Introduction to MongoDB:

The Introduction to MongoDB course guides you through the foundational skills and knowledge you need to get started with MongoDB. This includes connecting to a MongoDB database, how to conduct simple CRUD operations, and key topics such as aggregation, indexing, data modeling, and transactions.

Brief History of MongoDB

MongoDB was developed by Eliot Horowitz and Dwight Merriman in the year 2007, when they experienced some scalability issues with the relational database while developing enterprise web applications at their company DoubleClick. According to Dwight Merriman, one of the developers of MongoDB, this name of the database was derived from the word *humongous* to support the idea of processing large amount of data.

In 2009, MongoDB was made as an open source project, while the company offered commercial support services. Many companies started using MongoDB for its amazing features. The New York Times newspaper used MongoDB to build a web based application to submit the photos. In 2013, the company was officially named to MongoDB Inc.

Key Features of MongoDB

Apart from most of the NoSQL default features, MongoDB does bring in some more, very important and useful features :

1. MongoDB provides high performance. Input/Output operations are lesser than relational databases due to support of embedded documents(data models) and Select queries are also faster as Indexes in MongoDB supports faster queries.
2. MongoDB has a rich Query Language, supporting all the major CRUD operations. The Query Language also provides good Text Search and Aggregation features.
3. Auto Replication feature of MongoDB leads to High Availability. It provides an automatic failover mechanism, as data is restored through backup(replica) copy if server fails.

4. Sharding is a major feature of MongoDB. Horizontal Scalability is possible due to sharding.
5. MongoDB supports multiple Storage Engines. When we save data in form of documents(NoSQL) or tables(RDBMS) who saves the data? It's the Storage Engine. Storage Engines manages how data is saved in memory and on disk.

MongoDB is a cross-platform, document oriented database that provides, high performance, high availability, and easy scalability. MongoDB works on concept of collection and document.

Database: Database is a physical container for collections. Each database gets its own set of files on the file system. A single MongoDB server typically has multiple databases.

Collection: Collection is a group of MongoDB documents. It is the equivalent of an RDBMS table. A collection exists within a single database. Collections do not enforce a schema. Documents within a collection can have different fields. Typically, all documents in a collection are of similar or related purpose.

Document: A document is a set of key-value pairs. Documents have dynamic schema. Dynamic schema means that documents in the same collection do not need to have the same set of fields or structure, and common fields in a collection's documents may hold different types of data. The following table shows the relationship of RDBMS terminology with MongoDB.

Why Use MongoDB?

· Document Oriented Storage: Data is stored in the form of JSON style documents.

· Index on any attribute

· Replication and high availability

· Auto-sharding

· Rich queries

· Fast in-place updates

Where to Use MongoDB?

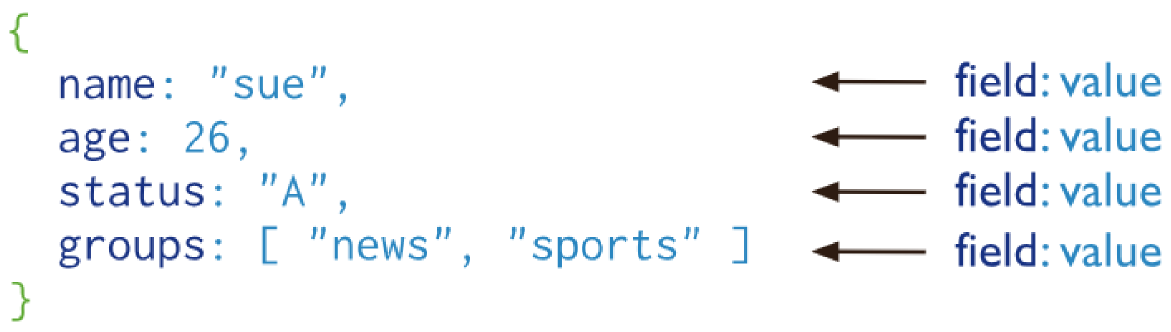
· Big Data

· Content Management and Delivery

· Mobile and Social Infrastructure

Document Database

A record in MongoDB is a document, which is a data structure composed of field and value pairs. MongoDB documents are similar to JSON objects. The values of fields may include other documents, arrays, and arrays of documents.



```
{
   name: "sue",
   age: 26,
   status: "A",
   groups: [ "news", "sports" ]
}
```

The advantages of using documents are:

- Documents correspond to native data types in many programming languages.
- Embedded documents and arrays reduce need for expensive joins.
- Dynamic schema supports fluent polymorphism.

Data Model Design

MongoDB provides two types of data models: — Embedded data model and Normalized data model. Based on the requirement, you can use either of the models while preparing your document.

Embedded Data Model

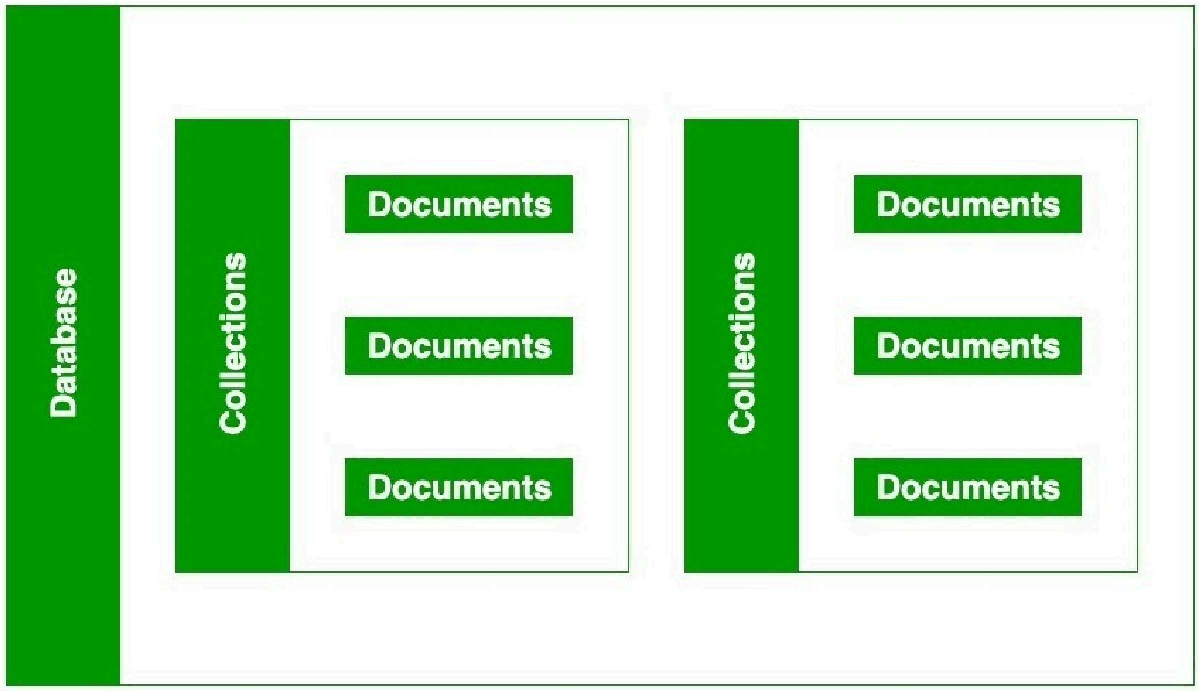
In this model, you can have (embed) all the related data in a single document, it is also known as de-normalized data model.

For example, assume we are getting the details of employees in three different documents namely, Personal_details, Contact and, Address, you can embed all the three documents in a single one as shown below –

```
{
   _id: POST_ID
   title: TITLE_OF_POST,
   description: POST_DESCRIPTION,
   by: POST_BY,
   url: URL_OF_POST,
   tags: [TAG1, TAG2, TAG3],
   likes: TOTAL_LIKES,
   comments: [
      {
         user:'COMMENT_BY',
         message: TEXT,
         dateCreated: DATE_TIME,
         like: LIKES
      },
      {
         user:'COMMENT_BY',
         message: TEXT,
         dateCreated: DATE_TIME,
         like: LIKES
      }
   ]
}
```

How it works ?

Now, we will see how actually thing happens behind the scene. As we know that MongoDB is a database server and the data is stored in these databases. Or in other words, MongoDB environment gives you a server that you can start and then create multiple databases on it using MongoDB.
Because of its NoSQL database, the data is stored in the collections and documents. Hence the database, collection, and documents are related to each other as shown below:



MongoDB | Delete Database using MongoShell

Short Description : A MongoDB Database is a container for all the collections, where Collection is a bunch of MongoDB documents similar to tables in RDBMS and Document is made of fields similar to a tuple in RDBMS, but it has a dynamic schema here.

Advantages of MongoDB

MongoDB has many advantages which makes it popular and highly demanded.

Schema Not Required: MongoDB doesn’t demand predefined schemas, but schema migration might be necessary for evolving data structures. However, it offers more flexibility than traditional relational databases that strictly require schemas.

Document Queries: MongoDB’s document-oriented approach aligns with dynamic queries. It allows flexible and varied query operations based on the nature of the documents, unlike static table-based queries of RDBMS.

Simplified Performance Optimization: Compared to relational databases, MongoDB’s performance optimization is comparatively simpler due to its architecture and the way it manages data internally.

Efficient Memory Utilization: MongoDB uses internal memory for data storage. So, it accesses the data very fast and enhances the overall performance.

Horizontal Scaling with Sharding: Using MongoDB, we can horizontally expand our database by distributing data across multiple servers. It is called Sharding. Sharding ensures data separation and even distribution across shards, which is very efficient for data retrieval.

Conclusion

MongoDB is like a special toolbox for organizing information in a flexible way. It works across different machines and can handle various types of tasks, like managing websites, games, online stores, and more. It is widely scalable with various programming languages, which makes it easier for developers to use. For instance, if a project requires very specific instructions or involves doing many complex things together, MongoDB might not be the best choice. However, it is possible to use MongoDB to support content management systems, online and offline gaming apps, e-commerce systems, mobile applications, data analytics sections, archiving, logging etc.

Overall, MongoDB is a good NOSQL database system for **unstructured**, **complex,** and large amounts of data because it can handle all things in an easy manner. But if a project needs super strict rules or involves lots of complicated connections between things, other databases might be better. MongoDB is still a popular choice because it's useful for many different jobs, even though it might not be perfect for everything.