2200030216 WEEK-3 SPRING DATA JPA HANDSON

Difference between Java Persistence API, Hibernate and Spring Data JPA

**JPA (Java Persistence API)**

* JPA stands for Java Persistence API.
* It is just a specification (JSR 338) — i.e., it defines interfaces and rules for persisting Java objects to relational databases.
* JPA does not provide any actual implementation.
* Its main goal: Provide a standard ORM layer for Java.

## ****Hibernate****

* **Hibernate** is the **most popular implementation** of JPA.
* It is also a **standalone ORM framework** that existed before JPA.
* Hibernate:
  + Implements all JPA features.
  + Also adds **extra proprietary features** (like Hibernate Criteria API, native caching, etc.)

Hibernate = JPA + extras

## ****Spring Data JPA****

* **Spring Data JPA** is a **part of the Spring Data project**.
* It provides **abstractions over JPA (and thereby Hibernate)** to:
  + Eliminate boilerplate repository/DAO code.
  + Enable auto-implemented query methods using method names.
  + Easily switch between JPA providers (like Hibernate, EclipseLink, etc.).

Spring Data JPA uses Hibernate **under the hood** by default in Spring Boot.

### ****Hibernate****

**@Entity**

**@Table(name = "students")**

**public class Student {**

**@Id**

**private int id;**

**private String name;**

**}**

**public class StudentDao {**

**private SessionFactory sessionFactory;**

**public void saveStudent(Student student) {**

**Session session = sessionFactory.openSession();**

**Transaction tx = session.beginTransaction();**

**session.save(student);**

**tx.commit();**

**session.close();**

**}**

**}**

### ****JPA****

**@Entity**

**@Table(name = "students")**

**public class Student {**

**@Id**

**private int id;**

**private String name;**

**}**

**public class StudentDao {**

**@PersistenceContext**

**private EntityManager entityManager;**

**public void saveStudent(Student student) {**

**entityManager.persist(student);**

**}**

**}**

### ****Spring Data JPA****

**@Entity**

**@Table(name = "students")**

**public class Student {**

**@Id**

**private int id;**

**private String name;**

**}**

**public interface StudentRepository extends JpaRepository<Student, Integer> {**

**List<Student> findByName(String name);**

**}**