



JavaEE 实验五

实验报告

姓　　名　　 Mcrivers

班　　级　　 软件工程2102班

学　　号

提交日期　　 2023.12.10

目录

[一、基础实验——Hibernate 框架搭建 1](#_Toc184215750)

[1. 运行结果截图 1](#_Toc184215751)

[2. 观察工程Hibernate 配置文件hibernate.cfg.xml，查找相关资料，总结配置文件中各元素及其属性的作用，并记录下来 2](#_Toc184215752)

[3. 观察工程 hibernate-prj1 中的 JAVA 持久化类 Customer.java、Hibernate 映射 文件 Customer.hbm.xml，总结持久化类与数据库表的映射关系，以及映射文件中主要元素及其属性的作用 3](#_Toc184215753)

[4. 根据实验过程，思考实验一的扩展实验中的 DAO类与本实验中 DAO 类的区别 3](#_Toc184215754)

[5. 碰到的问题及解决方案或思考 4](#_Toc184215755)

[6. 实验收获及总结 4](#_Toc184215756)

[二、提高实验——持久化对象与 Hibernate 映射文件 5](#_Toc184215757)

[1. 运行结果截图 5](#_Toc184215758)

[2. 总结 POJO 模式下持久化类的规范 6](#_Toc184215759)

[3. 结合实验过程，查找相关资料，总结映射文件中主要元素（如 class、id、generator、property）及其属性的含义与作用 7](#_Toc184215760)

[4. 结合实验过程，查找相关资料，总结设置复合主键的方法和步骤 9](#_Toc184215761)

[5. 查找相关资料，总结 Hibernate 映射文件中主键各种生成策略的作用 9](#_Toc184215762)

[6. 结合实验过程，总结粒度设计的方法及特点 10](#_Toc184215763)

[7. 碰到的问题及解决方案或思考 11](#_Toc184215764)

[8. 实验收获及总结 11](#_Toc184215765)

[三、扩展实验——HQL 语言 11](#_Toc184215766)

[1. 运行结果截图 11](#_Toc184215767)

[2.总结 HQL 的常用语句及语法规则 13](#_Toc184215768)

[3. 碰到的问题及解决方案或思考 15](#_Toc184215769)

[4. 实验收获及总结 15](#_Toc184215770)

## 一、基础实验——Hibernate 框架搭建

## 1. 运行结果截图

**12、运行测试类 HibernateTest，观察控制台的输出，并记录运行结果**

观察到，login.jsp页面可以正常访问。

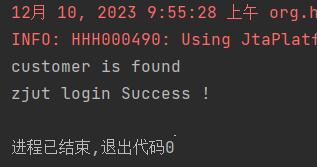


图1.1 数据库登录成功界面

**14、运行测试类 HibernateTest，观察控制台的输出，并记录运行结果**

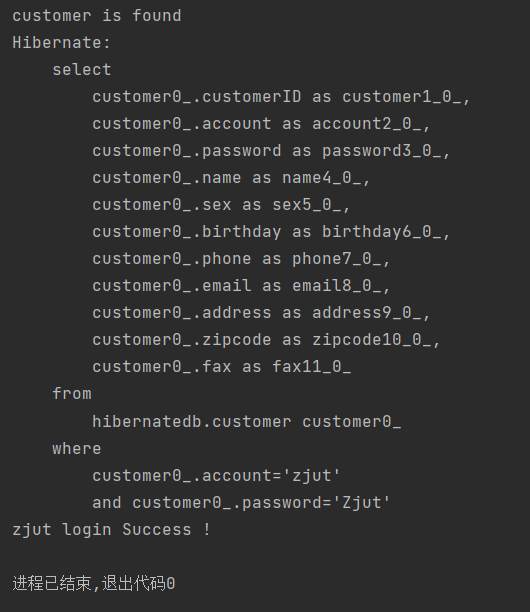


图1.2 SQL语句的打印

**19、运行测试类 HibernateTest**

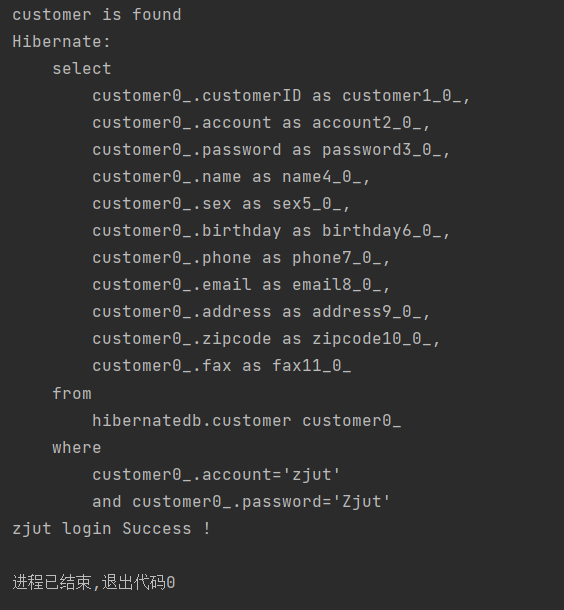


图1.2 测试结果

## 2. 观察工程Hibernate 配置文件hibernate.cfg.xml，查找相关资料，总结配置文件中各元素及其属性的作用，并记录下来

Ⅰ. <hibernate-configuration> 元素：表示Hibernate的配置文件的起始点

Ⅱ. <session-factory> 元素：用于配置Hibernate会话工厂负责创建和管理数据库连接以及Hibernate会话对象

Ⅲ.<property> 元素：用于设置Hibernate的配置属性

① hibernate.connection.driver\_class：指定数据库驱动程序的类名

② hibernate.connection.url：指定数据库的连接URL

③ hibernate.connection.username：指定连接数据库所需的用户名

④ hibernate.connection.password：指定连接数据库所需的密码

⑤ hibernate.dialect：指定Hibernate与数据库交互时要使用的SQL方言

⑥ hibernate.show\_sql：控制是否在控制台上打印生成的SQL语句

⑦ hibernate.format\_sql：控制是否对打印的SQL语句进行格式化

## 3. 观察工程 hibernate-prj1 中的 JAVA 持久化类 Customer.java、Hibernate 映射 文件 Customer.hbm.xml，总结持久化类与数据库表的映射关系，以及映射文件中主要元素及其属性的作用

Ⅰ.持久化类与数据库表的映射关系

Customer类中的成员属性对应了数据库表中的字段（列）

Ⅱ.映射文件中主要元素及其属性的作用

① <class> 元素

name 属性指定了映射到的Java类的名称，即 cn.edu.zjut.po.Customer。

table 属性指定了映射到的数据库表的名称，即 "customer"

catalog 属性指定了数据库的目录，即 "hibernatedb"

② <id> 元素

name 属性指定了主键属性在Java类中的名称，即 customerId

type 属性指定了主键属性的数据类型，即 int

<column> 元素用于指定主键列的映射，包括name和generator

name 属性映射到数据库表的列名，即 "customerID"

generator 属性指定了主键生成策略，这里是 "assigned"，表示主键的值由应用程序手动分配

③ <property> 元素

name 属性指定了属性在Java类中的名称，如 account、password、name 等

type 属性指定了属性的数据类型，如 string、java.lang.Boolean、date 等

<column> 元素用于指定属性对应的数据库表列的映射，包括name、length和unique等

## 4. 根据实验过程，思考实验一的扩展实验中的 DAO类与本实验中 DAO 类的区别

Ⅰ.数据访问方式

①扩展实验中的DAO类使用了JDBC来访问数据库，需要手动加载数据库驱动程序，建立数据库连接，手动执行查询，手动处理连接和资源的关闭

②本实验中的DAO类使用了Hibernate框架来访问数据库，不需要手动加载驱动程序、建立连接和关闭连接，而是通过Hibernate的Session对象来管理这些操作

Ⅱ.数据库连接方式

①扩展实验中的DAO类的数据库连接信息在代码中固定，修改起来比较不方便

②本实验中的DAO类通过Hibernate的配置文件配置数据库连接信息，可以直接在配置文件中修改，修改比较方便

Ⅲ.查询语言

①扩展实验中的DAO类使用SQL语句进行查询

②本实验中的DAO类使用了HQL进行查询

Ⅳ. 异常处理

①扩展实验中的DAO类使用了传统的异常处理方式，通过捕获SQLException来处理数据库访问中的异常

②本实验中的DAO类使用了Hibernate提供的异常处理机制

## 5. 碰到的问题及解决方案或思考

碰到的问题：程序运行时，出现报错（如下图）

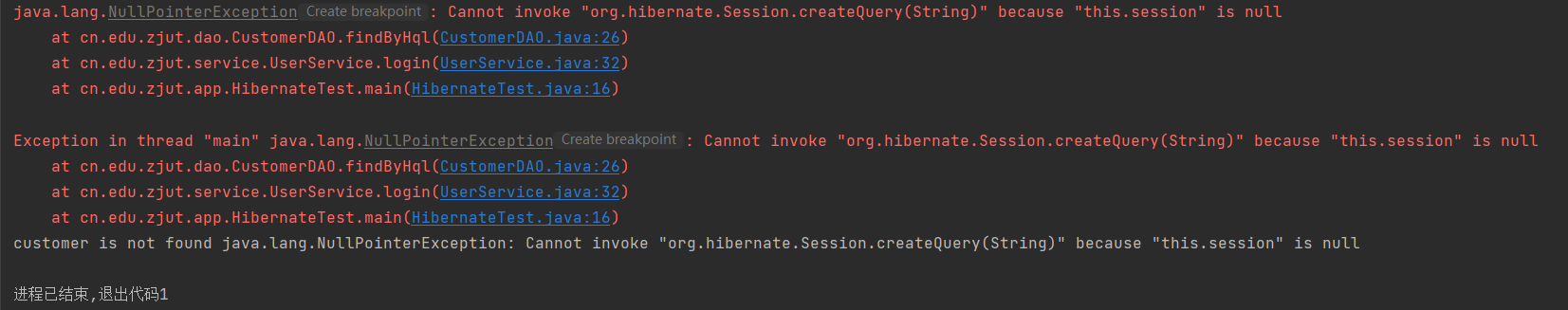


图1.3 报错情况

解决方法：经分析后发现是系统的MYSQL服务未启动；在“服务”中将“MYSQL80”服务开启后问题得到解决

## 6. 实验收获及总结

在这个实验中，我学会了搭建Hibernate开发环境的基本步骤，包括配置MySQL数据库、创建数据表、下载必要的库文件等。我还学习了如何使用Hibernate的Session对象来管理数据库连接和事务，并使用Hibernate的查询语言HQL来进行数据库查询。

## 二、提高实验——持久化对象与 Hibernate 映射文件

## 1. 运行结果截图

9、运行测试类 HibernateTest，观察控制台的输出，并记录运行结果

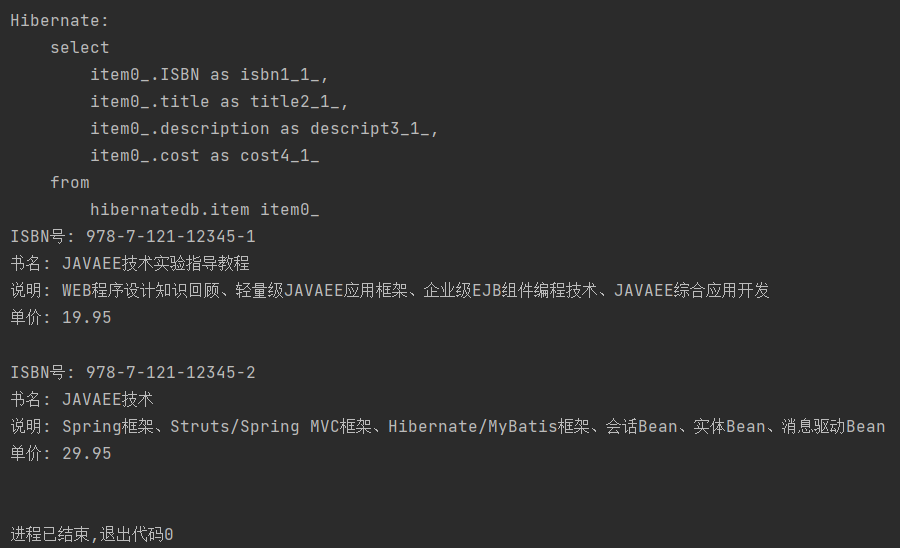


图2.1 运行结果

14、运行测试类 HibernateTest，观察控制台的输出，并记录运行结果



图2.2 运行结果

21、运行测试类 HibernateTest，观察数据库的变化，并记录运行结果

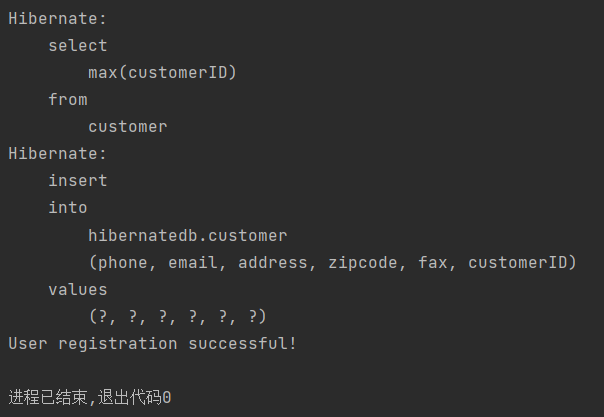


图2.3 注册成功（“？”是占位符）

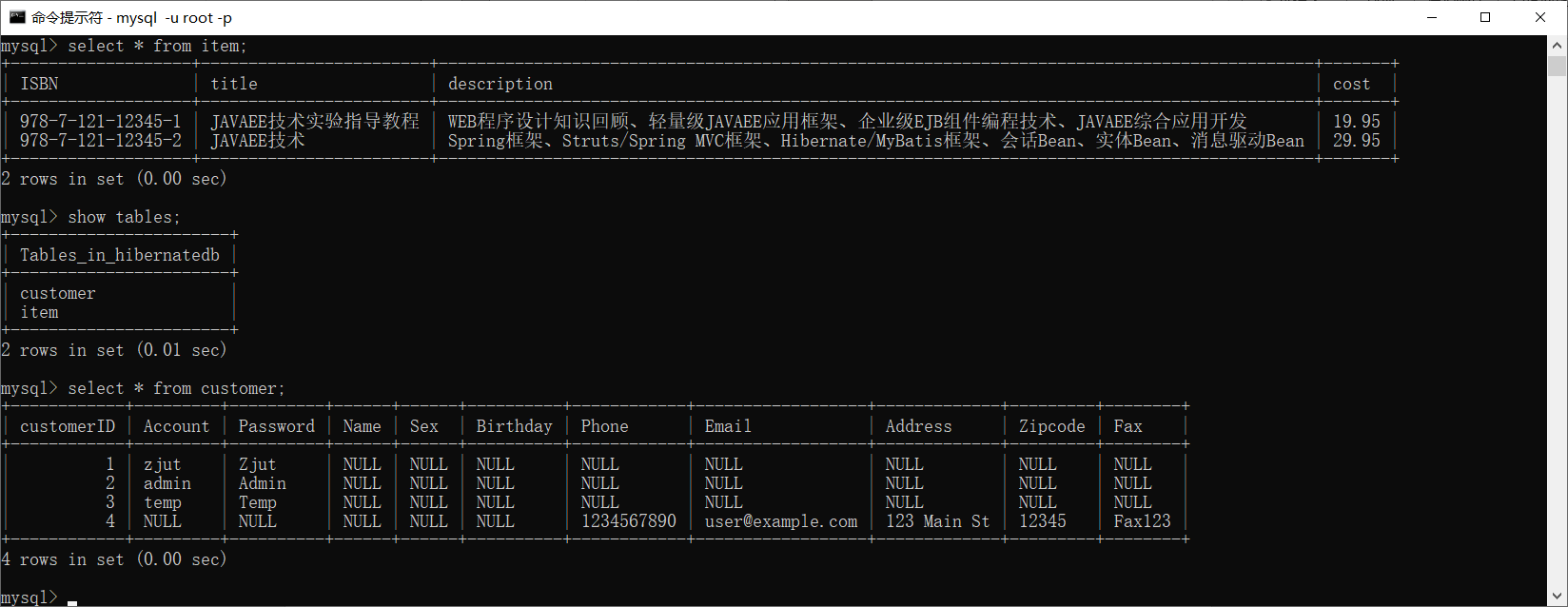


图2.4 数据库已经新增记录

## 2. 总结 POJO 模式下持久化类的规范

Ⅰ.命名规范

①类名应该是清晰、具有描述性的，通常与数据库表名对应。

②属性名应该与数据库表的列名对应。

③主键属性的命名应该明确表示它是主键。

Ⅱ.无参构造方法

每个持久化类应该提供一个无参数的构造函数。

Ⅲ.属性

为每个属性提供相应的setter和getter方法。

Ⅳ.主键：

可以使用注解或XML配置来指定主键生成策略。

Ⅴ.关联关系

①若类之间存在关联关系，需要在持久化类中定义相应的属性来表示这些关系。

②关联属性应该有适当的关联注解或配置。

Ⅵ.数据库映射

①使用注解或XML配置来指定持久化类与数据库表之间的映射关系，包括表名、列名、外键等。

②可以使用数据类型映射来确保Java属性正确映射到数据库列。

Ⅶ.序列化

如果需要将持久化类序列化，需要实现Serializable接口。

Ⅷ.日志和异常处理

①在持久化类中添加适当的日志记录，以便跟踪和调试数据库操作。

②处理数据库操作可能出现的异常，例如数据库连接问题或SQL语法错误。

Ⅸ.数据验证

在setter方法中可以添加数据验证逻辑，确保数据的合法性和完整性。

Ⅹ.生命周期方法

如果需要在持久化类的对象生命周期中执行特定的操作，可以使用生命周期方法，例如@PrePersist和@PreUpdate注解。

## 3. 结合实验过程，查找相关资料，总结映射文件中主要元素（如 class、id、generator、property）及其属性的含义与作用

Ⅰ.<hibernate-mapping> 元素：

含义：Hibernate映射文件的根元素，表示开始定义一个Hibernate映射文件。

属性：通常不包含属性，仅用于包裹其他元素。

Ⅱ.<class> 元素：

含义：用于定义一个持久化类（POJO）与数据库表之间的映射关系。

属性：

name：指定持久化类的完全限定名。

table：指定数据库表的表名。

catalog：指定数据库表所属的目录。

schema：指定数据库表所属的模式。

Ⅳ.<id> 元素：

含义：用于定义主键属性的映射关系。

属性：

name：指定主键属性的名称，通常与持久化类中的属性名对应。

type：指定主键属性的Java数据类型。

子元素：

<column>：用于指定主键列的映射关系，包括列名、长度、精度等信息。

<generator>：用于指定主键生成策略。

Ⅴ.<generator> 元素：

含义：用于指定主键生成策略。

属性：class：指定生成策略的类名。

Ⅵ.<property> 元素：

含义：用于定义普通属性（非主键属性）的映射关系。

属性：

name：指定属性的名称，通常与持久化类中的属性名对应。

type：指定属性的Java数据类型。

子元素：

<column>：用于指定属性对应的数据库列的映射关系，包括列名、长度、精度等信息。

Ⅶ.<component> 元素：

含义：用于定义一个组件属性，通常用于将多个属性组合成一个对象。

属性：

name：指定组件属性的名称，通常与持久化类中的属性名对应。

class：指定组件属性对应的Java类。

子元素：

<property>：定义组件属性中的子属性，具有与普通属性相同的属性和子元素。

## 4. 结合实验过程，查找相关资料，总结设置复合主键的方法和步骤

Ⅰ.创建复合主键的Java类

创建一个Java类来表示复合主键。这个类需要实现Serializable接口以确保它可以被序列化。

Ⅱ.定义复合主键的属性：

在复合主键的Java类中，定义每个组成复合主键的属性，并为这些属性创建相应的getter和setter方法。

Ⅲ.在持久化类中引用复合主键：

在持久化类中，引用复合主键的实例作为一个属性。

Ⅳ.在映射文件中配置复合主键：

在Hibernate映射文件中使用<composite-id>元素来配置复合主键。

在<composite-id>元素内部，使用<key-property>元素为每个复合主键属性指定相应的映射。

Ⅴ.将复合主键属性映射到数据库表：

在<key-property>元素中，使用<column>子元素来将复合主键属性映射到数据库表的列。

Ⅵ.执行增删改查操作：

使用Hibernate的会话执行增删改查操作，以确保复合主键属性已被正确地设置。

## 5. 查找相关资料，总结 Hibernate 映射文件中主键各种生成策略的作用

Ⅰ.assigned

使用此策略时，Hibernate期望应用程序为主键字段分配唯一的主键值。

Hibernate不会尝试生成或维护主键值，而是依赖应用程序提供的值。

Ⅱ.identity

使用此策略时，主键字段通常由数据库自动生成，通常是自增长的整数。

Hibernate在插入记录时不会为主键字段指定值，而是依赖数据库生成。

Ⅲ.sequence

此策略使用数据库中的序列来生成主键值。

序列是一个数据库对象，用于生成唯一的数字值，通常用于高并发环境。

Ⅳ.native

native策略会根据数据库类型自动选择identity或sequence策略。

Hibernate会根据数据库的支持情况选择最适合的主键生成策略。

Ⅴ.increment

此策略通过维护一个表来存储最大的主键值，然后递增该值来生成新的主键值。

适用于不支持自增长或序列的数据库。

Ⅵ.uuid

此策略使用UUID来生成主键值，通常以字符串形式存储。

主键值是全局唯一的，适用于分布式系统。

Ⅶ.guid

类似于UUID策略，使用全局唯一标识符来生成主键值，通常以字符串形式存储。

Ⅷ.foreign

此策略将一个对象的主键设置为另一个对象的外键，用于表示两个对象之间的关联。

适用于表示多对一或一对一关系的情况。

Ⅸ.sequence-identity：

此策略结合了sequence和identity策略。

它首先尝试使用identity策略，如果不支持则切换到sequence策略。

## 6. 结合实验过程，总结粒度设计的方法及特点

**方法：**

①数据表设计： 在数据库中首先设计表结构，确定各个字段的名称、类型和约束。这是粒度设计的起点，要根据业务需求合理地划分数据表。

②POJO类设计： 为每个数据表创建一个对应的POJO类，用于表示数据库中的记录。

③映射文件设计： 创建Hibernate映射文件，用于描述POJO类与数据表之间的映射关系。

④持久化操作类设计： 创建用于执行数据库操作的DAO类，包括增删改查等操作方法。每个DAO类应该与一个POJO类对应，用于对该POJO类的持久化操作。

⑤业务逻辑类设计：创建业务逻辑类，用于处理各种业务需求。

## 7. 碰到的问题及解决方案或思考

该实验中基本没有遇到问题。

## 8. 实验收获及总结

在该实验中，我熟悉了Hibernate开发方法、掌握了持久化类的设计、理解了Hibernate映射文件、了解了主键生成策略、掌握了复合主键的使用粒度设计。这些知识和技能对于我在Java应用程序开发领域的发展非常重要。

# 三、扩展实验——HQL 语言

## 1. 运行结果截图

3、运行测试类 HibernateTest，观察控制台的输出，并记录运行结果

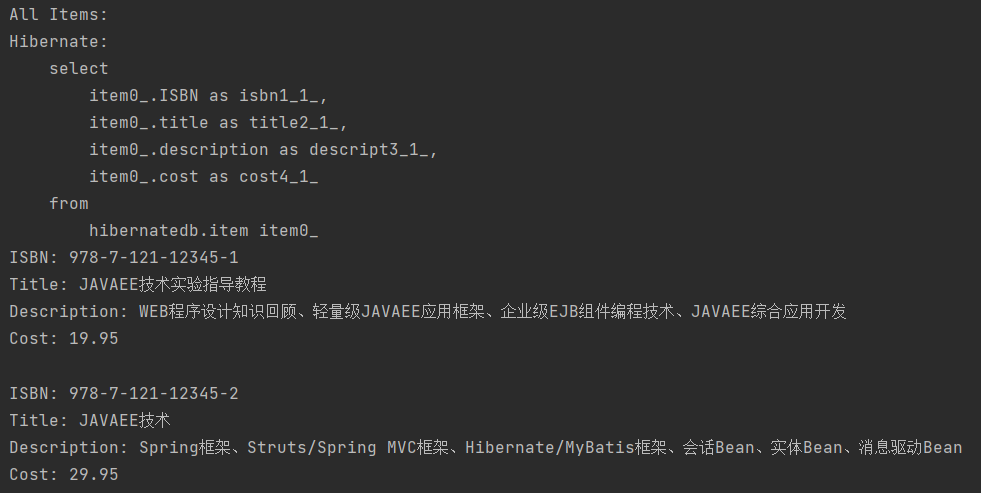


图3.1 运行结果

5、运行测试类 HibernateTest，观察控制台的输出，并记录运行结果



图3.2 运行结果

7、运行测试类 HibernateTest，观察控制台的输出，并记录运行结果

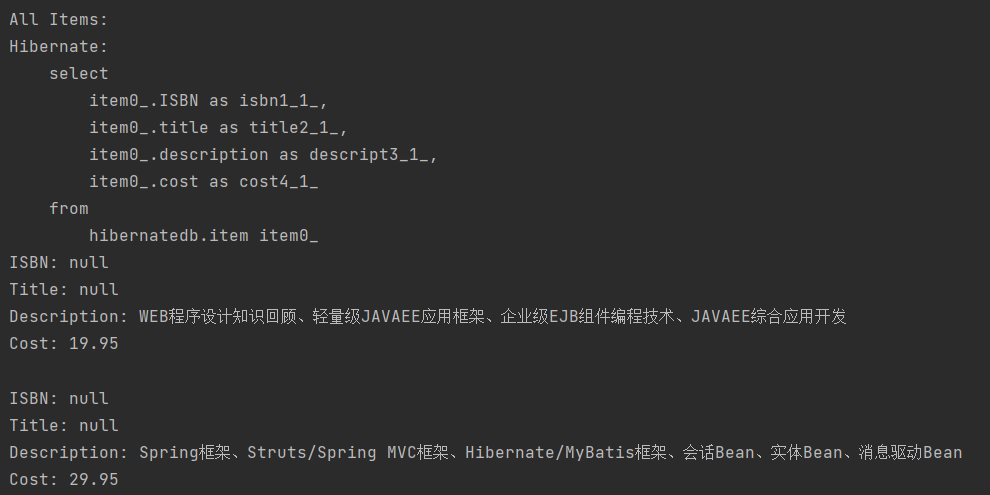


图3.3 运行结果

9、由于 select 子句只返回 title 属性，则返回结果将直接封装到该元素类型的集 合中，即封装到 List集合中，根据该变化修改并运行测试类 HibernateTest，观察控制台的输出，并记录运行结果

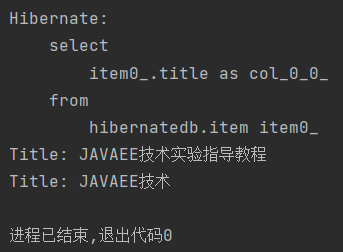


图3.4 运行结果

11、若 select 子句返回多个属性，则返回结果将封装到 List<Object[]>集合中，根据该变化修改并运行测试类 HibernateTest，观察控制台的输出，并记录运行

结果

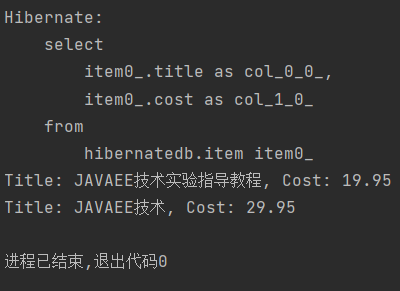


图3.5 运行结果

## 2.总结 HQL 的常用语句及语法规则

Ⅰ.SELECT 语句

使用 SELECT 关键字来选择要从数据库中检索的数据。

可以选择一个或多个实体属性。

使用 FROM 关键字指定要查询的实体类。

示例：SELECT item.title FROM Item as item

Ⅱ.FROM 语句

使用 FROM 关键字指定要查询的实体类。

可以使用别名来引用实体类。

示例：FROM Item as item

Ⅲ.WHERE 语句

使用 WHERE 关键字来指定查询的条件。

可以使用逻辑运算符（如 AND、OR、NOT）来组合条件。

示例：FROM Item as item WHERE item.cost > 100

Ⅳ.JOIN 语句

使用 JOIN 关键字来连接多个实体类。

可以指定连接的类型，如 INNER JOIN、LEFT JOIN、RIGHT JOIN。

示例：FROM Order as order INNER JOIN order.items as item

Ⅴ.ORDER BY 语句

使用 ORDER BY 关键字来指定查询结果的排序顺序。

可以按一个或多个属性进行升序或降序排序。

示例：FROM Item as item ORDER BY item.cost DESC

Ⅵ.GROUP BY 语句

使用 GROUP BY 关键字对查询结果进行分组。

通常与聚合函数一起使用，如 SUM、COUNT。

示例：SELECT item.category, COUNT(item) FROM Item as item GROUP BY item.category

Ⅶ.HAVING 语句

使用 HAVING 关键字来指定对分组后的结果进行过滤的条件。

类似于 WHERE，但用于分组后的结果。

示例：SELECT item.category, COUNT(item) FROM Item as item GROUP BY item.category HAVING COUNT(item) > 5

Ⅷ.参数绑定

可以使用参数占位符 ? 或命名参数 :paramName 来传递参数。

使用 setParameter 方法来绑定参数。

示例：FROM Item as item WHERE item.cost > ?

Ⅸ.子查询

可以在 SELECT、FROM 或 WHERE 子句中包含子查询。

子查询返回一个值或列表，可以用于主查询的条件。

示例：FROM Item as item WHERE item.cost > (SELECT AVG(cost) FROM Item)

Ⅹ.连接多个实体

可以通过实体之间的关联属性进行多表连接。

使用点号 . 来访问关联属性。

示例：FROM Order as order INNER JOIN order.items as item WHERE item.category = 'Electronics'

## 3. 碰到的问题及解决方案或思考

没有碰到问题。

## 4. 实验收获及总结

通过本次实验，我学会了HQL的基本语法，包括SELECT、FROM、WHERE、ORDER BY、GROUP BY、JOIN等子句的使用，帮助我更深入地了解了Hibernate中的HQL查询语言。