Assignment-7

Lab Practice:-

```
mysql> create trigger after_emp_insert
    -> after insert on emp
    -> for each row
    -> begin
    -> update deptsal
    -> set deptno=new.deptno,
    -> totalsalary=totalsalary+new.salary;
    -> end
    -> //
Query OK, 0 rows affected (0.01 sec)

mysql> select * from deptsal//
+--------+-------------+
| deptno | totalsalary |
+--------+-------------+
|      1 |         400 |
|     10 |         900 |
|     20 |        1800 |
|     30 |        1400 |
|     40 |         800 |
|     30 |         400 |
+--------+-------------+
6 rows in set (0.00 sec)

mysql> create trigger after_emp_insert
    -> after insert on emp
    -> for each row
    -> begin
    -> insert into deptsal
```

```
    -> set deptno=new.deptno,
    -> totalsalary=totalsalary+new.salary;
    -> end
    -> //
Query OK, 0 rows affected (0.01 sec)

mysql> insert into emp
values(7478,'Kavya','Employee',7371,'2021-09-06',400,50,30)//
Query OK, 1 row affected (0.00 sec)

mysql> select * from deptsal//
+--------+-------------+
| deptno | totalsalary |
+--------+-------------+
|      1 |         400 |
|     10 |         900 |
|     20 |        1800 |
|     30 |        1400 |
|     40 |         800 |
|     30 |         400 |
+--------+-------------+
6 rows in set (0.00 sec)
```

Batch 1 Exercise 1

Consider following three table and create trigger for below exercise:

- Highschooler(ID int, name text, grade int);
- Friend(ID1 int, ID2 int);
- Likes(ID1 int, ID2 int);

```
mysql> CREATE TABLE Highschooler (
    ->     ID INT PRIMARY KEY,
    ->     name TEXT,
    ->     grade INT
    -> )
    -> ;
Query OK, 0 rows affected (0.04 sec)

mysql> CREATE TABLE Friend (
    ->     ID1 INT,
    ->     ID2 INT,
    ->     PRIMARY KEY (ID1, ID2),
    ->     FOREIGN KEY (ID1) REFERENCES Highschooler(ID),
    ->     FOREIGN KEY (ID2) REFERENCES Highschooler(ID)
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> CREATE TABLE Likes (
    ->     ID1 INT,
    ->     ID2 INT,
    ->     PRIMARY KEY (ID1, ID2),
    ->     FOREIGN KEY (ID1) REFERENCES Highschooler(ID),
    ->     FOREIGN KEY (ID2) REFERENCES Highschooler(ID)
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> desc Highschooler;
+-------+------+------+-----+---------+-------+
| Field | Type | Null | Key | Default | Extra |
+-------+------+------+-----+---------+-------+
| ID    | int  | NO   | PRI | NULL    |       |
| name  | text | YES  |     | NULL    |       |
```

```
| grade | int  | YES  |     | NULL    |       |
+-------+------+------+-----+---------+-------+
3 rows in set (0.01 sec)

mysql> desc Likes;
+-------+------+------+-----+---------+-------+
| Field | Type | Null | Key | Default | Extra |
+-------+------+------+-----+---------+-------+
| ID1   | int  | NO   | PRI | NULL    |       |
| ID2   | int  | NO   | PRI | NULL    |       |
+-------+------+------+-----+---------+-------+
2 rows in set (0.00 sec)

mysql> desc Friend;
+-------+------+------+-----+---------+-------+
| Field | Type | Null | Key | Default | Extra |
+-------+------+------+-----+---------+-------+
| ID1   | int  | NO   | PRI | NULL    |       |
| ID2   | int  | NO   | PRI | NULL    |       |
+-------+------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

Write one or more triggers to maintain symmetry in friend relationships. Specifically, if (A,B) is deleted from Friend, then (B,A) should be deleted too. If (A,B) is inserted into Friend then (B,A) should be inserted too. Don't worry about updates to the Friend table.

```
mysql> Delimiter //
mysql> //
ERROR:
No query specified
```

```
mysql> insert into Highschooler values(1,Alice,10),
    -> (2, 'Bob', 11),
    -> (3, 'Charlie', 9),
    -> (4, 'David', 10),
    -> (5, 'Emma', 11)//
Query OK, 5 rows affected (0.01 sec)
Records: 5  Duplicates: 0  Warnings: 0

mysql> select * from Highschooler//
+----+---------+-------+
| ID | name    | grade |
+----+---------+-------+
|  1 | Alice   |    10 |
|  2 | Bob     |    11 |
|  3 | Charlie |     9 |
|  4 | David   |    10 |
|  5 | Emma    |    11 |
+----+---------+-------+
5 rows in set (0.00 sec)

mysql> CREATE TRIGGER friend_symmetry BEFORE INSERT ON Friend
    -> FOR EACH ROW
    -> BEGIN
    ->     INSERT INTO Friend (ID1, ID2)
    ->     VALUES (NEW.ID2, NEW.ID1);
    -> END;
    -> //
Query OK, 0 rows affected (0.01 sec)

mysql> CREATE TRIGGER friend_symmetry_delete BEFORE DELETE
ON Friend
    -> FOR EACH ROW
```

```
    -> BEGIN
    ->     DELETE FROM Friend WHERE ID1 = OLD.ID2 AND ID2 =
OLD.ID1;
    -> END;
    -> //
Query OK, 0 rows affected (0.01 sec)
```