5）求选修了C1 课程的学生姓名；

6）求没有选修C1 课程的学生姓名；

1. 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;

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

假设student表、student\_course表、course表的行数为x,y,z，student\_course表和course表的结合表的行数为p，则

方法1：b\*c+a\*d

方法2：a\*b\*c

其中d<=b\*c，而且有时小很多，一般来说可认为方法1降一维，方法1更好



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);

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);

方法1：except 设两个表a,d的行数分别为x,y O(xlogx+ylogy+x+y)

方法2：in 对于第一个表的任意一行，在第二个表里找，直到找到或到达末尾 O(x\*y)

看时间复杂度，方法1的最差时间复杂度小于方法2的最差时间复杂度，但是方法2找到对应行，就能结束某一行的判断，能提高时间复杂度。

这题中，表a包含表d，即表a有y行能在表d中找到，找到的平均行数为y/2，则总时间复杂度为O(y\*y/2+(x-y)\*y)=O(y\*(x-y/2))

|  |  |  |  |
| --- | --- | --- | --- |
| X | Y | Time1 | Time2 |
| 100 | 0 | 764.39 | 0 |
| 100 | 5 | 781.00 | 487.50 |
| 100 | 8 | 796.39 | 768.00 |
| 100 | 9 | 801.91 | 859.50 |
| 100 | 10 | 807.60 | 950.00 |
| 100 | 20 | 870.82 | 1800.00 |
| 100 | 50 | 1096.58 | 3750.00 |
| 100 | 80 | 1350.14 | 4800.00 |
| 100 | 100 | 1528.77 | 5000.00 |

当表d的数据量很少的时候(即表b和表c交集很少的时候)，如a%d<0.08，选择方法2，否则选择方法1。

实际上，在这题中，基本上所有学生都选择了课程，即a%d无限接近于1，即当然要选择方法1。

对于方法1，最后的结果是排好序的，但是对于方法2，还是原来表a的顺序。方法1和方法2的结果的顺序也许不同。



C++代码：

#include <cstdio>

#include <cmath>

#include <iostream>

using namespace std;

int main()

{

double x=100.0,y=9.0;

printf("%.2lf\n",x\*log(x)/log(2)+y\*log(y)/log(2)+x+y);

printf("%.2lf\n",y\*(x-y/2));

return 0;

}