**Exercise 10: Employee Management System - Hibernate-Specific Features**

**Business Scenario:**

Leverage Hibernate-specific features to enhance your application's performance and capabilities.

**Instructions:**

1. **Hibernate-Specific Annotations:**

Use Hibernate-specific annotations to customize entity mappings.

Configuring Hibernate Dialect and Properties:

Configure Hibernate dialect and properties for optimal performance.

1. **Batch Processing:**

Implement batch processing with Hibernate for bulk operations.

package com.company.service;

import com.company.entity.Employee;

import com.company.repository.EmployeeRepository;

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

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

import java.util.List;

@Service

public class Bulkservice {

@Autowired

private EmployeeRepository employeeRepository;

@Transactional

public void saveInBatch(List<Employee> employees) {

for (int i = 0; i < employees.size(); i++) {

employeeRepository.save(employees.get(i));

if (i % 20 == 0 && i > 0) {

employeeRepository.flush();

}

}

}

}

spring.application.name=EmployeeManagementSystem

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

spring.datasource.driverClassName=org.h2.Driver

spring.datasource.username=sa

spring.datasource.password=password

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.H2Dialect

spring.jpa.show-sql=true

spring.jpa.properties.hibernate.format\_sql=true

spring.jpa.properties.hibernate.jdbc.batch\_size=20

spring.jpa.properties.hibernate.order\_inserts=true

spring.jpa.properties.hibernate.order\_updates=true

spring.jpa.properties.hibernate.generate\_statistics=true

spring.jpa.hibernate.ddl-auto=update

spring.jpa.properties.hibernate.envers.audit\_table\_suffix=\_AUD

spring.h2.console.enabled=true

spring.h2.console.path=/h2-console

package com.company.entity;

import jakarta.persistence.\*;

import lombok.\*;

import org.springframework.data.annotation.CreatedBy;

import org.springframework.data.annotation.LastModifiedBy;

import org.springframework.data.jpa.domain.support.AuditingEntityListener;

import org.hibernate.annotations.CreationTimestamp;

import org.hibernate.annotations.UpdateTimestamp;

import org.hibernate.annotations.DynamicInsert;

import org.hibernate.annotations.DynamicUpdate;

import java.time.LocalDateTime;

@NamedQueries({

@NamedQuery(

name = "Employee.findByEmail",

query = "SELECT e FROM Employee e WHERE e.email = :email"

),

@NamedQuery(

name = "Employee.findByDeptName",

query = "SELECT e FROM Employee e WHERE e.department.name = :deptName"

)

})

@Entity

@EntityListeners(AuditingEntityListener.class)

@Table(name = "employees")

@Data

@NoArgsConstructor

@AllArgsConstructor

@DynamicInsert

@DynamicUpdate

public class Employee {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

private String name;

private String email;

@ManyToOne

@JoinColumn(name="department\_id")

private Department department;

@CreationTimestamp

private LocalDateTime createdDate;

@UpdateTimestamp

private LocalDateTime modifiedDate;

@CreatedBy

private String createdBy;

@LastModifiedBy

private String modifiedBy;

}









