YouTube Trending Video

Data Engineering Capstone Project
- Project Summary
- Step 1: Scope the Project and Gather Data
- Step 2: Explore and Asses the Data
- Define the Data Model
- Run pipelines to Model Data
- **Complete Project Write Up**

Data Engineering Capstone Project

Project Summary

The aims of this project is to build an etl pipeline on the daily record of the trending YouTube Videos. the end result is a star-schema model that gives us the possibility to explore and analyze the data in a multidimensional way.

YouTube maintains a list of the top trending videos on the platform. YouTube uses a combination of factors including measuring users interactions (number of views, shares, comments and likes). Note that they’re not the most-viewed videos overall for the calendar year”.

The source code is found at https://github.com/EngIcaro/YouTube-Trending.

Step 1: Scope the Project and Gather Data

Scope

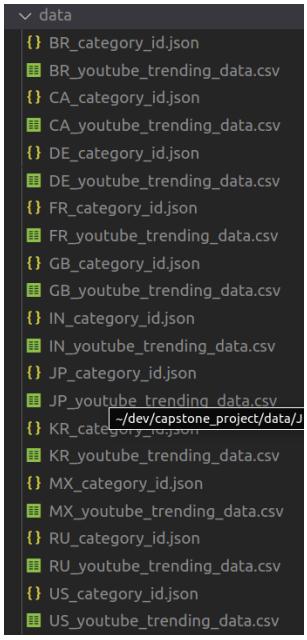
My plan was to extract the data from the .csv and .json sources, transform and clean all data to save in staging tables and finally build an olap cube for further analysis. All tables have been created in a local postgres database.

The dataset used in this project is (https://www.kaggle.com/datasets/rsrishav/youtube-trending-video-dataset?select=BR_youtube_trending_data.csv). This dataset includes several months of data on daily trending YouTube videos. Data is included for the IN, US, GB, DE, CA, FR, RU, BR, MX, KR, and JP regions (India, USA, Great Britain, Germany, Canada, France, Russia, Brazil, Mexico, South Korea, and, Japan respectively)

My end solution is a star schema in a postgres database. I used python, pandas, sql, postgres, docker and git.

Describe and Gather Data

The dataset consists of .csv and .json files. Each region's data is separate by .csv files and the .json files includes a caterogy_id and category_title which varies between regions.



The data come from The Kaggle DataSet (https://www.kaggle.com/datasets/rsrishav/youtube-trending-video-dataset?select=BR_youtube_trending_data.csv). Where This dataset was collected using the YouTube API(https://developers.google.com/youtube/v3).

The type of information included in *_youtube_trending_data.csv are :

Nome da coluna	#	Tipo de dado
staging_youtube	1	serial4
video_id	2	varchar
title	3	varchar
published_at	4	varchar
channel_id	5	varchar
channel_title	6	varchar
category_id	7	int2
trending_date	8	varchar
tags	9	varchar
view_count	10	int4
likes	11	int4
dislikes	12	int4
comment_count	13	int4
thumbnail_link	14	varchar
comments_disal	15	varchar
rating	16	varchar
description	17	varchar
country	18	varchar

some *_youtube_trending_data.csv samples:

video_id	title	published_at	channel_id	channel_title
w0BI68Ro-xU	Hiosaki - Desculpa	2020-08-09T23:30:11Z	UCBnjBPhqZdk5_70oaYuNyDw	Hiosaki
fz_4tSiYDvU	CBLoL 2020: 2ª Etapa - Fase de Pontos - Md1 | Semana 10 -	2020-08-09T20:44:27Z	UC48rkTlXjRd6pnqqBkdV0Mw	LoL eSports BR
GaKmUXdzT0Y	NETO DETONA LUAN POR NÃO BATER PÊNALTI CONTRA (	2020-08-09T01:14:05Z	UCg4y0neDAbTFqkcWQtBnWsA	Craque Neto 10
NMvOvq7GDbk	FLAMENGO 0x1 ATLÉTICO MG - CAMPEONATO BRASILEIR	2020-08-09T21:52:42Z	UC7Iw8YAkr08cuFYbVnmsw3w	CANHOTINHA 70

trending_date	tags	view_count	likes	dislikes	comment_count	thumbnail_link
2020-08-12T00:00:00Z	hirosaki|sad|não sou eu|sad songs|sad music|sad musics|	74.914	20.298	70	1.212	https://i.ytimg.com/vi/w0BI6
2020-08-12T00:00:00Z	CBLoL 2020|CBLoL 2020 1ª Etapa|CBLoL|Riot Games|LoLE	747.524	24.785	522	118	https://i.ytimg.com/vi/fz_4tS
2020-08-12T00:00:00Z	craque neto|craque neto 10|neto|craqueneto10|craque ne	570.150	28.675	1.715	3.616	https://i.ytimg.com/vi/GaKm
2020-08-12T00:00:00Z	[None]	86.879	11.866	1.088	2.943	https://i.ytimg.com/vi/NMvO

	comments_disabled	rating	description	country
g	false	false	Adicione Minha Nova Música em sua Playlist!smarturl.it/H	BR
	false	false	CBLoL 2020: 2ª Etapa acontece de 06 de Junho a 5 de Sete	BR
pg	false	false	Após a derrota do Corinthians para o Palmeiras, mandei	BR

The type of information included in *_category_id.json are :

Step 2: Explore and Asses the Data

Explore the Data

To do this step i created a python script called eda.py. In this script i created a few functions to check nan values, check unique values and explore the data behavior.

```
def check_nan_values(data_base):
    """This function checks if any column has a nan values
    Args:
        data_base (dataframe): database
    """
    for column in data_base:
        print(column,' ', data_base[column].isnull().values.any())
        print(data_base[column].isnull().sum())
```

```
# All columns has repeted values
def check_unique_values(data_base):
    """This functions checks if any column has repeted values
    Args:
        data_base (dataframe): database
    """
    for column in data_base:
        values_list = data_base[column].value_counts().values
        for values in values_list:
            if values > 1:
                print(column, ' Has repeted values')
                break
```

```
def explore_data_base(data_base):
    """This functions explores some columns in the database
    Args:
        data_base (dataframe): database
    """

    print(data_base.info())

    print(data_base['view_count'].describe())
    print(max(data_base['view_count']))

    print(data_base['likes'].describe())
    print(max(data_base['likes']))
    print(data_base[data_base['likes'] == max(data_base['likes'])])

    print(data_base['dislikes'].describe())
    print(max(data_base['dislikes']))
    print(data_base[data_base['dislikes'] == max(data_base['dislikes'])])

    print(data_base['comment_count'].describe())
    print(max(data_base['comment_count']))
    print(data_base[data_base['comment_count'] == max(data_base['comment_count'])])

    print(data_base['thumbnail_link'].describe())

    print(data_base['comments_disabled'].describe())

    print(data_base['ratings_disabled'].describe())

    print(data_base['description'].describe())
```

Cleaning Steps

The step to clean the data are:

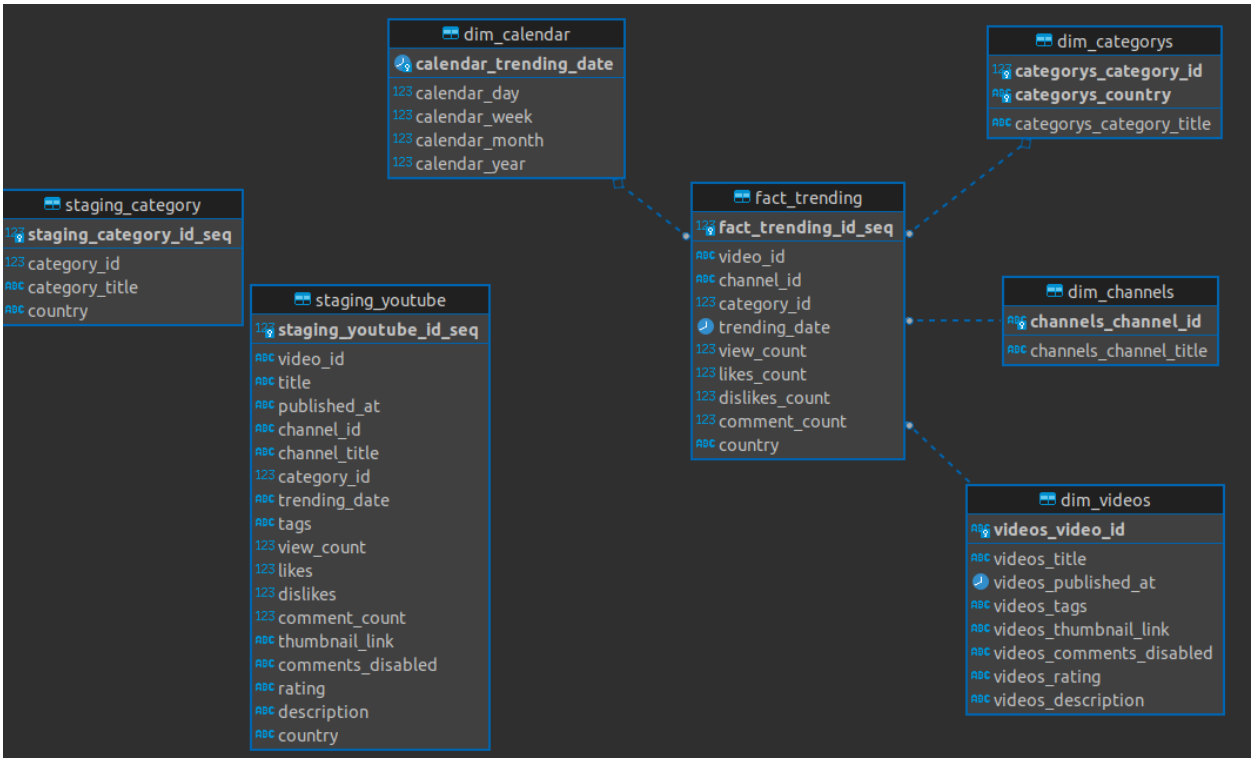
- Remove rows where ChannelTitle is nan (probably the channel has been removed)
- filter only category_id and category_title from category.json

- Get country by file name

Define the Data Model

Conceptual Data Model

I chose to create a staging area containing the two main sources of the already cleaned data and an area containing a multidimensional modeling of the data.



The multidumensional modeling was builded based on star schema. The star schema is composed of a single, central fact (trending) table that is surrounded by dimension tables (dim_categorys, dim_calendar, dim_videos, dim_channels). Among the main advantages we can mention: good response time, simplier queries, flexibility in the model, low complexity in the model.

another positive point is the possibility of enriching the data with new information. I think in the future to add more information through the youtube API

Mapping Out Data Pipelines

The Steps necessary to pipeline the data into the chosen data model are:

- Create youtube database in postgres

```
def create_database():

    """This function is responsible for creates and connects to the
       youtubedb. As also Returns the connection and cursor to
```

```
       youtubedb
    Returns:
        tuple[pg.cursor, pg.connection]: cursor and connection to
        youtubedb
    """
    #establishing the connection
    conn = pg.connect(
        database="postgres",user='postgres', password='postgres', host='localhost', port= '5432'
    )
    conn.autocommit = True
    cursor = conn.cursor()

    # create youtube database with UTF8
    cursor.execute("DROP DATABASE IF EXISTS youtubedb")
    cursor.execute("CREATE DATABASE youtubedb WITH ENCODING 'utf8'")

    conn.close()

    conn = pg.connect(
        database="youtubedb",user='postgres', password='postgres', host='localhost', port= '5432'
    )

    cursor = conn.cursor()
    conn.autocommit = True

    # close connection to default database
    return cursor, conn
```

- Drop pre-existing tables

```
def drop_tables(cur):
    """This function is responsible for dropping
    all tables in the youtubedb database
    Args:
        cur (_cursor): cursor to database session
    """
    for query in drop_table_queries:
        cur.execute(query)
```

- Create all tables

```
def create_tables(cur):
    """This function is responsible for creating all
    tables in the youtubedb
    Args:
        cur (_cursor): cursor to database session
    """
    for query in create_table_queries:
        cur.execute(query)
```

- Read and Clean the Data

```
     # get youtube trending_data
    data_base = read_trending_data("data/")
    # get category_data
    category_data = read_category_data("data/")
    # Remove nan rows of the trending_Data
    remove_nan_rows(data_base,['channelTitle'])
```

- Populating staging tables

```
# insert youtube_tranding_data in db
process_trending_data(cursor, data_base)
# insert category_data in db
process_category_data(cursor,category_data)
```

- Population Olap tables

```
# insert olap tables
insert_tables(cursor)
```

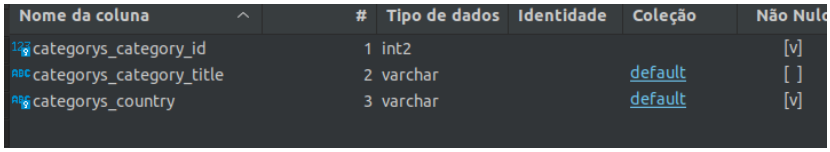
Run pipelines to Model Data

Create the data model

To build the data model I created the create_data_base.py script in which it performs the entire pipeline. this script uses as a base the queries created in the other script (sql_queries.py).

```
Python3 create_data_base.py
```

- **some samples of final data model.**
  - dim_categorys



Nome da coluna	#	Tipo de dados	Identidade	Coleção	Não Nulo
categorys_category_id	1	int2			[v]
categorys_category_title	2	varchar		default	[ ]
categorys_country	3	varchar		default	[v]

```
SELECT *
FROM dim_categorys dc
```

dim_categorys — Enter a SQL expression to filter results (use Ctrl+Space)

	categorys_category_id	categorys_category_title	categorys_country
1	43	Shows	JP
2	42	Shorts	JP
3	18	Short Movies	DE
4	26	Howto & Style	KR
5	30	Movies	MX
6	38	Foreign	CA
7	21	Videoblogging	CA
8	2	Autos & Vehicles	CA
9	15	Pets & Animals	GB
10	25	News & Politics	BR
11	24	Entertainment	US
12	35	Documentary	FR
13	1	Film & Animation	GB
14	24	Entertainment	JP
15	41	Thriller	US
16	43	Shows	BR
17	28	Science & Technology	DE
18	20	Gaming	MX
19	24	Entertainment	BR
20	23	Comedy	CA
21	38	Foreign	JP

- dim_videos

Nome da coluna	#	Tipo de dados	Identidade	Coleção
videos_video_id	1	varchar		default
videos_title	2	varchar		default
videos_published_at	3	timestamp		
videos_tags	4	varchar		default
videos_thumbnail_link	5	varchar		default
videos_comments_disabled	6	varchar		default
videos_rating	7	varchar		default
videos_description	8	varchar		default

```
SELECT *
FROM dim_videos
```

videos_video_id	videos_title	videos_published_at	videos_tags	videos_thum
001LbmXOb8A	BastiGHG du kannst stolz auf mich sein! CRAFT ATTACK 9	2021-11-10 00:00:00.000	byStegi|StegiTrash|MCExpertDE|Stegi|Minecraft|Highligh	https://i.ytim
001zHCpSP3g	Чем грозит сотрудникам ФСБ разоблачение Навальнс	2020-12-26 00:00:00.000	россия|сша|навальный|разведка|ФСБ|расследование	https://i.ytim
002eZXcY6E4	Turning Dr Mike into ME! FULL GLAM MAKEOVER	2022-03-30 00:00:00.000	manny mua|makeover|makeup transformation|dr mike|dr	https://i.ytim
002too2k4wo	IAN - MOIN MEISTER (prod. by IsyBeats & C55)	2020-10-22 00:00:00.000	TYPISCHIAN	https://i.ytim
003RHs5rrAc	Psycho Villa II Ep 1 The Family II Comedy Video II	2021-08-07 00:00:00.000	im4u|media|malayalam|anuraj and preena|malayalam wel	https://i.ytim
00b-zh318tM	НОВЫЙ МАКДАК, ВИННИ ПУХ МАНЬЯК, БАСКОВ В КИН	2022-05-31 00:00:00.000	чак|обзор|мак|винни пух|чип и дейл|гаечка|очень стр	https://i.ytim
1Iy-kUKVqkk	Among Us but you can't kill me	2020-09-17 00:00:00.000	[None]	https://i.ytim
003wpbBQSTs	Утро: Баблоян и Курников / Леонид Гозман / Михаил Х	2021-02-19 00:00:00.000	эхо москвы|онлайн смотреть|прямой эфир|echo.msk.r	https://i.ytim
006U2PlMiJc	Девушка первый раз за рулем ТЯГАЧА !!! Посетили СВЯ	2020-12-13 00:00:00.000	[None]	https://i.ytim
007lLstgd94	#shorts Poniendo a Prueba Trucos de Belleza de TikTok	2021-10-18 00:00:00.000	grwm|maquillaje|asrm|latina|makeup_mabell|tiktok|mak	https://i.ytim
00A93Z5c3uk	Lil Tecca - Money On Me (Directed by Cole Bennett)	2021-07-23 00:00:00.000	Lil Tecca|Money On Me|We Love You Tecca|WLYT2|Lyrical l	https://i.ytim
00aEQHLoKFo	РЕЙТИНГ ПТ-САУ 10 УРОВНЯ! От ХУДШЕГО до ЛУЧШЕГС	2022-04-24 00:00:00.000	Стримы Нира|Стримы с ниром|стримы нир|Near|Near,	https://i.ytim
00AEzeV6-34	КОСЯКОВобзор. ПЕРВАЯ 1/8 Высшей лиги КВН 2021 год	2021-04-20 00:00:00.000	косяков|денис косяков|косяков обзор|косякоыв обзор	https://i.ytim
00AI5xVR5SU	RB Leipzig vs. Atletico Madrid reaction: Diego Simeone's I	2020-08-13 00:00:00.000	RB leipzig|atletico madrid|rb leipzig atletico madrid|rb lei	https://i.ytim
00B86dCAjUw	Хэнсон Нива Тревел / Моя реакция / Полный ПЭ	2021-03-23 00:00:00.000	хэнсон|хенсон|90е|волга|лоурайдер|газ 24|mercedes|a	https://i.ytim
00bfb05nbiw	David Benavidez Says He 'Could Beat Canelo Alvarez Right	2021-03-14 00:00:00.000	showtime|shosports|sports|Showtime Championship Box	https://i.ytim

- dim_channels

Nome da coluna	#	Tipo de dados
channels_channel_id	1	varchar
channels_channel_title	2	varchar

```
SELECT *
FROM dim_channels
```

	channels_channel_id	channels_channel_title
1	UC002VNLHHw7kScTWawyxlWQ	bref. foot
2	UC0032Wkd3aCT4rRi1YOV3gQ	BigGucci Sosa
3	UC00B_fhCnLDA8GoB6jLjPHA	Remedios Con Saidi
4	UC00CbXSw-gXlykMA14ORYVQ	JuniorTV Life
5	UC00d6Dm1QdsTJ5VqW5aYsDg	Mateus Carrilho
6	UC00DIzrNfhYeXh9Z01fk_9w	BK'
7	UC00iFCvU8YOOzbL3RdiSTDw	GetsetflySCIENCE
8	UC00IKbzqr7BnyKZLq_XmUDA	Hersha Patel
9	UC00_LcMjTKAvTlDLNSpfZlA	UMN
10	UC00mopGCg4l0P_lhhE8ergQ	Canal da Mands
11	UC00uG71I6iPyx15EX6i_GDA	Star Wars Comics
12	UC00wbhsfrcZAEllw1Q9VRlw	Icewear Vezzo
13	UC00wMKeoF72UURd3WO9pZHw	장민호
14	UC00WS-TjaOpbembOX-ll68A	Dawson Production
15	UC014CTCdGJzeQfJtxnnMyjQ	농림축산식품부
16	UC01ly8C-KtixvgSjrrBaRIQ	PrincessNokiaVEVO
17	UC01OsvJ4CXxkVUfCdHi9rpg	친절한 경자씨의•고추농장 이야기
18	UC01ScqANYPir48jBi2zsuyw	Esporte TV Cultura

- fact_trending

Nome da coluna	#	Tipo de dados	Identidade
fact_trending_id_seq	1	serial4	
video_id	2	varchar	
channel_id	3	varchar	
category_id	4	int2	
trending_date	5	timestamp	
view_count	6	int4	
likes_count	7	int4	
dislikes_count	8	int4	
comment_count	9	int4	
country	10	varchar	

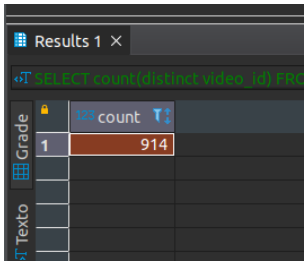
```
SELECT *
FROM fact_trending
```

fact_trending_id_seq	video_id	channel_id	category_id	trending_date	view_count	likes_count	dislikes
1	s9FH4rDMvds	UCGfBwrCoi9ZJjKiUK8MmJNw	22	2020-08-12 00:00:00.000	263.835	85.095	
2	jbGRowa5tik	UCaO6TYtlC8U5ttz62hTrZgg	10	2020-08-12 00:00:00.000	6.000.070	714.310	
3	3EfkCrXKZNs	UCoXZmVma073v5G1cW82UKkA	22	2020-08-12 00:00:00.000	2.296.748	39.761	
4	gBjox7vn3-g	UC6Xqz2pm50gDCORYztqhDpg	20	2020-08-12 00:00:00.000	300.510	46.222	
5	npoUGx7UW7o	UCEWOoncsrmirqnFqxer9lmA	23	2020-08-12 00:00:00.000	327.235	22.059	
6	Vu6PNpYKu2U	UCJVbvkrBLp7L2pnaqc5CmQQ	17	2020-08-12 00:00:00.000	117.217	14.220	
7	Iy8jXKq_9AE	UCg9nWuUISG69Hv2VaCrE72w	10	2020-08-12 00:00:00.000	93.022	7.595	
8	QAUqqcEU0Xc	UCOPS25AxMB9te9_-AHt3JEg	24	2020-08-12 00:00:00.000	1.427.499	225.365	
9	eA4FRvf6vdM	UCZD5qcen7lbLPFTjfvdLFcw	17	2020-08-12 00:00:00.000	97.711	17.153	
10	8f70QZQB4UA	UC2EWGw-KBjEReUbXMJEiaCA	24	2020-08-12 00:00:00.000	199.577	7.700	
11	oH8wiqTGKrM	UCIu-mBi1wc4Dt-WPzR0xRvA	24	2020-08-12 00:00:00.000	41.592	18.109	
12	OxwD-3E6M-k	UCwS58BcJEKW5huj_ZXESBww	10	2020-08-12 00:00:00.000	117.085	15.113	

- **One example about how user will do the query from the given schema**

Below I'm leaving a query on how to use the final model. The query below counts the **number of different videos** of the **'New & Politics'** type that appeared in the trend in the month of **July 2021**

```
SELECT count(distinct video_id)
FROM public.fact_trending ft
left join public.dim_categorys ca on (ft.category_id = ca.categorys_category_id  and ft.country = ca.categorys_country)
left join public.dim_calendar dc  on (ft.trending_date = dc.calendar_trending_date)
where  dc.calendar_month ='07' and dc.calendar_year ='2021' and ca.categorys_category_title = 'News & Politics';
```



Data Quality Checks

to ensure that the entire process went as expected, I performed the data quality process in the final data modeling. I checked the integrity of the (unique key), relationships between the tables, number of unique identifiers are the same in the fact table and in the dimension tables.

below are some queries that I used to do the tests: (FOR MORE DETAILS VISIT sql_queries.py)

- Integrity constraints

```
SELECT COUNT(videos_video_id)
FROM public.dim_videos
where videos_video_id is null ;


select count(channels_channel_id)
from public.dim_channels
where channels_channel_id is null;


select count(categorys_category_id)
from public.dim_categorys
where categorys_category_id is null;

select count(calendar_trending_date)
from dim_calendar
where calendar_trending_date is null;
```

- Source/Count checks to ensure completeness

```
select
COUNT(videos_video_id) as CONT
from public.dim_videos;

# EQUAL TO

select
COUNT( distinct video_id) as CONT
from public.fact_trending;
```

```
select
COUNT(channels_channel_id) as CONT
from public.dim_channels;

# EQUAL TO

select
COUNT( distinct channel_id) as CONT
from public.fact_trending;
```

Data dictionary

Field	Description	source
video_id	unique video identifier	dim_videos
video_title	video title	dim_videos
videos_published_at	Video publication date	dim_videos
channels_channel_id	unique channel identifier	dim_channels
channels_channel_title	channel title	dim_channels
categorys_category_id	Channel category identifier	dim_categorys
trending_date	data that was trending videos	fact_trending
videos_tags	videos tags	dim_videos
view_count	number of views on the vídeo	fact_trending
likes	number of likes on the vídeo	fact_trending
dislikes	number of dislikes on the vídeo	fact_trending
comment_count	number of comment on the vídeo	fact_trending
videos_thumbnail_link	link for thumbnail	dim_videos
videos_comments_disabled	boolean if comments is disabled	dim_videos
videos_rating	boolean for videos_rating	dim_videos
videos_description	video text description	dim_videos
country	channel country	fact_trending
categorys_category_title	channel title category	dim_categorys
calendar_day	trending day	dim_calendar
calendar_week	trending_week	dim_calendar
calendar_month	trending_month	dim_calendar
calendar_year	trending_year	dim_calendar

Complete Project Write Up

- Clearly state the rationale for the choice of tools and technologies for the project.
  Python: Python is an accessible language, Python's expansive library of open source data analysis tools, web frameworks, This versatility, along with its beginner-friendliness, has made it one of the most-used programming languages today.
  Postgres: Postgre has many features aimed to help developers build applications, administrators to protect data integrity and build fault-tolerant environments, and help you manage your data no matter how big or small the dataset. In addition to being free and open sourcePostgreSQL is highly extensible.
  Docker: Docker offers many other benefits besides this handy encapsulation, isolation, portability, and control. can use Docker to wrap up an application in such a way that its deployment and runtime issues—how to expose it on a network, how to manage its use of storage and memory and I/O, how to control access permissions—are handled outside of the application itself.
- Propose how often the data should be updated and why.
  Data will be updated once a day, because Trending helps viewers see what’s happening on YouTube and in

the world. Trending isn't personalized and displays the same list of trending videos to all viewers in the same country. Amongst the many great new videos on YouTube on any given day, Trending can only show a limited number.

- Write a description of how you would approach the problem differently under the following scenarios
  - **The data was increased by 100x:** would use apache spark because of its speed, performance and advanced analytics.If it continues to perform poorly, I would use a distributed data cluster like aws EMR
  - **The data populates a dashboard that must be updated on a daily basis by 7am every day.**

    Would Use Airflow to create a DAG and schedule interval(every day at 7am) for each DAG, which determines exactly when my pipeline is run and update dashboard.
  - **The database needed to be accessed by 100+ people.**

    Would use AWS Redshift becasuse it can handle up to 500 connections