Oracle PL/SQL – Procedures

codingcircuit23@gmail.com

PL/SQL sub programs

- A sub program is a program unit/module that performs a particular task.
- A sub program can be created at below levels
  - At schema level
  - Inside a package
  - Inside a PL/SQL block
- Standalone program which is created at schema level is created and stored at database using CREATE keyword and deleted using DROP.
- Two main sub programs supported by PL/SQL are
  - Procedures
  - Functions

PL/SQL Procedure Overview

- A **Procedure** is a named PL/SQL block or Subprogram which performs one or more specific task.
- Procedure is similar to an anonymous PL/SQL block but it is named for repeated usage.
- Sub programs help in dividing the program into well-defined modules.
- **Reusability** is the main advantage of sub programs. When a procedure is created, it is first compiled and stored in the database. This compiled code can be invoked from any number of PL/SQL blocks.
- Also referred as **Stored Procedure** or simply **Proc.**

Procedure Structure

A procedure should have a header and a body

- The header consists of the name of the procedure and the parameters or variables passed to the procedure.
- The body consists of the below sections similar to a PL/SQL anonymous block.

  - **Declarative Part** :

    Optional. DECLARE keyword not required. The scope of the declared items like variables, cursors, etc is local to the sub program and cannot be accessed once the sub program execution completes.

  - **Executable Part** :

    Mandatory. Contains statements that perform the designated action.

  - **Exception handling** :

    Optional. Contains code that handles run time errors.

Basic Syntax

```
CREATE [OR REPLACE] PROCEDURE <procedure_name>
[(parameter_name [IN | OUT | IN OUT] datatype [, ...])]
IS | AS
        /*Declarative section*/
BEGIN
        /*Execution section*/
EXCEPTION
        /*Exception section*/
END [procedure_name];
```

procedure_name – name of the procedure

OR REPLACE – If a procedure with the same name already exists then that would be dropped and recreated. Else a new procedure will be created.

IN or **OUT** or **IN OUT** – different modes of parameter

Either **IS** or **AS** keyword can be used. They are equivalent.

Executing a Procedure

A standalone procedure can be called in two ways:

i. Using the EXECUTE keyword
ii. Calling the procedure name from a PL/SQL block.

a) EXECUTE <procedure_name>[actual parameters];

b)

```
BEGIN
<procedure_name>[actual parameters];
END;
/
```

Sample procedure without parameters

```
CREATE OR REPLACE PROCEDURE employer_details IS
    --declarations here
    CURSOR emp_cur IS SELECT empname,designation FROM employee;
    emp_rc emp_cur%rowtype;
BEGIN
    FOR emp_rc in emp_cur
    LOOP
    dbms_output.put_line(emp_rc.empname||' '||emp_rc.designation);
    END LOOP;
END;
/
```

```
EXECUTE employer_details;

PL/SQL procedure successfully completed

JOHN  C1
Rahul ASE
```

Points to remember while creating a procedure

- If the procedure already exists and if the OR REPLACE keyword is not given , then the CREATE statement will return an Oracle error.
- If parameter mode is not specified then it will consider IN as default mode.
- While defining the formal parameters, only the data type should be mentioned and the size of the variable should not be specified.
- IN OUT parameter should always be a variable and not a constant
- In order to change/modify the code of a procedure, the procedure must be dropped and recreated and there is no separate option for modifying the procedure code.

Types of Parameters

Formal Parameters :

- A procedure heading declares formal parameters.
- Each formal parameter can specify a mode,data type and a default value.
- They are variables declared in the procedure header and referenced in the execution part.

Example : CREATE OR REPLACE PROCEDURE p1(c IN number,c1 IN number)

Actual Parameters :

- When procedures are invoked , the actual parameters are passed to it.
- They are the variables or expressions that are passed to the subprogram when invoked.

Example : p1(12,12)

NOTE:

Formal and its corresponding actual parameters must have compatible data types.

Parameter Modes

PARAMETER MODE	DESCRIPTION
IN	• Allows to pass a value to the program. • It is a read-only parameter. • It is the default mode of parameter passing. • The value cannot be changed inside the sub program. • It cannot be assigned a new value inside the sub program. • Constant, literal, variable or expression can be passed. • Parameters are passed by reference.
OUT	• It returns a value to the calling program. • Takes the value out of the sub program. • A variable to be passed to an OUT parameter. • It acts like a variable inside the sub program. • The value can be changed or a new value can be assigned. • Pass the value as pass by value.
IN OUT	• Passes an initial value to a sub program and returns an updated value to the caller. • It should be a variable and not a constant • Values can be modified inside the sub program.

Deleting a Procedure

Syntax for deleting a procedure is as follows

DROP PROCEDURE <procedure_name>

Example:

DROP PROCEDURE employer_details;

Sample procedure with parameters - 1


```
CREATE OR REPLACE PROCEDURE calc_total(n1 IN NUMBER,n2 IN
NUMBER,n3 OUT NUMBER) IS
BEGIN
          n3:=n1+n2;
END;
/
```

```
SET SERVEROUTPUT ON;
DECLARE
        a NUMBER;
        b NUMBER;
        c NUMBER;
BEGIN
     a:=10;
     b:=70;
     calc_total(a,b,c);
     dbms_output.put_line('SUM : '|| c);
END;
/
```

Sample Procedure Output - 1

Output:

```
SUM : 80
PL/SQL procedure successfully completed
```

- In the previous example, a procedure with name calc_total is created which takes two parameters n1 and n2 as input and will return the sum of n1 and n2.

- Then, calc_total procedure is invoked from an anonymous block with parameter values as 10 and 70. The sum which is returned by the procedure is printed.

Sample procedure with parameters - 2

```
CREATE OR REPLACE PROCEDURE calc_salary(empid IN NUMBER,
                                        c_salary IN OUT NUMBER) IS
    v_sal NUMBER;
BEGIN
    SELECT salary INTO v_sal FROM employee WHERE emp_id=empid;
    c_salary := (c_salary/100*v_sal)+v_sal;
END;
/
```

```
SET SERVEROUTPUT ON;
DECLARE
    x NUMBER := 10;
BEGIN
    calc_salary(101,x);
    dbms_output.put_line('Updated Salary : '|| x);
END;
/
```

Sample Procedure Output - 2

Output:

Updated Salary : 88000

PL/SQL procedure successfully completed

- A procedure with name calc_salary is created which takes the employee ID and commission as input and the updated salary is the output. c_salary acts as the commission input and the updated salary output.

- Then, calc_salary procedure is invoked from an anonymous block passing employee ID as 101 and commission percentage as 10. Assume the salary of employee is 80000 and then the updated salary would be 88000.