**Comparison Operators:**

| Operator | Description | Example |
| --- | --- | --- |
| = | Equal to | SELECT salary FROM employees WHERE salary = 5000; |
| > | Greater than | SELECT salary FROM employees WHERE salary > 5000; |
| < | Less than | SELECT salary FROM employees WHERE salary < 5000; |
| != | Not Equal to | SELECT salary FROM employees WHERE salary != 5000; |

**Logical Operators**:

1. **AND:** Returns TRUE when all conditions are true.
2. **OR:** Returns TRUE when any condition is true.
3. **NOT:** Reverses the result of a comparison.

**ARithmetic Operators:**

| Operator | Description | Example |
| --- | --- | --- |
| + | Addition | SELECT salary + bonus FROM employees; |
| - | Subtraction | SELECT price - discount FROM products; |
| \* | Multiplication | SELECT quantity \* price FROM orders; |
| / | Division | SELECT total / item\_count FROM purchases; |
| % | Modulo (remainder) | SELECT employee\_id % 10 FROM employees; |

**Implicit Data Type Conversion:**

* Happens **automatically by the SQL engine**
* Converts data from one type to another **without you asking**
* Useful but **can slow down queries or cause errors** if types mismatch

Example:

SELECT \* FROM products WHERE Price = ‘100’;

* ‘100’ is a sting literal.
* Price is likely of type DECIMAL or NUMERIC.
* SQL **implicitly converts** the string ‘100’ to a numeric value 100 to perform the comparison.
* So this query will return all products where Price = 100, even though the condition was written using a string.

**Explicit Data Type Conversion:**

Example:

SELECT CAST(‘100’ AS INT); – Converts string to integer

* Output : 100 (as an integer)
* SQL converts the string ‘100’ to an integer value 100.

SELECT COVERT(DECIMAL(5,2), 100) – Converts integer to decimal

* Output: 100.00
* SQL converts the integer 100 into a decimal number with 2 decimal places (DECIMAL(5,2)).
* Useful when formatting currency or prices.

**ORDER BY**:

* The **ORDER BY** clause is used **to sort** the result by one or more columns.
* You can sort in **ascending order (ASC)** or **descending order (DESC)**.
* It is usually placed after GROUP BY and after any aggregate functions.

Q. Find the total sales per city and sort it alphabetically.

SELECT city, SUM(sale\_amount) AS Total\_Sales

FROM sales

GROUP BY city

ORDER BY city;

Q. Find out the average marks per subject and sort by average\_marks in descending order.

SELECT subject, AVG(mark) AS avergae\_marks

FROM marks

GROUP BY subject

ORDER BY average\_marks DESC;

**Filtering Data in SQL:**

**BETWEEN** and **IN** operator: The BETWEEN and IN operators are widely used for filtering data based on specific criteria.

The BETWEEN operator helps filter results within a specified range of values such as numbers, text, or dates.

SELECT column\_name(s)

FROM table\_name

WHERE column\_name **BETWEEN** value1 **AND** value2;

Example:

SELECT name

FROM employee

WHERE salary BETWEEN 50000 AND 70000;

The IN operator filters results based on a specific list of values.

SELECT column\_name(s)

FROM table\_name

WHERE column\_name **IN** (value1, value2, value3,....);

Example

SELECT name

FROM employee

WHERE salary IN (20000, 30000, 40000);