Quiz-1, CSE-202, Fundamentals of Database Systems

Name:_____
Roll Number:_____
Maximum Marks: 20 Time: 1 hour

Question-1: [**5 marks**] Say that a transaction T precedes a transaction U in a schedule S if every action of T precedes every action of U in S. Note that if T and U are the only transaction in S then saying T precedes U is the same as saying that S is a serial schedule (T,U). However, if S involves other transactions other than T and U, then S might not be serializable, and in fact, because of the effect of other transactions, S might not even be conflict serializable. Give an example of schedule S such that all the following holds.

i) In S, T_1 precedes T_2, and

ii) S is conflict serializable, but

iii) Every serial schedule conflict-equivalent to S should have "T_2 precedes T_1 ".

Answer. $r_1(A); r_3(A); w_1(A); r_2(C); w_3(C)$

Precedence graph: T_2 -> T_3 -> T_1

[Binary marking - either 5 or 0]

Question-2: Consider the following schedule:

S1: $r_2(A); r_1(B); w_2(A); r_2(B); r_3(A); w_1(B); w_3(A); w_2(B)$

(a) [**1 mark**] What is the precedence graph for the schedule S1?

(b) [**2 marks**] Is the schedule conflict serializable? If yes, write all the equivalent serial schedules?

Answer: T_1 -> T_2; T_2 -> T_1; T_2 -> T_3. The graph has a cycle; schedule is not conflict serializable.

Question-3: [**3 marks**] Suppose there are eight transactions T1, T2, ..., T8, of which the odd-numbered transactions, T1, T3, T5 and T7, lock the root of the tree, in that order. There are three children of the root, first locked by T1, T2, T3 and T4 in that order. The second child is locked by T3, T6, and T5 in that order, and the third child is locked by T8 and T7, in that order.

Give one example of a serial order that is consistent with these statements if transactions are assumed to follow Tree protocol.

Answer. T1, T2, T3, T4, T6, T5, T8, T7

[Binary marking - either 3 or 0]

Question-4: [**3 marks**] Give an example of a schedule that is possible under the two-phase locking protocol, but not possible under timestamp protocol.

Answer.

Sno.	T1	T2
1	Lock-S(A)	
2	Read(A)	
3		Lock-X(B)
4		Write(B)
5		Unlock(B)
6	Lock-S(B)	
7	Read(B)	
8	Unlock (A)	
9	Unlock (B)	

This schedule is not allowed in the timestamp protocol because at step 7, the W-timestamp of *B* is 1.

[Binary marking - either 3 or 0]

Question-5: [**4 marks**] Consider a database organized in the form of a rooted tree. Suppose that we insert a dummy vertex between each pair of vertices. Show that, if we follow the tree protocol on the new tree, we get better concurrency than if we follow the tree protocol on the original tree.

Answer. Consider two nodes A and B, where A is a parent of B. Let dummy vertex D be added between A and B. Consider a case where transaction T2 has a lock on B, and T1, which has a lock on A wishes to lock B, and T3 wishes to lock A. With the original tree, T1 cannot release the lock on A until it gets the lock on B. With the modified tree, T1 can get a lock on D, and release the lock on A, which allows T3 to proceed while T1 waits for T2. Thus, the protocol allows locks on vertices to be released earlier to other transactions, instead of holding them when waiting for a lock on a child.

[Binary marking - either 4 or 0]

Question-6: [**2 marks**] Consider the schedule given below.

$r_1(A)$; $w_1(A)$; $r_2(A)$; $r_1(B)$; C1; C2;

C1 and C2 are the commit operations of transaction 1 and 2, respectively.

(a) Is the schedule recoverable? Justify your answer. **[1 mark]**
(b) Is the schedule cascadeless? Justify your answer. **[1 mark]**

Answer. The schedule is recoverable as transaction T_2 commits after T_1 only. It is not cascadeless are T_1 doesn't commit before the read (A) by T_2.