应用系统体系架构 — 作业4

# 学号： 522031910213 姓名： 朱涵 得分：

## 编写使用Redis进行数据缓存的代码

## 思路：

## 我打算在DAO层使用缓存，首先判断DAO层里哪些接口的返回结果适合缓存到内存里。缓存数据应该满足多读少写，像上课提到的stock字段属于多写，应该提取到另一个实体类里不作缓存，所以我新建了BookDetails实体类，包含了stock、sales等经常需要变动的字段。另外，并不是所有接口都适合作缓存，比如搜索书籍时用到的searchBooks接口，如果全部作缓存，每一个关键词就会对应一堆数据，内存根本放不下，并且不同用户查询相同关键词的概率也不大，因此在这里不作缓存，即使后续要作缓存也应该是提取最热门的几个搜索关键词的结果作缓存。最终在bookDAO层里，只对getBookById接口结果中的Book实体作缓存。

## 除了bookDAO，我也在UserDAO里作了缓存，包括getUserByUsername和getUserById两个接口。前者经常用于其他接口的鉴权，可以放入缓存加快所有需要鉴权的接口的速度；后者则是用于展示某个用户的个人信息时需要用到，在用户量不大的情况下，也可以适当作缓存。

## 具体实现：

## 按照教程配置一下RedisConfig，主要就是要自己注册一个RedisTemplate的Bean，并自定义一下键值序列化器。然后封装一个RedisUtil，编写一些常用的接口，比如set、get和delete。为了区分不同的DAO层的缓存键，在前面加上前缀（比如“book::”） 并加上标识唯一键的属性（比如“id::”）为了每一个DAO层实例注入属于自己的RedisUtil，声明scope为prototype，这样也可以适当并发。

## 然后get一个数据时，先去redis查找，如果有就返回，如果没有就去数据库查找，set到redis里然后返回

## B.处理通过Redis写数据的方式

## 增删改三个操作里，对于删除，只需要删除对应键的缓存即可，对于增和改，在DAO层都是通过save接口实现的，可以根据数据的id是否为空来判断是增还是改，当然也可以不管增改操作都重新set到redis里。我采用了后者，把所有新增的数据项也放入了redis。一个注意点是，**先保存到数据库后再保存到缓存里**，这样不会出现数据缺少ID字段而导致保存到缓存里的键错误的情况。

## 解释运行过程中的日志输出

## 以BookDAO为例，调用几个接口看看日志输出。

## 第一次调用getBook，输出下图：

## 

## 即从数据库里查询，然后保存到缓存里。

## 第二次开始调用getBook，输出下图：

## 即从缓存里查询。

## 然后调用updateBook，输出下图：

## 

## 即保存新的book到缓存里。

## 最后调用deleteBook，输出下图：

## 

## 即从缓存里删除对应键值。

## 关闭redis后，输出以下日志：

## 

## 即连接断开，尝试重连。

## 如果继续调用getBook接口，返回结果：

## 

## 即因为redis异常而查询失败。

## 另外附一个查询效果：（数据库位于服务器上，redis位于本地）

## 调用getUser接口，第一次如下：

## 第二次如下：

## 看到第一次延迟为37ms，第二次为6ms，差一个数量级，说明第一次是从服务器查询的，而第二次是从本地查询的，redis缓存确实生效了。（调用其他没有redis缓存的接口多次，延迟基本不变，所以不可能是数据库自身缓存导致延迟减少）