Lab 6. Trigger (Test Questions)

Note: You must make every effort to store all your queries and results in a text file. During evaluation, it creates problems. This time it would penalize some marks.

Basic Mysql queries:

Update:

```
UPDATE TABLE_NAME
SET
ATTRIBUTE_NAME = “new value”
WHERE
CONDITION;
```

Delete:

```
DELETE FROM TABLE_NAME
WHERE
CONDITION;
```

Q1) Create a customer audit table such that it is updated for each update in customer table.

Steps: Create a customer_audit table with adequate attribute list. Write the code for trigger. Update the customer table. See whether the customer table has been updated. Now, see the customer_audit table that you have created. It must be updated because of the trigger has already fired.

Q2) Create an account audit table such that it concatenates "update record-acc_no deleted' for each delete in accounts table.

Step: Create a table named Account_audit with adequate attribute list. Create a trigger to be fired after account deletion. See whether account table has been changed after deletion (which it must have). Most importantly, see whether Account_audit has been changed after deletion.

Q3) Use the following query to create a student_marks table and to update it with subject marks. For each update output the grade.

```
CREATE TABLE STUDENT_MARKS(
ID INT AUTO INCREMENT,
NAME VARCHAR(20),
SUB1 INT,
SUB2 INT,
SUB3 INT,
SUB4 INT,
SUB5 INT,
TOTAL INT,
PER_MARKS INT,
GRADE CHAR(2)
);


UPDATE STUDENT_MARKS
SET
SUB1 = 54,
SUB2 = 69,
SUB3 = 89,
SUB4 = 87,
SUB5 = 59
WHERE
STUDENT_ID = 1;
```

Total Marks: TOTAL = SUB1 + SUB2 + SUB3 + SUB4 + SUB5

Percentage of Marks : PER_MARKS = (TOTAL)/5

Grade (will be stored GRADE column):

- If PER_MARKS>=90 -> 'EXCELLENT'

- If PER_MARKS>=75 AND PER_MARKS<90 -> 'VERY GOOD'

- If PER_MARKS>=60 AND PER_MARKS<75 -> 'GOOD'

- If PER_MARKS>=40 AND PER_MARKS<60 -> 'AVERAGE'

- If PER_MARKS<40-> 'NOT PROMOTED'

Step: Do as asked in the question first. Create the correct trigger. To see whether the STUDENT_MARKS table has been changed, use `SELECT * FROM STUDENT_MARKS;`

Q4) Write a trigger such that for each update in table1

- **The values are copied in table2**
- **Those values are deleted from table3**
- **Frequency of those values are stored in second column of table4**

```
CREATE TABLE table1(a1 INT);
CREATE TABLE table2(a2 INT);
CREATE TABLE table3(
a3 INT NOT NULL AUTO_INCREMENT PRIMARY KEY
);

CREATE TABLE table4(
  a4 INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
  b4 INT DEFAULT 0
);
```

<u>Step:</u> Do as asked in the question. Then create the trigger.

To check use:

```
INSERT INTO table3 (a3) VALUES
  (NULL), (NULL), (NULL), (NULL), (NULL),
  (NULL), (NULL), (NULL), (NULL), (NULL);


INSERT INTO table4 (a4) VALUES
  (0), (0), (0), (0), (0), (0), (0), (0), (0), (0);


INSERT INTO test1 VALUES
       (1), (3), (1), (7), (1), (8), (4), (4);


Output - mysql> SELECT * FROM table1;
+------+
| a1   |
+------+
|    1 |
|    3 |
|    1 |
|    7 |
|    1 |
|    8 |
|    4 |
|    4 |
+------+
8 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM table2;
+-------+
| a2    |
+-------+
|     1 |
|     3 |
|     1 |
|     7 |
|     1 |
|     8 |
|     4 |
|     4 |
+-------+
8 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM table3;
+----+
| a3 |
+----+
|  2 |
|  5 |
|  6 |
|  9 |
| 10 |
+----+
5 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM table4;
+----+-------+
| a4 | b4    |
+----+-------+
|  1 |     3 |
|  2 |     0 |
|  3 |     1 |
|  4 |     2 |
|  5 |     0 |
|  6 |     0 |
|  7 |     1 |
|  8 |     1 |
|  9 |     0 |
| 10 |     0 |
+----+-------+
10 rows in set (0.00 sec)
```

Don’t hard-code guys. It may get dangerous. Just to show the results.

Q5) Create the following two tables:

```
ext_words
-------------
| id | word |
-------------
| 1  | this |
-------------
| 2  | that |
-------------
| 3  | this |
-------------
```

```
ext_words_count
---------------------
| id | word | count |
---------------------
| 1  | this |   2   |
---------------------
| 2  | that |   1   |
---------------------
```

Create a trigger which updates count when word is updated in ext_words and if ext_words.word does not exist in ext_words_count when ext_words is updated,insert the word into ext_words_count and set count as 1.

To check use - `insert into ext_words values ('there');`