

14. CAP Theorem with Proper Explanation

C - [Consistency]

for server with error



| OrderID | CustomerName | Book |
|---------|--------------|----------------|
| 123 | A | B ₁ |
| 134 | B | B ₂ |

L.B.

Request for orderID

Server2



134 goes to server2

Not Found

(Bad Customer Experience)

| OrderID | CustomerName | Book |
|---------|--------------|----------------|
| 131 | C | B ₃ |
| 141 | D | B ₄ |

Hence both servers

need to maintain

consistency among them

business logic

| OrderID | CustomerName | Book |
|---------|--------------|----------------|
| 123 | A | B ₁ |
| 134 | B | B ₂ |
| 131 | C | B ₃ |
| 141 | D | B ₄ |

L.B.

Data copied amongst servers asynchronously

async replication might not provide strong consistency

| | | |
|-----|---|----------------|
| 123 | A | B ₁ |
| 134 | B | B ₂ |
| 131 | C | B ₃ |
| 141 | D | B ₄ |

A → Availability

If Server 2 goes down, it does not mean that we stop serving traffic. It means that some other server or Server 1 itself will serve the load itself.

And when server 2 comes back online, it syncs the data (latest) from the server 1.

P → Partition Tolerance

Let's say server 2 is not responding well. Communication between server 1 and server 2 is broken. Then server 1 has to check and reestablish connection with server 2.

So when server 1 is trying to handle partition tolerance, the system's availability is compromised.

CAP theorem says that we cannot have C, A and P all available at the same time.

If I am willing to compromise on consistency,

I can have the system to be Available and Partition tolerant.

~~C~~ \rightarrow A & P

~~If I am willing to~~

If system is always available,

the either it will be not consistent enough,

or it will have to sacrifice Partition tolerance.