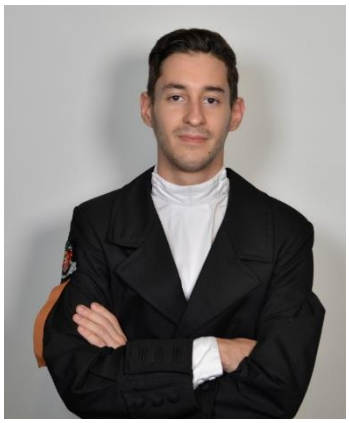
Engenharia de Dados para Suporte à Tomada de Decisão

Ano Letivo 2022/2023

RELATÓRIO



Gonçalo Fernandes Pereira

A96550

Joana Marília Pinto Sousa Pereira

A95539

Mariana Magalhães Araújo

A97245

Pedro Miguel Alves Brandão Soares

A97533

Índice

Incidentes de Petróleo

Link: https://www.kaggle.com/datasets/new-york-state/nys-spill-incidents?select=spill-incidents.csv

Questões Analíticas:

1 – Quais as localidades com maior número de derrames e quais fatores que mais contribuíram para estes?

2 – Qual a percentagem de derrame ao longo do século XXI?

3 – Quais as fontes que originaram um maior desperdicio de residuos?

KPI: Variação relativa da ocorrência de derrames no ano de 2015 em comparação ao de 2016.

Se esta variação apresentar valores negativos, então concluímos que variou desfavoravelmente, uma vez que, a ocorrência de derrames do primeiro ano foi mais elevada que a do segundo. A variação entre estes valores será negativa.

O mesmo se pode dizer do caso contrário, isto é, se a variação for uma valor positivo, então a ocorrência de derrames ao londo destes anos evoluiu favoravelmente. Tomando o valor 1% como referência, podemos dizer que, no caso de o valor da variação for acima deste, então evoluiu muito favoravelmente.

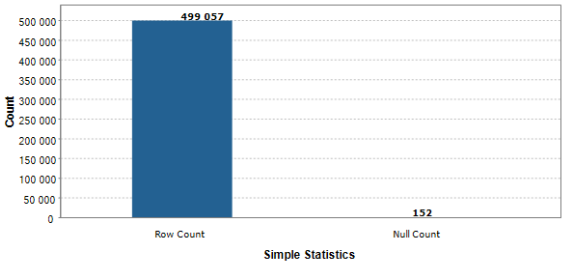
Problema de qualidade de dados:

Na coluna Spill_Date existem 152 datas que são nulas. De forma a resolver este problema, optamos por remover estas linhas, dado que é um número reduzido de dados em comparação com a grandeza do dataset.

Column: metadata.Spill_Date

Simple Statistics

Label	Count	%
Row Count	499057	100.00%
Null Count	152	0.03%

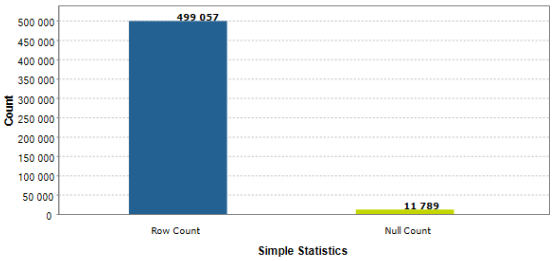


O mesmo acontece na coluna Close_Date, portanto optamos pela mesma solução usada na coluna anterior anterior.

Column: metadata.Close_Date

Simple Statistics

Label	Count	%
Row Count	499057	100.00%
Null Count	11789	2.36%

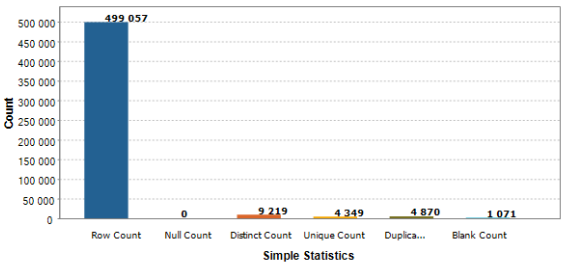


Relativamente às colunas Locality, Street 1, Units e Stream Group, para solucionar as linhas em branco optamos por colocar estes atributos como “Desconhecido”.

Column: metadata.Locality

Simple Statistics

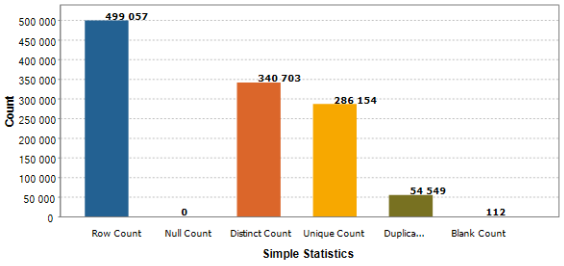
Label	Count	%
Row Count	499057	100.00%
Null Count	0	0.00%
Distinct Count	9219	1.85%
Unique Count	4349	0.87%
Duplicate Count	4870	0.98%
Blank Count	1071	0.21%



Column: metadata.Street_1

Simple Statistics

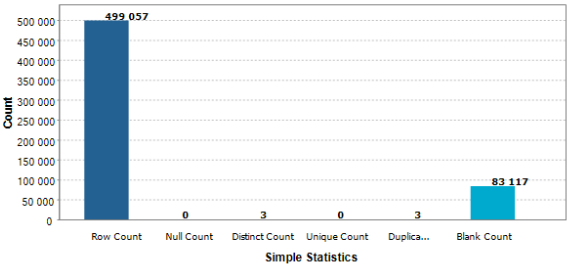
Label	Count	%
Row Count	499057	100.00%
Null Count	0	0.00%
Distinct Count	340703	68.27%
Unique Count	286154	57.34%
Duplicate Count	54549	10.93%
Blank Count	112	0.02%



Column: metadata.Units

Simple Statistics

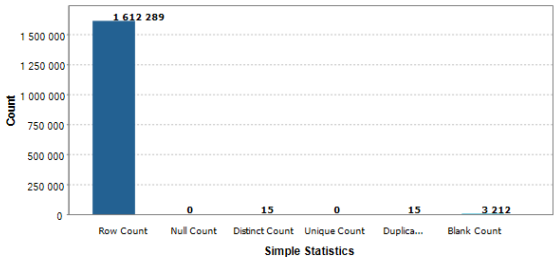
Label	Count	%
Row Count	499057	100.00%
Null Count	0	0.00%
Distinct Count	3	6.011E-4%
Unique Count	0	0.00%
Duplicate Count	3	6.011E-4%
Blank Count	83117	16.65%



Column: metadata.Stream_Group

Simple Statistics

Label	Count	%
Row Count	1612289	100.00%
Null Count	0	0.00%
Distinct Count	15	9.304E-4%
Unique Count	0	0.00%
Duplicate Count	15	9.304E-4%
Blank Count	3212	0.20%

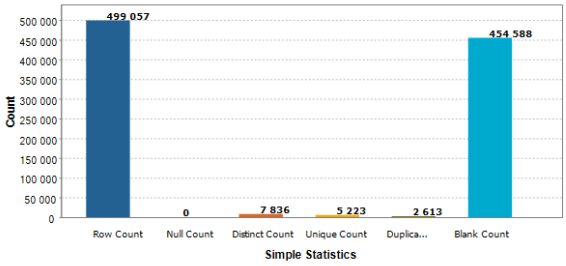


Os atributos Waterbody e Street 2 possuem 91,09% e 92,12% de linhas em branco, respetivamente. Uma vez que são 2 atributos considerados irrelevantes na nossa análise e devido à grande percentagem de elementos em branco, optamos pela sua remoção.

Column: metadata.Waterbody

Simple Statistics

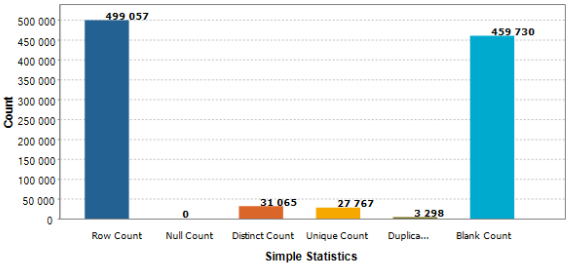
Label	Count	%
Row Count	499057	100.00%
Null Count	0	0.00%
Distinct Count	7836	1.57%
Unique Count	5223	1.05%
Duplicate Count	2613	0.52%
Blank Count	454588	91.09%



Column: metadata.Street_2

Simple Statistics

Label	Count	%
Row Count	499057	100.00%
Null Count	0	0.00%
Distinct Count	31065	6.22%
Unique Count	27767	5.56%
Duplicate Count	3298	0.66%
Blank Count	459730	92.12%



Qualidade do ar

Link: https://data.cityofnewyork.us/Environment/Air-Quality/c3uy-2p5r

Questões Analíticas:

1 – Quais os gases com efeitos cancerígenos (Benzene e Formaldehyde) que estão mais presentes em cada uma das localidades?

2 – Quais os períodos de tempo em que a presença do gás O_3 foi mais notória?

3 – Quais os principais efeitos, na população, resultantes da exposição ao PM2.5, por cada localidade?

KPI: Na zona geografica de Borough, a variação dos níveis de poluição do ar entre 2009 e 2015.

Se esta variação apresentar valores negativos, então concluímos que variou favoravelmente, uma vez que, se os níveis de poluição do ar no ano de 2009 foram mais elevados que em 2015, a variação entre estes valores será negativa.

Se a variação for um valor positivo, então a poluição do ar nestes anos evoluiu desfavoravelmente. Tomando o valor 1% como referência, podemos dizer que, no caso de o valor da variação for abaixo deste, então evoluiu muito favoravelmente.

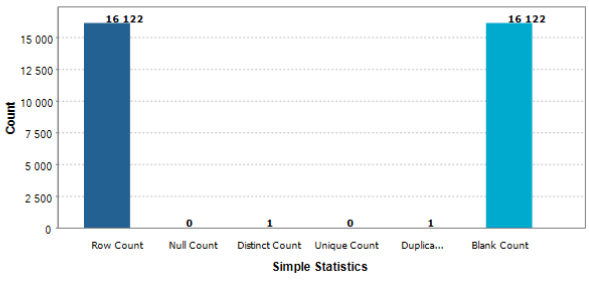
Problema de qualidade de dados:

O atributo Message não apresenta qualquer tipo de dado, sendo, por isso, um atributo a ser removido.

Column: metadata.Message

Simple Statistics

Label	Count	%
Row Count	16122	100.00%
Null Count	0	0.00%
Distinct Count	1	6.203E-3%
Unique Count	0	0.00%
Duplicate Count	1	6.203E-3%
Blank Count	16122	100.00%



Qualidade da Água da Bacia Hidrográfica

Link: https://data.cityofnewyork.us/Environment/Watershed-Water-Quality-Hydrology/e3xf-jwmj

Questões Analíticas:

1 – Qual a média de pH, turbidity e temperature (tipos de análise) em cada uma das redes de rios?

2 – Qual o tipo de amostra mais frequente em cada ano?

3 – Qual a evolução do pH, ao longos dos últimos 10 anos, nas diferentes redes de rios?

KPI: Na rede de rios Esopus, a variação do pH entre 2018 e 2019.

Se esta variação apresentar valores negativos, então concluímos que variou desfavoravelmente, uma vez que pretendemos que este aumente nestes dois anos.

Se a variação for um valor positivo, então o pH evoluiu desfavoravelmente. Tomando o valor 1% como referência, podemos dizer que, no caso de o valor da variação for acima deste, então evoluiu muito favoravelmente.

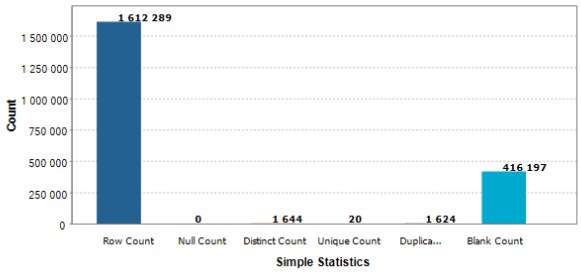
Problema de qualidade de dados:

A coluna Sample_Time apresenta 416197 linhas em branco. Para a resolução deste problema optamos por colocar estas linhas com o valor 0, uma vez que a remoção destes elementos implicaria a perda de cerca de 25% dos dados.

Column: metadata.Sample_Time

Simple Statistics

Label	Count	%
Row Count	1612289	100.00%
Null Count	0	0.00%
Distinct Count	1644	0.10%
Unique Count	20	1.24E-3%
Duplicate Count	1624	0.10%
Blank Count	416197	25.81%

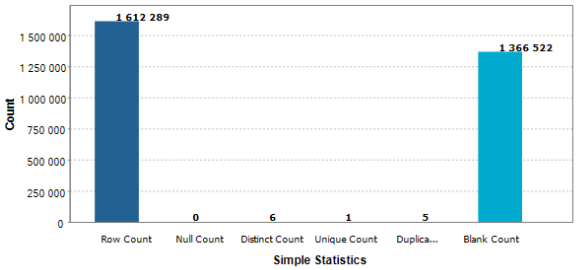


Em relação às analises em que o Status é apresentado em branco, optamos por colocar como “Desconhecido”.

Column: metadata.Status

Simple Statistics

Label	Count	%
Row Count	1612289	100.00%
Null Count	0	0.00%
Distinct Count	6	3.721E-4%
Unique Count	1	6.202E-5%
Duplicate Count	5	3.101E-4%
Blank Count	1366522	84.76%

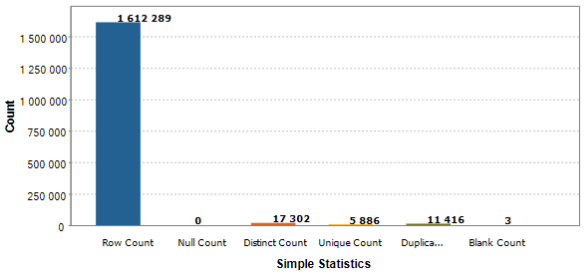


Os dados que apresentam o atributo Final_Result em branco serão removidos.

Column: metadata.Final_Result

Simple Statistics

Label	Count	%
Row Count	1612289	100.00%
Null Count	0	0.00%
Distinct Count	17302	1.07%
Unique Count	5886	0.37%
Duplicate Count	11416	0.71%
Blank Count	3	1.861E-4%



Taxas de Desvio e Captura de Reciclagem

Link: https://data.cityofnewyork.us/Environment/Recycling-Diversion-and-Capture-Rates/gaq9-z3hz?fbclid=IwAR2kg4D3xp7OzkE8yaSmj6bjJq0yGxumB86tPgDuEwziaqaQrwyXaCGMKkE

Questões Analíticas:

1 – Qual o tipo de resíduos mais reciclados em cada uma das localidades ?

2 – Qual o período de tempo no qual ocorreu um maior aumento da percentagem de reciclagem?

3 – Quais os anos em que o rácio entre o lixo recliclado e o lixo comum foi mais elevado?

KPI: Variação do rácio total de reciclagem recolhida entre o primeiro semestre de 2017 e o segundo semestre deste mesmo ano.

Se esta variação apresentar valores negativos, então concluímos que variou desfavoravelmente, pois se o total de reciclagem for maior no primeiro semestre e menor no segundo, a variação entre estes valores será negativa.

O mesmo se pode dizer do caso contrário, isto é, se a variação for uma valor positivo, então o total de reciclagem recolhida nestes anos evoluiu favoravelmente. Tomando o valor 1% como referência, podemos dizer que, no caso de o valor da variação for acima deste, então evoluiu muito favoravelmente.

Problema de qualidade de dados:

Este dataset não apresenta problemas ao nivel da qualidade de dados.

Questões Analíticas Gerais

1 – Em que anoss, o número de derrames afetou o pH da água? (Petróleo e Água)

2 – Qual a relação entre os níveis de ozono (O_3) e a temperatura da água nos ultimos 5 anos? (Ar e Água)

3 – Quais as zonas em que um maior rácio entre o lixo recliclado e o lixo comum implicou um menor nível de ozono? (Reciclagem e Ar)

Arquitetura

Camada Bronze

Depois de escolhido o tema e selecionados os datasets que iremos utilizar neste trabalho, fizemos upload destes para o HFDS, sem qualquer tratamento no que toca aos dados.

Este upload foi realizado através da execução dos seguintes scripts no Jupyter. Os ficheiros encontram-se estruturados como "/TrabalhoPL/bronze/nome_do_dataset.csv".

```
[3]: from os import PathLike
     from hdfs import InsecureClient
     client = InsecureClient("http://hdfs-nn:9870", user="anonymous")

     from_path = "./spill-incidents.csv"
     to_path = "/TrabalhoPL/bronze/spill-incidents.csv"

     client.delete(to_path)

     client.upload(to_path, from_path)

[3]: '/TrabalhoPL/bronze/spill-incidents.csv'
```

Figura 1 - Incidentes de Petróleo

```
[ ]: from os import PathLike
     from hdfs import InsecureClient
     client = InsecureClient("http://hdfs-nn:9870", user="anonymous")

     from_path = "./Air_Quality.csv"
     to_path = "/TrabalhoPL/bronze/Air_Quality.csv"

     client.delete(to_path)

     client.upload(to_path, from_path)

[1]: '/TrabalhoPL/bronze/Air_Quality.csv'
```

Figura 2 - Qualidade do ar

```
[4]: from os import PathLike
     from hdfs import InsecureClient
     client = InsecureClient("http://hdfs-nn:9870", user="anonymous")

     from_path = "./Watershed_Water_Quality_-_Hydrology.csv"
     to_path = "/TrabalhoPL/bronze/Watershed_Water_Quality_-_Hydrology.csv"

     client.delete(to_path)

     client.upload(to_path, from_path)

[4]: '/TrabalhoPL/bronze/Watershed_Water_Quality_-_Hydrology.csv'
```

Figura 3 - Qualidade da Água da Bacia Hidrográfica

```
[2]: from os import PathLike
     from hdfs import InsecureClient
     client = InsecureClient("http://hdfs-nn:9870", user="anonymous")

     from_path = "./Recycling_Diversion_and_Capture_Rates.csv"
     to_path = "/TrabalhoPL/bronze/Recycling_Diversion_and_Capture_Rates.csv"

     client.delete(to_path)

     client.upload(to_path, from_path)

[2]: '/TrabalhoPL/bronze/Recycling_Diversion_and_Capture_Rates.csv'
```

Figura 4 – Taxas de desvio e capturas de reciclagem

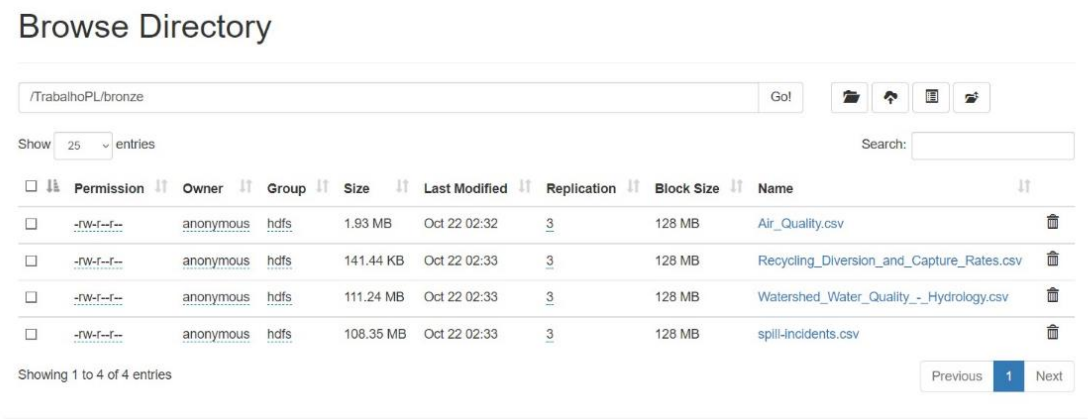


Figura 5 – Arquitetura bronze no HDFS

Camada Silver

Nesta segunda fase, o objetivo principal é o tratamento de dados. O tramento de dados consiste na resolução dos problemas que referimos em cima sobre cada um dos datasets, de forma a obtermos um dataset mais claro, organizado e acessível.

Inicialmente, procedemos à criação da base de dados e, de seguida, carregamos os dados, que se encontram na camada anterior, para StructType e procedemos às alterações necessárias para a correção do dataset.

```
spark.sql(
    """
    DROP DATABASE IF EXISTS Projeto CASCADE
    """
)

spark.sql(
    """
    create database Projeto location 'hdfs://hdfs-nn:9000/TrabalhoPL/silver/Projeto.db'
    """
)
```

Figura 1 – Criação da base de dados

Incidentes de petróleo

Primeiramente, vamos criar o StructType e carregar os dados para este.

```
hdfs_path = "hdfs://hdfs-nn:9000/TrabalhoPL/bronze/spill-incidents.csv"

customSchema = StructType([
    StructField("Spill Number", StringType(), True),
    StructField("Program Facility Name", StringType(), True),
    StructField("Street 1", StringType(), True),
    StructField("Street 2", StringType(), True),
    StructField("Locality", StringType(), True),
    StructField("Contry", StringType(), True),
    StructField("ZIP Code", StringType(), True),
    StructField("SWIS Code", IntegerType(), True),
    StructField("DEC Region", IntegerType(), True),
    StructField("Spill Date", StringType(), True),
    StructField("Received Date", StringType(), True),
    StructField("Contributing Factor", StringType(), True),
    StructField("Waterbody", StringType(), True),
    StructField("Source", StringType(), True),
    StructField("Close Date", StringType(), True),
    StructField("Material Name", StringType(), True),
    StructField("Material Family", StringType(), True),
    StructField("Quantity", FloatType(), True),
    StructField("Units", StringType(), True),
    StructField("Recovered", FloatType(), True)

])

projeto_spill = spark \
            .read\
            .option("delimiter",",")\
            .option("header","true")\
            .schema(customSchema) \
            .csv(hdfs_path)
projeto_spill.show()
```

Figura 2 - Criação do StructType

De seguida, iremos proceder à alteração dos dados desejados.

Nas colunas “Spill Date” e “Close Date” alteramos as linhas nulas/ em branco para o formato ano–mês–diaThora:minuto:segundo:milesegundo.

```
from pyspark.sql.functions import when, col, concat, lit

replaced_projeto_spill = projeto_spill.withColumn(
    "Spill Date",
    when(
        (col("Spill Date").isNull() | (col("Spill Date") == None)),
        "0000-00-00T00:00:00.000"
    ).otherwise(col("Spill Date")))
```

Figura 3 – Coluna Spill Date

```
replaced_projeto_spill3 = replaced_projeto_spill2.withColumn(
    "Close Date",
    when(
        (col("Close Date").isNull() | (col("Close Date") == None)),
        "0000-00-00T00:00:00.000"
    ).otherwise(col("Close Date")))
```

Figura 4

Posteriormente, criamos as colunas ”Data_Derrame”, “Data_Relatada”, “Data_Fim” que irão adotar o valor das colunas “Spill Date”, “Received Date” e “Close Date”, respetivamente, com a data no formato ano-mês-dia.

Por outro lado, procedemos à remoção destas últimas três colunas juntamente com as colunas “Street 2” e “ZIP Code”.

```
replaced_projeto_spill4 = replaced_projeto_spill3.withColumn('Data_Derrame', split(replaced_projeto_spill3['Spill Date'], 'T').getItem(0)) \
                                                 .withColumn('Data_Relatada', split(replaced_projeto_spill3['Received Date'], 'T').getItem(0)) \
                                                 .withColumn('Data_Fim', split(replaced_projeto_spill3['Close Date'], 'T').getItem(0)) \
                                                 .drop(col("Spill Date")) \
                                                 .drop(col("Received Date")) \
                                                 .drop(col("Close Date")) \
                                                 .drop(col("Street 2")) \
                                                 .drop(col("ZIP Code"))
replaced_projeto_spill4.toPandas()
```

Figura 5

Na colunas “Locality”, “Street 1” e “Waterbody” alteramos as linhas nulas/ em branco para desconhecido.

```
replaced_projeto_spill4 = replaced_projeto_spill4.withColumn(
    "Locality",
    when(
        (col("Locality").isNull() | (col("Locality") == None)),
        "Desconhecida"
    ).otherwise(col("Locality")))
```

Figura 6

```
replaced_projeto_spill4 = replaced_projeto_spill4.withColumn(
    "Street 1",
    when(
        (col("Street 1").isNull() | (col("Street 1") == None)),
        "Desconhecida"
    ).otherwise(col("Street 1")))
```

Figura 7

```
replaced_projeto_spill4 = replaced_projeto_spill4.withColumn(
    "Waterbody",
    when(
        (col("Waterbody").isNull() | (col("Waterbody") == None)),
        "Nenhuma"
    ).otherwise(col("Waterbody")))
```

Figura 8

O formato dos dados das colunas “Data_Derrame”, “Data_Relatada” e “Data_Fim” é modificado de StringType para DateType. Além disso, é criada a coluna “Ano” através da “Data_Derrame” que conterá dados do tipo IntegerType.

```
replaced_projeto_spill5 = replaced_projeto_spill4.withColumn("Data_Derrame", to_date(col("Data_Derrame"), "yyyy-MM-dd")) \
                                        .withColumn("Data_Relatada", to_date(col("Data_Relatada"), "yyyy-MM-dd")) \
                                         .withColumn("Data_Fim", to_date(col("Data_Fim"), "yyyy-MM-dd"))

replaced_projeto_spill5 = replaced_projeto_spill5.withColumn('Ano', (split(replaced_projeto_spill5['Data_Derrame'], '-').getItem(0)).cast(IntegerType()))
replaced_projeto_spill5.show()
```

Figura 9

É alterado o nome das seguintes colunas de forma a não conter espaços.

```
replaced_projeto_spill6 = replaced_projeto_spill5 \
    .withColumnRenamed("Spill Number","Spill_Number") \
    .withColumnRenamed("Program Facility Name","Program_Facility_Name") \
    .withColumnRenamed("Street 1","Street") \
    .withColumnRenamed("SWIS Code","SWIS_Code") \
    .withColumnRenamed("DEC Region","DEC_Region")\
    .withColumnRenamed("Contributing Factor","Contributing_Factor") \
    .withColumnRenamed("Material Name","Material_Name") \
    .withColumnRenamed("Material Family","Material_Family")
```

Figura 10

Criação de uma tabela delta na base de dados com as colunas correspondentes ao nosso dataset e particionada por anos.

```
spark.sql(
    """
    DROP TABLE IF EXISTS Projeto.Tabela_Petroleo
    """
)

spark.sql(
    """
    CREATE EXTERNAL TABLE Projeto.Tabela_Petroleo (
        Spill_Number VARCHAR(50),
        Program_Facility_Name VARCHAR(500),
        Street VARCHAR(500),
        Locality VARCHAR(50),
        Contry VARCHAR(50),
        SWIS_Code int,
        DEC_Region int,
        Contributing_Factor VARCHAR(500),
        Waterbody VARCHAR(50),
        Source VARCHAR(50),
        Material_Name VARCHAR(500),
        Material_Family VARCHAR(500),
        Quantity float,
        Units VARCHAR(100),
        Recovered float,
        Data_Derrame date,
        Data_Relatada date,
        Data_Fim date

    )
       USING DELTA

   PARTITIONED BY (
        Ano INT

    )
    LOCATION 'hdfs://hdfs-nn:9000/TrabalhoPL/silver/Projeto.db/Tabela_Petroleo'
    """
)
```

Figura 11 – Criação da tabela Petróleo

Por fim, carregamos os dados modificados para a nossa base de dados.

```
#write df to hive deltalake_table
replaced_projeto_spill6 \
    .select("Spill_Number","Program_Facility_Name","Street","Locality","Contry","SWIS_Code",
            "DEC_Region","Contributing_Factor","Waterbody", "Source", "Material_Name", "Material_Family",
            "Quantity", "Units", "Recovered", "Data_Derrame", "Data_Relatada", "Data_Fim", "Ano") \
    .write \
    .mode("overwrite") \
    .partitionBy("Ano") \
    .format("delta") \
    .save("hdfs://hdfs-nn:9000/TrabalhoPL/silver/Projeto.db/Tabela_Petroleo")
from pyspark.sql.types import *
```

Figura 12 – Upload dos dados para a base de dados

Qualidade do Ar

Primeiramente, vamos criar a StructType, carregar os dados e eliminar a coluna “Message”.

```
hdfs_path = "hdfs://hdfs-nn:9000/TrabalhoPL/bronze/Air_Quality.csv"

customSchema = StructType([
    StructField("Unique ID", StringType(), True),
    StructField("Indicator ID", IntegerType(), True),
    StructField("Name", StringType(), True),
    StructField("Measure", StringType(), True),
    StructField("Measure Info", StringType(), True),
    StructField("Geo Type Name", StringType(), True),
    StructField("Geo Join ID", StringType(), True),
    StructField("Geo Place Name", StringType(), True),
    StructField("Time Period", StringType(), True),
    StructField("Start_Date", StringType(), True),
    StructField("Data Value", FloatType(), True),
    StructField("Message", StringType(), True)
])

projeto_air = spark \
            .read\
            .option("delimiter",",")\
            .option("header","true")\
            .schema(customSchema) \
            .csv(hdfs_path)
projeto_air.toPandas()
```

Figura 13

```
replaced_projeto_air = projeto_air.drop("Message")
replaced_projeto_air.toPandas()
```

Figura 14

Criamos a coluna “End_Date” que tem como objetivo obter a data do fim através do período referido na coluna “Time Period”.

```
replaced_projeto_air2 = replaced_projeto_air.withColumn('End_Date',
                                    when(col("Time Period").startswith('Summer'), concat(split('Start_Date', '/').getItem(2), lit('/09/'), split('Start_Date', '/').getItem(1)))
                                    .when(col("Time Period").startswith('Winter'), concat(substring(split('Start_Date', '/').getItem(2)+1, 0, 4), lit('/03/'), split('Start_Date', '/').getItem(1)))
                                    .when(col("Time Period").startswith('2') & col('Time Period').contains('-'), concat(lit('20'), substring('Time Period', 8,50), lit('/12/'), lit('31')))
                                    .when(col("Time Period").startswith('Annual'), concat(substring('Time Period', 16,50), lit('/12/'), lit('/31')))
                                    .otherwise(concat(col('Time Period'), lit('/12'), lit('/31'))))
replaced_projeto_air2.toPandas()
```

Figura 15

A coluna “Localidade”, criada por nós, irá armazenar as locadidades obtidas através do “Geo Type Name” e “Geo Join ID”.

Figura 16

Nas colunas “Start_Date” e “End Date” convertemos os dados de StringType para DateType.

É criada, também, a coluna “Ano” que obtem o ano da coluna “Start_Date”.

Figura 17

É alterado o nome das seguintes colunas de forma a não conter espaços.

Figura 18

Criação de uma tabela delta na base de dados com as colunas correspondentes ao nosso dataset e particionada por anos.

```
spark.sql(
    """
    DROP TABLE IF EXISTS Projeto.Tabela_Ar
    """
)

spark.sql(
    """
    CREATE EXTERNAL TABLE Projeto.Tabela_Ar(
        Unique_ID VARCHAR(50),
        Indicator_ID int,
        Name VARCHAR(500),
        Measure VARCHAR(50),
        Measure_Info VARCHAR(50),
        Geo_Type_Name VARCHAR(50),
        Geo_Join_ID VARCHAR(50),
        Geo_Place_Name VARCHAR(500),
        Time_Period VARCHAR(50),
        Start_Date DATE,
        Data_Value float,
        End_Date DATE,
        Localidade VARCHAR(50)

    )
       USING DELTA

   PARTITIONED BY (
        Ano INT

    )
    LOCATION 'hdfs://hdfs-nn:9000/TrabalhoPL/silver/Projeto.db/Tabela_Ar'
    """
)
```

Figura 19

Por fim, carregamos os dados modificados para a nossa base de dados.

```
#write df to hive deltalake_table
replaced_projeto_air5 \
    .select("Unique_ID","Indicator_ID","Name","Measure","Measure_Info","Geo_Type_Name",
            "Geo_Join_ID","Geo_Place_Name","Time_Period", "Start_Date", "Data_Value", "End_Date", "Localidade" , "Ano") \
    .write \
    .mode("overwrite") \
    .partitionBy("Ano") \
    .format("delta") \
    .save("hdfs://hdfs-nn:9000/TrabalhoPL/silver/Projeto.db/Tabela_Ar")
from pyspark.sql.types import *
```

Figura 20

Qualidade da Água das Bacias Hidográficas

Criação da StructType e carregamento dos dados.

```
hdfs_path = "hdfs://hdfs-nn:9000/TrabalhoPL/bronze/Watershed_Water_Quality_-_Hydrology.csv"

customSchema = StructType([
    StructField("Sample Id", StringType(), True),
    StructField("Sample Site", StringType(), True),
    StructField("Sample Date", StringType(), True),
    StructField("Sample Time", StringType(), True),
    StructField("Analyte", StringType(), True),
    StructField("Status", StringType(), True),
    StructField("Final Result", StringType(), True),
    StructField("Units", StringType(), True),
    StructField("Stream Group", StringType(), True)

])

projeto_water = spark \
            .read\
            .option("delimiter",",")\
            .option("header","true")\
            .schema(customSchema) \
            .csv(hdfs_path)
projeto_water.show()
```

Figura 21

Na coluna “Sample Time”, substituimos as linhas nulas e/ou brancas pela estrutura 00:00.

```
replaced_projeto_water = projeto_water.withColumn(
    "Sample Time",
    when(
        (col("Sample Time").isNull() | (col("Sample Time") == None)),
        "00:00"
    ).otherwise(col("Sample Time")))

replaced_projeto_water.show()
```

Figura 22

Na coluna “Sample Time”, os dados no formato 00:00:00:00 são colocados no formato 00:00.

```
replaced_projeto_water = replaced_projeto_water.withColumn('Sample Time', concat(split(replaced_projeto_water['Sample Time'], ':').getItem(0), lit(':'), split(replaced_projeto_water['Sample Time'], ':').getItem(1)))
replaced_projeto_water.show()
```

Figura 23

Nas colunas “Status”, “Stream Group” e “Units” substituímos as linhas que têm valor nulo ou em branco por descohecido/ desconhecida.

```
replaced_projeto_water2 = replaced_projeto_water.withColumn(
    "Status",
    when(
        (col("Status").isNull() | (col("Status") == None)),
        "Desconhecido"
    ).otherwise(col("Status")))
replaced_projeto_water2.show()
```

Figura 24

```
replaced_projeto_water4 = replaced_projeto_water3.withColumn(
    "Units",
    when(
        (col("Units").isNull() | (col("Units") == None)),
        "Desconhecida"
    ).otherwise(col("Units")))
replaced_projeto_water4.show()
```

Figura 25

```
replaced_projeto_water5 = replaced_projeto_water4.withColumn(
    "Stream Group",
    when(
        (col("Stream Group").isNull() | (col("Stream Group") == None)),
        "Desconhecida"
    ).otherwise(col("Stream Group")))
replaced_projeto_water5.show()
```

Figura 26

Na coluna “Final Result”, quando as linhas têm valor nulo ou estão em branco, modificamos o valor para 0.

```
replaced_projeto_water3 = replaced_projeto_water2.withColumn(
    "Final Result",
    when(
        (col("Final Result").isNull() | (col("Final Result") == None)),
        "0"
    ).otherwise(col("Final Result")))
replaced_projeto_water3.show()
```

Figura 27

Na coluna “Sample Time”, colocamos o tempo no formato 00:00.

```
replaced_projeto_water = replaced_projeto_water.withColumn('Sample Time', concat(split(replaced_projeto_water['Sample Time'], ':').getItem(0), lit(':'),
                                                                                 split(replaced_projeto_water['Sample Time'], ':').getItem(1)))
replaced_projeto_water.show()
```

Figura 28

Nestas colunas, retiramos os espaços substituindo-os por _.

```
replaced_projeto_water7 = replaced_projeto_water6 \
    .withColumnRenamed("Sample Id","Sample_Id") \
    .withColumnRenamed("Sample Site","Sample_Site") \
    .withColumnRenamed("Sample Date","Sample_Date") \
    .withColumnRenamed("Sample Time","Sample_Time") \
    .withColumnRenamed("Final Result","Final_Result")\
    .withColumnRenamed("Stream Group","Stream_Group")
replaced_projeto_water7.show()
```

Figura 29

Nas colunas “Sample_Date” convertemos os dados de StringType para DateType.

É criada, também, a coluna “Ano” que obtem o ano da coluna “Start_Date”.

```
replaced_projeto_water6 = replaced_projeto_water5.withColumn("Sample Date", to_date(col("Sample Date"), "MM/dd/yyy"))
replaced_projeto_water6 = replaced_projeto_water6.withColumn('Ano', (split(replaced_projeto_water6['Sample Date'], '-').getItem(0)).cast(IntegerType()))
replaced_projeto_water6.show()
```

Figura 30

Criação de uma tabela delta na base de dados com as colunas correspondentes ao nosso dataset e particionada por anos.

```
spark.sql(
    """
    DROP TABLE IF EXISTS Projeto.Tabela_Agua
    """
)

spark.sql(
    """
    CREATE EXTERNAL TABLE Projeto.Tabela_Agua (
        Sample_Id VARCHAR(50),
        Sample_Site VARCHAR(50),
        Sample_Date date,
        Sample_Time VARCHAR(50),
        Analyte VARCHAR(500),
        Status VARCHAR(50),
        Final_Result VARCHAR(50),
        Units VARCHAR(50),
        Stream_Group VARCHAR(50)

    )
       USING DELTA

   PARTITIONED BY (
        Ano INT

    )
    LOCATION 'hdfs://hdfs-nn:9000/TrabalhoPL/silver/Projeto.db/Tabela_Agua'
    """
)
```

Figura 31

Por fim, carregamos os dados modificados para a nossa base de dados.

```
#write df to hive deltalake_table
replaced_projeto_water7 \
    .select("Sample_Id","Sample_Site","Sample_Date","Sample_Time","Analyte","Status",
            "Final_Result","Units","Stream_Group", "Ano") \
    .write \
    .mode("overwrite") \
    .partitionBy("Ano") \
    .format("delta") \
    .save("hdfs://hdfs-nn:9000/TrabalhoPL/silver/Projeto.db/Tabela_Agua")
from pyspark.sql.types import *
```

Figura 32

Taxas e Desvio de Reciclagem

Vamos criar o StructType e carregar os dados para este.

```
hdfs_path = "hdfs://hdfs-nn:9000/TrabalhoPL/bronze/Recycling_Diversion_and_Capture_Rates.csv"

customSchema = StructType([
    StructField("Zone", StringType(), True),
    StructField("District", StringType(), True),
    StructField("Fiscal Month Number", IntegerType(), True),
    StructField("Fiscal Year", IntegerType(), True),
    StructField("Month Name", StringType(), True),
    StructField("Diversion Rate-Total (Total Recycling / Total Waste)", FloatType(), True),
    StructField("Capture Rate-Paper (Total Paper / Max Paper)", FloatType(), True),
    StructField("Capture Rate-MGP (Total MGP / Max MGP)", FloatType(), True),
    StructField("Capture Rate-Total ((Total Recycling - Leaves (Recycling)) / (Max Paper + Max MGP))x100", FloatType(), True),
])

projeto_reci = spark \
            .read\
            .option("delimiter",",")\
            .option("header","true")\
            .schema(customSchema) \
            .csv(hdfs_path)
projeto_reci.toPandas()
```

Figura 33

Nas colunas indicadas abaixo, alteramos o nome destas para um nome sem espaços.

```
projeto_reci3 = projeto_reci \
    .withColumnRenamed("Fiscal Month Number","Fiscal_Month_Number") \
    .withColumnRenamed("Fiscal Year","Fiscal_Year") \
    .withColumnRenamed("Month Name","Month_Name") \
    .withColumnRenamed("Diversion Rate-Total (Total Recycling / Total Waste)","Diversion_Rate_Total") \
    .withColumnRenamed("Capture Rate-Paper (Total Paper / Max Paper)","Capture_Rate_Paper")\
    .withColumnRenamed("Capture Rate-MGP (Total MGP / Max MGP)","Capture_Rate_MGP")\
    .withColumnRenamed("Capture Rate-Total ((Total Recycling - Leaves (Recycling)) / (Max Paper + Max MGP))x100","Capture_Rate_Total")
projeto_reci3.toPandas()
```

Figura 34

Na coluna “Zone” agrupamos numa só zona as localidades que estavam divididas em norte, sul, este e oeste.

```
[11]: from pyspark.sql.types import *
      from pyspark.sql.functions import *

      projeto_reci4 = projeto_reci3.withColumn('Zone',
                  when(projeto_reci3.Zone.endswith('North'),regexp_replace(projeto_reci3.Zone,'North',' ')) \
                  .when(projeto_reci3.Zone.endswith('South'),regexp_replace(projeto_reci3.Zone,'South',' ')) \
                  .when(projeto_reci3.Zone.endswith('East'),regexp_replace(projeto_reci3.Zone,'East',' ')) \
                  .when(projeto_reci3.Zone.endswith('West'),regexp_replace(projeto_reci3.Zone,'West',' ')) \
                  .when (projeto_reci3.Zone.endswith('Manhattan'),regexp_replace(projeto_reci3.Zone,'Manhattan','Manhattan')) \
                  .when (projeto_reci3.Zone.endswith('Staten Island'),regexp_replace(projeto_reci3.Zone,'Staten Island','Staten Island')) \
                  .when (projeto_reci3.Zone.endswith('Bronx'),regexp_replace(projeto_reci3.Zone,'Bronx','Bronx')) \

      )

      projeto_reci4.toPandas()
```

Figura 35

Criação de uma tabela delta na base de dados com as colunas correspondentes ao nosso dataset e particionada por anos.

```
spark.sql(
    """
    CREATE EXTERNAL TABLE Projeto.Tabela_Reciclagem (
        Zone VARCHAR(50),
        District VARCHAR(50),
        Fiscal_Month_Number INT,
        Month_Name VARCHAR(50),
        Diversion_Rate_Total FLOAT,
        Capture_Rate_Paper FLOAT,
        Capture_Rate_MGP FLOAT,
        Capture_Rate_Total FLOAT


    )
       USING DELTA

   PARTITIONED BY (
        Fiscal_Year INT

    )
    LOCATION 'hdfs://hdfs-nn:9000/TrabalhoPL/silver/Projeto.db/Tabela_Reciclagem'
    """
)
```

Figura 36

Por fim, carregamos os dados modificados para a nossa base de dados.

```
#write df to hive deltalake_table
projeto_reci3 \
    .select("Zone","District","Fiscal_Month_Number","Fiscal_Year","Month_Name","Diversion_Rate_Total",
            "Capture_Rate_Paper","Capture_Rate_MGP","Capture_Rate_Total") \
    .write \
    .mode("overwrite") \
    .partitionBy("Fiscal_Year") \
    .format("delta") \
    .save("hdfs://hdfs-nn:9000/TrabalhoPL/silver/Projeto.db/Tabela_Reciclagem")
from pyspark.sql.types import *
```

Figura 37

Camada Gold

O objetivo desta fase final é a estruturação dos datasets, até agora analisados, de forma a obtermos respostas para as nossas questões analíticas.

Para isso começamos por carregar as respetivas tabelas silver, criadas anteriormente, para uma estrutura de dados do Spark.

```
from pyspark.sql.functions import substring, avg, sum

# read air_quality from the silver tables
hdfs_path = "hdfs://hdfs-nn:9000/TrabalhoPL/silver/Projeto.db/Tabela_Petroleo"

tabela_petroleo = spark\
            .read\
            .load(hdfs_path)

tabela_petroleo.show()
tabela_petroleo.printSchema()
```

Figura 38

De seguida, criamos a estrutura com as medidas de análise que pretendemos. Criamos, ainda, a tabela gold e importamos a estrutura para a tabela no HDFS.

```
spark.sql(
    """
    CREATE EXTERNAL TABLE Projeto_gold.Tabela_Petroleo (
        Ano INT,
        Derrames_Count LONG
    )
    USING DELTA
    LOCATION 'hdfs://hdfs-nn:9000/TrabalhoPL/gold/Projeto_gold.db/Tabela_Petroleo/'
    """
)
```

Figura 39

```
# write to delta table
gold_petroleo \
    .write  \
    .format("delta") \
    .mode("overwrite") \
    .save("hdfs://hdfs-nn:9000/TrabalhoPL/gold/Projeto_gold.db/Tabela_Petroleo/")
```

Figura 40

Por fim, de modo a podermos construir os dashboard, usando o Tableau, procedemos à criação das tabelas presto para cada tabela Gold criada.

```
spark.sql("""
GENERATE symlink_format_manifest FOR TABLE delta.`hdfs://hdfs-nn:9000/TrabalhoPL/gold/Projeto_gold.db/Tabela_Petroleo/`
""").show()
```

Figura 41

```
spark.sql("""
DROP TABLE IF EXISTS Projeto_gold.Tabela_Petroleo_Presto
""").show()

spark.sql("""
    CREATE EXTERNAL TABLE Projeto_gold.Tabela_Petroleo_Presto (
        Ano INT,
        Derrames_Count LONG
    )
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS INPUTFORMAT 'org.apache.hadoop.hive.ql.io.SymlinkTextInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION 'hdfs://hdfs-nn:9000/TrabalhoPL/gold/Projeto_gold.db/Tabela_Petroleo/_symlink_format_manifest/'
""").show()
```

Figura 42

Incidentes de Petróleo

Tabela com o número de derrames agrupados por ano.

```
from pyspark.sql.functions import count, countDistinct
gold_petroleo = tabela_petroleo \
    .groupBy("Ano")\
    .agg(
        count(tabela_petroleo.Spill_Number).alias("Derrames_Count"),
    ) \
```

Figura 43

Tabela com o número de derrames agrupado por cada localidade e por tipo de fator.

```
from pyspark.sql.functions import count, countDistinct
gold_petroleo = tabela_petroleo \
    .groupBy("Contry", "Contributing_Factor")\
    .agg(
        count(tabela_petroleo.Spill_Number).alias("Derrames_Count"),
    ) \
```

Figura 44

Tabela com o desperdício agrupado, para cada fator, e para cada tipo de material. O cálculo para o desperdício corresponde à subtração entre a quantidade perdida no derrame e a quantidade recuperada.

```
from pyspark.sql.functions import count, countDistinct
gold_petroleo = tabela_petroleo \
    .groupBy("Contributing_Factor", "Material_Family")\
    .agg(
        (sum(tabela_petroleo.Quantity)-sum(tabela_petroleo.Recovered)).alias("Desperdicio"),
    ) \
```

Figura 45

Qualidade do Ar

Tabela com a média, agrupada para cada tipo de análise e cada localidade.

```
from pyspark.sql.functions import avg
gold_air_quality = air_quality \
    .groupBy("Localidade", "Name")\
    .agg(
        avg(air_quality.Data_Value).alias("Media")
    ) \
```

Figura 46

Tabela com a média, agrupada para cada tipo de análise e para cada ano.

```
from pyspark.sql.functions import avg
gold_air_quality = air_quality \
    .groupBy("Ano", "Name")\
    .agg(
        avg(air_quality.Data_Value).alias("Media")
    ) \
```

Figura 47

Tabela com a soma dos valores de cada análise, para cada localidade e ano.

```
from pyspark.sql.functions import count
gold_air_quality = air_quality \
    .groupBy("Ano", "Localidade", "Name")\
    .agg(
        sum(air_quality.Data_Value).alias("Soma")
    ) \
```

Figura 48

Qualidade da Água da Bacia Hidrográfica

Tabela com a contagem de cada tipo de análise (Status) para cada ano.

```
from pyspark.sql.functions import count
gold_agua = agua \
    .groupBy("Ano", "Status")\
    .agg(
        count(agua.Status).alias("Contagem")
    ) \
```

Figura 49

Tabela com a média de cada analise agrupada para cada rede de rios

```
from pyspark.sql.functions import avg
gold_agua = agua \
    .groupBy("Stream_Group", "Analyte")\
    .agg(
        avg(agua.Final_Result).alias("Media")
    ) \
```

Figura 50

Taxas de Desvio e Captura de Reciclagem

Tabela com a quantidade total de papel reciclado e a quantidade total de vidro e plastico reciclado agrupados, para cada localidade.

```
from pyspark.sql.functions import count
gold_recycling = recycling \
    .groupBy("Zone")\
    .agg(
        sum(recycling.Capture_Rate_Paper).alias("Total_Papel"),
        sum(recycling.Capture_Rate_MGP).alias("Total_Vidro_Plastico")
    ) \
```

Figura 51

Tabela da média de lixo reciclado comum e a média de lixo total reciclado, agrupadas para cada ano e mês.

```
from pyspark.sql.functions import count
gold_recycling = recycling \
    .groupBy("Fiscal_Year","Month_Name")\
    .agg(
        avg(recycling.Capture_Rate_Total).alias("Media_Reciclado_Comum"),
        avg(recycling.Diversion_Rate_Total).alias("Media_Reciclagem")
    ) \
```

Figura 52

Questões Gerais

Para criar as tabelas para as questões gerais, utilizamos as tabelas gold (criadas previamente) que contém a informação necessária para responder à pergunta e fazemos o JOIN entre as tabelas, através do ano ou localidade, consoante aquilo que é pretendido analisar, obtendo assim a tabela que queremos.

```
geral = agua.join(petroleo, (agua.Ano == petroleo.Ano),"inner")
```

Figura 53

Tableau

Incidentes de Petróleo – Respostas às questões analíticas através do Tableau

1 – Quais as localidades com maior número de derrames e quais fatores que mais contribuíram para estes?

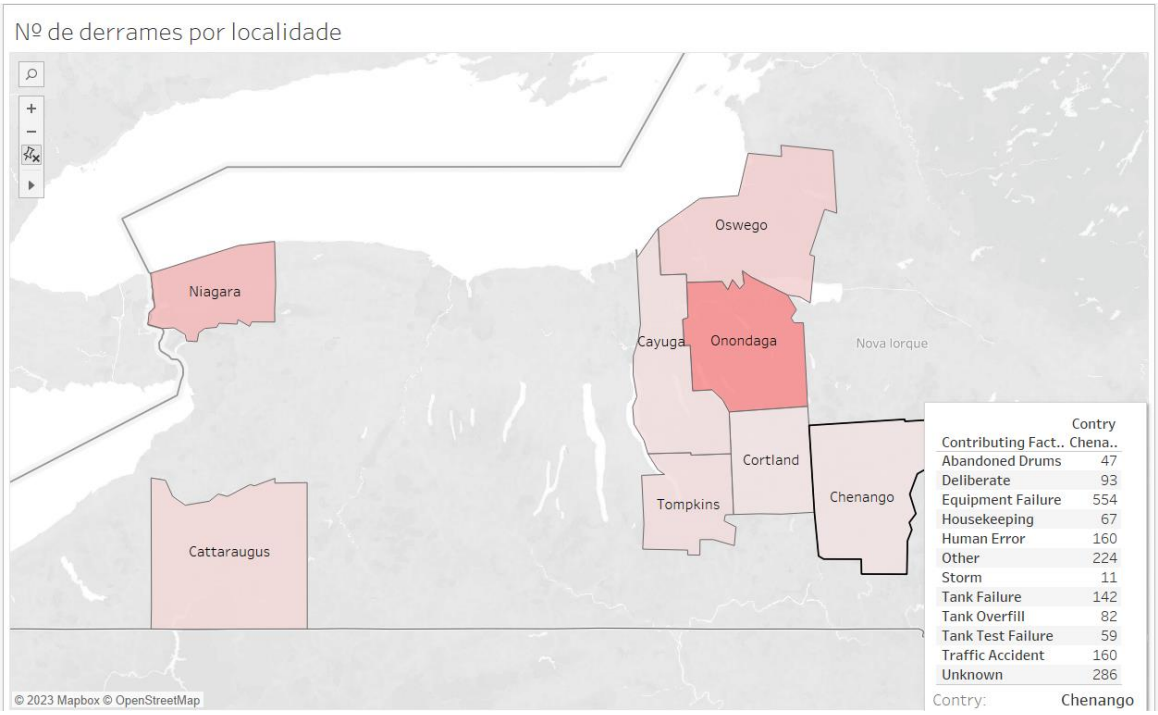


Figura 54

Através da análise do seguinte gráfico podemos concluir que Onondaga e Niagara são as localidades que apresentam um maior nº de derrames. Podemos, também, observar quais os fatores que originaram estas quantidades de derrames.

2 – Qual a percentagem de derrame ao longo do século XXI?

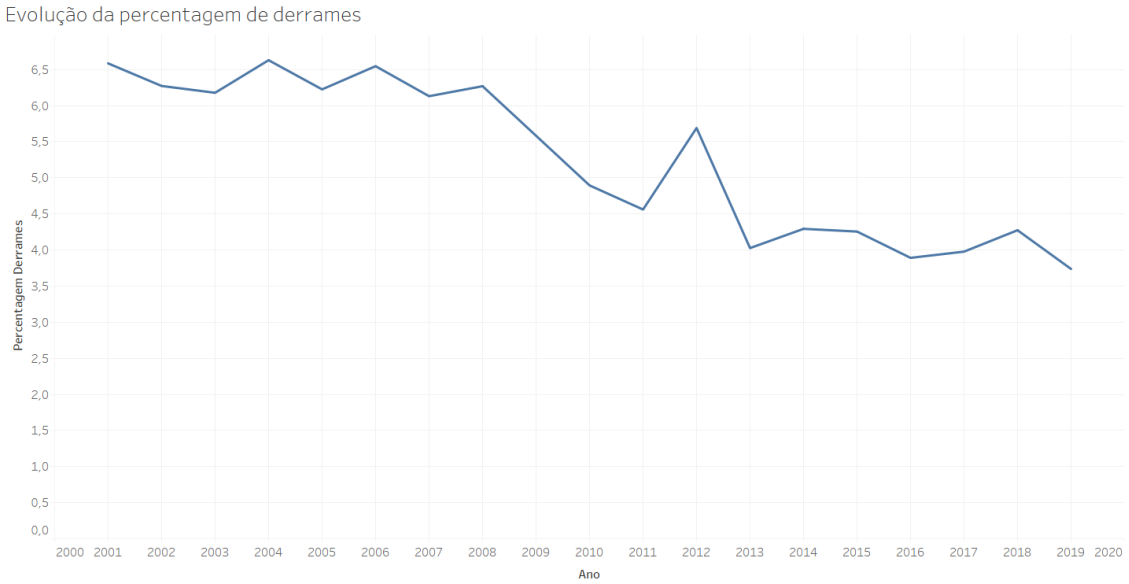


Figura 55

Através da análise do gráfico podemos concluir que a percentagem de derrames ao longo do século XXI diminui de cerca de 6% para 3%, apesar das diversas oscilações ao longo deste periodo.

3 – Quais as fontes que originaram um maior desperdício de resíduos?

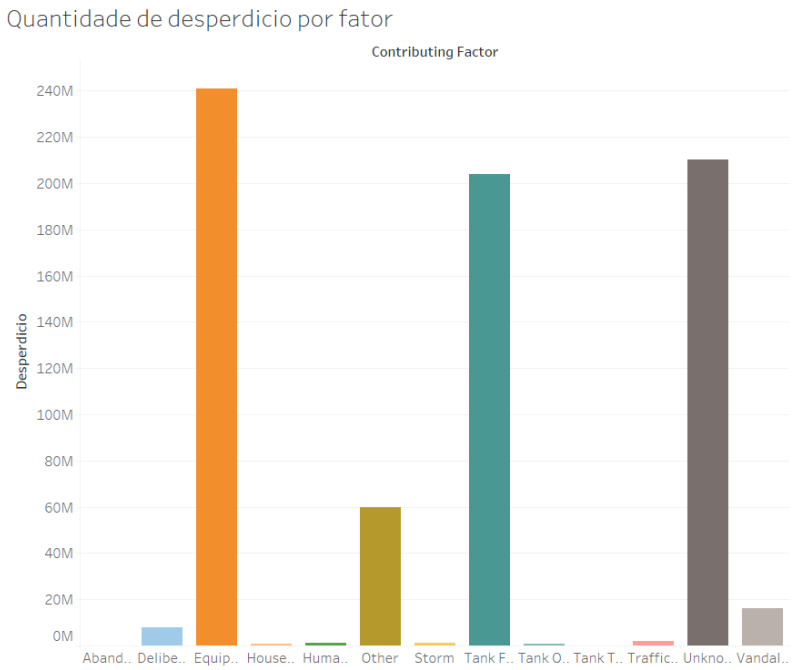


Figura 56

O gráfico anterior permite-nos concluir que a falha de equipamentos foi o fator que originou um maior desperdicio de residuos petroliferos.

KPI: Diminiuição de 1% na variação relativa da ocorrência de derrames no ano de 2015 em comparação ao de 2016.

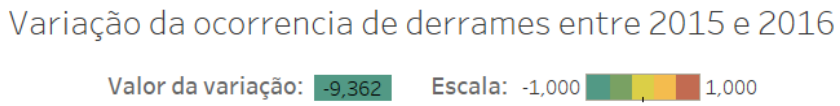


Figura 57

Para o cálculo da variação foi utilizada a seguinte fórmula:

$$\frac{nº\ derrames\ 2016 - nº\ derrames\ 2015}{nº\ derrames\ 2015} * 100$$

Uma vez que tivemos uma redução de cerca de 9% no nº de derrames no ano de 2015 para o ano de 2016, podemos concluir que o nosso objetivo foi atingido.

Dashboard do dataset:

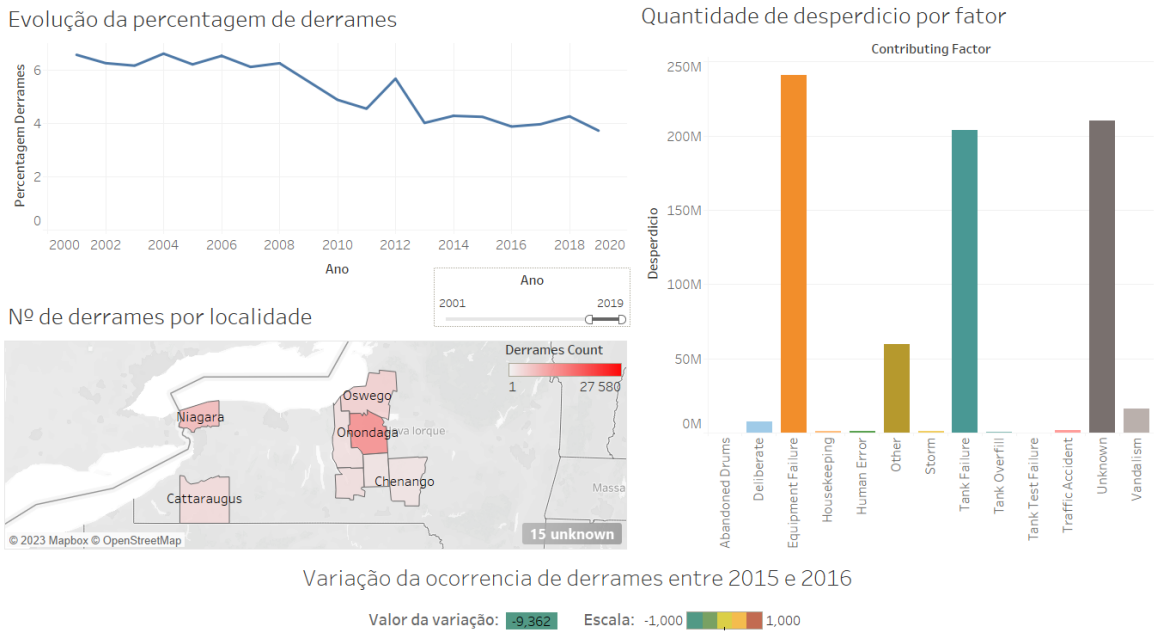


Figura 58

Qualidade do Ar – Respostas às questões analíticas através do Tableau

1 – Quais os gases com efeitos cancerígenos (Benzene e Formaldehyde) que estão mais presentes em cada uma das localidades?

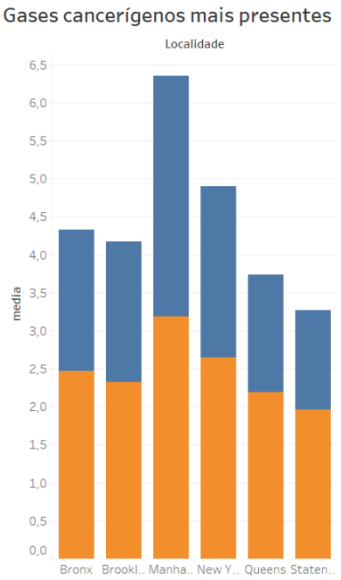


Figura 59

Através da analise do grafico podemos concluir que em todas as localidade, o Formaldehyde é o gás que se encontra mais presente na atmosfera.

2 – Quais os períodos de tempo em que a presença do gás O_3 foi mais notória?

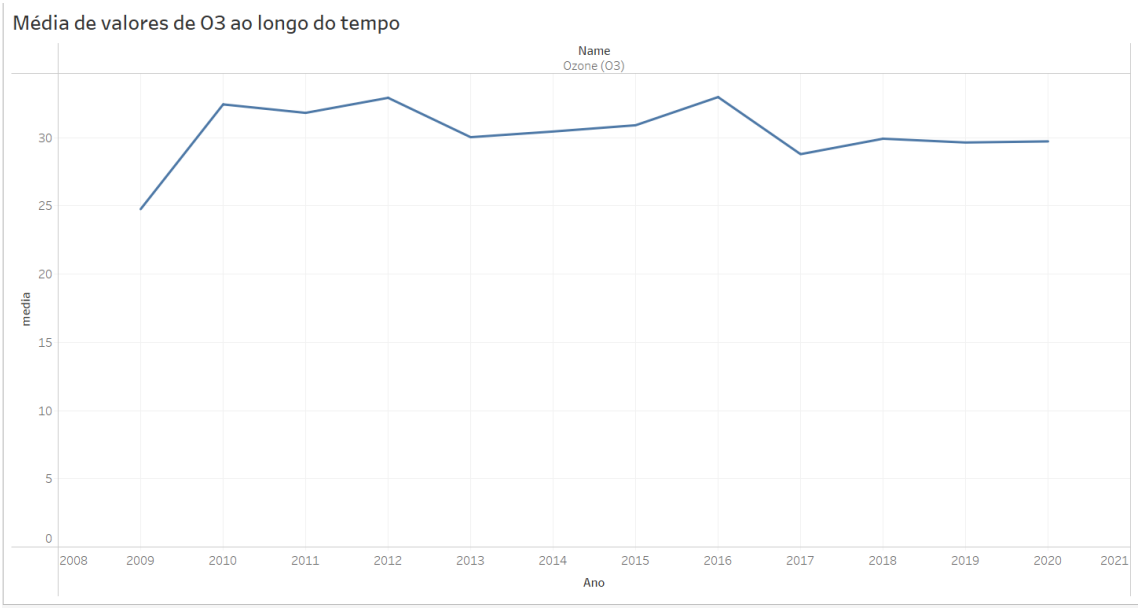


Figura 60

Com a análise do gráfico acima, conseguimos perceber que nos anos 2010, 2012 e 2016 os valores de O_3 na atmosfera atingiram os seus máximos. Contudo, apartir deste último ano, os seus valores têm vindo a reduzir.

3 – Quais os principais efeitos, na população, resultantes da exposição ao PM2.5, por cada localidade?

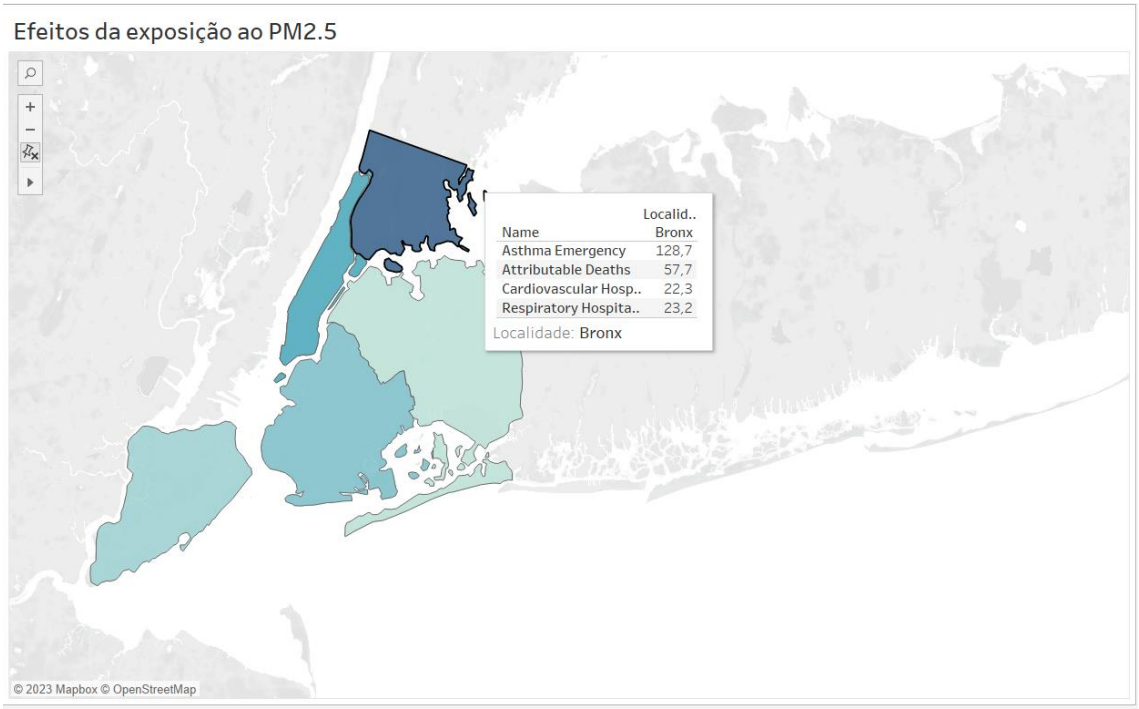


Figura 61

Através da análise do mapa acima, verificamos que a localidade mais afetada pela exposição ao PM2.5, é Brox, com uma média de 128,7 hospitalizações por asma em comparação aos restantes distritos.

KPI: Na zona geografica de Borough, diminuição de 1% na variação dos níveis de poluição do ar entre 2009 e 2015.

Figura 62

Para o cálculo da variação foi utilizada a seguinte fórmula:

$$\frac{média\_níveis\_2015 - média\_níveis\_2009}{média\_níveis\_2009} * 100$$

Uma vez que tivemos um aumento de cerca de 9% nos niveis de pouição do ar entre 2009 e 2015, podemos concluir que o nosso objetivo não foi concluído.

Dashoard do dataset:

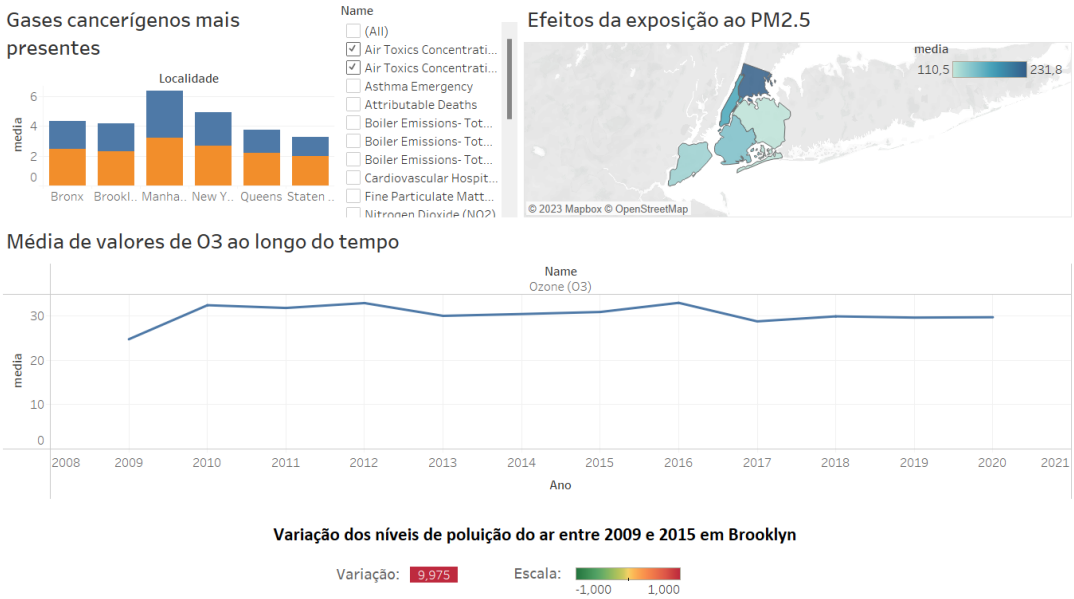


Figura 63

Qualidade da Água da Bacia Hidrográfica – Respostas às questões analíticas através do Tableau

1 – Qual a média de pH, turbidity e temperature (tipos de análise) em cada uma das redes de rios?

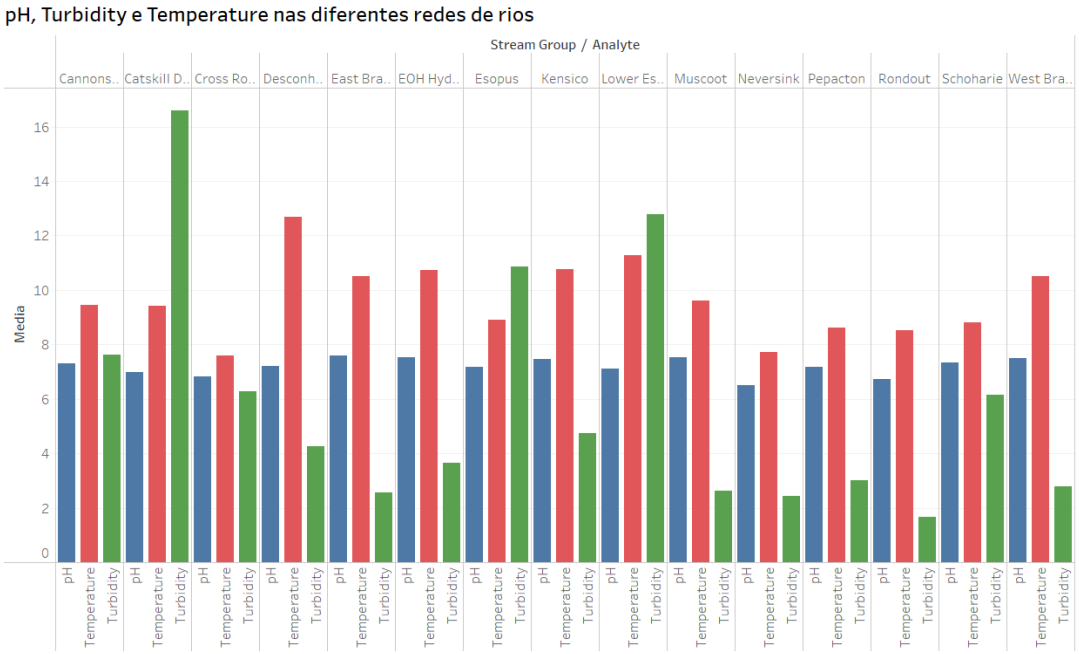


Figura 64

Com a análise do gráfico de barras podemos concluir que a temperatura e a turbidez têm um valor médio superior ao pH em quase todas as redes de rios.

2 – Quais os anos com maior número de recolhas de amostras concluídas?

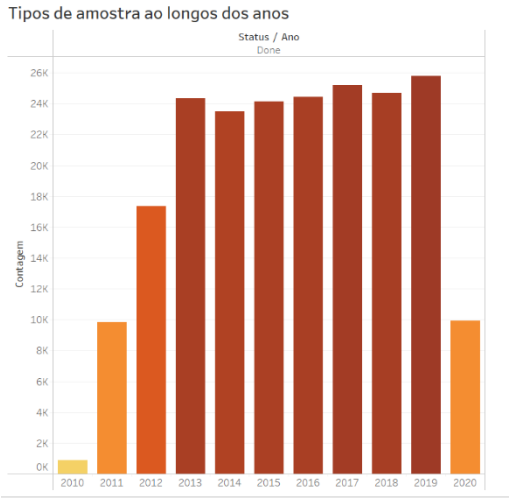


Figura 65

Analisando o grafico de barras acima podemos afirmar que 2019 e 2017 foram os anos com um maior numero de amostras concluidas.

3 – Qual a evolução do pH, ao longos dos últimos 10 anos, nas diferentes redes de rios?

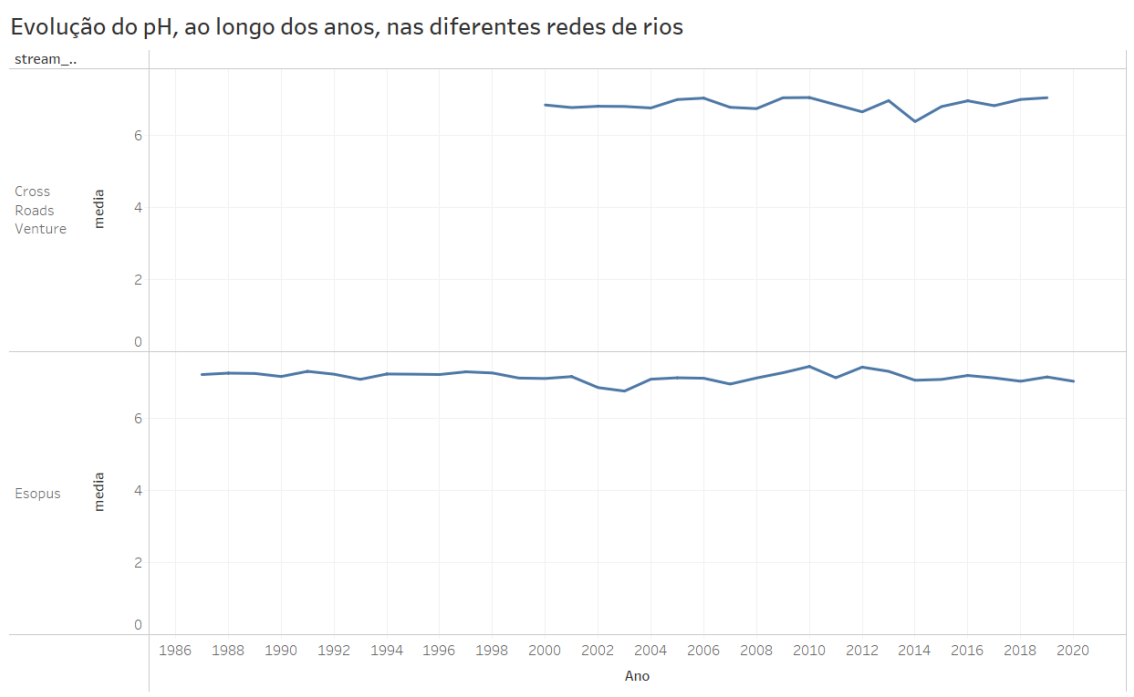


Figura 66

Podemos concluir que, na rede de rios Cross Roads Venture o valor médio do pH obteve valores mais elevados no ano de 2013. Já na rede de rios Esopus, o valor da média do pH obteve valores elevados nos anos 2010 e 2012.

KPI: Na rede de rios Esopus, a variação do pH entre 2018 e 2019.

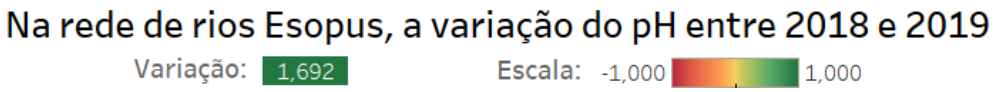


Figura 67

Obtivemos o valor da variação através da seguinte fórmula:

$$\frac{média\_pH\_2019 - média\_pH\_2018}{média\_pH\_2018} * 100$$

Como o valor obtido foi acima de 1%, então concluimos que o nosso objetivo foi alcançado.

Dashboard do dataset:

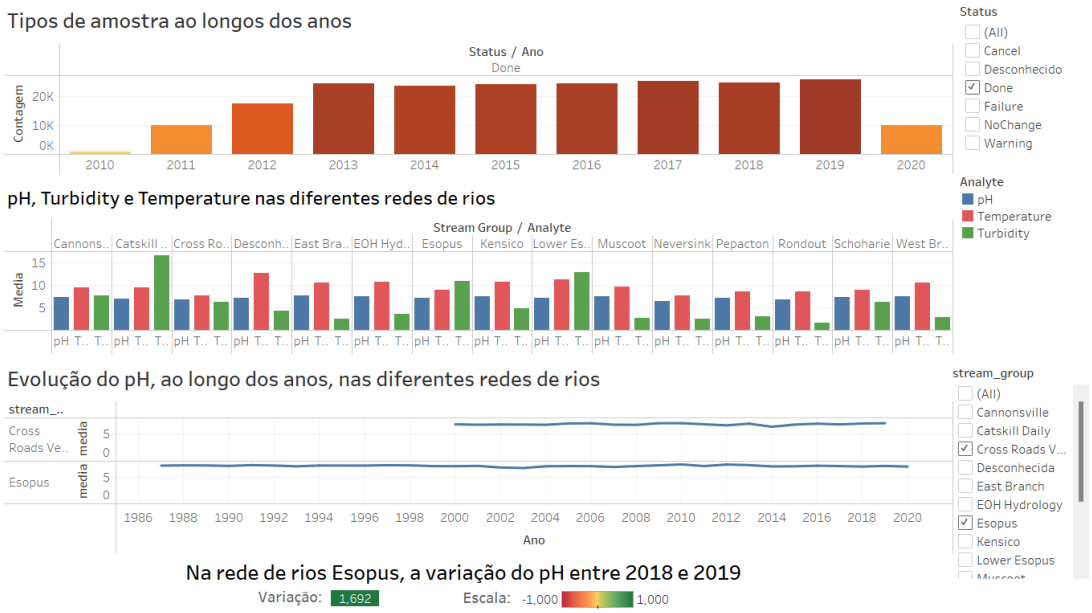


Figura 68

Taxas de Desvio e Captura de Reciclagem – Respostas às questões analíticas através do Tableau

1 – Quais os anos em que o rácio entre o lixo recliclado e o lixo comum foi mais elevado?

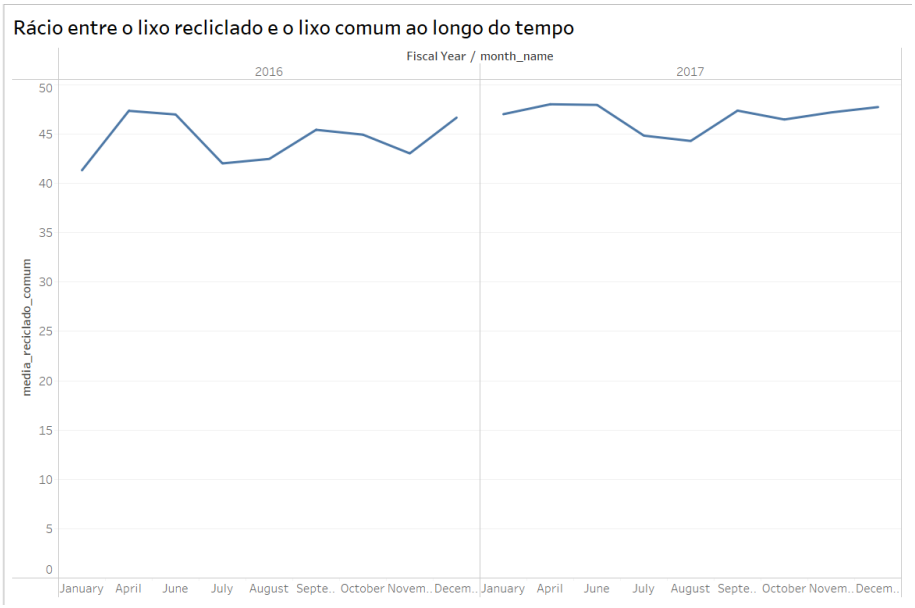


Figura 69

Entre o ano de 2016 e 2017, podemos concluir que o rácio entre o lixo reciclado e o lixo comum obteve maiores valores no ano de 2017.

2 – Qual o tipo de resíduos mais reciclados em cada uma das localidades ?

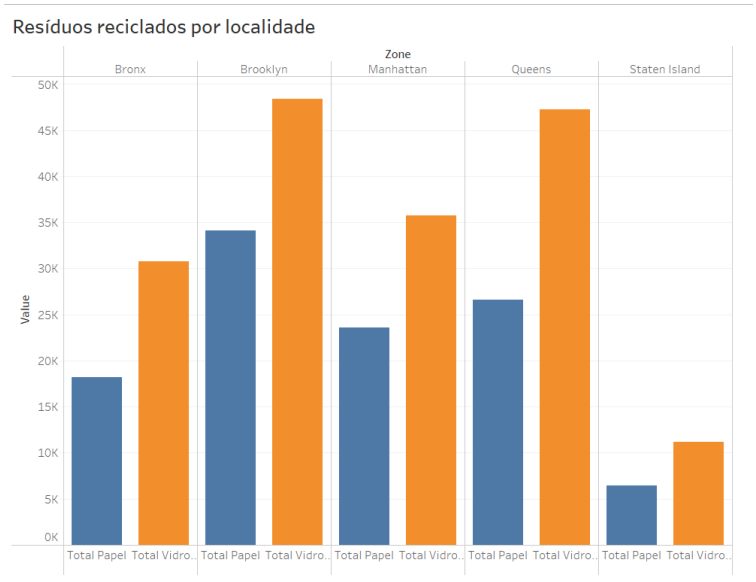


Figura 70

Concluimos que, em todas as localidades, os resíduos mais reciclados são o Vidro e o Plástico.

3 – Qual o período de tempo no qual ocorreu um maior aumento da percentagem de reciclagem?

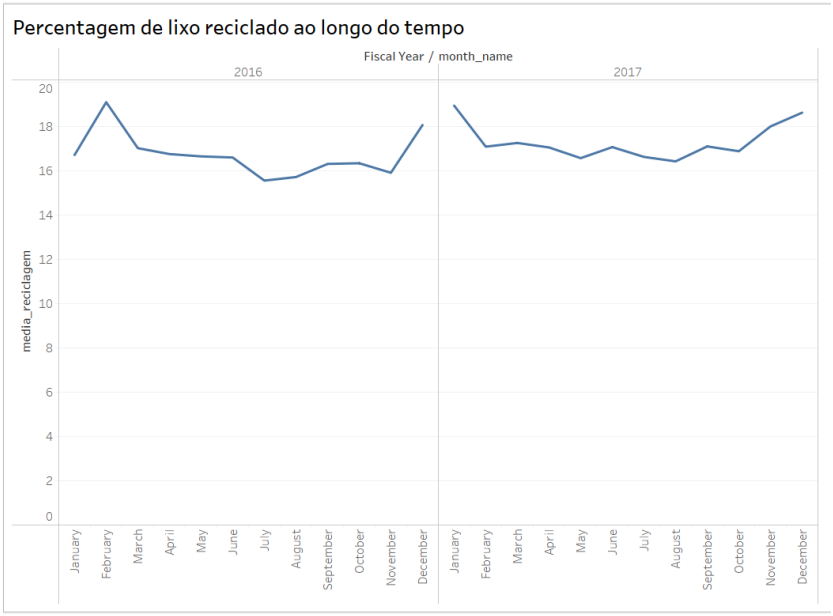


Figura 71

No ano de 2016 notou-se uma maior percentagem de reciclagem nos meses de fevereiro, e depois de uma diminuição até ao mês de julho, esta volta a subir até ao mês de setembro e torna a haver uma subida acentuada durante o mês de novembro. No ano de 2017, há um aumento da percentagem de reciclagem nos meses de fevereiro, maio, agosto e de outubro até ao fim do ano.

KPI: Variação do rácio total de reciclagem recolhida entre o primeiro semestre de 2017 e o segundo semestre deste mesmo ano.

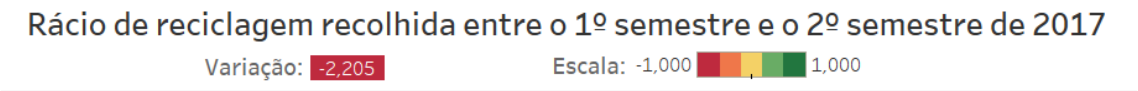


Figura 72

Para o cálculo da variação utilizamos a fórmula:

$$\frac{reciclagem\_2ºsem - reciclagem\_1ºsem}{reciclagem\_1ºsem} * 100$$

Como a variação tem valor negativo, concluimos que o objetivo não foi atingido.

Dashboard do dataset:

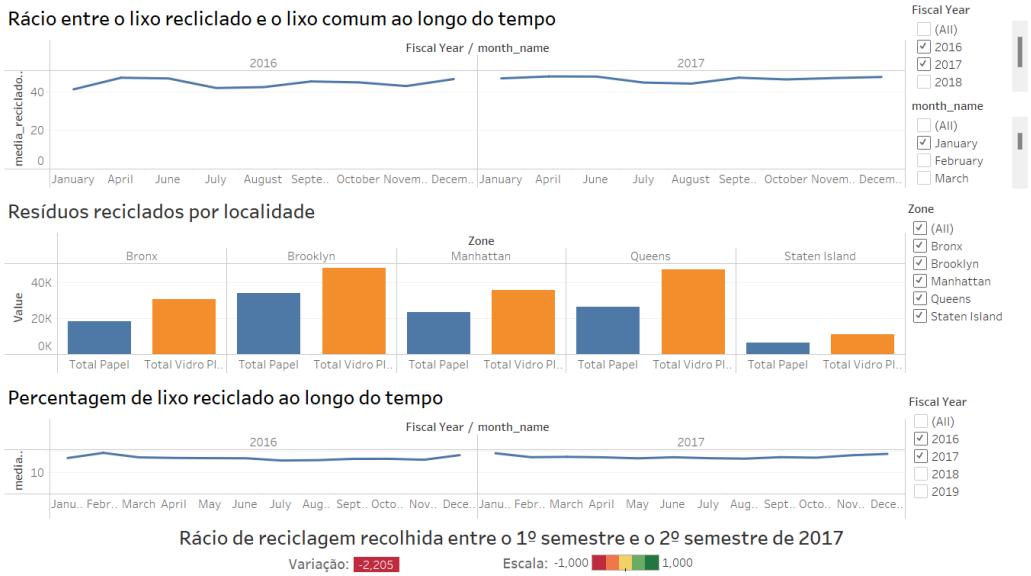


Figura 73

Questões analíticas gerais - Respostas às questões analíticas através do Tableau

1 – Em que anos, o número de derrames afetou o pH da água? (Petróleo e Água)

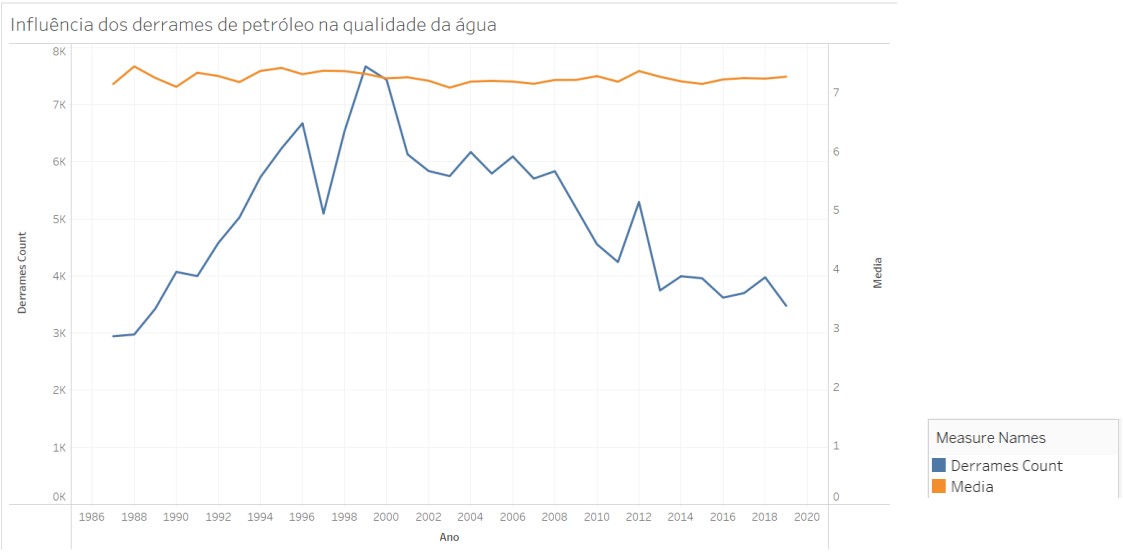


Figura 74

Concluimos daqui que não houve qualquer influência, por parte dos derrames de petóleo, no pH da água.

2 – Qual a relação entre os níveis de ozono (O_3) e a temperatura da água nos ultimos 5 anos? (Ar e Água)

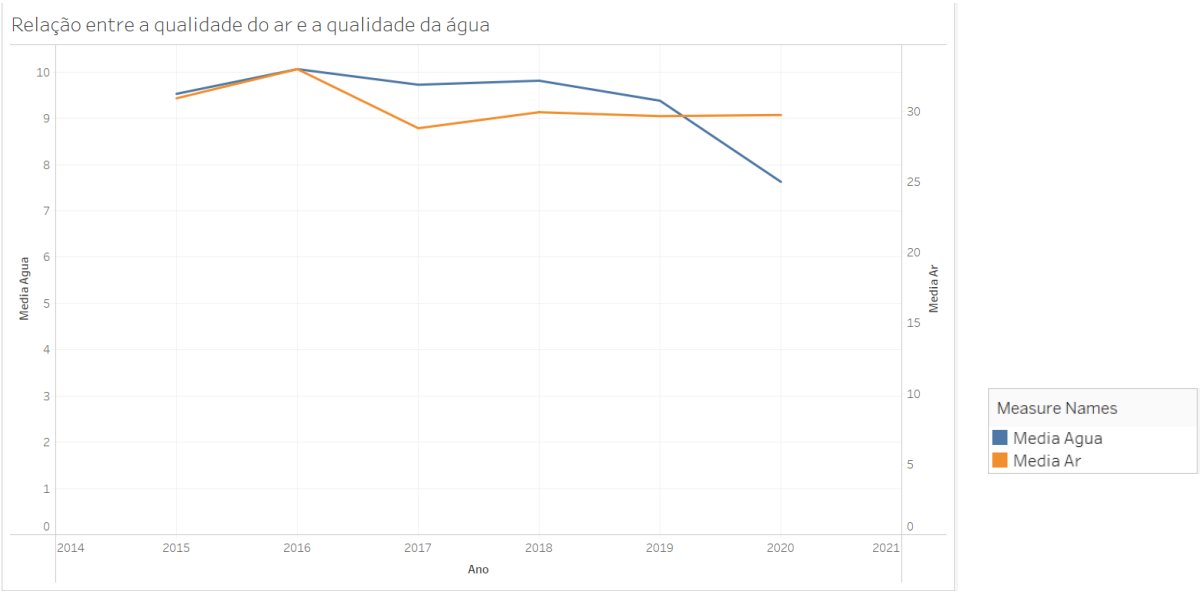


Figura 75

Seria expectável que um aumento do nível de ozono resultasse num aumento da temperatura da água, bem como o contrário. O que se verificou até ao ano de 2019.

3 – Quais as zonas em que um maior rácio entre o lixo recliclado e o lixo comum implicou um menor nível de ozono? (Reciclagem e Ar)

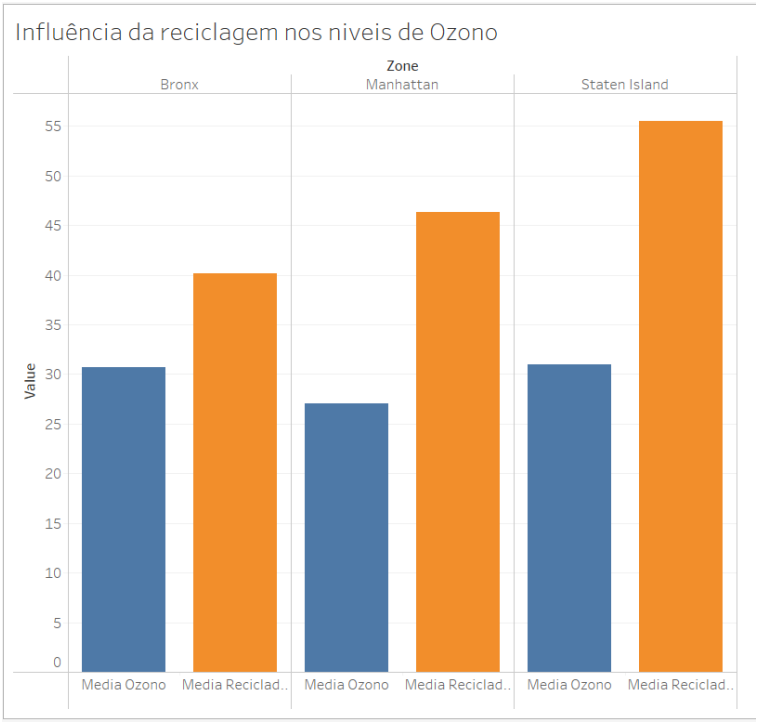


Figura 76

É previsível que um aumento da percentagem da reciclagem faça com que os níveis de ozono reduzam, o que é notável nas zonas analisadas.

Dashboard geral

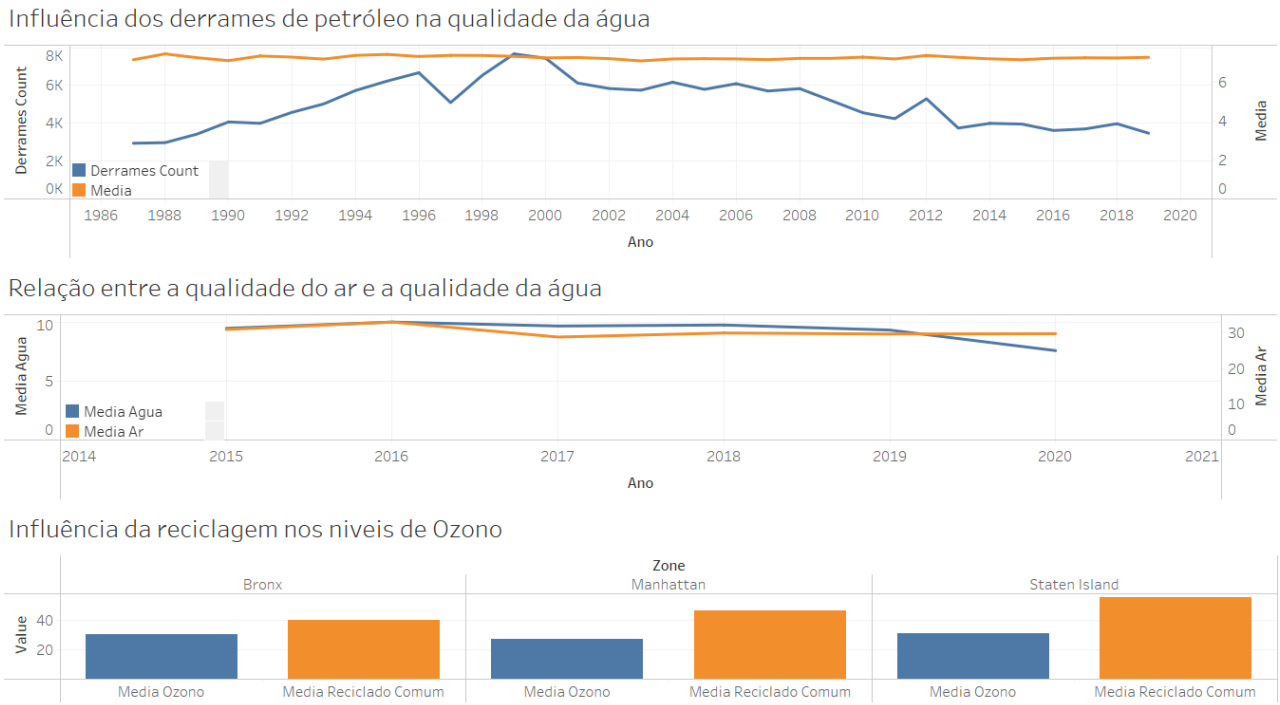
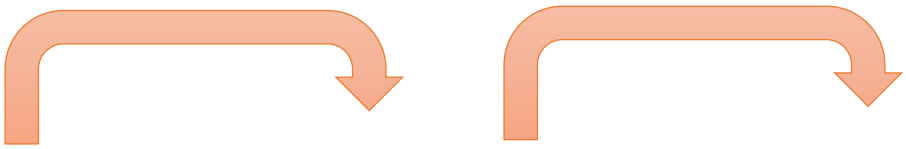


Figura 77

Diagrama de Pipelines



BRONZE	SILVER	GOLD
• Upload dos datasets para o HDFS. • Identificação dos erros dos dados no nosso dataset.	• Correção dos erros identificados na camada anterior; • Alteração das datas para o mesmo formato; • Renomeação das colunas; • Particionamento das tabelas por anos.	• Agregação das tabelas; • Remoção de atributos irrelevantes.

Link do vídeo

Sustentabilidade em Nova York: https://www.youtube.com/watch?v=Ytvnaoy_nZs