Spring Data JPA Hands-On Solutions (Part 2)

# Hands-On 1: Query Methods on Country Table

## Search countries by partial name

List<Country> findByNameContaining(String keyword);

## Search and sort countries by name ascending

List<Country> findByNameContainingOrderByNameAsc(String keyword);

## Find countries starting with a given alphabet

List<Country> findByNameStartingWith(String prefix);

# Hands-On 2: Query Methods on Stock Table

## Get all Facebook stock in September 2019

List<Stock> findByCodeAndDateBetween(String code, LocalDate start, LocalDate end);

## Get Google stock with price > 1250

List<Stock> findByCodeAndCloseGreaterThan(String code, BigDecimal price);

## Top 3 dates with highest volume

List<Stock> findTop3ByOrderByVolumeDesc();

## 3 lowest Netflix stock prices

List<Stock> findTop3ByCodeOrderByCloseAsc(String code);

# Hands-On 3: Setup Employee, Department, Skill Tables

Create model classes: Employee, Department, Skill. Annotate with @Entity, @Table, @Id, @GeneratedValue, and @Column.

# Hands-On 4: Many-to-One Relationship (Employee -> Department)

@ManyToOne  
@JoinColumn(name = "em\_dp\_id")  
private Department department;

# Hands-On 5: One-to-Many Relationship (Department -> Employees)

@OneToMany(mappedBy = "department", fetch = FetchType.EAGER)  
private Set<Employee> employeeList;

# Hands-On 6: Many-to-Many Relationship (Employee <-> Skill)

In Employee.java:

@ManyToMany  
@JoinTable(name = "employee\_skill",  
 joinColumns = @JoinColumn(name = "es\_em\_id"),  
 inverseJoinColumns = @JoinColumn(name = "es\_sk\_id"))  
private Set<Skill> skillList;

In Skill.java:

@ManyToMany(mappedBy = "skillList")  
private Set<Employee> employeeList;