

《数据库系统原理》实验报告

实验名称 数据查询分析实验

班 级 2015211307

学 号 2017526019

姓 名 刘禾子

实验六 数据查询分析实验

■ 实验目的

通过对不同情况下查询语句的执行分析，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，熟悉了解 MySQL 中查询分析器的使用，并进一步提高编写复杂查询的 SQL 程序的能力

■ 实验平台及环境

Windows10、MySQL 5.7、MySQL Workbench 6.3CE

■ 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进行执行（例如查询一个具体学生的信息）：

不建立索引，（学号上）建立非聚集索引，（学号上）建立聚集索引。用查询分析器的执行步骤和结果对执行进行分析比较。

(2) 对结果集中有多个元组的查询（例如查看某门课程的成绩表）分类似（1）的三种情况进行执行比较。

(3) 对查询条件为一个连续的范围的查询（例如查看学号在某个范围内的学生的选课情况）分类似（1）的三种情况进行执行比较，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插入数据（例如在选课情况表 SC 上插入数据），比较插入的执行效率。

2. 对相同查询功能不同查询语句的执行比较分析

```
(1) group by
    select avg(grade)
    from sc
    group by cno
    having cno =100

    select avg(grade)
    from sc
    where cno = 100
```

有和没有group by，比较其查询效率，并分析。

```

(2) select sno, sname, age
    from student s1
   where age =
      (select max(age)
       from student s2
       where s1.dept = s2.dept)
  另一个:
  select dept , max(age) as maxAge into tmp
 from student
group by dept;
select sno, sname , age
 from student, tmp
where student.age = tmp.maxAge 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, age
from student
where dept != 10 and age > all
(select age
 from student
 where dept = 10)

```

另:

```

select sname ,age
from student
where dept != 10 and age >
( select max(age)
 from student
 where dept = 10)

```

3. 查询优化

除了建立适当索引，对SQL 语句重写外，还有其他手段来进行查询调优，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。写出你的查询形式，以及调优过程；并说

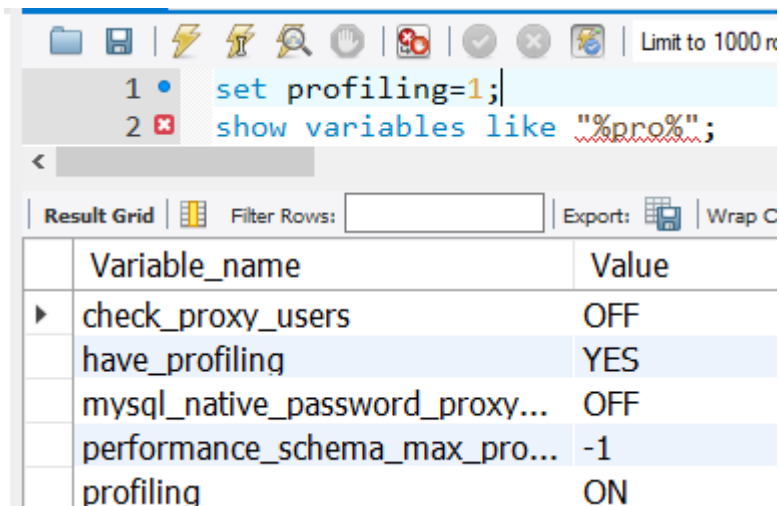
明最优情况下的运行时间。

(1) 查找选修了每一门课的学生。

(2) 查找至少选修课程数据库原理和操作系统的学生的学号。

■ 实验步骤及结果分析

由于涉及到比较查询时间问题需要设置 profiling 参数,以便后期打开查询分析器进行查询时间性能的比较如下图:



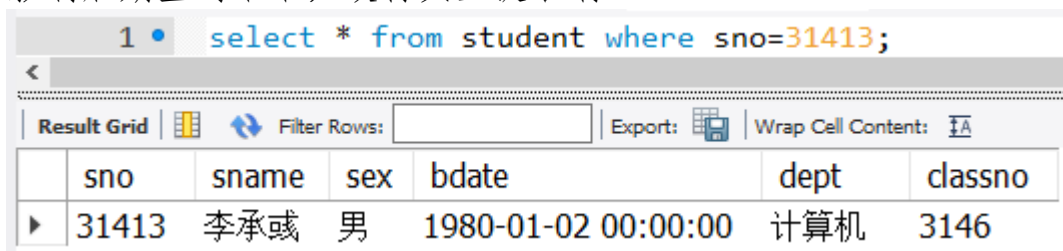
```
1 • set profiling=1;
2 ✖ show variables like "%pro%";
```

Variable_name	Value
check_proxy_users	OFF
have_profiling	YES
mysql_native_password_proxy...	OFF
performance_schema_max_pro...	-1
profiling	ON

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进行执行 (例如查询一个具体学生的信息):

由于 student 表中 sno 初始设置为主键,即已经存在聚集索引会影响后期查询结果,故将其主键取消



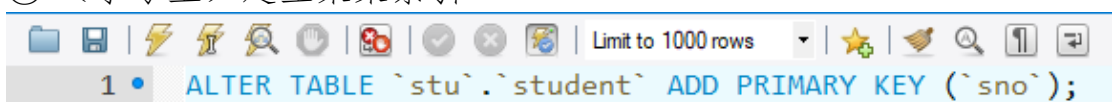
```
1 • select * from student where sno=31413;
```

sno	sname	sex	bdate	dept	classno
31413	李承或	男	1980-01-02 00:00:00	计算机	3146

① 不建立索引

② (学号上) 建立非聚集索引

② (学号上) 建立聚集索引



```
1 • ALTER TABLE `stu`.`student` ADD PRIMARY KEY (`sno`);
```

Query_ID	Duration	Query
1	0.00006650	SHOW WARNINGS
2	0.00037575	select * from student where sno='31413' LIMIT 0, ...
3	0.02913150	create index stu_index on student(sno)
4	0.00025350	select * from student where sno='31413' LIMIT 0, ...
5	0.01567125	ALTER TABLE `stu`.`student` DROP INDEX `stu...
6	0.10971975	ALTER TABLE `stu`.`student` ADD PRIMARY KEY...
7	0.00018325	select * from student where sno='31413' LIMIT 0, ...








对比如下表：

索引方式	执行时间	比较
不建立索引	0.00037575	时间最长
非聚集索引	0.00025350	时间较短
聚集索引	0.00018325	时间最短

(2) 对结果集中有多个元组的查询（如查看某门课程成绩表）

```
7 • select * from sc where cno='C01';
```

<

Result Grid   Filter Rows: | Edit:    | Export/Import:   | Wrap Cell Cont

	sno	cno	grade
▶	31401	C01	94
▶	31402	C01	40
	31403	C01	40
	31404	C01	40
	31405	C01	40
	31406	C01	40

- ① 不建立索引
- ② 建立非聚集索引
- ③ 建立聚集索引

三种情况执行情况如下表：

16	0.10545300	ALTER TABLE `stu`.`sc` DROP PRIMARY KEY
17	0.00089025	select * from sc where cno='C01' LIMIT 0, 1000
18	0.03002850	create index stu_cno_index on sc(sno,cno)
19	0.00025475	select * from sc where cno='C01' LIMIT 0, 1000
20	0.03527525	ALTER TABLE `stu`.`sc` DROP INDEX `stu_cno_ind
21	0.12832400	ALTER TABLE `stu`.`sc` ADD PRIMARY KEY (`sno` ...
22	0.00024125	select * from sc where cno='C01' LIMIT 0, 1000

对比如下表：

索引方式	执行时间	比较
不建立索引	0.00089025	时间最长
非聚集索引	0.00025475	时间较短
聚集索引	0.00024125	时间最短

(3) 对查询条件为一个连续的范围的查询（例如查看学号在某个范围内的学生的选课情况）

- ① 不建立索引
- ② 建立非聚集索引
- ③ 建立聚集索引

三种情况执行如下表：

	Query_ID	Duration	Query
▶	1	0.00006500	SHOW WARNINGS
	2	0.07788425	ALTER TABLE `stu`.`sc` DROP PRIMARY KEY
	3	0.00080350	select * from sc where sno between '30202' and '3022...
	4	0.04246150	create index stu_cno_index on sc(sno,cno)
	5	0.00031575	select * from sc where sno between '30202' and '3022...
	6	0.02280225	ALTER TABLE `stu`.`sc` DROP INDEX `stu_cno_ind
	7	0.09395275	ALTER TABLE `stu`.`sc` ADD PRIMARY KEY (`sno` ...
	8	0.00024375	select * from sc where sno between '30202' and '3022...

对比如下表：

索引方式	执行时间	比较
不建立索引	0.00080350	时间最长
非聚集索引	0.00031575	时间较短
聚集索引	0.00024375	时间最短

(4) 索引代价。在有索引和无索引的情况下插入数据（例如在选课情况表 SC 上插入数据），比较插入的执行效率。

- ① 不建立索引
- ② 建立非聚集索引
- ③ 建立聚集索引

三种方式执行如下表：

	35	0.07853725	ALTER TABLE `stu`.`sc` DROP PRIMARY KEY
	36	0.00212025	insert into sc values('11111','C01',99)
	37	0.04060000	create index stu_cno_index on sc(sno,cno)
	38	0.00185175	insert into sc values('11111','C02',99)
	39	0.02531150	ALTER TABLE `stu`.`sc` DROP INDEX `stu_cno_ind
	40	0.09881050	ALTER TABLE `stu`.`sc` ADD PRIMARY KEY (`sno` ...
	41	0.01239650	insert into sc values('11111','C03',99)

对比如下表：

索引方式	执行时间	比较
不建立索引	0.00212025	时间较短
非聚集索引	0.00185175	时间最短
聚集索引	0.01239650	时间最长

2. 对相同查询功能不同查询语句的执行比较分析

(1) group by

```
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。

22	0.00323925	select avg(grade) from sc group by cno having cno='C01' ;
23	0.00064750	select avg(grade) from sc where cno='C01' LIMIT 0, 1

两者对比如下表：

是否含有 group by	执行时间	比较
是	0.00323925	时间最长
否	0.00064750	时间最短

由上表可知，使用group by语句效率会降低，由于group by语句相比直接使用聚集函数，首先要进行数据集进行分组，再在分组中使用聚集函数，时间比不使用分组花费得要多。

(2) select sno, sname, age

```
from student s1
where age =
(select max(age)
from student s2
where s1.dept = s2.dept)
```

另一个：

```
select dept , max(age) as maxAge into tmp
from student
group by dept;
select sno, sname , age
from student, tmp
where student.age = tmp.maxAge 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。

```

/*未重写*/
/*select sno,sname,bdate
from student s1
where bdate =
(select max(bdate)
from student s2
where s1.dept = s2.dept);*/
/*重写后*/
/*create table tmp
(select dept,max(bdate) as maxdate from student group by dept);
select sno,sname,bdate
from student,tmp
where student.bdate=tmp.maxdate and
tmp.dept=student.dept;*/

```

执行情况如下：

44	0.02441050	select sno,sname,bdate from student s1 where bdate ...
45	0.02748875	create table tmp (select dept,max(bdate) as maxdate f...
46	0.00035375	select sno,sname,bdate from student,tmp where stude...

比较如下表：

情况	执行时间	比较
未重写	0.02441050（直接查询）	较长
重写	0.02748875（建表时间）	查询时间较未重写时 优化了两个量级，但 建表时间耗费更多。
	0.00035375（查询时间）	

(3) 对下面两个查询进行比较

```

select sname, age
from student
where dept != 10 and age > all
(select age
from student
where dept = 10)

```

另：

```

select sname ,age
from student
where dept != 10 and age >
( select max(age)
from student
where dept = 10)

```



```

/*使用all
select sname,bdate
from student
where dept != '计算机' and bdate > all
(select bdate
from student
where dept = '计算机');
/*不使用all
select sname,bdate
from student
where dept != '计算机' and bdate >
(select max(bdate)
from student
where dept = '计算机');

```

两者执行情况如下：

48	0.00034200	select sname,bdate from student where dept != '计算...
49	0.00031375	select sname,bdate from student where dept != '计算...

对比如下表：

情况	执行时间	比较
使用 all	0.00034200	时间最长
不使用 all	0.00031375	时间最短

3. 查询优化

除了建立适当索引，对SQL 语句重写外，还有其他手段来进行查询调优，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。写出你的查询形式，以及调优过程；并说明最优情况下的运行时间。

(1) 查询选修了每一门课的学生

① 直接查询

```

1  -- 直接查询
2  • select sname from stu.student
3  □ where sno in(
4      select sno from stu.sc
5      group by sno
6      having count(*) =(select count(*) from stu.course)
7  );

```

② 嵌套子查询

```

20 -- 嵌套子查询
21 • select sno from student where not exists
22 □ (select * from course
23     where not exists
24     (select * from sc where sno=student.sno and cno=course.cno));

```

③ 清理缓存

```
9 • flush query cache;
10 -- 建立视图
```

④ 建 view

```
10 -- 建立视图
11 • create view student1 as(
12   select distinct sno as sno1, count(cno) as nums
13   from sc
14   group by sno
15 );
16 • select sname
17   from student1, student
18   where student1.nums >= 5 and student1.sno1=student.sno;
19 • drop view student1;
```

四种方法的执行时间比较如下表

方法	直接查询	嵌套子查询	清理缓存	建立视图
时间	0.00447000	0.00760275	0.00429450	0.00088100
结论	次长	最长	较短	最短

若不考虑建立视图的额外开销，则建立视图的查询时间最短。

(2) 查找至少选修了课程数据库原理和操作系统的学生的学号

① 建立视图

```
12 -- 建立视图
13 • create view tmp as(
14   select sc.sno as sno, cname
15   from sc, course
16   where sc.cno=course.cno
17   and (course.cname='数据库原理' or course.cname='操作系统')
18 );
19 • select distinct sno from tmp;
```

② 建表

```
12 -- 建表
13 • create table tmp as(
14   select sc.sno as sno, cname
15   from sc, course
16   where sc.cno=course.cno
17   and (course.cname='数据库原理' or course.cname='操作系统')
18 );
```

两种方法的执行时间比较如下表

方法	建立视图	建立表
时间	0.00072625	0.00026725
结论	较长	最短

■ 实验小结

此次实验花费了较长时间才完成，在很多查询优化方法上遇到了问题。起初在比较不建立索引和建立索引上的时间比较上出现了与预期相反的结果，比如不建立索引的查询时间反而比建立索引的查询时间要短，后期经过分析得出也许是由于查询缓存动态变化的影响，或者还与其他因素有关，每次查询的耗时都是不确定的，之后在查询优化上也遇到了问题，清除缓存时因失误错把配置文件 `my.ini` 的相关参数更改造成环境无法打开，后期查阅资料后才可解决，在之后的学习中查询优化这块知识还需加强巩固。