Difference between JPA, Hibernate and Spring Data JPA

# Java Persistence API (JPA)

- JPA is a specification (JSR 338) that defines a standard for ORM in Java.  
- It doesn't provide actual code/implementation.  
- It only defines interfaces, annotations like @Entity, @Id, etc.  
- Requires a provider (like Hibernate) to work.

Example:

@Entity  
public class Employee {  
 @Id  
 private int id;  
 private String name;  
}

# Hibernate

- Hibernate is a concrete implementation of JPA.  
- It's a full-featured ORM tool that maps Java classes to DB tables.  
- It manages:  
 - Connection to DB  
 - Caching  
 - SQL generation  
 - Lazy loading  
 - Transactions

Example (Manual Hibernate):

Session session = factory.openSession();  
Transaction tx = session.beginTransaction();  
session.save(employee);  
tx.commit();  
session.close();

# Spring Data JPA

- Abstraction layer over JPA + Hibernate.  
- Simplifies database access by removing boilerplate code.  
- Built on top of Spring Framework and Hibernate.  
- Automatically handles:  
 - Repositories (JpaRepository)  
 - Transactions (@Transactional)  
 - Query generation (findByName, etc.)

Example:

public interface EmployeeRepository extends JpaRepository<Employee, Integer> { }  
  
@Autowired  
EmployeeRepository repository;  
  
repository.save(employee);

# Comparison Table

|  |  |  |  |
| --- | --- | --- | --- |
| Feature | JPA | Hibernate | Spring Data JPA |
| Type | Specification | Implementation of JPA | Framework built on JPA + Hibernate |
| Code Provided? | No | Yes | Yes |
| SQL Handling | No | Yes | Yes |
| Boilerplate Reduction | No | Partial | High |
| Query Methods | Manual or JPQL | HQL | Method name-based queries |
| Transaction Management | External (Spring, etc.) | Manual or Spring-managed | Handled automatically with @Transactional |
| Used In | Java EE, Spring | Standalone or in Spring | Spring Framework only |

# Real Code Difference

Hibernate Example:

Session session = factory.openSession();  
Transaction tx = session.beginTransaction();  
session.save(employee);  
tx.commit();  
session.close();

Spring Data JPA Example:

employeeRepository.save(employee); // one-liner!