**Spring Data JPA - Quick Example**

CREATE DATABASE ormlearn;

SHOW DATABASES;

USE ormlearn;

CREATE TABLE country (

co\_code VARCHAR(2) PRIMARY KEY,

co\_name VARCHAR(50)

);

INSERT INTO country (co\_code, co\_name) VALUES

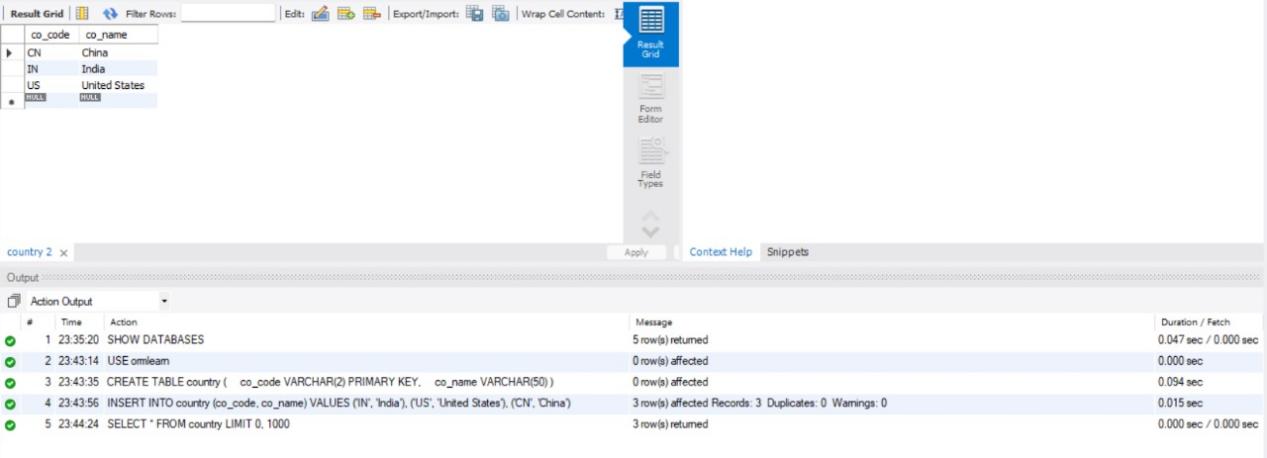
('IN', 'India'),

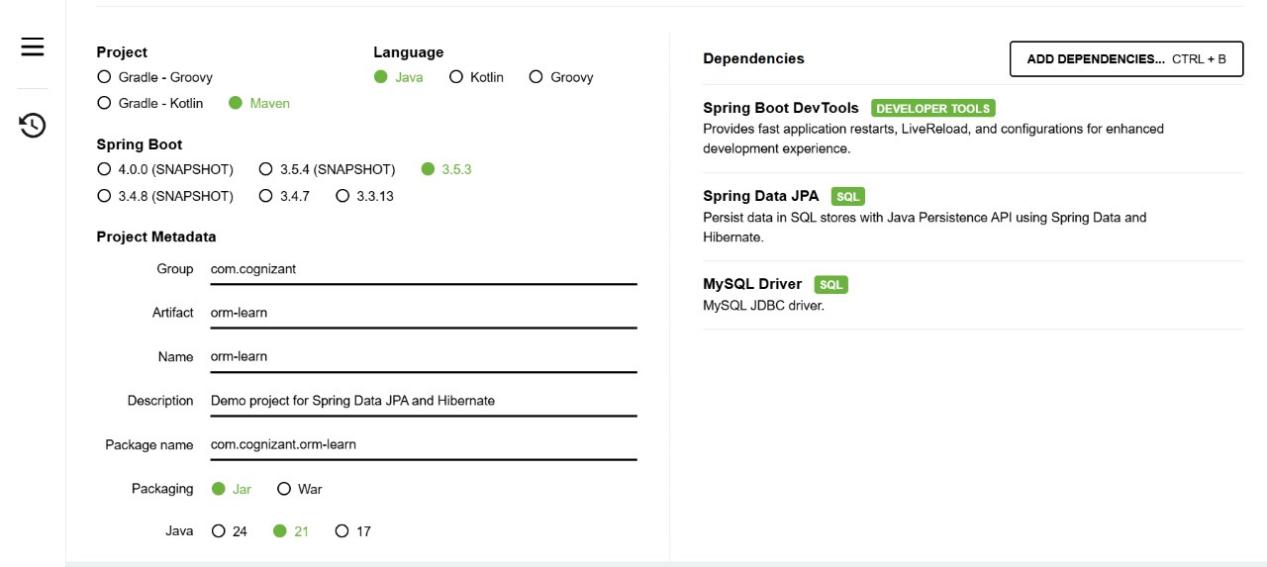
('US', 'United States'),

('CN', 'China');

SELECT \* FROM country;

**Output:**





package com.cognizant.orm\_learn;

import jakarta.persistence.Column;

import jakarta.persistence.Entity;

import jakarta.persistence.Id;

import jakarta.persistence.Table;

@Entity

@Table(name = "country")

public class Country {

@Id

@Column(name = "co\_code")

private String code;

@Column(name = "co\_name")

private String name;

public Country() {

// default constructor

}

public String getCode() {

return code;

}

public void setCode(String code) {

this.code = code;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

@Override

public String toString() {

return "Country [code=" + code + ", name=" + name + "]";

}

}

package com.cognizant.orm\_learn;

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

import org.springframework.stereotype.Repository;

@Repository

public interface CountryRepository extends JpaRepository<Country, String> {

}

package com.cognizant.orm\_learn;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class OrmLearnApplication implements CommandLineRunner {

@Autowired

private CountryRepository countryRepository;

public static void main(String[] args) {

SpringApplication.run(OrmLearnApplication.class, args);

}

@Override

public void run(String... args) throws Exception {

List<Country> countries = countryRepository.findAll();

for (Country country : countries) {

System.out.println(country);

}

}

}

spring.application.name=orm-learn

# Logging

logging.level.org.springframework=info

logging.level.com.cognizant=debug

# Hibernate SQL logs

logging.level.org.hibernate.SQL=trace

logging.level.org.hibernate.type.descriptor.sql=trace

# Log pattern

logging.pattern.console=%d{dd-MM-yy} %d{HH:mm:ss.SSS} %-20.20thread %-25.25logger{25} %25M %4L %m%n

# Database connection

spring.datasource.url=jdbc:mysql://localhost:3306/ormlearn

spring.datasource.username=root

spring.datasource.password=Sql123#

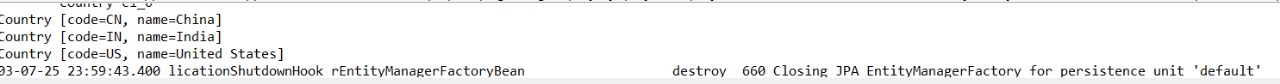
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

spring.jpa.properties.hibernate.format\_sql=true

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect



### ****Difference between JPA, Hibernate and Spring Data JPA****

### ****JPA (Java Persistence API)****

* JPA is a **Java specification (JSR 338)** that defines a standard approach for mapping Java objects to relational database tables using Object-Relational Mapping (ORM).  
  It provides annotations and interfaces to manage database operations in a platform-independent way but does **not provide its own implementation**.

### ****Hibernate****

* Hibernate is a **popular ORM framework** that provides a **concrete implementation of the JPA specification**.  
  It handles database operations using Java objects and includes **additional features** beyond JPA, such as caching, HQL (Hibernate Query Language), and advanced performance tuning.

### ****Spring Data JPA****

* Spring Data JPA is a **part of the Spring Data project** that provides a **higher-level abstraction over JPA and Hibernate**.  
  It simplifies data access by **automatically generating repository implementations**, reducing boilerplate code, and integrating tightly with Spring Boot.