SNS COLLEGE OF ENGINEERING

Kurumbapalayam (Po), Coimbatore – 641 107

An Autonomous Institution

Accredited by NAAC – UGC with 'A' Grade

Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

COURSE NAME : 23CST207 - DATABASE MANAGEMENT SYSTEMS

II YEAR / IV SEMESTER

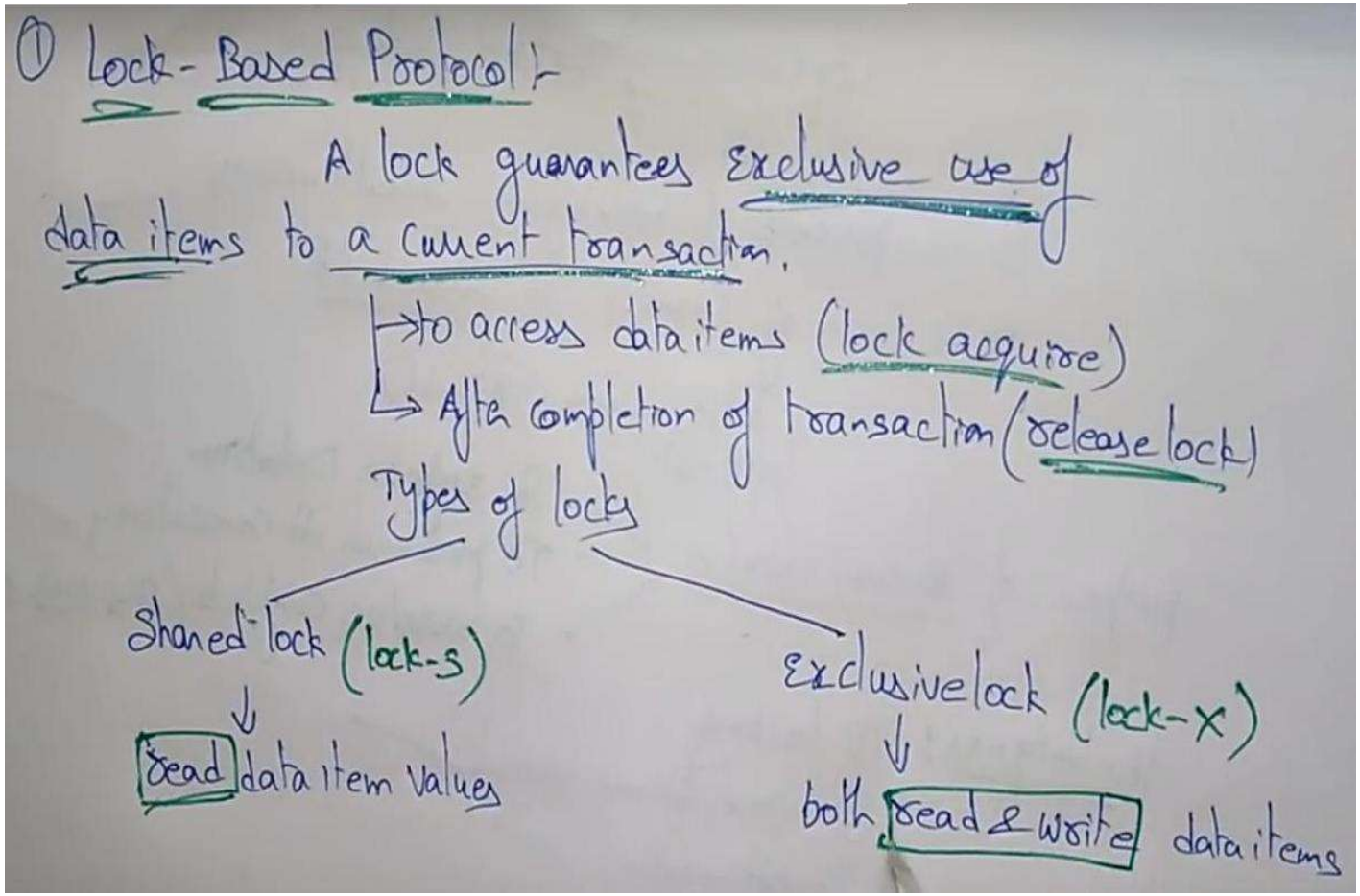
Unit 4- Transactions

Topic 5 : Lock – Based Protocol

Concurrency Control Technique

- Locking
- Time Stamp Based
- Optimistic
- Multi Version

Lock – Based Protocol



Lock-based Protocols

- A lock is a data variable which is associated with a data item. This lock signifies that operations that can be performed on the data item.
- Locks help synchronize access to the database items by concurrent transactions.
- All lock requests are made to the concurrency-control manager. Transactions proceed only once the lock request is granted.

Cont..

- A lock is a mechanism to control concurrent access to a data item
- Data items can be locked in two modes :

  1. *exclusive (X) mode*. Data item can be both read as well as written. X-lock is requested using **lock-X** instruction.
  2. *shared (S) mode*. Data item can only be read. S-lock is requested using **lock-S** instruction.

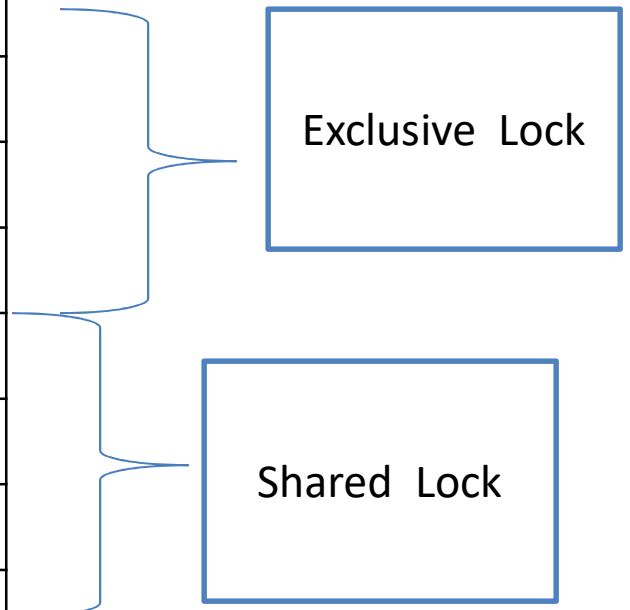
Binary Locks: A Binary lock on a data item can either locked or unlocked states.

Cont..

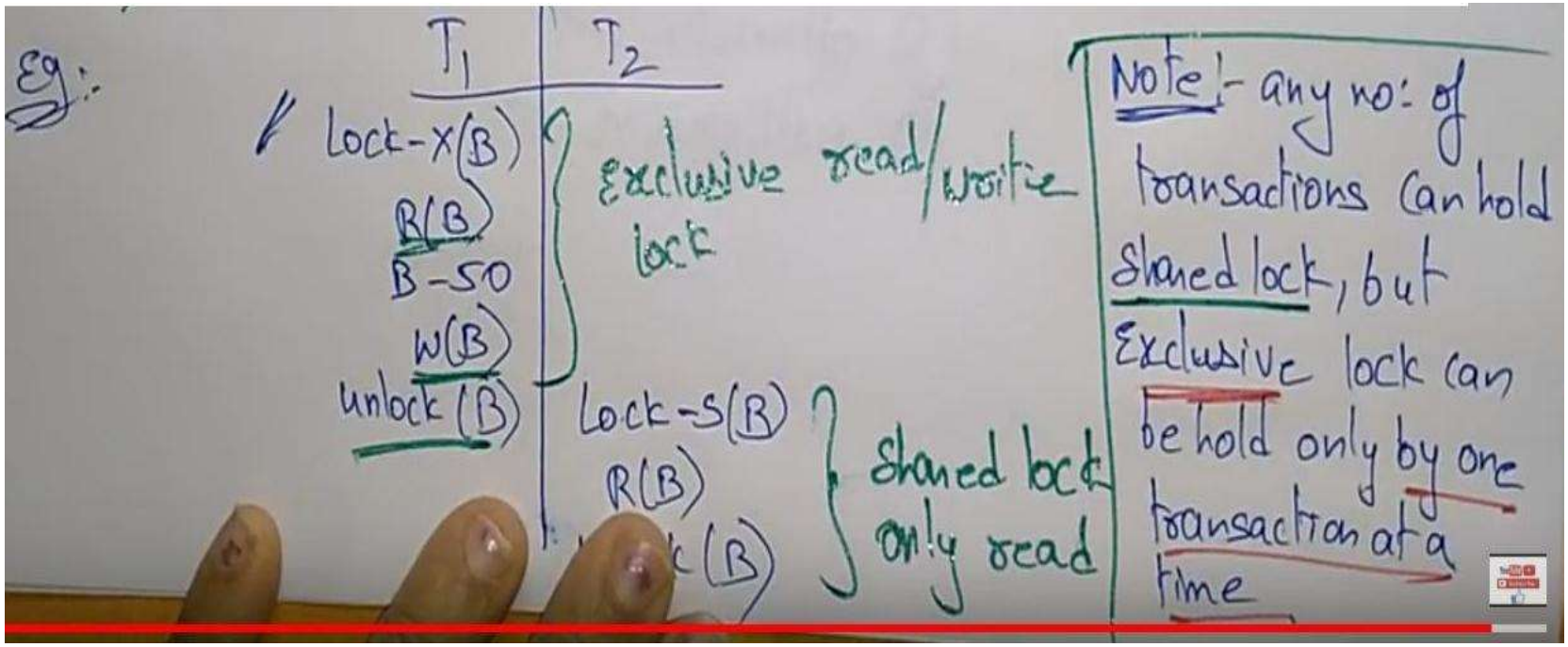
- **Shared Lock (S):**
- A shared lock is also called a Read-only lock. With the shared lock, the data item can be shared between transactions
- **Exclusive Lock (X):**
- With the Exclusive Lock, a data item can be read as well as written.

Example

	T_1	T_2
1	lock-X(B)	
2	read(B)	
3	B:=B-50	
4	write(B)	
5		lock-S(A)
6		read(A)
7		lock-S(B)
8	lock-X(A)	



Cont..


Eg:-
T1
T2
Lock-X(B)
R(B)
B-50
W(B)
Unlock(B)
Exclusive read/write lock
Lock-S(B)
R(B)
Shared lock only read
Note!- any no: of transactions can hold Shared lock, but Exclusive lock can be hold only by one transaction at a time

The Two-Phase Locking Protocol

- This protocol ensures conflict-serializable schedules.
- Phase 1: Growing Phase
  - Transaction may obtain locks
  - Transaction may not release locks
- Phase 2: Shrinking Phase
  - Transaction may release locks
  - Transaction may not obtain locks
- The protocol assures serializability. It can be proved that the transactions can be serialized in the order of their **lock points** (i.e., the point where a transaction acquired its final lock).

Cont..

- two-phase locking is needed for conflict serializability in the following sense:
  - Given a transaction T_i that does not follow two-phase locking, we can find a transaction T_j that uses two-phase locking, and a schedule for T_i and T_j that is not conflict serializable.

Lock Conversion

- Two-phase locking with lock conversions:
  - First Phase:
  - can acquire a lock-S on item
  - can acquire a lock-X on item
  - can convert a lock-S to a lock-X (upgrade)
  - Second Phase:
  - can release a lock-S
  - can release a lock-X
  - can convert a lock-X to a lock-S (downgrade)
- This protocol assures serializability. But still relies on the programmer to insert the various locking instructions.

Automatic Acquisition of Locks

- A transaction T_i issues the standard read/write instruction, without explicit locking calls.
- The operation **read**(*D*) is processed as:

if T_i has a lock on *D*
then
read(*D*)
else begin
if necessary wait until no other transaction has a **lock-X** on *D*
grant T_i a **lock-S** on *D*;
read(*D*)
end

Cont..

- **write***(D)* is processed as:

if T_i has a **lock-X** on *D*
then
write(*D*)
else begin
if necessary wait until no other transaction has any lock on *D*,
if T_i has a **lock-S** on *D*
then
upgrade lock on *D* to **lock-X**
else
grant T_i a **lock-X** on *D*
write(*D*)
end;

- All locks are released after commit or abort

Activity

Connection

Ans:_____

Thank you