Hands-on 5: Implement Services for Managing Country

This exercise involves creating services using Spring Data JPA to manage country-related features:

• Find a country based on country code  
• Add new country  
• Update country  
• Delete country  
• Find list of countries matching a partial country name

# 1. Hibernate Table Creation Configuration

The `spring.jpa.hibernate.ddl-auto` setting defines how Hibernate interacts with the database schema:  
  
• create - Drops existing data and creates new tables.  
• validate - Checks if the table and columns exist; throws error if missing.  
• update - Adds missing tables or columns.  
• create-drop - Creates tables, then drops them after use.  
  
Example configuration:  
spring.jpa.hibernate.ddl-auto=validate

# 2. Populate Country Table

Delete all records in the Country table, then populate using the provided SQL script.

Sample SQL insert:

insert into country (co\_code, co\_name) values ("AF", "Afghanistan");  
insert into country (co\_code, co\_name) values ("AL", "Albania");  
... (More entries in the script)

Refer to the full SQL script for inserting all countries into the table.

# 3. Feature Implementations

You will implement the following in the `CountryService` class using Spring Data JPA:

1. Find a country by code using `findById()`.  
2. Add a new country using `save()`.  
3. Update an existing country using `save()` (on an existing entity).  
4. Delete a country using `deleteById()`.  
5. Find countries by partial name using a custom query method such as `findByNameContaining()`.

These service methods will be utilized and tested in the subsequent hands-on exercises.