Day28 JPA Implementation – Hibernate. (ORM)

Agenda

How to do the CRUD operation using Hibernate.

Different types of Mapping in Hibernate (XML based mapping & Annotation based Mapping)

HQL vs JPQL

Entity Relationships (One to One, One to Many & Many to Many)

Day 27 JPA Implementation (EclipseLink)

1. Create JPA project directly in Eclipse EE IDE.
2. JPQL (Java Persistence Query Lang) – This is DB independent and also uses Java Objects instead of Table names.
3. EntityManagerFactory & EntityManager – persist, find/get, merge, remove. (CRUD)
4. JPA Entity – Persistence.xml (Configuration file name)
5. CRUD example using EclipseLink.
6. BeginTransaction, do\_actual\_process, commitTransaction.

SpringBoot is called as Opinionated Framework.

1. By Adding dependency in pom.xml
2. Customize it’s behavior using application.properties file or yml file.
3. Customize using POJO classes.

Hibernate (Very popular Open Source ORM framework)

1. SessionFactory & Session
2. Hibernate.cfg.xml (default config file name for hibernate ORM)
3. Caching, Connection Pooling

Create Maven based Java Project.

Each maven project is uniquely identified by two properties.

groupId, artifactId

groupId is the reverse of company url (com.sutherland)

artifactId (name of project)

hibernate.cfg.xml

<?xml version="1.0" encoding="utf-8"?>

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

<hibernate-configuration>

<session-factory>

<!-- Database Connection -->

<property name="hibernate.connection.url">jdbc:mysql://localhost:3306/hibernate</property>

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

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

<property name="hibernate.connection.driver\_class">com.mysql.cj.jdbc.Driver</property>

<!-- Hibernate Settings -->

<property name="hibernate.show\_sql">true</property>

<property name="hibernate.format\_sql">true</property>

<property name="hibernate.hbm2ddl.auto">update</property>

<mapping class="com.example.entity.Employee"/>

<mapping class="com.example.entity.Address"/>

<mapping class="com.example.entity.Department"/>

<mapping class="com.example.entity.Student"/>

<mapping class="com.example.entity.Course"/>

</session-factory>

</hibernate-configuration>

Pom.xml

<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.example</groupId>

<artifactId>day28hbndemo</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>day28hbndemo</name>

<url>http://maven.apache.org</url>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<maven.compiler.source>21</maven.compiler.source>

<maven.compiler.target>21</maven.compiler.target>

</properties>

<dependencies>

<dependency>

<groupId>org.hibernate.orm</groupId>

<artifactId>hibernate-core</artifactId>

<version>6.2.7.Final</version>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.33</version>

</dependency>

</dependencies>

</project>

|  |  |  |  |
| --- | --- | --- | --- |
| Sl No | Operation Name | Hibernate Operation | DB Operation |
| 1 | Create | Persist() | Insert |
| 2 | Read | Find()/get() | Select |
| 3 | Update | Merge() | Update |
| 4 | Delete | Remove() | Delete |

Hibernate:

create table tbl\_airline (

id integer not null auto\_increment,

mobile bigint not null,

name varchar(255),

url varchar(255),

primary key (id)

) engine=InnoDB

Hibernate:

insert

into

tbl\_airline

(mobile,name,url)

values

(?,?,?)

Creating Hibernate Project

1. Create a Maven based project in Eclipse/STS
2. Add Hibernate and Database Dependencies in the pom.xml file (Force update it to add all the dependencies in the buildpath)
3. Also update the Java version (21 is recommended)
4. Create hibernate.cfg.xml file in “src/main/resources” folder (create resources folder if not available)
5. Add the DB related properties in the hibernate.cfg.xml file.
6. Create Entity Bean class with constructor and getters & setters
7. Create a starter class to perform the CRUD operation.

<dependency>

<groupId>com.microsoft.sqlserver</groupId>

<artifactId>mssql-jdbc</artifactId>

<scope>runtime</scope>

</dependency>

Primary Key, Foreign Key.

One to One, (Employee – Address)

One to Many (Department – Employees)

Many to Many (Courses – Students)

Hibernate:

create table address\_tbl (

id integer not null auto\_increment,

city varchar(255),

country varchar(255),

state varchar(255),

primary key (id)

) engine=InnoDB

Hibernate:

create table Course (

id integer not null auto\_increment,

title varchar(255),

primary key (id)

) engine=InnoDB

Hibernate:

create table Department (

id integer not null auto\_increment,

name varchar(255),

primary key (id)

) engine=InnoDB

Hibernate:

create table employee\_tbl (

id integer not null auto\_increment,

email varchar(255),

name varchar(255),

address\_id integer,

department\_id integer,

primary key (id)

) engine=InnoDB

Hibernate:

create table Student (

id integer not null auto\_increment,

name varchar(255),

primary key (id)

) engine=InnoDB

Hibernate:

create table student\_course (

student\_id integer not null,

course\_id integer not null,

primary key (student\_id, course\_id)

) engine=InnoDB

Hibernate:

alter table employee\_tbl

drop index UK\_3jvqbl8c5ti8u53b14vxd1av0

Hibernate:

alter table employee\_tbl

add constraint UK\_3jvqbl8c5ti8u53b14vxd1av0 unique (address\_id)

Hibernate:

alter table employee\_tbl

add constraint FKi99fqjcgrkrb1t3w12pn82n39

foreign key (address\_id)

references address\_tbl (id)

Hibernate:

alter table employee\_tbl

add constraint FKh98ri028uvypegvkyh2079h25

foreign key (department\_id)

references Department (id)

Hibernate:

alter table student\_course

add constraint FKtao0ysp4xfmi9dlxc10ku58qq

foreign key (course\_id)

references Course (id)

Hibernate:

alter table student\_course

add constraint FKpwtumr5ofgydxbuhmvdql4fye

foreign key (student\_id)

references Student (id)

Hibernate:

insert

into

address\_tbl

(city,country,state)

values

(?,?,?)

Hibernate:

insert

into

employee\_tbl

(address\_id,department\_id,email,name)

values

(?,?,?,?)

Hibernate:

insert

into

Department

(name)

values

(?)

Hibernate:

insert

into

Student

(name)

values

(?)

Hibernate:

insert

into

Course

(title)

values

(?)

Hibernate:

insert

into

Course

(title)

values

(?)

Hibernate:

update

employee\_tbl

set

address\_id=?,

department\_id=?,

email=?,

name=?

where

id=?

Hibernate:

insert

into

student\_course

(student\_id,course\_id)

values

(?,?)

Hibernate:

insert

into

student\_course

(student\_id,course\_id)

values

(?,?)

✅ Data saved successfully.