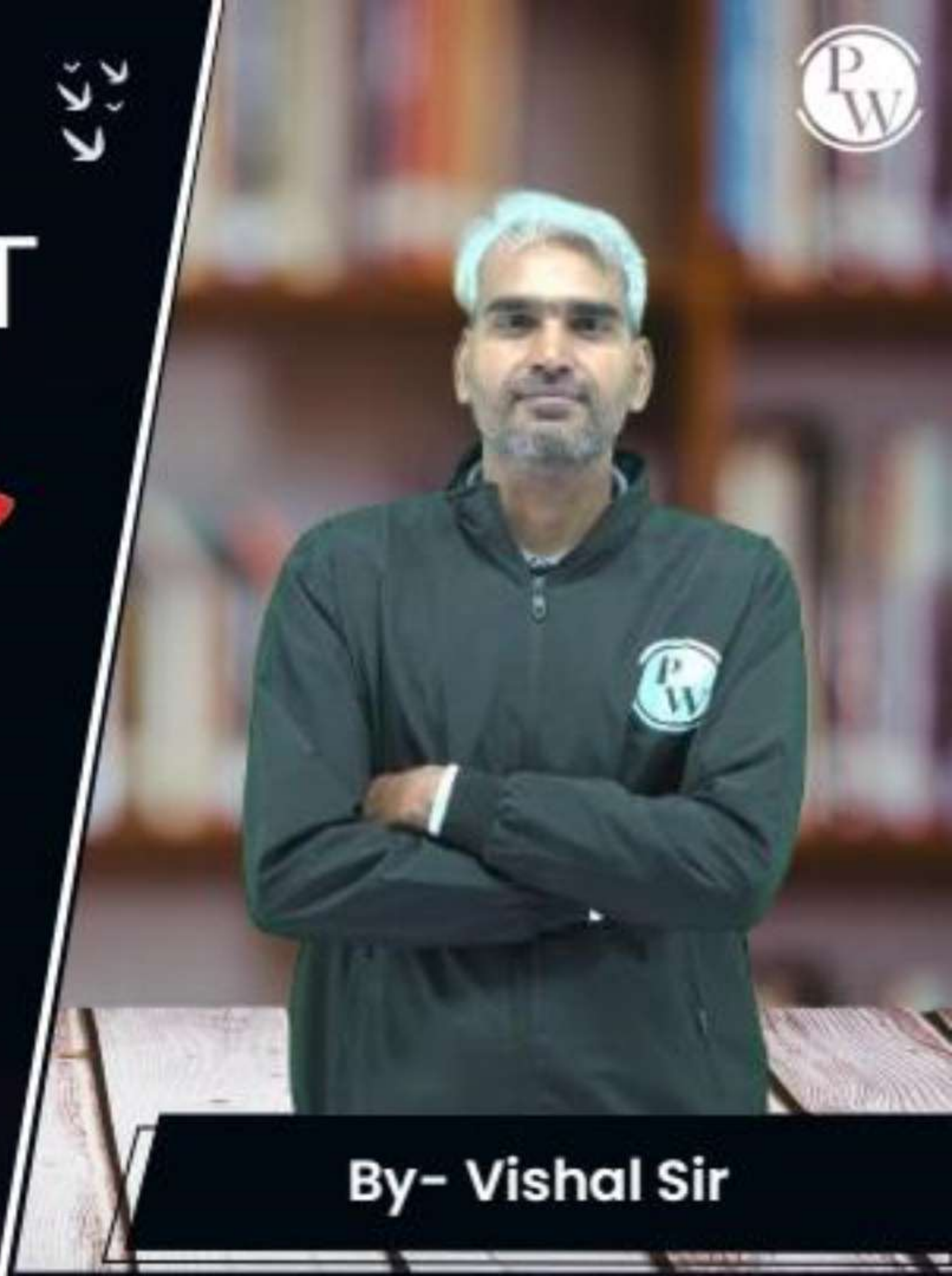
Computer Science & IT

Database Management System

Transaction & Concurrency control

Lecture No. 05



By- Vishal Sir

Recap of Previous Lecture :—

✓ Topic Serializable and non-serializable schedule

✓ Topic Problems because of concurrent execution

Topic RW problem

Topic WR problem

Topic WW problem

} Exist only if schedule is not serializable

Topic Lost update problem } Possible with both serializable as well as non-serializable schedule

Slide

Topics to be Covered :-

✓ Topic — Classification of schedule based on recoverability

Topic — Irrecoverable schedule

Topic — Recoverable schedule

Topic — Cascading rollback problem

Topic — Cascadeless recoverable schedule

Topic — Strict recoverable schedule

Slide

Topic : Classification of schedule

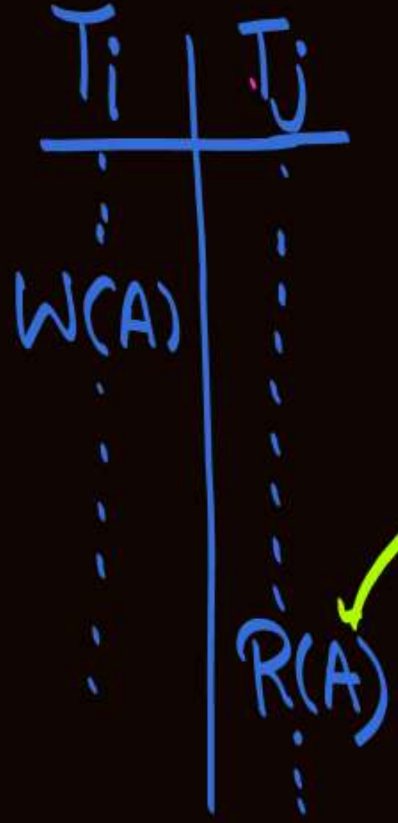
Based on Recoverability

1. Irrecoverable schedule
2. Recoverable schedule
3. Cascadeless Rollback Recoverable schedule
4. Strict Recoverable schedule

Based on serializability

1. Conflict serializable schedule
2. View serializable schedule

Note :- If transaction T_j reads the value written by an uncommitted transaction T_i, then we say that transaction T_j is dependent on transaction T_i.



it is uncommitted Read opn

↓

∴ T_j depends on T_j

↳ In this case if we rollback transaction T_i for some reason then we must rollback all transactions which are dependent on transaction T_i.

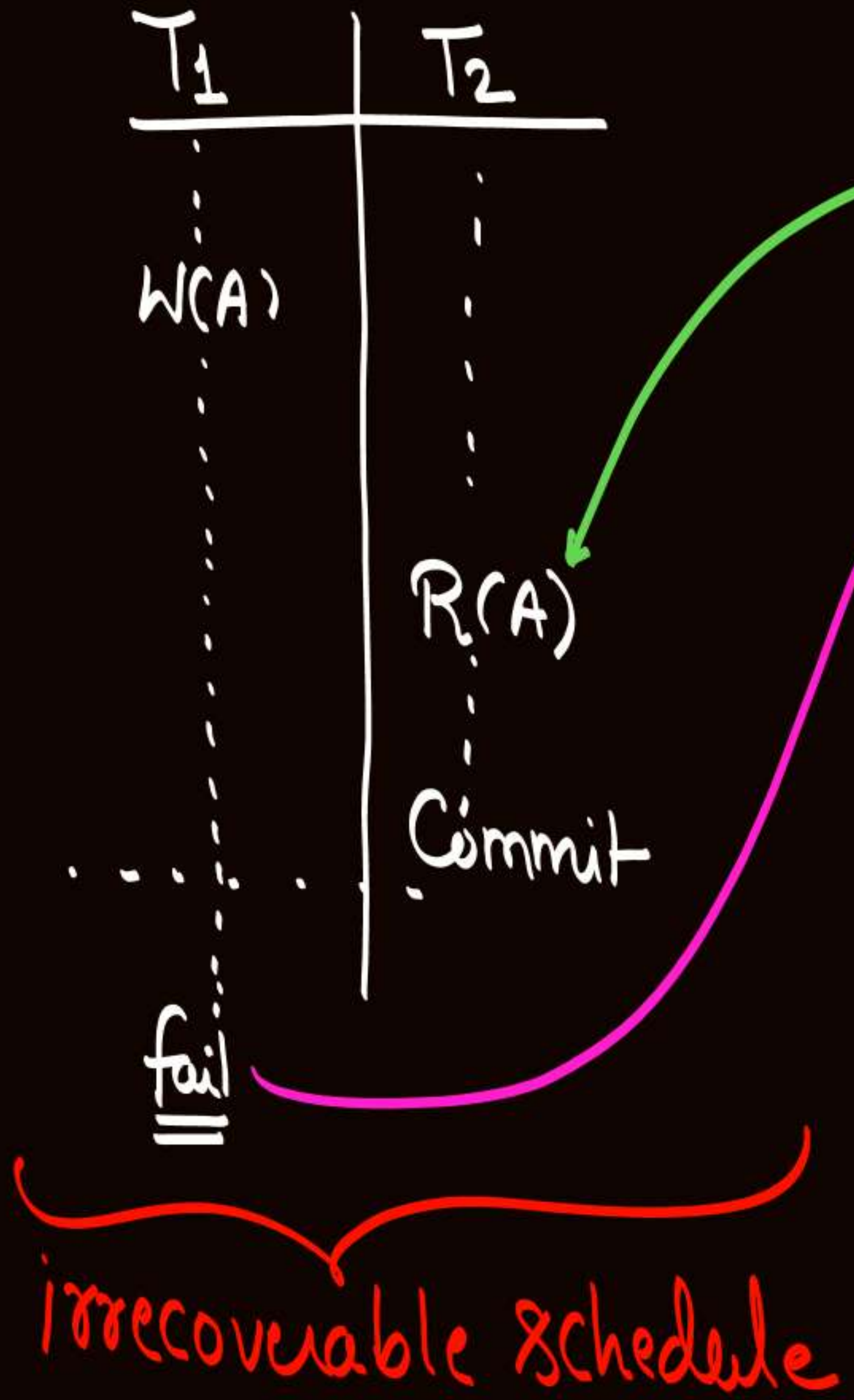
Note :- We can rollback a transaction only if that transaction has not yet Committed.

∴ Rollback of a Committed transaction is not possible

Topic : Irrecoverable schedule

<u>Note</u>:- It is not possible to rollback a Committed transaction

- If there is any situation in which we are supposed to rollback a Committed transaction, then it will not be possible, in that case we will not be able to recover from that failure, ∴ such schedule will be called irrecoverable schedule



T2 Reads the value written by uncommitted transaction T1,
∴ T2 is dependent on T1.

T1 failed after Commit of transaction T2

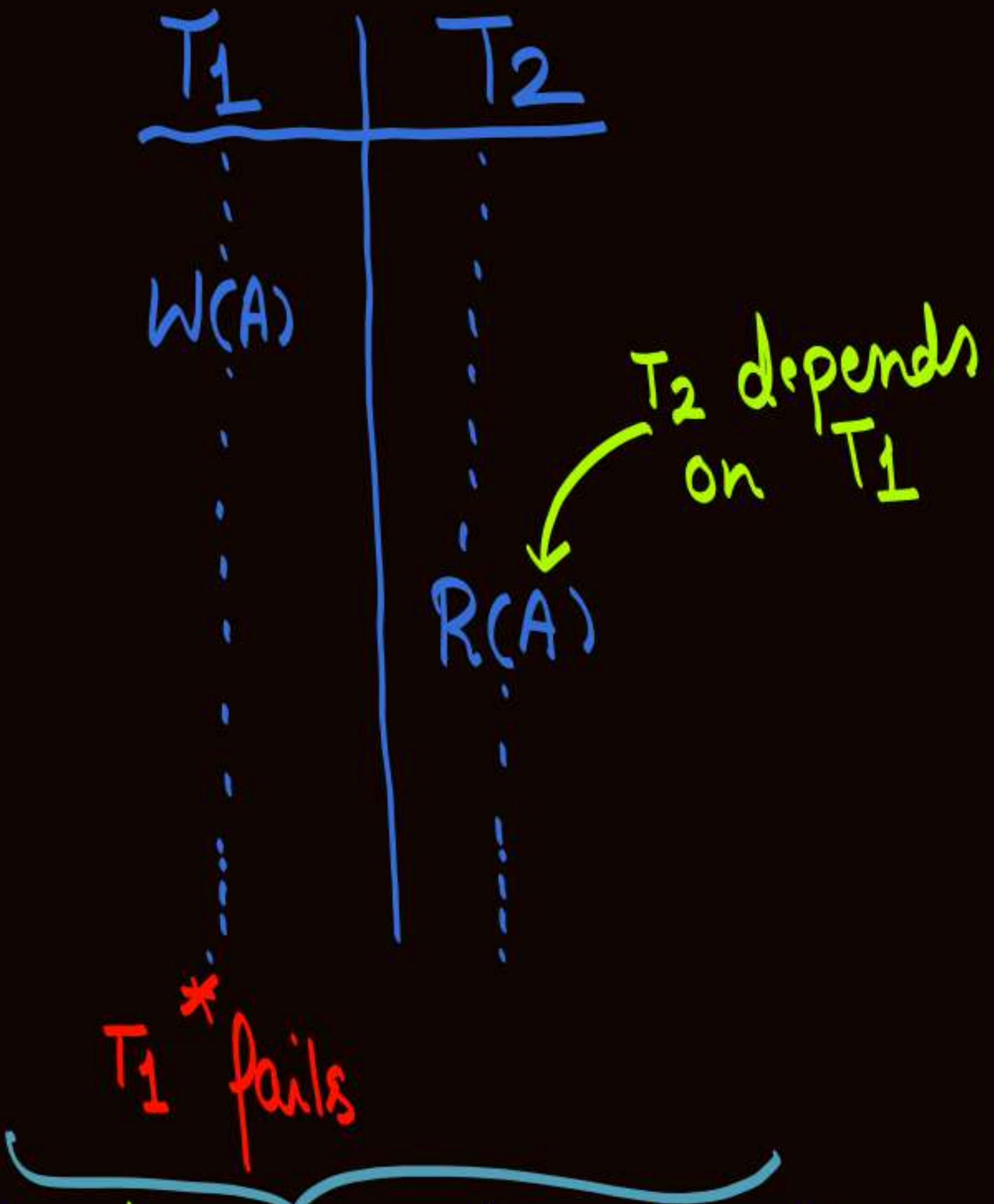
- Because of that failure we will rollback transaction T1, and we know that T2 is dependent on T1.

∴ Because of rollback of T1, we will try to rollback transaction T2

But transaction T2 has Committed ∴ We can not rollback T2, hence we are not able to recover from failure

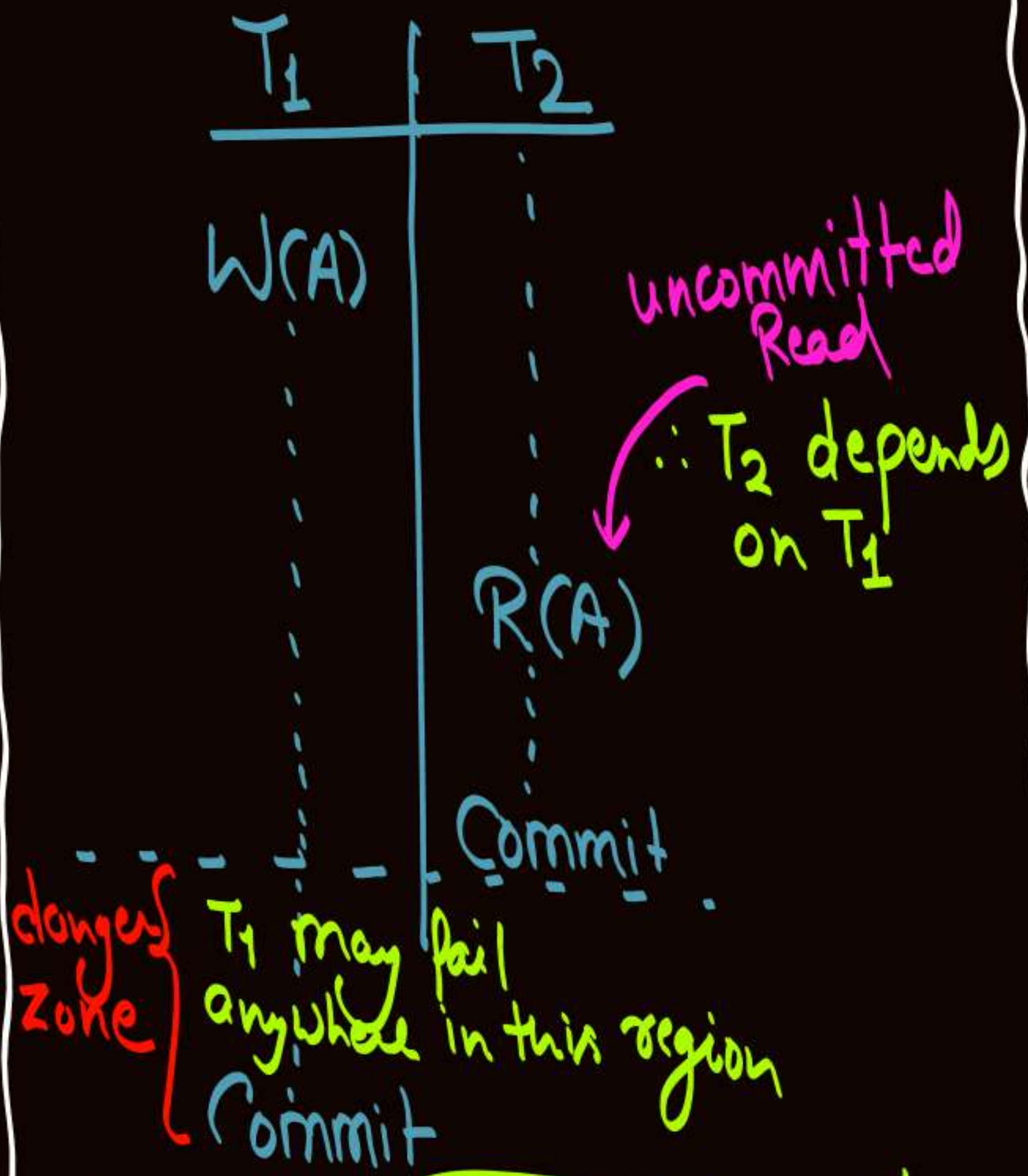
And hence schedule is irrecoverable schedule

Schedule S_1

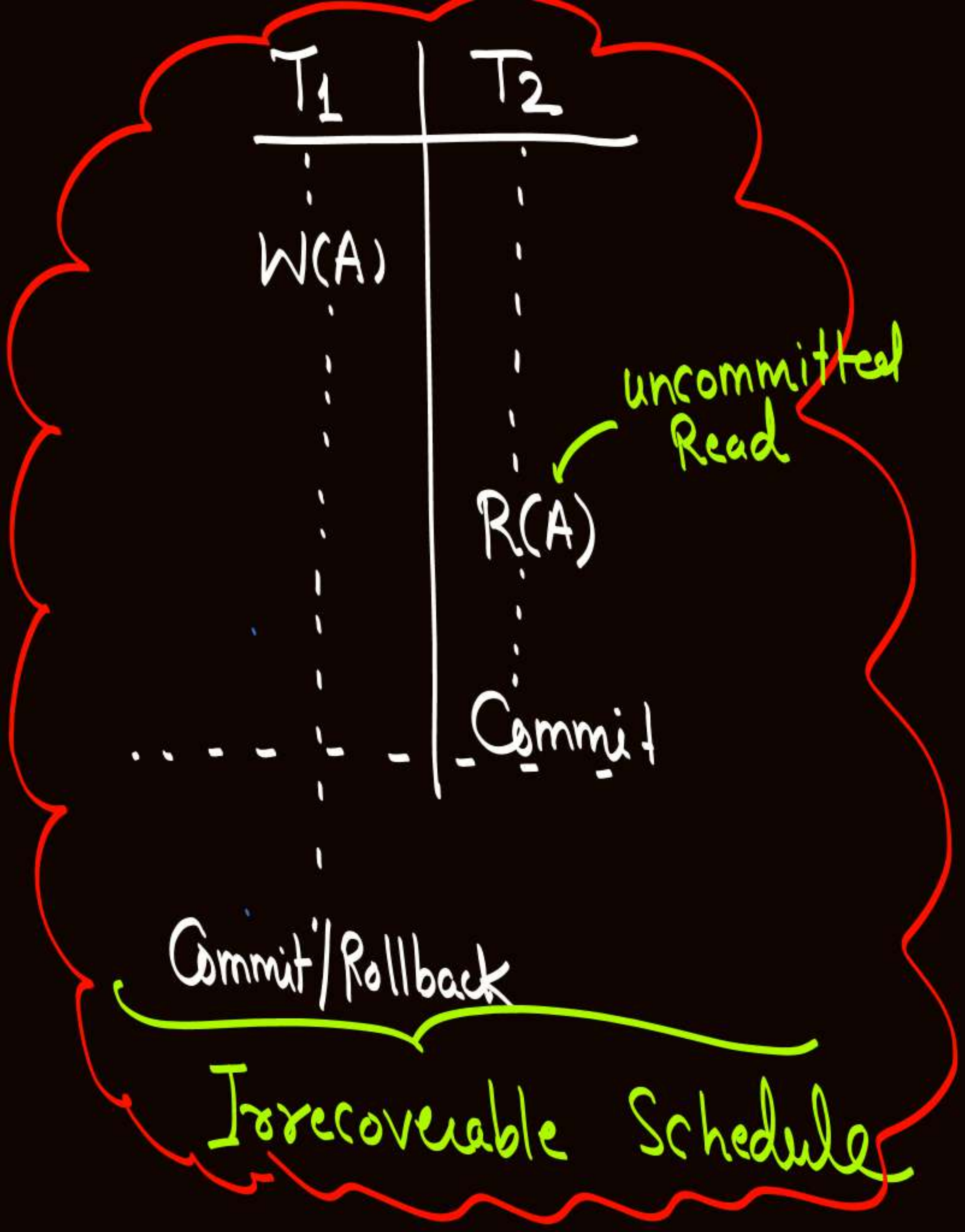


No transaction has Committed
∴ Both Can rollback
Hence, Recoverable Schedule

Schedule S_2



It is irrecoverable schedule



Irrecoverable Schedule :-

In a Schedule if transaction Tj depends on transaction Ti, and if transaction Tj {dependent transaction} Commit before Rollback/Commit of transaction Ti, then Schedule is called "Irrecoverable Schedule"

Note:-

① Uncommitted read opⁿ is a necessary Condⁿ (not sufficient) for a schedule to be "irrecoverable" schedule

② If no uncommitted read opⁿ exist in a schedule then that schedule is always a recoverable schedule

③ If uncommitted read opⁿ exist in a schedule, then that schedule may be irrecoverable <u>and</u> may be recoverable

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

Uncommitted read op^n exist, but schedule is a "recoverable schedule"

if T_1 fails in this region, then both T_1 & T_2 Can be rollbacked

∴ Recoverable Schedule

Topic : Recoverable schedule

T_1	T_2
W(A)	
Commit	
	R(A)

not an uncommitted Read opⁿ

T_2 does not depend on T_1

∴ Recoverable Schedule

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

uncommitted Read exist but

Recoverable Schedule

For a schedule to be called a recoverable schedule,

either uncommitted read opⁿ should not exist.

Or if uncommitted Read opⁿ exist, then Commit of dependent transaction must appear after Commit/Rollback of transaction on which it depends.

Topic : Cascading rollback problem

Because of rollbacking of one transaction if we need to rollback a set of other transactions as well, then it is called "Cascading Rollback Problem"

Topic : Cascading rollback problem

T_1	T_2	T_3	T_4	T_5
W(A)				
	R(A)			
		R(A)		
			W(A)	
			R(A)	
		W(A)		
				R(A)
* Fail				

No transaction has Committed.
∴ All Can rollback
Hence Recoverable.

- T_4 read the value updated by itself.
- T_5 read the value updated by T_3

- If T_1 fails at this point, then we will rollback transaction T_1.
- Because of rollback of transaction T_1, we will rollback transactions dependent on T_1 {i.e. T_2 & T_3}
- Because Rollbacking T_2 & T_3, ∴ we will have to rollback the transactions dependent on T_2 & T_3 as well {i.e, T_5}
- No transaction dependent on T_5.

In the above example if T_1 fails, then along with T_1 we will have to rollback T_2, T_3 & T_5 as well

∴ Cascading Rollback

→ Problem because of Cascading rollback is wastage of CPU time & IO time.

Topic : Cascading rollback problem

- If uncommitted read op^n exist, then dependency will exist among the transactions, and if dependency exist among the transactions then Cascading rollback Problem will also exist.

Topic : Cascadeless recoverable schedule

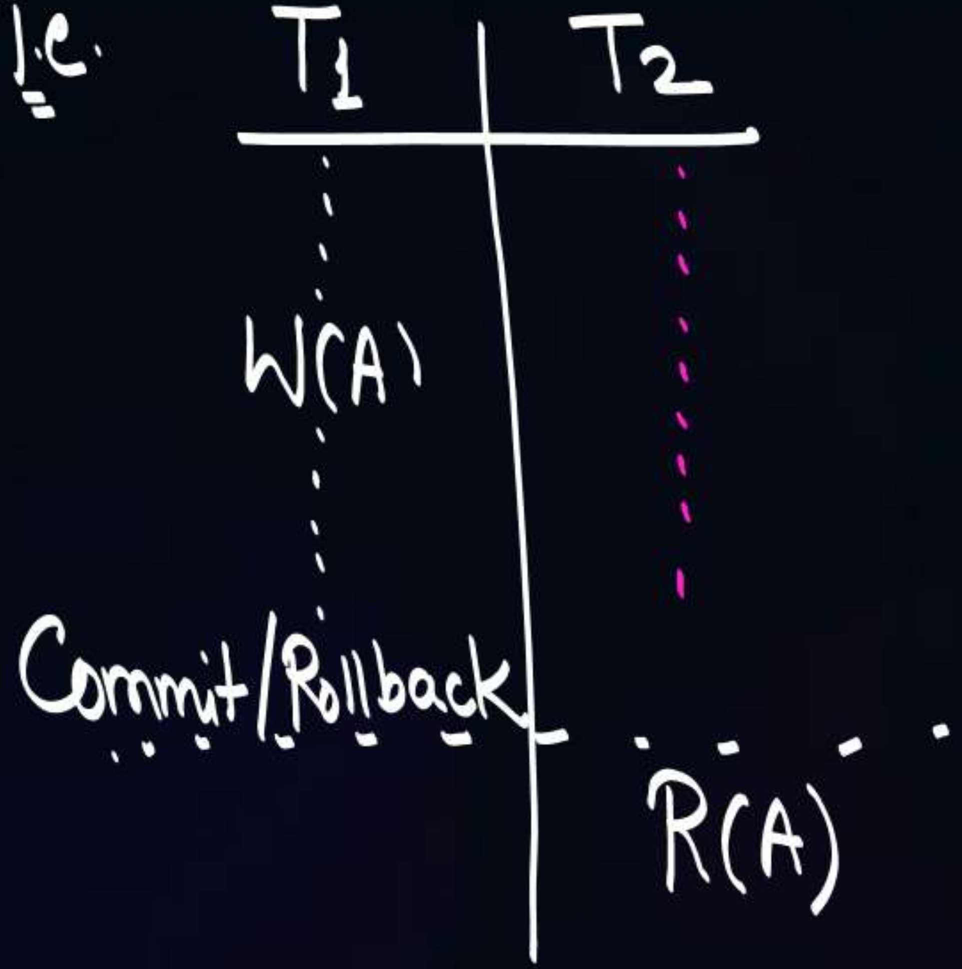
$$\text{Cascadeless Recoverable schedule} = \underbrace{\text{No Cascading Problem} + \text{Recoverable schedule}}$$

If want to ensure that Cascading rollback Problem does not exist, then uncommitted read opn must not be allowed. and If uncommitted read opn does not exist then schedule is always recoverable

Topic : Cascadeless recoverable schedule

- For Cascadeless recoverable schedule uncommitted read opn must not be allowed. { i.e. Simultaneous write - read opn not allowed }

i.e.



For Cascadeless recoverable schedule, if transaction T1 performs the write opn on some dataitem 'A', then no other transaction should be allowed to Read the data item 'A' until the Commit/Rollback of T1.

Note:-

Cascadeless recoverable schedule are

1. Free from
   - 1. Cascading Rollback Problem ✓
   - 2. WR problem ✓
2. Not free from
   - 1. RW Problem ✓
   - 2. WW Problem ✓
   - 3. Lost update problem ✓

We are not bothered about RW Problem, WR problem, or WW Problem, because we are anyway going to ensure that schedule is a serializable schedule, and RW, WR & WW Problem will never exist in serializable schedule

But we are bothered about Cascading Rollback Problem & Lost update problem, because they are possible even in a serializable schedule

In cascadeless recoverable schedule we are able to overcome cascading rollback problem, but lost update problem still possible.

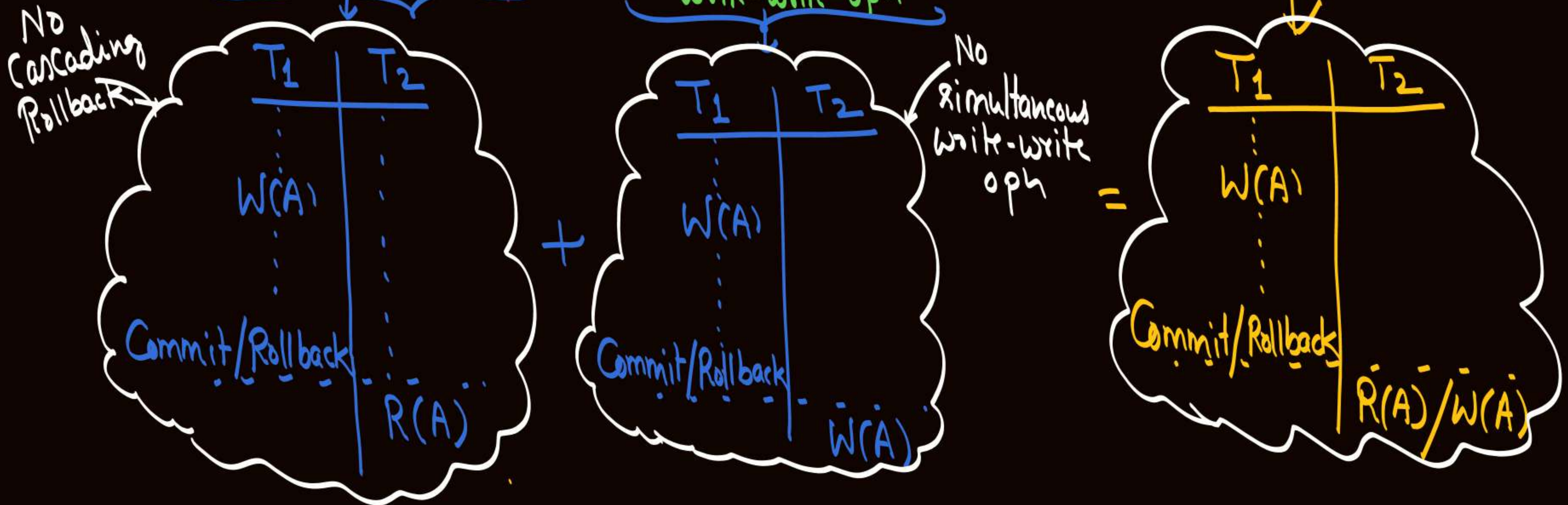
Topic : Strict recoverable schedule

Strict recoverable schedule are free from "Cascading rollback problem" as well as free from "lost update problem"

Strict Recoverable Schedule = No Cascading Rollback Problem + No Lost update Problem

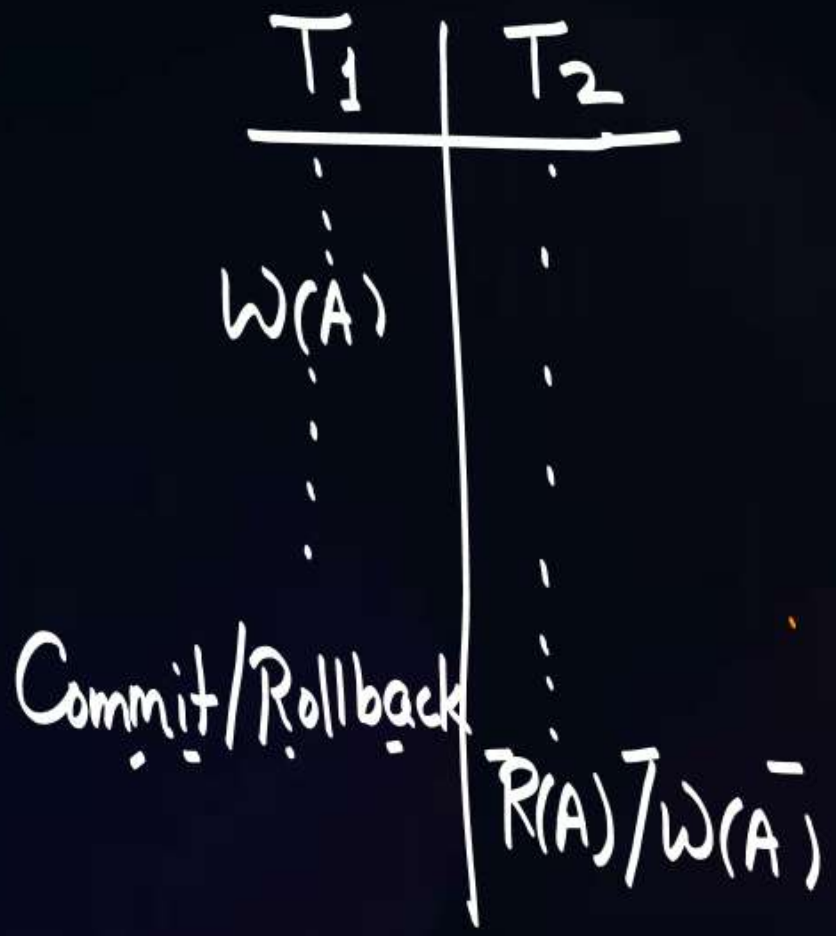
$$\underbrace{\text{No Cascading Rollback Problem}}_{\therefore\ \text{No uncommitted Read op}^n} + \underbrace{\text{No Lost update Problem}}_{\therefore\ \text{No simultaneous write-write op}^n} = \underbrace{\text{Strict Recoverable Schedule}}$$

∴ No uncommitted Read opⁿ + ∴ No simultaneous write-write opⁿ



Topic : Strict recoverable schedule

Strict recoverable schedule are free from "Cascading rollback problem" as well as free from "lost update problem"



For a schedule to be "Strict recoverable" if one transaction "T1" performs write operation on a dataitem 'A', then no other transaction is allowed to read or write the dataitem 'A' until the commit or Rollback of T1

Note: Strict Recoverable schedule are

1. free from
   - → Cascading Rollback problem
   - → Lost update problem
   - → WR problem
   - → WW Problem
2. Not free from
   - → RW Problem { it will also not create any Problem in serializable Schedule }

Criteria for Consistency

1. Schedule must be strict recoverable
2. Schedule must be serializable.

2 mins Summary

- ✓ Topic — Classification of schedule based on recoverability
- ✓ Topic — Irrecoverable schedule
- ✓ Topic — Recoverable schedule
- ✓ Topic — Cascading rollback problem
- ✓ Topic — Cascadeless recoverable schedule
- ✓ Topic — Strict recoverable schedule

Slide

THANK - YOU

Slide