

PROJETO FINAL

REDES SOCIAIS:

Analisando dados do

Youtube

e

LinkedIn

Elaborado por:

André Victor Moreira Costa

Eduardo Mathias

Thiago Regis

Victor de Oliveira Gonçalves

ÍNDICE

1. REQUISITOS OBRIGATÓRIOS.....	03
2. OBJETIVOS.....	04
3. MOTIVAÇÃO DO PROJETO.....	04
4. FERRAMENTAS.....	05
5. FLUXOGRAMA.....	06
6. KAGLE.....	07
7. GOOGLE CLOUD STORAGE (DATA LAKE)	07
8. ROTEIRO YOUTUBE.....	09
9. ROTEIRO LINKEDIN.....	32
10. CONTATOS.....	43

1. REQUISITOS OBRIGATÓRIOS

- Obrigatoriamente os datasets devem ter formatos diferentes (CSV / Json / Parquet / SQL / NoSQL) e 1 deles obrigatoriamente tem que ser em CSV.
- Operações com Pandas (limpezas, transformações e normalizações)
- Operações usando PySpark com a descrição de cada uma das operações.
- Operações utilizando o SparkSQL com a descrição de cada umas das operações.
- Os datasets utilizados podem ser em língua estrangeira, mas devem ao final terem seus dados/colunas exibidos na língua PT-BR.
- os datasets devem ser salvos e operados em armazenamento cloud, obrigatoriamente dentro da plataforma GCP (não pode ser usado Google drive ou armazenamento alheio ao google).
- os dados tratados devem ser armazenados também em GCP, mas obrigatoriamente em um data Lake (GStorage), DW (BigQuery) ou em ambos.
- Deve ser feito análises dentro do Big Query utilizando a linguagem padrão SQL com a descrição das consultas feitas.
- Deve ser criado no Data Studio um Dashboard simples para exibição gráfica dos dados tratados trazendo insights importantes.
- E deve ser demonstrado em um workflow simples (gráfico) as etapas de ETL.

2. OBJETIVO

A proposta deste trabalho consiste em elaborar uma ETL (Extract, Transform and Load) para bancos de dados de redes sociais.

Deverá conter nessa ETL a base do banco de dados, o repositório onde será armazenado, como os dados serão tratados e manipulados, seu destino em um Data Lake/ Data Warehouse e por fim um dashboard com insights retirados dessa base no Google Data Studio.

3. MOTIVAÇÃO DO PROJETO

Possibilitar que potenciais clientes possam entender de forma visual quais são as tendências e categorias que geram alto engajamento nas redes sociais propostas, a fim de através dessas visualizações possam tomar decisões de como agir de acordo com o cenário observando diversos cenários.

- Youtube:

Com os dados desse Dataset, o objetivo é realizar uma leitura dos canais e vídeos mais acessados, comentados e curtidos, traçando uma tendência do público ao consumo de cada tipo de vídeo separado por categoria e país. Um dos principais objetivos é analisar os vídeos mais acessados por categoria e país, para analisar qual tipo onde e qual tipo de conteúdo é mais consumido. Com essa análise também foi possível verificar qual categoria e qual país mais interage nessa rede social, fazendo uma relação de Curtida por Visualização, Descurtida por Visualização e Comentário por Visualização.

O intervalo de tempo deste Dataset é de 12/08/2020 à 17/11/2021

- LinkedIn:

Já com os dados do Dataset do LinkedIn, o objetivo é realizar uma análise de proporção referente à etnia, nacionalidade e gênero, assim como eles se apresentam na foto de perfil. Um conjunto de fotos de cada usuário foi analisada para identificar as suas emoções, como felicidade, raiva, medo, tristeza entre outros. O objetivo foi traçar um paralelo com a quantidade de seguidores e promoções que possuem

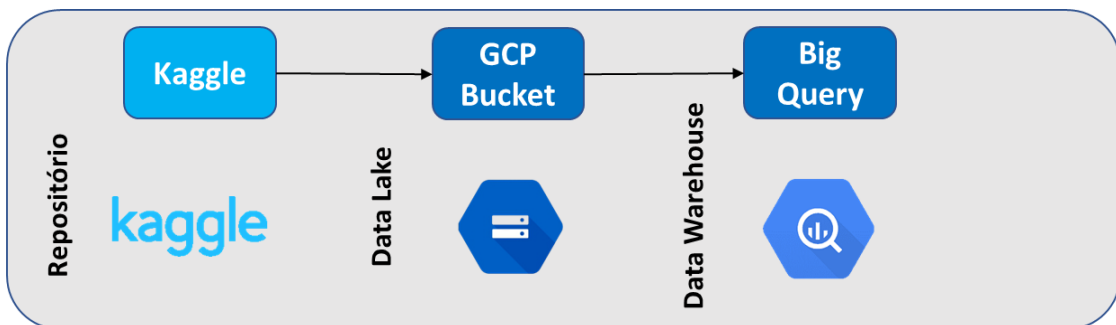
Esse Dataset foi atualizado 2 anos atrás, em 2019.

4. FERRAMENTAS

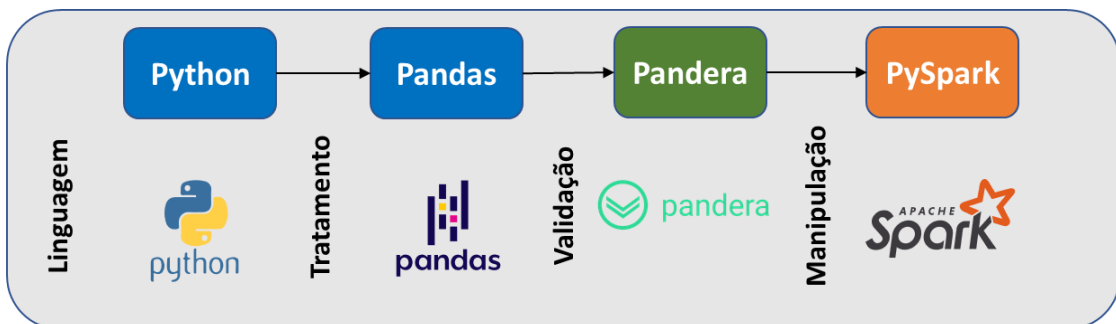
Planejamento



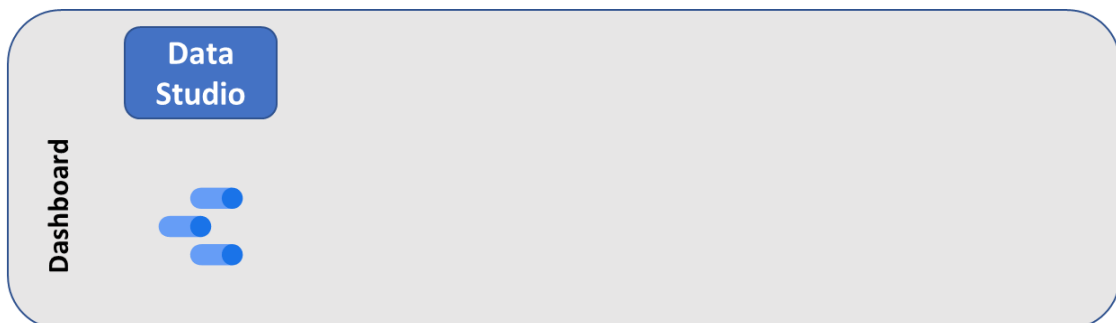
Armazenamento



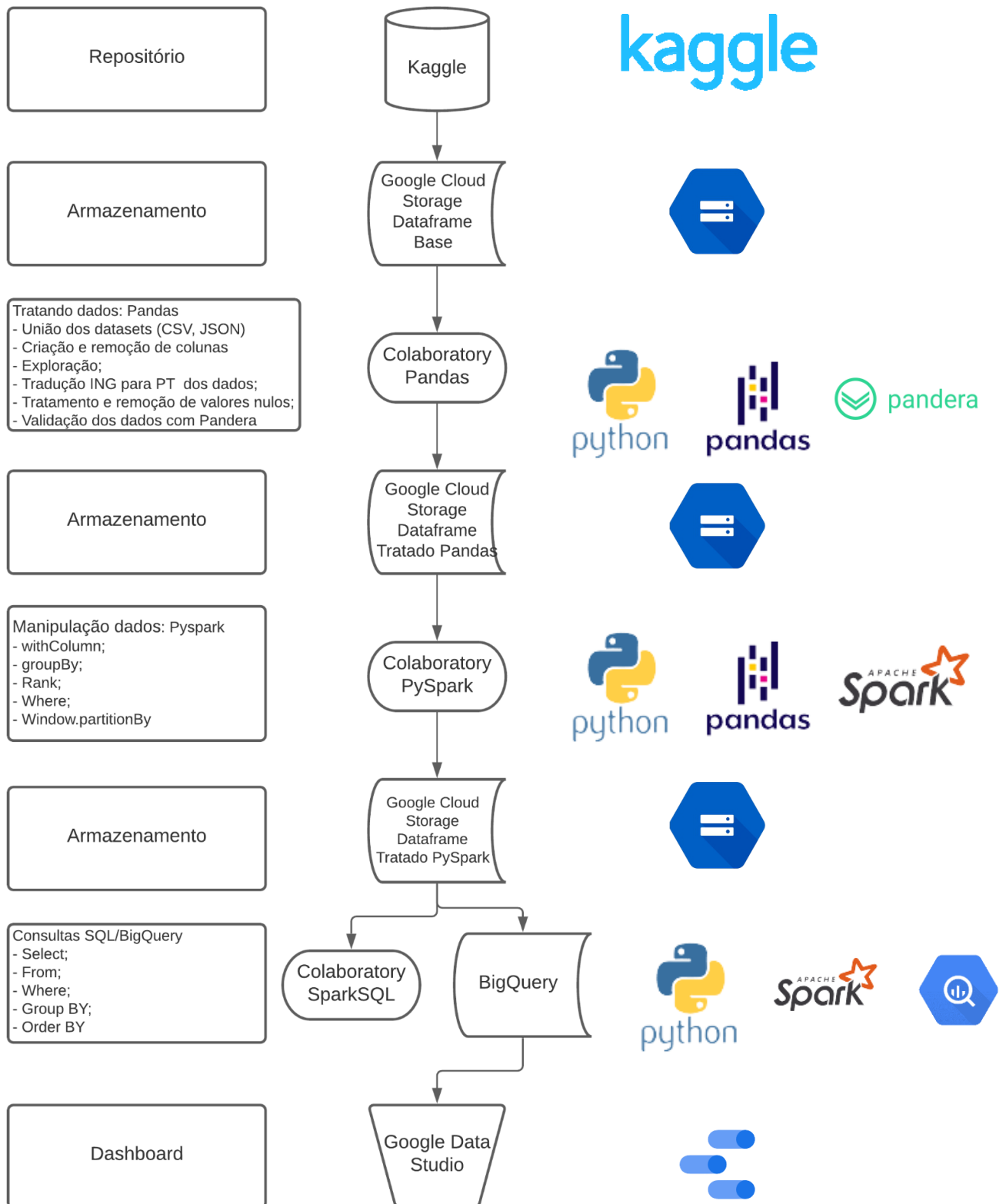
Transformação



Visualização

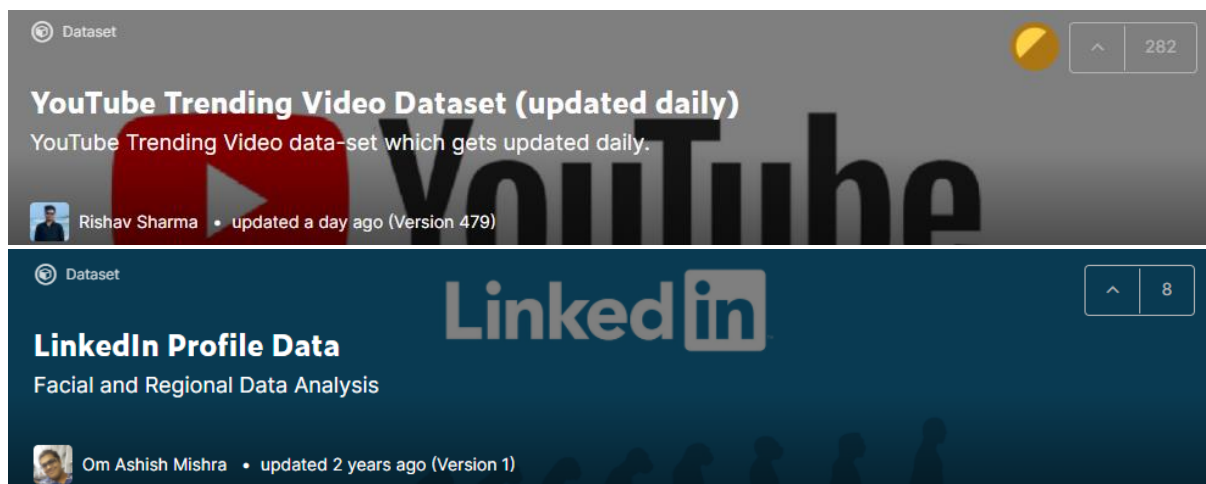


5. FLUXOGRAMA



6. KAGGLE

Kaggle é uma comunidade online voltada principalmente para cientistas de dados e praticantes de machine learning, onde os usuários podem procurar e publicar datasets de diversos assuntos. Os datasets públicas vem de fontes de usuários individuais, empresas de diversos portes e institutos de pesquisa, muitas estimulando os usuários a resolver cases da forma mais eficiente, estimulando com prêmios monetários. Devido esse ambiente propiciar bases de dados de alta qualidade, foram escolhidos dois datasets de redes sociais distintas (YouTube e LinkedIn) de acordo com o tema deste trabalho.



7. GOOGLE CLOUD STORAGE (Data Lake)

Todos os arquivos utilizados nos notebooks são armazenados no Bucket do Google Cloud Storage, para isso foram definidas, duas pastas, uma de “entrada”, onde ficam os dados retirados do Kaggle (a base de dados original) e uma pasta de “saída”, onde são armazenados os datasets resultantes das transformações feitas nos notebooks etapa por etapa.

Na pasta de “entrada”, estão os arquivos das duas bases de dados escolhidas:

- [YouTube Trending Video Dataset \(updated daily\)](#)
- [LinkedIn Profile Data](#)

Pasta de Entrada (Bucket)

Buckets > projetofinalgrupo8 > entrada

[UPLOAD FILES](#)[UPLOAD FOLDER](#)[CREATE FOLDER](#)[MANAGE HOLDS](#)[DOWNLOAD](#)[DELETE](#)

Filter by name prefix only ▼

Filter Filter objects and folders

<input type="checkbox"/>	Name	Size	Type	Created ?	Storage class
<input type="checkbox"/>	BR_category_id.json	9.9 KB	application/json	Nov 17, 2...	Standard
<input type="checkbox"/>	BR_youtube_trending_data.csv	122.1 MB	text/csv	Nov 17, 2...	Standard
<input type="checkbox"/>	CA_category_id.json	9.9 KB	application/json	Nov 17, 2...	Standard
<input type="checkbox"/>	CA_youtube_trending_data.csv	133.8 MB	text/csv	Nov 17, 2...	Standard
<input type="checkbox"/>	DE_category_id.json	9.9 KB	application/json	Nov 17, 2...	Standard
<input type="checkbox"/>	DE_youtube_trending_data.csv	149.9 MB	text/csv	Nov 17, 2...	Standard
<input type="checkbox"/>	FR_category_id.json	9.9 KB	application/json	Nov 17, 2...	Standard
<input type="checkbox"/>	FR_youtube_trending_data.csv	123.5 MB	text/csv	Nov 17, 2...	Standard

A pasta de “saída” armazena os arquivos tratados, transformados e manipulados nas etapas do projeto, cada um com um nome de acordo com a etapa que foi concluída e sendo usados cada um em um respectivo notebook do projeto, no BigQuery e também como base para o Dashboard no Google Data Studio.

Pasta de Saída (Bucket)

Buckets > projetofinalgrupo8 > saida

[UPLOAD FILES](#)[UPLOAD FOLDER](#)[CREATE FOLDER](#)[MANAGE HOLDS](#)[DOWNLOAD](#)[DELETE](#)

Filter by name prefix only ▼

Filter Filter objects and folders

<input type="checkbox"/>	Name	Size	Type	Created ?	Storage class
<input type="checkbox"/>	data_tratado_pyspark.csv	201.4 MB	text/csv	Nov 22, 2...	Standard
<input type="checkbox"/>	linkedin_tratado_pandas.csv	1.5 MB	text/csv	Nov 25, 2...	Standard
<input type="checkbox"/>	linkedin_tratado_pyspark.csv	1.2 MB	text/csv	Nov 25, 2...	Standard
<input type="checkbox"/>	youtube_data_base.csv	205.5 MB	text/csv	Nov 23, 2...	Standard
<input type="checkbox"/>	youtube_tratado_pandas.csv	204.4 MB	text/csv	Nov 23, 2...	Standard
<input type="checkbox"/>	youtube_tratado_pyspark.csv	201.4 MB	text/csv	Nov 23, 2...	Standard

8. ROTEIRO YOUTUBE

Base de dados youtube

A premissa de escolher esse dataset foi cumprir o requisito de usar arquivos de formatos diferentes (CSV e JSON) e ser um dataset interessante, onde fosse possível retirar insights. Esse banco de dados é composto por 11 “CSV’s” de países diferentes com dados dos vídeos que estiveram em destaque no período de 12/08/2020 até 17/11/2021 e também arquivos “JSON” complementares, que trabalham como dicionários das categorias dos vídeos.

```
[ ] 1 dados_br_csv = pd.read_csv('gs://projetofinalgrupo8/entrada/BR_youtube_trending_data.csv')
2 dados_ca_csv = pd.read_csv('gs://projetofinalgrupo8/entrada/CA_youtube_trending_data.csv')
3 dados_de_csv = pd.read_csv('gs://projetofinalgrupo8/entrada/DE_youtube_trending_data.csv')
4 dados_fr_csv = pd.read_csv('gs://projetofinalgrupo8/entrada/FR_youtube_trending_data.csv')
5 dados_gb_csv = pd.read_csv('gs://projetofinalgrupo8/entrada/GB_youtube_trending_data.csv')
6 dados_in_csv = pd.read_csv('gs://projetofinalgrupo8/entrada/IN_youtube_trending_data.csv')
7 dados_jp_csv = pd.read_csv('gs://projetofinalgrupo8/entrada/JP_youtube_trending_data.csv')
8 dados_kr_csv = pd.read_csv('gs://projetofinalgrupo8/entrada/KR_youtube_trending_data.csv')
9 dados_mx_csv = pd.read_csv('gs://projetofinalgrupo8/entrada/MX_youtube_trending_data.csv')
10 dados_us_csv = pd.read_csv('gs://projetofinalgrupo8/entrada/US_youtube_trending_data.csv')
11 dados_ru_csv = pd.read_csv('gs://projetofinalgrupo8/entrada/RU_youtube_trending_data.csv')
```

```
[ ] 1 dados_br_csv.shape
```

```
[ ] 1 dados_json = pd.read_json('gs://projetofinalgrupo8/entrada/BR_category_id.json')
```

1º notebook: 001 SOCIAL MEDIA COMBINACAO DE DATASETS Nivel Pandas

Neste notebook foram feitas as primeiras explorações dos dados, combinando os datasets de cada país em um só, criando uma coluna nova chamada “country” para que fosse possível identificar de qual país pertence o vídeo que entrou em destaque, usando o comando “concat” da biblioteca Pandas, assim gerando um dataset “mundo”.

O comando “concat” é uma forma de união dos datasets como se fosse um “empilhamento”, desde que as colunas sejam de mesmo nome e tipo de dados.

VERIFICANDO DIMENSÃO DE CADA PAÍS

```
1 print(f' BR: {dados_br_csv.shape},\n CA: {dados_ca_csv.shape},\n DE: {dados_de_csv.shape},\n FR: {dados_fr_csv.shape},\n GB: {dados_gb_csv.shape},\n IN: {dados_in_csv.shape},\n JP: {dados_jp_csv.shape},\n KR: {dados_kr_csv.shape},\n MX: {dados_mx_csv.shape},\n US: {dados_us_csv.shape},\n RU: {dados_ru_csv.shape}')
```

CONCATENANDO DADOS WORLD

```
[ ] 1 dataframes = [dados_br_csv, dados_ca_csv, dados_de_csv, dados_fr_csv, dados_gb_csv, dados_in_csv, dados_jp_csv, dados_kr_csv, dados_mx_csv, dados_ru_csv, dados_us_csv]\n2 dados_world_csv = pd.concat(dataframes)
```

Decidimos manter as siglas dos países que compõem os dados, são eles:

- BR = Brasil
- CA = Canadá
- DE = Deutschland - Alemanha
- FR = França
- GB = Grã-Bretanha
- IN = Índia
- JP = Japão
- KR = República da Coreia
- MX = México
- US = Estados Unidos da América
- RU = Rússia

Em seguida foi necessário “descompactar” o arquivo “JSON” para que fosse possível trazer o nome das categorias para o dataset “mundo”, através de um LOOP FOR, assim, gerando um novo dataset “base” com pouco mais de 1 milhão de linhas e 22 colunas.

```
[ ] 1 dados_json['items'][0]

{'etag': 'IfWa37JGcqZs-jZeAyFGkbeh6bc',
 'id': '1',
 'kind': 'youtube#videoCategory',
 'snippet': {'assignable': True,
 'channelId': 'UCBR8-60-B28hp2BmDPdntcQ',
 'title': 'Film & Animation'}}
```

DESCOMPRIANDO JSON

```
[ ] 1 new_dados = []
    2
    3 for i in range(len(dados_json['items'])):
    4     #print(dados_json['items'][i])
    5     new_data = {}
    6     new_data['json_kind'] = dados_json['items'][i]['kind']
    7     new_data['json_etag'] = dados_json['items'][i]['etag']
    8     new_data['categoryId'] = dados_json['items'][i]['id']
    9     new_data['json_title'] = dados_json['items'][i]['snippet']['title']
   10     new_data['json_assignable'] = dados_json['items'][i]['snippet']['assignable']
   11     new_data['json_channelId'] = dados_json['items'][i]['snippet']['channelId']
   12     new_dados.append(new_data)
```

COMBINANDO DADOS WORLD COM JSON

```
[ ] 1 data_raw = pd.merge(dados_world_csv, new_df, on=['categoryId'], how='left')
```

```
[ ] 1 data_raw.shape

(1021622, 22)
```

Optamos por remover 8 colunas que estão descritas dentro do notebook, por considerarmos que não haveriam informações interessantes dessas colunas, reduzindo para 14.

Por fim, exportamos essa primeira etapa do nosso banco de dados para o Google Cloud Storage com o nome “youtube_data_base.csv”.

Link do notebook:

<https://colab.research.google.com/drive/1WnvhPZPWYKgbGlbuDqBOdxzxD0CZp-x0?usp=sharing>

2º notebook: 002 SOCIAL MEDIA TRATAMENTO DE DADOS Nivel Pandas

Dando sequência ao projeto, foi carregado do Google Cloud Storage o “CSV” feito no final do 1º notebook.

Nesta etapa, o foco principal é o tratamento dos dados, para que possam ser usados de forma mais “limpa” em etapas posteriores.

O primeiro passo foi traduzir os termos em inglês nos nomes das colunas e dentro das colunas de categoria:

Renomeando colunas e dados Categóricos

Criando listas para renomear colunas

```
[ ] 1 col_old = ['video_id', 'title', 'publishedAt', 'channelId', 'channelTitle',
2         'trending_date', 'view_count', 'likes', 'dislikes', 'comment_count',
3         'comments_disabled', 'ratings_disabled', 'country', 'json_title']
4
5 col_new = ['id_video', 'titulo_video', 'publicado_em', 'id_canal', 'nome_canal',
6         'data_destaque', 'cont_visualizacao', 'curtidas', 'nao_curtidas', 'cont_comentarios',
7         'comentarios_desabilitados', 'curtidas_desabilitadas', 'pais', 'categoria']
```

```
[ ] 1 df2.columns = col_new
```

```
[ ] 1 df2.head(2)
```

	id_video	titulo_video	publicado_em	id_canal	nome_canal	data_destaque	cont_visualizacao	curtidas	nao_curtidas	cont_comentarios
0	s9FH4rDMvds	LEVEI UM FORA? FINGI ESTAR ABAIXANDO	2020-08-11T22:21:49Z	UCGfBwrCoi9ZjKlUK8MmJNw	Pietro Guedes	2020-08-12T00:00:00Z	263835	85095	487	4500

Em seguida foi verificado o tipo dos dados que estavam em cada coluna, através disso, pode-se notar que os dados dos campos relacionados à data, estavam em forma de texto, devido a isso, eles foram convertidos com seus devidos comandos como visto abaixo:

Antes

```
1 df2.dtypes
```

```
id_video          object
titulo_video      object
publicado_em      object
id_canal          object
nome_canal        object
data_destaque     object
cont_visualizacao int64
curtidas          int64
nao_curtidas      int64
cont_comentarios  int64
comentarios_desabilitados bool
curtidas_desabilitadas bool
pais             object
categoria         object
dtype: object
```

Depois

```
1 df2.dtypes
```

```
id_video          object
titulo_video      object
publicado_em      datetime64[ns]
id_canal          object
nome_canal        object
data_destaque     datetime64[ns]
cont_visualizacao int64
curtidas          int64
nao_curtidas      int64
cont_comentarios  int64
comentarios_desabilitados bool
curtidas_desabilitadas bool
pais             object
categoria         object
dtype: object
```

Mudança de String para Datetime

```
df2['publicado_em'] = pd.to_datetime(df2['publicado_em']).dt.strftime("%Y-%m-%d %H:%M:%S")
df2['publicado_em'] = pd.to_datetime(df2['publicado_em'])
```

```
df2['data_destaque'] = pd.to_datetime(df2['data_destaque']).dt.strftime("%Y-%m-%d")
df2['data_destaque'] = pd.to_datetime(df2['data_destaque'])
```

Logo foi a etapa de lidar com os dados nulos. Na coluna de categoria foram identificados quase 3000 linhas onde a categoria estava como dado nulo, foi decidido substituir para “outros” a fim de manter as informações.

Antes

```
1 df2.isna().sum()
```

```
id_video      0
titulo_video  0
publicado_em  0
id_canal      0
nome_canal    1
data_destaque 0
cont_visualizacao 0
curtidas      0
nao_curtidas  0
cont_comentarios 0
comentarios_desabilitados 0
curtidas_desabilitadas 0
pais          0
categoria     2951
dtype: int64
```

Depois

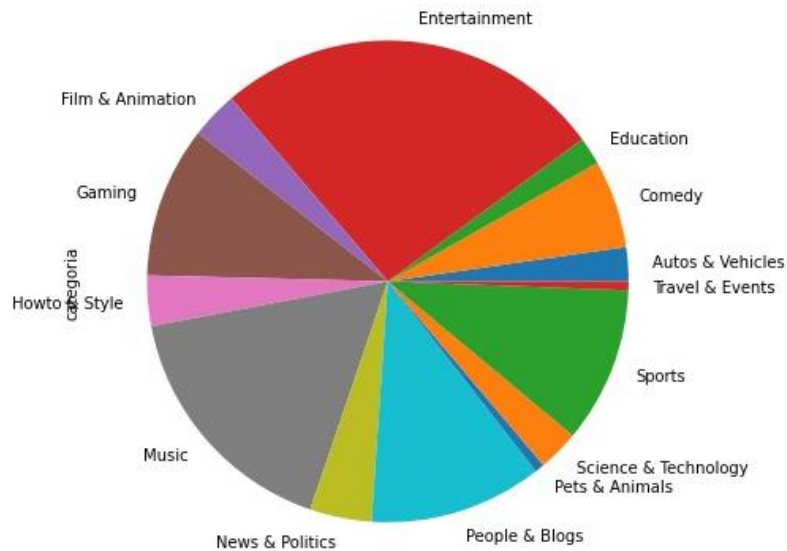
```
1 df2.isna().sum()
```

```
id_video      0
titulo_video  0
publicado_em  0
id_canal      0
nome_canal    1
data_destaque 0
cont_visualizacao 0
curtidas      0
nao_curtidas  0
cont_comentarios 0
comentarios_desabilitados 0
curtidas_desabilitadas 0
pais          0
categoria     0
dtype: int64
```

Plotagem de Categoria do vídeo

```
1 df2.groupby(["categoria"]).categoria.count().plot.pie(figsize=(12,7))
```

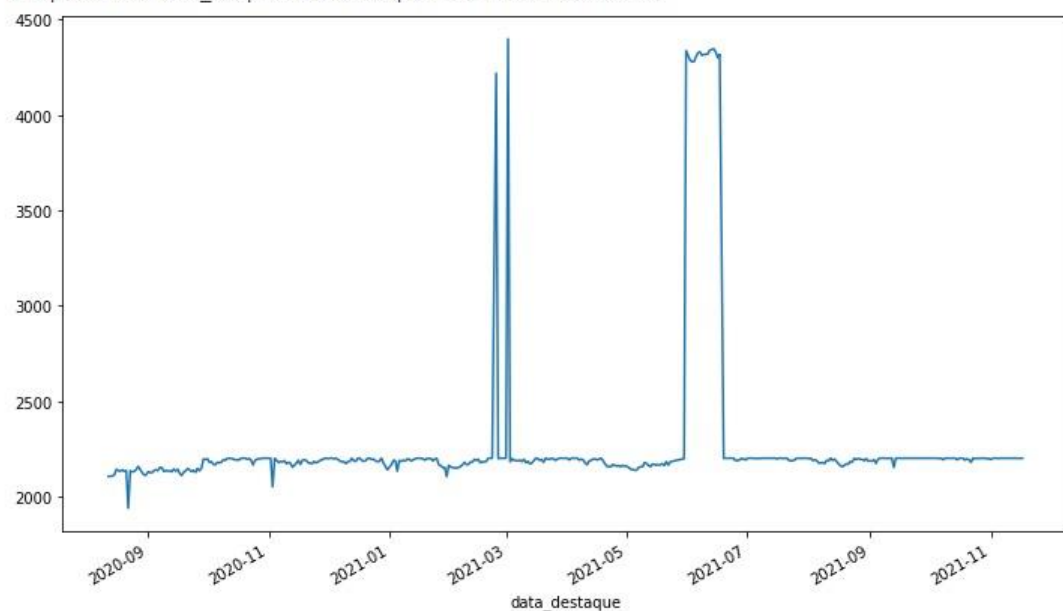
<matplotlib.axes._subplots.AxesSubplot at 0x7f22a13e9f10>



Plotagem da data que vídeo entrou em destaque

```
[24] 1 df2.groupby(["data_destaque"]).data_destaque.count().plot.line(figsize=(12,7))
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f22a0e95510>



Usamos a plotagem para analisar possíveis desvios de dados em algumas colunas.

Tratando os valores nulos da coluna "Categoria"

```
[ ] 1 categoria_antiga = ['People & Blogs', 'Music', 'Gaming', 'Comedy', 'Sports',
2      |                  'Entertainment', 'Education', 'Autos & Vehicles', 'Howto & Style',
3      |                  'News & Politics', 'Science & Technology', 'Film & Animation',
4      |                  'Travel & Events', 'Pets & Animals', np.nan]
5
6 categoria_nova = ['pessoas_e_blogs', 'musica', 'jogos', 'comedia', 'esportes',
7      |             'entretenimento', 'educacao', 'automoveis', 'como_faz_e_estilos',
8      |             'noticias_e_politicas', 'ciencias_e_tecnologia', 'filme_e_animacao',
9      |             'viagens_e_eventos', 'animais', 'outros']

[ ] 1 df2['categoria'] = df2['categoria'].replace(categoria_antiga, categoria_nova)
```

Ainda neste tópico, foi observado que apenas uma linha no dataset inteiro estava com a informação nula no coluna “nome_canal”, devido a isso neste caso foi decidido remover essa linha como mostrado abaixo.

Verificando nomes de canais com valor nulo

```
[ ] 1 df2[df2['nome_canal'].isnull()]
```

	id_video	titulo_video	publicado_em
499155	9b9MovPPewk	Kala Official Teaser Tovino Thomas Rohith ...	2021-01-21 12:30:29

Removendo linha com nome de canal nulo

```
[ ] 1 df2.drop(499155, inplace = True)
```

Seguindo a linha de raciocínio, onde os termos em inglês já foram traduzidos, o tipo dos dados foi estabelecidos e os dados nulos tratados, a fim de garantir a integridade do tipos dos dados por coluna, foi feita uma verificação dos dados com a biblioteca “Panderas” com sucesso no resultado.

```

schema = pa.DataFrameSchema(
    columns = {
        "id_video":pa.Column(pa.String),
        "titulo_video":pa.Column(pa.String),
        "publicado_em":pa.Column(pa.DateTime),
        "id_canal":pa.Column(pa.String),
        "nome_canal":pa.Column(pa.String),
        "data_destaque":pa.Column(pa.DateTime),
        "cont_visualizacao":pa.Column(pa.Int),
        "curtidas":pa.Column(pa.Int),
        "nao_curtidas":pa.Column(pa.Int),
        "cont_comentarios":pa.Column(pa.Int),
        "comentarios_desabilitados":pa.Column(pa.Bool),
        "curtidas_desabilitadas":pa.Column(pa.Bool),
        "pais":pa.Column(pa.String),
        "categoria":pa.Column(pa.String)
    }
)

```

Após todo esse tratamento os dados se apresentam da seguinte forma:

Informações gerais sobre os tipo de dados e quantidades de não nulos

```
[ ] 1 df2.info()
```

```

<class 'pandas.core.frame.DataFrame'>
Int64Index: 1021621 entries, 0 to 1021621
Data columns (total 14 columns):
#   Column                Non-Null Count  Dtype
---  -
0   id_video              1021621 non-null object
1   titulo_video          1021621 non-null object
2   publicado_em          1021621 non-null datetime64[ns]
3   id_canal              1021621 non-null object
4   nome_canal            1021621 non-null object
5   data_destaque         1021621 non-null datetime64[ns]
6   cont_visualizacao     1021621 non-null int64
7   curtidas              1021621 non-null int64
8   nao_curtidas          1021621 non-null int64
9   cont_comentarios      1021621 non-null int64
10  comentarios_desabilitados 1021621 non-null bool
11  curtidas_desabilitadas 1021621 non-null bool
12  pais                  1021621 non-null object
13  categoria              1021621 non-null object
dtypes: bool(2), datetime64[ns](2), int64(4), object(6)
memory usage: 103.3+ MB

```

Finalizando este 2º notebook salvando o arquivo como “youtube_tratado_pandas.csv” no Google Cloud Storage para dar sequência a próxima etapa.

Exportando DataFrame para o GCP

```
[ ] 1 from google.cloud import storage
    2 import os
    3 serviceAccount = '/content/projetofinalgrupo8-2dcd866c3f46.json'
    4 os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = serviceAccount
    5
    6
    7 client = storage.Client()
    8 bucket = client.get_bucket('projetofinalgrupo8')
    9
   10 bucket.blob('saida/youtube_tratado_pandas.csv').upload_from_string(df2.to_csv(index=False), 'text/csv')
```

Link notebook: https://colab.research.google.com/drive/1C8isdhWYarRfel_dlwKj-aMtJZgAGK4S?usp=sharing

3º notebook: 003 SOCIAL MEDIA PARAMETROS Nivel Pyspark

Iniciando a etapa de manipulação no PySpark, foi construído o “schema”, onde já é feita a verificação dos tipos dos dados, caso o dado seja incompatível com o tipo definido no “schema” a coluna vem com dados nulos, assim valendo como uma verificação da qualidade dos dados.

Para que fosse possível carregar o arquivo que foi tratado na etapa de Pandas, foi necessário primeiramente carregar o dataset com a biblioteca Pandas e logo na linha seguinte utilizar a criação de dataframe do PySpark para que fosse possível interpretar o dataset nesse formato como na figura abaixo:

```
customSchema = StructType([
    StructField("id_video", StringType(), True),
    StructField("titulo_video", StringType(), True),
    StructField("publicado_em", StringType(), True),
    StructField("id_canal", StringType(), True),
    StructField("nome_canal", StringType(), True),
    StructField("data_destaque", StringType(), True),
    StructField("cont_visualizacao", IntegerType(), True),
    StructField("curtidas", IntegerType(), True),
    StructField("nao_curtidas", IntegerType(), True),
    StructField("cont_comentarios", IntegerType(), True),
    StructField("comentarios_desabilitados", StringType(), True),
    StructField("curtidas_desabilitadas", StringType(), True),
    StructField("pais", StringType(), True),
    StructField("categoria", StringType(), True)
])

schema = customSchema
```

```
serviceAccount = '/content/projetofinalgrupo8-2dcd866c3f46.json'
os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = serviceAccount

client = storage.Client()
bucket = client.get_bucket('projetofinalgrupo8')

dfspark = pd.read_csv('gs://projetofinalgrupo8/saida/youtube_tratado_pandas.csv', sep=',', encoding='UTF-8', header=0)
df= spark.createDataFrame(dfspark, schema=schema)
```

Na sequência foram criadas dentro desse dataset as colunas de publicação do vídeo e a data que entrou em destaque para o formato apenas do dia em que isso ocorre e posteriormente removidas as que estavam no formato antigo, como mostrado abaixo:

Criação colunas publicação de data e data de destaque

```
[ ] 1 #CRIAR COLUNAS DE PUBLICAÇÃO DE DATA E DATA DE DESTAQUE
    2 df = df.withColumn("publicado_em_data", F.lit(F.substring("publicado_em", 1, 10))).withColumn("data_destaque1", F.lit(F.substring("data_destaque", 1, 10)))
    3 df.show(1)
```

	data_destaque1	cont_visualizacao	curtidas	nao_curtidas	cont_comentarios	comentarios_desabilitados	curtidas_desabilitadas	pais	categoria	publicado_em_data	data_destaque1
s	2020-08-12 00:00:00	263835	85095	487	4500	false	false	BR	pessoas_e_blogs	2020-08-11	2020-08-12

Remoção colunas publicado_em e data_destaque

```
1 #DROPAR COLUNA PUBLICADO_EM
2 df = df.drop('publicado_em')
3 df.show(1)
```

	data_destaque1	cont_visualizacao	curtidas	nao_curtidas	cont_comentarios	comentarios_desabilitados	curtidas_desabilitadas	pais	categoria	publicado_em_data	data_destaque1
s	2020-08-12 00:00:00	263835	85095	487	4500	false	false	BR	pessoas_e_blogs	2020-08-11	2020-08-12

Aqui algumas novas colunas são criadas com o comando “.withColumn”.

Relação curtida por visualização em porcentagem

```
[ ] 1 df = (df.withColumn("like_por_visualizacao_100", F.round((F.col("curtidas") / F.col("cont_visualizacao") * 100), 2)))
2 df.show(1)
```

t_visualizacao	curtidas	nao_curtidas	cont_comentarios	comentarios_desabilitados	curtidas_desabilitadas	pais	categoria	publicado_em_data	data_destaque	like_por_visualizacao_100
263835	85095	487	4500	false	false	BR	pessoas_e_blogs	2020-08-11	2020-08-12	32.25

Como esse é um novo notebook, usando PySpark, também foi feita a transformação das colunas de data para o tipo “date” usando o comando “.cast()”

▼ Transformando os tipos das colunas para data

```
[17] 1 df = df.withColumn("publicado_em_data", F.col("publicado_em_data").cast("date"))
2 df = df.withColumn("data_destaque", F.col("data_destaque").cast("date"))
3
4 df.select(F.col("publicado_em_data"), F.col("data_destaque")).show(5)
5 df.printSchema()
```

publicado_em_data	data_destaque
2020-08-11	2020-08-12
2020-08-11	2020-08-12
2020-08-10	2020-08-12
2020-08-11	2020-08-12
2020-08-11	2020-08-12

only showing top 5 rows

```
root
 |-- id_video: string (nullable = true)
 |-- titulo_video: string (nullable = true)
 |-- id_canal: string (nullable = true)
 |-- nome_canal: string (nullable = true)
 |-- cont_visualizacao: integer (nullable = true)
 |-- curtidas: integer (nullable = true)
 |-- nao_curtidas: integer (nullable = true)
 |-- cont_comentarios: integer (nullable = true)
 |-- comentarios_desabilitados: string (nullable = true)
 |-- curtidas_desabilitadas: string (nullable = true)
 |-- pais: string (nullable = true)
 |-- categoria: string (nullable = true)
 |-- publicado_em_data: date (nullable = true)
 |-- data_destaque: date (nullable = true)
```

A partir daqui são mostrados insights usando a combinação de códigos para chegar em um resultado, segue abaixo as consultas:

Rank de videos dentro de um canal

```
[ ] 1 #RANK DOS VÍDEOS COM MAIS CURTIDA/VISUALIZAÇÃO EM UM CANAL
2 w0 = Window.partitionBy(F.col("nome_canal")).orderBy("like_por_visualizacao_100")
3
4 (df.withColumn("RANK", F.rank().over(w0))
5 .withColumn("curtida_por_visualizacao", F.max(F.col("like_por_visualizacao_100")).over(w0))
6 .select("RANK", "curtida_por_visualizacao", "titulo_video", "nome_canal", "pais", "publicado_em_data").show(15)
7 )
```

RANK	curtida_por_visualizacao	titulo_video	nome_canal	pais	publicado_em_data
1	10.17	#Shorts ¿Es esta ...	PONGAMOSLO A PRU...	MX	2021-10-07
2	10.18	#Shorts ¿Es esta ...	PONGAMOSLO A PRU...	MX	2021-10-07
3	10.2	#Shorts ¿Es esta ...	PONGAMOSLO A PRU...	MX	2021-10-07
4	10.21	#Shorts ¿El mejor...	PONGAMOSLO A PRU...	MX	2021-11-09
5	10.22	#Shorts ¿El mejor...	PONGAMOSLO A PRU...	MX	2021-11-09

```

1 #RANK DOS VÍDEOS COM MAIS DESCURTIDA/VISUALIZAÇÃO EM UM CANAL
2 w0 = Window.partitionBy(F.col("nome_canal")).orderBy("deslike_por_visualizacao_100")
3
4 (df.withColumn("RANK",F.row_number().over(w0))
5  .withColumn("descurtida_por_visualizacao",F.max(F.col("deslike_por_visualizacao_100")).over(w0))
6  .select("RANK","descurtida_por_visualizacao",'titulo_video','nome_canal',"pais","publicado_em_data").show(15)
7 )

```

RANK	descurtida_por_visualizacao	titulo_video	nome_canal	pais	publicado_em_data
1	0.11	#Shorts, puse a p...	PONGAMOSLO A PRU...	MX	2021-08-09
2	0.11	#Shorts, puse a p...	PONGAMOSLO A PRU...	MX	2021-08-09
3	0.11	#Shorts Puse a pr...	PONGAMOSLO A PRU...	MX	2021-08-12
4	0.11	#Shorts, puse a p...	PONGAMOSLO A PRU...	MX	2021-08-09
5	0.11	#Shorts Puse a pr...	PONGAMOSLO A PRU...	MX	2021-08-12

Vídeos com mais visualizações

```

[ ] 1 df.groupBy(F.col("titulo_video")).agg(
2     F.max("cont_visualizacao").alias("total_visualizacao"),
3     F.max("publicado_em_data").alias("publicado_em_data"),
4     F.max("data_destaque").alias("data_destaque")).orderBy('total_visualizacao', ascending=False).show(10)
5

```

titulo_video	total_visualizacao	publicado_em_data	data_destaque
BTS (방탄소년단) 'Butt...	296314174	2021-05-21	2021-06-04
BTS (방탄소년단) 'Dyna...	262319276	2020-08-21	2020-08-31
Turn into orbeez ...	206202284	2021-07-03	2021-08-08
Filhaal2 Mohabbat...	202091414	2021-07-06	2021-07-16
BTS (방탄소년단) 'Perm...	194795844	2021-07-09	2021-07-24
JETSKI WAX PRANK!...	194625542	2021-07-04	2021-07-15
LISA - 'LALISA' M/V	192376395	2021-09-10	2021-09-22
BLACKPINK - 'Ice ...	184778248	2020-08-28	2020-09-05
INSANE strawberry...	164362471	2021-05-08	2021-05-19
BTS (방탄소년단) 'Life...	161912058	2020-11-20	2020-12-01

only showing top 10 rows

Canais com maior soma de visualizações

```
1 df.groupby(F.col("nome_canal")).agg(
2     F.sum("cont_visualizacao").alias("soma_visualizacao"),
3     F.round(F.avg("cont_visualizacao"),2).alias("media_visualizacao"),
4     F.sum("curtidas").alias("soma_curtidas"),
5     F.round(F.avg("curtidas"),2).alias("media_curtidas"),
6     F.sum("nao_curtidas").alias("soma_naocurtidas"),
7     F.max("publicado_em_data").alias("publicado_em_data"),
8     F.max("data_destaque").alias("data_destaque"),
9     F.round(F.avg("nao_curtidas"),2).alias("media_naocurtidas")).orderBy('soma_visualizacao', ascending=False).show(10)
10
```

nome_canal	soma_visualizacao	media_visualizacao	soma_curtidas	media_curtidas	soma_naocurtidas	publicado_em_data	data_destaque	media_naocurtidas
BLACKPINK	58230237271	2.82396883E7	5596684546	2714202.01	117843253	2021-10-28	2021-11-01	57149.98
HYBE LABELS	47383301434	3.197253808E7	4954981634	3343442.4	30903181	2021-10-27	2021-11-04	20852.35
SMTOWN	46008908926	2.024149095E7	2845074443	1251682.55	40529445	2021-10-25	2021-11-06	17830.82
JYP Entertainment	38480556253	1.798156834E7	2296325821	1073049.45	36878750	2021-11-12	2021-11-17	17233.06
Big Hit Labels	37893503610	3.558075456E7	3892442538	3654875.62	79900146	2021-03-25	2021-03-30	75023.61
BANGTANTV	34712821173	1.372590794E7	5500913381	2175133.8	31245017	2021-11-14	2021-11-17	12354.69
MrBeast	31370195618	2.45847928E7	2179495546	1708068.61	22562155	2021-10-29	2021-11-05	17681.94
T-Series	14571958656	2.396703726E7	464864916	764580.45	22596720	2021-11-10	2021-11-16	37165.66
BillieEilishVEVO	13294295105	1.243619748E7	1128743521	1055887.3	24037626	2021-10-21	2021-10-28	22486.09
Fortnite Fun TV	12162966626	6.435432077E7	407459849	2155872.22	12782233	2021-07-11	2021-07-13	67630.86

only showing top 10 rows

Vídeos com mais de 50% de curtidas em relação às visualizações

```
1 df.where((F.col("pais") == 'BR')).filter("like_por_visualizacao_100 > 50").show(20)
```

titulo_video	id_canal	nome_canal	cont_visualizacao	curtidas	nao_curtidas	cont_comentarios
Anitta - Girl Fro... UCqjyPUghDSSKFBA...	Anitta	544091	325644	3577	75510	

Finalizando a etapa de PySpark, chega a hora de salvar o dataset manipulado no Google Cloud Storage.

O arquivo foi salvo com o nome “youtube_tratado_pyspark.csv”.

Salvando no Bucket Final

```
[ ] 1 serviceAccount = '/content/projetofinalgrupo8-2dcd866c3f46.json'
2 os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = serviceAccount
3
4 client = storage.Client()
5 bucket = client.get_bucket('projetofinalgrupo8')
6 bucket.blob('saida/youtube_tratado_pyspark.csv').upload_from_string(df_pandas_def.to_csv(index=False), 'text/csv')
```

Link notebook: <https://colab.research.google.com/drive/1nitGJWjnnipgUPr-m8hntc1VHwyXvaN1?usp=sharing>

4º Notebook: 004 SOCIAL MEDIA CONSULTAS Nivel SparkSQL

Neste notebook foram feitas as consultas SQL através do SparkSQL, estas mesmas consultas estão sendo usadas no BigQuery (Data Warehouse) do projeto logo após este tópico, segue abaixo algumas amostras logo abaixo de todas as consultas.

00.Consulta geral da tabela:

```
1 spark.sql("SELECT * FROM leitura_sql LIMIT 10").show()
```

id_video	titulo_video	id_canal	nome_canal	cont_visualizacao	curtidas	nao_curtidas	cont_comentarios
s9FH4rDMvds	LEVEI UM FORA? FI...	UCGfBwrCoI9ZJjKiU...	Pietro Guedes	263835	85095	487	4500
jbGRowa5tIk	ITZY "Not Shy" M/...	UCa06TYtLC8U5ttz6...	JYP Entertainment	6000070	714310	15176	31040
3EfKCrXKZns	Oh Juliana PARÓDI...	UCoXZmVma073v5G1c...	As Irmãs Mota	2296748	39761	5484	0
gBjox7vn3-g	Contos de Runeter...	UC6Xqz2pm50gDCORY...	League of Legends BR	300510	46222	242	2748
npouGx7UW7o	Entrevista com Th...	UCEW0oncsrmirqnFq...	The Noite com Dan...	327235	22059	3972	2751
Vu6PNpyKu2U	DICAS DA RODADA 2...	UCJVBvkrBlp7L2pna...	Cartoleiros Gazet...	117217	14220	106	785
ly8jXKq_9AE	LIVE PLAYLIST DA ...	UCg9nWuUISG69Hv2V...	Tayara Andreza	93022	7595	166	136
QAUqcEU0Xc	PEDI ELA EM NAMOR...	UCOPS25AxMB9te9-...	PEIXE	1427499	225365	2287	9647
eA4FRvf6vdM	AO VIVO - Apresen...	UCZD5qcen7lBlPFTj...	Vasco TV	97711	17153	65	226
8f70QZQB4UA	MASTERCHEF BRASIL...	UC2EWGw-KBJEReUbX...	MasterChef Brasil	199577	7700	129	874

01.CONSULTA VIDEOS COM MAIS DE 50% DE CURTIDAS EM RELAÇÃO ÀS VISUALIZAÇÕES

```
1 spark.sql("SELECT * FROM leitura_sql WHERE pais = 'BR' AND like_por_visualizacao_100 > 50 AND cont_visualizacao > 0").show()
```

id_video	titulo_video	id_canal	nome_canal	cont_visualizacao	curtidas	nao_curtidas	cont_comentarios	comentarios_desabilitados
CuyTC8FLICY	Anitta - Girl Fro...	UCqjjyPUghDSSKFBA...	Anitta	544091	325644	3577	75510	false

02.Consulta vídeos da categoria filmes e animações nos EUA com sua visualizações e percentual de curtidas por visualização

```
1 spark.sql("SELECT titulo_video, MAX(cont_visualizacao) AS total_visualizacao_por_video, \
2 | ROUND((MAX(curtidas) / MAX(cont_visualizacao))*100, 2) AS curtidas_cada_100_visualizacao \
3 | FROM leitura_sql \
4 | WHERE (categoria = 'filme_e_animacao' AND pais = 'US') \
5 | GROUP BY titulo_video \
6 | ORDER BY total_visualizacao_por_video DESC \
7 | LIMIT 10").show()
```

titulo_video	total_visualizacao_por_video	curtidas_cada_100_visualizacao
THE BATMAN - Main...	31990632	3.09
F9 - Official Tra...	23430400	0.94
Friday Night Funk...	21184728	2.5
Did you know that...	18390451	6.06
David Blaine Asce...	17212348	2.47
Friends: The Reun...	16143739	2.2
Parkour - Animati...	16031212	3.13
THE CONJURING: TH...	15396299	1.53
Wonder Woman 1984...	14810365	1.86
Peacemaker Offi...	14510945	0.63

03. CONSULTA VÍDEOS QUE TENHAM O NOME DO BTS COM MAIS VISUALIZAÇÕES

```

1 spark.sql("SELECT titulo_video, MAX(cont_visualizacao) AS visualizacoes \
2           FROM leitura_sql \
3           WHERE (titulo_video LIKE '%BTS%') \
4           GROUP BY titulo_video \
5           ORDER BY visualizacoes DESC \
6           LIMIT 5").show()

```

titulo_video	visualizacoes
BTS (방탄소년단) 'Butt...	296314174
BTS (방탄소년단) 'Dyna...	262319276
BTS (방탄소년단) 'Perm...	194795844
BTS (방탄소년단) 'Life...	161912058
BTS (방탄소년단) 'Film...	78893765

04. CONSULTA VÍDEOS QUE TENHAM O NOME DO NEYMAR COM MAIS VISUALIZAÇÕES

```

1 spark.sql("SELECT titulo_video, MAX(cont_visualizacao) AS visualizacoes \
2           FROM leitura_sql \
3           WHERE (titulo_video LIKE '%NEYMAR%') \
4           GROUP BY titulo_video \
5           ORDER BY visualizacoes DESC \
6           LIMIT 10").show()

```

titulo_video	visualizacoes
SHOW DE NEYMAR! B...	5761326
NEYMAR MARCA DUAS...	4464807
PAQUETÁ MARCA, MA...	3878196
NEYMAR TÁ ON!!!...	3847530
NEYMAR DÁ ASSISTÊ...	2372773
NEYMAR JOGA MUITO...	2244898
VAI TER NEYMAR ...	2168694
DEU PSG!!! NEYM...	2122704
A CHEGADA SENSACI...	2078345
New *NEYMAR JR* U...	1942078

05.Consulta de vídeos com mais de 10 milhões de visualizações da categoria esportes do youtube Brasil

```
1 spark.sql("SELECT titulo_video, MAX(cont_visualizacao) AS visualizacoes \
2         FROM leitura_sql \
3         WHERE (cont_visualizacao > 10000000 AND categoria = 'esportes' AND pais = 'BR') \
4         GROUP BY titulo_video \
5         ORDER BY visualizacoes DESC \
6         LIMIT 10").show()
```

titulo_video	visualizacoes
The Weeknd's FULL...	27865785
#LEOMESSI: First ...	26294018
LEO MESSI, D...	20648273
Khabib Nurmagomed...	19819443
Amazing moment of...	10326487

06.CONSULTA VIDEOS ANITTA COM NÚMERO DE VISUALIZAÇÕES, CURTIDAS E COMENTÁRIOS PUBLICADOS DEPOIS DE 2021-06-30

```
1 spark.sql("SELECT titulo_video, MAX(cont_visualizacao) AS num_visualizacao, MAX(curtidas) AS num_curtidas, MAX(cont_comentarios) AS num_comentario:
2         FROM leitura_sql \
3         WHERE nome_canal = 'Anitta' AND publicado_em_data > '2021-06-30' \
4         GROUP BY titulo_video \
5         ORDER BY num_visualizacao DESC \
6         LIMIT 10").show()
```

titulo_video	num_visualizacao	num_curtidas	num_comentarios
Anitta - Faking L...	6890459	347663	45614
Anitta - Envolver...	5087138	363785	36343
Anitta - Faking L...	423824	85881	14683
Anitta Sings Girl...	202367	32648	2658

07.CONSULTA VIDEOS ANITTA COM NÚMERO DE VISUALIZAÇÕES, CURTIDAS E COMENTÁRIOS

```
1 spark.sql("SELECT titulo_video, MAX(cont_visualizacao) AS num_visualizacao, MAX(curtidas) AS num_curtidas, MAX(cont_comentarios) AS num_comentario:
2         FROM leitura_sql \
3         WHERE nome_canal = 'Anitta' \
4         GROUP BY titulo_video \
5         ORDER BY num_visualizacao DESC \
6         LIMIT 10").show()
```

titulo_video	num_visualizacao	num_curtidas	num_comentarios
Anitta Me Gusta (...)	26165923	1257868	348858
Anitta - Girl Fro...	15771509	1035748	150062
Anitta - Loco (Of...	7982621	310779	45966
Anitta - Faking L...	6890459	347663	45614
Anitta - Girl Fro...	5264455	156690	10821
Anitta - Envolver...	5087138	363785	36343
Anitta Me Gusta (...)	1448503	149877	11669
Anitta Performanc...	1373409	113345	9718

08. CONSULTA DE TOTAL DE VISUALIZAÇÕES E NUMERO DE VIDEOS DOS CANAIS DA CATEGORIA DE JOGOS

```
1 spark.sql("SELECT nome_canal, COUNT(DISTINCT titulo_video) AS numero_videos, SUM(DISTINCT cont_visualizacao) AS total_visualizacao_por_canal \
2 FROM leitura_sql \
3 WHERE categoria = 'jogos' \
4 GROUP BY nome_canal \
5 ORDER BY total_visualizacao_por_canal DESC \
6 LIMIT 10").show()
```

nome_canal	numero_videos	total_visualizacao_por_canal
MrBeast Gaming	61	4258737435
Brawl Stars	30	2667075201
SSundee	100	2090914442
Clash of Clans	40	1906048141
Dream	11	1561489175
AnthonySenpai	11	1262361580
Free Fire India O...	48	1249355714
Klem Family	7	1228945428
League of Legends	36	1185810637
TommyInnit	32	1091651516

09. Consulta número de canais e média de visualizações por vídeo da categoria

```
1 spark.sql("SELECT categoria, COUNT(DISTINCT nome_canal) AS numero_canais, ROUND(MEAN(cont_visualizacao), 2) AS media_visualizacao_por_video \
2 FROM leitura_sql \
3 GROUP BY categoria \
4 ORDER BY media_visualizacao_por_video DESC").show()
```

categoria	numero_canais	media_visualizacao_por_video
musica	5786	4239143.85
ciencias_e_tecnol...	784	2108160.57
jogos	3469	1749334.19
entretenimento	7338	1710892.29
comedia	1503	1571663.04
peessoas_e_blogs	5134	1359910.64
filme_e_animacao	1301	1308026.47
educacao	735	1266638.11
esportes	2008	1187144.65
como_faz_e_estilos	1292	878036.94
outros	137	835704.84
animais	286	793950.2
noticias_e_politicas	1532	781123.55
viagens_e_eventos	284	679063.8
automoveis	831	612652.73

10. CONSULTA VIDEOS COM MENOR NÚMERO DE DESCURTIDAS MAIORES QUE ZERO NA CATEGORIA MÚSICA NOS EUA, MOSTRANDO NÚMERO DE VISUALIZAÇÕES

```
1 spark.sql("SELECT titulo_video, MIN(nao_curtidas) AS menores_nao_curtidas, MAX(cont_visualizacao) AS numero_visualizacao \
2 FROM leitura_sql \
3 WHERE (categoria = 'musica' AND pais = 'US' AND nao_curtidas > 0) \
4 GROUP BY titulo_video \
5 ORDER BY menores_nao_curtidas \
6 LIMIT 10").show()
```

titulo_video	menores_nao_curtidas	numero_visualizacao
Opry Livestream - ...	7	56447
Morgan Wallen - S...	9	580319
Beautiful Trip	24	679656
Sleepy Hallow - 1...	28	525397
Carole King - So ...	30	527981
Phish Dinner And ...	34	127299
Sammy Hagar Remem...	36	351147
Eddie Van Halen's...	37	595557
Sammy Hagar Made ...	37	207672
Morgan Wallen - D...	39	259746

11. CONSULTA VIDEOS DO CANAL DO FELIPE NETO COM MAIOR NÚMERO DE VISUALIZAÇÕES

```
1 spark.sql("SELECT titulo_video, MAX(cont_visualizacao) AS numero_visualizacao \
2         FROM leitura_sql \
3         WHERE nome_canal = 'Felipe Neto' \
4         GROUP BY titulo_video \
5         ORDER BY numero_visualizacao DESC \
6         LIMIT 10").show()
```

titulo_video	numero_visualizacao
SHOW DA BLACK FRI...	5533110
AMONG US COM NETO...	5178230
AMONG US / NETOLA...	4894976
LIVE / AMONG US c...	4292845
AMONG US COM NAVA...	3943216
ENTREI NO ROUND 6!!!	3922688
LIVE - AMONG US -...	3675967
NOVA BATALHA! SIR...	3465543
LIVE DE 40 MILHÔE...	3192567
FIZ O PIOR MILKSH...	3148609

12. CONSULTA CANAIS BRASILEIROS COM MAIOR NÚMERO DE VÍDEOS PUBLICADOS

```
1 spark.sql("SELECT nome_canal, COUNT(DISTINCT titulo_video) AS numero_videos_destaque \
2         FROM leitura_sql \
3         WHERE pais = 'BR' \
4         GROUP BY nome_canal \
5         ORDER BY numero_videos_destaque DESC \
6         LIMIT 10").show()
```

nome_canal	numero_videos_destaque
ge	322
A Fazenda	207
Felipe Neto	166
De Placa	142
MasterChef Brasil	94
WebTVBrasileira	86
FutParódias	86
Invento na Hora	76
Free Fire - Brasil	69
AM3N1C	66

13. CONSULTA NÚMERO DE CANAIS POR PAÍS E SUA MÉDIA DE COMENTÁRIOS POR VÍDEO

```
1 spark.sql("SELECT pais, COUNT(DISTINCT nome_canal) AS numero_canais, ROUND(MEAN(cont_comentarios), 2) AS media_comentarios_por_video \
2 FROM leitura_sql \
3 GROUP BY pais \
4 ORDER BY numero_canais DESC").show()
```

pais	numero_canais	media_comentarios_por_video
RU	5345	5501.33
DE	5015	8761.68
CA	4923	13011.36
US	4547	13990.23
GB	4398	11287.87
FR	3636	7439.54
BR	3317	10544.9
MX	3078	13719.64
IN	2799	13110.12
JP	2714	7991.05
KR	2600	10752.21

14. CONSULTA VIDEOS COM MAIS VISUALIZAÇÕES

```
1 spark.sql("SELECT titulo_video, MAX(cont_visualizacao) AS total_visualizacao \
2 FROM leitura_sql \
3 GROUP BY titulo_video \
4 ORDER BY total_visualizacao DESC").show()
```

titulo_video	total_visualizacao
BTS (방탄소년단) 'Butt...	296314174
BTS (방탄소년단) 'Dyna...	262319276
Turn into orbeez ...	206202284
Filhaal2 Mohabbat...	202091414
BTS (방탄소년단) 'Perm...	194795844
JETSKI WAX PRANK!...	194625542
LISA - 'LALISA' M/V	192376395
BLACKPINK - 'Ice ...	184778248
INSANE strawberry...	164362471
BTS (방탄소년단) 'Life...	161912058
BLACKPINK - 'Love...	161416953
KGF Chapter2 TEAS...	156704924
Beach Money Ball!...	153462028
Paytm IPL 2021 Ad...	141191928
Adele - Easy On M...	139547582

Link do notebook:

<https://colab.research.google.com/drive/10s6seXWQckPZvza7XJozd80KutQ7jvt9?usp=sharing>

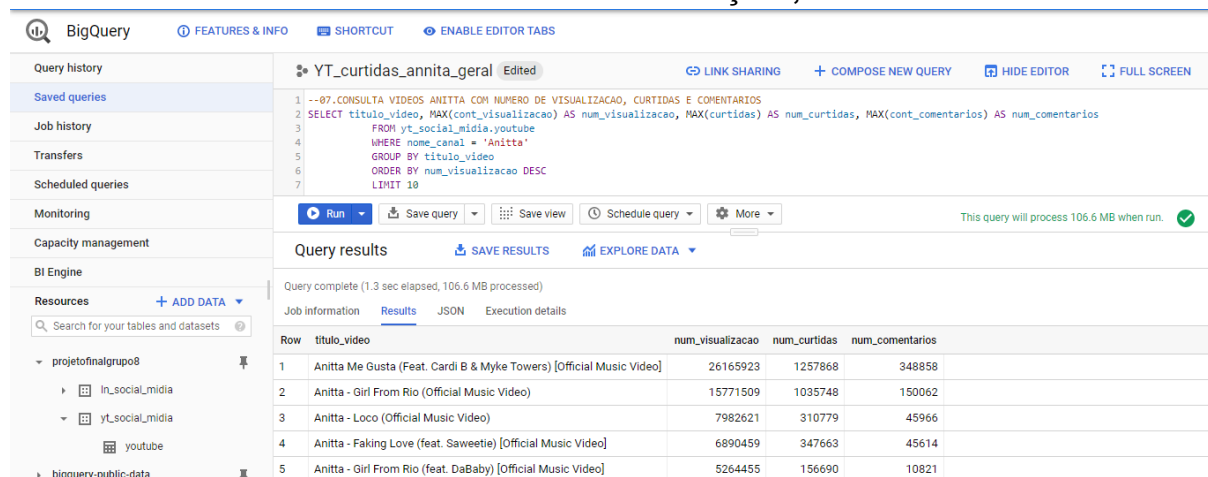
BIGQUERY (DATA WAREHOUSE) - YOUTUBE

Data Warehouse é onde os dados estão dispostos de forma mais organizada e prontos para consulta no Google Cloud.

Os datasets usados nessa etapa podem vir via upload externo, mas nesse caso, foi migrado diretamente do Google Cloud Storage, onde todas as etapas dos bancos de dados usados, estão armazenados, e nesse ponto sendo usado para consultar através da linguagem SQL dentro do BigQuery.

Logo abaixo será mostrado algumas amostras, de como são as consultas salvas nesse Data Warehouse. Elas estão seguindo a mesma lógica das consultas do sparkSQL.

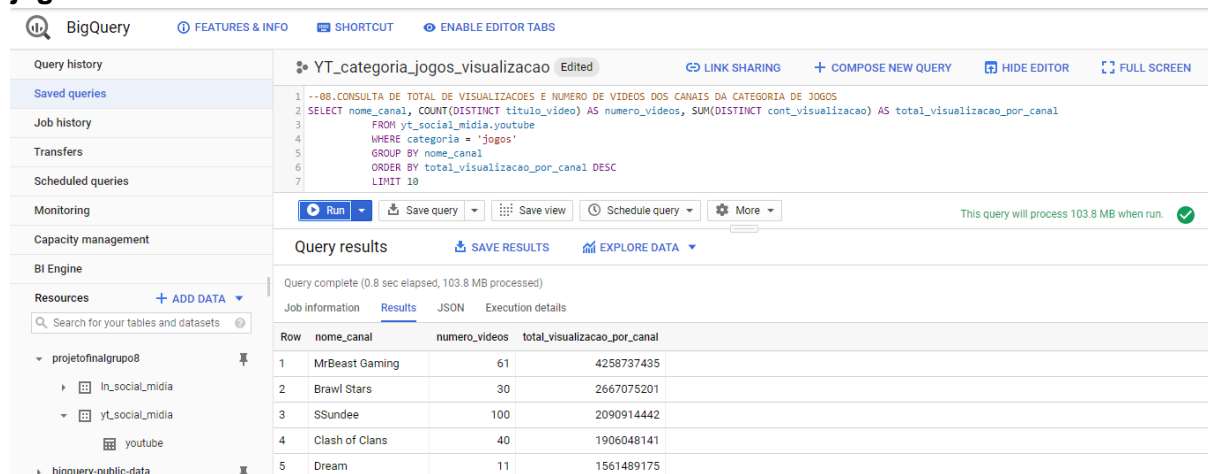
07.Consulta videos Anitta com número de visualizações, curtidas e comentários



The screenshot shows the BigQuery web interface. On the left is a sidebar with navigation options like 'Query history', 'Saved queries', 'Job history', etc. The main area displays a SQL query titled 'YT_curtidas_annita_geral'. The query selects video titles, view counts, likes, and comments for the channel 'Anitta'. Below the query editor, the 'Query results' section shows a table with 5 rows of data.

Row	titulo_video	num_visualizacao	num_curtidas	num_comentarios
1	Anitta Me Gusta (Feat. Cardi B & Myke Towers) [Official Music Video]	26165923	1257868	348858
2	Anitta - Girl From Rio (Official Music Video)	15771509	1035748	150062
3	Anitta - Loco (Official Music Video)	7982621	310779	45966
4	Anitta - Faking Love (feat. Saweetie) [Official Music Video]	6890459	347663	45614
5	Anitta - Girl From Rio (feat. DaBaby) [Official Music Video]	5264455	156690	10821

08.Consulta de total de visualizações e número de vídeos dos canais da categoria de jogos



The screenshot shows the BigQuery web interface with a query titled 'YT_categoria_jogos_visualizacao'. The query selects channel names, the number of videos, and the total view count for the category 'jogos'. The 'Query results' section displays a table with 5 rows of data.

Row	nome_canal	numero_videos	total_visualizacao_por_canal
1	MrBeast Gaming	61	4258737435
2	Brawl Stars	30	2667075201
3	SSundee	100	2090914442
4	Clash of Clans	40	1906048141
5	Dream	11	1561489175

12.Consulta canais brasileiros com maior número de vídeos publicados

BigQuery FEATURES & INFO SHORTCUT ENABLE EDITOR TABS

Query history

YT_canais_BR_com_maior_n_video_publica... Edited LINK SHARING + COMPOSE NEW QUERY HIDE EDITOR FULL SCREEN

```
--12.CONSULTA CANAIS BRASILEIROS COM MAIOR NÚMERO DE VÍDEOS PUBLICADOS
2 SELECT nome_canal, COUNT(DISTINCT titulo_video) AS numero_videos_destaque
3 FROM yt_social_midia.youtube
4 WHERE pais = 'BR'
5 GROUP BY nome_canal
6 ORDER BY numero_videos_destaque DESC
7 LIMIT 10
```

Run Save query Save view Schedule query More

This query will process 87.1 MB when run. ✓

Query results SAVE RESULTS EXPLORE DATA

Query complete (0.5 sec elapsed, 87.1 MB processed)

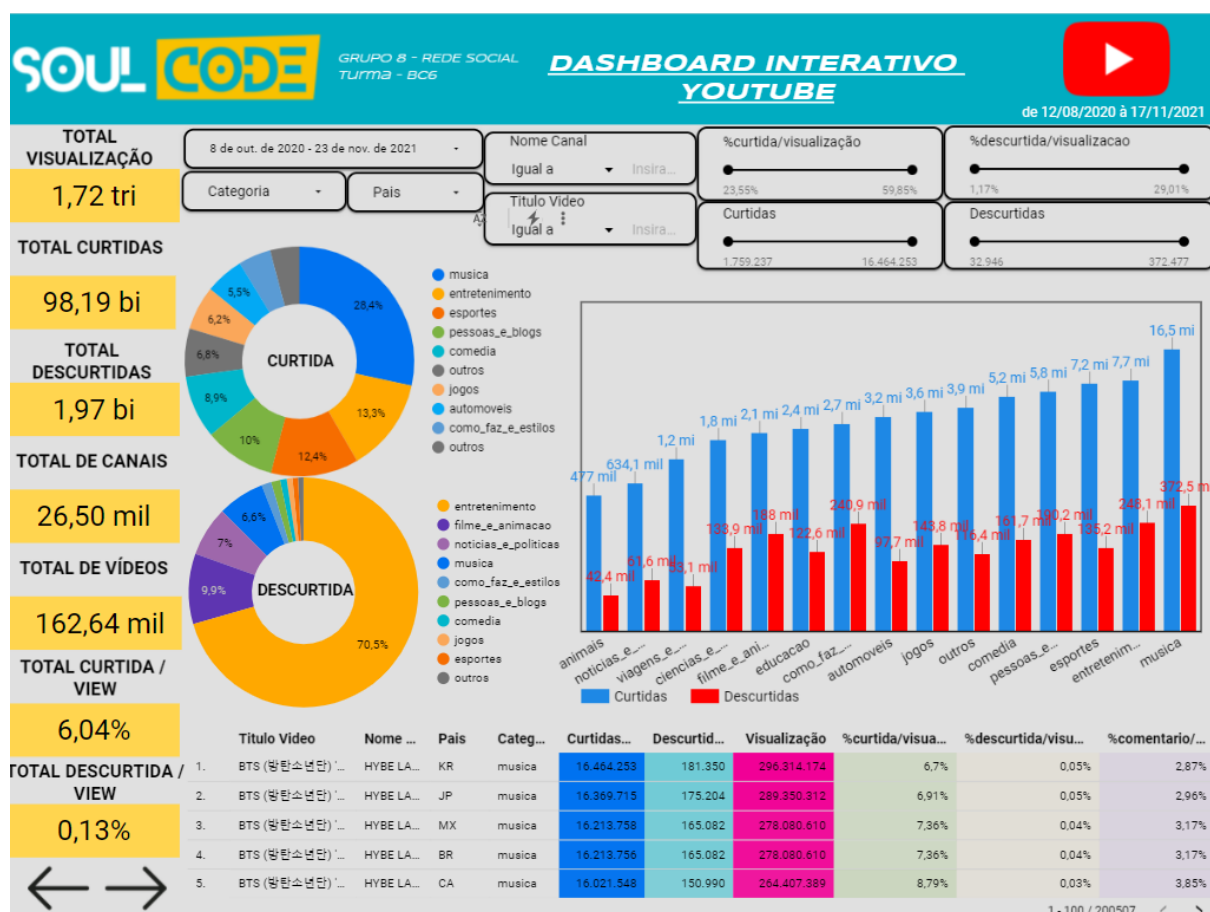
Job Information Results JSON Execution details

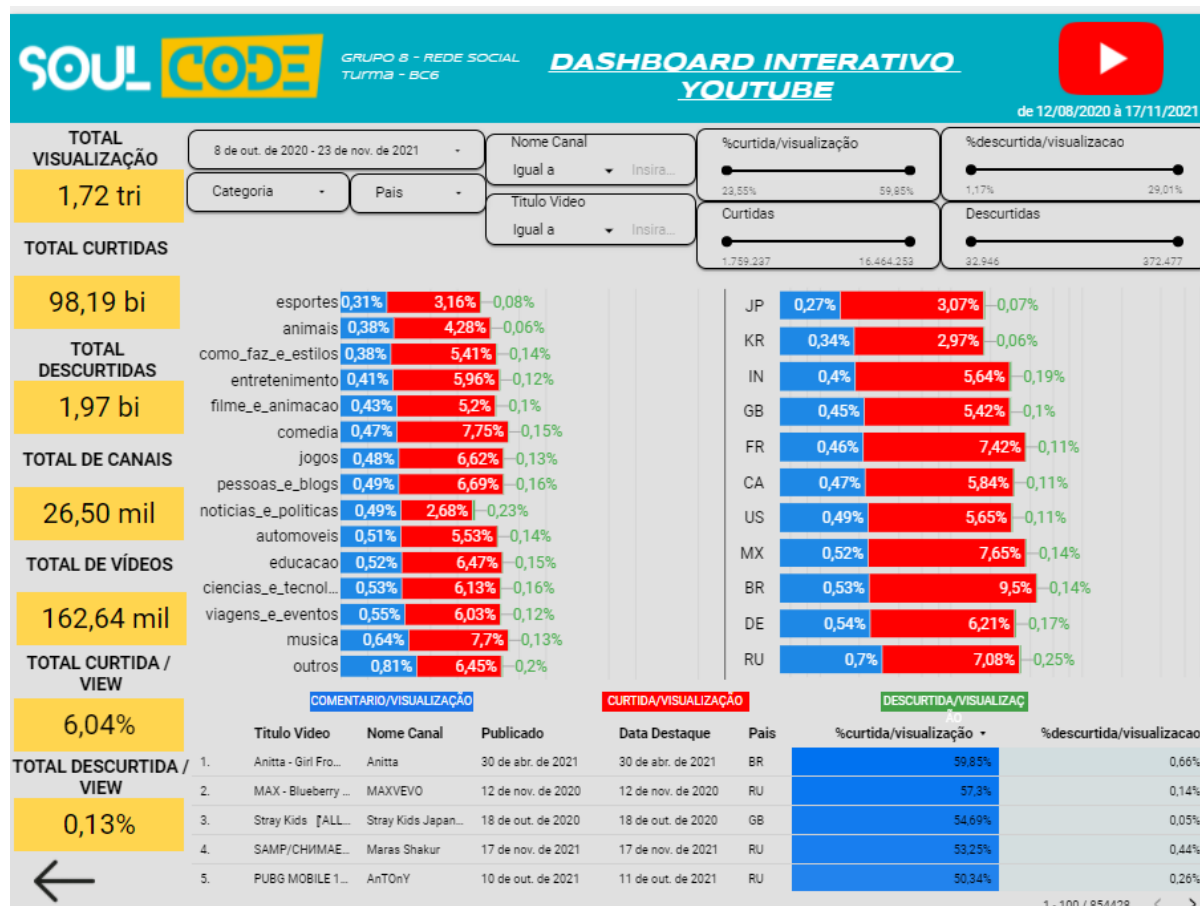
Row	nome_canal	numero_videos_destaque
1	ge	322
2	A Fazenda	207
3	Felipe Neto	166
4	De Placa	142
5	MasterChef Brasil	94

DATA STUDIO - YOUTUBE

O Google Data Studio é uma ferramenta online de Business Intelligence, onde é possível criar dashboards interativos, de forma a apresentar as conclusões de forma visual com gráficos e planilhas interagindo de forma simplificada.

Essa sendo a última etapa do projeto, é possível mostrar todas as consultas feitas ao longo do projeto graficamente, trazendo outra forma de apresentar os dados com gráficos e números, tornando a visão do usuário final mais clara e direta. O dashboard pode ser visualizado clicando [aqui](#).





Alguns insights extraídos:

- Vídeo com mais visualizações em 6 países: BTS (방탄소년단) 'Butter' Official MV
- Categoria que tem mais curtidas respectivamente são: Música, entretenimento e Esporte
- Categoria com mais descurtidas: Entretenimento
- Brasil e Rússia são os países que mais interagem na plataforma.
- O vídeo com mais curtidas em relação ao número de visualizações no Brasil é Anitta - Girl From Rio, já na Alemanha é **killing STALKING** Episode .O1 [Ref. Video]
- No Brasil, o top 3 categoria, respectivamente é: Música, Entretenimento, Pessoas e Blogs
- Os vídeos mais descurtidos são da Índia.

Vale frisar que, na categoria entretenimento há uma mescla de conteúdos que podem variar como jogos, filmes e comédia, por exemplo.

9. ROTEIRO LINKEDIN

[Base de dados linkedin](#)

A premissa deste banco de dados foi verificar que tipos de informações diferentes podem ser encontradas em uma rede social diferente.

1º notebook: 001_LINKEDIN_SOCIAL_MEDIA_TRATAMENTO_DE_DADOS_Nivel_Pandas

Seguindo a mesma tratativa da análise exploratória feita no roteiro de youtube, esse dataset do linkedin possuía a princípio 49 colunas e 62706 linhas.

A primeira ação foi remover os “id’s” duplicados da coluna “m_urn_id”, resumindo o perfil dos “influencers” deste dataset apenas para a vaga mais atual que eles ocupam.

Logo após foram renomeados os nomes das colunas traduzindo elas do inglês para o português e também o conteúdo das colunas com dados categóricos (texto).

A forma escolhida para tradução, foi criar duas listas, uma com as palavras em inglês e outra com as palavras em português. Depois, utilizamos o comando “replace” com as duas listas para alterar todos os campos de cada coluna.

Verificando valores da coluna

```
[ ] 1 df2['nacionalidade'].unique()

array(['east_asian', 'hispanic', 'celtic_english', 'european', 'muslim',
       'south_asian', 'nordic', 'african', 'jewish', 'greek'],
      dtype=object)
```

Realizando alterações dos valores

```
[ ] 1 nacionalidades_antigas = ['east_asian', 'hispanic', 'celtic_english', 'european', 'muslim',
2   |   |   | 'south_asian', 'nordic', 'african', 'jewish', 'greek']
3 nacionalidades_novas = ['asiatico_leste', 'hispanico', 'ingles', 'europeu', 'muculmano',
4   |   |   | 'asiatico_sul', 'nordico', 'africano', 'judeu', 'grego']

[ ] 1 df2['nacionalidade'] = df2['nacionalidade'].replace(nacionalidades_antigas, nacionalidades_novas)
```

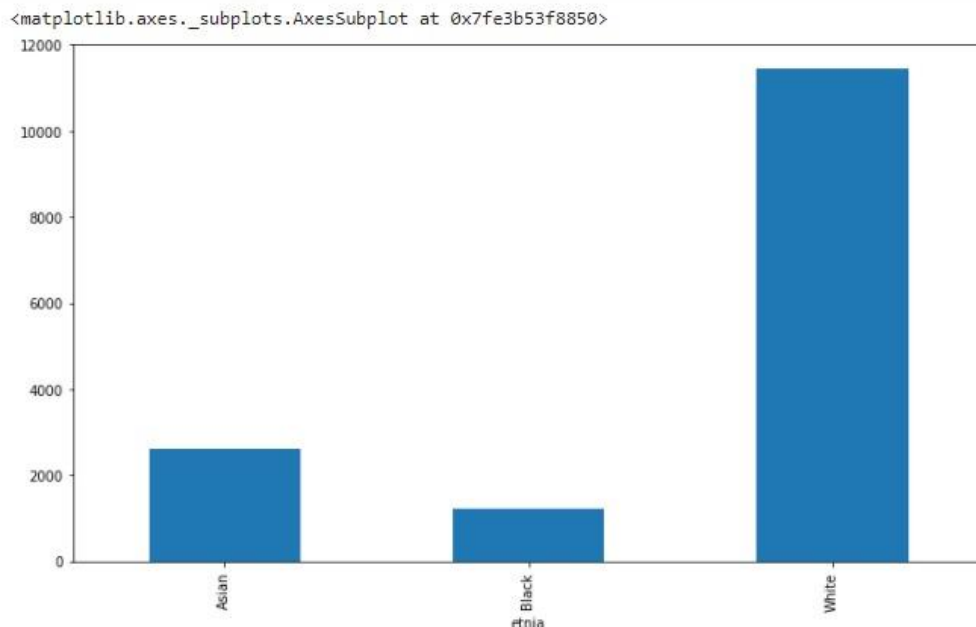
Para fim de maior compreensão do banco de dados, uma breve explicação sobre alguns grupos regionais:

- O Sul asiático (asiatico_sul) também é conhecido como Ásia Meridional, essa região é composta por um grupo de países, são eles: Bangladesh, Butão, Índia, Maldivas, Nepal, Paquistão e Sri Lanka.
- O Leste asiático (asiatico_leste) também é conhecido como Ásia Oriental, essa região é composta por: China, Coreia do Sul, Coreia do Norte, Japão, Mongólia e Taiwan.
- O Hispânico (hispanico) são os países da América Latina e Espanha.

- O Nórdico (nordico) são os países: Noruega, Suécia, Dinamarca, Finlândia e Islândia.

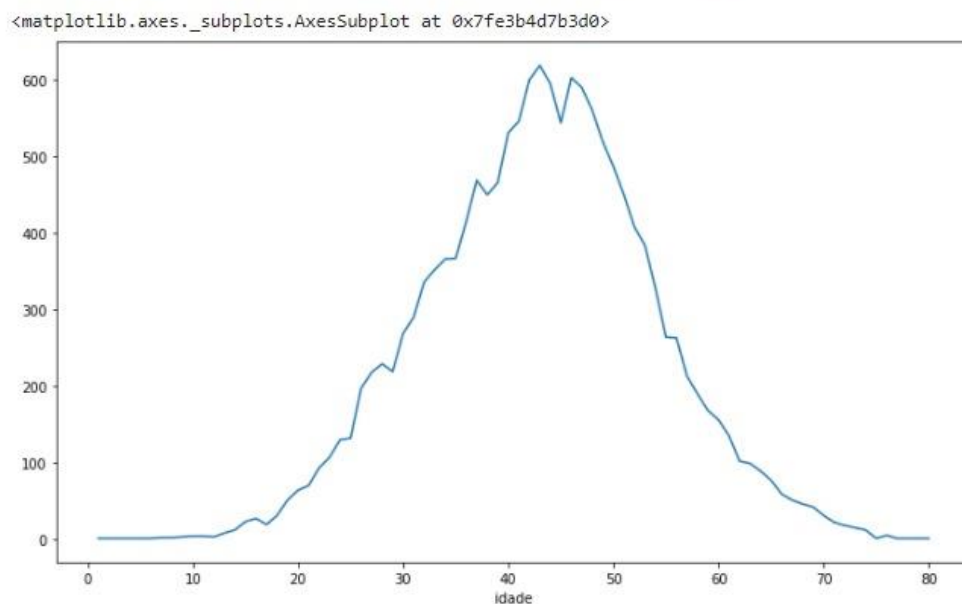
Plotagem de Etnia

```
[ ] 1 df2.groupby(["etnia"]).genero.count().plot.bar(figsize=(12,7))
```



Plotagem de Idade

```
[ ] 1 df2.groupby(["idade"]).idade.count().plot.line(figsize=(12,7))
```



Plotagem aqui também foi usada para verificar possíveis desvios.

Nessa linha foram feitos mais alguns filtros, como retirar as pessoas com idade inferior a 16 anos e aquelas com valor negativo no tempo de cargo atual

Filtrando idades menores que 16

```
[ ] 1 indexIdades = df2[ df2['idade'] < 16 ].index
```

Removendo as idades filtradas

```
[ ] 1 df2.drop(indexIdades , inplace=True)
```

```
[ ] 1 df2.idade.unique()
```

```
array([37, 39, 59, 31, 42, 51, 33, 50, 35, 47, 38, 61, 26, 44, 28, 60, 41,  
       19, 46, 25, 30, 58, 48, 40, 52, 34, 56, 49, 36, 45, 27, 55, 43, 63,  
       53, 57, 54, 20, 32, 22, 29, 65, 66, 23, 67, 24, 73, 71, 68, 16, 21,  
       62, 17, 18, 70, 69, 64, 72, 78, 74, 76, 75, 80, 77])
```

Por fim validando o tipo dos dados das colunas com Pandera:

```
schema = pa.DataFrameSchema(  
    columns = {  
        "tempo_cargo_anterior":pa.Column(pa.Float),  
        "id_usuario":pa.Column(pa.Int),  
        "promocoes":pa.Column(pa.Int),  
        "dias_cargo_anterior":pa.Column(pa.Int),  
        "idade":pa.Column(pa.Int),  
        "desfoque":pa.Column(pa.Float),  
        "raiva":pa.Column(pa.Float),  
        "desgosto":pa.Column(pa.Float),  
        "medo":pa.Column(pa.Float),  
        "felicidade":pa.Column(pa.Float),  
        "neutro":pa.Column(pa.Float),  
        "triste":pa.Column(pa.Float),  
        "surpresa":pa.Column(pa.Float),  
        "etnia":pa.Column(pa.String),  
        "genero":pa.Column(pa.String),  
        "olhos":pa.Column(pa.String),  
        "sorriso":pa.Column(pa.Float),  
        "nacionalidade":pa.Column(pa.String),  
        "seguidores":pa.Column(pa.Int),  
        "qualidade_imagem":pa.Column(pa.Float),  
    }  
)
```

```
schema.validate(df2)
```

Ao fim seguindo com todo tratamento e validação do dataset, este se resumiu a 20 colunas e 11841 linhas, por fim finalizando esta etapa salvando o dataset como "linkedin_tratado_pandas.csv" no Google Cloud Storage.

Link do notebook:

<https://colab.research.google.com/drive/1WXjY6Jf9ayKAqc29Val1zkP82wwYef4O?usp=sharing>

2º notebook: 002_LINKEDIN_SOCIAL_MEDIA_PARAMETROS_Nivel_Pyspark

A etapa de manipulação com PySpark se inicia com a validação do tipo dos dados, carregando o arquivo “linkedin_tratado_pandas.csv” com a montagem do “schema” do PySpark no carregamento.

```
customSchema = StructType([
    StructField("tempo_cargo_anterior", DoubleType(), True),
    StructField("id_usuario", IntegerType(), True),
    StructField("promocooes", IntegerType(), True),
    StructField("dias_cargo_anterior", IntegerType(), True),
    StructField("idade", IntegerType(), True),
    StructField("desfoque", DoubleType(), True),
    StructField("raiva", DoubleType(), True),
    StructField("desgosto", DoubleType(), True),
    StructField("medo", DoubleType(), True),
    StructField("felicidade", DoubleType(), True),
    StructField("neutro", DoubleType(), True),
    StructField("triste", DoubleType(), True),
    StructField("surpresa", DoubleType(), True),
    StructField("etnia", StringType(), True),
    StructField("genero", StringType(), True),
    StructField("oculos", StringType(), True),
    StructField("sorriso", DoubleType(), True),
    StructField("nacionalidade", StringType(), True),
    StructField("seguidores", IntegerType(), True),
    StructField("qualidade_imagem", DoubleType(), True)
])

schema = customSchema

dfspark = pd.read_csv('gs://projeto-final-grupo8/saida/linkedin_tratado_pandas.csv', sep=',', header=0)
df = spark.createDataFrame(dfspark, schema=schema)
```

Logo em seguida o primeiro passo feito foi arredondar para duas casas decimais os campos numéricos, a fim de criar um padrão para o dataset.

Arredondando os valores das colunas necessárias

```
[ ] 1 df = (df.withColumn("tempo_cargo_anterior", F.round(F.col("tempo_cargo_anterior"), 2))
2      .withColumn("desfoque", F.round(F.col("desfoque"), 2))
3      .withColumn("raiva", F.round(F.col("raiva"), 2))
4      .withColumn("felicidade", F.round(F.col("felicidade"), 2))
5      .withColumn("neutro", F.round(F.col("neutro"), 2))
6      .withColumn("triste", F.round(F.col("triste"), 2))
7      .withColumn("surpresa", F.round(F.col("surpresa"), 2))
8      .withColumn("sorriso", F.round(F.col("sorriso"), 2))
9      .withColumn("qualidade_imagem", F.round(F.col("qualidade_imagem"), 2))
10 )
11
```

Foram gerados na sequência alguns insights com a combinação dos comandos “groupBy” e “aggregate”.

Agrupando a coluna "nacionalidade" e utilizando Aggregate para fazer pesquisas

```
[ ] 1 df.groupby(F.col("nacionalidade")).agg(
2     F.round(F.mean("promocoes"), 2).alias("media_promocoes"),
3     F.round(F.max("promocoes"), 2).alias("max_promocoes"),
4     F.round(F.mean("idade"), 2).alias("media_idade"),
5     F.round(F.min("idade"), 2).alias("min_idade"),
6     F.max("idade").alias("max_idade")
7 ).orderBy("max_promocoes")
8 .show()
```

nacionalidade	media_promocoes	max_promocoes	media_idade	min_idade	max_idade
asiatico_leste	1.0	1	34.37	16	75
ingles	1.0	1	47.14	16	80
muculmano	1.0	1	40.71	19	71
judeu	1.0	1	47.16	26	66
grego	1.0	1	44.6	22	69
europau	1.0	1	44.73	17	73
hispanico	1.0	1	42.51	16	76
asiatico_sul	1.0	1	38.04	16	71
nordico	1.0	1	47.35	26	70
africano	1.0	1	40.68	20	68

Como uma próxima etapa, foram utilizadas as "Window Functions" de forma a demonstrar algumas outras ideias.

Ordenando por idade e selecionando apenas algumas colunas

```
[ ] 1 w0 = Window.partitionBy(F.col("idade")).orderBy("idade")
2
3 (df.withColumn("Rank", F.rank().over(w0))
4  .withColumn("idade", F.max(F.col("idade")).over(w0))
5  .select("id_usuario", "genero", "idade", "nacionalidade", "dias_cargo_anterior")
6  .filter(F.col("dias_cargo_anterior") > 1).show(10)
7 )
```

id_usuario	genero	idade	nacionalidade	dias_cargo_anterior
3511	Masculino	16	asiatico_leste	1003
3635	Masculino	16	hispanico	366
4451	Masculino	16	hispanico	61
4902	Masculino	16	ingles	152
6235	Masculino	16	asiatico_leste	1219
6990	Feminino	16	asiatico_leste	762
7008	Feminino	16	ingles	730
7962	Feminino	16	asiatico_leste	61
8158	Feminino	16	asiatico_leste	640
9054	Feminino	16	hispanico	3012

only showing top 10 rows

Finalizando mais uma etapa salvando o dataframe no Google Cloud Storage, o data lake usado durante as etapas deste projeto como "linkedin_tratado_pyspark".

Link do notebook:

<https://colab.research.google.com/drive/1EunkrtzvQD2AGaTleXVudzmj8BXrF4ja?usp=sharing>

3º Notebook: 003 LINKEDIN SOCIAL MEDIA CONSULTAS Nivel SparkSQL

Neste notebook foram feitas as consultas SQL através do SparkSQL, estas mesmas consultas estão sendo usadas no BigQuery (Data Warehouse) do projeto logo após este tópico, segue abaixo algumas amostras logo abaixo de todas as consultas.

00.Consulta geral tabela

```
[ ] 1 spark.sql("SELECT * FROM leitura_linkedin LIMIT 10").show()
```

tempo_cargo_anterior	id_usuario	promoco	dias_cargo_anterior	idade	desfoque	raiva	desgosto	medo	felicidade	neutro	triste	surpresa	etnia	genero	oculos
2.0	3008	1	457	37	0.42	76.6	0.7	0.7	99.04	0.09	0.01	0.09	Asiático	Masculino	Oculos_comum
1.0	3010	1	1827	37	64.26	8.2	8.2	8.2	9.59	63.07	0.1	27.0	Branco	Masculino	Oculos_comum
1.0	3012	1	243	31	2.13	0.7	17.5	5.7	73.23	14.74	0.01	11.78	Branco	Masculino	Oculos_comum
1.0	3013	1	3925	42	0.35	0.0	0.0	0.0	100.0	0.0	0.0	0.0	Branco	Feminino	Nenhum
1.0	3014	1	699	51	0.32	1.5	34.7	29.4	98.66	0.09	0.02	0.58	Branco	Feminino	Nenhum
1.0	3015	1	61	42	0.37	0.0	0.0	0.0	100.0	0.0	0.0	0.0	Branco	Masculino	Nenhum
1.0	3016	1	395	33	0.36	0.0	0.0	0.0	96.42	3.44	0.0	0.14	Asiático	Masculino	Oculos_comum
2.0	3017	1	1096	50	2.31	0.0	0.0	0.0	100.0	0.0	0.0	0.0	Branco	Masculino	Nenhum
1.0	3018	1	913	35	0.37	0.9	65.0	0.4	56.51	42.82	0.01	0.0	Asiático	Masculino	Oculos_comum
4.0	3020	1	820	39	0.36	0.7	0.7	23.1	92.46	7.29	0.01	0.01	Branco	Masculino	Nenhum

01.Pessoas com mais de 50 anos que usam óculos escuros

```
1 spark.sql("SELECT * FROM leitura_linkedin WHERE oculos = 'Oculos_escuros' AND idade > 50 ORDER BY idade DESC LIMIT 10").show()
```

tempo_cargo_anterior	id_usuario	promoco	dias_cargo_anterior	idade	desfoque	raiva	desgosto	medo	felicidade	neutro	triste	surpresa	etnia	genero	oculos	sorriso
1.0	2546	1	1795	74	0.22	12.7	1.8	2.8	96.59	0.07	3.15	0.02	Branco	Masculino	Oculos_escuros	62.73
1.0	12557	1	3622	71	0.27	0.0	0.1	0.0	0.17	99.82	0.0	0.01	Negro	Masculino	Oculos_escuros	12.58
1.0	4143	1	61	69	0.22	3.5	10.0	3.5	97.99	0.4	1.08	0.36	Branco	Masculino	Oculos_escuros	29.83
1.0	1054	1	92	68	0.75	0.4	0.1	2.2	0.0	97.1	2.8	0.07	Branco	Masculino	Oculos_escuros	14.53
1.0	7701	1	365	67	19.82	0.3	0.3	0.3	0.49	95.86	1.02	2.62	Branco	Masculino	Oculos_escuros	11.37
1.0	10984	1	212	67	74.03	0.0	0.0	1.0	0.02	99.97	0.0	0.0	Branco	Masculino	Oculos_escuros	15.31
1.0	11107	1	1673	66	87.99	0.4	15.2	0.4	87.37	12.46	0.0	0.0	Branco	Masculino	Oculos_escuros	70.34
1.0	12391	1	3774	65	32.98	17.2	68.4	5.6	0.06	0.51	14.31	84.21	Branco	Masculino	Oculos_escuros	52.2
1.0	7985	1	1065	64	0.98	56.7	2.6	2.6	67.07	31.96	0.03	0.32	Branco	Masculino	Oculos_escuros	85.42
1.0	42	1	1310	64	19.06	68.5	1.7	1.7	4.49	94.59	0.14	0.07	Negro	Masculino	Oculos_escuros	47.63

02.Consulta pessoas negras entre 16 e 20 anos com suas respectivas nacionalidades e número de seguidores

```
1 spark.sql("SELECT idade, genero, nacionalidade, seguidores, qualidade_imagem \
2 FROM leitura_linkedin \
3 WHERE (idade >= 16 AND idade <= 20 AND etnia = 'Negro') \
4 ORDER BY seguidores DESC").show()
```

idade	genero	nacionalidade	seguidores	qualidade_imagem
18	Feminino	asiatico_sul	1952	93.31
19	Masculino	asiatico_leste	1740	93.37
18	Masculino	asiatico_sul	1594	93.12
20	Feminino	africano	1460	82.49
18	Masculino	asiatico_sul	1293	88.53
20	Masculino	asiatico_sul	916	77.78
16	Masculino	asiatico_sul	867	87.7
20	Masculino	ingles	830	92.35
20	Masculino	europau	772	87.31
19	Masculino	asiatico_sul	743	0.46
20	Masculino	muculmano	679	90.02
19	Masculino	asiatico_sul	652	51.78
20	Masculino	asiatico_sul	600	6.56
17	Masculino	asiatico_leste	314	85.72
20	Masculino	asiatico_sul	244	82.47
20	Masculino	ingles	0	30.06

03.Consulta pessoas que usam ou não óculos por nacionalidade

```
1 spark.sql("SELECT nacionalidade, COUNT(nacionalidade) AS qty_nacionalidade, SUM(case when oculos = 'Oculos_comum' then 1 else 0 end) AS usa_oculos, \
2 SUM(case when oculos = 'Oculos_escuros' then 1 else 0 end) AS usa_oculos_escuros, \
3 SUM(case when oculos = 'Nenhum' then 1 else 0 end) AS sem_oculos \
4 FROM leitura_linkedin \
5 GROUP BY nacionalidade \
6 ORDER BY qty_nacionalidade DESC").show()
```

nacionalidade	qty_nacionalidade	usa_oculos	usa_oculos_escuros	sem_oculos
ingles	4491	755	78	3658
asiatico_sul	1947	487	88	1372
europetu	1893	324	39	1530
asiatico_leste	1220	432	17	771
hispanico	1177	186	27	964
muculmano	675	149	23	503
nordico	246	59	4	183
africano	85	11	1	73
grego	62	8	0	54
judeu	45	6	2	37

04.Consulta média de seguidores e quantidade de mulheres influenciadoras no LinkedIn por nacionalidade

```
1 spark.sql("SELECT nacionalidade, COUNT(nacionalidade) AS qty_mulheres, ROUND(MEAN(seguidores), 2) AS media_seguidores \
2 FROM leitura_linkedin \
3 WHERE genero = 'Feminino' \
4 GROUP BY nacionalidade \
5 ORDER BY media_seguidores DESC").show()
```

nacionalidade	qty_mulheres	media_seguidores
muculmano	106	1025.08
europetu	470	1017.66
africano	15	999.0
asiatico_sul	328	980.11
grego	16	943.31
ingles	1237	939.12
hispanico	367	821.1
judeu	6	786.5
asiatico_leste	392	698.7
nordico	56	695.02

05.Consulta quantidade de homens e mulheres por etnia

```
1 spark.sql("SELECT DISTINCT etnia, COUNT(etnia) AS qty_etnia, SUM(case when genero = 'Masculino' then 1 else 0 end) AS qty_homens, \
2 SUM(case when genero = 'Feminino' then 1 else 0 end) AS qty_mulheres \
3 FROM leitura_linkedin \
4 GROUP BY etnia").show()
```

etnia	qty_etnia	qty_homens	qty_mulheres
Negro	942	821	121
Asiático	1963	1346	617
Branco	8936	6681	2255

06.Consulta número de homens e mulheres por etnia, nacionalidade e sexo

```
1 spark.sql("SELECT DISTINCT etnia, nacionalidade, SUM(case when genero = 'Masculino' then 1 else 0 end) AS qty_homens, \
2                                     SUM(case when genero = 'Feminino' then 1 else 0 end) AS qty_mulheres \
3                                     FROM leitura_linkedin \
4                                     GROUP BY etnia, nacionalidade \
5                                     ORDER BY nacionalidade, etnia").show(50)
```

etnia	nacionalidade	qty_homens	qty_mulheres
Asiático	africano	6	4
Branco	africano	39	6
Negro	africano	25	5
Asiático	asiatico_leste	687	321
Branco	asiatico_leste	111	62
Negro	asiatico_leste	30	9
Asiático	asiatico_sul	129	27
Branco	asiatico_sul	1061	259
Negro	asiatico_sul	429	42
Asiático	européu	93	45
Branco	européu	1279	413
Negro	européu	51	12
Asiático	grego	0	2
Branco	grego	45	14
Negro	grego	1	0
Asiático	hispanico	126	72
Branco	hispanico	644	284
Negro	hispanico	40	11

07.Consulta quantidade de homens e mulheres entre 36 e 40 anos de acordo com a etnia

```
1 spark.sql("SELECT DISTINCT idade, etnia, SUM(case when genero = 'Masculino' then 1 else 0 end) AS qty_homens, \
2                                     SUM(case when genero = 'Feminino' then 1 else 0 end) AS qty_mulheres \
3                                     FROM leitura_linkedin \
4                                     WHERE idade >= 36 AND idade <= 40 \
5                                     GROUP BY idade, etnia \
6                                     ORDER BY etnia, idade").show(50)
```

idade	etnia	qty_homens	qty_mulheres
36	Asiático	59	15
37	Asiático	45	30
38	Asiático	50	11
39	Asiático	44	17
40	Asiático	64	17
36	Branco	166	51
37	Branco	188	63
38	Branco	185	62
39	Branco	200	64
40	Branco	247	68
36	Negro	36	2
37	Negro	32	9
38	Negro	31	6
39	Negro	21	4
40	Negro	42	0

08.Consulta idades (mais velho, média e mais novo) por etnia e quantos usam ou não usam óculos

```
1 spark.sql("SELECT DISTINCT etnia, MAX(idade) AS mais_velho, ROUND(MEAN(idade), 2) AS media_idade, MIN(idade) AS mais_novo, \
2                                     SUM(case when olhos = 'Oculos_comum' then 1 else 0 end) AS usa_olhos, \
3                                     SUM(case when olhos = 'Oculos_escuros' then 1 else 0 end) AS usa_olhos_escuros, \
4                                     SUM(case when olhos = 'Nenhum' then 1 else 0 end) AS sem_olhos \
5                                     FROM leitura_linkedin \
6                                     GROUP BY etnia \
7                                     ORDER BY etnia").show(50)
```

etnia	mais_velho	media_idade	mais_novo	usa_olhos	usa_olhos_escuros	sem_olhos
Asiático	75	36.37	16	624	68	1271
Branco	80	44.84	16	1556	179	7201
Negro	76	40.09	16	237	32	673

Link do notebook: <https://colab.research.google.com/drive/1YYSGxd9Irnmi0e8fQurUsb-eu9LMmdN1?usp=sharing>

BIGQUERY (DATA WAREHOUSE) - LINKEDIN

Seguindo a mesma lógica desse tópico para o dataset do youtube já mostrado anteriormente, neste ponto as consultas são feitas através da linguagem SQL dentro do BigQuery.

Logo abaixo será mostrado algumas amostras, de como são as consultas salvas nesse Data Warehouse. Elas estão seguindo a mesma lógica das consultas do SparkSQL.

03.Consulta pessoas que usam ou não óculos por nacionalidade

The screenshot shows the BigQuery interface with a query titled "03.In_pessoas_que_usam_oculos_nacionalidade". The query is as follows:

```
--03.Consulta pessoas que usam ou não óculos por nacionalidade
2 SELECT nacionalidade, COUNT(nacionalidade) AS qty_nacionalidade, SUM(case when oculos = 'Oculos_comum' then 1 else 0 end) AS usa_oculos,
3 SUM(case when oculos = 'Oculos_escuros' then 1 else 0 end) AS usa_oculos_escuros,
4 SUM(case when oculos = 'Nenhum' then 1 else 0 end) AS sem_oculos
5 FROM in_social_midia.linkedIn
6 GROUP BY nacionalidade
```

The query results are displayed in a table with the following columns: Row, nacionalidade, qty_nacionalidade, usa_oculos, usa_oculos_escuros, sem_oculos. The results are as follows:

Row	nacionalidade	qty_nacionalidade	usa_oculos	usa_oculos_escuros	sem_oculos
1	ingles	4491	755	78	3658
2	asiatico_sul	1947	487	88	1372
3	europau	1893	324	39	1530
4	asiatico_leste	1220	432	17	771
5	hispanico	1177	186	27	964
6	muculmano	675	149	23	503

05.Consulta quantidade de homens e mulheres por etnia

The screenshot shows the BigQuery interface with a query titled "05.In_homem_x_mulher_etnia". The query is as follows:

```
--05.Consulta quantidade de homens e mulheres por etnia
2 SELECT DISTINCT etnia, COUNT(etnia) AS qty_etnia, SUM(case when genero = 'Masculino' then 1 else 0 end) AS qty_homens,
3 SUM(case when genero = 'Feminino' then 1 else 0 end) AS qty_mulheres
4 FROM in_social_midia.linkedIn
5 GROUP BY etnia
```

The query results are displayed in a table with the following columns: Row, etnia, qty_etnia, qty_homens, qty_mulheres. The results are as follows:

Row	etnia	qty_etnia	qty_homens	qty_mulheres
1	Branco	8936	6681	2255
2	Asiático	1963	1346	617
3	Negro	942	821	121

08.Consulta idades (mais velho, média e mais novo) por etnia e quantos usam ou não usam óculos

```
1 spark.sql("SELECT DISTINCT etnia, MAX(idade) AS mais_velho, ROUND(MEAN(idade), 2) AS media_idade, MIN(idade) AS mais_novo, \
2 | SUM(case when oculos = 'Oculos_comum' then 1 else 0 end) AS usa_olculos, \
3 | SUM(case when oculos = 'Oculos_escuros' then 1 else 0 end) AS usa_olculos_escuros, \
4 | SUM(case when oculos = 'Nenhum' then 1 else 0 end) AS sem_olculos \
5 | FROM leitura_linkedin \
6 | GROUP BY etnia \
7 | ORDER BY etnia").show(50)
```

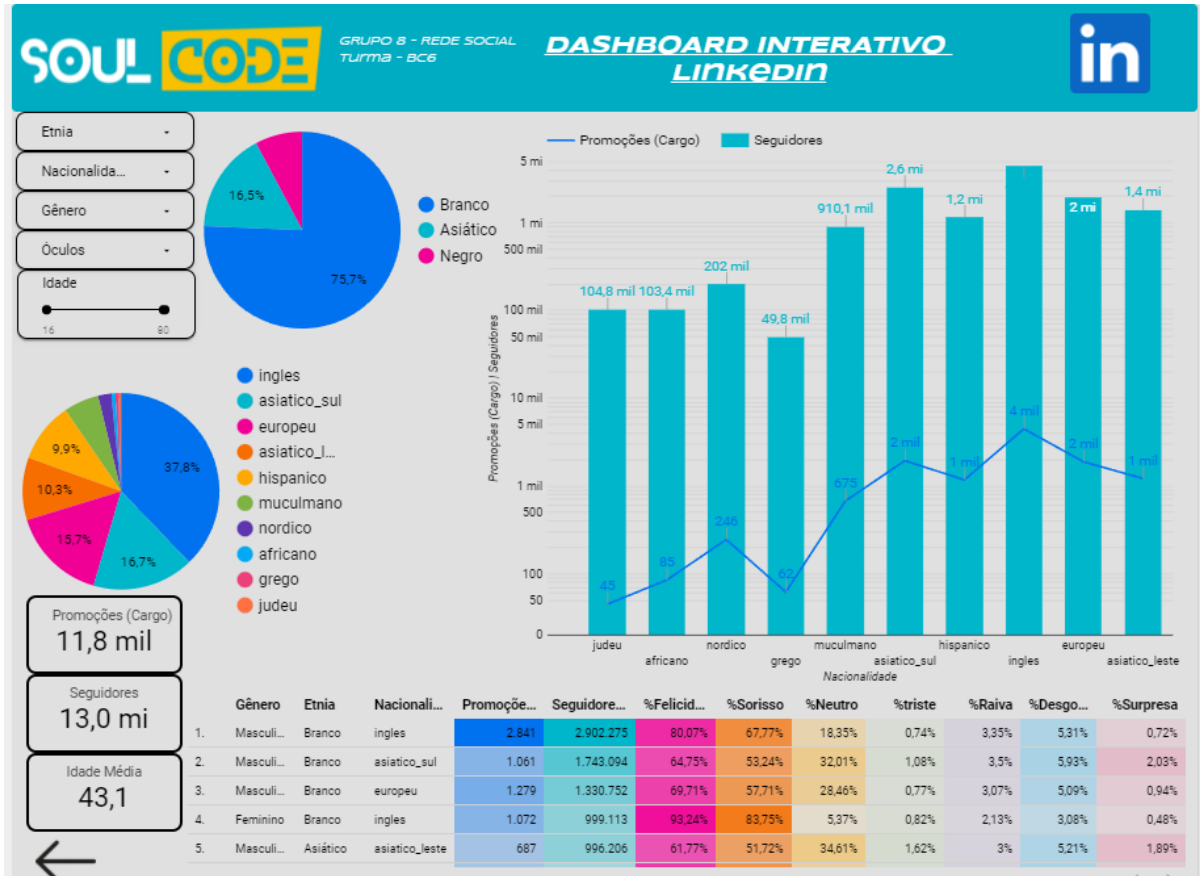
etnia	mais_velho	media_idade	mais_novo	usa_olculos	usa_olculos_escuros	sem_olculos
Asiático	75	36.37	16	624	68	1271
Branco	80	44.84	16	1556	179	7201
Negro	76	40.09	16	237	32	673

Link do notebook: <https://colab.research.google.com/drive/1YYSgxd9lrnmi0e8fQurUsb-eu9LMmdN1?usp=sharing>

DATA STUDIO - LINKEDIN

O Google Data Studio é uma ferramenta online de Business Intelligence, como já explicado neste mesmo tópico do dataset youtube.

Essa sendo a última etapa do projeto, serão mostradas abaixo graficamente as informações derivadas dos insights gerados. Pode acessar o dashboard clicando [aqui](#).



Alguns insights extraídos:

- Homens, brancos de nacionalidade Inglesa lideram o primeiro lugar em número de promoções recebidas e de seguidores, seguido de homens, brancos de nacionalidade Ásia Meridional, homens, brancos de nacionalidade europeia, mulheres, brancas de nacionalidade inglesa e Homens, brancos de nacionalidade da Ásia ocidental.
- No contexto geral, a etnia branca domina tem os maiores números de seguidores e promoções.
- Proporcionalmente, as etnias que mais estão registradas nesse banco de dados são branca (75,7%), asiática (16,5%) e negro (7,8%).
- As 5 maiores nacionalidades por quantidade de usuário, respectivamente Ingles, Ásia Meridional, Europeu, Ásia Oriental e Hispanico.
- A média de idade dos usuários é de 43 anos.
- O grupo de usuários que mais tem foto de perfil com característica de felicidade são mulheres, asiáticas e gregas.
- Já as que tiram foto mais sorrindo são mulheres, de etnia negra e muçulmano.
- Fotos com mais raiva são homens, asiáticos judeus.

Analisando bem todas as informações desse banco de dados, foi possível verificar que não há relação com as emoções transmitidas nas fotos com o número de seguidores e promoções. Não foi possível analisar, por exemplo, se uma foto de perfil passando mais alegria tem mais chances de ser chamada para uma entrevista de emprego, tendo em vista que não há esse tempo de informação na base de dados ou outras informações de interação que há dentro do LinkedIn.

10. CONTATOS



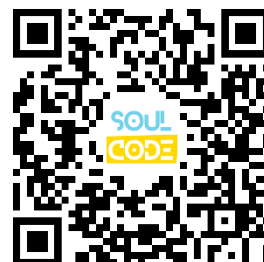
Nome: André Victor Moreira Costa
E-mail: andrevictorm2017@gmail.com
LinkedIn: andre-victor-moreira-costa



Victor Gonçalves
E-mail: victor.og17@gmail.com
LinkedIn: victor-de-oliveira-goncalves



Eduardo Mathias
E-mail: eduardo.mathiass09@gmail.com
LinkedIn: eduardo-mathias



Thiago Regis
E-mail: promothiagor@gmail.com
LinkedIn: thiagoregis1

