**EXERCISE 3**

**Overview of Spring Data Repositories**

**Benefits of Using Spring Data Repositories:**

* Simplicity: Spring Data repositories simplify data access by reducing the need for boilerplate code.
* CRUD Operations: Built-in support for CRUD (Create, Read, Update, Delete) operations.
* Query Derivation: Ability to define custom query methods using method naming conventions.
* Pagination and Sorting: Support for pagination and sorting out of the box.
* Integration with JPA: Provides seamless integration with JPA, making it easy to use JPA features.
* Custom Implementations: Ability to add custom method implementations if needed.

**Creating Repositories**

To create repositories for the Employee and Department entities, follow these steps:

**EmployeeRepository**

1. **Create a new interface** named EmployeeRepository in the package com.example.employeemanagementsystem.repository.
2. **Extend the JpaRepository interface**, specifying Employee as the entity type and Long as the ID type.
3. **Define derived query methods** based on requirements.

**DepartmentRepository**

1. **Create a new interface** named DepartmentRepository in the same package.
2. **Extend the JpaRepository interface**, specifying Department as the entity type and Long as the ID type.
3. **Define derived query methods** based on requirements.

**NOTE:**

**Derived Query Methods**: The methods like findByDepartmentId, findByEmail, and findByName are examples of derived query methods. Spring Data JPA generates the necessary SQL based on the method names and parameters.

**Interface Naming**: By convention, repository interfaces are named with the entity name followed by Repository (e.g., EmployeeRepository, DepartmentRepository). **JpaRepository Interface**: Provides basic CRUD operations and additional JPA-specific methods like flushing the persistence context and batch deletes.