🧩 What is a SQL JOIN?

A SQL JOIN is used to combine rows from two or more tables based on a related column between them.

It allows you to query data from multiple tables as if the data were part of a single table.

JOINs are essential for relational databases, where data is often split into separate, normalized tables to reduce redundancy and improve structure.

🔗 Types of JOINs in SQL

1. INNER JOIN

Definition:

An INNER JOIN returns only the rows that have matching values in both tables.

If a row in one table doesn't have a corresponding match in the other table, it is not included in the results.

This is the most common type of join used in SQL.

syntax:

```
SELECT columns

FROM table1

INNER JOIN table2

ON table1.common_column = table2.common_column;
```

Example:

```
SELECT employees.name, departments.department_name

FROM employees

INNER JOIN departments

ON employees.department_id = departments.id;
```

2. LEFT JOIN (LEFT OUTER JOIN)

Definition:

A LEFT JOIN returns all rows from the left table and the matched rows from the right table.

If there is no match, the result will contain NULL for columns from the right table.

It's useful for finding unmatched records in the right table.

Syntax:

```
SELECT columns

FROM table1

LEFT JOIN table2

ON table1.common_column = table2.common_column;
```

Example:

```
SELECT customers.name, orders.order_date

FROM customers

LEFT JOIN orders

ON customers.id = orders.customer_id;
```

3. RIGHT JOIN (RIGHT OUTER JOIN)

Definition:

A RIGHT JOIN returns all rows from the right table and the matched rows from the left table.

If there is no match, NULLs appear for columns from the left table.

This is the mirror image of a LEFT JOIN.

Syntax:

```
SELECT columns

FROM table1

RIGHT JOIN table2

ON table1.common_column = table2.common_column;
```

Example:

```
SELECT orders.order_date, customers.name

FROM orders

RIGHT JOIN customers

ON orders.customer_id = customers.id;
```

4. FULL JOIN (FULL OUTER JOIN)

Definition:

A FULL JOIN returns all rows from both tables, with NULLs in places where there is no match.

If a row exists in one table but not the other, the result still includes it with NULL for the missing side.

This join is useful when you need a complete set of data from both tables.

Syntax:

```
SELECT columns
FROM table1
FULL OUTER JOIN table2
ON table1.common_column = table2.common_column;
```

Example:

```
SELECT a.name, b.order_date
FROM customers a
FULL OUTER JOIN orders b
ON a.id = b.customer_id;
```

5. CROSS JOIN

Definition:

A CROSS JOIN returns the Cartesian product of two tables, combining each row from the first table with all rows from the second.

This can create very large result sets and is typically used in special cases such as generating combinations.

Syntax:

```
SELECT *
FROM table1
CROSS JOIN table2;
```

Example:

```
SELECT products.name, colors.color
FROM products
```

```
CROSS JOIN colors;
```