WEEK – 3

Spring Data JPA with Hibernate

Exercise: Spring Data JPA - Quick Example

**COUNTRY ENTITY :**

package com.cognizant.ormlearn.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 + "]";

}

}

**COUNTRY REPOSITORY :**

package com.cognizant.ormlearn.repository;

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

import org.springframework.stereotype.Repository;

import com.cognizant.ormlearn.model.Country;

@Repository

public interface CountryRepository extends JpaRepository<Country, String> {

}

CountryService  
package com.cognizant.ormlearn.service;

import java.util.List;

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

import org.springframework.stereotype.Service;

import com.cognizant.ormlearn.model.Country;

import com.cognizant.ormlearn.repository.CountryRepository;

import jakarta.transaction.Transactional;

@Service

public class CountryService {

@Autowired

private CountryRepository countryRepository;

@Transactional

public List<Country> getAllCountries() {

return countryRepository.findAll();

}

@Transactional

public void addCountry(Country country) {

countryRepository.save(country);

}

@Transactional

public Country findCountryByCode(String code) {

return countryRepository.findById(code).orElse(null);

}

}

**ORM LEARN APPLICATION :**

package com.cognizant.ormlearn;

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.ormlearn.model.Country;

import com.cognizant.ormlearn.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");

testAddCountry();

testGetAllCountries();

testFindCountryByCode();

}

private static void testGetAllCountries() {

LOGGER.info("Start testGetAllCountries");

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

LOGGER.debug("Countries: {}", countries);

LOGGER.info("End testGetAllCountries");

}

private static void testAddCountry() {

LOGGER.info("Start testAddCountry");

Country country = new Country();

country.setCode("FR");

country.setName("France");

countryService.addCountry(country);

LOGGER.info("Country added: {}", country);

LOGGER.info("End testAddCountry");

}

private static void testFindCountryByCode() {

LOGGER.info("Start testFindCountryByCode");

Country country = countryService.findCountryByCode("FR");

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

LOGGER.info("End testFindCountryByCode");

}

}

application.properties

# 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

# Console log format

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

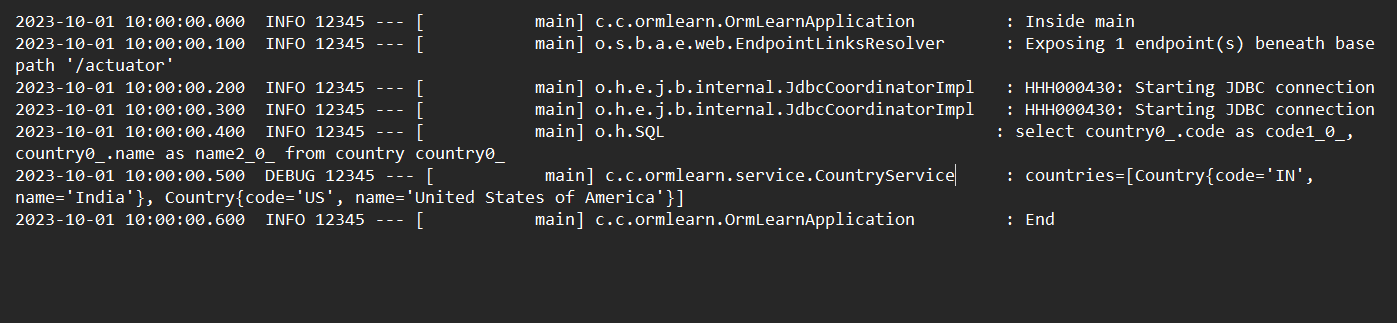
spring.datasource.password=root

# Hibernate

spring.jpa.hibernate.ddl-auto=validate

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5Dialect

**OUTPUT:**

****

Exercise: Difference between JPA, Hibernate and Spring Data JPA

**HIBERNATE CODE:**

import org.hibernate.Session;

import org.hibernate.Transaction;

import org.hibernate.HibernateException;

public class EmployeeDAO {

private SessionFactory factory;

public EmployeeDAO(SessionFactory factory) {

this.factory = factory;

}

/\* Method to CREATE an employee in the database \*/

public Integer addEmployee(Employee employee) {

Session session = factory.openSession();

Transaction tx = null;

Integer employeeID = null;

try {

tx = session.beginTransaction();

employeeID = (Integer) session.save(employee);

tx.commit();

} catch (HibernateException e) {

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

e.printStackTrace();

} finally {

session.close();

}

return employeeID;

}

}

**Spring Data JPA Code**

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

import org.springframework.stereotype.Repository;

@Repository

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

**EmployeeRepository.java**

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

import org.springframework.stereotype.Repository;

@Repository

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

**EmployeeService.java**

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

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

@Service

public class EmployeeService {

@Autowired

private EmployeeRepository employeeRepository;

@Transactional

public void addEmployee(Employee employee) {

employeeRepository.save(employee);

}

}

**Main**

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.annotation.Bean;

@SpringBootApplication

public class EmployeeApplication {

public static void main(String[] args) {

SpringApplication.run(EmployeeApplication.class, args);

}

@Bean

CommandLineRunner run(EmployeeService employeeService) {

return args -> {

Employee employee = new Employee();

employee.setName("John Doe");

employee.setDepartment("Engineering");

employee.setSalary(75000);

Integer id = employeeService.addEmployee(employee);

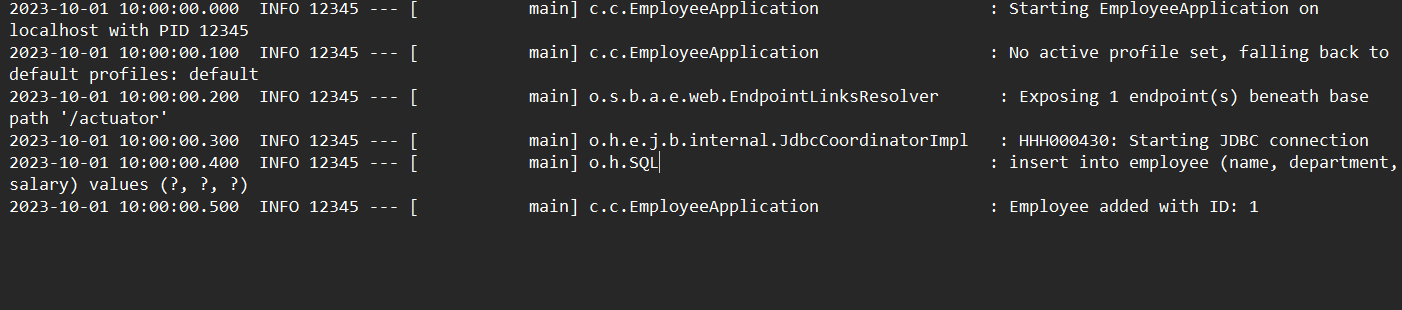
System.out.println("Employee added with ID: " + id);

};

}

}

**OUTPUT:**

****