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

## *Step 1: Update the Country Entity*

Add a population and date field:

import jakarta.persistence.\*;

import java.time.LocalDate;

@Entity

public class Country {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(unique = true)

private String code;

private String name;

private String capital;

private Long population;

private LocalDate independenceDate;

}

## *Step 2: CountryRepository with Query Methods*

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

import java.time.LocalDate;

import java.util.List;

public interface CountryRepository extends JpaRepository<Country, Long> {

List<Country> findByNameContainingIgnoreCase(String name);

List<Country> findByNameStartingWithIgnoreCase(String prefix);

List<Country> findAllByOrderByPopulationDesc();

List<Country> findByIndependenceDateBetween(LocalDate start, LocalDate end);

List<Country> findByPopulationGreaterThan(Long population);

List<Country> findByPopulationLessThan(Long population);

List<Country> findTop3ByOrderByPopulationDesc();

}

## *Step 3: Sample Test Cases (Use Postman or Swagger)*

### *1.* *****Search by name containing text*****

GET /api/countries/search?name=an

@GetMapping("/search")

public List<Country> searchByName(@RequestParam String name) {

return countryRepository.findByNameContainingIgnoreCase(name);

}

### *2.* *****Countries starting with text*****

GET /api/countries/start?prefix=In

@GetMapping("/start")

public List<Country> findStartingWith(@RequestParam String prefix) {

return countryRepository.findByNameStartingWithIgnoreCase(prefix);

}

### *3.* *****Sorted by population (desc)*****

GET /api/countries/sorted

@GetMapping("/sorted")

public List<Country> getSortedCountries() {

return countryRepository.findAllByOrderByPopulationDesc();

}

### *4.* *****Fetch countries between two dates*****

GET /api/countries/daterange?start=1950-01-01&end=2000-12-31

@GetMapping("/daterange")

public List<Country> getBetweenDates(@RequestParam LocalDate start, @RequestParam LocalDate end) {

return countryRepository.findByIndependenceDateBetween(start, end);

}

### *5.* *****Countries with population > X*****

GET /api/countries/greater?pop=100000000

@GetMapping("/greater")

public List<Country> getByGreaterPopulation(@RequestParam Long pop) {

return countryRepository.findByPopulationGreaterThan(pop);

}

### *6.* *****Top 3 most populous*****

GET /api/countries/top3

@GetMapping("/top3")

public List<Country> getTop3Countries() {

return countryRepository.findTop3ByOrderByPopulationDesc();

}