MongoDB Indexing

Estimated time needed: **30** minutes

Objectives

After completing this lab, you will be able to:

- Measure the time it takes to execute a query with the explain function
- Describe the process of creating, listing and deleting indexes
- Evaluate the effectiveness of an index

About Skills Network Cloud IDE

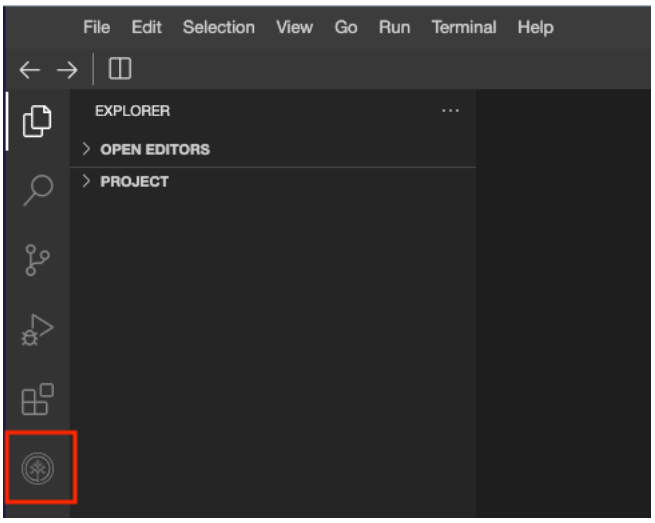
Skills Network Cloud IDE (based on Theia and Docker) provides an environment for hands on labs for course and project related labs. Theia is an open source IDE (Integrated Development Environment), that can be run on desktop or on the cloud. To complete this lab, we will be using the Cloud IDE based on Theia and MongoDB running in a Docker container.

Important Notice about this lab environment

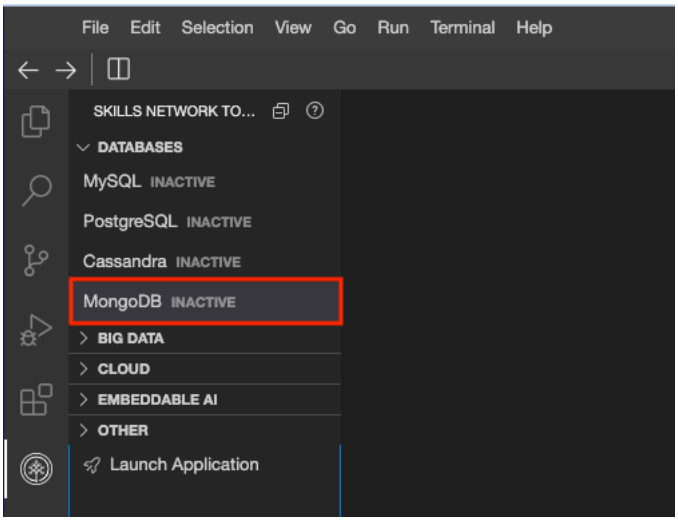
Please be aware that sessions for this lab environment are not persisted. Every time you connect to this lab, a new environment is created for you. Any data you may have saved in the earlier session would get lost. Plan to complete these labs in a single session, to avoid losing your data.

Set-up: Start MongoDB

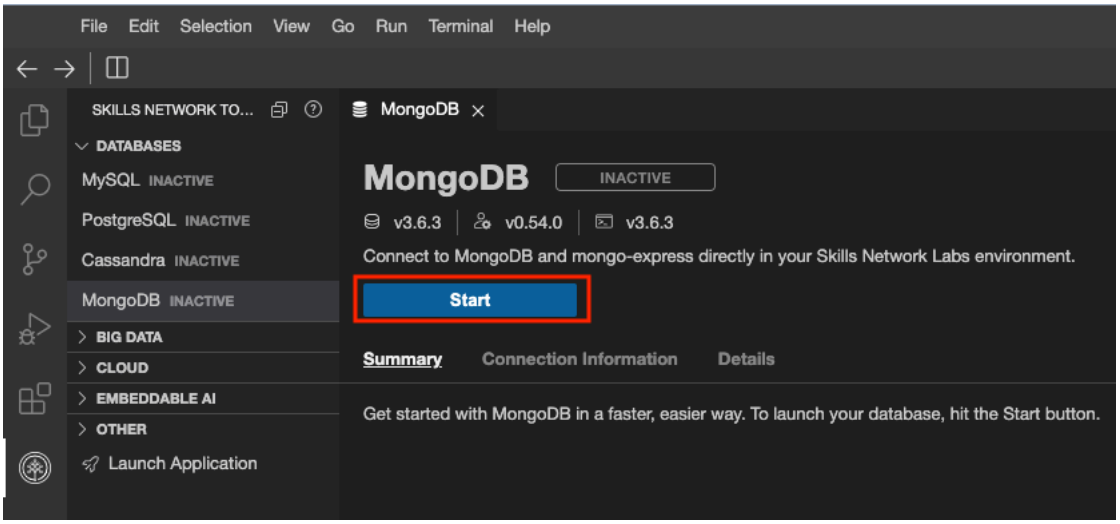
Navigate to Skills Network Toolbox.



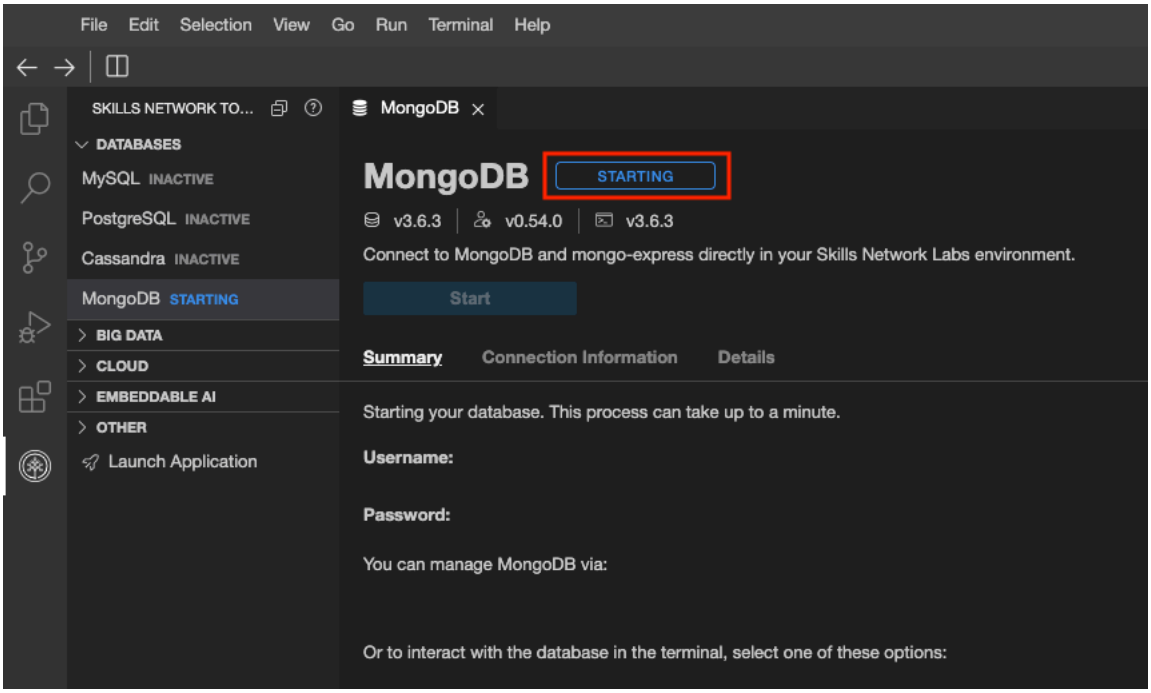
You will notice MongoDB listed there, but inactive. Which means the database is not available to use.



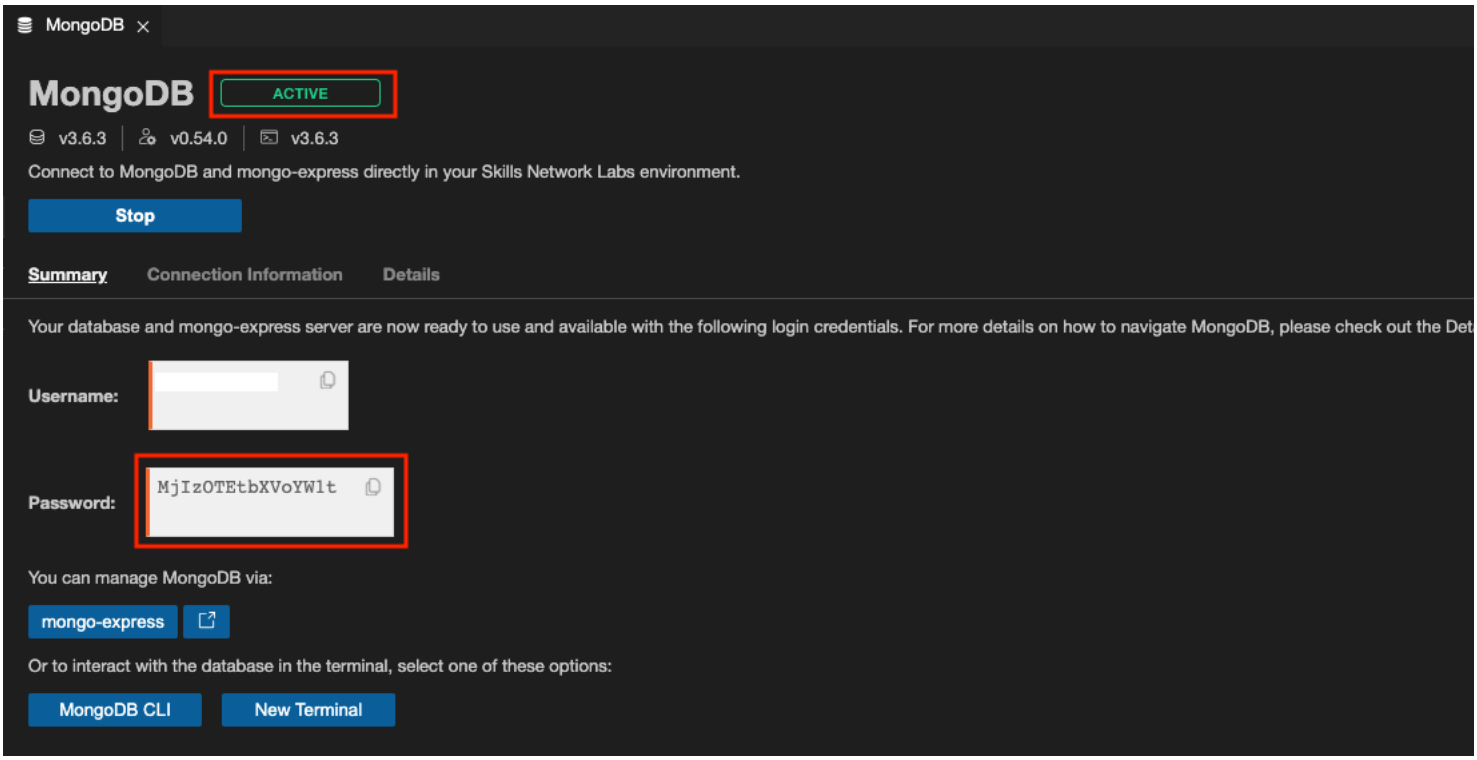
Once you click on it, you will see more details about it and a button to start it.



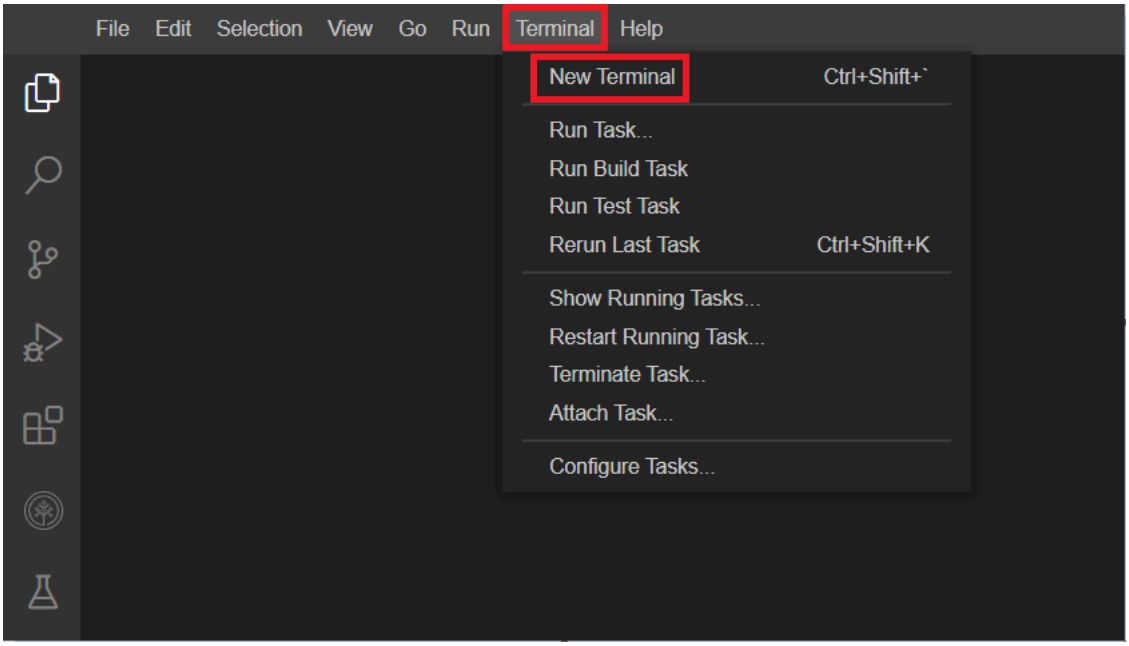
Clicking on the start button will run a background process to configure and start your MongoDB server.



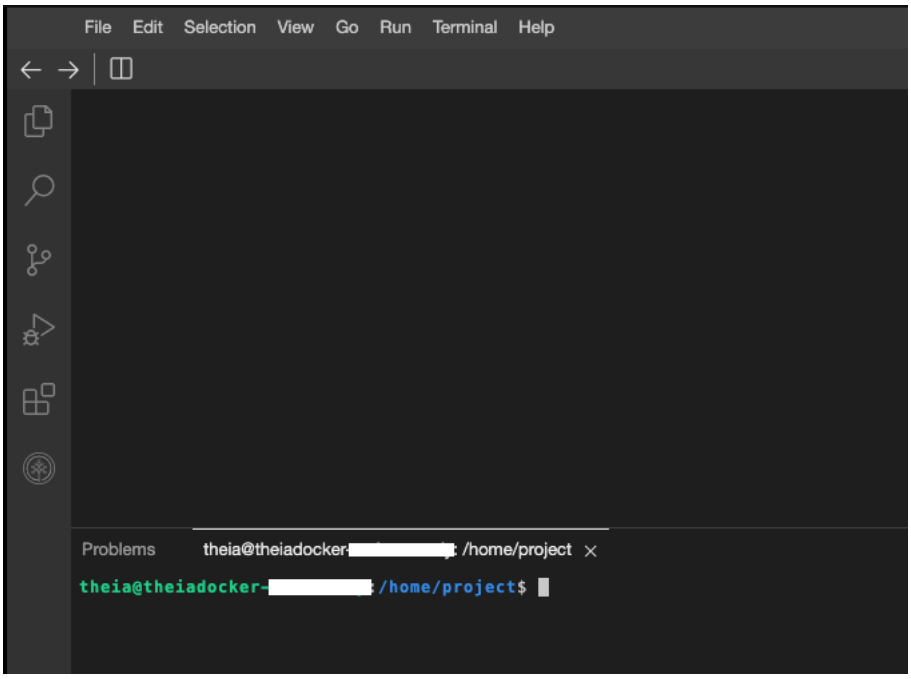
Shortly after that, your server is ready for use. This deployment has access control enabled and MongoDB enforces authentication. So, take note of the password as you will need it to login as `root` user.



You can now open terminal and enter details yourself.

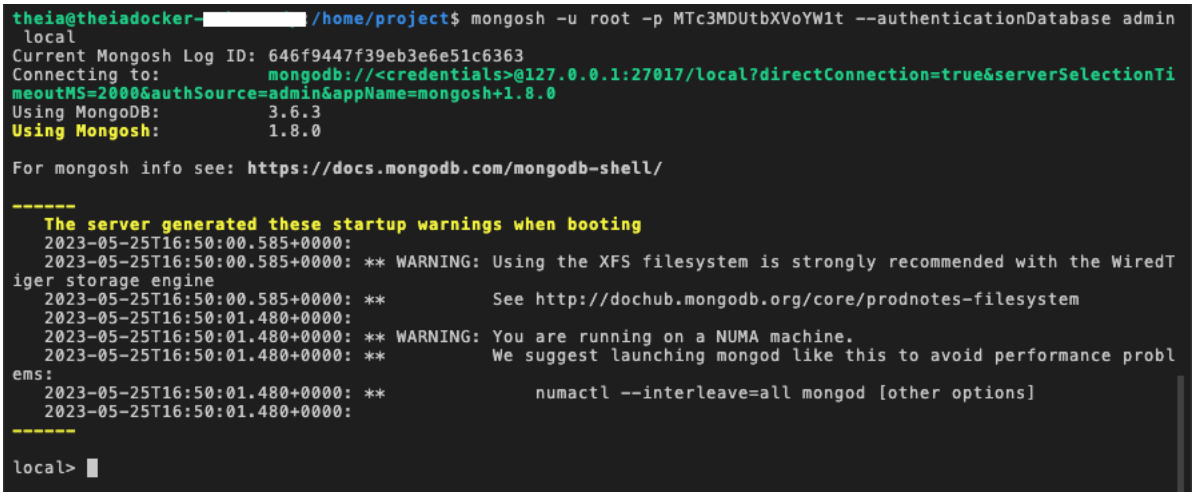


This will open a new terminal at the bottom of the screen as in the image below.



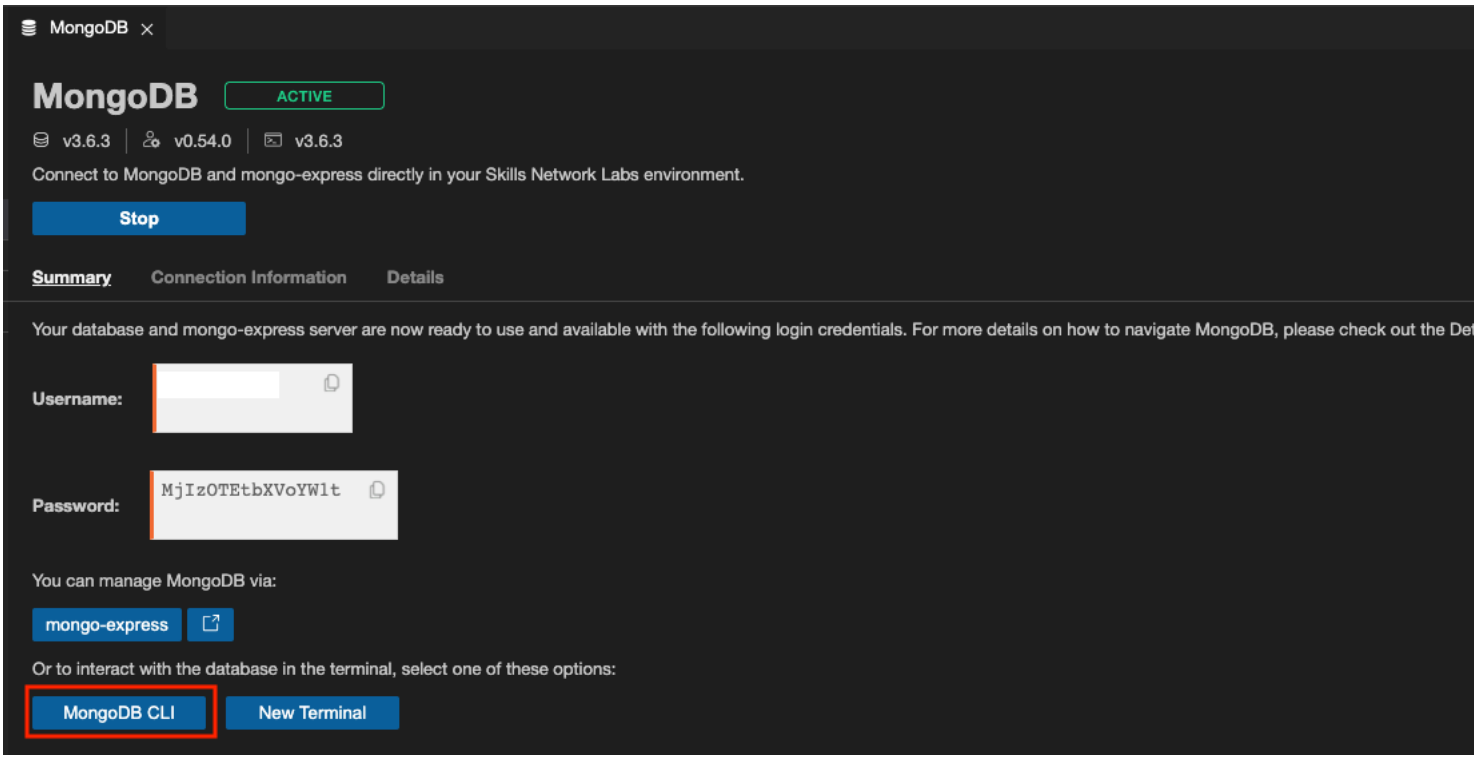
Run the below command on the newly opened terminal. (You can copy the code by clicking on the little copy button on the bottom right of the codeblock below and then paste it, wherever you wish.)

```
mongosh -u root -p PASSWORD --authenticationDatabase admin local
```



The command contains the username and password to connect to mongodb server (the text after the -p option is the password). Your output would be different from the one shown above. Copy the command given to you, and keep it handy. You will need it in the next step.

Or you can simply click on MongoDB CLI which does that for you.



In MongoDB CLI (for example, mongo shell), switch the context to `training` database.

```
use training
```

And create a collection called `bigdata`

```
db.createCollection("bigdata")
```

Exercise 1 - Insert documents

- Let's insert 200k documents into the newly created collection.
- This should take a few seconds to complete.
- The code given below will insert these documents into the `bigdata` collection.
- Each document has a field named `account_no` which is assigned to incrementing variable `i`.
- Another field `balance` contains a randomly generated number, to simulate the bank balance for the account.

Copy the below code and paste it on the mongo client.

```
docsToInsert = []
for (i = 1; i <= 200000; i++) { docsToInsert[i-1] = { "account_no": i, "balance": Math.round(Math.random() * 20000) } }
db.bigdata.insertMany(docsToInsert);
```

Verify that `200000` documents got inserted by running the below command.

```
db.bigdata.countDocuments()
```

Exercise 2 - Measure the time taken by a query

Let's run a query and find out how much time it takes to complete. You will query for the details of account number 58982.

We will make use of the `explain` function to find the time taken to run the query in milliseconds.

> The db.collection.explain("executionStats") method provides statistics about the performance of a query. These statistics can be useful in measuring if and how a query uses an index. See db.collection.explain() for details.

Run the below command.

```
db.bigdata.find({"account_no":58982}).explain("executionStats").executionStats.executionTimeMillis
```

Make a note of the milliseconds it took to run the query. We will need this at a later stage.

Exercise 3 - Working with indexes

Before you create an index, choose the field you wish to create an index on. It is usually the field that you query most.

Run the below command to create an index on the field `account_no`.

```
db.bigdata.createIndex({"account_no":1})
```

Where `1` means `ascending` order.

Run the below command to get a list of indexes on the `bigdata` collection.

```
db.bigdata.getIndexes()
```

You should see an index named `account_no_1`

Exercise 4 - Find out how effective an index is

You will now run the same query for account `58982` and compare the execution time from previous run. This way, you can compare the improvement.

Run the below command.

```
db.bigdata.find({"account_no":58982}).explain("executionStats").executionStats.executionTimeMillis
```

This time, the execution time should be a lot less than previously. If you see `0`, it means the query completed under 1 millisecond.

Exercise 6 - Delete an index

Use the below command to delete the index we created earlier. Here you can provide index definition or name.

```
db.bigdata.dropIndex({"account_no":1})
```

Bonus information

MongoDB creates a unique index on the `_id` field during the creation of a collection. The `_id` index prevents clients from inserting two documents with the same value for the `_id` field. You cannot drop this index on the `_id` field.

Practice exercises

1. Problem:

   Create an index on the `balance` field.

▶ Click here for Hint
▼ Click here for Solution

On the mongo client run the below commands.

```
db.bigdata.createIndex({"balance":1})
```

2. Problem:

   Query for documents with a balance of `10000` and record the time taken.

▶ Click here for Hint
▼ Click here for Solution

```
db.bigdata.find({"balance":10000}).explain("executionStats").executionStats.executionTimeMillis
```

3. Problem:

   Drop the index you have created.

▶ Click here for Hint
▼ Click here for Solution

```
db.bigdata.dropIndex({"balance":1})
```

4. Problem:

   Query for documents with a balance of 10000 and record the time taken, and compare it with the previously recorded time.

▶ Click here for Hint
▼ Click here for Solution

```
db.bigdata.find({"balance": 10000}).explain("executionStats").executionStats.executionTimeMillis
```

Summary

In this lab, you have gained an understanding of indexing in MongoDB.

Author(s)

Muhammad Yahya