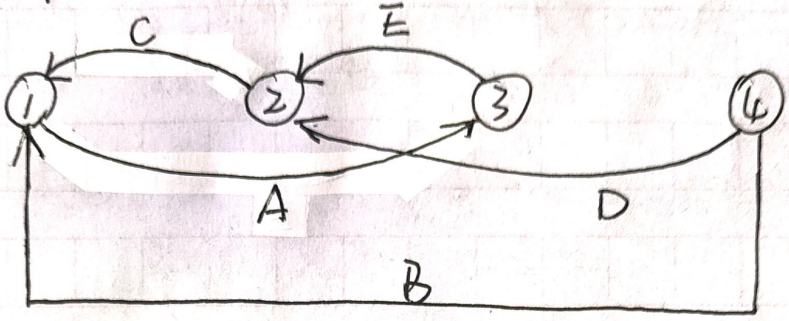
数据库系统第七次作业

2103601班 20211128张 张智雄

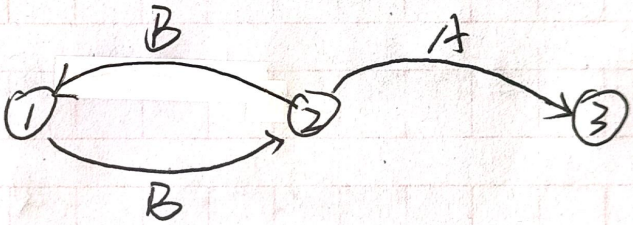
1. 事务调度

1) 不是串行调度

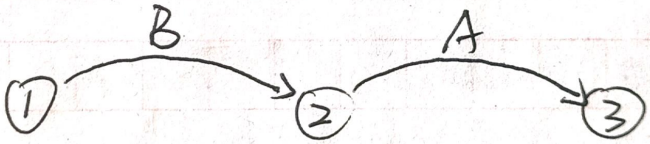
2) 

3) 不是。优先图中存在环路 $1 \xrightarrow{A} 3 \xrightarrow{E} 2 \xrightarrow{C} 1$.

2. 1) 调度S_1:



2) 调度S_2:



由此，S_2是可串行化的，而S_1是不可串行化的.

3. 1) X = 116512 属于丢失修改不一致性.

2)

T1	T2
lock(X)	
	lock(X)
a←Read(x)	
a = a-1	
Write(X,a)	
Unlock(X)	
	a←Read(x)
	a = a-2
	Write(X,a)
	Unlock(X)

4. 1) 对应缓冲区的处理策略是：steal + no force.

Steal：允许DBMS将未提交事物所做修改写到磁盘并覆盖现有数据. 而No steal则不允许.

Force：DBMS强制事物在提交前须将其所做修改全部写回磁盘. 而No Force不强制.

2) undo：T_2. T_3　　redo：T_1.

理由：T_2. T_3只有<T_i, begin>而无<T_i, commit>，属于未提交事物

T_1有<T_1, commit>，属于已提交的事物.

3) A = 114514　　B = hitcs.

具体过程：

① 从后向前扫描日志记录，建立两个事务表

（未提交事务表（T_2. T_3），已提交事务表（T_1））

有<T_i, begin>但无commit　　具有<T_i, commit>

② 对于未提交事务表中的T_2. T_3，执行UNDO(T)，并写<T_i, abort>日志记录，表明撤销完成.

③ 对于提交事务表中的T_1，执行REDO(T_1)，即对A修改为114514，B修改为"hitcs".

5. 1) 在执行检查点时才写回磁盘，因此

$X=3$　　$Y=3$　　$Z=2$

2) T_1 在检查点之前已提交，所以不做任何处理。

T_2 在故障之前已提交，所以需要 REDO。原因是因为 STEAL 的缓冲区策略，T_2 所做修改可能还未写入磁盘

T_2 在故障前未提交，执行 UNDO 撤销。

3) REDO T_2，UNDO T_2。

$X=3$　　$Y=1$。　　$Z=3$

6. 1) T_1, T_4 需要操作；T_2, T_6 需要重做；T_3, T_5 需要撤销。

2) redo：T_6。　　undo：T_7, T_8。

3) ①系统检查日志找到最后一条 <checkpoint>。

②对此中的事务，及检查点写入日志后开始的事务进行检查

→ T_7, T_8 无 $<T_i, commit>$ / $<T_i, abort>$ 记录，执行 undo 先 redo，再 undo。→ T_6 有 $<T_6, Commit>$，执行 redo。

对于 undo，从后向前扫描，对形如 $<T_i, X_j, V_1, V_2>$ 的记录，将 V_1 写入 X_j（对于 undo-list 中事务）。

对于 redo，正向扫描，对形如 $<T_i, X_j, V_1, V_2>$ 或 $<T_i, X_j, V_2>$ 重做此操作（此阶段假设 undo-list）