**WEEK – 3**

**SPRING DATA JPA**

**MANDATORY**

1) **Spring Data JPA - Quick Example**

OrmLearnApplication.java

**package** com.cognizant.orm\_learn;

**import** org.springframework.boot.SpringApplication;

**import** org.springframework.boot.autoconfigure.SpringBootApplication;

**import** org.springframework.context.ApplicationContext;

**import** org.slf4j.Logger;

**import** org.slf4j.LoggerFactory;

**import** com.cognizant.orm\_learn.service.CountryService;

**import** java.util.List;

**import** com.cognizant.orm\_learn.model.Country;

@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");

}

}

Country.java

**package** com.cognizant.orm\_learn.model;

**import** jakarta.persistence.Entity;

**import** jakarta.persistence.Id;

**import** jakarta.persistence.Table;

**import** jakarta.persistence.Column;

@Entity

@Table(name="country")

**public** **class** Country {

@Id

@Column(name="code")

**private** String code;

@Column(name="name")

**private** String name;

// getters and setters

// toString()

}

CountryRepository.java

**package** com.cognizant.orm\_learn.repository;

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

**import** org.springframework.stereotype.Repository;

**import** com.cognizant.orm\_learn.model.Country;

@Repository

**public** **interface** CountryRepository **extends** JpaRepository<Country, String> {

}

CountryService.java

**package** com.cognizant.orm\_learn.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.orm\_learn.model.Country;

**import** com.cognizant.orm\_learn.repository.CountryRepository;

@Service

**public** **class** CountryService {

@Autowired

**private** CountryRepository countryRepository;

@Transactional

**public** List<Country> getAllCountries() {

**return** countryRepository.findAll();

}

}

Application.properties

spring.application.name=orm-learn

# Spring Framework and application log

logging.level.org.springframework=info

logging.level.com.cognizant=debug

# Hibernate logs for displaying executed SQL, input and output

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 %5p %-25.25logger**{25}** %25M %4L %m%n

# Database configuration

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

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

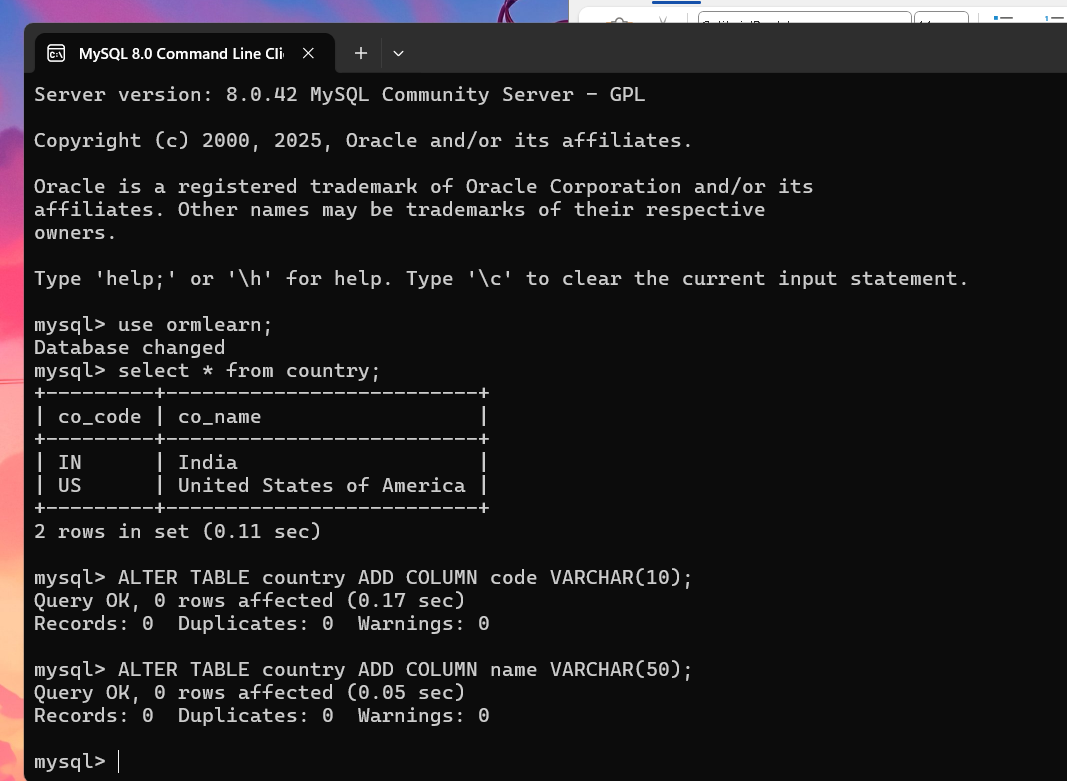
spring.datasource.username=root

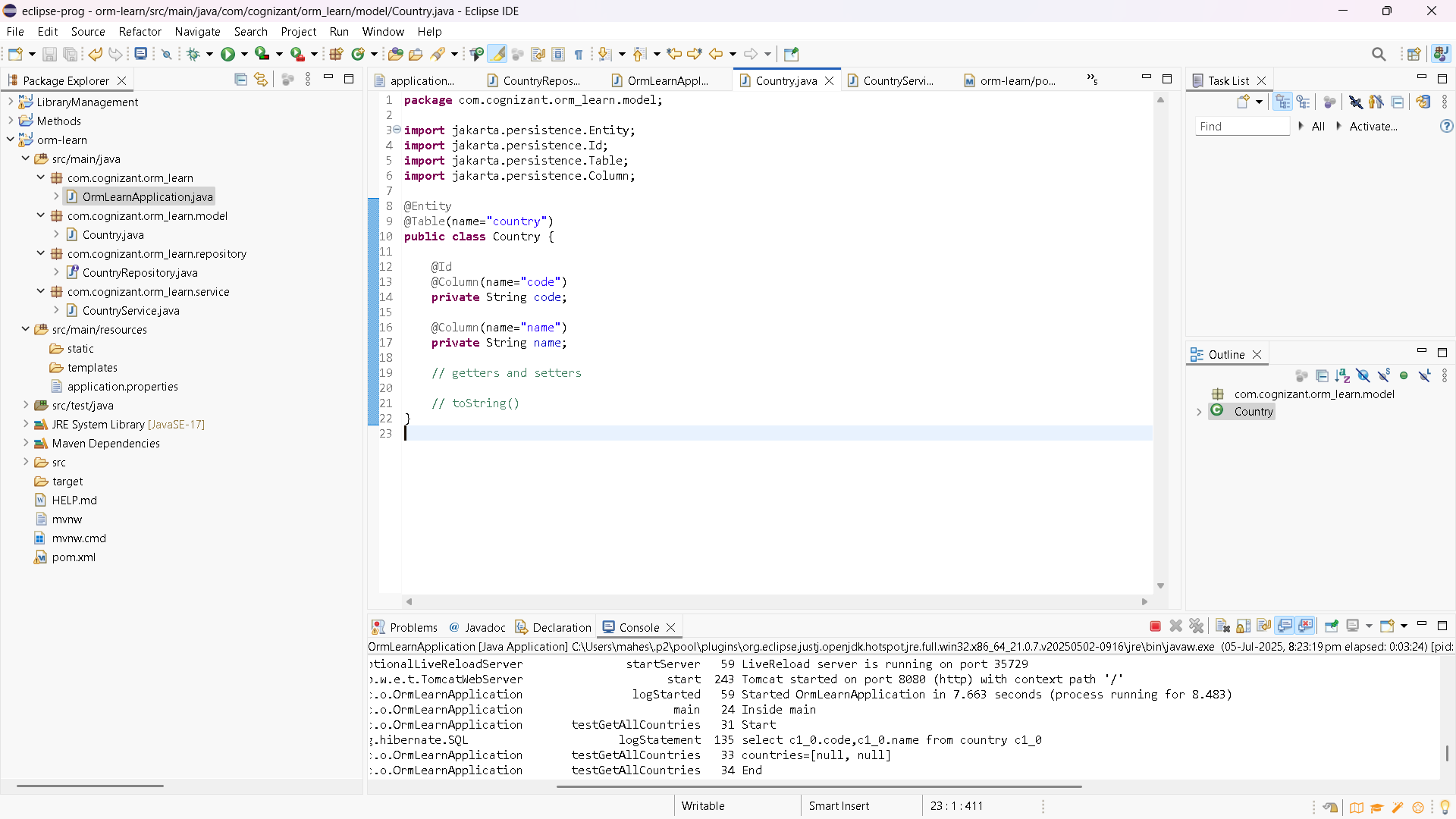
spring.datasource.password=Cherry@123

# Hibernate configuration

spring.jpa.hibernate.ddl-auto=validate

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





**2) Difference Between JPA, Hibernate, and Spring Data JPA**

**Java Persistence API (JPA)**

* JPA is a specification which specifies how to access, manage and persist information/data between java objects and relational databases.
* It provides a standard approach for ORM, Object Relational Mapping. Spring Boot provides a seem-less integration with JPA..
* It doesn't provide any implementation – it just defines how persistence should work.
* **Hibernate** is a popular framework that implements this JPA specification.

**Hibernate**

* Hibernate is an **ORM (Object Relational Mapping)** tool.
* It is a **concrete implementation of JPA** and also provides extra features like caching, lazy loading, etc.
* With Hibernate, you manage sessions and transactions manually.

**Spring Data JPA**

* Spring Data JPA is part of the Spring ecosystem that makes JPA usage even easier.
* It acts as a **layer on top of JPA providers like Hibernate** to reduce boilerplate code.
* It provides built-in methods for CRUD operations and supports custom query generation using method names.

The major difference between Hibernate and JPA is that Hibernate is a framework while JPA is API specifications. Hibernate is the implementation of all the JPA guidelines.

**Hibernate**is an Object-Relational Mapping (ORM) framework that provides a high-level API for interacting with relational databases. It allows you to map Java objects to database tables and perform database operations using a high-level API, instead of writing low-level SQL code.

**JPA**(Java Persistence API) is a specification that defines a set of interfaces and annotations for working with relational databases in Java applications. JPA provides a common API for ORM frameworks like Hibernate, EclipseLink, and OpenJPA, making it easier to switch between different ORM frameworks without changing your code.

**Spring Data JPA** is a part of the Spring Framework that provides a higher-level, easier-to-use API for working with JPA. It reduces the amount of boilerplate code required to interact with a database using JPA, and provides a repository abstraction that provides a number of methods out of the box for performing CRUD operations on JPA entities.

**2. Include Code Comparison**

**Example:**

**Using Hibernate:**

Session session = factory.openSession();

Transaction tx = session.beginTransaction();

session.save(employee);

tx.commit();

session.close();

**Using Spring Data JPA:**

employeeRepository.save(employee);

Spring handles the session, transaction, and connection behind the scenes.