TOPS TECHNOLOGY

Module 4 – Introduction to DBMS

Presented By :

Nandni Vala

SQL Syntax

1. What are the basic components of SQL syntax?

- The basic components of SQL syntax are the building blocks used to write and execute SQL queries for managing and interacting with relational databases.
- **1.Keywords :**
- Reserved words that define specific SQL operations.
- Examples:
  - SELECT, FROM, WHERE, INSERT, UPDATE, DELETE, CREATE, DROP, JOIN, etc.
- Keywords are not case-sensitive (though typically written in uppercase for readability)
- **2. Statements :**
- Commands or instructions that perform specific tasks in SQL.
- Common Types:
  - **Data Definition Language (DDL)**: CREATE, ALTER, DROP
  - **Data Manipulation Language (DML)**: SELECT, INSERT, UPDATE, DELETE
  - **Data Control Language (DCL)**: GRANT, REVOKE
  - **Transaction Control Language (TCL)**: COMMIT, ROLLBACK, SAVEPOINT

- **3. Clauses :**
- Components of SQL statements that specify conditions or modify behavior.
- Examples:
  - WHERE: Filters rows based on conditions.
  - GROUP BY: Groups rows that have the same values in specified columns.
  - ORDER BY: Sorts results in ascending or descending order.
  - HAVING: Filters groups based on conditions.
  - **4. Expressions :**
- Combinations of columns, constants, and operators that produce a single value.
- Examples:
  - Arithmetic: salary + bonus
  - String: CONCAT(first_name, ' ', last_name)
  - Logical: age > 18 AND city = 'New York'

5.Functions :

- Built-in methods that perform operations and return results.
- Types:
  - **Aggregate Functions**: SUM(), COUNT(), AVG(), MAX(), MIN()
  - **String Functions**: UPPER(), LOWER(), TRIM(), SUBSTRING()
  - **Date and Time Functions**: NOW(), DATE(), DATEDIFF()

2. Write the general structure of an SQL SELECT statement.

- Explanation of Each Clause:
- **SELECT**:
  - Specifies the columns you want to retrieve.
  - Use * to select all columns.
  - Example: SELECT name, age or SELECT *.

- **FROM**:
- Specifies the table(s) from which to retrieve the data.
- Example: FROM employees.
- **WHERE** *(optional)*:
- Filters rows based on specific conditions.
- Example: WHERE age > 30 AND department = 'Sales'.
- **GROUP BY** *(optional)*:
- Groups rows that have the same values in specified columns.
- Often used with aggregate functions like SUM, COUNT, etc.
- Example: GROUP BY department.
- **HAVING** *(optional)*:
- Filters groups based on conditions (used with GROUP BY).
- Example: HAVING COUNT(*) > 5.

- **ORDER BY** *(optional)*:
- Sorts the result set by one or more columns.
- Default sorting is ascending (ASC). Use DESC for descending.
- Example: ORDER BY age DESC.

3. Explain the role of clauses in SQL statements.

- Clauses in SQL statements define the structure and functionality of a query, specifying the operations to be performed on the data.
- **SELECT Clause**
- **Role**: Specifies the columns or data to retrieve from a table.
- **Functionality**:
  - Determines what data to include in the result set.
  - Can include expressions, functions, and aliases.
- **Example**:

- SELECT name, age FROM employees;
- **FROM Clause**
- **Role**: Identifies the table(s) from which to retrieve data.
- **Functionality**:
  - Serves as the source of data for the query.
  - Supports joins to combine data from multiple tables.
- **Example**:
- SELECT * FROM employees;
- **WHERE Clause**
- **Role**: Filters rows based on specified conditions.
- **Functionality**:
  - Eliminates rows that do not meet the condition(s).
  - Supports multiple conditions using AND, OR, and NOT.

- **Example :**
- SELECT * FROM employees WHERE age > 30;
- **GROUP BY Clause**
- **Role**: Groups rows that have the same values in specified columns.
- **Functionality**:
  - Used with aggregate functions (SUM, COUNT, AVG, etc.) to calculate values for each group.
- **Example**:
- SELECT department, COUNT(*) FROM employees GROUP BY department;
- **ORDER BY Clause**
- **Role**: Sorts the result set based on one or more columns.

- **Functionality**:
- Determines the order in which rows are displayed.
- Supports ascending (ASC, default) or descending (DESC) order.
- **Example**:
- SELECT name, age FROM employees ORDER BY age DESC;
- **JOIN Clause**
- **Role**: Combines rows from two or more tables based on related columns.
- **Functionality**:
  - Enables retrieving data from multiple tables in a single query.
  - Types: INNER JOIN, LEFT JOIN, RIGHT JOIN, FULL OUTER JOIN, CROSS JOIN.
- **Example**:
- SELECT employees.name, departments.name
- FROM employees
- JOIN departments ,ON employees.department_id = departments.id;

- DELETE: To remove records from a table.
- SELECT: To retrieve data from one or more tables.

- **Data Control Language (DCL):**
- **Purpose**: Manages access permissions and security of the database.
- **Key Commands**:
  - GRANT: To give access rights to users.
  - REVOKE: To remove access rights from users.
  - **4. Transaction Control Language (TCL):**
- **Purpose**: Manages database transactions to ensure data consistency.
- **Key Commands**:
  - COMMIT: To save changes made during the transaction.
  - ROLLBACK: To undo changes made during a transaction.
  - SAVEPOINT: To set a point in a transaction to which you can roll back.
  - **5. Querying Capabilities:**
- SQL allows complex queries to retrieve data using:
  - **Joins**: Combines data from multiple tables.
  - **Subqueries**: Embeds queries within queries.
  - **Aggregate Functions**: Performs calculations on data (SUM, COUNT, AVG, etc.).
  - **Grouping and Sorting**: Groups data (GROUP BY) and sorts results (ORDER BY).