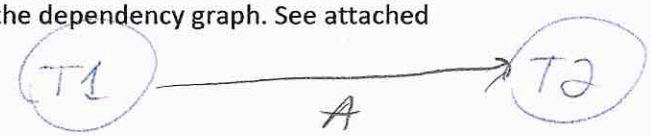
2.a.i:

We need to look at the dependency graph, and here, transaction 2 does not make any writes so there can be no cycles in the dependency graph. See attached



2.a.ii

Transaction 1	Transaction 2	Lock Requests
		X1(A) granted
R1(A)		S2(A) wait
W1(A)		
		X1(A) released
		S2(A) granted
	R2(A)	
		X1(B) wait
		S2(A) released
		S2(B) granted
	R2(B)	
		S2(B) released
	C2	
		X1(B) granted
R1(B)		
W1(B)		
		X1(B) released
C1		

2.a.iii

Yes this can result in a deadlock

- T2 requests exclusive lock on A that is granted
- T3 Requests an exclusive lock on D and It is granted
- T4 requests exclusive lock on A, but must wait
- T3 Writes D
- T3 releases the exclusive lock on D
- T3 Requests an exclusive lock on A, but also must wait
- T2 writes A
- T2 releases the exclusive lock on A
- Both T3 and T4 depend on A at this point, so we have a deadlock

Q2b. Timestamp-based Concurrency Control (10 pts)

Consider the following schedule. Show what happens when transactions try to execute actions in this schedule and time-stamp-based CC is used:

$$R_1(X);\ R_2(X);\ W_2(X);\ W_1(X);\ W_3(Y);\ W_2(Y);\ C_3;\ W_4(Z);\ C_4;\ R_2(Z)$$

- Show the updates of RT (read timestamp), WT (write timestamp), C (commit bit) in the respective column for X, Y, Z.
- If any request is delayed, ignored, or transaction is aborted, write DELAY, IGNORE, ABORT in the respective action columns for T1, T2, T3, T4.
- Timestamps of each transaction is written right below the transaction.

Step	T1	T2	T3	T4	X	Y	Z
	1	2	3	4	RT = 0 WT = 0 C = 1	RT = 0 WT = 0 C = 1	RT = 0 WT = 0 C = 1
1	$R_1(X)$				RT = 1		
2		$R_2(X)$			RT = 2		
3		$W_2(X)$			WT = 2 C = 0		
4	$W_1(X)$ ABORT						
5			$W_3(Y)$			WT = 3	
6		$W_2(Y)$ IGNORE					
7			C_3			FINAL	
8				$W_4(Z)$			WT = 4
9				C_4			
10		$R_2(Z)$					RT = 2
11							
12							
13							

Q2c. Multiversion Concurrency Control (8 pts)

Consider the following schedule. Show what happens when transactions try to execute actions in this schedule and multiversion CC is used:

$$R_1(X);\ R_2(X);\ W_2(X);\ W_1(X);\ W_4(Z);\ R_2(Z)$$

- Show the updates of RT (read timestamp), WT (write timestamp), and Create for each of X and Z.
- If any request is delayed, ignored, or transaction is aborted, write DELAY, IGNORE, ABORT in the respective action columns for T1, T2, T4.
- Timestamps of each transaction is written right below the transaction.
- Initially we have X_0 and Z_0. Add new versions in the empty columns as necessary.

Step	T1	T2	T4	X_0	Z_0	X_2	Z_4	
	1	2	4	RT = 0 WT = 0	RT = 0 WT = 0	RT = 2 WT = 2	RT = 4 WT = 4	
1	$R_1(X)$ ✓			RT = 1 READ				
		$R_2(X)$				CREATE RT = 2 WT = 2		
		$W_2(X)$				WAITS WT = 2		
	$W_1(X)$ IGNORE							
			$W_4(Z)$				CREATE Z4	
		$R_2(Z)$			READ RT = 2			

WT
$\geq TS(T_1)$

$RT(A_0) < R_2(X)$

$WT(X_2) \leq TS(T_2)$

$W_1(X) \geq WT(X)$

Q2d. UNDO logging (1 + 2 + 3 + 1 = 7 pts)

Consider the following nonquiescent UNDO log. Suppose a crash happens after step 11.

UNCOMP

UNCOMP.

Step	Log record
1	⟨ START T1 ⟩
2	⟨ T1, X, 15 ⟩
3	⟨ START T2 ⟩ ←
4	⟨ T1, Y, 21 ⟩
5	⟨ T2, X, 27 ⟩
6	⟨ START T3 ⟩
7	⟨ T3, Z, 33 ⟩
8	⟨ COMMIT T1 ⟩ ← ONLY COMMIT
9	⟨ START CKPT(T2, T3) ⟩
10	⟨ T2, X, 39 ⟩
11	⟨ T3, Y, 45 ⟩

(i) Does the recovery manager read the log forward (starting from 1) or backward (starting from 11)?

BACKWARD

(ii) What is the earliest (lowest possible numbered) step that the recovery manager reads? Explain.

(3) T1 IS THE ONLY TRANSACTION THAT COMMITS SO IT MUST FIND WHERE THE NEXT TRANSACTION STARTS.

(iii) What is the sequence of updates (in order) by the recovery manager? e.g. write $A = 5$ if A is set to 5 during recovery.

1. Y = 45
2. X = 39
3. Z = 33
4. X = 27
5. Y = 21
6. ..... DONE

(iv) What is the value of X after recovery?

(27)

Q2e. REDO logging (3 + 4 + 3 = 10 pts)

Consider the following REDO log. Suppose a crash happens after step 20.

Step	Log record
1	⟨ START T1 ⟩
2	⟨ T1, X, 14 ⟩
3	⟨ START T2 ⟩
4	⟨ T2, Y, 9 ⟩
5	⟨ T1, Z, 11 ⟩
6	⟨ START T3 ⟩
7	⟨ T3, W, 16 ⟩
8	⟨ COMMIT T1 ⟩
9	⟨ START CKPT(??????) ⟩
10	⟨ START T4 ⟩
11	⟨ T2, U, 9 ⟩
12	⟨ COMMIT T2 ⟩
13	⟨ T3, V, 5 ⟩
14	⟨ T4, M, 19 ⟩
15	⟨ END CKPT ⟩
16	⟨ COMMIT T3 ⟩
17	⟨ START T5 ⟩
18	⟨ T5, N, 7 ⟩
19	⟨ START CKPT(??????) ⟩
20	⟨ COMMIT T5 ⟩

FORWARD DIRECTION

T1 DONE

(i) What are the correct values of ??????? in the two ⟨ START CKPT .... ⟩ in step 9 and 19?

- Step 9: 2,3 (BECAUSE 4 STARTS AFTERWARDS)
- Step 19: 4,5 (BECAUSE 6 STARTS AFTERWARDS)

(ii) What fraction of the log the recovery manager reads (mention start and end step)? Explain your answer. DOES

FIND THE FIRST END CHECKPOINT (15)

AND THE START OF THE CHECKPOINT (16)

(ANYTHING BEFORE WILL HAVE ALREADY BEEN COMMITTED)

(iii) What is the sequence of updates (in order) by the recovery manager? e.g. write $A = 5$ if A is set to 5 during recovery.

1. $V = 9$
2. COMMIT T2
3. $V = 5$
4. $M = 19$
5.
6. .....