# 延迟加载

## 什么是延迟加载

延迟加载又叫懒加载，也叫按需加载。也就是说先加载主信息，在需要的时候，再去加载从信息。

在mybatis中，resultMap标签 的association标签和collection标签具有延迟加载的功能

## 原理

使用CGLIB创建目标对象的代理对象，当调用目标方法时，进入拦截器方法，比如调用a.getB().getName()，拦截器invoke()方法发现a.getB()是null值，那么就会单独发送事先保存好的查询关联B对象的sql，把B查询上来，然后调用a.setB(b)，于是a的对象b属性就有值了，接着完成a.getB().getName()方法的调用。这就是延迟加载的基本原理。

## 使用延迟加载意义

在进行数据查询时，**为了提高数据库查询性能**，尽量使用单表查询，因为单表查询比多表关联查询速度要快。

如果查询单表就可以满足需求，一开始先查询单表，当需要关联信息时，再关联查询，当需要关联信息再查询这个叫延迟加载。

mybatis中resultMap提供延迟加载功能，通过resultMap配置延迟加载。

## 配置mybatis支持延迟加载



<!-- 全局配置参数 -->

<settings>

<!-- 延迟加载总开关 -->

<setting name=*"lazyLoadingEnabled"* value=*"true"* />

<!-- 设置按需加载 -->

<setting name=*"aggressiveLazyLoading"* value=*"false"* />

</settings>

## 一对一延迟加载实现

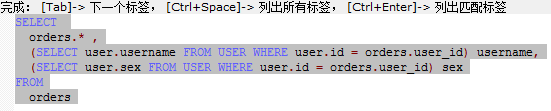
### 需求：

查询订单及用户的信息，一对一查询。

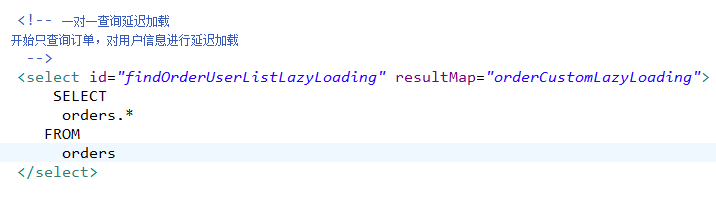
刚开始只查询订单信息

当需要用户时调用 Orders类中的getUser()方法执行延迟加载 ，向数据库发出sql。

实际sql：



### mapper.xml



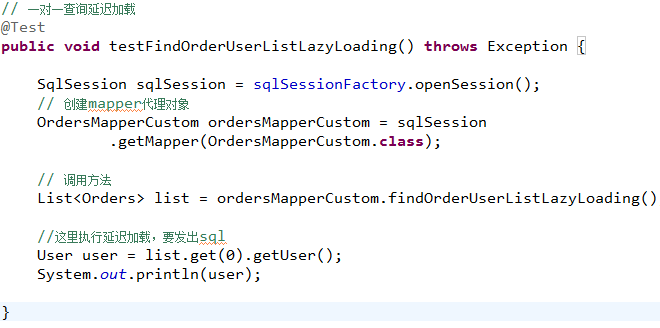
resultMap



### mapper.java



### 测试代码



## 一对多延迟加载

一对多延迟加载的方法同一对一延迟加载，在collection标签中配置select内容。

# resultType、resultMap延迟加载使用场景总结

延迟加载：

延迟加载实现的方法多种多样，在只查询单表就可以满足需求，为了提高数据库查询性能使用延迟加载，再查询关联信息。

mybatis提供延迟加载的功能用于service层。

resultType：

作用：

将查询结果按照sql列名pojo属性名一致性映射到pojo中。

场合：

常见一些明细记录的展示，将关联查询信息全部展示在页面时，此时可直接使用resultType将每一条记录映射到pojo中，在前端页面遍历list（list中是pojo）即可。

resultMap：

使用association和collection完成一对一和一对多高级映射。

association：

作用：

将关联查询信息映射到一个pojo类中。

场合：

为了**方便获取关联**信息可以使用association将关联订单映射为pojo，比如：查询订单及关联用户信息。

collection：

作用：

将关联查询信息映射到一个list集合中。

场合：

为了**方便获取关联信息**可以使用collection将关联信息映射到list集合中，比如：查询用户权限范围模块和功能，可使用collection将模块和功能列表映射到list中。