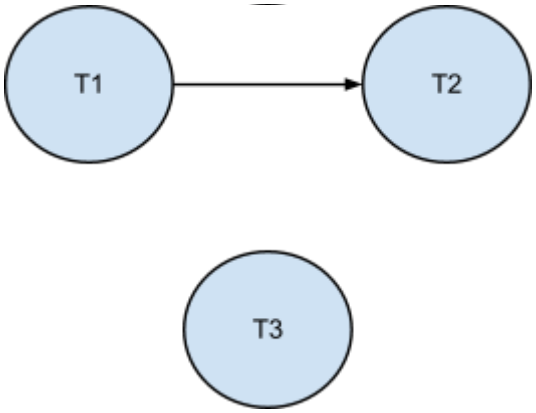
Ferhat Elmas 214805

Advanced Databases Hw-11

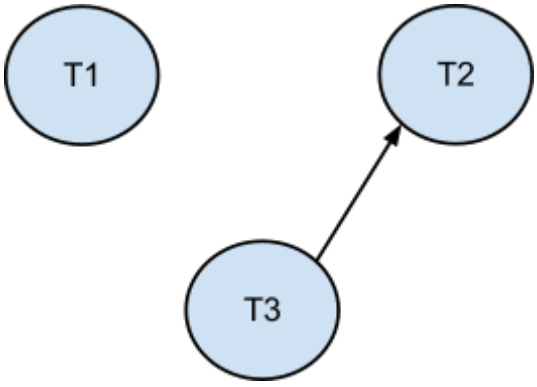
1-)

Precedence Graph of schedule s1:



There is no cycle in the graph so s1 is conflict serializable and since it is conflict serializable, it is also view serializable. s1 doesn't avoid cascading aborts since T2 reads value of C which is uncommitted value by T1.

Precedence Graph of schedule s2:



There is no cycle in the graph so s2 is also conflict serializable and view serializable. s2 doesn't avoid cascading aborts since T2 reads value of Y which is uncommitted value by T3.

- 2PL:
    - In s1, T1 releases its lock after it writes C so T2 can get lock on C and do reading. T2 reads A and releases its lock so T3 can get lock on A and do writing.

B is only accessed by T3 so it isn't also a problem. s1 can be produced as a result of 2PL.
    - In s2, T2 reads Y two times so it will release the lock after second read. However, at the same time, T3 tries to write the value of Y but T3 cannot get the lock so s2 cannot be produced by 2PL.
- Strict 2PL:
    - In s1, T1 writes C so it has the exclusive lock but T2 reads C before T1 completes and releases the lock. Therefore, s1 cannot be produced by strict 2PL.
    - In s2, T3 writes Y so it should have the exclusive lock but T1 and T2 read Y before T3 and since they are the first, they have the shared lock. Therefore, T3 cannot get exclusive lock on Y. s2 cannot also be produced by strict 2PL.
- Optimized:
    - In s1, T1 completes its read phase before T2 (so test 1 and 2 intrinsically pass), and intersection of writeset of T1 and readset of T2 isn't empty (so test 3 fails), so s1 cannot be produced by optimistic.
    - In s2,
        - T2 < T1 : $WS(T2) \cap WS(T1) = \phi$ and $WS(T2) \cap RS(T1) = \phi$
        - T2 < T3 : $WS(T2) \cap WS(T3) = \phi$ and $WS(T2) \cap RS(T3) = \phi$
        - T1 < T3 : $WS(T1) \cap WS(T3) = \phi$ and $WS(T1) \cap RS(T3) = \phi$
        - It passes from validation test so it can be produced by optimistic.
- Timestamp:
    - Assumption: transaction start times are the first time they accessed an object so TS(T1) = 1 < TS(T3) = 2 < TS(T2) = 3
    - In s1, T2 reads A with TS of 3 while T3 is trying to update it via TS of 2. As a result, this scheme isn't permitted by TS.
    - In s2, T2 reads Y which is written more recent transaction T3, so timestamp doesn't permit this scheme.
- Snapshot Isolation
    - S1 isn't permitted because T3 wants to write A but TS(T3) < RTS(A) since it is read by T2.
    - S2 is permitted because W(Z) and W(X) don't have any conflict. in T3, W(Y) is done but TS(T3) > RTS(Y) so it is ok.

2-)

a)

LSN	Log	prevLSN	undonextLSN
LSN 0	T1 starts	-	(not an update log record)
LSN 1	update: T1 writes page 2	0	LSN 0
LSN 2	T2 starts	-	(not an update log record)

LSN 3	update: T1 writes page 1	1	LSN 1
LSN 4	update: T2 writes page 5	2	LSN 2
LSN 5	T3 starts	-	(not an update log record)
LSN 6	update: T3 writes page 3	5	LSN 5
LSN 7	update: T2 writes page 5	4	LSN 4
LSN 8	T3 commits	6	(not an update log record)
LSN 9	update: T2 writes page 3	7	LSN 7
LSN 10	T2 aborts	9	(not an update log record)

b)

- Restore Page 3 to the before image stored in LSN 9
- Restore Page 5 to the before image stored in LSN 7
- Restore Page 5 to the before image stored in LSN 4

c)

LSN	Log	prevLSN	undonextLSN
LSN 0	T1 starts	-	(not an update log record)
LSN 1	update: T1 writes page 2	LSN 0	LSN 0
LSN 2	T2 starts	-	(not an update log record)
LSN 3	update: T1 writes page 1	LSN 1	LSN 1
LSN 4	update: T2 writes page 5	LSN 2	LSN 2
LSN 5	T3 starts	-	(not an update log record)
LSN 6	update: T3 writes page 3	LSN 5	LSN 5
LSN 7	update: T2 writes page 5	LSN 4	LSN 4
LSN 8	T3 commits	LSN 6	(not an update log record)
LSN 9	Update: T2 writes page 3	LSN 7	LSN 7
LSN 10	T2 aborts	LSN 9	(not an update log record)
LSN 11	Clr: T2 undo page 3	LSN 10	LSN 7
LSN 12	Clr: T2 undo page 5	LSN 11	LSN 4

LSN 13	Clr: T2 undo page 5	LSN 12	LSN 2

d)

LSN	Log	prevLSN	undonextLSN
LSN 0	T1 starts	-	(not an update log record)
LSN 1	update: T1 writes page 2	LSN 0	LSN 0
LSN 2	T2 starts	-	(not an update log record)
LSN 3	update: T1 writes page 1	LSN 1	LSN 1
LSN 4	update: T2 writes page 5	LSN 2	LSN 2
LSN 5	T3 starts	-	(not an update log record)
LSN 6	update: T3 writes page 3	LSN 5	LSN 5
LSN 7	update: T2 writes page 5	LSN 4	LSN 4
LSN 8	T3 commits	LSN 6	(not an update log record)
LSN 9	Update: T2 writes page 3	LSN 7	LSN 7
LSN 10	T2 aborts	LSN 9	(not an update log record)
LSN 11	Clr: T2 undo page 3	LSN 10	LSN 7
LSN 12	Clr: T2 undo page 5	LSN 11	LSN 4
LSN 13	Clr: T2 undo page 5	LSN 12	LSN 2
LSN 14	Clr: T1 undo page 1	LSN 13	LSN 3
LSN 15	Clr: T1 undo page 2	LSN 14	LSN 1