**Demonstrate writing Hibernate Query Language and Native Query**

**HibernateQueryDemoApplication.java:**

package com.example.hibernatequerydemo;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class HibernateQueryDemoApplication {

public static void main(String[] args) {

SpringApplication.run(HibernateQueryDemoApplication.class, args);

}}

**Employee.java:**

package com.example.hibernatequerydemo.entity;

import jakarta.persistence.\*;

import lombok.Data;

*@Entity*

*@Data*

public class Employee {

*@Id*

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

private Long id;

private String name;

private double salary;

*@ManyToOne*

*@JoinColumn*(name = "department\_id")

private Department department;

public Long getId() {

return id;

}

public String getName() {

return name;

}

public double getSalary() {

return salary;

}

public Department getDepartment() {

return department;

}

public void setId(Long id) {

this.id = id;

}

public void setName(String name) {

this.name = name;

}

public void setSalary(double salary) {

this.salary = salary;

}

public void setDepartment(Department department) {

this.department = department;

}

*@Override*

public String toString() {

return "Employee{id=" + id + ", name='" + name + "', salary=" + salary + "}";

}

}

**Department.java:**

package com.example.hibernatequerydemo.entity;

import jakarta.persistence.\*;

import java.util.List;

import lombok.Data;

*@Entity*

*@Data*

public class Department {

*@Id*

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

private Long id;

private String name;

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

private List<Employee> employees;

public Long getId() {

return id;

}

public String getName() {

return name;

}

public List<Employee> getEmployees() {

return employees;

}

public void setId(Long id) {

this.id = id;

}

public void setName(String name) {

this.name = name;

}

public void setEmployees(List<Employee> employees) {

this.employees = employees;

}

*@Override*

public String toString() {

return "Department{id=" + id + ", name='" + name + "'}";

}

}

**DepartmentRepository.java:**

package com.example.hibernatequerydemo.repository;

import com.example.hibernatequerydemo.entity.Department;

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

public interface DepartmentRepository extends JpaRepository<Department, Long> {

}

**EmployeeRepository.java:**

package com.example.hibernatequerydemo.repository;

import com.example.hibernatequerydemo.entity.Department;

import com.example.hibernatequerydemo.entity.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.name = :name")

List<Employee> findByNameHQL(@Param("name") String name);

@Query("SELECT d FROM Department d JOIN FETCH d.employees WHERE d.name = :name")

Department fetchDepartmentWithEmployees(@Param("name") String name);

@Query("SELECT AVG(e.salary) FROM Employee e")

Double getAverageSalary();

@Query(value = "SELECT \* FROM employee WHERE salary > :salary", nativeQuery = true)

List<Employee> findBySalaryNative(@Param("salary") double salary);

}

**AppRunner.java:**

package com.example.hibernatequerydemo.runner;

import com.example.hibernatequerydemo.entity.Department;

import com.example.hibernatequerydemo.entity.Employee;

import com.example.hibernatequerydemo.repository.DepartmentRepository;

import com.example.hibernatequerydemo.repository.EmployeeRepository;

import org.springframework.boot.CommandLineRunner;

import org.springframework.stereotype.Component;

import java.util.Arrays;

@Component

public class AppRunner implements CommandLineRunner {

private final EmployeeRepository employeeRepository;

private final DepartmentRepository departmentRepository;

public AppRunner(EmployeeRepository employeeRepository, DepartmentRepository departmentRepository) {

this.employeeRepository = employeeRepository;

this.departmentRepository = departmentRepository;

}

@Override

public void run(String... args) {

Department d1 = new Department();

d1.setName("IT");

departmentRepository.save(d1);

Employee e1 = new Employee();

e1.setName("Alice");

e1.setSalary(50000);

e1.setDepartment(d1);

Employee e2 = new Employee();

e2.setName("Bob");

e2.setSalary(70000);

e2.setDepartment(d1);

employeeRepository.saveAll(Arrays.asList(e1, e2));

System.out.println("==== Find by HQL ====");

employeeRepository.findByNameHQL("Alice").forEach(System.out::println);

System.out.println("==== Department with Fetch ====");

System.out.println(employeeRepository.fetchDepartmentWithEmployees("IT"));

System.out.println("==== Average Salary ====");

System.out.println(employeeRepository.getAverageSalary());

System.out.println("==== Native Query: Salary > 60000 ====");

employeeRepository.findBySalaryNative(60000).forEach(System.out::println);

}

}

**application.properties:**

spring.datasource.url=jdbc:h2:mem:testdb

spring.datasource.driverClassName=org.h2.Driver

spring.datasource.username=sa

spring.datasource.password=

spring.jpa.database-platform=org.hibernate.dialect.H2Dialect

spring.h2.console.enabled=true

spring.jpa.show-sql=true

spring.jpa.hibernate.ddl-auto=create

**pom.xml:**

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>hibernate-query-demo</artifactId>

<version>1.0.0</version>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>3.2.4</version>

</parent>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<dependency>

<groupId>com.h2database</groupId>

<artifactId>h2</artifactId>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<version>1.18.30</version>

<scope>provided</scope>

</dependency>

</dependencies>

</project>

