5. Explain the difference between Java Persistence API, Hibernate and Spring Data JPA

1. Java Persistence API (JPA)

JPA (Java Persistence API) is a specification (JSR 338) provided by Oracle that defines a standard for object-relational mapping (ORM) in Java.

Key Features:

- It is only an interface/specification, not an implementation.
- Provides annotations like @Entity, @Table, @Id, and APIs for managing persistence contexts and transactions.

Differences JPA

- It is a specification, not a tool.
- Requires an implementation like Hibernate to function.
- Provides standard APIs and annotations for persistence.
- Promotes portability across ORM tools.

2. Hibernate

Hibernate is a popular ORM tool and a concrete implementation of the JPA specification.

Key Features:

- It supports both native Hibernate API and JPA-based API.
- Offers features beyond JPA like caching, lazy loading, and custom query language (HQL).
- Handles mapping of Java classes to database tables and automates SQL generation.

Differences Hibernate

- A concrete implementation of the JPA specification.
- Can work both with and without JPA.
- Offers extra features beyond JPA like caching and HQL.
- Requires more boilerplate code compared to Spring Data JPA.

3. Spring Data JPA

Spring Data JPA is a Spring-based abstraction built on top of JPA (and often uses Hibernate under the hood) that further simplifies data access.

Key Features:

- Reduces boilerplate code by providing interfaces like JpaRepository and CrudRepository.
- Integrates tightly with Spring Boot for rapid development.

Differences Spring Data JPA

- A Spring abstraction over JPA.
- Internally uses Hibernate (or another JPA provider).
- Highly integrated with Spring Boot for rapid development.