except 和 in 时间复杂度的分析

目录

[问题 1](#_Toc517734064)

[方法 2](#_Toc517734065)

[代码 2](#_Toc517734066)

[代码1 2](#_Toc517734067)

[代码2 3](#_Toc517734068)

[不同代码时间复杂度的分析（except和in） 3](#_Toc517734069)

[方法1 not in 3](#_Toc517734070)

[方法2 except 3](#_Toc517734071)

[方法1和方法2的时间复杂度比较 3](#_Toc517734072)

[关于如果询问的是in而非not in 5](#_Toc517734073)

[两者相同部分的时间复杂度分析（查询的两个表是包含关系，两个表的先后顺序） 6](#_Toc517734074)

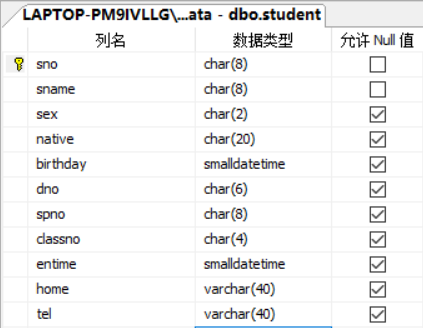
[1.对于表a的任意一行，在表d中寻找 6](#_Toc517734075)

[2.对于表d的任意一行，在表a中寻找 7](#_Toc517734076)

# 问题

求没有选修C2课程的学生姓名。

表



表





# 方法

先查询选修C2 课程的学生姓名，再用全体减去它。

而查询选修C2 课程的学生姓名的方法是先用student\_course表和course表求出选修C2课程的学生编号，得到d表，再用student表和d表求得选修C2 课程的学生姓名。

# 代码

## 代码1

(select sname from student a)

except

(select sname from student a,(select sno,score from student\_course b,course c where b.tcid=c.cno and c.cname='C1') d where a.sno=d.sno) e;

## 代码2

select sname from student a

where sname not in

(select sname from student a,(select sno,score from student\_course b,course c where b.tcid=c.cno and c.cname='C1') d where a.sno=d.sno) e;

# 不同代码时间复杂度的分析（except和in）

## 方法1 not in

对于第一个表的任意一行（sname），在第二个表从表开头到表末尾里找。

由于询问的是not in，所以要查询到第一个表的末尾。

时间复杂度：O(x\*y)。

## 方法2 except

except的处理方法是先对两个表进行排序（xlogx，ylogy），然后通过两个指针判断第一个表的每一行是否在第二个表的每一行出现（x+y）[类似归并排序的处理思想]。

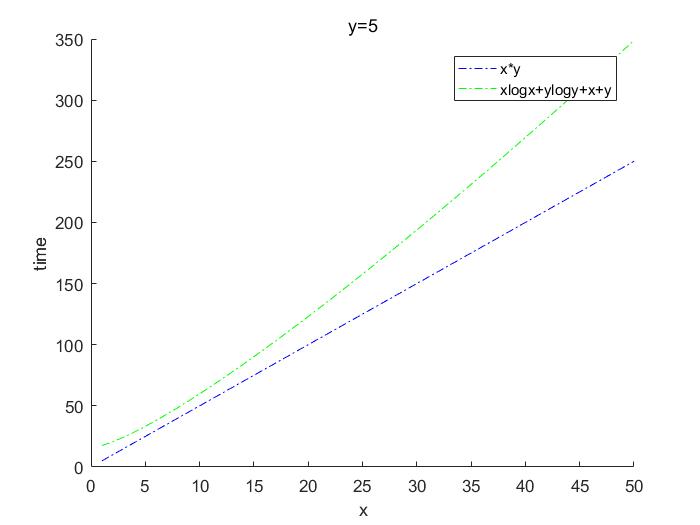
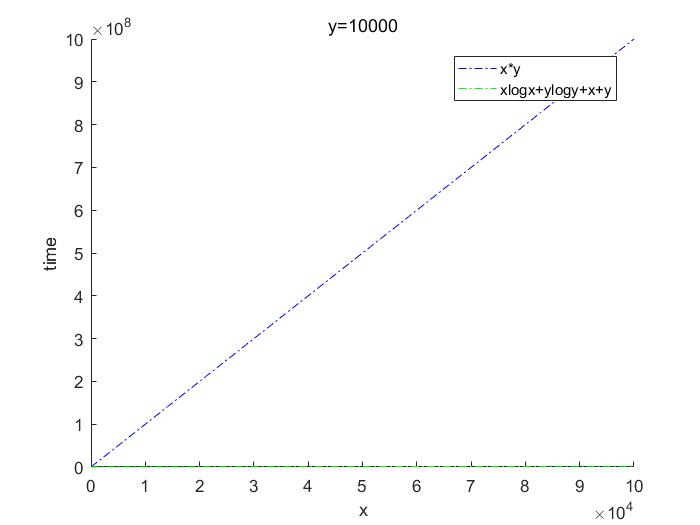
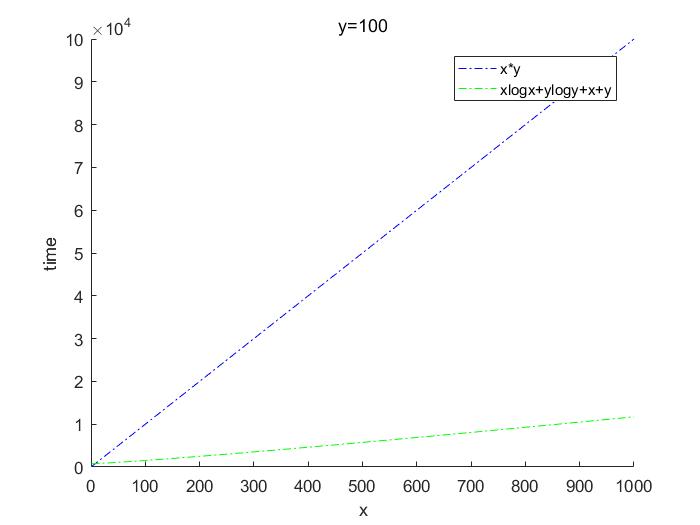
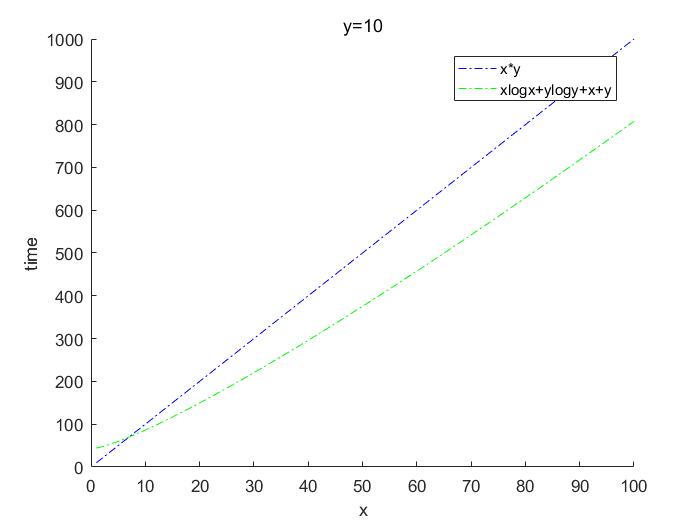
时间复杂度：O(xlogx+ylogy+x+y)。

## 方法1和方法2的时间复杂度比较

方法2需要预处理，而后续的时间效率高。

两个方法的时间复杂度 x\*y 和 xlogx+ylogy+x+y 的比较很大程度上由x与y的比例决定。假设y=某个值，x从0到无穷大，用matlab进行处理和展示。

x,y的大小也影响时间复杂度的比较。而常数x,y是一维的，当x,y很大时，常数x,y可以忽略不计，而x\*y比xlogx和ylogy大很多，可认为后者更好。

以下为matlab代码（注意，matlaab中的log指的是log10，不是log2）：

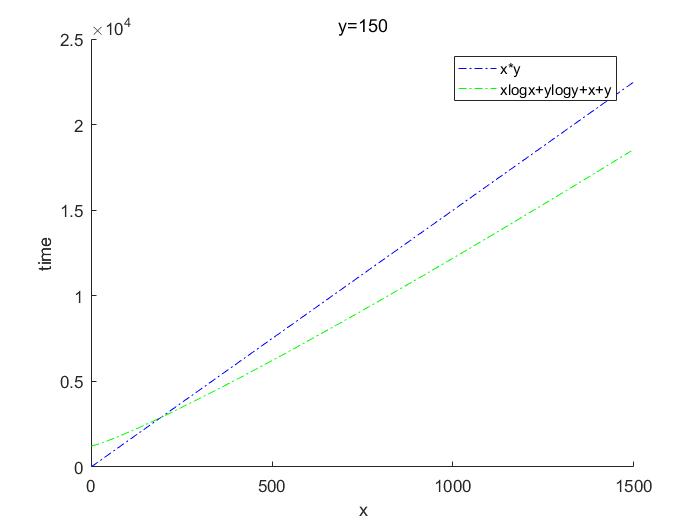
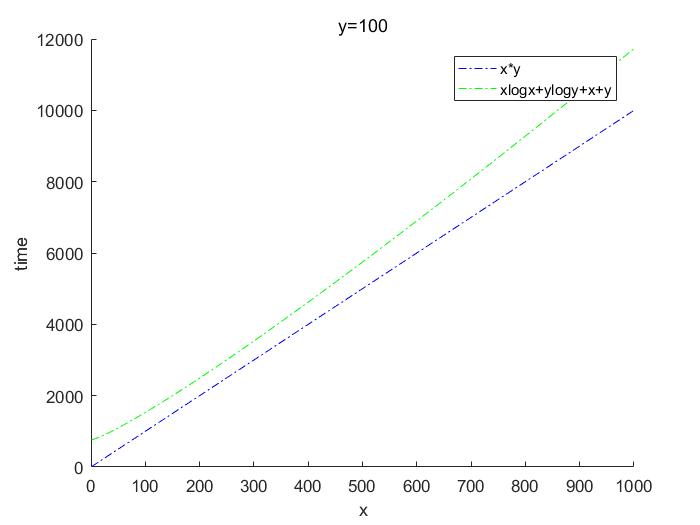
1. close all
2. clear
3. clc
4. hold on
6. y=10; %5 10 100 10000
7. i=0;
8. **for** x=1:y\*10
9. i=i+1;
10. R(i)=x;
11. S(i)=x\*y;
12. T(i)=x\*log(x)/log(2)+y\*log(y)/log(2)+x+y;
13. end
14. plot(R,S,'-.b');
15. plot(R,T,'-.g');
17. legend('x\*y','xlogx+ylogy+x+y');
18. xlabel('x');
19. ylabel('time');
20. title(sprintf('y=%d',y));

## 关于如果询问的是in而非not in

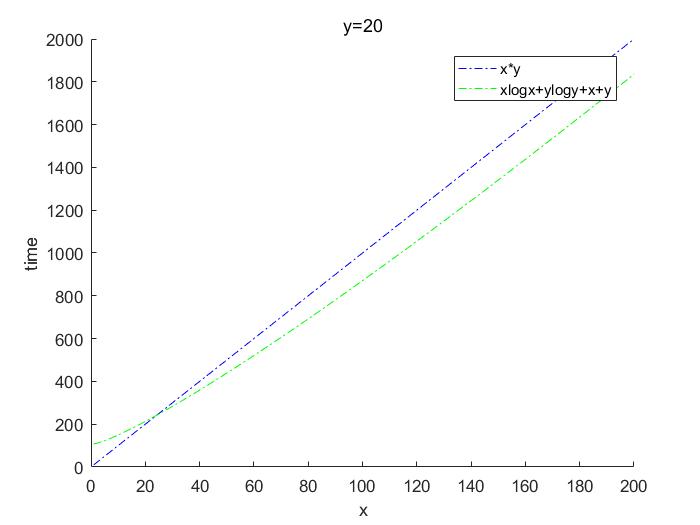
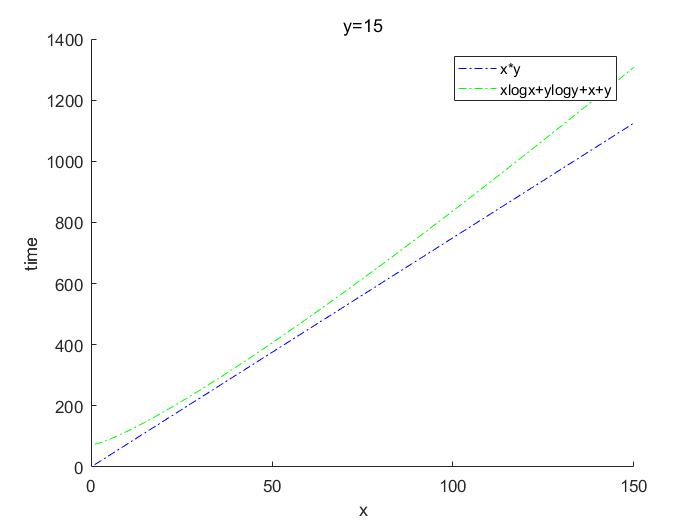
方法2时间复杂度：O(xlogx+ylogy+x+y) 不变

方法1时间复杂度：x\*y\*k k为对于表1的每一行，在表2的查询平均次数为y\*k，k为(0,1]

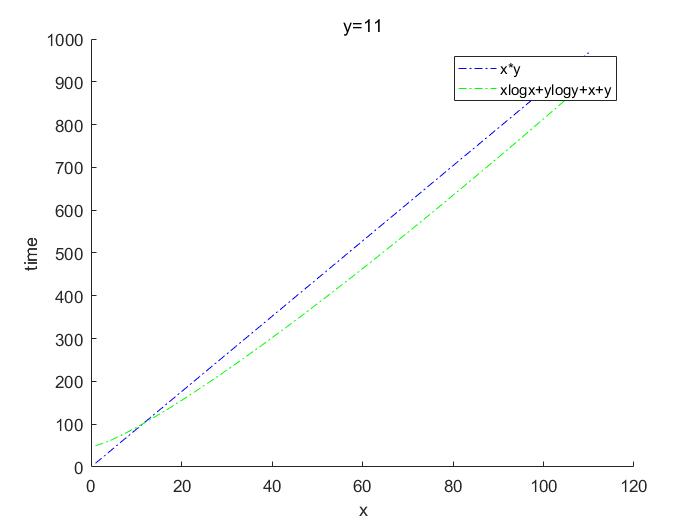
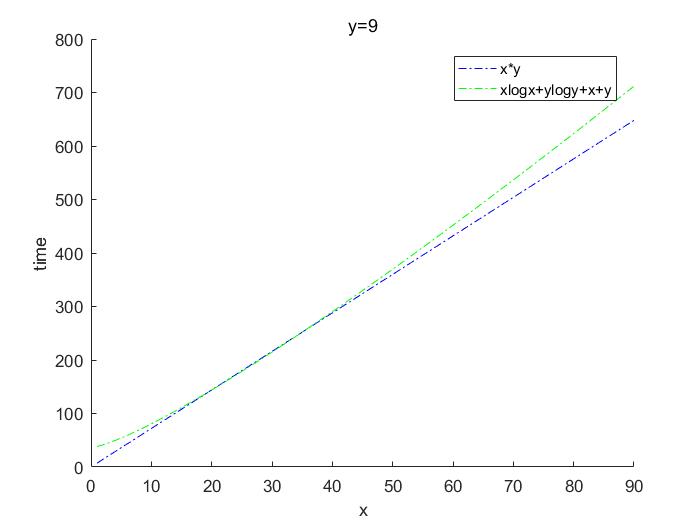
当k=0.1



当k=0.5



当k=0.8



当k=1，为上文的图

## 两者相同部分的时间复杂度分析（查询的两个表是包含关系，两个表的先后顺序）

假设student表的行数为x，d表的行数为z。

使用 in 和 = 的效果相同。

但是两个表的前后顺序对于时间复杂度是不相同的！

(select sname from student a,(select sno,score from student\_course b,course c where b.tcid=c.cno and c.cname='C1') d where a.sno=d.sno) e

(select sname from (select sno,score from student\_course b,course c where b.tcid=c.cno and c.cname='C1'),student a d where a.sno=d.sno) e

假设student表的数据分布（不同行）是随机的。

表a包含表d，即表d的任意一行必定可以在表a找到。

## 1.对于表a的任意一行，在表d中寻找

若该行在表d中（有z行），查询次数为z/2；

若该行不在表d中（有x-z行），查询次数为z。

时间复杂度：z\*(z/2) + (x-z)\*z = z\*(x-z/2) = x\*z - z\*z/2

## 2.对于表d的任意一行，在表a中寻找

对于表d的任意一行，必定在表a中找到

时间复杂度：x\*z

所以使用前一个代码更好！