01: Spring JDBC Introduction

Spring JDBC simplifies the use of JDBC (Java Database Connectivity) by providing an abstraction through the JDBC Template class.

👉 JDBC Steps Without Spring:

In JDBC, the following steps are required to interact with a database:

1. **Load the Driver:** Load the JDBC driver for the database being used.
2. **Define Connection URL:** Define the URL where the database is hosted along with credentials.
3. **Establish Connection:** Establish a connection to the database using DriverManager.
4. **Create Statement Object:** Create a Statement or PreparedStatement object to execute queries.
5. **Execute Query:** Execute the query and retrieve the result using ResultSet.
6. **Process the Result:** Process the ResultSet to fetch the required data.
7. **Close the Connection:** Close the connection, statement, and result set to avoid memory leaks.

👉 Spring JDBC Template:

The JDBC Template in Spring simplifies the process of working with databases by reducing boilerplate code. It internally handles:

- **Connection with database**: It manages connection to the database using a DataSource.
- **Query Execution:** SQL queries can be executed directly using provided methods (queryForObject, query, update, etc.).
- **Output Retrieval:** Retrieve results from the query in the form of objects, lists, or directly as primitive data types.

👉 Datasource in Spring JDBC:

- DataSource is used by Spring JDBC to provide a connection to the database. It acts as a factory for Connection objects.
- Spring can manage the DataSource using connection pooling, improving the application's performance by reusing connections.

👉 In-Memory Database (H2):

- Spring Boot provides built-in support for the H2 database, which is an in-memory database.
- By default in H2, the data is stored temporarily in memory, and it is lost when the application is stopped or restarted.
- H2 is ideal for development and testing environments where persistent storage is not required.