NoSQL

▼ What is RDBMS

DBMS is a software for storing and retrieving user data while considering appropriate security measures.

- A relational database is a type of database that stores and provides access to data points that are related to one another.
- Relational databases are based on the relational model, an intuitive, straightforward way of representing data in tables.
- In a relational database, each row in the table is a record with a unique ID called the key.
- The columns of the table hold attributes of the data, and each record usually has a value for each attribute, making it easy to establish the relationships among data points.
- Example : MySQL, Postgress

▼ Drawbacks of RDBMS

1. **Cost:** The underlaying cost involved in a relational database is quite expensive. For
   setting up a relational database, there must be separate software which needs to be
   purchased.
2. **Performance:** Always the performance of the relational database depends on the number of tables. If there are more number of tables, the response given to the queries
   will be slower.
3. **Physical Storage:** A relational database also requires tremendous amount of physical memory since it is with rows and columns. Each of the operations

depend on
separate physical storage.

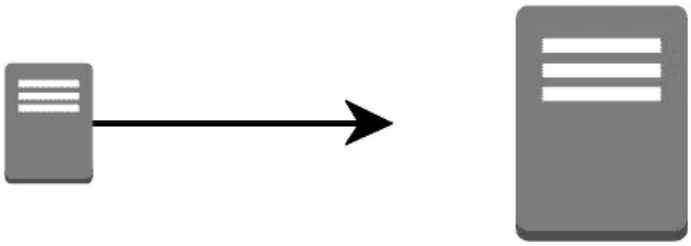
4. **Complexity**: Although a relational database is free from complex structuring, occasionally it may become complex too. When the amount of data in a relational
database increases, it eventually makes the system more complicated.
5. **Information Loss:** Large organizations tends to use more number of database systems with more tables. These information can be used to be transferred from one system to another. This could pose a risk of data loss.

▼ What is DataBase Scaling?

- **Scalability** is the ability to expand or contract the capacity of system resources in order to support the changing usage of your application. This can refer both to increasing and decreasing usage of the application.
- A system is said to be scalable if it can increase its workload and throughput when additional resources are added.
- There are two broad categories for scaling database systems: vertical scaling and horizontal scaling.

  ▼ Vertical Scaling

  - **Vertical scaling** refers to increasing the processing power of a single server or cluster.
  - **Vertical scaling,** also known as scaling up, is the process of adding resources, such as memory or more powerful CPUs to an existing server.
  - Removing memory or changing to a less powerful CPU is known as scaling down.
  - Both relational and non-relational databases can scale up, but eventually, there will be a limit in terms of maximum processing power and throughput.
  - Additionally, there are increased costs with high-performance hardware, as costs do not scale linearly.



▼ Horizontal Scaling

- **Horizontal scaling,** sometimes referred to as scaling out, is the process of adding more hardware to a system.
- This typically means adding nodes (new servers) to an existing system.
- Doing the opposite, that is removing hardware, is known as scaling in.
- This is difficult with relational databases due to the difficulty in spreading out related data across nodes.
- With non-relational databases, this is made simpler since collections are self-contained and not coupled relationally.
- This allows them to be distributed across nodes more simply, as queries do not have to “join” them together across nodes.
- Scaling MongoDB horizontally is achieved through sharding (preferred).

▼ What is Database Sharding?

- The word “**Shard**” means “**a small part of a whole”.**
- Hence Sharding means dividing a larger part into smaller parts.

- In DBMS, Sharding is a type of DataBase partitioning in which a large DataBase is divided or partitioned into smaller data and different nodes.
- It is a method for distributing a single dataset across multiple databases, which can then be stored on multiple machines.
- This allows for larger datasets to be split into smaller chunks and stored in multiple data nodes, increasing the total storage capacity of the system.
- Similarly, by distributing the data across multiple machines, a sharded database can handle more requests than a single machine can.
- Sharding is a form of scaling known as **horizontal scaling** or **scale-out**, as additional nodes are brought on to share the load.
- These shards are not only smaller, but also faster and hence easily manageable.

▼ Define NoSQL

- not only SQL
- non tabular DB’s and store data differently than relational tables
- provide flexible schemas and scale easily with large amounts of data
- they break the traditional mindset of storing data at a single location
- instead, it distributes and stores data over a set of multiple servers to distribute the load on the DB tier

▼ Advantages of NoSQL Databases

▼ Flexible Scalability

NoSQL databases are highly scalable and can be modified to meet the unique scaling needs of your business.

They can be scaled horizontally as opposed to vertically, which provides a clear advantage over SQL databases.

Instead of scaling up by adding more servers, NoSQL databases can scale out by using commodity hardware. This has the ability to support increased

traffic in
order to meet demand with zero downtime.

▼ Flexible data types

NoSQL databases allow you to store and retrieve data with only limited or no requirements for the predefined schema.

This means that your application can quickly adapt as new types of information are added without adjusting table structures, modifying indexes, and so forth.

▼ Data Storage

Many NoSQL databases can handle extensive datasets, making them ideal for big data applications, IoT (Internet of Things), and other real-time analytics.

For example, BigTable is a distributed storage system built to store structured data in the cloud at a scale that allows easy addition and deletion with no downtime or interruption.

By storing this type of information on BigTable, it's possible to quickly retrieve it while providing access control support across all levels down to individual fields within your documents without requiring additional security layers.

Additionally, you don't have any single points of failure, which means there is no limit to how much data it can hold.

▼ Simple

Many NoSQL database management systems require only a few lines of code, which is ideal for developers who want to get started quickly.

For example, MongoDB is a NoSQL database management system that allows developers to store data in flexible structures and retrieve it with code written in the language of their choice.

▼ Low maintenance

NoSQL databases don't require the same level of ongoing database administration as traditional relational databases because they can automatically partition and replicate information across nodes.

his means you won't need additional IT infrastructure, software licenses, or expensive hardware like standard SQL-relational databases, which can be a significant financial burden for your business.

▼ Disadvantages of NoSQL Databases

Lack of Standardization: There is no standard that defines rules and roles of NoSQL databases. The design and query languages of NoSQL databases vary widely between different NoSQL products – much more widely than they do among traditional SQL databases.

(ii) Backup of Database: Backups are a drawback in NoSQL databases. Though some NoSQL databases like MongoDB provide some tools for backup, these tools are not mature enough to ensure proper complete data backup solution.

(iii) Consistency: NoSQL puts a scalability and performance first but when it comes to
a consistency of the data NoSQL doesn’t take much consideration so it makes it little
insecure as compared to the relational database e.g., in NoSQL databases if you enter
same set of data again, it will take it without issuing any error whereas relational databases ensure that no duplicate rows get entry in databases.

Less mature: Since SQL has been around a lot longer, it is generally universal and more mature.One of the most significant advantages is that SQL has endless online support, documentation, and tutorials.With NoSQL, it may be more challenging to find
solutions to problems.

▼ Features of NoSQL

▼ Non-Relational

- Relations in a database establish connections between tables of data

- With a NoSQL database, this information is stored as an aggregate — a single record with everything about the transaction, including the delivery address.

▼ Schema Flexibility

A database schema is the description of all possible data and data structures in a relational database. With a NoSQL database, a schema isn't required, giving you the freedom to store information without doing up-front schema design.

▼ Commodity hardware

Some databases are designed to operate best (or only) with specialized storage and processing hardware. With a NoSQL database, cheap off-the-shelf servers can be used. Adding more of these cheap servers allows NoSQL databases to scale to handle more data.

▼ Highly distributable

Distributed databases can store and process a set of information on more than one device. With a NoSQL database, a cluster of servers can be used to hold a single large database.

▼ Horizontal Scaling

As NoSQL databases are highly distributable, it is easier to scale them horizontally as there are no relations between the tables. So a dataset is sharded and stored into multiple nodes in different computers

▼ Types of NoSQL Databases

▼ Key-value pair based

Redis, dynamo, riak

▼ Column-oriented graph

Cassandra, hypertable

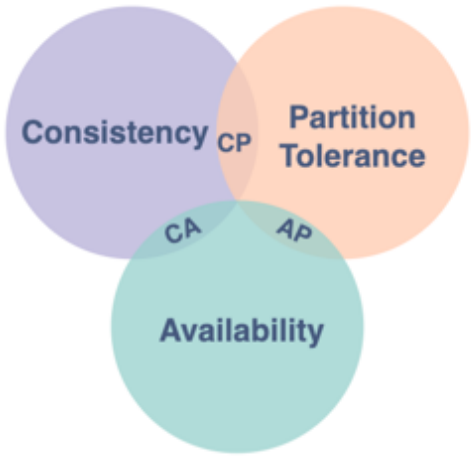
- ▼ graphs based

  Neo4j, oracle, flock

- ▼ Document-oriented

  Mongo, couch

▼ CAP Theorem



The three letters in CAP refer to three desirable properties of distributed systems with replicated data:

consistency (among replicated copies),

availability (of the system for read and write operations) and

partition tolerance (in the face of the nodes in the system being partitioned by a network fault).

The CAP theorem states that it is not possible to guarantee all three of the desirable properties – consistency, availability, and partition tolerance at the same time in a distributed system with data replication.

The theorem states that networked shared-data systems can only strongly support two of the following three properties:

Consistency :

1. Consistency means that the nodes will have the same copies of a replicated data item visible for various transactions.

2. A guarantee that every node in a distributed cluster returns the same, most recent and a successful write.
3. Consistency refers to every client having the same view of the data.
4. There are various types of consistency models.
5. Consistency in CAP refers to **sequential consistency**, a very strong form of consistency.

Availability :

1. Availability in a distributed system ensures that the system remains operational 100% of the time.
2. 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.
3. Every non-failing node returns a response for all the read and write requests in a reasonable amount of time.

Partial Tolerance :

1. 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.
2. That means, the system continues to function and upholds its consistency guarantees in spite of network partitions.
3. Partition tolerance has become more of a necessity than an option in distributed
systems. It is made possible by sufficiently replicating records across combinations of
nodes and networks

▼ BASE properties

NoSQL relies upon a softer model known, appropriately, as the BASE model. This model accommodates the flexibility offered by NoSQL and similar approaches to the

management and curation of unstructured data. BASE consists of three principles:

Basic Availability :

1. The NoSQL database approach focuses on the availability of data even in the presence of multiple failures.
2. It achieves this by using a highly distributed approach to database management.
3. Instead of maintaining a single large data store and focusing on the fault tolerance of that store, NoSQL databases spread data across many storage systems with a high degree of replication.

Soft State :

1. Due to the lack of immediate consistency, the data values may change over time.
2. The BASE model breaks off with the concept of a database that obligates its own consistency, delegating that responsibility to developers.

Eventually Consistent :

1. The fact that BASE does not obligates immediate consistency but it does not mean that it never achieves it.
2. The system will eventually become consistent once it stops receiving input.
3. However, until it does, the data reads are still possible (even though they might not reflect reality).

▼ ACID v/s BASE

Data consistency models

ACID V/S BASE

Criteria	ACID	BASE
Simplicity	Simple	Complex
Focus	Commits	Best attempt
Maintenance	High	Low

Criteria	ACID	BASE
Consistency Of Data	Strong	Weak/Loose
Concurrency scheme	Nested Transactions	Close to answer
Scaling	Vertical (limited)	Horizontal (unlimited)
Implementation	Easy to implement	Difficult to implement
Upgrade	Harder to upgrade	Easy to upgrade
Type of database	Robust	Simple
Type of code	Simple	Harder
Time required for completion	Less time	More time.
Examples	Oracle, MySQL, SQL Server, etc.	DynamoDB, Cassandra, CouchDB, SimpleDB, etc.

▼ Features of MongoDB

MongoDB is a scalable, flexible NoSQL document database platform designed to overcome the relational databases approach.

1. supports adhoc queries
2. indexing - you can index any field in a document
3. replication - it supports master slave replication.

   A master can perform Reads and Writes and a Slave copies data from the master and can only be used for reads or back up (not writes)
4. Data duplication - mongodb can run over multiple servers. this data is duplicated to keep the system up and running incase of any hardware failure
5. load balancing - automatic load balancing because of data placed in shards
6. supports map reduce and aggregation tools
7. it is a schema less database written in C++
8. provides high performance
9. stores files of any size without complicating your stack
10. supports JSON datamodel with dynamic schemas
11. also supports auto sharding for horizontal scalability

Personalization, fraud and error prevention, preemptive maintenance, performance optimization

References

CQLSH Commands