**1.Spring-Data-JPA-Hands-On**

**Hands on 1**

**Spring Data JPA - Quick Example**

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 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 + "]";

    }

}

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> {

}

package com.cognizant.orm\_learn.service;

import java.util.List;

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;

import jakarta.transaction.Transactional;

@Service

public class CountryService {

    @Autowired

    private CountryRepository countryRepository;

    @Transactional

    public List<Country> getAllCountries() {

        return countryRepository.findAll();

    }

}

package com.cognizant.orm\_learn;

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

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

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 java.util.List;

@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);

        LOGGER.info("Inside main");

        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");

    }

}

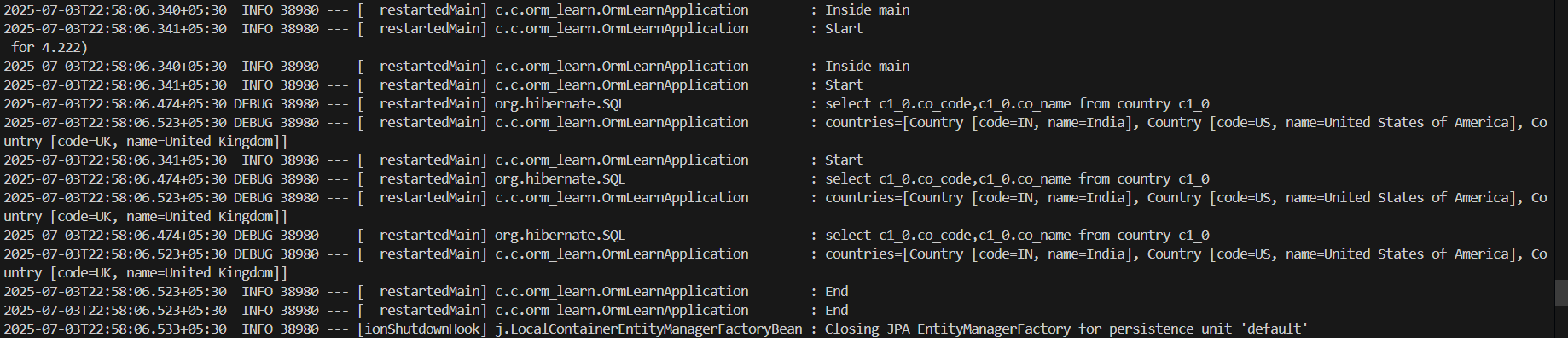
**Data.sql**

INSERT INTO country (co\_code, co\_name) VALUES ('IN', 'India');

INSERT INTO country (co\_code, co\_name) VALUES ('US', 'United States of America');

INSERT INTO country (co\_code, co\_name) VALUES ('UK', 'United Kingdom');

**Output**

****

**Hands on 4**

**Difference between JPA, Hibernate and Spring Data JPA**

package com.cognizant.orm\_learn.dao;

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

import org.hibernate.HibernateException;

import org.hibernate.Session;

import org.hibernate.Transaction;

import org.hibernate.SessionFactory;

public class HibernateEmployeeDAO {

    private SessionFactory factory;

    public HibernateEmployeeDAO(SessionFactory factory) {

        this.factory = factory;

    }

    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;

    }

}

package com.cognizant.orm\_learn.service;

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

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

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

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

@Service

public class EmployeeService {

    @Autowired

    private EmployeeRepository employeeRepository;

    @Transactional

    public void addEmployee(Employee employee) {

        employeeRepository.save(employee);

    }

}

package com.cognizant.orm\_learn.repository;

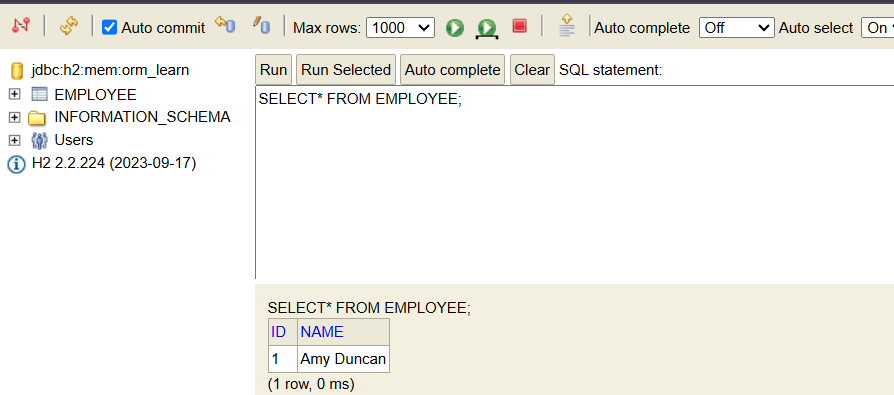
import org.springframework.data.jpa.repository.JpaRepository;

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

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

**Output**

****

**Hands on 5**

**Implement services for managing Country**

package com.cognizant.orm\_learn.model;

import jakarta.persistence.Entity;

import jakarta.persistence.Id;

@Entity

public class Country {

    @Id

    private String coCode;

    private String coName;

    public Country() {}

    public Country(String coCode, String coName) {

        this.coCode = coCode;

        this.coName = coName;

    }

    public String getCoCode() {

        return coCode;

    }

    public void setCoCode(String coCode) {

        this.coCode = coCode;

    }

    public String getCoName() {

        return coName;

    }

    public void setCoName(String coName) {

        this.coName = coName;

    }

}

package com.cognizant.orm\_learn.repository;

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

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

import java.util.List;

public interface CountryRepository extends JpaRepository<Country, String> {

    List<Country> findByCoNameContaining(String name);

}

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.List;

import java.util.Optional;

@Service

public class CountryService {

    @Autowired

    private CountryRepository countryRepository;

    public Country findCountryByCode(String code) {

        Optional<Country> country = countryRepository.findById(code);

        return country.orElse(null);

    }

    public Country addCountry(Country country) {

        return countryRepository.save(country);

    }

    public Country updateCountry(String code, Country updatedCountry) {

        Optional<Country> existing = countryRepository.findById(code);

        if (existing.isPresent()) {

            Country country = existing.get();

            country.setCoName(updatedCountry.getCoName());

            return countryRepository.save(country);

        }

        return null;

    }

    public void deleteCountry(String code) {

        countryRepository.deleteById(code);

    }

    public List<Country> findCountriesByPartialName(String partialName) {

        return countryRepository.findByCoNameContaining(partialName);

    }

}

package com.cognizant.orm\_learn;

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.cognizant.orm\_learn.model.Country;

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

@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) {

        try {

            Country country = new Country();

            country.setCoCode("XX");

            country.setCoName("Sampleland");

            countryService.addCountry(country);

            System.out.println("✅ Country added using Spring Data JPA!");

            Country found = countryService.findCountryByCode("XX");

            System.out.println("✅ Found: " + found.getCoName());

            found.setCoName("Updatedland");

            countryService.updateCountry(found.getCoCode(), found);

            System.out.println("✅ Country updated!");

            System.out.println("✅ Countries with 'land':");

            countryService.findCountriesByPartialName("land")

                    .forEach(c -> System.out.println(c.getCoName()));

            countryService.deleteCountry("XX");

            System.out.println("✅ Country deleted!");

        } catch (Exception e) {

            e.printStackTrace();

        }

    }

}

INSERT INTO country (co\_code, co\_name) VALUES ('AF', 'Afghanistan');

INSERT INTO country (co\_code, co\_name) VALUES ('AL', 'Albania');

INSERT INTO country (co\_code, co\_name) VALUES ('DZ', 'Algeria');

INSERT INTO country (co\_code, co\_name) VALUES ('AS', 'American Samoa');

INSERT INTO country (co\_code, co\_name) VALUES ('AD', 'Andorra');

INSERT INTO country (co\_code, co\_name) VALUES ('AO', 'Angola');

INSERT INTO country (co\_code, co\_name) VALUES ('AI', 'Anguilla');

INSERT INTO country (co\_code, co\_name) VALUES ('AQ', 'Antarctica');

INSERT INTO country (co\_code, co\_name) VALUES ('AG', 'Antigua and Barbuda');

INSERT INTO country (co\_code, co\_name) VALUES ('AR', 'Argentina');

INSERT INTO country (co\_code, co\_name) VALUES ('AM', 'Armenia');

INSERT INTO country (co\_code, co\_name) VALUES ('AW', 'Aruba');

INSERT INTO country (co\_code, co\_name) VALUES ('AU', 'Australia');

INSERT INTO country (co\_code, co\_name) VALUES ('AT', 'Austria');

INSERT INTO country (co\_code, co\_name) VALUES ('AZ', 'Azerbaijan');

INSERT INTO country (co\_code, co\_name) VALUES ('BS', 'Bahamas');

INSERT INTO country (co\_code, co\_name) VALUES ('BH', 'Bahrain');

INSERT INTO country (co\_code, co\_name) VALUES ('BD', 'Bangladesh');

INSERT INTO country (co\_code, co\_name) VALUES ('BB', 'Barbados');

INSERT INTO country (co\_code, co\_name) VALUES ('BY', 'Belarus');

INSERT INTO country (co\_code, co\_name) VALUES ('BE', 'Belgium');

INSERT INTO country (co\_code, co\_name) VALUES ('BZ', 'Belize');

INSERT INTO country (co\_code, co\_name) VALUES ('BJ', 'Benin');

INSERT INTO country (co\_code, co\_name) VALUES ('BM', 'Bermuda');

INSERT INTO country (co\_code, co\_name) VALUES ('BT', 'Bhutan');

INSERT INTO country (co\_code, co\_name) VALUES ('BO', 'Bolivia');

INSERT INTO country (co\_code, co\_name) VALUES ('BA', 'Bosnia and Herzegovina');

INSERT INTO country (co\_code, co\_name) VALUES ('BW', 'Botswana');

INSERT INTO country (co\_code, co\_name) VALUES ('BV', 'Bouvet Island');

INSERT INTO country (co\_code, co\_name) VALUES ('BR', 'Brazil');

INSERT INTO country (co\_code, co\_name) VALUES ('IO', 'British Indian Ocean Territory');

INSERT INTO country (co\_code, co\_name) VALUES ('BN', 'Brunei Darussalam');

INSERT INTO country (co\_code, co\_name) VALUES ('BG', 'Bulgaria');

INSERT INTO country (co\_code, co\_name) VALUES ('BF', 'Burkina Faso');

INSERT INTO country (co\_code, co\_name) VALUES ('BI', 'Burundi');

INSERT INTO country (co\_code, co\_name) VALUES ('KH', 'Cambodia');

INSERT INTO country (co\_code, co\_name) VALUES ('CM', 'Cameroon');

INSERT INTO country (co\_code, co\_name) VALUES ('CA', 'Canada');

INSERT INTO country (co\_code, co\_name) VALUES ('CV', 'Cape Verde');

INSERT INTO country (co\_code, co\_name) VALUES ('KY', 'Cayman Islands');

INSERT INTO country (co\_code, co\_name) VALUES ('CF', 'Central African Republic');

INSERT INTO country (co\_code, co\_name) VALUES ('TD', 'Chad');

INSERT INTO country (co\_code, co\_name) VALUES ('CL', 'Chile');

INSERT INTO country (co\_code, co\_name) VALUES ('CN', 'China');

INSERT INTO country (co\_code, co\_name) VALUES ('CX', 'Christmas Island');

INSERT INTO country (co\_code, co\_name) VALUES ('CC', 'Cocos (Keeling) Islands');

INSERT INTO country (co\_code, co\_name) VALUES ('CO', 'Colombia');

INSERT INTO country (co\_code, co\_name) VALUES ('KM', 'Comoros');

INSERT INTO country (co\_code, co\_name) VALUES ('CG', 'Congo');

INSERT INTO country (co\_code, co\_name) VALUES ('CD', 'Congo, the Democratic Republic of the');

INSERT INTO country (co\_code, co\_name) VALUES ('CK', 'Cook Islands');

INSERT INTO country (co\_code, co\_name) VALUES ('CR', 'Costa Rica');

INSERT INTO country (co\_code, co\_name) VALUES ('CI', 'Côte d\Ivoire');

INSERT INTO country (co\_code, co\_name) VALUES ('HR' , 'Croatia');

INSERT INTO country (co\_code, co\_name) VALUES ('CU', 'Cuba');

INSERT INTO country (co\_code, co\_name) VALUES ('CY', 'Cyprus');

INSERT INTO country (co\_code, co\_name) VALUES ('CZ', 'Czech Republic');

INSERT INTO country (co\_code, co\_name) VALUES ('DK', 'Denmark');

INSERT INTO country (co\_code, co\_name) VALUES ('DJ', 'Djibouti');

INSERT INTO country (co\_code, co\_name) VALUES ('DM', 'Dominica');

INSERT INTO country (co\_code, co\_name) VALUES ('DO', 'Dominican Republic');

INSERT INTO country (co\_code, co\_name) VALUES ('EC', 'Ecuador');

INSERT INTO country (co\_code, co\_name) VALUES ('EG', 'Egypt');

INSERT INTO country (co\_code, co\_name) VALUES ('SV', 'El Salvador');

INSERT INTO country (co\_code, co\_name) VALUES ('GQ', 'Equatorial Guinea');

INSERT INTO country (co\_code, co\_name) VALUES ('ER', 'Eritrea');

INSERT INTO country (co\_code, co\_name) VALUES ('EE', 'Estonia');

INSERT INTO country (co\_code, co\_name) VALUES ('ET', 'Ethiopia');

INSERT INTO country (co\_code, co\_name) VALUES ('FK', 'Falkland Islands (Malvinas)');

INSERT INTO country (co\_code, co\_name) VALUES ('FO', 'Faroe Islands');

INSERT INTO country (co\_code, co\_name) VALUES ('FJ', 'Fiji');

INSERT INTO country (co\_code, co\_name) VALUES ('FI', 'Finland');

INSERT INTO country (co\_code, co\_name) VALUES ('FR', 'France');

INSERT INTO country (co\_code, co\_name) VALUES ('GF', 'French Guiana');

INSERT INTO country (co\_code, co\_name) VALUES ('PF', 'French Polynesia');

INSERT INTO country (co\_code, co\_name) VALUES ('TF', 'French Southern Territories');

INSERT INTO country (co\_code, co\_name) VALUES ('GA', 'Gabon');

INSERT INTO country (co\_code, co\_name) VALUES ('GM', 'Gambia');

INSERT INTO country (co\_code, co\_name) VALUES ('GE', 'Georgia');

INSERT INTO country (co\_code, co\_name) VALUES ('DE', 'Germany');

INSERT INTO country (co\_code, co\_name) VALUES ('GH', 'Ghana');

INSERT INTO country (co\_code, co\_name) VALUES ('GI', 'Gibraltar');

INSERT INTO country (co\_code, co\_name) VALUES ('GR', 'Greece');

INSERT INTO country (co\_code, co\_name) VALUES ('GL', 'Greenland');

INSERT INTO country (co\_code, co\_name) VALUES ('GD', 'Grenada');

INSERT INTO country (co\_code, co\_name) VALUES ('GP', 'Guadeloupe');

INSERT INTO country (co\_code, co\_name) VALUES ('GU', 'Guam');

INSERT INTO country (co\_code, co\_name) VALUES ('GT', 'Guatemala');

INSERT INTO country (co\_code, co\_name) VALUES ('GN', 'Guinea');

INSERT INTO country (co\_code, co\_name) VALUES ('GW', 'Guinea-Bissau');

INSERT INTO country (co\_code, co\_name) VALUES ('GY', 'Guyana');

INSERT INTO country (co\_code, co\_name) VALUES ('HT', 'Haiti');

INSERT INTO country (co\_code, co\_name) VALUES ('HM', 'Heard Island and McDonald Islands');

INSERT INTO country (co\_code, co\_name) VALUES ('VA', 'Holy See (Vatican City State)');

INSERT INTO country (co\_code, co\_name) VALUES ('HN', 'Honduras');

INSERT INTO country (co\_code, co\_name) VALUES ('HK', 'Hong Kong');

INSERT INTO country (co\_code, co\_name) VALUES ('HU', 'Hungary');

INSERT INTO country (co\_code, co\_name) VALUES ('IS', 'Iceland');

INSERT INTO country (co\_code, co\_name) VALUES ('IN', 'India');

INSERT INTO country (co\_code, co\_name) VALUES ('ID', 'Indonesia');

INSERT INTO country (co\_code, co\_name) VALUES ('IR', 'Iran, Islamic Republic of');

INSERT INTO country (co\_code, co\_name) VALUES ('IQ', 'Iraq');

INSERT INTO country (co\_code, co\_name) VALUES ('IE', 'Ireland');

INSERT INTO country (co\_code, co\_name) VALUES ('IL', 'Israel');

INSERT INTO country (co\_code, co\_name) VALUES ('IT', 'Italy');

INSERT INTO country (co\_code, co\_name) VALUES ('JM', 'Jamaica');

INSERT INTO country (co\_code, co\_name) VALUES ('JP', 'Japan');

INSERT INTO country (co\_code, co\_name) VALUES ('JO', 'Jordan');

INSERT INTO country (co\_code, co\_name) VALUES ('KZ', 'Kazakhstan');

INSERT INTO country (co\_code, co\_name) VALUES ('KE', 'Kenya');

INSERT INTO country (co\_code, co\_name) VALUES ('KI', 'Kiribati');

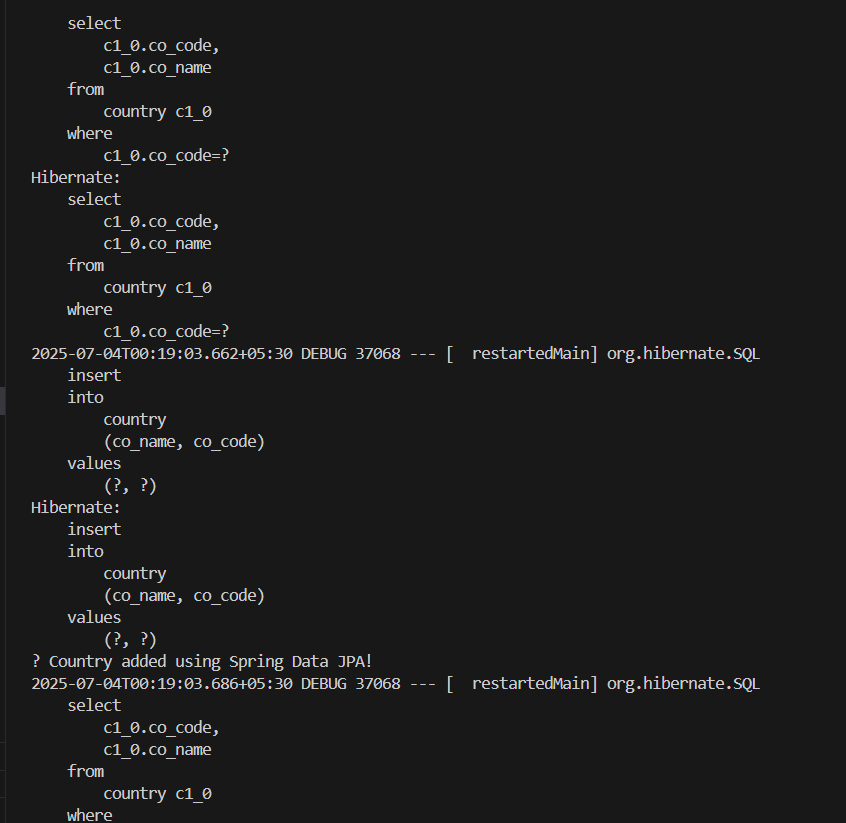
INSERT INTO country (co\_code, co\_name) VALUES ('KP', 'Korea, Democratic Peoples Republic of');

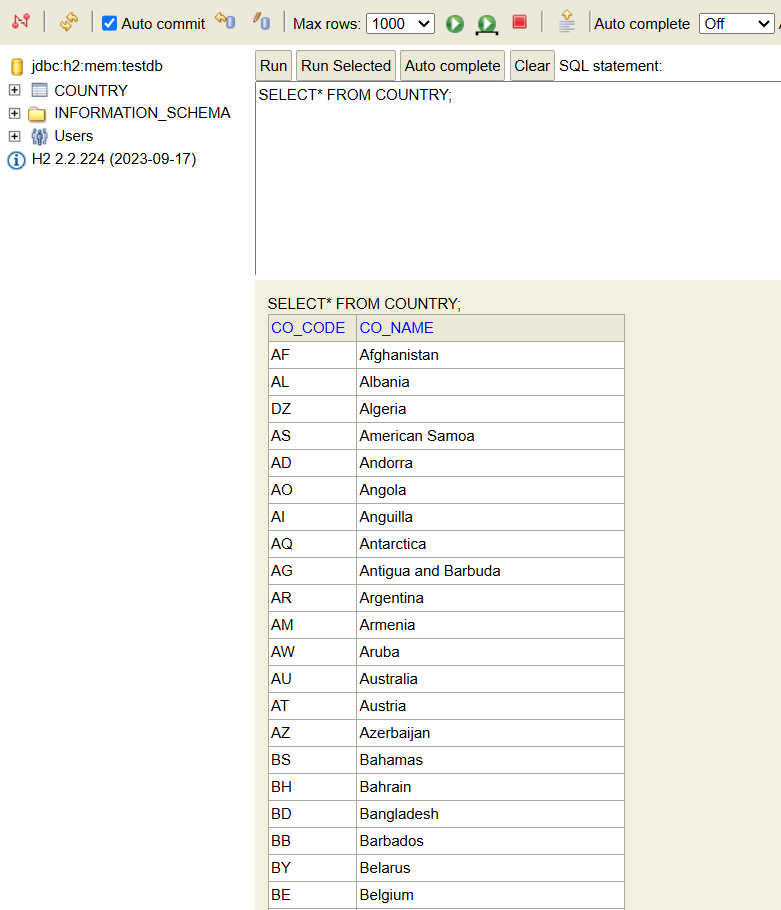
INSERT INTO country (co\_code, co\_name) VALUES ('KR', 'Korea, Republic of');

INSERT INTO country (co\_code, co\_name) VALUES ('KW', 'Kuwait');

INSERT INTO country (co\_code, co\_name) VALUES ('KG', 'Kyrgyzstan');

**Output**

****

****

**Hands on 6**

**Find a country based on country code**

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

public class CountryNotFoundException extends Exception {

    public CountryNotFoundException(String message) {

        super(message);

    }

}

package com.cognizant.orm\_learn.service;

import java.util.List;

import java.util.Optional;

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

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

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

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

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

@Service

public class CountryService {

    @Autowired

    private CountryRepository countryRepository;

    @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);

    }

    @Transactional

    public void updateCountry(String countryCode, Country updatedCountry) throws CountryNotFoundException {

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

        if (!result.isPresent()) {

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

        }

        Country country = result.get();

        country.setCoName(updatedCountry.getCoName());

        countryRepository.save(country);

    }

    @Transactional

    public void deleteCountry(String countryCode) throws CountryNotFoundException {

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

        if (!result.isPresent()) {

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

        }

        countryRepository.deleteById(countryCode);

    }

    @Transactional

    public List<Country> findCountriesByPartialName(String partialName) {

        return countryRepository.findByCoNameContaining(partialName);

    }

    @Transactional

    public List<Country> getAllCountries() {

        return countryRepository.findAll();

    }

}

package com.cognizant.orm\_learn;

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;

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

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

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

@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 {

        LOGGER.info("START");

        testGetCountryByCode();

        LOGGER.info("END");

    }

    private void testGetCountryByCode() {

        LOGGER.info("Start testGetCountryByCode");

        try {

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

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

        } catch (CountryNotFoundException e) {

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

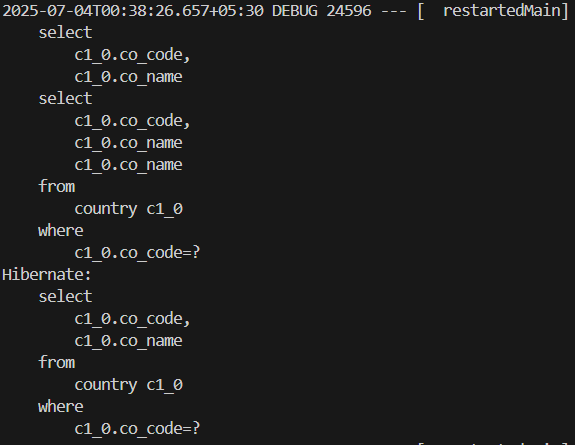
        }

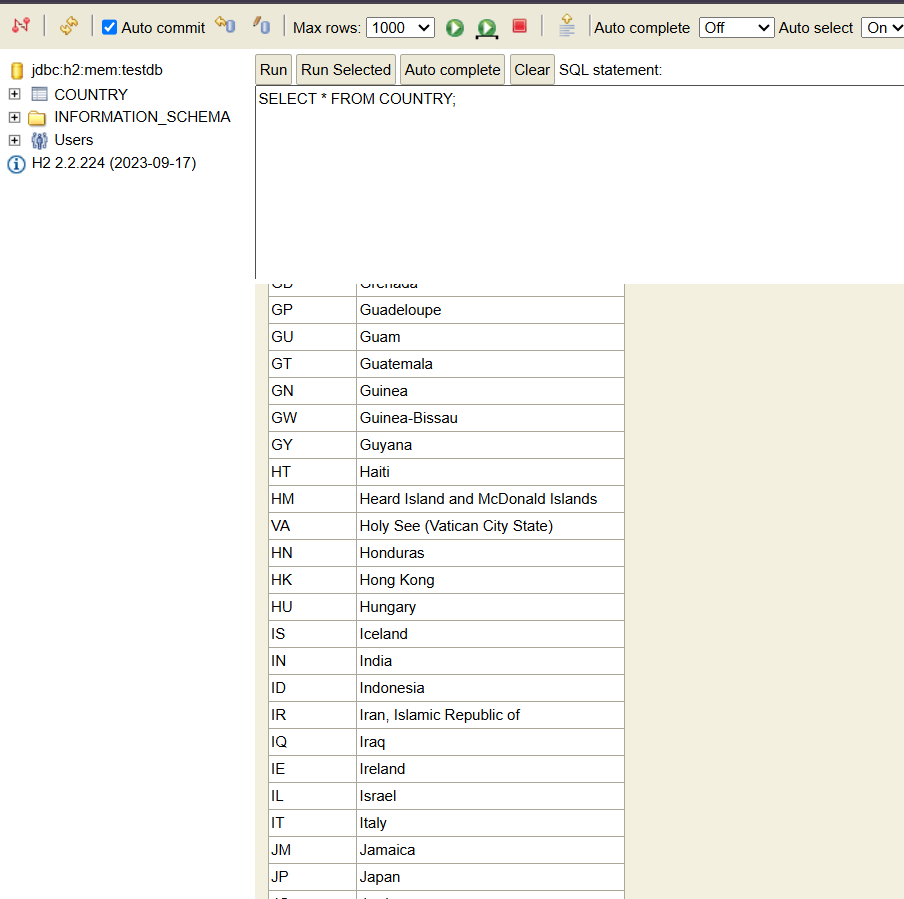
        LOGGER.info("End testGetCountryByCode");

    }

}

**Output**

****

****

**Hands on 7**

**Add a new country**

package com.cognizant.orm\_learn.service;

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

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

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

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

@Service

public class CountryService {

    @Autowired

    private CountryRepository countryRepository;

    @Transactional

    public void addCountry(Country country) {

        countryRepository.save(country);

    }

    @Transactional

    public Country findCountryByCode(String countryCode) throws com.cognizant.orm\_learn.service.exception.CountryNotFoundException {

        return countryRepository.findById(countryCode)

                .orElseThrow(() -> new com.cognizant.orm\_learn.service.exception.CountryNotFoundException("Country not found with code: " + countryCode));

    }

}

package com.cognizant.orm\_learn;

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;

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

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

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

@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 {

        testAddCountry();

    }

    private void testAddCountry() throws CountryNotFoundException {

        LOGGER.info("Start testAddCountry");

        Country country = new Country();

        country.setCoCode("JP");

        country.setCoName("Japan");

        countryService.addCountry(country);

        Country countryFromDB = countryService.findCountryByCode("JP");

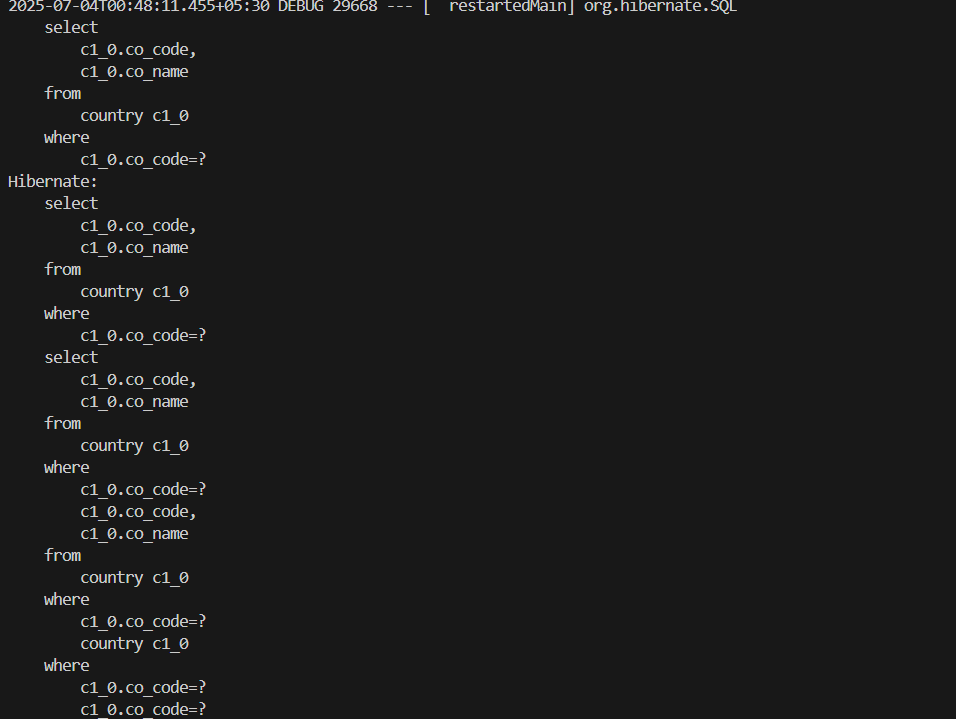
        LOGGER.debug("Added Country: {}", countryFromDB);

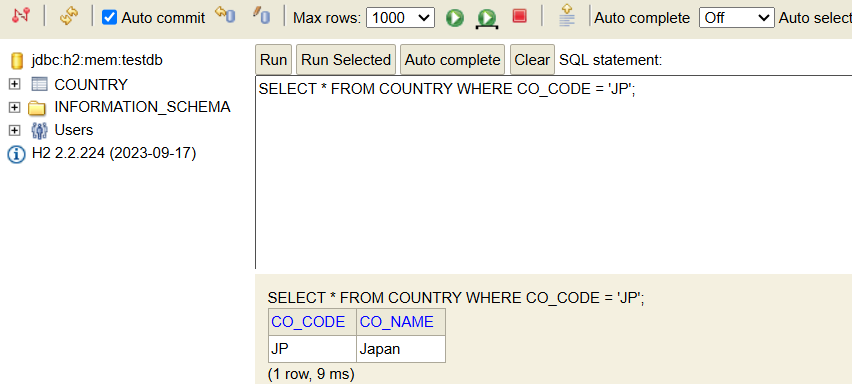
        LOGGER.info("End testAddCountry");

    }

}

**Output**

****

****