**Practicing Queries using Joins**

**Objective:**

Practice retrieving complex data sets that span across multiple related tables, allowing for better understanding of multi-dimensional data analysis.

**Joins: Join, Left Outer Join, Right Outer Join, Self-Join**

Consider the following database tables and write the solution for the given queries.

**Tables: Employee**(eid, ename, salary, doj, comm,did)

**Department**(did, departmentname, location)

**Sample Data in Employees Table:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **eid** | **ename** | **salary** | **doj** | **comm** | **did** |
| 106 | Jim Halpert | 48000 | 2017-09-18 | 400 | 10 |
| 107 | Stanley Hudson | 5200 | 2016-02-23 | 550 | 20 |
| 108 | Phyllis Vance | 16000 | 2015-08-11 | 350 | 30 |
| 109 | Angela Martin | 23000 | 2014-07-07 | 600 | 10 |
| 110 | Kevin Malone | 44000 | 2013-05-15 | 250 | 20 |
| 111 | Meredith Palmer | 24000 | 2012-03-28 | 150 | 30 |
| 112 | Andy Bernard | 36000 | 2011-01-10 | 700 | 10 |
| 113 | Darryl Philbin | 26000 | 2010-12-01 | 450 | 20 |
| 114 | Oscar Martinez | 28000 | 2009-11-17 | 800 | 30 |
| 115 | Toby Flenderson | 42000 | 2008-10-22 | NULL | 10 |
| 116 | Jane Smith | 32000 | 2012-03-28 | 400 | 20 |
| 117 | Alice Brown | 75000 | 2011-01-10 | 550 | 10 |
| 118 | Bob Davis | 48000 | 2010-12-01 | 350 | 30 |
| 119 | Carol White | 67000 | 2009-11-17 | 600 | 40 |

**Departments Table:**

|  |  |  |
| --- | --- | --- |
| **DID** | **DepartmentName** | **Location** |
| 10 | HR | New York |
| 20 | Finance | London |
| 30 | IT | San Francisco |
| 40 | Marketing | Chicago |

**Salgrade Table:**

|  |  |  |
| --- | --- | --- |
| **losal** | **hisal** | **grade** |
| 5000 | 15000 | 1 |
| 15001 | 25000 | 2 |
| 25001 | 35000 | 3 |
| 35001 | 45000 | 4 |
| 45001 | 100000 | 5 |

**INNER JOIN (Join):**

**Q1) Problem Statement: Write a query to retrieve the names of employees along with their corresponding department names.**

**Expected Output:**

|  |  |
| --- | --- |
| **Name** | **DepartmentName** |
| John Doe | HR |
| Jane Smith | Finance |
| Alice Brown | HR |
| Bob Davis | IT |

**Solution:**

SELECT Employees.Name, Departments.DepartmentName

FROM Employees

INNER JOIN Departments ON Employees.DepartmentID = Departments.DepartmentID;

**Explanation:**

The INNER JOIN returns only the rows where there is a match between the Employees and Departments tables based on the DepartmentID.

**Q2) Problem Statement: Write a query to find employees who work in departments located in either New York or London and have a salary greater than 60,000. Display their names, department names, locations, and salaries.**

**Expected Output:**

|  |  |  |  |
| --- | --- | --- | --- |
| **EmployeeName** | **DepartmentName** | **Location** | **Salary** |
| Jane Smith | Finance | London | 32000 |
| Alice Brown | HR | New York | 75000 |

**Solution:**

SELECT Employees.Name AS EmployeeName,

       Departments.DepartmentName,

       Departments.Location,

       Employees.Salary

FROM Employees

INNER JOIN Departments ON Employees.DepartmentID = Departments.DepartmentID

WHERE Departments.Location IN ('New York', 'London')

  AND Employees.Salary > 60000;

**Explanation:**

The query uses an INNER JOIN to filter employees based on department locations and salary. The WHERE clause ensures only those in New York or London with salaries over 60,000 are included.

**LEFT OUTER JOIN (Left Join):**

**Q3 )Problem Statement: Write a query to retrieve all employees and their corresponding department names, even if the employee is not assigned to any department.**

**Expected Output:**

|  |  |
| --- | --- |
| **Name** | **DepartmentName** |
| John Doe | HR |
| Jane Smith | Finance |
| Alice Brown | HR |
| Bob Davis | IT |
| Carol White | NULL |

**Solution:**

SELECT Employees.Name, Departments.DepartmentName

FROM Employees

LEFT OUTER JOIN Departments ON Employees.DepartmentID = Departments.DepartmentID;

**Explanation:**

The LEFT OUTER JOIN returns all records from the Employees table, and the matching records from the Departments table. If no match is found, the result is NULL on the Department's side.

**Q4) Problem Statement: Write a query to retrieve the names of all employees, their department names, and the location of their departments. Also, include employees who are not assigned to any department.**

**Expected Output:**

|  |  |  |
| --- | --- | --- |
| **EmployeeName** | **DepartmentName** | **Location** |
| John Doe | HR | New York |
| Jane Smith | Finance | London |
| Alice Brown | HR | New York |
| Bob Davis | IT | San Francisco |
| Carol White | NULL | NULL |

**Solution:**

SELECT Employees.Name AS EmployeeName,

       Departments.DepartmentName,

       Departments.Location

FROM Employees

LEFT OUTER JOIN Departments ON Employees.DepartmentID = Departments.DepartmentID;

**Explanation:**

This query uses a LEFT OUTER JOIN to ensure all employees are included, even those without a department assignment. It combines employee names with department names and locations.

**RIGHT OUTER JOIN (Right Join):**

**Q5) Problem Statement: Write a query to retrieve all departments and their corresponding employees, even if a department has no employees assigned to it.**

**Expected Output:**

|  |  |
| --- | --- |
| **Name** | **DepartmentName** |
| John Doe | HR |
| Jane Smith | Finance |
| Alice Brown | HR |
| Bob Davis | IT |
| NULL | Marketing |

**Solution:**

SELECT Employees.Name, Departments.DepartmentName

FROM Employees

RIGHT OUTER JOIN Departments ON Employees.DepartmentID = Departments.DepartmentID;

**Explanation:**

The RIGHT OUTER JOIN returns all records from the Departments table, and the matching records from the Employees table. If no match is found, the result is NULL on the Employees side.

**Q6) Problem Statement: Write a query to retrieve all departments along with the names of employees assigned to them. If no employees are assigned, display "No Employees" in the employee name field. Also, include the department's location.**

**Expected Output:**

|  |  |  |
| --- | --- | --- |
| **EmployeeName** | **DepartmentName** | **Location** |
| John Doe | HR | New York |
| Alice Brown | HR | New York |
| Jane Smith | Finance | London |
| Bob Davis | IT | San Francisco |
| No Employees | Marketing | Chicago |

**Solution:**

SELECT COALESCE(Employees.Name, 'No Employees') AS EmployeeName,

       Departments.DepartmentName,

       Departments.Location

FROM Employees

RIGHT OUTER JOIN Departments ON Employees.DepartmentID = Departments.DepartmentID;

**Explanation:**

The RIGHT OUTER JOIN ensures that all departments are included, even those without employees. The COALESCE function is used to replace NULL employee names with "No Employees."

**SELF JOIN:**

**Q7) Problem Statement: Write a query to find pairs of employees who work in the same department.**

**Expected Output:**

|  |  |  |
| --- | --- | --- |
| **Employee1** | **Employee2** | **DepartmentID** |
| John Doe | Alice Brown | 10 |

**Solution:**

SELECT E1.Name AS Employee1, E2.Name AS Employee2, E1.DepartmentID

FROM Employees E1

INNER JOIN Employees E2 ON E1.DepartmentID = E2.DepartmentID AND E1.EmployeeID < E2.EmployeeID;

**Explanation:**

The SELF JOIN is used here to compare rows within the Employees table. We join the table with itself to find pairs of employees (Employee1 and Employee2) who share the same DepartmentID.

The condition E1.EmployeeID < E2.EmployeeID ensures that each pair is listed only once.

**Q9) Problem Statement: Write a query to calculate the average salary of employees in each department and list the department names and locations along with this average salary. Include only those departments that have more than one employee.**

**Expected Output:**

|  |  |  |
| --- | --- | --- |
| **DepartmentName** | **Location** | **AverageSalary** |
| HR | New York | 65000 |

**Solution:**

SELECT Departments.DepartmentName,

       Departments.Location,

       AVG(Employees.Salary) AS AverageSalary

FROM Employees

INNER JOIN Departments ON Employees.DepartmentID = Departments.DepartmentID

GROUP BY Departments.DepartmentName, Departments.Location

HAVING COUNT(Employees.EmployeeID) > 1;

**Explanation:**

The query uses an INNER JOIN to link Employees and Departments, then groups by department name and location to calculate the average salary. The HAVING clause filters for departments with more than one employee.

**Q10) Write an SQL query to retrieve details of the employee with the highest salary in the "HR" department. The query should return:**

**All columns of the selected employee (i.e., employee name, salary, department number).**

**The department location.**

**The salary grade.**

**Solution (Please use appropriate column names):**

SELECT E.\*,D.LOC,S.GRADE

FROM(SELECT ENAME,SAL,DEPTNO

FROM EMP

WHERE SAL=(SELECT MAX(SAL)

  FROM EMP

  WHERE DEPTNO=(SELECT DEPTNO

          FROM DEPT

                WHERE DNAME='HR'))) AS E JOIN DEPT AS D JOIN SALGRADE AS S

ON E.DEPTNO=D.DEPTNO AND E.SAL BETWEEN S.LOSAL AND S.HISAL

**Q11) Write an SQL query to find employees who earn more than the average salary of their respective departments.**

**Requirements:**

**Retrieve employee name, salary, department name, and department average salary.**

**Use a subquery to calculate the average salary for each department and join it with EMP to filter employees whose salary is above their department’s average.**

**Sort the result by department name and employee salary in descending order.**

**Solution(Please use appropriate column names):**

SELECT E.ENAME, E.SAL, D.DNAME, AVG\_DEPT.AVG\_SAL

FROM EMP AS E

JOIN DEPT AS D ON E.DEPTNO = D.DEPTNO

JOIN (

    SELECT DEPTNO, AVG(SAL) AS AVG\_SAL

    FROM EMP

    GROUP BY DEPTNO

) AS AVG\_DEPT ON E.DEPTNO = AVG\_DEPT.DEPTNO

WHERE E.SAL > AVG\_DEPT.AVG\_SAL

ORDER BY D.DNAME, E.SAL DESC;

**Q12) Write an SQL query to find the top 3 employees with the highest salary in each department.**

**Requirements:**

**Retrieve the top 3 highest-paid employees in each department.**

**Include employee name, salary, department name, and department number.**

**Use a correlated subquery to count the number of distinct salaries in the same department that are higher than the current employee's salary..**

**Display only the top 3 employees per department, even if there are more employees with the same salary.**

**Solution(Please use appropriate column names):**

SELECT E.ENAME, E.SAL, D.DNAME, E.DEPTNO

FROM EMP AS E

JOIN DEPT AS D ON E.DEPTNO = D.DEPTNO

WHERE (

    SELECT COUNT(DISTINCT SAL)

    FROM EMP AS E2

    WHERE E2.DEPTNO = E.DEPTNO

    AND E2.SAL > E.SAL

) < 3

ORDER BY E.DEPTNO, E.SAL DESC;

### **Explanation**

**The inner correlated subquery (SELECT COUNT(DISTINCT SAL) FROM EMP AS E2 WHERE E2.DEPTNO = E.DEPTNO AND E2.SAL > E.SAL) counts the number of distinct salaries in the same department that are higher than the current employee's salary.**

**If this count is less than 3, it means the employee is among the top 3 highest-paid in their department.**

**Finally, the ORDER BY E.DEPTNO, E.SAL DESC sorts the results by department and salary in descending order, ensuring the top employees are shown in rank order.**