COURSE NAME: DBMS
COURSE CODE:23AD2102A

TOPIC:

CONCURRENCY CONTROL-I

Session - 4

AIM OF THE SESSION

To familiarize students with the Basic concepts about Concurrency Techinques and Locking Protocols

INSTRUCTIONAL OBJECTIVES

This Session is designed to:

1. Discuss about Terminology of Concurrency and Locking Protocols.
2. Describe the Locking Protocols.
3. Describe Two phase locking with suitable example.
4. Discuss about Deadlock prevention in locking .

LEARNING OUTCOMES

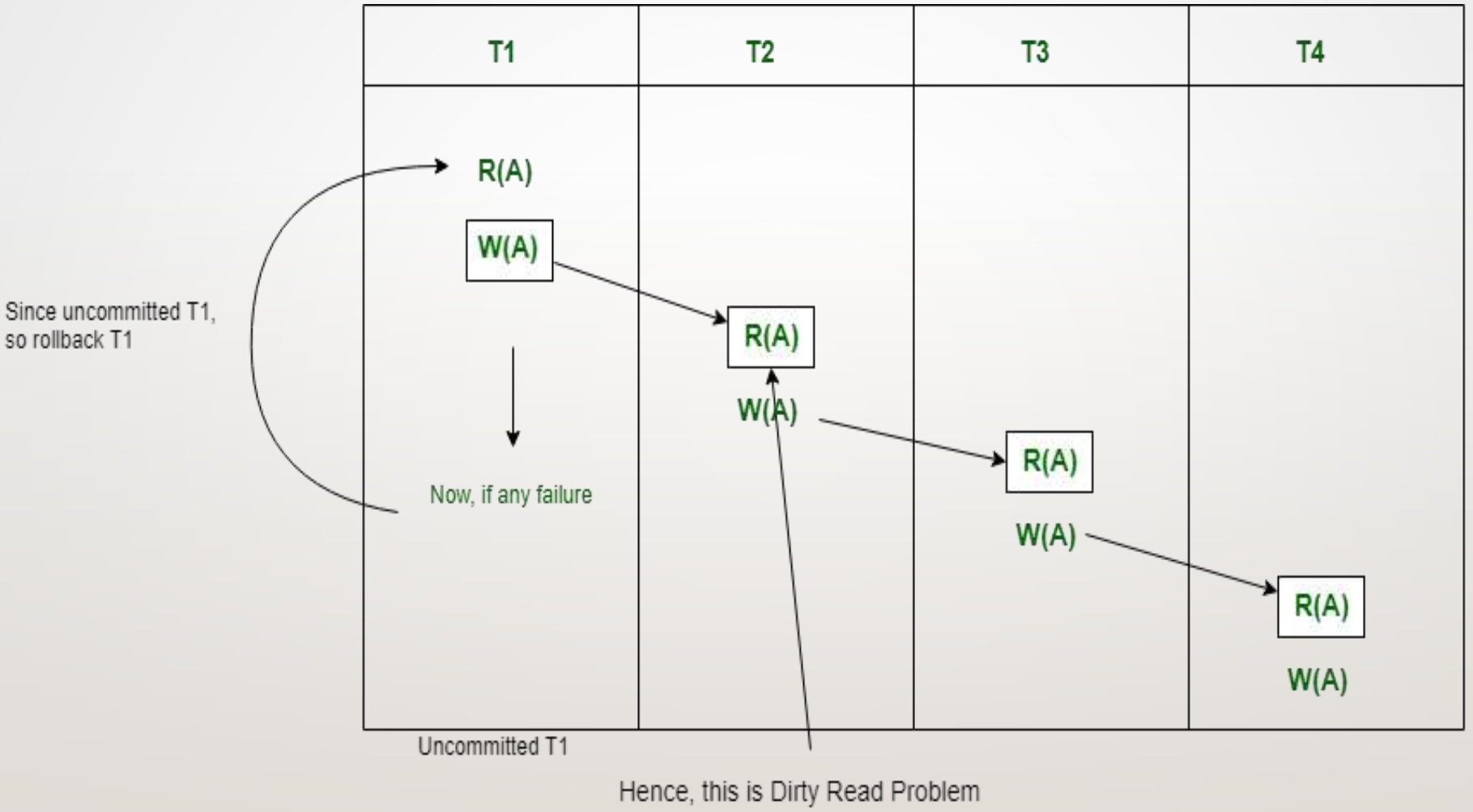
At the end of this session, students should be able to:

1. Review of Cascading Rollback & Cascade less Schedule
2. Concurrency Control: Lock based Protocols

REVIEW OF CASCADING ROLLBACK

- If in a schedule, failure of one transaction causes several other dependent transactions to rollback or abort, then such a schedule is called as a Cascading Rollback or Cascading Abort or Cascading Schedule. It simply leads to the wastage of CPU time.
  These Cascading Rollbacks occur because of **Dirty Read problems**.
- For example, transaction T1 writes uncommitted x that is read by Transaction T2.
- Transaction T2 writes uncommitted x that is read by Transaction T3
  Suppose at this point T1 fails.
  T1 must be rolled back, since T2 is dependent on T1, T2 must be rolled back, and since T3 is dependent on T2, T3 must be rolled back

CASCADING ROLLBACK DUE TO DIRTY READ PROBLEM

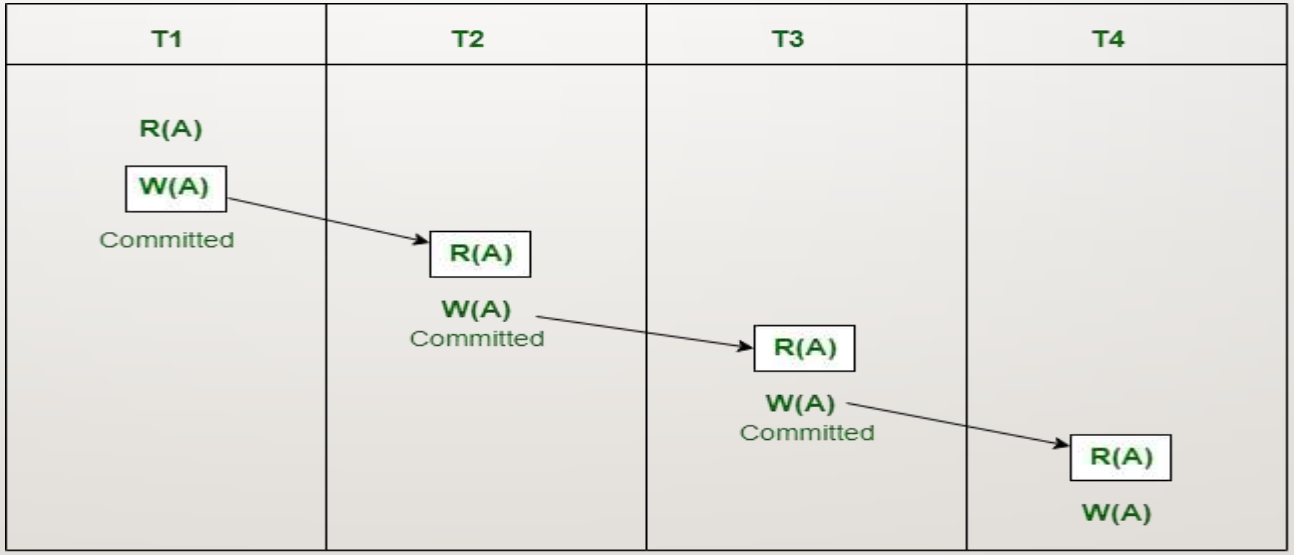


CASCADING ROLLBACK DUE TO DIRTY READ PROBLEM

- Because of T1 rollback, all T2, T3, and T4 should also be rollback (Cascading dirty read problem).
- This phenomenon, in which a single transaction failure leads to a series of transaction rollbacks is called **Cascading rollback**.

REVIEW OF CASCADE LESS SCHEDULE

- This schedule avoids all possible *Dirty Read Problem*.
- In Cascadeless Schedule, if a transaction is going to perform read operation on a value, it has to wait until the transaction who is performing write on that value commits. That means there must not be **Dirty Read**. Because Dirty Read Problem can cause *Cascading Rollback*, which is inefficient.



No Dirty Read problem

REVIEW OF CASCADELESS SCHEDULE

- Cascadeless Schedule avoids cascading aborts/rollbacks (ACA). Schedules in which transactions read values only after all transactions whose changes they are going to read commit are called cascadeless schedules.
- Avoids that a single transaction abort leads to a series of transaction rollbacks.
- A strategy to prevent cascading aborts is to disallow a transaction from reading uncommitted changes from another transaction in the same schedule.

CONCURRENCY CONTROL

- Concurrency Control is the management procedure that is required for controlling concurrent execution of the operations that take place on a database.
- Problems with Concurrent Execution

  a***) Dirty read Problem***

  b)Lost update Problems

  c)Unrepeatable read Problems

Thus, in order to maintain consistency in the database and avoid such problems that take place in concurrent execution, management is needed, and that is where the concept of Concurrency Control comes into role.

CONCURRENCY CONTROL PROTOCOLS

- The concurrency control protocols ensure the *atomicity, consistency, isolation, durability* and *serializability* of the concurrent execution of the database transactions. Therefore, these protocols are categorized as:

- ***Lock Based Concurrency Control Protocol***
- ***Time Stamp Concurrency Control Protocol***
- ***Validation Based Concurrency Control Protocol***

LOCK BASED PROTOCOLS-INTRODUCTION

- While performing concurrent transactions them those transactions MUST be ***isolated***, otherwise data consistency will be lost.
- Oneway to ensure isolation is to require that data items be accessed in a mutually exclusive manner; that is, while one transaction is accessing a data item, no other transaction can modify that data item.
- The most common method used to implement this requirement is to allow a transaction to access a data item only if it is currently holding a ***lock*** on that item.

LOCK BASED PROTOCOL

In this type of protocol, any transaction cannot read or write data until it acquires an appropriate lock on it. There are two types of locking Mechanisms:

1. Shared lock, S:

- It is also known as a Read-only lock. In a shared lock, the data item can only read by the transaction.
- It can be shared between the transactions because when the transaction holds a lock, then it can't update the data on the data item.

2. Exclusive lock,X:

- In the exclusive lock, the data item can be both reads as well as written by the transaction.
- This lock is exclusive, and in this lock, multiple transactions do not modify the same data simultaneously.

TYPES OF LOCK BASED PROTOCOLS

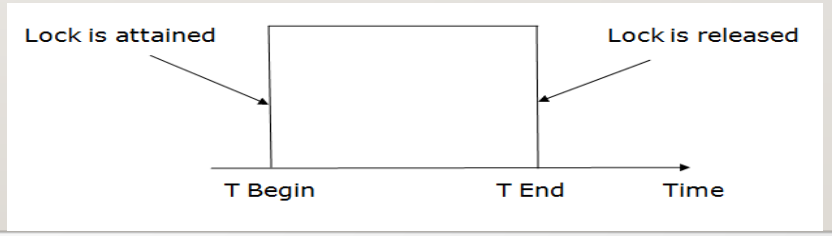
- **1. Simplistic lock protocol**

It is the simplest way of locking the data while transaction. Simplistic lock-based protocols allow all the transactions to get the lock on the data before insert or delete or update on it. It will unlock the data item after completing the transaction.

TYPES OF LOCK BASED PROTOCOLS

2. Pre-claiming Lock Protocol

- Pre-claiming Lock Protocols evaluate the transaction to list all the data items on which they need locks.
- Before initiating an execution of the transaction, it requests DBMS for all the lock on all those data items.
- If all the locks are granted then this protocol allows the transaction to begin. When the transaction is completed then it releases all the lock.
- If all the locks are not granted then this protocol allows the transaction to rolls back and waits until all the locks are granted.

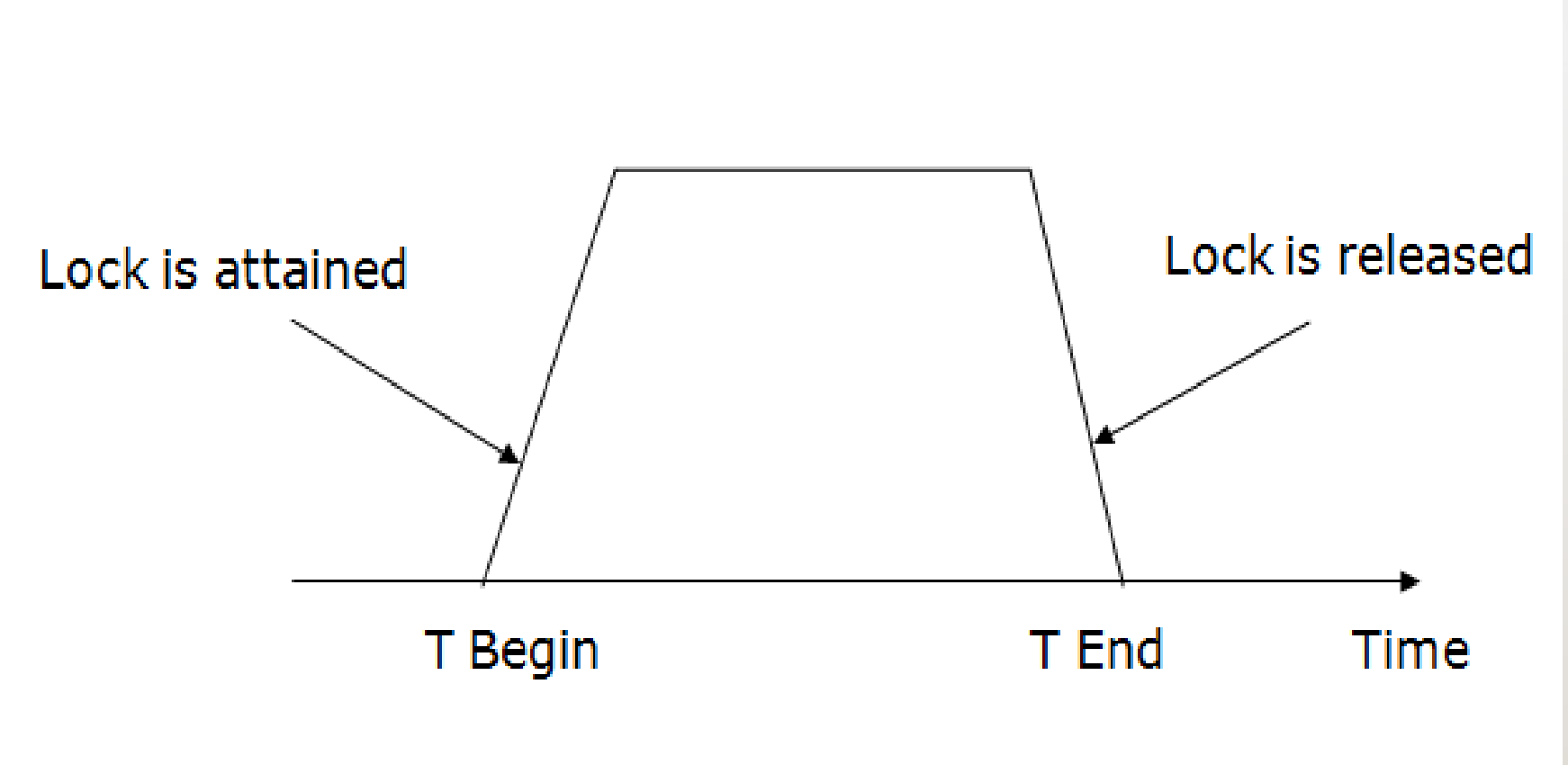


TYPES OF LOCK BASED PROTOCOLS

3. Two-Phase locking (2PL)

- The two-phase locking protocol divides the execution phase of the transaction into three parts.
- In the first part, when the execution of the transaction starts, it seeks permission for the lock it requires.
- In the second part, the transaction acquires all the locks. The third phase is started as soon as the transaction releases its first lock.
- In the third phase, the transaction cannot demand any new locks. It only releases the acquired locks.

TWO-PHASE LOCKING



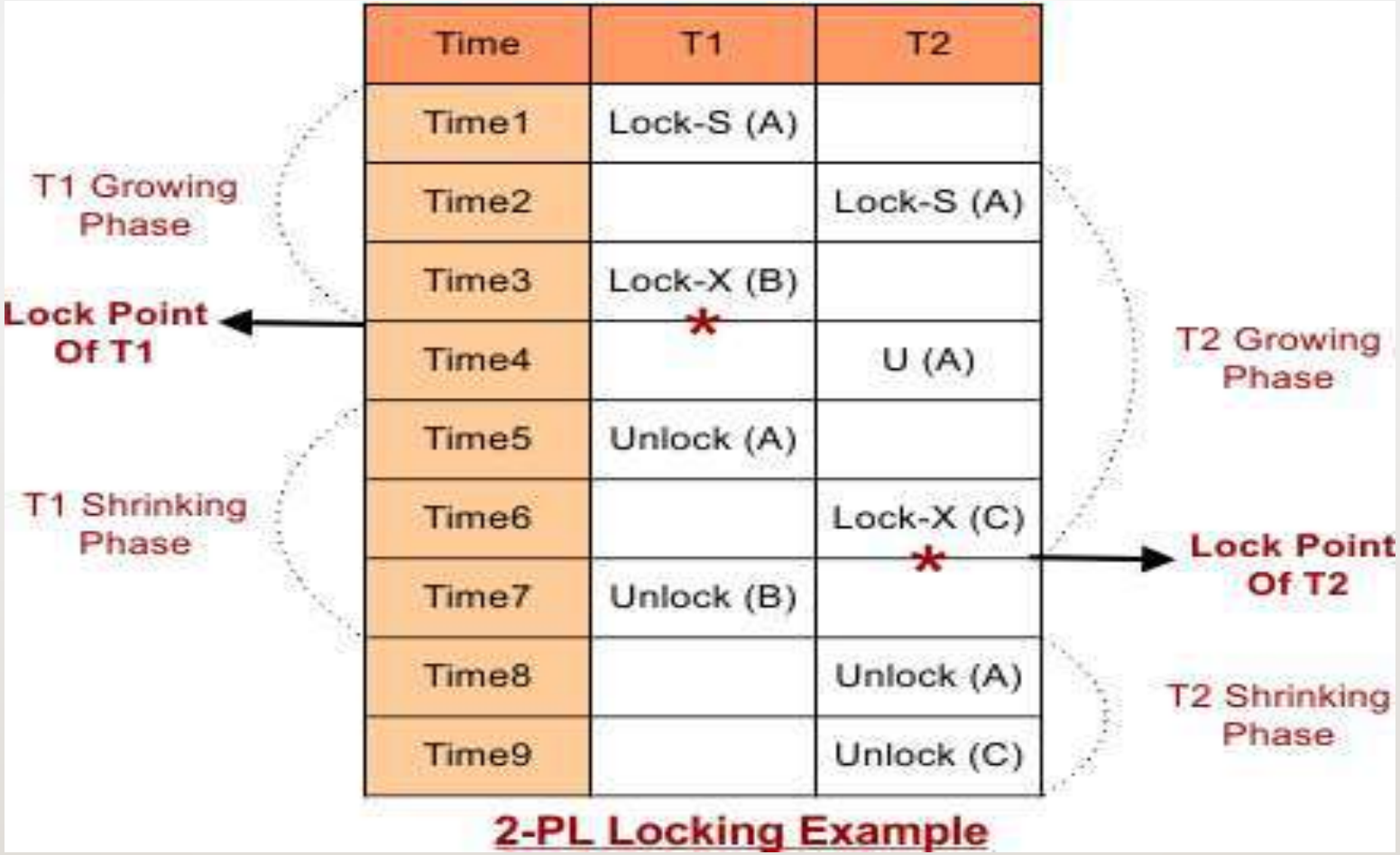
TWO PHASE LOCKING

- There are two phases of 2PL:
- **Growing phase:** In the growing phase, a new lock on the data item may be acquired by the transaction, but none can be released.
- **Shrinking phase:** In the shrinking phase, existing lock held by the transaction may be released, but no new locks can be acquired.

Crucial in 2PL is Lock Point :

Lock Point: The Point at which the growing phase ends (i.e., when transaction takes the final lock) is called Lock Point.

TWO-PHASE LOCKING-EXAMPLE



IMPLEMENTATION OF TWO-PHASE LOCKING

Transaction T1

- **Growing Phase:** From Time 1 to 3.
- **Shrinking Phase:** From Time 5 to 7
- **Lock Point:** At time 3

Transaction T2

- **Growing Phase:** From Time 2 to 6
- **Shrinking Phase:** From Time 8 to 9
- **Lock Point:** At Time 6

ADVANTAGES & DISADVANTAGES OF 2PL

Advantages of 2-PL

- 2PL always ensure the Serilizability. It means the schedule in 2PL must be serial.
- No loop existing in transaction execution sequence graph.

Disadvantages of 2-PL

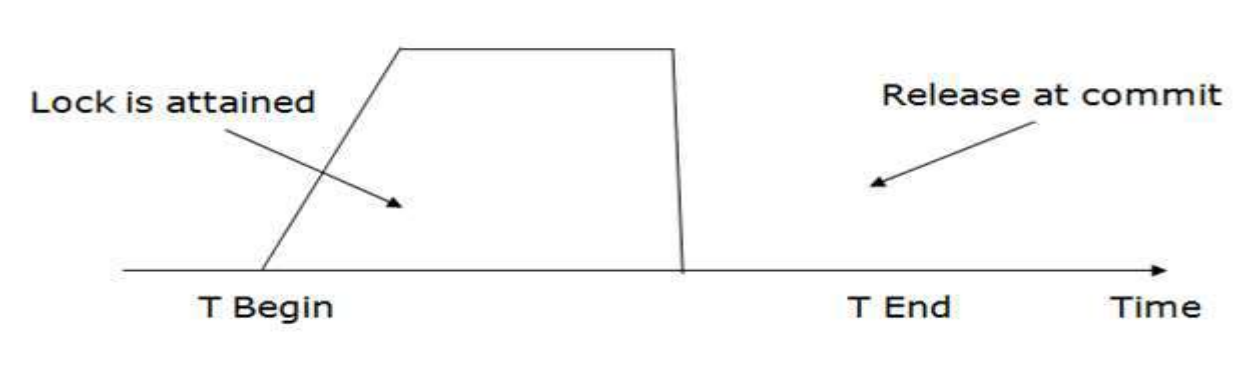
- Produces Schedules may contain Deadlock Problem
- Produced Schedules may be irrecoverable
- Produced schedules may contain Starvation Problem
- May suffer from Cascading Rollback Problem

TYPES OF TWO-PHASE LOCK PROTOCOLS

4. Strict Two-phase locking (Strict-2PL)

The first phase of Strict-2PL is similar to 2PL. In the first phase, after acquiring all the locks, the transaction continues to execute normally.

- The only difference between 2PL and strict 2PL is that Strict-2PL does not release a lock after using it.
- Strict-2PL waits until the whole transaction to commit, and then it releases all the locks at a time.
- Strict-2PL protocol does not have shrinking phase of lock release.



DEADLOCK PROBLEM IN 2-PHASE LOCKING

- A deadlock occurs during two-phase locking when a transaction is waiting for an item that is locked exclusively by another.
- Suppose Transaction 1 has locked item X and then attempts to lock item Y, but item Y has already been locked by Transaction 2. Under two-phase locking rules, Transaction 1 cannot proceed until it acquires the lock for item Y, so it is forced to wait.
- Meanwhile Transaction 2 has a lock on item Y, but needs item X to proceed with its operation. Transaction 2 cannot proceed either until it acquires the lock for item X. Neither transaction can proceed until the other releases the lock, but neither can release the lock until its operation is completed.

DEADLOCK PROBLEM IN 2-PHASE LOCKING

Time	T1	T2
t1	begin-trans	
t2	read-lock(A) → shared lock	
t3	read(A)	
t4	write-lock(B) →exclusive lock	
t5	read(B)	begin-trans
t6		write-lock(B) →not-granted
t7		wait
t8		wait
t9		wait
t10		

In Transaction 1 there are two locks being implemented. During time t2, read-lock is being used on value A because value A is needed to read only and write-lock is being used on value B because value B is needed for modifying or writing data. Meanwhile Transaction 2 begins and write-lock is being requested and this is when dead-lock happens which the write-lock in Transaction 2 Is not granted because value A is being write-lock in Transaction T1. As we know in growing phase only locks are allowed until all the operations is done in that following Transaction only, release lock will be performed which is known as during shrinking phase.

SUMMARY OF THE SESSION

- A locking protocol is a set of rules that state when a transaction may lock and unlock each of the data items in the database.
- The two-phase locking protocol allows a transaction to lock a new data item only if that transaction has not yet unlocked any data item.
- The protocol ensures serializability, but not deadlock freedom. In the absence of information concerning the manner in which data items are accessed, the two-phase locking protocol is both necessary and sufficient for ensuring serializability.
- The strict two-phase locking protocol permits release of exclusive locks onlyat the end of transaction, in order to ensure recoverability and cascadelessness of the resulting schedules.
- The rigorous two-phase locking protocol releases all locks only at the end of the transaction.

SELF-ASSESSMENT QUESTIONS

A transaction is made to wait until all _____ locks held on the item are released

(a) Compatible
(b) Equivalent
(c) Compatible
(d) Executable

2. The protocol that indicates when a transaction may lock and unlock each of the data items is called as

a) Locking protocol
b) Unlocking protocol
c) Granting protocol
d) Conflict protocol

TERMINAL QUESTIONS

- 1. Explain different types of Locking Protocols

2. Describe the advantages and disadvantages of Two phase locking protocol

3. What do you mean by concurrency control techniques?

4. Explain the working of locking technique in concurrency control. What benefits does Rigorous two-phase locking provide? How does it compare with other forms of two-phase locking?