MongoDB

* Non-SQL database
* Open-source
* Cross-platform
* Document-Oriented Database written in c++.
* Distributed-Database
* Non-relational database system that uses documents instead of tables or rows for data storage .
* This data model makes it possible to manipulate related data in a single database operation.
* MongoDB documents are a use JSON-like documents and files , and they are JS supported.
* MongoDB is considered schema-less, as it doesn't require a predefined database schema .
* MongoDB uses flexible key-value pairs called documents to store data.
* Greater reliability and Efficiency.
* High availability , Horizontal scaling and geographic distribution.
* Faster than MySQL.

**All the modern applications require big data , fast features development , flexible deployment and the older database systems not competent enough SO THE MONGODB was NEEDED.**

Why mongoDB is needed :

* Scalability
* Performance
* High Availability
* Key points of MongoDB
* Develop Faster
* Deploy Easier
* Scale Bigger
* Scaling from single server deployments to large , complex multi-site architectures.

Features

* Support ad hoc queries
* In MongoDB , you can search by field , range query and it also supports regular expression searches.
* Indexing
* You can index any field in a document .
* Replication
* Supports Master Slave replication - Where a master can perform reads and writes and a slave copies data from master and can only be used for reads or backup
* Duplication of Data
* MongoDB can run over multiple servers. Data is duplicated to keep the system up .
* Load Balancing
* Automatic Load balancing
* Supports map reduce and aggregation tools.
* Uses JS instead of Procedures.
* Schema-Less
* High-Performance
* Store files of any size easily
* Easy to administer in case of failures.
* Easy to Use
* Light weight
* Faster than RDBMS

Used where

* Big and complex data
* Mobile and Social infrastructure
* Content management and delivery
* User data management
* Data hub

MongoDB is a good choice if:

* You want high data availability along with automatic and instant data recovery.
* You are working with an unstable schema and want to lower the cost of schema migration…
* Your services are cloud-based.
* You want to speed up development.

MongoDB can be the right choice if you are working with real-time analytics, mobile applications, internet of things, etc., where you may have structure or unstructured data that has the potential for rapid growth.

Advantages and Disadvantages

* **MongoDB is schema less**. It is a document database in which one collection holds different documents.
* There may be **difference between number of fields, content and size of the document** from one to other.
* **Structure of a single object is clear** in MongoDB.
* There are **no complex joins** in MongoDB.
* MongoDB provides the **facility of deep query** because it supports a powerful dynamic query on documents.
* It is very **easy to scale**.
* It **uses internal memory for storing working sets** and this is the reason of its fast access.

Limitations :

* MongoDB doesn't support joins
* Data redundancy .
* Documents have limit of 16 MB.
* Doesn't support ACID so complex transactions can get complicated.
* Doesn't support Stored Procedures .

Datatypes

* 1. String
  2. Integer
  3. Boolean
  4. Double
  5. Min/Max Keys - Compare value against the lowest and highest bson elements.
  6. Arrays - list or multiple values into a single keys
  7. Object - for embedded documents
  8. Null -Used to store null values.
  9. Symbol -generally used for languages that use a specific type.
  10. Date- stores current date and time in unix time format . [date/month/year]

|  |  |
| --- | --- |
| **SQL Terms** | **MongoDB Terms** |
| database | Database |
| table | Collection |
| row | document or BSON document |
| column | field |
| index | index |
| table joins | $lookup, embedded document |
| primary key | primary key |
| In SQL, we can specify any unique column or column combination as the primary key. | In MongoDB, we don't need to set the primary key. The \_id field is automatically set to the primary key. |
| aggregation | aggregation pipeline |
| SELECT INTO NEW\_TABLE | $out |
| MERGE INTO TABLE | $merge |
| transactions | transactions |