北京邮电大学　计算机学院

《数据库系统原理》实验报告

姓名　　王睿嘉

学号 2015211906

班级 2015211307

实验六 数据查询分析

一、 实验内容和环境描述

1. 实验目的

1） 通过对不同查询语句执行情况的分析，巩固和加深对查询和查询优化相关理论知识的理解，提高优化数据库系统的实践能力；

2） 熟悉了解 MySQL 中查询分析器的使用，进一步提高编写复杂查询程序的能力。

2. 实验内容

1） 索引对查询的影响

① 对结果集只有一个元组的查询，分三种情况执行：

不建立索引

（学号上）建立非聚集索引

（学号上）建立聚集索引

对执行步骤和结果进行分析比较；

② 对结果集中有多个元组的查询，分类似①的三种情况进行比较；

③ 对查询条件为一个连续范围的查询，分类似①的三种情况进行比较，注意系统处理的选择；

④ 索引代价

在有索引和无索引的情况下，分别插入数据，比较执行效率。

2） 分析比较相同查询功能的查询语句

① select avg(grade)
from sc
group by cno
having cno=’C01’

select avg(grade)
from sc
where cno=’C01’

有和没有 group by，比较其查询效率，并分析；

② select sno,sname,bdate
from student as s1
where bdate=
(select max(bdate)
from student as s2
where s1.dept=s2.dept
)

create table tmp as(
select dept,max(bdate) as maxBdate
from student

```
        group by dept);
    select sno,sname,bdate
    from student,tmp
    where student.bdate=tmp.maxBdate and tmp.dept=student.dept
    drop table tmp
```

重写后的查询一定比原始查询更优吗？通过执行分析结果；

③ 对下面两个查询进行比较

```
select sname,bdate
from student
where dept!=’计算机’ and bdate>all
(select bdate
from student
where dept=’计算机’
)

select sname,bdate
from student
where dept!=’计算机’ and bdate>
(select max(bdate)
from student
where dept=’计算机’
);
```

3） 查询优化

除了建立适当索引，对 SQL 语句重写外，还有其他手段进行查询调优，例如调整缓冲区大小、事先建立视图等。设计实现下列查询，使之运行效率最高。

写出查询形式及调优过程，并说明最优情况下的运行时间：

① 查找选修了每一门课的学生；

② 查找至少选修了数据库原理和操作系统的学生的学号。

3. 实验环境

数据库系统：5.7.21 MySQL Community Server（GPL）

二、 实验结论和心得

1. 实验结论

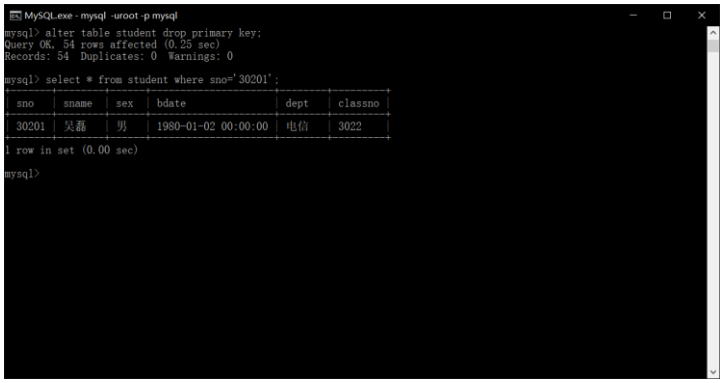
1.1 对结果集只有一个元组的查询，分三种情况执行

以查询一个具体学生的信息为例：

1）不建立索引

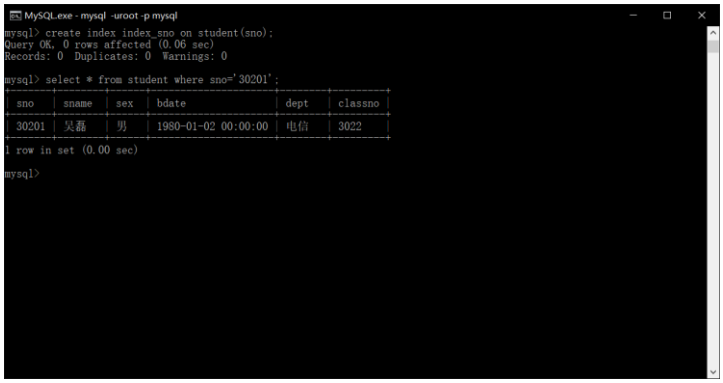
创建表时，由于设置了 primary key，MySQL 将自动生成主键索引，即聚集索引。

为了测试在不建立索引的情况下，查询学生信息的执行时间，首先将主键索引删除。然后查询学号为30201的学生的信息：



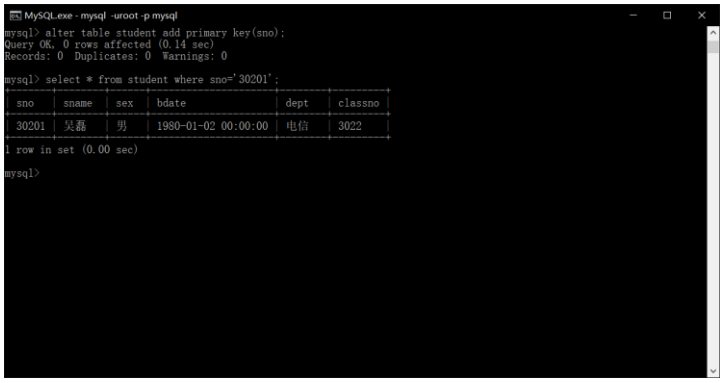
2)（学号上）建立非聚集索引

先建立非聚集索引，然后进行同样的查询操作：

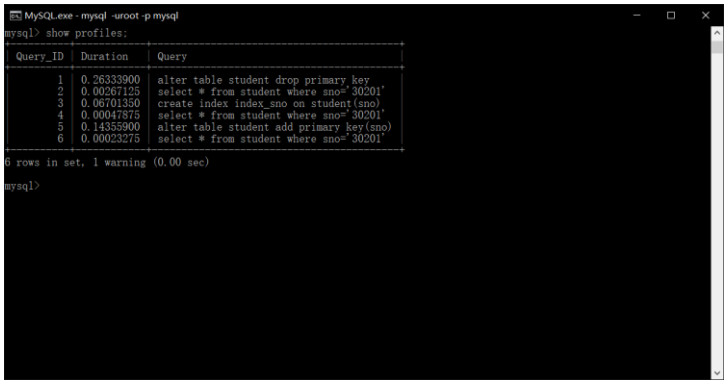


3)（学号上）建立聚集索引

重新设置该表的主键，即聚集索引，然后进行同样的查询操作：



三种查询语句的时间对比如下：



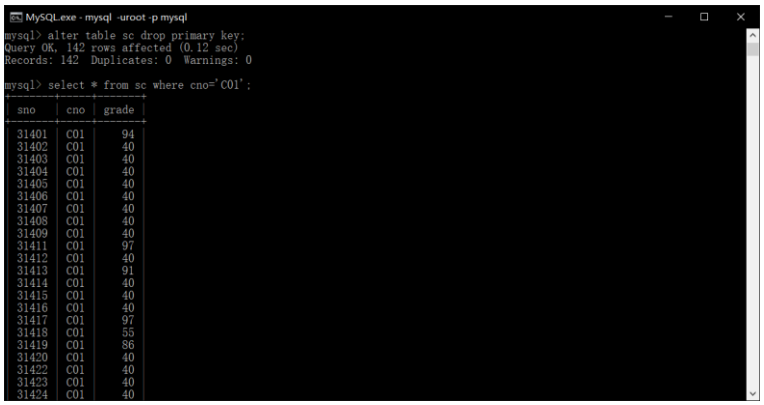
查询方法	时间（单位：秒）	比较
不建立索引	0.00267125	时间最长
（学号上）建立非聚集索引	0.00047875	时间较短
（学号上）建立聚集索引	0.00023275	时间最短

1.2 对结果集中有多个元组的查询，分三种情况执行

以查询某课程的成绩表为例：

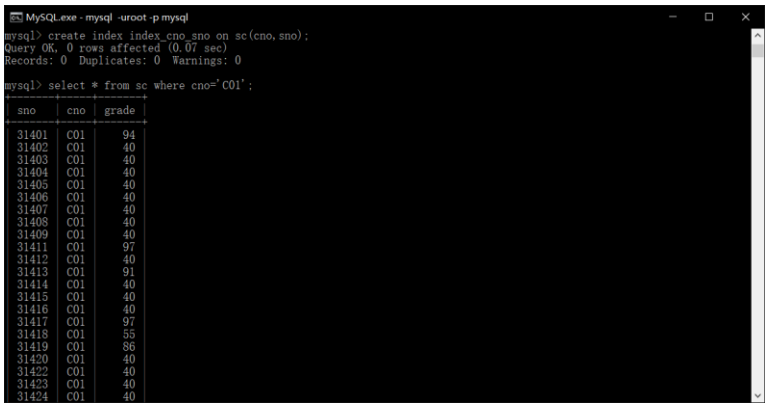
1）不建立索引

为了测试在不建立索引的情况下，查询成绩表的执行时间，首先将主键索引删除。然后查询课程号为C01 的课程的成绩表信息：



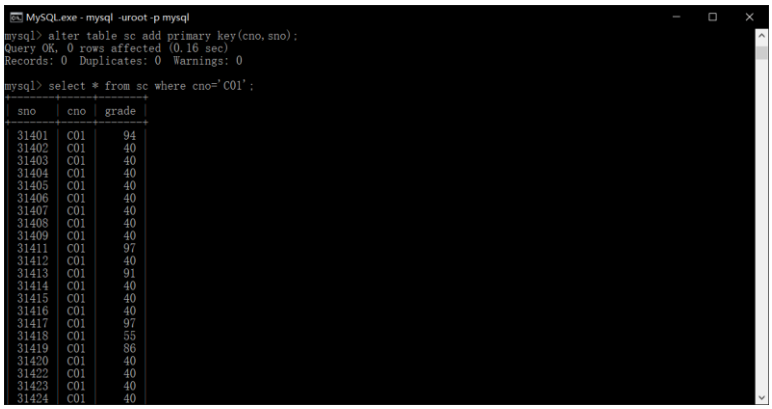
2）（课程号，学号上）建立非聚集索引

先建立非聚集索引，然后进行同样的查询操作：

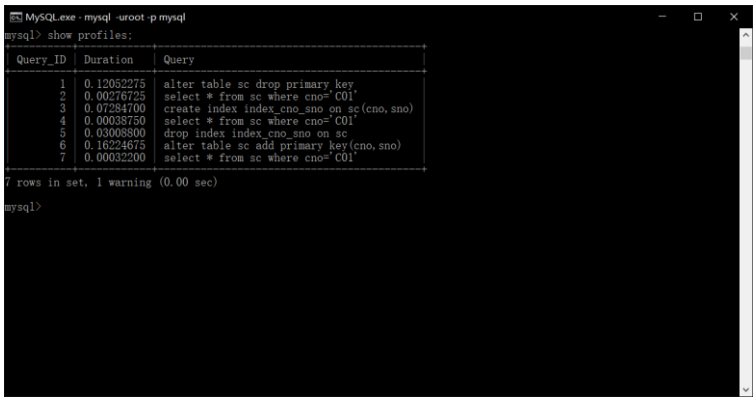


3）（课程号，学号上）建立聚集索引

重新设置该表的主键，即聚集索引，然后进行同样的查询操作：



三种查询语句的时间对比如下：



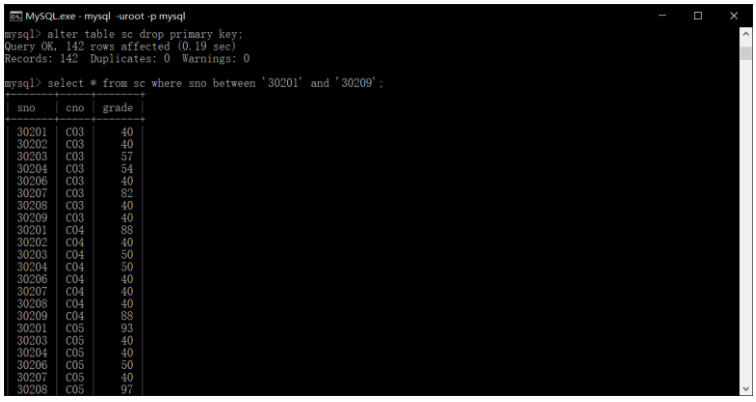
查询方法	时间（单位：秒）	比较
不建立索引	0.00276725	时间最长
（课程号，学号上）建立非聚集索引	0.00038750	时间较短
（课程号，学号上）建立聚集索引	0.00032200	时间最短

1.3 对查询条件为一个连续范围的查询，分三种情况执行

以查询学号在某范围内的学生选课情况为例：

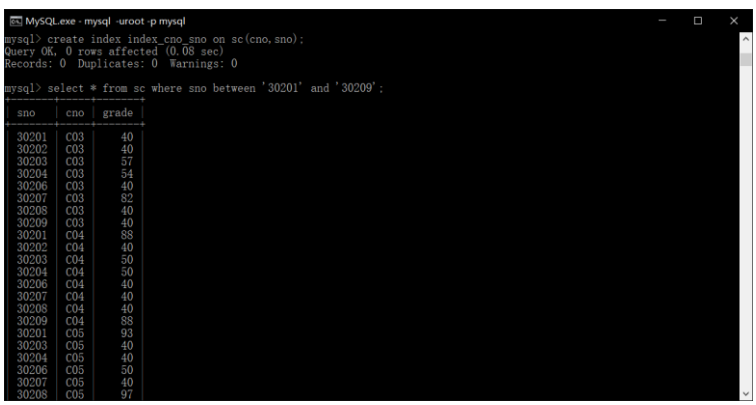
1）不建立索引

为了测试在不建立索引的情况下，查询学生选课情况的执行时间，首先将主键索引删除。然后查询学号在 30201-30209 的学生的选课情况：



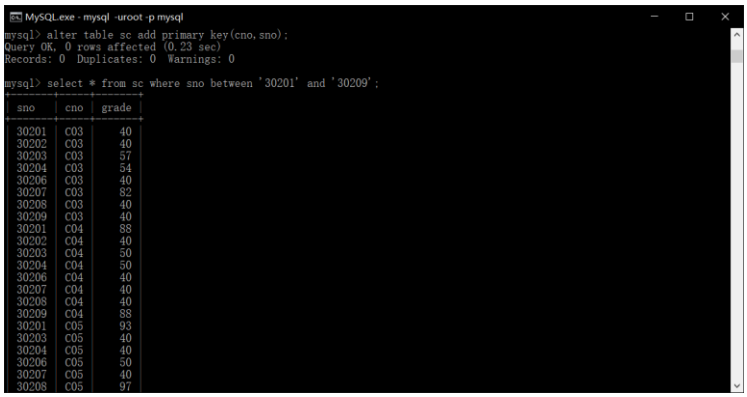
2）（课程号，学号上）建立非聚集索引

先建立非聚集索引，然后进行同样的查询操作：



3）（课程号，学号上）建立聚集索引

重新设置该表的主键，即聚集索引，然后进行同样的查询操作：



三种查询语句的时间对比如下：

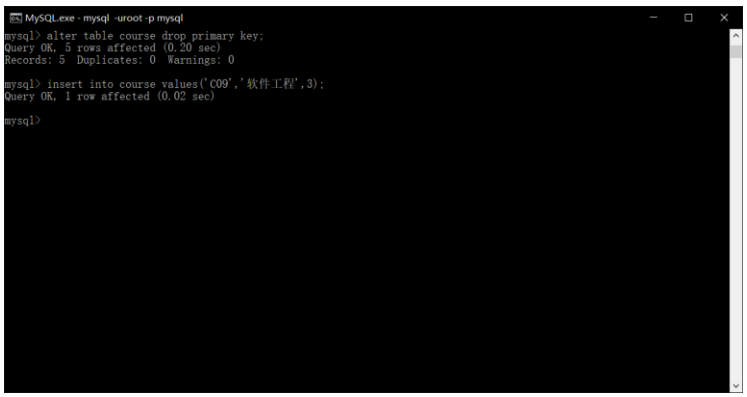
查询方法	时间（单位：秒）	比较
不建立索引	0.10966125	时间最长
（课程号，学号上）建立非聚集索引	0.00032800	时间较短
（课程号，学号上）建立聚集索引	0.00032300	时间最短

1.4 索引代价

以在课程信息表 course 中插入数据为例：

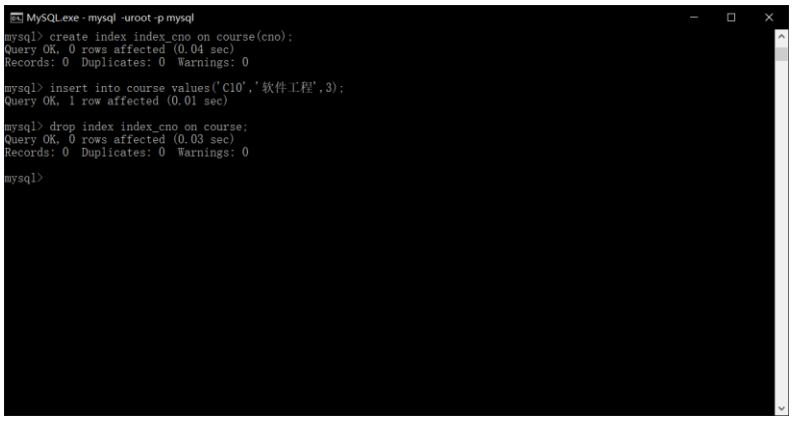
1）不建立索引

为了测试在不建立索引的情况下，插入课程信息的执行时间，首先将主键索引删除：



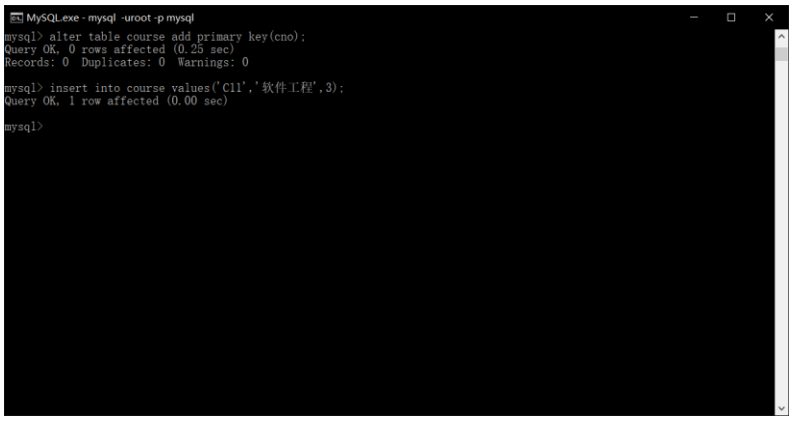
2）（课程号上）建立非聚集索引

先建立非聚集索引，然后进行同样的插入操作：

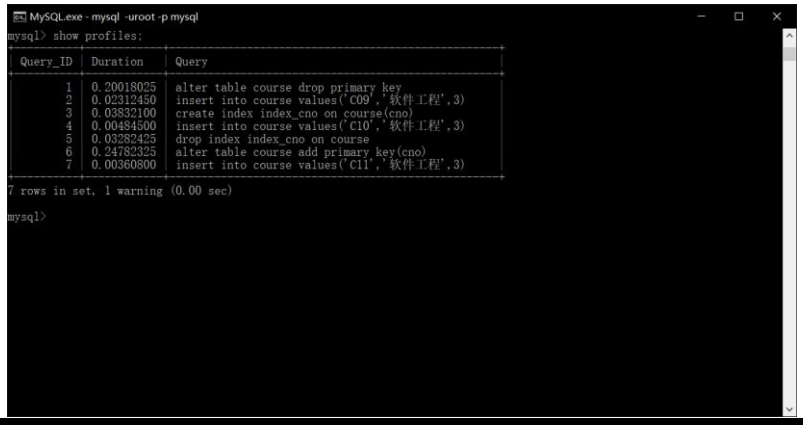


3)（课程号上）建立聚集索引

重新设置该表的主键，即聚集索引，然后进行同样的插入操作：



三种查询语句的时间对比如下：



查询方法	时间（单位：秒）	比较
不建立索引	0.02312450	时间最长
（课程号上）建立非聚集索引	0.00484500	时间较短
（课程号上）建立聚集索引	0.00360800	时间最短

2.1 有和没有 group by，比较其查询效率

1）select avg(grade)

from sc

group by cno

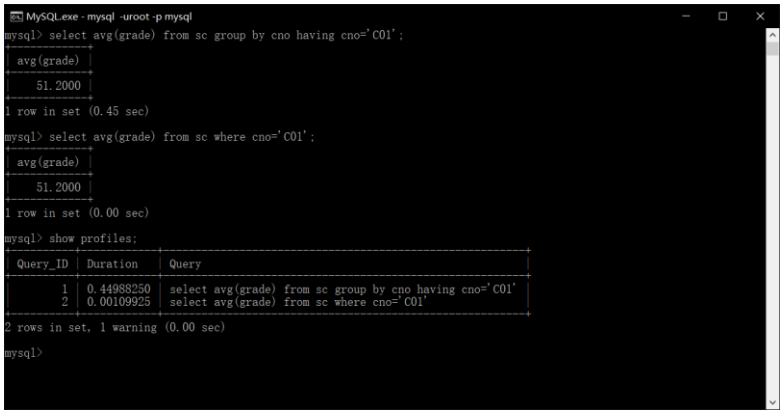
having cno=’C01’

2）select avg(grade)

from sc

where cno=’C01’

运行结果相同，但时间相差极大，如下图所示：



查询语句	时间（单位：秒）	比较
有 group by	0.44988250	时间长
没有 group by	0.00109925	时间短

2.2 重写后的查询一定比原始查询更优吗？

1）select sno,sname,bdate

from student as s1

where bdate=

(select max(bdate)

from student as s2

where s1.dept=s2.dept

)

2）create table tmp as(

select dept,max(bdate) as maxBdate

from student

group by dept);

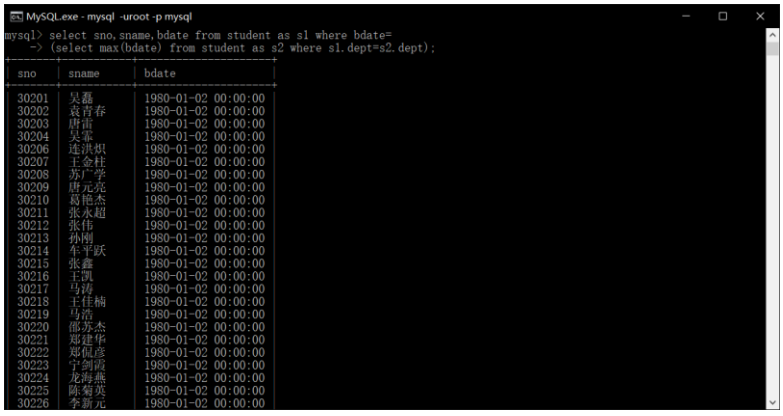
select sno,sname,bdate

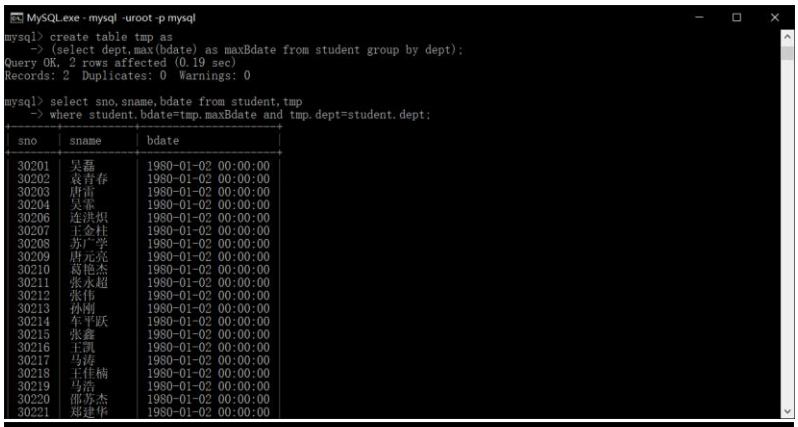
from student,tmp

where student.bdate=tmp.maxBdate and tmp.dept=student.dept

drop table tmp

运行结果相同，但时间相差极大，如下图所示：





第一种方法只包含一条语句，执行时间为 0.00556125s；

第二种方法包含三条语句，执行时间分别为 0.18543825s、0.00048100s 和 0.02175525s。其中，单纯的查询时间为 0.00048100s。

重写后，单纯查询会比之前的用时少很多，甚至相差一个数量级。但是，重写后的查询需要创建表和删除表。如果没有对这个新表进行多次相关查询，那么额外的操作时间会使得这个优化得不偿失。

查询语句	时间（单位：秒）
直接查询	0.00556125
建表查询	0.18543825（创建表）
	0.00048100（查询）
	0.02175525（删除表）

2.3 对下面两个查询进行比较

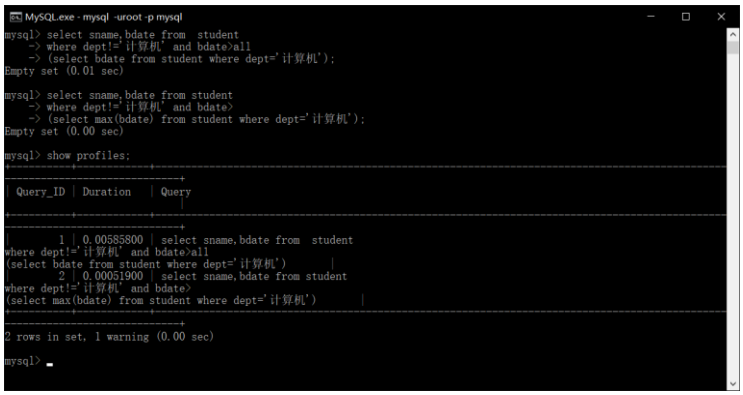
1）select sname,bdate
from student
where dept!=’计算机’ and bdate>all
(select bdate
from student
where dept=’计算机’
)

2）select sname,bdate
from student
where dept!=’计算机’ and bdate>
(select max(bdate)
from student

where dept=’计算机’

);

运行结果相同，如下图所示：



查询语句	时间（单位：秒）	比较
直接比较	0.00585800	时间长
选择比较	0.00051900	时间短

3.1 查找选修了每一门课的学生

1）直接查询

```
select sname
from student
where sno in(
select sno
from sc
group by sno
having count(*)=(
select count(*)
from course))
```

2）创建 table

```
create table tmp as(
select distinct sno as sno1,count(cno) as amount
from sc
group by sno);
select sname
from tmp,student
where tmp.amount>=5 and tmp.sno1=student.sno
drop table tmp
```

3）创建 view

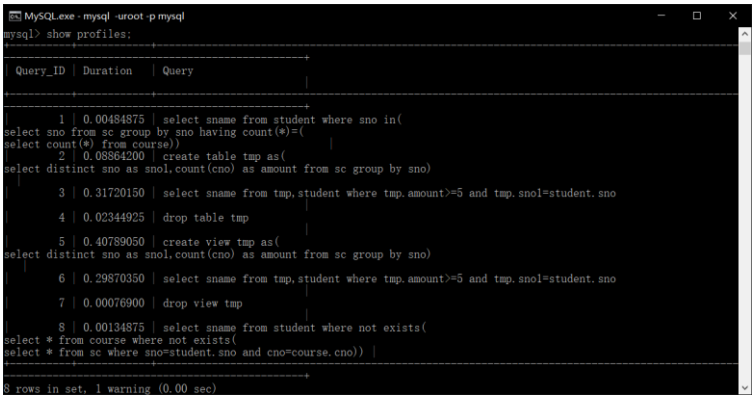
```
create view tmp as(
select distinct sno as sno1,count(cno) as amount
from sc
group by sno);
select sname
from tmp,student
```

```
where tmp.amount>=5 and tmp.sno1=student.sno;
drop view tmp
```

4）嵌套子查询

```
select sname
from student
where not exists(
select *
from course
where not exits(
select *
from sc
where student.sno=course.cno))
```

运行结果相同，时间如下：



方法	时间（单位：秒）	比较
直接查询	0.00484875	时间较短
创建 table	0.31720150	时间最长
创建 view	0.29870350	时间较长
嵌套子查询	0.00134875	时间最短

3.2 查找至少选修了数据库原理和操作系统的学生的学号

1）创建 table

```
create table tmp as(
select sc.sno as sno,cname
from sc,course
where sc.cno=course.cno and (course.cname='数据库原理' or course.cname='操作系统'));
select sno
from tmp
group by sno
having count(*)>=2;
drop table tmp;
```

2）创建 view

```
create view tmp as(
select sc.sno as sno,cname
from sc,course
```

```
where sc.cno=course.cno and (course.cname='数据库原理'or course.cname='操作系统'));
select sno
from tmp
group by sno
having count(*)>=2;
drop view tmp
```

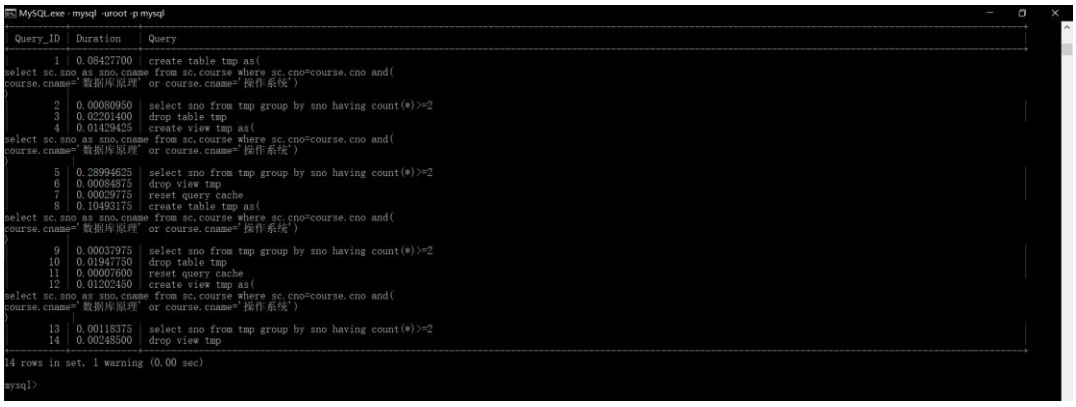
3） 清空缓存后，创建 table

```
reset query cache;
create table tmp as(
select sc.sno as sno,cname
from sc,course
where sc.cno=course.cno and (course.cname='数据库原理'or course.cname='操作系统'));
select sno
from tmp
group by sno
having count(*)>=2;
drop table tmp;
```

2）清空缓存后，创建 view

```
reset query cache;
create view tmp as(
select sc.sno as sno,cname
from sc,course
where sc.cno=course.cno and (course.cname='数据库原理'or course.cname='操作系统'));
select sno
from tmp
group by sno
having count(*)>=2;
drop view tmp
```

运行结果相同，时间如下：



	创建 table（单位：秒）	创建 view（单位：秒）	比较
未清空缓存	0.00080950	0.28994625	创建 table 较快
清空缓存	0.00037975	0.00118375	创建 table 较快
比较	未清空缓存较快	清空缓存较快	

2. 实验心得

在本次实验中，遇到的主要问题有以下三点：

① 时间精度。起初，意图通过直接显示的执行时间进行不同查询方式的比较。但由于其精度不够，无法得出结论。最终，查阅网上资料，开启 profiling 参数，该问题得到解决；

② 执行时间与理论不符。在不建立索引、建立非聚集索引和建立聚集索引三种方式的对比实验中，实验结果与预期结果有时相同，有时相反，每次查询的执行时间均存在一定偏差。经分析：首先，查询语句会存在缓存，从而影响之后的查询；其次，查询的执行时间还与其他因素有关，如当时 CPU 的运行状态等，存在很大的不确定性；

③ 实验手册某些语句与 MySQL 的兼容性。MySQL 不支持 select...into...的语法，因而必须新建表来实现相同的功能。

实践出真知，本次数据查询分析实验是对课堂和书本所学知识的补充。通过自己动手、亲力亲为编写 SQL 语句，加深了对查询优化的理解和记忆，收获颇丰。