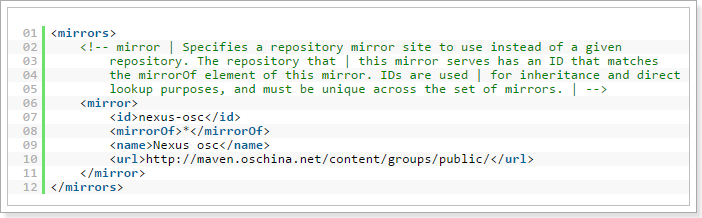
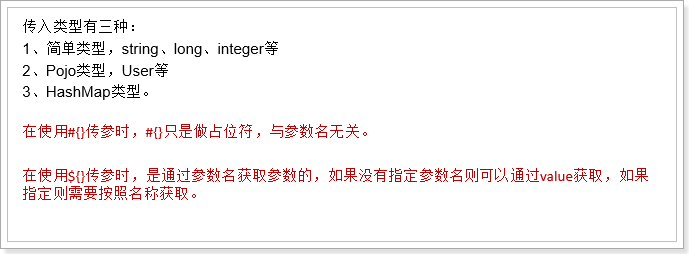
# Maven问题

1. 回家后连不到私服
   1. 在教室里将所有需要的依赖下载下来
   2. 换私服
      1. 使用apache的中央仓库，将maven的配置文件中的私服配置注释掉即可
      2. 可以使用国内私服，开源中国私服，<http://maven.oschina.net/help.html>  
         
2. pom.xml中报红叉，依赖丢失
   1. maven update
   2. 本地仓库中的依赖没有下载完整，有没有以.lastUpdate结尾文件，解决方案：删除该文件
      1. 手动删除
      2. 删除脚本
         1. 1、切换到maven的本地仓库
         2. 2、在当前目录打开cmd命令行
         3. 3、执行命令：for /r %i in (\*.lastUpdated) do del %i

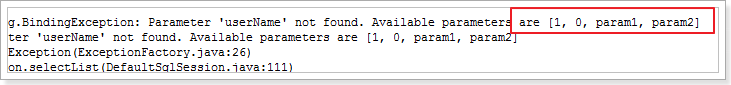
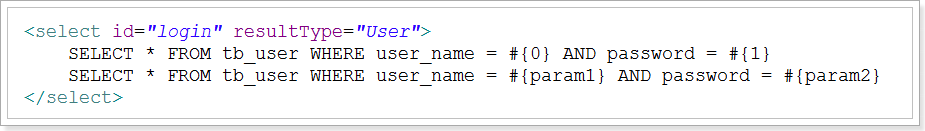
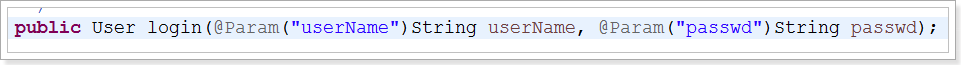
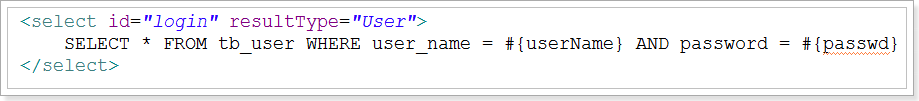
# Mybatis高级

## parameterType的传入参数



## parameterType的传入多个参数

当DAO接口中需要传递多个参数时有两种方法传递：

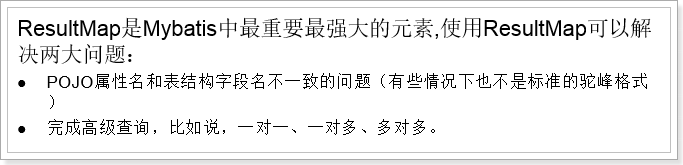
1. 使用默认规则获取参数；  
     
   
2. 使用@Param注解表明参数传递  
     
   

## ResultType结果输出



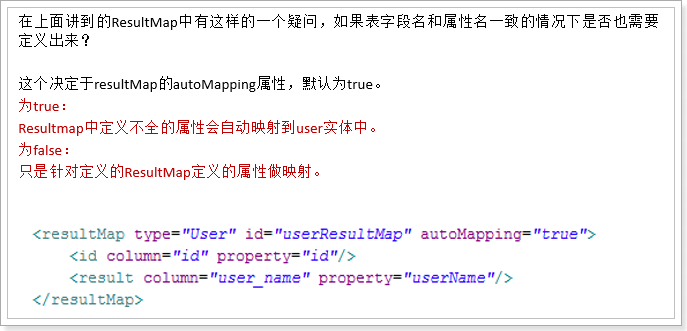
作业：实现用户数量的查询。

## ResultMap





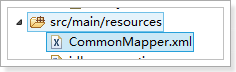
## 自动映射

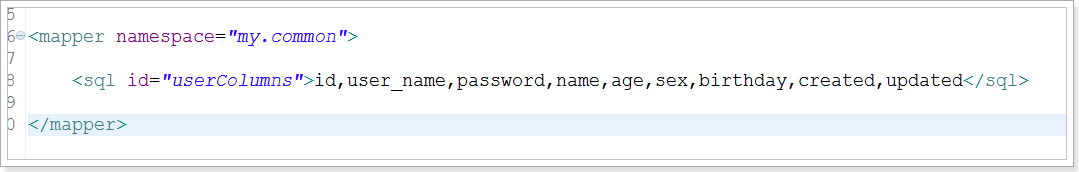


## SQL片段

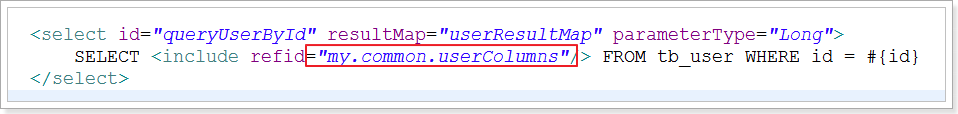


跨命名空间引用：

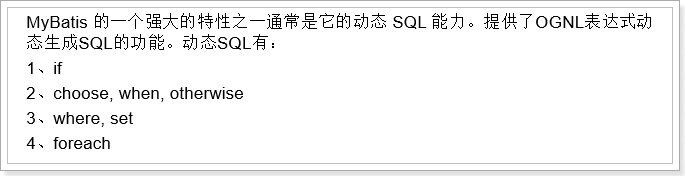




命名空间.sql的id：

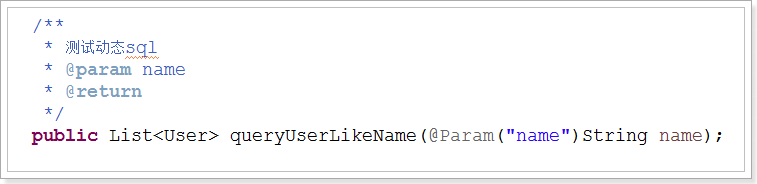


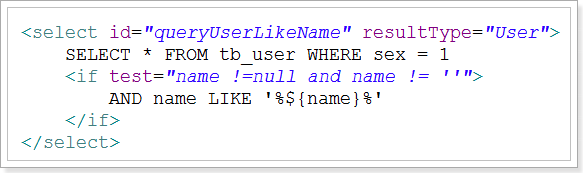
## 动态SQL



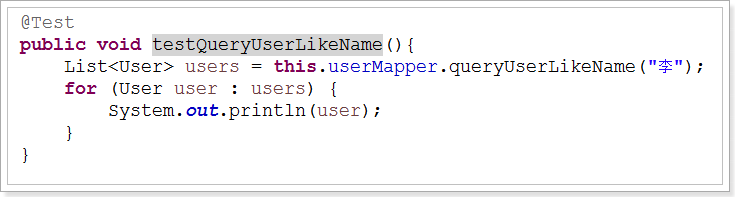
### If

需求1：查询男性用户，如果输入了姓名，进行模糊查找。

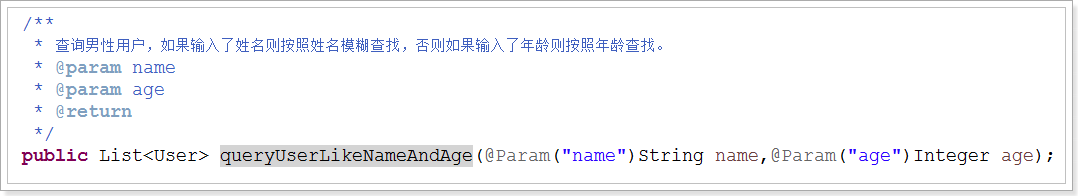


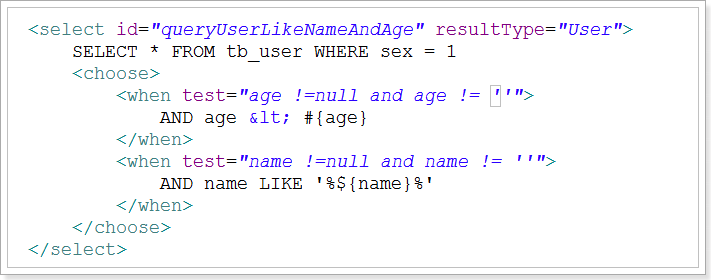


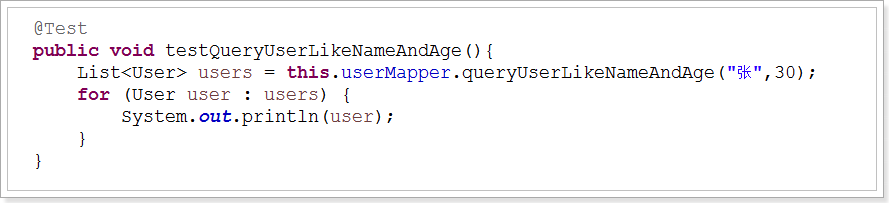
测试：



### choose, when, otherwise

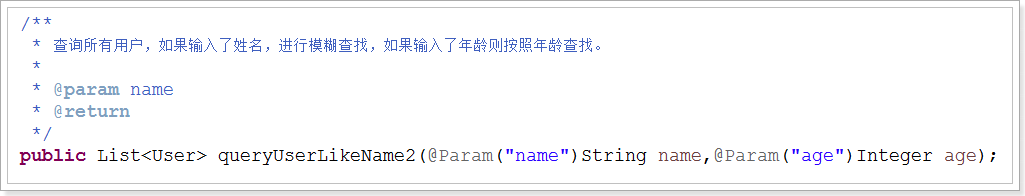


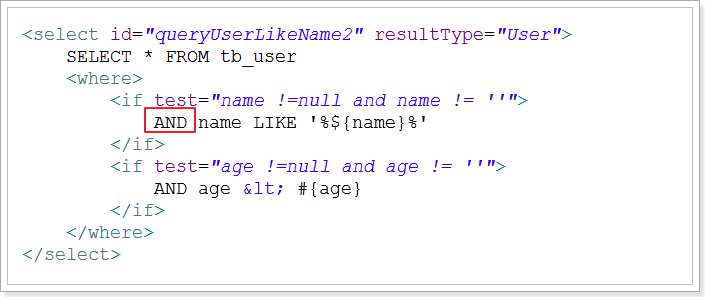


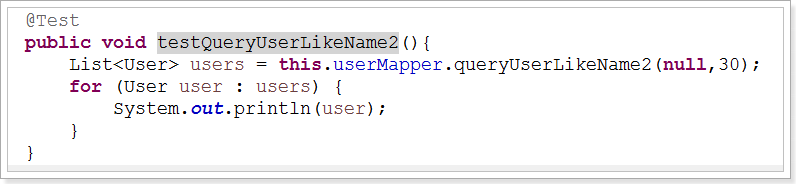


### where, set

查询所有用户，如果输入了姓名，进行模糊查找，如果输入了年龄则按照年龄查找。





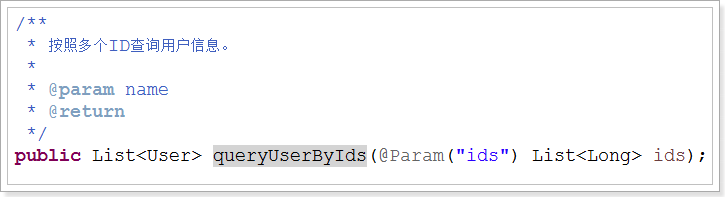


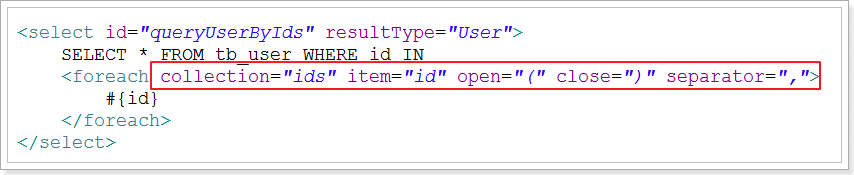
SET 标签



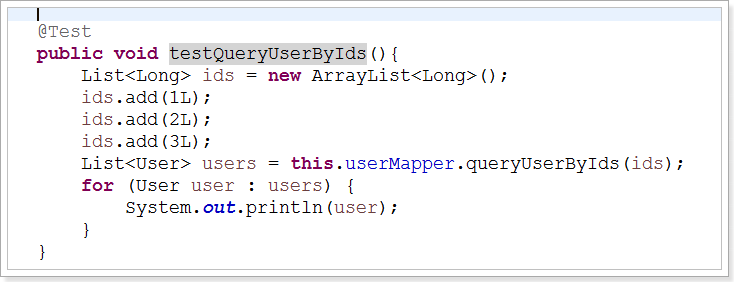
### foreach

按照多个ID查询用户信息。





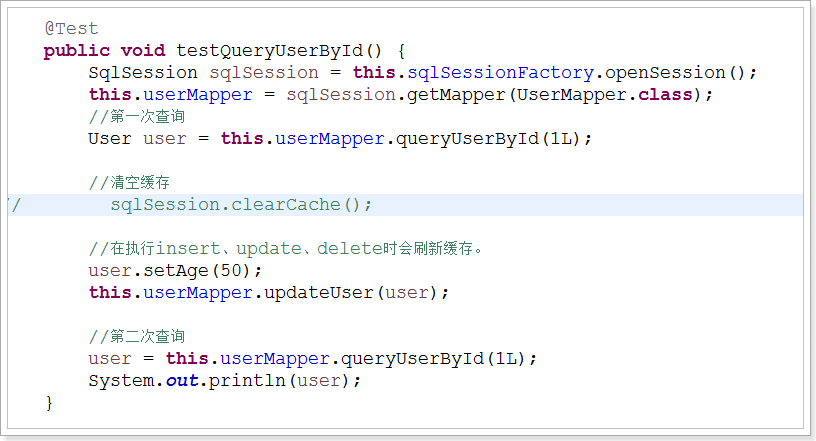
id IN (?,?,?)



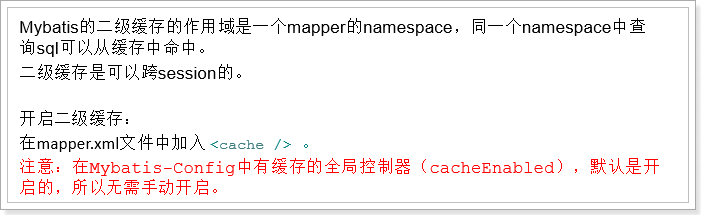
# 缓存

## 一级缓存



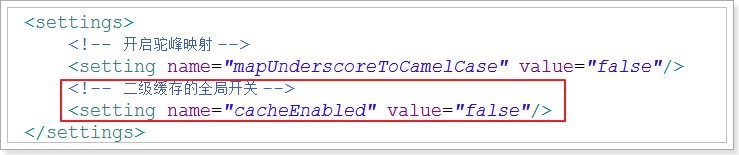


## 二级缓存

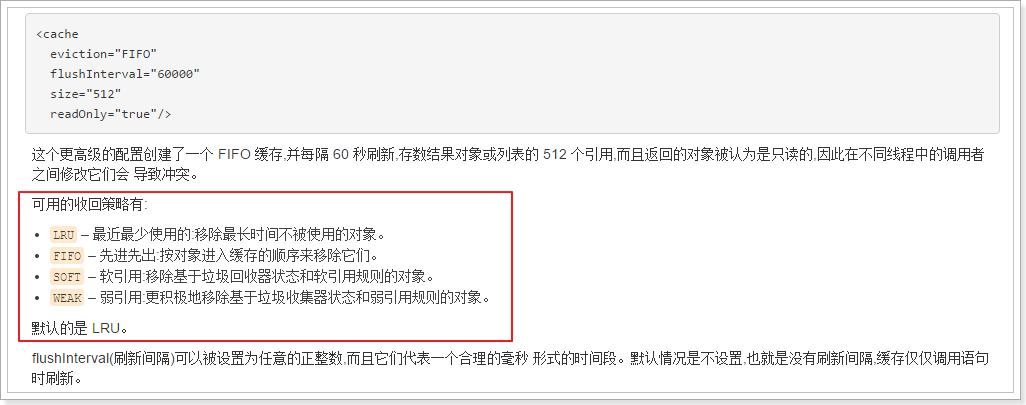




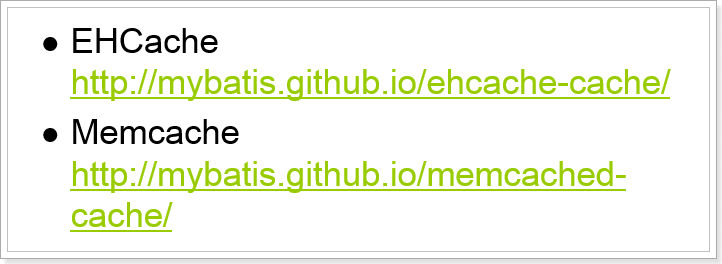
禁用二级缓存：



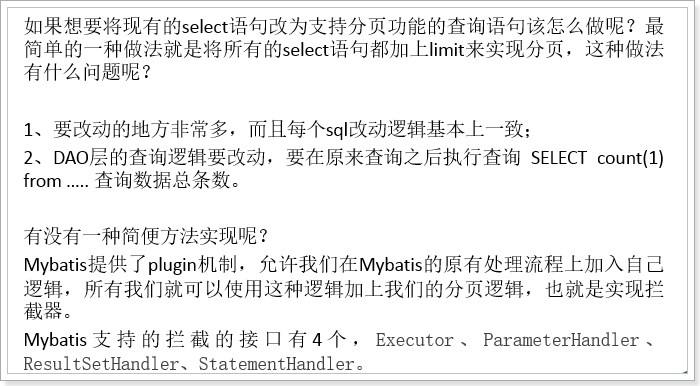
二级缓存高级用法：



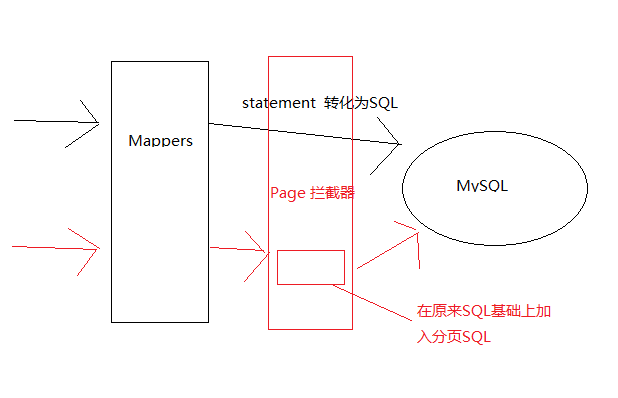
## 使用第三方缓存实现二级缓存



# 分页插件



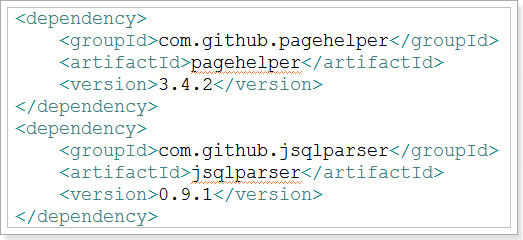
## Mybatis的plugin实现原理

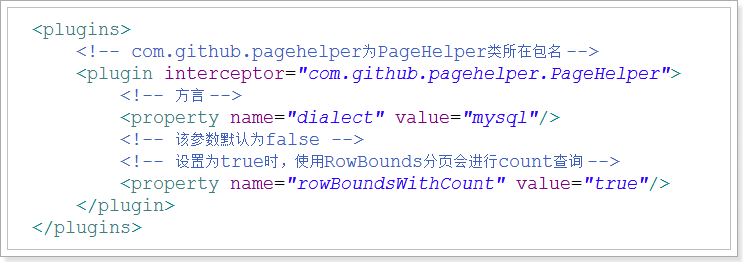


## 使用PageHelper实现分页



## 分页助手的使用

1. 导入依赖  
   
2. 配置拦截器

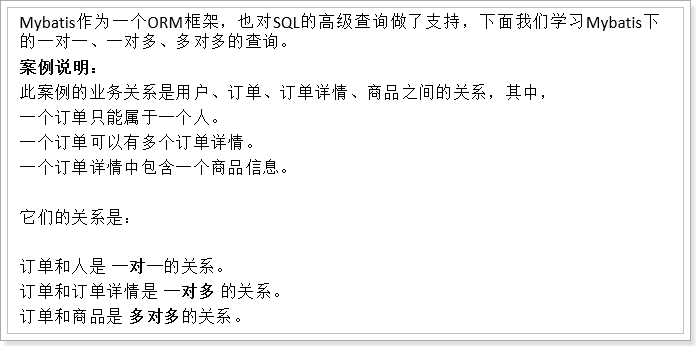


1. 设置分页参数和获取分页信息  
   

## 重要提示



# 高级查询



关系：



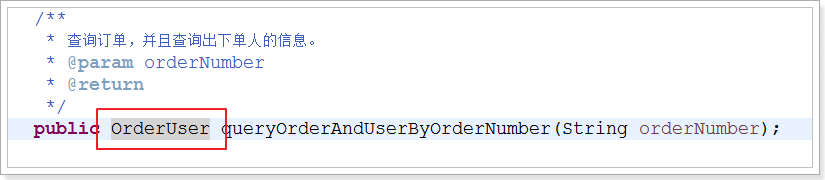
## 一对一查询

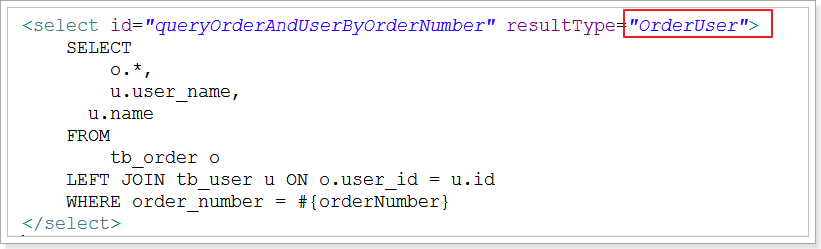
**一对一查询：**查询订单，并且查询出下单人的信息。

### 第一种实现

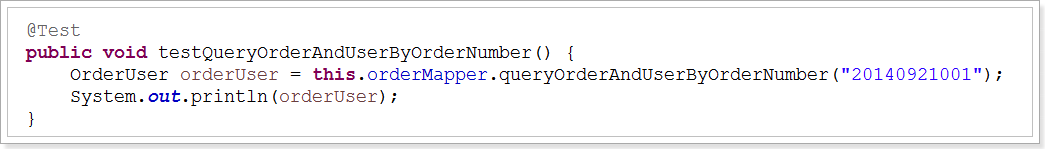


实现：





测试：

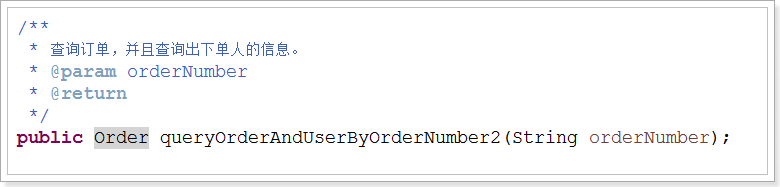


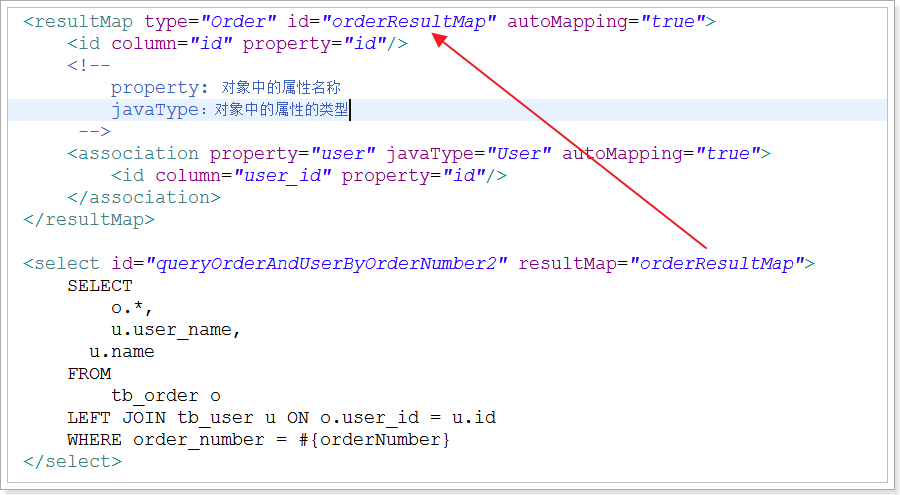
### 第二种实现

在Order对象中添加User对象，完成映射。

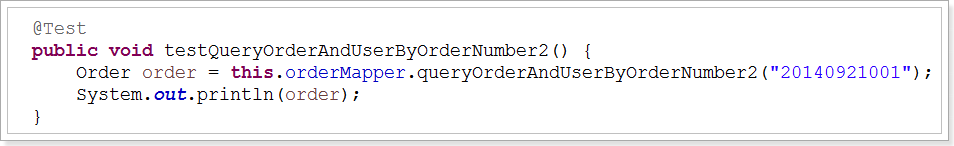


实现：





测试：



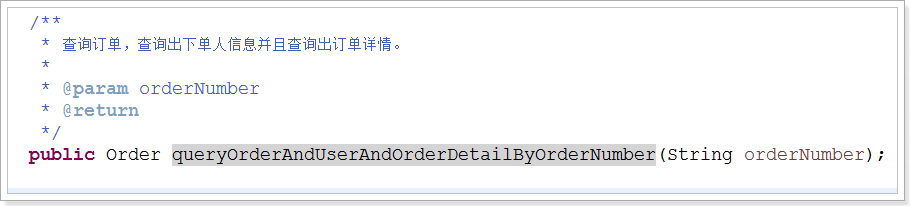
## 一对多查询

需求：查询订单，查询出下单人信息并且查询出订单详情。

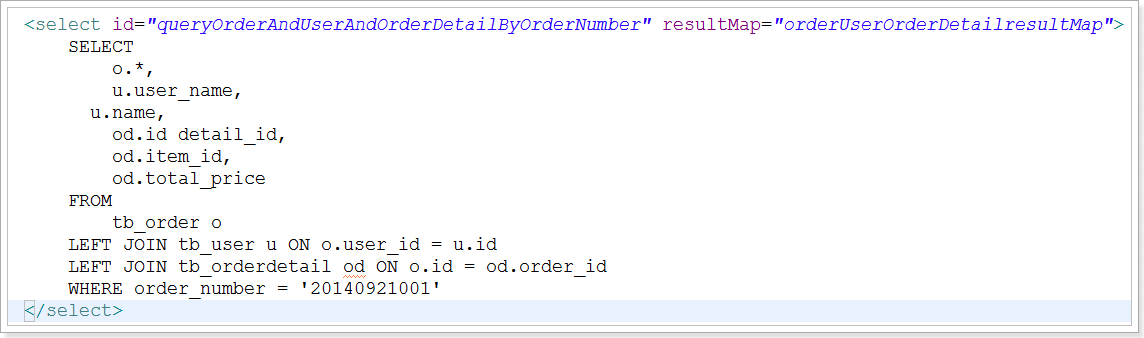
解决方案：

只能使用resultMap来实现结果集的映射。

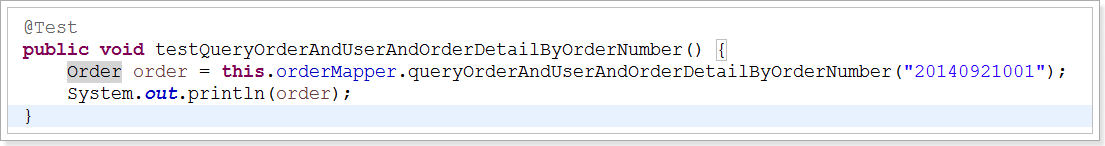
实现：







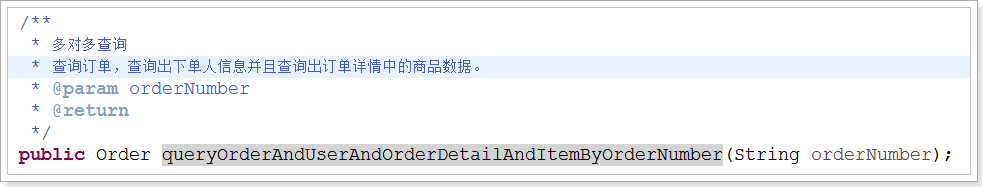
测试：



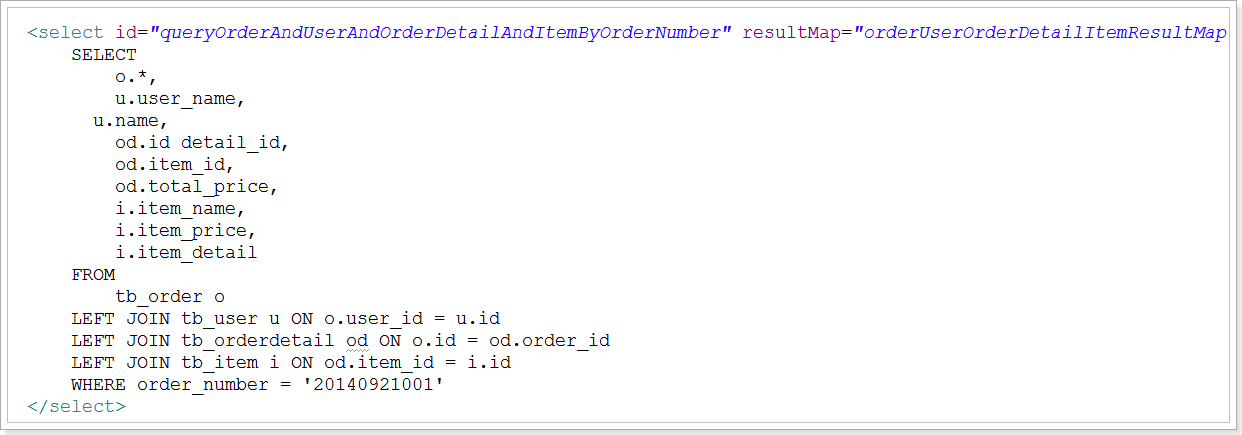
## 多对多查询

需求：查询订单，查询出下单人信息并且查询出订单详情中的商品数据。

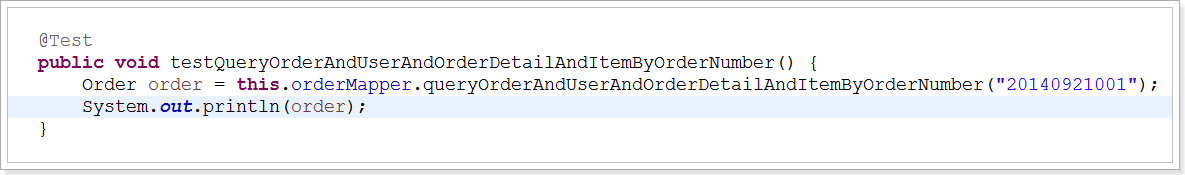
实现：



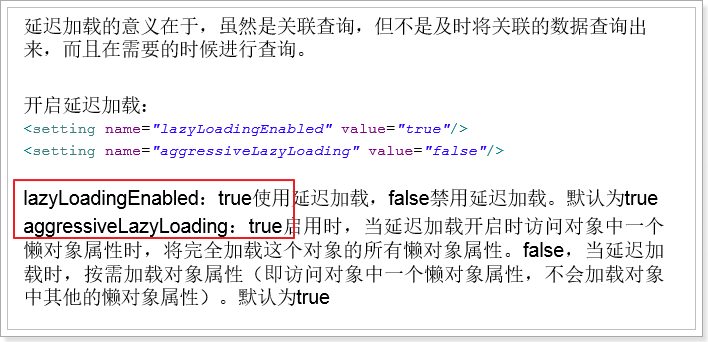




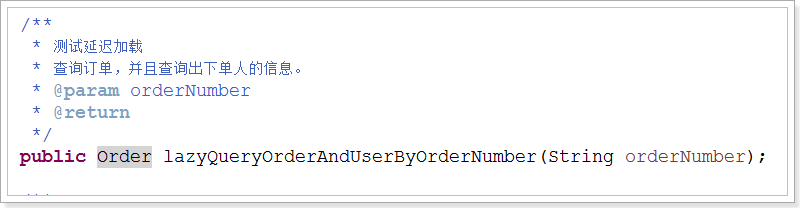
测试：

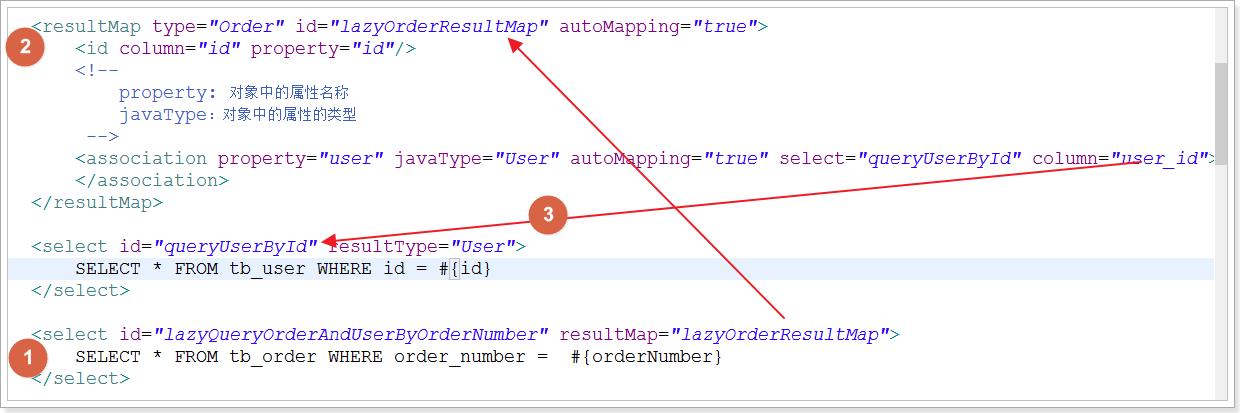


# 延迟加载



实现：





测试：

