Telecomunicações e conectividade móvel no Brasil em 2020 e 2021

Projeto de Conclusão de Curso – Engenharia de Dados BC17

DESENVOLVEDORES

Gabriel Janjacomo

Jum Sahek i

Luma Guimarães

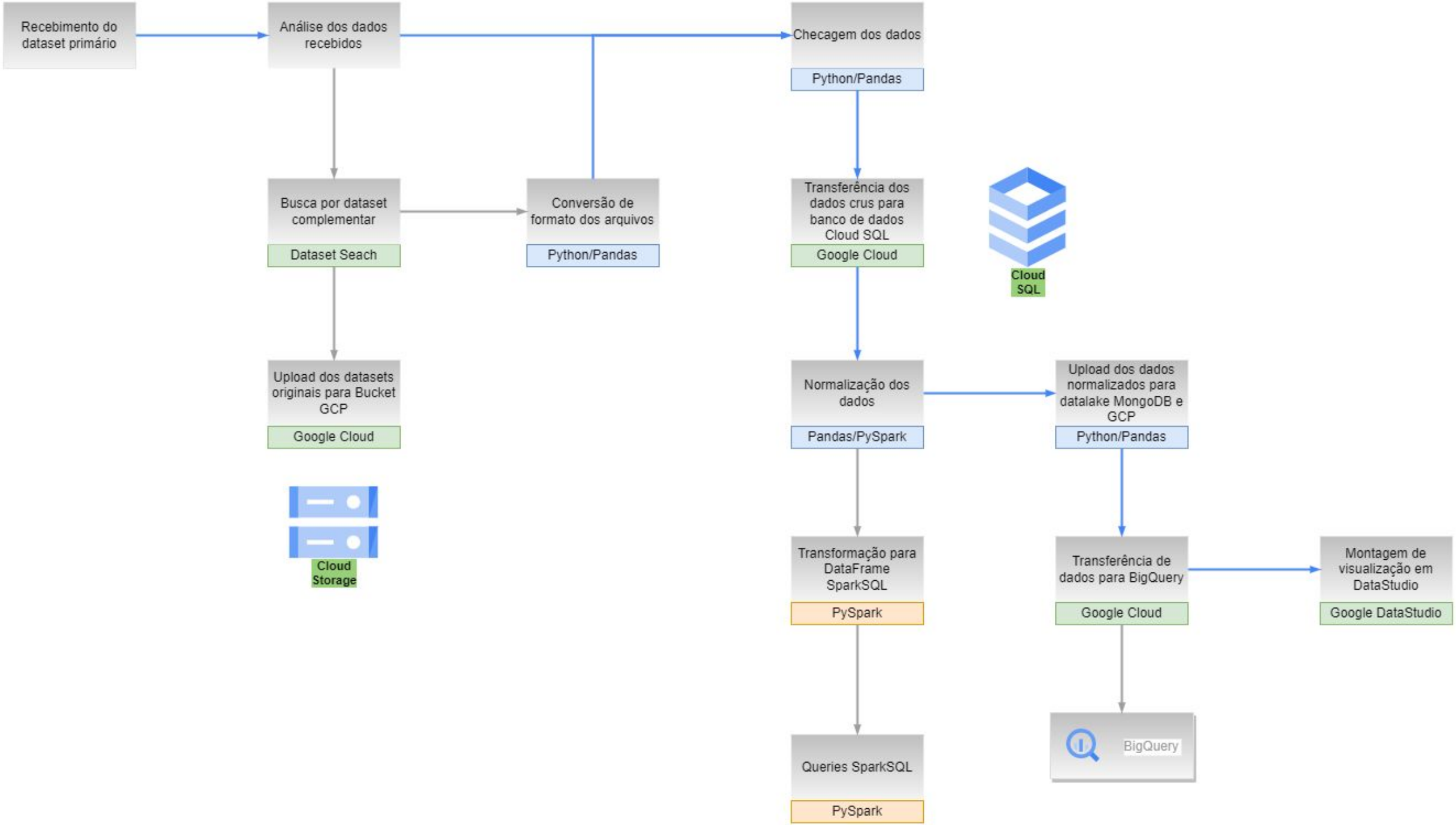
Victor Ribeir o

SUMÁRIO

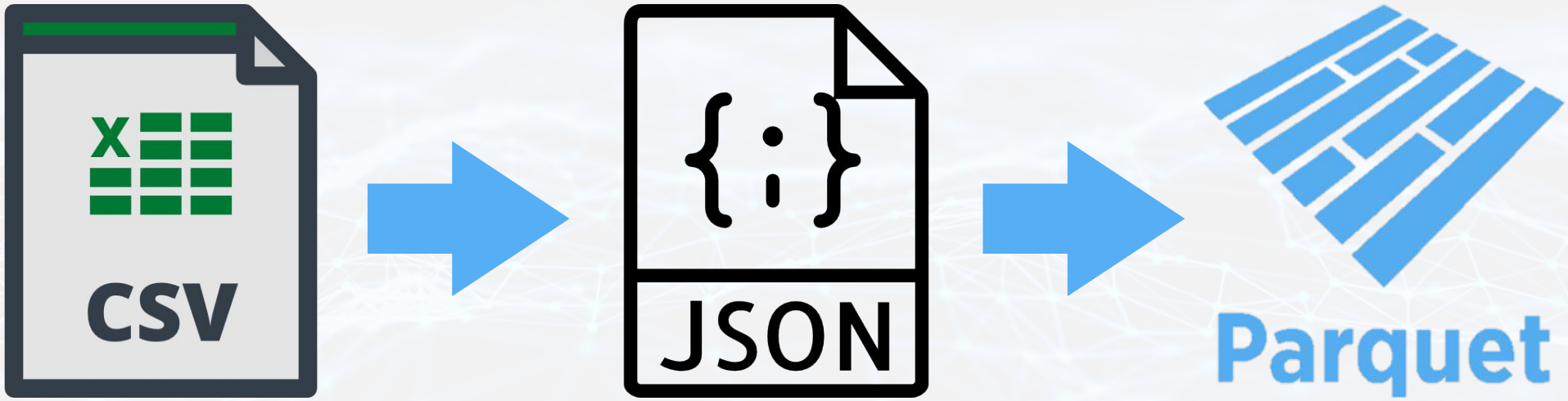
Esse projeto teve por objetivo exercitar e demonstra o aprendizado do conjunto de tecnologias e infraestruturas que foram passados ao longo do bootcamp.

Utilizando uma base fornecida pelos professores, fomos encarregados de analisar, somar, normalizar e tratar dados a fim de gerar insights e finalmente visualizações gráficas.

TECNOLOGIAS UTILIZADAS


Recebimento do dataset primário
Análise dos dados recebidos
Checagem dos dados
Python/Pandas
Busca por dataset complementar
Dataset Seach
Conversão de formato dos arquivos
Python/Pandas
Transferência dos dados crus para banco de dados Cloud SQL
Google Cloud
Cloud SQL
Upload dos datasets originais para Bucket GCP
Google Cloud
Cloud Storage
Normalização dos dados
Pandas/PySpark
Upload dos dados normalizados para datalake MongoDB e GCP
Python/Pandas
Transformação para DataFrame SparkSQL
PySpark
Transferência de dados para BigQuery
Google Cloud
Montagem de visualização em DataStudio
Google DataStudio
Queries SparkSQL
PySpark
BigQuery

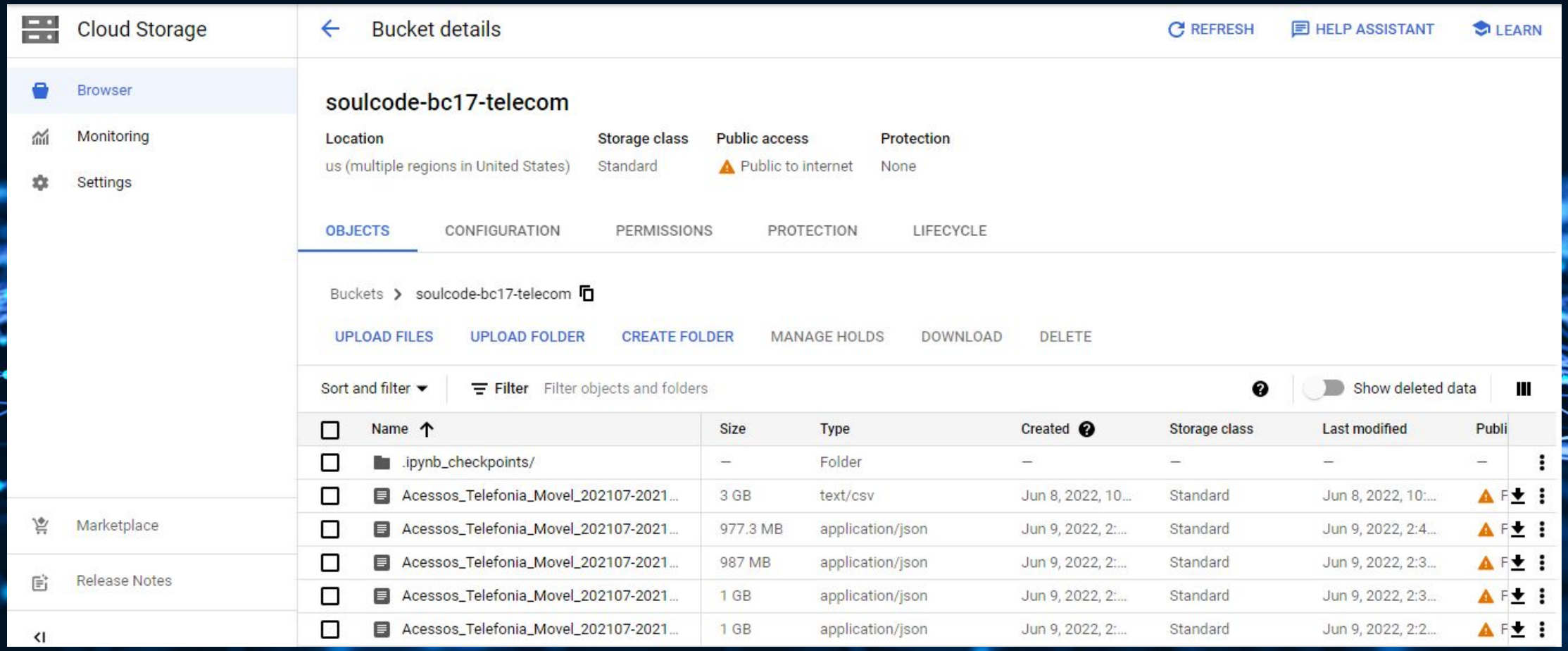
TRANSFORMAÇÕES NOS FORMATOS DE ARQUIVO



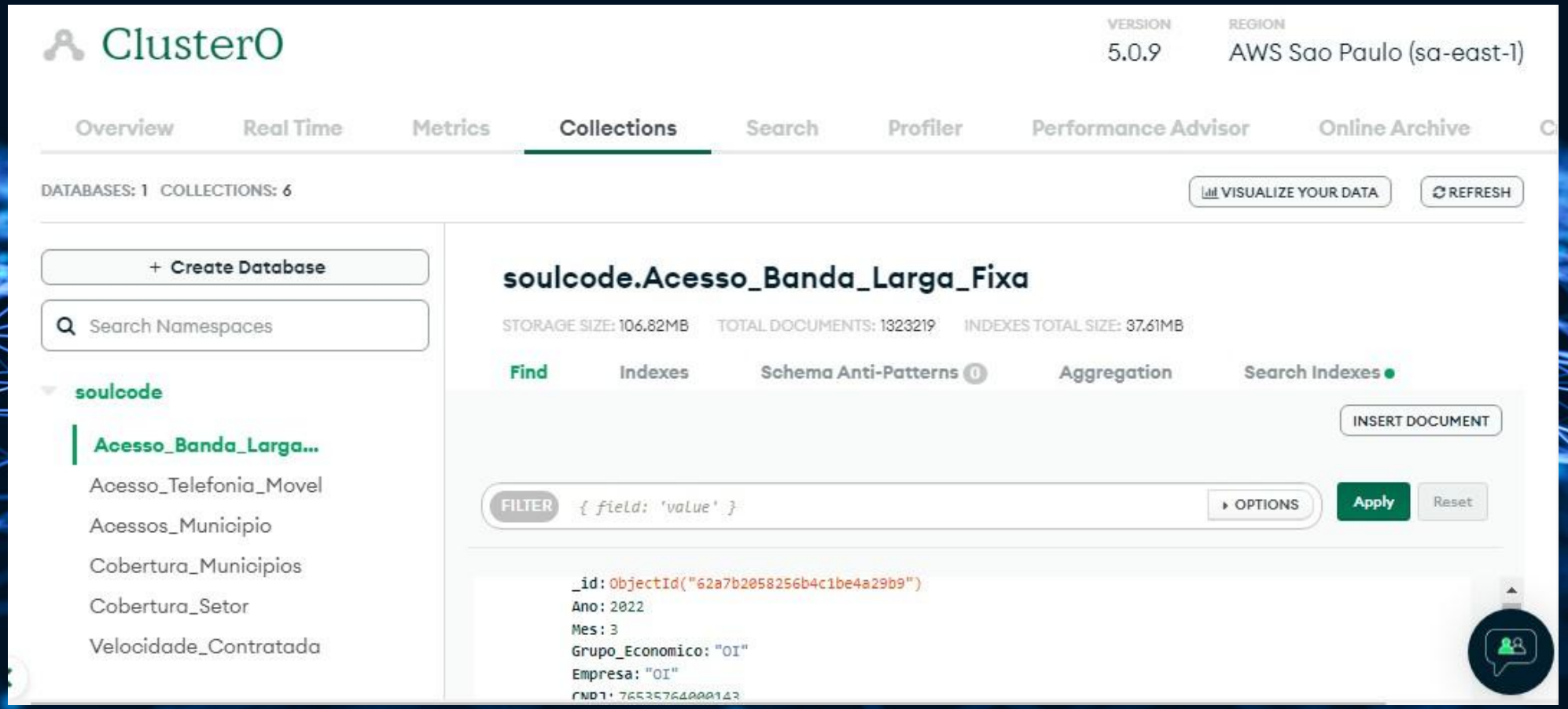
TRANSFORMAÇÕES NOS FORMATOS DE ARQUIVO

```
'''
PUXA ARQUIVO CSV DO BUCKET GCP CONVERTE PARA JSON E SALVA LOCAL
'''

listaNomes = ['Acessos_Telefonia_Movel_202107-202112.csv',
              'Acessos_Telefonia_Movel_202107-202112_Colunas.csv', 'Acessos_Telefonia_Movel_202201-202106_Colunas.csv',
              'Acessos_Telefonia_Movel_202201-202206.csv', 'Acessos_Telefonia_Movel_Pre_Pos_Total.csv',
              'Acessos_Telefonia_Movel_Total.csv', 'Densidade_Telefonia_Movel.csv']

for nomeArquivo in listaNomes:

    GCpath = "gs://soulcode-bc17-telecom/" + nomeArquivo

    if nomeArquivo == 'Acessos_Telefonia_Movel_202107-202112.csv':
      colsList = [0, 1, 3, 6, 7, 11, 12, 13, 14, 15,16]
      with pd.read_csv(GCpath, sep=';', usecols=colsList, chunksize=5000000) as leitor:
        chunkNO = 1
        for chunk in leitor:
          nomeArquivo = listaNomes[0].split('.')
          nomeProcc = (str(nomeArquivo[0]) + '_chunk' + str(chunkNO) + '.json')
          localpath = r"/content/ArquivosOut/" + nomeProcc
          chunk.to_json(localpath)
          chunkNO = chunkNO + 1

    else:
      dfCSV = pd.read_csv(GCpath, sep=';')

      nomeProcc = nomeArquivo.split('.')
      nomeArquivo = nomeProcc[0] + '.json'
      localpath = r"/content/ArquivosOut/" + nomeArquivo
      dfCSV.to_json(localpath)


```

INFRAESTRUTURA DE ARMAZENAMENTO EM NUVEM



INFRAESTRUTURA DE ARMAZENAMENTO EM NUVEM



CRIAÇÃO DOS TRIGGERS EM POSTGRESQL

```
CREATE OR REPLACE FUNCTION public.reg_acessos()
    RETURNS trigger
    LANGUAGE 'plpgsql'
    COST 100
    VOLATILE NOT LEAKPROOF
AS $BODY$
BEGIN
--Trigger operation inserção
IF (TG_OP = 'INSERT') THEN
INSERT INTO Log_changes (log_id, data_criacao, operacao)
VALUES (new.acessos, current_timestamp, ' Operação de inserção.
A linha:  ' || NEW.acessos || 'foi inserida');
RETURN NEW;
-- Trigger operation atualização
ELSIF (TG_OP = 'UPDATE') THEN
INSERT INTO Log_changes (log_id, data_criacao, operacao)
VALUES (NEW.acessos, current_timestamp, 'Operação de atualização.
A linha: ' || NEW.acessos || ' foi atualizada'
|| OLD || ' com ' || NEW.* || '.');
RETURN NEW;
-- Trigger operation exclusão
ELSIF (TG_OP = 'DELETE') THEN
INSERT INTO Log_changes (log_id, data_criacao, operacao)
VALUES (OLD.acessos, current_timestamp, 'Operação de exclusão. A linha: '
|| OLD.acessos || ' foi excluída');
RETURN OLD;
END IF;
RETURN NULL;
END;
$BODY$;

ALTER FUNCTION public.reg_acessos()
    OWNER TO postgres;
```

```
CREATE TRIGGER trigger_acessos
    AFTER INSERT OR DELETE OR UPDATE
    ON public."Acessos_Telefonia_Movel_Pre_Pos_Total"
    FOR EACH ROW
    EXECUTE FUNCTION public.reg_acessos();
```

CONECTIVIDADE VIA PYTHON
Google Cloud Platform

```
# Conectando ao drive para acessar a chave
drive.mount('/content/drive')

# Utilizando a chave para obter as informações de acesso
serviceAccount = '/content/drive/MyDrive/SoulCode/Datasets/macro-mercury-349020-d9ed9a670580.json'
os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = serviceAccount

# Conectando ao bucket
client_bucket = storage.Client()
bucket = client_bucket.get_bucket('soulcode-bc17-telecom')
```

CONECTIVIDADE VIA PYTHON
MongoDB

```python
# Credenciais de acesso
path = 'mongodb://soulcode:a1b2c3@cluster0-shard-00-00.ap3gr.mongodb.net/test?replicaSet=atlas-fg5u6j-shard-0&ssl=true&authSource=admin'
# Configurações do banco
client = MongoClient(path)
db = client['soulcode']
```

ANÁLISE PRELIMINAR DOS DATASETS

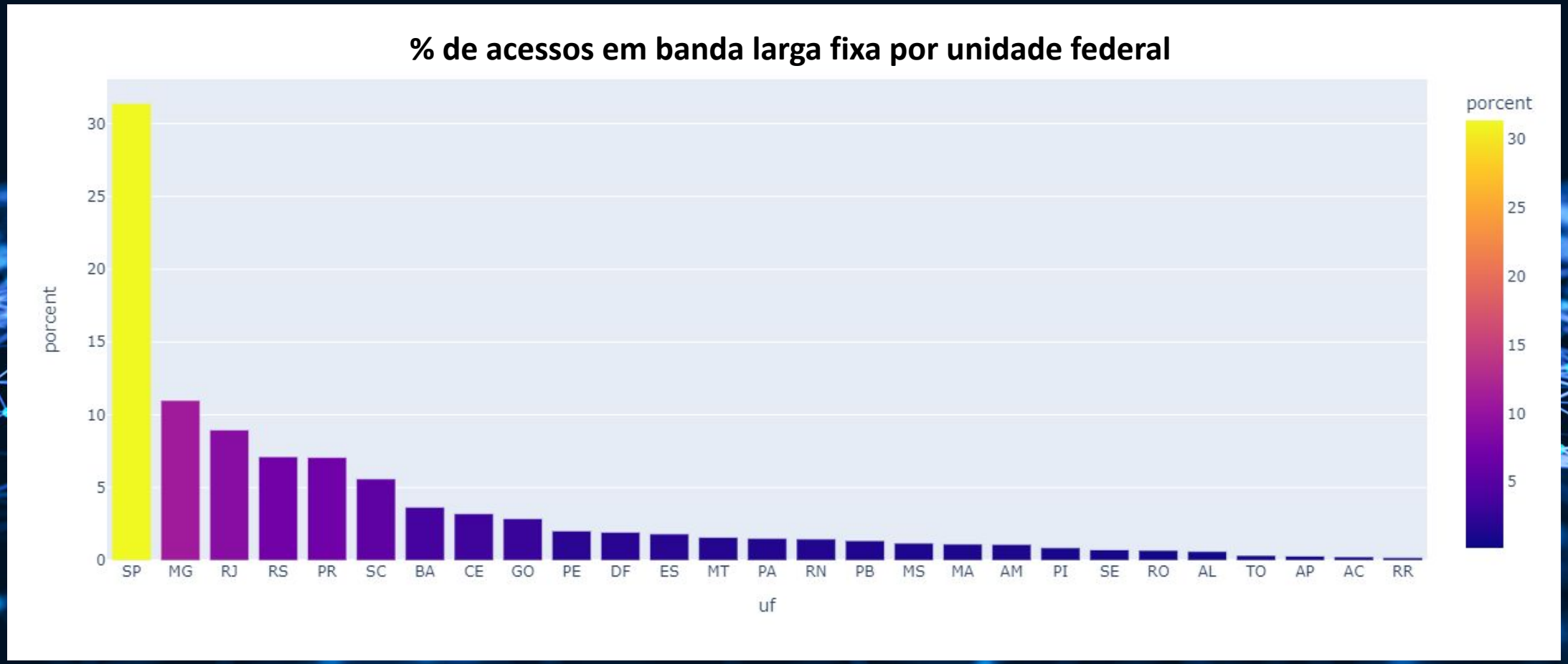
```python
# Quantidade de linhas
df_blf2021.shape
# Tipo de dado de cada coluna
df_blf2021.dtypes
# Valores únicos por coluna
df_blf2021.nunique()
# Valores vazios
df_blf2021.isna().sum()
# Determinando o número de duplicatas
df_blf2021.duplicated().sum()
pd.unique(df_blf2021['Mês'])
```

```python
# Quantidade de linhas
df_blf2022.shape
# Tipo de dado de cada coluna
df_blf2022.dtypes
# Valores únicos por coluna
df_blf2022.nunique()
# Valores vazios
df_blf2022.isna().sum()
# Determinando o número de duplilcatas
df_blf2022.duplicated().sum()
```

ANÁLISE COM PANDAS

```python
df2_extrai=pd.read_csv('gs://telecomunicacao2/Velocidade_Contratada_SCM.csv',sep=";")
# Copy
df2_pd = df2_extrai.copy()

df2_pd.head(2)
# Unique
pd.unique(df2_pd['Mês'])
# Info
df2_pd.info()
# Duplicated
df2_pd.duplicated().sum()
df_pd=df2_pd[df2_pd.duplicated()]

# Drop duplicates
df2_pd = df2_pd.drop_duplicates()
pd.isnull(df2_pd).sum()
# Drop
df2_pd=df2_pd.drop(columns=['municipio_uf'])

```

VISUALIZAÇÃO COM PANDAS



TRANSFORMAÇÃO PARA SPARK

```
df_spark_blf = spark.createDataFrame(df_blf)
df_spark_bfl = (spark.read.format("parquet")
                .option("header", True)
                .option("delimiter", ",")
                .load("https://storage.googleapis.com/soulcode-bc17-telecom/Tratados/Acessos_Banda_Larga_Fixa.parquet")
)
df_spark_bfl=spark.read.parquet("https://storage.googleapis.com/soulcode-bc17-telecom/Tratados/Acessos_Banda_Larga_Fixa.parquet")
```

CHECAGEM DOS DADOS (PYSPARK)

```python
# Número de linhas e colunas
print((df_spark_cob.count(), len(df_spark_cob.columns)))
# Tipo de dados
df_spark_cob.printSchema()
# Contagem de valores únicos por coluna
df_spark_cob.select([F.countDistinct(c).alias(c) for c in df_spark_cob.columns]).show()
# Valores nulos
df_spark_cob.select([F.count(F.when(F.col(c).contains('None') | \
                            F.col(c).contains('NULL') | \
                            (F.col(c) == '' ) | \
                            F.col(c).isNull() | \
                            F.isnan(c), c
                           )).alias(c)
                    for c in df_spark_cob.columns]).show()
# Valores duplicados
total = (df_spark_cob.count()) - (df_spark_cob.dropDuplicates().count())
print(total)
```

```python
# Tipo de dado de cada coluna
df_blf2021.dtypes
```

```
Ano                        int64
Mês                        int64
Grupo Econômico           object
Empresa                   object
CNPJ                       int64
Porte da Prestadora       object
UF                        object
Município                 object
Código IBGE Município      int64
Faixa de Velocidade       object
Tecnologia                object
Meio de Acesso            object
Tipo de Pessoa            object
Acessos                    int64
dtype: object
```

TRANSFORMAÇÃO DE DADOS (PYSPARK)

```python
# Trocando o tipo dos dados de float para inteiro
df_mov_col['2021-07'] = df_mov_col['2021-07'].astype('Int64')
df_mov_col['2021-08'] = df_mov_col['2021-08'].astype('Int64')
df_mov_col['2021-09'] = df_mov_col['2021-09'].astype('Int64')
df_mov_col['2021-10'] = df_mov_col['2021-10'].astype('Int64')
df_mov_col['2021-11'] = df_mov_col['2021-11'].astype('Int64')
df_mov_col['2021-12'] = df_mov_col['2021-12'].astype('Int64')
# Removendo colunas
df_mov_col = df_mov_col.drop('Código Nacional', axis=1)
df_mov_col = df_mov_col.drop('Grupo Econômico', axis=1)
df_mov_col = df_mov_col.drop('Código IBGE Município', axis=1)
# Trocando o tipo dos dados de float para inteiro
df_mov_col2['2022-01'] = df_mov_col2['2022-01'].astype('Int64')
df_mov_col2['2022-02'] = df_mov_col2['2022-02'].astype('Int64')
df_mov_col2['2022-03'] = df_mov_col2['2022-03'].astype('Int64')
```

```python
# Removendo colunas
df_mov_col2 = df_mov_col2.drop('Código Nacional', axis=1)
df_mov_col2 = df_mov_col2.drop('Grupo Econômico', axis=1)
df_mov_col2 = df_mov_col2.drop('Código IBGE Município', axis=1)
# Renomeando colunas devido à compatibilidade com a Bigquery
(df_mov_col.rename(columns = {'2021-07':'_2021-07',
                              '2021-08':'_2021-08',
                              '2021-09':'_2021-09',
                              '2021-10':'_2021-10',
                              '2021-11':'_2021-11',
                              '2021-12':'_2021-12'}, inplace = True)
)

(df_mov_col2.rename(columns = {'2022-01':'_2022-01',
                               '2022-02':'_2022-02',
                               '2022-03':'_2022-03'}, inplace = True)
)
```

NORMALIZAÇÃO DOS DADOS (PYSPARK)

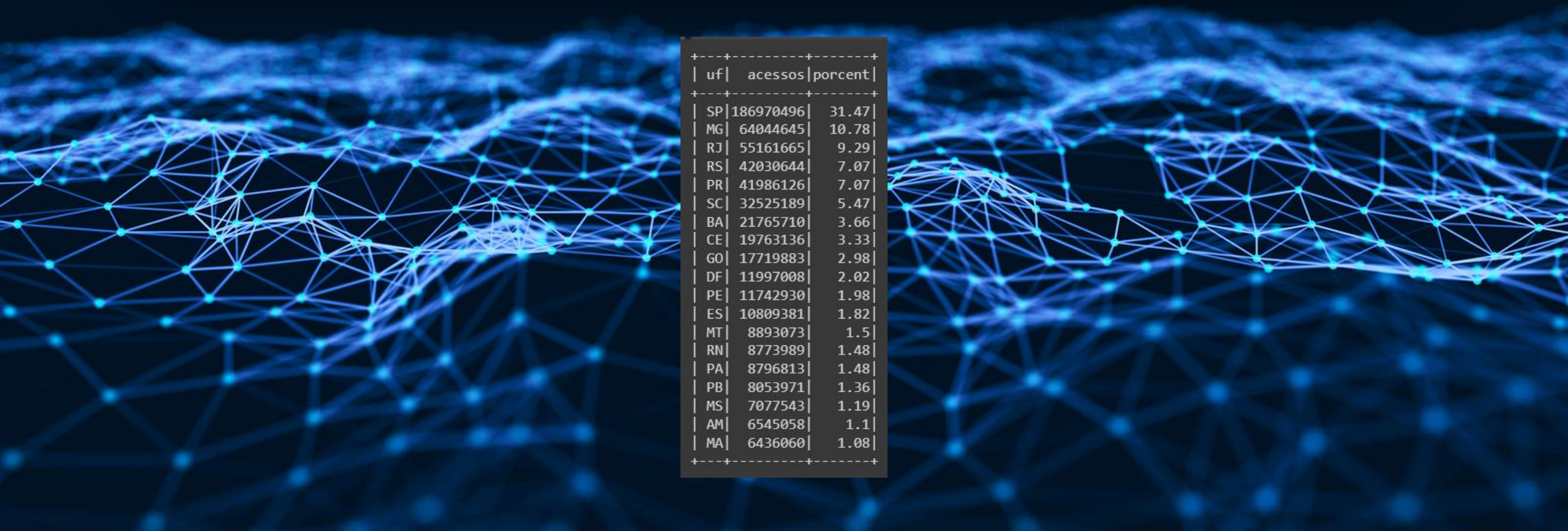
```python
# Eliminando a coluna 'Operadora' pois ela possuiu somente um valor
df_spark_cob = df_spark_cob.drop('Operadora')
# Eliminando a coluna 'Localidade Agregadora' pois ela está vazia
df_spark_cob = df_spark_cob.drop('Localidade Agregadora')
# Substituindo os valores nulos por 'Não informado'
df_spark_cob = df_spark_cob.fillna(0)
# Renomeando as colunas devido à compatibilidade com Spark
df_spark_cob = (df_spark_cob.withColumnRenamed('Código Setor Censitário',"Codigo_Setor_Censitario")
                            .withColumnRenamed('Tipo Setor','Tipo_Setor')
                            .withColumnRenamed('Código Localidade','Codigo_Localidade')
                            .withColumnRenamed('Nome Localidade','Nome_Localidade')
                            .withColumnRenamed('Categoria Localidade','Categoria_Localidade')
                            .withColumnRenamed('Código Município','Codigo_Municipio')
                            .withColumnRenamed('Município','Municipio')
                            .withColumnRenamed('Região','Regiao')
                            .withColumnRenamed('Área (km2)','Area_km2')
                            .withColumnRenamed('Domicílios','Domicilios')
                            .withColumnRenamed('Percentual Cobertura','Percentual_Cobertura')
)
```

UPLOAD PARA BANCO POSTGRESQL/GCP

```
1  from sqlalchemy import create_engine
2
3  # Credenciais de acesso para o banco de dados Postgres sql criado na GCP
4  engine = create_engine("postgresql://postgres:xk5fMiHn74FDBw_@34.132.83.213/projeto_final")
5  df_cob.to_sql('Cobertura_Todas',engine,if_exists='replace',index=False)
```

QUERY COM SPARK

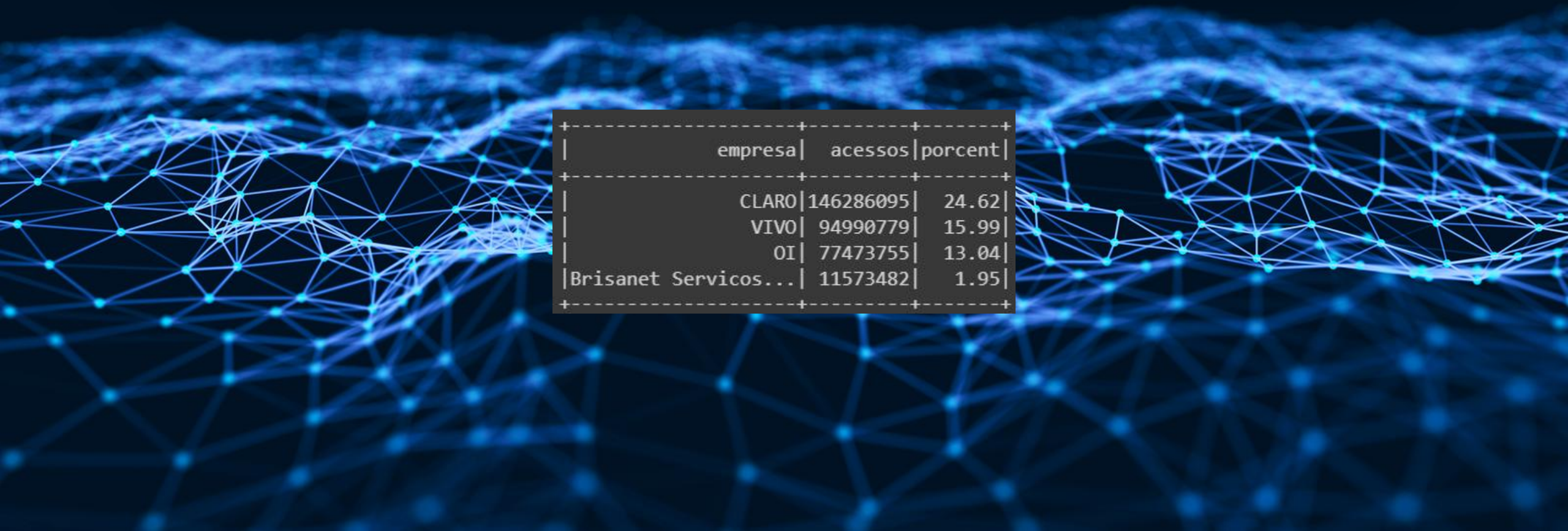
```
1  # Acessos por Estado
2  spark.sql("select uf,sum(acessos) as acessos,round((sum(acessos)/122858439)*100,2) as porcent From bandfixa group by uf order by porcent desc ").show(3)
```



```
+---+---------+-------+
| uf|  acessos|porcent|
+---+---------+-------+
| SP|186970496|  31.47|
| MG| 64044645|  10.78|
| RJ| 55161665|   9.29|
| RS| 42030644|   7.07|
| PR| 41986126|   7.07|
| SC| 32525189|   5.47|
| BA| 21765710|   3.66|
| CE| 19763136|   3.33|
| GO| 17719883|   2.98|
| DF| 11997008|   2.02|
| PE| 11742930|   1.98|
| ES| 10809381|   1.82|
| MT|  8893073|    1.5|
| RN|  8773989|   1.48|
| PA|  8796813|   1.48|
| PB|  8053971|   1.36|
| MS|  7077543|   1.19|
| AM|  6545058|    1.1|
| MA|  6436060|   1.08|
+---+---------+-------+
```

QUERY COM SPARK

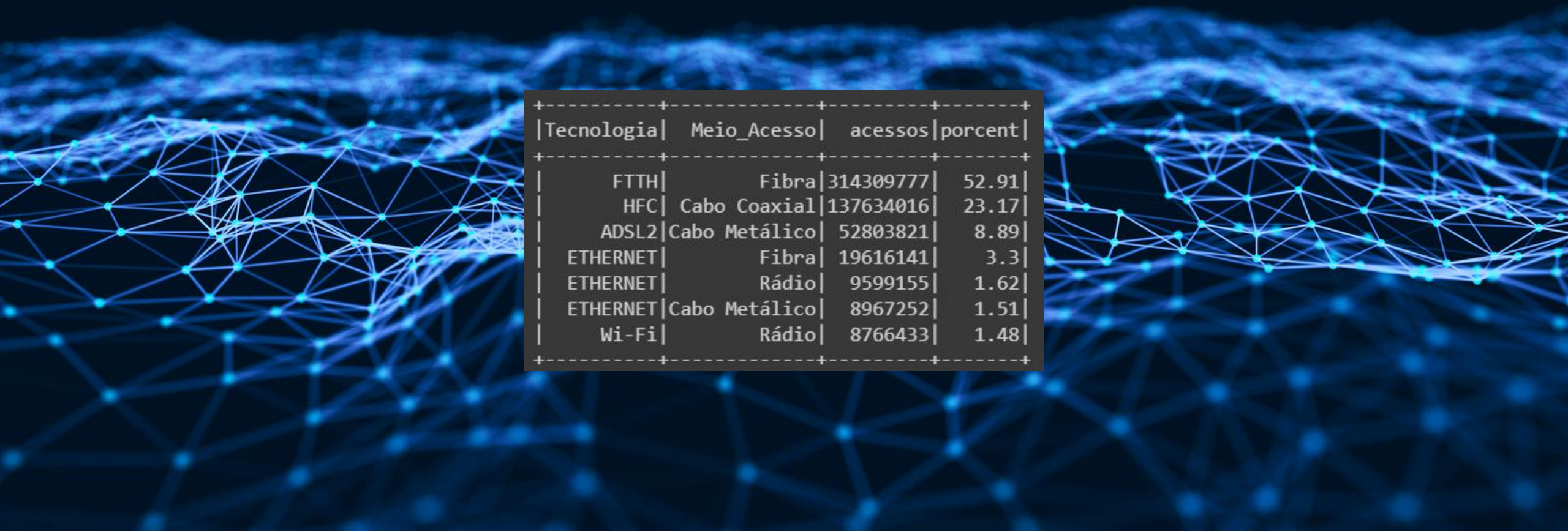
```
1  # Empresas responsáveis pelas maiores quantidades de acesso
2  spark.sql("select empresa,sum(acessos) as acessos,round((sum(acessos)/122858439)*100,2) as porcent From bandfixa group by empresa order by porcent desc ").show(4)
```



```
+--------------------+---------+-------+
|             empresa|  acessos|porcent|
+--------------------+---------+-------+
|               CLARO|146286095|  24.62|
|                VIVO| 94990779|  15.99|
|                  OI| 77473755|  13.04|
|Brisanet Servicos...| 11573482|   1.95|
+--------------------+---------+-------+
```

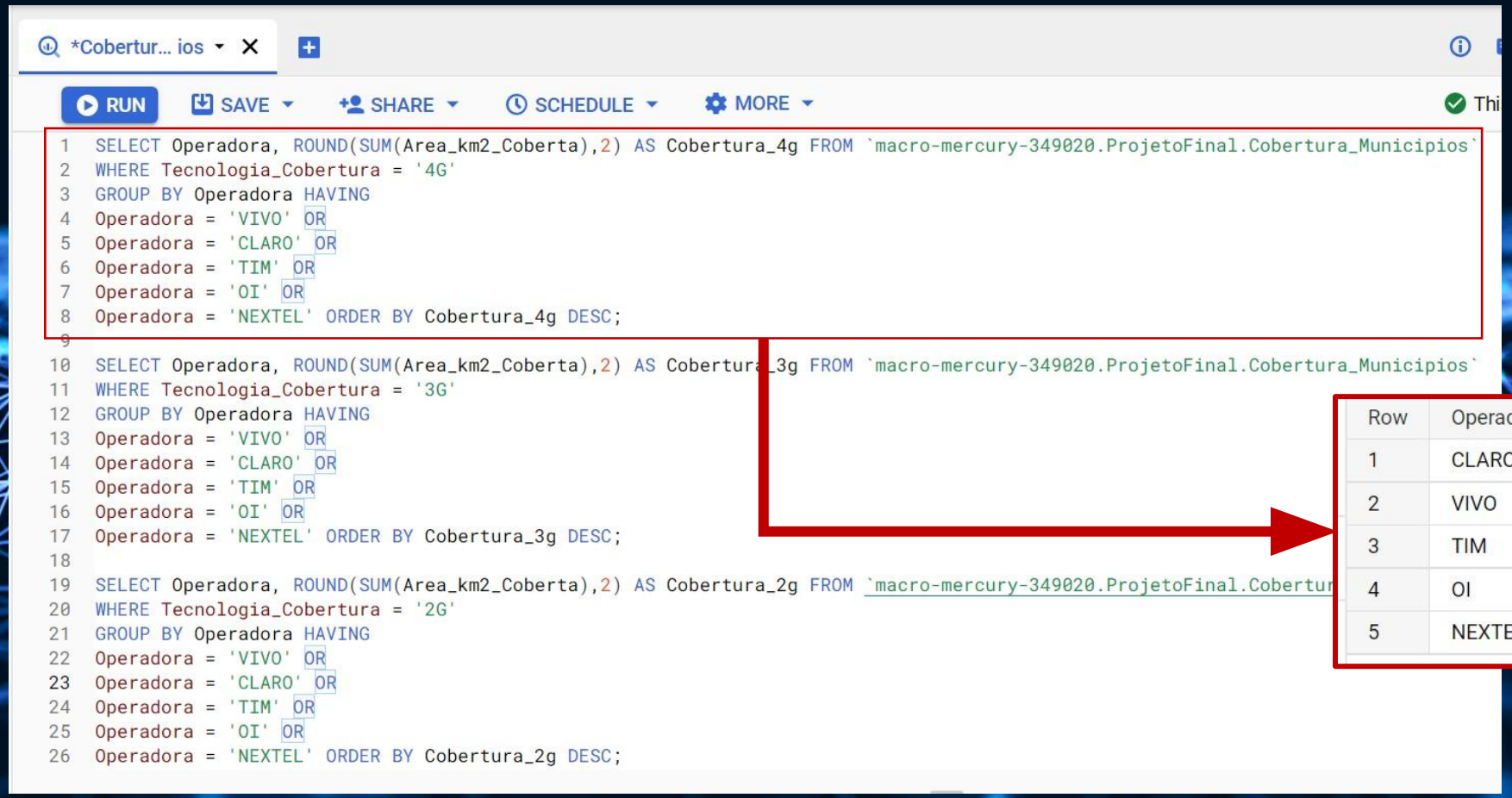
QUERY COM SPARK

```
1  # Tecnologia e meio de acessos
2  spark.sql("select Tecnologia,meioacesso, sum(acessos) as acessos,round((sum(acessos)/122858439)*100,2) as porcent From bandfixa group by tecnologia, meioacesso  order by porcent desc ").show(7)
```



Tecnologia	Meio_Acesso	acessos	porcent
FTTH	Fibra	314309777	52.91
HFC	Cabo Coaxial	137634016	23.17
ADSL2	Cabo Metálico	52803821	8.89
ETHERNET	Fibra	19616141	3.3
ETHERNET	Rádio	9599155	1.62
ETHERNET	Cabo Metálico	8967252	1.51
Wi-Fi	Rádio	8766433	1.48

BIGQUERY


*Cobertur... ios
RUN
SAVE
SHARE
SCHEDULE
MORE
SELECT Operadora, ROUND(SUM(Area_km2_Coberta),2) AS Cobertura_4g FROM `macro-mercury-349020.ProjetoFinal.Cobertura_Municipios`
WHERE Tecnologia_Cobertura = '4G'
GROUP BY Operadora HAVING
Operadora = 'VIVO' OR
Operadora = 'CLARO' OR
Operadora = 'TIM' OR
Operadora = 'OI' OR
Operadora = 'NEXTEL' ORDER BY Cobertura_4g DESC;
SELECT Operadora, ROUND(SUM(Area_km2_Coberta),2) AS Cobertura_3g FROM `macro-mercury-349020.ProjetoFinal.Cobertura_Municipios`
WHERE Tecnologia_Cobertura = '3G'
GROUP BY Operadora HAVING
Operadora = 'VIVO' OR
Operadora = 'CLARO' OR
Operadora = 'TIM' OR
Operadora = 'OI' OR
Operadora = 'NEXTEL' ORDER BY Cobertura_3g DESC;
SELECT Operadora, ROUND(SUM(Area_km2_Coberta),2) AS Cobertura_2g FROM `macro-mercury-349020.ProjetoFinal.Cobertur
WHERE Tecnologia_Cobertura = '2G'
GROUP BY Operadora HAVING
Operadora = 'VIVO' OR
Operadora = 'CLARO' OR
Operadora = 'TIM' OR
Operadora = 'OI' OR
Operadora = 'NEXTEL' ORDER BY Cobertura_2g DESC;
Row | Operadora | Cobertura_4g
1 | CLARO | 513038.52
2 | VIVO | 504152.35
3 | TIM | 424294.77
4 | OI | 131109.62
5 | NEXTEL | 69570.68

LIDERANÇA DE COBERTURA MÓVEL NO BRASIL

Row	Operadora	Cobertura
1	VIVO	1560608.61
2	CLARO	1442949.09
3	TIM	885242.64
4	OI	529072.33
5	NEXTEL	279764.12

https://canaltech.com.br/telecom/tim-assume-lideranca-de-operadoras-moveis-no-brasil-apos-integracao-com-oi-movel-216591/

BIGQUERY

```
1  SELECT Regiao, Tipo_Setor, ROUND(100*(SUM(Area_km2*(Percentual_Cobertura/100))/sum(Area_km2)),2) AS
   porcentagem_cobertura FROM `macro-mercury-349020.ProjetoFinal.Cobertura_Setor` WHERE Tecnologia = '2G3G4G' GROUP
   BY Regiao, Tipo_Setor order by Regiao;
```

Press Alt+F1 for Accessibility Options.

Query results

SAVE RESULTS | EXPLORE DATA

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	Regiao	Tipo_Setor	porcentagem_cobertura
1	Centro-oeste	Rural	13.46
2	Centro-oeste	Urbano	83.25
3	Nordeste	Rural	30.85
4	Nordeste	Urbano	98.67
5	Norte	Rural	10.41
6	Norte	Urbano	93.18
7	Sudeste	Rural	36.96
8	Sudeste	Urbano	92.89
9	Sul	Rural	38.14
10	Sul	Urbano	96.4

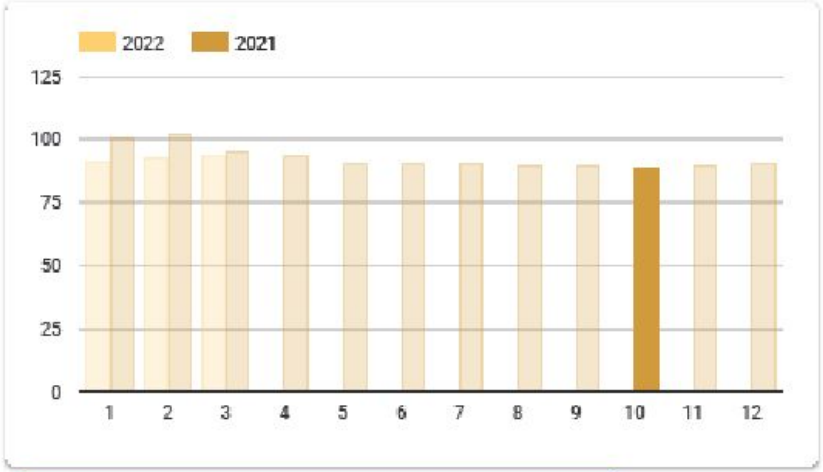
DATA STUDIO

Banda Larga fixa

Total de Acessos

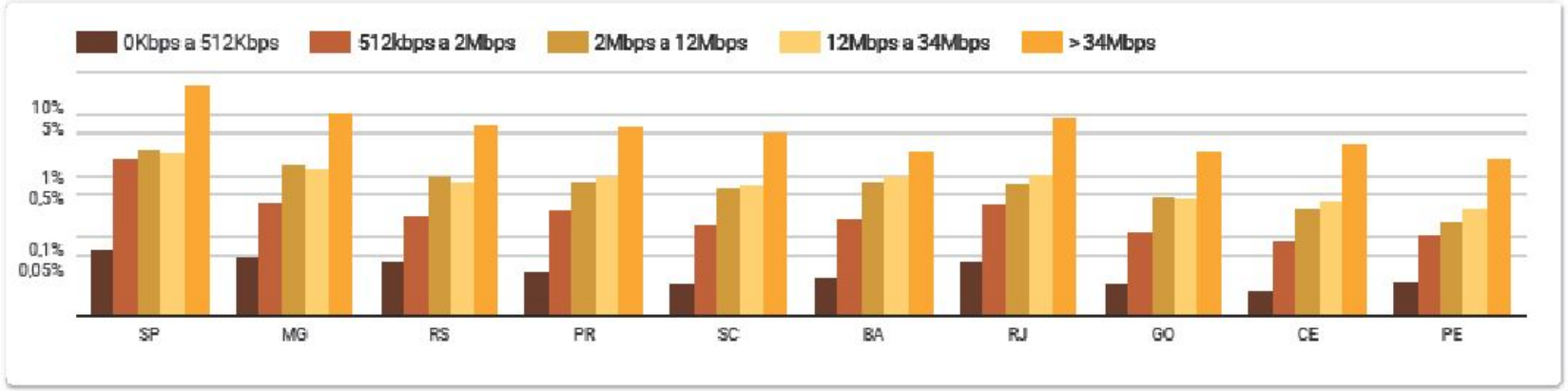
40,8 mi

Acessos por Mês



UF	Acessos
SP	12,8 mi
MG	4,4 mi
RJ	3,7 mi
RS	2,9 mi
PR	2,9 mi
SC	2,2 mi
BA	1,5 mi
CE	1,4 mi
GO	1,2 mi

Faixa de velocidade por estado



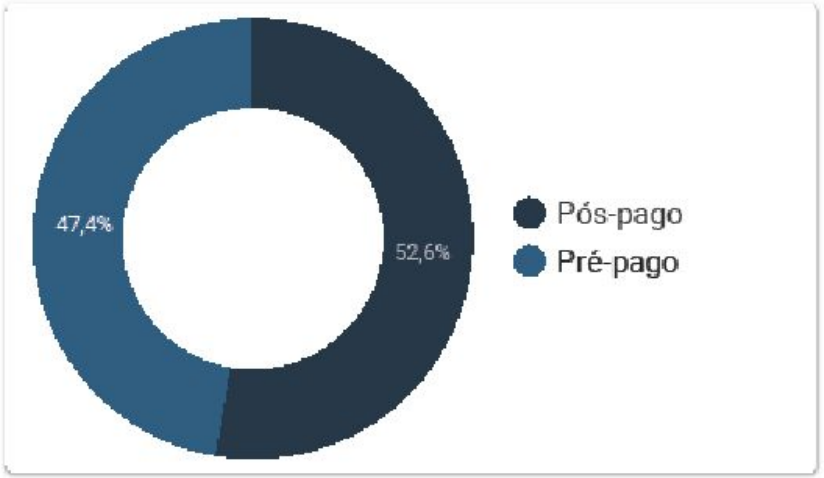
DATA STUDIO

Telefonia Móvel

Total de Acessos

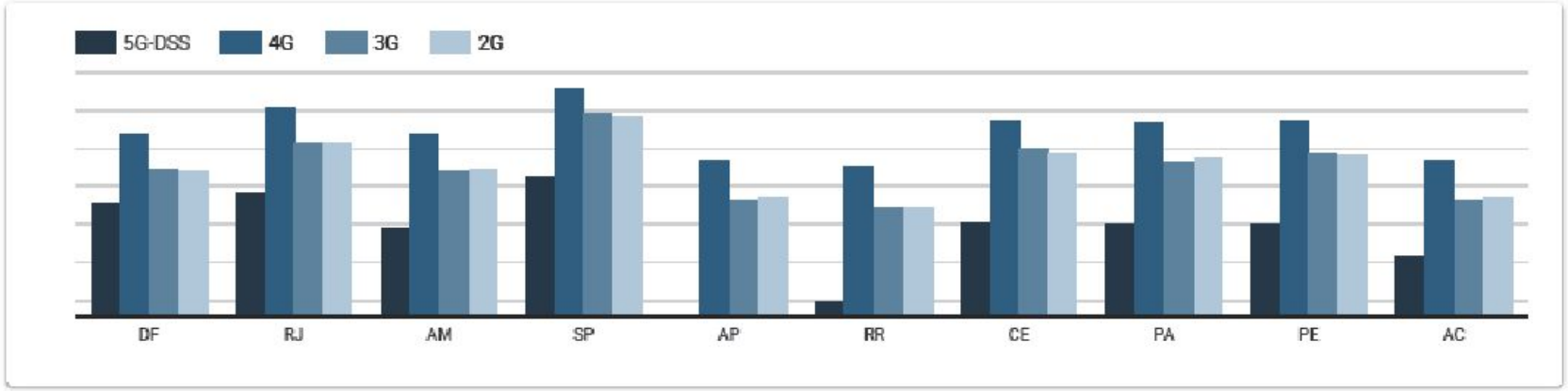
3,4 bi

Modalidade de cobrança



UF	Acessos
SP	1 bi
MG	326,8 mi
RJ	280,2 mi
BA	203,1 mi
RS	184,2 mi
PR	181,4 mi
PE	137,7 mi
CE	131,9 mi
SC	114,7 mi

Tecnologia por estado



RELATÓRIO DE ACOMPANHAMENTO DO SETOR DE TELECOMUNICAÇÕES / ANATEL

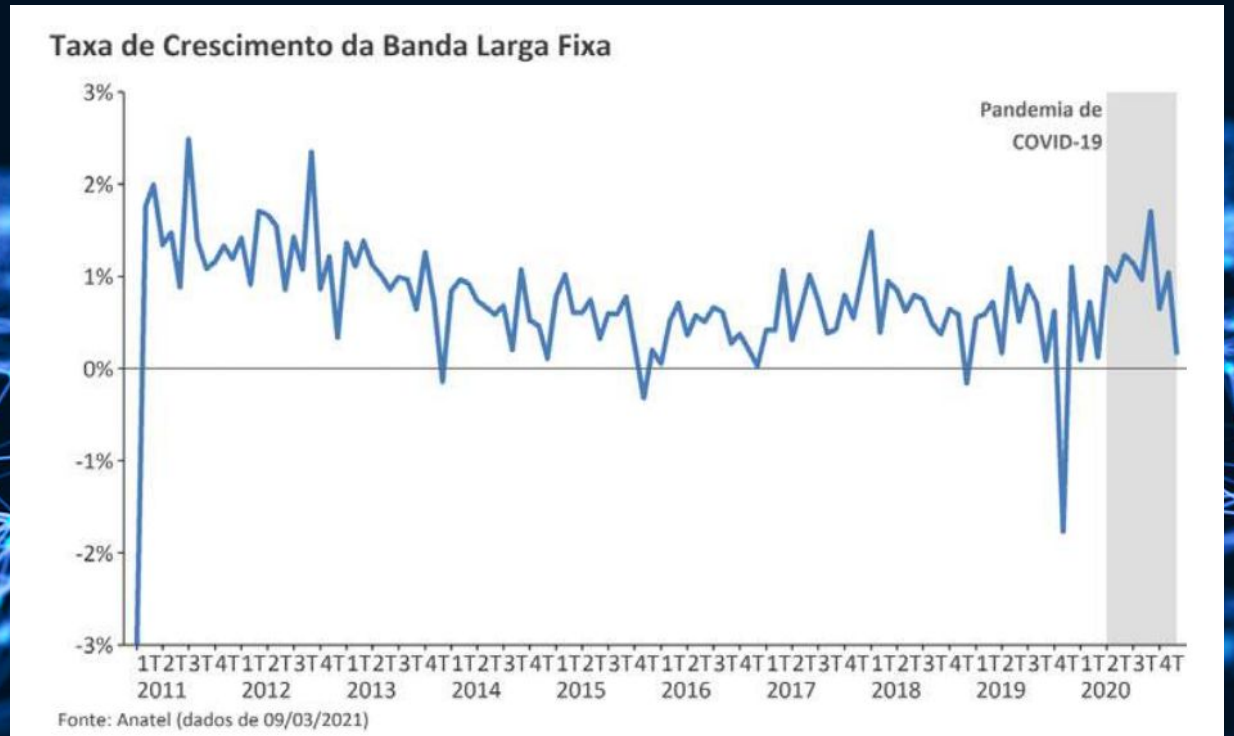


Figura 2 - Evolução das taxas de crescimento trimestral da Banda Larga Fixa, 2010 a 2020

https://www.eventos.momentoeditorial.com.br/wp-content/uploads/2021/05/ATC_Relatorio_SCM_V2.pdf


OBRIGADO