Distributed Relational Database Service

Quick Start

Quick Start

Buy a DRDS instance

Log on to the DRDS console.

Click **Create Instance** at the upper-right corner.

On the purchase page, set the instance configurations and click **Buy Now**.

Review order information, read and select the service agreement, and click **Pay**.

After the payment is made and the instance is successfully created, you can view the DRDS instance in the instance list on the DRDS console.

Create a DRDS database

After a DRDS instance is created, you can create DRDS databases.

Attention:

DRDS databases can be created only on the console.

When creating a DRDS database, you must select one or more RDS instances as data storage nodes. If you do not have RDS instances, buy at least one before database creation.

Database creation procedure

On the DRDS console, select **Instances** in the left-side navigation pane.

Click the DRDS instance in which you want to create a database.

On the **Basic Information** page, click **Create Database** at the upper-right corner.

Select one or more RDS instances as storage nodes of the DRDS database, click the right arrow, and click **Next Step**.

Note:

> The RDS instances selected as DRDS data storage nodes must meet the following requirements: (1) The instance is a MySQL instance; (2) The instance is in the running state; (3) The RDS is in the same region as the DRDS instance.
>
> If a selected RDS instance has a **master account**, you will be asked to enter the master account and password, which will not be accessed or stored by DRDS and will only be used temporarily in subsequent steps.

Set basic information.

Creation Type indicates the way to use DRDS.

> **Sharded**: Indicates database/table sharding, which splits data into multiple databases and tables according to the sharding rules. The DRDS instance acts as a proxy of SQL. Sharding involves data import/export and SQL function/performance testing and transformation. This may affect the functions and performance of applications.
>
> **Non-Sharded**: The existing RDS databases are transferred to the DRDS instance for proxied access to implement read/write splitting. You only need to change the database connection string, username, and password, without the need to import data or modify code.

Review the database information.

By default, the DRDS instance creates eight physical databases on one RDS instance. The total number of physical databases is the number of RDS instances multiplied by 8.

Click **Next** to create the DRDS database.

The DRDS instance creates the database, account, DRDS system tables, and configurations

through RDS APIs or the master account. Wait until the database creation is complete.

You can view the database creation status in the DRDS database list.

Create a DRDS table

After a DRDS database is created, you need to create tables. The database creation syntax is different from that of a single-host database.

Procedure

Log on to the DRDS console.

Click **Instances** in the left-side navigation pane.

On the "Instances" page, click the name of the instance you want to operate.

Click **Databases** in the left-side navigation pane.

Click the name of the database for which you want to create a table.

On the **Basic Information** page, copy **Command Line URL**, which is the DRDS database connection string.

Connect to the DRDS database through a MySQL client.

For example, you can establish a connection using the MySQL command line as follows:

```
mysql -h${DRDS_IP_ADDRESS} -P${DRDS_PORT} -u${user} -p${password} -D${DRDS_DBNAME}
```

Note:

> If the DRDS instance is an exclusive instance, only the intranet address is provided by default. We recommend that you install the MySQL command line on an ECS instance that is in the same region as the DRDS instance.

If the DRDS instance is a shared instance, you can connect to the DRDS database by using its Internet address from an Internet-based or office computer.

Run the DRDS DDL statement to create a table.

```
//DRDS DDL
CREATE TABLE shard_table(
id int,
name varchar(30),
primary key(id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 dbpartition by hash(id) tbpartition by hash(id) tbpartitions 3;
```

Connect to DRDS

After creating the DRDS instance, database, and table on the console, connect to DRDS to perform futher operations.

Procedure

On the **Basic Information** page of a database, find **Command Line URL**, which is the DRDS connection information, including the IP address or domain name, port number, user name, and password

Connect to the DRDS instance through a **third-party tool** or **program code**.

Third-party tool

DRDS complies with the MySQL protocol; therefore, you can connect to DRDS using a third-party tool.

```
//Run the connection command
mysql -h${DRDS_IP_ADDRESS} -P${DRDS_PORT} -u${user} -p${password} -
D${DRDS_DBNAME}
```

Note:DRDS is fully compatible with the official MySQL command-line clients of version 5.1 or later. DRDS does not support the commands of earlier versions (such as 3.x and 4.x) or infrequently used commands. Therefore, DRDS only guarantees that basic database operations (DDL operations and add/delete/modify/query

data) can be performed on a third-party GUI client.

Program code

You can connect to the DRDS instance through the MySQL driver or a third-party program compliant with the MySQL interaction protocol.

Client tools supported by DRDS

- (Recommended) MySQL Command-Line Tool
- (Recommended) MySQL Workbench
- SQLyog
- Sequel Pro
- Navicat for MySQL

Program driver supported by DRDS

JDBC Driver for MySQL (Connector/J)

```
//JDBC
Class.forName("com.mysql.jdbc.Driver");
Connection conn =
DriverManager.getConnection("jdbc:mysql://drdsxxxxx.drds.aliyuncs.com:3306/doc_test","doc_test","doc_
test_password");
//...
conn.close();
```

- Python Driver for MySQL (Connector/Python)
- C++ Driver for MySQL (Connector/C++)
- C Driver for MySQL (Connector/C)
- ADO.NET Driver for MySQL (Connector/NET)
- ODBC Driver for MySQL (Connector/ODBC)
- PHP Drivers for MySQL (mysqli, ext/mysqli, PDO_MYSQL, PHP_MYSQLND)
- Perl Driver for MySQL (DBD::mysql)
- Ruby Driver for MySQL (ruby-mysql)

Connection string configuration example**

We recommend that you use the Druid connection pool to connect to DRDS. For more information on Druid, see Druid Github.

Configuration example:

```
<bean id="dataSource" class="com.alibaba.druid.pool.DruidDataSource" init-method="init" destroy-method="close">
<property name="url" value="jdbc:mysql://drdsxxxxx.drds.aliyuncs.com:3306/doc_test" />
<property name="username" value="doc_test" />
<property name="password" value="doc_test_password" />

<property name="filters" value="stat" />

<property name="maxActive" value="100" />
<property name="initialSize" value="20" />
<property name="maxWait" value="60000" />
<property name="minIdle" value="1" />

<property name="timeBetweenEvictionRunsMillis" value="60000" />
<property name="minEvictableIdleTimeMillis" value="300000" />

<property name="testWhileIdle" value="true" />
<property name="testOnBorrow" value="false" />
<property name="testOnReturn" value="false" />

<property name="poolPreparedStatements" value="true" />
<property name="maxOpenPreparedStatements" value="20" />

<property name="asyncInit" value="true" />
</bean>
```