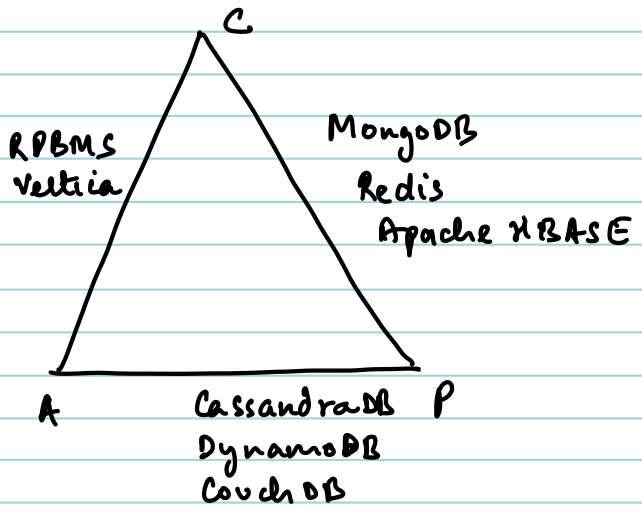
CAP Theoram [Brewer's Theoram]

C = Consistency → All nodes see same data all the time
A = Availability → Each request recieves a response (recent or not)
P = Partition Tolerance. → System continues to operate despite n/w partitions of comm^n failures b/w nodes



" For a distributed System, It is impossible to achieve all 3(CAP) simultaneously.

During Partition Tolerance [Network failure], either Consistency or Availability can be provide by a system (DB).

ACID

- Atomicity
- Consistency
- Isolation
- Durability

} properly to ensure reliable transaction processing.

Txn is

Atomicity: "All or Nothing"

If any part of txn fails the entire transaction is rolled back. & no changes are applied.

Txn
Account A → Acc B
100 80
→ sent 100 80 (CN of updated) → Rolled back

Consistency: follows rules

brings DB from one valid state to another valid state

eg: -ve balance (Rule Violation)
Sum of total remains same

Isolation: Concurrent Txns are isolated from each other.

T1 | T2 | T3
R(A) | R(B) | R(A).

Customer withdrawing money from same account, isolation ensures that both the txn don't see each other.

// schedule to [Serial schedule] → It is always consistent.

Durability: All changes in DB should be permanent even if system fail

replication

[SQL]

BASE Properties

NoSQL

BA → Basically Available

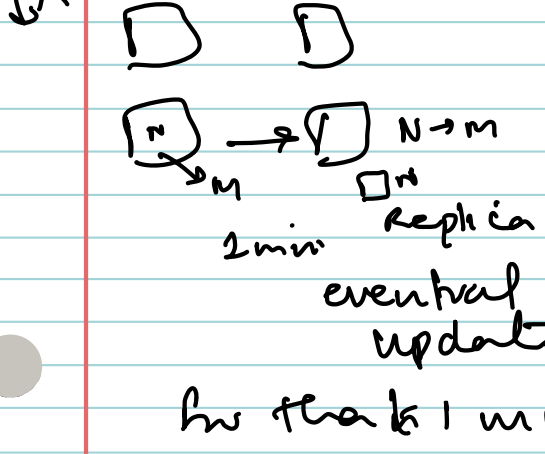
S → Soft State

E → Eventually consistent

Basically Available: System Guarantees availability.
DB will respond to all requests even if data sent is stale or incomplete

eg:
Inventory
Item - sold out
not updated
on website
during high
traffic

Soft state: Even w/o Input from outside, the state of system changes.
Due to the eventual updates across distributed system



Eventually consistent : System will eventually become consistent given enough time.

For update it will make time (replicas) But eventually become consistent-