ASSIGNMENT NO. 9

Aim

Write and execute PL/SQL stored procedure and function to perform a suitable task on the database.

Objective

To study and implement function and procedure for suitable database application.

Theory

A database stored program (stored module or stored routine) is a computer program that is stored within and executes within the database server.
When program is executed, it is executed within the memory address of a database server process or thread.

Syntax

```
Create procedure example()
Begin
//declare program variables
//statements
End;

Call example();
```

Use of Stored Procedure

1 Allow modular programming.
2 Allow faster execution.
3 Reduce network traffic.
4 Used as a security mechanism.

Hello World Procedure

```
mysql> use mydb;
Database changed

mysql> delimiter $$
mysql> drop procedure if exists helloworld$$
Query OK, 0 rows affected, 1 warning (0.00 sec)

mysql> create procedure helloworld()
    -> begin
    -> select 'Hello world';
    -> end$$
Query OK, 0 rows affected (0.03 sec)
```

```
mysql> call helloworld()$$
+-------------+
| Hello world |
+-------------+
| Hello world |
+-------------+
1 row in set (0.03 sec)

Query OK, 0 rows affected (0.03 sec)
```

Procedure to implement in, out parameters

```
mysql> create procedure calroot(in num1 int, out result int)
    -> begin
    -> set result = sqrt(num1);
    -> select result;
    -> end$$
Query OK, 0 rows affected (0.00 sec)

mysql> call calroot(16, @out_value)$$
+--------+
| result |
+--------+
|      4 |
+--------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

Procedure to calculate Square Root of Number

```
Drop procedure if exists calsqrt $$
Create procedure calsqrt(num1 int)
Begin
Declare result int;
Set result = sqrt(num1);
```

```
Select result;
End $$
```

Procedure to find even or odd number

```
delimiter $$
drop procedure if exists evenodd $$
create procedure evenodd(num1 int)
begin
if mod(num1, 2) = 0 then
select ‘Even number';
end if;
end$$
```

Procedure to get the all employee details

```
create procedure proc_emp()
begin
select * from emp;
end$$
```

Procedure to get the details of Employee

```
create procedure proc_emp1(empid int)
begin
select * from emp where id=empid;
end$$
```

PL/SQL Function Example

```
mysql> create function func1(price int)
    -> returns int
    -> deterministic
    -> begin
    -> declare discount int;
    -> if price<1000 then
    -> set discount=100;
    -> else
    -> set discount=200;
    -> end if;
    -> return discount;
    -> end$
Query OK, 0 rows affected (0.26 sec)

mysql> select func1(800)$
+------------+
| func1(800) |
+------------+
|        100 |
+------------+
1 row in set (0.06 sec)

mysql> select func1(1200)$
+-------------+
| func1(1200) |
+-------------+
|         200 |
+-------------+
1 row in set (0.00 sec)

mysql> create table product(id int, price int)$
Query OK, 0 rows affected (0.46 sec)

mysql> insert into product values(1, 900)$
Query OK, 1 row affected (0.02 sec)

mysql> insert into product values(2, 1200)$
Query OK, 1 row affected (0.05 sec)

mysql> select * from product$
+------+-------+
| id   | price |
+------+-------+
|    1 |   900 |
|    2 |  1200 |
+------+-------+
2 rows in set (0.02 sec)
```

```
mysql> select id, func1(price) from product$
+------+--------------+
| id   | func1(price) |
+------+--------------+
|    1 |          100 |
|    2 |          200 |
+------+--------------+
2 rows in set (0.00 sec)

mysql> select id, func1(price) as discount from product$
+------+----------+
| id   | discount |
+------+----------+
|    1 |      100 |
|    2 |      200 |
+------+----------+
2 rows in set (0.00 sec)

mysql>
```

Procedure to Find Class of Employee

```
create procedure proc_emp2(empid int)
 begin
 declare sal int;
 select salary into sal from emp where id=empid;
 if sal<10000 then
 select 'Class III'as Status;
 elseif sal between 10000 and 50000 then
 select 'Class II';
 else
 select 'Class I';
 end if;
end$$
```

PL/SQL Function

```
CREATE FUNCTION func_emp(esal int) RETURNS VARCHAR(10)
    DETERMINISTIC
```

```
BEGIN
   DECLARE status varchar(10);

   IF esal < 10000 THEN
 SET status = 'Class III';
   ELSEIF (esal <= 50000 AND esal >= 10000) THEN
      SET status = 'Class II';
   ELSE
      SET status = 'Class I';
   END IF;

 RETURN (status);
END$$
```

Output

- Procedure and function.

References:

1. Raghu Ramkrishanan, Johannes Gehrke 4 th Edition "Database Management Systems" 2. Avi Silberschatz , Henry F. Korth , S. Sudarshan, "Database System Concepts, Sixth Edition", ISBN-13: 978-93-3290-138-4, MCGraw Hill

Frequently Asked Questions

Q. No	**Questions**	BT	CO
1	Compare procedure and function.	2	2,3
2	Explain different types of parameters	2	2,3

Guidelines for Students

The experiments should be completed and get checked by the concerned teacher in the lab on or before the date of submission. After which the experiment will not be signed.

Every experiment must be included in the file in following format.

a. **Aim**: In this section write complete objective of the program you are going to make in the lab. This section specifies the complete description of the including problem analysis, input description, method used, fundamental concept and desired output format.

b. **Theory**: Write brief theory related to practical.

c. **Algorithm**: Write Algorithm for given task.

d. **Input**: Write input test data/ or program that are used to test program objective to see whether program is achieving the given objective or not.
e. **Output**: describe the results in few lines
f. **Conclusion**: Write complete conclusion whether what the student has learned from this experiment.
g. **Source Code**: Submit in the form of soft copies.

- **Marking criteria.**

  Experiment completion (Timely)
  Lab file (neatness and regularity)
  Viva (from time to time)
  Mock Practical Exam
  Exam (end term): Practical + Viva

- **Assessment Methodology**

  Timely completion of assignment- 2marks
  Program demonstration- 4 marks