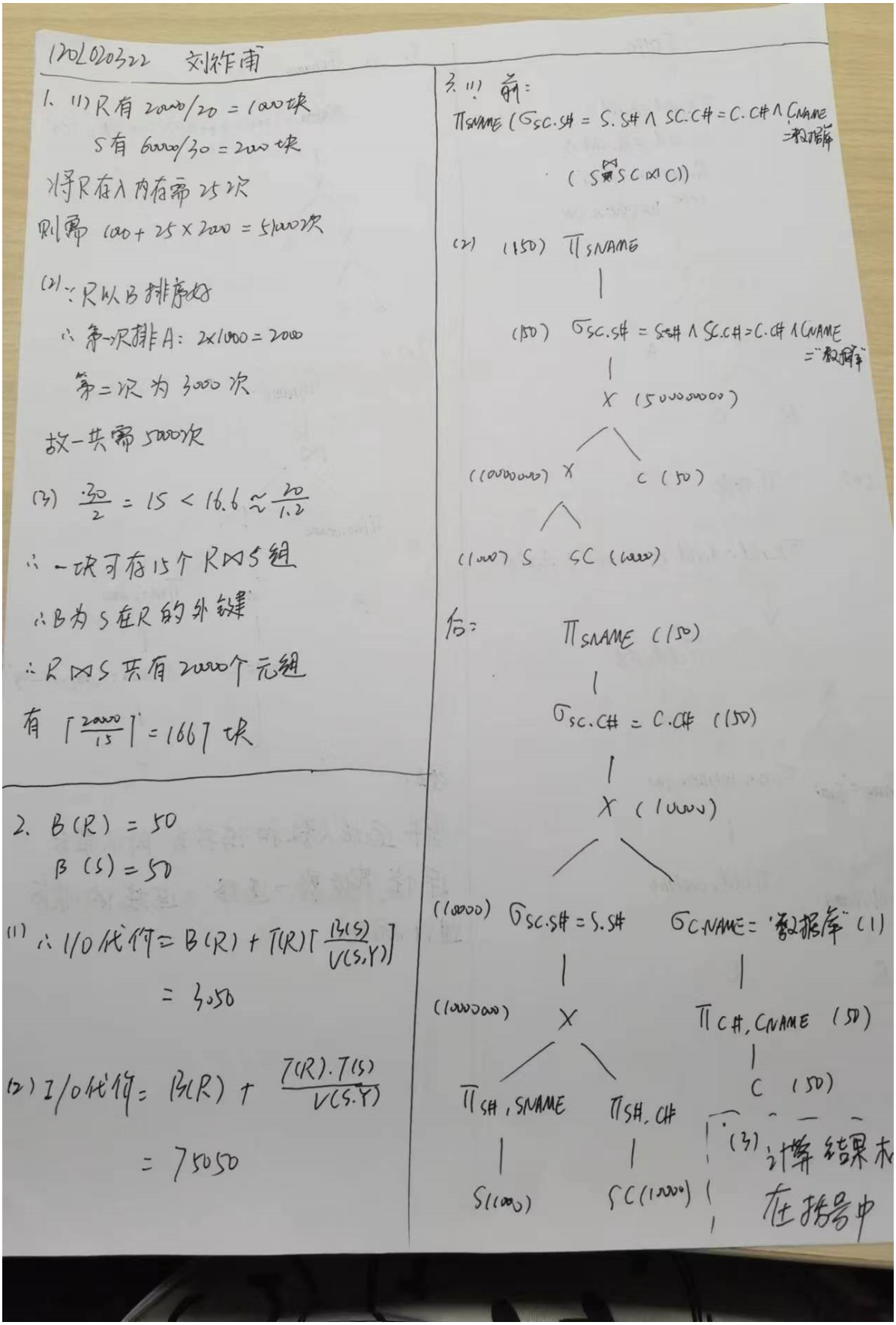
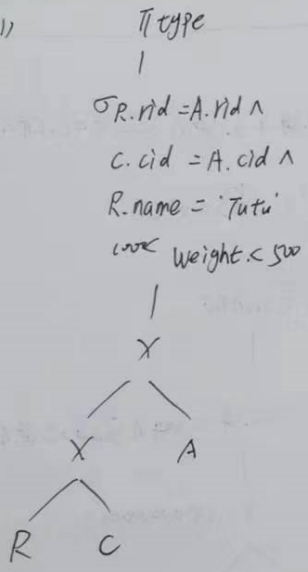
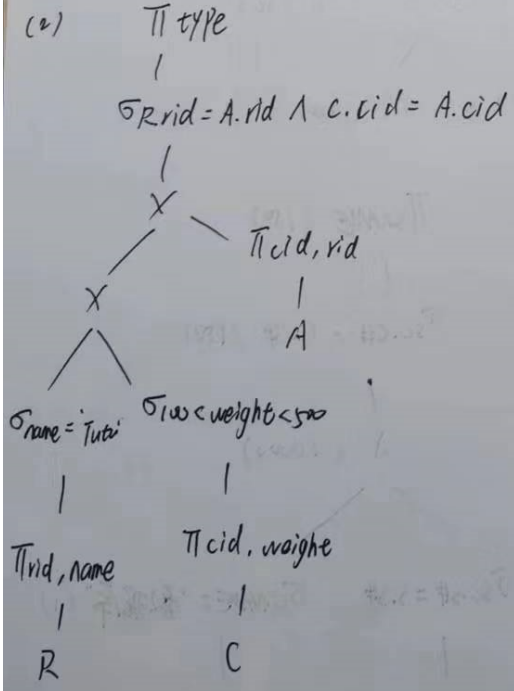
120L020322-刘祚甫-数据库第三次作业



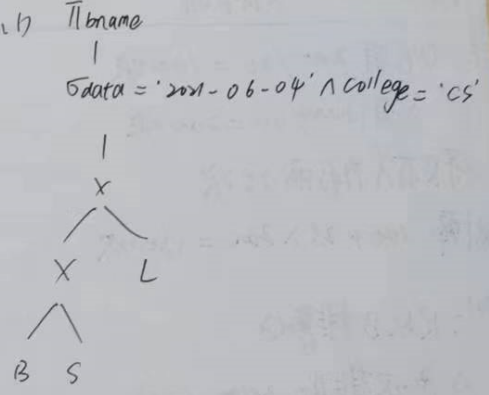
4. (1)



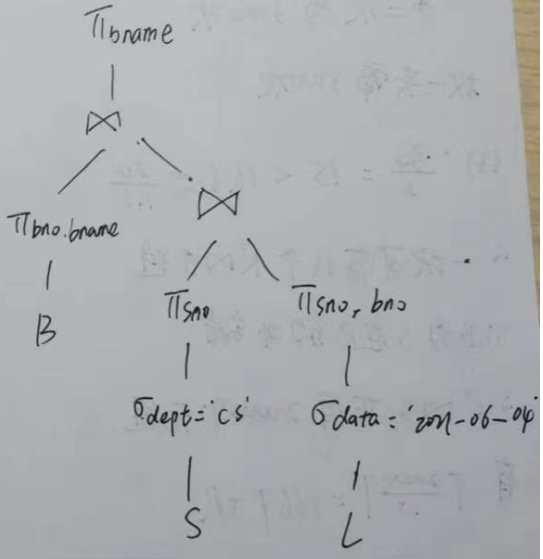
(2)



5. (1)



(2)



理由：

图书远比人数和借书多，所以最后连接，投影 > 选择 > 连接的顺序进行优化

1202020322 刘祚甫

6. 对于第一个，可形成1→2→3循环
故是可串行化的
对于第2个，
∵ $r_1(B)$, $r_2(B)$, $w_1(B)$, $w_2(B)$
∴ 1 ⇝ 2，不可串行

7. (1)
两段封锁协议：读者写数据前要获得锁，每个事务中所有封锁请求先于任何一个解锁请求

(2) 两阶段：加锁段，解锁段，加锁段中不可有解锁操作，解锁段中不可有加锁操作

```
T1: L1(A)
    L1(B)
read(A, t1)
read(B, t2)
if t1 > t2:
  t2 = t1
  while (B, t1)
U1(A)
U2(B)

T2:
L2(B)
L2(A)
  read(B, t1)
  read(A, t2)
if t1 < 0
  t2 = t1 × t1
  write(A, t2)
U2(B)
U2(A)
```

(2)

T_1	T_2
$L_1(A)$, read(A, t_1)	$L_2(A)$, 拒绝
if $(t_1 > t_2)$: $t_2 = t_1$	$L_2(B)$, 拒绝
write (B, t_2)	获得锁：
$U_2(B)$, $U_1(A)$	read (B, t_3)
	read (A, t_4)
	if $(t_3 < 0)$: $t_4 = t_3 \times t_3$
	write (A, t_4)
	$U_2(A)$, $U_2(B)$

(3)

T_1	T_2
$L(A)$, read(A, t_1)	$L_2(B)$, read(B, t_3)
$L_1(B)$, 拒绝	$L_2(A)$, 拒绝

(4) 等待图：将事务之间的等待图关系表示为一个有向图，若这个图存在环路，则说明发生死锁

超时机制：每个等待资源的事务设置一个超时时间，若超时则主动放弃并将资源回滚

8. (1) 检查点：在日志文件中增加检查点

WAL机制：在执行更改数据前，先将日志写入日志文件

undo/redo：在修改数据时，改变前后的数据都要记录

(2) T_1, T_3 需要 undo，T_2 需要 redo，

因为三者均有 start，但只有 T_2 commit.

(3) A：114514，B：hitcs

过程：先正向扫描

T_1 → undo-list

A：114 → 114514

T_2 → undo-list

B：hit → hitcs

undo-list.delete(T_1)

T_3 → undo-list

B：hitcs → hitcsdb

A：114514 → 1919810

再反向扫描：

A：1919810 → 114514

B：hitcsdb → hitcs

9.

(1) T_1 不用操作

T_2, T_4：redo

T_3, T_5：undo

(2) redo：T_6

undo：T_7, T_8

(3) 正向：

undo-list.add(T_6)

X：100 → 1

undolist.add(T_1)

X：1 → 3

undolist.add(T_8)

Y：6 → 8

Z：10 → 9

undolist.delete(T_6)

Z：9 → 10

反向：

Z：10 → 9

Z：9 → 10

Y：8 → 6

Y：6 → 50

X：3 → 1