SNS COLLEGE OF ENGINEERING

Kurumbapalayam (Po), Coimbatore – 641 107

An Autonomous Institution

Accredited by NBA – AICTE and 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 8 : Serializability

Serializability

- Serializability is the classical concurrency scheme.
- It ensures that a schedule for executing concurrent transactions is equivalent to one that executes the transactions serially in some order.
- It assumes that all accesses to the database are done using read and write operations.

- *Read-Write Synchronization*: If a transaction reads a value written by another transaction in one schedule, then it also does so in the other schedule.
- *Write-Write Synchronization*: If a transaction overwrites the value of another transaction in one schedule, it also does so in the other schedule.

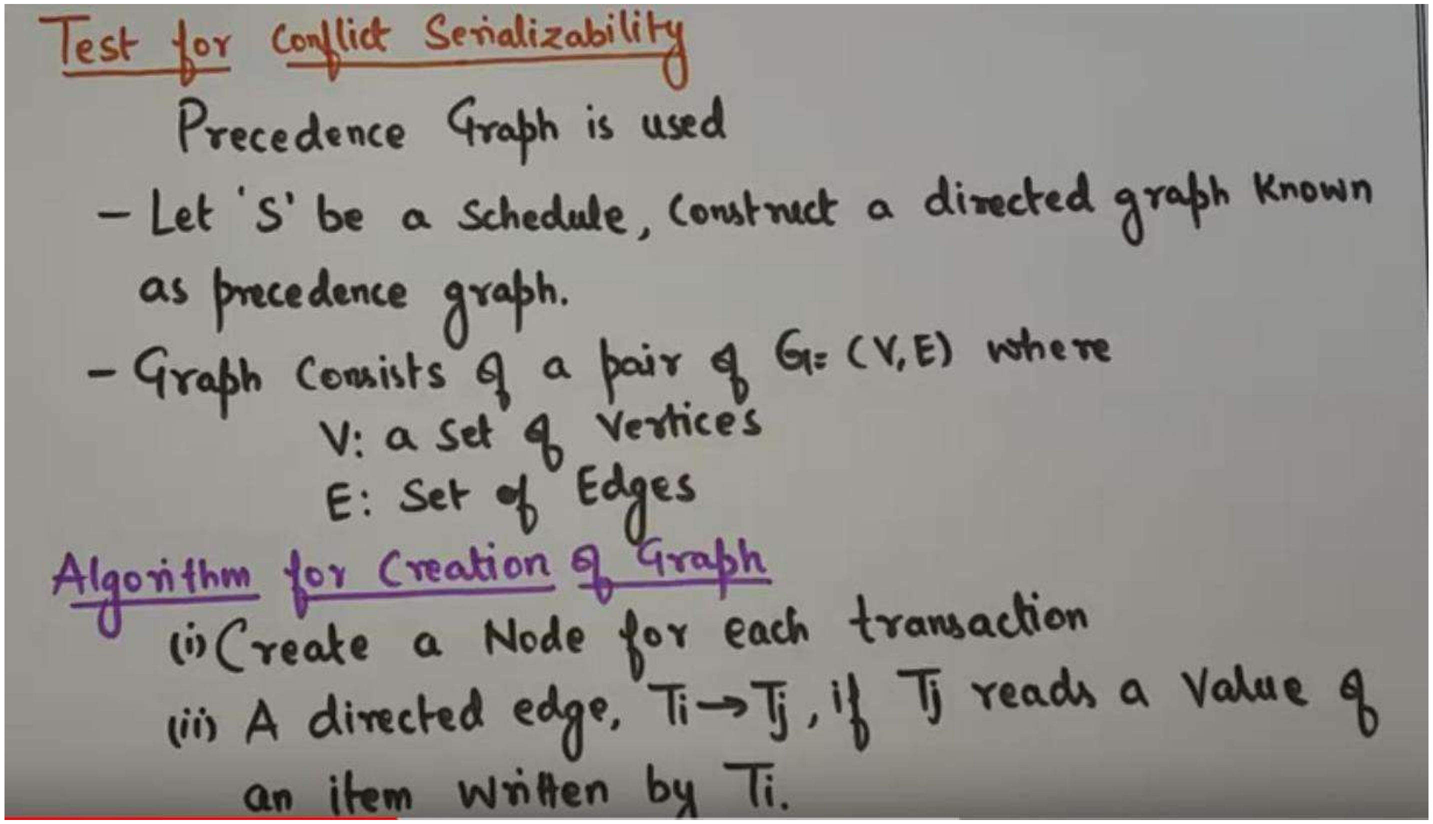
Conflict Serializability

- **Conflict Serializability**
  Two instructions of two different transactions may want to access the same data item in order to perform a read/write operation.
- Conflict Serializability deals with detecting whether the instructions are conflicting in any way, and specifying the order in which these two instructions will be executed in case there is any conflict.
- A **conflict** arises if at least one (or both) of the instructions is a write operation. :

- The following rules are important in Conflict Serializability
- If two instructions of the two concurrent transactions are both for read operation, then they are not in conflict, and can be allowed to take place in any order.
- If one of the instructions wants to perform a read operation and the other instruction wants to perform a write operation, then they are in conflict, hence their ordering is important.
- If the read instruction is performed first, then it reads the old value of the data item and after the reading is over, the new value of the data item is written.
- It the write instruction is performed first, then updates the data item with the new value and the read instruction reads the newly updated value.

- Execute all the operations of transaction T1 (in sequence) followed by all the operations of transaction T2 (in sequence).

Test for Conflict Serialzability



Test for Conflict Serializability

Precedence Graph is used

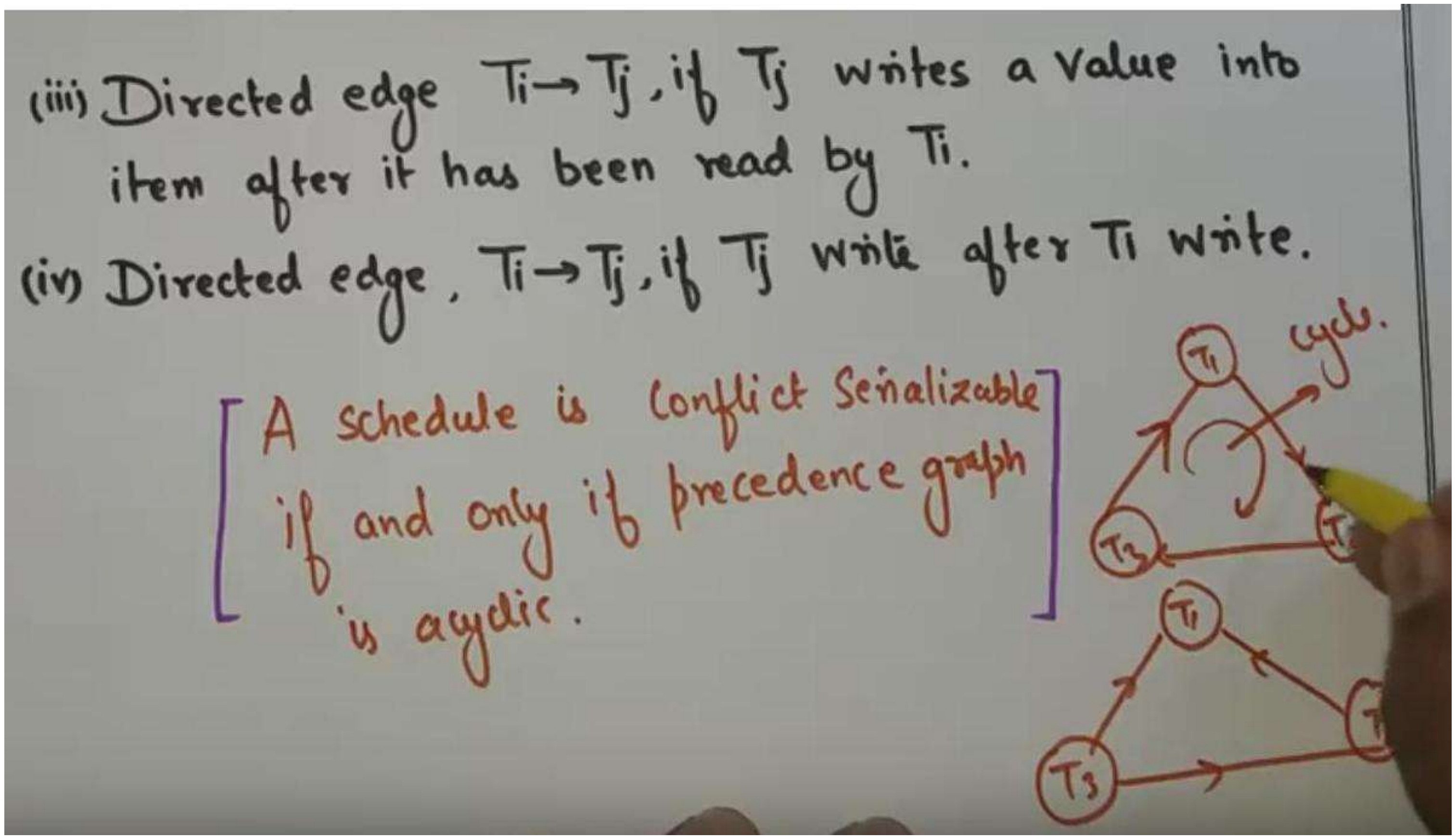
- Let 'S' be a Schedule, Construct a directed graph known as precedence graph.
- Graph Consists of a pair of G=(V,E) where
  - V: a set of vertices
  - E: Set of Edges

Algorithm for Creation of Graph

(i) Create a Node for each transaction

(ii) A directed edge, $T_i \rightarrow T_j$, if T_j reads a value of an item written by T_i.

Cont..



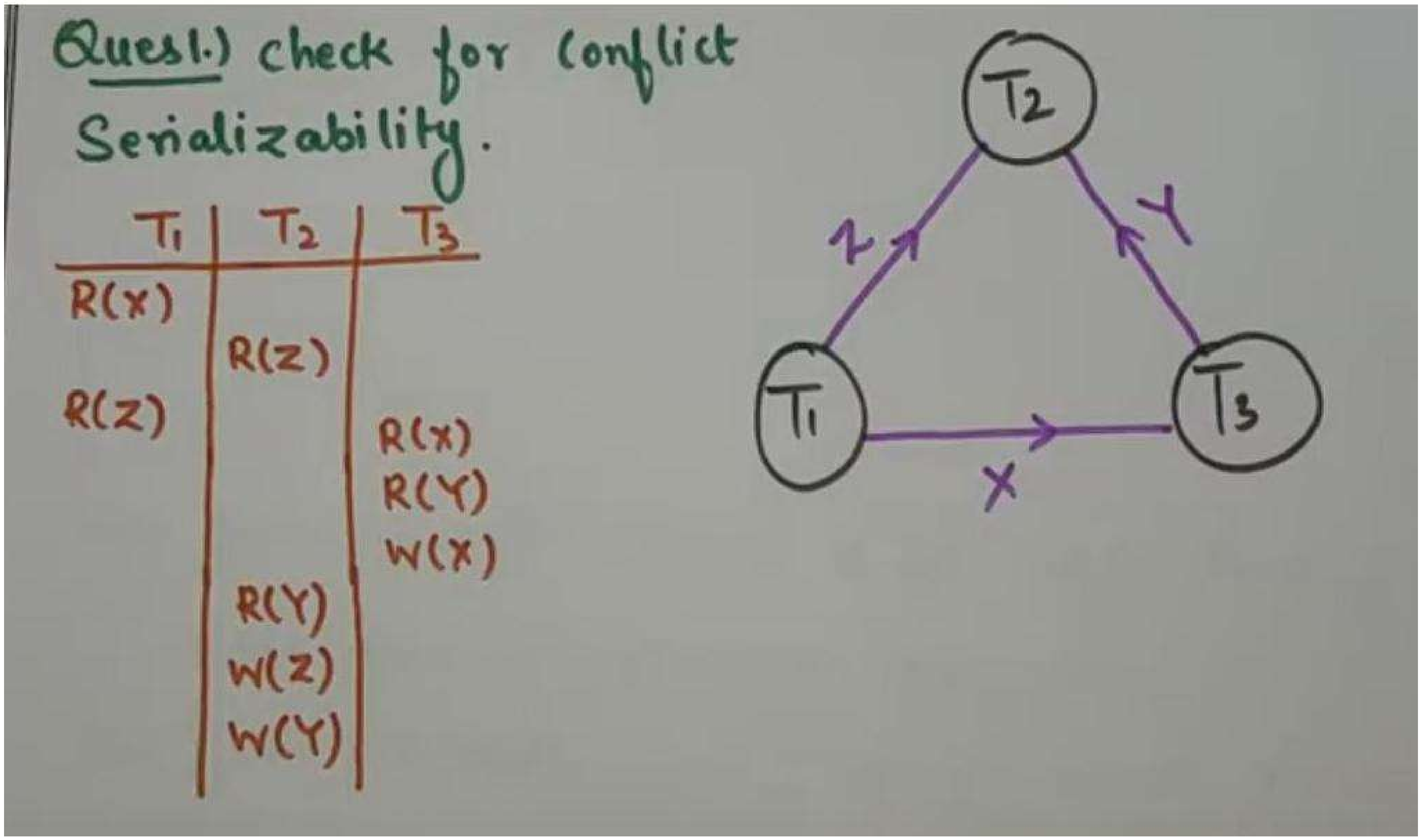
(iii) Directed edge $T_i \rightarrow T_j$, if T_j writes a value into item after it has been read by T_i.

(iv) Directed edge, $T_i \rightarrow T_j$, if T_j write after T_i write.

[A schedule is conflict Serializable if and only if precedence graph is acyclic.]

cycle.
T1
T2
T3
T1
T2
T3

Cont..

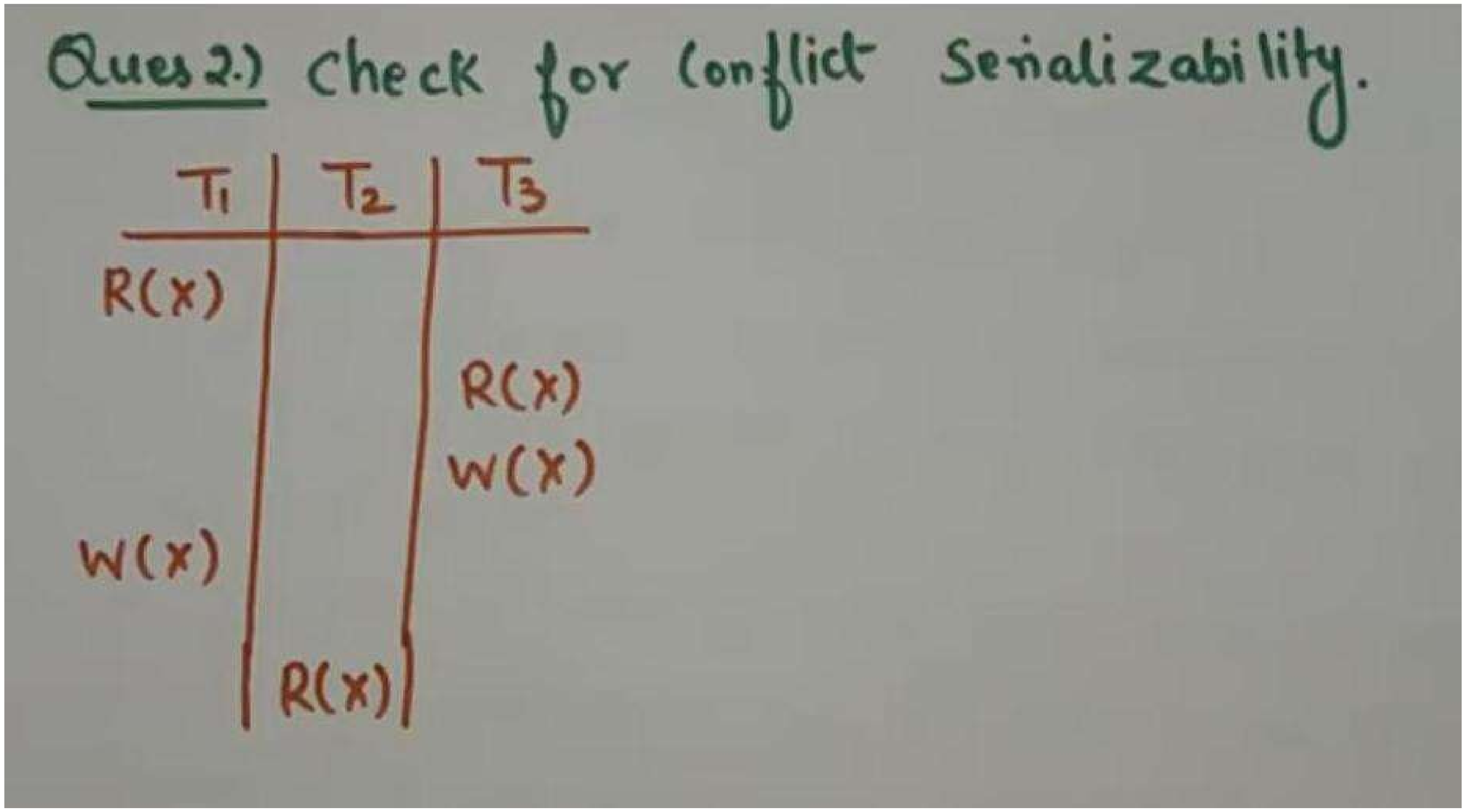


Ques1.) check for conflict Serializability.

T1	T2	T3
R(X)		
	R(Z)	
R(Z)		
		R(X)
		R(Y)
		W(X)
	R(Y)	
	W(Z)	
	W(Y)	

$$\begin{tikzcd} & T_2 & \\ T_1 \arrow[ur, "Z"] \arrow[rr, "X"'] & & T_3 \arrow[ul, "Y"'] \end{tikzcd}$$

Cont..

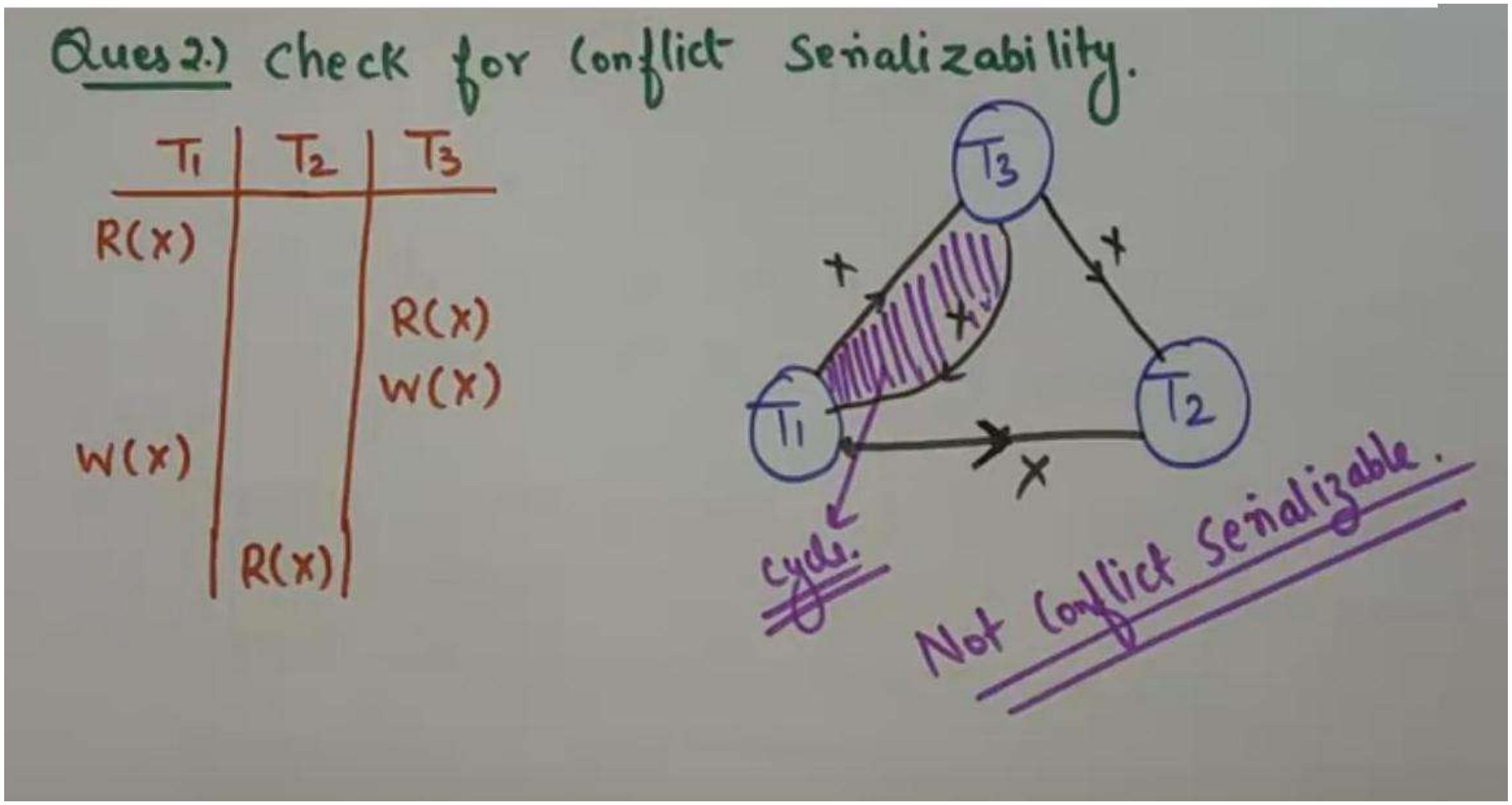


Ques 2.) Check for Conflict Serializability.

T1	T2	T3
R(x)		
		R(x)
		W(x)
W(x)		
	R(x)	

Cont..

Answer



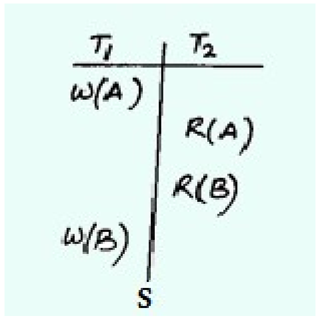
Ques 2.) Check for conflict Serializability.

T1	T2	T3
R(X)		
		R(X)
		W(X)
W(X)		
	R(X)	

Example of Serializable Schedule

Let us consider a schedule S.

•

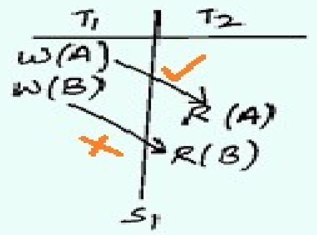


What the schedule S says ??

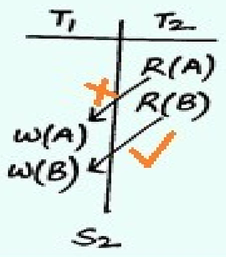
•Read A after updation.

•Read B before updation.

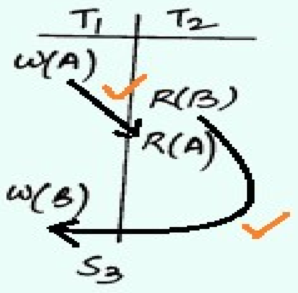
- Let us consider 3 schedules S1, S2, and S3. We have to check whether they are serializable with S or not ?



It is reading B after updation.
∴ Not serializable.
$\boxed{S \neq S_1}$

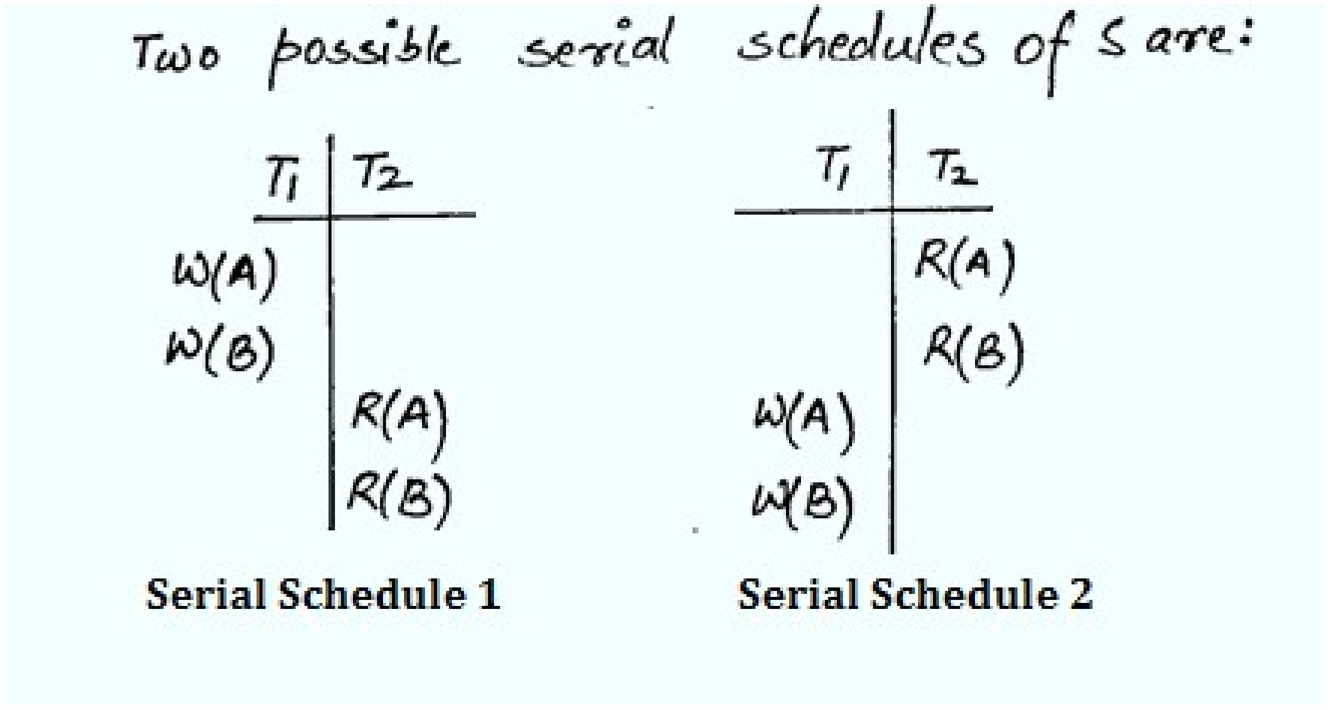


As it is reading A from DB
i.e before updation.
∴ Not serializable
$\boxed{S \neq S_2}$



As it is reading A after updation
& reading B before updation
∴ serializable
$\boxed{S = S_3}$

- **Example of Serial Schedule :**
- Consider the above schedule S. The serial schedules will be



Two possible serial schedules of S are:

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

Serial Schedule 1

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

Serial Schedule 2

- If both the transactions are for write operation, then they are in conflict but can be allowed to take place in any order, because the transaction do not read the value updated by each other. However, the value that persists in the data item after the schedule is over is the one written by the instruction that performed the last write.

Thank you