# CAP Theorem - DATABASE

# What is Network Shared System ?

Sometimes the Network Shared System referred System Shared Resources, Network sharing enables access to information by more than one person through more than one device at the same or at different times.

resource : <https://www.techopedia.com/>

But in this kind of Network Shared System We need the reliable and efficient and it can be achieved by CAP Theorem.

The main Three Property of CAP :

1. Consistency

2. Availability

3. Partition Tolerance

Developers must carefully balance these attributes according to their particular application demands because of this underlying restriction. Designers may decide which qualities to prioritize to obtain the best performance and reliability for their systems by knowing the CAP theorem.

The CAP theorem is a fundamental concept in distributed systems theory that was first proposed by Eric Brewer in 2000 and subsequently shown by Seth Gilbert and Nancy Lynch in 2002.

Let’s Explore One by One Element of CAP Theorem.

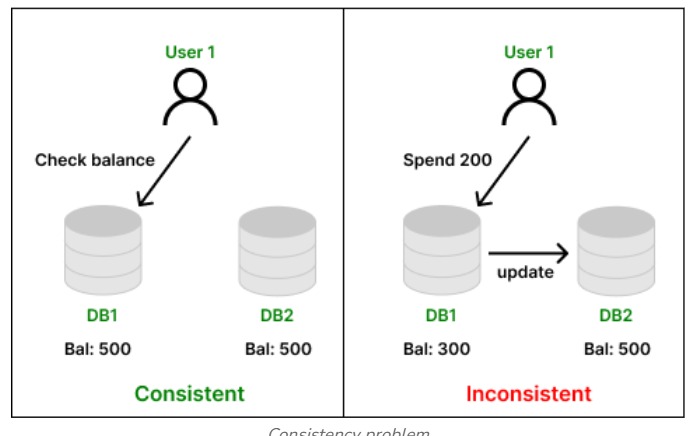
## 1. Consistency

Consistency means that all the nodes (databases) inside a network will have the same copies of a replicated data item visible for various transactions. It guarantees that every node in a distributed cluster returns the same, most recent, and successful write. It refers to every client having the same view of the data. There are various types of consistency models. Consistency in CAP refers to sequential consistency, a very strong form of consistency.

Note that the concept of Consistency in ACID and CAP are slightly different since in CAP, it refers to the consistency of the values in different copies of the same data item in a replicated distributed system.

In ACID, it refers to the fact that a transaction will not violate the integrity constraints specified on the database schema.

**For example**, a user checks his account balance and knows that he has 500 rupees. He spends 200 rupees on some products. Hence the amount of 200 must be deducted changing his account balance to 300 rupees. This change must be committed and communicated with all other databases that hold this user's details. Otherwise, there will be inconsistency, and the other database might show his account balance as 500 rupees which is not true.



## 2. Availability

Availability means that each read or write request for a data item will either be processed successfully or will receive a message that the operation cannot be completed. Every non-failing node returns a response for all the read and write requests in a reasonable amount of time. The key word here is "every". In simple terms, every node (on either side of a network partition) must be able to respond in a reasonable amount of time.

**Example :**

Let’s say we’ve got a big-time creator, User A, who has a huge fanbase—1000+ people subscribed to his content.

Now enters User B, chillin' in a faraway city, wanting to subscribe to User A’s channel. But here’s the twist:

User A is connected to one server node (maybe in Mumbai).

User B is connected to a different server node (say in Delhi).

These two servers are part of a distributed system, meaning data is stored and shared across multiple servers.

But oh no—there's a network partition! That means Mumbai and Delhi servers can’t talk to each other properly right now—like there’s a traffic jam on the data highway.

If the system prioritizes availability, then:

Even though the servers can’t sync up instantly,

User B will still be able to subscribe to User A.

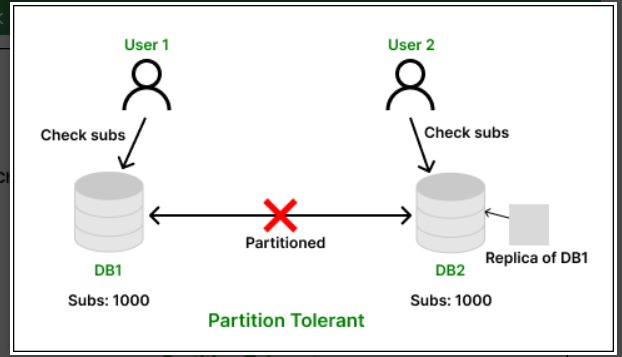
The Delhi node (where User B is connected) will accept the subscription request and reply quickly.

It might not check with Mumbai right away to confirm everything is in sync, but it won’t keep User B waiting. That’s the promise of availability: every request gets a response.

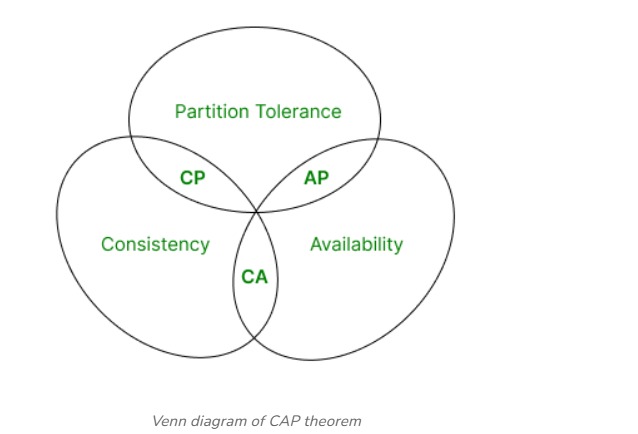
## 3. Partition Tolerance :

Partition tolerance means that the system can continue operating even if the network connecting the nodes has a fault that results in two or more partitions, where the nodes in each partition can only communicate among each other. That means, the system continues to function and upholds its consistency guarantees in spite of network partitions. Network partitions are a fact of life. Distributed systems guaranteeing partition tolerance can gracefully recover from partitions once the partition heals.

**For example**, take the example of the same social media network where two users are trying to find the subscriber count of a particular channel. Due to some technical fault, there occurs a network outage, the second database connected by user B losses its connection with first database. Hence the subscriber count is shown to the user B with the help of replica of data which was previously stored in database 1 backed up prior to network outage. Hence the distributed system is partition tolerant.



**The Combination using this Diagram :**

****

**1. CA (Consistency and Availability)**

**These types of system always accept the request to view or modify the data sent by the user and they are always responded with data which is consistent among all the database nodes of a big, distributed network.**

**2 . AP (Availability and Partition Tolerance)**

**These types of system are distributed in nature, ensuring that the request sent by the user to view or modify the data present in the database nodes are not dropped and are processed in presence of a network partition.**

**3. CP (Consistency and Partition Tolerance)**

**These types of system are distributed in nature, ensuring that the request sent by the user to view or modify the data present in the database nodes are dropped instead of responding with inconsistent data in presence of a network partition.**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_