Exp No. 12 Date : 23-08-2019

# PROCEDURES, FUNCTIONS AND PACKAGES

**AIM:**

To study Procedures, Functions and Packages

**QUESTIONS:**

1. Create a function factorial to find the factorial of a number. Use this function in a PL/SQL Program to display the factorial of a number read from the user

```
procedures=# CREATE OR REPLACE FUNCTION factorial(n INT) RETURNS INTEGER AS $$
procedures$# DECLARE
procedures$#     fact INTEGER := 1;
procedures$#     temp INTEGER := n;
procedures$# BEGIN
procedures$#     LOOP
procedures$#         EXIT WHEN temp <= 0;
procedures$#         fact := temp * fact;
procedures$#         temp := temp - 1;
procedures$#     END LOOP;
procedures$#     RETURN fact;
procedures$# END
procedures$# $$ LANGUAGE PLPGSQL;
CREATE FUNCTION
procedures=# SELECT factorial(7);
 factorial
-----------
      5040
(1 row)

procedures=#
```

2. Create a table student_details(roll int, marks int, phone int). Create a procedure pr1 to update all rows in the database. Boost the marks of all students by 5%

```
procedures=# CREATE TABLE student_details(roll INT, marks INT, phone BIGINT);
CREATE TABLE
procedures=# INSERT INTO student_details VALUES(1, 70, 9496947423),
procedures-#                                   (2, 85, 9495941358),
procedures-#                                   (3, 78, 8281865009);
INSERT 0 3
procedures=# SELECT * FROM student_details;
 roll | marks |   phone
------+-------+------------
    1 |    70 | 9496947423
    2 |    85 | 9495941358
    3 |    78 | 8281865009
(3 rows)

procedures=#
```

```
procedure=# CREATE OR REPLACE PROCEDURE pr1() AS $$
procedure$# BEGIN
procedure$#     UPDATE student_details
procedure$#     SET marks = marks + (marks * 0.05);
procedure$# END
procedure$# $$ LANGUAGE plpgsql;
CREATE PROCEDURE
procedure=#
procedure=# call pr1();
CALL
procedure=# SELECT * FROM student_details ;
 roll | marks |   phone
------+-------+------------
    1 |    74 | 9496947423
    2 |    89 | 9495941358
    3 |    82 | 8281865009
(3 rows)

procedure=#
```

3. Create table student (id, name, m1, m2, m3, total, grade). Create a function f1 to calculate grade. Create a procedure p1 to update the total and grade

```
procedure=# CREATE OR REPLACE FUNCTION func_calculate_total_and_grade(stud_id INTEGER) RETURNS VOID AS $$
procedure$# BEGIN
procedure$#     UPDATE student
procedure$#     SET total = m1 + m2 + m3
procedure$#     WHERE id = stud_id;
procedure$#
procedure$#     UPDATE student
procedure$#     SET grade = CASE
procedure$#                     WHEN ((m1 + m2 + m3) / 3) > 40 THEN 'P'
procedure$#                     ELSE 'F'
procedure$#                 END
procedure$#     WHERE id = stud_id;
procedure$# END
procedure$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
procedure=#
procedure=# CREATE OR REPLACE PROCEDURE pro_update_total_and_grade(sid INT, sname TEXT, mark1 INT, mark2 INT, mark3 INT) AS $$
procedure$# BEGIN
procedure$#     INSERT INTO student VALUES(sid, sname, mark1, mark2, mark3);
procedure$#     COMMIT;
procedure$#     PERFORM func_calculate_total_and_grade(sid);
procedure$# END
procedure$# $$ LANGUAGE plpgsql;
CREATE PROCEDURE
procedure=#
procedure=#
```

```
procedure=# CALL pro_update_total_and_grade (1, 'hisham', 55, 87, 63);
CALL
procedure=# SELECT * FROM student;
 id |  name  | m1 | m2 | m3 | total | grade
----+--------+----+----+----+-------+-------
  1 | hisham | 55 | 87 | 63 |   205 | P
(1 row)

procedure=# CALL pro_update_total_and_grade (2, 'john', 35, 58, 21);
CALL
procedure=# CALL pro_update_total_and_grade (3, 'ravi', 87, 36, 94);
CALL
procedure=# SELECT * FROM student;
 id |  name  | m1 | m2 | m3 | total | grade
----+--------+----+----+----+-------+-------
  1 | hisham | 55 | 87 | 63 |   205 | P
  2 | john   | 35 | 58 | 21 |   114 | F
  3 | ravi   | 87 | 36 | 94 |   217 | P
(3 rows)

procedure=#
```

4 .Create a package pk1 consisting of the following functions and procedures

Procedure proc1 to find the sum, average and product of two numbers

Procedure proc2 to find the square root of a number

Function named fn11 to check whether a number is even or not

A function named fn22 to find the sum of 3 numbers

Use this package in a PL/SQL program. Call the functions f11, f22 and procedures pro1, pro2 within the program and display their results.

Creating schema pk1 procedures

```
postgres=# CREATE SCHEMA pk1;
CREATE SCHEMA
postgres=#
```

Creating procedures

```
postgres=# CREATE OR REPLACE PROCEDURE pk1.proc1(num1 NUMERIC, num2 NUMERIC) AS $$
postgres$# DECLARE
postgres$#     sum NUMERIC;
postgres$#     avg FLOAT;
postgres$#     prod NUMERIC;
postgres$# BEGIN
postgres$#     sum := num1 + num2;
postgres$#     avg := (num1 + num2) / 2;
postgres$#     prod := num1 * num2;
postgres$#     RAISE NOTICE ' ';
postgres$#     RAISE NOTICE 'Numbers are % and %', num1, num2;
postgres$#     RAISE NOTICE 'Sum : % ---- Average : % ---- Product : %', sum, avg, prod;
postgres$#     RAISE NOTICE ' ';
postgres$# END
postgres$# $$ LANGUAGE plpgsql;
CREATE PROCEDURE
postgres=#
postgres=#
postgres=# CREATE OR REPLACE PROCEDURE pk1.proc2(num1 FLOAT) AS $$
postgres$# BEGIN
postgres$#     RAISE NOTICE 'Square root of % is %', num1, SQRT(num1);
postgres$#     RAISE NOTICE ' ';
postgres$# END
postgres$# $$ LANGUAGE plpgsql;
CREATE PROCEDURE
postgres=#
```

Creating functions

```
postgres=# CREATE OR REPLACE FUNCTION pk1.fn1(num1 INT) RETURNS VOID AS $$
postgres$# BEGIN
postgres$#     IF num1 % 2 = 0 THEN
postgres$#         RAISE NOTICE '% is even number', num1;
postgres$#     ELSE
postgres$#         RAISE NOTICE '% is odd number', num1;
postgres$#     END IF;
postgres$#     RAISE NOTICE ' ';
postgres$# END
postgres$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
postgres=#
postgres=# CREATE OR REPLACE FUNCTION pk1.fn2(num1 INT, num2 INT, num3 INT) RETURNS VOID AS $$
postgres$# BEGIN
postgres$#         RAISE NOTICE 'Sum of %, % and % is : %', num1, num2, num3, (num1+num2+num3);
postgres$#         RAISE NOTICE ' ';
postgres$# END
postgres$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
postgres=#
```

Creating procedure to call functions and procedures inside schema

```
postgres=# CREATE OR REPLACE PROCEDURE pk1.ALL() AS $$
postgres$# BEGIN
postgres$#     CALL pk1.proc1(10, 5);
postgres$#     CALL pk1.proc2(25);
postgres$#     PERFORM pk1.fn1(12);
postgres$#     PERFORM pk1.fn2(2, 6, 1);
postgres$#     COMMIT;
postgres$# END
postgres$# $$ LANGUAGE plpgsql;
CREATE PROCEDURE
postgres=#
postgres=# CALL pk1.ALL();
NOTICE:
NOTICE:  Numbers are 10 and 5
NOTICE:  Sum : 15 ---- Average : 7.5 ---- Product : 50
NOTICE:
NOTICE:  Square root of 25 is 5
NOTICE:
NOTICE:  12 is even number
NOTICE:
NOTICE:  Sum of 2, 6 and 1 is : 9
NOTICE:
CALL
postgres=#
```

**RESULT:**

The PL/SQL program was executed successfully and the output was obtained.