EX.NO:12 .05.2025	**PROCEDURE AND FUNCTIONS**

AIM:

Use PL/SQL to create simple procedures for executing reusable tasks, and functions to return computed results. Both enhance modularity and maintainability in databases.

PL/SQL:Procedural Language/Structural Query Language.

PL/SQL is,

To add programming logic to SQL

To create triggers

A PL/SQL procedure can be,

Named block

Unnamed block

The part in a block are,

Declaration part

Execution part

Execution part(optional)

CREATE TABLE:

```
SQL> CREATE TABLE circumference1 (
      radius NUMBER(2,0),
      circumference1 NUMBER(10,3)
   );
Table created.
```

A SIMPLE PL/SQL PROCEDURE:

```
SQL> DECLARE
       pi CONSTANT NUMBER := 3.14;
       radius INTEGER := 5;
       circumference1 NUMBER(6,3);
   BEGIN
       circumference1 := 2 * pi * radius;
     INSERT INTO circumference1 values(radius, circumference1)
```

```
  END;
  /
PL/SQL procedure successfully completed.
SQL> select * from circumference1;

   RADIUS       CIRCUMFERENCE1
  ----------    -------------------------
        5                 31.4
```

PL/SQL PROCEDURE WITH FOR LOOP:

```
 SQL> DECLARE
      pi CONSTANT NUMBER := 3.14;
      circumference1_value NUMBER(6,3);
   BEGIN
     FOR radius IN 1..7 LOOP
         circumference1_value := 2 * pi * radius;
        INSERT INTO circumference1 values (radius, circumference1)
    END LOOP;
  END;
  /
PL/SQL procedure successfully completed.
SQL> select * from circumference1;
RADIUS      CIRCUMFERENCE1
----------      ----------------------
        1                6.28
        2               12.56
        3               18.84
        4               25.12
        5                31.4
        6               37.68
        7               43.96
```

PL/SQL PROCEDURE WITH WHILE LOOP:

```
SQL> DECLARE
       pi CONSTANT NUMBER := 3.14;

       radius INTEGER := 1;  -- Start radius at 1
       circumference1 NUMBER(6,3);
   BEGIN
       WHILE radius <= 7 LOOP
          circumference1 := 2 * pi * radius;

         INSERT INTO circumference1 (radius, circumference1)
          VALUES (radius, circumference1);
        radius := radius + 1;
      END LOOP;

  END;
  /
PL/SQL procedure successfully completed.

SQL> select * from circumference1;

    RADIUS         CIRCUMFERENCE1
  ----------     ----------------------------
         1                   6.28
         2                  12.56
         3                  18.84
         4                  25.12
         5                   31.4
         6                  37.68
         7                  43.96
7 rows selected.
```

PL/SQL PROCEDURE WITH EXCEPTION:

```
SQL> declare
    pi constant number:=3.14;
    radius integer(5);
    circumference1 number(10,3);
    temp number(10,3);
    begin
    radius:=3;
    while radius<=7
    loop
   temp:=1/(radius-4);
   circumference1:=2*pi*radius;
   insert into circumference1 values(radius,circumference1);
   radius:=radius+1;
   end loop;
   exception

   when ZERO_DIVIDE then
   insert into circumference1 values(0,0);
   end;
   /
PL/SQL procedure successfully completed.
SQL> select * from circumference1;
    RADIUS      CIRCUMFERENCE1
----------------   --------------------------
         3                 18.84
         0                   0
```

CONTENTS	MARKS ALLOTED	MARKS OBTAINED
Aim,algorithm,SQL,PL/SQL	30	
Execution and Result	20	
Viva	10	
Total	60	

RESULT:

Thus, simple procedures and functions were created and executed successfully using PL/SQL. They performed the desired operations, confirming correct implementation.