MongoDB

Introduction

- **MongoDB** is a **No SQL** database. It is an **open-source, cross-platform, document-oriented** database written in C++.
- **MongoDB** uses the concept of the **document** to store data, which is more flexible than the **row concept** in the relational database management system.
- **MongoDB** doesn't **require predefined schemas** that allow you to add to or remove fields from documents more quickly.
- Like any database system, **MongoDB** allows you **to insert, update, and delete, and select data.** In addition, it supports other **features** like **Indexing, Aggregation, Specify collection and index types, File Storage**
- The philosophy of **MongoDB** is to create a full-featured database that is **scalable, flexible, and fast.**

History

- **MongoDB** was developed and is supported by a company named **10gen** which is a **New York based organization.**
- The initial development of **MongoDB** began in **2007** when the company was building a platform as a service like window azure.
- **MongoDB** was initially developed as a **PAAS** (Platform as a Service). Later in 2009, it is introduced in the market as an **open-source database** server that was maintained and supported by **MongoDB Inc.**
- The first version of **MongoDB** was released in **August 2009** as **1.0**
- The first ready production of **MongoDB** has been considered from **version 1.6** which was released in **August 2010.**
- **MongoDB 6.0** was the latest and stable version which was released in **July 2022.**
- **MongoDB 8.0 preview** is the latest version as of **2024.**
- Complete **MongoDB Version History** with features: Link

MongoDB Installation Guide

Installation Guide 1 Installation Guide 2

MongoDB Server connection using terminal (command prompt)

- Open the bin path of the MongoDB server folder and copy the path.
- Open the terminal and navigate to the bin directory: **cd C:\Program Files\MongoDB\Server\6.0\bin** ***-> press enter.***
- Use the command **mongosh** to connect to the MongoDB server: C:\Program Files\MongoDB\Server\6.0\bin>***mongosh -> press enter.***
- You are connected to the MongoDB server and can start writing commands: ***test> show databases***

Data Types in MongoDB

Data Types are used to define the type of data stored in each field of a document. Some of the common MongoDB data types are:

- **String:** Used to store textual data. Strings are the most used data type.

➔ Example: {"name": "RVK"}

- **Integer:** Used to store numerical data (whole numbers).

➔ Example: {"age": 21}

- **Double:** Used to store floating-point numbers.

➔ Example: {"cgpa": 9.49}

- **Boolean:** Used to store a Boolean (true/false) value.

➔ Example: {"isPlaced": false}

- **Date:** Used to store dates in ISODate format.

➔ Example: {"joinedAt": ISODate("2023-07-24T00:00:00Z")}

- **Array:** Used to store arrays or lists of values.

➔ Example: {"tags": ["full stack dev", "cloud aspirant", "team player"]}

- **Object:** Used to store embedded documents (sub-documents).

➔ Example: {"address": {"Village": "NKP", "City": "Bargarh", "State": "OD"}}

- **ObjectId:** Used to store unique identifiers for documents.

➔ Example: {"_id": ObjectId("507f191e810c19729de860ea")}

➔ We cannot provide **ObjectId** while inserting a document into collection as it will be provided by default to out document.

- **Null:** Used to store a null value.

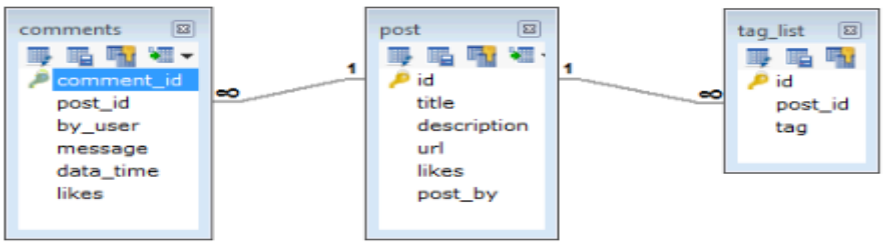
➔ Example: {"relationship": null}

Data Modelling in MongoDB

In MongoDB, data has a flexible schema. It is totally different from SQL database where you had to determine and declare a table's schema before inserting data. MongoDB collections do not enforce document structure.

For example:

- Let us take an example of a client who needs a database design for his website. His website has the following requirements:
- Every post is distinct (contains unique title, description and url).
- Every post can have one or more tags.
- Every post has the name of its publisher and total number of likes.
- Each post can have zero or more comments and the comments must contain username, message, data-time and likes.
- For the above requirement, a minimum of three tables are required in RDBMS.



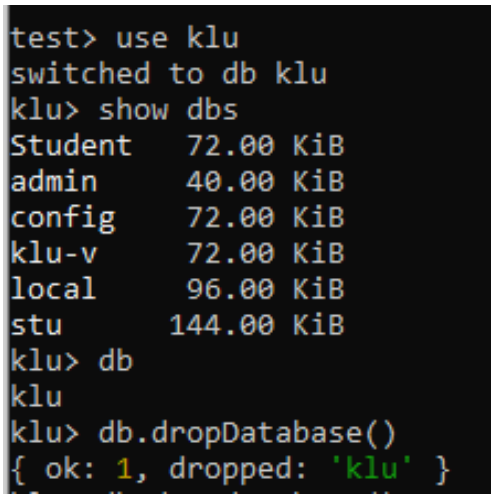
But in **MongoDB,** schema design will have one collection post and has the following **structure:**

{ _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 }] }

MongoDB Database commands

- There is no create database command in MongoDB. MongoDB do not provide any command to create databases.
- If we want a new database, we require **use** command, the use command in MongoDB creates a database if the database with the specified name doesn't exist. If a database with the same name already exists then it switches to that database.
- **Syntax:** use Database_Name
- **Example:** use klu
  - After executing above command if a database named as klu already exists then it switches to klu or else creates a database named as klu
- To check the **currently selected database** we can use **db** command.
- To get the **list of all available databases,** we can use below commands:
  - **Show dbs**
  - **Show databases**
- To **drop or to delete** an existing database, we can use the below command:
  - **db.dropDatabase()**
  - The above command will delete the current database that you are using; to delete other databases, you need to switch to them through **use db** command.

Example:



- In the example a database named as klu is created and switched with **use klu**
- **show dbs** command is used to display the list of all available databases
- **db** command is used to check the currently selected database
- **db.dropDatabase()** command is used to delete the database klu.

MongoDB Collection Commands

A **collection** is a **group of MongoDB documents.** Documents within a collection can have different fields. A **collection** is the equivalent of a **table** in a relational database system.