CS 162 Section 9

Short Answer

1. What does ACID stand for? Explain each of them.
   Atomicity: all transactions either finish to completion, or none happen

   Consistency: if each transaction is consistent, and the database starts in a consistent state, it will remain in a consistent state after committing the transactions

   Isolation: execution of one transaction is isolated from all others

   Durability: if a transaction commits, its effects persist

2. What are types of possible conflicts in an execution of multiple transactions? Express your answer in terms of Read and Writes.

   Read-write (unrepeatable reads)
   Write-read (reading uncommitted data)
   Write-write conflict (overwriting uncommitted data)

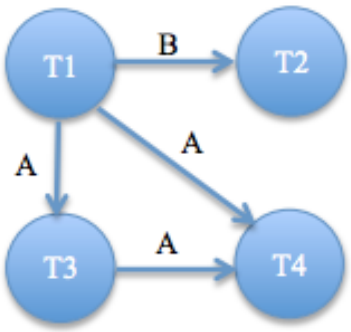
3. Two schedules are conflict equivalent if and only if:

   Involve the same operations of the same operations, such that every pair of conflicting operations is ordered the same way.

b. (7 points total) Consider the following schedule of four transactions:

T1:		R(B)	R(A)	W(A)			W(D)
T2:					W(B)		
T3:						W(A)	
T4:	W(C)						W(A)

i) (4 points) Draw the dependency graph for this schedule.



ii) (3 points) Is this schedule serializable? If so, give a possible serial schedule. Otherwise, explain why not.

Yes, because the dependency graph is acyclic.
Consider: 1, 2, 3, 4 or 1, 3, 4, 2 or 1, 3, 2, 4

Long Answer: 2PL

1. Quick Facts
   a. State the difference between 2PL and Strict 2PL.

In strict 2PL, we only release locks at the **end** of the transaction. In standard 2PL, we are allowed to release locks upon acquiring all of them.

   b. What condition does Strict 2PL avoid that 2PL does not?

With strict 2PL we avoid cascading aborts – if we abort a transaction, we only have to rollback that single transaction because we have not released any of its locks (and therefore waiting transactions could not have made any progress).

2. Consider the two transactions below. Assume each instruction takes one time unit, and acquiring/releasing a lock takes **zero time units**. Once a transaction acquires a shared lock it cannot upgrade it to an exclusive lock, and once a transaction acquires an exclusive lock it cannot downgrade to a shared lock.

Transaction 1

```
R(A);
A = A + 100;
W(A);
R(B);
B = B - 100;
W(B);
```

Transaction 2

```
R(A);
A = A - 50;
W(A);
```

What is the minimum possible execution time taken by both transactions when using 2PL? Show a schedule that achieves the minimum time. The diagram below shows the first several instructions – note that Transaction 2 is only **requesting** the lock, it has to wait for it to be released

Transaction 1: Lock_X(A) <granted> (0 time)	Transaction 2:
R(A);	Lock_X(A) <requested> (0 time)
A = A + 100;	
W(A);	
Lock_X(B) <granted> (0 time)	
Unlock(A) (0 time)	Lock_X(A) <granted> (0 time)
R(B);	R(A);
B = B – 100;	A = A – 50;
W(B);	W(A);
Unlock(B) (0 time)	Unlock(A) (0 time)

Total of 6 time units.