MySQLTUTORIAL

MySQL CREATE TABLE

Summary: in this tutorial, we will show you how to use the MySQL `CREATE TABLE` statement to create a new table in the database.

MySQL CREATE TABLE syntax

The `CREATE TABLE` statement allows you to create a new table in a database.

The following illustrates the basic syntax of the `CREATE TABLE` statement:

```
CREATE TABLE [IF NOT EXISTS] table_name(
   column_1_definition,
   column_2_definition,
   ...,
   table_constraints
) ENGINE=storage_engine;
```

Let's examine the syntax in greater detail.

First, you specify the name of the table that you want to create after the `CREATE TABLE` keywords. The table name must be unique within a database. The `IF NOT EXISTS` is optional. It allows you to check if the table that you create already exists in the database. If this is the case, MySQL will ignore the whole statement and will not create any new table.

Second, you specify a list of columns of the table in the `column_list` section, columns are separated by commas.

Third, you can optionally specify the storage engine (https://www.mysqltutorial.org/understand-mysql-table-types-innodb-myisam.aspx) for the table in the `ENGINE` clause. You can use any storage engine such as InnoDB and MyISAM. If you don't explicitly declare a storage engine, MySQL will use InnoDB by default.

InnoDB became the default storage engine since MySQL version 5.5. The InnoDB storage engine brings many benefits of a relational database management system such as ACID transaction, referential integrity,

and crash recovery. In the previous versions, MySQL used MyISAM as the default storage engine.

The following shows the syntax for a column's definition:

```
column_name data_type(length) [NOT NULL] [DEFAULT value] [AUTO_INCREMENT] column_constraint;
```

Here are the details:

- The `column_name` specifies the name of the column. Each column has a specific data type (https://www.mysqltutorial.org/mysql-data-types.aspx) and optional size e.g., `VARCHAR(255)`
- The `NOT NULL` (https://www.mysqltutorial.org/mysql-not-null-constraint/) constraint ensures that the column will not contain `NULL`. Besides the `NOT NULL` constraint, a column may have additional constraint such as CHECK (https://www.mysqltutorial.org/mysql-check-constraint/), and UNIQUE (https://www.mysqltutorial.org/mysql-unique-constraint/).
- The `DEFAULT` specifies a default value for the column.
- The `AUTO_INCREMENT` (https://www.mysqltutorial.org/mysql-sequence/) indicates that the value of the column is incremented by one automatically whenever a new row is inserted (https://www.mysqltutorial.org/mysql-insert-statement.aspx) into the table. Each table has a maximum one `AUTO_INCREMENT` column.

After the column list, you can define table constraints such as UNIQUE (https://www.mysqltutorial.org/mysql-unique-constraint/), CHECK (https://www.mysqltutorial.org/mysql-check-constraint/), PRIMARY KEY (https://www.mysqltutorial.org/mysql-primary-key/) and FOREIGN KEY (https://www.mysqltutorial.org/mysql-foreign-key/).

For example, if you want to set a column or a group of columns as the primary key, you use the following syntax:

```
PRIMARY KEY (col1,col2,...)
```

MySQL CREATE TABLE statement examples

Let's take some examples of creating new tables.

1) MySQL CREATE TABLE simple example

The following statement creates a new table named `tasks`:

```
CREATE TABLE IF NOT EXISTS tasks (
    task_id INT AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(255) NOT NULL,
    start_date DATE,
    due_date DATE,
    status TINYINT NOT NULL,
    priority TINYINT NOT NULL,
    description TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
)  ENGINE=INNODB;
```

The tasks table has the following columns:

- The `task_id` is an auto-increment column. If you use the `INSERT (https://www.mysqltutorial.org/mysql-insert-statement.aspx)` statement to insert a new row into the table without specifying a value for the `task_id` column, MySQL will automatically generate a sequential integer for the `task_id` starting from 1.
- The `title` column is a variable character string column whose maximum length is 255. It means that you cannot insert a string whose length is greater than 255 into this column. The `NOT NULL (https://www.mysqltutorial.org/mysql-not-null-constraint/)` constraint indicates that the column does not accept `NULL`. In other words, you have to provide a non-NULL value when you insert (https://www.mysqltutorial.org/mysql-insert-statement.aspx) or update (https://www.mysqltutorial.org/mysql-update-data.aspx) this column.
- The `start_date` and `due_date` are `DATE (https://www.mysqltutorial.org/mysql-date-functions/)` columns. Because these columns do not have the `NOT NULL` constraint, they can store `NULL`. The start_date column has a default value of the current date. In other words, if you don't provide a value for the start_date column when you insert a new row, the start_date column will take the current date of the database server.
- The `status` and `priority` are the `TINYINT (https://www.mysqltutorial.org/mysql-int/)` columns which do not allow `NULL`.
- The `description` column is a `TEXT (https://www.mysqltutorial.org/mysql-text/)` column that accepts `NULL`.

- The `created_at` is a `TIMESTAMP` (https://www.mysqltutorial.org/mysql-timestamp.aspx) column that accepts the current time as the default value.

The `task_id` is the primary key column of the `tasks` table. It means that the values in the `task_id` column will uniquely identify rows in the table.

Once you execute the `CREATE TABLE` statement to create the `tasks` table, you can view its structure by using the `DESCRIBE` statement:

```
DESCRIBE tasks;
```

This picture shows the database diagram of the `tasks` table:

2) MySQL CREATE TABLE with a foreign key primary key example

Suppose each task has a checklist or to-do list. To store checklists of tasks, you can create a new table named `checklists` as follows:

```
CREATE TABLE IF NOT EXISTS checklists (
    todo_id INT AUTO_INCREMENT,
    task_id INT,
```

```
    todo VARCHAR(255) NOT NULL,
    is_completed BOOLEAN NOT NULL DEFAULT FALSE,
    PRIMARY KEY (todo_id , task_id),
    FOREIGN KEY (task_id)
        REFERENCES tasks (task_id)
        ON UPDATE RESTRICT ON DELETE CASCADE
);
```

The table `checklists` has a primary key that consists of two columns. Therefore, we used a table constraint to define the primary key (https://www.mysqltutorial.org/mysql-primary-key/) :

```
PRIMARY KEY (todo_id , task_id)
```

In addition, the `task_id` is the foreign key column that references to the `task_id` column of the table `tasks` , we used a foreign key constraint to establish this relationship:

```
FOREIGN KEY (task_id)
    REFERENCES tasks (task_id)
    ON UPDATE RESTRICT
    ON DELETE CASCADE
```

You will learn more about the foreign key constraint (https://www.mysqltutorial.org/mysql-foreign-key/) in the subsequent tutorial.

This picture illustrates the `checklists` table and its relationship with the `tasks` table:

In this tutorial, you have learned how to use MySQL `CREATE TABLE` statement to create a new table in the database.