PROJECT: PUBLIC TRANSPORTATION EFFICIENCY ANALYSIS

Public Transportation Analysis is the process of collecting, analysing and interpreting data on public transportation systems in order to improve their Efficiency, Effectiveness and Accessibility.

PROJECT'S OBJECTIVE:

The objective of a project focused on public transportation analysis can vary depending on the specific goals and scope of the project.

1. **Improving Service Efficiency**: Analyzing public transportation to identify areas where service can be made more efficient. This might include optimizing routes, schedules, or modes of transportation.

2. **Enhancing User Experience**: Understanding the needs and preferences of passengers to improve their experience. This could involve making stations more accessible, enhancing safety measures, or providing real-time updates to passengers.

3. **Reducing Environmental Impact**: Analyzing the environmental impact of public transportation systems and seeking ways to reduce it. This could involve transitioning to cleaner energy sources or promoting the use of public transportation over private vehicles.

4. **Cost Reduction**: Finding ways to reduce the operating costs of public transportation systems without compromising service quality.

5. **Increasing Ridership**: Developing strategies to attract more passengers to public transportation services, which can reduce traffic congestion and environmental impact.

6. **Evaluating Infrastructure**: Assessing the condition of transportation infrastructure such as bridges, tunnels, and rail lines to ensure safety and reliability.

7. **Sustainability and Accessibility**: Evaluating the sustainability and accessibility of public transportation for all segments of the population, including those with disabilities or low income.

8.. **Demand Forecasting**: Using data and analytics to forecast future demand for public transportation services and plan accordingly.

DEVELOPMENTAL PHASE: PART-1 :

Step 1: Define Objectives and Scope

- Clearly define the objectives of your public transportation analysis, such as improving service efficiency, reducing congestion, or increasing accessibility.

- Define the scope of the project, including the geographic area, the modes of transportation to be considered (e.g., buses, subways, trams), and the time frame.

Step 2: Data Collection and Preprocessing

- Gather relevant data, including demographic information, traffic data, existing transportation infrastructure, and passenger counts.

- Clean and preprocess the data to ensure it's accurate and compatible for analysis.

Step 3: Performance Measurement and Evaluation

- After implementation, continuously monitor the performance of the transportation system.
- Use key performance indicators (KPIs) to evaluate the effectiveness of the project.

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib
import matplotlib.pyplot as plt
%matplotlib inline
sns.set_style('darkgrid')
matplotlib.rcParams['font.size']=14
matplotlib.rcParams['figure.figsize']=(9,5)
matplotlib.rcParams['figure.facecolor']='#00000000'
import numpy as np
import pandas as pd
import seaborn as sns
import plotly.graph_objects as go
from plotly.offline import download_plotlyjs,init_notebook_mode,plot,iplot
from plotly.colors import n_colors
from wordcloud import WordCloud,ImageColorGenerator
init_notebook_mode(connected=True)
from plotly.subplots import make_subplots
```

1.)Calculate the maximum:

```
[ ] df.max()
```

```
<ipython-input-5-4c1ddf8920ff>:1: FutureWarning:

The default value of numeric_only in DataFrame.max is deprecated. In a future version, it will default to False.

TripID                                    62585
StopID                                    18493
StopName             Zone D Port Adelaide Interchan
WeekBeginning                    30/06/2013 00:00
NumberOfBoardings                            193
dtype: object
```

2.)Calculate the minimum:

```
[ ] df.min()
```

```
<ipython-input-6-c3612c624a3f>:1: FutureWarning:

The default value of numeric_only in DataFrame.min is deprecated. In a future version, it will default to False.

TripID                           3017
StopID                          10817
StopName                1 Anzac Hwy
WeekBeginning        01/06/2014 00:00
NumberOfBoardings                   1
dtype: object
```

3.)Accessing column from a given dataset:

```
df.StopName
```

```
0                          181 Cross Rd
1                          177 Cross Rd
2                          175 Cross Rd
3          Zone A Arndale Interchange
4                          178 Cross Rd
                          ...
1048570                    8 Fullarton Rd
1048571                3 Glen Osmond Rd
1048572                    9 Fullarton Rd
1048573                    5 Fullarton Rd
1048574                    6 Fullarton Rd
Name: StopName, Length: 1048575, dtype: object
```

4.) The nunique() method returns the number of unique values for each column. By specifying the column axis (axis='columns'), the nunique() method searches column-wise and returns the number of unique values for each row.

```
[8]  df.StopName.nunique()
```

```
583
```

5.) EXPLORING A COLUMN by computing the number of Stop names:

```
df.StopName.value_counts()
```

```
I1 North Tce                    12678
23  Findon Rd                   10558
21 Port Rd                       9835
R1 North Tce                     9221
B1 East Tce                      8557
                                ...
V2 King William St                  2
I2 North Tce                        1
Aust. Submarine Corp Gate 640       1
11 East Av                          1
L1 Unley Rd                         1
Name: StopName, Length: 583, dtype: int64
```

6.)**ANALYSE THE MINIMUM AND MAXIMUM OF** Stop Names and TRIP ID's mentioned in the dataset("Calculates through the lexicographic order")

```
[10] df.TripID.min()
3017

df.TripID.max()
62585

[13] df.StopName.min()
'1 Anzac Hwy'

[14] df.StopName.max()
'Zone D Port Adelaide Interchan'
```

CREATIING VISUALIZATIONS OF DATA USING THE GIVEN DATASET:

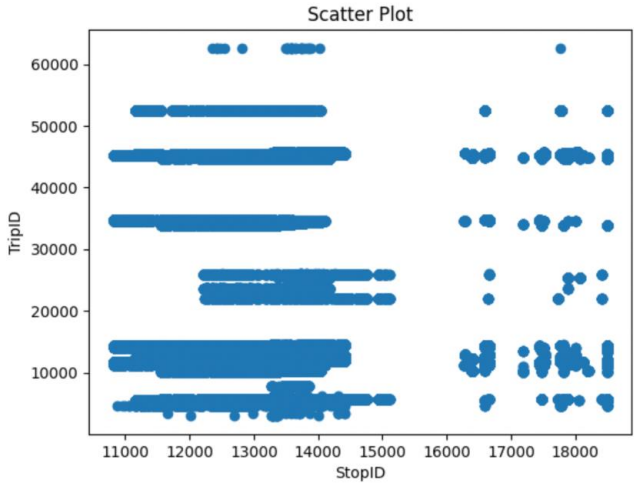
Creating a scatter plot is a simple way to visualize data, and it can be explained in just two steps:

- Prepare Your Data: First, you need to have a set of data that includes two variables, typically referred to as X and Y.
- Each data point should have a pair of values (x, y). Make sure your data is organized and ready for plotting.
- Create the Scatter Plot: Use a data visualization tool like Excel, Python's Matplotlib, or any other plotting software.
- Label your axes for clarity.

7.)SCATTER PLOT TO VISUALIZE STOP ID AND TRIP ID

```
data = pd.read_csv("/content/drive/MyDrive/public_trans.csv")

# Scatter plot with day against tip
plt.scatter(data['StopID'], data['TripID'])
# Adding Title to the Plot
plt.title("Scatter Plot")
# Setting the X and Y labels
plt.xlabel('StopID')
plt.ylabel('TripID')
plt.show()
```

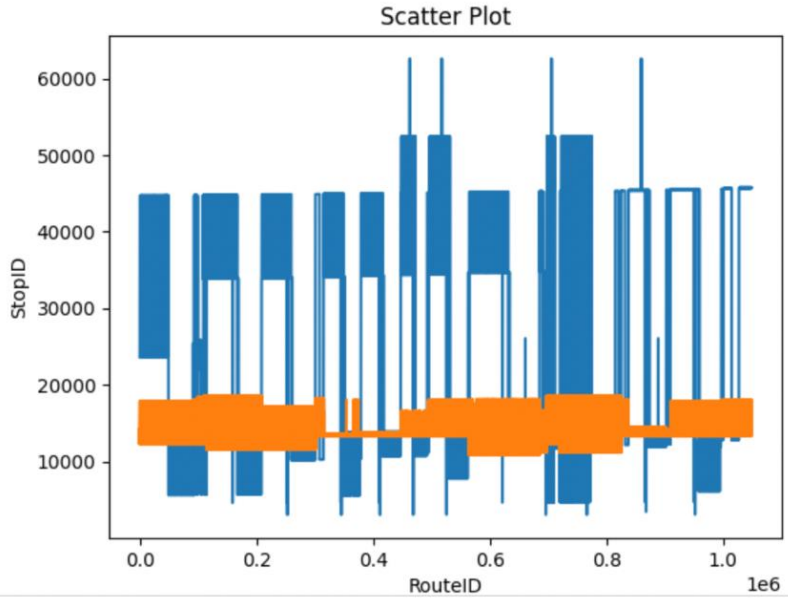
```
<ipython-input-6-571bf029a562>:1: DtypeWarning: Columns (1) have mixed types. Specify dtype option on import or set low_memory=False.
  data = pd.read_csv("/content/drive/MyDrive/public_trans.csv")
```



8.)SCATTER PLOT TO VISUALIZE ROUTE ID AND STOP ID:

```
data = pd.read_csv("/content/drive/MyDrive/public_trans.csv")

plt.plot(data['TripID'])
plt.plot(data['StopID'])

# Adding Title to the Plot
plt.title("Scatter Plot")

# Setting the X and Y labels
plt.xlabel('RouteID')
plt.ylabel('StopID')

plt.show()
```

```
<ipython-input-15-2b18550ea2f1>:1: DtypeWarning: Columns (1) have mixed types. S
  data = pd.read_csv("/content/drive/MyDrive/public_trans.csv")
```



9. HISTOGRAM:

Creating a histogram can be done in two steps:

Prepare Data: Gather the dataset that should be visualized.

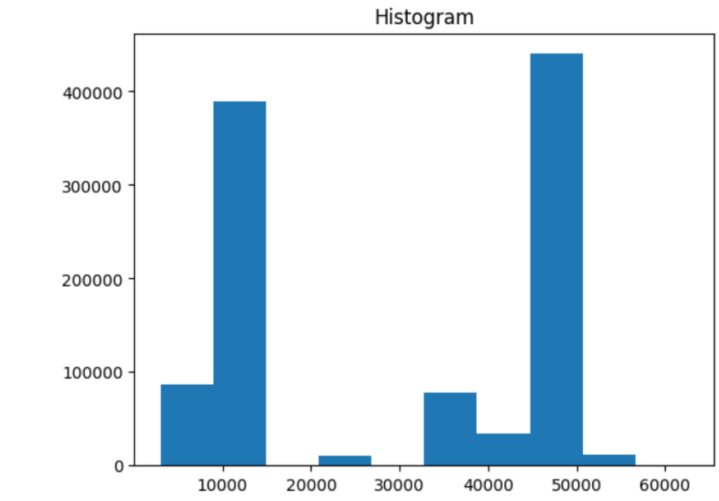
- This data should consist of a single variable for which you want to create a histogram.
- Ensure your data is organized and ready for plotting.

Create the Histogram: Utilize data visualization tools like Excel, Python's Matplotlib, or other software.

- Input your data and instruct the software to generate a histogram.
- Specify the variable you want to plot, the number of bins (intervals) to divide the data.
- Label the axes for clarity.

```
data = pd.read_csv("/content/drive/MyDrive/public_trans.csv")

# histogram of total_bills
plt.hist(data['TripID'])

plt.title("Histogram")

# Adding the legends
plt.show()
```

```
<ipython-input-5-56be36da2417>:1: DtypeWarning: Columns (1) have mixed types.
  data = pd.read_csv("/content/drive/MyDrive/public_trans.csv")
```



10.) BAR CHART TO VISUALIZE TRIP ID AND STOP ID:

Creating a bar chart can be explained in two steps:

1. Prepare Your Data:

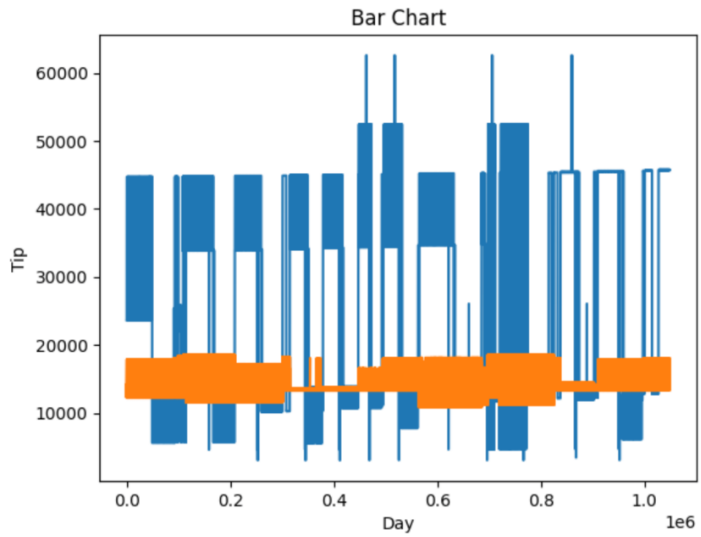
- Collect the dataset you want to represent in a bar chart.
- This data should typically consist of categories or groups and their corresponding values.
- Ensure your data is organized and ready for visualization.

2. Create the Bar Chart:

- Use data visualization software like Excel, Python's Matplotlib, or other charting tools.
- Input your data and instruct the software to generate a bar chart.
- Assign the categories to the horizontal axis (X-axis) and the values to the vertical axis (Y-axis).
- Label the axes and format the chart.

```
data = pd.read_csv("/content/drive/MyDrive/public_trans.csv")

plt.plot(data['TripID'])
plt.plot(data['StopID'])

plt.title("Bar Chart")

# Setting the X and Y labels
plt.xlabel('Day')
plt.ylabel('Tip')

# Adding the legends
plt.show()
```

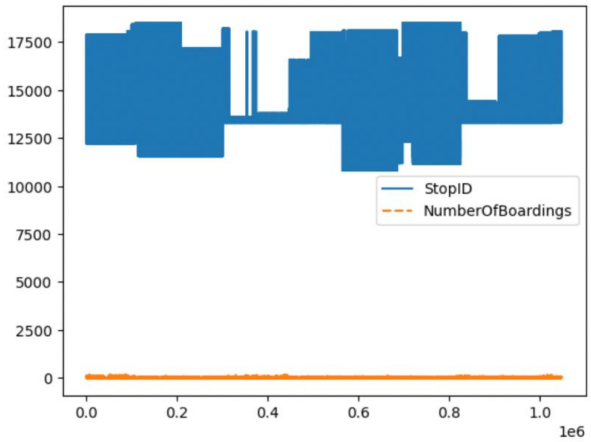
```
<ipython-input-10-e8cc8560101b>:1: DtypeWarning: Columns (1) have mixed types.
  data = pd.read_csv("/content/drive/MyDrive/public_trans.csv")
```



11.) Line plot by using seaborn by importing it and # using only data attribute

```
data = pd.read_csv("/content/drive/MyDrive/public_trans.csv")

# using only data attribute
sns.lineplot(data=data.drop(['TripID'], axis=1))
plt.show()
```

```
<ipython-input-17-8a37436779bb>:1: DtypeWarning: Columns (1) have mixed types.
  data = pd.read_csv("/content/drive/MyDrive/public_trans.csv")
```



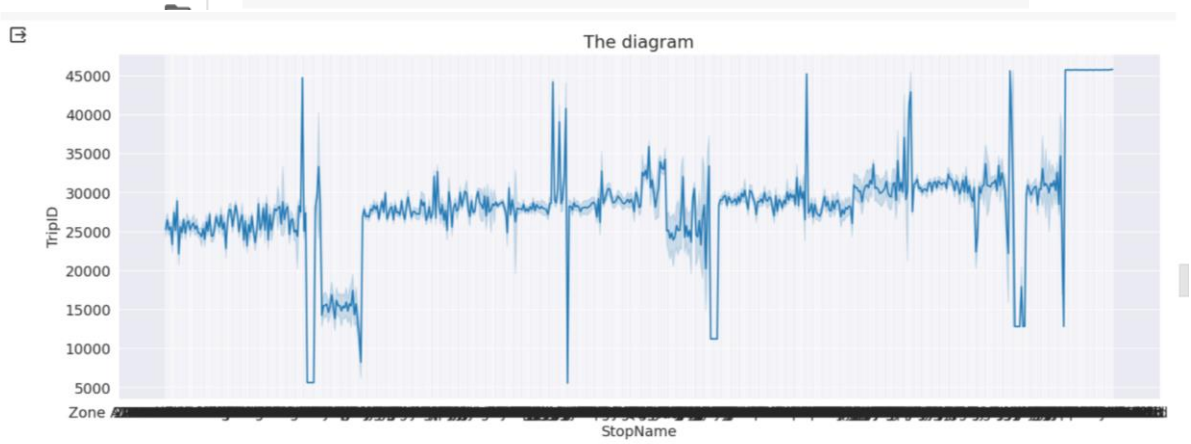
12.) CREATION AND VISUALIZATION OF ECG-LIKE PLOTS USING SEABORN LIBRARIES:

Creating line plots similar to ECG (electrocardiogram) graphs :

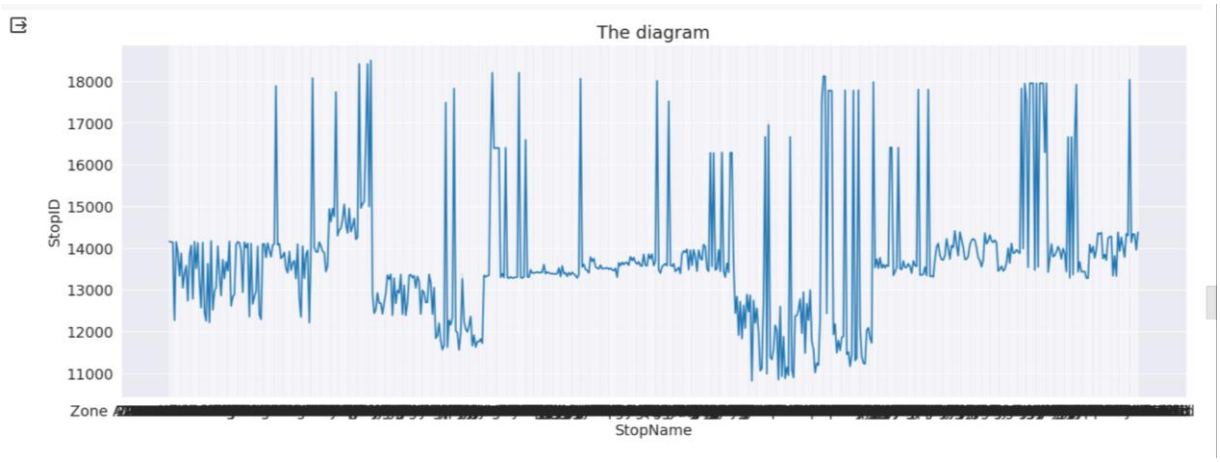
- **Data Collection:** Collect the data you want to represent in your ECG-like line plot.
- **Plotting the Line Graph:** Use software or libraries suited for time-series data visualization, such as Python's Matplotlib, R, or specialized medical visualization tools. Input your time-series data and create a line plot. Ensure the x-axis represents time, and the y-axis represents voltage or amplitude.

- Creating ECG-like line plots can be more complex than standard line plots due to the specific requirements for representing cardiac electrical activity accurately.
- Therefore, it's often done using dedicated ECG analysis software to ensure the necessary level of detail and precision.

```
plt.figure(figsize=(18,6))
sns.lineplot(x=df.StopName, y=df.TripID)
plt.title("The diagram")
plt.show()
```




```
plt.figure(figsize=(18,6))
sns.lineplot(x=df.StopName, y=df.StopID)
plt.title("The diagram")
plt.show()
```



13.)ACCESSING THE USED STOP NAMES:

```
trans = df["StopName"].unique()
for i in trans:
  c=list(df[df["StopName"]==1]['StopID'])
  print(f"StopName: {i}nUsed countries:{c}")
print('-'*70)
```

```
trans = df["StopName"].unique()
for i in trans:
  c=list(df[df["StopName"]==1]['StopID'])
  print(f"StopName: {i}nUsed countries:{c}")
print('-'*70)

StopName: 181 Cross RdnUsed countries:[]
StopName: 177 Cross RdnUsed countries:[]
StopName: 175 Cross RdnUsed countries:[]
StopName: Zone A Arndale InterchangenUsed countries:[]
StopName: 178 Cross RdnUsed countries:[]
StopName: 9A  Marion RdnUsed countries:[]
StopName: 9A  Holbrooks RdnUsed countries:[]
StopName: 9  Marion RdnUsed countries:[]
StopName: 206 Holbrooks RdnUsed countries:[]
StopName: 8A  Marion RdnUsed countries:[]
StopName: 8D  Marion RdnUsed countries:[]
StopName: 23  Findon RdnUsed countries:[]
StopName: 8K  Marion RdnUsed countries:[]
StopName: 20  Cross RdnUsed countries:[]
StopName: 22A  Crittenden RdnUsed countries:[]
StopName: 180 Cross RdnUsed countries:[]
StopName: 8C  Marion RdnUsed countries:[]
StopName: 173 Cross RdnUsed countries:[]
StopName: 13  Holbrooks RdnUsed countries:[]
StopName: 218 Findon RdnUsed countries:[]
StopName: 11A  Marion RdnUsed countries:[]
StopName: 220 Woodville RdnUsed countries:[]
StopName: 25 Torrens RdnUsed countries:[]
StopName: 8E  Marion RdnUsed countries:[]
StopName: 224 Woodville RdnUsed countries:[]
StopName: 183 Cross RdnUsed countries:[]
StopName: 219 Woodville RdnUsed countries:[]
StopName: 17  Grange RdnUsed countries:[]
StopName: 205 Holbrooks RdnUsed countries:[]
StopName: 10A  Marion RdnUsed countries:[]
StopName: 201 Marion RdnUsed countries:[]
StopName: 20  Crittenden RdnUsed countries:[]
StopName: 8G  Marion RdnUsed countries:[]
StopName: 10  Holbrooks RdnUsed countries:[]
StopName: 8F  Marion RdnUsed countries:[]
StopName: 8B  Marion RdnUsed countries:[]
```

14.) USING GROUPBY METHOD:

A **groupby operation** involves some combination of splitting the object, applying a function, and combining the results. This can be used to group large amounts of data and compute operations on these groups.

15.) PATCHES TO ANALYSE THE STOPNAMES AND STOP ID:

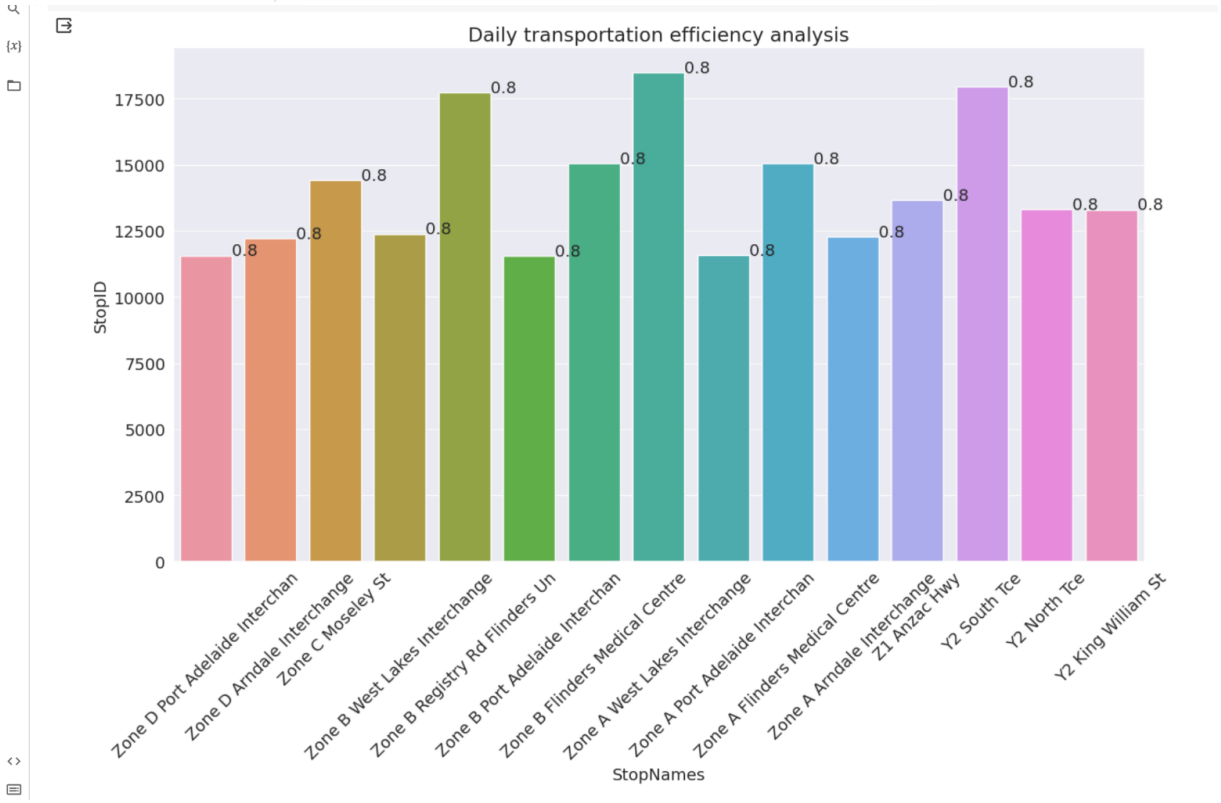
1. Import a Graphics Library:

- To create patches, you need to use a graphics library or software such as Matplotlib in Python.
- These libraries provide functions and classes for drawing and manipulating graphical objects, including patches.

2. Defining and Drawing the Patch:

- Use the library's functions or methods to define the characteristics of your patch, such as its shape (e.g., rectangle, circle, polygon), size, position, and style (e.g., color, fill, outline).
- Then, instruct the library to draw the patch on your canvas or graphical display.
- This typically involves specifying the coordinates and attributes of the patch within the graphical context.

```
plt.figure(figsize=(15,8))
ax=sns.barplot(x=StopName, y=StopName.index)
plt.xlabel("Daily transportation")
plt.xlabel("StopNames")
plt.title("Daily transportation efficiency analysis")
for patch in ax.patches:
  width=patch.get_width()
  height=patch.get_height()
  x=patch.get_x()
  y=patch.get_y()
  plt.xticks(rotation=45)
  plt.text(width+x, height+y, '{:.1f} '.format(width))
```

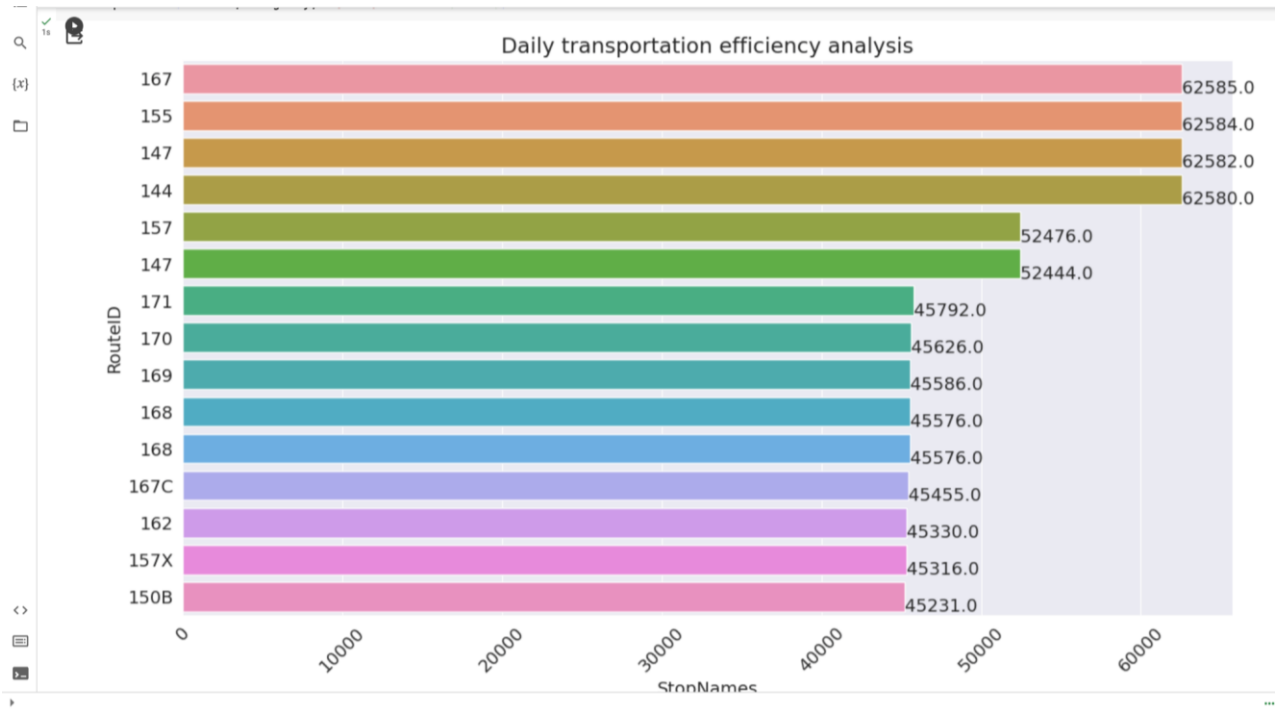


17.COMPUTING MEAN FROM THE GIVEN DATASET:

```
[56] df.mean()

<ipython-input-56-c61f0c8f89b5>:1: FutureWarning:

The default value of numeric_only in DataFrame.mean is deprecated.

TripID              28602.993331
StopID              13301.143187
NumberOfBoardings       4.132290
dtype: float64
```

18.BARPLOT TO VISUALIZE THE TRIP ID:

```
plt.figure(figsize=(15,8))
ax=sns.barplot(x=TripID, y=TripID.index)
plt.xlabel("Daily transportation")
plt.xlabel("StopNames")
plt.title("Daily transportation efficiency analysis")
for patch in ax.patches:
  width=patch.get_width()
  height=patch.get_height()
  x=patch.get_x()
  y=patch.get_y()
  plt.xticks(rotation=45)
  plt.text(width+x, height+y, '{:.1f} '.format(width))
```



DESIGN THINKING PROCESS:

1. **Empathize:** The project team takes measures to understand the needs and pain points of public transportation users. They learn that users are frustrated with long wait times, crowded vehicles, and unreliable service.

2. **Define:** The project team synthesizes the findings from the empathy phase and defines the key problem: the city's public transportation system is not meeting the needs of its users.

3.**Ideate**: The project team generates a variety of creative solutions to the problem, such as:

***** Increasing the frequency of bus and train service

* Implementing real-time arrival information.

* Offering discounted fares to students and seniors

* Providing more comfortable and accessible vehicles

4.**Prototype**: The project team selects the most promising solutions and develops prototypes. For example, they pilot a new bus route with more frequent service and real-time arrival information.

5. **Test**: The project team implements the prototypes and collects data on user satisfaction, ridership, and other metrics. The results show that the new bus route is a success, with more riders and higher satisfaction rates. By using a design thinking approach, the project team will be able to develop and implement solutions that meet the needs of public transportation users and improve the overall system.

ANALYSIS OBJECTIVES:

Descriptive Analysis:

- Begin with a descriptive analysis to provide an overview of the transportation system's current state.
- Create summary statistics and visualizations to illustrate key metrics, such as ridership, service coverage, and on-time performance.

- **Identify Trends and Patterns:**
  - Use data visualization techniques to identify trends and patterns in the data. For example, look for seasonal variations, peak travel times, or areas with high passenger volumes.
- **Segmentation:**
  - Segment the data to analyze the behavior of different user groups, such as commuters, students, tourists, or residents of various neighborhoods.

DATA COLLECTION PROCESS:

- **Data Collection and Cleaning:**
  - Start by reviewing the data collection process.
  - Ensure that data is clean, complete, and accurate.
  - **Data Collection Methods:** Choose the appropriate data collection methods. Methods can include surveys, interviews, observations, sensors, automated data feeds, and web scraping. The method selected should align with the data source and research objectives.
- **Data Collection Tools:**
  - Select the tools and equipment needed to collect the data. This may involve hardware like sensors or software for surveys and data entry.
- **Data Collection Schedule:**
  - Create a data collection schedule that outlines when and how data will be collected. Ensure that the schedule aligns with the project's timeline and objectives.
- **Data Collection Procedures:**
  - Develop standardized procedures for data collection to ensure consistency and reliability.

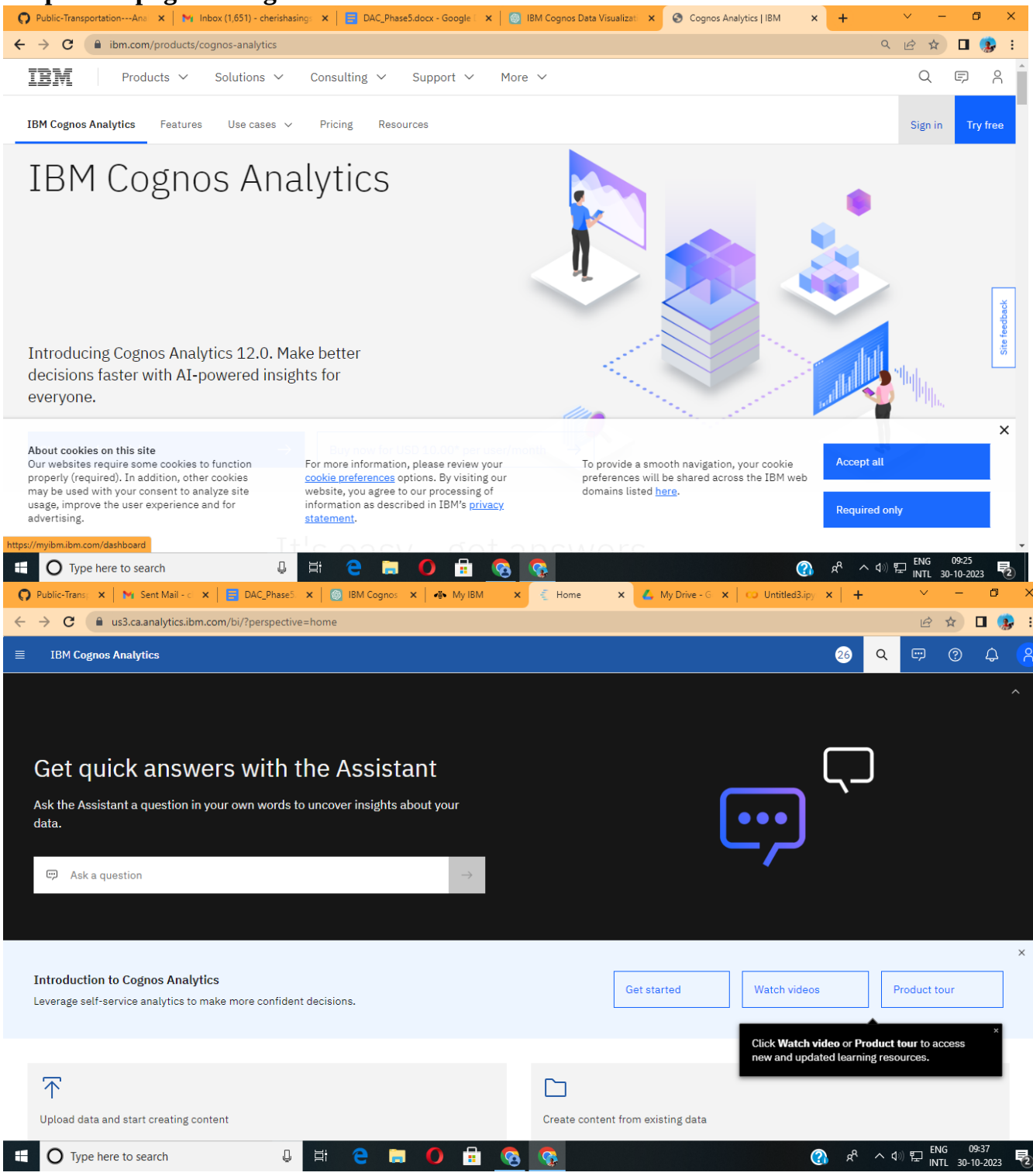
DATA VISUALIZATION:

- Data visualization is the graphical representation of data to help people understand complex information more easily and make data-driven decisions.
- It involves creating visual representations of data sets to reveal patterns, trends, and insights that might not be apparent in raw data.
- Effective data visualization can significantly enhance data comprehension and communication.

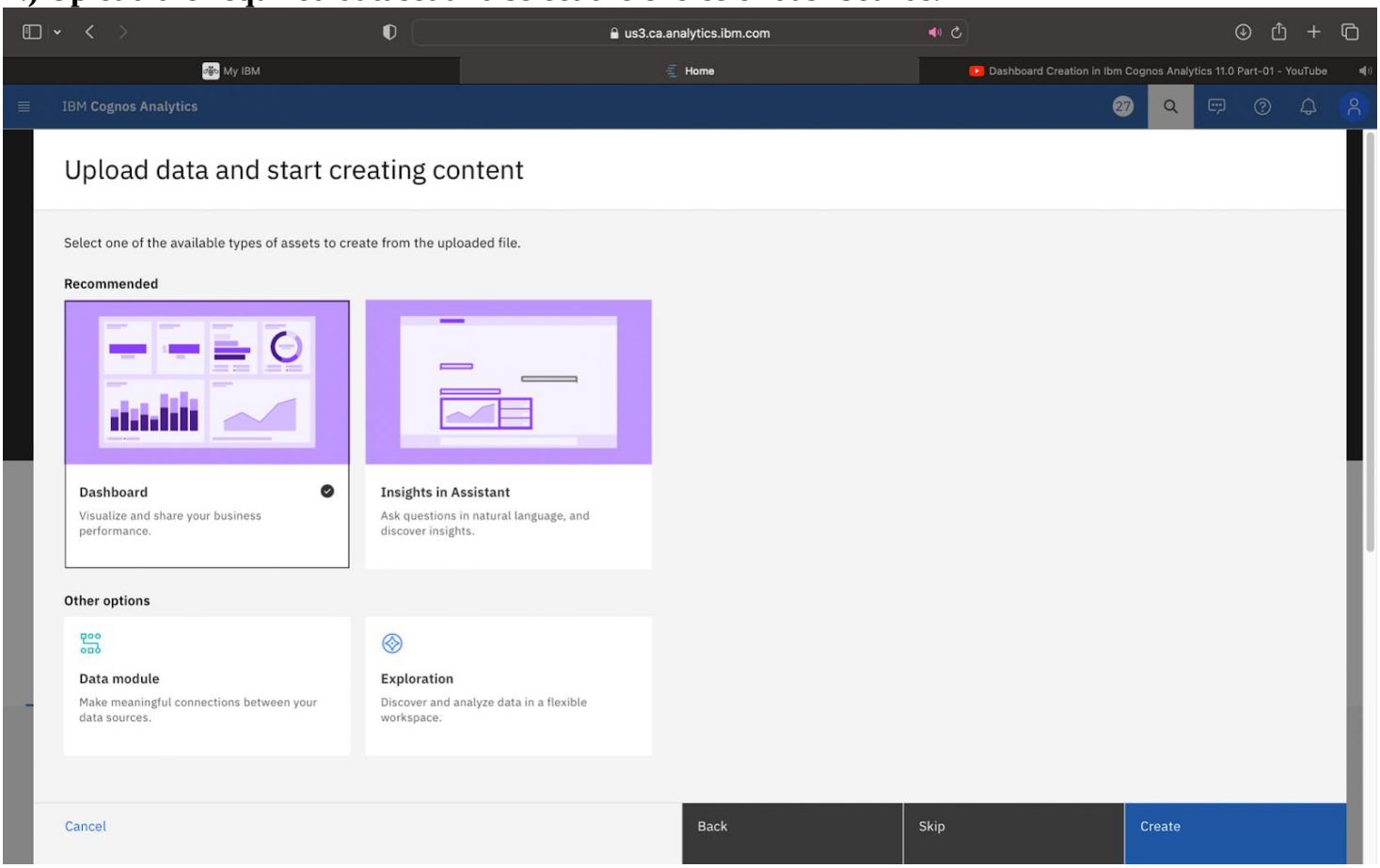
IBM Cognos analytics:

IBM Cognos Analytics is a business intelligence and data visualization tool that helps users create interactive reports and dashboards from their data.

1.Open the page and sign in to the account.



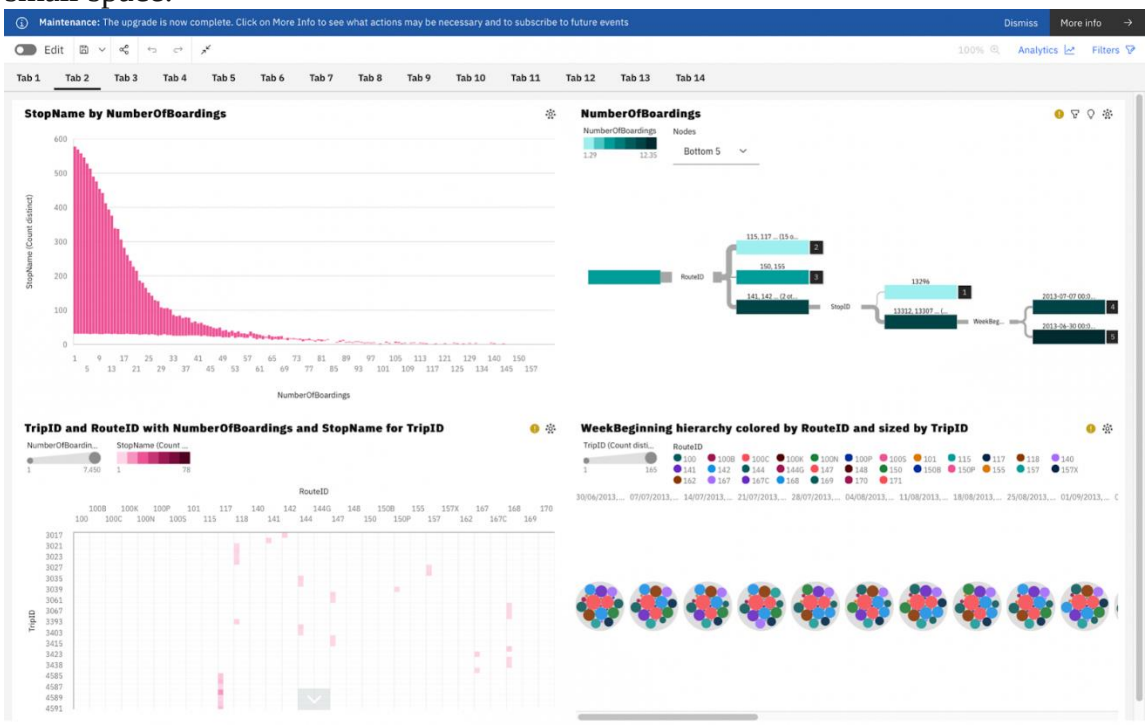
2.) Upload the required dataset and select the choice of dashboards.



3.)Create the visualizations by selecting tabs.

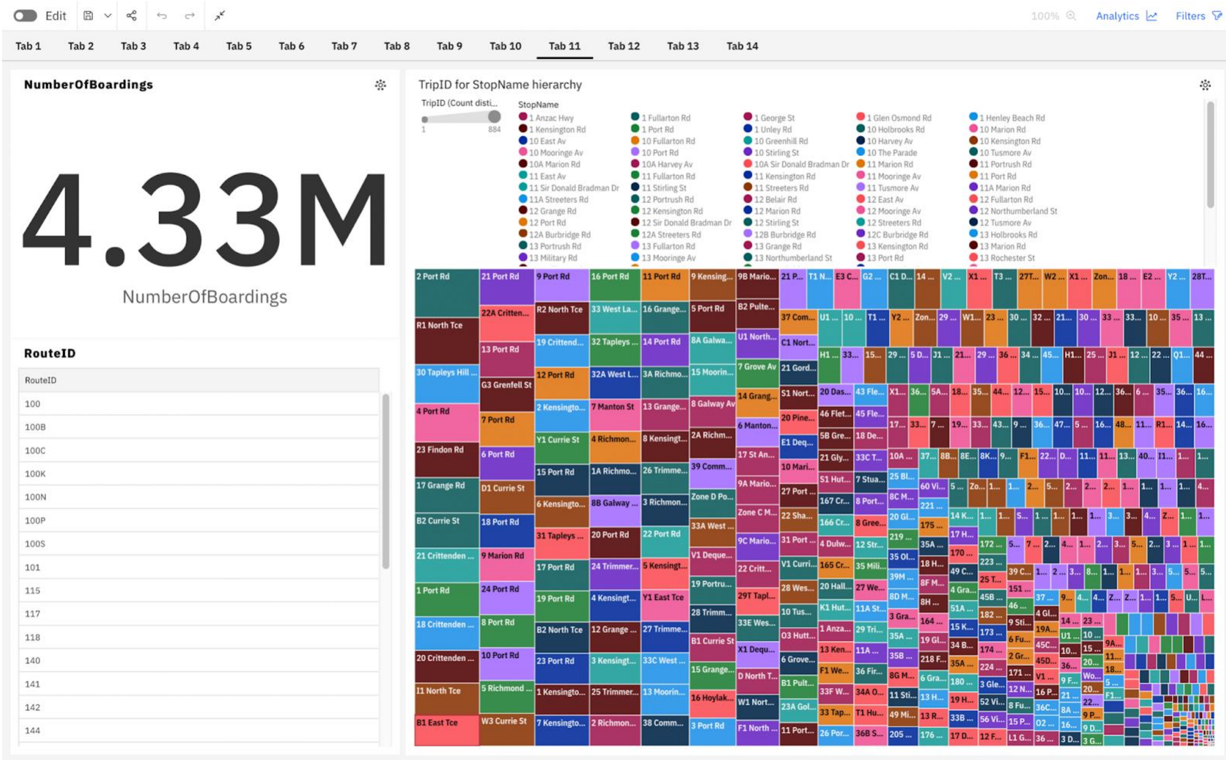
- **Area:**
  Use an area visualization to emphasize the magnitude of change over time.
- **Bubble:**
  Use a bubble visualization to show relationships among columns that contain numeric values, such as revenue and profit.
- **Bullet:**
  Use bullet charts to show measures that need to be compared against a target value.
- **Clustered column:**
  Use a clustered column visualization to compare values by one or more columns, such as sales for products per country.

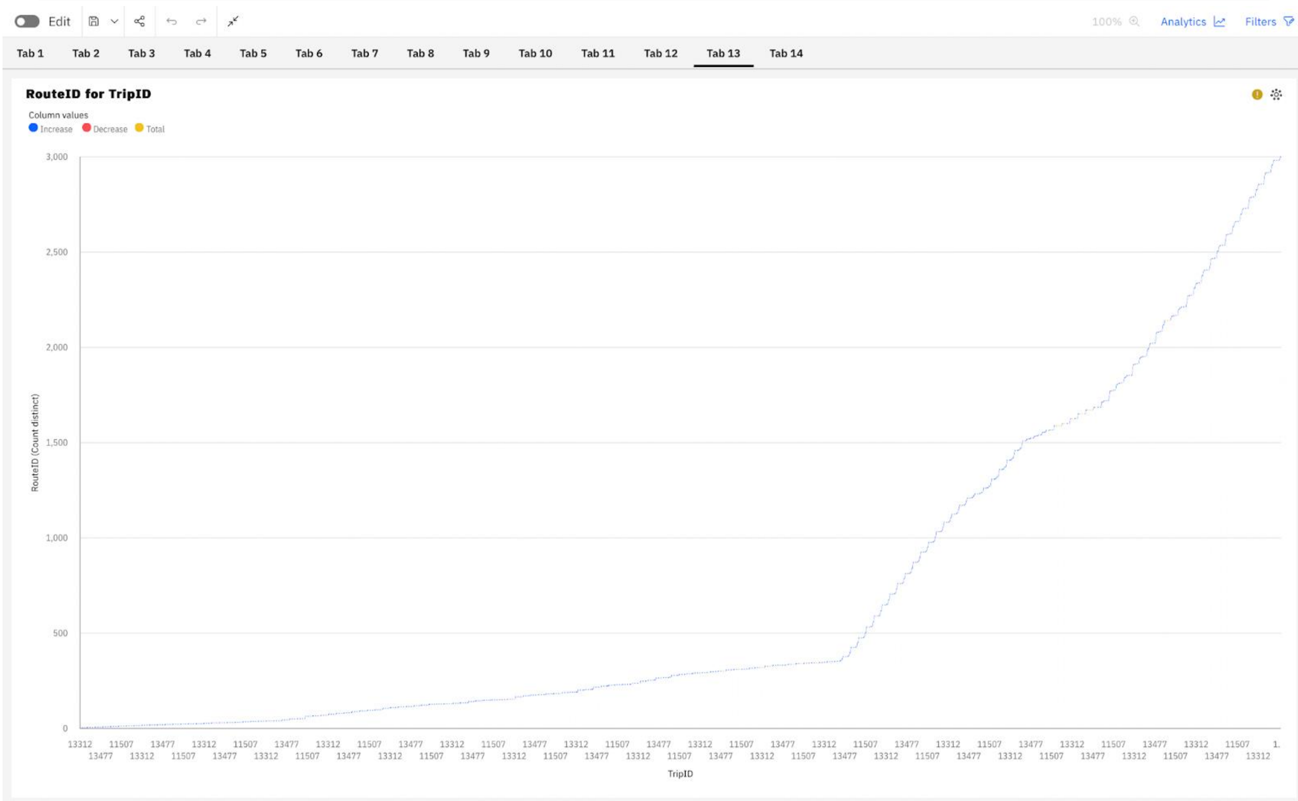
- **Decision tree:** A decision tree shows a connected hierarchy of boxes to represent the values of records.
- **Heat map**: Use a heat map visualization to visualize the relationship between columns, represented in a matrix type view.
- **Bubble Hierarchy**: A hierarchy bubble visualization shows a large amount of data in a small space.

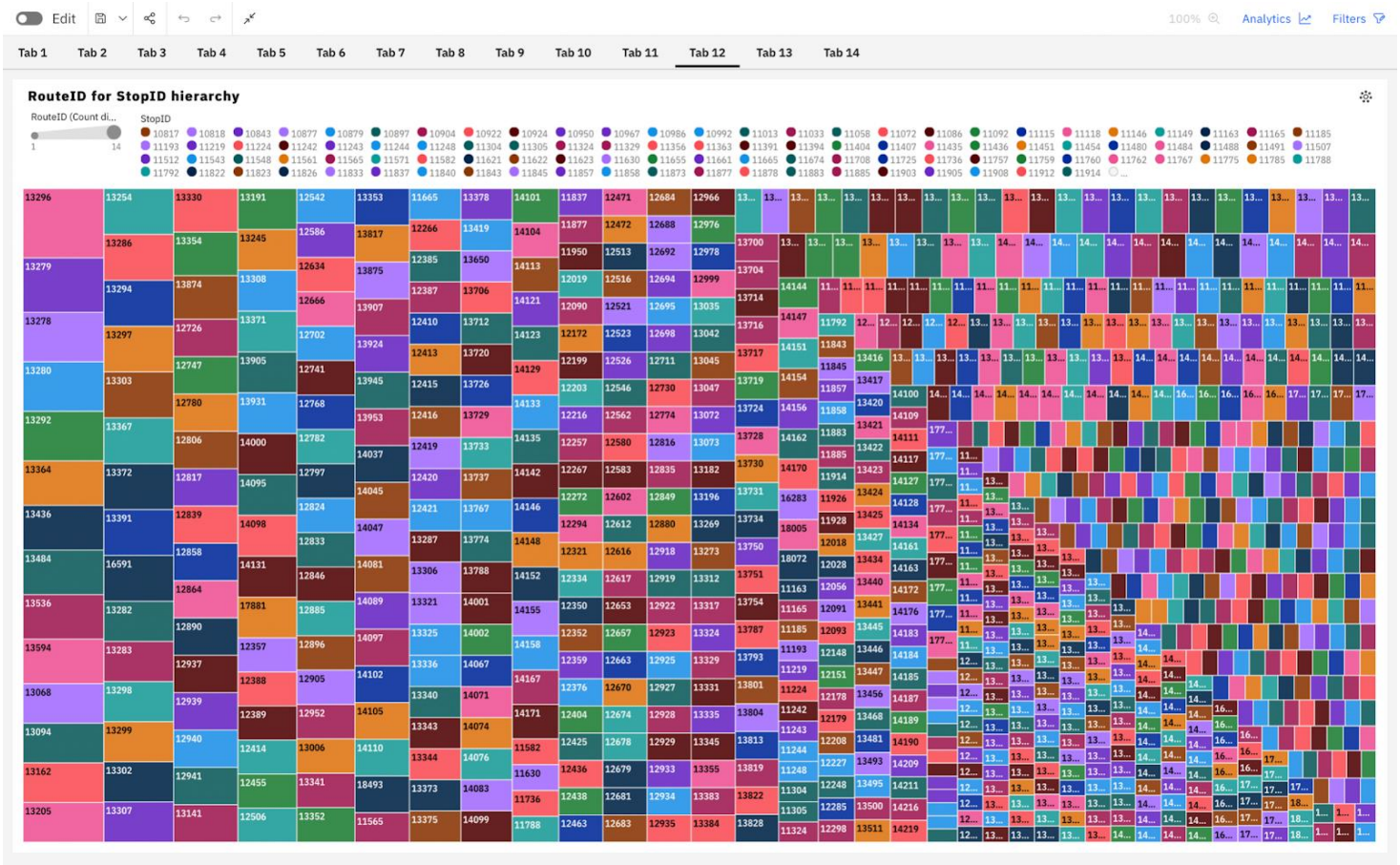


- **Line**:Use a line visualization to show trends over time.
- **Line and Column:**Use a line and column visualization to highlight relationships between multiple data series by combining bars and lines with one visualization.

- **Summary:**Data visualization and summary statistics are an important part of statistical analysis.
- It can help you identify trends in your data and communicate your research in presentations.
- **Table:**A table shows data in rows and columns.
- **Treemap:** Use a treemap visualization to identify patterns and exceptions in a large, complex data asset.
- **Waterfall:**Use a waterfall visualization to understand the cumulative effect a series of positive and negative values have on an initial value. The bars in a waterfall visualization are not totals.
- **Word cloud:**
  Use a word cloud visualization when you want to see a text-based visualization of a column. The text height represents the scale. The name itself is the different members of the column.

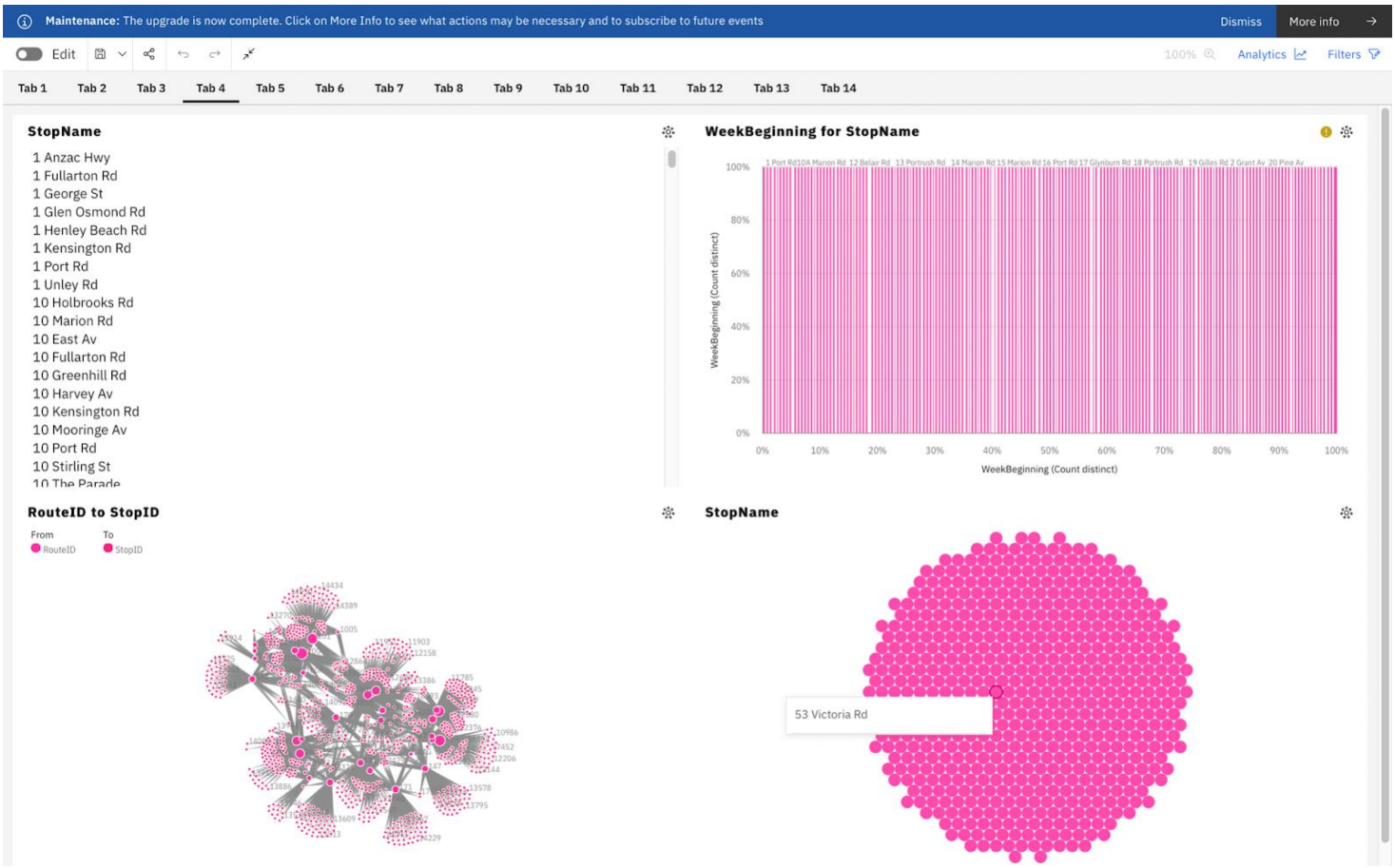

Edit
Analytics
Filters
Tab 1 Tab 2 Tab 3 Tab 4 Tab 5 Tab 6 Tab 7 Tab 8 Tab 9 Tab 10 Tab 11 Tab 12 Tab 13 Tab 14
NumberOfBoardings
4.33M
NumberOfBoardings
RouteID
RouteID
100
100B
100C
100K
100N
100P
100S
101
115
117
118
140
141
142
144
TripID for StopName hierarchy
TripID (Count disti...
StopName


Edit
Analytics
Filters
Tab 1 Tab 2 Tab 3 Tab 4 Tab 5 Tab 6 Tab 7 Tab 8 Tab 9 Tab 10 Tab 11 Tab 12 Tab 13 Tab 14
RouteID for TripID
Column values
Increase Decrease Total
RouteID (Count distinct)
TripID

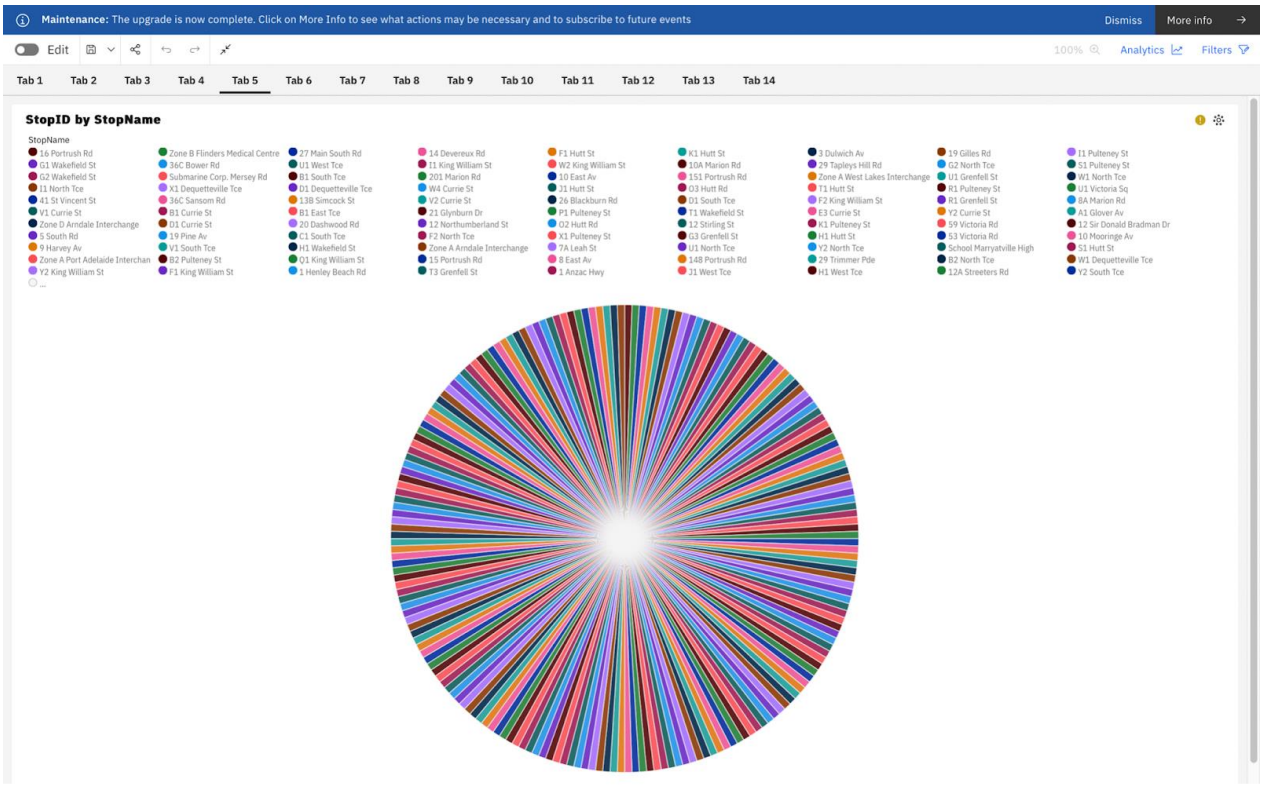


- **List:**
  Use a list visualization to create an overview of the data in a hierarchical way.

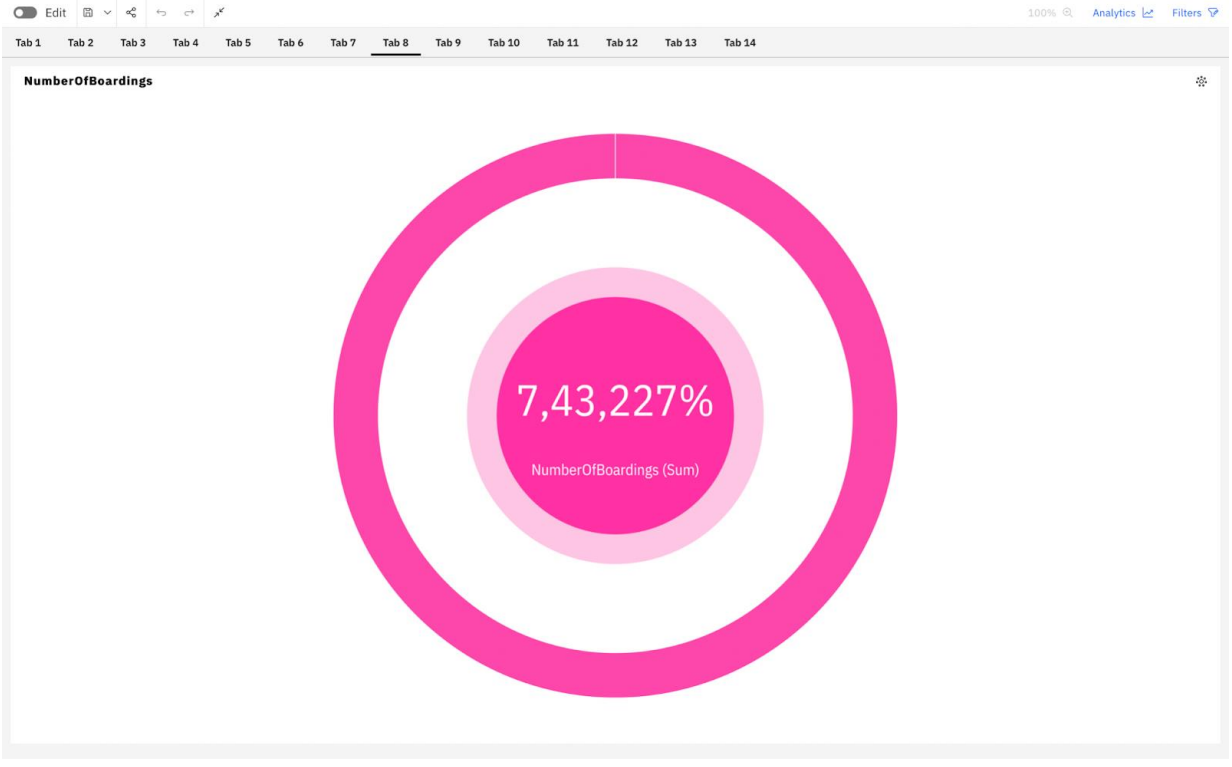
- **Marimekko**
  A marimekko visualization is similar to a stacked column visualization. It shows data through varying heights and includes an added dimension of data through varying column widths. The width of the columns is based on the value that is assigned to the width field. Individual segment height is a percentage of the respective column total value.
- **Network**
  Use a network visualization when you want to see the connections among columns in your data asset. A network visualization is a good choice to show connections, networks, and points of intersection.
- **Packed bubble**
  Use a packed bubble visualization when you want to show relationships among columns that contain numeric values, such as revenue. It is similar to the bubble visualization but the bubbles are tightly packed instead of spread over a grid. A packed bubble visualization shows a large amount of data in a small space.



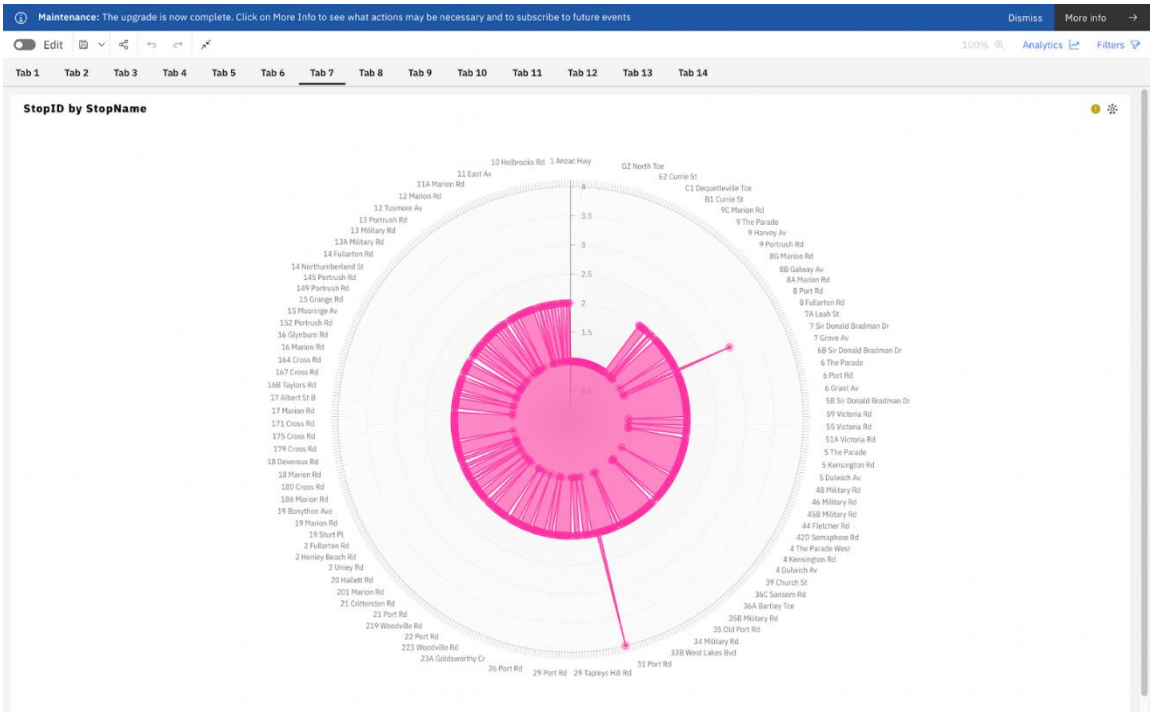
- **Pie**
  Use a pie visualization to highlight proportions. Each slice shows the relative relationship of each part to the whole.



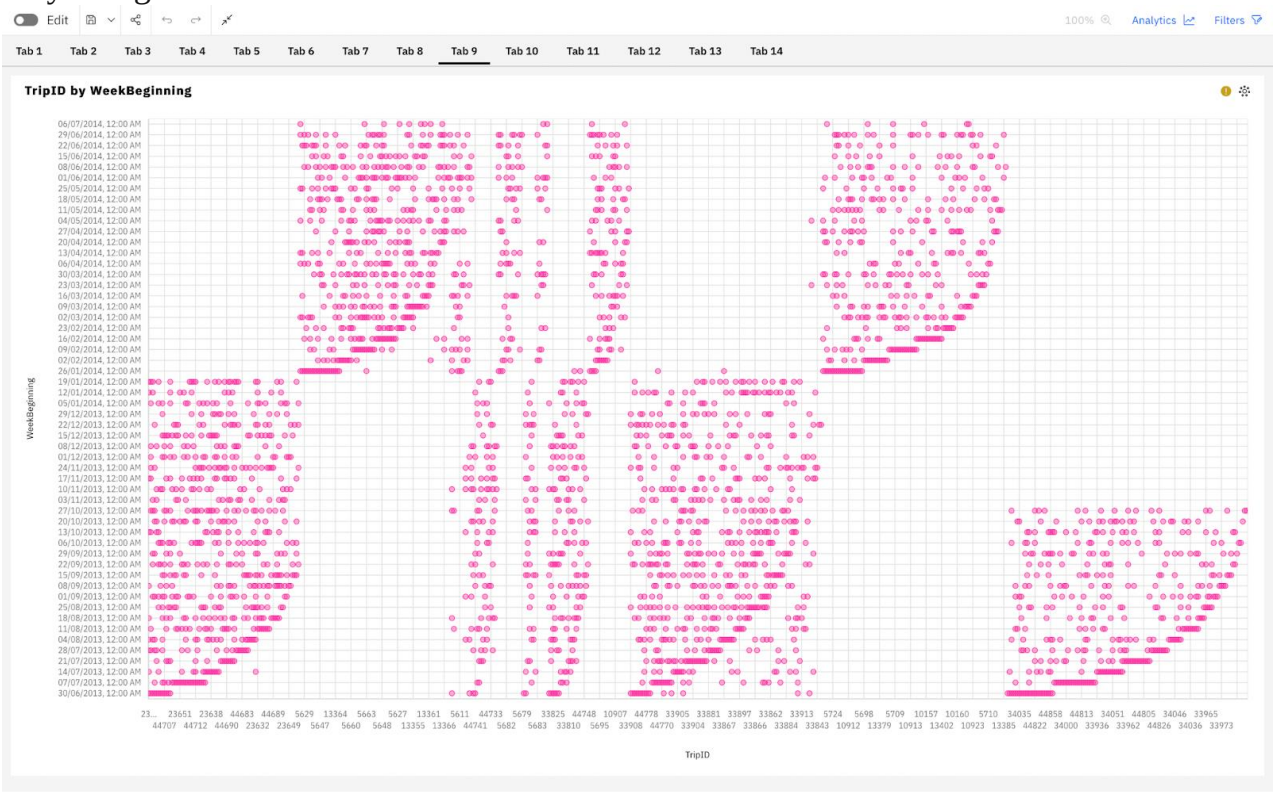
- **Radial:** In a radial visualization, each bar appears in a circle with longer bars that represent larger values. Hover over a bar to see the details about it, such as the exact value represented by the bar.



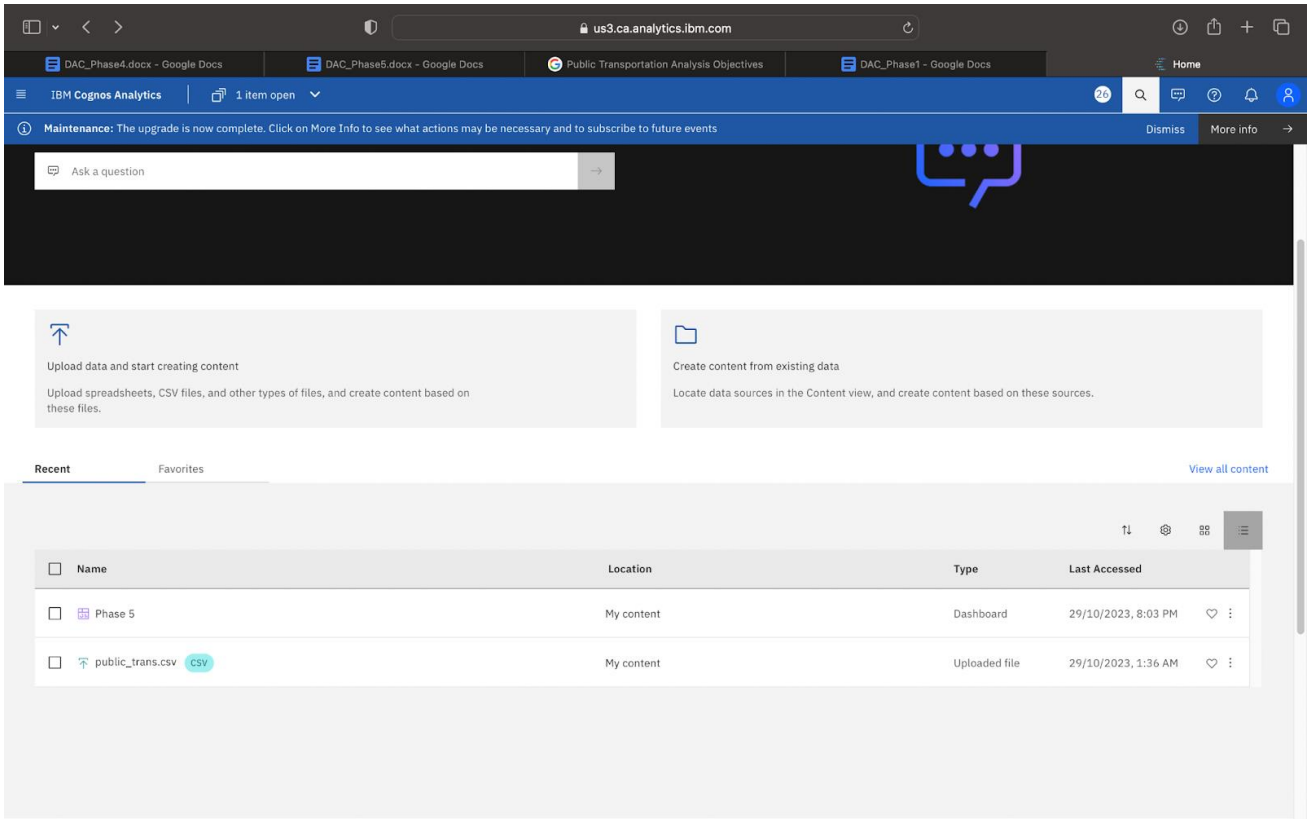
- [Radar](): Use a radar visualization for comparing multiple quantitative variables. The radar visualization below shows which variables have similar values or if there are any outliers amongst each variable.



- **Scatter**: Scatter visualizations use data points to plot two measures anywhere along a scale, not only at regular tick marks.



4.)Finally, Save the dashboard and explore the visualizations for better understanding of the analysis.



INSIGHTS:

- We have made progress in actively acquiring information about the analysis.
- A public transportation analysis project can yield valuable insights that benefit both transportation authorities and the communities they serve.
- **Ridership Patterns:** Understanding when and where people use public transportation services the most. This insight can help optimize schedules and routes to match demand.
- Service Efficiency: Identifying bottlenecks and inefficiencies in the transportation system, helping authorities allocate resources more effectively.
- **Technology Integration**: Assessing the effectiveness of new technologies and data analytics in enhancing transportation services.
- **Demand Forecasting**: Predicting future demand for transportation services, allowing for proactive planning and resource allocation.
- These insights can inform decision-making, policy development, and investments in public transportation infrastructure and services, ultimately leading to more efficient, sustainable, and passenger-friendly transportation systems.

CONCLUSION:

The success of a **public transportation analysis** project often depends on the quality of data collected, the tools and methodologies used, and the collaboration between transportation authorities, data analysts, and the communities served.