**SQL Joins**

# Agenda:

* Introduction to Joins in SQL
* Types of Joins in SQL
  + Inner Join
  + Outer Join

1. Left Join
2. Right Join

* Full Join
* Cross Join
* Self Join
* Delete Join
* Update Join

# Introduction

SQL Joins are used to combine rows from two or more tables in a relational database, based on a related column between them. Joins are essential because data in relational databases is often spread across multiple normalized tables. By using Joins, you can retrieve meaningful combined information.

# Join Predicate

The JOIN predicate specifies the condition under which rows from different tables should be combined. This is usually defined using the ON clause. Common operators include:

* =, <, >, <=, >=, <>, !=
* BETWEEN, LIKE, IN, NOT

# Key Features of Joins

1. **Flexibility:** Combine tables in multiple ways depending on requirements.
2. **Multiple Tables:** You can join more than two tables at once.
3. **Comparison Operators:** Not limited to equality, other operators can be used.
4. **No Storage:** Joins only retrieve combined results without storing them permanently.

# General Syntax of a Join

**Basic syntax** for a JOIN in SQL:

|  |
| --- |
| SELECT column\_name(s) FROM table1 JOIN table2 ON table1.column\_name = table2.column\_name; |

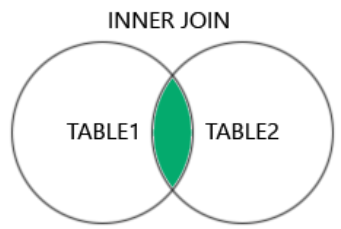
**Types of Joins in SQL**

There are several types of joins in SQL. They can be broadly classified as:

1. INNER JOIN
2. OUTER JOIN (LEFT, RIGHT & FULL JOIN)
3. CROSS JOIN
4. SELF JOIN
5. DELETE JOIN
6. UPDATE... JOIN

## I. Inner Join

INNER JOIN returns only those records that have matching values in both tables. If no match is found, the row is not included.



**Syntax:**

|  |
| --- |
| SELECT column\_name(s) FROM table1 INNER JOIN table2 ON table1.column\_name = table2.column\_name; |

**Example: Retrieve information about employees along with the name of their departments?**

|  |
| --- |
| SELECT Employees.EmployeeID, Employees.EmployeeName, Departments.DepartmentName FROM Employees INNER JOIN Departments ON Employees.DepartmentID = Departments.DepartmentID; |

**Joining Multiple Tables [Using Inner Join]**

**Example: Retrieving Employee details, their Department names, and their Salaries?**

|  |
| --- |
| SELECT E.EmployeeID, E.EmployeeName, D.DepartmentName, S.Salary FROM Employees E INNER JOIN Departments D ON E.DepartmentID = D.DepartmentID INNER JOIN Salaries S ON E.EmployeeID = S.EmployeeID; |

**Inner Join with WHERE Clause**

**Example: Retrieve employee details and their department names, where employees with a salary greater than 55000?**

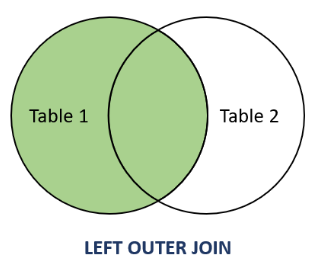
|  |
| --- |
| SELECT E.EmployeeID, E.EmployeeName, D.DepartmentName, S.Salary FROM Employees E INNER JOIN Departments D ON Employees.DepartmentID = Departments.DepartmentID INNER JOIN Salaries S ON E.EmployeeID = S.EmployeeID WHERE S.Salary > 55000; |

## II. Outer Joins

Outer Joins return all rows from one or both tables, even if there are no matches. Unmatched rows will contain NULL values.

### 2.1 Left Join

LEFT JOIN returns all rows from the left table, and matching rows from the right table.



**Example: Retrieve Employees with or without any assigned department?**

|  |
| --- |
| SELECT E.EmployeeID, E.EmployeeName, D.DepartmentName FROM Employees E LEFT JOIN Departments D ON E.DepartmentID = D.DepartmentID; |

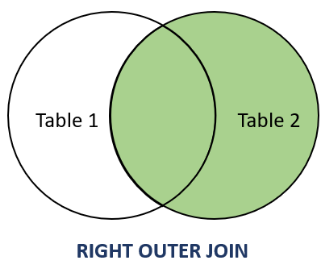
**LEFT JOIN with WHERE Clause**

**Example: Retrieve only those Employees and their Department where employees have a salary greater than 55000?**

|  |
| --- |
| SELECT E.EmployeeID, E.EmployeeName, D.DepartmentName, S.Salary FROM Employees E LEFT JOIN Departments D ON E.DepartmentID = D.DepartmentID LEFT JOIN Salaries S ON E.EmployeeID = S.EmployeeID WHERE S.Salary >= 55000; |

### 2.2 Right Join

RIGHT JOIN returns all rows from the right table, and matching rows from the left table.



**Example: Retrieve all department names with or without any employees?**

|  |
| --- |
| SELECT E.EmployeeID, E.EmployeeName, D.DepartmentName FROM Employees E RIGHT JOIN Departments D ON E.DepartmentID = D.DepartmentID; |

**RIGHT JOIN with WHERE Clause**

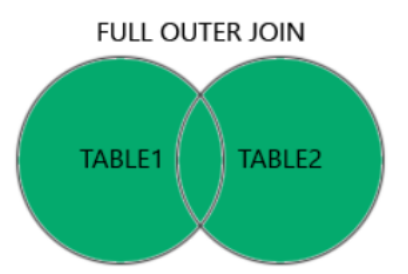
**Example: Retrieve all Departments, Employees & their Salary, ensuring that all rows from salary are included even if no matching records?**

|  |
| --- |
| SELECT E.EmployeeID, E.EmployeeName, D.DepartmentName, S.Salary FROM Employees E RIGHT JOIN Departments D ON E.DepartmentID = D.DepartmentID RIGHT JOIN Salaries S ON E.EmployeeID = S.EmployeeID WHERE S.Salary > 55000; |

### 2.3 Full Join

FULL JOIN returns all rows from both tables. If there is no match, NULL values are shown.

**Note:** MySQL does not directly support FULL JOIN, but it can be emulated using UNION of LEFT and RIGHT JOIN.



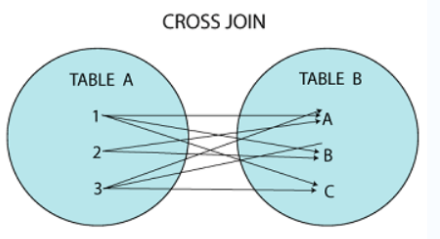
**Example: Retrieve all employees & departments, even if none in any department or employees?**

|  |
| --- |
| SELECT E.EmployeeID, E.EmployeeName, D.DepartmentName FROM Employees E LEFT JOIN Departments D ON E.DepartmentID = D.DepartmentID  UNION  SELECT E.EmployeeID, E.EmployeeName, D.DepartmentName FROM Employees E RIGHT JOIN Departments D ON E.DepartmentID = D.DepartmentID; |

## 3. Cross Join

CROSS JOIN produces a Cartesian product of two tables, i.e., every row from the first table is combined with every row from the second.

Note: No Joining condition is required to perform a Cartesian product.

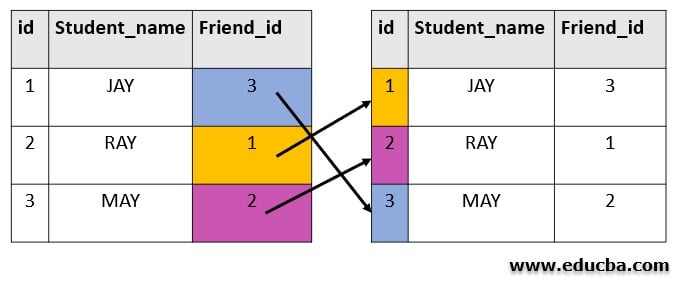


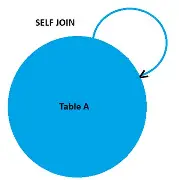
**Example: Retrieve all possible combination between a employee & a department?**

|  |
| --- |
| SELECT Employees.EmployeeID, Employees.EmployeeName, Departments.DepartmentName FROM Employees CROSS JOIN Departments; |

## 4. Self Join

SELF JOIN is a join where a table is joined with itself to compare rows within the same table.





**Example: Perform a Self-Join to find employees that belong to the same department?**

|  |
| --- |
| SELECT a.EmployeeID, a.EmployeeName, b.EmployeeID, b.EmployeeName, a.DepartmentID FROM Employees a, Employees b WHERE a.DepartmentID = b.DepartmentID  AND a.EmployeeID < > b.EmployeeID; |

Note: Self Join uses an alias for the table (e.g., a and b) to differentiate between the two instances of same table.

## 5. DELETE JOIN

DELETE JOIN allows deleting rows from one table based on conditions in another joined table.

**Example: Delete customers with a salary less than Rs. 2000.00?**

|  |
| --- |
| DELETE e FROM Employees AS e JOIN Salaries AS s ON e.EmployeeID = s.EmployeeID WHERE s.Salary < 2000.00; |

## 6. UPDATE... JOIN

UPDATE... JOIN is used to update rows in one table based on related data in another table.

**Example: Update the salary of Employees based on their department?**

|  |
| --- |
| UPDATE Salaries AS s JOIN Employees AS e ON s.EmployeeID = e.EmployeeID SET s.Salary = s.Salary \* 1.10 WHERE e.DepartmentID = 2; |