CS3402 Tutorial
Transactions (Part 2) and Concurrency (Part 1):

Question 1

Consider the following arrival order of operations to the scheduler. Assume that the scheduler uses strict two-phase locking (S2PL) and add lock/unlock operations in the table below to show the new schedule.

Ta	Tb	Tc
	Write(x)	
Read(y)		
	Read(z)	
		Read(x)
	Write(y)	
Write(x)		
	Read(x)	
	Commit	
		Write(z)
Commit		
		Commit

Question 2

Consider the following schedule.

T_1	T_2
Read(a)	
	Read(a)
Write(a)	
	Write(a)
Commit	
	Commit

(1) Add lock and unlock operations to the schedule assuming that Conservative 2PL (C2PL) is used.
(2) Add lock and unlock operations to the schedule assuming that Strict 2PL (S2PL) is used.
(3) Which one (S2PL or C2PL) is preferable for scheduling the two transactions?

Question 3

Consider the three schedules below. Determine whether each schedule is strict, cascadeless, recoverable, or nonrecoverable. Determine the strictest recoverability condition that each schedule satisfies.
(a) $r_1(X)$; $w_1(X)$; $r_2(X)$; $r_1(Y)$; $w_2(X)$; c_2; c_1;
(b) $r_1(X)$; $w_1(X)$; $r_2(X)$; $r_1(Y)$; $w_2(X)$; $w_1(Y)$; c_1; c_2;
(c) $r_1(X)$; $w_1(X)$; $w_2(X)$; $w_1(Y)$; c_1; $r_2(X)$; c_2;
(d) Make the schedule in (c) strict.

Answers

Question 1

Serial schedule: Tb Ta Tc

Ta	**Tb**	**Tc**
	WL(x); Write(x)	
RL(y); Read(y)		
	RL(z); Read(z)	
		RL(x) → blocked
	WL(y) → blocked	
WL(x) → blocked		

It is a deadlock.

Question 2
(1)

T_1	T_2
WriteLock(a)	
Read(a)	
	WriteLock(a) rejected → blocked
Write(a)	
Unlock(a)	
	WriteLock(a)
	Read(a)
	Write(a)
	Unlock(a)
Commit	
	Commit

(2)

T_1	T_2
ReadLock(a)	
Read(a)	
	ReadLock(a)
	Read(a)
WriteLock(a) rejected → blocked	
	WriteLock(a) rejected → blocked

(3)
In this case, C2PL is preferable since the transactions are short and it avoids running into a deadlock.

Question 3
(a) Non-recoverable, because T2 reads X written by T1，but T1 commits after T2.
If T1 abort after T2 commits, the value of X is not recoverable.

(b) Recoverable, because T2 reads X written by T1 and T2 commits after T1, which satisfy the condition of recoverable". (A schedule S is recoverable if no transaction T in S commits until all transactions T′ that have written some item X that T reads have committed."
It is not cascadeless, because T2 reads X written by T1 before T1 commits. If T1 fails, T1 has to be rolled back and T2 also need to be rolled back.

(c) Cascadeless. Because T2 reads X written by T1 after T1 commits, it satisfies the condition of a cascadeless schedule ("Every transaction reads only the items that are written by committed transactions.")
It is not a strict schedule because T2 writes X after T1 writes X but before T1 commits. Thus, it does not satisfy the condition of strict schedule. ("A schedule in which a transaction can neither read or write an item X until the last transaction that wrote X has committed.")

Schedule (c) can be changed into the strict schedule:
r1(X); w1(X); w1(Y); c1; w2(X); r2(X); c2;