LAB-8 MySQL Trigger

A trigger in MySQL is a set of SQL statements that reside in a system catalog. **It is a special type of stored procedure that is invoked automatically in response to an event**. Each trigger is associated with a table, which is activated on any DML statement such as **INSERT, UPDATE**, or **DELETE**.

A trigger is called a special procedure because it cannot be called directly like a stored procedure. The main difference between the trigger and procedure is that a trigger is called automatically when a data modification event is made against a table. In contrast, a stored procedure must be called explicitly.

Generally, **triggers are of two types** according to the SQL standard: row-level triggers and statement-level triggers.

Row-Level Trigger: It is a trigger, which is activated for each row by a triggering statement such as insert, update, or delete. For example, if a table has inserted, updated, or deleted multiple rows, the row trigger is fired automatically for each row affected by the insert, update, or delete statement.

Statement-Level Trigger: It is a trigger, which is fired once for each event that occurs on a table regardless of how many rows are inserted, updated, or deleted.

Why we need/use triggers in MySQL?

We need/use triggers in MySQL due to the following features:

- Triggers help us to enforce business rules.
- Triggers help us to validate data even before they are inserted or updated.
- Triggers help us to keep a log of records like maintaining audit trails in tables.
- SQL triggers provide an alternative way to check the integrity of data.
- Triggers provide an alternative way to run the scheduled task.
- Triggers increases the performance of SQL queries because it does not need to compile each time the query is executed.
- Triggers reduce the client-side code that saves time and effort.
- Triggers help us to scale our application across different platforms.

Limitations of Using Triggers in MySQL

- MySQL triggers do not allow to use of all validations; they only provide extended validations. **For example**, we can use the NOT NULL, UNIQUE, CHECK and FOREIGN KEY constraints for simple validations.
- Triggers are invoked and executed invisibly from the client application. Therefore, it isn't easy to troubleshoot what happens in the database layer.
- Triggers may increase the overhead of the database server.

Types of Triggers in MySQL?

We can define the maximum six types of actions or events in the form of triggers:

1. **Before Insert:** It is activated before the insertion of data into the table.
2. **After Insert:** It is activated after the insertion of data into the table.
3. **Before Update:** It is activated before the update of data in the table.
4. **After Update:** It is activated after the update of the data in the table.
5. **Before Delete:** It is activated before the data is removed from the table.
6. **After Delete:** It is activated after the deletion of data from the table.

When we use a statement that does not use INSERT, UPDATE or DELETE query to change the data in a table, the triggers associated with the trigger will not be invoked.

An example of trigger in mysql is given below:

QUERIES

CREATE TABLE account (acct_num INT, amount DECIMAL(10,2));

```
mysql> CREATE TABLE account (acct_num INT, amount DECIMAL(10,2));
Query OK, 0 rows affected (0.08 sec)
```

CREATE TRIGGER ins_sum BEFORE INSERT ON account
FOR EACH ROW
SET @sum = @sum + NEW.amount;

```
mysql> CREATE TRIGGER ins_sum BEFORE INSERT ON account        FOR EACH ROW SET @
sum = @sum + NEW.amount;
Query OK, 0 rows affected (0.02 sec)
```

mysql> SET @sum = 0;
mysql> INSERT INTO account VALUES(137,154.98),(141,1937.50),(97,-160.00);
mysql> SELECT @sum AS 'Total amount inserted';

```
mysql>  SET @sum = 0;
Query OK, 0 rows affected (0.00 sec)

mysql> INSERT INTO account VALUES(137,154.98),(141,1937.50),(97,-160.00);
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> SELECT @sum AS 'Total amount inserted';
+----------------------+
| Total amount inserted |
+----------------------+
|               1932.48 |
+----------------------+
1 row in set (0.00 sec)
```

DROP TRIGGER test.ins_sum;

```
mysql> DROP TRIGGER LAB.ins_sum;
Query OK, 0 rows affected (0.02 sec)
```