Modern Education Society's
College of Engineering, Pune
19, Late Prin. V. K. Joag Path, Wadia College Campus, Pune-411001
Affiliated to Savitribai Phule Pune University & Approved by AICTE, New Delhi.

Unit V

NoSQL Databases

Prof. S. B. Shinde

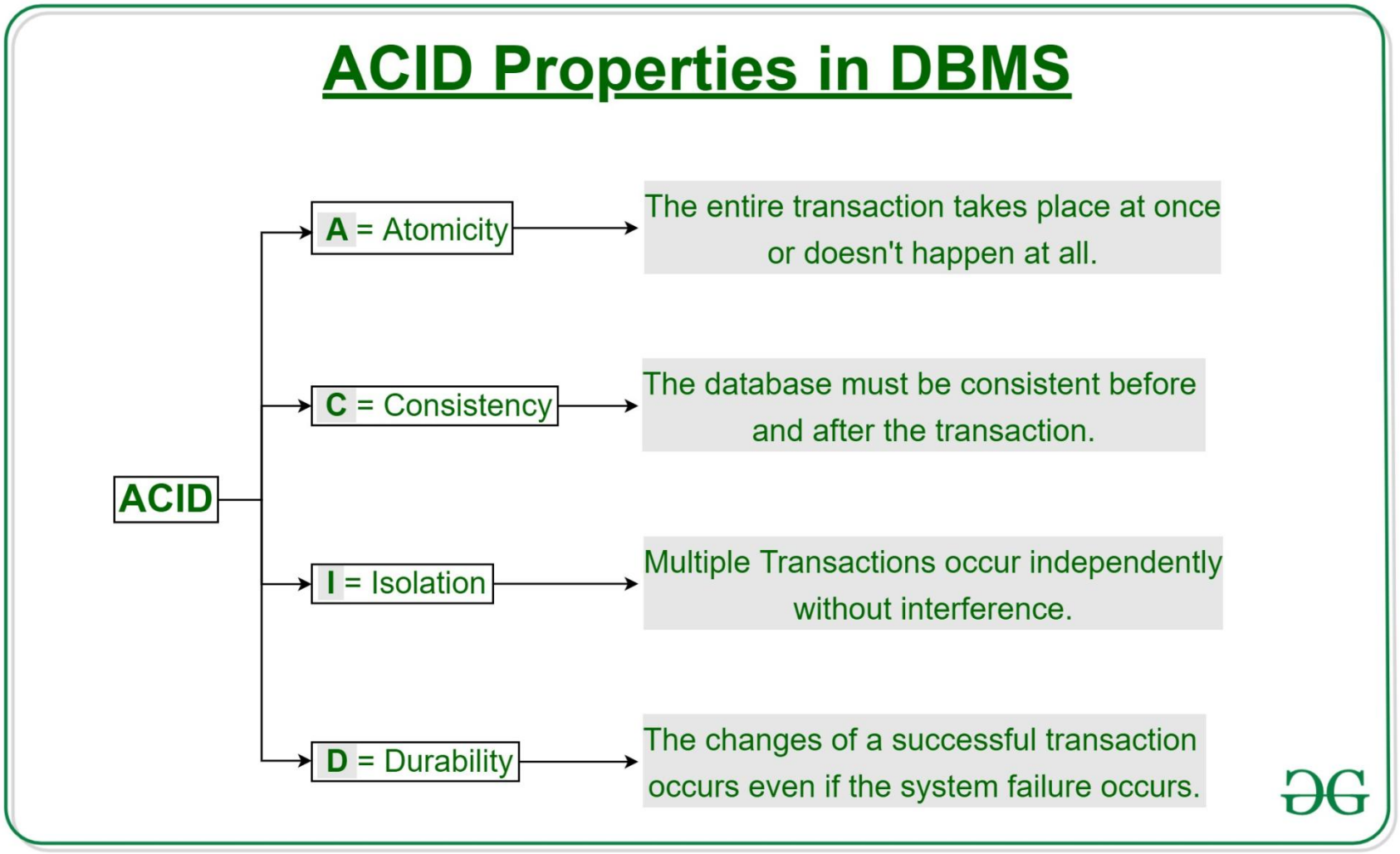
Asst Professor, MESCOE Pune

Introduction

- In the computing system (web and business applications), there are enormous data that comes out every day from the web. A large section of these data is handled by Relational database management systems (RDBMS).

- **Classical relation database follow the ACID Rules:**
  - A database transaction must be atomic, consistent, isolated and durable.

Introduction



Distributed Systems

- A distributed system consists of multiple computers and software components that communicate through a computer network (a local network or by a wide area network).
- A distributed system can consist of any number of possible configurations, such as mainframes, workstations, personal computers, and so on.
- The computers interact with each other and share the resources of the system to achieve a common goal.

Distributed Computing

Advantages:

- Reliability (fault tolerance)
- Scalability
- Sharing of Resources
- Flexibility
- Speed
- Open System
- Performance

Disadvantages:

- Troubleshooting
- Less Software Support
- Network infrastructure
- Security

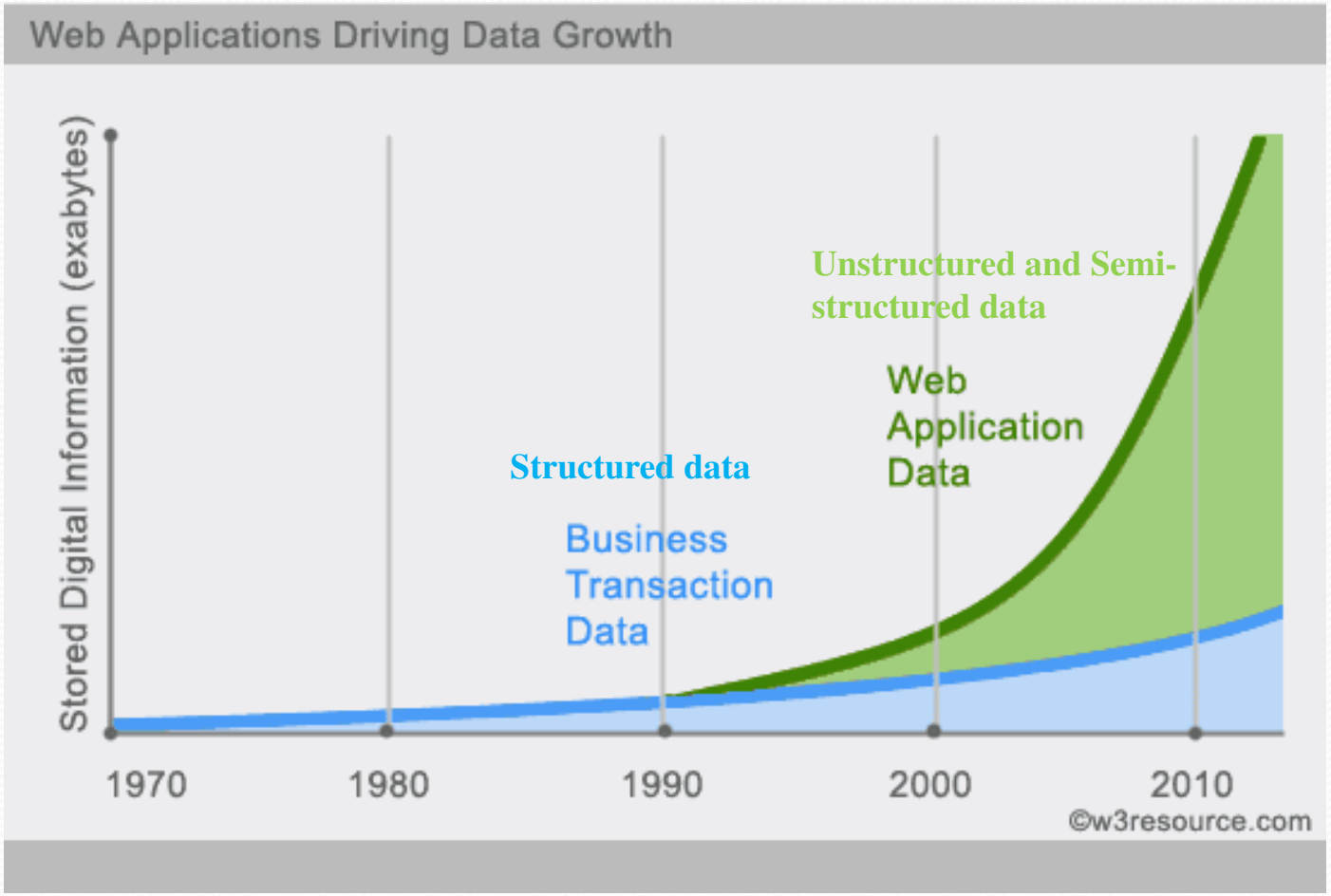
What is NoSQL?

- Stands for **Not Only SQL.**
- NoSQL is a non-relational database management systems.
- NoSQL database were developed in response to a rise in the volume of data stored about users, objects and products, the frequency in which this data is accessed, and performance and processing needs.
- It is designed for distributed data stores where very large scale of data storing needs (for example Google or Facebook which collects TB of data every day for their users).
- These type of data storing may not require fixed schema, avoid join operations and typically scale horizontally.

Why NoSQL ?

- In today's time data is becoming easier to access and capture through third parties such as Facebook, Google+ and others.
- Personal user information, social graphs, geo location data, user-generated content and machine logging data are just a few examples where the data has been increasing exponentially.
- To avail the above service properly, it is required to process huge amount of data. Which SQL databases were never designed.
- Instead of using structured tables to store multiple related attributes in a row, NoSQL databases use the concept of a key/value store.

Why NoSQL ?


Web Applications Driving Data Growth
Stored Digital Information (exabytes)
Unstructured and Semi-structured data
Web Application Data
Structured data
Business Transaction Data
1970
1980
1990
2000
2010
©w3resource.com

RDBMS vs NoSQL

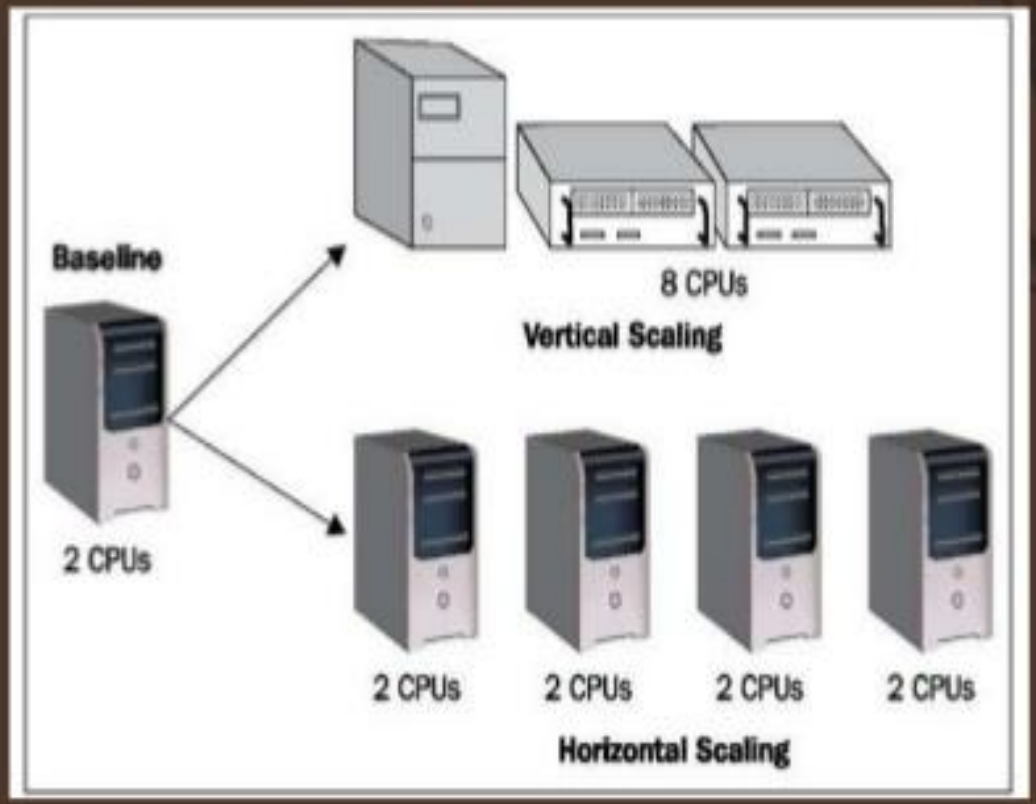
RDBMS

- Structured and organized data
- Structured query language (SQL)
- Data and its relationships are stored in separate tables
- DDL,DML
- Tight Consistency
- ACID Transaction

NoSQL

- Stands for Not Only SQL
- No declarative query language
- No predefined schema
- Key-Value pair storage, Column Store, Document Store, Graph databases
- Eventual consistency rather ACID property
- Unstructured and unpredictable data
- CAP Theorem
- Prioritizes high performance, high availability and scalability
- BASE Transaction

Horizontal Scaling Vs Vertical Scaling

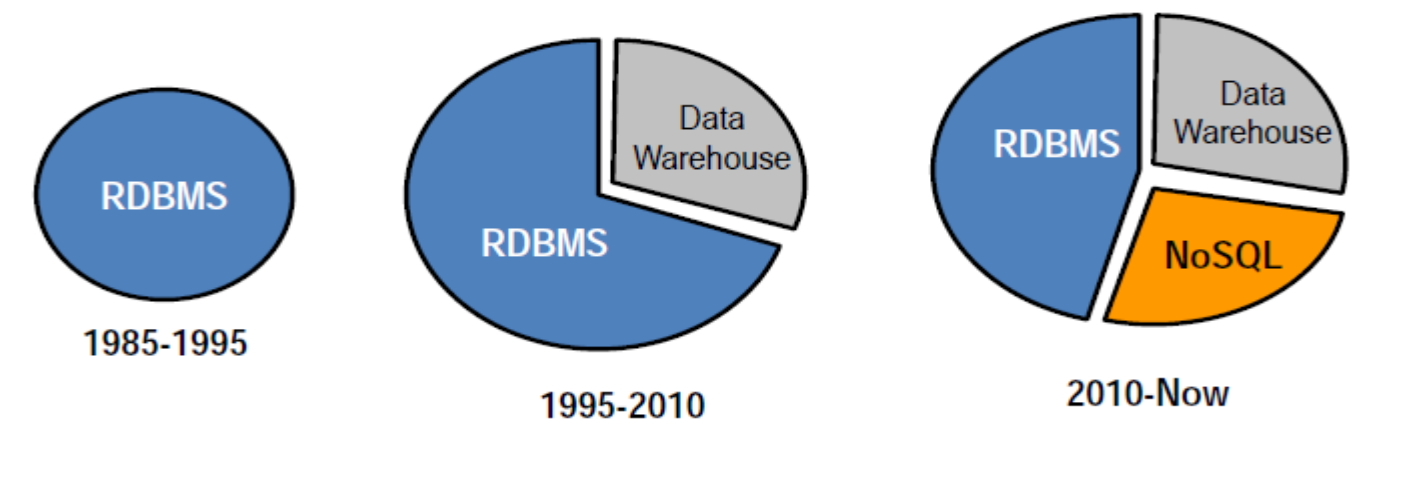


WEBONISE LAB

mongoDB

Brief History of NoSQL

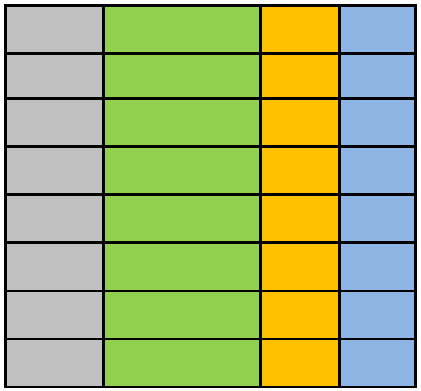
- The term NoSQL was coined by Carlo Strozzi in the year 1998. He used this term to name his Open Source, Light Weight, DataBase which did not have an SQL interface.
- **Three Eras of Databases**



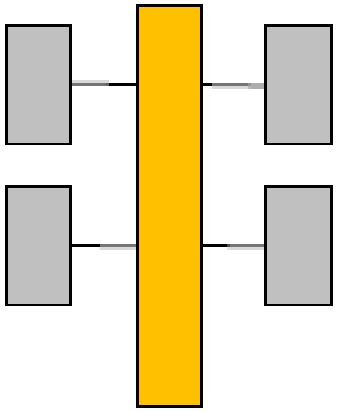
- RDBMS for transactions, Data Warehouse for analytics and NoSQL for …?

Before NoSQL

Relational

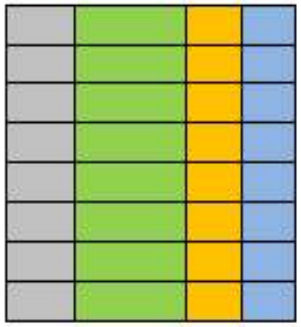


Analytical (OLAP)

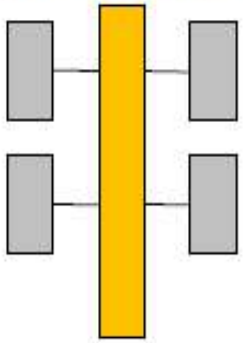


After NoSQL

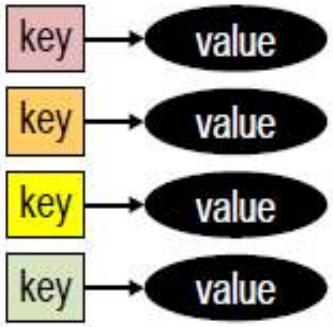
Relational



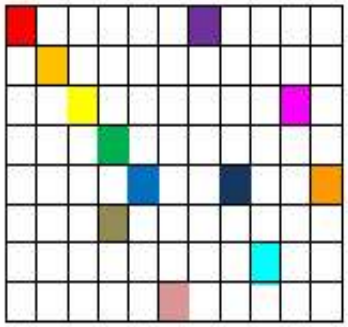
Analytical (OLAP)



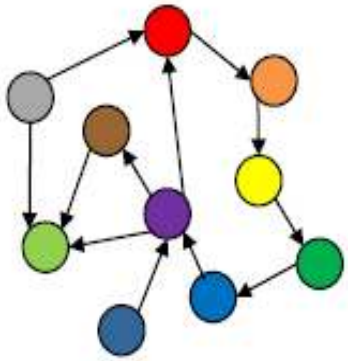
Key-Value



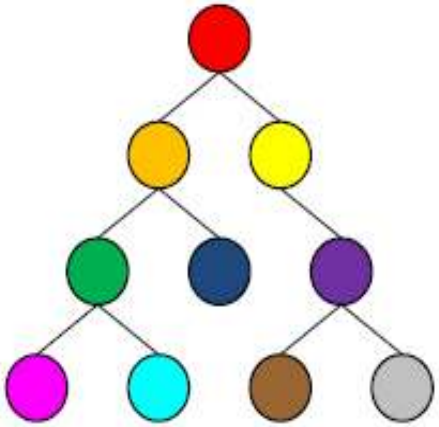
Column-Family



Graph



Document



Type of NoSQL

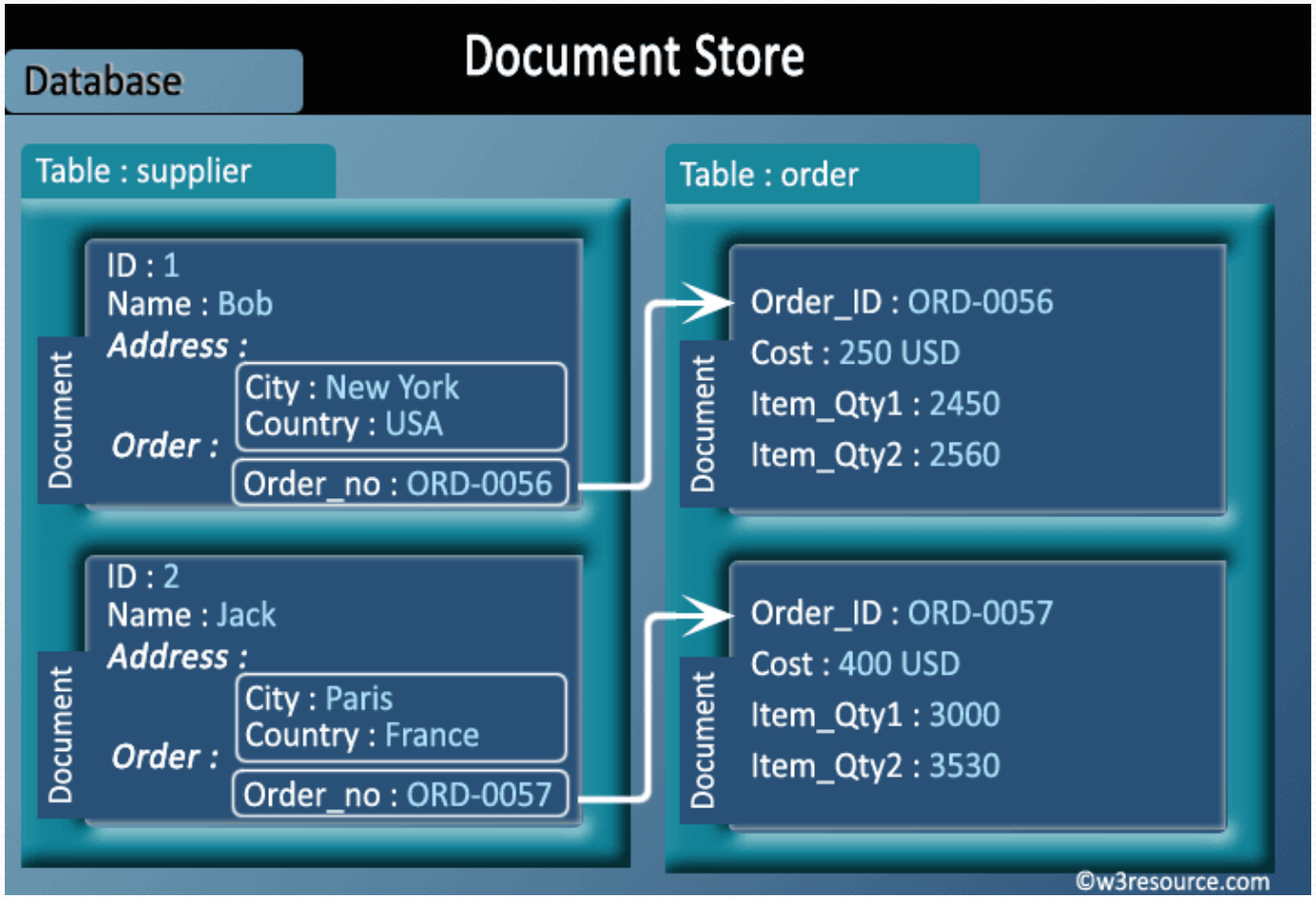
- **Document Oriented Databases:**
  - Document oriented database stores data in the form of documents.
  - A collection of documents.
  - A document can be in a JSON, BSON, XML, YAML, etc format.
  - Data in this model is stored inside documents.
  - A document is a key value collection where the key allows access to its value.
  - Documents are stored into collections in order to group different kinds of data.

Relational model	Document model
Tables	Collections
Rows	Documents
Columns	Key/value pairs
Joins	not available

  - Example: MongoDB, Elasticsearch, Couchbase Server, CouchDB, RethinkDB, Terrastore, MarkLogic Server etc.

Type of NoSQL

- **Document Oriented Databases:**



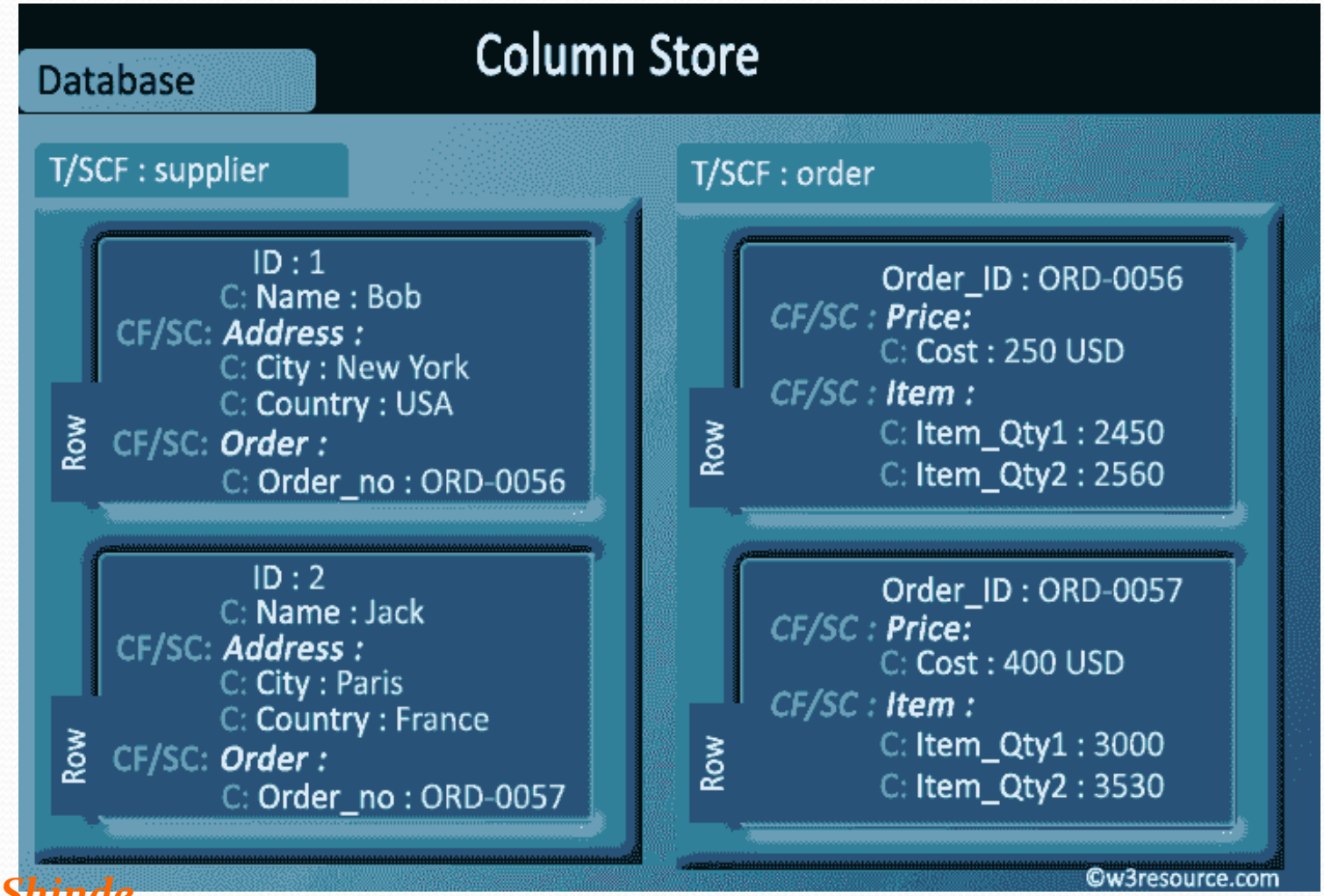
Type of NoSQL

- **Column Oriented Databases ::**
  - Column-oriented databases primarily work on columns and every column is treated individually.
  - Column stores can improve the performance of queries as it can access specific column data.
  - High performance on aggregation queries (e.g. COUNT, SUM, AVG, MIN, MAX).
  - Works on data warehouses and business intelligence, customer relationship management (CRM), Library card catalogs etc.

  - Example: Hadoop/Hbase, Cassandra, Amazon SimpleDB, HPCC, Cloudera etc.

Type of NoSQL

- **Column Oriented Databases**



Type of NoSQL

- **Key-Value Databases:**
  - In key-value database each item in the database is stored as an attribute name (or "key"), together with its value.
  - Key-value stores are most basic types of NoSQL databases.
  - Designed to handle huge amounts of data.
  - In the key-value storage, database stores data as hash table where each key is unique and the value can be string, JSON, BLOB (basic large object) etc.
  - Key-Value stores follows the 'Availability' and 'Partition' aspects of CAP theorem.
  - Key-Value stores can be used as collections, dictionaries, associative arrays etc.
  - Example: Redis, Riak, Azure Table Storage, DynamoDB, Berkeley DB, LevelDB, FoundationDB etc.

Type of NoSQL

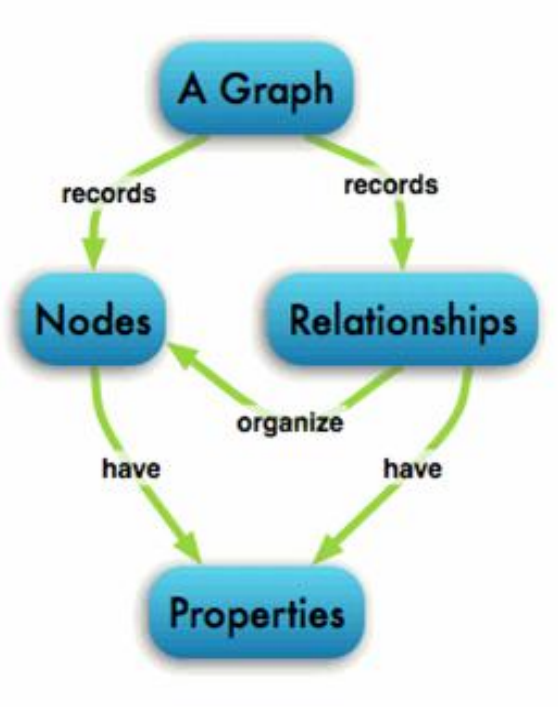
- **Key-Value Databases:**

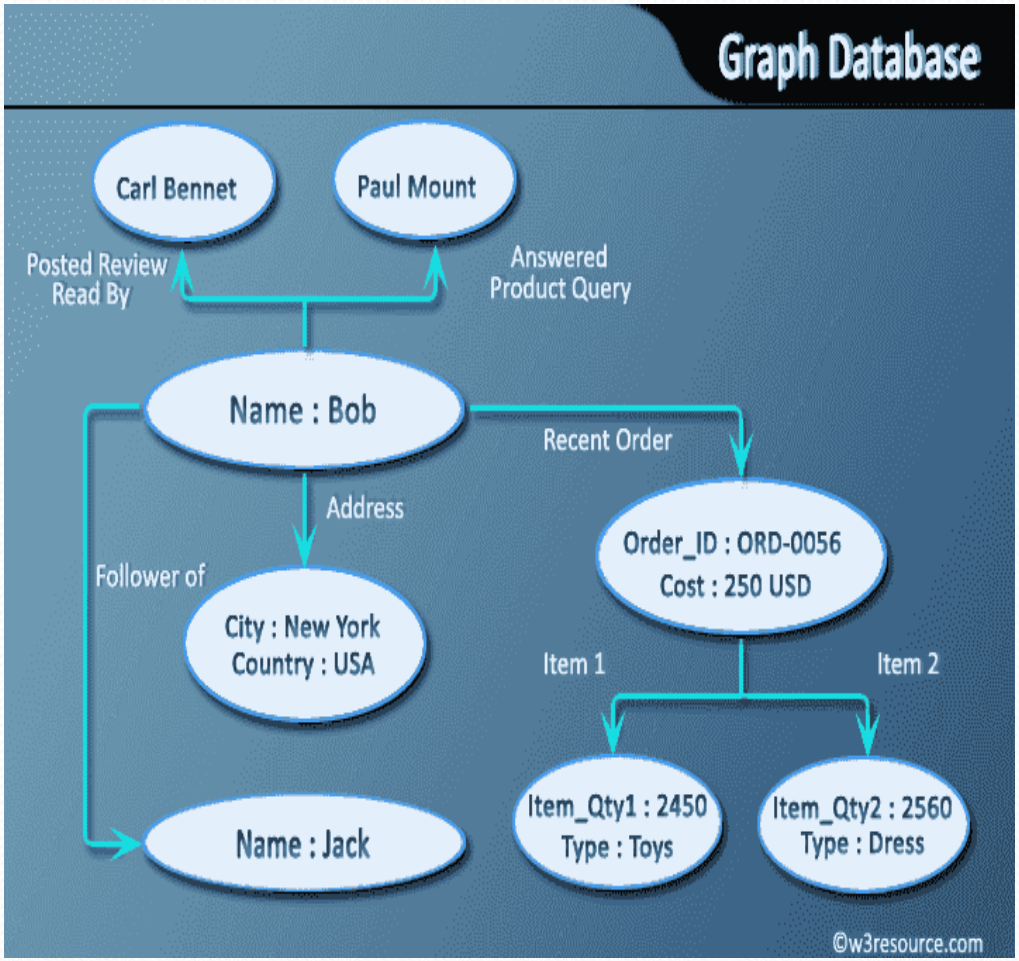
Type of NoSQL

- **Graph database:**
  - A graph database uses graph structures with nodes, edges, and properties to represent and store data in database.
  - A graph databases is faster for associative data sets and hence it's gaining popularity these days.
  - Graph stores are used to store information about networks, such as social connections.
  - Each node represents an entity (such as a student or business) and each edge represents a connection or relationship between two node
  - Example: HyperGraphDB, GraphBase Neo4J, WhiteDB, Infinite Graph, BrightstarDB etc.

Type of NoSQL

- **Graph database:**





SQL v/s NoSQL

	SQL Databases	NoSQL Databases
Types	One type (SQL database) with minor variations	Many different types including key-value stores, document databases, wide-column stores, and graph databases.
Development History	Developed in 1970s to deal with first wave of data storage applications.	Developed in 2000s to deal with limitations of SQL databases, particularly concerning scale, replication and unstructured data storage.
Examples	MySQL, Postgres, Oracle Database	MongoDB, Cassandra, HBase, Neo4j
Data Manipulation	Specific language using Select, Insert, and Update statements,	Through object-oriented APIs
Consistency	Can be configured for strong consistency	Depends on product. Some provide strong consistency (e.g., MongoDB) whereas others offer eventual consistency (e.g., Cassandra)

SQL v/s NoSQL

	SQL Databases	NoSQL Databases
Scaling	Vertically, meaning a single server must be made increasingly powerful in order to deal with increased demand.	Horizontally, meaning that to add capacity, a database administrator can simply add more commodity servers or cloud instances.
Development Model	Mix of open-source (e.g., Postgres, MySQL) and closed source (e.g., Oracle Database)	Open-source
Supports Transactions	Yes, updates can be configured to complete entirely or not at all	In certain circumstances and at certain levels (e.g., document level vs. database level)
Data Storage Model	Individual records are stored as rows in tables, with each column storing a specific piece of data about that record much like a spreadsheet.	Varies based on database type.
Schemas	Structure and data types are fixed in advance. To store information about a new data item, the entire database must be altered, during which time the database must be taken offline.	Typically dynamic. Records can add new information on the fly, and unlike SQL table rows, dissimilar data can be stored together as necessary.

CAP Theorem (Brewer's Theorem)

CAP Theorem states that there are three basic requirements which exist in a special relation when designing applications for a distributed architecture.

- **Consistency:**
- **Availability:**
- **Partition tolerance:**

CAP Theorem

Consistency:

- **Consistency:** (all nodes see the same data at the same time)

- **Consistency** - the **data in the database** remains consistent after the execution of an operation.

- For example **after** an **update** operation **all clients see the same data.**

CAP Theorem

- **Availability:**

- **Availability:** (a guarantee that every request receives a response about whether it was successful or failed)

- **Availability - the system is always on** (service guarantee availability), **no downtime**.

CAP Theorem

- **Partition Tolerance**

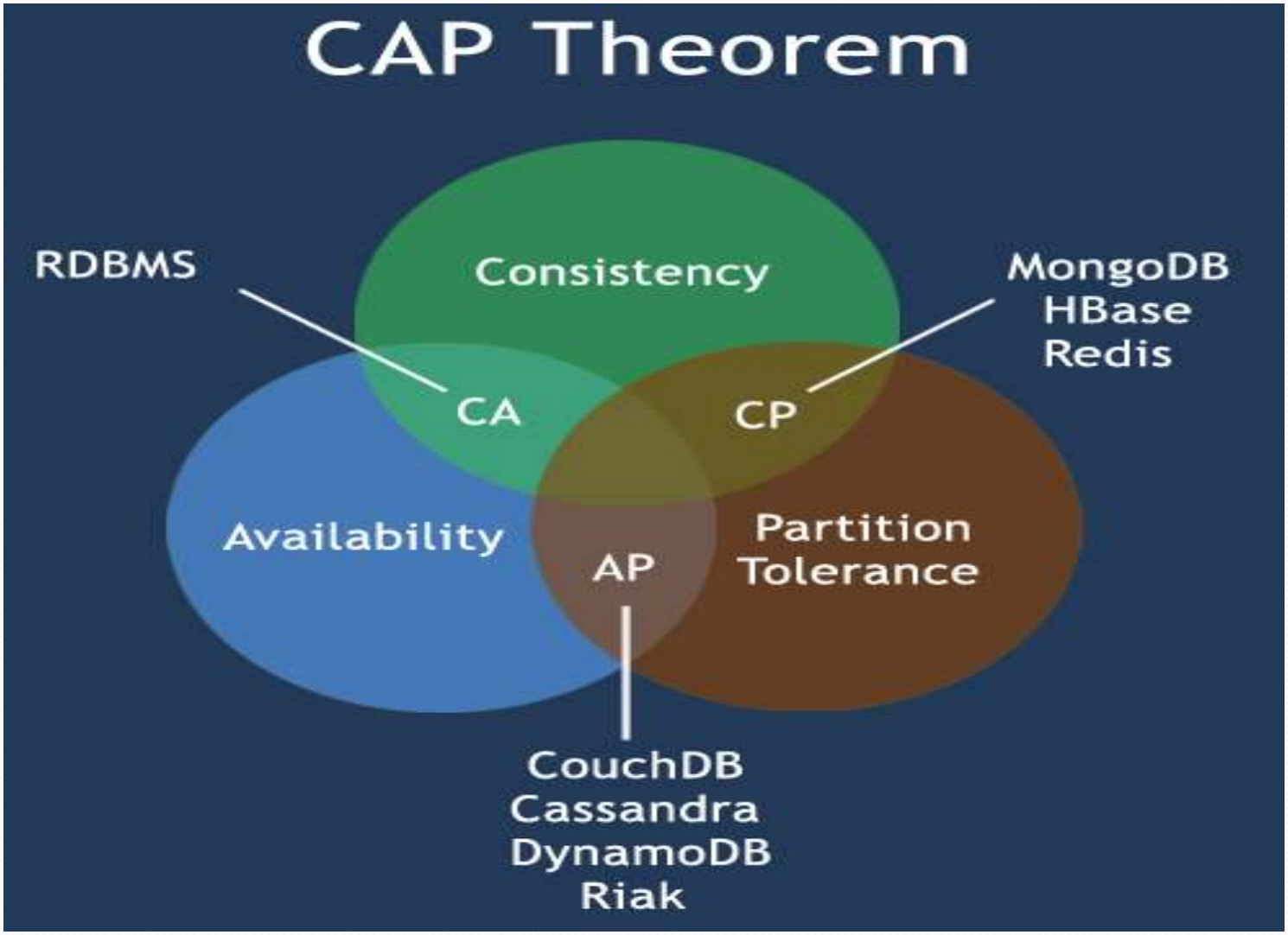
➢ **Partition Tolerance** (the system continues to operate despite arbitrary message loss or failure of part of the system)

➢ **Partition Tolerance - the system continues to function** even the communication among the **servers is unreliable**, i.e. the **servers** may be **partitioned into multiple groups** that cannot communicate with one another.

CAP Theorem

- Theoretically it is impossible to fulfill all 3 requirements.
- **A distributed system can support only 2 out of the 3 characteristics:**
  - **CA** - Single site cluster, therefore all nodes are always in contact. When a partition occurs, the system blocks.
  - **CP** - Some data may not be accessible, but the rest is still consistent/accurate.
  - **AP** - System is still available under partitioning, but some of the data returned may be **inaccurate**.

CAP Theorem


CAP Theorem
RDBMS
Consistency
MongoDB
HBase
Redis
CA
CP
Availability
Partition
Tolerance
AP
CouchDB
Cassandra
DynamoDB
Riak

The BASE System

- The BASE acronym was defined by Eric Brewer, who is also known for formulating the CAP theorem.
- The CAP theorem states that a distributed computer system cannot guarantee all of the following three properties at the same time:
  - **Consistency:**
  - **Availability:**
  - **Partition tolerance:**
- A BASE system gives up on consistency.

The BASE System Contd..

- Basically Available indicates that the system does guarantee availability, in terms of the CAP theorem.
- Soft state indicates that the state of the system may change over time, even without input. This is because of the eventual consistency model.
- Eventual consistency indicates that the system will become consistent over time, given that the system doesn't receive input during that time.

ACID	BASE
Atomic	Basically Available
Consistency	Soft state
Isolation	Eventual consistency
Durable	

NoSQL pros/cons

Advantages:

- High scalability
- Dynamic Schemas
- Replication
- Auto-sharding
- Integrated Caching
- Distributed Computing
- Low Cost
- No complicated Relationships

Disadvantages:

- Maturity
- Enterprise Support
- Transaction Support
- Expertise (Highly Skilled Programmers)

Advantages of NoSQL

- **Scalability:** NoSQL database can be scaled up easily and with minimum effort and hence it's well suited for today's every increasing database need (bit data). NoSQL database have scalable architecture, so it can efficiently manage data and can scale up to many machines instead of costly machines that are required while scaling using of SQL DBMS.
- With **dynamic schema,** if we want to change the length of column, or add new column we don't need to change whole table data instead the new data will be stored with the new structure without affecting the previous data /structure. In NoSQL databases we can insertion data without a predefined schema.
- **Replication** provides redundancy and increases data availability. With multiple copies of data on different database servers, replication protects a database from the loss of a single server.
- To use replication with sharding, deploy each shard as a replica set.
- **Sharding** is the process of storing data records across multiple machines and is MongoDB's approach to meeting the demands of data growth.
- Many NoSQL database have **integrated caching** mechanism, hence frequently used data are stored in system memory as much as possible and discarding the need for a separate caching layer.

Disadvantages of NoSQL

- **Maturity –** NoSQL database are new and emerging technologies. Since its under heavy development bugs, new features, keep on arising.
- **Enterprise Support –** If system fails company must be able to get timely support. In case of NoSQL, there are very few companies which know the technology and hence can be a deciding factor before using NOSQL.
- **Transaction Support –** NOSQL doesn't support SQL transaction features and hence for financial application SQL are still one the best in industry.
- **Expertise (Highly Skilled Programmers) –** There are millions of people around the world who knows RDBMS while very few people are aware of such technology hence getting a NOSQL programmer can be difficult.

Thank You