

Lecture 14, Fall 2017/2018

数据库系统实验

Yubao Liu (刘玉葆)

School of Data and Computer Science

Sun Yat-sen University

- 本节课提纲

- 实验目的
- 实验内容
- 实验示例
- 练习

- 实验目的

理解事务并发中不一致的问题，以及通过设置隔离级别解决不一致问题。

• 实验内容

事务并发不一致问题：

- 读“脏”数据：一个事务读取另一个事务尚未提交的数据引起。
- 不可重复读：事务T1读取数据a后，事务T2对数据a进行更新，事务T1再次读取，无法读取前一次的结果。
- 幻象读：事务T1两次查询过程中，事务T2对数据进行插入或删除，导致事务T1两次查询的记录数不一致。

事务隔离级别：

- READ UNCOMMITTED(未提交读，读脏)
- READ COMMITTED(已提交读，不读脏，但允许不重复读，SQL默认级别)
- REPEATABLE READ(可重复读，禁止读脏和不重复读，但允许幻象读)
- SERIALIZABLE(可串行化，最高级别，事务不能并发，只能串行)

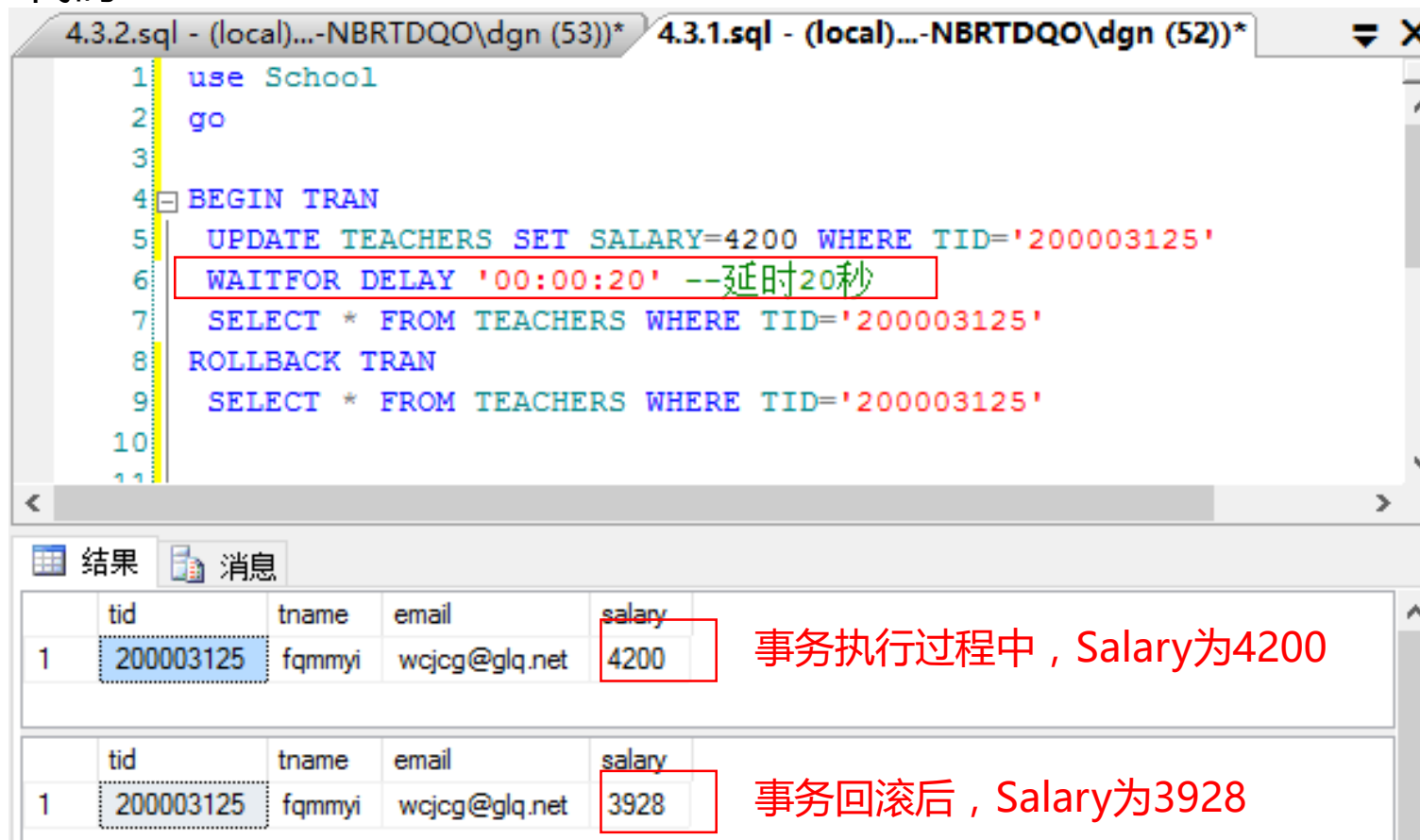
• 实验示例

1.读“脏”数据的实例。

Step1：新建查询1，实现在事务1中更新salary，延时20秒后，事务回滚至初始状态，如代码1所示。

Step2：**在事务1执行过程中，执行查询2。**查询2为查询salary，延时20秒后，再次查询salary。

代码1：



```
4.3.2.sql - (local)...-NBRTDQO\dgn (53))* 4.3.1.sql - (local)...-NBRTDQO\dgn (52))*
1 use School
2 go
3
4 BEGIN TRAN
5 UPDATE TEACHERS SET SALARY=4200 WHERE TID='200003125'
6 WAITFOR DELAY '00:00:20' --延时20秒
7 SELECT * FROM TEACHERS WHERE TID='200003125'
8 ROLLBACK TRAN
9 SELECT * FROM TEACHERS WHERE TID='200003125'
10
11
```

	tid	tname	email	salary
1	200003125	fqmmyi	wcjcg@glq.net	4200

事务执行过程中，Salary为4200

	tid	tname	email	salary
1	200003125	fqmmyi	wcjcg@glq.net	3928

事务回滚后，Salary为3928

实验示例

代码2：

The screenshot displays a SQL script in a window titled '4.3.2.sql - (local)...-NBRTDQO\dgn (53))*'. The script is as follows:

```
1 use School
2 go
3
4 SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED
5 -- 模拟实现脏读
6 SELECT * FROM TEACHERS WHERE TID='200003125'
7 IF @@ROWCOUNT <> 0
8 BEGIN
9     WAITFOR DELAY '00:00:20'
10    -- PRINT 模拟实现不可重复读
11    SELECT * FROM TEACHERS WHERE TID='200003125'
12 END
```

Below the script, the 'Results' tab shows two query results. The first result shows a salary of 4200, and the second result shows a salary of 3928.

	tid	tname	email	salary
1	200003125	fqmmyi	wcjcg@glq.net	4200

	tid	tname	email	salary
1	200003125	fqmmyi	wcjcg@glq.net	3928

事务隔离的最低级别，仅保证不读物理损坏的数据。

读“脏”数据，查询的是事务1没有提交前的数据

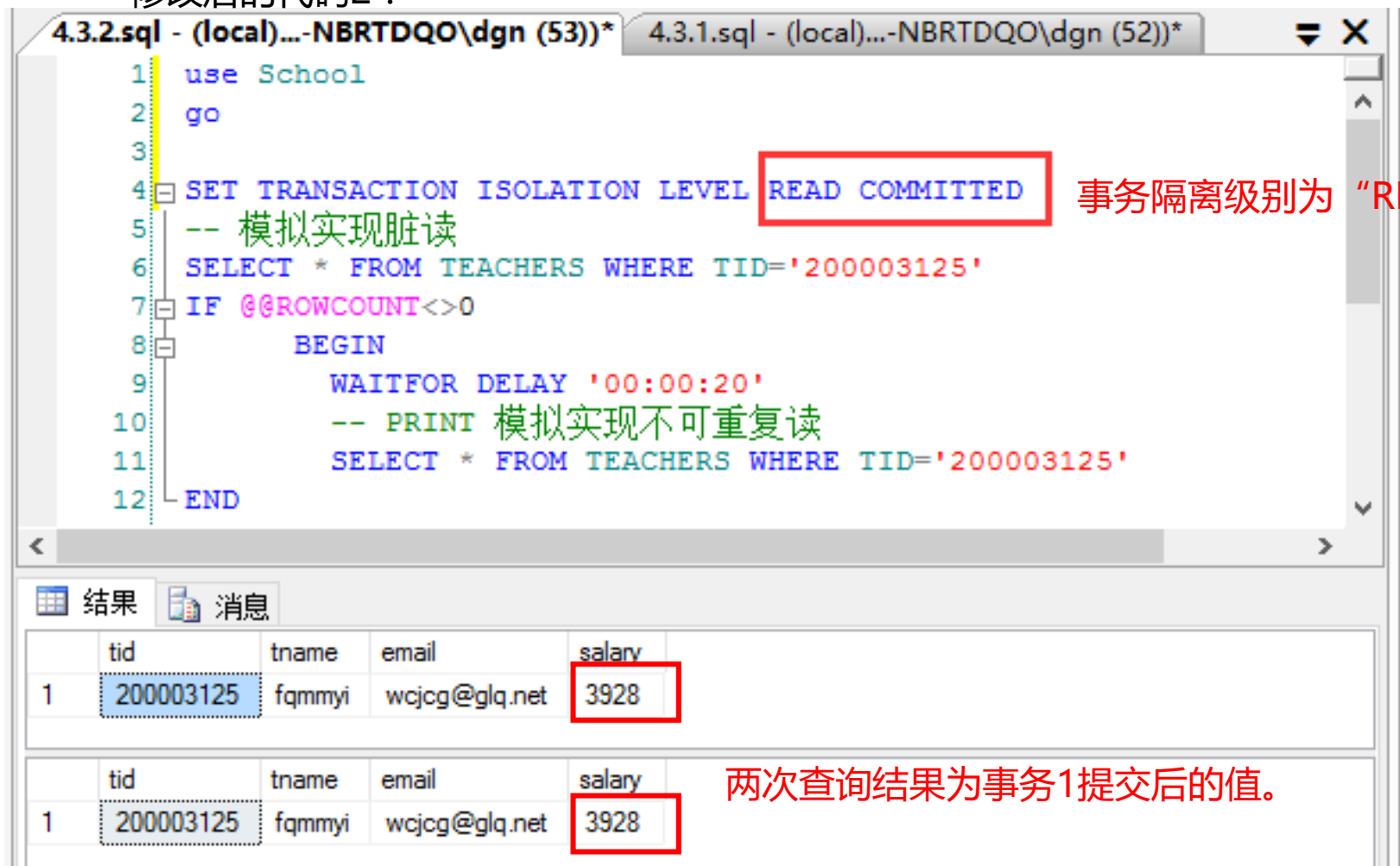
延时20秒后再次查询，与第一次查询结果不一样，发生了“不可重复读”。

原因：事务1更新数据过程与查询2的执行过程没有隔离开来，所以导致上述现象。

实验示例

2.为了解决示例1中“读脏数据”的问题，将查询2的隔离级别改为“READ COMMITTED”（提交读），可保证不读到脏数据。再次重复示例1的过程，可以发现查询2读到的是事务1提交后的结果。

修改后的代码2：



```
4.3.2.sql - (local)...\NBRTDQO\dgn (53))* 4.3.1.sql - (local)...\NBRTDQO\dgn (52))*
1 use School
2 go
3
4 SET TRANSACTION ISOLATION LEVEL READ COMMITTED
5 -- 模拟实现脏读
6 SELECT * FROM TEACHERS WHERE TID='200003125'
7 IF @@ROWCOUNT<>0
8 BEGIN
9     WAITFOR DELAY '00:00:20'
10    -- PRINT 模拟实现不可重复读
11    SELECT * FROM TEACHERS WHERE TID='200003125'
12 END
```

事务隔离级别为“READ COMMITTED”，提交读。

	tid	tname	email	salary
1	200003125	fqmmmyi	wcjcg@glq.net	3928

	tid	tname	email	salary
1	200003125	fqmmmyi	wcjcg@glq.net	3928

两次查询结果为事务1提交后的值。

• 实验示例

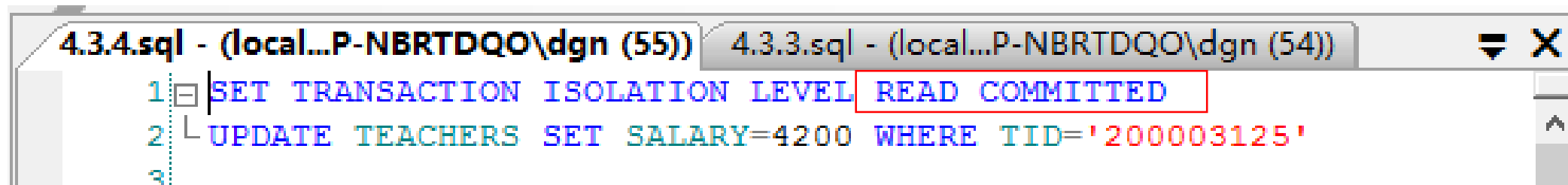
3.设置“提交读”隔离级别，避免脏读，允许不可重复读的实例。

Step1：新建查询3，设置隔离级别为“提交读”，在事务3中查询salary，延时20秒后，再次相同查询。

Step2：新建查询4，**在执行事务3过程中，执行代码4更新salary。**

会发现查询3中，两次查询结果不同，出现了“不可重复读”。

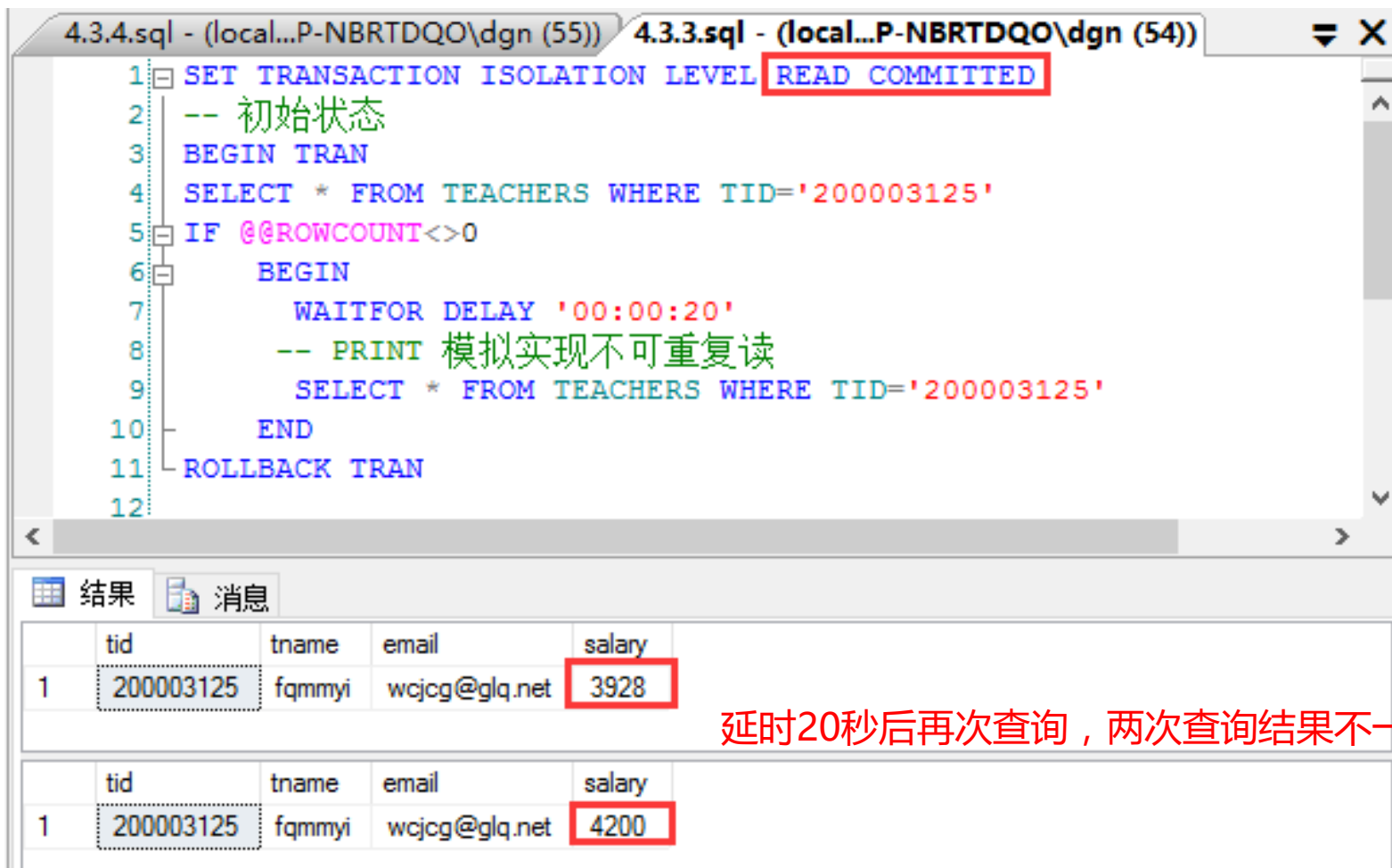
代码4如下：



```
4.3.4.sql - (local...P-NBRTDQO\dgn (55)) 4.3.3.sql - (local...P-NBRTDQO\dgn (54))
1 SET TRANSACTION ISOLATION LEVEL READ COMMITTED
2 UPDATE TEACHERS SET SALARY=4200 WHERE TID='200003125'
3
```


实验示例

代码3如下：



The screenshot shows a SQL Server query window with two tabs. The active tab, '4.3.4.sql - (local...P-NBRTDQO\dgn (55))', contains the following SQL code:

```
1 SET TRANSACTION ISOLATION LEVEL READ COMMITTED
2 -- 初始状态
3 BEGIN TRAN
4 SELECT * FROM TEACHERS WHERE TID='200003125'
5 IF @@ROWCOUNT <> 0
6 BEGIN
7     WAITFOR DELAY '00:00:20'
8     -- PRINT 模拟实现不可重复读
9     SELECT * FROM TEACHERS WHERE TID='200003125'
10 END
11 ROLLBACK TRAN
12
```

The 'RESULTS' tab shows the results of the first query, which is a table with 5 columns: tid, tname, email, salary. The first row has the value 3928 in the salary column, which is highlighted with a red box.

	tid	tname	email	salary
1	200003125	fqmmys	wcjcg@glq.net	3928

Below the first result table, the same table structure is shown again, but with the salary value 4200, also highlighted with a red box.

	tid	tname	email	salary
1	200003125	fqmmys	wcjcg@glq.net	4200

注：SQL默认隔离级别为“提交读”。

延时20秒后再次查询，两次查询结果不一样，发生了“不可重复读”。

原因：同一事务中两次相同数据读取之间，“提交读”允许其他事务在两次读取的间隙修改资源，所以导致读取不一致的现象。

• 实验示例

4. 为了解决示例3 “不可重复读” 的问题，设置 “可重复读” 隔离级别 (REPEATABLE READ) 。

将代码3的隔离级别改为 “REPEATABLE READ” ,在执行代码3的过程中，执行以下代码更新salary为4500。

4.3.5.sql - (local)...-NBRTDQO\dgn (56))*

```
1 SET TRANSACTION ISOLATION LEVEL REPEATABLE READ
2 UPDATE TEACHERS SET SALARY=4500 WHERE TID='200003125'
3
```

4.3.4.sql - (local)...-NBRTDQO\dgn (55))*

实验示例

修改后的代码3：

The screenshot shows a SQL IDE with two tabs. The active tab is 4.3.5.sql, which contains the following SQL code:

```
1 SET TRANSACTION ISOLATION LEVEL REPEATABLE READ
2 -- 初始状态
3 BEGIN TRAN
4 SELECT * FROM TEACHERS WHERE TID='200003125'
5 IF @@ROWCOUNT <> 0
6 BEGIN
7     WAITFOR DELAY '00:00:20'
8     -- PRINT 模拟实现不可重复读
9     SELECT * FROM TEACHERS WHERE TID='200003125'
10 END
11 ROLLBACK TRAN
12
```

Red annotations highlight the isolation level and the second query:

- A red box around line 1: `SET TRANSACTION ISOLATION LEVEL REPEATABLE READ`
- Red text to the right: 可重复读级别
- Red text to the right of the second query: 保证了读一致.

Below the code, the '结果' (Results) tab is active, showing two identical query results:

	tid	tname	email	salary
1	200003125	fqmmmyi	wcjcg@glq.net	4200

Red arrows point from the '保证了读一致.' annotation to the salary value '4200' in both result sets.

实验示例

5. “可重复读”级别的局限性，可能出现“幻象读”。

Step1：执行代码5，代码5为查询teachers表，延迟10秒后再次相同查询。

Step2：在执行代码5过程中，执行代码6。代码6为删除teachers表该记录。

代码5如下：

```
4.3.8.sql - (local...P-NBRTDQO\dgn (60)) 4.3.7.sql - (local...P-NBRTDQO\dgn (59))*
1 insert into teachers values ('300000000', 'AA', 'BBB@163.COM', 5000)
2 set transaction isolation level repeatable read
3 BEGIN TRAN
4 SELECT * FROM TEACHERS WHERE TID='300000000 '
5 IF @@ROWCOUNT <> 0
6 BEGIN
7     WAITFOR DELAY '00:00:10'
8     SELECT * FROM TEACHERS WHERE TID='300000000'
9 END
10 ROLLBACK TRAN
```

结果

消息

	tid	tname	email	salary
1	3000000000	AA	BBB@163.COM	5000

可以发现，两次查

经被删除，所以发

可以发现，两次查询结果均相同，但事实上，结果显示的记录已经被删除，所以发生了“幻象读”的问题。

代码6如下：

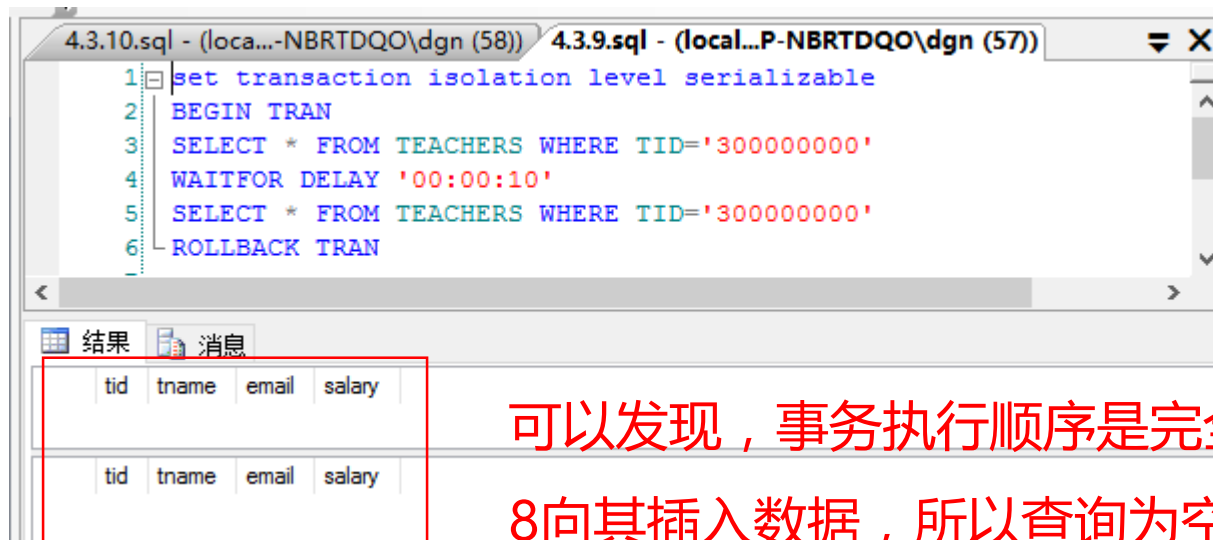
```
4.3.8.sql - (local...P-NBRTDQO\dgn (60)) 4.3.7.sql - (local...P-NBRTDQO\dgn (59))
1 set transaction isolation level repeatable read
2 delete FROM TEACHERS WHERE TID='300000000'
3
```

实验示例

6. 隔离级别设置为“可串行化”（SERIALIZABLE），这是事务隔离的最高级别，事务之间完全隔离，在该级别可以保证并发事务均是可串行的。“可串行化”可以防止用户在事务完成之前更新或插入数据。

例如：在执行代码7两次查询的过程中，执行代码8插入数据。

代码7如下：



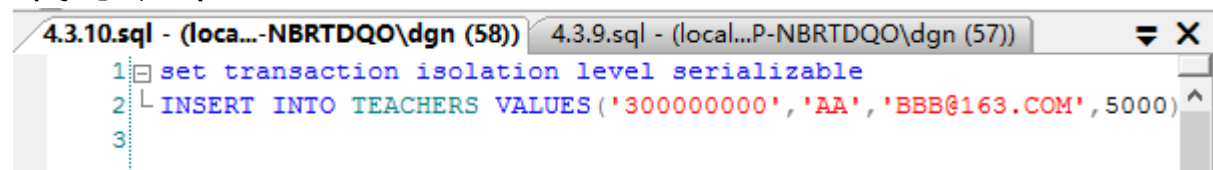
The screenshot shows a SQL Developer window with two tabs. The active tab is '4.3.10.sql - (local...-NBRTDQO\dgn (58))'. It contains the following SQL code:

```
1 set transaction isolation level serializable
2 BEGIN TRAN
3 SELECT * FROM TEACHERS WHERE TID='3000000000'
4 WAITFOR DELAY '00:00:10'
5 SELECT * FROM TEACHERS WHERE TID='3000000000'
6 ROLLBACK TRAN
```

Below the code editor, the 'Results' tab is selected, showing a table with the following columns: tid, tname, email, salary. The table is empty, indicating that no data was inserted during the execution of the transaction.

可以发现，事务执行顺序是完全串行的，事务7在执行过程中防止代码8向其插入数据，所以查询为空。

代码8如下：



The screenshot shows a SQL Developer window with two tabs. The active tab is '4.3.10.sql - (local...-NBRTDQO\dgn (58))'. It contains the following SQL code:

```
1 set transaction isolation level serializable
2 INSERT INTO TEACHERS VALUES ('3000000000', 'AA', 'BBB@163.COM', 5000)
3
```

- 练习

以下练习均在school数据库中students表上进行。

- (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。