Analysis of Youtube Trending Videos

Mudit Aggarwal 2020UCD2157, Rupali Bisht 2020UCD2121

March 23, 2023

Abstract

This Project encompasses the ingestion and analysis of Youtube trending videos from 12th August 2020 till date.

1 Overview

Youtube is one of the most popular social media sites with over 122 million daily active users and more than 1 billion hours of content spread across the globe. globalmediainsight.
One of the most popular and important features of the youtube website is unarguably the trending tab (recently name chanegd to explore). This section contains some of the most popular/upcoming videos of a region. It was one of the first features added to the youtube site which enabled content creators on the platform to reach a wider audience.

2 Problem Statement

Youtube trending videos dataset is an important asset which allows us to monitor the growth of the platform and change in paradigm of the amount and type of content present on the platform.

But there is a significant problem, there is no publicly available official historical dataset of youtube trending videos. The youtube API only allows for ”mostPopular” video list of the date of the API call.
So this project is aimed to create a tool to regularly query and store youtube trending videos data set so as to eventually create a historical data set and perform analysis on it to understand YouTube usage and content paradigms.

Thus the steps taken to accomplish this are:

1. Create a persistent PostgreSQL database on Microsoft Azure cloud platform
2. Run a regular cloud edge computing function to query Youtube API and store the current date most popular videos for 10 countries/regions
3. Create an Analysis script to query that data and draw insights from it

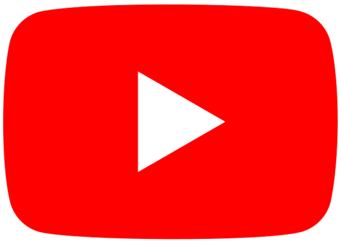
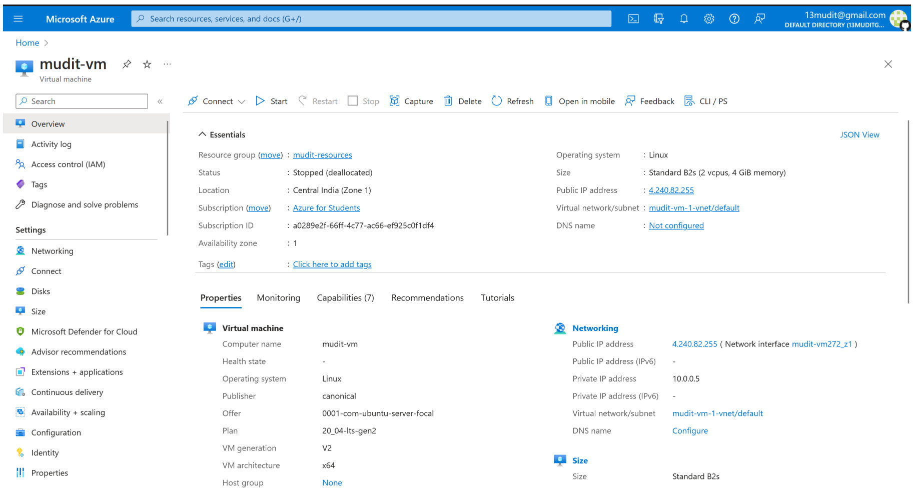


Figure 1: Youtube, the most popular Video streaming service.

Figure 2: Some major cloud providers.



mudit-vm
Virtual machine

Essentials

Resource group (move) : mudit-resources
Status : Stopped (deallocated)
Location : Central India (Zone 1)
Subscription (move) : Azure for Students
Subscription ID : a0289e2f-66ff-4c77-ac66-ef925c0f1df4
Availability zone : 1
Tags (edit) : Click here to add tags

Operating system : Linux
Size : Standard B2s (2 vcpus, 4 GiB memory)
Public IP address : 4.240.82.255
Virtual network/subnet : mudit-vm-1-vnet/default
DNS name : Not configured

Virtual machine

Computer name	mudit-vm
Health state	-
Operating system	Linux
Publisher	canonical
Offer	0001-com-ubuntu-server-focal
Plan	20_04-lts-gen2
VM generation	V2
VM architecture	x64
Host group	None

Networking

Public IP address	4.240.82.255 (Network interface mudit-vm272_z1)
Public IP address (IPv6)	-
Private IP address	10.0.0.5
Private IP address (IPv6)	-
Virtual network/subnet	mudit-vm-1-vnet/default
DNS name	Configure

Size

Size	Standard B2s

Figure 3: Azure VM console overview.

3 Create a Cloud PostgreSQL Server

3.1 Introduction

Cloud services like Microsoft Azure, Google Cloud Platform, AWS, Alibaba Cloud provide a persistent machine running in a server. These machines are very useful for tasks like data storage or edge computing as they are fully maintained and we don't have to rely on our local machine's resources.

These services are generally aimed at larger organisations as it is much more convenient for them to deploy to a cloud instance than set up their own servers.

3.2 Creating a cloud instance on Azure

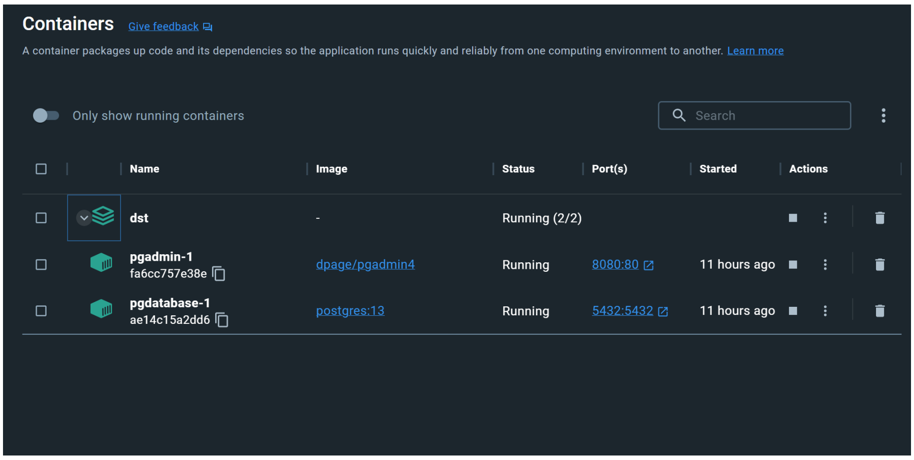
Of the popular Cloud Service Providers in 2, we decided to go with Microsoft Azure as it provides a $100 credit to Students enrolled in an accredited institute without the need of any credit card.

To get started with Azure, we created a Microsoft Azure account and started a Cloud Virtual Machine 3

This enabled us to control a virtual machine with 2 Compute cores and 4 GiB of RAM

3.3 Create a Docker Container to Host PostgreSQL server

Next step was to start a PostgreSQL server on the created VM. Instead of directly running the PostgreSQL server instance, we checked and monitored its use and performance on our local machine, then to ensure identical behaviour we set up a docker container and deployed that container on the cloud instance.



Containers Give feedback

A container packages up code and its dependencies so the application runs quickly and reliably from one computing environment to another. Learn more

Only show running containers

Search

□	Name	Image	Status	Port(s)	Started	Actions
□	dst	-	Running (2/2)			
□	pgadmin-1 fa6cc757e38e	dpage/pgadmin4	Running	8080:80	11 hours ago	
□	pgdatabase-1 ae14c15a2dd6	postgres:13	Running	5432:5432	11 hours ago	

Figure 4: Running docker containers on local machine.

4 Interfacing with Youtube API

4.1 Introduction

Yotube API is a part of a multitude of freely accessbile services under Google APIs. Google APIs documentation and reference is very well documented and easy to use. To learn more Google API explorer

4.2 Youtube API

To get started with YouTube API, refer to youtueb api v3 reference

Particularly for Trending videos the API query will look like:

```
request = youtube.videos().list(
    part=”snippet,statistics”,
    chart=”mostPopular”,
    regionCode=country_code,
    maxResults=50,
)
response = request.execute()
```

And the response will look like:

```
{
  ”kind”: ”youtube#videoListResponse”,
  ”etag”: etag,
  ”nextPageToken”: string,
  ”prevPageToken”: string,
  ”pageInfo”: {
    ”totalResults”: integer,
    ”resultsPerPage”: integer
  },
  ”items”: [
    video Resource
  ]
}
```

We Also query the the Youtube Video Cateogories index, which contains the snippet names and category numbers of all the video Categories.

This index can be directly joined with Video List reponse to get the category name of any given video

Category response query:

```
def get_video_categories(country_code):
    request = youtube.videoCategories().list(
        part="snippet",
        regionCode=country_code
    )
    return request.execute()
```

Category Response:

```
{
  "kind": "youtube#videoCategoryListResponse",
  "etag": etag,
  "nextPageToken": string,
  "prevPageToken": string,
  "pageInfo": {
    "totalResults": integer,
    "resultsPerPage": integer
  },
  "items": [
    videoCategory resource
  ]
}
```

5 Uploading Data to Cloud Storage

Finally after extracting data we upload the data to the PostgreSQL server in Microsoft Azure using Pandas and SQLalchemy module of python.

We also use the historical data collected by Rishav Sharma and upload it to the cloud database too.