LAB ASSIGNMENT NO : 05

Q1, Write procedure to print natural numbers from 1 to 10 using do-While control flow statement.

```
mysql> create database proceduredb ;
Query OK, 1 row affected (0.02 sec)

mysql> use proceduredb;
Database changed
mysql> DELIMITER //
mysql> CREATE PROCEDURE PrintNaturalNumbers()
    -> BEGIN
    ->     DECLARE counter INT DEFAULT 1;
    ->     DECLARE result VARCHAR(255) DEFAULT '';
    ->     WHILE counter <= 10 DO
    ->         -- Concatenate the current natural number to the result
    ->         SET result = CONCAT(result, counter, ':');
    ->         SET counter = counter + 1;
    ->     END WHILE;
    ->     -- Print the result without trailing colon
    ->     SELECT result AS Result;
    -> END //
Query OK, 0 rows affected (0.02 sec)

mysql> DELIMITER ;
mysql> CALL PrintNaturalNumbers();
+----------------------+
| Result               |
+----------------------+
| 1:2:3:4:5:6:7:8:9:10: |
+----------------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.02 sec)
```

Q2, Write an Sql program in Procedure print even numbers up to 10 using- loop – flow control statement.

```
mysql> DELIMITER //
mysql> CREATE PROCEDURE PrintEvenNumbersUpTo10()
    -> BEGIN
    ->     DECLARE counter INT DEFAULT 2;
    ->     DECLARE result VARCHAR(255) DEFAULT '';
    ->
    ->     WHILE counter <= 10 DO
    ->         -- Concatenate the current even number to the result
    ->         SET result = CONCAT(result, IF(result = '', '', ','), counter);
    ->         -- Increment counter by 2 to get the next even number
    ->         SET counter = counter + 2;
    ->     END WHILE;
    -> SELECT result AS Result;
    -> END //
Query OK, 0 rows affected (0.02 sec)

mysql> DELIMITER ;
mysql> CALL PrintEvenNumbersUpTo10();
+------------+
| Result     |
+------------+
| 2,4,6,8,10 |
+------------+
1 row in set (0.00 sec)

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

Q3, a) Create a student table with RollNo, name, Course, Mark.

b) Create a Trigger for INSERT operation on STUDENT table to ensure that Mark field never goes below 20 and above 100. If Mark is less than 20 then change it to 20,similarly if mark is greater than 100 change it to 100.

c) Create a Trigger for UPDATE operations on STUDENT table to ensure that Mark field never goes below 20 and above 100. If Mark is less than 20 then change it to 20,similarly if mark is greater than 100 change it to 100.

```
mysql> CREATE TABLE student (
    ->     RollNo INT PRIMARY KEY,
    ->     name VARCHAR(255),
    ->     Course VARCHAR(255),
    ->     Mark INT
    -> );
Query OK, 0 rows affected (0.08 sec)

mysql> DELIMITER //
mysql> CREATE TRIGGER before_insert_student
    -> BEFORE INSERT ON student
    -> FOR EACH ROW
    -> BEGIN
    ->     -- Ensure Mark is between 20 and 100
    ->     IF NEW.Mark < 20 THEN
    ->         SET NEW.Mark = 20;
    ->     ELSEIF NEW.Mark > 100 THEN
    ->         SET NEW.Mark = 100;
    ->     END IF;
    -> END;
    -> //
Query OK, 0 rows affected (0.03 sec)

mysql> DELIMITER ;
mysql> DELIMITER //
mysql> CREATE TRIGGER before_update_student
    -> BEFORE UPDATE ON student
    -> FOR EACH ROW
    -> BEGIN
    ->     -- Ensure Mark is between 20 and 100
    ->     IF NEW.Mark < 20 THEN
    ->         SET NEW.Mark = 20;
    ->     ELSEIF NEW.Mark > 100 THEN
    ->         SET NEW.Mark = 100;
    ->     END IF;
    -> END;
    -> //
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> DELIMITER ;
mysql> insert into student values(1, 'Nisha', 'java', 55),(2, 'Ann', 'c++', 70);
Query OK, 2 rows affected (0.02 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> insert into student values(3, 'Remya', 'DBMS', 45);
Query OK, 1 row affected (0.01 sec)

mysql> select * from student;
+--------+-------+--------+------+
| RollNo | name  | Course | Mark |
+--------+-------+--------+------+
|      1 | Nisha | java   |   55 |
|      2 | Ann   | c++    |   70 |
|      3 | Remya | DBMS   |   45 |
+--------+-------+--------+------+
3 rows in set (0.00 sec)

mysql> update student set marks=100 where name= 'Ann';
ERROR 1054 (42S22): Unknown column 'marks' in 'field list'
mysql> update student set mark=100 where name= 'Ann';
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> update student set mark= 250 where name ='Nisha';
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from student;
+--------+-------+--------+------+
| RollNo | name  | Course | Mark |
+--------+-------+--------+------+
|      1 | Nisha | java   |  100 |
|      2 | Ann   | c++    |  100 |
|      3 | Remya | DBMS   |   45 |
+--------+-------+--------+------+
3 rows in set (0.00 sec)
```

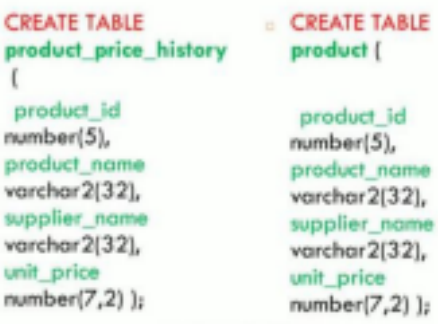
Q4, Create a row level trigger for the customers table that would fire for UPDATE operations performed on the CUSTOMERS table. This trigger will display the salary difference between the old values and new values if old salary is greater than new salary. The schema of customer table is given below.

CUSTOMERS (ID ,NAME,AGE.ADDRESS,SALARY)

```
mysql> CREATE TABLE customers (
    ->     ID INT PRIMARY KEY,
    ->     NAME VARCHAR(255),
    ->     AGE INT,
    ->     ADDRESS VARCHAR(255),
    ->     SALARY INT
    -> );
Query OK, 0 rows affected (0.07 sec)

mysql> insert into CUSTOMERS values(1,'Nisha',25,'Pune',80000),(2,'Ivan',24,'Mumbai',50000),(3,'Alice',29,'Delhi',5000);
Query OK, 3 rows affected (0.02 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> select * from CUSTOMERS;
+----+-------+------+---------+--------+
| ID | NAME  | AGE  | ADDRESS | SALARY |
+----+-------+------+---------+--------+
|  1 | Nisha |   25 | Pune    |  80000 |
|  2 | Ivan  |   24 | Mumbai  |  50000 |
|  3 | Alice |   29 | Delhi   |   5000 |
+----+-------+------+---------+--------+
3 rows in set (0.00 sec)
```

```
mysql> DELIMITER ;
mysql> DELIMITER //
mysql> CREATE TRIGGER salary_difference_trigger
    -> BEFORE UPDATE ON CUSTOMERS
    -> FOR EACH ROW
    -> BEGIN
    ->     DECLARE salary_difference INT;
    ->
    ->     IF OLD.SALARY > NEW.SALARY THEN
    ->         SET salary_difference = OLD.SALARY - NEW.SALARY;
    ->         SELECT CONCAT('Salary decreased by ', salary_difference) INTO @message;
    ->         SET @message = 'No salary decrease detected.';
    ->         -- Handle the exception by storing the message in a user-defined variable
    ->     END IF;
    -> END;
    -> //
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql> UPDATE CUSTOMERS SET SALARY = 55000 WHERE ID = 1;
Query OK, 1 row affected (0.03 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select @message;
+------------------------------+
| @message                     |
+------------------------------+
| No salary decrease detected. |
+------------------------------+
1 row in set (0.01 sec)
```

Q5,The price of a product changes constantly. It is important to maintain the history of the prices of the products.Two tables are created as below .Create a trigger to update the ‘product_price_history’ table when the price of the product is updated in the ‘product ‘ table.




```
mysql> create table product(product_id int, product_name varchar(32),
    -> supplier_name varchar(32), unit_price int)
    -> ;
Query OK, 0 rows affected (0.07 sec)

mysql> create table product_price_history(product_id int, product_name varchar(32),
    -> supplier_name varchar(32), unit_price int);
Query OK, 0 rows affected (0.11 sec)

mysql> DELIMITER //
mysql> CREATE TRIGGER update_price_history
    -> AFTER UPDATE ON product
    -> FOR EACH ROW
    -> BEGIN
    ->     INSERT INTO product_price_history (product_id, product_name, supplier_name, unit_price)
    ->     VALUES (OLD.product_id, OLD.product_name, OLD.supplier_name, OLD.unit_price);
    -> END //
Query OK, 0 rows affected (0.02 sec)

mysql> DELIMITER ;
mysql> insert into product values(1,'Laptop','Dell',60000),(2,'Keyboard','Hp',500),(3,'Pendrive','Sandisk',850);
Query OK, 3 rows affected (0.02 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> insert into product values(4,'Bag','Kitex',670),(5,'Chair','Nilkamal',490),(6,'Watch','Boat',6990);
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> select * from product;
+------------+--------------+---------------+------------+
| product_id | product_name | supplier_name | unit_price |
+------------+--------------+---------------+------------+
|          1 | Laptop       | Dell          |      60000 |
|          2 | Keyboard     | Hp            |        500 |
|          3 | Pendrive     | Sandisk       |        850 |
|          4 | Bag          | Kitex         |        670 |
|          5 | Chair        | Nilkamal      |        490 |
|          6 | Watch        | Boat          |       6990 |
+------------+--------------+---------------+------------+
6 rows in set (0.00 sec)

mysql> UPDATE product
    -> SET unit_price = 690
    -> WHERE product_id = 5;
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from product_price_history;
+------------+--------------+---------------+------------+
| product_id | product_name | supplier_name | unit_price |
+------------+--------------+---------------+------------+
|          5 | Chair        | Nilkamal      |        490 |
+------------+--------------+---------------+------------+
```

Q6 , Create a Procedure to read all records from Product table using a cursor. After fetching Done by one the rows, display Product Id and Product name of products with price > 500.

```
mysql> DELIMITER //
mysql> CREATE PROCEDURE fetch_products_with_price_above_500()
    -> BEGIN
    ->     DECLARE done INT DEFAULT 0;
    ->     DECLARE product_id_val INT;
    ->     DECLARE product_name_val VARCHAR(255);DECLARE product_cursor CURSOR FOR
    ->         SELECT product_id, product_name
    ->         FROM product
    ->         WHERE unit_price > 500; DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
    ->  OPEN product_cursor;
    -> FETCH_LOOP: LOOP
    -> FETCH product_cursor INTO product_id_val, product_name_val;
    -> IF done = 1 THEN
    ->             LEAVE FETCH_LOOP;
    ->         END IF;
    ->  SELECT CONCAT('Product Id: ', product_id_val, ', Product Name: ', product_name_val) AS Product_Info;
    ->
    ->     END LOOP;
    -> CLOSE product_cursor;
    ->
    -> END //
Query OK, 0 rows affected (0.02 sec)

mysql>
mysql> DELIMITER ;
```

```
mysql> CALL fetch_products_with_price_above_500;
+-----------------------------------+
| Product_Info                      |
+-----------------------------------+
| Product Id: 1, Product Name: Laptop |
+-----------------------------------+
1 row in set (0.01 sec)

+-------------------------------------+
| Product_Info                        |
+-------------------------------------+
| Product Id: 3, Product Name: Pendrive |
+-------------------------------------+
1 row in set (0.03 sec)

+---------------------------------+
| Product_Info                    |
+---------------------------------+
| Product Id: 4, Product Name: Bag |
+---------------------------------+
1 row in set (0.07 sec)

+-----------------------------------+
| Product_Info                      |
+-----------------------------------+
| Product Id: 5, Product Name: Chair |
+-----------------------------------+
1 row in set (0.09 sec)

+-----------------------------------+
| Product_Info                      |
+-----------------------------------+
| Product Id: 6, Product Name: Watch |
+-----------------------------------+
1 row in set (0.12 sec)

Query OK, 0 rows affected (0.14 sec)
```