**Exercise 1: Employee Management System - Overview and Setup**

* **Objective**: Kickstart the development of an Employee Management System by setting up a Spring Boot project.
* **Key Tasks**:
  + **Spring Boot Project Initialization**:
    - Create a new Spring Boot project named EmployeeManagementSystem using Spring Initializr or a preferred IDE.
    - **Dependencies**: Add essential dependencies:
      * **Spring Data JPA**: To handle the persistence layer and database operations.
      * **H2 Database**: A lightweight, in-memory database for quick setup and testing.
      * **Spring Web**: For building RESTful web services.
      * **Lombok**: To reduce boilerplate code with annotations like @Getter, @Setter, @ToString, etc.
  + **Application Configuration**:
    - **Database Configuration**: Set up the H2 database connection in the application.properties file:
      * Define the JDBC URL, driver class name, username, password, and Hibernate dialect.
    - **Example Configuration**:

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

spring.datasource.driverClassName=org.h2.Driver

spring.datasource.username=sa

spring.datasource.password=password

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

**Exercise 2: Creating Entities**

1. **Define Employee Entity**:
   * Fields: id, name, email, department.
   * Use annotations: @Entity, @Id, @GeneratedValue.
2. **Define Department Entity**:
   * Fields: id, name.
   * Use similar annotations as the Employee entity.
3. **Map Relationships**:
   * Set up a one-to-many relationship between Department and Employee.

**Exercise 3: Creating Repositories**

1. **Create EmployeeRepository Interface**:
   * Extend JpaRepository<Employee, Long>.
2. **Create DepartmentRepository Interface**:
   * Extend JpaRepository<Department, Long>.
3. **Define Basic Query Methods**:
   * You can add methods like findByDepartmentName(String name).

**Exercise 4: Implementing CRUD Operations**

1. **Implement Basic CRUD Operations**:
   * Use save(), findById(), findAll(), and deleteById() from JpaRepository.
2. **Create REST Controllers**:
   * EmployeeController and DepartmentController to handle requests.
3. **Map Endpoints**:
   * Define endpoints like /employees, /departments, etc., for CRUD operations.

**Exercise 5: Defining Query Methods**

1. **Custom Query Methods**:
   * Add methods in repositories using naming conventions, like findByName(String name).
2. **Use @Query Annotation**:
   * For more complex queries, define methods with @Query annotations.
3. **Implement Named Queries**:
   * Use @NamedQuery and @NamedQueries annotations for reusable queries.