Ex.No. 10

PL/SQL Procedures

Aim:

To manipulate Stored procedures in PL/SQL.

PL/SQL Procedure

A PL/SQL procedure is a reusable unit that encapsulates specific business logic of the application. Technically speaking, a PL/SQL procedure is a named block stored as a schema object in the Oracle Database.

The following illustrates the basic syntax of creating a procedure in PL/SQL:

```
CREATE OR REPLACE PROCEDURE <procedure_name>
(<variable_name>IN/OUT/IN OUT <datatype>, <variable_name>IN/OUT/IN OUT
<datatype>,...)  IS/AS
variable/constant declaration;
BEGIN
-- PL/SQL subprogram body;
EXCEPTION
-- Exception Handling block ;
END <procedure_name>;
```

PL/SQL procedure header

A procedure begins with a header that specifies its name and an optional parameter list. Each parameter can be in either IN, OUT, or INOUT mode. The parameter mode specifies whether a parameter can be read from or write to.

- **IN** - An IN parameter is read-only. One can reference an IN parameter inside a procedure, but cannot change its value. Oracle uses IN as the default mode. It means that if you don't specify the mode for a parameter explicitly, Oracle will use the IN mode.
- **OUT** - An OUT parameter is writable. Typically, one set a returned value for the OUT parameter and return it to the calling program. Note that a procedure ignores the value that supply for an OUT parameter.
- **INOUT -** An INOUT parameter is both readable and writable. The procedure can read and modify it.

The **OR REPLACE** option allows you to overwrite the current procedure with the new code.

PL/SQL procedure body

Similar to an anonymous block, the procedure body has three parts. The executable part is mandatory whereas the declarative and exception-handling parts are optional. The executable part must contain at least one executable statement.

1) Declarative part

In this part, one can declare variables, constants, cursors, etc. Unlike an anonymous block, a declaration part of a procedure does not start with the DECLARE keyword.

2) **Executable part**

This part contains one or more statements that implement specific business logic. It might contain only a NULL statement.

3) Exception-handling part
This part contains the code that handles exceptions.

EXAMPLE:

```
CREATE OR REPLACE PROCEDURE greetings
AS BEGIN
   dbms_output.put_line('Hello World!'); END; /
```

To Execute Query from sqlplus terminal

```
EXECUTE greetings;

Hello World

PL/SQL procedure successfully completed.
```

Q1) Procedure to add Two numbers

```
CREATE OR REPLACE PROCEDURE sumTwoNum(num1 IN number, num2 IN
number) IS
DECLARE
 tot number;

BEGIN
   tot := num1 + num2;
END;
/
```

PL/SQL Block that invokes the Procedure `sumTwoNum`

```
set serveroutput on;
DECLARE
x number;
y number;
BEGIN
x := &x;
y := &y;
Sum(x,y);
END;
/
```

Q2) Write a PL/SQL Procedure to update the salary of employee whose job is Clerk.

```
create or replace procedure emp_sal_update
 IS
 BEGIN
 update emp
 set sal=sal+sal*0.10
 where job = 'Clerk';
 END;
```

Q3) Write a PL/SQL Procedure to find the number of managers in the employee table

Q4) Write a PL/SQL Procedure to display the details of employees from the emp table whose name are starting with ‘A’ and ‘M’.