Lock-based Protocols or lock management

LOCK

LOCK TABLE

Transaction ID	Data item id	lock mode	Ptr to next data item
T1	X1	Read	Next

Lock manager

Types of locks: binary and shared/exclusive locks

Lock conversions

Two Phase Locking

LOCK

LOCK TABLE

Transaction ID	Data item id	lock mode	Ptr to next data item
T1	X1	Read	Next

Lock manager

Types of locks: binary and shared/exclusive locks

Lock conversions

2PL or basic 2pl :

A transaction is said to follow Two Phase Locking protocol if Locking and Unlocking can be done in two phases.

1. **Growing Phase:** New locks on data items may be acquired but none can be released.
2. **Shrinking Phase:** Existing locks may be released but no new locks can be acquired.
3.

Note – If lock conversion is allowed, then upgrading of lock(from S(a) to X(a)) is allowed in Growing Phase and downgrading of lock (from X(a) to S(a)) must be done in shrinking phase.

Let’s see a transaction implementing 2-PL.

	T_1	T_2
1	LOCK-S(A)	
2		LOCK-S(A)
3	LOCK-X(B)	
4	…….	……
5	UNLOCK(A)	
6		LOCK-X(C)
7	UNLOCK(B)	
8		UNLOCK(A)
9		UNLOCK(C)
10	…….	……

This is just a skeleton transaction which shows how unlocking and locking works with 2-PL. Note for:

Transaction T_1:

- Growing Phase is from steps 1-3.
- Shrinking Phase is from steps 5-7.
- Lock Point at 3
-

Transaction T_2:

- Growing Phase is from steps 2-6.
- Shrinking Phase is from steps 8-9.
- Lock Point at 6

What is **LOCK POINT ?**The Point at which the growing phase ends, i.e., when transaction takes the final lock it needs to carry on its work.

I have said that 2-PL ensures serializablity, but there are still some drawbacks of 2-PL. Let's glance at the drawbacks:

- Cascading Rollback is possible under 2-PL.
- Deadlocks and Starvation is possible.

The above mentioned type of 2-PL is called **Basic 2PL**.

Strict-2PL:

Strict-2PL does not release a lock after using it.

Strict-2PL holds all the locks until the commit point and releases all the locks at a time.

Conservative 2 PL: A transaction has to lock all the items it access before the transaction begins execution.