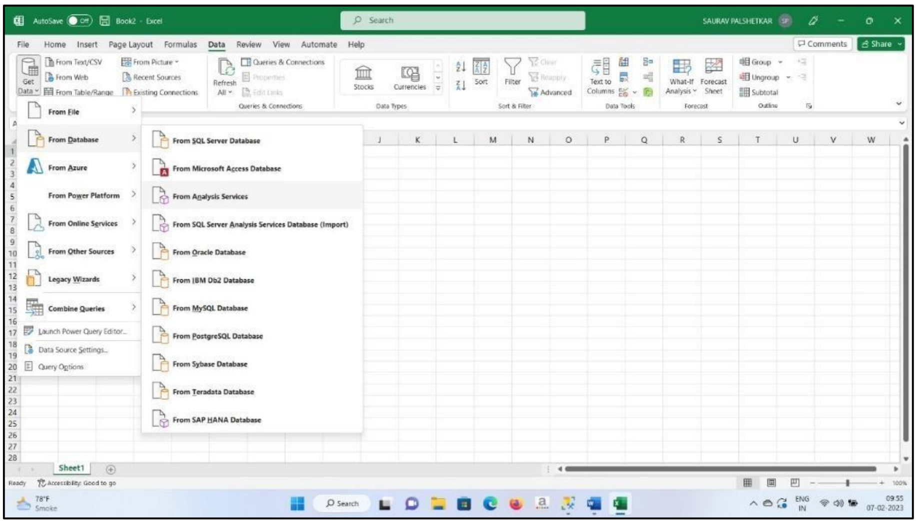
PRACTICAL NO 1

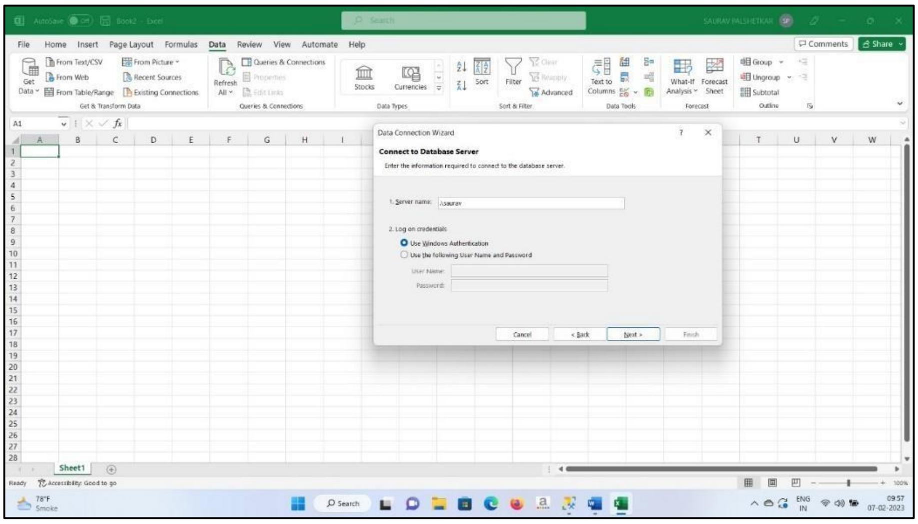
Import the Datawarehouse data in Microsoft Excel and create the Pivot table and Pivot Chart.

Pivot Tables allow you to create a powerful view with data summarized in a grid, both in horizontal and vertical columns (also known as Matrix Views or Cross Tabs)
A pivot chart is the visual representation of a pivot table in Excel. Pivot charts and pivot tables relate to each other.

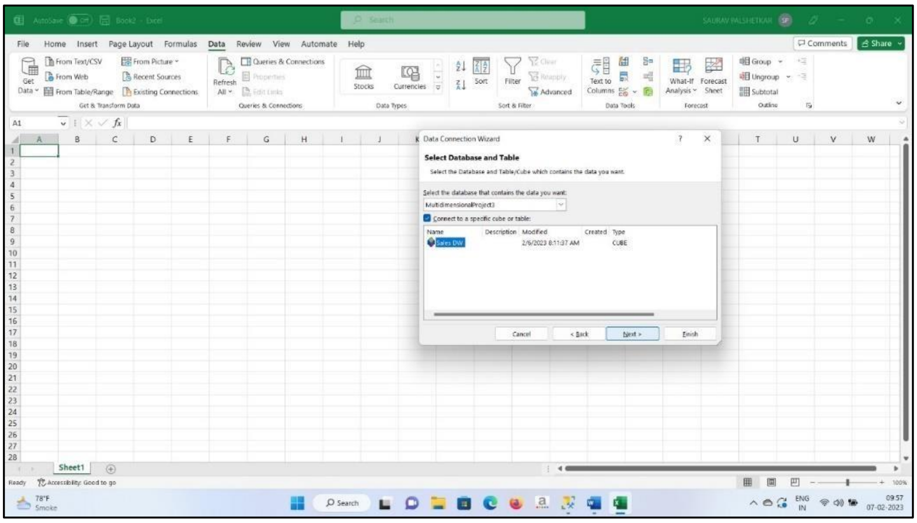
Step 1: Open Excel 2013 (Professional)
Go to Data tab → Get External Data → From Other Sources → From Analysis Services



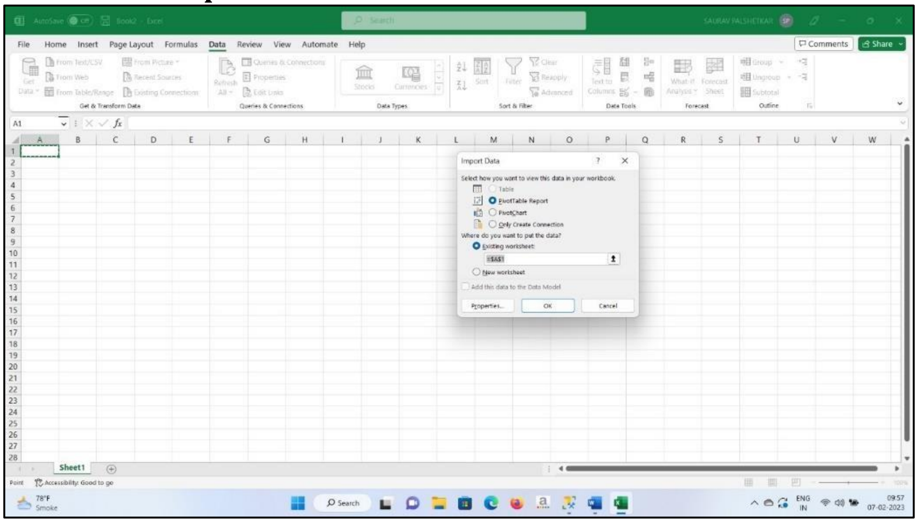
Step 2: Select Server name and Windows Authentication and click on Next



Step 3: Select OLAP (as per created before) click on Next

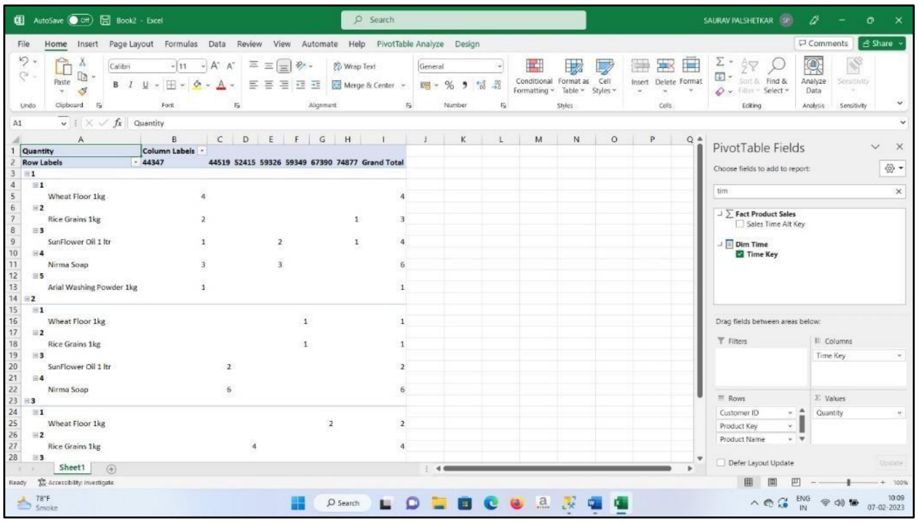


Step 4: Browse and select path name and click on Finish

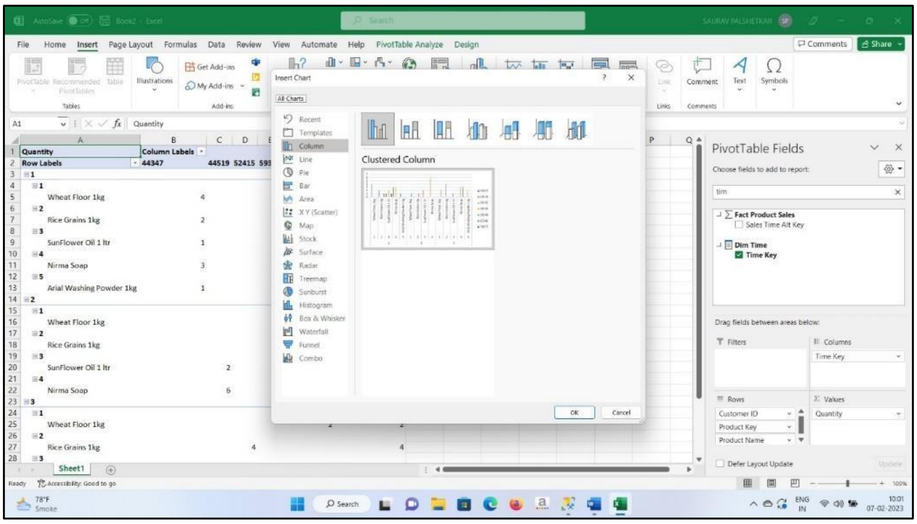


Step 5: Select PivotTableReport → OK

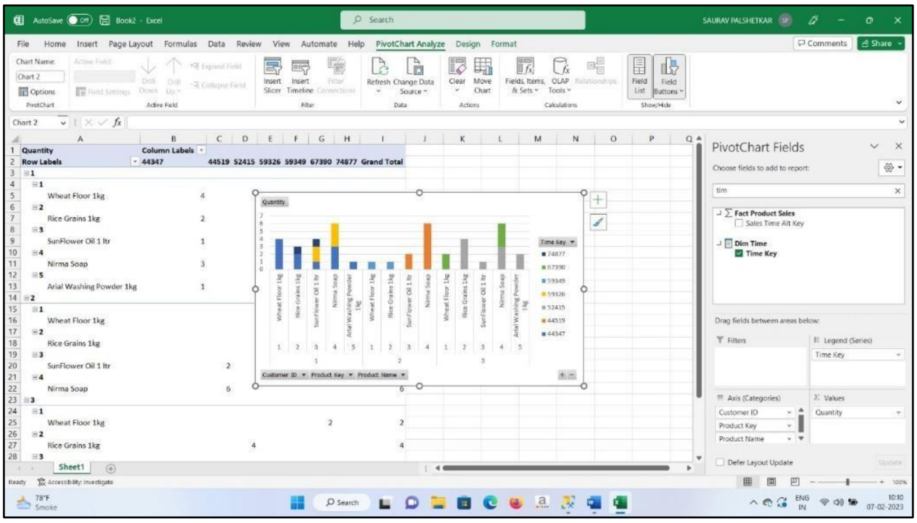
Step 6 : Drag and Drop Fields in rows column and values



Step 7: Go to Insert tab → pivot chart and select Pivot Chart from drop down
Step 8: Select existing connection OLAP Sales DW and click on Open



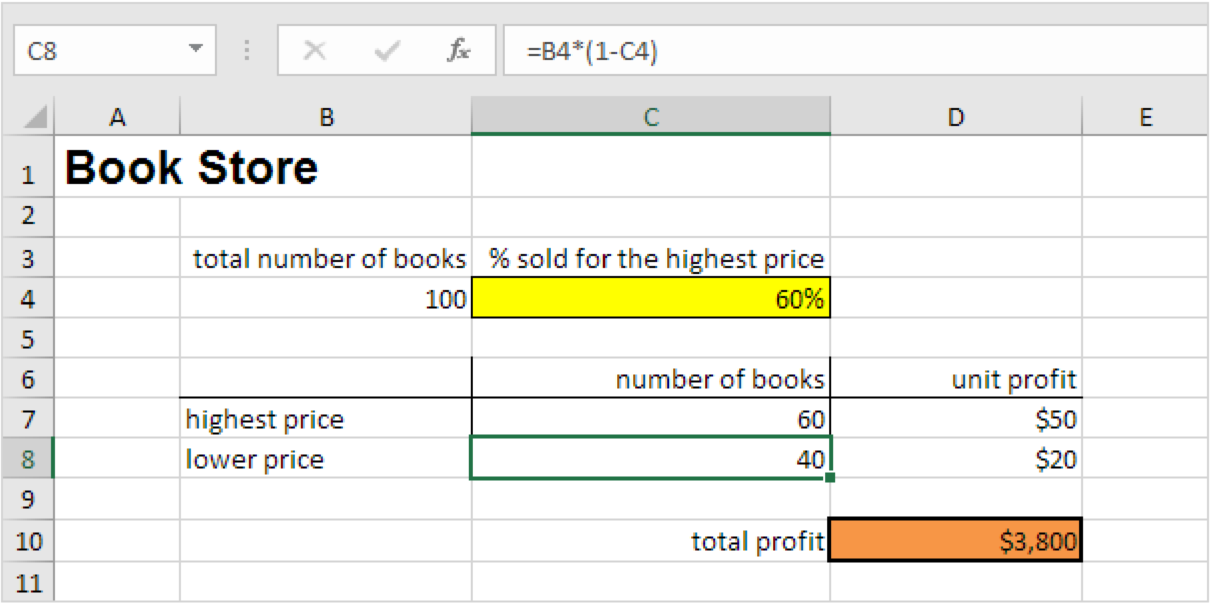
Step 9: Click Ok



PRACTICAL NO 2

Apply the what – if Analysis for data visualization. Design and generate necessary reports based on the data warehouse data.

A book store and have 100 books in storage. You sell a certain % for the highest price of $50 and a certain % for the lower price of $20.
If you sell 60% for the highest price, cell D10 calculates a total profit of 60 * $50 + 40 * $20 = $3800.

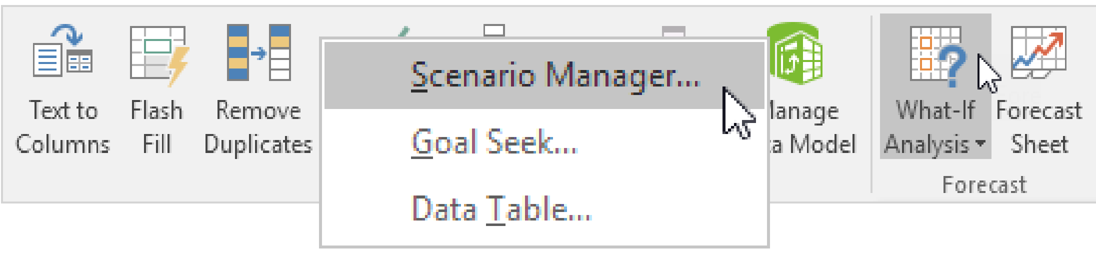


C8 =B4*(1-C4)

	A	B	C	D	E
1	Book Store				
2					
3		total number of books	% sold for the highest price		
4		100	60%		
5					
6			number of books	unit profit	
7		highest price	60	$50	
8		lower price	40	$20	
9					
10			total profit	$3,800	
11					

Create Different Scenarios
But what if you sell 70% for the highest price? And what if you sell 80% for the highest price? Or 90%, or even 100%? Each different percentage is a different scenario. You can use the Scenario Manager to create these scenarios.
Note: You can simply type in a different percentage into cell C4 to see the corresponding result of a scenario in cell D10. However, what-if analysis enables you to easily compare the results of different scenarios. Read on.

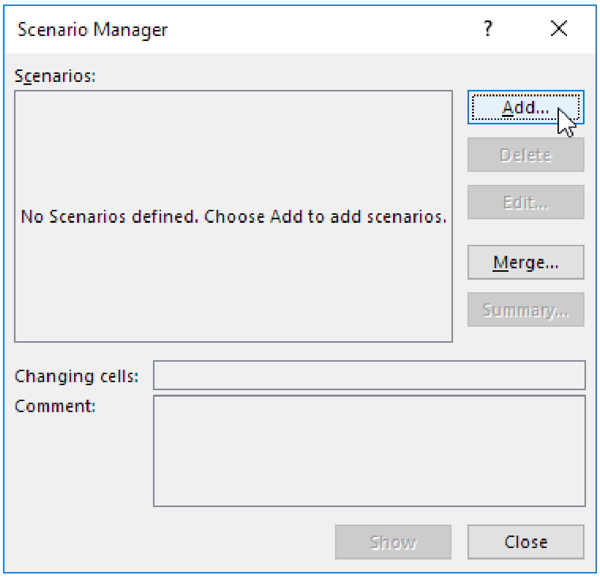
1. On the Data tab, in the Forecast group, click What-If Analysis.



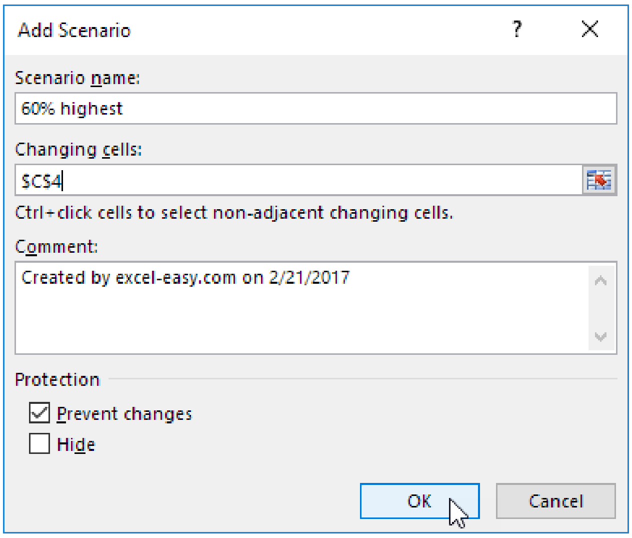
2. Click Scenario Manager.

The Scenario Manager dialog box appears.

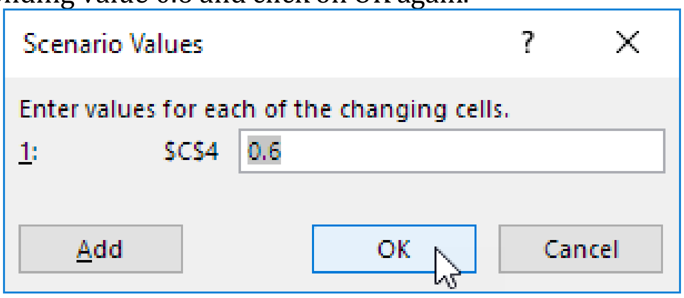
3. Add a scenario by clicking on Add.



4. Type a name (60% highest), select cell C4 (% sold for the highest price) for the Changing cells and click on OK.

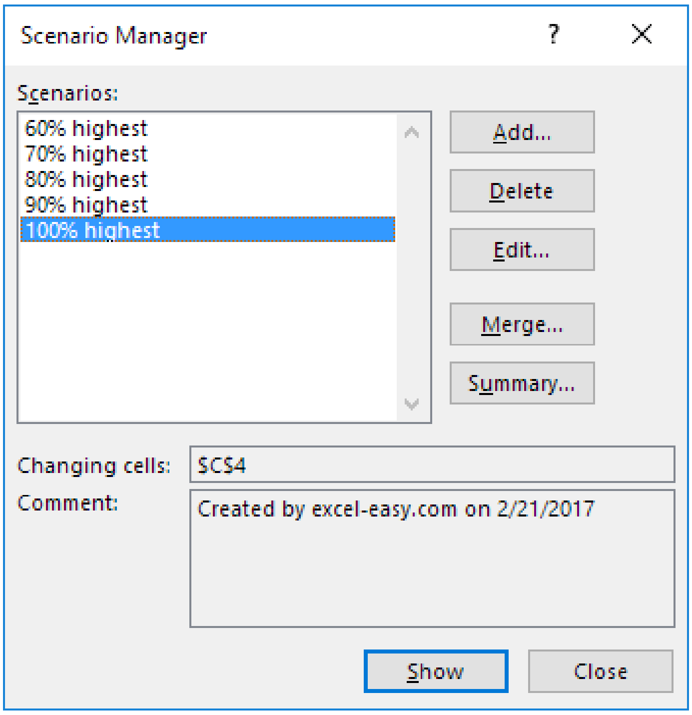


5. Enter the corresponding value 0.6 and click on OK again.



6. Next, add 4 other scenarios (70%, 80%, 90% and 100%).

Finally, your Scenario Manager should be consistent with the picture below:



PRACTICAL NO 3

Perform the data classification using classification algorithm.

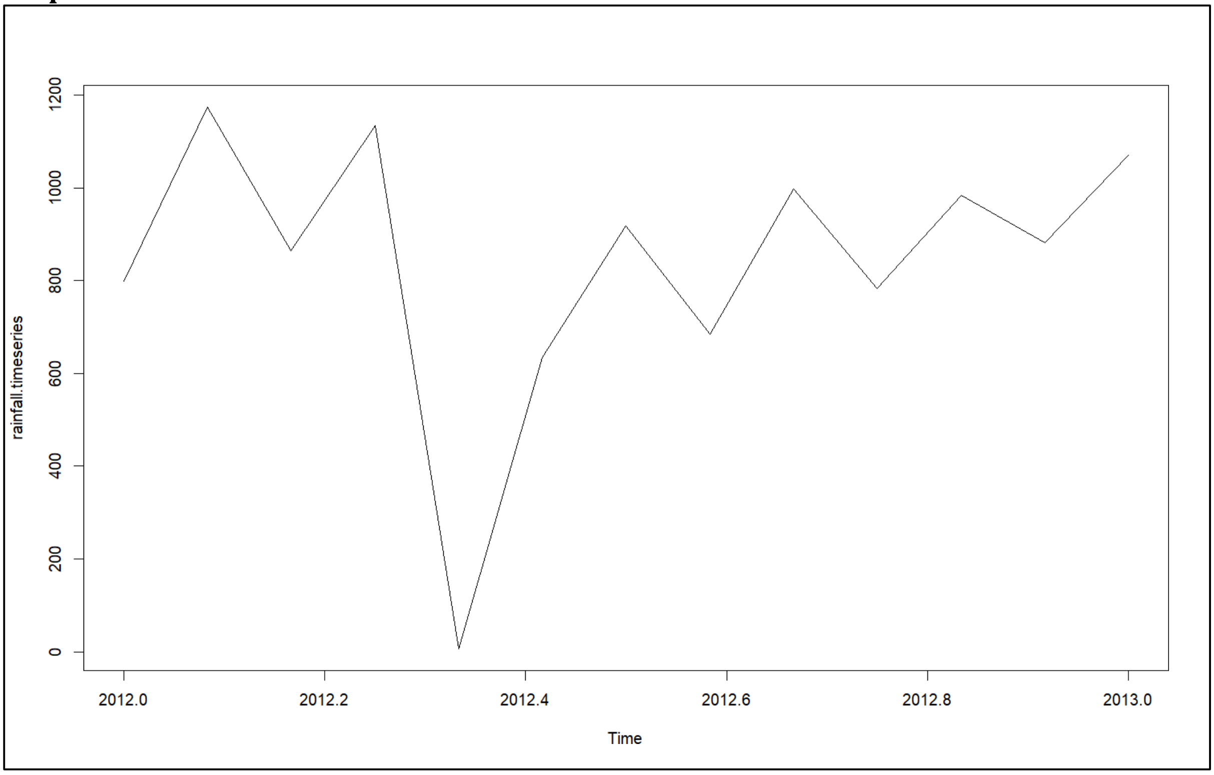
R Script:

```
rainfall <- c(799,1174.8,865.1,1134.6,6,635.4,918.5,685.5,998.6,784.2,985,882.8,1071)
rainfall.timeseries <- ts(rainfall,start = c(2012,1),frequency=12)
print(rainfall.timeseries)
plot(rainfall.timeseries)
```

Results:

```
        Jan    Feb    Mar    Apr    May    Jun    Jul    Aug    Sep    Oct
2012  799.0 1174.8  865.1 1134.6    6.0  635.4  918.5  685.5  998.6  784.2
2013 1071.0
        Nov    Dec
2012  985.0  882.8
2013
```

Graph:

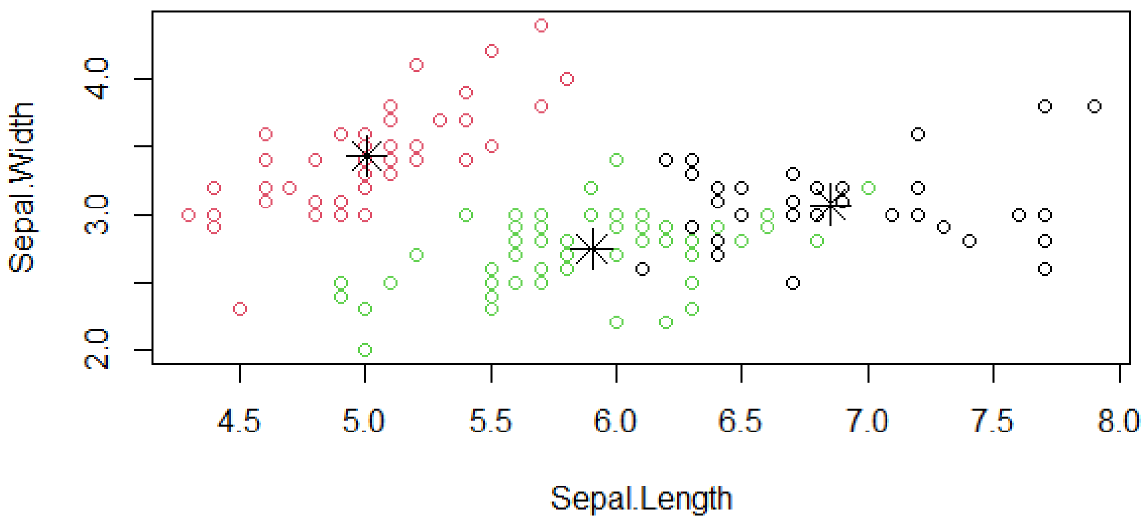


PRACTICAL NO 4

Perform the data clustering using clustering algorithm.

R Script:

```
newiris<-iris
print(iris)
##Sepal.Length ,Sepal.Width ,Petal.Length ,Petal.Width ,Species
newiris$Species<-NULL
(kc<-kmeans(newiris,3))

table(iris$Species,kc$cluster)
plot(newiris[c("Sepal.Length","Sepal.Width")],col=kc$cluster)
points(kc$centers[,c("Sepal.Length","Sepal.Width")],col=1.3,pch=8,cex=2)
```

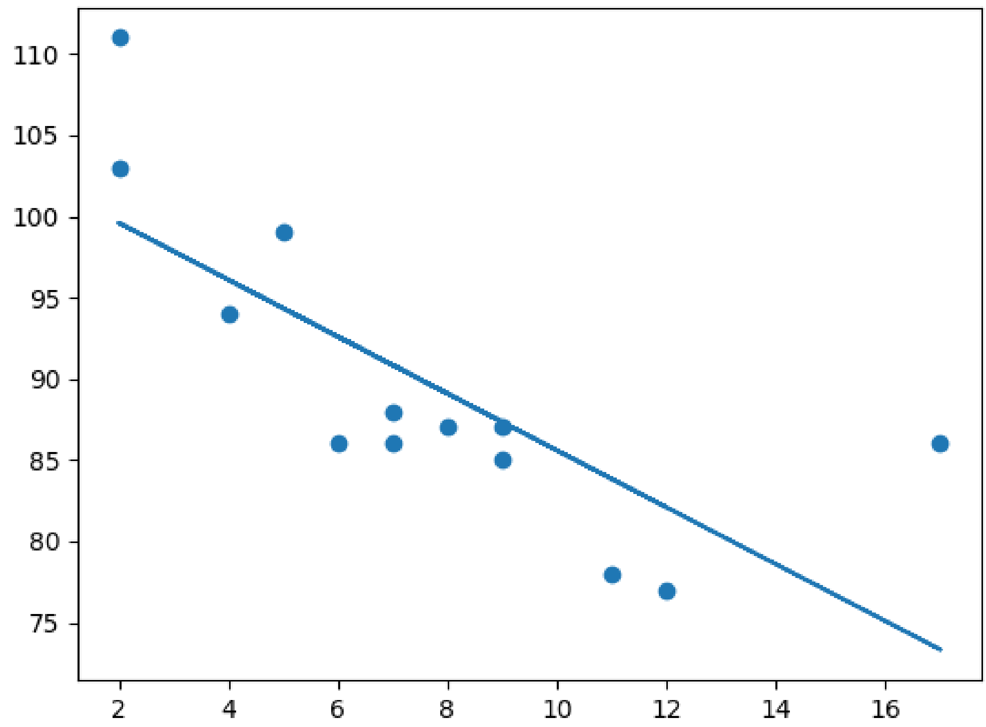


PRACTICAL NO 5

Perform the logistic regression on the given data warehouse data using python

Python:

```
#x: age of the car , y: speed of the car


import matplotlib.pyplot as plt
from scipy import stats

x = [5,7,8,7,2,17,2,9,4,11,12,9,6]
y = [99,86,87,88,111,86,103,87,94,78,77,85,86]


slope, intercept, r, p, std_err = stats.linregress(x, y)


def myfunc(x):
  return slope * x + intercept
mymodel = list(map(myfunc, x))


#Plot the graph
plt.scatter(x, y)
plt.plot(x, mymodel)
plt.show()
```



#The r value ranges from -1 to 1,

#where 0 means no relationship, and 1 (and -1) means 100% related.

print(r)

#The result -0.76 shows that there is a relationship, not perfect,

#but it indicates that we could use linear regression in future predictions.

##Predict Future Values : car age is 10 , predict speed

speed = myfunc(10)

print("Speed of 10 year old car is : ",speed)

Outpt: Speed of 10 year old car is : 85.59308314937454

PRACTICAL NO 6

Perform the logistic regression on the given data warehouse data

Python (Jupyter notebook):

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from imblearn.over_sampling import SMOTE  # For handling imbalanced data

# Load the dataset
df = pd.read_csv("data.csv")

# Prepare features and target
X = df.iloc[:, :-1].values  # Features
y = df.iloc[:, -1].values   # Target (0 or 1)

# Split data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Balance the dataset using SMOTE
smote = SMOTE()
X_train, y_train = smote.fit_resample(X_train, y_train)

# Standardize features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Train Logistic Regression with class balancing
model = LogisticRegression(class_weight='balanced')
model.fit(X_train, y_train)

# Predict
y_pred = model.predict(X_test)

# Evaluate
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
report = classification_report(y_test, y_pred, zero_division=1)

# Print results
print(f"Model Accuracy: {accuracy:.2f}")
print("Confusion Matrix:\n", conf_matrix)
print("Classification Report:\n", report)
```

Output of Practical 6:-

```
Class distribution before SMOTE: Counter({np.int64(1): 3, np.int64(0): 1})
Skipping SMOTE because the minority class has only one sample.
Model Accuracy: 0.00
Confusion Matrix:
 [[0 1]
 [1 0]]
Classification Report:
               precision    recall  f1-score   support

           0       0.00      0.00      0.00       1.0
           1       0.00      0.00      0.00       1.0

    accuracy                           0.00       2.0
   macro avg       0.00      0.00      0.00       2.0
weighted avg       0.00      0.00      0.00       2.0
```

PRACTICAL NO 7

Write a Python Program to read data from a csv file,perform simple data analysis and generate basic insights

Python(Jupyter notebook):

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

# Load the CSV file
df = pd.read_csv('sales_data.csv')

# Display first 5 rows
print("First 5 rows of the dataset:")
print(df.head())
```

```
First 5 rows of the dataset:
   Order_ID Product     Category  Sales  Quantity  Profit
0       101  Laptop  Electronics  50000         2    7000
1       102  Mobile  Electronics  20000         1    3000
2       103   Chair    Furniture   5000         4    1000
3       104    Desk    Furniture  15000         2    2000
```

```
# Dataset information
print("Dataset  Info:")
print(df.info())
```

```
Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4 entries, 0 to 3
Data columns (total 6 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Order_ID  4 non-null      int64
 1   Product   4 non-null      object
 2   Category  4 non-null      object
 3   Sales     4 non-null      int64
 4   Quantity  4 non-null      int64
 5   Profit    4 non-null      int64
dtypes: int64(4), object(2)
memory usage: 324.0+ bytes
None
```

```
# Check for missing values
print("Missing Values in Dataset:")
print(df.isnull().sum())
```

```
Missing Values in Dataset:
Order_ID    0
Product     0
Category    0
Sales       0
Quantity    0
Profit      0
dtype: int64
```

#Dropping rows with missing values:
#df.dropna(inplace=True)

# Summary statistics
print("Summary Statistics:")
print(df.describe())

```
Summary Statistics:
         Order_ID         Sales  Quantity      Profit
count    4.000000      4.000000  4.000000     4.00000
mean   102.500000  22500.000000  2.250000  3250.00000
std      1.290994  19364.916731  1.258306  2629.95564
min    101.000000   5000.000000  1.000000  1000.00000
25%    101.750000  12500.000000  1.750000  1750.00000
50%    102.500000  17500.000000  2.000000  2500.00000
75%    103.250000  27500.000000  2.500000  4000.00000
max    104.000000  50000.000000  4.000000  7000.00000
```

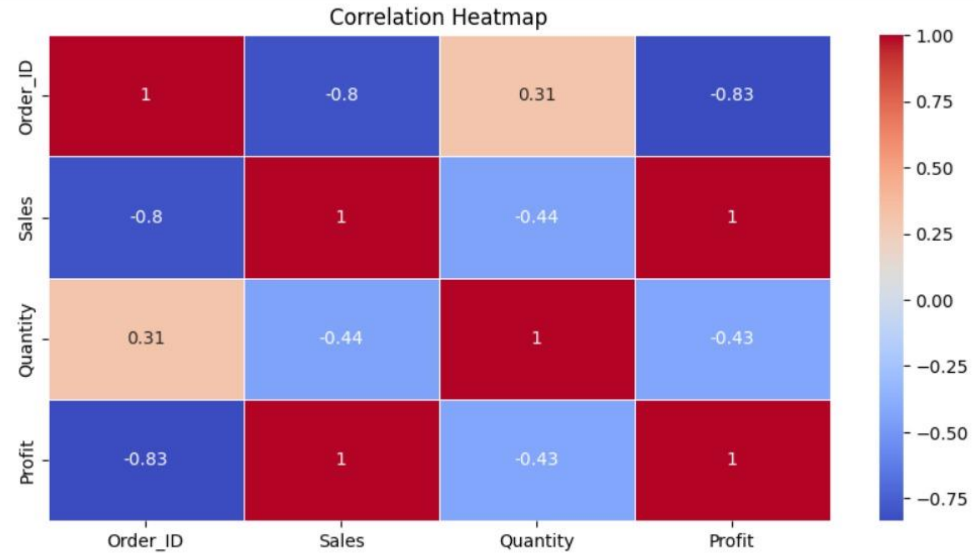
# Count unique values in each column
print("Unique Values per Column:")
print(df.nunique())

```
Unique Values per Column:
Order_ID    4
Product     4
Category    2
Sales       4
Quantity    3
Profit      4
dtype: int64
```

print("Correlation Matrix:")
print(df.select_dtypes(include=['number']).corr())

```
Correlation Matrix:
          Order_ID     Sales  Quantity    Profit
Order_ID  1.000000 -0.800000  0.307794 -0.834497
Sales    -0.800000  1.000000 -0.444591  0.998124
Quantity  0.307794 -0.444591  1.000000 -0.428088
Profit   -0.834497  0.998124 -0.428088  1.000000
```

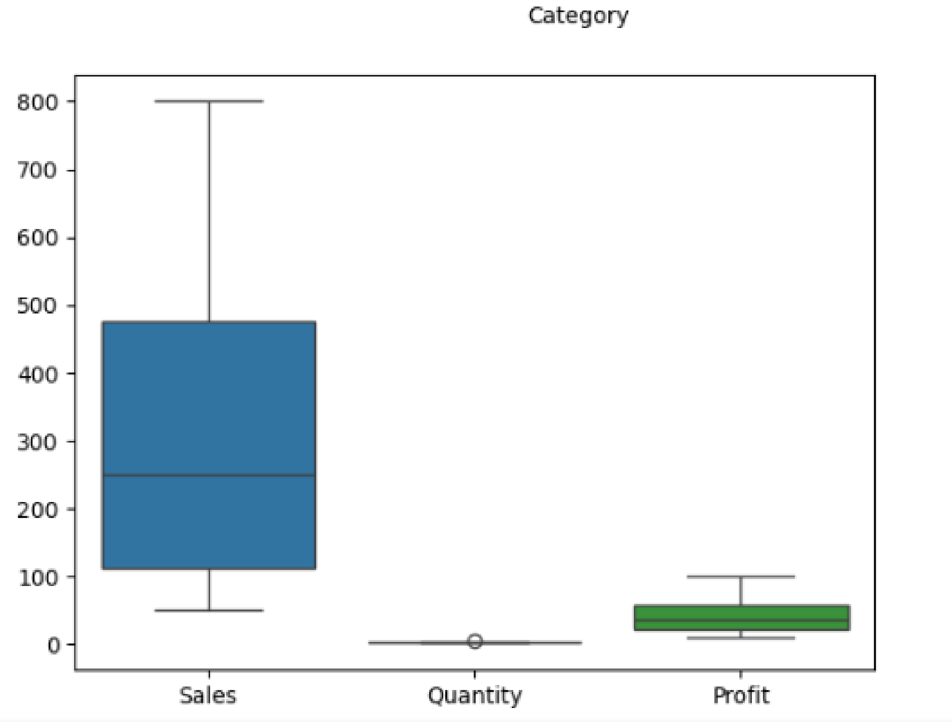
```
# Visualization: Correlation Heatmap
# Select only numeric columns for correlation numeric_
df = df.select_dtypes(include=['number'])

# Plot heatmap
plt.figure(figsize=(10, 5))
sns.heatmap(numeric_df.corr(), annot=True, cmap='coolwarm', linewidths=0.5)
plt.title("Correlation Heatmap")
plt.show()
```




```
''' (for data which do not contain date column)
# Correlation Matrix
print("\nCorrelation Matrix:")
print(df.corr())

# Visualization: Correlation Heatmap
plt.figure(figsize=(10, 5))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm', linewidths=0.5)
plt.title("Correlation Heatmap")
plt.show()
'''

# Visualization: Count plot for Product Category
plt.figure(figsize=(8, 4))
sns.countplot(x='Category', data=df, palette="Set2")
plt.title("Count of Products in Each Category")
plt.xticks(rotation=45)
plt.show()
```

#Box plot
sns.boxplot(data=df[['Sales', 'Quantity', 'Profit']])
plt.show()

OUTPUT:-



PRACTICAL NO 8

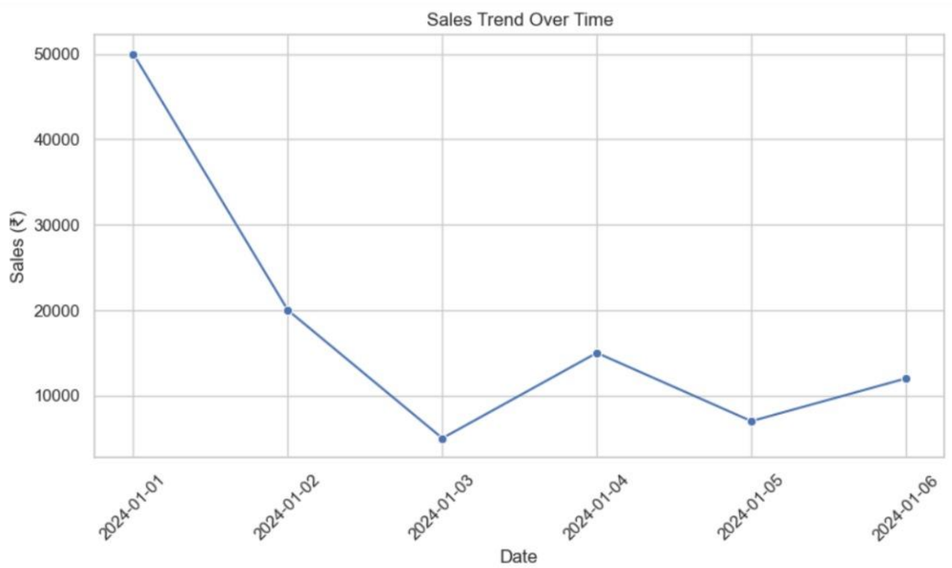
Perform data visualization using python on the sales data.

Python (Jupyter notebook):

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

# Load the CSV file
df = pd.read_csv('sales_data_visualization.csv')

# Convert Date column to datetime format
df["Date"] = pd.to_datetime(df["Date"])

# Set Seaborn style
sns.set(style="whitegrid")
```

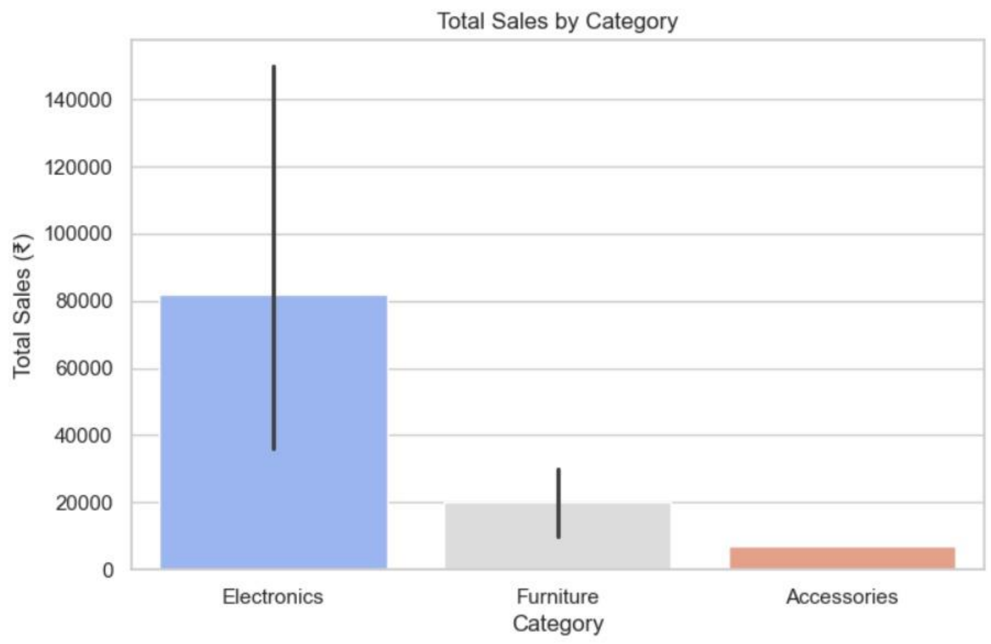
 # 1. Line Plot: Sales Trend Over Time

```
plt.figure(figsize=(10, 5))
sns.lineplot(x="Date", y="Sales", data=df, marker="o", color="b")
plt.title("Sales Trend Over Time")
plt.xlabel("Date")
plt.ylabel("Sales (₹)")
plt.xticks(rotation=45)
plt.show()
```

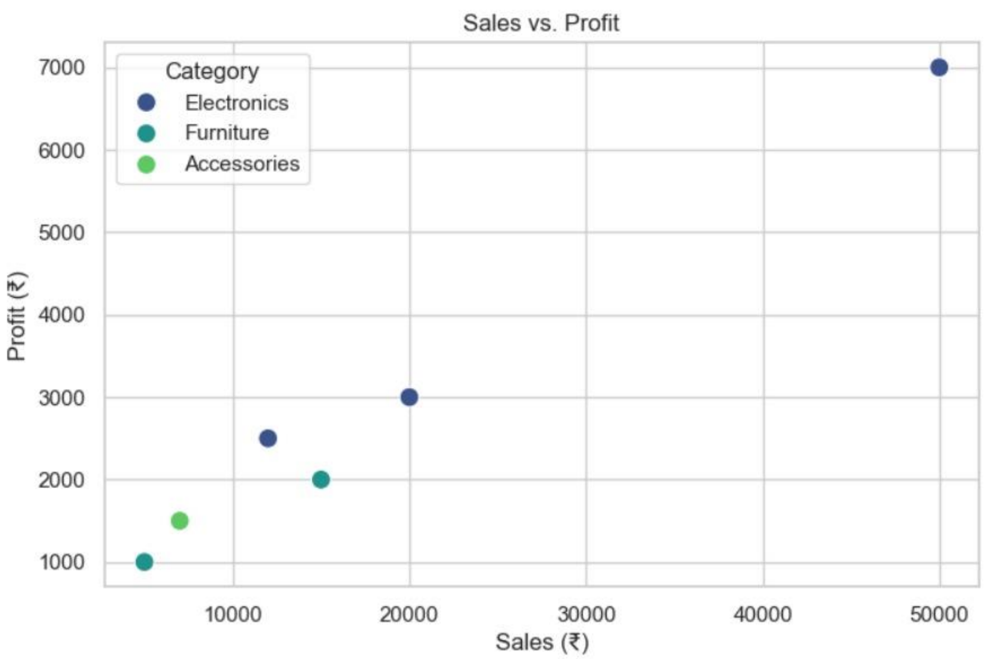


 # 2. Bar Plot: Total Sales by Category

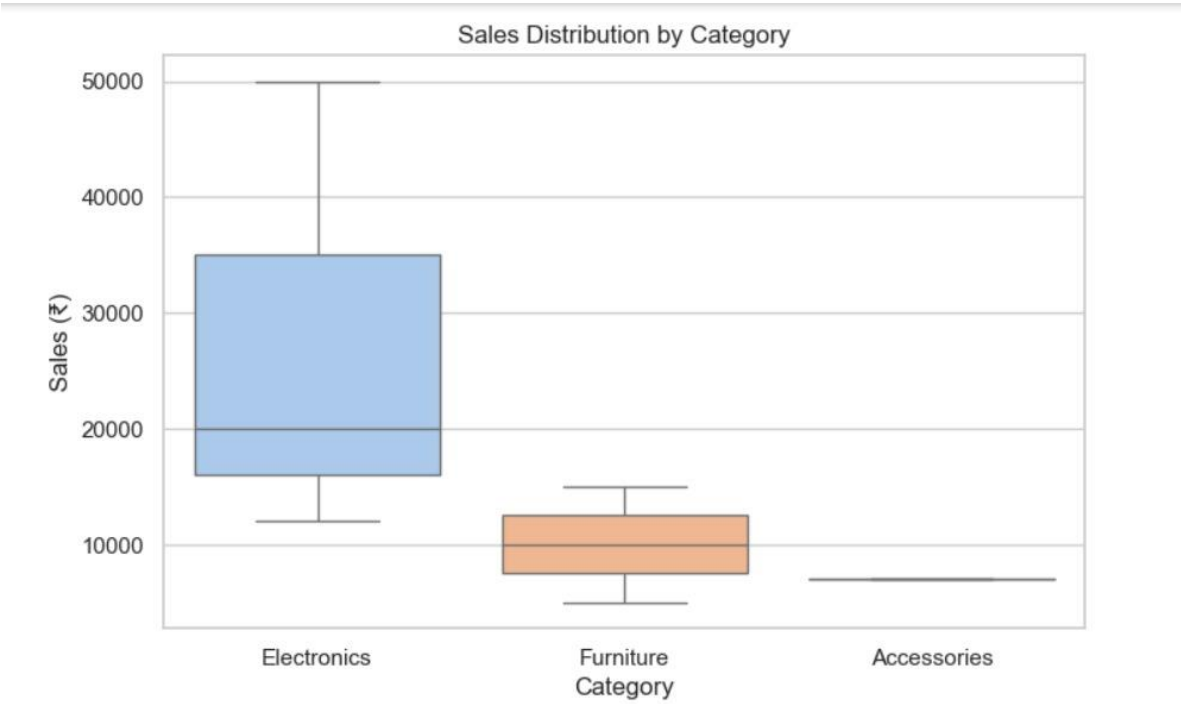
```
plt.figure(figsize=(8, 5))
sns.barplot(x="Category", y="Sales", data=df, estimator=sum, palette="coolwarm")
plt.title("Total Sales by Category")
plt.xlabel("Category")
plt.ylabel("Total Sales (₹)")
plt.show()
```



 # 3. Scatter Plot: Sales vs. Profit
```
plt.figure(figsize=(8, 5))
sns.scatterplot(x="Sales", y="Profit", hue="Category", data=df, s=100, palette="viridis")
plt.title("Sales vs. Profit")
plt.xlabel("Sales (₹)")
plt.ylabel("Profit (₹)")
plt.show()
```

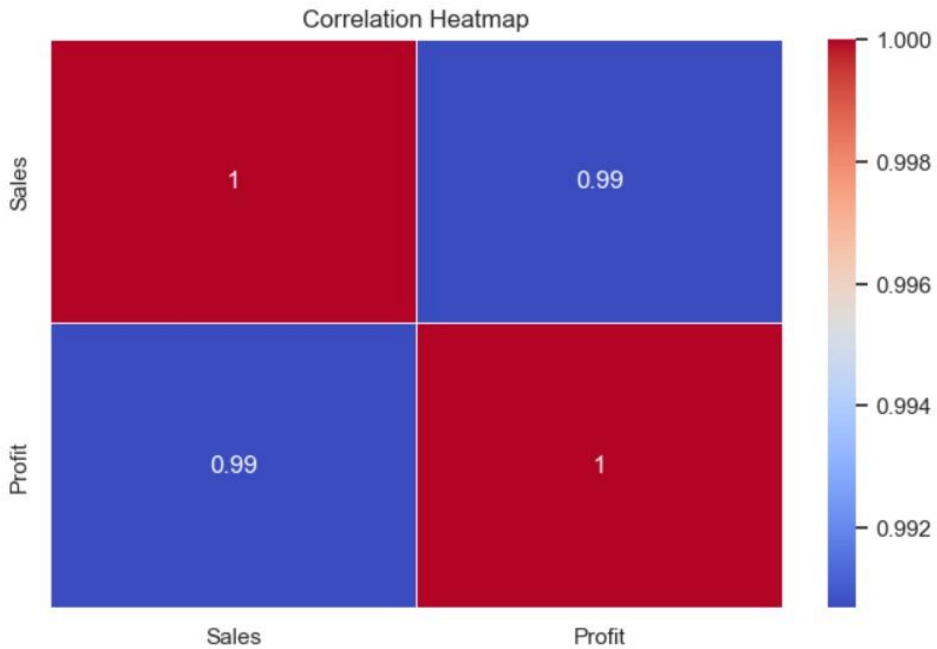


 # 4. Box Plot: Sales Distribution by Category
```
plt.figure(figsize=(8, 5))
sns.boxplot(x="Category", y="Sales", data=df, palette="pastel")
plt.title("Sales Distribution by Category")
plt.xlabel("Category")
plt.ylabel("Sales (₹)")
plt.show()
```



 # 5. Correlation Heatmap

```
plt.figure(figsize=(8, 5))
sns.heatmap(df.select_dtypes(include=["number"]).corr(), annot=True, cmap="coolwarm",
linewidths=0.5)
plt.title("Correlation Heatmap")
plt.show()
```



PRACTICAL NO 9

Perform data visualization using Power BI on the sales data.

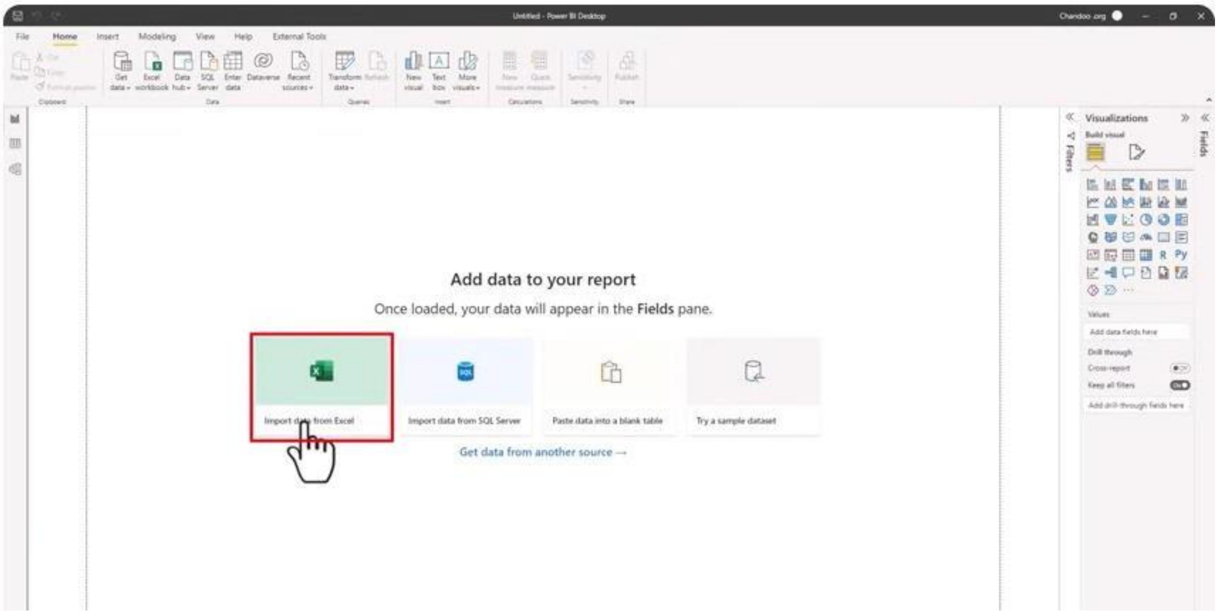
Data visualization is a crucial step in understanding sales data, identifying trends, and making informed business decisions. Power BI is a powerful tool that allows users to create interactive and visually appealing dashboards. This journal documents the step-by-step process of performing data visualization using Power BI on sales data.

Step 1: Understanding the Sales Data

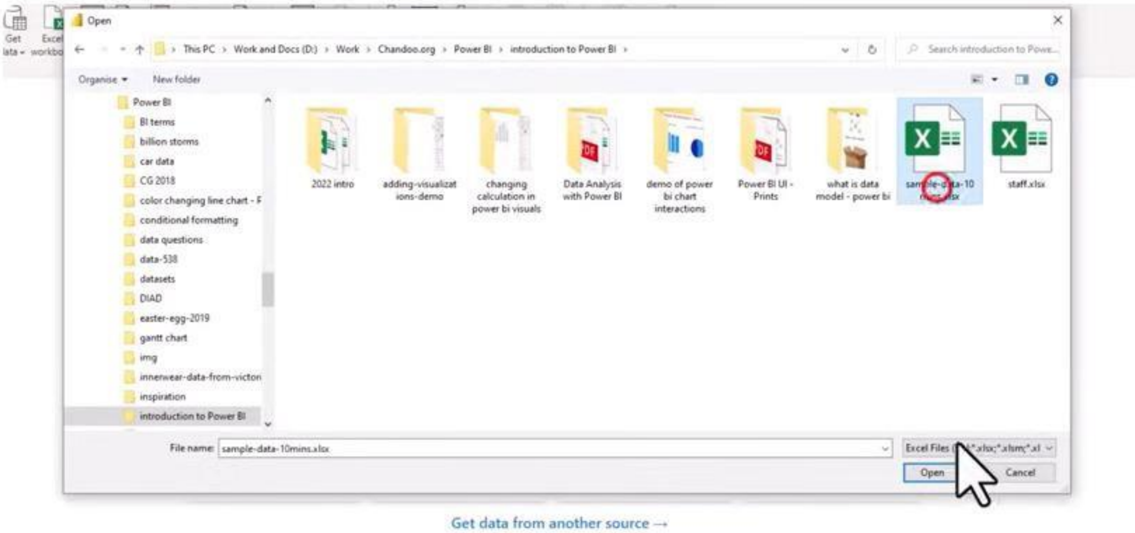
Step 2: Importing Data into Power BI

Open Power BI Desktop.

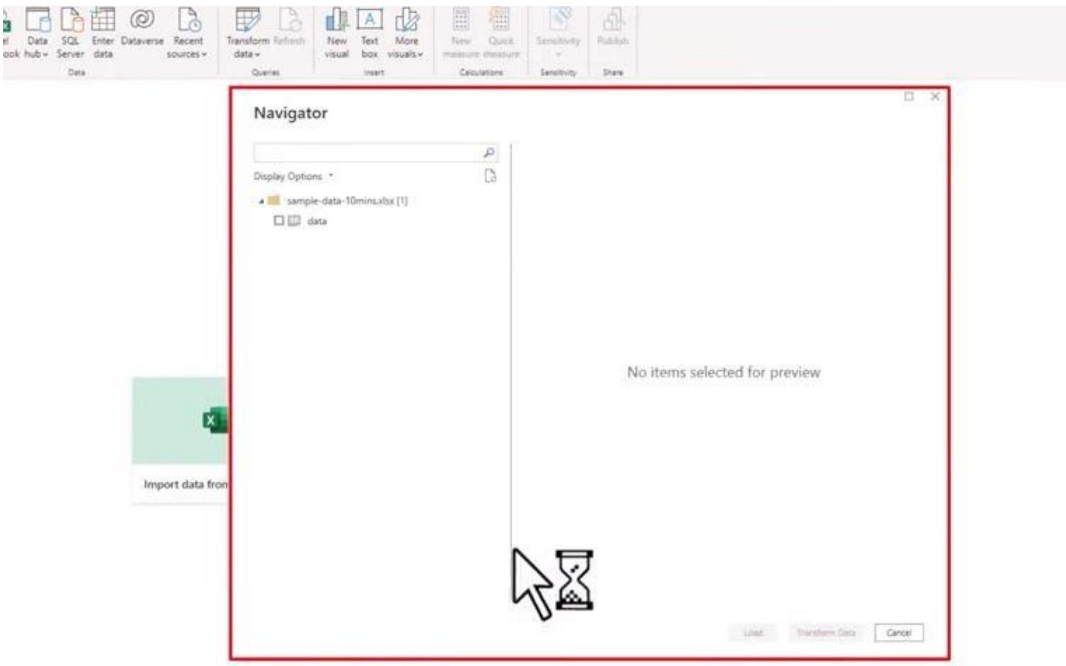
1. Go to power bi and select import data from excel



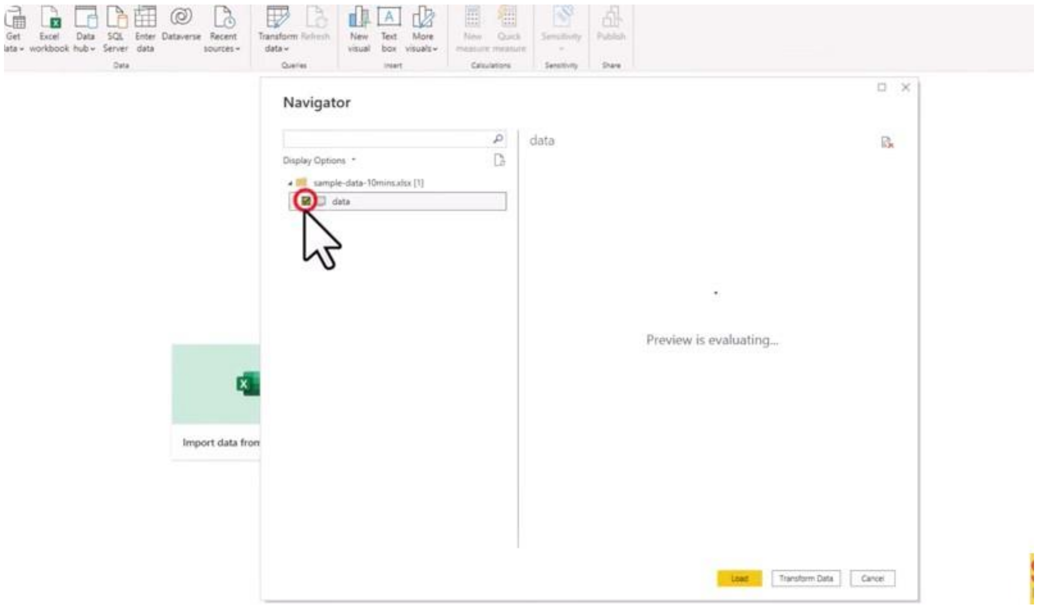
2. Select the excel data file then click on open



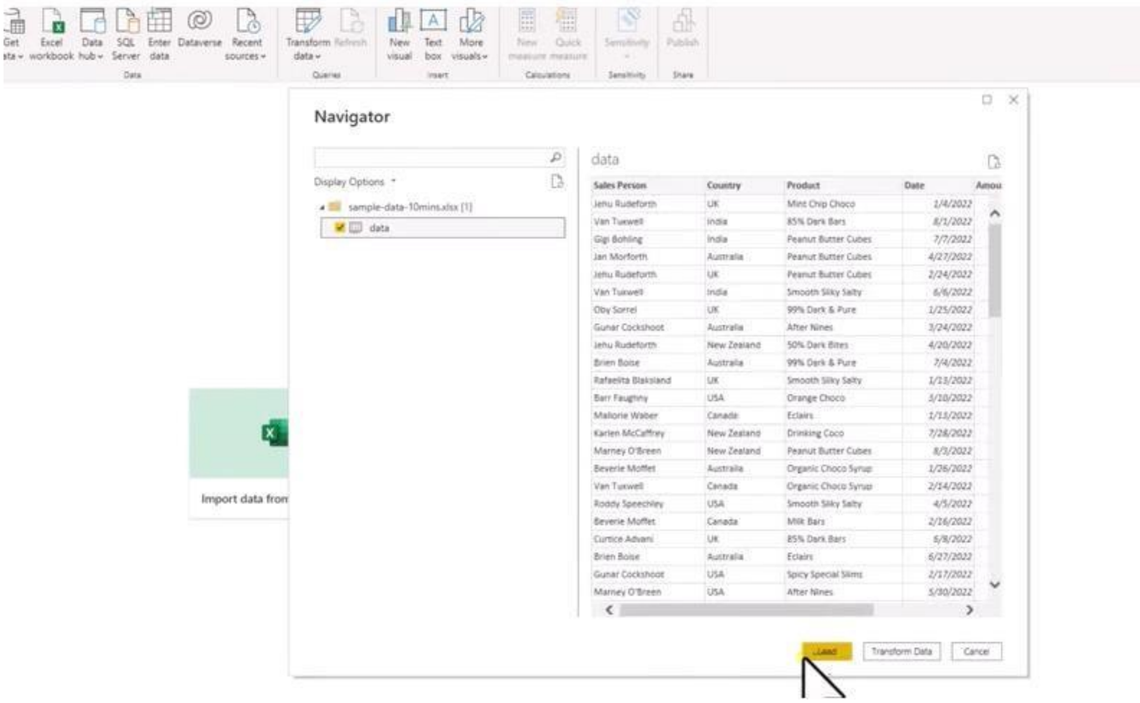
3. Then a navigator screen ask you to select your particular data



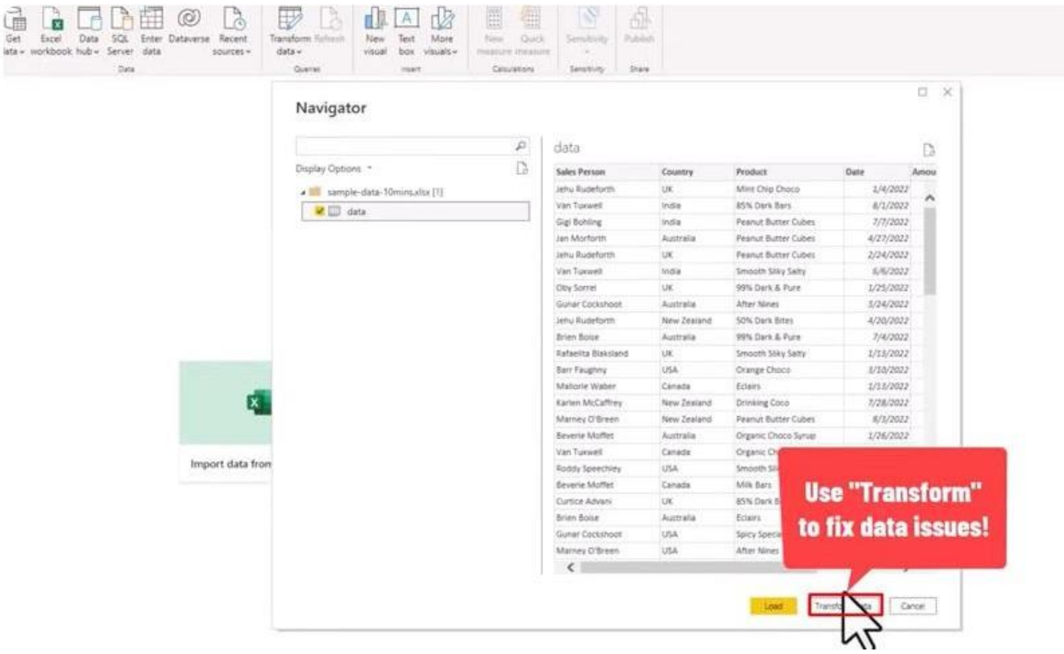
4. Select the particular data



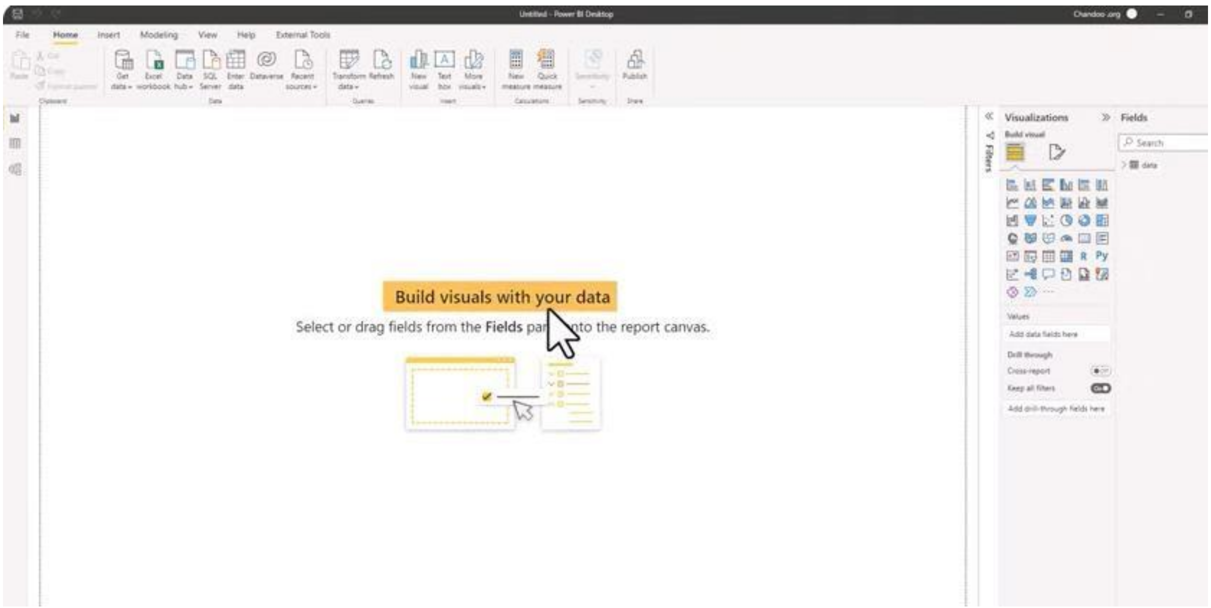
5. Click on load



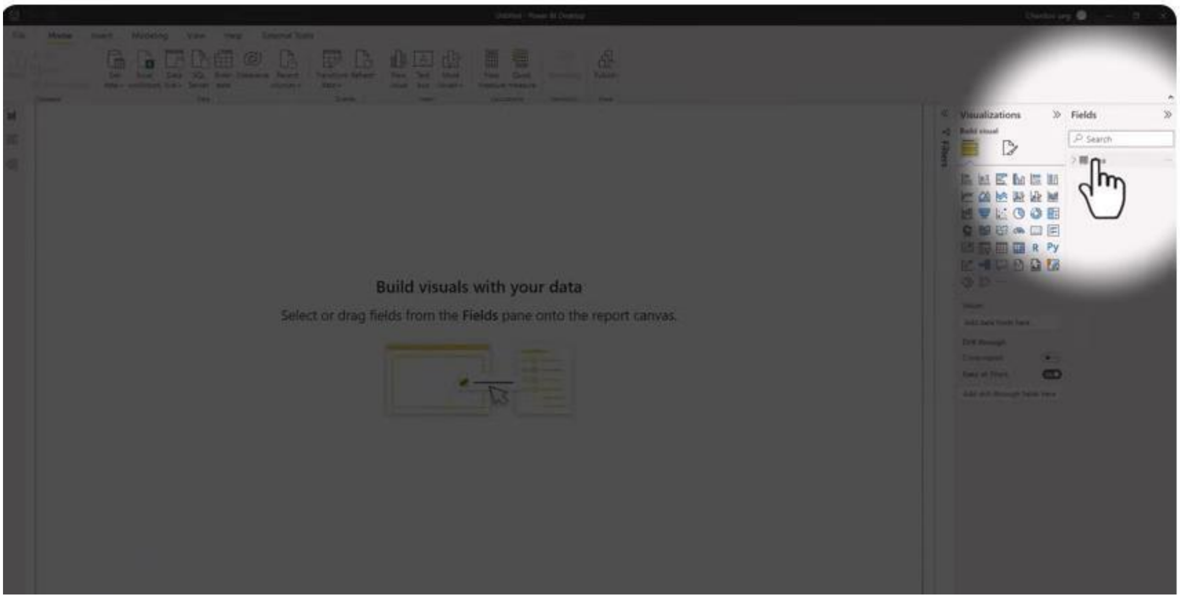
6. If you want to clean or fix your data click on transform data and do the changes and then click on load



7. When you load the data screen will change and look like this now you can build you visuals on your data

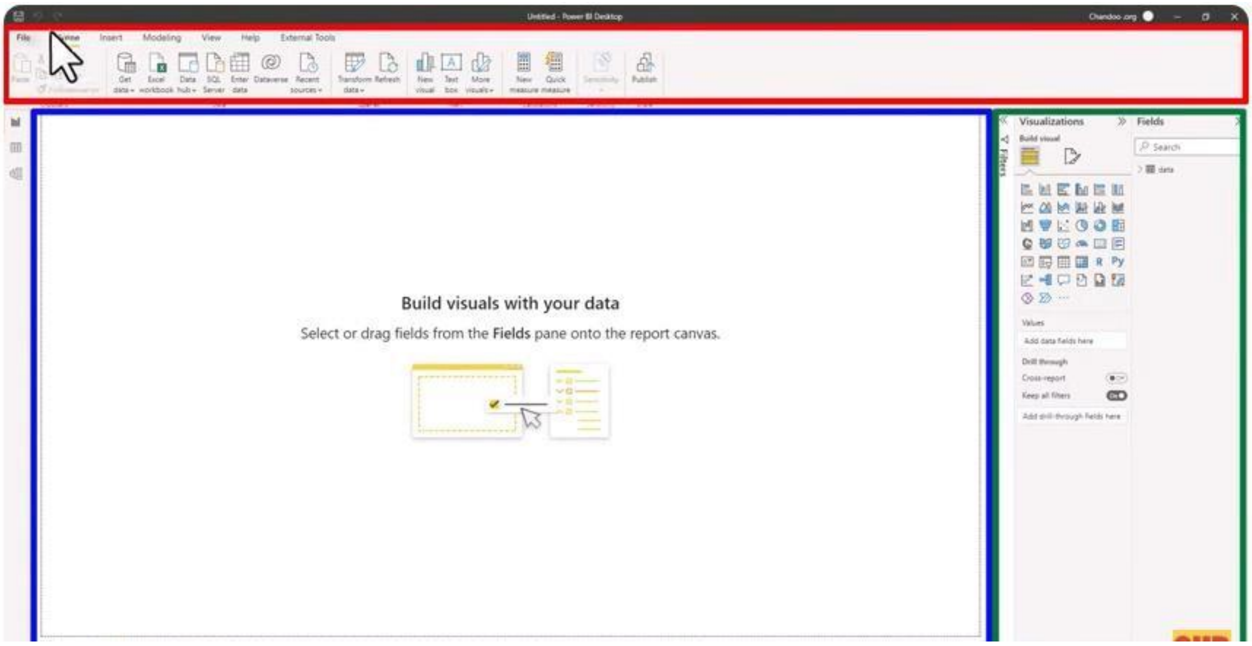


8. Your data will show up in the field panel

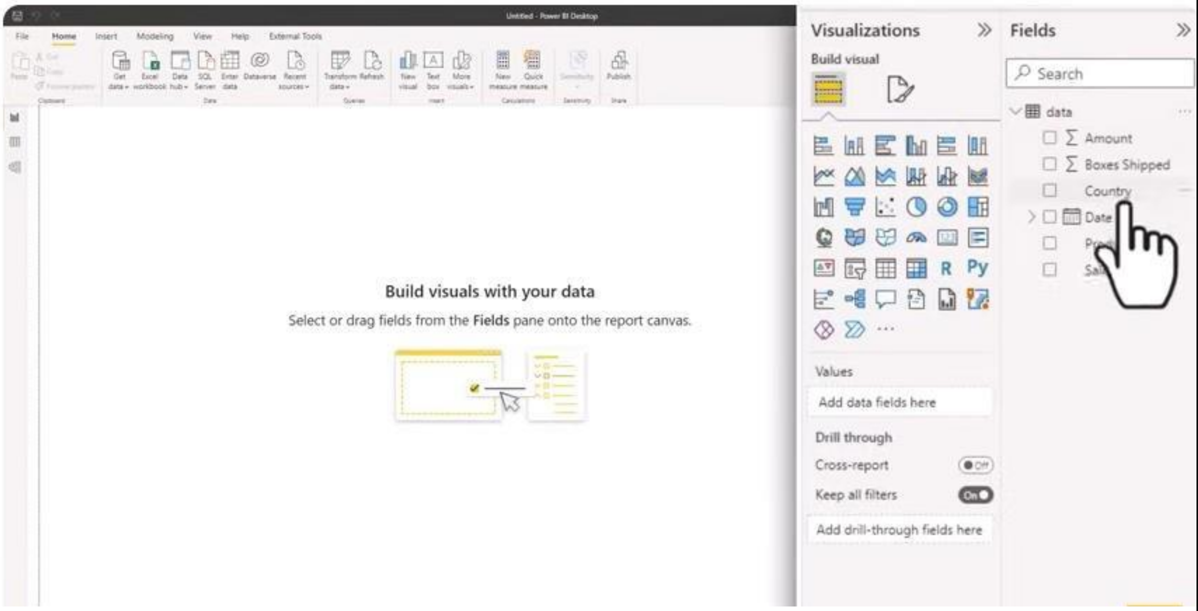


9. Before we move ahead let us get some more details about Power BI: Your Power BI screen in divide in 3 main areas

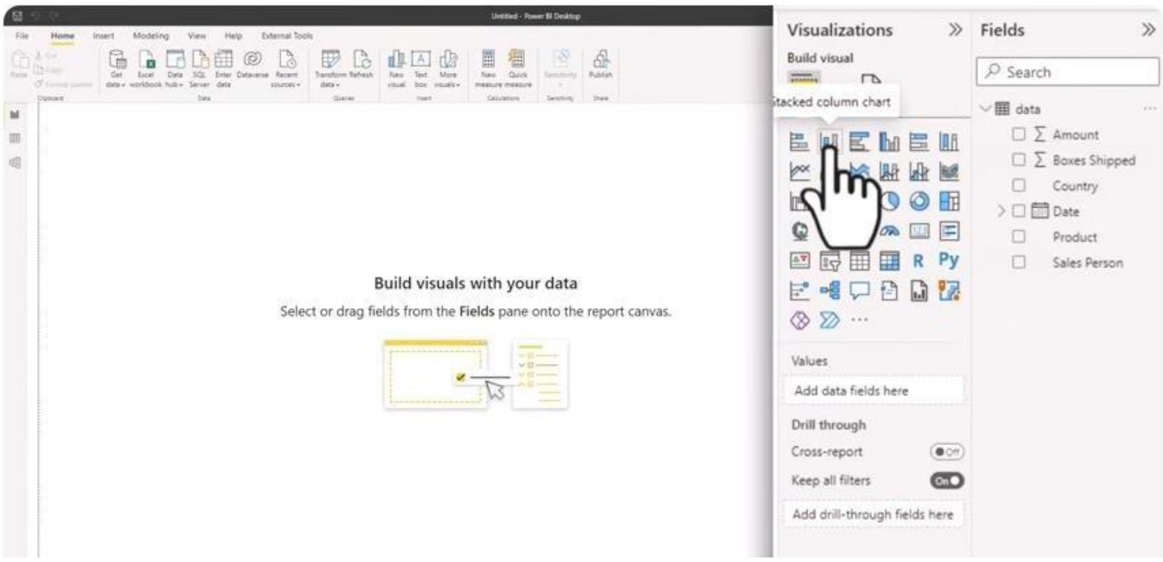
RED: Ribbon (same as excel,word)
BLUE: Canvas where tables or charts appear
GREEN: Panels which is used to build stuffs or change things

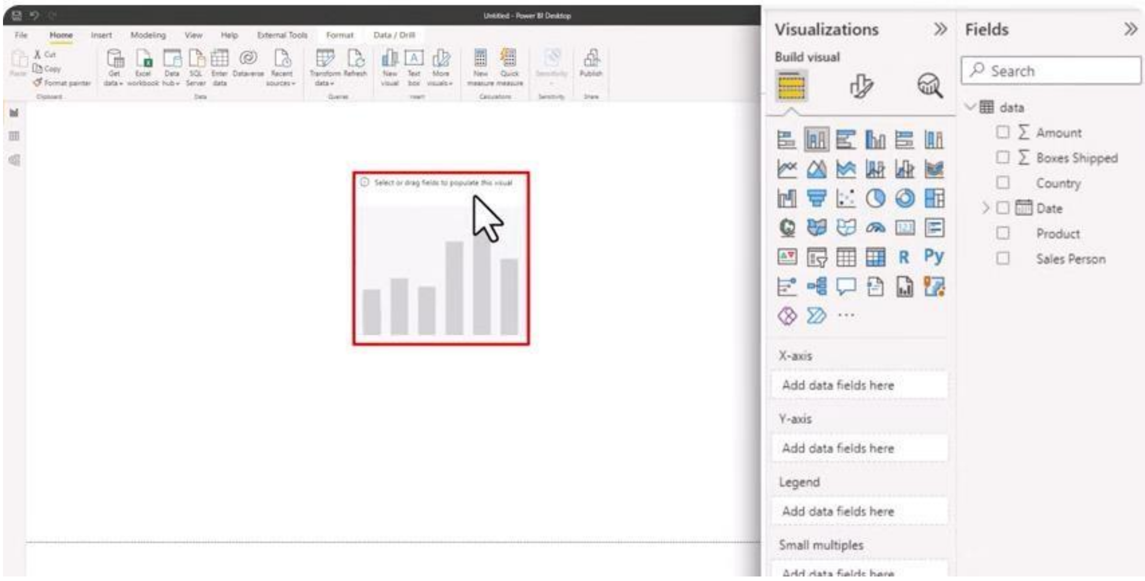


10. Now let's start with the visualization part. When you click on field panel -> data then you can see all the columns present in the data

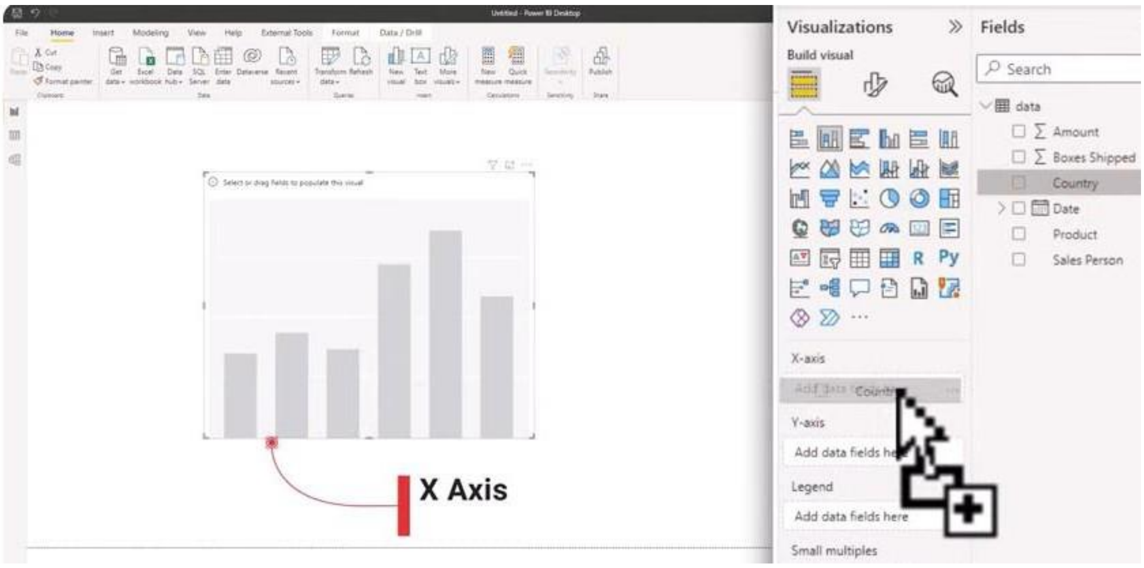


11. Lets see how many boxes are shipped by country : use column chart

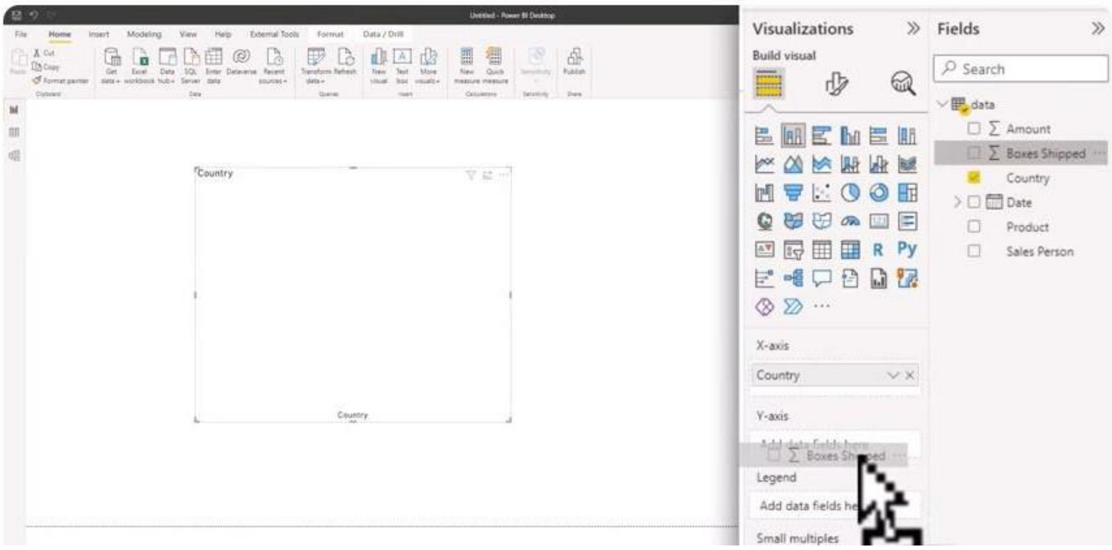




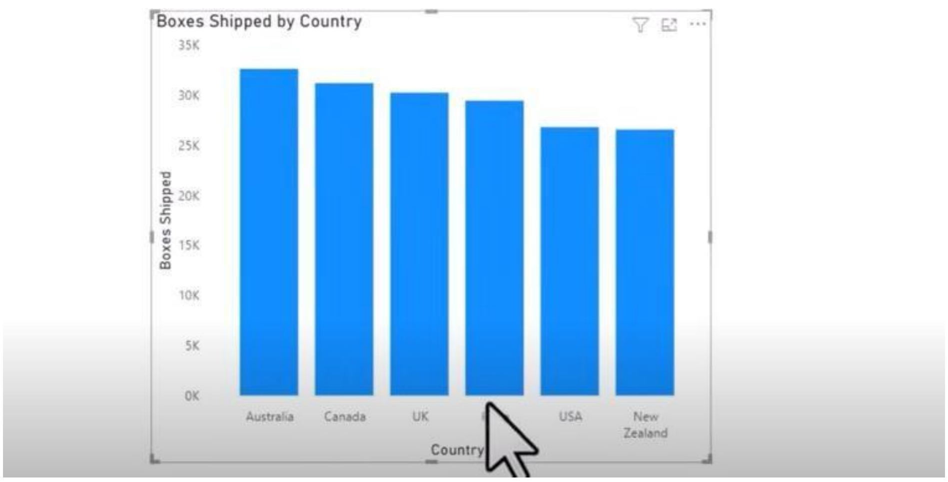
12. On X-axis we will drag and drop country column by field panel



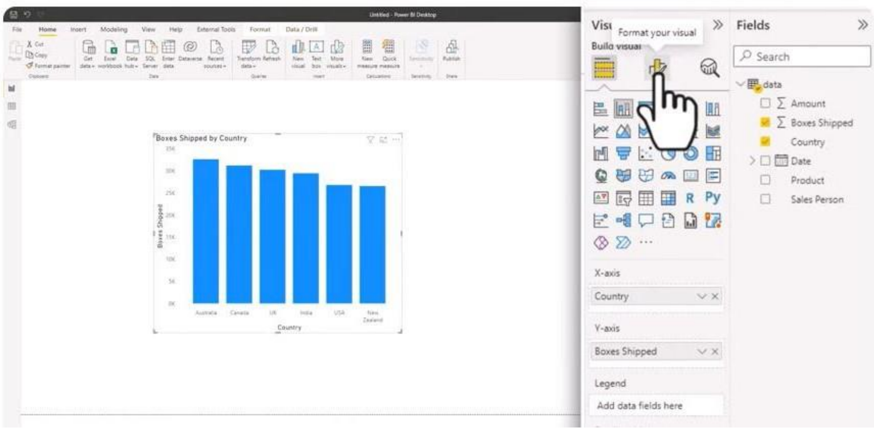
13. On Y-axis we will drag and drop Boxes shipped column from field panel

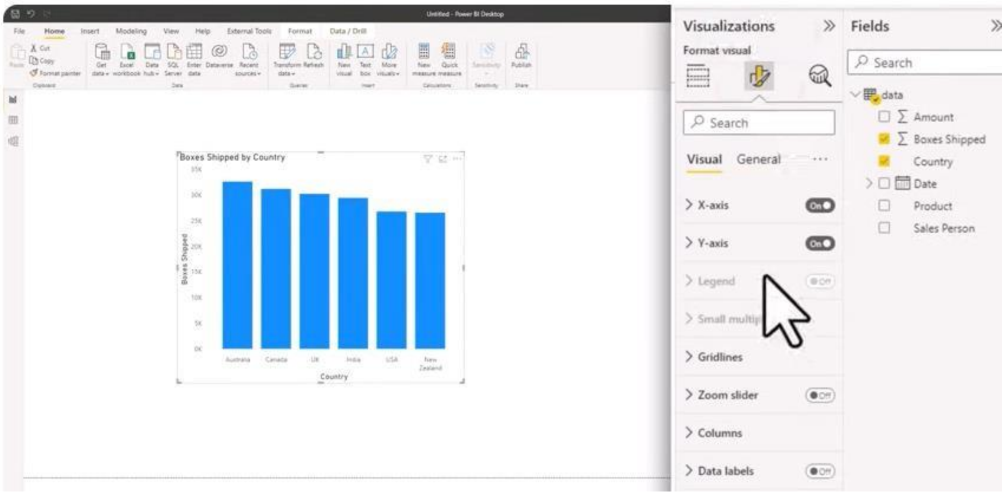


14. And then the visual is created

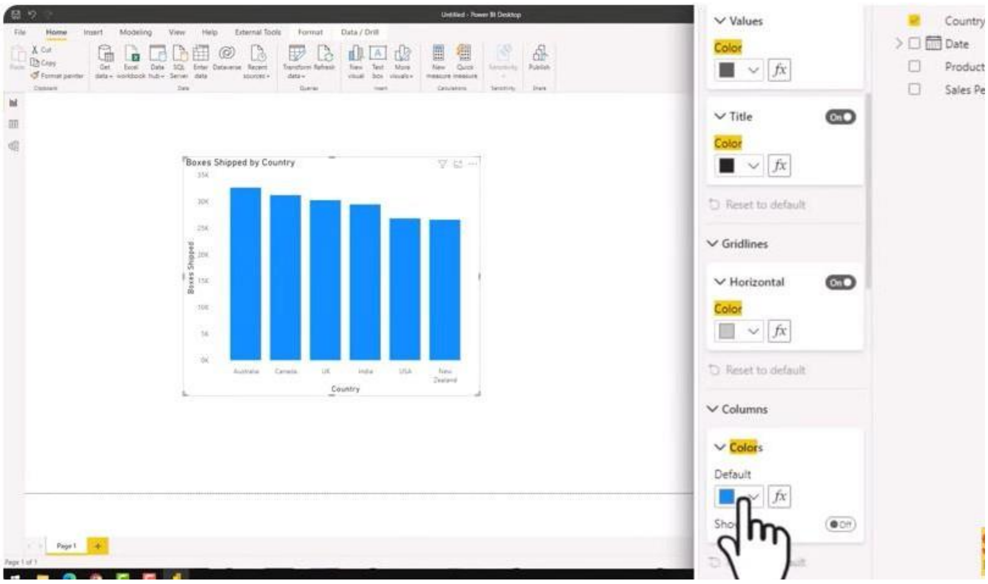


15. If you want to change the colour of the graph we can go to visualization panel and select format your visual option

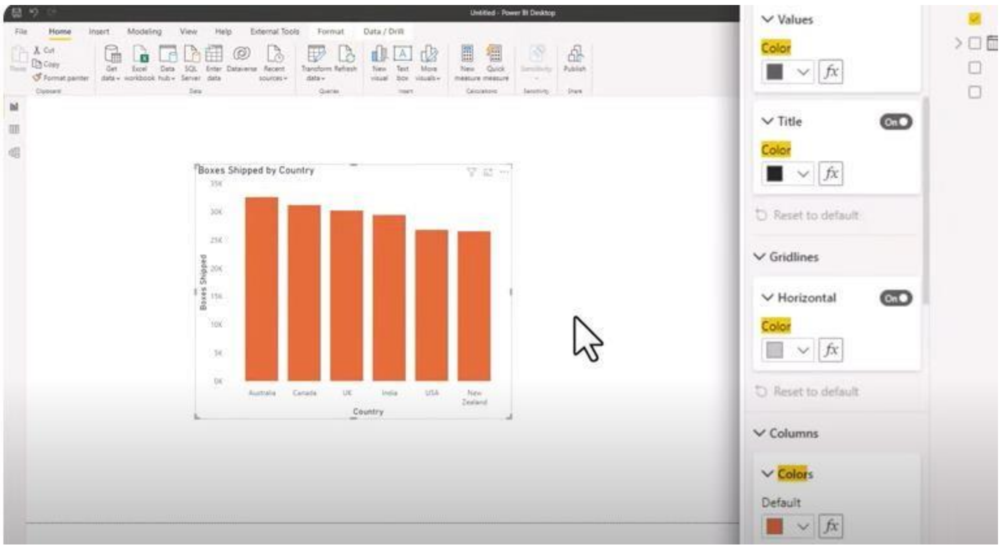




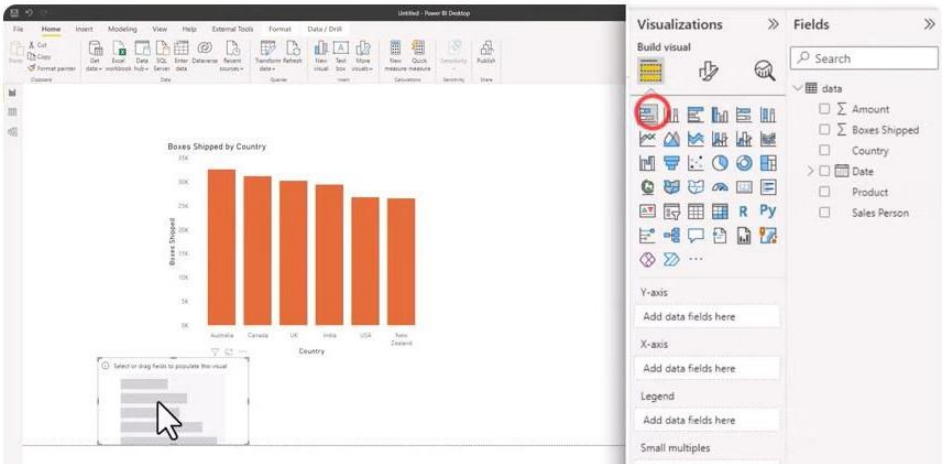
Visualizations
Format visual
Search
Visual
General
X-axis
Y-axis
Legend
Small multip
Gridlines
Zoom slider
Columns
Data labels
Fields
Search
data
Amount
Boxes Shipped
Country
Date
Product
Sales Person
Boxes Shipped by Country
Country

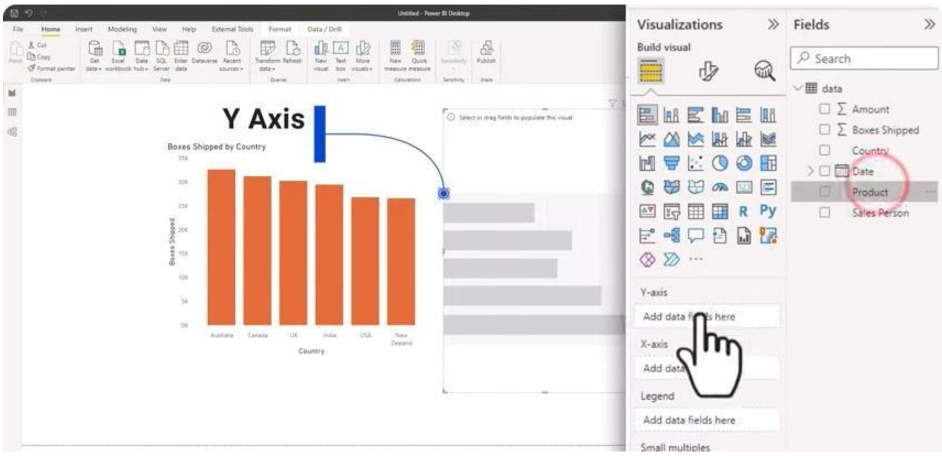


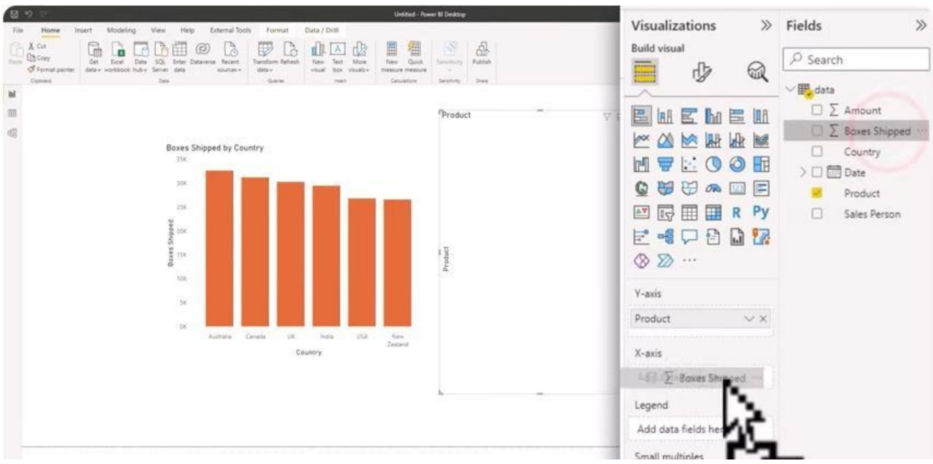
Values
Color
Title
Color
Reset to default
Gridlines
Horizontal
Color
Reset to default
Columns
Colors
Default
Country
Date
Product
Boxes Shipped by Country
Country

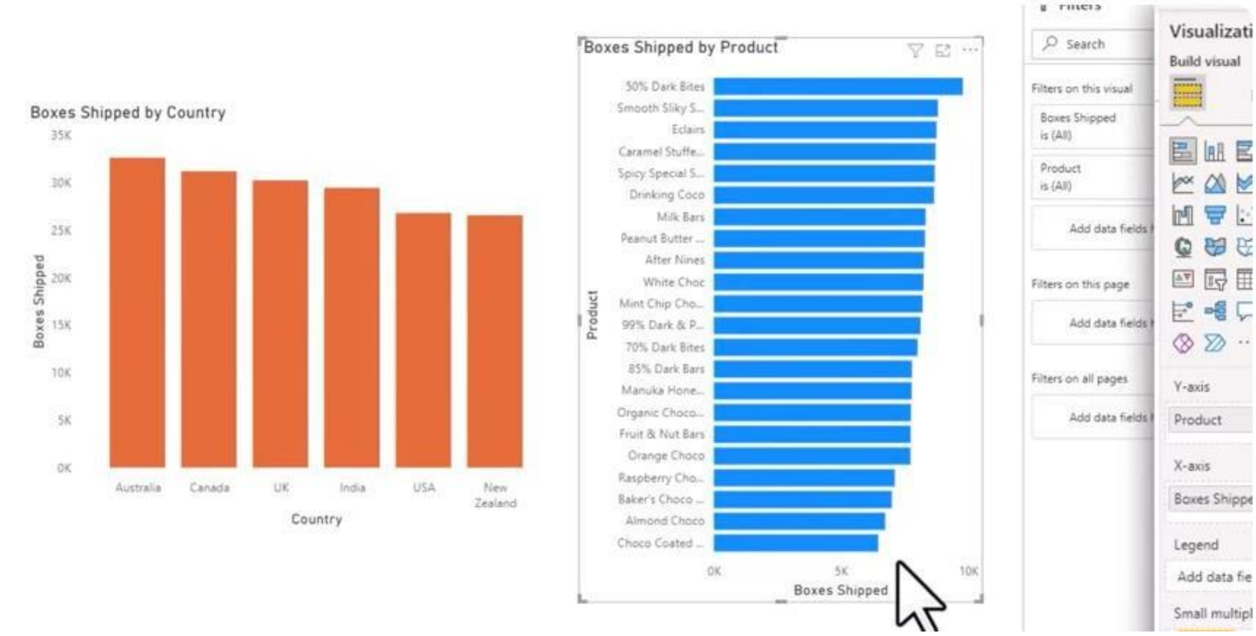


Values
Color
Title
Color
Reset to default
Gridlines
Horizontal
Color
Reset to default
Columns
Colors
Default
Boxes Shipped by Country
Country

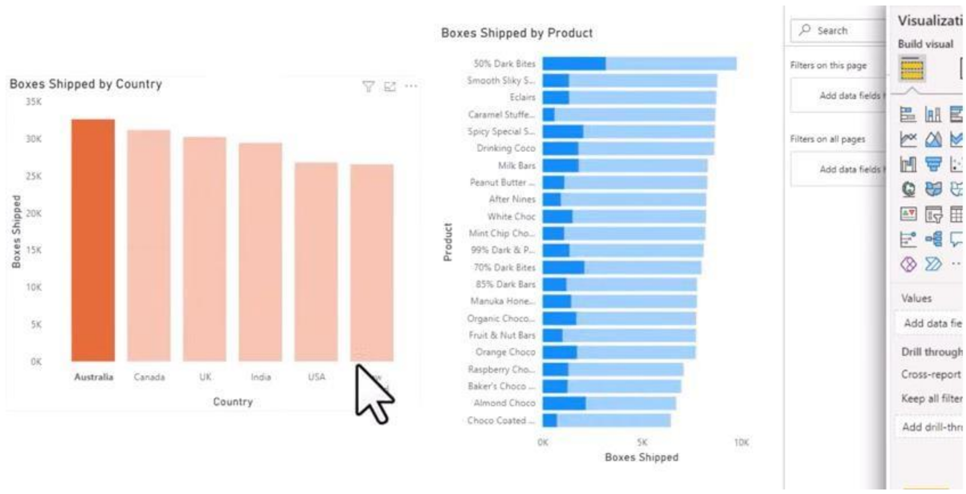

Visualizations
Build visual
Fields
Search
data
Amount
Boxes Shipped
Country
Date
Product
Sales Person
Boxes Shipped by Country
Country
Y-axis
Add data fields here
X-axis
Add data fields here
Legend
Add data fields here
Small multiples

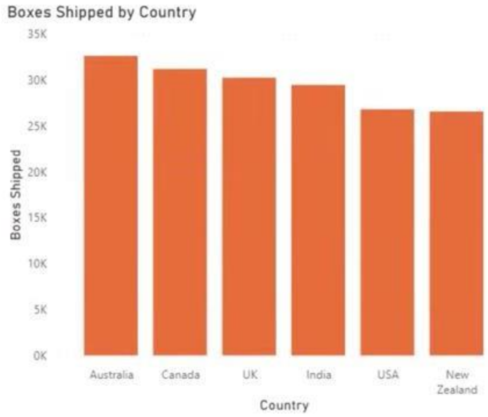

Y Axis
Visualizations
Build visual
Fields
Search
data
Amount
Boxes Shipped
Country
Date
Product
Sales Person
Boxes Shipped by Country
Country
Y-axis
X-axis
Legend
Add data fields here
Small multiples

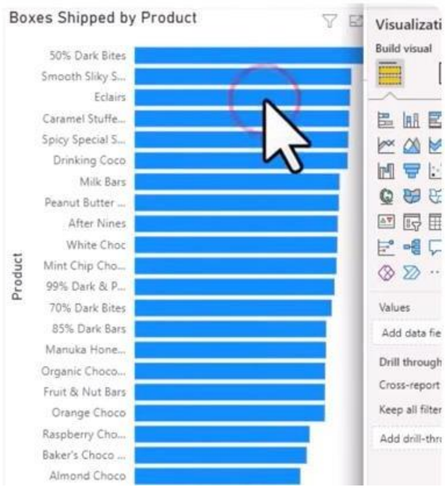

Visualizations
Build visual
Fields
Search
data
Amount
Boxes Shipped
Country
Date
Product
Sales Person
Product
Boxes Shipped by Country
Country
Y-axis
Product
X-axis
Legend
Small multiples

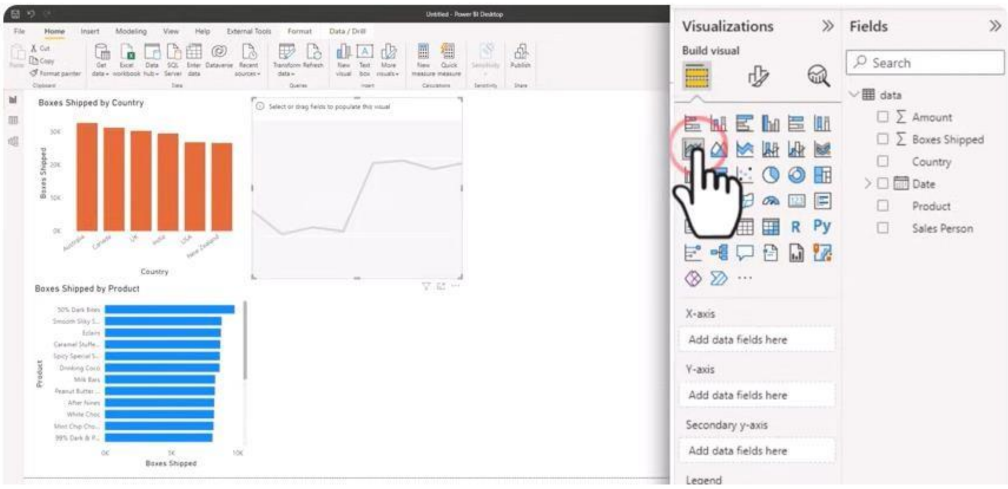


If want to cheeck a specific counytry click on that bar

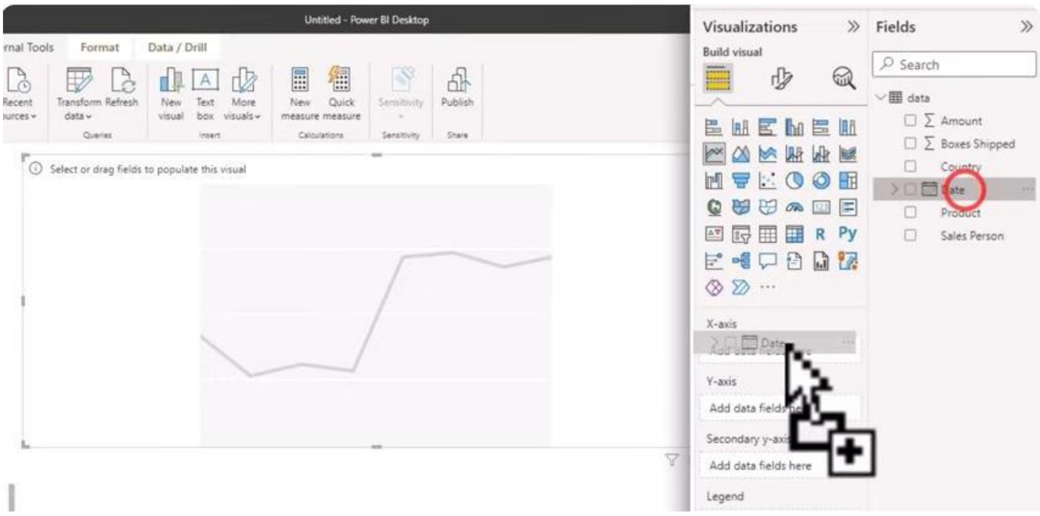


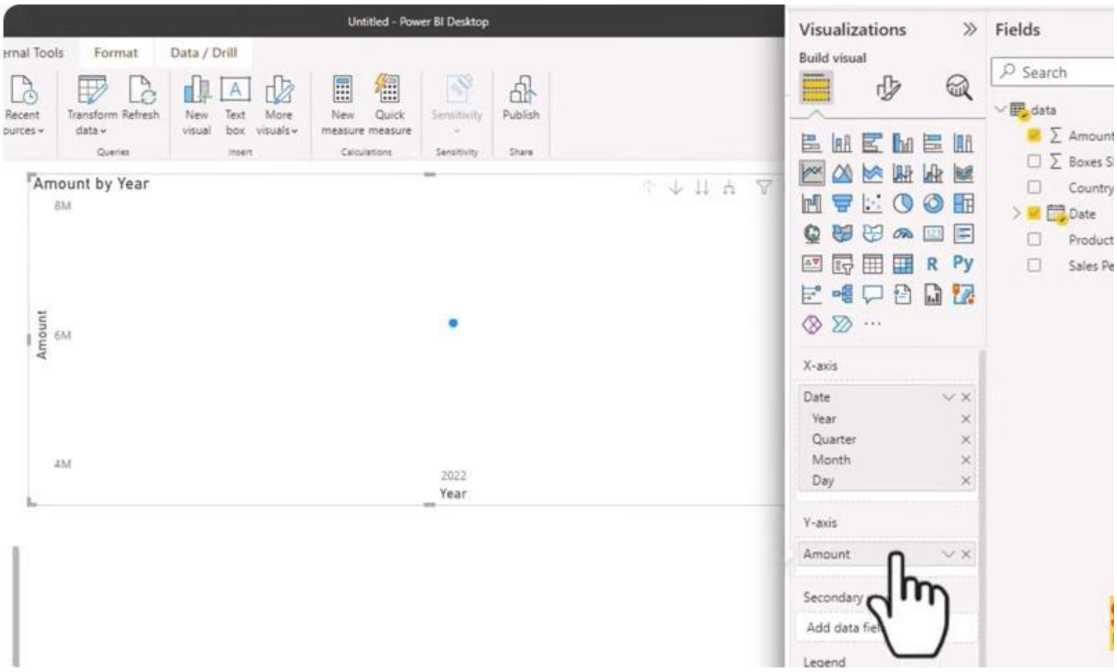

Boxes Shipped by Country
35K
30K
25K
20K
15K
10K
5K
0K
Boxes Shipped
Australia
Canada
UK
India
USA
New Zealand
Country


Boxes Shipped by Product
50% Dark Bites
Smooth Sliky S...
Eclairs
Caramel Stuffe...
Spicy Special S...
Drinking Coco
Milk Bars
Peanut Butter ...
After Nines
White Choc
Mint Chip Cho...
99% Dark & P...
70% Dark Bites
85% Dark Bars
Manuka Hone...
Organic Choco...
Fruit & Nut Bars
Orange Choco
Raspberry Cho...
Baker's Choco ...
Almond Choco
Product
Build visual
Values
Drill through
Cross-report
Keep all filter

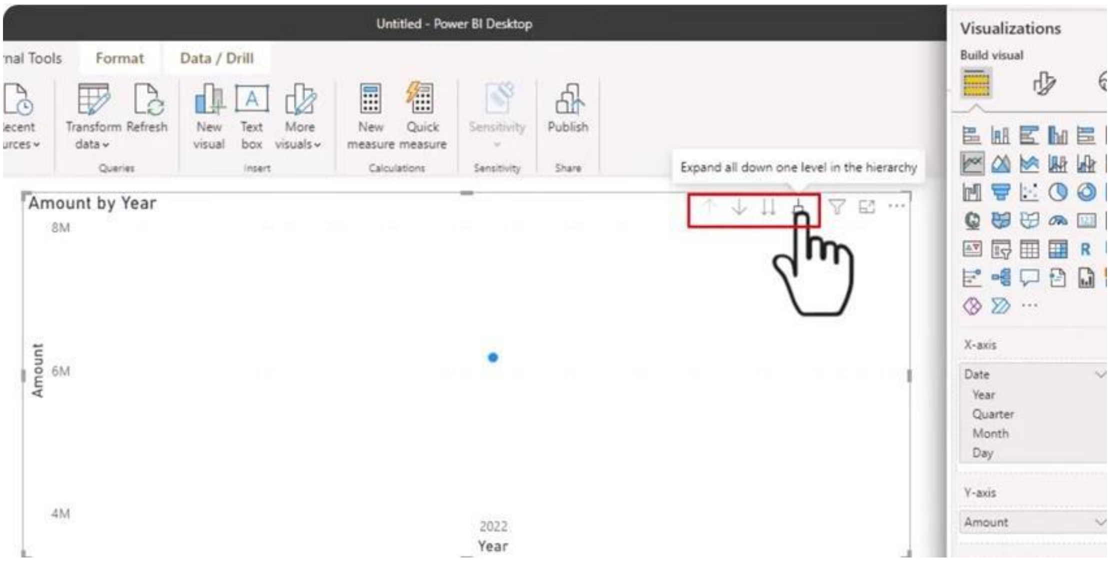

Visualizations
Build visual
Fields
Search
data
Amount
Boxes Shipped
Country
Date
Product
Sales Person
X-axis
Add data fields here
Y-axis
Add data fields here
Secondary y-axis
Add data fields here
Legend
Boxes Shipped by Country
Boxes Shipped by Product

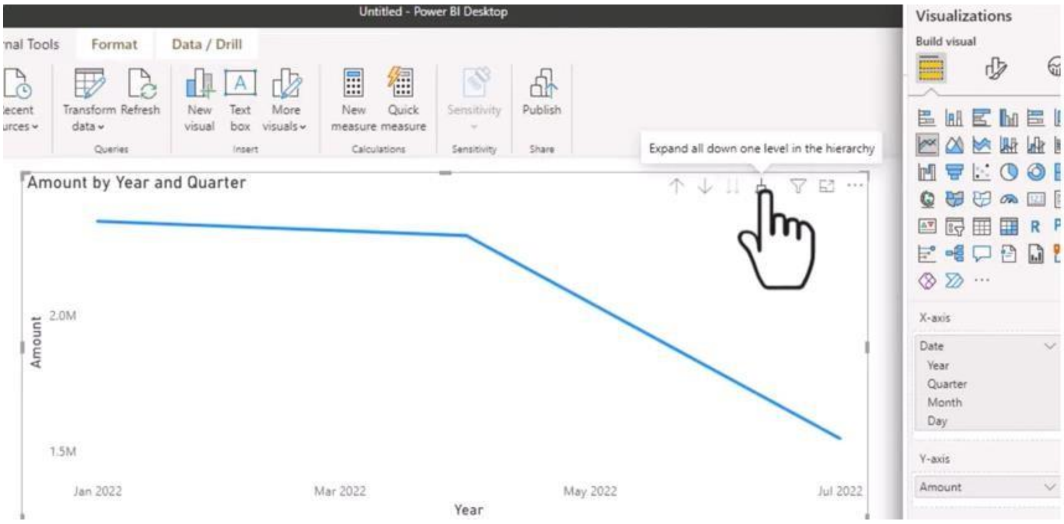
Line graph:


Untitled - Power BI Desktop
Format
Data / Drill
Transform data
Refresh
New visual
Text box
More visuals
New measure
Quick measure
Sensitivity
Publish
Queries
Insert
Calculations
Share
Select or drag fields to populate this visual
Visualizations
Build visual
Fields
Search
data
Amount
Boxes Shipped
Country
Date
Product
Sales Person
X-axis
Y-axis
Secondary y-axis
Add data fields here
Legend

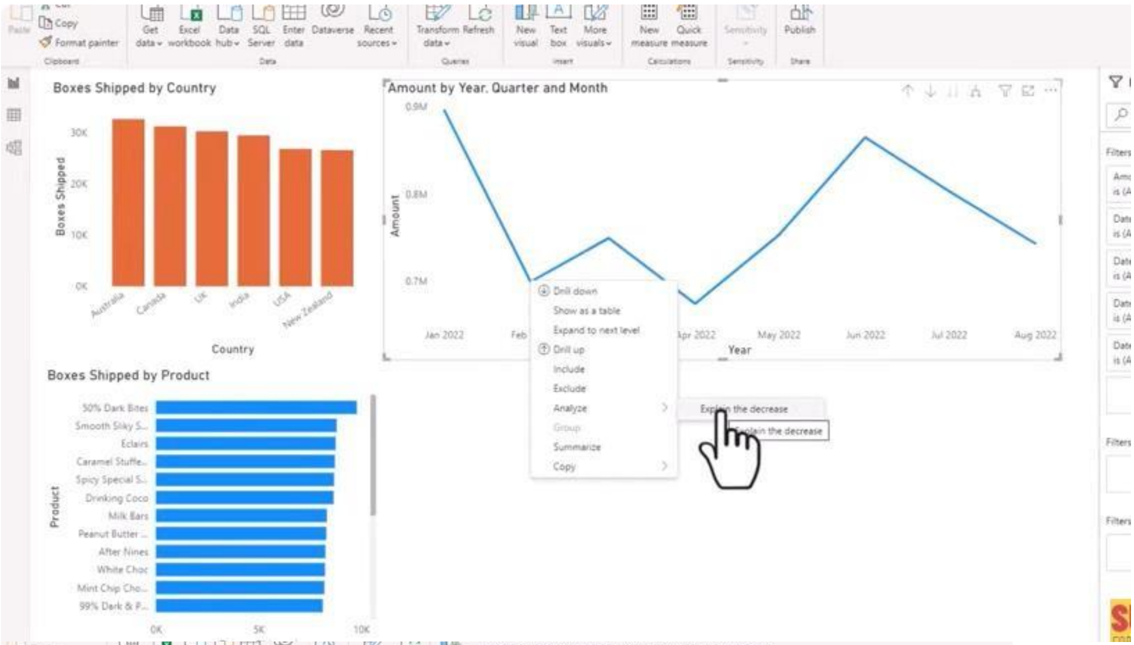


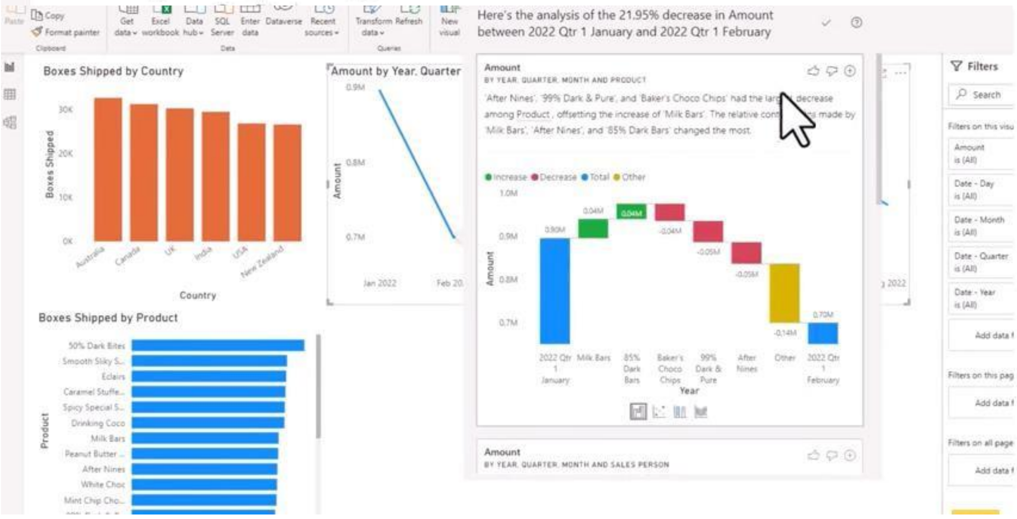
To segregate graph further by months we will follow the below steps

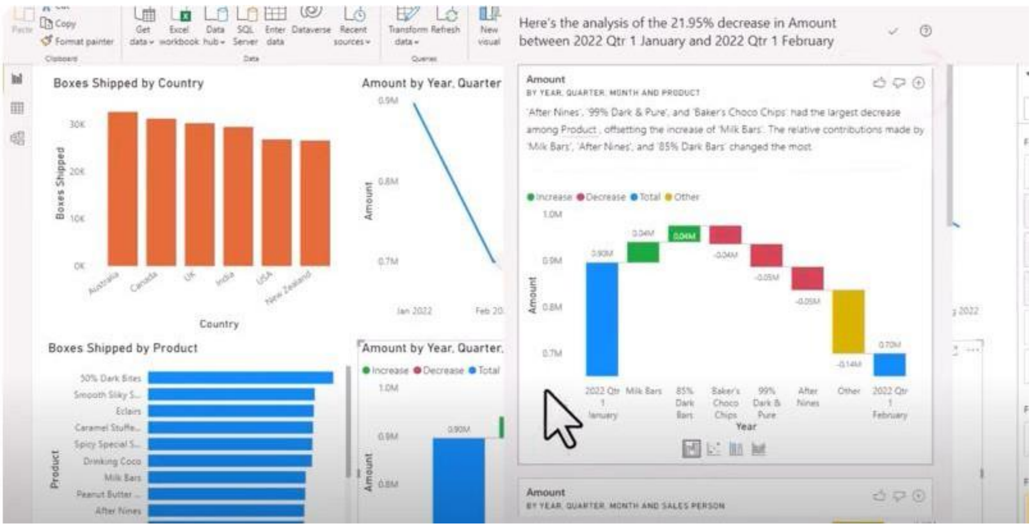


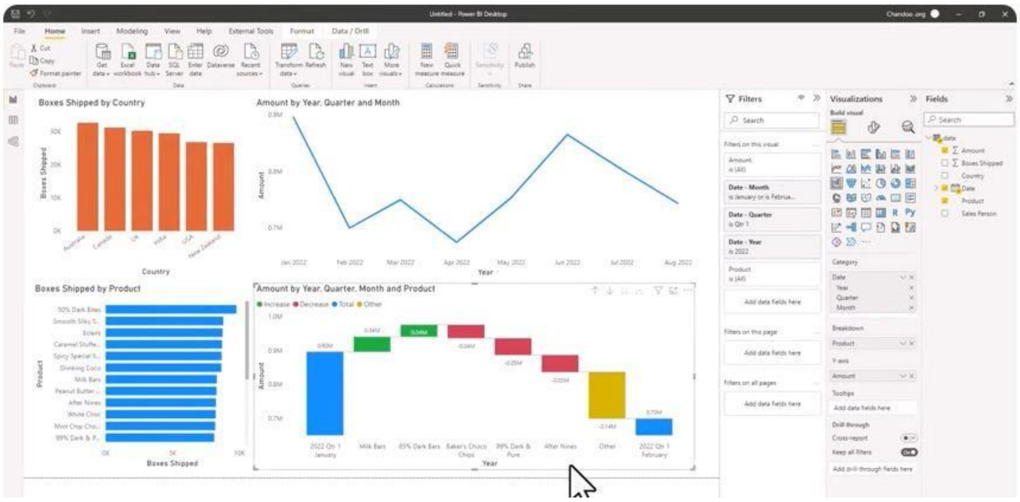


To collect more insights right click on the points of line and follow the steps


Boxes Shipped by Country
Amount by Year, Quarter and Month
Boxes Shipped by Product
Drill down
Show as a table
Expand to next level
Drill up
Include
Exclude
Analyze
Group
Summarize
Copy
Explain the decrease


Here's the analysis of the 21.95% decrease in Amount between 2022 Qtr 1 January and 2022 Qtr 1 February
Boxes Shipped by Country
Amount by Year, Quarter
Boxes Shipped by Product
Amount
BY YEAR, QUARTER, MONTH AND PRODUCT
'After Nines', '99% Dark & Pure', and 'Baker's Choco Chips' had the largest decrease among Product, offsetting the increase of 'Milk Bars'. The relative contributions made by 'Milk Bars', 'After Nines', and '85% Dark Bars' changed the most.
Increase Decrease Total Other
Filters


Here's the analysis of the 21.95% decrease in Amount between 2022 Qtr 1 January and 2022 Qtr 1 February
Boxes Shipped by Country
Amount by Year, Quarter
Boxes Shipped by Product
Amount
BY YEAR, QUARTER, MONTH AND PRODUCT
'After Nines', '99% Dark & Pure', and 'Baker's Choco Chips' had the largest decrease among Product, offsetting the increase of 'Milk Bars'. The relative contributions made by 'Milk Bars', 'After Nines', and '85% Dark Bars' changed the most.
Increase Decrease Total Other
Amount
BY YEAR, QUARTER, MONTH AND SALES PERSON



Boxes Shipped by Country
Amount by Year, Quarter and Month
Boxes Shipped by Product
Amount by Year, Quarter, Month and Product
Filters
Visualizations
Fields