Noticias Projeto Integrador

O Time “E” conseguiu?

Discussão entre os membros?

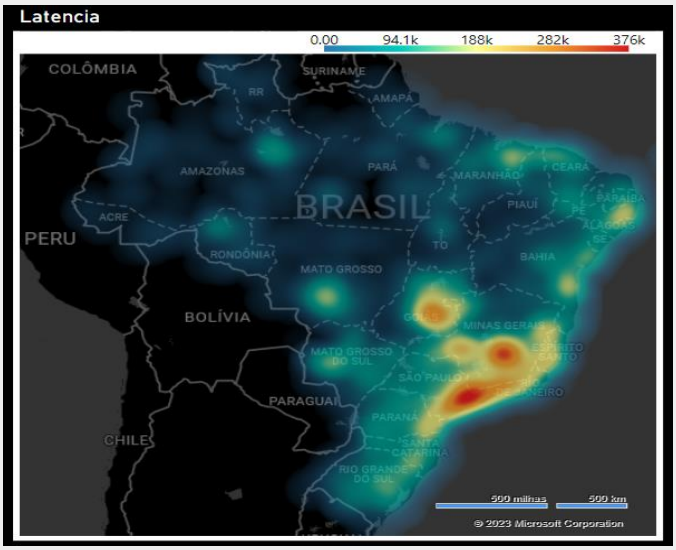
Paulo diz que AWS é muito dificil, enquanto Erik defende configuração e conexão na S3. Will come pipoca assistindo a treta.

Qual arquitetura escolhida?

Qual seria a melhor escolha? A arquitetura escolhida foi ruim ou a equipe que não soube utilizar? Will defende que aprender ferramentas novas leva tempo.

Dashboard não foi conectado no Star Schema?

Apenas uma tabela? Entenda o motivo do PowerBI não ter sido conectado no Star Schema.

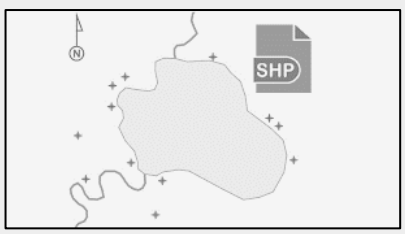


Mapa de latência

Trabalhar com dados shapefiles?

Um shapefile consiste em vários arquivos com extensões diferentes que trabalham juntos para representar e armazenar os dados geoespaciais. Esses arquivos incluem:

- .shp para geometria
- .shx para índice espacial
- .dbf para atributos em formato de tabela
- E também pode haver outros.



Noticias Projeto Integrador

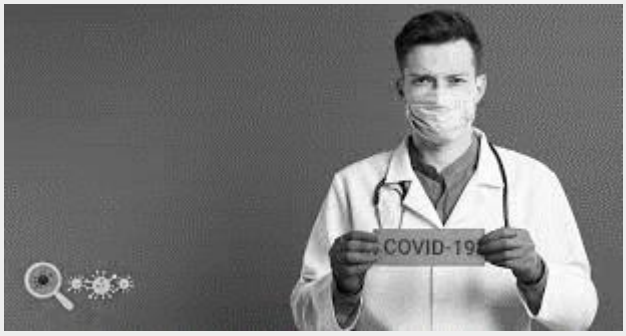
Qual base de dados foi utilizada e quais os motivos da escolha do dataset?

Quais bases de dados foram consideradas para uso?

A Realistic Cyber Defense Dataset:

Séries de cenários de ataque e defesa que foram criados para simular situações reais enfrentadas por profissionais de segurança cibernética.

“Podemos estudar padrões de ataque ou identificar vulnerabilidades” – Defende Erik, estudante de engenharia de dados e big data e empresário nas horas vagas.



COVID-19 Harmonized Data

“Eu não aguento mais coisa sobre COVID, POR FAVOR NÃO!!” – Paulo desabafa, indignado sobre essa base de dados ter sido considerada para escolha.

Speed Test by Ookla – Base de dados vencedora, cria vida e comemora a escolha do time E

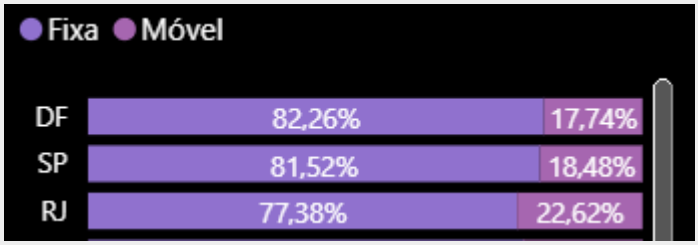
A base de dados "Speedtest by Ookla Global Fixed and Mobile Network Performance Map" é um conjunto de dados que compila informações sobre a velocidade da internet em redes fixas e móveis em todo o mundo.

Essa base de dados é fornecida pela Ookla, uma empresa especializada em testes de velocidade de internet e análise de desempenho de redes.

Noticias Projeto Integrador

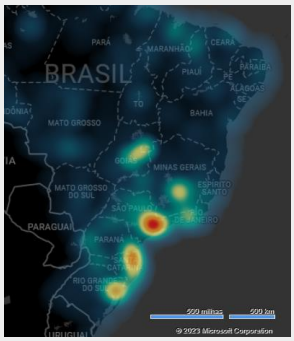
As 3 questões bases do projeto

- Que tipo de internet é mais usada? Fixa ou móvel?



- O uso da internet muda dependendo do período do ano?

- Quais os estados com a maior velocidade de internet?



Noticias Projeto Integrador

Conhecendo as fontes de dados utilizadas

Speedtest Ookla (Móvel e Fixa)

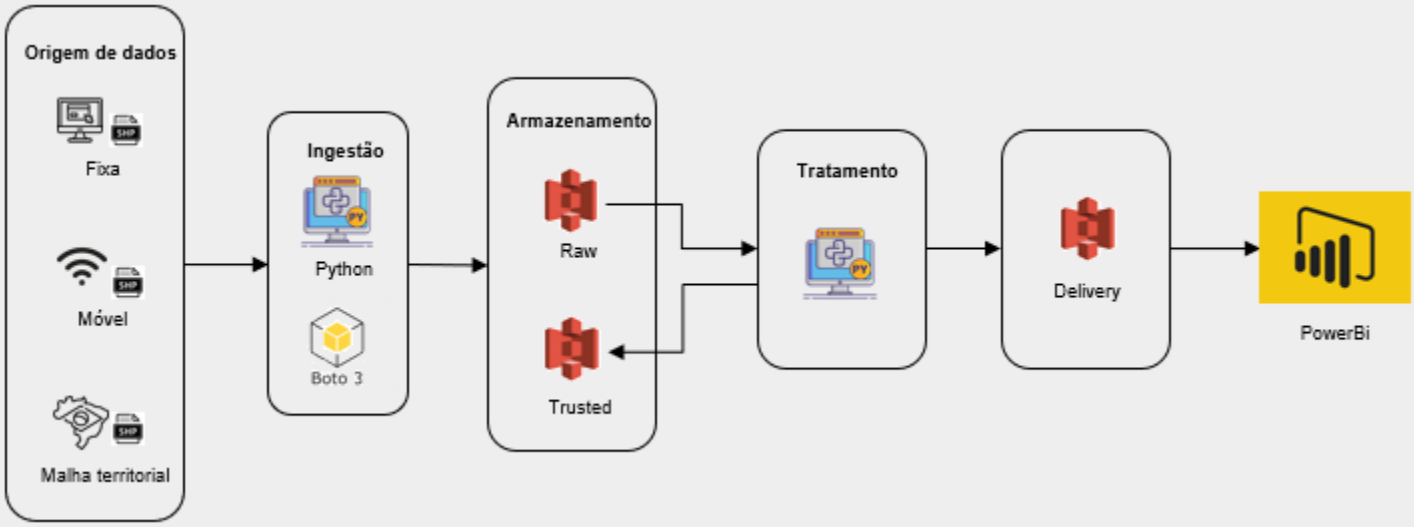
Campo	Tipo	Descrição
avg_d_kbps	Integer	A velocidade média de download de todos os testes realizados no bloco, representada em kbps por segundo.
avg_u_kbps	Integer	A velocidade média de upload de todos os testes realizados no bloco, representada em kbps por segundo.
avg_lat_ms	Integer	A latência média de todos os testes realizados no bloco, representada em milissegundos
tests	Integer	O número de testes realizados no bloco.
devices	Integer	O número de dispositivos exclusivos que contribuem com testes no bloco.
quadkey	Text	Código Quadkey
Geometry	Geometry	conjunto de pontos espaciais que descrevem a forma e a localização dos objetos geográficos

Malha territorial do Brasil (IBGE)

Coluna	Tipo	Descrição
geometry	geometry	conjunto de pontos espaciais que descrevem a forma e a localização dos objetos geográficos
cd_mun	int	código do município
nm_mun	varchar	nome do município
sigla_uf	varchar	sigla do município
area_km2	int	tamanho da área em km2

Noticias Projeto Integrador

Arquitetura de dados

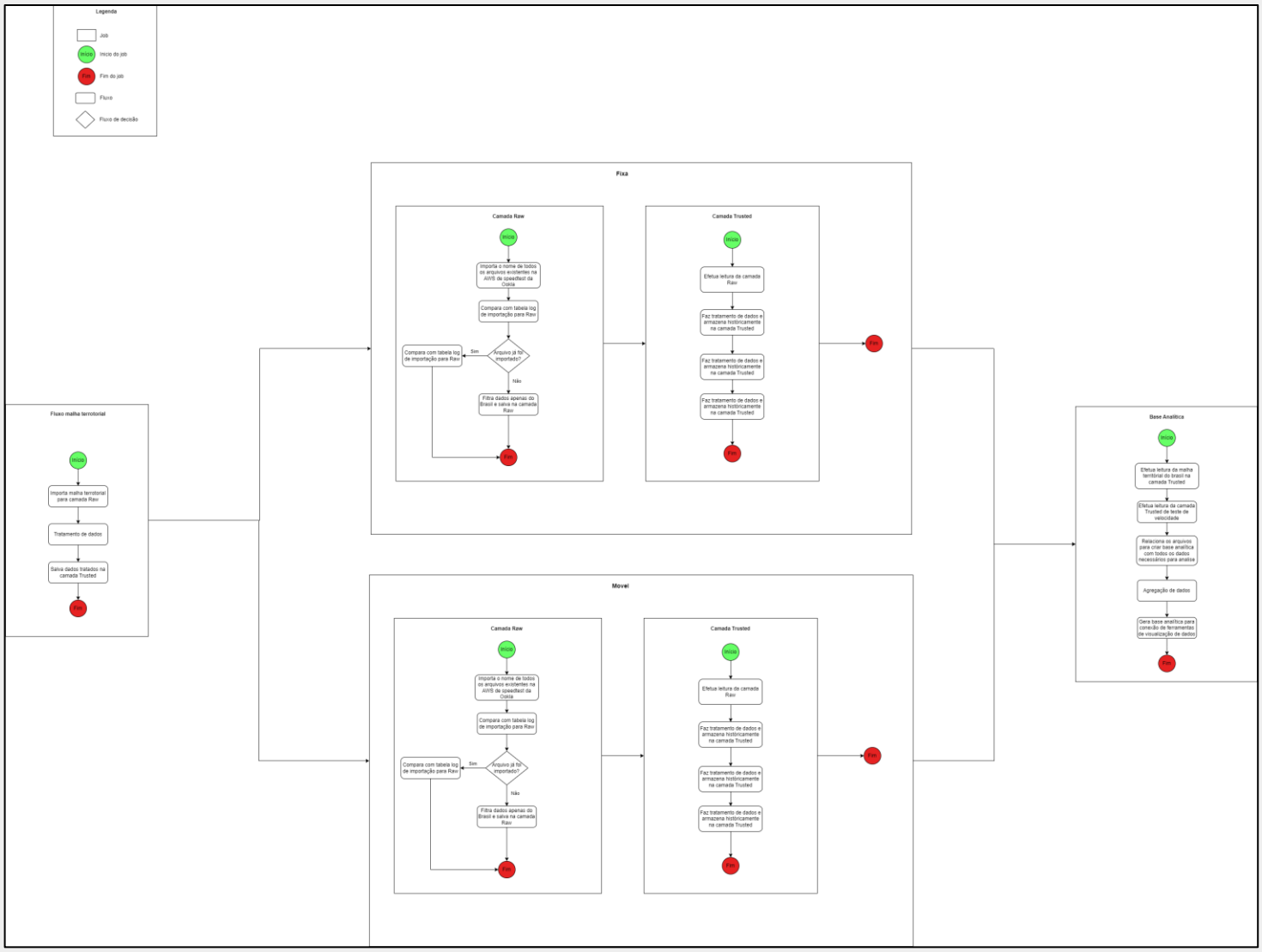


2023-06-25

Primeira Edição

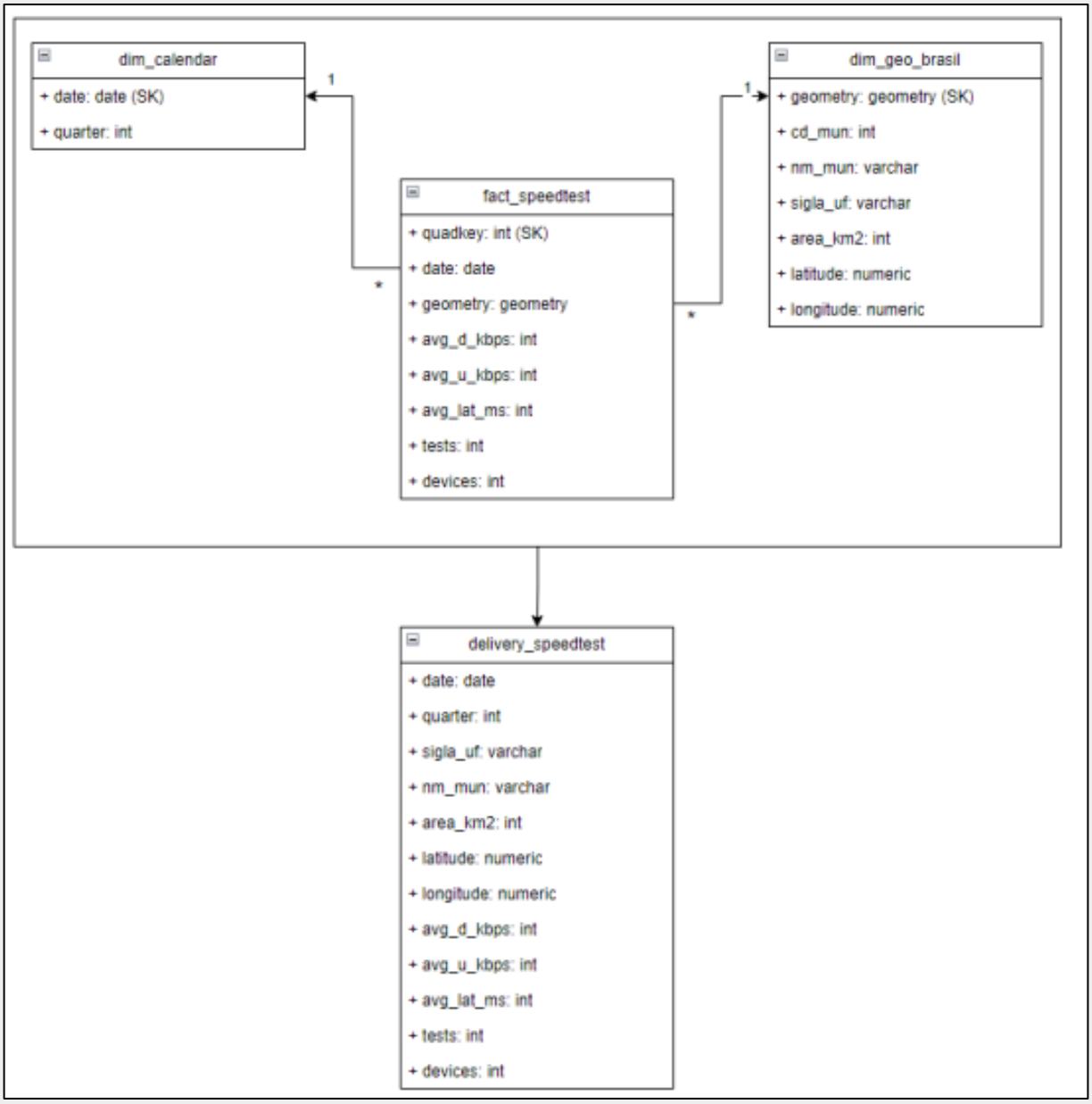
Noticias Projeto Integrador

Fluxo de dados



Noticias Projeto Integrador

Por que o modelo Star schema não foi conectado no PowerBi?



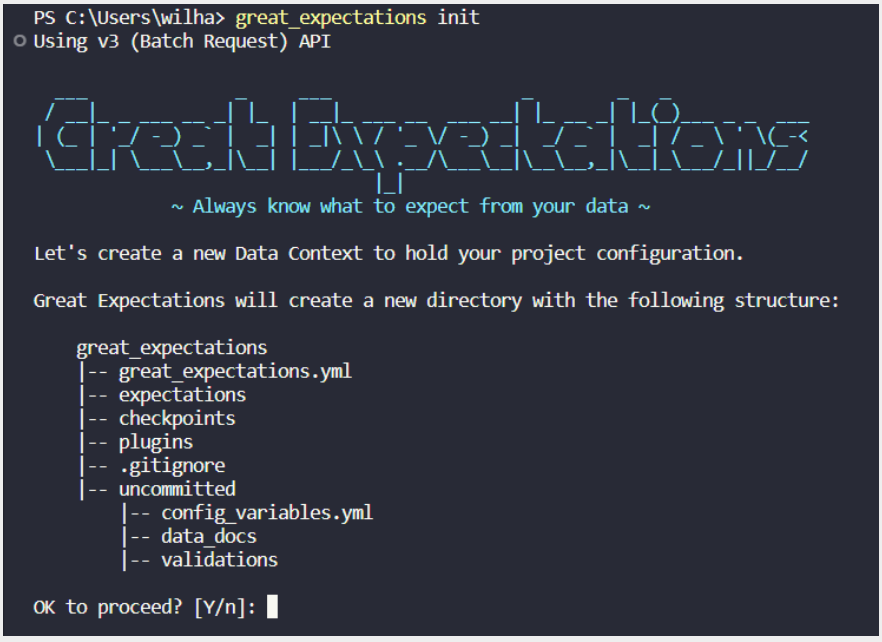
Noticias Projeto Integrador

Qualidade de dados

A qualidade de dados foi apenas um teste realizado antes do início do projeto?

Paulo (Jogador profissional de Apex Legends e engenheiro de dados nas horas vagas) defende que a qualidade de dados trata-se de um processo contínuo que deve ser realizado antes, durante e depois do processamento dos dados.

Erik, concorda e sugere utilizar Great Expectations para a qualidade de dados.



Noticias Projeto Integrador

Plano de Teste

Testes na camada trusted para os dados sobre de geolocalização.

Testes realizados após transformar os dataset de formato shapefile em csv

```
#importando bibliotecas necessárias para os testes
import pandas as pd
import great_expectations as gx

#Criar o df tax_df com as informacoes doa arquivo us_tax_data_2016.csv
path ='./dados/validacoes_trusted/trusted_geo_brasil.csv'
df_geo = pd.read_csv(path, delimiter=';', encoding="ansi")
```

```
# verificacao do layout do dataset
print(df_geo.head(3))
```

```
    CD_MUN                NM_MUN SIGLA_UF distancia_
0  1100015  Alta Floresta D'Oeste       RO        7km  \
1  1100023              Ariquemes       RO        4km
2  1100031                 Cabixi       RO        1km
```

Crie uma expectation para utilizar no great expectation para relizações de algumas validações:

```
df_gx = gx.dataset.PandasDataset(df_geo)
df_gx.head(3)
```

	CD_MUN	NM_MUN	SIGLA_UF	distancia_	geometry
0	1100015	Alta Floresta D'Oeste	RO	7km	POLYGON ((-62.0080637729999 -12.1337851349999,...
1	1100023	Ariquemes	RO	4km	POLYGON ((-63.179325881 -10.139244048, -63.177...
2	1100031	Cabixi	RO	1km	POLYGON ((-60.524079765 -13.32137253, -60.3716...

Noticias Projeto Integrador

1 Verifica se o código município (CD_MUN) contém apenas valores únicos.

Usar a função `expect_column_values_to_be_unique` para definir essa expectativa

```
validation_result = df_gx.expect_column_values_to_be_unique(column = 'CD_MUN')
validation_result
```


```
{
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  },
  "meta": {},
  "result": {
    "element_count": 5572,
    "missing_count": 0,
    "missing_percent": 0.0,
    "unexpected_count": 0,
    "unexpected_percent": 0.0,
    "unexpected_percent_total": 0.0,
    "unexpected_percent_nonmissing": 0.0,
    "partial_unexpected_list": []
  },
  "success": true
}
```


```
if validation_result['success']:
    print('A coluna de código de município contem apenas valores únicos')
else:
    print('A coluna código de município possui valores duplicados')
```

```
A coluna de código de município contem apenas valores únicos
```

Noticias Projeto Integrador

2 Validação de valores nulos em uma coluna

```
validation_result = df_gx.expect_column_values_to_not_be_null(column= 'CD_MUN')
validation_result
```


```
{
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  },
  "meta": {},
  "result": {
    "element_count": 5572,
    "unexpected_count": 0,
    "unexpected_percent": 0.0,
    "unexpected_percent_total": 0.0,
    "partial_unexpected_list": []
  },
  "success": true
}
```


```
if validation_result['success']:
    print('A coluna código do município não possui valores nulos')
else:
    print('A coluna código do município possui valores nulos, verificar a necessidade de realizar tratamento no campo')
```

```
A coluna código do município não possui valores nulos
```

Noticias Projeto Integrador

3 Validação de siglas de UF incorretos

```
# Lista de siglas de UF válidas no Brasil
uf_validas = ['AC', 'AL', 'AM', 'AP', 'BA', 'CE', 'DF', 'ES', 'GO', 'MA',
              'MG', 'MS', 'MT', 'PA', 'PB', 'PE', 'PI', 'PR', 'RJ', 'RN',
              'RO', 'RR', 'RS', 'SC', 'SE', 'SP', 'TO']

# Filtrar valores de 'uf' que não estão na lista de UF válidas
uf_invalida = df_geo[~df_geo['SIGLA_UF'].isin(uf_validas)]

if uf_invalida.empty:
    print('A coluna sigla UF não possui valor(es) incorreto(s) na coluna SIGLA_UF.')
else:
    print("Existe(m) valor(es) incorreto(s) na coluna 'uf':")
    print(uf_invalida['SIGLA_UF'].unique())
```

```
A coluna sigla UF não possui valor(es) incorreto(s) na coluna SIGLA_UF.
```

```
uf_invalida_count = uf_invalida['SIGLA_UF'].count()
print("Quantidade de Siglas de estado incorreta(s):", uf_invalida_count)
```

```
Quantidade de Siglas de estado incorreta(s): 0
```

4 Verifica se existe(m) distância que não esteja(m) em km.

```
# Verificar se não há valores na coluna 'distancia_' que contenham a palavra "km"
distancia_sem_km = df_geo[~df_geo['distancia_'].str.contains('km', case=False, na=False)]

if distancia_sem_km.empty:
    print("Todos os valores na coluna distancia_ contêm a palavra 'km'.")
else:
    print("Existem valores na coluna distancia_ que não contêm a palavra km:")
    print('Valor(es) incorreto(s):')
    print(distancia_sem_km['distancia_'].to_string(index=False))
```

```
Todos os valores na coluna distancia_ contêm a palavra 'km'.
```

Noticias Projeto Integrador

Plano de Teste

Testes na camada trusted para os dados sobre de internet fixa.

Testes realizados após transformar os dataset de formato shapefile em csv

```
#importando bibliotecas necessárias para os testes
import pandas as pd
import great_expectations as gx

#Criar o df tax_df com as informacoes doa arquivo us_tax_data_2016.csv
path ='./dados/validacoes_trusted/trusted_speedtest_fixed.csv'
df_fixed = pd.read_csv(path, delimiter=';', encoding="ansi")
```

```
# verificacao do layout do dataset
print(df_fixed.head(3))
```

```
           quadkey  avg_d_kbps  avg_u_kbps  avg_lat_ms  tests  devices  \
0  2110223123011000      157341       56947          13   1095      259  \
1  2110223121232032      184049      107142           4    120       45
2  2110223123001110      176971       90063          23    221       52

   index_righ   CD_MUN   latitude  longitude        date  quarter network_ty  \
0        3174  3205309 -20.305993 -40.295105  2022-01-01        1      fixed  \
1        3174  3205309 -20.280232 -40.328064  2022-01-01        1      fixed
2        3174  3205309 -20.305993 -40.350037  2022-01-01        1      fixed

                                      geometry
0  POLYGON ((-40.2978515625 -20.3034175184893, -4...
1  POLYGON ((-40.330810546875 -20.2776560563371, ...
2  POLYGON ((-40.352783203125 -20.3034175184893, ...
```

Crie uma expectation para utilizar no great expectation para relizações de algumas validações:

```
df_gx = gx.dataset.PandasDataset(df_fixed)
df_gx.head(3)
```

	quadkey	avg_d_kbps	avg_u_kbps	avg_lat_ms	tests	devices	index_righ	CD_MUN	latitude	longitude	date
0	2110223123011000	157341	56947	13	1095	259	3174	3205309	-20.305993	-40.295105	2022-01-01
1	2110223121232032	184049	107142	4	120	45	3174	3205309	-20.280232	-40.328064	2022-01-01
2	2110223123001110	176971	90063	23	221	52	3174	3205309	-20.305993	-40.350037	2022-01-01

Noticias Projeto Integrador

1 Verifica se o código município (CD_MUN) contém apenas valores únicos.

Usar a função `expect_column_values_to_be_unique` para definir essa expectativa

```
validation_result = df_gx.expect_column_values_to_be_unique(column = 'CD_MUN')
validation_result
```


```
{
  "result": {
    "element_count": 1807570,
    "missing_count": 0,
    "missing_percent": 0.0,
    "unexpected_count": 1807569,
    "unexpected_percent": 99.99994467710795,
    "unexpected_percent_total": 99.99994467710795,
    "unexpected_percent_nonmissing": 99.99994467710795,
    "partial_unexpected_list": [
      3205309,
      3205309,
      3205309,
      3205309,
      3205309,
      3205309,
      3205309,
      3205309,
      3205309,
      3205309,
      3205309,
      3205309,
      3205309,
      3205309,
      3205309,
      3205309,
      3205309,
      3205309,
      3205309,
      3205309,
      3205309
    ]
  },
  "success": false,
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  },
  "meta": {}
}
```


```
if validation_result['success']:
    print('A coluna de código de município contem apenas valores únicos')
else:
    print('A coluna código de município possui valores duplicados')
```

```
A coluna código de município possui valores duplicados
```

Noticias Projeto Integrador

2 Validação de valores nulos em uma coluna

```
validation_result = df_gx.expect_column_values_to_not_be_null(column= 'CD_MUN')
validation_result
```

[27]

```
{
  "result": {
    "element_count": 1807570,
    "unexpected_count": 0,
    "unexpected_percent": 0.0,
    "unexpected_percent_total": 0.0,
    "partial_unexpected_list": []
  },
  "success": true,
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  },
  "meta": {}
}
```

```
if validation_result['success']:
    print('A coluna código do município não possui valores nulos')
else:
    print('A coluna código do município possui valores nulos, verificar a necessidade de realizar tratamento no campo')
```

[28]

```
A coluna código do município não possui valores nulos
```

3 Verifica se há valores inválidos para latitude.

```
latitude_invalida = df_gx[(df_gx['latitude'] < -90) | (df_gx['latitude'] > 90)]

if latitude_invalida.empty:
    print("Não há valores incorretos na coluna latitude.")
else:
    print("Existem valores incorretos na coluna latitude:")
    print('Valor(es) incorreto(s):')
    print(latitude_invalida['latitude'].to_string(index=False))
```

[29]

```
Não há valores incorretos na coluna latitude.
```

```
null_latitude_count = latitude_invalida['latitude'].count()
print("Quantidade de valor(es) de latitude incorreto(s):", null_latitude_count)
```

[30]

```
Quantidade de valor(es) de latitude incorreto(s): 0
```

```
longitude_invalida = df_gx[(df_gx['longitude'] < -180) | (df_gx['longitude'] > 180)]

if longitude_invalida.empty:
    print("Não há valores incorretos na coluna longitude.")
else:
    print("Existem valores incorretos na coluna longitude:")
    print('Valor(es) incorreto(s):')
    print(longitude_invalida['longitude'].to_string(index=False))
```

[31]

```
Não há valores incorretos na coluna longitude.
```

Noticias Projeto Integrador

Plano de Teste

Testes na camada trusted para os dados sobre de internet móvel.

Testes realizados após transformar os dataset de formato shapefile em csv

```
#importando bibliotecas necessárias para os testes
import pandas as pd
import great_expectations as gx

#Criar o df tax_df com as informacoes doa arquivo us_tax_data_2016.csv
path ='./dados/validacoes_trusted/trusted_speedtest_mobile.csv'
df_mobile = pd.read_csv(path, delimiter=';', encoding="ansi")
```

[25]

```
# verificacao do layout do dataset
print(df_mobile.head(2))
```

[26]

```
            quadkey  avg_d_kbps  avg_u_kbps  avg_lat_ms  tests  devices  \
0  323230233223102       15591       17249          26      4        2  \
1  323230233232230       16998       20375          25      2        1

   index_righ   CD_MUN  latitude  longitude        date  quarter network_ty  \
0         138  1400100  2.847033 -60.751648  2022-01-01        1     mobile  \
1         138  1400100  2.819601 -60.718689  2022-01-01        1     mobile

                                      geometry
0  POLYGON ((-60.75439453125 2.8497764266656, -60...
1  POLYGON ((-60.721435546875 2.82234424689409, -...
```

Crie uma expectation para utilizar no great expectation para relizações de algumas validações:

```
df_gx = gx.dataset.PandasDataset(df_mobile)
df_gx.head(3)
```

[27]

	quadkey	avg_d_kbps	avg_u_kbps	avg_lat_ms	tests	devices	index_righ	CD_MUN	latitude	longitude	date
0	323230233223102	15591	17249	26	4	2	138	1400100	2.847033	-60.751648	2022-01-01
1	323230233232230	16998	20375	25	2	1	138	1400100	2.819601	-60.718689	2022-01-01
2	323232011001023	17268	8038	29	2	2	138	1400100	2.792168	-60.768127	2022-01-01

Noticias Projeto Integrador

1 Verifica se o código município (CD_MUN) contém apenas valores únicos.

Usar a função `expect_column_values_to_be_unique` para definir essa expectativa

```
validation_result = df_gx.expect_column_values_to_be_unique(column = 'CD_MUN')
validation_result
```

[28]


```
{
  "success": false,
  "meta": {},
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  },
  "result": {
    "element_count": 588373,
    "missing_count": 0,
    "missing_percent": 0.0,
    "unexpected_count": 588342,
    "unexpected_percent": 99.99473123341825,
    "unexpected_percent_total": 99.99473123341825,
    "unexpected_percent_nonmissing": 99.99473123341825,
    "partial_unexpected_list": [
      1400100,
      1400100,
      1400100,
      1400100,
      1400100,
      1400100,
      1400100,
      1400100,
      1400100,
      1400100,
      1400100,
      1400100,
      1400100,
      1400100,
      1400100,
      1400100,
      1400100,
      1400100,
      1400100,
      1400100,
      1400100,
      1400100
    ]
  }
}
```


Resultado da validação.

```
if validation_result['success']:
    print('A coluna de código de município contem apenas valores únicos')
else:
    print('A coluna código de município possui valores duplicados')
```

[29]

```
A coluna código de município possui valores duplicados
```

Noticias Projeto Integrador

2 Validação de valores nulos em uma coluna

```
[30]
validation_result = df_gx.expect_column_values_to_not_be_null(column= 'CD_MUN')
validation_result
```


```
{
  "success": true,
  "meta": {},
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  },
  "result": {
    "element_count": 588373,
    "unexpected_count": 0,
    "unexpected_percent": 0.0,
    "unexpected_percent_total": 0.0,
    "partial_unexpected_list": []
  }
}
```


```
[31]
if validation_result['success']:
    print('A coluna código do município não possui valores nulos')
else:
    print('A coluna código do município possui valores nulos, verificar a necessidade de realizar tratamento no campo')
```

```
A coluna código do município não possui valores nulos
```

3 Verifica se há valores inválidos para latitude.

```
[32]
latitude_invalida = df_gx[(df_gx['latitude'] < -90) | (df_gx['latitude'] > 90)]

if latitude_invalida.empty:
    print("Não há valores incorretos na coluna latitude.")
else:
    print("Existe(m) valor(es) incorreto(s) na coluna latitude:")
    print('Valo(res) invalido(s):')
    print(latitude_invalida['latitude'].to_string(index=False))
```

```
Não há valores incorretos na coluna latitude.
```

```
[33]
null_latitude_count = latitude_invalida['latitude'].count()
print("Quantidade de valor(es) de latitude incorreto(s):", null_latitude_count)
```

```
Quantidade de valor(es) de latitude incorreto(s): 0
```

Noticias Projeto Integrador

Falando sobre o processo criado em Python

Principais bibliotecas utilizadas:

- Pandas
    - Fornece estrutura de dados e ferramentas para análise e manipulação de dados: Filtrar, Transformar, Agregar e Visualizar dados.

- Geopandas
    - Extensão especializada do pandas que adiciona capacidades de geoespacialização aos DataFrames.
    - Suporte diferentes tipos de dados geoespaciais e ainda possui recursos para visualização, permitindo até mesmo a criação de mapas

Noticias Projeto Integrador

Falando sobre o processo criado em Python

Camada Raw speedtest:

	quadkey	avg_d_kbps	avg_u_kbps	avg_lat_ms	tests	devices	geometry	CD_MUN	fonte
0	0323230233223102	15591	17249	26	4	2	POLYGON ((-60.75439 2.84978, -60.74890 2.84978...	1400100	https://ookla-open-data.s3-us-west-2.amazonaws...
1	0323230233232230	16998	20375	25	2	1	POLYGON ((-60.72144 2.82234, -60.71594 2.82234...	1400100	https://ookla-open-data.s3-us-west-2.amazonaws...
2	0323232011001023	17268	8038	29	2	2	POLYGON ((-60.77087 2.79491, -60.76538 2.79491...	1400100	https://ookla-open-data.s3-us-west-2.amazonaws...
3	0323230233232011	116923	4563	36	1	1	POLYGON ((-60.71594 2.85526, -60.71045 2.85526...	1400100	https://ookla-open-data.s3-us-west-2.amazonaws...
4	0323230233232033	5092	7704	31	2	2	POLYGON ((-60.71594 2.83880, -60.71045 2.83880...	1400100	https://ookla-open-data.s3-us-west-2.amazonaws...

Camada Trusted speedtest:

	quadkey	avg_d_kbps	avg_u_kbps	avg_lat_ms	tests	devices	CD_MUN	geometry	latitude	longitude	date	quarter	network_type
0	0323230233223102	15591	17249	26	4	2	1400100	POLYGON ((-60.75439 2.84978, -60.74890 2.84978...	2.847033	-60.751648	2022-01-01	1	mobile
1	0323230233232230	16998	20375	25	2	1	1400100	POLYGON ((-60.72144 2.82234, -60.71594 2.82234...	2.819601	-60.718689	2022-01-01	1	mobile
2	0323232011001023	17268	8038	29	2	2	1400100	POLYGON ((-60.77087 2.79491, -60.76538 2.79491...	2.792168	-60.768127	2022-01-01	1	mobile
3	0323230233232011	116923	4563	36	1	1	1400100	POLYGON ((-60.71594 2.85526, -60.71045 2.85526...	2.852520	-60.713196	2022-01-01	1	mobile
4	0323230233232033	5092	7704	31	2	2	1400100	POLYGON ((-60.71594 2.83880, -60.71045 2.83880...	2.836060	-60.713196	2022-01-01	1	mobile

Noticias Projeto Integrador

Falando sobre o processo criado em Python

Camada Raw malha territorial do brasil:

CD_MUN	NM_MUN	SIGLA_UF	AREA_KM2	geometry
1100015	Alta Floresta D'Oeste	RO	7067.127	POLYGON ((-62.00806 -12.13379, -62.00784 -12.2...
1100023	Ariquemes	RO	4426.571	POLYGON ((-63.17933 -10.13924, -63.17746 -10.1...
1100031	Cabixi	RO	1314.352	POLYGON ((-60.52408 -13.32137, -60.37162 -13.3...
1100049	Cacoal	RO	3793.000	POLYGON ((-61.35502 -11.50452, -61.35524 -11.5...
1100056	Cerejeiras	RO	2783.300	POLYGON ((-60.82135 -13.11910, -60.81773 -13.1...

Camada Trusted malha territorial do brasil:

CD_MUN	NM_MUN	SIGLA_UF	geometry	distancia_km
1100015	Alta Floresta D'Oeste	RO	POLYGON ((-62.00806 -12.13379, -62.00784 -12.2...	7km
1100023	Ariquemes	RO	POLYGON ((-63.17933 -10.13924, -63.17746 -10.1...	4km
1100031	Cabixi	RO	POLYGON ((-60.52408 -13.32137, -60.37162 -13.3...	1km
1100049	Cacoal	RO	POLYGON ((-61.35502 -11.50452, -61.35524 -11.5...	4km
1100056	Cerejeiras	RO	POLYGON ((-60.82135 -13.11910, -60.81773 -13.1...	3km

Noticias Projeto Integrador

Falando sobre o processo criado em Python

Camada Raw malha territorial do brasil:

CD_MUN	NM_MUN	SIGLA_UF	AREA_KM2	geometry
1100015	Alta Floresta D'Oeste	RO	7067.127	POLYGON ((-62.00806 -12.13379, -62.00784 -12.2...
1100023	Ariquemes	RO	4426.571	POLYGON ((-63.17933 -10.13924, -63.17746 -10.1...
1100031	Cabixi	RO	1314.352	POLYGON ((-60.52408 -13.32137, -60.37162 -13.3...
1100049	Cacoal	RO	3793.000	POLYGON ((-61.35502 -11.50452, -61.35524 -11.5...
1100056	Cerejeiras	RO	2783.300	POLYGON ((-60.82135 -13.11910, -60.81773 -13.1...

Camada Trusted malha territorial do brasil:

CD_MUN	NM_MUN	SIGLA_UF	AREA_KM2	geometry
1100015	Alta Floresta D'Oeste	RO	7067.127	POLYGON ((-62.00806 -12.13379, -62.00784 -12.2...
1100023	Ariquemes	RO	4426.571	POLYGON ((-63.17933 -10.13924, -63.17746 -10.1...
1100031	Cabixi	RO	1314.352	POLYGON ((-60.52408 -13.32137, -60.37162 -13.3...
1100049	Cacoal	RO	3793.000	POLYGON ((-61.35502 -11.50452, -61.35524 -11.5...
1100056	Cerejeiras	RO	2783.300	POLYGON ((-60.82135 -13.11910, -60.81773 -13.1...

Noticias Projeto Integrador

Falando sobre o processo criado em Python

Principais tratamentos de dados realizados:

- O processo verifica sozinho quais são os arquivos que devem ser carregados.

```
In [23]: import csv
         import datetime

         # Criando o GeoDataFrame vazio
         geo_appended = gp.GeoDataFrame()

         # Criando o arquivo de log
         with open('log_raw.csv', mode='a', newline='') as log_file:
             log_writer = csv.writer(log_file, delimiter=',', quotechar='"', quoting=csv.QUOTE_MINIMAL)

             for x,y in tqdm(zip(para_inserir.last_url,para_inserir.url_completa)):
                 print(x)
                 print(y)
                 raw_new = gp.read_file(y)
                 raw_new_2 = raw_new.set_geometry('geometry') # Adicionando esta linha
                 print('etapa 1 - Leitura do arquivo - Concluida')
                 # filtra dados apenas do Brasil
                 raw_new_2 = raw_new_2.to_crs(geo_brasil.crs)
                 raw_new_3 = gp.sjoin(raw_new_2, geo_brasil, how="inner", predicate='intersects')

                 raw_new_3 = raw_new_3.assign(fonte=y)
                 raw_new_3
                 print('etapa 2 - Adicionando coluna fonte - Concluida')
                 # Concatenando o GeoDataFrame com o novo dado
                 geo_appended = pd.concat([geo_appended, raw_new_3], ignore_index=True)
                 # Adicionando a etapa no arquivo de log
                 log_writer.writerow([datetime.date.today(), y])

                 del raw_new
                 del raw_new_2
                 del raw_new_3

         # Salvando o GeoDataFrame em um único arquivo shapefile
         geo_appended.to_file('raw_speedtest_mobile.shp', driver='ESRI Shapefile')

         0it [00:00, ?it/s]

         2022-01-01_performance_mobile_tiles
         https://ookla-open-data.s3-us-west-2.amazonaws.com/shapefiles/performance/type=mobile/year=2022/quarter=1/2022-01-01_performance_mobile_tiles.zip
         etapa 1 - Leitura do arquivo - Concluida
         etapa 2 - Adicionando coluna fonte - Concluida

         1it [17:22, 1042.60s/it]

         2022-04-01_performance_mobile_tiles
         https://ookla-open-data.s3-us-west-2.amazonaws.com/shapefiles/performance/type=mobile/year=2022/quarter=2/2022-04-01_performance_mobile_tiles.zip
         etapa 1 - Leitura do arquivo - Concluida
         etapa 2 - Adicionando coluna fonte - Concluida

         2it [33:57, 1014.79s/it]

         2022-07-01_performance_mobile_tiles
```

Noticias Projeto Integrador

Falando sobre o processo criado em Python

Principais tratamentos de dados realizados:

- Confirmando que o tipo de dados geoespacial é o mesmo para todas as fontes de dados

```
geo_brasil = geo_brasil.to_crs(epsg=4674)
trusted_mobile = trusted_mobile.to_crs(epsg=4674)
trusted_fixed = trusted_fixed.to_crs(epsg=4674)
```

- Filtrando dados do Brasil na camada diretamente na raw, para evitar o carregamento de dados que não seriam usados, já que a fonte de dados possui dados do mundo todo.

```
print(geo_brasil.crs)
```

```
EPSG:4674
```

```
import csv
import datetime

# Criando o GeoDataFrame vazio
geo_appended = gp.GeoDataFrame()

# Criando o arquivo de log
with open('log_raw.csv', mode='a', newline='') as log_file:
    log_writer = csv.writer(log_file, delimiter=',', quotechar='"', quoting=csv.QUOTE_MINIMAL)

    for x,y in tqdm(zip(para_inserir.last_url,para_inserir.url_completa)):
        print(x)
        print(y)
        raw_new = gp.read_file(y)
        raw_new_2 = raw_new.set_geometry('geometry') # Adicionando esta linha
        print('etapa 1 - Leitura do arquivo - Concluída')
        # filtra dados apenas do Brasil
        raw_new_2 = raw_new_2.to_crs(geo_brasil.crs)
        raw_new_3 = gp.sjoin(raw_new_2, geo_brasil, how="inner", predicate='intersects')

        raw_new_3 = raw_new_3.assign(fonte=y)
        raw_new_3
        print('etapa 2 - Adicionando coluna fonte - Concluída')
        # Concatenando o GeoDataFrame com o novo dado
        geo_appended = pd.concat([geo_appended, raw_new_3], ignore_index=True)
        # Adicionando a etapa no arquivo de log
        log_writer.writerow([datetime.date.today(), y])

        del raw_new
        del raw_new_2
        del raw_new_3

# Salvando o GeoDataFrame em um único arquivo shapefile
geo_appended.to_file('raw_speedtest_mobile.shp', driver='ESRI Shapefile')
```

Noticias Projeto Integrador

Falando sobre o processo criado em Python

Principais tratamentos de dados realizados:

- Extração da latitude e longitude na camada Trusted.

```
df['latitude'] = df.geometry.centroid.y
df['longitude'] = df.geometry.centroid.x
df.head()
```

	quadkey	avg_d_kbps	avg_u_kbps	avg_lat_ms	tests	devices	CD_MUN	geometry	latitude	longitude	date	quarter	network_type
0	0323230233223102	15591	17249	26	4	2	1400100	POLYGON ((-60.75439 2.84978, -60.74890 2.84978...	2.847033	-60.751648	2022-01-01	1	mobile
1	0323230233232230	16998	20375	25	2	1	1400100	POLYGON ((-60.72144 2.82234, -60.71594 2.82234...	2.819601	-60.718689	2022-01-01	1	mobile
2	0323232011001023	17268	8038	29	2	2	1400100	POLYGON ((-60.77087 2.79491, -60.76538 2.79491...	2.792168	-60.768127	2022-01-01	1	mobile
3	0323230233232011	116923	4563	36	1	1	1400100	POLYGON ((-60.71594 2.85526, -60.71045 2.85526...	2.852520	-60.713196	2022-01-01	1	mobile
4	0323230233232033	5092	7704	31	2	2	1400100	POLYGON ((-60.71594 2.83880, -60.71045 2.83880...	2.836060	-60.713196	2022-01-01	1	mobile

Noticias Projeto Integrador

Falando sobre o processo criado em Python

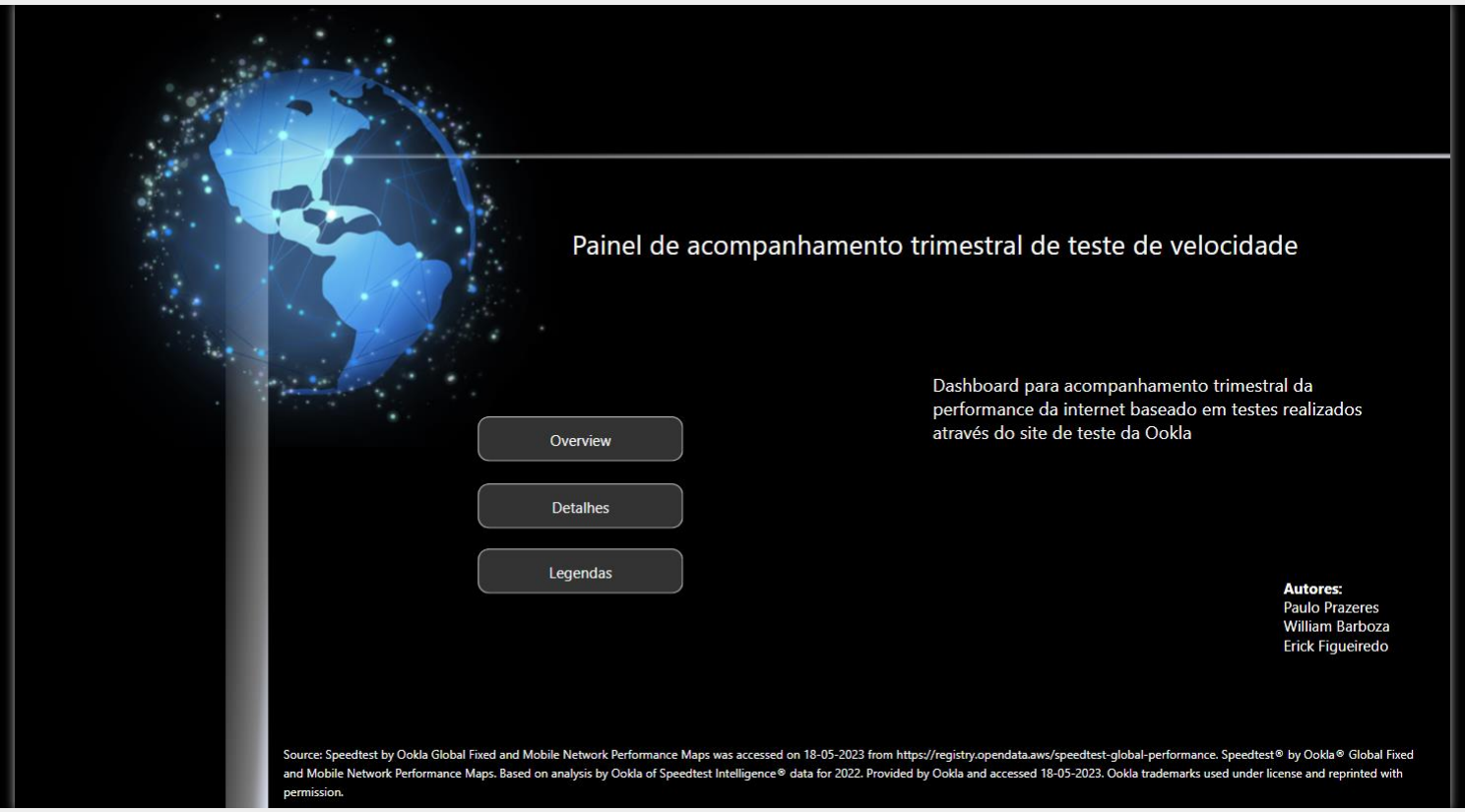
Principais tratamentos de dados realizados:

- Na fonte de dados utilizada, não existia uma coluna referente ao período, então essa informação foi extraída diretamente da coluna "fonte" que trata-se do caminho/nome original do arquivo.

```
df["date"] = df["fonte"].str[106:116]
df["quarter"] = df["fonte"].str[104:105]
df = df.drop('fonte', axis=1)
df['network_type'] = 'mobile'
df.head()
```

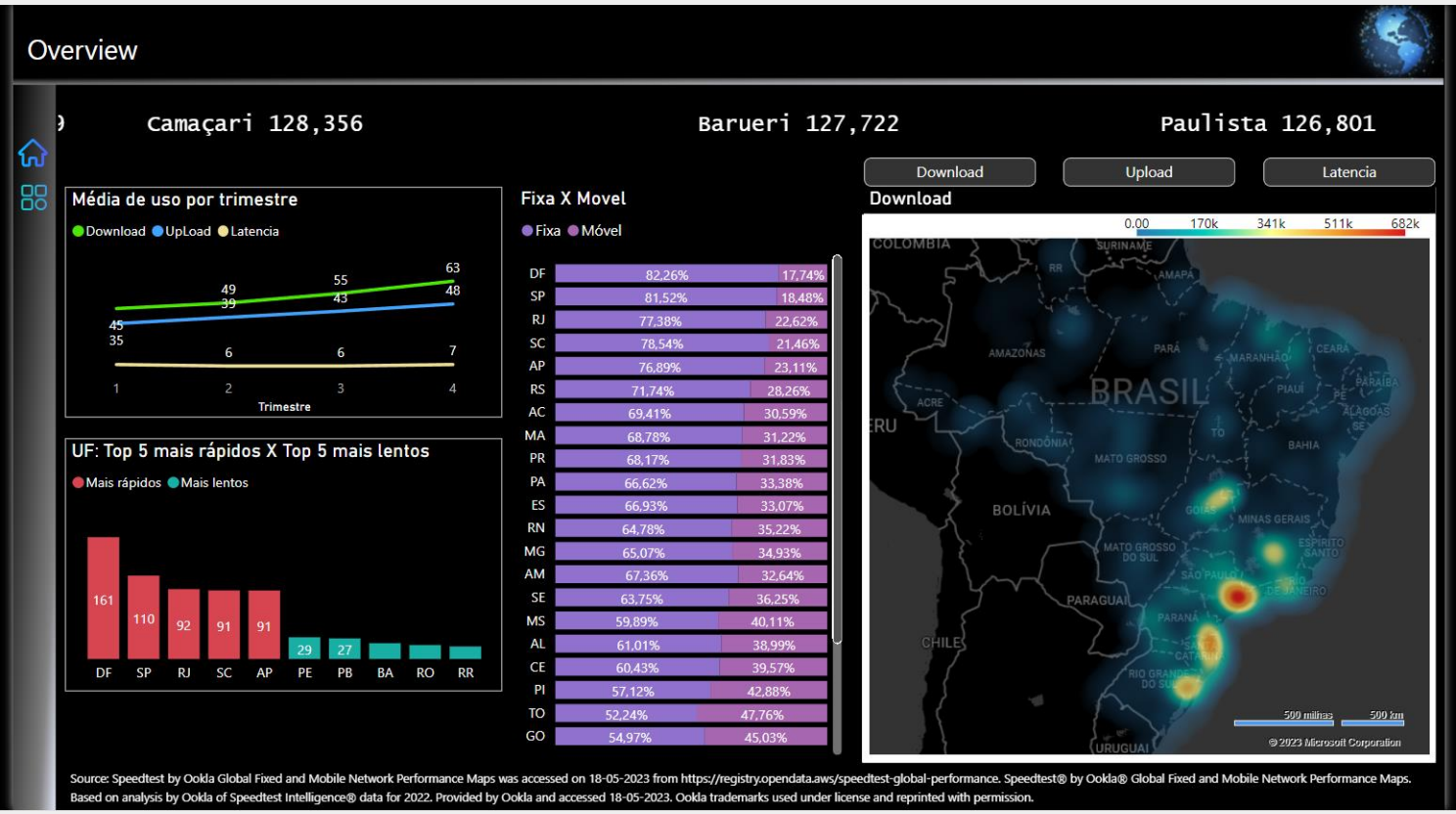
```
0it [00:00, ?it/s]

2022-01-01_performance_mobile_tiles
https://ookla-open-data.s3-us-west-2.amazonaws.com/shapefiles/performance/type=mobile/year=2022/quarter=1/2022-01-01_performance_mobile_tiles.zip
etapa 1 - Leitura do arquivo - Concluida
etapa 2 - Adicionando coluna fonte - Concluida

1it [17:22, 1042.60s/it]

2022-04-01_performance_mobile_tiles
https://ookla-open-data.s3-us-west-2.amazonaws.com/shapefiles/performance/type=mobile/year=2022/quarter=2/2022-04-01_performance_mobile_tiles.zip
etapa 1 - Leitura do arquivo - Concluida
etapa 2 - Adicionando coluna fonte - Concluida

2it [33:57, 1014.79s/it]

2022-07-01_performance_mobile_tiles
```

Noticias Projeto Integrador



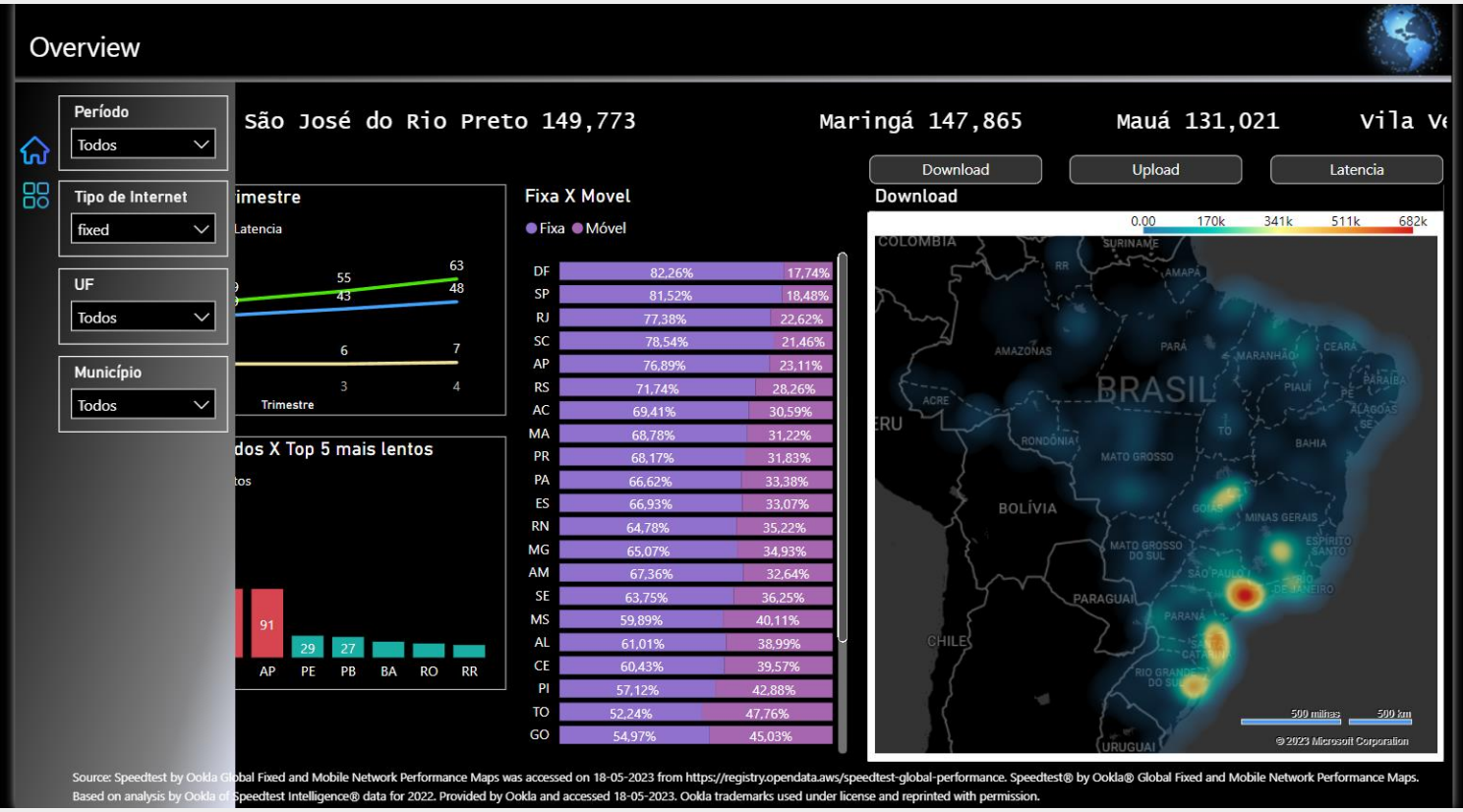
2023-06-25 **Primeira Edição**

Noticias Projeto Integrador



Source: Speedtest by Ookla Global Fixed and Mobile Network Performance Maps was accessed on 18-05-2023 from https://registry.opendata.aws/speedtest-global-performance. Speedtest® by Ookla® Global Fixed and Mobile Network Performance Maps. Based on analysis by Ookla of Speedtest Intelligence® data for 2022. Provided by Ookla and accessed 18-05-2023. Ookla trademarks used under license and reprinted with permission.

Noticias Projeto Integrador



Noticias Projeto Integrador

Proposta:

Vamos analisar todos juntos?

2023-06-25 Primeira Edição