# **Lab Assignment 7**

Q1. Create the following table and insert the values. Job\_History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Emp\_id | Start\_Date | End\_Date | Job\_Type | D\_Name |
| 1 | 4-Jan-1998 | 30-Jun-2001 | Engineer | Production |
| 2 | 9-Feb-1998 | 28-Feb-2002 | Salesman | Sales |
| 1 | 1-Jul-2001 | 31-Dec-2010 | Manager | R&D |
| 4 | 27-Dec-2001 | 19-Sep-2016 | Sales\_Executive | Marketing |
| 2 | 1-Mar-2002 | 30-Mar-2015 | Sales\_Executive | Marketing |
| 2 | 1-Apr-2016 | 15-Dec-2017 | Manager | Sales |
| 4 | 20-Sep-2016 | 16-Dec-2017 | Asst.Manager | Sales |
| 6 | 16-Jul-2000 | 30-Nov-2006 | Clerk | Accounts |
| 5 | 20-Mar-2002 | 12-Aug-2011 | Engineer | R&D |
| 1 | 1-Jan-2011 | 31-Jan-2012 | Engineer | Production |

Using the above Job\_History table and the Employee table (of assignment 2) write SQL statements for the following queries.

🡺 CREATE TABLE Job\_History ( Emp\_id INT, Start\_Date DATE, End\_Date DATE, Job\_Type VARCHAR(20), D\_Name VARCHAR(20));

INSERT INTO Job\_History VALUES (&Emp\_id, &Start\_Date, &End\_Date, ‘&Job\_Type’, ‘&D\_Name’); VALUES

(1, '1998-01-04', '2001-06-30', 'Engineer', 'Production'),

(2, '1998-02-09', '2002-02-28', 'Salesman', 'Sales'),

(1, '2001-07-01', '2010-12-31', 'Manager', 'R&D'),

(4, '2001-12-27', '2016-09-19', 'Sales\_Executive', 'Marketing'),

(2, '2002-03-01', '2015-03-30', 'Sales\_Executive', 'Marketing'),

(2, '2016-04-01', '2017-12-15', 'Manager', 'Sales'),

(4, '2016-09-20', '2017-12-16', 'Asst.Manager', 'Sales'),

(6, '2000-07-16', '2006-11-30', 'Clerk', 'Accounts'),

(5, '2002-03-20', '2011-08-12', 'Engineer', 'R&D'),

(1, '2011-01-01', '2012-01-31', 'Engineer', 'Production');

Q2. 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 SELECT Emp\_id, Job\_Type FROM Employee;

Q3. 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 SELECT Emp\_id, D\_Name, Job\_Type FROM Employee;

Q4. 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;

Q5. Find the employees who have not changed their job for once.(use minus)

🡺 SELECT Emp\_id FROM Employee MINUS (SELECT Emp\_id FROM Job\_History);

Q6. 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');

Q7. 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);

Q8.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);

Q9. Display the employees earning less than the average salary. (use single-row subquery)

🡺 SELECT \* FROM Employee WHERE Salary < (SELECT AVG(Salary) FROM Employee);

Q10. 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);

Q11. Display all the department names whose minimum salary is greater than the minimum salary of the Sales department.

🡺 SELECT D\_Name FROM Employee WHERE MIN(Salary) > (SELECT MIN(Salary) FROM Employee WHERE D\_Name = 'Sales');

Q12. 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, D\_Name, Salary FROM Employee e WHERE Salary = (SELECT MIN(Salary) FROM Employee WHERE D\_Name = e.D\_Name);

Q13. Find the highest earners of each job\_type.(use multi-row subquery).

🡺 SELECT \* FROM Employee e WHERE Salary = (SELECT MAX(Salary) FROM Employee WHERE Job\_Type = e.Job\_Type);

Q14. Display the employees who are not engineers and earn less than any engineer.(use multi-row subquery).

🡺 SELECT \* FROM Employee WHERE Job\_Type != 'Engineer' AND Salary < (SELECT MIN(Salary) FROM Employee WHERE Job\_Type = 'Engineer');

Q15. Display the employees who are not clerks but earn more than all clerks.(use multi-row subquery).

🡺 SELECT \* FROM Employee WHERE Job\_Type != 'Clerk' AND Salary > ALL (SELECT Salary FROM Employee WHERE Job\_Type = 'Clerk');

Q16. Display the top 5 highest earning employees.

🡺 SELECT \* FROM Employee ORDER BY Salary DESC LIMIT 5;

Q17. Display the name and department of the top 2 highest paid managers.

🡺 SELECT F\_name, L\_name, D\_Name FROM Employee WHERE Job\_Type = 'Manager' ORDER BY Salary DESC LIMIT 2;

Q18. 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';