
500 +500.

1) Atomicity
2) consistency – a transaction shouldn't have database in an inconsistent state
3) Isolation – different transactions should be exclusive of each other
4) Durability

26/3/18

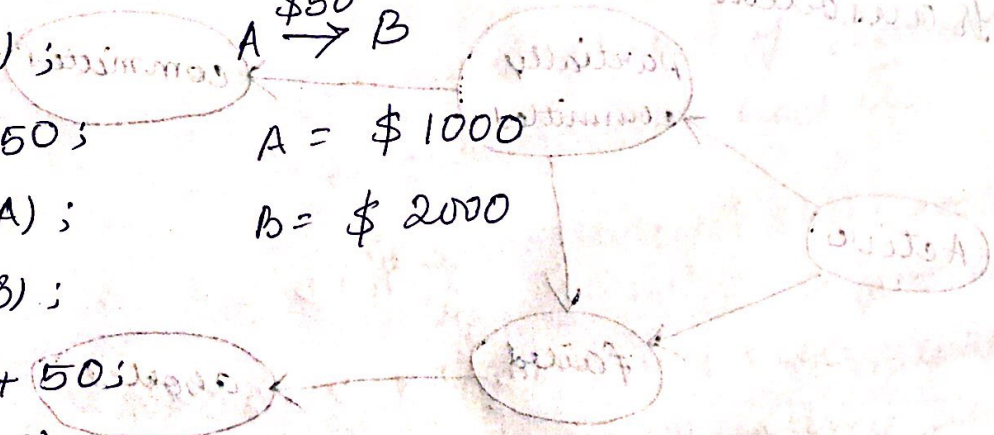
read(X) – copy of x will be transferred to local buffer
write(X) – X = X + 50 – update in the local buffer first
← write then changes made in the secondary memory (disk)

Ti: local buffer

$A \xrightarrow{\$50} B$

$T_1$ : read(A);
A := A - 50;
write(A);
read(B);
B := B + 50;
write(B);

A = $1000
B = $2000



Consistency – after execution of a transaction, DB must arrive at some consistent state.

Atomicity – effect of transaction must be fully updated in the DB & transaction is successful. changes should be permanent if somehow transaction is aborted then the changes must be brought back to Original.

**Durability** – whenever a transaction is successfully completed, the effect of the transaction persist in the database in a durable manner.

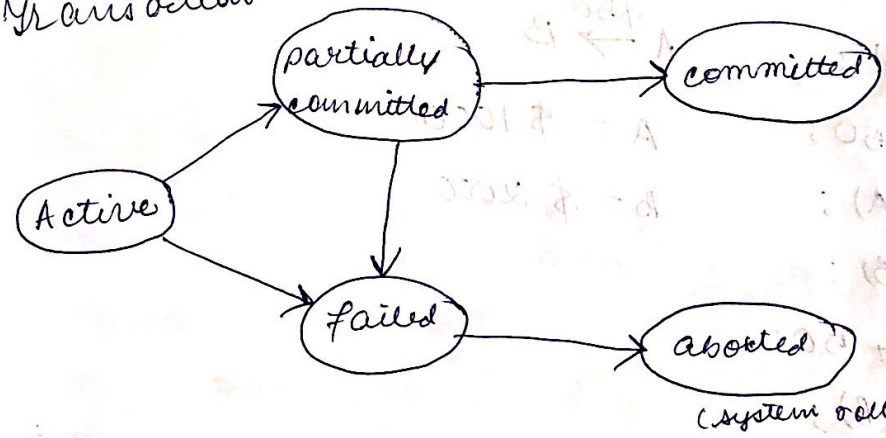
**Isolation** – $T_1$ & $T_2$ ← serial schedule – two transactions execute (~~simultaneously~~).

Increase throughput, decreases waiting time

## States of Transaction

1) Aborted –
2) committed –
3) Partially committed –
4) Failed – prior state to abortion
5) Active –

Transaction transition state Diagram :




committed and aborted states ends in terminated state

28/3/18

## Transaction schedules

$T_1$ : read (A);
A := A - 50;
write (A);
read (B);
B := B + 50;
write (B)

$T_2$ : read (A);
temp := A * 0.1;
A := A - temp;
write (A);
read (B);
B := B + temp;
write (B);