**1. Country Entity**

package com.example.model;

import jakarta.persistence.\*;

import java.util.List;

@Entity

public class Country {

@Id

private String code;

private String name;

@OneToMany(mappedBy = "country", cascade = CascadeType.ALL, fetch = FetchType.LAZY)

private List<State> states;

// Getters and setters

}

**2. State Entity**

package com.example.model;

import jakarta.persistence.\*;

import java.time.LocalDate;

@Entity

public class State {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

private String name;

private LocalDate establishedDate;

private int population;

@ManyToOne(fetch = FetchType.EAGER)

@JoinColumn(name = "country\_code")

private Country country;

// Getters and setters

}

**StateRepository.java**

package com.example.repository;

import com.example.model.State;

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

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

import java.time.LocalDate;

import java.util.List;

public interface StateRepository extends JpaRepository<State, Long> {

// Find states with name containing substring (case insensitive)

List<State> findByNameContainingIgnoreCase(String name);

// Sort by population descending

List<State> findAllByOrderByPopulationDesc();

// Filter states starting with

List<State> findByNameStartingWith(String prefix);

// Fetch between two dates

List<State> findByEstablishedDateBetween(LocalDate start, LocalDate end);

// Population greater than

List<State> findByPopulationGreaterThan(int population);

// Top 3 states by population

List<State> findTop3ByOrderByPopulationDesc();

// Custom JPQL

@Query("SELECT s FROM State s WHERE s.population > ?1 AND s.name LIKE %?2%")

List<State> findLargeStatesWithName(int pop, String nameFragment);

// Native SQL

@Query(value = "SELECT \* FROM state WHERE population < ?1", nativeQuery = true)

List<State> findStatesWithLowPopulation(int population);

}

**StateController.java**

@RestController

@RequestMapping("/states")

public class StateController {

@Autowired

private StateRepository stateRepository;

@GetMapping("/search")

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

return stateRepository.findByNameContainingIgnoreCase(name);

}

@GetMapping("/top")

public List<State> topStates() {

return stateRepository.findTop3ByOrderByPopulationDesc();

}

@GetMapping("/between")

public List<State> betweenDates(

@RequestParam String from,

@RequestParam String to) {

return stateRepository.findByEstablishedDateBetween(

LocalDate.parse(from), LocalDate.parse(to));

}

@GetMapping("/custom")

public List<State> customQuery(@RequestParam int pop, @RequestParam String name) {

return stateRepository.findLargeStatesWithName(pop, name);

}

@GetMapping("/native")

public List<State> nativeQuery(@RequestParam int pop) {

return stateRepository.findStatesWithLowPopulation(pop);

}

}

**Sample Requests**

**1. Search by Name (contains)**

GET /states/search?name=an

**2. Top 3 populous**

GET /states/top

**3. States between 1950 and 2000**

GET /states/between?from=1950-01-01&to=2000-12-31

**4. Custom JPQL**

GET /states/custom?pop=1000000&name=York

**5. Native SQL**

GET /states/native?pop=500000