**Difference between JPA, Hibernate and Spring Data JPA :**

**JPA (Java Persistence API)**

JPA is a specification provided by Jakarta EE (formerly Java EE) for object-relational mapping (ORM) in Java. It defines a standard set of interfaces and annotations (like @Entity, @Id, EntityManager) for managing relational data in Java applications. JPA itself does not provide an implementation—it just outlines how ORM should work, and requires a provider (like Hibernate) to do the actual work.

**Hibernate**

Hibernate is a popular implementation of the JPA specification. It provides the concrete functionality for the ORM tasks that JPA defines. Hibernate also includes extra features not defined in JPA, such as caching, lazy loading, dirty checking, and custom HQL (Hibernate Query Language). You can use Hibernate either directly (without JPA) or as a JPA provider.

**Spring Data JPA**

Spring Data JPA is a part of the Spring Framework that builds on top of JPA. It simplifies data access by providing a repository abstraction layer. Developers only need to define interfaces for data access (like JpaRepository), and Spring automatically generates the required implementation at runtime. It reduces boilerplate code, and lets developers focus on the business logic rather than query building. Internally, Spring Data JPA still relies on a JPA provider like Hibernate.