what is operators in sql

An operator is a symbol or keyword used to perform a specific operation on one or more values (called operands). The result of that operation is usually a new value — often used in filtering, calculating, comparing, or combining data.

In SQL, operators are used in queries to:

Perform math (like +, -, *, /)

Compare values (like =, >, <, !=)

Combine conditions (like AND, OR, NOT)

Work with sets (like UNION, INTERSECT)

Do bit-level logic (&, |, etc.)

✅ 1. Arithmetic Operators

Arithmetic operators are used to perform mathematical operations on numeric data in SQL queries. They're commonly used in SELECT, WHERE, UPDATE, and other clauses for calculations like totals, discounts, averages, etc.

➤ Syntax

SELECT column1 + column2 AS total FROM table;

➤ Examples

SELECT salary + bonus AS total_income FROM employees;

SELECT price - discount AS final_price FROM products;

SELECT quantity * unit_price AS total_cost FROM order_items;

SELECT total_amount / items_count AS average_price FROM invoices;

SELECT product_id % 2 AS is_even FROM products;

➤ Usage

Used to compute values like income, profit, discounts, average prices, or check number patterns (like even/odd using %). Often used in dashboards and reports.

⚖️ 2. Comparison Operators

These operators are used to compare two values and return a boolean result (TRUE/FALSE). They're fundamental in filtering records in WHERE, JOIN, CASE, and HAVING.

➤ Syntax

SELECT * FROM table WHERE column1 < column2;

➤ Examples

SELECT * FROM employees WHERE salary = 50000;

SELECT * FROM products WHERE stock <> 0;

SELECT * FROM users WHERE age >= 18;

➤ Usage

Used to filter records based on numeric, string, or date comparisons. Supports filtering exact matches, ranges, and conditional logic.

➤ Operators

= : Equal

!= or <> : Not Equal

> : Greater than

< : Less than

>= : Greater than or equal

<= : Less than or equal

🧠 3. Logical Operators

Logical operators are used to combine multiple conditions. They return TRUE or FALSE depending on the logic evaluation. Most common in WHERE, JOIN ON, HAVING, and CASE.

➤ Syntax

SELECT * FROM table WHERE condition1 AND condition2;

➤ Examples

SELECT * FROM orders WHERE status = 'shipped' AND total > 500;

SELECT * FROM employees WHERE department = 'IT' OR department = 'HR';

SELECT * FROM users WHERE NOT (email_verified = true);

➤ Usage

Used when you need to filter with multiple conditions. AND requires all conditions to be true, OR requires at least one, and NOT reverses a condition.

➤ Operators

AND

OR

NOT

🔗 4. Set Operators

Set operators allow you to combine results from two or more SELECT statements. These are mostly used for reporting or merging similar data from different tables or sources.

➤ Syntax

SELECT column FROM table1

UNION

SELECT column FROM table2;

➤ Examples

SELECT city FROM customers

UNION

SELECT city FROM suppliers;

SELECT product_id FROM warehouse_a

UNION ALL

SELECT product_id FROM warehouse_b;

SELECT user_id FROM app_a

INTERSECT

SELECT user_id FROM app_b;

SELECT email FROM users_2023

EXCEPT

SELECT email FROM unsubscribed;

➤ Usage

UNION removes duplicates

UNION ALL keeps duplicates (faster)

INTERSECT returns common rows

EXCEPT returns rows in the first query but not the second

🧪 5. Bitwise Operators (Advanced)

Bitwise operators are used to work at the binary level — useful for permission systems, feature flags, or compact flag storage.

➤ Examples

```
SELECT * FROM users WHERE permission_flags & 4 != 0;

UPDATE users SET flags = flags | 2 WHERE user_id = 101;

SELECT ~5;  -- Bitwise NOT
```

➤ Usage

Great for managing multiple binary attributes in a single integer field (e.g., is_admin, can_edit, etc.). Requires understanding of binary logic.