



Hands-On Workshop

Part 1 - Atlas

Overview

This hands-on workshop is designed to get you familiar with all aspects of MongoDB, from deploying a cluster, to loading and analyzing data, and finally to creating services to access that data.

This workshop includes 11 lab exercises and several more optional lab exercises you can try as time allows. Don't worry about completing all optional lab exercises in this sitting. The free environment you create in this lab will be yours forever.

Prerequisites

To successfully complete this workshop:

- Privileges to install software on your computer. We will be installing [MongoDB Compass](#) in this workshop.

Hands-on Labs

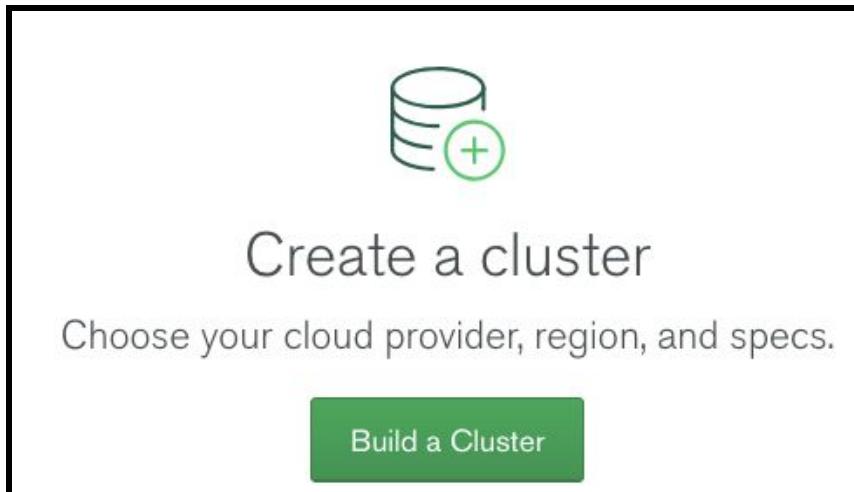
Lab 1 - Create the Cluster

Create an Account or Log In to Atas

We'll be using [MongoDB Atlas](#), our fully managed MongoDB-as-a-service, for this workshop. Go to <https://cloud.mongodb.com> and either create a new account or log into an existing account you may have previously created.

Create a Free Tier Cluster

Click **Build a Cluster**:



Take a moment to browse the options (Provider & Region, Cluster Tier, Version, Backup, ...). For our workshop, you can select **ANY** Cloud Provider but let's select AWS for consistency.

Cloud Provider & Region AWS, N. Virginia (us-east-1) ▾

AWS Google Cloud Platform Azure

Create a **free tier cluster** by selecting a region with **FREE TIER AVAILABLE** and choosing the M0 cluster tier below.

★ recommended region ⓘ

NORTH AMERICA	EUROPE	AUSTRALIA
N. Virginia (us-east-1) ★ FREE TIER AVAILABLE	Stockholm (eu-north-1) ★	Sydney (ap-southeast-2) ★
Ohio (us-east-2) ★	Ireland (eu-west-1) ★ FREE TIER AVAILABLE	
N. California (us-west-1)	London (eu-west-2) ★	Tokyo (ap-northeast-1) ★
Oregon (us-west-2) FREE TIER AVAILABLE	Paris (eu-west-3) ★	Seoul (ap-northeast-2)
Montreal (ca-central-1)	Frankfurt (eu-central-1) ★ FREE TIER AVAILABLE	Singapore (ap-southeast-1) ★ FREE TIER AVAILABLE
SOUTH AMERICA		Mumbai (ap-south-1) FREE TIER AVAILABLE
		Sao Paulo (sa-east-1)

and set the Cluster Name to **Workshop**:

The screenshot shows a user interface for creating a MongoDB cluster. On the left, there is a label "Cluster Name". To its right is a dropdown menu with "Workshop" selected. Below the dropdown is a note: "One time only: once your cluster is created, you won't be able to change its name." Underneath the dropdown, there is a text input field containing "Workshop". A tooltip below the input field states: "Cluster names can only contain ASCII letters, numbers, and hyphens." At the top right of the dropdown, there is a small downward arrow icon.

The remaining defaults will suffice.

Click **Create Cluster**:

The screenshot shows a confirmation message: "Free forever! Your MO cluster is ideal for experimenting in a limited sandbox. You can upgrade to a production cluster anytime." To the right of the message are two buttons: "Cancel" and a green "Create Cluster" button.

Continue to Lab 2 while the cluster is provisioning.

Lab 2 - Connect to the Cluster

2.A. Install Compass

[Compass](https://www.mongodb.com/download-center/compass) is the GUI for MongoDB. Go to <https://www.mongodb.com/download-center/compass> to download and install Compass for your platform. Note, there are several editions of Compass. Make sure you download the “Stable” edition:

MongoDB Compass

As the GUI for MongoDB, MongoDB Compass allows you to make smarter decisions about document structure. MongoDB Compass is available as part of our subscriptions.

MongoDB Compass is available in several versions, described below. For more information on version differences, see the [MongoDB Compass Release Notes](#).

Version	Platforms
<input checked="" type="checkbox"/> 1.18.0 (Stable) 1.18.0 (Community Edition Stable) 1.18.0 (Readonly Edition Stable) 1.18.0 (Isolated Edition Stable) 1.19.0-beta.1 (Beta) 1.19.0-beta.1 (Community Edition Beta) 1.19.0-beta.1 (Readonly Edition Beta) 1.19.0-beta.1 (Isolated Edition Beta)	OS X 64-bit (10.10+)

2.B. Setup Connection Security in Atlas

Return to the Atlas UI. Your cluster should now be provisioned. Click the **CONNECT** button, which will prompt you to set up connection security:

Connect to Workshop

Setup connection security > Choose a connection method > Connect

You need to secure your MongoDB Atlas cluster before you can use it. Set which users and IP addresses can access your cluster now. [Read more](#)

You can't connect yet. Set up your firewall access and user security permission below.

1 Whitelist your connection IP address

Add Your Current IP Address Add a Different IP Address

2 Create a MongoDB User

This first user will have **atlasAdmin** permissions for all clusters in this project. Keep your credentials handy, you'll need them for the next step.

Username	Password
ex. dbUser	<input type="password"/> Autogenerate Secure Password ex. dbUserPassword Show

Create MongoDB User

Close **Choose a connection method >**

Add Your Current IP Address and **Create a MongoDB User**. I'm using Username **workshop** and password **workshop**:

You can't connect yet. Set up your firewall access and user security permission below.

1 Whitelist your connection IP address

IP Address	Description (Optional)
97.76.196.230	Hilton Garden Inn
<button>Cancel</button> <button>Add IP Address</button>	

2 Create a MongoDB User

This first user will have **atlasAdmin** permissions for all clusters in this project.

Keep your credentials handy, you'll need them for the next step.

Username	Password	<input type="button" value="Autogenerate Secure Password"/>
workshop	*****	<input type="button" value="SHOW"/>
<input type="button" value="Create MongoDB User"/>		

Click **Choose a connection method** and select **I have Compass**.

Where you choose your version of Compass, select **1.12 or later** and **COPY** the connection string presented:

X

Connect to Cluster0

✓ Setup connection security

✓ Choose a connection method

Connect

I do not have Compass

I have Compass

1 Choose your version of Compass:

1.12 or later

See your Compass version in "About Compass"

2 Copy the connection string below, and open Compass:

```
mongo "mongodb+srv://cluster0-nsdia.mongodb.net/test" --  
username workshop
```

 Copy

You will be prompted for the password for the **workshop** user's (MongoDB User) username.
When entering your password, make sure that any special characters are **URL encoded**.

Having trouble connecting? [View our troubleshooting documentation](#)

Go Back

Close

Connect Compass

Start Compass and it should detect the connection string in your copy buffer:

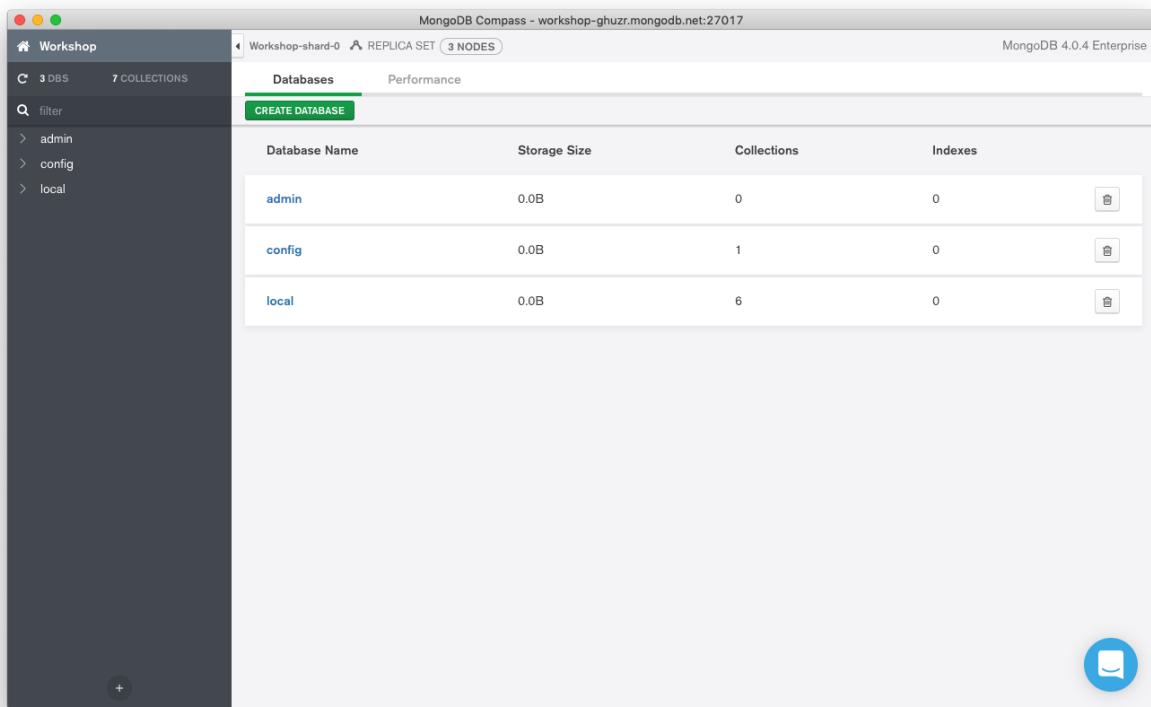


Select **Yes**.

Make sure that the **SRV Record** is selected and the authentication is set to username/password. Provide the password (workshop) and *before clicking CONNECT, CREATE a FAVORITE named Workshop*. This will allow us to quickly connect to the cluster in the future.

Click **CONNECT**.

If successful, you'll see some internal databases used by MongoDB:

A screenshot of the MongoDB Compass application interface. The title bar reads "MongoDB Compass - workshop-ghuzr.mongodb.net:27017" and "MongoDB 4.0.4 Enterprise". The left sidebar shows a project named "Workshop" with 3 DBS and 7 COLLECTIONS. Under "filter", there are entries for "admin", "config", and "local". The main pane is titled "Databases" and shows a table of databases. The table has columns: Database Name, Storage Size, Collections, and Indexes. It lists three databases: "admin" (Storage Size 0.0B, Collections 0, Indexes 0), "config" (Storage Size 0.0B, Collections 1, Indexes 0), and "local" (Storage Size 0.0B, Collections 6, Indexes 0). Each database row has a small edit icon on the right. A "CREATE DATABASE" button is located at the top of the table area. The bottom right corner of the window has a blue circular icon with a white speech mark symbol.

Lab 3 - Load Data

For your convenience, Atlas offers a new “Load Sample Data” feature to allow you to load and explore six different datasets (350MB) with a click of a button.

Return to your Atlas UI, click on the ellipsis ... button to Load Sample Dataset.

The screenshot shows the Atlas UI interface. At the top, there's a navigation bar with tabs: 'Sandbox' (highlighted in blue), 'Workshop' (highlighted with a green dot), 'Metrics', 'Collections', and an ellipsis (...). Below the tabs, the 'Workshop' section displays the following details:

- Version:** 4.0.10
- CONNECT**, **METRICS**, **COLLECTIONS** buttons
- INSTANCE SIZE:** M0 Sandbox (General)
- REGION:** AWS / N. Virginia (us-east-1)
- TYPE:** Replica Set - 3 nodes
- LINKED STITCH APP:** None Linked

A context menu is open at the position of the ellipsis (...). The menu items are:

- Edit Configuration
- Command Line Tools
- Load Sample Dataset** (highlighted with a blue border)
- Terminate

On the right side of the interface, there are two large input fields with sliders. The top field is labeled "Operations R: 0 W: 0" with a slider scale from 0 to 100.0/s. The bottom field is labeled "Last 6 Hours" with a slider scale from 0 to 100 max.

A modal dialog box titled "Load Sample Dataset" is displayed. The dialog contains the following text:

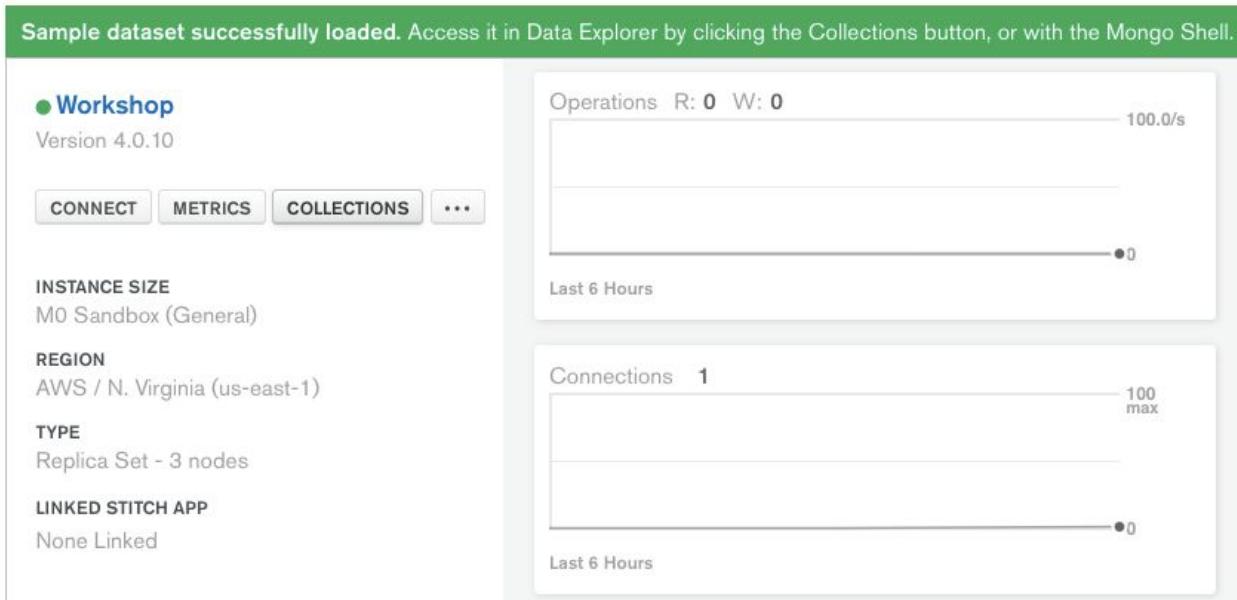
We've created a sample dataset to help you test features on Workshop.
Sample Dataset
Size: ~350 MB

Please confirm that you want to load this sample dataset.

At the bottom of the dialog are two buttons: "Cancel" (gray) and "Load Sample Dataset" (green).

The background of the dialog shows a blurred view of the Atlas UI, including the "Workshop" tab, version information, and some metrics like "Disk IOPS" and "Logical Size".

When the loading is done, you can click on the Collections tab inside the Atlas UI to show 6 different databases of 19 different collections.



- [sample_airbnb](#) has one collection of AirBnB reviews and listings, already indexed by property type, room type, and bed, name and location.
- [sample_geospatial](#) is a collection of shipwrecks and their locations, indexed for searching.
- [sample_mflix](#) is a database with five collections all about movies, movie theatres and the metadata like users, comments and sessions. Get visualizing that data with a [MongoDB Charts tutorial](#) which already uses the mflix dataset.
- [sample_supplies](#) is a typical sales data collection from a mock office supplies company. There's a [MongoDB Charts tutorial which shows how to visualize it](#).
- [sample_training](#) is a database with nine collections used in [MongoDB's private training](#). It's based on a range of well known data sources such as [Openflights](#), [NYC's OpenData](#) and [Twitter's Decahose](#).
- [sample_weatherdata](#) is another collection loaded with geodata, this time for the locations of weather reports on temperature, wind and visibility.

(Feel free to explore these datasets and the various accompanying tutorials.)

For this workshop, we will use the [movies](#) collection in the [sample_mflix](#) database. This collection contains details on movies. Each document contains a single movie, and information such as its title, release year and cast.

VERSION 4.0.10 REGION

DATABASES: 6 COLLECTIONS: 18

[+ Create Database](#)

NAMESPACES

- sample_airbnb
- sample_geospatial
- sample_mflix**
 - comments
 - movies**
 - sessions
 - theaters
 - users
- sample_supplies
- sample_training
- sample_weatherdata

sample_mflix.movies

COLLECTION SIZE: 35.9MB TOTAL DOCUMENTS: 23539 INDEXES TOTAL SIZE: 13.14MB

Find Indexes Aggregation

FILTER {"filter": "example"}

QUERY RESULTS 1-20 OF MANY

```
_id: ObjectId("573a1390f29313caabcd4135")
plot: "Three men hammer on an anvil and pass a bottle of beer around."
> genres: Array
> runtime: 1
> cast: Array
> num_mflix_comments: 1
title: "Blacksmith Scene"
fullplot: "A stationary camera looks at a large anvil with a blacksmith behind it..."
> countries: Array
released: 1893-05-09T00:00:00.000+00:00
directors: Array
rated: "UNRATED"
> awards: Object
lastupdated: "2015-08-26 00:03:50.133000000"
year: 1893
> imbd: Object
type: "movie"
> tomatoes: Object
```

Lab 4 - Browse the Documents in Compass

We will return to Compass to browse and analyze the sample_mflix.movies collection. In Compass, select the **Documents** tab if it is not already selected. You will see this collection has 23539 movie documents.

My Cluster

9 DBS 26 COLLECTIONS

filter

- admin
- config
- local
- sample_airbnb
- sample_geospatial
- sample_mflix**
 - comments
 - movies**
 - sessions
 - theaters
 - users
- sample_supplies
- sample_training
- sample_weatherdata

Workshop-shard-0 REPLICA SET (3 NODES)

MongoDB 4.0.10 Enterprise

sample_mflix.movies

DOCUMENTS 23.5k TOTAL SIZE 35.9MB AVG. SIZE 1.6KB INDEXES 2 TOTAL SIZE 13.1MB AVG. SIZE 6.6MB

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER

INSERT DOCUMENT VIEW LIST TABLE

Displaying documents 1 - 20 of 23539

```
_id: ObjectId("573a1390f29313caabcd4135")
plot: "Three men hammer on an anvil and pass a bottle of beer around."
> genres: Array
> runtime: 1
> cast: Array
> num_mflix_comments: 1
title: "Blacksmith Scene"
fullplot: "A stationary camera looks at a large anvil with a blacksmith behind it..."
> countries: Array
released: 1893-05-09T00:00:00.000+00:00
directors: Array
rated: "UNRATED"
> awards: Object
lastupdated: "2015-08-26 00:03:50.133000000"
year: 1893
> imbd: Object
type: "movie"
> tomatoes: Object
```

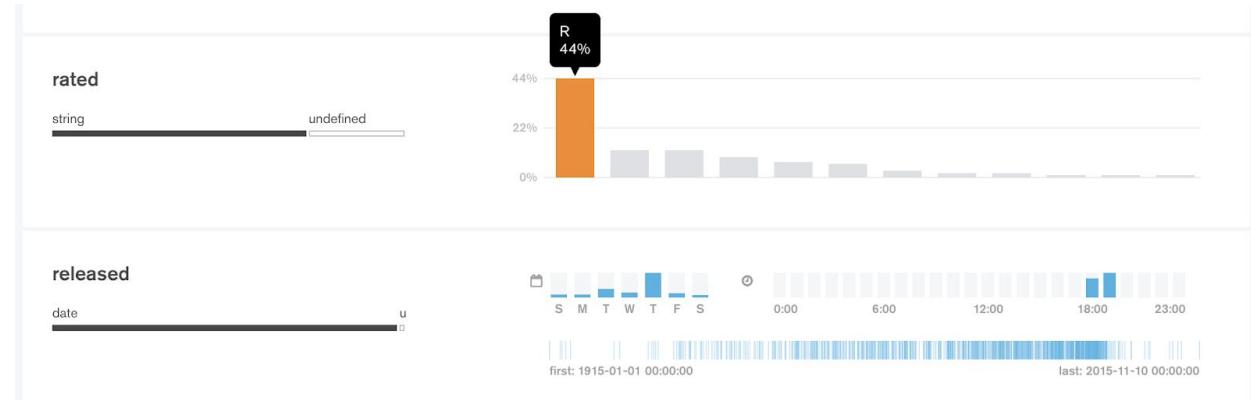
Examine the documents in the collection. Notice how each document has several fields such as `_id`, `title`, and `plot`. The movie documents also have nested subdocuments (`awards`, `imdb`, `tomatoes`) and an arrays (`cast`, `countries`, `directors`). In a relational database, these fields would most likely be separate tables, but MongoDB allows us to embed this information. Working with data in this natural way is much **easier** than decomposing and composing from relational tables.

Lab 5 - View the Schema

You might be thinking, “Wait, I thought MongoDB is a NoSQL database, and hence, didn’t have a schema?” While that’s technically true, no dataset would be of any use without a schema. So while MongoDB doesn’t enforce a schema, your collections of documents will still always have one. The key difference with MongoDB is that the schema can be **flexible**.

Continuing to work in the `movies` collection, select the **Schema** tab and click **Analyze Schema**. Compass will sample the documents in the collection to derive a schema. In addition to providing field names and types, Compass will also provide a summary of the data values.

For example, for the `rated` field, we can see that **R** is the most common type at 44% and most movies are released on a Thursday (your results may differ slightly based on the sample that was taken) and :



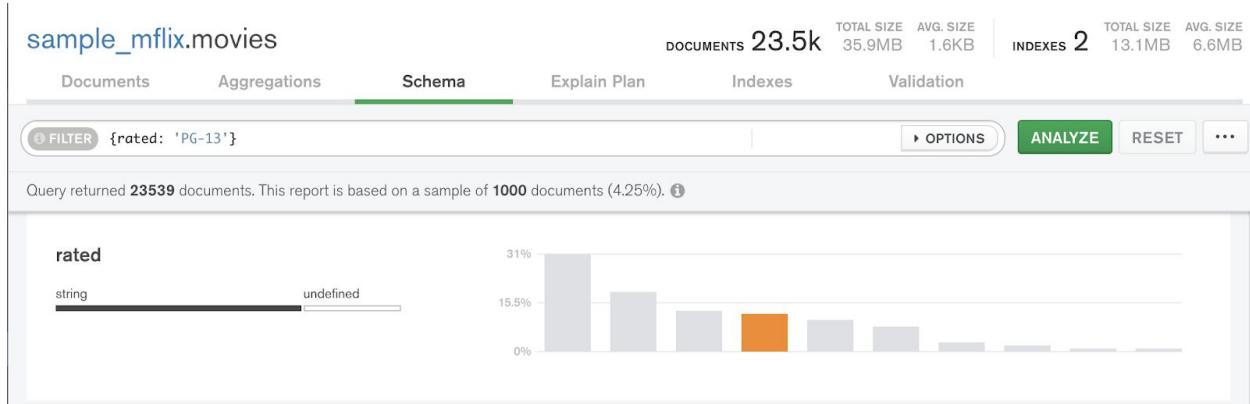
Lab 6 - Query the Data

The MongoDB Query Language (MQL) is based on JSON and provides rich query features for fast, flexible, and highly scalable data. Although you won’t become an expert in every aspect of the MongoDB Query Language today, we will work through a few fun and basic examples to get you started with your MongoDB data in your application.

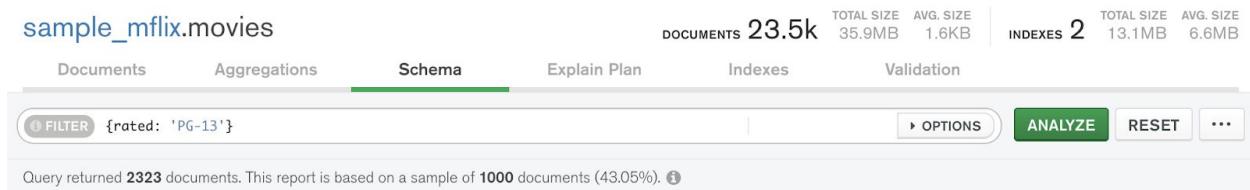
The Schema Analyzer in Compass provides an easy way to learn the language.

6.A. Basic Finds:

For example, when you select the **PG-13** from the **rated** field and notice as you made the selection, the FILTER field at the top of the window gets populated with **{rated: 'PG-13'}**.



Click the **ANALYZE** button to filter for PG-13, of which there are 2323:



To query from the MongoDB shell or your application, MongoDB provides the following methods to read documents from a collection:

```
db.collection.find()
```

So in order to find these 2323 MongoDB documents from your movies collection from your application, you will write:

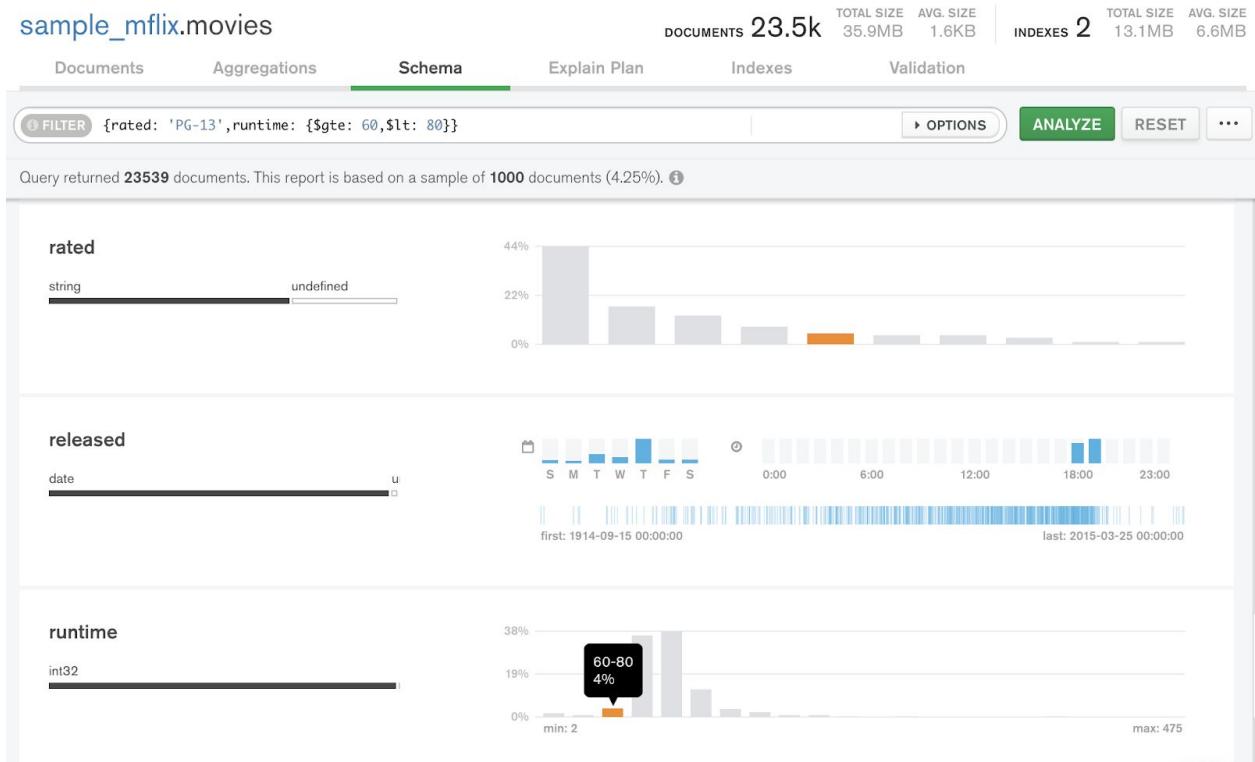
```
db.movies.find({<<INSERT FILTER>>})
```

or

```
db.movies.find({rated: 'PG-13'})
```

6.B. Multiple Conditions:

You can list multiple conditions in your query by inserting a comma in between conditions. Let's add another condition as a range by making a selection in the **runtime** field:



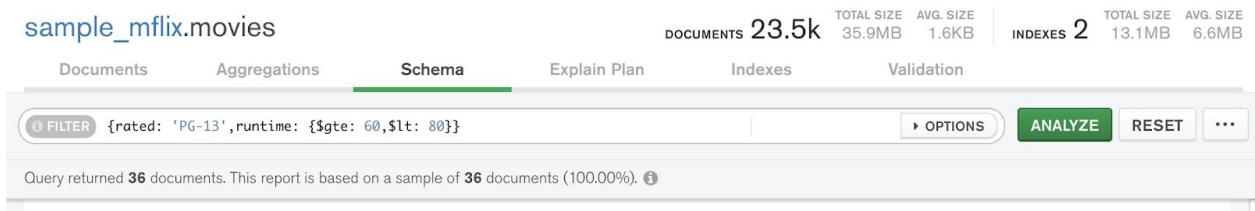
In the screenshot above, to query for movies in the 60-80 minute range, MongoDB query language adds `runtime:{$gte: 60, $lt:80}` to the query. Feel free to play with the values to look for movies for any duration.

Note that selectors in MongoDB are ANDed together by default with a comma. So in the MongoDB Query Language, to find movies that are BOTH rated PG-13 with runtimes between 60 and 80 minutes, you would write:

```
{ rated:'PG-13', runtime:{ $gte: 60, $lt:80 } }
```

And finding documents that match this query from your application would then be:

```
db.movies.find({ rated:'PG-13', runtime:{ $gte: 60, $lt:80 } })
```



Switch to the Documents tab in Compass to see these 36 movies.

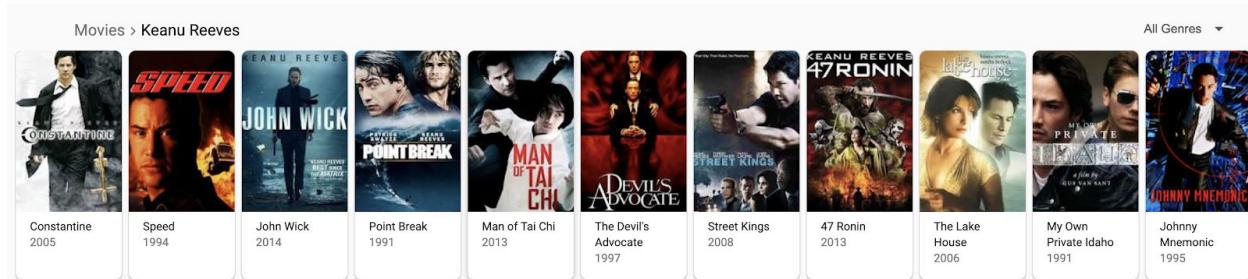
6.C Embedded Documents:

Now let's discuss how to find equality for embedded documents. Go back to schema and scroll down to tomatoes. Open up the Object and click on any of the **boxOffice** values. You will see that to query for a nested field, you will use the dot notation :

Ex. { 'tomatoes.boxOffice': '\$14.8M'}

6.D Querying Arrays:

Finally let's look at exact matches for an array field by looking for our favorite actors in the cast field. I am a Keanu Reeves fan so let's look for some of his movies. 😍❤️



First, let's reset our fields in Compass by clearing the filters and clicking the RESET button. You will notice in any movie document opening the **cast** array will list the main actors as strings in an array.

```
> _id: ObjectId("573a1390f29313caabcd4135")
  plot: "Three men hammer on an anvil and pass a bottle of beer around."
  > genres: Array
    runtime: 1
  < cast: Array
    0: "Charles Kayser"
    1: "John Ott"
  num_mflix_comments: 1
  title: "Blacksmith Scene"
  fullplot: "A stationary camera looks at a large anvil with a blacksmith behind it..."
  > countries: Array
    released: 1893-05-09T00:00:00.000+00:00
  > directors: Array
    rated: "UNRATED"
  > awards: Object
    lastupdated: "2015-08-26 00:03:50.133000000"
    year: 1893
  > imdb: Object
    type: "movie"
  > tomatoes: Object
```

The above document for the movie **Blacksmith Scene** has Charles Kayser and John Ott.

To find movies with Keanu Reeves- or any singular actor, simply do a normal find as if cast is a scalar field:

{ **cast: "Keanu Reeves"** }

and you will find 27 different movie documents:

Now, since it is an array, if you want to find movies with a certain ensemble, **more than 1 actor**, we will use the **\$all** operator:

```
{ cast: { $all:[ "Keanu Reeves", "Sandra Bullock"]} }
```

to return 2 movies: **Speed** and **The Lake House** (which make me completely jealous of Sandra Bullock !)

Now that you have learned some of the fundamentals of the MongoDB query language, play around in Compass or your application to find your favorite movies or ways to impress your friends with movie trivia!

