**Assignment No. 4 (Stored Procedures)**

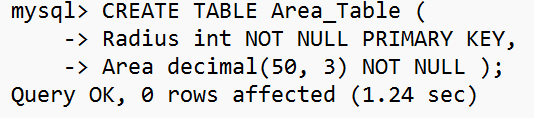
**Title**:- Write a PL/SQL code block to calculate the area of a circle for a value of radius varying from 5 to 20. Store the radius and the corresponding values of calculated area in an empty table named areas, consisting of two columns, radius and area.

**Commands and Screenshots**:-

CREATE TABLE Area\_Table (

-> Radius int NOT NULL PRIMARY KEY,

-> Area decimal(50, 3) NOT NULL );



Procedure :- Insert\_Records

DELIMITER :

mysql> CREATE PROCEDURE Insert\_Records (IN Radius int)

-> BEGIN

-> IF Radius BETWEEN 5 AND 15 THEN

-> IF EXISTS (SELECT \* FROM Area\_Table WHERE Area\_Table.Radius = Radius) THEN

-> SELECT 'Record Already Exists' AS Output;

-> ELSE

-> INSERT INTO Area\_Table VALUES (Radius, 3.14 \* Radius \* Radius);

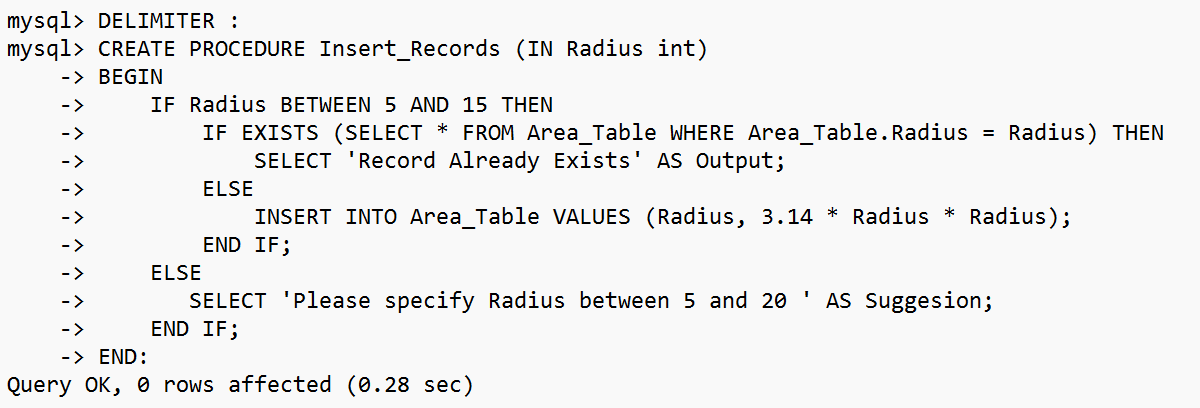
-> END IF;

-> ELSE

-> SELECT 'Please specify Radius between 5 and 20 ' AS Suggesion;

-> END IF;

-> END:



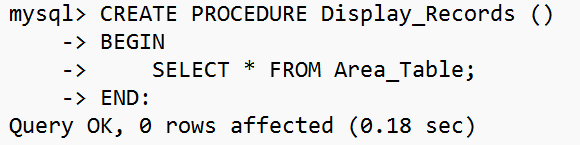
Procedure :- Display\_Records

mysql> CREATE PROCEDURE Display\_Records ()

-> BEGIN

-> SELECT \* FROM Area\_Table;

-> END:



Working of Procedure :-

mysql> DELIMITER ;

mysql> CALL Insert\_Records (5);

Query OK, 1 row affected (0.21 sec)

mysql> CALL Insert\_Records (5);

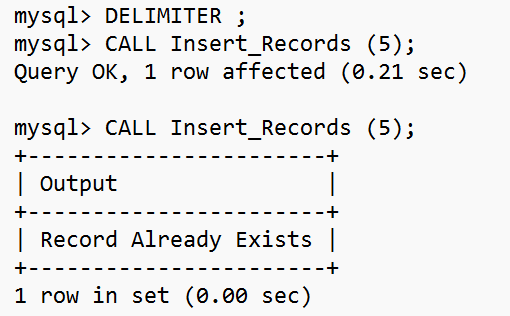
+-------------------------------+

| Output |

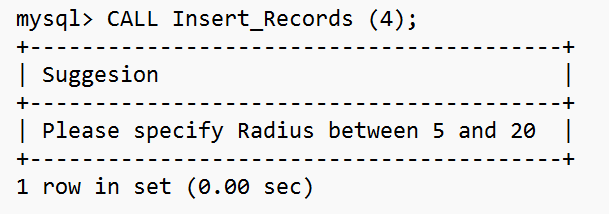
+-------------------------------+

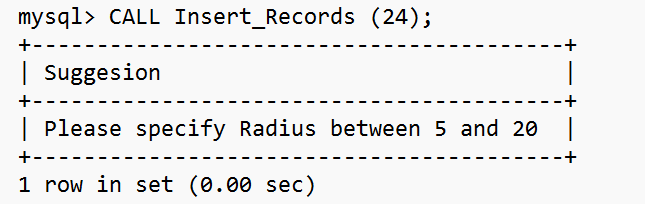
| Record Already Exists |

+-------------------------------+ To avoid Duplicate Records



Working of Procedure :-





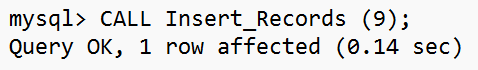
Working of Procedure :- Inserting Records in Area\_Table

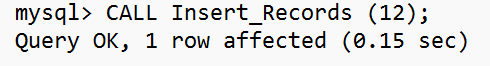
mysql> CALL Insert\_Records (5);

mysql> CALL Insert\_Records (9);

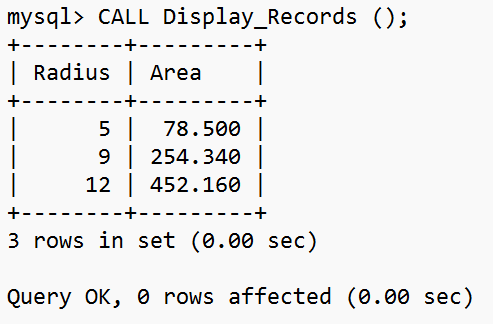
mysql> CALL Insert\_Records (12);







Working of Procedure :- Displaying Records of Area\_Table



**Unnamed PL/SQL code block: Use of Control structure and Exception handling is mandatory.**

Suggested Problem statement:

Consider Tables:

1. Borrower(Roll\_no, Name, DateofIssue, NameofBook, Status)

2. Fine(Roll\_no,Date,Amt)

Accept Roll\_no & NameofBook from user.

* Check the number of days (from date of issue),
* If days are between 15 to 30 then fine amount will be Rs 5per day.
* If no. of days>30, per day fine will be Rs 50 per day & for days less than 30, Rs. 5 per day.
* After submitting the book, status will change from I to R.
* If condition of fine is true, then details will be stored into fine table.
* Also handles the exception by named exception handler or user define exception handler.

mysql> DELIMITER ;

mysql> CREATE TABLE Borrower(rNO INTEGER, name VARCHAR(30), dateOfIssue DATE, bookName VARCHAR(60), status CHAR(3));

Query OK, 0 rows affected (0.36 sec)

mysql> DESC Borrower;

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

| Field | Type | Null | Key | Default | Extra |

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

| rNO | int(11) | YES | | NULL | |

| name | varchar(30) | YES | | NULL | |

| dateOfIssue | date | YES | | NULL | |

| bookName | varchar(60) | YES | | NULL | |

| status | char(3) | YES | | NULL | |

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

5 rows in set (0.00 sec)

mysql> CREATE TABLE Fine(rNo INTEGER, endDate DATE, amt INTEGER);

Query OK, 0 rows affected (0.25 sec)

mysql> DESC Fine;

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

| Field | Type | Null | Key | Default | Extra |

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

| rNo | int(11) | YES | | NULL | |

| endDate | date | YES | | NULL | |

| amt | int(11) | YES | | NULL | |

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

3 rows in set (0.00 sec)

DROP PROCEDURE IF EXISTS LibraryHandler //

CREATE PROCEDURE LibraryHandler(IN rollNum INTEGER, IN Bname VARCHAR(60))

BEGIN

DECLARE totalDays INTEGER;

DECLARE issuedDate DATE;

DECLARE F INTEGER;

SELECT dateOfIssue INTO issuedDate FROM Borrower WHERE rNO = rollNum AND bookName = Bname;

SET totalDays = DATEDIFF(CURDATE(), issuedDate);

UPDATE Borrower SET status = 'R' WHERE rNO = rollNum AND bookName = Bname;

SET F = 0;

SET totalDays = totalDays - 14;

SELECT totalDays AS "DUE Days";

CASE

WHEN totalDays <= 0 THEN SET F = 0;

WHEN totalDays <= 15 THEN SET F = totalDays\*5;

WHEN totalDays > 15 THEN SET F = 15\*5 + ((totalDays - 15)\*50);

END CASE;

IF F > 0 THEN INSERT INTO Fine VALUES(rollNum, CURDATE(), F);

END IF;

END

//

mysql> call LIbraryHandler(3, "SPOS")//

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

| DUE Days |

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

| 40 |

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

1 row in set (0.04 sec)

Query OK, 1 row affected (0.07 sec)

mysql> SELECT \* FROM Borrower //

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

| rNO | name | dateOfIssue | bookName | status |

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

| 1 | Puja | 2022-08-20 | TOC | R |

| 2 | DIV | 2022-08-01 | DBMS | R |

| 3 | Khushi | 2022-07-02 | SPOS | R |

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

3 rows in set (0.00 sec)

mysql> SELECT \* FROM Fine //

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

| rNo | endDate | amt |

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

| 2 | 2022-08-25 | 50 |

| 3 | 2022-08-25 | 1325 |

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

2 rows in set (0.00 sec)

Write a PL/SQL code block to calculate the area of a circle for a value of radius varying from 5 to 9. Store the radius and the corresponding values of calculated area in an empty table named areas, consisting of two columns, radius, and area.

mysql> CREATE TABLE areaTab(radius INTEGER, area DOUBLE(100, 10));

Query OK, 0 rows affected (0.24 sec)

DROP PROCEDURE IF EXISTS CircleArea //

mysql> CREATE PROCEDURE CircleArea(IN radius INTEGER)

BEGIN

IF (radius BETWEEN 5 AND 9) THEN INSERT INTO areaTab VALUES(radius, 3.14\*radius\*radius);

END IF;

END

//

Query OK, 0 rows affected (0.00 sec)

mysql> DROP PROCEDURE IF EXISTS Display //

Query OK, 0 rows affected (0.00 sec)

mysql> CREATE PROCEDURE Display()

-> BEGIN

-> SELECT \* FROM areaTab;

-> END

-> //

Query OK, 0 rows affected (0.00 sec)

mysql> call CircleArea(5) //

Query OK, 1 row affected (0.04 sec)

mysql> call Display() //

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

| radius | area |

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

| 5 | 78.5000000000 |

| 6 | 113.0400000000 |

| 5 | 78.5000000000 |

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

3 rows in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)