LECTURE 8

SQL OPERATORS

1. Arithmetic Operators

Used for mathematical operations in SQL.

- **+ (Addition)** Example: SELECT salary + 500 AS increased_salary FROM employees; **Use Case**: Increase employee salaries by a fixed bonus.
- **- (Subtraction)** Example: SELECT price - discount AS final_price FROM products; **Use Case**: Calculate the final price of a product after applying a discount.
- *** (Multiplication)** Example: SELECT quantity * price AS total_cost FROM orders; **Use Case**: Calculate total order cost based on quantity and price.
- **/ (Division)** Example: SELECT total_marks / subjects AS average_marks FROM students; **Use Case**: Compute the average marks per subject for students.
- **% (Modulus)** Example: SELECT id % 2 AS remainder FROM users; **Use Case**: Identify even or odd IDs in a dataset.

2. Comparison Operators

Used to compare values and filter data.

- **= (Equal to)** Example: SELECT * FROM employees WHERE department = 'HR'; **Use Case**: Retrieve all employees working in the HR department.
- **<> or != (Not equal to)** Example: SELECT * FROM employees WHERE department != 'HR'; **Use Case**: Retrieve employees who are not in the HR department.
- **> (Greater than)** Example: SELECT * FROM orders WHERE total_amount > 1000; **Use Case**: Find all orders exceeding a certain amount.
- **< (Less than)** Example: SELECT * FROM products WHERE stock < 10; **Use Case**: Identify products with low stock levels.
- **>= (Greater than or equal to)** Example: SELECT * FROM employees WHERE experience >= 5; **Use Case**: List employees with 5 or more years of experience.

- **<= (Less than or equal to)** Example: SELECT * FROM loans WHERE interest_rate <= 7; **Use Case**: Filter loans with affordable interest rates.

3. Bitwise Operators

Perform operations at the bit level.

- **& (AND)** Example: SELECT 6 & 3 AS result; (Result: 2) **Use Case**: Used in specific computations involving binary values.
- **| (OR)** Example: SELECT 6 | 3 AS result; (Result: 7) **Use Case**: Combine binary flags or permissions.
- **^ (XOR)** Example: SELECT 6 ^ 3 AS result; (Result: 5) **Use Case**: Used in encryption or checksum calculations.

4. Compound Operators

Combine operations with assignment.

- **+= (Add and assign)** Example: SET @total += 10; **Use Case**: Increment a variable dynamically in calculations.
- **-= (Subtract and assign)** Example: SET @total -= 10; **Use Case**: Decrease a value during a financial transaction.
- ***= (Multiply and assign)** Example: SET @total *= 2; **Use Case**: Double a variable's value in batch processing.
- **/= (Divide and assign)** Example: SET @total /= 2; **Use Case**: Halve a value during normalization.
- **&= (AND and assign)** Example: SET @flags &= 4; **Use Case**: Update bit-level permissions using AND logic.
- **|= (OR and assign)** Example: SET @flags |= 2; **Use Case**: Add a permission flag to the variable.
- **^= (XOR and assign)** Example: SET @flags ^= 3; **Use Case**: Toggle certain bits in a flag value.

5. Logical Operators

Used for logical comparisons.

- **AND** Example: SELECT * FROM employees WHERE age > 25 AND department = 'IT'; **Use Case**: Retrieve IT employees older than 25.
- **OR** Example: SELECT * FROM employees WHERE department = 'HR' OR department = 'Finance'; **Use Case**: Get employees working in HR or Finance.

- **NOT** Example: SELECT * FROM employees WHERE NOT age > 50; **Use Case**: Find employees who are 50 years old or younger.

6. Pattern Matching Operators

Used for flexible pattern-based searches.

- **LIKE with Wildcards**
  - **% (Multiple characters)**
    - **Start with**: SELECT * FROM customers WHERE name LIKE 'A%'; **Use Case**: Find names starting with "A".
    - **End with**: SELECT * FROM customers WHERE name LIKE '%son'; **Use Case**: Find names ending with "son".
    - **Contains**: SELECT * FROM customers WHERE name LIKE '%John%'; **Use Case**: Find names containing "John".
  - **_ (Single character)**
    - **Example**: SELECT * FROM customers WHERE name LIKE 'J_n'; **Use Case**: Find names with "J" as the first letter and "n" as the third.
  - **[] (Character range)** Example: SELECT * FROM employees WHERE name LIKE '[A-C]%'; **Use Case**: Find names starting with A, B, or C.
  - **[^] (Not in range)** Example: SELECT * FROM employees WHERE name LIKE '[^A-C]%'; **Use Case**: Exclude names starting with A, B, or C.

7. IN and BETWEEN Operators

Filter data based on a set of values or a range.

- **IN**: Checks if a value matches any value in a list.
  Example: SELECT * FROM employees WHERE department IN ('HR', 'IT');
  Use Case: Retrieve employees from specific departments.
- **BETWEEN**: Filters data within a range.
  Example: SELECT * FROM orders WHERE order_date BETWEEN '2023-01-01' AND '2023-12-31';
  Use Case: Get orders within a specific date range.

8. ANY and ALL Operators

Used with subqueries for flexible comparisons.

- **ANY**: Returns true if any condition matches.
  Example: SELECT * FROM products WHERE price > ANY (SELECT price FROM competitors);
  Use Case: Identify products priced higher than at least one competitor.
- **ALL**: Returns true if all conditions are satisfied.
  Example: SELECT * FROM products WHERE price > ALL (SELECT price FROM competitors);
  Use Case: Identify products priced higher than all competitors.