**MongoDB**

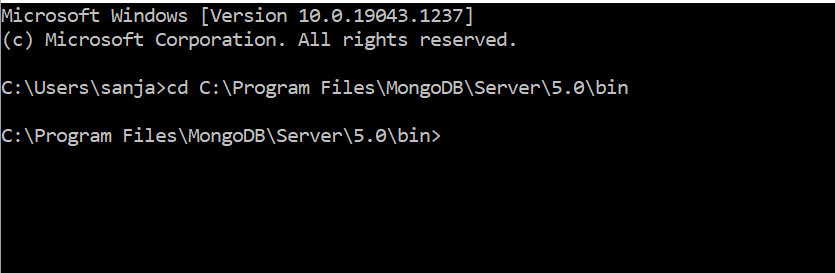
**MongoDB** is an **open-source document** database and leading **NoSQL database**. MongoDB is written in C++.

MongoDB is a **cross-platform, document oriented** database that provides, **high performance, high availability, and easy scalability.**

**Install MongoDB on Windows**

To install MongoDB on Windows, first download the latest release of MongoDB from [**http://www.mongodb.org/downloads**](http://www.mongodb.org/downloads)**.**

In case you have extracted the **MongoDB** at different location, then go to that path by using command **cd FOLDER/DIR.**

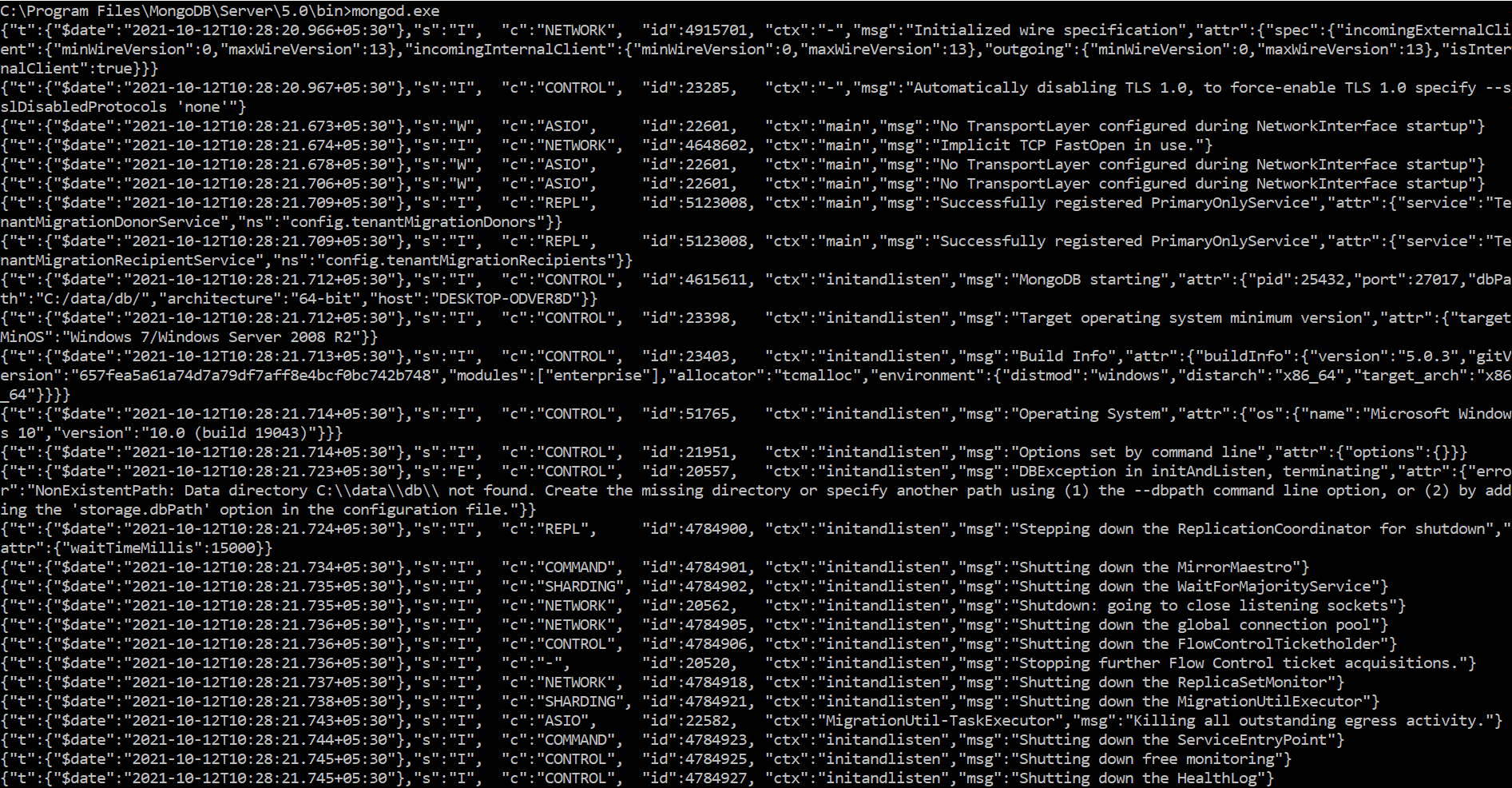


**mongod.exe and mongo.exe**

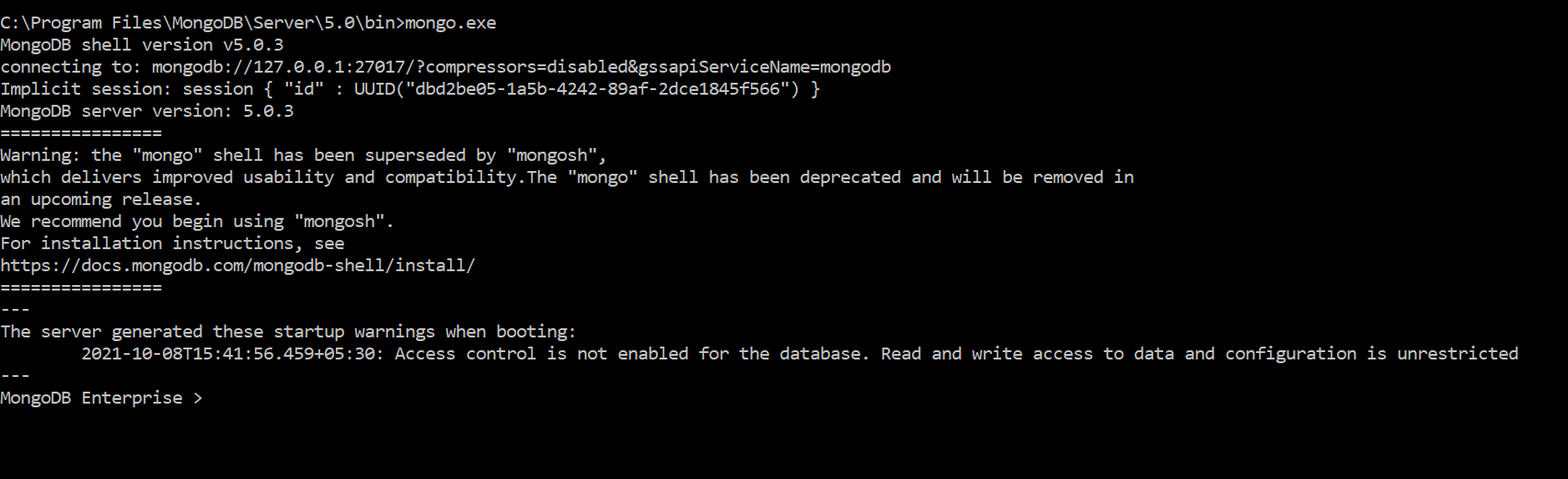
**mongod** is the **daemon background process running** the **database server** and **mongo.exe** is the **interactive shell**.

**exe** is **the build of the MongoDB daemon** (i.e. mongod ) for the **Windows platform**.exe has **all of the features of mongod** on **Unix-like platforms** and is completely **compatible** with the other builds of mongod.

**mongod.exe**



**mongo.exe**

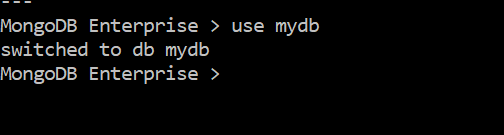


**The use Command**

**MongoDB** use **DATABASE\_NAME** is used to **create database**. The command will create a **new database if it doesn't exist**, otherwise it will **return the existing database**.

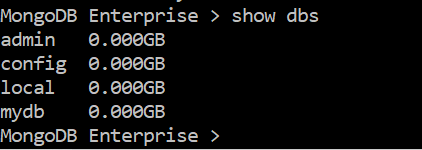
**Syntax :-**

**use DATABASE\_NAME**



**The show Command**

To check your **databases list**, use the command **show dbs**. To display database, you need to insert **at least one document into it**. In **MongoDB** default database is **test**. If you didn't create any database, then **collections will be stored in test database.**

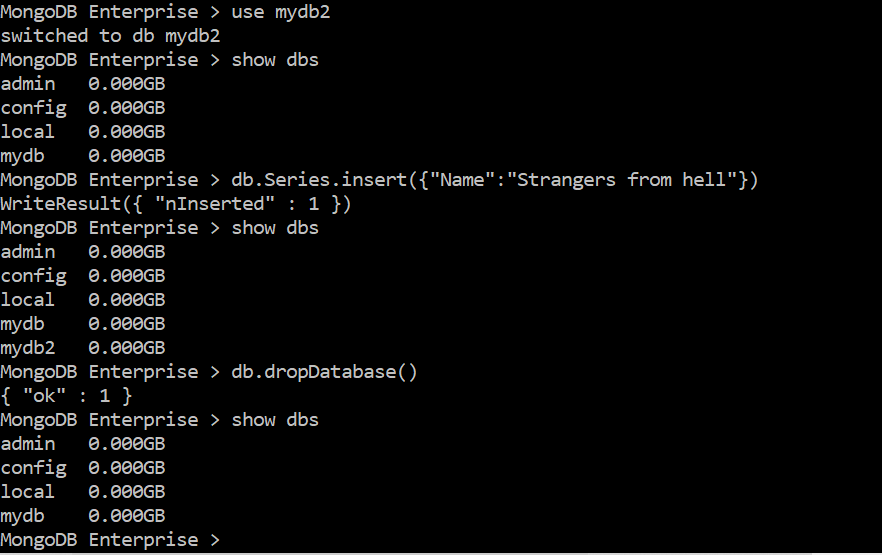


**The dropDatabase() Method**

MongoDB **db.dropDatabase()** command is used to **drop a existing database**. It deletes the **selected database**. If you have not selected any database, then it will delete **default 'test'** database.

**Syntax:-**

db.dropDatabase()



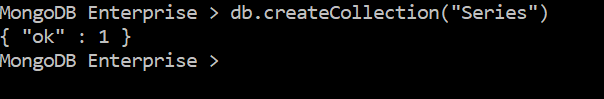
**createCollection() Method**

MongoDB **db.createCollection(name, options)** is used to **create collection.**

**Syntax:-**

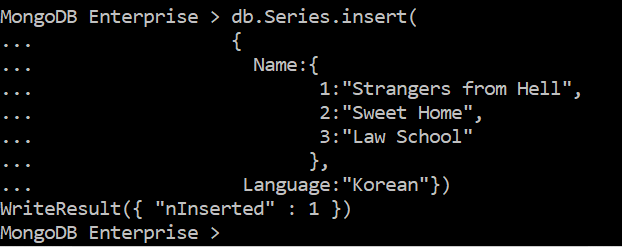
**db.createCollection(name, options)**

In the command, **name is name of collection to be created. Options is a document and is used to specify configuration of collection (Specifies the memory sizing and indexing)**



**MongoDB insert documents**

In MongoDB, the**db.collection.insert()** method is used to **add or insert new documents** into a collection in your database.

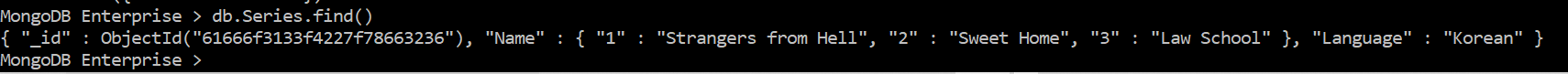


## **Check the inserted documents**

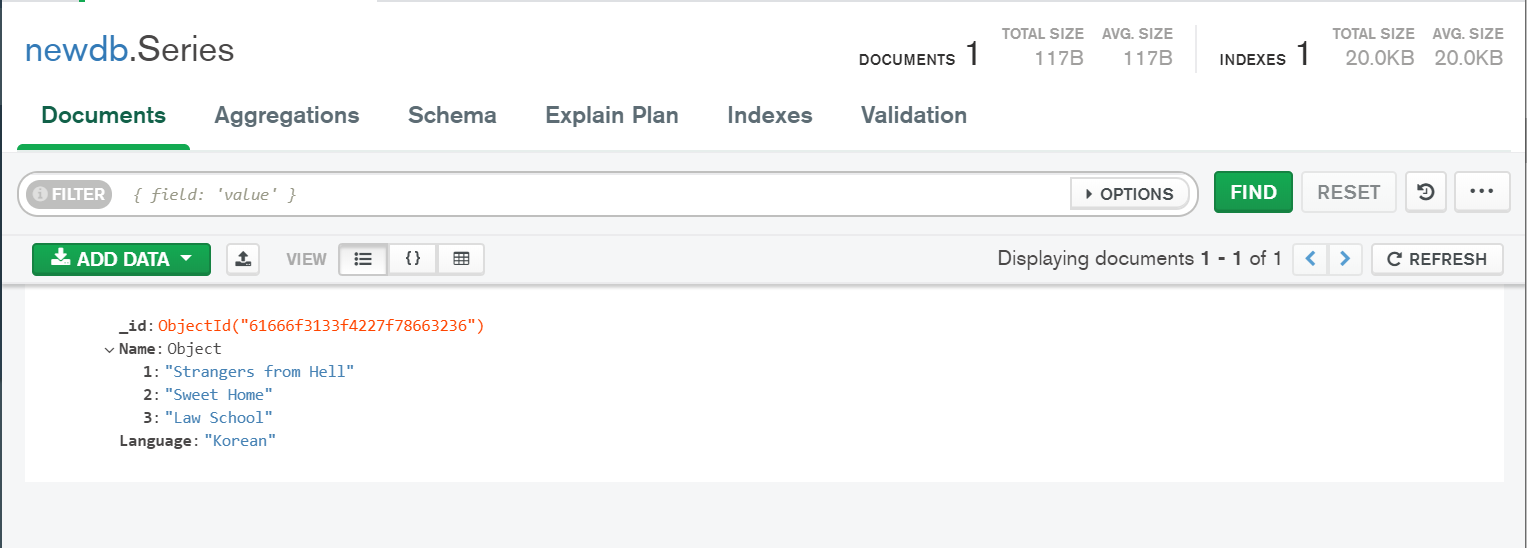
The inserted document can be viewed by the following query.

**Syntax :-**

**db.*collection\_name*.find()**



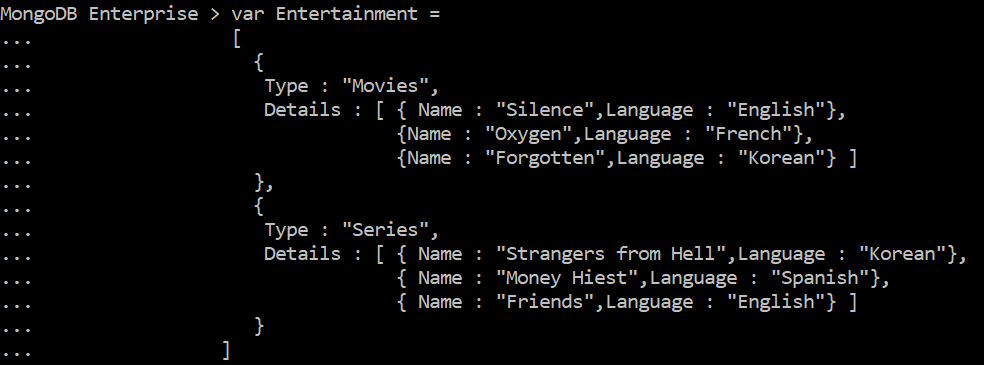
**Or open MongoDB**



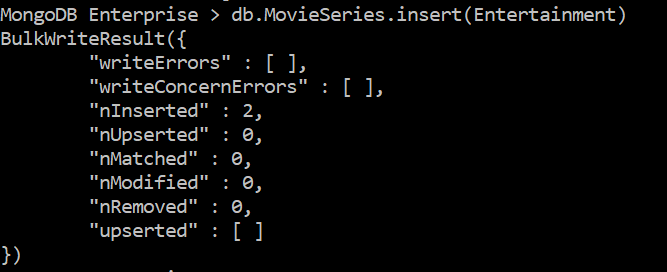
## **MongoDB insert multiple documents**

To insert **multiple documents** in a collection, you have to **pass an array of documents** to the **db.collection.insert()** method.

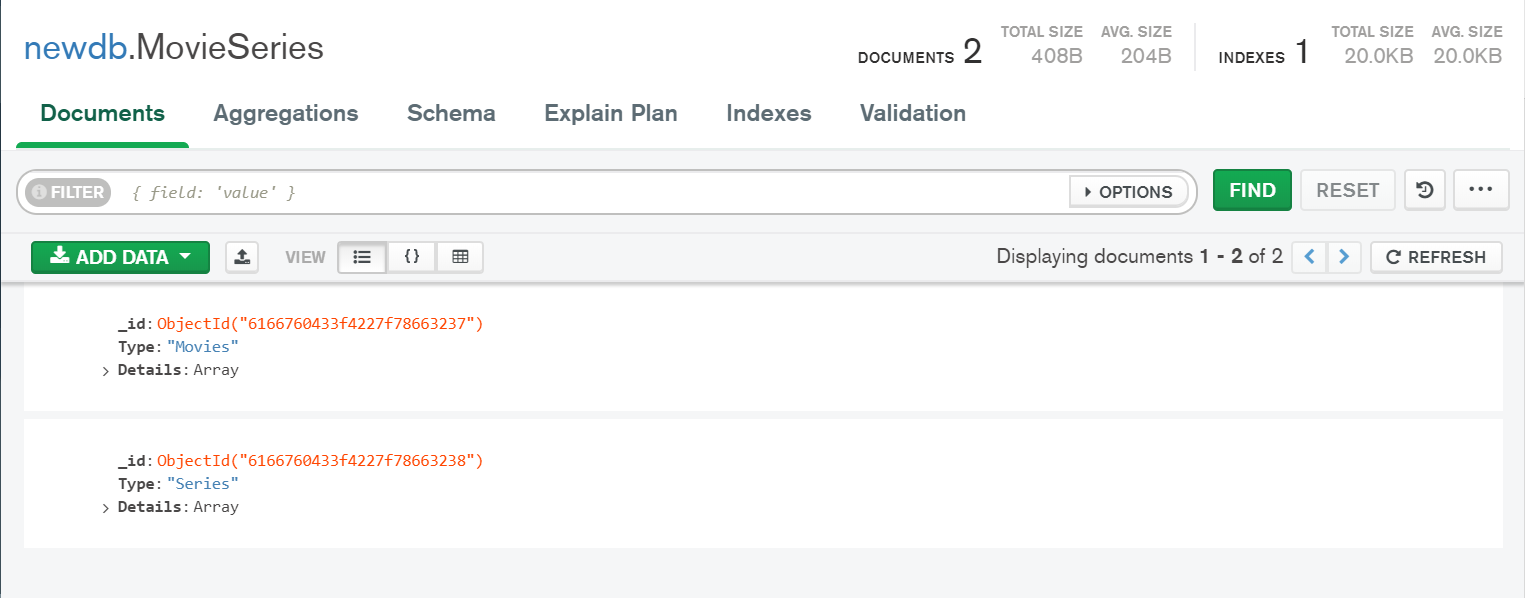
## **Create an array of documents :-**

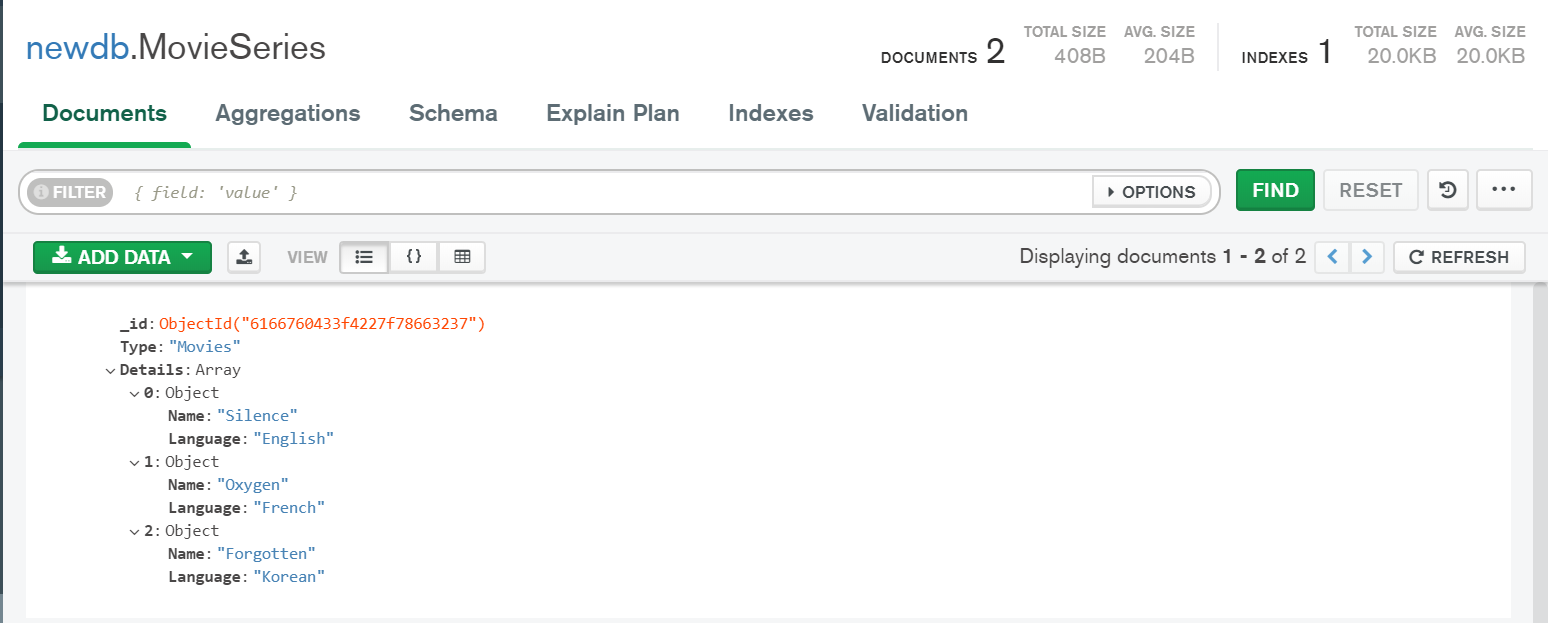


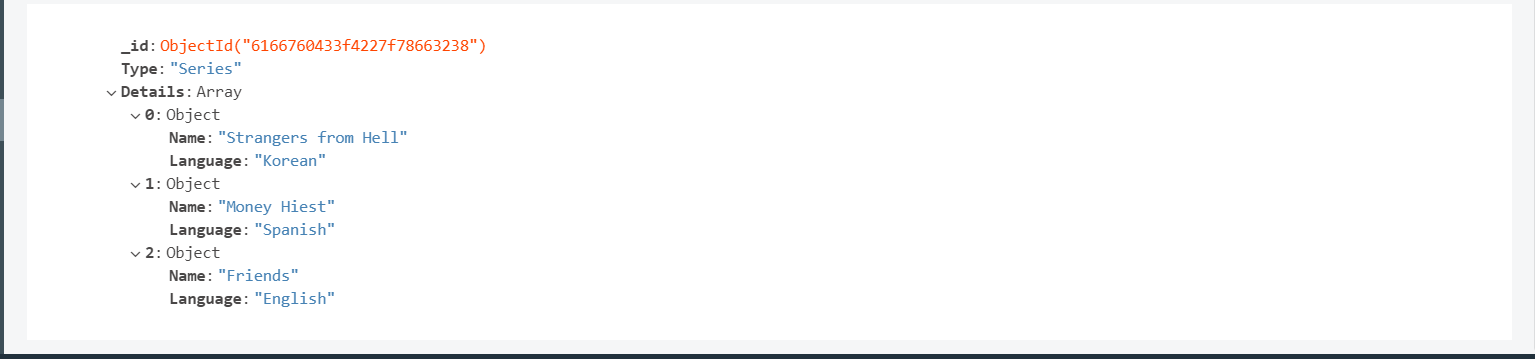
## **Inserts the documents:-**



**Contents:-**





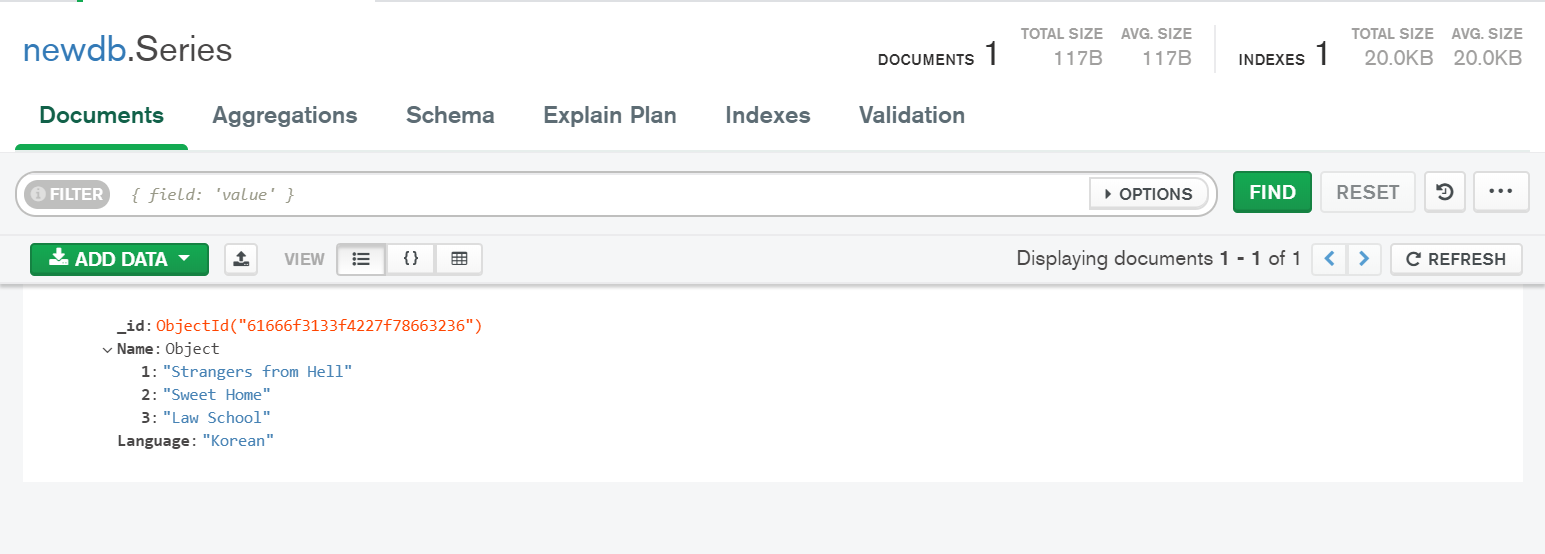


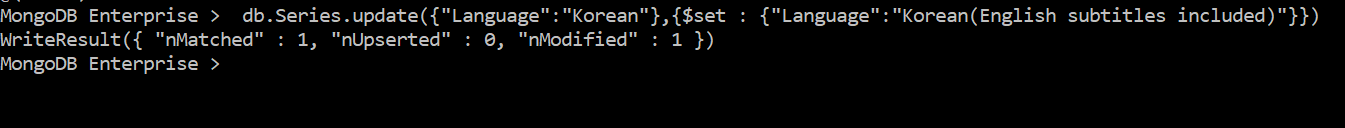
# MongoDB update documents

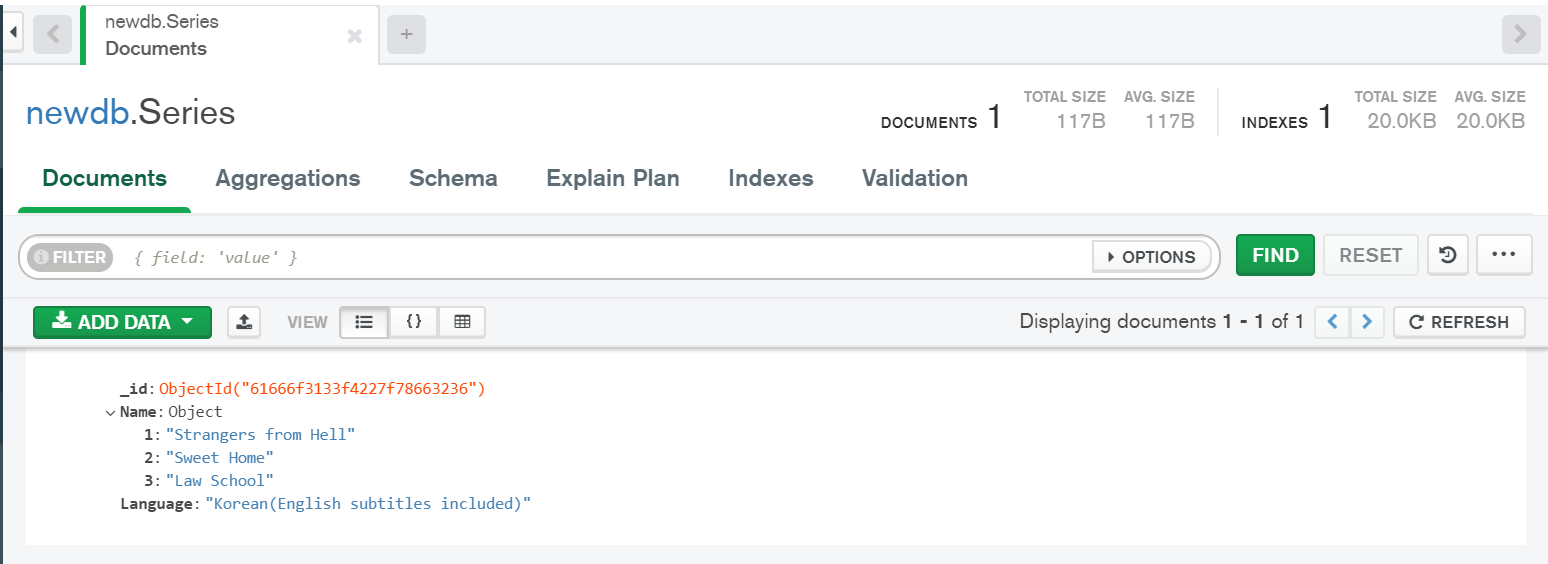
In MongoDB, **update()** method is used to **update or modify the existing documents of a collection.**

**Syntax:-**

**db.COLLECTION\_NAME.update(SELECTIOIN\_CRITERIA, UPDATED\_DATA)**







# MongoDB Delete documents

In MongoDB, the **db.colloction.remove()** method is used to **delete documents from a collection.** The remove() method works on **two parameters**.

**1. Deletion criteria:** With the use of its syntax you can **remove the documents from the collection.**

**2. JustOne:** It removes only **one document when set to true or 1.**

**Syntax :-**

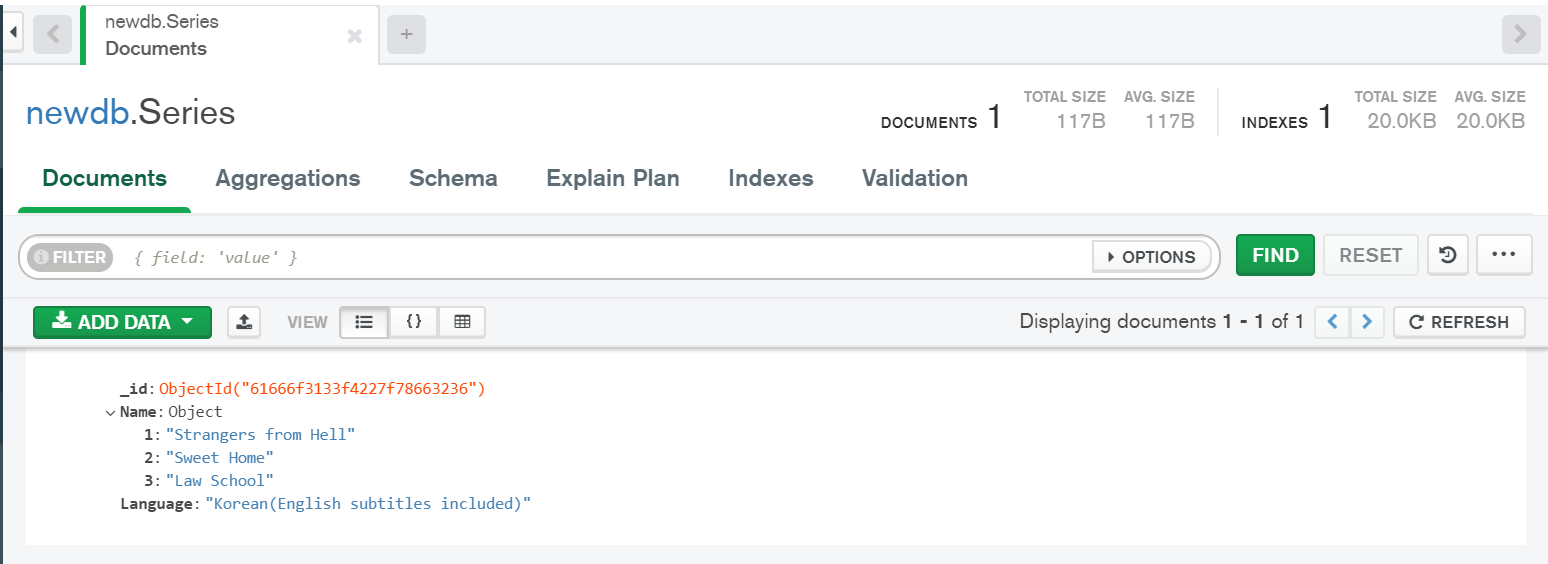
**db.*collection\_name*.remove (DELETION\_CRITERIA)**

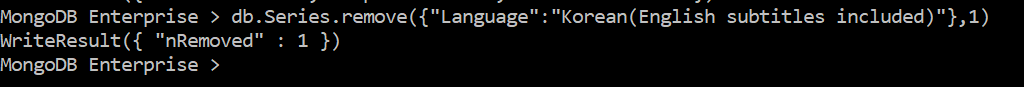
## **Remove all documents**

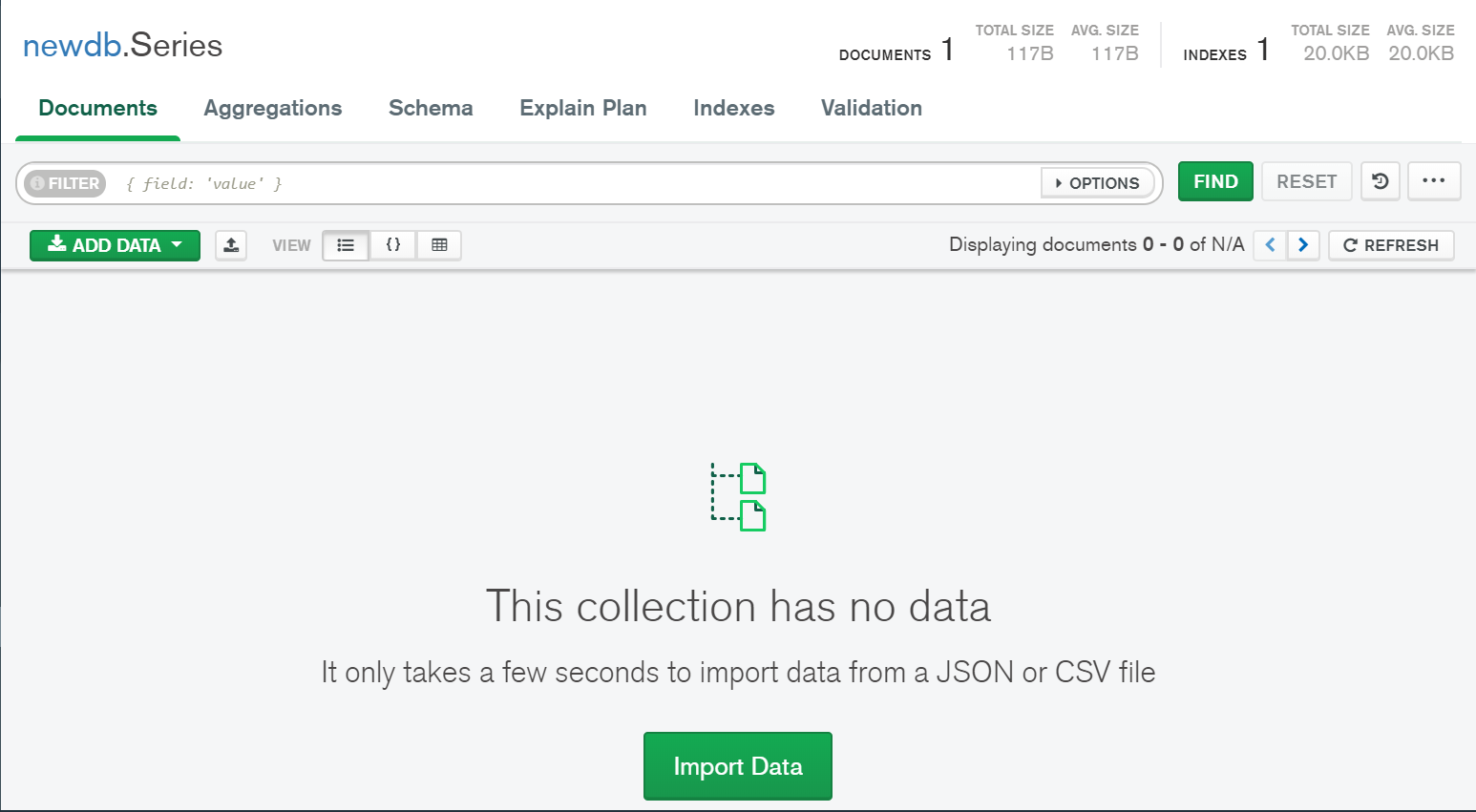
**db.*collection\_name*.remove({})**

## **Remove a single document that match a condition**

**db.*collection\_name*.remove ({DELETION\_CRITERIA},1)**







**Aggregation operations**

* **Aggregation operations** process **data records and return computed results**.
* The **Aggregate** **function** accepts an **array of data transformations** which are applied to the data in the order they're defined.
* This makes aggregation a lot like other data flow pipelines: **the transformations that are defined first will be executed first and the result will be used by** **the next transformation in the sequence.**

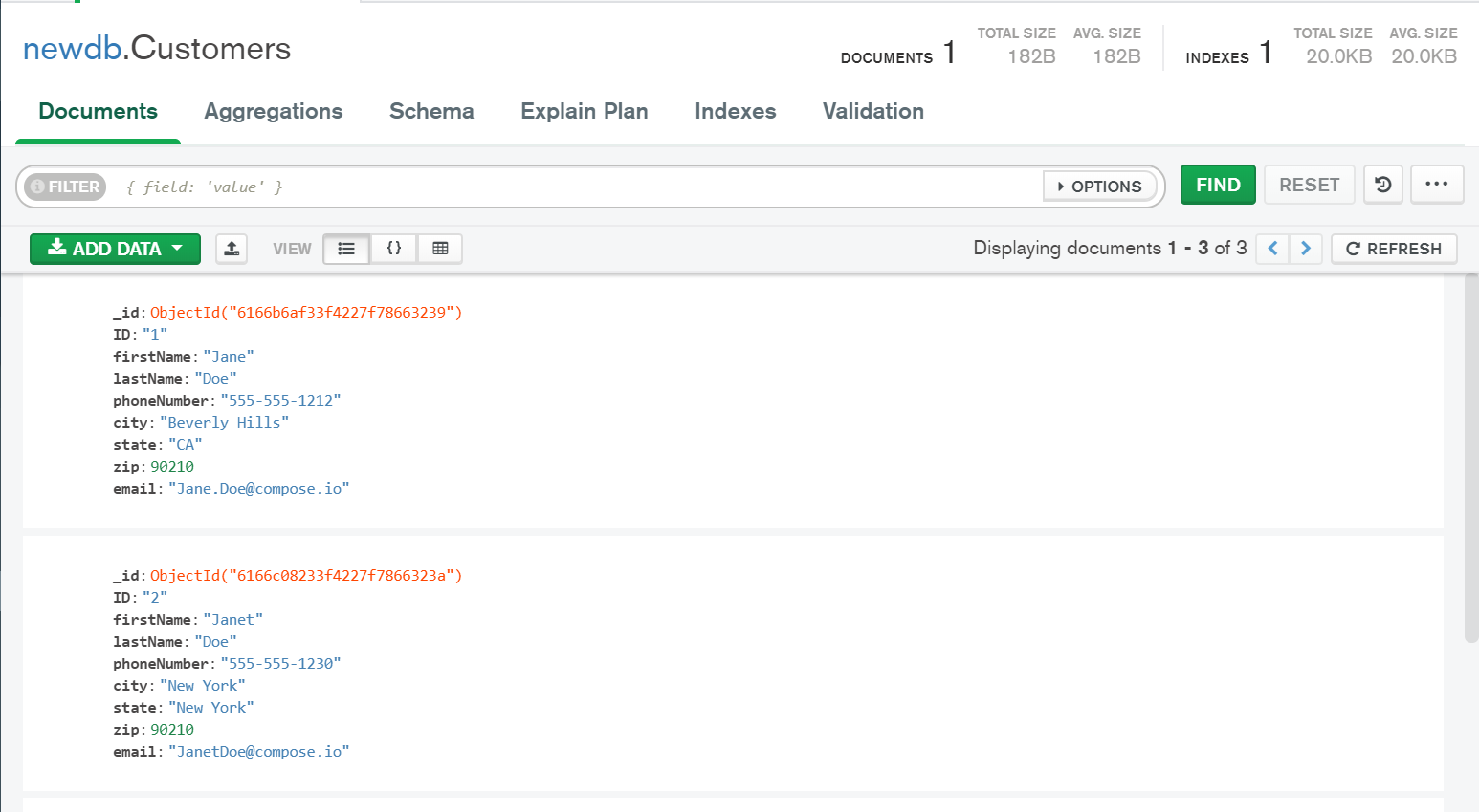
**$match :-**

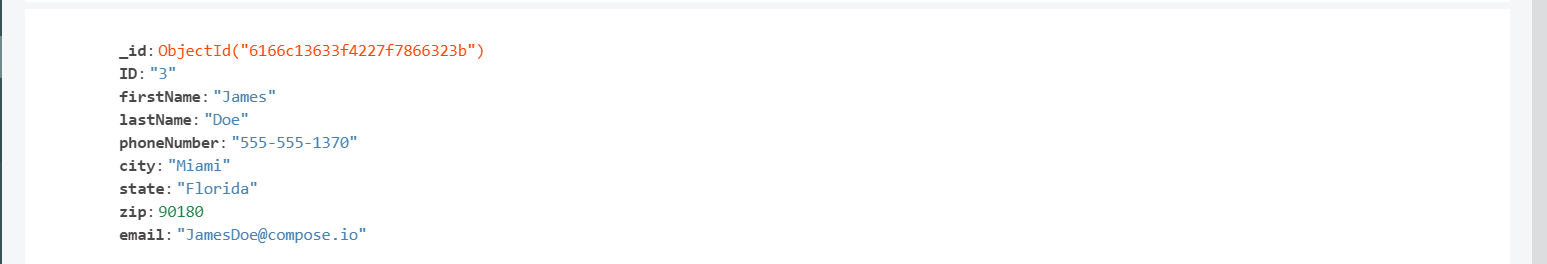
Allows us to **filter out documents** so that we're only manipulating the documents we care about.

The matching expression looks and acts much like the **MongoDB find** function or a **SQL** **WHERE** clause.

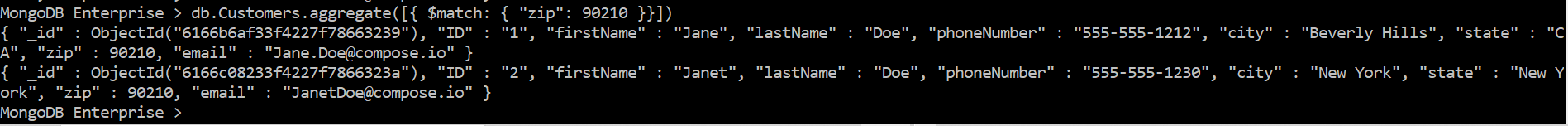
**Syntax:-**

**db.collection\_name.aggregate([{ $match: { value }}]);**









**$group**

This is used to **group the data** from the collection

**$avg**

This is used to calculate the **average of all documents** in the collection.



**$sum**

This is used to calculate the **sum of the defined values in the documents** of the collection.



**$avg**

This is used to find the **maximum of the values in the document** of the collection.



**$min**

This is used to find the **minimum of the values in the document** of the collection.

