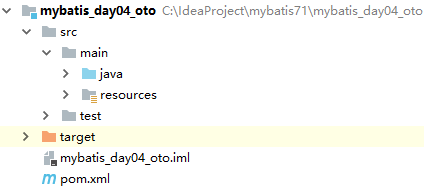
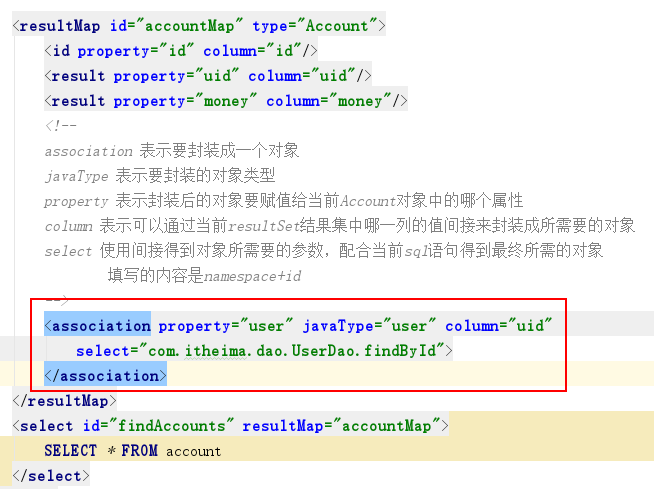
# 一对一关联查询进行改造变成分步查询

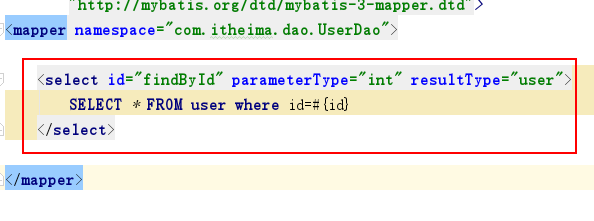
## 复制昨天一对一的项目改名字



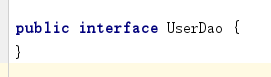
## 修改AccountDao.xml中映射文件



## 创建UserDao.xml映射文件

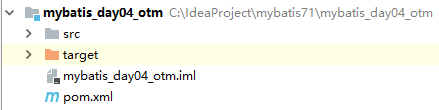


## 创建UserDao接口



# 一对多关联查询进行改造变成分步查询

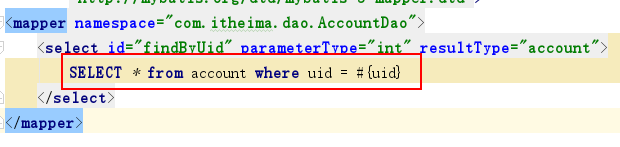
## 复制昨天一对多项目改名字



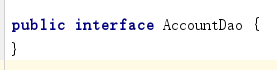
## 修改UserDao.xml中的映射文件



## 创建AccountDao.xml

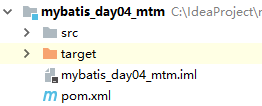


## 创建AccountDao接口



# 多对多关联查询进行改造变成分步查询

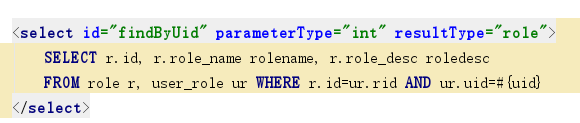
## 复制昨天多对多项目改名字



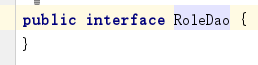
## 修改UserDao.xml中的映射



## 提供RoleDao.xml



## 提供RoleDao接口



# 延迟加载的概念

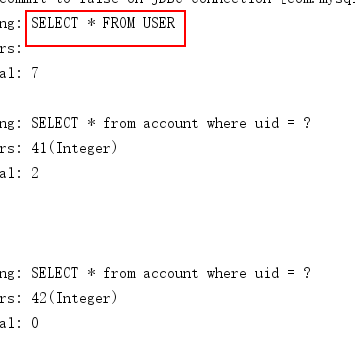
延迟加载是一种均衡服务器压力的技术，就是我们需要什么数据的时候，才执行对应的数据库操作，否则先不查询。前提条件必须是采用mybatis分步查询的方式。

# 延迟加载的使用

## 打开延迟加载配置



## 测试【略】



# Mybatis缓存的使用

## 一级缓存

Mybatis默认情况下一级缓存就是开启的状态。就是在我们要重复查询某一个对象的时候，只会在第一次查询的时候去数据库中查询，其余都直接从缓存中拿数据。除非中间出现了增删改等SqlSession的提交或者关闭操作。可见一级缓存是SqlSeesion级别的。只要SqlSession没有被提交或关闭，数据就一直在。

回顾：

脏读：一个线程读到了其他线程未提交的数据。

不可重复读：一个线程在操作数据的时候，该数据被其他线程修改了。

幻读：一个线程在按照条件查询的时候，其他线程插入或者删除了满足该条件的数据，导致前后查询数量不一致。

### 在一对一项目环境中验证一级缓存的存在

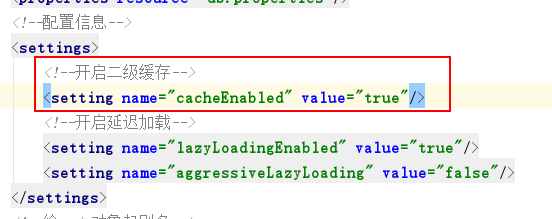


## 二级缓存

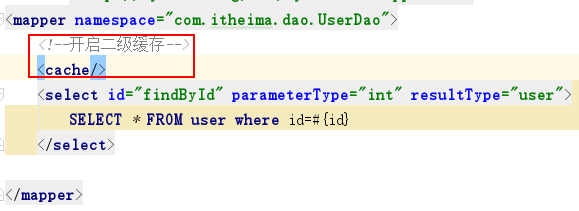
二级缓存需要手动开启，而且要使用二级缓存的对象必须实现序列化接口。即便是SqlSession对象被关闭了。数据依然存在。所以要谨慎使用。

### 开启二级缓存

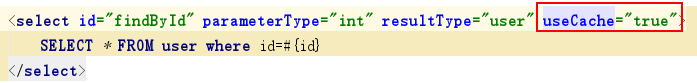
#### 在主配置文件中开启二级缓存【可省略】



#### 在mapper映射文件中开启二级缓存【必须配置】

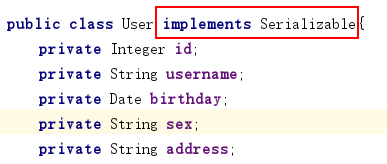


#### 在需要缓存的sql语句标签中开启二级缓存



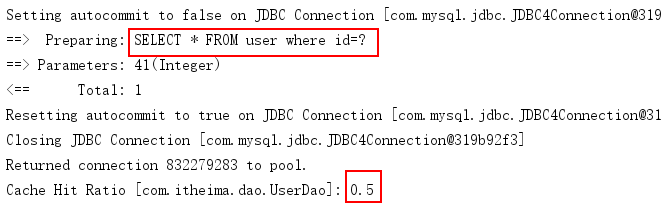
此步骤可以省略，因为默认该属性就是true。但是配置完之后，一定要记得把该mapper中其他所有查询标签里面都加上useCache="false"。

### 让使用二级缓存的对象实现序列化



### 验证二级缓存是否其作用

依然使用刚才一级缓存的测试类，直接测试，不做任何修改。



## Mybatis缓存使用的总结

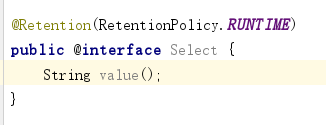
在mybatis中如果要查询一条记录，默认情况下会先从一级缓存找数据，如果找不到，再去二级缓存找，如果二级缓存中有就拿出来，没有最后再去数据库中查询。二级缓存有命中率的概念。

# Mybatis注解开发自定义【环境还是第一天自定义项目的环境】

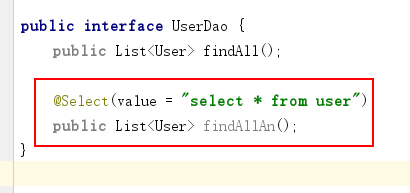
## 说明

现在使用注解开发，sql语句就不需要写在映射文件中了。而是直接放在接口的注解上。那么就不需要提供映射文件了。注解开发和xml开发无法混合使用。

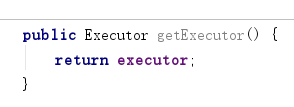
## 自定义一个Select注解，做查询使用



## 提供一个接口在上面添加Select注解



## 在DefaultSqlSession中提供一个Executor的get方法

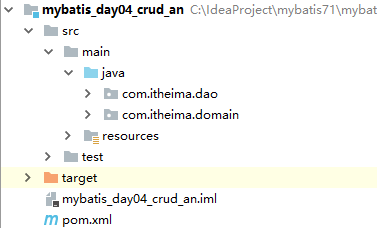


## 在动态代理第三个参数动作类里面添加注解的分支

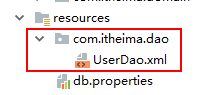


# Mybatis注解开发实现简单的CRUD

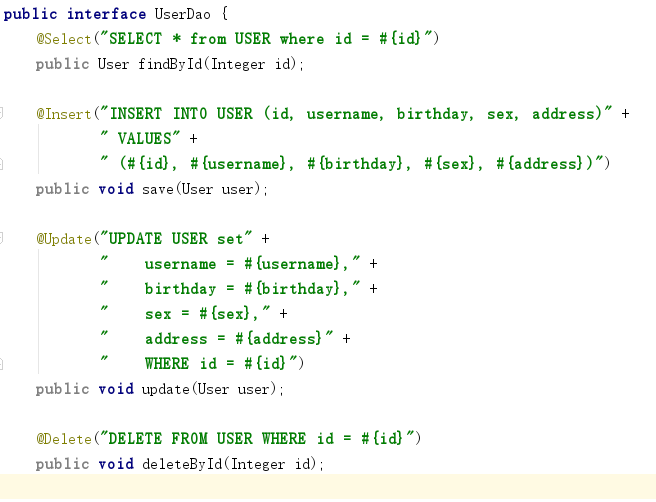
## 拿第三天反射的dao改个名字



## 直接删除所有xml映射文件



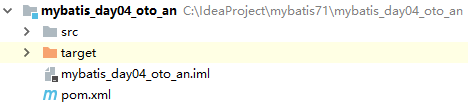
## 在接口上添加注解



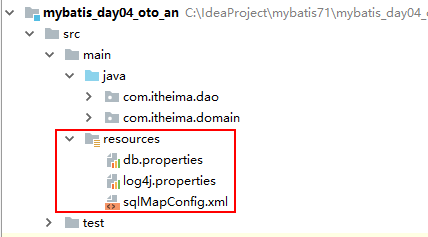
## 测试【还是原来的】

# 一对一注解开发

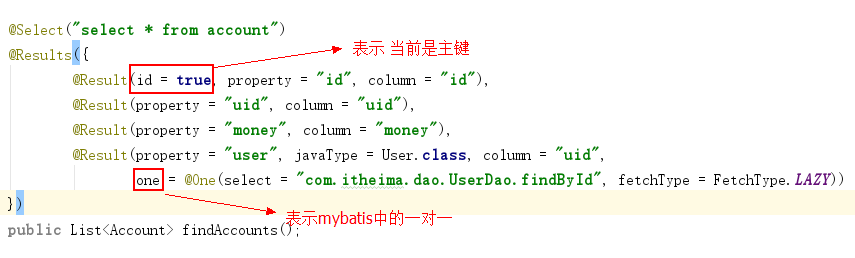
## 复制今天一对一项目改名字



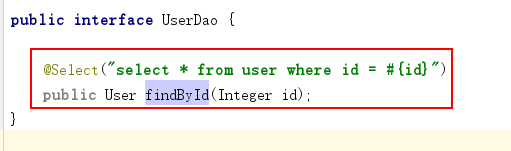
## 删除所有xml映射文件



## 在AccountDao接口中添加注解



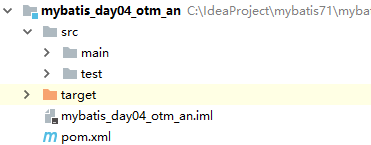
## 在UserDao接口中添加注解



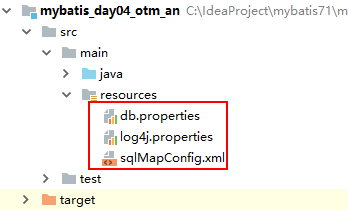
## 测试【略】

# 一对多注解开发

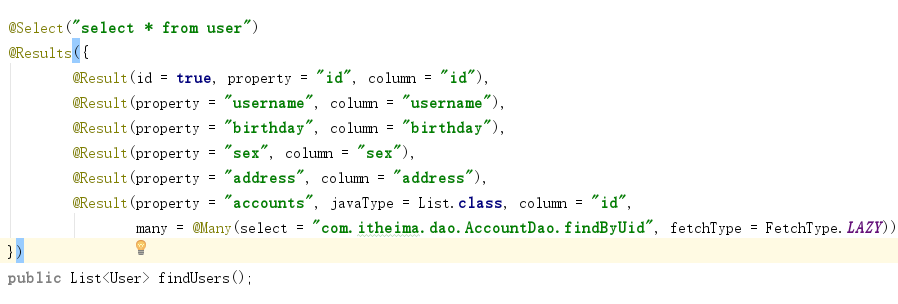
## 复制今天一对多项目改名字



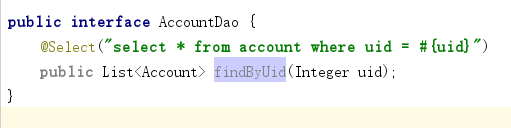
## 删除所有xml映射文件



## 在UserDao接口中添加注解



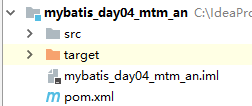
## 在AccountDao接口中添加注解



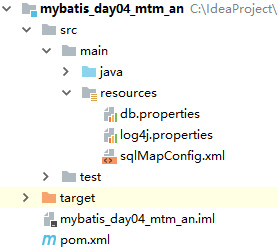
## 测试【略】

# 多对多注解开发

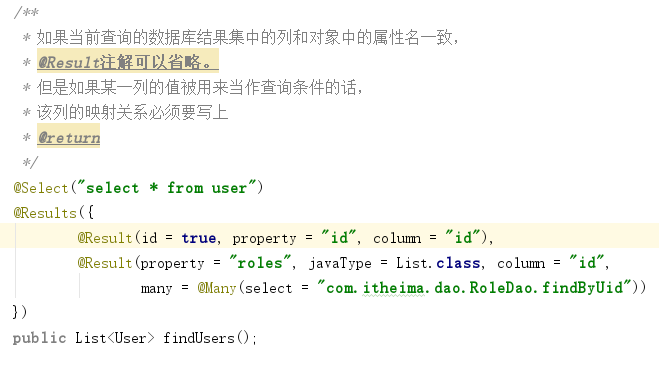
## 复制今天多对多项目改名字



## 删除所有xml映射文件

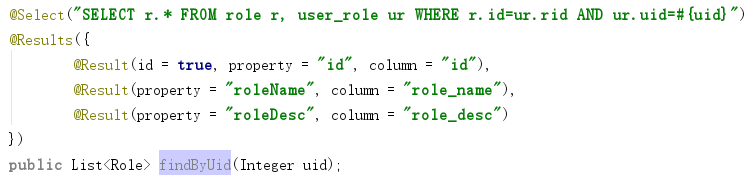


## 在UserDao接口中添加注解

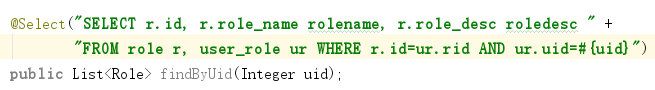


## 在RoleDao接口中提供注解

### 方式一



### 方式二



## 测试【略】

# Maven回顾

Maven有两大核心功能：依赖管理和一键构建。

依赖管理就是对jar包的管理过程。

一键构建，就是使用maven自身继承的tomcat插件对项目进行构建。

一件构建主要是通过maven提供的命令来完成的。

这些命令可以分成三个生命周期：清理生命周期，默认生命周期，站点生命周期。

清理生命周期的命令是：clean。作用是删除Target目录。

默认生命周期的命令有：

Compile：编译src/main/java下的代码

Test：编译src/test/java下的代码

Package：打包，把包打在Target目录下

Install：安装，把项目安装到本地仓库

Deploy：发布，把项目发布到远程仓库

默认生命周期中每一个命令都必须顺序执行，不能跳着走。

也就是说如果执行了其中任一命令，就表示之前所有命令肯定都已经执行过了。