**Spring Data JPA with Spring Boot, Hibernate**

**(Additional hands-on)**

**Demonstrate implementation of Query Methods feature of Spring Data JPA**

**Source Code**

Create Query Methods in CountryRepository:

package com.himaja.ormlearn.repository;

import java.util.List;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

import com.himaja.ormlearn.model.Country;

@Repository

public interface CountryRepository extends JpaRepository<Country, String> {

// 1. Find countries where name contains a specific substring

List<Country> findByNameContaining(String keyword);

// 2. Same as above, but sorted by name (ascending)

List<Country> findByNameContainingOrderByNameAsc(String keyword);

// 3. Find countries where name starts with a specific prefix (e.g., 'Z')

List<Country> findByNameStartingWith(String prefix);

}

OrmLearnApplication.java:

private static void testSearchCountryByPartialName() {

LOGGER.info("Start testSearchCountryByPartialName");

List<Country> countries = countryService.searchCountriesByName("ou");

countries.forEach(country -> LOGGER.debug("Match: {}", country));

LOGGER.info("End testSearchCountryByPartialName");

}

private static void testSearchCountryByPartialNameSorted() {

LOGGER.info("Start testSearchCountryByPartialNameSorted");

List<Country> countries = countryService.searchCountriesByNameSorted("ou");

countries.forEach(country -> LOGGER.debug("Sorted Match: {}", country));

LOGGER.info("End testSearchCountryByPartialNameSorted");

}

private static void testSearchCountryByStartingAlphabet() {

LOGGER.info("Start testSearchCountryByStartingAlphabet");

List<Country> countries = countryService.searchCountriesByAlphabet("Z");

countries.forEach(country -> LOGGER.debug("Starts with Z: {}", country));

LOGGER.info("End testSearchCountryByStartingAlphabet");

}

CountryService:

public List<Country> searchCountriesByName(String keyword) {

return countryRepository.findByNameContaining(keyword);

}

public List<Country> searchCountriesByNameSorted(String keyword) {

return countryRepository.findByNameContainingOrderByNameAsc(keyword);

}

public List<Country> searchCountriesByAlphabet(String prefix) {

return countryRepository.findByNameStartingWith(prefix);

}

**Call the Methods from main() in OrmLearnApplication**

Java

testSearchCountryByPartialName();

testSearchCountryByPartialNameSorted();

testSearchCountryByStartingAlphabet();

MySQL Table Query Validation:

-- Matches countries containing 'ou'

SELECT \* FROM country WHERE co\_name LIKE '%ou%';

-- Sorted by name ascending

SELECT \* FROM country WHERE co\_name LIKE '%ou%' ORDER BY co\_name;

-- Starting with Z

SELECT \* FROM country WHERE co\_name LIKE 'Z%';

Output:

