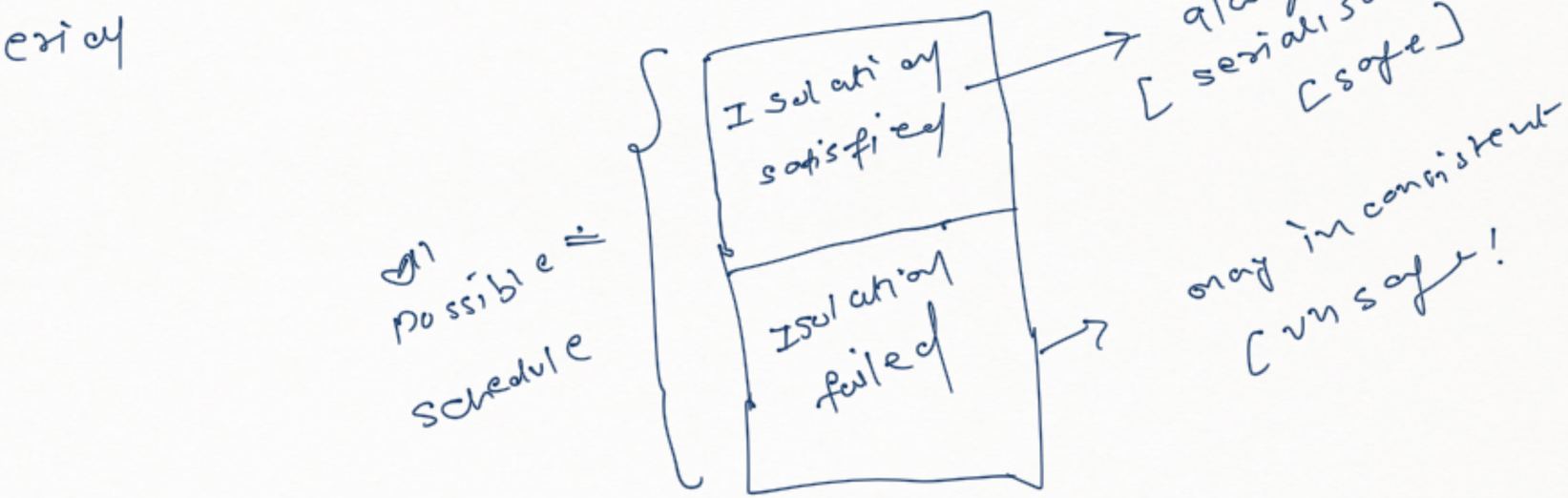
Isolation : Concurrent execution of two or more trans. result must be equal to the result of any serial schedule :

* If any concurrent schedule follow Isolation (or) equal to any serial schedule, know as serializable schedule (or)

Concurrent execution of trans. must be equal to any serial



Goal of concurrency control component! - " schedule if schedule is
" should not allow execution of any
failed Isolation rule ! -

ACI D

C: Consistency! [It is User responsibility]
⇒ DB operations requested by the user must be logically correct.
⇒ Transaction written by the user must be logically correct if it is wrong then the o/p is wrong

⇒ Atomicity : [Recov. mgmt, comp.]
Durability : — " —
Isolation : [concurrency control] } DBMS resp
Consistency : [user] [DBA] ⟶ DBA responsibility ! -

Q. $T_1, T_2 \_ \_ \_ T_n$: n Transactions :-

How many possible serial schedule :

$T_1 \to T_2$

$T_2 - T_1$

| $T_1 : T_2 : T_3$ —

$T_1 : T_3 : T_2$ ~

$T_2 : T_1 : T_3$

$T_2 : T_3 : T_1$

$T_3 : T_1 : T_2$

$T_3 : T_2 : T_1$

$\Rightarrow$ n! possible serial schedule.

$= 4!$ ~

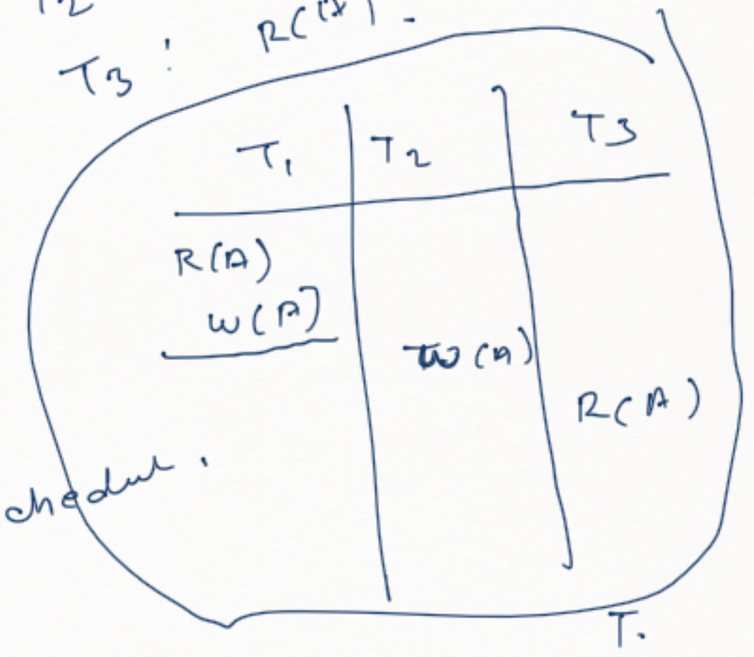
$= 8 \times 3$

$= 24$

$T_1 : R(A)\ W(A)$

$T_2 : W(A)$.

$T_3 : R(A)$.



T_1 : $R_1(A)\ W_1(A)\ R_1(B)\ W_1(B)$

T_2 : $R_2(A)\ R_2(B)$.

How many concurrent schedule possible ?

$\Rightarrow$ $R_1(A)\ W_1(A)\ R_1(B)\ W_1(B)$

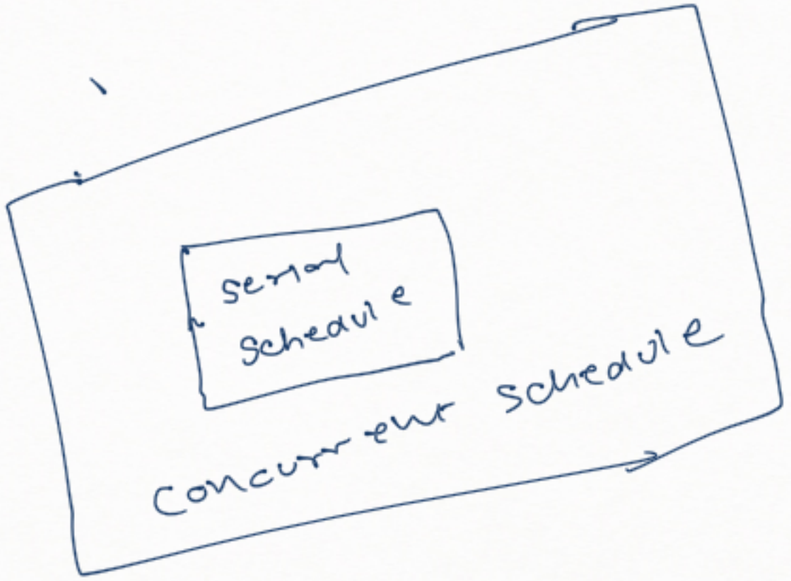
$\_\ \_\ \_\ \_\ \_\ \_ = \frac{6!}{4! * 2!} = 15$

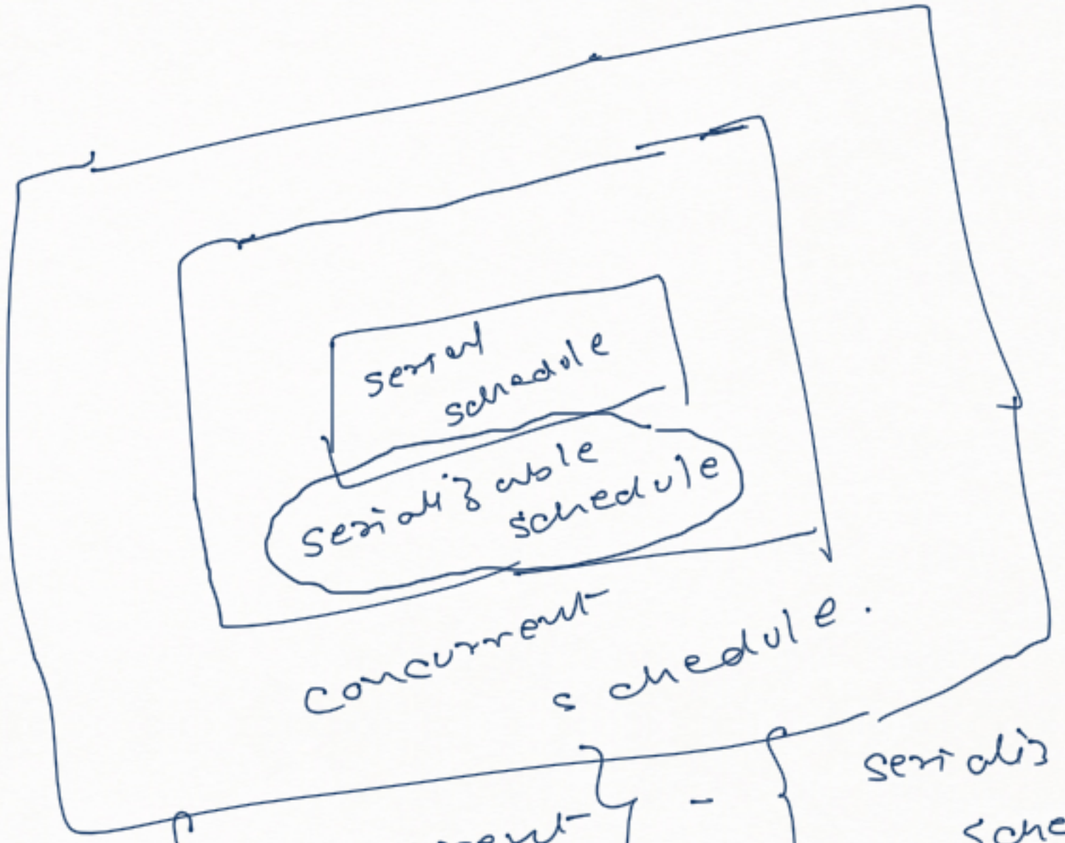
$\Rightarrow$ 15 concurrent schedule = 15 [serial + Non-serial]

$= {}^6C_2 =$

* = # of non serial schedules = { 11 concurrent schedule }
- { serial schedule }

$= 15 - 2 = 13$

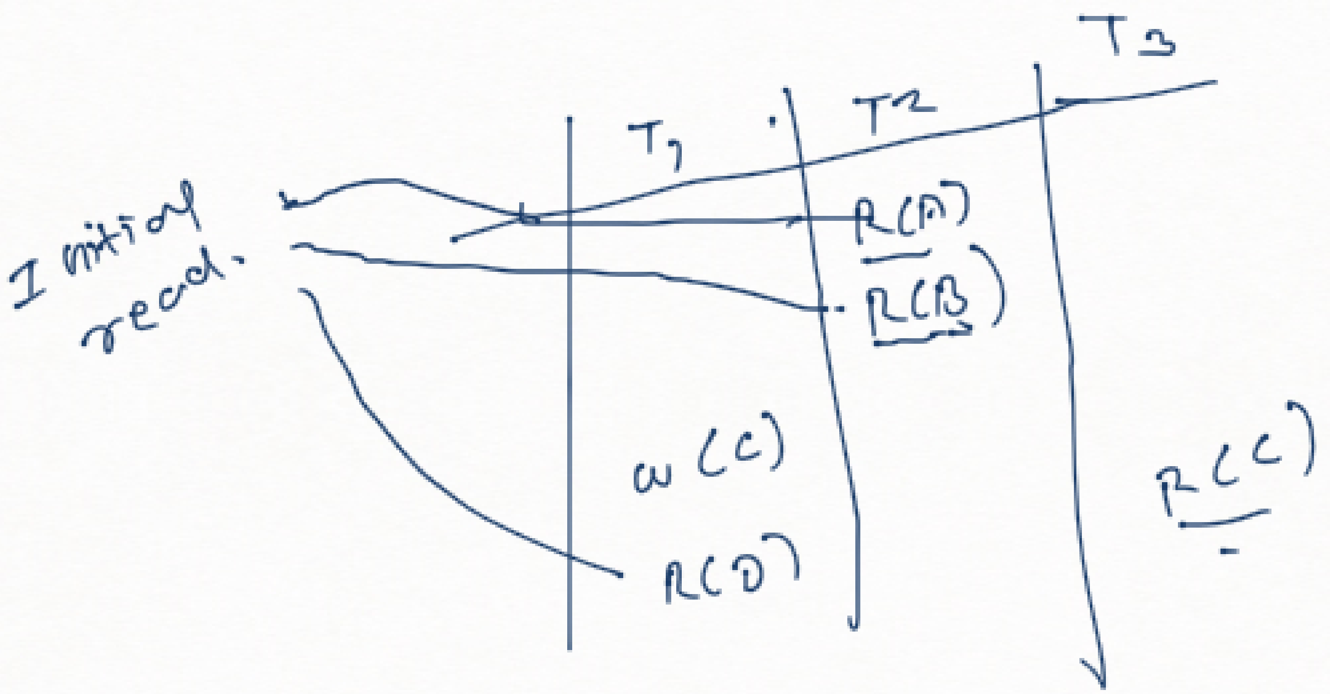

serial
Schedule
Concurrent Schedule

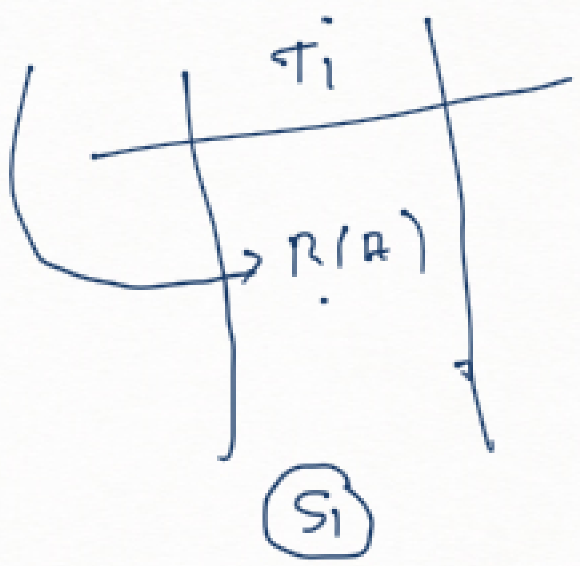


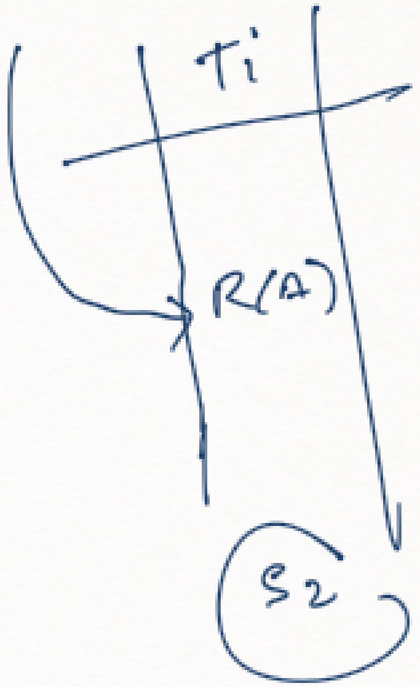
{ Non serializable schedule } = { concurrent schedule } − { serializable schedule }

⇒ S_1, S_2 (view) equal iff ⇒

(i) If some transaction T_i Reading "A" from initial DB in S_1. }
Then in S_2 also T_i should read A from initial DB.
[all initial read op. of S_1 & S_2 should be same].

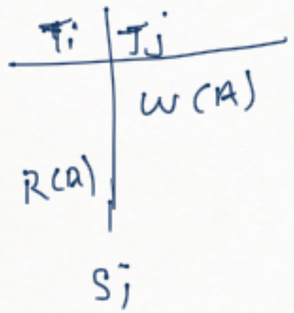


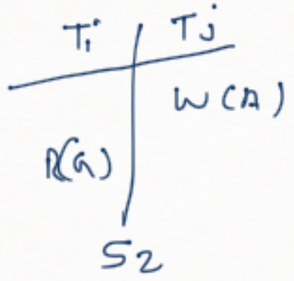


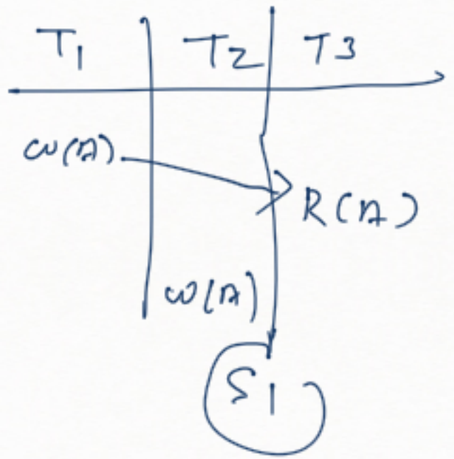


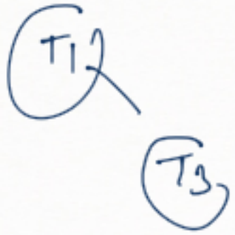
and (2) If Transaction T_i reads "A" which is updated by T_j in S_i then in S_2 also T_i should read "A" updated by T_j only

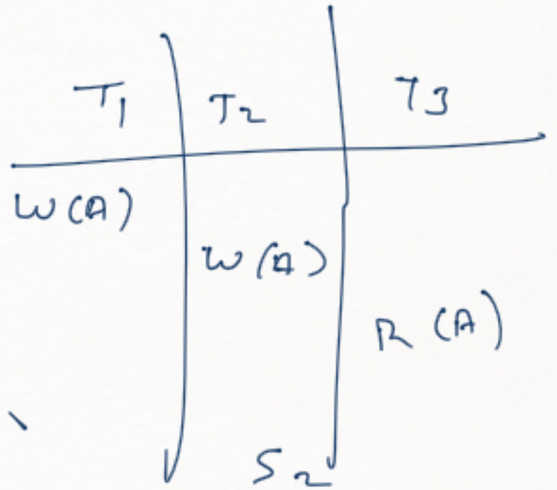
[write read sequence should be same in S_1 & S_2).











(3) If transaction T_i update "A" finally in S_1 they in S_2 also "A" should update by T_i finally