MongoDB Developer and Administrator Course

Introduction to mongodb

Anju Munoth

Objectives

By the end of this course, you will be able to:

- Explain the functionalities of MongoDB as document database
- Identify the benefits of MongoDB
- Explain the different use cases of MongoDB
- Explain how to create and manage different types of indexes in MongoDB for query execution
- Explain how the replication and sharding features in MongoDB help in scaling read and write operations
- Explain the process of developing Java using MongoDB

Course Overview

The course provides the following:

- Difference between NoSQL and RDBMS databases
- A detailed introduction of MongoDB as a document database
- Knowledge about MongoDB's role in the Big Data Ecosystem, Database Scaling, Replication, and Sharding
- Knowledge about CRUD operations in MongoDB
- The detailed process of developing Java and Node JS applications using MongoDB
- A detailed explanation of administering cluster operations in MongoDB
- Detailed steps of installing MongoDB in different operating systems and performing various functions

Value to Professionals and Organizations

This course is beneficial for professionals who administer, manage, and analyze large and complex data, such as:

Database Developers

- Schema design
- Developing Web applications in Java and Node JS
- Developing Mobile applications in Java and Node JS

Database Administrators

- Schema Design
- Performance optimization
- Manages security, availability, and consistency
- Monitors and manages cluster

Data Analysts

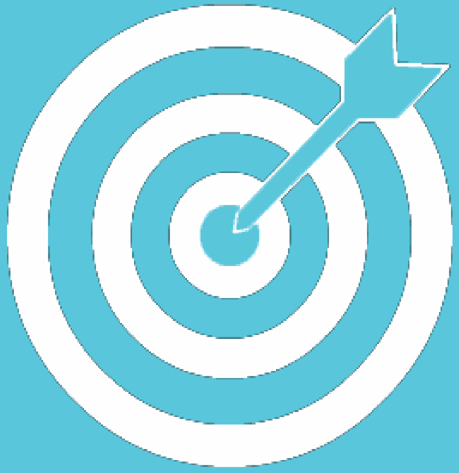
- Perform CRUD operations
- Analyze the data stored in MongoDB

System and Enterprise Architects

- Analyze use cases
- Design system architecture
- Scale read and write functions
- Optimize query performance

Objectives

After completing this lesson, you will be able to:



- Explain what NoSQL databases are
- Explain the purpose of NoSQL databases
- List the benefits of NoSQL database over traditional RDBMS database
- Identify various types of NoSQL databases
- List the differences between NoSQL and RDBMS
- Explain MongoDB in relation to the CAP theorem

What is NoSQL?

Not Only SQL(NoSQL) is a new set of database that is not based on the Relational Database Management Systems (RDBMS) principles.

NoSQL was introduced by Carl Strozzi in 1998 to name his file-based database.

NoSQL represents a group of products and a various related data concepts for storage and management of large data sets.

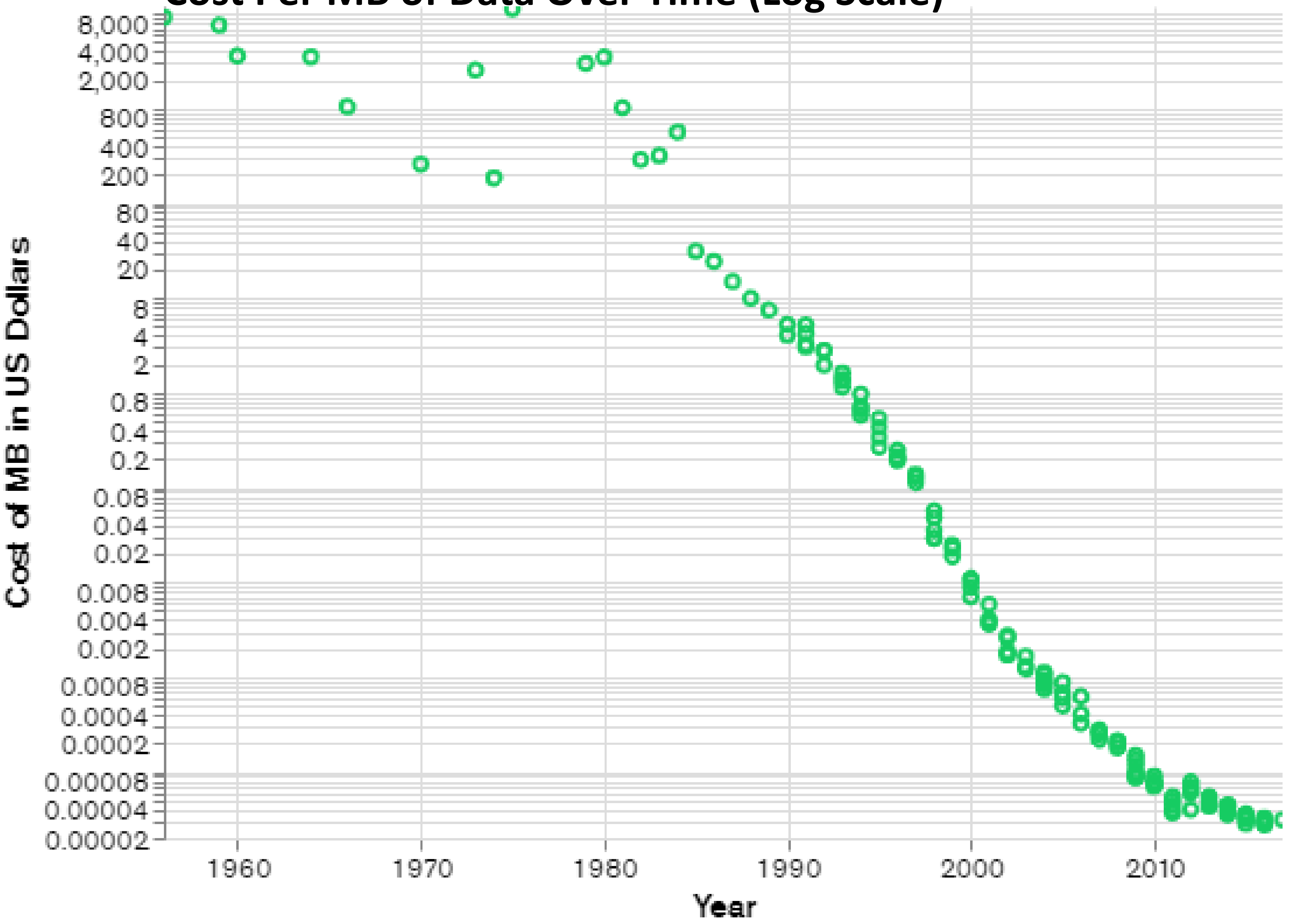
! This was the name of the hashtag (#nosql) for a meetup to discuss open source distributed databases.

What is NoSQL? (contd.)

NoSQL databases can have some common set of features such as:

- Non-relational data model
- Runs well on clusters
- Mostly open source
- Build for new generation web applications
- Is schema-less
- Scale easily with large amounts of data and high user loads.
- No need to create a complex, difficult-to-manage data model simply for the purposes of reducing data duplication
- Databases optimized for developer productivity

Cost Per MB of Data Over Time (Log Scale)



Why NoSQL?

With the explosion of social media sites, such as Facebook and Twitter, the demand to manage large data has grown tremendously. NoSQL database:

- Resolved the challenges that were faced in storing, managing, analyzing, and archiving data
- As storage costs rapidly decreased, the amount of data applications needed to store and query increased.
- Data came in all shapes and sizes—structured, semistructured, and polymorphic—and defining the schema in advance became nearly impossible.
- NoSQL databases allow developers to store huge amounts of unstructured data, giving them a lot of flexibility.

! The NoSQL databases offers capabilities to handle large volumes of data using various available features.

Why NoSQL?

- Agile Manifesto -Recognizing the need to rapidly adapt to changing requirements.
  - Needed the ability to iterate quickly and make changes throughout their software stack—all the way down to the database model
- Cloud computing- developers began using public clouds to host their applications and data.
  - Wanted the ability to distribute data across multiple servers and regions to make their applications resilient, to scale-out instead of scale-up, and to intelligent geo-place their data

Difference Between RDBMS and NoSQL Databases

Following are the differences between RDBMS and NoSQL databases:

RDBMS

- Data stored in a relational model, with rows and columns
- Follows a fixed schema
- Supports vertical scaling
- Atomicity, Consistency, Isolation, and Durability (ACID) compliant

NoSQL

- Data stored in a host of different databases—each with different data storage models
- Follows dynamic schemas
- Supports horizontal scaling
- Is not ACID complaint

Benefits of NoSQL

NoSQL solutions offer the following benefits:

- Serves as the primary datasource/operational datastore for online applications.
- Handles “big data” use cases that involve data velocity, variety, volume, and complexity.
- Offers inherent continuous availability.
- Excels at distributed database and multi-data center operations.
- Provides strong replication abilities along with read-anywhere and write-anywhere capability with full location-independence support.
- Eliminates the need for a specific caching layer to store data.

Benefits of NoSQL(contd.)

Some more NoSQL benefits include:

- Can operate in the cloud settings and exploit the benefits of cloud computing.
- Supports inclusion of additional nodes in a cluster for performance enhancement.
- Offers a flexible schema design that can be altered without downtime or service disruption.
- Supports all major operating systems, proprietary add-ons, and all common developer languages.
- Are easy to implement and use; offer sturdy functionality to handle various enterprise applications.
- Supports open source communities, which makes regular contribution to enhance the core software.

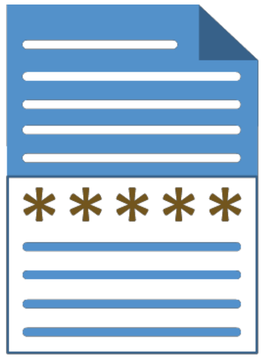
Types of NoSQL

The types of NoSQL databases are:

Key Value

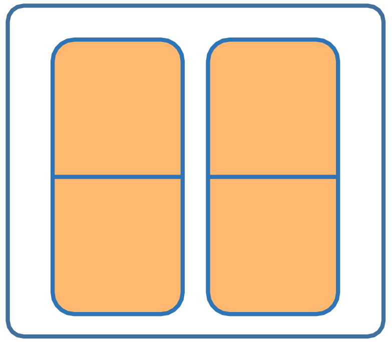
Example:
Riak, Tokyo Cabinet, Redis server, Memcached, Scalaris

Document-Based



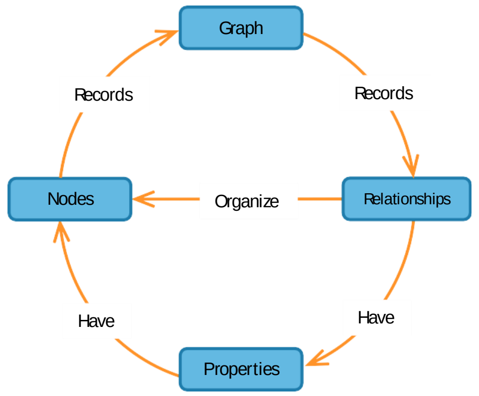
Example:
MongoDB, CouchDB, OrientDB, RavenDB

Column-Based



Example:
BigTable, Cassandra, Hbase, Hypertable

Graph-Based

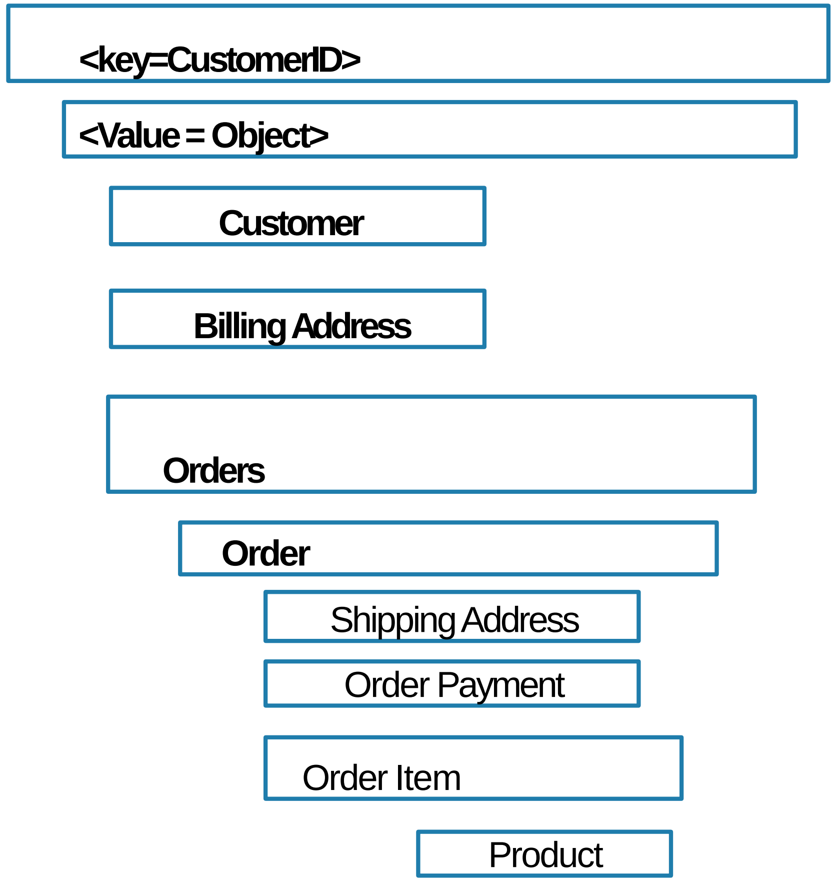


Example:
Neo4J, InfoGrid, Infinite Graph, Flock DB

Key-Value Database

Key-value stores are the simplest NoSQL databases that:

- Store every single item in the database as an attribute name (key) together with its value.
- Are ideal for handling Web scale operations that need to scale across thousands of servers and millions of users with extremely quick and optimized retrieval.



! Data loss may occur in Memcached when implementing caching of user performances. The same data stored in Riak, may not be lost, but may need update.

Key-Value Database (contd.)

Following are the advantages and disadvantages of the Key-Value database.

Advantages

- Can perform queries
- Supports schemas
- Data can be accessed using a key

Disadvantages

- Does not provide any traditional database capabilities
- Maintaining unique keys are difficult as the volume of data increases

Document Database

Based on the concept of documents, the Document database:

- Stores and retrieves various documents in JavaScript Object Notation (JSON), Extensible Markup Language (XML), BSON
- Consists of maps, structures, and scalar values
- Store documents in the value part of the key-value store

Document Database Example

Examples of Document databases are:

- **MongoDB**: Provides a rich query language and many useful features such as built-in support for MapReduce-style aggregation and geospatial indexes.
- **Apache CouchDB**: Uses JSON for documents, JavaScript for MapReduce indexes, and regular HTTP for its API.

Example: (MongoDB) document

```
    {Name:“",
        Address:" 10685 Hazelhurst Dr, Houston, TX 77043, United States”,
        Courses: [“Big Data”,”Python”,”Android”,”PMP”,”ITIL”],
        Offices: [ “NYK”,”Dubai”,”BLR”],
        }
```

Column-Based Database

Column-based databases store data in column families as rows. These rows contain multiple columns associated with a row key.

In a Column-Based database:

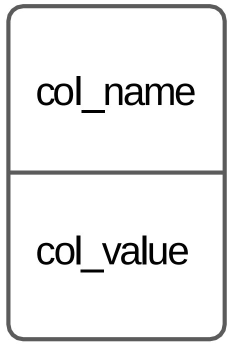
- The key identifies the rows.
- The various rows need not have the same columns.

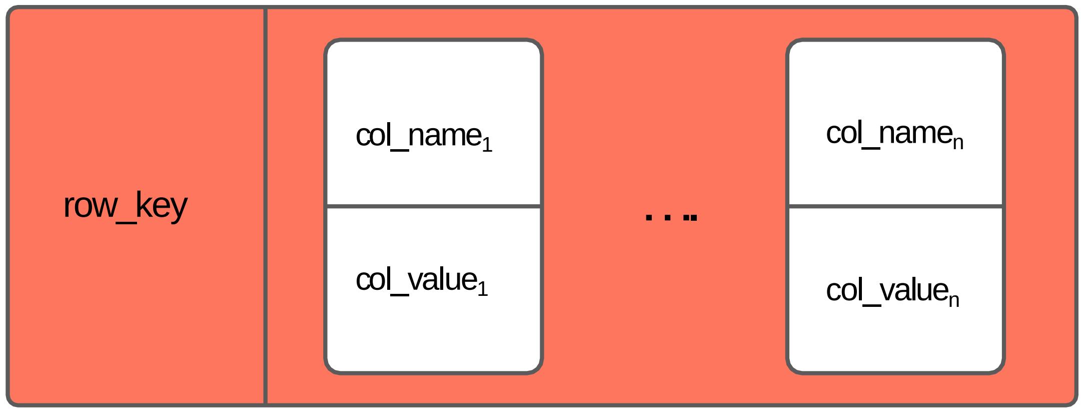
Can add a column to a row at any time without adding it to other rows.

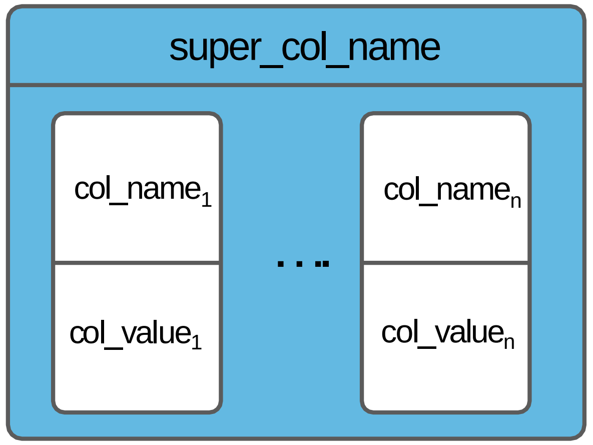
Column-Based Database (contd.)

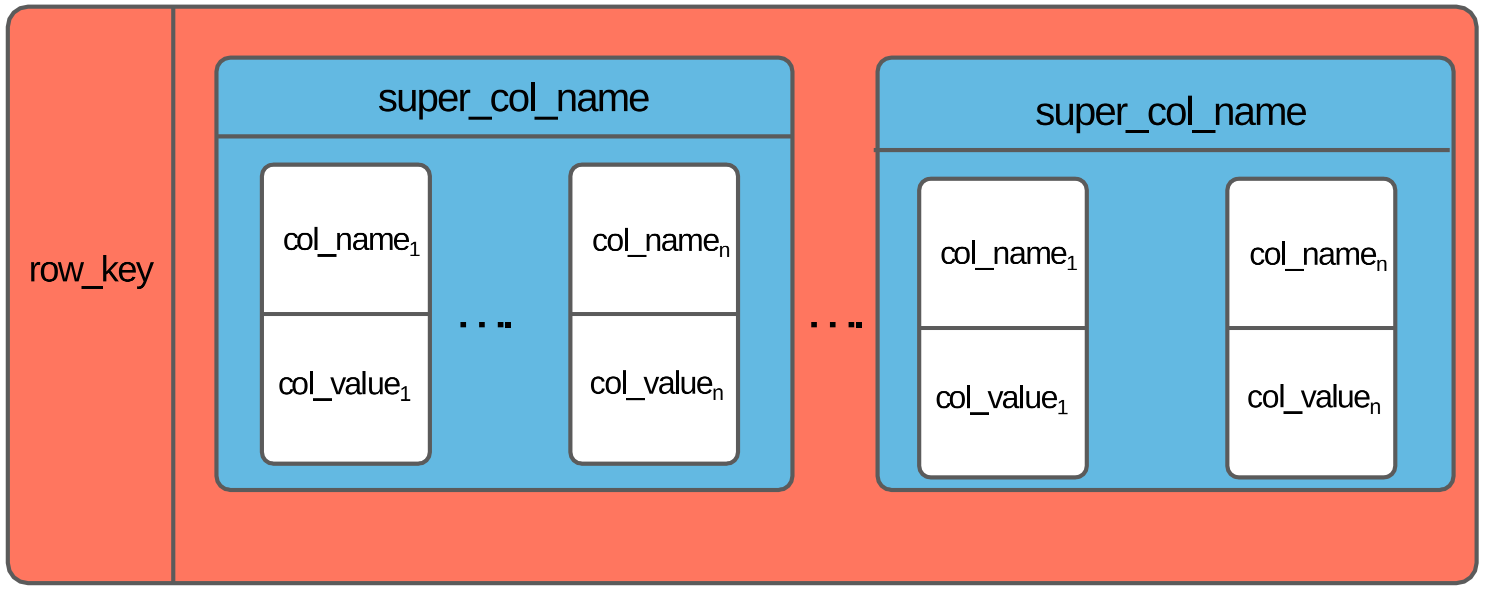
A Column-Based database reads and writes data to and from hard disk storage to quickly return a query. It lets you access individual data elements as a group in columns rather than access individually row-by-row.

Column-Based Database (contd.)









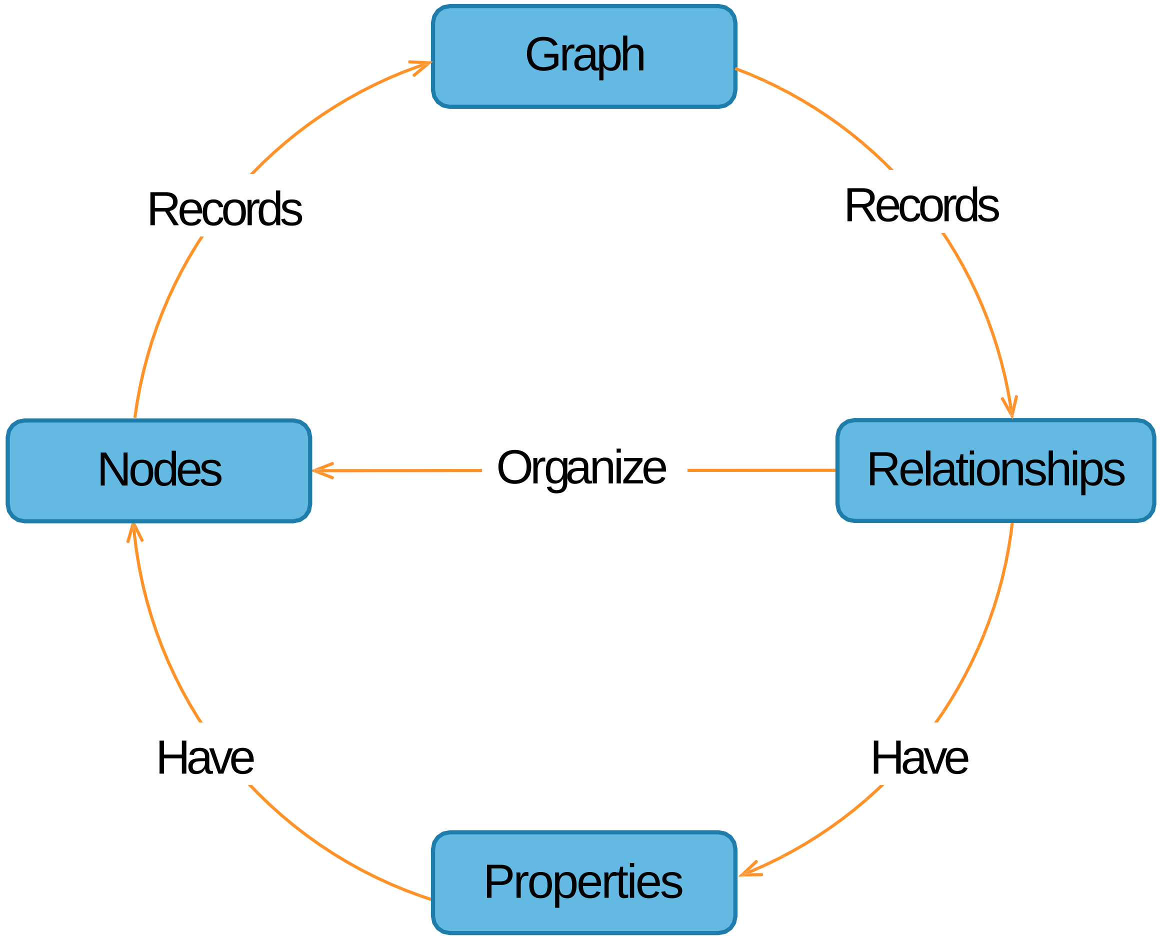
Column-Based Database Example

Cassandra is a column-based database. The features include:

- Fast and easily scalable
- Write operations spread across the cluster
- Any node can handle the read and write operations

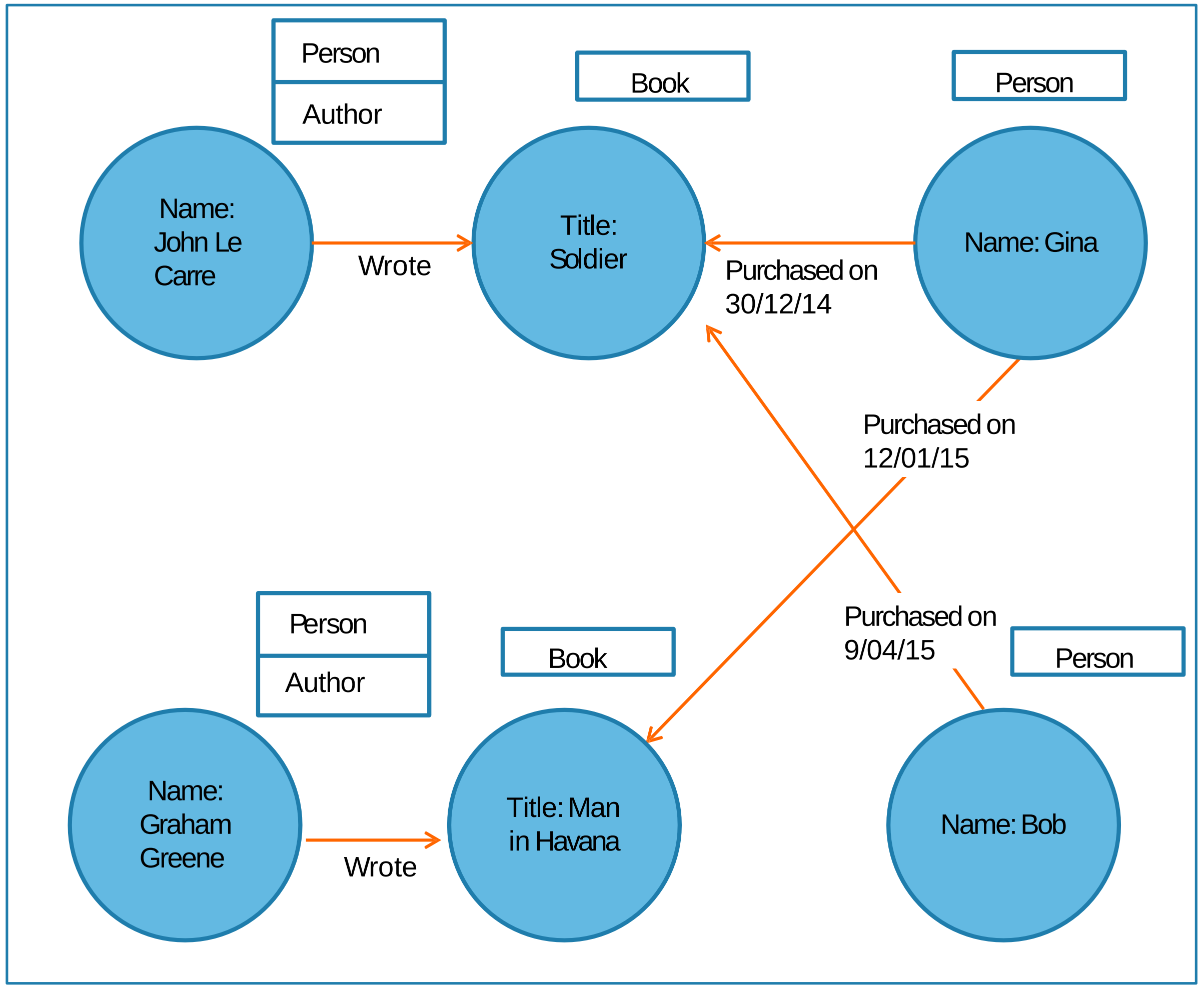
Graph Database

A Graph database makes relationships readily available for any join-like execution allowing quick access of millions of connections. It lets you store data and its relationships with other data in the form of nodes and edges.



Graph Database (contd.)

- Examples of Graph database are—Neo4J, InfiniteGraph, OrientDB, and FlockDB.
- Neo4J is ACID compliant.
- FlockDB was created by Twitter for relationship related analytics.



Labeled Property Graph Data Model

When Would You Want to Use NoSQL over SQL?

- Pace of development with NoSQL databases can be much faster than with a SQL database.
- Structure of many different forms of data is more easily handled and evolved with a NoSQL database.
- Amount of data in many applications cannot be served affordably by a SQL database.
- Scale of traffic and need for zero downtime cannot be handled by SQL.
- New application paradigms can be more easily supported.

Schema Design for NoSQL Databases

- NoSQL databases are designed to store data that does not have a fixed structure that is specified prior to developing the physical model, developers focus on the physical data model.
- Developing applications for massive, horizontally distributed environments.
- Emphasis on figuring out how the scalability and performance of the system will work.
- Need to optimize data access, which puts the focus on query patterns and business workflows.
- Goal for schema design is to plan keys and indexes that are fast and effective for application queries and that complement workflow patterns.
- There will be an iterative process of schema design throughout the lifetime of the application.

What makes document databases different from relational databases?

1. Intuitive Data Model: Faster and Easier for Developers

- Documents map to the objects in your code, so they are much more natural to work with.
- no need to decompose data across tables, run expensive JOINs, or integrate a separate ORM layer.
- Data that is accessed together is stored together, so you have less code to write and your users get higher performance.

2. Flexible Schema: Dynamically Adapt to Change

- A document's schema is dynamic and self-describing, so you don't need to first pre-define it in the database.
- Fields can vary from document to document and you modify the structure at any time, avoiding disruptive schema migrations.
- Some document databases offer JSON Schema so you can optionally enforce rules governing document structures.

What makes document databases different from relational databases?

3. Universal: JSON Documents are Everywhere

- Lightweight, language-independent, and human readable, JSON has become an established standard for data interchange and storage.
- Documents are a superset of all other data models so you can structure data any way your application needs – rich objects, key-value pairs, tables, geospatial and time-series data, and the nodes and edges of a graph.
- can work with documents using a single query language, giving you a consistent development experience however you've chosen to model your data.

What makes document databases different from relational databases?

4. Powerful: Query Data Anyway You Need

- An important difference between document databases is the expressivity of the query language and richness of indexing.
- MongoDB Query Language is comprehensive and expressive.
- Ad hoc queries, indexing, and real time aggregations provide powerful ways to access, transform, and analyze your data.
- With ACID transactions you maintain the same guarantees you’re used to in SQL databases, whether manipulating data in a single document, or across multiple documents living in multiple shards.

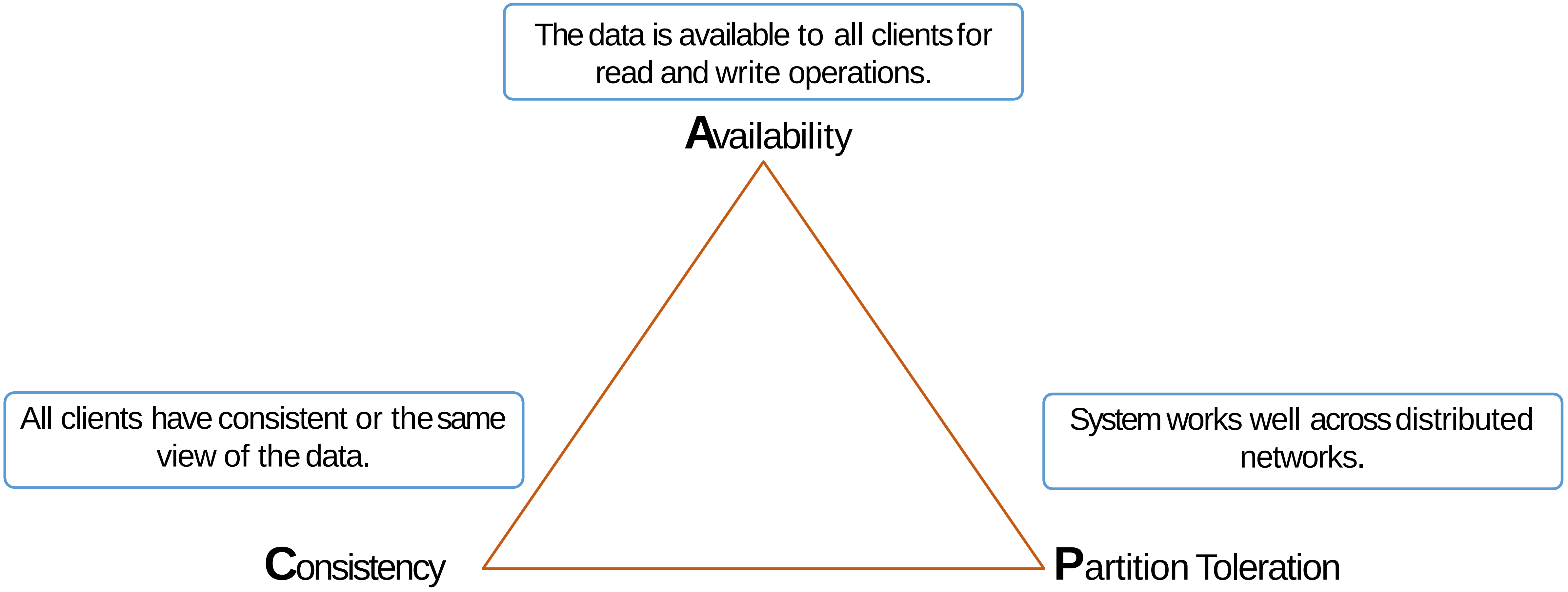
What makes document databases different from relational databases?

5. Distributed: Resilient and Globally Scalable

- Unlike monolithic, scale-up relational databases, document databases are distributed systems at their core.
- Documents are independent units which makes it easier to distribute them across multiple servers while preserving data locality.
- Replication with self-healing recovery keeps your applications highly available while giving you the ability to isolate different workloads from one another in a single cluster.
- Native sharding provides elastic and application-transparent horizontal scale-out to accommodate your workload’s growth, along with geographic data distribution for data sovereignty.

CAP Theorem

In a distributed system, the following three properties are important.


The data is available to all clients for read and write operations.
Availability
All clients have consistent or the same view of the data.
System works well across distributed networks.
Consistency
Partition Toleration

CAP Theorem (contd.)

The CAP theorem states that in any distributed system only two of the three properties can be used simultaneously—consistency, availability, or partition tolerance. To adjust the database, understanding the following requirements is essential:

- How the data is consumed by the system
- Whether the data is read or write heavy
- If there is a need to query data with random parameters
- If the system is capable of handling inconsistent data

Consistency

Below are the definitions of consistency in CAP theorem and ACID.

CAP Theorem

Consistency in CAP theorem means consistent read and write operations for the same sets of data so that concurrent operations see the same valid and consistent data state.

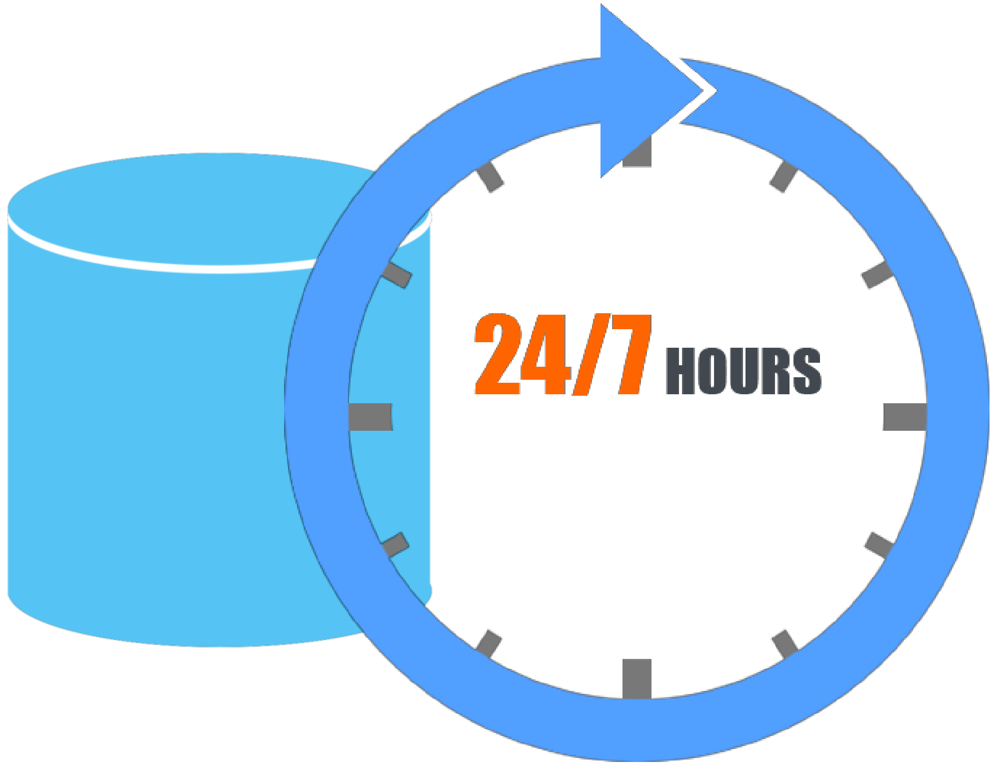
ACID

Consistency in ACID means if the data does not satisfy predefined constraints, it is not persisted.

Availability

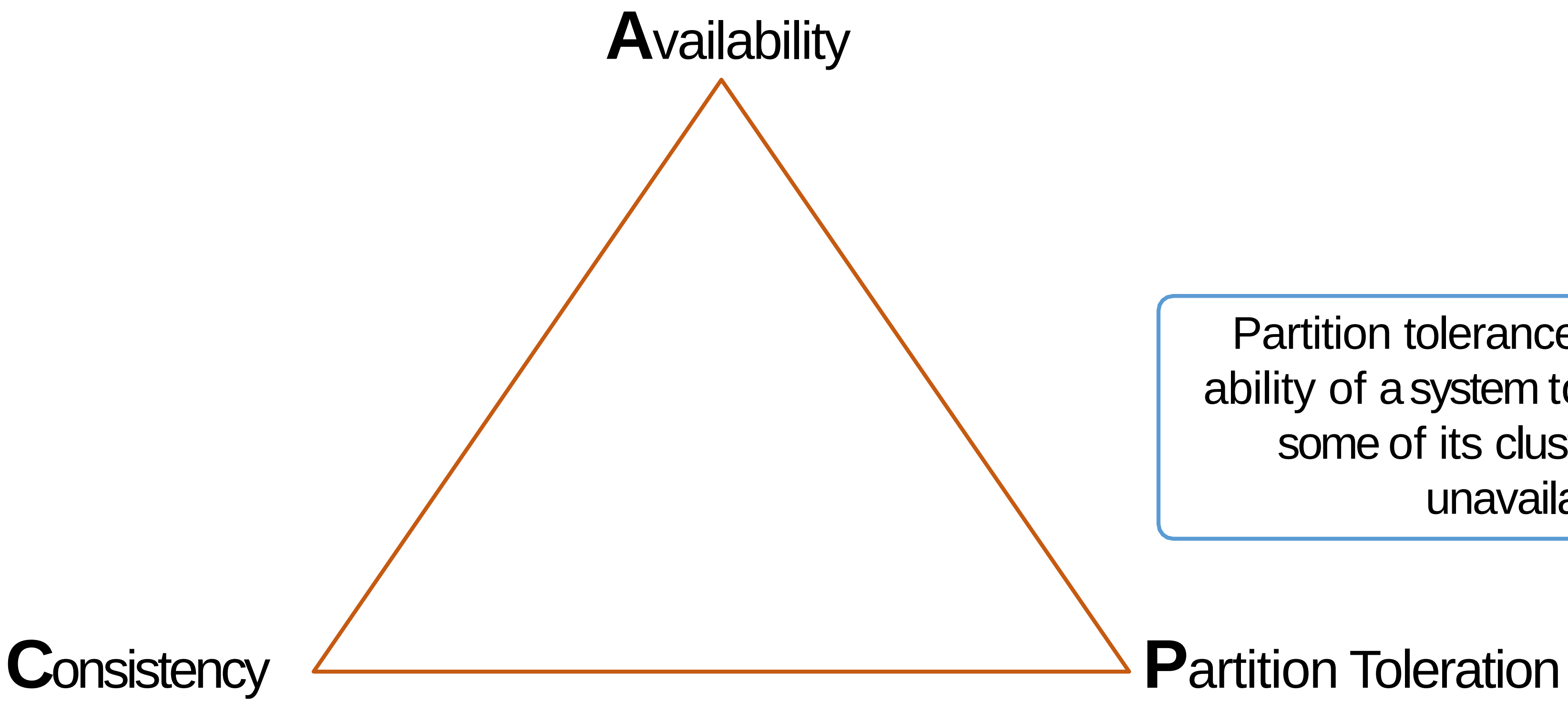
Availability means:

- The database system is available to operate when required.
- If a system is not available to serve a request when needed, it is not available.



Partition Tolerance

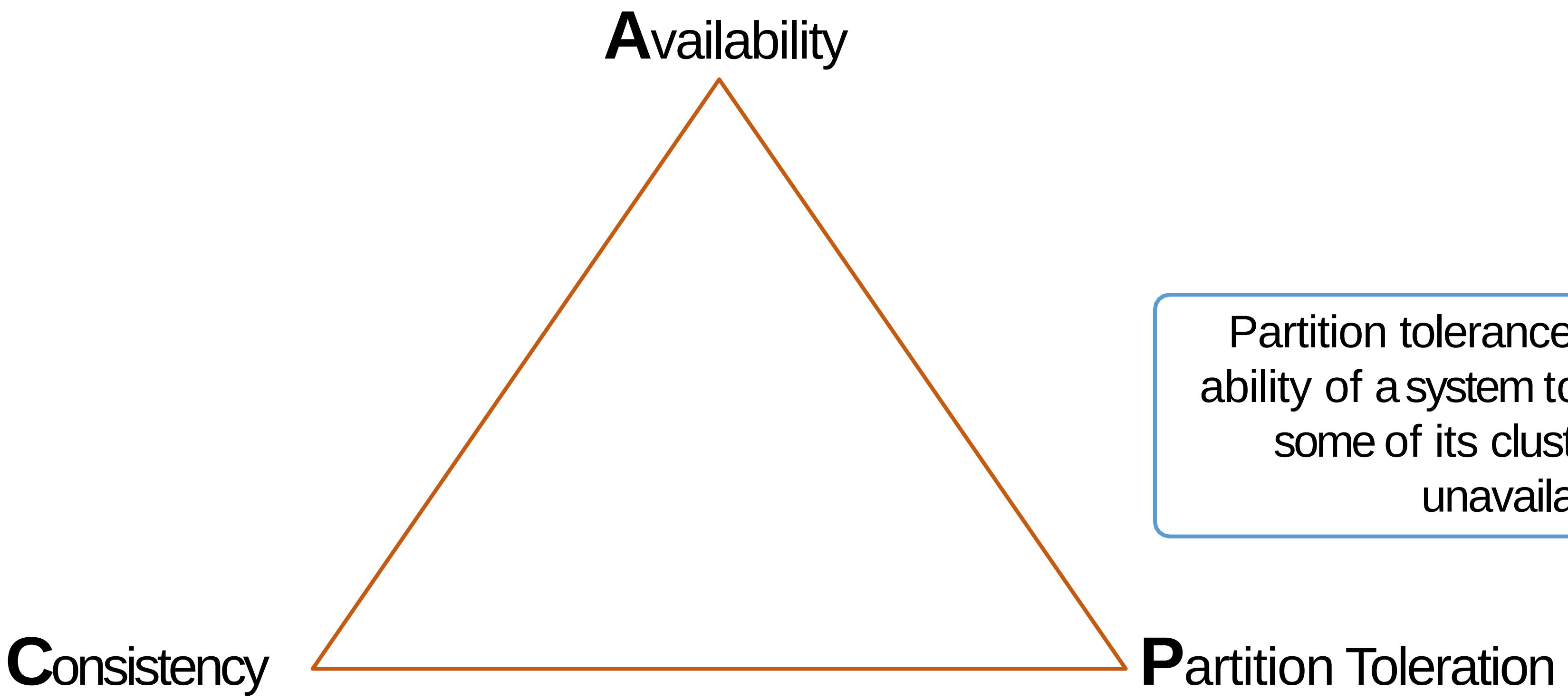
Partition tolerance or fault-tolerance is the third element of the CAP theorem. It measures the ability of a system to continue its service when some of its clusters become unavailable.



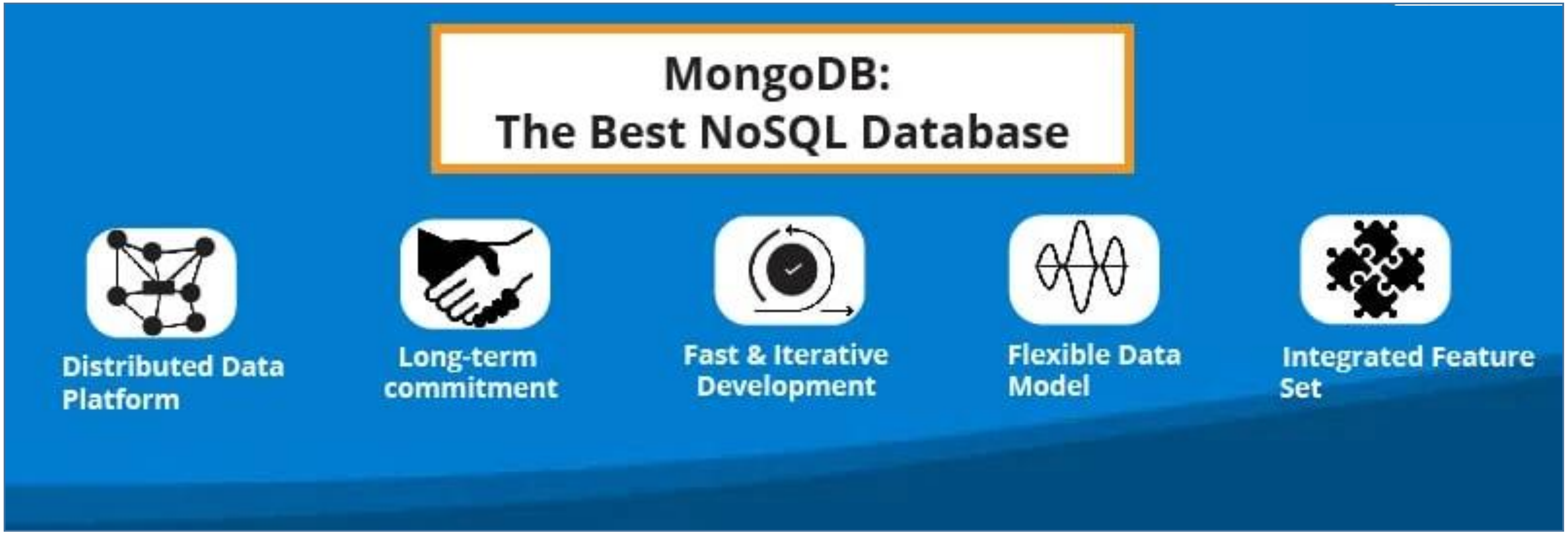
Partition tolerance measures the ability of a system to continue when some of its clusters become unavailable.

MongoDB As per CAP

Partition tolerance or fault-tolerance is the third element of the CAP theorem. It measures the ability of a system to continue its service when some of its clusters become unavailable.



Partition tolerance measures the ability of a system to continue when some of its clusters become unavailable.



MongoDB:
The Best NoSQL Database

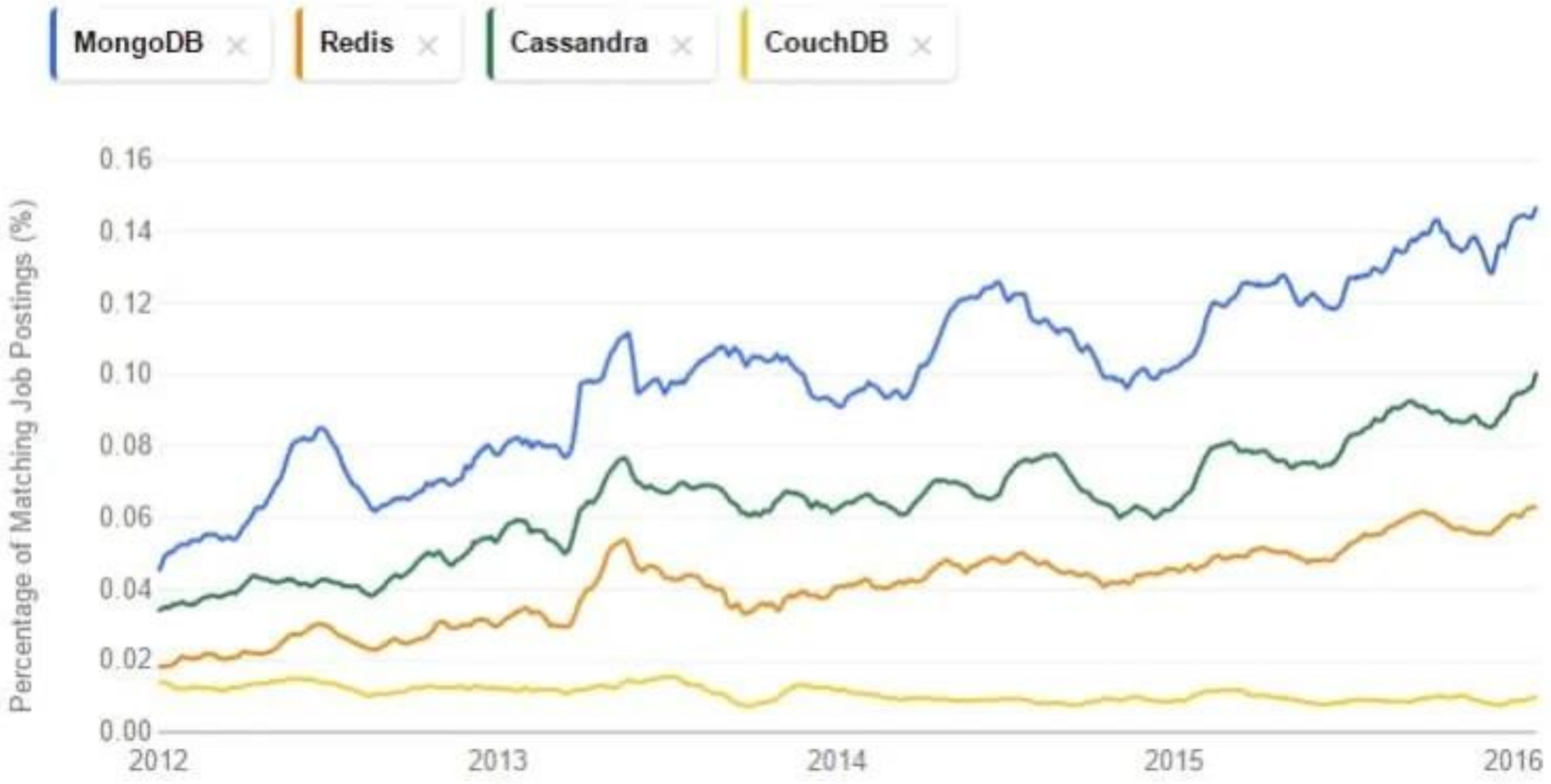
Distributed Data Platform

Long-term commitment

Fast & Iterative Development

Flexible Data Model

Integrated Feature Set


MongoDB
Redis
Cassandra
CouchDB
Percentage of Matching Job Postings (%)
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
2012
2013
2014
2015
2016

Knowledge Check

Case Study

Scenario | Analysis | Solution | Architecture

An eCommerce organization receives numerous hits per second from its customers, who browse and purchase products from its website and mobile application. The organization decides to categorize the sale per hour in various categories and geographicallocation.

This categorization task is allotted to the research team, who needs to perform data backup for the analysis. The team has to choose the best database that supports the replication, load distribution, and deep analytics.

Click ***Analysis*** *to know the team's next move.*

mongoDB

Case Study

Scenario Analysis Solution Architecture

The research team examines various databases and prefers MongoDB, which is a popular distributed database and supports deep analytics, replication, and load distribution. It offers 11000 hits per second on one node.

Following are the reasons to prefer MongoDB:

1. By default, MongoDB performs read and write primary node. It also allows read option from secondary node.
2. The scale out of the database is achieved using sharding.
3. The nodes are flexible and therefore a node can be set up only for data backup.

Click **Solution** *to know the steps to perform data backup.*

mongoDB

Case Study

Scenario Analysis Solution Architecture

Following assumptions and observations are taken to choose the best database:

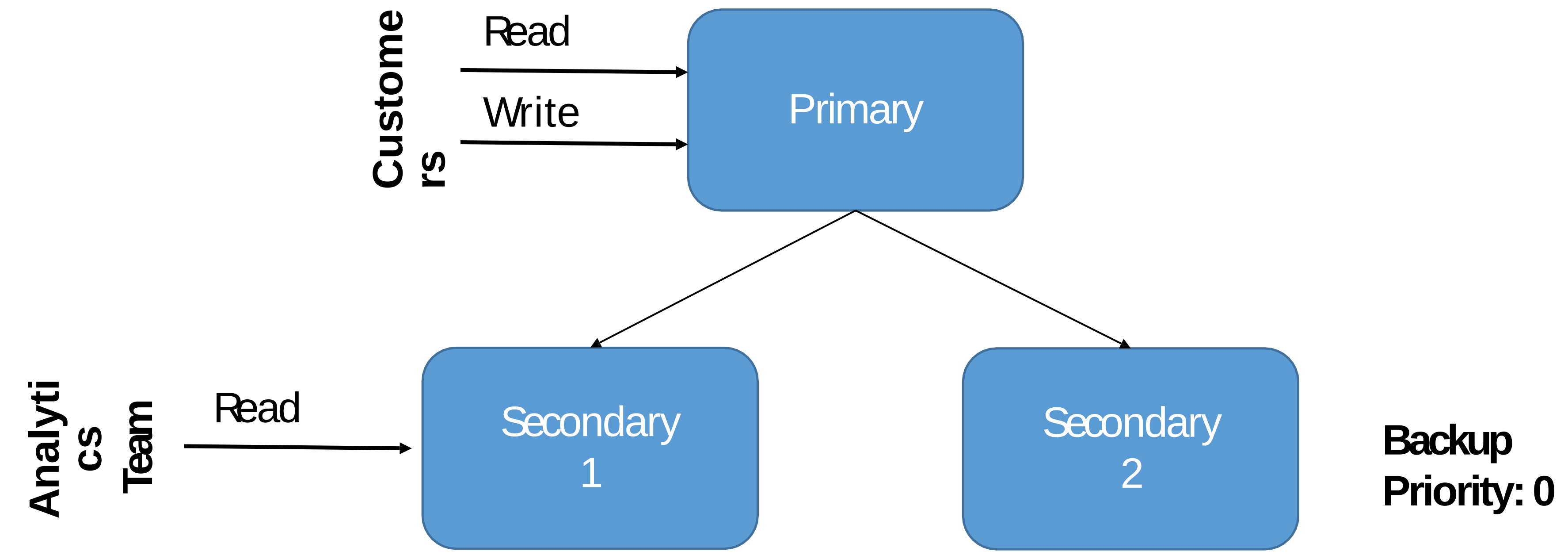
1. Replication factor of 3 can be set up so that there will be 1 primary and 2 secondary nodes.
2. The hits from customers go to primary for both read and write. It can satisfy 10k hits and has buffer for an extra 1000.
3. The hits from internal team who performs analytics go to one of the secondary. As the secondary performs only read operations, it won't matter much even if it misses the last few seconds of data.
4. The second secondary can be just kept for backup so that the data is safe always. So its priority will be 0 when setting up the replication configuration.

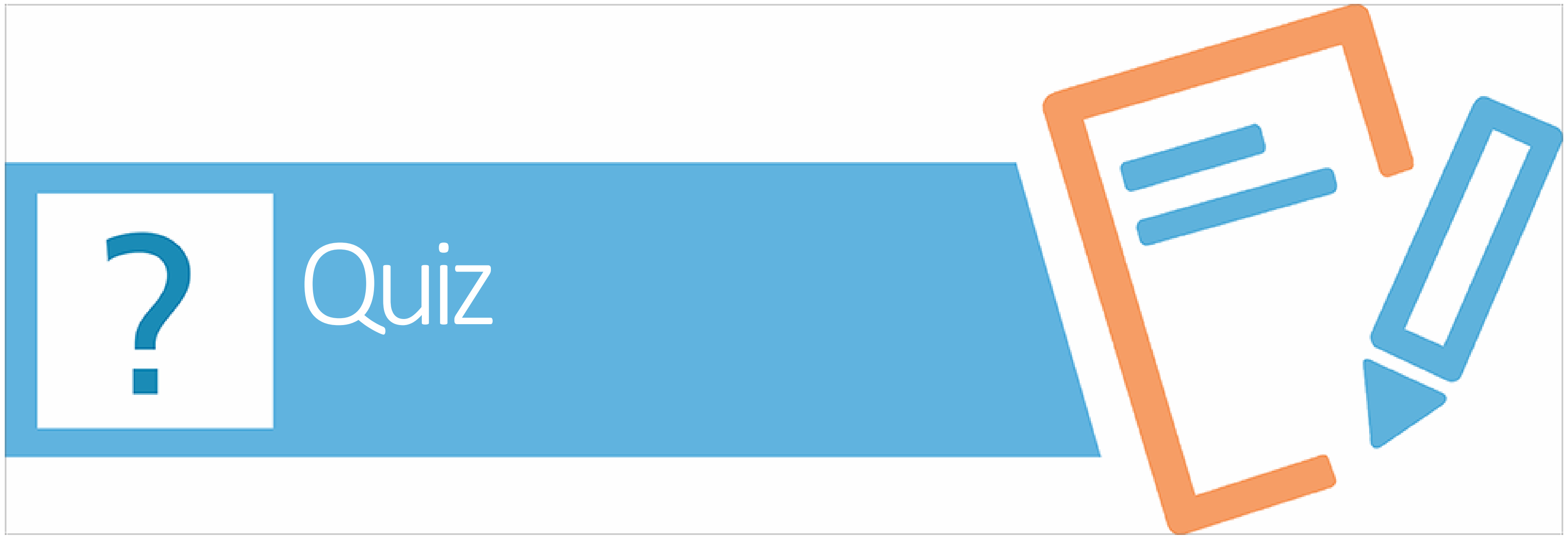
mongoDB

*Click **Demo** to know the query execution process.*

Case Study

Scenario | Analysis | Solution | Architecture




?
Quiz

QUIZ 1

Which of the following is a benefit of using NoSQL database?

a. Scalability

b. Flexible schema

c. Open source technology

d. All of the above

QUIZ 1

Which of the following is a benefit of using NoSQL database?

a. Scalability

b. Flexible schema

c. Open source technology

d. All of the above

The correct answer is d.

Explanation: NoSQL database provides scalability, flexible schema, and open source technology.

QUIZ 2

Which of following is type of Document database ?

a. MongoDB
b. Redis server
c. Riak
d. Cassandra

QUIZ 2

Which of following is type of Document database ?

a. MongoDB

b. Redis server

c. Riak

d. Cassandra

The correct answer is a.

Explanation: MongoDB is document database that stores data as BSON.

QUIZ 3

According to CAP theorem, what kind of a system is MongoDB?

a. CA

b. CP

c. AP

d. CAP

QUIZ 3

According to CAP theorem, what kind of a system is MongoDB?

a. CA

b. CP

c. AP

d. CAP

The correct answer is b.

Explanation: MongoDB is CP system.

Summary

Here is a quick recap of what was covered

- NoSQL represents a class of products and a collection of diverse or related data concepts for storage and manipulation.
- NoSQL databases are used to efficiently manage large-volume and semi-structured data.
- The four basic NoSQL database types are— Key-Value, Document-based, Column-based, and Graph-based.
- According to the CAP theorem, a distributed computer system cannot provide all the three properties together—consistency, availability, and partition tolerance.