《数据库原理》大作业设计文档

常珂华 2019010798 刘臣洋 2019011250 杨健辉 2019011248 张博闻 2019013267

一、完成情况

小组按照《大作业说明》完成了所有基础要求，并实现了两个进阶项：where条件支持逻辑运算符（and/or），多事务并发的恢复。

二、实现方法

1. 查询模块

CREATE TABLE

在 `ImpVisitor` 中，根据生成的语法分析树设置列属性和约束。其中，约束需要分别处理紧接在属性定义之后的约束和在语句末尾用 `Primary Key(AttrName)` 定义的约束。

该语句要求且只能指定一个属性作为主键。主键属性将被设置为 `not null`。之后，调用当前 `Database` 创建数据表。

DROP TABLE

获取当前表的互斥锁，再删除表 `Database.drop()` 以删除数据表。

SHOW TABLE

通过表名访问该表，根据指定格式输出该表的名称、各属性的名称和类别。

INSERT INTO

分有无声明属性列处理。

对未声明属性列的情况，检查输入数据个数与表属性数是否相等。若相等，则插入数据表。若违反一致性，则报错。

对声明属性列的情况，检查三个指标：输入数据个数与声明属性数是否相同，声明属性是否包括主键，声明属性是否均为表中属性。若通过检查，则插入数据表。若违反一致性，则报错。

UPDATE

首先获取目标行的集合S。若语句不包含 `where`，则S为所有行的集合。若语句包含 `where`，则根据比较式筛选数据，得到S。

具体的筛选方法是：每次使用一个比较式筛选，依次检查所有行，运算结果根据逻辑连接词取交或并集。

然后将S的每一行执行更新操作。若违反一致性，则报错。

DELETE

首先获取目标行的集合S。若语句不包含 `where`，则S为所有行的集合。若语句包含 `where`，则根据比较式筛选数据，得到S。筛选方式与UPDATE的方法一致。

然后删除S里的每一行。

SELECT

选择语句处理流程为：构建查询工具表，进行条件筛选，进行投影。

首先依据代码框架中所给出工具类 `QueryResult` 完善其构造函数。由于 `select` 最终返回列的名称与所对应的元组，因此将 `Columns` 和 `ArrayList<Row>` 作为其构造函数的参数。代码框架中 `QueryTable` 类为查询时的临时数据表，因此需要保存列的属性以及行的元组数据，将其作为类属性添加。`SELECT` 语句中构建Table为核心工作，涉及到取笛卡尔积等操作，因此其构造函数需要重载，分别对应一张表为参数的情况和两张表及连接条件为参数的情况。

从一张表中选择时，以该表为参数构建工具表；从笛卡尔积中选择时，先分别根据两表构建工具表，而后依据两工具表生成结果工具表；自然连接时与笛卡尔积操作相同，但需要去除重复名称的属性，若没有重复属性则需退化为笛卡尔积。

筛选时通过框架中的 `Cell` 类实现的 `compareTo` 方法实现，获得比较符后判断比较条件而后从迭代器中选择满足条件的元组加入新工具表。

投影时获取需要的属性的 `index` 并且从元组中选择对应 `index` 的数据加入新工具表。

2.事务并发与恢复模块

BEGIN TRANSACTION与COMMIT

维护一个队列 `currentSessions`，用于记录处于事务状态的 `Session`。对于非事务状态 `Session` 的语句S，将之转化为 `Begin Transaction; S; Commit;` 的语句序列再执行。对于事务状态 `Session` 的语句S，直接执行即可。此外，还需要在执行语句前记录下日志。

实现read committed隔离级别

采用读写锁机制：读锁之间相容，可以被多个对象获取；写锁与写锁、读锁互斥，只能被一个对象获取；未获取到目标锁时，进程阻塞一段时间。对每一个 `Table` 对象，维护两个数组：`sLockSessions` 和 `xLockSessions`，用以记录当前获取锁的 `Session`。从而实现读写锁机制。

当执行**DROP TABLE, UPDATE, INSERT INTO, DELETE**指令时，需要在修改数据前获取写锁。当执行**SELECT**指令时，需要在读取数据前获取读锁。当事务获取写锁后，只能在执行**COMMIT**指令时释放写锁，从而实现**read committed**隔离级别。

实现事务的WAL机制

实现了 `writeLog` 和 `readLog` 两个方法:

每当一个 `Session` 执行一条 `Statement` 时，我们调用 `writeLog` 将这条 `Statement` 与 `Session` 的编号一同写入到对应数据库的日志文件中；

每次重启 `server`，我们调用 `readLog` 方法，分别处理上次 `server` 运行过程中每一个 `Session` 的操作，具体方法是查找每个 `Session` 的最后一次 `commit`，并忽略它在此后的所有操作，然后按时间顺序处理所有 `Statement` 即可。

由于多个 `Session` 可能同时写入同一个日志文件，需要给每个数据库开设专门的读写锁 `ReentrantReadWriteLock`。
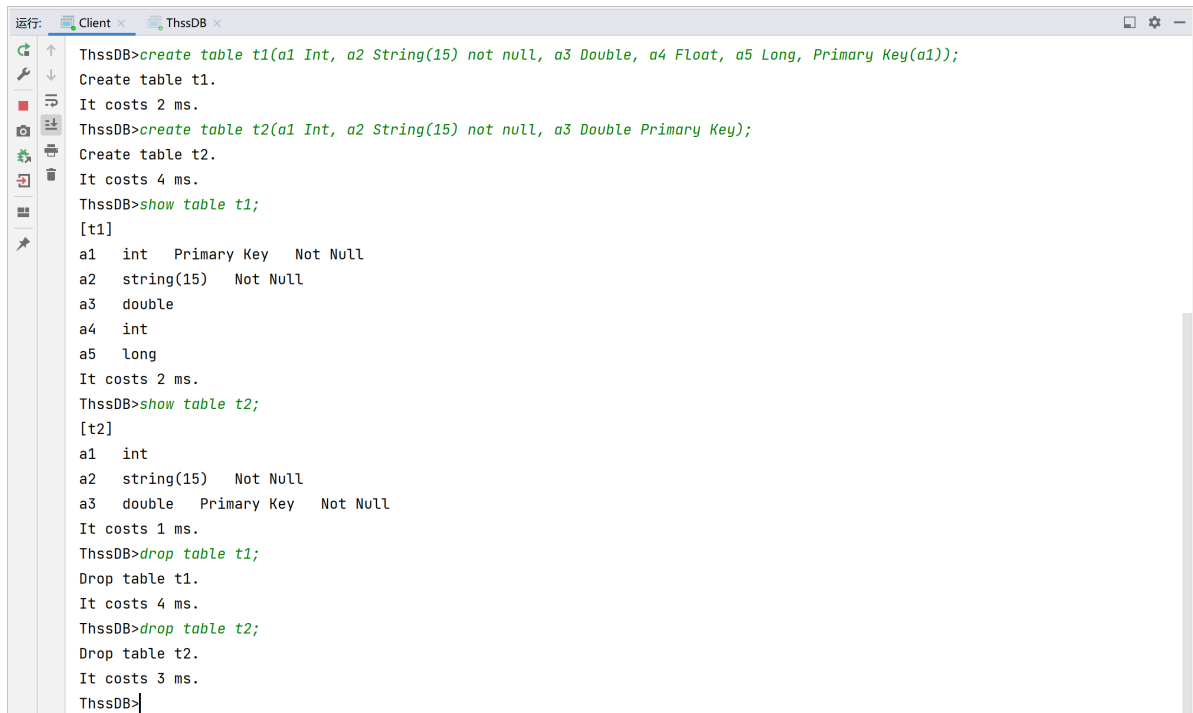
三、功能演示

1. 查询模块

数据表的创建、查询、删除

```
connect;

create table t1(a1 Int, a2 String(15) not null, a3 Double, a4 Float, a5 Long,
Primary Key(a1));
create table t2(a1 Int, a2 String(15) not null, a3 Double Primary Key);
show table t1;
show table t2;
drop table t1;
drop table t2;
```

```
运行:  Client ×    ThssDB ×
ThssDB>create table t1(a1 Int, a2 String(15) not null, a3 Double, a4 Float, a5 Long, Primary Key(a1));
Create table t1.
It costs 2 ms.
ThssDB>create table t2(a1 Int, a2 String(15) not null, a3 Double Primary Key);
Create table t2.
It costs 4 ms.
ThssDB>show table t1;
[t1]
a1    int    Primary Key   Not Null
a2    string(15)   Not Null
a3    double
a4    int
a5    long
It costs 2 ms.
ThssDB>show table t2;
[t2]
a1    int
a2    string(15)   Not Null
a3    double   Primary Key   Not Null
It costs 1 ms.
ThssDB>drop table t1;
Drop table t1.
It costs 4 ms.
ThssDB>drop table t2;
Drop table t2.
It costs 3 ms.
ThssDB>
```

数据的查询、插入、删除、修改

```
connect;

drop table t1;
drop table t2;
create table t1(id1 Int, name String(15) not null, Primary Key(id1));
create table t2(id2 Double Primary Key, name String(15) not null);

insert into t1 values (1,'Sichuan');
insert into t1 values (2,'Shanghai');
insert into t1(id1) values (3);#提示name不应为空
insert into t1(name, id1) values ('Shanghai',3);
insert into t1 values (4,'Beijing');
insert into t1 values (10,'Xinjiang');

insert into t2 values (1,'Sichuan');
insert into t2 values (2,'Shanghai');
insert into t2 values (3,'Beijing');
insert into t2 values (4,'Beijing');
insert into t2 values (5,'Tianjin');

select * from t1;
select * from t1,t2 where t1.id1 ≥2;
select * from t1,t2 where t1.name='Beijing' and t2.id2 >3;
select t1.name, t2.id2 from t1,t2 where t1.name='Beijing' or t2.id2 >3;
select * from t1 join t2 on t1.id1<t2.id2;
select * from t1 join t2 on t1.id1<3;
select * from t1 join t2 on t1.id1>t2.id2 where t2.name<>'Beijing';
```

```sql
select * from t1;
select * from t2;
update t1 set name='Guangdong' where id1>2 and name='Beijing';
select * from t1;
update t1 set name='Guizhou' where id1>2 or name='Guangdong';
select * from t1;
update t1 set name='Hebei' where id1=1;
select * from t1;

delete from t2 where name='Shanghai' or id2 <3;
select * from t2;
delete from t1 where name='Guangdong';
select * from t1;
delete from t1 where name='Guizhou' and id1 = 4;
select * from t1;
```

```
ThssDB>insert into t2 values (2,'Shanghai');
Insert into t2.
----------------------------------------
It costs 4 ms.
ThssDB>insert into t2 values (3,'Beijing');
Insert into t2.
----------------------------------------
It costs 3 ms.
ThssDB>insert into t2 values (4,'Beijing');
Insert into t2.
----------------------------------------
It costs 3 ms.
ThssDB>insert into t2 values (5,'Tianjin');
Insert into t2.
----------------------------------------
It costs 3 ms.
ThssDB>select * from t1;
t1.id1, t1.name
----------------------------------------
1, Sichuan
2, Shanghai
3, Shanghai
4, Beijing
10, Xinjiang
----------------------------------------
It costs 5 ms.
ThssDB>select * from t1,t2 where t1.id1 >=2;
t1.id1, t1.name, t2.id2
----------------------------------------
2, Shanghai, 2
3, Shanghai, 2
4, Beijing, 3
4, Beijing, 4
----------------------------------------
It costs 5 ms.
ThssDB>select * from t1,t2 where t1.name='Beijing' and t2.id2 >3;
t1.id1, t1.name, t2.id2
----------------------------------------
4, Beijing, 4
----------------------------------------
It costs 5 ms.
ThssDB>select t1.name, t2.id2 from t1,t2 where t1.name='Beijing' or t2.id2 >3;
t1.name, t2.id2
----------------------------------------
Beijing, 3
Beijing, 4
Beijing, 4
----------------------------------------
It costs 3 ms.
ThssDB>select * from t1 join t2 on t1.id1<t2.id2;
```

```
ThssDB>select * from t1 join t2 on t1.id1<t2.id2;
t1.id1, t1.name, t2.id2, t2.name
----------------------------------------
1, Sichuan, 2, Shanghai
1, Sichuan, 3, Beijing
1, Sichuan, 4, Beijing
1, Sichuan, 5, Tianjin
2, Shanghai, 3, Beijing
2, Shanghai, 4, Beijing
2, Shanghai, 5, Tianjin
3, Shanghai, 4, Beijing
3, Shanghai, 5, Tianjin
4, Beijing, 5, Tianjin
----------------------------------------
It costs 3 ms.
ThssDB>select * from t1 join t2 on t1.id1<3;
t1.id1, t1.name, t2.id2, t2.name
----------------------------------------
1, Sichuan, 1, Sichuan
1, Sichuan, 2, Shanghai
1, Sichuan, 3, Beijing
1, Sichuan, 4, Beijing
1, Sichuan, 5, Tianjin
2, Shanghai, 1, Sichuan
2, Shanghai, 2, Shanghai
2, Shanghai, 3, Beijing
2, Shanghai, 4, Beijing
2, Shanghai, 5, Tianjin
----------------------------------------
It costs 4 ms.
ThssDB>select * from t1 join t2 on t1.id1>t2.id2 where t2.name<>'Beijing';
t1.id1, t1.name, t2.id2, t2.name
----------------------------------------
2, Shanghai, 1, Sichuan
3, Shanghai, 1, Sichuan
3, Shanghai, 2, Shanghai
4, Beijing, 1, Sichuan
4, Beijing, 2, Shanghai
10, Xinjiang, 1, Sichuan
10, Xinjiang, 2, Shanghai
10, Xinjiang, 5, Tianjin
----------------------------------------
It costs 4 ms.
```

```
ThssDB>select * from t1;
t1.id1, t1.name
----------------------------------------
1, Sichuan
2, Shanghai
3, Guangdong
4, Beijing
10, Xinjiang
----------------------------------------
It costs 6 ms.
ThssDB>select * from t2;
t2.id2, t2.name
----------------------------------------
1, Sichuan
2, Shanghai
3, Beijing
4, Beijing
5, Tianjin
----------------------------------------
It costs 4 ms.
ThssDB>update t1 set name='Guangdong' where id1>2 and name='Beijing';
Update table t1.
----------------------------------------
It costs 2533 ms.
ThssDB>select * from t1;
t1.id1, t1.name
----------------------------------------
1, Sichuan
2, Shanghai
3, Guangdong
4, Guangdong
10, Xinjiang
----------------------------------------
It costs 3 ms.
ThssDB>update t1 set name='Guizhou' where id1>2 or name='Guangdong';
Update table t1.
----------------------------------------
It costs 6 ms.
ThssDB>select * from t1;
t1.id1, t1.name
----------------------------------------
1, Sichuan
2, Shanghai
3, Guizhou
4, Guizhou
10, Guizhou
----------------------------------------
```

```
It costs 3 ms.
ThssDB>update t1 set name='Hebei' where id1=1;
Update table t1.
----------------------------------------
It costs 4 ms.
ThssDB>select * from t1;
t1.id1, t1.name
----------------------------------------
1, Hebei
2, Shanghai
3, Guizhou
4, Guizhou
10, Guizhou
----------------------------------------
It costs 4 ms.
ThssDB>delete from t1 where name='Guangdong';
Delete from table t1.
----------------------------------------
It costs 4 ms.
ThssDB>select * from t1;
t1.id1, t1.name
----------------------------------------
1, Hebei
2, Shanghai
3, Guizhou
4, Guizhou
10, Guizhou
----------------------------------------
It costs 7 ms.
ThssDB>delete from t1 where name='Hebei';
Delete from table t1.
----------------------------------------
It costs 5 ms.
ThssDB>select * from t1;
t1.id1, t1.name
----------------------------------------
2, Shanghai
3, Guizhou
4, Guizhou
10, Guizhou
----------------------------------------
It costs 3 ms.
```

```
ThssDB>delete from t2 where name='Shanghai' or id2 <3;
Delete from table t2.
------------------------------------
It costs 30782 ms.
ThssDB>select * from t2;
t2.id2, t2.name
------------------------------------
3, Beijing
4, Beijing
5, Tianjin
------------------------------------
It costs 3 ms.
ThssDB>delete from t1 where name='Guangdong';
Delete from table t1.
------------------------------------
It costs 1742 ms.
ThssDB>select * from t1;
t1.id1, t1.name
------------------------------------
2, Shanghai
3, Guizhou
4, Guizhou
10, Guizhou
------------------------------------
It costs 4 ms.
ThssDB>delete from t1 where name='Guizhou' and id1 = 4;
Delete from table t1.
------------------------------------
It costs 1615 ms.
ThssDB>select * from t1;
t1.id1, t1.name
------------------------------------
2, Shanghai
3, Guizhou
10, Guizhou
------------------------------------
It costs 4 ms.
ThssDB>
```

2.事务并发与恢复模块

事务与恢复

```
#测试用例
connect;

drop table t1;
create table t1(id1 Int, name String(15) not null, Primary Key(id1));

begin transaction;
insert into t1 values (1,'Sichuan');
insert into t1 values (3,'Beijing');
insert into t1 values (4,'Beijing');
update t1 set name='Hubei' where id1=3;
delete from t1 where id1=4;
select * from t1;
commit;

begin transaction;
insert into t1 values (5,'Sichuan');
insert into t1 values (6,'Shanghai');
update t1 set name='Guangdong' where id1=1;
delete from t1 where id1=3;
select * from t1;

#关闭服务端和用户端
#重启服务端和用户端

connect;
select * from t1;
```

```
运行:  Client ×

       D:\Java\jdk1.8.0_321\bin\java.exe ...
       ----------------------
       Starting ThssDB Client
       ----------------------
       ThssDB>connect;
       ConnectResp(status:Status(code:0), sessionId:0)
       It costs 49 ms.
       ThssDB>drop table t1;
       Drop table t1.
       ---------------------------------------
       It costs 31 ms.
       ThssDB>create table t1(id1 Int, name String(15) not null, Primary Key(id1));
       Create table t1.
       ---------------------------------------
       It costs 10 ms.
       ThssDB>begin transaction;
       start transaction.
       ---------------------------------------
       It costs 3 ms.
       ThssDB>insert into t1 values (1,'Sichuan');
       Insert into t1.
       ---------------------------------------
       It costs 5 ms.
       ThssDB>insert into t1 values (3,'Beijing');
       Insert into t1.
       ---------------------------------------
       It costs 2 ms.
       ThssDB>insert into t1 values (4,'Beijing');
       Insert into t1.
       ---------------------------------------
       It costs 2 ms.
       ThssDB>update t1 set name='Hubei' where id1=3;
       Update table t1.
       ---------------------------------------
       It costs 4 ms.
       ThssDB>delete from t1 where id1=4;
       Delete from table t1.
       ---------------------------------------
       It costs 3 ms.


       ThssDB>select * from t1;
       t1.id1, t1.name
       ---------------------------------------
       1, Sichuan
       3, Hubei
       ---------------------------------------
       It costs 6 ms.
       ThssDB>commit;
       commit transaction.
       ---------------------------------------
       It costs 3 ms.
       ThssDB>begin transaction;
       start transaction.
       ---------------------------------------
       It costs 3 ms.
       ThssDB>insert into t1 values (5,'Sichuan');
       Insert into t1.
       ---------------------------------------
       It costs 4 ms.
       ThssDB>insert into t1 values (6,'Shanghai');
       Insert into t1.
       ---------------------------------------
       It costs 2 ms.
       ThssDB>update t1 set name='Guangdong' where id1=1;
       Update table t1.
       ---------------------------------------
       It costs 2 ms.
       ThssDB>delete from t1 where id1=3;
       Delete from table t1.
       ---------------------------------------
       It costs 3 ms.
       ThssDB>select * from t1;
       t1.id1, t1.name
       ---------------------------------------
       1, Guangdong
       5, Sichuan
       6, Shanghai
       ---------------------------------------
       It costs 4 ms.
```

重启服务端和用户端后：

```
运行:  Client ×

       D:\Java\jdk1.8.0_321\bin\java.exe ...
       ----------------------
       Starting ThssDB Client
       ----------------------
       ThssDB>connect;
       ConnectResp(status:Status(code:0), sessionId:0)
       It costs 51 ms.
       ThssDB>select * from t1;
       t1.id1, t1.name
       ---------------------------------------
       1, Sichuan
       3, Hubei
       ---------------------------------------
       It costs 36 ms.
       ThssDB>
```

实现read committed隔离级别

```
#测试用例
connect;#session 0
connect;#session 1

#以下代码块顺序代表执行顺序

###session0
drop table t1;
drop table t2;
create table t1(id1 Int, name String(15) not null, Primary Key(id1));
create table t2(id2 Double Primary Key, name String(15) not null);
insert into t2 values (5,'Tianjin');
begin transaction;
insert into t1 values (1,'Sichuan');
###

###session1
begin transaction;
select * from t2;#正常访问
select * from t1;#由于session0在访问, 阻塞
###

###session0
commit;#session1恢复执行
select * from t1;#正常访问
###

###session1
insert into t1 values (8,'Xinjiang');
###

###session0
select * from t1;#阻塞
###

###session1
commit;#session0恢复执行
###
```

```
运行:  ThssDB ×   Client ×                              Client ×
D:\Java\jdk1.8.0_321\bin\java.exe ...           D:\Java\jdk1.8.0_321\bin\java.exe ...
---------------------                           ---------------------
Starting ThssDB Client                          Starting ThssDB Client
---------------------                           ---------------------
ThssDB>connect;                                 ThssDB>connect;
ConnectResp(status:Status(code:0), sessionId:0) ConnectResp(status:Status(code:0), sessionId:1)
It costs 34 ms.                                 It costs 33 ms.
ThssDB>drop table t1;                           ThssDB>begin transaction;
Drop table t1.                                  start transaction.
--------------------------------------          --------------------------------------
It costs 41 ms.                                 It costs 14 ms.
ThssDB>drop table t2;                           ThssDB>select * from t2;
Drop table t2.                                  t2.id2, t2.name
--------------------------------------          --------------------------------------
It costs 4 ms.                                  5.0, Tianjin
ThssDB>create table t1(id1 Int, name String(15) not null, Primary Key(id1)); --------------------------------------
Create table t1.                                It costs 7 ms.
--------------------------------------          ThssDB>select * from t1;
It costs 8 ms.
ThssDB>create table t2(id2 Double Primary Key, name String(15) not null);
Create table t2.
--------------------------------------
It costs 9 ms.
ThssDB>insert into t2 values (5,'Tianjin');
Insert into t2.
--------------------------------------
It costs 4 ms.
ThssDB>begin transaction;
start transaction.
--------------------------------------
It costs 1 ms.
ThssDB>insert into t1 values (1,'Sichuan');
Insert into t1.
--------------------------------------
It costs 2 ms.
ThssDB>
```

Git   运行   TODO   终端   问题   构建   依赖项   调试

多事务并发与恢复

```
#测试用例
connect;#session 0
connect;#session 1

#以下代码块顺序代表执行顺序

###session0
drop table t1;
drop table t2;
create table t1(id1 Int, name String(15) not null, Primary Key(id1));
create table t2(id2 Double Primary Key, name String(15) not null);
begin transaction;
insert into t1 values (1,'Sichuan');
insert into t1 values (2,'Sichuan');
###

###session1
begin transaction;
insert into t2 values (1,'Hunan');
insert into t2 values (2,'Hunan');
insert into t2 values (3,'Hebei');
update t2 set name='Tianjin' where id2=3;
###

###session0
delete from t2 where id2=1;#阻塞
###

###session1
commit;
###

###session0
insert into t2 values (4,'Beijing');
commit;
select * from t1;
select * from t2;
###

#关闭服务端和用户端
#重启服务端和用户端

###session0
connect;
select * from t1;
select * from t2;
###
```

```
D:\Java\jdk1.8.0_321\bin\java.exe ...
----------------------
Starting ThssDB Client
----------------------
ThssDB>connect;
ConnectResp(status:Status(code:0), sessionId:1)
It costs 24 ms.
ThssDB>drop table t1;
Drop table t1.
----------------------------------------
It costs 19 ms.
ThssDB>drop table t2;
Drop table t2.
----------------------------------------
It costs 4 ms.
ThssDB>create table t1(id1 Int, name String(15) not null, Primary Key(id1));
Create table t1.
----------------------------------------
It costs 5 ms.
ThssDB>create table t2(id2 Double Primary Key, name String(15) not null);
Create table t2.
----------------------------------------
It costs 6 ms.
ThssDB>begin transaction;
start transaction.
----------------------------------------
It costs 2 ms.
ThssDB>insert into t1 values (1,'Sichuan');
Insert into t1.
----------------------------------------
It costs 3 ms.
ThssDB>insert into t1 values (2,'Sichuan');
Insert into t1.
----------------------------------------
It costs 2 ms.
ThssDB>delete from t2 where id2=1;
```

```
D:\Java\jdk1.8.0_321\bin\java.exe ...
----------------------
Starting ThssDB Client
----------------------
ThssDB>connect;
ConnectResp(status:Status(code:0), sessionId:2)
It costs 28 ms.
ThssDB>begin transaction;
start transaction.
----------------------------------------
It costs 16 ms.
ThssDB>insert into t2 values (1,'Hunan');
Insert into t2.
----------------------------------------
It costs 3 ms.
ThssDB>insert into t2 values (2,'Hunan');
Insert into t2.
----------------------------------------
It costs 3 ms.
ThssDB>insert into t2 values (3,'Hebei');
Insert into t2.
----------------------------------------
It costs 3 ms.
ThssDB>update t2 set name='Tianjin' where id2=3;
Update table t2.
----------------------------------------
It costs 5 ms.
ThssDB>
```

```
Create table t1.
----------------------------------------
It costs 5 ms.
ThssDB>create table t2(id2 Double Primary Key, name String(15) not null);
Create table t2.
----------------------------------------
It costs 6 ms.
ThssDB>begin transaction;
start transaction.
----------------------------------------
It costs 2 ms.
ThssDB>insert into t1 values (1,'Sichuan');
Insert into t1.
----------------------------------------
It costs 3 ms.
ThssDB>insert into t1 values (2,'Sichuan');
Insert into t1.
----------------------------------------
It costs 2 ms.
ThssDB>delete from t2 where id2=1;
Delete from table t2.
----------------------------------------
It costs 43143 ms.
ThssDB>insert into t2 values (4,'Beijing');
Insert into t2.
----------------------------------------
It costs 2 ms.
ThssDB>commit;
commit transaction.
----------------------------------------
It costs 2 ms.
ThssDB>select * from t1;
t1.id1, t1.name
----------------------------------------
1, Sichuan
2, Sichuan
----------------------------------------
It costs 6 ms.
ThssDB>select * from t2;
t2.id2, t2.name
----------------------------------------
2.0, Hunan
3.0, Tianjin
4.0, Beijing
----------------------------------------
It costs 4 ms.
ThssDB>
```

```
D:\Java\jdk1.8.0_321\bin\java.exe ...
----------------------
Starting ThssDB Client
----------------------
ThssDB>connect;
ConnectResp(status:Status(code:0), sessionId:2)
It costs 28 ms.
ThssDB>begin transaction;
start transaction.
----------------------------------------
It costs 16 ms.
ThssDB>insert into t2 values (1,'Hunan');
Insert into t2.
----------------------------------------
It costs 3 ms.
ThssDB>insert into t2 values (2,'Hunan');
Insert into t2.
----------------------------------------
It costs 3 ms.
ThssDB>insert into t2 values (3,'Hebei');
Insert into t2.
----------------------------------------
It costs 3 ms.
ThssDB>update t2 set name='Tianjin' where id2=3;
Update table t2.
----------------------------------------
It costs 5 ms.
ThssDB>commit;
commit transaction.
----------------------------------------
It costs 2 ms.
ThssDB>
```

重启后：

```
运行:    ThssDB ×    Client ×

    D:\Java\jdk1.8.0_321\bin\java.exe ...
    ----------------------
    Starting ThssDB Client
    ----------------------
    ThssDB>connect;
    ConnectResp(status:Status(code:0), sessionId:0)
    It costs 43 ms.
    ThssDB>select * from t1;
    t1.id1, t1.name
    ----------------------------------------
    1, Sichuan
    2, Sichuan
    ----------------------------------------
    It costs 36 ms.
    ThssDB>select * from t2;
    t2.id2, t2.name
    ----------------------------------------
    2.0, Hunan
    3.0, Tianjin
    4.0, Beijing
    ----------------------------------------
    It costs 4 ms.
    ThssDB>
```