

MONGODB

Installation of mongodb:

Mongo Shell download link

All the work is expected to do it in mongo shell not in mongo compass

OR

You can also install Studio3T

Connect to mongodb://localhost:27017

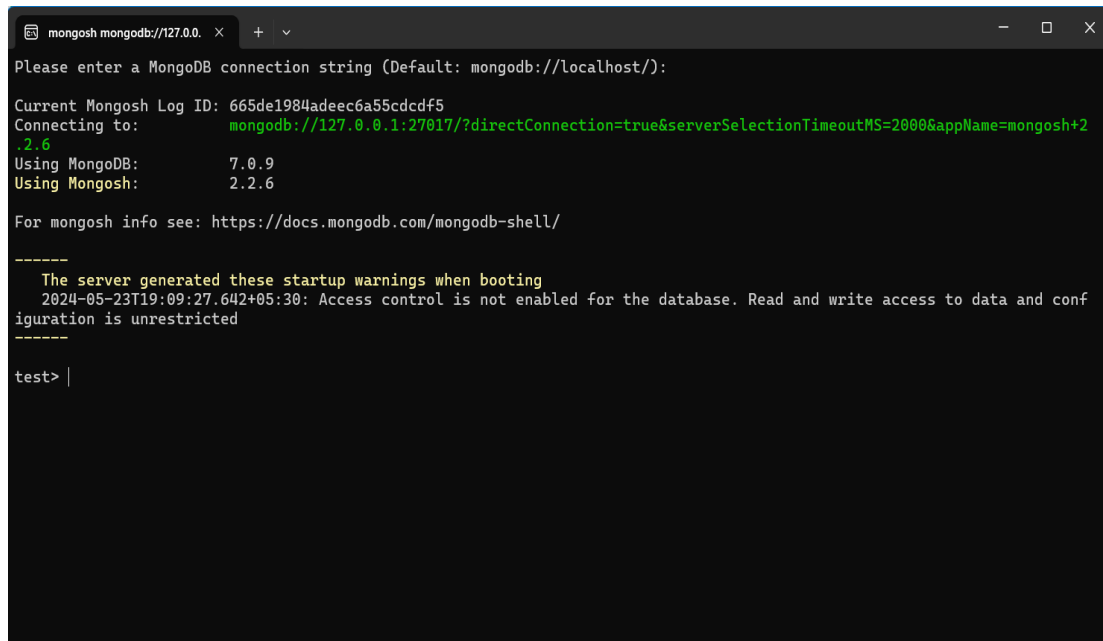
Mongocompass:

MongoDB Compass is a graphical user interface (GUI) for MongoDB, designed to provide an intuitive and visual way to interact with MongoDB databases. As the official GUI tool provided by MongoDB, Compass simplifies the process of managing and visualizing data, making it accessible even for those who may not be deeply familiar with database commands. Users can easily perform CRUD (Create, Read, Update, Delete) operations, visualize schema structures, and analyze documents within their databases. Compass also offers advanced features such as performance monitoring, index analysis, and the ability to run queries using a visual query builder. This tool supports a wide range of MongoDB operations and configurations, helping developers, data scientists, and database administrators efficiently manage their data without the need for extensive command-line interaction. Its user-friendly interface, combined with powerful features, makes MongoDB Compass a valuable asset for optimizing database performance and maintaining the integrity of MongoDB deployments.

MONGOSHELL:

The MongoDB Shell, commonly referred to as mongosh, is an interactive JavaScript interface for MongoDB, providing a powerful command-line environment to interact with MongoDB databases. As a critical tool for database administrators and developers, mongosh allows users to perform a wide range of database operations, including querying, updating, and managing MongoDB collections and documents. The shell supports JavaScript, enabling the execution of complex scripts and functions directly within the MongoDB environment. Users can leverage mongosh to connect to local or remote MongoDB instances, run administrative commands, and

perform tasks such as indexing, aggregation, and data import/export. It also supports modern JavaScript features and offers an improved user experience compared to its predecessor, mongo. With its rich set of features, MongoDB Shell is essential for anyone looking to harness the full potential of MongoDB through a command-line interface, facilitating efficient database management and streamlined workflows.



```
mongosh mongodb://127.0.0.1:27017
Please enter a MongoDB connection string (Default: mongodb://localhost/):
Current Mongosh Log ID: 665de1984adeec6a55cdcdf5
Connecting to:      mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.2.6
Using MongoDB:      7.0.9
Using Mongosh:      2.2.6

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

-----
  The server generated these startup warnings when booting
  2024-05-23T19:09:27.642+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
-----

test> |
```

Documents:

At the heart of MongoDB is the document:

an ordered set of keys with associated values.

The representation of a document varies by programming language, but most languages have a data structure that is a natural fit, such as a map, hash, or dictionary. A document in MongoDB is a record in a collection, akin to a row in a relational database, but it is stored in a flexible, JSON-like format called BSON (Binary JSON). This flexibility allows for the storage of nested structures and varying data types without a predefined schema. Using Compass, users can easily visualize document structures, modify field values, add new fields, and run queries to filter documents, all without needing to write complex code. This makes MongoDB Compass a powerful tool for both developers and database administrators, enabling efficient management of data and

simplification of database operations.

```
{ "greeting" : "Hello, world!" }
```

```
{ "greeting" : "Hello, world!" }
```

Collections:

Collections A collection is a group of documents.

If a document is the MongoDB analog of a row in a relational database, then a collection can be thought of as the analog to a table. Collections in MongoDB Compass are analogous to tables in a relational database but offer much greater flexibility due to MongoDB's schema-less nature. A collection is a grouping of MongoDB documents, where each document can have a different structure, making it easy to store diverse data within the same collection. In MongoDB Compass, collections are visually represented, allowing users to explore and manage their data seamlessly. Users can create, delete, and rename collections, as well as view detailed statistics about the documents they contain. Compass provides tools to index collections, ensuring efficient query performance, and offers various options to filter, sort, and aggregate data within collections. This user-friendly interface empowers users to perform complex database operations and gain insights into their data without needing extensive MongoDB query language knowledge.

Database:

MongoDB groups collections into databases.

A single instance of MongoDB can host several databases, each grouping together zero or more collections.

A database has its own permissions, and each database is stored in separate files on disk.

A good rule of thumb is to store all data for a single application in the same database.

With Compass, users can easily create, rename, and delete databases, as well as explore their contents through a graphical interface. The intuitive design of Compass allows users to drill down from the database level to individual collections and documents, providing a comprehensive view of the data hierarchy. Additionally, Compass offers functionalities such as monitoring database performance, managing indexes, and executing queries, making it a powerful tool for database administration and data analysis. This seamless

interaction with databases in MongoDB Compass enhances productivity and simplifies the management of complex data structures.

Datatype:

Basically each document will be in JSON format which will be as follows. Where each attributes inside can be of multiple data types. In MongoDB, data types are crucial for defining the nature of data stored in documents. MongoDB supports a wide array of data types, reflecting its versatility and flexibility in handling diverse data structures. Common data types include String, Number (int, long, double, decimal), Boolean, Date, Array, Object (embedded document), Null, and Binary Data. Additionally, MongoDB supports unique data types such as ObjectId, which is a 12-byte identifier used as a primary key, and Min/Max Key, which represent the lowest and highest BSON values respectively. This variety allows for the storage of complex and hierarchical data within a single document. Each data type is efficiently encoded in BSON (Binary JSON) format, ensuring optimal performance and storage. Understanding these data types is essential for designing effective schemas and optimizing queries in MongoDB, allowing for a more efficient and scalable database system.

```
{
  "name" : "John Doe",
  "address" : {
    "street" : "123 Park Street",
    "city" : "Anytown",
    "state" : "NY"
  }
}
```

Different datatypes:

1.Date

2.Int32

3.Decimal

4.Timestamp

Few Commands to test after connections

Command	Expected Output	Notes
show dbs	<code>admin 40.00 KiB</code> <code>config 72.00 KiB</code> <code>db 128.00 KiB</code> <code>local 40.00 KiB</code>	All Databases are shown
use db	<code>switched to db db</code>	Connect and use db
show collections	<code>Students</code>	Show all tables
db.foo.insert({"bar" : "baz"})		Insert a record to collection. Create Collection if not exists

Few Commands to test after connections

Command	Notes
db.foo.batchInsert([{"_id" : 0}, {"_id" : 1}, {"_id" : 2}])	Insert more than one document
db.foo.find()	Print all rows
db.foo.remove()	Remove foo table