****

**数据库系统实验五**

2020年5月11日

专业:18软件工程

班级:软件一班

姓名:严灏

学号:201830020446

# 实验五 查询优化

### 实验目的

* 1. 熟悉SQL查询语句。
  2. 了解查询执行计划、加深理解关系代数理论

### 实验内容及要求

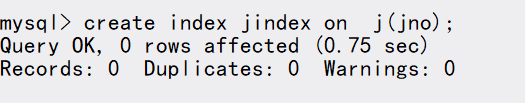
参考实验三手册。建议对于同一个查询，请同学们自行设计多种形式的查询；尤其是对那些复杂的查询，可以写出很多种形式的查询。不同形式的查询，效率可能相差很大。

### 建立关系表，并建立索引；导入数据。

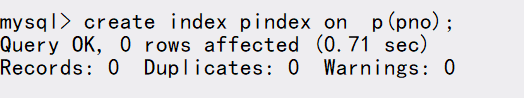
* 1. 建立索引。

1. 在四张表的主码上建立聚簇索引；

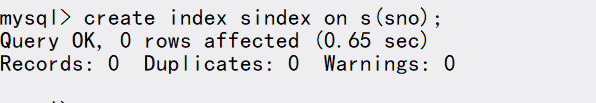
1.在j表的主键上建索引



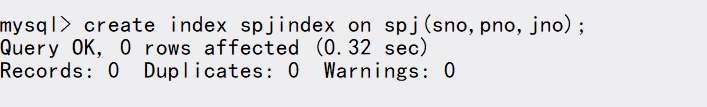
2.在p表的主键上建索引



3.在s表的主键上建索引

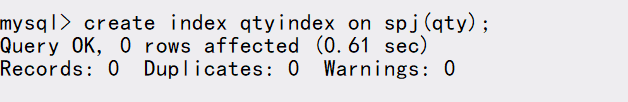


4.在spj的表上建索引



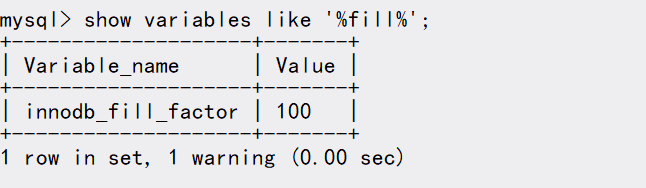
1. 自选在一些属性上建立非聚簇索引；

选择把spj的表的qty也建索引



1. 建立索引时，请思考索引的填充因子是什么意思，有什么作用；

1.首先查看填充因子



2. 填充因子是一种为以后扩容分配的空间.填充因子为100时,它预留页大小的1/16，也就是预留256Byte的空间为以后的Update留出空闲的空间，避免类似行迁移的问题，使每次访问增加了IO操作。

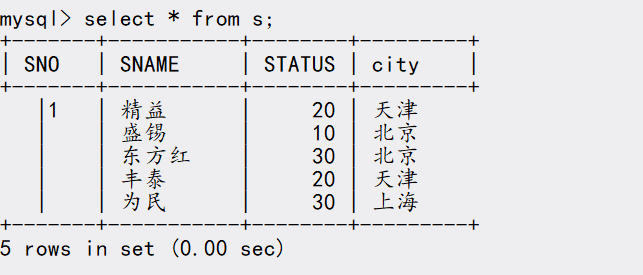
请思考索引的数据结构有哪些？可否指定索引的数据结构？如果不可以，请说明为什么？如果可以，请说明你的操作过程或SQL操作。

在学数据结构的时候,老师就已经告诉我们,索引的数据结构是B+树了. 而非关系型数据库MongoDB使用B树作为数据库索引。大部分关系型数据库，比如Mysql，则使用B+树作为索引。MyISAM引擎使用B+Tree作为索引结构，叶节点的data域存放的是数据记录的地址。 不能 除非你换一个数据库引擎或者换一个数据库软件

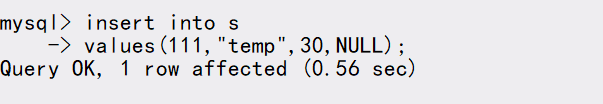
* 1. 导入空数据。

请同学们在**db\_SPJ中**自行增加一些有空值的元组。

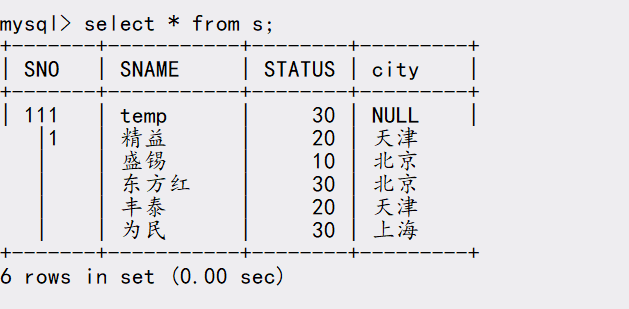
1.先在s表查询



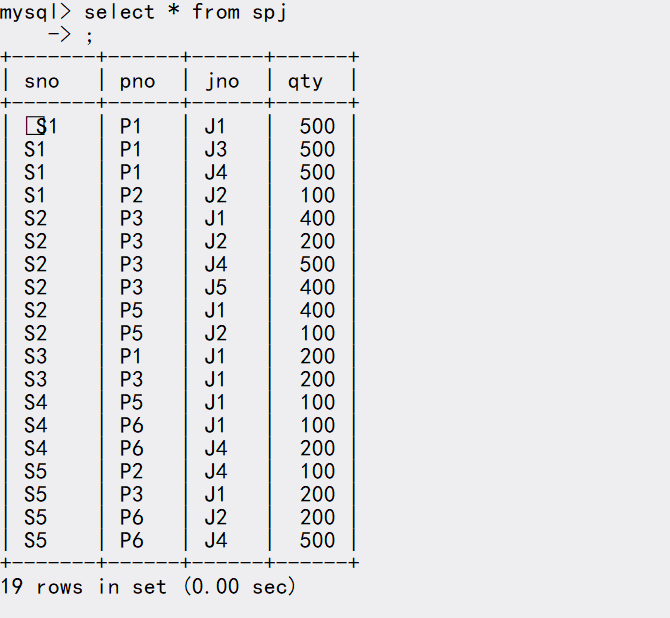
2.在s表中插入



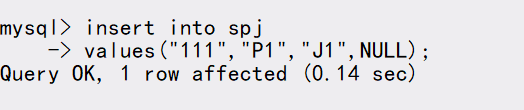
3.检查有没有成功插入



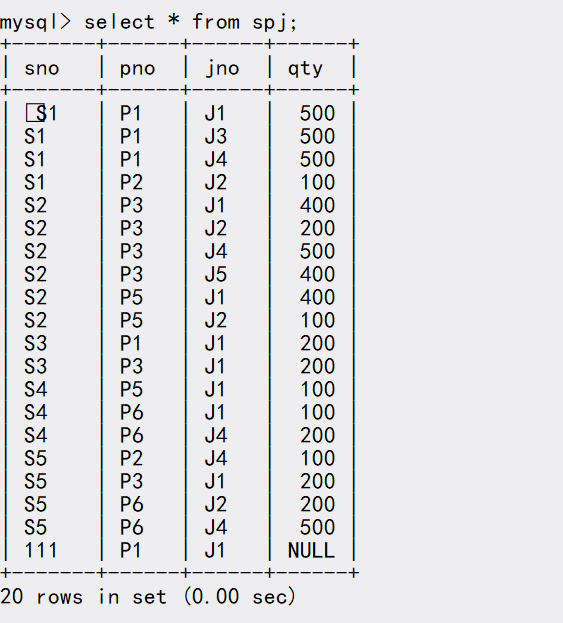
4.查询spj



5.往spj里面插空值

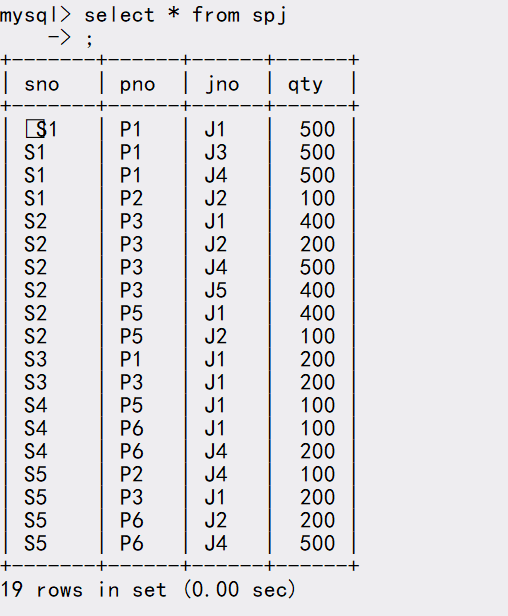


6.看看结果

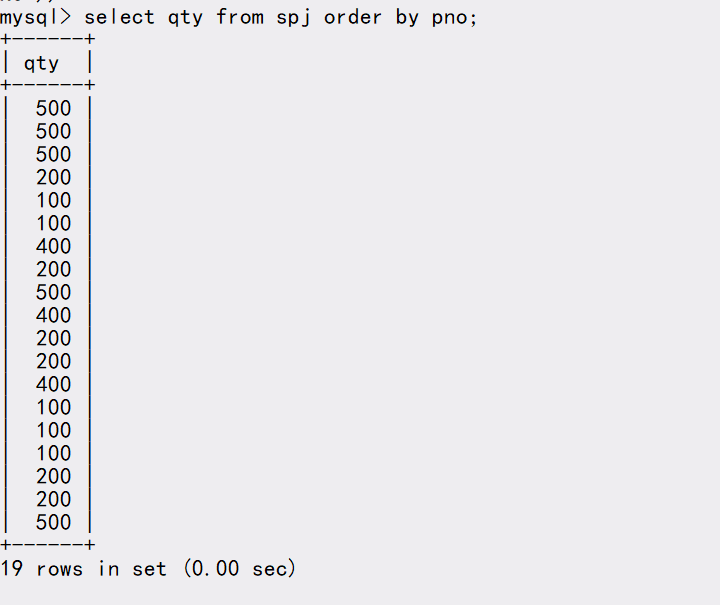


### 简单查询

* 1. 在单表上进行查询，查看某个或多个特定属性.

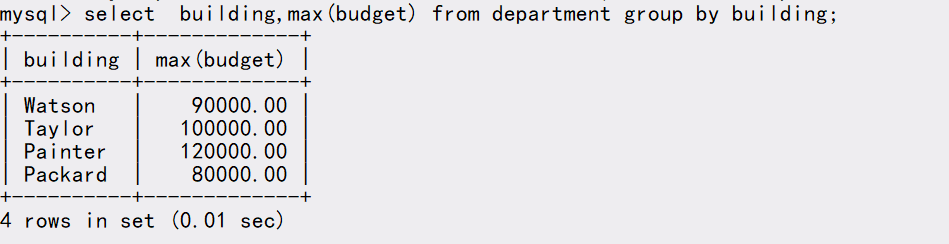


* 1. 使用排序操作Order By

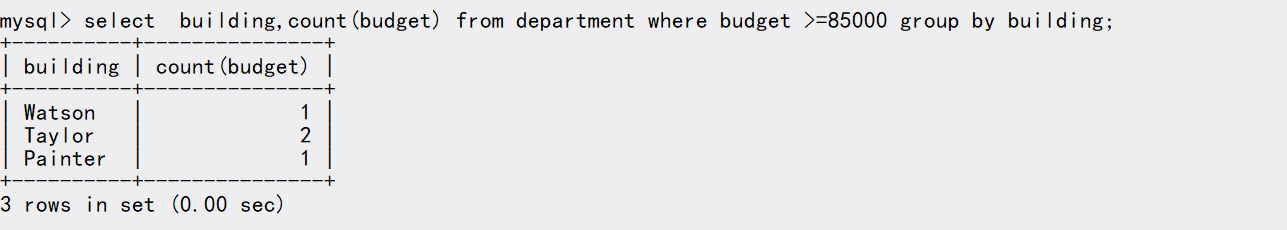


* 1. 使用分组操作Group by、having；并作相关的集函数查询；例如sum，count，avg，max，min等

1.查询每栋楼里最高收入

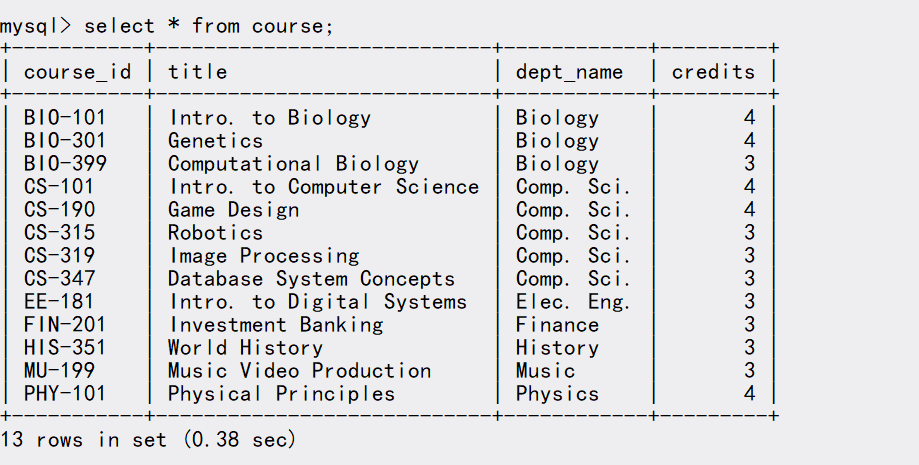


2.查找每栋楼里收入大于85000的数量

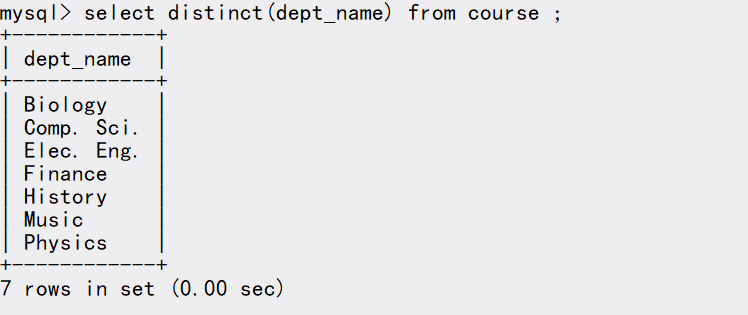


* 1. 取消重复distinct

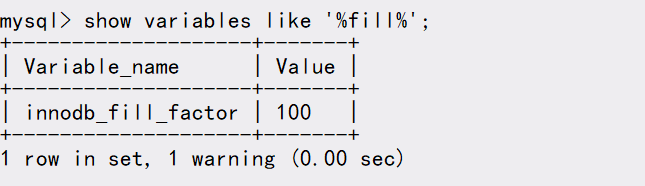
1.先看原来的表



2.再加上distinct(dept\_name)



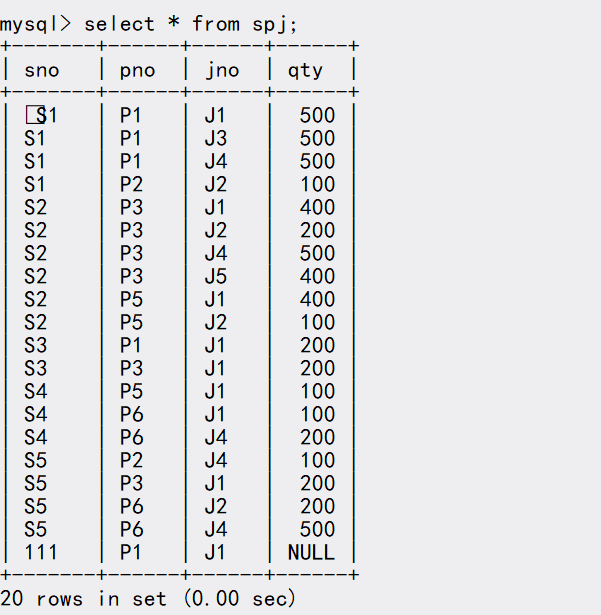
* 1. 通配符的使用。



**对于以上查询，请大家仔细看各个查询的查询计划图，回答以下问题并举出实例：**

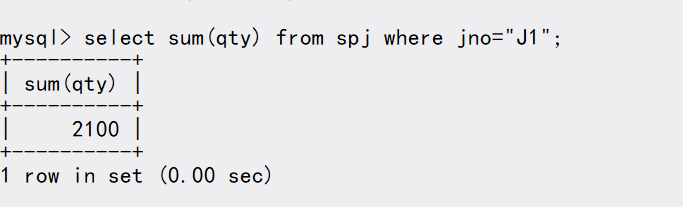
1. 空值对结果的影响（例如在计算sum, avg, min集函数时系统如何处理空值属性）？

1. 我们统计一下J1的qty的合是多少



2.手工计算好应该是2100

3.查询一下,利用sum函数

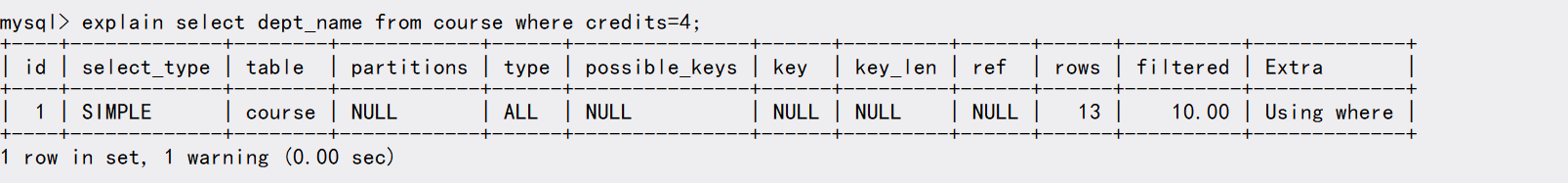


4.发现对结果没有影响

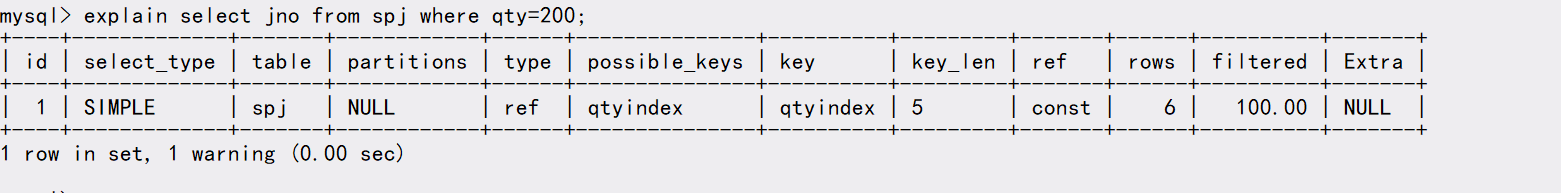
SQL将涉及空值的任何比较运算的结果视为unknown（既不是谓词is null，也不是is not null）。unknown是在SQL中除true和false之外的第三个逻辑值。

1. 注意察看查询执行计划：如果察看的属性列上有索引和没有索引，系统是读取数据的方式有何不同？

1.先看没有索引的



2.再看有索引的



区别在type:

ALL：Full Table Scan， MySQL将遍历全表以找到匹配的行

ref：使用非唯一索引扫描或者唯一索引的前缀扫描，返回匹配某个单独值的记录行

1. 是不是在一个表上有索引，所有的查询都会使用该索引去读取数据？为什么？

不是的.在有些情况下不走索引:1) 没有查询条件，或者查询条件没有建立索引2) 索引本身失效3) 提示不使用索引　4) 查询的数量是大表的大部分，应该是30％以上。

因为这种时候用索引可能并不能提高效率,反而可能会因为计算索引花费更大

1. 为什么有时候即使使用了Order by操作，但查询计划里并没有相应的操作？对于distinct也有类似的情况。

因为有些时候我们是通过有序索引而直接取得有序的数据，这样不用进行任何排序操作即可得到满足客户端要求的有序数据返回给客户端；

还有就是我们真正需要去排序的通过MySQL的排序算法将存储引擎中返回的数据进行排序然后再将排序后的数据返回给客户端。

Distinct也是一样的道理,有些时候本身就是唯一的

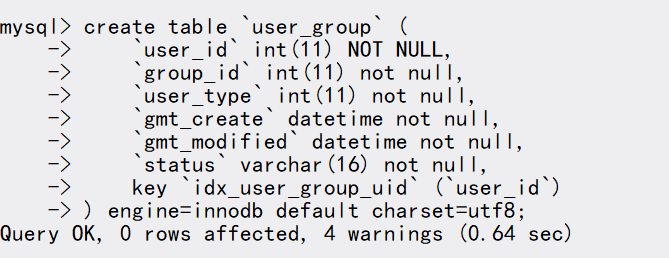
### 复杂的查询

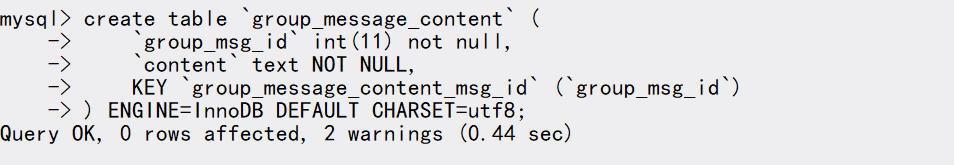
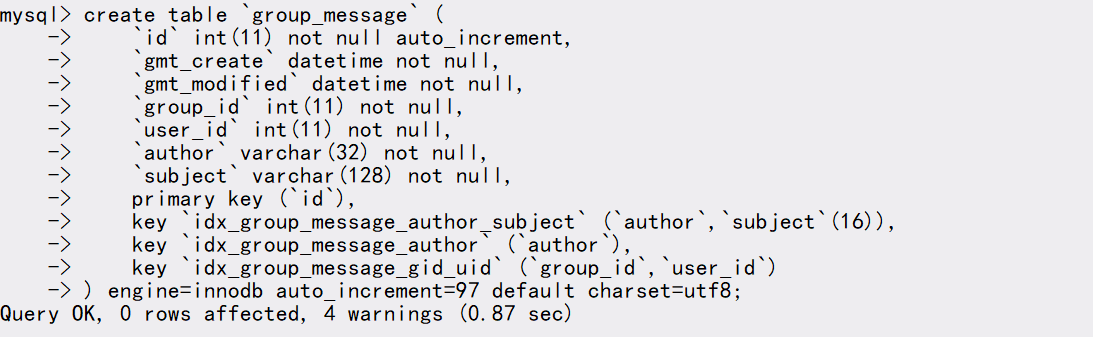
* 1. 连接查询。

1. 观察查询计划图，Join操作系统有哪些不同的实现join的方式？

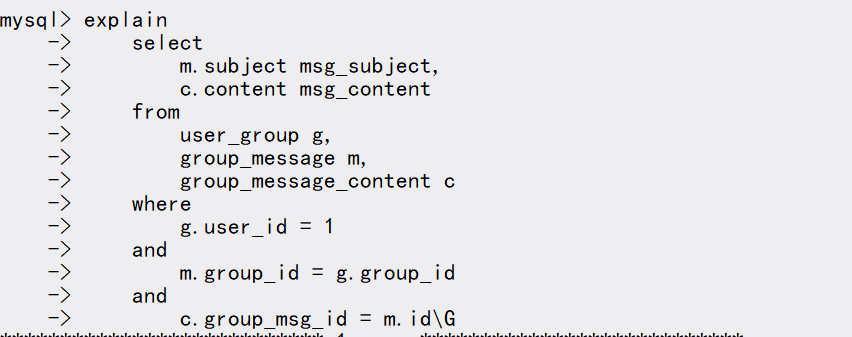
参考这篇文章: <https://www.cnblogs.com/JohnABC/p/7150921.html>

1.首先创建表结构

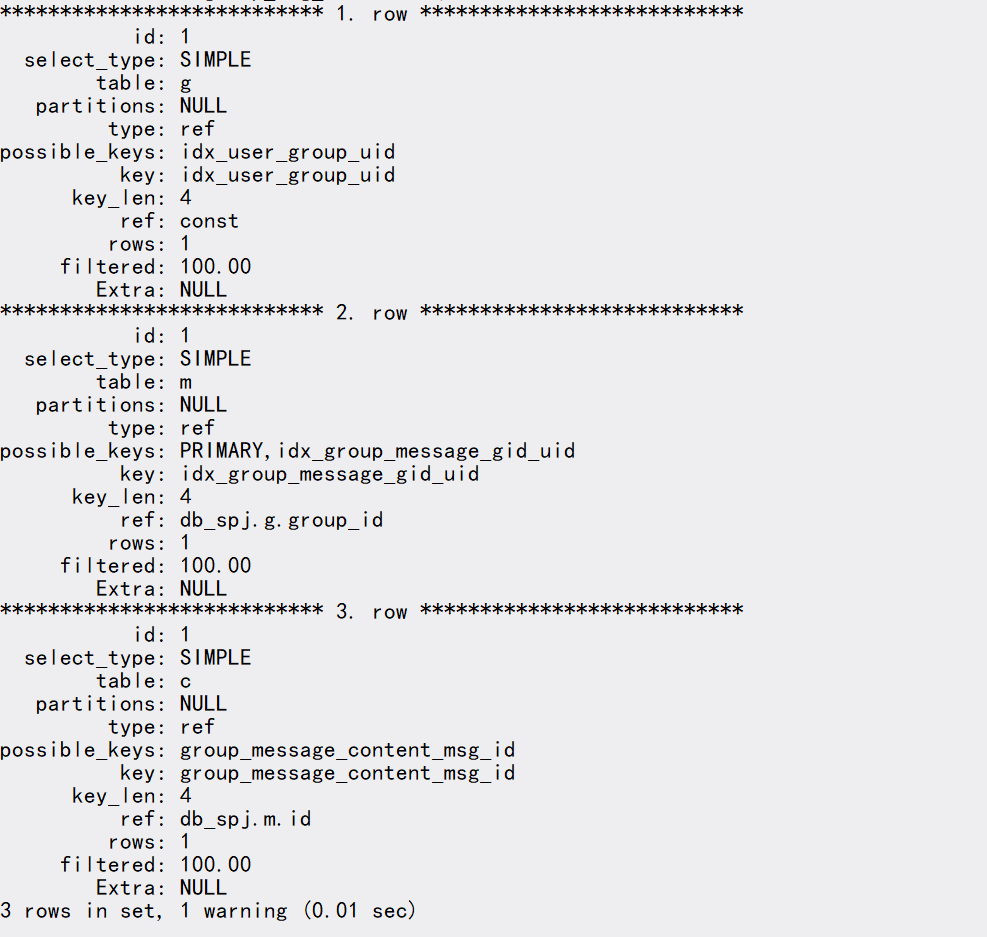




2.然后根据执行计划查询:



3.查看结果



　MySQL 会根据条件选用不同的执行策略。比如说在上面的 d 和 c 表中，如果按照当前的 c 和 d 的结构，执行 explain 之后，是 c 驱动 d 表，因为 c 表较小。

那么如果在c的id上加一个index之后，mysql就会采用d驱动c表了。

因为此时，在Nested Loop Join算法中，内部循环可以使用c表上的索引，加速执行c表的查询。内部查询每加快一点，对整个join来说都是效率上比较大的提升

总的来说: 　用小结果集驱动大结果集，尽量减少join语句中的Nested Loop的循环总次数；对被驱动表的join字段上建立索引；当被驱动表的join字段上无法建立索引的时候，设置足够的Join Buffer Size

1. 系统是否区别等值连接和自然连接？如果不区分，为什么？如果区分，请说明系统是如何区分的？

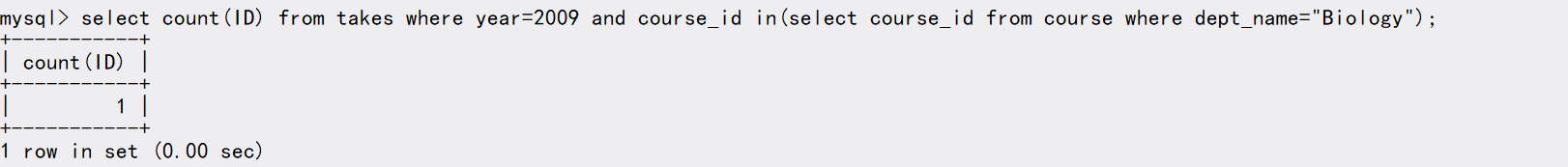
1.等值连接必须要有等值的条件，当条件不同时连接的结果也不相同，两个关系可以没有相同的属性列

2.自然连接必须要有相同的属性列才能进行，即等值连接之后要去除相同的属性列

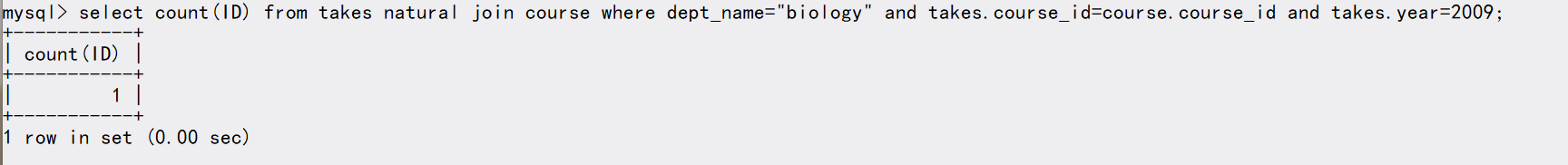
* 1. 嵌套查询

1. 对同一个查询，写出非嵌套查询和嵌套查询两种形式；观察他们的执行速度差异；观察他们的执行计划有何不同，并解释为什么。

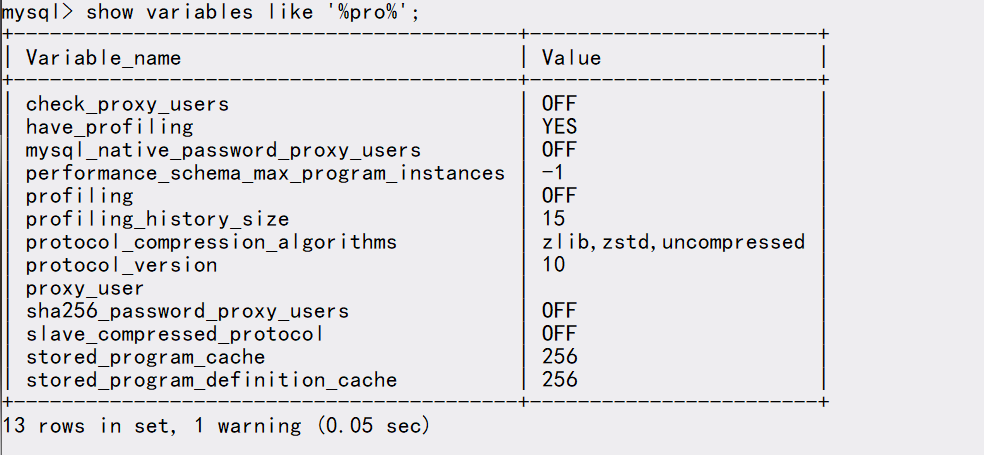
1. 我们来首先用嵌套查询



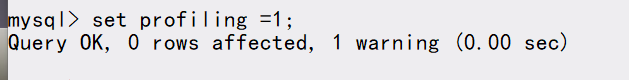
2.然后用子查询



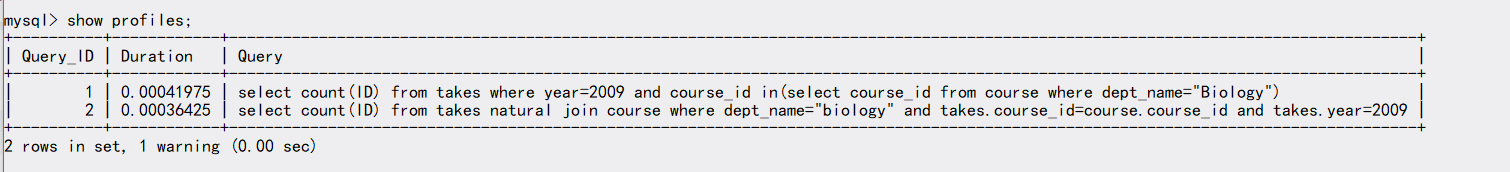
3.我们检查一下profile有没有打开



4.打开profile



5.重新执行一次并看效率



可以看出并没有太大区别,但是用了in的慢一点,可能是因为要建临时对象的原因

1. 对同一个查询，写出相关嵌套查询和非相关嵌套查询的形式，执行计划和执行效率对比
2. 通过查询验证IN，Exists，ANY，ALL，Some之间的等价关系

假设any内部的查询语句返回的结果个数是三个，如:result1,result2,result3,那么，

1.ANY

select ...from ... where a > any(...);

select ...from ... where a > result1 or a > result2 or a > result3;

2.ALL关键字与any关键字类似，只不过上面的or改成and。即:

select ...from ... where a > all(...);

select ...from ... where a > result1 and a > result2 and a > result3;

3.some关键字和any关键字是一样的功能。所以:

select ...from ... where a > some(...);

select ...from ... where a > result1 or a > result2 or a > result3;

5.IN运算符用于WHERE表达式中，以列表项的形式支持多个选择，语法如下：

　　WHERE column IN (value1,value2,...)

　　WHERE column NOT IN (value1,value2,...)

当 IN 前面加上 NOT运算符时，表示与 IN 相反的意思，即不在这些列表项内选择。

词语IN是"＝ANY"的别名。因此，这两个语句是一样的：

SELECT s1 FROM t1 WHERE s1 = ANY (SELECT s1 FROM t2);

SELECT s1 FROM t1 WHERE s1 IN (SELECT s1 FROM t2);

IN运算符用于WHERE表达式中，以列表项的形式支持多个选择，语法如下：

　　WHERE column IN (value1,value2,...)

　　WHERE column NOT IN (value1,value2,...)

当 IN 前面加上 NOT运算符时，表示与 IN 相反的意思，即不在这些列表项内选择。代码如下:

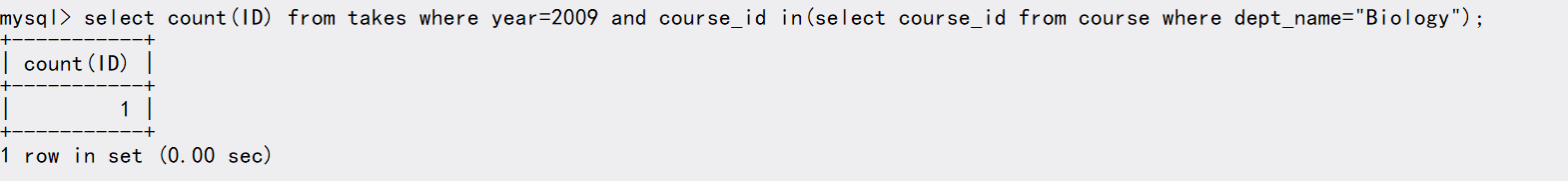
6.EXISTS 和 NOT EXISTS 子查询语法如下

　　SELECT ... FROM table WHERE EXISTS (subquery)

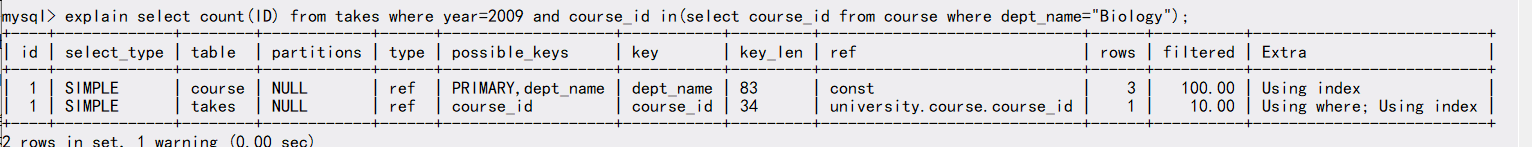
该语法可以理解为：将主查询的数据，放到子查询中做条件验证，根据验证结果（TRUE 或 FALSE）来决定主查询的数据结果是否得以保留。

* 1. 复合条件查询

1. Where条件中除了有连接条件外，还有一些选择条件。



1. 注意察看它们的执行计划，执行这些操作的顺序，是否是安装语句中条件的顺序来执行，是先做选择还是先做连接？



先做了选择,再做的连接

**思考和要求：**

1. 对于复杂的查询，尽量使用不同的形式表示出来。对比其结果，是否一样？查询计划是否一样？
2. 对同样的查询语句，有无索引的情况下，他们的查询计划有何不一样？

不一样的. 嵌套循环连接：mysql表与表之间的关联查询使用Nested-Loop join算法，顾名思义就是嵌套循环连接

在使用索引关联的情况下，有Index Nested-Loop join和Batched Key Access join两种算法；

在未使用索引关联的情况下，有Simple Nested-Loop join和Block Nested-Loop join两种算法；

### 引入视图

实验步骤3中的查询，先建立相应的视图。再在视图上进行查询。观察他们结果是否一样。它们的执行计划是否一样。分别给出例子。

### 考核内容

1. 确定自己熟练掌握之后联系助教进行考核，助教将从上述操作中随机抽取4个，考核分数将登记在册，分值占比80%。
2. 完成《实验报告》，对所有的考核点进行操作说明（对应的SQL语句/操作、输出截图），分值占比20%。