在pom.xml中，<dependency>中还引入了<scope>，它主要管理依赖的部署。目前<scope>可以使用5个值：

compile，缺省值，适用于所有阶段，会随着项目一起发布。

provided，类似compile，期望JDK、容器或使用者会提供这个依赖。如servlet.jar。

runtime，只在运行时使用，如JDBC驱动，适用运行和测试阶段。

test，只在测试时使用，用于编译和运行测试代码。不会随项目发布。

system，类似provided，需要显式提供包含依赖的jar，Maven不会在Repository中查找它。

1.添加hibernate依赖(hibernate 4.3.11) 数据库驱动

2.创建hibernate.cfg.xml主配置文件(在src/main/resource)

3.创建实体类（持久化类）和映射文件 （hbm.xml）(在src/main/java)

3.1 映射文件（.hbm.xml） 说明： class 节点中 name 对应实体类的全类名 table 对应是表名

id 代表是主键列映射 name代表是实体类中的属性 column 代表是字段 type 代表是属性的数据类型 property 代表是非主键列映射 name 代表是实体类中的属性 column 代表是字段 type 代表是属性的数据类型

4.在hibernate.cfg.xml中导入映射文件（hbm.xml）

5.使用hibernate api 进行数据库操作

/\* pom.xml \*/

<properties>

<hibernate.version>4.3.11.Final</hibernate.version>

<mysql.version>5.1.39</mysql.version>

<junit.version>4.12</junit.version>

</properties>

<dependencies>

<!-- hibernate-core -->

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

<version>${hibernate.version}</version>

<!-- 4 重大版本更新 3 小版本的更新 11 4.3 版本进行了多少次的修复 -->

</dependency>

<dependency>

<groupId>mysql</groupId>

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

<version>${mysql.version}</version>

</dependency>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>${junit.version}</version>

<scope>test</scope>

</dependency>

</dependencies>

/\* hibernate.cfg.xml \*/

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

<!DOCTYPE hibernate-configuration PUBLIC

"-//Hibernate/Hibernate Configuration DTD 3.0//EN"

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

<hibernate-configuration>

<session-factory>

<!-- 主要配置数据库的连接信息 -->

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

<property name="connection.password">123456</property>

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

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

<!-- 方言 -->

<property name="dialect">org.hibernate.dialect.MySQLDialect</property>

<!-- 显示sql -->

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

<!-- 格式化sql -->

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

<!-- 创建数据库连接(session)的方式

每次打开数据库连接的时候，都会从线程池中查看有没有已经打开的数据库连接，

如果有的话，就用现成的数据库连接。如果没有的话，就新创建一个新的连接，然后放入线程池中，

用这种方法打开的数据库连接，不需要.close关闭，只需要.commit()就可以关闭数据库连接。

-->

<property name="current\_session\_context\_class">thread</property>

<mapping resource="com/lanou/entity/Type.hbm.xml"/>

<mapping resource="com/lanou/entity/Users.hbm.xml"/>

</session-factory>

</hibernate-configuration>

/\* Type.java \*/

package com.lanou.entity;

import java.io.Serializable;

public class Type implements Serializable {

/\*\*

\* 因为数据库字段默认都是null

\* 所以实体类每个属性，都要采用包装类型作为属性的数据类型

\*/

private Integer id;

private String name;

public Integer getId() {

return id;

}

public void setId(Integer id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

@Override

public String toString() {

return "Type [id=" + id + ", name=" + name + "]";

}

}

/\* Type.hbm.xml \*/

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

<!DOCTYPE hibernate-mapping PUBLIC

"-//Hibernate/Hibernate Mapping DTD 3.0//EN"

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

<hibernate-mapping>

<!-- 类名对应表名

属性对应列名

-->

<class name="com.lanou.entity.Type" table="type">

<!-- increment -->

<id name="id" column="id" type="java.lang.Integer">

<generator class="increment"></generator>

</id>

<property name="name" column="name" type="java.lang.String"></property>

</class>

</hibernate-mapping>