FULL STACK DEVELOPMENT

Unit-5

Imp Q & A

1) What is SQL? define some commands DDL,DML,DQL?

Ans) SQL commands are essential for managing databases effectively. These commands are divided into categories such as Data Definition Language (**DDL**), Data Manipulation Language (**DML**), Data Control Language (**DCL**), Data Query Language (**DQL**), and Transaction Control Language (**TCL**).

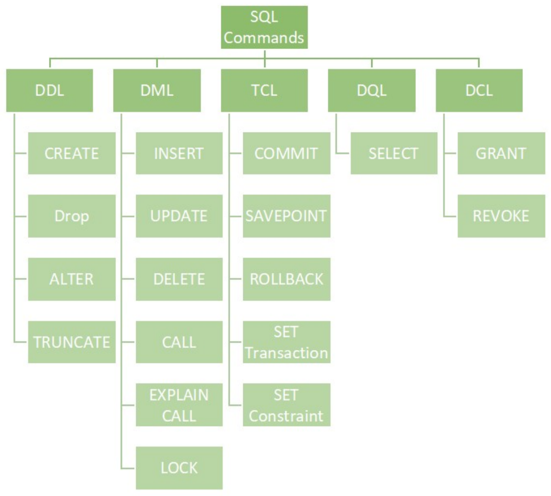
In this article, we will explain the different **types of SQL commands**, including **DDL**, **DML**, **DCL**, **DQL**, and **TCL**. These **SQL sublanguages** serve specific purposes and are important for effective **database management**.

What are SQL Commands?

SQL Commands are like instructions to a table. It is used to interact with the database with some operations. It is also used to perform specific tasks, functions, and queries of data. SQL can perform various tasks like creating a table, adding data to tables, dropping the table, modifying the table, set permission for users.

SQL Commands are mainly categorized into five categories:

- **DDL** – Data Definition Language
- **DQL** – Data Query Language
- **DML** – Data Manipulation Language
- **DCL** – Data Control Language
- **TCL** – Transaction Control Language



1. Data Definition Language (DDL) in SQL

DDL or Data Definition Language actually consists of the SQL commands that can be used to **defining**, **altering**, and **deleting** database structures such as **tables**, **indexes**, and **schemas**. It simply deals with descriptions of the database schema and is used to **create** and **modify** the structure of database objects in the database

Common DDL Commands

Command	**Description**	**Syntax**
CREATE	Create database or its objects (table, index, function, views, store procedure, and triggers)	**CREATE** TABLE table_name (column1 data_type, column2 data_type, ...);
DROP	Delete objects from the database	**DROP** TABLE table_name;
ALTER	Alter the structure of the database	**ALTER** TABLE table_name ADD COLUMN column_name data_type;
TRUNCATE	Remove all records from a table, including all spaces allocated for the records are removed	**TRUNCATE** TABLE table_name;
COMMENT	Add comments to the data dictionary	**COMMENT** 'comment_text' ON TABLE table_name;
RENAME	Rename an object existing in the database	**RENAME** TABLE old_table_name TO new_table_name;

Example of DDL

```
CREATE TABLE employees (
   employee_id INT PRIMARY KEY,
   first_name VARCHAR(50),
   last_name VARCHAR(50),
   hire_date DATE
);
```

In this example, a new table called employees is created with columns for employee ID, first name, last name, and hire date.

2. Data Query Language (DQL) in SQL

DQL statements are used for performing queries on the data within **schema objects**. The purpose of the DQL Command is to get some **schema relation** based on the query passed to it. This command allows getting the data out of the database to perform operations with it. When a SELECT is fired against a table or tables the result is compiled into a further **temporary table**, which is displayed or perhaps received by the program.

DQL Command

Command	**Description**	**Syntax**
SELECT	It is used to retrieve data from the database	SELECT column1, column2, ...FROM table_name WHERE condition;

Example of DQL

```
SELECT first_name, last_name, hire_date
FROM employees
WHERE department = 'Sales'
ORDER BY hire_date DESC;
```

This query retrieves employees' first and last names, along with their hire dates, from the employees table, specifically for those in the 'Sales' department, sorted by hire date.

3. Data Manipulation Language (DML) in SQL

The **SQL commands** that deal with the **manipulation of data** present in the database belong to **DML** or Data Manipulation Language and this includes most of the **SQL statements.** It is the component of the SQL statement that controls access to data and to the database. Basically, DCL statements are grouped with DML statements.

Common DML Commands

Command	Description	Syntax
INSERT	Insert data into a table	**INSERT** INTO table_name (column1, column2, ...) VALUES (value1, value2, ...);
UPDATE	Update existing data within a table	**UPDATE** table_name SET column1 = value1, column2 = value2 WHERE condition;
DELETE	Delete records from a database table	**DELETE** FROM table_name WHERE condition;
LOCK	Table control concurrency	**LOCK** TABLE table_name IN lock_mode;
CALL	Call a PL/SQL or JAVA subprogram	**CALL** procedure_name(arguments);
EXPLAIN PLAN	Describe the access path to data	**EXPLAIN PLAN** FOR SELECT * FROM table_name;

Example of DML

```
INSERT INTO employees (first_name, last_name, department)
VALUES ('Jane', 'Smith', 'HR');
```

This query inserts a new record into the employees table with the first name ‘Jane’, last name ‘Smith’, and department ‘HR’.

4. Data Control Language (DCL) in SQL

DCL (Data Control Language) includes commands such as **GRANT** and **REVOKE** which mainly deal with the **rights**, **permissions**, and other controls of the **database system**. These commands are used to **control access** to data in the database by **granting** or **revoking permissions**.

Common DCL Commands

Command	Description	Syntax
GRANT	Assigns new privileges to a user account, allowing access to specific database objects, actions, or functions.	**GRANT** privilege_type [(column_list)] ON [object_type] object_name TO user [WITH GRANT OPTION];
REVOKE	Removes previously granted privileges from a user account, taking away their access to certain database objects or actions.	**REVOKE** [GRANT OPTION FOR] privilege_type [(column_list)] ON [object_type] object_name FROM user [CASCADE];

Example of DCL

GRANT SELECT, UPDATE ON employees TO user_name;

This command grants the user user_name the permissions to select and update records in the employees table.

5. Transaction Control Language (TCL) in SQL

Transactions group a set of tasks into a **single execution unit**. Each transaction begins with a specific task and ends when all the tasks in the group are successfully completed. If any of the **tasks fail**, the transaction fails. Therefore, a transaction has only two results: **success** or **failure**. We can explore more about **transactions** here.

Common TCL Commands

Command	Description	Syntax
BEGIN TRANSACTION	Starts a new transaction	BEGIN TRANSACTION [transaction_name];
COMMIT	Saves all changes made during the transaction	COMMIT;

Command	Description	Syntax
ROLLBACK	Undoes all changes made during the transaction	ROLLBACK;
SAVEPOINT	Creates a savepoint within the current transaction	SAVEPOINT savepoint_name;

Example of TCL

```
BEGIN TRANSACTION;
UPDATE employees SET department = 'Marketing' WHERE department = 'Sales';
SAVEPOINT before_update;
UPDATE employees SET department = 'IT' WHERE department = 'HR';
ROLLBACK TO SAVEPOINT before_update;
COMMIT;
```

In this example, a transaction is started, changes are made, and a savepoint is set. If needed, the transaction can be rolled back to the savepoint before being committed.

Important SQL Commands

1. **SELECT**: Used to retrieve data from a database.
2. **INSERT**: Used to add new data to a database.
3. **UPDATE**: Used to modify existing data in a database.
4. **DELETE**: Used to remove data from a database.
5. **CREATE TABLE**: Used to create a new table in a database.
6. **ALTER TABLE**: Used to modify the structure of an existing table.
7. **DROP TABLE**: Used to delete an entire table from a database.
8. **WHERE**: Used to filter rows based on a specified condition.
9. **ORDER BY**: Used to sort the result set in ascending or descending order.
10. **JOIN**: Used to combine rows from two or more tables based on a related column between them.

2) What is data Persistance using spring JDBC write a sample code?

Ans) Data Persistence using Spring JDBC

Spring JDBC provides an easy way to interact with databases. It helps in **data persistence** by allowing applications to store, retrieve, update, and delete data from a relational database using JDBC.

Steps to Implement Spring JDBC

1. **Add Dependencies**: Spring Boot Starter JDBC, MySQL Driver
2. **Configure Database Connection (application.properties)**
3. **Create Model (Student.java)**
4. **Create Repository (StudentRepository.java)**
5. **Create Service (StudentService.java)**
6. **Create Controller (StudentController.java)**

Add Dependencies (pom.xml)

```
<dependencies>
  <!-- Spring Boot Starter JDBC -->
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-jdbc</artifactId>
  </dependency>

  <!-- MySQL Driver -->
  <dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <scope>runtime</scope>
  </dependency>
</dependencies>
```

Create the database in MySQL:

```
CREATE DATABASE studentdb;
USE studentdb;
CREATE TABLE students (
  id INT PRIMARY KEY AUTO_INCREMENT,
  name VARCHAR(100),
  course VARCHAR(100)
);
```

Create Student Model

```
package com.example.studentapp.model;

public class Student {
    private int id;
    private String name;
    private String course;

    public Student() {}

    public Student(int id, String name, String course) {
        this.id = id;
        this.name = name;
        this.course = course;
    }

    public int getId() { return id; }
    public void setId(int id) { this.id = id; }

    public String getName() { return name; }
    public void setName(String name) { this.name = name; }

    public String getCourse() { return course; }
    public void setCourse(String course) { this.course = course; }
}
```

API Testing (Using Postman or Browser)

HTTP Method	**Endpoint**	**Description**
GET	/students	Get all students
GET	/students/{id}	Get student by ID
POST	/students	Add a new student
PUT	/students/{id}	Update student
DELETE	/students/{id}	Delete student