WEEK:3

**Difference between JPA, Hibernate and Spring Data JPA**   
  
Java Persistence API (JPA)

JSR 338 Specification for persisting, reading and managing data from Java objects

Does not contain concrete implementation of the specification

Hibernate is one of the implementation of JPA

Hibernate

ORM Tool that implements JPA

Spring Data JPA

Does not have JPA implementation, but reduces boiler plate code

This is another level of abstraction over JPA implementation provider like Hibernate

Manages transactions

**Java Persistence API (JPA)**

JPA is a **specification** (JSR 338) that defines a standard for persisting, reading, and managing data from Java objects to relational databases.

It provides a set of **interfaces**, **annotations**, and **rules** for ORM (Object Relational Mapping).

JPA **does not provide an actual implementation** — it is only the standard (like a contract).

Developers need a JPA **provider/implementation** to use JPA (e.g., Hibernate, EclipseLink, OpenJPA).

Hibernate is a **popular ORM tool** and the most widely used **JPA implementation**.

It provides the actual code (implementation) that fulfills the JPA specification.

Hibernate can be used:

As a pure ORM tool with its own APIs.

As a JPA provider to comply with JPA standards.

Hibernate adds extra features beyond JPA, such as:

First-level and second-level caching

Lazy loading

Better fetching strategies

**Spring Data JPA**

* Spring Data JPA is **not a JPA implementation**.
* It is a **framework built on top of JPA (and its implementations like Hibernate)**.
* It provides an **abstraction layer** to reduce boilerplate repository code.
* Automatically generates standard CRUD operations — no need for manual DAO code.
* Seamlessly integrates with Spring features (e.g., transaction management, dependency injection).