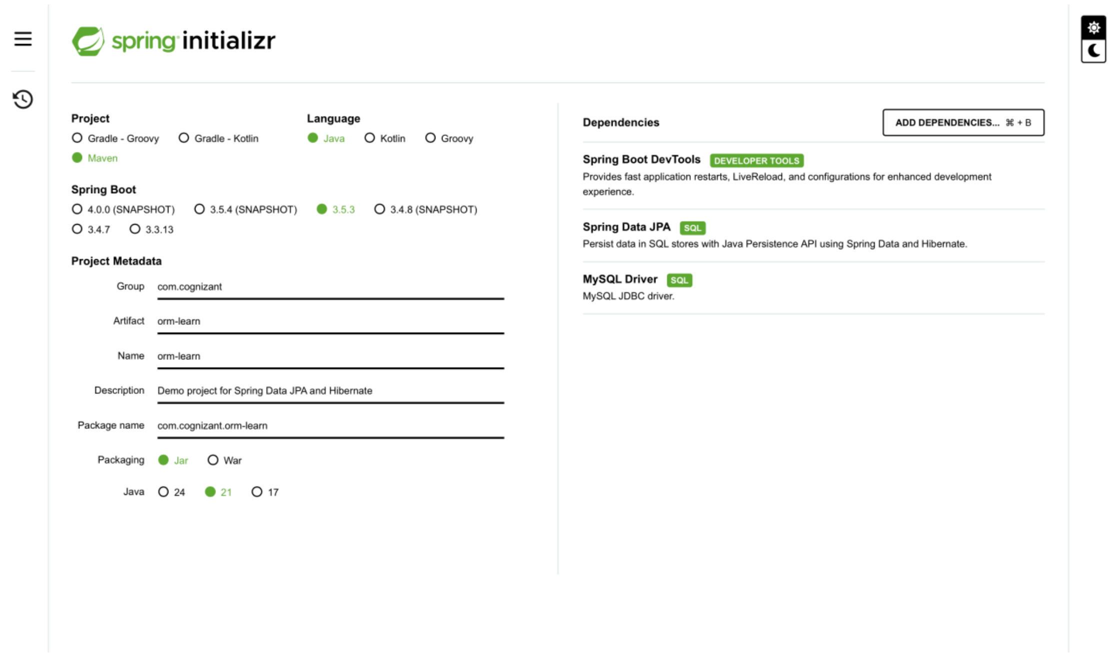
SuperSet ID : 6314413 **Shaik Balaji Mahammad Rafi**

HANDSON EXERCISES - WEEK 3

Skill : Spring Data JPA with Spring Boot, Hibernate

Exercise 1 : Spring Data JPA - Quick Example



spring initializr

Project
Gradle - Groovy Gradle - Kotlin [✓ Maven]

Language
[✓ Java] Kotlin Groovy

Spring Boot
4.0.0 (SNAPSHOT) 3.5.4 (SNAPSHOT) [✓ 3.5.3] 3.4.8 (SNAPSHOT) 3.4.7 3.3.13

Project Metadata

Group	com.cognizant
Artifact	orm-learn
Name	orm-learn
Description	Demo project for Spring Data JPA and Hibernate
Package name	com.cognizant.orm-learn
Packaging	[✓ Jar] War
Java	24 [✓ 21] 17

Dependencies

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.

CODE:

pom.xml :

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
                  https://maven.apache.org/xsd/maven-4.0.0.xsd">

   <modelVersion>4.0.0</modelVersion>

   <parent>
     <groupId>org.springframework.boot</groupId>
     <artifactId>spring-boot-starter-parent</artifactId>
     <version>3.1.3</version>
     <relativePath/>
   </parent>

   <groupId>com.cognizant</groupId>
   <artifactId>orm-learn</artifactId>
   <version>0.0.1-SNAPSHOT</version>
   <name>orm-learn</name>
   <description>Demo project for Spring Data JPA and Hibernate</description>

   <properties>
     <java.version>21</java.version>
   </properties>
```

```
  <dependencies>

    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-data-jpa</artifactId>
    </dependency>

    <dependency>
      <groupId>mysql</groupId>
      <artifactId>mysql-connector-java</artifactId>
      <version>8.0.33</version>
    </dependency>

    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-devtools</artifactId>
      <scope>runtime</scope>
      <optional>true</optional>
    </dependency>

    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-test</artifactId>
      <scope>test</scope>
    </dependency>

    <dependency>
      <groupId>javax.transaction</groupId>
      <artifactId>javax.transaction-api</artifactId>
      <version>1.3</version>
    </dependency>

  </dependencies>

  <build>
    <plugins>
      <plugin>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-maven-plugin</artifactId>
      </plugin>
    </plugins>
  </build>

</project>
```

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

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.url=jdbc:mysql://localhost:3306/ormlearn_fresh
spring.datasource.username=root
spring.datasource.password=root

spring.jpa.hibernate.ddl-auto=validate
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect
```

com.cognizant.orm_learn.model : Country.java :

```java
package com.cognizant.orm_learn.model;

import jakarta.persistence.Entity;
import jakarta.persistence.Table;
import jakarta.persistence.Id;
import jakarta.persistence.Column;

@Entity
@Table(name = "country")
public class Country {

    @Id
    @Column(name = "code")
    private String code;

    @Column(name = "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 + "]";
    }
}
```

com.cognizant.orm_learn.repository : 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> {
}
```

com.cognizant.orm_learn.service : CountryService.java :

```
package com.cognizant.orm_learn.service;

import java.util.List;
import javax.transaction.Transactional;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

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();
    }
}
```

com.cognizant.orm_learn : OrmLearnApplication.java :

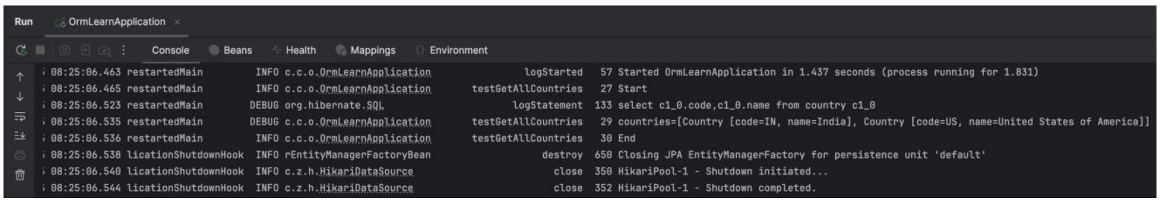
```
package com.cognizant.orm_learn;

import com.cognizant.orm_learn.model.Country;
import com.cognizant.orm_learn.service.CountryService;
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 java.util.List;
```

```
@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();
    }

    private static void testGetAllCountries() {
        LOGGER.info("Start");
        List<Country> countries = countryService.getAllCountries();
        LOGGER.debug("countries={}", countries);
        LOGGER.info("End");
    }
}
```

MYSQL : Country Table -

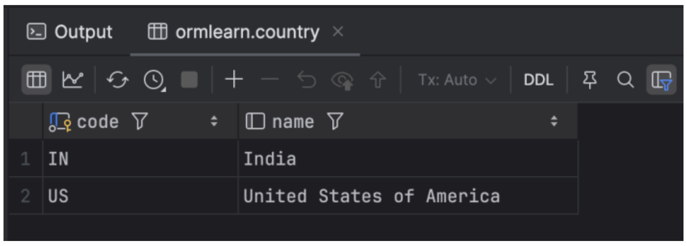
```
create table country (
  code varchar(2) primary key,
  name varchar(50)
);
insert into country values ('IN', 'India');
insert into country values ('US', 'United States of America');
```

OUTPUT:



```
08:25:06.463 restartedMain         INFO c.c.o.OrmLearnApplication          logStarted   57 Started OrmLearnApplication in 1.437 seconds (process running for 1.831)
08:25:06.465 restartedMain         INFO c.c.o.OrmLearnApplication     testGetAllCountries   27 Start
08:25:06.523 restartedMain        DEBUG org.hibernate.SQL                logStatement  133 select c1_0.code,c1_0.name from country c1_0
08:25:06.535 restartedMain        DEBUG c.c.o.OrmLearnApplication     testGetAllCountries   29 countries=[Country [code=IN, name=India], Country [code=US, name=United States of America]]
08:25:06.536 restartedMain         INFO c.c.o.OrmLearnApplication     testGetAllCountries   30 End
08:25:06.538 licationShutdownHook  INFO rEntityManagerFactoryBean              destroy  650 Closing JPA EntityManagerFactory for persistence unit 'default'
08:25:06.540 licationShutdownHook  INFO c.z.h.HikariDataSource                   close  350 HikariPool-1 - Shutdown initiated...
08:25:06.544 licationShutdownHook  INFO c.z.h.HikariDataSource                   close  352 HikariPool-1 - Shutdown completed.
```

SELECT * FROM COUNTRY;



	code	name
1	IN	India
2	US	United States of America

Exercise 4 : Difference between JPA, Hibernate and Spring Data JPA

A. Hibernate Manual configuration - wihtout Spring Boot

com.hibernate : Employee.java :

```java
package com.hibernate;

import javax.persistence.*;

@Entity
@Table(name = "EMPLOYEE")
public class Employee {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;

    private String firstName;
    private String lastName;
    private int salary;

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    public int getSalary() {
        return salary;
    }

    public void setSalary(int salary) {
        this.salary = salary;
    }
}
```

com.hibernate : ManageEmployee.java :

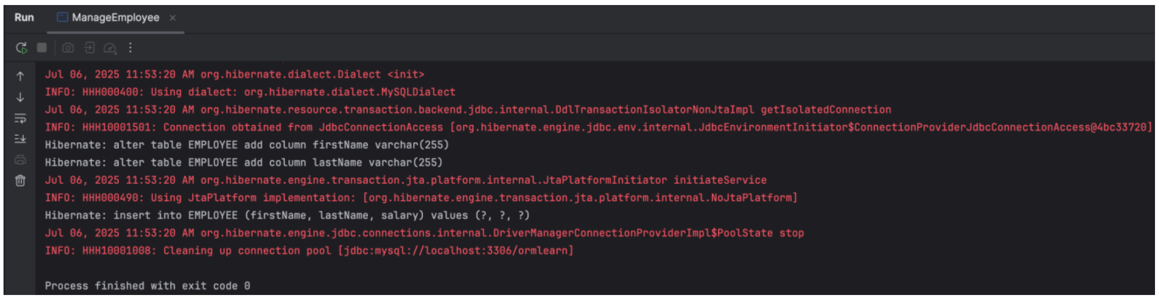
```
package com.hibernate;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class ManageEmployee {
    public static void main(String[] args) {
        SessionFactory factory = new Configuration()
                .configure("hibernate.cfg.xml")
                .addAnnotatedClass(Employee.class)
                .buildSessionFactory();

        Session session = factory.openSession();

        try {
            session.beginTransaction();
            Employee emp = new Employee();
            emp.setFirstName("Rafi");
            emp.setLastName("SBM");
            emp.setSalary(90000);
            session.save(emp);
            session.getTransaction().commit();
        } finally {
            session.close();
            factory.close();
        }
    }
}
```

resources/hibernate.cfg.xml :

```
<!DOCTYPE hibernate-configuration PUBLIC
        "-//Hibernate/Hibernate Configuration DTD//EN"
        "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">

<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/ormlearn</property>
        <property name="hibernate.connection.username">root</property>
        <property name="hibernate.connection.password">root</property>
        <property
name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
```

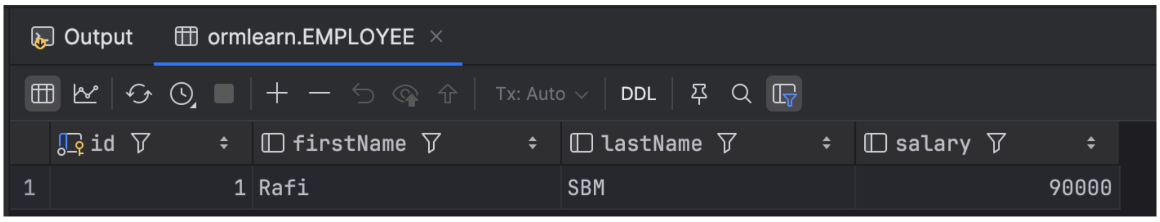
```
    <property name="hibernate.hbm2ddl.auto">update</property>
    <property name="show_sql">true</property>
    <mapping class="com.hibernate.Employee"/>

  </session-factory>
</hibernate-configuration>
```

OUTPUT :



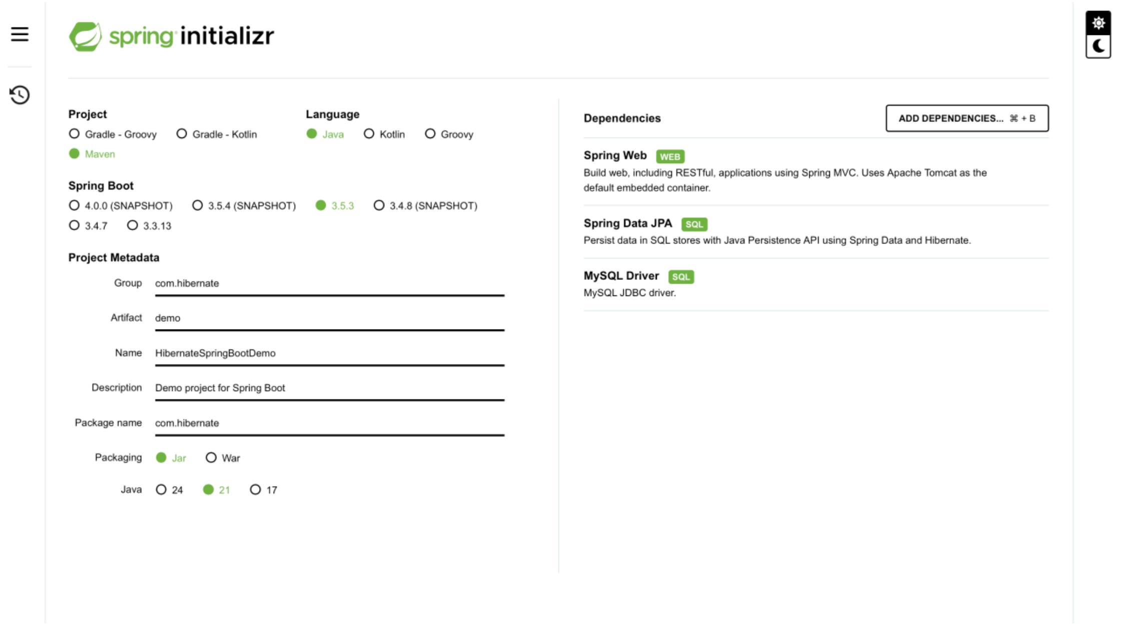
```
Jul 06, 2025 11:53:20 AM org.hibernate.dialect.Dialect <init>
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQLDialect
Jul 06, 2025 11:53:20 AM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdbcConnectionAccess@4bc33720]
Hibernate: alter table EMPLOYEE add column firstName varchar(255)
Hibernate: alter table EMPLOYEE add column lastName varchar(255)
Jul 06, 2025 11:53:20 AM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: insert into EMPLOYEE (firstName, lastName, salary) values (?, ?, ?)
Jul 06, 2025 11:53:20 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PoolState stop
INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost:3306/ormlearn]

Process finished with exit code 0
```

SELECT * FROM EMPLOYEE;



	id	firstName	lastName	salary
1	1	Rafi	SBM	90000

B. Spirng Boot + Sptring Data JPA



Project
○ Gradle - Groovy ○ Gradle - Kotlin [✓ Maven]

Language
[✓ Java] ○ Kotlin ○ Groovy

Spring Boot
○ 4.0.0 (SNAPSHOT) ○ 3.5.4 (SNAPSHOT) [✓ 3.5.3] ○ 3.4.8 (SNAPSHOT) ○ 3.4.7 ○ 3.3.13

Project Metadata

Group	com.hibernate
Artifact	demo
Name	HibernateSpringBootDemo
Description	Demo project for Spring Boot
Package name	com.hibernate
Packaging	[✓ Jar] ○ War
Java	○ 24 [✓ 21] ○ 17

Dependencies

Spring Web WEB
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

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

MySQL Driver SQL
MySQL JDBC driver.

com.hibernate: Employee.java :

```
package com.hibernate;

import jakarta.persistence.*;

@Entity
@Table(name = "employee")
public class Employee {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;

    private String firstName;
    private String lastName;
    private int salary;

    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }

    public String getFirstName() {
        return firstName;
    }
    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }
    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    public int getSalary() {
        return salary;
    }
    public void setSalary(int salary) {
        this.salary = salary;
    }
}
```

com.hibernate: EmployeeRepository.java :

```
package com.hibernate;

import org.springframework.data.jpa.repository.JpaRepository;
public interface EmployeeRepository extends JpaRepository<Employee, Integer> {
}
```

com.hibernate: EmployeeService.java :

```java
package com.hibernate;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import jakarta.transaction.Transactional;

@Service
public class EmployeeService {

    @Autowired
    private EmployeeRepository employeeRepository;

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

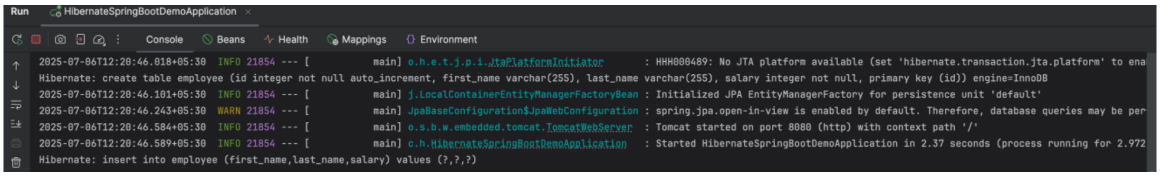
com.hibernate: HibernateSpringBootDemoApplication.java :

```java
package com.hibernate;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class HibernateSpringBootDemoApplication implements CommandLineRunner {

    @Autowired
    private EmployeeService employeeService;

    public static void main(String[] args) {
        SpringApplication.run(HibernateSpringBootDemoApplication.class, args);
    }

    @Override
    public void run(String... args) throws Exception {
        Employee emp = new Employee();
        emp.setFirstName("Rafi");
        emp.setLastName("SBM");
        emp.setSalary(60000);
        employeeService.addEmployee(emp);
    }
}
```

application.properties :

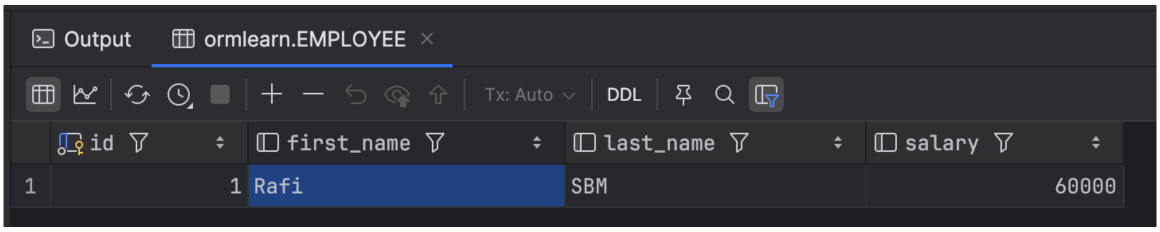
```
spring.datasource.url=jdbc:mysql://localhost:3306/ormlearn
spring.datasource.username=root
spring.datasource.password=root
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect
```

OUTPUT :



```
2025-07-06T12:20:46.018+05:30  INFO 21854 --- [           main] o.h.e.t.j.p.i.JtaPlatformInitiator       : HHH000489: No JTA platform available (set 'hibernate.transaction.jta.platform' to ena
Hibernate: create table employee (id integer not null auto_increment, first_name varchar(255), last_name varchar(255), salary integer not null, primary key (id)) engine=InnoDB
2025-07-06T12:20:46.101+05:30  INFO 21854 --- [           main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
2025-07-06T12:20:46.243+05:30  WARN 21854 --- [           main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries may be per
2025-07-06T12:20:46.584+05:30  INFO 21854 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port 8080 (http) with context path '/'
2025-07-06T12:20:46.589+05:30  INFO 21854 --- [           main] c.h.HibernateSpringBootDemoApplication   : Started HibernateSpringBootDemoApplication in 2.37 seconds (process running for 2.972
Hibernate: insert into employee (first_name,last_name,salary) values (?,?,?)
```

SELECT * FROM employee;



	id	first_name	last_name	salary
1	1	Rafi	SBM	60000

Exercise 2 : Hibernate XML Config implementation walk through

pom.xml :

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
   <modelVersion>4.0.0</modelVersion>

   <groupId>com.library</groupId>
   <artifactId>HibernateXMLDemo</artifactId>
   <version>1.0-SNAPSHOT</version>

   <properties>
      <maven.compiler.source>21</maven.compiler.source>
      <maven.compiler.target>21</maven.compiler.target>
      <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
   </properties>

   <dependencies>

      <dependency>
         <groupId>org.hibernate</groupId>
         <artifactId>hibernate-core</artifactId>
         <version>5.6.15.Final</version>
      </dependency>
```

```
        <dependency>
            <groupId>com.mysql</groupId>
            <artifactId>mysql-connector-j</artifactId>
            <version>8.2.0</version>
        </dependency>

        <dependency>
            <groupId>jakarta.persistence</groupId>
            <artifactId>jakarta.persistence-api</artifactId>
            <version>2.2.3</version>
        </dependency>

        <dependency>
            <groupId>org.slf4j</groupId>
            <artifactId>slf4j-simple</artifactId>
            <version>2.0.7</version>
        </dependency>
    </dependencies>
</project>
```

com.hibernate.xmlconfig : Employee.java :

```
package com.hibernate.xmlconfig;

public class Employee {
    private int id;
    private String firstName;
    private String lastName;
    private int salary;

    public int getId() {
            return id;
    }

    public void setId(int id) {
            this.id = id;
    }

    public String getFirstName() {
            return firstName;
    }

    public void setFirstName(String firstName) {
            this.firstName = firstName;
    }

    public String getLastName() {
            return lastName;
    }
    public void setLastName(String lastName) {
            this.lastName = lastName;
    }

    public int getSalary() {
            return salary;
    }

    public void setSalary(int salary) {
            this.salary = salary;
  }
}
```

com.hibernate.xmlconfig : ManageEmployee.java :

```
package com.hibernate.xmlconfig;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.cfg.Configuration;

import java.util.List;

public class ManageEmployee {
    public static void main(String[] args) {

        SessionFactory factory = new Configuration().configure().buildSessionFactory();
        Session session = factory.openSession();

        Transaction tx = null;

        try {
            tx = session.beginTransaction();

            Employee emp = new Employee();
            emp.setFirstName("Rafi");
            emp.setLastName("SBM");
            emp.setSalary(75000);
            session.save(emp);

            List<Employee> list = session.createQuery("FROM Employee", Employee.class).list();

            for (Employee e : list) {
                System.out.println(e.getId() + ": " + e.getFirstName() + " " + e.getLastName()
+ " - " + e.getSalary());
            }

            tx.commit();
        } catch (Exception e) {
            if (tx != null) tx.rollback();
            e.printStackTrace();
        } finally {
            session.close();
            factory.close();
        }
    }
}
```

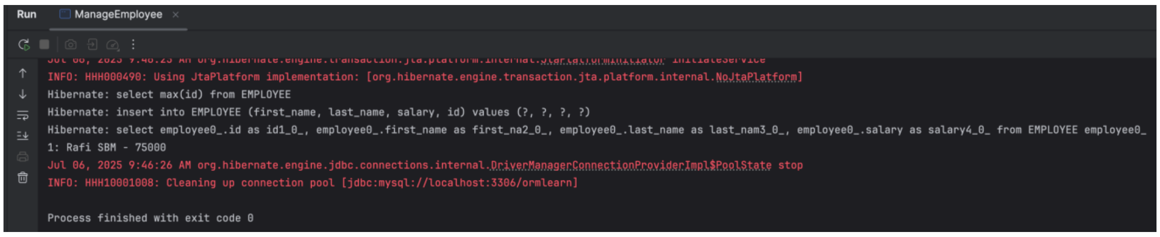
resources/hibernate.cfg.xml :

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE hibernate-configuration PUBLIC
        "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
        "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">

<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/ormlearn</property>
        <property name="hibernate.connection.username">root</property>
        <property name="hibernate.connection.password">root</property>


        <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
        <property name="hibernate.show_sql">true</property>
        <property name="hibernate.hbm2ddl.auto">update</property>

        <mapping resource="employee.hbm.xml"/>
    </session-factory>
</hibernate-configuration>
```

resources/employee.hbm.xml :

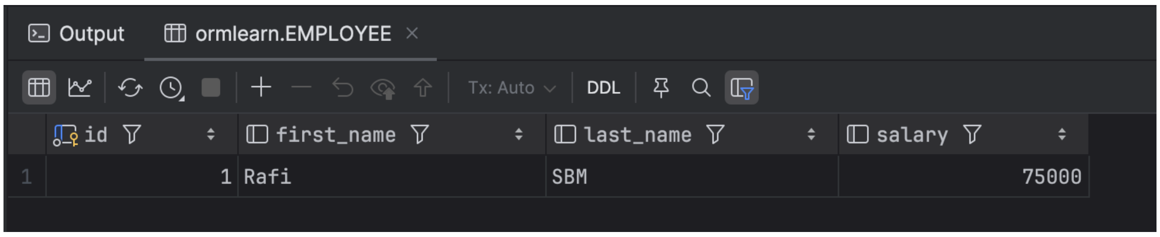
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-mapping PUBLIC
        "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
        "http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

<hibernate-mapping>
    <class name="com.hibernate.xmlconfig.Employee" table="EMPLOYEE">
        <id name="id" type="int" column="id">
            <generator class="increment"/>
        </id>
        <property name="firstName" column="first_name" type="string"/>
        <property name="lastName" column="last_name" type="string"/>
        <property name="salary" column="salary" type="int"/>
    </class>
</hibernate-mapping>
```

OUTPUT :



```
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: select max(id) from EMPLOYEE
Hibernate: insert into EMPLOYEE (first_name, last_name, salary, id) values (?, ?, ?, ?)
Hibernate: select employee0_.id as id1_0_, employee0_.first_name as first_na2_0_, employee0_.last_name as last_nam3_0_, employee0_.salary as salary4_0_ from EMPLOYEE employee0_
1: Rafi SBM - 75000
Jul 06, 2025 9:46:26 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PoolState stop
INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost:3306/ormlearn]

Process finished with exit code 0
```

SELECT * FROM EMPLOYEE;



	id	first_name	last_name	salary
1	1	Rafi	SBM	75000

Exercise 3 : Hibernate Annotation Config implementation walk through

com.hibernate.annotation : Employee.java :

```
package com.hibernate.annotation;

import javax.persistence.*;

@Entity
@Table(name = "EMPLOYEE")
public class Employee {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;

    @Column(name = "first_name")
    private String firstName;

    @Column(name = "last_name")
    private String lastName;

    @Column(name = "salary")
    private int salary;

      public int getId() {
            return id;
    }

    public void setId(int id) {
            this.id = id;
    }
    public String getFirstName() {
           return firstName;
    }

    public void setFirstName(String firstName) {
           this.firstName = firstName;
    }

    public String getLastName() {
           return lastName;
    }

    public void setLastName(String lastName) {
           this.lastName = lastName;
    }

    public int getSalary() {
           return salary;
    }

    public void setSalary(int salary) {
           this.salary = salary;
  }
}
```

com.hibernate.annotation : ManageEmployee.java :

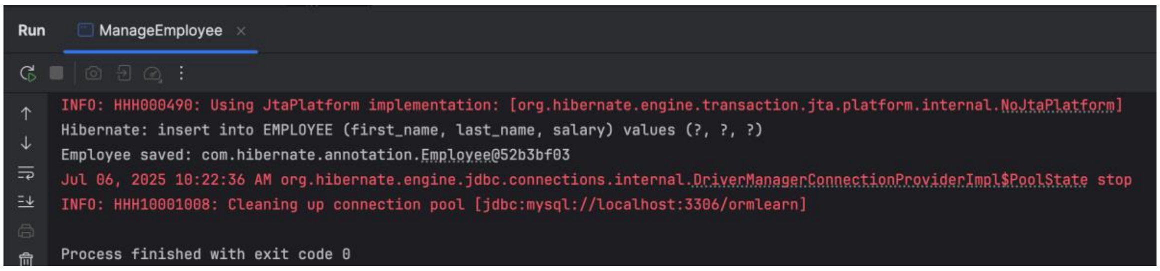
```
package com.hibernate.annotation;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class ManageEmployee {
    public static void main(String[] args) {
        SessionFactory factory = new Configuration()
                .configure("hibernate.cfg.xml")
                .addAnnotatedClass(Employee.class)
                .buildSessionFactory();

        Session session = factory.openSession();

        try {
            session.beginTransaction();
            Employee emp = new Employee();
            emp.setFirstName("Rafi");
            emp.setLastName("SBM");
            emp.setSalary(80000);
            session.save(emp);
            session.getTransaction().commit();
            System.out.println("Employee saved: " + emp);
        }
```

```
            finally {
                session.close();
                factory.close();
            }
        }
}
```

resources/hibernate.cfg.xml :

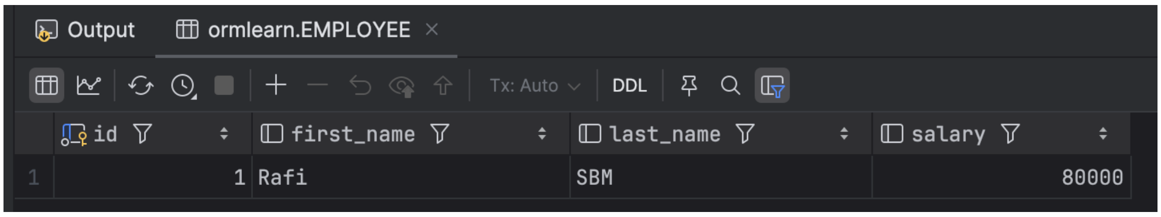
```
<!DOCTYPE hibernate-configuration PUBLIC
        "-//Hibernate/Hibernate Configuration DTD//EN"
        "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
    <session-factory>

        <property
name="hibernate.connection.driver_class">com.mysql.cj.jdbc.Driver</property>
        <property
name="hibernate.connection.url">jdbc:mysql://localhost:3306/ormlearn</property>
        <property name="hibernate.connection.username">root</property>
        <property name="hibernate.connection.password">root</property>

        <property  name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
        <property name="hibernate.hbm2ddl.auto">update</property>
        <property name="show_sql">true</property>

        <mapping class="com.hibernate.annotation.Employee"/>
    </session-factory>
</hibernate-configuration>
```

OUTPUT :



```
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: insert into EMPLOYEE (first_name, last_name, salary) values (?, ?, ?)
Employee saved: com.hibernate.annotation.Employee@52b3bf03
Jul 06, 2025 10:22:36 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PoolState stop
INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost:3306/ormlearn]

Process finished with exit code 0
```

SELECT * FROM EMPLOYEE;



	id	first_name	last_name	salary
1	1	Rafi	SBM	80000