《数据库系统实验》
实验报告

题目	实验 12
姓名	宋渝杰
学号	18340146
班级	计科六班

一、实验环境：

win10+MySQL 8.0

二、实验内容与完成情况：

第 12 章表 12-7/8/9/10：

先创建数据库和表、输入数据，代码如下：

```
create database jxgl;
use jxgl;

#  建表
create table student
(sno varchar(7),
sname varchar(5),
sage numeric(2,0),
ssex varchar(1),
sdept varchar(5),
primary key(sno));

create table course
(cno varchar(4),
cname varchar(6),
cpno varchar(4),
ccredit numeric(1,0),
primary key(cno));

create table sc
(sno varchar(7),
cno varchar(4),
grade float(3),
primary key(sno,cno),
foreign key(sno) references student(sno),
foreign key(cno) references course(cno));

# 加入课本数据
insert into student values('2005001','钱横',18,'男','Cs');
insert into student values('2005002','王林',19,'女','Cs');
insert into student values('2005003','李民',20,'男','Is');
insert into student values('2005004','赵欣然',16,'女','Ma');
insert into course values('1','数据库系统','5',4);
insert into course values('2','数学分析',null,2);
insert into course values('3','信息系统导论','1',3);
insert into course values('4','操作系统原理','6',3);
insert into course values('5','数据结构','7',4);
insert into course values('6','数据处理基础',null,4);
insert into course values('7','C 语言','6',3);
insert into sc values('2005001','1',87);
insert into sc values('2005001','2',67);
insert into sc values('2005001','3',90);
insert into sc values('2005002','2',95);
insert into sc values('2005003','3',88);
```

表 12-7：

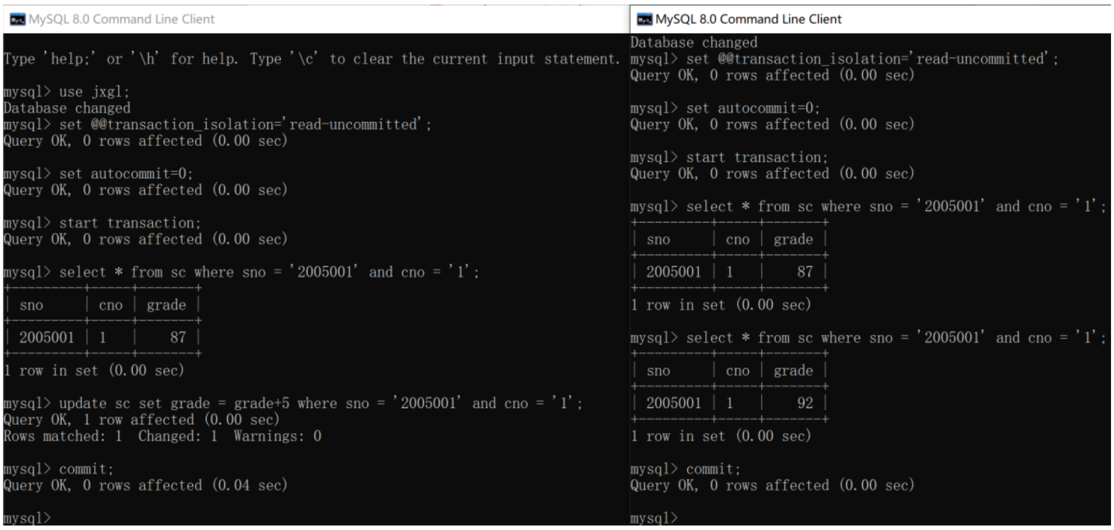
Session1：

```
use jxgl;
set @@transaction_isolation='read-uncommitted';
set autocommit=0;
start transaction;
select * from sc where sno = '2005001' and cno = '1';
update sc set grade = grade+5 where sno = '2005001' and cno = '1';
commit;
```

Session2：

```
use jxgl;
set @@transaction_isolation='read-uncommitted';
set autocommit=0;
start transaction;
select * from sc where sno = '2005001' and cno = '1';
select * from sc where sno = '2005001' and cno = '1';
commit;
```

结果：



```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use jxgl;
Database changed
mysql> set @@transaction_isolation='read-uncommitted';
Query OK, 0 rows affected (0.00 sec)

mysql> set autocommit=0;
Query OK, 0 rows affected (0.00 sec)

mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from sc where sno = '2005001' and cno = '1';
+---------+-----+-------+
| sno     | cno | grade |
+---------+-----+-------+
| 2005001 | 1   |    87 |
+---------+-----+-------+
1 row in set (0.00 sec)

mysql> update sc set grade = grade+5 where sno = '2005001' and cno = '1';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> commit;
Query OK, 0 rows affected (0.04 sec)

mysql>
```

```
Database changed
mysql> set @@transaction_isolation='read-uncommitted';
Query OK, 0 rows affected (0.00 sec)

mysql> set autocommit=0;
Query OK, 0 rows affected (0.00 sec)

mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from sc where sno = '2005001' and cno = '1';
+---------+-----+-------+
| sno     | cno | grade |
+---------+-----+-------+
| 2005001 | 1   |    87 |
+---------+-----+-------+
1 row in set (0.00 sec)

mysql> select * from sc where sno = '2005001' and cno = '1';
+---------+-----+-------+
| sno     | cno | grade |
+---------+-----+-------+
| 2005001 | 1   |    92 |
+---------+-----+-------+
1 row in set (0.00 sec)

mysql> commit;
Query OK, 0 rows affected (0.00 sec)

mysql>
```

左边为事务 1 的结果，右边为事务 2 的结果。
可以看出，出现了不可重复读的情况，事务 2 两次读取的数值不一样了

表 12-8：

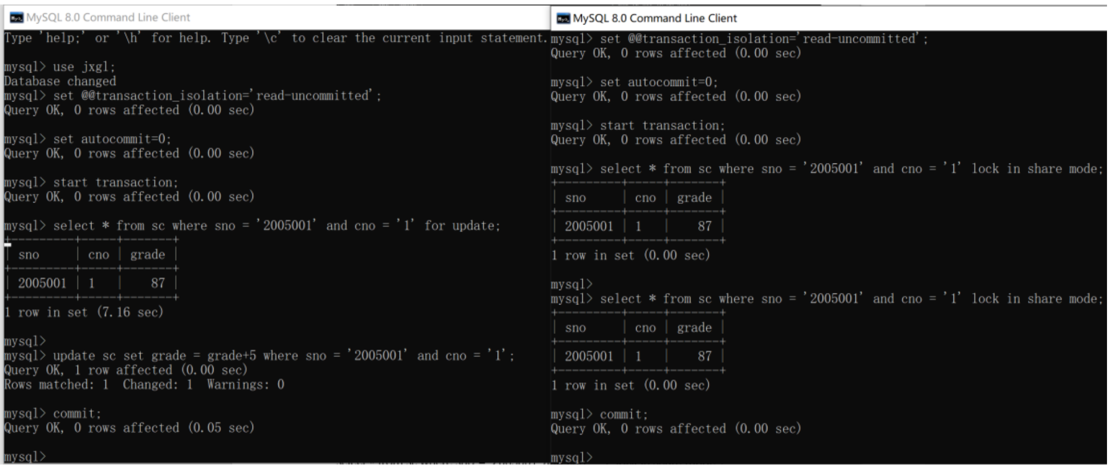
Session1：

```
use jxgl;
set @@transaction_isolation='read-uncommitted';
set autocommit=0;
start transaction;
select * from sc where sno = '2005001' and cno = '1' for update;
update sc set grade = grade+5 where sno = '2005001' and cno = '1';
commit;
```

Session2：

```
use jxgl;
set @@transaction_isolation='read-uncommitted';
set autocommit=0;
start transaction;
select * from sc where sno = '2005001' and cno = '1' lock in share mode;
select * from sc where sno = '2005001' and cno = '1' lock in share mode;
commit;
```

结果：



```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use jxgl;
Database changed
mysql> set @@transaction_isolation='read-uncommitted';
Query OK, 0 rows affected (0.00 sec)

mysql> set autocommit=0;
Query OK, 0 rows affected (0.00 sec)

mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from sc where sno = '2005001' and cno = '1' for update;
+---------+-----+-------+
| sno     | cno | grade |
+---------+-----+-------+
| 2005001 | 1   |    87 |
+---------+-----+-------+
1 row in set (7.16 sec)

mysql>
mysql> update sc set grade = grade+5 where sno = '2005001' and cno = '1';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> commit;
Query OK, 0 rows affected (0.05 sec)

mysql>
```

```
mysql> set @@transaction_isolation='read-uncommitted';
Query OK, 0 rows affected (0.00 sec)

mysql> set autocommit=0;
Query OK, 0 rows affected (0.00 sec)

mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from sc where sno = '2005001' and cno = '1' lock in share mode;
+---------+-----+-------+
| sno     | cno | grade |
+---------+-----+-------+
| 2005001 | 1   |    87 |
+---------+-----+-------+
1 row in set (0.00 sec)

mysql>
mysql> select * from sc where sno = '2005001' and cno = '1' lock in share mode;
+---------+-----+-------+
| sno     | cno | grade |
+---------+-----+-------+
| 2005001 | 1   |    87 |
+---------+-----+-------+
1 row in set (0.00 sec)

mysql> commit;
Query OK, 0 rows affected (0.00 sec)

mysql>
```

左边为事务 1 的结果，右边为事务 2 的结果。

可以看出，使用了共享锁之后，事务 1 的更新操作会等待事务 2 的第二次查询解锁之后再进行，因此事务 1 的更新操作会被固定限制在事务 2 第二次查询之后，确保了事务 2 两次查询结果的相同。

表 12-9：

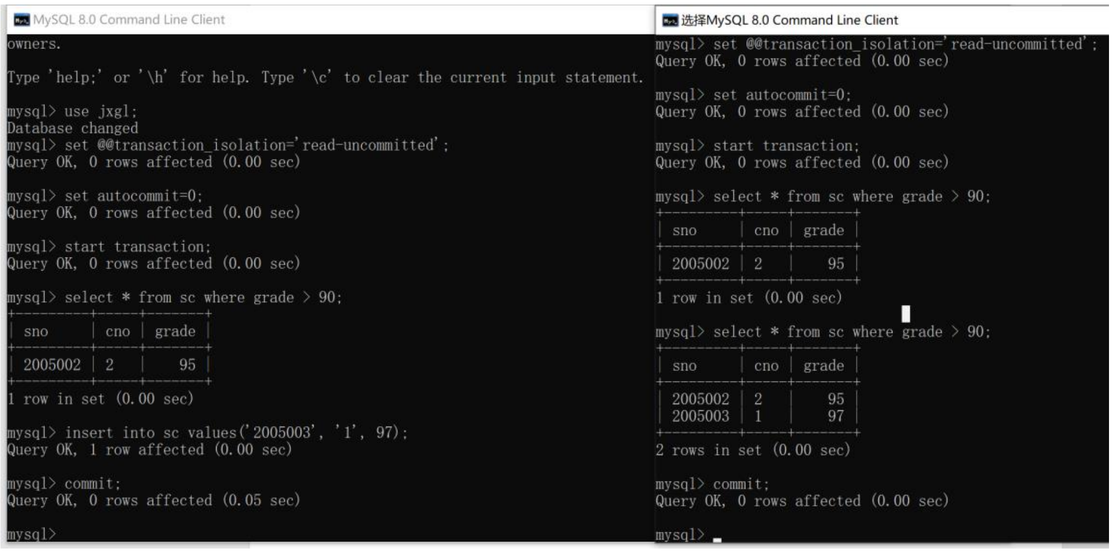
Session1：

```
use jxgl;
set @@transaction_isolation='read-uncommitted';
set autocommit=0;
start transaction;
select * from sc where grade > 90;
insert into sc values('2005003', '1', 97);
commit;
```

Session2：

```
use jxgl;
set @@transaction_isolation='read-uncommitted';
set autocommit=0;
start transaction;
select * from sc where grade > 90;
select * from sc where grade > 90;
commit;
```

结果：


MySQL 8.0 Command Line Client
选择MySQL 8.0 Command Line Client

```
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use jxgl;
Database changed
mysql> set @@transaction_isolation='read-uncommitted';
Query OK, 0 rows affected (0.00 sec)

mysql> set autocommit=0;
Query OK, 0 rows affected (0.00 sec)

mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from sc where grade > 90;
+---------+-----+-------+
| sno     | cno | grade |
+---------+-----+-------+
| 2005002 | 2   |    95 |
+---------+-----+-------+
1 row in set (0.00 sec)

mysql> insert into sc values('2005003', '1', 97);
Query OK, 1 row affected (0.00 sec)

mysql> commit;
Query OK, 0 rows affected (0.05 sec)

mysql>
```

```
mysql> set @@transaction_isolation='read-uncommitted';
Query OK, 0 rows affected (0.00 sec)

mysql> set autocommit=0;
Query OK, 0 rows affected (0.00 sec)

mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from sc where grade > 90;
+---------+-----+-------+
| sno     | cno | grade |
+---------+-----+-------+
| 2005002 | 2   |    95 |
+---------+-----+-------+
1 row in set (0.00 sec)

mysql> select * from sc where grade > 90;
+---------+-----+-------+
| sno     | cno | grade |
+---------+-----+-------+
| 2005002 | 2   |    95 |
| 2005003 | 1   |    97 |
+---------+-----+-------+
2 rows in set (0.00 sec)

mysql> commit;
Query OK, 0 rows affected (0.00 sec)

mysql> _
```

左边为事务 1 的结果，右边为事务 2 的结果。

事务 2 在第一次 select 访问数据之后，事务 1 插入一行数据，所以事务 2 第二次 select 访问数据的结果和第一次不一样，即产生了幻影现象。

表 12-10：

Session1：

use jxgl;

set @@transaction_isolation='read-uncommitted';

set autocommit=0;

start transaction;

select * from sc where grade > 90 for update;

insert into sc values('2005003', '1', 97);

commit;

Session2：

use jxgl;

set @@transaction_isolation='read-uncommitted';

set autocommit=0;

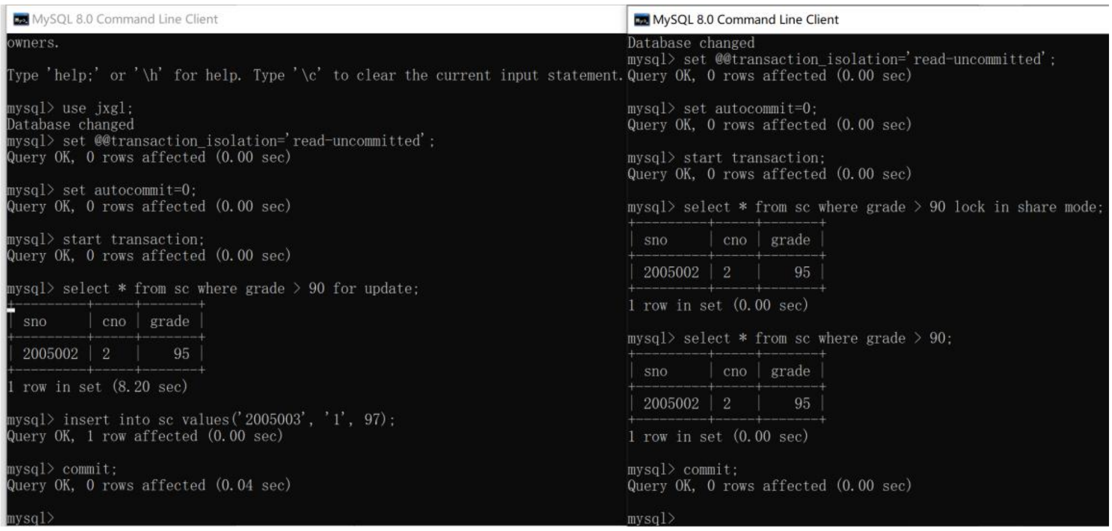
start transaction;

select * from sc where grade > 90 lock in share mode;

select * from sc where grade > 90;

commit;

结果：



```
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use jxgl;
Database changed
mysql> set @@transaction_isolation='read-uncommitted';
Query OK, 0 rows affected (0.00 sec)

mysql> set autocommit=0;
Query OK, 0 rows affected (0.00 sec)

mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from sc where grade > 90 for update;
+---------+-----+-------+
| sno     | cno | grade |
+---------+-----+-------+
| 2005002 | 2   |    95 |
+---------+-----+-------+
1 row in set (8.20 sec)

mysql> insert into sc values('2005003', '1', 97);
Query OK, 1 row affected (0.00 sec)

mysql> commit;
Query OK, 0 rows affected (0.04 sec)

mysql>
```

```
Database changed
mysql> set @@transaction_isolation='read-uncommitted';
Query OK, 0 rows affected (0.00 sec)

mysql> set autocommit=0;
Query OK, 0 rows affected (0.00 sec)

mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from sc where grade > 90 lock in share mode;
+---------+-----+-------+
| sno     | cno | grade |
+---------+-----+-------+
| 2005002 | 2   |    95 |
+---------+-----+-------+
1 row in set (0.00 sec)

mysql> select * from sc where grade > 90;
+---------+-----+-------+
| sno     | cno | grade |
+---------+-----+-------+
| 2005002 | 2   |    95 |
+---------+-----+-------+
1 row in set (0.00 sec)

mysql> commit;
Query OK, 0 rows affected (0.00 sec)

mysql>
```

左边为事务 1 的结果，右边为事务 2 的结果。

可以看出，使用了共享锁之后，事务 1 的插入操作会等待事务 2 的第二次查询解锁之后再进行，因此事务 1 的插入操作会被固定限制在事务 2 第二次查询之后，确保了事务 2 两次查询结果的相同。

三、实验心得：

本次实验内容不多，而且内容就是复现实验课本的内容，因此也基本没有遇到困难，只需要了解一下复现的实验原理即可~