

### **Tables Starting with 'JO' Owned by Current User:**

sql

Copy code

```
SELECT table_name
FROM user_tables
WHERE table_name LIKE 'JO%';
```

### **First Initial and Last Name of Employees:**

```
SELECT SUBSTR(first_name, 1, 1) || ' ' || last_name AS employee_name
FROM employees;
```

#### **1. Full Name and Email of Employees with 'IN' in Email Address:**

```
SELECT first_name || ' ' || last_name AS full_name, email
FROM employees
WHERE email LIKE '%IN%';

SELECT MIN(last_name) AS smallest_last_name, MAX(last_name) AS highest_last_name
FROM employees;
```

#### **2. Weekly Salaries Between \$700 and \$3000:**

```
SELECT salary / 52 AS weekly_salary
FROM employees
WHERE salary / 52 BETWEEN 700 AND 3000;
```

#### **3. Employees and Their Job Titles Sorted by Job Title:**

sql

Copy code

```
SELECT e.last_name, j.job_title
FROM employees e
JOIN jobs j ON e.job_id = j.job_id
ORDER BY j.job_title;
```

#### **4. Job, Salary Ranges, and Employee's Salary:**

sql

Copy code

```
SELECT e.job_id, MIN(e.salary) || ' - ' || MAX(e.salary) AS salary_range, e.salary
FROM employees e
GROUP BY e.job_id, e.salary;
```

**5. Employee's Initial, Last Name, and Department Name (ANSI Join):**

sql

Copy code

```
SELECT SUBSTR(e.first_name, 1, 1) || ' ' || e.last_name AS employee_name, d.department_name
FROM employees e
JOIN departments d ON e.department_id = d.department_id;
```

**6. Employees and Department Names with Outer Join:**

sql

Copy code

```
SELECT e.last_name, d.department_name
FROM employees e
RIGHT OUTER JOIN departments d ON e.department_id = d.department_id;
```

**7. Employees' Names, Salary, and Commission Status:**

sql

Copy code

```
SELECT SUBSTR(first_name, 1, 1) || ' ' || last_name AS employee_name, salary,
       DECODE(commission_pct, NULL, 'No', 'Yes') AS commission_status
FROM employees;
```

**8. Employees' Last Name and Department Details with Outer Join:**

sql

Copy code

```
SELECT e.last_name, d.department_name, l.city, l.state_province
FROM employees e
RIGHT OUTER JOIN departments d ON e.department_id = d.department_id
JOIN locations l ON d.location_id = l.location_id;
```

**9. Employee's First and Last Names and First Occurrence of Commission, Manager ID, or -1:**

sql

Copy code

```
SELECT first_name || ' ' || last_name AS full_name,  
       NVL2(commission_pct, commission_pct, NVL2(manager_id, manager_id, -1)) AS first_occurrence  
FROM employees;
```

**10. Employees' Last Name, Salary, and Job Grade for Departments with ID Greater Than 50:**

sql

Copy code

```
SELECT e.last_name, e.salary, jg.grade_level  
FROM employees e  
JOIN job_grades jg ON e.job_id = jg.job_id  
WHERE e.department_id > 50;
```

**11. List of Employees and Departments Including Unassigned:**

sql

Copy code

```
SELECT e.last_name, d.department_name  
FROM employees e  
FULL OUTER JOIN departments d ON e.department_id = d.department_id;
```

**12. Treewalking List of Employees' Last Name, Manager's Last Name, and Position:**

sql

Copy code

```
SELECT e1.last_name AS employee, e2.last_name AS manager,  
       LEVEL AS position  
FROM employees e1  
JOIN employees e2 ON e1.manager_id = e2.employee_id  
START WITH e1.employee_id = 100  
CONNECT BY PRIOR e1.employee_id = e1.manager_id;
```

**13. Earliest Hire Date, Latest Hire Date, and Number of Employees:**

sql

Copy code

```
SELECT MIN(hire_date) AS earliest_hire_date, MAX(hire_date) AS latest_hire_date, COUNT(*) AS  
number_of_employees
```

FROM employees;

**14. Departments with Salary Costs Between 15000 and 31000:**

sql

Copy code

```
SELECT d.department_name, SUM(e.salary) AS total_salary
FROM employees e
JOIN departments d ON e.department_id = d.department_id
GROUP BY d.department_name
HAVING SUM(e.salary) BETWEEN 15000 AND 31000
ORDER BY total_salary;
```

**15. Department Names, Manager ID, Manager Name, and Average Salary:**

sql

Copy code

```
SELECT d.department_name, e.manager_id, e.last_name AS manager_name, AVG(e.salary) AS
average_salary
FROM employees e
JOIN departments d ON e.department_id = d.department_id
GROUP BY d.department_name, e.manager_id, e.last_name;
```

**16. Highest Average Salary in Departments:**

sql

Copy code

```
SELECT ROUND(MAX(avg_salary)) AS highest_avg_salary
FROM (SELECT AVG(salary) AS avg_salary
      FROM employees
      GROUP BY department_id);
```

**17. Monthly Salary Cost by Department:**

sql

Copy code

```
SELECT d.department_name, SUM(e.salary) / 12 AS monthly_cost
FROM employees e
JOIN departments d ON e.department_id = d.department_id
```

GROUP BY d.department\_name;

**18. Monthly Salary Cost by Job ID and Department:**

sql

Copy code

```
SELECT d.department_name, e.job_id, SUM(e.salary) / 12 AS monthly_cost
FROM employees e
JOIN departments d ON e.department_id = d.department_id
GROUP BY d.department_name, e.job_id
WITH ROLLUP;
```

**19. Employee Names, Department IDs, and Department Names with Cities:**

sql

Copy code

```
SELECT first_name || ' ' || last_name AS employee_name, department_id
FROM employees
UNION
SELECT department_id, department_name
FROM departments
UNION
SELECT city
FROM locations;
```

**20. Employees' Names, Salary, and Department Name for Salaries Above Department Average:**

sql

Copy code

```
SELECT first_name || ' ' || last_name AS employee_name, salary, d.department_name
FROM employees e
JOIN departments d ON e.department_id = d.department_id
WHERE e.salary > (SELECT AVG(salary) FROM employees WHERE department_id = e.department_id);
```

These queries should help ensure that you produce the desired data while practicing SQL.

4o

next

ChatGPT

Certainly! Here are more detailed SQL queries for the remaining problems:

**24. List of Every Employee's Job, Salary Ranges Within the Job, and the Employee's Salary:**

sql

Copy code

```
SELECT j.job_title, MIN(e.salary) || ' - ' || MAX(e.salary) AS salary_range, e.salary
FROM employees e
JOIN jobs j ON e.job_id = j.job_id
GROUP BY j.job_title, e.salary;
```

**25. Employees' First Initial and Last Name, and Department Name Using ANSI Join:**

sql

Copy code

```
SELECT SUBSTR(e.first_name, 1, 1) || ' ' || e.last_name AS employee_name, d.department_name
FROM employees e
JOIN departments d ON e.department_id = d.department_id;
```

**26. Previous List Joined Only on Department ID:**

sql

Copy code

```
SELECT SUBSTR(e.first_name, 1, 1) || ' ' || e.last_name AS employee_name, d.department_name
FROM employees e
JOIN departments d ON e.department_id = d.department_id;
```

**27. Employees' Last Name and Whether or Not They Have a Manager Using DECODE:**

sql

Copy code

```
SELECT last_name, DECODE(manager_id, NULL, 'Nobody', 'Somebody') AS manager_status
FROM employees;
```

**28. Employees' First Initial and Last Name, Salary, and Commission Status:**

sql

Copy code

```
SELECT SUBSTR(first_name, 1, 1) || ' ' || last_name AS employee_name, salary,
       DECODE(commission_pct, NULL, 'No', 'Yes') AS commission_status
FROM employees;
```

**29. Employees' Last Name, Department Name, City, and State/Province with Outer Join:**

sql

Copy code

```
SELECT e.last_name, d.department_name, l.city, l.state_province
FROM employees e
RIGHT OUTER JOIN departments d ON e.department_id = d.department_id
JOIN locations l ON d.location_id = l.location_id;
```

**30. Employees' First and Last Names and First Occurrence of Commission, Manager ID, or -1:**

sql

Copy code

```
SELECT first_name || ' ' || last_name AS full_name,
       NVL2(commission_pct, commission_pct, NVL2(manager_id, manager_id, -1)) AS first_occurrence
FROM employees;
```

**31. Employees' Last Name, Salary, and Job Grade for Departments with ID Greater Than 50:**

sql

Copy code

```
SELECT e.last_name, e.salary, jg.grade_level
FROM employees e
JOIN job_grades jg ON e.job_id = jg.job_id
WHERE e.department_id > 50;
```

**32. List of Employees and Departments Including Unassigned:**

sql

Copy code

```
SELECT e.last_name, d.department_name
FROM employees e
FULL OUTER JOIN departments d ON e.department_id = d.department_id;
```

**33. Treewalking List of Employees' Last Name, Manager's Last Name, and Position:**

sql

Copy code

```
SELECT e1.last_name AS employee, e2.last_name AS manager,
       LEVEL AS position
```

```
FROM employees e1  
JOIN employees e2 ON e1.manager_id = e2.employee_id  
START WITH e1.employee_id = 100  
CONNECT BY PRIOR e1.employee_id = e1.manager_id;
```

**34. Earliest Hire Date, Latest Hire Date, and Number of Employees:**

sql

Copy code

```
SELECT MIN(hire_date) AS earliest_hire_date, MAX(hire_date) AS latest_hire_date, COUNT(*) AS  
number_of_employees  
FROM employees;
```

**35. Departments with Salary Costs Between 15000 and 31000:**

sql

Copy code

```
SELECT d.department_name, SUM(e.salary) AS total_salary  
FROM employees e  
JOIN departments d ON e.department_id = d.department_id  
GROUP BY d.department_name  
HAVING SUM(e.salary) BETWEEN 15000 AND 31000  
ORDER BY total_salary;
```

**36. Department Names, Manager ID, Manager Name, and Average Salary:**

sql

Copy code

```
SELECT d.department_name, e.manager_id, e.last_name AS manager_name, AVG(e.salary) AS  
average_salary  
FROM employees e  
JOIN departments d ON e.department_id = d.department_id  
GROUP BY d.department_name, e.manager_id, e.last_name;
```

**37. Highest Average Salary in Departments:**

sql

Copy code

```
SELECT ROUND(MAX(avg_salary)) AS highest_avg_salary  
FROM (SELECT AVG(salary) AS avg_salary
```



```
FROM employees  
GROUP BY department_id);
```

**38. Monthly Salary Cost by Department:**

sql

Copy code

```
SELECT d.department_name, SUM(e.salary) / 12 AS monthly_cost  
FROM employees e  
JOIN departments d ON e.department_id = d.department_id  
GROUP BY d.department_name;
```

**39. Monthly Salary Cost by Job ID and Department:**

sql

Copy code

```
SELECT d.department_name, e.job_id, SUM(e.salary) / 12 AS monthly_cost  
FROM employees e  
JOIN departments d ON e.department_id = d.department_id  
GROUP BY d.department_name, e.job_id  
WITH ROLLUP;
```

**40. Employee Names, Department IDs, and Department Names with Cities:**

sql

Copy code

```
SELECT first_name || ' ' || last_name AS employee_name, department_id  
FROM employees  
UNION  
SELECT department_id, department_name  
FROM departments  
UNION  
SELECT city  
FROM locations;
```

**41. Employees' Names, Salary, and Department Name for Salaries Above Department Average:**

sql

Copy code

```

SELECT first_name || ' ' || last_name AS employee_name, salary, d.department_name
FROM employees e
JOIN departments d ON e.department_id = d.department_id
WHERE e.salary > (SELECT AVG(salary) FROM employees WHERE department_id = e.department_id);

```

**42. Employees' Names and Position in the Company:**

sql

Copy code

```

SELECT e1.last_name AS employee, e2.last_name AS manager, LEVEL AS position
FROM employees e1
JOIN employees e2 ON e1.manager_id = e2.employee_id
START WITH e1.employee_id = 100
CONNECT BY PRIOR e1.employee_id = e1.manager_id;

```

**43. List of Each Employee's First Initial and Last Name, Salary, and Department Name for Each Employee Earning More Than the Average for Their Department:**

sql

Copy code

```

SELECT SUBSTR(e.first_name, 1, 1) || ' ' || e.last_name AS employee_name, e.salary,
d.department_name
FROM employees e
JOIN departments d ON e.department_id = d.department_id
WHERE e.salary > (SELECT AVG(salary)
FROM employees
WHERE department_id = e.department_id);

```

**44. Employees' Last Name, Department Name, City, and State/Province with Outer Join:**

sql

Copy code

```

SELECT e.last_name, d.department_name, l.city, l.state_province
FROM employees e
RIGHT OUTER JOIN departments d ON e.department_id = d.department_id
JOIN locations l ON d.location_id = l.location_id;

```

**45. Departments with Salary Costs Between 15000 and 31000:**

sql

Copy code

```
SELECT d.department_name, SUM(e.salary) AS total_salary
FROM employees e
JOIN departments d ON e.department_id = d.department_id
GROUP BY d.department_name
HAVING SUM(e.salary) BETWEEN 15000 AND 31000
ORDER BY total_salary;
```