SPRING DATA JPA HANDS ON 2

Superset ID : 6384831

Name : Mohana Priya N

E-mail : mohanapriya.2205056@srec.ac.in

1) Hands on 1: Write queries on country table using Query Methods

Solution:

//countryRepository.java

```
package com.cognizant.ormlearn.repository;
import java.util.List;
import org.springframework.data.jpa.repository.JpaRepository;
import com.cognizant.ormlearn.model.Country;
public interface CountryRepository extends JpaRepository<Country, String> {
   List<Country> findByNameContaining(String search);
   List<Country> findByNameContainingOrderByNameAsc(String search);
   List<Country> findByNameStartingWith(String letter);
}
```

//OrmLearnApplication.java

```
private static void testFindCountriesBySearch() {
   LOGGER.info("Start - find countries containing 'ou'");
   List<Country> countries = countryService.findByNameContaining("ou");
   countries.forEach(c -> LOGGER.info(c.toString()));
   LOGGER.info("End");
}
private static void testFindCountriesBySearchOrdered() {
   LOGGER.info("Start - find countries containing 'ou' ordered by name asc");
   List<Country> countries = countryService.findByNameContainingOrderByNameAsc("ou");
   countries.forEach(c -> LOGGER.info(c.toString()));
   LOGGER.info("End");
}
private static void testFindCountriesByStartingLetter() {
   LOGGER.info("Start - find countries starting with 'Z'");
   List<Country> countries = countryService.findByNameStartingWith("Z");
   countries.forEach(c -> LOGGER.info(c.toString()));
   LOGGER.info("End");
}
```

2) Hands on 2: Write queries on stock table using Query Methods

Solution:

//Stock.java

```
package com.cognizant.ormlearn.model;
import jakarta.persistence.*;
import java.util.Date;
@Entity
@Table(name = "stock")
public class Stock {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int stId;
    @Column(name = "st_code")
    private String stCode;
    @Column(name = "st_date")
    @Temporal(TemporalType.DATE)
    private Date stDate;
    @Column(name = "st_open")
    private double stOpen;
    @Column(name = "st_close")
    private double stClose;
    @Column(name = "st_volume")
    private long stVolume;
}
```

//StockRepository.java

```
package com.cognizant.ormlearn.repository;

import java.util.Date;

import java.util.List;

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

import com.cognizant.ormlearn.model.Stock;

public interface StockRepository extends JpaRepository<Stock, Integer> {

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

   List<Stock> findByStCodeAndStOpenGreaterThan(String code, double price);

   List<Stock> findTop3ByOrderByStVolumeDesc();

   List<Stock> findTop3ByStCodeOrderByStOpenAsc(String code);

}
```

//OrmLearnApplication.java

```
private static void testFacebookStockSep2019() throws ParseException {

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

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

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

   List<Stock> stocks = stockRepository.findByStCodeAndStDateBetween("FB", start, end);

   stocks.forEach(s -> LOGGER.info(s.toString()));

}

private static void testGooglePriceAbove1250() {

   List<Stock> stocks = stockRepository.findByStCodeAndStOpenGreaterThan("GOOGLE",
1250);

   stocks.forEach(s -> LOGGER.info(s.toString()));

}

private static void testTop3HighestVolume() {

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

   stocks.forEach(s -> LOGGER.info(s.toString()));

}
```

```
private static void testNetflixLowest3() {
    List<Stock> stocks =
stockRepository.findTop3ByStCodeOrderByStOpenAsc("NETFLIX");
    stocks.forEach(s -> LOGGER.info(s.toString()));
}
```

3) Hands on 3 : Create payroll tables and bean mapping

Solution:

//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)
    private int id;
    private String name;
    private double salary;
    private boolean permanent;
    @Column(name = "date_of_birth")
    @Temporal(TemporalType.DATE)
    private Date dateOfBirth;
}
```

//Department.java

```
package com.cognizant.ormlearn.model;
import jakarta.persistence.*;
@Entity
@Table(name = "department")
public class Department {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;
    private String name;
}
```

//Skill.java

```
package com.cognizant.ormlearn.model;
import jakarta.persistence.*;
@Entity
@Table(name = "skill")
public class Skill {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;
    private String name;
}
```

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

Solution:

//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)
    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 = "em_dp_id")
private Department department;
}
```

//EmployeeService.java

```
@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

```
private static void testGetEmployee() {
    Employee employee = employeeService.get(1);
    LOGGER.info("Employee: " + employee);
    LOGGER.info("Department: " + employee.getDepartment());
}
private static void testAddEmployee() {
    Employee emp = new Employee();
    emp.setName("John");
    emp.setPermanent(true);
    emp.setSalary(50000);
    emp.setDateOfBirth(new Date());
    Department dept = departmentService.get(1);
    emp.setDepartment(dept);
    employeeService.save(emp);
    LOGGER.info("Added Employee: " + emp);
}
private static void testUpdateEmployee() {
    Employee emp = employeeService.get(1);
    Department dept = departmentService.get(2);
    emp.setDepartment(dept);
    employeeService.save(emp);
    LOGGER.info("Updated Employee: " + emp);
}
```

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

Solution:

//Department.java

```
package com.cognizant.ormlearn.model;
import jakarta.persistence.*;
@Entity
@Table(name = "department")
public class Department {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;
    private String name;
@OneToMany(mappedBy = "department", fetch = FetchType.EAGER)
private Set<Employee> employeeList;
}
```

//OrmLearnApplication.java

```
private static void testGetEmployee() {
    Employee employee = employeeService.get(1);
    LOGGER.info("Employee: " + employee);
    LOGGER.info("Department: " + employee.getDepartment());
}
private static void testAddEmployee() {
    Employee emp = new Employee();
    emp.setName("John");
    emp.setPermanent(true);
    emp.setSalary(50000);
    emp.setDateOfBirth(new Date());
    Department dept = departmentService.get(1);
    emp.setDepartment(dept);
    employeeService.save(emp);
    LOGGER.info("Added Employee: " + emp);
}
private static void testUpdateEmployee() {
    Employee emp = employeeService.get(1);
    Department dept = departmentService.get(2);
    emp.setDepartment(dept);
    employeeService.save(emp);
    LOGGER.info("Updated Employee: " + emp);
}
private static void testGetDepartment() {
    Department dept = departmentService.get(1);
    LOGGER.info("Department: " + dept);
    LOGGER.info("Employees: " + dept.getEmployeeList());
}
```

6) Hands on 6: Implement many to many relationship between Employee and Skill

Solution:

//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)
    private int id;
    private String name;
    private double salary;
    private boolean permanent;
    @Column(name = "date_of_birth")
    @Temporal(TemporalType.DATE)
    private Date dateOfBirth;
@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<>();
}
```

//Skill.java

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

```
@Entity
@Table(name = "skill")
public class Skill {
   @Id
   @GeneratedValue(strategy = GenerationType.IDENTITY)
   private int id;
   private String name;
@ManyToMany(mappedBy = "skillList")
private Set<Employee> employeeList = new HashSet<>();
}
```

//OrmLearnApplication.java

```
private static void testGetEmployee() {
   Employee employee = employeeService.get(1);
   LOGGER.info("Employee: " + employee);
   LOGGER.info("Department: " + employee.getDepartment());
}
private static void testAddEmployee() {
   Employee emp = new Employee();
   emp.setName("John");
   emp.setPermanent(true);
   emp.setSalary(50000);
   emp.setDateOfBirth(new Date());
   Department dept = departmentService.get(1);
   emp.setDepartment(dept);
   employeeService.save(emp);
   LOGGER.info("Added Employee: " + emp);
}
private static void testUpdateEmployee() {
   Employee emp = employeeService.get(1);
   Department dept = departmentService.get(2);
```

```
    emp.setDepartment(dept);
    employeeService.save(emp);
    LOGGER.info("Updated Employee: " + emp);
}
private static void testGetDepartment() {
    Department dept = departmentService.get(1);
    LOGGER.info("Department: " + dept);
    LOGGER.info("Employees: " + dept.getEmployeeList());
}
private static void testAddSkillToEmployee() {
    Employee emp = employeeService.get(1);
    Skill skill = skillService.get(2);
    emp.getSkillList().add(skill);
    employeeService.save(emp);
    LOGGER.info("Added skill to employee");
}
```