**WEEK -3\_Spring – data – jpa – handson2**

**SUPERSET ID: 6393674**

**Demonstrate implementation of Query Methods feature of Spring Data JPA**

**Hands on 1**

**Write queries on country table using Query Methods** 

**SOLUTION:**

**MYSQL WORKBENCH**

CREATE DATABASE country\_db;

USE country\_db;

CREATE TABLE country (

code VARCHAR(2) PRIMARY KEY,

name VARCHAR(100)

);

INSERT INTO country (code, name) VALUES

('BV', 'Bouvet Island'),

('DJ', 'Djibouti'),

('TF', 'French Southern Territories'),

('GP', 'Guadeloupe'),

('LU', 'Luxembourg'),

('ZA', 'South Africa'),

('GS', 'South Georgia and the South Sandwich Islands'),

('SS', 'South Sudan'),

('UM', 'United States Minor Outlying Islands'),

('ZM', 'Zambia'),

('ZW', 'Zimbabwe');

**application.properties**

spring.datasource.url=jdbc:mysql://localhost:3306/country\_db

spring.datasource.username=root

spring.datasource.password=Vasundhara@123

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

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

**Country.java**

package com.example.ormlearn.country\_query\_model;

import jakarta.persistence.Entity;

import jakarta.persistence.Id;

@Entity

public class Country {

@Id

private String code;

private String name;

// Getters and Setters

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.example.ormlearn.country\_query\_repository;

import com.example.ormlearn.country\_query\_model.Country;

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

import java.util.List;

public interface CountryRepository extends JpaRepository<Country, String> {

List<Country> findByNameContaining(String keyword);

List<Country> findByNameContainingOrderByNameAsc(String keyword);

List<Country> findByNameStartingWith(String prefix);

}

**CountryQueryDemoApplication.java**

package com.example.ormlearn.country\_query\_demo;

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 org.springframework.boot.autoconfigure.domain.EntityScan;

import org.springframework.data.jpa.repository.config.EnableJpaRepositories;

import com.example.ormlearn.country\_query\_model.Country;

import com.example.ormlearn.country\_query\_repository.CountryRepository;

@SpringBootApplication(scanBasePackages = "com.example.ormlearn")

@EntityScan("com.example.ormlearn.country\_query\_model")

@EnableJpaRepositories("com.example.ormlearn.country\_query\_repository")

public class CountryQueryDemoApplication implements CommandLineRunner {

@Autowired

private CountryRepository countryRepository;

public static void main(String[] args) {

SpringApplication.run(CountryQueryDemoApplication.class, args);

}

@Override

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

System.out.println("Countries containing 'ou':");

List<Country> result1 = countryRepository.findByNameContaining("ou");

result1.forEach(c -> System.out.println(c.getCode() + " - " + c.getName()));

System.out.println("\nCountries containing 'ou' sorted by name:");

List<Country> result2 = countryRepository.findByNameContainingOrderByNameAsc("ou");

result2.forEach(c -> System.out.println(c.getCode() + " - " + c.getName()));

System.out.println("\nCountries starting with 'Z':");

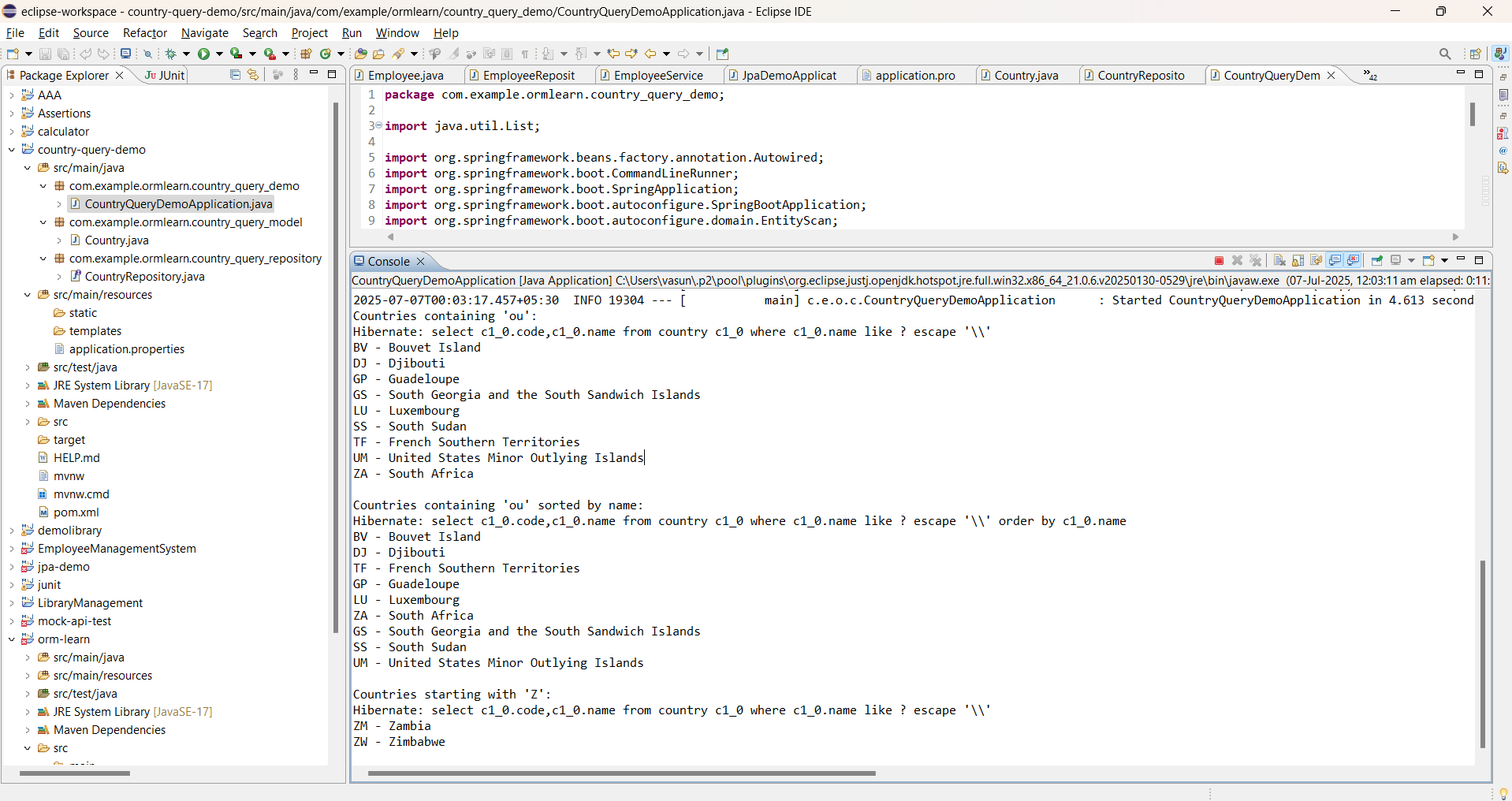
List<Country> result3 = countryRepository.findByNameStartingWith("Z");

result3.forEach(c -> System.out.println(c.getCode() + " - " + c.getName()));

}

}

**OUTPUT**



**Hands on 2**

**Write queries on stock table using Query Methods**   
  
With one year stock data of Facebook, Google and Netflix, we need to implement Spring Data JPA Query Methods for the following :

**SOLUTION:**

**MYSQL WORKBENCH**

CREATE TABLE stock (

id BIGINT AUTO\_INCREMENT PRIMARY KEY,

st\_code VARCHAR(20) NOT NULL,

st\_date DATE NOT NULL,

st\_close DOUBLE,

st\_volume BIGINT,

UNIQUE (st\_code, st\_date)

);

**Stock-data.sql**

INSERT INTO stock (st\_code, st\_date, st\_close, st\_volume) VALUES

('FB', '2023-01-01', 150.25, 100000),

('FB', '2023-01-02', 152.00, 95000),

('GOOGL', '2023-01-01', 2750.00, 120000),

('NFLX', '2023-01-01', 500.75, 110000);

**Application.properties**

# src/main/resources/application.properties

spring.datasource.url=jdbc:mysql://localhost:3306/ormlearn\_stock

spring.datasource.username=root

spring.datasource.password=Vasundhara@123

spring.jpa.hibernate.ddl-auto=none

spring.jpa.show-sql=true

spring.jpa.properties.hibernate.format\_sql=true

server.port=8085

**StockQueryDemoApplication.java**

**package** com.example.stock\_query\_demo;

**import** com.example.stock\_query\_demo.model.Stock;

**import** com.example.stock\_query\_demo.repository.StockRepository;

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

**import** org.springframework.boot.CommandLineRunner;

**import** org.springframework.boot.SpringApplication;

**import** org.springframework.boot.autoconfigure.SpringBootApplication;

**import** java.time.LocalDate;

**import** java.util.List;

@SpringBootApplication

**public** **class** StockQueryDemoApplication **implements** CommandLineRunner {

@Autowired

**private** StockRepository stockRepository;

**public** **static** **void** main(String[] args) {

SpringApplication.*run*(StockQueryDemoApplication.**class**, args);

}

@Override

**public** **void** run(String... args) {

System.***out***.println("=== Stock Query Demo ===");

// 1. Facebook stocks in Jan 2020

List<Stock> fbStocksInJan = stockRepository.findByStCodeAndStDateBetween(

"FB",

LocalDate.*of*(2020, 1, 1),

LocalDate.*of*(2020, 1, 31)

);

System.***out***.println("\n📈 Facebook stocks in Jan 2020:");

fbStocksInJan.forEach(System.***out***::println);

// 2. Google stocks with close > 1500

List<Stock> highCloseStocks = stockRepository.findByStCodeAndStCloseGreaterThan("GOOGL", 1500.0);

System.***out***.println("\n📊 Google stocks with close > 1500:");

highCloseStocks.forEach(System.***out***::println);

// 3. Top 3 stocks with highest volume

List<Stock> topVolumeStocks = stockRepository.findTop3ByOrderByStVolumeDesc();

System.***out***.println("\n🔥 Top 3 stocks with highest volume:");

topVolumeStocks.forEach(System.***out***::println);

// 4. Top 3 lowest close prices for NFLX

List<Stock> lowestClose = stockRepository.findTop3ByStCodeOrderByStCloseAsc("NFLX");

System.***out***.println("\n📉 Top 3 lowest close prices for NFLX:");

lowestClose.forEach(System.***out***::println);

System.***out***.println("\n=== End of Query Results ===");

**Stock.java**

**package** com.example.stock\_query\_demo.model;

**import** jakarta.persistence.\*;

**import** java.time.LocalDate;

@Entity

@Table(name = "stock")

**public** **class** Stock {

@Id

@GeneratedValue(strategy = GenerationType.***IDENTITY***)

**private** Long id;

@Column(name = "st\_code")

**private** String stCode;

@Column(name = "st\_date")

**private** LocalDate stDate;

@Column(name = "st\_close")

**private** Double stClose;

@Column(name = "st\_volume")

**private** Long stVolume;

// Getters and Setters

**public** Long getId() {

**return** id;

}

**public** **void** setId(Long id) {

**this**.id = id;

}

**public** String getStCode() {

**return** stCode;

}

**public** **void** setStCode(String stCode) {

**this**.stCode = stCode;

}

**public** LocalDate getStDate() {

**return** stDate;

}

**public** **void** setStDate(LocalDate stDate) {

**this**.stDate = stDate;

}

**public** Double getStClose() {

**return** stClose;

}

**public** **void** setStClose(Double stClose) {

**this**.stClose = stClose;

}

**public** Long getStVolume() {

**return** stVolume;

}

**public** **void** setStVolume(Long stVolume) {

**this**.stVolume = stVolume;

}

}

**StockRepository.java**

**package** com.example.stock\_query\_demo.repository;

**import** com.example.stock\_query\_demo.model.Stock;

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

**import** java.time.LocalDate;

**import** java.util.List;

**public** **interface** StockRepository **extends** JpaRepository<Stock, Long> {

List<Stock> findByStCodeAndStDateBetween(String stCode, LocalDate startDate, LocalDate endDate);

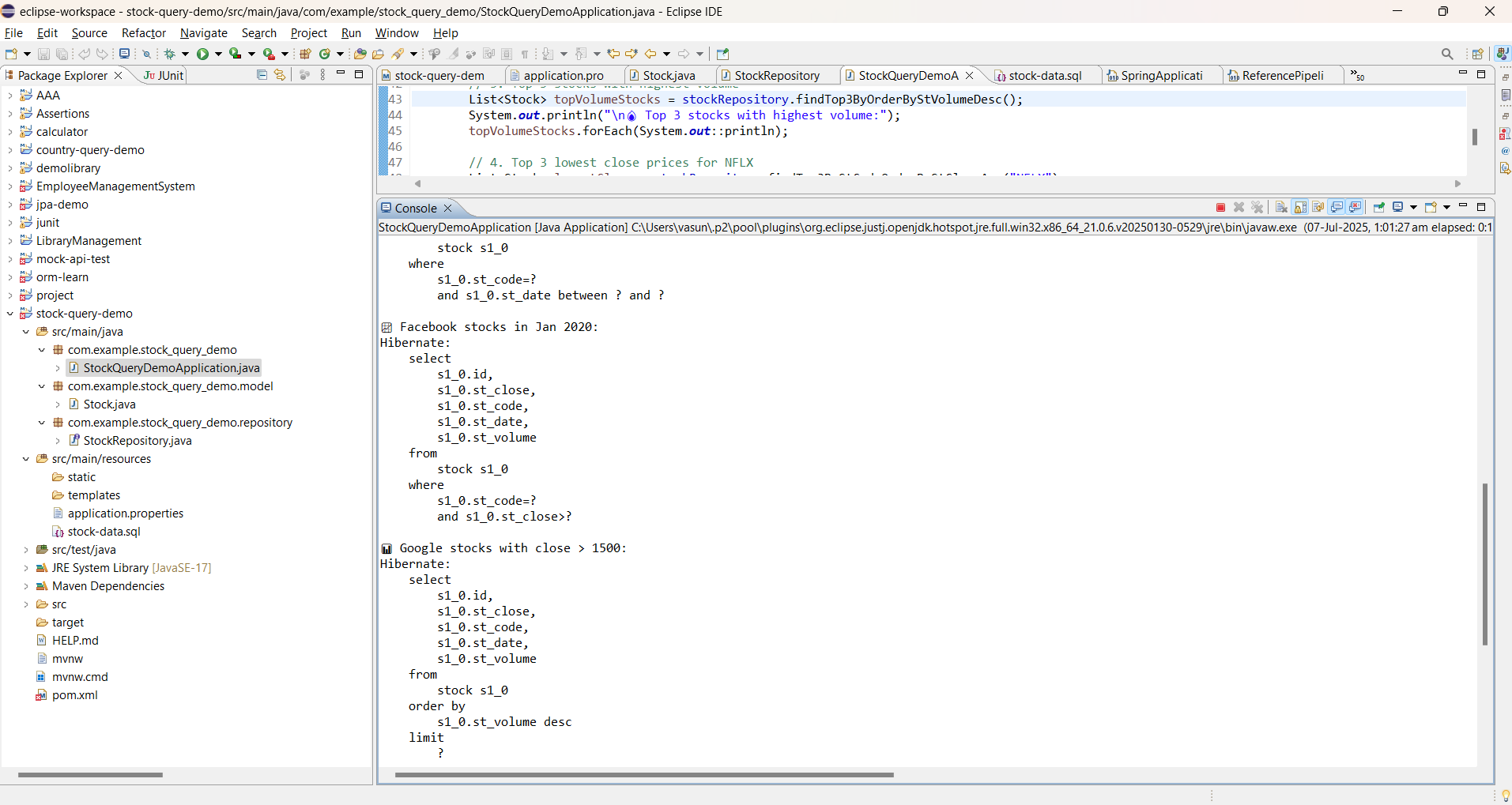
List<Stock> findByStCodeAndStCloseGreaterThan(String stCode, **double** closePrice);

List<Stock> findTop3ByOrderByStVolumeDesc();

List<Stock> findTop3ByStCodeOrderByStCloseAsc(String stCode); // <-- Add this

}

**OUTPUT**

****

**Hands on 3**

**Create payroll tables and bean mapping**

**SOLUTION:**

**MYSQL WORKBENCH**

DROP TABLE IF EXISTS employee\_skill;

DROP TABLE IF EXISTS employee;

DROP TABLE IF EXISTS skill;

DROP TABLE IF EXISTS department;

-- Create department table

CREATE TABLE department (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(255) NOT NULL

);

-- Create skill table

CREATE TABLE skill (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(255) NOT NULL

);

-- Create employee table

CREATE TABLE employee (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(255),

date\_of\_birth DATE,

salary DOUBLE,

permanent BOOLEAN,

dp\_id INT,

FOREIGN KEY (dp\_id) REFERENCES department(id)

);

-- Create employee\_skill join table

CREATE TABLE employee\_skill (

employee\_id INT,

skill\_id INT,

PRIMARY KEY (employee\_id, skill\_id),

FOREIGN KEY (employee\_id) REFERENCES employee(id) ON DELETE CASCADE,

FOREIGN KEY (skill\_id) REFERENCES skill(id) ON DELETE CASCADE

);

-- Insert sample departments

INSERT INTO department (name) VALUES

('HR'),

('Finance'),

('IT');

-- Insert sample skills

INSERT INTO skill (name) VALUES

('Java'),

('Python'),

('SQL'),

('Spring Boot');

-- Insert sample employees

INSERT INTO employee (name, date\_of\_birth, salary, permanent, dp\_id) VALUES

('Alice', '1995-06-15', 50000, TRUE, 1),

('Bob', '1992-08-10', 60000, FALSE, 2),

('Charlie', '1990-01-20', 70000, TRUE, 3);

-- Insert sample employee-skill relationships

INSERT INTO employee\_skill (employee\_id, skill\_id) VALUES

(1, 1), -- Alice - Java

(1, 2), -- Alice - Python

(2, 3), -- Bob - SQL

(3, 1), -- Charlie - Java

(3, 4); -- Charlie - Spring Boot

**Application.properties**

# MySQL DB Connection

spring.datasource.url=jdbc:mysql://localhost:3306/payroll

spring.datasource.username=root

spring.datasource.password=Vasundhara@123

# JPA + Hibernate

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

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

# Optional: Logging SQL

logging.level.org.hibernate.SQL=DEBUG

logging.level.org.hibernate.type.descriptor.sql.BasicBinder=TRACE

**data.sql**

-- Insert into department

INSERT INTO department (id, name) VALUES

(1, 'HR'),

(2, 'Finance'),

(3, 'Engineering');

-- Insert into skill

INSERT INTO skill (id, name) VALUES

(1, 'Java'),

(2, 'SQL'),

(3, 'Spring Boot');

-- Insert into employee

INSERT INTO employee (em\_id, em\_name, em\_salary, em\_permanent, em\_date\_of\_birth, em\_dp\_id) VALUES

(1, 'John Doe', 50000.00, true, '1990-05-10', 1),

(2, 'Jane Smith', 60000.00, false, '1992-08-15', 2),

(3, 'Robert Brown', 70000.00, true, '1985-12-01', 3);

-- Insert into employee\_skill (many-to-many relationship)

INSERT INTO employee\_skill (employee\_id, skill\_id) VALUES

(1, 1), -- John knows Java

(1, 2), -- John knows SQL

(2, 2), -- Jane knows SQL

(3, 1), -- Robert knows Java

(3, 3); -- Robert knows Spring Boot

**OrmLearnApplication.java**

**package** com.cognizant.orm\_learn;

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

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

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

**import** org.springframework.boot.CommandLineRunner;

**import** org.springframework.boot.SpringApplication;

**import** org.springframework.boot.autoconfigure.SpringBootApplication;

**import** java.util.List;

@SpringBootApplication

**public** **class** OrmLearnApplication **implements** CommandLineRunner {

@Autowired

**private** EmployeeRepository employeeRepository;

**public** **static** **void** main(String[] args) {

SpringApplication.*run*(OrmLearnApplication.**class**, args);

}

@Override

**public** **void** run(String... args) {

System.***out***.println("=== ORM Learn Application Started ===");

List<Employee> employeeList = employeeRepository.findAll();

System.***out***.println("All Employees:");

employeeList.forEach(System.***out***::println);

}

}

**Department.java**

package com.cognizant.orm\_learn.model;

import jakarta.persistence.\*;

import java.util.List;

@Entity

@Table(name = "department")

public class Department {

@Id

private int id;

private String name;

@OneToMany(mappedBy = "department")

private List<Employee> employeeList;

// Getters and setters

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() { // This is used in Employee.toString()

return name;

}

public void setName(String name) {

this.name = name;

}

public List<Employee> getEmployeeList() {

return employeeList;

}

public void setEmployeeList(List<Employee> employeeList) {

this.employeeList = employeeList;

}

@Override

public String toString() {

return "Department{" +

"id=" + id +

", name='" + name + '\'' +

'}';

}

}

**Employee.java**

**package** com.cognizant.orm\_learn.model;

**import** jakarta.persistence.\*;

**import** java.util.Date;

**import** java.util.List;

@Entity

@Table(name = "employee")

**public** **class** Employee {

@Id

**private** **int** id;

**private** String name;

**private** **double** salary;

**private** **boolean** permanent;

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

@Temporal(TemporalType.***DATE***)

**private** Date dateOfBirth;

@ManyToOne

@JoinColumn(name = "dp\_id")

**private** Department department;

@ManyToMany

@JoinTable(

name = "employee\_skill",

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

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

)

**private** List<Skill> skillList;

// Getters and setters

**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;

}

// Overridden toString method for better console output

@Override

**public** String toString() {

**return** "Employee{" +

"id=" + id +

", name='" + name + '\'' +

", salary=" + salary +

", permanent=" + permanent +

", dateOfBirth=" + dateOfBirth +

", department=" + (department != **null** ? department.getName() : "N/A") +

'}';

}

}

**Skill.java**

**package** com.cognizant.orm\_learn.model;

**import** jakarta.persistence.\*;

**import** java.util.List;

@Entity

@Table(name = "skill")

**public** **class** Skill {

@Id

**private** **int** id;

**private** String name;

@ManyToMany(mappedBy = "skillList")

**private** List<Employee> employeeList;

// Getters and setters

**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** List<Employee> getEmployeeList() {

**return** employeeList;

}

**public** **void** setEmployeeList(List<Employee> employeeList) {

**this**.employeeList = employeeList;

}

@Override

**public** String toString() {

**return** "Skill{" +

"id=" + id +

", name='" + name + '\'' +

'}';

}

}

**DepartmentRepository.java**

**package** com.cognizant.orm\_learn.repository;

**import** com.cognizant.orm\_learn.model.Department;

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

**public** **interface** DepartmentRepository **extends** JpaRepository<Department, Integer> {}

**EmployeeRepository.java**

**package** com.cognizant.orm\_learn.repository;

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

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

**public** **interface** EmployeeRepository **extends** JpaRepository<Employee, Integer> {}

**SkillRepository.java**

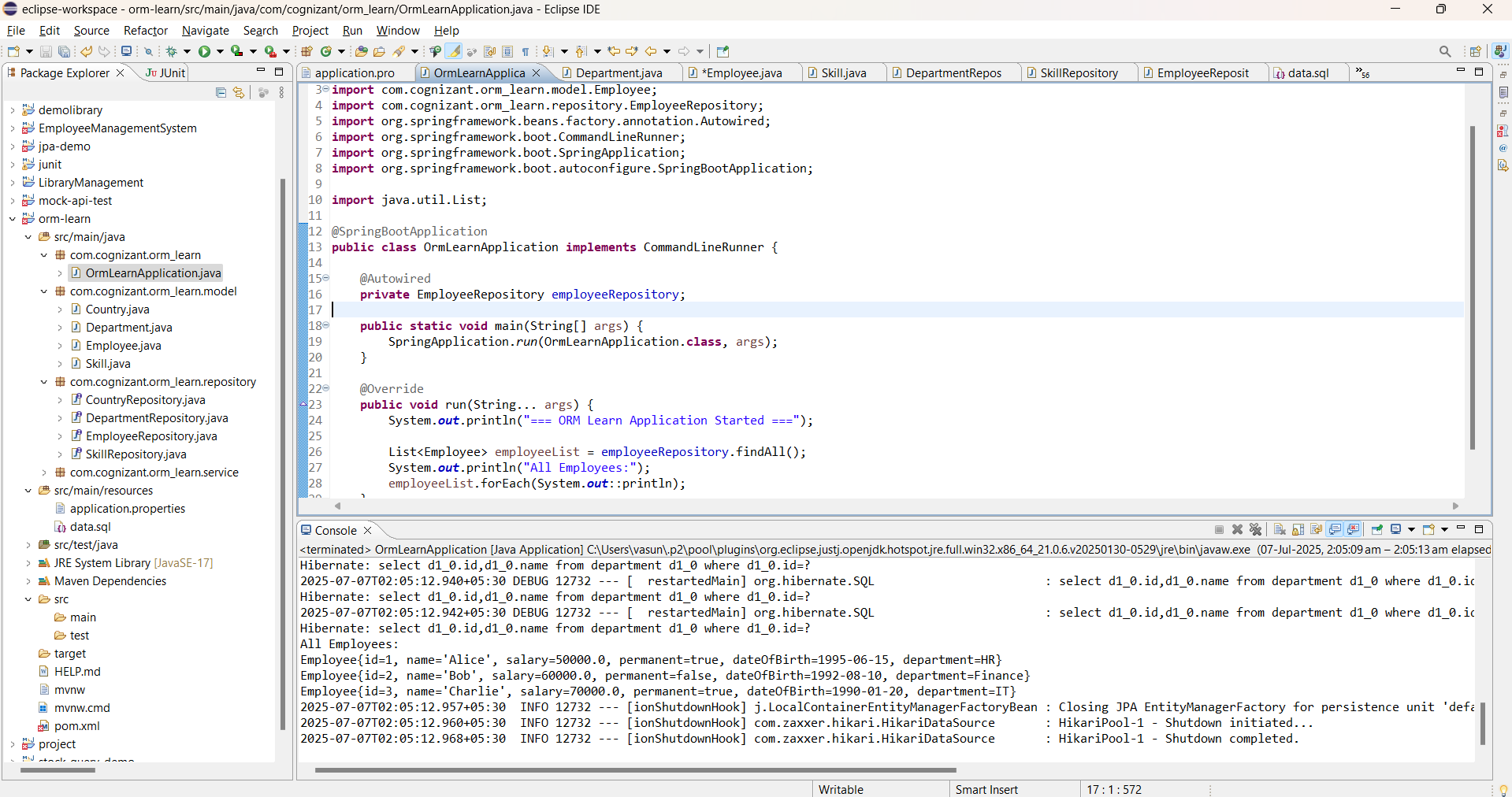
**package** com.cognizant.orm\_learn.repository;

**import** com.cognizant.orm\_learn.model.Skill;

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

**public** **interface** SkillRepository **extends** JpaRepository<Skill, Integer> {}

**OUTPUT**

****

**Hands on 4**

**Implement many to one relationship between Employee and Department**

**SOLUTION:**

**MySQL workbench:**

CREATE TABLE department (

id INT PRIMARY KEY,

name VARCHAR(100) NOT NULL

);

INSERT INTO department (id, name) VALUES

(1, 'HR'),

(2, 'IT'),

(3, 'Finance');

**Application.properties:**

# --- Application Name ---

spring.application.name=orm-learn

# --- Logging Levels ---

logging.level.org.springframework=info

logging.level.com.cognizant=debug

logging.level.org.hibernate.SQL=debug

logging.level.org.hibernate.type.descriptor.sql=trace

# --- SQL Log Formatting (Optional) ---

spring.jpa.properties.hibernate.format\_sql=true

# --- Log Output Pattern (Optional but helpful) ---

logging.pattern.console=%d{dd-MM-yy} %d{HH:mm:ss.SSS} %-20.20thread %5p %-25.25logger{25} %25M %4L %m%n

# --- Database Configuration ---

spring.datasource.url=jdbc:mysql://localhost:3306/ormlearn

spring.datasource.username=root

spring.datasource.password=Vasundhara@123

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

# --- Hibernate Settings ---

spring.jpa.hibernate.ddl-auto=update

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

# --- DevTools (Optional) ---

spring.devtools.restart.enabled=true

**Employee.java**

package com.cognizant.orm\_learn.model;

import jakarta.persistence.\*;

import java.util.Date;

@Entity

@Table(name = "employee")

public class Employee {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

private String name;

private double salary;

private boolean permanent;

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

private Date dateOfBirth;

@ManyToOne

@JoinColumn(name = "em\_dp\_id") // foreign key column in employee table

private Department department;

// Getters and Setters

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;

}

// toString

@Override

public String toString() {

return "Employee [id=" + id + ", name=" + name + ", salary=" + salary +

", permanent=" + permanent + ", dateOfBirth=" + dateOfBirth +

", department=" + department + "]";

}

}

**Department.java**

package com.cognizant.orm\_learn.model;

import jakarta.persistence.Entity;

import jakarta.persistence.Id;

import jakarta.persistence.Column;

import jakarta.persistence.Table;

@Entity

@Table(name = "department")

public class Department {

@Id

private int id;

@Column(name = "name")

private String name;

// Getters and Setters

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;

}

// toString

@Override

public String toString() {

return "Department [id=" + id + ", name=" + name + "]";

}

}

**DepartmentRepository.java**

package com.cognizant.orm\_learn.repository;

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

import org.springframework.stereotype.Repository;

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

@Repository

public interface DepartmentRepository extends JpaRepository<Department, Integer> {

// Custom query methods can be added here if needed

}

**EmployeeRepository.java**

package com.cognizant.orm\_learn.repository;

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

import org.springframework.stereotype.Repository;

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

@Repository

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

// You can add custom query methods here if needed

}

**EmployeeService.java**

package com.cognizant.orm\_learn.service;

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

import org.springframework.stereotype.Service;

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

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

import jakarta.transaction.Transactional;

@Service

public class EmployeeService {

@Autowired

private EmployeeRepository employeeRepository;

@Transactional

public Employee get(int id) {

return employeeRepository.findById(id).orElse(null); // safer than .get()

}

@Transactional

public void save(Employee employee) {

employeeRepository.save(employee);

}

}

**DepartmentService.java**

package com.cognizant.orm\_learn.service;

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

import org.springframework.stereotype.Service;

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

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

import jakarta.transaction.Transactional;

@Service

public class DepartmentService {

@Autowired

private DepartmentRepository departmentRepository;

@Transactional

public Department get(int id) {

return departmentRepository.findById(id).orElse(null); // Avoid .get() without checking

}

@Transactional

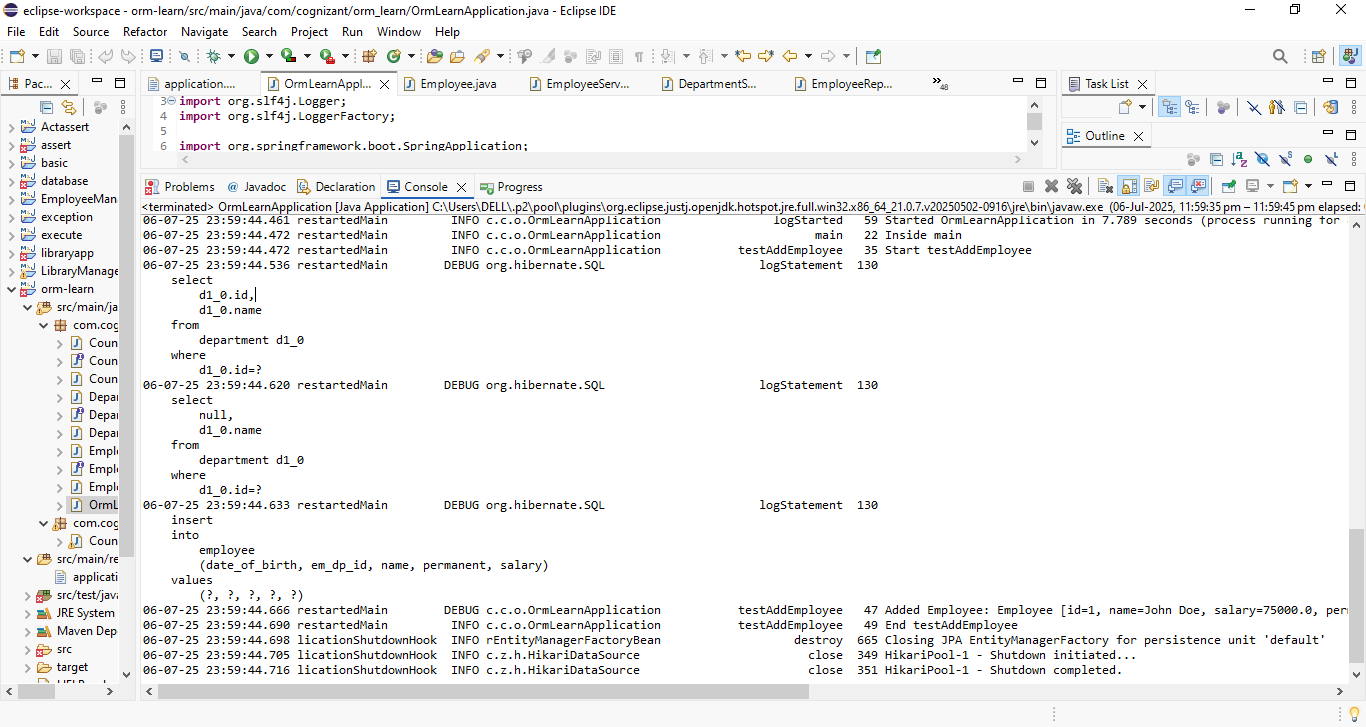
public void save(Department department) {

departmentRepository.save(department);

}

}

**OUTPUT:**



**Hands on 5**

**Implement one to many relationship between Employee and Department**

**SOLUTION:**

**Department.java**

package com.cognizant.orm\_learn.model;

import jakarta.persistence.\*;

import java.util.Set;

@Entity

public class Department {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

private String name;

// OneToMany mapping with Employee

@OneToMany(mappedBy = "department", fetch = FetchType.EAGER)

private Set<Employee> employeeList;

// --- Getters and Setters ---

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 Set<Employee> getEmployeeList() {

return employeeList;

}

public void setEmployeeList(Set<Employee> employeeList) {

this.employeeList = employeeList;

}

// --- toString() (optional for logging) ---

@Override

public String toString() {

return "Department [id=" + id + ", name=" + name + "]";

}

}

**Employee.java**

package com.cognizant.orm\_learn.model;

import jakarta.persistence.\*;

@Entity

public class Employee {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

private String name;

private double salary;

private boolean permanent;

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

private java.sql.Date dateOfBirth;

@ManyToOne

@JoinColumn(name = "em\_dp\_id")

private Department department;

// --- Getters and Setters ---

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 java.sql.Date getDateOfBirth() {

return dateOfBirth;

}

public void setDateOfBirth(java.sql.Date dateOfBirth) {

this.dateOfBirth = dateOfBirth;

}

public Department getDepartment() {

return department;

}

public void setDepartment(Department department) {

this.department = department;

}

@Override

public String toString() {

return "Employee [id=" + id + ", name=" + name + ", salary=" + salary + "]";

}

}

**OrmLearnApplication.java**

package com.cognizant.orm\_learn;

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.Employee;

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

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

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

import java.util.Date;

@SpringBootApplication

public class OrmLearnApplication {

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

private static EmployeeService employeeService;

private static DepartmentService departmentService;

public static void main(String[] args) {

ApplicationContext context = SpringApplication.run(OrmLearnApplication.class, args);

LOGGER.info("Inside main");

// Initialize services

employeeService = context.getBean(EmployeeService.class);

departmentService = context.getBean(DepartmentService.class);

// Run test methods one at a time:

testAddEmployee(); // ✅ Adds a new employee

// testGetEmployee(); // ✅ Fetch an employee and their department

// testUpdateEmployee(); // ✅ Update employee department

}

private static void testAddEmployee() {

LOGGER.info("Start testAddEmployee");

Employee employee = new Employee();

employee.setName("John Doe");

employee.setSalary(75000);

employee.setPermanent(true);

employee.setDateOfBirth(new java.sql.Date(new Date().getTime()));

Department department = departmentService.get(1); // Assuming department with ID 1 exists

employee.setDepartment(department);

employeeService.save(employee);

LOGGER.debug("Added Employee: {}", employee);

LOGGER.info("End testAddEmployee");

}

private static void testGetEmployee() {

LOGGER.info("Start testGetEmployee");

Employee employee = employeeService.get(1); // Assuming employee with ID 1 exists

LOGGER.debug("Employee: {}", employee);

LOGGER.debug("Department: {}", employee.getDepartment());

LOGGER.info("End testGetEmployee");

}

private static void testUpdateEmployee() {

LOGGER.info("Start testUpdateEmployee");

Employee employee = employeeService.get(1); // Get existing employee

Department newDepartment = departmentService.get(2); // Set to a different department

employee.setDepartment(newDepartment);

employeeService.save(employee);

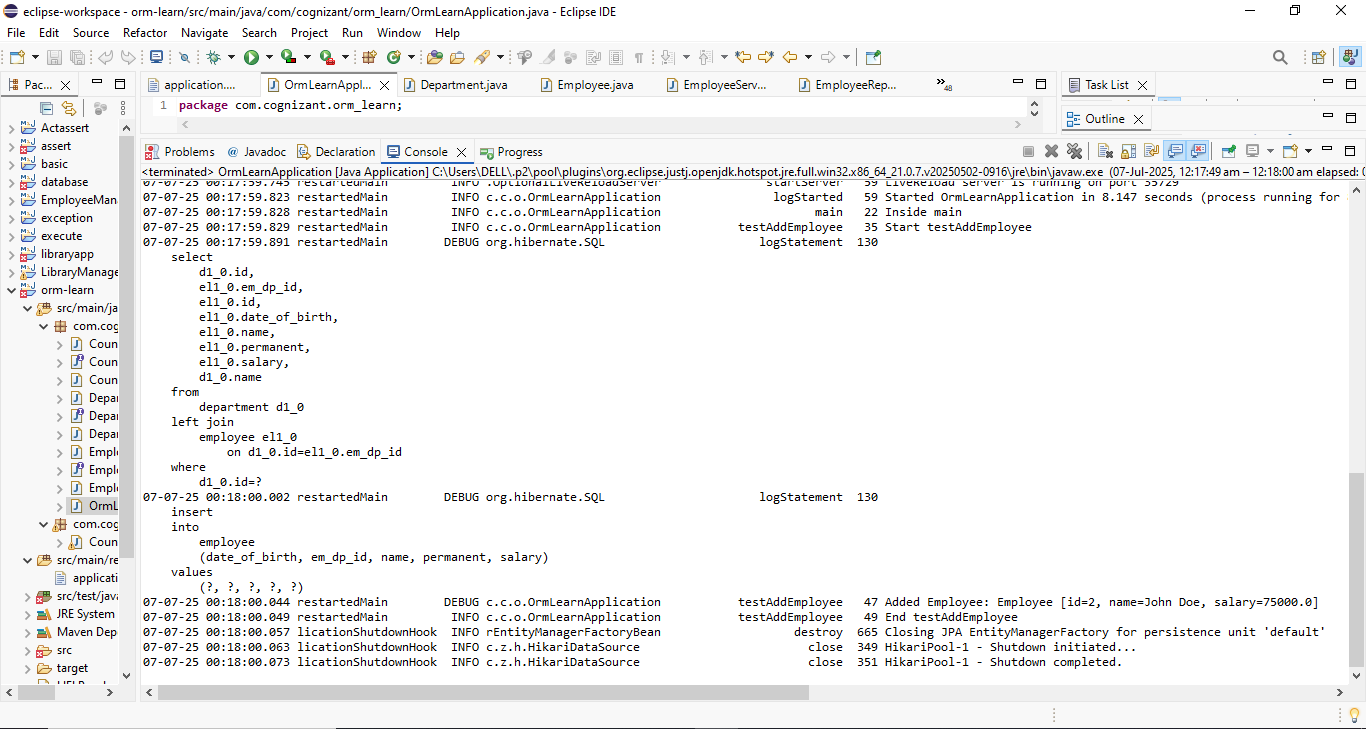
LOGGER.debug("Updated Employee: {}", employee);

LOGGER.info("End testUpdateEmployee");

}

}

**OUTPUT:**



**Hands on 6**

**Implement many to many relationship between Employee and Skill**

**SOLUTION:**

**MySQL WorkBench**

CREATE TABLE skill (

sk\_id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

sk\_name VARCHAR(100) NOT NULL

);

CREATE TABLE employee\_skill (

es\_em\_id INT NOT NULL,

es\_sk\_id INT NOT NULL,

PRIMARY KEY (es\_em\_id, es\_sk\_id),

FOREIGN KEY (es\_em\_id) REFERENCES employee(id),

FOREIGN KEY (es\_sk\_id) REFERENCES skill(sk\_id)

);

INSERT INTO skill (sk\_name) VALUES ('Java'), ('SQL'), ('Spring Boot');

**application.properties**

# --- Application Name ---

spring.application.name=orm-learn

# --- Logging Levels ---

logging.level.org.springframework=info

logging.level.com.cognizant=debug

logging.level.org.hibernate.SQL=debug

logging.level.org.hibernate.type.descriptor.sql=trace

# --- SQL Log Formatting ---

spring.jpa.properties.hibernate.format\_sql=true

# --- Console Log Output Format ---

logging.pattern.console=%d{dd-MM-yy} %d{HH:mm:ss.SSS} %-20.20thread %5p %-25.25logger{25} %25M %4L %m%n

# --- Database Configuration ---

spring.datasource.url=jdbc:mysql://localhost:3306/ormlearn

spring.datasource.username=root

spring.datasource.password=Vasundhara@123

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

# --- Hibernate Settings ---

spring.jpa.hibernate.ddl-auto=update

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

# --- Run as Console App (No Web Server) ---

spring.main.web-application-type=none

# --- Optional DevTools Restart ---

spring.devtools.restart.enabled=true

**Employee.java**

package com.cognizant.orm\_learn.model;

import jakarta.persistence.\*;

import java.sql.Date;

import java.util.HashSet;

import java.util.Set;

@Entity

public class Employee {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

private String name;

private double salary;

private boolean permanent;

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

private Date dateOfBirth;

@ManyToOne

@JoinColumn(name = "em\_dp\_id")

private Department department;

@ManyToMany(fetch = FetchType.EAGER)

@JoinTable(

name = "employee\_skill",

joinColumns = @JoinColumn(name = "es\_em\_id"),

inverseJoinColumns = @JoinColumn(name = "es\_sk\_id")

)

private Set<Skill> skillList = new HashSet<>();

// Getters and setters

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 Set<Skill> getSkillList() {

return skillList;

}

public void setSkillList(Set<Skill> skillList) {

this.skillList = skillList;

}

@Override

public String toString() {

return "Employee [id=" + id + ", name=" + name + ", salary=" + salary + "]";

}

}

**Skill.java**

package com.cognizant.orm\_learn.model;

import jakarta.persistence.\*;

import java.util.Set;

@Entity

public class Skill {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name = "sk\_id")

private int id;

@Column(name = "sk\_name")

private String name;

@ManyToMany(mappedBy = "skillList")

private Set<Employee> employeeList;

// Getters and setters

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 Set<Employee> getEmployeeList() {

return employeeList;

}

public void setEmployeeList(Set<Employee> employeeList) {

this.employeeList = employeeList;

}

@Override

public String toString() {

return "Skill [id=" + id + ", name=" + name + "]";

}

}

**OrmLearnApplication.java**

package com.cognizant.orm\_learn;

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

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

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

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

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

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

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.sql.Date;

import java.util.HashSet;

import java.util.Set;

@SpringBootApplication

public class OrmLearnApplication {

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

private static EmployeeService employeeService;

private static DepartmentService departmentService;

private static SkillService skillService;

public static void main(String[] args) {

ApplicationContext context = SpringApplication.run(OrmLearnApplication.class, args);

LOGGER.info("Inside main");

employeeService = context.getBean(EmployeeService.class);

departmentService = context.getBean(DepartmentService.class);

skillService = context.getBean(SkillService.class);

// Only run one test at a time for clarity

// testAddEmployee();

// testGetEmployee();

// testUpdateEmployee();

// testGetDepartment();

testAddSkillToEmployee();

}

private static void testAddEmployee() {

LOGGER.info("Start testAddEmployee");

Employee employee = new Employee();

employee.setName("John Doe");

employee.setSalary(75000);

employee.setPermanent(true);

employee.setDateOfBirth(Date.valueOf("1990-05-15"));

Department department = departmentService.get(1); // Make sure department with id=1 exists

employee.setDepartment(department);

employeeService.save(employee);

LOGGER.debug("Added Employee: {}", employee);

LOGGER.info("End testAddEmployee");

}

private static void testGetEmployee() {

LOGGER.info("Start testGetEmployee");

Employee employee = employeeService.get(1); // Make sure employee with id=1 exists

LOGGER.debug("Employee: {}", employee);

LOGGER.debug("Department: {}", employee.getDepartment());

LOGGER.debug("Skills: {}", employee.getSkillList());

LOGGER.info("End testGetEmployee");

}

private static void testUpdateEmployee() {

LOGGER.info("Start testUpdateEmployee");

Employee employee = employeeService.get(1);

Department newDept = departmentService.get(2); // Change to another department

employee.setDepartment(newDept);

employeeService.save(employee);

LOGGER.debug("Updated Employee: {}", employee);

LOGGER.info("End testUpdateEmployee");

}

private static void testGetDepartment() {

LOGGER.info("Start testGetDepartment");

Department dept = departmentService.get(1); // Make sure this department exists

if (dept != null) {

LOGGER.debug("Department: {}", dept);

LOGGER.debug("Employees: {}", dept.getEmployeeList());

} else {

LOGGER.warn("Department with ID 1 not found.");

}

LOGGER.info("End testGetDepartment");

}

private static void testAddSkillToEmployee() {

LOGGER.info("Start testAddSkillToEmployee");

Employee employee = employeeService.get(1); // Replace with actual employee ID

Skill skill = skillService.get(2); // Replace with actual skill ID

if (employee != null && skill != null) {

Set<Skill> skills = employee.getSkillList();

if (skills == null) {

skills = new HashSet<>();

employee.setSkillList(skills);

}

skills.add(skill);

employeeService.save(employee);

LOGGER.debug("Skill added to Employee: {}", skill);

} else {

LOGGER.warn("Employee or Skill not found.");

}

LOGGER.info("End testAddSkillToEmployee");

}

}

**SkillService.java**

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.Skill;

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

@Service

public class SkillService {

@Autowired

private SkillRepository skillRepository;

@Transactional

public Skill get(int id) {

return skillRepository.findById(id).orElse(null);

}

}

**SkillRepository.java**

package com.cognizant.orm\_learn.repository;

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

import org.springframework.stereotype.Repository;

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

@Repository

public interface SkillRepository extends JpaRepository<Skill, Integer> {

}

**DepartmentRepository.java**

package com.cognizant.orm\_learn.repository;

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

import org.springframework.stereotype.Repository;

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

@Repository

public interface DepartmentRepository extends JpaRepository<Department, Integer> {

}

**Department.java**

package com.cognizant.orm\_learn.model;

import java.util.List;

import jakarta.persistence.\*;

@Entity

@Table(name = "department")

public class Department {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name = "dp\_id")

private int id;

@Column(name = "dp\_name")

private String name;

@OneToMany(mappedBy = "department", fetch = FetchType.EAGER)

private List<Employee> employeeList;

// getters, setters, toString()

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 List<Employee> getEmployeeList() {

return employeeList;

}

public void setEmployeeList(List<Employee> employeeList) {

this.employeeList = employeeList;

}

@Override

public String toString() {

return "Department [id=" + id + ", name=" + name + "]";

}

}

**DepartmentService.java**

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.Department;

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

@Service

public class DepartmentService {

@Autowired

private DepartmentRepository departmentRepository;

@Transactional

public Department get(int id) {

return departmentRepository.findById(id).orElse(null);

}

}

**EmployeeService.java**

package com.cognizant.orm\_learn.service;

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

import org.springframework.stereotype.Service;

import jakarta.transaction.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 Employee get(int id) {

return employeeRepository.findById(id).orElse(null);

}

@Transactional

public void save(Employee employee) {

employeeRepository.save(employee);

}

}

**EmployeeRepository.java**

package com.cognizant.orm\_learn.repository;

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

import org.springframework.stereotype.Repository;

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

@Repository

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

**OUTPUT:**

