Código Fonte

Aqui está tudo o código que utilizei para entregar a analise a Tokio School Viagens.

1. Temos o código do pre processamento.

```
import pandas as pd
```

[1] Python

```
df = pd.read_csv("/Users/tiagosilva/Air_Traffic_Passenger_Statistics.csv")

df.head()
```

[2] Python

	Activity Period	Operating Airline	Operating Airline IATA Code	Published Airline	Published Airline IATA Code	GEO Summary	GEO Region	Activity Type Code	Price Category Code	Terminal	Boarding Area	Passenger Count	Adjusted Activity Type Code	Adjusted Passenger Count	Year	Month
0	200507	ATA Airlines	TZ	ATA Airlines	TZ	Domestic	US	Deplaned	Low Fare	Terminal 1	B	27271	Deplaned	27271	2005	July
1	200507	ATA Airlines	TZ	ATA Airlines	TZ	Domestic	US	Enplaned	Low Fare	Terminal 1	B	29131	Enplaned	29131	2005	July
2	200507	ATA Airlines	TZ	ATA Airlines	TZ	Domestic	US	Thru / Transit	Low Fare	Terminal 1	B	5415	Thru / Transit * 2	10830	2005	July
3	200507	Air Canada	AC	Air Canada	AC	International	Canada	Deplaned	Other	Terminal 1	B	35156	Deplaned	35156	2005	July
4	200507	Air Canada	AC	Air Canada	AC	International	Canada	Enplaned	Other	Terminal 1	B	34090	Enplaned	34090	2005	July

```
# VISUALIZAR VALORES NULOS

valores_nulos =  df.isnull().sum()

print(valores_nulos)
```

[3] Python

```
Activity Period                0
Operating Airline              0
Operating Airline IATA Code    54
Published Airline              0
Published Airline IATA Code    54
GEO Summary                    0
GEO Region                     0
Activity Type Code             0
Price Category Code            0
Terminal                       0
Boarding Area                  0
Passenger Count                0
Adjusted Activity Type Code    0
Adjusted Passenger Count       0
Year                           0
Month                          0
dtype: int64
```

```
# VERIFICAR DUPLICADOS

duplicados = df.duplicated().sum()

print(duplicados)
```

[4] Python

```
0
```

```
# ESTATÍSTICAS DESCRITIVAS

estatistica_descritivas = df.describe()

print(estatistica_descritivas)
```

[5] Python

```
       Activity Period  Passenger Count  Adjusted Passenger Count  \
count     15007.000000     15007.000000              15007.000000
mean     201045.073366     29240.521090              29331.917105
std         313.336196     58319.509284              58284.182219
min      200507.000000         1.000000                  1.000000
25%      200803.000000      5373.500000               5495.500000
50%      201011.000000      9210.000000               9354.000000
75%      201308.000000     21158.500000              21182.000000
max      201603.000000    659837.000000             659837.000000

              Year
count  15007.000000
mean    2010.385220
std        3.137589
min     2005.000000
25%     2008.000000
50%     2010.000000
75%     2013.000000
max     2016.000000
```

```python
colunas_categoricas = df.select_dtypes(include=["object"]).columns
valores_unicos_categoricos = {col: df[col].unique()for col in colunas_categoricas}

print(valores_unicos_categoricos)
```

[7] Python

```
{'Operating Airline': array(['ATA Airlines', 'Air Canada ', 'Air China', 'Air France',
       'Air New Zealand', 'AirTran Airways', 'Alaska Airlines',
       'All Nippon Airways', 'American Airlines',
       'American Eagle Airlines', 'Asiana Airlines',
       'Atlantic Southeast Airlines', 'BelAir Airlines',
       'British Airways', 'Cathay Pacific', 'China Airlines',
       'Delta Air Lines', 'EVA Airways', 'Frontier Airlines',
       'Hawaiian Airlines', 'Horizon Air ', 'Icelandair',
       'Independence Air', 'Japan Airlines', 'KLM Royal Dutch Airlines',
       'Korean Air Lines', 'Lufthansa German Airlines', 'Mesa Airlines',
       'Mexicana Airlines', 'Midwest Airlines', 'Northwest Airlines',
       'Philippine Airlines', 'Singapore Airlines', 'SkyWest Airlines',
       'Sun Country Airlines', 'TACA', 'US Airways', 'United Airlines',
       'United Airlines - Pre 07/01/2013', 'Virgin Atlantic',
       'WestJet Airlines', 'Boeing Company', 'Miami Air International',
       'Air Canada Jazz', 'Qantas Airways', 'Ameriflight',
       'Spirit Airlines', 'Xtra Airways',
       'Evergreen International Airlines', 'Aeromexico',
       'JetBlue Airways ', 'ExpressJet Airlines', 'Southwest Airlines',
       'Virgin America', 'Aer Lingus', 'Allegiant Air', 'Jet Airways',
       'Emirates ', 'Mesaba Airlines', 'World Airways', 'Air Berlin',
       'Republic Airlines', 'Servisair', 'Pacific Aviation',
       'Swiss International', 'LAN Peru', 'Swissport USA',
       'XL Airways France', 'China Eastern', 'SAS Airlines',
       'Atlas Air, Inc', 'Compass Airlines', 'Etihad Airways',
...
       'Central America', 'Middle East', 'South America'], dtype=object), 'Activity Type Code': array(['Deplaned', 'Enplaned', 'Thru / Transit'], dtype=object), 'Price Category Code': array(['Low Fare', 'Other'], dtyp
      dtype=object), 'Boarding Area': array(['B', 'G', 'A', 'E', 'C', 'F', 'Other', 'D'], dtype=object), 'Adjusted Activity Type Code': array(['Deplaned', 'Enplaned', 'Thru / Transit * 2'], dtype=object), 'Month': ar
       'December', 'January', 'February', 'March', 'May', 'June'],
      dtype=object)}
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

```python
# APLICAR strip()e lower() NAS COLUNAS CATEGÓRICAS PARA REMOVER ESPAÇOS E PADRONIZAR

df[colunas_categoricas] = df[colunas_categoricas].apply(lambda x: x.str.strip().str.lower())

print(valores_unicos_categoricos)
```

[8] Python

```
{'Operating Airline': array(['ATA Airlines', 'Air Canada ', 'Air China', 'Air France',
       'Air New Zealand', 'AirTran Airways', 'Alaska Airlines',
       'All Nippon Airways', 'American Airlines',
       'American Eagle Airlines', 'Asiana Airlines',
       'Atlantic Southeast Airlines', 'BelAir Airlines',
       'British Airways', 'Cathay Pacific', 'China Airlines',
       'Delta Air Lines', 'EVA Airways', 'Frontier Airlines',
       'Hawaiian Airlines', 'Horizon Air ', 'Icelandair',
       'Independence Air', 'Japan Airlines', 'KLM Royal Dutch Airlines',
       'Korean Air Lines', 'Lufthansa German Airlines', 'Mesa Airlines',
       'Mexicana Airlines', 'Midwest Airlines', 'Northwest Airlines',
       'Philippine Airlines', 'Singapore Airlines', 'SkyWest Airlines',
       'Sun Country Airlines', 'TACA', 'US Airways', 'United Airlines',
       'United Airlines - Pre 07/01/2013', 'Virgin Atlantic',
       'WestJet Airlines', 'Boeing Company', 'Miami Air International',
       'Air Canada Jazz', 'Qantas Airways', 'Ameriflight',
       'Spirit Airlines', 'Xtra Airways',
       'Evergreen International Airlines', 'Aeromexico',
       'JetBlue Airways ', 'ExpressJet Airlines', 'Southwest Airlines',
       'Virgin America', 'Aer Lingus', 'Allegiant Air', 'Jet Airways',
       'Emirates ', 'Mesaba Airlines', 'World Airways', 'Air Berlin',
       'Republic Airlines', 'Servisair', 'Pacific Aviation',
       'Swiss International', 'LAN Peru', 'Swissport USA',
       'XL Airways France', 'China Eastern', 'SAS Airlines',
       'Atlas Air, Inc', 'Compass Airlines', 'Etihad Airways',
...
       'Central America', 'Middle East', 'South America'], dtype=object), 'Activity Type Code': array(['Deplaned', 'Enplaned', 'Thru / Transit'], dtype=object), 'Price Category Code': array(['Low Fare', 'Other'], dtyp
      dtype=object), 'Boarding Area': array(['B', 'G', 'A', 'E', 'C', 'F', 'Other', 'D'], dtype=object), 'Adjusted Activity Type Code': array(['Deplaned', 'Enplaned', 'Thru / Transit * 2'], dtype=object), 'Month': ar
       'December', 'January', 'February', 'March', 'May', 'June'],
      dtype=object)}
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

```python
# REMOVER OS VALORES NULOS

df_sem_nulos = df.dropna()

df_sem_nulos.isnull().sum(), df_sem_nulos.shape
```

[9] Python

```
(Activity Period               0
 Operating Airline             0
 Operating Airline IATA Code   0
 Published Airline             0
 Published Airline IATA Code   0
 GEO Summary                   0
 GEO Region                    0
 Activity Type Code            0
 Price Category Code           0
 Terminal                      0
 Boarding Area                 0
 Passenger Count               0
 Adjusted Activity Type Code   0
 Adjusted Passenger Count      0
 Year                          0
 Month                         0
 dtype: int64,
 (14953, 16))
```

```python
df.to_csv("/Users/tiagosilva/backup_air_traffic.csv", index=False)

print("Arquivo CSV guardado com sucesso!")
```

[10] Python

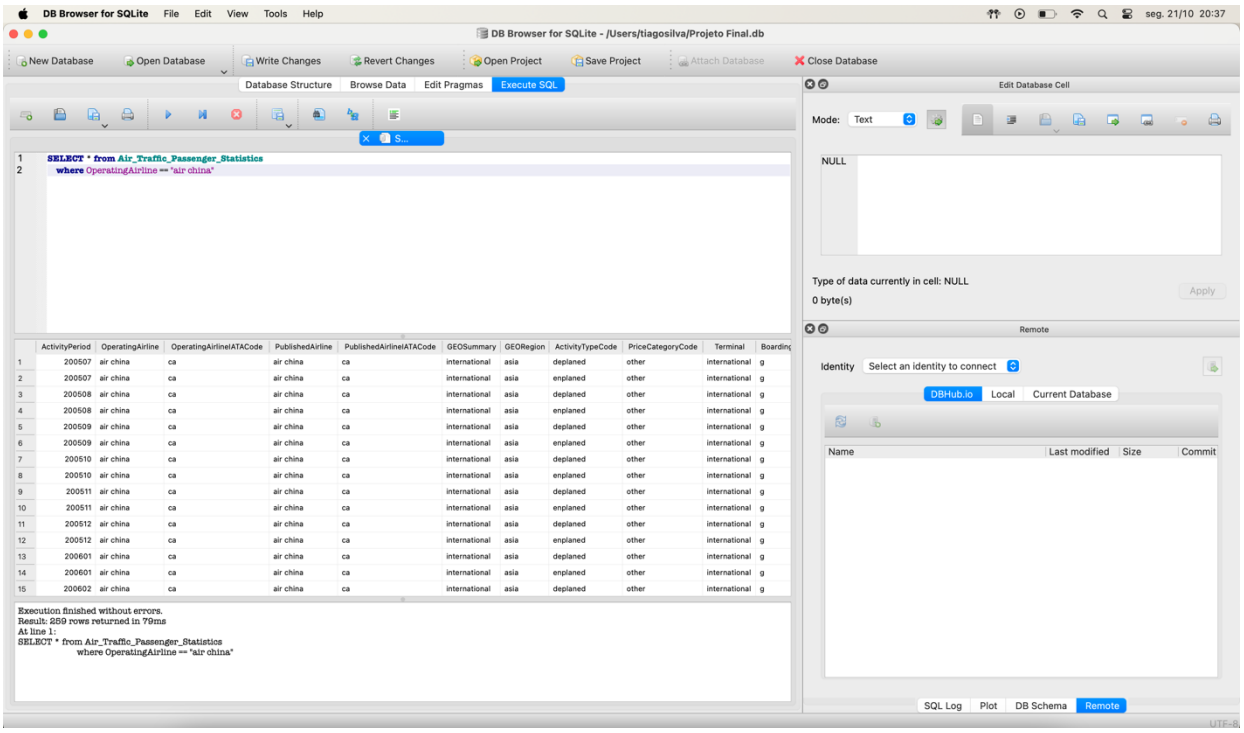
```
Arquivo CSV guardado com sucesso!
```

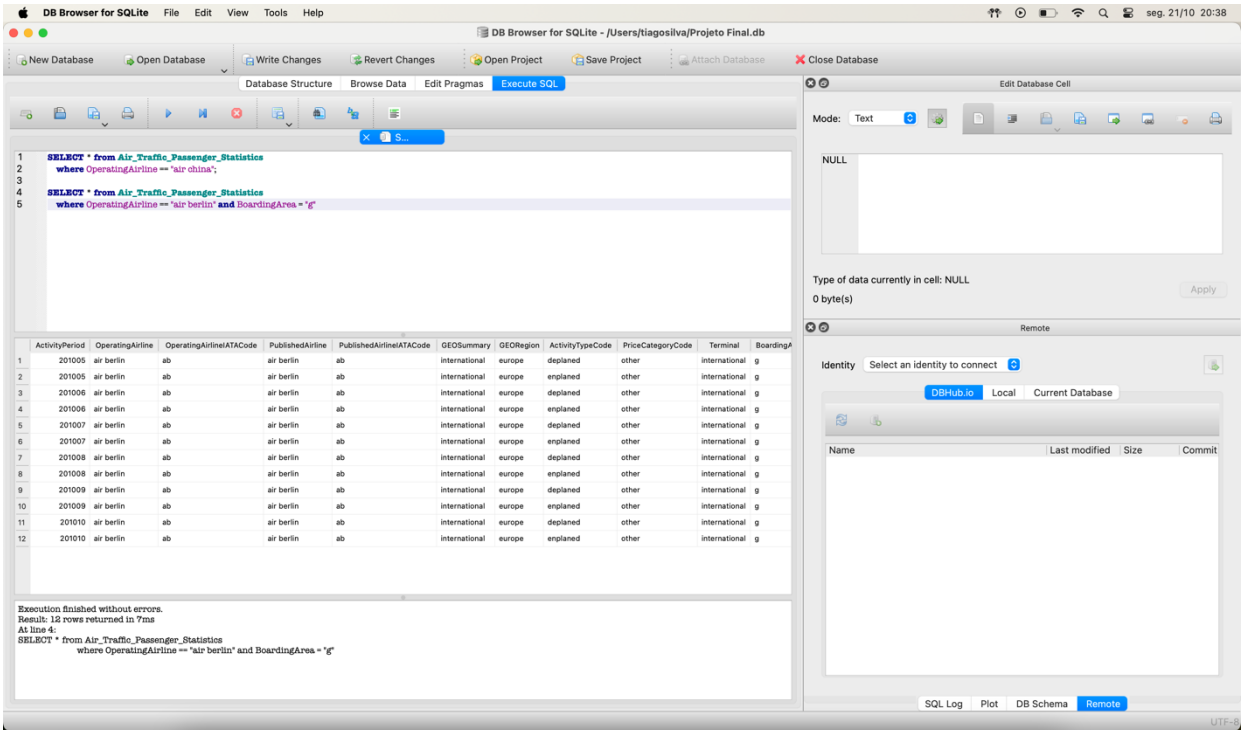
```python
df.to_csv("/Users/tiagosilva/Air_Traffic_Passenger_Statistics.csv", index=False)

print("Arquivo CSV substituido com sucesso!")
```

[11] Python

```
Arquivo CSV substituido com sucesso!
```

2. Código para mostrar os dois pedidos detalhados.





3. Código PySpark

1 Jupyter Notebook

[7]:
```
from pyspark.sql import SparkSession
```

[9]:
```
spark = SparkSession.builder \
    .appName("ProjetoFinal1") \
    .config("spark.driver.bindAddress", "127.0.0.1") \
    .getOrCreate()
```

[10]:
```
# Carregar o arquivo CSV em um DataFrame
df = spark.read.option("header", "true").csv("/Users/tiagosilva/Air_Traffic_Passenger_Statistics.csv")

# Visualizar os primeiros registros
df.show(5)
```

[11]:
```
# Contar companhias aéreas distintas
df.select("Operating Airline").distinct().count()
```

[13]:
```
import matplotlib.pyplot as plt
```

[34]:
```
import pandas as pd
```

[33]:
```
# Calcular a média de passageiros por companhia aérea
media1 = df.groupBy("Operating Airline").agg({"Passenger Count": "avg"})

media1.show()

# Calcular a média de passageiros por companhia aérea
media = df.groupBy("Operating Airline").agg({"Passenger Count": "avg"}).toPandas()

# Criar gráfico com os 20 principais
media_top20 = media.head(20)

plt.figure(figsize=(10, 6))
plt.barh(media_top20['Operating Airline'], media_top20['avg(Passenger Count)'], color="skyblue")
plt.xlabel("Average Passenger Count")
plt.title("Average Number of Passengers by Airline")
plt.tight_layout()

# Exibir gráfico
plt.show()
```

[35]:
```
import pyspark.sql.functions as F
# Ordenar o DataFrame pela coluna "Região GEO" e "Passageiros" de forma decrescente
df_sorted = df.orderBy("GEO Region", F.desc("Passenger Count"))

# Eliminar duplicados com base na coluna "Região GEO", mantendo o primeiro (que terá o maior número de passageiros)
sem_duplicados = df_sorted.dropDuplicates(["GEO Region"])

# Mostrar o DataFrame final
sem_duplicados.show()
```

[37]:
```
# Salvar o DataFrame em CSV
media1.write.option("header", "true").csv("/Users/tiagosilva/media")
sem_duplicados.write.option("header", "true").csv("/Users/tiagosilva/sem_duplicados")
```

2 Jupyter Notebook

[2]:
```
from pyspark.sql import SparkSession

spark = SparkSession.builder \
    .appName("ProjetoFinal2") \
    .config("spark.driver.bindAddress", "127.0.0.1") \
    .getOrCreate()
```

```
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
24/10/23 21:43:23 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicabl
e
24/10/23 21:43:24 WARN Utils: Service 'SparkUI' could not bind on port 4040. Attempting port 4041.
```

[3]:
```
# Carregar o arquivo CSV em um DataFrame
df = spark.read.option("header", "true").csv("/Users/tiagosilva/Air_Traffic_Passenger_Statistics.csv")
```

[27]:
```
# Vamos calcular as médias e desvios padrão por diferentes categorias

# Número de passageiros por ano
group_by_year = df.groupBy('Year').agg({'Passenger Count': 'avg', 'Passenger Count': 'stddev'}).toPandas()

# Número de passageiros por mês
group_by_month = df.groupBy('Month').agg({'Passenger Count': 'avg', 'Passenger Count': 'stddev'}).toPandas()

# Número de passageiros por companhia aérea
group_by_airline = df.groupBy('Operating Airline').agg({'Passenger Count': 'avg', 'Passenger Count': 'stddev'}).toPandas()

# Número de passageiros por tipo de atividade
group_by_activity = df.groupBy('Activity Type Code').agg({'Passenger Count': 'avg', 'Passenger Count': 'stddev'}).toPandas()

# Número de passageiros por terminal
group_by_terminal = df.groupBy('Terminal').agg({'Passenger Count': 'avg', 'Passenger Count': 'stddev'}).toPandas()

# Exibir as tabelas
print("Passenger Count per Year")
display(group_by_year)

print("\nPassenger Count per Month")
display(group_by_month)

print("\nPassenger Count per Airline")
display(group_by_airline)

print("\nPassenger Count per Activity Type")
display(group_by_activity)

print("\nPassenger Count per Terminal")
display(group_by_terminal)
```

[5]:
```
import pandas as pd
```

[14]:
```
# Para usar Pandas e gerar gráficos, convertemos o DataFrame PySpark para Pandas
df_pandas = df.toPandas()

# Importar matplotlib
import matplotlib.pyplot as plt

# Função para gerar gráficos de barras e também retornar os dados numericamente
def plot_bar_chart_with_data(dataframe, x_col, y_col, title, xlabel, ylabel):
    plt.figure(figsize=(10, 6))
    plt.bar(dataframe[x_col], dataframe[y_col], color='skyblue')
    plt.title(title)
    plt.xlabel(xlabel)
    plt.ylabel(ylabel)
    plt.xticks(rotation=45)
    plt.tight_layout()
    plt.show()
    return dataframe[[x_col, y_col]]

# Converter as colunas numéricas corretamente (garantir que a conversão de texto para número está correta)
df_pandas['Passenger Count'] = pd.to_numeric(df_pandas['Passenger Count'], errors='coerce')
df_pandas['Year'] = pd.to_numeric(df_pandas['Year'], errors='coerce')

# 1. Número de passageiros por ano
group_by_year = df_pandas.groupby('Year')['Passenger Count'].agg(['mean']).reset_index()
numerical_data_year = plot_bar_chart_with_data(group_by_year, 'Year', 'mean', 'Média de Passageiros por Ano', 'Ano', 'Média de Passageiros')

# 2. Número de passageiros por mês
group_by_month = df_pandas.groupby('Month')['Passenger Count'].agg(['mean']).reset_index()
numerical_data_month = plot_bar_chart_with_data(group_by_month, 'Month', 'mean', 'Média de Passageiros por Mês', 'Mês', 'Média de Passageiros'

# 3. Número de passageiros por companhia aérea (Top 10)
group_by_airline = df_pandas.groupby('Operating Airline')['Passenger Count'].agg(['mean']).reset_index()
top_10_airlines = group_by_airline.sort_values(by='mean', ascending=False).head(10)
numerical_data_airlines = plot_bar_chart_with_data(top_10_airlines, 'Operating Airline', 'mean', 'Top 10 Companhias Aéreas por Média de Passag

# 4. Número de passageiros por tipo de atividade
group_by_activity = df_pandas.groupby('Activity Type Code')['Passenger Count'].agg(['mean']).reset_index()
numerical_data_activity = plot_bar_chart_with_data(group_by_activity, 'Activity Type Code', 'mean', 'Média de Passageiros por Tipo de Atividad

# 5. Número de passageiros por terminal
group_by_terminal = df_pandas.groupby('Terminal')['Passenger Count'].agg(['mean']).reset_index()
numerical_data_terminal = plot_bar_chart_with_data(group_by_terminal, 'Terminal', 'mean', 'Média de Passageiros por Terminal', 'Terminal', 'Mé
```

[24]:
```python
from pyspark.ml.feature import StringIndexer

# Verificar o esquema do DataFrame
df.printSchema()

# Codificar colunas categóricas que podem ser relevantes para a correlação
categorical_cols = ['Operating Airline', 'Published Airline', 'GEO Summary', 'GEO Region',
                    'Activity Type Code', 'Price Category Code', 'Terminal', 'Boarding Area',
                    'Adjusted Activity Type Code', 'Month']

# Para cada coluna categórica, aplicar o StringIndexer
for col in categorical_cols:
    indexer = StringIndexer(inputCol=col, outputCol=col + "_Indexed")
    df = indexer.fit(df).transform(df)

# Exibir algumas linhas para verificar as mudanças
df.show(5)
```

[25]:
```python
# Após codificar, posso calcular a matriz de correlação apenas para colunas numéricas
# Selecionar colunas numéricas para a correlação
numeric_cols = ['Passenger Count', 'Adjusted Passenger Count', 'Year'] + [col + "_Indexed" for col in categorical_cols]

# Criar uma nova tabela apenas com as colunas numéricas
df_numeric = df.select(numeric_cols)

# Converter para Pandas para calcular a correlação (ou usar um método alternativo no PySpark)
df_numeric_pandas = df_numeric.toPandas()

# Calcular a matriz de correlação
correlation_matrix = df_numeric_pandas.corr()

# Exibir a matriz de correlação
print(correlation_matrix)
```