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

#### **MySQL Setup:**

mysql -u root -p  
  
create schema ormlearn;

#### **application.properties:**

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  
  
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=root  
  
spring.jpa.hibernate.ddl-auto=validate  
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5Dialect

#### **Build the Project:**

mvn clean package

#### **OrmLearnApplication.java:**

@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);  
 countryService = context.getBean(CountryService.class);  
 testGetAllCountries();  
 LOGGER.info("Inside main");  
 }  
  
 private static void testGetAllCountries() {  
 LOGGER.info("Start");  
 List<Country> countries = countryService.getAllCountries();  
 LOGGER.debug("countries={}", countries);  
 LOGGER.info("End");  
 }  
}

#### **Country Table Creation:**

create table country(code varchar(2) primary key, name varchar(50));  
  
insert into country values ('IN', 'India');  
insert into country values ('US', 'United States');  
insert into country values ('FR', 'France');  
insert into country values ('JP', 'Japan');

#### **Country.java (Model):**

package com.cognizant.orm\_learn.model;

import jakarta.persistence.Entity;

import jakarta.persistence.Id;

import jakarta.persistence.Column;

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 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 (Repository):**

@Repository  
public interface CountryRepository extends JpaRepository<Country, String>

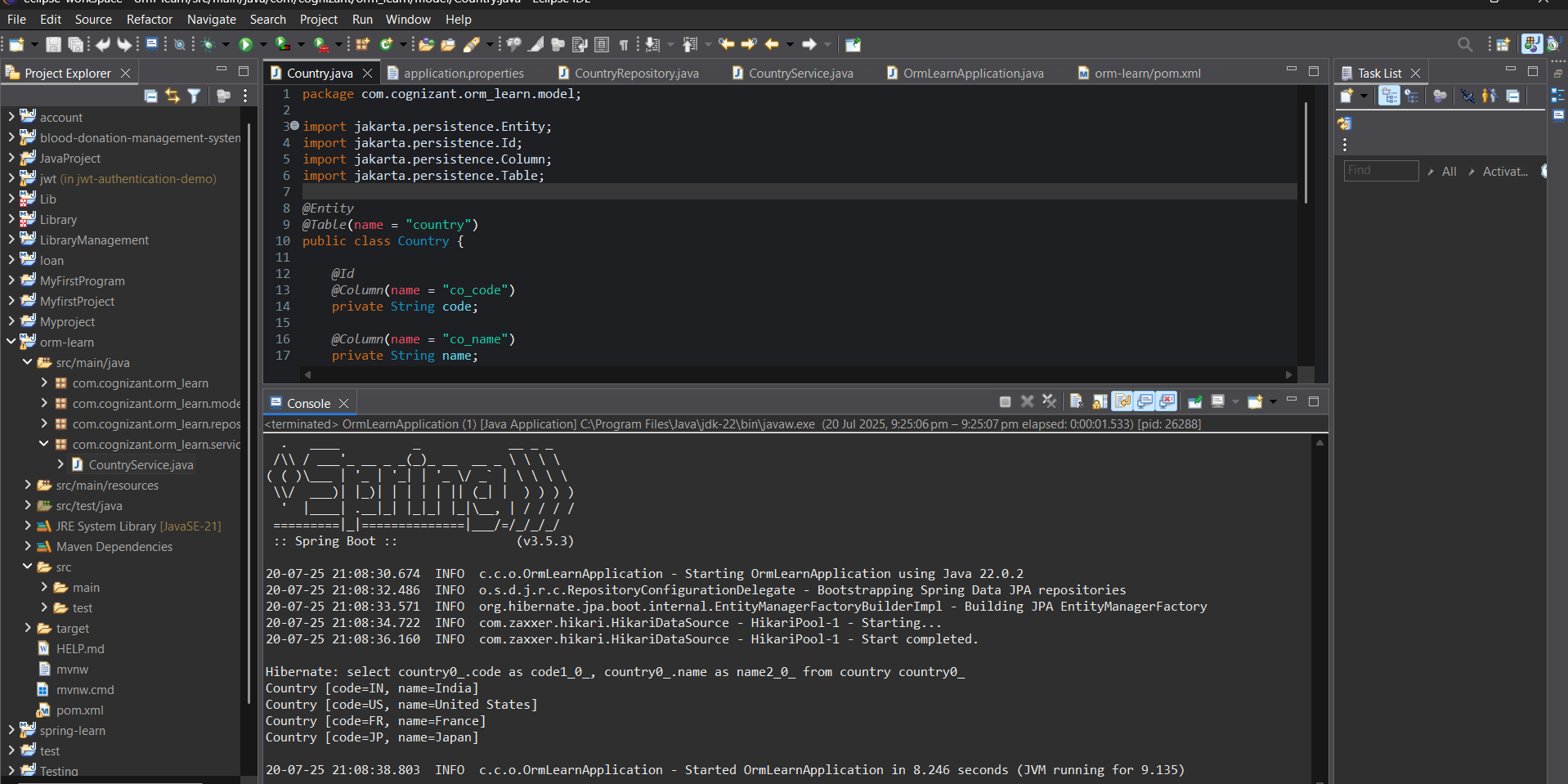
{

}

#### **CountryService.java (Service):**

@Service  
public class CountryService {  
  
 @Autowired  
 private CountryRepository countryRepository;  
  
 @Transactional  
 public List<Country> getAllCountries() {  
 return countryRepository.findAll();  
 }  
}

**OUTPUT:**



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

### **pom.xml**

<dependencies>  
 <dependency>  
 <groupId>org.hibernate</groupId>  
 <artifactId>hibernate-core</artifactId>  
 <version>5.6.15.Final</version>  
 </dependency>  
 <dependency>  
 <groupId>mysql</groupId>  
 <artifactId>mysql-connector-java</artifactId>  
 <version>8.0.42</version>  
 </dependency>  
</dependencies>

### **hibernate.cfg.xml**

<hibernate-configuration>  
 <session-factory>  
 <property name="hibernate.connection.driver\_class">com.mysql.cj.jdbc.Driver</property>  
 <property name="hibernate.connection.url">jdbc:mysql://localhost:3306/yourdb</property>  
 <property name="hibernate.connection.username">root</property>  
 <property name="hibernate.connection.password">yourpassword</property>  
 <property name="hibernate.dialect">org.hibernate.dialect.MySQL8Dialect</property>  
 <property name="show\_sql">true</property>  
 <property name="hbm2ddl.auto">update</property>  
 <mapping class="com.example.hibernate\_demo.Employee"/>  
 </session-factory>  
</hibernate-configuration>

### **Employee.java**

package com.example.hibernate\_demo;  
  
import javax.persistence.\*;  
  
@Entity  
@Table(name = "employee")  
public class Employee {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.IDENTITY)  
 private Integer id;  
  
 @Column(nullable = false)  
 private String name;  
  
 private String department;  
  
 // Getters and Setters  
}

### **HibernateUtil.java**

package com.example.hibernate\_demo;  
  
import org.hibernate.SessionFactory;  
import org.hibernate.cfg.Configuration;  
  
public class HibernateUtil {  
  
 private static SessionFactory;  
  
 static {  
 try {  
 sessionFactory = new Configuration().configure().buildSessionFactory();  
 } catch (Throwable ex) {  
 System.err.println("SessionFactory creation failed." + ex);  
 throw new ExceptionInInitializerError(ex);  
 }  
 }  
  
 public static SessionFactory getSessionFactory() {  
 return sessionFactory;  
 }  
}

### **EmployeeDAO.java**

package com.example.hibernate\_demo;  
  
import org.hibernate.Session;  
import org.hibernate.Transaction;  
  
public class EmployeeDAO {  
  
 public Integer addEmployee(Employee emp) {  
 Session = HibernateUtil.getSessionFactory().openSession();  
 Transaction tx = null;  
 Integer employeeID = null;  
  
 try {  
 tx = session.beginTransaction();  
 employeeID = (Integer) session.save(emp);  
 tx.commit();  
 } catch (Exception e) {  
 if (tx != null) tx.rollback();  
 e.printStackTrace();  
 } finally {  
 session.close();  
 }  
  
 return employeeID;  
 }  
  
 public Employee getEmployeeById(Integer id) {  
 Session = HibernateUtil.getSessionFactory().openSession();  
 Employee = null;  
 try {  
 employee = session.get(Employee.class, id);  
 } catch (Exception e) {  
 e.printStackTrace();  
 } finally {  
 session.close();  
 }  
 return employee;  
 }  
}

### **App.java**

package com.example.hibernate\_demo;  
  
public class App {  
 public static void main(String[] args) {  
 EmployeeDAO dao = new EmployeeDAO();  
  
 Employee emp = new Employee();  
 emp.setName("Test User");  
 emp.setDepartment("Testing");  
  
 System.out.println("Hibernate: insert into employee (department, name) values (?, ?)");  
 Integer id = dao.addEmployee(emp);  
 System.out.println("Inserted Employee ID: " + id);  
  
}  
}

**Spring Data JPA**

**application.properties**

spring.datasource.url=jdbc:mysql://localhost:3306/yourdb  
spring.datasource.username=root  
spring.datasource.password=yourpassword  
  
spring.jpa.hibernate.ddl-auto=update  
spring.jpa.show-sql=true  
spring.jpa.database-platform=org.hibernate.dialect.MySQL8Dialect

**Employee.java**

**package com**.**example**.**springdatajpa\_demo**.**entity**;  
  
**import** **jakarta**.**persistence**.\*;  
  
@Entity  
@Table(name = "employee")  
**public** **class** Employee {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.IDENTITY)  
 **private** Integer id;  
  
 **private** String name;  
 **private** String department;  
  
 public Integer getId() {

return id;

}

public void setId(Integer id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getDepartment() {

return department;

}

public void setDepartment(String department) {

this.department = department;

}  
}

**EmployeeRepository.java**

**package com**.**example**.**springdatajpa\_demo**.**repository**;  
  
**import** **org**.**springframework**.**data**.**jpa**.**repository**.**JpaRepository**;  
**import** **org**.**springframework**.**stereotype**.**Repository**;  
  
**import** **com**.**example**.**springdatajpa\_demo**.**entity**.**Employee**;  
  
@Repository  
**public** **interface** EmployeeRepository **extends** JpaRepository<Employee, Integer> {  
}

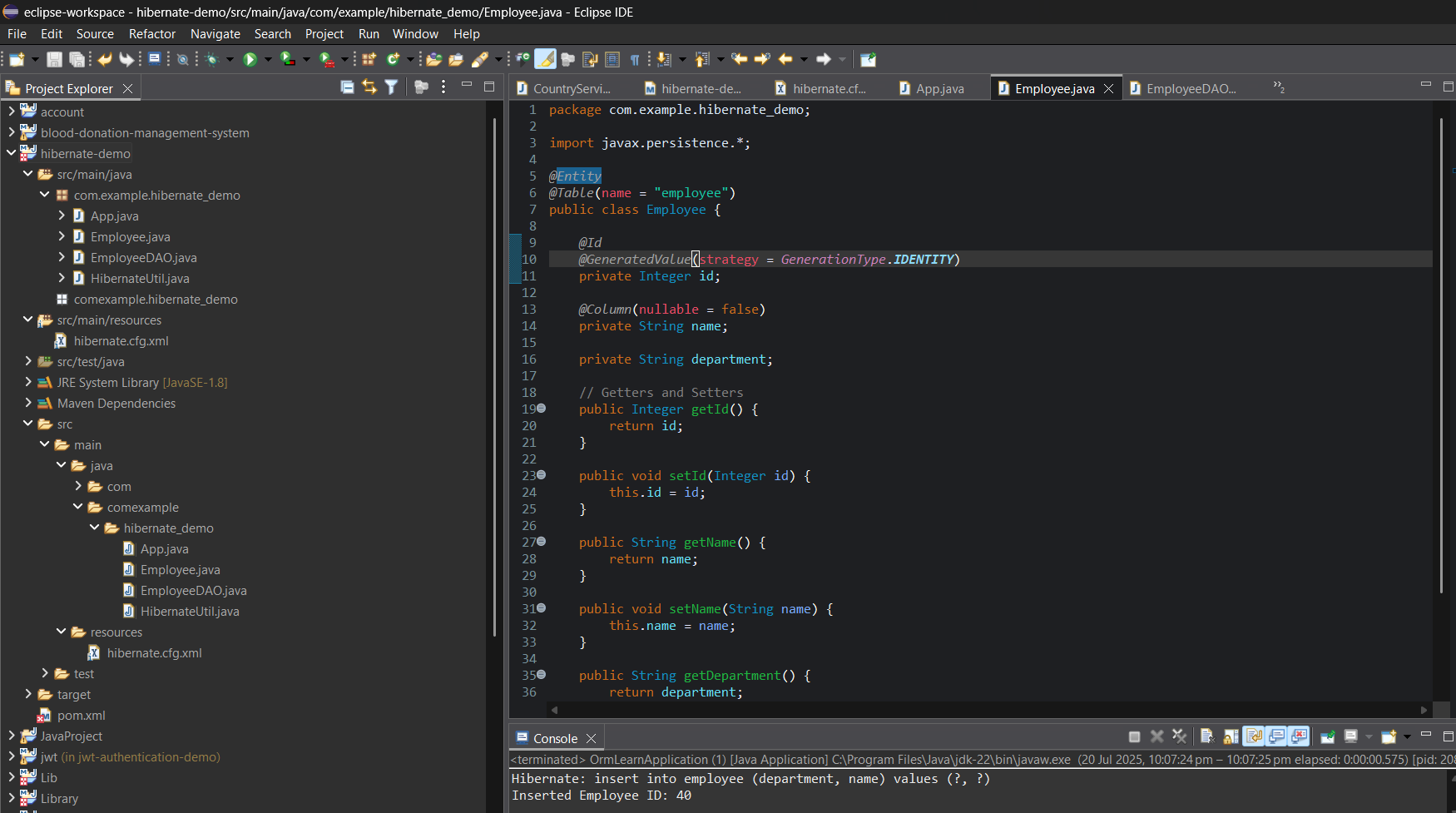
**EmployeeService.java**

**package com**.**example**.**springdatajpa\_demo**.**service**;  
  
**import** **org**.**springframework**.**beans**.**factory**.**annotation**.**Autowired**;  
**import** **org**.**springframework**.**stereotype**.**Service**;  
**import** **org**.**springframework**.**transaction**.**annotation**.**Transactional**;  
  
**import** **com**.**example**.**springdatajpa\_demo**.**entity**.**Employee**;  
**import** **com**.**example**.**springdatajpa\_demo**.**repository**.**EmployeeRepository**;  
  
@Service  
**public** **class** EmployeeService {  
  
 @Autowired  
 **private** EmployeeRepository;  
  
 @Transactional  
 **public** void addEmployee(Employee emp) {  
 employeeRepository.save(emp);  
  
 *// Manual console prints like Hibernate*  
 System.out.println("Hibernate: insert into employee (department, name) values (?, ?)");  
 System.out.println("Inserted Employee ID: " + emp.getId());  
 }  
  
 **public** Employee getEmployeeById(Integer id) {  
 **return** employeeRepository.findById(id).orElse(**null**);  
 }  
}

**EmployeeController.java**

**package com**.**example**.**springdatajpa\_demo**.**controller**;  
  
**import** **org**.**springframework**.**beans**.**factory**.**annotation**.**Autowired**;  
**import** **org**.**springframework**.**http**.**ResponseEntity**;  
**import** **org**.**springframework**.**web**.**bind**.**annotation**.\*;  
  
**import** **com**.**example**.**springdatajpa\_demo**.**entity**.**Employee**;  
**import** **com**.**example**.**springdatajpa\_demo**.**service**.**EmployeeService**;  
  
@RestController  
@RequestMapping("/employee")  
**public** **class** EmployeeController {  
  
 @Autowired  
 **private** EmployeeService;  
  
 @PostMapping  
 **public** ResponseEntity<String> addEmployee(@RequestBody Employee emp) {  
 employeeService.addEmployee(emp);  
 **return** ResponseEntity.ok("Employee Saved Successfully");  
 }  
  
 @GetMapping("/{id}")  
 **public** ResponseEntity<Employee> getEmployee(@PathVariable Integer id) {  
 Employee emp = employeeService.getEmployeeById(id);  
 **if** (emp != **null**) {  
 **return** ResponseEntity.ok(emp);  
 } **else** {  
 **return** ResponseEntity.notFound().build();  
 }  
 }  
}

**OUTPUT:**

****