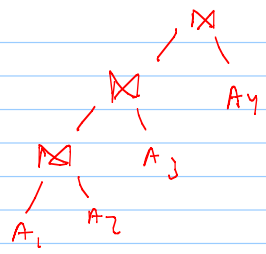
Announcements

- Sample Final Review ⇒ Discussions this week!
  - solution will be posted, too...
- Final topics: everything up to midterm (ER, RA, SQL, ...) plus modules since then: 5. ICs, 6. Sharx, 7. Indexing, 8. Query Processing 9. Transactions (see sample final, too!)
  also: Skyline queries! (last week) ⇒ email to group w/ details
- Today: last lecture!
- Next week: Additional Office Hours!
- Monday 1-4pm: 2144 Academic Surge.
  7-8pm: Google Hangout.

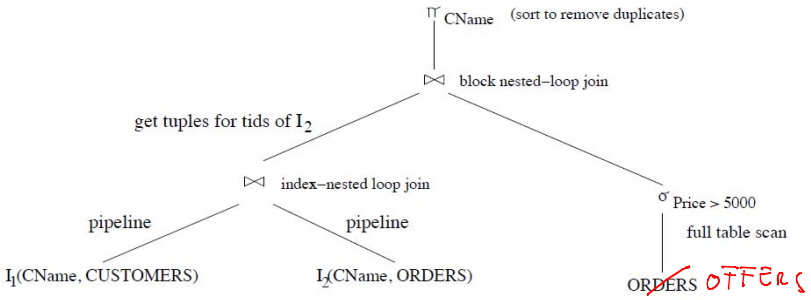
  + addt. TA office hours!
  ~ ta-165@googlegroups.com
- SmartSite grades:
  - check them!!
  - email to ta-165@googlegroups.com
- HW#4: up tomorrow!
- Group Project: Fr → Sat!?
- Eval.ucdavis.edu

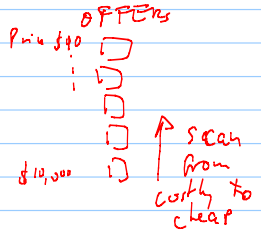
8. Query Processing Wrap Up

$A_1 \bowtie A_2 \bowtie \dots \bowtie A_n$



$4! = 24$ ways to create a "left-deep" join order





Lost Update

Time	Transaction T_1	Transaction T_2
1	**read**(A,x) 1000	
2	x:=x+200	
3		**read**(A,y) 1000
4		y:=y+100 1100
5	**write**(x,A)	
6	1200	**write**(y,A) 1100
7		**commit**
8	**commit**	

9

read(A,y) ...

The update performed by T_1 gets lost; possible solution: T_1 locks/unlocks database object A

$\Longrightarrow$ T_2 cannot read A while A is modified by T_1

$A = 1000$

$T_1; T_2 \rightsquigarrow A' = 1300$

$T_2; T_1 \rightsquigarrow A' = 1300$

$T_1 \,||\, T_2 \rightsquigarrow A' = 1100$

Inconsistent Analysis (Incorrect Summary Problem)

Time	Transaction T_1	Transaction T_2
1	**read**(A,y1)	
2	110	**read**(A,x1)
3		x1 := x1 - 100
4		**write**(x1, A) 10
5		**read**(C,x2) 130
6		x2 := x2+x1 140
7		**write**(x2,C)
8		**commit**
9	**read**(B,y2) 120	
10	**read**(C,y3) 140	
11	sum := y1 + y2 + y3	
12	**commit**	

In this schedule, the total computed by T_1 is wrong (off by 100).
$\Longrightarrow$ T_1 must lock/unlock several db objects

$T_1 ; T_2$ ↳ $\Sigma = 360$

A	B	C
110	120	130
10		140

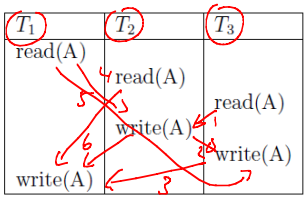
$T_2 ; T_1$ ↳ $\Sigma = 270$

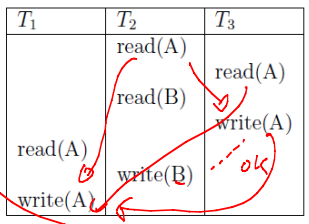
X_1	X_2	X_3
10	130	
	140	

$T_1 \,||\, T_2$ ↳ $\Sigma = 370$

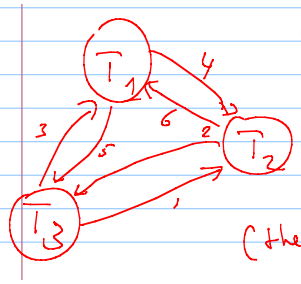
Problem 7 (Transaction Processing) (12 Points)

a. Assume the following two schedules:

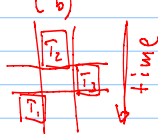




(i) Draw the precedence graph for each schedule!
(ii) Which of the two schedules is conflict-serializable?
(iii) For each serializable schedule, give an equivalent serial schedule.

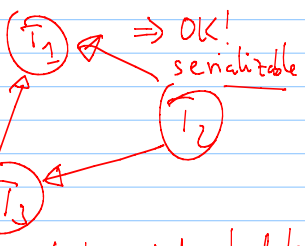


precedence graph is cyclic
⇒ not serializable
(there is no equivalent serial schedule)



Equivalent serial schedule

no cycle!
⇒ OK!
serializable



equivalent serial schedule:
(a) $T_2 ; T_3 ; T_1$

(c)
T_2 : read(A) read(B) write(A)
T_3 : read(A) write(A)
T_1 : read(A) write(A)

2PL

T_i :

slock (A)

:

xlock (B)

:

xlock (C)

:

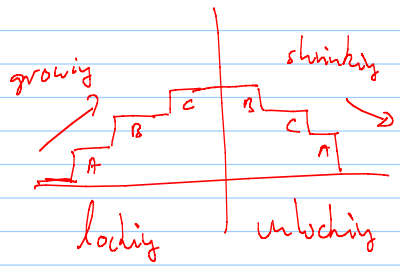
unlock (B)

:

unlock (C)

:

unlock (A)



strict 2PL