Concurrency Control

Concurrency Control: To ensure that the system must control the interaction among the concurrent transactions.

Lock-Based Protocol: It is used to ensure that while one transaction is accessing a data item, no other transaction can modify that data item

Eg; Consider the transaction T_1.

T_1: Transfer Rs. 500/- from account B to account A.

T_1 (without lock)

```
read (B)
B = B-500
write (B)
read (A)
A = A+500
write (A)
```

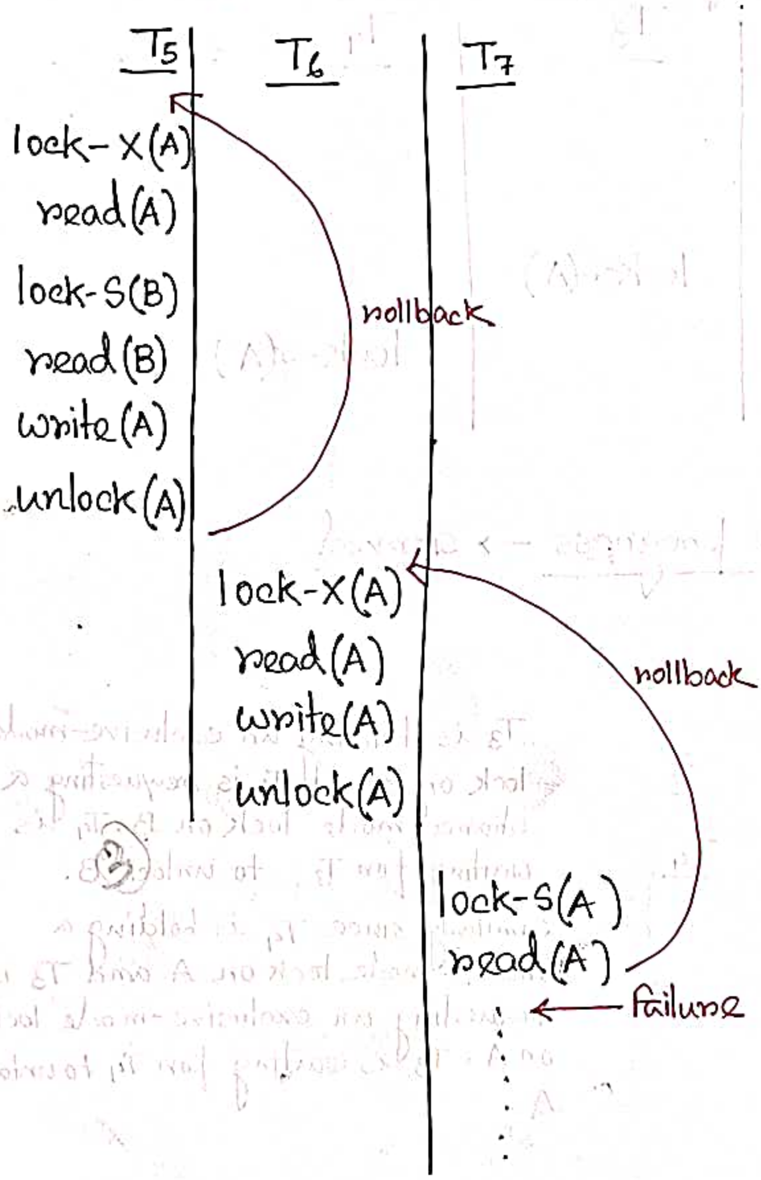
T_1 (with lock)

```
lock-X (B)
read (B)
B = B-500
write (B)
unlock (B)
lock-X (A)
read (A)
A = A+500
write (A)
unlock (A)
```

1. **Shared**: If a transaction T_i has obtained a shared-mode lock (denoted by S) on item Q, then T_i can read, but cannot write, Q

2. **Exclusive**: If a transaction T_i has obtained an exclusive-mode lock (denoted by X) on item Q, then T_i can both read and write Q.

Cascading Rollback :-



T5	T6	T7
lock-X(A)		
read(A)		
lock-S(B)		
read(B)		
write(A)		
unlock(A)		
	lock-X(A)	
	read(A)	
	write(A)	
	unlock(A)	
		lock-S(A)
		read(A)
		⋮

Failure of T5 after the read(A) step of T7 leads to cascading rollback of T6 and T7.

Solution: Strict two-phase locking protocol.

This protocol requires not only that locking be two phase, but also that all exclusive-mode locks taken by transaction be held until the transaction commits.

arvation:

T1 has to wait for T2 to release the shared-mode lock. Meanwhile transaction T3 may request a shared-mode lock on the same data item. The lock request is compatible so, T3 may be granted shared mode lock.

T_1	T_2	T_3	T_4
	lock-S(A)		
lock-X(A)			
		lock-S(A)	
			lock-S(A)

** T1 may never make progress → starved

Deadlock:

T3 is holding an exclusive-mode lock on B and T4 is requesting a shared mode lock on B, T4 is waiting for T3 to unlock B. Similarly, since T4 is holding a shared-mode lock on A and T3 is requesting an exclusive-mode lock on A, T3 is waiting for T4 to unlock A.

T_3	T_4
lock-X(B)	
read(B)	
B = B - 500	
write(B)	
	lock-S(A)
	read(A)
	lock-S(B)
lock-X(A)	

Solution: The system must roll back one of the two transactions. Once a transaction has been rolled back, the data items that were locked by that transaction are unlocked.

T_1	T_2
lock-X(B)	
read(B)	
B = B - 500	
write(B)	
lock-X(A)	
read(A)	
A = A + 500	
write(A)	
unlock(B)	
unlock(A)	
	lock-S(A)
	read(A)
	lock-S(B)
	read(B)
	display(A+B)
	unlock(A)
	unlock(B)

Two-Phase locking Protocol (2PL) : Protocol that ensures serializability.

1. Growing phase: A transaction may obtain locks, but may not release any lock

2. Shrinking phase: A transaction may release locks, but may not obtain any new locks.

** Two phase locking protocol ensures conflict serializability

T₁	T₂
lock-X(B)	
read(B)	
B = B-500	
write(B)	
unlock(B)	
	lock-S(A)
	read(A)
	unlock(A)
	lock-S(B)
	read(B)
	unlock(B)
	display(A+B)
lock-X(A)	
read(A)	
A = A+500	
write(A)	
unlock(A)	

	S	X
S	true	false
X	false	false

Lock compatibility matrix

** Shared mode is compatible with shared mode, but the other cases are incompatible.

T2: Display the total amount of money in accounts A and

T2

lock-S(A) ⟶ Shared lock
read (A)
unlock (A)
lock-S(B) ⟶ shared lock
read (B)
unlock (B)
display (A+B)