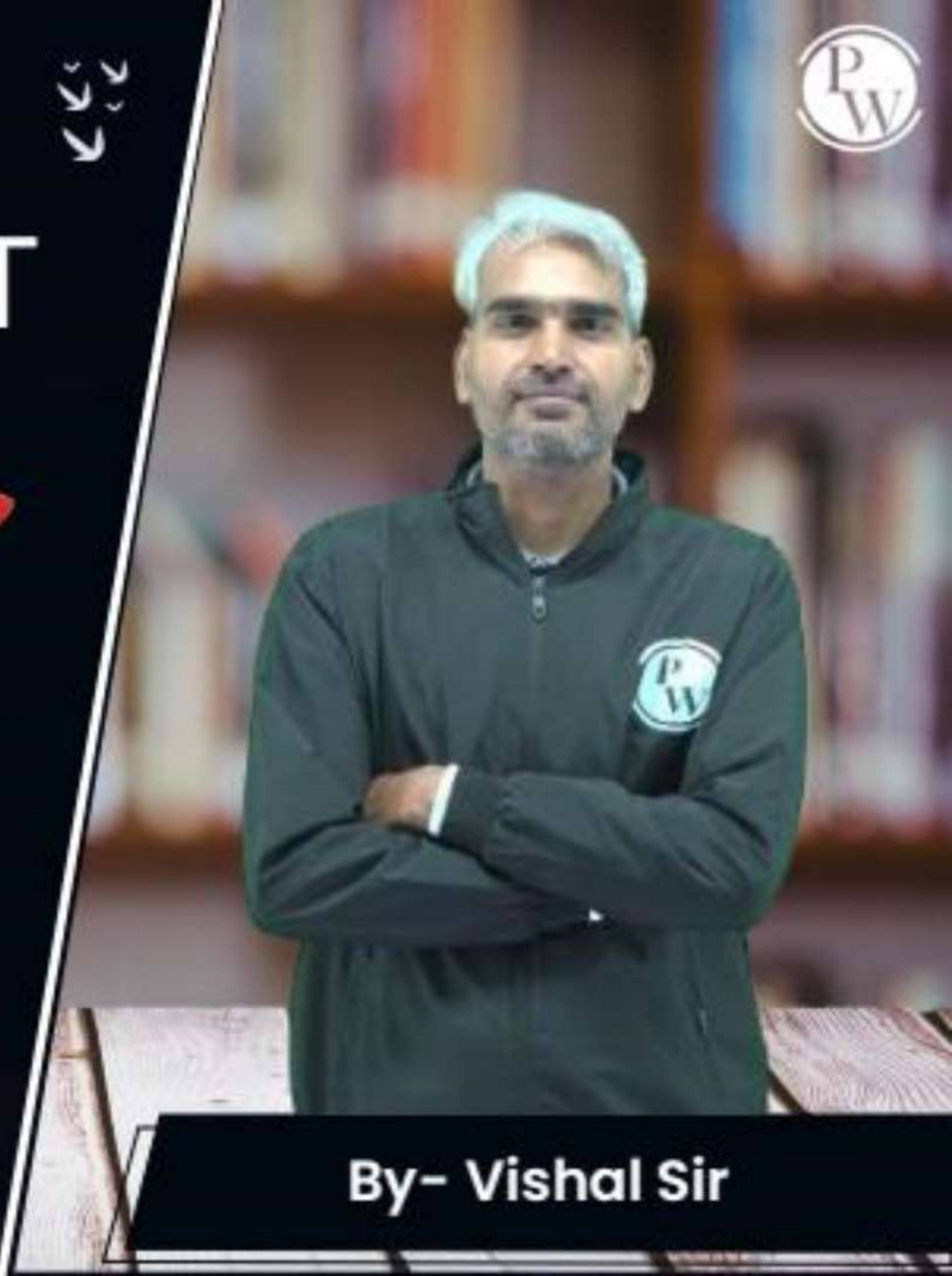
Computer Science & IT

Database Management System

Transaction & Concurrency control

Lecture No. 10



By- Vishal Sir

Recap of Previous Lecture

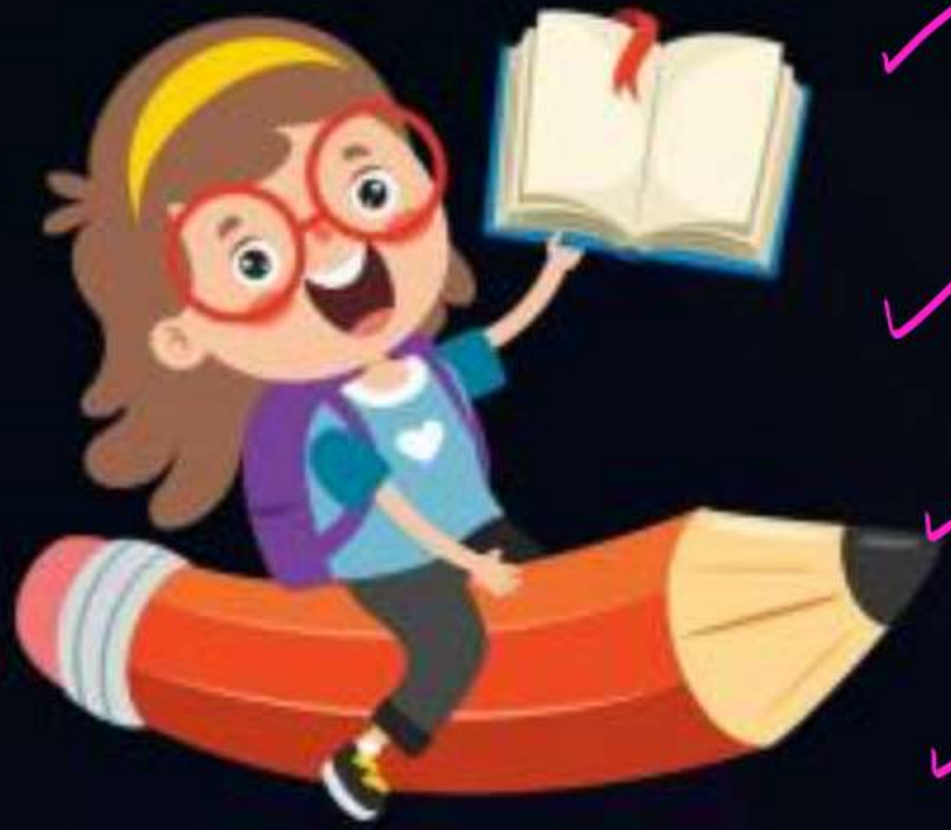
✓ **Topic** Simple use of shared and exclusive locks

✓ **Topic** Basic Two phase lock (2PL)

✓ **Topic** Basic 2PL and lock upgrading/downgrading

Slide

Topics to be Covered



- ✓ Topic Basic 2PL and lock upgrading/downgrading
- ✓ Topic Problems possible with Basic 2PL
- ✓ Topic Strict 2PL
- ✓ Topic Conservative 2PL
- ✓ Topic Rigorous 2PL

Slide

HW. eg.

Schedule S

T1	T2	T3
	X(A)	
	R(A)	
X(B)		
R(B)		
	W(A)	
	~~X(B)~~	
	U(A)	
		R(A)
W(B)		
		W(A)
	R(B)	
	W(B)	

denied ∵ Not allowed by basic 2PL without lock upgrading

Q.1. Check whether the Schedule is allowed to execute using Basic 2PL without lock upgrading or not

HW. eg.

Schedule S

T1	T2	T3
	S(A)	
	R(A)	
S(B)		
R(B)		
	X(A)	
	W(A)	
	S(B)	
	U(A)	
		S(A) ✓
		R(A)
~~X(B)~~		
~~W(B)~~		
		X(A)
		W(A)
		U(A)
	R(B)	
	W(B)	

denied → X(B)

Because 'B' is already locked by T2

Q.2 Check whether the Schedule is allowed to execute using Basic 2PL with lock upgrading or not

Schedule is not allowed to execute using basic 2PL even when lock upgrading is allowed.

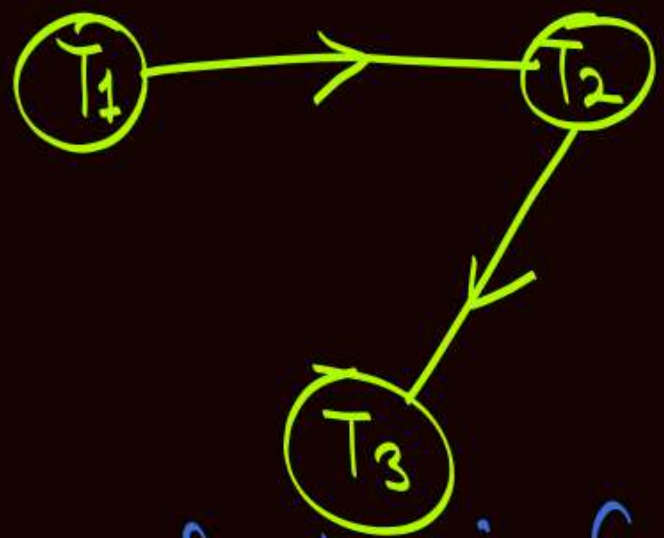
HW.
eg.

Schedule S

T1	T2	T3
	R(A)	
R(B)		
	W(A)	
		R(A)
W(B)		
		W(A)

	R(B)
	W(B)	

Precedence graph



Acyclic ∴ C.S.S.

Schedule is a Conflict serializable schedule, but not allowed to execute using basic 2PL even when lock upgrading is allowed.

Note:- ① If lock upgrading is allowed then basic 2PL May allow some extra Conflict serializable schedules, but still there will be many Conflict serializable schedules which will not be allowed to execute using basic 2PL even when lock upgrading is possible.

② If schedule is not a C.S.S., then schedule is never allowed to execute using basic 2PL. No matter whether lock upgrading is allowed or not.

By default 2PL is without lock upgrading.

Note:- Definition of 2PL does not allow all serializable schedules, it Only allows some (not all) of the Conflict serializable schedules.

Topic : Problems possible with Basic 2PL

A schedule which is allowed to execute using $\boxed{\text{basic 2PL}}$ protocol may suffer from,

✓ (1) Irreccoverability & Cascading Rollback

✓ (2) Deadlock

✓ (3) Starvation

Topic : Irrecoverability with Basic 2PL + Cascading Rollback

T_1	T_2
X(A)	
R(A)	
W(A)	
X(B)	
S(C)	
U(A)	
	S(A)
	R(A)
R(B)	
W(B)	
U(B)	
	S(B)
	R(B)
	U(A)
	U(B)
	Commit
R(C)	
U(C)	
Commit	

Transaction T_2 reads the value updated by an uncommitted transaction T_1

∴ Uncommitted Read opn exist

T_2 depends on T_1 and T_2 Committed before commit of T_1

Schedule is a conflict serializable schedule, and it is allowed to execute using basic 2PL

Solution is strict 2PL

∴ Cascading rollback problem is possible in the schedules allowed by basic 2PL

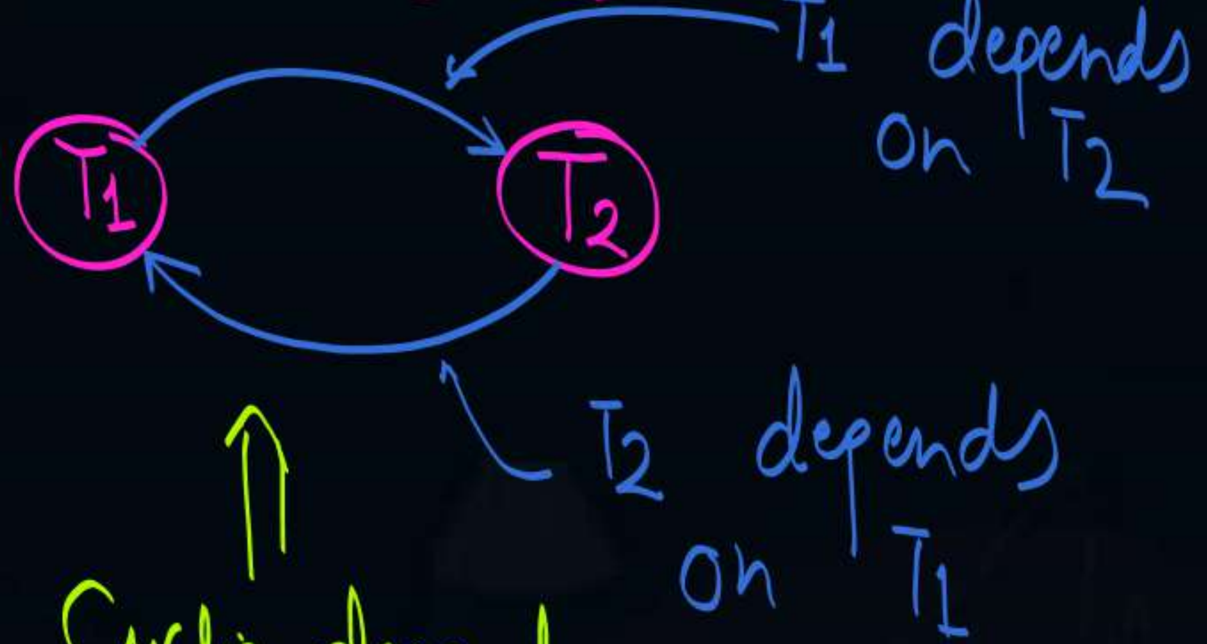
∴ Irrecoverability is also possible in the schedules allowed by basic 2PL

Topic : Deadlock with Basic 2PL

T_1	T_2
✓ S(A)	
R(A)	
	X(B)
	W(B)
denied ~~S(B)~~	
R(B)	
	~~X(A)~~ denied
	W(A)

Permanent block

Dependency graph

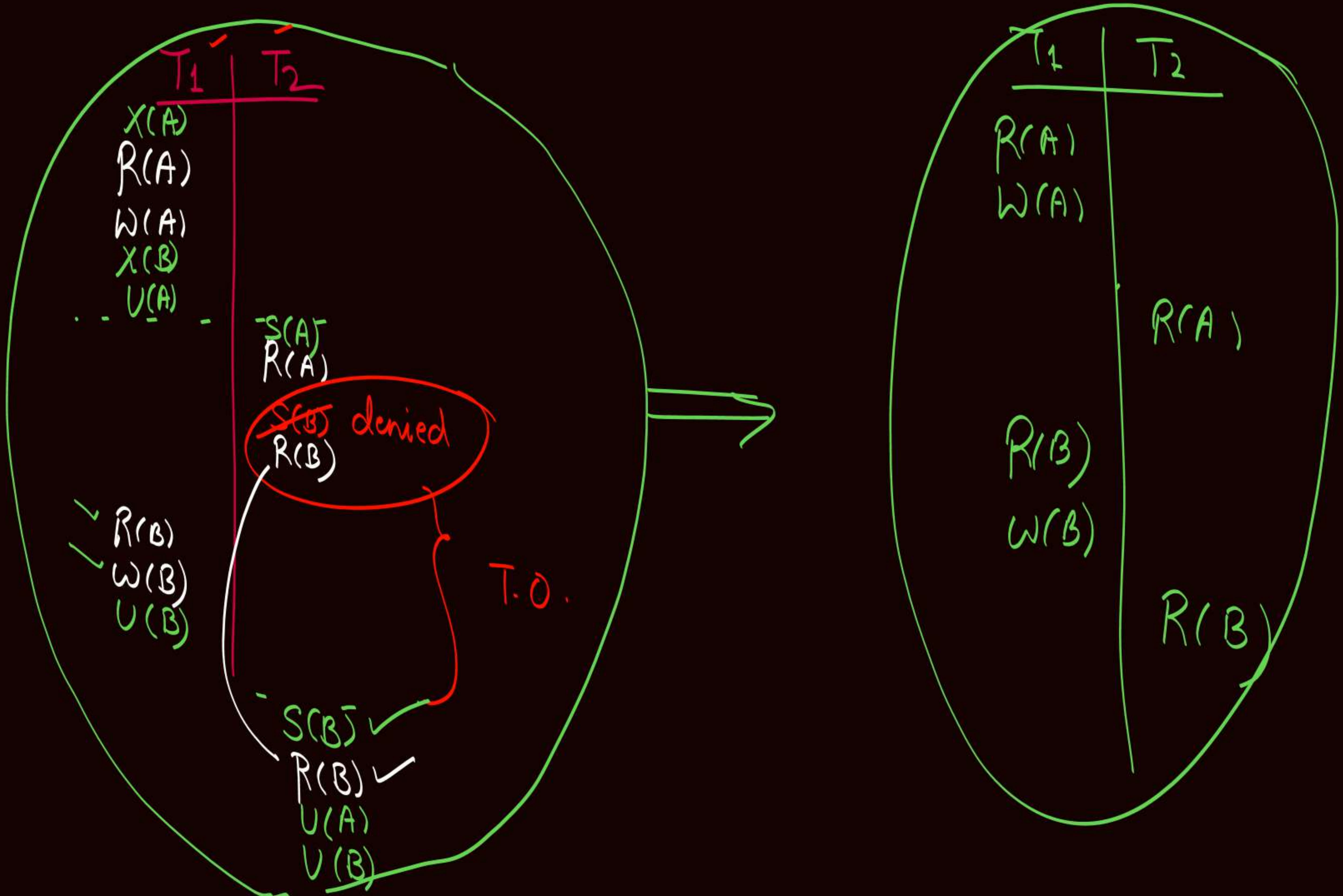


⇑

Cyclic dependency graph

∴ Deadlock is possible

Transactions T_1 & T_2 are involved in deadlock ⇒ Solution to avoid deadlock is Conservative 2PL.



T1	T2
X(A)	
R(A)	
W(A)	
X(B)	
U(A)	
	S(A)
	R(A)
	~~S(B)~~ denied
	R(B)
R(B)	
W(B)	
U(B)	
	S(B)
	R(B) ✓
	U(A)
	U(B)

T1	T2
R(A)	
W(A)	
	R(A)
R(B)	
W(B)	
	R(B)

Topic : Starvation with Basic 2PL

	T_1	T_2	T_3	T_4	T_5	T_6 - - - - -
	⋮	S(A)				
denied because of transaction T_2 →	~~X(A)~~					
T.O.			S(A)			
		U(A)				
denied because of transaction T_3 →	~~X(A)~~					
T.O.			U(A)			
				S(A)		
denied because of transaction T_4 →	~~X(A)~~					

If new transactions keep acquiring the lock on dataitem A, then T_1 will starve for lock on dataitem A } ∴ T_1 is under Starvation

No proper solution for this problem

Topic : Problems possible with Basic 2PL

A schedule which is allowed to execute using basic 2PL protocol may suffer from,

✓ (1) Irrecoverability & Cascading Rollback { Solution is Strict 2PL }

✓ (2) Deadlock { Solution is Conservative 2PL }

✓ (3) Starvation { No proper solution }

Topic : Different types of “Two phase locking protocols”

- Basic 2PL {we have already discussed}
- Strict 2PL
- Conservative 2PL
- Rigorous 2PL

Topic : Strict 2PL

(Basic 2PL + Strict recoverability Condition) = Strict 2PL

{ A transaction T is allowed to request for a lock only if it has not performed any unlock operation }

it ensures serializability

+

T_1	T_2
W(A)	
Commit	
	R(A)/W(A)

then ① free from irrecoverability
② free from Cascading Rollback
& ③ free from lost update

≡

T_1	T_2
X(A)	
Commit U(A)	
	S(A)/X(A)

Topic : Strict 2PL

"Strict 2PL" is basic 2PL with a restriction that Every Exclusive lock acquired by a transaction must be unlocked only after the Commit operation of that transaction

{Shared locks can be unlocked at any time, but in accordance with 2PL restriction}

Strict 2PL

⇓

T_1	T_2
X(A)	
Commit	
U(A)	
	S(A)/X(A)

Topic : Strict 2PL

Strict 2PL →
1. Ensures serializability
2. Ensures strict recoverability

Strict 2PL may suffer from deadlock & starvation

Strict 2PL

⇓

T_1	T_2
X(A)	
Commit	
U(A)	
	S(A)/X(A)

Topic : Conservative 2PL

$X(A)$

H.W.

How Hold & wait is dis-satisfied using Hold-or-wait in operating system

2 mins Summary

Topic Basic 2PL and lock upgrading/downgrading

Topic Problems possible with Basic 2PL

Topic Strict 2PL

Topic Conservative 2PL

Topic Rigorous 2PL

Next class

Slide

THANK - YOU

Slide