**National University of Computer & Emerging Sciences, Karachi**



**Computer Science Department**

**Fall 2024, Lab Manual – 05**

|  |  |
| --- | --- |
| **Course Code: CL-2005** | **Course: Database Systems Lab** |
| **Instructor(s):** | **Syeda Mahnoor Javed** |

**Contents:**

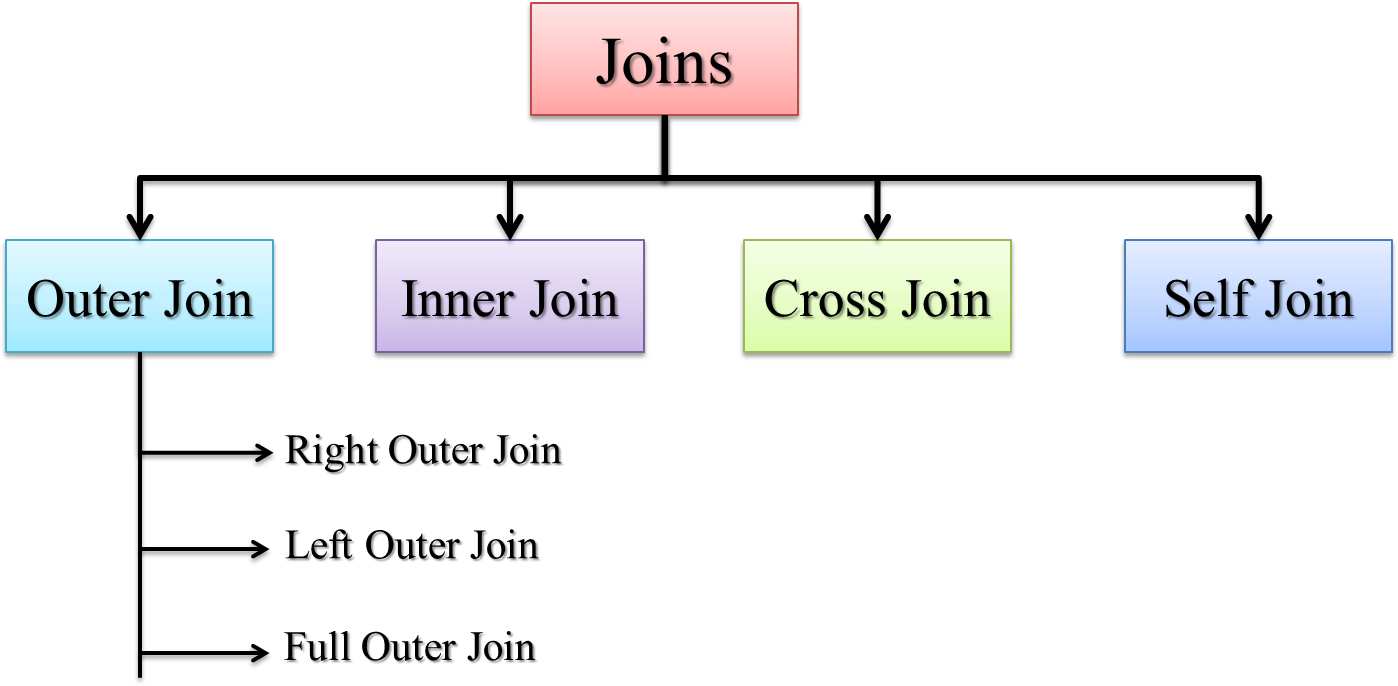
* Introduction to Joins
* Types of Joins
* Introduction to Outer Joins and Its Types
* Introduction to Set Operator
* Types of Set Operator
* Exercise

**INTRODUCTION TO JOIN**

The JOIN keyword is used in an SQL statement to query data from two or more tables based on a relationship between certain columns in these tables.

**TYPES OF JOINS:**

Following are the types of joins. They are:

* Cross Join / Cartesian Join
* Inner Join / Equity Join
* Outer Join
* Left Outer
* Right Outer
* Full Outer
* Self-Join

**Cross Join / Cartesian Join:**

In a Cartesian join, also called a Cartesian product or cross join, each record in the first table is matched with each record in the second table.

**(# rows in Table 1) \* (# rows in Table 2)**

****

**Syntax for Cross Join/Cartesian Join:**

SELECT \* FROM TABLE1, TABLE2;

**ISO Standard:**

SELECT \* FROM TABLE1 CROSS JOIN TABLE2;

**Inner Join / Equality Joins:**

If the join contains an equality condition, it is also called Equi Join, Natural Join, Inner Join.

**Syntax For Inner Join:**

SELECT TABLE1\_COLUMN, TABLE2\_COLUMN FROM TABLE1 T1, TABLE2 T2 WHERE T1. TABLE1\_COLUMN = T2. TABLE2\_COLUMN;

Example

**To retrieve the employee name, their job and department name, we need to extract data from two tables, EMP and DEPT:**

SELECT E.ENAME, E.JOB, D.DNAME FROM EMP E, DEPT D WHERE E.DEPTNO = D.DEPTNO;

The SQL-1999 standard:

SELECT ENAME, JOB, DNAME FROM EMP NATURAL JOIN DEPT;

**Using Clause:**

No matter how many common columns are available in the tables, NATURAL JOIN will join with all the common columns.

Use USING clause to join with specified columns.

**Syntax for Using Clause:**

SELECT TABLE1\_COLUMN, TABLE2\_COLUMN FROM TABLE1 JOIN TABLE2 USING (TABLE2\_COLUMN, TABLE1\_COLUMN)

Example

SELECT EMPNO, ENAME, MGR, DNAME FROM EMP JOIN DEPT USING (DEPTNO, MGR);

**Self-Join:**

When a table is joined to itself then it is called as Self join or in less words we

Can just say “joining a table to itself is called self-join”.

**Syntax for Self-join:**

SELECT T1.TABLE1\_COLUMN,T2. TABLE2\_COLUMN FROM TABLE T1,TABLE T2 WHERE T1.COLUMN = T2.COLUMN;

Example

SELECT WORKER.ENAME, MANAGER.ENAME FROM EMP WORKER, EMP MANAGER WHERE WORKER.MGR = MANAGER.EMPNO;

**INTRODUCTION TO OUTER JOIN & ITS TYPES**

Use Outer join to return records which don’t have direct match.

In outer join operation, all records from the source table included in the result even though they don't satisfy the join condition.

**Syntax for Outer Join:**

SELECT column names from both tables FROM table name 1 LEFT|RIGHT|FULL OUTER JOIN table name 2 on condition;

**Types of Outer Joins:**

Outer joins are classified into three types:

1. Left Outer Join
2. Right Outer Join
3. Full Outer Join

**Left Outer Join:**

The left outer join produces a table that contains the matched data from the two tables, as well as the remaining rows of the left table and null from the columns of the right table.

**Syntax for Left Outer Join:**

SELECT T1. TABLE1\_COLUMN, T2. TABLE2\_COLUMN FROM TABLE1 T1, TABLE2 T2

WHERE T1.TABLE1\_COLUMN = T2. TABLE2\_COLUMN(+);

Example

SELECT E.ENAME, D.DEPTNO, D.DNAME FROM EMP E, DEPT D WHERE E.DEPTNO = D.DEPTNO (+);

**NOTE**: The outer join operator appears on only that side that has information missing.

**The SQL-1999 standard:**

SELECT T1. TABLE1\_COLUMN, T2. TABLE2\_COLUMN FROM TABLE1 T1 LEFT OUTER JOIN TABLE2 T2 ON T1.TABLE1\_COLUMN = T2. TABLE2\_COLUMN;

**Example**

SELECT E.ENAME, D.DEPTNO, D.DNAME FROM EMP E LEFT OUTER JOIN DEPT D ON (E.DEPTNO = D.DEPTNO);

**Right Outer Join:**

The right outer join returns a table with the matched data from the two tables being joined, then the remaining rows of the right table and null for the remaining left table's columns.

**Syntax for Right Outer Join:**

SELECT T1. TABLE1\_COLUMN, T2. TABLE2\_COLUMN FROM TABLE1 T1, TABLE2 T2

WHERE T1.TABLE1\_COLUMN (+)= T2. TABLE2\_COLUMN;

**Example:**

SELECT E.ENAME, D.DEPTNO, D.DNAME FROM EMP E, DEPT D WHERE E.DEPTNO(+) = D.DEPTNO;

**SQL-1999 standard:**

SELECT T1. TABLE1\_COLUMN, T2. TABLE2\_COLUMN FROM TABLE1 T1 RIGHT OUTER JOIN TABLE2 T2 ON T1.TABLE1\_COLUMN = T2. TABLE2\_COLUMN;

**Example:**

SELECT E.ENAME, D.DEPTNO, D.DNAME FROM EMP E RIGHT OUTER JOIN DEPT D ON (E.DEPTNO = D.DEPTNO);

**Full Outer Join:**

The full outer join returns a table with the matched data of two table then remaining rows of both left table and then the right table.

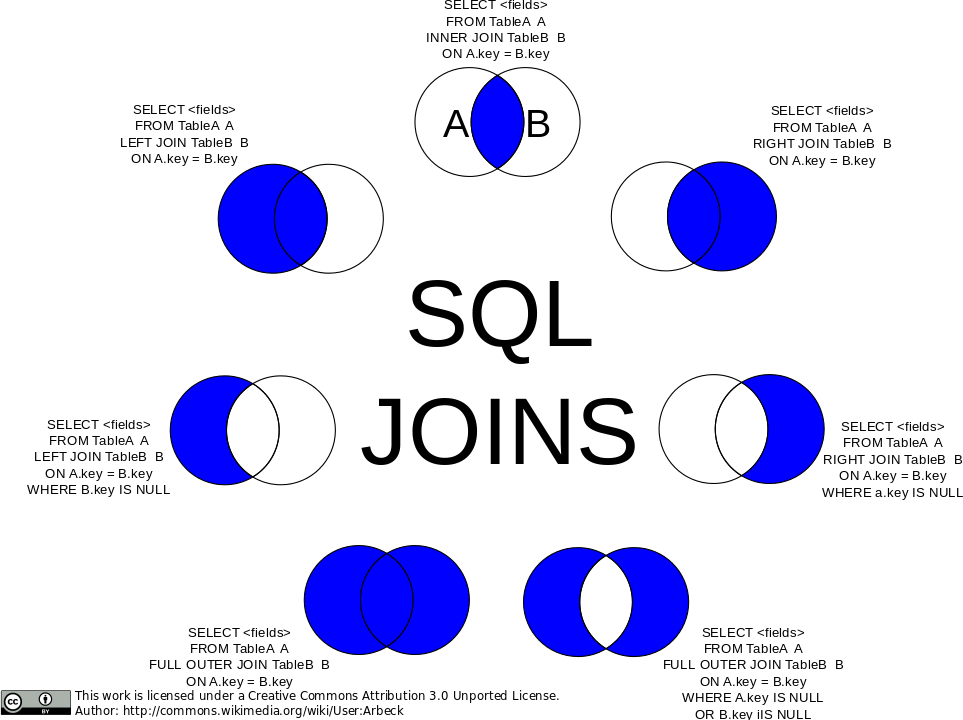
**Syntax for Full Outer Join:**

SELECT T1. TABLE1\_COLUMN, T2. TABLE2\_COLUMN FROM TABLE1 T1 FULL OUTER JOIN TABLE2 T2 ON T1.TABLE1\_COLUMN = T2. TABLE2\_COLUMN;

Example

SELECT E.ENAME, D.DEPTNO, D.DNAME FROM EMP E FULL OUTER JOIN DEPT D ON (E.DEPTNO = D.DEPTNO);

**Conclusion for Joins:**



**INTRODUCTION TO SET OPERATOR**

Set operators are used to join the results of two (or more) SELECT statements. The SET operators available in Oracle 11g are UNION, UNION ALL, INTERSECT and MINUS.

All of the SET operators have the same order of precedence. Instead, Oracle evaluates queries from left to right or top to bottom during execution. If parentheses are used explicitly, the order may change because parentheses take precedence over dangling operators.

**TYPES OF SET OPERATOR**

Following are the types of operators that are used for set in oracle. They are:

1. Union
2. Union all
3. Intersect
4. Minus

**Union Operator:**

The SQL Union function joins the results of two or more SQL SELECT queries. The number of datatypes and columns in both tables on which the UNION operation is performed must be the same in order to perform the union operation. The duplicate rows are removed from the result of the union operation.

**Diagrammatic view of Union operator**

**Syntax for Union Operator**

SELECT TABLE1\_COLUMN, TABLE2\_COLUMN FROM TABLE1 **UNION** SELECT TABLE1\_COLUMN\_ID, TABLE2\_COLUMN\_ID FROM TABLE2

Example

SELECT employee\_id, job\_id FROM employees **UNION** SELECT employee\_id, job\_id FROM job\_history;

**Union All Operator:**

With one exception, UNION and UNION ALL operate in a similar manner. UNION ALL, on the other hand, returns the result set without eliminating duplication or sorting the data.

**Diagrammatic view of Union all operator**

**Syntax for Union All Operator**

SELECT TABLE1\_COLUMN, TABLE2\_COLUMN FROM TABLE1 UNION ALL SELECT TABLE1\_COLUMN\_ID, TABLE2\_COLUMN\_ID FROM TABLE2

Example

SELECT employee\_id, job\_id, department\_id FROM employees **UNION ALL**

SELECT employee\_id, job\_id, department\_id FROM job\_history

**Intersect Operator**

It's used to join two SELECT statements together. The common rows from both SELECT statements are returned by the Intersect procedure. The number of datatypes and columns in the Intersect operation must be the same. There are no duplicates, and the data is arranged in ascending order by default.

**Diagrammatic view of Intersect operator**

Syntax for Intersect Operator

SELECT TABLE1\_COLUMN, TABLE2\_COLUMN FROM TABLE1 **INTERSECT** SELECT TABLE1\_COLUMN\_ID, TABLE2\_COLUMN\_ID FROM TABLE2

Example

SELECT employee\_id, job\_id FROM employees **INTERSECT** SELECT employee\_id, job\_id FROM job\_history;

**Minus Operator**

It combines the results of two SELECT statements into a single statement. The minus operator is used to show rows that are present in the first query but not in the second. There are no duplicates, and the data is sorted ascending by default.

**Diagrammatic view of Minus operator**

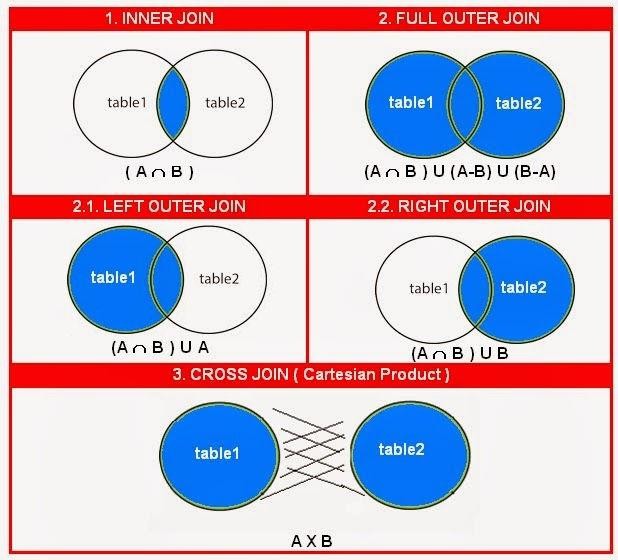
**Syntax for Minus Operator**

SELECT TABLE1\_COLUMN, TABLE2\_COLUMN FROM TABLE1 **MINUS** SELECT TABLE1\_COLUMN\_ID, TABLE2\_COLUMN\_ID FROM TABLE2

Example

SELECT employee\_id, job\_id FROM employees **MINUS** SELECT employee\_id, job\_id FROM job\_history

**How to implement joins as set operator?**



**LAB EXERCISE**

1. Write a query to list the name, job title, department name, and salary of the employees in ascending order of their department.
2. Write a query to list the departments where at least two employees are working.
3. Fetch all records where the employee's salary is less than the lowest salary in the company.
4. Write a query to list the name, job title, annual salary, department name, and city of employees who earn 60000 or more annually and are not working as ANALYST.
5. Write a query to print details of the employees who are also Managers.
6. List department number and department name for all departments that have no employees.
7. Display employee name, salary, and department name where all employees match their department, including employees with no assigned department.
8. Display the name, job title, department name, and city of employees who are working in departments located in cities without a state province.
9. Write an SQL query to show records from one table that do not exist in another table.
10. Display all employees who belong to the US but not to the state of Washington.
11. Write a query to list the name, job title, department name, and location of employees who have a salary higher than the average salary in their department.
12. Write a query to list employees who have changed their job title at least once in their job history.
13. List employees who work in the same department as their managers.
14. Write a query to list the name, department name, and location of employees who work in the same country as their department location.
15. Write a query to find employees who work in departments with more than 5 employees.
16. Display a list of employees along with their managers' names.
17. Write a query to list the employee names and their department names where the department is located in a different country than the employee’s residence.
18. Write a query to find employees who earn more than their department's average salary but less than the highest salary in the company.
19. Display a list of all employees who have worked in multiple departments, showing their job history and department names.
20. Write a query to find employees who have worked in more than one region throughout their career.
21. List all employees and the region they are working in.
22. Find employees who have the same last name but work in different departments.
23. List employees who have changed job titles more than twice.
24. Show job titles that are not currently assigned to any employee.
25. Find the top 3 employees with the highest salaries in each department.