

NAME:- Vidya CS


TRACK:JAVA STACK

# 1) Spring Data JPA - Quick Example

## 1. Install Prerequisites

1. MySQL Server 8.0
2. MySQL Workbench 8.0
3. Eclipse IDE for Enterprise Java Developers 2019-03 R
4. Maven 3.6.2
5. Java 8 or higher installed

## 2. Create Spring Boot Project via Spring Initializer



☐ Gradle - Groovy

☐ Gradle - Kotlin

☒ Maven

☒ Java

☐ Kotlin

☐ Groovy

☐ 4.0.0 (SNAPSHOT)

☐ 3.5.4 (SNAPSHOT)

☒ 3.5.3

☐ 3.4.8 (SNAPSHOT)

☐ 3.4.7

Group

com.cognizant

Artifact

orm-learn

Name

orm-learn

Dependencies

ADD DEPENDENCIES... CTRL + B

**Spring Boot DevTools** DEVELOPER TOOLS

Provides fast application restarts, LiveReload, and configurations for enhanced development experience.

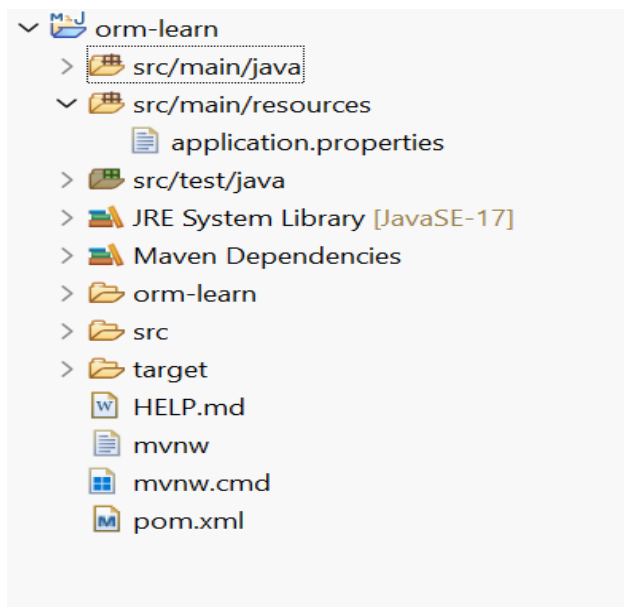
**Spring Data JPA** SQL

Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.

**MySQL Driver** SQL

MySQL JDBC driver.

### 3. Import into Eclipse



### 4. Create MySQL Schema

```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> create schema ormllearn;
Query OK, 1 row affected (0.01 sec)

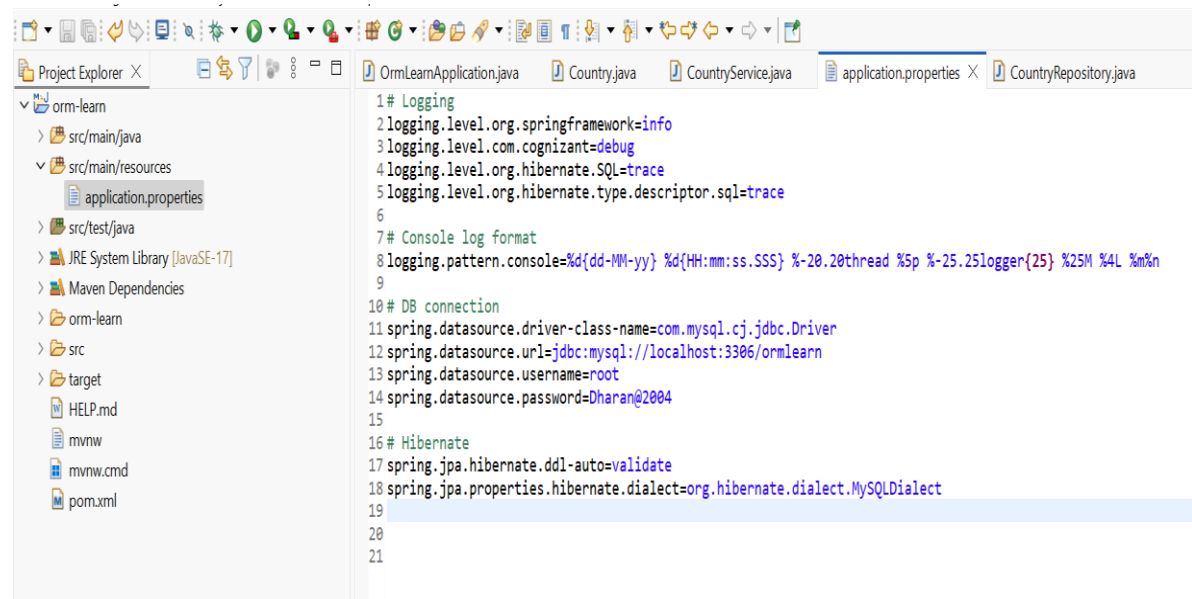
mysql> USE ormllearn;
Database changed
mysql> CREATE TABLE country (
    ->     code VARCHAR(2) PRIMARY KEY,
    ->     name VARCHAR(50)
    -> );
Query OK, 0 rows affected (0.05 sec)

mysql> INSERT INTO country VALUES ('IN', 'India');
Query OK, 1 row affected (0.02 sec)

mysql> INSERT INTO country VALUES ('US', 'United States of America');
Query OK, 1 row affected (0.01 sec)

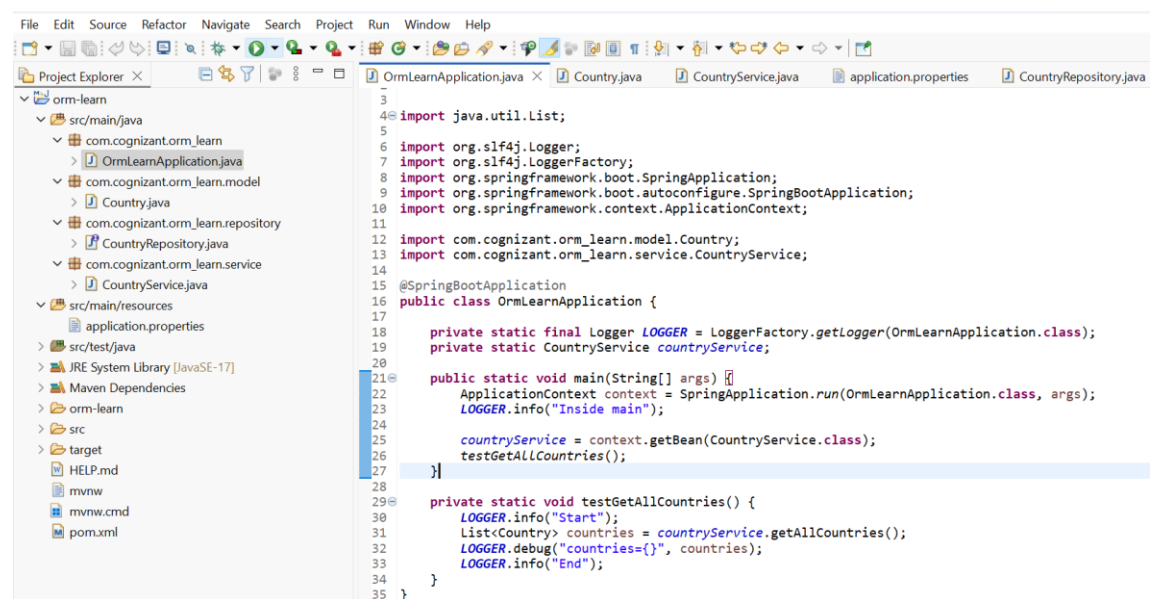
mysql> |
```

## 5. Configure application.properties



```
1# Logging
2logging.level.org.springframework=info
3logging.level.com.cognizant=debug
4logging.level.org.hibernate.SQL=trace
5logging.level.org.hibernate.type.descriptor.sql=trace
6
7# Console log format
8logging.pattern.console=%d{dd-MM-yy} %d{HH:mm:ss.SSS} %-20.20thread %5p %-25.25logger{25} %25M %4L %m%n
9
10# DB connection
11spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
12spring.datasource.url=jdbc:mysql://localhost:3306/ormlearn
13spring.datasource.username=root
14spring.datasource.password=Dharan@2004
15
16# Hibernate
17spring.jpa.hibernate.ddl-auto=validate
18spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect
19
20
21
```

## 6. Add Logging to OrmLearnApplication.java



```
File Edit Source Refactor Navigate Search Project Run Window Help
OrmLearnApplication.java Country.java CountryService.java application.properties CountryRepository.java

3
4import java.util.List;
5
6import org.slf4j.Logger;
7import org.slf4j.LoggerFactory;
8import org.springframework.boot.SpringApplication;
9import org.springframework.boot.autoconfigure.SpringBootApplication;
10import org.springframework.context.ApplicationContext;
11
12import com.cognizant.orm_learn.model.Country;
13import com.cognizant.orm_learn.service.CountryService;
14
15@SpringBootApplication
16public class OrmLearnApplication {
17
18    private static final Logger LOGGER = LoggerFactory.getLogger(OrmLearnApplication.class);
19    private static CountryService countryService;
20
21    public static void main(String[] args) {
22        ApplicationContext context = SpringApplication.run(OrmLearnApplication.class, args);
23        LOGGER.info("Inside main");
24
25        countryService = context.getBean(CountryService.class);
26        testGetAllCountries();
27    }
28
29    private static void testGetAllCountries() {
30        LOGGER.info("Start");
31        List<Country> countries = countryService.getAllCountries();
32        LOGGER.debug("countries={}", countries);
33        LOGGER.info("End");
34    }
35}
```

## 7.Create MySQL

```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> create schema ormlearn;
Query OK, 1 row affected (0.01 sec)

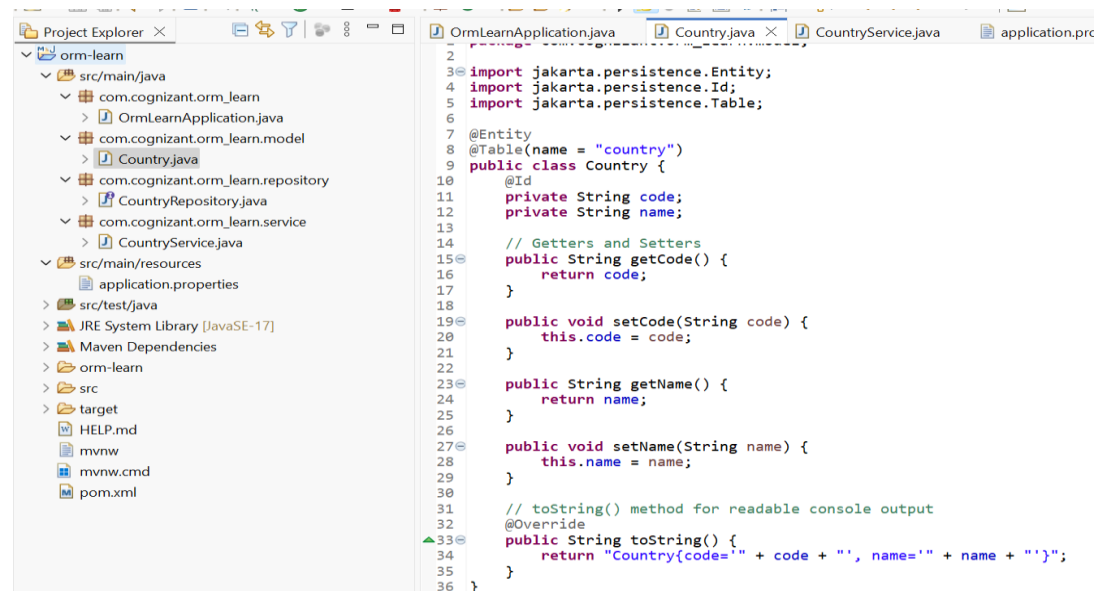
mysql> USE ormlearn;
Database changed
mysql> CREATE TABLE country (
  ->     code VARCHAR(2) PRIMARY KEY,
  ->     name VARCHAR(50)
  -> );
Query OK, 0 rows affected (0.05 sec)

mysql> INSERT INTO country VALUES ('IN', 'India');
Query OK, 1 row affected (0.02 sec)

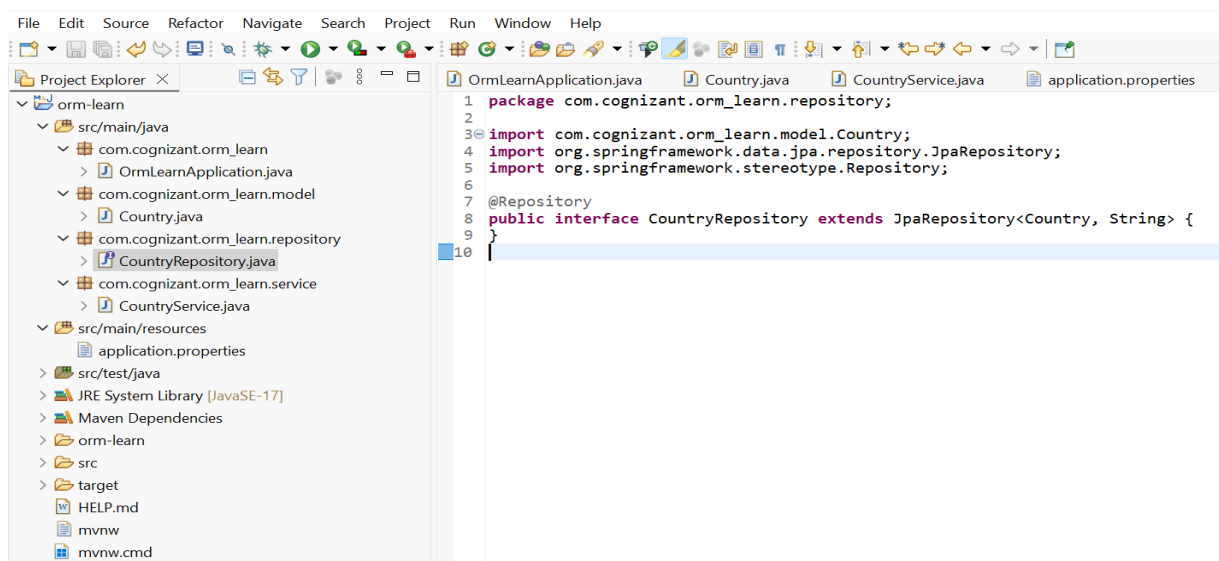
mysql> INSERT INTO country VALUES ('US', 'United States of America');
Query OK, 1 row affected (0.01 sec)

mysql> |
```

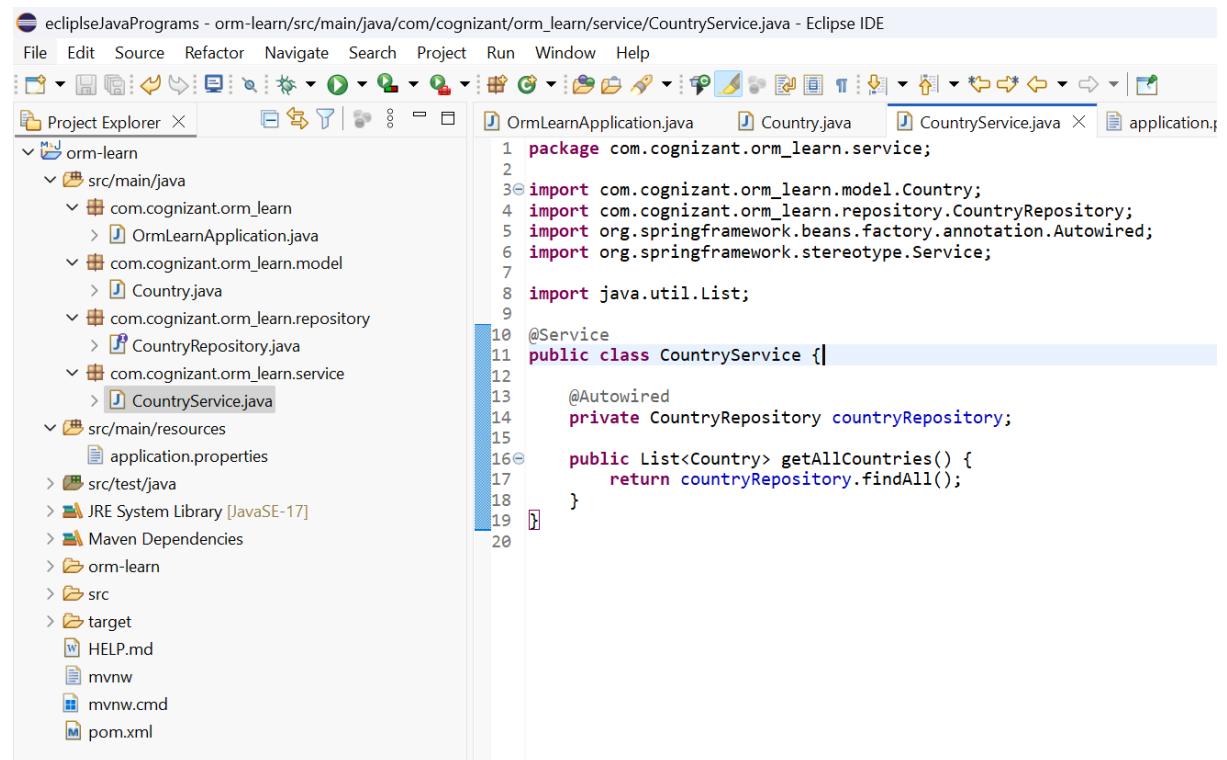
## 8.Create Entity Class



## 9.Create Repository



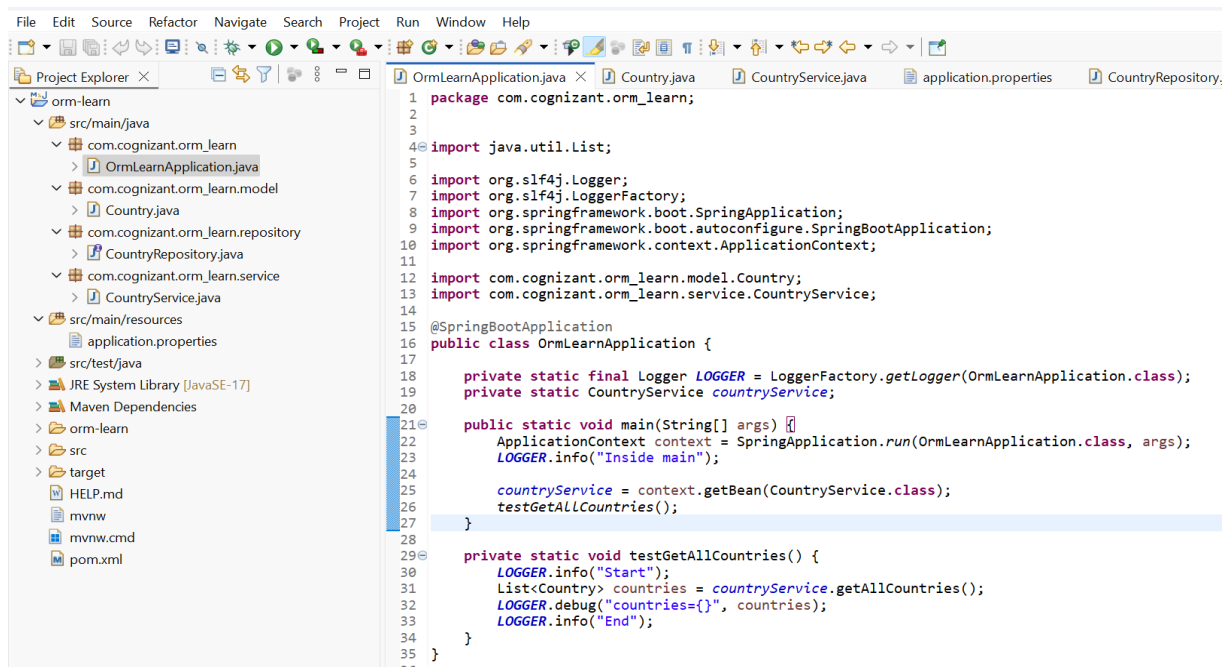
## 10. Create Service



The screenshot shows the Eclipse IDE with the Project Explorer on the left and the Editor on the right. The Project Explorer shows the project structure for 'orm-learn', including the 'com.cognizant.orm\_learn.service' package. The Editor shows the code for 'CountryService.java'.

```
1 package com.cognizant.orm_learn.service;
2
3 import com.cognizant.orm_learn.model.Country;
4 import com.cognizant.orm_learn.repository.CountryRepository;
5 import org.springframework.beans.factory.annotation.Autowired;
6 import org.springframework.stereotype.Service;
7
8 import java.util.List;
9
10 @Service
11 public class CountryService {
12
13     @Autowired
14     private CountryRepository countryRepository;
15
16     public List<Country> getAllCountries() {
17         return countryRepository.findAll();
18     }
19
20 }
```

## 11. Modify OrmLearnApplication.java for Testing



The screenshot shows the Eclipse IDE with the Project Explorer on the left and the Editor on the right. The Project Explorer shows the project structure for 'orm-learn', including the 'com.cognizant.orm\_learn' package. The Editor shows the code for 'OrmLearnApplication.java'.

```
1 package com.cognizant.orm_learn;
2
3
4 import java.util.List;
5
6 import org.slf4j.Logger;
7 import org.slf4j.LoggerFactory;
8 import org.springframework.boot.SpringApplication;
9 import org.springframework.boot.autoconfigure.SpringBootApplication;
10 import org.springframework.context.ApplicationContext;
11
12 import com.cognizant.orm_learn.model.Country;
13 import com.cognizant.orm_learn.service.CountryService;
14
15 @SpringBootApplication
16 public class OrmLearnApplication {
17
18     private static final Logger LOGGER = LoggerFactory.getLogger(OrmLearnApplication.class);
19     private static CountryService countryService;
20
21     public static void main(String[] args) {
22         ApplicationContext context = SpringApplication.run(OrmLearnApplication.class, args);
23         LOGGER.info("Inside main");
24
25         countryService = context.getBean(CountryService.class);
26         testGetAllCountries();
27     }
28
29     private static void testGetAllCountries() {
30         LOGGER.info("Start");
31         List<Country> countries = countryService.getAllCountries();
32         LOGGER.debug("countries={}", countries);
33         LOGGER.info("End");
34     }
35
36 }
```

## 12.Run the Project

### Output:

```
30 Start
135 select c1_0.code,c1_0.name from country c1_0
32 countries=[Country{code='IN', name='India'}, Country{code='US', name='United States of America'}]
33 End
660 Closing JPA EntityManagerFactory for persistence unit 'default'
```

## 4)Difference between JPA, 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;
}
```

## OUTPUT :-

```
Hibernate: insert into employee (dept, name, salary) values (?, ?, ?)
Employee Saved with ID: 1
```

## Spring Data JPA

### EmployeeRepository.java

```
public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}
```

### EmployeeService.java

```
@Autowired
private EmployeeRepository employeeRepository;

@Transactional
public void addEmployee(Employee employee) {
    employeeRepository.save(employee);
}
```

## OUTPUT:-

```
Hibernate: insert into employee (dept, name, salary) values (?, ?, ?)
Saved via Spring Data JPA: 2
```