1. **Difference between JPA, Hibernate and Spring Data JPA.**

Java Persistence API (JPA)

- JPA stands for Java Persistence API.
- It is a specification (JSR 338) provided by Java EE to define a standard for persisting Java objects into relational databases.
- JPA itself does not provide any implementation — it only defines interfaces such as EntityManager, Entity, and Query.
- It is vendor-independent, meaning any ORM tool can implement this specification (e.g., Hibernate, EclipseLink).

Hibernate

- Hibernate is an **Object-Relational Mapping (ORM)** tool for Java.
- It is one of the **most popular implementations of JPA**.
- It provides additional features beyond the JPA specification like:
- HQL (Hibernate Query Language)
- First- and second-level caching
- Dirty checking
- Lazy loading
- Hibernate can work both with **JPA-based** and **native Hibernate APIs**.

Spring Data JPA

- Spring Data JPA is a **Spring framework module** built on top of JPA and commonly uses **Hibernate** as the JPA provider.
- It provides an **abstraction layer over JPA** to reduce boilerplate code for database access.

Features:

- Auto-generates repository implementations
- Supports method query derivation (e.g., findByUsername())
- Integrates easily with Spring Boot

Summary of Differences

Feature	JPA	Hibernate	Spring Data JPA
Type	Specification	Implementation	Spring abstraction
Provided By	Java EE	Hibernate ORM project	Spring Framework
Boilerplate Reduction	No	Partial	Yes
Implementation	None	Yes	Uses Hibernate or other JPA providers
Main Role	API definition for persistence	ORM framework	Simplifies data access using JPA

Use Cases

1.You want a **standardized API**	- JPA
2.You want **fine control** with more features	- Hibernate
3.You want **minimal code with Spring support**	- Spring Data JPA