**Hands-On 1: Write queries on country table using Query Methods**

**Country.java:**

package com.cognizant.spring\_data\_jpa\_querymethods.entity;

import jakarta.persistence.Entity;

import jakarta.persistence.Id;

@Entity

public class Country {

@Id

private String code;

private String name;

public Country() {}

public Country(String code, String name) {

this.code = code;

this.name = 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;

}

}

**CountryRepository.java:**

package com.cognizant.spring\_data\_jpa\_querymethods.repository;

import java.util.List;

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

import com.cognizant.spring\_data\_jpa\_querymethods.entity.Country;

public interface CountryRepository extends JpaRepository<Country, String> {

List<Country> findByNameContainingIgnoreCase(String keyword);

List<Country> findByNameContainingIgnoreCaseOrderByNameAsc(String keyword);

List<Country> findByNameStartingWithIgnoreCase(String prefix);

}

**SpringDataJpaQuerymethodsApplication.java:**

package com.cognizant.spring\_data\_jpa\_querymethods;

import java.util.List;

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.spring\_data\_jpa\_querymethods.entity.Country;

import com.cognizant.spring\_data\_jpa\_querymethods.repository.CountryRepository;

@SpringBootApplication

public class SpringDataJpaQuerymethodsApplication implements CommandLineRunner {

@Autowired

private CountryRepository countryRepository;

public static void main(String[] args) {

SpringApplication.run(SpringDataJpaQuerymethodsApplication.class, args);

}

@Override

public void run(String... args) {

System.out.println(">> Inside run()");

if (countryRepository.count() == 0) {

countryRepository.saveAll(List.of(

new Country("IN", "India"),

new Country("US", "United States"),

new Country("FR", "France"),

new Country("ZA", "South Africa"),

new Country("ZM", "Zambia"),

new Country("BR", "Brazil")

));

}

searchContaining("ou");

searchContainingSorted("ou");

searchStartingWith("Z");

}

private void searchContaining(String keyword) {

System.out.println(">> Inside searchContaining()");

List<Country> result = countryRepository.findByNameContainingIgnoreCase(keyword);

result.forEach(c -> System.out.println(c.getCode() + " - " + c.getName()));

}

private void searchContainingSorted(String keyword) {

System.out.println(">> Inside searchContainingSorted()");

List<Country> result = countryRepository.findByNameContainingIgnoreCaseOrderByNameAsc(keyword);

result.forEach(c -> System.out.println(c.getCode() + " - " + c.getName()));

}

private void searchStartingWith(String prefix) {

System.out.println(">> Inside searchStartingWith()");

List<Country> result = countryRepository.findByNameStartingWithIgnoreCase(prefix);

result.forEach(c -> System.out.println(c.getCode() + " - " + c.getName()));

}

}

**application.properties:**

spring.datasource.url=jdbc:h2:mem:testdb

spring.datasource.driverClassName=org.h2.Driver

spring.datasource.username=sa

spring.datasource.password=

spring.jpa.database-platform=org.hibernate.dialect.H2Dialect

spring.h2.console.enabled=true

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

**import.sql:**

INSERT INTO country (code, name) VALUES ('IN', 'India');

INSERT INTO country (code, name) VALUES ('US', 'United States');

INSERT INTO country (code, name) VALUES ('AU', 'Australia');

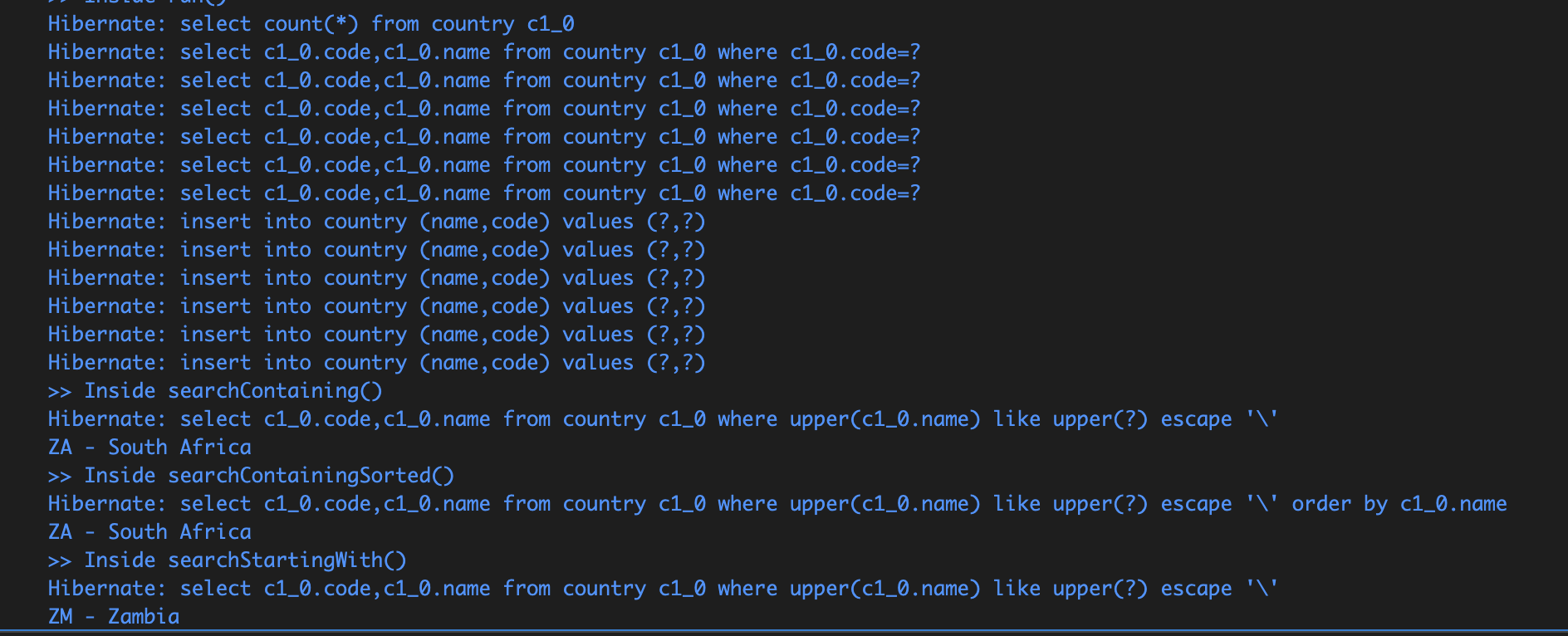
INSERT INTO country (code, name) VALUES ('ZA', 'South Africa');

INSERT INTO country (code, name) VALUES ('ZM', 'Zambia');

INSERT INTO country (code, name) VALUES ('ZW', 'Zimbabwe');

INSERT INTO country (code, name) VALUES ('BV', 'Bouvet Island');

**Output:**

****

**Hands-On 2: Write queries on stock table using Query Methods**

**Country.java:**

package com.cognizant.spring\_data\_jpa\_querymethods.entity;

import jakarta.persistence.Entity;

import jakarta.persistence.Id;

@Entity

public class Country {

@Id

private String code;

private String name;

public Country() {}

public Country(String code, String name) {

this.code = code;

this.name = 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;

}

}

**Stock.java:**

package com.cognizant.spring\_data\_jpa\_querymethods.entity;

import java.math.BigDecimal;

import java.time.LocalDate;

import jakarta.persistence.Column;

import jakarta.persistence.Entity;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

import jakarta.persistence.Table;

@Entity

@Table(name = "stock")

public class Stock {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name = "st\_id")

private int id;

@Column(name = "st\_code")

private String code;

@Column(name = "st\_date")

private LocalDate date;

@Column(name = "st\_open")

private BigDecimal open;

@Column(name = "st\_close")

private BigDecimal close;

@Column(name = "st\_volume")

private BigDecimal volume;

public Stock() {}

public Stock(String code, LocalDate date, BigDecimal open, BigDecimal close, BigDecimal volume) {

this.code = code;

this.date = date;

this.open = open;

this.close = close;

this.volume = volume;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getCode() {

return code;

}

public void setCode(String code) {

this.code = code;

}

public LocalDate getDate() {

return date;

}

public void setDate(LocalDate date) {

this.date = date;

}

public BigDecimal getOpen() {

return open;

}

public void setOpen(BigDecimal open) {

this.open = open;

}

public BigDecimal getClose() {

return close;

}

public void setClose(BigDecimal close) {

this.close = close;

}

public BigDecimal getVolume() {

return volume;

}

public void setVolume(BigDecimal volume) {

this.volume = volume;

}

}

**CountryRepository.java:**

package com.cognizant.spring\_data\_jpa\_querymethods.repository;

import java.util.List;

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

import com.cognizant.spring\_data\_jpa\_querymethods.entity.Country;

public interface CountryRepository extends JpaRepository<Country, String> {

List<Country> findByNameContainingIgnoreCase(String keyword);

List<Country> findByNameContainingIgnoreCaseOrderByNameAsc(String keyword);

List<Country> findByNameStartingWithIgnoreCase(String prefix);

}

**StockRepository.java:**

package com.cognizant.spring\_data\_jpa\_querymethods.repository;

import java.math.BigDecimal;

import java.time.LocalDate;

import java.util.List;

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

import com.cognizant.spring\_data\_jpa\_querymethods.entity.Stock;

public interface StockRepository extends JpaRepository<Stock, Integer> {

List<Stock> findByCodeAndDateBetween(String code, LocalDate start, LocalDate end);

List<Stock> findByCodeAndCloseGreaterThan(String code, BigDecimal price);

List<Stock> findTop3ByOrderByVolumeDesc();

List<Stock> findTop3ByCodeOrderByCloseAsc(String code);

}

**SpringDataJpaQuerymethodsApplication.java:**

package com.cognizant.spring\_data\_jpa\_querymethods;

import java.math.BigDecimal;

import java.time.LocalDate;

import java.util.List;

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.spring\_data\_jpa\_querymethods.entity.Country;

import com.cognizant.spring\_data\_jpa\_querymethods.entity.Stock;

import com.cognizant.spring\_data\_jpa\_querymethods.repository.CountryRepository;

import com.cognizant.spring\_data\_jpa\_querymethods.repository.StockRepository;

@SpringBootApplication

public class SpringDataJpaQuerymethodsApplication implements CommandLineRunner {

@Autowired

private CountryRepository countryRepository;

@Autowired

private StockRepository stockRepository;

public static void main(String[] args) {

SpringApplication.run(SpringDataJpaQuerymethodsApplication.class, args);

}

@Override

public void run(String... args) {

if (stockRepository.count() == 0) {

stockRepository.saveAll(List.of(

new Stock("FB", LocalDate.of(2019, 9, 3), new BigDecimal("184.00"), new BigDecimal("182.39"), new BigDecimal("9779400")),

new Stock("FB", LocalDate.of(2019, 9, 4), new BigDecimal("184.65"), new BigDecimal("187.14"), new BigDecimal("11308000")),

new Stock("GOOGL", LocalDate.of(2019, 4, 22), new BigDecimal("1236.67"), new BigDecimal("1253.76"), new BigDecimal("954200")),

new Stock("GOOGL", LocalDate.of(2019, 4, 23), new BigDecimal("1256.64"), new BigDecimal("1270.59"), new BigDecimal("1593400")),

new Stock("NFLX", LocalDate.of(2018, 12, 24), new BigDecimal("242.00"), new BigDecimal("233.88"), new BigDecimal("9547600")),

new Stock("NFLX", LocalDate.of(2018, 12, 21), new BigDecimal("263.83"), new BigDecimal("246.39"), new BigDecimal("21397600")),

new Stock("NFLX", LocalDate.of(2018, 12, 26), new BigDecimal("233.92"), new BigDecimal("253.67"), new BigDecimal("14402700")),

new Stock("FB", LocalDate.of(2019, 1, 31), new BigDecimal("165.60"), new BigDecimal("166.69"), new BigDecimal("77233600"))

));

}

System.out.println(" Facebook Stock - Sep 2019");

stockRepository.findByCodeAndDateBetween(

"FB", LocalDate.of(2019, 9, 1), LocalDate.of(2019, 9, 30)

).forEach(this::printStock);

System.out.println("\n Google Stock - Close > 1250");

stockRepository.findByCodeAndCloseGreaterThan(

"GOOGL", new BigDecimal("1250")

).forEach(this::printStock);

System.out.println("\n Top 3 by Volume");

stockRepository.findTop3ByOrderByVolumeDesc()

.forEach(this::printStock);

System.out.println("\n Netflix - Lowest Closing Prices");

stockRepository.findTop3ByCodeOrderByCloseAsc("NFLX")

.forEach(this::printStock);

}

private void printStock(Stock stock) {

System.out.println(

stock.getCode() + " | " + stock.getDate() + " | " +

stock.getOpen() + " | " + stock.getClose() + " | " +

stock.getVolume()

);

}

private void searchContaining(String keyword) {

System.out.println(">> Inside searchContaining()");

List<Country> result = countryRepository.findByNameContainingIgnoreCase(keyword);

result.forEach(c -> System.out.println(c.getCode() + " - " + c.getName()));

}

private void searchContainingSorted(String keyword) {

System.out.println(">> Inside searchContainingSorted()");

List<Country> result = countryRepository.findByNameContainingIgnoreCaseOrderByNameAsc(keyword);

result.forEach(c -> System.out.println(c.getCode() + " - " + c.getName()));

}

private void searchStartingWith(String prefix) {

System.out.println(">> Inside searchStartingWith()");

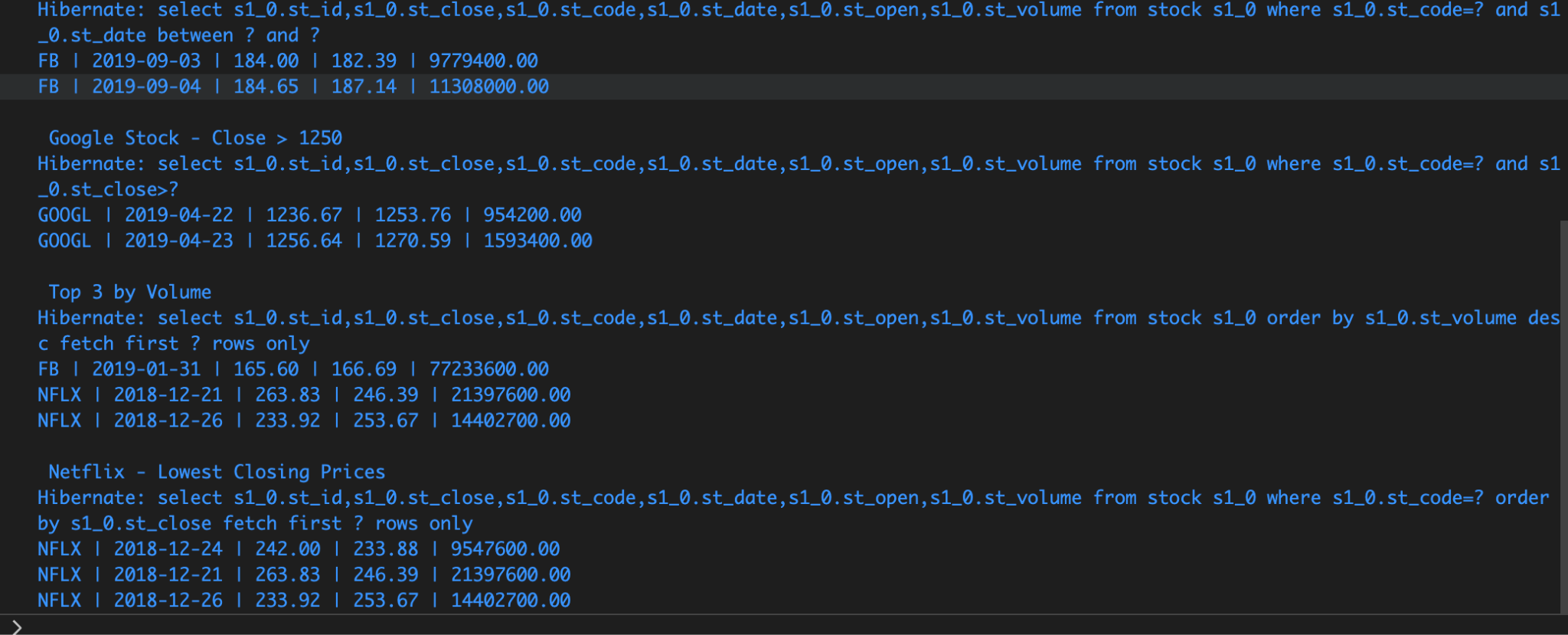
List<Country> result = countryRepository.findByNameStartingWithIgnoreCase(prefix);

result.forEach(c -> System.out.println(c.getCode() + " - " + c.getName()));

}

}

**Output:**



**Hands-On 3: Create payroll tables and bean mapping**

**Employee.java:**

package com.cognizant.spring\_data\_jpa\_querymethods.model;

import java.util.Date;

import java.util.List;

import jakarta.persistence.Column;

import jakarta.persistence.Entity;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

import jakarta.persistence.JoinColumn;

import jakarta.persistence.JoinTable;

import jakarta.persistence.ManyToMany;

import jakarta.persistence.ManyToOne;

import jakarta.persistence.Table;

@Entity

@Table(name = "employee")

public class Employee {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

@Column

private String name;

@Column

private double salary;

@Column

private boolean permanent;

@Column(name = "date\_of\_birth")

private Date dateOfBirth;

@ManyToOne

@JoinColumn(name = "department\_id")

private Department department;

@ManyToMany

@JoinTable(name = "employee\_skill",

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

inverseJoinColumns = @JoinColumn(name = "skill\_id"))

private List<Skill> skillList;

public Employee() {}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public double getSalary() {

return salary;

}

public void setSalary(double salary) {

this.salary = salary;

}

public boolean isPermanent() {

return permanent;

}

public void setPermanent(boolean permanent) {

this.permanent = permanent;

}

public Date getDateOfBirth() {

return dateOfBirth;

}

public void setDateOfBirth(Date dateOfBirth) {

this.dateOfBirth = dateOfBirth;

}

public Department getDepartment() {

return department;

}

public void setDepartment(Department department) {

this.department = department;

}

public List<Skill> getSkillList() {

return skillList;

}

public void setSkillList(List<Skill> skillList) {

this.skillList = skillList;

}

}

**Department.java:**

package com.cognizant.spring\_data\_jpa\_querymethods.model;

import jakarta.persistence.Column;

import jakarta.persistence.Entity;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

import jakarta.persistence.Table;

@Entity

@Table(name = "department")

public class Department {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

@Column

private String name;

public Department() {}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

}

**Skill.java:**

package com.cognizant.spring\_data\_jpa\_querymethods.model;

import jakarta.persistence.Column;

import jakarta.persistence.Entity;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

import jakarta.persistence.Table;

@Entity

@Table(name = "skill")

public class Skill {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

@Column

private String name;

public Skill() {}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

}

**EmployeeRepository.java:**

package com.cognizant.spring\_data\_jpa\_querymethods.repository;

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

import com.cognizant.spring\_data\_jpa\_querymethods.model.Employee;

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

**DepartmentRepository.java:**

package com.cognizant.spring\_data\_jpa\_querymethods.repository;

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

import com.cognizant.spring\_data\_jpa\_querymethods.model.Department;

public interface DepartmentRepository extends JpaRepository<Department, Integer> {

}

**SkillRepository.java:**

package com.cognizant.spring\_data\_jpa\_querymethods.repository;

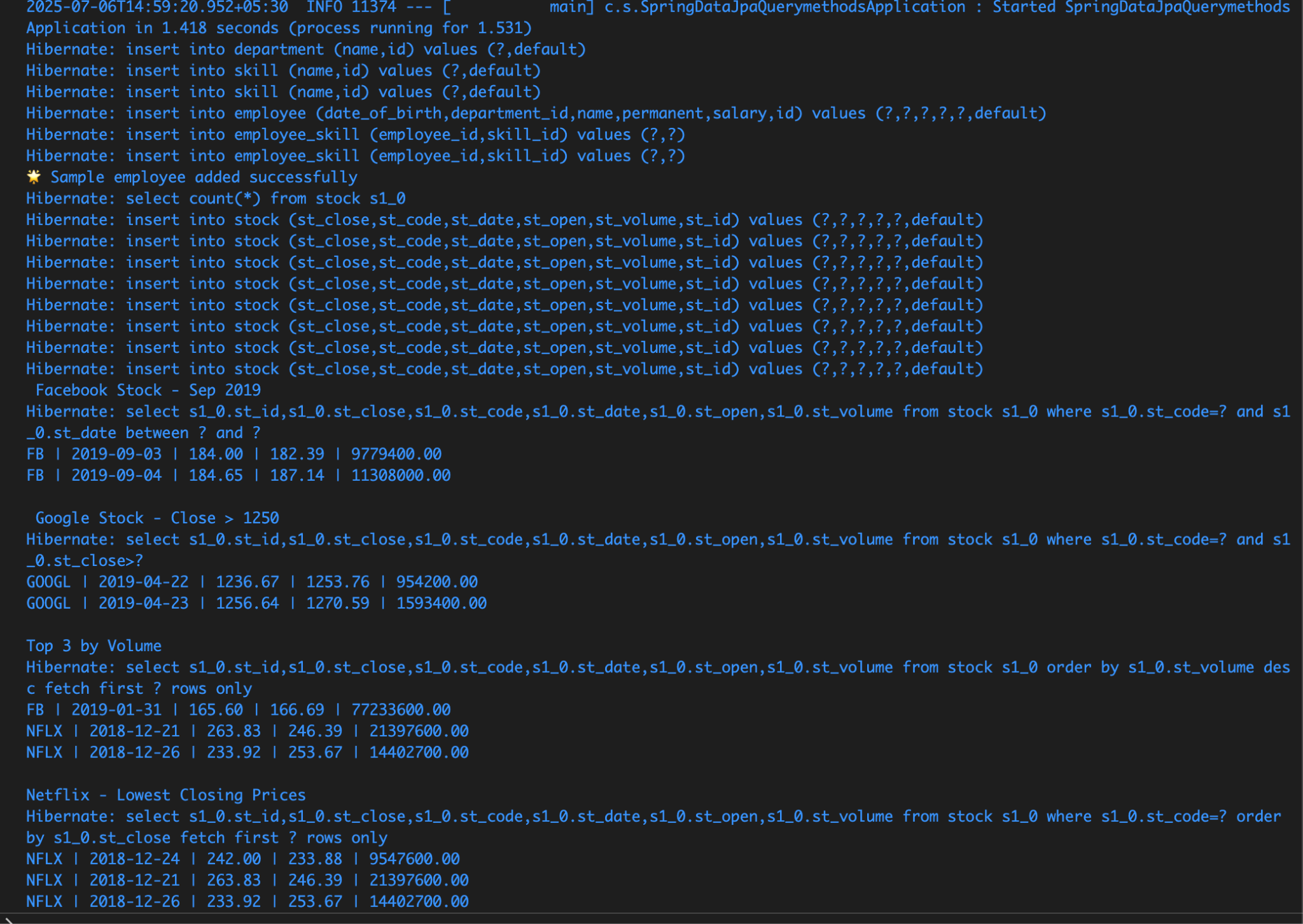
import org.springframework.data.jpa.repository.JpaRepository;

import com.cognizant.spring\_data\_jpa\_querymethods.model.Skill;

public interface SkillRepository extends JpaRepository<Skill, Integer> {

}

**Output:**



**Hands-On 4: Implement many to one relationship between Employee and Department**

**EmployeeService.java:**

package com.cognizant.spring\_data\_jpa\_querymethods.service;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

import com.cognizant.spring\_data\_jpa\_querymethods.model.Employee;

import com.cognizant.spring\_data\_jpa\_querymethods.repository.EmployeeRepository;

@Service

public class EmployeeService {

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

@Autowired

private EmployeeRepository employeeRepository;

@Transactional

public Employee get(int id) {

LOGGER.info("Start");

return employeeRepository.findById(id).get();

}

@Transactional

public void save(Employee employee) {

LOGGER.info("Start");

employeeRepository.save(employee);

LOGGER.info("End");

}

}

**DepartmentService.java:**

package com.cognizant.spring\_data\_jpa\_querymethods.service;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

import com.cognizant.spring\_data\_jpa\_querymethods.model.Department;

import com.cognizant.spring\_data\_jpa\_querymethods.repository.DepartmentRepository;

@Service

public class DepartmentService {

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

@Autowired

private DepartmentRepository departmentRepository;

@Transactional

public Department get(int id) {

LOGGER.info("Start");

return departmentRepository.findById(id).get();

}

@Transactional

public void save(Department department) {

LOGGER.info("Start");

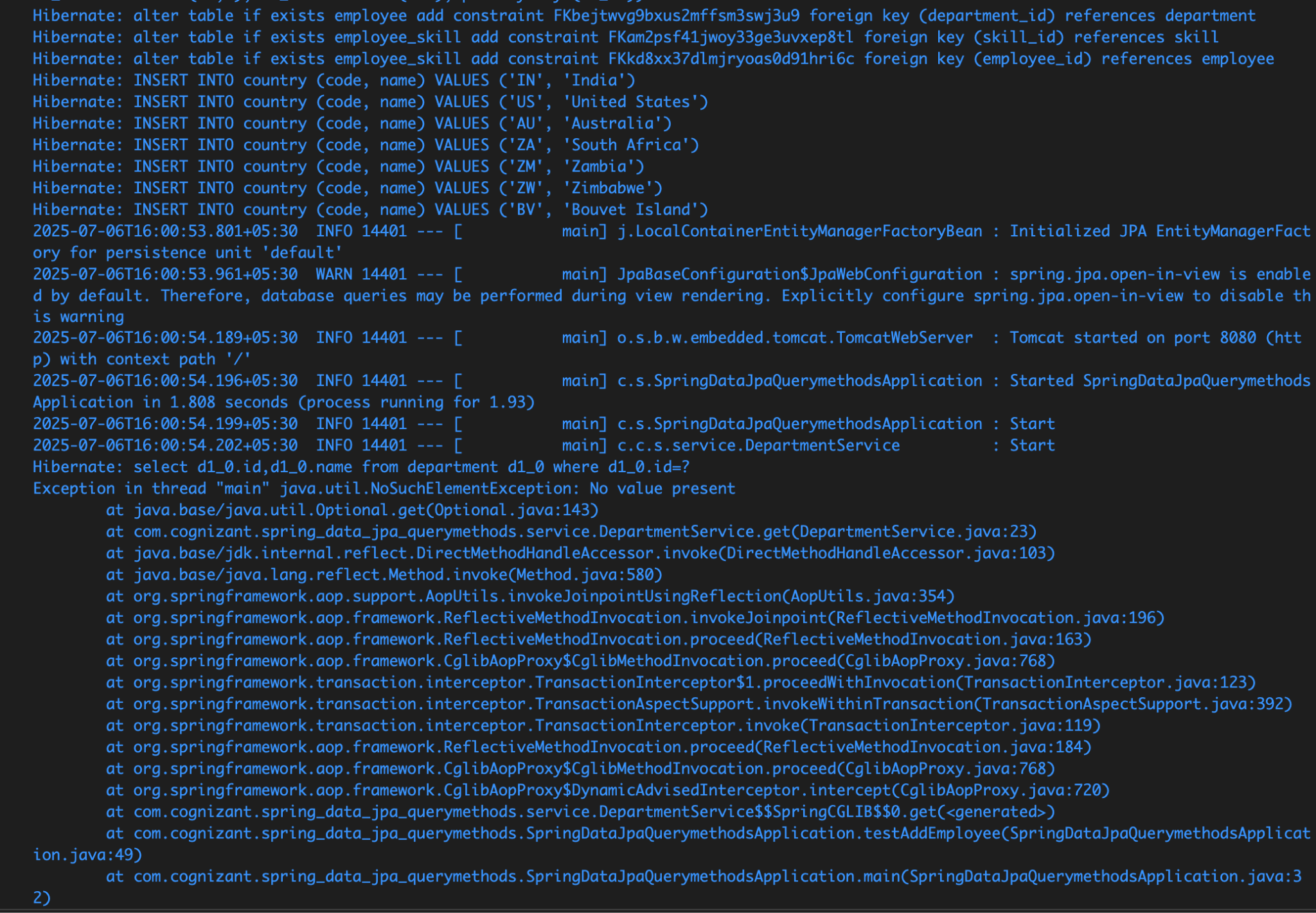
departmentRepository.save(department);

LOGGER.info("End");

}

}

**Output:**



**Hands-On 5: Implement one to many relationship between Employee and Department**

**Output**