**DBMS LAB 7**

1. Create the following table and insert the following.

-- Create the Job\_History table

CREATE TABLE Job\_History (

Emp\_id NUMBER,

Start\_Date DATE,

End\_Date DATE,

Job\_Type VARCHAR2(50),

D\_Name VARCHAR2(50)

);

-- Insert the values into the Job\_History table

INSERT INTO Job\_History (Emp\_id, Start\_Date, End\_Date, Job\_Type, D\_Name)

VALUES (1, TO\_DATE('4-Jan-1998', 'DD-Mon-YYYY'), TO\_DATE('30-Jun-2001', 'DD-Mon-YYYY'), 'Engineer', 'Production');

INSERT INTO Job\_History (Emp\_id, Start\_Date, End\_Date, Job\_Type, D\_Name)

VALUES (2, TO\_DATE('9-Feb-1998', 'DD-Mon-YYYY'), TO\_DATE('28-Feb-2002', 'DD-Mon-YYYY'), 'Salesman', 'Sales');

INSERT INTO Job\_History (Emp\_id, Start\_Date, End\_Date, Job\_Type, D\_Name)

VALUES (1, TO\_DATE('1-Jul-2001', 'DD-Mon-YYYY'), TO\_DATE('31-Dec-2010', 'DD-Mon-YYYY'), 'Manager', 'R&D');

INSERT INTO Job\_History (Emp\_id, Start\_Date, End\_Date, Job\_Type, D\_Name)

VALUES (4, TO\_DATE('27-Dec-2001', 'DD-Mon-YYYY'), TO\_DATE('19-Sep-2016', 'DD-Mon-YYYY'), 'Sales\_Executive', 'Marketing');

INSERT INTO Job\_History (Emp\_id, Start\_Date, End\_Date, Job\_Type, D\_Name)

VALUES (2, TO\_DATE('1-Mar-2002', 'DD-Mon-YYYY'), TO\_DATE('30-Mar-2015', 'DD-Mon-YYYY'), 'Sales\_Executive', 'Marketing');

INSERT INTO Job\_History (Emp\_id, Start\_Date, End\_Date, Job\_Type, D\_Name)

VALUES (2, TO\_DATE('1-Apr-2016', 'DD-Mon-YYYY'), TO\_DATE('15-Dec-2017', 'DD-Mon-YYYY'), 'Manager', 'Sales');

INSERT INTO Job\_History (Emp\_id, Start\_Date, End\_Date, Job\_Type, D\_Name)

VALUES (4, TO\_DATE('20-Sep-2016', 'DD-Mon-YYYY'), TO\_DATE('16-Dec-2017', 'DD-Mon-YYYY'), 'Asst.Manager', 'Sales');

INSERT INTO Job\_History (Emp\_id, Start\_Date, End\_Date, Job\_Type, D\_Name)

VALUES (6, TO\_DATE('16-Jul-2000', 'DD-Mon-YYYY'), TO\_DATE('30-Nov-2006', 'DD-Mon-YYYY'), 'Clerk', 'Accounts');

INSERT INTO Job\_History (Emp\_id, Start\_Date, End\_Date, Job\_Type, D\_Name)

VALUES (5, TO\_DATE('20-Mar-2002', 'DD-Mon-YYYY'), TO\_DATE('12-Aug-2011', 'DD-Mon-YYYY'), 'Engineer', 'R&D');

INSERT INTO Job\_History (Emp\_id, Start\_Date, End\_Date, Job\_Type, D\_Name)

VALUES (1, TO\_DATE('1-Jan-2011', 'DD-Mon-YYYY'), TO\_DATE('31-Jan-2012', 'DD-Mon-YYYY'), 'Engineer', 'Production');

1. Display the emp\_id along with every job\_type they have worked (including their current job\_type). (use union/union all)

SELECT emp\_id, job\_type FROM Job\_History

UNION ALL

SELECT emp\_id, job\_type FROM Employee;

EMP\_ID JOB\_TYPE

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

1 Engineer

2 Salesman

1 Manager

4 Sales\_Executive

2 Sales\_Executive

2 Manager

4 Asst.Manager

6 Clerk

5 Engineer

1 Engineer

1 MANAGER

EMP\_ID JOB\_TYPE

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

2 MANAGER

3 ENGINEER

4 MANAGER

5 ENGINEER

6 ACCOUNTANT

7 CLERK

8 ENGINEER

10 SALESMAN

11 ENGINEER

13 ENGINEER

14 CLERK

22 rows selected.

1. Display the emp\_id, d\_name, and job\_types current and previous (if any) of all employees. (use union/union all)

SELECT emp\_id, d\_name, job\_type FROM Job\_History

UNION ALL

SELECT emp\_id, dept, job\_type FROM Employee;

EMP\_ID D\_NAME JOB\_TYPE

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

1 Production Engineer

2 Sales Salesman

1 R Manager

4 Marketing Sales\_Executive

2 Marketing Sales\_Executive

2 Sales Manager

4 Sales Asst.Manager

6 Accounts Clerk

5 R Engineer

1 Production Engineer

1 PRODUCTION MANAGER

EMP\_ID D\_NAME JOB\_TYPE

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

2 MARKETING MANAGER

3 PRODUCTION ENGINEER

4 SALES MANAGER

5 PRODUCTION ENGINEER

6 ACCOUNTS ACCOUNTANT

7 ACCOUNTS CLERK

8 R AND D ENGINEER

10 MARKETING SALESMAN

11 R AND D ENGINEER

13 ADMIN ENGINEER

14 CLERK

1. Display the emp\_id and the job\_type of employees who currently have a job title that they held previously. (use intersect)

SELECT emp\_id, job\_type FROM Job\_History

INTERSECT

SELECT emp\_id, job\_type FROM Employee;

1. Find the employees who have not changed their job for once. (use minus)

SELECT emp\_id, job\_type FROM Employee

MINUS

SELECT emp\_id, job\_type FROM Job\_History;

1. Find the employees who earn more than Chitra. (use single-row subquery)

SELECT \* FROM Employee

WHERE salary > (SELECT salary FROM Employee WHERE f\_name = 'CHITRA');

1. Find the employees details who have the same job\_type as of emp\_id 7. (use single-row subquery)

SELECT \* FROM Employee

WHERE job\_type = (SELECT job\_type FROM Employee WHERE emp\_id = 7);

1. Display the employee names whose job is the same as employee 3 and earn more than employee 7. (use single-row subquery)

SELECT f\_name, l\_name FROM Employee

WHERE job\_type = (SELECT job\_type FROM Employee WHERE emp\_id = 3)

AND salary > (SELECT salary FROM Employee WHERE emp\_id = 7);

1. Display the employees earning less than the average salary. (use single-row subquery)

SELECT \* FROM Employee

WHERE salary < (SELECT AVG(salary) FROM Employee);

1. Find the job\_type with the lowest average salary. (use single-row subquery)

SELECT job\_type FROM Employee

GROUP BY job\_type

HAVING AVG(salary) = (

SELECT MIN(AVG(salary)) FROM Employee

GROUP BY job\_type

);

JOB\_TYPE

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

SALESMAN

1. Display all the department names whose minimum salary is greater than the minimum salary of the Sales department.

SELECT DISTINCT DEPT FROM Employee

WHERE salary > (SELECT MIN(salary) FROM Employee WHERE DEPT = 'SALES');

DEPT

----------

PRODUCTION

MARKETING

1. Select the employee names, department, and salary who are the lowest earners of their corresponding department (use multi-row subquery).

SELECT F\_NAME, L\_NAME, DEPT, SALARY

FROM EMPLOYEE e

WHERE SALARY = (

SELECT MIN(SALARY)

FROM EMPLOYEE

WHERE DEPT = e.DEPT

);

1. Find the highest earners of each job\_type. (use multi-row subquery).

SELECT F\_NAME, L\_NAME, JOB\_TYPE, SALARY

FROM EMPLOYEE e

WHERE (JOB\_TYPE, SALARY) IN (

SELECT JOB\_TYPE, MAX(SALARY)

FROM EMPLOYEE

GROUP BY JOB\_TYPE

);

1. Display the employees who are not engineers and earn less than any engineer. (use multi-row subquery).

SELECT F\_NAME, L\_NAME, JOB\_TYPE, SALARY

FROM EMPLOYEE

WHERE JOB\_TYPE <> 'ENGINEER'

AND SALARY < (

SELECT MIN(SALARY)

FROM EMPLOYEE

WHERE JOB\_TYPE = 'ENGINEER'

);

1. Display the employees who are not clerks but earn more than all clerks. (use multi-row subquery)

SELECT F\_NAME, L\_NAME, JOB\_TYPE, SALARY

FROM EMPLOYEE

WHERE JOB\_TYPE <> 'CLERK'

AND SALARY > ALL (

SELECT SALARY

FROM EMPLOYEE

WHERE JOB\_TYPE = 'CLERK'

);

1. Display the top 5 highest earning employees.

SELECT \*

FROM (

SELECT e.\*, ROW\_NUMBER() OVER (ORDER BY SALARY DESC) AS rn

FROM EMPLOYEE e

)

WHERE rn <= 5;

1. Display the name and department of the top 2 highest paid managers.

SELECT F\_NAME, L\_NAME, DEPT

FROM (

SELECT e.\*, ROW\_NUMBER() OVER (PARTITION BY DEPT ORDER

BY SALARY DESC) AS rn

FROM EMPLOYEE e

WHERE JOB\_TYPE = 'MANAGER'

)

WHERE rn <= 2;

1. Update the salary of the employees working as managers to the average salary of all the employees.

UPDATE EMPLOYEE

SET SALARY = (

SELECT AVG(SALARY)

FROM EMPLOYEE

)

WHERE JOB\_TYPE = 'MANAGER';