Oracle PL/SQL – Functions

PI/SQL Function Overview

- A function is a named PL/SQL block or subprogram which is similar to procedure except that a function always returns a value.
- Procedures may produce an output but functions will return a value
- Functions should be declared and defined before they are invoked.
- Accepts zero to many parameters as input.
- Also referred as **stored function** or **Func**.

Syntax for Creating Functions

```
CREATE [OR REPLACE] FUNCTION <function_name>
[(parameter_name [IN | OUT | IN OUT] datatype [, …])]
RETURN return_datatype
IS | AS
            /*Declarative section*/
BEGIN
          /*Execution section*/
      RETURN <variable_name>;
EXCEPTION
          /*Exception section*/
END [function_name];
```

- **function_name** is the name of the function to be created.
- **CREATE OR REPLACE** creates a new function or modifies an existing function.
- **IN/ OUT/ IN OUT** are the different modes of parameter
- **RETURN** is mandatory. Specifies the data type which is being returned from the function.
- Function body should contain a return statement returning a value.

Sample function - 1

```
CREATE OR REPLACE FUNCTION func_get_employee_name(e_id
                                                  number)
RETURN VARCHAR IS
    v_emp_name VARCHAR(20);
BEGIN
    SELECT emp_name INTO v_emp_name FROM employee
                                WHERE emp_id = e_id;
RETURN v_emp_name;
END;
/
```

Sample function - 2

```
CREATE or REPLACE FUNCTION func_calc_oddeven(a number)
RETURN VARCHAR2
AS
        b NUMBER := 2;
BEGIN
    IF ((a MOD b = 0)) THEN
        RETURN  'The number is even';
    ELSE
        RETURN  'The number is odd';
    END IF;
END;
/
```

```
BEGIN
    dbms_output.put_line(func_calc_oddeven(12);
END;
/

The number is even.
```

RETURN statement in function

- Inside the function body, the RETURN statement is used to return control to the calling environment with a value.
- The header section defines the return data type like VARCHAR, NUMBER,etc
- Both the **execution and the exception section** should return a value which is of the data type defined in the header.
- There can be more than one RETURN statement in a function, but only one of them will be executed based on the logic.

Executing a Function

A function can be executed/invoked in the following ways:

i. From blocks by defining a variable to catch the return value.
ii. From a SELECT statement
iii. In a PL/SQL statement

Example: 1 – From block

```
DECLARE
        employee_name VARCHAR2(30);
         v_emp_id  NUMBER;
BEGIN
         v_emp_id := 101;
        employee_name :=  func_get_employee_name(v_emp_id); – invoking function
        dbms_output.put_line('employee name is -->'||employee_name);
END;
/
```

Executing a Function

Example: 2 – From SELECT statement

```
select * from employee_allocation where ename = func_get_employee_name(101) ;
```

Example: 3 – In a PL/SQL statement

```
dbms_output.put_line(func_get_employee_name(101));
```

Deleting a Function

Syntax for deleting a function is as follows

DROP FUNCTION <function_name>

Example:

DROP FUNCTION calc_salary;

PL/SQL Procedures Vs Functions

PROCEDURE	FUNCTION
Not mandatory to return a value	Mandatory to return one value
Can produce multiple values using OUT parameter	Can return only one value
Cannot be invoked from SQL statements	Can be invoked from SQL statements
Generally, used to perform an action	Used to compute a value
DDL statements can be incorporated using EXECUTE IMMEDIATE statement.	DDL statements cannot be used

Anonymous

```
[DECLARE]


BEGIN
  --statements


[EXCEPTION]

END;
```

Procedure

```
CREATE
PROCEDURE name
IS

BEGIN
  --statements

[EXCEPTION]

END;
```

Function

```
CREATE
FUNCTION name
RETURN datatype
IS
BEGIN
  --statements
  RETURN value;
[EXCEPTION]

END;
```