Assignment - 8

Date of Completion: 05/10/2020

Date of Submission: 12/10/2020

TITLE:
Database trigger (all types, row level and statement level triggers, before & after triggers)

PROBLEM STATEMENT
Write a database trigger on Student table the system should keep track of the records that are being updated or deleted. The old value of updated or deleted records should be added in alumni table
Student (Rollno, Name, Date of Admission, branch, percent, status)

OBJECTIVE:
- To understand row level & statement level triggers.
- To understand before & after triggers.

OUTCOME:
Students will be able to:
- write triggers before & after update/delete statements on a table.

- write row level & statement level triggers.
- S/W & H/W Requirements.
  MYSQL, window/linux, i5 procesor
- THEORY:

My SQL Triggers
Triggers are a set of actions run independently & automatically when a specified change operation is performed

Syntax:
```
create triggrer tname
    trigger_time trigger-event
        on tab-name for each row
        begin
            ...
        end;
```

trigger time: before/after
trigger event: insert/update/delete

Executed in response to:
- DML statements
- DDL statements.
- a database operation (LOGON, LOGOFF)

Before & After Trigger

- before triggers run the trigger action befor the triggering statement is run.
- after triggers run the trigger action after the triggering statement is run.

Row level trigger.

The trigger is fired everytime some content in a row is changed.

Statement level trigger

The trigger is fired everytime that specific triggering statement is called.

Examples:

1) before insert for each row
2) after delete for each statement
3) before update for each row

Conclusion:

Thus we implemented

1) all types of triggers in a database.
2) row-level & statement level triggers
3) befor & after trigger.

```
mysql> DROP DATABASE if EXISTS A8;
Query OK, 1 row affected (0.01 sec)

mysql> CREATE DATABASE A8;
Query OK, 1 row affected (0.01 sec)

mysql> USE A8;
mysql>
mysql> CREATE TABLE student(
    ->     roll_no INT,
    ->     name VARCHAR(20),
    ->     doa date,
    ->     branch VARCHAR(20),
    ->     percent INT,
    ->     status VARCHAR(20)
    ->     );
Query OK, 0 rows affected (0.02 sec)

mysql>
mysql> CREATE TABLE alumni(
    ->     roll_no INT,
    ->     name VARCHAR(20),
    ->     doa date,
    ->     branch VARCHAR(20),
    ->     percent INT,
    ->     status VARCHAR(20)
    ->     );

mysql>
mysql> INSERT INTO student VALUES(1, 'Prathamesh', '2020-04-8', 'Comp', '89', 'Pass');
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO student VALUES(2, 'Aditya', '2020-04-5', 'ENTC', '45', 'Fail');
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO student VALUES(3, 'Utkarsh', '2020-04-3', 'IT', '81', 'Pass');
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO student VALUES(4, 'Varun', '2020-03-8', 'IT', '91', 'Pass');
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO student VALUES(5, 'Shreya', '2020-01-23', 'Comp', '94', 'Pass');
Query OK, 1 row affected (0.00 sec)

mysql>
mysql>
mysql> CREATE TRIGGER student_delete
    ->     BEFORE DELETE ON student
```

```
    ->    FOR EACH ROW
    ->    INSERT INTO alumni VALUES(
    ->    old.roll_no,
    ->    old.name,
    ->    old.doa,
    ->    old.branch,
    ->    old.percent,
    ->    old.status
    ->    );
Query OK, 0 rows affected (0.00 sec)

mysql>
mysql> DELIMITER $$
mysql> CREATE TRIGGER student_update
    ->    BEFORE UPDATE ON student
    ->    FOR EACH ROW
    ->    BEGIN
    ->    INSERT INTO alumni VALUES(
    ->    old.roll_no,
    ->    old.name,
    ->    old.doa,
    ->    old.branch,
    ->    old.percent,
    ->    old.status
    ->    );
    ->    END$$
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> delimiter ;
mysql> UPDATE student SET percent = 81 WHERE roll_no=2;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from alumni;
+---------+--------+------------+--------+---------+--------+
| roll_no | name   | doa        | branch | percent | status |
+---------+--------+------------+--------+---------+--------+
|       2 | Aditya | 2020-04-05 | ENTC   |      45 | Fail   |
+---------+--------+------------+--------+---------+--------+
1 row in set (0.00 sec)

mysql> UPDATE student SET percent = 11 WHERE roll_no=2;
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from alumni;
+---------+--------+------------+--------+---------+--------+
| roll_no | name   | doa        | branch | percent | status |
```

```
+---------+--------+------------+--------+---------+--------+
|       2 | Aditya | 2020-04-05 | ENTC   |      45 | Fail   |
|       2 | Aditya | 2020-04-05 | ENTC   |      81 | Fail   |
+---------+--------+------------+--------+---------+--------+
2 rows in set (0.00 sec)

mysql> DELETE FROM student WHERE roll_no=1;
Query OK, 1 row affected (0.01 sec)

mysql> select * from alumni;
+---------+------------+------------+--------+---------+--------+
| roll_no | name       | doa        | branch | percent | status |
+---------+------------+------------+--------+---------+--------+
|       2 | Aditya     | 2020-04-05 | ENTC   |      45 | Fail   |
|       2 | Aditya     | 2020-04-05 | ENTC   |      81 | Fail   |
|       1 | Prathamesh | 2020-04-08 | Comp   |      89 | Pass   |
+---------+------------+------------+--------+---------+--------+
3 rows in set (0.00 sec)

mysql>

mysql> drop database if exists A5;
Query OK, 2 rows affected (0.03 sec)

mysql> create database A5;
Query OK, 1 row affected (0.00 sec)

mysql> use A5;
Database changed
mysql> DROP PROCEDURE IF EXISTS setFine;
Query OK, 0 rows affected, 1 warning (0.00 sec)

mysql>
mysql> create table Customer(
    ->              Cust_id int not null,
    ->              Name varchar(30),
    ->              DateOfPayment date,
    ->              NameOfScheme varchar(20),
    ->              Status varchar(10),
    ->              primary key(Cust_id)
    ->              );
Query OK, 0 rows affected (0.02 sec)

mysql>
mysql> create table Fine(
    ->              Cust_id int not null,
    ->              Date date,
    ->              Amt int,
    ->              foreign key(Cust_id) references Customer(Cust_id) on delete cascade
```

```
   ->                  );
Query OK, 0 rows affected (0.03 sec)

mysql>
mysql> insert into Customer VALUES(1, "Prathamesh", "2020-04-8", "High-return", "N" );
Query OK, 1 row affected (0.00 sec)

mysql> insert into Customer VALUES(2, "Aditya", "2020-03-15", "Low-return", "N" );
Query OK, 1 row affected (0.01 sec)

mysql> insert into Customer VALUES(3, "Sourav", "2020-03-12", "High-return", "N" );
Query OK, 1 row affected (0.00 sec)

mysql> insert into Customer VALUES(4, "Rajesh", "2020-03-1", "Low-return", "N" );
Query OK, 1 row affected (0.00 sec)

mysql> insert into Customer VALUES(5, "Suman", "2020-03-27", "Low-return", "N" );
Query OK, 1 row affected (0.01 sec)

mysql>
mysql> delimiter @@
mysql> select * from Customer@@
+---------+------------+---------------+--------------+--------+
| Cust_id | Name       | DateOfPayment | NameOfScheme | Status |
+---------+------------+---------------+--------------+--------+
|       1 | Prathamesh | 2020-04-08    | High-return  | N      |
|       2 | Aditya     | 2020-03-15    | Low-return   | N      |
|       3 | Sourav     | 2020-03-12    | High-return  | N      |
|       4 | Rajesh     | 2020-03-01    | Low-return   | N      |
|       5 | Suman      | 2020-03-27    | Low-return   | N      |
+---------+------------+---------------+--------------+--------+
5 rows in set (0.00 sec)

mysql>
mysql> CREATE TRIGGER fine_update
   -> BEFORE UPDATE ON Customer
   -> FOR EACH ROW
   -> BEGIN
   ->     declare myFine INT;
   ->     declare myDate date;
   ->     declare myStatus VARCHAR(10);
   ->     declare days int;
   ->     declare diff int;
   ->
   ->     select DateOfPayment into myDate FROM Customer where Cust_id = old.CUST_id;
   ->     SELECT Status into myStatus FROM Customer where Cust_id = old.CUST_id;
   ->     select DATEDIFF(CURDATE() , myDate) into diff;
   ->
   ->     IF myStatus="N" THEN
```

```
    ->         IF diff>15 AND diff<=30 THEN
    ->             set myFine = 5*diff;
    ->         END IF;
    ->         IF diff>30 THEN
    ->             set myFine = 50*(diff-30) + 75;
    ->         END IF;
    ->
    ->         INSERT INTO Fine VALUES(old.Cust_id, myDate, myFine);
    ->     END IF;
    -> END @@
Query OK, 0 rows affected (0.02 sec)

mysql> delimiter ;
mysql>
mysql> UPDATE Customer SET Status="P" WHERE Name="Prathamesh";
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from Fine;
+---------+------------+------+
| Cust_id | Date       | Amt  |
+---------+------------+------+
|       1 | 2020-04-08 | 7575 |
+---------+------------+------+
1 row in set (0.00 sec)

mysql>
```