# SPRING\_DATA\_JPA

#### Hands on 1-Write queries on country table using Query Method

Country.java

package com.in28minutes.jpa.hibernate.demo.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;

}

@Override

public String toString() { return code + " " + name;

}

}

CountryRepository.java

package com.in28minutes.jpa.hibernate.demo.repository; import com.in28minutes.jpa.hibernate.demo.entity.Country; import org.springframework.data.jpa.repository.JpaRepository; import org.springframework.stereotype.Repository;

import java.util.List;

@Repository

public interface CountryRepository extends JpaRepository<Country, String> {

// Search country names containing substring

List<Country> findByNameContaining(String keyword);

// Search country names containing substring sorted alphabetically

List<Country> findByNameContainingOrderByNameAsc(String keyword);

// Find countries starting with a given letter

List<Country> findByNameStartingWith(String prefix);

}

#### OrmLearnApplication.java

package com.in28minutes.jpa.hibernate.demo;

import com.in28minutes.jpa.hibernate.demo.entity.Country;

import com.in28minutes.jpa.hibernate.demo.repository.CountryRepository; 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 CountryRepository countryRepository;

public static void main(String[] args) {

SpringApplication.run(OrmLearnApplication.class, args);

}

@Override

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

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

List<Country> countriesWithOu = countryRepository.findByNameContaining("ou"); countriesWithOu.forEach(System.out::println);

System.out.println("=== Countries containing 'ou' sorted ==="); List<Country> sortedCountriesWithOu =

countryRepository.findByNameContainingOrderByNameAsc("ou"); sortedCountriesWithOu.forEach(System.out::println);

System.out.println("=== Countries starting with 'Z' ===");

List<Country> countriesStartingWithZ = countryRepository.findByNameStartingWith("Z"); countriesStartingWithZ.forEach(System.out::println);

}

}

# SQL Table

CREATE TABLE country (

code VARCHAR(2) PRIMARY KEY, name VARCHAR(255) NOT NULL

);

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

('DJ', 'Djibouti'),

('GP', 'Guadeloupe'),

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

('SS', 'South Sudan'),

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

('UM', 'United States Minor Outlying Islands'), ('ZA', 'South Africa'),

('ZM', 'Zambia'),

('ZW', 'Zimbabwe');

Hands on 2-Write queries on stock table using Query Methods

### Stock.java

package com.in28minutes.jpa.hibernate.demo.entity;

import jakarta.persistence.\*; import java.math.BigDecimal; import java.time.LocalDate;

@Entity

@Table(name = "stock") public class Stock {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY) @Column(name = "st\_id")

private Integer 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;

}

// Getters and Setters

@Override

public String toString() {

return code + " | " + date + " | Open: " + open + ", Close: " + close + ", Volume: " + volume;

}

}

StockRepository.java

package com.in28minutes.jpa.hibernate.demo.repository;

import com.in28minutes.jpa.hibernate.demo.entity.Stock; import org.springframework.data.jpa.repository.JpaRepository; import org.springframework.stereotype.Repository;

import java.math.BigDecimal; import java.time.LocalDate; import java.util.List;

@Repository

public interface StockRepository extends JpaRepository<Stock, Integer> {

// Get all stock details of Facebook in September 2019

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

// Get all Google stock details where closing price > 1250

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

// Get top 3 stocks by volume descending List<Stock> findTop3ByOrderByVolumeDesc();

// Get 3 lowest Netflix closing stocks

List<Stock> findTop3ByCodeOrderByCloseAsc(String code);

}

## OrmLearnApplication.java

package com.in28minutes.jpa.hibernate.demo;

import com.in28minutes.jpa.hibernate.demo.entity.Stock;

import com.in28minutes.jpa.hibernate.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.math.BigDecimal; import java.time.LocalDate; import java.util.List;

@SpringBootApplication

public class OrmLearnApplication implements CommandLineRunner {

@Autowired

private StockRepository stockRepository;

public static void main(String[] args) {

SpringApplication.run(OrmLearnApplication.class, args);

}

@Override

public void run(String... args) {

System.out.println("\n--- FB Stocks in Sep 2019 ---"); LocalDate start = LocalDate.of(2019, 9, 1);

LocalDate end = LocalDate.of(2019, 9, 30);

List<Stock> fbSeptStocks = stockRepository.findByCodeAndDateBetween("FB", start, end);

fbSeptStocks.forEach(System.out::println);

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

List<Stock> googleStocks = stockRepository.findByCodeAndCloseGreaterThan("GOOGL", new BigDecimal("1250"));

googleStocks.forEach(System.out::println);

System.out.println("\n--- Top 3 Stocks with Highest Volume ---");

List<Stock> topVolumeStocks = stockRepository.findTop3ByOrderByVolumeDesc(); topVolumeStocks.forEach(System.out::println);

System.out.println("\n--- Netflix's 3 Lowest Closing Stocks ---"); List<Stock> lowNetflixStocks =

stockRepository.findTop3ByCodeOrderByCloseAsc("NFLX"); lowNetflixStocks.forEach(System.out::println);

}

}

## Hands on 3-Create payroll tables and bean mapping

Employee.java

package com.cognizant.ormlearn.model;

import java.util.Date; import java.util.List;

import jakarta.persistence.\*;

@Entity

@Table(name = "employee") public class Employee {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY) private int id;

@Column(name = "name") private String name;

@Column(name = "salary") private double salary;

@Column(name = "permanent") 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;

// 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; }

@Override

public String toString() {

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

}

}

## Department.java

package com.cognizant.ormlearn.model;

import jakarta.persistence.\*; import java.util.List;

@Entity

@Table(name = "department") public class Department {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY) private int id;

@Column(name = "name") private String name;

@OneToMany(mappedBy = "department", cascade = CascadeType.ALL) 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 "Department [id=" + id + ", name=" + name + "]";

}

}

## Skill.java

package com.cognizant.ormlearn.model;

import jakarta.persistence.\*; import java.util.List;

@Entity @Table(name = "skill") public class Skill {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY) private int id;

@Column(name = "name") 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 + "]";

}

}

## EmployeeRepository.java

package com.cognizant.ormlearn.repository;

import org.springframework.data.jpa.repository.JpaRepository; import com.cognizant.ormlearn.model.Employee;

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

DepartmentRepository.java

package com.cognizant.ormlearn.repository;

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

import com.cognizant.ormlearn.model.Department;

public interface DepartmentRepository extends JpaRepository<Department, Integer> {

}

## SkillRepository.java

package com.cognizant.ormlearn.repository;

import org.springframework.data.jpa.repository.JpaRepository; import com.cognizant.ormlearn.model.Skill;

public interface SkillRepository extends JpaRepository<Skill, Integer> {

}

## OrmLearnApplication.java

package com.cognizant.ormlearn;

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.ormlearn.model.Employee;

import com.cognizant.ormlearn.repository.EmployeeRepository;

@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) {

Employee emp = employeeRepository.findById(1).orElse(null); if (emp != null) {

System.out.println("Employee: " + emp);

System.out.println("Department: " + emp.getDepartment()); System.out.println("Skills: " + emp.getSkillList());

} else {

System.out.println("Employee with ID 1 not found.");

}

}

}

Hands on 4-Implement many to one relationship between Employee and Department

### Employee.java

package com.cognizant.ormlearn.model;

import jakarta.persistence.\*; import java.util.Date;

@Entity

@Table(name = "employee") public class Employee {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY) @Column(name = "em\_id")

private int id;

@Column(name = "em\_name") private String name;

@Column(name = "em\_salary") private double salary;

@Column(name = "em\_permanent") private boolean permanent;

@Column(name = "em\_date\_of\_birth") private 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 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;

}

@Override

public String toString() {

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

}

}

## Department.java

package com.cognizant.ormlearn.model;

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;

// 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;

}

@Override

public String toString() {

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

}

}

## EmployeeRepository.java

package com.cognizant.ormlearn.repository;

import org.springframework.data.jpa.repository.JpaRepository; import com.cognizant.ormlearn.model.Employee;

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

DepartmentRepository.java

package com.cognizant.ormlearn.repository;

import org.springframework.data.jpa.repository.JpaRepository; import com.cognizant.ormlearn.model.Department;

public interface DepartmentRepository extends JpaRepository<Department, Integer> {

}

#### EmployeeService.java

package com.cognizant.ormlearn.service;

import com.cognizant.ormlearn.model.Employee;

import com.cognizant.ormlearn.repository.EmployeeRepository; import jakarta.transaction.Transactional;

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

import org.springframework.stereotype.Service; import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

@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.ormlearn.service;

import com.cognizant.ormlearn.model.Department;

import com.cognizant.ormlearn.repository.DepartmentRepository; import jakarta.transaction.Transactional;

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

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

@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");

}

}

## OrmLearnApplication.java

package com.cognizant.ormlearn;

import com.cognizant.ormlearn.model.Employee; import com.cognizant.ormlearn.model.Department;

import com.cognizant.ormlearn.service.EmployeeService;

import com.cognizant.ormlearn.service.DepartmentService;

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;

@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); employeeService = context.getBean(EmployeeService.class);

departmentService = context.getBean(DepartmentService.class);

// testGetEmployee();

// testAddEmployee(); testUpdateEmployee();

}

private static void testGetEmployee() { LOGGER.info("Start");

Employee employee = employeeService.get(1); LOGGER.debug("Employee: {}", employee);

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

LOGGER.info("End");

}

private static void testAddEmployee() { LOGGER.info("Start");

Employee employee = new Employee(); employee.setName("Ravi Kumar"); employee.setSalary(70000); employee.setPermanent(true);

employee.setDateOfBirth(Date.valueOf("1998-05-20"));

Department department = departmentService.get(1); employee.setDepartment(department);

employeeService.save(employee);

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

}

private static void testUpdateEmployee() { LOGGER.info("Start");

Employee employee = employeeService.get(1);

Department newDept = departmentService.get(2); employee.setDepartment(newDept);

employeeService.save(employee);

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

}

}

Hands on 5-Implement one to many relationship between Employee and Department

Department.java

package com.cognizant.ormlearn.model; import jakarta.persistence.\*;

import java.util.Set;

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

}

}

## Employee.java

package com.cognizant.ormlearn.model;

import jakarta.persistence.\*; import java.util.Date;

@Entity

@Table(name = "employee") public class Employee {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY) @Column(name = "em\_id")

private int id;

@Column(name = "em\_name") private String name;

@Column(name = "em\_salary") private double salary;

@Column(name = "em\_permanent") private boolean permanent;

@Column(name = "em\_date\_of\_birth") private 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 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;

}

@Override

public String toString() {

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

}

}

## DepartmentRepository.java

package com.cognizant.ormlearn.repository;

import org.springframework.data.jpa.repository.JpaRepository; import com.cognizant.ormlearn.model.Department;

public interface DepartmentRepository extends JpaRepository<Department, Integer> {

}

EmployeeRepository.java

package com.cognizant.ormlearn.repository;

import org.springframework.data.jpa.repository.JpaRepository; import com.cognizant.ormlearn.model.Employee;

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

## DepartmentService.java

package com.cognizant.ormlearn.service;

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

import org.springframework.transaction.annotation.Transactional;

import com.cognizant.ormlearn.model.Department;

import com.cognizant.ormlearn.repository.DepartmentRepository;

@Service

public class DepartmentService {

@Autowired

private DepartmentRepository departmentRepository;

@Transactional

public Department get(int id) {

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

}

@Transactional

public void save(Department department) { departmentRepository.save(department);

}

}

## EmployeeService.java

package com.cognizant.ormlearn.service;

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

import org.springframework.transaction.annotation.Transactional;

import com.cognizant.ormlearn.model.Employee;

import com.cognizant.ormlearn.repository.EmployeeRepository;

@Service

public class EmployeeService {

@Autowired

private EmployeeRepository employeeRepository;

@Transactional

public Employee get(int id) {

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

}

@Transactional

public void save(Employee employee) { employeeRepository.save(employee);

}

}

## OrmLearnApplication.java

package com.cognizant.ormlearn;

import com.cognizant.ormlearn.model.Department;

import com.cognizant.ormlearn.service.DepartmentService; import com.cognizant.ormlearn.service.EmployeeService; import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication; import org.springframework.context.ApplicationContext;

@SpringBootApplication

public class OrmLearnApplication {

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

private static DepartmentService departmentService; private static EmployeeService employeeService;

public static void main(String[] args) {

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

employeeService = context.getBean(EmployeeService.class);

// testGetEmployee(); // Previously used

testGetDepartment(); // Only run this for now

}

private static void testGetDepartment() { LOGGER.info("Start");

Department department = departmentService.get(1); // Ensure ID 1 has multiple employees

LOGGER.debug("Department: {}", department); LOGGER.debug("Employee List: {}", department.getEmployeeList()); LOGGER.info("End");

}

}

## Hands on 6-Implement many to many relationship between Employee and Skill

Skill.java

package com.cognizant.ormlearn.model;

import jakarta.persistence.\*; import java.util.Set;

@Entity @Table(name = "skill") 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 + "]";

}

}

## Employee.java

// Add these imports import java.util.Set;

import jakarta.persistence.FetchType;

import jakarta.persistence.JoinTable; import jakarta.persistence.JoinColumn; import jakarta.persistence.ManyToMany;

// Inside the Employee class:

@ManyToMany(fetch = FetchType.EAGER) @JoinTable(

name = "employee\_skill",

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

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

)

private Set<Skill> skillList;

public Set<Skill> getSkillList() { return skillList;

}

public void setSkillList(Set<Skill> skillList) { this.skillList = skillList;

}

// Make sure other existing code stays as is.

## SkillRepository.java

package com.cognizant.ormlearn.repository;

import org.springframework.data.jpa.repository.JpaRepository; import com.cognizant.ormlearn.model.Skill;

public interface SkillRepository extends JpaRepository<Skill, Integer> {

}

## OrmLearnApplication.java

package com.cognizant.ormlearn;

import com.cognizant.ormlearn.model.Employee; import com.cognizant.ormlearn.model.Skill;

import com.cognizant.ormlearn.service.DepartmentService; import com.cognizant.ormlearn.service.EmployeeService; import com.cognizant.ormlearn.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;

@SpringBootApplication

public class OrmLearnApplication {

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

private static DepartmentService departmentService; private static EmployeeService employeeService; private static SkillService skillService;

public static void main(String[] args) {

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

employeeService = context.getBean(EmployeeService.class); skillService = context.getBean(SkillService.class);

// Comment other tests and run this one

// testGetDepartment();

// testGetEmployee(); testAddSkillToEmployee();

}

private static void testGetEmployee() { LOGGER.info("Start");

Employee employee = employeeService.get(1); // use an employee ID that exists LOGGER.debug("Employee: {}", employee);

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

LOGGER.info("End");

}

private static void testAddSkillToEmployee() { LOGGER.info("Start");

int employeeId = 2; // Use valid employee id

int skillId = 3; // Use valid skill id not already assigned to employee

Employee employee = employeeService.get(employeeId); Skill skill = skillService.get(skillId);

// Add skill to employee's skill set employee.getSkillList().add(skill);

// Save employee to update relationship table employee\_skill employeeService.save(employee);

LOGGER.debug("Added skill {} to employee {}", skill.getName(), employee.getName());

LOGGER.info("End");

}

}

## SkillService.java

package com.cognizant.ormlearn.service;

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

import org.springframework.transaction.annotation.Transactional;

import com.cognizant.ormlearn.model.Skill;

import com.cognizant.ormlearn.repository.SkillRepository;

@Service

public class SkillService {

@Autowired

private SkillRepository skillRepository;

@Transactional

public Skill get(int id) {

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

}

@Transactional

public void save(Skill skill) { skillRepository.save(skill);

}

}