**Exercise 2: Employee Management System - Creating Entities Business Scenario:**

Define JPA entities for Employee and Department with appropriate relationships. Employee Entity:

package com.example.employeemanagementsystem.entity;

import jakarta.persistence.Entity;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

import jakarta.persistence.JoinColumn;

import jakarta.persistence.ManyToOne;

import jakarta.persistence.Table;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

@Entity

@Table(name = "employees") @Data

@NoArgsConstructor @AllArgsConstructor public class Employee {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY) private Long id;

private String name; private String email;

@ManyToOne

@JoinColumn(name = "department\_id", nullable = false) private Department department;

}

Department Entity:

package com.example.employeemanagementsystem.entity;

import jakarta.persistence.Entity;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

import jakarta.persistence.OneToMany;

import jakarta.persistence.Table;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

import java.util.List;

@Entity

@Table(name = "departments")

@Data

@NoArgsConstructor @AllArgsConstructor

public class Department {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY) private Long id;

private String name;

@OneToMany(mappedBy = "department") private List<Employee> employees;

}

**Mapping Entities to Database Tables**

* **Annotations:**
  + @Entity: Marks the class as a JPA entity.
  + @Table: Specifies the table name in the database (optional, defaults to the class name).
  + @Id: Marks the primary key field.
  + @GeneratedValue: Configures the strategy for primary key generation (e.g., IDENTITY).
  + @ManyToOne: Defines a many-to-one relationship from Employee to Department.
  + @OneToMany: Defines a one-to-many relationship from Department to Employee.
  + @JoinColumn: Specifies the foreign key column in the Employee table.

**Explanation of the Relationship**

* **Employee and Department:**
  + Each Employee belongs to a single Department, represented by a ManyToOne relationship.
  + A Department can have multiple Employees, represented by a OneToMany relationship.

**Final Steps**

After defining these entities:

1. **Create Repositories:** Create Spring Data JPA repositories for Employee and Department to handle database operations.
2. **Create Services and Controllers:** Implement services and controllers to manage business logic and expose RESTful APIs.

Your entities are now ready, and you can proceed with further implementation to manage the employee and department data.