**Employee Management System - Creating Repositories**

1. **Overview of Spring Data Repositories:**

Benefits of using Spring Data repositories:

* **Simplified Data Access Layer:** Spring Data JPA repositories provide a straightforward way to interact with the database without writing boilerplate code. They handle common operations like save, find, delete, etc. through simple method calls.
* **Derived Query Methods:** Spring Data allows you to define query methods in the repository interface by following a naming convention, which automatically translates to SQL queries.
* **Custom Query Methods:** If anyone need more complex queries, they can define custom JPQL or native SQL queries within the repository interface.
  + - * + **JPQL stands for Java Persistence Query Language**. Spring Data provides multiple ways to create and execute a query, and JPQL is one of these. It defines queries using the @Query annotation in Spring to execute both JPQL and native SQL queries. The query definition uses JPQL by default.
        + **A native query** is a SQL statement that is specific to a particular database like MySQL. It varies a little from JPQL (Java Persistence Query Language) which is used by Spring Data JPA by default.

1. **Creating Repositories:**
   * **JpaRepository Interface:**
     + - * **JpaRepository<T, ID>** interface is a JPA specific extension of CrudRepository. It provides extra methods like flushing the persistence context and batch deleting records.
         * **EmployeeRepository and DepartmentRepository** interfaces extend JpaRepository, enabling CRUD operations for Employee and Department entities.
2. **Defined Derived Query Methods in these repositories:**
   * + **findByName(String name)** Finds employees or departments by their name.
     + **findByDepartmentId(Long departmentId)** Finds employees by their department ID. This method allows you to query all employees belonging to a specific department.