Conditional Statements in SQL

SQL provides various ways to handle conditional logic, enabling us to execute different operations based on specific conditions. The most commonly used conditional statements in SQL include:

1. **CASE Statement** (Used in SELECT, UPDATE, INSERT, etc.)
2. **IF Statement** (Used in procedural SQL, such as stored procedures)
3. **IF ELSE and IF EXISTS** (Used in procedural SQL for control flow)
4. **WHERE Clause** (Used for filtering data)
5. **HAVING Clause** (Used for filtering aggregate results)

---

1. CASE Statement

The CASE statement is used within queries to perform conditional evaluations.

Example 1: Using CASE in a SELECT Statement

```
SELECT
    id,
    name,
    salary,
    CASE
        WHEN salary > 5000 THEN 'High Salary'
        WHEN salary BETWEEN 3000 AND 5000 THEN
'Medium Salary'
        ELSE 'Low Salary'
    END AS salary_category

FROM Employees;
```

Explanation:

- If salary > 5000, the result is 'High Salary'.
- If salary is between 3000 and 5000, the result is 'Medium Salary'.
- Otherwise, the result is 'Low Salary'.

Example 2: Using CASE in an UPDATE Statement

```
UPDATE Employees
SET salary = CASE
                WHEN salary < 3000 THEN salary + 500
-- Increase low salaries
                WHEN salary BETWEEN 3000 AND 5000
THEN salary + 300 -- Moderate increase
                ELSE salary  -- Keep salary unchanged

            END;
```

Explanation:

- If salary is below 3000, increase it by 500.
- If salary is between 3000 and 5000, increase it by 300.
- Otherwise, keep the salary unchanged.

---

2. IF Statement (Procedural SQL)

The IF statement is used inside **stored procedures or functions**.

Example: Using IF Inside a Stored Procedure

```
DELIMITER //
CREATE PROCEDURE check_salary(IN emp_salary INT)
BEGIN
    IF emp_salary > 5000 THEN
        SELECT 'High Salary';
    ELSEIF emp_salary BETWEEN 3000 AND 5000 THEN
        SELECT 'Medium Salary';
    ELSE
        SELECT 'Low Salary';
    END IF;
END //

DELIMITER ;
```

Explanation:

- The procedure check_salary takes emp_salary as input.

- It categorizes the salary into 'High Salary', 'Medium Salary', or 'Low Salary' accordingly.

---

3. IF EXISTS (Conditional Execution)

The IF EXISTS statement checks if a condition is met before executing a query.

Example: Delete an Employee Only If Exists

```
IF EXISTS (SELECT 1 FROM Employees WHERE id = 10)
BEGIN
    DELETE FROM Employees WHERE id = 10;

END;
```

Explanation:

- If an employee with id = 10 exists, the query deletes that employee.

---

4. WHERE Clause (Basic Conditional Filtering)

The WHERE clause is used to filter rows based on conditions.

Example: Find Employees with High Salary

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

Explanation:

- Returns only employees whose salary is greater than 5000.

---

5. HAVING Clause (Conditional Filtering for Aggregates)

The HAVING clause is used to filter results after applying aggregate functions.

Example: Find Departments with an Average Salary Above 4000

```sql
SELECT department, AVG(salary) AS avg_salary
FROM Employees
GROUP BY department
HAVING AVG(salary) > 4000;
```

Explanation:

- Groups employees by department.
- Calculates AVG(salary).
- Returns only departments where the avg_salary is greater than 4000.

Conclusion

SQL provides multiple ways to implement conditional logic:

- CASE is used for inline conditional logic within queries.
- IF and IF EXISTS are used in procedural SQL for control flow.
- WHERE and HAVING are used for filtering data in queries.

Each method serves different purposes, depending on the **context** and **database system** (MySQL, SQL Server, PostgreSQL, etc.).