**COMP835**

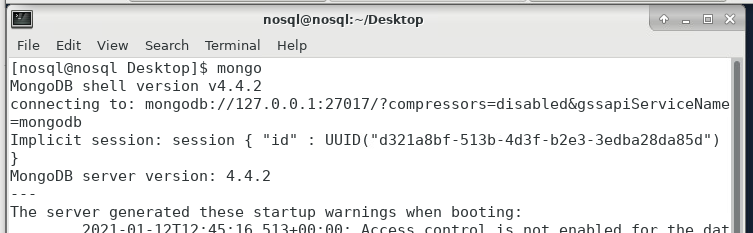
**Lab 6. Introduction to MongoDB database**

**Overview**

In this lab you will work with different data types in MongoDB data store.

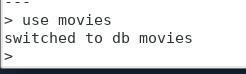
**Lab 6.1 Ensure your MongoDB server is running.**

Open in Terminal and start MongoDB, by using command : **mongo**



Create a database called **movies**

The command for this is: **use movies,** and press Enter.



You will notice that it will now indicate that you have switched to the db called **movies.**

You can confirm that the database has now been selected by entering the command **db** and then pressing enter.



**Lab 6.2 Entering data into MongoDB.**

**6.2.1**

We now want to enter some documents into the **movies** database and then add these to a collection call **video\_records**.

Our first document will have the following data in it:

1. \_id: 1
2. Title: “Les Miserables”
3. Director: “Tom Hooper”
4. Runtime: 158

There are different ways to enter the document into mongoDB.

First method is to create a variable/document and then use the name of the document in the insert command.

Create document called **lesmis** with the following data:

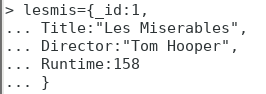
**lesmis = {\_id: 1,**

**Title: “Les Miserables”,**

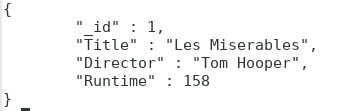
**Director: “Tom Hooper”,**

**Runtime: 158**

**}**



Note: Mongo will display your document in response:



After you have created your **lesmis** document, you must insert it into a collection which we have called **video\_records.** Use the following command and press enter:



Mongo shows the result 

Verify that your document has been added by using the **find()** command:





By inserting a first document, the collection video\_records has been created as well.

You can display collections by using **show collections** command:



**6.2.2**

Now create the second document with new document name **skyfall** for the following records details:

\_id=2

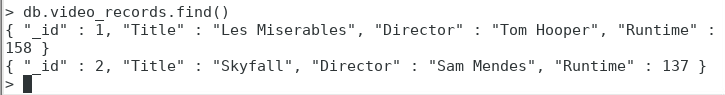
Title= “Skyfall”

Director= “Sam Mendes”

RunTime= 137

And then insert the document into **video\_records** collection.

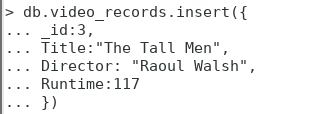
Query all documents in the collection video\_records. You should get two documents displayed.



**Note:** If are you in the middle of the entering command and you need to exit, use <Ctrl-C>

**6.2.3**

Now you will try another method of inserting the document using **db.video\_records.insert** method with the new document specified inside the method:



Verify that your collection **video\_records** has three documents now. Which command will you use?

Both methods of inserting data are similar, but the use of the document name has a benefit: you can query the data using the document name:



**6.2.4**

Now enter a new document which has some intentional data mistakes in it. Leave them as is.

You can use either method : creating a document first and then inserting it or inserting on fly.

\_id:4,

Title: “Sam Mendes”,

Director: “Skyfall”,

Runtime: 143,

Price: 9.99,

Year: 2010

Also note the structure of this new document is different. Extra data (**Price** and **Year**) being included before inserting it the **video\_records** collection and the database doesn’t complain about different structure.

Insert another document for director Sam Mendes:

\_id:5,

Title: “Sam Mendes”,

Director: “Spectre”,

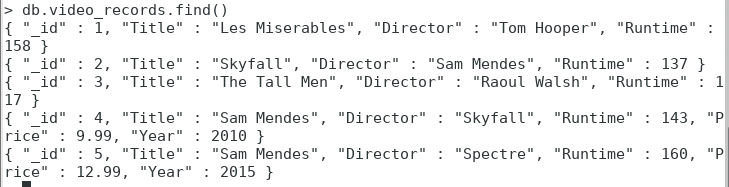
Runtime: 160,

Price: 12.99,

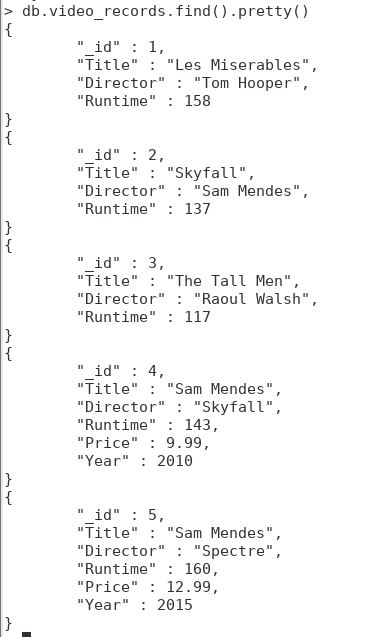
Year: 2015

As you probably noticed that documents 4 and 5 have mistakes: “Sam Mendes” value is assigned to a wrong key. This was done deliberately, so leave it as it is for the moment.

Verify that you have all 5 documents in the collection:

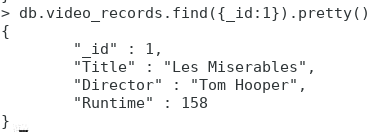


Use pretty() method to produce better formatted output:

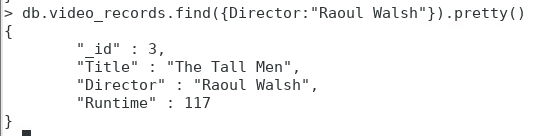


**Lab 6.3. Querying the Video Records Data with ‘where’ conditions.**

1. Select all documents from the collection video\_records that have an \_id of 1



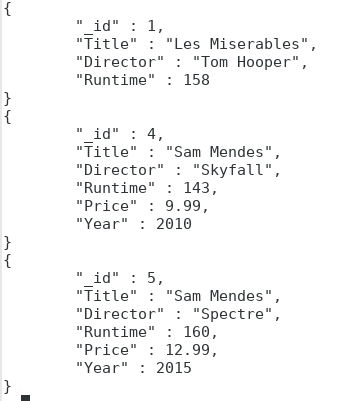
1. Select all documents from the collection **video\_records** that have a director Raoul Walsh.



1. Select all documents from the video recording collection which have a runtime longer than 140 minutes.

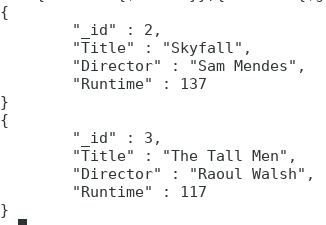
Use the lecture notes to get the correct syntax.

You should get this output:

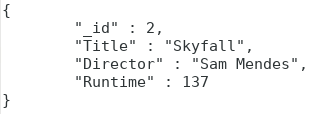


4. Select all documents from the video recording collection which have a runtime between 110 and 140 minutes.

You should get this output:

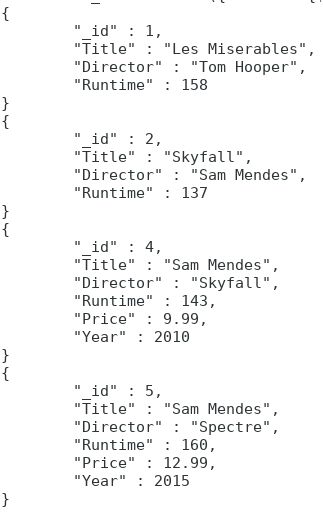


5. Select all documents from the video recording collection which have a runtime of less than 140 and not equal to 117. You should get this output:



6. Select all documents from the collection **video\_records** that DO NOT have a director Raoul Walsh.

You should get this output:



**Lab 6.4 Updating and Removing data.**

We left deliberately some mistakes in the documents 4 and 5. So, now we would like to fix them by updating the document 4 and removing document 5 from the collection **video\_records.**

Hint: use lecture notes for guidelines.

**6.4.1**

Update document 4 using UPDATE command.

We need to change the Title of the movie to “Skyfall” and then Director to “Sam Mendes”.

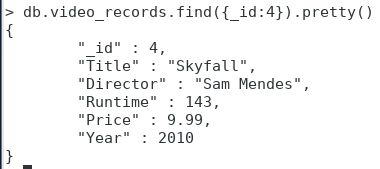


You should get a confirmation:



Now update the Director to Sam Mendes

Check your data:



**6.4.1**

Then, delete document 5 using REMOVE command.

You should get a confirmation:



Now, check to see if the collection has the correct data using the command:

**db.video\_records.find().pretty()**

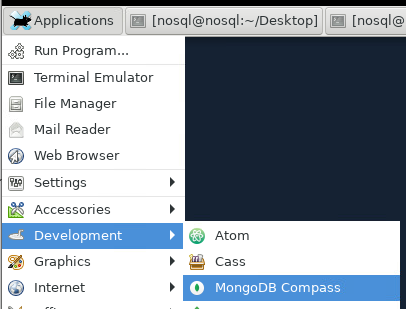
You should have only 4 documents in your collection.

**Lab 6.5 Using MongoDB Compass.**

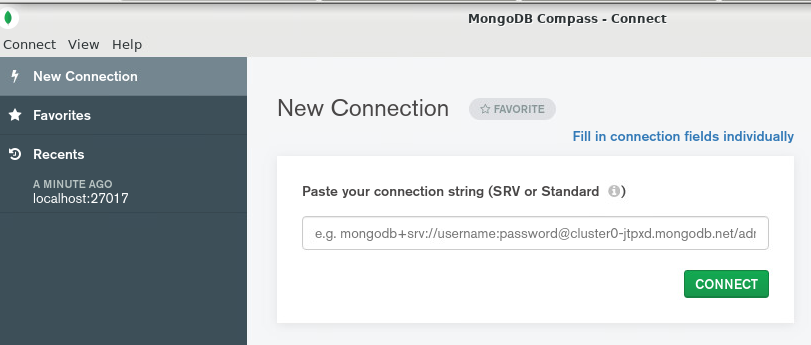
**6.5.1**

In this step you will use GUI that MongoDB provides with every MongoDB database, called MongoDB Compass.

Navigate to Applications->Development-> MongoDB Compass.



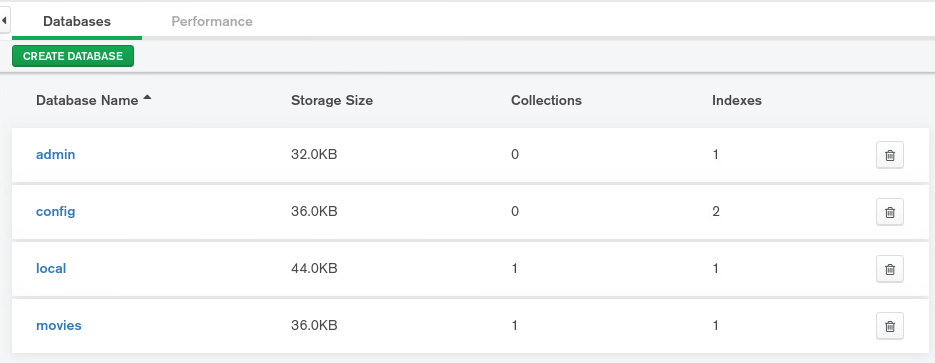
On the New Connection screen, just click on Connect without putting anything into connection field:



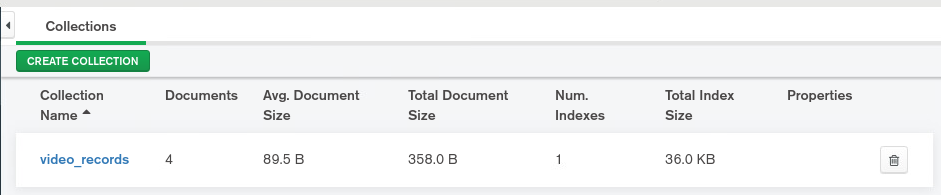
On the next screen, put password= **nosql** and press Unlock



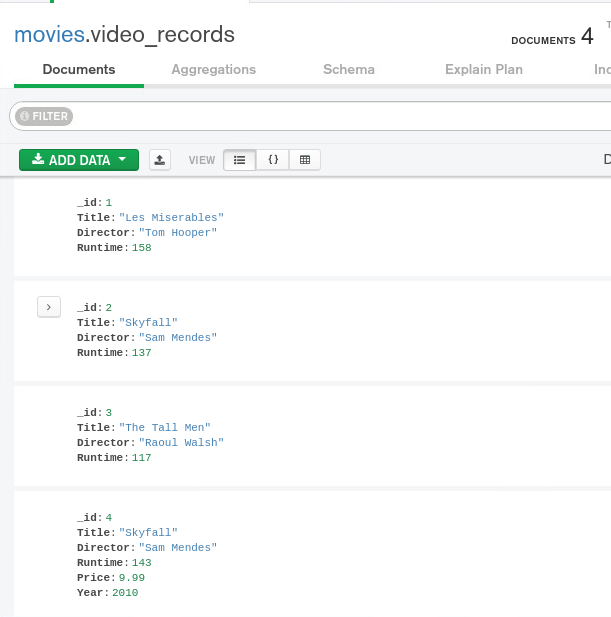
It will open the home page for all your databases:



Click on movies database, it will open all collections you have in that database:



Click on video\_records to see all documents.



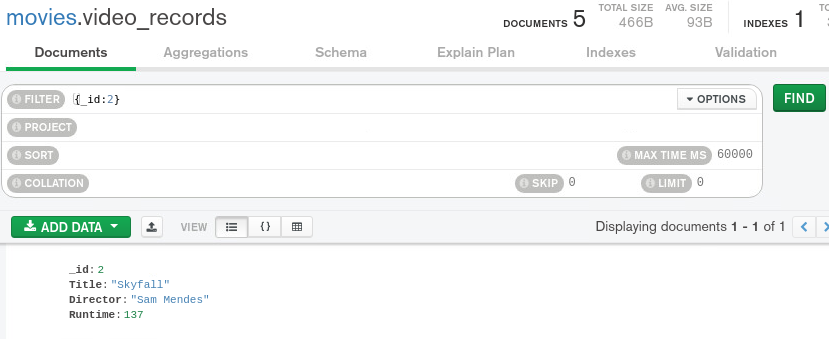
**6.5.2**

Now you need to perform some queries with the ‘where’ conditions, as you did in the command line.

MongoDB Compass uses **Filter** for that.

1. We want to see only document #2:

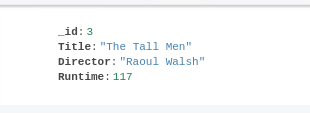
Click on the button **Options** and put the condition in the **FILTER** and the press **FIND:**



Using Lab 6.3 steps as you guide, perform the same queries in the MongoDB Compass:

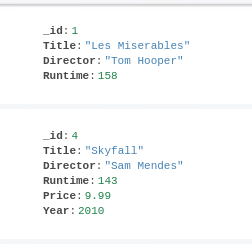
2. Select all documents from the collection **video\_records** that have a director Raoul Walsh.

You should get:



3. Select all the documents from the video recording collection which have a runtime longer than 140 minutes.

You should get this:



4. Select all documents from the video recording collection which have a runtime between 110 and 140 minutes. The result should be:



5. Select all documents from the video recording collection which have a runtime of less than 140 and not equal to 117. You should get this output:



6. Select all documents from the collection video\_records that DO NOT have a director Raoul Walsh

**OR** their Runtime is longer than 140 minutes. You should get this output:



**6.5.3.**

You can insert a document with more complex data type, for example, an array.

Insert a document with the following data:

\_id:5,

Title: “True Grit”,

Director:[“Joel Coen”, Ethan Coen”],

Runtime:110,

Year: 2010

Then perform some query either in the command line or in MongoDB Compass.

**Optional** **Lab 6.6 Using projection, sort, and skip.**

**6.6.1**

We will be covering projection, sorting data, skipping data, etc in the next week lecture.

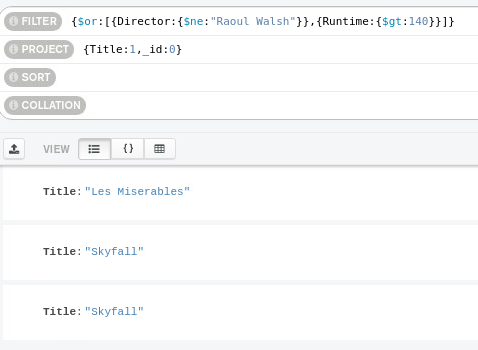
But if you want to try, you can use MongoDB Compass and put conditions into PROJECT filed to specify which fields should be included into the results. In order to sort your output you will include condition into SORT.

All condition should follow the same rule, key-value pairs.

For example, in our last query, if I include condition **{Title:1}** (1 means true) in the PROJECT area**,** it will tell MongoDB, to only include Title in the output:

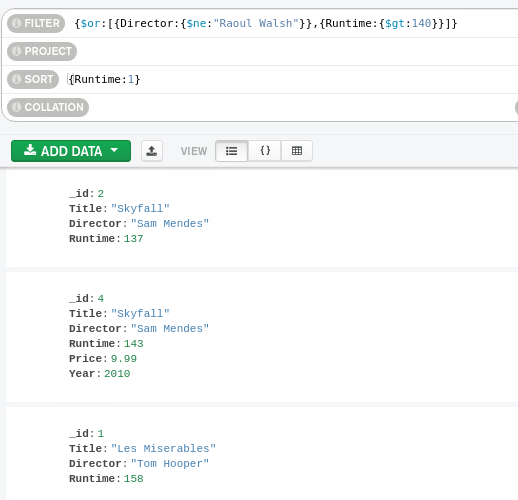


Notice, that \_id is still included, as it is mandatory field. If you do not want to display it, you would nee to add explicit condition {\_id:0} (0 means false)



**6.6.1**

Similarly, you can include a key-value pair to notify MongoDB how you would like your output be sorted. For example, if you would like the output to be sorted by the Runtime, you need to add {Runtime:1} in the SORT area:



**6.6.1**

If you specify LIMIT as 1, it will mean to return only the first document that matches your condition:



If you need to display only the second document, then you need to tell MongoDB to skip the first document by specifying 1 in the SKIP field and LIMIT the output only to 1:

