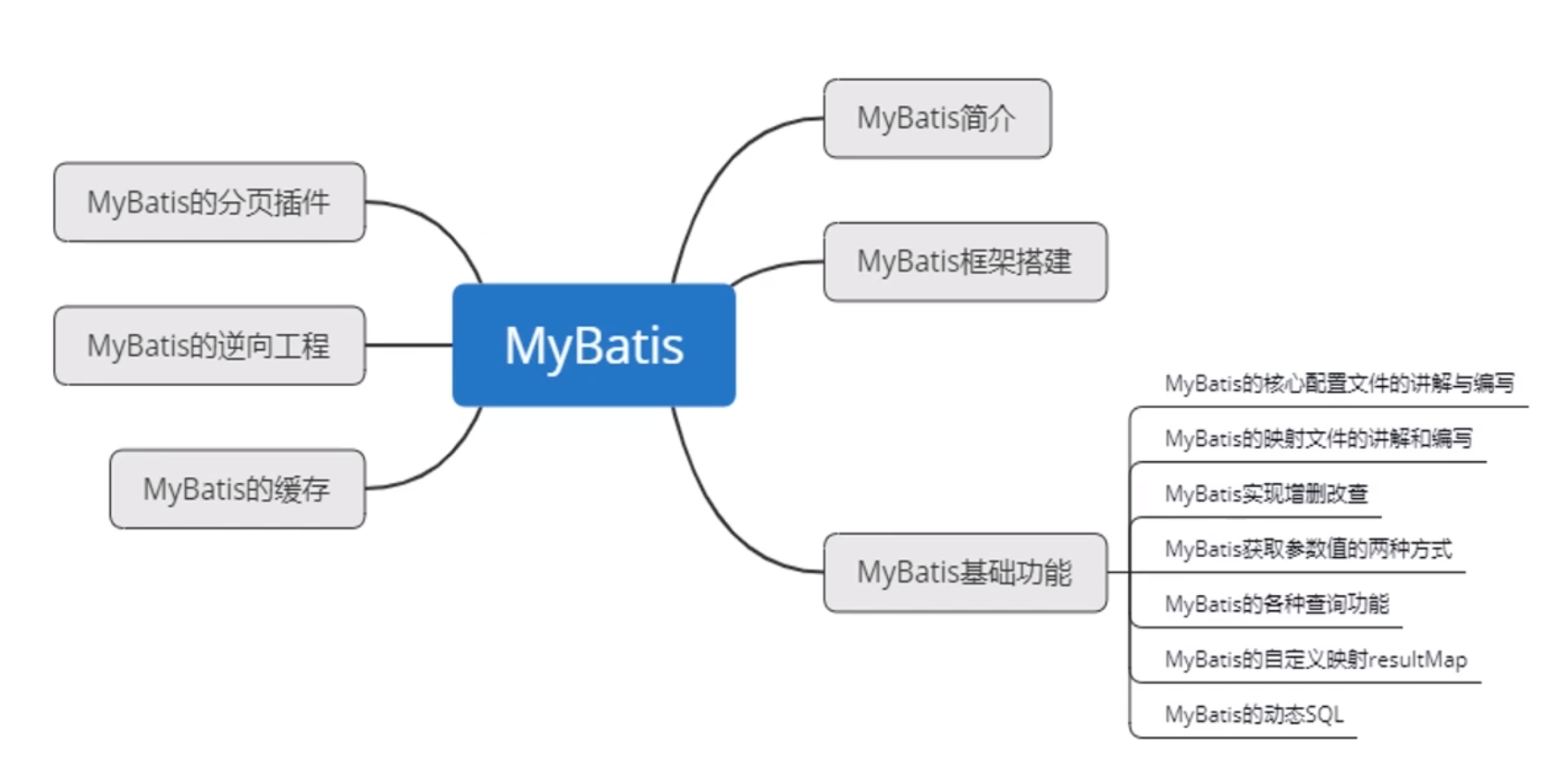
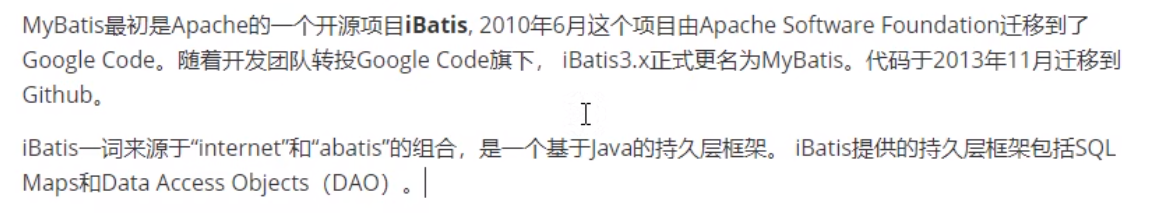
# Mybatis笔记



# Mybatis简介

Mybatis是对jdbc的封装

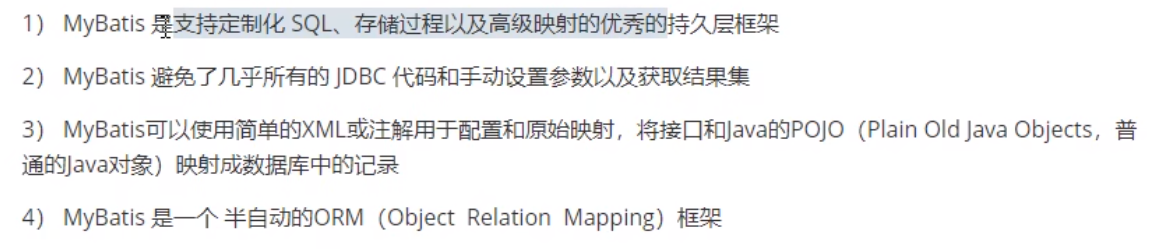
# 1.1、Mybatis历史



SQLMaps:数据库中的数据和java之间的映射关系，换句话说，SQLMaps就是封装jdbc的一个过程。

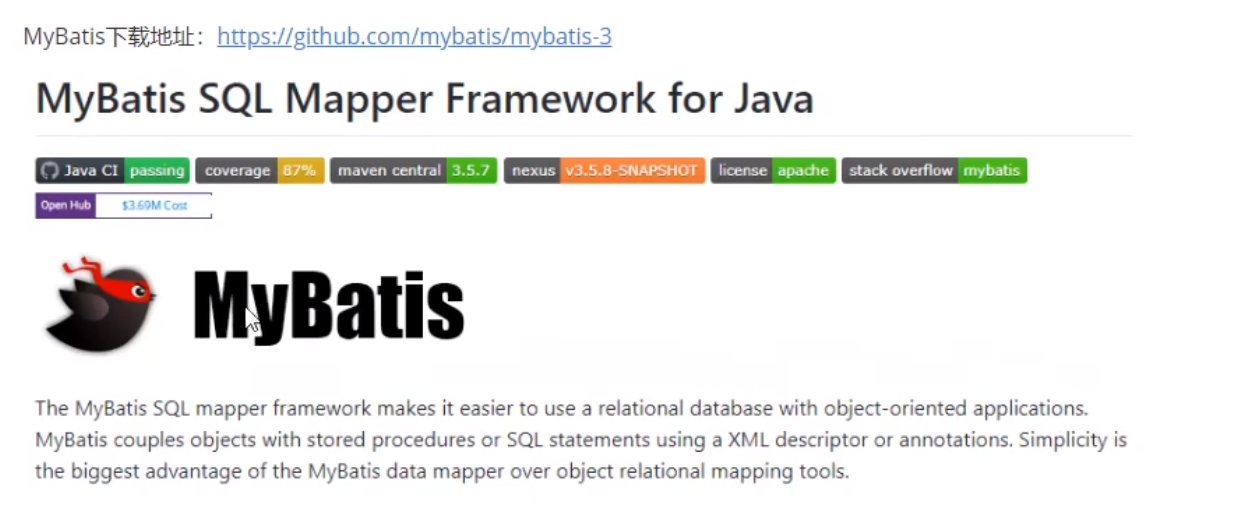
Data Access Objects(DAO):数据访问对象，就是通过jdbc访问数据库中的数据，操作数据库中的数据。

# 1.2、Mybatis特性

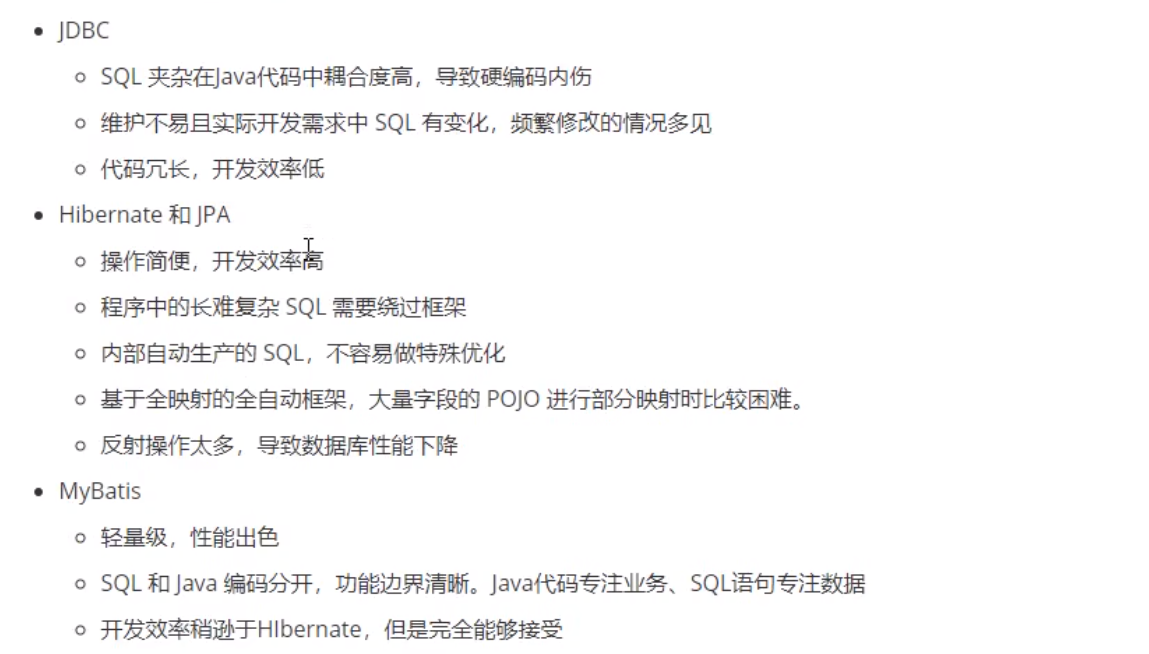


ORM:对象关系映射，O:Object-对象指java中的对象，R:Relation-关系指关系型数据库，M:Mapping:映射。就是将java中的实体类对象和数据库中的数据创建映射关系。

# 1.3、Mybatis下载



# 1.4、和其它持久层技术对比



# 2、搭建MyBatis

# 2.1、开发环境

IDE:idea 2022.2.2

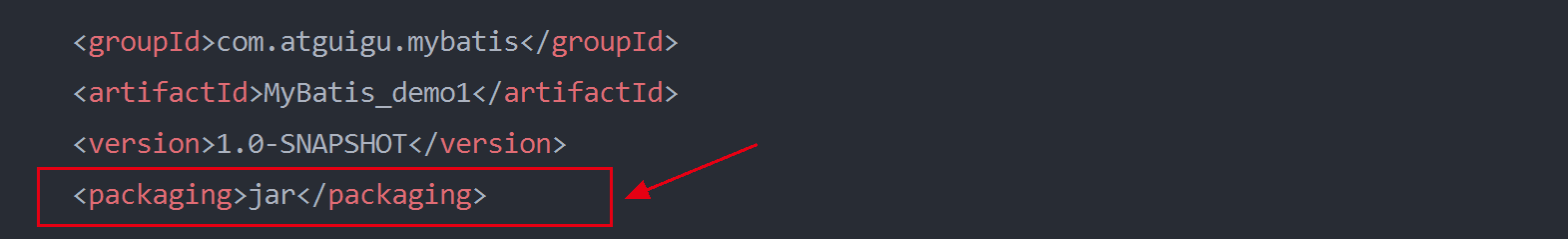
构建工具：maven 3.8.1

MySql版本：MySql 5.5.52

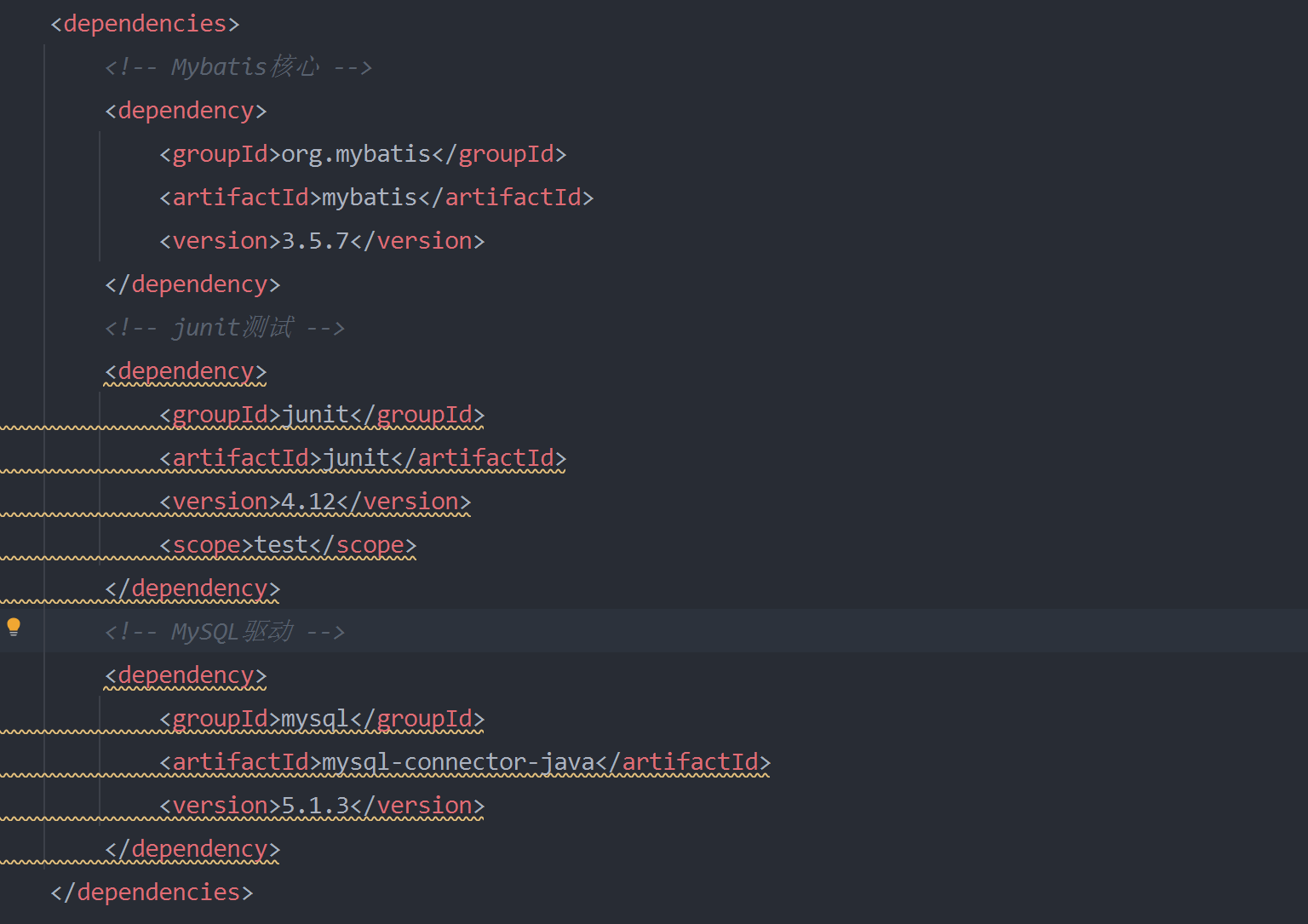
MyBatis版本：MyBatis 3.5.7

# 2.2、创建Maven工程

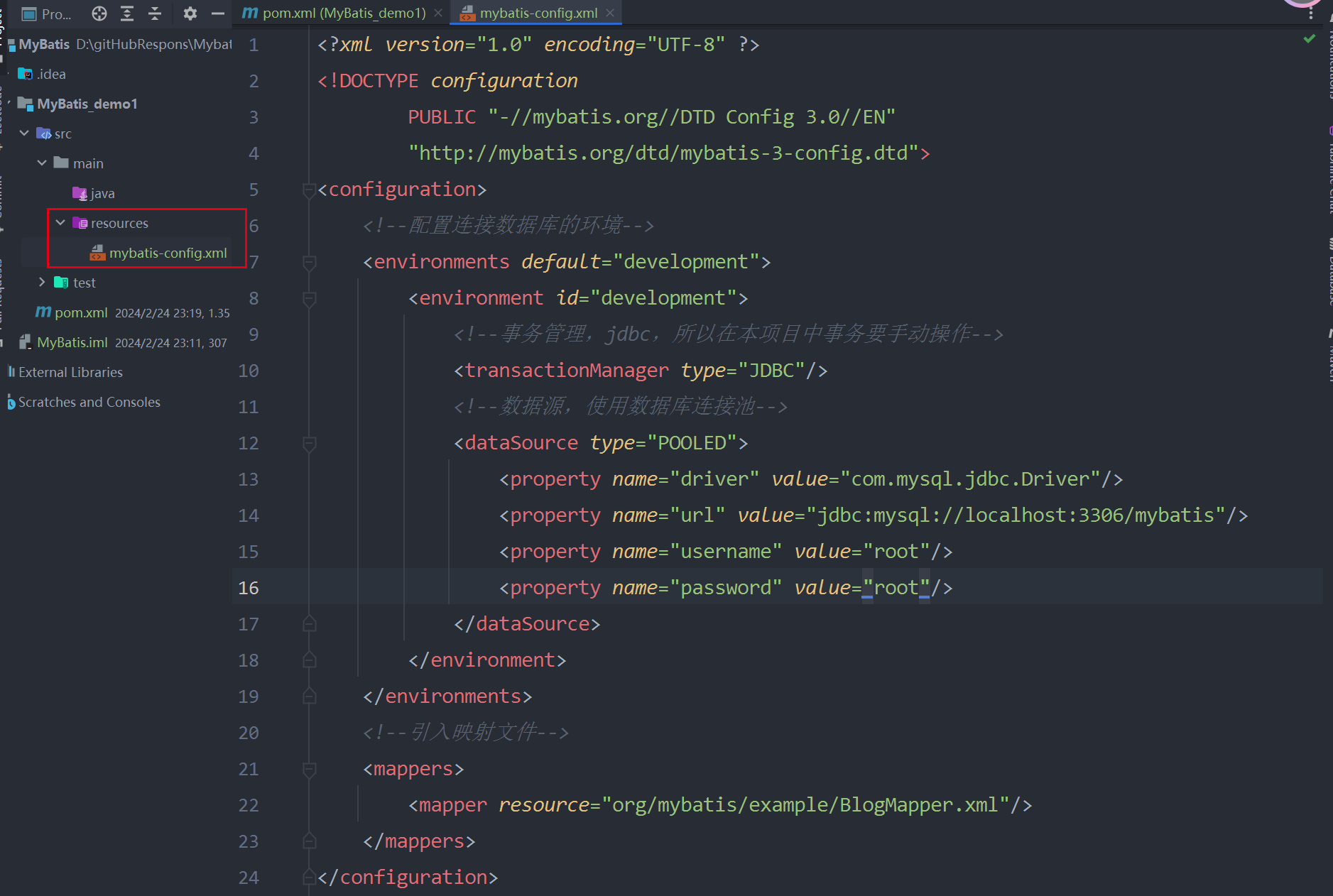
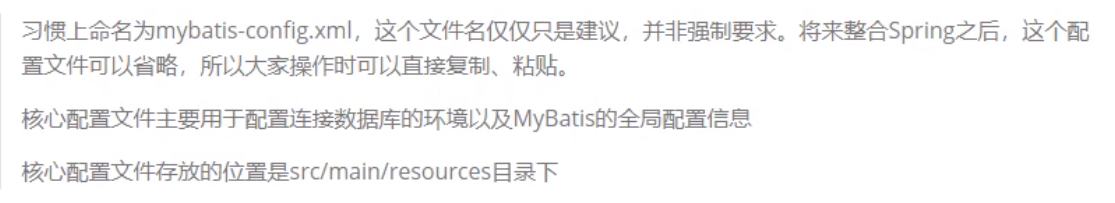
### 2.2.1、打包方式：jar



### 2.2.2、引入依赖



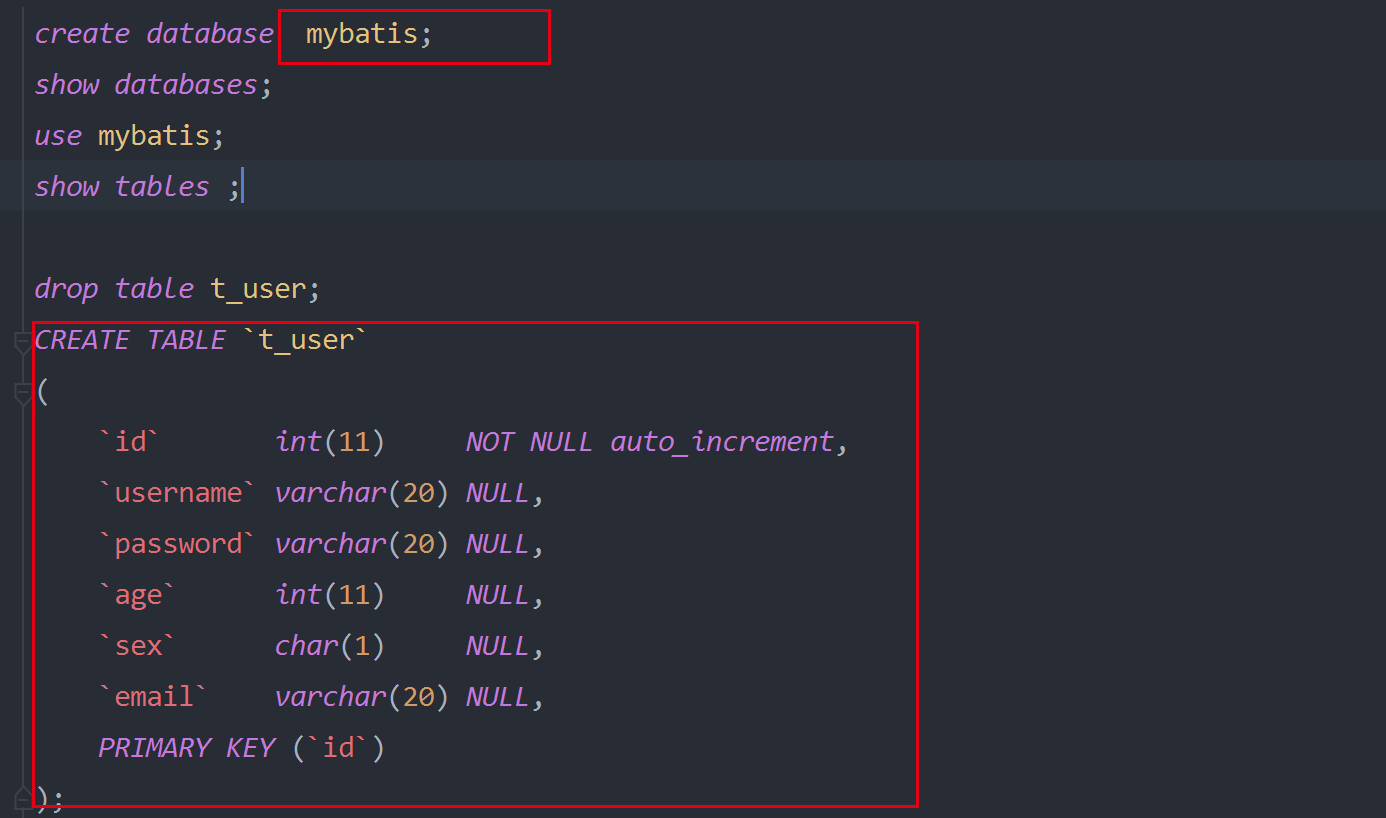
# 2.3、创建MyBatis的核心配置文件



# 2.4、创建mapper接口



在mysql中创建数据库mybatis和数据表t\_user;



# 2.5、创建MyBatis的映射文件

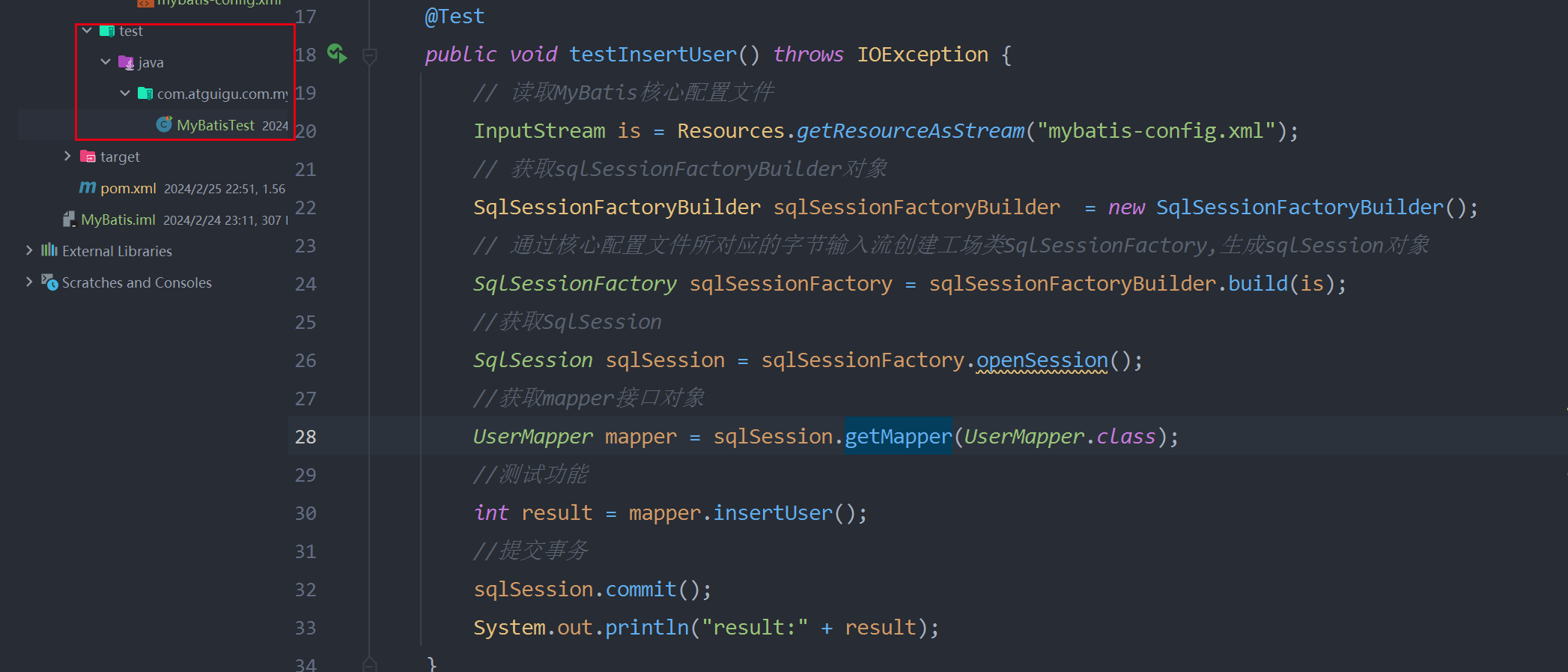


# 2.6、通过junit测试功能

SqlSession：代表Java程序和数据库之间的会话。（HttpSession是Java程序和浏览器之间的会话）

SqlSessionFactory：是生产SqlSession的工厂

工厂模式：如果创建一个对象，使用的过程基本固定，那么我盟就可以把创建这个对象的相关代码封装到一个“工厂类”中，以后都使用这个工厂类来生产我们需要的对象

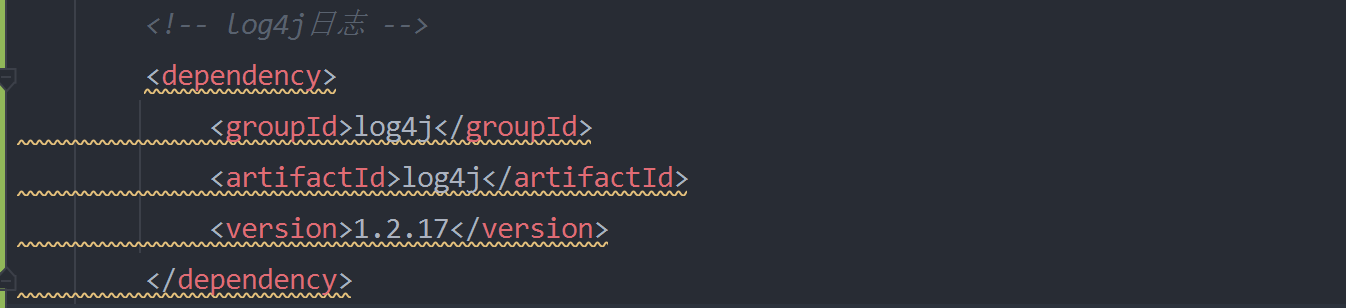


此时需要手动提交事务，如果要自动提交事务，则在获取sqlSession对象时，使用SqlSession sqlSession = sqlSessionFactory.openSession(true);，传入一个Boolean类型的参数，值为true，这样就可以自动提交。

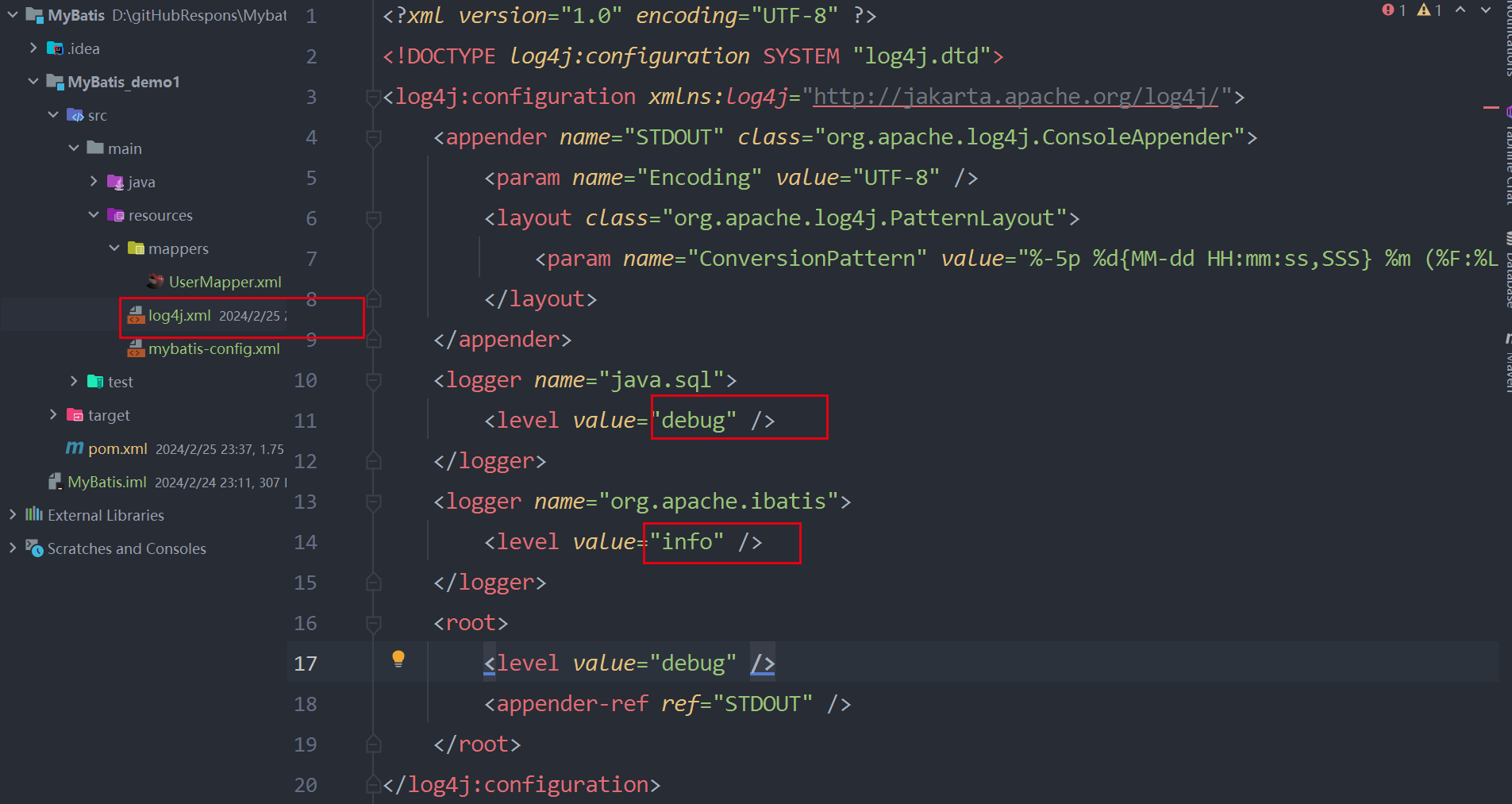


# 2.7、加入log4j日志功能

### 2.7.1、加入依赖



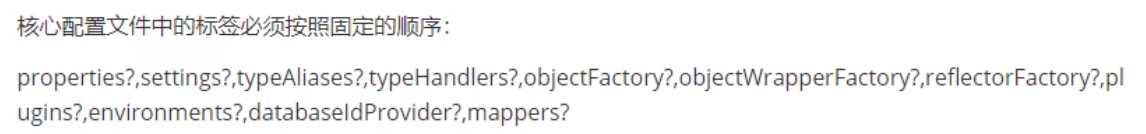
### 2.7.2、加入log4j的配置文件

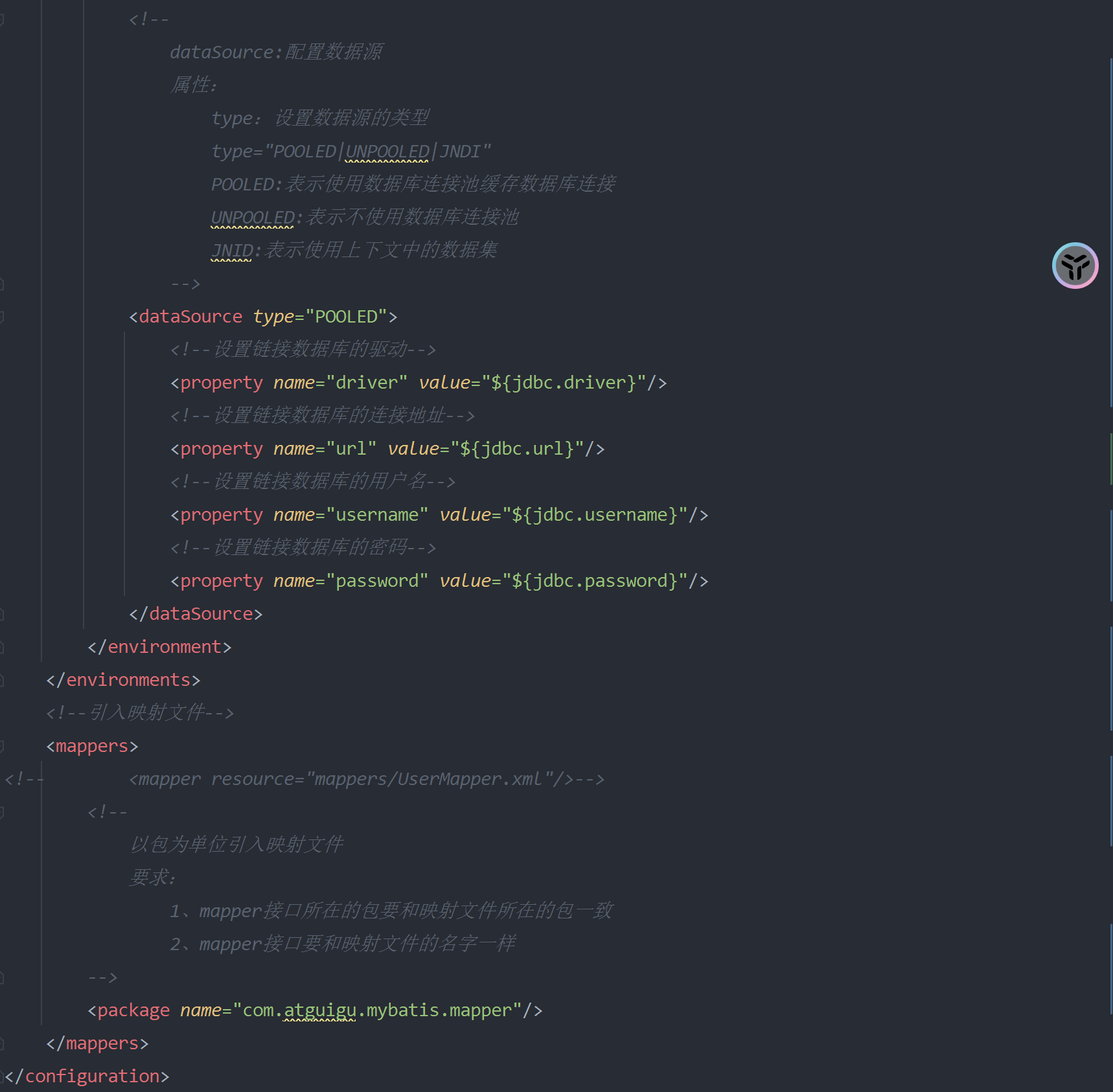
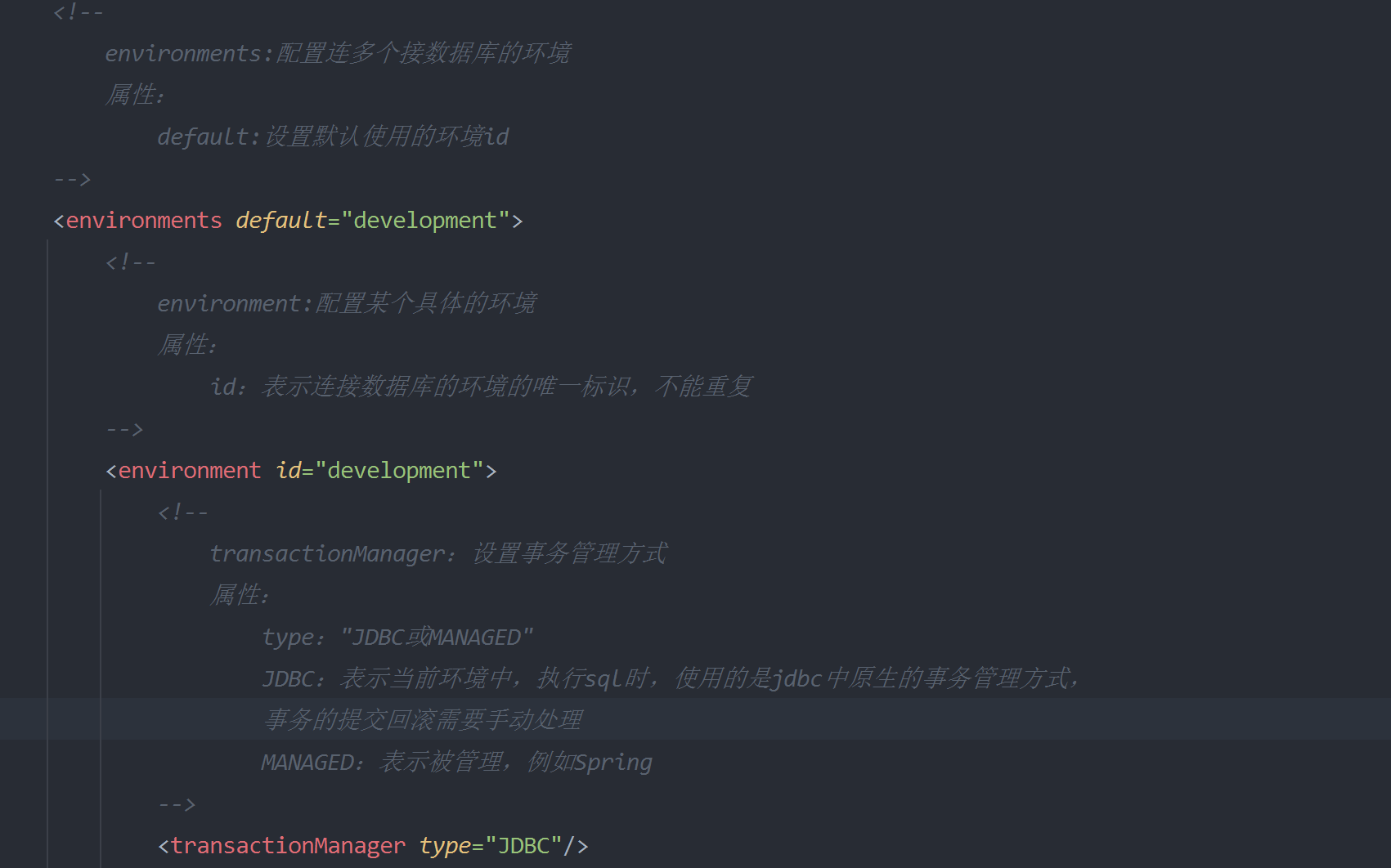


1、log4j的配置文件名为log4j.xml，存放的位置是src/main/resources目录下。

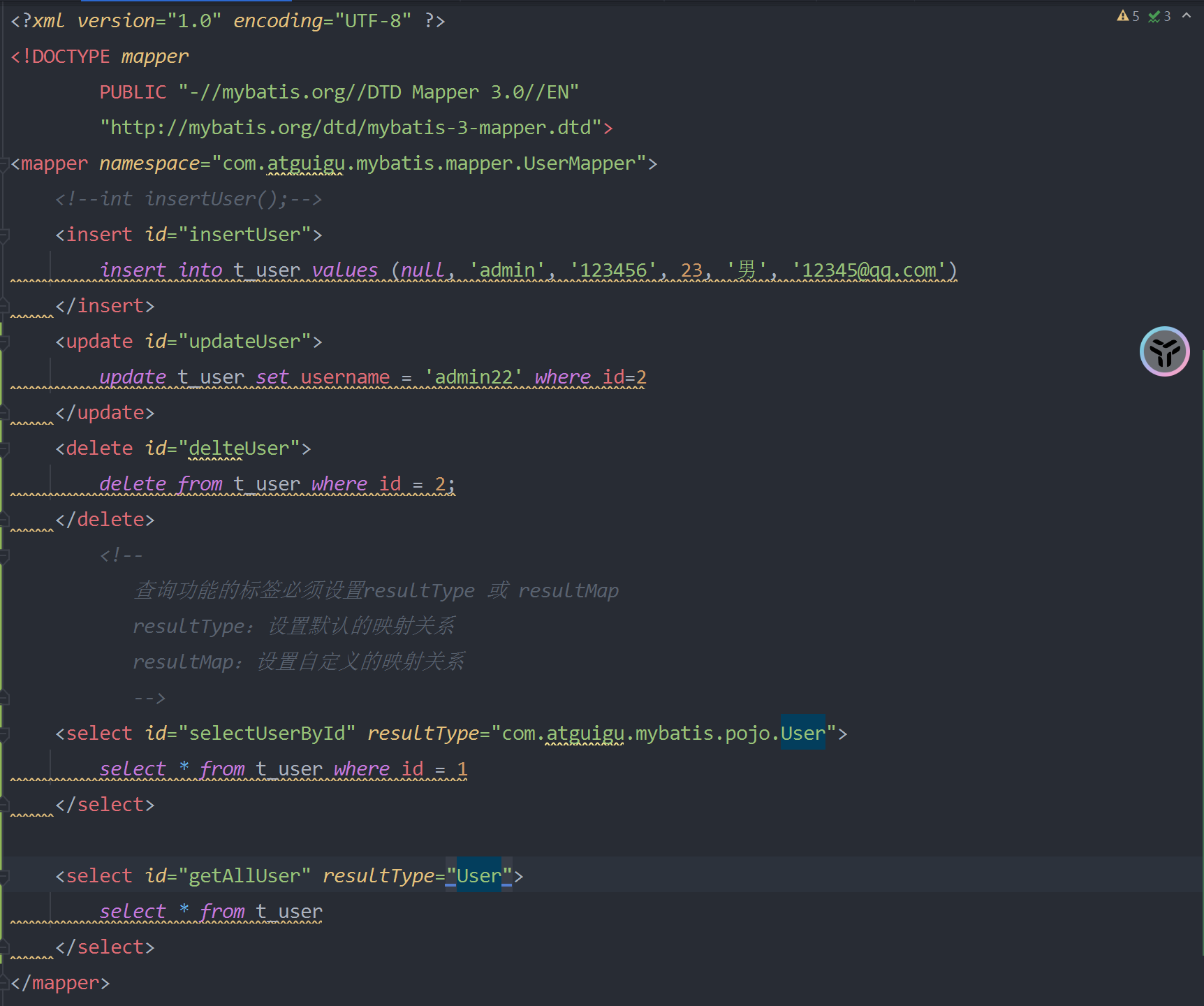
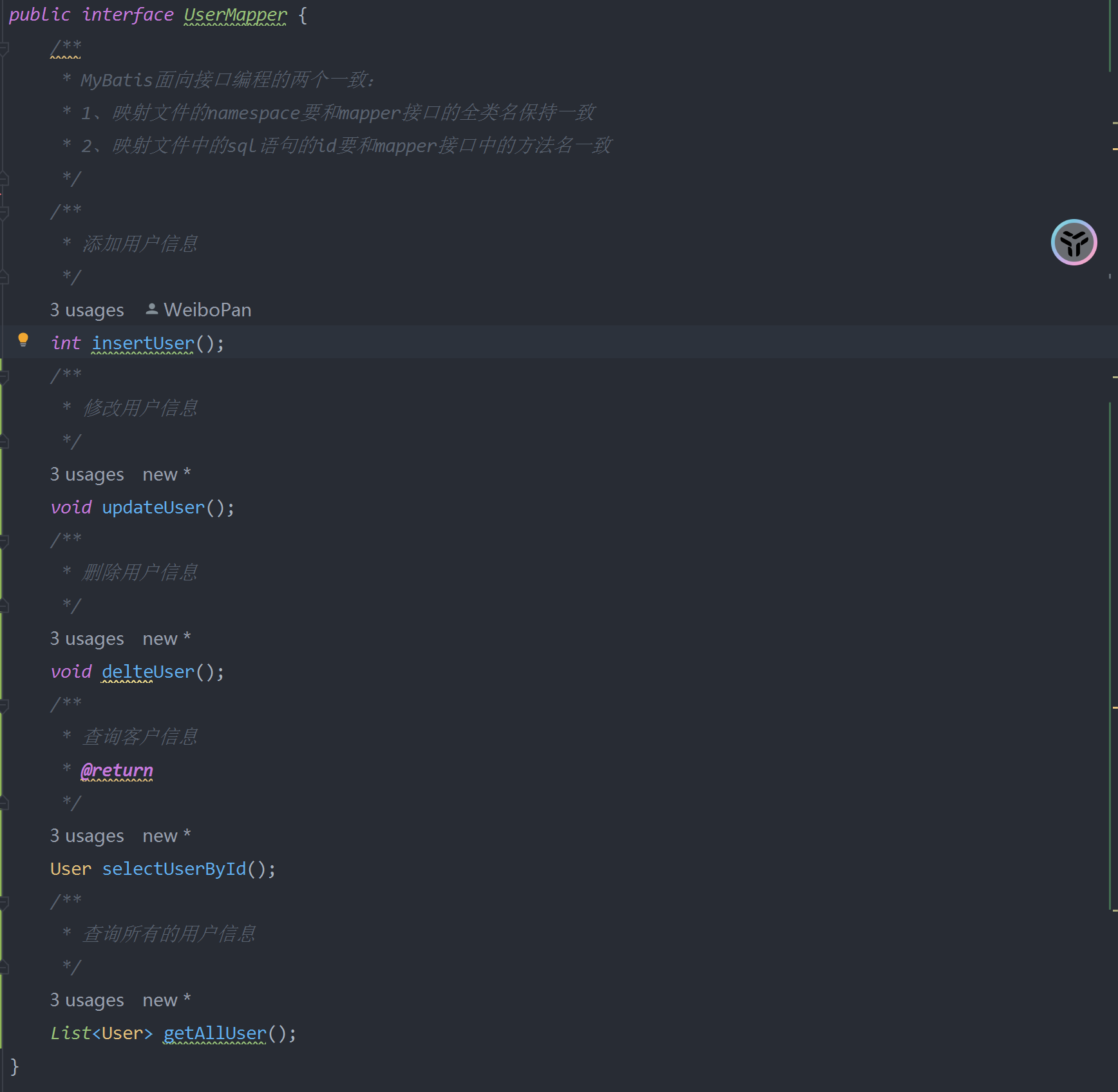
2、日志的级别：FATAL(致命)>ERROR(错误)>WARN(警告)>INFO(信息)>DEBUG(调试) 从左到右打印的内容越来越详细。

# 3、核心配置文件详解

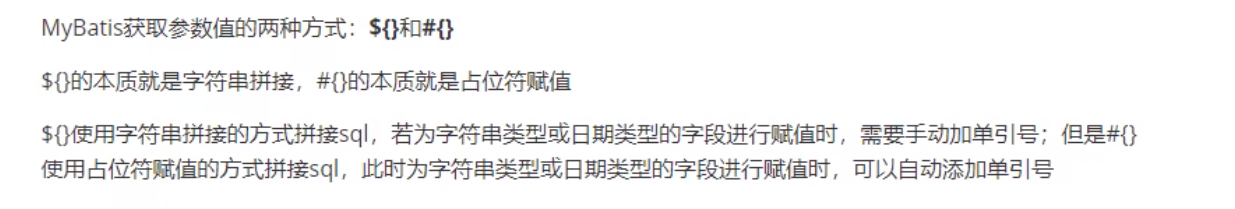




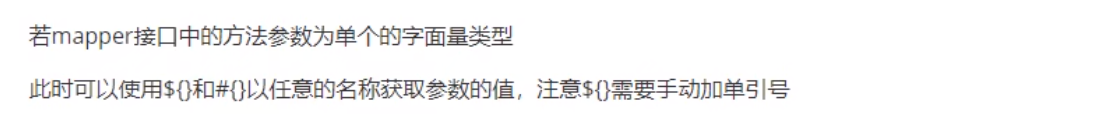
# 4、MyBatis的增删改查



# 5、MyBatis获取参数值的两种方式（重点）



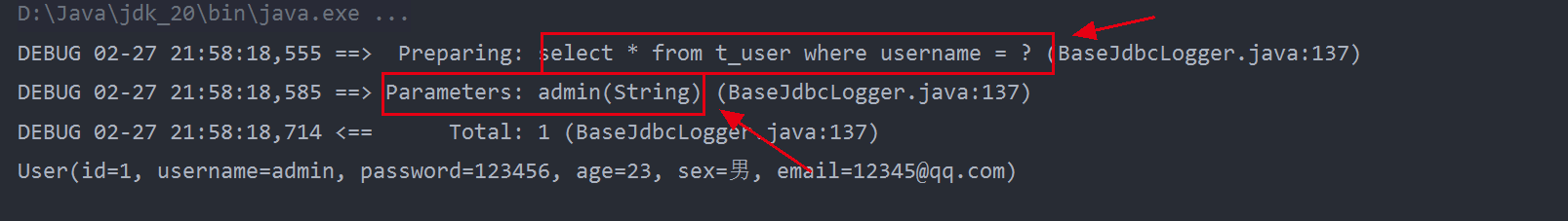
# 5.1、单个字段量类型的参数



测试案例：使用#{}方式，这中方式的本质就是jdbc中的占位符赋值

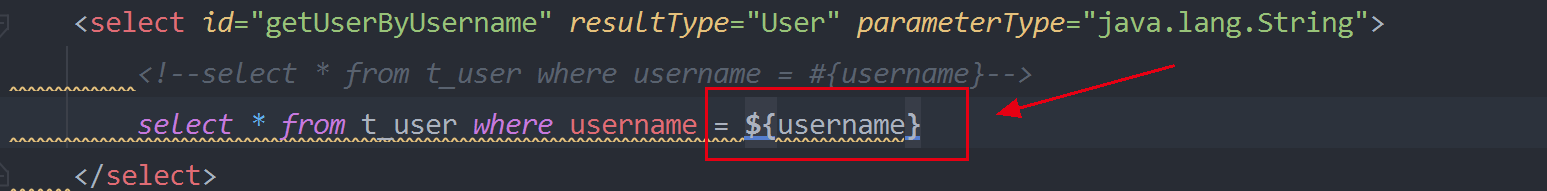


测试结果：

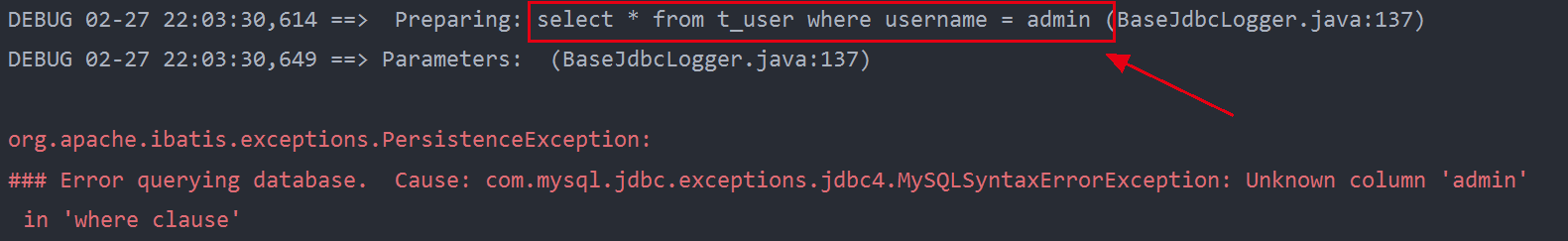


测试案例：使用${}范式，这种方式的本质就是jdbc中的字符串拼接，需要注意的点和jdbc中字符串拼接的问题一样，需要手动加上 单引号。

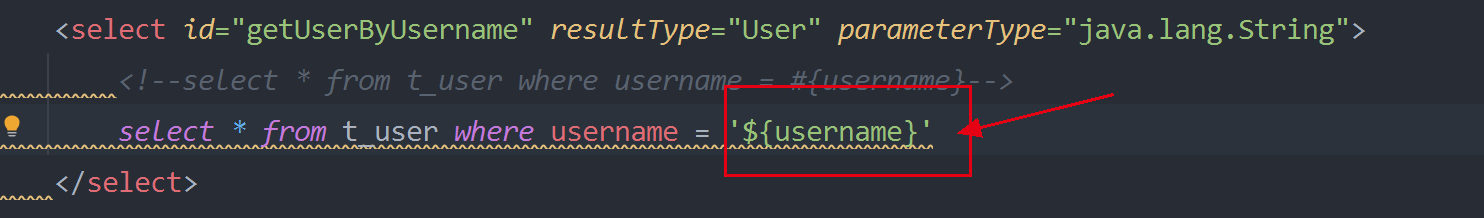
1、在${}不加单引号



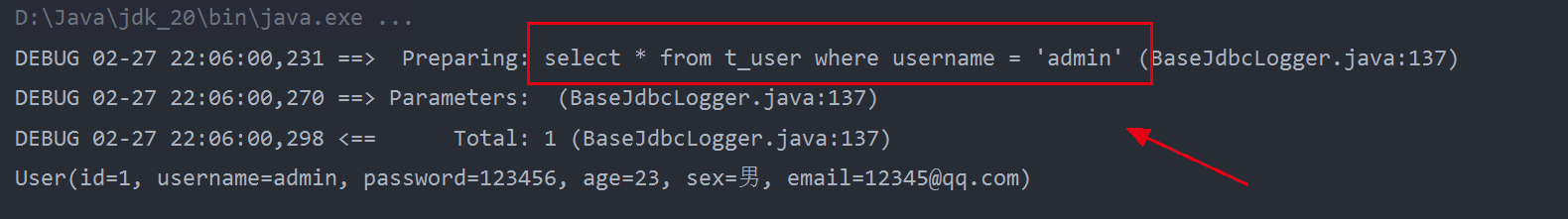
测试结果：报错，因为没有加单引号导致。



2、在${}加单引号



测试结果：



# 5.2、多个字面量类型的参数

mapper接口方法的参数为多个时：

\* 此时MyBatis会将这些参数放在一个map集合中，以两种方式进行存储。

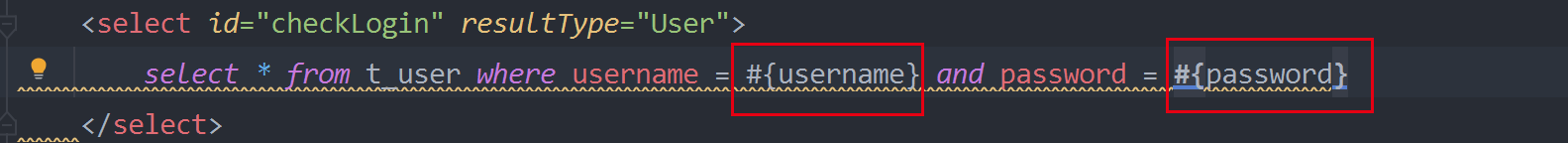
\* 2.1、以arg0,arg1...为键，以参数为值。

\* 2.2、以param1,param2...为键，以参数为值。

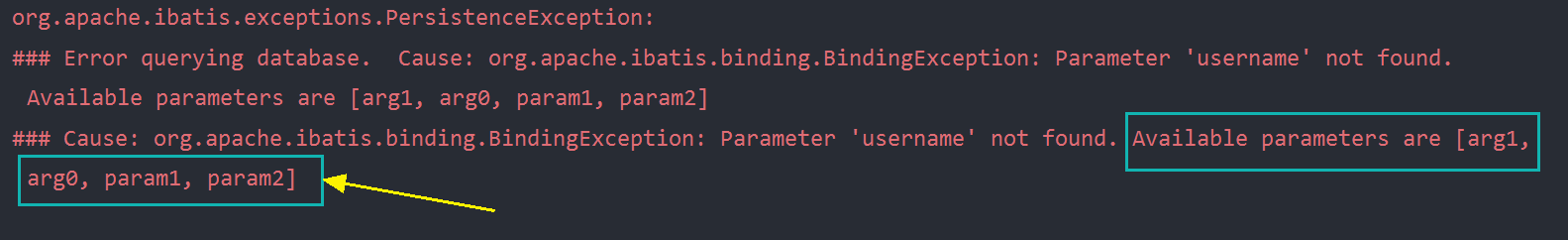
\* 因此只需通过#{} 和 ${}以键的方式访问值即可，但是需要注意${}的单引号问题。

测试案例：



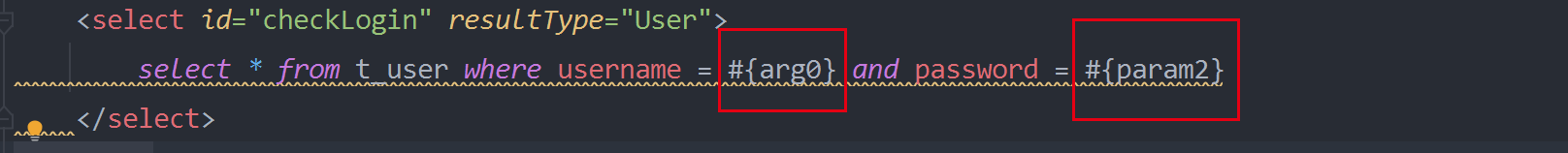
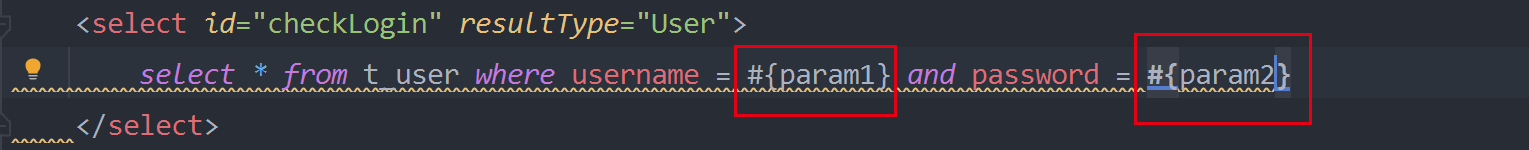


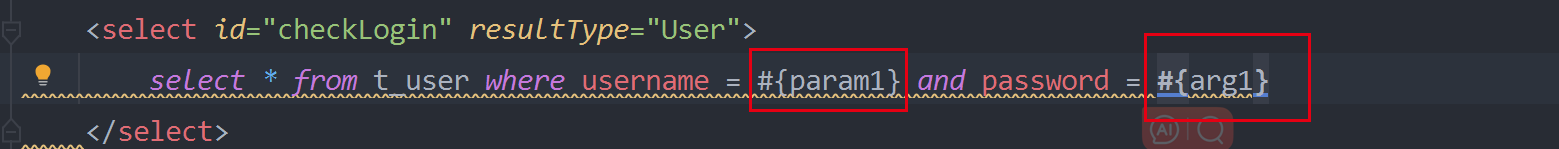
测试结果：



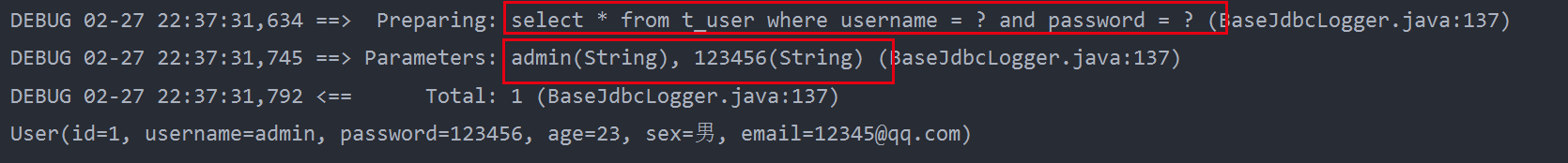
1、使用#{}方式：



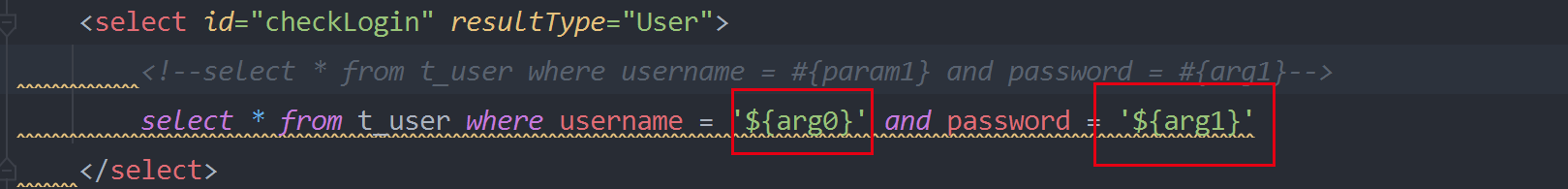




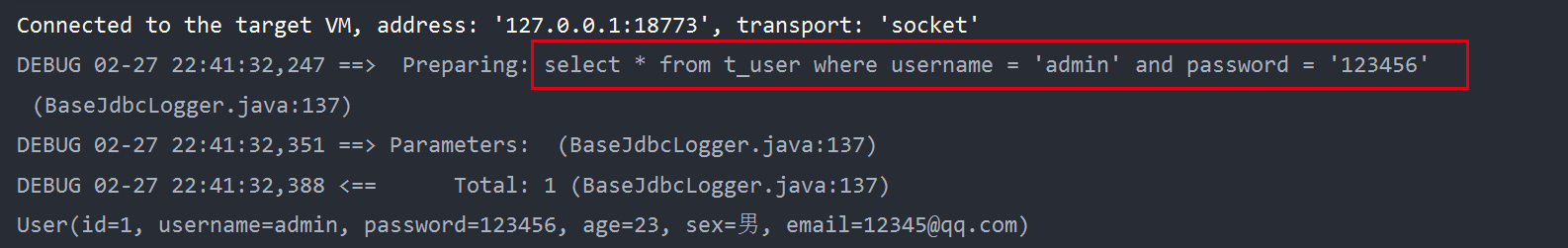
测试结果：



2、使用${}的方式同#{}，但是需要添加单引号，此处指展示一种；



测试结果：

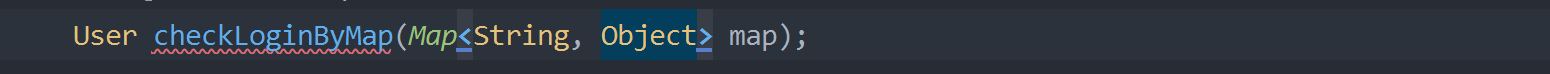


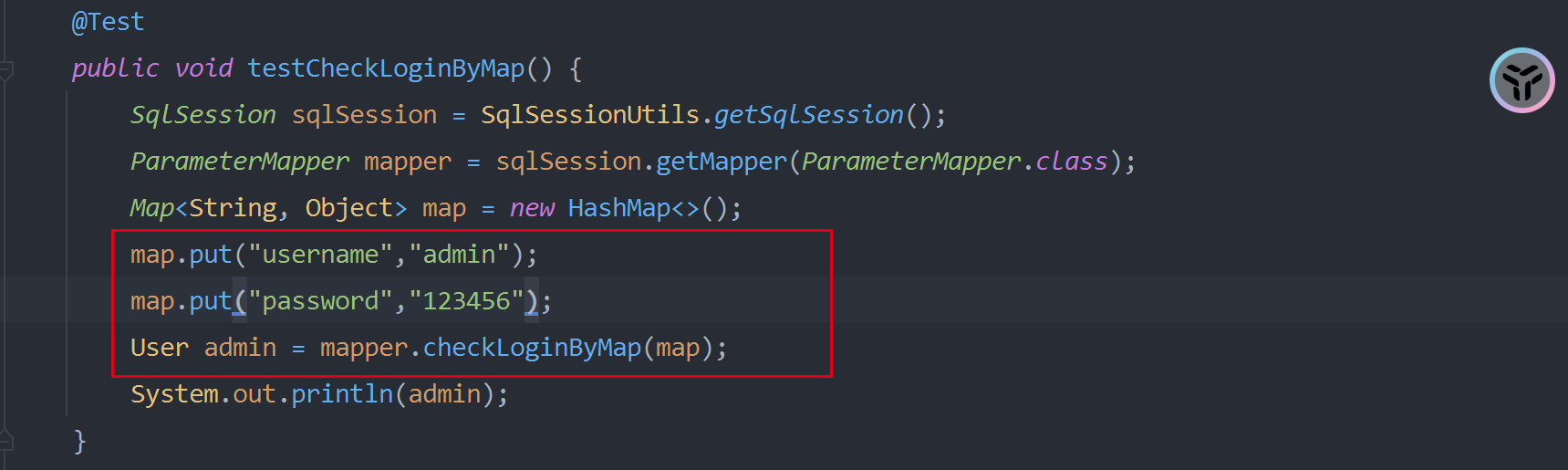
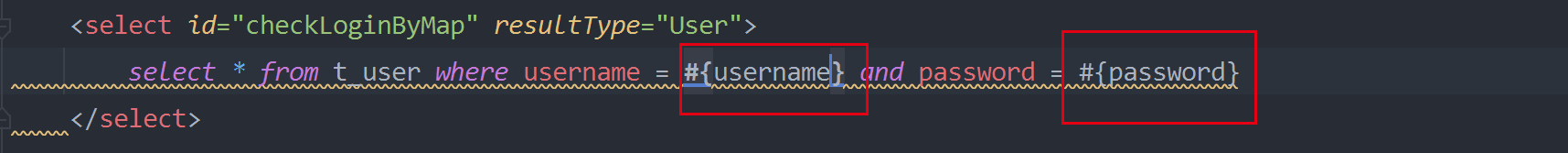
# 5.3、map集合类型的参数

mapper接口方法的参数有多个时，可以手动将这些参数放在一个map中存储

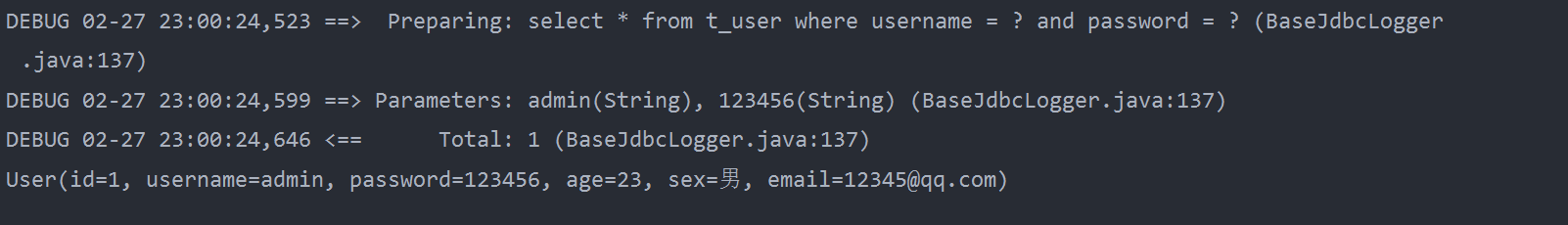
\* 只需要通过#{}和${}以键的方式访问值即可，但是需要注意${}的单引号问题

测试案例：





测试结果：

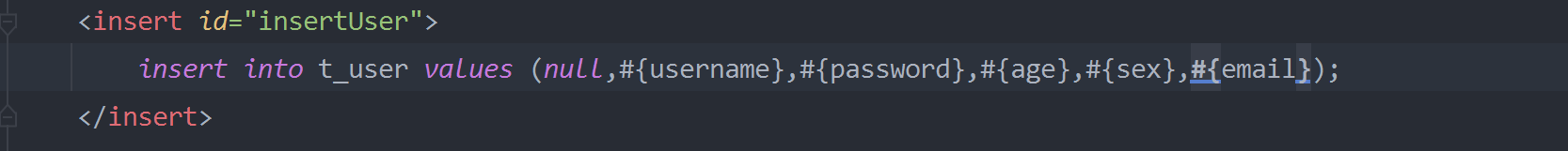
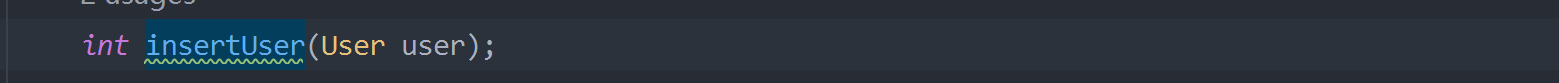


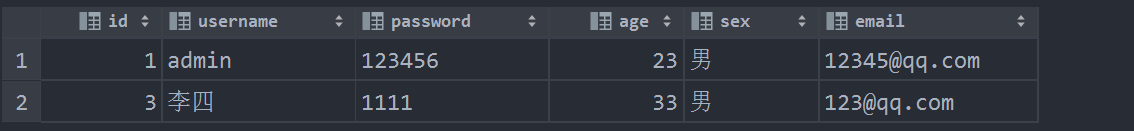
# 5.4、实体类类型的参数

mapper接口方法的参数是实体类类型的参数

\* 只需要通过#{}和${}以属性的方式访问属性即可，但是需要注意${}的单引号问题

测试案例：

测试结果：



# 5.5、使用@Param注解命名参数

使用@Param注解命名参数

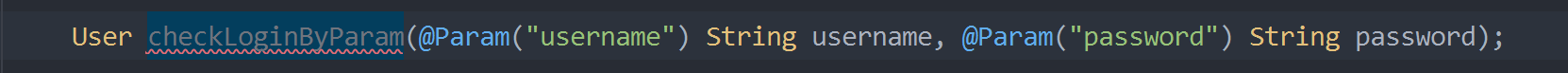
\* 此时MyBatis会将这些参数放在一个map集合中，以两种方式进行存储

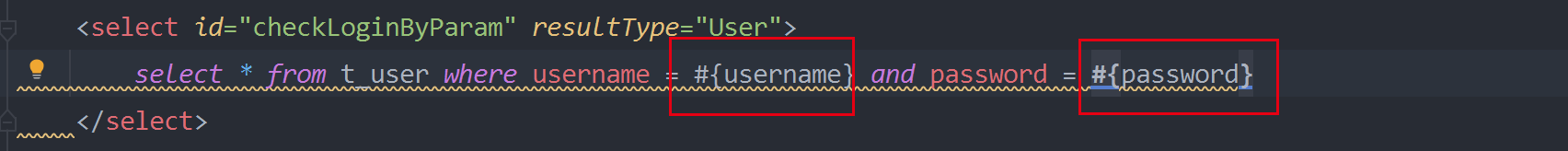
\* 5.1、以@Param注解的值为键，以参数为值

\* 5.2、以param1,param2...为键，以参数为值

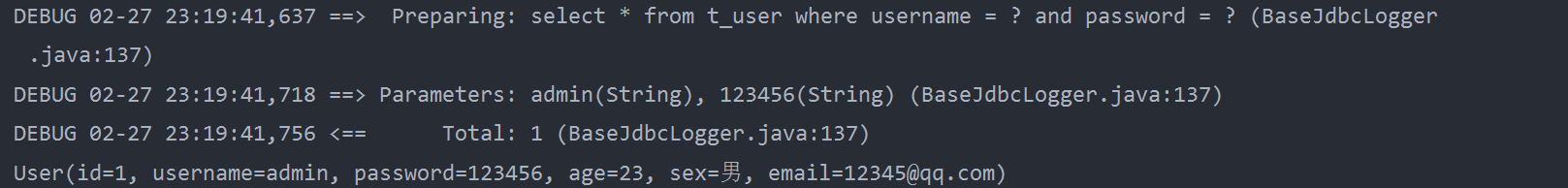
\* 因此只需通过#{} 和 ${}以键的方式访问值即可，但是需要注意${}的单引号问题

测试案例：





测试结果



# 5.6、@Param注解源码解析