Java Persistence API (JPA)

* JPA is a **specification** (JSR 338) for managing relational data using Java objects.
* It defines **standard interfaces and annotations** to perform ORM (Object-Relational Mapping).
* **JPA does not provide an implementation**—it’s just a set of guidelines.
* Common implementations of JPA:
  + Hibernate
  + EclipseLink
  + OpenJPA

Hibernate

* **Hibernate** is an **Object-Relational Mapping (ORM)** tool and **implementation of JPA**.
* It provides a **concrete implementation** of JPA's specification, along with extra features.
* With Hibernate, developers manually manage:
  + Session lifecycle
  + Transactions
  + Queries (using HQL or Criteria API)

Spring Data JPA

* **Spring Data JPA** is **not a JPA implementation**.
* It is a **Spring abstraction layer** built on top of JPA providers like Hibernate.
* Reduces boilerplate by:
  + Providing built-in JpaRepository interfaces
  + Automatically generating query methods
  + Managing transactions with Spring annotations

Code Comparison: Hibernate vs. Spring Data JPA

**Hibernate Code**

Method to CREATE an employee in the database \*/  
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**

// EmployeeRepository.java  
public interface EmployeeRepository extends JpaRepository<Employee, Integer> {  
}  
  
EmployeeService.java  
@Autowired  
private EmployeeRepository employeeRepository;  
  
@Transactional  
public void addEmployee(Employee employee) {  
 employeeRepository.save(employee);  
}

**Side-by-Side Comparison**

|  |  |  |
| --- | --- | --- |
| **Feature** | **Hibernate** | **Spring Data JPA** |
| Boilerplate Code | High – manual session, transaction, save | Minimal – repository abstraction |
| Transaction Handling | Manual via Transaction object | Auto-managed via @Transactional |
| Repository/DAO Layer | Manual DAO class with session factory | Interface-based using JpaRepository |
| Query Writing | HQL or Criteria API | Method names (findByName), JPQL, native |
| Implementation Type | JPA Implementation | Abstraction on top of JPA implementations |