Home MySQL SQL PL/SQL MongoDB PostgreSQL SQL Server Oracle SQLite Cassandra

MySQL Create Trigger

In this article, we are going to learn how to create the first trigger in MySQL. We can create a new trigger in MySQL by using the CREATE TRIGGER statement. It is to ensure that we have trigger privileges while using the CREATE TRIGGER command. The following is the basic syntax to create a trigger:

```
CREATE TRIGGER trigger_name  trigger_time trigger_event
ON table_name FOR EACH ROW
BEGIN
   --variable declarations
   --trigger code
END;
```

Parameter Explanation

The create trigger syntax contains the following parameters:

trigger_name: It is the name of the trigger that we want to create. It must be written after the CREATE TRIGGER statement. It is to make sure that the trigger name should be unique within the schema.

trigger_time: It is the trigger action time, which should be either BEFORE or AFTER. It is the required parameter while defining a trigger. It indicates that the trigger will be invoked before or after each row modification occurs on the table.

trigger_event: It is the type of operation name that activates the trigger. It can be either INSERT, UPDATE, or DELETE operation. The trigger can invoke only one event at one time. If we want to define a trigger which is invoked by multiple events, it is required to define multiple triggers, and one for each event.

table_name: It is the name of the table to which the trigger is associated. It must be written after the ON keyword. If we did not specify the table name, a trigger would not exist.

BEGIN END Block: Finally, we will specify the statement for execution when the trigger is activated. If we want to execute multiple statements, we will use the BEGIN END block that contains a set of queries to define the logic for the trigger.

The trigger body can access the column's values, which are affected by the DML statement. The **NEW** and **OLD** modifiers are used to distinguish the column values **BEFORE** and **AFTER** the execution of the DML statement. We can use the column name with NEW and OLD modifiers as **OLD.col_name** and **NEW.col_name**. The

OLD.column_name indicates the column of an existing row before the updation or deletion occurs. NEW.col_name indicates the column of a new row that will be inserted or an existing row after it is updated.

For example, suppose we want to update the column name **message_info** using the trigger. In the trigger body, we can access the column value before the update as **OLD.message_info** and the new value **NEW.message_info**.

We can understand the availability of OLD and NEW modifiers with the below table:

Trigger Event	OLD	NEW
INSERT	No	Yes
UPDATE	Yes	Yes
ELETE	Yes	No

MySQL Trigger Example

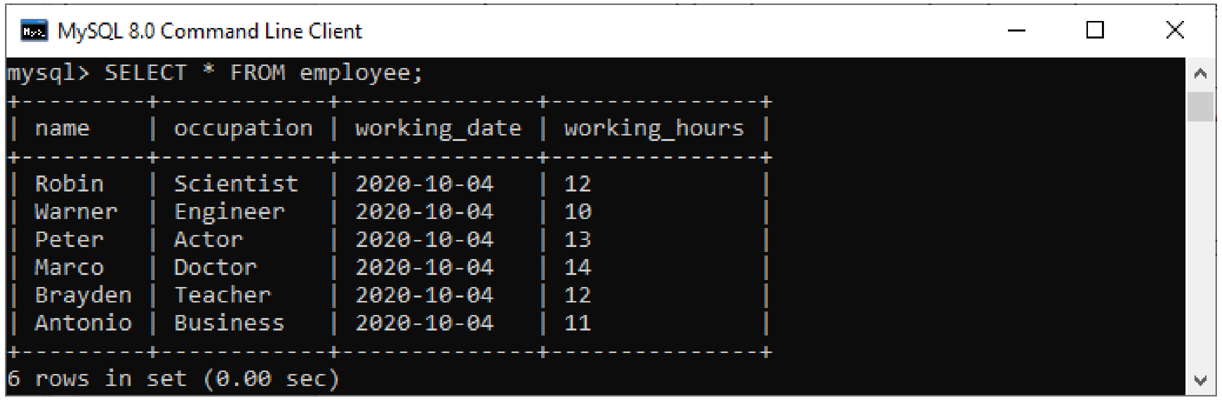
Let us start creating a trigger in MySQL that makes modifications in the employee table. First, we will create a new table named **employee** by executing the below statement:

```
CREATE TABLE employee(
    name varchar(45) NOT NULL,
    occupation varchar(35) NOT NULL,
    working_date date,
    working_hours varchar(10)
);
```

Next, execute the below statement to **fill the records** into the employee table:

```
INSERT INTO employee VALUES
('Robin', 'Scientist', '2020-10-04', 12),
('Warner', 'Engineer', '2020-10-04', 10),
('Peter', 'Actor', '2020-10-04', 13),
('Marco', 'Doctor', '2020-10-04', 14),
('Brayden', 'Teacher', '2020-10-04', 12),
('Antonio', 'Business', '2020-10-04', 11);
```

Next, execute the **SELECT statement** to verify the inserted record:

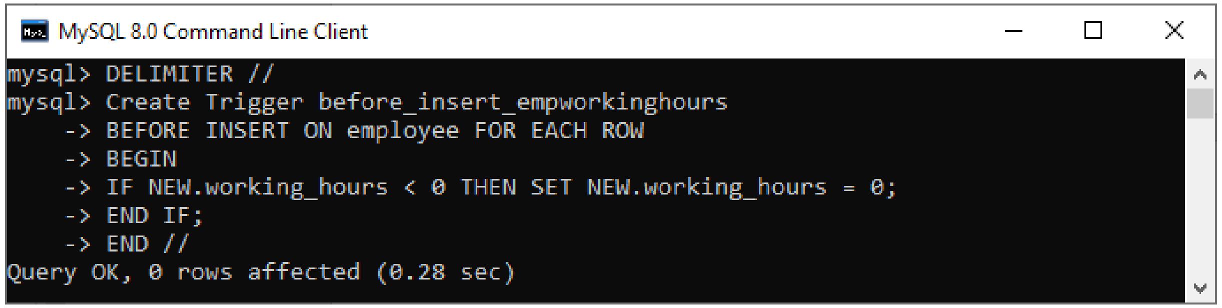

MySQL 8.0 Command Line Client

```
mysql> SELECT * FROM employee;
+---------+------------+--------------+---------------+
| name    | occupation | working_date | working_hours |
+---------+------------+--------------+---------------+
| Robin   | Scientist  | 2020-10-04   | 12            |
| Warner  | Engineer   | 2020-10-04   | 10            |
| Peter   | Actor      | 2020-10-04   | 13            |
| Marco   | Doctor     | 2020-10-04   | 14            |
| Brayden | Teacher    | 2020-10-04   | 12            |
| Antonio | Business   | 2020-10-04   | 11            |
+---------+------------+--------------+---------------+
6 rows in set (0.00 sec)
```

Next, we will create a **BEFORE INSERT trigger**. This trigger is invoked automatically insert the **working_hours = 0** if someone tries to insert **working_hours < 0**.

```
mysql> DELIMITER //
mysql> Create Trigger before_insert_empworkinghours
BEFORE INSERT ON employee FOR EACH ROW
BEGIN
IF NEW.working_hours < 0 THEN SET NEW.working_hours = 0;
END IF;
END //
```

If the trigger is created successfully, we will get the output as follows:



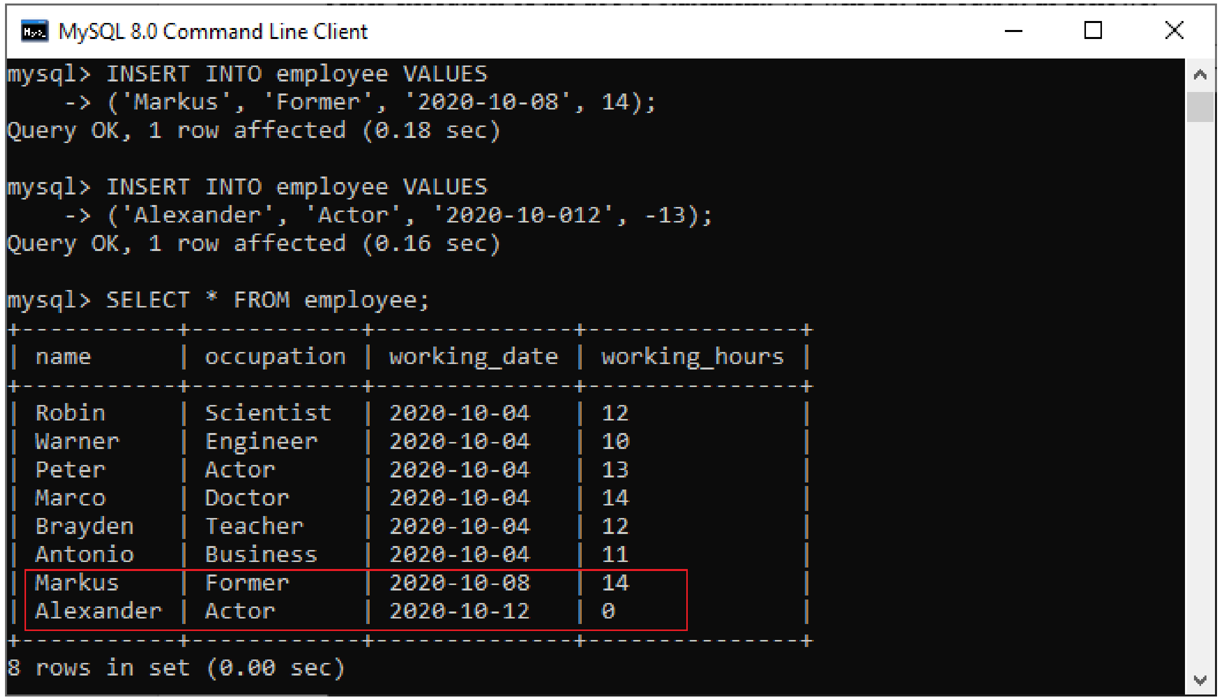
```
mysql> DELIMITER //
mysql> Create Trigger before_insert_empworkinghours
    -> BEFORE INSERT ON employee FOR EACH ROW
    -> BEGIN
    -> IF NEW.working_hours < 0 THEN SET NEW.working_hours = 0;
    -> END IF;
    -> END //
Query OK, 0 rows affected (0.28 sec)
```

Now, we can use the following statements to invoke this trigger:

```
mysql> INSERT INTO employee VALUES
('Markus', 'Former', '2020-10-08', 14);

mysql> INSERT INTO employee VALUES
('Alexander', 'Actor', '2020-10-012', -13);
```

After execution of the above statement, we will get the output as follows:



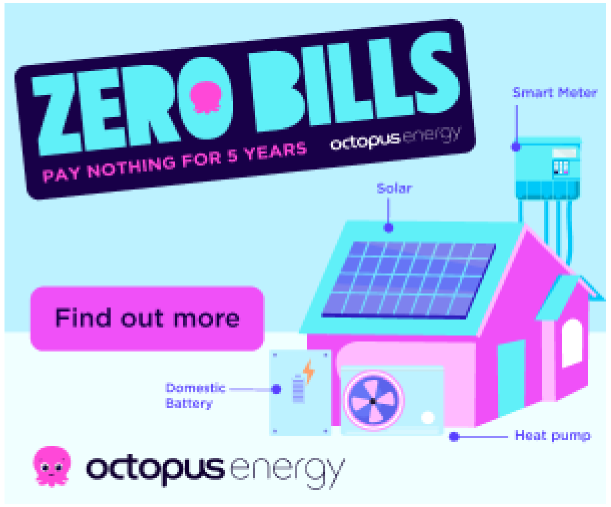
```
mysql> INSERT INTO employee VALUES
    -> ('Markus', 'Former', '2020-10-08', 14);
Query OK, 1 row affected (0.18 sec)

mysql> INSERT INTO employee VALUES
    -> ('Alexander', 'Actor', '2020-10-012', -13);
Query OK, 1 row affected (0.16 sec)

mysql> SELECT * FROM employee;
+-----------+------------+--------------+---------------+
| name      | occupation | working_date | working_hours |
+-----------+------------+--------------+---------------+
| Robin     | Scientist  | 2020-10-04   | 12            |
| Warner    | Engineer   | 2020-10-04   | 10            |
| Peter     | Actor      | 2020-10-04   | 13            |
| Marco     | Doctor     | 2020-10-04   | 14            |
| Brayden   | Teacher    | 2020-10-04   | 12            |
| Antonio   | Business   | 2020-10-04   | 11            |
| Markus    | Former     | 2020-10-08   | 14            |
| Alexander | Actor      | 2020-10-12   | 0             |
+-----------+------------+--------------+---------------+
8 rows in set (0.00 sec)
```

In this output, we can see that on inserting the negative values into the working_hours column of the table will automatically fill the zero value by a trigger.

← Prev Next →



Youtube For Videos Join Our Youtube Channel: Join Now

Feedback

- Send your Feedback to feedback@javatpoint.com

Help Others, Please Share

Learn Latest Tutorials

Splunk tutorial

Splunk

SPSS tutorial

SPSS

Swagger tutorial

Swagger

T-SQL tutorial

Transact-SQL

Tumblr tutorial

Tumblr

React tutorial

ReactJS

Regex tutorial

Regex

Reinforcement learning tutorial

R Programming tutorial

Reinforcement Learning

R Programming

RxJS tutorial

RxJS

React Native tutorial

React Native

Python Design Patterns

Python Design Patterns

Python Pillow tutorial

Python Pillow

Python Turtle tutorial

Python Turtle

Keras tutorial

Keras

Preparation

Aptitude

Aptitude

Logical Reasoning

Reasoning

Verbal Ability

Verbal Ability

Interview Questions

Interview Questions

Company Interview Questions

Company Questions

Trending Technologies

Artificial Intelligence Tutorial Artificial Intelligence	AWS Tutorial AWS	Selenium tutorial Selenium
Cloud Computing tutorial Cloud Computing	Hadoop tutorial Hadoop	ReactJS Tutorial ReactJS
Data Science Tutorial Data Science	Angular 7 Tutorial Angular 7	Blockchain Tutorial Blockchain
Git Tutorial Git	Machine Learning Tutorial Machine Learning	DevOps Tutorial DevOps

B.Tech / MCA

DBMS tutorial

DBMS

Data Structures tutorial

Data Structures

DAA tutorial

DAA

Operating System tutorial

Operating System

Computer Network tutorial

Computer Network

Compiler Design tutorial

Compiler Design

Computer Organization and Architecture

Computer Organization

Computer Graphics Tutorial

Computer Graphics

Cyber Security tutorial

Cyber Security

Automata Tutorial

Automata

C Language tutorial

C Programming

C++ tutorial

C++

Java tutorial

Java

.Net Framework tutorial

.Net

Python tutorial

Python

List of Programs

Programs

Control Systems tutorial

Control System

Data Mining Tutorial

Data Mining

Data Warehouse Tutorial

Data Warehouse