**Spring Data JPA - Quick Example**

1.Navigate to <https://start.spring.io>

2.Set **Group** to com.cognizant and Artifact to orm-learn.

3. Add a short description: Demo project for Spring Data JPA and Hibernate.

4. Add Spring Boot DevTools for hot-reloading support.

5. Include Spring Data JPA for ORM features.

6. Choose MySQL Driver to connect with your MySQL database.

7. Click Generate to download the ZIP file, then extract it into your Eclipse workspace directory.

8. Open Eclipse, go to File > Import > Maven > Existing Maven Projects.

9. Browse to the folder where you extracted orm-learn, select it, and finish the import.

10. Launch MySQL Workbench or use a terminal.

11. Run the command to create your database: CREATE DATABASE ormlearn;

12. Create table and insert data:

use ormlearn;

CREATE TABLE country(

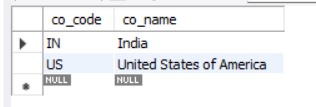
co\_code VARCHAR(2) PRIMARY KEY,

co\_name VARCHAR(50)

);

INSERT INTO country VALUES ('IN', 'India');

INSERT INTO country VALUES ('US', 'United States of America');

select \* from country;

**src/main/resources/application.properties:**

logging.level.org.springframework=info

logging.level.com.cognizant=debug

logging.level.org.hibernate.SQL=trace

logging.level.org.hibernate.type.descriptor.sql=trace

logging.pattern.console=%d{dd-MM-yy} %d{HH:mm:ss.SSS} %-20.20thread %5p %-25.25logger{25} %25M %4L %m%n

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

spring.datasource.url=jdbc:mysql://localhost:3306/ormlearn

spring.datasource.username=root

spring.datasource.password=root

spring.jpa.hibernate.ddl-auto=validate

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect

**Country.java:**

package com.cognizant.orm\_learn.model;

import jakarta.persistence.Entity;

import jakarta.persistence.Id;

import jakarta.persistence.Table;

import jakarta.persistence.Column;

@Entity

@Table(name="country")

public class Country{

@Id

@Column(name="co\_code")

private String code;

@Column(name="co\_name")

private String name;

public String getCode(){

return code;

}

public void setCode(String code){

this.code=code;

}

public String getName(){

return name;

}

public void setName(String name){

this.name=name;

}

@Override

public String toString(){

return "Country [code="+code+", name="+name+"]";

}

}

**CountryRepository.java:**

package com.cognizant.orm\_learn.repository;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

import com.cognizant.orm\_learn.model.Country;

@Repository

public interface CountryRepository extends JpaRepository<Country,String>{

}

**CountryService.java:**

package com.cognizant.orm\_learn.service;

import java.util.List;

import jakarta.transaction.Transactional;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import com.cognizant.orm\_learn.model.Country;

import com.cognizant.orm\_learn.repository.CountryRepository;

@Service

public class CountryService {

@Autowired

private CountryRepository countryRepository;

@Transactional

public List<Country> getAllCountries(){

return countryRepository.findAll();

}

}

**OrmLearnApplication.java:**

package com.cognizant.orm\_learn;

import java.util.List;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

import com.cognizant.orm\_learn.model.Country;

import com.cognizant.orm\_learn.service.CountryService;

@SpringBootApplication

public class OrmLearnApplication{

private static final Logger LOGGER=LoggerFactory.getLogger(OrmLearnApplication.class);

private static CountryService countryService;

public static void main(String[] args){

ApplicationContext context=SpringApplication.run(OrmLearnApplication.class,args);

countryService=context.getBean(CountryService.class);

testGetAllCountries();

}

private static void testGetAllCountries() {

LOGGER.info("Start");

List<Country> countries = countryService.getAllCountries();

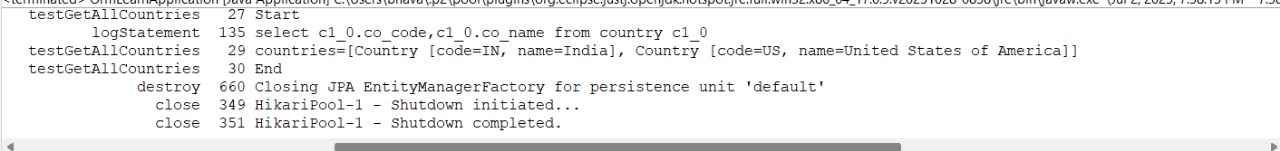
LOGGER.debug("countries={}", countries);

LOGGER.info("End");

}

}

**Output:**



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

**Java Persistence API (JPA)**

* JSR 338 Specification for persisting, reading and managing data from Java objects
* Does not contain concrete implementation of the specification
* Hibernate is one of the implementation of JPA

**Hibernate**

* ORM Tool that implements JPA

**Spring Data JPA**

* Does not have JPA implementation, but reduces boiler plate code
* This is another level of abstraction over JPA implementation provider like Hibernate
* Manages transactions

**MySql:**

CREATE DATABASE company\_info;

USE company\_info;

CREATE TABLE employee (

emp\_id INT PRIMARY KEY AUTO\_INCREMENT,

full\_name VARCHAR(150),

job\_title VARCHAR(100),

email VARCHAR(100) UNIQUE,

hire\_date DATE,

salary DECIMAL(10, 2)

);

**Hibernate Approach:**

Create a new Maven project:

* File → New → Maven Project → Choose quickstart archetype.

**pom.xml:**

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>HibernateDemo</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.hibernate.orm</groupId>

<artifactId>hibernate-core</artifactId>

<version>7.0.0.Final</version>

</dependency>

<dependency>

<groupId>com.mysql</groupId>

<artifactId>mysql-connector-j</artifactId>

<version>8.3.0</version>

</dependency>

<dependency>

<groupId>jakarta.platform</groupId>

<artifactId>jakarta.jakartaee-api</artifactId>

<version>10.0.0</version>

<scope>provided</scope>

</dependency>

</dependencies>

<build>

<resources>

<resource>

<directory>src/main/resources</directory>

</resource>

</resources>

</build>

</project>

**hibernate.cfg.xml:** (in src/main/resources)

<!DOCTYPE hibernate-configuration PUBLIC

"-//Hibernate/Hibernate Configuration DTD 3.0//EN"

"http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

<session-factory>

<property name="hibernate.connection.driver\_class">com.mysql.cj.jdbc.Driver</property>

<property name="hibernate.connection.url">jdbc:mysql://localhost:3306/company\_info </property>

<property name="hibernate.connection.username">root</property>

<property name="hibernate.connection.password">root</property>

<property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>

<property name="hibernate.hbm2ddl.auto">update</property>

<property name="show\_sql">true</property>

<mapping class="com.example.Employee"/>

</session-factory>

</hibernate-configuration>

**Employee.java:** (in **com.example** package)

package com.example;

import jakarta.persistence.\*;

import java.time.LocalDate;

@Entity

@Table(name = "employee")

public class Employee {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int empId;

private String fullName;

private String jobTitle;

@Column(unique = true)

private String email;

private LocalDate hireDate;

private double salary;

public int getEmpId() {

return empId;

}

public void setEmpId(int empId) {

this.empId = empId;

}

public String getFullName() {

return fullName;

}

public void setFullName(String fullName) {

this.fullName = fullName;

}

public String getJobTitle() {

return jobTitle;

}

public void setJobTitle(String jobTitle) {

this.jobTitle = jobTitle;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public LocalDate getHireDate() {

return hireDate;

}

public void setHireDate(LocalDate hireDate) {

this.hireDate = hireDate;

}

public double getSalary() {

return salary;

}

public void setSalary(double salary) {

this.salary = salary;

}

}

**HibernateExample.java:** (in com.example package)

package com.example;

import org.hibernate.\*;

import org.hibernate.cfg.Configuration;

import java.time.LocalDate;

public class HibernateExample {

public static void main(String[] args) {

// Build the SessionFactory using hibernate.cfg.xml

SessionFactory factory = new Configuration().configure().buildSessionFactory();

Session session = factory.openSession();

Transaction tx = null;

try {

tx = session.beginTransaction();

Employee emp = new Employee();

emp.setFullName("Alice Johnson");

emp.setJobTitle("Software Engineer");

emp.setEmail("alice.johnson@example.com");

emp.setHireDate(LocalDate.of(2023, 6, 15));

emp.setSalary(85000.00);

session.persist(emp);

tx.commit();

} catch (HibernateException e) {

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

e.printStackTrace();

} finally {

session.close();

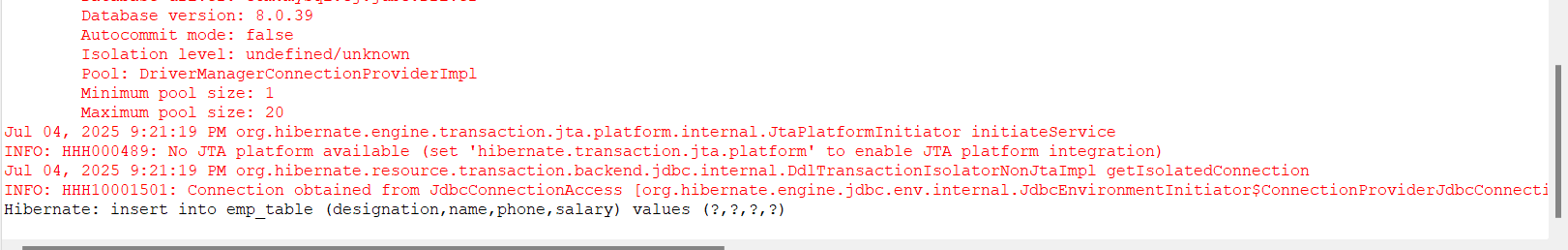
}

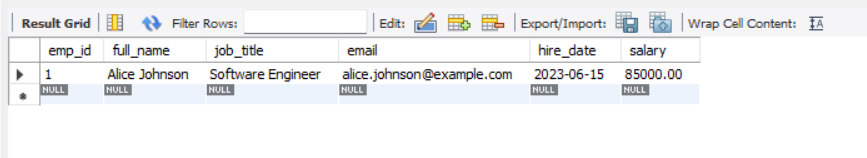
factory.close();

}

}

**Output:**

****

****

**Spring Data JPA Approach:**

1.Go to <https://start.spring.io/>

2.Fill in:

* Group: com.example
* Artifact: company
* Description: Demo project for Spring Data JPA and Hibernate

3.Add dependencies:

* Spring Web
* Spring Data JPA
* MySQL Driver

4.Click Generate, then extract the downloaded ZIP to your Eclipse workspace folder.

5.In Eclipse:  
File > Import > Maven > Existing Maven Projects,Select the extracted orm-learn folder.

Click Finish.

**application.properties:**

spring.datasource.url=jdbc:mysql://localhost:3306/company\_info

spring.datasource.username=root

spring.datasource.password=root

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

**Employee.java: (in com.example.demo.model package)**

package com.example.company.model;

import jakarta.persistence.\*;

import java.time.LocalDate;

@Entity

@Table(name = "employee")

public class Employee {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int empId;

private String fullName;

private String jobTitle;

@Column(unique = true)

private String email;

private LocalDate hireDate;

private double salary;

public int getEmpId() {

return empId;

}

public void setEmpId(int empId) {

this.empId = empId;

}

public String getFullName() {

return fullName;

}

public void setFullName(String fullName) {

this.fullName = fullName;

}

public String getJobTitle() {

return jobTitle;

}

public void setJobTitle(String jobTitle) {

this.jobTitle = jobTitle;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public LocalDate getHireDate() {

return hireDate;

}

public void setHireDate(LocalDate hireDate) {

this.hireDate = hireDate;

}

public double getSalary() {

return salary;

}

public void setSalary(double salary) {

this.salary = salary;

}

}

**EmployeeRepository.java:** (in **com.example.demo.repository** package)

package com.example.company.repository;

import com.example.company.model.Employee;

import org.springframework.data.jpa.repository.JpaRepository;

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

**EmployeeService.java:** (in **com.example.demo.service** package)

package com.example.company.service;

import com.example.company.model.Employee;

import com.example.company.repository.EmployeeRepository;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import jakarta.transaction.Transactional;

@Service

public class EmployeeService {

@Autowired

private EmployeeRepository employeeRepository;

@Transactional

public void addEmployee(Employee emp) {

employeeRepository.save(emp);

}

}

**DemoApplication.java:**

package com.example.company;

import com.example.company.model.Employee;

import com.example.company.service.EmployeeService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import java.time.LocalDate;

@SpringBootApplication

public class DemoApplication implements CommandLineRunner {

@Autowired

private EmployeeService employeeService;

public static void main(String[] args) {

SpringApplication.run(DemoApplication.class, args);

}

@Override

public void run(String... args) {

Employee emp = new Employee();

emp.setFullName("Alice Johnson");

emp.setJobTitle("HR Manager");

emp.setEmail("alice.johnson@company.com");

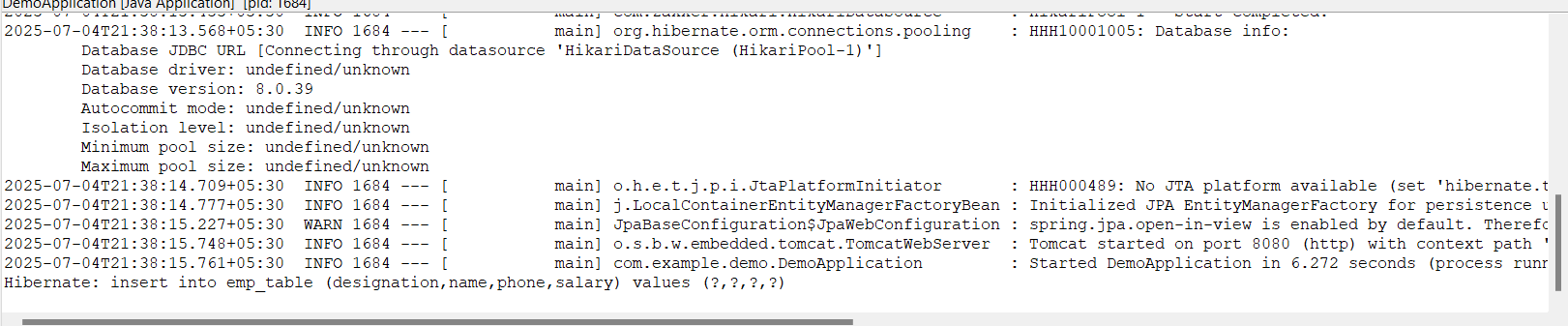
emp.setHireDate(LocalDate.of(2023, 4, 10));

emp.setSalary(60000.00);

employeeService.addEmployee(emp);

}

}

**Output:**

