**EXERCISE 5**

**1.** Defining query methods

a. finding employee by department

package com.example.EmployeeManagementSystem.repository;

import com.example.EmployeeManagementSystem.model.Employee;

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

import java.util.List;

public interface EmployeeRepository extends JpaRepository<Employee, Long> {

List<Employee> findByDepartmentName(String departmentName);

List<Employee> findByNameStartingWith(String prefix);

}

b. finding departments

package com.example.EmployeeManagementSystem.repository;

import com.example.EmployeeManagementSystem.model.Department;

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

import java.util.List;

public interface DepartmentRepository extends JpaRepository<Department, Long> {

List<Department> findByNameContaining(String keyword);

}

2. Custom query

a. employee

package com.example.EmployeeManagementSystem.repository;

import com.example.EmployeeManagementSystem.model.Employee;

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

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

import org.springframework.data.repository.query.Param;

import java.util.List;

public interface EmployeeRepository extends JpaRepository<Employee, Long> {

@Query("SELECT e FROM Employee e WHERE e.email LIKE %:domain")

List<Employee> findEmployeesByEmailDomain(@Param("domain") String domain);

}

b. departments

package com.example.EmployeeManagementSystem.repository;

import com.example.EmployeeManagementSystem.model.Department;

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

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

import java.util.List;

public interface DepartmentRepository extends JpaRepository<Department, Long> {

@Query(value = "SELECT \* FROM departments d WHERE d.name LIKE %:keyword%", nativeQuery = true)

List<Department> findDepartmentsByNameKeyword(@Param("keyword") String keyword);

}

3.Named queries

package com.example.EmployeeManagementSystem.model;

import lombok.Data;

import javax.persistence.\*;

import java.util.List;

@Data

@Entity

@Table(name = "employees")

@NamedQueries({

@NamedQuery(name = "Employee.findByDepartmentName",

query = "SELECT e FROM Employee e WHERE e.department.name = :departmentName"),

@NamedQuery(name = "Employee.findByEmailDomain",

query = "SELECT e FROM Employee e WHERE e.email LIKE :domain")

})

public class Employee {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(nullable = false)

private String name;

@Column(nullable = false, unique = true)

private String email;

@ManyToOne

@JoinColumn(name = "department\_id")

private Department department;

}

4.named queries in repository

package com.example.EmployeeManagementSystem.repository;

import com.example.EmployeeManagementSystem.model.Employee;

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

import org.springframework.data.repository.query.Param;

import java.util.List;

public interface EmployeeRepository extends JpaRepository<Employee, Long> {

List<Employee> findByDepartmentName(@Param("departmentName") String departmentName);

List<Employee> findByEmailDomain(@Param("domain") String domain);

}