

| **Sr.No.** | **PracticalList** | **Date** | **Sign** |
| --- | --- | --- | --- |
| 1. | Implement Data Loading, Storage and File Formats.Readdataandstorethemintext format | 02/11/2024 |  |
| 2. | ImplementthecodetointeractwithWebAPIs and to perform web Scraping | 09/11/2024 |  |
| 3. | DemonstrateDataCleaningandPreparation | 16/11/2024 |  |
| 4. | ImplementDataWranglingonaDataset | 23/11/2024 |  |
| 5. | Demonstratethehandlingofmissingdataand string manipulation | 30/11/2024 |  |
| 6. | Createcommonchartswithtitle,labelsand descriptions using Tableau | 07/12/2024 |  |
| 7. | PerformsortingandfilteringusingTableau, create Visualization  publishitonTableauCloud | 14/12/2024 |  |
| 8. | PerformDataVisualizationusingPowerBI | 21/12/2024 |  |
| 9. | CreatereportsusingpowerBI. | 04/1/2025 |  |

**Practical1**

## Aim:ImplementDataLoading,StorageandFileFormats.Readdataandstorethemintextformat.

**Theory-** Implementingdataloading,storage,andfileformatsisa critical.Data loadinginvolvesthe process ofretrievingdatafromvarious sourcessuch asdatabases,APIs,or externalfiles.Choosing therightstorage solution is crucial to ensure data durability, accessibility, and scalability. File formats play a pivotal role in structuring and organizing data, influencing factors like storage efficiency and ease of processing. Reading data involves parsing and interpreting information from different file formats, be it CSV, JSON, XML, or others.

## Code:

defload\_data(file\_path):

try:

withopen(file\_path,'r')asfile:

data=file.readlines()#Readalllinesinthefile print("Data loaded successfully.")

returndata

exceptFileNotFoundError:

print("Filenotfound.Pleasecheckthefilepath.") return []

#Defineafunctiontoprocesstheloadeddata def process\_data(data):

processed\_data=[] for line in data:

processed\_line=line.strip().upper()#Example:Converteachlinetouppercase processed\_data.append(processed\_line)

returnprocessed\_data

#Defineafunctiontosavetheprocesseddatabacktoatext file def save\_data(file\_path, data):

withopen(file\_path,'w')asfile: for line in data:

file.write(line + '\n') print("Datasavedsuccessfully.")

#Mainfunctiontodemonstratetheprocess def main():

#Loaddatafromasamplefile input\_file = 'input.txt'

data=load\_data(input\_file)

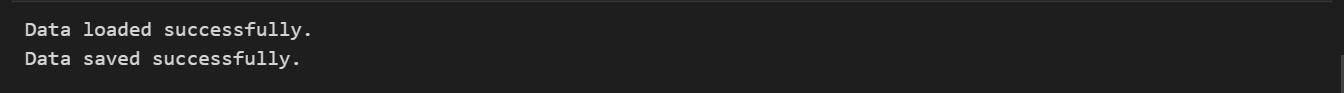
#Processdata if data:

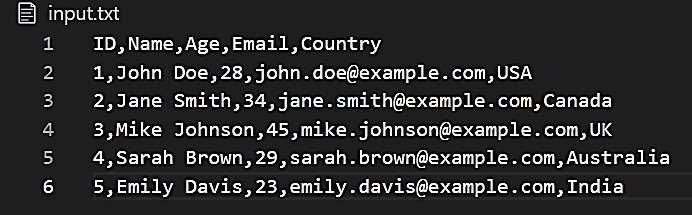
processed\_data=process\_data(data)

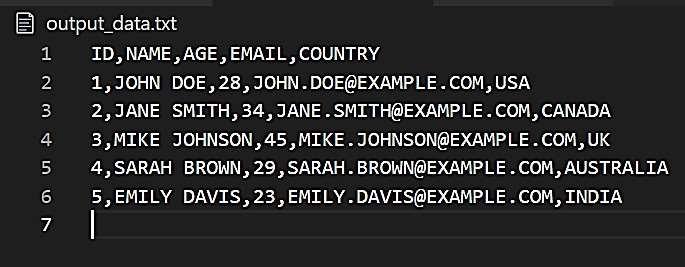
# Save processed data to a new file output\_file = 'output\_data.txt' save\_data(output\_file,processed\_data)

ifname =="main": main

**Output:**





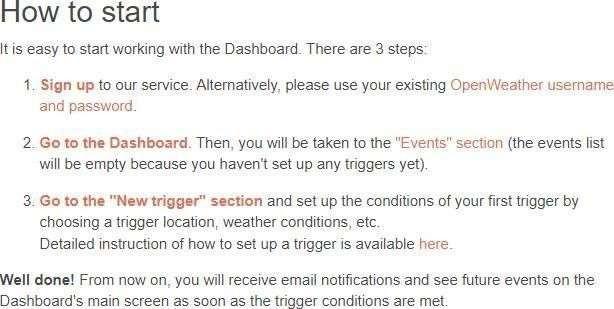


# Practical2

## Aim:ImplementthecodetointeractwithWebAPIsandtoperformwebscrapping.

**Theory-** ToimplementcodeforinteractingwithWebAPIsandperformingwebscraping,developerscanuse Python with libraries like requests and BeautifulSoup. The requests library facilitates sending HTTP requests to Web APIs, while BeautifulSoup simplifies the parsing of HTML for efficient web scraping. By combining thesetools, developerscan createscriptstofetchdata fromAPIsandextractvaluable informationfromweb pages. It's important to prioritize ethical considerations and adhere to website terms of service to ensure responsible and legal data acquisition.

## PartA:GetAPIAccessforWeather Information



**Code:**

importrequests

frombs4importBeautifulSoup

#OpenWeatherMapAPI functionforinteractingwithawebAPI usinglatitudeandlongitudedef get\_weather\_data(lat,lon):

#ReplacewithyourAPIkey

api\_key='35879ab0c5fbcd67b3b649f3119fa468'

base\_url='[http://api.openweathermap.org/data/2.5/weather'](http://api.openweathermap.org/data/2.5/weather%27) params = {

'lat':lat,

'lon':lon, 'appid':

api\_key,

'units':'metric'#GettemperatureinCelsius

}

response=requests.get(base\_url,params=params)if response.status\_code == 200:

data=response.json()

print(f"Weatherforcoordinates({lat},{lon}):") print(f"Temperature: {data['main']['temp']}°C")

print(f"Weather:{data['weather'][0]['description']}") return data

else:

print("LocationnotfoundorinvalidAPIkey.") return None

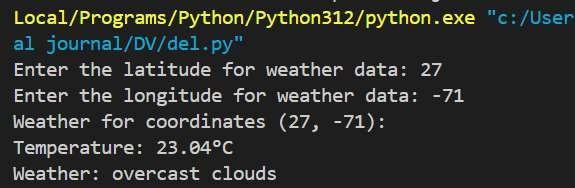
#Mainfunctiontodemonstratebothfunctionalitiesdef main():

#Part1:InteractingwiththeOpenWeatherMapAPIusingcoordinateslat= input("Enter the latitude for weather data: ")

lon=input("Enterthelongitudeforweatherdata:") get\_weather\_data(lat,lon)

ifname =="main": main()

**Output:**



## PartB:PerformWebScraping

* Installtheselibraries:*pipinstallbeautifulsoup4requestslxml*

**Code:**

#Simplewebscrapingfunctiontofetchheadlinesfromanewswebsite def scrape\_news\_headlines():

url='https://[www.bbc.com/news'](http://www.bbc.com/news%27)#Examplenewswebsite response = requests.get(url)

ifresponse.status\_code==200:

soup=BeautifulSoup(response.text,'html.parser')

headlines=soup.find\_all('h3')#Graballtheh3tagswhichusuallyholdheadlines

print("\nTopNewsHeadlines:")

fori,headlineinenumerate(headlines[:5],1):#Limitingtothefirst5headlinesprint(f"{i}.

{headline.get\_text().strip()}") return headlines[:5]

else:

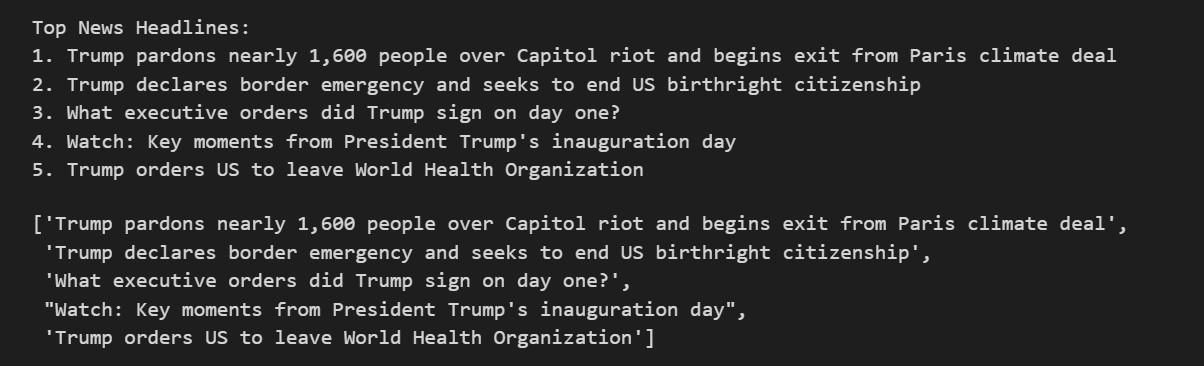
print("Failedtoretrievenewsheadlines.") return None

#Mainfunctiontodemonstratebothfunctionalitiesdef main():

#Part2:Webscrapingtogetnewsheadlines scrape\_news\_headlines()

ifname =="main": main()

**Output:**



# Practical3

## Aim:DemonstrateDataCleaningand Preparation.

**Theory -** Data cleaning and preparation are essential for converting raw datasets into reliable sources for analysis.Tasks includehandlingduplicates,correctingerrors,andfillingmissingvalues.Demonstratingthese processes often involves using programming languages like Python or R along with libraries such as Pandas or tidyverse. EDA techniques can also be showcased for visual inspection. Through effective data cleaning and preparation, analysts ensure the quality of their datasets, improving the reliability and significance of subsequent analysis and modeling efforts.

### Code

*importpandasaspd importnumpyas np*

*#Step1:Loadthedataset*

*file\_path="data\_cleaning\_demo.csv"#Replacewithyourfilepathifneeded df = pd.read\_csv(file\_path)*

*print("OriginalData:") print(df)*

*#Step2: Handlemissingvalues*

*df['Age'].fillna(df['Age'].mean(), inplace=True) # Fill missing Age with mean df['Email'].fillna("*[*unknown@example.com*](mailto:unknown@example.com)*",inplace=True)#FillmissingEmail df.dropna(subset=['Name'], inplace=True) # Drop rows with missing Name*

*# Step 3: Remove duplicates df.drop\_duplicates(inplace=True)*

*#Step4: Convertdatatypes*

*df['Joining\_Date']=pd.to\_datetime(df['Joining\_Date'],errors='coerce')#Converttodatetime*

*#Step5: Rename columns*

*df.rename(columns={'Salary':'Annual\_Salary'},inplace=True)*

*#Step6:Filterdata(e.g.,keeprowswithAge>20) df = df[df['Age'] > 20]*

*# Step 7: Export cleaned data output\_file = "cleaned\_data.csv" df.to\_csv(output\_file,index=False)*

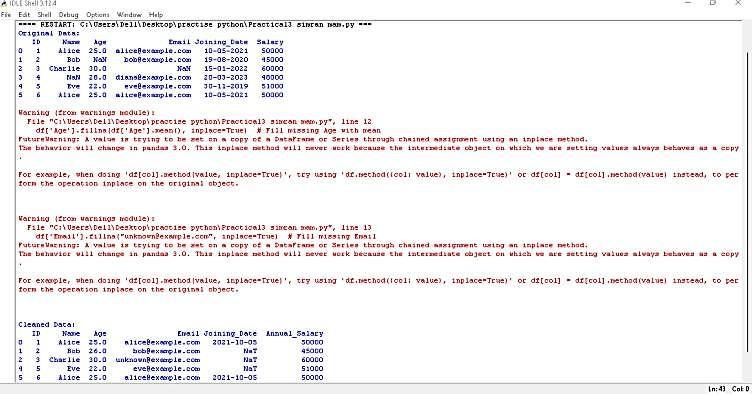
*print("\nCleanedData:") print(df)*

*print(f"\nCleaneddatasavedto{output\_file}")*

threshold=3

outliers=(abs(z\_scores)>threshold).any(axis=1) df\_outliers = df\_outliers[~outliers]

print("\nTask7:HandlingOutliersusingZ-score") print("Rows with outliers removed:") print(df\_outliers)

***Output***

# Practical4

## Aim:ImplementData wranglingonthedataset

**Theory -** Data cleaning and preparation are essential for converting raw datasets into reliable sources for analysis.Tasks includehandlingduplicates,correctingerrors,andfillingmissingvalues.Demonstratingthese processes often involves using programming languages like Python or R along with libraries such as Pandas or tidyverse. EDA techniques can also be showcased for visual inspection. Through effective data cleaning and preparation, analysts ensure the quality of their datasets, improving the reliability and significance of subsequent analysis and modeling efforts.

### Code

*import* pandas *as*pd *import* numpy*as* np *import* os

data={

"Transaction\_ID": [1, 2, 3, 4, 5], "Product":["A","B","C","A","B"], "Quantity": [10, -5, 20, 15, 8],

"Price":[15.0,20.0,25.0,None,30.0],

"Date":["2023-01-01","2023-01-02","2023-01-03","2023-01-04","2023-01-05"],

}

df=pd.DataFrame(data)

print(*f*"Task1:LoadedDataset\n{df.head()}")

missing\_cols=df.columns[df.isnull().any()] df.fillna(0, inplace=True)

print(*f*"\nTask2:MissingValues- AfterHandling\n{df}")

numeric\_cols=df.select\_dtypes(include=[np.number]).columns df[numeric\_cols] = df[numeric\_cols].abs()

print(*f*"\nTask3:NegativeValues-AfterHandling\n{df}")

df["Date"]= pd.to\_datetime(df["Date"])

print(*f*"\nTask4:Converted'Date'columntoDateTime format\n{df}")

df["Day"]= df["Date"].dt.day

df["Month"]=df["Date"].dt.month

df["Year"]= df["Date"].dt.year

print(*f*"\nTask5:Extractedday, month,andyearinformationfrom'Date'\n{df}")

df["Total Sales"] = df["Quantity"] \* df["Price"] print(*f*"\nTask6:Calculated'TotalSales'\n{df}")

grouped\_df=(df.groupby("Product").agg({"Quantity":"sum","TotalSales":"sum"}).reset\_index()) print(*f*"\nTask 7: Grouped and Aggregated Data by 'Product'\n {grouped\_df}")

wrangled\_file\_path=os.path.abspath("wrangled\_dataset.csv") df.to\_csv(wrangled\_file\_path, index=False)

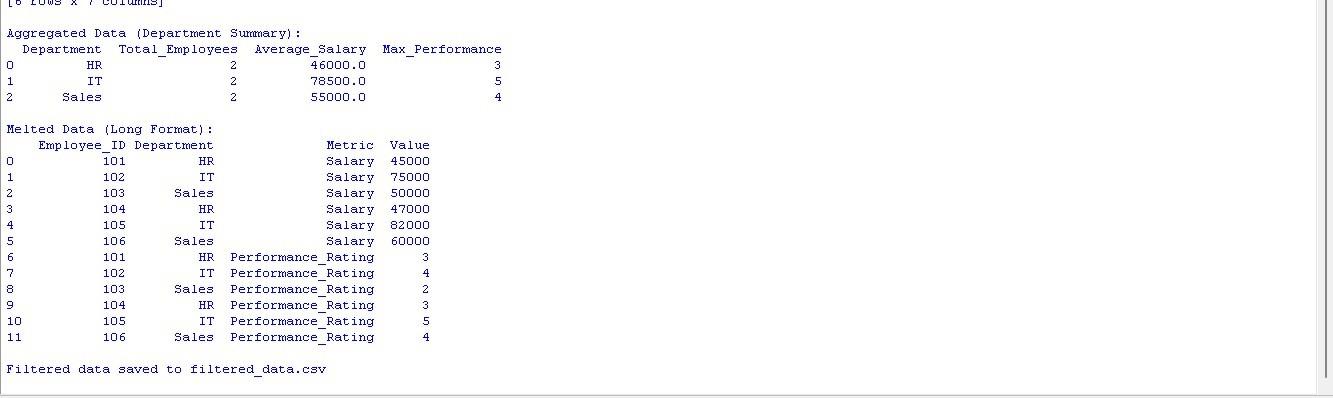
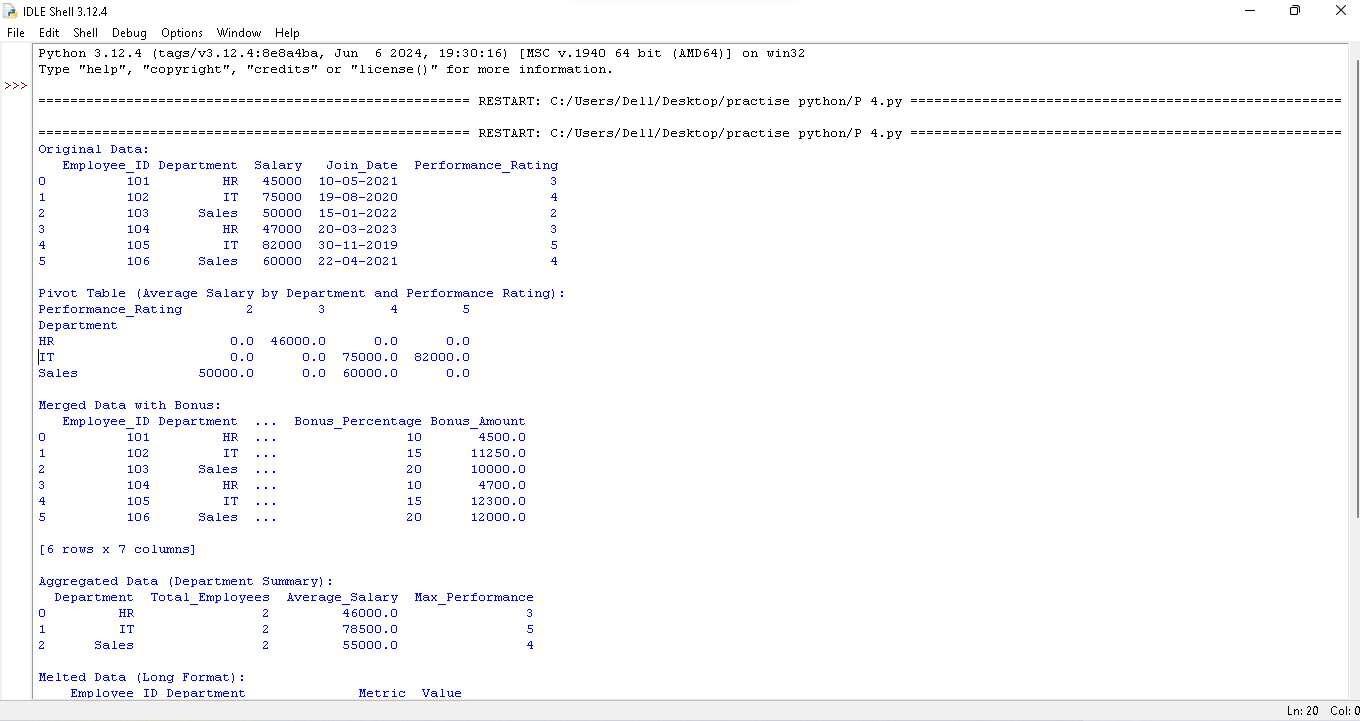
print("\nTask8:SavedthewrangledDatasetto",wrangled\_file\_path)

transformed\_file\_path = os.path.abspath("transformed\_dataset.csv")

df.to\_csv(transformed\_file\_path,index=False)

print("\nTask9:SavedtheTransformedDatasetto",transformed\_file\_path) transposed\_df = df.transpose()

transposed\_file\_path=os.path.abspath("transposed\_dataset.csv")



# Practical5

**Aim:**Demonstratethehandlingofmissingdata andstring manipulation

**Theory -** In data analysis, adept handling of missing data involves strategies like imputation or deletion, ensuringtheintegrityoftheanalysis.Stringmanipulationiscrucialforcleaningandtransformingtextualdata by tasks such as removing unwanted characters or extracting meaningful information. Proficiency in these techniques enhances the reliability of data-driven insights and supports more accurate decision-making processes.

## Code:

*importpandasaspd importnumpyas np*

*#Step1:Loadthedataset*

*file\_path="missing\_data\_and\_strings\_demo.csv"#Replacewithyourfilepathifneeded df = pd.read\_csv(file\_path)*

*print("OriginalData:") print(df)*

*#Step2: Handlemissingdata*

*# Fill missing 'Department' with 'Unknown' df['Department'].fillna('Unknown',inplace=True)*

*# Drop rows with missing 'Name' df.dropna(subset=['Name'],inplace=True)*

*# Fill missing 'Email' with a placeholder df['Email'].fillna("*[*noemail@example.com*](mailto:noemail@example.com)*",inplace=True)*

*#Step3: StringManipulation*

*#Stripwhitespacefrom'Name'column df['Name'] = df['Name'].str.strip()*

*# Convert 'Name' to title case df['Name']=df['Name'].str.title()*

*#Extractdomain from'Email'*

*df['Email\_Domain']=df['Email'].str.extract(r'@([a-zA-Z0-9.-]+)')*

*#Createanewcolumnwithdepartment initials*

*df['Department\_Initials']=df['Department'].str.extract(r'(\b\w)')#Firstletterofeach word*

*# Replace missing 'Joining\_Date' with today's date df['Joining\_Date'].fillna(pd.Timestamp.today().strftime('%Y-%m-%d'),inplace=True)*

*#Step4: Exportcleaneddata*

*output\_file="cleaned\_data\_with\_strings.csv" df.to\_csv(output\_file, index=False)*

*print("\nCleanedData:") print(df)*

*print(f"\nCleaneddatasavedto{output\_file}")*



# Practical6

**Aim:**Createcommonchartswithtitle,labelsanddescriptionsusing Tableau

**Theory -** Tableau simplifies the creation of common charts by offering an intuitive interface for adding titles, labels, and descriptions. Whether it's a bar chart, line graph, or pie chart, users can easily customize visual elements for clarity. The platform's drag-and-drop functionality and interactive features allow seamless data exploration. Tableauenables professionalstocommunicateinsightseffectively, makingdata-driven decision- making accessible and efficient.

Step1:PrepareYourExcelData

1. CreateanExcelsheetwiththefollowingdata(orsimilar):

Category Sub-CategorySalesProfitRegion Furniture Chairs 3000 500 East

Technology Laptops 70001500West

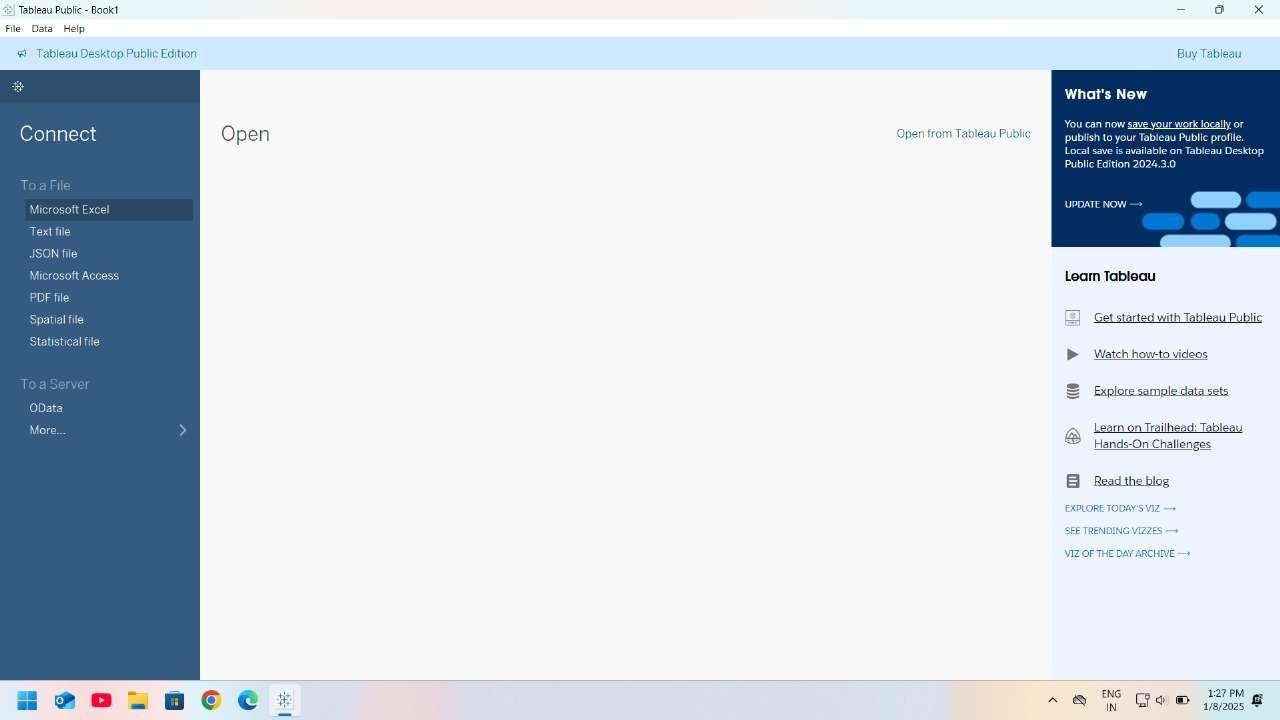
OfficeSuppliesPaper 2000300 South

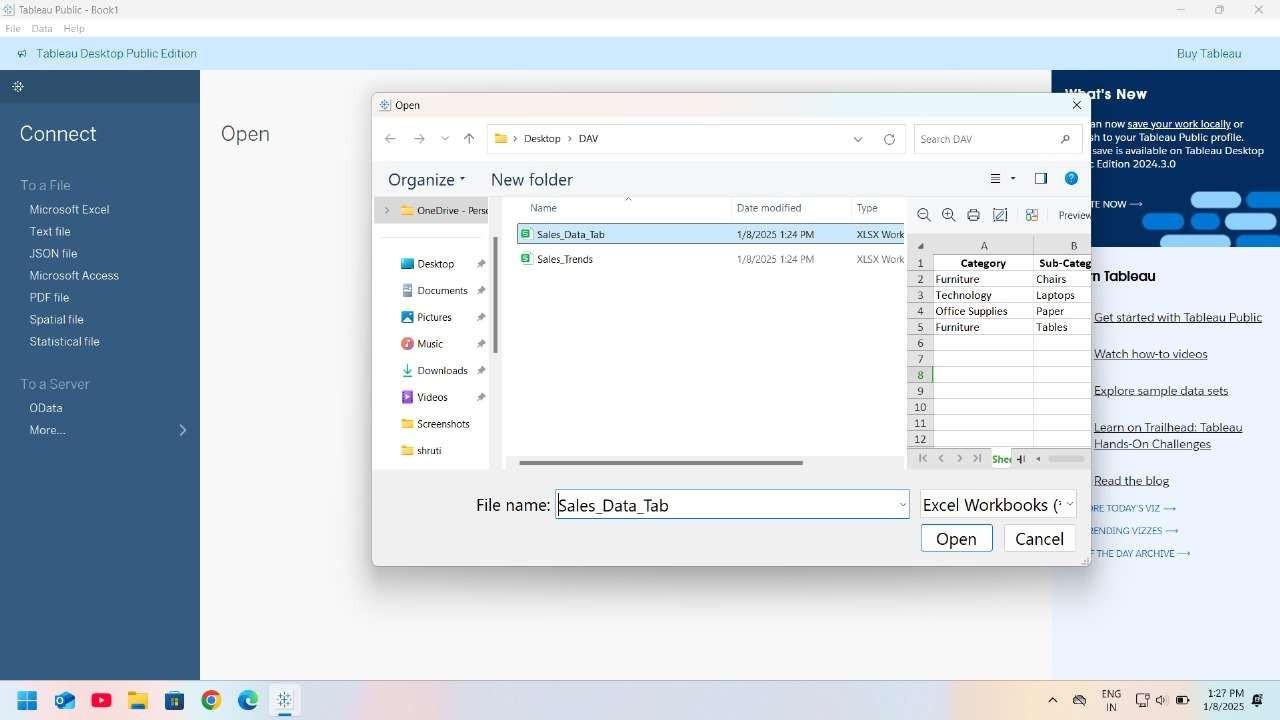
Furniture Tables 4000600 North

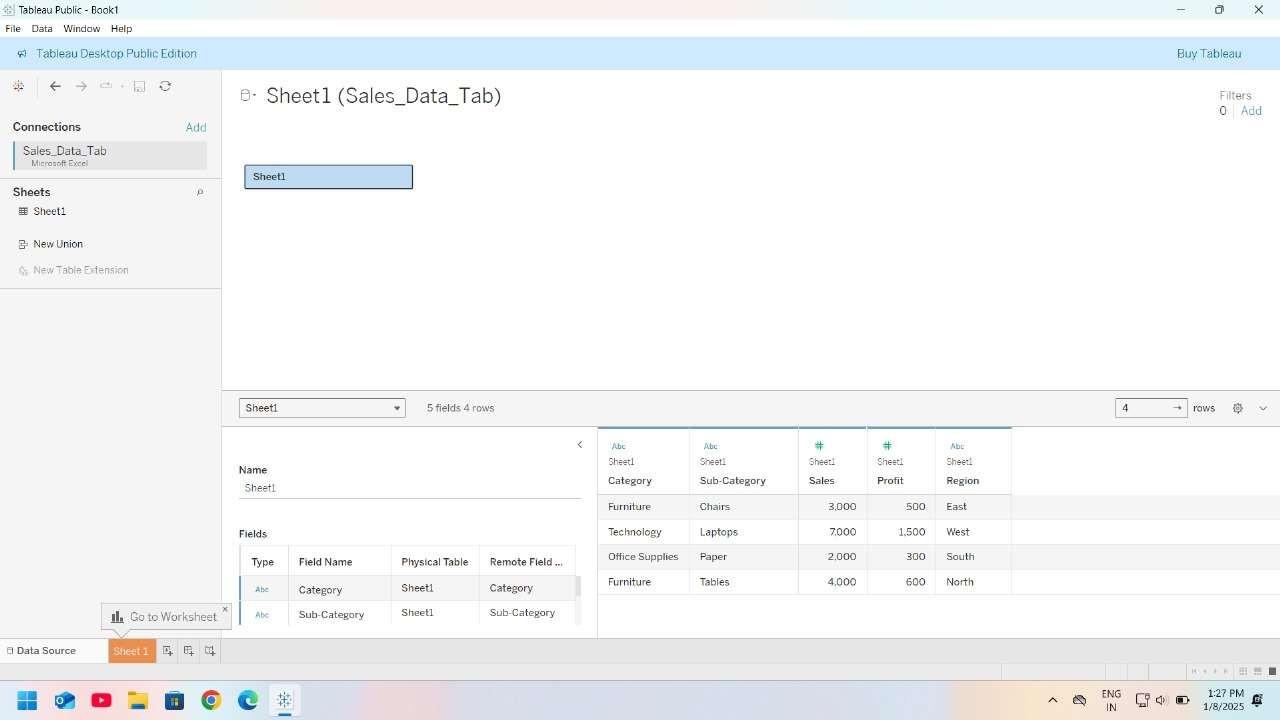
1. Savethefileas Sales\_Data.xlsx.

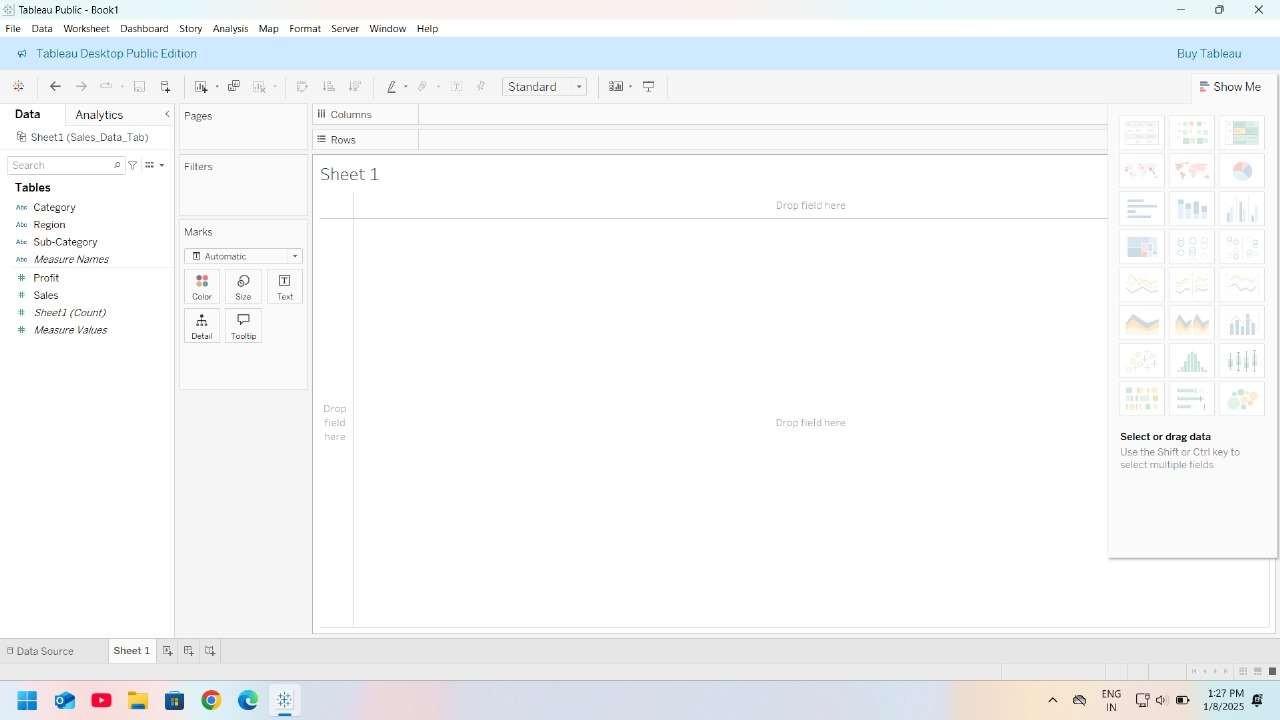
Step2:ConnectExcelDatatoTableau

1. OpenTableauDesktoporTableauPublic.
2. Onthestartpage,underConnect,selectMicrosoftExcel.
3. BrowseandopenSales\_Data.xlsx.
4. Dragthesheet(e.g.,"Sheet1")tothecanvas.









Step3:CreateBasicCharts

1. BarChart

Purpose:CompareSalesacrossCategories.

* + DragCategorytoRowsandSalestoColumns.
  + ChangethecharttypetoBarbyclickingontheShowMe paneandselecting Bar Chart.
  + Addatitle:Double-clickthecharttitleandnameit"SalesbyCategory".
  + Addlabels:Right-clickonSalesintheMarkscard,selectShowMarkLabels.
  + Add description: Add a Text object above or below the chart, describing its purpose.Example:"Thisbarchartshowsthetotalsalesacrossdifferentproduct categories."

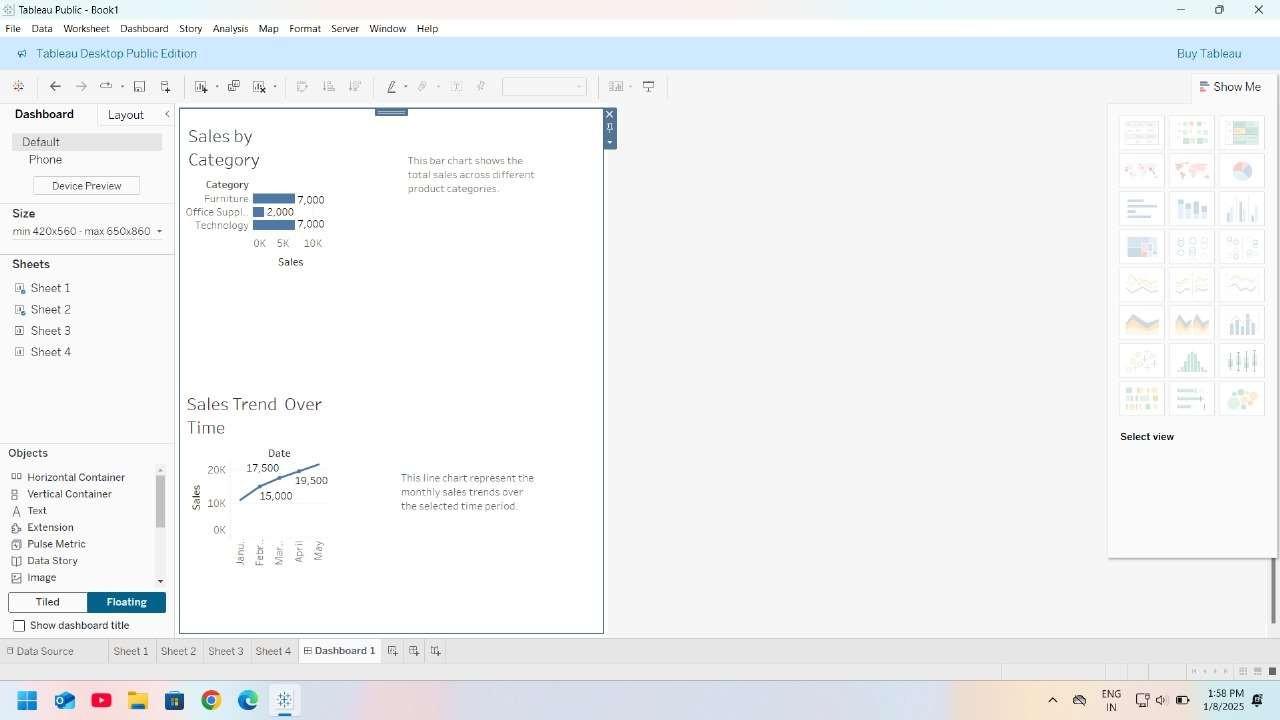










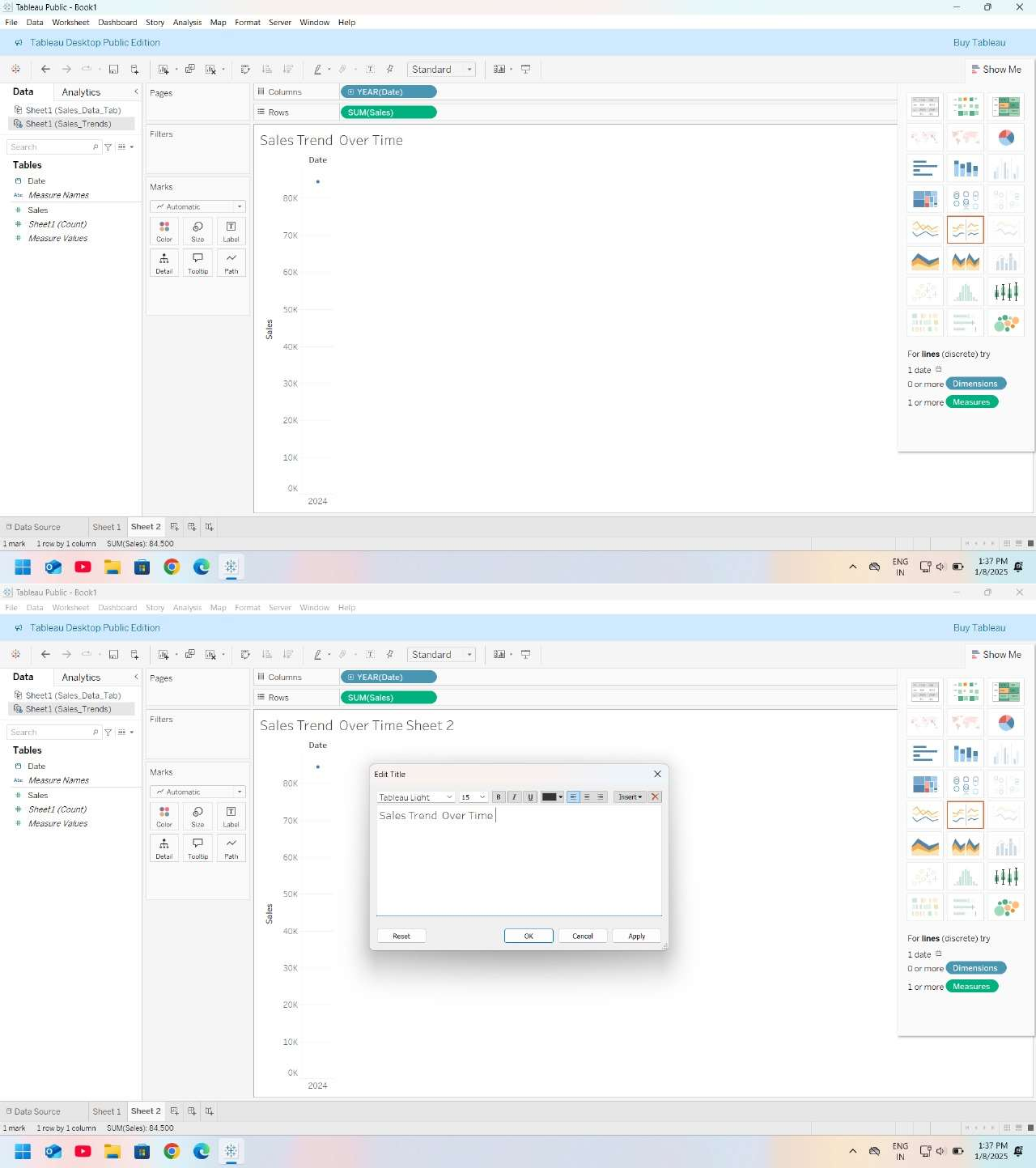


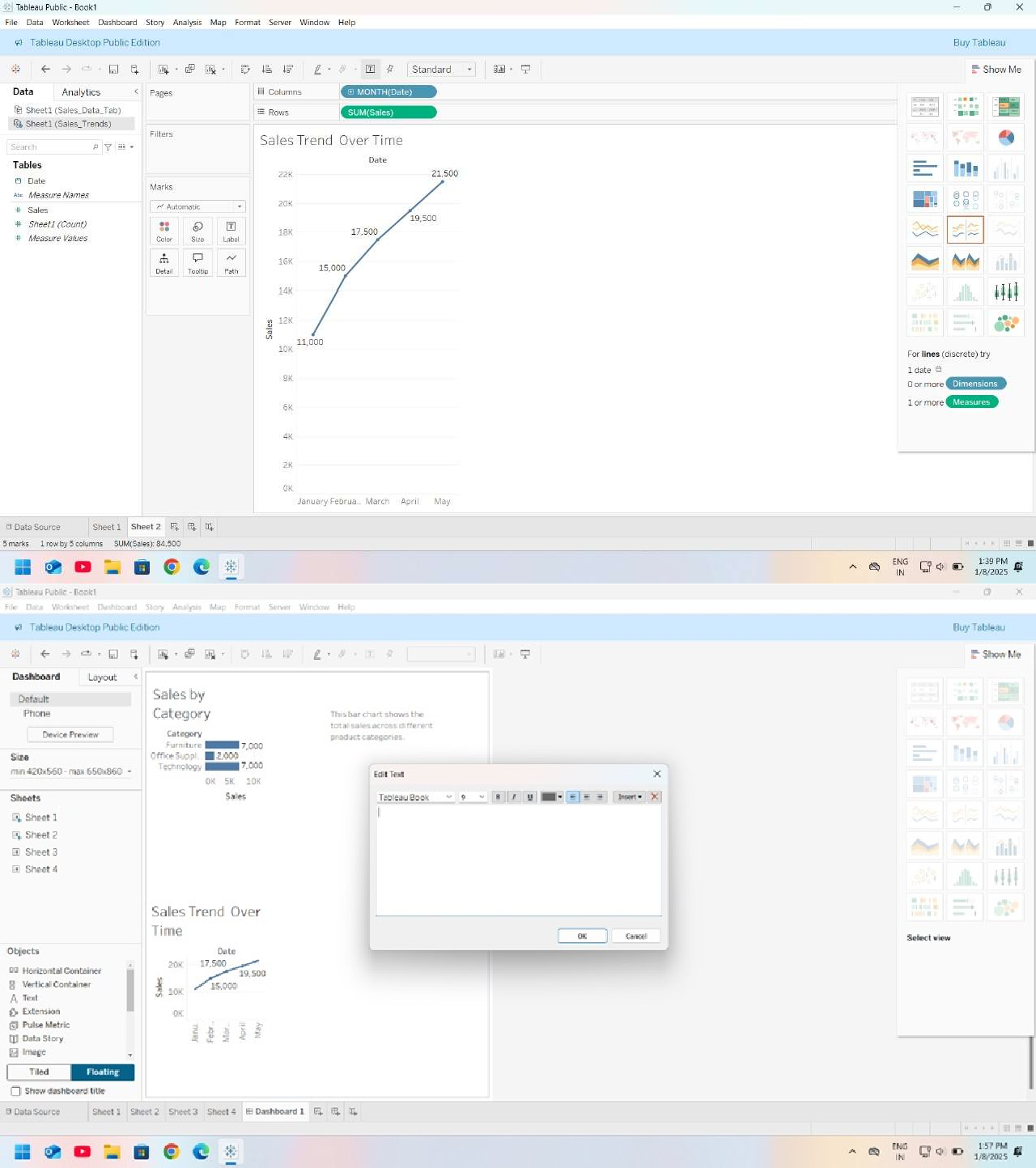
1. LineChart

Purpose:Showtrendsovertime.

(ModifythedatasettoincludeaDatecolumnforthisexample.)

* + DragDatetoColumnsandSalestoRows.
  + ChangethecharttypetoLine.
  + Title:"SalesTrendsOverTime".
  + Description:"This linechartrepresentsthemonthlysalestrendoverthe selected time period."



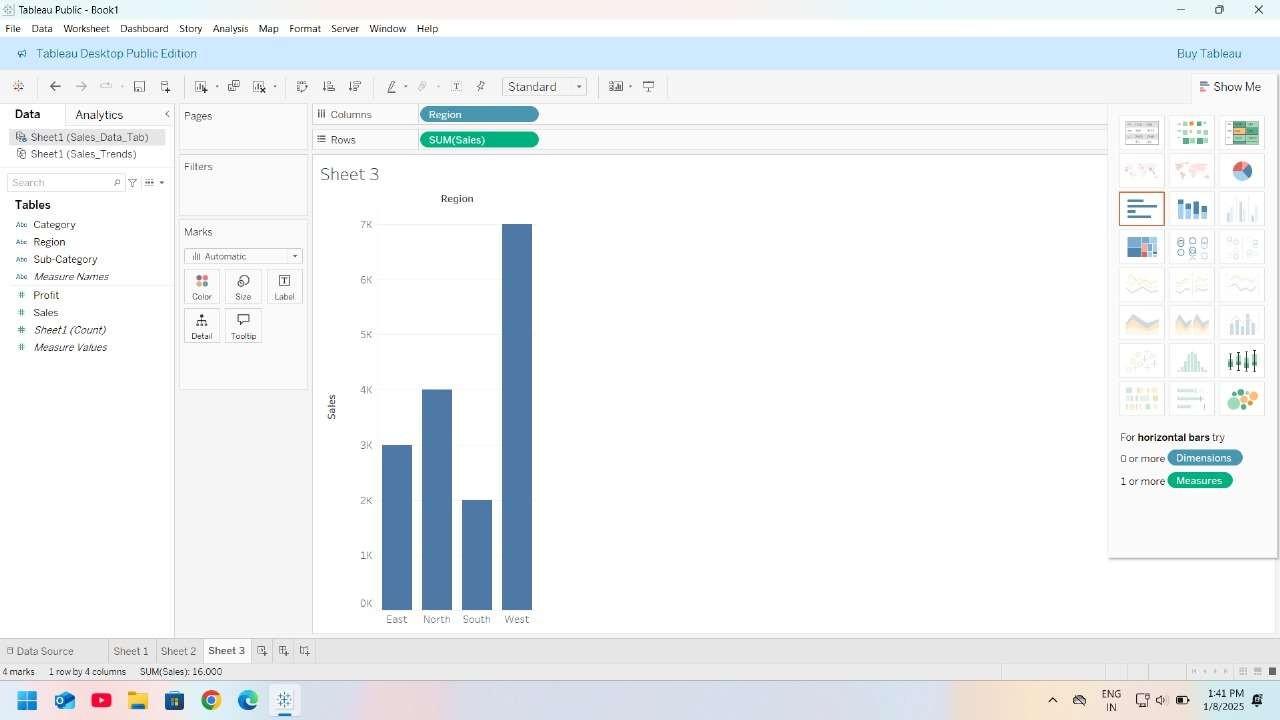


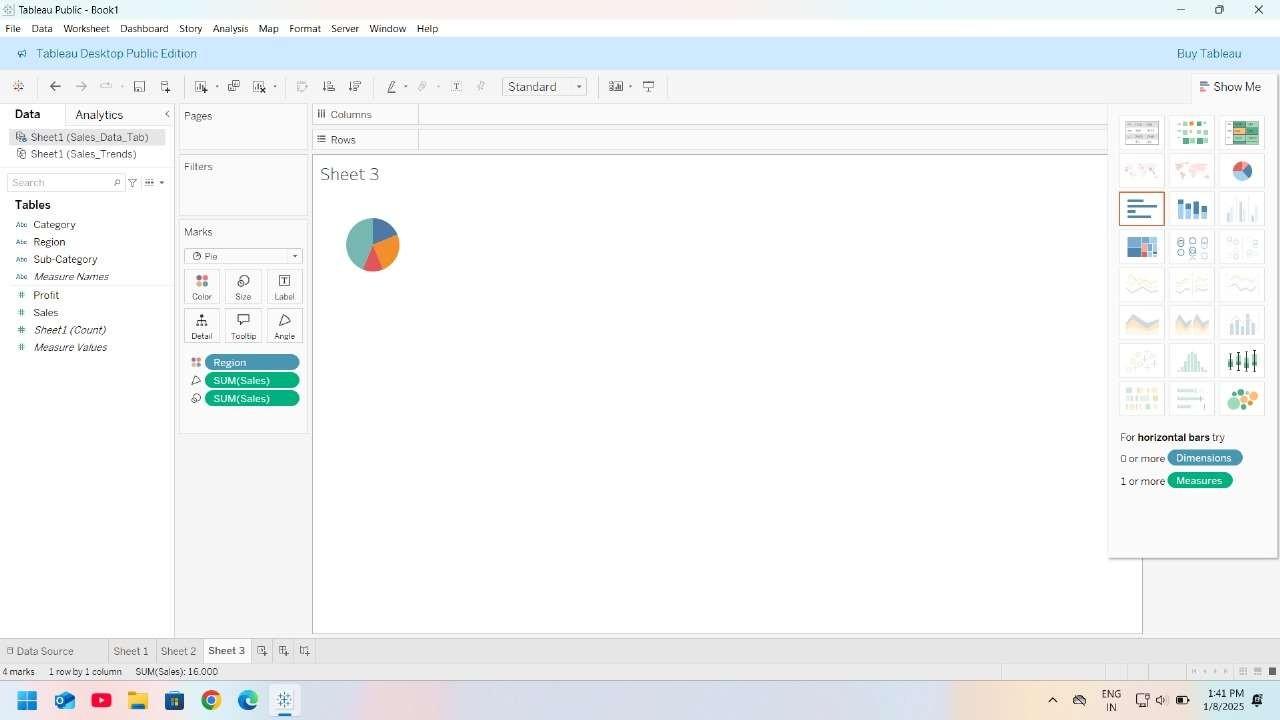


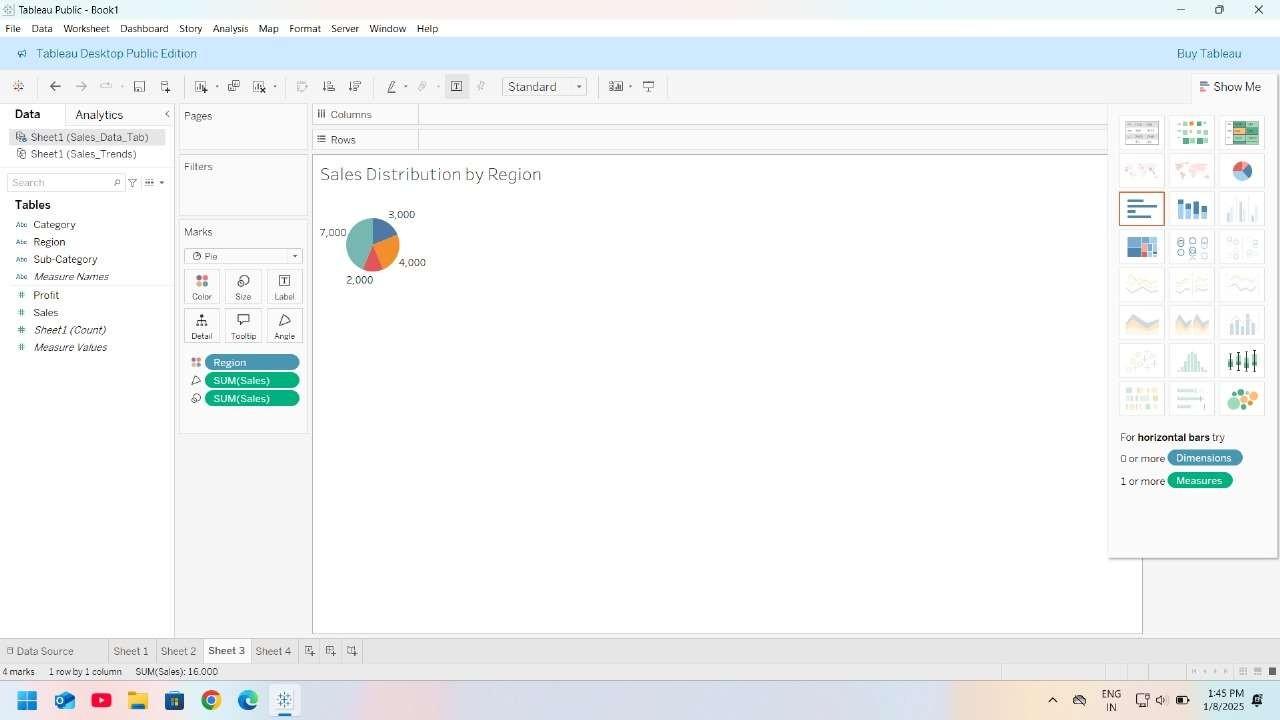
1. PieChart

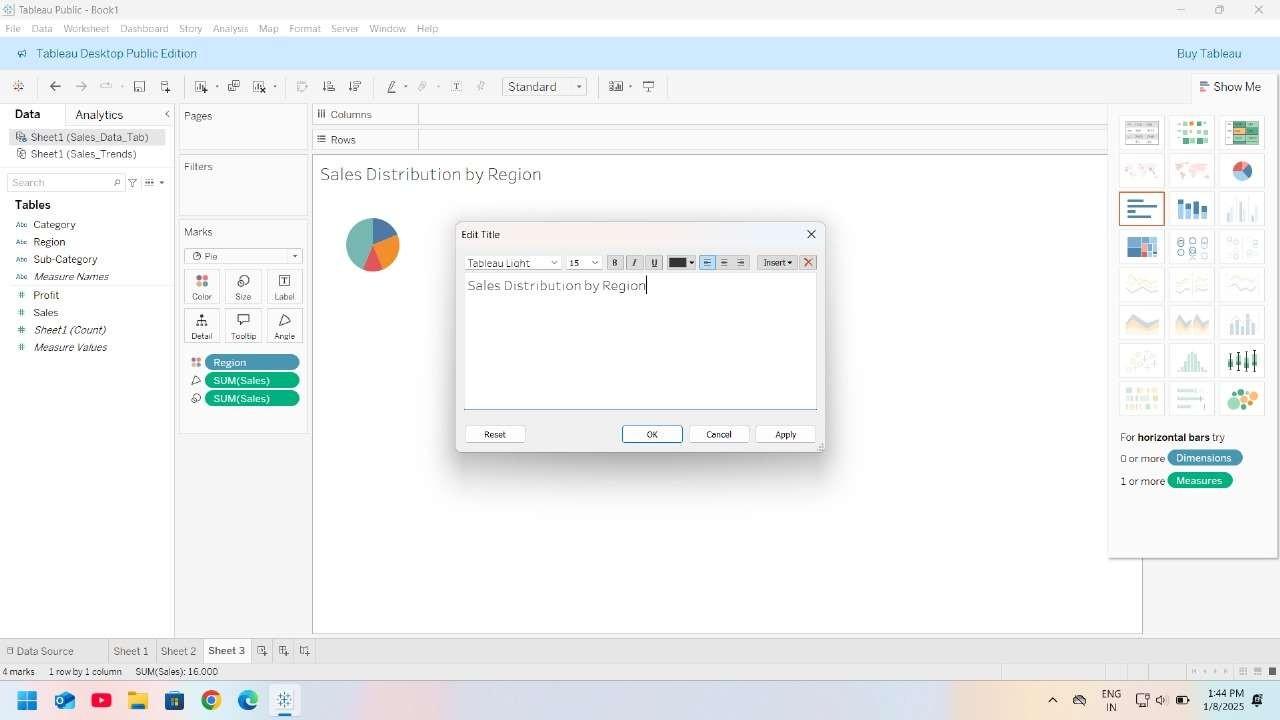
Purpose:Showsalesdistributionbyregion.

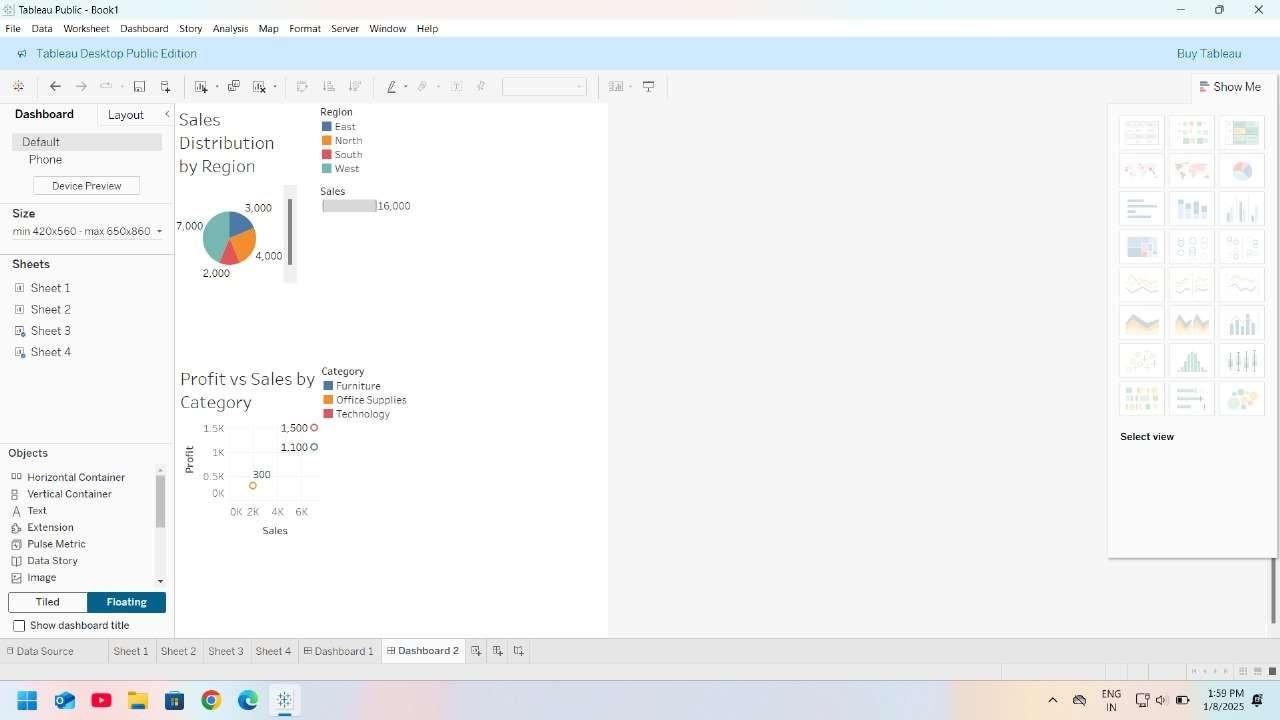
* + DragRegiontoColumnsandSalestoRows.
  + ClickontheShowMepaneandselectPieChart.
  + DragSalestoAngleandRegiontoColorontheMarkscard.
  + Title:"SalesDistributionbyRegion".
  + Description:"This pie chartdepictstheproportionofsalescontributedbyeach region."

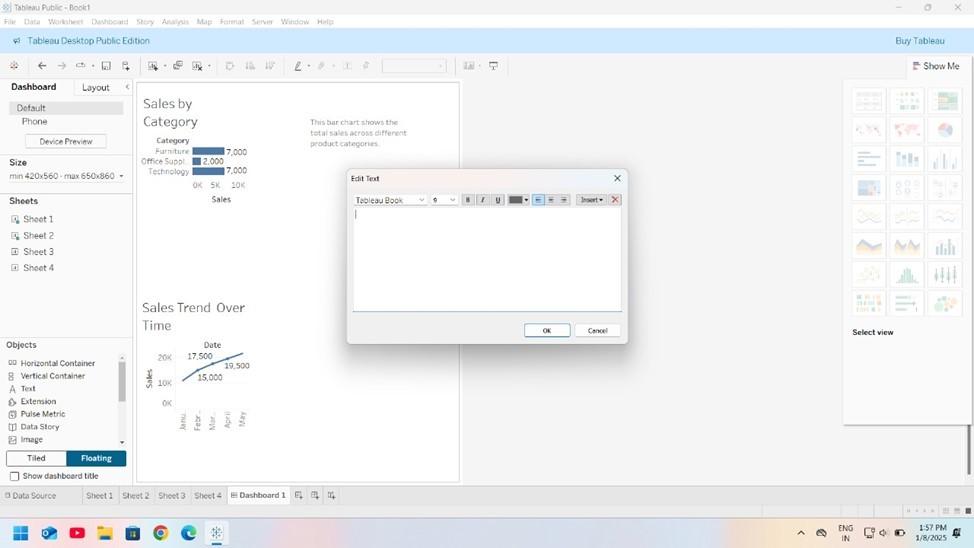


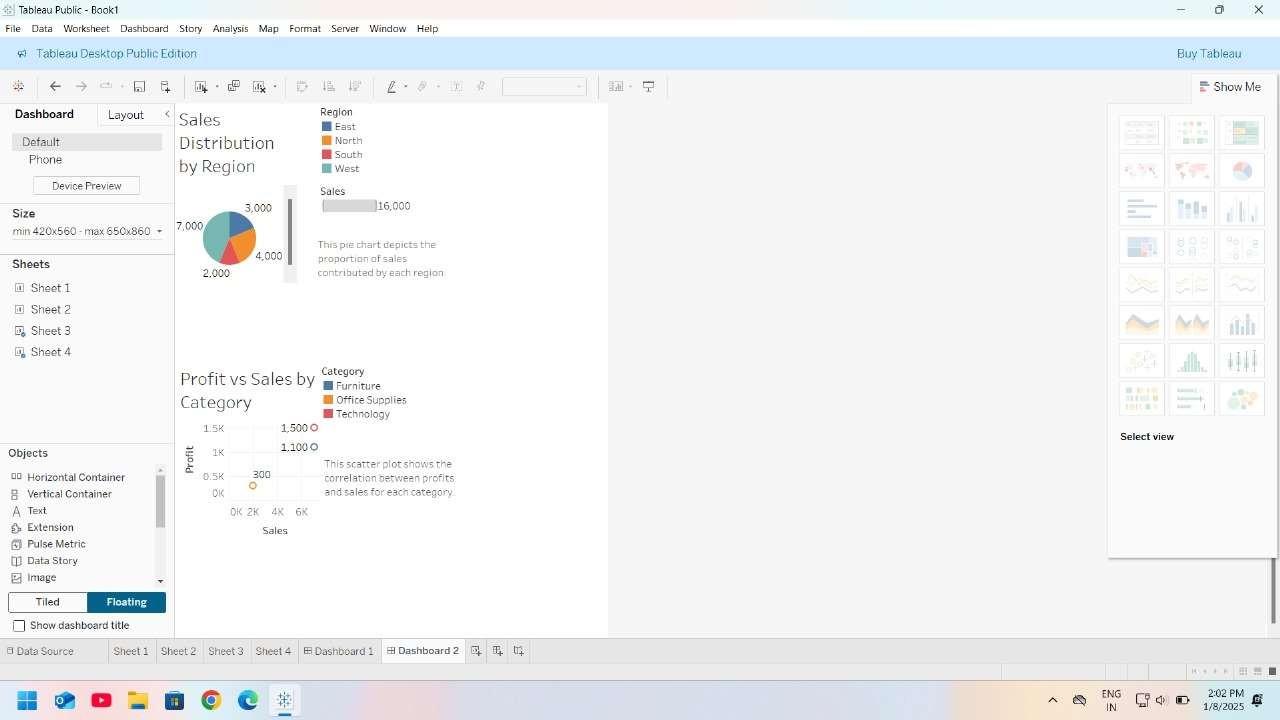








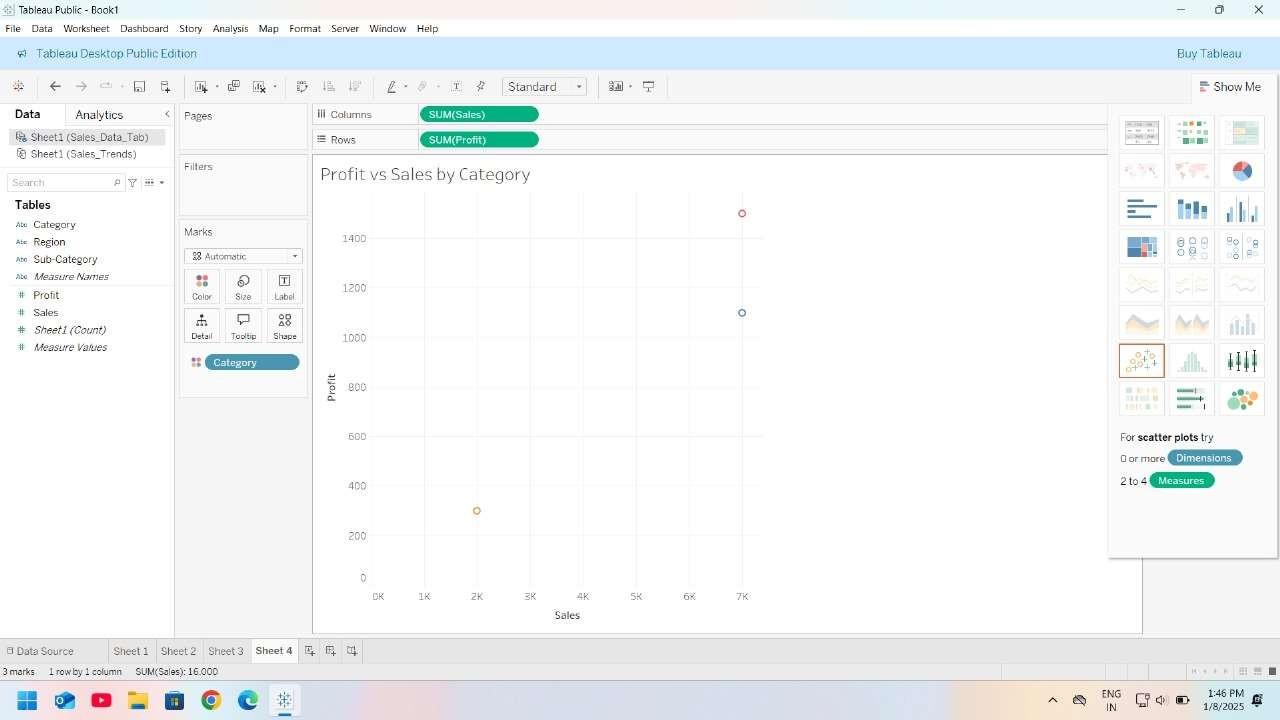


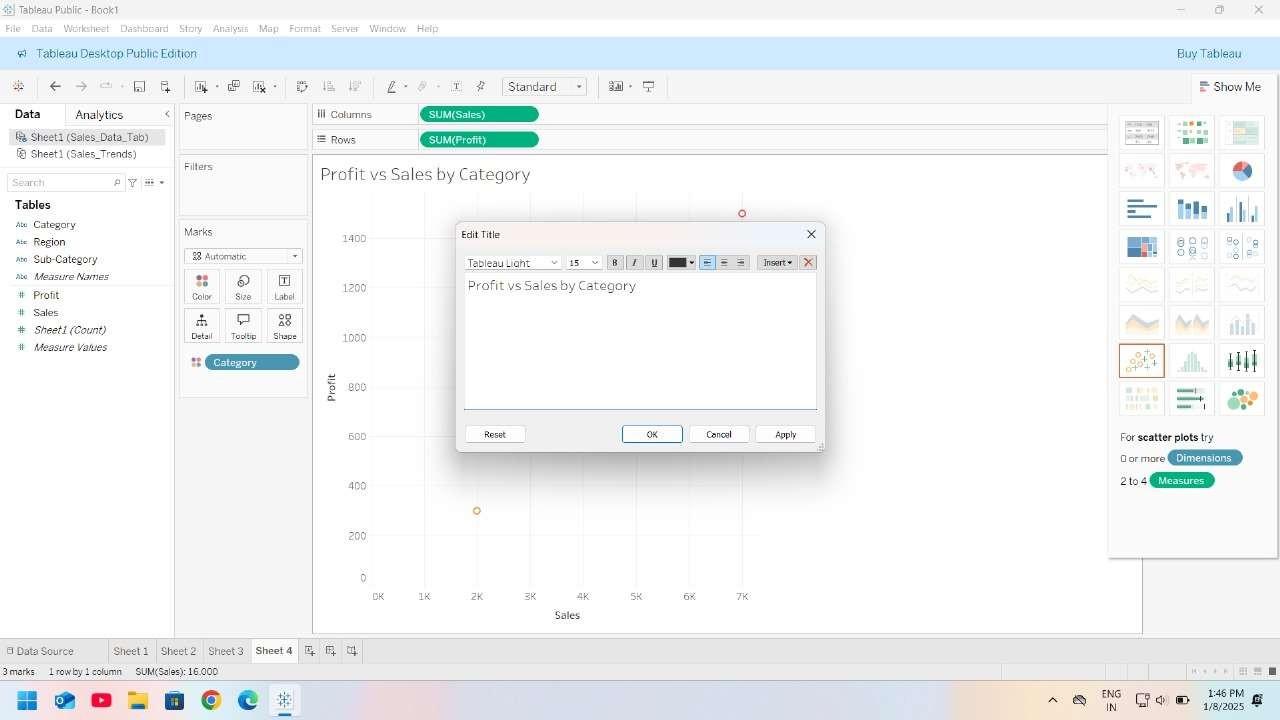


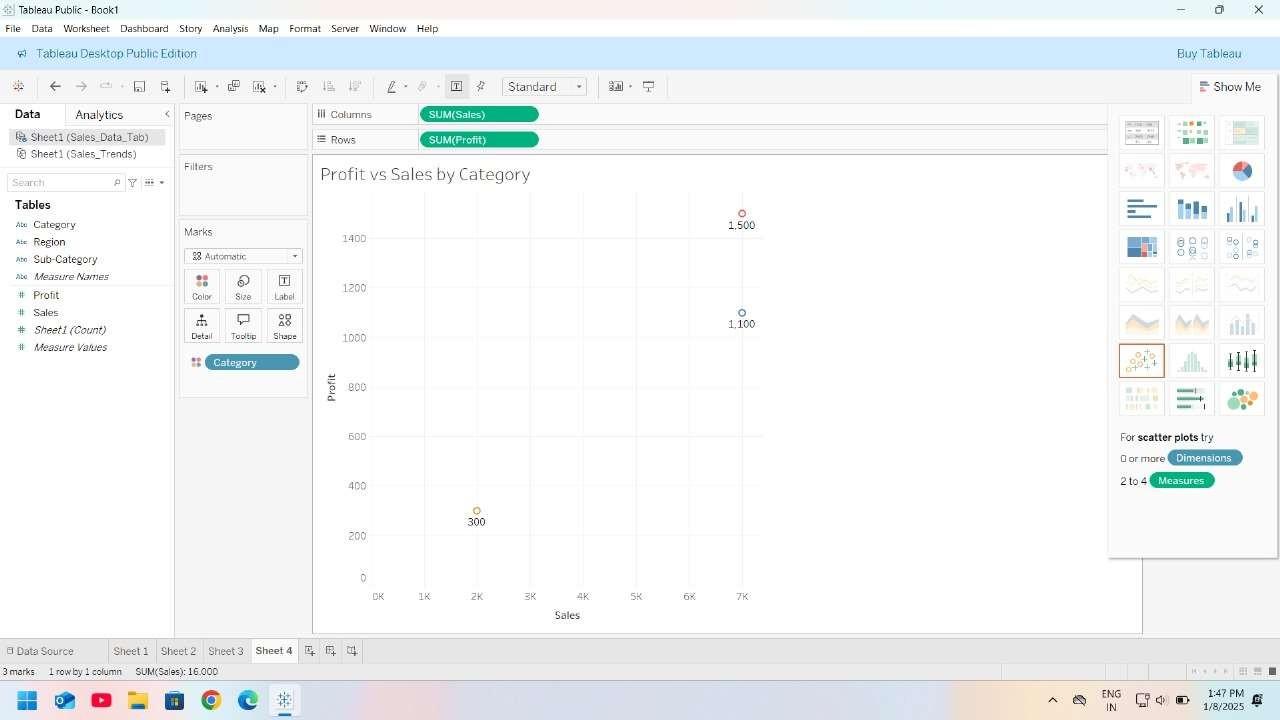
1. ScatterPlot

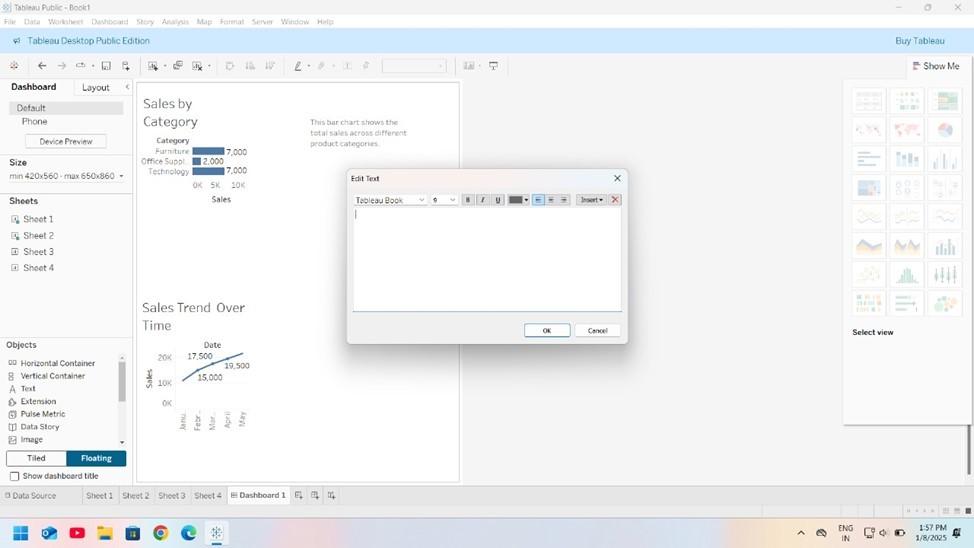
Purpose:AnalyzerelationshipbetweenSalesandProfit.

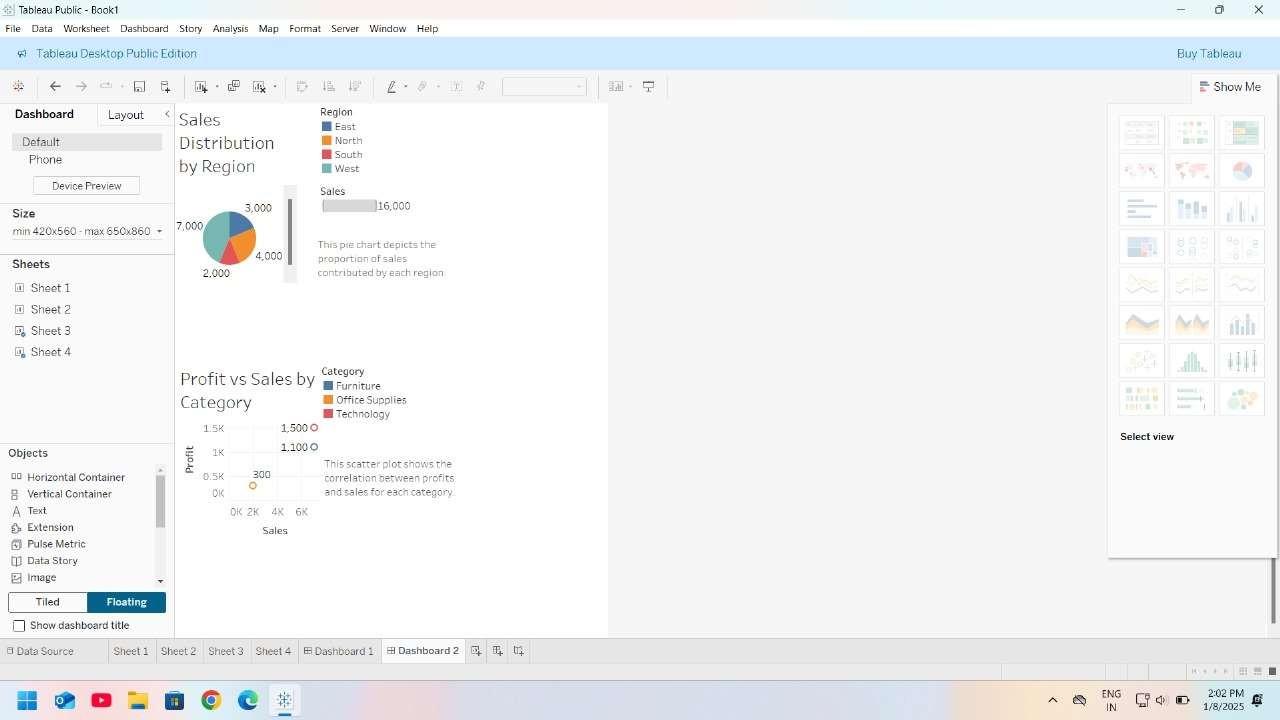
* + DragSalestoColumnsandProfittoRows.
  + AddCategorytoColorontheMarkscard.
  + Title:"ProfitvsSalesbyCategory".
  + Description:"This scatterplotshowsthecorrelationbetweenprofitandsales for each category."

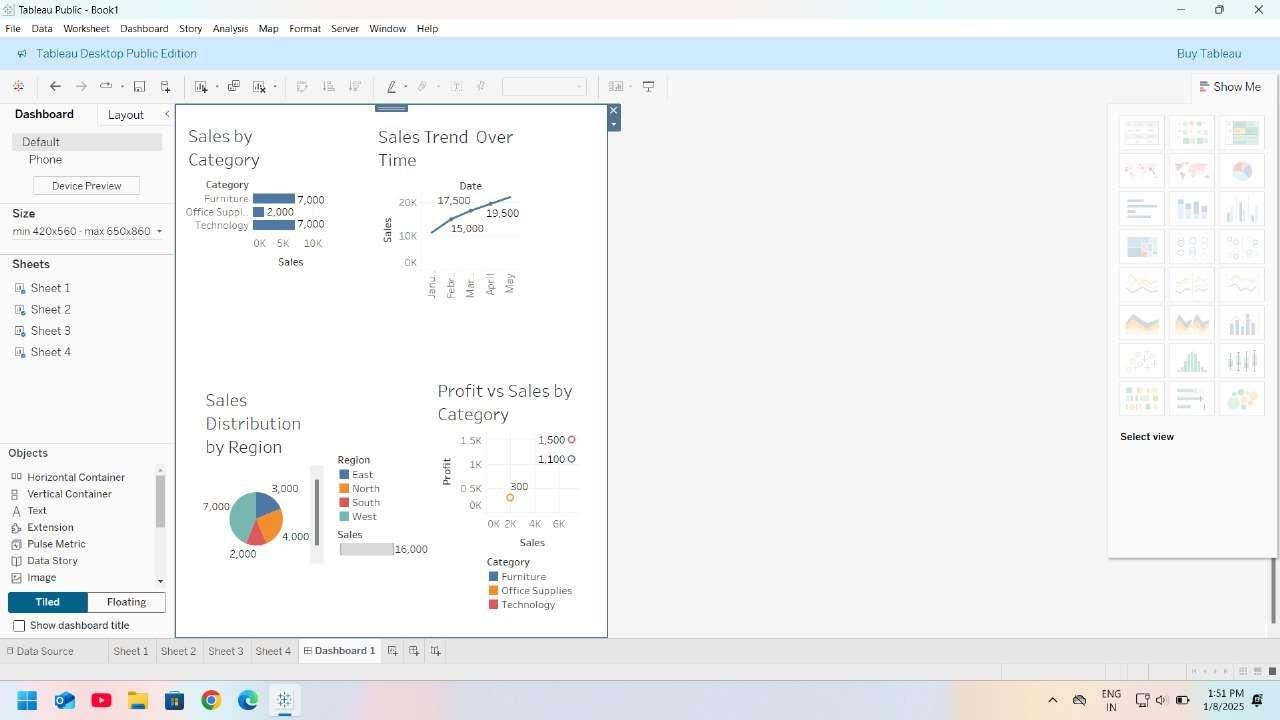












Step4:SaveandExport

1. SavetheTableauWorkbook.
2. ExportthechartsasimagesordashboardsbygoingtoFile>Export>Image/PDF.

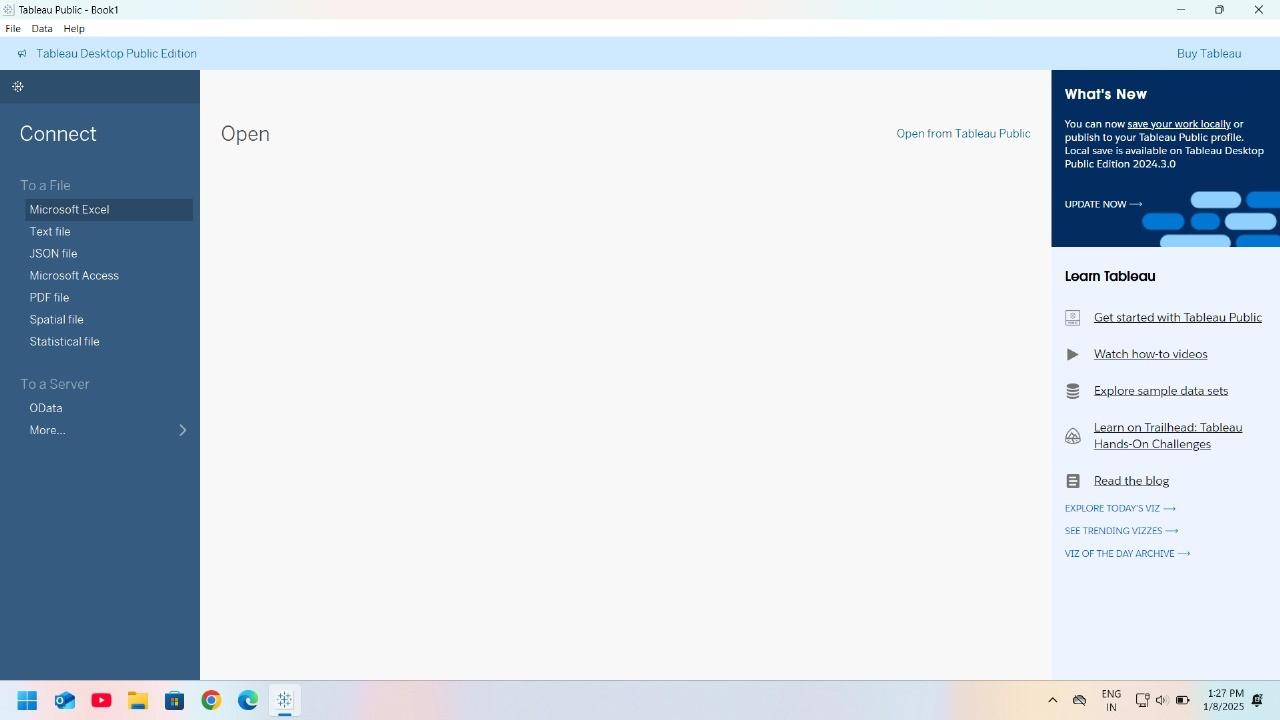
# Practical7

**Aim:** Perform sorting and filtering using tableau, create visualizations and publish it on Tableau Cloud. **Theory:** Tableau is a versatile data visualization tool that enables sorting and filtering of data to uncover meaningfulinsights.Userscaneasilymanipulatedatasetstofocusonspecificinformation,enhancing clarity and relevance.

Tableau provides a wide array of visualization options, allowing users to create interactive charts, graphs, and dashboards. Once visualizations are complete, they can be seamlessly published on Tableau Cloud, facilitating collaboration and real-time access to data insights across teams and organizations.

## Step1:ConnecttoData

1. **OpenTableauDesktop**andconnecttoyourdata source.
   * Click**File**>**Open**andselectyourExcel file(Sales\_Trends.xlsx)oranyother dataset.
   * Tableauwill displaythedatainthe**DataSource** pane.



## Step2:ApplyingFilters

Filtersareusedtodisplayonlyrelevantdata.Here's howtoapply filters:

## FilterbyDate:

* + DragDate tothe**Filters**shelf.
  + Inthefilter options,chooseafilter type.For example:
    - **RelativeDate**:Filtertoshowdata forthepastmonth,quarter, or year.
    - **RangeofDates**:Specifyacustomdate range.
    - **ExactDate**:Filtertodisplayspecificdays.

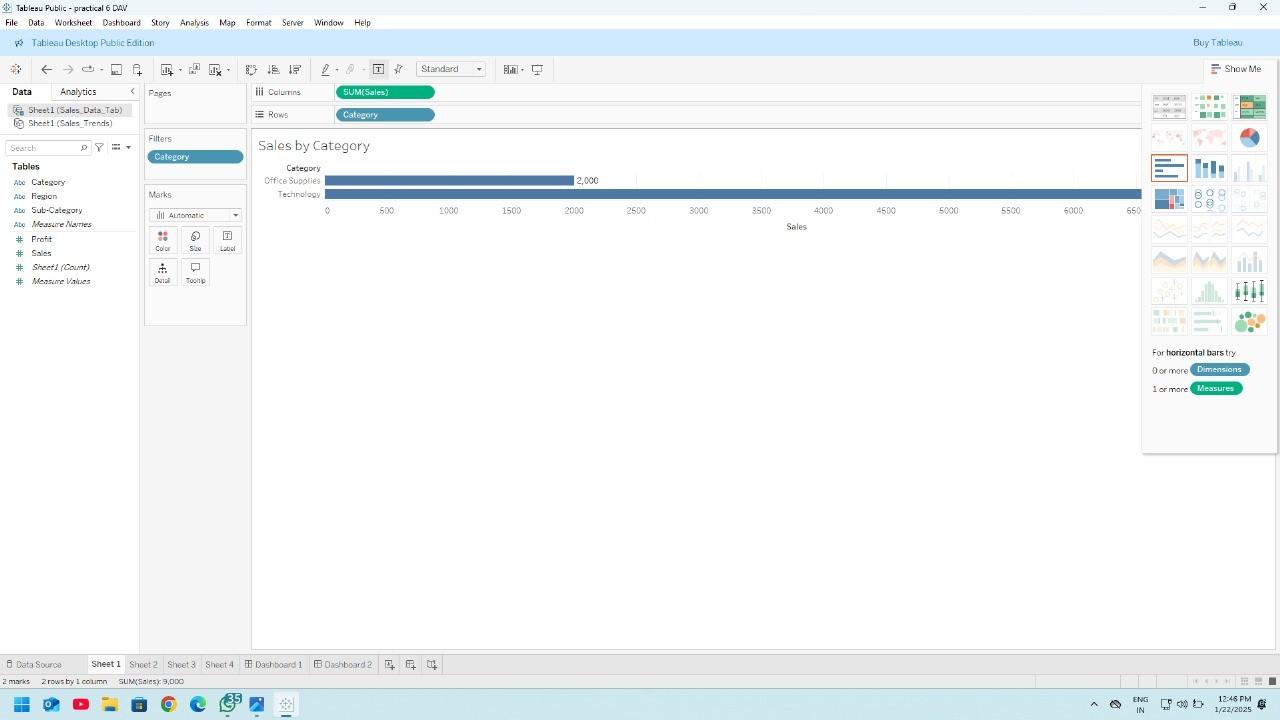
## FilterbySales:

* + DragSalestothe**Filters**shelf.
  + Setafilterrangetoshowonlysales abovea certainvalue,suchasgreaterthan$5000.



3.**ApplyMultipleFilters**:

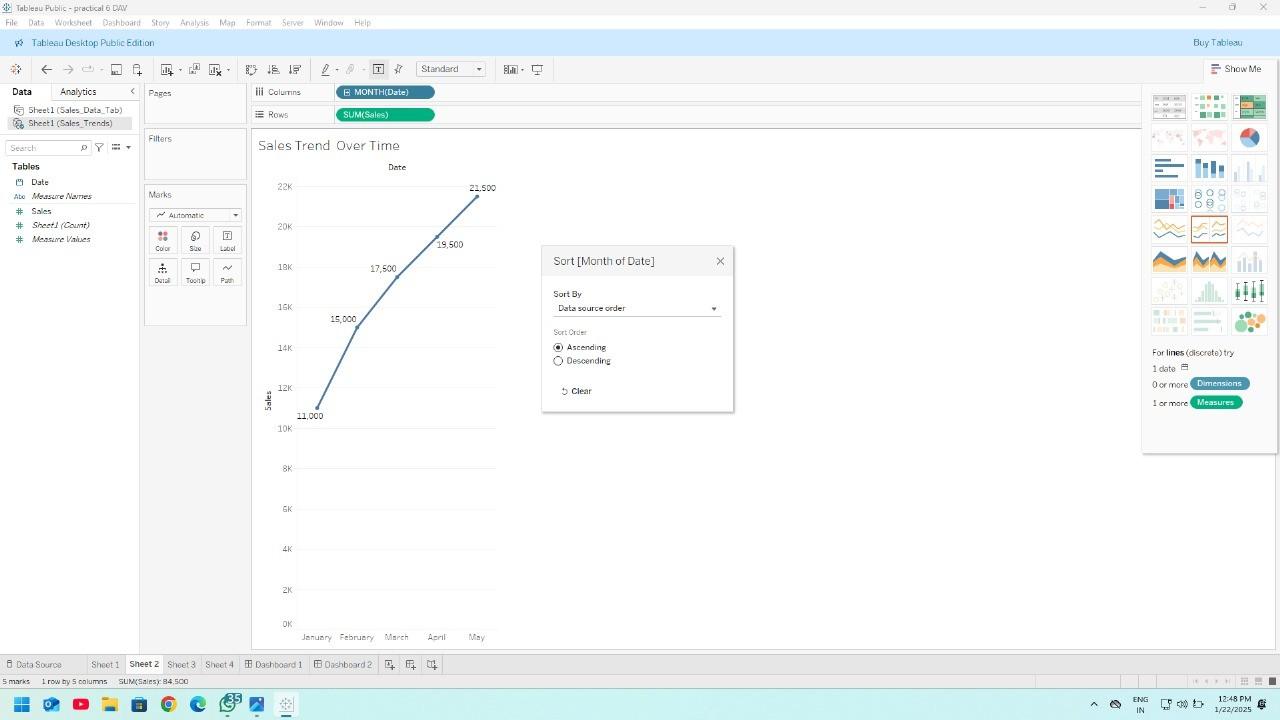
Youcanfilterbymultiplefields.Forexample,filterby**Date**and**Region**atthesametime.



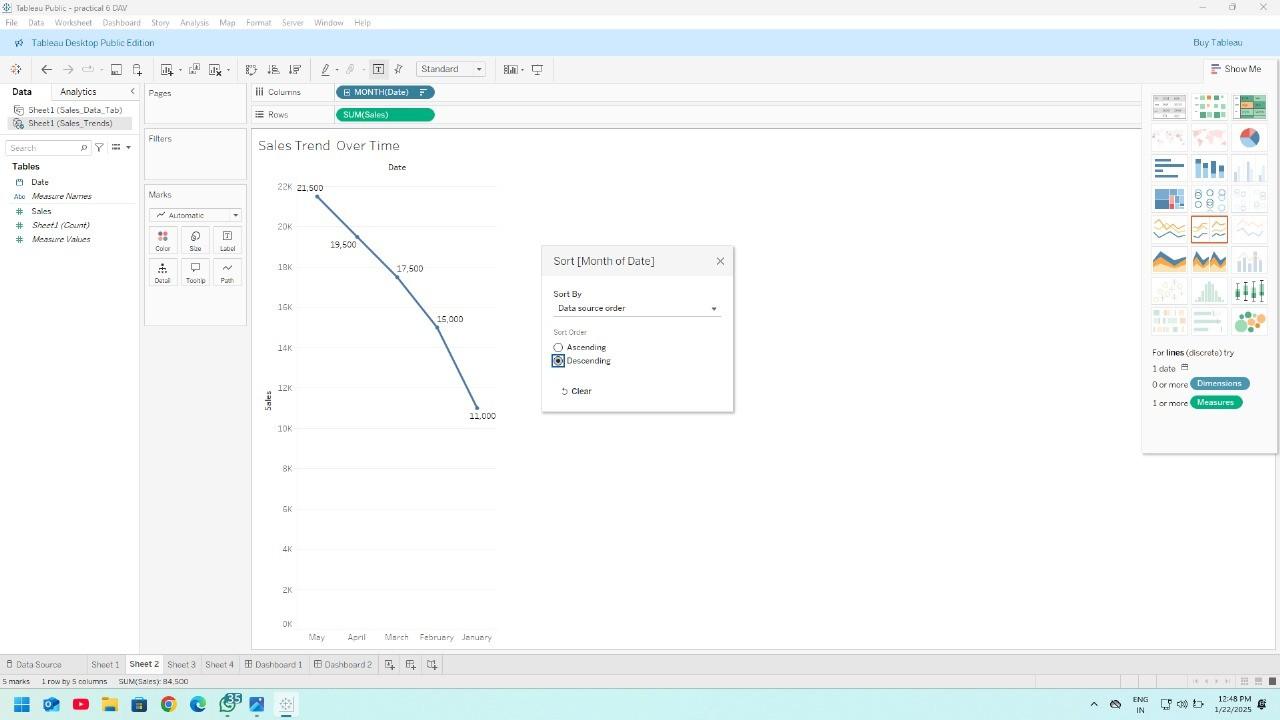
## Step3:SortingData

Sortingallowsyoutoreorderyourdatabasedonspecificfields.Here'showtosortdata:

## SortbyDate:

* + DragDateto**Columns**andSalesto **Rows**tocreateabasicline chart.
  + Right-clicktheDate fieldontheColumnsshelfandselect**Sort**.
  + Choose**Sortby Field**:
    - **SortOrder**:AscendingorDescending.
    - **SortBy**:Fieldvalue(youcanchooseanother fieldtosortby,suchas Sales).

## SortbySales:

* + Right-clickonthe**Sales** axisandchoose**Sort**.
  + Select**Descending**toshowthehighestsalesatthetopor**Ascending**toshowthelowestfirst.

## Step4:PublishtoTableau Cloud

1. **SignintoTableauCloud**:
   * Goto**Server**>**TableauCloud**inthemenubar.
   * LoginwithyourTableauCloudcredentials.

## PublishtheWorkbook:

* + Click**Server**>**TableauCloud**>**Publish Workbook**.
  + Choosetheprojectwhereyouwanttosave theworkbook(orcreateanewproject).
  + Provideanameforyour workbook.
  + Setpermissionsforwho canviewtheworkbook(e.g., yourteamorspecificusers).

1. **PublishtheDataSource**(optional):
   * Ifyouwanttoreusethedatasourceforfutureworkbooks, select**PublishDataSource**.
   * Choosetheproject,give ita name,andselectwhethertomake itavailabletoallusersorspecific ones.

## FinishandConfirm:

* + Afterclicking**Publish**,yourworkbookwillbeavailableonTableauCloud.
  + YoucannowaccessthedashboardandvisualizationsfromTableauCloudorsharethelinkwith others.

# Practical8

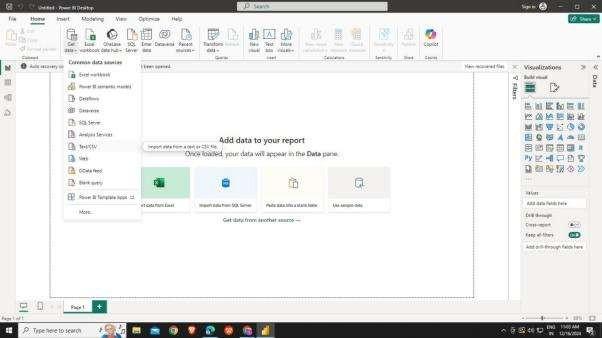
**Aim:**PerformdatavisualizationusingPowerBI

**Theory -** Power BI is a robust business intelligence tool designed for seamless data visualization. It enables users to connect to various data sources, clean and model data, and create interactive reports and dashboards with diverse visualization options. The platform's user-friendly interface allows for easy customization,real-timeanalysis,andeffectivecommunicationofinsights.WithPowerBI,organizationscan quickly transform raw data into compelling visuals, fostering informed decision-making and a deeper understanding of key metrics.

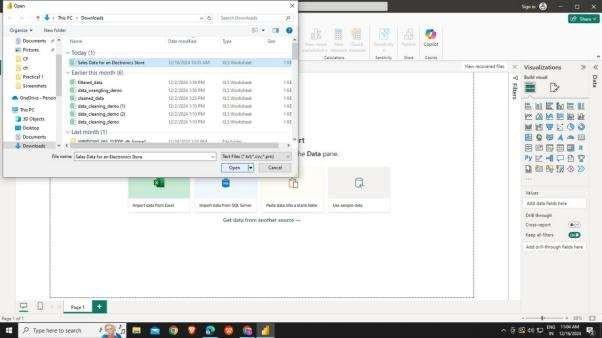
Step1:OpenPowerBIandselectBlankReport.



Step2:ClickonGetDataandSelectText/CSV.

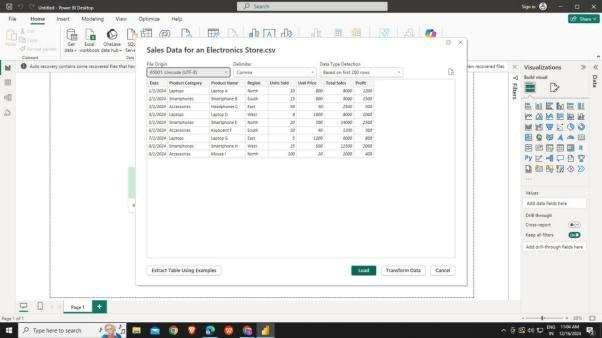


Step3:SelecttheCSVfileandClickonOpen.

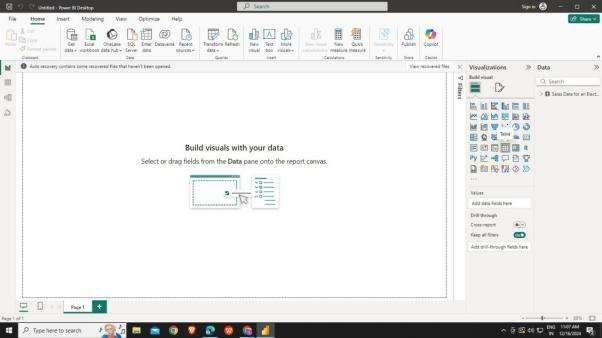


\

Step4:LoadtheCSVfileafterOpeningtheCSVfile.

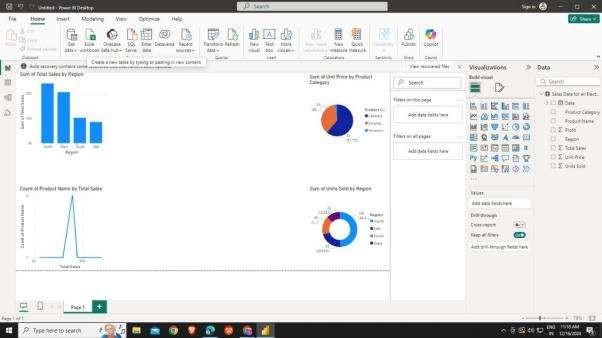


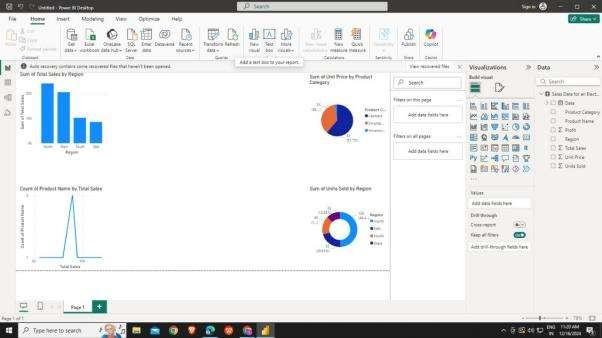
Step5:ADataPanel willbeopenedbesidesVisualizationPanel.Clickonthedrag-down arrow besides the loaded CSV file.





Step6:SelectandVisualizationanddragthedatacomponents intothevarious fieldsof the Visualization.





# Practical9

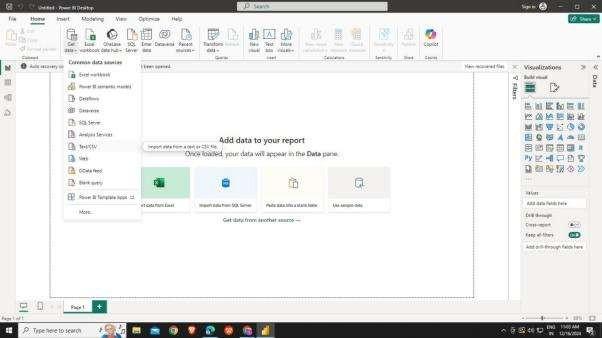
**Aim:**CreateReportsusingPowerBI

**Theory -** Power BI is a user-friendly business analytics tool that facilitates the creation of dynamic and visuallyappealing reports.With seamless connectivityto various data sources, intuitive data transformation features, and a diverse range of visualization options, users can easily design impactful reports. The platform's drag-and-drop functionality simplifies data arrangement, making it accessible for both beginners and seasoned analysts. Real-time updates and cloud-based sharing enhance collaboration, making Power BI avaluabletoolforgeneratinginsightfulreportsthatsupportinformeddecision-makingacrossorganizations.

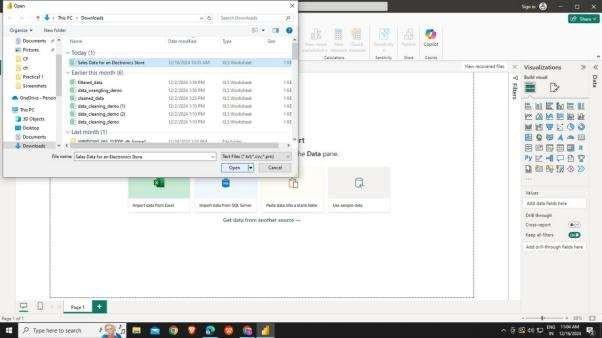
Step1:OpenPowerBIandselectBlankReport.



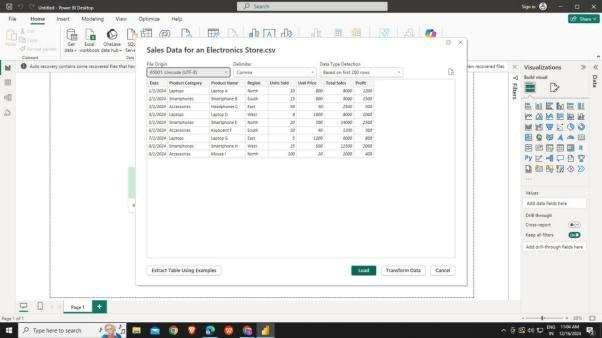
Step2:ClickonGetDataandSelectText/CSV.



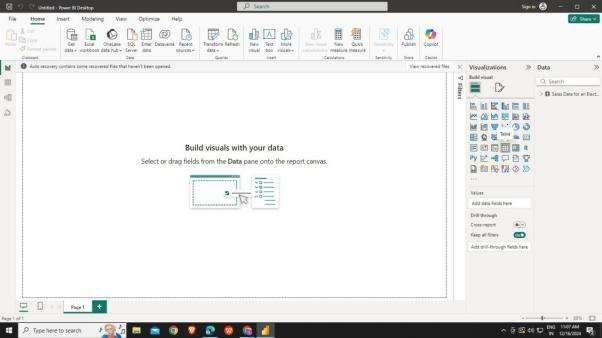
Step3:SelecttheCSVfileandClickonOpen.



Step4:LoadtheCSVfileafterOpeningtheCSVfile.

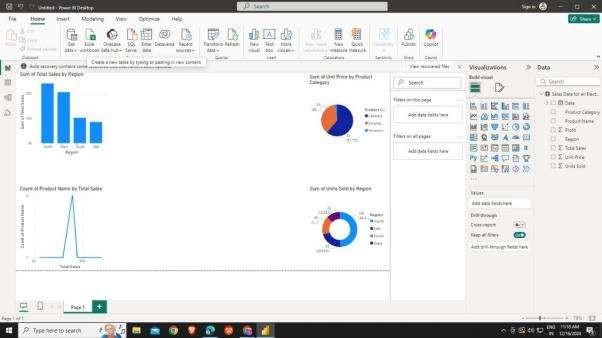


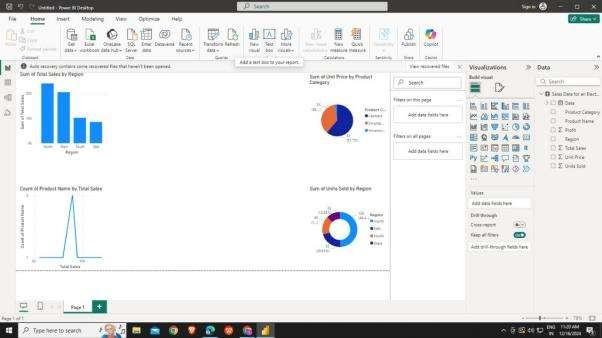
Step5:ADataPanel willbeopenedbesidesVisualizationPanel.Clickonthedrag-down arrow besides the loaded CSV file.



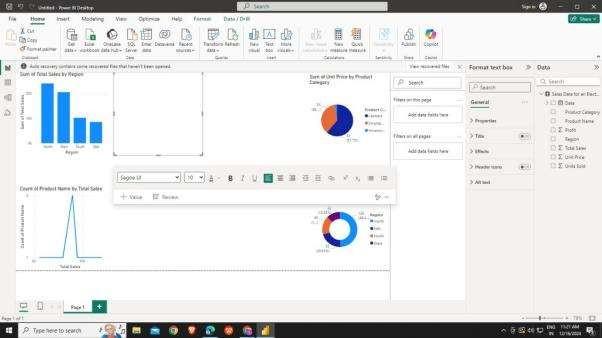


Step6:SelectandVisualizationanddragthedatacomponents intothevarious fieldsof the Visualization.





Step7:AddTextboxfromHomemenuforeachvisualizationtoaddthereport.



Step8:WriteaReportintheTextboxforeachofthevisualizatio

