cloudbees

alessandra mayumi | filipe gonçalves | paola pagotto | thiago franchini

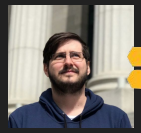
accenture

GAMA ACADEMY

o time

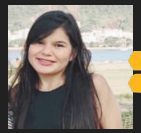
Alessandra Mayumi

Analista de Desenvolvimento
Campinas, SP



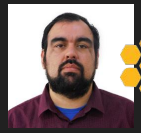
Filipe Gonçalves

Desenvolvedor Backend
Rio de Janeiro, RJ



Paola Pagotto

Desenvolvedora Mobile | Mestranda em SI
Rio de Janeiro, RJ



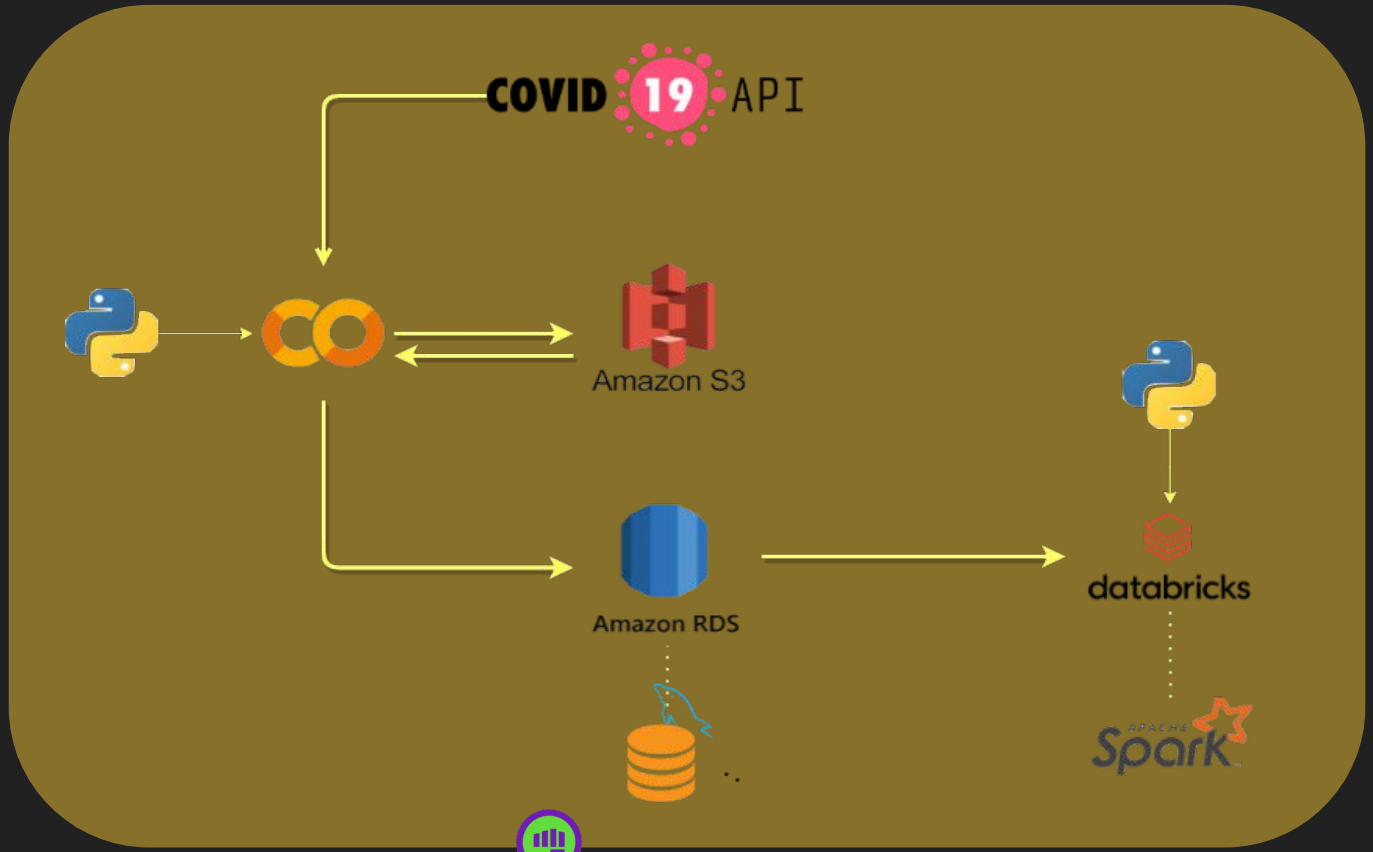
Thiago Franchini

Analista de Infraestrutura
São Paulo, SP

accenture

GAMA ACADEMY

arquitetura



accenture

GAMA ACADEMY

Colab - consulta à API

Utilizando a linguagem Python se fez a leitura da API covid19api com a bilblioteca requests. Em seguida, os registros foram armazenados em arquivo json com a biblioteca json.

https://documenter.getpostman.com/view/10808728/SzS8rjbc

GET from API to Colab

```python
import json
import requests

base_url = 'https://api.covid19api.com'

countries_req = requests.get(base_url + '/countries')
countries_str = countries_req.text
countries_json = json.loads(countries_str)
response_list = []

for country in countries_json:
    country_req = requests.get(base_url + '/country/' + country['Slug'])
    country_str = country_req.text
    if country_req.status_code == 200 and len(country_str) > 10:
        try:
            country_json = json.loads(country_str)
            response_list += country_json
        except:
            pass
response_str = json.dumps(response_list)
filename = 'covid19api.json'
f = open(filename, 'w')
f.write(response_str)
f.close()
```

Colab - comunicação com o S3 bucket

```
import boto3

aws_access_key_id = [redacted]
aws_secret_access_key = [redacted]

bucket = 'cloudbees-bucket'
key = 'covid19api.json'
keyUsa = 'covid19api-usa.json'

s3_resource = boto3.resource('s3',
                    aws_access_key_id = aws_access_key_id,
                    aws_secret_access_key= aws_secre

s3_resource_usa = boto3.resource('s3',
                    aws_access_key_id = aws_access_key_id,
                    aws_secret_access_key = aws_secret_access_key)
```

```
s3_resource.Bucket(bucket).upload_file(key, key)
s3_resource_usa.Bucket(bucket).upload_file(keyUsa, keyUsa)
```

```
s3_resource.Bucket(bucket).download_file(key, key)
s3_resource_usa.Bucket(bucket).download_file(keyUsa, keyUsa)
```

Amazon S3

accenture

GAMA ACADEMY

Colab - armazenamento no banco de dados MySQL

```python
import mysql.connector

mydb = mysql.connector.connect(
  host="gama.cn27cseohzrm.us-east-2.rds.amazonaws.com",
  user="admin",
  password=[redacted],
  database="cloudbees"
)
mycursor = mydb.cursor()
```


```python
with open('covid19api-usa.json', 'r') as json_file:
  data = json.load(json_file)

for i in data:
  if i['CountryCode'] not in countries:
    countries.add(i['CountryCode'])
    country = (i['CountryCode'] , i['Country'], i['L
    mycursor.execute(sqlCountry, country)

cases = []
for i in data:
  cases.append((i['CountryCode'],i['Confirmed'],i['D
                i['Active'],i['Province'],i['City'],
try:
    mycursor.executemany(sqlUsaCases, cases)
    mydb.commit()
except:
    mydb.rollback()
```

MySQL - diagrama do banco de dados

Cases		
	CountryCode	varchar
	Confirmed	bigint
	Deaths	bigint
	Recovered	bigint
	Active	bigint
	Date	date
	Province	varchar
	City	varchar
	CityCode	varchar

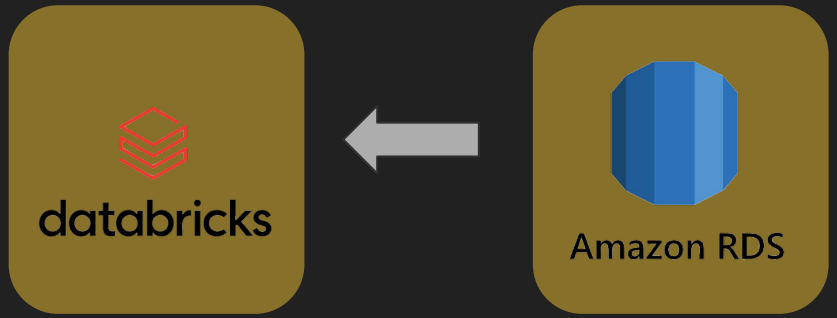
Country		
PK	CountryCode	varchar
	Name	varchar
	Lat	bigint
	Lon	bigint

UsaCases		
	CountryCode	varchar
	Confirmed	bigint
	Deaths	bigint
	Recovered	bigint
	Active	bigint
	Date	date
	Province	varchar
	City	varchar
	CityCode	varchar

accenture

GAMA ACADEMY

Databricks - Ingestão




```
jdbcHostname = "gama.cn27cseohzrm.us-east-2.rds.amazonaws.com"
jdbcPort = 3306
jdbcDatabase = "cloudbees"
jdbcUsername = "admin"
jdbcPassword = '*********'

jdbcUrl = f"jdbc:mysql://{jdbcHostname}:{jdbcPort}/{jdbcDatabase}"
connectionProperties = {'user' : jdbcUsername,'password':
jdbcPassword}
```

A ingestão dos dados no Amazon RDS MySQL é feita em script python utilizando o JDBC driver, sendo armazenados em formato JSON no diretório raw

```
country_query = "(SELECT * FROM Country) country_raw"
countryRawDf = spark.read.jdbc(url=jdbcUrl, table=country_query,
properties=connectionProperties)
countryRawDf.write.format("json").mode("overwrite").save("raw/coun
try.json")

cases_query = "(SELECT * FROM Cases) cases_raw"
casesRawDf = spark.read.jdbc(url=jdbcUrl, table=cases_query,
properties=connectionProperties)
casesRawDf.write.format("json").mode("overwrite").save("raw/cases.
json")

usaCases_query = "(SELECT * FROM UsaCases) usaCases_raw"
usaCasesRawDf = spark.read.jdbc(url=jdbcUrl, table=usaCases_query,
properties=connectionProperties)
usaCasesRawDf.write.format("json").mode("overwrite").save("raw/usa
Cases.json")
```

accenture

GAMA ACADEMY

Databricks - Transformações

Cases		
	CountryCode	varchar
	Confirmed	bigint
	Deaths	bigint
	Recovered	bigint
	Active	bigint
	Date	date
	Province	varchar
	City	varchar
	CityCode	varchar

Country		
PK	CountryCode	varchar
	Name	varchar
	Lat	bigint
	Lon	bigint

Juntar a tabela com informação dos países (Country) com a tabela com detalhes da covid 19 (Cases) e remover as informações que não são necessárias

```
countryDf = spark.read.format("json").load("/raw/country.json")
casesDf = spark.read.format("json").load("/raw/cases.json")
usaCasesDf = spark.read.format("json").load("/raw/usaCases.json")

joinDF = countryDf.join(casesDf, 'countryCode', how='inner')
joinDF = joinDF.drop("Lat").drop("Lon")
```

accenture

GAMA ACADEMY

Databricks - Transformações

```
usaCasesDf = usaCasesDf.groupBy('date').agg(
  sum_spark("Confirmed").alias("Confirmed"),
  sum_spark("Deaths").alias("Deaths"),
  sum_spark("Recovered").alias("Recovered"),
  sum_spark("Active").alias("Active"),
)
usaCasesDf = usaCasesDf.withColumn("CountryCode", lit("US"))
usaCasesDf.createOrReplaceTempView("usa_cases_table")

joinUsaCases = countryDf.join(usaCasesDf, 'countryCode',
how='inner')
joinUsaCases = joinUsaCases.drop("Lat").drop("Lon")

unionDf = joinUsaCases.union(joinDF)
```

Country		
PK	CountryCode	varchar
	Name	varchar
	Lat	bigint
	Lon	bigint

UsaCases		
	CountryCode	varchar
	Confirmed	bigint
	Deaths	bigint
	Recovered	bigint
	Active	bigint
	Date	date
	Province	varchar
	City	varchar
	CityCode	varchar

As informações dos EUA foram acrescentadas separadamente, pois era o caso de dados por província.

accenture

GAMA ACADEMY

Databricks - Transformações - particionamento

Para o particionamento, criou-se uma coluna com o ano e mês para o particionamento da tabela.

```
from pyspark.sql.functions import date_trunc,
year, month, concat_ws

unionDf = unionDf.withColumn("yearMonth",
          concat_ws("-", year(unionDf.date),
          month( unionDf.date)))
unionDf.write.partitionBy("yearMonth")
       .mode("Overwrite").format("parquet")
       .save("covidPartByYearMonth.parquet")
```

accenture

GAMA ACADEMY

queries

Cmd 6

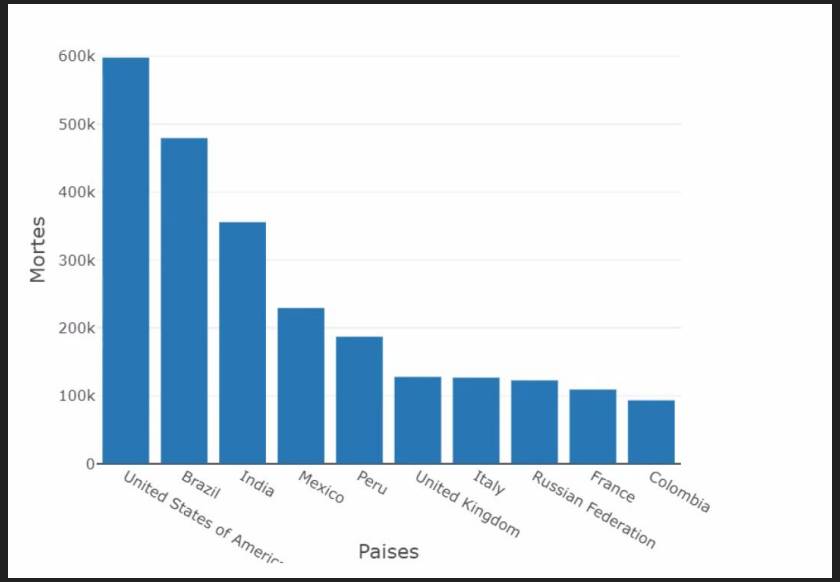
```sql
%sql
SELECT
  Name as Paises,
  date schema,
  Deaths AS Mortes
FROM covid
where CountryCode in(
    Select  countryCode
    FROM covid
    where date= "2021-06-09"
    order by Deaths DESC
    limit 3
)
and date like "%-01"
ORDER BY date
```

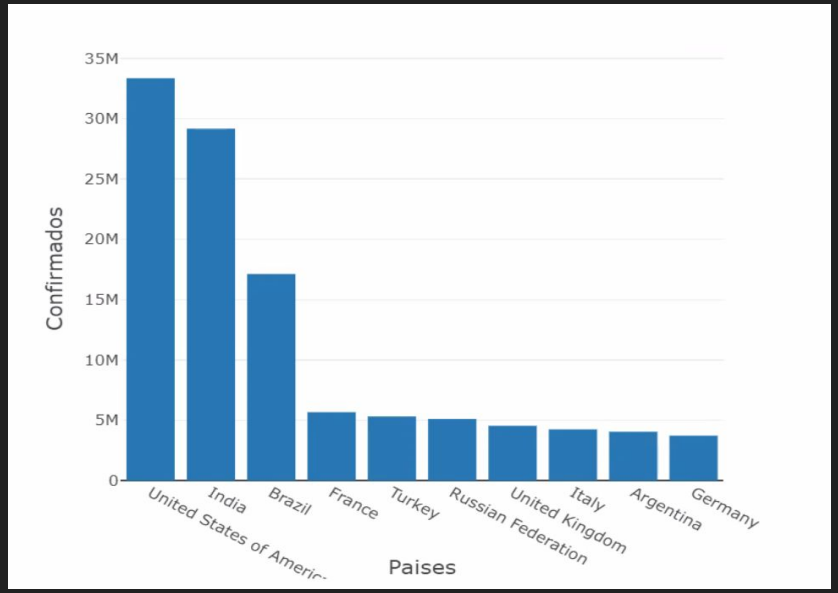
```sql
%sql
SELECT
  Name as Paises,
  Deaths AS Mortes
FROM covid
where
date = "2021-06-09"
ORDER BY Mortes DESC
limit 10
```

accenture

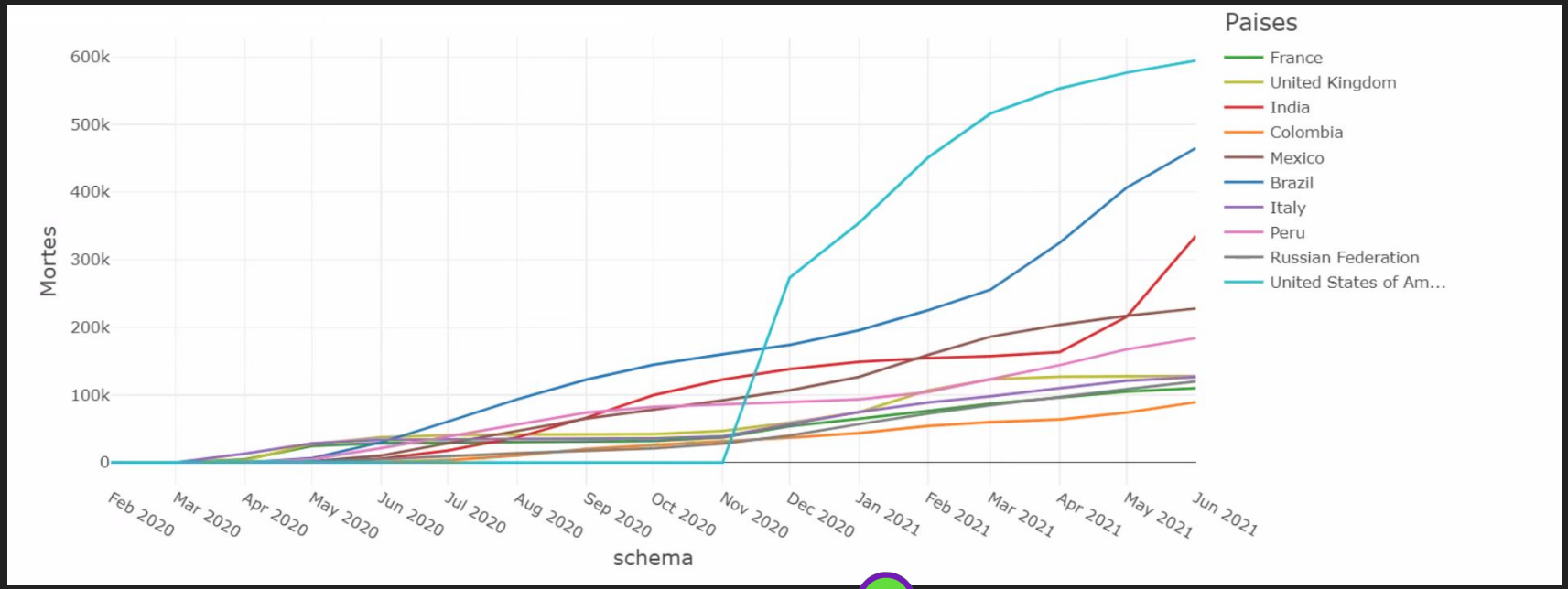
GAMA ACADEMY

visualizações: Os 10 Países com mais mortes e Confirmados





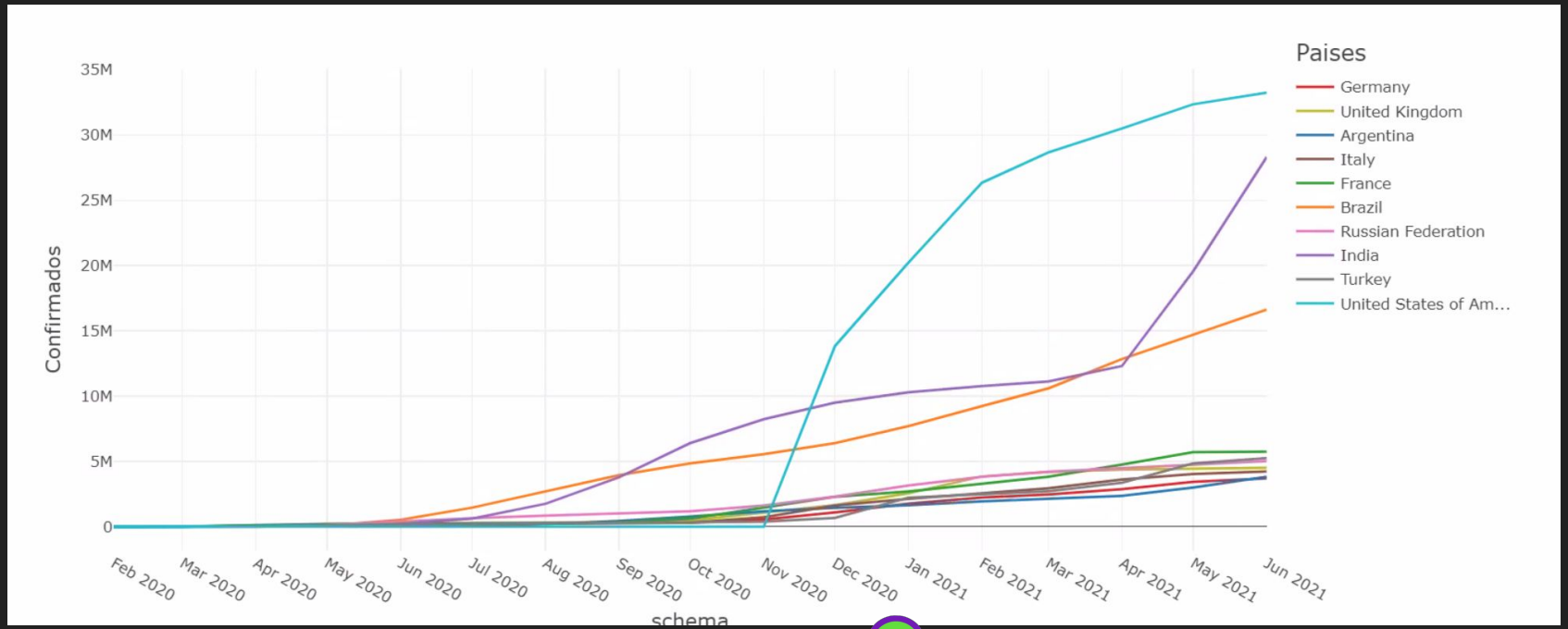
visualizações: Panorama diário com os 10 países com mais mortes



accenture

GAMA ACADEMY

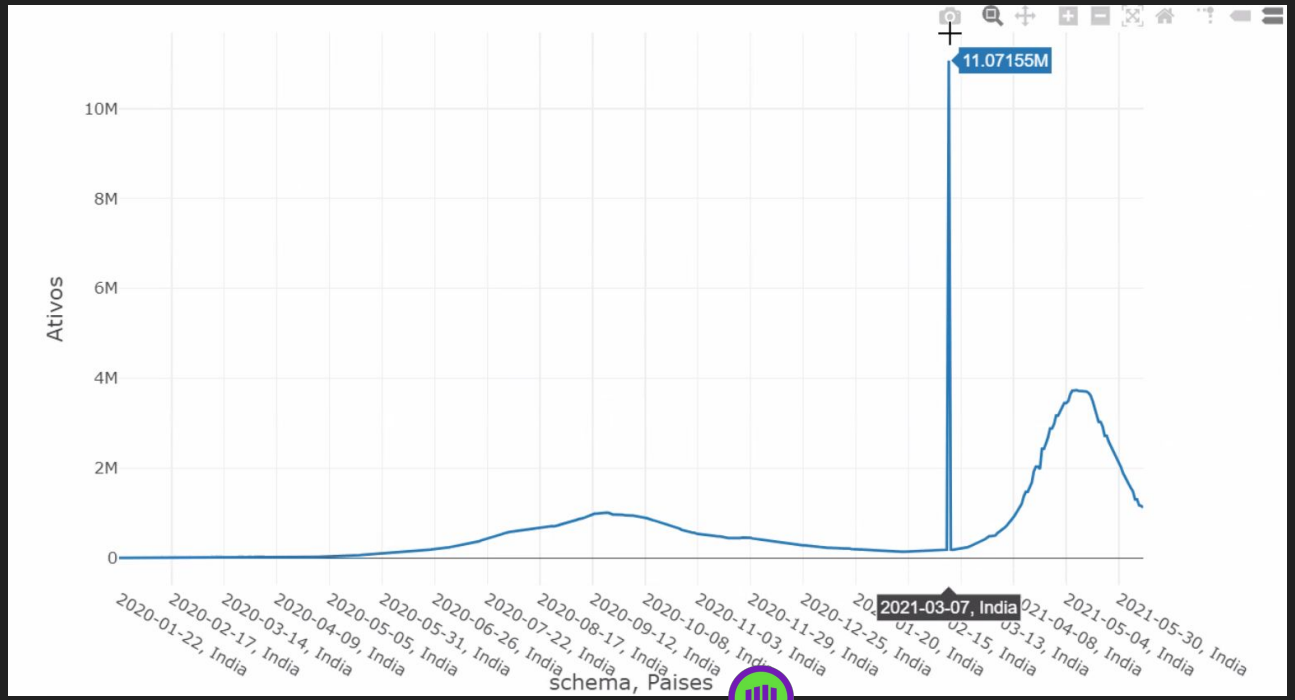
visualizações: Panorama diário com os 10 países com mais casos confirmados



accenture

GAMA ACADEMY

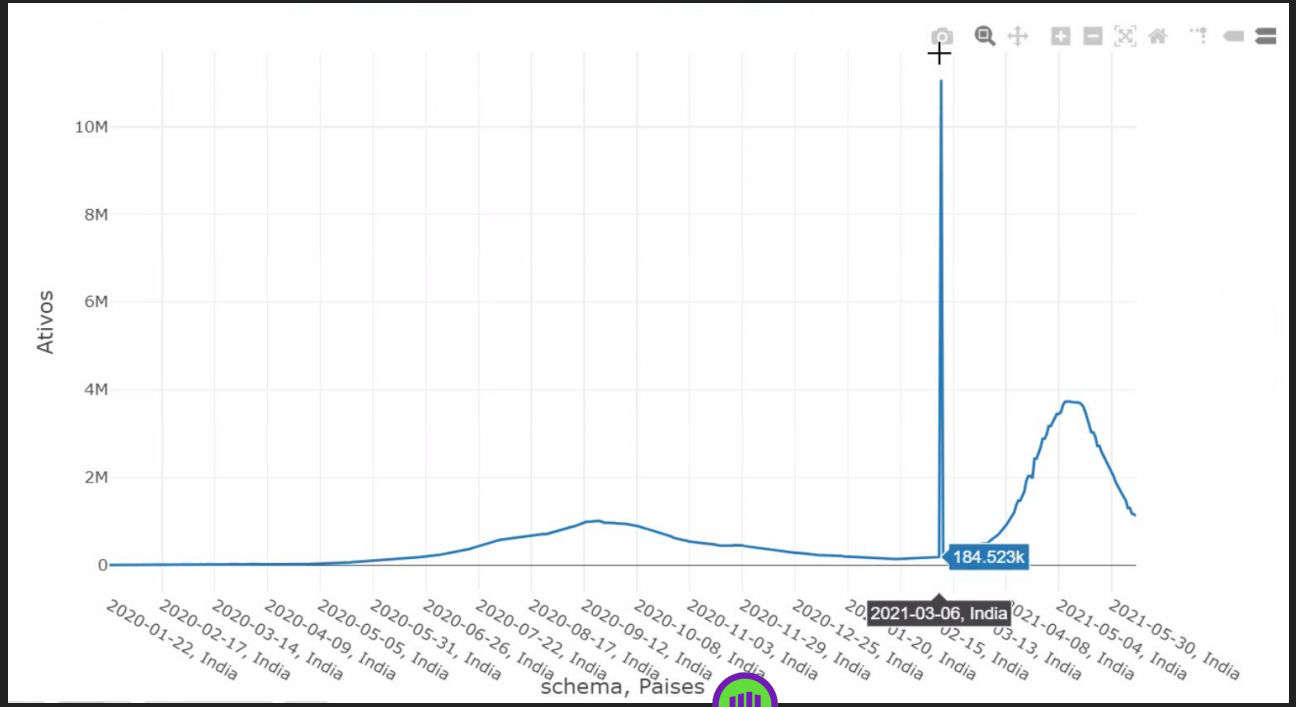
dificuldades encontradas: Outlier casos ativos Índia.



accenture

GAMA ACADEMY

dificuldades encontradas: Outlier casos ativos Índia.



accenture

GAMA ACADEMY

dificuldades encontradas: Erro 429 - Excesso de requisições

```
Sucesso :sweden
Sucesso :ecuador
Status: 429 Erro: israel payload: {"message":"Too Many Requests on /country/israel. Please upgrade to a subscription at https://covid19api.
Status: 429 Erro: peru payload: {"message":"Too Many Requests on /country/peru. Please upgrade to a subscription at https://covid19api.com/
Status: 429 Erro: saint-kitts-and-nevis payload: {"message":"Too Many Requests on /country/saint-kitts-and-nevis. Please upgrade to a subsc
Status: 429 Erro: qatar payload: {"message":"Too Many Requests on /country/qatar. Please upgrade to a subscription at https://covid19api.co
Status: 429 Erro: lebanon payload: {"message":"Too Many Requests on /country/lebanon. Please upgrade to a subscription at https://covid19ap
Sucesso :afghanistan
Sucesso :cambodia
Sucesso :chile
Sucesso :comoros
Sucesso :finland
Sucesso :gabon
Sucesso :niger
Status: 200Erro: turkmenistan payload: []
Sucesso :croatia
Sucesso :djibouti
Status: 429 Erro: dominican-republic payload: {"message":"Too Many Requests on /country/dominican-republic. Please upgrade to a subscriptio
Status: 429 Erro: haiti payload: {"message":"Too Many Requests on /country/haiti. Please upgrade to a subscription at https://covid19api.co
Status: 429 Erro: lithuania payload: {"message":"Too Many Requests on /country/lithuania. Please upgrade to a subscription at https://covid
Status: 429 Erro: nauru payload: {"message":"Too Many Requests on /country/nauru. Please upgrade to a subscription at https://covid19api.co
Status: 429 Erro: united-states payload: {"message":"Too Many Requests on /country/united-states. Please upgrade to a subscription at https
Sucesso :botswana
Sucesso :uruguay
Sucesso :germany
```

```
<mysql.connector.connection.MySQLConnection object at 0x000001C2191EEAC0>
(36078,)
PS C:\Users\filip\Documents\Gama - Accenture\Desafio20210529>
```

dificuldades encontradas: Divergência de dados na API com os demais países

```python
from pyspark.sql.functions import sum as sum_spark, lit

usaCasesDf = usaCasesDf.groupBy('date').agg(

  sum_spark("Confirmed").alias("Confirmed"),
  sum_spark("Deaths").alias("Deaths"),
  sum_spark("Recovered").alias("Recovered"),
  sum_spark("Active").alias("Active"),
)
usaCasesDf = usaCasesDf.withColumn("CountryCode", lit("US"))

usaCasesDf.createOrReplaceTempView("usa_cases_table")

```

accenture

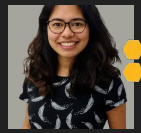
GAMA ACADEMY

obrigadx!

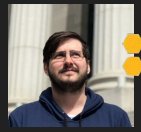
cloudbees

accenture

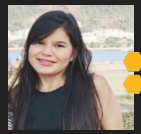
GAMA ACADEMY



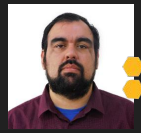
Alessandra Mayumi
linkedin.com/in/alessandra-mayumi-miazato-de-souza-569821137/



Filipe Gonçalves
linkedin.com/in/filipe-esteves-goncalves



Paola Pagotto
linkedin.com/in/paolapagotto



Thiago Franchini
linkedin.com/in/thiago-franchini-dos-santos