MongoDB quick tour

Aim : "Connecting to NoSQL database/s and querying to provide analysis using api like aggregation, etc. To be able to successfully import/export from/to CSV."

MongoDB

NoSQL – Not Only SQL database/s have evolved because of the requirement of handling schema less/free data;

MongoDB is one of the NoSQL database management systems. The 'mongod' utility is a daemon process responsible to act as a server. The internal provided database client is 'mongo' utility. Optionally you may use other clients like Compass, Jupyter Notebook (imongo-kernel), and programmatically to interact with MongoDB.

MongoDB is not part of Hadoop Eco System.

Below steps are done as part of MongoDB setup

Run MongoDB server using 'mongod' command

Either keep the window open or may start in the background using 'mongod &'. The letter 'd' in mongod can also be assumed to stand for 'daemon' – a service.

Know that another utility 'mongo' as shown below is actually inbuilt client provided.

```
> show dbs;
admin   0.000GB
config  0.000GB
local   0.000GB
mng     0.000GB
> db;
test
> use demodb;
switched to db demodb
> db.createCollection('democollection');
{ "ok" : 1 }
> db.democollection.save({__id:1,name:"James"});
WriteResult({ "nInserted" : 1 })
> db.democollection.find({});
{ "_id" : ObjectId("5cecdc7c78529695a945ccab"), "__id" : 1, "name" : "James" }
> db.democollection.drop();
true
> db.dropDatabase();
{ "dropped" : "demodb", "ok" : 1 }
> use test;
switched to db test
> show dbs;
admin   0.000GB
config  0.000GB
local   0.000GB
mng     0.000GB
> quit();
Jigar-Pandyas-MacBook:~ JigarPandya$
```

There are many other mongodb clients available in the market. 'Compass' named GUI based client can also be installed and used as required. JuPyter can be configured with mongodb kernel and can be used to interact to MongoDB.

Know that by default MongoDB is installed with no username and password to be used listening on 27017 port localhost.

References :

1. https://www.mongodb.com

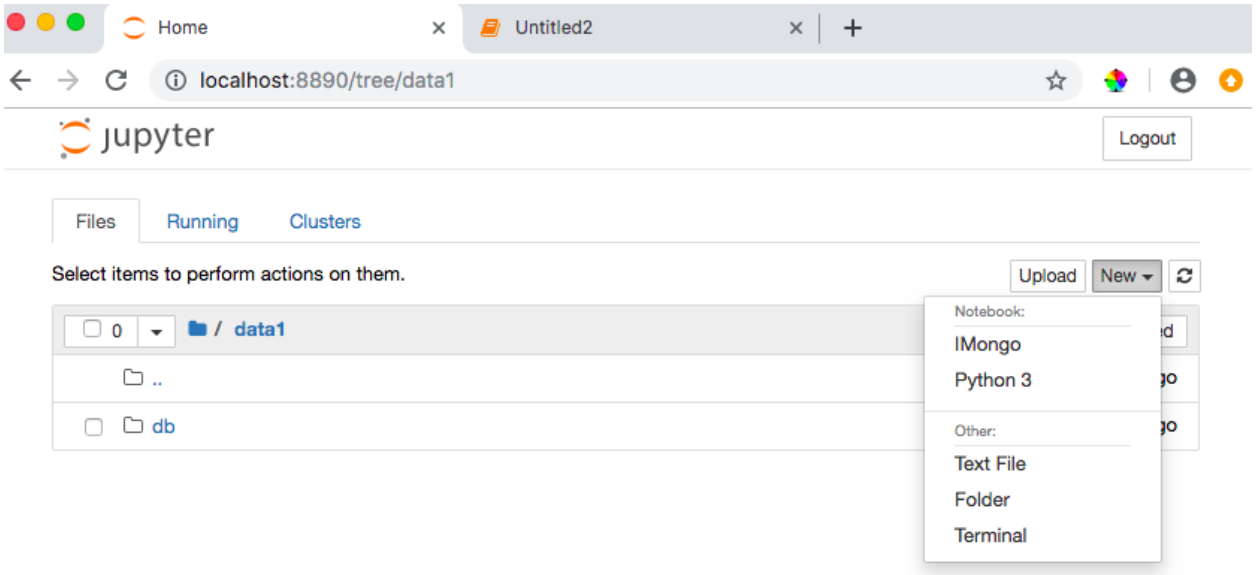
2. https://docs.mongodb.com/manual/

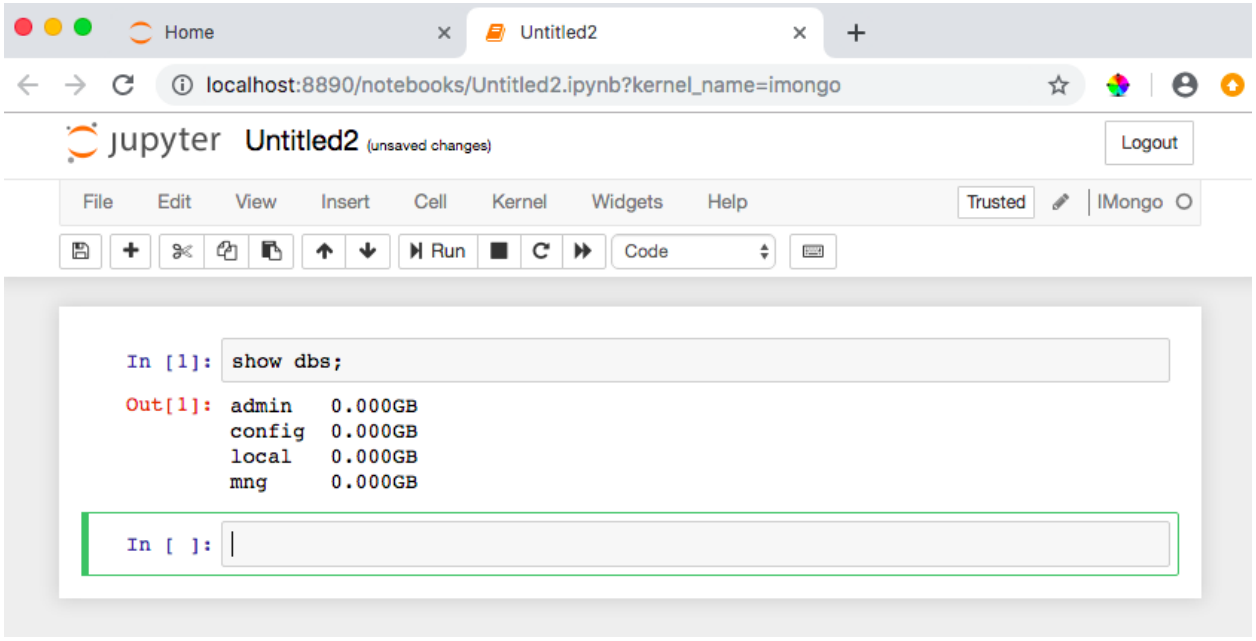
3. https://github.com/gusutabopb/imongo for interacting to MongoDB by JupPyter as a client

4. https://docs.mongodb.com/manual/reference/method/db.collection.aggregate/#db.collection.aggregate

Exercise:

1. Run JuPyter Notebook and using MongoDB kernel, do the same exercise shown in the screenshot by mongo client above.





Note that the folder location to create notebook file is irrelevant to mongoDB. Otherwise /data/db is the location where mongoDB is default configured to store metadata and database.

2. Write the insert method to store the following document in MongoDB

Name : "Stephen More"

Address : {

"City" : "Banglore",

"Street" : "Electronics City",

"Affiliation" : "XYZ Ltd"

}

Hobbies : Chess, Lawn Tennis, Base Ball

3. To practice MapReduce programming in MongoDB.

Step. 3.1: Insert 5 documents as shown below in collection named 'books'.

```
> db.books.find({}).pretty();

{

"_id" : 1,

"Category" : "Machine Learning",

"Bookname" : "Machine Learning for Hackers",

"Author" : "Drew Conway",

"qty" : 25,

"price" : 400,

"rol" : 30,

"pages" : 350

}

{

"_id" : 2,

"Category" : "Business Intelligence",

"Bookname" : "Fundamentals of Business Analytics",

"Author" : "Seema Acharya",

"qty" : 55,
```

"price" : 500,

"rol" : 30,

"pages" : 250

}

{

"_id" : 3,

"Category" : "Analytics",

"Bookname" : "Competing on Analytics",

"Author" : "Thomas Davenport",

"qty" : 8,

"price" : 150,

"rol" : 20,

"pages" : 150

}

{

"_id" : 4,

"Category" : "Visualization",

"Bookname" : "Visualizing Data",

"Author" : "Ben Fry",

"qty" : 12,

"price" : 325,

"rol" : 6,

"pages" : 450

}

{

"_id" : 5,

"Category" : "Web Mining",

"Bookname" : "Learning R",

"Author" : "Richard Cotton",

"qty" : 5,

"price" : 850,

"rol" : 10,

"pages" : 120

}

Step. 3.2: Confirm the presence of above documents in the “books” collection.

Step. 3.3: Write map and reduce functions to split the books into the following two categories:

(a) Big Books

(b) Small Books

Books which have more than 300 pages should be in the big book category. Books wichi have less than 300 pages should be in the small book category.

Step. 3.4: Count the number of books in each category.

Step 3.5: Store the output as follow as documents in a new collection, called “Book_Result”.

Book Category Count of the books

“Big Books" 2

"Small Books" 3

4. To practice import, export and aggregation in MongoDB.

Step 4.1 Pick and public dataset from the site www.kdnuggets.com Convert it into CSV format. Make sure that you have at least two numeric columns.

Step 4.2 Use MongoImport to import data from the CSV format file into MongoDB collection named “MongoDBHandsOn” in test database.

Step 4.3: Identifying a grouping column.

Step 4.4: Compute the sum of the values in the first numeric column.

Step 4.5: Compute the average of the values in the second numeric column.

5. Exercise Python to MongoDB Connectivity using JuPyter Notebook.

Solution:

db.createCollection('objective4');

db.objective4.save({name:"Stephen More",Address:{City:"Banglore",Street:"Electronics City",Affiliation:"XYZ Ltd"},Hobbies:['Chess','Lawn Tennis','Base Ball']})

db.books.save({_id:1, Category:"Machine Learning",Bookname:"Machine Learning for Hackers", Author:"Drew Conway", qty:25, price:400, rol:30, pages:350});

db.books.save({_id:2, Category:"Business Intelligence",Bookname:"Fundamentals of Business Analytics", Author:"Seema Acharya", qty:55, price:500, rol:30, pages:250});

db.books.save({_id:3, Category:"Analytics",Bookname:"Competing on Analytics", Author:"Thomas Davenport", qty:8, price:150, rol:20, pages:150});

db.books.save({_id:4, Category:"Visualization",Bookname:"Visualizing Data", Author:"Ben Fry", qty:12, price:325, rol:6, pages:450});

db.books.save({_id:5, Category:"Web Mining",Bookname:"Learning R", Author:"Richard Cotton", qty:5, price:850, rol:10, pages:120});

```
var map = function(){ if ( this.pages > 300) emit ('Big Books',1); else emit ('Small Books',1); }

var reduce = function(key, values){ return Array.sum(values);}

db.books.mapReduce(map, reduce, {out:"Book_Result",query:{}});

> db.Book_Result.find({});
{ "_id" : "Big Books", "value" : 2 }
{ "_id" : "Small Books", "value" : 3 }
>
```

mongoimport --db=mng --collection=SampleJSON --type=csv --headerline --file="/Users/JigarPandya/Desktop/BDT_2019/Big Data and Analytics/Data sets/Dataset to practice MongoDB_Import/sample.txt"

mongoimport --db=mng --collection=student_master --type=csv --headerline --file="/Users/JigarPandya/Desktop/BDT_2019/Fundamentals of Business Analytics/Data Sheet in Chapter 9/Source Data.csv"

Create fields file

/Users/JigarPandya/Desktop/BDT_2019/export_book/fields.txt

_id

Category

Bookname

Author

qty

price

rol

pages

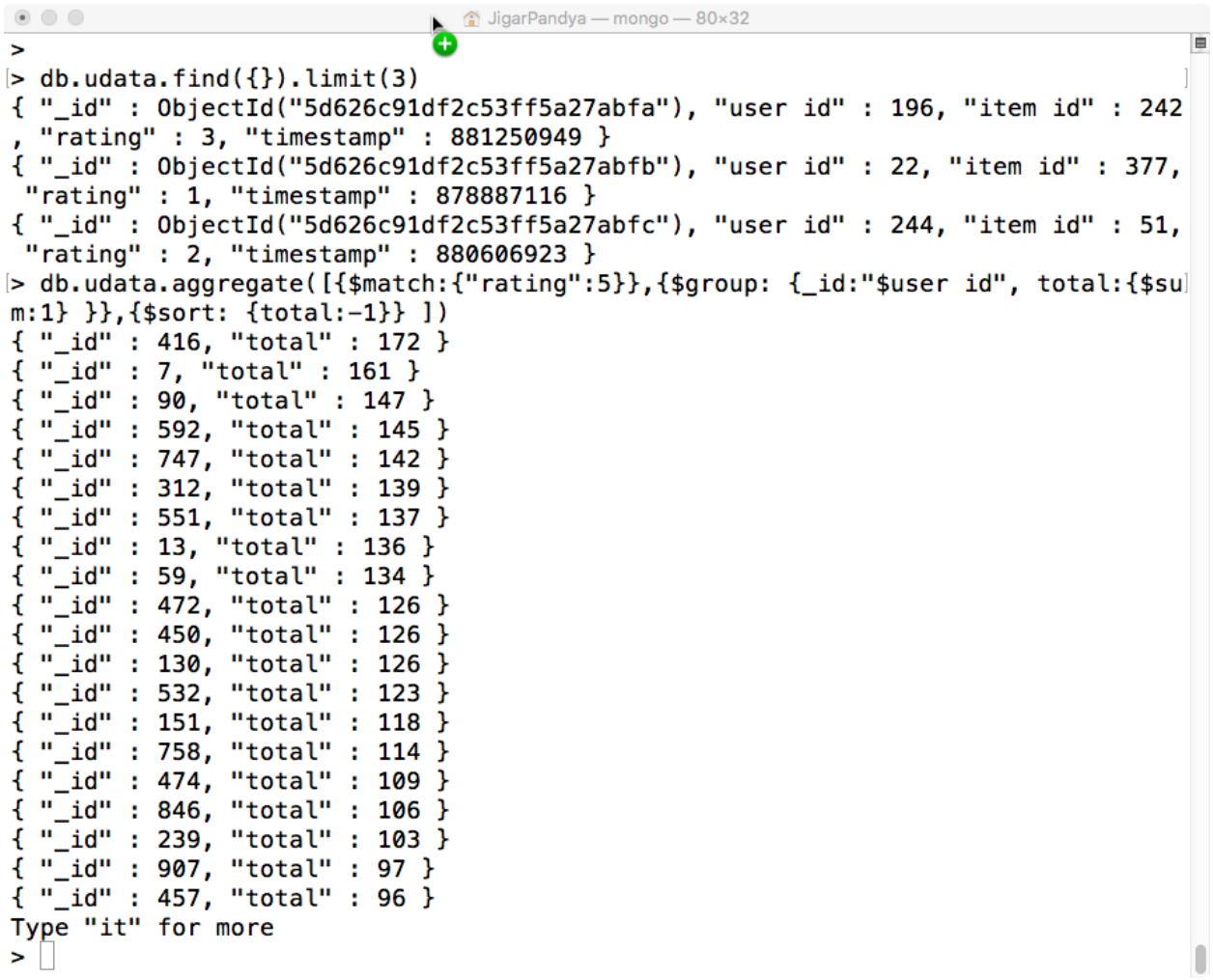
mongoexport --db=mng --collection=books --type=csv --fieldFile="/Users/JigarPandya/Desktop/BDT_2019/export_book/fields.txt" --out="/Users/JigarPandya/Desktop/BDT_2019/export_book/book.csv"

AGGREGATION EXAMPLE

mongoimport --db=mng --collection=udata --type=tsv --fieldFile="/Users/JigarPandya/Desktop/BDA_2019/mongoDBAggregateDemo/udatafields.txt" --file="/Users/JigarPandya/Desktop/BDA_2019/mongoDBAggregateDemo/u.data"

Find the total number of feedback given by different users with rating 5. Display in sorted order highest being on the top.

```
db.udata.find({}).limit(3)

db.udata.aggregate([{$match:{"rating":5}},{$group: {_id:"$user
id", total:{$sum:1} }},{$sort: {total:-1}} ])
```



Python JuPyter to connect to MongoDBHandsOn

coding: utf-8

```
# In[37]:

from pymongo import MongoClient

from pprint import pprint

connection = MongoClient("mongodb://localhost:27017/admin")

db = connection.test

db.inventory.insert_one(

{"_id":1,

"item": "canvas",

"qty": 100,

"tags": ["cotton"],

"size": {"h": 28, "w": 35.5, "uom": "cm"}})

db.inventory.insert_one(

{"_id":2,

"item": "painting",

"qty": 50,

"tags": ["nature"],

"size": {"h": 100, "w": 55, "uom": "cm"}})

cursor = db.inventory.find({})
```

```
for inventory in cursor:

pprint(inventory)

# In[38]:



# Subdocument key order matters in a few of these examples so we have

# to use bson.son.SON instead of a Python dict.

from bson.son import SON

db.inventory.insert_many([

{"_id":3,

"item": "journal",

"qty": 25,

"size": SON([("h", 14), ("w", 21), ("uom", "cm")]),

"status": "A"},

{"_id":4,

"item": "notebook",

"qty": 50,

"size": SON([("h", 8.5), ("w", 11), ("uom", "in")]),

"status": "A"},

{"_id":5,
```

"item": "paper",

"qty": 100,

"size": SON([("h", 8.5), ("w", 11), ("uom", "in")]),

"status": "D"},

{"_id":6,

"item": "planner",

"qty": 75,

"size": SON([("h", 22.85), ("w", 30), ("uom", "cm")]),

"status": "D"},

{"_id":7,

"item": "postcard",

"qty": 45,

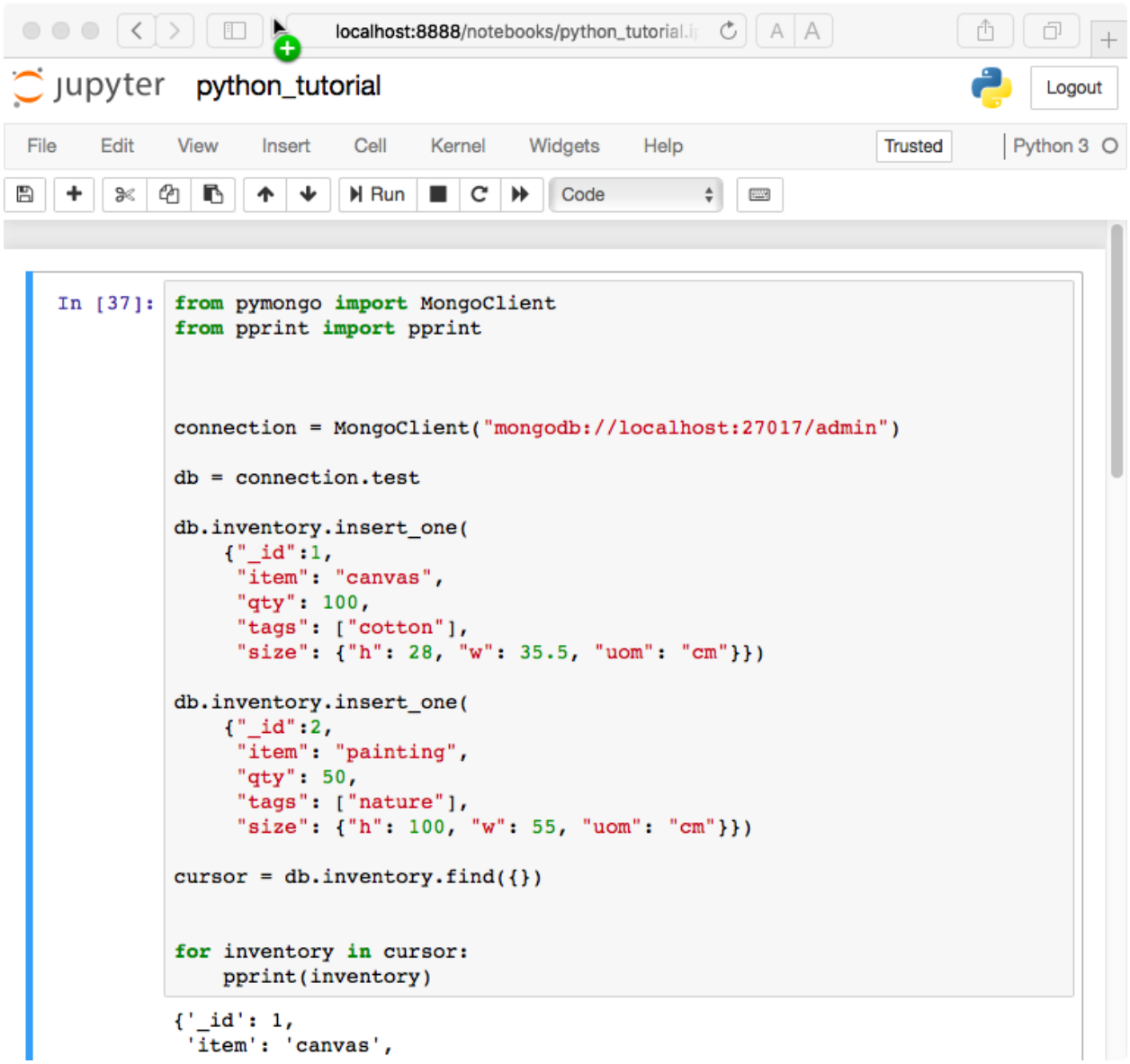
"size": SON([("h", 10), ("w", 15.25), ("uom", "cm")]),

"status": "A"}])

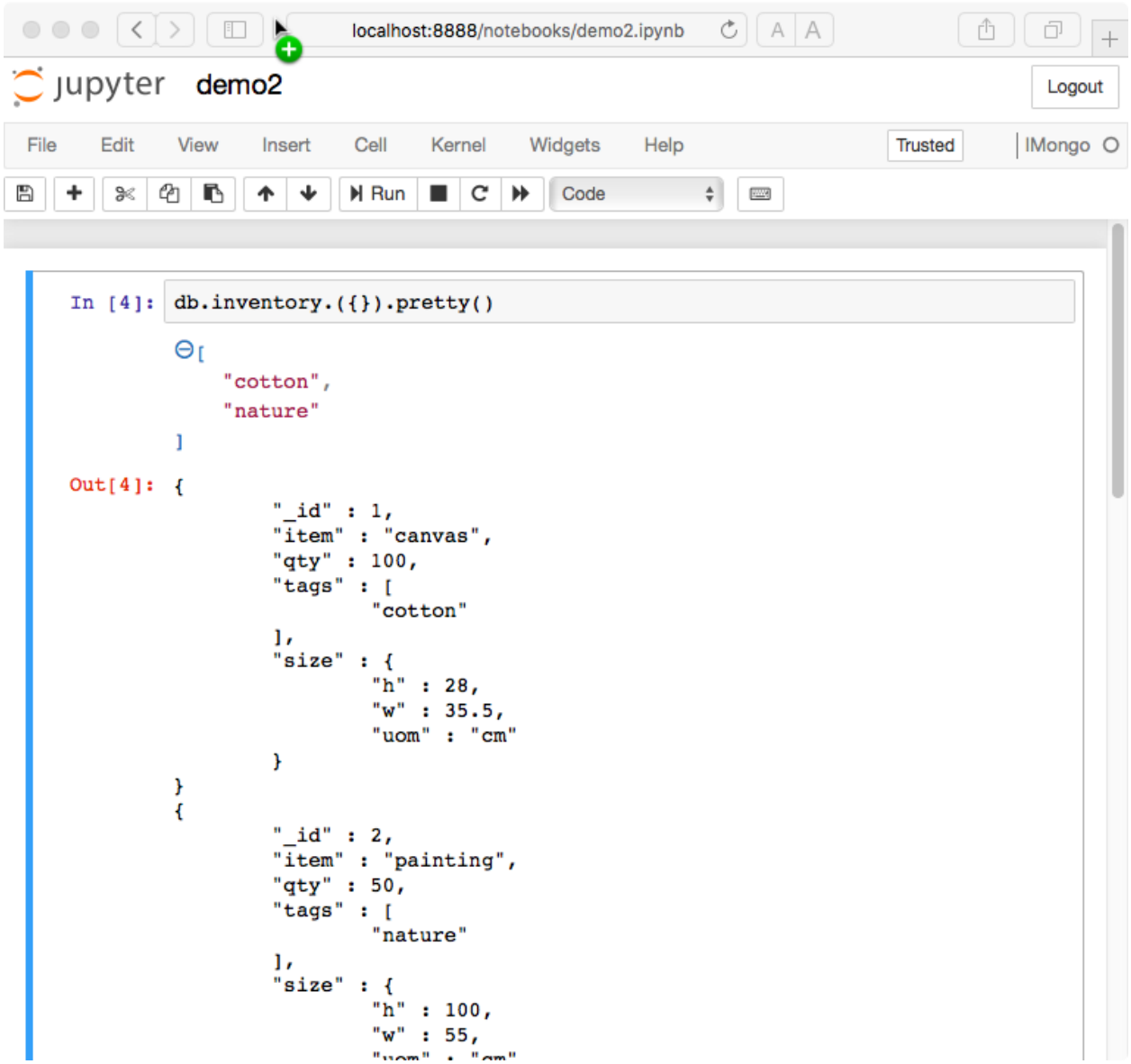
In[44]:

```
cursor = db.inventory.find({"status": "D"})

# In[46]:

cursor = db.inventory.find(

{"size": SON([("h", 14), ("w", 21), ("uom", "cm")])})

# In[48]:

cursor = db.inventory.find({"size.uom": "in"})

# In[49]:

from pprint import pprint

for inventory in cursor:

pprint(inventory)
```

Python Kernel



iMongo Kernel



Documented By:

Jigar M. Pandya

https://www.linkedin.com/in/jigar-pandya

Document Last updated: Aug 5, 2020.