* SPRING DATA JPA- QUICK EXAMPLE:

**MySql Cli:**

mysql -u root -p

create schema ormlearn;

**Database:**

create table country (

co\_code varchar(2) primary key,

co\_name varchar(50)

);

insert into country values ('IN', 'India');

insert into country values ('US', 'United States of America');

application.properties:

# 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

# DB Config

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

# Hibernate

spring.jpa.hibernate.ddl-auto=validate

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5Dialect

**Building Project:**

mvn clean package -Dhttp.proxyHost=proxy.cognizant.com -Dhttp.proxyPort=6050 -Dhttps.proxyHost=proxy.cognizant.com -Dhttps.proxyPort=6050 -Dhttp.proxyUser=123456

**OrmLearnApplication.java**

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

public static void main(String[] args) {

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

countryService = context.getBean(CountryService.class);

LOGGER.info("Inside main");

testGetAllCountries();

}

**Country.java**

@Entity

@Table(name = "country")

public class Country {

@Id

@Column(name = "co\_code")

private String code;

@Column(name = "co\_name")

private String name;

}

**CountryRepository.java**

@Repository

public interface CountryRepository extends JpaRepository<Country, String> {

}

**CountryService.java**

@Service

public class CountryService {

@Autowired

private CountryRepository countryRepository;

@Transactional

public List<Country> getAllCountries() {

return countryRepository.findAll();

}

}

**Testing:**

private static CountryService countryService;

private static void testGetAllCountries() {

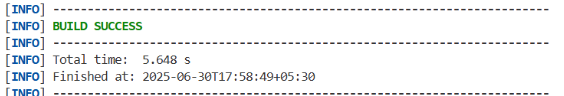
LOGGER.info("Start");

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

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

LOGGER.info("End");

}



* Difference Between JPA, Hibernate, and Spring Data JPA:

1. **JPA (Java Persistence API):**

* JPA is a specification, defined in JSR 338.
* It sets standards for ORM in Java (e.g., annotations like @Entity, @Id, @Table).
* JPA does not provide implementation, only rules or interfaces.
* You need a JPA provider like Hibernate to use JPA in real applications.

**2. Hibernate:**

- Hibernate is an ORM framework and the most popular JPA provider.

* It provides the actual implementation of JPA's interfaces.
* Hibernate also offers additional features beyond JPA, such as caching and custom SQL support.
* You can use Hibernate with or without JPA.

**3. Spring Data JPA:**

- Spring Data JPA is a Spring module that sits on top of JPA and Hibernate.

* It provides a high-level abstraction for JPA-based data access.
* Removes boilerplate code like DAO implementations.
* Supports derived query methods (e.g., findByEmail()), paging, and sorting.
* Internally, Spring Data JPA still uses Hibernate (or any JPA provider) for actual DB operations.