体系架构作业5

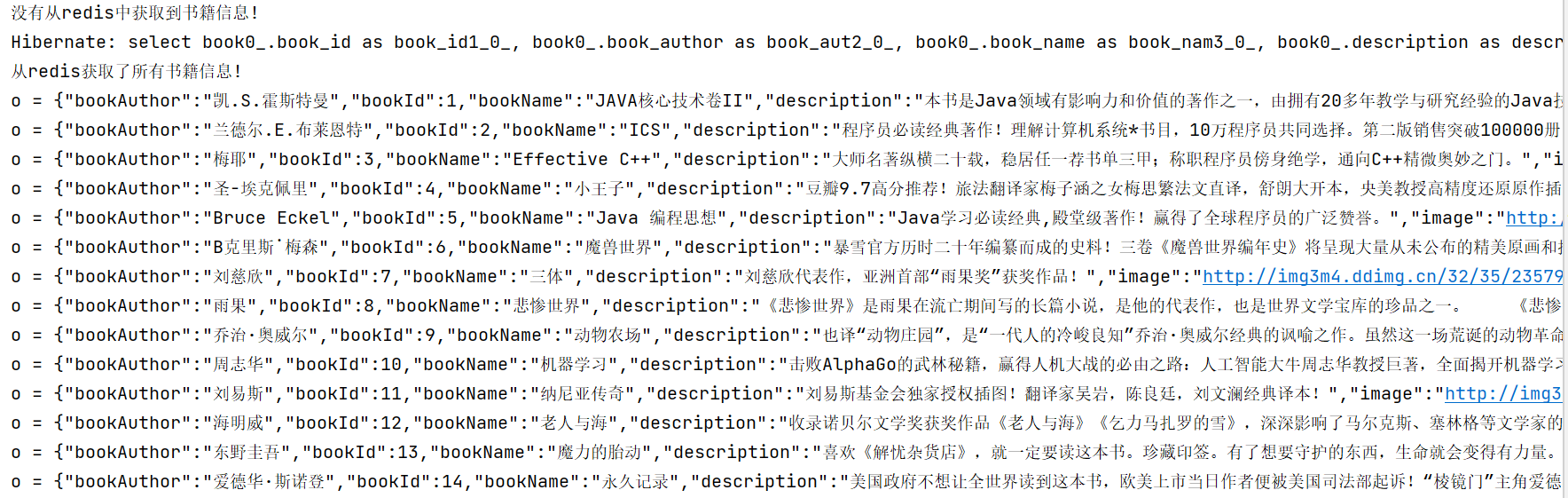
在本次作业中，我主要对书**籍信息的相关操作**使用Redis进行缓存。

1. **获取包含全部书籍信息的书籍列表**：

代码实现如下：



截图如下：

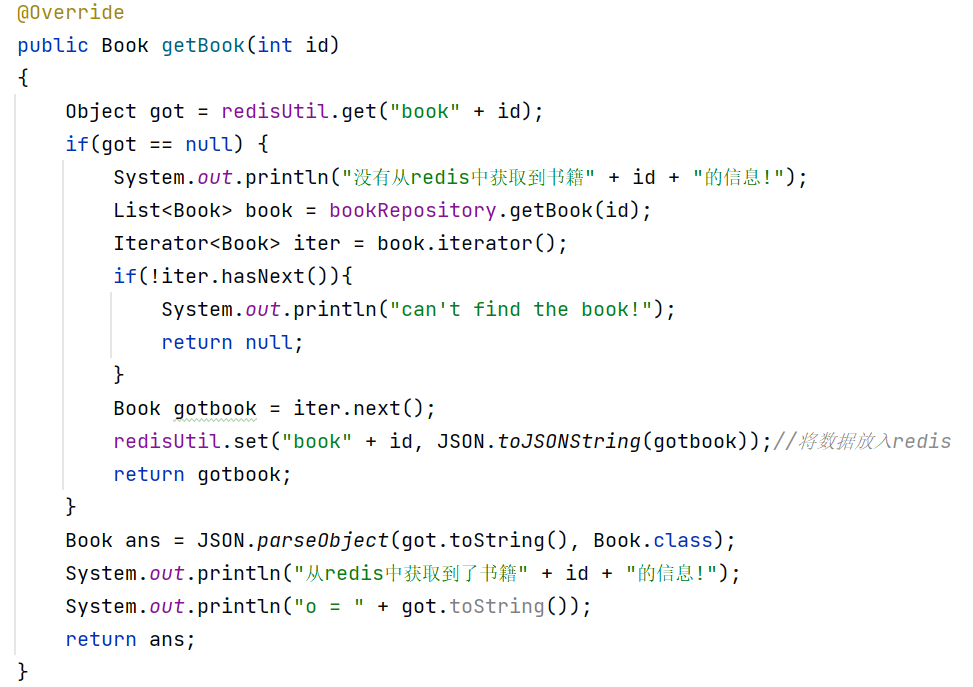


输出LOG解释如下：

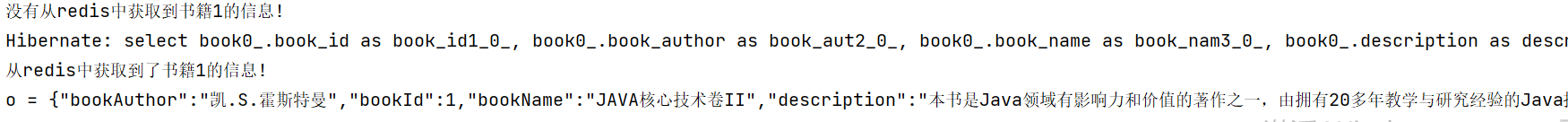
第一行打印出来的信息指的是在初次登录时，用户在获取BooksList时，相关数据没有存储在Redis中，所以显示redis中无法获取所有课程信息。之后，在首次获取之后，会将从数据库中拉去的数据放入redis。这时候刷新页面就可以直接从redis中获取到之前存入缓存中的信息。最下面的一部分是打印出来了从redis中获取的每一个list元素(即每一本书)的信息。

1. **获取单独一本书的信息**：

代码实现如下：



输出截图如下：



输出解释如下：

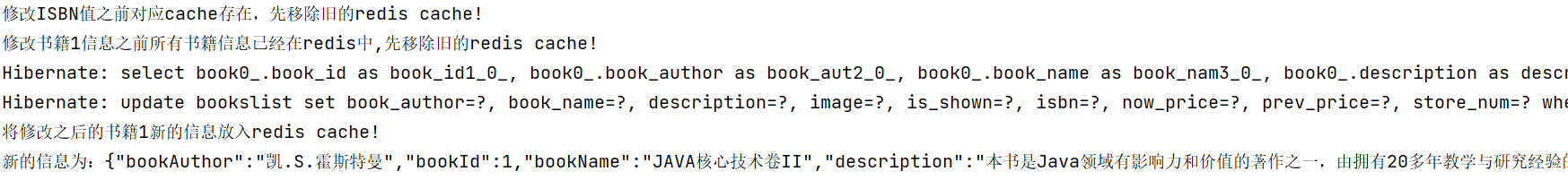
在获取一本书的时候(即点击一本书进入其详情页面的时候)，第一次进行读取的时候并不会存储在redis中(在这次的实现中，我采用的是将书籍列表与单独的书籍分开存储的方式，key的结构为”book” + id)，所以会显示不会从redis最后那个获取，并直接从数据库中获取，并将获取的内容存储在redis中，这时候刷新页面就可以直接从redis中获取对应数据了。最后一行log输出时打印出了从redis中获取的JSON格式的书籍id为id的书籍的信息。

1. **修改书籍信息**：

代码实现如下:(由于对于价格，ISBN,书名，作者等的修改操作基本相同，所以在这里只展示修改ISBN的部分)



输出如下：



输出解释如下：

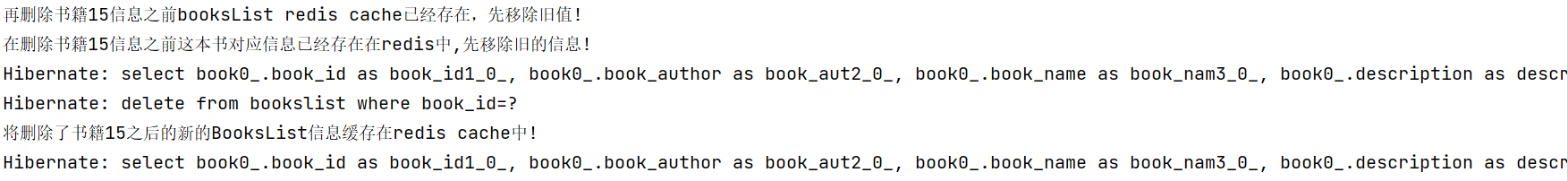
前两行是在检查在进行修改时是否有已经存储在cache中的数据会受到影响。因为是修改单独一本书，所以需要先检查单独一本书的cache，之后也需要移除BooksList的cache(如果有的话)。之后的两行就是关于修改数据库中信息的操作。最后一部分就是将新的书籍信息写入redis缓存中时打印的输出，并在最后一行打印了新的书籍信息。由于之后在进行后续修改时都需要进行cache的替换，操作逻辑相同，所以输出相同。

1. **删除一本已经存在的书籍**：

代码实现如下：



输出截图如下：



输出解释如下：

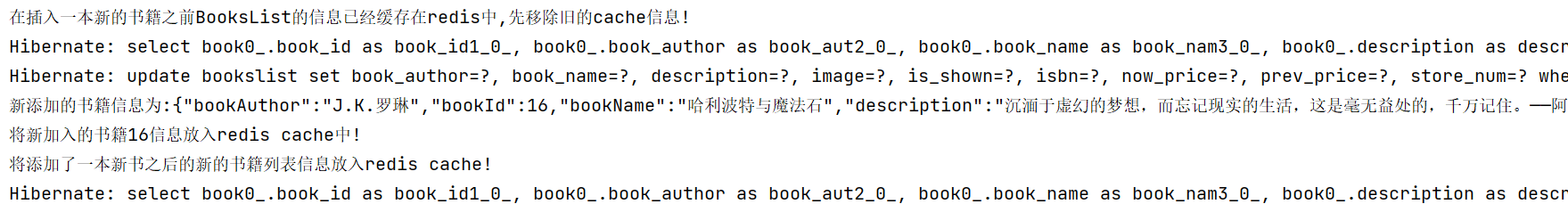
第一行是在检查进行删除之前是否已经将BooksList信息cache在redis中了。因为删除数据之后，会使得booksList发生变化，所以需要先将其存储在cache中的旧信息移除。之后两行是进行删除书籍并保存修改的数据库操作的输出信息。之后输出的内容显示的是将删除书籍之后的新的BooksList信息缓存到redis的cache中去。由于后续进行删除操作时都需要进行cache的替换，操作逻辑相同，所以输出基本相同。

1. **添加一本之前没有出现过的书籍**：

代码实现如下：



输出截图如下：

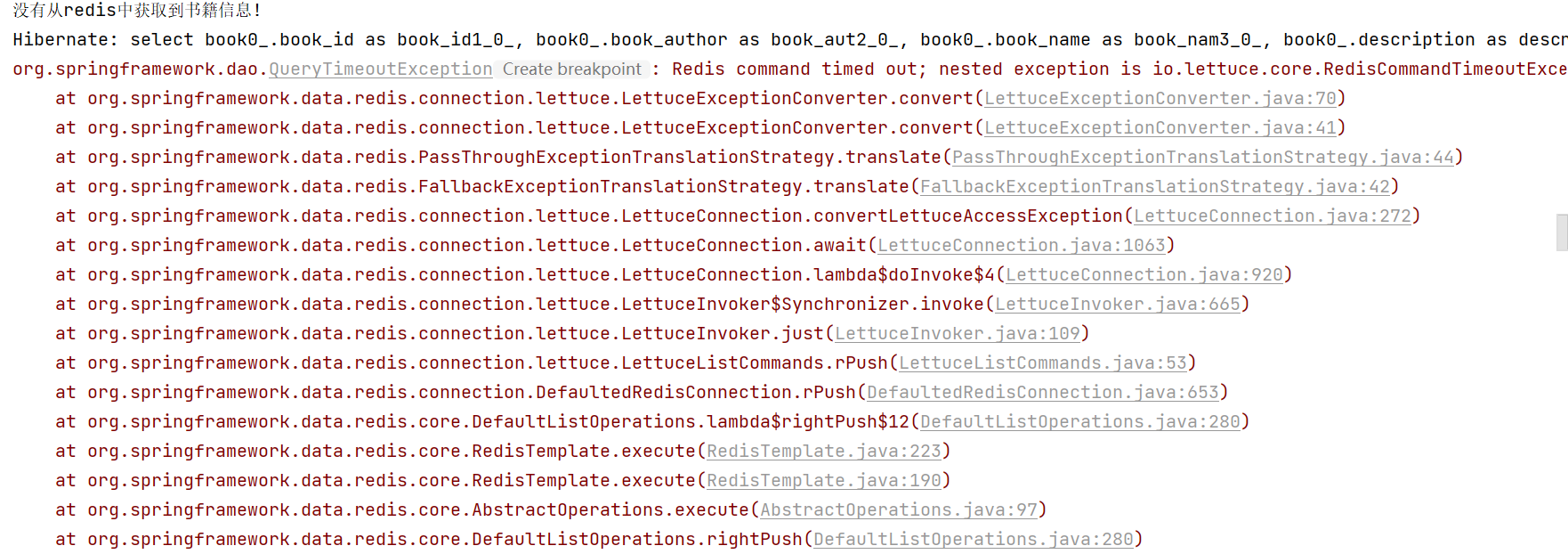


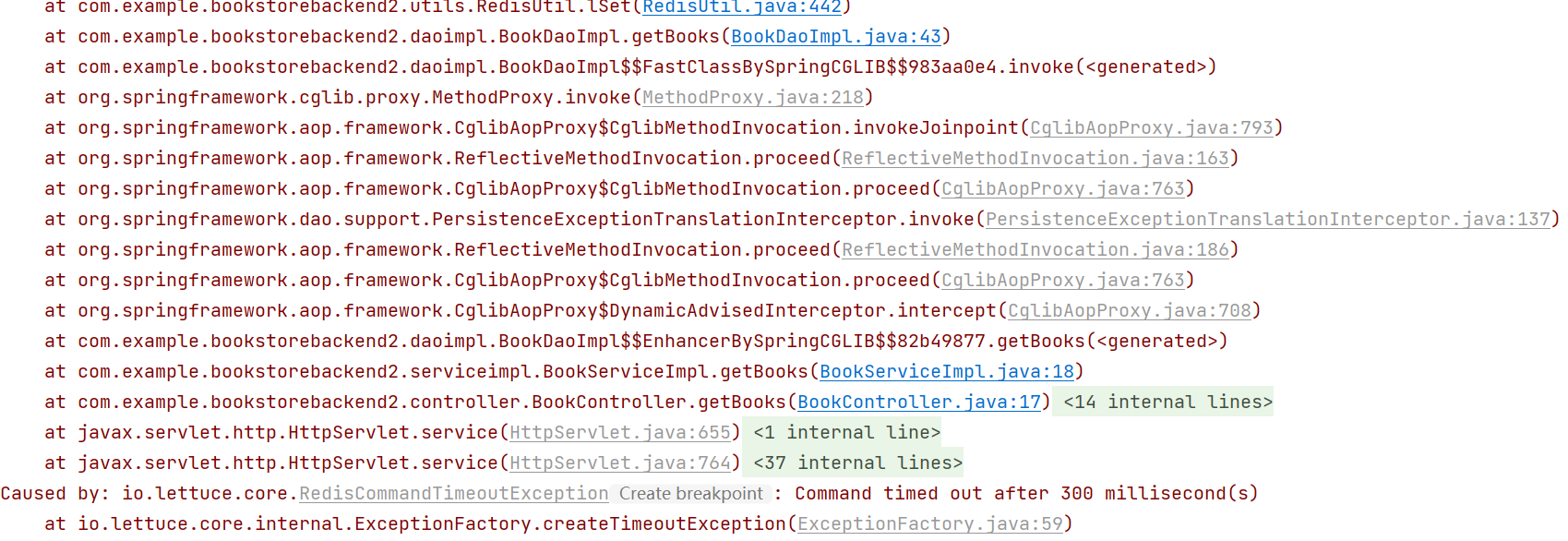
输出解释如下：

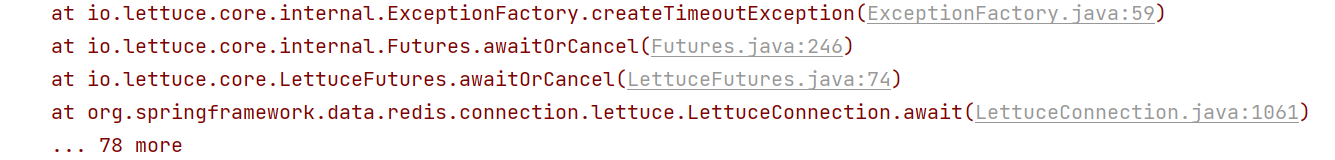
第一行是在检查加入新书籍之前BooksList数据是否已经缓存到redis中。因为涉及到对于这个BooksList的修改，所以需要首先移除可能存在的旧的cache数据。之后两行为向数据库中添加新书的操作的输出LOG。之后打印的内容为新添加的书籍信息。并在最后需要将新生成的BooksList数据信息再次缓存到redis中去。由于之后的添加新书的操作都需要进行redis cache的替换，所以逻辑相同，所以输出基本相同。

1. 手动关闭redis时候的现象：

输出截图如下：







输出解释如下：

从上图可以看出，由于在中途强制关闭了redis服务，所以在执行增删改查等逻辑时，由于无法连接redis服务器，所以会报出connection timeout的error，所以会导致对于redis中向右key的检查，添加/移除cache的操作失败，所以会使得系统无法正确运行。