**Difference Between the JPA, Hibernate, and Spring Data JPA**

# Java Persistence API (JPA)

* JPA is a **standard specification** (JSR 338) provided by Java for ORM
* It defines a **set of rules and annotations** to persist Java objects into relational databases.
* JPA itself **does not provide an implementation**. It only specifies *how* the interaction between Java and the database should occur.
* Common implementations of JPA include **Hibernate**.
* **Example:**

|  |
| --- |
| @Entity  public class Students {  @Id private int id1;  private String name;  } |

# Hibernate

* Hibernate is the **concrete implementation** of the JPA specification.
* It’s also an advanced **ORM framework** that offers features **beyond JPA**, such as:
* Caching
* Lazy/eager intialization
* Criteria API
* Hibernate **manages database connections**, **SQL generation**, and **object lifecycle**.

**Traditional Hibernate Code Example:**

|  |
| --- |
| Configuration conf = new Configuration().configure();  SessionFactory factory = conf.buildSessionFactory();  Session ses = factory.openSession();  Transaction tran = ses.beginTransaction();    Student students = new Student(1, "Vilendiran"); session.save(student);    tran.commit();  ses.close(); |

**3. Spring Data JPA**

* Spring Data JPA is part of the **Spring Framework**.
* It provides an abstraction over JPA by **removing boilerplate code** like writing DAOs or CRUD operations.
* It internally uses a **JPA provider** like Hibernate but makes it easier to use via **repositories** and **annotations**.
* It supports features like **pagination, sorting, and derived queries** out of the box.

**Comparison** :

# 🔹 Hibernate Example

public Integer addEmployee(Employee employee) {

Session session = factory.openSession();

Transaction tx = null;

# Spring Data JPA Example

**Repository Interface**

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

Service Class

@Service

public class

EmployeeService {

Integer empId = null;

try {

tx = session.beginTransaction();

empId = (Integer) ses.save(employee);

tx.commit();

}

catch (Exception e)

{

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

e.printStackTrace();

}

finally

{

session.close();

}

return empId;

}

@Autowired

private EmployeeRepository employeeRepository;

public void addEmployee(Employee employee) {

employeeRepository.save(employee);

}

}