MySQLTUTORIAL

MySQL DROP TABLE

Summary: in this tutorial, you will learn how to use the MySQL `DROP TABLE` statement to drop a table from the database.

MySQL DROP TABLE statement syntax

To remove existing tables, you use the MySQL `DROP TABLE` statement.

Here is the basic syntax of the `DROP TABLE` statement:

```
DROP [TEMPORARY] TABLE [IF EXISTS] table_name [, table_name] ...
[RESTRICT | CASCADE]
```

The `DROP TABLE` statement removes a table and its data permanently from the database. In MySQL, you can also remove multiple tables using a single `DROP TABLE` statement, each table is separated by a comma (,).

The `TEMPORARY` option allows you to remove temporary tables (https://www.mysqltutorial.org/mysql-temporary-table/) only. It ensures that you do not accidentally remove non-temporary tables.

The `IF EXISTS` option conditionally drop a table only if it exists. If you drop a non-existing table with the `IF EXISTS` option, MySQL generates a NOTE, which can be retrieved using the `SHOW WARNINGS (https://www.mysqltutorial.org/mysql-stored-procedure/mysql-show-warnings/)` statement.

Note that the `DROP TABLE` statement only drops tables. It doesn't remove specific user privileges associated with the tables. Therefore, if you create a table with the same name as the dropped one, MySQL will apply the existing privileges to the new table, which may pose a security risk.

The `RESTRICT` and `CASCADE` options are reserved for the future versions of MySQL.

To execute the `DROP TABLE` statement, you must have `DROP` privileges for the table that you want to remove.

MySQL DROP TABLE examples

Let's take some examples of using the `DROP TABLE` statement.

A) Using MySQL DROP TABLE to drop a single table example

First, create a table named `insurances` for testing purpose:

```
CREATE TABLE insurances (
    id INT AUTO_INCREMENT,
    title VARCHAR(100) NOT NULL,
    effectiveDate DATE NOT NULL,
    duration INT NOT NULL,
    amount DEC(10 , 2 ) NOT NULL,
    PRIMARY KEY(id)
);
```

Second, use the `DROP TABLE` to delete the `insurances` table:

```
DROP TABLE insurances;
```

A) Using MySQL DROP TABLE to drop multiple tables

First, create two tables named `CarAccessories` and `CarGadgets` :

```
CREATE TABLE CarAccessories (
    id INT AUTO_INCREMENT,
    name VARCHAR(100) NOT NULL,
    price DEC(10 , 2 ) NOT NULL,
    PRIMARY KEY(id)
);

CREATE TABLE CarGadgets (
    id INT AUTO_INCREMENT,
    name VARCHAR(100) NOT NULL,
    price DEC(10 , 2 ) NOT NULL,
    PRIMARY KEY(id)
);
```

Second, use the `DROP TABLE` statement to drop the two tables:

```
DROP TABLE CarAccessories, CarGadgets;
```

C) Using MySQL DROP TABLE to drop a non-existing table

This statement attempts to drop a non-existing table:

```
DROP TABLE aliens;
```

MySQL issued the following error:

```
Error Code: 1051. Unknown table 'classicmodels.aliens'
```

However, if you use the `IF EXISTS` option in the `DROP TABLE` statement:

```
DROP TABLE IF EXISTS aliens;
```

MySQL issued a warning instead:

```
0 row(s) affected, 1 warning(s): 1051 Unknown table 'classicmodels.aliens'
```

To show the warning, you can use the `SHOW WARNINGS` statement:

```
SHOW WARNINGS;
```

MySQL DROP TABLE based on a pattern

Suppose that you have many tables whose names start with `test` in your database and you want to remove all of them using a single `DROP TABLE` statement.

Unfortunately, MySQL does not have the `DROP TABLE LIKE` statement that can remove tables based on pattern matching:

```
DROP TABLE LIKE '%pattern%'
```

However, there are some workarounds. We will discuss one of them here for your reference.

First, create three tables `test1`, `test2`, `test4` for demonstration:

```
CREATE TABLE test1(
  id INT AUTO_INCREMENT,
  PRIMARY KEY(id)
);

CREATE TABLE IF NOT EXISTS test2 LIKE test1;
CREATE TABLE IF NOT EXISTS test3 LIKE test1;
```

Suppose you that want to remove all `test*` tables.

Second, declare two variables that accept database schema and a pattern that you want to the tables to match:

```
-- set table schema and pattern matching for tables
SET @schema = 'classicmodels';
SET @pattern = 'test%';
```

Third, construct a dynamic `DROP TABLE` statement:

```
-- construct dynamic sql (DROP TABLE tbl1, tbl2...;)
SELECT CONCAT('DROP TABLE ',GROUP_CONCAT(CONCAT(@schema,'.',table_name)),';')
INTO @droplike
FROM information_schema.tables
```

```
WHERE @schema = database()
AND table_name LIKE @pattern;
```

Basically, the query instructs MySQL to go to the `information_schema` table, which contains information on all tables in all databases, and to concatenate all tables in the database `@schema` (`classicmodels`) that matches the pattern `@pattern` (*`test%`*) with the prefix `DROP TABLE` . The `GROUP_CONCAT` `(https://www.mysqltutorial.org/mysql-group_concat/)` function creates a comma-separated list of tables.

Fourth, display the dynamic SQL to verify if it works correctly:

```
-- display the dynamic sql statement
SELECT @droplike;
```

As you can see, the output is what we expected.

Fifth, execute the statement using a prepared statement (https://www.mysqltutorial.org/mysql-prepared-statement.aspx) as shown in the following query:

```
-- execute dynamic sql
PREPARE stmt FROM @droplike;
EXECUTE stmt;
DEALLOCATE PREPARE stmt;
```

Putting it all together.

```
-- set table schema and pattern matching for tables
SET @schema = 'classicmodels';
SET @pattern = 'test%';

-- build dynamic sql (DROP TABLE tbl1, tbl2...;)
SELECT CONCAT('DROP TABLE ',GROUP_CONCAT(CONCAT(@schema,'.',table_name)),';')
INTO @droplike
FROM information_schema.tables
WHERE @schema = database()
```

```
AND table_name LIKE @pattern;

-- display the dynamic sql statement
SELECT @droplike;

-- execute dynamic sql
PREPARE stmt FROM @droplike;
EXECUTE stmt;
DEALLOCATE PREPARE stmt;
```

So if you want to drop multiple tables that have a specific pattern in a database, you just use the script above to save time. All you need to do is replacing the *pattern* and the *database schema* in `@pattern` and `@schema` variables. If you often have to deal with this task, you can develop a stored procedure (https://www.mysqltutorial.org/mysql-stored-procedure-tutorial.aspx) based on the script and reuse this stored procedure.

In this tutorial, you have learned how to use the MySQL `DROP TABLE` statement to remove existing tables from a database.