SQL Training: Beginner to Intermediate Syllabus

Introduction

This tutorial is designed for individuals who are new to SQL or have some basic understanding and wish to deepen their knowledge to an intermediate level. We'll cover fundamental SQL concepts, progress to more advanced topics like complex joins, subqueries, and data manipulation, all while using a set of interconnected tables to demonstrate practical applications.

Target Audience

- Beginners with no prior SQL experience.
- Individuals who have used SQL but want to solidify their understanding of core concepts and explore intermediate features.
- Anyone looking to improve their data querying and manipulation skills.

Prerequisites

- No prior SQL knowledge is required.
- Basic computer literacy.
- A SQL environment (e.g., MySQL, PostgreSQL, SQL Server, SQLite) to practice the code examples. We'll primarily use standard SQL syntax that should be compatible across most systems.

Learning Objectives

By the end of this tutorial, you will be able to:

- Understand relational database concepts.
- Write basic to intermediate SQL queries for data retrieval.
- Filter, sort, and group data effectively.
- Perform various types of joins to combine data from multiple tables.
- Utilise subqueries for more complex data retrieval.
- Perform data manipulation operations (**INSERT**, **UPDATE**, **DELETE**).
- Understand and apply aggregate functions.
- Work with common SQL functions.
- Create and manage tables.

Module 1: Introduction to Databases and SQL

- **1.1 What is a Database?**
    - Relational Databases (RDBMS) vs. Non-Relational Databases (NoSQL)
    - **Tables**, **Rows**, **Columns** (Entities, Attributes, Records)

    - **Primary Keys** and **Foreign Keys** (Relationships)
    - Schemas and Databases
- **1.2 What is SQL?**
    - SQL Standards
    - Types of SQL Commands: **DDL**, **DML**, **DCL**, **TCL**
- **1.3 Setting Up Your Environment (Brief Overview)**
    - Common RDBMS: MySQL, PostgreSQL, SQL Server, SQLite
    - Tools for executing SQL queries (e.g., command line, GUI tools like DBeaver, DataGrip, MySQL Workbench, pgAdmin, **SSMS**)

Module 2: Designing Our Database and Tables

- **2.1 Database Schema Design**
    - We'll create a simple e-commerce database.
    - **Tables:**
        - Customers
        - Products:
        - Orders:
        - Order_Items:
        - Categories:
        - Suppliers:

- **2.2 SQL DDL: Creating Tables**
    - **CREATE DATABASE**
    - **CREATE TABLE** statement with **PRIMARY KEY**, **FOREIGN KEY**, **NOT NULL**, **UNIQUE**, **DEFAULT** constraints.
    - Data Types (**INT**, **VARCHAR**, **DECIMAL**, **DATE**, **BOOLEAN**, etc.)

Module 3: Basic Data Manipulation (DML) and Retrieval

- **3.1 Inserting Data (INSERT INTO)**
    - Inserting single rows
    - Inserting multiple rows
    - Inserting specific columns
- **3.2 Retrieving Data (SELECT)**
    - **SELECT *** (all columns)
    - **SELECT column1, column2** (specific columns)
    - **AS** (aliasing columns)
- **3.3 Filtering Data (WHERE)**
    - Comparison Operators: =, !=, >, <, >=, <=
    - Logical Operators: **AND**, **OR**, **NOT**
    - Special Operators: **LIKE**, **IN**, **BETWEEN**, **IS NULL**, **IS NOT NULL**
- **3.4 Sorting Data (ORDER BY)**
    - **ASC** (ascending), **DESC** (descending)
    - Sorting by multiple columns

- **3.5 Limiting Results (TOP)**
    - **TOP** (SQL Server)

Module 4: Aggregate Functions and Grouping

- **4.1 Aggregate Functions**
    - **COUNT()** (counting rows, counting non-null values)
    - **SUM()** (summing numeric values)
    - **AVG()** (calculating average)
    - **MIN()** (finding minimum value)
    - **MAX()** (finding maximum value)
- **4.2 Grouping Data (GROUP BY)**
    - Grouping by one or multiple columns
- **4.3 Filtering Groups (HAVING)**
    - Distinction between **WHERE** and **HAVING**

Module 5: Joining Tables (Intermediate Level)

- **5.1 Understanding Joins**
    - Why we need joins
    - Types of relationships (One-to-One, One-to-Many, Many-to-Many)
- **5.2 INNER JOIN**
- **5.3 LEFT JOIN** (or LEFT OUTER JOIN)
- **5.4 RIGHT JOIN** (or RIGHT OUTER JOIN)
- **5.5 FULL JOIN** (or FULL OUTER JOIN)
- **5.6 SELF JOIN**
- **5.7 Multiple Joins**
    - Joining three or more tables

Module 6: Subqueries and Common Table Expressions (CTEs)

- **6.1 Subqueries (Nested Queries)**
    - Subqueries in **WHERE** clause (**IN**, **EXISTS**, comparison operators)
    - Subqueries in **FROM** clause (derived tables)
    - Subqueries in **SELECT** clause (scalar subqueries)
- **6.2 Common Table Expressions (CTEs) (WITH clause)**
    - Advantages of CTEs over subqueries
    - Defining and using CTEs
    - Recursive CTEs (brief mention for advanced)

Module 7: Data Manipulation (DML) - Advanced

- **7.1 Updating Data (UPDATE)**
    - Updating single column
    - Updating multiple columns
    - Using **WHERE** clause for conditional updates
- **7.2 Deleting Data (DELETE)**
    - Deleting specific rows
    - **TRUNCATE TABLE** vs. **DELETE FROM** (performance and logging)
- **7.3 Renaming Tables/Columns (ALTER TABLE)**
    - **ALTER TABLE ... RENAME TO** (or **RENAME COLUMN**)
- **7.4 Dropping Tables (DROP TABLE)**
    - Understanding the impact of **DROP TABLE**
    - **DROP DATABASE**

Module 8: Advanced SQL Concepts (Briefly Touched)

- **8.1 UNION and UNION ALL**
    - Combining result sets
    - Differences between **UNION** (removes duplicates) and **UNION ALL** (keeps duplicates)
- **8.2 CASE Statement**
    - Conditional logic in queries
- **8.3 Common SQL Functions**
    - String Functions: **LENGTH**, **UPPER**, **LOWER**, **SUBSTRING**, **CONCAT**
    - Numeric Functions: **ROUND**, **CEIL**, **FLOOR**
    - Date and Time Functions: **CURRENT_DATE**, **CURRENT_TIMESTAMP**, **DATE_ADD**, **DATEDIFF** (RDBMS specific)
- **8.4 Views**
    - Creating virtual tables for simplified queries
    - Benefits and use cases