# SQL
# OPERATORS
## ( Crash course )


By @engineer_bhaiya_yt

**Shivan kumar**
*Data Science & Analytics*

# SQL OPERATOR

SQL (Structured Query Language) operators are essential tools that perform operations on data in a database. They allow you to filter, compare, and manipulate data in SQL queries. SQL operators are broadly categorized into several types, including arithmetic, comparison, logical, and set operators.

## SQL operators categorized by type

1. Arithmetic Operators
2. Comparison Operators
3. Logical Operators
4. Bitwise Operators
5. Set Operators
6. String Operators
7. Assignment Operator

**Shivan kumar**
*Data Science & Analytics*

# 1. Arithmetic Operators

Arithmetic operators perform mathematical operations on numeric data.

| Operator | Description | Example |
|---|---|---|
| + | Addition | SELECT 10 + 5; -- Result: 15 |
| - | Subtraction | SELECT 10 - 5; -- Result: 5 |
| * | Multiplication | SELECT 10 * 5; -- Result: 50 |
| / | Division | SELECT 10 / 5; -- Result: 2 |
| % | Modulus | SELECT 10 % 3; -- Result: 1 |

Example:

```
SELECT salary + 500 AS increased_salary FROM employees;
```

**Shivan kumar**
*Data Science & Analytics*

## 2. Comparison Operators

Comparison operators are used to compare two values and return true or false.

| Operator | Description | Example |
|---|---|---|
| = | Equal to | SELECT * FROM users WHERE age = 25; |
| <> or != | Not equal to | SELECT * FROM users WHERE age <> 30; |
| > | Greater than | SELECT * FROM users WHERE salary > 5000; |
| < | Less than | SELECT * FROM users WHERE salary < 3000; |
| >= | Greater than or equal to | SELECT * FROM users WHERE age >= 18; |

Example:

```
SELECT * FROM employees WHERE salary > 5000;
```

**Shivan kumar**
*Data Science & Analytics*

# 3. Logical Operators

**Logical operators are used to combine multiple conditions.**

| Operator | Description | Example |
|---|---|---|
| AND | Returns true if all conditions are true | SELECT * FROM users WHERE age > 18 AND city = 'New York'; |
| OR | Returns true if any condition is true | SELECT * FROM users WHERE age > 18 OR city = 'New York'; |
| NOT | Reverses the condition | SELECT * FROM users WHERE NOT city = 'New York'; |

**Example:**

```
SELECT * FROM employees WHERE age > 30 AND
department = 'HR';
```

**Shivan kumar**
*Data Science & Analytics*

# 4. Bitwise Operators

**Bitwise operators perform bit-level operations on integer data.**

| Operator | Description | Example |
|---|---|---|
| & | Bitwise AND | SELECT 5 & 3; -- Result: 1 |
| ^ | Bitwise XOR | SELECT 5 ^ 3; -- Result: 6 |
| ~ | Bitwise NOT | SELECT ~5; -- Result: -6 |

**Example:**

```
SELECT salary & 5 AS bitwise_and FROM employees;
```

**Shivan kumar**
*Data Science & Analytics*

# 5. Set Operators

**Set operators combine the results of two or more SELECT statements.**

| Operator | Description | Example |
|---|---|---|
| UNION | Combines results (removes duplicates) | SELECT name FROM students UNION SELECT name FROM teachers; |
| UNION ALL | Combines results (keeps duplicates) | SELECT name FROM students UNION ALL SELECT name FROM teachers; |
| INTERSECT | Returns common results | SELECT name FROM students INTERSECT SELECT name FROM teachers; |
| EXCEPT | Returns results in one set but not the other | SELECT name FROM students EXCEPT SELECT name FROM teachers; |

**Example:**

```
SELECT employee_id FROM employees WHERE salary > 5000
UNION SELECT employee_id FROM employees WHERE
department = 'Sales';
```

# 6. Special Operators

**Special operators perform operations on specific data types or patterns.**

| Operator | Description | Example |
|---|---|---|
| BETWEEN | Within a range | SELECT * FROM users WHERE age BETWEEN 18 AND 30; |
| IN | Matches any value in a list | SELECT * FROM users WHERE city IN ('London', 'Paris'); |
| LIKE | Pattern matching | SELECT * FROM users WHERE name LIKE 'J%'; |
| IS NULL | Checks for null values | SELECT * FROM users WHERE address IS NULL; |

**Example:**

```
SELECT * FROM orders WHERE status IN ('Pending', 'Processing');
```