**Database Management Systems**

**Lab Cycle-III (PL/SQL-Cursors, Functions, Procedures, Triggers and Packages)**

**CURSORS:**

1. Write the PL/SQL script to display the employee name, job, salary and department number from the employee table.

|  |
| --- |
| SET SERVEROUTPUT ON;  DECLARE  ename employee.e\_name%type;  ejob employee.designation%type;  esalary employee.salary%type;  edeptid employee.dept\_id%type;  CURSOR e\_cursor IS  SELECT e\_name,designation,salary,dept\_id from employee;  BEGIN  OPEN e\_cursor;  LOOP  FETCH e\_cursor INTO ename,ejob,esalary,edeptid;  exit WHEN e\_cursor%NOTFOUND;  dbms\_output.Put\_line('Employee Name:'||ename||'\tJob Title:'||ejob||  'Employee Salary : ' || esalary || 'Employee Dept Id' || edeptid );  END LOOP;  CLOSE e\_cursor;  END; |

1. Write a PL/SQL script to increase the salary as per following criteria: SALARY AMT INCREMENTED BY <1200 8% , <2500 12% , <4500 15% , OTHERWISE 20%

|  |
| --- |
| SET SERVEROUTPUT ON;  CREATE TABLE emp\_temp AS  SELECT \*  FROM employee;  DECLARE  e\_id emp\_temp.eid%type;  ename emp\_temp.e\_name%type;  esalary emp\_temp.salary%type;  inc number;  CURSOR e\_cursor IS  SELECT eid,e\_name,salary from emp\_temp;  BEGIN  OPEN e\_cursor;  dbms\_output.put\_line(rpad('Employee ID',15)||rpad('Name',25) || 'New Salary %');  dbms\_output.Put\_line('------------------------------------------------------');  LOOP  FETCH e\_cursor INTO e\_id,ename,esalary;  exit WHEN e\_cursor%NOTFOUND;  if esalary < 50000 THEN  inc := esalary \* 0.8;  elsif esalary <60000 then  inc := esalary \* 1.2;  elsif esalary <70000 then  inc := esalary \* 2.5;  else  inc := esalary \* 5;  end if;  UPDATE emp\_temp  SET salary = salary + inc  WHERE e\_id = emp\_temp.eid;  dbms\_output.put\_line(rpad(e\_id,15) ||rpad(ename,25)|| esalary);  END LOOP;  CLOSE e\_cursor;  END; |

1. Write the PL/SQL script to display the employee name, job, salary of particular department that is input by user using parameter.

|  |
| --- |
| SET SERVEROUTPUT ON;  DECLARE  ename employee.e\_name%type;  edesig employee.designation%type;  esalary employee.salary%type;  edeptid employee.dept\_id%type;  dept number := &dept;  CURSOR e\_cursor IS  SELECT e\_name,edesig,esalary,dept\_id from employee;  BEGIN  OPEN e\_cursor;  dbms\_output.put\_line(rpad('Employee Name',25));  dbms\_output.Put\_line('-------------------------------------------------------');  LOOP  FETCH e\_cursor INTO ename,edesig,esalary,edeptid;  exit WHEN e\_cursor%NOTFOUND;  if edeptid = dept THEN  dbms\_output.put\_line(rpad(ename,15));  end if;  END LOOP;  CLOSE e\_cursor;  END; |

1. Write a PL/SQL script to display the name, salary and bonus (salary \* .12) for each employee using cursor for loop.

|  |
| --- |
| SET SERVEROUTPUT ON;  CREATE TABLE emp\_temp1 AS  SELECT \*  FROM employee;  DECLARE  e\_id emp\_temp1.eid%type;  ename emp\_temp1.e\_name%type;  esalary emp\_temp1.salary%type;  bonus number;  CURSOR e\_cursor IS  SELECT eid,e\_name,salary from emp\_temp1;  BEGIN  OPEN e\_cursor;  dbms\_output.put\_line(rpad('Employee ID',15)||rpad('Name',25) ||rpad('Salary',25) || 'Bonus %');  dbms\_output.Put\_line('------------------------------------------------------');  LOOP  FETCH e\_cursor INTO e\_id,ename,esalary;  exit WHEN e\_cursor%NOTFOUND;  bonus := esalary + (esalary\* 1.2);  dbms\_output.put\_line(rpad(e\_id,15) ||rpad(ename,25)|| rpad(esalary,25)|| bonus);  END LOOP;  CLOSE e\_cursor;  END; |

**PROCEDURE**

1. Write a PL/SQL procedure called Multi\_table that take two numbers as parameter and display the product of first number till second number;

|  |
| --- |
| SET SERVEROUTPUT ON;  DECLARE  n1 number:=&n1;  n2 number:=&n2;  PROCEDURE Multi\_Table (a number,b number) IS  BEGIN  FOR i IN a..b LOOP  FOR j IN 1..10 LOOP  dbms\_output.put\_line(i\*j);  END loop;  END loop;  END;  BEGIN  Multi\_table(n1,n2);  END; |

1. Write a PL/SQL procedure that take the department number as parameter and display the name and salary of employees working in that department and return the sum of salary of such employees using out parameter.

|  |
| --- |
| SET SERVEROUTPUT ON;  DECLARE  dep number;  summ number;  PROCEDURE sum\_salary (dep IN number,salsum OUT number) IS  e\_id employee.eid%type;  ename employee.e\_name%type;  esalary employee.salary%type;  CURSOR e\_cursor IS  SELECT eid,e\_name,salary from employee where dept\_id = dep;    BEGIN  OPEN e\_cursor;  dbms\_output.put\_line(rpad('Employee ID',15)||rpad('Name',25) ||rpad('Salary',25));  dbms\_output.Put\_line('------------------------------------------------------');  LOOP  FETCH e\_cursor INTO e\_id,ename,esalary;  exit WHEN e\_cursor%NOTFOUND;  salsum := esalary + salsum;  dbms\_output.put\_line(rpad(e\_id,15) ||rpad(ename,25)|| rpad(esalary,25));  END LOOP;  CLOSE e\_cursor;  END;  BEGIN  dbms\_output.put\_line('Enter department id');  dep := &dep;  sum\_salary(dep,summ);  dbms\_output.put\_line('Sum of the salary of the department' || summ);  END; |

1. Write a procedure raise\_sal, which increases the salary of an employee. It accepts employee’s number and salary increment amount.

|  |
| --- |
| SET SERVEROUTPUT ON;  DECLARE  empid number;  incre number;  PROCEDURE raise\_sal (empi IN number,inc IN number) IS  e\_id employee.eid%type;  ename employee.e\_name%type;  esalary employee.salary%type;  CURSOR e\_cursor IS  SELECT eid,e\_name,salary from employee where eid = empi;    BEGIN  OPEN e\_cursor;  dbms\_output.put\_line(rpad('Employee ID',15)||rpad('Name',25) ||rpad('Salary',25));  dbms\_output.Put\_line('----------------------------------------------------');  LOOP  FETCH e\_cursor INTO e\_id,ename,esalary;  exit WHEN e\_cursor%NOTFOUND;  esalary := esalary + inc;  dbms\_output.put\_line(rpad(e\_id,15) ||rpad(ename,25)|| rpad(esalary,25));  END LOOP;  CLOSE e\_cursor;  END;  BEGIN  dbms\_output.put\_line('Enter Employee id');  empid := &empid;  dbms\_output.put\_line('Enter Increment in salary');  incre := &incre;  raise\_sal(empid,incre);    END; |

**FUNCTION:**

1. Write a PL/SQL function power that takes two numbers as arguments and returns the value of the first number raised to the power of the second.

|  |
| --- |
| SET SERVEROUTPUT ON;  DECLARE  a number;  b number;  c number;  FUNCTION findPower(x IN number, y IN number)  RETURN number  IS  z number;  BEGIN  z := power(x,y);  RETURN z;  END;  BEGIN  a:= &a;  b:= &b;  c := findPower(a, b);  dbms\_output.put\_line(' Power of ('|| a || ' and ' || b || ' is : ' || c);  END; |

**TRIGGERS:**

1. Write a set of triggers to maintain the employee name and department name fields redundantly in the employee-department relation , so that you don’t have to join the employee and department tables just to get a simple department listing.
2. Write a trigger that verifies the joining date when a new row is inserted in the Employee table. Joining date should be greater or equal to current date.
3. Write a trigger that is fired before any row is inserted in the Employee table.

|  |
| --- |
| SET SERVEROUTPUT ON;  CREATE TABLE emp\_temp AS  SELECT \*  FROM employee;  CREATE TRIGGER my\_trigger  before INSERT on employee  FOR EACH ROW  DECLARE  e\_id emp\_temp.eid%type;  ename emp\_temp.e\_name%type ;  sal emp\_temp.salary%type ;  des emp\_temp.designation%type;  dep emp\_temp.dept\_id%type;  BEGIN  insert INTO emp\_temp (eid,e\_name,salary,designation,dept\_id)  values (&e\_id,'&ename',&sal,'&des',&dep);  commit;  dbms\_output.put\_line('Record Inserted');  END; |