PL EXERSISE 2

1. Select from any table a number and determine whether it is within a given range (for example, between 1 and 10)

```
mysql> CREATE TABLE numbers_table (number_value INT);
Query OK, 0 rows affected (0.05 sec)

mysql> INSERT INTO numbers_table (number_value)
   -> VALUES (5), (8), (12), (3), (15);
Query OK, 5 rows affected (0.00 sec)
Records: 5  Duplicates: 0  Warnings: 0

mysql> drop procedure check_range;
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER //
mysql>
mysql> CREATE PROCEDURE check_range()
   -> BEGIN
   ->    DECLARE v_number INT;
   ->    DECLARE done INT DEFAULT 0;
   ->
   ->    DECLARE cur CURSOR FOR SELECT number_value FROM numbers_table;
   ->    DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
   ->
   ->    OPEN cur;
   ->
   ->    read_loop: LOOP
   ->       FETCH cur INTO v_number;
   ->       IF done THEN
   ->          LEAVE read_loop;
   ->       END IF;
   ->
   ->       IF v_number BETWEEN 1 AND 10 THEN
   ->          SELECT v_number AS 'Number', 'Within range' AS 'Range Status';
   ->       ELSE
   ->          SELECT v_number AS 'Number', 'Out of range' AS 'Range Status';
   ->       END IF;
   ->    END LOOP;
   ->
   ->    CLOSE cur;
   -> END;
   -> //
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
mysql>
mysql> Call check_range();
+--------+--------------+
| Number | Range Status |
+--------+--------------+
|      5 | Within range |
+--------+--------------+
1 row in set (0.01 sec)
```

```
+--------+--------------+
| Number | Range Status |
+--------+--------------+
|      8 | Within range |
+--------+--------------+
1 row in set (0.02 sec)

+--------+--------------+
| Number | Range Status |
+--------+--------------+
|     12 | Out of range |
+--------+--------------+
1 row in set (0.03 sec)

+--------+--------------+
| Number | Range Status |
+--------+--------------+
|      3 | Within range |
+--------+--------------+
1 row in set (0.04 sec)

+--------+--------------+
| Number | Range Status |
+--------+--------------+
|     15 | Out of range |
+--------+--------------+
1 row in set (0.05 sec)

Query OK, 0 rows affected (0.06 sec)
```

2. Select from any table three positive integers representing the sides of a triangle, and determine whether they form a valid triangle. Hint: In a triangle, the sum of any two sides must always be greater than the third side.

```
mysql> CREATE TABLE triangle_sides (
    ->     side1 INT,
    ->     side2 INT,
    ->     side3 INT
    -> );
Query OK, 0 rows affected (0.05 sec)

mysql> INSERT INTO triangle_sides (side1, side2, side3)
    -> VALUES (3, 4, 5), (5, 10, 25), (7, 7, 7), (10, 2, 2);
Query OK, 4 rows affected (0.01 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> DELIMITER //
mysql>
mysql> CREATE PROCEDURE check_triangle()
    -> BEGIN
    ->     DECLARE s1 INT;
    ->     DECLARE s2 INT;
    ->     DECLARE s3 INT;
    ->     DECLARE done INT DEFAULT 0;
    ->
    ->     DECLARE cur CURSOR FOR SELECT side1, side2, side3 FROM triangle_sides;
```

```
    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
    ->
    ->     OPEN cur;
    ->
    ->     read_loop: LOOP
    ->         FETCH cur INTO s1, s2, s3;
    ->         IF done THEN
    ->             LEAVE read_loop;
    ->         END IF;
    ->
    ->         IF (s1 + s2 > s3) AND (s1 + s3 > s2) AND (s2 + s3 > s1) THEN
    ->             SELECT s1 AS side1, s2 AS side2, s3 AS side3, 'Valid Triangle' AS status;
    ->         ELSE
    ->             SELECT s1 AS side1, s2 AS side2, s3 AS side3, 'Invalid Triangle' AS status;
    ->         END IF;
    ->     END LOOP;
    ->
    ->     CLOSE cur;
    -> END;
    -> //
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
mysql>
mysql> CALL check_triangle();
+-------+-------+-------+----------------+
| side1 | side2 | side3 | status         |
+-------+-------+-------+----------------+
|     3 |     4 |     5 | Valid Triangle |
+-------+-------+-------+----------------+
1 row in set (0.00 sec)

+-------+-------+-------+------------------+
| side1 | side2 | side3 | status           |
+-------+-------+-------+------------------+
|     5 |    10 |    25 | Invalid Triangle |
+-------+-------+-------+------------------+
1 row in set (0.01 sec)

+-------+-------+-------+----------------+
| side1 | side2 | side3 | status         |
+-------+-------+-------+----------------+
|     7 |     7 |     7 | Valid Triangle |
+-------+-------+-------+----------------+
1 row in set (0.01 sec)

+-------+-------+-------+------------------+
| side1 | side2 | side3 | status           |
+-------+-------+-------+------------------+
|    10 |     2 |     2 | Invalid Triangle |
+-------+-------+-------+------------------+
1 row in set (0.03 sec)

Query OK, 0 rows affected (0.04 sec)
```

3. Check if a given a year is a leap year. The condition is:- year should be (divisible by 4 and not divisible by 100) or (divisible by 4 and divisible by 400.). The year should be Selected from some table.

```
mysql> CREATE TABLE years_table (
    ->     year INT
    -> );
Query OK, 0 rows affected (0.10 sec)

mysql> INSERT INTO years_table (year)
    -> VALUES (2020), (1900), (2000), (2023), (2400), (2100);
Query OK, 6 rows affected (0.00 sec)
Records: 6  Duplicates: 0  Warnings: 0

mysql> DELIMITER //
mysql>
mysql> CREATE PROCEDURE check_leap_year()
    -> BEGIN
    ->     DECLARE y INT;
    ->     DECLARE done INT DEFAULT 0;
    ->
    ->     DECLARE cur CURSOR FOR SELECT year FROM years_table;
    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
    ->
    ->     OPEN cur;
    ->
    ->     read_loop: LOOP
    ->         FETCH cur INTO y;
    ->         IF done THEN
    ->             LEAVE read_loop;
    ->         END IF;
    ->
    ->         IF ((y % 4 = 0 AND y % 100 != 0) OR (y % 400 = 0)) THEN
    ->             SELECT y AS Year, 'Leap Year' AS Status;
    ->         ELSE
    ->             SELECT y AS Year, 'Not a Leap Year' AS Status;
    ->         END IF;
    ->     END LOOP;
    ->
    ->     CLOSE cur;
    -> END;
    -> //
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
mysql>
mysql> CALL check_leap_year();
+------+-----------+
| Year | Status    |
+------+-----------+
| 2020 | Leap Year |
+------+-----------+
1 row in set (0.00 sec)

+------+-----------------+
```

```
| Year | Status         |
+------+-----------------+
| 1900 | Not a Leap Year |
+------+-----------------+
1 row in set (0.01 sec)

+------+-----------+
| Year | Status    |
+------+-----------+
| 2000 | Leap Year |
+------+-----------+
1 row in set (0.02 sec)

+------+-----------------+
| Year | Status         |
+------+-----------------+
| 2023 | Not a Leap Year |
+------+-----------------+
1 row in set (0.03 sec)

+------+-----------+
| Year | Status    |
+------+-----------+
| 2400 | Leap Year |
+------+-----------+
1 row in set (0.04 sec)

+------+-----------------+
| Year | Status         |
+------+-----------------+
| 2100 | Not a Leap Year |
+------+-----------------+
1 row in set (0.05 sec)

Query OK, 0 rows affected (0.06 sec)
```

4. Write a program that Selects from any table two character strings. Your program should then determine if one character string exists inside another character string

```
mysql> CREATE TABLE string_table (
    ->     str1 VARCHAR(100),
    ->     str2 VARCHAR(100)
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql> INSERT INTO string_table (str1, str2)
    -> VALUES
    -> ('Hello World', 'World'),
    -> ('MySQL Stored Procedure', 'Code'),
    -> ('OpenAI is amazing', 'AI'),
    -> ('Database Management', 'Data'),
    -> ('Test String', 'XYZ');
Query OK, 5 rows affected (0.02 sec)
Records: 5  Duplicates: 0  Warnings: 0

mysql> DELIMITER //
mysql>
```

```
mysql> CREATE PROCEDURE check_substring()
   -> BEGIN
   ->     DECLARE s1 VARCHAR(100);
   ->     DECLARE s2 VARCHAR(100);
   ->     DECLARE done INT DEFAULT 0;
   ->
   ->     DECLARE cur CURSOR FOR SELECT str1, str2 FROM string_table;
   ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
   ->
   ->     OPEN cur;
   ->
   ->     read_loop: LOOP
   ->         FETCH cur INTO s1, s2;
   ->         IF done THEN
   ->             LEAVE read_loop;
   ->         END IF;
   ->
   ->         IF INSTR(s1, s2) > 0 THEN
   ->             SELECT s1 AS Main_String, s2 AS Sub_String, 'Substring Found' AS Result;
   ->         ELSE
   ->             SELECT s1 AS Main_String, s2 AS Sub_String, 'Substring NOT Found' AS Result;
   ->         END IF;
   ->     END LOOP;
   ->
   ->     CLOSE cur;
   -> END;
   -> //
Query OK, 0 rows affected (0.03 sec)

mysql>
mysql> DELIMITER ;
mysql> CALL check_substring();
+-------------+------------+-----------------+
| Main_String | Sub_String | Result          |
+-------------+------------+-----------------+
| Hello World | World      | Substring Found |
+-------------+------------+-----------------+
1 row in set (0.00 sec)

+------------------------+------------+---------------------+
| Main_String            | Sub_String | Result              |
+------------------------+------------+---------------------+
| MySQL Stored Procedure | Code       | Substring NOT Found |
+------------------------+------------+---------------------+
1 row in set (0.01 sec)

+-------------------+------------+-----------------+
| Main_String       | Sub_String | Result          |
+-------------------+------------+-----------------+
| OpenAI is amazing | AI         | Substring Found |
+-------------------+------------+-----------------+
1 row in set (0.01 sec)

+---------------------+------------+-----------------+
| Main_String         | Sub_String | Result          |
+---------------------+------------+-----------------+
| Database Management | Data       | Substring Found |
```

```
+--------------------+------------+-----------------+
1 row in set (0.02 sec)

+-------------+------------+--------------------+
| Main_String | Sub_String | Result             |
+-------------+------------+--------------------+
| Test String | XYZ        | Substring NOT Found |
+-------------+------------+--------------------+
1 row in set (0.04 sec)

Query OK, 0 rows affected (0.05 sec)
```