

本科生实验报告

学生姓名: 丁晓琪

学生学号: 22336057

专业名称: 计科

一: 实验任务:

(1)设置"未提交读"隔离级别(READ UNCOMMITTED),在students表上演示读"脏"数据。

(2)设置"提交读"隔离级别(READ COMMITTED),在students表上演示避免读"脏"数据。

(3)设置"可重复读"隔离级别(REPEATABLE READ),在students表上演示避免读"脏"数据、不可重复读,但不能避免幻象读。

(4)设置 "可串行化"隔离级别(SERIALIZABLE),在students表上演示防止其他用户在事务提交之前更新数据。

二:实验过程:

- 1. 设置"未提交读"隔离级别(READ UNCOMMITTED),在students表上演示读"脏"数据
 - (1) 建立事务1: 在事务1中更新grade, 延时20s后, 回滚到初始状态

```
use School
go

BEGIN TRAN

UPDATE STUDENTS SET grade=7 where sid='1' --修改年级
WAITFOR DELAY '00:00:20' --延时20s
select * from STUDENTS where sid='1'
ROLLBACK TRAN
SELECT * FROM STUDENTS where sid='1'
```

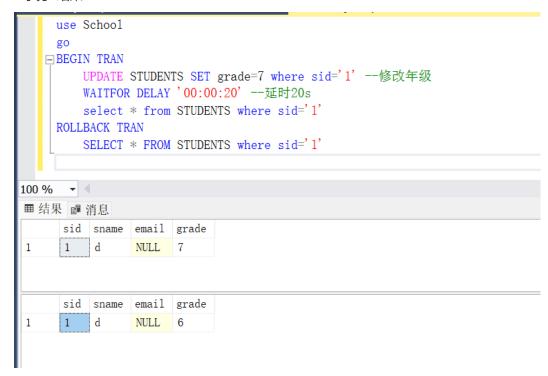
(2) 建立查询2:设置事务隔离级别为read uncommitted,查询sid,延时20s后再次查询

```
use School
go

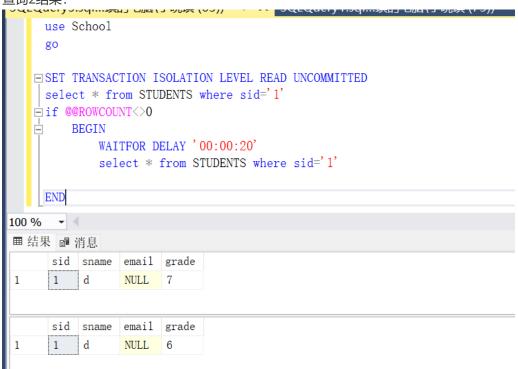
SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED
select * from STUDENTS where sid='1'
Fif @@ROWCOUNT<>0
BEGIN
WAITFOR DELAY '00:00:20'
select * from STUDENTS where sid='1'

END
```

- (3) 在执行事务1的过程中执行查询2: 可以看见查询2的结果两次select不一样,读取到了脏数据
- 事务1结果:



。 查询2结果:



- 2. 设置"提交读"隔离级别(READ COMMITTED),在students表上演示避免读"脏"数据。
 - (1) 事务1: 和1的事务1相同,在事务1中更新grade,延时20s后,回滚到初始状态

```
use School
go

BEGIN TRAN

UPDATE STUDENTS SET grade=7 where sid='1' --修改年级
WAITFOR DELAY '00:00:20' --延时20s
select * from STUDENTS where sid='1'
ROLLBACK TRAN
SELECT * FROM STUDENTS where sid='1'
```

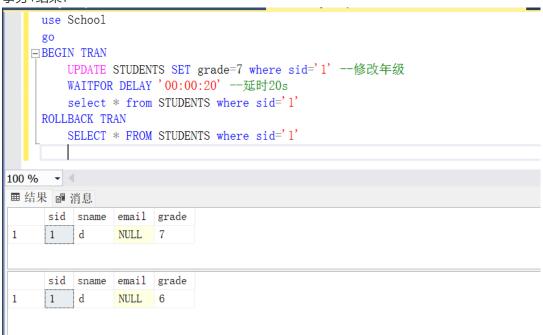
(2) 查询2: 将1的查询2修改隔离级别为COMMITTED, 查询sid, 延时20s后再次查询

```
use School
go

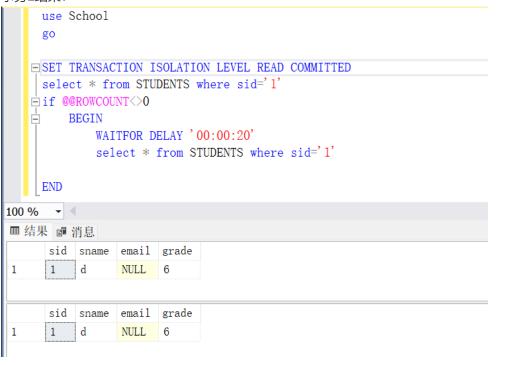
ESET TRANSACTION ISOLATION LEVEL READ COMMITTED
select * from STUDENTS where sid='1'
eif @@ROWCOUNT<>0
BEGIN
WAITFOR DELAY '00:00:20'
select * from STUDENTS where sid='1'
END
```

(3) 在执行事务1的过程中执行查询2,查询2的两次grade相同,且为原始数据,说明查询2没有读取事务1执行过程中的脏数据

事务1结果:



。 事务2结果:



- 3. 设置"可重复读"隔离级别(REPEATABLE READ),在students表上演示避免读"脏"数据、不可重复 读,但不能避免幻象读
 - (1)建立事务1:设置事务隔离级别为可重复读级别,先查询grade,延迟20s后再次查询grade

```
□ use School

SET TRANSACTION ISOLATION LEVEL REPEATABLE READ

BEGIN TRAN

select * from STUDENTS where sid='1'

□ if @@ROWCOUNT<>0

□ BEGIN

WAITFOR DELAY '00:00:20'

select * from STUDENTS where sid='1'

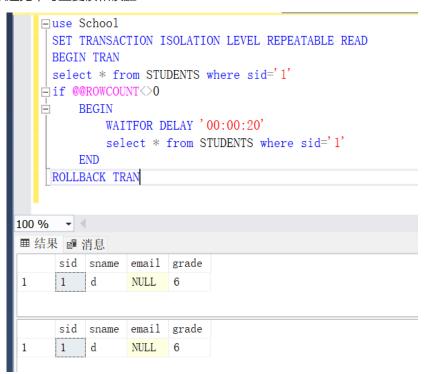
END

ROLLBACK TRAN
```

(2) 建立查询2: 设置事务隔离级别为可重复读级别, 更新grade=7

```
□ use School
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ
UPDATE STUDENTS SET grade=7 where sid='1'
```

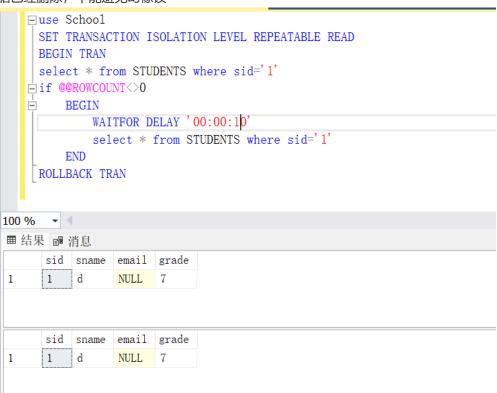
(3) 在执行事务1的过程中执行查询2: 事务1的两次查询结果相同均为grade的初始值6, 这表明可重复读级别可以避免不可重复读和读脏



(4) 将事务1的延时改为10s且将查询2更改为删除students表中sid='1'的记录

```
∃use School
    SET TRANSACTION ISOLATION LEVEL REPEATABLE READ
    delete from STUDENTS where sid='1
100 % -
■ 消息
   (1 行受影响)
  完成时间: 2024-12-02T17:14:58.2124263+08:00
  ⊟use School
   SET TRANSACTION ISOLATION LEVEL REPEATABLE READ
   select * from STUDENTS where sid='1'
  if @@ROWCOUNT<>0
  Ė BEGIN
          WAITFOR DELAY '00:00:10'
        select * from STUDENTS where sid='1'
       END
   ROLLBACK TRAN
```

(5) 先执行事务1, 在事务1执行过程中执行查询2: 事务1两次查询结果相同, 但是sid='1'对应的数据已经删除, 不能避免幻像读



4. 设置 "可串行化"隔离级别(SERIALIZABLE),在students表上演示防止其他用户在事务提交之前更新数据

(1) 建立事务1: 如同3的事务1, 但是隔离级别改为可串行

```
□ use School

SET TRANSACTION ISOLATION LEVEL SERIALIZABLE

BEGIN TRAN

select * from STUDENTS where sid='1'

□ if @@ROWCOUNT<>0

□ BEGIN

WAITFOR DELAY '00:00:10'

select * from STUDENTS where sid='1'

END

ROLLBACK TRAN
```

(2)建立查询2:隔离级别为可串行,向students表插入表项

```
回use School
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE
Insert into STUDENTS (sid, sname, email, grade) values ('1', 'd', NULL, 6)

100% ▼
□ 消息

(1 行受影响)
完成时间: 2024-12-02T17:20:51.1982941+08:00
```

(3) 在事务1执行过程中,执行查询2:事务1两次查询结果都为空,表明在事务1执行过程中防止查询2向其插入数据

```
□ use School

SET TRANSACTION ISOLATION LEVEL SERIALIZABLE

BEGIN TRAN

select * from STUDENTS where sid='1'
□ if @@ROWCOUNT<>0
□ BEGIN

WAITFOR DELAY '00:00:10'

select * from STUDENTS where sid='1'
END

ROLLBACK TRAN
```

三: 实验总结:

- 事物隔离级别:
 - read uncommitted:未提交读,读脏
 - o read committed:已提交读,不读脏,不允许重复读,SQL默认级别
 - o repeatable read: 可重复读,禁止读脏和不重复读,但允许幻象读
 - o serializable:可串行化,最高级别,事务不能并发,只能串行
- 设置事务的隔离级别

- 1 | SET TRANSACTION ISOLATION LEVEL {可选隔离级别}
- 延时操作:

1 WAITFOR DELAY '00:00:10'