DBMS – LAB -10

NAME : RAHUL VARMA

ROLL NO: S20200010212

SECTION: C

TASK: (LAB EXERCISES)

SQL COMMANDS:

Topic: MYSQL CURSOR

AND ERROR HANDLING

1. Consider a table with the schema Distances (fromCity, toCity, distance) that stores distances between cities. Since the distance from Y to X is always the same as the distance from X to Y, it would be redundant to store them both. How can you guarantee that the table never stores the distance from Y to X if it already has the distance from X to Y? Show the exact CREATE TABLE statement and/or trigger.

```
mysql> use lab10;
Database changed
mysql> create table distance(fromCity varchar(20),toCity varchar(20), distance int);
Query OK, 0 rows affected (0.05 sec)

mysql> insert into distance values('vijayawada','vizag',400);
Query OK, 1 row affected (0.02 sec)

mysql> insert into distance values('chittor','vizag',800);
Query OK, 1 row affected (0.01 sec)
```

```
mysql> delimiter //
mysql> create trigger Q1
    -> BEFORE INSERT
    -> ON Distance FOR EACH ROW
    -> BEGIN
    -> declare fromCity1, toCity1 varchar(30);
    -> declare dist, t1, t2 INT;
    -> set fromCity1 = new.fromCity;
    -> set tocity1 = new.tocity;
    -> set dist = new.distance;
    -> select count(*) into t1 from Distance where fromCity=fromCity1 and toCity=toCity1;
    -> select count(*) into t2 from Distance where fromCity=tocity1 and toCity=fromCity1;
    -> if t1>0 or t2> 0 then
    -> SIGNAL SQLSTATE '02000' SET MESSAGE_TEXT = "Distance between these two cities already exist";
    -> END IF;
    -> END //
Query OK, 0 rows affected (0.03 sec)
```

```
mysql> DELIMITER ;
mysql> insert into distance values('vizag','vijayawada',390);
ERROR 1643 (02000): Distance between these two cities already exist
mysql> insert into distance values('vizag','tata nagar',800);
Query OK, 1 row affected (0.01 sec)

mysql>
```

2. Consider a table with the schema BankCustomers (accNum, name and loan). Raise anexception when the customer initiates loan amount above 10lakhs

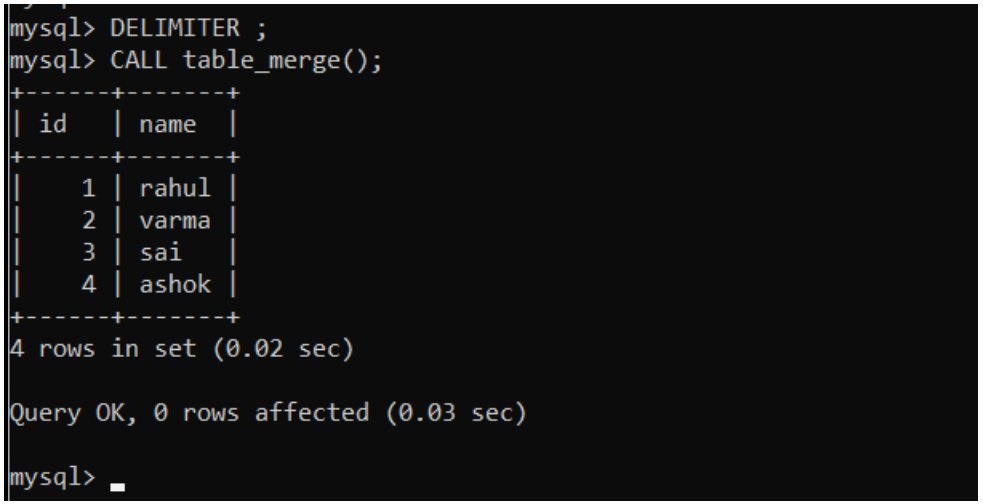
```
mysql> create table BankCustomers(accnum varchar(20), name varchar(20), loan int);
Query OK, 0 rows affected (0.06 sec)

mysql> delimiter //
mysql> CREATE PROCEDURE customer_insert(in accnum varchar(20), in name varchar(20), in loan int)
    -> deterministic
    -> begin
    -> if loan > 1000000 then
    -> select 'SORRY Max limit is 10 Lakhs only, enter loan less than 10 lakhs' as error;
    -> else
    -> insert into bankcustomers values(accnum,name,loan);
    -> end if;
    -> end //
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> delimiter ;
mysql> insert into bankcustomers values('101', 'Rahul Varma', 1100000);
Query OK, 1 row affected (0.03 sec)

mysql> call customer_insert('101','Rahul Varma', 1100000);
+-------------------------------------------------------------------+
| error                                                             |
+-------------------------------------------------------------------+
| SORRY Max limit is 10 Lakhs only, enter loan less than 10 lakhs |
+-------------------------------------------------------------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.01 sec)

mysql>
```

3. Write a PL/SQL block of code using parameterized Cursor that will merge the data available in the newly created table new_table with the data available in the table old_table. Note: If the data in the first table already exist in the second table then that data should be skipped.

```
mysql> create table old_table(id int,name varchar(30));
Query OK, 0 rows affected (0.05 sec)

mysql> create table new_table(id int,name varchar(30));
Query OK, 0 rows affected (0.06 sec)

mysql> insert into old_table values(1,'rahul');
Query OK, 1 row affected (0.02 sec)

mysql> insert into old_table values(2,'varma');
Query OK, 1 row affected (0.01 sec)

mysql> insert into new_table values(3,'sai');
Query OK, 1 row affected (0.01 sec)

mysql> insert into new_table values(4,'ashok');
Query OK, 1 row affected (0.01 sec)
```

```
mysql> DELIMITER //
mysql> create procedure table_merge()
    -> begin
    -> declare done int default 0;
    -> declare nt_name varchar(10);
    -> declare nt_id int;
    -> declare cur_merge cursor for
    -> select id,name from new_table;
    -> declare continue handler for not found set done = 1;
    -> open cur_merge;
    -> getdata: loop
    -> fetch cur_merge into nt_id,nt_name;
    -> IF done = 1 then
    -> leave getdata;
    -> END IF;
    -> insert into old_table values(nt_id,nt_name);
    -> end loop;
    -> select * from old_table;
    -> close cur_merge;
    -> end //
Query OK, 0 rows affected (0.02 sec)
```


mysql> DELIMITER ;
mysql> CALL table_merge();
+------+-------+
| id | name |
+------+-------+
1	rahul
2	varma
3	sai
4	ashok
+------+-------+
4 rows in set (0.02 sec)
Query OK, 0 rows affected (0.03 sec)
mysql>

4. Write a program in CURSOR to create a cursor displays the name and salary of each employee in the EMPLOYEES table whose salary is less than that specified by a passed-in parameter value.

```
mysql> create table employees(name varchar(30), age int, salary int(10));
Query OK, 0 rows affected, 1 warning (0.07 sec)

mysql> insert into employees values('Rahul Varma', 19, 20000);
Query OK, 1 row affected (0.02 sec)

mysql> insert into employees values('ashok', 31, 500000);
Query OK, 1 row affected (0.01 sec)

mysql> insert into employees values('abhiram', 47, 15000);
Query OK, 1 row affected (0.01 sec)
```

```
mysql> delimiter //
mysql> create procedure FindEmployee(in input INT)
    -> begin
    -> declare done INT DEFAULT FALSE;
    -> declare name1 varchar(40);
    -> declare salary1 INT;
    -> DECLARE cur1 CURSOR FOR SELECT name, salary FROM employees;
    -> DECLARE CONTINUE HANDLER FOR NOT FOUND SET done=TRUE;
    -> OPEN cur1;
    -> read_loop: LOOP
    -> FETCH cur1 INTO name1, salary1;
    -> IF done THEN
    -> LEAVE read_loop;
    -> END IF;
    -> if(salary1 < input) THEN
    -> SELECT name1 as Name, salary1 as Salary;
    -> END IF;
    -> END LOOP;
    -> CLOSE cur1;
    -> END //
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> delimiter ;
mysql> call FindEmployee(10000);
Query OK, 0 rows affected (0.00 sec)

mysql> call FindEmployee(25000);
+-------------+--------+
| Name        | Salary |
+-------------+--------+
| Rahul Varma |  20000 |
+-------------+--------+
1 row in set (0.00 sec)

+---------+--------+
| Name    | Salary |
+---------+--------+
| abhiram |  15000 |
+---------+--------+
1 row in set (0.01 sec)

Query OK, 0 rows affected (0.01 sec)
```

5. Create a cursor to increment the salary of employees based on experience
If experience>30 years, Increment of 30%
If experience is between 20-30 years, Increment of 20%
If experience is between 10-20 years, Increment of 10%.

```
mysql> create table employees(name varchar(30), experience int, salary int(10));
Query OK, 0 rows affected, 1 warning (0.05 sec)

mysql> insert into employees values('Rahul', 15, 10000);
Query OK, 1 row affected (0.03 sec)

mysql> insert into employees values('varma', 37, 100000);
Query OK, 1 row affected (0.01 sec)

mysql> insert into employees values('Sai', 25, 25000);
Query OK, 1 row affected (0.01 sec)
```

```
mysql> delimiter //
mysql> create procedure UpdateSalaryofEmployees()
    -> begin
    -> declare done INT DEFAULT FALSE;
    -> declare temp decimal(15,2);
    -> declare name1 varchar(40);
    -> declare salary1, exp1 INT;
    -> DECLARE cur1 CURSOR FOR SELECT name, experience, salary FROM employees;
    -> DECLARE CONTINUE HANDLER FOR NOT FOUND SET done=TRUE;
    -> OPEN cur1;
    -> read_loop: LOOP
    -> FETCH cur1 INTO name1, exp1, salary1;
    -> IF done then
    -> leave read_loop;
    -> END IF;
    -> set temp=salary1;
    -> IF exp1>30 then
    -> set temp = (1.3) *salary1;
    -> elseif exp1>20 and exp1<=30 then set temp=(1.2)*salary1;
    -> elseif exp1>10 and exp1<=20 then set temp=(1.1)*salary1;
    -> END IF;
    -> SELECT name1 as name, exp1 as experience, salary1 as salary, temp as incrementedSalary;
    -> END LOOP;
    -> CLOSE cur1;
    -> END //
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> delimiter ;
mysql> call UpdateSalaryofEmployees();
+-------+------------+--------+-------------------+
| name  | experience | salary | incrementedSalary |
+-------+------------+--------+-------------------+
| Rahul |         15 |  10000 |          11000.00 |
+-------+------------+--------+-------------------+
1 row in set (0.01 sec)

+-------+------------+--------+-------------------+
| name  | experience | salary | incrementedSalary |
+-------+------------+--------+-------------------+
| varma |         37 | 100000 |         130000.00 |
+-------+------------+--------+-------------------+
1 row in set (0.02 sec)

+------+------------+--------+-------------------+
| name | experience | salary | incrementedSalary |
+------+------------+--------+-------------------+
| Sai  |         25 |  25000 |          30000.00 |
+------+------------+--------+-------------------+
1 row in set (0.03 sec)
```

THANK YOU