CENG352 Written Assignment 3

1) For each of the following schedules, state which of the following properties hold: conflict serializable, recoverable, cascadeless, or strict.

Note that a conflict serializable schedule is one that has no conflicting operations among its committed transactions. Aborted transactions are ignored when determining conflict serializability.

a) R1(A) W1(B) R2(B) W2(A) W1(A) Commit T1 Commit T2
b) W1(A) R2(A) W1(A) Commit T2 Abort T1
c) W3(A) R2(A) Commit T3 W1(B) Commit T1 R2(B) W2(C) Commit T2
d) W1(A) R1(B) R2(B) Commit T1 R2(A) W2(B) Commit T2
e) W1(A) R2(B) R1(B) R2(A) Commit T1 Commit T2

2) Consider the following schedule:

R1(A) W1(A) R2(A) W2(A) R2(B) W2(B) T2:Commit, T1:Abort

Is this schedule allowed by **Two Phase Locking**? If not explain why. If so, show the schedule with lock requests and releases obeying 2PL.

3) Consider the same schedule:

R1(A) W1(A) R2(A) W2(A) R2(B) W2(B) T2:Commit, T1:Abort

Is this schedule allowed by **Strict Two Phase Locking**? If not explain why. If so, show the schedule with lock requests obeying Strict 2PL.

4)

Consider the following two transactions:
T1 = R(A) W(A) R(B) W(B)
T2 = R(B) R(A) W(A) W(B)

Assume that exclusive lock (X) and unlock (U) actions are inserted by the scheduler, resulting in the following annotated transactions:
T1 = X(A) R(A) W(A) X(B) R(B) W(B), AFTER COMMIT: U(A) U(B)
T2 = X(B) R(B) X(A) R(A) W(A) W(B), AFTER COMMIT: U(A) U(B)

Now, say that lock and unlock actions are inserted in the following way instead:
T1 = X(A) R(A) W(A) X(B) U(A) R(B) W(B) U(B)
T2 = X(A) X(B) R(B) R(A) W(A) U(A) W(B) U(B)

(d) Is conflict serializability guaranteed? Why or why not?

(e) Is deadlock possible? If so, then assuming that T1 starts first, which transaction(s) would be rolled back (aborted) under the wait-die deadlock prevention scheme?

(f) Is cascading rollback possible? If not, explain why not. If so, show a scenario that results in cascading rollback.

Consider concurrent execution of these two transactions and answer the following questions:

(a) Is conflict serializability guaranteed? Why or why not?

(b) Is deadlock possible? If so, then assuming that T1 starts first, which transaction(s) would be rolled back (aborted) under the wait-die deadlock prevention scheme?

(c) Is cascading rollback possible? If not, explain why not. If so, show a scenario that results in cascading rollback.

5)

Consider the schedule below. The symbol $r_i(x)$ stands for a read by transaction T_i to item x, $w_i(x)$ stands for a write by T_i to item x and c_i stand for the commit of T_i. Suppose **timestamp-based scheduler** is used as the concurrency control protocol.

$r_1(A), r_2(B), r_3(A), w_1(A), r_2(C), w_3(B), w_2(C), c_1, r_2(A), w_3(C), c_3, w_2(B), c_2$

Operation	A			B			C		
	RTS	WTS	C	RTS	WTS	C	RTS	WTS	C

a Use the above table to give describe what happens at each operation for the given timestamps:

i. TS(T1)=1, TS(T2)=2, TS(T3)=3.

ii. TS(T1)=2, TS(T2)=3, TS(T3)=1.

Justify whether the operation is accepted or rejected and the RTS, WTS and C (commit bit) of the data items after the operation is executed (or rejected).

b Explain the importance of the commit bit, what could have changed if we don't use the commit bit.