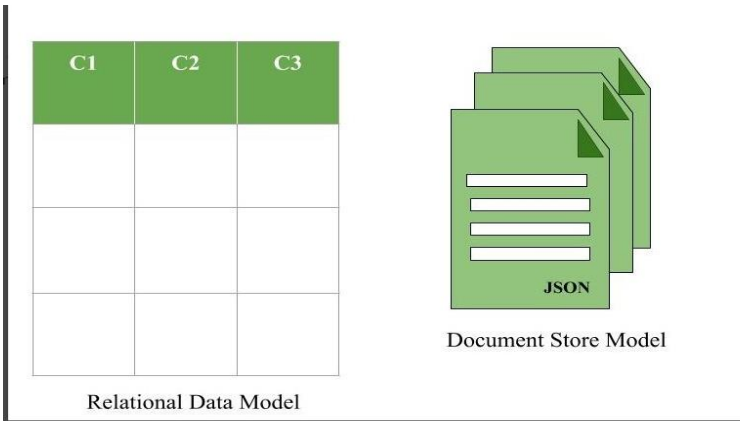
How do Document Databases Work?

A document database has information retrieved or stored in the form of a document or other words semi-structured database. Since they are non-relational, so they are often referred to as NoSQL data.

The document database fetches and accumulates data in forms of key-value pairs but here, the values are called as Documents. A document can be stated as a complex data structure. Document here can be a form of text, arrays, strings, JSON, XML, or any such format. The use of nested documents is also very common. It is very effective as most of the data created is usually in the form of JSON and is unstructured.



Advantages

- They offer higher productivity and faster evolution for a developer.
- Document databases are easier to store and query data in a database for a developers by using the same document-model format they use in their application code.
- For use cases such as catalogs, user profiles, and content management systems where each document is unique and evolves over time, it is much better to use document model.
- Document databases provide flexible indexing, powerful ad hoc queries, and analytics over collections of documents.

Disadvantages:

- Handling multiple documents is challenging
- Aggregation operations may not work accurately.
- Comparing MongoDB to other databases

Comparing MongoDB to other Databases

With so many database management solutions currently available, it can be hard to choose the right solution for your enterprise. Here are some common solution comparisons and best use cases that can help you decide.

MongoDB vs. MySQL

MySQL (link resides outside IBM.com) uses a structured query language to access stored data. In this format, schemas are used to create database structures, utilizing tables as a way to standardize data types so that values are searchable and can be queried properly. A mature solution, MySQL is useful for a variety of situations including website databases, applications and commercial product management.

Because of its rigid nature, MySQL is preferable to MongoDB when data integrity and isolation are essential, such as when managing transactional data. But MongoDB's less-restrictive format and higher performance make it a better choice, particularly when availability and speed are primary concerns.

MongoDB vs. Cassandra

While Cassandra (link resides outside IBM.com) and MongoDB are both considered NoSQL databases, they have different strengths. Cassandra uses a traditional table structure with rows and columns, which enables users to maintain uniformity and durability when formatting data before it's compiled.

Cassandra can offer an easier transition for enterprises looking for a NoSQL solution because it has a syntax similar to SQL; it also reliably handles deployment and replication without a lot of configuration. However, it can't match MongoDB's flexibility for handling structured and unstructured data sets or its performance and reliability for mission-critical cloud applications.

SQL to MongoDB Mapping Chart

Terminology and Concepts

The following table presents the various SQL terminology and concepts and the corresponding MongoDB terminology and concepts.

SQL Terms/Concepts	MongoDB Terms/Concepts
database	database
table	collection
row	document or BSON document
column	field
index	index
table joins	$lookup, embedded documents
primary key Specify any unique column or column combination as primary key.	primary key In MongoDB, the primary key is automatically set to the _id field.
aggregation (e.g. group by)	aggregation pipeline See the SQL to Aggregation Mapping Chart.
SELECT INTO NEW_TABLE	$out See the SQL to Aggregation Mapping Chart.
MERGE INTO TABLE	$merge See the SQL to Aggregation Mapping Chart.
UNION ALL	$unionWith
transactions	transactions

MONGO DB : INTRODUCTION

MongoDB, the most popular NoSQL database, is an open-source document-oriented database. The term ‘NoSQL’ means ‘non-relational’. It means that MongoDB isn’t based on the table-like relational database structure but provides an altogether different mechanism for storage and retrieval of data. This format of storage is called BSON (similar to JSON format).

A simple MongoDB document Structure:

```
{
  title: 'chrome',
  by: 'nidhiii,
  url: 'https://www.chrome.org,
  type: 'NoSQL'
}
```

SQL databases store data in tabular format. This data is stored in a predefined data model which is not very much flexible for today’s real-world highly growing applications. **Modern applications are more networked, social and interactive than ever**. Applications are storing more and more data and are accessing it at higher rates. Relational Database Management System(RDBMS) is **not the correct choice when it comes to handling big data by the virtue of their design since they are not horizontally scalable**. If the database runs on a single server, then it will reach a scaling limit. NoSQL databases are more scalable and provide superior performance. MongoDB is such a NoSQL database that scales by adding more and more servers and increases productivity with its flexible document model.

History of MONGODB:

- The American software company 10gen began developing MongoDB in 2007 as a component of a planned platform-as-a-service product. In 2009, the company shifted to an open-source development model and began offering commercial support and other services. In 2013, 10gen changed its name to MongoDB Inc.[5]
- On October 20, 2017, MongoDB became a publicly traded company, listed on NASDAQ as MDB with an IPO price of $24 per share.[6]
- On November 8, 2018 with the stable release 4.0.4, the software's license changed from AGPL 3.0 to SSPL.[7][8]
- On October 30, 2019, MongoDB teamed with Alibaba Cloud to offer Alibaba Cloud customers a MongoDB-as-a-service solution. Customers can use the managed offering from Alibaba's global data centers.

How MongoDB works ?

MongoDB is an open-source document-oriented database. It is used to store a larger amount of data and also allows you to work with that data. MongoDB is not based on the table-like relational database structure but provides an altogether different mechanism for storage and retrieval of data, that’s why known as NoSQL database. Here, the term ‘NoSQL’ means ‘non-relational’. The format of storage is called BSON (similar to JSON format).

Now, let’s see how actually this MongoDB works?

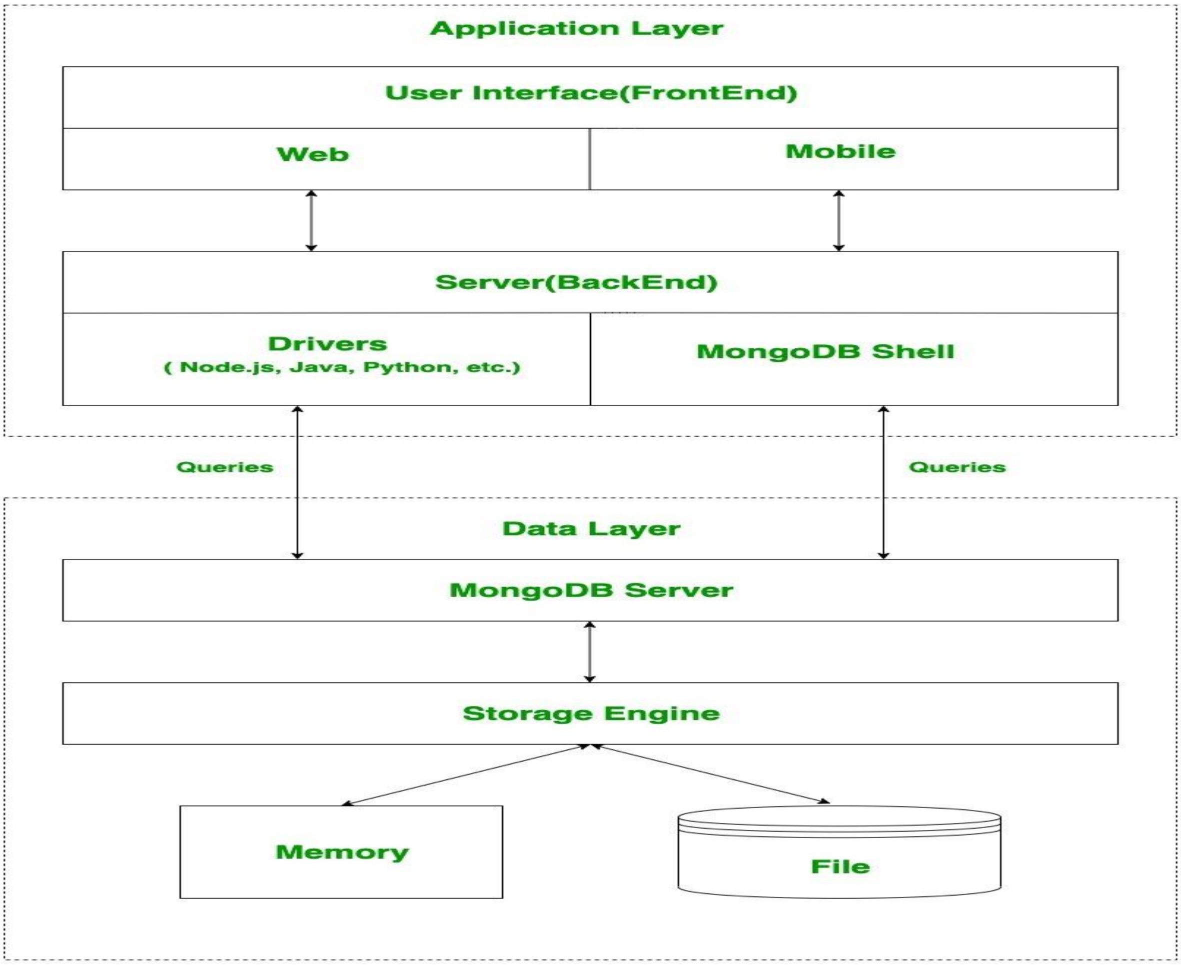
But before proceeding to its working, first, let’s discuss some important parts of MongoDB –

- **Drivers:** Drivers are present on your server that are used to communicate with MongoDB. The drivers support by the MongoDB are C, C++, C#, and .Net, Go, Java, Node.js, Perl, PHP, Python, Motor, Ruby, Scala, Swift, Mongoid.
- **MongoDB Shell:** MongoDB Shell or mongo shell is an interactive JavaScript interface for MongoDB. It is used for queries, data updates, and it also performs administrative operations.

- **Storage Engine:** It is an important part of MongoDB which is generally used to manage how data is stored in the memory and on the disk. MongoDB can have multiple search engines. You are allowed to use your own search engine and if you don't want to use your own search engine you can use the default search engine, known as *WiredTiger Storage Engine* which is an excellent storage engine, it efficiently works with your data like reading, writing, etc.

Working of MongoDB –

The following image shows how the MongoDB works:



MongoDB work in two layers –

- **Application Layer** and
- **Data layer**

Application Layer is also known as the **Final Abstraction Layer**, it has two-parts, first is a **Frontend (User Interface)** and the second is **Backend (server)**. The frontend is the place where the user uses MongoDB with the help of a Web or Mobile. This web and mobile include web pages, mobile applications, android default applications, IOS applications, etc. The backend contains a server which is used to perform server-side logic and also contain drivers or mongo shell to interact with MongoDB server with the help of queries.

These queries are sent to the MongoDB server present in the **Data Layer**. Now, the MongoDB server receives the queries and passes the received queries to the storage engine. MongoDB server itself does not directly read or write the data to the files or disk or memory. After passing the received queries to the storage engine, the storage engine is responsible to read or write the data in the files or memory basically it manages the data.

Features of MongoDB:

- **Document Oriented**: MongoDB stores the main subject in the minimal number of documents and not by breaking it up into multiple relational structures like RDBMS. For example, it stores all the information of a computer in a single document called Computer and not in distinct relational structures like CPU, RAM, Hard disk, etc.
- **Indexing**: Without indexing, a database would have to scan every document of a collection to select those that match the query which would be inefficient. So, for efficient searching Indexing is a must and MongoDB uses it to process huge volumes of data in very less time.
- **Scalability**: MongoDB scales horizontally using sharding (partitioning data across various servers). Data is partitioned into data chunks using the shard key, and these data chunks are evenly distributed across shards that reside across many physical servers. Also, new machines can be added to a running database.
- **Replication and High Availability**: MongoDB increases the data availability with multiple copies of data on different servers. By providing redundancy, it protects the database from hardware failures. If one server goes down, the data can be retrieved easily from other active servers which also had the data stored on them.
- **Aggregation**: Aggregation operations process data records and return the computed results. It is similar to the GROUPBY clause in SQL. A few aggregation expressions are sum, avg, min, max, etc

Where do we use MongoDB?

MongoDB is preferred over RDBMS in the following scenarios:

- **Big Data**: If you have huge amount of data to be stored in tables, think of MongoDB before RDBMS databases. MongoDB has built-in solution for partitioning and sharding your database.
- **Unstable Schema**: Adding a new column in RDBMS is hard whereas MongoDB is schema-less. Adding a new field does not effect old documents and will be very easy.
- **Distributed data** Since multiple copies of data are stored across different servers, recovery of data is instant and safe even if there is a hardware failure.

Language Support by MongoDB:

MongoDB currently provides official driver support for all popular programming languages like C, C++, Rust, C#, Java, Node.js, Perl, PHP, Python, Ruby, Scala, Go, and Erlang.

MongoDB Query API

The MongoDB Query API is the way you will interact with your data.

The MongoDB Query API can be used two ways:

- CRUD Operations
- Aggregation Pipelines

MongoDB Query API Uses

You can use the MongoDB Query API to perform:

- Adhoc queries with mongosh, Compass, VS Code, or a MongoDB driver for the programming language you use.
- Data transformations using aggregation pipelines.
- Document join support to combine data from different collections.
- Graph and geospatial queries.
- Full-text search.
- Indexing to improve MongoDB query performance.
- Time series analysis.