**WEEK 3 SPRING DATA JPA HANDSON**

**Superset ID : 6386707**

**Handson 1**

**Country Entity**

@Entity

@Table(name = "country")

public class Country {

@Id

@Column(name = "co\_code")

private String code;

@Column(name = "co\_name")

private String name;

}

**CountryRepository.java**

@Repository

public interface CountryRepository extends JpaRepository<Country, String> {

List<Country> findByNameContainingIgnoreCase(String name);

List<Country> findByNameContainingIgnoreCaseOrderByNameAsc(String name);

List<Country> findByNameStartingWithIgnoreCase(String prefix);

}

**CountryService.java**

@Service

public class CountryService {

@Autowired

private CountryRepository countryRepository;

@Transactional

public List<Country> searchCountriesByPartialName(String partialName) {

return countryRepository.findByNameContainingIgnoreCase(partialName);

}

@Transactional

public List<Country> searchCountriesByPartialNameSorted(String partialName) {

return countryRepository.findByNameContainingIgnoreCaseOrderByNameAsc(partialName);

}

@Transactional

public List<Country> getCountriesStartingWith(String prefix) {

return countryRepository.findByNameStartingWithIgnoreCase(prefix);

}

}

**OrmLearnApplication.java**

private static void testSearchByPartialName() {

LOGGER.info("Start: testSearchByPartialName");

List<Country> countries = countryService.searchCountriesByPartialName("ou");

countries.forEach(country -> LOGGER.debug("{}", country));

LOGGER.info("End: testSearchByPartialName");

}

private static void testSearchByPartialNameSorted() {

LOGGER.info("Start: testSearchByPartialNameSorted");

List<Country> countries = countryService.searchCountriesByPartialNameSorted("ou");

countries.forEach(country -> LOGGER.debug("{}", country));

LOGGER.info("End: testSearchByPartialNameSorted");

}

private static void testCountriesStartingWithLetter() {

LOGGER.info("Start: testCountriesStartingWithLetter");

List<Country> countries = countryService.getCountriesStartingWith("Z");

countries.forEach(country -> LOGGER.debug("{}", country));

LOGGER.info("End: testCountriesStartingWithLetter");

}

**Main()**

public static void main(String[] args) {

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

countryService = context.getBean(CountryService.class);

LOGGER.info("Inside main");

testSearchByPartialName();

testSearchByPartialNameSorted();

testCountriesStartingWithLetter();

}

**Handson 2**

**Stock Entity**

package com.cognizant.ormlearn.model;

import jakarta.persistence.\*;

import java.util.Date;

@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")

@Temporal(TemporalType.DATE)

private Date date;

@Column(name = "st\_open")

private Float open;

@Column(name = "st\_close")

private Float close;

@Column(name = "st\_volume")

private Long volume;

}

**StockRepository.java**

package com.cognizant.ormlearn.repository;

import com.cognizant.ormlearn.model.Stock;

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

import org.springframework.stereotype.Repository;

import java.util.Date;

import java.util.List;

@Repository

public interface StockRepository extends JpaRepository<Stock, Integer> {

List<Stock> findByCodeAndDateBetween(String code, Date startDate, Date endDate);

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

List<Stock> findTop3ByOrderByVolumeDesc();

List<Stock> findTop3ByCodeOrderByCloseAsc(String code);

}

**OrmLearnApplication.java**

@Autowired

private static StockRepository stockRepository;

private static void testFacebookStocksInSep2019() throws Exception {

LOGGER.info("Start: testFacebookStocksInSep2019");

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");

Date startDate = sdf.parse("2019-09-01");

Date endDate = sdf.parse("2019-09-30");

List<Stock> stocks = stockRepository.findByCodeAndDateBetween("FB", startDate, endDate);

stocks.forEach(stock -> LOGGER.debug("{}", stock));

LOGGER.info("End");

}

private static void testGoogleStockGreaterThan1250() {

LOGGER.info("Start: testGoogleStockGreaterThan1250");

List<Stock> stocks = stockRepository.findByCodeAndCloseGreaterThan("GOOGL", 1250f);

stocks.forEach(stock -> LOGGER.debug("{}", stock));

LOGGER.info("End");

}

private static void testTop3ByVolume() {

LOGGER.info("Start: testTop3ByVolume");

List<Stock> stocks = stockRepository.findTop3ByOrderByVolumeDesc();

stocks.forEach(stock -> LOGGER.debug("{}", stock));

LOGGER.info("End");

}

private static void testNetflixLowest() {

LOGGER.info("Start: testNetflixLowest");

List<Stock> stocks = stockRepository.findTop3ByCodeOrderByCloseAsc("NFLX");

stocks.forEach(stock -> LOGGER.debug("{}", stock));

LOGGER.info("End");

}

**Main()**

public static void main(String[] args) {

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

stockRepository = context.getBean(StockRepository.class);

testStockQueries(); // Call your query test method

}

**Handson 3**

**Employee.java**

package com.cognizant.ormlearn.model;

import jakarta.persistence.\*;

import java.util.Date;

import java.util.List;

@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")

@Temporal(TemporalType.DATE)

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 List<Skill> skillList;

}

**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)

@Column(name = "dp\_id")

private int id;

@Column(name = "dp\_name")

private String name;

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

private List<Employee> employeeList;

}

**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)

@Column(name = "sk\_id")

private int id;

@Column(name = "sk\_name")

private String name;

@ManyToMany(mappedBy = "skillList")

private List<Employee> employeeList;

}

**EmployeeRepository.java**

package com.cognizant.ormlearn.repository;

import com.cognizant.ormlearn.model.Employee;

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

import org.springframework.stereotype.Repository;

@Repository

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

**DepartmentRepository.java**

package com.cognizant.ormlearn.repository;

import com.cognizant.ormlearn.model.Department;

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

import org.springframework.stereotype.Repository;

@Repository

public interface DepartmentRepository extends JpaRepository<Department, Integer> {

}

**SkillRepository.java**

package com.cognizant.ormlearn.repository;

import com.cognizant.ormlearn.model.Skill;

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

import org.springframework.stereotype.Repository;

@Repository

public interface SkillRepository extends JpaRepository<Skill, Integer> {

}

**Handson 4**

**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;

@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);

}

}

**Employee.java**

@ManyToOne

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

private Department department;

public Department getDepartment() {

return department;

}

public void setDepartment(Department department) {

this.department = department;

}

**OrmLearnApplication.java**

private static EmployeeService employeeService;

private static DepartmentService departmentService;

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

employeeService = context.getBean(EmployeeService.class);

departmentService = context.getBean(DepartmentService.class);

**TestGetEmployee()**

private static void testGetEmployee() {

LOGGER.info("Start");

Employee employee = employeeService.get(1);

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

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

LOGGER.info("End");

}

**TestAddEmployee()**

private static void testAddEmployee() {

LOGGER.info("Start");

Employee employee = new Employee();

employee.setName("Leona");

employee.setSalary(52000.00);

employee.setPermanent(true);

employee.setDateOfBirth(Date.valueOf("2000-05-20"));

Department dept = departmentService.get(1);

employee.setDepartment(dept);

employeeService.save(employee);

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

LOGGER.info("End");

}

**testUpdateEmployee()**

private static void testUpdateEmployee() {

LOGGER.info("Start");

Employee employee = employeeService.get(3);

Department newDept = departmentService.get(2);

employee.setDepartment(newDept);

employeeService.save(employee);

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

LOGGER.info("End");

}

**Handson 5**

**Modify Department.java**

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

private Set<Employee> employeeList;

public Set<Employee> getEmployeeList() {

return employeeList;

}

public void setEmployeeList(Set<Employee> employeeList) {

this.employeeList = employeeList;

}

**OrmLearnApplication.java**

private static void testGetDepartment() {

LOGGER.info("Start");

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

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

LOGGER.debug("Employees in Department: {}", department.getEmployeeList());

LOGGER.info("End");

}

**Hibernate Query**

select department0\_.dp\_id as dp\_id1\_1\_0\_,

department0\_.dp\_name as dp\_name2\_1\_0\_,

employees1\_.em\_dp\_id as em\_dp\_id6\_2\_1\_,

employees1\_.em\_id as em\_id1\_2\_1\_,

from department department0\_

left outer join employee employees1\_

on department0\_.dp\_id=employees1\_.em\_dp\_id

where department0\_.dp\_id=?

**Handson 6**

**Employee.java**

@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;

}

**Skill.java**

@ManyToMany(mappedBy = "skillList")

private Set<Employee> employeeList;

public Set<Employee> getEmployeeList() {

return employeeList;

}

public void setEmployeeList(Set<Employee> employeeList) {

this.employeeList = employeeList;

}

**OrmLearnApplication.java**

private static void testGetEmployee() {

LOGGER.info("Start");

Employee employee = employeeService.get(1);

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

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

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

LOGGER.info("End");

}

**testAddSkillToEmployee()**

private static void testAddSkillToEmployee() {

LOGGER.info("Start");

Employee employee = employeeService.get(1);

Skill skill = skillService.get(2);

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

skills.add(skill);

employee.setSkillList(skills);

employeeService.save(employee);

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

LOGGER.info("End");

}