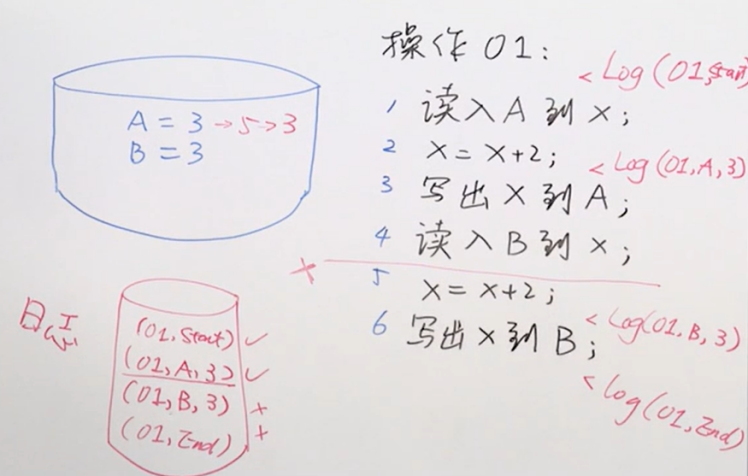
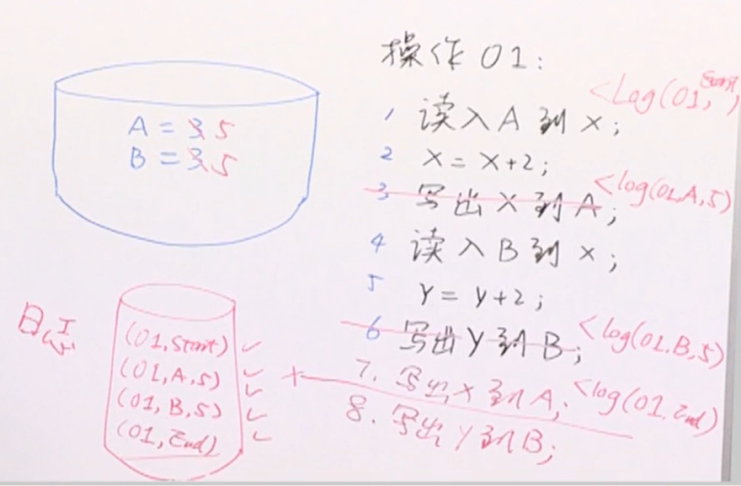
* OLTP和OLAP
  1. OLTP：事务性；OLAP：分析性
  2. 数据：
     + 状态数据：账户、购物车
     + 历史数据：购物历史、打车记录、行车轨迹
  3. OLTP：稳定性，正确性，更新——事务处理功能
  4. OLAP：处理能力，复杂SQL
* 数据正确性问题
* 数据库操作的原子性
* 日志机制
  1. UNDO日志  
     
  2. REDO日志  
     
  3. UNDO/REDO记录

**第*1*题：是系统使用了undo日志，在故障发生后，发现日志记录如下：<o1,start>, <o2,start>, <o2,A=5>, <o2,B=4>, <o2,end>, <o1,A=3>，<o1,C=3>。请问：系统恢复后A的取值是多少？**

A：5

B：4

C：3 √

D：不知道

在o2结束和o1开始之间发生故障

<o2,A=5>代表o2将A从5修改到另一个值

**第*2*题：我们用log表示将日志写到硬盘，用write表示把数据写到硬盘。如果系统使用undo日志，那么以下哪个操作执行序列是不能保证原子性的？**

A：log(o1,start), log(o1,A=5), write(A=6), log(o1,B=3), write(B=4), log(o1,end)

B：log(o1,start), log(o1,A=5), write(A=6), write(B=4), log(o1,B=3), log(o1,end) √

C：log(o1,start), log(o1,A=5), log(o1,B=3), write(A=6), write(B=4), log(o1,end)

D：log(o1,start), log(o1,A=5), log(o1,B=3), write(B=4), write(A=6), log(o1,end)

对应的log操作要在write前

**第*3*题：我们用log表示将日志写到硬盘，用write表示把数据写道硬盘。如果系统使用redo日志，那么以下哪个操作执行序列是不正确或不可能发生的？**

A：log(o1,start), log(o1,A=5), log(o1,end), write(A=5), log(o2,start), log(o2,A=6), log(o2,end), write(A=6)

B：log(o1,start), log(o1,A=5), log(o1,end), log(o2,start), log(o2,A=6), write(A=5), log(o2,end), write(A=6)

C：log(o1,start), log(o1,A=5), log(o1,end), log(o2,start), log(o2,A=6), log(o2,end), write(A=6)

D：log(o1,start), log(o2,start), log(o2,A=6), log(o2,end), log(o1,A=5), log(o1,end), write(A=5), write(A=6) √ o1和o2的操作不应该存在交叉，否则如果发生故障就很难识别

**第*4*题：以下哪种情况是数据库系统的操作原子性（atomicity）无法保证的？**

A：两个并发的CRUD操作从效果上一定一个在前一个在后。

B：任意一个CRUD操作要么发生在故障之前，要么发生在故障之后。

C：在故障发生前已经开始的CRUD操作一定会顺利完成。√

D：在故障发生前没有结束的CRUD操作一定会被撤销。

课程中提到了Undo/Redo日志，但没有介绍其具体工作机制。请思考它的具体工作机制。假设有如下一个操作，请使用你想到的Undo/Redo日志机制阐述整个操作的执行过程。

Begin

read A from Disk into x;

x = x+5;

write x to A on Disk;

read B from Disk into y;

y = y+x;

write y to B on Disk;

End

Undo(O1, start)

Redo(O1, start)

Undo(O1, A, x)

Redo(O1, A, x+5)

Undo(O1, B, y)

Redo(O1, B, y+x+5)

Redo(O1, end)

Write(A, x+5)

Write(B, y+x+5)

Log(O1, start)

Log(O1, A, x, x+5)

Log(O1, B, y, y+x+5)

Log(O1, end)

Write(A, x+5)

Write(B, y+x+5)