What Are SQL Joins?

SQL Joins combine rows from two or more tables based on a related column between them. Joins are essential for fetching data spread across multiple tables in relational databases.

Importance of SQL Joins

1. **Data Integration:** Joins help gather related data stored in different tables.
2. **Efficiency:** Allow queries to process specific rows by matching keys, reducing unnecessary data retrieval.
3. **Flexibility:** Enable complex data relationships to be queried effectively.
4. **Real-world Utility:** Commonly used in business analytics, reporting, and applications that work with normalized databases.

Common SQL Joins and Examples

Let's consider the following tables:

Table: `Customers`

CustomerID	Name	Country
1	Alice	USA
2	Bob	Canada
3	Charlie	UK

Table: `Orders`

OrderID	CustomerID	OrderAmount
101	1	250
102	1	150
103	2	300

---

1. INNER JOIN

Definition: Returns rows with matching values in both tables.

Query:

```sql
SELECT Customers.CustomerID, Customers.Name, Orders.OrderID, Orders.OrderAmount FROM Customers INNER JOIN Orders ON Customers.CustomerID = Orders.CustomerID;
```

Result:

CustomerID	Name	OrderID	OrderAmount
1	Alice	101	250
1	Alice	102	150
2	Bob	103	300

Row Count Before Joining:

- Customers: 3 rows
- Orders: 3 rows

Row Count After Joining: 3 rows (only matching rows are included).

---

2. **LEFT JOIN (or LEFT OUTER JOIN)**

Definition: Returns all rows from the left table and the matching rows from the right table. If no match, NULL values are shown for the right table.

Query:

```sql
SELECT Customers.CustomerID, Customers.Name, Orders.OrderID, Orders.OrderAmount FROM Customers LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID;
```

Result:

CustomerID	Name	OrderID	OrderAmount
1	Alice	101	250
1	Alice	102	150
2	Bob	103	300
3	Charlie	NULL	NULL

Row Count After Joining: 4 rows (includes non-matching rows from `Customers`).

3. **RIGHT JOIN (or RIGHT OUTER JOIN)**

Definition: Returns all rows from the right table and the matching rows from the left table. If no match, NULL values are shown for the left table.

Query:

```sql
SELECT Customers.CustomerID, Customers.Name, Orders.OrderID, Orders.OrderAmount FROM Customers RIGHT JOIN Orders ON Customers.CustomerID = Orders.CustomerID;
```

Result:

CustomerID	Name	OrderID	OrderAmount
1	Alice	101	250
1	Alice	102	150
2	Bob	103	300

Row Count After Joining: 3 rows (includes all rows from `Orders`).

---

4. **FULL JOIN (or FULL OUTER JOIN)**

Definition: Returns all rows from both tables. Non-matching rows are filled with NULLs.

Query:

```sql
SELECT Customers.CustomerID, Customers.Name, Orders.OrderID, Orders.OrderAmount FROM Customers FULL JOIN Orders ON Customers.CustomerID = Orders.CustomerID;
```

Result:

CustomerID	Name	OrderID	OrderAmount
1	Alice	101	250
1	Alice	102	150
2	Bob	103	300
3	Charlie	NULL	NULL

Row Count After Joining: 4 rows (includes all non-matching rows from both tables).

5. **CROSS JOIN**

Definition: Returns the Cartesian product of the two tables (all combinations of rows).

Query:

```sql
SELECT Customers.Name, Orders.OrderID FROM Customers CROSS JOIN Orders;
```

Result:

Name	OrderID
Alice	101
Alice	102
Alice	103
Bob	101
Bob	102
Bob	103
Charlie	101
Charlie	102
Charlie	103

Row Count After Joining: 9 rows (3 customers × 3 orders).

How to Merge More Than Two Tables

You can join more than two tables by chaining JOIN operations.

Example:

Tables:

- `Products`：

ProductID	ProductName	Price
201	Laptop	1200
202	Mouse	20

Query:

```sql
SELECT Customers.Name, Orders.OrderID, Products.ProductName, Products.Price FROM
Customers INNER JOIN Orders ON Customers.CustomerID = Orders.CustomerID INNER JOIN
Products ON Orders.OrderID = Products.ProductID;
```

Steps:

1. Join `Customers` and `Orders` first.
2. Add `Products` to the result using another join.

This approach can be extended to any number of tables, provided there are relationships between them.

Summary Table for Joins

Join Type	Includes Non-Matching Rows	Combines All Rows
INNER JOIN	No	No
LEFT JOIN	Yes (Left Table)	No
RIGHT JOIN	Yes (Right Table)	No
FULL JOIN	Yes (Both Tables)	No
CROSS JOIN	No	Yes

Why Learn Joins?

SQL Joins are indispensable in:

- Building relational data queries.
- Creating business reports.
- Analyzing data trends across multiple datasets.
- Powering data-driven applications efficiently.