**Spring Data JPA with Spring Boot, Hibernate**

**Hands-on**

**Name : Udit Bhargava**

**Superset Id : 6386801**

**Hands-on 1: Spring Data JPA - Quick Example**

**CODE :**

**MySQL Workbench :**

create schema ormlearn;

**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');

**application.properties :**

spring.application.name=orm-learn  
server.port = 8282  
  
# 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=1604  
  
# Hibernate configuration  
spring.jpa.hibernate.ddl-auto=validate  
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect  
  
spring.banner.mode=off

**com.cognizant.orm\_learn.model.Country :**

package com.cognizant.orm\_learn.model;  
  
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 = "code")  
 private String id;  
  
 @Column(name = "name")  
 private String name;  
  
 public String getId() {  
 return id;  
 }  
  
 public void setId(String id){  
 this.id = id;  
 }  
  
 public String getName(){  
 return name;  
 }  
  
 public void setName(String name){  
 this.name = name;  
 }  
  
 @Override  
 public String toString(){  
 return "Country{" + "id=" + id + ", name='" + name + '\'' + "}";  
 }  
}

**com.cognizant.orm\_learn.CountryRepository :**

package com.cognizant.orm\_learn.repository;  
  
import com.cognizant.orm\_learn.model.Country;  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface CountryRepository extends JpaRepository<Country, String> {  
}

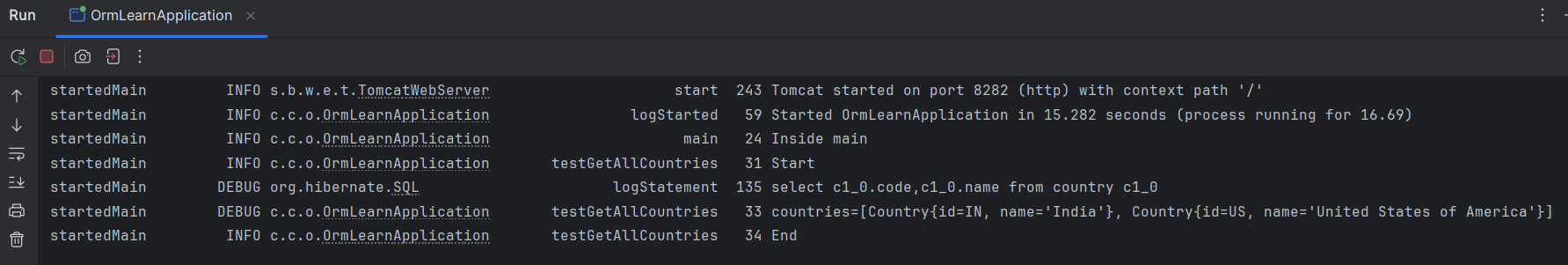
**com.cognizant.orm\_learn.service.CountryService :**

package com.cognizant.orm\_learn.service;  
  
import com.cognizant.orm\_learn.model.Country;  
import com.cognizant.orm\_learn.repository.CountryRepository;  
import jakarta.transaction.Transactional;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Service;  
  
import java.util.List;  
  
@Service  
public class CountryService {  
  
 @Autowired  
 private CountryRepository countryRepository;  
  
 @Transactional  
 public List<Country> getAllCountries(){  
 return countryRepository.findAll();  
 }  
}

**OrmLearnApplication.java :**

package com.cognizant.orm\_learn;  
  
import java.util.List;  
  
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 com.cognizant.orm\_learn.model.Country;  
import com.cognizant.orm\_learn.service.CountryService;  
  
@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");  
 }  
}

**Output :**

****

**Hands-on 2: Difference between JPA, Hibernate and Spring Data JPA**

**Java Persistence API (JPA) :**

* JSR 338 Specification for persisting, reading and managing data from Java objects
* Does not contain concrete implementation of the specification
* Hibernate is one of the implementation of JPA
* JPA is a specification provided by Java to define how Java objects should be persisted in a relational database. It is just a set of interfaces and rules, not an actual implementation. It defines things like entities, relationships, and query methods using annotations or XML, but it does not do anything on its own. To use JPA, you need a JPA provider like Hibernate or EclipseLink.

**Hibernate :**

* ORM Tool that implements JPA
* Hibernate is a JPA provider — it is an actual implementation of the JPA specification.
* It allows you to interact with the database using Java objects and provides features such as connection pooling, lazy loading, caching, etc. Hibernate also adds some extra features beyond JPA, like its own query language (HQL), interceptors, and multi-tenancy. When you write JPA code, Hibernate can be the engine that executes that code.

**Spring Data JPA :**

* Does not have JPA implementation, but reduces boiler plate code
* This is another level of abstraction over JPA implementation provider like Hibernate
* Manages transactions
* Spring Data JPA is a Spring framework module that simplifies the use of JPA in Spring applications.
* It builds on top of JPA and a provider like Hibernate to eliminate boilerplate code — like writing queries or handling entity managers manually. It allows you to create repositories just by defining interfaces, and Spring automatically generates the implementation at runtime. It also makes configuration easier using application.properties instead of XML files or manual setups.

**CODE :**

**com.example.spring1.model :**

package com.example.spring1.model;  
  
import jakarta.persistence.\*;  
  
@Entity  
@Table(name = "employee")  
public class Employee {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private int id;  
  
 private String name;  
 private double salary;  
  
 // Getters and Setters  
 public int getId() { return id; }  
 public void setId(int id) { this.id = id; }  
  
 public String getName() { return name; }  
 public void setName(String name) { this.name = name; }  
  
 public double getSalary() { return salary; }  
 public void setSalary(double salary) { this.salary = salary; }  
}

**com.example.spring1.repository :**

package com.example.spring1.repository;  
  
import com.example.spring1.model.Employee;  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface EmployeeRepository extends JpaRepository<Employee, Integer> {  
}

**com.example.spring1.service :**

package com.example.spring1.service;  
  
import com.example.spring1.model.Employee;  
import com.example.spring1.repository.EmployeeRepository;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Service;  
import org.springframework.transaction.annotation.Transactional;  
  
@Service  
public class EmployeeService {  
 @Autowired  
 private EmployeeRepository employeeRepository;  
  
 @Transactional  
 public void addEmployee(Employee employee) {  
 employeeRepository.save(employee);  
 }  
}

**com.example.spring1.controller :**

package com.example.spring1.controller;  
  
import com.example.spring1.model.Employee;  
import com.example.spring1.service.EmployeeService;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.web.bind.annotation.\*;  
  
@RestController  
@RequestMapping("/employees")  
public class EmployeeController {  
 @Autowired  
 private EmployeeService employeeService;  
  
 @GetMapping("/test")  
 public String test() {  
 return "Spring Boot is working!";  
 }  
  
 @PostMapping("/add")  
 public String add(@RequestBody Employee employee) {  
 employeeService.addEmployee(employee);  
 return "Employee added!";  
 }  
}

**com.example.spring1.hibernate :**

package com.example.spring1.hibernate;  
  
import com.example.spring1.model.Employee;  
import org.hibernate.\*;  
import org.hibernate.cfg.Configuration;  
  
  
public class HibernateUtil {  
  
 private static SessionFactory *factory*;  
  
 static {  
 try {  
 Configuration config = new Configuration().configure(); // hibernate.cfg.xml  
 config.addAnnotatedClass(Employee.class);  
 *factory* = config.buildSessionFactory();  
 } catch (Throwable ex) {  
 throw new ExceptionInInitializerError(ex);  
 }  
 }  
  
 public static SessionFactory getSessionFactory() {  
 return *factory*;  
 }  
}

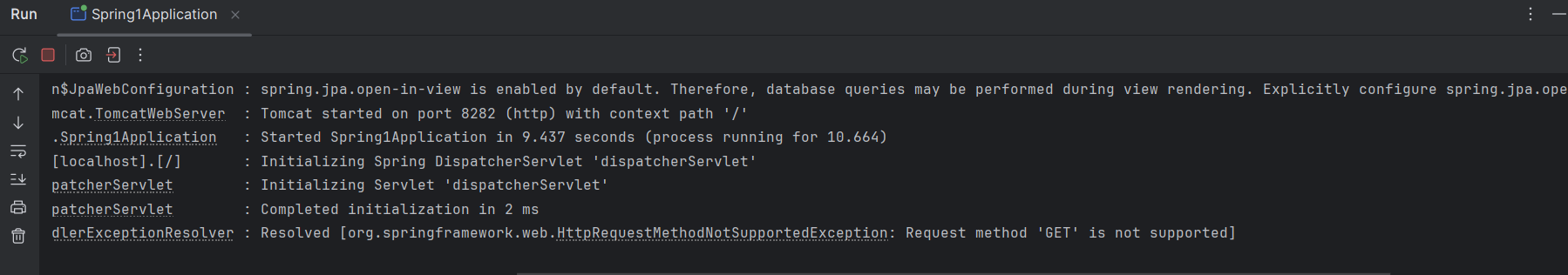
**com.example.spring1.hibernate :**

package com.example.spring1.hibernate;  
  
import com.example.spring1.model.Employee;  
import org.hibernate.Session;  
import org.hibernate.SessionFactory;  
import org.hibernate.Transaction;  
import org.hibernate.cfg.Configuration;  
  
public class EmployeeHibernate {  
  
 private static final SessionFactory *sessionFactory*;  
  
 static {  
 try {  
 Configuration config = new Configuration();  
 config.configure("hibernate.cfg.xml");  
 config.addAnnotatedClass(Employee.class);  
 *sessionFactory* = config.buildSessionFactory();  
 } catch (Throwable ex) {  
 throw new ExceptionInInitializerError("Initial SessionFactory creation failed: " + ex);  
 }  
 }  
  
 public Integer addEmployee(Employee employee) {  
 Transaction tx = null;  
 Integer employeeID = null;  
 try (Session session = *sessionFactory*.openSession()) {  
 tx = session.beginTransaction();  
 employeeID = (Integer) session.save(employee);  
 tx.commit();  
 } catch (Exception e) {  
 if (tx != null) tx.rollback();  
 e.printStackTrace();  
 }  
 return employeeID;  
 }  
}

**MainApplication.java :**

package com.example.spring1;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
  
@SpringBootApplication  
public class Spring1Application {  
 public static void main(String[] args) {  
 SpringApplication.*run*(Spring1Application.class, args);  
 }  
}

**Output :**

****