**JPA(Java Persistence API) :**

JPA is specification provided by java or it is standard way to map java object in RBMS,it provide only specification not implementation so you will need to use JPA specification provider framework like HIBERNATE

 JPA acts as a bridge between object-oriented domain models and relational database systems. As JPA is just a specification, it doesn't perform any operation by itself.

The Java Persistence API (JPA) is a specification of Java. It is used to persist data between Java object and relational database. JPA acts as a bridge between object-oriented domain models and relational database systems.

As JPA is just a specification, it doesn't perform any operation by itself. It requires an implementation. So, ORM tools like Hibernate, TopLink and iBatis implements JPA specifications for data persistence.

**Spring ORM-Object To Relational Mapping**

Less coding required

Easy to test

Better exception handling

Integrated transaction management

Normal Approach

Java -----------------🡪 JDBC -------------------🡪 Database

Program

ORM Approach

Java ----------------🡪ORM Database

Program

In Spring JDBC we have JdbcTemplate class to execute query and in spring ORM we have HibernateTemplate class

ProductDao

DataAccessObject-contain all method which perform all operation on product like CRUD

Product Dao-------------------HibernateTemplate SessionFactory

LocalSessionFactory

DataSource

(url,username,pass)

**Spring Framework with Hibernate integration(using maven):**

Advantages of ORM framework with spring :

1.Less coding.

2.Easy to test

3.No need to handle exception explicitly.

4.Integrated transaction management

**Why ?**

**1.No need to define hibernate.cfg.xml file separately.**

**2.We provide all configuration in spring-config.xml file**

**3.Hibernate Steps :**

Configuration cfg=**new** Configuration();

cfg.configure("hibernate.cfg.xml");

SessionFactory factory=cfg.buildSessionFactory();

Session session=factory.openSession();

Transaction tx=session.beginTransaction();

We no need to follow these steps when spring framework integrate with Hibernate because spring having predefine HibernateTemplate class we simply use its reference.

**Steps to create spring app with hibernate or Spring ORM application**

**Step 1)**New🡪project🡪maven project🡪simple project🡪(groupId,artifactId,name,jar file)--🡪finish

**Step2)**open pom.xml🡪add spring core,spring context,mysql-connector and very important spring-orm dependencies due to this we able to use HibernateTemplate class reference to perform operation on Database

*Spring ORM release==Spring context release*

**Also add hibernate-core and spring orm dependencies in pom.xml**

<!-- https://mvnrepository.com/artifact/org.springframework/spring-orm -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-orm</artifactId>

<version>5.3.14</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.hibernate/hibernate-core -->

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

<version>5.3.14.Final</version>

</dependency>

**Step3)** update project

**Step4)**create spring-config.xml file in src/main/resources for adding spring and hibernate configuration

<bean

class=*"org.springframework.jdbc.datasource.DriverManagerDataSource"*

name=*"ds"*>

<property name=*"driverClassName"*

value=*"com.mysql.cj.jdbc.Driver"*></property>

<property name=*"url"*

value=*"jdbc:mysql://localhost:3306/springjdbc"*></property>

<property name=*"username"* value=*"root"*></property>

<property name=*"password"* value=*"sairam"*></propert>

</bean>

<bean

class=*"org.springframework.orm.hibernate5.LocalSessionFactoryBean"*

name=*"localfactory"*>

<property name=*"dataSource"* ref=*"ds"*></property>

<property name=*"hibernateProperties"*>

<props>

<prop key=*"hibernate.dialect"*>org.hibernate.dialect.MySQL57Dialect</prop>

<prop key=*"hibernate.show\_sql"*>true</prop>

<prop key=*"hibernate.hbm2ddl.auto"*>update</prop>

</props>

</property>

<property name=*"annotatedClasses"*>

<list>

<value>com.app.pojo.Student</value>

</list>

</property>

</bean>

<bean class=*"org.springframework.orm.hibernate5.HibernateTemplate"*

**name=*"hibernateTemp"*>**

**<property name=*"sessionFactory"*>**

**<ref bean=*"localfactory"*></ref>**

**</property>**

**</bean>**

<bean class=*"com.app.Dao.DaoClass"* name=*"daoRef"*>

<property name=*"hibernateTemp"*>

<ref bean=*"hibernateTemp"*></ref>

</property>

</bean>