instance

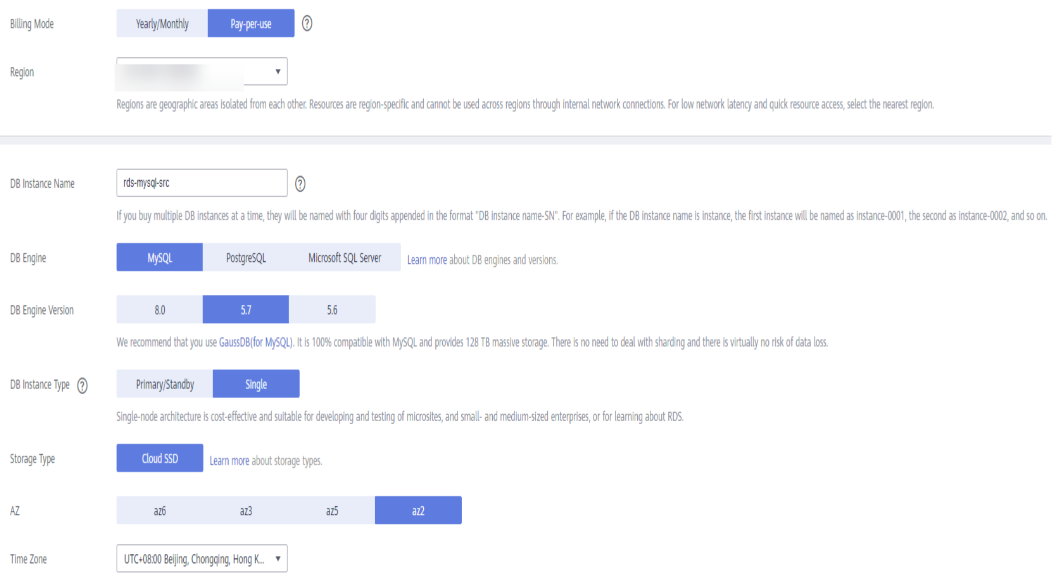
Step 1 Log in to the **management console**.

Step 2 Click in the upper left corner and select AP-Singapore.

Step 3 Click the service list icon on the left and choose **Databases** > **Relational Database Service**.

Step 4 Click **Buy DB Instance**.

Step 5 Configure the instance name and basic information.



Billing Mode: Yearly/Monthly [✓ Pay-per-use] ?

Region: [redacted]

Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region.

DB Instance Name: rds-mysql-src ?

If you buy multiple DB instances at a time, they will be named with four digits appended in the format "DB instance name-SN". For example, if the DB instance name is instance, the first instance will be named as instance-0001, the second as instance-0002, and so on.

DB Engine: [✓ MySQL] PostgreSQL Microsoft SQL Server — Learn more about DB engines and versions.

DB Engine Version: 8.0 [✓ 5.7] 5.6

We recommend that you use GaussDB(for MySQL). It is 100% compatible with MySQL and provides 128 TB massive storage. There is no need to deal with sharding and there is virtually no risk of data loss.

DB Instance Type ?: Primary/Standby [✓ Single]

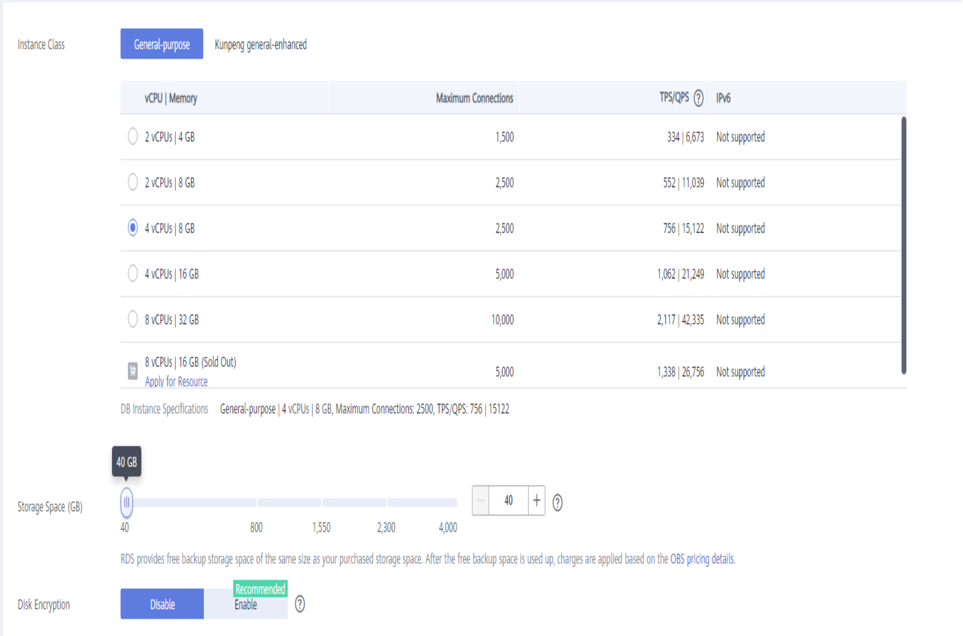
Single-node architecture is cost-effective and suitable for developing and testing of microsites, and small- and medium-sized enterprises, or for learning about RDS.

Storage Type: [✓ Cloud SSD] Learn more about storage types.

AZ: az6 az3 az5 [✓ az2]

Time Zone: UTC+08:00 Beijing, Chongqing, Hong K...

Step 6 Select an instance class.



Instance Class: [✓ General-purpose] Kunpeng general-enhanced

	vCPU | Memory	Maximum Connections	TPS/QPS ?	IPv6
○	2 vCPUs | 4 GB	1,500	334 | 6,673	Not supported
○	2 vCPUs | 8 GB	2,500	552 | 11,039	Not supported
●	4 vCPUs | 8 GB	2,500	756 | 15,122	Not supported
○	4 vCPUs | 16 GB	5,000	1,062 | 21,249	Not supported
○	8 vCPUs | 32 GB	10,000	2,117 | 42,335	Not supported
	8 vCPUs | 16 GB (Sold Out) Apply for Resource	5,000	1,338 | 26,756	Not supported

DB Instance Specifications: General-purpose | 4 vCPUs | 8 GB, Maximum Connections: 2500, TPS/QPS: 756 | 15122

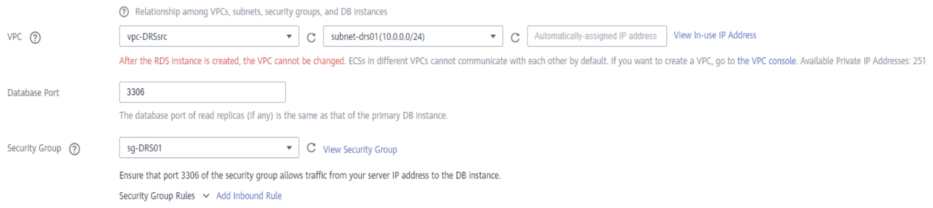
Storage Space (GB): 40 GB (slider 40, 800, 1,550, 2,300, 4,000) — 40 ?

RDS provides free backup storage space of the same size as your purchased storage space. After the free backup space is used up, charges are applied based on the OBS pricing details.

Disk Encryption: [✓ Disable] Enable (Recommended) ?

Step 7 Select a VPC and security group and configure the database port.

The VPC and security group have been created in **Creating a VPC and Security Group**.



? Relationship among VPCs, subnets, security groups, and DB instances

VPC ?: vpc-DRSsrc — subnet-drs01(10.0.0.0/24) — Automatically-assigned IP address — View In-use IP Address

After the RDS instance is created, the VPC cannot be changed. ECSs in different VPCs cannot communicate with each other by default. If you want to create a VPC, go to the VPC console. Available Private IP Addresses: 251

Database Port: 3306

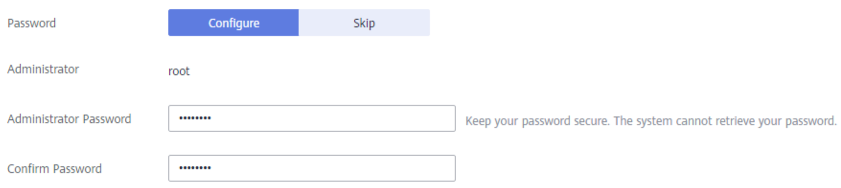
The database port of read replicas (if any) is the same as that of the primary DB instance.

Security Group ?: sg-DRS01 — View Security Group

Ensure that port 3306 of the security group allows traffic from your server IP address to the DB instance.

Security Group Rules — Add Inbound Rule

Step 8 Set the instance password.



Password: Configure | Skip

Administrator: root

Administrator Password: •••••••• Keep your password secure. The system cannot retrieve your password.

Confirm Password: ••••••••

Step 9 Click **Next** and complete the whole process.

Step 10 Return to the instance list.

If the instance status becomes **Available**, the instance is created.

----End

Creating Test Data

Step 1 Log in to the **management console**.

Step 2 Click 📍 in the upper left corner and select AP-Singapore.

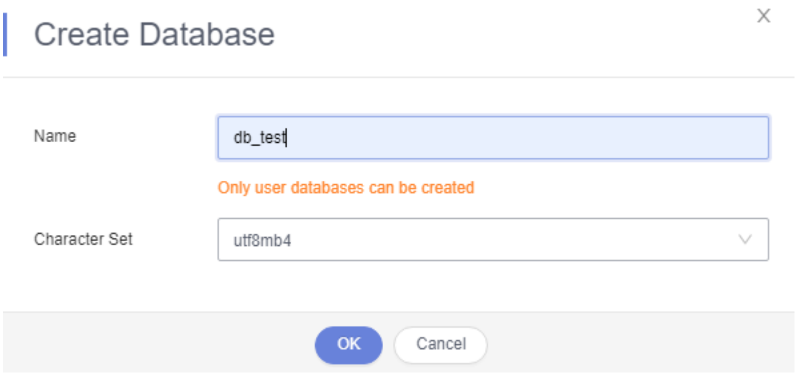
Step 3 Click the service list icon on the left and choose **Databases** > **Relational Database Service**.

Step 4 Locate the created RDS instance and choose **More** > **Log In**.

Step 5 In the displayed dialog box, enter the username and password of the instance and click **Test Connection**.

Step 6 After the connection is successful, click **Log In**.

Step 7 Click **Create Database** to create the **db_test** database.



Create Database

Name: db_test

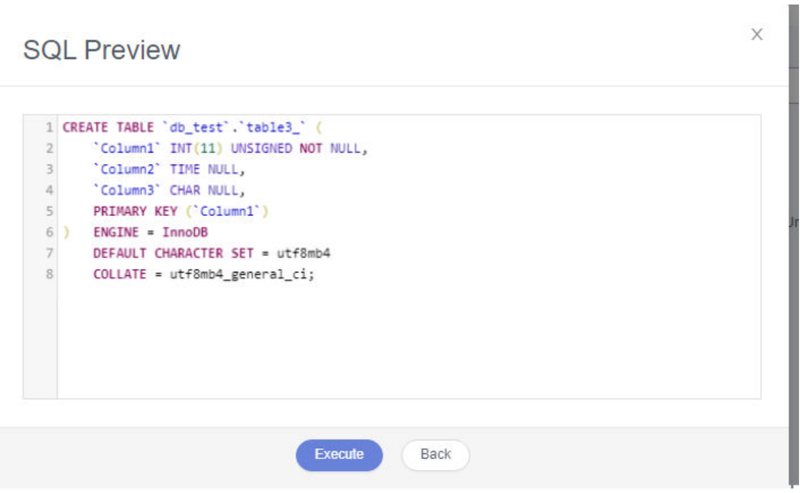
Only user databases can be created

Character Set: utf8mb4

OK | Cancel

Step 8 Run the following SQL statement in **db_test** to create table **table3_**:

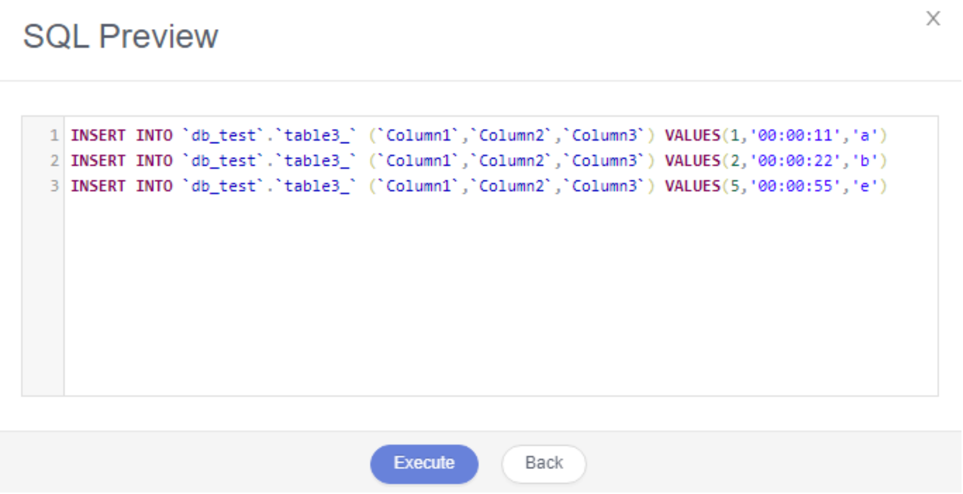
```
CREATE TABLE `db_test`.`table3_` (
    `Column1` INT(11) UNSIGNED NOT NULL,
    `Column2` TIME NULL,
    `Column3` CHAR NULL,
    PRIMARY KEY (`Column1`)
)    ENGINE = InnoDB
    DEFAULT CHARACTER SET = utf8mb4
    COLLATE = utf8mb4_general_ci;
```



```
CREATE TABLE `db_test`.`table3_` (
    `Column1` INT(11) UNSIGNED NOT NULL,
    `Column2` TIME NULL,
    `Column3` CHAR NULL,
    PRIMARY KEY (`Column1`)
)   ENGINE = InnoDB
    DEFAULT CHARACTER SET = utf8mb4
    COLLATE = utf8mb4_general_ci;
```

Step 9 Run the following statements in table **table3_** to insert three lines of data:

```
INSERT INTO `db_test`.`table3_` (`Column1`,`Column2`,`Column3`) VALUES(1,'00:00:11','a');
INSERT INTO `db_test`.`table3_` (`Column1`,`Column2`,`Column3`) VALUES(2,'00:00:22','b');
INSERT INTO `db_test`.`table3_` (`Column1`,`Column2`,`Column3`) VALUES(5,'00:00:55','e');
```



```
INSERT INTO `db_test`.`table3_` (`Column1`,`Column2`,`Column3`) VALUES(1,'00:00:11','a')
INSERT INTO `db_test`.`table3_` (`Column1`,`Column2`,`Column3`) VALUES(2,'00:00:22','b')
INSERT INTO `db_test`.`table3_` (`Column1`,`Column2`,`Column3`) VALUES(5,'00:00:55','e')
```

----End

11.4.3 Creating a VPN for the Source

Step 1 Log in to the **management console**.

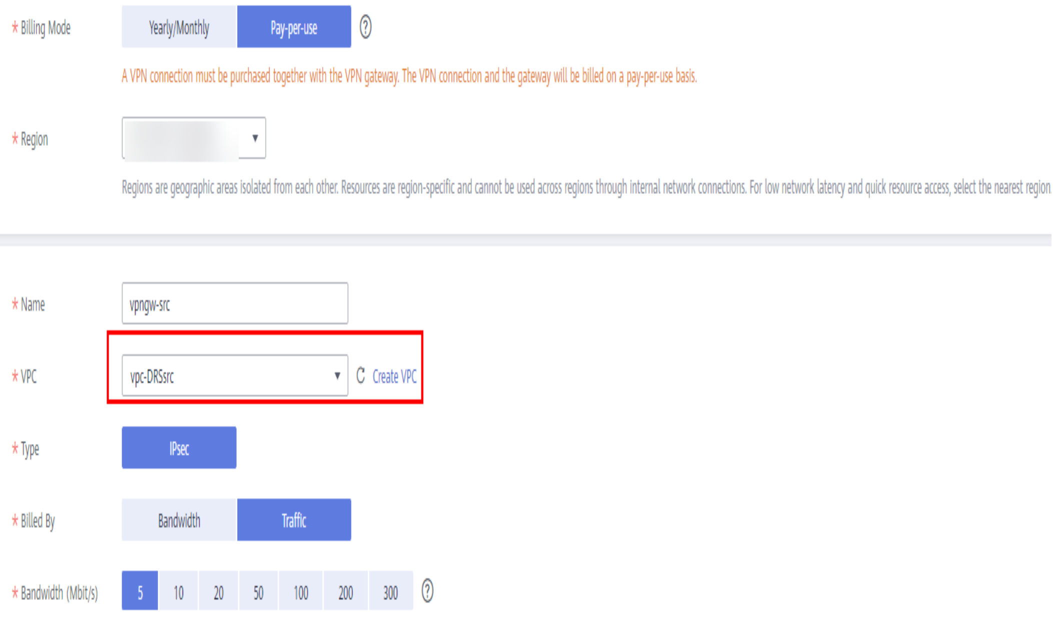
Step 2 Click 📍 in the upper left corner and select AP-Singapore.

Step 3 Click the service list icon on the left and choose **Networking** > **Virtual Private Network**.

Step 4 In the navigation pane on the left, choose **Virtual Private Network** > **VPN Gateways**.

Step 5 On the **VPN Gateways** page, click **Buy VPN Gateway**.

Step 6 Specify a gateway name and other information.



* Billing Mode: Yearly/Monthly | **Pay-per-use**

A VPN connection must be purchased together with the VPN gateway. The VPN connection and the gateway will be billed on a pay-per-use basis.

* Region

Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region

* Name: vpngw-src

* VPC: vpc-DRSsrc

* Type: **IPsec**

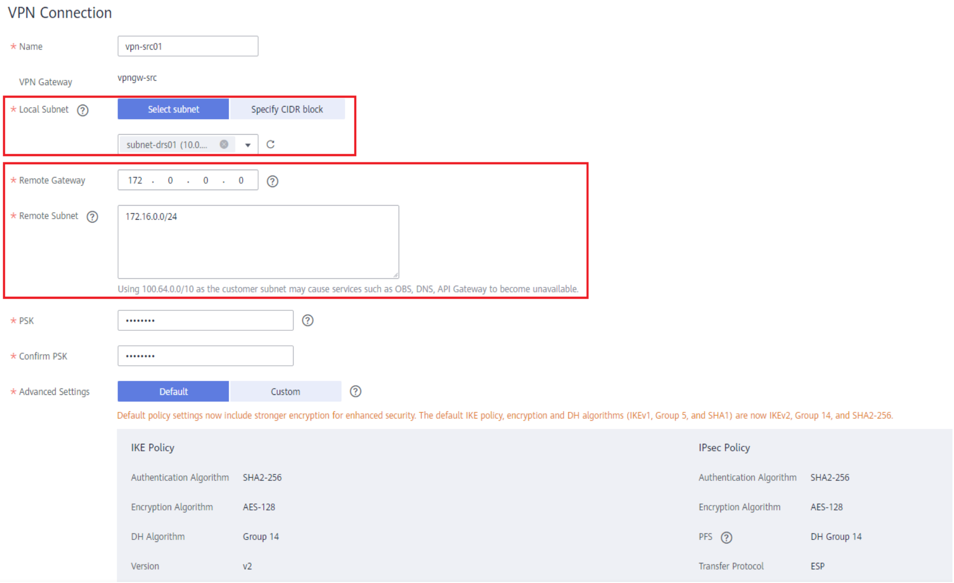
* Billed By: Bandwidth | **Traffic**

* Bandwidth (Mbit/s): **5** | 10 | 20 | 50 | 100 | 200 | 300

CAUTION

The connected VPN gateway must have the same VPC as the source RDS for MySQL instance, that is, the VPC created in **Creating a VPC and Security Group**.

Step 7 Specify the VPN connection information and click **Submit**.



VPN Connection

* Name: vpn-src01

VPN Gateway: vpngw-src

* Local Subnet: **Select subnet** | Specify CIDR block

subnet-drs01 (10.0....

* Remote Gateway: 172 . 0 . 0 . 0

* Remote Subnet: 172.16.0.0/24

Using 100.64.0.0/10 as the customer subnet may cause services such as OBS, DNS, API Gateway to become unavailable.

* PSK: ••••••••

* Confirm PSK: ••••••••

* Advanced Settings: **Default** | Custom

Default policy settings now include stronger encryption for enhanced security. The default IKE policy, encryption and DH algorithms (IKEv1, Group 5, and SHA1) are now IKEv2, Group 14, and SHA2-256.

IKE Policy

Authentication Algorithm	SHA2-256
Encryption Algorithm	AES-128
DH Algorithm	Group 14
Version	v2

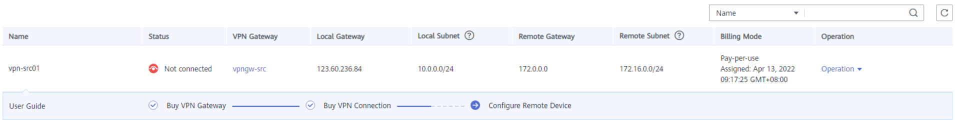
IPsec Policy

Authentication Algorithm	SHA2-256
Encryption Algorithm	AES-128
PFS	DH Group 14
Transfer Protocol	ESP

CAUTION

- The local subnet must be the same as the VPC subnet where the source RDS for MySQL instance is located, that is, the subnet created in **Creating a VPC and Security Group**.
- The remote gateway and subnet are the gateway and subnet of the destination VPN. If the destination VPN is not created, enter a random value and change it after the destination VPN is created.

Step 8 After the VPN gateway is created, view its information in the VPN gateway list. Its status is **Not connected**. If this VPN gateway is in use by a VPC connection, the VPN gateway status changes to **Normal**.



Name	Status	VPN Gateway	Local Gateway	Local Subnet	Remote Gateway	Remote Subnet	Billing Mode	Operation
vpn-src01	Not connected	vpngw-src	123.60.236.84	10.0.0.0/24	172.0.0.0	172.16.0.0/24	Pay-per-use Assigned: Apr 13, 2022 09:17:25 GMT+08:00	Operation

----End

11.5 Preparing for the Destination DDM Instance

11.5.1 Creating a VPC and Security Group

Create a VPC and security group to prepare for the destination DDM instance.

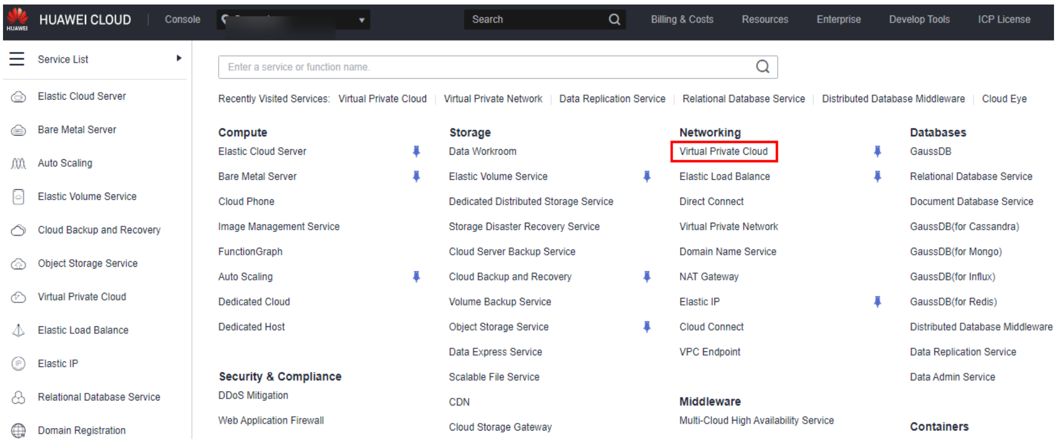
Creating a VPC

Step 1 Log in to the **management console**.

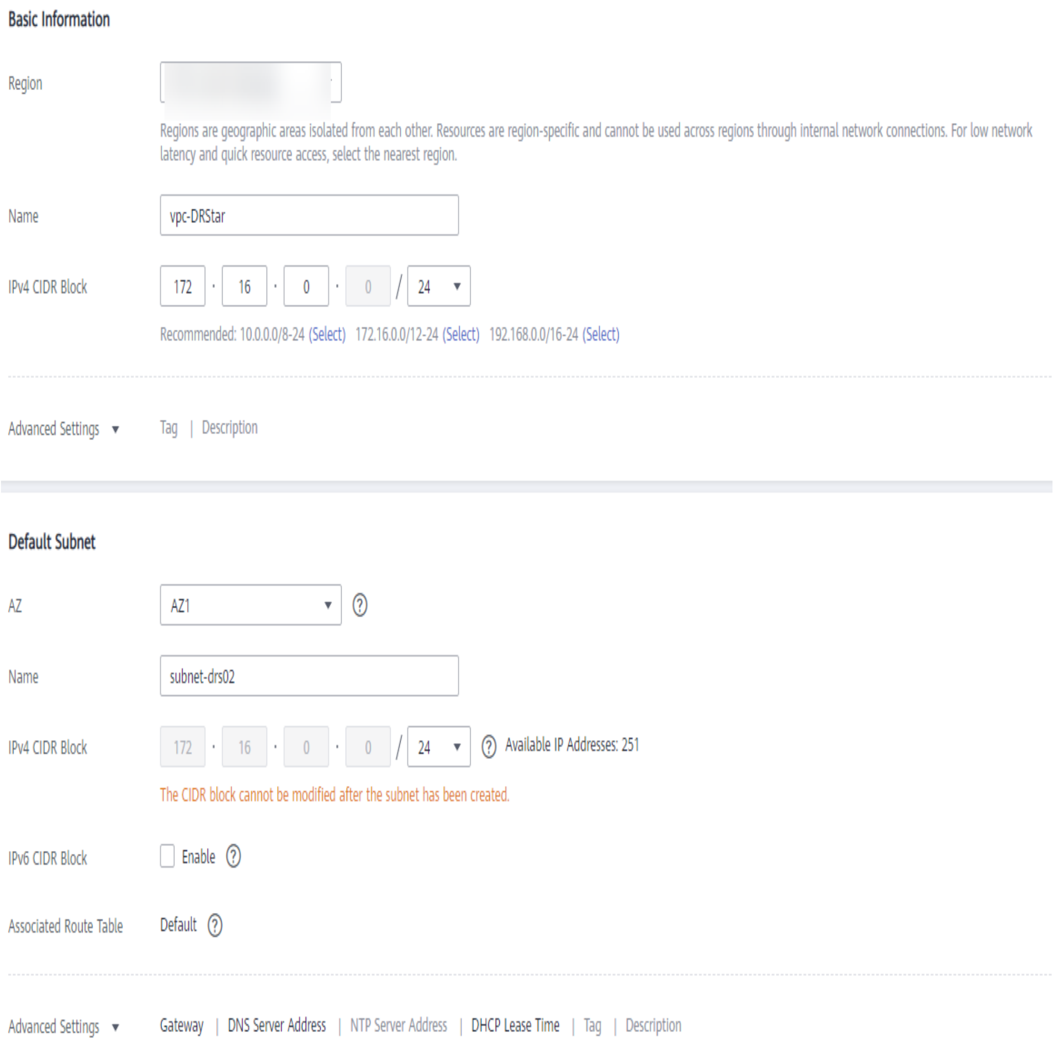
Step 2 Click in the upper left corner and select region CN-Hong Kong.

Step 3 Click the service list icon on the left and choose **Networking** > **Virtual Private Cloud**.

The VPC console is displayed.



Step 4 Click **Create VPC**.



Basic Information

Region: [redacted]

Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region.

Name: vpc-DRStar

IPv4 CIDR Block: 172 . 16 . 0 . 0 / 24

Recommended: 10.0.0.0/8-24 (Select) 172.16.0.0/12-24 (Select) 192.168.0.0/16-24 (Select)

Advanced Settings ▼ Tag | Description

Default Subnet

AZ: AZ1

Name: subnet-drs02

IPv4 CIDR Block: 172 . 16 . 0 . 0 / 24 Available IP Addresses: 251

The CIDR block cannot be modified after the subnet has been created.

IPv6 CIDR Block: □ Enable

Associated Route Table: Default

Advanced Settings ▼ Gateway | DNS Server Address | NTP Server Address | DHCP Lease Time | Tag | Description

Step 5 Configure parameters as needed and click **Create Now**.

Step 6 Return to the VPC list and check whether the VPC is created.

When its status becomes available, the VPC is created.

----End

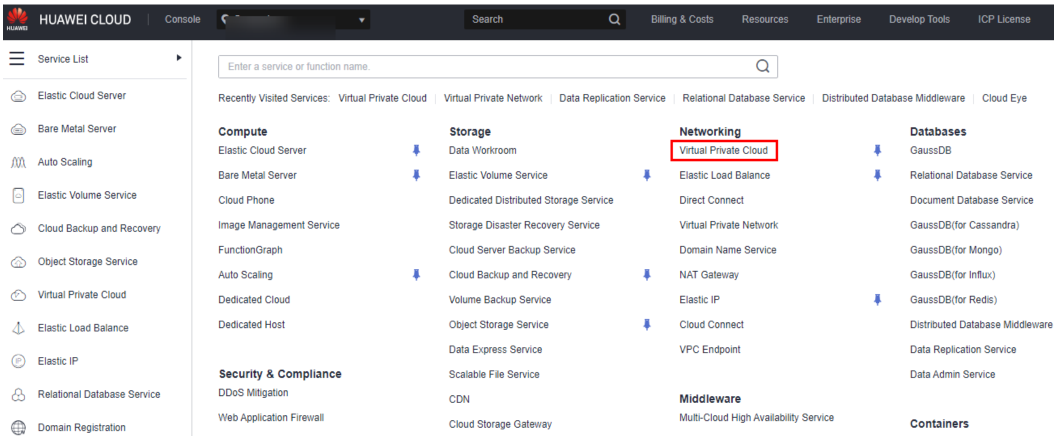
Creating a Security Group

Step 1 Log in to the **management console**.

Step 2 Click 📍 in the upper left corner and select region CN-Hong Kong.

Step 3 Click the service list icon on the left and choose **Networking** > **Virtual Private Cloud**.

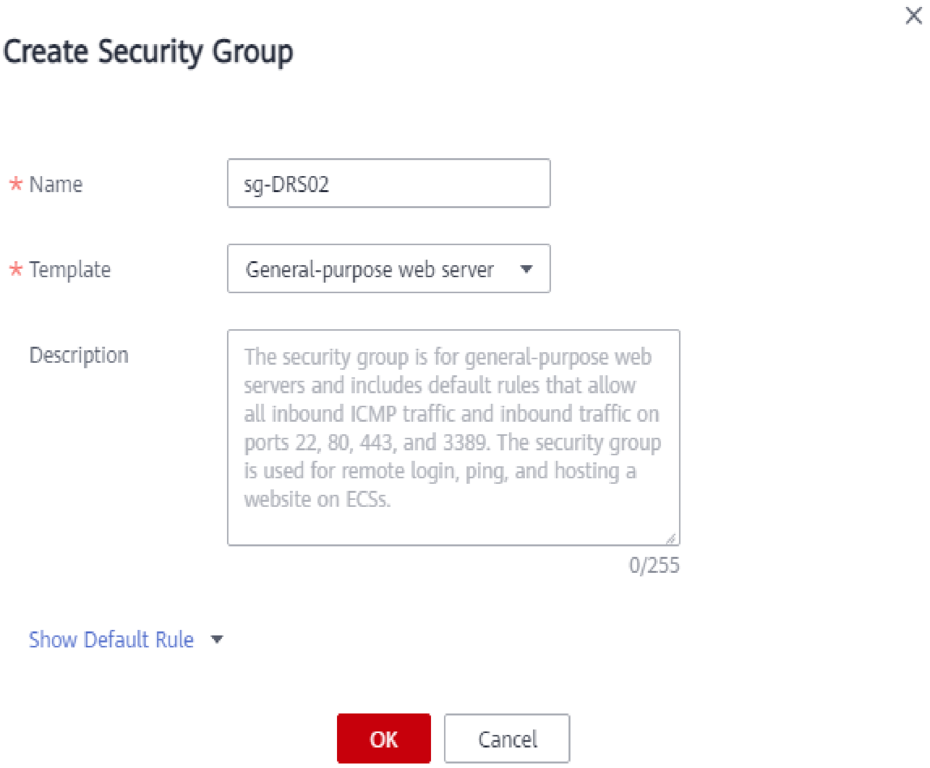
The VPC console is displayed.



Step 4 In the navigation pane, choose **Access Control** > **Security Groups**.

Step 5 Click **Create Security Group**.

Step 6 Specify the security group name and other information, and click **OK**.



Create Security Group

* Name: sg-DRS02

* Template: General-purpose web server

Description: The security group is for general-purpose web servers and includes default rules that allow all inbound ICMP traffic and inbound traffic on ports 22, 80, 443, and 3389. The security group is used for remote login, ping, and hosting a website on ECSs.

0/255

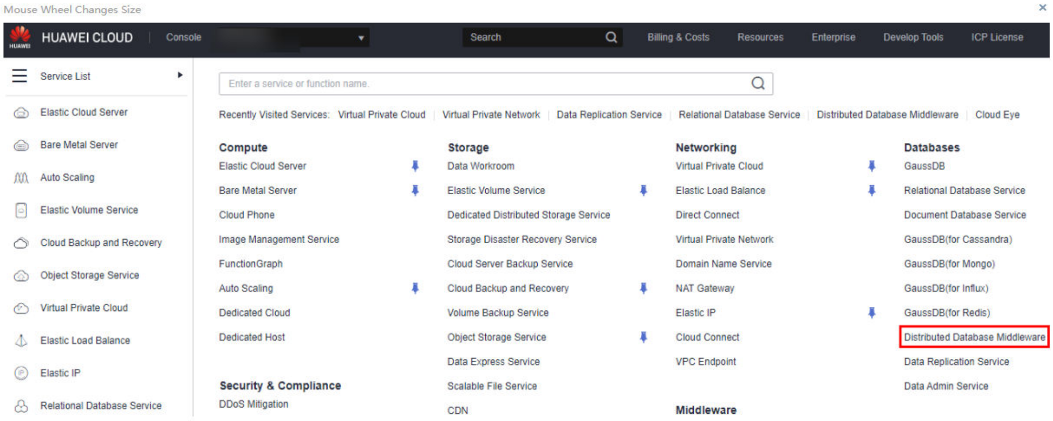
----End

11.5.2 Creating a DDM Instance

Step 1 Log in to the **management console**.

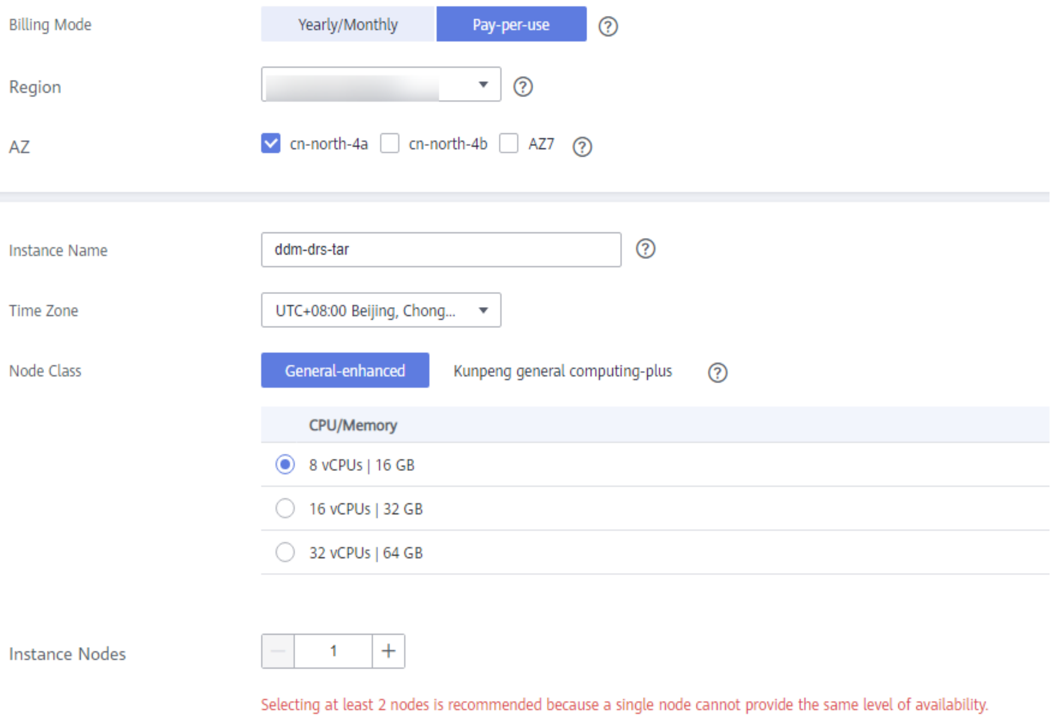
Step 2 Click in the upper left corner and select region CN-Hong Kong.

Step 3 Click the service list icon on the left and choose **Databases** > **Distributed Database Middleware**.



Step 4 On the displayed page, in the upper right corner, click **Buy DDM Instance**.

Step 5 Specify a node class and other information.



Billing Mode: Yearly/Monthly **Pay-per-use** ?

Region: ▾ ?

AZ: ☑ cn-north-4a ☐ cn-north-4b ☐ AZ7 ?

Instance Name: ddm-drs-tar ?

Time Zone: UTC+08:00 Beijing, Chong... ▾

Node Class: **General-enhanced** Kunpeng general computing-plus ?

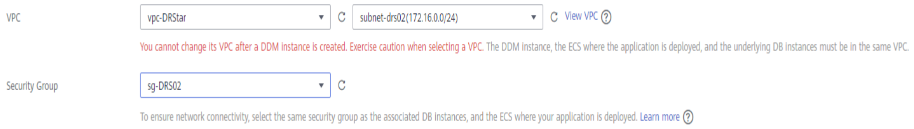
CPU/Memory
● 8 vCPUs | 16 GB
○ 16 vCPUs | 32 GB
○ 32 vCPUs | 64 GB

Instance Nodes: − 1 +

Selecting at least 2 nodes is recommended because a single node cannot provide the same level of availability.

Step 6 Select a VPC and security group and configure the database port.

The VPC and security group have been created in **Creating a VPC and Security Group**.



VPC: vpc-DRStar ▾ C subnet-drs02(172.16.0.0/24) ▾ C View VPC ?

You cannot change its VPC after a DDM instance is created. Exercise caution when selecting a VPC. The DDM instance, the ECS where the application is deployed, and the underlying DB instances must be in the same VPC.

Security Group: sg-DRS02 ▾ C

To ensure network connectivity, select the same security group as the associated DB instances, and the ECS where your application is deployed. Learn more ?

Step 7 After the configuration is complete, click **Next** at the bottom of the page.

Step 8 Go to the **Instances** page to view and manage the instance.

The default database port is **5066** and cab be changed after the instance is created. If the status of the instance is **Running**, the instance has been created.

----End

11.5.3 Creating an RDS for MySQL Instance

Create an RDS for MySQL instance and associate it with the DDM instance.

Procedure

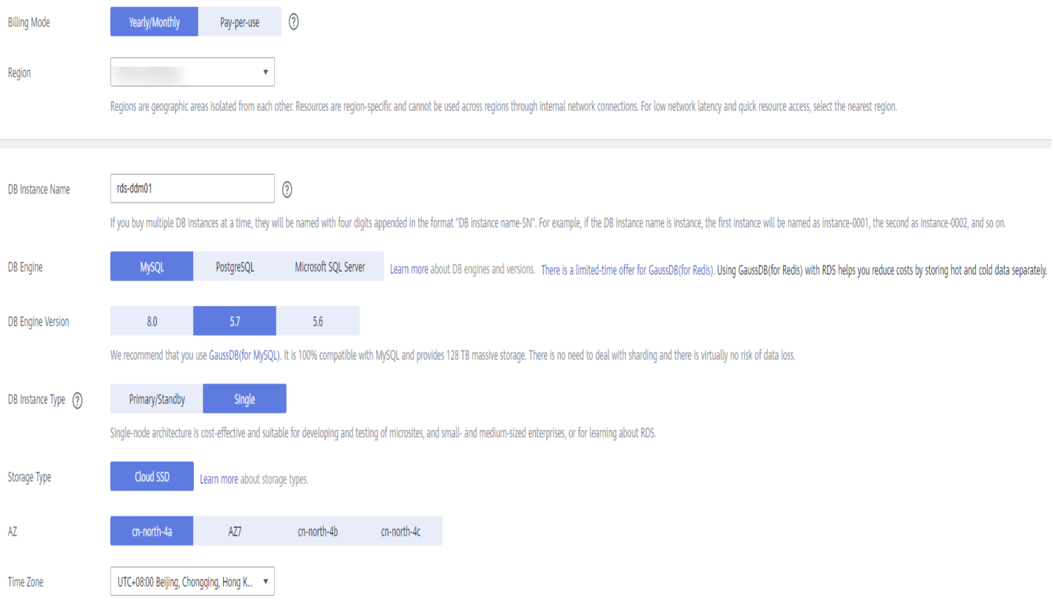
Step 1 Log in to the **management console**.

Step 2 Click 📍 in the upper left corner and select region CN-Hong Kong.

Step 3 Click the service list icon on the left and choose **Databases** > **Relational Database Service**.

Step 4 Click **Buy DB Instance**.

Step 5 Configure the instance name and basic information.



Billing Mode: Yearly/Monthly

Region:

Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region.

DB Instance Name: rds-ddm01

If you buy multiple DB instances at a time, they will be named with four digits appended in the format "DB instance name-SN". For example, if the DB instance name is instance, the first instance will be named as instance-0001, the second as instance-0002, and so on.

DB Engine: MySQL

Learn more about DB engines and versions. There is a limited-time offer for GaussDB(for Redis). Using GaussDB(for Redis) with RDS helps you reduce costs by storing hot and cold data separately.

DB Engine Version: 5.7

We recommend that you use GaussDB(for MySQL). It is 100% compatible with MySQL and provides 128 TB massive storage. There is no need to deal with sharding and there is virtually no risk of data loss.

DB Instance Type: Single

Single-node architecture is cost-effective and suitable for developing and testing of microsites, and small- and medium-sized enterprises, or for learning about RDS.

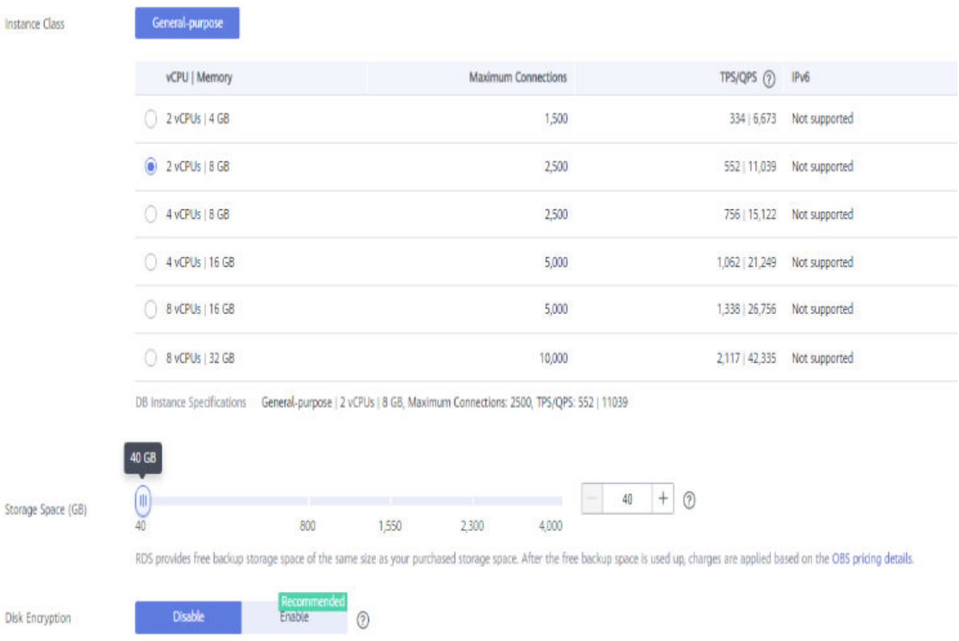
Storage Type: Cloud SSD

Learn more about storage types.

AZ: cn-north-4a

Time Zone: UTC+08:00 Beijing, Chongqing, Hong K...

Step 6 Select an instance class.



Instance Class: General-purpose

	vCPU | Memory	Maximum Connections	TPS/QPS	IPv6
○	2 vCPUs | 4 GB	1,500	334 | 6,673	Not supported
●	2 vCPUs | 8 GB	2,500	552 | 11,039	Not supported
○	4 vCPUs | 8 GB	2,500	756 | 15,122	Not supported
○	4 vCPUs | 16 GB	5,000	1,062 | 21,249	Not supported
○	8 vCPUs | 16 GB	5,000	1,338 | 26,756	Not supported
○	8 vCPUs | 32 GB	10,000	2,117 | 42,335	Not supported

DB Instance Specifications General-purpose | 2 vCPUs | 8 GB, Maximum Connections: 2500, TPS/QPS: 552 | 11039

Storage Space (GB): 40 GB

RDS provides free backup storage space of the same size as your purchased storage space. After the free backup space is used up, charges are applied based on the OBS pricing details.

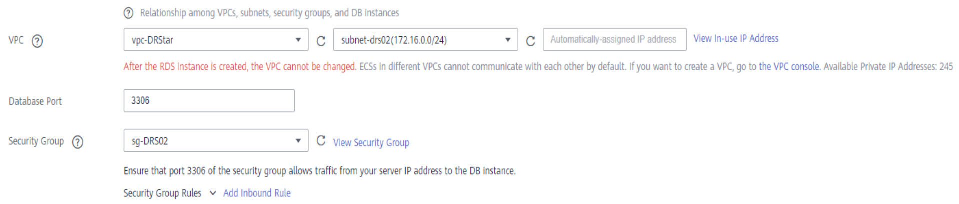
Disk Encryption: Disable

Step 7 Select a VPC and security group and configure the database port.

The VPC and security group have been created in **Creating a VPC and Security Group**.

⚠ CAUTION

The RDS for MySQL instance must be in the same VPC and subnet as the DDM instance.



Relationship among VPCs, subnets, security groups, and DB instances

VPC: vpc-DRStar | subnet-drs02(172.16.0.0/24) | Automatically-assigned IP address

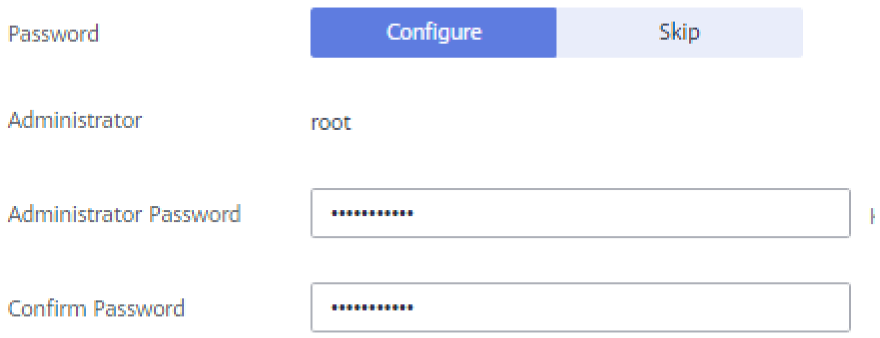
After the RDS instance is created, the VPC cannot be changed. ECSs in different VPCs cannot communicate with each other by default. If you want to create a VPC, go to the VPC console. Available Private IP Addresses: 245

Database Port: 3306

Security Group: sg-DRS02

Ensure that port 3306 of the security group allows traffic from your server IP address to the DB instance.

Step 8 Set the instance password.



Password: Configure

Administrator: root

Administrator Password: ••••••••••••

Confirm Password: ••••••••••••

Step 9 Click **Next** and complete the whole process.

Step 10 Return to the instance list.

If the instance status is **Available**, the instance is created.

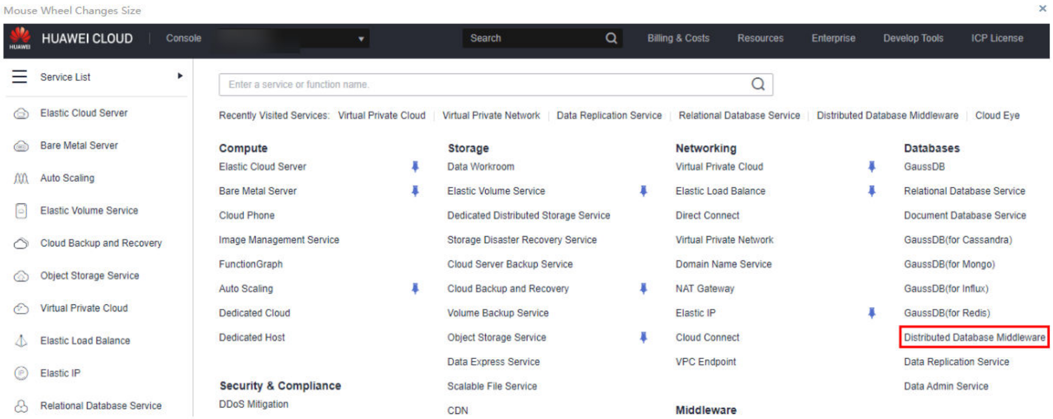
----End

11.5.4 Creating a Schema and Associating It with the RDS for MySQL Instance

Step 1 Log in to the **management console**.

Step 2 Click in the upper left corner and select region CN-Hong Kong.

Step 3 Click the service list icon on the left and choose **Databases** > **Distributed Database Middleware**.



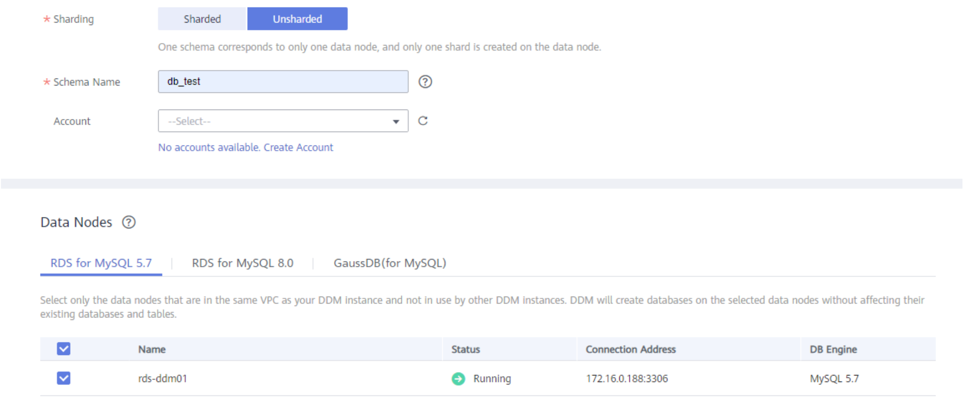
Step 4 On the **Instances** page, locate the required DDM instance and click **Create Schema** in the **Operation** column.

Step 5 On the displayed page, specify a sharding method, enter a schema name, set the number of shards, select the required DDM accounts, and click **Next**.

In this example, the schema is unsharded, and the schema name is **db_test**.

⚠ CAUTION

Currently, only data can be migrated from the RDS for MySQL to DDM. To migrate table structures and other objects, you need to create schemas in the destination DDM instance based on table structures of the source RDS for MySQL instance.



* Sharding: Sharded | Unsharded

One schema corresponds to only one data node, and only one shard is created on the data node.

* Schema Name: db_test

Account: --Select--

No accounts available. Create Account

Data Nodes

RDS for MySQL 5.7 | RDS for MySQL 8.0 | GaussDB(for MySQL)

Select only the data nodes that are in the same VPC as your DDM instance and not in use by other DDM instances. DDM will create databases on the selected data nodes without affecting their existing databases and tables.

☑	Name	Status	Connection Address	DB Engine
☑	rds-ddm01	Running	172.16.0.188:3306	MySQL 5.7

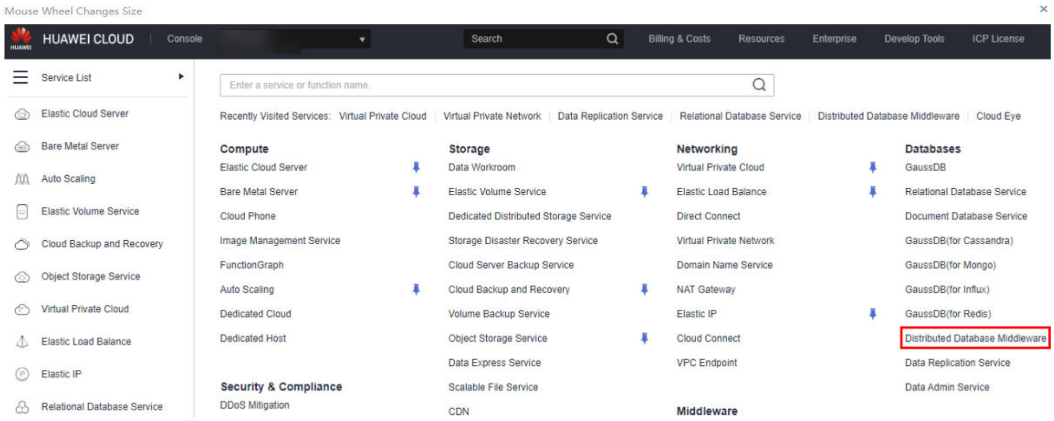
----End

11.5.5 Creating a DDM Account

Step 1 Log in to the **management console**.

Step 2 Click 📍 in the upper left corner and select region CN-Hong Kong.

Step 3 Click the service list icon on the left and choose **Databases** > **Distributed Database Middleware**.

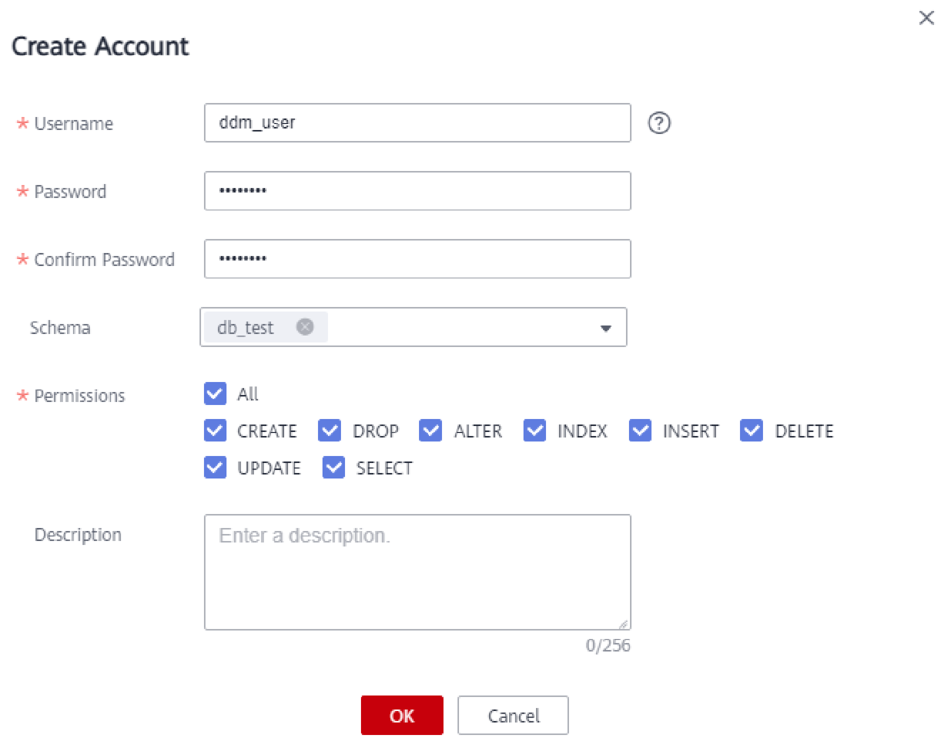


Step 4 On the **Instances** page, locate the required DDM instance and click its name.

Step 5 In the navigation pane, choose **Accounts**.

Step 6 On the displayed page, click **Create Account** and configure parameters as needed.

For details about the permissions required by the DDM account, see **Precautions**



Create Account

* Username: ddm_user

* Password: ••••••••

* Confirm Password: ••••••••

Schema: db_test

* Permissions: ☑ All
☑ CREATE ☑ DROP ☑ ALTER ☑ INDEX ☑ INSERT ☑ DELETE
☑ UPDATE ☑ SELECT

Description: Enter a description.
0/256

Step 7 Click **OK**.

----End

11.5.6 Creating Table Structures in the Destination Database

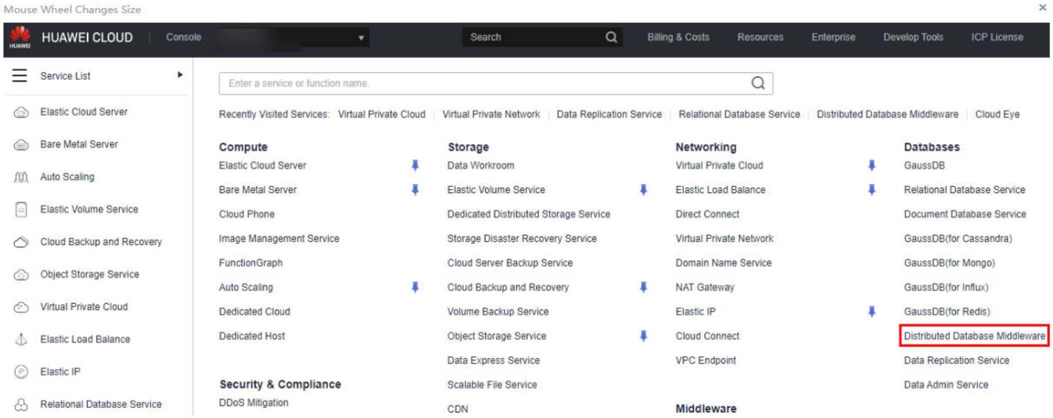
Currently, only data can be migrated from RDS for MySQL instances to DDM instances. To migrate table structures and other objects, you need to create table structures and indexes in the destination database based on table structures of the source schema. Any source objects that have no corresponding objects created in the destination cannot be migrated. For more constraints, see **Before You Start**.

Procedure

Step 1 Log in to the **management console**.

Step 2 Click 📍 in the upper left corner and select region CN-Hong Kong.

Step 3 Click the service list icon on the left and choose **Databases** > **Distributed Database Middleware**.



Step 4 On the **Instances** page, locate the required instance and click **Log In** in the **Operation** column.

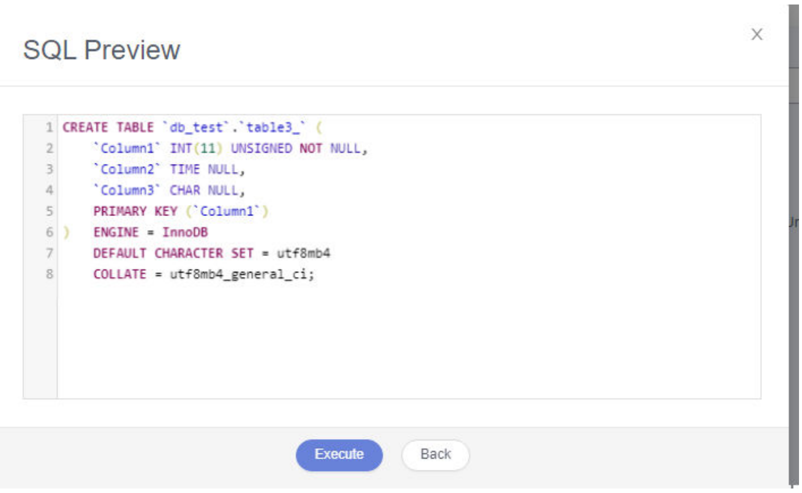
Step 5 In the displayed dialog box, enter the username and password created in Creating a DDM Account, and click **Test Connection**.

Step 6 After the connection is successful, click **Log In** to log in to the DDM instance.

Step 7 Click the **db_test** schema created in Creating a Schema and Associating It with the RDS for MySQL Instance.

Step 8 Run the following SQL statements in database **db_test** to create table **table3_** with the same structure as the source:

```
CREATE TABLE `db_test`.`table3_` (
    `Column1` INT(11) UNSIGNED NOT NULL,
    `Column2` TIME NULL,
    `Column3` CHAR NULL,
    PRIMARY KEY (`Column1`)
)   ENGINE = InnoDB
    DEFAULT CHARACTER SET = utf8mb4
    COLLATE = utf8mb4_general_ci;
```



```
CREATE TABLE `db_test`.`table3_` (
    `Column1` INT(11) UNSIGNED NOT NULL,
    `Column2` TIME NULL,
    `Column3` CHAR NULL,
    PRIMARY KEY (`Column1`)
)   ENGINE = InnoDB
    DEFAULT CHARACTER SET = utf8mb4
    COLLATE = utf8mb4_general_ci;
```

----End

11.5.7 Creating a VPN for the Destination

Step 1 Log in to the management console.

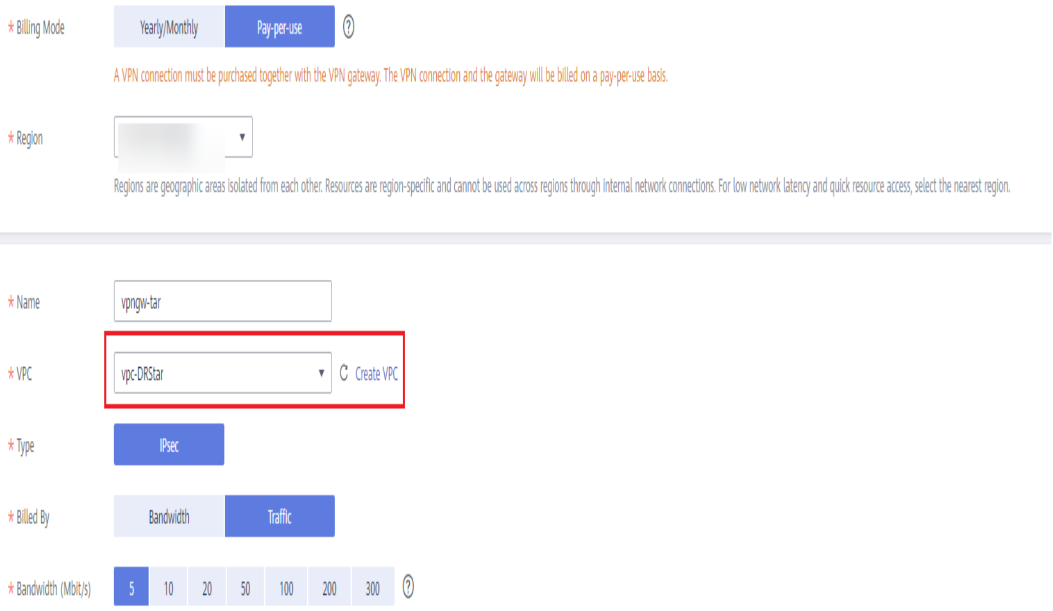
Step 2 Click in the upper left corner and select region CN-Hong Kong.

Step 3 Click the service list icon on the left and choose **Networking** > **Virtual Private Network**.

Step 4 In the navigation pane on the left, choose **Virtual Private Network** > **VPN Gateways**.

Step 5 On the **VPN Gateways** page, click **Buy VPN Gateway**.

Step 6 Specify a gateway name and other information.



* Billing Mode: Yearly/Monthly | Pay-per-use (selected)

A VPN connection must be purchased together with the VPN gateway. The VPN connection and the gateway will be billed on a pay-per-use basis.

* Region:

Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region.

* Name: vpngw-tar

* VPC: vpc-DRStar

* Type: IPsec

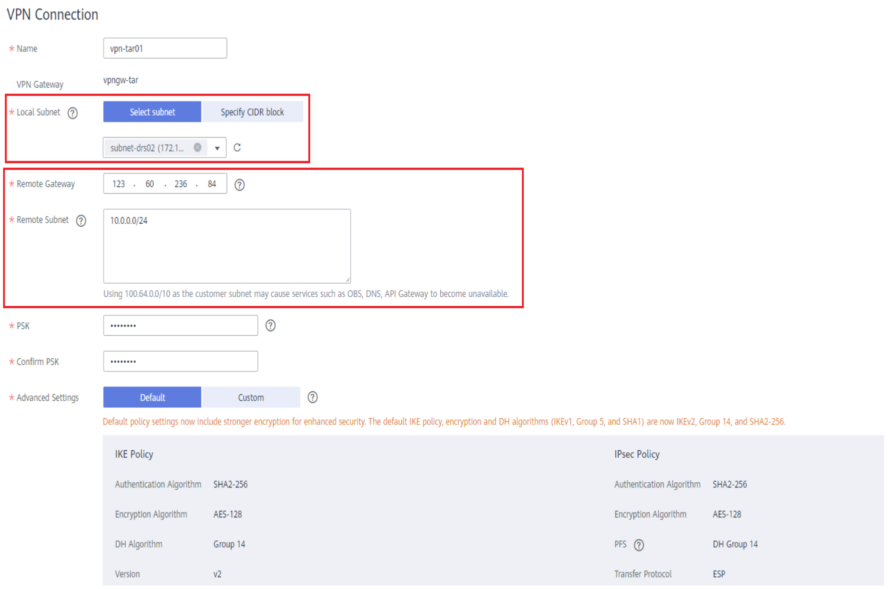
* Billed By: Bandwidth | Traffic (selected)

* Bandwidth (Mbit/s): 5 (selected) | 10 | 20 | 50 | 100 | 200 | 300

CAUTION

The connected VPN gateway must have the same as VPC as the destination DDM instance, that is, the VPC created in **Creating a VPC and Security Group**.

Step 7 Specify the VPN connection information.



VPN Connection

* Name: vpn-tar01

VPN Gateway: vpngw-tar

* Local Subnet: Select subnet (selected) | Specify CIDR block — subnet-drs02 (172.1...

* Remote Gateway: 123 . 60 . 236 . 84

* Remote Subnet: 10.0.0.0/24

Using 100.64.0.0/10 as the customer subnet may cause services such as OBS, DNS, API Gateway to become unavailable.

* PSK: ••••••••

* Confirm PSK: ••••••••

* Advanced Settings: Default (selected) | Custom

Default policy settings now include stronger encryption for enhanced security. The default IKE policy, encryption and DH algorithms (IKEv1, Group 5, and SHA1) are now IKEv2, Group 14, and SHA2-256.

IKE Policy		IPsec Policy	
Authentication Algorithm	SHA2-256	Authentication Algorithm	SHA2-256
Encryption Algorithm	AES-128	Encryption Algorithm	AES-128
DH Algorithm	Group 14	PFS	DH Group 14
Version	v2	Transfer Protocol	ESP

⚠ CAUTION

- The local subnet must be the same as the VPC subnet where the destination DDM instance is located, that is, the subnet created in **Creating a VPC and Security Group**.
- The remote gateway and subnet are the gateway and subnet of the source VPN. Configure parameters based on the VPN created in **Creating a VPN for the Source**.

----End

11.6 Modifying the VPN Configuration for the Source

Step 1 Log in to the **management console**.

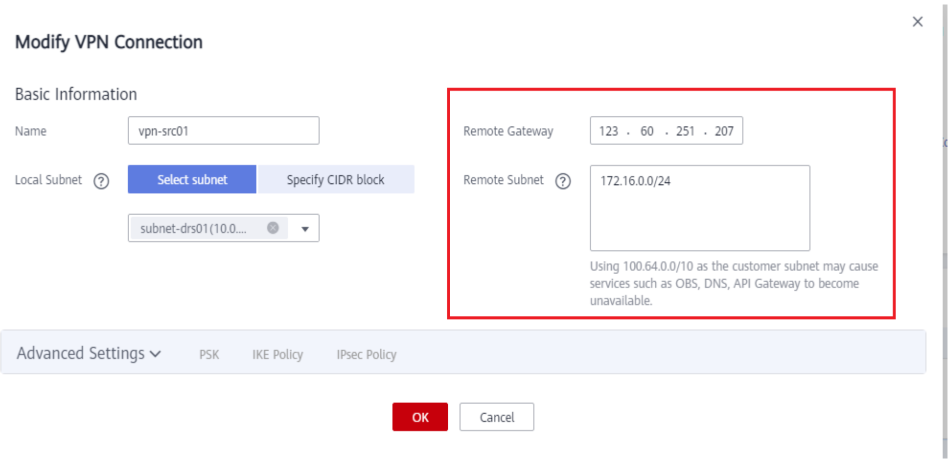
Step 2 Click in the upper left corner and select AP-Singapore.

Step 3 Click the service list icon on the left and choose **Networking** > **Virtual Private Network**.

Step 4 In the navigation pane on the left, choose **Virtual Private Network** > **VPN Connections**.

Step 5 On the **VPN Connections** page, locate the VPN connection created in **Creating a VPN for the Source** and click **Modify** in the **Operation** column.

Step 6 On the **Modify VPN Connection** page, change values of **Remote Gateway** and **Remote Subnet**.



Modify VPN Connection

Basic Information

Name: vpn-src01

Local Subnet: Select subnet | Specify CIDR block

subnet-drs01(10.0....

Remote Gateway: 123 . 60 . 251 . 207

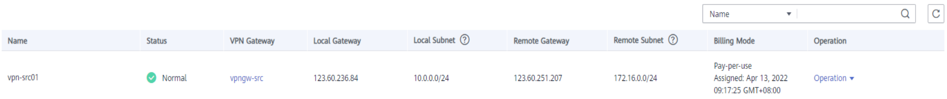
Remote Subnet: 172.16.0.0/24

Using 100.64.0.0/10 as the customer subnet may cause services such as OBS, DNS, API Gateway to become unavailable.

CAUTION

The remote gateway and subnet are the gateway and subnet of the destination VPN. Configure parameters based on the VPN created in **Creating a VPN for the Destination**.

Step 7 After the configuration is complete, view the VPN gateway in the list. The VPN gateway status is **Normal**.



Name	Status	VPN Gateway	Local Gateway	Local Subnet	Remote Gateway	Remote Subnet	Billing Mode	Operation
vpn-src01	✓ Normal	vpngw-src	123.60.236.84	10.0.0.0/24	123.60.251.207	172.16.0.0/24	Pay-per-use Assigned: Apr 13, 2022 09:17:25 GMT+08:00	Operation ▾

----End

11.7 Creating a DRS Migration Task

Create a DRS migration task to migrate data from RDS for MySQL to DDM in a different region.

Pre-Migration Check

Before you create a migration task, check whether migration conditions are met.

This section describes how to migrate data from RDS for MySQL to DDM. For details, see **Before You Start**.

Creating a Migration Task

Step 1 Log in to the **management console**.

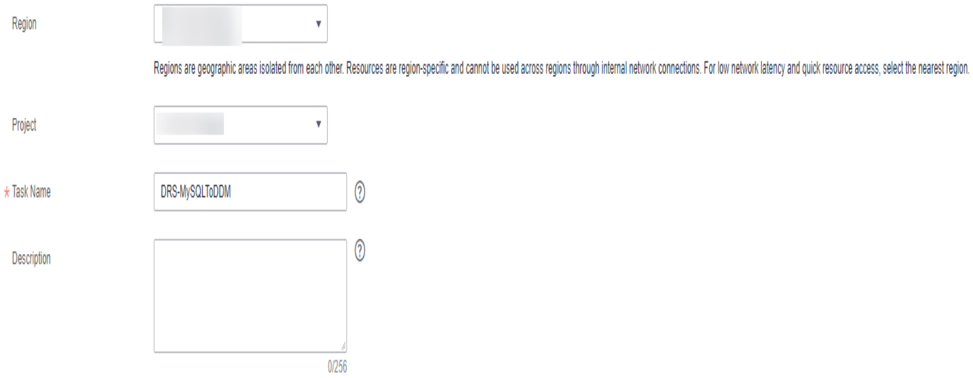
Step 2 Click 📍 in the upper left corner and select region CN-Hong Kong.

Step 3 Click the service list icon on the left and choose **Databases** > **Data Replication Service**.

Step 4 Click **Create Migration Task**.

Step 5 Configure the required parameters.

1. Specify a migration task name.



Region [redacted]

Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region.

Project [redacted]

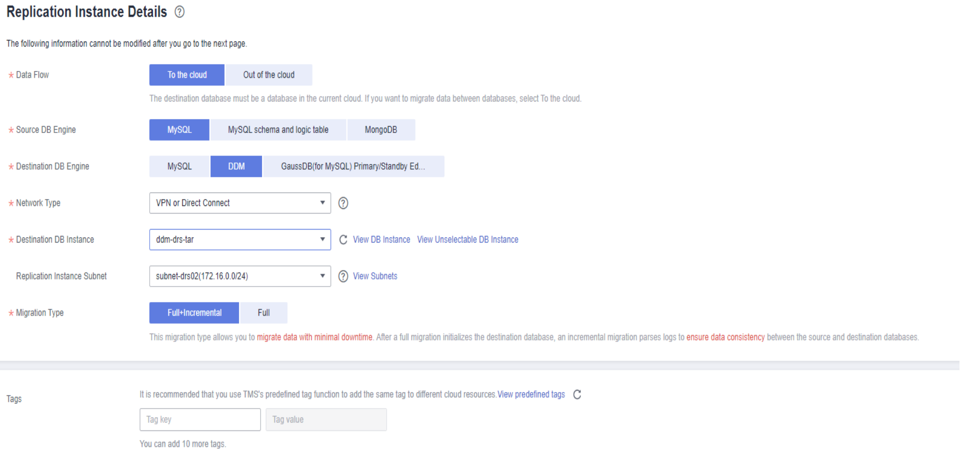
* Task Name DRS-MySQLToDDM

Description

0/256

2. Configure replication instance information and specify a destination DB instance.

   Select the DDM instance created in **Creating a DDM Instance** as the destination.



Replication Instance Details

The following information cannot be modified after you go to the next page.

* Data Flow: To the cloud | Out of the cloud

The destination database must be a database in the current cloud. If you want to migrate data between databases, select To the cloud.

* Source DB Engine: MySQL | MySQL schema and logic table | MongoDB

* Destination DB Engine: MySQL | DDM | GaussDB(for MySQL) Primary/Standby Ed...

* Network Type: VPN or Direct Connect

* Destination DB Instance: ddm-drs-tar View DB Instance View Unselectable DB Instance

Replication Instance Subnet: subnet-drs02(172.16.0.0/24) View Subnets

* Migration Type: Full+Incremental | Full

This migration type allows you to migrate data with minimal downtime. After a full migration initializes the destination database, an incremental migration parses logs to ensure data consistency between the source and destination databases.

Tags

It is recommended that you use TMS's predefined tag function to add the same tag to different cloud resources. View predefined tags

Tag key Tag value

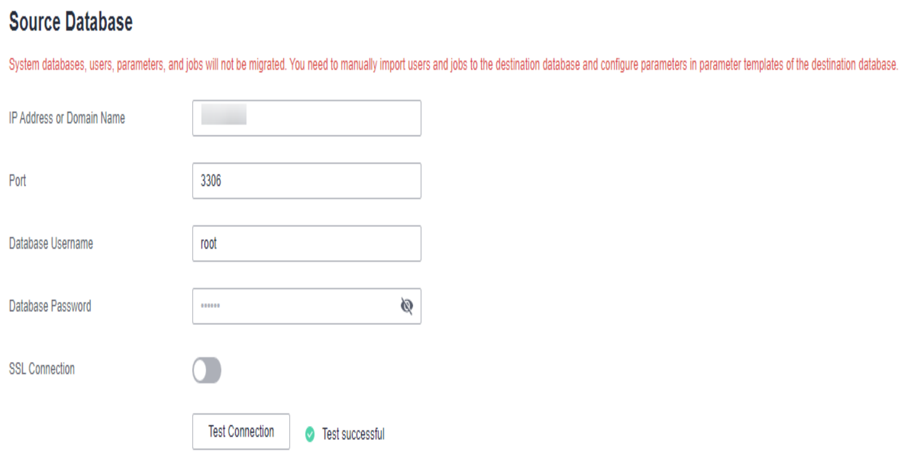
You can add 10 more tags.

Step 6 Click **Next**.

It takes about 5 to 10 minutes to create a replication instance.

Step 7 Configure task information.

1. Configure the source database information and click **Test Connection**. If a successful test message is returned, the database is connected.



Source Database

System databases, users, parameters, and jobs will not be migrated. You need to manually import users and jobs to the destination database and configure parameters in parameter templates of the destination database.

IP Address or Domain Name: [redacted]

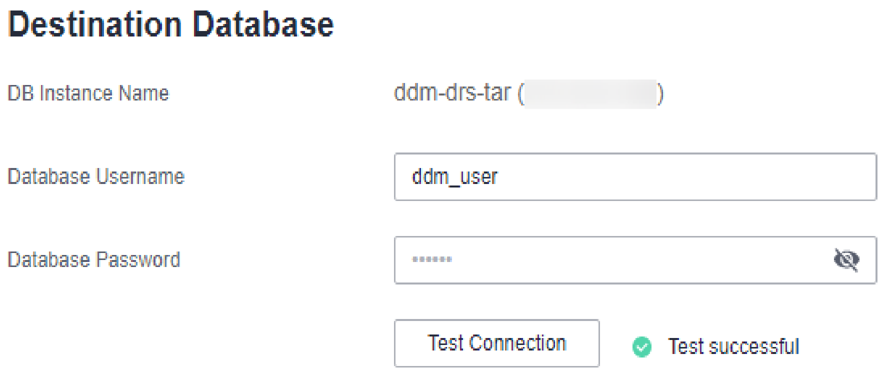
Port: 3306

Database Username: root

Database Password: ••••••

Test successful

2. Configure the destination database information and click **Test Connection**. If a successful test message is returned, the database is connected.



Destination Database

DB Instance Name: ddm-drs-tar ([redacted])

Database Username: ddm_user

Database Password: ••••••

Test successful

Step 8 Click **Next**.

Step 9 On the **Set Task** page, select a migration type.

- **Migrate Object**: Select **Tables**.

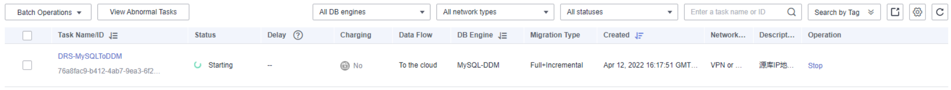
Step 10 Click **Next**. On the **Check Task** page, check the migration task.

- If any check item fails, view the cause and rectify the fault. After the fault is rectified, click **Check Again**.
- If all check items are successful, click **Next**.

Step 11 Click **Submit**.

Return to the **Online Migration Management** page and check the migration task status.

It takes several minutes to complete.



	Task Name/ID	Status	Delay	Charging	Data Flow	DB Engine	Migration Type	Created	Network...	Descript...	Operation
□	DRS-MySQLToDDM 76a8fac9-b412-4ab7-9ea3-6f2...	Starting	--	No	To the cloud	MySQL-DDM	Full+Incremental	Apr 12, 2022 16:17:51 GMT...	VPN or ...	源库IP地...	Stop

If the status changes to **Full migration**, the migration task has been started.

NOTE

- For migration from MySQL to DDM, full migration and full+incremental migration are both supported.
- If you create a full migration task, the task automatically stops after full data is migrated to the destination.
- If you create a full+incremental migration task, a full migration is executed first. After the full migration is complete, an incremental migration starts.
- During the incremental migration, data is continuously migrated so the task will not automatically stop.

----End

11.8 Checking Migration Results

You can use either of the following methods to check the migration results:

1. DRS compares migration objects and data and provides comparison results. For details, see **Viewing Migration Results on the DRS Console**.
2. Log in to the destination to check whether databases, tables, and data are migrated. Confirm the data migration status. For details, see **Viewing Migration Results on the DDM Console**.

Viewing Migration Results on the DRS Console

Step 1 Log in to the **management console**.

Step 2 Click in the upper left corner and select region CN-Hong Kong.

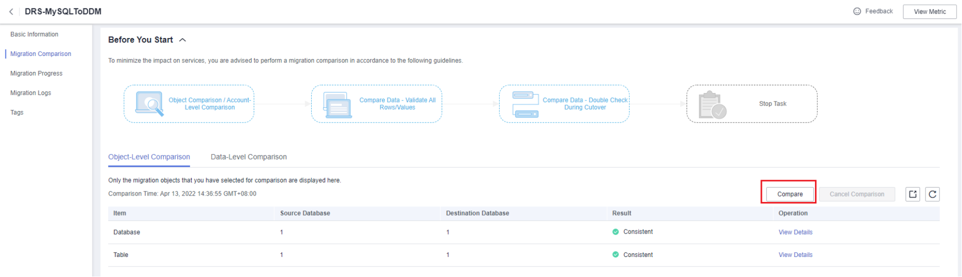
Step 3 Click the service list icon on the left and choose **Databases** > **Data Replication Service**.

Step 4 Locate the required DRS instance and click its name.

Step 5 In the navigation pane on the left, choose **Migration Comparison**.

Step 6 Click the **Object-Level Comparison** tab and check whether some objects are missing.

Click **Compare**. After the comparison is complete, view the comparison results.



To minimize the impact on services, you are advised to perform a migration comparison in accordance to the following guidelines.

Only the migration objects that you have selected for comparison are displayed here.

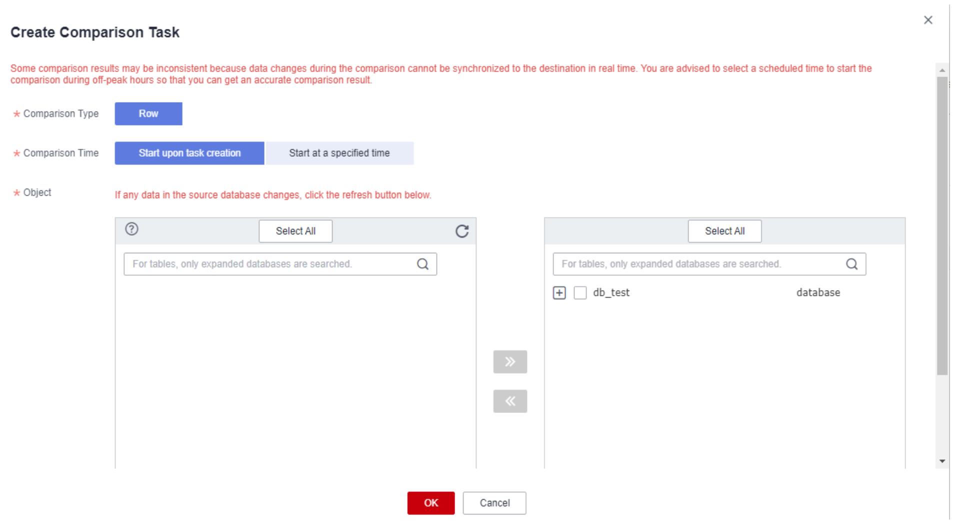
Comparison Time: Apr 13, 2022 14:36:55 GMT+08:00

Item	Source Database	Destination Database	Result	Operation
Database	1	1	Consistent	View Details
Table	1	1	Consistent	View Details

Step 7 Choose **Data-Level Comparison** and check whether the number of rows of migrated objects is consistent between the source and destination.

1. Click **Create Comparison Task**.

2. In the displayed dialog box, select the comparison type, time, and object.



Create Comparison Task

Some comparison results may be inconsistent because data changes during the comparison cannot be synchronized to the destination in real time. You are advised to select a scheduled time to start the comparison during off-peak hours so that you can get an accurate comparison result.

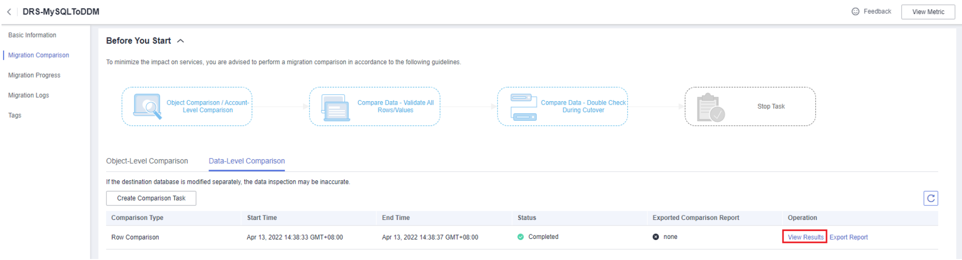
* Comparison Type: Row

* Comparison Time: Start upon task creation | Start at a specified time

* Object: If any data in the source database changes, click the refresh button below.

Select All
For tables, only expanded databases are searched.
Select All
For tables, only expanded databases are searched.
db_test
database
OK
Cancel

3. After the comparison task is complete, view the data comparison results.



DRS-MySQLToDDM

Before You Start

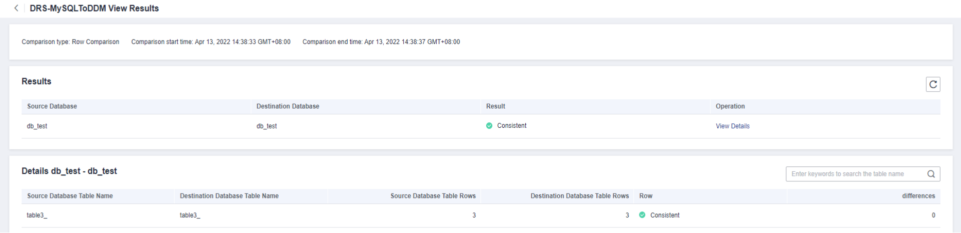
To minimize the impact on services, you are advised to perform a migration comparison in accordance to the following guidelines.

Object Comparison / Account-Level Comparison
Compare Data - Validate All Rows/Values
Compare Data - Double Check During Cutover
Stop Task
Object-Level Comparison
Data-Level Comparison

If the destination database is modified separately, the data inspection may be inaccurate.

Comparison Type	Start Time	End Time	Status	Exported Comparison Report	Operation
Row Comparison	Apr 13, 2022 14:38:33 GMT+08:00	Apr 13, 2022 14:38:37 GMT+08:00	Completed	none	View Results Export Report

4. To view the comparison details, click **View Results** next to the comparison task.



DRS-MySQLToDDM View Results

Comparison type: Row Comparison Comparison start time: Apr 13, 2022 14:38:33 GMT+08:00 Comparison end time: Apr 13, 2022 14:38:37 GMT+08:00

Results

Source Database	Destination Database	Result	Operation
db_test	db_test	Consistent	View Details

Details db_test - db_test

Source Database Table Name	Destination Database Table Name	Source Database Table Rows	Destination Database Table Rows	Row	differences
table3_	table3_	3	3	Consistent	0

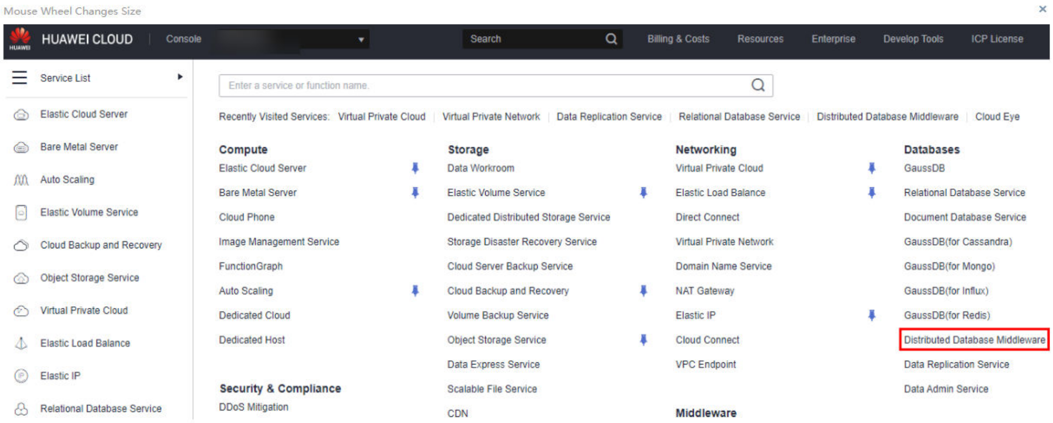
----End

Viewing Migration Results on the DDM Console

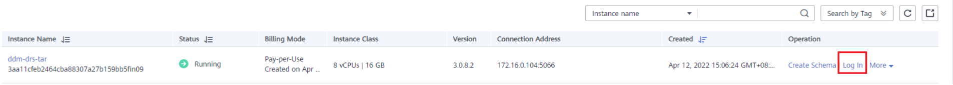
Step 1 Log in to the management console.

Step 2 Click 📍 in the upper left corner and select region CN-Hong Kong.

Step 3 Click the service list icon on the left and choose **Databases** > **Distributed Database Middleware**.



Step 4 Locate the target DDM instance and click **Log In** in the **Operation** column.



Instance Name	Status	Billing Mode	Instance Class	Version	Connection Address	Created	Operation
ddm-drs-tar 3aa11cfeb2464cba88307a27b159bb5fin09	Running	Pay-per-Use Created on Apr ...	8 vCPUs | 16 GB	3.0.8.2	172.16.0.104:5066	Apr 12, 2022 15:06:24 GMT+08:...	Create Schema Log In More

Step 5 In the displayed dialog box, enter the password and click **Test Connection**.

Step 6 After the connection is successful, click **Log In**.

Step 7 View destination database and table names and check whether all data is migrated.

----End

12 Sharding Database and Table Data of an RDS for MySQL Instance

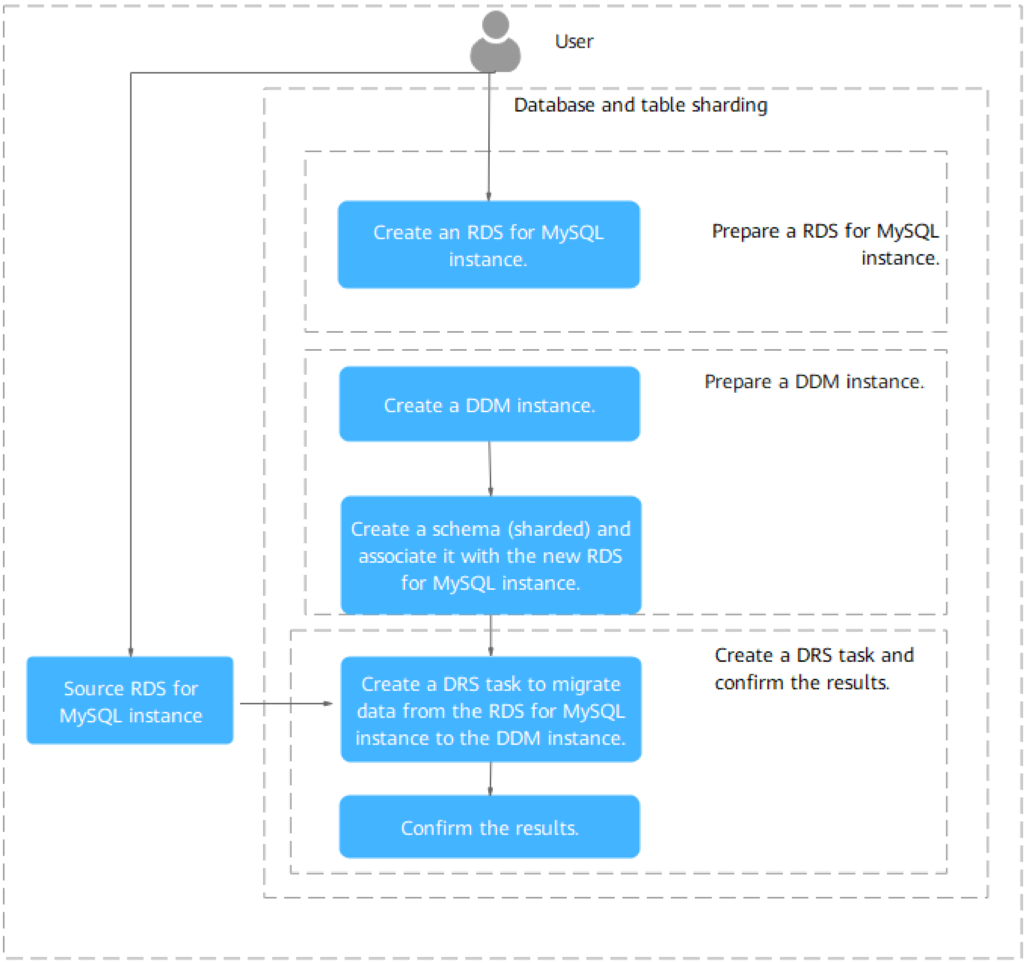
12.1 Overview

Scenarios

This section describes how to shard database and table data of an existing RDS for MySQL instance using DDM.

Operation Process

Figure 12-1 Flowchart


User
Database and table sharding
Create an RDS for MySQL instance.
Prepare a RDS for MySQL instance.
Create a DDM instance.
Prepare a DDM instance.
Create a schema (sharded) and associate it with the new RDS for MySQL instance.
Source RDS for MySQL instance
Create a DRS task to migrate data from the RDS for MySQL instance to the DDM instance.
Create a DRS task and confirm the results.
Confirm the results.

Prerequisites

- You have a Huawei Cloud account.
- There is a new RDS for MySQL instance available.
- The existing RDS for MySQL instance, destination DDM instance, and new RDS for MySQL instance are in the same VPC and have the same security group rules.

12.2 Preparing an RDS for MySQL Instance

There is a new RDS for MySQL instance available. If there are no RDS for MySQL instances available, create one. For details, see **Buying an Instance**.

12.3 Preparing a DDM Instance

12.3.1 Creating a DDM Instance

Procedure

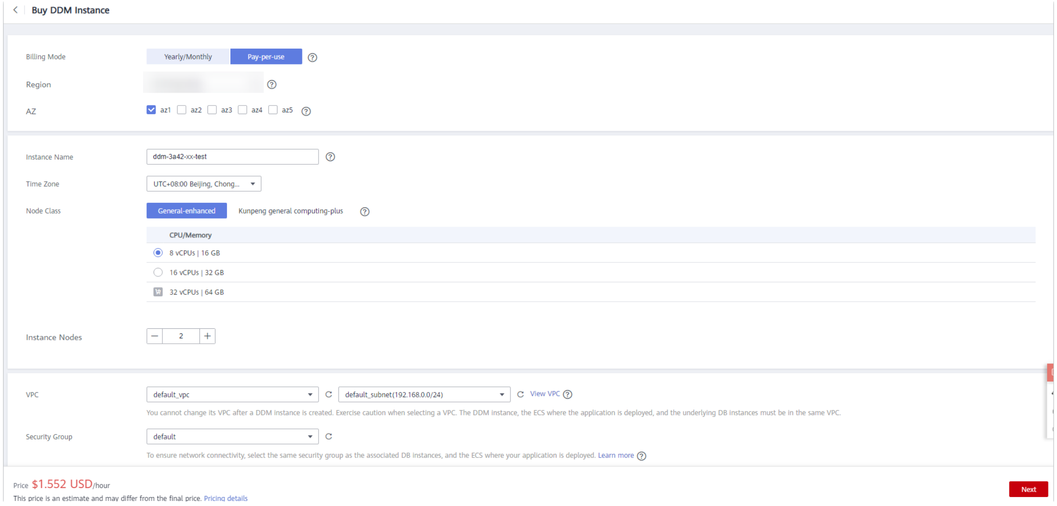
Step 1 Log in to the **management console**.

Step 2 Click the service list icon on the left and choose **Databases** > **Distributed Database Middleware**.

Step 3 On the displayed page, in the upper right corner, click **Buy DDM Instance**.

Step 4 Specify a node class and other information. For parameter details, see **Buying a DDM Instance**.

Figure 12-2 Buying a DDM instance



Buy DDM Instance

Billing Mode: Yearly/Monthly | Pay-per-use

Region:

AZ: ☑ az1 □ az2 □ az3 □ az4 □ az5

Instance Name: ddm-3a42-xx-test

Time Zone: UTC+08:00 Beijing, Chong...

Node Class: General-enhanced | Kunpeng general computing-plus

CPU/Memory
● 8 vCPUs | 16 GB
○ 16 vCPUs | 32 GB
▨ 32 vCPUs | 64 GB

Instance Nodes: 2

VPC: default_vpc | default_subnet(192.168.0.0/24) View VPC

You cannot change its VPC after a DDM instance is created. Exercise caution when selecting a VPC. The DDM instance, the ECS where the application is deployed, and the underlying DB instances must be in the same VPC.

Security Group: default

To ensure network connectivity, select the same security group as the associated DB instances, and the ECS where your application is deployed. Learn more

Price $1.552 USD/hour

This price is an estimate and may differ from the final price. Pricing details

Step 5 After the configuration is complete, click **Next** at the bottom of the page.

Step 6 Confirm the configuration information and perform subsequent operations based on the billing mode you selected.

----End

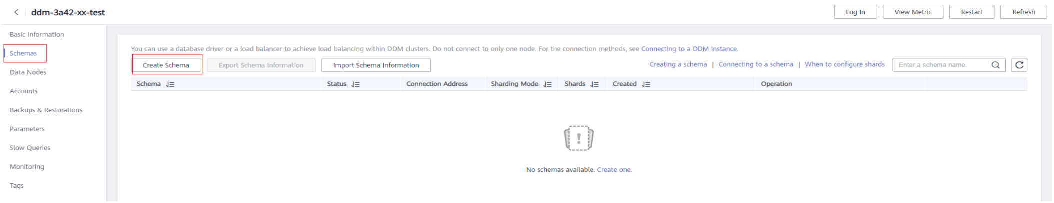
12.3.2 Creating a Schema and Associating It with the Prepared RDS for MySQL Instance

Procedure

Step 1 Locate the prepared DDM instance, click its name, and choose **Schemas** on the displayed page.

Step 2 In the upper left corner, click **Create Schema**.

Figure 12-3 Creating a schema



ddm-3a42-xx-test

You can use a database driver or a load balancer to achieve load balancing within DDM clusters. Do not connect to only one node. For the connection methods, see Connecting to a DDM Instance.

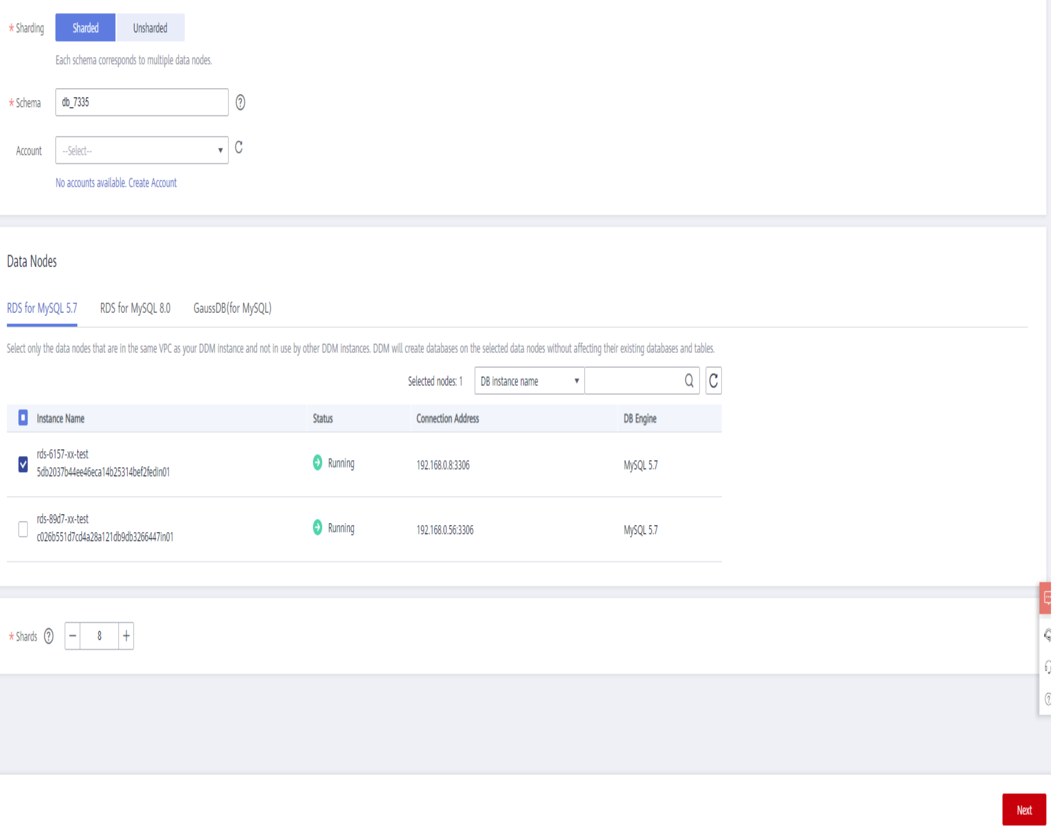
Create Schema
Export Schema Information
Import Schema Information
Creating a schema | Connecting to a schema | When to configure shards
Enter a schema name.

Schema	Status	Connection Address	Sharding Mode	Shards	Created	Operation

No schemas available. Create one.

Step 3 Select **Sharded** for **Sharding**, specify **Shards** based on service requirements, select the RDS for MySQL instance created in **Preparing an RDS for MySQL Instance** for **Data Nodes**, and click **Next**.

Figure 12-4 Selecting an RDS for MySQL instance



* Sharding: [✓ Sharded] Unsharded

Each schema corresponds to multiple data nodes.

* Schema: db_7335

Account: --Select--

No accounts available. Create Account

Data Nodes

RDS for MySQL 5.7 | RDS for MySQL 8.0 | GaussDB(for MySQL)

Select only the data nodes that are in the same VPC as your DDM instance and not in use by other DDM instances. DDM will create databases on the selected data nodes without affecting their existing databases and tables.

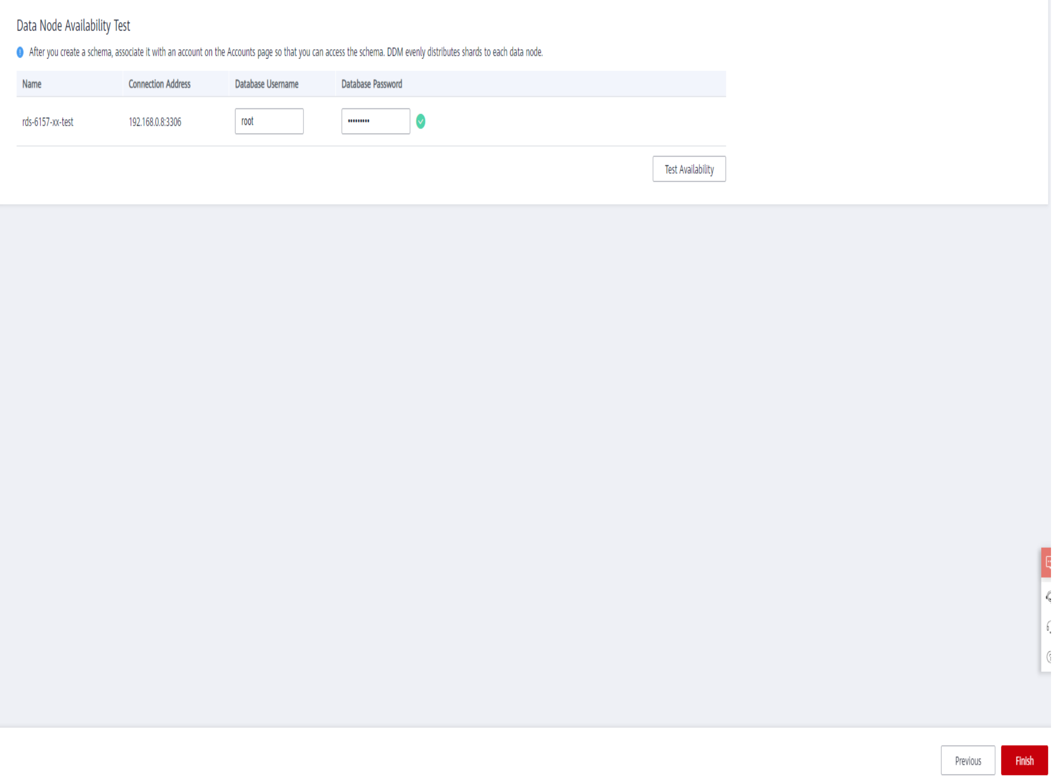
Selected nodes: 1 | DB instance name

	Instance Name	Status	Connection Address	DB Engine
☑	rds-6157-xx-test 5db2037b44ee46eca14b25314bef2fedin01	Running	192.168.0.8:3306	MySQL 5.7
□	rds-89d7-xx-test c026b551d7cd4a28a121db9db3266447in01	Running	192.168.0.56:3306	MySQL 5.7

* Shards: 8

Step 4 Enter the password of the selected RDS for MySQL instance and click **Test Availability**. After the test succeeds, click **Finish**.

Figure 12-5 Testing availability of data nodes



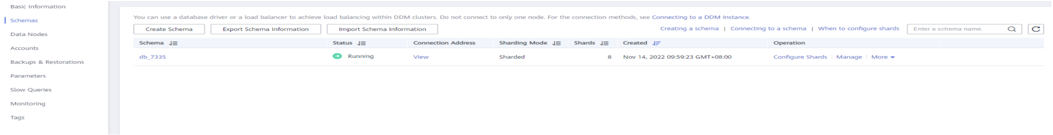
Data Node Availability Test

After you create a schema, associate it with an account on the Accounts page so that you can access the schema. DDM evenly distributes shards to each data node.

Name	Connection Address	Database Username	Database Password
rds-6157-xx-test	192.168.0.8:3306	root	•••••••• ✓

Step 5 View the created schema in the schema list.

Figure 12-6 Schema created successfully



You can use a database driver or a load balancer to achieve load balancing within DDM clusters. Do not connect to only one node. For the connection methods, see Connecting to a DDM Instance.

Create Schema
Export Schema Information
Import Schema Information
Creating a schema | Connecting to a schema | When to configure shards
Enter a schema name.

Schema	Status	Connection Address	Sharding Mode	Shards	Created	Operation
db_7335	Running	View	Sharded	8	Nov 14, 2022 09:59:23 GMT+08:00	Configure Shards | Manage | More

----End

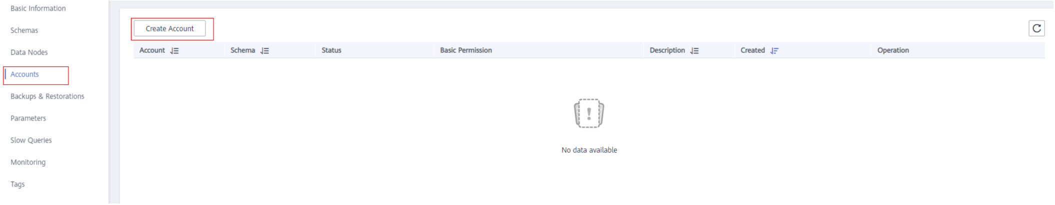
12.3.3 Creating a DDM Account

Procedure

Step 1 Locate the prepared DDM instance, click its name, and choose **Accounts** on the displayed page.

Step 2 In the upper left corner, click **Create Account**.

Figure 12-7 Accounts

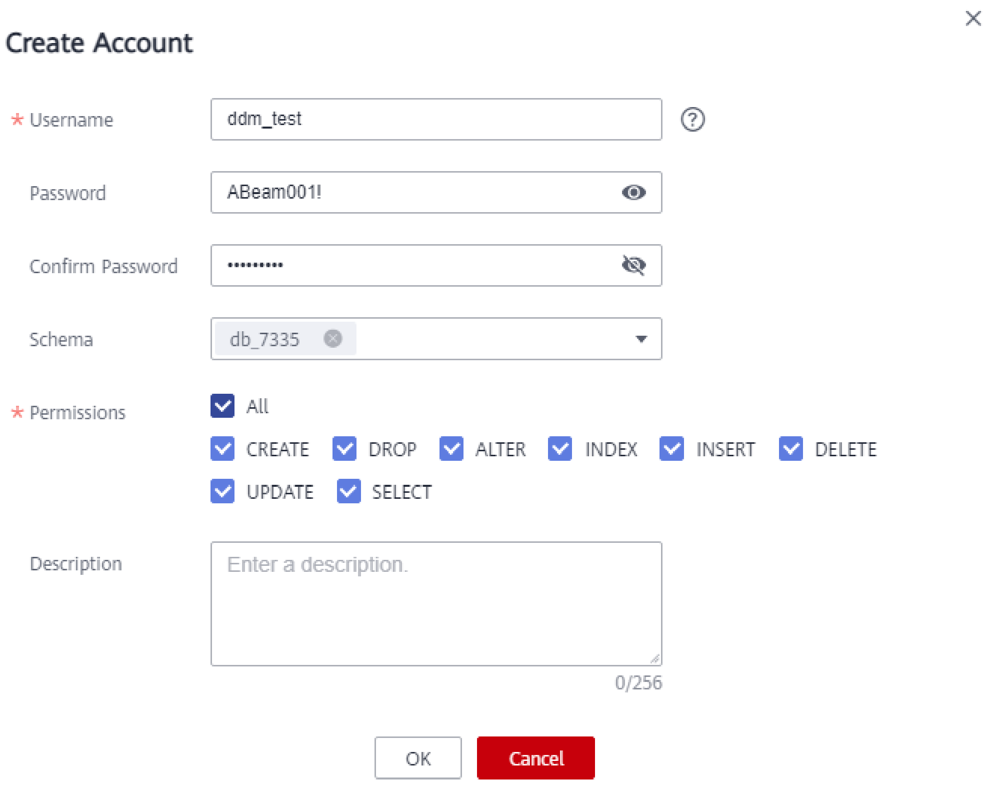


Account	Schema	Status	Basic Permission	Description	Created	Operation

No data available

Step 3 Specify a username, password, and permissions, select the schema created in **Creating a Schema and Associating It with the Prepared RDS for MySQL Instance**, and click **OK**.

Figure 12-8 Creating a DDM account

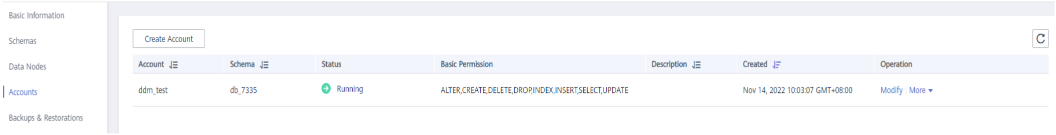


Create Account

* Username	ddm_test
Password	ABeam001!
Confirm Password	•••••••••
Schema	db_7335
* Permissions	☑ All ☑ CREATE ☑ DROP ☑ ALTER ☑ INDEX ☑ INSERT ☑ DELETE ☑ UPDATE ☑ SELECT
Description	Enter a description. 0/256

Step 4 View the created account in the account list.

Figure 12-9 Account created successfully



Account	Schema	Status	Basic Permission	Description	Created	Operation
ddm_test	db_7335	Running	ALTER,CREATE,DELETE,DROP,INDEX,INSERT,SELECT,UPDATE		Nov 14, 2022 10:03:07 GMT+08:00	Modify | More

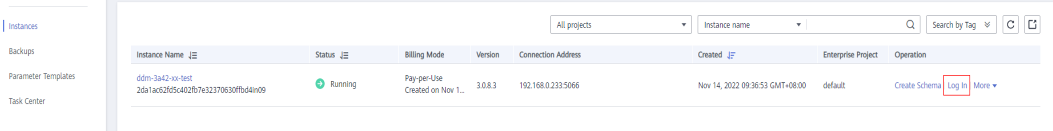
----End

12.3.4 Creating Table Structures in the DDM Instance

Procedure

Step 1 Switch back to the **Instances** page, locate the instance you created and click **Log In** in the **Operation** column.

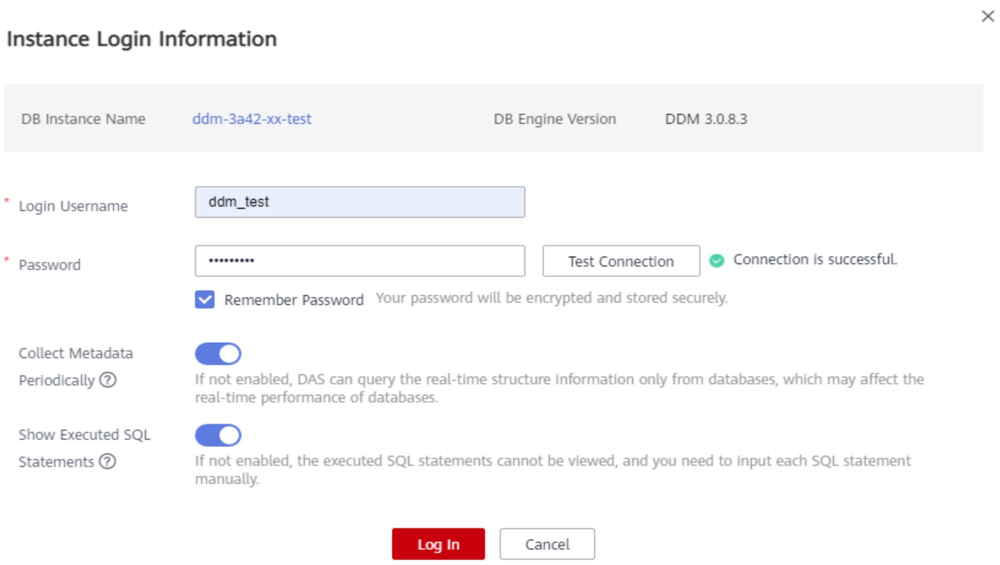
Figure 12-10 Logging in to the DDM instance



Instance Name	Status	Billing Mode	Version	Connection Address	Created	Enterprise Project	Operation
ddm-3a42-xx-test 2da1ac62fd5c402fb7e32370630ffbd4in09	Running	Pay-per-Use Created on Nov 1...	3.0.8.3	192.168.0.233:5066	Nov 14, 2022 09:36:53 GMT+08:00	default	Create Schema Log In More

Step 2 In the displayed dialog box, enter the username and password of the DDM account created in **Creating a DDM Account** and click **Log In**.

Figure 12-11 Login window



Instance Login Information

DB Instance Name ddm-3a42-xx-test DB Engine Version DDM 3.0.8.3

* Login Username: ddm_test

* Password: ••••••••• Test Connection ✓ Connection is successful.

☑ Remember Password Your password will be encrypted and stored securely.

Collect Metadata Periodically: If not enabled, DAS can query the real-time structure information only from databases, which may affect the real-time performance of databases.

Show Executed SQL Statements: If not enabled, the executed SQL statements cannot be viewed, and you need to input each SQL statement manually.

Step 3 Create the same tables as those in the existing RDS instance. You can select to create broadcast, unsharded, or sharded tables based on source table properties. For details about how to use a broadcast and unsharded table, see **Using Broadcast and Unsharded Tables**.

Figure 12-12 Source tables

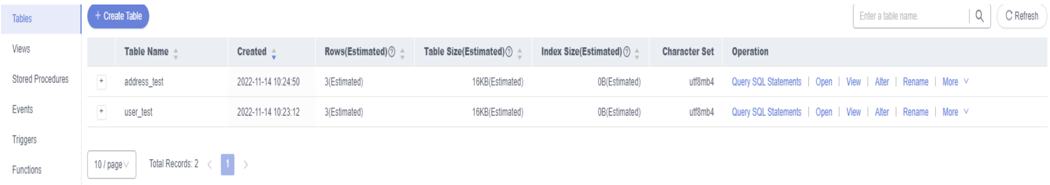


	Table Name	Created	Rows(Estimated)	Table Size(Estimated)	Index Size(Estimated)	Character Set	Operation
+	address_test	2022-11-14 10:24:50	3(Estimated)	16KB(Estimated)	0B(Estimated)	utf8mb4	Query SQL Statements | Open | View | Alter | Rename | More
+	user_test	2022-11-14 10:23:12	3(Estimated)	16KB(Estimated)	0B(Estimated)	utf8mb4	Query SQL Statements | Open | View | Alter | Rename | More

10 / page Total Records: 2 1

Figure 12-13 Data in table address_test

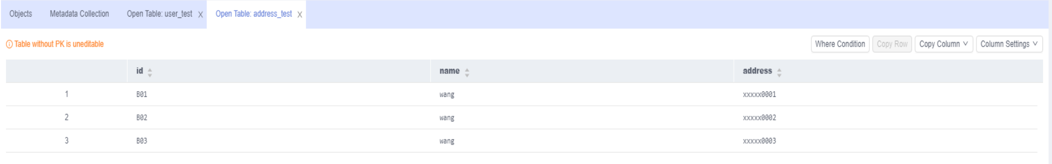
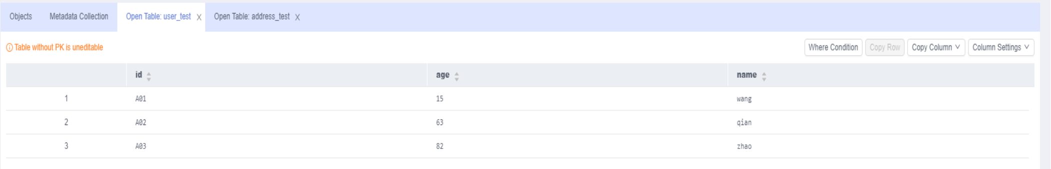


Table without PK is uneditable

	id	name	address
1	B01	wang	xxxxx0001
2	B02	wang	xxxxx0002
3	B03	wang	xxxxx0003

Figure 12-14 Data in table user_test

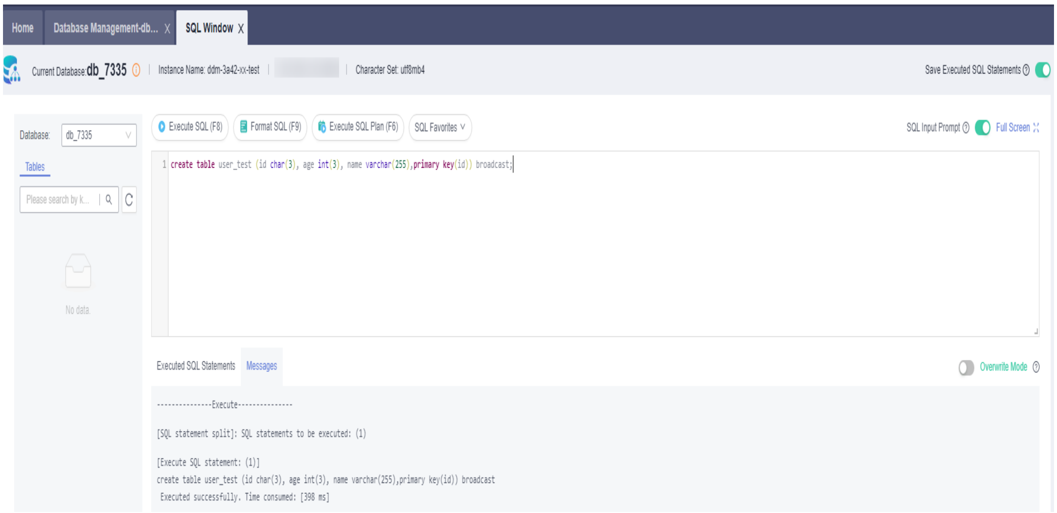


	id	age	name
1	A01	15	wang
2	A02	63	qian
3	A03	82	zhao

Step 4 Run the following command to create a broadcast table as source table **user_test** and a sharded table as source table **address_test**: For SQL statements for creating tables, see **Creating a Table**.

```
create table user_test (id char(3), age int(3), name varchar(255),primary key(id)) broadcast;
```

Figure 12-15 Creating a broadcast table



Current Database: db_7335 | Instance Name: ddm-3a42-xx-test | [redacted] | Character Set: utf8mb4

Database: db_7335

```
create table user_test (id char(3), age int(3), name varchar(255),primary key(id)) broadcast;
```

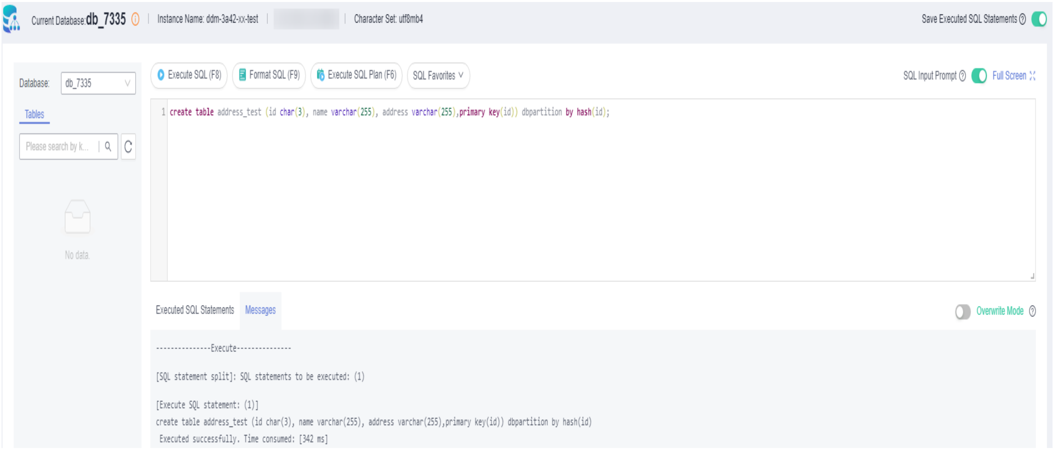
```
--------------Execute--------------

[SQL statement split]: SQL statements to be executed: (1)

[Execute SQL statement: (1)]
create table user_test (id char(3), age int(3), name varchar(255),primary key(id)) broadcast
 Executed successfully. Time consumed: [398 ms]
```

```
create table address_test (id char(3), name varchar(255), address varchar(255),primary key(id)) dbpartition by hash(id);
```

Figure 12-16 Creating a sharded table



Current Database: db_7335 | Instance Name: ddm-3a42-xx-test | [redacted] | Character Set: utf8mb4

Database: db_7335

```
create table address_test (id char(3), name varchar(255), address varchar(255),primary key(id)) dbpartition by hash(id);
```

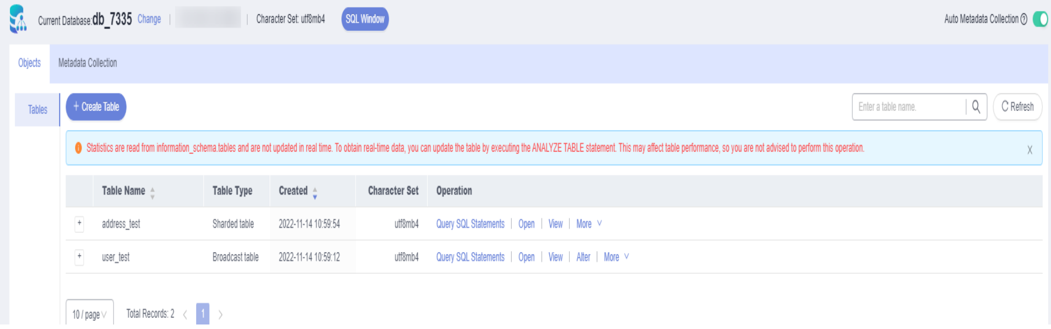
```
--------------Execute--------------

[SQL statement split]: SQL statements to be executed: (1)

[Execute SQL statement: (1)]
create table address_test (id char(3), name varchar(255), address varchar(255),primary key(id)) dbpartition by hash(id)
 Executed successfully. Time consumed: [342 ms]
```

Step 5 View the table creation results.

Figure 12-17 Viewing table creation results



Current Database db_7335 Change | [redacted] | Character Set: utf8mb4 SQL Window Auto Metadata Collection

Objects Metadata Collection

Tables + Create Table Enter a table name. Refresh

Statistics are read from information_schema.tables and are not updated in real time. To obtain real-time data, you can update the table by executing the ANALYZE TABLE statement. This may affect table performance, so you are not advised to perform this operation.

Table Name	Table Type	Created	Character Set	Operation
address_test	Sharded table	2022-11-14 10:59:54	utf8mb4	Query SQL Statements | Open | View | More
user_test	Broadcast table	2022-11-14 10:59:12	utf8mb4	Query SQL Statements | Open | View | Alter | More

10 / page Total Records: 2 1

----End

12.4 Creating a DRS Migration Task

12.4.1 Creating a DRS Migration Task

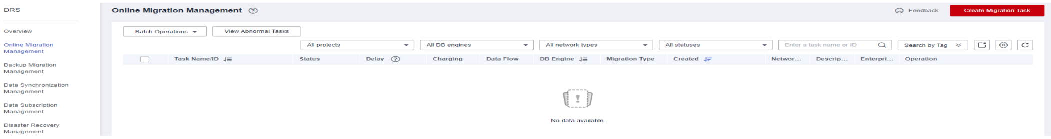
Create a DRS migration task to migrate data from an existing RDS for MySQL instance to a DDM instance in a different region.

Procedure

Step 1 Click the service list icon on the left and choose **Databases** > **Data Replication Service**.

Step 2 Click **Create Migration Task** to migrate data from the existing RDS for MySQL instance to the DDM instance.

Figure 12-18 Creating a migration task



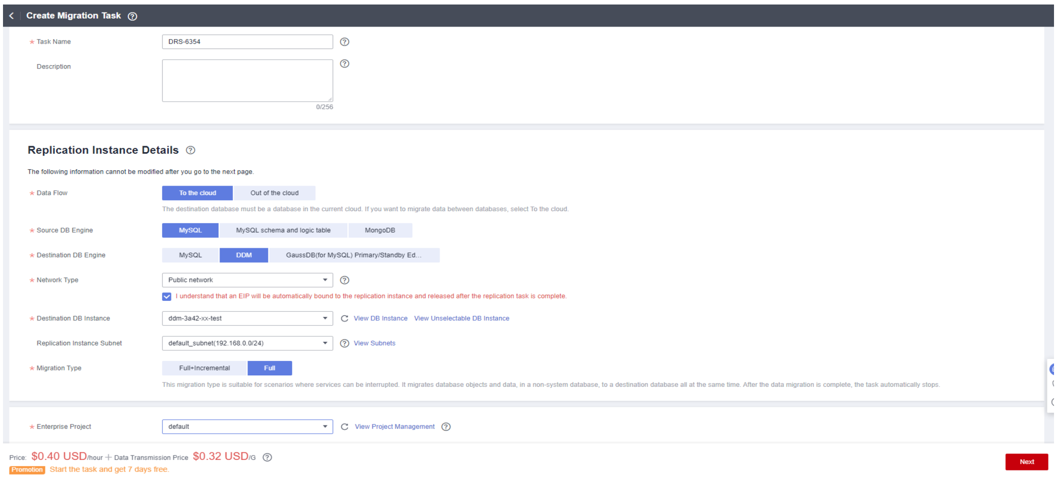
DRS
Overview
Online Migration Management
Backup Migration Management
Data Synchronization Management
Data Subscription Management
Disaster Recovery Management
Online Migration Management
Feedback
Create Migration Task
Batch Operations
View Abnormal Tasks
All projects
All DB engines
All network types
All statuses
Enter a task name or ID
Search by Tag

Task Name/ID	Status	Delay	Charging	Data Flow	DB Engine	Migration Type	Created	Networ...	Descrip...	Enterpri...	Operation

No data available.

Step 3 Configure migration parameters.

Figure 12-19 Entering migration information



Create Migration Task

* Task Name: DRS-6354

Description: 0/256

Replication Instance Details

The following information cannot be modified after you go to the next page.

* Data Flow: To the cloud | Out of the cloud — selected: To the cloud

The destination database must be a database in the current cloud. If you want to migrate data between databases, select To the cloud.

* Source DB Engine: MySQL | MySQL schema and logic table | MongoDB — selected: MySQL

* Destination DB Engine: MySQL | DDM | GaussDB(for MySQL) Primary/Standby Ed... — selected: DDM

* Network Type: Public network

☑ I understand that an EIP will be automatically bound to the replication instance and released after the replication task is complete.

* Destination DB Instance: ddm-3a42-xx-test View DB Instance View Unselectable DB Instance

Replication Instance Subnet: default_subnet(192.168.0.0/24) View Subnets

* Migration Type: Full+Incremental | Full — selected: Full

This migration type is suitable for scenarios where services can be interrupted. It migrates database objects and data, in a non-system database, to a destination database all at the same time. After the data migration is complete, the task automatically stops.

* Enterprise Project: default View Project Management

Price: $0.40 USD/hour + Data Transmission Price $0.32 USD/G

Promotion Start the task and get 7 days free.

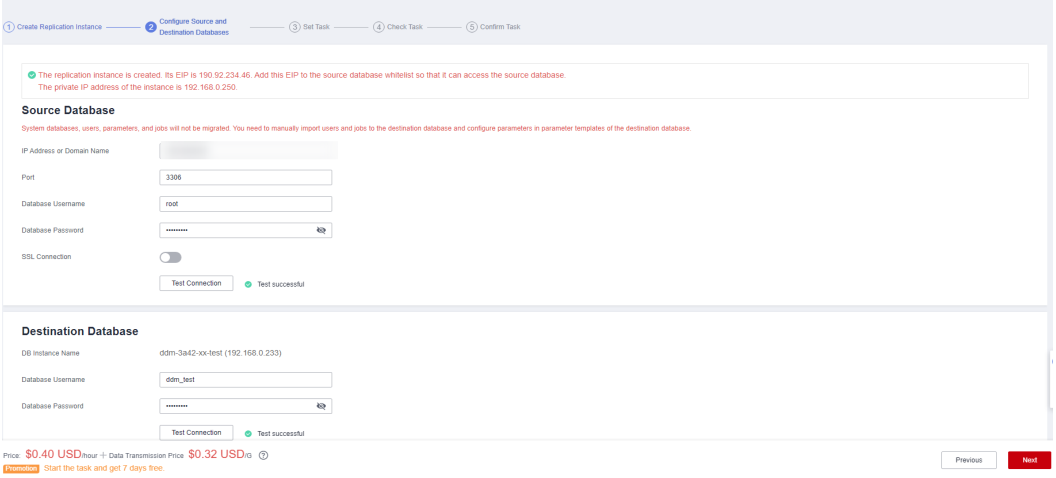
Next

NOTE

- **Data Flow**: Select **To the cloud**.
- **Source DB Engine**: Select **MySQL**.
- **Destination DB Engine**: Select **DDM**.
- **Network Type**: Set this parameter based on service requirements.
- **Destination DB Instance**: Select the DDM instance created in **Creating a DDM Instance**.
- **Migration Type**: Select **Full+Incremental** or **Full**.

Step 4 In the task list, locate the created migration task and click **Edit** in the **Operation** column. At the **Source Database** area, enter the IP address, port, username, and password of the existing RDS for MySQL instance, and click **Test Connection**. At the **Destination Database** area, enter the username and password of the created DDM account, and click **Test Connection**. After both of the connection tests are successful, click **Next**.

Figure 12-20 Editing a migration task


1 Create Replication Instance
2 Configure Source and Destination Databases
3 Set Task
4 Check Task
5 Confirm Task

The replication instance is created. Its EIP is 190.92.234.46. Add this EIP to the source database whitelist so that it can access the source database.
The private IP address of the instance is 192.168.0.250.

Source Database

System databases, users, parameters, and jobs will not be migrated. You need to manually import users and jobs to the destination database and configure parameters in parameter templates of the destination database.

IP Address or Domain Name: [redacted]
Port: 3306
Database Username: root
Database Password: ••••••••
SSL Connection
Test Connection ✓ Test successful

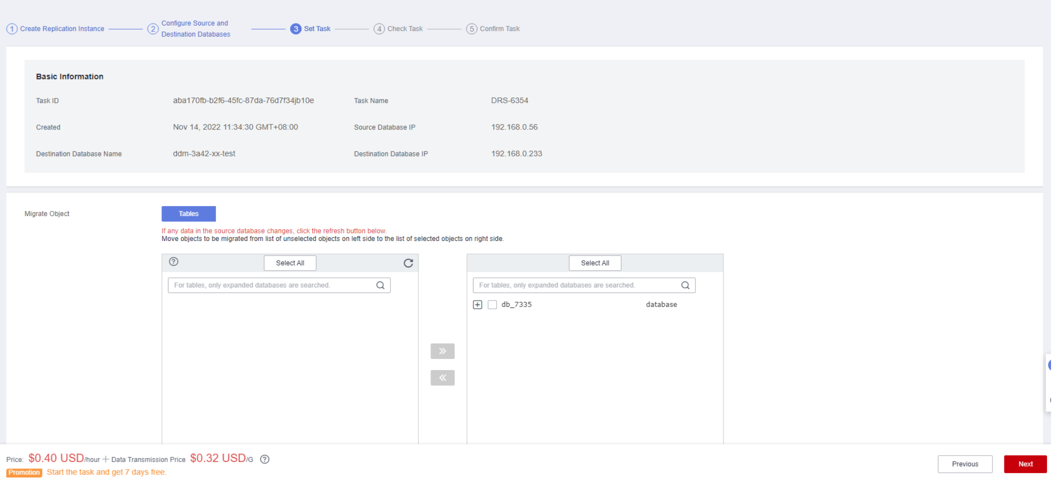
Destination Database

DB Instance Name: ddm-3a42-xx-test (192.168.0.233)
Database Username: ddm_test
Database Password: ••••••••
Test Connection ✓ Test successful

Price: $0.40 USD/hour + Data Transmission Price $0.32 USD/G
Promotion Start the task and get 7 days free.

Step 5 Select the objects you want to migrate and click **Next**.

Figure 12-21 Selecting the objects to be migrated


1 Create Replication Instance
2 Configure Source and Destination Databases
3 Set Task
4 Check Task
5 Confirm Task

Basic Information

Task ID	aba170fb-b2f6-45fc-87da-76d7f34jb10e	Task Name	DRS-6354
Created	Nov 14, 2022 11:34:30 GMT+08:00	Source Database IP	192.168.0.56
Destination Database Name	ddm-3a42-xx-test	Destination Database IP	192.168.0.233

Migrate Object: Tables

If any data in the source database changes, click the refresh button below.
Move objects to be migrated from list of unselected objects on left side to the list of selected objects on right side.

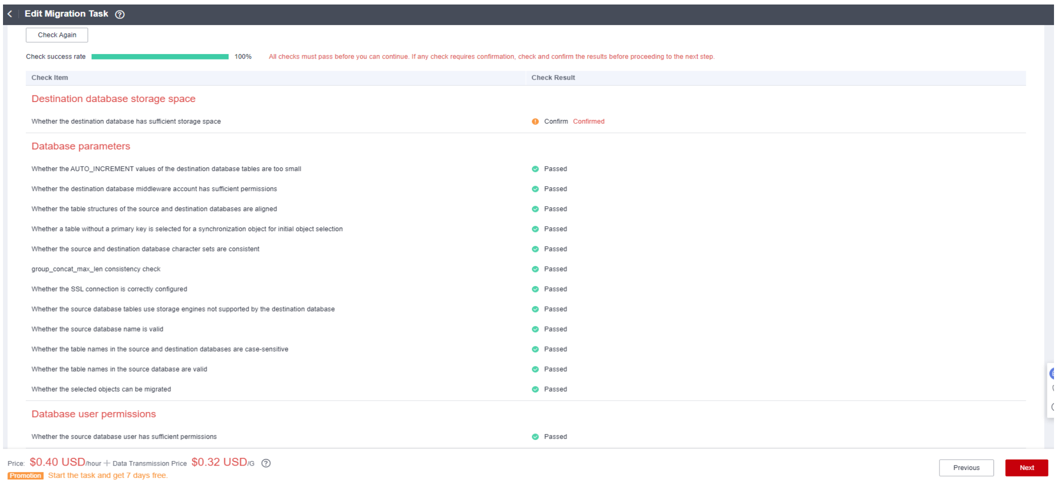
Select All
For tables, only expanded databases are searched.
Select All
For tables, only expanded databases are searched.

□ db_7335 database

Price: $0.40 USD/hour + Data Transmission Price $0.32 USD/G
Promotion Start the task and get 7 days free.

Step 6 On the **Check Task** page, check the migration task. If any check item fails, view the cause and rectify the fault. After the fault is rectified, click **Check Again**. After all check items are passed, click **Next**.

Figure 12-22 Performing a pre-check



Edit Migration Task

Check Again

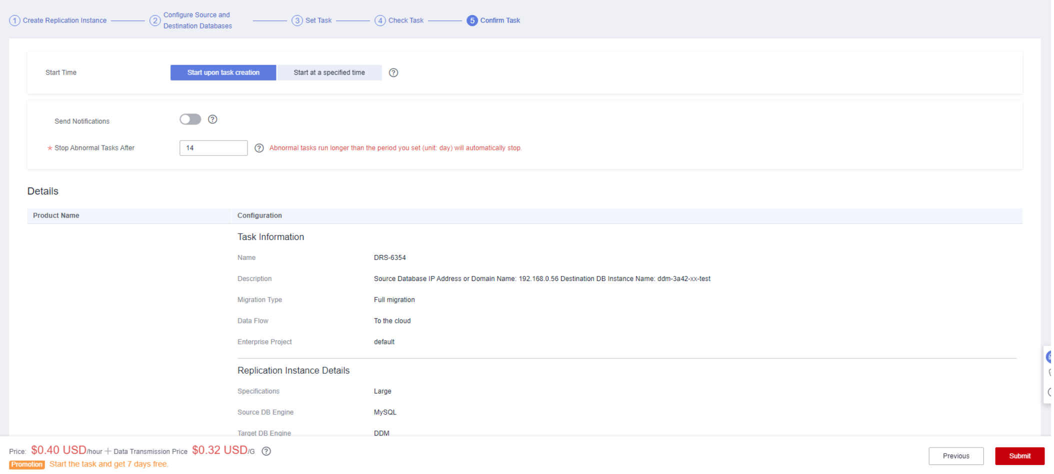
Check success rate 100% All checks must pass before you can continue. If any check requires confirmation, check and confirm the results before proceeding to the next step.

Check Item	Check Result
Destination database storage space	
Whether the destination database has sufficient storage space	Confirm Confirmed
Database parameters	
Whether the AUTO_INCREMENT values of the destination database tables are too small	✓ Passed
Whether the destination database middleware account has sufficient permissions	✓ Passed
Whether the table structures of the source and destination databases are aligned	✓ Passed
Whether a table without a primary key is selected for a synchronization object for initial object selection	✓ Passed
Whether the source and destination database character sets are consistent	✓ Passed
group_concat_max_len consistency check	✓ Passed
Whether the SSL connection is correctly configured	✓ Passed
Whether the source database tables use storage engines not supported by the destination database	✓ Passed
Whether the source database name is valid	✓ Passed
Whether the table names in the source and destination databases are case-sensitive	✓ Passed
Whether the table names in the source database are valid	✓ Passed
Whether the selected objects can be migrated	✓ Passed
Database user permissions	
Whether the source database user has sufficient permissions	✓ Passed

Price: $0.40 USD/hour + Data Transmission Price $0.32 USD/G
Promotion Start the task and get 7 days free.

Step 7 Configure the start of the migration task

Figure 12-23 Start configurations



1 Create Replication Instance
2 Configure Source and Destination Databases
3 Set Task
4 Check Task
5 Confirm Task

Start Time: Start upon task creation | Start at a specified time

Send Notifications

* Stop Abnormal Tasks After: 14 Abnormal tasks run longer than the period you set (unit: day) will automatically stop.

Details

Product Name	Configuration	
	Task Information	
	Name	DRS-6354
	Description	Source Database IP Address or Domain Name: 192.168.0.56 Destination DB Instance Name: ddm-3a42-xx-test
	Migration Type	Full migration
	Data Flow	To the cloud
	Enterprise Project	default
	Replication Instance Details	
	Specifications	Large
	Source DB Engine	MySQL
	Target DB Engine	DDM

Price: $0.40 USD/hour + Data Transmission Price $0.32 USD/G
Promotion Start the task and get 7 days free.

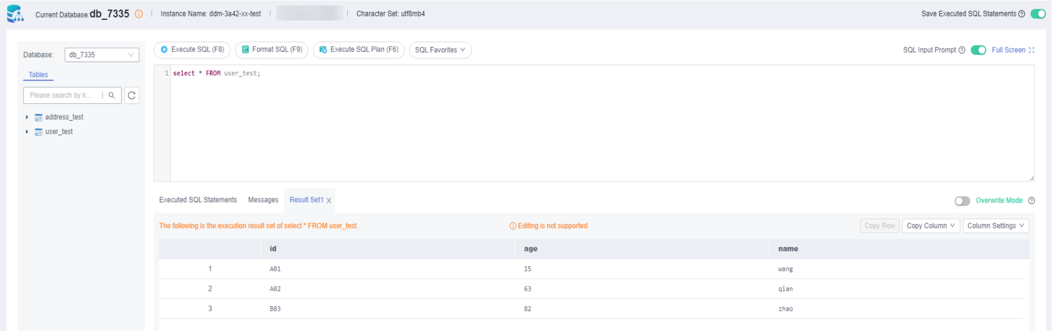
----End

12.4.2 Checking Migration Results

Procedure

Step 1 After the migration task is complete, log in to the DDM instance to view migration results. Ensure that the DDM instance has the same data as the existing RDS for MySQL instance.

Figure 12-24 Viewing data in table **user_test**



Current Database:db_7335 | Instance Name: ddm-3a42-xx-test | [redacted] | Character Set: utf8mb4

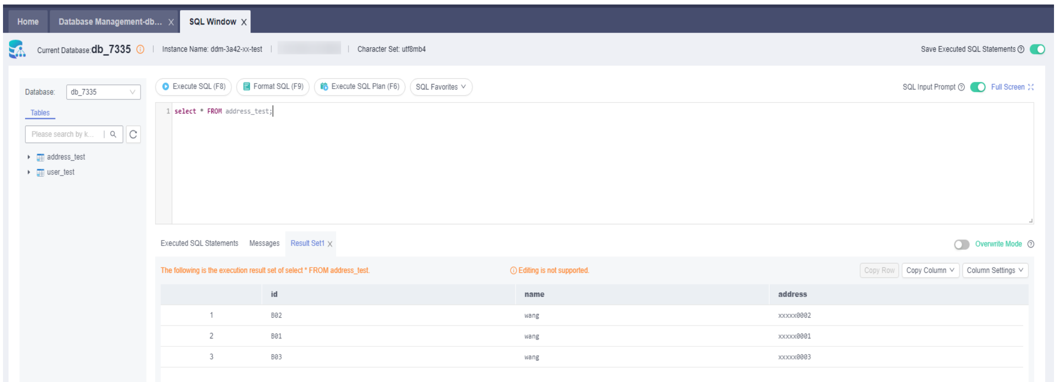
```
select * FROM user_test;
```

The following is the execution result set of select * FROM user_test.

Editing is not supported.

	id	age	name
1	A01	15	wang
2	A02	63	qian
3	B03	82	zhao

Figure 12-25 Viewing data in table **address_test**



Current Database:db_7335 | Instance Name: ddm-3a42-xx-test | [redacted] | Character Set: utf8mb4

```
select * FROM address_test;
```

The following is the execution result set of select * FROM address_test.

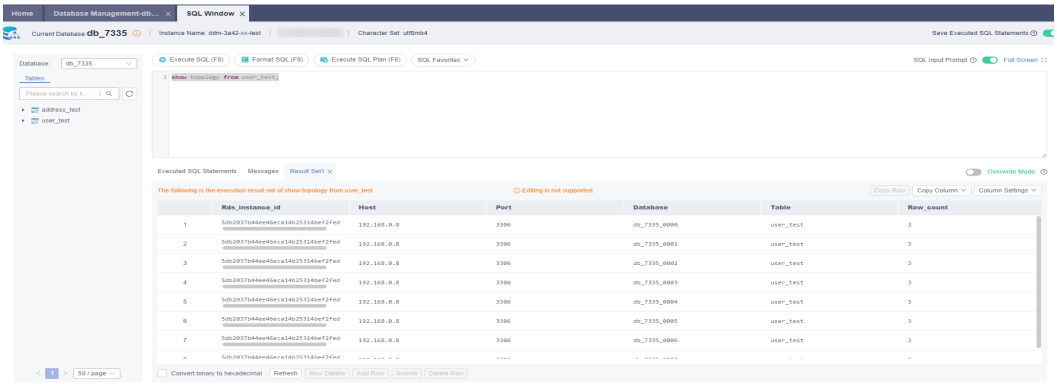
Editing is not supported.

	id	name	address
1	B02	wang	xxxxx0002
2	B01	wang	xxxxx0001
3	B03	wang	xxxxx0003

Step 2 Run the following command to view data distribution of tables. The broadcast table stores the same data in each shard. Data in the sharded table is distributed based on the algorithms you specified.

```
show topology from <table_name>;
```

Figure 12-26 Broadcast table **user_test**



Current Database:db_7335 | Instance Name: ddm-3a42-xx-test | [redacted] | Character Set: utf8mb4

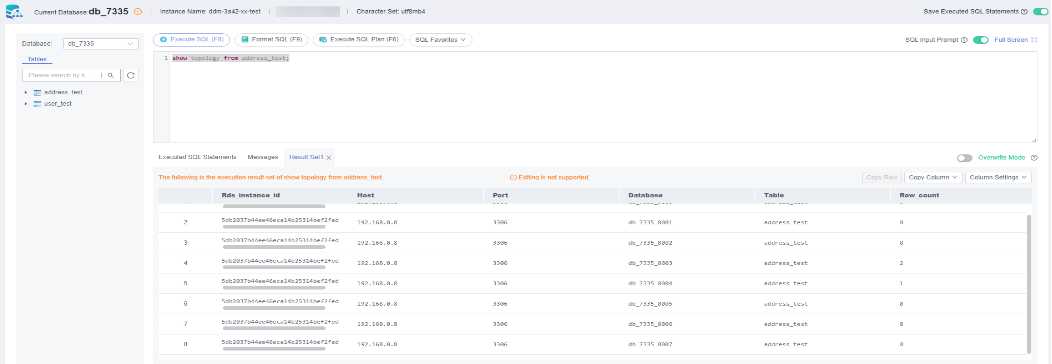
```
show topology from user_test;
```

The following is the execution result set of show topology from user_test.

Editing is not supported.

	Rds_instance_id	Host	Port	Database	Table	Row_count
1	5db2037b44ee46eca14b25314bef2fed	192.168.0.8	3306	db_7335_0000	user_test	3
2	5db2037b44ee46eca14b25314bef2fed	192.168.0.8	3306	db_7335_0001	user_test	3
3	5db2037b44ee46eca14b25314bef2fed	192.168.0.8	3306	db_7335_0002	user_test	3
4	5db2037b44ee46eca14b25314bef2fed	192.168.0.8	3306	db_7335_0003	user_test	3
5	5db2037b44ee46eca14b25314bef2fed	192.168.0.8	3306	db_7335_0004	user_test	3
6	5db2037b44ee46eca14b25314bef2fed	192.168.0.8	3306	db_7335_0005	user_test	3
7	5db2037b44ee46eca14b25314bef2fed	192.168.0.8	3306	db_7335_0006	user_test	3
[illegible]	5db2037b44ee46eca14b25314bef2fed	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Figure 12-27 Sharded table **address_test**



```
show topology from address_test;
```

The following is the execution result set of show topology from address_test.

Editing is not supported.

	Rds_instance_id	Host	Port	Database	Table	Row_count
2	5db2037b44ee46eca14b25314bef2fed	192.168.0.8	3306	db_7335_0001	address_test	0
3	5db2037b44ee46eca14b25314bef2fed	192.168.0.8	3306	db_7335_0002	address_test	0
4	5db2037b44ee46eca14b25314bef2fed	192.168.0.8	3306	db_7335_0003	address_test	2
5	5db2037b44ee46eca14b25314bef2fed	192.168.0.8	3306	db_7335_0004	address_test	1
6	5db2037b44ee46eca14b25314bef2fed	192.168.0.8	3306	db_7335_0005	address_test	0
7	5db2037b44ee46eca14b25314bef2fed	192.168.0.8	3306	db_7335_0006	address_test	0
8	5db2037b44ee46eca14b25314bef2fed	192.168.0.8	3306	db_7335_0007	address_test	0

----End