```
/*******************************************************************************
***********************

Name : Pradip S Karmakar

Class : M.C.A 2

Roll_No : 10

Subject : RDBMS

********************************************************************************
***********************

Question : Creare a procedure for emp which will fetch all employees details and display on screen.

********************************************************************************
**********************/

// Procedure For Creating Tables

DELIMITER //
CREATE PROCEDURE create_table()
BEGIN
   create table emp(
      emp_id int primary key auto_increment,
      empname varchar(50),
      position varchar(50),
      salary decimal(8,2)
   );
END //


// Procedure For Inserting Data in Tables
```

```
CREATE PROCEDURE insert_data()
BEGIN
   insert into emp values(1,'Pradip','CEO_APPLE',80000),
   (2,'Ajinkya','CEO_GOOGLE','60000'),
   (3,'Nirav','CEO_MIRCOSOFT','50000'),
   (4,'Milind','CEO_SAMSUNG','60000'),
   (5,'Lakshya','CEO_FACEBOOK','90000');
END //
```

```
// Procedure With CURSORS for Display Data on the Screen

CREATE PROCEDURE cur_pro()
BEGIN
   DECLARE emp_id int;
   DECLARE emp_name varchar(50);
   DECLARE position varchar(50);
   DECLARE salary decimal(8,2);
   DECLARE c_finish integer DEFAULT 0;
   DECLARE curs cursor for select * from emp;
   DECLARE CONTINUE HANDLER for NOT FOUND set c_finish = 1;
   OPEN curs;
   get_line : LOOP
      FETCH curs into emp_id,emp_name,position,salary;
      IF c_finish = 1 THEN
         LEAVE get_line;
      END IF;
      SELECT CONCAT(emp_id,CONCAT(' | ',CONCAT(emp_name,CONCAT(' | 
',CONCAT(position,CONCAT(' | ',salary)))))) as Employee_Data;
   END LOOP get_line;
   CLOSE curs;
```

```
END //

DELIMITER ;

/*****************************************************************************************************

OUTPUT :

MariaDB [test]> call create_table();
Query OK, 0 rows affected (0.042 sec)

MariaDB [test]> call insert_data();
Query OK, 5 rows affected (0.022 sec)

MariaDB [test]> call cur_pro;
+-----------------------------------+
| Employee_Data                     |
+-----------------------------------+
| 1 | Pradip | CEO_APPLE | 80000.00 |
+-----------------------------------+
1 row in set (0.001 sec)

+-------------------------------------+
| Employee_Data                       |
+-------------------------------------+
| 2 | Ajinkya | CEO_GOOGLE | 60000.00 |
+-------------------------------------+
1 row in set (0.003 sec)

+--------------------------------------+
| Employee_Data                        |
+--------------------------------------+
| 3 | Nirav | CEO_MIRCOSOFT | 50000.00 |
+--------------------------------------+
1 row in set (0.005 sec)

+-------------------------------------+
| Employee_Data                       |
+-------------------------------------+
| 4 | Milind | CEO_SAMSUNG | 60000.00 |
+-------------------------------------+
1 row in set (0.007 sec)

+---------------------------------------+
| Employee_Data                         |
+---------------------------------------+
| 5 | Lakshya | CEO_FACEBOOK | 90000.00 |
+---------------------------------------+
1 row in set (0.011 sec)

Query OK, 0 rows affected (0.013 sec)


*****************************************************************************************
**********************/
```

==

Question 2 : Create a cursor to find list of all employees jn a deoartment passed as an argument from

the employee table.

==

```
DELIMITER //

MariaDB [test]> CREATE PROCEDURE create_tables()
    ->     BEGIN
    ->         create table department(
    ->             dept_id int primary key,
    ->             deptname varchar(30)
    ->         );
    ->
    ->         create table employee(
    ->             emp_id int primary key auto_increment,
    ->             empname varchar(30),
    ->             dept_id int,
    ->             designation varchar(20),
    ->             salary decimal(10,2),
    ->             FOREIGN KEY (dept_id) REFERENCES department(dept_id)
    ->         );
    ->
    ->     END //
Query OK, 0 rows affected (0.021 sec)

MariaDB [test]> call create_tables() //
```

```
Query OK, 0 rows affected (0.125 sec)

**********************************************************************************
*****

MariaDB [test]> CREATE PROCEDURE insert_data()
    ->     BEGIN
    ->         insert into department values(1,'Accounts'),
    ->         (2,'Production'),
    ->         (3,'Marketing');
    ->
    ->         insert into employee values(1,'Pradip',1,'Manager',80000),
    ->         (2,'Ajinkya',2,'Clerk',60000),
    ->         (3,'Nirav',3,'Staff',50000),
    ->         (4,'Milind',2,'Manager',60000),
    ->         (5,'Lakshya',3,'Staff',90000);
    ->     END //
Query OK, 0 rows affected (0.020 sec)

MariaDB [test]> call insert_data() //
Query OK, 8 rows affected (0.032 sec)

MariaDB [test]> select * from employee //
+--------+---------+---------+-------------+----------+
| emp_id | empname | dept_id | designation | salary   |
+--------+---------+---------+-------------+----------+
|      1 | Pradip  |       1 | Manager     | 80000.00 |
|      2 | Ajinkya |       2 | Clerk       | 60000.00 |
|      3 | Nirav   |       3 | Staff       | 50000.00 |
|      4 | Milind  |       2 | Manager     | 60000.00 |
|      5 | Lakshya |       3 | Staff       | 90000.00 |
+--------+---------+---------+-------------+----------+
5 rows in set (0.000 sec)

MariaDB [test]> select * from department //
+---------+------------+
| dept_id | deptname   |
+---------+------------+
|       1 | Accounts   |
|       2 | Production |
|       3 | Marketing  |
+---------+------------+
3 rows in set (0.000 sec)
```

****

```
MariaDB [test]> set @dname = 'Production' //
Query OK, 0 rows affected (0.000 sec)

MariaDB [test]> set @list = '' //
Query OK, 0 rows affected (0.000 sec)

MariaDB [test]> CREATE PROCEDURE cur_pro(IN dept_name varchar(100), INOUT list varchar(100))
    ->     BEGIN
    ->     DECLARE emp_name varchar(50);
    ->     DECLARE c_finish integer DEFAULT 0;
    ->     DECLARE curs cursor for select empname from employee where dept_id = (select dept_id
from department where deptname = dept_name );
    ->     DECLARE CONTINUE HANDLER for NOT FOUND set c_finish = 1;
    ->     OPEN curs;
    ->     get_line : LOOP
    ->         FETCH curs into emp_name;
```

```
   ->        IF c_finish = 1 THEN
   ->           LEAVE get_line;
   ->        END IF;
   ->        set list = CONCAT(list,CONCAT(emp_name, " | "));
   ->    END LOOP get_line;
   ->    CLOSE curs;
   ->    END //
Query OK, 0 rows affected (0.021 sec)

MariaDB [test]> DELIMITER ;
MariaDB [test]> call cur_pro(@dname,@list);
Query OK, 0 rows affected (0.000 sec)

MariaDB [test]> select @list as LISTS;
+---------------------+
| LISTS               |
+---------------------+
| Ajinkya | Milind |  |
+---------------------+
1 row in set (0.000 sec)
```

==

Qusetion 3 : Create a cursor to increment the salary based on the designation

==

```
DELIMITER //
```

```
MariaDB [test]> set @increment = 1000 //

Query OK, 0 rows affected (0.000 sec)


MariaDB [test]> set @designation = 'Staff' //

Query OK, 0 rows affected (0.000 sec)



MariaDB [test]> CREATE PROCEDURE salary_increment(IN desig varchar(30) , IN incre decimal(10,2) )
    -> BEGIN
    ->     DECLARE empid int;
    ->     DECLARE saly decimal(10,2);
    ->     DECLARE c_finish integer DEFAULT 0;
    ->     DECLARE curs cursor for select emp_id,salary from employee where designation = desig;
    ->     DECLARE CONTINUE HANDLER for NOT FOUND set c_finish = 1;
    ->     OPEN curs;
    ->     get_line : LOOP
    ->         FETCH curs into empid,saly;
    ->         IF c_finish = 1 THEN
    ->             LEAVE get_line;
    ->         END IF;
    ->         UPDATE employee set salary = (saly + incre) where emp_id = empid;
    ->     END LOOP get_line;
    ->     CLOSE curs;
    -> END //
Query OK, 0 rows affected (0.022 sec)


MariaDB [test]> DELIMITER ;


MariaDB [test]> call salary_increment(@designation,@increment);

Query OK, 2 rows affected (0.022 sec)
```

```
MariaDB [test]> select * from employee;
+--------+---------+---------+-------------+----------+
| emp_id | empname | dept_id | designation | salary   |
+--------+---------+---------+-------------+----------+
|      1 | Pradip  |       1 | Manager     | 80000.00 |
|      2 | Ajinkya |       2 | Clerk       | 60000.00 |
|      3 | Nirav   |       3 | Staff       | 51000.00 |
|      4 | Milind  |       2 | Manager     | 60000.00 |
|      5 | Lakshya |       3 | Staff       | 91000.00 |
+--------+---------+---------+-------------+----------+
5 rows in set (0.000 sec)
```

===

/***
****

Name : Pradip S Karmakar

Class : M.C.A 2

Roll_No : 10

Subject : RDBMS

****
************/

==

GENERAL PL/SQL BLOCKS

==

Question 1 : Input two numbers and find out all arthmetic operations(+, -, x, /).

==

```
MariaDB [test]> DELIMITER //

MariaDB [test]>

MariaDB [test]> create procedure question1( IN a int,IN b int )

    -> BEGIN

    -> DECLARE c INT;

    ->

    -> set c = a+b;

    -> select c as Addition;

    ->

    -> set c = a-b;

    -> select c as Subtraction;

    ->
    -> set c = a*b;
    -> select c as Multiplication;
    ->
    -> set c = a/b;
    -> select c as Division;
    -> END //
Query OK, 0 rows affected (0.021 sec)

MariaDB [test]> delimiter ;
MariaDB [test]> call artmetic(10,5);
+----------+
| Addition |
+----------+
|       15 |
+----------+
1 row in set (0.000 sec)

+-------------+
| Subtraction |
+-------------+
|           5 |
+-------------+
1 row in set (0.006 sec)

+----------------+
| Multiplication |
+----------------+
|             50 |
+----------------+
1 row in set (0.008 sec)
```

```
+----------+
| Division |
+----------+
|        2 |
+----------+
1 row in set (0.012 sec)

Query OK, 0 rows affected (0.015 sec)
```

===

Question 2 : Enter rollno and three subject marks. Find out Total, percentage, result

& Grade.

===

```
MariaDB [test]> DELIMITER //
MariaDB [test]>
MariaDB [test]> CREATE PROCEDURE question2( IN r_no int, IN marks1 int, IN marks2 int, IN marks3 int )
    -> BEGIN
    -> DECLARE total INT;
    -> DECLARE percentage FLOAT;
    -> DECLARE Grade VARCHAR(15);
    -> DECLARE result VARCHAR(4);
    ->
    -> set total = marks1 + marks2 + marks3;
    -> set percentage = (total * 100) / 300;
    ->
    -> IF percentage > 80 THEN
    -> set Grade = "DISTINCTION";
    -> set result = "PASS";
    -> ELSEIF percentage > 70 THEN
    -> set Grade = "FIRST CLASS";
    -> set result = "PASS";
    -> ELSEIF percentage > 60 THEN
    -> set Grade = "SECOND CLASS";
    -> set result = "PASS";
    -> ELSEIF percentage > 50 THEN
    -> set Grade = "THIRD CLASS";
    -> set result = "PASS";
    -> ELSEIF percentage > 35 THEN
    -> set Grade = "PASS";
    -> set result = "PASS";
    -> ELSE
    -> set Grade = "FAIL";
    -> set result = "FAIL";
    -> END IF;
    ->
    -> SELECT r_no,marks1,marks2,marks3,total,percentage,Grade,result as RESULT;
    ->
    -> END //
Query OK, 0 rows affected (0.026 sec)

MariaDB [test]>
MariaDB [test]> DELIMITER ;
MariaDB [test]> call question2(10,78,95,71);
+------+--------+--------+--------+-------+------------+-------------+--------+
| r_no | marks1 | marks2 | marks3 | total | percentage | Grade       | RESULT |
+------+--------+--------+--------+-------+------------+-------------+--------+
|   10 |     78 |     95 |     71 |   244 |    81.3333 | DISTINCTION | PASS   |
+------+--------+--------+--------+-------+------------+-------------+--------+
1 row in set (0.001 sec)

Query OK, 0 rows affected (0.005 sec)
```

==
================

Question 3 : Print First 10 Odd Number unsing Loops.

==
================

```
MariaDB [test]> DELIMITER //

MariaDB [test]> CREATE PROCEDURE question3()
    -> BEGIN
    -> DECLARE odd varchar(50);
    -> DECLARE cnt INT;
    -> DECLARE num INT;
    -> SET odd = '';
    -> SET num = 1;
    -> SET cnt = 1;
    -> loop_odd: LOOP
    ->     IF cnt > 10 THEN
    ->         LEAVE loop_odd;
    ->     END IF;
    ->     IF (num mod 2) THEN
    ->         SET odd = CONCAT(odd,num,"  ");
    ->         SET cnt= cnt + 1;
    ->         SET num = num + 1;
    ->     ELSE
```

```
   ->        SET num = num + 1;
   ->    END IF;
   -> END LOOP;
   ->
   -> select odd as FIRST_10_ODD_NUMBERS;
   ->
   -> END //
Query OK, 0 rows affected (0.021 sec)

MariaDB [test]> DELIMITER ;
MariaDB [test]> call question3;
+-------------------------------------+
| FIRST_10_ODD_NUMBERS                |
+-------------------------------------+
| 1  3  5  7  9  11  13  15  17  19   |
+-------------------------------------+
1 row in set (0.000 sec)

Query OK, 0 rows affected (0.006 sec)
```

==

Question 4 : Print Prime Number Upto 10 using While Loops.

==

```
MariaDB [test]> DELIMITER //

MariaDB [test]> CREATE PROCEDURE question4()
```

```
    -> BEGIN
    -> DECLARE prime varchar(50);
    -> DECLARE cnt INT;
    -> DECLARE i INT;
    -> DECLARE num INT;
    -> SET prime = '';
    -> SET i = 1;
    ->
    -> WHILE i <= 10 DO
    ->     SET cnt = 0;
    ->     SET num = 1;
    ->     WHILE num <= (i/2) DO
    ->         IF (i mod num = 0) THEN
    ->             SET cnt = cnt + 1;
    ->         END IF;
    ->         SET num = num + 1;
    ->     END WHILE;
    ->     IF cnt = 1 THEN
    ->         SET prime = CONCAT(prime,i,"   ");
    ->     END IF;
    ->     SET i = i + 1;
    -> END WHILE;
    ->
    -> select prime as PRIME_NUMBER_UPTO_10;
    ->
    -> END //
Query OK, 0 rows affected (0.021 sec)


MariaDB [test]> DELIMITER ;
MariaDB [test]> call question4;
+----------------------+
| PRIME_NUMBER_UPTO_10 |
+----------------------+
| 2   3   5   7        |
+----------------------+
1 row in set (0.000 sec)

Query OK, 0 rows affected (0.005 sec)
```

==

Question 5 : Print MAX & MIN number from 3 numbers.

==

```
MariaDB [test]> DELIMITER //
MariaDB [test]>
MariaDB [test]> CREATE PROCEDURE question5(IN num1 int, IN num2 int, IN num3 int)
    -> BEGIN
    -> DECLARE Minimum INT;
    -> DECLARE Maximum INT;
    -> set Maximum = 0;
    -> set Minimum = 0;
    -> IF (num1 > num2) AND (num1 > num3) THEN
    ->     set Maximum = num1;
    -> ELSEIF (num2 > num1) AND (num2 > num3) THEN
    ->     set Maximum = num2;
    -> ELSE
    ->     set Maximum = num3;
    -> END IF;
    -> IF (num1 < num2) AND (num1 < num3) THEN
    ->     set Minimum = num1;
    -> ELSEIF (num2 < num1) AND (num2 < num3) THEN
    ->     set Minimum = num2;
    -> ELSE
    ->     set Minimum = num3;
    -> END IF;
    ->
    -> select Maximum,Minimum;
    ->
    -> END //
Query OK, 0 rows affected (0.021 sec)

MariaDB [test]> DELIMITER ;
MariaDB [test]> call question5(10,12,15);
+---------+---------+
| Maximum | Minimum |
+---------+---------+
|      15 |      10 |
+---------+---------+
1 row in set (0.000 sec)

Query OK, 0 rows affected (0.005 sec)
```

==

Question 6 : Get Input From user as empid and check whether that empid is exist, if

Not then Show appropriate Message else show empname and salary.

==

```
MariaDB [test]> DELIMITER //

MariaDB [test]>

MariaDB [test]> CREATE PROCEDURE question6(IN empid int)

    -> BEGIN

    -> IF (select emp_id from employee where emp_id = empid) = empid THEN

    ->     select empname,salary from employee where emp_id = empid;

    -> ELSE

    ->     select "NO SUCH EMPLOYEE ID EXIST" as MESSAGE;

    -> END IF;

    -> END //

Query OK, 0 rows affected (0.019 sec)


MariaDB [test]> DELIMITER ;

MariaDB [test]> call question6(1);

+---------+----------+

| empname | salary   |

+---------+----------+

| Pradip  | 80000.00 |

+---------+----------+

1 row in set (0.000 sec)


Query OK, 0 rows affected (0.007 sec)


MariaDB [test]> call question6(4);

+---------+----------+

| empname | salary   |

+---------+----------+

| Milind  | 60000.00 |

+---------+----------+

1 row in set (0.000 sec)
```

```
Query OK, 0 rows affected (0.007 sec)


MariaDB [test]> call question6(8);
+---------------------------+
| MESSAGE                   |
+---------------------------+
| NO SUCH EMPLOYEE ID EXIST |
+---------------------------+
1 row in set (0.000 sec)


Query OK, 0 rows affected (0.007 sec)
```

==

Question 7 : Get Input From user as empid and check whether that empid is exist, if

Not then Show appropriate Message else show empname and salary.

==

```
MariaDB [test]> DELIMITER //
MariaDB [test]>
MariaDB [test]> CREATE PROCEDURE create_table()
    -> BEGIN
    ->     create table customer(
    ->         cust_id int primary key auto_increment,
    ->         cust_name varchar(15),
    ->         address varchar(150),
    ->         city varchar(25)
    ->     );
    -> END //
Query OK, 0 rows affected (0.018 sec)
```

```
MariaDB [test]> call create_table //

Query OK, 0 rows affected (0.040 sec)


MariaDB [test]> CREATE PROCEDURE insert_data()

    -> BEGIN

    ->     insert into customer values(1,'Pradip','P-block','Navsari'),

    ->     (2,'Ajinkya','E-block','Gandhidham'),

    ->     (3,'Nirav','C-block','Mundra'),

    ->     (4,'Milind','F-block','Navranpura'),

    ->     (5,'Lakshya','G-block','Gandhidham');

    -> END //

Query OK, 0 rows affected (0.020 sec)


MariaDB [test]> call insert_data //

Query OK, 5 rows affected (1.609 sec)


MariaDB [test]> CREATE PROCEDURE question7(IN custid int,IN custname varchar(15), IN
cust_address varchar(150), IN cust_city varchar(25))

    -> BEGIN

    -> IF (select cust_id from customer where cust_id = custid) = custid THEN

    ->     select "CUSTOMER ID ALREADY EXIST" as MESSAGE;

    -> ELSE

    ->     insert into customer values(custid,custname,cust_address,cust_city);

    -> END IF;

    -> END //

Query OK, 0 rows affected (0.020 sec)


MariaDB [test]> DELIMITER ;

MariaDB [test]> call question7(6,'sudip','Kolkata','S-block');

Query OK, 1 row affected (0.007 sec)
```

```
MariaDB [test]> select * from customer;
+---------+-----------+---------+------------+
| cust_id | cust_name | address | city       |
+---------+-----------+---------+------------+
|       1 | Pradip    | P-block | Navsari    |
|       2 | Ajinkya   | E-block | Gandhidham |
|       3 | Nirav     | C-block | Mundra     |
|       4 | Milind    | F-block | Navranpura |
|       5 | Lakshya   | G-block | Gandhidham |
|       6 | sudip     | Kolkata | S-block    |
+---------+-----------+---------+------------+
6 rows in set (0.000 sec)

MariaDB [test]> call question7(3,'kamal','Kolkata','k-block');
+---------------------------+
| MESSAGE                   |
+---------------------------+
| CUSTOMER ID ALREADY EXIST |
+---------------------------+
1 row in set (0.000 sec)

Query OK, 0 rows affected (0.007 sec)
```

==
================

Functions

==
================

Question 1 : Input name and count the length of the name.

==
================

```
MariaDB [test]> CREATE FUNCTION fun_question1(name varchar(20))
    -> RETURNS INT
    ->
    -> BEGIN
    -> DECLARE len INT DEFAULT 0;
    ->
    -> set len = LENGTH(name);
    ->
    -> Return len;
    -> END //
Query OK, 0 rows affected (1.578 sec)


MariaDB [test]> CREATE PROCEDURE Q1(IN name varchar(20))
    -> BEGIN
    -> DECLARE len INT DEFAULT 0;
    -> set len = fun_question1(name);
    -> select len;
    -> END //
Query OK, 0 rows affected (0.025 sec)
```

```
MariaDB [test]> delimiter ;

MariaDB [test]> call Q1("pradip");

+------+

| len  |

+------+

|    6 |

+------+

1 row in set (0.001 sec)


Query OK, 0 rows affected (0.005 sec)


MariaDB [test]> call Q1("pradip karmakar");

+------+

| len  |

+------+

|   15 |

+------+

1 row in set (0.000 sec)


Query OK, 0 rows affected (0.007 sec)
```

==
================

Question 2 : WAF which accepts one number and return TRUE if no is prime and return FALSE if

No. is not prime.

==
================

```
MariaDB [test]> DELIMITER //
MariaDB [test]> CREATE FUNCTION Prime(n INT)
    -> RETURNS BOOL
    -> BEGIN
    ->     DECLARE i INT DEFAULT 0;
    ->     DECLARE FLAG INT DEFAULT 0;
    ->     IF n = 1 THEN
    ->         RETURN FALSE;
    ->     ELSE
    ->         SET i = 2;
    ->         MYLOOP : WHILE i <= (n/2) DO
    ->             IF(n mod i = 0) THEN
    ->                 SET FLAG = 1;
    ->                 LEAVE MYLOOP;
    ->             END IF;
    ->             SET i = i + 1;
    ->         END WHILE;
    ->         IF FLAG = 1 THEN
    ->             RETURN FALSE;
    ->         ELSE
    ->             RETURN TRUE;
    ->         END IF;
    ->     END IF;
    -> END //
Query OK, 0 rows affected (0.021 sec)

MariaDB [test]> DELIMITER ;
MariaDB [test]> select  Prime(1);
+----------+
| Prime(1) |
+----------+
|        0 |
+----------+
1 row in set (0.000 sec)

MariaDB [test]> select  Prime(2);
+----------+
| Prime(2) |
+----------+
|        1 |
+----------+
1 row in set (0.000 sec)

MariaDB [test]> select  Prime(3);
+----------+
| Prime(3) |
+----------+
|        1 |
+----------+
1 row in set (0.000 sec)

MariaDB [test]> select  Prime(4);
+----------+
| Prime(4) |
+----------+
|        0 |
+----------+
1 row in set (0.000 sec)
```

==

Question 3 : Write a function which accepts the department no and returns maximum salary of that

Department. Handle the error if deptno does not exist or select statement return

more than one row.

==
================

```
MariaDB [test]> DELIMITER //

MariaDB [test]> CREATE FUNCTION get_max(dept_no INT)
    -> RETURNS int
    -> BEGIN
    -> DECLARE get_salary DECIMAL(8,2) DEFAULT 0;
    -> DECLARE row INT default 0;
    -> SELECT COUNT(*) INTO row FROM employee WHERE dept_id = dept_no;
    ->    IF (row > 0) THEN
    ->        SELECT MAX(salary) INTO get_salary FROM employee WHERE dept_id = dept_no GROUP BY
dept_id;
    ->        RETURN get_salary;
    ->    ELSE
    ->        RETURN -404;
    ->    END IF;
    -> END //
Query OK, 0 rows affected (0.021 sec)

MariaDB [test]> DELIMITER ;

MariaDB [test]> select * from employee;
+--------+---------+---------+-------------+----------+
| emp_id | empname | dept_id | designation | salary   |
+--------+---------+---------+-------------+----------+
|      1 | Pradip  |       1 | Manager     | 80000.00 |
|      2 | Ajinkya |       2 | Clerk       | 60000.00 |
|      3 | Nirav   |       3 | Staff       | 51000.00 |
|      4 | Milind  |       2 | Manager     | 60000.00 |
|      5 | Lakshya |       3 | Staff       | 91000.00 |
+--------+---------+---------+-------------+----------+
5 rows in set (0.000 sec)


MariaDB [test]> select get_max(3);
+------------+
| get_max(3) |
+------------+
|      91000 |
+------------+
1 row in set (0.001 sec)


MariaDB [test]> select get_max(9);
+------------+
| get_max(9) |
+------------+
|       -404 |
+------------+
1 row in set (0.000 sec)
```

===

Question 4 : Write a function to display whether the entered (User Input) employee no exists

or not.

===

```
MariaDB [test]> DELIMITER //
MariaDB [test]> CREATE FUNCTION isexists(emp_no INT)
    -> RETURNS VARCHAR(25)
    -> BEGIN
    ->     DECLARE row INT DEFAULT 0;
    ->     SELECT COUNT(*) INTO row FROM employee WHERE emp_id = emp_no;
    ->     IF row > 0 THEN
    ->         RETURN "EMPLOYEE EXIST";
    ->     ELSE
    ->         RETURN "EMPLOYEE DOES NOT EXIST";
    ->     END IF;
    -> END //
Query OK, 0 rows affected (0.022 sec)

MariaDB [test]> DELIMITER ;
MariaDB [test]>
MariaDB [test]> select * from employee;
+--------+---------+---------+-------------+----------+
| emp_id | empname | dept_id | designation | salary   |
+--------+---------+---------+-------------+----------+
|      1 | Pradip  |       1 | Manager     | 80000.00 |
|      2 | Ajinkya |       2 | Clerk       | 60000.00 |
|      3 | Nirav   |       3 | Staff       | 51000.00 |
|      4 | Milind  |       2 | Manager     | 60000.00 |
|      5 | Lakshya |       3 | Staff       | 91000.00 |
+--------+---------+---------+-------------+----------+
5 rows in set (0.000 sec)

MariaDB [test]> select isexists(4);
+----------------+
| isexists(4)    |
+----------------+
| EMPLOYEE EXIST |
+----------------+
1 row in set (0.003 sec)

MariaDB [test]> select isexists(9);
+-------------------------+
| isexists(9)             |
+-------------------------+
| EMPLOYEE DOES NOT EXIST |
+-------------------------+
1 row in set (0.000 sec)
```

==
================

Question 5 : WAF which accepts one no and returns that no+100. Use INOUT mode.

==
================

```
MariaDB [test]> DELIMITER //
MariaDB [test]> CREATE FUNCTION summation(num INT)
    -> RETURNS INT
    -> BEGIN
    -> SET num = num + 100;
    -> RETURN num;
    -> END //
Query OK, 0 rows affected (0.021 sec)

MariaDB [test]> DELIMITER ;
MariaDB [test]> SELECT summation(95);

+---------------+

| summation(95) |

+---------------+

|           195 |

+---------------+

1 row in set (0.000 sec)



MariaDB [test]> SELECT summation(-45);

+----------------+

| summation(-45) |

+----------------+

|             55 |

+----------------+

1 row in set (0.000 sec)
```

==
================

Question 6 : WAF which accepts the empno.

If salary<10000 than give raise by 30%.

If salary<20000 and salary>=10000 than give raise by 20%.

If salary>20000 than give raise by 10%. Handle the error if any.

==
================

```
MariaDB [test]> DELIMITER //

MariaDB [test]> CREATE FUNCTION salary_raise(emp_no INT)

    -> RETURNS VARCHAR(30)

    -> BEGIN
```

```
  -> DECLARE get_sal DECIMAL(8,2) DEFAULT 0;
  -> DECLARE row INT DEFAULT 0;
  -> SELECT COUNT(*) INTO row FROM employee WHERE emp_id = emp_no;
  ->    IF(row > 0) THEN
  ->       SELECT salary INTO get_sal FROM employee WHERE emp_id = emp_no;
  ->       IF get_sal > 20000 THEN
  ->          SET get_sal = get_sal + (get_sal*10)/100;
  ->          update employee set salary = get_sal WHERE emp_id = emp_no;
  ->       ELSEIF get_sal > 10000 THEN
  ->          SET get_sal = get_sal + (get_sal*20)/100;
  ->          update employee set salary = get_sal WHERE emp_id = emp_no;
  ->       ELSE
  ->          SET get_sal = get_sal + (get_sal*30)/100;
  ->          update employee set salary = get_sal WHERE emp_id = emp_no;
  ->       END IF;
  ->       RETURN CONCAT('Salary Raised To : ',get_sal);
  ->    ELSE
  ->       RETURN CONCAT('No Such Employee ID Exits');
  ->    END IF;
  -> END //
Query OK, 0 rows affected (1.785 sec)

MariaDB [test]> DELIMITER ;


MariaDB [test]> select * from employee;
+--------+---------+---------+-------------+----------+
| emp_id | empname | dept_id | designation | salary   |
+--------+---------+---------+-------------+----------+
|      1 | Pradip  |       1 | Manager     | 80000.00 |
|      2 | Ajinkya |       2 | Clerk       | 60000.00 |
|      3 | Nirav   |       3 | Staff       | 15000.00 |
|      4 | Milind  |       2 | Manager     | 60000.00 |
|      5 | Lakshya |       3 | Staff       |  9000.00 |
+--------+---------+---------+-------------+----------+
5 rows in set (0.000 sec)


MariaDB [test]> SELECT salary_raise(3);
+----------------------------+
| salary_raise(3)            |
+----------------------------+
| Salary Raised To : 18000.00 |
+----------------------------+
1 row in set (0.004 sec)


MariaDB [test]> SELECT salary_raise(5);
+----------------------------+
| salary_raise(5)            |
+----------------------------+
| Salary Raised To : 11700.00 |
+----------------------------+
1 row in set (0.027 sec)


MariaDB [test]> SELECT salary_raise(2);
+----------------------------+
| salary_raise(2)            |
+----------------------------+
| Salary Raised To : 66000.00 |
+----------------------------+
1 row in set (0.004 sec)



MariaDB [test]> SELECT salary_raise(6);
+---------------------------+
| salary_raise(6)           |
+---------------------------+
| No Such Employee ID Exits |
+---------------------------+
1 row in set (0.000 sec)



MariaDB [test]> select * from employee;
+--------+---------+---------+-------------+----------+
| emp_id | empname | dept_id | designation | salary   |
+--------+---------+---------+-------------+----------+
|      1 | Pradip  |       1 | Manager     | 80000.00 |
|      2 | Ajinkya |       2 | Clerk       | 66000.00 |
|      3 | Nirav   |       3 | Staff       | 18000.00 |
|      4 | Milind  |       2 | Manager     | 60000.00 |
|      5 | Lakshya |       3 | Staff       | 11700.00 |
+--------+---------+---------+-------------+----------+
5 rows in set (0.000 sec)
```

==

Question 7 : WAF which accepts the empno and returns the experience in years. Handle the

error if empno does not exist.

EMP(Empno, Empname, DOJ);

==

```
MariaDB [test]> DELIMITER //

MariaDB [test]> CREATE FUNCTION exp_in_year(emp_no INT)

   -> RETURNS VARCHAR(30)

   -> BEGIN

   ->     DECLARE row INT DEFAULT 0;

   ->     DECLARE experience INT DEFAULT 0;

   ->     SELECT COUNT(*) INTO row FROM employee WHERE emp_id = emp_no;

   ->     IF ( row > 0 ) THEN

   ->         SELECT YEAR(CURDATE())-YEAR(date_of_join) INTO experience FROM employee WHERE
emp_id = emp_no;

   ->         RETURN CONCAT('Experience : ',experience,' years');

   ->     ELSE

   ->         RETURN CONCAT('No Such Employee Id Exists.');

   ->     END IF;

   -> END //

Query OK, 0 rows affected (0.024 sec)


MariaDB [test]> DELIMITER ;


MariaDB [test]> SELECT * FROM employee;

+--------+---------+---------+-------------+----------+--------------+

| emp_id | empname | dept_id | designation | salary   | date_of_join |

+--------+---------+---------+-------------+----------+--------------+

|      1 | Pradip  |       1 | Manager     | 80000.00 | 2013-02-03   |

|      2 | Ajinkya |       2 | Clerk       | 66000.00 | 2015-08-13   |

|      3 | Nirav   |       3 | Staff       | 18000.00 | 2003-11-09   |

|      4 | Milind  |       2 | Manager     | 60000.00 | 2019-02-22   |

|      5 | Lakshya |       3 | Staff       | 11700.00 | 2010-10-10   |

+--------+---------+---------+-------------+----------+--------------+

5 rows in set (0.000 sec)
```

```
MariaDB [test]> select exp_in_year(1);

+----------------------+

| exp_in_year(1)       |

+----------------------+

| Experience : 7 years |

+----------------------+

1 row in set (0.006 sec)



MariaDB [test]> select exp_in_year(4);

+----------------------+

| exp_in_year(4)       |

+----------------------+

| Experience : 1 years |

+----------------------+

1 row in set (0.000 sec)



MariaDB [test]> select exp_in_year(7);

+-----------------------------+

| exp_in_year(7)              |

+-----------------------------+

| No Such Employee Id Exists. |

+-----------------------------+

1 row in set (0.000 sec)
```

==
================

CURSORS

==
================

Question 1 : Create a cursor for the emp table. Produce the output in following format:

{empname} employee working in department {deptno} earns Rs. {salary}.

EMP(empno, empname, salary, deptno);

==
================

```
MariaDB [test]> DELIMITER //

MariaDB [test]> drop procedure cursor_get_detail //

Query OK, 0 rows affected (0.013 sec)


MariaDB [test]> CREATE PROCEDURE cursor_get_detail()

    -> BEGIN

    ->     DECLARE name VARCHAR(20);

    ->     DECLARE deptid INT;

    ->     DECLARE emp_salary DECIMAL(8,2);

    ->     DECLARE stats VARCHAR(100);

    ->     DECLARE FINISHED INT DEFAULT 0;

    ->     DECLARE C1 CURSOR FOR SELECT empname,dept_id,salary FROM employee;

    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET FINISHED = 1;

    ->     OPEN C1;

    ->         data :LOOP

    ->             IF (FINISHED = 1) THEN

    ->                 LEAVE data;

    ->             END IF;

    ->             FETCH C1 INTO name,deptid,emp_salary;
```

```
   ->            SET stats = '';
   ->            SET stats = CONCAT(stats,name,' EMPLOYEE WORKING IN DEPARTMENT ',deptid,' EARNS
RS. ',emp_salary);
   ->            SELECT stats as EMP_DETAIL;
   ->        END LOOP;
   ->    CLOSE C1;
   -> END //
Query OK, 0 rows affected (0.014 sec)

MariaDB [test]> DELIMITER ;
MariaDB [test]> call cursor_get_detail;
+---------------------------------------------------------+
| EMP_DETAIL                                              |
+---------------------------------------------------------+
| Pradip EMPLOYEE WORKING IN DEPARTMENT 1 EARNS RS. 80000.00 |
+---------------------------------------------------------+
1 row in set (0.000 sec)

+----------------------------------------------------------+
| EMP_DETAIL                                               |
+----------------------------------------------------------+
| Ajinkya EMPLOYEE WORKING IN DEPARTMENT 2 EARNS RS. 66000.00 |
+----------------------------------------------------------+
1 row in set (0.004 sec)

+--------------------------------------------------------+
| EMP_DETAIL                                             |
+--------------------------------------------------------+
| Nirav EMPLOYEE WORKING IN DEPARTMENT 3 EARNS RS. 18000.00 |
+--------------------------------------------------------+
1 row in set (0.009 sec)
```

```
+------------------------------------------------------------+
| EMP_DETAIL                                   |
+------------------------------------------------------------+
| Milind EMPLOYEE WORKING IN DEPARTMENT 2 EARNS RS. 60000.00 |
+------------------------------------------------------------+
1 row in set (0.012 sec)

+-------------------------------------------------------------+
| EMP_DETAIL                                    |
+-------------------------------------------------------------+
| Lakshya EMPLOYEE WORKING IN DEPARTMENT 3 EARNS RS. 11700.00 |
+-------------------------------------------------------------+
1 row in set (0.015 sec)

Query OK, 0 rows affected (0.023 sec)
```

==

Question 2 : Create a cursor for updating the salary of emp working in deptno 10 by 20%.

If any rows are affected than display the no of rows affected.

Use implicit cursor.

==

```
MariaDB [test]> DELIMITER //
MariaDB [test]> CREATE PROCEDURE cursor_upate_implicit()
    -> BEGIN
    ->     DECLARE row INT DEFAULT -1;
    ->     DECLARE empid INT;
    ->     DECLARE FINISHED INT DEFAULT 0;
    ->    DECLARE C1 CURSOR FOR SELECT emp_id FROM employee WHERE dept_id = 10;
    ->    DECLARE CONTINUE HANDLER FOR NOT FOUND SET FINISHED = 1;
    ->    OPEN C1;
    ->        data : LOOP
    ->        IF FINISHED = 1 THEN
    ->            LEAVE data;
    ->        END IF;
    ->        FETCH C1 INTO empid;
    ->            UPDATE employee SET salary = salary + (salary * 20)/100 WHERE emp_id = empid;
    ->            SET row = row+1;
    ->        END LOOP;
    ->    CLOSE C1;
    ->    IF row > 0 THEN
    ->        SELECT CONCAT('Row Affected : ', row) as Message;
    ->    ELSE
    ->        SELECT 'No Row Effected' as Message;
    ->    END IF;
    -> END //
Query OK, 0 rows affected (0.023 sec)

MariaDB [test]> DELIMITER ;

MariaDB [test]> select * from employee;
+--------+---------+---------+-------------+----------+--------------+
| emp_id | empname | dept_id | designation | salary   | date_of_join |
+--------+---------+---------+-------------+----------+--------------+
|      1 | Pradip  |       1 | Manager     | 80000.00 | 2013-02-03   |
|      2 | Ajinkya |       2 | Clerk       | 66000.00 | 2015-08-13   |
|      3 | Nirav   |      10 | Staff       | 18000.00 | 2003-11-09   |
|      4 | Milind  |       2 | Manager     | 60000.00 | 2019-02-22   |
|      5 | Lakshya |      10 | Staff       | 11700.00 | 2010-10-10   |
+--------+---------+---------+-------------+----------+--------------+
5 rows in set (0.000 sec)

MariaDB [test]> CALL cursor_upate_implicit;
+------------------+
| Message          |
+------------------+
| Row Affected : 2 |
+------------------+
1 row in set (0.023 sec)

MariaDB [test]> select * from employee;
+--------+---------+---------+-------------+----------+--------------+
| emp_id | empname | dept_id | designation | salary   | date_of_join |
+--------+---------+---------+-------------+----------+--------------+
|      1 | Pradip  |       1 | Manager     | 80000.00 | 2013-02-03   |
|      2 | Ajinkya |       2 | Clerk       | 66000.00 | 2015-08-13   |
|      3 | Nirav   |      10 | Staff       | 21600.00 | 2003-11-09   |
|      4 | Milind  |       2 | Manager     | 60000.00 | 2019-02-22   |
|      5 | Lakshya |      10 | Staff       | 14040.00 | 2010-10-10   |
+--------+---------+---------+-------------+----------+--------------+
5 rows in set (0.000 sec)

Query OK, 2 rows affected (0.026 sec)
```

==

Question 3 : Create a cursor for updating the salary of emp working in deptno 10 by 20%.

If any rows are affected than display the no of rows affected.

Use EXPLICIT cursor.

==

```
MariaDB [test]> CREATE PROCEDURE cursor_upate_explicit()
   -> BEGIN
   ->     DECLARE empid INT;
   ->     DECLARE i INT DEFAULT 0;
   ->     DECLARE FINISHED INT DEFAULT 0;
   ->     DECLARE C1 CURSOR FOR SELECT emp_id FROM employee WHERE dept_id = 10;
   ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET FINISHED = 1;
   ->     OPEN C1;
   ->         data : LOOP
   ->         IF FINISHED = 1 THEN
   ->             LEAVE data;
   ->             set i = i + 1;
   ->             SELECT i as LEAVING;
   ->         ELSE
   ->         FETCH C1 INTO empid;
   ->             UPDATE employee SET salary = salary + (salary * 20)/100 WHERE emp_id = empid;
   ->             set i = i + 1;
   ->             SELECT i as FETCHING;
   ->         END IF;
   ->         END LOOP;
   ->     CLOSE C1;
   -> END //
Query OK, 0 rows affected (0.023 sec)

MariaDB [test]> DELIMITER ;

MariaDB [test]> call cursor_upate_explicit;
```

```
Query OK, 2 rows affected, 1 warning (0.025 sec)

MariaDB [test]> select * from employee;
+--------+---------+---------+-------------+----------+--------------+
| emp_id | empname | dept_id | designation | salary   | date_of_join |
+--------+---------+---------+-------------+----------+--------------+
|      1 | Pradip  |       1 | Manager     | 80000.00 | 2013-02-03   |
|      2 | Ajinkya |       2 | Clerk       | 66000.00 | 2015-08-13   |
|      3 | Nirav   |      10 | Staff       | 25920.00 | 2003-11-09   |
|      4 | Milind  |       2 | Manager     | 60000.00 | 2019-02-22   |
|      5 | Lakshya |      10 | Staff       | 20217.60 | 2010-10-10   |
+--------+---------+---------+-------------+----------+--------------+
5 rows in set (0.000 sec)
```

===

Question 4 : WAP that will display the name, department and salary of the first 10 employees

getting the highest salary.

===

```
MariaDB [test]> DELIMITER //
MariaDB [test]> drop procedure top_10_salary //
Query OK, 0 rows affected (0.022 sec)

MariaDB [test]> CREATE PROCEDURE top_10_salary()
    -> BEGIN
    ->     DECLARE name VARCHAR(20);
    ->     DECLARE deptid INT;
    ->     DECLARE emp_salary FLOAT;
    ->     DECLARE FINISHED INTEGER DEFAULT 0;
    ->     DECLARE C1 CURSOR FOR SELECT empname,dept_id,salary FROM employee ORDER BY salary
DESC LIMIT 10;
    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET FINISHED = 1;
    ->     OPEN C1;
    ->         data :LOOP
    ->             IF FINISHED = 1 THEN
    ->                 LEAVE data;
    ->             END IF;
    ->             FETCH C1 INTO name,deptid,emp_salary;
    ->                 select CONCAT( name,' | ',deptid,' | ',emp_salary) as Employee_Data;
    ->         END LOOP;
    ->     CLOSE C1;
    -> END //
Query OK, 0 rows affected (0.021 sec)

MariaDB [test]> DELIMITER ;
MariaDB [test]> CALL top_10_salary;
+--------------------+
| Employee_Data      |
+--------------------+
| Pradip | 1 | 80000 |
+--------------------+
1 row in set (0.000 sec)

+---------------------+
| Employee_Data       |
+---------------------+
| Ajinkya | 2 | 66000 |
+---------------------+
1 row in set (0.006 sec)
```

```
+------------------+
| Employee_Data    |
+------------------+
| Neel | 4 | 62000 |
+------------------+
1 row in set (0.008 sec)

+-----------------------+
| Employee_Data         |
+-----------------------+
| Lakshya | 10 | 60369.4 |
+-----------------------+
1 row in set (0.014 sec)

+-------------------+
| Employee_Data     |
+-------------------+
| Milind | 2 | 60000 |
+-------------------+
1 row in set (0.017 sec)

+-------------------+
| Employee_Data     |
+-------------------+
| Pratik | 8 | 56000 |
+-------------------+
1 row in set (0.021 sec)

+--------------------+
| Employee_Data      |
+---------------------+
| Shubham | 3 | 49000 |
+---------------------+
1 row in set (0.024 sec)

+----------------------+
| Employee_Data        |
+----------------------+
| Nirav | 10 | 44789.8 |
+----------------------+
1 row in set (0.026 sec)

+--------------------+
| Employee_Data      |
+--------------------+
| Dhaval | 5 | 35000 |
+--------------------+
1 row in set (0.030 sec)

+--------------------+
| Employee_Data      |
+--------------------+
| Hemang | 6 | 32000 |
+--------------------+
1 row in set (0.035 sec)

Query OK, 0 rows affected (0.042 sec)
```

==
================

Question 5 : WAP using parameterized cursor to display all the information of employee living in

specified city. Ask the city from user.

==
================

```
MariaDB [test]> DELIMITER //

MariaDB [test]> CREATE PROCEDURE search_city(user_city VARCHAR(20))
    -> BEGIN
    ->     DECLARE custid INT;
    ->     DECLARE custname VARCHAR(15);
    ->     DECLARE addr VARCHAR(30);
    ->     DECLARE FINISHED INTEGER DEFAULT 0;
    ->     DECLARE C1 CURSOR FOR SELECT cust_id,cust_name,address FROM customer WHERE city =
user_city;
    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET FINISHED = 1;
    ->     OPEN C1;
    ->         data :LOOP
    ->         IF (FINISHED =1) THEN
    ->             LEAVE data;
    ->         END IF;
    ->         FETCH C1 INTO custid,custname,addr;
    ->             SELECT CONCAT(custid,' | ',custname,' | ',addr,' | ',user_city) as User_Detail;
    ->         END LOOP;
    ->     CLOSE C1;
    -> END //
Query OK, 0 rows affected (0.022 sec)


MariaDB [test]> DELIMITER ;

MariaDB [test]> CALL search_city('Gandhidham');

+-----------------------------------+
| User_Detail                    |
+-----------------------------------+
| 2 | Ajinkya | E-block | Gandhidham |
+-----------------------------------+
1 row in set (0.001 sec)

+-----------------------------------+
| User_Detail                    |
+-----------------------------------+
| 5 | Lakshya | G-block | Gandhidham |
+-----------------------------------+
1 row in set (0.005 sec)

Query OK, 0 rows affected (0.014 sec)
```

===

Question 6 : WAP which display the sum of salary department wise.

===

```
MariaDB [test]> DELIMITER //
MariaDB [test]> CREATE PROCEDURE salary_dept()
    -> BEGIN
    ->     DECLARE emp_salary DECIMAL(8,2);
    ->     DECLARE deptid INT;
    ->     DECLARE stats VARCHAR(100) DEFAULT ' ';
    ->     DECLARE FINISHED INTEGER DEFAULT 0;
    ->     DECLARE C1 CURSOR FOR SELECT dept_id FROM department;
```

```
   ->     DECLARE C2 CURSOR FOR SELECT SUM(salary) FROM employee WHERE dept_id = deptid
GROUP BY dept_id;
   ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET FINISHED = 1;
   ->     OPEN C1;
   ->        data :LOOP
   ->           FETCH C1 INTO deptid;
   ->           IF FINISHED = 1 THEN
   ->              LEAVE data;
   ->           END IF;
   ->           OPEN C2;
   ->              data2 :LOOP
   ->                 FETCH C2 INTO emp_salary;
   ->                 IF FINISHED = 1 THEN
   ->                    LEAVE data2;
   ->                 END IF;
   ->                 SET stats = '';
   ->                 SET stats = CONCAT(stats,deptid,' | ',emp_salary);
   ->              END LOOP data2;
   ->           CLOSE C2;
   ->           SET FINISHED = 0;
   ->           SELECT stats;
   ->        END LOOP data;
   ->     CLOSE C1;
   -> END //
Query OK, 0 rows affected (0.022 sec)

MariaDB [test]>
MariaDB [test]> DELIMITER ;
MariaDB [test]> CALL salary_dept;
+--------------+
| stats        |
+--------------+
| 1 | 80000.00 |
+--------------+
1 row in set (0.002 sec)

+---------------+
| stats         |
+---------------+
| 2 | 126000.00 |
+---------------+
1 row in set (0.008 sec)

+--------------+
| stats        |
+--------------+
| 3 | 49000.00 |
+--------------+
1 row in set (0.010 sec)

+--------------+
| stats        |
+--------------+
| 4 | 62000.00 |
+--------------+
1 row in set (0.015 sec)

+--------------+
| stats        |
+--------------+
| 5 | 35000.00 |
+--------------+
1 row in set (0.016 sec)

+--------------+
| stats        |
+--------------+
| 6 | 32000.00 |
+--------------+
1 row in set (0.021 sec)


+--------------+
| stats        |
+--------------+
| 8 | 56000.00 |
+--------------+
1 row in set (0.026 sec)


+----------------+
| stats          |
+----------------+
| 10 | 105159.18 |
+----------------+
1 row in set (0.033 sec)

Query OK, 0 rows affected (0.036 sec)
```

==

Question 7 : Create a cursor to generate defferent two tables from one master table.

Students(Rno, Name, Std, B_date, Sex);

Girl_Table(Rno, Name, Std, B_date);

Boy_Table(Rno, Name, Std, B_date);

First fetch the row from Student table. If sex is ‘M’ then insert that row in

Boy_Table and if ‘F’ then insert that row in Girl_Table.

In both table Rollno entry must be in Sequence(Using create sequence command).

==

```
MariaDB [test]> DELIMITER //
MariaDB [test]> CREATE PROCEDURE stud_gender()
    -> BEGIN
    ->     DECLARE row INT;
    ->     DECLARE s_rno INT;
    ->     DECLARE s_name VARCHAR(20);
    ->     DECLARE s_std INT;
    ->     DECLARE s_bday DATE;
    ->     DECLARE s_sex VARCHAR(1);
    ->     DECLARE FINISHED INTEGER DEFAULT 0;
    ->     DECLARE C1 CURSOR FOR SELECT * FROM students;
    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET FINISHED = 1;
    ->     OPEN C1;
    ->         data :LOOP
    ->             FETCH C1 INTO s_rno,s_name,s_std,s_bday,s_sex;
    ->             IF (FINISHED =1) THEN
    ->                 LEAVE data;
    ->             END IF;
    ->             IF (s_sex = 'F') THEN
    ->            SELECT COUNT(*) INTO row FROM information_schema.tables WHERE table_schema =
'test' AND table_name = 'girl';
    ->            IF row = 0 THEN
    ->              CREATE TABLE girl(Rno INT AUTO_INCREMENT PRIMARY KEY, Name VARCHAR(20),
Std INT, B_date DATE);
    ->              INSERT INTO girl(Name,Std,B_date) VALUES(s_name,s_std,s_bday);
    ->            ELSE
    ->              INSERT INTO girl(Name,Std,B_date) VALUES(s_name,s_std,s_bday);
    ->            END IF;
    ->          END IF;
    ->          IF (s_sex = 'M') THEN
    ->            SELECT COUNT(*) INTO row FROM information_schema.tables WHERE table_schema =
'test'AND table_name = 'boy';
    ->            IF row = 0 THEN
    ->              CREATE TABLE boy(Rno INT AUTO_INCREMENT PRIMARY KEY, Name VARCHAR(20),
Std INT, B_date DATE);
    ->              INSERT INTO boy (Name,Std,B_date) VALUES(s_name,s_std,s_bday);
    ->            ELSE
    ->              INSERT INTO boy (Name,Std,B_date) VALUES(s_name,s_std,s_bday);
    ->            END IF;
    ->          END IF;
    ->        END LOOP data;
    ->    CLOSE C1;
    -> END //
Query OK, 0 rows affected (0.023 sec)


MariaDB [test]> DELIMITER ;


MariaDB [test]> CALL stud_gender;
Query OK, 14 rows affected (1.786 sec)
```

```
MariaDB [test]> SELECT * FROM GIRL;
+-----+---------+------+------------+
| Rno | Name    | Std  | B_date     |
+-----+---------+------+------------+
|   1 | Kanchan |    8 | 2000-06-01 |
|   2 | Shivani |    8 | 1999-03-26 |
|   3 | Riddhi  |    8 | 1998-08-17 |
+-----+---------+------+------------+
3 rows in set (0.000 sec)


MariaDB [test]> SELECT * FROM BOY;
+-----+---------+------+------------+
| Rno | Name    | Std  | B_date     |
+-----+---------+------+------------+
|   1 | Pradip  |    8 | 1998-04-25 |
|   2 | Monil   |    8 | 1999-12-19 |
|   3 | Piyush  |    8 | 1998-02-21 |
|   4 | Anubhav |    8 | 1997-07-22 |
+-----+---------+------+------------+
4 rows in set (0.000 sec)


MariaDB [test]> SELECT * FROM STUDENTS;
+-----+---------+-----+------------+-----+
| Rno | Name    | Std | B_date     | Sex |
+-----+---------+-----+------------+-----+
|   1 | Pradip  |   8 | 1998-04-25 | M   |
|   2 | Monil   |   8 | 1999-12-19 | M   |
|   3 | Kanchan |   8 | 2000-06-01 | F   |
|   4 | Piyush  |   8 | 1998-02-21 | M   |
|   5 | Shivani |   8 | 1999-03-26 | F   |
|   6 | Riddhi  |   8 | 1998-08-17 | F   |
|   7 | Anubhav |   8 | 1997-07-22 | M   |
+-----+---------+-----+------------+-----+
7 rows in set (0.000 sec)
```

==

Procedure

==

Question 1 : Write a procedure which accepts the empno and returns the associated empname.

If empno does not exist than give proper error message.

EMP(Empno, Empname).

==

```
MariaDB [test]> DELIMITER //
MariaDB [test]> CREATE PROCEDURE emp_call(IN EMP_NO VARCHAR(20))
    -> BEGIN
    ->     DECLARE row INT;
    ->     SELECT COUNT(*) INTO row FROM emp1 WHERE Empno = EMP_NO;
    ->     IF row > 0 THEN
    ->         SELECT Empname FROM emp1 as Name WHERE Empno = EMP_NO;
    ->     ELSE
    ->         SELECT "EMPLOYEE DOSE NOT EXIST." as MESSAGE;
    ->     END IF;
    -> END //
```

```
Query OK, 0 rows affected (0.021 sec)


MariaDB [test]> DELIMITER ;

MariaDB [test]> CALL emp_call(1);

+---------+

| Empname |

+---------+

| Pradip  |

+---------+

1 row in set (0.002 sec)


Query OK, 1 row affected (0.005 sec)


MariaDB [test]> CALL emp_call(5);

+---------+

| Empname |

+---------+

| Lakshya |

+---------+

1 row in set (0.000 sec)


Query OK, 1 row affected (0.005 sec)


MariaDB [test]> CALL emp_call(8);

+-------------------------+

| MESSAGE                 |

+-------------------------+

| EMPLOYEE DOSE NOT EXIST. |

+-------------------------+

1 row in set (0.000 sec)
```

```
Query OK, 1 row affected (0.006 sec)
```

==

Question 2 : WAP which accepts the student rollno and returns the name,city and marks of

all the subjects of that student.

STUDENT (Stud_ID, Stud_name, m1, m2, m3).

==

```
MariaDB [test]> DELIMITER //
MariaDB [test]> CREATE PROCEDURE std_data(IN R_NO INT)
    -> BEGIN
    -> DECLARE row INT DEFAULT 0;
    ->     SELECT COUNT(*) INTO row FROM student1 WHERE Stud_ID = R_NO;
    ->     IF row > 0 THEN
    ->         SELECT Stud_name,m1,m2,m3 from student1 where Stud_ID = R_NO;
    ->     ELSE
    ->         SELECT "NO DETAIL FOUND" as MESSAGE;
    ->     END IF;
    -> END //
Query OK, 0 rows affected (0.023 sec)

MariaDB [test]> DELIMITER ;
MariaDB [test]> call std_data(1);
+-----------+------+------+------+
| Stud_name | m1   | m2   | m3   |
+-----------+------+------+------+
| Pradip    |   45 |   76 |   66 |
+-----------+------+------+------+
1 row in set (0.000 sec)

Query OK, 1 row affected (0.002 sec)

MariaDB [test]> call std_data(2);
+-----------+------+------+------+
| Stud_name | m1   | m2   | m3   |
+-----------+------+------+------+
| Nirav     |   96 |   97 |   99 |
+-----------+------+------+------+
1 row in set (0.000 sec)

Query OK, 1 row affected (0.002 sec)

MariaDB [test]> call std_data(3);
+-----------------+
| MESSAGE         |
+-----------------+
| NO DETAIL FOUND |
+-----------------+
1 row in set (0.000 sec)

Query OK, 1 row affected (0.006 sec)
```

===
================

Question 3 : WAP which accepts the name from the user. Return UPPER if name is in uppercase,

LOWER if name is in lowercase, MIXCASE if name is entered using both the case.

===
================

```
MariaDB [test]> DELIMITER //
MariaDB [test]> CREATE PROCEDURE case_check(IN user_input VARCHAR(20))
    -> BEGIN
    ->     DECLARE i INT DEFAULT 1;
    ->     DECLARE up INT DEFAULT 0;
    ->     DECLARE low INT DEFAULT 0;
    ->     DECLARE len INT;
    ->     DECLARE ch int;
    ->     SET LEN = LENGTH(user_input);
    ->     WHILE i <= len DO
    ->        SET ch = ASCII(SUBSTR(user_input,i,1));
    ->        IF ch >= 65 AND ch <= 90 THEN
    ->           SET up = up + 1;
    ->        ELSE
    ->           SET low = low + 1;
    ->        END IF;
    ->        SET i = i + 1;
    ->     END WHILE;
    ->     IF ( len = up) THEN
    ->        SELECT "STRING IS IN UPPERCASE FORM." as MESSAGE;
    ->     ELSEIF(len = low) THEN
    ->        SELECT "STRING IS IN LOWERCASE FORM." as MESSAGE;
    ->     ELSE
    ->        SELECT "STRING IS IN MIXCASE FORM" as MESSAGE;
    ->     END IF;

    -> END //

Query OK, 0 rows affected (0.021 sec)


MariaDB [test]> DELIMITER ;

MariaDB [test]> call case_check('pradip');

+------------------------------+

| MESSAGE                      |

+------------------------------+

| STRING IS IN LOWERCASE FORM. |

+------------------------------+

1 row in set (0.000 sec)


Query OK, 0 rows affected (0.003 sec)


MariaDB [test]> call case_check('PRADIP');

+------------------------------+

| MESSAGE                      |

+------------------------------+

| STRING IS IN UPPERCASE FORM. |

+------------------------------+

1 row in set (0.000 sec)


Query OK, 0 rows affected (0.004 sec)


MariaDB [test]> call case_check('PrAdip');

+---------------------------+

| MESSAGE                   |

+---------------------------+

| STRING IS IN MIXCASE FORM |

+---------------------------+
1 row in set (0.000 sec)

Query OK, 0 rows affected (0.003 sec)
```

==

Question 4 : WAP which accepts the student rollno and returns the highest percent and name

of that student to the calling block.

STUDENT(Stud_ID,Stud_name,percent);

==

```
MariaDB [test]> DELIMITER //
MariaDB [test]> CREATE PROCEDURE std_percent(IN R_NO INT)
    -> BEGIN
    -> DECLARE row INT DEFAULT 0;
    -> DECLARE name varchar(30);
    -> DECLARE total FLOAT DEFAULT 0;
    -> DECLARE percent FLOAT DEFAULT 0;
    ->    SELECT COUNT(*) INTO row FROM student1 WHERE Stud_ID = R_NO;
    ->    IF row > 0 THEN
    ->       Select Stud_name INTO name from student1 where Stud_ID = R_NO;
    ->       select m1+m2+m3 INTO total from student1 where Stud_ID = R_NO;
    ->       set percent = (total*100)/300;
    ->       SELECT CONCAT('Highest Percent of ', percent ,' Obtain By ',name) as STUDENT_DATA;
    ->    ELSE
    ->       SELECT "NO DETAIL FOUND" as MESSAGE;
    ->    END IF;
    -> END //
Query OK, 0 rows affected (0.021 sec)
```

```
MariaDB [test]> DELIMITER ;

MariaDB [test]> call std_percent(1);

+---------------------------------------------+

| STUDENT_DATA                              |

+---------------------------------------------+

| Highest Percent of 62.3333 Obtain By Pradip |

+---------------------------------------------+

1 row in set (0.001 sec)


Query OK, 3 rows affected (0.006 sec)


MariaDB [test]> call std_percent(2);

+--------------------------------------------+

| STUDENT_DATA                             |

+--------------------------------------------+

| Highest Percent of 97.3333 Obtain By Nirav |

+--------------------------------------------+

1 row in set (0.000 sec)


Query OK, 3 rows affected (0.006 sec)


MariaDB [test]> call std_percent(3);

+-----------------+

| MESSAGE         |

+-----------------+

| NO DETAIL FOUND |

+-----------------+

1 row in set (0.000 sec)


Query OK, 1 row affected (0.005 sec)
```

==

Question 5 : WAP which accepts the date of joining for specific employee and returns the years of experience along with its name. Accept the Employee no from user.

EMP (empno, empname, DOJ);

==

```
MariaDB [test]> DELIMITER //
MariaDB [test]> CREATE PROCEDURE exp_name(IN empno INT)
    -> BEGIN
    ->     DECLARE row INT;
    ->     DECLARE experience INT;
    ->     DECLARE name VARCHAR(20);
    ->     SELECT COUNT(*) INTO row FROM employee WHERE emp_id = empno;
    ->     IF row > 0 THEN
    ->         SELECT YEAR(CURDATE())-YEAR(date_of_join) INTO experience FROM employee WHERE
emp_id = empno;
    ->         SELECT empname INTO name FROM employee WHERE emp_id = empno;
    ->         SELECT name as NAME,experience as Experience;
    ->     ELSE
    ->         SELECT 'NO such Employee ID Found' as MESSAGE;
    ->     END IF;
    -> END //
Query OK, 0 rows affected (0.022 sec)

MariaDB [test]> DELIMITER ;
MariaDB [test]> CALL exp_name(1);
+--------+------------+
| NAME   | Experience |
+--------+------------+
| Pradip |          7 |
+--------+------------+
1 row in set (0.001 sec)

Query OK, 3 rows affected (0.006 sec)

MariaDB [test]> CALL exp_name(3);
+-------+------------+
| NAME  | Experience |
+-------+------------+
| Nirav |         17 |
+-------+------------+
1 row in set (0.000 sec)

Query OK, 3 rows affected (0.005 sec)

MariaDB [test]> CALL exp_name(19);
+---------------------------+
| MESSAGE                   |
+---------------------------+
| NO such Employee ID Found |
+---------------------------+
1 row in set (0.000 sec)

Query OK, 1 row affected (0.006 sec)
```

==
================

Question 6 : WAP which accepts the student roll no and returns the result (in the form of

class: first class, second class, third class or fail).

STUDENT (Stud_ID, Stud_name,m1, m2, m3).

==
================

```
MariaDB [test]> DELIMITER //
MariaDB [test]> drop procedure std_result //
Query OK, 0 rows affected (0.021 sec)

MariaDB [test]> CREATE PROCEDURE std_result(IN R_NO INT)
    -> BEGIN
    ->     DECLARE row INT DEFAULT 0;
    ->     DECLARE name varchar(30);
    ->     DECLARE total FLOAT DEFAULT 0;
    ->     DECLARE percent FLOAT DEFAULT 0;
    ->         SELECT COUNT(*) INTO row FROM student1 WHERE Stud_ID = R_NO;
    ->         IF row > 0 THEN
    ->             Select Stud_name INTO name from student1 where Stud_ID = R_NO;
    ->             select m1+m2+m3 INTO total from student1 where Stud_ID = R_NO;
    ->             set percent = (total*100)/300;
    ->             IF percent > 80 THEN
    ->                 SELECT name as Name, "DISTINCTION" as RESULT;
    ->             ELSEIF percent > 70 THEN
    ->                 SELECT name as Name, "FIRST CLASS" as RESULT;
    ->             ELSEIF percent > 60 THEN
    ->                 SELECT name as Name, "SECOND CLASS" as RESULT;
    ->             ELSEIF percent > 50 THEN
    ->                 SELECT name as Name, "THIRD CLASS" as RESULT;
```

```
   ->            ELSEIF percent > 35 THEN
   ->                SELECT name as Name, "PASS CLASS" as RESULT;
   ->            ELSE
   ->                SELECT name as Name, "FAIL" as RESULT;
   ->            END IF;
   ->        ELSE
   ->            SELECT "NO DETAIL FOUND" as MESSAGE;
   ->        END IF;
   -> END //
Query OK, 0 rows affected (0.021 sec)

MariaDB [test]> DELIMITER ;
MariaDB [test]> call std_result(1);
+--------+--------------+
| Name   | RESULT       |
+--------+--------------+
| Pradip | SECOND CLASS |
+--------+--------------+
1 row in set (0.001 sec)

Query OK, 3 rows affected (0.006 sec)

MariaDB [test]> call std_result(2);
+-------+-------------+
| Name  | RESULT      |
+-------+-------------+
| Nirav | DISTINCTION |
+-------+-------------+
1 row in set (0.000 sec)

Query OK, 3 rows affected (0.006 sec)
```

```
MariaDB [test]> call std_result(3);
+-----------------+
| MESSAGE         |
+-----------------+
| NO DETAIL FOUND |
+-----------------+
1 row in set (0.000 sec)

Query OK, 1 row affected (0.005 sec)
```

E.O.F

```
*****************************************************************************
**************/
```

****
****

Name : Pradip S Karmakar

Class : M.C.A 2

Roll_No : 10

Subject : RDBMS

****
************/

===
==============

TOPIC : Triggers

===
==============

1. Q(example 11.2) : This example is divided in three categories : Insert, Update and Delete

a. Insert : Write a trigger which updates the sale value if customer already exists else create new entry of customer.
b. Update : If the customer is updating , WAT to update the sales value by incrementing the Sale_vale field.
c. Delete : If the customer is deleting , WAT to update the sales value by decrementing the Sale_vale field.

===
==============

INSERT

===
==============

```
MariaDB [test]> DELIMITER //
MariaDB [test]> CREATE TRIGGER sales_bi_trg BEFORE INSERT ON sales
    -> FOR EACH ROW
    -> BEGIN
    -> DECLARE row_count INTEGER;
    ->     SELECT COUNT(*)
    ->     INTO row_count
    ->     FROM customer_sales_total
    ->     WHERE cust_id=NEW.cust_id;
    ->
    ->     IF row_count > 0 THEN
    ->        UPDATE customer_sales_total
    ->        SET sale_value=sale_value+NEW.sale_value
    ->        WHERE cust_id=NEW.cust_id;
    ->     ELSE
    ->        INSERT INTO customer_sales_total
    ->        (cust_id,sale_value)
    ->        VALUES(NEW.cust_id,NEW.sale_value);
    ->     END IF;
    -> END//
Query OK, 0 rows affected (0.021 sec)

MariaDB [test]> DELIMITER ;



MariaDB [test]> insert into sales(cust_id,product_name,sale_value) values(1,'printer',3500);
Query OK, 1 row affected (0.016 sec)



MariaDB [test]> select * from customer_sales_total;
```

```
+---------+------------+
| cust_id | sale_value |
+---------+------------+
|       1 |    3500.00 |
+---------+------------+
1 row in set (0.000 sec)


MariaDB [test]> select * from sales;
+----------+---------+--------------+------------+
| sales_id | cust_id | product_name | sale_value |
+----------+---------+--------------+------------+
|        1 |       1 | printer      |    3500.00 |
+----------+---------+--------------+------------+
1 row in set (0.000 sec)


MariaDB [test]> insert into sales(cust_id,product_name,sale_value) values(1,'Page Bundle',400);
Query OK, 1 row affected (0.008 sec)


MariaDB [test]> select * from customer_sales_total;
+---------+------------+
| cust_id | sale_value |
+---------+------------+
|       1 |    3900.00 |
+---------+------------+
1 row in set (0.000 sec)


MariaDB [test]> select * from sales;
```

```
+----------+---------+--------------+------------+
| sales_id | cust_id | product_name | sale_value |
+----------+---------+--------------+------------+
|        1 |       1 | printer      |    3500.00 |
|        2 |       1 | Page Bundle  |     400.00 |
+----------+---------+--------------+------------+
2 rows in set (0.000 sec)

MariaDB [test]> insert into sales(cust_id,product_name,sale_value) values(2,'mouse',870);
Query OK, 1 row affected (0.007 sec)

MariaDB [test]> select * from customer_sales_total;
+---------+------------+
| cust_id | sale_value |
+---------+------------+
|       1 |    3900.00 |
|       2 |     870.00 |
+---------+------------+
2 rows in set (0.000 sec)

MariaDB [test]> select * from sales;
+----------+---------+--------------+------------+
| sales_id | cust_id | product_name | sale_value |
+----------+---------+--------------+------------+
|        1 |       1 | printer      |    3500.00 |
|        2 |       1 | Page Bundle  |     400.00 |
|        3 |       2 | mouse        |     870.00 |
+----------+---------+--------------+------------+
3 rows in set (0.000 sec)
```

===
=============

UPDATE

===
=============

```
MariaDB [test]> DELIMITER //
MariaDB [test]> CREATE TRIGGER sales_bu_trg BEFORE UPDATE ON sales FOR EACH ROW
    -> BEGIN
    ->     UPDATE customer_sales_total
    ->     SET sale_value=sale_value+(NEW.sale_value-OLD.sale_value)
    ->     WHERE cust_id=NEW.cust_id;
    -> END //
Query OK, 0 rows affected (0.019 sec)

MariaDB [test]> DELIMITER ;

MariaDB [test]> update sales set sale_value = 550 where sales_id = 2;
Query OK, 1 row affected (0.007 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [test]> select * from customer_sales_total;
+---------+------------+
| cust_id | sale_value |
+---------+------------+
|       1 |    4050.00 |
|       2 |     870.00 |
+---------+------------+
2 row in set (0.000 sec)

MariaDB [test]> select * from sales;
+----------+---------+--------------+------------+
| sales_id | cust_id | product_name | sale_value |
+----------+---------+--------------+------------+
|        1 |       1 | printer      |    3500.00 |
|        2 |       1 | Page Bundle  |     550.00 |
|        3 |       2 | mouse        |     870.00 |
+----------+---------+--------------+------------+
3 rows in set (0.000 sec)
```

```
===============================================================================
=============
                              DELETE
===============================================================================
=============
```

```
MariaDB [test]> DELIMITER //
MariaDB [test]> CREATE  TRIGGER sales_bd_trg BEFORE DELETE ON sales FOR EACH ROW
    -> BEGIN
    ->     UPDATE customer_sales_total
    ->     SET sale_value=sale_value-OLD.sale_value
    ->     WHERE cust_id=OLD.cust_id;
    -> END //
Query OK, 0 rows affected (0.022 sec)

MariaDB [test]> DELIMITER ;

MariaDB [test]> delete from sales where sales_id = 3;
Query OK, 1 row affected (0.008 sec)

MariaDB [test]> select * from sales;
+----------+---------+--------------+------------+
| sales_id | cust_id | product_name | sale_value |
+----------+---------+--------------+------------+
|        1 |       1 | printer      |    3500.00 |
|        2 |       1 | Page Bundle  |     550.00 |
+----------+---------+--------------+------------+
2 rows in set (0.000 sec)

MariaDB [test]> select * from customer_sales_total;
+---------+------------+
| cust_id | sale_value |
+---------+------------+
|       1 |    4050.00 |
|       2 |       0.00 |
+---------+------------+
```

****

===

2. Q(example 11.4) Wirte a program to create trigger signal to restrict entering negative value
   in balance.

===

```
MariaDB [test]> DELIMITER //
MariaDB [test]> CREATE TRIGGER account_balance_bu BEFORE UPDATE ON account_balance
    -> FOR EACH ROW
    -> BEGIN
```

```
   ->     IF (NEW.balance < 0) THEN
   ->        SIGNAL SQLSTATE '80000'
   ->        SET MESSAGE_TEXT='Account balance cannot be less than 0';
   ->     END IF;
   -> END //
Query OK, 0 rows affected (0.028 sec)

MariaDB [test]> DELIMITER ;



MariaDB [test]> insert into account_balance(balance) values(10000)
   -> ,(23000),
   -> (45000);
Query OK, 3 rows affected (0.005 sec)
Records: 3  Duplicates: 0  Warnings: 0



MariaDB [test]> select * from account_balance;
+--------+----------+
| acc_id | balance  |
+--------+----------+
|      1 | 10000.00 |
|      2 | 23000.00 |
|      3 | 45000.00 |
+--------+----------+
3 rows in set (0.000 sec)



MariaDB [test]> update account_balance set balance = -2000 where acc_id = 2;
ERROR 1644 (80000): Account balance cannot be less than 0
```

****

===

3. Q(example 11.5) Write a trigger to perform data validation using select statement.

===

```
MariaDB [test]> DELIMITER //

MariaDB [test]> CREATE TRIGGER account_balance_bu  BEFORE UPDATE ON account_balance FOR EACH ROW

    -> BEGIN

    ->     DECLARE dummy INT;

    ->     IF NEW.balance<0 THEN

    ->         SELECT `Account balance cannot be less than 0` INTO dummy

    ->         FROM account_balance WHERE acc_id=NEW.acc_id;

    ->     END IF;

    -> END //

Query OK, 0 rows affected (0.024 sec)


MariaDB [test]>

MariaDB [test]> DELIMITER ;

MariaDB [test]> update account_balance set balance = -6000 where acc_id = 3;

ERROR 1054 (42S22): Unknown column 'Account balance cannot be less than 0' in 'field list'
```

****

===
=============

4. Q(figure 2.17) :write a example to create a sales table whichprovides free shipping on orders
   above 500

===
=============

```
MariaDB [test]> DELIMITER //
MariaDB [test]> CREATE TRIGGER sales_bi_trg1 BEFORE INSERT ON sales1
    -> FOR EACH ROW
    -> BEGIN
    ->     IF NEW.sale_value>500 THEN
    ->         SET NEW.free_shipping='Y';
    ->     ELSE
    ->         SET NEW.free_shipping='N';
    ->     END IF;
    ->         IF NEW.sale_value>1000 THEN
    ->             SET NEW.discount=NEW.sale_value*0.5;
    ->         ELSE
    ->             SET NEW.discount=0;
    ->     END IF;
    -> END //
Query OK, 0 rows affected (0.025 sec)

MariaDB [test]> DELIMITER ;

MariaDB [test]> INSERT INTO sales1(customer_id,sale_value,free_shipping,discount)
VALUES(201,20000,'N',0);
Query OK, 1 row affected (0.008 sec)
```

```
MariaDB [test]> select * from sales1;
+----------+-------------+------------+---------------+----------+
| sales_id | customer_id | sale_value | free_shipping | discount |
+----------+-------------+------------+---------------+----------+
|        1 |         201 |      20000 | Y             |    10000 |
+----------+-------------+------------+---------------+----------+
1 row in set (0.000 sec)
```

===

TOPIC : Transaction

===

5. Q(example 8.1) : Create a procedure to commence a transaction using auto commit.

===

```
MariaDB [test]> DELIMITER //
MariaDB [test]> CREATE PROCEDURE transfer_funds (from_account int, to_account int,transfer_amount decimal(10,2))
    -> BEGIN
    ->     SET autocommit=0;
    ->     UPDATE ACCOUNTS SET amount_balance = amount_balance - transfer_amount WHERE acc_id=from_account;
    ->     UPDATE ACCOUNTS SET amount_balance = amount_balance + transfer_amount WHERE acc_id=to_account;
    ->     COMMIT;
    -> END //
```

```
Query OK, 0 rows affected (1.759 sec)


MariaDB [test]> DELIMITER ;

MariaDB [test]> insert into accounts(branch_name,amount_balance)
values('Navsari',43900),('Surat',23090),('Ahmedabad',60897);

Query OK, 3 rows affected (0.008 sec)

Records: 3  Duplicates: 0  Warnings: 0


MariaDB [test]> select * from accounts;

+--------+-------------+----------------+

| acc_id | branch_name | amount_balance |

+--------+-------------+----------------+

|      1 | Navsari     |       43900.00 |

|      2 | Surat       |       23090.00 |

|      3 | Ahmedabad   |       60897.00 |

+--------+-------------+----------------+

3 rows in set (0.000 sec)



MariaDB [test]> call transfer_funds(3,1,4500);

Query OK, 2 rows affected (0.007 sec)



MariaDB [test]> select * from accounts;

+--------+-------------+----------------+

| acc_id | branch_name | amount_balance |

+--------+-------------+----------------+

|      1 | Navsari     |       48400.00 |

|      2 | Surat       |       23090.00 |

|      3 | Ahmedabad   |       56397.00 |

+--------+-------------+----------------+
3 rows in set (0.000 sec)
```

===

6. Q(example 8.2) : Create a procedure to commence a transaction using start transaction.

===

```
MariaDB [test]> DELIMITER //
MariaDB [test]> CREATE PROCEDURE trans_tfer_funds(from_account int, to_account int,tfer_amount decimal(10,2))
    -> BEGIN
    ->     START TRANSACTION;
    ->         UPDATE ACCOUNTS SET amount_balance =amount_balance - tfer_amount WHERE acc_id=from_account;
    ->         UPDATE ACCOUNTS SET amount_balance =amount_balance + tfer_amount WHERE acc_id=to_account;
    ->     COMMIT;
    -> END //
Query OK, 0 rows affected (0.021 sec)

MariaDB [test]> DELIMITER ;
MariaDB [test]> select * from accounts;
+--------+-------------+----------------+
| acc_id | branch_name | amount_balance |
+--------+-------------+----------------+
|      1 | Navsari     |       48400.00 |
|      2 | Surat       |       23090.00 |
|      3 | Ahmedabad   |       56397.00 |
+--------+-------------+----------------+
3 rows in set (0.000 sec)



MariaDB [test]> call transfer_funds(2,3,3000);
Query OK, 2 rows affected (0.007 sec)



MariaDB [test]> select * from accounts;
+--------+-------------+----------------+
| acc_id | branch_name | amount_balance |
+--------+-------------+----------------+
|      1 | Navsari     |       48400.00 |
|      2 | Surat       |       20090.00 |
|      3 | Ahmedabad   |       59397.00 |
+--------+-------------+----------------+
3 rows in set (0.000 sec)
```

==

7. Q(example 8.3) : create a procedure which displays use of Savepoint with a transaction

==

```
DELIMITER //
create procedure creating_table()
BEGIN
   create table location(location varchar(20),address1 varchar(20),address2 varchar(20),zipcode int);
   create table AUDIT_LOG (audit_message varchar(20));
   create table departments(department_name varchar(20),location varchar(20),manager_id int);
```

```
END //


MariaDB [test]> CREATE PROCEDURE savepoint_example(in_department_name
VARCHAR(30),in_location VARCHAR(30),in_address1 VARCHAR(30),in_address2
VARCHAR(30),in_zipcode VARCHAR(10), in_manager_id INT)
    ->    BEGIN
    ->    DECLARE location_exists INT DEFAULT 0;
    ->    DECLARE duplicate_dept INT DEFAULT 0;
    ->    START TRANSACTION;
    ->       SELECT COUNT(*) INTO location_exists FROM location WHERE location=in_location;
    ->       IF location_exists=0 THEN
    ->          INSERT INTO AUDIT_LOG (audit_message) VALUES (CONCAT('Creating new location
',in_location));
    ->          INSERT INTO location (location,address1,address2,zipcode) VALUES
(in_location,in_address1,in_address2,in_zipcode);
    ->       ELSE
    ->          UPDATE location set address1=in_address1, address2=in_address2, zipcode=in_zipcode
WHERE location=in_location;
    ->       END IF;
    ->    SAVEPOINT savepoint_location_exists;
    ->    BEGIN
    ->    DECLARE DUPLICATE_KEY CONDITION FOR 1062;
    ->    DECLARE CONTINUE HANDLER FOR DUPLICATE_KEY /*Duplicate key value*/
    ->    BEGIN
    ->    SET duplicate_dept=1;
    ->    ROLLBACK TO SAVEPOINT savepoint_location_exists;
    ->    END;
    ->    INSERT INTO AUDIT_LOG (audit_message) VALUES (CONCAT('Creating new
department',in_department_name));
    ->    INSERT INTO DEPARTMENTS (department_name,location,manager_id) VALUES
(in_department_name,in_location, in_manager_id);
    ->       IF duplicate_dept=1 THEN
    ->          UPDATE departments SET location=in_location,manager_id=in_manager_id WHERE
department_name=in_department_name;
```

```
    ->          END IF;
    ->          END;
    ->      COMMIT;
    -> END //
Query OK, 0 rows affected (0.022 sec)

MariaDB [test]> DELIMITER ;

MariaDB [test]>  CALL savepoint_example('Designing','Navsari','Grid road','Bhagvati Sankul
Society',396445,1);
Query OK, 5 rows affected (0.007 sec)

MariaDB [test]> select * from location;
+----------+-----------+----------------------+---------+
| location | address1  | address2             | zipcode |
+----------+-----------+----------------------+---------+
| Navsari  | Grid road | Bhagvati Sankul Soci |  396445 |
+----------+-----------+----------------------+---------+
1 row in set (0.000 sec)

MariaDB [test]> select * from AUDIT_LOG;
+-----------------------------------+
| audit_message                     |
+-----------------------------------+
| Creating new location Navsari     |
| Creating new department Designing |
+-----------------------------------+
2 rows in set (0.002 sec)

MariaDB [test]> select * from departments;
+-----------------+----------+------------+
| department_name | location | manager_id |
+-----------------+----------+------------+
| Designing       | Navsari  |          1 |
+-----------------+----------+------------+
1 row in set (0.000 sec)
```

===

TOPIC : Triggers (DO IT YOURSELF)

===

1. Write a Trigger that stores the old data table of student table in student_backup while updating the student table.

   Student_backup (Stud_ID, Stud_name, Address, Contact_no, Branch, Operation_date)

   Student (Stud_ID, Stud_name, Address, Contact_no, Branch)

===

```
MariaDB [test]> DELIMITER //
MariaDB [test]> CREATE PROCEDURE creating_table1()
    -> BEGIN
    ->     CREATE TABLE Student(Stud_ID INT PRIMARY KEY, Stud_name VARCHAR(20), Address
VARCHAR(30), Contact_no INT(11), Branch VARCHAR(60));
    ->     CREATE TABLE Student_backup(Stud_ID INT PRIMARY KEY, Stud_name VARCHAR(20),
Address VARCHAR(30), Contact_no INT(11), Branch VARCHAR(60),Operation_date date);
    -> END //
Query OK, 0 rows affected (0.009 sec)
```

```
MariaDB [test]> call creating_table1() //

Query OK, 0 rows affected (0.291 sec)



MariaDB [test]> CREATE TRIGGER stud_backup BEFORE UPDATE ON student FOR EACH ROW

    -> BEGIN

    ->     INSERT INTO Student_backup
values(OLD.Stud_ID,OLD.Stud_name,OLD.Address,OLD.Contact_no,OLD.Branch,curdate());

    -> END //

Query OK, 0 rows affected (0.028 sec)



MariaDB [test]> CREATE PROCEDURE insert_table1()

    -> BEGIN

    ->     insert into Student(Stud_name,Address,Contact_no,Branch)
values('Pradip','Navsari',8882228888,'Kabilpore'),

    ->     ('Ajinkya','Kutch',8881118888,'Gandhidham'),

    ->     ('Milind','Ahmedabad',8268228888,'Kalupur'),

    ->     ('Lakshya','Kutch',9888221358,'Gandhidham'),

    ->     ('Nirav','Kutch',8892220088,'Gandhidham');

    -> END //

Query OK, 0 rows affected (0.020 sec)


MariaDB [test]> DELIMITER ;

MariaDB [test]> call insert_table1();

Query OK, 5 rows affected, 5 warnings (0.007 sec)


MariaDB [test]> UPDATE Student SET Contact_no = 8238118848 WHERE Stud_ID=1;

Query OK, 0 rows affected, 1 warning (0.010 sec)

Rows matched: 1  Changed: 0  Warnings: 1
```

```
MariaDB [test]> select * from students;
ERROR 1146 (42S02): Table 'test.students' doesn't exist
MariaDB [test]> select * from student;
+---------+-----------+-----------+------------+------------+
| Stud_ID | Stud_name | Address   | Contact_no | Branch     |
+---------+-----------+-----------+------------+------------+
|       1 | Pradip    | Navsari   | 2147483647 | Kabilpore  |
|       2 | Ajinkya   | Kutch     | 2147483647 | Gandhidham |
|       3 | Milind    | Ahmedabad | 2147483647 | Kalupur    |
|       4 | Lakshya   | Kutch     | 2147483647 | Gandhidham |
|       5 | Nirav     | Kutch     | 2147483647 | Gandhidham |
+---------+-----------+-----------+------------+------------+
5 rows in set (0.000 sec)


MariaDB [test]> select * from student_backup;
+---------+-----------+---------+------------+-----------+----------------+
| Stud_ID | Stud_name | Address | Contact_no | Branch    | Operation_date |
+---------+-----------+---------+------------+-----------+----------------+
|       1 | Pradip    | Navsari | 2147483647 | Kabilpore | 2020-05-06     |
+---------+-----------+---------+------------+-----------+----------------+
1 row in set (0.000 sec)
```

==

2. Write a trigger, that ensures the empno of emp table is in a format ‘E00001’ (empno must start with ‘E’ and must be 6 characters long). If not, than complete empno with this format before inserting into the employee table.

===

```
MariaDB [test]> DELIMITER //
MariaDB [test]> CREATE TRIGGER emp_format BEFORE INSERT ON emp_tr1 FOR EACH ROW
    -> BEGIN
    ->     DECLARE I INT DEFAULT 1;
    ->     DECLARE CH INT;
    ->     DECLARE LEN INT;
    ->     DECLARE FLAG INT DEFAULT 0;
    ->     DECLARE EMP_ID VARCHAR(10);
    ->     SET EMP_ID=NEW.empid;
    ->     SET LEN=LENGTH(NEW.empid);
    ->     IF (LEN<6) THEN
    ->         SIGNAL SQLSTATE '80000'
    ->         SET MESSAGE_TEXT='EMPLOYEE ID MUST BE 6 CHARACTER LONG';
    ->     ELSEIF (LEN>6) THEN
    ->         SIGNAL SQLSTATE '80001'
    ->         SET MESSAGE_TEXT='EMPLOYEE ID MUST BE 6 CHARACTER LONG';
    ->     ELSE
    ->         SET CH=ASCII(SUBSTR(EMP_ID,I,1));
    ->         IF (CH=69) THEN
    ->             SET I=I+1;
    ->             MYLOOP : WHILE (I<LEN) DO
    ->                 SET CH=ASCII(SUBSTR(EMP_ID,I,1));
    ->                 IF (CH>=48 AND CH<=57) THEN
    ->                     SET I=I+1;
    ->                 ELSE
    ->                     SET FLAG=1;
    ->                     LEAVE MYLOOP;
    ->               END IF;
    ->            END WHILE;
    ->         ELSE
    ->            SIGNAL SQLSTATE '80002'
    ->            SET MESSAGE_TEXT='EMPLOYEE ID MUST BE LIKE E00001';
    ->         END IF ;
    ->     END IF;
    ->     IF (FLAG=1) THEN
    ->         SIGNAL SQLSTATE '80003'
    ->         SET MESSAGE_TEXT='EMPLOYEE ID MUST BE LIKE E00001';
    ->     END IF;
    -> END //
Query OK, 0 rows affected (0.032 sec)

MariaDB [test]> DELIMITER ;

MariaDB [test]> insert into emp_tr1 values(1,'pradip');
ERROR 1644 (80000): EMPLOYEE ID MUST BE 6 CHARACTER LONG

MariaDB [test]> insert into emp_tr1 values(123456,'pradip');
ERROR 1644 (80002): EMPLOYEE ID MUST BE LIKE E00001

MariaDB [test]> insert into emp_tr1 values('E00001','pradip');
Query OK, 1 row affected (0.007 sec)

MariaDB [test]> select * from emp_tr1;
+--------+--------+
| empid  | name   |
+--------+--------+
| E00001 | pradip |
+--------+--------+
1 row in set (0.000 sec)
```

==
=============

3. Write a trigger which checks the age of employee while inserting the record in emp table.
   If it is negative than generate the error and display proper message.

==
=============

```
MariaDB [test]> DELIMITER //
MariaDB [test]>
MariaDB [test]> CREATE TRIGGER check_age BEFORE INSERT ON emp_tr1
    -> FOR EACH ROW
    -> BEGIN
    -> DECLARE AGE INT;
    ->     SET AGE=YEAR(CURDATE())-YEAR(NEW.birth_day);
    ->     IF AGE<0 THEN
    ->         SIGNAL SQLSTATE '80005'
    ->         SET MESSAGE_TEXT='Please Enter Valid BirthDay';
    ->     END IF;
    -> END //
Query OK, 0 rows affected (0.020 sec)

MariaDB [test]> DELIMITER ;

MariaDB [test]> insert into emp_tr1 values('E00002','Nirav','2021-06-23');
ERROR 1644 (80005): Please Enter Valid BirthDay

MariaDB [test]> insert into emp_tr1 values('E00002','Nirav','1999-06-23');
```

```
Query OK, 1 row affected (0.007 sec)

MariaDB [test]> select * from emp_tr1;
+--------+--------+------------+
| empid  | name   | birth_day  |
+--------+--------+------------+
| E00001 | pradip | 1998-04-25 |
| E00002 | Nirav  | 1999-06-23 |
+--------+--------+------------+
2 rows in set (0.000 sec)
```

===

4. Write a trigger which converts the employee name in upper case if it is inserted in any other
   case. Change should be done before the insertion only.

===

```
MariaDB [test]> DELIMITER //
MariaDB [test]> CREATE TRIGGER uppercase_name BEFORE INSERT ON emp_tr1 FOR EACH ROW
    ->     BEGIN
    ->     DECLARE I INT DEFAULT 1;
    ->     DECLARE NAME VARCHAR(20) default ' ';
    ->     DECLARE STRING VARCHAR(20) DEFAULT " ";
    ->     DECLARE RES VARCHAR(2) DEFAULT "";
    ->     DECLARE CH INT;
    ->     DECLARE LEN INT;
    ->     SET NAME=NEW.name;
    ->     SET LEN=LENGTH(NAME);
    ->    WHILE (I<=LEN) do
    ->       SET CH=ASCII(SUBSTR(NAME,I,1));
    ->       IF (CH>=97 AND CH<=122) THEN
    ->          SET CH=CH-32;
    ->       END IF;
    ->          SET RES=CHAR(CH);
    ->          SET STRING=CONCAT(STRING,RES);
    ->          SET I=I+1;
    ->    END WHILE;
    ->    set new.name=string;
    -> END //
Query OK, 0 rows affected (0.020 sec)

MariaDB [test]> DELIMITER ;
MariaDB [test]> select * from emp_tr1;
+--------+--------+------------+
| empid  | name   | birth_day  |
+--------+--------+------------+
| E00001 | pradip | 1998-04-25 |
| E00002 | Nirav  | 1999-06-23 |
+--------+--------+------------+
2 rows in set (0.001 sec)

MariaDB [test]> insert into emp_tr1 values('E00002','ajinkya','1999-01-26');
Query OK, 1 row affected (0.011 sec)

MariaDB [test]> select * from emp_tr1;
+--------+----------+------------+
| empid  | name     | birth_day  |
+--------+----------+------------+
| E00001 | pradip   | 1998-04-25 |
| E00002 | Nirav    | 1999-06-23 |
| E00002 |  AJINKYA | 1999-01-26 |
+--------+----------+------------+
3 rows in set (0.000 sec)
```

===
=============

5. WAT that stores the data of emp table in emp_backup table for every delete operation and
  store the old data for every update operation.

  EMP(Empno, Empname, salary);

  Emp_Backup(Empno,Empname,Date_of_operation,Type_of_operation (i.e.update or delete));

===
=============

```
MariaDB [test]> DELIMITER //

MariaDB [test]> CREATE TRIGGER emp_bu BEFORE UPDATE ON EMP1  FOR EACH ROW
    -> BEGIN
    ->     INSERT INTO Emp_Backup(Empno,Empname,Date_of_operation,Type_of_operation) values
(NEW.Empno,NEW.Empname,CURDATE(),'Update');
    -> END //
Query OK, 0 rows affected (0.020 sec)

MariaDB [test]> CREATE TRIGGER emp_bd BEFORE DELETE ON EMP1  FOR EACH ROW
    -> BEGIN
    ->     INSERT INTO Emp_Backup(Empno,Empname,Date_of_operation,Type_of_operation) values
(old.Empno,old.Empname,CURDATE(),'Delete');
    -> END //
```

```
Query OK, 0 rows affected (0.022 sec)

MariaDB [test]> DELIMITER ;
MariaDB [test]> insert into EMP1 values(1,'Pradip',50000);
Query OK, 1 row affected (0.008 sec)

MariaDB [test]> select * from EMP1;
+-------+---------+--------+
| Empno | Empname | salary |
+-------+---------+--------+
|     1 | Pradip  |  50000 |
+-------+---------+--------+
1 row in set (0.002 sec)

MariaDB [test]> update EMP1 set salary = 60000 where Empno = 1;
Query OK, 1 row affected (0.008 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [test]> select * from EMP1;
+-------+---------+--------+
| Empno | Empname | salary |
+-------+---------+--------+
|     1 | Pradip  |  60000 |
+-------+---------+--------+
1 row in set (0.000 sec)

MariaDB [test]> delete from EMP1 where Empno = 1;
Query OK, 1 row affected (0.007 sec)

MariaDB [test]> select * from emp_backup;
+-------+---------+------------------+------------------+
| Empno | Empname | Date_of_operation | Type_of_operation |
+-------+---------+-------------------+-------------------+
|     1 | Pradip  | 2020-05-06        | Update            |
|     1 | Pradip  | 2020-05-06        | Delete            |
+-------+---------+-------------------+-------------------+
2 rows in set (0.000 sec)
```

===

6. WAT which display the message ‘Updating’,’Deleting’ or ’Inserting’ when Update, Delete or
   Insert operation is performed on the emp table respectively.

===

```
MariaDB [test]> DELIMITER //
MariaDB [test]> CREATE TRIGGER emp_insert BEFORE INSERT ON emp FOR EACH ROW
    -> BEGIN
    ->     SIGNAL SQLSTATE '80000'
    ->     SET MESSAGE_TEXT='Inserting An EMP.';
    -> END //
Query OK, 0 rows affected (0.023 sec)

MariaDB [test]>
MariaDB [test]> INSERT INTO emp(empname,position,salary)
VALUES('Shubham','CEO_TENCENT',70000) //
ERROR 1644 (80000): Inserting An EMP.
MariaDB [test]>
MariaDB [test]> CREATE TRIGGER emp_update BEFORE UPDATE ON emp  FOR EACH ROW
    -> BEGIN
    ->     SIGNAL SQLSTATE '80001'
    ->     SET MESSAGE_TEXT='Updating An EMP.';
    ->
    -> END //
Query OK, 0 rows affected (0.045 sec)

MariaDB [test]>
MariaDB [test]> UPDATE emp SET salary = '75000' WHERE emp_id = 6 //
ERROR 1644 (80001): Updating An EMP.
MariaDB [test]>
MariaDB [test]> CREATE TRIGGER emp_delete BEFORE DELETE ON emp  FOR EACH ROW
    -> BEGIN
    ->     SIGNAL SQLSTATE '80002'
    ->     SET MESSAGE_TEXT='Deleting An EMP.';
    -> END //
Query OK, 0 rows affected (0.019 sec)

MariaDB [test]> DELETE from emp where empname = 'Lakshya' //
ERROR 1644 (80002): Deleting An EMP.
MariaDB [test]>
MariaDB [test]> DELIMITER ;
```

===

7. WAT which generate an error if any user try to delete from product_master table on weekends

===

```
MariaDB [test]> DELIMITER //
MariaDB [test]> CREATE TRIGGER emp_day_bd BEFORE DELETE ON product_master
    -> FOR EACH ROW
    -> BEGIN
    -> DECLARE DAY VARCHAR(20);
    ->     SET DAY=DAYNAME(curdate());
    ->     IF DAY='Thursday' THEN
    ->         SIGNAL SQLSTATE '80007'
    ->         SET MESSAGE_TEXT='Deletion is not possible on Thursday.';
    ->     END IF;
    -> END //
Query OK, 0 rows affected (0.291 sec)

MariaDB [test]> select * from product_master;
+------------+--------------+
| product_id | product_name |
+------------+--------------+
|          1 | TV           |
|          2 | LAPTOP       |
|          3 | FRIDGE       |
+------------+--------------+
3 rows in set (0.000 sec)

MariaDB [test]> delete from product_master where product_id='3';
ERROR 1644 (80007): Deletion is not possible on Thursday.
```

==
=============

8. We have two tables student_mast and stu_log. student_mast have three columns

STUDENT_ID, NAME, ST_CLASS. stu_log table has two columns user_id and description.

WAT which inserts the student details in stu_log table as soon as we promote the students in student master table(e.g. when a student is promoted from sem 2 to 3, auto entry in log table)

==

```
MariaDB [test]> DELIMITER //
MariaDB [test]> CREATE TRIGGER stu_log BEFORE UPDATE ON student_mast
    -> FOR EACH ROW
    -> BEGIN
    ->     DECLARE DES VARCHAR(100) DEFAULT ' ';
    ->     DECLARE SID INT;
    ->     DECLARE SEM_NEW INT;
    ->     DECLARE SEM_OLD INT;
    ->     SET SEM_OLD=OLD.CLASS;
    ->     SET SEM_NEW =SEM_OLD +1;
    ->     SET DES= CONCAT('Student is promoted from semister ',SEM_OLD,' to ',SEM_NEW,DES);
    ->     SET SID=OLD.student_id;
    ->     INSERT INTO stu_log VALUES(SID,DES);
    -> END //
Query OK, 0 rows affected (0.026 sec)

MariaDB [test]> DELIMITER ;


MariaDB [test]> insert into student_mast(name,class) value('pradip',10),
    -> ('Ajinkya',9);
Query OK, 2 rows affected (0.004 sec)
```

```
Records: 2  Duplicates: 0  Warnings: 0



MariaDB [test]> SELECT * FROM student_mast;
+------------+---------+-------+
| student_id | name    | class |
+------------+---------+-------+
|          1 | pradip  |    10 |
|          2 | Ajinkya |     9 |
+------------+---------+-------+
2 rows in set (0.000 sec)



MariaDB [test]> UPDATE student_mast SET class=class + 1 WHERE student_id = 1;
Query OK, 1 row affected (0.007 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [test]> SELECT * FROM student_mast;
+------------+---------+-------+
| student_id | name    | class |
+------------+---------+-------+
|          1 | pradip  |    11 |
|          2 | Ajinkya |     9 |
+------------+---------+-------+
2 rows in set (0.000 sec)

MariaDB [test]> SELECT * FROM stu_log;
+---------+------------------------------------------+
| user_id | description                              |
+---------+------------------------------------------+
|       1 | Student is promoted from semister 10 to 11 |
+---------+-------------------------------------------+
1 row in set (0.000 sec)
```

===

9. WAT to calculate the Income Tax amount and insert it in emp table. EMP(emp_no,emp_name, emp_income, income_tax);

If emp_income <100000 and >=50000 then incometax = 10%

If emp_income <200000 and >=100000 then incometax = 15%

If emp_income <300000 and >=200000 then incometax = 20%

===

```
MariaDB [test]> DELIMITER //
MariaDB [test]> drop trigger income_tax_decide //
Query OK, 0 rows affected (0.000 sec)

MariaDB [test]> CREATE TRIGGER income_tax_decide BEFORE INSERT ON emp3
    -> FOR EACH ROW
    -> BEGIN
    -> DECLARE tax FLOAT;
    ->     IF (NEW.emp_income >= 50000 AND NEW.emp_income < 100000) THEN
    ->         set tax = (NEW.emp_income*10)/100;
    ->         set NEW.income_tax = tax;
    ->     ELSEIF (NEW.emp_income >= 100000 AND NEW.emp_income < 200000) THEN
    ->         set tax = (NEW.emp_income*15)/100;
    ->         set NEW.income_tax = tax;
    ->     ELSEIF (NEW.emp_income >= 200000 AND NEW.emp_income < 300000) THEN
    ->        set tax = (NEW.emp_income*20)/100;
    ->        set NEW.income_tax = tax;
    ->    END IF;
    -> END //
Query OK, 0 rows affected (0.019 sec)

MariaDB [test]> DELIMITER ;
MariaDB [test]>
MariaDB [test]> insert into emp3(emp_name,emp_income,income_tax) values('pradip',80000,0);
Query OK, 1 row affected (0.006 sec)

MariaDB [test]>
MariaDB [test]>
MariaDB [test]> SELECT * FROM EMP3;
+--------+----------+------------+------------+
| emp_no | emp_name | emp_income | income_tax |
+--------+----------+------------+------------+
|      8 | pradip   |      80000 |          0 |
|      9 | pradip   |      80000 |          0 |
|     10 | pradip   |      80000 |          0 |
|     11 | pradip   |      80000 |          0 |
|     12 | pradip   |      15000 |          0 |
|     13 | pradip   |      80000 |       8000 |
+--------+----------+------------+------------+
6 rows in set (0.000 sec)

MariaDB [test]> insert into emp3(emp_name,emp_income,income_tax) values('pradip',80000,0);
Query OK, 1 row affected (0.004 sec)

MariaDB [test]> SELECT * FROM EMP3;
+--------+----------+------------+------------+
| emp_no | emp_name | emp_income | income_tax |
+--------+----------+------------+------------+
|      1 | pradip   |      80000 |       8000 |
+--------+----------+------------+------------+
1 row in set (0.000 sec)


MariaDB [test]> insert into emp3(emp_name,emp_income,income_tax) values('Ajinkya',190000,0);
Query OK, 1 row affected (0.004 sec)

MariaDB [test]> SELECT * FROM EMP3;
+--------+----------+------------+------------+
| emp_no | emp_name | emp_income | income_tax |
+--------+----------+------------+------------+
|      1 | pradip   |      80000 |       8000 |
|      2 | Ajinkya  |     190000 |      28500 |
+--------+----------+------------+------------+
2 rows in set (0.000 sec)
```