《分布式旅游预定系统》测试用例及说明

2020 年 8 月 10 日

目　录

《分布式旅游预定系统》测试用例及说明

1 运行环境

操作系统：Ubuntu 18.04

编译环境：openjdk 11.0.7

2 运行说明

首先进入 src/transaction 目录下依次启动 rmiregistry, TM, RMFlights, RMRooms, RMCars, RMCustomers, RMReservations 以及 WC。然后，进入 src/test 目录下启动测试程序。关于启动项目及测试程序的详细说明请参见 README。

3 测试用例说明

3.1 正常情况

此部分测试整个系统在无故障情况下的业务逻辑是否正确，测试用例涵盖了 Workflow Controller 提供的所有业务接口。在每个测试结束后会调用 Abort，以确保测试数据不会保留在系统之中。

3.1.1 测试用例 1：航班新增和航班查询

a) 用例说明
   1. 查询不存在的航班
   2. 以非法参数新增航班
   3. 以合法参数新增航班
   4. 查询 3 中所新增的航班
   5. 用 3 中的航班号再次添加航班
   6. 再次查询 3 中所添加的航班

b) 预期结果

步骤 1 返回结果-1；步骤 2 返回 False；步骤 3 返回 True；步骤 4 能正确查询到步骤 3 新增的航班；步骤 5 返回 True；步骤 6 查询到的结果为步骤 3 与步骤 5 的结合。

3.1.2 测试用例 2：查询航班价格

a) 用例说明
   1. 查询不存在的航班价格
   2. 新增航班

3. 查询 2 中新增的航班的价格

b) 预期结果

步骤 1 返回-1；步骤 3 能正确查询到步骤 2 新增的航班的价格。

3.1.3 测试用例 3：删除航班

a) 用例说明

1. 删除不存在的航班
2. 新增航班
3. 删除 2 中新增的航班
4. 查询 2 中新增航班
5. 新增航班并预订该航班
6. 删除 4 中新增的航班

b) 预期结果

步骤 1 返回 False；步骤 3 返回 True；步骤 4 返回-1；步骤 6 返回 False。

3.1.4 测试用例 4：预订航班

a) 用例说明

1. 新增顾客
2. 预订不存在的航班
3. 新增航班
4. 预订 3 中新增的航班

b) 预期结果

步骤 2 返回 False；步骤 4 返回 True。

3.1.5 测试用例 5：添加航班异常

a) 用例说明

使用无效事务 ID 添加航班。

b) 预期结果

抛出 InvalidTransactionException。

3.1.6 测试用例 6：删除航班异常

a) 用例说明

使用无效事务 ID 删除航班。

b) 预期结果
抛出 InvalidTransactionException。

3.1.7 测试用例 7：查询航班异常

a) 用例说明
使用无效事务 ID 查询航班。

b) 预期结果
抛出 InvalidTransactionException。

3.1.8 测试用例 8：查询航班价格异常

a) 用例说明
使用无效事务 ID 查询航班价格。

b) 预期结果
抛出 InvalidTransactionException。

3.1.9 测试用例 9：预订航班异常

a) 用例说明
使用无效事务 ID 预订航班。

b) 预期结果
抛出 InvalidTransactionException。

3.1.10 测试用例 10：房间新增和房间查询

a) 用例说明
1. 查询不存在的旅馆
2. 以非法参数新增旅馆
3. 以合法参数新增旅馆
4. 查询 3 中所新增的旅馆
5. 用 3 中的地点再次添加房间
6. 再次查询 3 中所添加的旅馆

b) 预期结果
步骤 1 返回结果-1；步骤 2 返回 False；步骤 3 返回 True；步骤 4 能正确查询到步骤 3 新增的旅馆；步骤 5 返回 True；步骤 6 查询到的结果为步骤 3 与步骤 5 的结合。

3.1.11 测试用例 11：查询房间价格

a) 用例说明
   1. 查询不存在的旅馆价格
   2. 新增旅馆
   3. 查询 2 中新增的旅馆的价格

b) 预期结果
   步骤 1 返回-1；步骤 3 能正确查询到步骤 2 中新增的旅馆的价格。

3.1.12 测试用例 12：删除房间

a) 用例说明
   1. 删除不存在的房间
   2. 新增旅馆（房间数为 n）
   3. 删除 2 中新增的旅馆的 n+1 个房间
   4. 查询 2 中新增的旅馆
   5. 删除 2 中新增的旅馆的 n-1 个房间
   6. 查询 2 中新增的旅馆

b) 预期结果
   步骤 1 返回 False；步骤 3 返回 False；步骤 4 查询到的房间数为 n；步骤 5 返回 True；步骤 6 查询到的房间数目为 1。

3.1.13 测试用例 13：预订房间

a) 测试数据
   1. 新增顾客
   2. 预订不存在的旅馆
   3. 新增旅馆（房间数为 0）
   4. 预订 3 中新增的旅馆
   5. 为 3 中新增的旅馆增加 100 个房间
   6. 再次预订 3 中新增的旅馆
   7. 查询 3 中新增的旅馆

b) 预期结果
   步骤 2 返回 False；步骤 4 返回 False；步骤 6 返回 True；步骤 7 查到的房间数为 99。

3.1.14 测试用例 14：添加房间异常

a) 用例说明
使用无效事务 ID 添加房间。

b) 预期结果
抛出 InvalidTransactionException。

3.1.15 测试用例 15：删除房间异常

a) 用例说明
使用无效事务 ID 删除房间。

b) 预期结果
抛出 InvalidTransactionException。

3.1.16 测试用例 16：查询房间异常

a) 用例说明
使用无效事务 ID 查询房间。

b) 预期结果
抛出 InvalidTransactionException。

3.1.17 测试用例 17：查询房间价格异常

a) 用例说明
使用无效事务 ID 查询房间价格。

b) 预期结果
抛出 InvalidTransactionException。

3.1.18 测试用例 18：预订房间异常

a) 用例说明
使用无效事务 ID 预订房间。

b) 预期结果
抛出 InvalidTransactionException。

3.1.19 测试用例 19：车辆新增和车辆查询

a) 用例说明

1. 查询不存在的租车行
2. 以非法参数新增租车行
3. 以合法参数新增租车行
4. 查询 3 中所新增的租车行
5. 用 3 中的地点再次添加车辆
6. 再次查询 3 中所添加的租车行

b) 预期结果

步骤 1 返回结果-1；步骤 2 返回 False；步骤 3 返回 True；步骤 4 能正确查询到步骤 3 新增的租车行；步骤 5 返回 True；步骤 6 查询到的结果为步骤 3 与步骤 5 的结合。

3.1.20 测试用例 20：查询车辆价格

a) 用例说明
   1. 查询不存在的租车行价格
   2. 新增租车行
   3. 查询 2 中新增的租车行的价格

b) 预期结果

步骤 1 返回-1；步骤 3 能正确查询到步骤 2 中新增的租车行的价格。

3.1.21 测试用例 21：删除车辆

a) 用例说明
   1. 删除不存在的车辆
   2. 新增租车行（车辆数为 n）
   3. 删除 2 中新增的租车行的 n+1 个车辆
   4. 查询 2 中新增的租车行
   5. 删除 2 中新增的租车行的 n-1 个车辆
   6. 查询 2 中新增的租车行

b) 预期结果

步骤 1 返回 False；步骤 3 返回 False；步骤 4 查询到的车辆数为 n；步骤 5 返回 True；步骤 6 查询到的车辆数为 1。

3.1.22 测试用例 22：预订车辆

a) 测试数据
   1. 新增顾客

2. 预订不存在的车辆
3. 新增租车行（车辆数为 0）
4. 预订 3 中新增的租车行的车辆
5. 为 3 中新增的租车行增加 100 辆车
6. 再次预订 3 中新增的租车行的车辆
7. 查询 3 中新增的租车行

b) 预期结果

步骤 2 返回 False；步骤 4 返回 False；步骤 6 返回 True；步骤 7 查到的车辆数为 99。

3.1.23 测试用例 23：车辆新增异常

a) 用例说明
使用无效事务 ID 新增车辆。

b) 预期结果
抛出 InvalidTransactionException。

3.1.24 测试用例 24：删除车辆异常

a) 用例说明
使用无效事务 ID 删除车辆。

b) 预期结果
抛出 InvalidTransactionException。

3.1.25 测试用例 25：查询车辆异常

a) 用例说明
使用无效事务 ID 查询车辆。

b) 预期结果
抛出 InvalidTransactionException。

3.1.26 测试用例 26：查询车辆价格异常

a) 用例说明
使用无效事务 ID 查询车辆价格。

b) 预期结果
抛出 InvalidTransactionException。

3.1.27 测试用例 27：预订车辆异常

a) 用例说明
使用无效事务 ID 预订车辆。

b) 预期结果
抛出 InvalidTransactionException。

3.1.28 测试用例 28：顾客新增

a) 用例说明
1. 使用非法顾客名新增顾客
2. 使用合法顾客名新增顾客
3. 再次使用 2 中的顾客名新增顾客

b) 预期结果
步骤 1 返回 False；步骤 2 返回 True；步骤 3 返回 True。

3.1.29 测试用例 29：顾客删除

a) 用例说明
1. 删除不存在的顾客
2. 新增顾客
3. 删除 2 中新增的顾客
4. 新增顾客，新增航班（座位数 100），新增旅馆（房间数 100）
5. 使用 4 中新增的顾客预订 4 中新增的航班与旅馆
6. 查询 4 中新增的航班与旅馆
7. 删除 4 中新增的顾客
8. 查询 4 中新增的航班与旅馆

b) 预期结果
步骤 1 返回 False；步骤 3 返回 True；步骤 6 查询到的航班座位数为 99，旅馆的房间数为 99；步骤 7 返回 True；步骤 8 查询到的航班座位数为 100，旅馆的房间数为 100。

3.1.30 测试用例 30：查询顾客账单

a) 用例说明
1. 新增顾客，新增航班（价格 100），新增旅馆（价格 100），新增租车行（价

格 100）

2. 使用 1 中新增的顾客预订 1 中新增的航班
3. 查询 1 中新增的顾客的账单
4. 使用 1 中新增的顾客预订 1 中新增的旅馆
5. 查询 1 中新增的顾客的账单
6. 使用 1 中新增的顾客预订 1 中新增的租车行
7. 查询 1 中新增的顾客的账单

b) 测试结果
步骤 3 返回 100；步骤 5 返回 200；步骤 7 返回 300。

3.1.31 测试用例 31：新增顾客异常

a) 用例说明
使用无效事务 ID 新增。

b) 预期结果
抛出 InvalidTransactionException。

3.1.32 测试用例 32：删除顾客异常

a) 用例说明
使用无效事务 ID 删除顾客。

b) 预期结果
抛出 InvalidTransactionException。

3.1.33 测试用例 33：查询顾客账单异常

a) 用例说明
使用无效事务 ID 查询顾客账单。

b) 预期结果
抛出 InvalidTransactionException。

3.2 资源管理器故障测试

此部分测试在各种资源管理器故障情况下系统中所有事务的 ACID 特性能否得到保证。

3.2.1 测试用例 1：资源管理器在 enlist 后出现故障

a) 用例说明
   1. 新增航班
   2. 设置 RMRooms 的 dieTime 为 AfterEnlist
   3. 新增房间
   4. 待 RMRooms 恢复后查询 1 中新增的航班
   5. 待 RMRooms 恢复后查询 3 中新增的房间

b) 预期结果
   整个事务被 abort，即步骤 4 返回-1，步骤 5 也返回-1。

3.2.2 测试用例 2：资源管理器在 prepare 前出现故障

a) 用例说明
   1. 新增航班
   2. 新增租车行
   3. 设置 RMCars 的 dieTime 为 BeforePrepare
   4. 提交事务
   5. 待 RMCars 恢复后查询 1 中新增的航班
   6. 待 RMCars 恢复后查询 2 中新增的租车行

b) 预期结果
   整个事务被 abort，即步骤 5 返回-1，步骤 6 也返回-1。

3.2.3 测试用例 3：资源管理器在 prepare 后出现故障

a) 用例说明
   1. 新增航班
   2. 新增旅馆
   3. 设置 RMFlights 的 dieTime 为 AfterPrepare
   4. 提交事务
   5. 待 RMFlights 恢复后查询 1 中新增的航班
   6. 待 RMFlights 恢复后查询 2 中新增的旅馆

b) 预期结果
   整个事务被 abort，即步骤 5 返回-1，步骤 6 也返回-1。

3.2.4 测试用例 4：资源管理器在 commit 前出现故障

a) 用例说明
   1. 新增航班（座位数 100，价格 999）

2. 新增顾客并预订 1 中的航班
3. 将 RMReservations 的 dieTime 设为 BeforeCommit
4. 提交事务
5. 待 RMReservations 恢复后查询 1 中新增的航班
6. 待 RMReservations 恢复后查询 2 中新增的顾客的账单

b) 预期结果
整个事务最终被 commit。即步骤 5 的结果 99，步骤 6 的结果为 999。

3.2.5 测试用例 5：资源管理器在 abort 前出现故障

a) 用例说明
1. 新增顾客
2. 将 RMCustomers 的 dieTIme 设为 BeforeAbort
3. 终止事务
4. 待 RMCustomers 恢复后查询 1 中新增顾客的账单

b) 预期结果
整个事务被 abort，即步骤 4 的结果为-1。

3.3 事务管理器故障测试

此部分测试在各种事务管理器故障情况下系统中所有事务的 ACID 特性能否得到保证。

3.3.1 测试用例 1：事务管理器在 commit 前出现故障

a) 用例说明
1. 新增顾客，新增航班，新增旅馆，新增租车行
2. 用 1 中的顾客预订 1 中的航班，旅馆以及租车行
3. 将 TM 的 dieTime 设为 BeforeCommit
4. 提交事务
5. 待 TM 恢复后查询 1 中顾客的账单
6. 待 TM 恢复后查询 1 中的航班，旅馆以及租车行

b) 预期结果
整个事务被 abort，即步骤 5 返回-1，步骤 6 中的所有查询也均返回-1。

3.3.2 测试用例 2：事务管理器在 commit 后出现故障

a) 用例说明

1. 新增顾客，新增航班，新增旅馆，新增租车行
2. 用 1 中的顾客预订 1 中的航班，旅馆以及租车行
3. 将 TM 的 dieTime 设为 AfterCommit
4. 提交事务
5. 待 TM 恢复后查询 1 中顾客的账单
6. 待 TM 恢复后查询 1 中的航班，旅馆以及租车行

b) 预期结果

整个事务最终被 commit，即步骤 5 正确返回顾客账单，且步骤 6 中的所有查询均能查询到 1 中新增的结果。

3.4 系统故障测试

此部分测试当整个系统发生故障时，系统重新启动后所有事务的 ACID 特性能否得到保证。

3.4.1 测试用例 1：系统所有组件在某一时刻全部故障

a) 用例说明

1. 新建事务 A 并提交。
2. 新建事务 B 并终止。
3. 新建事务 C 并执行一部分操作
4. 杀死系统中的所有组件
5. 待系统恢复后查询事务 A，B，C

b) 预期结果

系统中所有事务的 ACID 特性得到保证，即事务 A 的所有操作结果可以被查询到，而事务 B，C 的操作结果全部都已经从系统中抹去（abort）。

4 测试结果

4.1 正常情况测试结果

3.1.1-3.1.33 测试用例的测试结果请见图 1。

```
xss@DESKTOP-SDCJDHM:/mnt/e/Java_Code/ddb_pj/src/test$ make run_normal_test
/usr/bin/javac -classpath ..:/usr/share/java/junit-4.12.jar NormalTest.java ReservationSystemTest.java
/usr/bin/java -classpath ..:/usr/share/java/junit-4.12.jar:/usr/share/java/hamcrest-core-1.3.jar -DrmiPort=3345 org.junit.runner.JUnitCore test.No
rmalTest
JUnit version 4.12
.................................
Time: 2.173

OK (33 tests)
```

图 1 正常情况测试结果

4.2 资源管理器故障测试结果

3.2.1-3.2.5 测试用例的测试结果请见图 2。

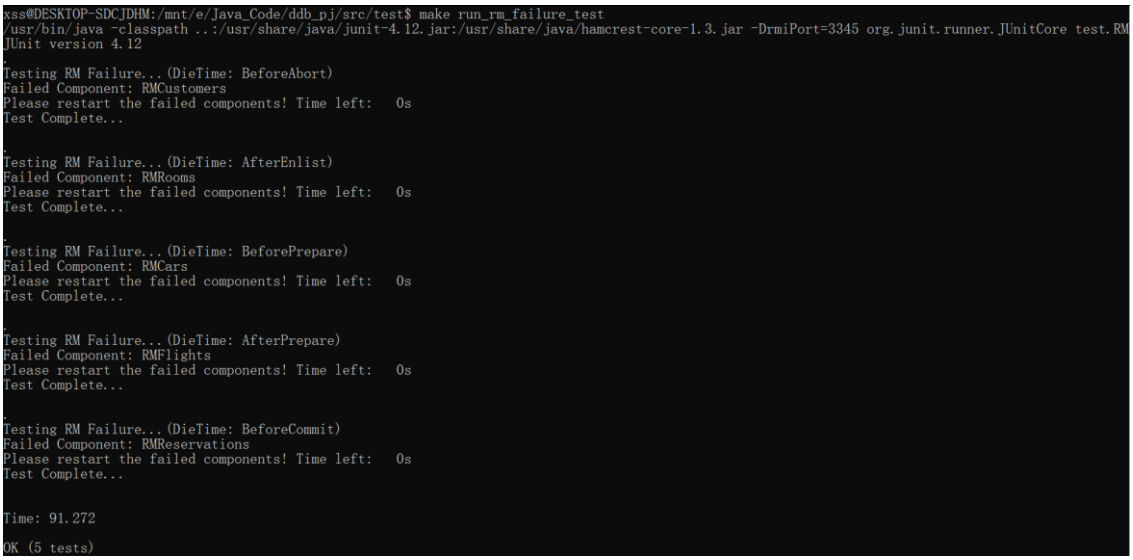

图 2　资源管理器故障测试结果

4.3　事务管理器故障测试结果

3.3.1-3.3.2 测试用例的测试结果请见图 3。

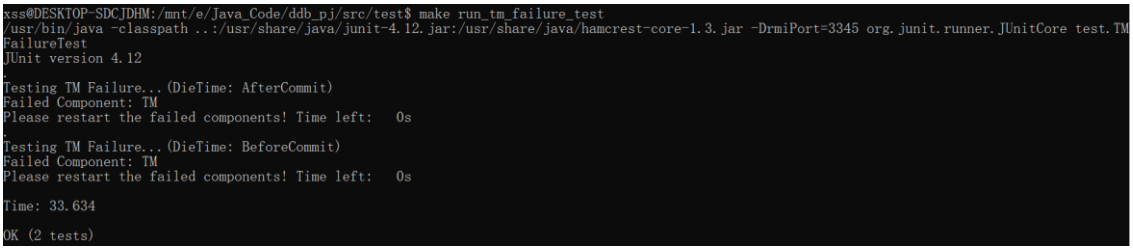

图 3　事务管理器故障测试结果

4.4　系统故障测试结果

3.4.1 测试用例的测试结果请见图 4。

图 4　系统故障测试结果