数据库作业纸(4)

1.

(a)

- `T` 和 `U` 都未完成提交
- 恢复时不做任何重做；对 `T` 和 `U` undo
- 日志追加 `<ABORT U>` 和 `<ABORT T>`

(b)

- `T` 已提交, `U` 未提交
- 恢复时对 `T` redo；对 `U` undo：撤销对 `A` 和 `C` 的更改
- 日志追加 `<ABORT U>`

(c)

- `T` 已提交, `U` 未提交
- 恢复时对 `T` redo；对 `u` undo：撤销对 `A`、`C` 和 `E` 的更改
- 日志追加 `<ABORT U>`

(d)

- `T` 和 `U` 都完成提交
- 恢复时对 `T` 和 `U` redo

2.

$T_1 : sl_1(A); xl_1(B); read(A); read(B); if A = 0 then B := B + 1; write B; u_1(A); u_1(B)$

$T_2 : xl_2(A); sl_2(B); read(A); read(B); if B = 0 then A := A + 1; write A; u_2(A); u_2(B)$

满足两阶段封锁协议，且不会发生死锁：

如果T_1先执行，则T_2申请不到A的排他锁，会等待$xl_2(A)$，直到T_1全部完成，两个事务线性完成

如果T_2先执行，则T_1申请不到A的共享锁，会等待$sl_1(A)$，直到T_2全部完成，两个事务线性完成