**Exercise 1: Spring Data JPA - Quick Example**

**Mysql:**

create schema ormlearn;

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');

**Country.java**

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="co\_code")

    private String code;

    @Column(name="co\_name")

    private String name;

    public String getCode(){ return  code; }

    public String getName(){ return name; }

    public void setCode(String code){ this.code=code; }

    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;

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.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

import com.cognizant.orm\_learn.CountryRepository;

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

@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();

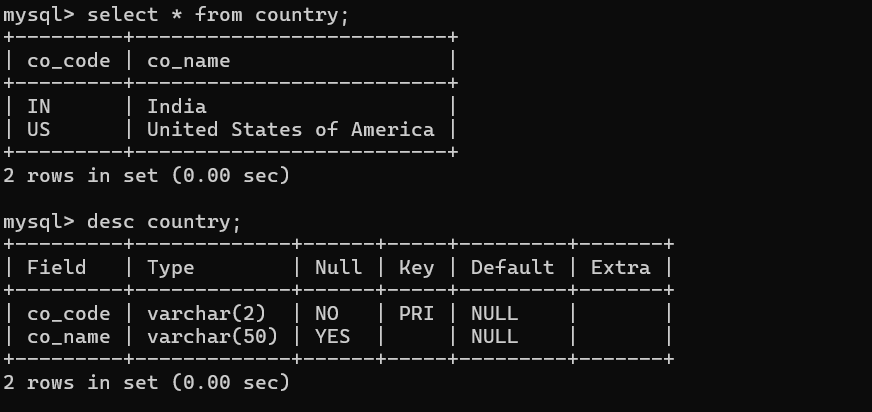
        LOGGER.debug("countries={}", countries);

        LOGGER.info("End");

    }

}

**Output:**





**Exercise 4: Difference between JPA, Hibernate and Spring Data JPA**

Hibernate is the ORM tool which implements JPA. In this there are no annotations. Every is explicitly mentioned as given in the example:

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;

}

But in Spring JPA we do not write this whole we just use annotations which are defined in libraries:

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

@Autowire

private EmployeeRepository employeeRepository;

@Transactional

public void addEmployee(Employee employee) {

employeeRepository.save(employee);

}