# WEEK-3

# Difference between JPA, Hibernate, and Spring Data JPA

JPA, Hibernate, and Spring Data JPA are related to object-relational mapping (ORM) in Java applications. They differ in purpose, abstraction level, and usage.

## Comparison Table:

|  |  |  |  |
| --- | --- | --- | --- |
| **Aspect** | **JPA** | **Hibernate** | **Spring Data JPA** |
| **Definition** | Java Persistence API -a specification that defines ORM standards | An implementation of JPA with extended features | A Spring abstraction layer built on top of JPA for rapid data access |
| **Type** | API / Specification | Framework / Implementation | Framework / Abstraction Layer |
| **Provider** | Oracle / Jakarta EE | Red Hat | Spring (Pivotal / VMware) |
| **Configuration** | Needs a JPA provider like Hibernate, EclipseLink | Manual configuration of mappings and sessions | Auto-configuration with Spring Boot, reduces boilerplate |
| **Ease of Use** | Moderate — requires manual setup, EntityManager, etc. | More control with some overhead | Very simple — reduces boilerplate using Repository interfaces |
| **Query Language** | JPQL (Java Persistence Query Language) | JPQL + HQL (Hibernate Query Language) | JPQL + Derived Queries + @Query support |
| **Use Case** | Define ORM standards and abstract the persistence layer | Custom ORM needs with rich features and control | Quick CRUD operations in Spring projects with minimal code |

## Summary:

JPA is like a rulebook (API/specification). Hibernate is a popular engine that follows the rulebook (and adds extra features). Spring Data JPA lets you use that engine in a simple way with auto-wiring and minimal code.