Create a new MariaDB database and user

Introduction

One of the most common tasks when administering a database is to oversee access and permissions. MariaDB is an open-source, fully compatible, relational database management system (RDBMS). The MariaDB client makes it easy to add new users and grant them different degrees of privileges.

To identify which database server is used in your stack, run the command below :

```
test -d /opt/bitnami/mariadb && echo "MariaDB" || echo "MySQL"
```

The output of the command indicates which database server (MySQL or MariaDB) is used by the installation, and will allow you to identify which guides to follow in our documentation for common database-related operations.

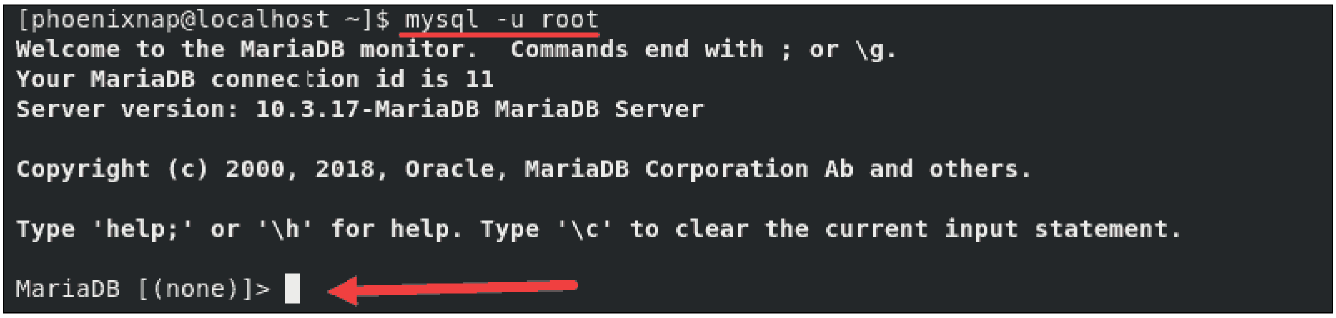
Enter the following command in your command-line terminal to access the MariaDB client shell:

```
sudo mysql -u root
```

If your root user has a predefined password, modify the command to reflect that fact:

```
sudo mysql -u root -p
```

Enter your password and access the MariaDB client.



```
[phoenixnap@localhost ~]$ mysql -u root
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 11
Server version: 10.3.17-MariaDB MariaDB Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

If you do not have any databases created yet, you can easily do so by typing the following command in your MariaDB client shell:

```
CREATE DATABASE 'yourDB';
```

Access a list of existing databases by typing this command:

```
SHOW DATABASES;
```

Create New MariaDB User

Basically, MariaDB is an open source database management software it is useful to store data, retrieve data, and organize data. Normally MariaDB transmits data between client and server without encryption of data; it is possible when client and server run in the same networks. It has a different privilege, or we can say permissions. Let’s see how the create statement works in MariaDB as follows.

Some common permission or privilege are as follows:

- **All Privileges:** In all privileges, it allows all access to the MariaDB user to design database or global access.
- **Create:** Create privilege allows to the user to create a new database or new table.
- **Drop:** In this privilege, the user is able to delete tables or databases.
- **Delete:** In delete privilege, the user has permission to delete rows from a specified table.
- **Insert:** In delete privilege, the user has permission to insert rows from a specified table.
- **Select:** We the help of a select command user can read all databases.
- **Update:** With the help of update privilege, user can update rows from a table.
- **Grant Option:** It allows granting or revoking other user permission or privileges.

Examples of MariaDB create user

To create a new MariaDB user, type the following command:

```
CREATE USER 'user1'@localhost IDENTIFIED BY 'password1';
```

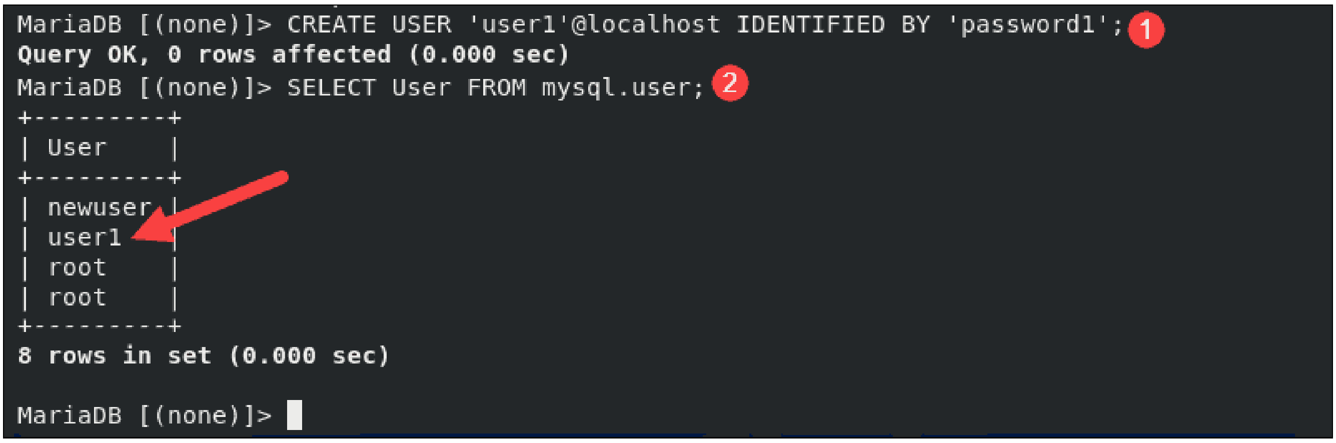
In this case, we use the 'localhost' host-name and not the server's IP. This practice is commonplace if you plan to SSH in to your server, or when using the local client to connect to a local MySQL server.

Note : Substitute user1 and password1 with the credentials for the user you are creating.

Once you create ***user1***, check its status by entering:

```
SELECT User FROM mysql.user;
```

The output lists all existing users.



```
MariaDB [(none)]> CREATE USER 'user1'@localhost IDENTIFIED BY 'password1';
Query OK, 0 rows affected (0.000 sec)
MariaDB [(none)]> SELECT User FROM mysql.user;
+---------+
| User    |
+---------+
| newuser |
| user1   |
| root    |
| root    |
+---------+
8 rows in set (0.000 sec)

MariaDB [(none)]> 
```

Explanation:

- In the above example, we use to create user statement to create a new account on the MariaDB server, here we created a demo user with password pass123 to secure a new user account.
- Here we used the identified clause to create a password for the demo user and test is the localhost name. Here we use localhost to run client and server in the same network. The final output of the above query we illustrate by using the following snapshot.

Grant Privileges to MariaDB User

The newly created user does not have privileges to manage databases nor to access the MariaDB shell.

To grant all privileges to *user1* :

```
GRANT ALL PRIVILEGES ON *.* TO 'user1'@localhost IDENTIFIED BY 'password1';
```

The *.* in the statement refers to the database or table for which the user is given privileges. This specific command provides access to all databases located on the server. As this might be a major security issue, you should replace the symbol with the name of the database you are providing access to.

To grant privileges only for *yourDB*, type the following statement :

```
GRANT ALL PRIVILEGES ON 'yourDB'.* TO 'user1'@localhost;
```

It's crucial to refresh the privileges once new ones have been awarded with the command :

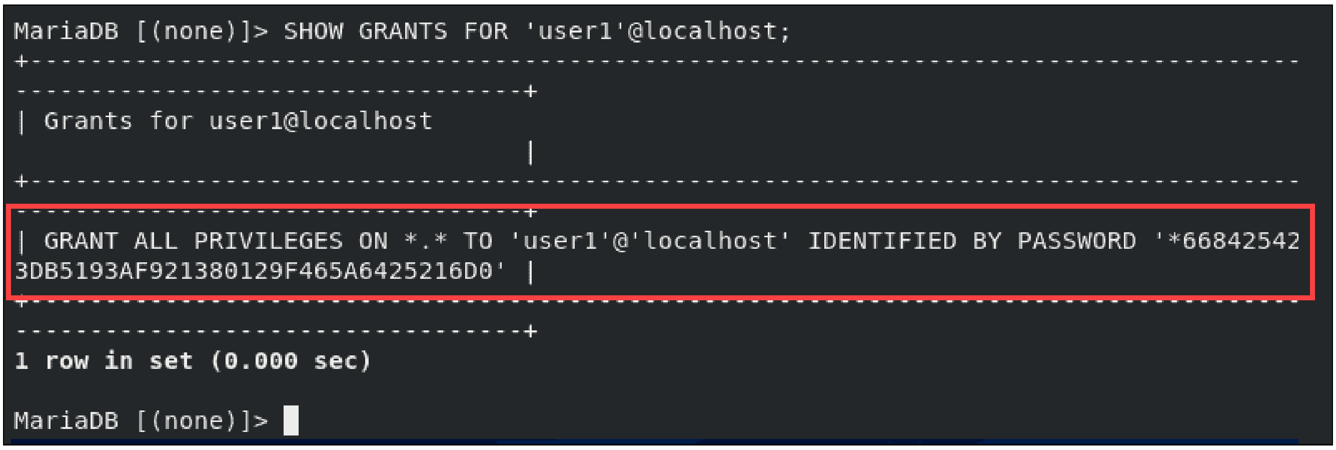
```
FLUSH PRIVILEGES;
```

The user you have created now has full privileges and access to the specified database and tables.

Once you have completed this step, you can verify the new ***user1*** has the right permissions by using the following statement:

```
SHOW GRANTS FOR 'user1'@localhost;
```

The information provided by the system is displayed on the terminal.



```
MariaDB [(none)]> SHOW GRANTS FOR 'user1'@localhost;
+------------------------------------------------------------------------------------
----------------------------------+
| Grants for user1@localhost
                                  |
+------------------------------------------------------------------------------------
----------------------------------+
| GRANT ALL PRIVILEGES ON *.* TO 'user1'@'localhost' IDENTIFIED BY PASSWORD '*66842542
3DB5193AF921380129F465A6425216D0' |
+------------------------------------------------------------------------------------
----------------------------------+
1 row in set (0.000 sec)

MariaDB [(none)]> 
```

Now we can see all grant permission by using the following statement as follows.

Explanation:

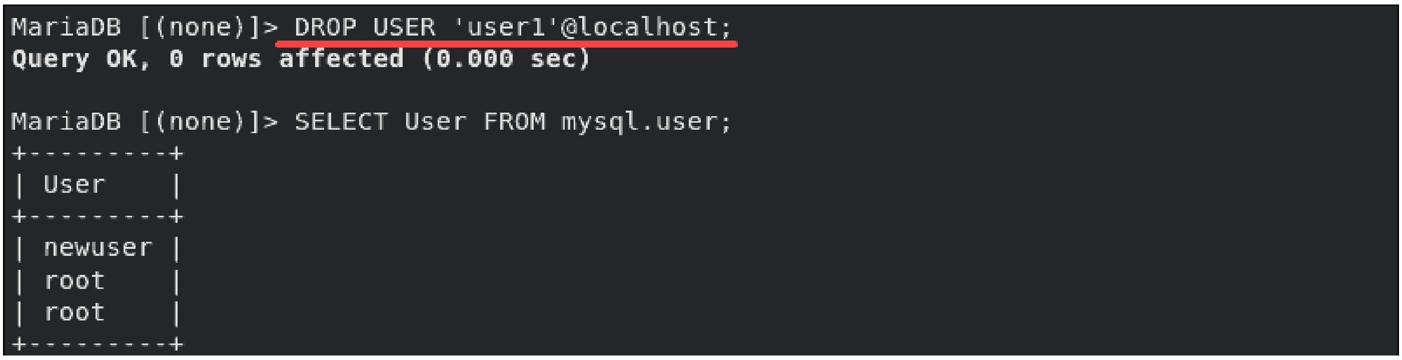
- In the above example, we use the show grants command to see all grants of a specific user; in this example, the user is a demo, and we need all privileges of this user; at that time, we use the above statement.
- The final output of the above query we illustrate by using the following snapshot.

Remove MariaDB User Account

If you need to remove a user, you can employ the `DROP` statement:

```
DROP USER 'user1'@localhost;
```

The output confirms that ***user1*** no longer has access nor privileges.



```
MariaDB [(none)]> DROP USER 'user1'@localhost;
Query OK, 0 rows affected (0.000 sec)

MariaDB [(none)]> SELECT User FROM mysql.user;
+---------+
| User    |
+---------+
| newuser |
| root    |
| root    |
+---------+
```

Example #1

Sometimes a user needs to set a password for a particular time period for security purposes. At that time, we use the following statement.

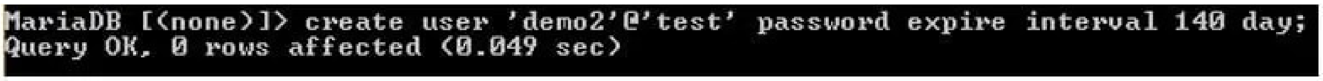
Code:

```
create user 'demo2'@'test' password expire interval 140 day;
```

Explanation:

- In the above example, we use to create user command to create a user; in this example, we created a user name as demo2 on the local environment as shown in the above statement. Here we assign an expiry date of the password that is 140 days.
- After the date of user creation, the final output of the above query we illustrate by using the following snapshot.

Output:



```
MariaDB [(none)]> create user 'demo2'@'test' password expire interval 140 day;
Query OK, 0 rows affected (0.049 sec)
```

Now see created user properties by using the show command as follows.

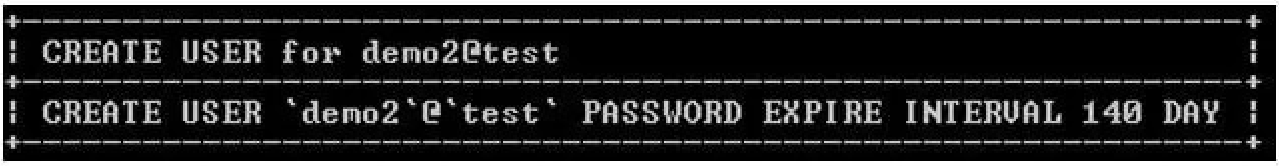
Code:

```
SHOW CREATE USER 'demo2'@'test';
```

Explanation:

- In the above example, we use the show command to see users; in this example, we show details of demo2 users with their local environments.
- The final output of the above query we illustrate by using the following snapshot.

Output:



```
+--------------------------------------------------------------+
| CREATE USER for demo2@test                                   |
+--------------------------------------------------------------+
| CREATE USER `demo2`@`test` PASSWORD EXPIRE INTERVAL 140 DAY  |
+--------------------------------------------------------------+
```

Now let's see how we can assign resource limits to specific users as follows.

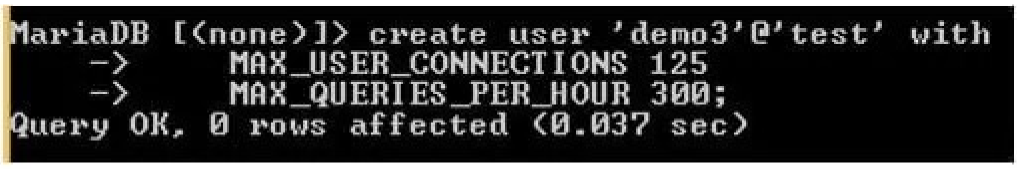
Code:

```
create user 'demo3'@'test' with
MAX_USER_CONNECTIONS 125
MAX_QUERIES_PER_HOUR 300;
```

Explanation:

- In the above example, we used to create a user statement the same as the previous example. Here we created a new user name as demo3 and assigned max user connection as well as max queries per hour as shown above statement.
- The final output of the above query we illustrate by using the following snapshot.

Output:



```
MariaDB [(none)]> create user 'demo3'@'test' with
    ->     MAX_USER_CONNECTIONS 125
    ->     MAX_QUERIES_PER_HOUR 300;
Query OK, 0 rows affected (0.037 sec)
```