Spring Data JPA Hands-On Solutions

# Hands-on 1: Country Table Query Methods

1. Search countries by text (e.g., "ou"):  
List<Country> findByNameContaining(String text);

2. Search countries by text and sort ascending:  
List<Country> findByNameContainingOrderByNameAsc(String text);

3. Get countries starting with a specific alphabet:  
List<Country> findByNameStartingWith(String prefix);

# Hands-on 2: Stock Table Query Methods

1. Facebook stock data for September 2019:  
List<Stock> findByCodeAndDateBetween(String code, LocalDate startDate, LocalDate endDate);

2. Google stock data where close price > 1250:  
List<Stock> findByCodeAndCloseGreaterThan(String code, Double price);

3. Top 3 by volume:  
List<Stock> findTop3ByOrderByVolumeDesc();

4. Bottom 3 Netflix stock close prices:  
List<Stock> findTop3ByCodeOrderByCloseAsc(String code);

# Hands-on 3-6: ORM Relationship Mappings

Entities: Employee, Department, Skill

Mapping and annotations for each entity:

* Employee.java:  
  @Entity  
  public class Employee {  
   @Id  
   @GeneratedValue(strategy = GenerationType.IDENTITY)  
   private int id;  
   private String name;  
   private double salary;  
   private boolean permanent;  
   private Date dateOfBirth;  
    
   @ManyToOne  
   @JoinColumn(name = "em\_dp\_id")  
   private Department department;  
    
   @ManyToMany(fetch = FetchType.EAGER)  
   @JoinTable(name = "employee\_skill",  
   joinColumns = @JoinColumn(name = "es\_em\_id"),  
   inverseJoinColumns = @JoinColumn(name = "es\_sk\_id"))  
   private Set<Skill> skillList;  
  }

Department.java:  
@Entity  
public class Department {  
 @Id  
 @GeneratedValue(strategy = GenerationType.IDENTITY)  
 private int id;  
 private String name;  
  
 @OneToMany(mappedBy = "department", fetch = FetchType.EAGER)  
 private Set<Employee> employeeList;  
}

Skill.java:  
@Entity  
public class Skill {  
 @Id  
 @GeneratedValue(strategy = GenerationType.IDENTITY)  
 private int id;  
 private String name;  
  
 @ManyToMany(mappedBy = "skillList")  
 private Set<Employee> employeeList;  
}