MSc in Data Science

Big Data Management

Project 2

Zoi Papakonstantinou
AM: dit2118dsc
Email: dit2118dsc@go.uop.gr

Nikolaos-Marios Tsarouchas
AM: dit2120dsc
Email: dit2120dsc@go.uop.gr

Contents

Abstract

The purpose of this project is to study data related to the most popular YouTube videos during the period Nov'17-Mar'18. The data are reported in ten regions: U.S.A(US), Great Britain (GB), Germany (DE), Canada (CA), France (FR), Russia (RU), Mexico (MX), South Korea (KR), Japan (JP), and India (IN). Data processing was implemented into a NoSQL database, MongoDB. In the first chapter, the necessary conversions and transformations of the data needed for the purposes of this project is performed. Next, in chapter two we showed the number of views, "likes" and "dislikes" for each video for the channel "Saturday Night Live". In the third chapter, we created a new metric that counted the number of tags for each video and visualized the number of views and tags in a scatter plot. In the fourth chapter, we compared the number of videos displayed on each tag between the two areas of Grant Britain (GB) and the USA. The results were represented by a bar chart. In the fifth chapter, we calculated the average number of views, "likes" and "dislikes" for videos that have disabled comments. We visualized and analyzed the results. Finally, we grouped the videos by publication date from December 5, 2017 to March 5, 2018. We were asked to visualize the data with a scatter plot, however since the scatter plot did not indicate a clear correlation, we also visualized the data with a bar chart.

All the scripts are available at the git repo.

Glossary

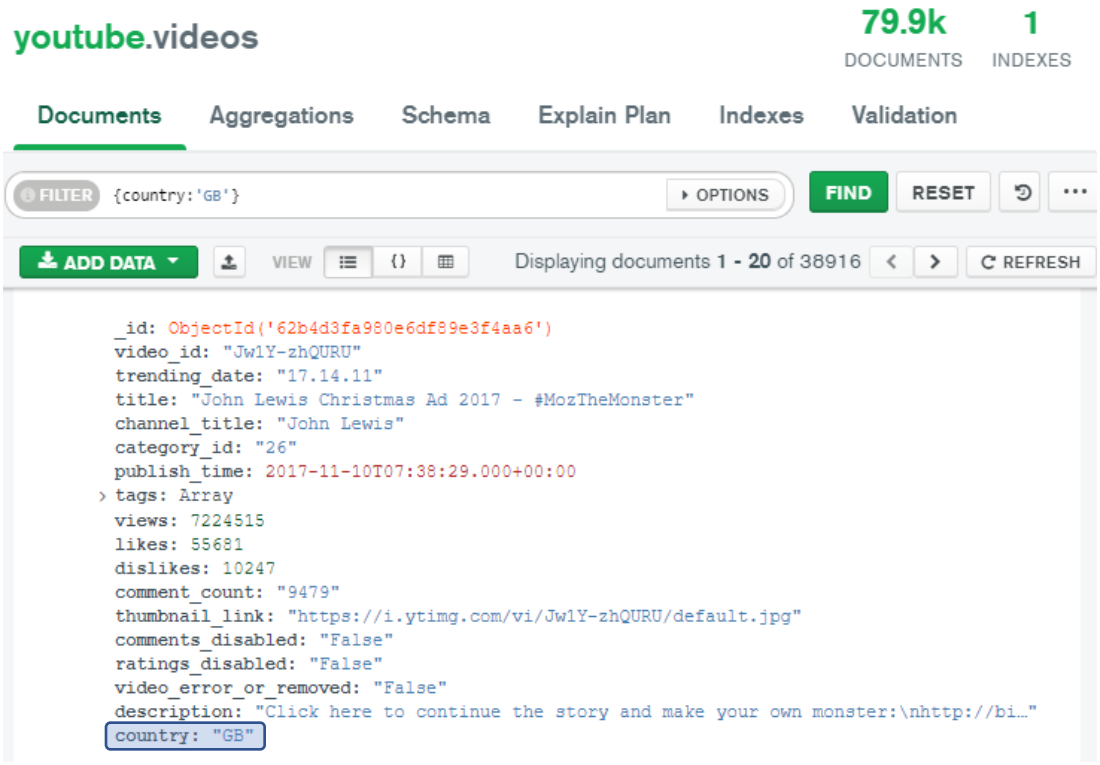
Existing Dimensions/Measures	
video_id	Unique video code
trending_date	The video date that was found in the list of popular videos (in YY.DD.MM format)
title	The title of the video
channel_title	The title of the channel that posted the video
category_id	The code of the category to which the video belongs
publish_time	The date of publication of the video (in ISO 8601 format),
tags	The tags used in the video
New Measures	
Num_tags	Numerous tags that appeared in a video
Count_tags	Count the appearance of tags

1. Preprocessing of the data

The data provided for the purposes of this project comprised of 20 files 10 in json format and the rest in csv format. Based on the assignment requirements we requested to convert the csv files to json. In order to do this, we developed a function and ran it to all csv files. The following code was used in python:

```
# Import the necessary libraries
import csv
import json
import glob
import os
import time

cwd = os.getcwd()
path =cwd+"\Data"
extension = 'csv'
all_filenames = [i for i in glob.glob('*.{}'.format(extension))]


# Create the function for converting the data

def csv_to_json(csvFilePath, jsonFilePath):
    jsonArray = []

    #read csv file
    with open(csvFilePath, encoding='latin-1') as csvf:
        #load csv file data using csv library's dictionary reader
        csvReader = csv.DictReader(csvf)

        #convert each csv row into python dict
        for row in csvReader:
            #add country in each dictionary
            row["country"] = csvFilePath[0:2]
            #add this python dict to json array
            jsonArray.append(row)
 extension = 'csv'
all_filenames = [i for i in glob.glob('*.{}'.format(extension))]

    #convert python jsonArray to JSON String and write to file
    with open(jsonFilePath, 'w', encoding='latin-1') as jsonf:
        jsonString = json.dumps(jsonArray, indent=4)
        jsonf.write(jsonString)


# Convert files

for i in all_filenames:

    csv_to_json(i, i.split('.')[0]+'.json')
```

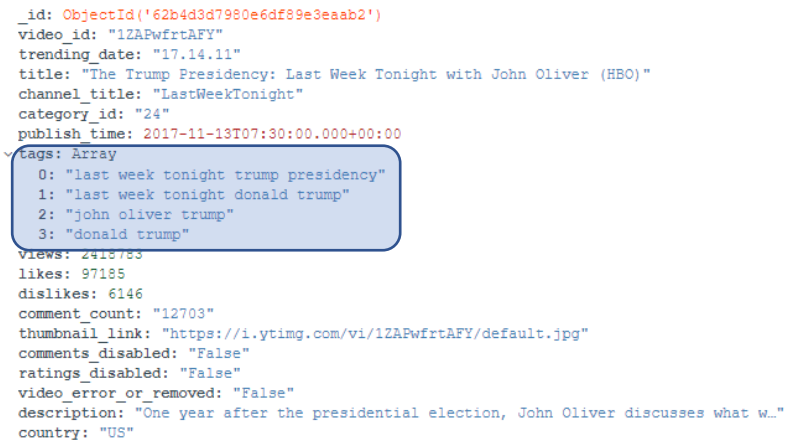
As shown in the above code we decided to add an extra name/value pair for county in order to add all the json files in one collection and separate them by filtering based on country. An example of how the data are portrayed in the compass is shown below:



Apart from this addition, in order to answer the necessary queries, some transformations were necessary. These transformations were:

- Converting publish time to date format
- Converting views, likes and dislikes from string to integer
- Converting tags in Array

The tags after the conversion[1] are in an array format and displayed in compass in the following format:



Values conversion is possible either via code or via the environment of compass and Studio 3T. For publish time we used code while for the rest the environment of Studio 3T in order to explore more options.

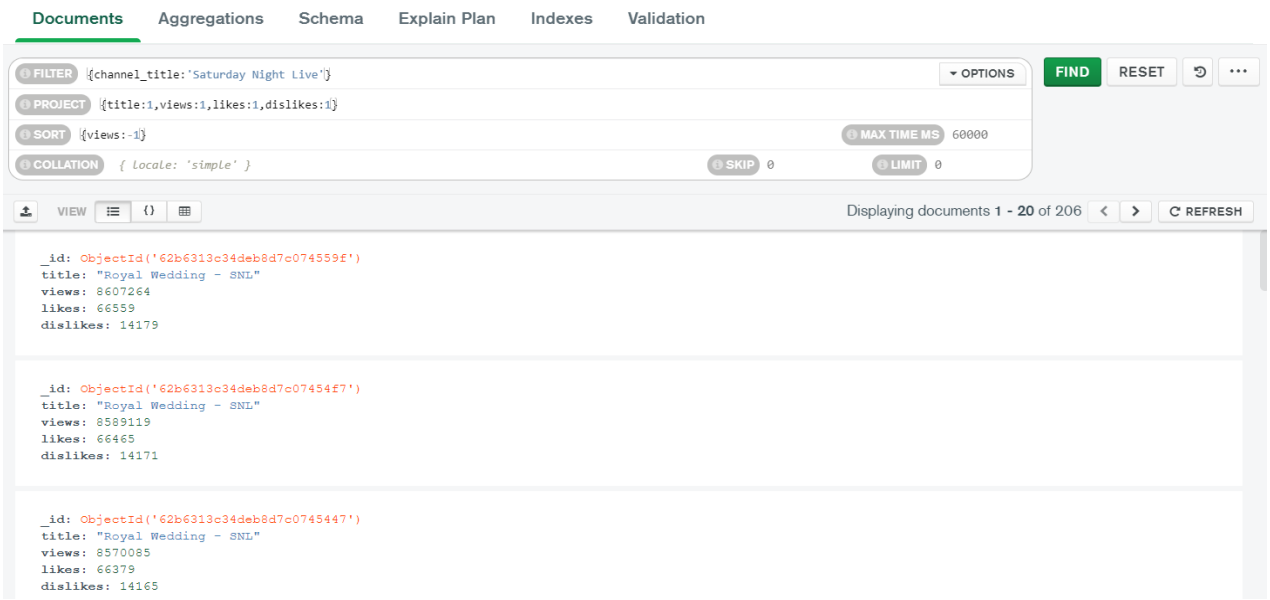
[1] The code is provided in chapter 2

2. What do we know about the publications of the very popular channel Saturday Night Live?

We found the publications of the channel "Saturday Night Live" in the GB (Grant Britain) area. For each video, we displayed the title of the video, the number of views, the number of "likes" and "dislikes" and sorted them in descending order of views.

Firstly, we imported the data into MongoDB Compass and filled in the fields as below.

MongoDB



Secondly, we copied the code and ran in Python.

Python

We displayed the results with aggregation and with mongo query:

```
q1 = videos.aggregate([
{"$match" : { "country" : "GB", 'channel_title': 'Saturday Night Live'} },
{"$project" : {"title": 1 ,"views":1,'likes': 1,'dislikes': 1}},
{"$sort": {"views": -1}}])

# convert to dataframe
p1_data = pd.DataFrame(q1)


p1_data.head(20)
```

	_id	title	views	likes	dislikes
0	62ae135f763701a6ac20ad70	Royal Wedding - SNL	8607264	66559	14179
1	62ae135f763701a6ac20acc8	Royal Wedding - SNL	8589119	66465	14171
2	62ae135f763701a6ac20ac18	Royal Wedding - SNL	8570085	66379	14165
3	62ae135f763701a6ac20ab6c	Royal Wedding - SNL	8548321	66287	14154

```
# query
filter={
    'country': 'GB',
    'channel_title': 'Saturday Night Live'
}
project={
    'title': 1,
    'views': 1,
    'likes': 1,
    'dislikes': 1
}
sort=list({
    'views': -1
}.items())


q1 = videos.find(
  filter=filter,
  projection=project,
  sort=sort
)
```

```
# convert to dataframe
q1_data = pd.DataFrame(q1)
```

```
q1_data.head(3)
```

	_id	title	views	likes	dislikes
0	62ae135f763701a6ac20ad70	Royal Wedding - SNL	8607264	66559	14179
1	62ae135f763701a6ac20acc8	Royal Wedding - SNL	8589119	66465	14171
2	62ae135f763701a6ac20ac18	Royal Wedding - SNL	8570085	66379	14165

It is observed that videos with many views have more likes as opposed to dislikes, which is to be expected because the user often watches the video he likes.

2. How many tags are commonly used in video posts?

We tried to approach the number of tags in two ways in Python. In the first way, we created a new column that counts the number of tags and the second way was to break the tags into an array.

Python

Create new column:

```
#Create a new column that counts the number of tags

gb_data['num_tags'] = gb_data['tags'].str.split('|').str.len()
gb_data.head()
```

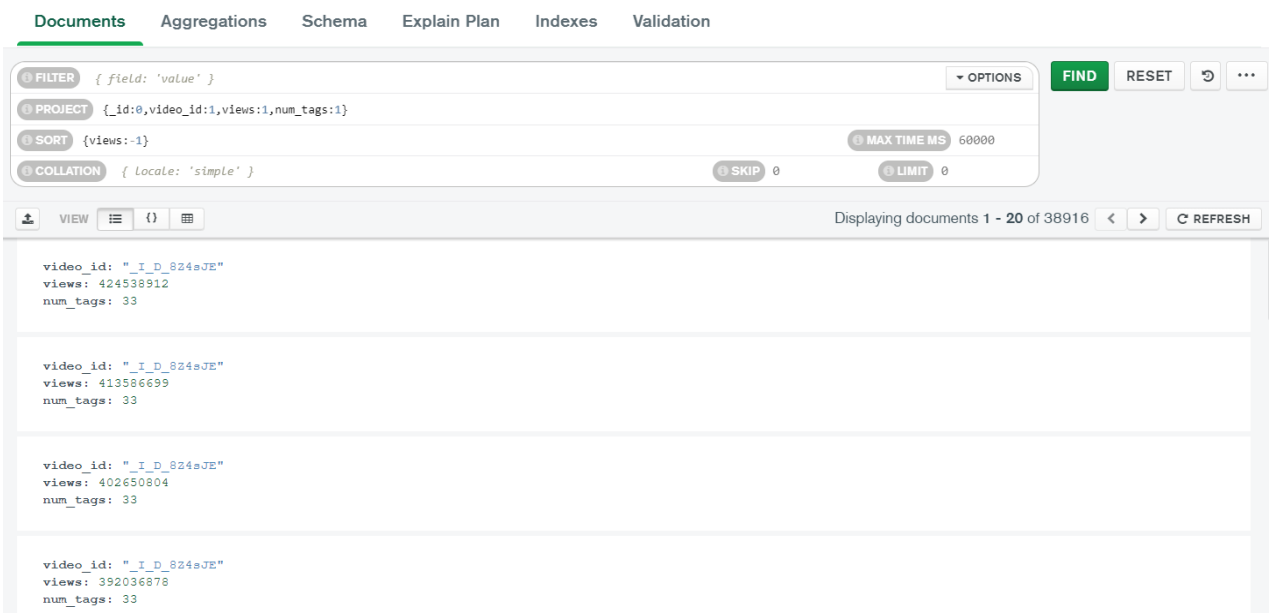
Break tags into an array:

```
# Break tags into an Array

tags_format = [
   {
      "$addFields":{
         "tags":{
            "$replaceAll":{
               "input":"$tags",
               "find":"\"",
               "replacement":""
            }
         }
      }
   },
   {
      "$addFields":{
         "tags":{
            "$split":[
               "$tags",
               "|"
            ]
         }
      }
   }
]

videos.update_many({}, tags_format)
```

```
<pymongo.results.UpdateResult at 0x1aab5a670c8>
```

After calculating the number of tags, we imported the new file into MongoDB Compass and selected the required fields.

MongoDB

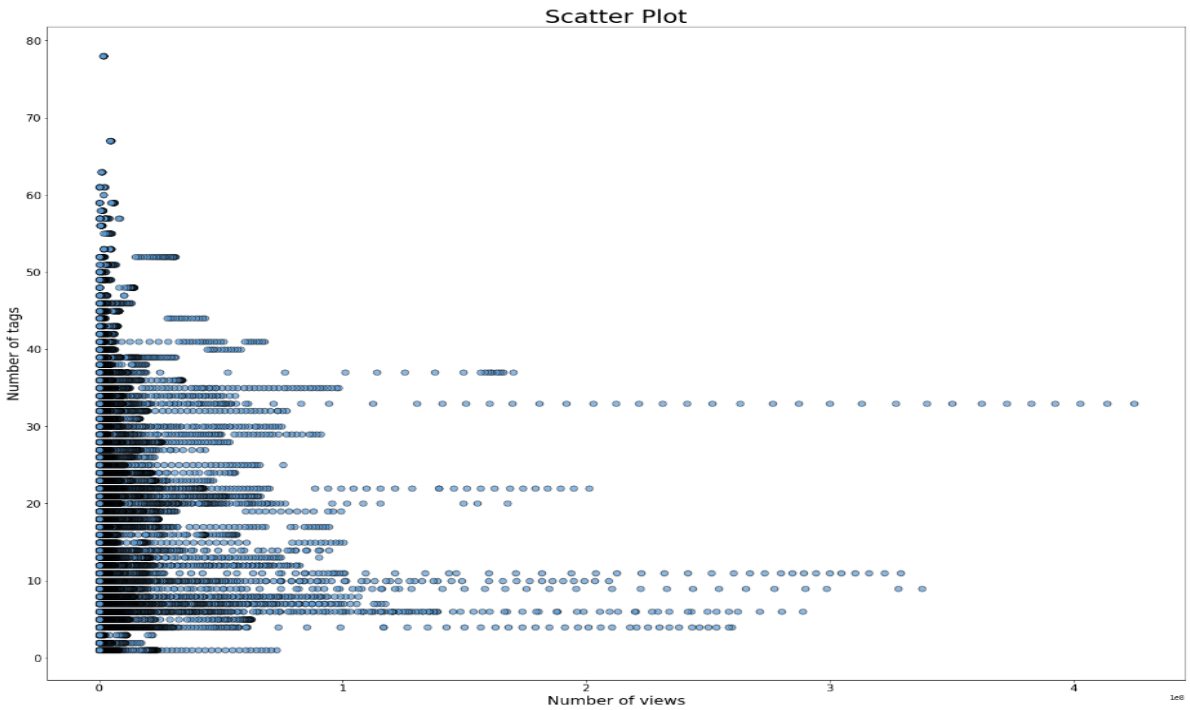


As mentioned above, we copied the code to run in Python and created a scatter plot that shows the number of views on the x-axis and the number of tags on the y-axis.

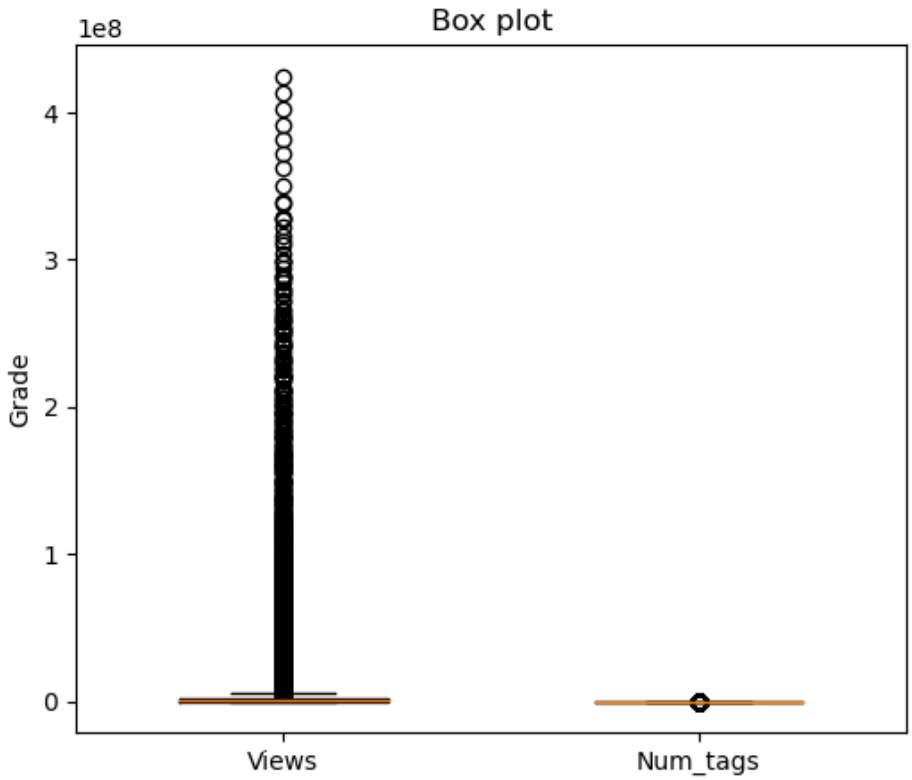
Python

```
from pymongo import MongoClient

# Requires the PyMongo package.
# https://api.mongodb.com/python/current

client = MongoClient('mongodb+srv://YoutubeProject_v2:123@cluster0.amrnw.mongodb.net/test?authSource=admin')
filter={}
project={
    '_id': 0,
    'video_id': 1,
    'views': 1,
    'num_tags': 1
}
sort=list({
    'views': -1
}.items())

result = client['Youtube_Music']['Question 2.2'].find(
  filter=filter,
  projection=project,
  sort=sort
)
# convert to dataframe
q2_data = pd.DataFrame(result)
q2_data.head(10)
```

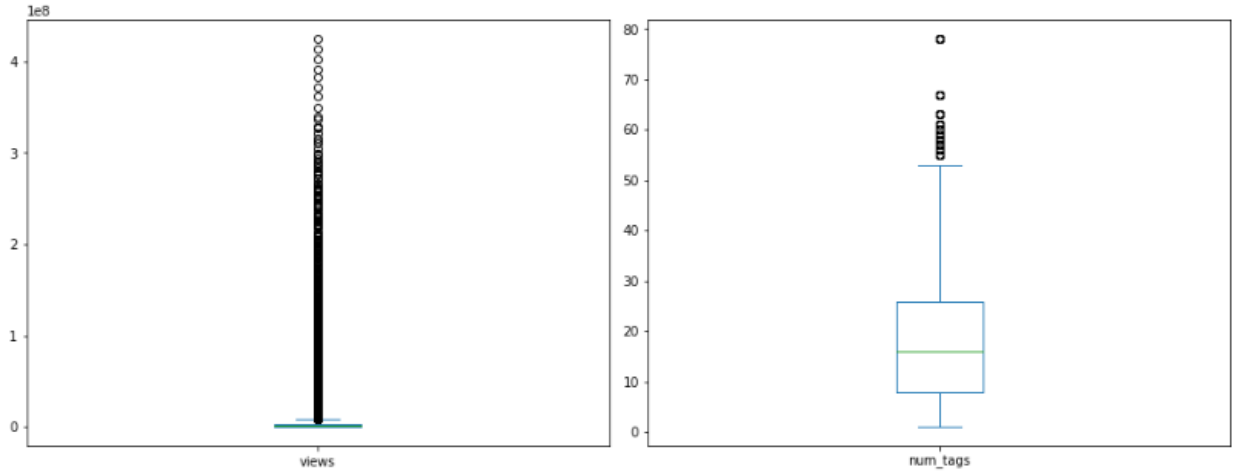
	video_id	views	num_tags
0	_I_D_8Z4sJE	424538912	33
1	_I_D_8Z4sJE	413586699	33
2	_I_D_8Z4sJE	402650804	33
3	_I_D_8Z4sJE	392036878	33
4	_I_D_8Z4sJE	382401497	33
5	_I_D_8Z4sJE	372399338	33
6	_I_D_8Z4sJE	362111555	33
7	_I_D_8Z4sJE	349987176	33
8	_I_D_8Z4sJE	339629489	33
9	9jl-z9QN6g8	337621571	9



A useful graph for understanding the extreme values in this case is the box plot:



Because the range of numbers is different, we split the graphs into two to better understand the outliers:



According to our results, the number of views and tags are independent measures. The number of views from the tags is not affected, so the views in a video do not increase if it has more tags. Clearly, in each case, there are outliers but the issue of views we consider to be other factors and not the tags. For example, the large number of views may be due to the advertising of the video that has been made, the artist the type of video.

3. What are the most popular tags in upcoming videos?

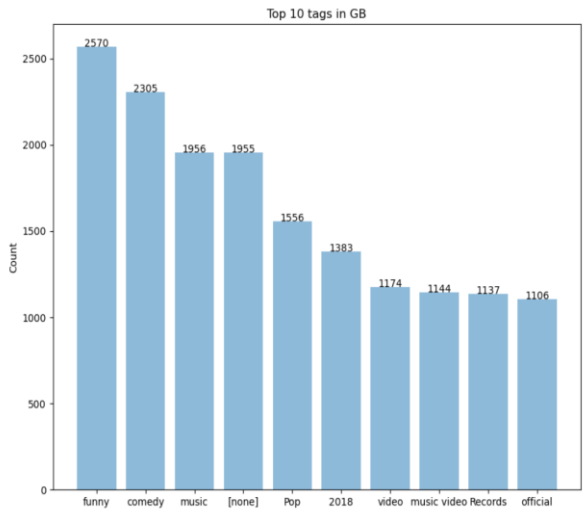
We calculated the number of videos for each tag in GB and USA.

Python

Grant Britain (GB)

```
q3_1 = videos.aggregate([
    {"$match" : { "country" : "GB", 'comments_disabled': 'False'}},
    { "$unwind": "$tags"},
    {"$group" : { "_id" : "$tags"  ,"count_tags": {"$count":{}}}},
    {"$sort": {"count_tags":-1}}
])


# convert to dataframe
q3_1_data = pd.DataFrame(q3_1)


q3_1_data.head(20)
```

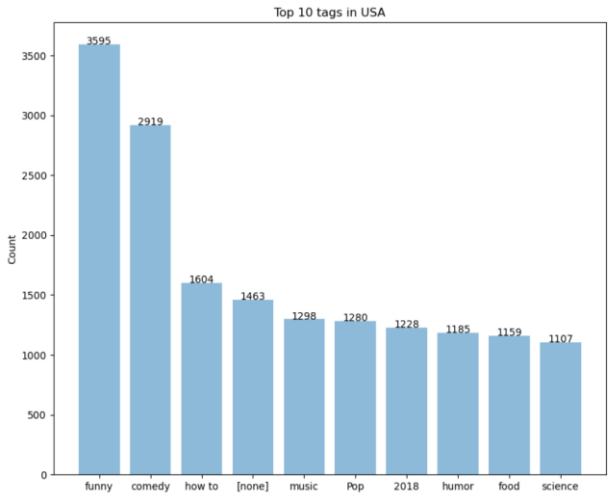
	_id	count_tags
0	funny	2570
1	comedy	2305
2	music	1956
3	[none]	1955
4	Pop	1556
5	2018	1383
6	video	1174
7	music video	1144



USA

```
q3 = videos.aggregate([
    {"$match" : { "country" : "US", 'comments_disabled': 'False'}},
    { "$unwind": "$tags"},
    {"$group" : { "_id" : "$tags"  ,"count_tags": {"$count":{}}}},
    {"$sort": {"count_tags":-1}}
])


# convert to dataframe
q3_data = pd.DataFrame(q3)


q3_data.head(20)
```

	_id	count_tags
0	funny	3595
1	comedy	2919
2	how to	1604
3	[none]	1463
4	music	1298
5	Pop	1280
6	2018	1228
7	humor	1185



It is observed that the main labels such as funny, comedy appear in both areas. In general, the number of labels in the US is larger than the GB, this makes sense as one area has a larger population than the other. Note that there are many "None" in the two regions, perhaps because the youtuber has blocked the tags or the user did not find the video interesting.

4. What impact does the deactivation of comments have on the public?

We calculated the average number of views, likes and dislikes per comment disabled in Python.

Python

<table>
<tr><th>Comments Disabled=False</th><th>Comments Disabled=True</th></tr>
<tr><td>

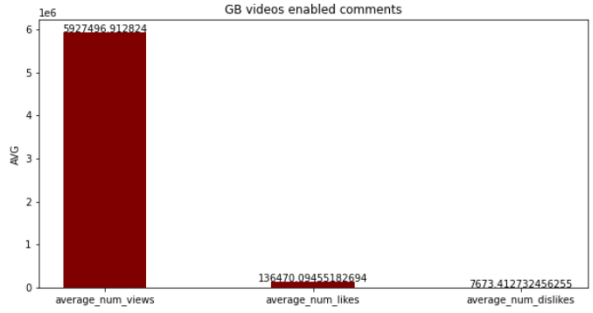
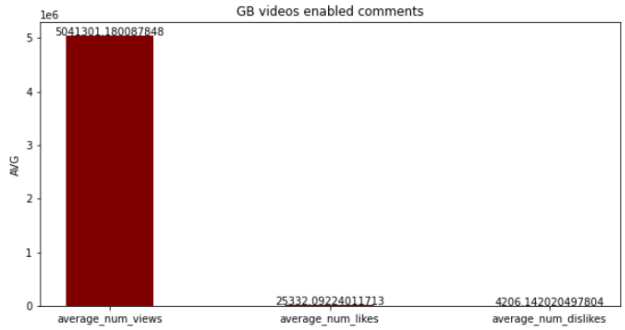
```
q4 = videos.aggregate([

    {"$match" :
                  { "country" : "GB", 'comments_disabled': 'False'}

    },

    {
        "$group" :
            {
              "_id" : "Enabled" ,
              "average_num_views": { "$avg" : "$views"},
              "average_num_likes": { "$avg" : "$likes"},
              "average_num_dislikes":{"$avg" :"$dislikes"}
            }
    }

])


# convert to dataframe
q4_data = pd.DataFrame(q4)


q4_data.head(20)
```

	_id	average_num_views	average_num_likes	average_num_dislikes
0	Enabled	5.927497e+06	136470.094552	7673.412732

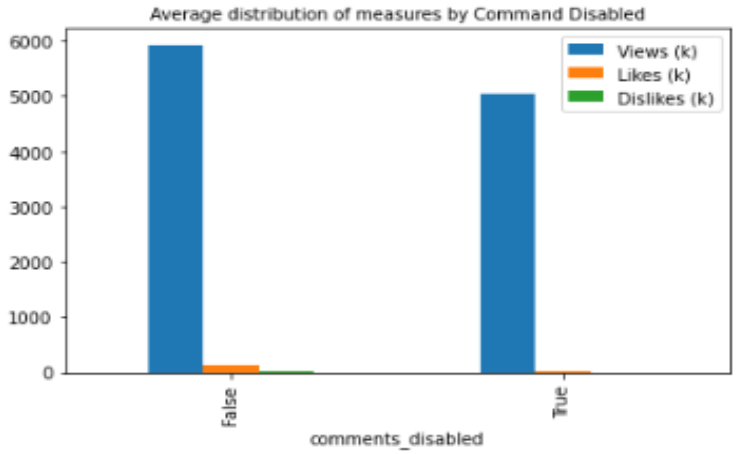
</td><td>

```
q4 = videos.aggregate([

    {"$match" :
                  { "country" : "GB", 'comments_disabled': 'True'}

    },

    {
        "$group" :
            {
              "_id" : "Disabled" ,
              "average_num_views": { "$avg" : "$views"},
              "average_num_likes": { "$avg" : "$likes"},
              "average_num_dislikes":{"$avg" :"$dislikes"}
            }
    }

])


# convert to dataframe
q4_data = pd.DataFrame(q4)


q4_data.head(20)
```

	_id	average_num_views	average_num_likes	average_num_dislikes
0	Disabled	5.041301e+06	25332.09224	4206.14202

</td></tr>
<tr><td> 
</td><td> 
</td></tr>
</table>

To have a complete picture, we divided all the measures by 1000 so that we can compare them.



According to our results, users did not prefer video comments to be turned off. The specific preference can be found with the number of views. The videos that have the "dislike" comments off are no more than "likes". Users do not think that disabling comments is a good way to avoid negative attention.

5. What were the most popular dates for video posting?

Python

We approached the request in two ways. Initially, we changed the data format to display the video publish date split by year, month and day.

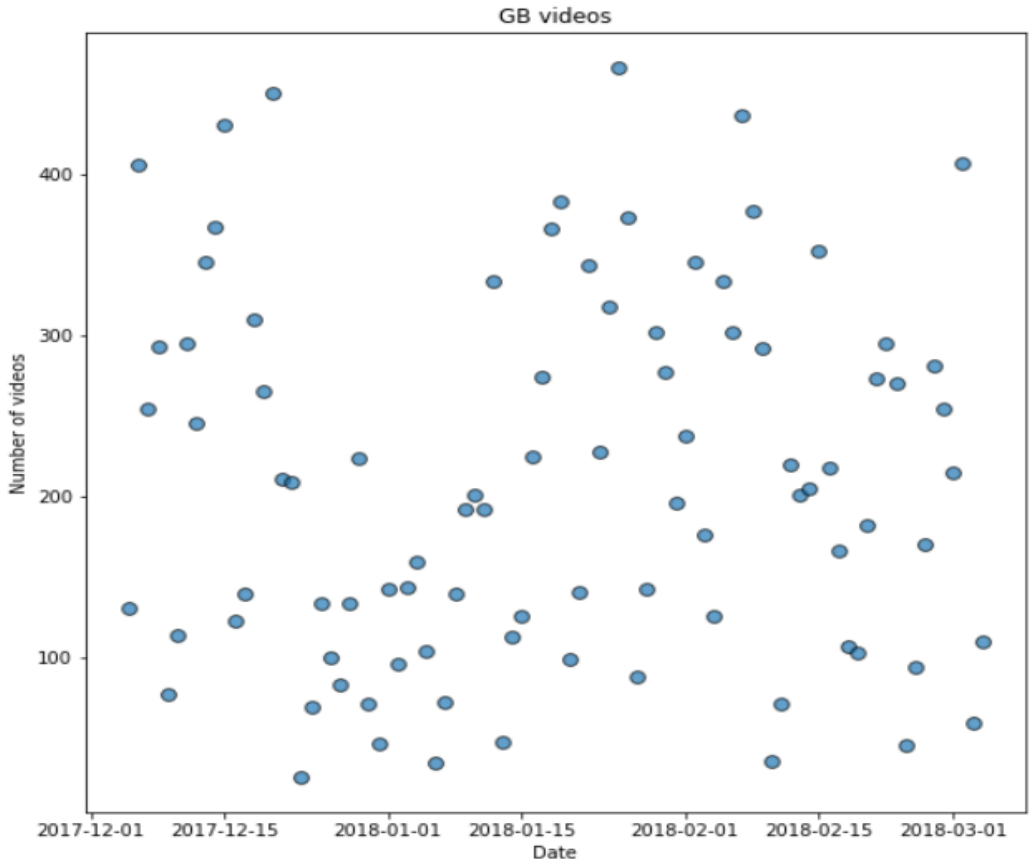
```
# Change date format

time_format = [
    {
        '$set': {
            'publish_time': {
                '$dateFromString': {
                    'dateString': '$publish_time'
                }
            }
        }
    }
]
videos.update_many({}, time_format)
```

We limited the period between December 5, 2017 and March 5, 2018 to calculate the number of videos per day.

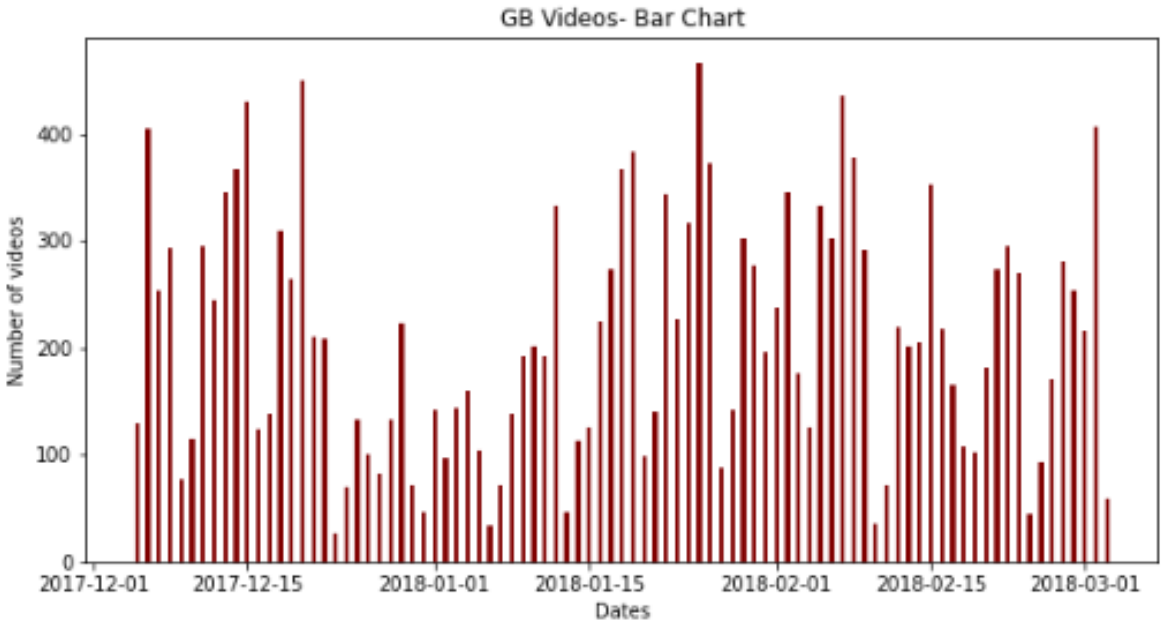
```
from datetime import datetime

# convert your date string to datetime object
start = datetime(2017, 12, 5)
end = datetime(2018, 3, 5)


q5 = videos.aggregate([
    {"$match" :{'publish_time': {'$lt': end, '$gte': start}, "country" : "GB"}},
      {
        "$group": {
          "_id": {
            "year": { "$year": "$publish_time" },
            "month": { "$month": "$publish_time" },
            "day": { "$dayOfMonth": "$publish_time"}
          },
          "count": { "$sum": 1 }
        }
      },
    {"$sort": {"_id":1}}
    ])


# convert to dataframe
q5_data = pd.DataFrame(q5)


q5_data.head(20)
```

	_id	count
0	{'year': 2017, 'month': 12, 'day': 5}	130
1	{'year': 2017, 'month': 12, 'day': 6}	405
2	{'year': 2017, 'month': 12, 'day': 7}	254

We represented the data on Scatter plot. At first glance, it is observed that most videos are played in December'17 and February'18.



Due to the volume of data, we could not conclude anything from the scatter plot. There is not clear correlation between Date and the number of videos. We decided to visualize the data with a bar graph showing value fluctuations. It is noticed that most of the videos were published at the end of the year and most of them appear at the end of January'18. In addition, we found that most of the videos are published at the end of each month.



Conclusion

In recent years, people's interest in YouTube videos has increased. A useful tool for managing the volume of data produced by the videos is Mongo DB, which we also used in our work to answer the questions posed to us. After preprocessing our data, we focused on GB and USA areas primarily. We concluded that the main tags are "funny" and "comedy" that appear in both areas and we noticed that users do not prefer to disable comments on videos. Most videos were published at the end of the year 2017 and in January 2018.

Analytical Results

2.1 Query (First 20 results)

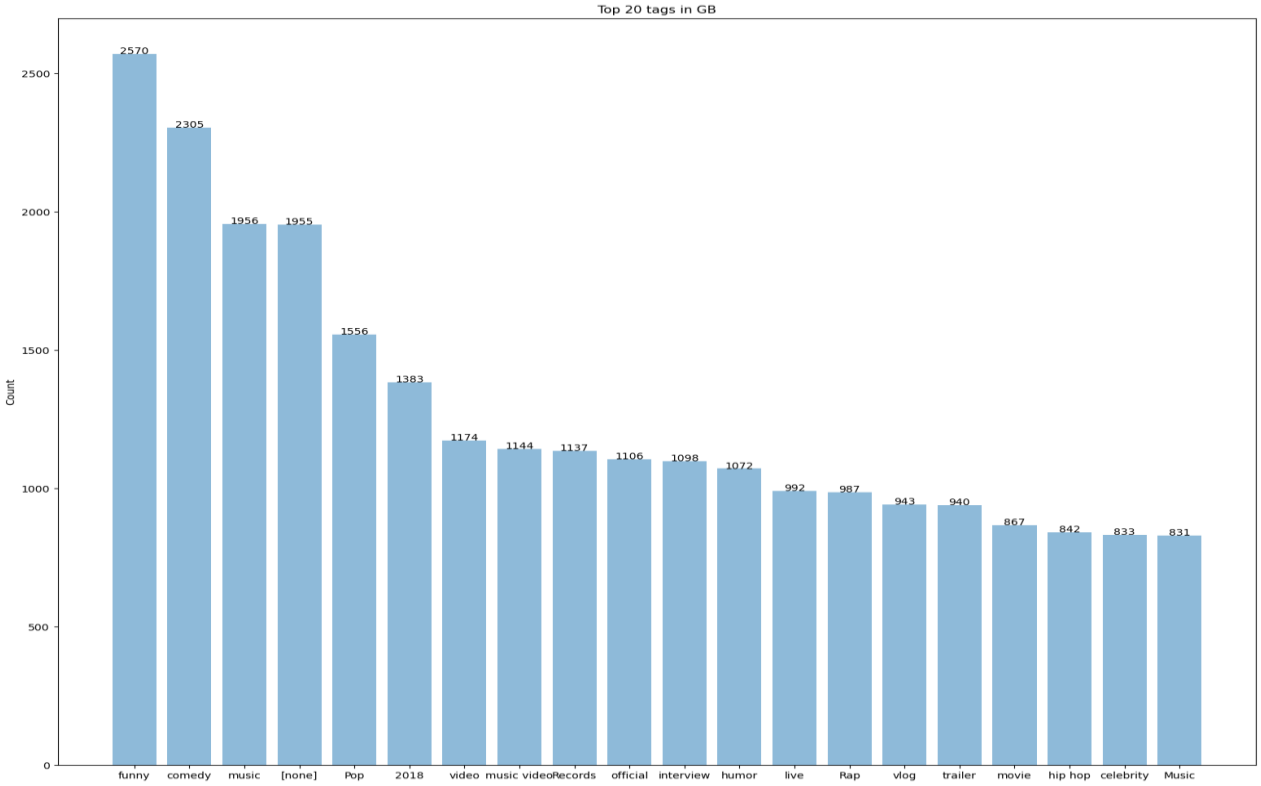
	_id	title	views	likes	dislikes
0	62b4d400980e6df89e3fe27f	Royal Wedding - SNL	8607264	66559	14179
1	62b4d400980e6df89e3fe1d7	Royal Wedding - SNL	8589119	66465	14171
2	62b4d400980e6df89e3fe127	Royal Wedding - SNL	8570085	66379	14165
3	62b4d400980e6df89e3fe07b	Royal Wedding - SNL	8548321	66287	14154
4	62b4d400980e6df89e3fdfd3	Royal Wedding - SNL	8534145	66248	14148
5	62b4d400980e6df89e3fdf21	Royal Wedding - SNL	8503729	66130	14116
6	62b4d400980e6df89e3fde6e	Royal Wedding - SNL	8482595	66039	14099
7	62b4d400980e6df89e3fddbd	Royal Wedding - SNL	8457468	65930	14088
8	62b4d400980e6df89e3fdd0d	Royal Wedding - SNL	8431913	65829	14075
9	62b4d400980e6df89e3fdc60	Royal Wedding - SNL	8390391	65649	14046
10	62b4d400980e6df89e3fdbb7	Royal Wedding - SNL	8353873	65499	14015
11	62b4d400980e6df89e3fdb13	Royal Wedding - SNL	8303192	65282	13960
12	62b4d400980e6df89e3fda6b	Royal Wedding - SNL	8256003	65091	13922
13	62b4d400980e6df89e3fd9bf	Royal Wedding - SNL	8208371	64862	13884
14	62b4d400980e6df89e3fd914	Royal Wedding - SNL	8166389	64706	13863
15	62b4d400980e6df89e3fd860	Royal Wedding - SNL	8069531	64279	13751
16	62b4d400980e6df89e3fd7b7	Royal Wedding - SNL	7981097	63919	13667
17	62b4d400980e6df89e3fd70c	Royal Wedding - SNL	7834146	63310	13483
18	62b4d400980e6df89e3fd660	Royal Wedding - SNL	7655742	62655	13269
19	62b4d400980e6df89e3fd5b1	Royal Wedding - SNL	7436025	61913	13015

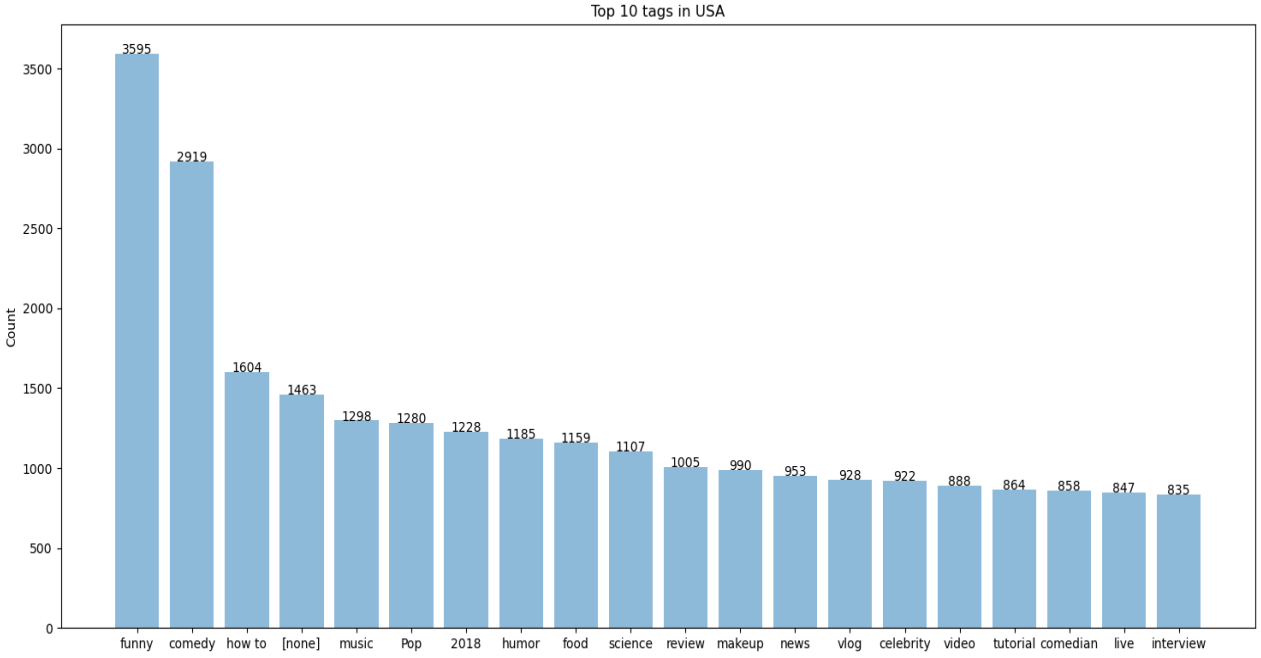
2.2 Query (First 20 Results)

	_id	video_id	views	num_tags
0	62b4d3ff980e6df89e3fb9a2	_I_D_8Z4sJE	424538912	33
1	62b4d3ff980e6df89e3fb8da	_I_D_8Z4sJE	413586699	33
2	62b4d3ff980e6df89e3fb80e	_I_D_8Z4sJE	402650804	33
3	62b4d3fe980e6df89e3fb749	_I_D_8Z4sJE	392036878	33
4	62b4d3fe980e6df89e3fb685	_I_D_8Z4sJE	382401497	33
5	62b4d3fe980e6df89e3fb5c6	_I_D_8Z4sJE	372399338	33
6	62b4d3fe980e6df89e3fb50f	_I_D_8Z4sJE	362111555	33
7	62b4d3fe980e6df89e3fb452	_I_D_8Z4sJE	349987176	33
8	62b4d3fe980e6df89e3fb393	_I_D_8Z4sJE	339629489	33
9	62b4d400980e6df89e3fd142	9jl-z9QN6g8	337621571	9
10	62b4d3fe980e6df89e3fae6f	kLpH1nSLJSs	328860380	11
11	62b4d400980e6df89e3fd09a	9jl-z9QN6g8	328024035	9
12	62b4d3fe980e6df89e3fb2d5	_I_D_8Z4sJE	326890258	33
13	62b4d3fe980e6df89e3fada3	kLpH1nSLJSs	322211201	11
14	62b4d3fe980e6df89e3facd7	kLpH1nSLJSs	315860445	11
15	62b4d3fe980e6df89e3fb213	_I_D_8Z4sJE	312945464	33
16	62b4d3fe980e6df89e3fac0d	kLpH1nSLJSs	310074566	11
17	62b4d3fe980e6df89e3fab48	kLpH1nSLJSs	304530584	11
18	62b4d3fe980e6df89e3faa82	kLpH1nSLJSs	299857997	11
19	62b4d3fe980e6df89e3fb14e	_I_D_8Z4sJE	299697019	33

2.3 Query (First 20 results)

	_id	count_tags
0	funny	2570
1	comedy	2305
2	music	1956
3	[none]	1955
4	Pop	1556
5	2018	1383
6	video	1174
7	music video	1144
8	Records	1137
9	official	1106
10	interview	1098
11	humor	1072
12	live	992
13	Rap	987
14	vlog	943
15	trailer	940
16	movie	867
17	hip hop	842
18	celebrity	833
19	Music	831


Top 20 tags in GB
Count
2570
2305
1956
1955
1556
1383
1174
1144
1137
1106
1098
1072
992
987
943
940
867
842
833
831
funny
comedy
music
[none]
Pop
2018
video
music video
Records
official
interview
humor
live
Rap
vlog
trailer
movie
hip hop
celebrity
Music


Top 10 tags in USA
Count
3595
2919
1604
1463
1298
1280
1228
1185
1159
1107
1005
990
953
928
922
888
864
858
847
835
funny
comedy
how to
[none]
music
Pop
2018
humor
food
science
review
makeup
news
vlog
celebrity
video
tutorial
comedian
live
interview

2.5 Query (First 20 results)

	_id	count
0	{'year': 2017, 'month': 12, 'day': 5}	130
1	{'year': 2017, 'month': 12, 'day': 6}	405
2	{'year': 2017, 'month': 12, 'day': 7}	254
3	{'year': 2017, 'month': 12, 'day': 8}	293
4	{'year': 2017, 'month': 12, 'day': 9}	77
5	{'year': 2017, 'month': 12, 'day': 10}	114
6	{'year': 2017, 'month': 12, 'day': 11}	295
7	{'year': 2017, 'month': 12, 'day': 12}	245
8	{'year': 2017, 'month': 12, 'day': 13}	345
9	{'year': 2017, 'month': 12, 'day': 14}	367
10	{'year': 2017, 'month': 12, 'day': 15}	430
11	{'year': 2017, 'month': 12, 'day': 16}	123
12	{'year': 2017, 'month': 12, 'day': 17}	139
13	{'year': 2017, 'month': 12, 'day': 18}	309
14	{'year': 2017, 'month': 12, 'day': 19}	265
15	{'year': 2017, 'month': 12, 'day': 20}	450
16	{'year': 2017, 'month': 12, 'day': 21}	211
17	{'year': 2017, 'month': 12, 'day': 22}	209
18	{'year': 2017, 'month': 12, 'day': 23}	26
19	{'year': 2017, 'month': 12, 'day': 24}	69