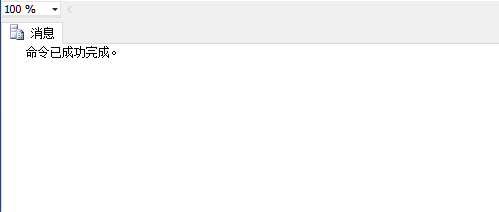
# 11.1

### 11.1.1 违反 check 约束的 update 操作

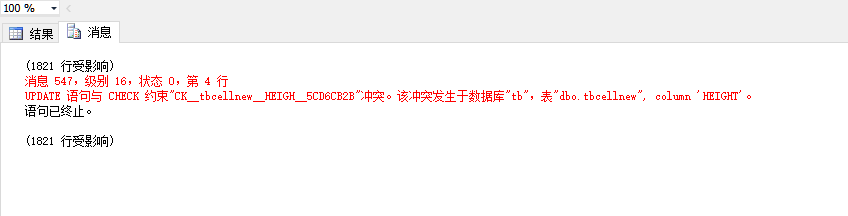
添加check约束

|  |
| --- |
| alter table tbcellnew add CHECK (HEIGHT >=0); |



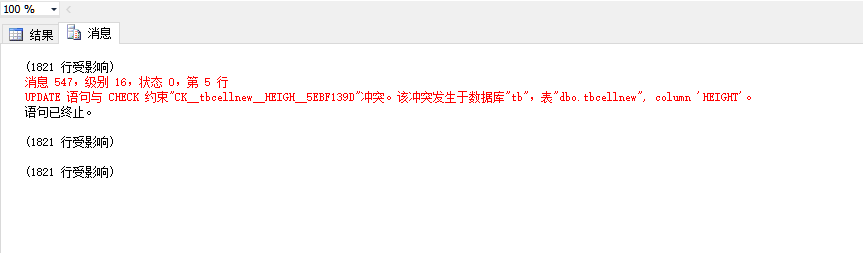
#### 11.1.1.1 顺序执行

|  |
| --- |
| select SECTOR\_ID,SECTOR\_NAME,HEIGHT from tbcellnew  where HEIGHT < 20  update tbcellnew set HEIGHT = HEIGHT - 15  where HEIGHT < 20  select SECTOR\_ID,SECTOR\_NAME,HEIGHT from tbcellnew  where HEIGHT < 20 |



#### 11.1.1.2 组织成事务执行

|  |
| --- |
| begin tran  select SECTOR\_ID,SECTOR\_NAME,HEIGHT from tbcellnew  where HEIGHT < 20  update tbcellnew set HEIGHT = HEIGHT - 15  where HEIGHT < 20  select SECTOR\_ID,SECTOR\_NAME,HEIGHT from tbcellnew  where HEIGHT < 20  commit tran  select SECTOR\_ID,SECTOR\_NAME,HEIGHT from tbcellnew  where HEIGHT < 20 |



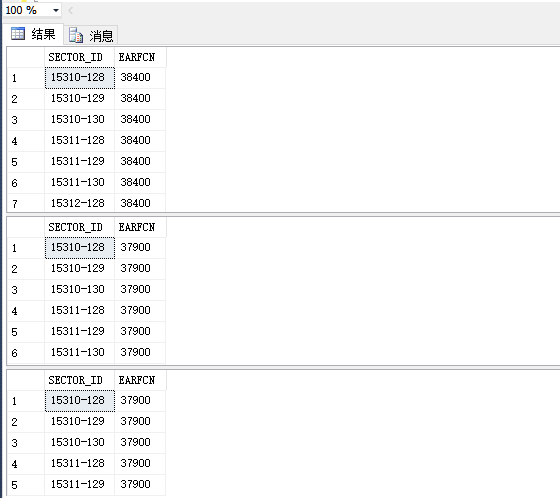
### 11.1.2 数据表更新

#### 11.1.2.1 显示执行模式

由于tbcell表中并没有数据的SECTOR\_ID位于’122880-0’和’1228822-2’之间，我们改为

‘15310-0’到’15380-0’之间

|  |
| --- |
| begin tran  begin try  select SECTOR\_ID,EARFCN from tbcellnew  where SECTOR\_ID between '15310-0' and '15380-0'  update tbCellnew set EARFCN=37900  where SECTOR\_ID between '15310-0' and '15380-0'  select SECTOR\_ID,EARFCN from tbcellnew  where SECTOR\_ID between '15310-0' and '15380-0'  end try  begin catch  select Error\_number() as ErrorNumber, --错误代码  Error\_severity() as ErrorSeverity, --错误严重级别，级别小于try catch 捕获不到  Error\_state() as ErrorState , --错误状态码  Error\_Procedure() as ErrorProcedure , --出现错误的存储过程或触发器的名称。  Error\_line() as ErrorLine, --发生错误的行号  Error\_message() as ErrorMessage --错误的具体信息  if(@@trancount>0) --全局变量@@trancount，事务开启此值+1，他用来判断是有开启事务  rollback tran ---由于出错，这里回滚到开始，第一条语句也没有插入成功。  end catch  if(@@trancount>0)  commit tran  select SECTOR\_ID,EARFCN from tbcellnew  where SECTOR\_ID between '15310-0' and '15380-0' |



只执行rollback过程：

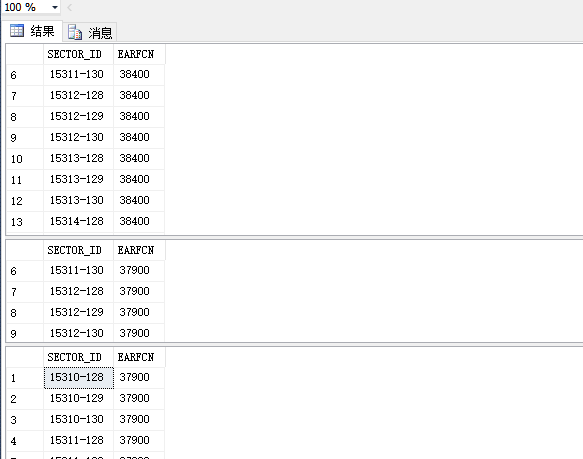
|  |
| --- |
| begin tran  select SECTOR\_ID,EARFCN from tbcellnew  where SECTOR\_ID between '15310-0' and '15380-0'  update tbCellnew set EARFCN=37900  where SECTOR\_ID between '15310-0' and '15380-0'  select SECTOR\_ID,EARFCN from tbcellnew  where SECTOR\_ID between '15310-0' and '15380-0'  rollback tran  select SECTOR\_ID,EARFCN from tbcellnew  where SECTOR\_ID between '15310-0' and '15380-0' |



#### 11.1.2.2 隐式事务

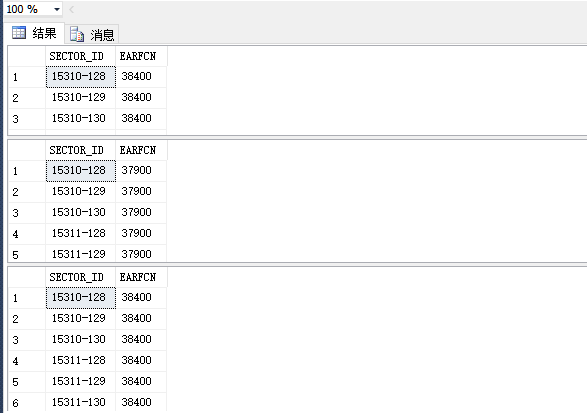
commit 提交事务：

|  |
| --- |
| Set IMPLICIT\_TRANSACTIONS ON  select SECTOR\_ID,EARFCN from tbcellnew  where SECTOR\_ID between '15310-0' and '15380-0'  update tbCellnew set EARFCN=37900  where SECTOR\_ID between '15310-0' and '15380-0'  select SECTOR\_ID,EARFCN from tbcellnew  where SECTOR\_ID between '15310-0' and '15380-0'  COMMIT TRANSACTION  Set IMPLICIT\_TRANSACTIONS OFF  select SECTOR\_ID,EARFCN from tbcellnew  where SECTOR\_ID between '15310-0' and '15380-0' |



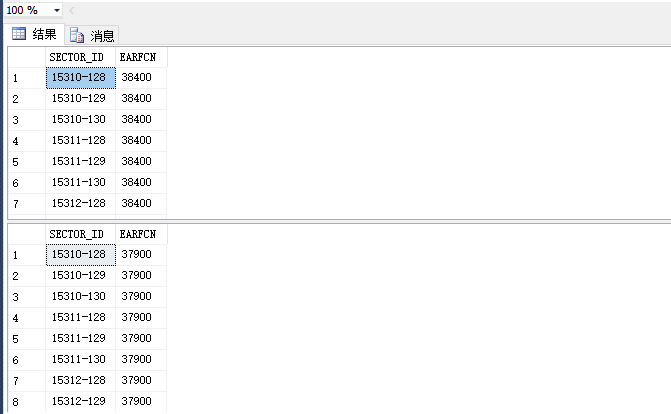
Rollback回滚事务：

|  |
| --- |
| Set IMPLICIT\_TRANSACTIONS ON  select SECTOR\_ID,EARFCN from tbcellnew  where SECTOR\_ID between '15310-0' and '15380-0'  update tbCellnew set EARFCN=37900  where SECTOR\_ID between '15310-0' and '15380-0'  select SECTOR\_ID,EARFCN from tbcellnew  where SECTOR\_ID between '15310-0' and '15380-0'  ROLLBACK TRANSACTION  Set IMPLICIT\_TRANSACTIONS OFF  select SECTOR\_ID,EARFCN from tbcellnew  where SECTOR\_ID between '15310-0' and '15380-0' |



#### 11.1.2.3 自动提交模式：

|  |
| --- |
| select SECTOR\_ID,EARFCN from tbcellnew  where SECTOR\_ID between '15310-0' and '15380-0'  update tbCellnew set EARFCN=37900  where SECTOR\_ID between '15310-0' and '15380-0'  select SECTOR\_ID,EARFCN from tbcellnew  where SECTOR\_ID between '15310-0' and '15380-0' |

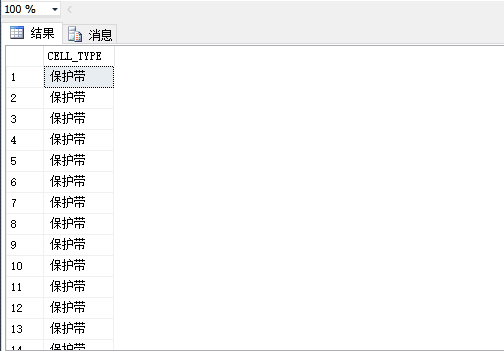


### 11.1.3 数据库模式修改

#### 11.1.3.1 删除列CELL\_TYPE

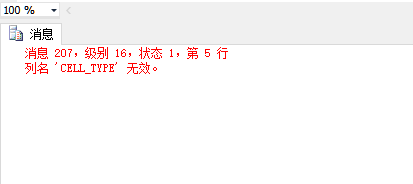
11.1.3.1.1 只有rollback的过程：

|  |
| --- |
| begin tran  alter table tboptcellnew  drop column CELL\_TYPE  rollback tran  select CELL\_TYPE from tboptcellnew |



11.3.1.1.2 执行commit

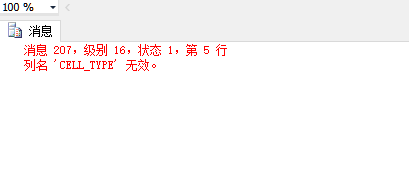
|  |
| --- |
| begin tran  alter table tboptcellnew  drop column CELL\_TYPE  commit tran  select CELL\_TYPE from tboptcellnew |



#### 11.1.3.2 增加列CELL\_TYPE

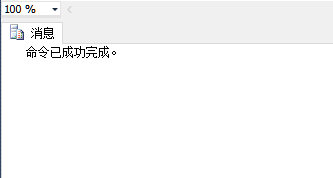
11.1.3.2.1 只有rollback的过程：

|  |
| --- |
| begin tran  alter table tboptcellnew  add CELL\_TYPE varchar NULL  rollback tran  select CELL\_TYPE from tboptcellnew |

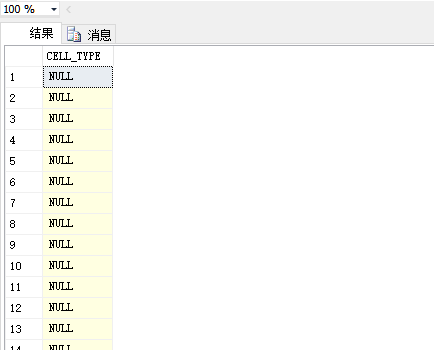


11.3.1.1.2 执行commit

|  |
| --- |
| begin tran  alter table tboptcellnew  add CELL\_TYPE varchar NULL  commit tran |



|  |
| --- |
| select CELL\_TYPE from tboptcellnew |



### 11.1.4 多条insert/delete操作执行比较

#### 11.1.4.1 顺序执行

|  |
| --- |
| select SECTOR\_ID,SECTOR\_NAME,HEIGHT from tbcellnew  where HEIGHT < 7  insert into tbcellnew (SECTOR\_ID,HEIGHT) values ('211100-2',6)  delete from tbcellnew where SECTOR\_ID = '211100-2'  select SECTOR\_ID,SECTOR\_NAME,HEIGHT from tbcellnew  where HEIGHT < 7 |



#### 11.1.4.2 组织成事务执行

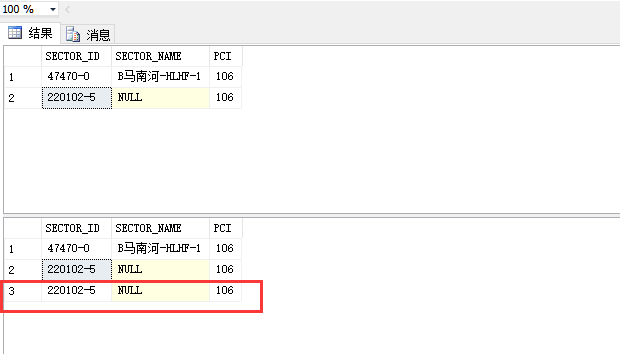
|  |
| --- |
| begin tran  select SECTOR\_ID,SECTOR\_NAME,HEIGHT from tbcellnew  where HEIGHT < 7  insert into tbcellnew (SECTOR\_ID,HEIGHT) values ('211100-2',6)  delete from tbcellnew where SECTOR\_ID = '211100-2'  select SECTOR\_ID,SECTOR\_NAME,HEIGHT from tbcellnew  where HEIGHT < 7  commit tran  select SECTOR\_ID,SECTOR\_NAME,HEIGHT from tbcellnew  where HEIGHT < 7 |

执行结果：



### 11.1.5 保存点 Savepoint 设置与回滚实验

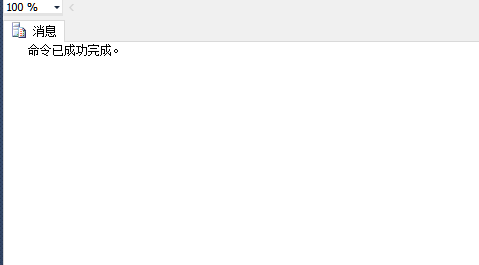
|  |
| --- |
| begin tran  select SECTOR\_ID,SECTOR\_NAME,PCI from tbpciassignment  where PCI = 106  insert into tbpciassignment (SECTOR\_ID,PCI) values ('220102-5',106)  save tran ppp  delete from tbpciassignment where SECTOR\_ID = '220102-5'  rollback tran ppp  commit tran  select SECTOR\_ID,SECTOR\_NAME,PCI from tbpciassignment  where PCI = 106 |



### 11.1.6 事务局部回滚/整体回滚

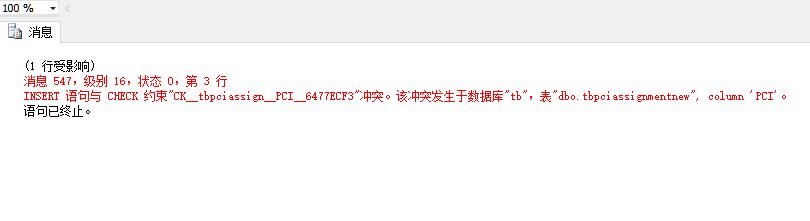
添加约束：

|  |
| --- |
| alter table tbpciassignmentnew add CHECK (PCI between 0 and 503); |



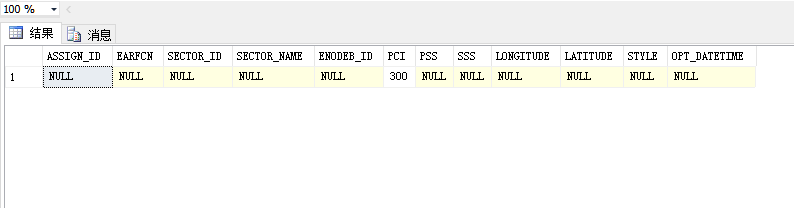
### 11.1.6.1 局部回滚

|  |
| --- |
| begin tran  insert into tbpciassignmentnew (PCI) values (300)  insert into tbpciassignmentnew (PCI) values (600)  commit tran |



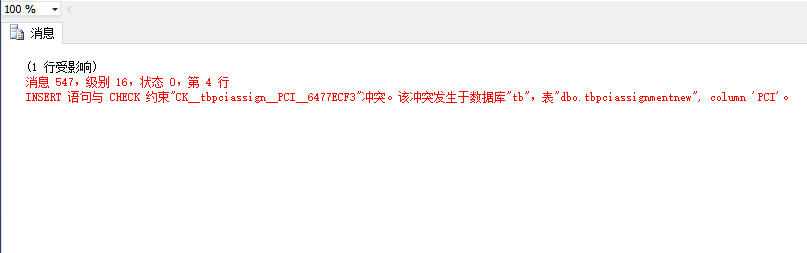
select:

|  |
| --- |
| select \* from tbpciassignmentnew where PCI = 300; |



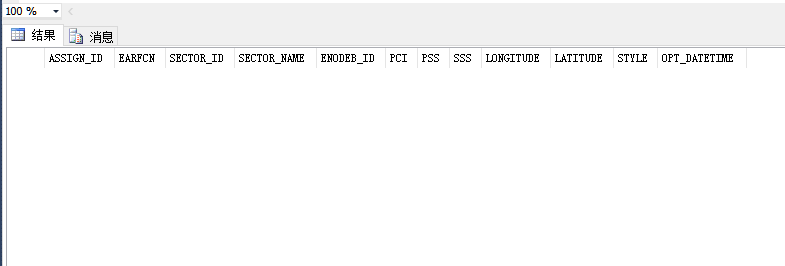
### 11.1.6.2 整体回滚

|  |
| --- |
| SET XACT\_ABORT ON  begin tran  insert into tbpciassignmentnew (PCI) values (300)  insert into tbpciassignmentnew (PCI) values (600)  commit tran |



select:

|  |
| --- |
| select \* from tbpciassignmentnew where PCI = 300 |

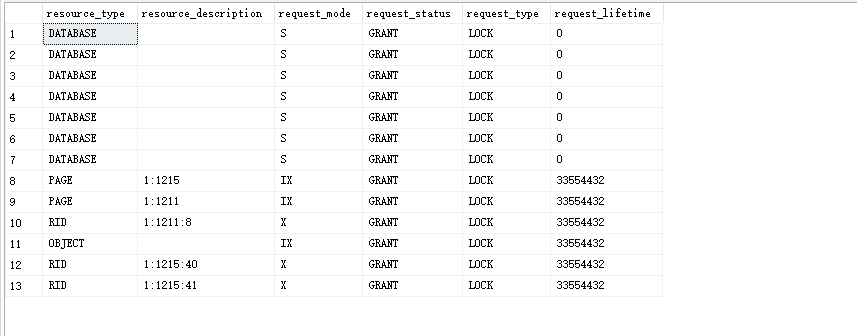


# 11.2

### 11.2.1 查看锁信息

|  |
| --- |
| begin tran  insert into tbpciassignmentnew (PCI) values (500)  select PCI from tbpciassignmentnew where PCI = 500  delete from tbpciassignmentnew where PCI = 500  update tbpciassignmentnew set PCI = PCI -5  where PCI = 180  select resource\_type,resource\_description,request\_mode,request\_status,request\_type,request\_lifetime  from sys.dm\_tran\_locks  commit tran |

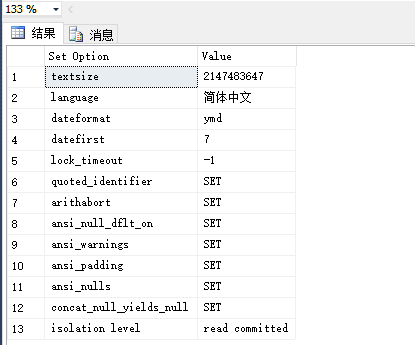




### 11.2.2 单事务隔离级别及加锁信息和执行结果观察

#### 11.2.2.1 查看系统默认的当前隔离级别

|  |
| --- |
| DBCC USEROPTIONS |



#### 11.2.2.2 设置不同的隔离级别并查看并发副作用

**Read-uncommitted 隔离级别下的脏读**

**实验内容：**

读未提交：在该隔离级别下，事务可以读取其他事务未提交的数据。

脏读：读取其他事务未提交的数据。

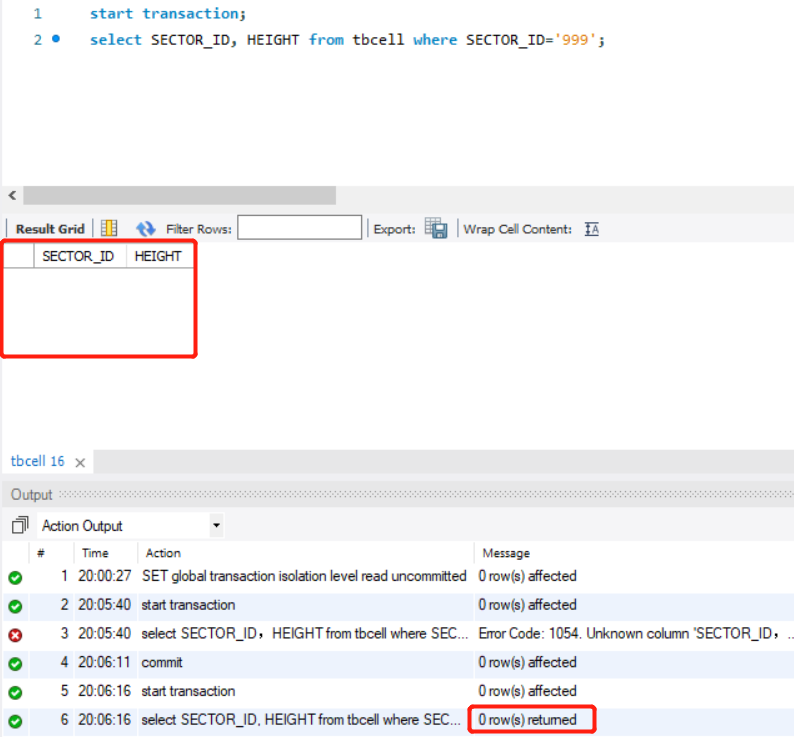
**实验过程：**

**Step1**：先使用下面语句将数据库系统隔离级别修改为 read-uncommitted



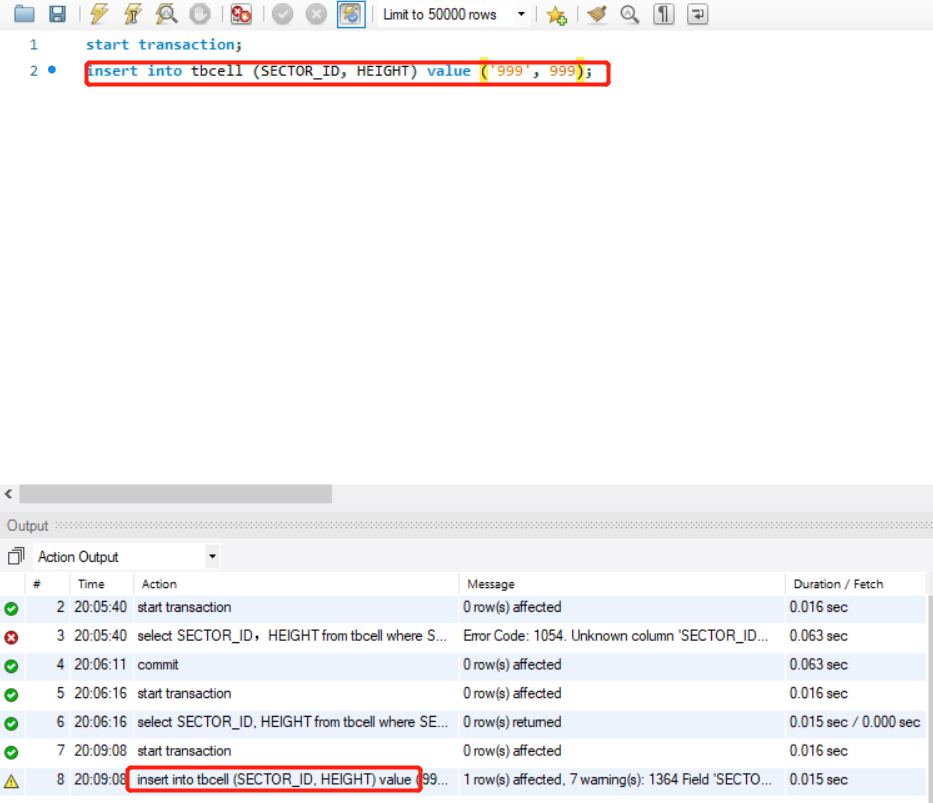
**Step2.** 开启事务一：执行查询 sector\_id=999 的数据行，没有数据返回

1. start **transaction**;
2. select SECTOR\_ID, HEIGHT **from** tbcell **where** SECTOR\_ID='999';



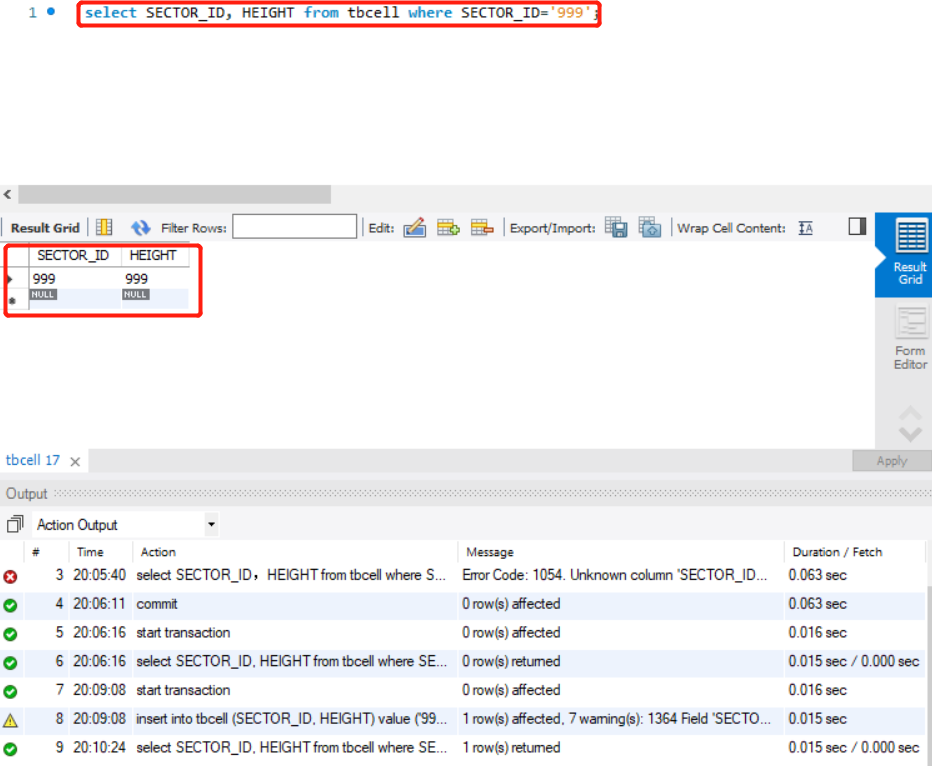
**Step3**. 开启事务二，执行插入 sector\_id=999 的操作，此时并未提交事务。

1. start **transaction**;
2. insert **into** tbcell (SECTOR\_ID, HEIGHT) value ('999', 999);



**Step4**. 在事务一中再次执行查询，发现有了数据，返回查看正是事务二中插入的数据。

1. **select** SECTOR\_ID, HEIGHT **from** tbcell **where** SECTOR\_ID='999';

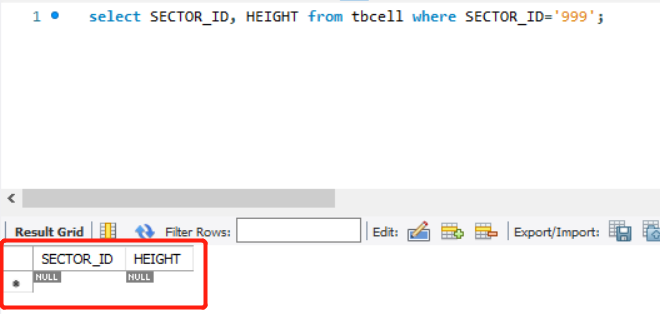


**Step5**. 返回事务二窗口，将事务回滚。



**Step6**.在事务一中再次执行查询，发现没有数据返回，此前查询到的数据就是脏数据。

1. **select** SECTOR\_ID, HEIGHT **from** tbcell **where** SECTOR\_ID='999';



**Read-committed 隔离级别下的不可重复读**

**实验内容：**

读提交：Read-committed是只能读取被提交后的数据。但一个事务两次读取相同行之间，另一个事务可以对该行进行修改，这就会导致两次读取之间出现差异，也就是不可重复读。

不可重复读：在同一事务内，不同的时刻读到的同一批数据不一样，可能会受到其他事务的影响，例如，在一个事务两次读取同一批数据期间，其他事务修改了这批数据并提交，导致该事务读取的同一批数据内容前后不一致。通常针对数据更新（UPDATE）操作。

**实验过程：**

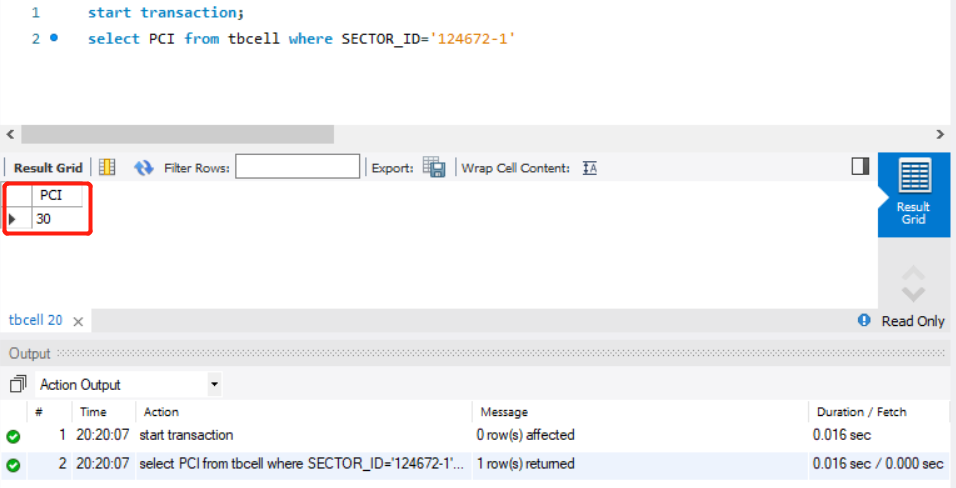
**Step1**. 先使用下面语句将数据库系统隔离级别修改为 read-committed：



**SET** **global** **transaction** **isolation** **level** **read** **committed**;

**Step2**. 先显式开启一个事务，并执行查询，此时为原始数据：

1. start **transaction**;
2. select PCI **from** tbcell **where** SECTOR\_ID='124672-1'



**Step3**. 在另一个窗口使用隐式事务（每条语句都看作是自动提交的事务），执行更新操作：



**update** tbcell **set** PCI=9999 **where** SECTOR\_ID='124672-1'

**Step4**. 返回显式事务中再次执行查询：

1. **select** PCI **from** tbcell **where** SECTOR\_ID='124672-1'

查询到了刚更新的数据

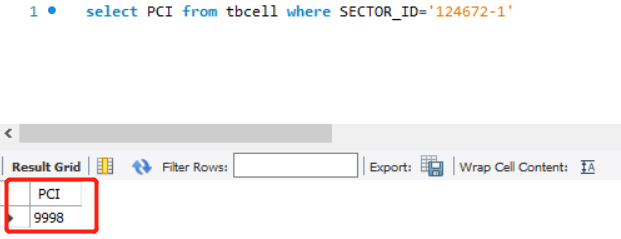


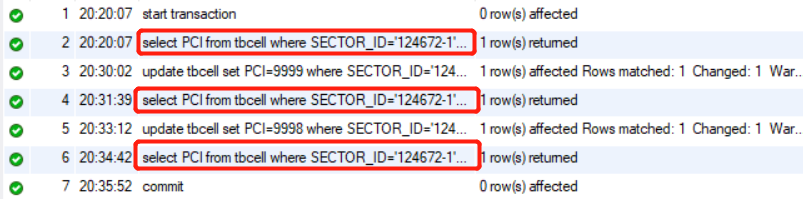
**Step5** 在隐式事务下再次执行更新操作，将 PCI 更新为 9998：



**Step6**. 返回事务中查询，返回的是 9998，在这次事务中，执行了三次相同的查询，但是出现了三个不同的结果， 出现不可重复读。

1. **select** PCI **from** tbcell **where** SECTOR\_ID='124672-1'





**Repeatable-read 隔离级别下的幻读**

**实验内容：**

可重复读：在隔离界别为Repeatable-read隔离级别下，只能读取已提交的数据，同时两次读取之间，其他事务不能修改，但是其他事务可以插入和删除。在该情况下，会造成幻读。

幻读：针对数据插入（INSERT）操作。假设事务 A 对某些行的内容作了更改，但是还未提交，此时事务 B 插入了与事务 A 更改前的记录相同的记录行，并且在事务 A 提交之前先提交了。而这时，在事务 A中查询，会发现好像刚刚的更改对于某些数据未起作用，但其实是事务 B 刚插入进来的，让用户感觉出现了幻觉，称为幻读。

**实验过程：**

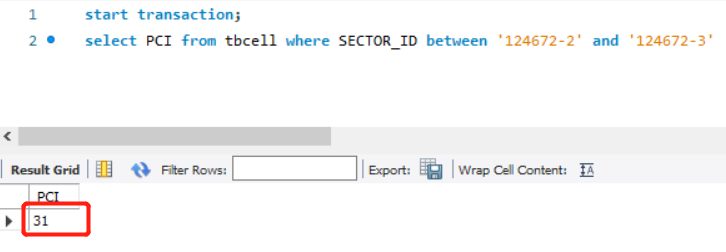
**Step1**. 先使用下面语句将数据库系统隔离级别修改为 Repeatable-read：

1. **SET** **global** **transaction** **isolation** **level** **repeatable** **read**;



**Step2**. 开启一个事务，并执行查询语句：

1. start **transaction**;
2. select PCI **from** tbcell **where** SECTOR\_ID between '124672-2' and '124672-3'

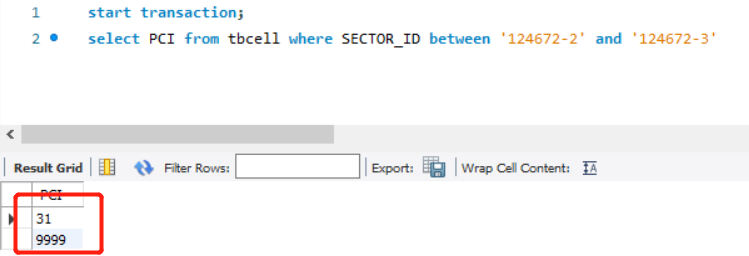


**Step3**. 在另一个窗口执行隐式事务，执行下面这条插入语句：

1. **insert** **into** tbcell(SECTOR\_ID, PCI) **values** ('124672-3', 9999)



**Step4**. 返回事务窗口中再次执行查询，发现返回集中多了一条数据，出现幻读：



**小结：**

幻读和不可重复读表现形式上看似都是多次查询但是结果不同，但两者本质上是不同的。不可重复读关注点在update操作上，而幻读关注点则是insert和delete操作。简单来说，幻读前后数据量不同，不可重复读是同一行数据前后读取到的内容不同。

如果使用锁机制实现这两种隔离级别，在可重复读中，该SQL语句第一次读取到数据后，就将这些数据加锁，其它事务无法修改这些数据，从而实现可重复读。但这种方法却无法锁住insert的数据，所以当事务A先前读取了数据，或者修改了全部数据，事务B还是可以insert数据提交，这时事务A就会发现莫名其妙多了一条之前没有的数据，这就是幻读，无法通过行锁来避免。

为避免幻读，需要采用Serializable隔离级别 ，读用读锁，写用写锁，读锁和写锁互斥，可以有效避免幻读、不可重复读、脏读等问题，但会极大的降低数据库的并发能力。

Mysql 的最高隔离级别是serializable此时事务对某个表操作时会将表加上表锁，其他事务要想也操作这个表那就只能等带持锁的事务完成推出后才能执行，此时失去了并发能力。

**Serializable隔离级别下多事务访问**

**实验内容：**

设置隔离级别为Serializable，检测是否出现脏读，不可重复读，幻读等数据不一致。

**实验过程：**

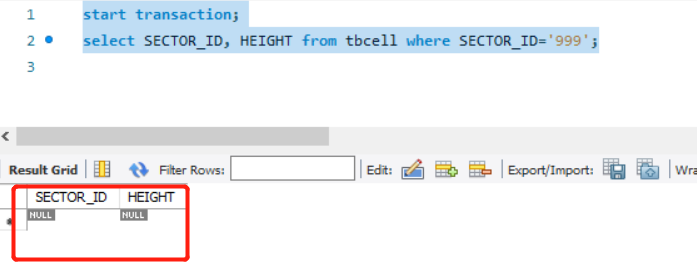
1. **SET** **global** **transaction** **isolation** **level** **serializable**;



**脏读：**

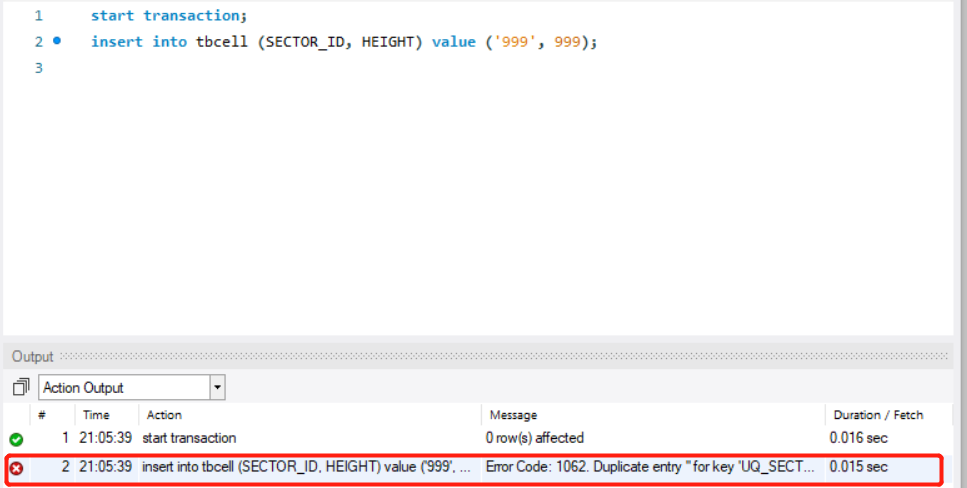
**Step1**. 开启事务一：执行查询 sector\_id=999 的数据行，没有数据返回

1. start **transaction**;
2. select SECTOR\_ID, HEIGHT **from** tbcell **where** SECTOR\_ID='999';



**Step2**. 开启事务二，执行插入 sector\_id=999 的操作，此时并未提交事务。

1. start **transaction**;
2. selert **into** tbcell (SECTOR\_ID, HEIGHT) value ('999', 999);

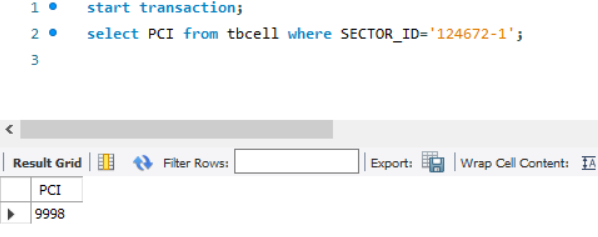


可以看到，当隔离级别为Serializable，当有一个事务查询后，若有其他事务通过插入的方式改变查询结果是不可能的，事务插入会被阻断。可见在该隔离级别下，通过限制可能造成脏读的插入，来保证不会发生脏读。

**不可重复读：**

**Step1**. 先显式开启一个事务，并执行查询，此时为原始数据：

1. start **transaction**;
2. select PCI **from** tbcell **where** SECTOR\_ID='124672-1';



**Step2**. 在另一个窗口使用隐式事务（每条语句都看作是自动提交的事务），执行更新操作：



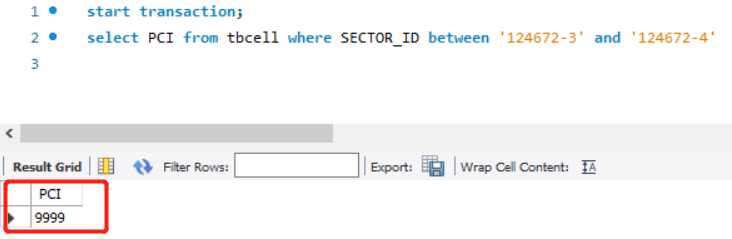
发现不能更新。

由于在上一部查询中，已经对于该行加锁，更新操作只能等待。这样不会造成在一个事务内的多次查询造成查询结果不一致的错误，也就是不可重复读错误。

**幻读：**

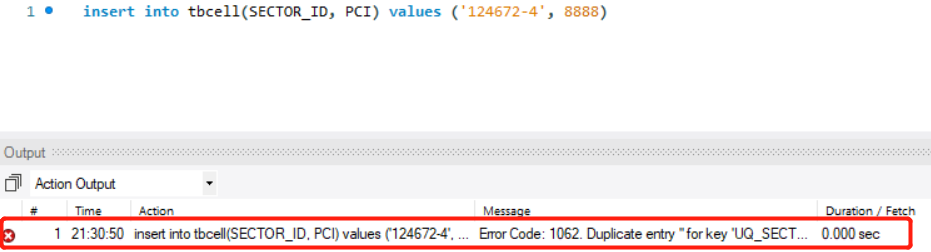
**Step1**. 开启一个事务，并执行查询语句：

1. start **transaction**;
2. select PCI **from** tbcell **where** SECTOR\_ID between '124672-3' and '124672-4'



**Step2**. 在另一个窗口执行隐式事务，执行下面这条插入语句：

1. **insert** **into** tbcell(SECTOR\_ID, PCI) **values** ('124672-4', 8888)



可以看到，插入失败。

插入失败，也就保证了事务的两次查询之间不会造成查询结果数量的差异，保证了不会出现幻读。

**总结：**

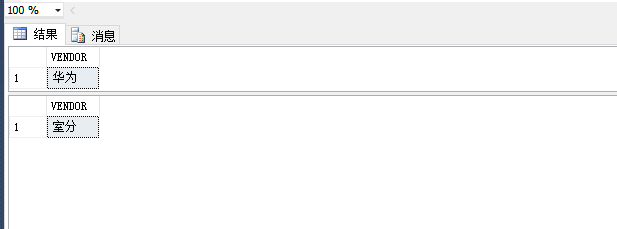
|  |  |
| --- | --- |
| **隔离级别** | **可能出现的不一致现象** |
| Read-uncommitted | 脏读，不可重复读，幻读 |
| Read-committed | 不可重复读，幻读 |
| Repeatable-read | 幻读 |
| Serializable | - |

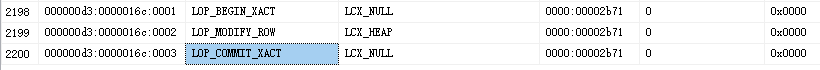
# 11.3 事务日志观察

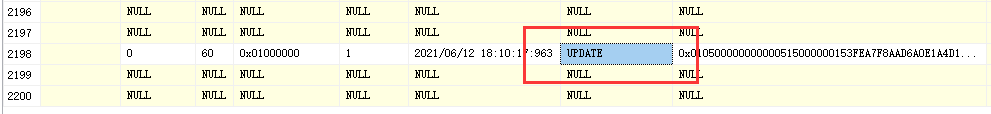
### 11.3.1 单条SQL语句(select/update/insert/delete)在自动提交模式下的日志观察

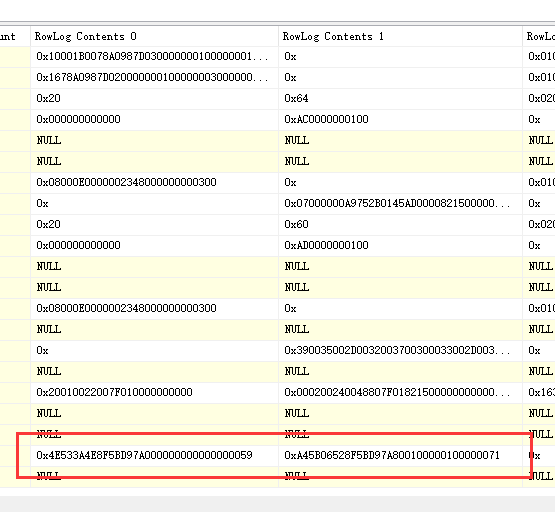
执行过程：

|  |
| --- |
| select VENDOR from tbcellnew where SECTOR\_ID = '124674-1'  update tbcellnew set VENDOR = '室分' where SECTOR\_ID = '124674-1'  select VENDOR from tbcellnew where SECTOR\_ID = '124674-1'  insert into tbcellnew (SECTOR\_ID,HEIGHT) values ('211100-2',6)  delete from tbcellnew where SECTOR\_ID = '211100-2'  SELECT \* FROM [sys].[fn\_dblog](NULL,NULL) |

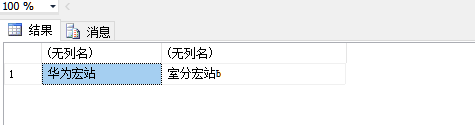








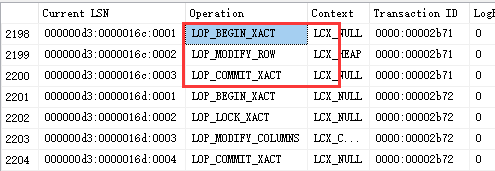
|  |
| --- |
| select cast(0x4E533A4E8F5BD97A000000000000000059 as nvarchar(255)),cast(0xA45B06528F5BD97A800100000100000071 as nvarchar(255)) |



### 11.3.2 显示执行模式下，多条语句组成的单个事务的日志的观察

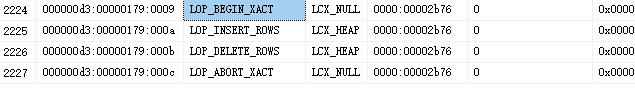
#### 11.3.2.1 以commit结束的事务日志实验

|  |
| --- |
| begin tran  select SECTOR\_ID from tbcellnew where HEIGHT < 7  update tbcellnew set HEIGHT = 10 where SECTOR\_ID = '1246778-2'  insert into tbcellnew (SECTOR\_ID,SECTOR\_NAME,HEIGHT) values ('211100-2','123',6)  delete from tbcellnew where SECTOR\_ID = '211100-2'  commit tran  SELECT \* FROM [sys].[fn\_dblog](NULL,NULL) |



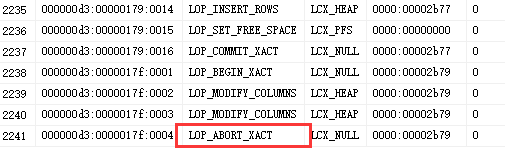
#### 11.3.2.2 以rollback结束的事务日志实验

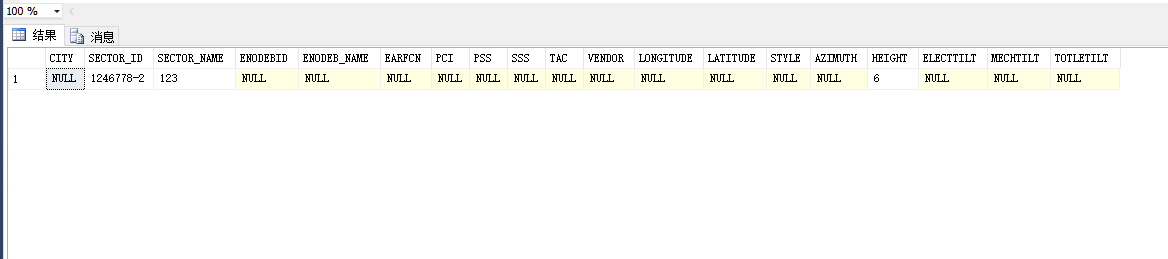
|  |
| --- |
| begin tran  select SECTOR\_ID from tbcellnew where HEIGHT < 7  update tbcellnew set HEIGHT = 10 where SECTOR\_ID = '1246778-2'  insert into tbcellnew (SECTOR\_ID,SECTOR\_NAME,HEIGHT) values ('211100-2','123',6)  delete from tbcellnew where SECTOR\_ID = '211100-2'  rollback tran  SELECT \* FROM [sys].[fn\_dblog](NULL,NULL) |



#### 11.3.2.3 事务发生中断的事务日志实验

|  |
| --- |
| begin tran  select SECTOR\_ID from tbcellnew where HEIGHT < 7  update tbcellnew set HEIGHT = 10 where SECTOR\_ID = '1246778-2'  WAITFOR DELAY '00:03:00'  insert into tbcellnew (SECTOR\_ID,SECTOR\_NAME,HEIGHT) values ('211100-2','123',6)  delete from tbcellnew where SECTOR\_ID = '211100-2'  commit tran  SELECT \* FROM [sys].[fn\_dblog](NULL,NULL) |





### 11.3.3 显示执行模式下，多条或单条语句组成的多个并发事务的日志观察

会话一：

|  |
| --- |
| SET TRANSACTION ISOLATION LEVEL REPEATABLE READ  begin tran  select PCI,SECTOR\_NAME from tbcellnew where SECTOR\_ID = '124694-0'  WAITFOR DELAY '00:00:05'  select PCI from tbcellnew where SECTOR\_ID = '124694-0'  commit tran  select \* from [sys].[fn\_dblog](NULL,NULL) |

会话二：

|  |
| --- |
| update tbcellnew set PCI = 400 where SECTOR\_ID = '124694-0' |

