DBMS →

Transaction : Set of logical relational operation

T_1	T_2
R(A)	
A=A+50	
W(A)	
	R(B)

T_1	T_2

defn

Trans

Schedule : set of transaction. It may be single also.

Serial Schedule

Non-Serial
Concurrent Schedule
Concurrency schedule

T_1	T_2
R(A)	
A=A-50	
W(A)	
	R(B)
	A=A+50
	W(B)

T_1	T_2
R(A)	
	R(B)
A=A+50	
	B=B+50
W(A)	
	W(B)
C	

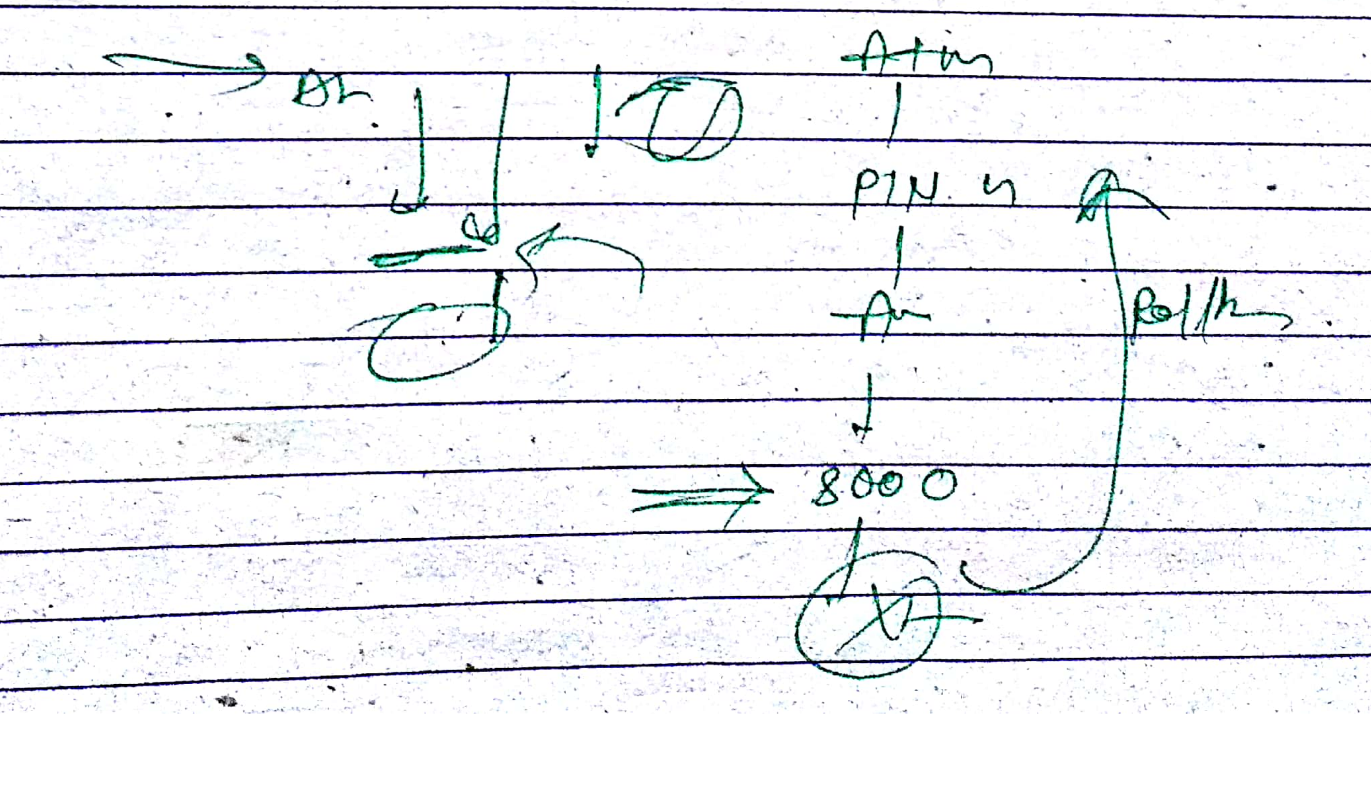
Why we use concurrency Schedule

- less time requ
- Repose time ↓
- Efficiency ↑
- Better perfon of resource

Properties of Transaction

- A - Atomicity
- C - Consistency
- I - Isolation
- D - Durability

Atomicity - Either all operation execute in trasactn or none.



Consistency :- before transaction value of A = after value of transaction u ... be equal

A = 50   B = 870

T_1	T_2
A	
A = A - 20	
w(A)	
	B
	B = B + 20
	w(B)

A + B = Total

A + B = 16

A = 31   ... = 10

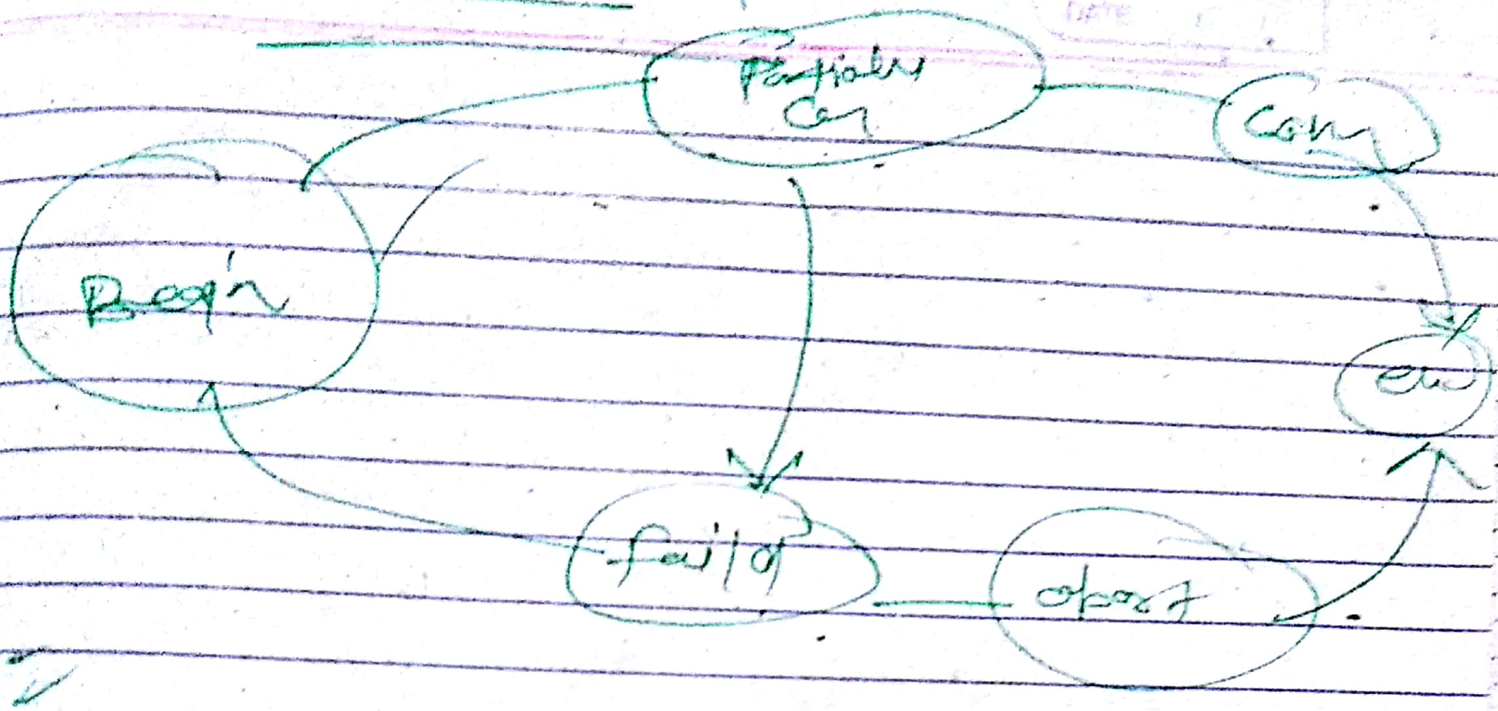
Isolation :- at one time only one transaction will be executed → T_1 | T_n

Durability - In durability our partially committed data is fully commited or update into database

database

14 - 15 ... used backup and Recovery

Transaction State



Checking consisten

1. conflict ser
2. view
3. Recov
4. Cascadless
5. Strict Schedule

⇒ Conflict → tay is process by which we can check give transaction is consisten or not

In other process of checking consistan of a transaction dependency

for checking conflict ser

T_1	T_2

1) Transaction never same
2) Data item same

- R(A) → W(A)
- W(A) → R(A)
- W(A) → W(A)

R(A) → R(A) (Never)

T_1	T_2	T_3
R(A)		
		W(A)
	R(B)	
		W(B)
R(C)		
	R(C)	
		W(C)
R(D)		

W(D)

W(D)

R(D)

Dead lock Concept