11 SQL Joins - NikhilSharma X KirkYagami

Introduction to Joins

In SQL, a **join** is a method of combining rows from two or more tables based on a related column between them. Joins are essential for querying data that resides in multiple tables and retrieving meaningful information by leveraging relationships between tables.

Types of Joins

1. **Inner Join**
2. **Left (Outer) Join**
3. **Right (Outer) Join**
4. **Full (Outer) Join**
5. **Cross Join**
6. **Self Join**

1. Inner Join

Definition:
An **inner join** returns only the rows that have matching values in both tables.

When to Use:
Use an inner join when you want to retrieve only the records that have corresponding matches in both tables.

Example:

```
SELECT
    p.Programmer_Name,
    p.Institute,
    s.Software_Name,
    s.Developed_In
FROM
    studies p
INNER JOIN
    software s
ON
    p.Programmer_Name = s.Programmer_Name;
```

2. Left (Outer) Join

Definition:
A **left join** returns all the rows from the left table and the matched rows from the right table. If no match is found, NULL values are returned for columns from the right table.

When to Use:
Use a left join when you want to retrieve all records from the left table, along with the matching records from the right table. If there are no matches, the result will include NULL for columns from the right table.

Example:

```
SELECT
    p.Programmer_Name,
    p.Institute,
    s.Software_Name,
    s.Developed_In
FROM
    studies p
LEFT JOIN
    software s
ON
    p.Programmer_Name = s.Programmer_Name;
```

3. Right (Outer) Join

Definition:
A **right join** returns all the rows from the right table and the matched rows from the left table. If no match is found, NULL values are returned for columns from the left table.

When to Use:
Use a right join when you want to retrieve all records from the right table, along with the matching records from the left table. If there are no matches, the result will include NULL for columns from the left table.

Example:

```
SELECT
    s.Programmer_Name,
    s.Software_Name,
    p.Institute,
    p.Course
FROM
    software s
RIGHT JOIN
    studies p
ON
    s.Programmer_Name = p.Programmer_Name;
```

4. Full (Outer) Join

Definition:
A **full join** returns all the rows when there is a match in either the left table or the right table. If there are rows in one table that do not have matches in the other table, the result will contain NULL values for columns from the other table.

When to Use:
Use a full join when you want to retrieve all records from both tables, and include rows with no matches as well.

Example:

```
SELECT
    p.Programmer_Name,
    p.Institute,
    s.Software_Name,
    s.Developed_In
```

```
FROM
    studies p
FULL OUTER JOIN
    software s
ON
    p.Programmer_Name = s.Programmer_Name;
```

5. Cross Join

Definition:
A **cross join** returns the Cartesian product of the two tables, i.e., it returns all possible combinations of rows from the tables.

When to Use:
Use a cross join when you need to combine all rows from the first table with all rows from the second table.

Example:

```
SELECT
    p.Programmer_Name,
    s.Software_Name
FROM
    studies p
CROSS JOIN
    software s;
```

```
-- Better
   SELECT
    p.Programmer_Name,
    s.Software_Name
FROM
    studies p
CROSS JOIN
    softwareApp s
WHERE
    p.Programmer_Name = s.Programmer_Name;
```

6. Self Join

Definition:
A **self join** is a regular join, but the table is joined with itself.

When to Use:
Use a self join when you need to compare rows within the same table.

Example:

```
SELECT
    a.Programmer_Name AS Programmer1,
    b.Programmer_Name AS Programmer2,
    a.Institute AS Institute1,
    b.Institute AS Institute2
FROM
    studies a
```

```
JOIN
    studies b
ON
    a.Institute = b.Institute
AND
    a.Programmer_Name <> b.Programmer_Name;
```

Practical Examples

Inner Join Example

```
SELECT
    p.Programmer_Name,
    p.Institute,
    s.Software_Name,
    s.Developed_In
FROM
    studies p
INNER JOIN
    software s
ON
    p.Programmer_Name = s.Programmer_Name;
```

Left Join Example

```
SELECT
    p.Programmer_Name,
    p.Institute,
    s.Software_Name,
    s.Developed_In
FROM
    studies p
LEFT JOIN
    software s
ON
    p.Programmer_Name = s.Programmer_Name;
```

Right Join Example

```
SELECT
    s.Programmer_Name,
    s.Software_Name,
    p.Institute,
    p.Course
FROM
    software s
RIGHT JOIN
    studies p
ON
    s.Programmer_Name = p.Programmer_Name;
```

Full Outer Join Example

```sql
SELECT
    p.Programmer_Name,
    p.Institute,
    s.Software_Name,
    s.Developed_In
FROM
    studies p
FULL OUTER JOIN
    software s
ON
    p.Programmer_Name = s.Programmer_Name;
```

Cross Join Example

```sql
SELECT
    p.Programmer_Name,
    s.Software_Name
FROM
    studies p
CROSS JOIN
    software s;
```

Self Join Example

```sql
SELECT
    a.Programmer_Name AS Programmer1,
    b.Programmer_Name AS Programmer2,
    a.Institute AS Institute1,
    b.Institute AS Institute2
FROM
    studies a
JOIN
    studies b
ON
    a.Institute = b.Institute
AND
    a.Programmer_Name <> b.Programmer_Name;
```

Summary

Understanding and utilizing joins in SQL allows you to retrieve and combine data from multiple tables efficiently. Each type of join serves a specific purpose and should be chosen based on the desired result and the relationship between the tables.