



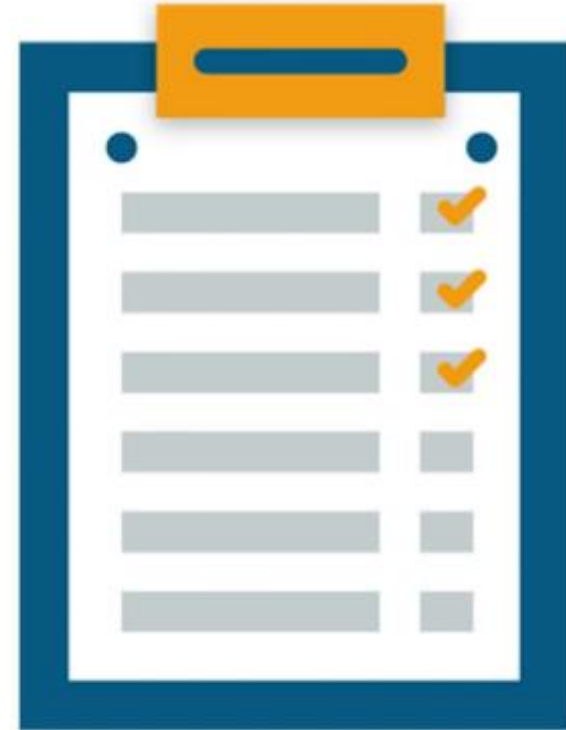
Getting Started with MongoDB



Today's Training Objectives

After completing this module, you should be able to:

- Differentiate between the various categories of database
- Understand the goals and design of MongoDB
- List MongoDB tools
- Describe JSON and BSON
- Install MongoDB on Windows, Linux, Mac OS, etc.
- Setup MongoDB environment



Database – The Need

1

Store, organize, and manage large amounts of data on a single platform

Facilitate a well-planned platform for data analysis

2

3

Promote a disciplined approach to data management

One stop solution to manage security, multiuser access control, backup & recovery, etc.

4

Introduction to Database Categories

Categories	Also known as	Example 1	Example 2
OLTP	RDBMS/ Real Time	Oracle	MS SQL
OLAP	DSS/ DW	Netezza	Sap Hana
NewSQL	NoSQL/ Big Data	MongoDB	CouchDB

What is OLTP?

- An OLTP system captures and maintains transaction data in a database. Each transaction involves individual database records made up of multiple fields or columns. Examples include banking and credit card activity or retail checkout scanning.
- In OLTP, the emphasis is on fast processing, because OLTP databases are read, written, and updated frequently. If a transaction fails, built-in system logic ensures data integrity.

What is OLAP?

- OLAP applies complex queries to large amounts of historical data, aggregated from OLTP databases and other sources, for data mining, analytics, and **business intelligence** projects. In OLAP, the emphasis is on response time to these complex queries. Each query involves one or more columns of data aggregated from many rows. Examples include year-over-year financial performance or marketing lead generation trends.
- OLAP databases and **data warehouses** give analysts and decision-makers the ability to use custom reporting tools to turn data into information. Query failure in OLAP does not interrupt or delay transaction processing for customers, but it can delay or impact the accuracy of business intelligence insights.

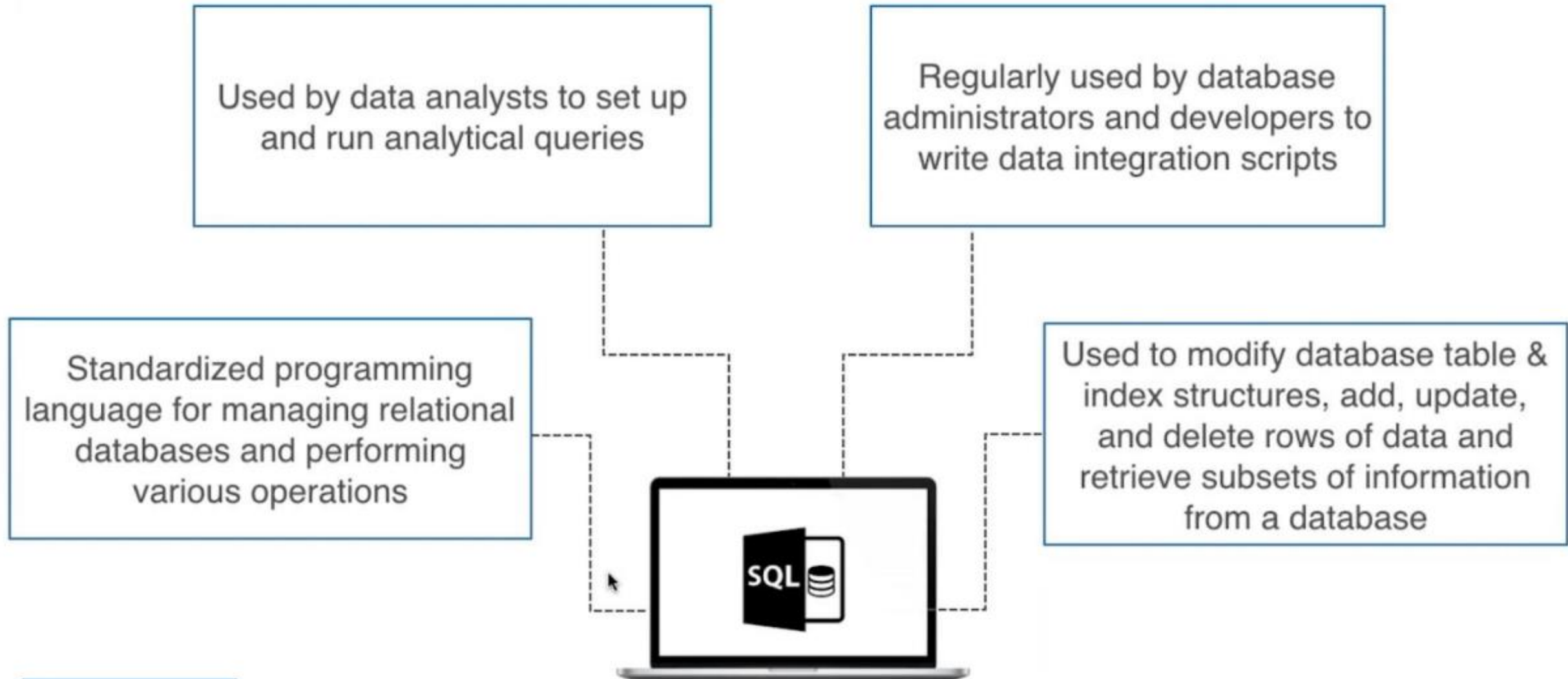
What is a NewSQL Database?

- NewSQL databases aim to improve upon these flaws by reincorporating some relational database features and combining them with the benefits of NoSQL.
- By using the modern data structure of NoSQL but maintaining the ACID (atomicity, consistency, isolation and durability) guarantees of RDBs, NewSQL offers the best of both worlds

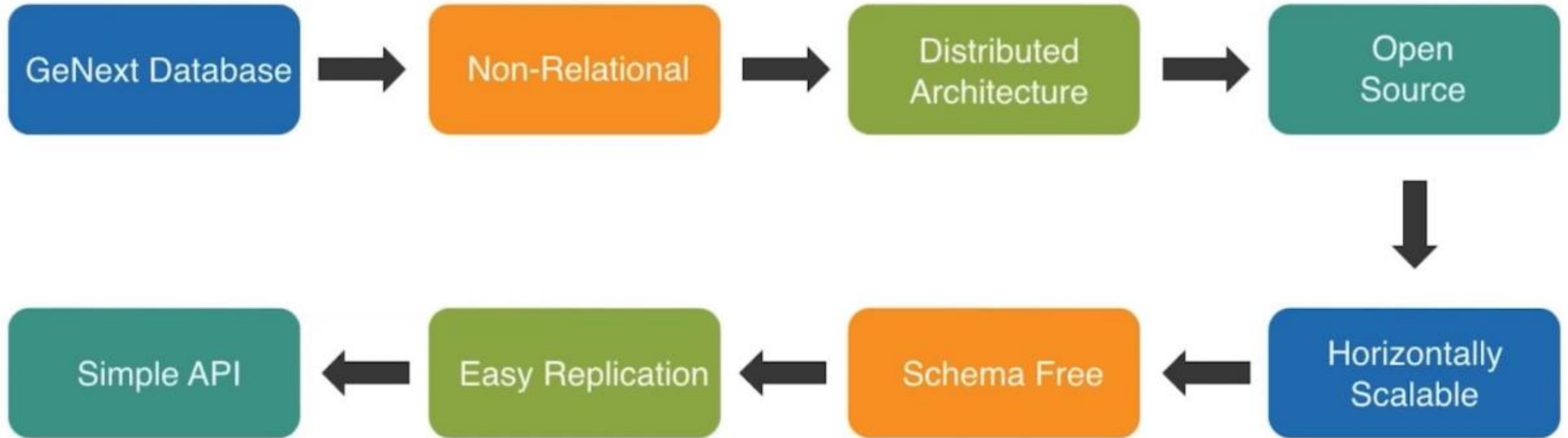
Specifics of Database Categories

OLTP	OLAP	NewSQL
<ul style="list-style-type: none">▪ Stores real-time data▪ Heterogeneous in nature▪ ATM/ Retail Transaction▪ Short term storage	<ul style="list-style-type: none">▪ Stores processed data▪ Homogeneous in nature▪ Access to multiple DBMS's▪ Long term storage	<ul style="list-style-type: none">▪ Stores all Application data▪ Schema agnostic▪ Handling Big Data requirements▪ Long term storage

SQL - Structured Query Language



NoSQL – You don't need SQL



SQL vs NOSQL

Entity	SQL Databases	NoSQL Databases
Type	One Type – SQL (with slight variations)	Multiple Types – Document, Key-Value, Tabular, Graph
Developed In	1970	2000
Examples	Oracle, MSSQL, DB2 etc.	MongoDB, Cassandra, HBase, Neo4j
Schemas	Fixed	Dynamic
Scaling	Vertical	Horizontal
Dev Model	Mix	Open Source
Properties Followed	ACID	BASE

SQL:- ACID Property

Atomicity

Entire transaction is either successful or fails to load completely, no partial execution

Consistency

Integrity constraints are maintained within database to maintain consistency

Isolation

Modification in a transaction will not be visible to any other transaction until the change is committed

Durability

Committed data is persisted even after system failure as it is stored in non-volatile memory

NoSQL Base Property



Basic Availability

- The database appears to work most of the time

Soft-state

- Stores don't have to be write-consistent or mutually consistent all the time

Eventual consistency

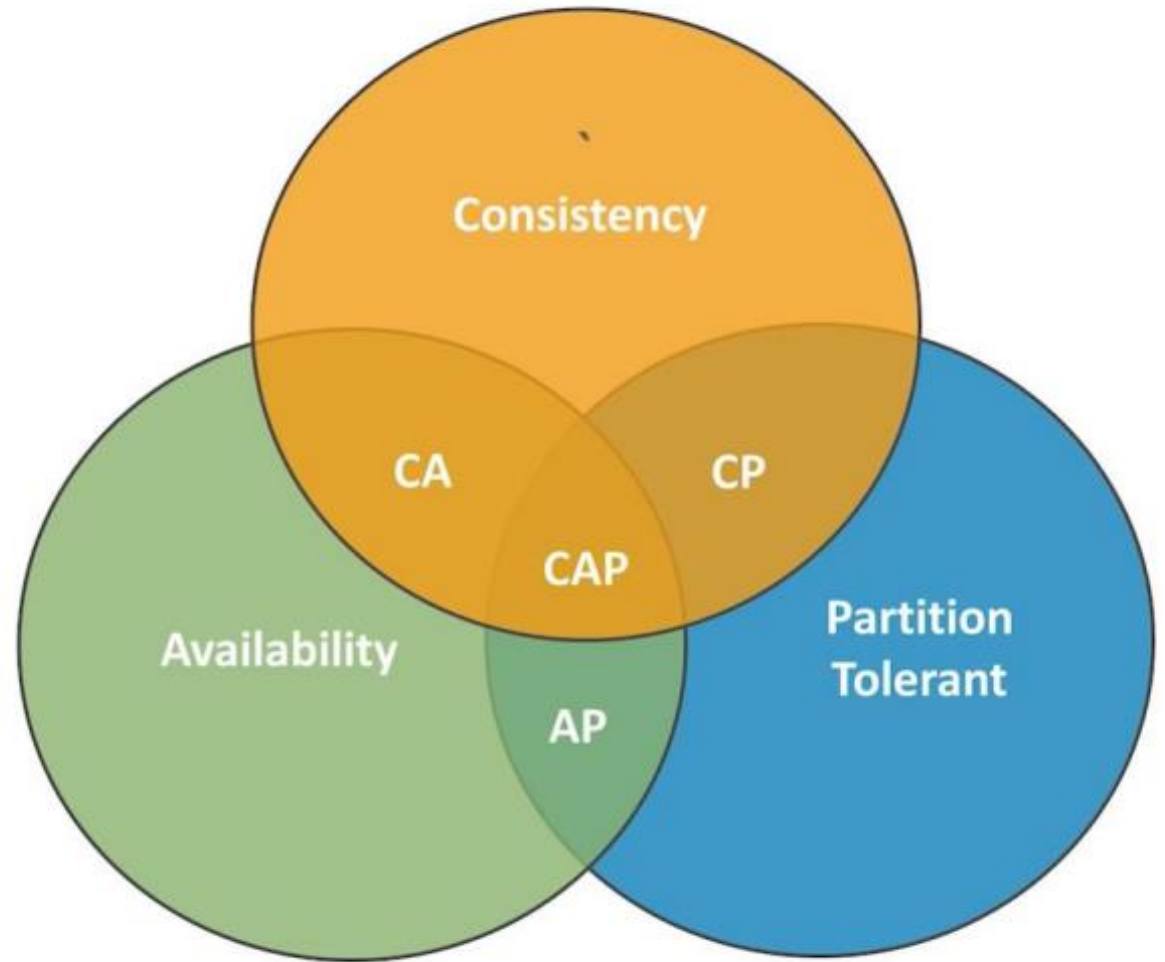
- Stores exhibit consistency at some later point

The CAP Theory

Consistency	Availability	Partition Tolerance
<ul style="list-style-type: none">▪ This means that 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	<ul style="list-style-type: none">▪ This means that the system will not have downtime (100% service uptime guaranteed)▪ Every node (if not failed) always executes query	<ul style="list-style-type: none">▪ This means that the system continues to function even when the communication between servers is unreliable▪ The servers may be partitioned into multiple groups that cannot communicate with one another

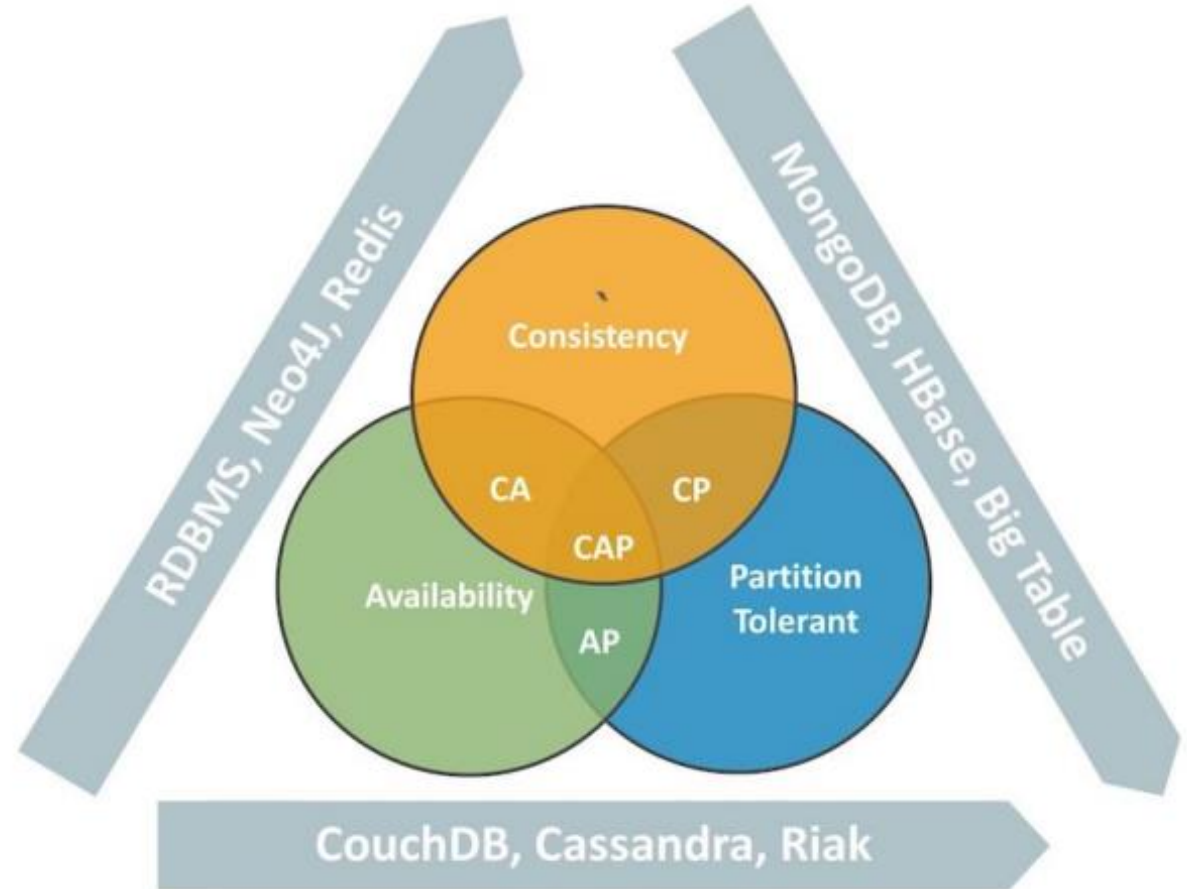
CAP Combinations

- Theoretically, it is not possible to fulfill all the 3 requirements
- CAP provides the basic requirement for a distributed system to follow 2 of the 3 requirements
- Hence, all the current NoSQL databases follow different combinations of C, A, P from the CAP theorem



CAP – CA, CP, AP

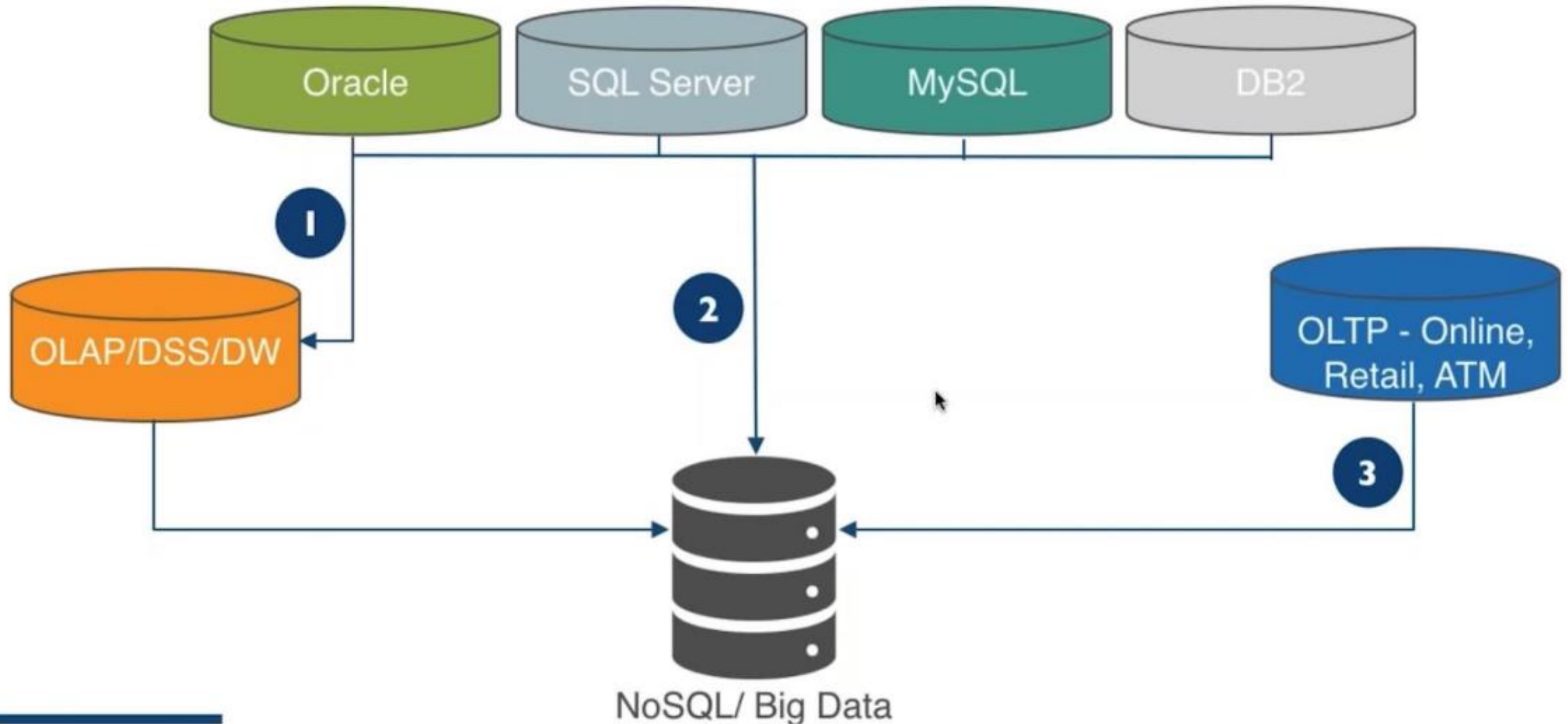
- CA: Single site cluster, all nodes are always in contact
- CP: Some data may not be accessible, however, the rest is still consistent and accurate
- AP: System is available under partitioning, however, some of the data returned may be inaccurate



Features of NoSQL Database

- Supports massive number of concurrent users – tens of thousands to millions
 - Provides extremely responsive experience to a globally distributed base of users
 - Always available with no downtime
 - Quickly adapts to changing requirements with frequent updates and new features
 - Handles semi and unstructured data
-

NoSQL Database – Storage



DSS/DW and OLAP Workload Considerations

- Typical OLAP queries and batch or ETL jobs against DW databases see long-running, sequential, and large-block read or write patterns to the data file. Traffic to the data files is typically larger than 64 KB, although it can span a wide range of block sizes. Data file I/O can range from 8 KB to 8 MB, depending on the workload and on the use of new features such as columnstore indexes in a data warehouse.
- As with OLTP databases, HPE recommends that data files and transaction log files for OLAP and DW databases be placed in separate volumes. Create a custom performance policy for data files that specifies a larger block size (either 16 KB or 32 KB) with compression enabled. When SQL Server is deployed on adaptive flash arrays, enable caching on data volumes because ad-hoc queries present a more random I/O pattern to the storage.
- Very large OLAP and DW databases benefit from the use of multiple filegroups spread across different volumes in conjunction with table partitioning. Filegroups allow the overall increase of queue depth and parallel efficiency on the server, while table partitioning allows the organization of data so that only relevant information is accessed or loaded.
- **Best Practice:** For optimum performance, place SQL Server data files and log files in separate volumes, and use dedicated performance policies that are geared toward larger-block workloads. Consider the use of multiple filegroups across different volumes, and enable table partitioning.

Oracle Database


- Oracle database products offer customers cost-optimized and high-performance versions of Oracle Database, the world's leading converged, multi-model database management system, as well as in-memory, NoSQL and MySQL databases.
- Oracle Autonomous Database, available on premises via Oracle Cloud@Customer or in the Oracle Cloud Infrastructure, enables customers to simplify relational database environments and reduce management workloads.

SQL Server

- SQL Server is a relational database management system (RDBMS) developed and marketed by Microsoft. As a database server, the primary function of the SQL Server is to store and retrieve data used by other applications.

A large orange circle occupies the left side of the slide, partially cut off by the edge.

MySQL

- MySQL is a widely used relational database management system (RDBMS).
 - MySQL is free and open-source.
 - MySQL is ideal for both small and large applications.
- 
- A yellow dashed line consisting of several short, curved segments is located in the bottom right corner of the slide.

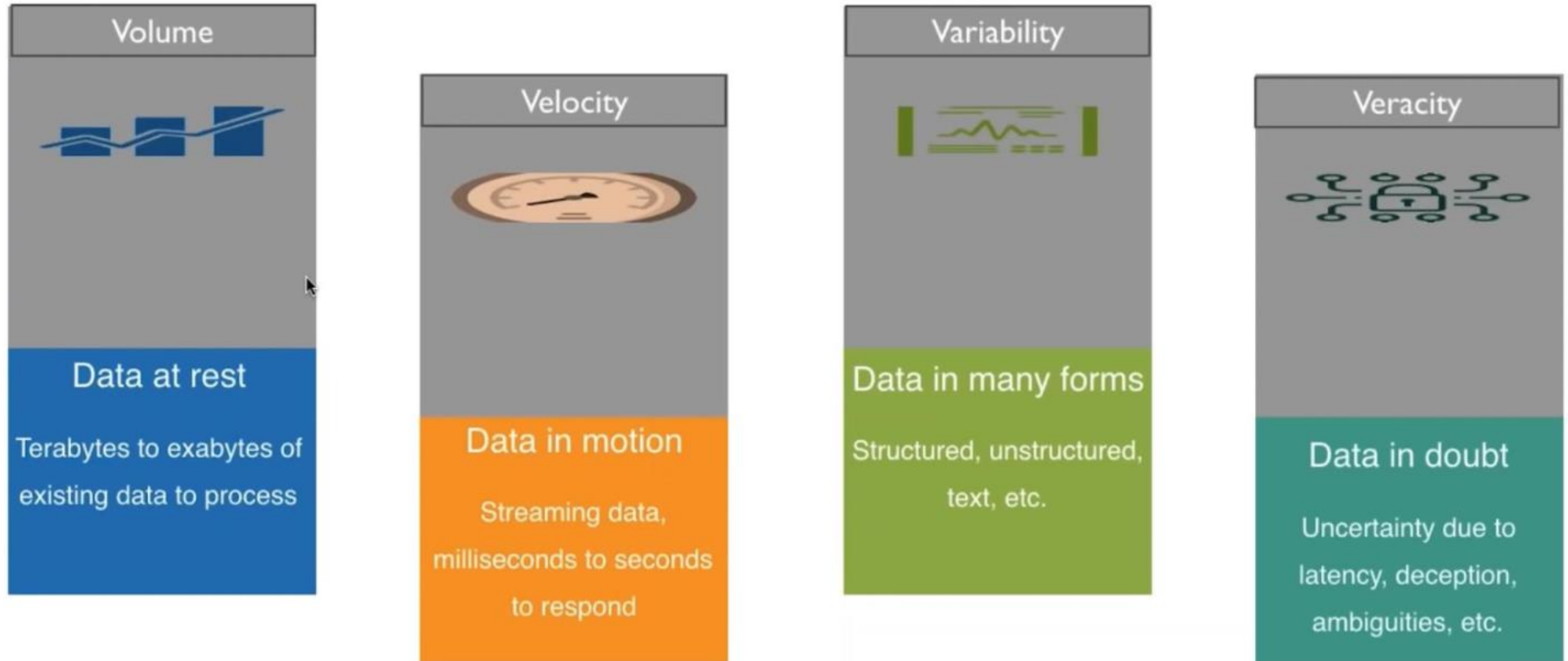
What is OLTP

- OLTP, or online transactional processing, enables the real-time execution of large numbers of database transactions by large numbers of people, typically over the internet.
- A database transaction is a change, insertion, deletion, or query of data in a database. OLTP systems (and the database transactions they enable) drive many of the financial transactions we make every day, including online banking and ATM transactions, e-commerce and in-store purchases, and hotel and airline bookings, to name a very few. In each of these cases, the database transaction also remains as a record of the corresponding financial transaction. OLTP can also drive non-financial database exchanges, including password changes and text messages.
- In OLTP, the common, defining characteristic of any database transaction is its *atomicity* (or indivisibility)—a transaction either succeeds as a whole or fails (or is canceled). It cannot remain in a pending or intermediate state.

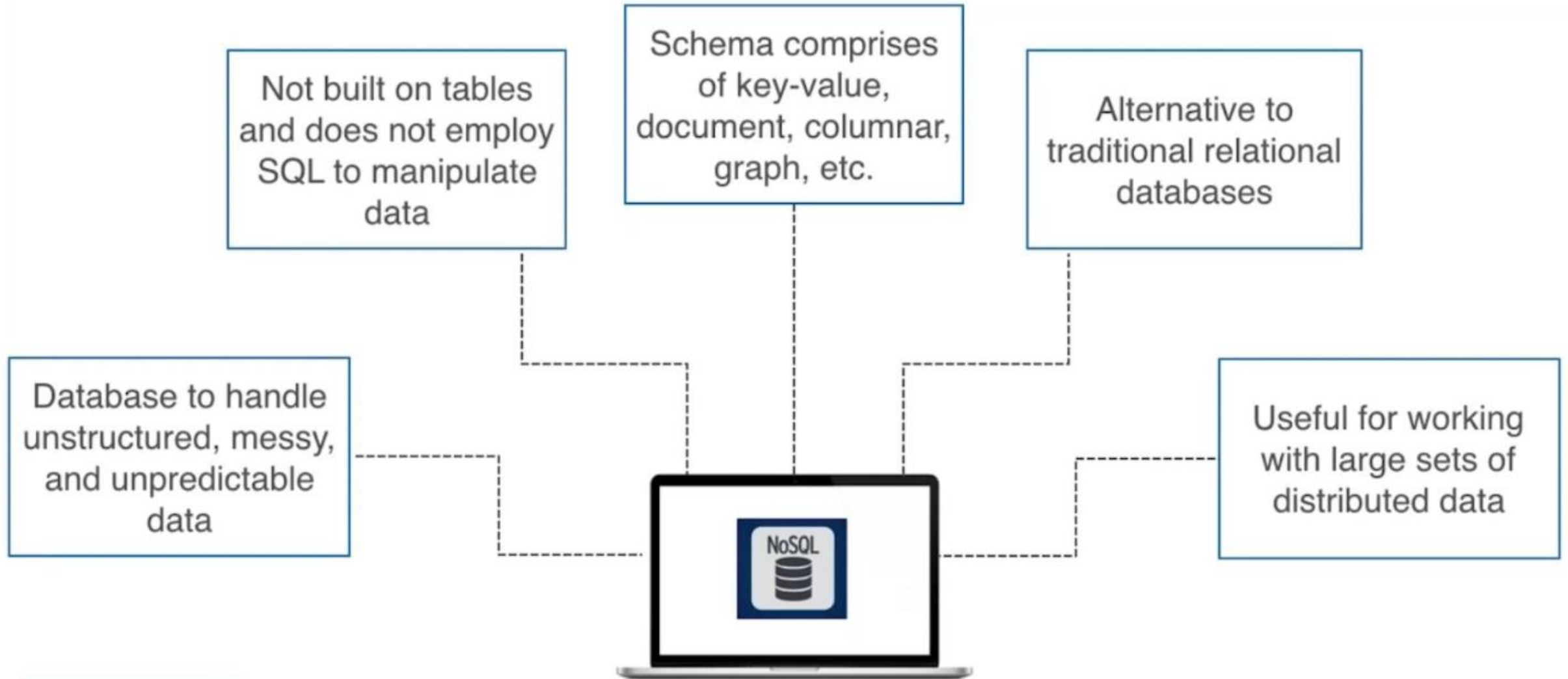
Examples of OLTP systems

- Since the inception of the internet and the e-commerce era, OLTP systems have grown ubiquitous. They're found in nearly every industry or vertical market and in many consumer-facing systems. Everyday examples of OLTP systems include the following:
- ATM machines (this is the classic, most often-cited example) and online banking applications
- Credit card payment processing (both online and in-store)
- Order entry (retail and back-office)
- Online bookings (ticketing, reservation systems, etc.)
- Record keeping (including health records, inventory control, production scheduling, claims processing, customer service ticketing, and many other applications)

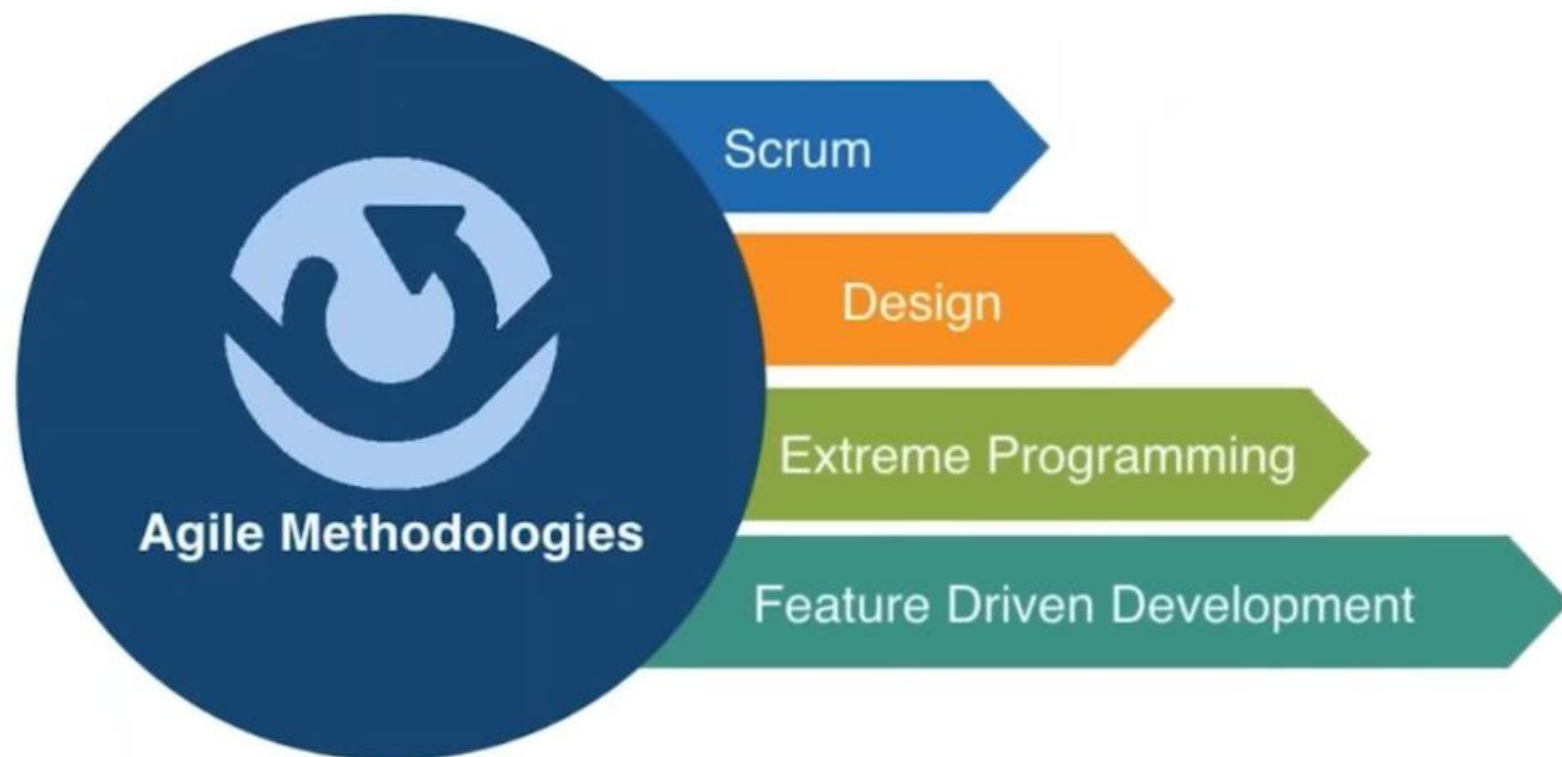
Advantages of NoSQL - 1



Advantages of NoSQL - 2



Advantages of NoSQL - 3



Categories Of NoSQL Databases

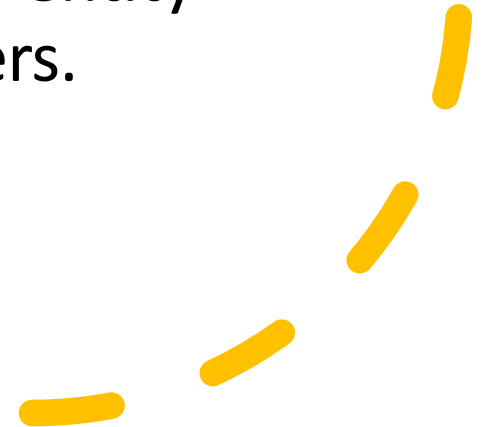
Document Base	Key Value Store	Graph Store	Wide Column Store
<ul style="list-style-type: none">• Pairs each key with a complex data structure known as a document• Contains many different key-value pairs, or key-array pairs, or even nested documents	<ul style="list-style-type: none">• Key-value stores are the simplest NoSQL databases.• Every single item in the database is stored as an attribute name (or "key"), together with its value	<ul style="list-style-type: none">• Used to store information about networks, such as social connections.• Graph stores include Neo4J and HyperGraphDB.	<ul style="list-style-type: none">• Cassandra and HBase are optimized for large dataset queries• Stores columns of data together, instead of rows.

What is a Graph Database?

Very simply, a graph database is a database designed to treat the relationships between data as equally important to the data itself.

It is intended to hold data without constricting it to a pre-defined model.

Instead, the data is stored like we first draw it out - showing how each individual entity connects with or is related to others.



Why Graph Databases?

We live in a connected world! There are no isolated pieces of information, but rich, connected domains all around us. Only a database that natively embraces relationships is able to store, process, and query connections efficiently. While other databases compute relationships at query time through expensive JOIN operations, a graph database stores connections alongside the data in the model.

Accessing nodes and relationships in a native graph database is an efficient, constant-time operation and allows you to quickly traverse millions of connections per second per core.

Independent of the total size of your dataset, graph databases excel at managing highly-connected data and complex queries. With only a pattern and a set of starting points, graph databases explore the neighboring data around those initial starting points — collecting and aggregating information from millions of nodes and relationships — and leaving any data outside the search perimeter untouched.

What is Neo4j?

Neo4j is an open-source, NoSQL, native graph database that provides an ACID-compliant transactional backend for your applications. Initial development began in 2003, but it has been publicly available since 2007. The source code, written in Java and Scala, is available for free on GitHub or as a user-friendly desktop application download. Neo4j has both a Community Edition and Enterprise Edition of the database. The Enterprise Edition includes all that Community Edition has to offer, plus extra enterprise requirements such as backups, clustering, and failover abilities.

Neo4j is referred to as a native graph database because it efficiently implements the property graph model down to the storage level. This means that the data is stored exactly as you whiteboard it, and the database uses pointers to navigate and traverse the graph. In contrast to graph processing or in-memory libraries, Neo4j also provides full database characteristics, including ACID transaction compliance, cluster support, and runtime failover - making it suitable to use graphs for data in production scenarios.

Some of the following particular features make Neo4j very popular among developers, architects, and DBAs:

- **Cypher**, a declarative query language similar to SQL, but optimized for graphs. Now used by other databases like SAP HANA Graph and Redis graph via the openCypher project.
- **Constant time traversals** in big graphs for both depth and breadth due to efficient representation of nodes and relationships. Enables scale-up to billions of nodes on moderate hardware.
- **Flexible** property graph schema that can adapt over time, making it possible to materialize and add new relationships later to shortcut and speed up the domain data when the business needs change.
- Drivers for popular programming languages, including Java, JavaScript, .NET, Python, and many more.

What is a Wide- column Database?

A wide-column database is a type of NoSQL database in which the names and format of the columns can vary across rows, even within the same table. Wide-column databases are also known as column family databases.

Because data is stored in columns, queries for a particular value in a column are very fast, as the entire column can be loaded and searched quickly. Related columns can be modeled as part of the same column family.

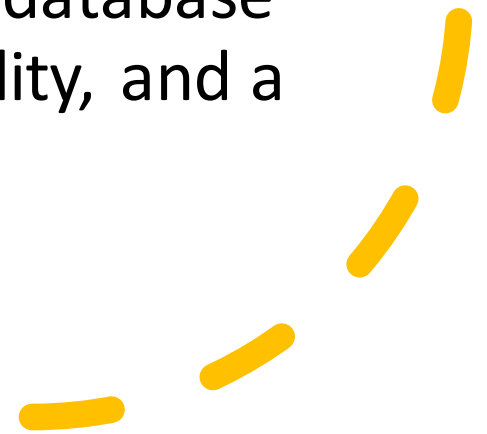


What is a Wide-column Database?

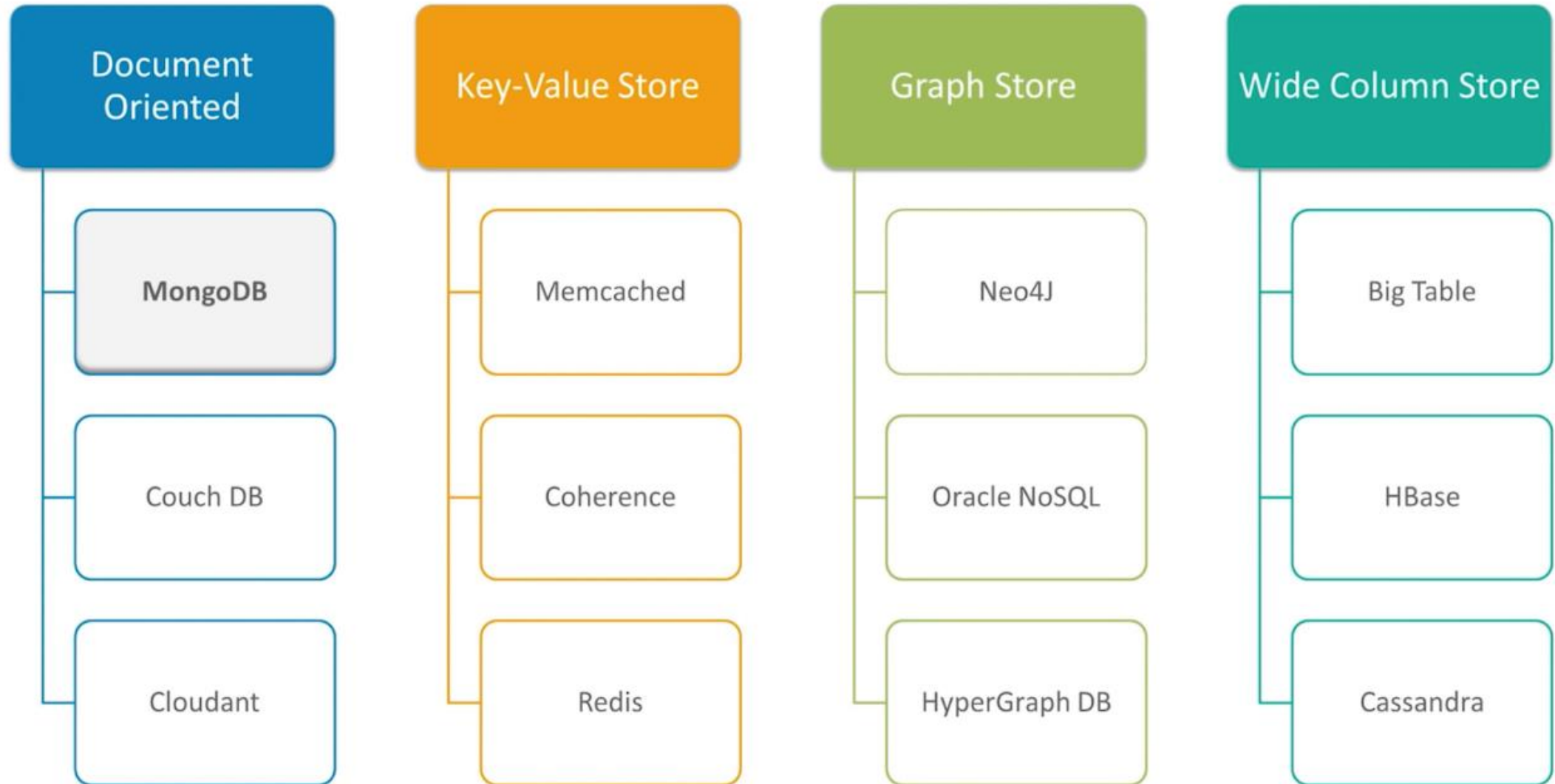
A wide-column database is a type of NoSQL database in which the names and format of the columns can vary across rows, even within the same table. Wide-column databases are also known as column family databases.

Because data is stored in columns, queries for a particular value in a column are very fast, as the entire column can be loaded and searched quickly. Related columns can be modeled as part of the same column family.

Benefits of a wide-column NoSQL database include speed of querying, scalability, and a flexible data model.



Types Of NoSQL Databases



NoSQL Database – Selection and Implementation





Introduction To MongoDB

Overview of MongoDB

- To avoid complex SQL queries with long processing time, you can shift to a NoSQL Database which is designed for ease of development and scaling i.e. MongoDB.
- MongoDB is an open source document database which is capable of handling big data.
- It works on the concept of collection and document.
- There is no concept of primary key and foreign key in the tables.
- It provides:



High Performance

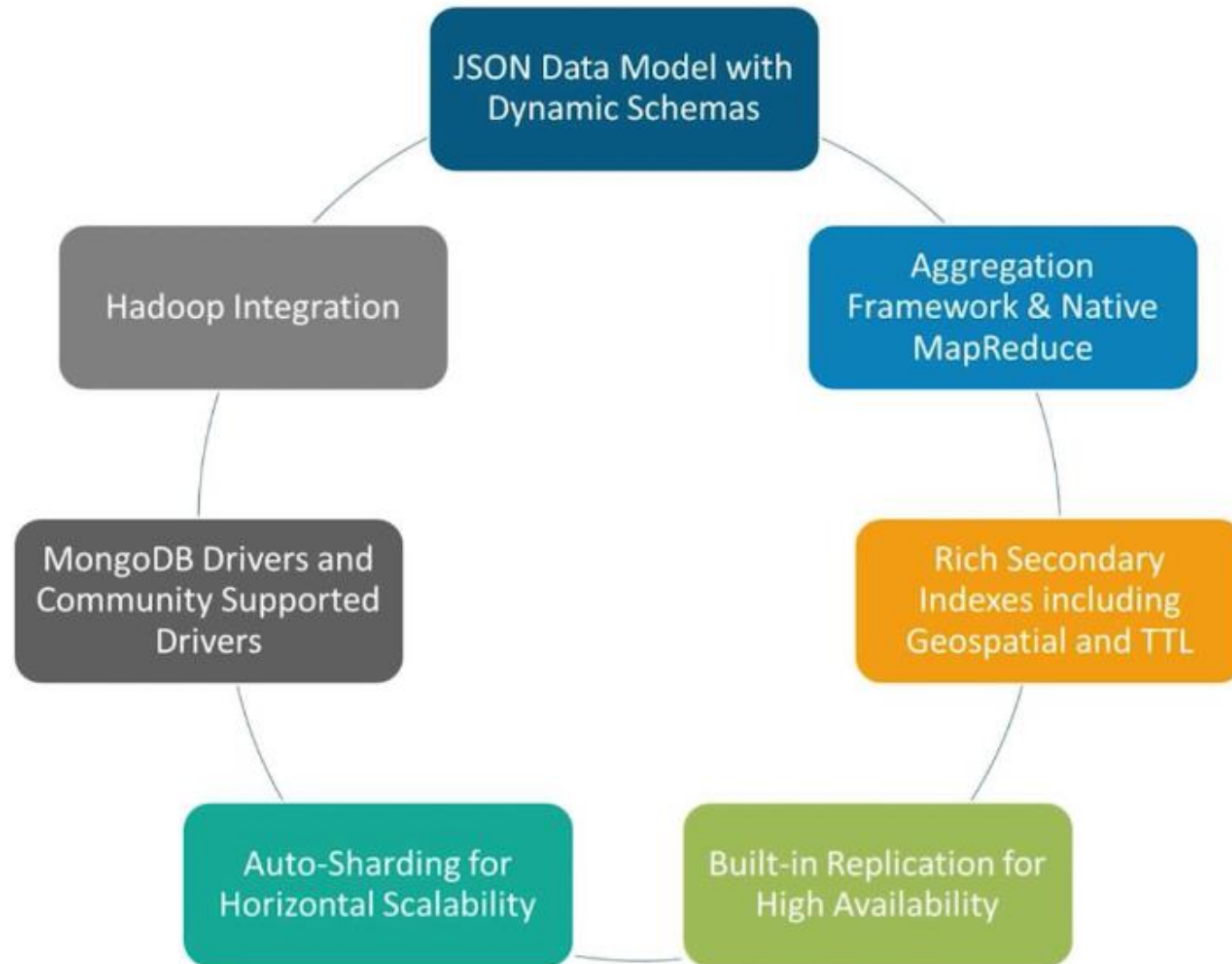


High Availability



Easy Scalability

Features Of MongoDB



Dynamic schemas

Development teams are constantly searching for new ways to quickly enhance applications and satisfy the rapid progression of customer needs.

The dynamic schema evolution in MongoDB enables such a reality through the power and flexibility of storing data in a JSON document format instead of in relational tables.

Notably, developers love the flexibility and schema-less nature of the JSON document format.



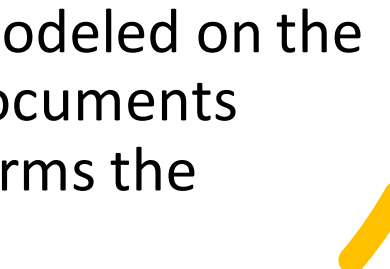
Aggregation Framework

Aggregation operations process data records and return computed results.

Aggregation operations group values from multiple documents together, and can perform a variety of operations on the grouped data to return a single result.

MongoDB provides three ways to perform aggregation: the aggregation pipeline, the map-reduce function, and single purpose aggregation methods.


MongoDB's aggregation framework is modeled on the concept of data processing pipelines. Documents enter a multi-stage pipeline that transforms the documents into an aggregated result.



Hadoop MapReduce

Hadoop MapReduce is a software framework for easily writing applications which process vast amounts of data (multi-terabyte data-sets) in-parallel on large clusters (thousands of nodes) of commodity hardware in a reliable, fault-tolerant manner.

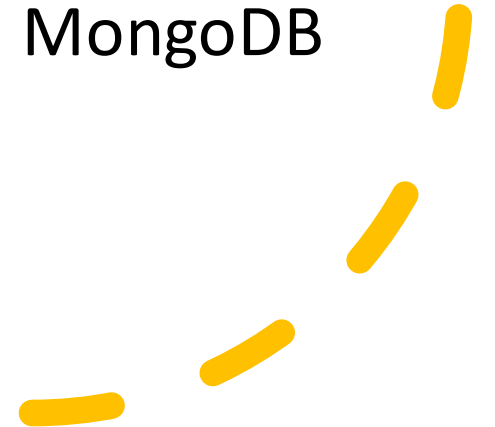
A MapReduce job usually splits the input data-set into independent chunks which are processed by the map tasks in a completely parallel manner. The framework sorts the outputs of the maps, which are then input to the reduce tasks. Typically both the input and the output of the job are stored in a file-system. The framework takes care of scheduling tasks, monitoring them and re-executes the failed tasks.



MongoDB Python Drivers

You can access MongoDB from your Python application using one of official MongoDB Python drivers.

- **PyMongo** is the recommended driver to work with MongoDB using Python.
- **Motor** is the recommended driver for when you need non-blocking access to MongoDB using Python.



Reasons to use MongoDB - 1

I. Speed To Develop

- a) Change streams to create powerful data pipelines
- b) Shard-aware secondary reads ensure data consistency from any secondary
- c) Fully expressive array updates allow you to perform complex array manipulations
- d) Retryable writes reduce error handling

2. Speed To Scale

- a) Operational simplicity making you 10x to 20x more productive
 - b) Data governance with JSON schema lets you precisely control document structures
 - c) Most advanced security control with comprehensive access controls, end-to-end encryption, and auditing
 - d) Offers greater network, memory, and storage efficiency than almost any other database, thus saving you time, money, and precious platform resources
-

Reasons to use MongoDB - II

3. Speed To Insight

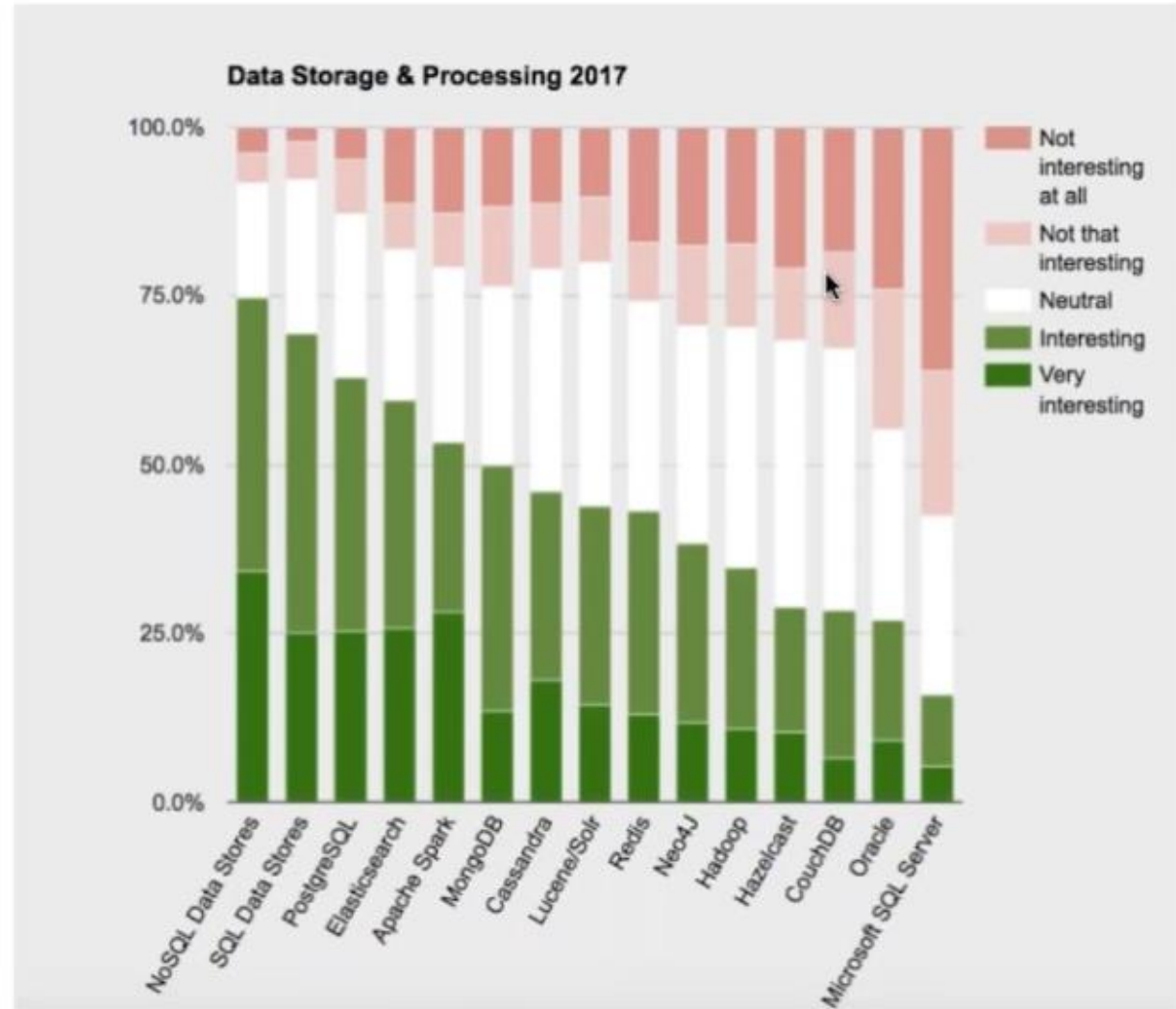
- a) Richer aggregation pipeline enables you to unlock new classes of analytics with less client-side code
- b) MongoDB Connector for BI provides integration with all leading SQL-based BI platforms such as Tableau, QlikView, SAP Business Objects, etc.
- c) R driver for MongoDB provides you with idiomatic, native language access to MongoDB to simplify and accelerate the speed of statistical analysis and data mining

4. Run Anywhere

- a) Globally distributed MongoDB service through MongoDB Atlas
 - b) Best way to run MongoDB is, in the public cloud
 - c) Automatic storage scaling reduces DevOps overhead by provisioning additional cluster storage capacity when it's needed
-

Latest Market Trends in MongoDB

- NoSQL Market is expected to garner \$4.2 billion by 2020, registering a CAGR of 35.1% during the forecast period 2014-2020.
- MongoDB is the leading NoSQL database, with significant adoption among the Fortune 500 and Global 500.
- Software developers favor MongoDB's NoSQL database over most of the other databases, according to a recent survey by Stack Overflow of 64,000 developers.



Few Customers of MongoDB



MetLife



Google



facebook



SAP



NOKIA



Forbes



AstraZeneca

Sectors using MongoDB - 1



Surveillance Data Aggregation

Crime Data Management and Analytics

Citizen Engagement Platform

Program Data and Healthcare Record
Management

Risk Analytics and Reporting

Reference Data & Market Data Management

Portfolio Management & Order Capture

Time Series Data



Sectors using MongoDB - II



360-Degree Patient View

Population Management for At-Risk
Demographics

Lab Data Management and Analytics

Mobile Apps for Doctors and Nurses

Content Management and Delivery

User Data and Digital Asset Management

Mobile and Social Apps

Content Archiving



Sectors using MongoDB - III



Rich Product Catalogs

Customer Data Management

New Services and Digital Coupons

Real-Time Price Optimization

Consumer Cloud and Product Catalog

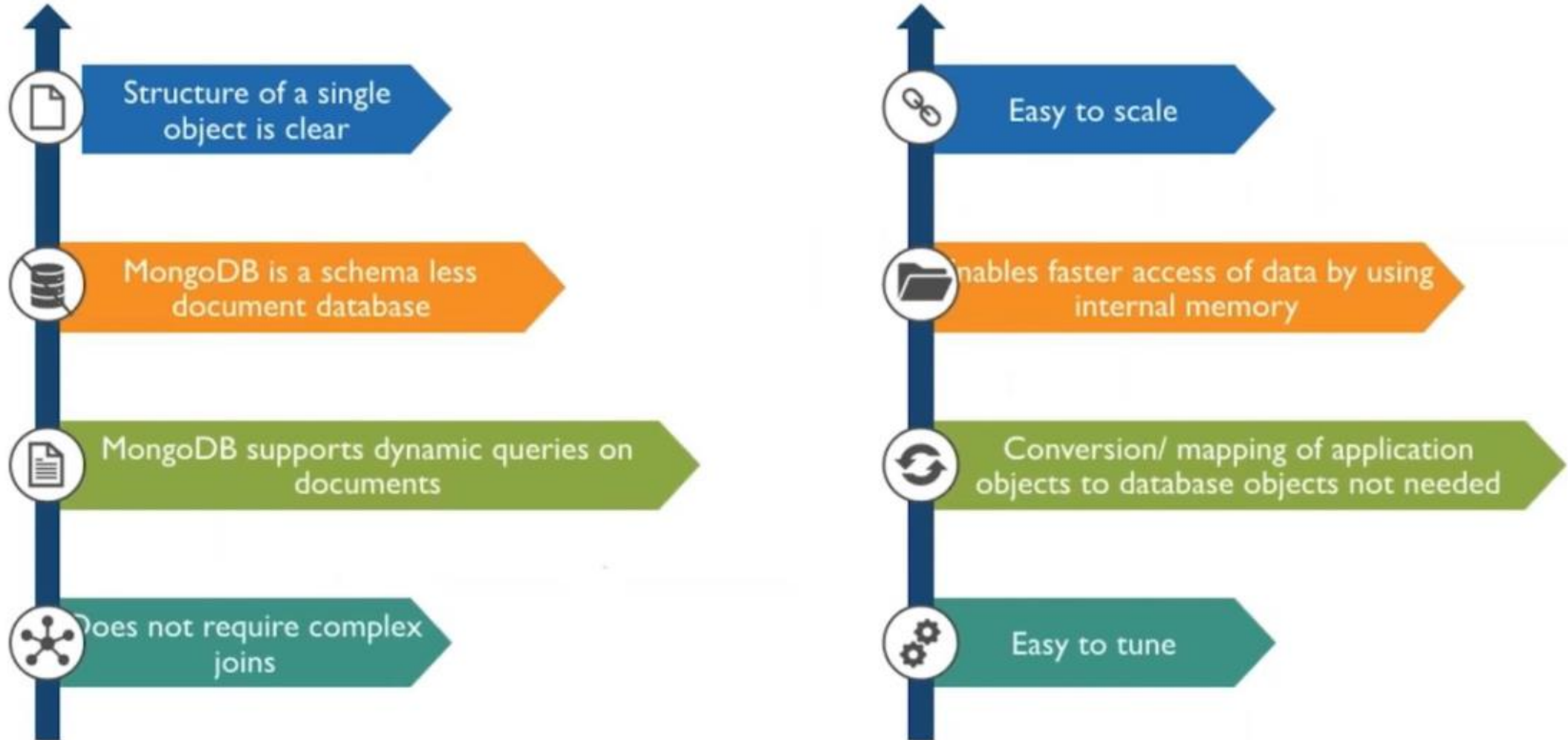
Customer Service Improvement

Machine-to-Machine (M2M) Platform

Real-Time Network Analysis and Optimization



Advantages of using MongoDB





MongoDB Tools And Terminologies

Mongo Database

- Database is a physical container for collections
- Each database gets its own set of files on the file system
- A single MongoDB server has multiple databases
- Mongod is the primary resource for the MongoDB system
- It handles data requests, manages data format, and performs background management operations

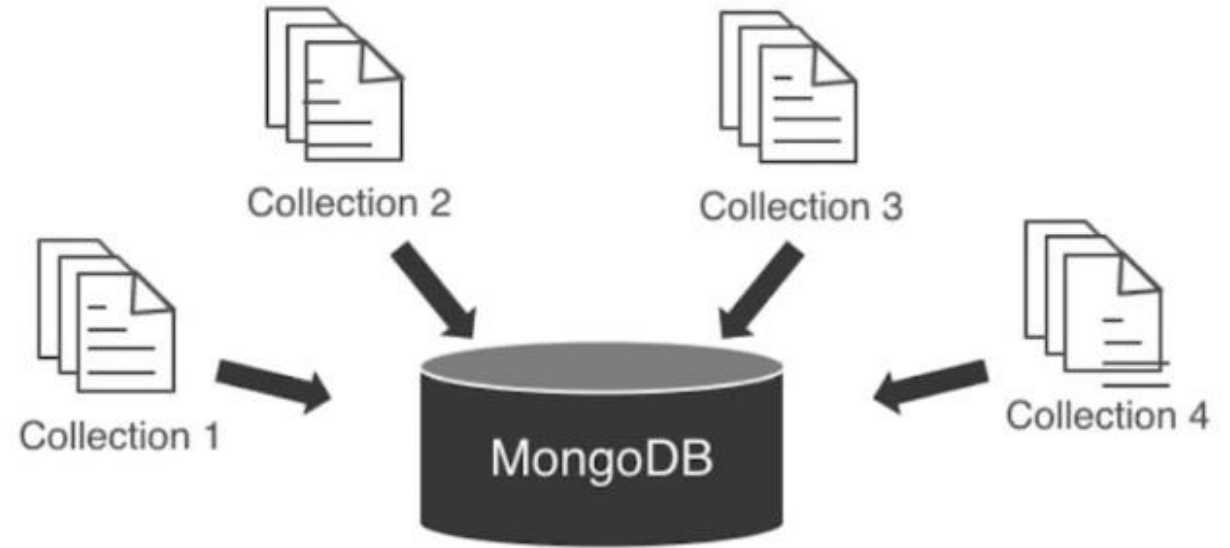


mongoDB®



MongoDB Collection

- Collection is a group of similar or related purpose MongoDB documents within a single database
- MongoDB collection is equivalent to a RDBMS table
- Collections do not enforce a schema
- Documents within a collection can have different fields



RDBMS Terminology and MongoDB

RDBMS	MongoDB
Database	Database
Table	Collection
Tuple/Row	Document
Column/ Attribute/ Variable	Field
Table Join	Embedded Documents
Database Server and Client	
Primary Key	Primary Key (Default key _id provided by mongodb itself)
Mysqld/Oracle	mongod
mysql/sqlplus	mongo

Mongo DB tools

Core Processes

- mongod
- mongos
- mongo

Data Import and Export

- mongoimport
- mongoexport

Diagnostic Tools

- mongostat
- mongotop
- mongosniff
- mongoperf

Windows Services

- mongod.exe
- mongos.exe

Binary Import and Export

- mongodump
- mongorestore
- bsondump
- mongooplog

GridFS

- mongofiles



Introduction To JSON And BSON

JavaScript Object Notation (JSON)

JSON is a

- Lightweight, text-based, open standard format
- Syntax for storing and exchanging data
- Text, written with JavaScript object notation

Conventions used by JSON include C, C++, Java, Python, Perl, so on.



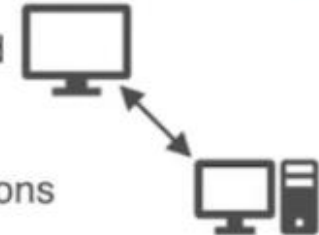
Web services and APIs use JSON format to provide public data



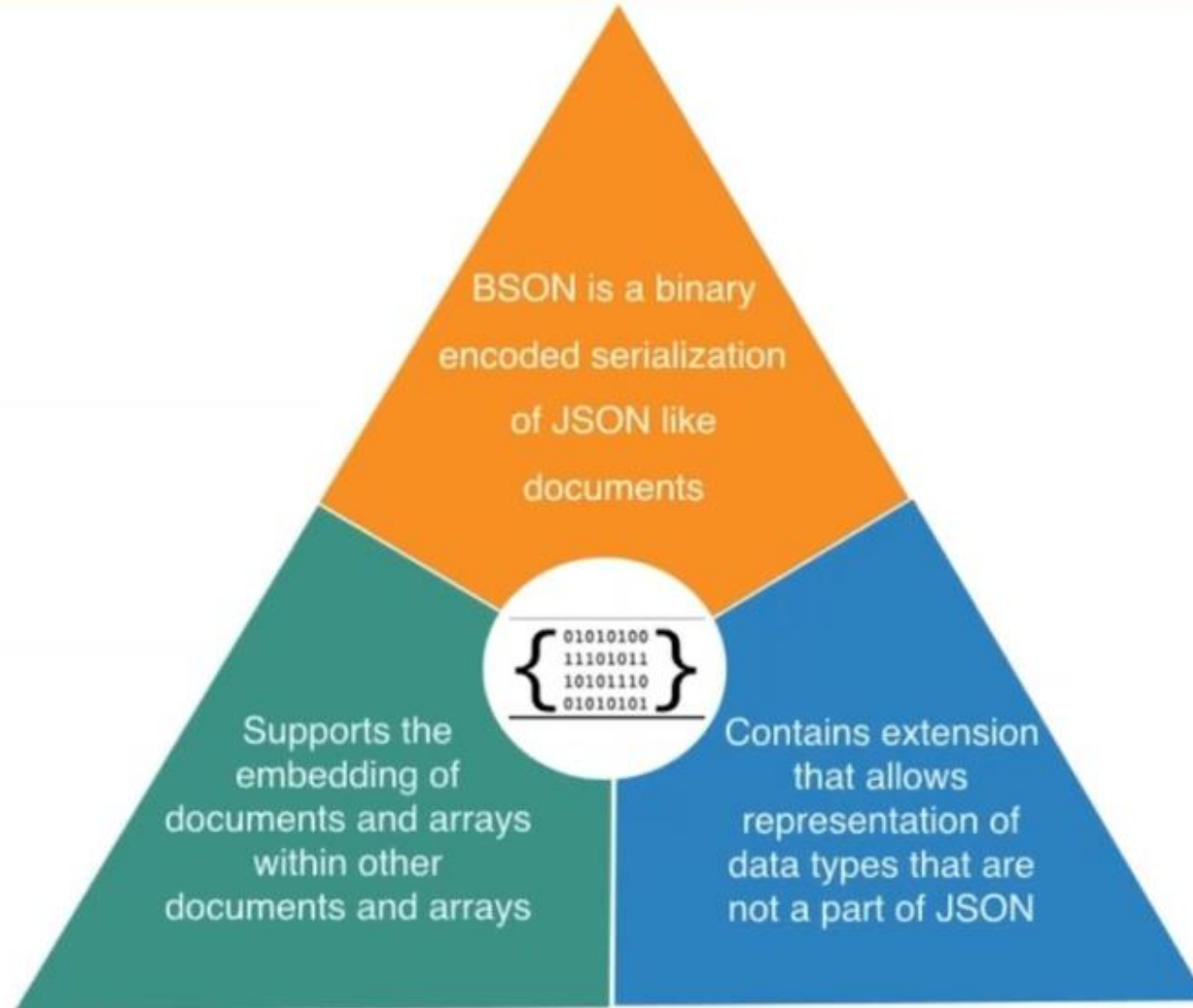
It is used while writing JavaScript based applications that include browser extensions and websites



It is primarily used to transmit data between a server and web applications



Binary JSON (BSON)



Characteristics of BSON

BSON is designed to have the following characteristics:

Lightweight

Optimizing spatial overhead is important for any data representation format, specifically when used over the network

Traversable

For primary data representation, BSON can be used, which traverses data easily

Efficient

Encoding data to BSON and decoding from BSON can be performed quickly in most languages due to the use of 'C' data types

JSON vs. BSON

JSON:

`{"hello": "world"}`



BSON:

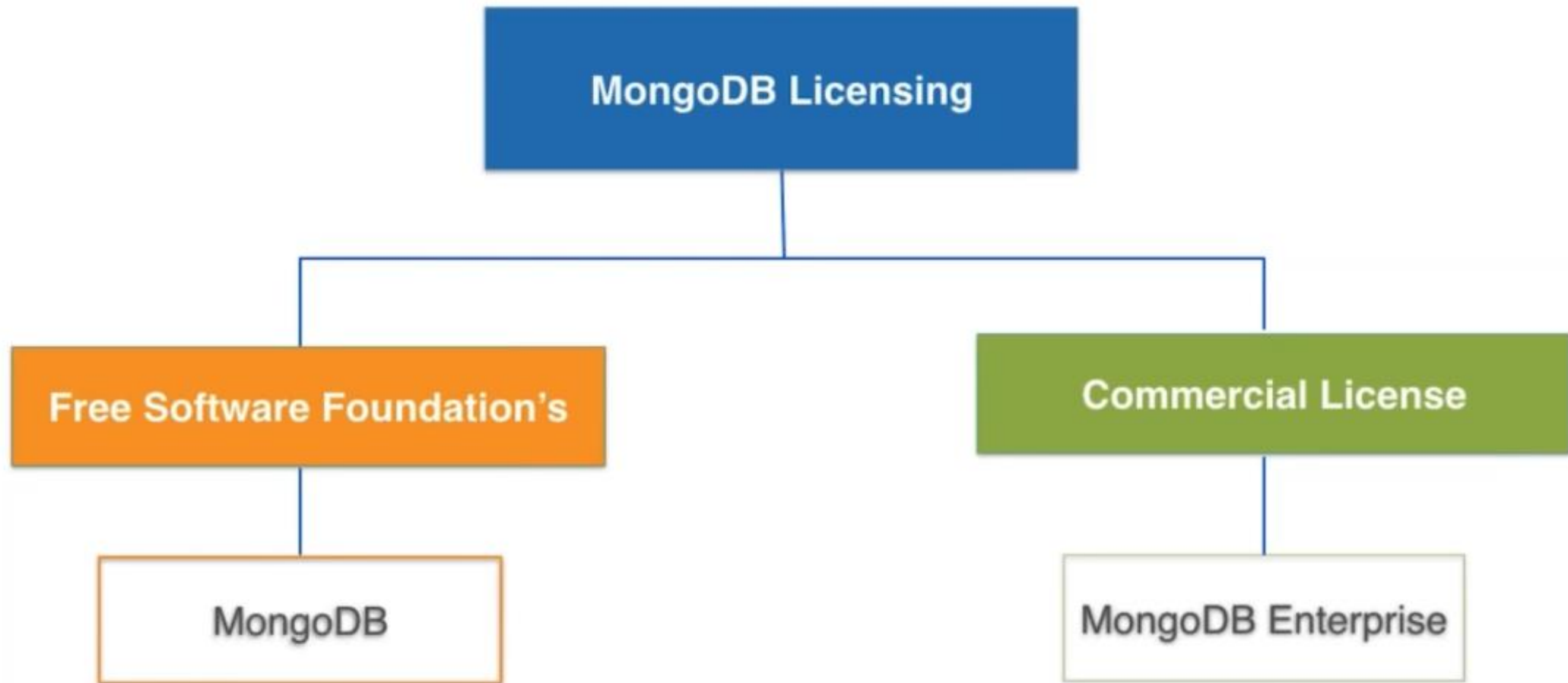
`"\x16\x00\x00\x00\x02 hello\x00
\x06\x00\x00\x00 world\x00\x00"`

`{"BSON": ["awesome", 5.05, 1986]}`



`"1\x00\x00\x00\x04BSON\x00&\x00\x00\x00\x0
20\x00\x08\x00\x00\x00awesome\x00\x011\x00
333333\x14@\x102\x00\xc2\x07\x00\x00\x00\x0
0"`

MongoDB Licensing



MongoDB Enterprise

- MongoDB Enterprise is the commercial edition of MongoDB that provides enterprise-grade capabilities
- MongoDB Enterprise includes advanced **security features**, **management tools**, **software integration** and **certification**
- These value-added capabilities are not included in the open-source edition of MongoDB



MongoDB Enterprise Includes

- Advanced Security
- Management Support
- Certified OS Support
- Enterprise Software Integration
- On-demand Training

A teal parallelogram is positioned in the upper left corner. A green parallelogram and a yellow triangle are located in the bottom left corner.

THANK YOU!

