Assignment No. - 8

Date:

Title: Study of all types of database trigger.

Problem Statement:
Write a database trigger on student table. The system should keep track of the records that are being updated or deleted. The odd value of updated or deleted record should be added in alumni table.

Objective:
- Understand concept of database trigger.
- Understand mysql commands.

Outcome:
Student will be able to implement and apply all types of database triggers.

Requirements:
Fedora 29, i5 processor, 8 GB ram, mysql package.

Theory:
Triggers: A trigger defines an action that the database should take when some database related event (such as insert, update or delete) occurs. Triggers are similar to procedures, in that they are name PL/SQL blocks.

Triggers are used when:

- Maintaining. complex integrity constraints (referential integrity) or buissness rules.
- Auditing information in a table by recording the changes.
- Automatically signalling other programs that action need's to take place when changes are made to a table.
- Collecting / maintaning stabilised data.

Types of Triggers:

- There are 2 types of trigger in oracle in row level triggers and statement level triggers.

- Row Level Triggers:

1. Row level triggers are for data related activities.
2. Row level triggers execute once for each row in atransaction.
3. It is identified by ~~ts~~ the for each row clause in the create trigger command.
4.

2. Statement Level Trigger:

1. Statement level triggers are for transactional related activities.
2. They're executed once for each transaction
3. It is identified by omitting the for each clause in create trigger command.

Managing MySQL Triggers:

1. Before insert trigger: It is automatically fired before an insert event occurs on the table.

2. After insert trigger: It is automatically invoked after an ~~ine~~ insert event occurs on the table.

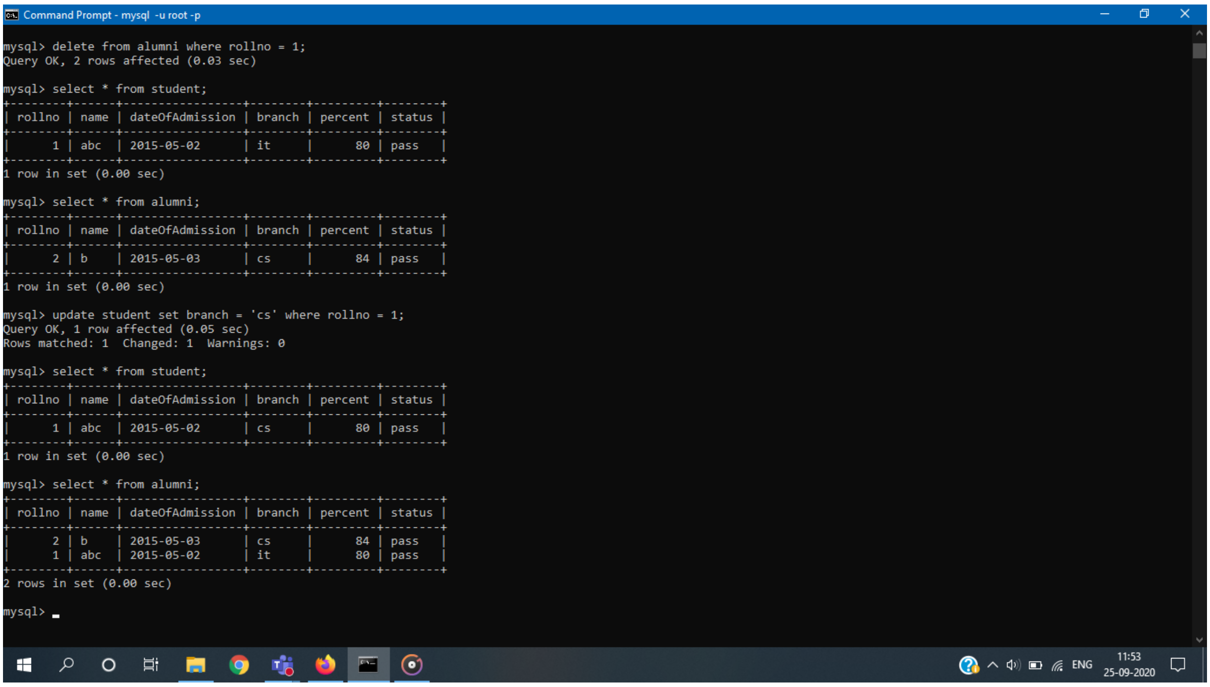
3. Befor and After Update trigger: These are automatically invoked before/after update. event occurs on the table.

4. Before and after Delete trigger: These are automatically invoked before/after delete event occurs on the table.

Conclusion:

We have understood and implemented successfully all types of database triggers in MySQL.

Output



```
mysql> delete from alumni where rollno = 1;
Query OK, 2 rows affected (0.03 sec)

mysql> select * from student;
+--------+------+-----------------+--------+---------+--------+
| rollno | name | dateOfAdmission | branch | percent | status |
+--------+------+-----------------+--------+---------+--------+
|      1 | abc  | 2015-05-02      | it     |      80 | pass   |
+--------+------+-----------------+--------+---------+--------+
1 row in set (0.00 sec)

mysql> select * from alumni;
+--------+------+-----------------+--------+---------+--------+
| rollno | name | dateOfAdmission | branch | percent | status |
+--------+------+-----------------+--------+---------+--------+
|      2 | b    | 2015-05-03      | cs     |      84 | pass   |
+--------+------+-----------------+--------+---------+--------+
1 row in set (0.00 sec)

mysql> update student set branch = 'cs' where rollno = 1;
Query OK, 1 row affected (0.05 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from student;
+--------+------+-----------------+--------+---------+--------+
| rollno | name | dateOfAdmission | branch | percent | status |
+--------+------+-----------------+--------+---------+--------+
|      1 | abc  | 2015-05-02      | cs     |      80 | pass   |
+--------+------+-----------------+--------+---------+--------+
1 row in set (0.00 sec)

mysql> select * from alumni;
+--------+------+-----------------+--------+---------+--------+
| rollno | name | dateOfAdmission | branch | percent | status |
+--------+------+-----------------+--------+---------+--------+
|      2 | b    | 2015-05-03      | cs     |      84 | pass   |
|      1 | abc  | 2015-05-02      | it     |      80 | pass   |
+--------+------+-----------------+--------+---------+--------+
2 rows in set (0.00 sec)

mysql> _
```