Lesson:

Creating and Dropping Database & collections

PW SKILLS

Topics

- Introduction
- Installation of MongoDB
- Using the MongoDB compass
- Using the MongoDB shell
- Best practices for Creating Database and collection
- Best practices for Dropping Database and collection
- General best practices
- Interview points

Introduction

MongoDB stands out as a flexible and scalable NoSQL database, and understanding how to create and drop databases and collections is fundamental for effective data management. In this module, we'll explore three different interfaces for managing databases and collections in MongoDB: the MongoDB Compass, and the MongoDB Shell.

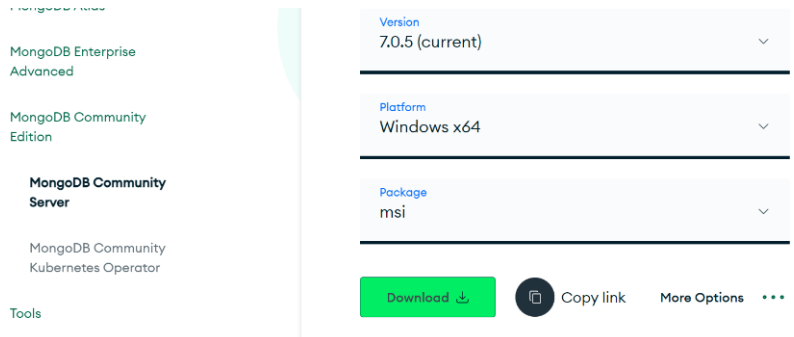
Installation of MongoDB

MongoDB stands out as a flexible and scalable NoSQL database, and understanding how to create and drop databases and collections is fundamental for effective data management. In this module, we'll explore three different interfaces for managing databases and collections in MongoDB: the MongoDB Compass, and the MongoDB Shell.

To install MongoDB, follow these steps -

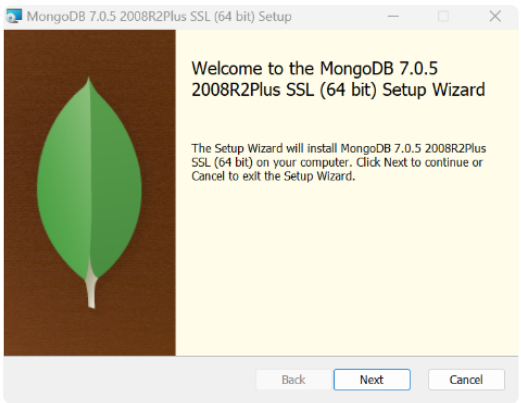
Step 1 - Download MongoDB

- Visit the official MongoDB website at **MongoDB download Centre**
- Choose the appropriate version of MongoDB for your operating system and download the installer.

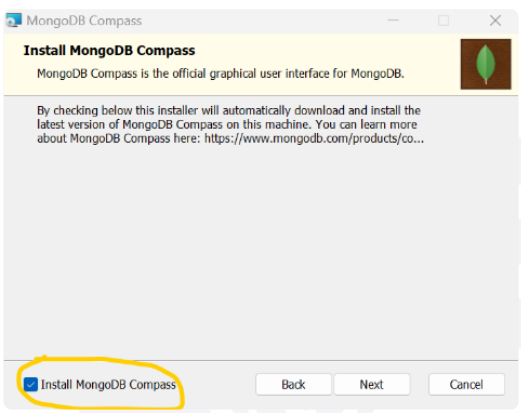


Step 2 - Run the Installer

- Locate the downloaded installer file and double-click on it to run the MongoDB installer.



- Ready to install? Just follow the instructions on your screen! You can pick where to put the software and choose extra stuff like MongoDB Compass, which gives you a cool visual interface for working with your database.

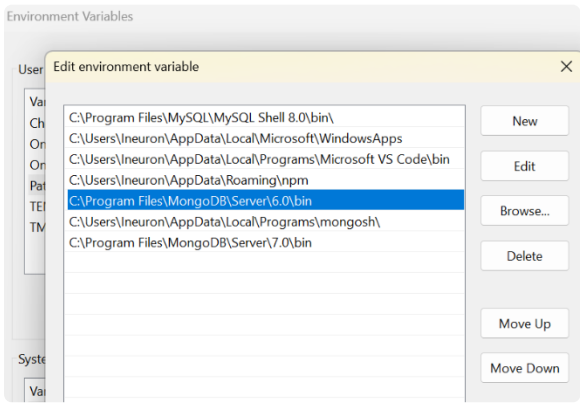


Check mark the blue checkbox to install MongoDB compass as well

Step 3 - Add MongoDB Bin Directory to the system Path (windows)

- Open the Start menu and search for "environment variables."
- Click on "Edit the system environment variables."
- In the System Properties window, click on the "Environment Variables" button.
- In the "System variables" section, locate the "Path" variable and click on "Edit."
- Click on "New" and add the path to the MongoDB bin directory. By default, the bin directory is located at "C:\Program Files\MongoDB\Server{version}\bin".
- Click "OK" to save the changes.
- Once save the changes, make sure the bin directory path is available in the system environment variable path

i.e



Step 4 - Verify the MongoDB Installation

- Open a new command prompt window.
- Run the following command to check if MongoDB is installed and configured correctly:

```
C:\Users\Ineuron>mongod --version
db version v6.0.7
Build Info: {
    "version": "6.0.7",
    "gitVersion": "202ad4fda2618c652e35f5981ef2f903d8dd1f1a",
    "modules": [],
    "allocator": "tcmalloc",
    "environment": {
        "distmod": "windows",
        "distarch": "x86_64",
        "target_arch": "x86_64"
    }
}
```

This command displays the installed MongoDB version

Step 5 - Create the MongoDB Data Directory

- Open the C drive in File Explorer.
- Create a new folder named "data".
- Inside the "data" folder, create another folder named "db". This is the default data directory used by MongoDB.

Step 6 - Start the MongoDB server

- In the command prompt, run the following command to start the MongoDB server:

```
Unset
mongod
```

- If everything is set up correctly, the MongoDB server should start successfully without any errors.

Note: Keep the command prompt window running to keep the server active.

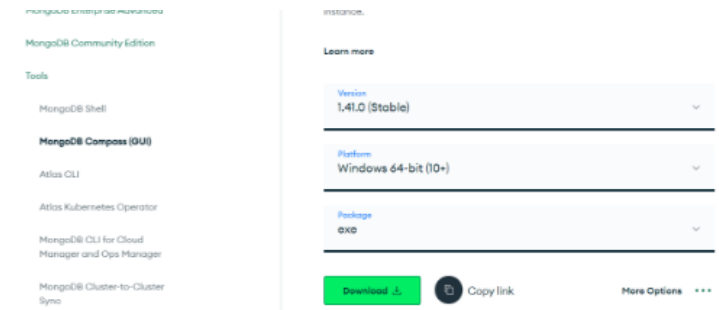
Using the MongoDB Compass

No worries if you missed installing MongoDB Compass initially! Along with the mongoDB,

Here's how to get it up and running quickly:

Step 1 - Download MongoDB Compass

- Visit the official MongoDB website **(MongoDB Compass Download)** and download the latest version of MongoDB Compass.
- Choose the installer that matches your operation system (windows, macOS, or Linux)

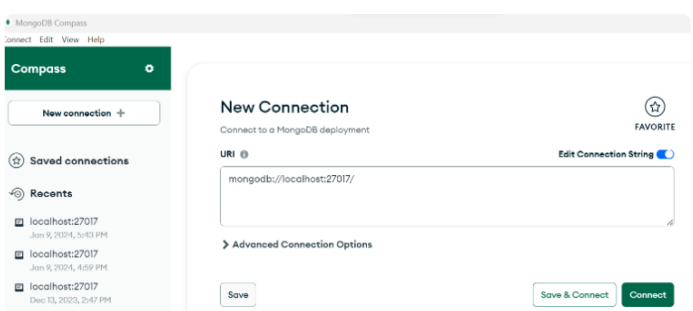


Step 2 - Run the installer

- Double-click the downloaded installer file
- Follow the on-screen instructions to proceed with the installation
- You'll be prompted to choose an installation directory and select any additional components if desired

Step 3 - Launch MongoDB Compass

- Once the installation is complete, locate MongoDB Compass on your computer and launch it.
- You may be asked to configure privacy settings and specify update preferences upon first launch.



Create a Database and Collection

Certain users prefer utilizing a graphical user interface (GUI) for tasks involving creating and modifying data and collections. MongoDB's GUI, Compass, not only provides CRUD operations (create, read, update, delete) for data, databases, and collections but also offers enhanced features such as data visualization and performance profiling.

Note - Prerequisites for using Compass with a self-managed (local) MongoDB cluster
If you are using self-managed MongoDB:

- Make sure the MongoDB self-managed cluster is installed and running on your machine or server
- Make sure you have a database user on the MongoDB cluster you want to use Make sure you have MongoDB Compass installed on your computer. If not, download and install Compass for your operating system.

First, open the MongoDB compass and connect the instance locally as shown below by clicking on the **"connect button"** -

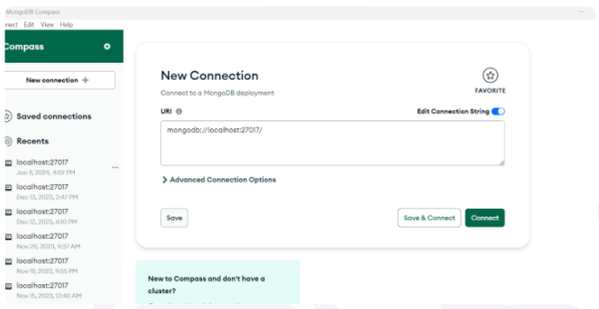


fig. MongoDB Compass

The Databases tab in MongoDB Compass has a "Create Database" button.

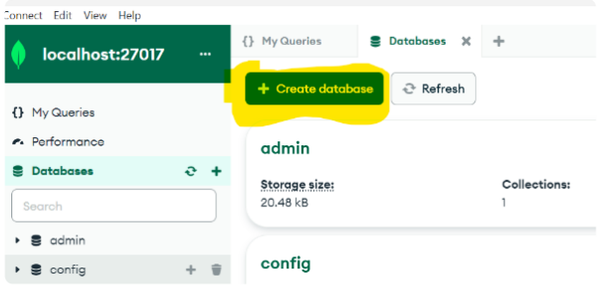
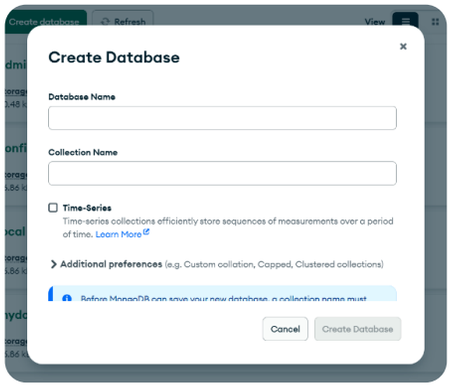


Fig MongoDB compass



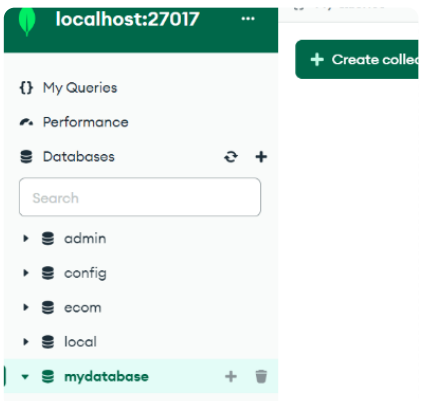
Specify the database name and collection name, then click the "Create Database" button to create your database and collection successfully.

Delete / Drop Database and Collection

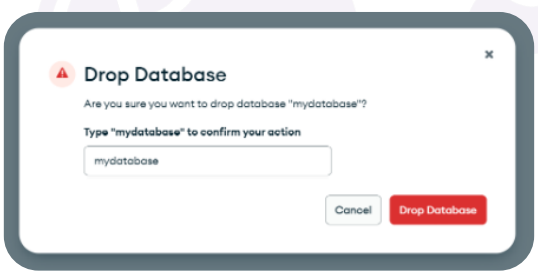
Deleting a database and collection in MongoDB Compass involves a straightforward process -

Follow these steps to drop a database

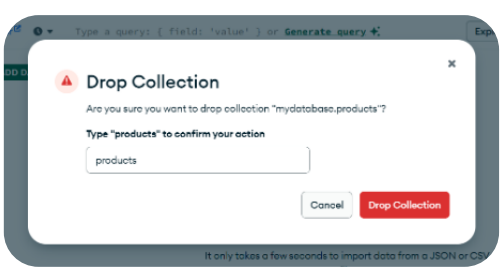
- **Open MongoDB Compass:** Launch MongoDB Compass on your machine.
- **Connect to MongoDB Server:** If you are not already connected to your MongoDB server, click on the "Connect" button and provide the necessary connection details.
- **Navigate to Databases:** Once connected, you'll see the list of databases on the left-hand side of the Compass interface. Click on the database you want to delete.



- **Drop database:** After selecting the database, click on the “delete icon” next to the database name. This will open a dialog box to confirm the database by entering the database name.



The red button labeled with drop database is clicked to successfully delete the database.
Similarly, select the collection inside the database, click on the “three dot” next to the collection name, and select “drop collection” This will open a dialog box to confirm the collection to be deleted by entering the the collection name.



The red button labeled with drop collection is clicked to successfully delete the collection.

- **Enter Password (if required):** For security purposes, you might need to enter your MongoDB credentials or follow any additional authentication steps. Provide the necessary information.

Using the MongoDB shell

Create a Database and collection

MongoDB, like many intricate software systems, can be managed through a command-line interface, commonly known as a CLI. This interface enables users to issue commands to MongoDB, control its operations, obtain cluster status information, and execute essential tasks such as creating databases and collections. To initiate database and collection creation via the command line, the initial step is gaining access to the MongoDB cluster through the MongoDB Shell, a program facilitating command entry into the system.

Note - Prerequisites

Before you begin, make sure you have the following:

- MongoDB is installed on your machine.
- A MongoDB database instance running (either locally or hosted).

Once you have access to a cluster via the MongoDB Shell, you can see all the databases in the cluster that you have access to using the “show” command:

```
Unset
test> show dbs
admin 40.00 KiB
local 72.00 KiB
test>
```

Note that admin and local are databases that are part of every MongoDB cluster. There is no “create” command in the MongoDB Shell. To create a database, you will first need to switch the context to a non-existing database using the “use” command:

```
Unset
test> use myDatabase
```

Note that, for now, only the context has been changed. If you enter the show dbs command, the result should still be the same:

```
Unset
test> show dbs
admin 40.00 KiB
local 72.00 KiB
test>
```

Wait a second. Where's myshinynewdb?
MongoDB only creates the database when you first store data in that database. This data could be a collection or a document.

To add a document to your database, use the db.collection.insertOne()
Command, this command will automatically create a collection with the given name.

```
Unset
mydatabase> db.products.insertOne({name: "Hoddies", price: 750})
{
  acknowledged: true,
  insertedId: ObjectId("659d3504174ef62ed85ecce4")}
```

The "products" in the command refers to the collection that the document was being inserted in.
Collections can be created just like databases, by writing a document to them. They can also be created using the createCollection command.
Now, if you run the show dbs command, you will see your database.

```
Unset
mydatabase> show databases
admin        40.00 KiB
config      108.00 KiB
local        72.00 KiB
mydatabase   72.00 KiB
```

How did the insert command know to put the data into mydatabase?
It turns out that when you entered the use command, then mydatabase became the current database on which commands operate.
To find out which database is the current one, enter the db command:

```
Unset
mydatabase> db
mydatabase
```

The db command displays the name of the current database. To switch to a different database, type the use command and specify that database.

The "show collections" command can be used to display all the available collections in the database

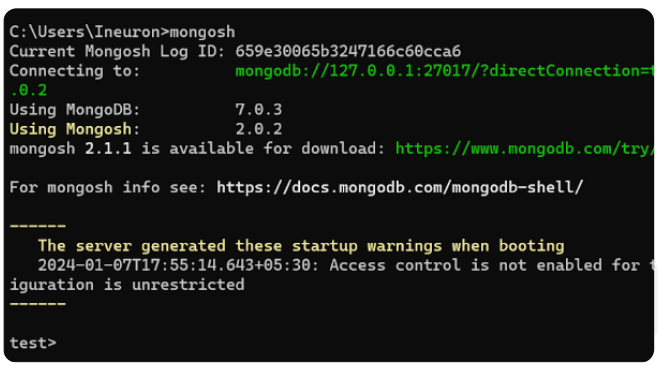
```
Unset
mydatabase> show collections
products
```

Delete / Drop Database and Collection

In MongoDB, dropping a database and collection can be done through the MongoDB shell using the dropDatabase() method. However, it's crucial to exercise caution as dropping a database and collections is irreversible and deletes all data within that database.

Here are the steps to delete a database and collections using the MongoDB shell:

- **Open MongoDB Shell** - Open your terminal or command prompt
- **Connect to MongoDB** - Connect to your MongoDB instance using the “mongosh” command. If your MongoDB instance is running locally, you can simply run “mongosh” in the terminal.



- **Select the Database** - Switch to the database you want to drop using the “use” command. For example, to switch to a database named "mydatabase," you would run:

```
Unset
test> use myDatabase
switched to db mydatabse
mydatabase>
```

- **Drop the database and collection** - once you are inside the desired database, run “db.dropDatabase()” command to drop the selected database, similarly run “db.collectionName.drop()” command to drop the selected collections

Delete database

```
Unset
mydatabase> db.dropDatabase()
{ ok: 1, dropped: 'mydatabase' }
```

Delete collections

```
Unset
mydatabase> db.products.drop()
true
```

Confirm Deletion - MongoDB will give you a confirmation message with "{ok: 1, dropped: 'mydatabase'}" for deletion of the database and "true" for deletion of the collection

Caution: Deleting the database will result in the deletion of all collections contained within it.

Best practices Creating Databases and collections

The following are some of the best practices -

- **Naming Conventions:** Follow a consistent and meaningful naming convention for databases and collections. Use clear, descriptive names that reflect the purpose and content.
- **Authentication and Authorization:** Implement strong authentication and authorization mechanisms. Create dedicated users with minimal necessary privileges to limit access and reduce the risk of unauthorized operations.
- **Indexes and Data Types:** Plan and create indexes based on query patterns and access requirements. Choose appropriate data types for fields to optimize storage and query performance.
- **Sharding Consideration:** Consider sharding early if you anticipate the need for horizontal scaling. Plan your data model with sharding in mind to ensure a smooth transition if scaling becomes necessary.
- **Document Design:** Design your documents to match the query patterns and retrieval needs of your application. Avoid embedding large arrays or sub-documents that could lead to performance issues.

Best practices Dropping Databases and collections

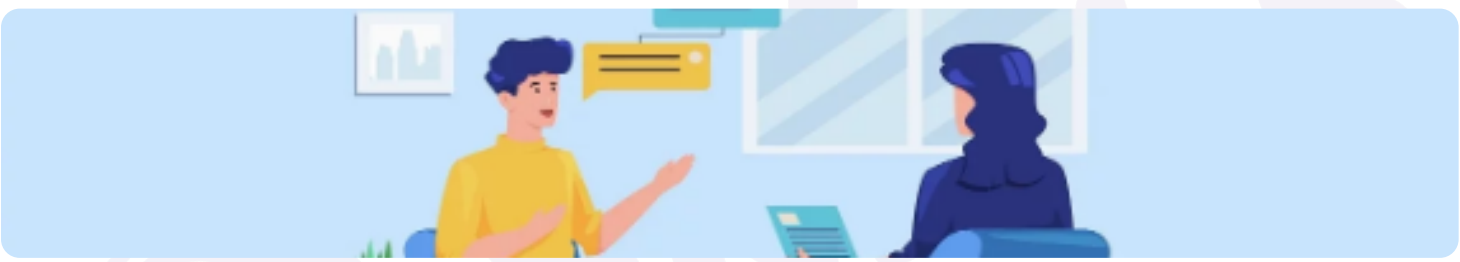
The following are some of the best practices -

- **Backup Procedures:** Before dropping a database or collection, ensure you have a recent backup. This is a crucial safety net in case of accidental deletion or the need to recover data.
- **Review Dependencies:** Before dropping a collection, review any dependencies, such as applications or queries relying on that collection. This helps prevent unintended consequences.
- **Confirmation and Auditing:** Implement confirmation steps for dropping databases or collections, especially in production environments. Enable auditing to track such operations and identify responsible users.
- **Temporary Collections:** For temporary or transient data, consider using temporary collections rather than dropping and recreating collections. This avoids the need for frequent create-drop operations.
- **Rebuilding Indexes:** After dropping and recreating collections, rebuild necessary indexes. This ensures optimal query performance and prevents potential issues with missing indexes.

General Best Practices

- **Documentation:** Maintain comprehensive documentation for databases and collections. Document the purpose, structure, and any specific considerations for future reference.
- **Regular Maintenance:** Schedule regular maintenance tasks, including index optimization and data compaction, to ensure the ongoing health and performance of your MongoDB deployment.
- **Monitoring:** Implement monitoring tools to track the performance and resource utilization of your MongoDB instance. Identify and address any anomalies or potential issues proactively.
- **Testing in a Sandbox:** Before making significant changes in a production environment, test the database and collection creation or deletion in a sandbox or staging environment to validate the process.
- **Educate Users:** Educate database users and administrators on best practices and potential risks associated with creating and dropping databases and collections. Encourage cautious and informed actions.

Interview points



Question 1 - How can you ensure secure practices when creating databases or collections in a production MongoDB environment?

Answer - Implement proper authentication, set strong access controls, and regularly audit permissions. Monitor for any suspicious activities related to database or collection modifications.

Question 2 - Question: Why is it advisable to follow naming conventions when creating databases and collections in MongoDB?

Answer - Naming conventions enhance clarity and consistency in data organization, making it easier for developers and administrators to understand the purpose of databases and collections.

PW SKILLS

THANK YOU !