Spring Data JPA with Hibernate

**Exercise 1 – Spring Data JPA Quick Example**

**1. Created an Eclipse Project using Spring Initializr**

**2. application.properties**

spring.application.name=orm-learn

# Logging Configuration

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

#Database configuration

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

spring.datasource.username=root

spring.datasource.password=Hemanth808@

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

# Hibernate Configuration

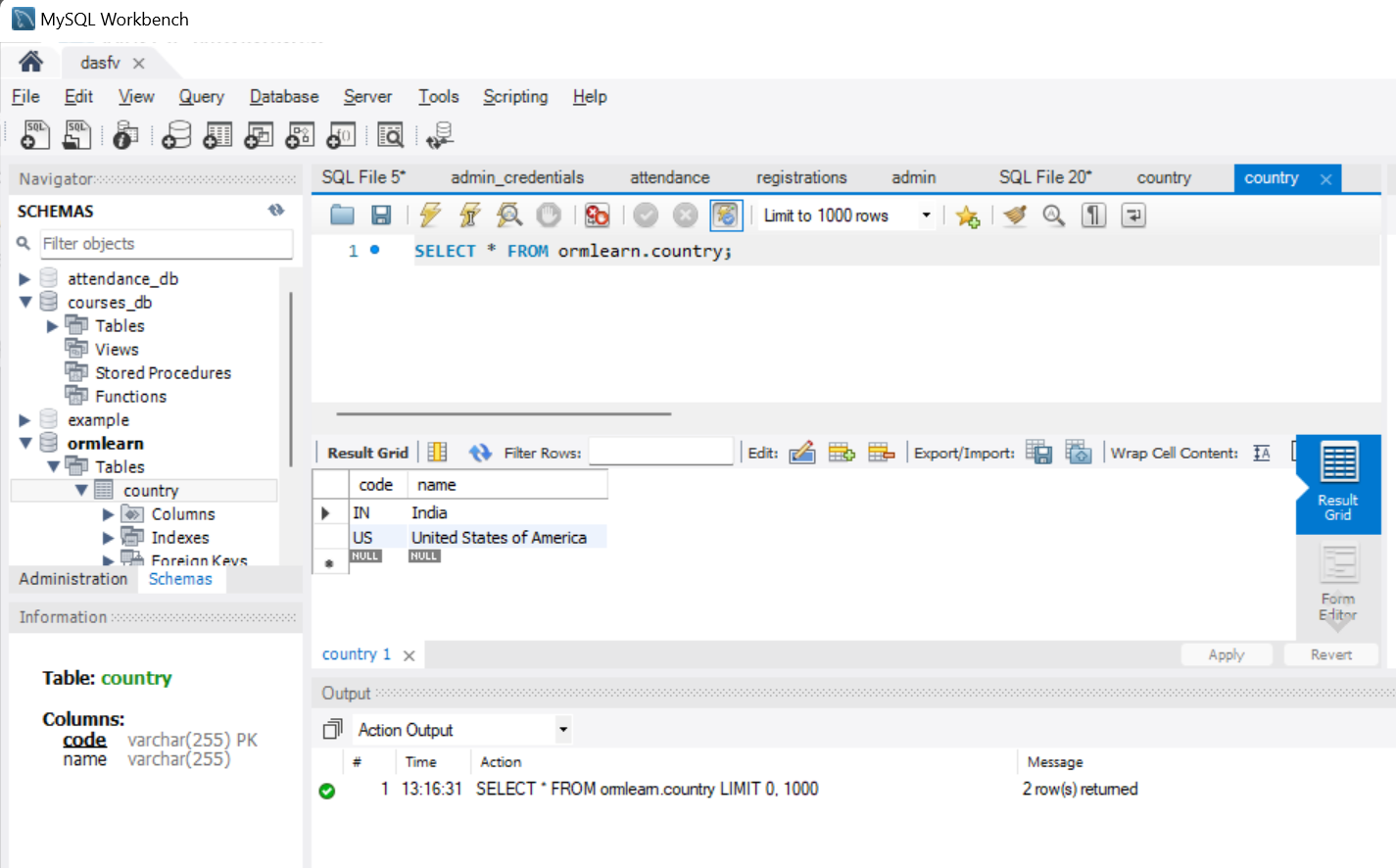
spring.jpa.show-sql=true

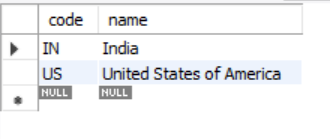
spring.jpa.hibernate.ddl-auto=update

logging.level.org.hibernate=debug

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

**ormlearn Schema:**

****

**Schema:   
**

**3. Persistence Class - com.cognizant.orm-learn.model.Country**

**Code:**

package com.cognizant.orm\_learn.model;

import jakarta.persistence.Column;

import jakarta.persistence.Entity;

import jakarta.persistence.Id;

import jakarta.persistence.Table;

*@Entity*

*@Table*(name = "country")

public class Country {

*@Id*

*@Column*(name = "Code:")

private String Code:;

*@Column*(name = "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 + "]";

}

}

**4. Repository Class - com.cognizant.orm-learn.CountryRepository**

**Code:**

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> {

}

**Service Class - com.cognizant.orm-learn.service.CountryService**

**Code:**

package com.cognizant.orm\_learn.service;

import java.util.List;

import javax.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();

}

}

**5. Testing in OrmLearnApplication.java**

**Code:**

package com.cognizant.orm\_learn;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.boot.autoconfigure.domain.EntityScan;

import org.springframework.context.ApplicationContext;

import org.springframework.data.jpa.repository.config.EnableJpaRepositories;

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

*@SpringBootApplication*(scanBasePackages = "com.cognizant.orm\_learn")

*@EntityScan*(basePackages = "com.cognizant.orm\_learn.model")

*@EnableJpaRepositories*(basePackages = "com.cognizant.orm\_learn.repository")

public class OrmLearnApplication implements CommandLineRunner {

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

private static CountryService *countryService*;

public static void main(String[] args) {

***LOGGER***.info("Inside main");

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

*countryService* = context.getBean(CountryService.class);

*testGetAllCountries*();

}

private static void testGetAllCountries() {

***LOGGER***.info("Start");

***LOGGER***.info("countries={}", *countryService*.getAllCountries());

System.***out***.println(*countryService*.getAllCountries());

***LOGGER***.info("End");

}

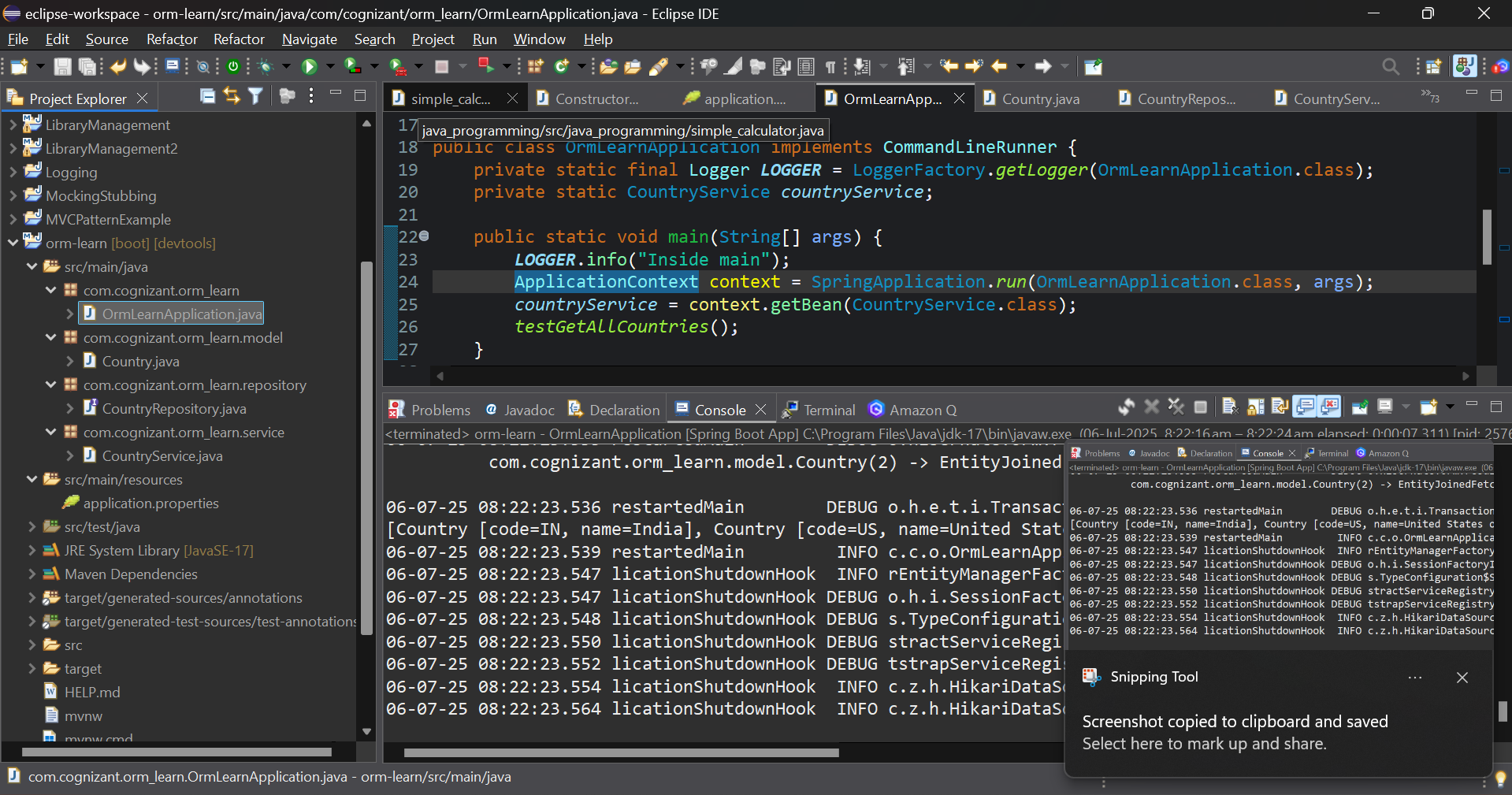
*@Override*

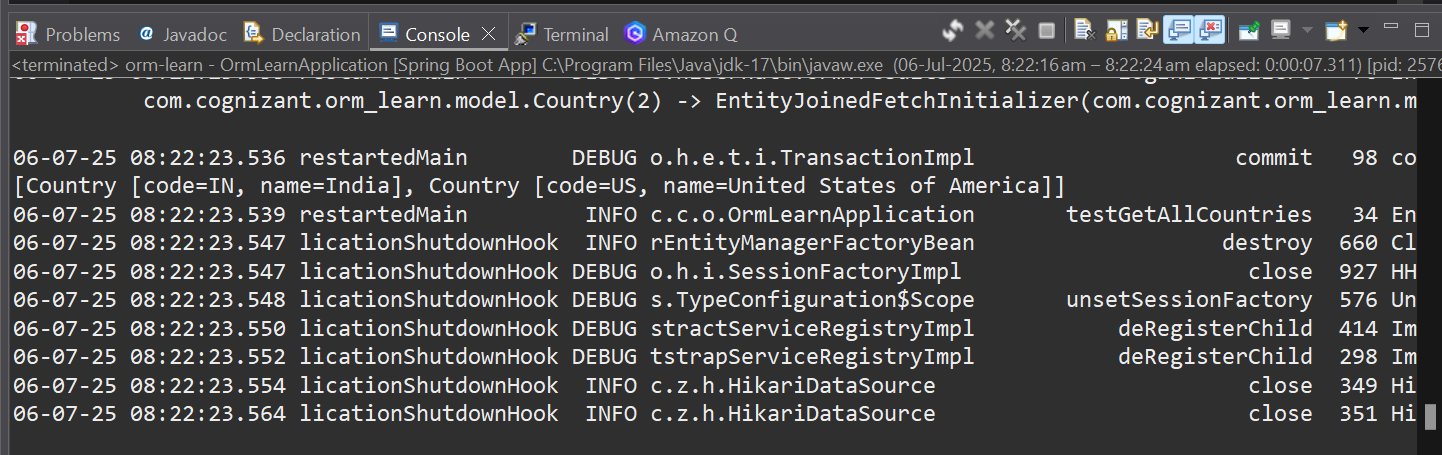
public void run(String... args) throws Exception {

}

}

**Output:**

****



**Exercise 4 – Difference between JPA, Hibernate and Spring JPA**

**JPA (Java Persistence API):**

A specification for managing relational data in Java using annotations. It defines interfaces but has no implementation. Requires a provider like Hibernate to function.

**Hibernate:**

A popular ORM tool that implements the JPA specification. It maps Java objects to database tables and generates SQL automatically. Handles transactions, caching, and lazy loading.

**Spring Data JPA:**

A Spring framework abstraction over JPA and its provider (e.g., Hibernate). It simplifies database access by auto-generating repository code. Reduces boilerplate and manages transactions seamlessly.

**Project Code**

**1. Employee.java – Entity Class (JPA)**

java

CopyEdit

package com.example.ormlearn.model;

import jakarta.persistence.\*;

@Entity

public class Employee {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Integer id;

private String name;

private String department;

// Getters and Setters

}

**2. EmployeeRepository.java – Spring Data JPA**

java

CopyEdit

package com.example.ormlearn.repository;

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

import com.example.ormlearn.model.Employee;

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

**3. EmployeeService.java – Spring + Spring Data JPA**

java

package com.example.ormlearn.service;

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

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

import com.example.ormlearn.model.Employee;

import com.example.ormlearn.repository.EmployeeRepository;

@Service

public class EmployeeService {

@Autowired

private EmployeeRepository employeeRepository;

@Transactional

public void addEmployee(Employee employee) {

employeeRepository.save(employee);

}

}

**4. OrmLearnEmployeeApplication.java – Main Class (Spring Boot)**

java

CopyEdit

package com.example.ormlearn;

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

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import com.example.ormlearn.model.Employee;

import com.example.ormlearn.service.EmployeeService;

@SpringBootApplication

public class OrmLearnApplication implements CommandLineRunner {

@Autowired

private EmployeeService employeeService;

public static void main(String[] args) {

SpringApplication.run(OrmLearnApplication.class, args);

}

@Override

public void run(String... args) {

Employee emp = new Employee();

emp.setName("Alice");

emp.setDepartment("Finance");

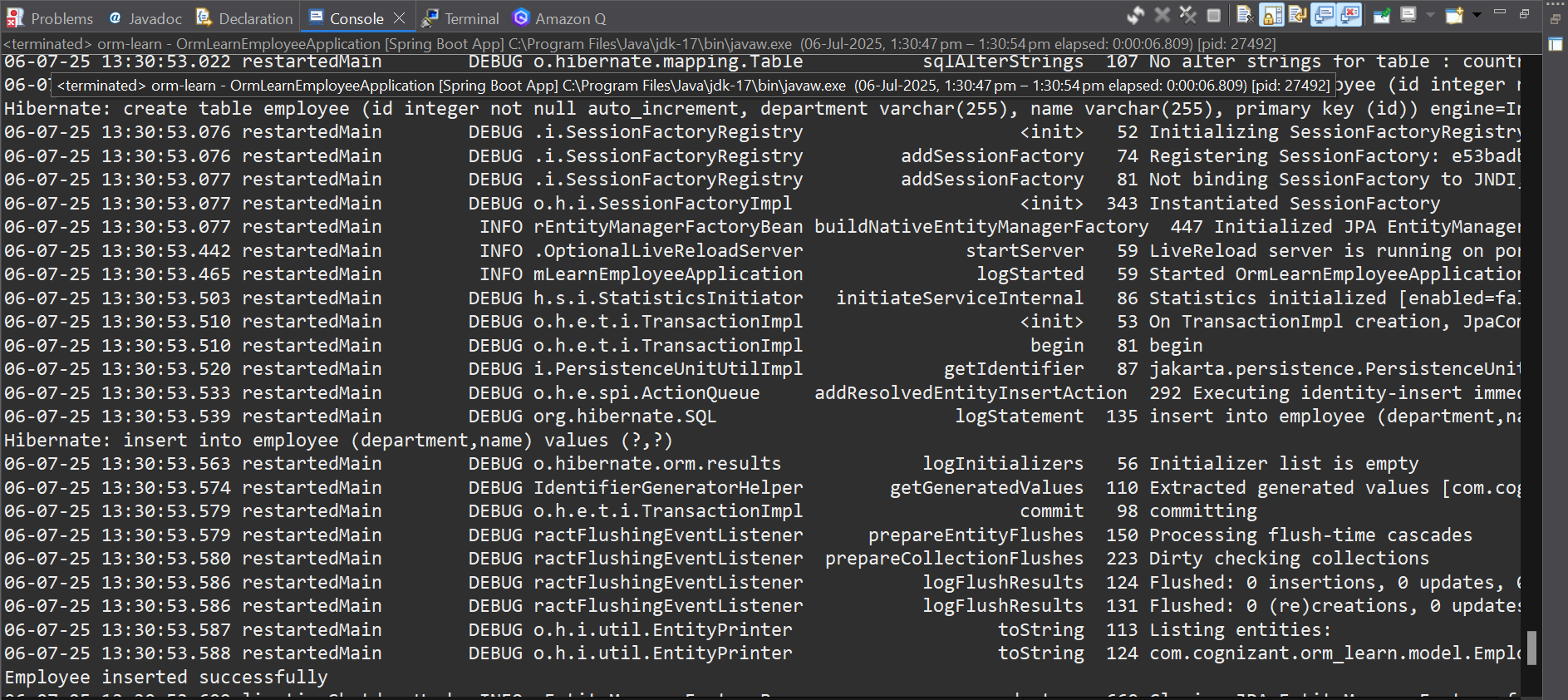
employeeService.addEmployee(emp);

System.out.println("Employee inserted successfully");

}

}

**Output:**

****

**Employee Table Created and an Employee is inserted into the table**

A screenshot of a computer

AI-generated content may be incorrect.