Queries

1. Display details of jobs where the minimum salary is greater than 10000.

```
SELECT * FROM JOBS WHERE MIN_SALARY > 10000
```

2. Display the first name and join date of the employees who joined between 2002 and 2005.

```
SELECT FIRST_NAME, HIRE_DATE FROM EMPLOYEES
WHERE TO_CHAR(HIRE_DATE, 'YYYY') BETWEEN 2002 AND 2005 ORDER BY
HIRE_DATE
```

3. Display first name and join date of the employees who is either IT Programmer or Sales Man.

```
SELECT FIRST_NAME, HIRE_DATE
FROM EMPLOYEES WHERE JOB_ID IN ('IT_PROG', 'SA_MAN')
```

4. Display employees who joined after 1st January 2008.

```
SELECT * FROM EMPLOYEES  where hire_date > '01-jan-2008'
```

5. Display details of employee with ID 150 or 160.

```
SELECT * FROM EMPLOYEES WHERE EMPLOYEE_ID in (150,160)
```

6. Display first name, salary, commission pct, and hire date for employees with salary less than 10000.

```
SELECT FIRST_NAME, SALARY, COMMISSION_PCT, HIRE_DATE FROM EMPLOYEES
WHERE SALARY < 10000
```

7. Display job Title, the difference between minimum and maximum salaries for jobs with max salary in the range 10000 to 20000.

```
SELECT JOB_TITLE, MAX_SALARY-MIN_SALARY DIFFERENCE FROM JOBS WHERE
MAX_SALARY BETWEEN 10000 AND 20000
```

8. Display first name, salary, and round the salary to thousands.

```
SELECT FIRST_NAME, SALARY, ROUND(SALARY, -3) FROM EMPLOYEES
```

9. Display details of jobs in the descending order of the title.

```
SELECT * FROM JOBS ORDER BY JOB_TITLE
```

10. Display employees where the first name or last name starts with S.

```
SELECT FIRST_NAME, LAST_NAME FROM EMPLOYEES WHERE  FIRST_NAME  LIKE
'S%' OR LAST_NAME LIKE 'S%'
```

11. Display employees who joined in the month of May.

```
SELECT * FROM EMPLOYEES WHERE TO_CHAR(HIRE_DATE, 'MON')= 'MAY'
```

12. Display details of the employees where commission percentage is null and salary in the range 5000 to 10000 and department is 30.

```
SELECT * FROM EMPLOYEES WHERE COMMISSION_PCT IS NULL AND SALARY BETWEEN
5000 AND 10000 AND DEPARTMENT_ID=30
```

13. Display first name and date of first salary of the employees.

```
SELECT FIRST_NAME, HIRE_DATE, LAST_DAY(HIRE_DATE)+1 FROM EMPLOYEES
```

14. Display first name and experience of the employees.

```
SELECT FIRST_NAME, HIRE_DATE, FLOOR((SYSDATE-HIRE_DATE)/365)FROM
EMPLOYEES
```

15. Display first name of employees who joined in 2001.

```
SELECT FIRST_NAME, HIRE_DATE FROM EMPLOYEES WHERE TO_CHAR(HIRE_DATE,
'YYYY')=2001
```

16. Display first name and last name after converting the first letter of each name to upper case and the rest to lower case.

```
SELECT INITCAP(FIRST_NAME), INITCAP(LAST_NAME) FROM EMPLOYEES
```

17. Display the first word in job title.

```
SELECT JOB_TITLE,  SUBSTR(JOB_TITLE,1, INSTR(JOB_TITLE, ' ')-1) FROM
JOBS
```

18. Display the length of first name for employees where last name contain character ‘b’ after 3rd position.

```
SELECT FIRST_NAME, LAST_NAME FROM EMPLOYEES WHERE INSTR(LAST_NAME,'B')
> 3
```

19. Display first name in upper case and email address in lower case for employees where the first name and email address are same irrespective of the case.

```
SELECT UPPER(FIRST_NAME), LOWER(EMAIL) FROM EMPLOYEES WHERE
UPPER(FIRST_NAME)= UPPER(EMAIL)
```

20. Display employees who joined in the current year.

```
SELECT * FROM EMPLOYEES WHERE
TO_CHAR(HIRE_DATE,'YYYY')=TO_CHAR(SYSDATE, 'YYYY')
```

21. Display the number of days between system date and 1st January 2011.

```
SELECT SYSDATE - to_date('01-jan-2011') FROM DUAL
```

22. Display how many employees joined in each month of the current year.

```
SELECT TO_CHAR(HIRE_DATE,'MM'), COUNT (*) FROM EMPLOYEES
WHERE TO_CHAR(HIRE_DATE,'YYYY')= TO_CHAR(SYSDATE,'YYYY') GROUP BY
TO_CHAR(HIRE_DATE,'MM')
```

23. Display manager ID and number of employees managed by the manager.

```
 SELECT MANAGER_ID, COUNT(*) FROM EMPLOYEES GROUP BY MANAGER_ID
```

24. Display employee ID and the date on which he ended his previous job.

```
 SELECT EMPLOYEE_ID, MAX(END_DATE) FROM JOB_HISTORY GROUP BY
EMPLOYEE_ID
```

25. Display number of employees joined after 15th of the month.

```
SELECT COUNT(*) FROM EMPLOYEES WHERE TO_CHAR(HIRE_DATE,'DD') > 15
```

26. Display the country ID and number of cities we have in the country.

```
SELECT COUNTRY_ID,  COUNT(*)  FROM LOCATIONS GROUP BY COUNTRY_ID
```

27. Display average salary of employees in each department who have commission percentage.

```
SELECT DEPARTMENT_ID, AVG(SALARY) FROM EMPLOYEES
WHERE COMMISSION_PCT IS NOT NULL GROUP BY DEPARTMENT_ID
```

28. Display job ID, number of employees, sum of salary, and difference between highest salary and lowest salary of the employees of the job.

```
SELECT JOB_ID, COUNT(*), SUM(SALARY), MAX(SALARY)-MIN(SALARY) SALARY
FROM EMPLOYEES GROUP BY JOB_ID
```

29. Display job ID for jobs with average salary more than 10000.

```
SELECT JOB_ID, AVG(SALARY) FROM EMPLOYEES
GROUP BY JOB_ID
HAVING AVG(SALARY)>10000
```

30. Display years in which more than 10 employees joined.

```
SELECT TO_CHAR(HIRE_DATE,'YYYY') FROM EMPLOYEES
GROUP BY TO_CHAR(HIRE_DATE,'YYYY')
HAVING COUNT(EMPLOYEE_ID) > 10
```

31. Display departments in which more than five employees have commission percentage.

```
SELECT DEPARTMENT_ID FROM EMPLOYEES
WHERE COMMISSION_PCT IS NOT NULL
GROUP BY DEPARTMENT_ID
HAVING COUNT(COMMISSION_PCT)>5
```

32. Display employee ID for employees who did more than one job in the past.

```
SELECT EMPLOYEE_ID FROM JOB_HISTORY GROUP BY EMPLOYEE_ID HAVING
COUNT(*) > 1
```

33. Display job ID of jobs that were done by more than 3 employees for more than 100 days.

```
SELECT JOB_ID FROM JOB_HISTORY
WHERE END_DATE-START_DATE > 100
GROUP BY JOB_ID
HAVING COUNT(*)>3
```

34. Display department ID, year, and Number of employees joined.

```
SELECT DEPARTMENT_ID, TO_CHAR(HIRE_DATE,'YYYY'), COUNT(EMPLOYEE_ID)
FROM EMPLOYEES
GROUP BY DEPARTMENT_ID, TO_CHAR(HIRE_DATE, 'YYYY')
ORDER BY DEPARTMENT_ID
```

35. Display departments where any manager is managing more than 5 employees.

```
SELECT DISTINCT DEPARTMENT_ID
FROM EMPLOYEES
GROUP BY DEPARTMENT_ID, MANAGER_ID
HAVING COUNT(EMPLOYEE_ID) > 5
```

36. Change salary of employee 115 to 8000 if the existing salary is less than 6000.

```
UPDATE EMPLOYEES SET SALARY = 8000 WHERE EMPLOYEE_ID = 115 AND SALARY <
6000
```

37. Insert a new employee into employees with all the required details.

```
INSERT INTO EMPLOYEES  (EMPLOYEE_ID, FIRST_NAME, LAST_NAME, EMAIL,
PHONE_NUMBER, HIRE_DATE,JOB_ID, SALARY, DEPARTMENT_ID)
VALUES (207, 'ANGELA', 'SNYDER','ANGELA','215 253 4737', SYSDATE,
'SA_MAN', 12000, 80)
```

38. Delete department 20.

```
DELETE FROM DEPARTMENTS WHERE DEPARTMENT_ID=20
```

39. Change job ID of employee 110 to IT_PROG if the employee belongs to department 10 and the existing job ID does not start with IT.

```
UPDATE EMPLOYEES SET JOB_ID= 'IT_PROG'
WHERE EMPLOYEE_ID=110 AND DEPARTMENT_ID=10 AND NOT JOB_ID LIKE 'IT%'
```

40. Insert a row into departments table with manager ID 120 and location ID in any location ID for city Tokyo.

```
INSERT INTO DEPARTMENTS (150,'SPORTS',120,1200)
```

41. Display department name and number of employees in the department.

```
SELECT DEPARTMENT_NAME, COUNT(*) FROM EMPLOYEES NATURAL JOIN
DEPARTMENTS GROUP BY DEPARTMENT_NAME
```

42. Display job title, employee ID, number of days between ending date and starting date for all jobs in department 30 from job history.

```
SELECT EMPLOYEE_ID, JOB_TITLE, END_DATE-START_DATE DAYS
FROM JOB_HISTORY NATURAL JOIN JOBS
WHERE DEPARTMENT_ID=30
```

43. Display department name and manager first name.

```
SELECT DEPARTMENT_NAME, FIRST_NAME FROM DEPARTMENTS D JOIN EMPLOYEES E
ON (D.MANAGER_ID=E.EMPLOYEE_ID)
```

44. Display department name, manager name, and city.

```
SELECT DEPARTMENT_NAME, FIRST_NAME, CITY FROM DEPARTMENTS D JOIN
EMPLOYEES E ON (D.MANAGER_ID=E.EMPLOYEE_ID) JOIN LOCATIONS L USING
(LOCATION_ID)
```

45. Display country name, city, and department name.

```
SELECT COUNTRY_NAME, CITY, DEPARTMENT_NAME
FROM COUNTRIES JOIN LOCATIONS USING (COUNTRY_ID)
JOIN DEPARTMENTS USING (LOCATION_ID)
```

46. Display job title, department name, employee last name, starting date for all jobs from 2000 to 2005.

```
SELECT JOB_TITLE, DEPARTMENT_NAME, LAST_NAME, START_DATE
FROM JOB_HISTORY JOIN JOBS USING (JOB_ID) JOIN DEPARTMENTS
USING (DEPARTMENT_ID) JOIN  EMPLOYEES USING (EMPLOYEE_ID)
WHERE TO_CHAR(START_DATE,'YYYY') BETWEEN 2000 AND 2005
```

47. Display job title and average salary of employees

```
SELECT JOB_TITLE, AVG(SALARY) FROM EMPLOYEES
NATURAL JOIN JOBS GROUP BY JOB_TITLE
```

48. Display job title, employee name, and the difference between maximum salary for the job and salary of the employee.

```
SELECT JOB_TITLE, FIRST_NAME, MAX_SALARY-SALARY DIFFERENCE FROM
EMPLOYEES NATURAL JOIN JOBS
```

49. Display last name, job title of employees who have commission percentage and belongs to department 30.

```
SELECT JOB_TITLE, FIRST_NAME, MAX_SALARY-SALARY DIFFERENCE FROM
EMPLOYEES NATURAL JOIN JOBS WHERE DEPARTMENT_ID  = 30
```

50. Display details of jobs that were done by any employee who is currently drawing more than 15000 of salary.

```
SELECT JH.*
FROM  JOB_HISTORY JH JOIN EMPLOYEES E ON (JH.EMPLOYEE_ID =
E.EMPLOYEE_ID)
WHERE SALARY > 15000
```

51. Display department name, manager name, and salary of the manager for all managers whose experience is more than 5 years.

```
SELECT DEPARTMENT_NAME, FIRST_NAME, SALARY
FROM DEPARTMENTS D JOIN EMPLOYEES E ON (D.MANAGER_ID=E.MANAGER_ID)
WHERE  (SYSDATE-HIRE_DATE) / 365 > 5
```

52. Display employee name if the employee joined before his manager.

```
SELECT FIRST_NAME FROM  EMPLOYEES E1 JOIN EMPLOYEES E2 ON
(E1.MANAGER_ID=E2.EMPLOYEE_ID)
WHERE E1.HIRE_DATE < E2.HIRE_DATE
```

53. Display employee name, job title for the jobs employee did in the past where the job was done less than six months.

```
SELECT FIRST_NAME, JOB_TITLE FROM EMPLOYEES E JOIN JOB_HISTORY  JH ON
(JH.EMPLOYEE_ID = E.EMPLOYEE_ID) JOIN JOBS J  ON( JH.JOB_ID = J.JOB_ID)
WHERE  MONTHS_BETWEEN(END_DATE,START_DATE)  < 6
```

54. Display employee name and country in which he is working.

```
SELECT FIRST_NAME, COUNTRY_NAME FROM EMPLOYEES JOIN DEPARTMENTS
USING(DEPARTMENT_ID)
JOIN LOCATIONS USING( LOCATION_ID)
JOIN COUNTRIES USING ( COUNTRY_ID)
```

55. Display department name, average salary and number of employees with commission within the department.

```
SELECT DEPARTMENT_NAME, AVG(SALARY), COUNT(COMMISSION_PCT)
FROM DEPARTMENTS JOIN EMPLOYEES USING (DEPARTMENT_ID)
GROUP BY DEPARTMENT_NAME
```

56. Display the month in which more than 5 employees joined in any department located in Sydney.

```
SELECT TO_CHAR(HIRE_DATE,'MON-YY')
FROM EMPLOYEES JOIN DEPARTMENTS USING (DEPARTMENT_ID) JOIN  LOCATIONS
USING (LOCATION_ID)
WHERE  CITY = 'Seattle'
GROUP BY TO_CHAR(HIRE_DATE,'MON-YY')
HAVING COUNT(*) > 5
```

57. Display details of departments in which the maximum salary is more than 10000.

```
SELECT * FROM DEPARTMENTS WHERE DEPARTMENT_ID IN
( SELECT DEPARTMENT_ID FROM EMPLOYEES
  GROUP BY DEPARTMENT_ID
  HAVING MAX(SALARY)>10000)
```

58. Display details of departments managed by ‘Smith’.

```
SELECT * FROM DEPARTMENTS WHERE MANAGER_ID IN
  (SELECT EMPLOYEE_ID FROM EMPLOYEES WHERE FIRST_NAME='SMITH')
```

59. Display jobs into which employees joined in the current year.

```
SELECT * FROM JOBS WHERE JOB_ID IN
       (SELECT JOB_ID FROM EMPLOYEES WHERE
TO_CHAR(HIRE_DATE,'YYYY')=TO_CHAR(SYSDATE,'YYYY'))
```

60. Display employees who did not do any job in the past.

```
SELECT * FROM EMPLOYEES WHERE EMPLOYEE_ID NOT IN
       (SELECT EMPLOYEE_ID FROM JOB_HISTORY)
```

61. Display job title and average salary for employees who did a job in the past.

```
SELECT JOB_TITLE, AVG(SALARY) FROM JOBS NATURAL JOIN EMPLOYEES
GROUP BY JOB_TITLE
WHERE EMPLOYEE_ID IN
    (SELECT EMPLOYEE_ID FROM JOB_HISTORY)
```

62. Display country name, city, and number of departments where department has more than 5 employees.

```
SELECT COUNTRY_NAME, CITY, COUNT(DEPARTMENT_ID)
FROM COUNTRIES JOIN LOCATIONS USING (COUNTRY_ID) JOIN DEPARTMENTS USING
(LOCATION_ID)
WHERE DEPARTMENT_ID IN
    (SELECT DEPARTMENT_ID FROM EMPLOYEES
   GROUP BY DEPARTMENT_ID
   HAVING COUNT(DEPARTMENT_ID)>5)
GROUP BY COUNTRY_NAME, CITY;
```

63. Display details of manager who manages more than 5 employees.

```
SELECT FIRST_NAME FROM EMPLOYEES
WHERE EMPLOYEE_ID IN
(SELECT MANAGER_ID FROM EMPLOYEES
 GROUP BY MANAGER_ID
 HAVING COUNT(*)>5)
```

64. Display employee name, job title, start date, and end date of past jobs of all employees with commission percentage null.

```
SELECT FIRST_NAME, JOB_TITLE, START_DATE, END_DATE
FROM JOB_HISTORY JH JOIN JOBS J USING (JOB_ID) JOIN EMPLOYEES E  ON (
JH.EMPLOYEE_ID = E.EMPLOYEE_ID)
WHERE COMMISSION_PCT IS NULL
```

65. Display the departments into which no employee joined in last two years.

```
SELECT  * FROM DEPARTMENTS
WHERE DEPARTMENT_ID NOT IN
( SELECT DEPARTMENT_ID FROM EMPLOYEES WHERE FLOOR((SYSDATE-
HIRE_DATE)/365) < 2)
```

66. Display the details of departments in which the max salary is greater than 10000 for employees who did a job in the past.

```
SELECT * FROM DEPARTMENTS
WHERE DEPARTMENT_ID IN
(SELECT DEPARTMENT_ID FROM EMPLOYEES
 WHERE EMPLOYEE_ID IN (SELECT EMPLOYEE_ID FROM JOB_HISTORY)
 GROUP BY DEPARTMENT_ID
 HAVING MAX(SALARY) >10000)
```

67. Display details of current job for employees who worked as IT Programmers in the past.

```
SELECT * FROM JOBS
WHERE JOB_ID IN
 (SELECT JOB_ID FROM EMPLOYEES WHERE EMPLOYEE_ID IN
        (SELECT EMPLOYEE_ID FROM JOB_HISTORY WHERE JOB_ID='IT_PROG'))
```

68. Display the details of employees drawing the highest salary in the department.

```
SELECT DEPARTMENT_ID,FIRST_NAME, SALARY FROM EMPLOYEES OUTER WHERE
SALARY =
    (SELECT MAX(SALARY) FROM EMPLOYEES WHERE DEPARTMENT_ID =
OUTER.DEPARTMENT_ID)
```

69. Display the city of employee whose employee ID is 105.

```
SELECT CITY FROM LOCATIONS WHERE LOCATION_ID =
    (SELECT LOCATION_ID FROM DEPARTMENTS WHERE DEPARTMENT_ID =
                (SELECT DEPARTMENT_ID FROM EMPLOYEES WHERE
EMPLOYEE_ID=105)
  )
```

70. Display third highest salary of all employees

```
select salary
from employees main
where  2 = (select count( distinct salary )
            from employees
            where  salary > main.salary)
```

PL/SQL Programs

1. Write a program to interchange the salaries of employee 120 and 122.

```
Declare
   V_salary_120   employees.salary%type;
Begin
  Select  salary into v_salary_120
  From employees where  employee_id = 120;

  Update employees set salary  = ( select salary from employees where
employee_id = 122)
  Where employee_id = 120;

  Update employees set salary  =  v_salary_120  Where employee_id =
122;

  Commit;
End;
```

2. Increase the salary of employee 115 based on the following conditions: If experience is more than 10 years, increase salary by 20% If experience is greater than 5 years, increase salary by 10% Otherwise 5% Case by Expression:

```
declare
    v_exp  number(2);
    v_hike number(5,2);
begin
    select  floor((sysdate-hire_date) / 365 ) into v_exp
    from employees
    where employee_id = 115;

    v_hike := 1.05;

    case
      when  v_exp > 10 then
            v_hike := 1.20;
      when  v_exp > 5  then
            v_hike := 1.10;
    end case;

    update employees set salary = salary * v_hike
    where employee_id = 115;
end;
```

3. Change commission percentage as follows for employee with ID = 150. If salary is more than 10000 then commission is 0.4%, if Salary is less than 10000 but experience is more than 10 years then 0.35%, if salary is less than 3000 then commission is 0.25%. In the remaining cases commission is 0.15%.

```
declare
    v_salary  employees.salary%type;
    v_exp     number(2);
    v_cp      number(5,2);
begin
    select  v_salary,  floor ( (sysdate-hire_date)/365) into v_salary,
v_exp
    from  employees
    where employee_id = 150;

    if v_salary > 10000 then
           v_cp := 0.4;
    elsif  v_exp > 10 then
           v_cp := 0.35;
    elsif  v_salary < 3000 then
           v_cp := 0.25;
    else
           v_cp := 0.15;

    end if;
```

```
    update employees set commission_pct = v_cp
    where employee_id = 150;
end;
```

4. Find out the name of the employee and name of the department for the employee who is managing for employee 103.

```
declare
    v_name     employees.first_name%type;
    v_deptname departments.department_name%type;
begin
    select  first_name , department_name into v_name, v_deptname
    from  employees join departments using (department_id)
    where employee_id = ( select manager_id from employees    where
employee_id = 103);

    dbms_output.put_line(v_name);
    dbms_output.put_line(v_deptname);

end;
```

5. Display missing employee IDs.

```
declare
     v_min  number(3);
     v_max  number(3);
     v_c    number(1);
begin
     select min(employee_id), max(employee_id) into v_min, v_max
     from employees;


     for i in  v_min + 1 .. v_max - 1
     loop
           select count(*) into v_c
           from employees
           where employee_id = i;

           if  v_c = 0 then
                dbms_output.put_line(i);
           end if;
    end loop;

end;
```

6. Display the year in which maximum number of employees joined along with how many joined in each month in that year.

```
declare

      v_year  number(4);
```

```
      v_c     number(2);
begin
      select  to_char(hire_date,'yyyy') into v_year
      from  employees
      group by to_char(hire_date,'yyyy')
      having count(*) =
             ( select  max( count(*))
               from  employees
               group by to_char(hire_date,'yyyy'));

      dbms_output.put_line('Year : ' || v_year);

      for month in 1 .. 12
      loop
          select  count(*) into v_c
          from employees
          where  to_char(hire_date,'mm') = month and
to_char(hire_date,'yyyy') = v_year;

          dbms_output.put_line('Month : ' || to_char(month) || '
Employees : ' || to_char(v_c));

     end loop;

end;
```

7. Change salary of employee 130 to the salary of the employee with first name ‘Joe’. If Joe is not found then take average salary of all employees. If more than one employee with first name ‘Joe’ is found then take the least salary of the employees with first name Joe.

```
declare
    v_salary  employees.salary%type;
begin
     select salary into v_salary
     from employees where first_name = 'Joe';

     update employees set salary = v_salary
     where employee_id = 130;

exception
     when no_data_found then
       update employees set salary = (select avg(salary) from
employees)
     where employee_id = 130;
end;
```

8. Display Job Title and Name of the Employee who joined the job first day.

```
declare
       cursor  jobscur is select  job_id, job_title from jobs;
    v_name  employees.first_name%type;
begin
       for jobrec in jobscur
     loop
```

```
           select first_name into v_name
              from employees
              where hire_date = ( select min(hire_date) from employees
where job_id = jobrec.job_id)
                   and  job_id = jobrec.job_id;

              dbms_output.put_line( jobrec.job_title || '-' || v_name);
     end loop;
end;
```

9. Display 5th and 10th employees in Employees table.

```
declare

     cursor empcur is
        select employee_id, first_name
        from employees;

begin
     for emprec  in empcur
     loop
         if empcur%rowcount > 4 then
              dbms_output.put_line( emprec.first_name);
              exit  when   empcur%rowcount > 10;
         end if;
     end loop;

end;
```

10. Update salary of an employee based on department and commission percentage. If department is 40 increase salary by 10%. If department is 70 then 15%, if commission is more than .3% then 5% otherwise 10%.

```
declare
    cursor empcur is
     select employee_id, department_id, commission_pct
     from employees;

    v_hike  number(2);
begin

    for emprec in empcur
    loop
         if  emprec.department_id = 40 then
              v_hike := 10;
         elsif emprec.department_id = 70 then
              v_hike := 15;
         elsif emprec.commission_pct  > 0.30 then
              v_hike := 5;
         else
              v_hike := 10;
         end if;

         update employees set salary = salary + salary * v_hike/100
         where employee_id = emprec.employee_id;
```

```
    end loop;
end;
```

11. Create a function that takes department ID and returns the name of the manager of the department.

```
create or replace function get_dept_manager_name(deptid number)
return varchar is

   v_name  employees.first_name%type;
begin
   select first_name into v_name
   from employees
   where  employee_id = ( select manager_id from departments where
department_id = deptid);

   return v_name;
end;
```

12. Create a function that takes employee ID and return the number of jobs done by the employee in the past.

```
create or replace function get_no_of_jobs_done(empid number)
return number is
   v_count  number(2);
begin
   select count(*) into v_count
   from job_history
   where  employee_id = empid;

   return v_count;
end;
```

13. Create a procedure that takes department ID and changes the manager ID for the department to the employee in the department with highest salary. (Use Exceptions).

```
create or replace procedure change_dept_manager(deptid number)
is
   v_empid  employees.employee_id%type;
begin
   select employee_id  into v_empid
   from employees
   where  salary = ( select max(salary) from employees where
department_id = deptid)
     and department_id = deptid;

   update departments set manager_id = v_empid
   where  department_id = deptid;
end;
```

14. Create a function that takes a manager ID and return the names of employees who report to this manager. The names must be returned as a string with comma separating names.

```
create or replace function get_employees_for_manager(manager number)
return varchar2
is
   v_employees varchar2(1000) := '';
   cursor empcur is
      select  first_name from employees
      where   manager_id = manager;
begin

     for emprec in empcur
     loop
         v_employees :=  v_employees ||  ',' || emprec.first_name;
     end loop;
     -- remove extra  , at the beginning
     return  ltrim(v_employees,',');
end;
```

15. Ensure no changes can be made to EMPLOYEES table before 6am and after 10pm in a day.

```
create or replace trigger  trg_employees_time_check
before update or insert or delete
on employees
for each row
begin
   if  to_char(sysdate,'hh24') < 6 or to_char(sysdate,'hh24') > 10 then
         raise_application_error(-20111,'Sorry! No change can be made
before 6 AM and after 10 PM');
   end if;
end;
```

16. Create a Trigger to ensure the salary of the employee is not decreased.

```
create or replace trigger  trg_employees_salary_check
before update
on employees
for each row
begin
   if  :old.salary > :new.salary then
         raise_application_error(-20111,'Sorry! Salary can not be
decreased!');
   end if;
end;
```

17. Create a trigger to ensure the employee and manager belongs to the same department.

Note: This trigger need to read the row that is being modified, which causes mutating problem. The solution to mutating problem is explained at : Work around for mutating problem in Oracle Triggers. Please check it out.

18. Whenever the job is changed for an employee write the following details into job history. Employee ID, old job ID, old department ID, hire date of the employee for start date,

system date for end date. But if a row is already present for employee job history then the start date should be the end date of that row +1.

```
 create or replace trigger trg_log_job_change
after update of job_id
on employees
for each row
declare
    v_enddate   date;
    v_startdate date;
begin
   -- find out whether the employee has any row in job_history table
   select max(end_date) into v_enddate
   from job_history
   where employee_id = :old.employee_id;

   if v_enddate is null then
      v_startdate := :old.hire_date;
   else
      v_startdate := v_enddate + 1;
   end if;

   insert into  job_history values (:old.employee_id, v_startdate,
sysdate, :old.job_id, :old.department_id);
end;
```

Note: Before testing the above trigger, you need to disable UPDATE_JOB_HISTORY trigger, which is already present in HR account, as it does the same.