**Cognizant\_Digital Nurture 4.0\_Deep Skilling**

**Products and Frameworks**

**Module 6 - Spring Data JPA with Spring Boot, Hibernate**

**Spring Data JPA - Quick Example**

**Country table creation**

**pom.xml**

<?xml version="1.0" encoding="UTF-8"?>

<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>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>3.5.3</version>

<relativePath/> </parent> <groupId>com.cognizant</groupId>

<artifactId>orm-learn</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>orm-learn</name>

<description>Demo project for Spring Data JPA and Hibernate</description>

<url/>

<licenses>

<license/>

</licenses>

<developers>

<developer/>

</developers>

<scm>

<connection/>

<developerConnection/>

<tag/>

<url/>

</scm>

<properties>

<java.version>17</java.version>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

<optional>true</optional>

</dependency>

<dependency>

<groupId>com.mysql</groupId>

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

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-tx</artifactId>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

**application.properties**

spring.application.name=orm-learn

# Logging

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

# DB Configuration

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=R@@pine.?4021

# Hibernate

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.Column;

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 + "]";

}

}

**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 org.springframework.transaction.annotation.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);

LOGGER.info("Inside main");

testGetAllCountries();

}

private static void testGetAllCountries() {

LOGGER.info("Start");

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

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

LOGGER.info("End");

}

}

**Output:**

05-07-25 23:08:56.872 restartedMain INFO c.c.o.OrmLearnApplication logStarted 59 **Started OrmLearnApplication in 4.434 seconds (process running for 4.957)**

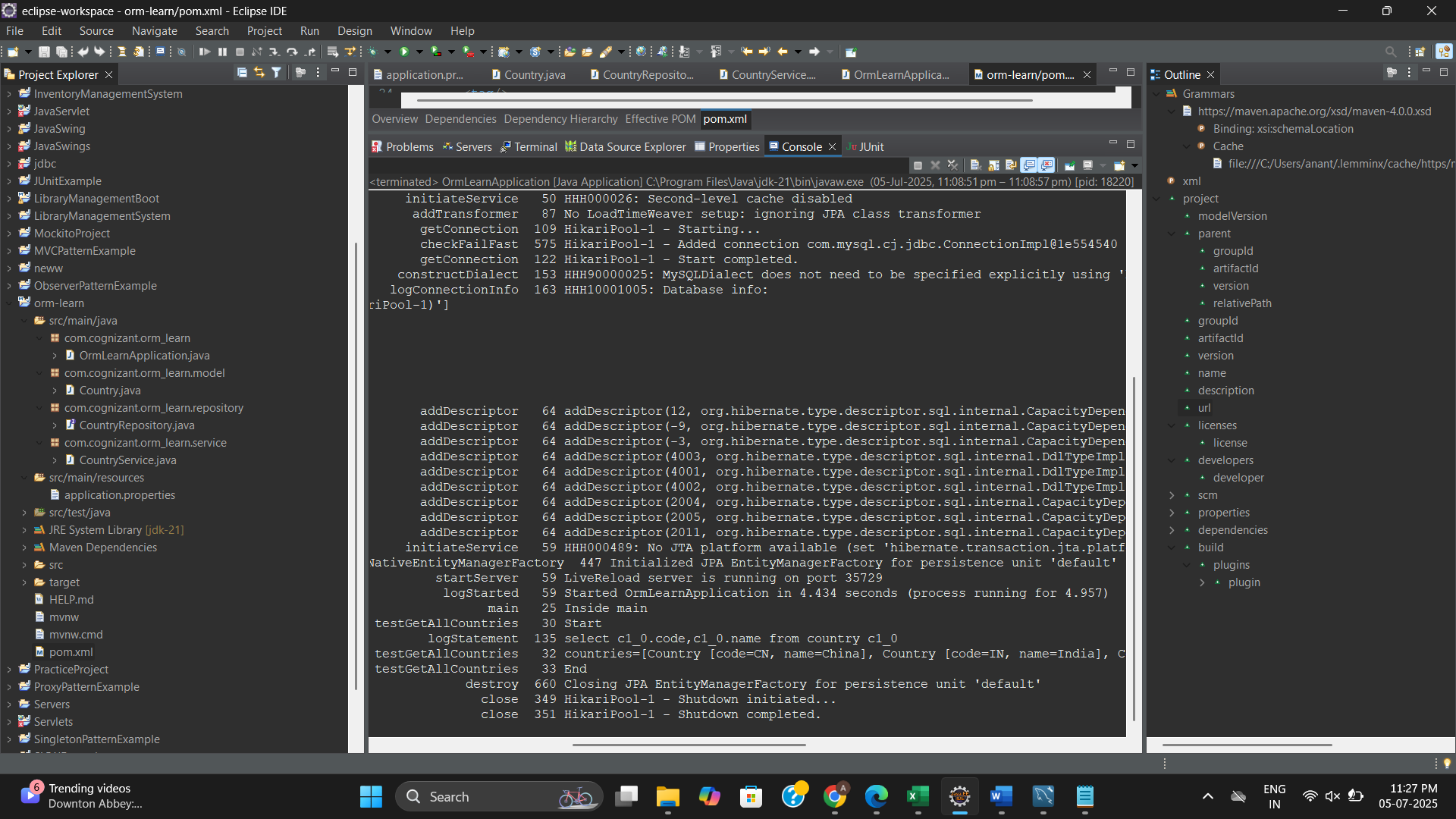
05-07-25 23:08:56.877 restartedMain INFO c.c.o.OrmLearnApplication main 25 **Inside main**

05-07-25 23:08:56.877 restartedMain INFO c.c.o.OrmLearnApplication **testGetAllCountries 30 Start**

05-07-25 23:08:57.068 restartedMain DEBUG org.hibernate.SQL logStatement 135 **select c1\_0.code,c1\_0.name from country c1\_0**

05-07-25 23:08:57.123 restartedMain DEBUG c.c.o.OrmLearnApplication testGetAllCountries **32 countries=[Country [code=CN, name=China], Country [code=IN, name=India], Country [code=US, name=United States of America]]**

05-07-25 23:08:57.123 restartedMain INFO c.c.o.OrmLearnApplication **testGetAllCountries 33 End**

****

**Find a country based on country code**

**application.properties**

spring.application.name=orm-learn

# Logging

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

# DB Configuration

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=R@@pine.?4021

# Hibernate

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;

*@Entity*

*@Table*(name = "country")

public class Country {

*@Id*

private String code;

private String name;

public Country() {}

public Country(String code, String name) {

this.code = code;

this.name = name;

}

// Getters and Setters

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;

}

// toString()

*@Override*

public String toString() {

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

}

}

**CountryRepository.java**

package com.cognizant.orm\_learn.repository;

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

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

public interface CountryRepository extends JpaRepository<Country, String> {

}

**CountryService.java**

package com.cognizant.orm\_learn.service;

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

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

import com.cognizant.orm\_learn.service.exception.CountryNotFoundException;

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

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

import java.util.List;

import java.util.Optional; // Import Optional

*@Service*

public class CountryService {

*@Autowired*

private CountryRepository countryRepository;

*@Transactional*

public List<Country> getAllCountries() {

return countryRepository.findAll();

}

*@Transactional*

public Country findCountryByCode(String countryCode) throws CountryNotFoundException {

Optional<Country> result = countryRepository.findById(countryCode);

if (!result.isPresent()) {

throw new CountryNotFoundException("Country not found for code: " + countryCode);

}

return result.get();

}

}

**CountryNotFoundException.java**

package com.cognizant.orm\_learn.service.exception;

public class CountryNotFoundException extends Exception {

public CountryNotFoundException(String message) {

super(message);

}

}

**OrmLearnApplication.java**

package com.cognizant.orm\_learn;

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

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

import com.cognizant.orm\_learn.service.exception.CountryNotFoundException;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

*@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);

*getCountryByCodeTest*();

}

private static void getCountryByCodeTest() {

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

try {

Country country = *countryService*.findCountryByCode("IN");

***LOGGER***.debug("Country: {}", country);

} catch (CountryNotFoundException e) {

***LOGGER***.error("Exception: {}", e.getMessage());

}

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

}

}

**Output:**

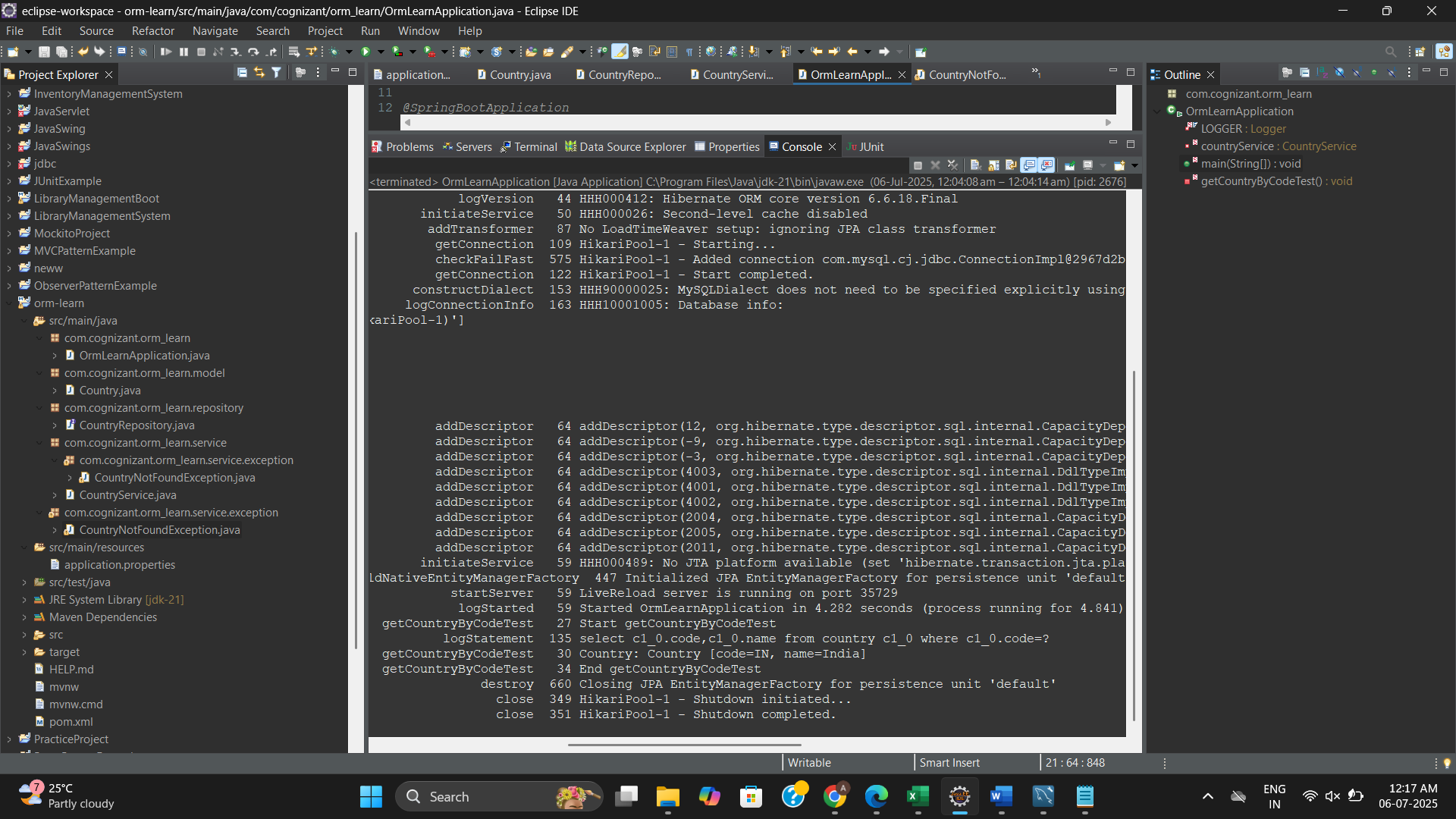
06-07-25 00:04:13.341 restartedMain INFO c.c.o.OrmLearnApplication logStarted 59 **Started OrmLearnApplication in 4.282 seconds (process running for 4.841)**

06-07-25 00:04:13.347 restartedMain INFO c.c.o.OrmLearnApplication **getCountryByCodeTest 27 Start getCountryByCodeTest**

06-07-25 00:04:13.448 restartedMain DEBUG org.hibernate.SQL logStatement 135 **select c1\_0.code,c1\_0.name from country c1\_0 where c1\_0.code=?**

06-07-25 00:04:13.502 restartedMain DEBUG c.c.o.OrmLearnApplication **getCountryByCodeTest 30 Country: Country [code=IN, name=India]**

06-07-25 00:04:13.502 restartedMain INFO c.c.o.OrmLearnApplication **getCountryByCodeTest 34 End getCountryByCodeTest**



**Add a New Country**

**pom.xml**

<?xml version="1.0" encoding="UTF-8"?>

<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>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>3.5.3</version>

<relativePath/> </parent> <groupId>com.cognizant</groupId>

<artifactId>orm-learn</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>orm-learn</name>

<description>Demo project for Spring Data JPA and Hibernate</description>

<url/>

<licenses>

<license/>

</licenses>

<developers>

<developer/>

</developers>

<scm>

<connection/>

<developerConnection/>

<tag/>

<url/>

</scm>

<properties>

<java.version>17</java.version>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

<optional>true</optional>

</dependency>

<dependency>

<groupId>com.mysql</groupId>

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

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-tx</artifactId>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

**application.properties**

spring.application.name=orm-learn

# Logging

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

# DB Configuration

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

spring.datasource.url=jdbc:mysql://localhost:3306/ormlearn?useSSL=false&allowPublicKeyRetrieval=true&serverTimezone=UTC

spring.datasource.username=root

spring.datasource.password=R@@pine.?4021

# Hibernate

spring.jpa.hibernate.ddl-auto=update

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;

*@Entity*

*@Table*(name = "country")

public class Country {

*@Id*

private String code;

private String name;

// Getters and setters

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

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

import org.springframework.stereotype.Repository;

*@Repository*

public interface CountryRepository extends JpaRepository<Country, String> {

}

**CountryService.java**

package com.cognizant.orm\_learn.service;

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

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

import com.cognizant.orm\_learn.service.exception.CountryNotFoundException;

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

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

import java.util.List;

import java.util.Optional;

*@Service*

public class CountryService {

*@Autowired*

private CountryRepository countryRepository;

*@Transactional*

public List<Country> getAllCountries() {

return countryRepository.findAll();

}

*@Transactional*

public Country findCountryByCode(String countryCode) throws CountryNotFoundException {

Optional<Country> result = countryRepository.findById(countryCode);

if (!result.isPresent()) {

throw new CountryNotFoundException("Country not found for code: " + countryCode);

}

return result.get();

}

*@Transactional*

public void addCountry(Country country) {

countryRepository.save(country);

}

}

**CountryNotFoundException.java**

package com.cognizant.orm\_learn.service.exception;

public class CountryNotFoundException extends Exception {

public CountryNotFoundException(String message) {

super(message);

}

}

**OrmLearnApplication.java**

package com.cognizant.orm\_learn;

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

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

import com.cognizant.orm\_learn.service.exception.CountryNotFoundException;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

*@SpringBootApplication*

public class OrmLearnApplication {

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

private static CountryService *countryService*;

public static void main(String[] args) throws CountryNotFoundException {

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

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

*getCountryByCodeTest*();

*testAddCountry*();

}

private static void getCountryByCodeTest() throws CountryNotFoundException {

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

Country country = *countryService*.findCountryByCode("IN");

***LOGGER***.debug("Country: {}", country);

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

}

private static void testAddCountry() {

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

Country newCountry = new Country();

newCountry.setCode("ZZ");

newCountry.setName("Zootopia");

*countryService*.addCountry(newCountry);

try {

Country fetched = *countryService*.findCountryByCode("ZZ");

***LOGGER***.debug("Added Country: {}", fetched);

} catch (CountryNotFoundException e) {

***LOGGER***.error("Country not found after adding: {}", e.getMessage());

}

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

}

}

**Output:**

06-07-25 16:00:27.945 restartedMain INFO c.c.o.OrmLearnApplication logStarted 59 Started OrmLearnApplication in 3.6 seconds (process running for 4.089)

06-07-25 16:00:27.950 restartedMain INFO c.c.o.OrmLearnApplication getCountryByCodeTest 27 **Start getCountryByCodeTest**

06-07-25 16:00:28.010 restartedMain DEBUG org.hibernate.SQL logStatement 135 **select c1\_0.code,c1\_0.name from country c1\_0 where c1\_0.code=?**

06-07-25 16:00:28.041 restartedMain DEBUG c.c.o.OrmLearnApplication **getCountryByCodeTest 29 Country: Country [code=IN, name=India]**

06-07-25 16:00:28.041 restartedMain INFO c.c.o.OrmLearnApplication getCountryByCodeTest 30 **End getCountryByCodeTest**

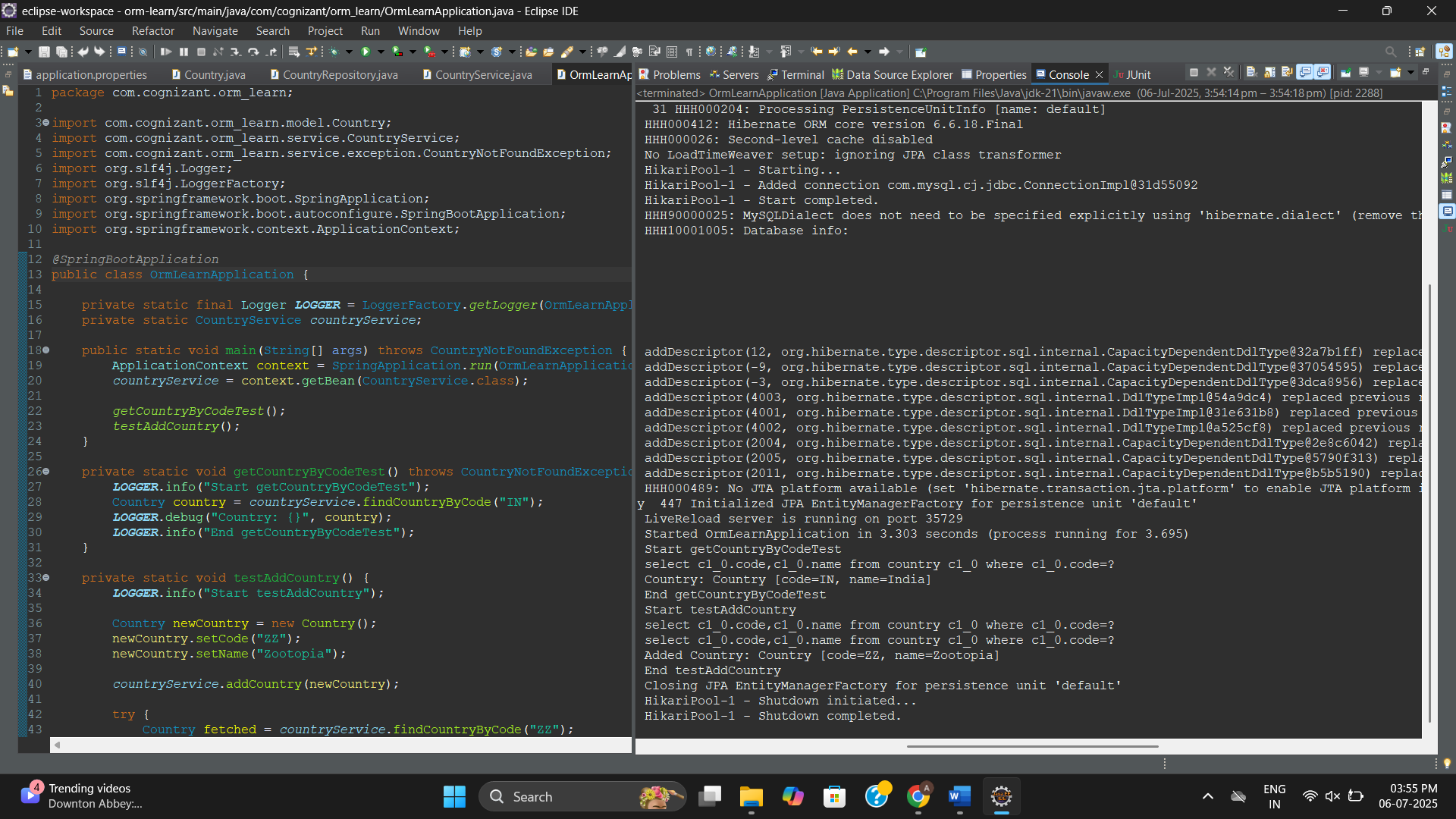
06-07-25 16:00:28.041 restartedMain INFO c.c.o.OrmLearnApplication testAddCountry 34 **Start testAddCountry**

06-07-25 16:00:28.050 restartedMain DEBUG org.hibernate.SQL logStatement 135 **select c1\_0.code,c1\_0.name from country c1\_0 where c1\_0.code=?**

06-07-25 16:00:28.053 restartedMain DEBUG org.hibernate.SQL logStatement 135 **select c1\_0.code,c1\_0.name from country c1\_0 where c1\_0.code=?**

06-07-25 16:00:28.055 restartedMain DEBUG c.c.o.OrmLearnApplication **testAddCountry 44 Added Country: Country [code=ZZ, name=Zootopia]**

06-07-25 16:00:28.055 restartedMain INFO c.c.o.OrmLearnApplication testAddCountry 49 **End testAddCountry**

****

**Demonstrate implementation of Query Methods feature of Spring Data JPA**

**application.properties**

spring.datasource.url=jdbc:mysql://localhost:3306/ormlearn?allowPublicKeyRetrieval=true&useSSL=false&serverTimezone=UTC

spring.datasource.username=root

spring.datasource.password=R@@pine.?4021

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

spring.jpa.hibernate.ddl-auto=update

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

spring.jpa.show-sql=true

spring.jpa.properties.hibernate.format\_sql=true

**pom.xml**

<?xml version="1.0" encoding="UTF-8"?>

<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>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>3.5.3</version> <relativePath/> </parent>

<groupId>com.cognizant</groupId>

<artifactId>orm-learn</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>orm-learn</name>

<description>Demo project for Spring Data JPA and Hibernate</description>

<properties>

<java.version>17</java.version> </properties>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

<optional>true</optional>

</dependency>

<dependency>

<groupId>com.mysql</groupId>

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

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-tx</artifactId>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

**Country.java**

package com.cognizant.orm\_learn.model;

import jakarta.persistence.\*;

import java.util.Date;

*@Entity*

*@Table*(name = "country")

public class Country {

*@Id*

*@Column*(name = "code")

private String code;

*@Column*(name = "name")

private String name;

*@Column*(name = "population")

private Long population;

*@Column*(name = "independence\_date")

*@Temporal*(*TemporalType*.***DATE***)

private Date independenceDate;

// Getters and setters

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;

}

public long getPopulation() {

return population;

}

public void setPopulation(long population) {

this.population = population;

}

public Date getIndependenceDate() {

return independenceDate;

}

public void setIndependenceDate(Date independenceDate) {

this.independenceDate = independenceDate;

}

*@Override*

public String toString() {

return "Country{" +

"code='" + code + '\'' +

", name='" + name + '\'' +

", population=" + population +

", independenceDate=" + independenceDate +

'}';

}

}

**CountryRepository.java**

package com.cognizant.orm\_learn.repository;

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

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

import org.springframework.stereotype.Repository;

import java.util.Date;

import java.util.List;

*@Repository*

public interface CountryRepository extends JpaRepository<Country, String> {

List<Country> findByNameContainingIgnoreCase(String name);

List<Country> findByNameStartingWith(String prefix);

List<Country> findAllByOrderByNameAsc();

List<Country> findByIndependenceDateBetween(Date start, Date end);

List<Country> findByPopulationGreaterThan(long population);

List<Country> findTop3ByOrderByPopulationDesc();

}

**CountryService.java**

package com.cognizant.orm\_learn.service;

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

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

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

import org.springframework.stereotype.Service;

import java.util.Date;

import java.util.List;

*@Service*

public class CountryService {

*@Autowired*

private CountryRepository countryRepository;

public List<Country> searchByNameContains(String text) {

return countryRepository.findByNameContainingIgnoreCase(text);

}

public List<Country> searchByStartingText(String prefix) {

return countryRepository.findByNameStartingWith(prefix);

}

public List<Country> sortByName() {

return countryRepository.findAllByOrderByNameAsc();

}

public List<Country> findByIndependenceDateBetween(Date start, Date end) {

return countryRepository.findByIndependenceDateBetween(start, end);

}

public List<Country> populationGreaterThan(long pop) {

return countryRepository.findByPopulationGreaterThan(pop);

}

public List<Country> top3MostPopulated() {

return countryRepository.findTop3ByOrderByPopulationDesc();

}

}

**OrmLearnApplication.java**

package com.cognizant.orm\_learn;

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

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

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.text.SimpleDateFormat;

import java.util.Date;

*@SpringBootApplication*

public class OrmLearnApplication implements CommandLineRunner {

*@Autowired*

private CountryService countryService;

public static void main(String[] args) {

SpringApplication.*run*(OrmLearnApplication.class, args);

}

*@Override*

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

System.***out***.println("==== Contains 'an' ====");

countryService.searchByNameContains("an").forEach(System.***out***::println);

System.***out***.println("==== Starts with 'In' ====");

countryService.searchByStartingText("In").forEach(System.***out***::println);

System.***out***.println("==== Sorted by Name ====");

countryService.sortByName().forEach(System.***out***::println);

System.***out***.println("==== Independence between 1945 and 1960 ====");

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");

Date start = sdf.parse("1945-01-01");

Date end = sdf.parse("1960-12-31");

countryService.findByIndependenceDateBetween(start, end).forEach(System.***out***::println);

System.***out***.println("==== Population > 10,000,000 ====");

countryService.populationGreaterThan(10000000).forEach(System.***out***::println);

System.***out***.println("==== Top 3 populated countries ====");

countryService.top3MostPopulated().forEach(System.***out***::println);

}

}

**Output:**

2025-07-06T16:38:13.961+05:30 INFO 8352 --- [ restartedMain] c.c.orm\_learn.OrmLearnApplication : Started OrmLearnApplication in 3.341 seconds (process running for 3.713)

==== Contains 'an' ====

Hibernate:

select

c1\_0.code,

c1\_0.independence\_date,

c1\_0.name,

c1\_0.population

from

country c1\_0

where

upper(c1\_0.name) like upper(?) escape '\\'

==== Starts with 'In' ====

Hibernate:

select

c1\_0.code,

c1\_0.independence\_date,

c1\_0.name,

c1\_0.population

from

country c1\_0

where

c1\_0.name like ? escape '\\'

Country{code='IN', name='India', population=1380000000, independenceDate=1947-08-15}

==== Sorted by Name ====

Hibernate:

select

c1\_0.code,

c1\_0.independence\_date,

c1\_0.name,

c1\_0.population

from

country c1\_0

order by

c1\_0.name

Country{code='CN', name='China', population=1400000000, independenceDate=1949-10-01}

Country{code='IN', name='India', population=1380000000, independenceDate=1947-08-15}

Country{code='US', name='United States of America', population=330000000, independenceDate=1776-07-04}

Country{code='ZZ', name='Zootopia', population=1000000, independenceDate=2000-01-01}

==== Independence between 1945 and 1960 ====

Hibernate:

select

c1\_0.code,

c1\_0.independence\_date,

c1\_0.name,

c1\_0.population

from

country c1\_0

where

c1\_0.independence\_date between ? and ?

Country{code='CN', name='China', population=1400000000, independenceDate=1949-10-01}

Country{code='IN', name='India', population=1380000000, independenceDate=1947-08-15}

==== Population > 10,000,000 ====

Hibernate:

select

c1\_0.code,

c1\_0.independence\_date,

c1\_0.name,

c1\_0.population

from

country c1\_0

where

c1\_0.population>?

Country{code='CN', name='China', population=1400000000, independenceDate=1949-10-01}

Country{code='IN', name='India', population=1380000000, independenceDate=1947-08-15}

Country{code='US', name='United States of America', population=330000000, independenceDate=1776-07-04}

==== Top 3 populated countries ====

Hibernate:

select

c1\_0.code,

c1\_0.independence\_date,

c1\_0.name,

c1\_0.population

from

country c1\_0

order by

c1\_0.population desc

limit

?

Country{code='CN', name='China', population=1400000000, independenceDate=1949-10-01}

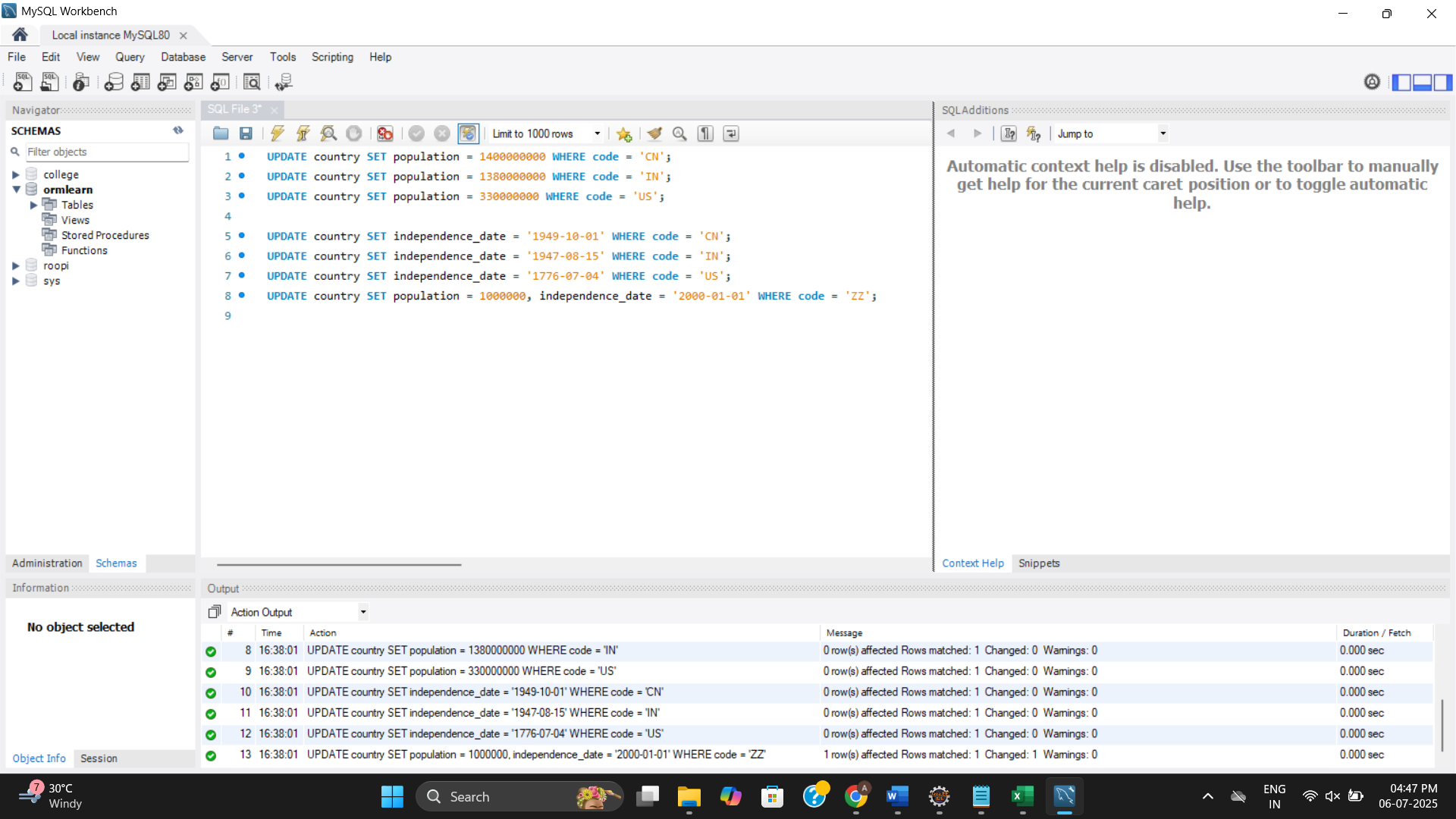
Country{code='IN', name='India', population=1380000000, independenceDate=1947-08-15}

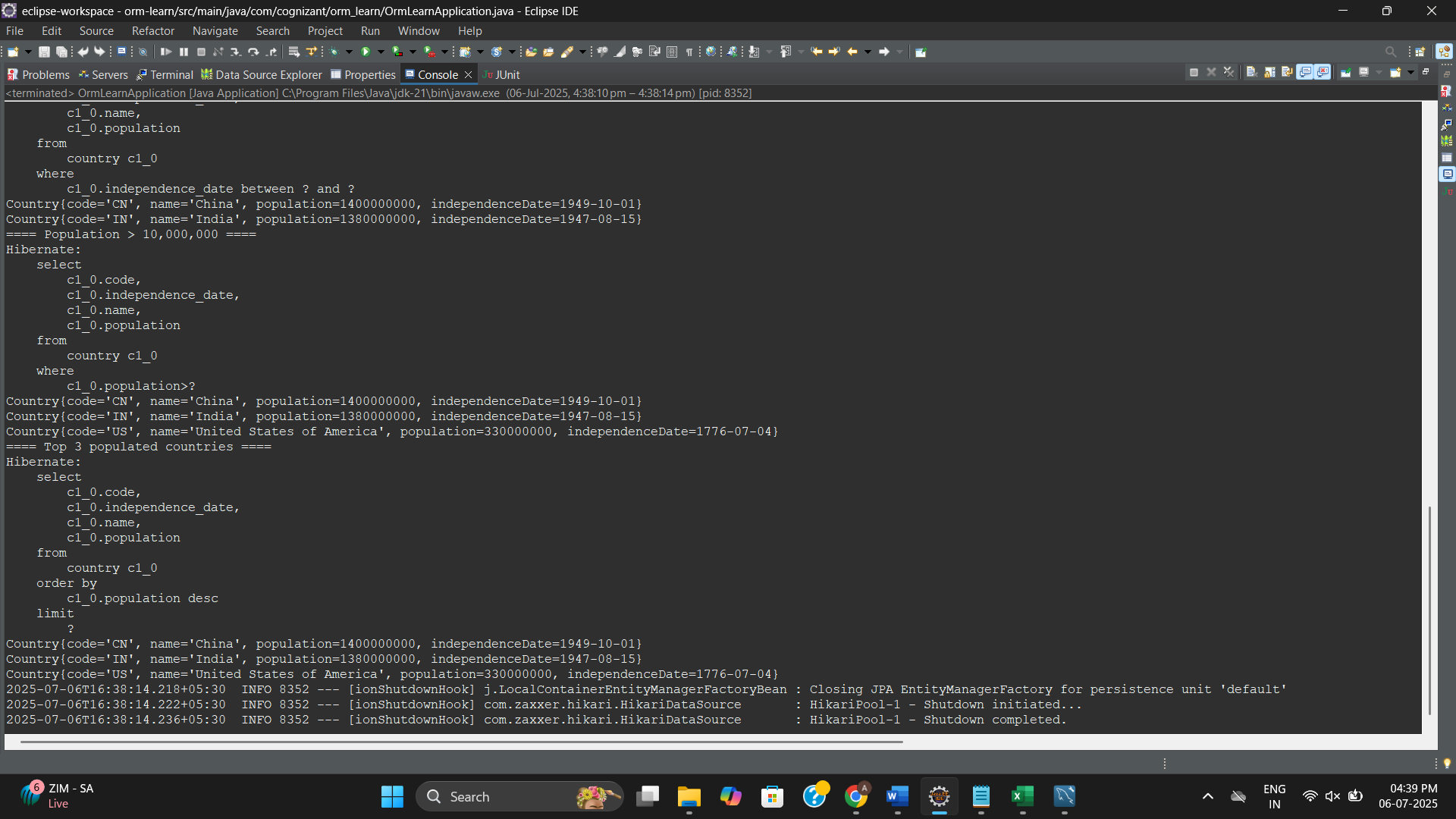
Country{code='US', name='United States of America', population=330000000, independenceDate=1776-07-04}

2025-07-06T16:38:14.218+05:30 INFO 8352 --- [ionShutdownHook] j.LocalContainerEntityManagerFactoryBean : Closing JPA EntityManagerFactory for persistence unit 'default'

2025-07-06T16:38:14.222+05:30 INFO 8352 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Shutdown initiated...

2025-07-06T16:38:14.236+05:30 INFO 8352 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Shutdown completed.





**Demonstrate implementation of O/R Mapping**

**application.properties**

spring.datasource.url=jdbc:mysql://localhost:3306/ormdemo?useSSL=false&serverTimezone=UTC

spring.datasource.username=root

spring.datasource.password=R@@pine.?4021

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

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect

server.port=8082

**Employee.java**

package com.example.ormdemo.model;

import jakarta.persistence.\*;

import java.util.List;

*@Entity*

public class Employee {

*@Id*

*@GeneratedValue*(strategy = *GenerationType*.***IDENTITY***)

private Long id;

private String name;

*@ManyToOne*(fetch = *FetchType*.***EAGER***)

*@JoinColumn*(name = "department\_id")

private Department department;

*@ManyToMany*

*@JoinTable*(

name = "employee\_project",

joinColumns = *@JoinColumn*(name = "employee\_id"),

inverseJoinColumns = *@JoinColumn*(name = "project\_id")

)

private List<Project> projects;

}

**Department.java**

package com.example.ormdemo.model;

import jakarta.persistence.\*;

import java.util.List;

*@Entity*

public class Department {

*@Id*

*@GeneratedValue*(strategy = *GenerationType*.***IDENTITY***)

private Long id;

private String name;

*@OneToMany*(mappedBy = "department", fetch = *FetchType*.***LAZY***, cascade = *CascadeType*.***ALL***)

private List<Employee> employees;

}

**Project.java**

package com.example.ormdemo.model;

import jakarta.persistence.\*;

import java.util.List;

*@Entity*

public class Project {

*@Id*

*@GeneratedValue*(strategy = *GenerationType*.***IDENTITY***)

private Long id;

private String name;

*@ManyToMany*(mappedBy = "projects")

private List<Employee> employees;

}

**EmployeeRepository.java**

package com.example.ormdemo.repository;

import com.example.ormdemo.model.Employee;

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

public interface EmployeeRepository extends JpaRepository<Employee, Long> {

}

**DepartmentRepository.java**

package com.example.ormdemo.repository;

import com.example.ormdemo.model.Department;

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

public interface DepartmentRepository extends JpaRepository<Department, Long> {

}

**ProjectRepository.java**

package com.example.ormdemo.repository;

import com.example.ormdemo.model.Project;

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

public interface ProjectRepository extends JpaRepository<Project, Long> {

}

**OrmLearnApplication.java**

package com.example.ormdemo;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

*@SpringBootApplication*

public class OrmdemoApplication {

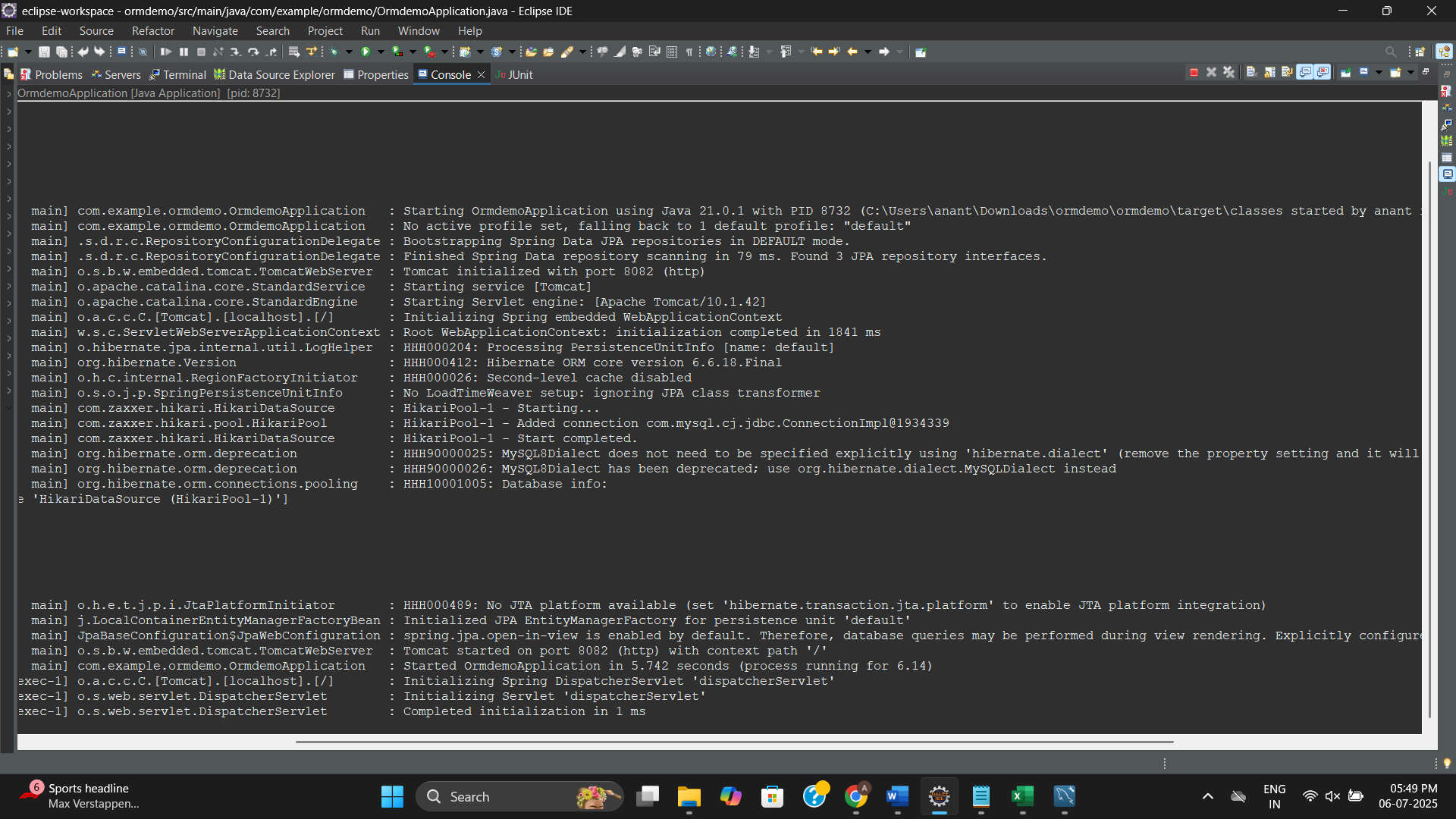
public static void main(String[] args) {

SpringApplication.*run*(OrmdemoApplication.class, args);

}

}

**Output:**

****

**Demonstrate writing Hibernate Query Language and Native Query**

**OrmLearnApplication.java**

package com.example.ormdemo;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

*@SpringBootApplication*

public class OrmdemoApplication {

public static void main(String[] args) {

SpringApplication.*run*(OrmdemoApplication.class, args);

}

}

**Employee.java**

package com.example.ormdemo.model;

import jakarta.persistence.\*;

*@Entity*

*@Table*(name = "employee")

public class Employee {

*@Id*

*@GeneratedValue*(strategy = *GenerationType*.***IDENTITY***)

private Long id;

private String name;

public Employee() {}

public Employee(String name) {

this.name = name;

}

// Getters and setters

public Long getId() { return id; }

public void setId(Long id) { this.id = id; }

public String getName() { return name; }

public void setName(String name) { this.name = name; }

}

**EmployeeController.java**

package com.example.ormdemo.controller;

import com.example.ormdemo.model.Employee;

import com.example.ormdemo.repository.EmployeeRepository;

import jakarta.transaction.Transactional;

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

import org.springframework.web.bind.annotation.\*;

import java.util.List;

*@RestController*

*@RequestMapping*("/employees")

public class EmployeeController {

*@Autowired*

private EmployeeRepository employeeRepository;

*@GetMapping*("/add")

*@Transactional*

public String addSampleEmployees() {

Employee emp1 = new Employee();

emp1.setName("John");

Employee emp2 = new Employee();

emp2.setName("Alice");

employeeRepository.save(emp1);

employeeRepository.save(emp2);

return "Sample employees added.";

}

*@GetMapping*("/hql/{name}")

public List<Employee> findByHQL(*@PathVariable* String name) {

return employeeRepository.findByNameHQL(name);

}

*@GetMapping*("/native/{name}")

public List<Employee> findByNative(*@PathVariable* String name) {

return employeeRepository.findByNameNative(name);

}

}

**EmployeeRepository.java**

package com.example.ormdemo.repository;

import com.example.ormdemo.model.Employee;

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

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

import java.util.List;

public interface EmployeeRepository extends JpaRepository<Employee, Long> {

*@Query*("FROM Employee e WHERE e.name = ?1") // HQL Query

List<Employee> findByNameHQL(String name);

*@Query*(value = "SELECT \* FROM employee WHERE name = ?1", nativeQuery = true) // Native SQL Query

List<Employee> findByNameNative(String name);

}

**application.properties**

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

spring.datasource.username=root

spring.datasource.password=R@@pine.?4021

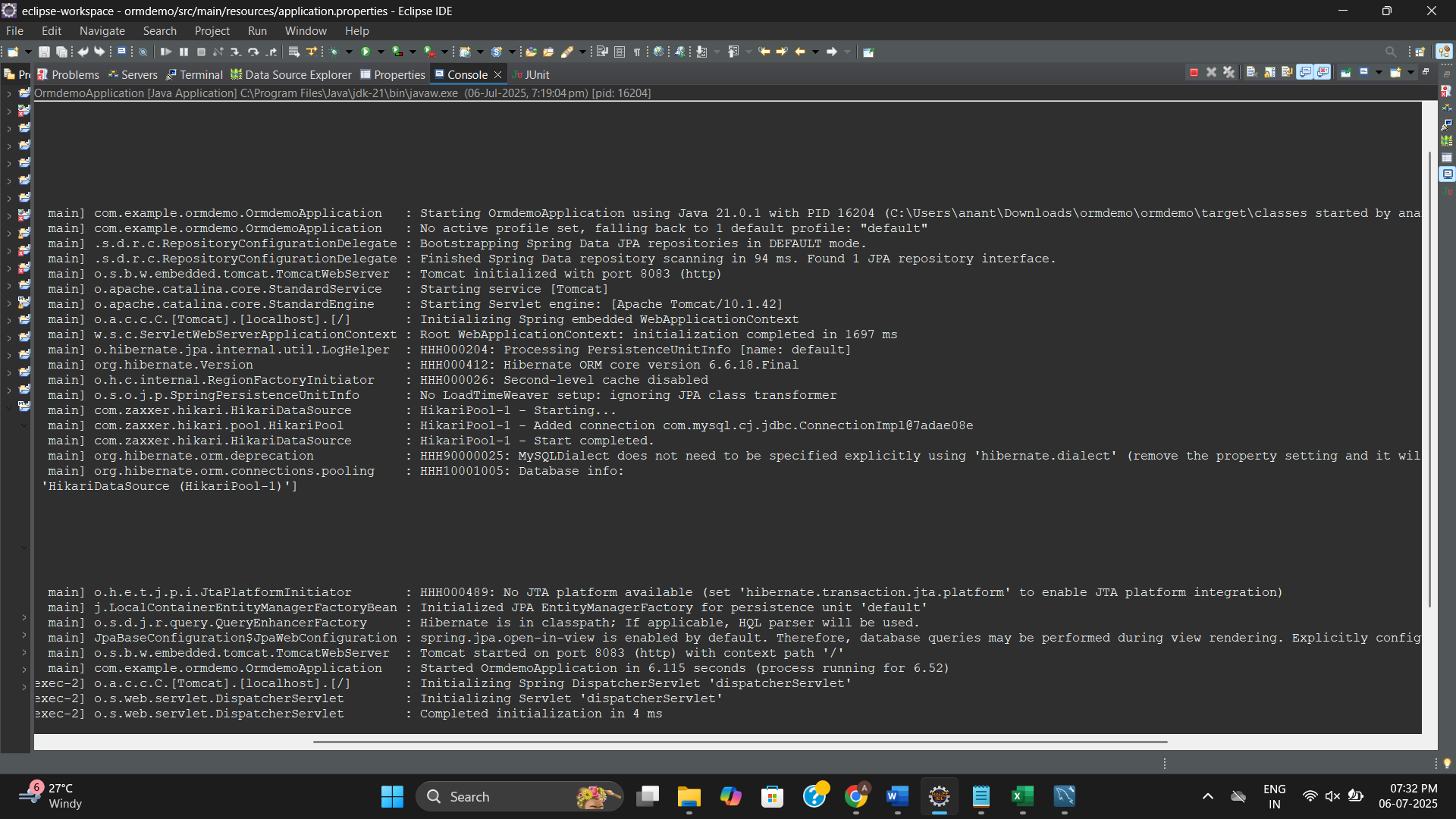
spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

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

server.port=8084

**Output:**

****

