CHAPTER 20

Introduction to Transaction Processing Concepts and Theory

CHAPTER 21

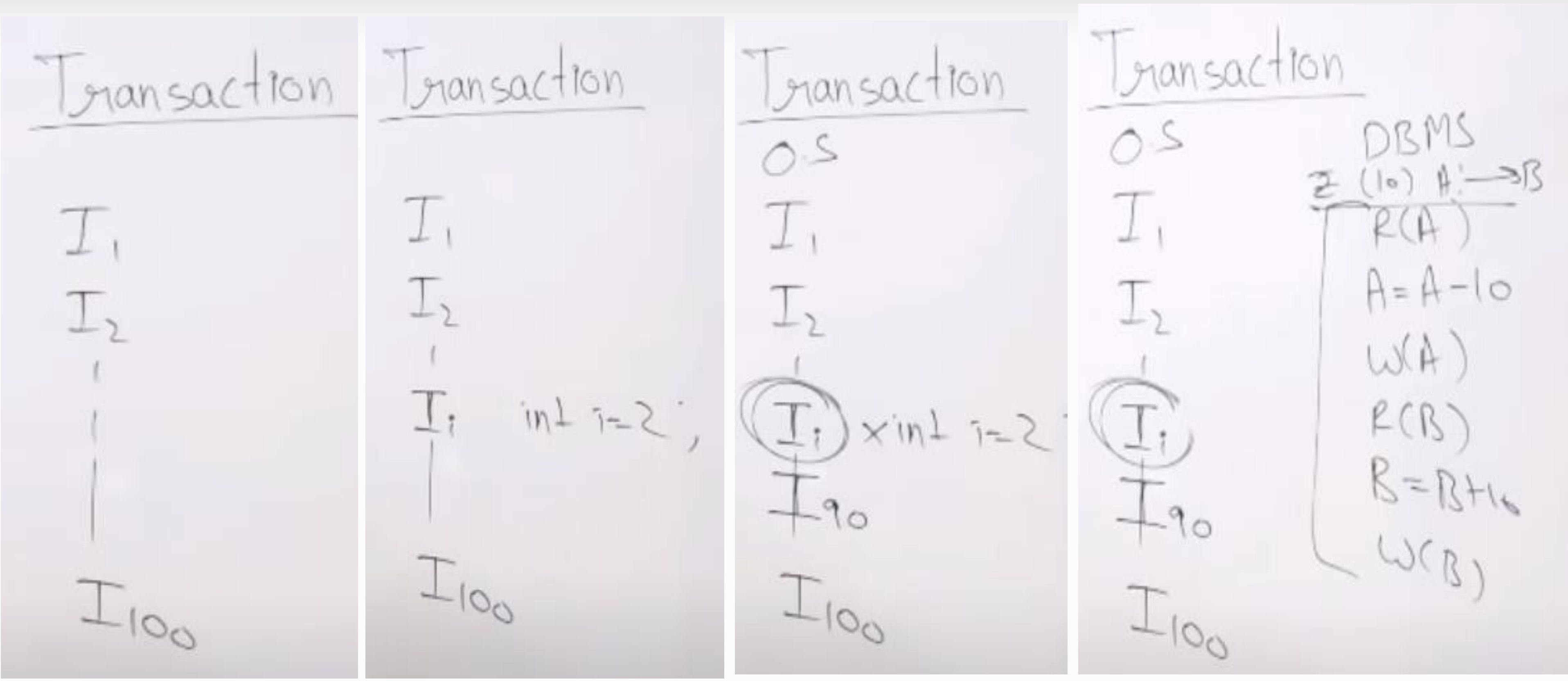
Concurrency Control Techniques

Objectives

1. What is transaction.
2. What are ACID Properties
3. Transaction states
4. Concurrency and Problems
5. Conflict and View Serializabiltiy
6. Cascade and Cascadeless Schedule

https://www.youtube.com/watch?v=HAAhn--tZV8&list=PL8UQESWu2gUlRlWjeIDsBp-QKN7OcmwLC

Transaction.



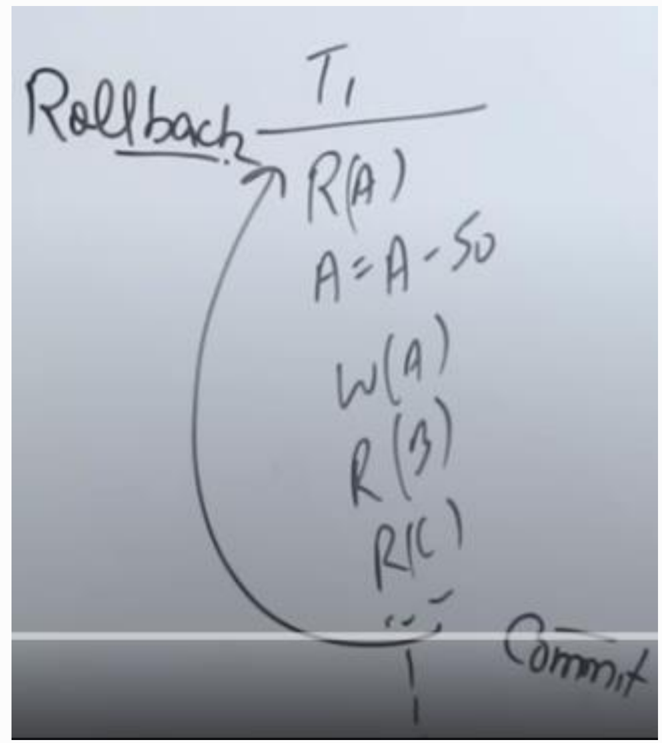
https://www.youtube.com/watch?v=HAAhn--tZV8&list=PL8UQESWu2gUlRlWjeIDsBp-QKN7OcmwLC

ACID Properties

Atomicity Consistency Isolation Durability

Atomicity

Either all or None.



https://www.youtube.com/watch?v=-GS0OxFJsYQ&list=PLxCzCOWd7aiFAN6I8CuViBuCdJgiOkT2Y&index=73

ACID Properties

Atomicity — Consistency — Isolation — Durability

Consistency

Before trans. Start
and
after the tran Completed
Sum of money should be same

1000
A → B
A = 2000
B = 3000

T_1	
R(A)	2000
A = A - 1000	
W(A)	1000
R(B)	3000
B = B + 1000	
W(B)	4000
Commit	

1000
A → B

5000 { 1000, 4000 }

A = ~~2000~~
B = ~~3000~~

5000

ACID Properties

↓

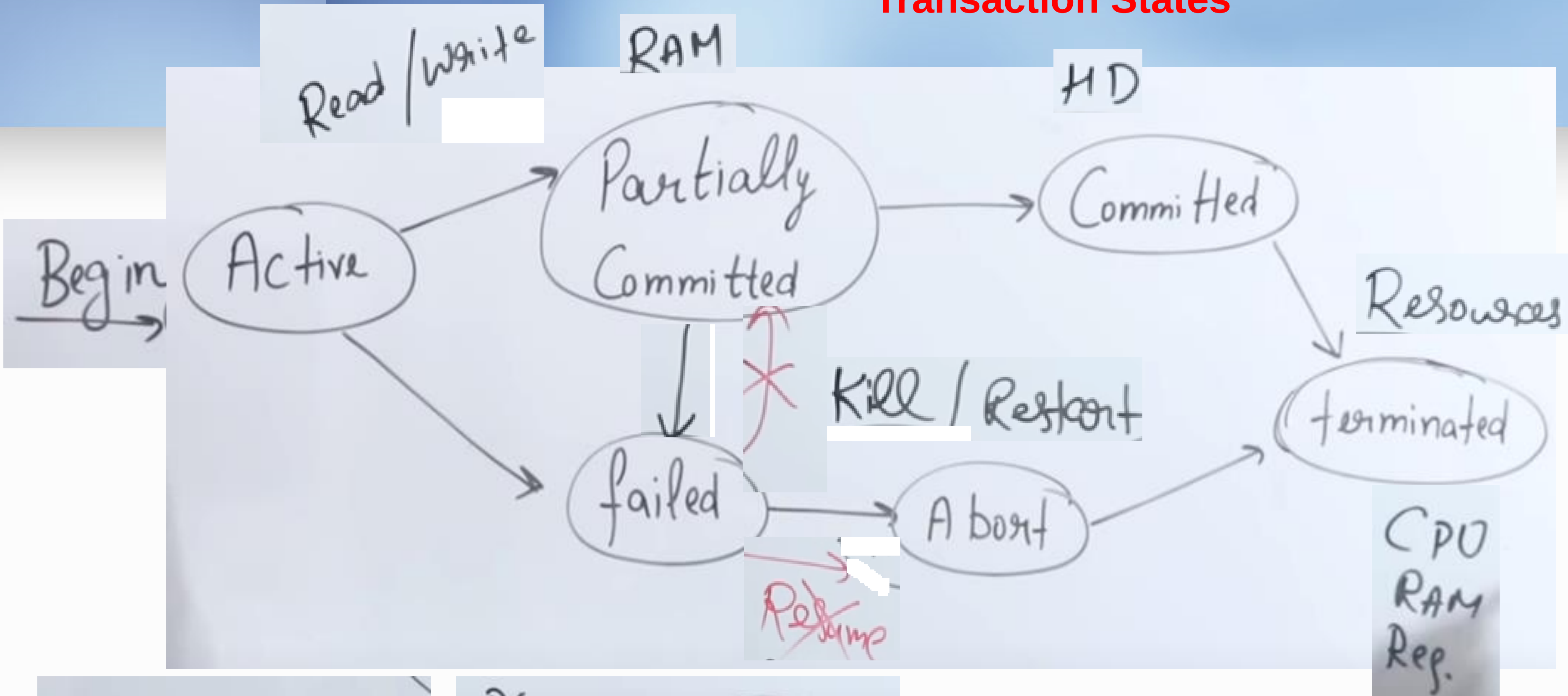
Atomicity   Consistency   Isolation   Durability

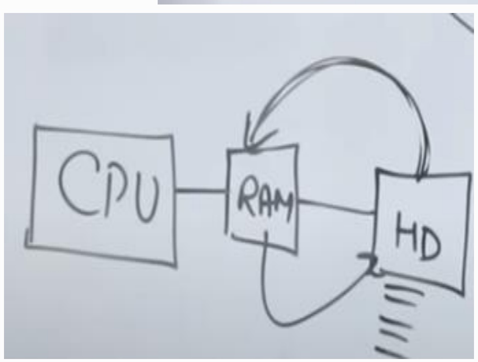
Isolation

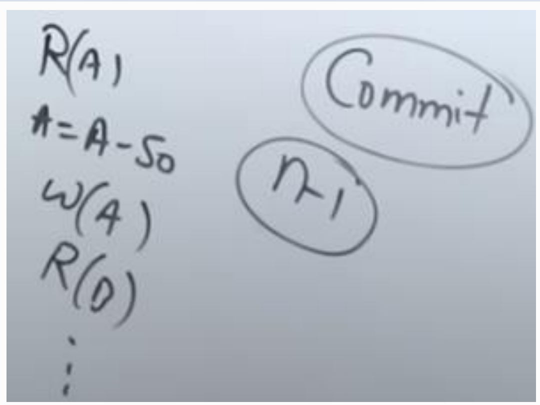
T_1	T_2	T_3
R(A)		
.	R(B)	R(A)

https://www.youtube.com/watch?v=-GS0OxFJsYQ&list=PLxCzCOWd7aiFAN6I8CuViBuCdJgiOkT2Y&index=73

Transaction States







Advantages Of Concurrency

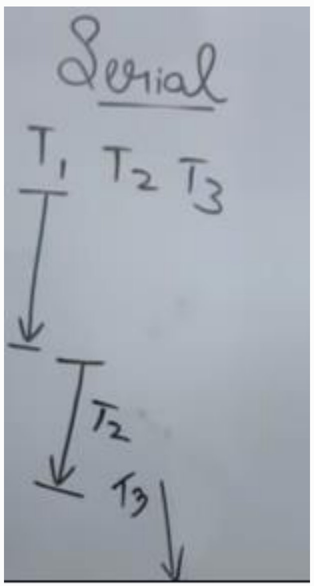
1	**Waiting Time**	↓
2	Response Time	↓
3	Resource utilization	↑
4	Efficiency	↑

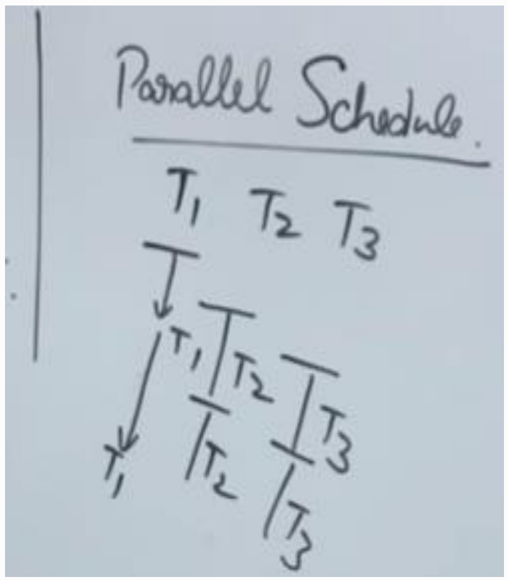
https://www.youtube.com/watch?v=JrP7B106D9M

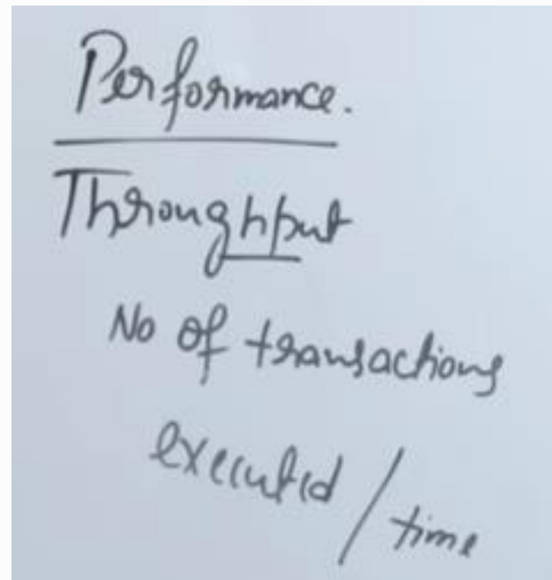
Schedule

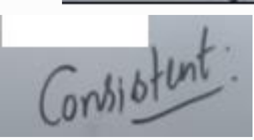
It is chronological execution sequence of multiple transactions

TI **T2** **T3** **Tn….**





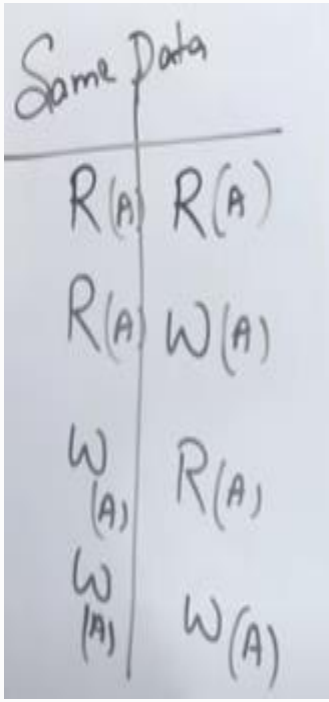


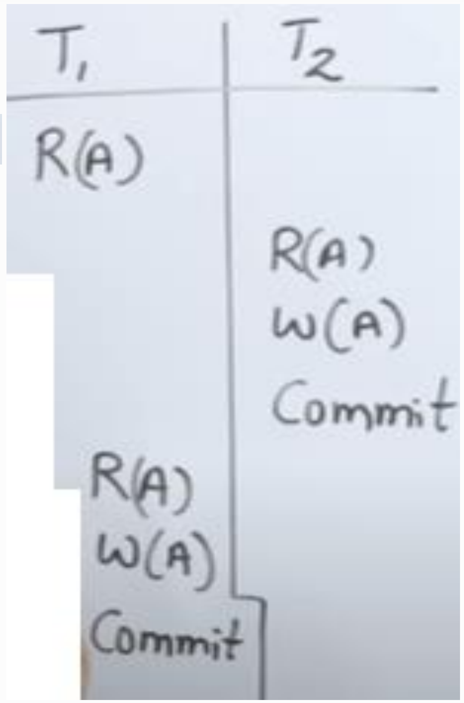


Wait

https://www.youtube.com/watch?v=1cbmhsSJRWc&list=PLxCzCOWd7aiFAN6I8CuViBuCdJgiOkT2Y&index=75

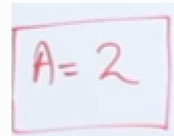
Read- Write Conflict or Unrepeatable Read





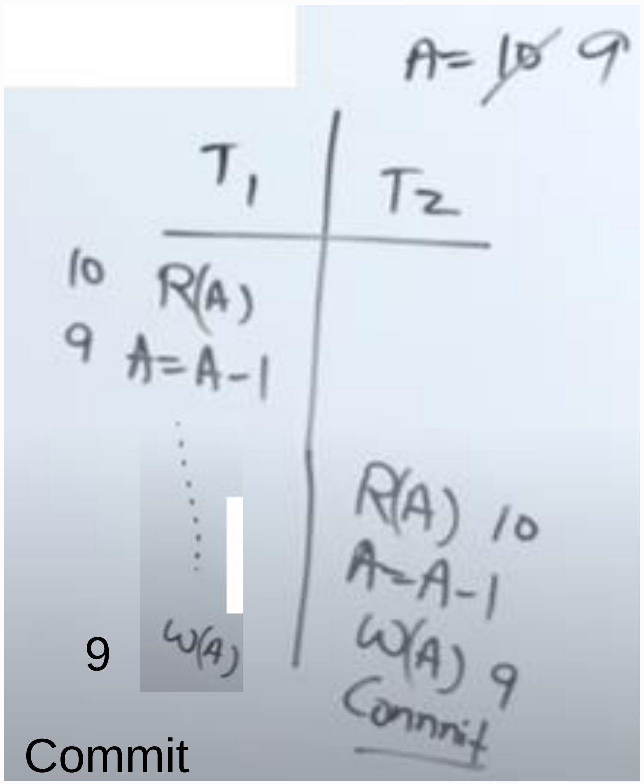
0

A=A-2



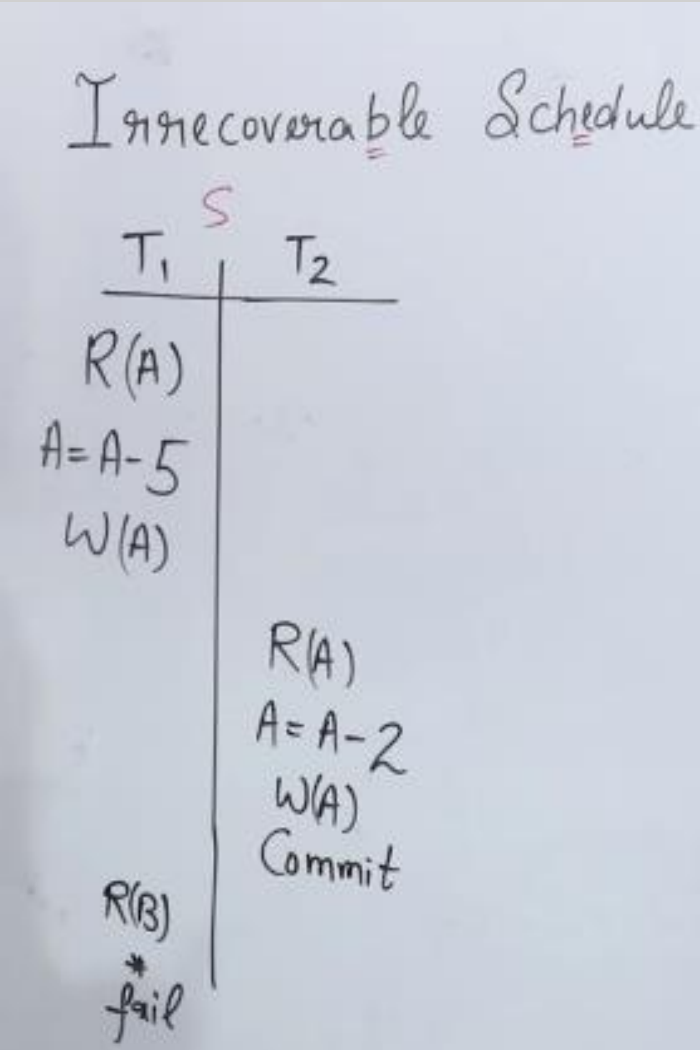
A=0

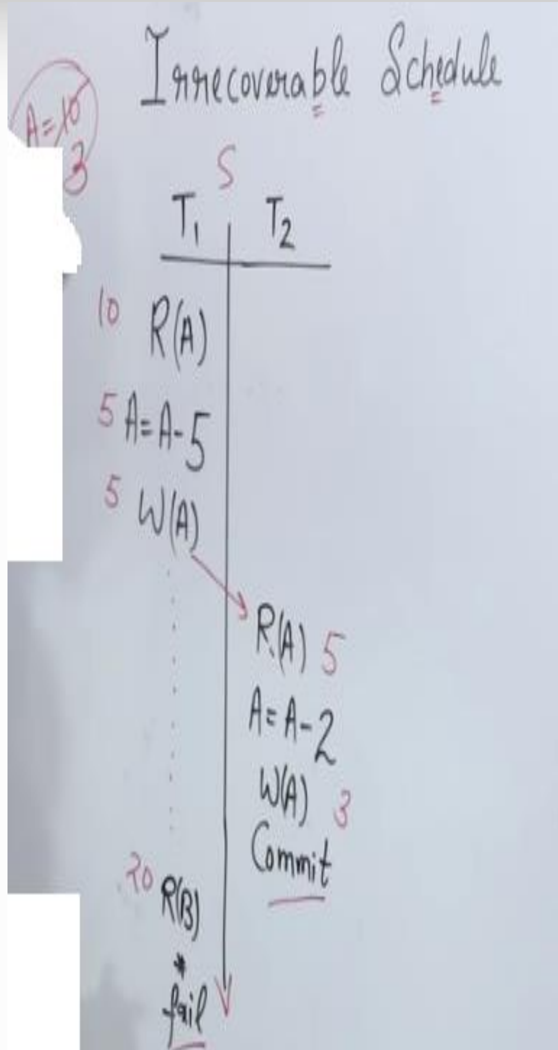
https://www.youtube.com/watch?v=vwIeKYGXmjI&list=PLxCzCOWd7aiFAN6I8CuViBuCdJgiOkT2Y&index=76

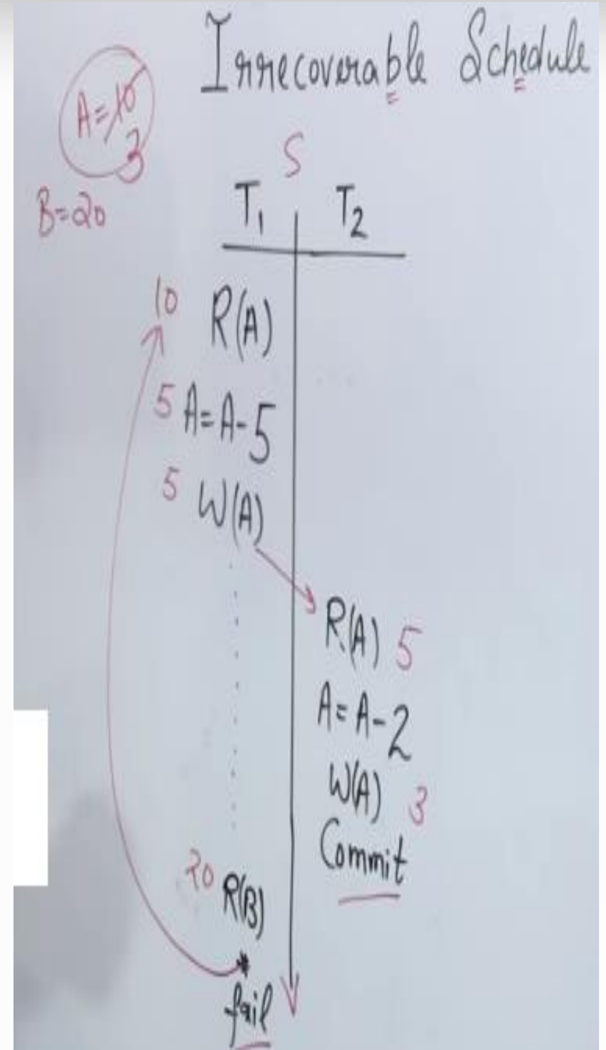


9

Irrecoverable Schedule

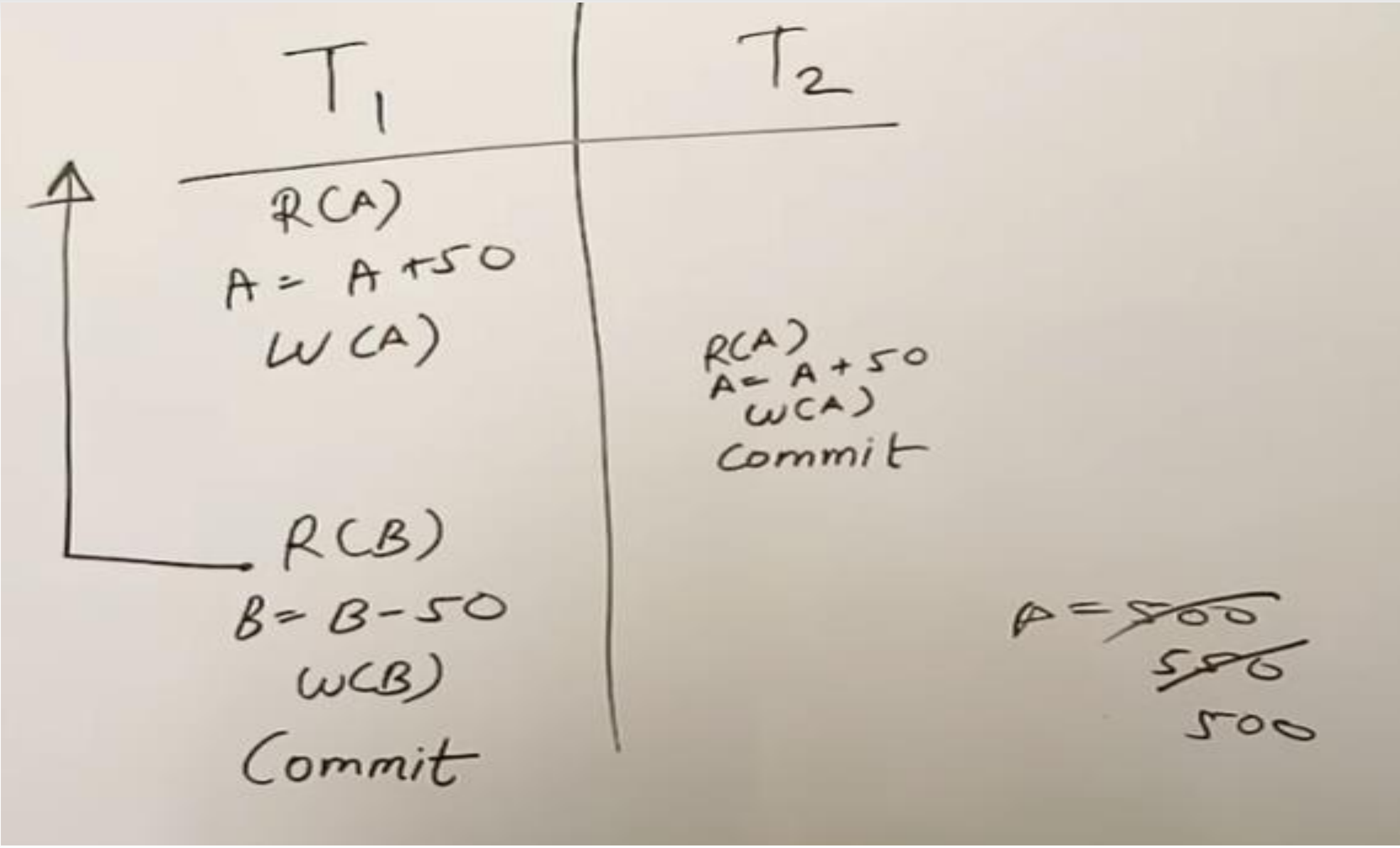






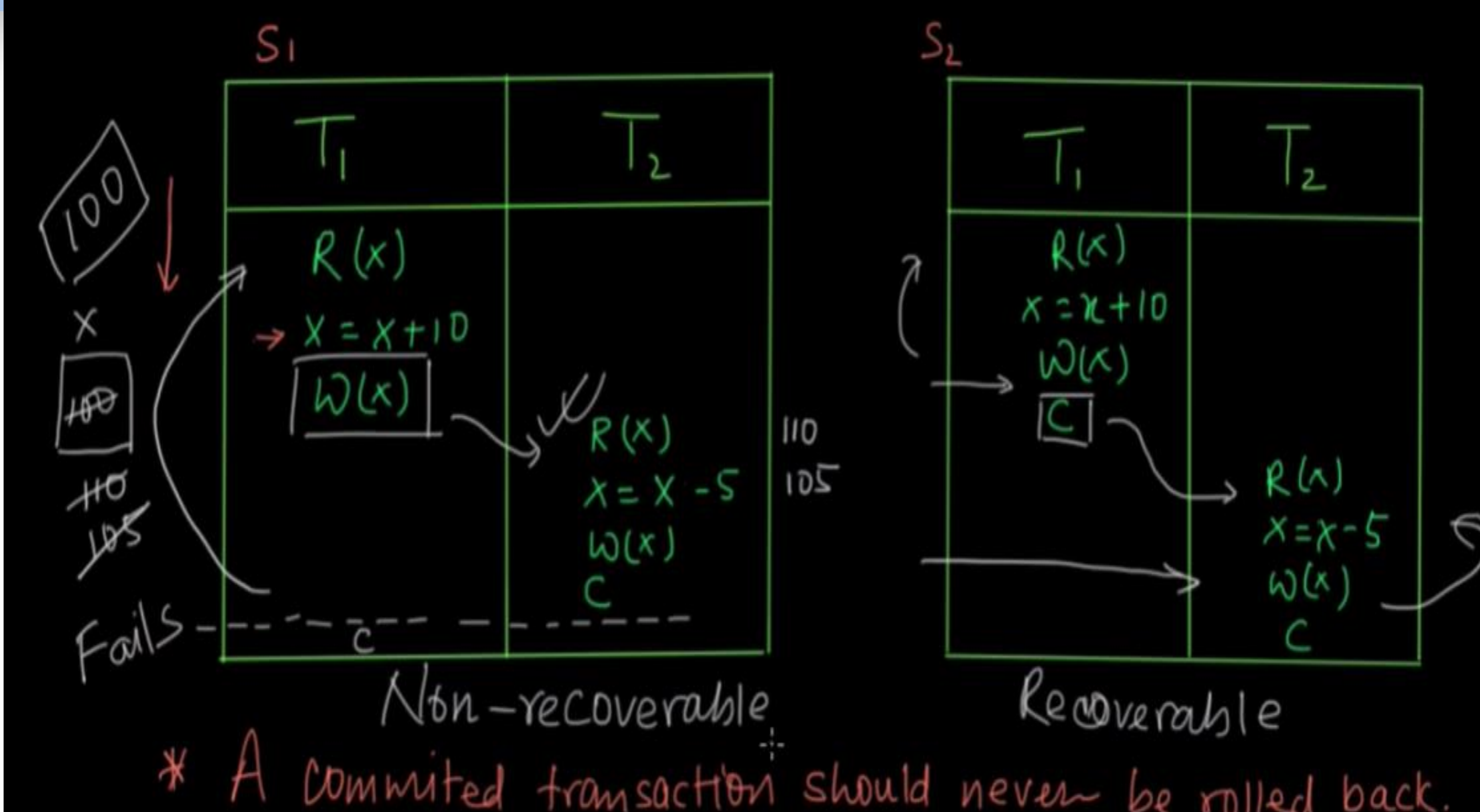
https://www.youtube.com/watch?v=g2gZKA8E1yA&list=PLxCzCOWd7aiFAN6I8CuViBuCdJgiOkT2Y&index=77

Irrecoverable Schedule Example 02



https://www.youtube.com/watch?v=nILQlEmaEcA

Recoverable and Nonrecoverable Schedule

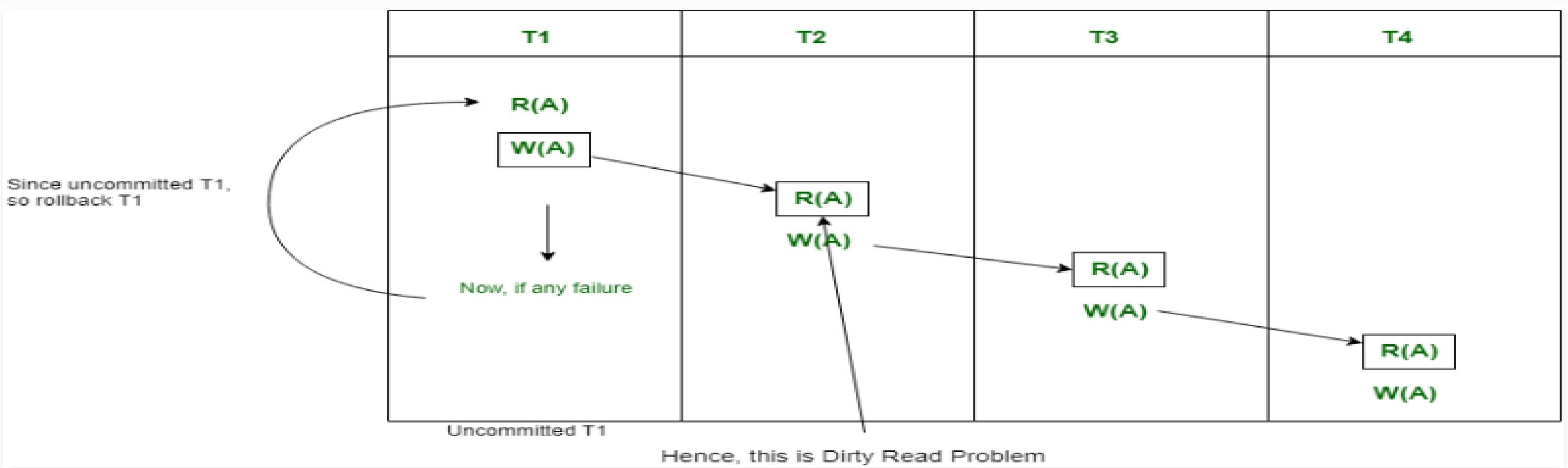


https://www.youtube.com/watch?v=jEqnfpaEO6M

What is cascading Schedule

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.



Cascadeless schedule?

When a transaction is not allowed to read data until the last transaction which has written it is committed or aborted, these types of schedules are called cascadeless schedules

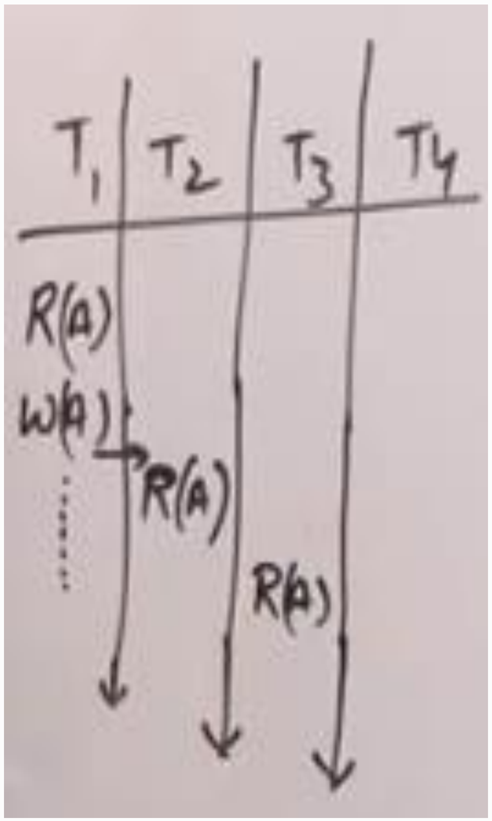
T1	T2
R(X)	
W(X)	
	W(X)
commit	
	R(X)
	Commit

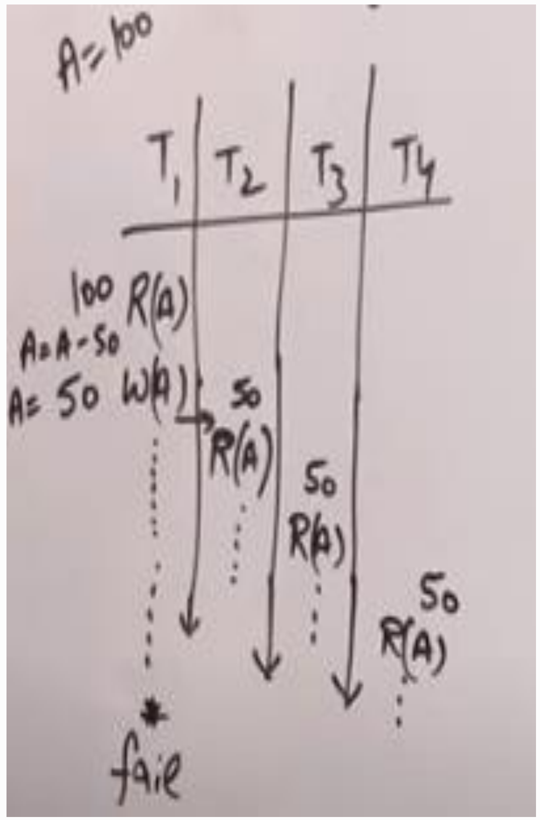
Here, the updated value of **X** is read by transaction T2 only after the commit of transaction T1. Hence, the schedule is cascadeless schedule.

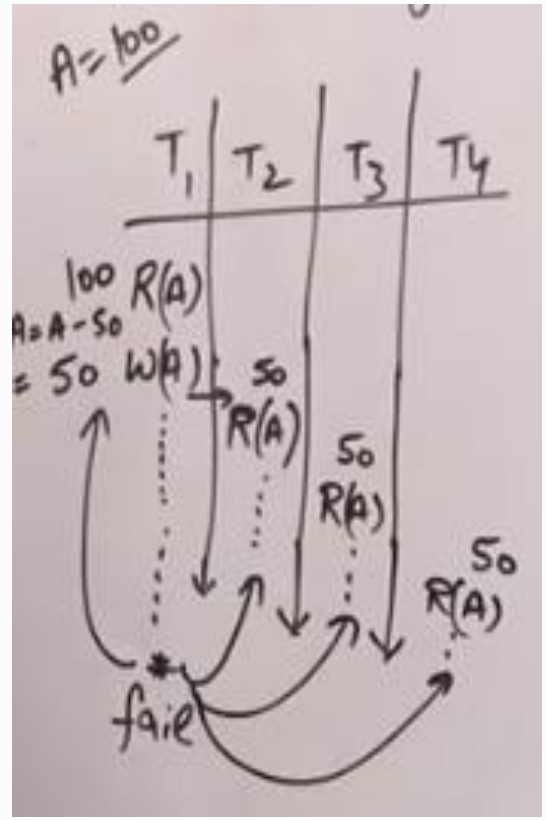
https://www.tutorialspoint.com/what-are-different-types-of-recoverability-of-schedules-dbms

Cascading Schedule VS Cascadeless Schedule

Cascading Schedule





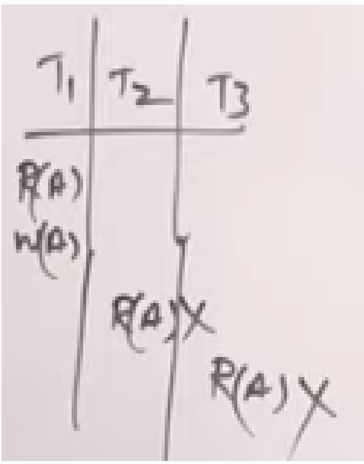


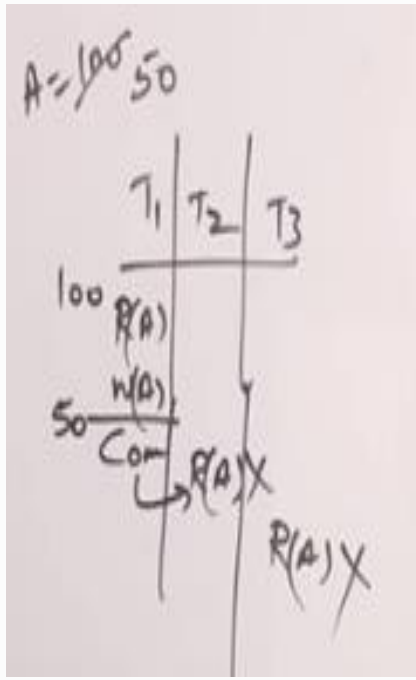
Disadvantage : Performance degrade (CPU was not utilized properly)

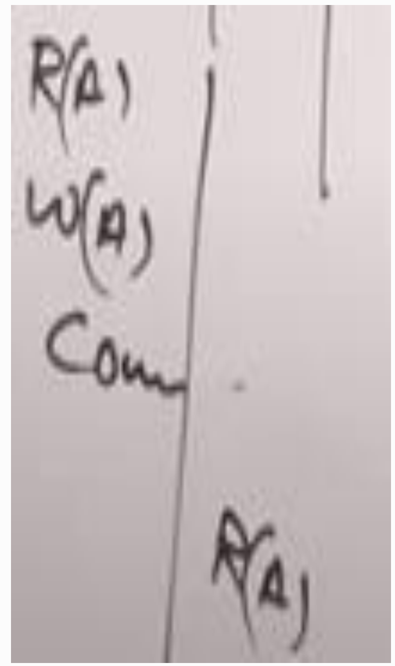
https://www.youtube.com/watch?v=Fxt1pk5LEBU&list=PLxCzCOWd7aiFAN6I8CuViBuCdJgiOkT2Y&index=78

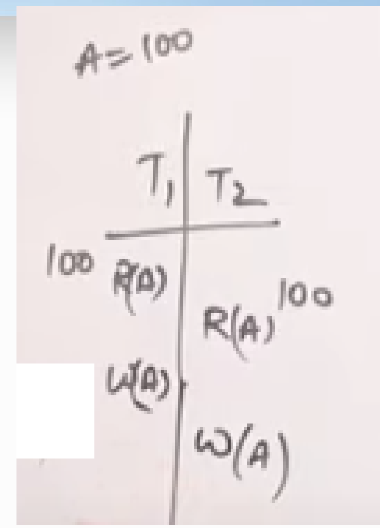
Cascading Schedule VS Cascadeless Schedule

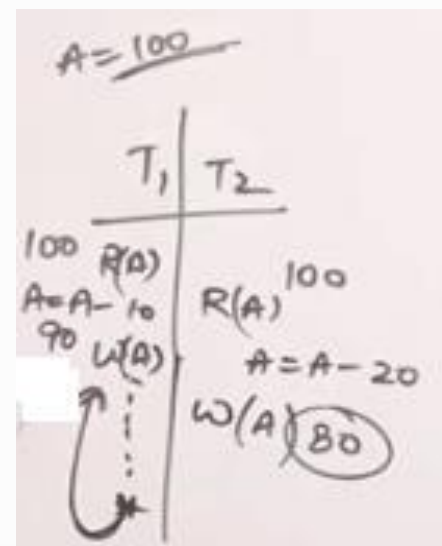
Cascadeless Schedule











https://www.youtube.com/watch?v=Fxt1pk5LEBU&list=PLxCzCOWd7aiFAN6I8CuViBuCdJgiOkT2Y&index=78

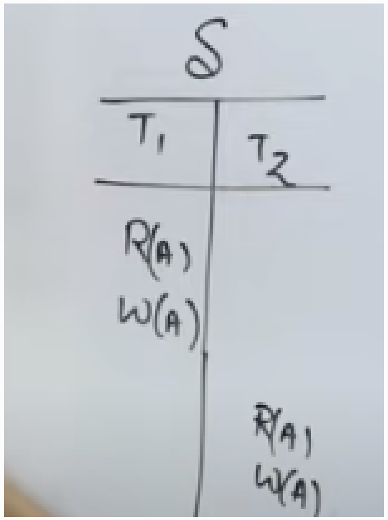
Strict schedule

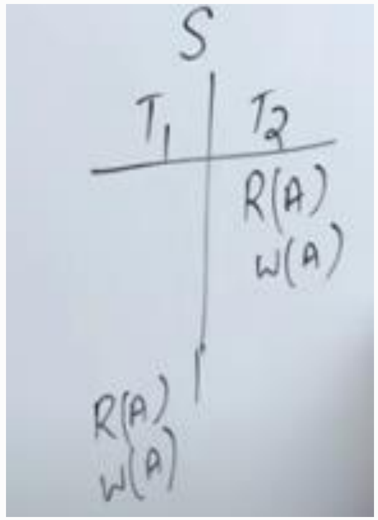
Given below is an example of a strict schedule −

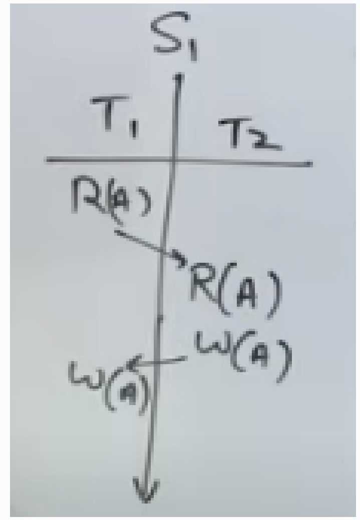
T1	T2
R(X)	
	R(X)
W(X)	
commit	
	W(X)
	R(X)
	Commit

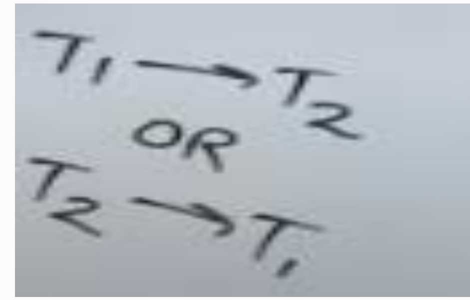
Here, transaction T2 reads and writes the updated or written value of transaction T1 only after the transaction T1 commits. Hence, the schedule is strict schedule.

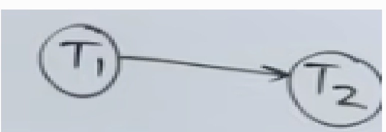
Serializability

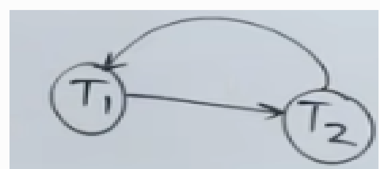






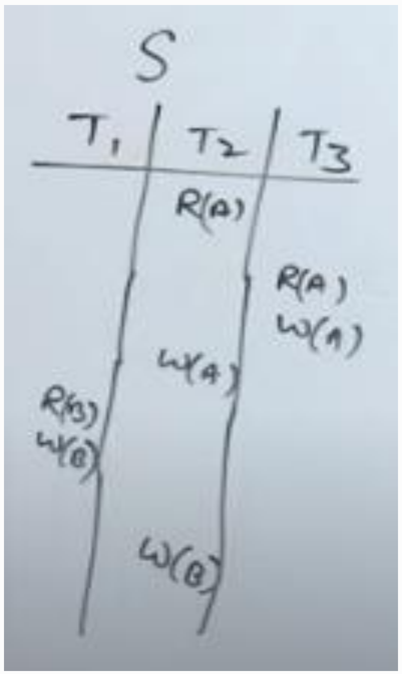


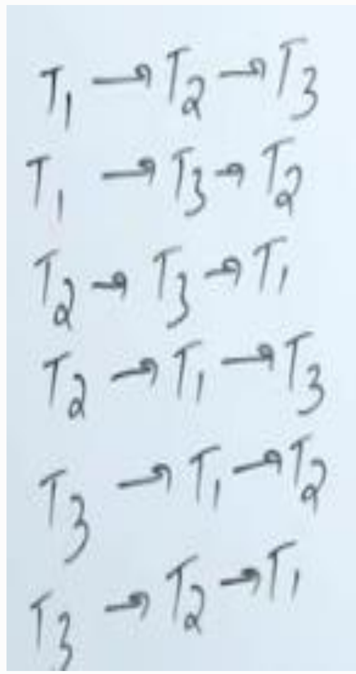


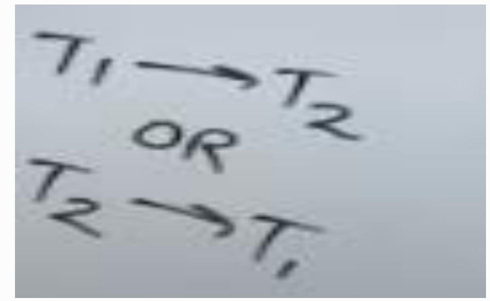


https://www.youtube.com/watch?v=s8QlJoL1G6w&list=PLxCzCOWd7aiFAN6I8CuViBuCdJgiOkT2Y&index=79

Serializability

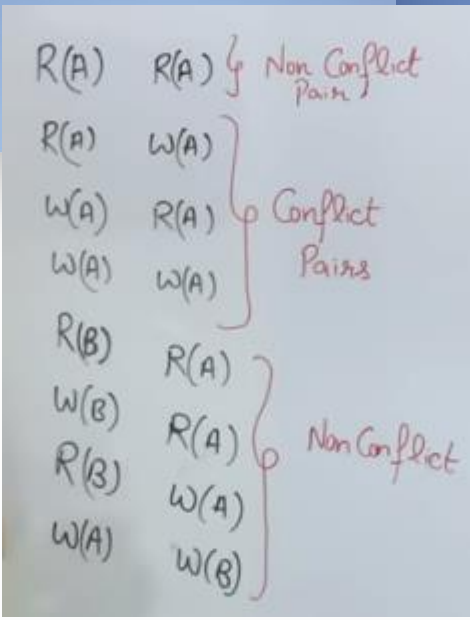


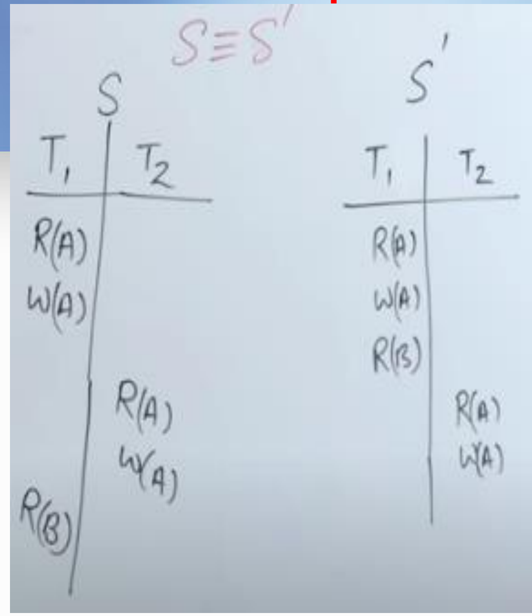


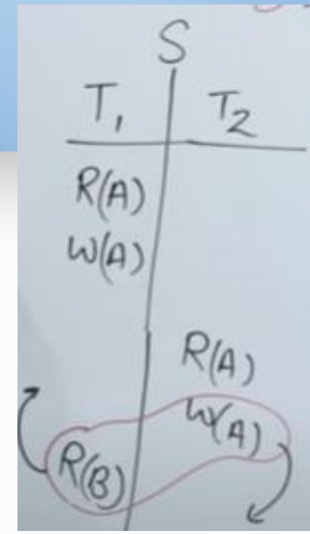


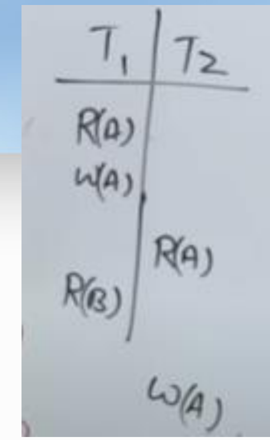
https://www.youtube.com/watch?v=s8QlJoL1G6w&list=PLxCzCOWd7aiFAN6I8CuViBuCdJgiOkT2Y&index=79

Conflict Equivalent

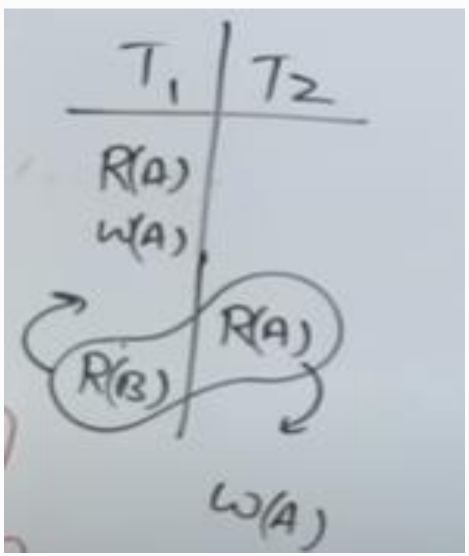


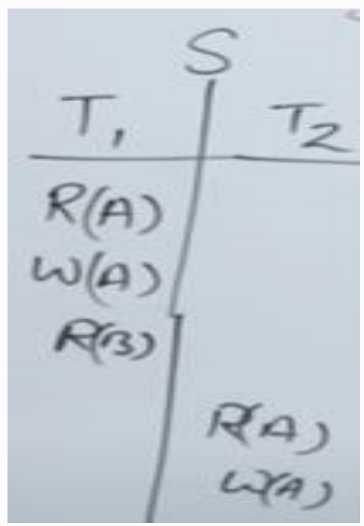






Adjacent Non Conflicting Pairs





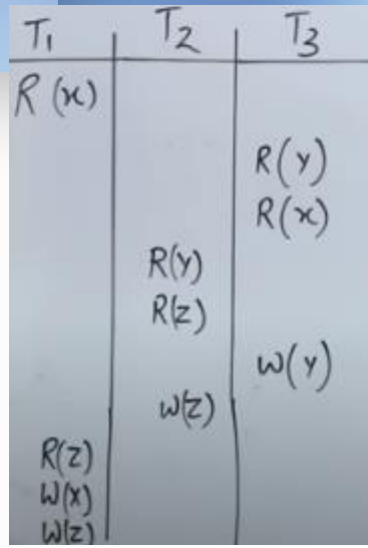
https://www.youtube.com/watch?v=ckqDozxECp0&list=PLxCzCOWd7aiFAN6I8CuViBuCdJgiOkT2Y&index=80

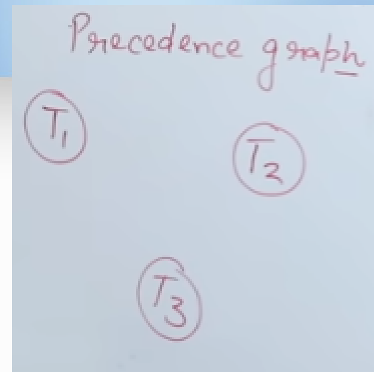
Conflict Serializability

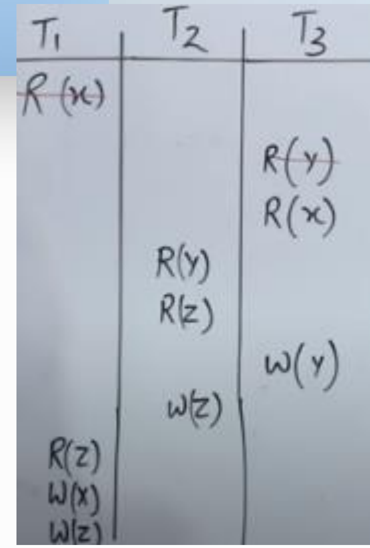
Check Conflict pairs in other transactions and draw edge

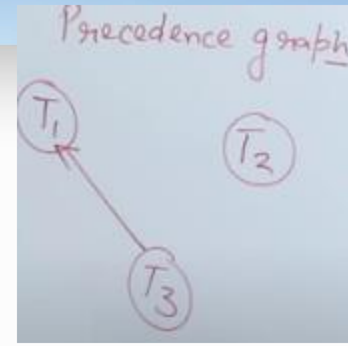
R(A) – W(A)
W(A)- R(A)
W(A)- W(A)

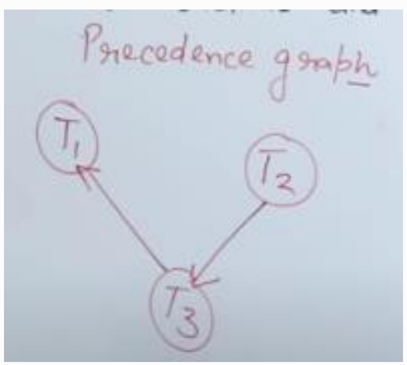
Check
Conflict Pairs





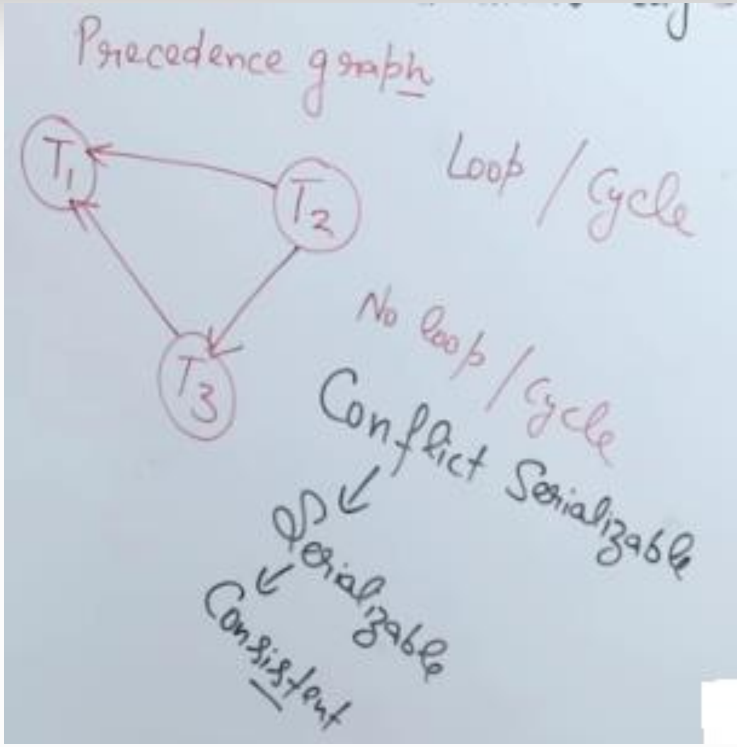






https://www.youtube.com/watch?v=zv0ba0Iok1Y&list=PLxCzCOWd7aiFAN6I8CuViBuCdJgiOkT2Y&index=81

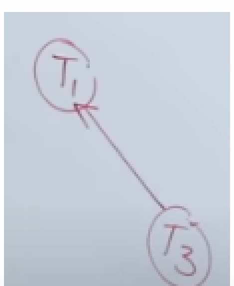
Conflict Serializability



T1->T2->T3
T1->T3->T2
T2->T1->T3
T2->T3->T1
T3->T1->T2
T3->T2->T1

Now Find vertex whose indegree =0

Here is T2



Now again Find vertex with indegre=0

T2->T3->T1

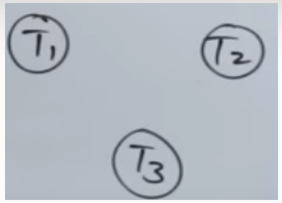
https://www.youtube.com/watch?v=zv0ba0Iok1Y&list=PLxCzCOWd7aiFAN6I8CuViBuCdJgiOkT2Y&index=81

Check whether Schedule is Conflict Serializable or not ?

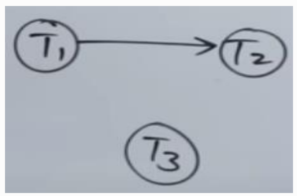
Use View Seriality to check Non serializable schedule

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

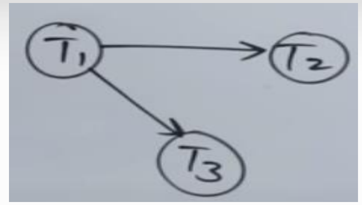
We need to check Conflict serializable through precedence graph



Check Conflict pairs

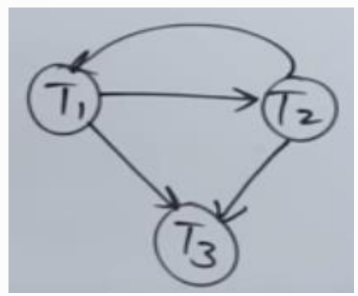


R(A) -> W(A)



R(A) -> W(A)

WA < R(A) W(A)



W(A)->W(A) from T2-T1 And T2-T3

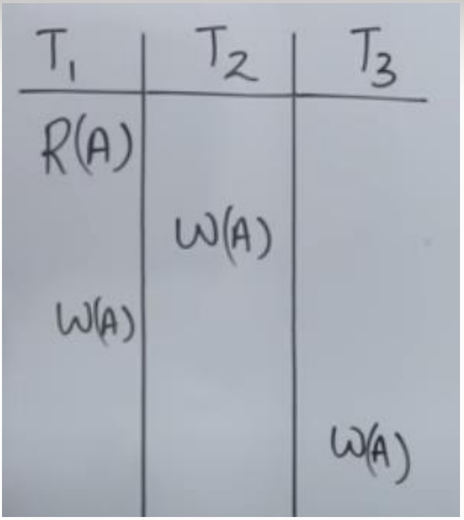
Non Conflict Serializable

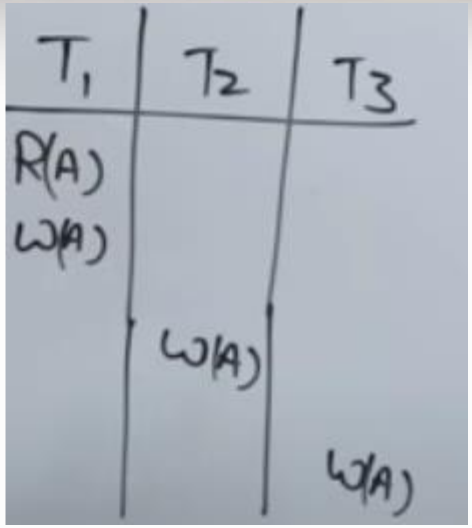
No Loop / Cycle
↓
CS
↓
Serial
↓
Consistent

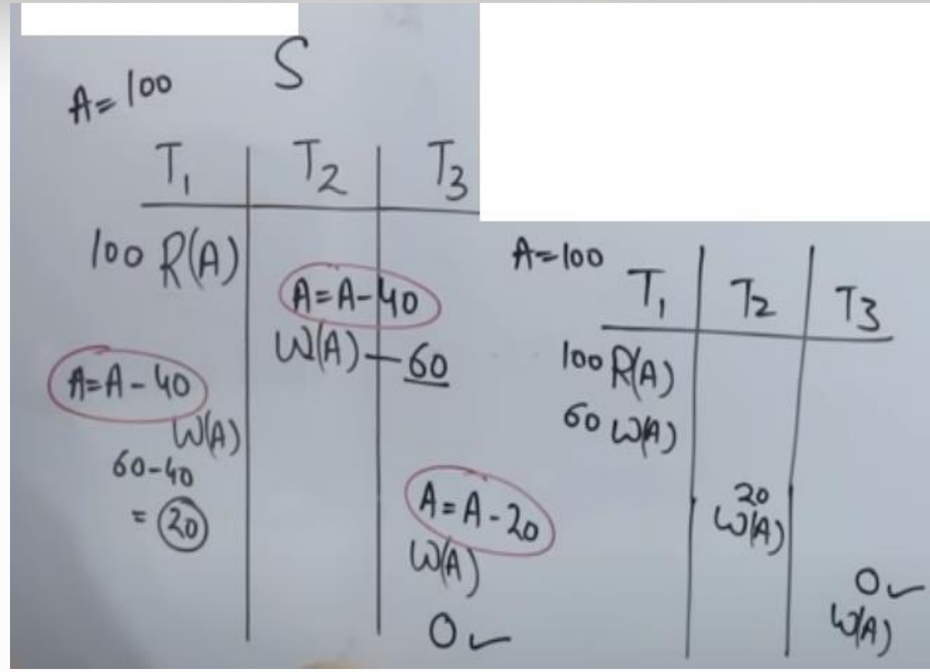
If it is non serializable, you can check it by view serialiable method

S

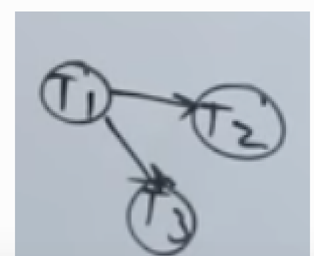
View Serialibility







Both schedules are equivalent.
It is not conflict equivalent but it is view equivalent



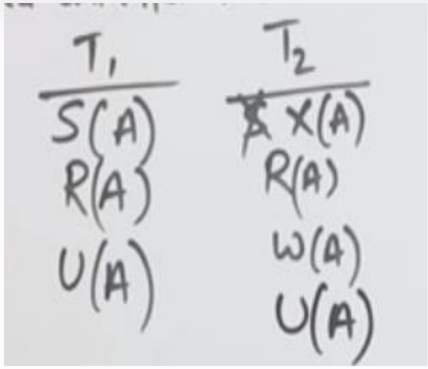
Shared - Exclusive Locking

Shared Lock(S) : If transaction is locked data item in shared mode, then allowed to read only.
Exclusive Lock(X) :If transaction is locked data item in shared mode, then allowed to read and write both .
Problems in Shared/Exclusive Locking

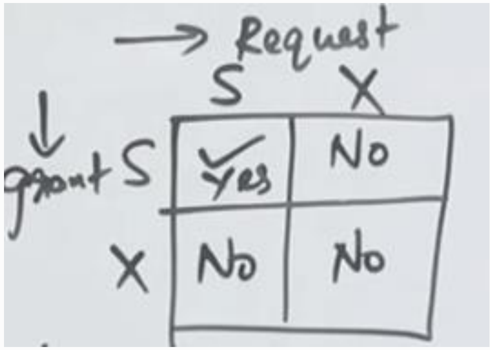
(i) May not sufficient to produce only serializable schedule
(ii) May not free from irrecoverabilty
(iii) May not free from deadlock
(iv) May not free from starvation

https://www.youtube.com/watch?v=94C0V7f2zm4&list=PLxCzCOWd7aiFAN6I8CuViBuCdJgiOkT2Y&index=83

Shared Exclusive Locking Continue



Compatibility table



SL->SL :
R(A) allowed

SL->XL :
R(A) & W(A) is not allowed when it is already in R(A)

XL-> SL :R(A) & W(A) is not allowed when it is already in R(A)

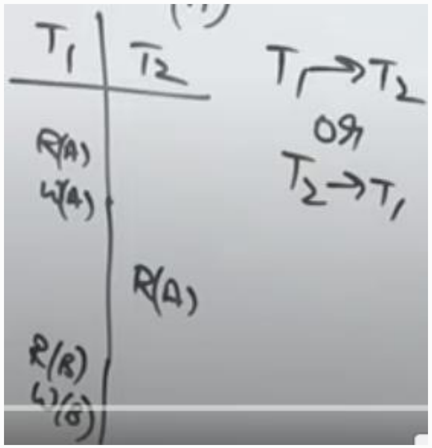
XL-> XL :R(A) & W(A) is not allowed when it is already in R(A) & W(A)

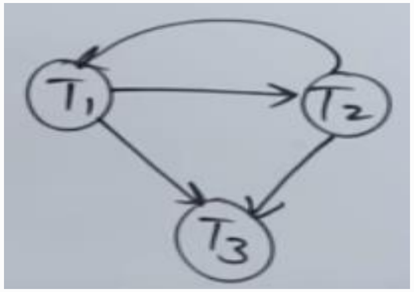
https://www.youtube.com/watch?v=94C0V7f2zm4&list=PLxCzCOWd7aiFAN6I8CuViBuCdJgiOkT2Y&index=83

Problems in Shared - Exclusive Locking Continue

.Problems in Shared/Exclusive Locking

(i) May not sufficient to produce only serializable schedule





T1	T2
X(A)	
R(A)	
W(A)	
U(A)	
	S(A)
	R(A)
	U(A)
X(B)	
R(B)	
W(B)	
U(B)	

In T2 : Can we get Shared lock on R(A)?

Answer No.

It will get shared lock when it will be released by T1

In T1 : Can we get Exclusive lock on R(B) & W(B)?

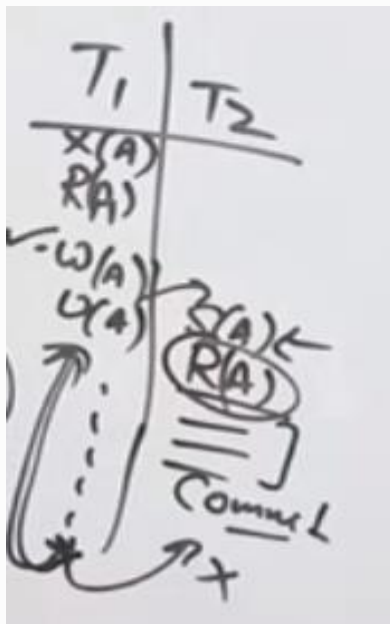
Answer Yes.

It will get Exclusive lock because it is using different variable i.e. B

https://www.youtube.com/watch?v=UsqtDD1VriY&list=PLxCzCOWd7aiFAN6I8CuViBuCdJgiOkT2Y&index=84

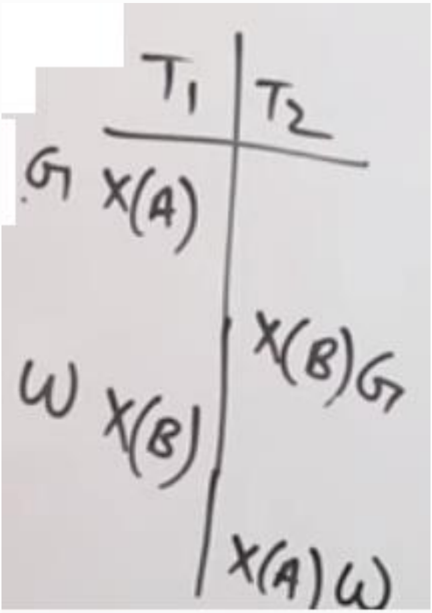
Problems Shared - Exclusive Locking Continue

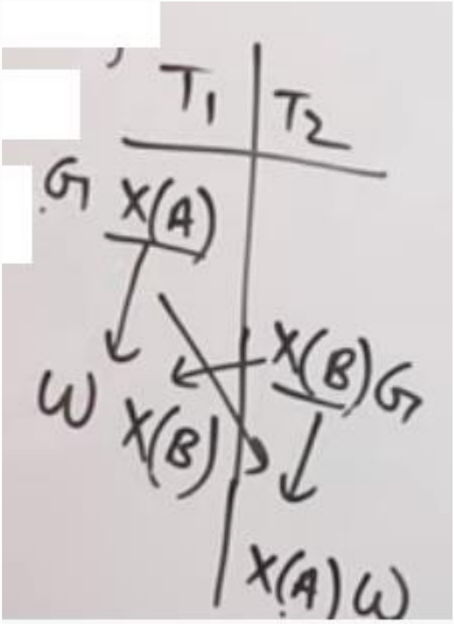
(ii)May not free from irrecoverabilty



Problems Shared - Exclusive Locking Continue

(iii) May not free from deadlock

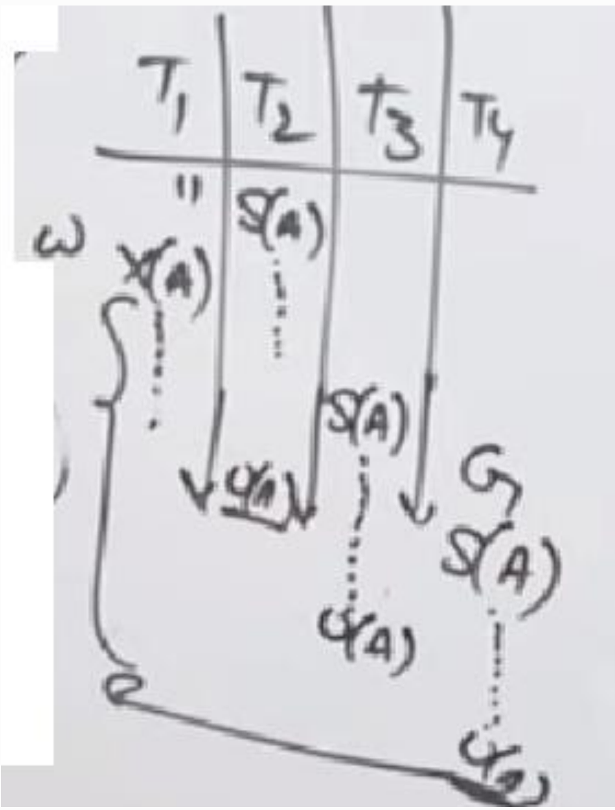




It is infinite waiting

Problems Shared - Exclusive Locking Continue

(iv) May not free from starvation



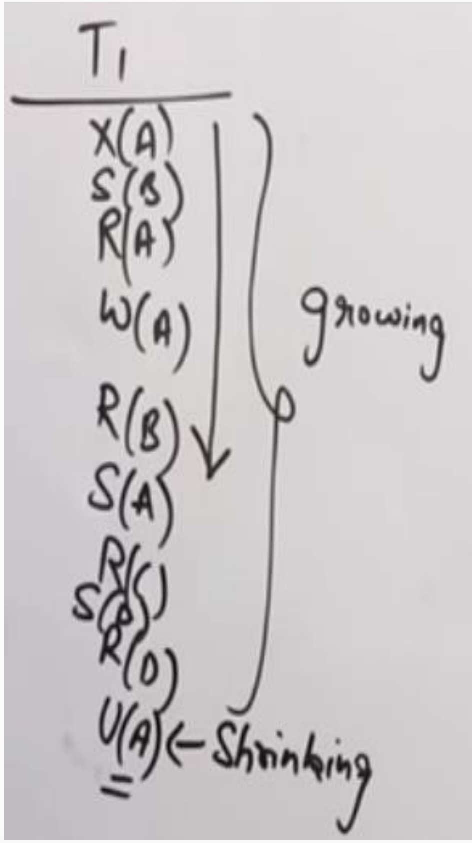
T1 is waiting for Exclusive Locking on **A**, It will be available as and when shared lock released by other transactions.

It is finite waiting

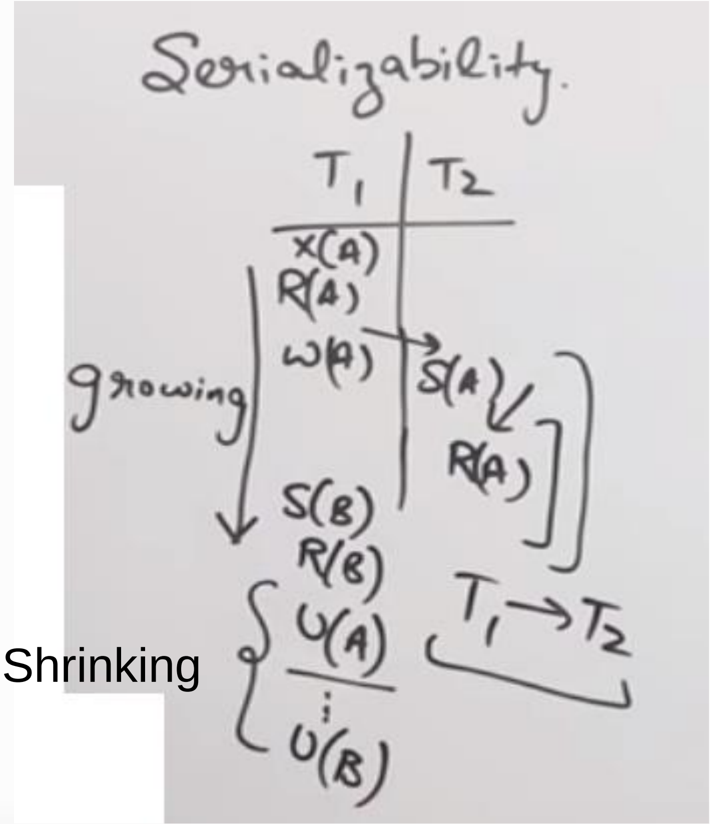
2 - Phase Locking (2 PL)

2 PL : It is an extension of Shared-Exclusive Locking.

Growing Phase : Locks are acquired and no lock is released.
Shrinking Phase:Locks are released and no locks are acquired.



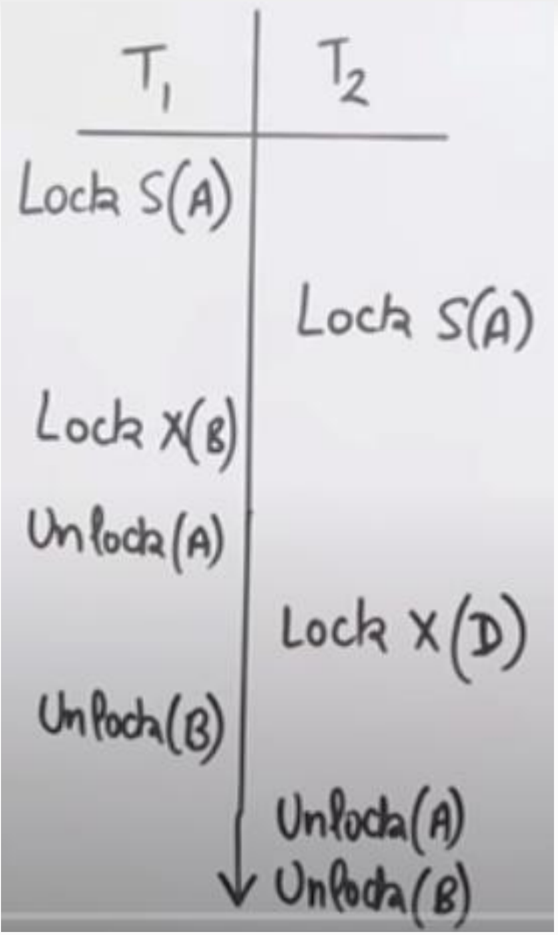
With this model, we can achieve serialiability



Shrinking

S(A) will not be granted to T2 because of X(A) in T1

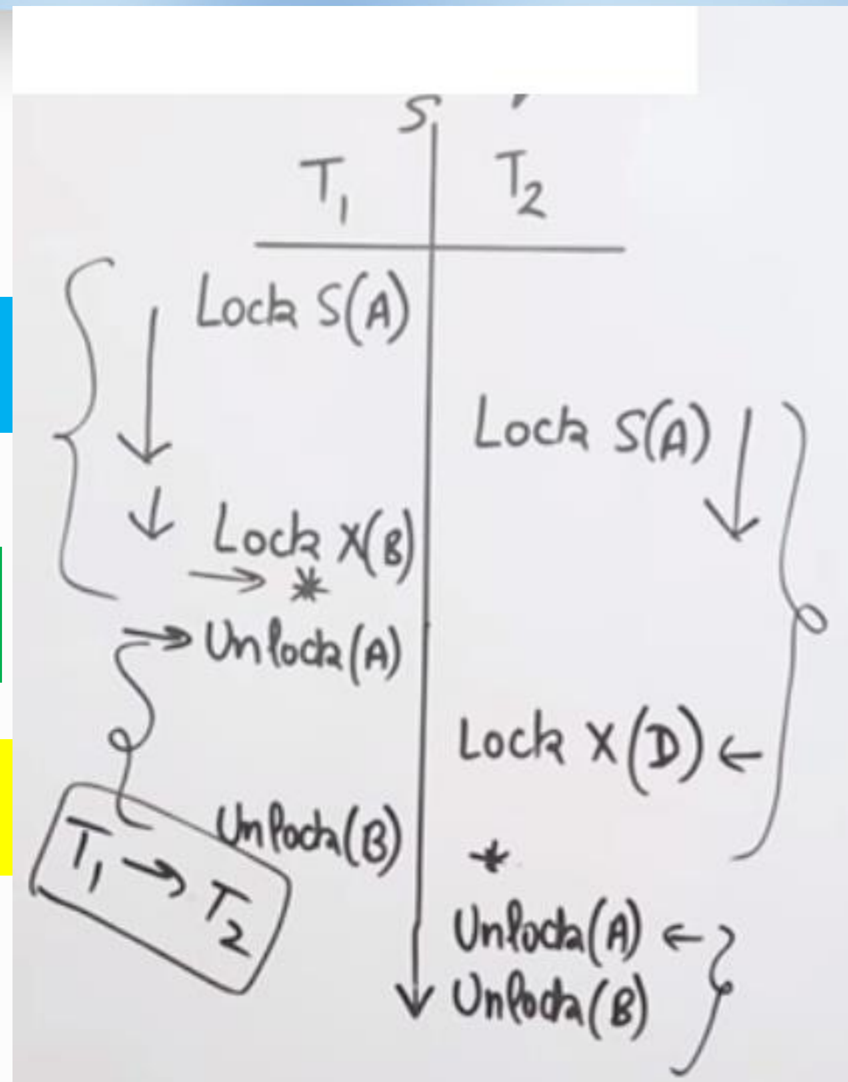
2 - Phase Locking (2 PL) Continue



Growing Phase

Lock Point

Shrinking Phase



Growing Phase

Shrinking Phase