Ex.No.: 11		PL SQL PROGRAMS
Date:		

PROGRAMS

TO DISPLAY HELLO MESSAGE

```
SQL> set serveroutput on;
SQL> declare
  2  a varchar2(20);
  3  begin
  4  a:='Hello';
  5  dbms_output.put_line(a);
  6  end;
  7  /
Hello

PL/SQL procedure successfully completed.
```

TO INPUT A VALUE FROM THE USER AND DISPLAY IT

```
SQL> set serveroutput on;
SQL> declare
  2  a varchar2(20);
  3  begin
  4  a:=&a;
  5  dbms_output.put_line(a);
  6  end;
  7  /
Enter value for a: 5
old   4: a:=&a;
new   4: a:=5;
5

PL/SQL procedure successfully completed.
```

GREATEST OF TWO NUMBERS

```
SQL> set serveroutput on;

SQL> declare
  2  a number(7);
```

PROGRAM 1

Write a PL/SQL block to calculate the incentive of an employee whose ID is 110.

```
v_employee_id   employee employee_id %.TYPE  := 100;
v_salary        employees . salary %. Type ;
v_incentive     NUMBER
v_incentive_pct CONSTANT NUMBER  :  = 0.10
BEGIN
   SELECT salary into v_salary from employees
   where employee_id, v_employee_id ;
 v_incentive  : = v_salary ←  v_incentive_pct
DBMS_OUTPUT PUT_LINE ("Incentive for empID" || v_employee_id ||
 END :                    'is' || v_Incentive.
```

PROGRAM 2

Write a PL/SQL block to show an invalid case-insensitive reference to a quoted and without quoted user-defined identifier.

```
Declare v_test variable NUMBER := 100;
BEGIN   select salary into v-salary from employees
        where employee_id   = v_employee_id
     v_incentive ! = v_salary ← v_incentive_pct ;
   DBMS_OUTPUT. PUTLINE (' Error: ' || su learn) ;
END ;
   DBMS_OUTPUT.PUTLINE ('Value :' || v_Test_VARIABLE') ;
```

PROGRAM 3

Write a PL/SQL block to adjust the salary of the employee whose ID 122.
Sample table: employees

```
Declare  v_employee_id      employees.employee_id %Type := 122;
         v_newsalary        employees.salary %TYPE;
         v_currentsal       employees.salary %.TYPE;
         v_adj              NUMBER := 500;
BEGIN
Select salary into v_currentsal from employees where
   employee_id = v_employee_id;
  v_new_sal := v_current_sal * v_adj;
UPDATE employees SET salary = v_newsalary where
employee_id = v_employee_id;
COMMIT
  DBMS.OUTPUT.PUTLINE("Salary updated!");
END;
```

PROGRAM 4

Write a PL/SQL block to create a procedure using the "IS [NOT] NULL Operator" and show AND operator returns TRUE if and only if both operands are TRUE.

```
create or replace procedure update_emp_status (pemp_id IN
  employees emp_ID %.TYE) AS
   v_sal      employees.salary %TYPE
   v_dept_id  employees.department_id %TYPE;
BEGIN
select sal, dept_id into v_sal _v.depid from employees
where emp_id = p_emp_id
IF v_sal IS NOT NULL AND v_depid IS NOT NULL then
Update employees SET status = 'Active' where emp_id = p.emp_id;
```

PROGRAM 5

Write a PL/SQL block to describe the usage of LIKE operator including wildcard characters and escape character.

```
Declare
    V_emp_name  emp.name %.Type.
BEGIN
    For rec IN (Select name from emp where name like 'J-%'
    ESCAPE '\')
 LOOP
  DBMS_OUTPUT.PUTLINE (rec.name);
  END LOOP ;
END ;
```

PROGRAM 6

Write a PL/SQL program to arrange the number of two variable in such a way that the small number will store in num_small variable and large number will store in num_large variable.

```
 Declare
num1  number := 25 ;
num2  NUMBER :  = 10 ;
num_small  number;
num-large  number;
Begin
       If  num1 < num 2 then
       num_s : = num1 ;
       num_l  : = num2 ;
     Else
          num_s  : = num2
          num_l  : = num 2
   DBMS_ OUTPUT PUT_LINE (NUM_S)
```

PROGRAM 7

Write a PL/SQL procedure to calculate the incentive on a target achieved and display the message either the record updated or not.

```
END IF
update emp SET INC = V_inc where emp_id
 If SQL% ROWCOUNT >0 Then
    DBMS_OUTPUT.BUTLINE ("No record updated");
 END IF ;
 END ;
```

PROGRAM 8

Write a PL/SQL procedure to calculate incentive achieved according to the specific sale limit.

```
create or replace procedure cal_inc (p_ salantIN
number) AS  V_inc_num ;
Begin
     If  P_sal_amt >= 5000 then
       V_inc  := p_ sel_ amt  : 0.15 ;
    Else
        V_inc  :  P sel_amt  := 0.05 ;
   END If  :
     DBNS_ OUTPUT. PUTLINE  (p_ sel_amt ||V_inc);
```

PROGRAM 9

Write a PL/SQL program to count number of employees in department 50 and check whether this department have any vacancies or not. There are 45 vacancies in this department.

```
DBMS_OUTPUT.PUTLINE(v_emp_count);
If v_emp_count < v_vacancies Then
    DBMS_OUTPUT.PUTLINE("So vacancies available");
Else
    DBMS_OUTPUT.PUTLINE('No vacancies');
END IF;
END;
```

PROGRAM 10

Write a PL/SQL program to count number of employees in a specific department and check whether this department have any vacancies or not. If any vacancies, how many vacancies are in that department.

```
Declare
  v_dept_id number := 50;
  v_emp_count number;
  v_total_pos number := 100;
  v_vac Number;
BEGIN
  select count(1) into v_emp_count from employees
  where dept_ID = v_dept_id
    v_vacancies := v_total_pos - v_emp_count;
    DBMS_OUTPUT.PUTLINE(v_emp_count);
    If v_vac > 0 Then
    DBMS_OUTPUT.PUTLINE('v_vac vacancies Available');
```

PROGRAM 11

Write a PL/SQL program to display the employee IDs, names, job titles, hire dates, and salaries of all employees.

```
Declare
Begin  for rec in (select emp_id, first name || 'last name
AS  emp_name job,title, Hiredate, salary from employees
JOIN jobs on employees_job_id = jobs.job_id)
LOOP
 DBMS_OUTPUT.PUTLINE (rec.emp_id || rec.empname ||rec
job title).
 END LOOP;
```

PROGRAM 12

Write a PL/SQL program to display the employee IDs, names, and department names of all employees.

```
Declare
Begin  for rec IN (select e.emp_id, e.f_name ||' ' ||
e.l_name AS emp_name dept_name from employees &
JOIN department d on e.dept_id = d.dept_id)
LOOP
     DBMS.OUTPUT.PUTLINE (rec.emp_id || rec.emp_name ||
          rec.dept_name
   END loop ;
        END;
```

PROGRAM 13
Write a PL/SQL program to display the job IDs, titles, and minimum salaries of all jobs.

```
Declare
Begin
   for rec IN Select bobjid , job title, min. Sal from Jobs
LOOP
 DBMS -OUTDUT. PUT_LINE(REC. job_id  ||rec. job _title ||
 rec._ min._ sal);
END loop ;
END ;
```

PROGRAM 14
Write a PL/SQL program to display the employee IDs, names, and job history start dates of all employees.

```
Declare
Begin
 for  rec  IN ( Seclat e.emp_id , e. first name || " " ||
e.lastname  AS  emp_name,  In start date  from
employees  e  Join  job_history  on e emp_id = jh . emp_id
LOOP
BNS.DUSPUT .PUTLINE (rec. emp_id , REC emp. name,
   To_char (rec Start date : DD.MON YYYY);
    END LOOP ;
END ;
```

PROGRAM 15

Write a PL/SQL program to display the employee IDs, names, and job history end dates of all employees.

```
Declare  v_employee_id  employee_id %Type ;
         v_first_name   employee_first_name %TYPE ;
         v_last_name    employees.last_name %Type ;
         v_end_date     job_history.end_date %Type ;
cursor emp_cur IS
select  e.employee_id e.First name, e.last_name
    j.end_date from employees e JOIN
    job_history j ON e.employee_id = j.employee_id ;
```

Evaluation Procedure	Marks awarded
PL/SQL Procedure(5)	
Program/Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	