VITMAB04 – Databases – Tutorial 5

Dr János Varga – 24 November 2022 – Transaction Management

Exercise 1. Is the schedule on Table 2 legal? If it is not, what should be changed to make it legal?

Exercise 2. Check if the schedule in Table 1 is legal. Draw the precedence graph and determine if the schedule is serializable. If it is, give its serial equivalent. If it is not, show why not. How does it change the graph if we use the simple LOCK-UNLOCK transaction model?

Exercise 3. Is the schedule in Table 3 legal? Do transactions follow the 2PL protocol? Where are the synchronization points of the transactions? Can you describe a serialized equivalent schedule?

Exercise 4. A system uses timestamp-based concurrency control with a Read-Write lock model. Give the values of R(A), R(B), W(A), W(B) after every operation, all are initially 0. Which transactions abort? r_i and w_i designate (r)ead and (w)rite operations of transaction T_i, and $\mathrm{t}(T_i) = \mathrm{i}$.

$$r_1(A), r_2(B), r_1(B), w_3(B), r_2(B), w_4(A), r_4(B), w_1(A), w_3(B)$$

Exercise 5. Our system suffered a fatal failure. The end of the transaction log looks like Table 5. What are the correct steps of Redo restore? What will be the values of A, B and C after the restore operation is finished?

Exercise 6. Is the schedule in Table 4 legal? Draw the precedence graph and determine if the schedule is serializable. If it is, give its serial equivalent. If it is not, show why not. How does it change the graph if we use the RLOCK-WLOCK transaction model? (When reading we replace LOCK with RLOCK; when we write (or read and write) we replace LOCK with WLOCK.)

Exercise 7. A transaction wants to access data items D, I and J in the hierarchy shown in Figure 1, then stop. What kind of locks will the lock manager have to place and free, and where in the hierarchy, if it wants to maintain as few locks as possible?

Theory / Brainteasers

Exercise 8. How does 2PL guarantee that the system will avoid deadlocks?

Exercise 9. Is it true that in a system using 2PL all transactions execute correctly? (What is ,,correct"?)

Exercise 10. Can one concurrently modify a table on which there is a B* tree index? When can the system release the lock on the root element?

Exercise 11. Why does serializability matter?

Exercise 12. If the transaction log contains all information about data changes, why do we need the database?

Exercise 13. What is the obvious serial equivalent of any set of transactions in a system using 2PL protocol and timestamp-based scheduling?

Exercise 14. When is it beneficial to use 2PL protocol or timestamp-based scheduling?

Exercise 15. What happens to the precedence graph if a transaction aborts?

Exercise 16. How does timestamp-based concurrency control protect a system against deadlocks?

Exercise 17. Give an example or counter-example to the following cases!

1. Timestamp-based scheduling, a transaction tries to place READ and aborts.
2. Timestamp-based scheduling, a transaction tries to place WRITE and aborts.
3. Timestamp-based scheduling with multiversions. A transaction tries to place READ and aborts.
4. Timestamp-based scheduling with multiversions. A transaction tries to place WRITE and aborts.

Exercise 18. How does a database store timestamps? How many should the system expect to have? How long does the system have to maintain them?

	T_1	T_2	T_3	T_4
(1)		RLOCK F		
(2)	RLOCK A			
(3)			RLOCK D	
(4)			UNLOCK D	
(5)	UNLOCK A			
(6)			WLOCK B	
(7)	RLOCK D			
(8)				RLOCK A
(9)	WLOCK E			
(10)				UNLOCK A
(11)			UNLOCK B	
(12)				WLOCK A
(13)	UNLOCK D			
(14)	RLOCK C			
(15)				UNLOCK A
(16)		UNLOCK F		
(17)			WLOCK A	
(18)		WLOCK B		
(19)	UNLOCK E			
(20)	UNLOCK C			
(21)				RLOCK D
(22)				UNLOCK D
(23)			WLOCK C	
(24)			UNLOCK C	
(25)		WLOCK D		
(26)			UNLOCK A	
(27)		RLOCK E		
(28)		UNLOCK E		
(29)				RLOCK E
(30)		UNLOCK B		
(31)	RLOCK F			
(32)		UNLOCK D		
(33)	UNLOCK F			
(34)				UNLOCK E

Table 1

	T_1	T_2	T_3
(1)			RLOCK B
(2)			READ B
(3)		WLOCK B	
(4)			RLOCK A
(5)	RLOCK A		
(6)		WRITE B	
(7)			READ A
(8)	READ A		
(9)	UNLOCK A		
(10)			UNLOCK B
(11)			UNLOCK A

Table 2

	T_1	T_2	T_3
(1)	LOCK A		
(2)		LOCK B	
(3)			LOCK C
(4)			LOCK D
(5)	LOCK E		
(6)	UNLOCK A		
(7)			UNLOCK D
(8)		LOCK A	
(9)		LOCK D	
(10)	UNLOCK E		
(11)		UNLOCK B	
(12)			UNLOCK C
(13)		UNLOCK A	
(14)		UNLOCK D	

Table 3

	T_1	T_2	T_3	T_4
(1)	LOCK A			
(2)		LOCK B		
(3)	READ A			
(4)	UNLOCK A			
(5)		WRITE B		
(6)		LOCK C		
(7)			LOCK A	
(8)		READ C		
(9)			READ A	
(10)		UNLOCK C		
(11)			UNLOCK A	
(12)			LOCK C	
(13)		UNLOCK B		
(14)			READ C	
(15)	LOCK A			
(16)				LOCK B
(17)	READ A			
(18)				READ B
(19)			UNLOCK C	
(20)				UNLOCK B
(21)			LOCK B	
(22)	WRITE A			
(23)	UNLOCK A			
(24)			READ B	
(25)			UNLOCK B	
(26)		LOCK A		
(27)				LOCK B
(28)		WRITE A		
(29)				WRITE B
(30)		UNLOCK A		
(31)				UNLOCK B

Table 4

checkpoint
(T_1, begin)
(T_2, begin)
(T_2, A, 20)
(T_2, B, 10)
(T_1, A, 2)
(T_3, begin)
(T_1, C, 5)
(T_1, commit)
(T_3, C, 6)
(T_3, commit)

Table 5

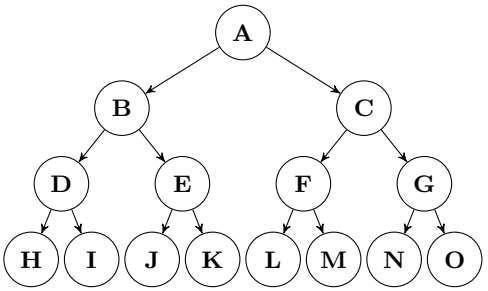

Figure 1: Locking on a hierarchy of items