**1.Spring Data JPA - Quick Example**

CountryNotFoundException

package com.cognizant.springlearn.service.exception;

public class CountryNotFoundException extends Exception {

public CountryNotFoundException(String message) {

super(message);

}

}

CountryService

package com.cognizant.springlearn.service;

import com.cognizant.springlearn.model.Country;

import com.cognizant.springlearn.service.exception.CountryNotFoundException;

import com.cognizant.springlearn.repository.CountryRepository;

import java.util.Optional;

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

@Service

public class CountryService {

private final CountryRepository countryRepository;

public CountryService(CountryRepository countryRepository) {

this.countryRepository = countryRepository;

}

@Transactional

public Country findCountryByCode(String countryCode) throws CountryNotFoundException {

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

if (!result.isPresent()) {

throw new CountryNotFoundException("Country with code " + countryCode + " not found.");

}

return result.get();

}

}

CountryRepository

package com.cognizant.springlearn.repository;

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

import com.cognizant.springlearn.model.Country;

public interface CountryRepository extends JpaRepository<Country, String> {

}

OrmLearnApplication.java

package com.cognizant.springlearn;

import com.cognizant.springlearn.model.Country;

import com.cognizant.springlearn.service.CountryService;

import com.cognizant.springlearn.service.exception.CountryNotFoundException;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class OrmLearnApplication implements CommandLineRunner {

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

@Autowired

private CountryService countryService;

public static void main(String[] args) {

SpringApplication.run(OrmLearnApplication.class, args);

}

@Override

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

testFindCountryByCode();

}

private void testFindCountryByCode() {

LOGGER.info("Start - testFindCountryByCode");

try {

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

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

} catch (CountryNotFoundException e) {

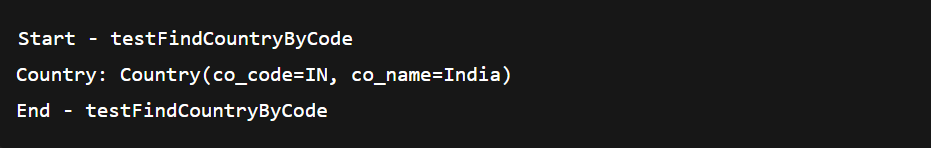
LOGGER.error("Exception: {}", e.getMessage());

}

LOGGER.info("End - testFindCountryByCode");

}

}



**Country**

@Entity

public class Country {

@Id

private String code;

private String name;

}

**CountryRepository**

public interface CountryRepository extends JpaRepository<Country, String> {

}

**CountryService**

@Service

public class CountryService {

@Autowired

private CountryRepository countryRepository;

public Country findCountryByCode(String code) throws CountryNotFoundException {

return countryRepository.findById(code)

.orElseThrow(() -> new CountryNotFoundException("Country with code " + code + " not found"));

}

}

**Exception - CountryNotFoundException**

public class CountryNotFoundException extends Exception {

public CountryNotFoundException(String message) {

super(message);

}

}

**Test Method - OrmLearnApplication.java**

@Autowired

private CountryService countryService;

@Override

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

try {

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

System.out.println(country);

} catch (CountryNotFoundException e) {

e.printStackTrace();

}

}



**2.Add a new country**

Country country = new Country();

country.setCode("JP");

country.setName("Japan");

countryService.addCountry(country);



**3.Demonstrate Query Methods Feature of Spring Data JPA**

List<Country> findByNameContaining(String keyword);

List<Country> findByNameStartingWith(String prefix);

List<Country> findByNameOrderByNameAsc();

Call from CommandLineRunner

List<Country> countries = countryRepository.findByNameContaining("in");

countries.forEach(System.out.println);



**4.Demonstrate implementation of O/R Mapping**

@Entity

public class Country {

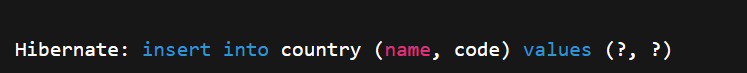
@Id

private String code;

@Column(nullable = false)

private String name;

}



**5. Demonstrate writing HQL and Native Queries**

@Query("SELECT c FROM Country c WHERE c.name LIKE %:keyword%")

List<Country> searchByNameUsingHql(@Param("keyword") String keyword);

@Query(value = "SELECT \* FROM country WHERE name LIKE %:keyword%", nativeQuery = true)

List<Country> searchByNameUsingNative(@Param("keyword") String keyword);

