



JavaEE 实验六

实验报告

姓　　名　　 Mcrivers

班　　级　　 软件工程2102班

学　　号

提交日期　　 2023.12.24

目录

[一、基础实验——MyBatis 框架搭建 1](#_Toc154341843)

[1. 运行结果截图 1](#_Toc154341844)

[2. 简述 MyBatis 的工作原理 1](#_Toc154341845)

[3. 简述 MyBatis 和 Hibernate 的异同点和优缺点 2](#_Toc154341846)

[4. 碰到的问题及解决方案或思考 3](#_Toc154341847)

[5. 实验收获及总结 3](#_Toc154341848)

[二、提高实验——映射器 4](#_Toc154341849)

[1. 运行结果截图 4](#_Toc154341850)

[2. 结合实验过程，总结 MyBatis 实现查询时返回的结果集中常见的存储方式 7](#_Toc154341851)

[3. 碰到的问题及解决方案或思考 7](#_Toc154341852)

[4. 实验收获及总结 8](#_Toc154341853)

[三、扩展实验——级联查询 9](#_Toc154341854)

[1. 运行结果截图 9](#_Toc154341855)

[2.碰到的问题及解决方案 10](#_Toc154341856)

[3.实验收获及总结 10](#_Toc154341857)

# 一、基础实验——MyBatis 框架搭建

## 1. 运行结果截图

**11、运行以上测试类，并记录运行结果**

观察到，查询操作、插入操作、更新操作、删除操作都可以正常进行

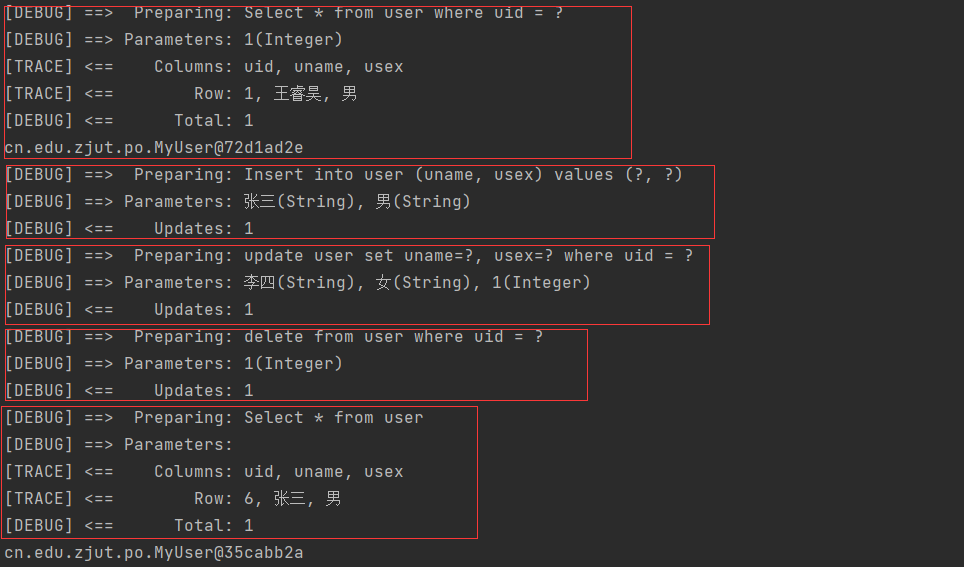


图1.1 项目成功运行的日志截图

## 2. 简述 MyBatis 的工作原理

①**配置文件**：MyBatis配置文件指定了 MyBatis 如何连接到数据库、如何映射 Java 对象和数据库表，以及其他运行时的行为。

②**SQL 映射文件**：SQL 映射文件包含了 SQL 查询和操作的定义，以及如何将查询结果映射到 Java 对象的规则，同时定义了各种 SQL 语句，包括查询、插入、更新和删除操作。

③**SqlSessionFactory**：SqlSessionFactory 是 MyBatis 的核心接口之一。它负责创建 SqlSession 对象，SqlSession 是与数据库的一次会话。SqlSessionFactory 使用配置文件和 SQL 映射文件来构建应用程序的数据库连接。

④**SqlSession**：SqlSession 是 MyBatis 中用于执行 SQL 语句的主要接口。通过 SqlSession，程序可以执行 SQL 查询、插入、更新和删除操作。

⑤**Mapper 接口**：Mapper 接口是定义了 SQL 映射文件中 SQL 操作的方法接口，对应了 SQL 映射文件中定义的 SQL 语句。

⑥**MyBatis 缓存**：MyBatis 提供了一级缓存和二级缓存，用于提高查询性能。

⑦**执行 SQL**：在应用程序中，开发人员通过 SqlSession 或 Mapper 接口来执行 SQL 操作。MyBatis 会将 SQL 语句发送到数据库，并将查询结果映射为 Java 对象，然后返回给应用程序。

## 3. 简述 MyBatis 和 Hibernate 的异同点和优缺点

**MyBatis：**

**优点：**

1. **灵活性高：** MyBatis 提供了更灵活的 SQL 控制，开发者可以完全控制 SQL 语句的编写和执行过程。这对于需要特定优化或复杂查询的应用程序来说非常有用。
2. **直观性好：** MyBatis 的 SQL 映射文件以及配置文件使用直观的 XML 或注解，易于理解和维护。开发者可以更容易地编写自定义 SQL，而无需深入学习框架。
3. **性能可控：** MyBatis 允许开发者手动优化 SQL 查询，包括缓存查询结果、延迟加载等，因此可以更好地控制应用程序的性能。
4. **轻量级：** MyBatis 的核心库相对较小，不依赖于大量的第三方库，因此部署和维护相对简单。

**缺点：**

1. **需要手动编写 SQL：** 虽然灵活性高，但需要开发者手动编写大部分 SQL 查询语句，这可能增加了开发的工作量。
2. **复杂性增加：** 随着项目的复杂性增加，可能需要更多的配置和映射文件，导致项目变得复杂。

**Hibernate：**

**优点：**

1. **无需编写 SQL：** Hibernate 自动生成 SQL 查询语句，开发者不需要手动编写大量的 SQL。这降低了开发工作量，特别适用于简单的 CRUD 操作。
2. **面向对象：** Hibernate 使用对象-关系映射（ORM）的方式，将数据库表映射为对象，开发者可以使用面向对象的方式操作数据，减少了关注数据库细节的需求。
3. **自动维护：** Hibernate 负责数据库的 CRUD 操作、事务管理和缓存管理，减轻了开发者的工作负担。
4. **生态系统丰富：** Hibernate 拥有庞大的社区和丰富的生态系统，提供了许多扩展和插件，可以快速构建功能丰富的应用程序。

**缺点：**

1. **复杂性高：** Hibernate 的底层复杂性较高，可能需要开发者学习更多的概念和配置，特别是在处理复杂查询时。
2. **性能难以控制：** Hibernate 自动生成的 SQL 查询可能不总是最优化的，开发者可能需要进行额外的优化工作。

## 4. 碰到的问题及解决方案或思考

碰到的问题：程序可以正常运行，但是无法在控制台中正常输出日志信息，如图所示

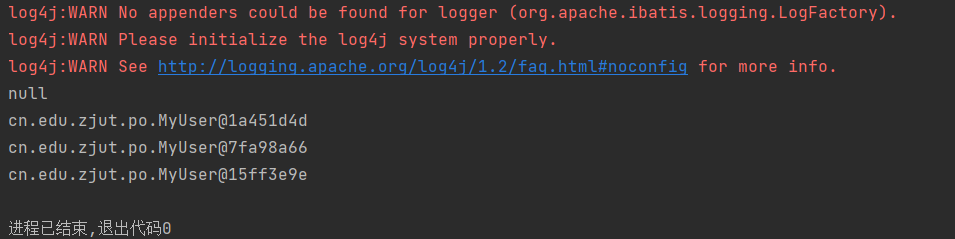


图1.2 程序运行时无法输出日志信息

解决方案：在resources目录下创建log4j的配置文件log4j.properties，如下图

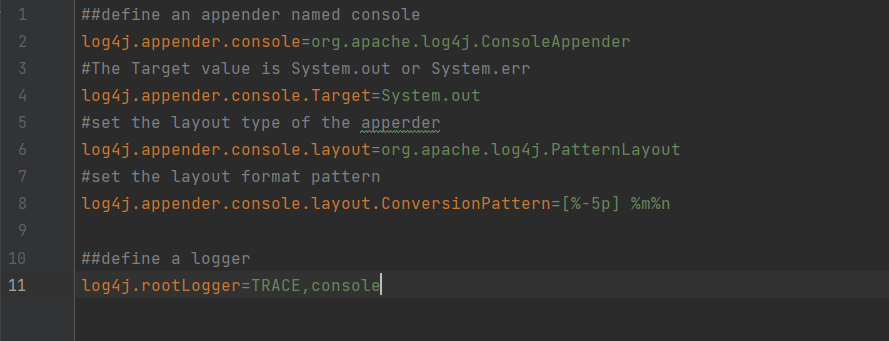


图1.2 log4j.properties配置

配置完成后，程序可以正常运行。

## 5. 实验收获及总结

这个实验使我了解了 MyBatis 的基本工作原理和使用方法。我学会了如何配置映射文件、核心配置文件以及如何执行数据库操作。此外，我还熟悉了 MySQL 数据库的基本操作。这个实验为我进一步深入学习 MyBatis 打下了基础。

# 二、提高实验——映射器

## 1. 运行结果截图

**2、修改测试代码，将查询所有用户信息的代码片段修改为“查询陈姓男性用户 信息”，并记录运行结果**

观察到，程序成功运行，查询结果为姓为“陈”的男性，总数为1人。

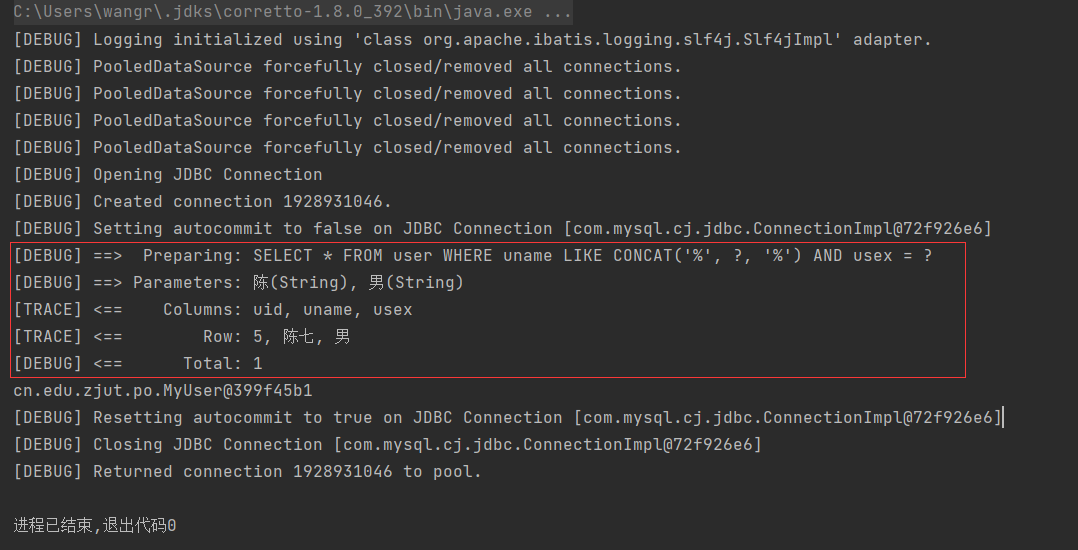


图2.1 程序正常运行

**4、修改测试代码，使用映射器 UserDao 实现“查询陈姓男性用户信息”，并记 录运行结果**

观察到，程序成功运行，查询结果为姓为“陈”的男性，总数为1人。

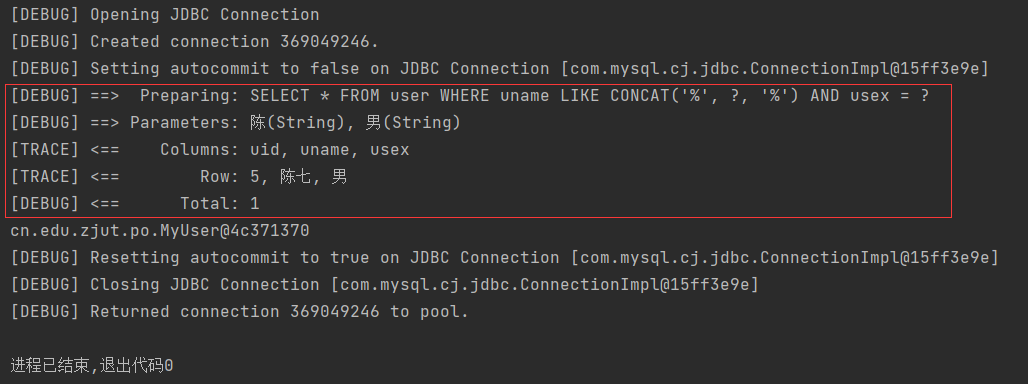


图2.2 程序正常运行

**7、修改测试代码中“查询陈姓男性用户信息”的部分代码，并记录运行结果**

观察到，程序成功运行，查询结果为姓为“陈”的男性，总数为1人。

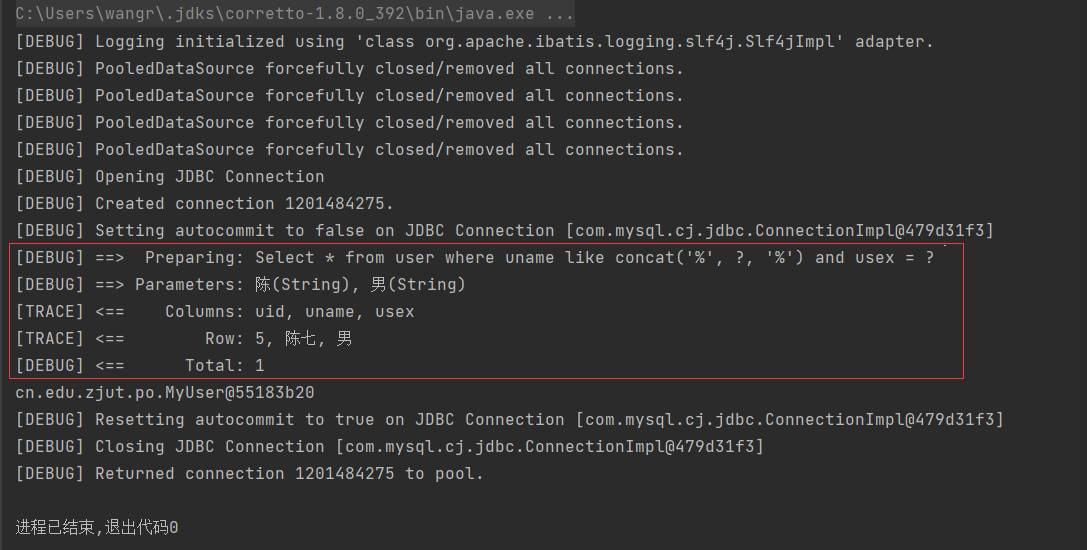


图2.3 程序正常运行

**9、修改测试代码，将查询所有用户信息的返回结果保存到 Map 中**

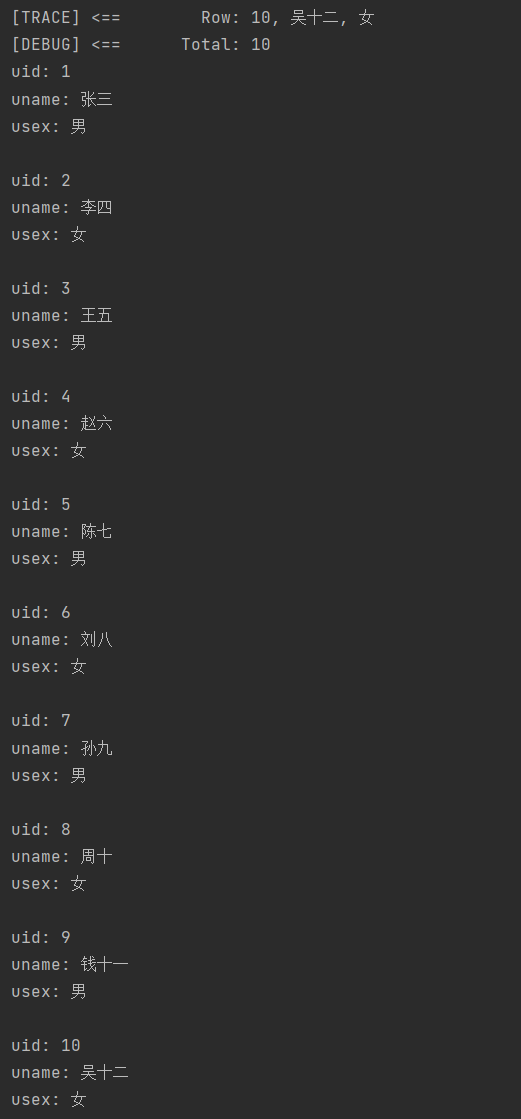


图2.4 程序正常运行

**13、修改测试代码，使用 id 为 selectResultMap 的 select 操作完成查询，并记录 运行结果**

观察到，程序成功运行，完成了查询操作

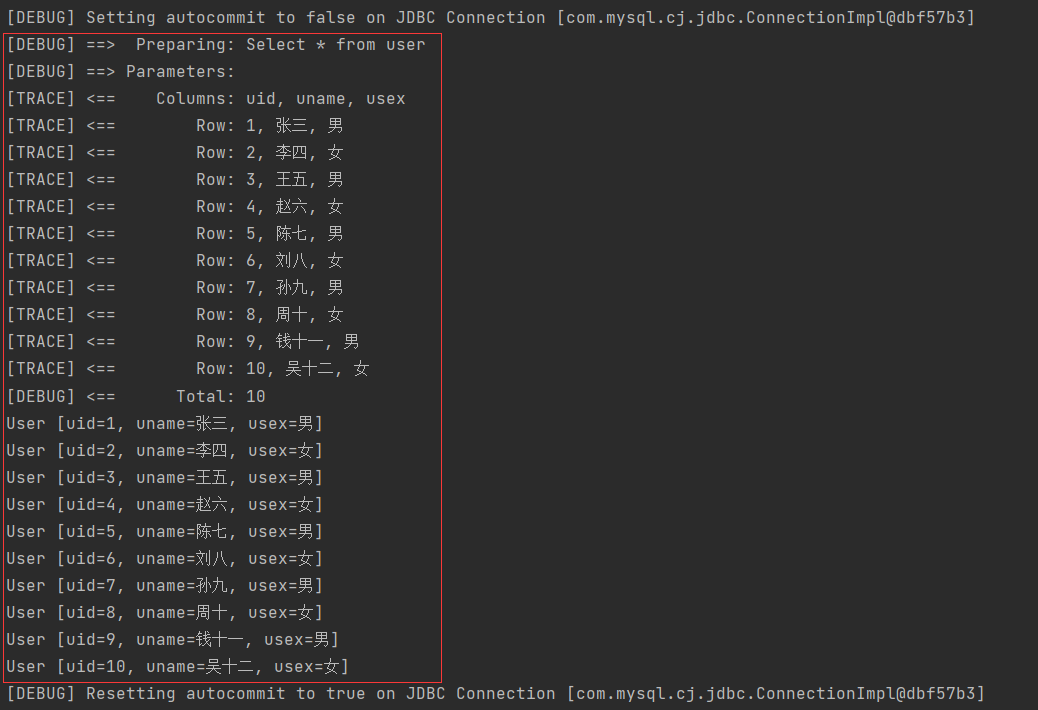


图2.5 程序正常运行

**14、尝试使用 resultMap 属性和 parameterMap 属性，完成增删改操作，并记录运行结果。**

观察到，程序成功进行了增删改操作，并输出了影响行数。

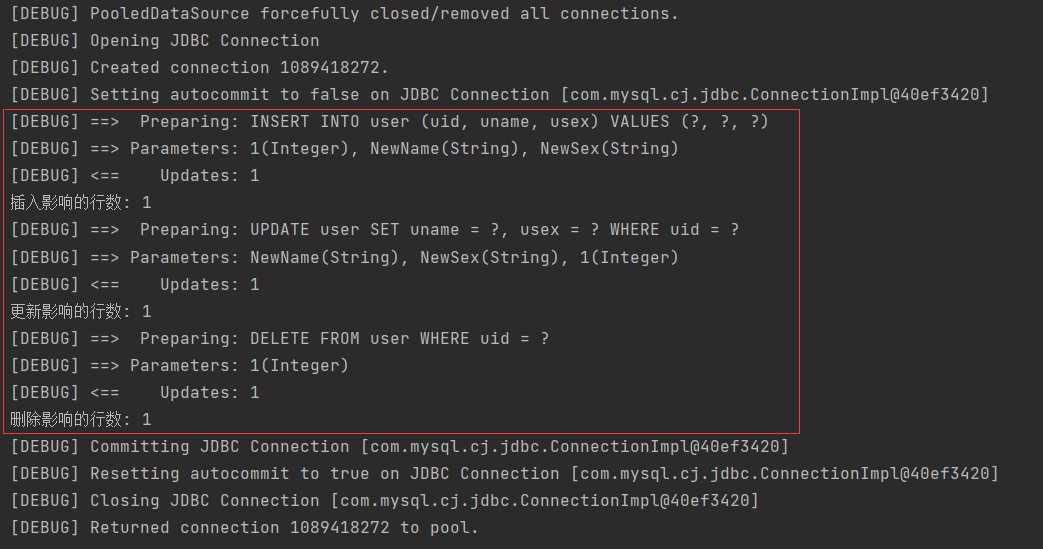


图2.6 程序正常运行

## 2. 结合实验过程，总结 MyBatis 实现查询时返回的结果集中常见的存储方式

①**返回单个对象**：将查询结果映射为一个Java对象，通常用于查询单条记录。

②**返回List集合**：将查询结果映射为一个Java对象的List集合，通常用于查询多条记录。

③**返回Map对象**：将查询结果映射为一个Map对象，其中键为列名，值为列值。

④**返回自定义POJO对象**：将查询结果映射为自定义的Java对象，这些对象通常与数据库表的结构不完全相同，需要使用resultType或resultMap配置来指定映射关系。

⑤**返回嵌套对象**：将查询结果映射为具有复杂结构的嵌套Java对象。这需要使用resultMap配置来定义嵌套映射关系。

⑥**返回联合查询结果**：将多个表的查询结果映射为一个复杂的Java对象，通常用于联合查询的情况。这需要使用resultMap配置来定义联合映射关系。

⑦**返回存储过程的输出参数**：如果您调用了存储过程，MyBatis可以将存储过程的输出参数映射到Java对象中，以便访问过程的结果。

⑧**返回游标**：当需要处理大量数据集时，MyBatis可以将查询结果映射为游标，以便逐行处理数据而不会加载整个结果集到内存中。

## 3. 碰到的问题及解决方案或思考

遇到的问题：使用映射器UserDao进行查询操作时，程序出现报错，如图所示

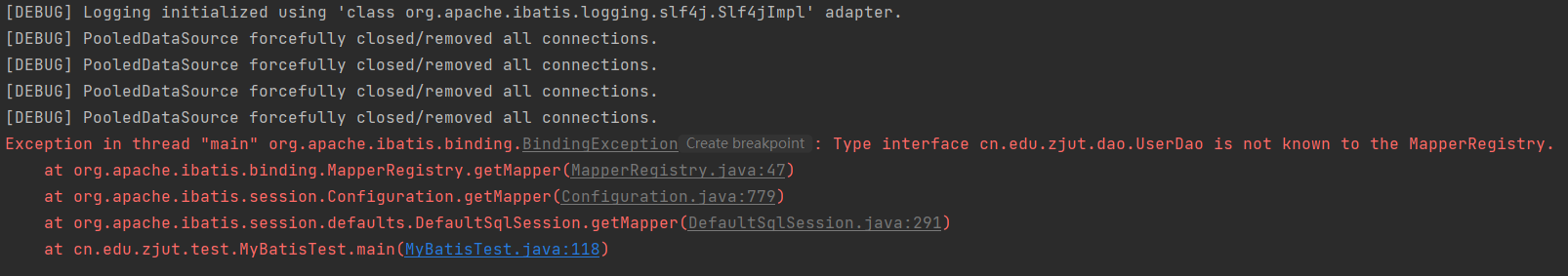


图2.7 程序报错

解决方法：在mybatis-config.xml中注册UserDao.xml，如图所示

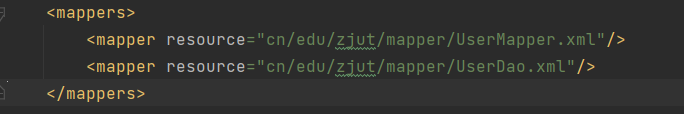


图2.8注册UserDao.xml

随后程序可以正常运行。

## 4. 实验收获及总结

这次实验中，我进一步了解了MyBatis的映射器的使用方法和映射文件的配置。通过实验，我学会了如何创建一个映射器接口，定义SQL语句的映射方法，并在映射文件中配置这些SQL语句的具体内容。

我也学到了如何在SQL映射文件中使用常见的配置元素，如<select>、<insert>、<update>和<delete>，以及它们的属性，如id、parameterType、resultType、resultMap等。这些元素和属性对于配置和执行SQL语句非常重要。

此外，我了解了不同的结果集存储方式，包括将结果映射为Java对象、List集合、Map对象以及自定义的POJO对象。这些存储方式可以根据查询需求的不同来选择，并能够有效地处理查询结果。

总的来说，通过这次实验，我更熟悉了MyBatis的映射器和映射文件的使用，掌握了在实际开发中如何配置和执行SQL语句，以及如何存储和处理查询结果。这些知识将在今后的项目中对我有很大帮助。

# 三、扩展实验——级联查询

## 1. 运行结果截图

**9、创建测试类 TestOneToOne 并记录运行结果**

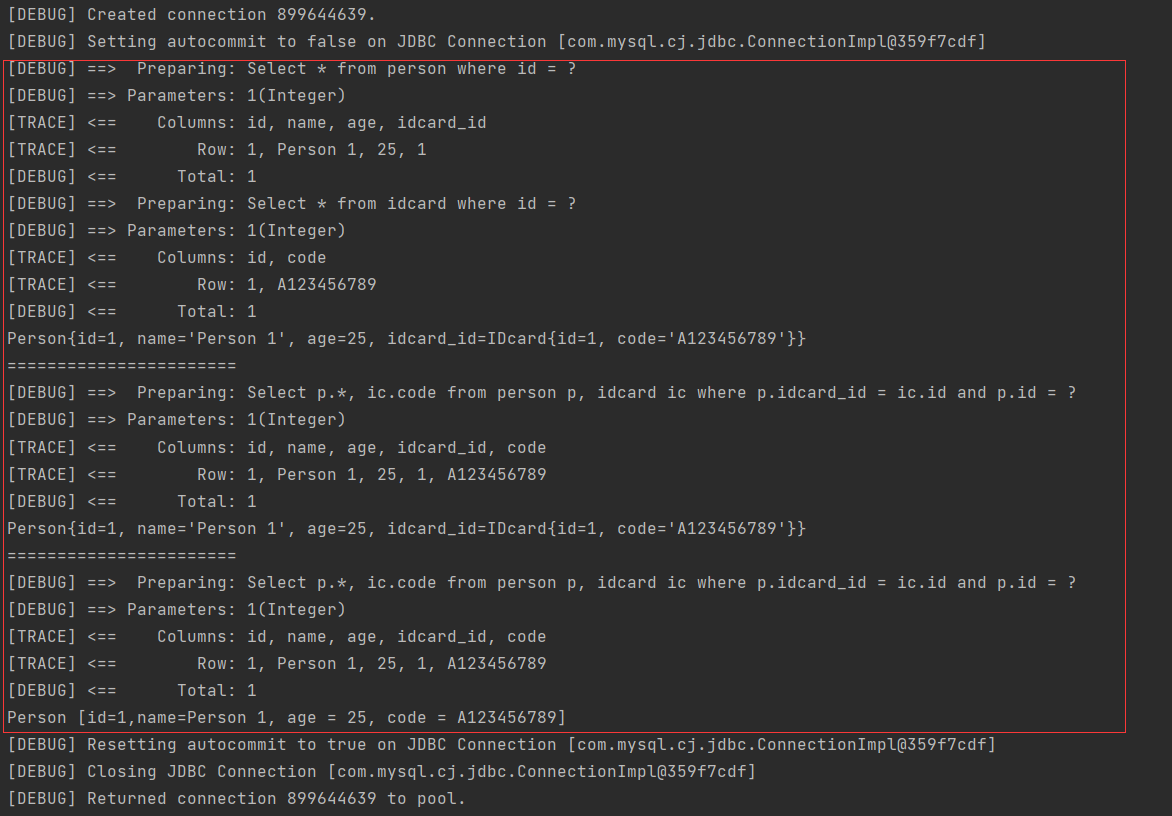


图3.1程序成功运行

**10、尝试以用户和订单之间的一对多关系为例，实现一对多级联查询的处理过程， 并记录过程**

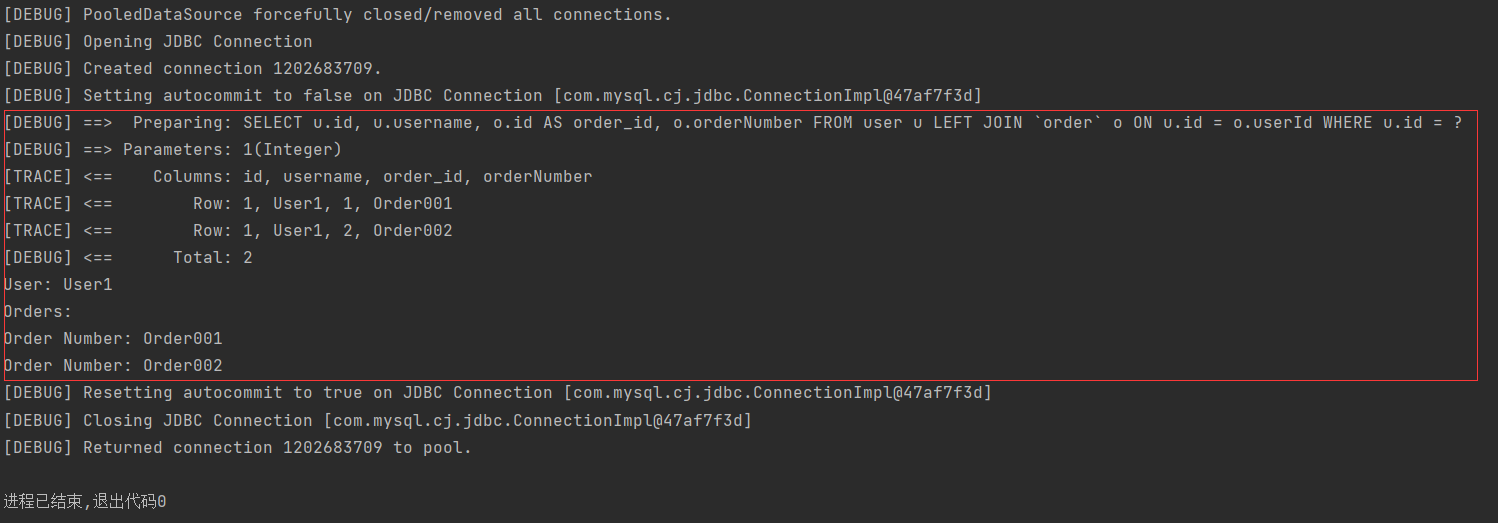


图3.2程序成功运行

## 2.碰到的问题及解决方案

碰到的问题：程序运行时出现如下报错

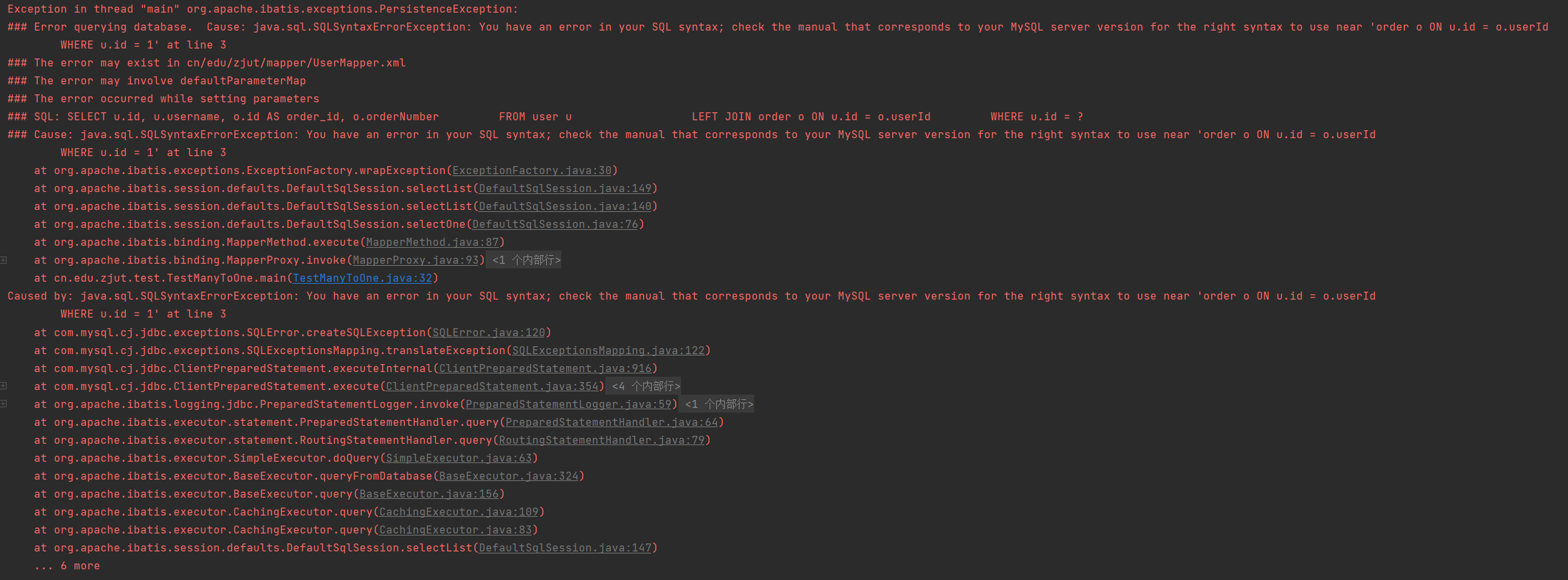


图3.3程序运行报错

解决方案：因为order是数据库中的关键字，所以需要将order用反引号包围起来，如下图

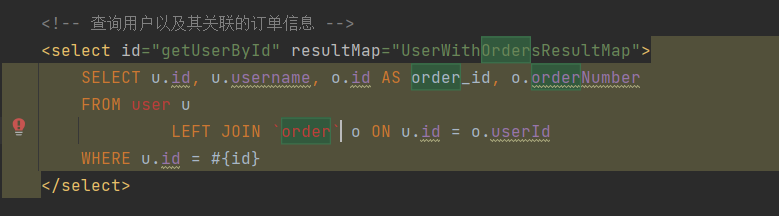


图3.4将order用反引号包围

修改代码后程序成功运行

## 3.实验收获及总结

这个实验为我提供了宝贵的经验，使我更加熟悉了MyBatis框架的使用，对一对一级联查询的概念和实现方式有了更深入的了解。这些知识和经验对于开发数据库相关的应用程序非常有价值。