@Embeddable

The @Embeddable annotation is used to mark a class as being embeddable, meaning its properties can be included in another class as a value type. The class marked with @Embeddable is called the embeddable class.

Hibernate.cfg.xml:

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

<!DOCTYPE hibernate-configuration SYSTEM

"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

<session-factory>

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

<property name=*"connection.url"*>jdbc:mysql://localhost:3306/empdb?useSSL=false</property>

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

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

<property name=*"dialect"*>org.hibernate.dialect.MySQL5Dialect</property>

<property name=*"hbm2ddl.auto"*>create</property>

<property name=*"show\_sql"*>true</property>

<!-- tag added to create a new table from the class. -->

<mapping class=*"com.hibersessions.Student"* />

<mapping class=*"com.hibersessions.Address"* />

<mapping class=*"com.hibersessions.StudentEmbed"* />

</session-factory>

</hibernate-configuration>

**package** com.hibersessions;

**import** javax.persistence.Entity;

**import** javax.persistence.Id;

@Entity

**public** **class** StudentEmbed {

@Id

**private** **int** id;

**private** String name;

**private** String city;

**private** Certificate certi;

**public** StudentEmbed(**int** id, String name, String city, Certificate certi) {

**super**();

**this**.id = id;

**this**.name = name;

**this**.city = city;

**this**.certi = certi;

}

**public** StudentEmbed() {

**super**();

}

**public** **int** getId() {

**return** id;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getCity() {

**return** city;

}

**public** **void** setCity(String city) {

**this**.city = city;

}

**public** Certificate getCerti() {

**return** certi;

}

**public** **void** setCerti(Certificate certi) {

**this**.certi = certi;

}

@Override

**public** String toString() {

**return** "StudentEmbed [id=" + id + ", name=" + name + ", city=" + city + "]";

}

}

**package** com.hibersessions;

**import** javax.persistence.Embeddable;

@Embeddable

**public** **class** Certificate {

**private** String course;

**private** String duration;

**public** Certificate(String course, String duration) {

**super**();

**this**.course = course;

**this**.duration = duration;

}

**public** Certificate() {

**super**();

}

**public** String getCourse() {

**return** course;

}

**public** **void** setCourse(String course) {

**this**.course = course;

}

**public** String getDuration() {

**return** duration;

}

**public** **void** setDuration(String duration) {

**this**.duration = duration;

}

}

**package** com.hibersessions;

**import** org.hibernate.Session;

**import** org.hibernate.SessionFactory;

**import** org.hibernate.Transaction;

**import** org.hibernate.cfg.Configuration;

**public** **class** CallingEmbeddable {

**public** **static** **void** main(String[] args) {

SessionFactory factory = **new** Configuration().configure("hibernate.cfg.xml").buildSessionFactory();

StudentEmbed stu1 = **new** StudentEmbed();

stu1.setId(12345);

stu1.setName("Nagesh");

stu1.setCity("Khammam");

Certificate certificate1 = **new** Certificate();

certificate1.setCourse("Python");

certificate1.setDuration("2 months");

stu1.setCerti(certificate1);

Session s = factory.openSession();

Transaction tx = s.beginTransaction();

s.save(stu1);

tx.commit();

s.close();

factory.close();

System.***out***.println("Data inserted successfully!!!");

}

}

Oct 16, 2023 12:11:33 AM org.hibernate.Version logVersion

INFO: HHH000412: Hibernate ORM core version 5.6.15.Final

Oct 16, 2023 12:11:34 AM org.hibernate.annotations.common.reflection.java.JavaReflectionManager <clinit>

INFO: HCANN000001: Hibernate Commons Annotations {5.1.2.Final}

Oct 16, 2023 12:11:35 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl configure

WARN: HHH10001002: Using Hibernate built-in connection pool (not for production use!)

Oct 16, 2023 12:11:35 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator

INFO: HHH10001005: using driver [com.mysql.cj.jdbc.Driver] at URL [jdbc:mysql://localhost:3306/empdb?useSSL=false]

Oct 16, 2023 12:11:35 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator

INFO: HHH10001001: Connection properties: {password=\*\*\*\*, user=root}

Oct 16, 2023 12:11:35 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator

INFO: HHH10001003: Autocommit mode: false

Oct 16, 2023 12:11:35 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PooledConnections <init>

INFO: HHH000115: Hibernate connection pool size: 20 (min=1)

Oct 16, 2023 12:11:35 AM org.hibernate.dialect.Dialect <init>

INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQL5Dialect

Hibernate: drop table if exists stu\_address

Oct 16, 2023 12:11:36 AM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection

INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdbcConnectionAccess@444f44c5] for (non-JTA) DDL execution was not in auto-commit mode; the Connection 'local transaction' will be committed and the Connection will be set into auto-commit mode.

Hibernate: drop table if exists student\_details

Hibernate: drop table if exists StudentEmbed

Hibernate: create table stu\_address (address\_id integer not null auto\_increment, CityName varchar(30), image longblob, isOpen bit not null, street\_renvo\_dt date, stu\_street varchar(50), primary key (address\_id)) engine=MyISAM

Oct 16, 2023 12:11:36 AM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection

INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdbcConnectionAccess@3a894088] for (non-JTA) DDL execution was not in auto-commit mode; the Connection 'local transaction' will be committed and the Connection will be set into auto-commit mode.

Hibernate: create table student\_details (stuId integer not null, marks integer not null, stuName varchar(255), primary key (stuId)) engine=MyISAM

Hibernate: create table StudentEmbed (id integer not null, course varchar(255), duration varchar(255), city varchar(255), name varchar(255), primary key (id)) engine=MyISAM

Oct 16, 2023 12:11:37 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 StudentEmbed (course, duration, city, name, id) values (?, ?, ?, ?, ?)

Oct 16, 2023 12:11:37 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PoolState stop

INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost:3306/empdb?useSSL=false]

Data inserted successfully!!!

