|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | Feature | Procedure | Function | | Type | Named PL/SQL subprogram | Named PL/SQL subprogram | | Structure | Block structure similar to anonymous block | Block structure similar to anonymous block | | Parameters | Supports **IN**, **OUT**, and **IN OUT** parameters | Supports **IN**, **OUT**, and **IN OUT** parameters | | Declarative Section | Optional (uses **IS** or **AS** instead of DECLARE) | Optional (uses **IS** or **AS** instead of DECLARE) | | Executable Section | Mandatory | Mandatory | | Exception Handling | Optional | Optional | | Return Value | Can return data via **OUT** or **IN OUT** parameters | Must return a value using the **RETURN** statement | | RETURN Clause in Header | Not required | Must contain a RETURN clause in the header | | RETURN Statement | Optional; returns control to calling program | Mandatory; returns control and result to calling program | | SQL Usage | Cannot be called from SQL | Can be called from SQL | | Invocation | Executed as a PL/SQL statement | Invoked as part of an expression | | Considered As | Not considered an expression | Considered an expression | |

**Different between procedures and function**

**Procedure without parameter**

create or replace procedure add\_dept as

v\_dpt\_id departments.department\_id%TYPE := 1001;

v\_dpt\_name departments.department\_name%TYPE := 'Accounts';

begin

Insert into departments(department\_id,department\_name)

values(v\_dpt\_id,v\_dpt\_name);

DBMS\_OUTPUT.PUT\_LINE('Id :' || v\_dpt\_id || 'name:' || v\_dpt\_name);

end;

**Calling**

BEGIN

add\_dept;

END;

-------------------------------------------------------------------

**Procedure with parameter. raise\_salary**

create or replace procedure insert\_dept(p\_id IN number,p\_name IN varchar2) as

begin

Insert into departments(department\_id,department\_name)

values(p\_id,p\_name);

DBMS\_OUTPUT.PUT\_LINE('Department inserted: ' || p\_id || ' - ' || p\_name);

end;

**Calling**

BEGIN

insert\_dept(2000, 'Alaa');

END;

**Write a procedure named raise\_salary for all the employees by 10 percent**

CREATE OR REPLACE PROCEDURE raise\_salary(p\_percent IN NUMBER) IS

BEGIN

FOR rec IN (SELECT employee\_id , Salary FROM employees) LOOP

UPDATE employees

SET salary = salary \* (1 + p\_percent / 100)

WHERE employee\_id = rec.employee\_id;

DBMS\_OUTPUT.PUT\_LINE('Salary updated for employee ID: ' || rec.employee\_id || 'salary :' || rec.salary);

END LOOP;

END;

**Calling**

-- For all employees

BEGIN

raise\_salary(10);

END;

**Function**

create or replace function get\_salary(p\_id IN number)

return number is

v\_salary number(8,2);

begin

select salary INTO v\_salary from employees where employee\_id = p\_id;

return v\_salary;

end;

**Calling ( for one )**

declare

v\_salary number(8,2);

begin

v\_salary := get\_salary(101);

dbms\_output.put\_line(v\_salary);

end;

**Calling ( for all )**

declare

v\_salary number(8,2);

begin

for rec In ( select employee\_id from employees ) loop

v\_salary := get\_salary(rec.employee\_id);

dbms\_output.put\_line(v\_salary);

end loop;

end;