BIG DATA ANALYTICS (2CE/IT709)

VENUS R. PATEL

UVPCE , GANPAT UNIVERSITY

"The goal is to turn data into information, and information into insight."

– Carly Fiorina, former CEO, Hewlett-Packard Co.

Session Outline :

- Concept of NOSQL
- Terminologies Related to NOSQL
- Features
- Database Types and Examples
- Available platforms
- Advantages
- Comparison with SQL
- Some Facts

NOSQL

What it is ?

NoSQL stands for Not Only SQL.

These are non-relational, open source, distributed databases.

They are hugely popular today owing to their ability to scale horizontally.

It has adeptness at dealing with a rich variety of data: structured, semi-structured and unstructured data.

Features of NOSQL:

- **Non-relational:**

They do not adhere to relational data model. In fact, they are either key–value pairs or document-oriented or column-oriented or graph-based databases.

Features of NOSQL:

- **Are distributed:**

They are distributed meaning the data is distributed across several nodes in a cluster constituted of low-cost commodity hardware.

Features of NOSQL:

- **Offer no support for ACID properties (Atomicity, Consistency, Isolation, and Durability):**
- They do not offer support for ACID properties of transactions.
- On the contrary, they have adherence to Brewer's CAP (Consistency, Availability, and Partition tolerance) theorem.

Features of NOSQL:

Provide no fixed table schema:

- NoSQL databases are becoming increasing popular owing to their support for flexibility to the schema.
- They do not mandate for the data to strictly adhere to any schema structure at the time of storage.

Types of NoSQL Databases

- **1)Key–value:** It maintains a big hash table of keys and values.
- Sample Key–Value Pair in Key–Value Database.
- Ex-

Key:	**Value:**
First Name	Simmonds
Last Name	David

Types of NoSQL Databases

- **2) Document:** It maintains data in collections constituted of documents.
- Sample Document in Document Database…..

{

"Book Name": "Fundamentals of Business Analytics",

"Publisher": "Wiley India",

"Year of Publication": "2011"

}

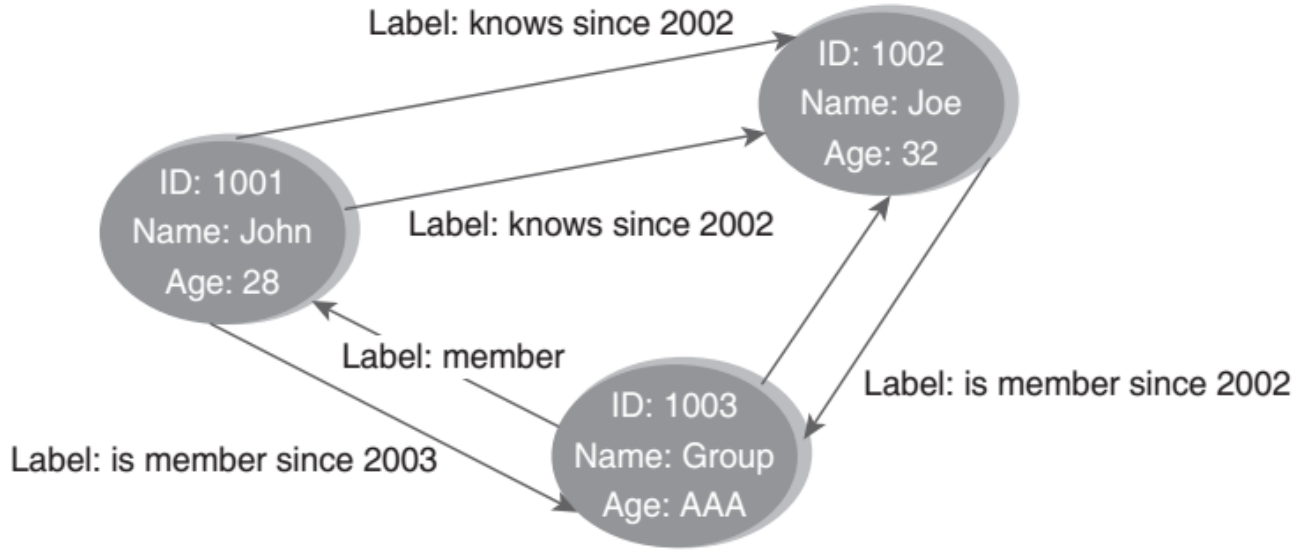
Types of NoSQL Databases

- **3) Column:** Each storage block has data from only one column.

Source: *Seema Acharya, Subhashini Chellappan. Big Data and Analytics, 2end Edition.*

Types of NoSQL Databases

- **4) Graph:** They are also called network database. A graph stores data in nodes.



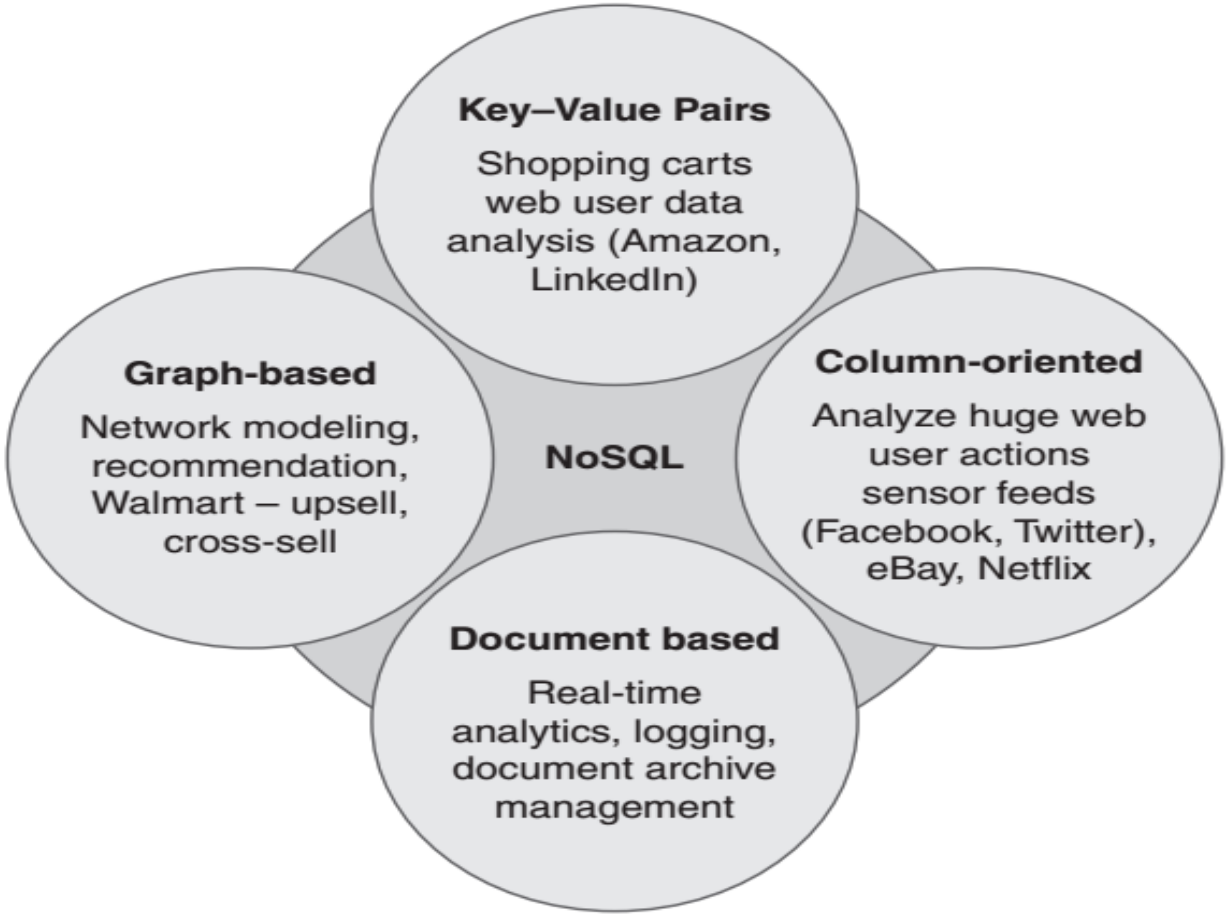
Source*: *Seema Acharya, Subhashini Chellappan. Big Data and Analytics, 2end Edition.

Popular schema-less databases

Key–Value Data Store	Column-Oriented Data Store	Document Data Store	Graph Data Store
• Riak	• Cassandra	• MongoDB	• InfiniteGraph
• Redis	• HBase	• CouchDB	• Neo4j
• Membase	• HyperTable	• RavenDB	• AllegroGraph

***Source**: Seema Acharya, Subhashini Chellappan. Big Data and Analytics, 2end Edition.*

Use of NoSQL in industry :



Source*: *Seema Acharya, Subhashini Chellappan. Big Data and Analytics, 2end Edition.

Few popular NoSQL vendors :

Company	Product	Most Widely Used by
Amazon	DynamoDB	LinkedIn, Mozilla
Facebook	Cassandra	Netflix, Twitter, eBay
Google	BigTable	Adobe Photoshop

Source*: *Seema Acharya, Subhashini Chellappan. Big Data and Analytics, 2end Edition.

Advantages :

- Can easily scale up and down: NoSQL database supports scaling rapidly and elastically and even allows to scale to the cloud.

(a) Cluster scale: It allows distribution of database across 100+ nodes often in multiple data centers.

(b) Performance scale: It sustains over 100,000+ database reads and writes per second.

(c) Data scale: It supports housing of 1 billion+ documents in the database.

Advantages :

- **Cheap, easy to implement:** Deploying NoSQL properly allows for all of the benefits of scale, high availability, fault tolerance, etc. while also lowering operational costs.

- **Relaxes the data consistency requirement:** NoSQL databases have adherence to CAP theorem (Consistency, Availability, and Partition tolerance). Most of the NoSQL databases compromise on consistency in favor of availability and partition tolerance. However, they do go for eventual consistency.

Advantages :

Data can be replicated to multiple nodes and can be partitioned:

- **Sharding**
- **Replication**

Sharding :

- Sharding is when different pieces of data are distributed across multiple servers.
- NoSQL databases support auto-sharding; this means that they can natively and automatically spread data across an arbitrary number of servers, without requiring the application to even be aware of the composition of the server pool.
- Servers can be added or removed from the data layer without application downtime.

Sharding :

- This would mean that data and query load are automatically balanced across servers.
- When a server goes down, it can be quickly and transparently replaced with no application disruption.

Replication:

- Replication is when multiple copies of data are stored across the cluster and even across data centers.
- This promises high availability and fault tolerance.

What does NoSQL Not Provide?

- Joins
- Group by
- ACID transactions
- Integration with applications that are based on SQL

SQL vs. NOSQL :

SQL

- Relational database
- Relational model Model-less approach
- Pre-defined schema
- Table based databases

NOSQL

- Non-relational,distributed database
- Model-less approach
- Dynamic schema for unstructured data
- Document-based or graph-based or wide column store or key–value pairs databases

SQL vs. NOSQL :

SQL

- Vertically scalable (by increasing system resources)
- Uses SQL
- Not preferred for large datasets
- Not a best fit for hierarchical data
- Emphasis on ACID properties

NOSQL

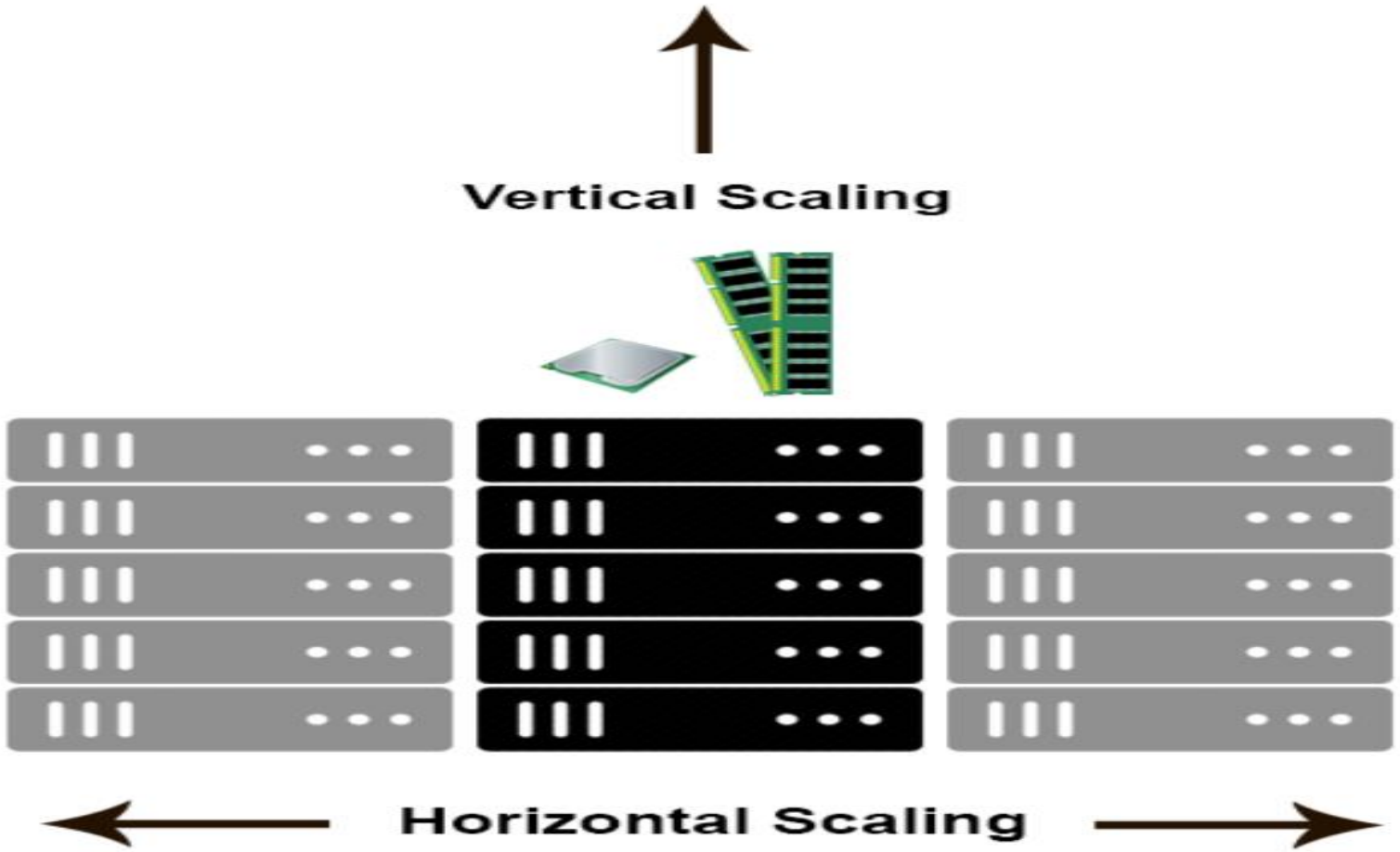
- Horizontally scalable (by creating a cluster of commodity machines)
- Uses UnQL (Unstructured Query Language)
- Largely preferred for large datasets
- Best fit for hierarchical storage as it follows the key–value pair of storing data similar to JSON (Java Script Object Notation)
- Follows Brewer's CAP theorem

Vertical vs Horizontal Scaling :

Horizontal scaling means that you scale by adding more machines into your pool of resources.

Vertical scaling means that you scale by adding more power (CPU, RAM) to an existing machine.

Think of a machine on a server rack, we add more machines across the **horizontal** direction and add more resources to a machine in the **vertical** direction.



Source: Google Images

Vertical vs Horizontal Scaling :

- With horizontal-scaling it is often easier to scale dynamically by adding more machines into the existing pool.
- Vertical-scaling is often limited to the capacity of a single machine, scaling beyond that capacity often involves downtime and comes with an upper limit.

NOSQL

Outline :

- ACID Properties
- CAP Theorem
- BASE Approach

- Polling Q & A

ACID Properties in SQL Transactions

Atomicity

- Atomicity of transaction means all operations in the transaction must complete, and if interrupted, then must be undone (rolled back).

- Example: If a customer withdraws an amount then the bank in first operation enters the withdrawn amount in the table and in the next operation modifies the balance with new amount available. Atomicity means both should be completed, else undone if interrupted in between.

Consistency

- Consistency in transactions means that a transaction must maintain the integrity constraint, and follow the consistency principle.

- Example, The difference of sum of deposited amounts and withdrawn amounts in a bank account must equal the last balance. All three data need to be consistent.

Isolation

- Isolation of transactions means two transactions of the database must be isolated from each other and done separately.

- Example : User A withdraws $100 and user B withdraws $250 from user Z's account, which has a balance of $1,000. Since both A and B draw from Z's account, one of the users is required to wait until the other user transaction is completed, avoiding inconsistent data.

  If B is required to wait, then B must wait until A's transaction is completed, and Z's account balance changes to $900. Now, B can withdraw $250 from this $900 balance.

Durability

- Durability means a transaction must persist once completed.
- Which guarantees that transactions that have committed will survive permanently.
- Example : If a flight booking reports that a seat has successfully been booked, then the seat will remain booked even if the system crashes.

CAP Theorem

Theorem:

- It states that in a distributed computing environment (a collection of interconnected nodes that share data), it is impossible to provide the following guarantees.
- At best you can have two of the following three – one must be sacrificed.

1. C = **Consistency**
2. A= **Availability**
3. P= **Partition Tolerance**

Consistency

- A system is said to be consistent if all nodes see the same data at the same time.

- **Example :**

If we perform a read operation on a consistent system, it should return the value of the most recent write operation. This means that, the read should cause all nodes to return the same data, i.e., the value of the most recent write.

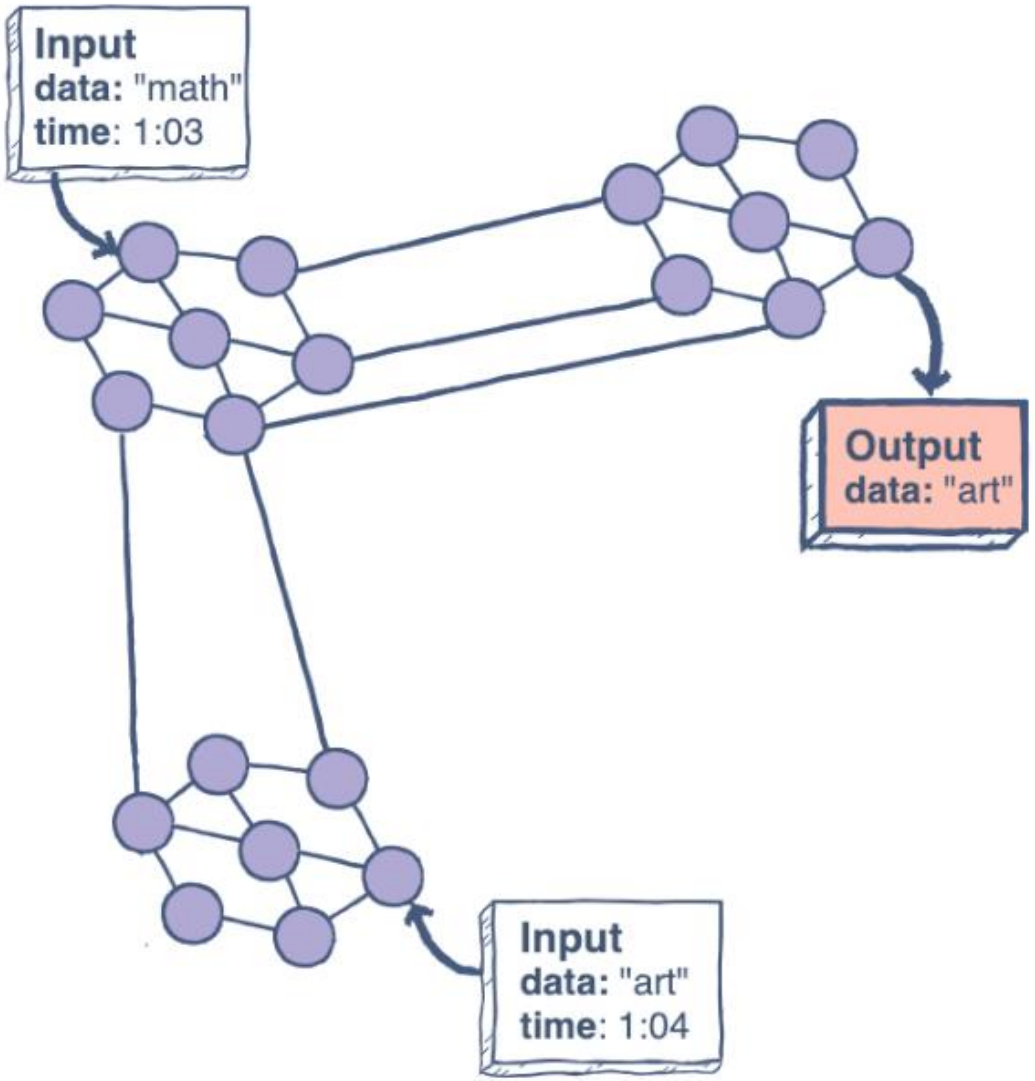


Fig. 1
Source : Google Images

Availability

- Availability in a distributed system ensures that the system remains operational 100% of the time.
- That during the transactions, the field values must be available in other partitions of the database so that each request receives a response on success as well as failure.

Partition Tolerance

- This condition states that the system does not fail, regardless of if messages are dropped or delayed between nodes in a system.

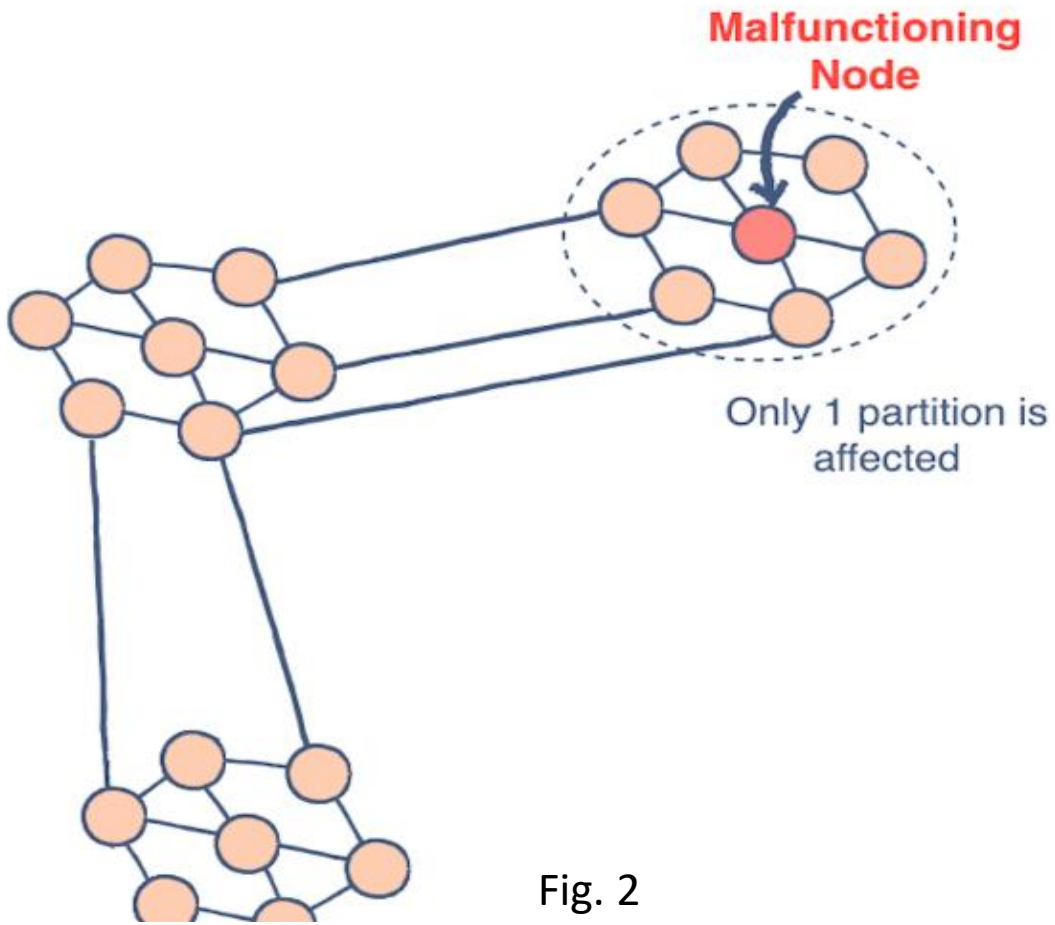


Fig. 2
Source : Google Images

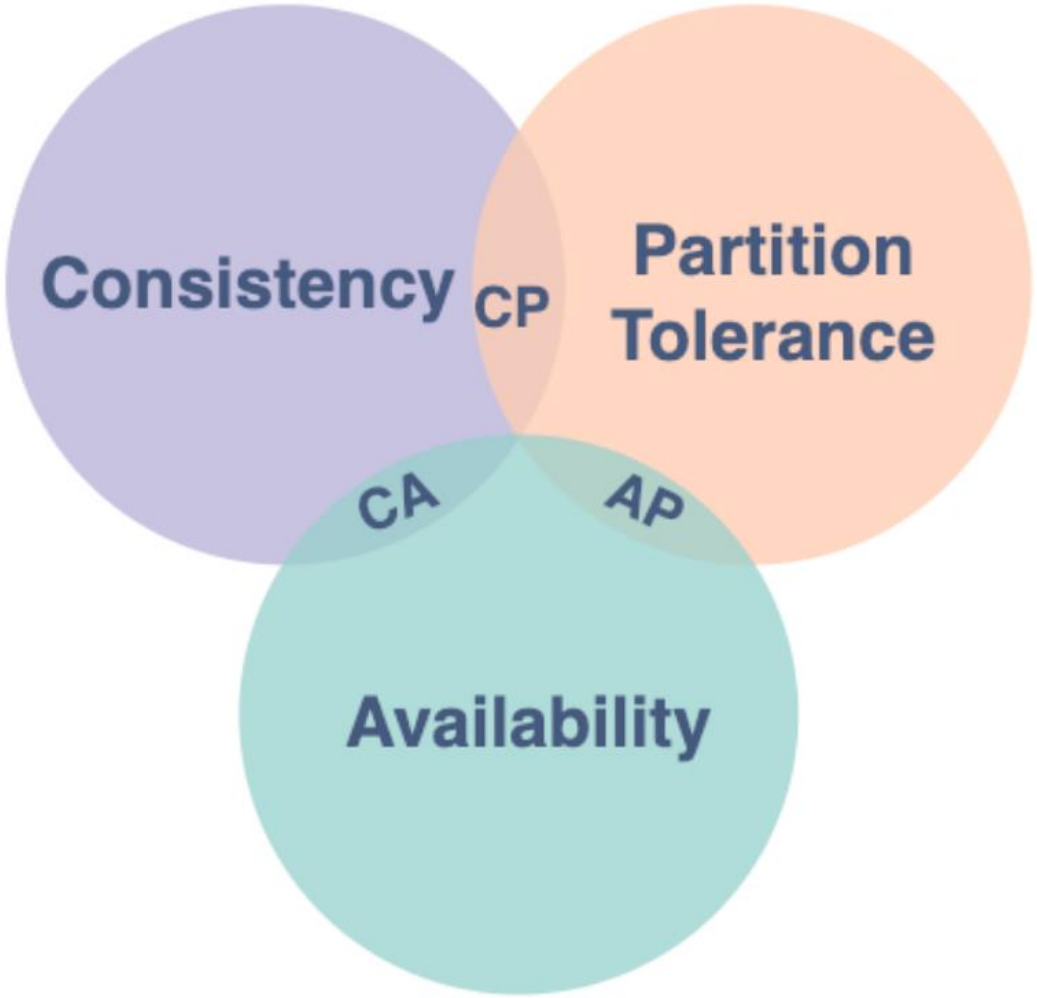


Fig. 3
Source : Google Images

Brewer’s CAP (Consistency, Availability and Partition Tolerance) theorem demonstrates that any distributed system **cannot** guarantee C, A and P together.

Partition tolerance :

- Partition tolerance cannot be overlooked for achieving reliability in a distributed database system.

- In case of any network failure, a choice can be:
  1. Database must answer, and that answer would be old or wrong data **(AP).**
  2. Database should not answer, unless it receives the latest copy of the data **(CP).**

CAP Theorem in Big Data Solution :

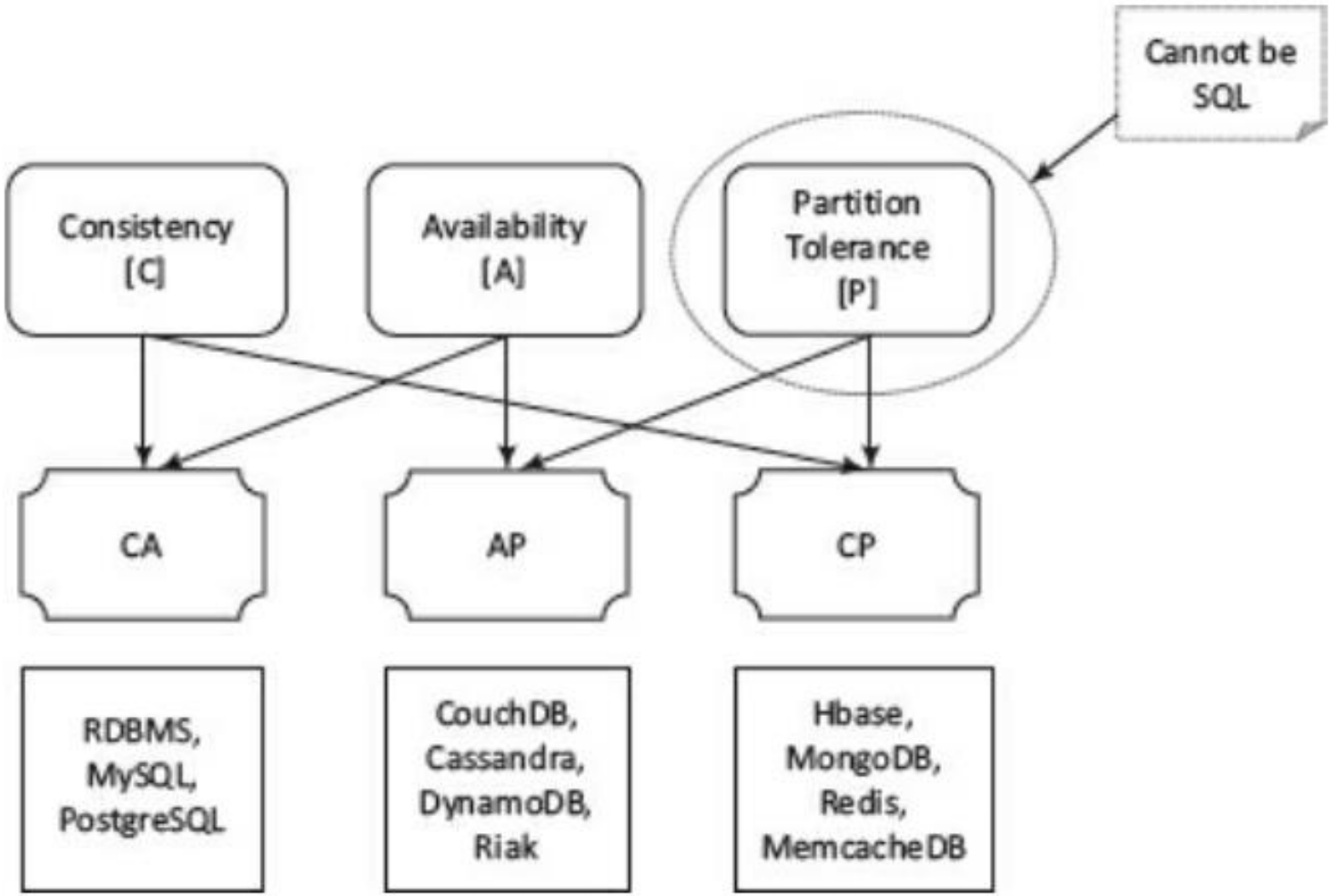
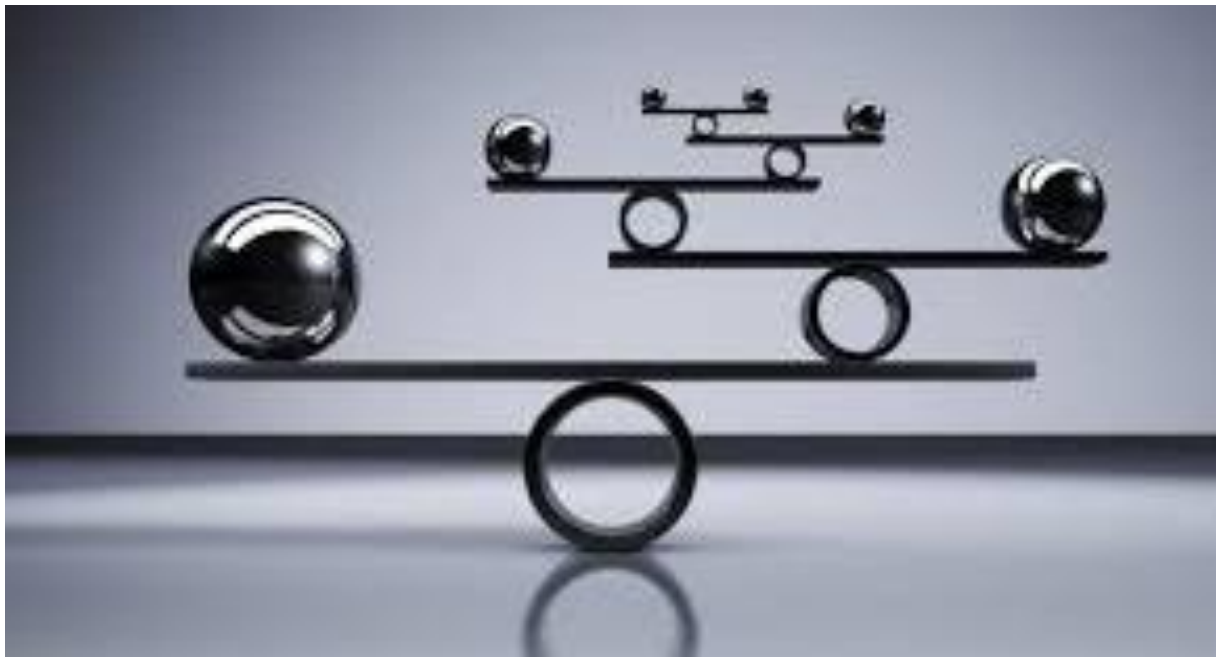


Fig. 4
Source : ***Seema Acharya, Subhashini Chellappan. Big Data and Analytics, 2end Edition***



Think and Connect If You can !

Image Source :Google Images

CAP :

- **Consistency:** All nodes observe the same data at the same time.
- **Availability:** Each request receives a response on success/failure.
- **Partition Tolerance:** The system continues to operate as a whole even in case of message loss, node failure or node not reachable.

Basic availability :

- Basic availability ensures by distribution of shards (many partitions of huge data store) across many data nodes with a high degree of replication. Then, a segment failure does not necessarily mean a complete data store unavailability.

Soft state :

- Soft state ensures processing even in the presence of inconsistencies but achieving consistency eventually. A program suitably takes into account the inconsistency found during processing. NoSQL database design does not consider the need of consistency all along the processing time.

Eventual consistency :

- Eventual consistency means consistency requirement in NoSQL databases meeting at some point of time in future.

- Data converges eventually to a consistent state with no time-frame specification for achieving that. ACID rules require consistency all along the processing on completion of each transaction. BASE does not have that requirement and has the flexibility.

BASE Approach :

- BA = Basic Availbilty
- S = Soft State
- E = Eventual Consistency

- ACID rules require consistency all along the processing on completion of each transaction.

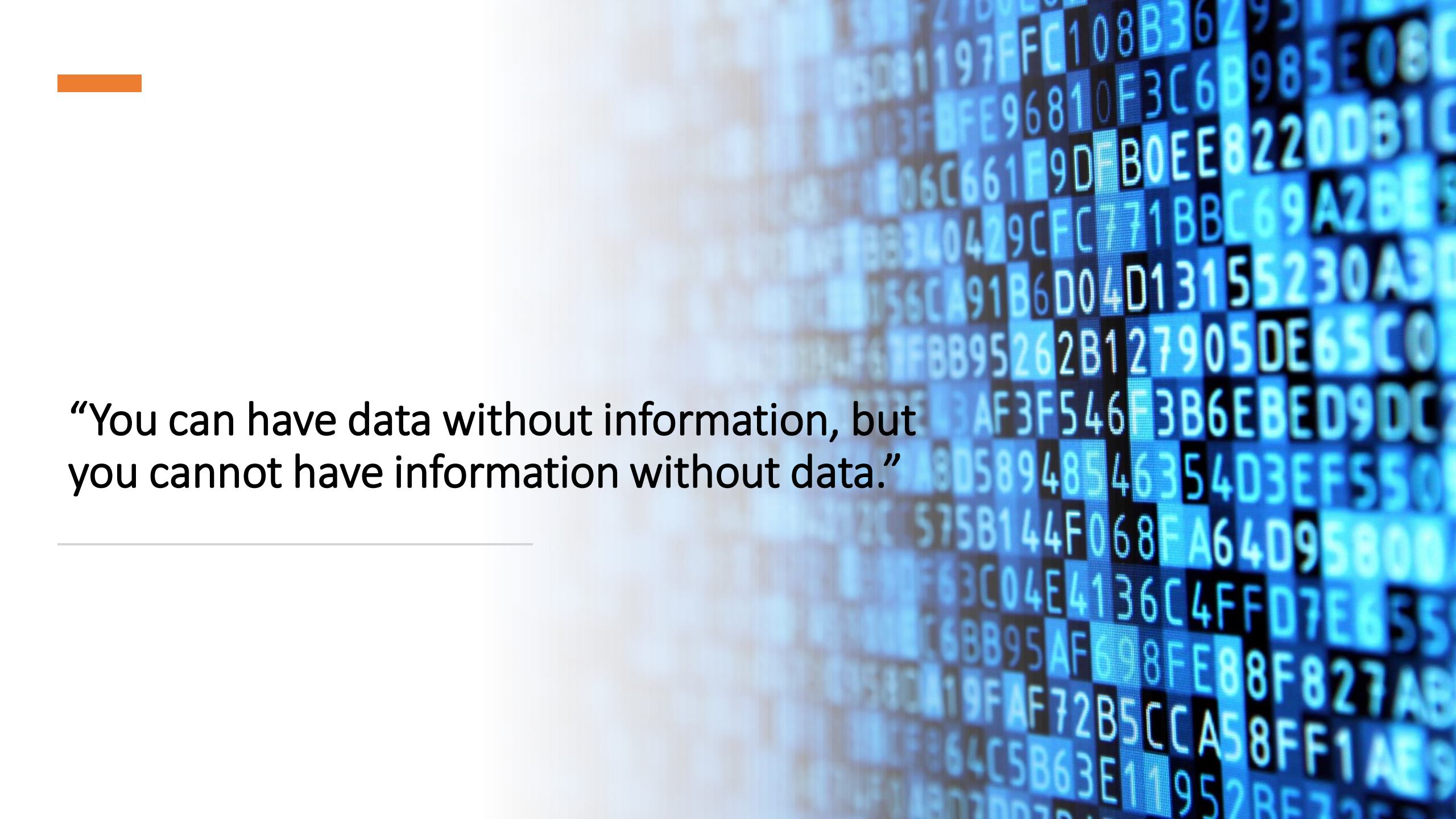
- BASE does not have that requirement and has the flexibility.

BASE :

- How is it achieved?

Assume a given data item. If no new updates are made to this given data item for a stipulated period of time, eventually all accesses to this data item will return the updated value. In other words, if no new updates are made to a given data item for a stipulated period of time, all updates that were made in the past and not applied to this given data item and the several replicas of it will percolate to this data item so that it stays as current/recent as is possible.

.



“You can have data without information, but you cannot have information without data.”

“In God we trust, all others must bring data.”

– W. Edwards Deming

BIG DATA HANDLING

1)SQL
2)NOSQL
3) ???

What is __???_?___ and how is it different from SQL and NoSQL?

Think :

What is that we love about NoSQL and is not there with our traditional RDBMS ?

&

What is that we love about SQL that NoSQL does not have support for?

Answer :

We need a database that has the **same scalable performance of** NoSQL systems for On Line Transaction Processing (OLTP) while still maintaining **the ACID guarantees of a traditional database.**

“New modern RDBMS is called **NewSQL.**”

Comparison Table :

	SQL	NoSQL	NewSQL
Adherence to ACID properties	Yes	No	Yes
OLTP/OLAP	Yes	No	Yes
Schema rigidity	Yes	No	Maybe
Adherence to data model	Adherence to relational model		
Data Format Flexibility	No	Yes	Maybe
Scalability	Scale up Vertical Scaling	Scale out Horizontal Scaling	Scale out
Distributed Computing	Yes	Yes	Yes
Community Support	Huge	Growing	Slowly gro

Big Data Risks

- Large volume and velocity of Big Data provide greater insights but also associate risks with the data used.

- Data included may be **erroneous, less accurate or far from reality**. Analytics introduces new errors due to such data.

Big Data Risks

- Big Data can cause potential harm to individuals.

- **For example :** When someone puts false or distorted data about an individual in a blog, Facebook post, WhatsApp groups or tweets, the individual may suffer loss of educational opportunity, job or credit for his/her urgent needs. A company may suffer financial losses.

Big Data Risks

- Five data risks, are

1) Data security,
2) Data privacy breach,
3) Costs affecting profits,
4) Bad analytics and
5) Bad data.

Characteristics of Big Data NoSQL solution

- **High and easy scalability:** NoSQL data stores are designed to expand horizontally. Horizontal scaling means that scaling out by adding more machines as data nodes (servers) into the pool of resources (processing, memory, network connections). The design scales out using multi-utility cloud services.
- **Support to replication:** Multiple copies of data store across multiple nodes of a cluster. This ensures high availability, partition, reliability and fault tolerance.
- **Distributable**: Big Data solutions permit sharding and distributing of shards on multiple clusters which enhances performance and throughput.

Characteristics of Big Data NoSQL solution

- **Usages of NoSQL servers** :which are less expensive. NoSQL data stores require less management efforts. It supports many features like automatic repair, easier data distribution and simpler data models that makes database administrator (DBA) and tuning requirements less stringent.

Characteristics of Big Data NoSQL solution

- **Usages of open-source tools:** NoSQL data stores are cheap and open source. Database implementation is easy and typically uses cheap servers to manage the exploding data and transaction while RDBMS databases are expensive and use big servers and storage systems. So, cost per gigabyte data store and processing of that data can be many times less than the cost of RDBMS.

Characteristics of Big Data NoSQL solution

- **Support to schema-less data model:** NoSQL data store is schema less, so data can be inserted in a NoSQL data store without any predefined schema. So, the format or data model can be changed any time, without disruption of application. Managing the changes is a difficult problem in SQL.

Characteristics of Big Data NoSQL solution

- **Support to integrated caching:** NoSQL data store support the caching in system memory. That increases output performance. SQL database needs a separate infrastructure for that.

- **No inflexibility unlike the SQL/RDBMS :** NoSQL DBs are flexible (not rigid) and have no structured way of storing and manipulating data. SQL stores in the form of tables consisting of rows and columns. NoSQL data stores have flexibility in following ACID rules.