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

**Java Persistence API (JPA):**

* **JPA** is a **specification** that provides a standard for persisting, reading, and managing data between Java objects and relational databases.
* It is just an **interface layer,** it **does not provide a concrete implementation**.
* JPA defines how objects should be persisted in a relational database, but actual functionality is provided by an implementation such as **Hibernate**, **EclipseLink**, or **OpenJPA**.

**Hibernate**:

* **Hibernate** is a **popular Object-Relational Mapping (ORM) tool** that implements the JPA specification.
* It provides the **actual implementation** of JPA's standard.
* Hibernate offers additional features **beyond JPA**, such as:
* Caching mechanisms (first-level and second-level cache)
* Native SQL and HQL support
* Lazy/eager fetching strategies
* Developers using Hibernate directly have to **manually manage sessions, transactions, and exception handling**.

**Spring Data JPA**:

* **Spring Data JPA** is **not a JPA implementation**.
* It is an abstraction layer built on top of **JPA and implementation**  and is part of the larger Spring ecosystem.
* It significantly **reduces boilerplate code** by providing:
* Repository interfaces like JpaRepository.
* Built-in CRUD methods.
* Auto-generated queries using method names.
* It also manages **transactions automatically** with the help of Spring's @Transactional annotation.

**Hibernate Code Example:**

public Integer addEmployee(Employee employee){

Session session = factory.openSession();

Transaction tx = null;

Integer employeeID = null;

try {

tx = session.beginTransaction();

employeeID = (Integer) session.save(employee);

tx.commit();

} catch (HibernateException e) {

if (tx != null) tx.rollback();

e.printStackTrace();

} finally {

session.close();

}

return employeeID;

}

**Spring Data JPA Code Example:**

**EmployeeRepository.java**

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

**EmployeeService.java**

@Autowired

private EmployeeRepository employeeRepository;

@Transactional

public void addEmployee(Employee employee) {

employeeRepository.save(employee);

}