Practical 1

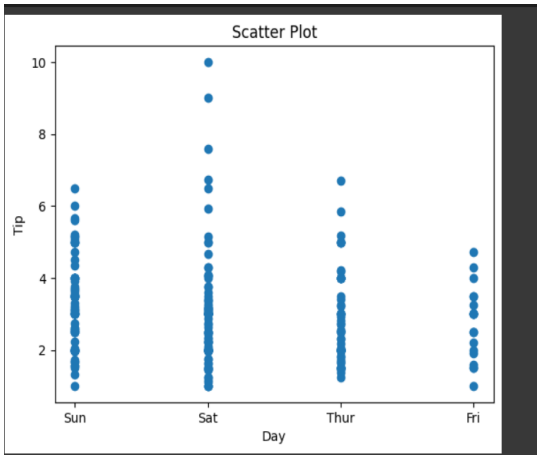
Aim: Show Basic Visualization in Python

1.Scatter Plot

Code:

```
import pandas as pd
import matplotlib.pyplot as plt


# reading the database
data = pd.read_csv("tips.csv")

# Scatter plot with day against tip
plt.scatter(data['day'], data['tip'])

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

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

plt.show()
```

Output:

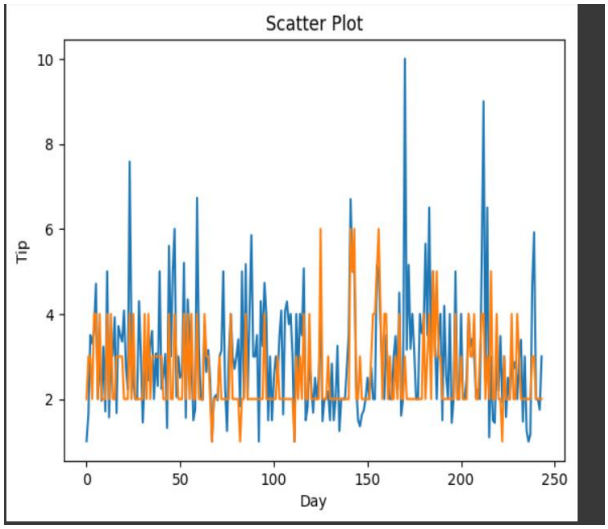


2.Line Plot

Code:

```
import pandas as pd
import matplotlib.pyplot as plt


# reading the database
data = pd.read_csv("tips.csv")

# Scatter plot with day against tip
plt.plot(data['tip'])
plt.plot(data['size'])

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

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

plt.show()
```

Output:

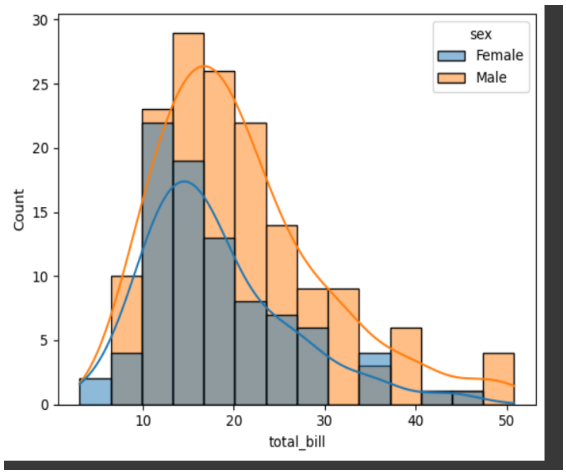


3.Histogram

Code:

```
# importing packages
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
# reading the database
data = pd.read_csv("tips.csv")
#sns.scatterplot(x='day', y='tip', data=data, hue='sex')
sns.histplot(x='total_bill',data=data,kde=True,hue='sex')
plt.show()
```

Output:

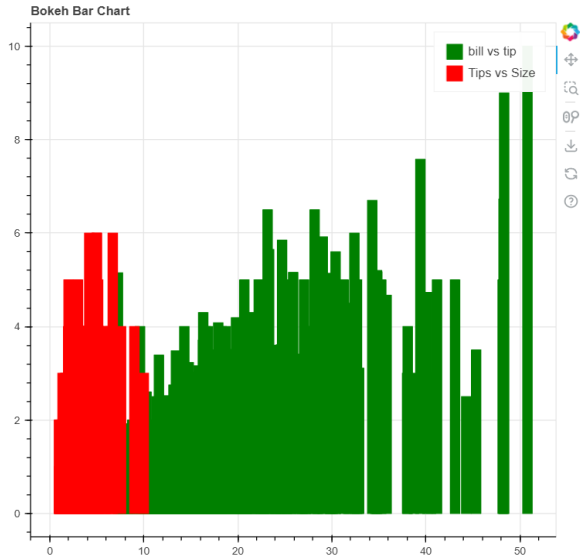


4.Broken Bar Chart

Code:

```
import bokeh.plotting
import bokeh.models
import bokeh.layouts
import bokeh.io

# importing the modules
from bokeh.plotting import figure, output_file, show
import pandas as pd
graph=figure(title="Bokeh Bar Chart")
output_file("bar.html")
data=pd.read_csv("tips.csv")
graph.vbar(x=data['total_bill'],top=data['tip'],legend_label="bill vs tip",
           color='green')
graph.vbar(data['tip'],top=data['size'],legend_label="Tips vs Size",
           color='red')
graph.legend.click_policy="hide"
show(graph)
```

Output:

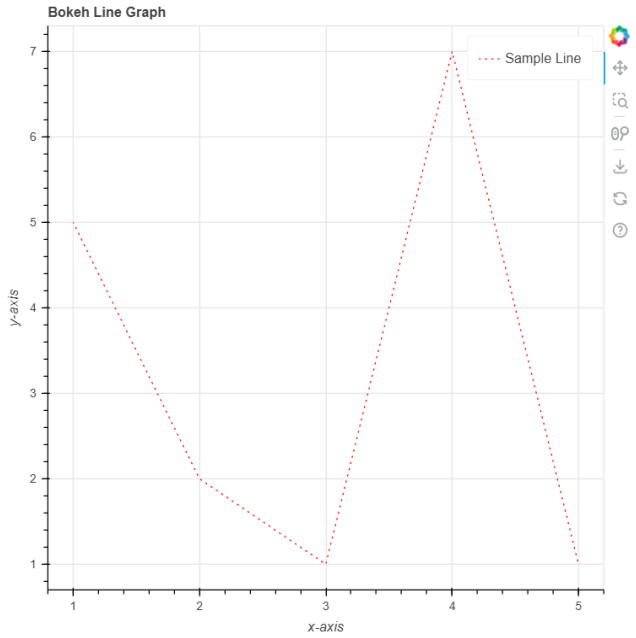


5.Broken Line Graph

Code:

```
# importing the modules
from bokeh.plotting import figure, output_file, show
# file to save the model
output_file("gfg.html")
# instantiating the figure object
graph = figure(title = "Bokeh Line Graph")
# name of the x-axis
graph.xaxis.axis_label = "x-axis"
# name of the y-axis
graph.yaxis.axis_label = "y-axis"
# the points to be plotted
x = [1, 2, 3, 4, 5]
y = [5, 2, 1, 7, 1]
# color of the line
line_color = "red"
# type of line
line_dash = "dotted"
# offset of line dash
line_dash_offset = 1
# name of the legend
legend_label = "Sample Line"
# plotting the line graph for AAPL
graph.line(x, y,
           line_color = line_color,
           line_dash = line_dash,
           line_dash_offset = line_dash_offset,
           legend_label = legend_label)
# displaying the model
show(graph)
```

Output:



6.Button,CheckBox and Radio Button

Code:

```
from bokeh.io import show
from bokeh.models import Button,CheckboxGroup,RadioGroup,CustomJS
output_file("bar1.html")
button=Button(label="Click Me",button_type="success")
button.js_on_click(CustomJS(code="alert('Button Clicked')"))
L=["First","Second","third"]
checkbox_group=CheckboxGroup(labels=L,active=[0,2])
checkbox_group.js_event_callbacks['active'] = [CustomJS(code="""console.log('checkbox_group: value=' + this.active, this.toString());""")]
radio_group=RadioGroup(labels=L,active=0)
radio_group.js_event_callbacks['active'] = [CustomJS(code="alert('Button Clicked')")]
show(button)
show(radio_group)
show(checkbox_group)
```

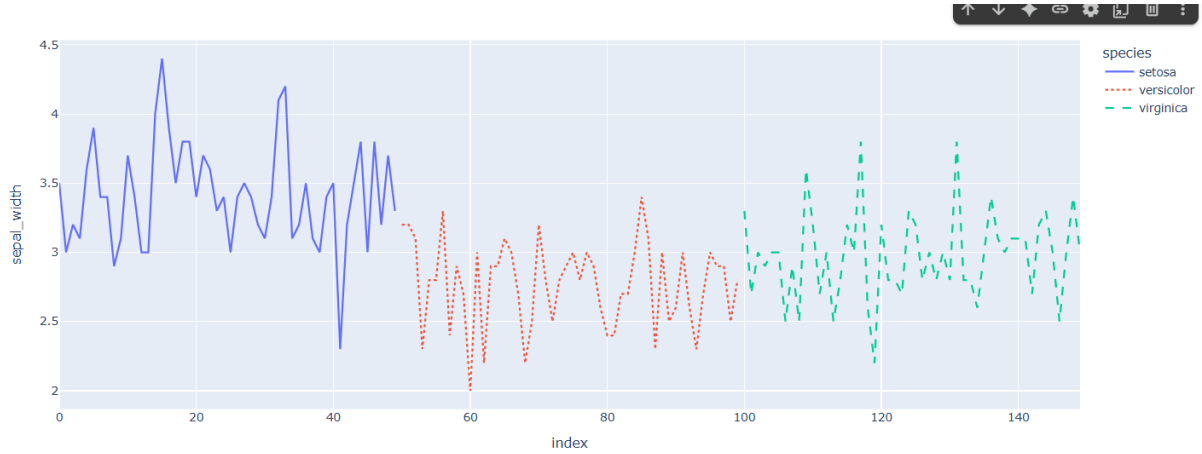
Output:

7.Interactive Plotly figure

Code:

```
import plotly.express as px
df=px.data.iris()
fig=px.line(df,y="sepal_width",line_dash='species',color='species')
fig.show()
```

Output:



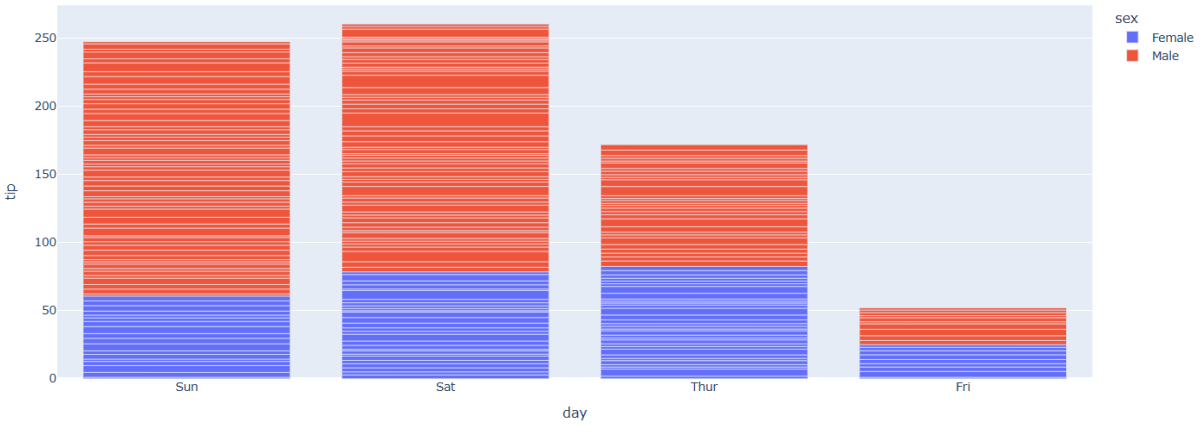
8.Different Visualiztion using Plotly.express

Code:

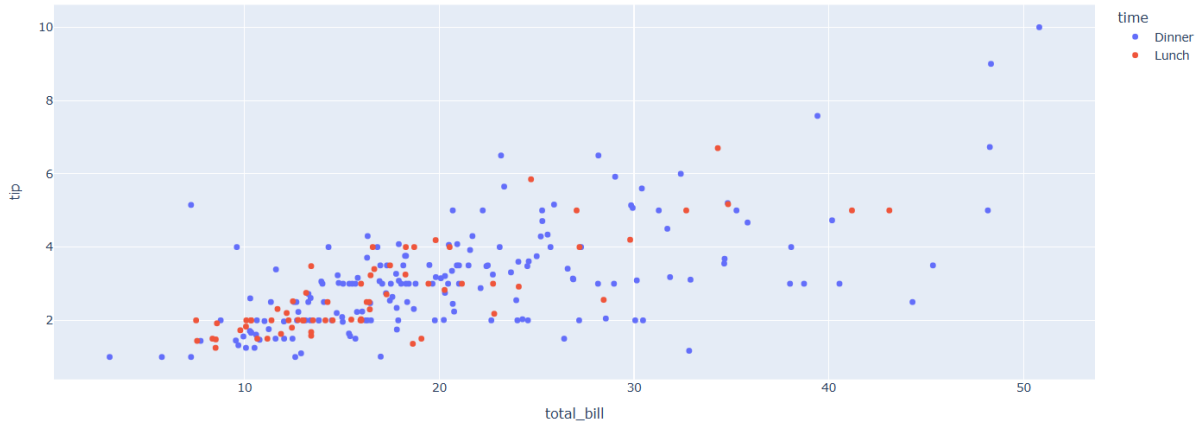
```
import plotly.express as px
df=px.data.tips()
fig=px.bar(df,x='day',y='tip',color='sex')
fig1=px.scatter(df,x='total_bill',y='tip',color='time')
fig2=px.histogram(df,x="total_bill", color='sex')
fig4=px.pie(df,values="total_bill",names="day")
fig3=px.pie(df,values="total_bill",names="day",color_discrete_sequence=px.colors.sequential.RdBu,opacity=0.7,hole=0.1)
fig.show()
fig1.show()
fig2.show()
fig3.show()
fig4.show()
```

Output:

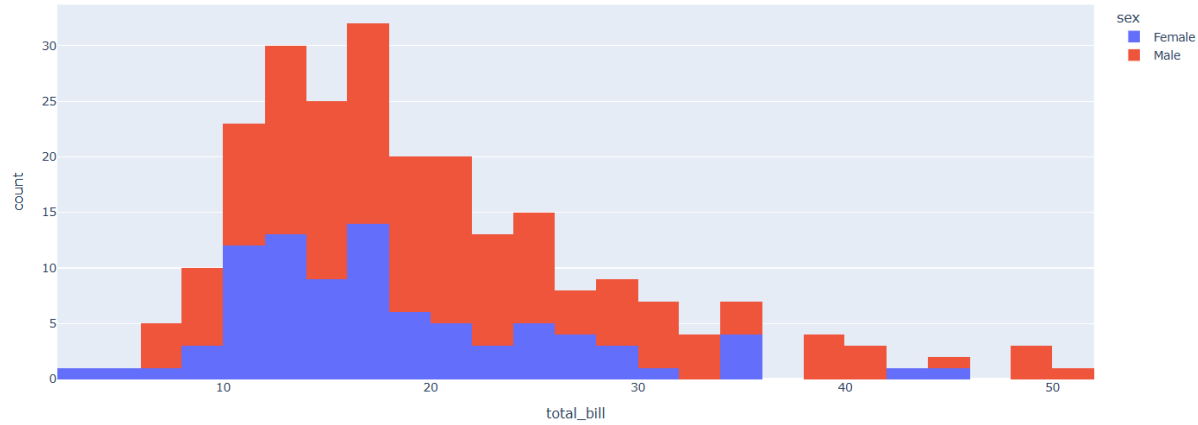
- **Bar plot**



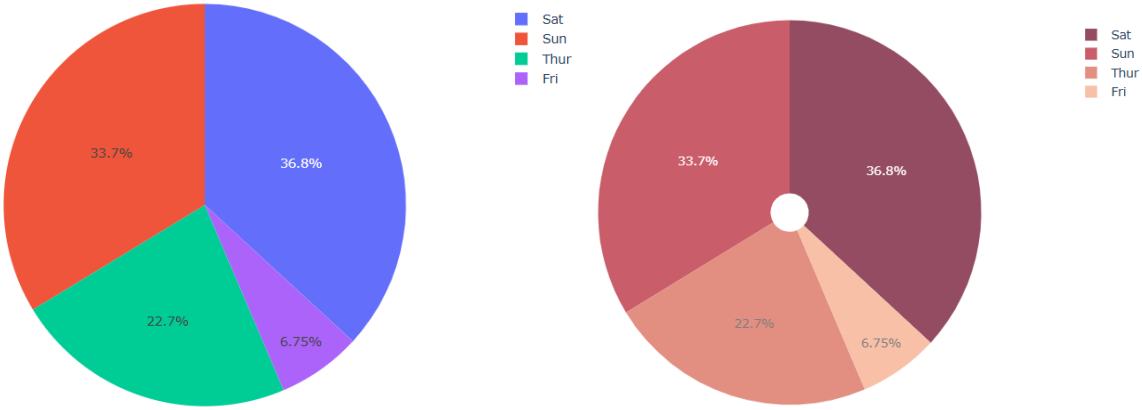
- **Scatter Plot**



- **Histogram**



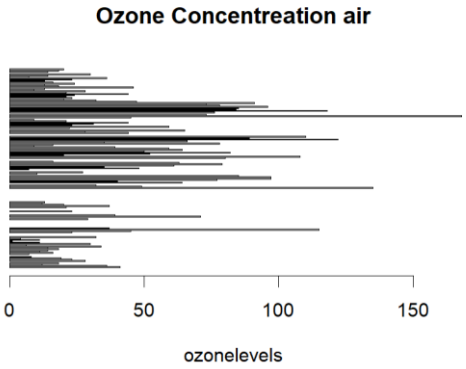
- **Pie chart**



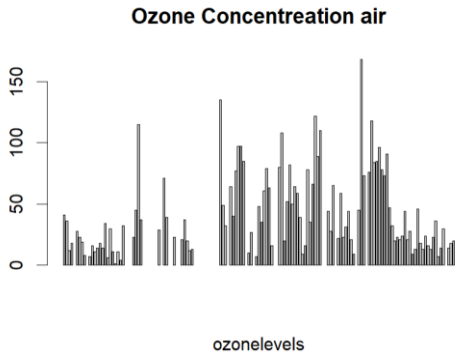
Practical 2

Aim: Show Basic Visualization in R

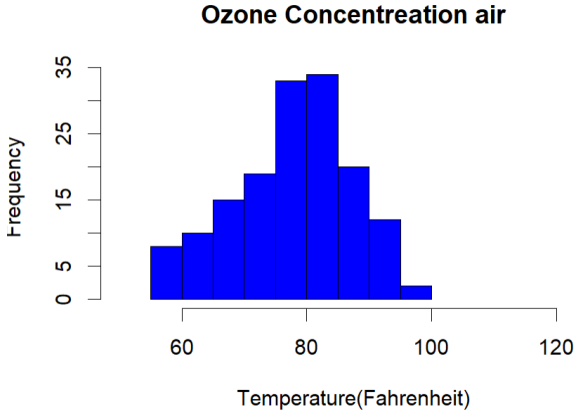
```
> barplot(airquality$Ozone,main='Ozone Concentreation air',
xlab="ozonelevels",horiz=TRUE)
>
```



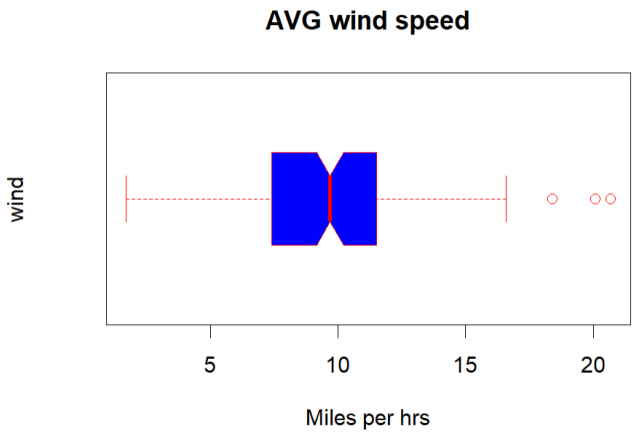

```
> barplot(airquality$Ozone,main='Ozone Concentreation air',
xlab="ozonelevels",horiz=FALSE)
>
```



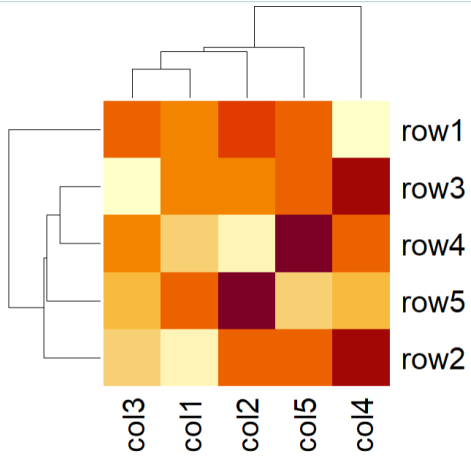

```
> hist(airquality$Temp,main = "Ozone Concentreation air",xl
ab="Temperature(Fahrenheit)",xlim=c(50,125),col="blue",freq
=TRUE)
>
```



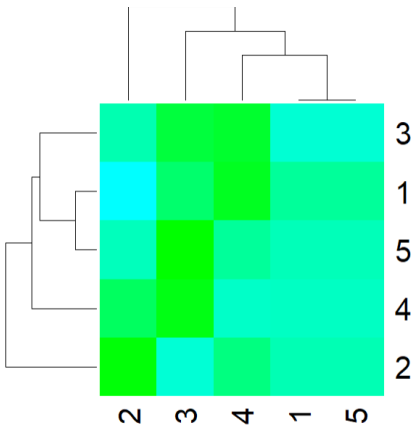
```
> boxplot(airquality$Wind,main="AVG wind speed",xlab="Miles
per hrs",ylab="wind",col='blue',border = 'red',horizontal =
TRUE,notch = TRUE)
>
```




```
data<-matrix(rnorm(20,1,5),nrow=5,ncol=5)
colnames(data)<-paste0("col",3:5)
row.names(data)<-paste0("row",6:5)
heatmap(data)
```




```
> mycolors<-colorRampPalette(c("green","cyan"))
> heatmap(data,col=mycolors(100))
```



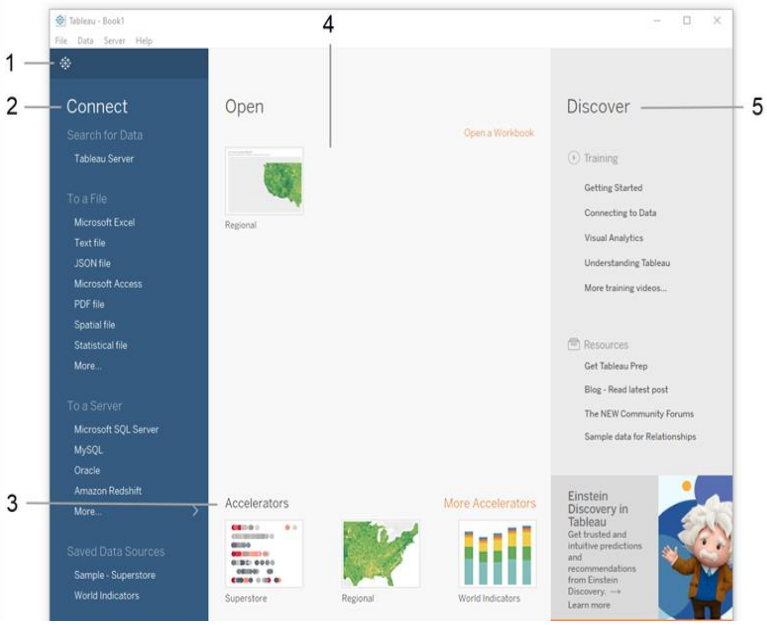
Practical 3

Aim: Connecting to Data and preparing data for visualization in Tableau

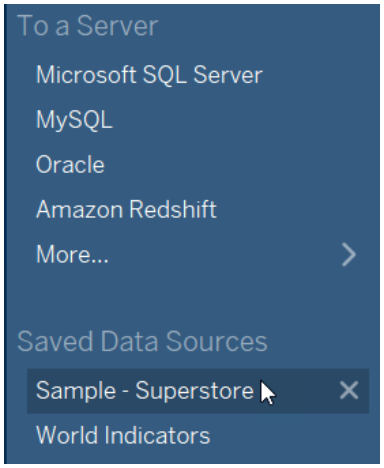
Step 1: Connect to Your Data

To analyze overall sales and profitability, start by connecting to your data using Tableau Desktop.

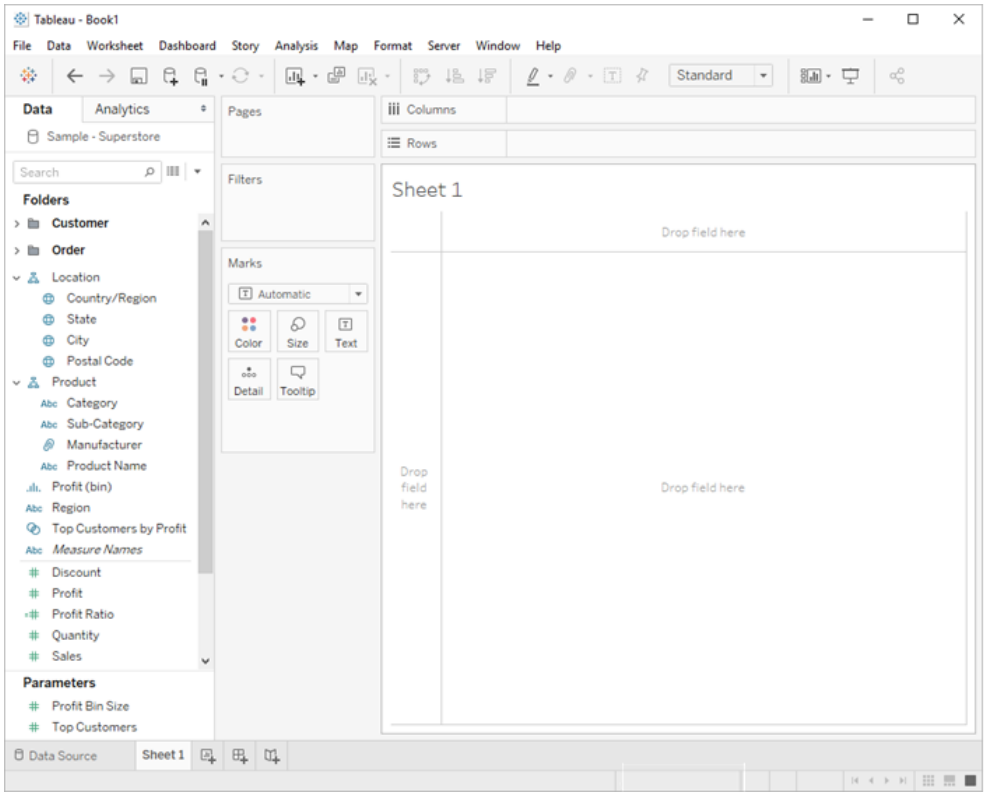
1. **Open Tableau Desktop**
    - The start page appears, where you can choose how to connect to your data.
2. **Tableau Start Page Options**
    - **Tableau icon:** Toggle between start page and authoring workspace.
    - **Connect pane:**
        - Connect to files (Excel, PDF, spatial files, etc.).
        - Connect to servers (Tableau Server, SQL Server, Google Analytics, etc.).
        - Connect to previously used data sources.
    - **Accelerators:** Access sample workbooks and prebuilt templates.
    - **Open:** Open existing workbooks.
    - **Discover:** Find tutorials, forums, and "Viz of the Week" ideas.



3. **Connect to Sample Data**
    - Browse and select file., click tips.csv



4. **Result**
    - After connecting, a **blank worksheet** appears, ready for analysis.



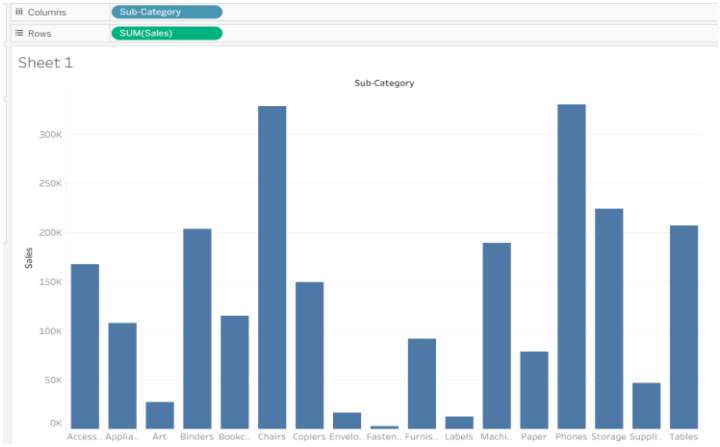
Practical 4

Aim: Use Data aggregation and statistical functions in Tableau

1. SUM (Total)

Steps:

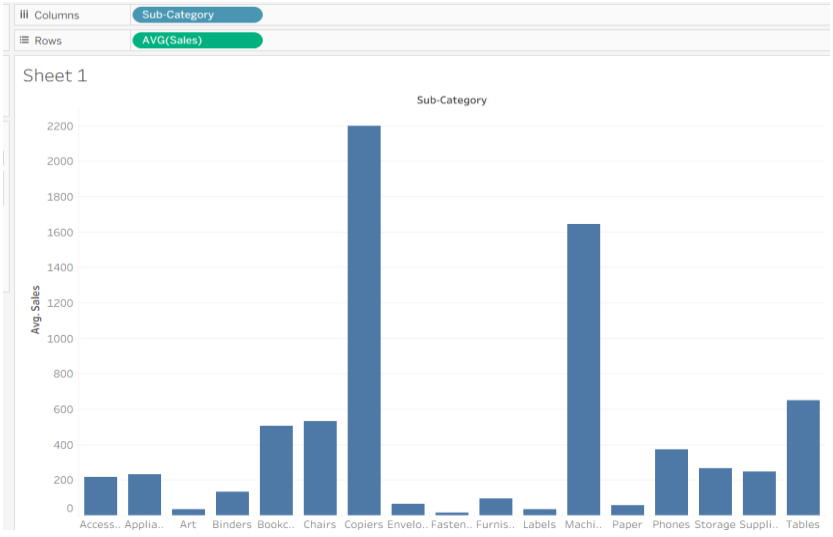
1. Connect to Sample - Superstore.xls.
   Drag Category to Columns.
2. Drag Sales to Rows.
3. By default, Tableau shows the SUM of Sales.



2. AVG (Average)

Steps:

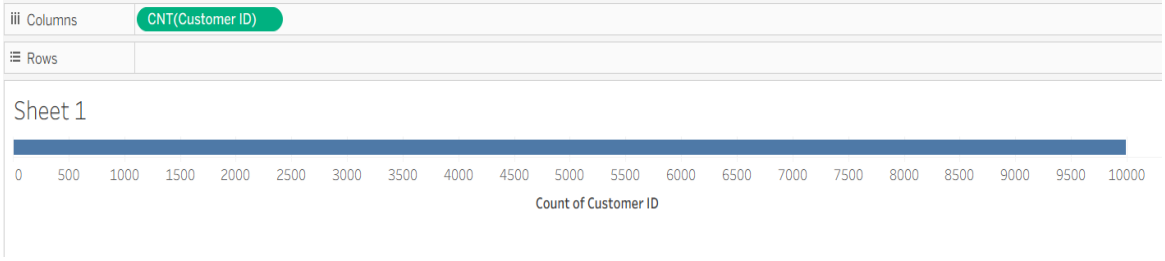
1. Right-click the Sales pill on Rows.
2. Select Measure → Average.
3. Tableau updates the chart to show average sales per category.



3. COUNT

Steps:

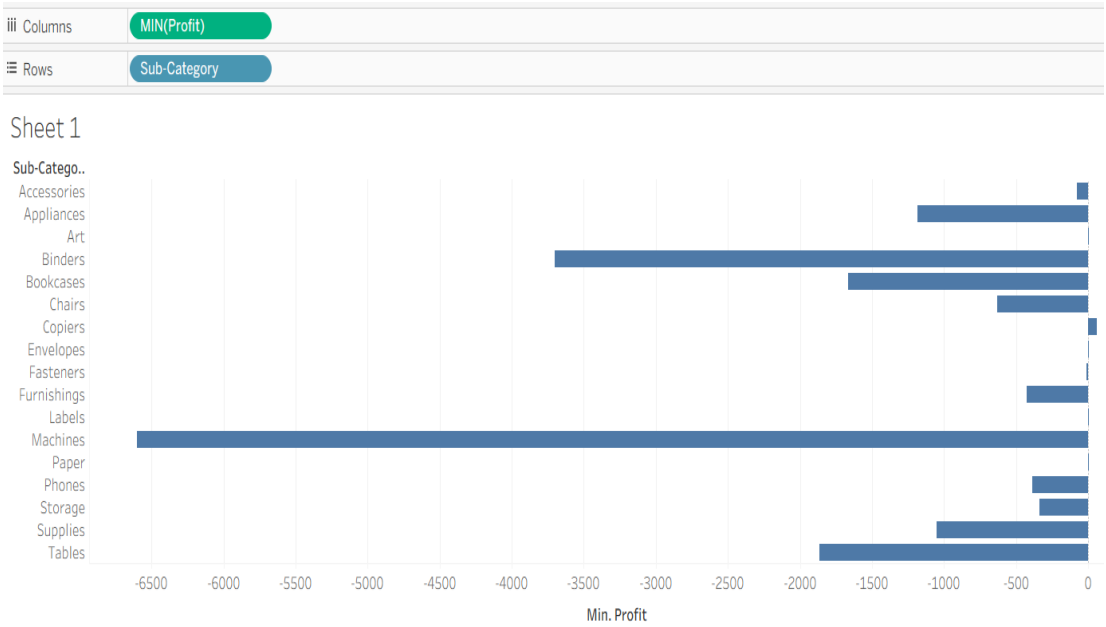
1. Drag Customer ID to Rows.
2. Right-click it → Measure → Count.
3. Tableau shows the number of customers.



4. MIN (Minimum)

Steps:

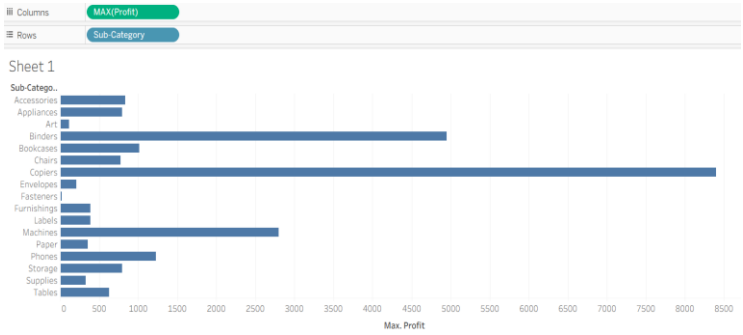
1. Drag Profit to Rows and Sub-Category to Columns.
2. Right-click Profit → Measure → Minimum.
3. Tableau shows the lowest profit value for each sub-category.



5. MAX (Maximum)

Steps:

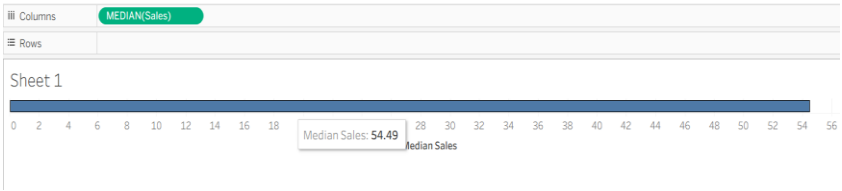
1. Right-click Profit → Measure → Maximum.
2. Tableau now shows the highest profit value for each sub-category.



6. Median

Steps:

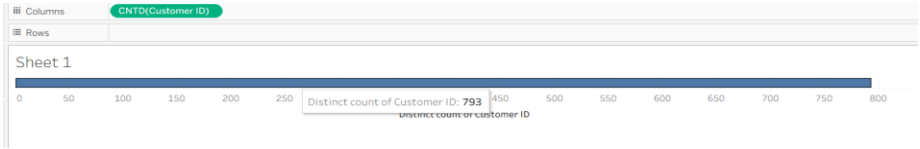
1. Drag a measure (e.g., **Sales**) to Rows.
2. Right-click → **Measure** → **Median**.
3. Tableau shows the **middle value** (50th percentile) of sales distribution.



7. Count (Distinct)

Steps:

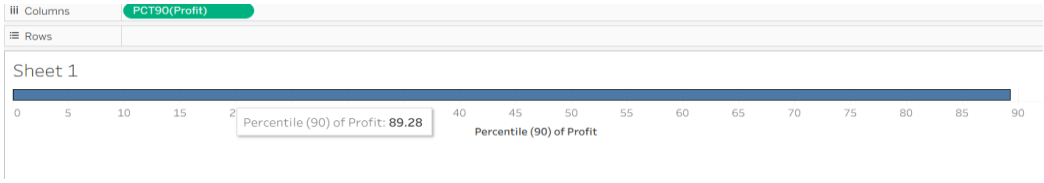
1. Drag **Customer ID** to Rows.
2. Right-click → **Measure** → **Count (Distinct)**.
3. Tableau counts the **unique customers** (ignores duplicates).



8. Percentile

Steps:

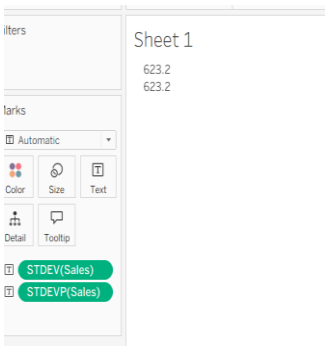
1. Drag a measure (e.g., **Profit**) to view.
2. Right-click → **Measure** → **Percentile** → **25, 50, 75, etc.**
3. Tableau calculates the **percentile value** for profit distribution.
    - Example: 90th percentile = value below which 90% of profits fall.



9. Standard Deviation (Std. Dev / Std. Dev (Pop.))

Steps:

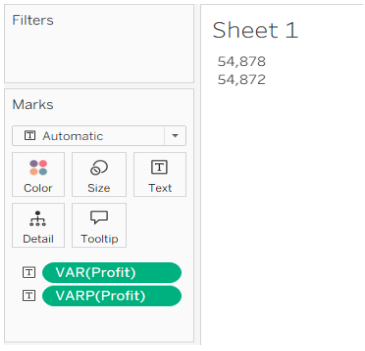
1. Drag a measure (e.g., **Sales**) to view.
2. Right-click → **Measure** → **Standard Deviation**.
    - **Std. Dev** → Sample standard deviation.
    - **Std. Dev (Pop.)** → Population standard deviation.
3. Shows how much values vary from the mean.



10. Variance (Var / Var (Pop.))

Steps:

1. Drag a measure (e.g., **Profit**) to view.
2. Right-click → **Measure** → **Variance**.
    - **Variance** → Sample variance.
    - **Variance (Pop.)** → Population variance.
3. Indicates the **spread of data** (square of Std. Dev).

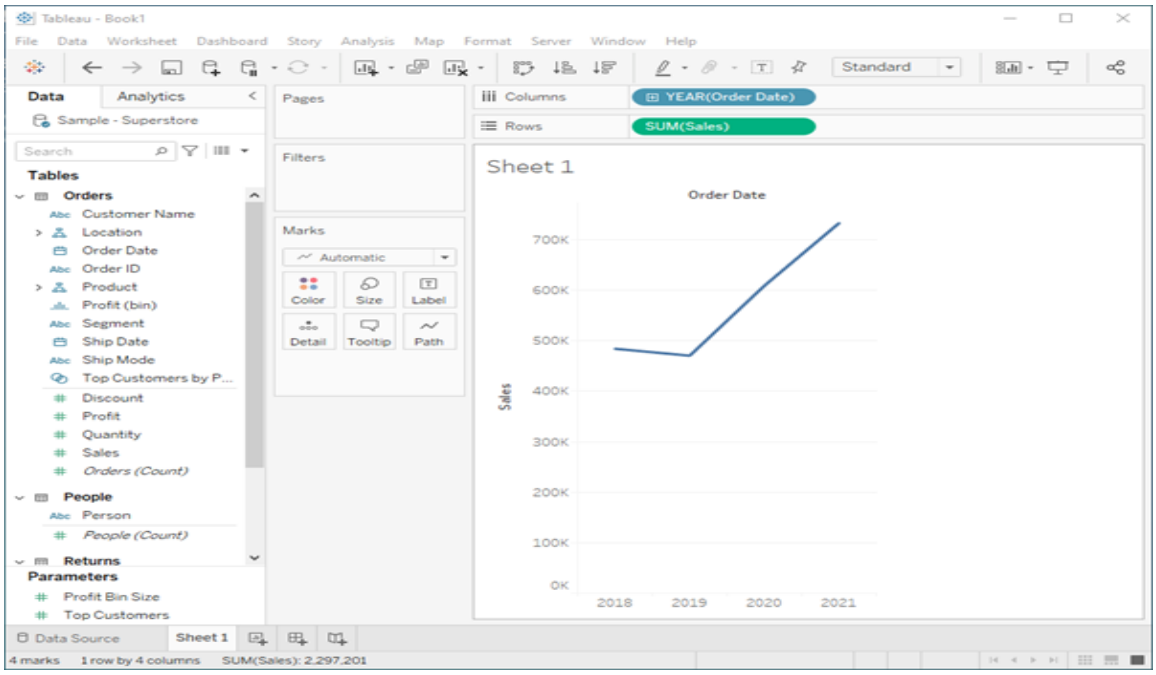


Practical 5

Aim: Show Data Visualization using Tableau

Step 1: Create a Basic Chart

1. From the **Data pane**, drag **Order Date** to the **Columns** shelf.
    - Tableau creates a column for each year.
    - The Abc indicator below each column shows where text or numerical data can be placed.
2. Drag **Sales** to the **Rows** shelf.
    - Tableau generates a **line chart** with sales aggregated by year.
    - This shows total sales increasing over time, giving a general overview.

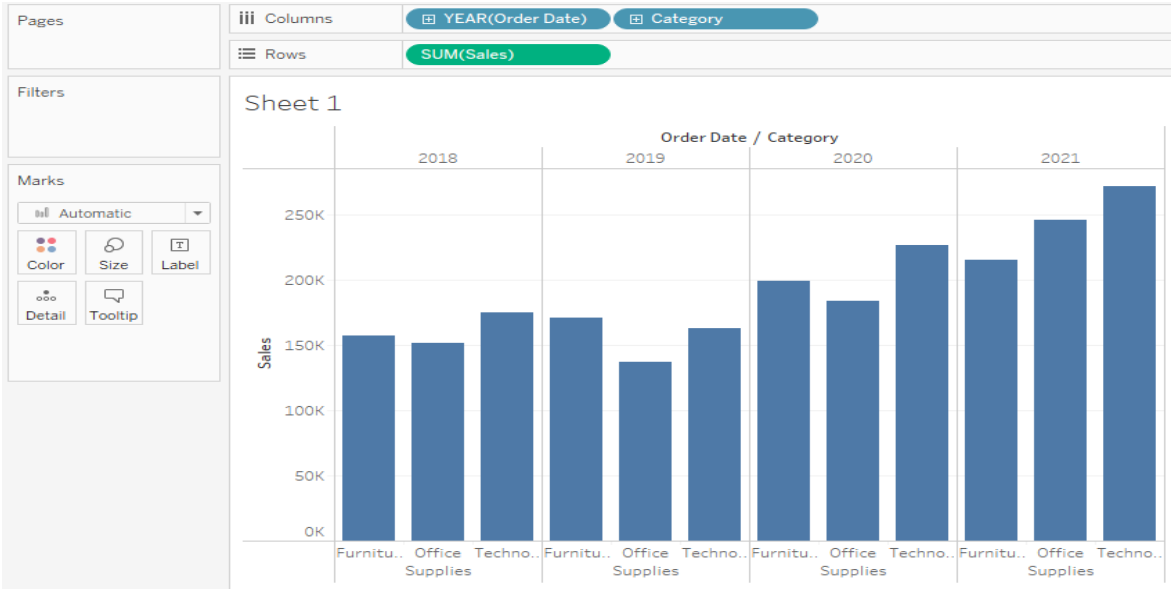


Step 2: Refine Your View

1. To gain more insight, add **product categories**:
    - Drag **Category** to the **Columns** shelf, next to **YEAR(Order Date)**.
    - Tableau updates the view to a **bar chart**, showing sales totals for each category by year.
    - This lets you compare categories like Furniture, Office Supplies, and Technology.

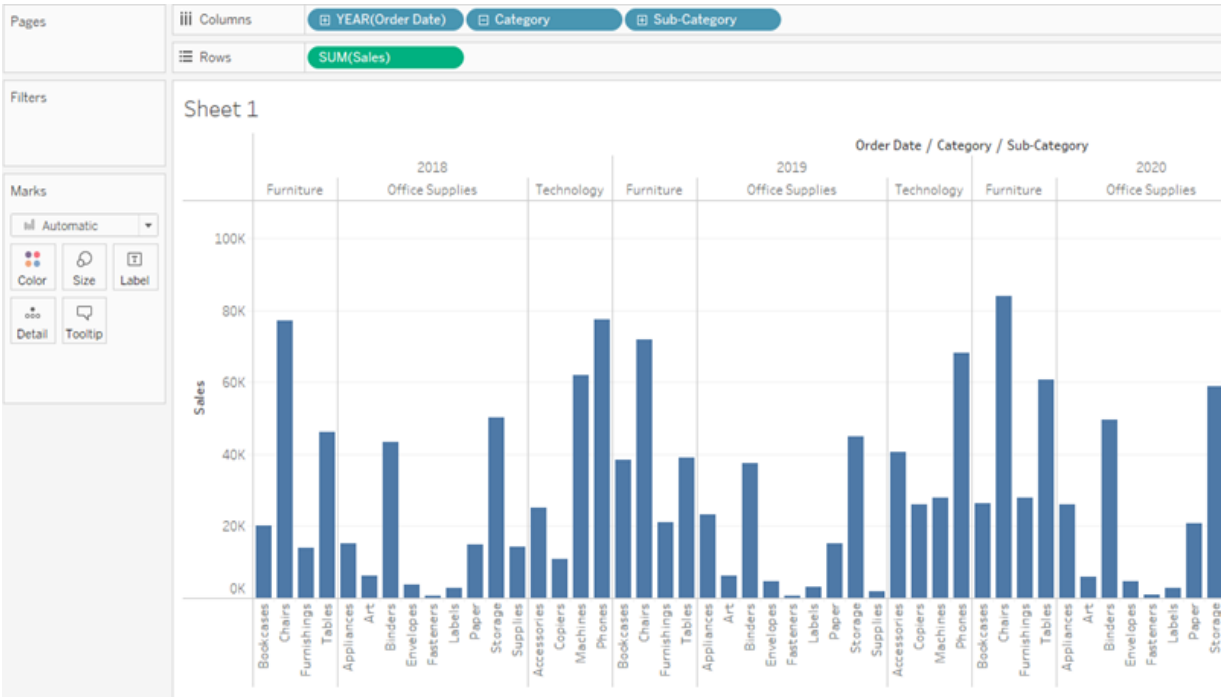
- Insight: Furniture sales are growing faster than Office Supplies, suggesting potential focus areas.

2. Explore **sub-categories** for more detail:
    - Drag or double-click **Sub-Category** to the **Columns** shelf.
    - Tableau adds a new header and creates bars for each sub-category (e.g., bookcases, chairs, tables) broken down by category and year.
    - This helps identify top-selling items and potential areas for improving profitability.



Step 3: Preparing for Visual Exploration

- With Order Date, Category, and Sub-Category in your view, you have detailed sales data.
- Next, you can add **color, filters, and other visual cues** to highlight specific results and trends.



Step 4: Key Takeaways

- Start with a simple chart to understand overall trends.
- Add fields to adjust the **level of detail** in your view.
- Refine by breaking data into categories and sub-categories to identify insights.
- This approach helps focus on areas for improvement and prepares your dashboard for further exploration.

Practical 6

Aim: Use dashboards of Tableau

Example Dataset Columns (common COVID dataset fields):

• Date

• Country/Region

• New Cases

• New Deaths

• Total Cases

• Total Deaths

• Population

Step 1: Prepare and Load Data

1. Get a COVID-19 dataset — for example, the Johns Hopkins COVID-19 dataset (CSV

format).

2. Open Tableau Desktop.

3. Click Connect to Data > Text File and select your COVID-19 CSV file.

4. Tableau loads the data. Check that the fields are recognized correctly (dates as date

type, numbers as integers).

Step 2: Create Worksheets (Charts)

Sheet 1: Total Cases by state

1. Click Sheet 1.

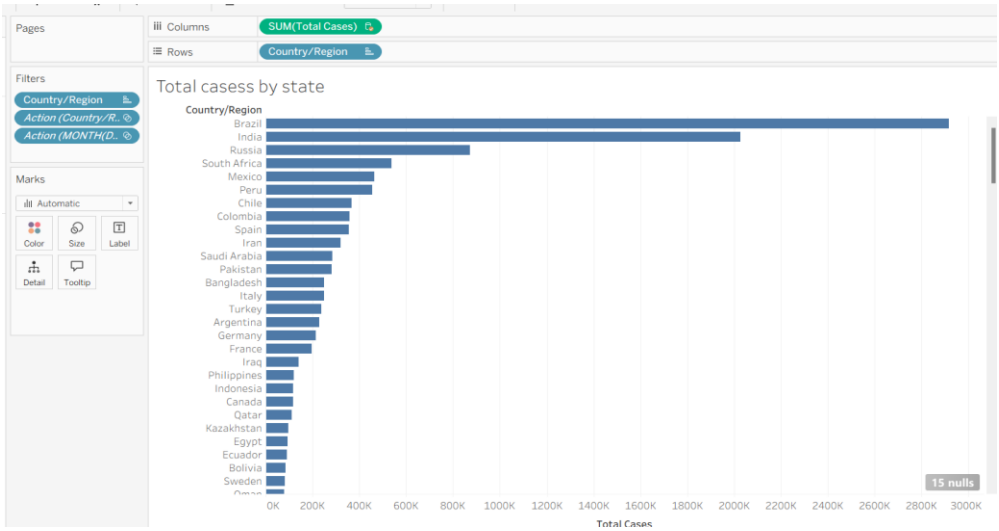
2. Drag Country/Region to Rows.

3. Drag Total Cases to Columns.

4. Sort descending by Total Cases.

5. Change to Bar Chart (default is usually fine).

6. Optional: Add Total Deaths to Color to see severity.



Sheet 2: New Cases Over Time (Line Chart)

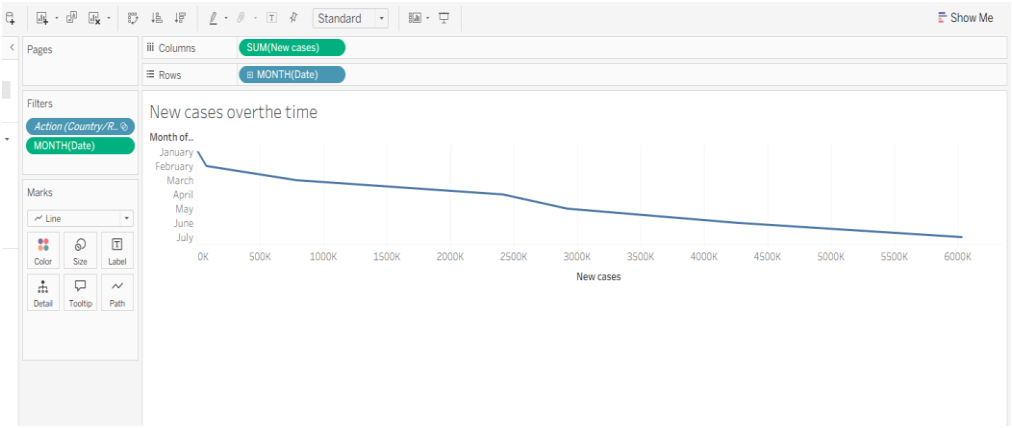
1. Click New Worksheet.

2. Drag Date to Columns.

3. Drag New Cases to Rows.

4. Set Date as a continuous Day or Month.

5. This shows how new cases change over time.



Sheet 3: Death Rate by Country (Calculated Field + Bar Chart)

1. Create a calculated field:

Name it Death Rate (%)

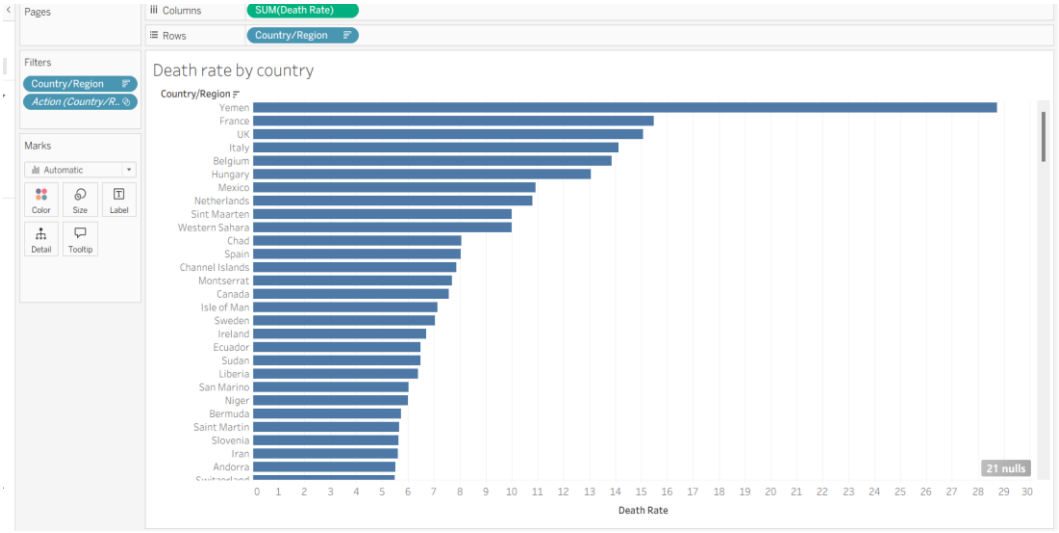
Formula: [Total Deaths] / [Total Cases] * 100

2. Drag Country/Region to Rows.

3. Drag Death Rate (%) to Columns.

4. Sort descending to see countries with highest death rates.

5. Make it a Bar Chart.



Sheet 4: Map of Total Cases

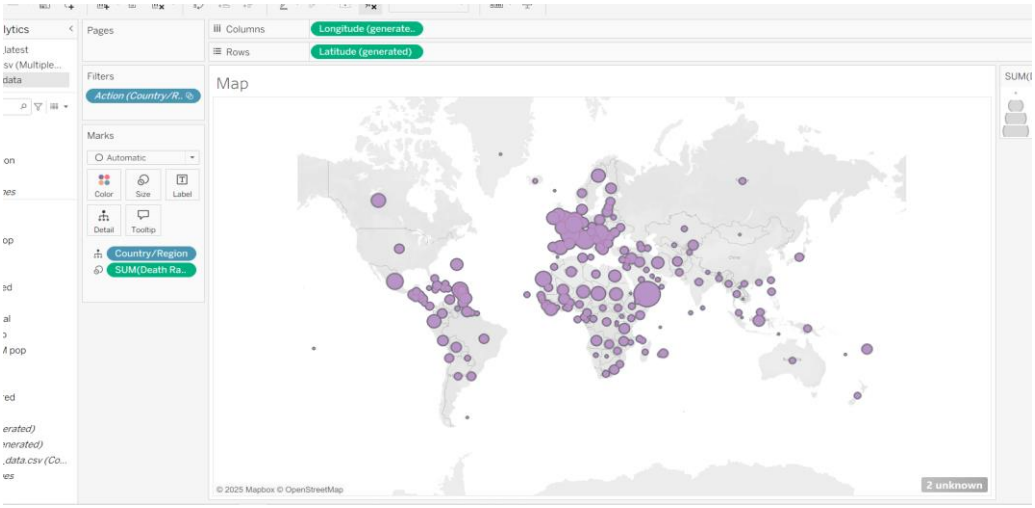
1. New Worksheet.

2. Drag Country/Region to Detail on the Marks card.

3. Drag Total Cases to Color.

4. Change Marks type to Map.

5. Adjust color gradient to visualize hotspots.



Step 3: Create Dashboard

1. Click New Dashboard tab.

2. Drag and drop sheets you created:

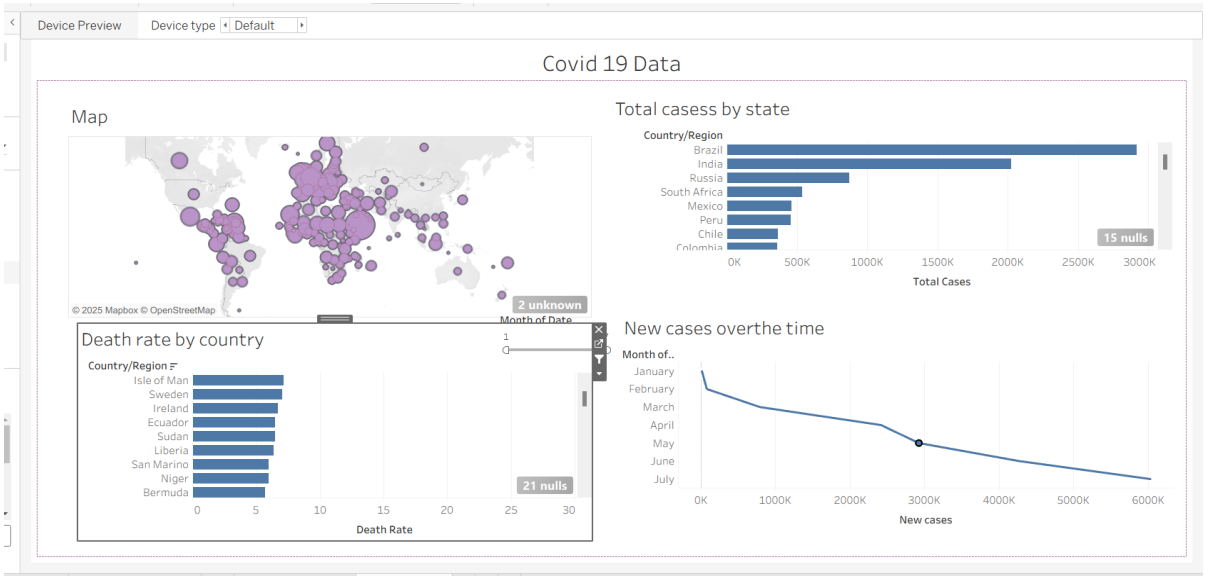
o Map (Sheet 4) at the top or left.

o Total Cases by Country (Sheet 1) next to or below the map.

o New Cases Over Time (Sheet 2) below or beside.

o Death Rate (Sheet 3) somewhere visible.

3. Resize elements for clarity.



Step 4: Add Filters and Interactivity

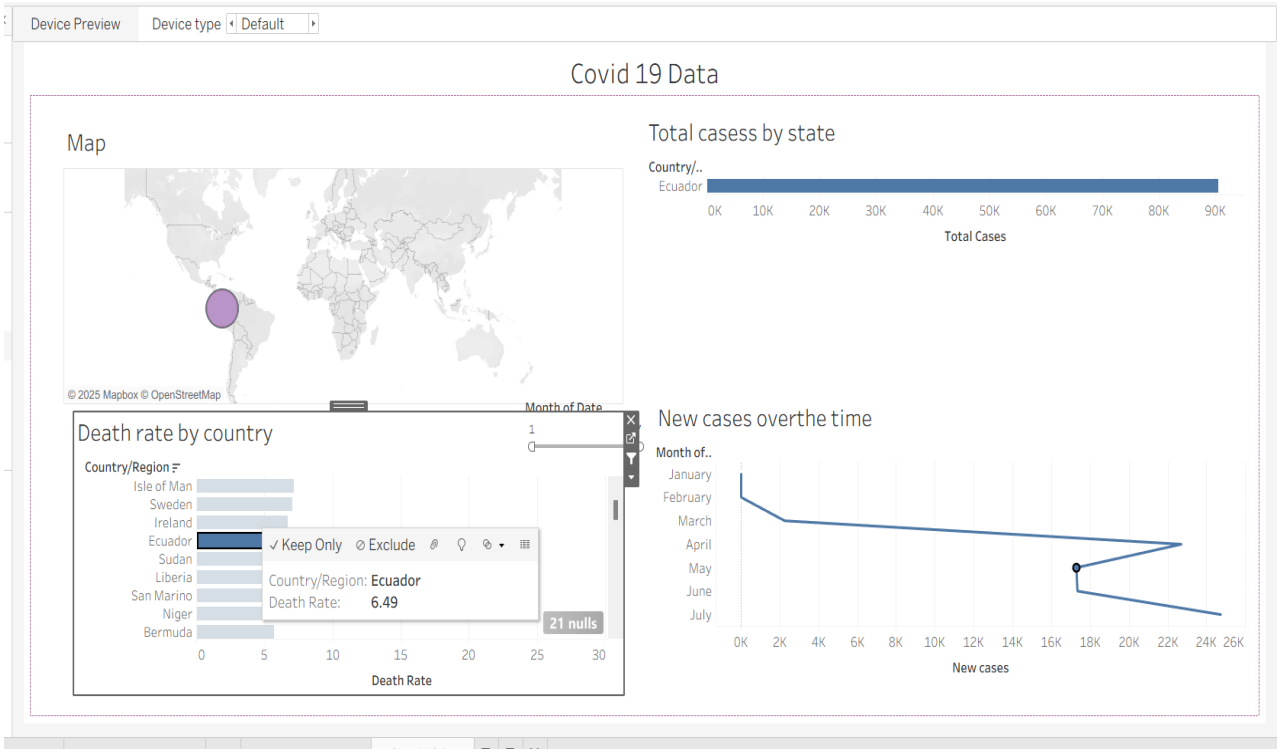
• On the dashboard, click the dropdown arrow on Sheet 1 (Total Cases by Country).

• Select Use as Filter — clicking a country filters other charts by that country.

• Add filter controls for Date or Country if needed:

o Drag Country/Region from the data pane onto Filters shelf in any sheet.

o Show filter on the dashboard for user to select countries.



Step 5: Polish and Format

• Add titles: “COVID-19 Dashboard”, “Total Cases by Country”, etc.

• Add text boxes explaining charts.

• Adjust fonts, colors for clarity and impact.

• Use device layout to optimize for different screens if needed.

Step 6: Save and Share

• Save your workbook.

• Publish on Tableau Server, Tableau Public, or export PDF/image.

Summary

Dashboard Component Purpose

Map of Total Cases Visualize hotspots geographically

Total Cases by Country (Bar) Compare countries by total cases

New Cases Over Time (Line) Track pandemic progress over time

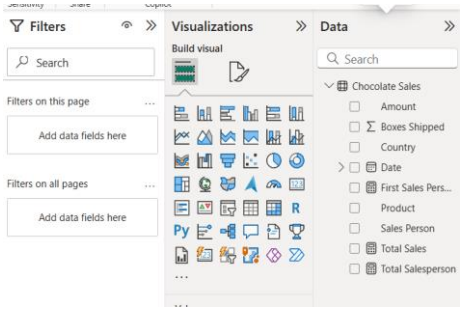
Death Rate by Country (Bar) Understand severity via death percentages

Practical 7

Aim: Show Data Visualization using PowerBi

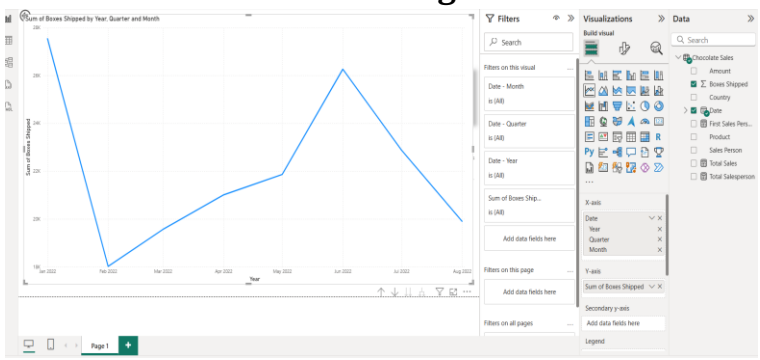
Step 1: Import Dataset

1. Launch **Power BI Desktop**.
2. Go to **Home → Get Data → Text/CSV**.
3. Browse and select the **Chocolate Sales CSV file**.
4. Click **Load** to import the dataset.



Visualizations Created:

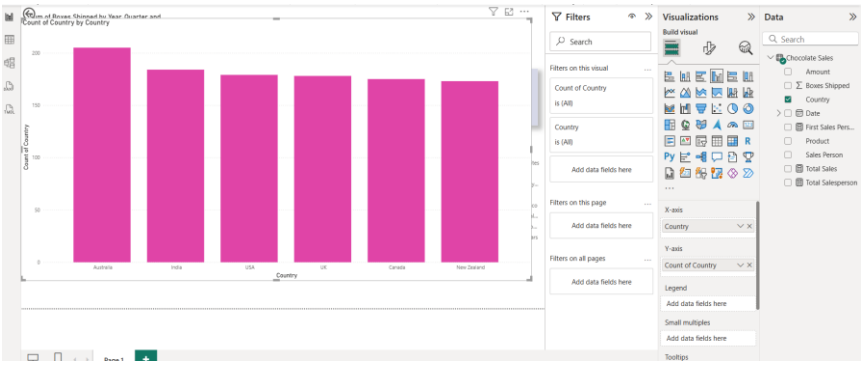
1. Line Chart – Monthly/Yearly Sales Trends

- **Purpose:** To analyze the trend of boxes shipped over time.
- **Steps:**
    1. Select **Line Chart** from the Visualizations pane.
    2. Drag Date to the X-axis.
    3. Drag Boxes Shipped to the Y-axis.
    4. Use the hierarchy to drill down by **Year → Quarter → Month**.
    5. Format the axis to show **categorical labels** and enable **data labels**.
- 

2. Bar Chart – Count of Sales Person by Country

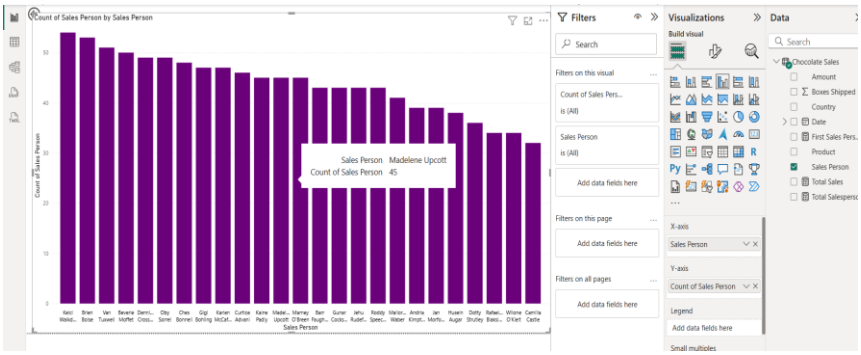
- **Purpose:** To show the number of salespersons in each country.
- **Steps:**

1. Select **Bar Chart**.
2. Drag Country to the X-axis.
3. Drag Sales Person to Values and set aggregation to **Count**.



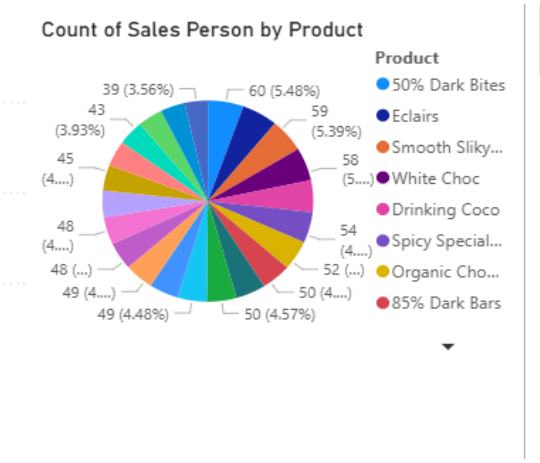
3. Bar Chart – Count of Country by Sales Person

- **Purpose:** To display how many countries each salesperson is associated with.
- **Steps:**
    1. Select **Bar Chart**.
    2. Drag Sales Person to the X-axis.
    3. Drag Country to Values and set aggregation to **Count**.



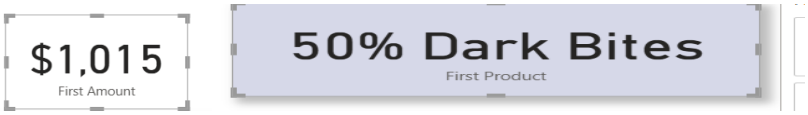
5. Pie Chart – Count of Sales Person by Product

- **Purpose:** To visualize the distribution of salespersons across products.
- **Steps:**
    1. Select **Pie Chart**.
    2. Drag Product to Legend.
    3. Drag Sales Person to Values and set aggregation to **Count**.



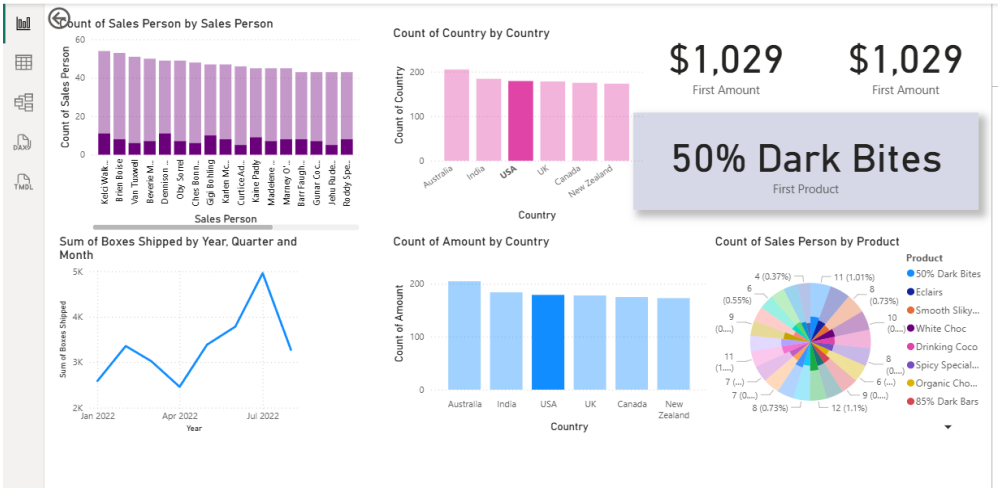
6. Card Visuals – First Amount and First Product

- **Purpose:** To highlight key metrics.
- **Steps:**
    1. Select **Card Visual**.
    2. Drag Amount to the card and apply filter for first transaction.
    3. Repeat with Product to show "First Product".



Filters Applied:

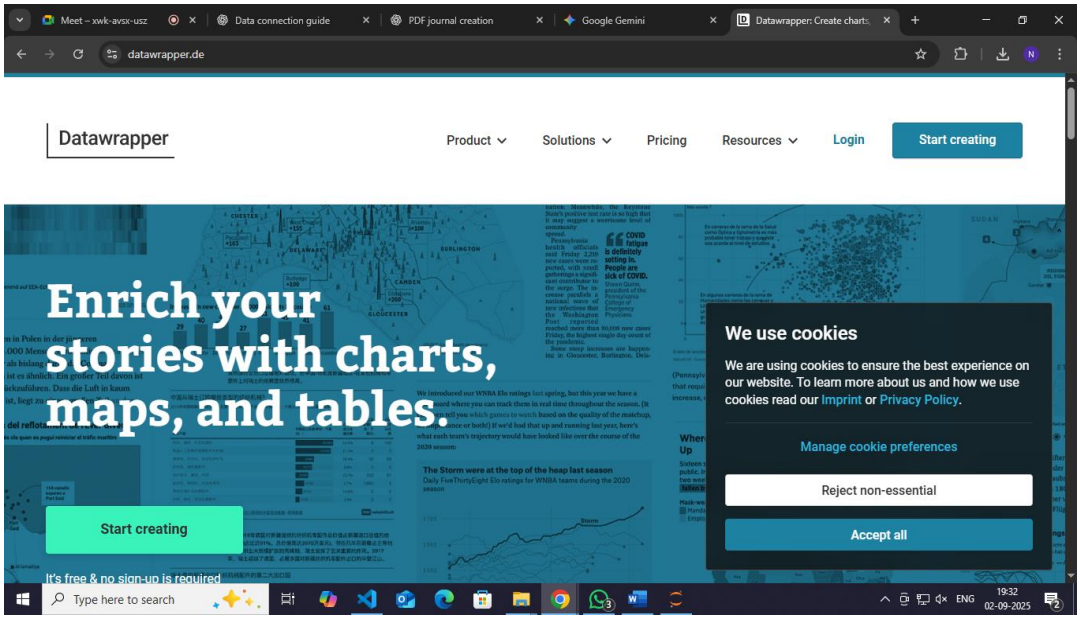
- **Visual Level Filters:** Applied to individual charts for focused analysis.
- **Page Level Filters:** Used to filter all visuals on the current report page.
- **Fields Used for Filtering:**
    - Country
    - Sales Person
    - Date



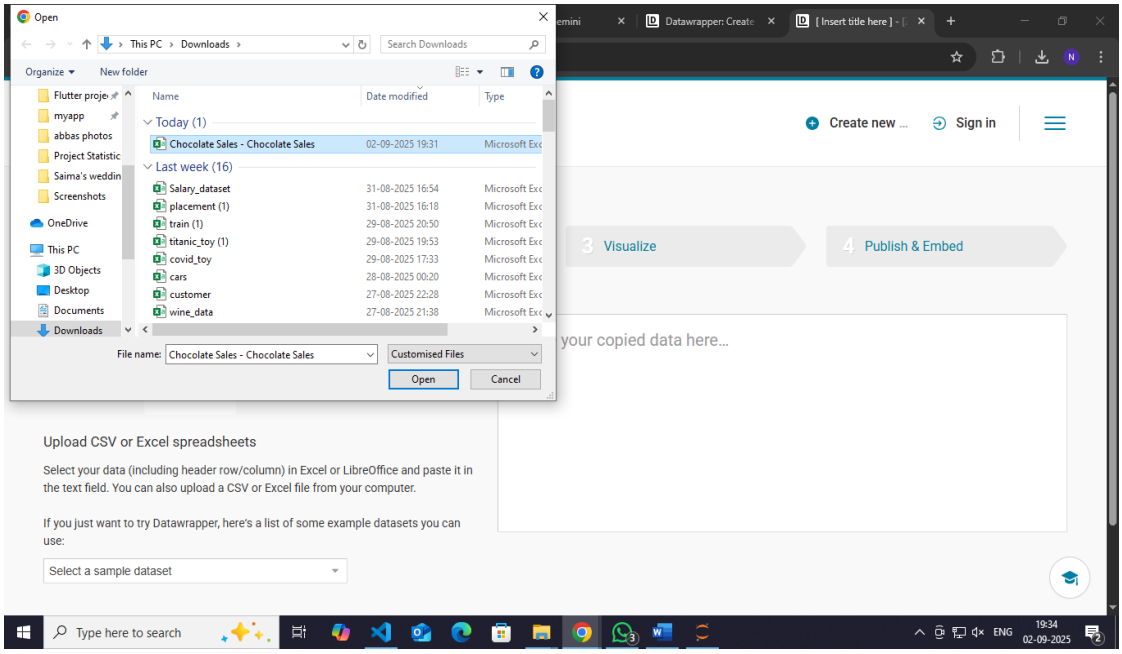
Practical 8

Aim: Show Data Visualization using DataWrapper

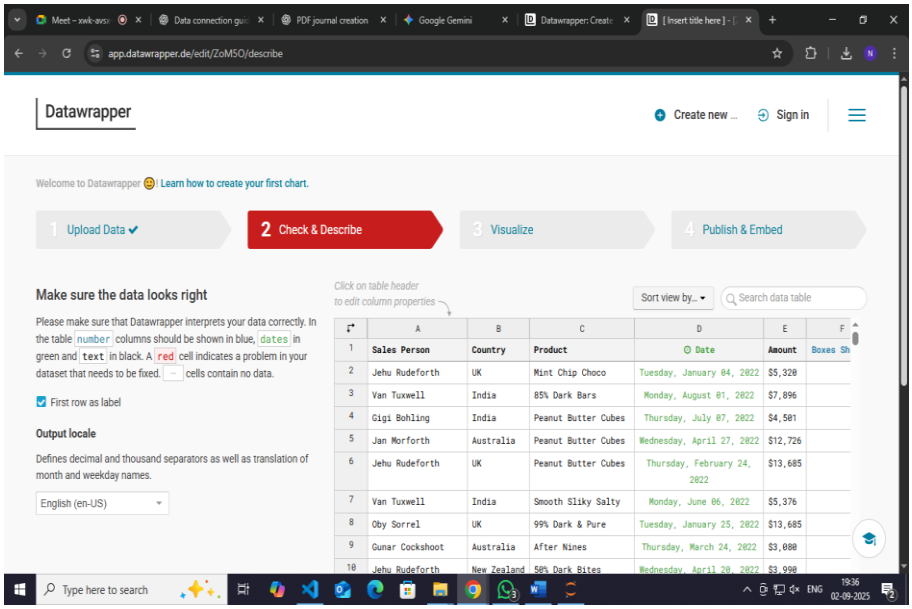
Step 1: Open DataWrapper Website



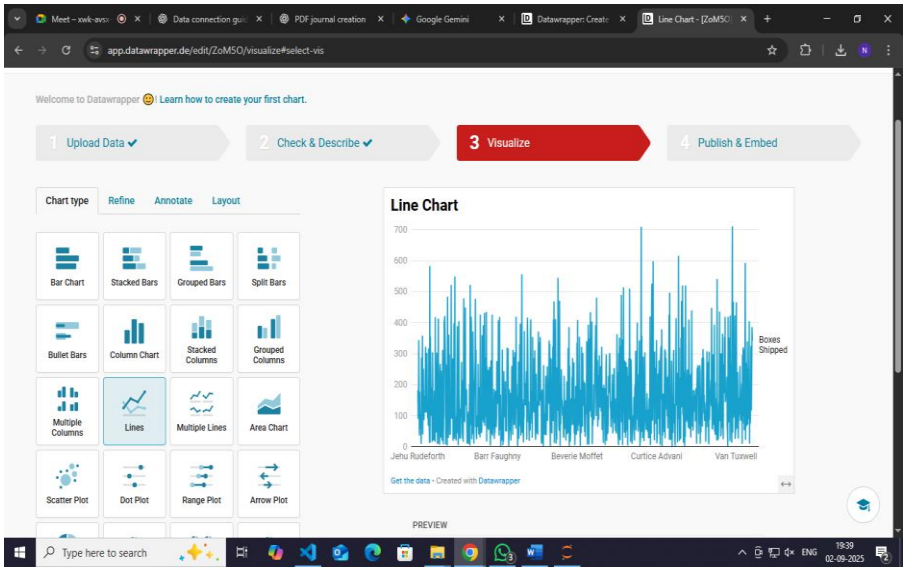
Step 2: Click on 'Start creating' button, upload dataset file



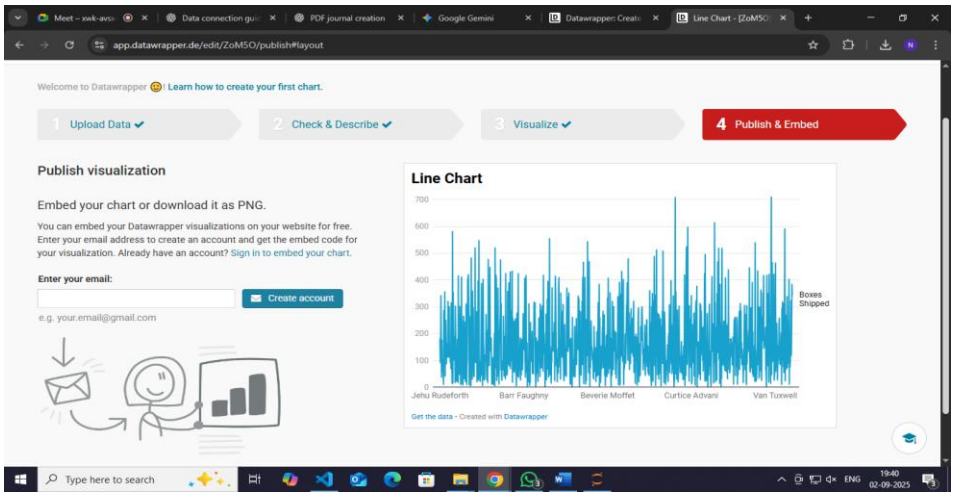
Step 3: Click on 'proceed' button, check your uploaded data.



Step 4: Click on ‘proceed’ button, visualize your data using different charts



Step 5: Click on ‘proceed’ button, publish or download your charts



Practical 9

Aim: Show Data Visualization using Gantt Chart

Excel

Step 1: Set Up Your Data

First, you need a table with at least three columns: **Task**, **Start Date**, and **Duration** (the number of days the task will take).

Task Name	Start Date	Duration	End Date
Project Kick-off	02-Sep-25	1	02-Sep-25
Phase 1: Research	03-Sep-25	5	07-Sep-25
Phase 2: Design	08-Sep-25	7	14-Sep-25
Phase 3: Development	15-Sep-25	10	24-Sep-25
Testing & QA	25-Sep-25	5	29-Sep-25
Project Launch	30-Sep-25	1	30-Sep-25

Step 2: Insert a Stacked Bar Chart

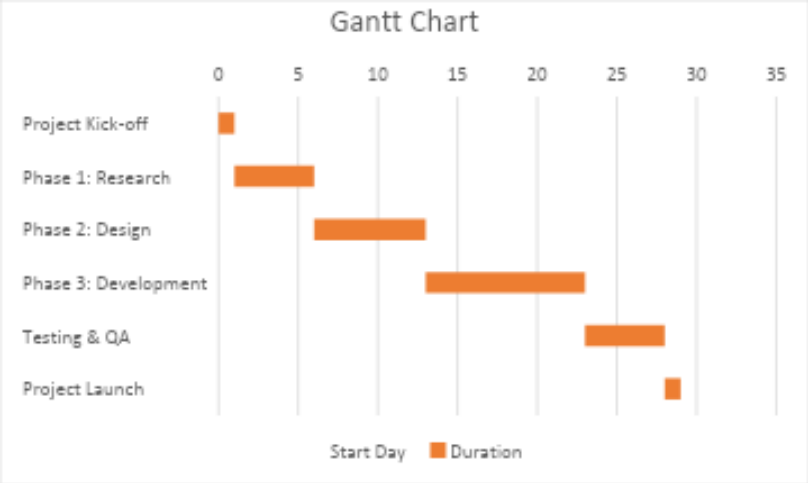
1. Select the data for your **Task Name**, **Start Date**, and **Duration**. Do *not* select the End Date column.
2. Go to the **Insert** tab on the ribbon.
3. Click on **Chart** and choose **Bar**, then select **Stacked Bar**.

Step 3: Format the Chart

This is the most important part. You need to format the chart to look like a Gantt chart.

1. **Hide the 'Start Date' Bars**:
    - Click on any of the blue bars (the 'Start Date' series) to select them all.
    - Right-click and select **Format Data Series**.
    - In the format pane, go to the **Fill & Line** (paint bucket) icon.
    - Under **Fill**, select **No fill**. This makes the first part of each bar invisible.

2. **Reverse the Task Order**:
    - Click on the list of tasks on the left side of the chart (the Y-axis).
    - Right-click and select **Format Axis**.
    - Under **Axis Options**, check the box for **Categories in reverse order**. This will put your first task at the top.
3. **Adjust the Date Range**:
    - Click on the dates at the top of the chart (the X-axis).
    - Right-click and select **Format Axis**.
    - You can set the **Minimum** and **Maximum** bounds to fit your project's start and end dates more closely.



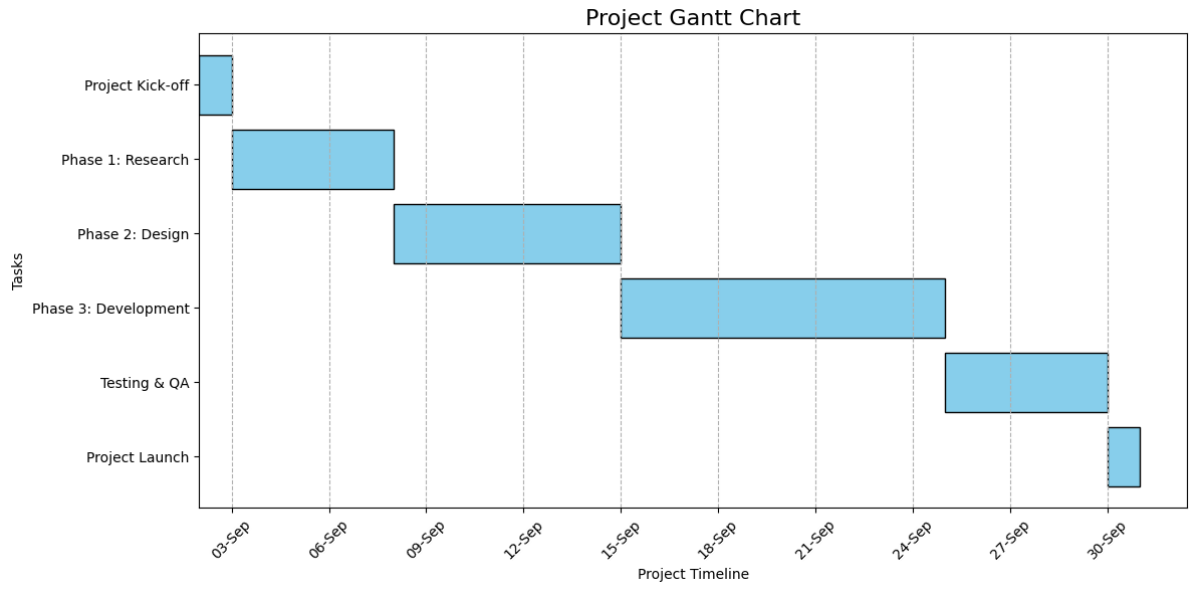
Python

Source Code:

```
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
# 1. Create the project data as a pandas DataFrame
data = {
    "Task": [
        "Project Kick-off",
        "Phase 1: Research",
```

```
        "Phase 2: Design",
        "Phase 3: Development",
        "Testing & QA",
        "Project Launch",
    ],
    "Start": [
        "2025-09-02",
        "2025-09-03",
        "2025-09-08",
        "2025-09-15",
        "2025-09-25",
        "2025-09-30",
    ],
    "End": [
        "2025-09-02",
        "2025-09-07",
        "2025-09-14",
        "2025-09-24",
        "2025-09-29",
        "2025-09-30",
    ],
}
df = pd.DataFrame(data)
# 2. Convert date strings to datetime objects
df['Start'] = pd.to_datetime(df['Start'])
df['End'] = pd.to_datetime(df['End'])
df['Duration'] = (df['End'] - df['Start']).dt.days + 1
```

```
# 3. Create the plot
fig, ax = plt.subplots(figsize=(12, 6))
ax.barh(y=df['Task'], width=df['Duration'], left=df['Start'], color='skyblue', edgecolor='black')
# 4. Format the chart
ax.xaxis.set_major_locator(mdates.DayLocator(interval=3)) # Set major ticks every 3 days
ax.xaxis.set_major_formatter(mdates.DateFormatter('%d-%b')) # Format date as 'DD-Mon'
plt.xticks(rotation=45)
ax.set_xlabel('Project Timeline')
ax.set_ylabel('Tasks')
ax.set_title('Project Gantt Chart', fontsize=16)
ax.invert_yaxis()
ax.grid(True, which='major', axis='x', linestyle='--')
plt.tight_layout()
plt.savefig('python_gantt_chart.png')
print("Gantt chart saved as python_gantt_chart.")
```

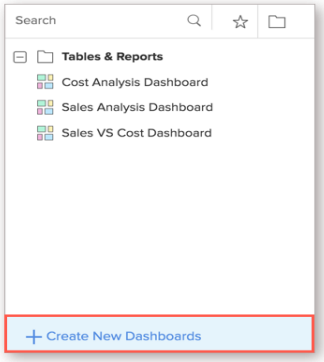


Practical 10

Aim: Show Data Visualization using Zoho

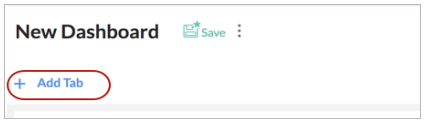
Step 1: Create a Dashboard

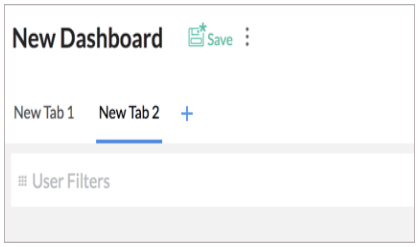
- Click **Create > New Dashboard** or use the **Dashboards menu > Create New Dashboards**.
- The **Dashboard Editor** opens for designing your dashboard.



Step 2: Add Tabs

- You can add up to **10 tabs** per dashboard.
- Steps:
    1. Open **Workspace > Create > New Dashboard**
    2. Click **Add Tab** to create a new tab
    3. Rename tabs via the **Action icon**
    4. Use **Duplicate** to copy a tab or **Remove** to delete

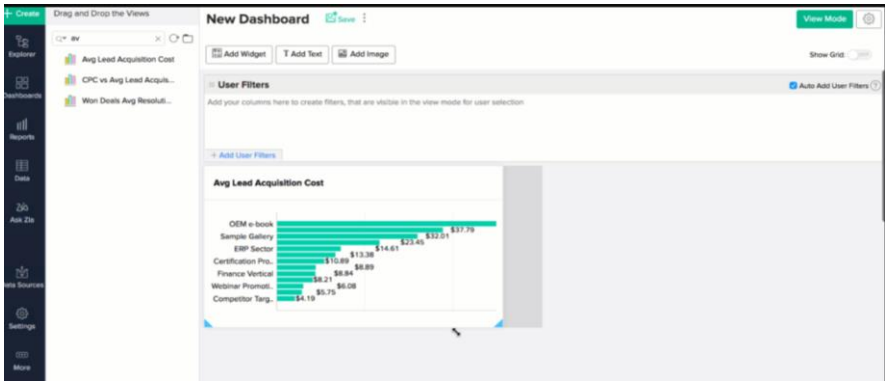




Step 3: Add Reports

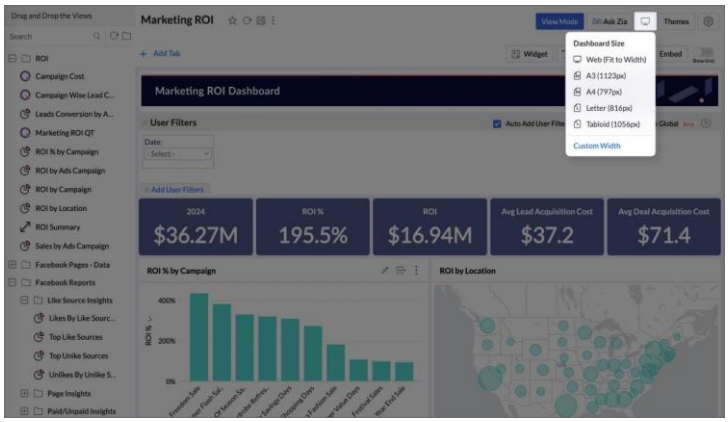
- Drag and drop reports from the **Workspace panel** into the **Dashboard Design Area**.

- Any number of reports can be added and arranged freely.



Step 4: Layout and Sizing

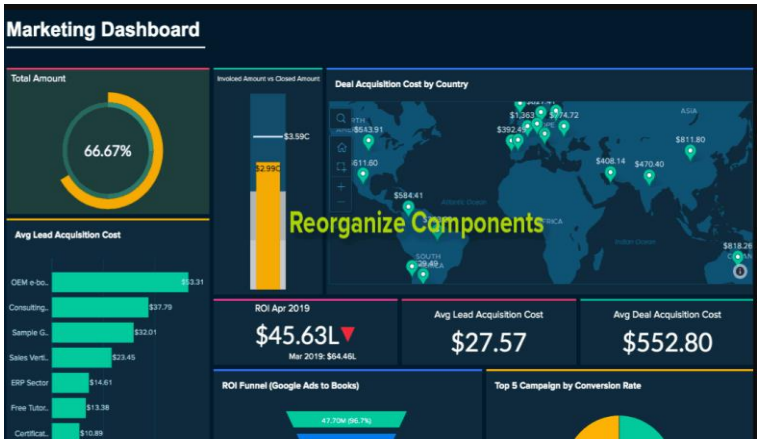
- **Drag and drop** components to organize your dashboard.
- Adjust **dashboard width** using the **Screen icon**:
  - Web (Fit to Width), A3, A4, Letter, Tabloid, or **Custom Width (797–4000 px)**
- **Reorganize multiple components** by selecting and dragging them together.



Step 5: Resize Components

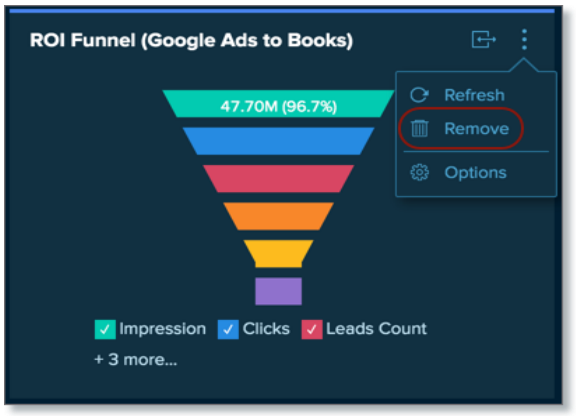
Options for resizing:

1. **Drag to Resize** – Adjust manually
2. **Copy Dimensions** – Apply width/height from one component to others
3. **Fit to Width** – Resize proportionally to fill available space
4. **Fill Space** – Expand components to fill dashboard area



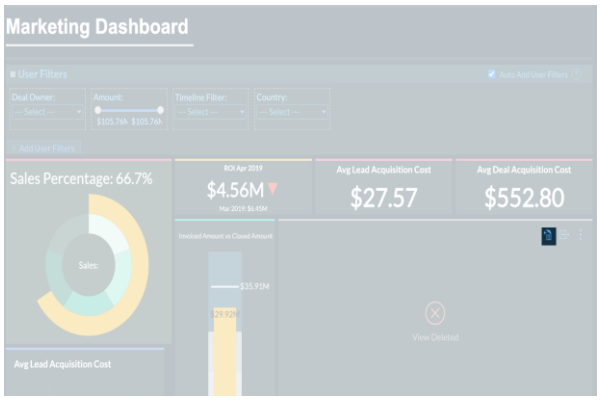
Step 6: Remove Components

- Remove individually via **action menu > Remove**
- Remove multiple components using the **Dashboard toolbar trash icon**



Step 7: Handle Deleted Views

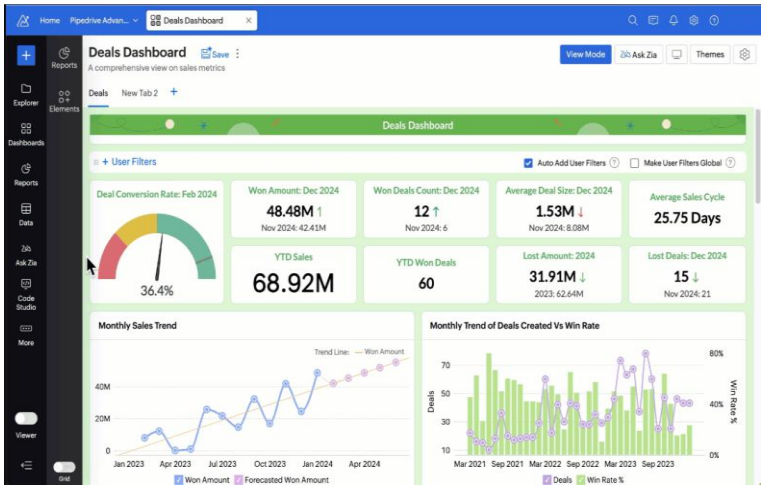
- **Restore**: Hover over deleted view → click **Restore icon**
- **Remove**: Click **Close icon** to remove the view permanently



Step 8: Copy/Paste Reports Across Tabs

- Reuse reports or KPI widgets using **Ctrl/Cmd + C** and **Ctrl/Cmd + V**

- Alignment is preserved when copying to another tab

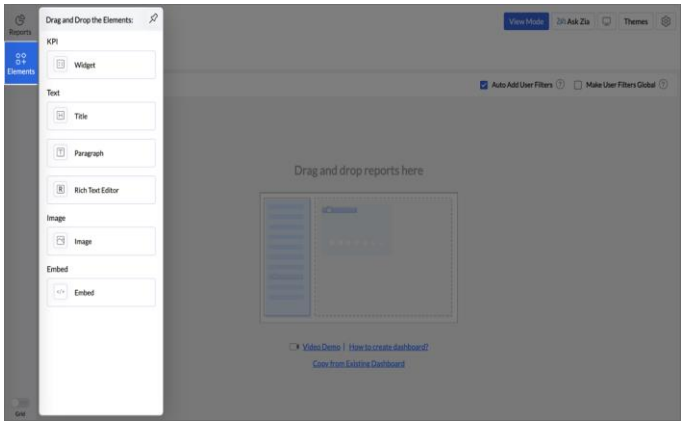


Step 9: Customize Dashboard

- Apply **themes** and **context-specific customizations**
- Enhance dashboards with additional elements for context

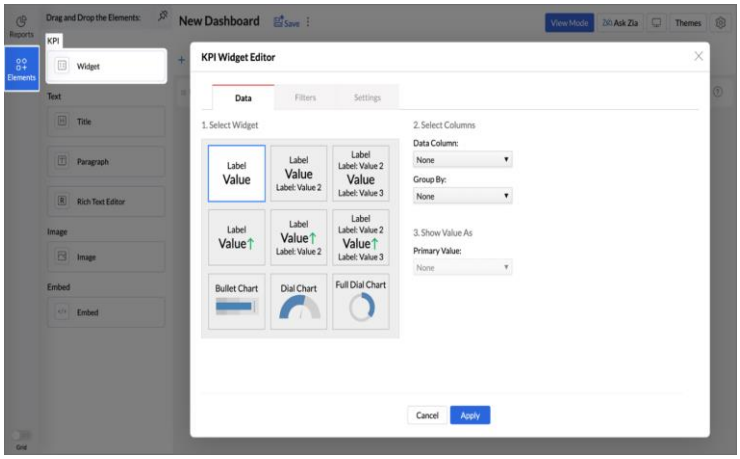
Step 10: Add Elements

- **Elements menu** includes: KPI, Text, Image, Embed
- Drag elements into the dashboard for enrichment



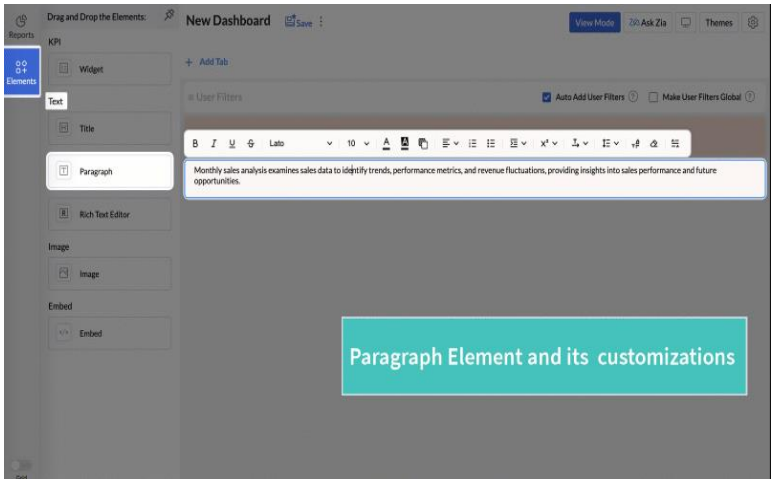
Step 11: Add KPIs

- KPI widgets highlight key metrics:
    1. **Single number widget** – Displays a metric
    2. **Chart type widget** – Displays trends, comparisons, or target progress



Step 12: Add Text

- Types of text elements:
    - **Title**: Headings with font and color options
    - **Paragraph**: Descriptions with formatting, indentation, line height
    - **Rich Text Editor**: Advanced formatting, tables, images, HTML



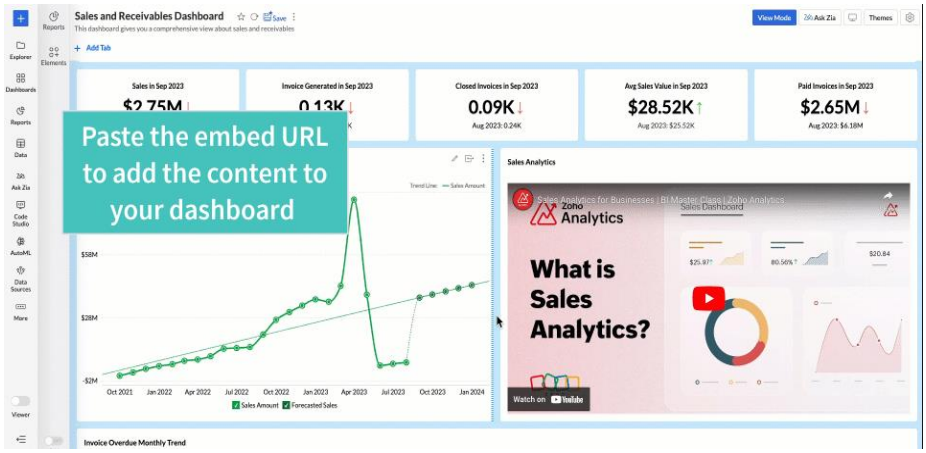
Step 13: Add Images

- Drag and drop image element

- Options: Upload, URL, My Library
- Customize image with:
    - Color, Brightness, Contrast, Transparency
    - Crop, Stroke, Drop Shadow, Rotate, Flip

Step 14: Add Embed Components

- Use **HTML iframe** to embed Google Maps, YouTube videos, or online forms
- Drag Embed element → Enter Title & URL → Apply



Step 15: Add User Filters

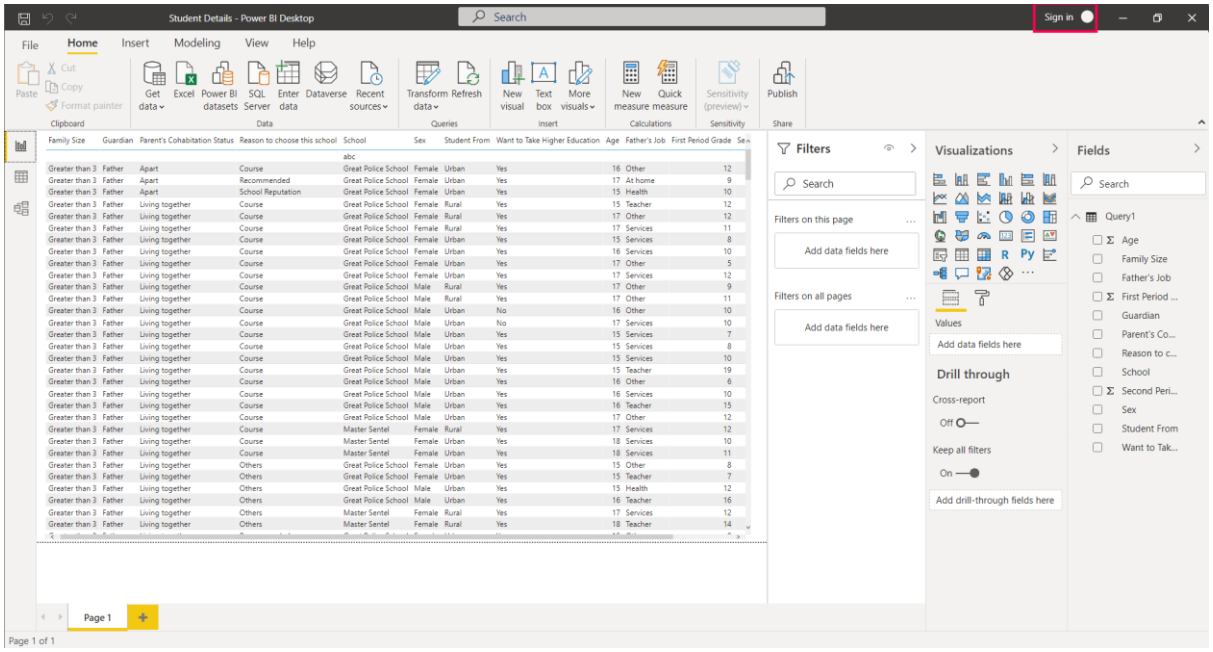
- Dynamic filters for dashboards or individual reports
- Users can filter data in **view mode** to focus on required insights

Practical 11

Aim: Publish visualised data on Cloud in PowerBi

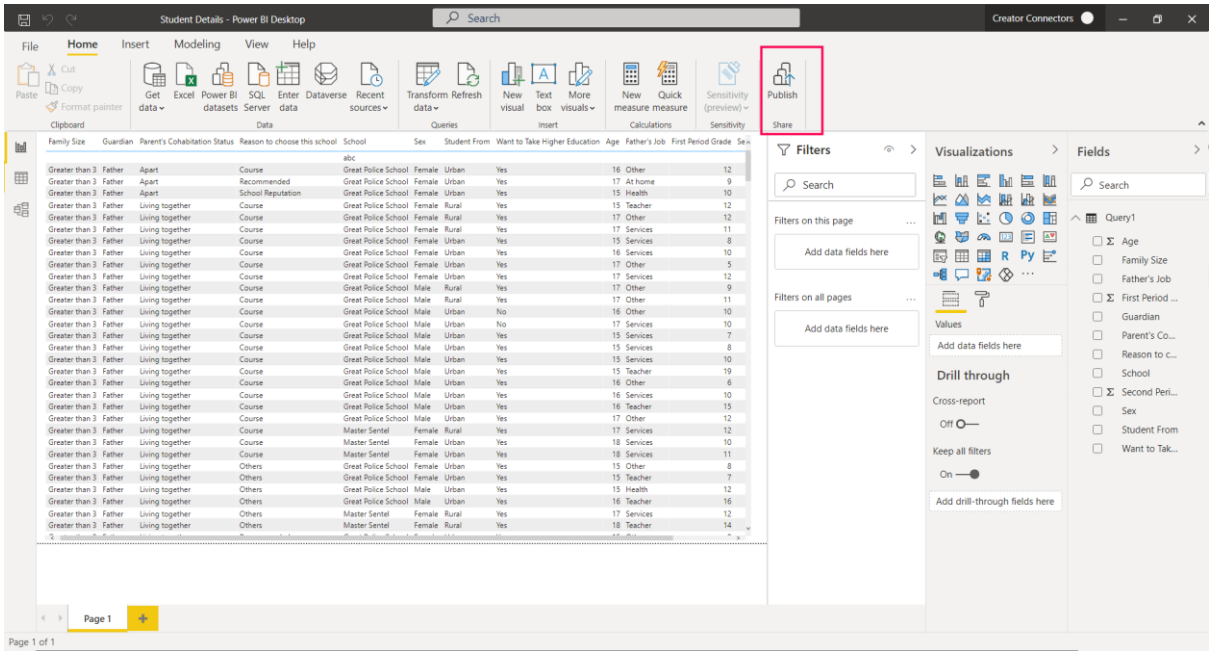
Step 1: Sign In

- Open **Power BI Desktop** and sign in with your **Microsoft account**.



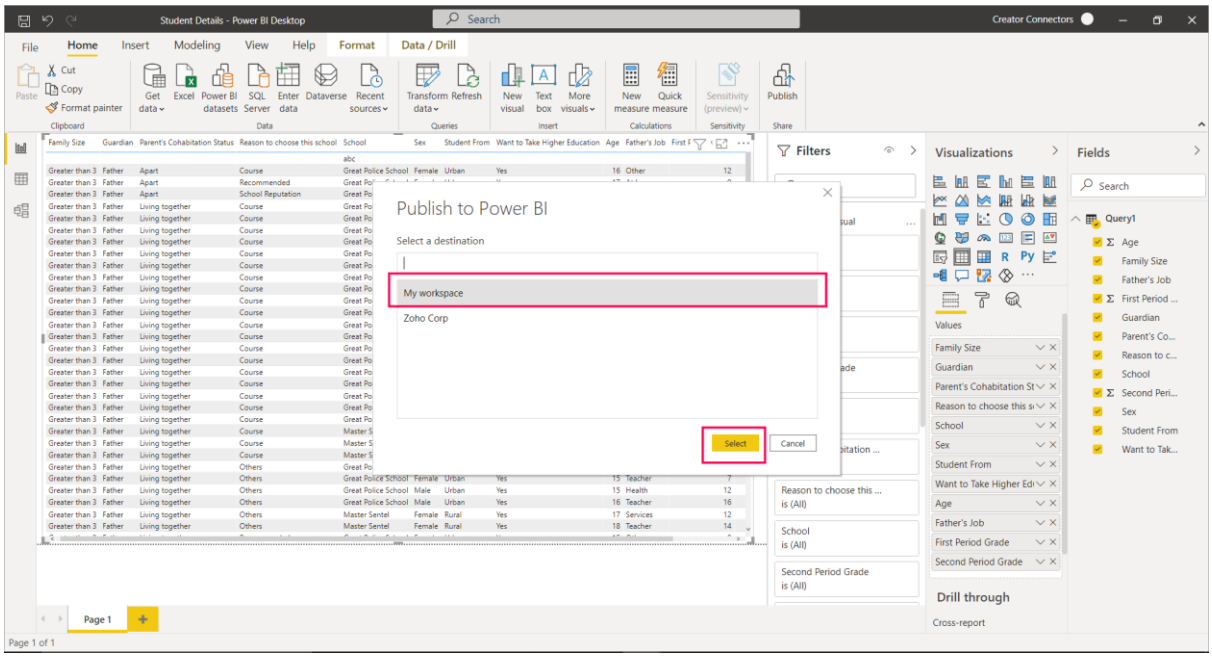
Step 2: Save and Publish

- Save your dataset locally.
- Click **Publish** on the Home ribbon.



Step 3: Select Workspace

- Choose the **Workspace** where you want to publish the dataset.
- Click **Select**.



Step 4: Confirm Upload

- The dataset will be **uploaded to Power BI cloud**.
- You can now access it online in **Power BI Service** to create reports and dashboards.