

Prueba BDD Multidimensionales

Estudiante: Sebastián Morales

Primero lo que se va a realizar es crear las bases de datos respectivas (bdd1 y bdd2) en CouchDB:

Name	Size	# of Docs	Partitioned	Actions
_replicator	11.8 KB	7	No	[Icons]
bdd1	0 bytes	0	No	[Icons]
bdd2	0 bytes	0	No	[Icons]

Ahora procedemos a crear réplicas de manera bidireccional con las bases creadas anteriormente:

Name	Source	Destination	Status	Actions
http://localhost:5984/bdd1	http://localhost:5984/bdd2	Aug 18th, 5:11 pm	Continuous Running	[Icons]
http://localhost:5984/bdd2	http://localhost:5984/bdd1	Aug 18th, 5:12 pm	Continuous Running	[Icons]

Ahora se va a recopilar los tweets geolocalizados en New York, para ello se va a usar el archivo tweetsNY.py

Para obtener las coordenadas de New York se utiliza la siguiente página web: <https://boundingbox.klokantech.com/> Una vez aquí se procede a buscar new york y se deben copiar los resultados que están en la parte inferior izquierda de la página web seleccionando la opción “CSV” en el drop down:

Find a place with OpenStreetMap ...

Copy & Paste CSV: -79.76,40.48,-71.8,45.02

Estas mismas coordenadas son las que deben estar colocadas en el script que se va a utilizar, en este caso haremos que se guarde en la bdd1 y se verá replicada en la bdd2:

```
Welcome tweetsNY.py X
tweetsNY.py > ...
1 import couchdb
2 from tweepy import Stream
3 from tweepy import OAuthHandler
4 from tweepy.streaming import StreamListener
5 import json
6 ###API #####
7 ckey = "6Zyv4XxVypDqHDpFoHwSTrMzX"
8 csecret = "3J5TpltHtmEZGEw8RhRLABc3KQ2Quhjj2SVVykfw5zs02fjtpC"
9 atoken = "153168970-C8H0rPCjztDmLQMrjtgOYSPiZjLMyegrtrAZQQrq"
10 asecret = "WxWpMOMlghN1tVYZRFugRWTeFM1SShLWVI4lL4oPWTAlO"
11 #####
12
13 class listener(StreamListener):
14
15     def on_data(self, data):
16         dictTweet = json.loads(data)
17         try:
18
19             dictTweet["_id"] = str(dictTweet['id'])
20             doc = db.save(dictTweet)
21             print ("SAVED" + str(doc) + "=>" + str(data))
22         except:
23             print ("Already exists")
24             pass
25         return True
26
27     def on_error(self, status):
28         print (status)
29
30 auth = OAuthHandler(ckey, csecret)
31 auth.set_access_token(atoken, asecret)
32 twitterStream = Stream(auth, listener())
33
34 '''=====couchdb====='''
35 server = couchdb.Server('http://sebas:123456@localhost:5984/')
36 try:
37     db = server.create('bdd1')
38 except:
39     db = server['bdd1']
40
41 '''=====LOCATIONS====='''
42 twitterStream.filter(locations=[-79.76,40.48,-71.8,45.02])
```

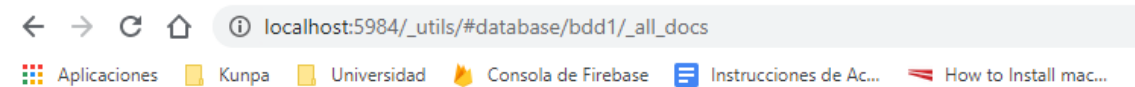
Ejecutamos el script

```
c:\Users\Sebastian\OneDrive\Escritorio\Quinto Semestre\Bases De Datos Multidimensionales\segundo bimestre\Pruebapypython tweetsNY.py
SAVED('1295849494895099909', '1-388889dd5808626682f805b4a19a55c3')->{"created_at":"Tue Aug 18 22:26:09 +0000 2020", "id":12958494948
95099909, "id_str":"1295849494895099909", "text":"Aye man love yah girl bro don\u0020t listen to the haters", "source":"\u003ca href=
'http://twitter.com/download/iphone' rel='nofollow'\u003eTwitter for iPhone\u003c/a\u003e", "truncated":false, "in_reply_to_s
tatus_id":null, "in_reply_to_status_id_str":null, "in_reply_to_user_id":null, "in_reply_to_user_id_str":null, "in_reply_to_screen_name":
null, "user":{"id":32801440, "id_str":"32801440", "name":"I Live One Hell Of A Life", "screen_name":"SirEzioAuditore", "location":"Bron
```

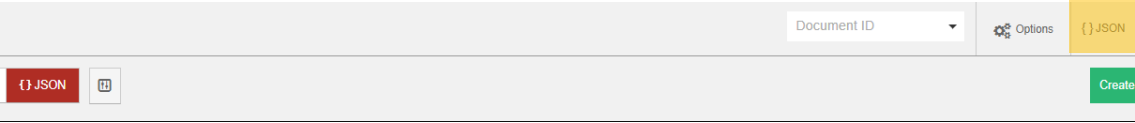
Se verifica que los datos se estén cargando y la réplica cumpla su función:

bdd1	0.5 MB	218
bdd2	0.5 MB	218

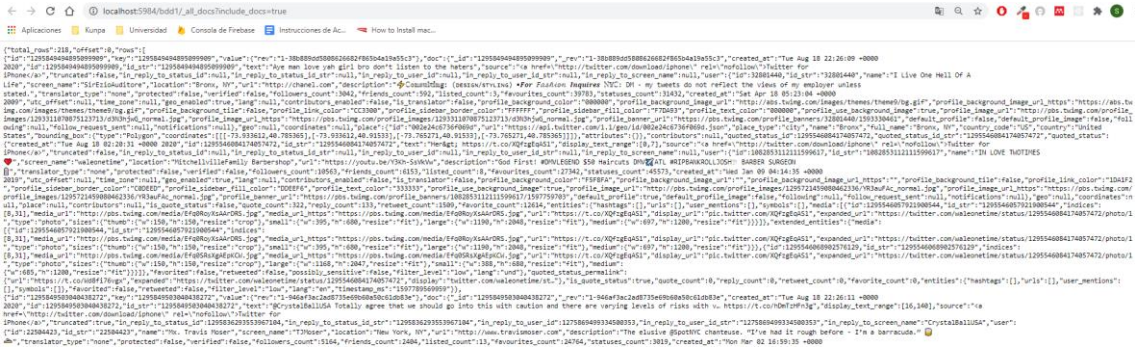
Ahora procederemos a crear un archivo llamado bdd1.json que contendrá los datos en formato .json de una de las bases de datos anteriormente, para ello en CouchDB se debe ingresar a una de las dos bases creadas, seleccionar la opción “{} JSON”



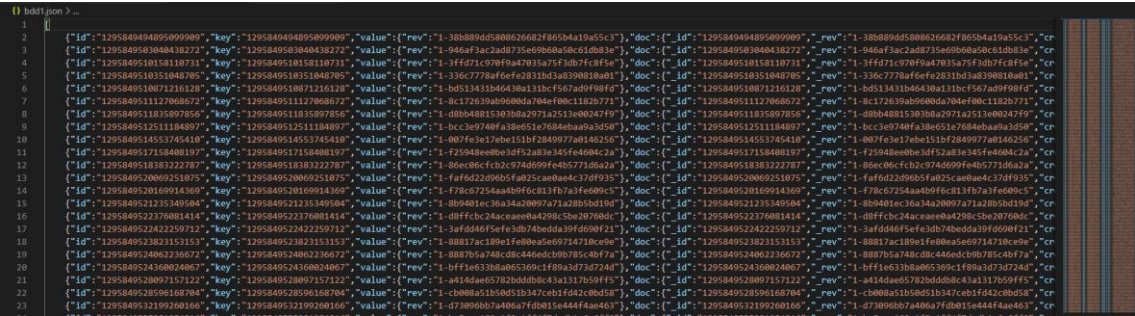
Y posteriormente a ello se selecciona la opción “{} JSON” que esta subrayada a continuación:



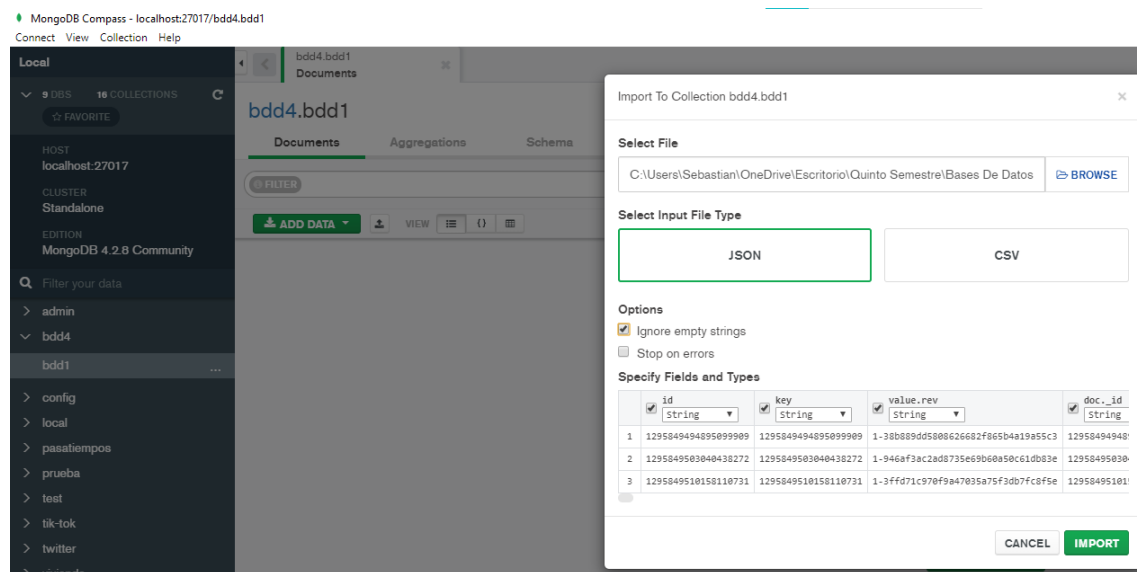
Aparecerá la siguiente página que contiene todos los datos en formato .json



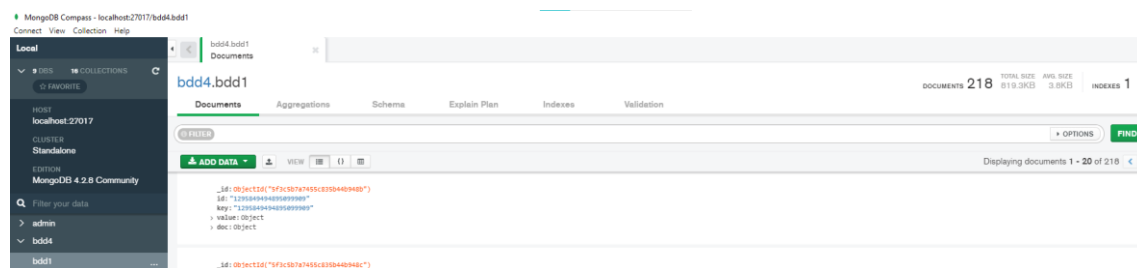
Copiamos todos los datos y lo pegamos en nuestro archivo creado dejando la primera línea vacía



Ahora se procede a cargar a la base de datos de mongoDB denominada bdd4



Comprobamos que se hayan cargado los datos:



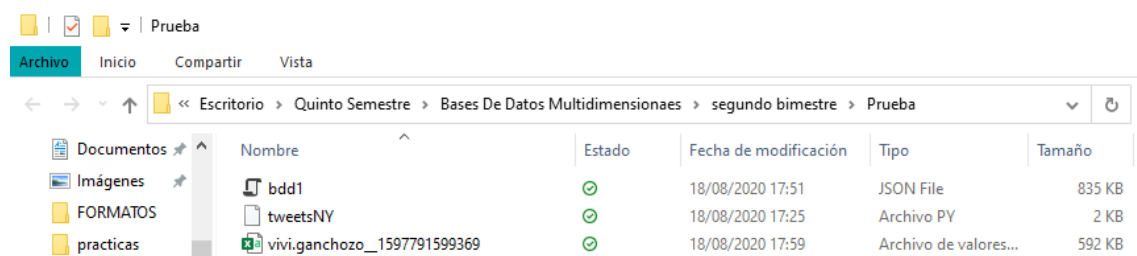
Post de TikTok:

Se utiliza el comando “tiktok-scraper user USERNAME -n 300 -t csv” donde se debe colocar el nombre del usuario de TikTok en vez de “USERNAME”.

El comando que se ejecutará es el siguiente:

```
C:\Users\Sebastian\OneDrive\Escritorio\Quinto Semestre\Bases De Datos Multidimensionales\segundo bimestre\Prueba>tiktok-scraper user vivi.ganchozo -n 300 -t csv
CSV path: C:\Users\Sebastian\OneDrive\Escritorio\Quinto Semestre\Bases De Datos Multidimensionales\segundo bimestre\Prueba\vivi.ganchozo_1597791599369.csv
```

Se debe comprobar que el archivo se ha bajado en formato .csv:



Y por último lo subimos a la base de datos de mongoDB:

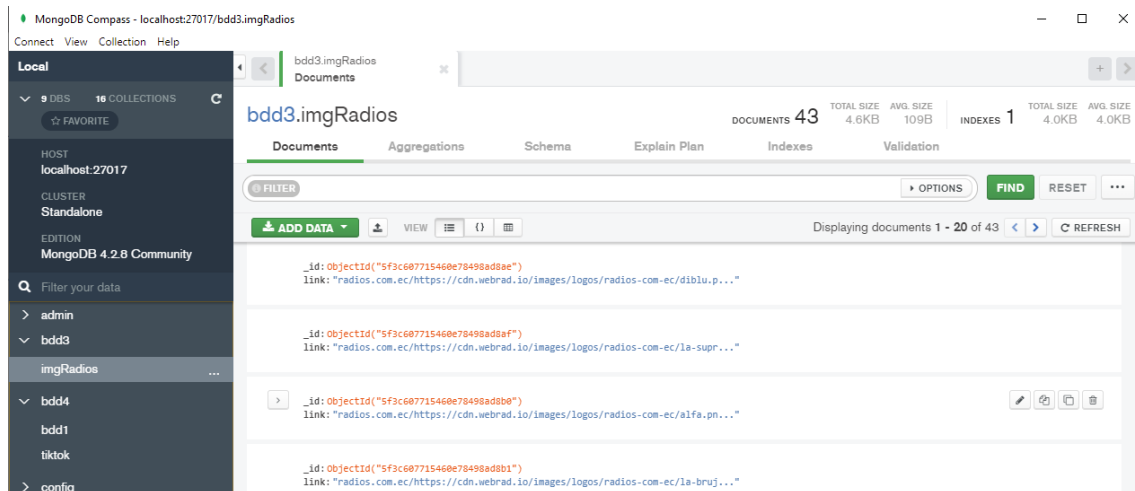
Para ello se utilizará el script imgRadios.py

```
imgRadios.py > ...
1  import requests
2  from bs4 import BeautifulSoup
3  # Import MongoClient from pymongo so we can connect to the database
4  from pymongo import MongoClient
5
6
7  if __name__ == '__main__':
8      # Instantiate a client to our MongoDB instance
9      db_client = MongoClient('mongodb://localhost:27017')
10     bdd3 = db_client.bdd3
11     imgRadios = bdd3.posts
12
13
14     response=requests.get("https://radios.com.ec/")
15     soup=BeautifulSoup(response.content, "lxml")
16
17     post_titles=soup.find_all("img", class_="cover")
18
19     extracted=[]
20
21     for post_title in post_titles:
22         extracted.append({
23             'link': "radios.com.ec/" + post_title['src']
24         })
25
26     for post in extracted:
27         if db_client.bdd3.imgRadios.find_one({'link': post['link']}) is None:
28             # Let's print it out to verify that we added the new post
29             print("Found a new listing at the following url: ", post['link'])
30             db_client.bdd3.imgRadios.insert(post)
31
```

Ejecutamos el script

```
C:\Users\Sebastian\OneDrive\Escritorio\Quinto Semestre\Bases De Datos Multidimensionales\segundo bimestre
\Prueba>python imgRadios.py
Found a new listing at the following url: radios.com.ec/https://cdn.webrad.io/images/logos/radios-com-e
c/diblu.png
imgRadios.py:30: DeprecationWarning: insert is deprecated. Use insert_one or insert_many instead.
  db_client.bdd4.imgRadios.insert(post)
Found a new listing at the following url: radios.com.ec/https://cdn.webrad.io/images/logos/radios-com-e
c/la-suprema.png
Found a new listing at the following url: radios.com.ec/https://cdn.webrad.io/images/logos/radios-com-e
c/alfa.png
Found a new listing at the following url: radios.com.ec/https://cdn.webrad.io/images/logos/radios-com-e
c/la-bruja.png
Found a new listing at the following url: radios.com.ec/https://cdn.webrad.io/images/logos/radios-com-e
c/hcjb.png
Found a new listing at the following url: radios.com.ec/https://cdn.webrad.io/images/logos/radios-com-e
c/la-voz-cuenca.png
Found a new listing at the following url: radios.com.ec/https://cdn.webrad.io/images/logos/radios-com-e
c/maria.png
Found a new listing at the following url: radios.com.ec/https://cdn.webrad.io/images/logos/radios-com-e
c/tropicalida.png
```

Comprobamos que se hayan cargado a la base de datos:



Tweets relacionados a Donal Trump:

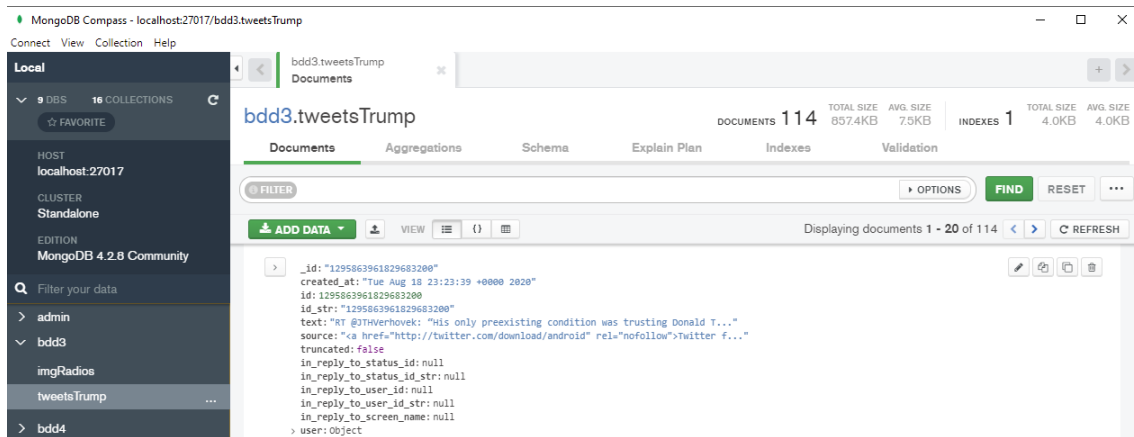
Debemos ejecutar el script llamado Trump.py

```
trump.py > ...
1 import pymongo
2 import pprint
3 from tweepy import Stream
4 from tweepy import OAuthHandler
5 from tweepy.streaming import StreamListener
6 import json
7
8 ckey = "6Zyv4XxVypDqHDPFoHwSTrMzX"
9 csecret = "3J5Tp1tHtmEZGEw8RhRLABc3KQ2Quhjj2SVVykfw5zs02fjtpC"
10 atoken = "153168970-C8H0rPCjztDmLQMrjtg0YSPIzjLMyegnrtrAZQQrq"
11 asecret = "WxWpMOMlghN1tVYZRFugRWtefM1SShLwVI4lL4oPWTA10"
12
13 class listener(StreamListener):
14
15     def on_data(self, data):
16         dictTweet = json.loads(data)
17         try:
18             dictTweet["_id"] = str(dictTweet['id'])
19             doc = mycol.save(dictTweet)
20             print ("SAVED" + str(doc) + ">" + str(data))
21         except:
22             print ("Already exists")
23             pass
24         return True
25
26     def on_error(self, status):
27         print (status)
28
29 auth = OAuthHandler(ckey, csecret)
30 auth.set_access_token(atoken, asecret)
31 twitterStream = Stream(auth, listener())
32
33 '''=====mongodb'====='''
34 myclient = pymongo.MongoClient("mongodb://localhost:27017")
35 try:
36     mydb=myclient["bdd3"]
37     mycol=mydb["tweetsTrump"]
38 except:
39     mydb=myclient["bdd3"]
40     mycol=mydb["tweetsTrump"]
41
42
43 twitterStream.filter(track=['Donald Trump'])
```

Se ejecuta el Script

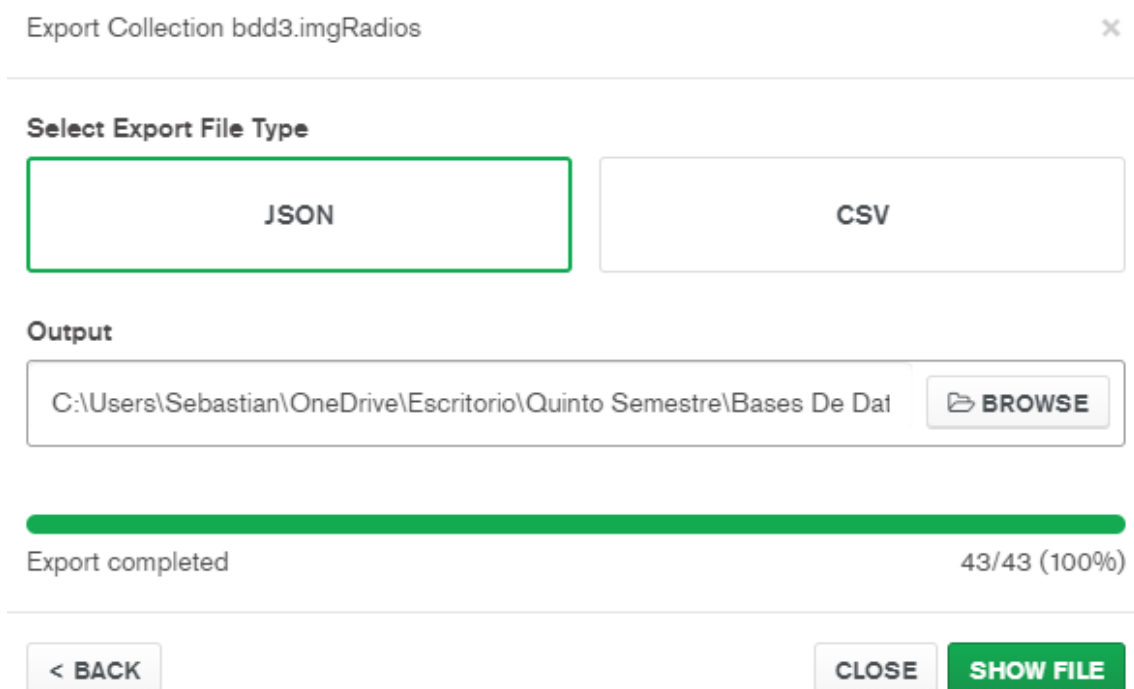
```
C:\Users\Sebastian\OneDrive\Escritorio\Quinto Semestre\Bases De Datos Multidimensionales\segundo bimestre\Prueba>python trump.py
trump.py:19: DeprecationWarning: save is deprecated. Use insert_one or replace_one instead
  doc = mycol.save(dicttweet)
SAVE[{"_id":1295863961829683200,"created_at":"Tue Aug 18 23:23:39 +0000 2020","id":"1295863961829683200","id_str":"1295863961829683200","text":"RT @JTHVerhovek: \u201cHis only preexisting condition was trusting Donald Trump, and for that, he paid with his life.\u201d\n\nA look back at what was\u201d","source":"\u003csrc href=\u201chttp://v/twitter.com/download/android\u201d rel=\u201cnofollow\u201d\u003c/a\u003e","truncated":false,"in_reply_to_status_id":null,"in_reply_to_status_id_str":null,"in_reply_to_user_id":null,"in_reply_to_user_id_str":null,"in_reply_to_screen_name":null,"user":{"id":213103084,"id_str":"213103084","name":"mary ann white","screen_name":"gapuppie","location":"Dacula, GA","url":null,"description":"Retired from crime scene unit. I love all animals, but especially dogs. I think Trump is a self centered imbecile. He has no soul.", "translator_type": "none", "protected": false, "verified": false, "followers_count": 2275, "friends_count": 4405, "listed_count": 3, "favourites_count": 29974, "statuses_count": 38556, "created_at": "Sun Nov 07 23:58:02 +0000 2010", "utc_offset": null, "time_zone": null, "geo_enabled": true, "lang": null, "contributors_enabled": false, "is_translator": false, "profile_background_color": "f0f0f0", "profile_background_image_url": "http://va
```

Se verifica que se haya cargado la información a la base de datos bdd3 en mongoDB:



Ahora se debe importar los archivos de la base de datos bdd3 a la base de datos bdd4:

Se exportan las colecciones a una carpeta local:



Export Collection bdd3.tweetsTrump



Select Export File Type

JSON

CSV

Output

C:\Users\Sebastian\OneDrive\Escritorio\Quinto Semestre\Bases De Da

BROWSE

Export completed

114/114 (100%)

< BACK

CLOSE

SHOW FILE

Archivo Inicio Compartir Vista

← → ↶ ↷ ⌂ << Escritorio > Quinto Semestre > Bases De Datos Multidimensionaes > segundo bimestre > Prueba

	Nombre	Estado	Fecha de modificación	Tipo	Tamaño
Imágenes					
Bases De Datos	trump		18/08/2020 18:35	JSON File	658 KB
FORMATOS	imgRadios		18/08/2020 18:33	JSON File	7 KB

Ahora importamos los archivos obtenidos a la base de datos bdd4:

bdd4.imgRadios

Documents Aggregations Schema

FILTER

ADD DATA VIEW

_id: ObjectId("5f3c607715460e78498adb0e")
link: "radios.com.ec/https://cdn.webrad.io/images/logos/radios-com-ec/"

_id: ObjectId("5f3c607715460e78498adb0f")
link: "radios.com.ec/https://cdn.webrad.io/images/logos/radios-com-ec/"

_id: ObjectId("5f3c607715460e78498adb0e")
link: "radios.com.ec/https://cdn.webrad.io/images/logos/radios-com-ec/"

_id: ObjectId("5f3c607715460e78498adb01")
link: "radios.com.ec/https://cdn.webrad.io/images/logos/radios-com-ec/"

_id: ObjectId("5f3c607715460e78498adb02")
link: "radios.com.ec/https://cdn.webrad.io/images/logos/radios-com-ec/"

_id: ObjectId("5f3c607715460e78498adb03")
link: "radios.com.ec/https://cdn.webrad.io/images/logos/radios-com-ec/"

_id: ObjectId("5f3c607715460e78498adb04")
link: "radios.com.ec/https://cdn.webrad.io/images/logos/radios-com-ec/"

_id: ObjectId("5f3c607715460e78498adb05")
link: "radios.com.ec/https://cdn.webrad.io/images/logos/radios-com-ec/"

Import To Collection bdd4.imgRadios

Select File

C:\Users\Sebastian\OneDrive\Escritorio\Quinto Semestre\Bases De Datos BROWSE

Select Input File Type

JSON CSV

Options

☒ Ignore empty strings

☐ Stop on errors

Specify Fields and Types

	id	link
1	ObjectId	String
2	ObjectId	String
3	ObjectId	String
4	ObjectId	String
5	ObjectId	String
6	ObjectId	String
7	ObjectId	String
8	ObjectId	String
9	ObjectId	String
10	ObjectId	String

Import completed 43 (100%)

DONE

DOCUMENTS 43

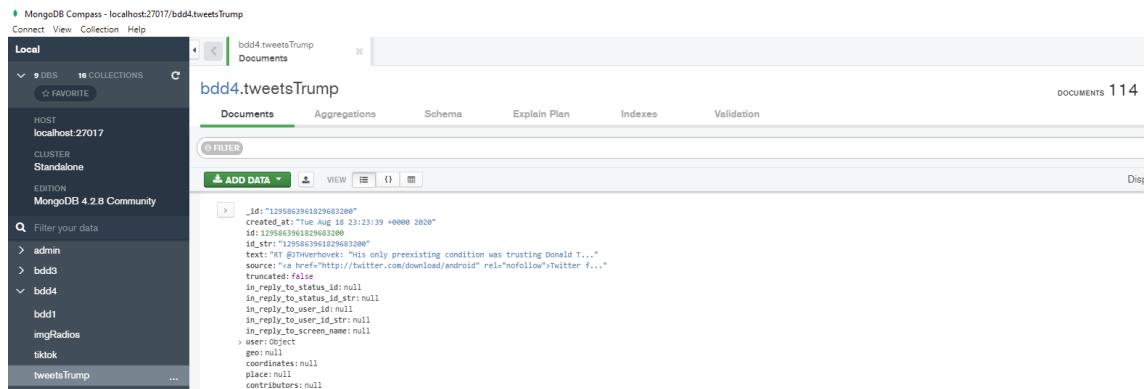
Insert to Collection bdd4.tweetsTrump

VIEW  

```
1 [{
2   "_id": "1295863961829683200",
3   "created_at": "Tue Aug 18 23:23:39 +0000 2020",
4   "id": "1295863961829683200",
5   "id_str": "1295863961829683200",
6   "text": "RT @JTHVerhovek: \"His only preexisting condition was trusting Donald Tr",
7   "source": "<a href='\"http://twitter.com/download/android\"' rel='\"nofollow\">Twit",
8   "truncated": false,
9   "in_reply_to_status_id": null,
10  "in_reply_to_status_id_str": null,
11  "in_reply_to_user_id": null,
12  "in_reply_to_user_id_str": null,
13  "in_reply_to_screen_name": null,
14  "user": {
15    "id": "213103084",
16    "id_str": "213103084",
17    "name": "mary ann white",
18    "screen_name": "gapuppie",
19    "location": "Dacula, GA",
20    "url": null,
21    "description": "Retired from crime scene unit. I love all animals, but especia",
22    "translator_type": "none",
```

CANCEL

INSERT

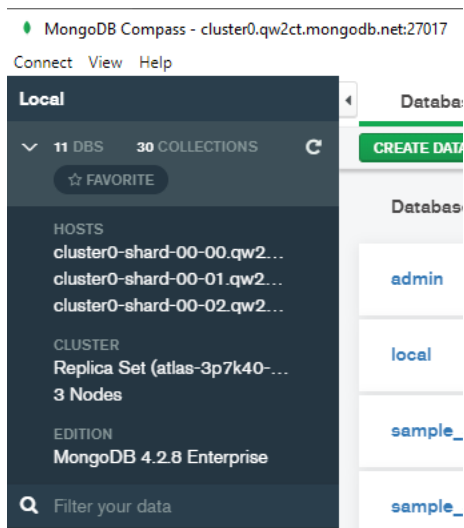


The screenshot shows the MongoDB Compass interface. On the left, the 'Local' sidebar lists the database 'bdd4' and the collection 'tweetsTrump'. The main panel displays the 'Documents' tab for 'bdd4.tweetsTrump', showing a list of documents. The first document is expanded, showing its JSON structure, which matches the data shown in the previous code block. The interface includes a search bar, a filter button, and a 'Display' button.

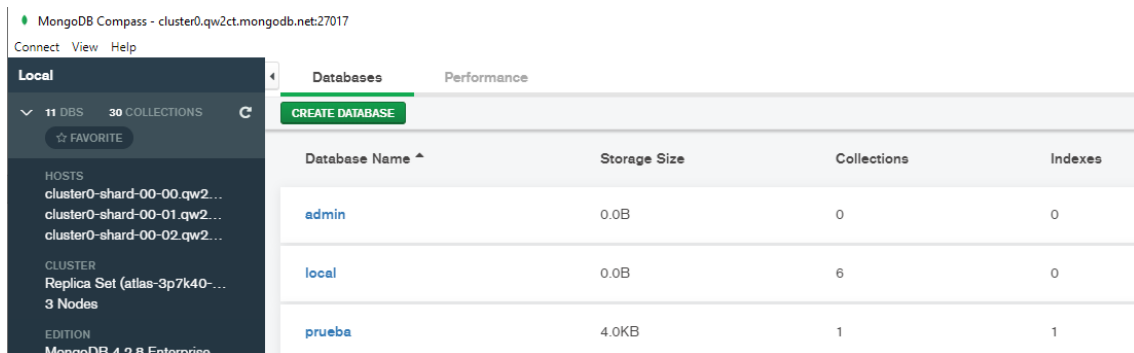
Ahora que todo esta en la base de datos llamada bdd4 localmente se procederá a exportar todo lo que está contenido en la base de datos bdd4 a la base de datos de datos de la nube mongoDB Atlas.

Cadena de conexión: mongodb+srv://esfot2020:esfot2020@cluster0.qw2ct.mongodb.net/test

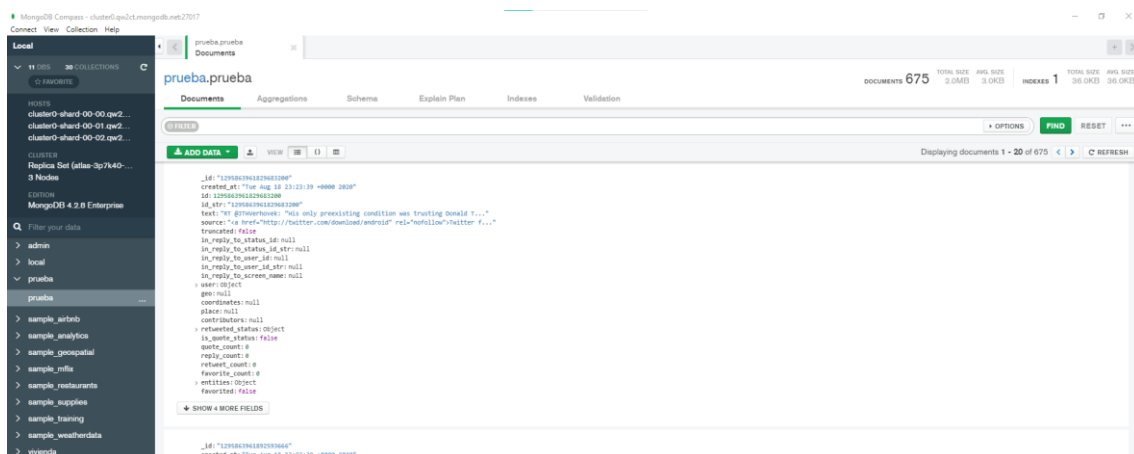
Se comprueba el acceso a la misma:



Ahora procedemos a crear una base de datos llamada “prueba” con una colección “prueba” donde se cargarán todos los archivos:



Y en esta se cargan todos los archivos .json obtenidos:



Como resultado se obtuvo una colección de 675 documentos.

Por ultimo exportamos los archivos resultantes tanto en formato resultados.csv como en resultados.json :

Export Collection prueba.prueba

Select Export File Type

JSON

CSV

Output

C:\Users\Sebastian\OneDrive\Escritorio\Quinto Semestre\Bases De Dat

BROWSE

Export completed

675/675 (100%)

< BACK

CLOSE

SHOW FILE

Export Collection prueba.prueba

Select Export File Type

JSON

CSV

Output

C:\Users\Sebastian\OneDrive\Escritorio\Quinto Semestre\Bases De Dat

BROWSE

Export completed

675/675 (100%)

< BACK

CLOSE

SHOW FILE

La carpeta resultante es la siguiente:

Imágenes

Bases De Datos

FORMATOS

Prueba

segundo bimestre

Creative Cloud Fil

Dropbox

OneDrive

Este equipo

Descargas

Nombre

Estado

Fecha de modificación

Tipo

Tamaño

Prueba

resultados

resultados

tiktok

trump

imgRadios

trump

imgRadios

vivi.ganchozo__1597791599369

bdd1

tweetsNY

18/08/2020 19:08

18/08/2020 19:04

18/08/2020 19:04

18/08/2020 18:54

18/08/2020 18:35

18/08/2020 18:33

18/08/2020 18:21

18/08/2020 18:12

18/08/2020 17:59

18/08/2020 17:51

18/08/2020 17:25

Microsoft Edge P...

Archivo de valores...

JSON File

JSON File

JSON File

Archivo PY

Archivo PY

Archivo de valores...

JSON File

Archivo PY

1.573 KB

509 KB

1.065 KB

803 KB

658 KB

7 KB

2 KB

1 KB

592 KB

835 KB

2 KB