<https://www.jianshu.com/p/0a92b7c97c65>

**5.2 测试点：正常购买**

我们添加的账户余额测试数据够我们购买两件商品，我们先来购买一件商品验证下接口访问是否成功，通过如下命令访问下单接口：

~ curl -X POST http://localhost:8082/order?goodId=1&accountId=1&buyCount=1

下单成功.

通过我们访问/order下单接口，根据响应的内容我们确定商品已经购买成功。

通过查看order-service控制台内容：

2019-10-11 16:52:15.477 INFO 13142 --- [nio-8082-exec-4] i.seata.tm.api.DefaultGlobalTransaction : [10.180.98.83:8091:2024417333] commit status:Committed

2019-10-11 16:52:16.412 INFO 13142 --- [atch\_RMROLE\_2\_8] i.s.core.rpc.netty.RmMessageListener : onMessage:xid=10.180.98.83:8091:2024417333,branchId=2024417341,branchType=AT,resourceId=jdbc:mysql://localhost:3306/test,applicationData=null

2019-10-11 16:52:16.412 INFO 13142 --- [atch\_RMROLE\_2\_8] io.seata.rm.AbstractRMHandler : Branch committing: 10.180.98.83:8091:2024417333 2024417341 jdbc:mysql://localhost:3306/test null

2019-10-11 16:52:16.412 INFO 13142 --- [atch\_RMROLE\_2\_8] io.seata.rm.AbstractRMHandler : Branch commit result: PhaseTwo\_Committed

我们可以看到本次事务已经成功Committed。

再去验证下数据库内的账户余额、商品库存是否有所扣减。

**5.3 测试点：库存不足**

测试商品添加了10个库存，在之前测试已经销售掉了一件商品，我们测试购买数量超过库存数量时，是否有回滚日志，执行如下命令：

~ curl -X POST http://localhost:8082/order?goodId=1&accountId=1&buyCount=10

{"timestamp":"2019-10-11T08:57:13.775+0000","status":500,"error":"Internal Server Error","message":"status 500 reading GoodClient#reduceStock(Integer,int)","path":"/order"}

在我们good-service服务控制台已经打印了商品库存不足的异常信息：

java.lang.RuntimeException: 商品：1库存不足.

at org.minbox.chapter.seata.service.GoodService.reduceStock(GoodService.java:42) ~[classes/:na]

....

我们再看order-service的控制台打印日志：

Begin new global transaction [10.180.98.83:8091:2024417350]

2019-10-11 16:57:13.771 INFO 13142 --- [nio-8082-exec-5] i.seata.tm.api.DefaultGlobalTransaction : [10.180.98.83:8091:2024417350] rollback status:Rollbacked

通过日志可以查看本次事务进行了回滚。

由于**库存的验证在账户余额扣减之前**，所以我们本次并不能从数据库的数据来判断事务是真的回滚。

**5.4 测试点：余额不足**

既然商品库存不足我们不能直接验证数据库事务回滚，我们从账户余额不足来下手，在之前成功购买了一件商品，账户的余额还够购买一件商品，商品库存目前是9件，我们本次测试购买5件商品，这样就会出现购买商品库存充足而余额不足的应用场景，执行如下命令发起请求：

~ curl -X POST http://localhost:8082/order\?goodId\=1\&accountId\=1\&buyCount\=5

{"timestamp":"2019-10-11T09:03:00.794+0000","status":500,"error":"Internal Server Error","message":"status 500 reading AccountClient#deduction(Integer,Double)","path":"/order"}

我们通过查看account-service控制台日志可以看到：

java.lang.RuntimeException: 账户：1，余额不足.

at org.minbox.chapter.seata.service.AccountService.deduction(AccountService.java:33) ~[classes/:na]

已经抛出了余额不足的异常。

通过查看good-service、order-serivce控制台日志，可以看到事务进行了回滚操作。

接下来查看seata\_account表数据，我们发现账户余额没有改变，账户服务的事务回滚**验证成功**。

查看seata\_good表数据，我们发现商品的库存也没有改变，商品服务的事务回滚**验证成功**。

**6. 总结**

本章主要来验证分布式事务框架Seata在MySQL下提交与回滚有效性，是否能够完成我们预期的效果，Seata作为SpringCloud Alibaba的核心框架，更新频率比较高，快速的解决使用过程中遇到的问题，是一个潜力股，不错的选择。

作者：恒宇少年  
链接：https://www.jianshu.com/p/0a92b7c97c65  
来源：简书  
著作权归作者所有。商业转载请联系作者获得授权，非商业转载请注明出处。