**Week 3**

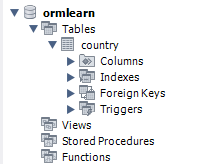
**Spring Data JPA and Hibernate**

**Hands on 1**

**Spring Data JPA - Quick Example**   
  
**Software Pre-requisites**

* MySQL Server 8.0
* MySQL Workbench 8
* Eclipse IDE for Enterprise Java Developers 2019-03 R
* Maven 3.6.2

**Creating ormlearn schema**



**CODE**

CREATE SCHEMA ormlearn;

CREATE TABLE country (

code VARCHAR(2) PRIMARY KEY,

name VARCHAR(50)

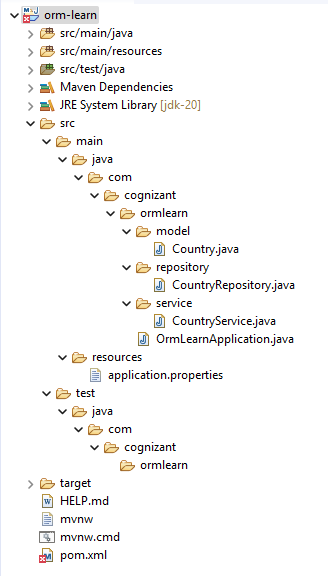
);

INSERT INTO country values ('IN', 'India');

INSERT INTO country values ('US', 'United States of America');



**FILE STRUCTURE**



**CODE**

*pom.xml*

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<project xmlns=*"http://maven.apache.org/POM/4.0.0"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>2.7.18</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<groupId>com.cognizant</groupId>

<artifactId>orm-learn</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>orm-learn</name>

<description>Spring Boot ORM Learn Project</description>

<packaging>jar</packaging>

<properties>

<java.version>20</java.version>

</properties>

<dependencies>

<!-- Spring Boot Starters -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter</artifactId>

</dependency>

<!-- MySQL Driver -->

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.33</version>

<scope>runtime</scope>

</dependency>

<!-- Test dependencies -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.11.0</version>

<configuration>

<source>20</source>

<target>20</target>

</configuration>

</plugin>

<!-- Spring Boot plugin -->

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

*Country.java*

***com.cognizant.orm-learn.model.Country***

package com.cognizant.ormlearn.model;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.Id;

import javax.persistence.Table;

@Entity

@Table(name = "country")

public class Country {

@Id

@Column(name = "code")

private String code;

@Column(name = "name")

private String name;

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 + "]";

}

}

*CountryRepository.java*

***com.cognizant.orm-learn.CountryRepository***

package com.cognizant.ormlearn.repository;

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

import org.springframework.stereotype.Repository;

import com.cognizant.ormlearn.model.Country;

@Repository

public interface CountryRepository extends JpaRepository<Country, String> {

}

*CountryService.java*

***com.cognizant.orm-learn.service.CountryService***

package com.cognizant.ormlearn.service;

import java.util.List;

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

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

import com.cognizant.ormlearn.model.Country;

import com.cognizant.ormlearn.repository.CountryRepository;

@Service

public class CountryService {

@Autowired

private CountryRepository countryRepository;

@Transactional

public List<Country> getAllCountries() {

return countryRepository.findAll();

}

}

*OrmLearnApplication.java*

package com.cognizant.ormlearn;

import com.cognizant.ormlearn.model.Country;

import com.cognizant.ormlearn.service.CountryService;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

import java.util.List;

@SpringBootApplication

public class OrmLearnApplication {

private static final Logger LOGGER = LoggerFactory.getLogger(OrmLearnApplication.class);

private static CountryService countryService;

public static void main(String[] args) {

ApplicationContext context = SpringApplication.run(OrmLearnApplication.class, args);

LOGGER.info("Inside main");

countryService = context.getBean(CountryService.class);

testGetAllCountries();

}

private static void testGetAllCountries() {

LOGGER.info("Start");

List<Country> countries = countryService.getAllCountries();

LOGGER.debug("countries={}", countries);

LOGGER.info("End");

}

}

*applicationproperties*

spring.application.name=orm-learn

# Logging

logging.level.org.springframework=info

logging.level.com.cognizant=debug

logging.level.org.hibernate.SQL=trace

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

logging.pattern.console=%d{dd-MM-yy} %d{HH:mm:ss.SSS} %-20.20thread %5p %-25.25logger**{25}** %25M %4L %m%n

# Database

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

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

spring.datasource.username=root

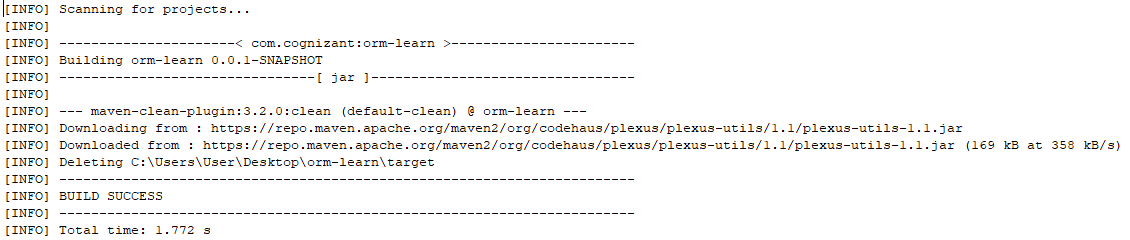
spring.datasource.password=\*\*\*\*\*\*\*\*\*\*\*\*\*\*

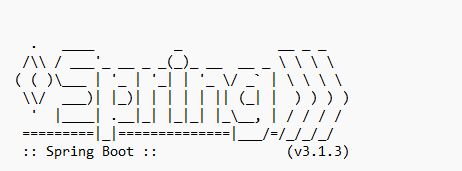
# Hibernate

spring.jpa.hibernate.ddl-auto=validate

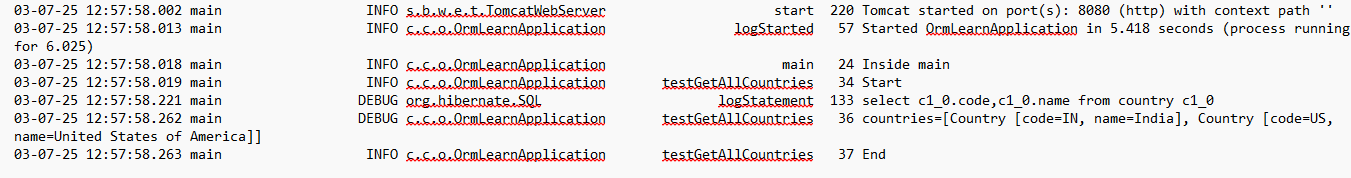
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect

**OUTPUT**





……………………………………………………………………………………………………………



**HandsOn 4**

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

JPA is the *standard*, Hibernate is the *implementation*, and Spring Data JPA is the *Spring-powered wrapper* that simplifies JPA usage.

1. JPA (Java Persistence API) – *The Specification*

A standard specification provided by Java (Jakarta EE) for Object-Relational Mapping (ORM). It is like an interface — it defines what to do, but not how.

Example features: @Entity, @Id

It provides Standard annotations & methods for ORM, without implementation.

It is just a set of rules; it requires an implementation. Implemented by Hibernate, EclipseLink, OpenJPA, etc.

2. Hibernate – *The Implementation*

A popular implementation of the JPA specification.It is like a class that implements an interface.

Hibernate implements JPA and adds extra features.

3. Spring Data JPA – *The Abstraction Layer*

A Spring framework module that builds on top of JPA and Hibernate to simplify data access. It is a helper or assistant that automatically generates data access code. It provides JpaRepository, CrudRepository, query method derivation, pagination, etc.

Spring Data JPA helps reduce boilerplate and simplifies database interaction using Spring.

**CODE Comparison between Hibernate and Spring Data JPA**

*Hibernate*

/\* Method to CREATE an employee in the database \*/

   public Integer addEmployee(Employee employee){

      Session session = factory.openSession();

      Transaction tx = null;

      Integer employeeID = null;

      try {

         tx = session.beginTransaction();

         employeeID = (Integer) session.save(employee);

         tx.commit();

      } catch (HibernateException e) {

         if (tx != null) tx.rollback();

         e.printStackTrace();

} finally {

         session.close();

      }

      return employeeID;

   }

**Spring Data JPA**  
*EmployeeRespository.java*

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

*EmployeeService.java*

@Autowire

  private EmployeeRepository employeeRepository;

@Transactional

public void addEmployee(Employee employee) {

  employeeRepository.save(employee);

  }