UNIVERSITY OF MUMBAI

DEPARTMENT OF COMPUTER SCIENCE

M.Sc. Computer Science – Semester III

Track D: Data Science

Elective I: Data Visualization

JOURNAL

2022-2023

Seat No. __________________

मुंबई विद्यापीठ
University of Mumbai
Re-accredited with A++ Grade
(CGPA 3.65) by NAAC (3rd Cycle 2021)

UNIVERSITY OF MUMBAI

DEPARTMENT OF COMPUTER SCIENCE

CERTIFICATE

This is to certify that the work entered in this journal was done in the University Department of Computer Science laboratory by Mr./Ms.____ Seat No. ________________ for the course of M.Sc. Computer Science - Semester III (CBCS) (Revised) during the academic year 2022-2023 in a satisfactory manner.

______________ ______________

Subject In-charge **Head of Department**

____

External Examiner

Index

PRACTICAL -1

Aim: Create one-dimensional data using series and perform various operations on it.

Theory: Panda series is a one-dimensional labeled array capable of holding data of any type (integer, string, float, python objects, etc.). The axis labels are collectively called *index*. Labels need not be unique but must be a hashable type. The object supports both integer and label-based indexing and provides a host of methods for performing operations involving the index. To create Series with any of the methods make sure to import pandas library.

Creating a series from Scalar value: In order to create a series from scalar value, an index must be provided. The scalar value will be repeated to match the length of the index.

Code:

```
import pandas as pd
d1 = pd.Series([33,44,55,66], index=['A','B','C','D'])
d2 = pd.Series([34,45,56,67], index=['A','B','D','E'])

# printing of data 1
print(d1)

# printing of data 2
print(d2)

# addition of data 1 and data 2
d1.add(d2, fill_value=0)

# subtraction of data1 by data 2
d1.sub(d2, fill_value=0)

# multiplication of data 1 and data 2
d1.mul(d2)

# division of data1 by data 2
d1.div(d2)

# power of data1 to the power data2
d1.pow(d2)
```

Output:-

```
A    33
B    44
C    55
```

```
D    66
dtype: int64
```

```
A    34
B    45
D    56
E    67
dtype: int64
```

```
A 67.0
B 89.0
C 55.0
D 122.0
E 67.0
dtype: float64
```

```
A -1.0
B -1.0
C 55.0
D 10.0
E -67.0
dtype: float64
```

```
A 1122.0
B 1980.0
C NaN
D 3696.0
E NaN
dtype: float64
```

```
A 0.970588
B 0.977778
C NaN
D 1.178571
E NaN
dtype: float64
```

```
A 4.260631e+51
B 9.023405e+73
C NaN
D 7.842606e+101
E NaN
dtype: float64
```

PRACTICAL- 2

Aim: Create a Two dimensional data with the help of dataframe and perform different operation on it.

Theory: Python is a great language for doing data analysis, primarily because of the fantastic ecosystem of data-centric Python packages. Pandas is one of those packages and makes importing and analyzing data much easier.

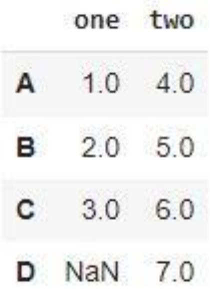
Pandas Dataframe can be achieved in multiple ways. In this article, we will learn how to create a dataframe using two-dimensional List.

Code:

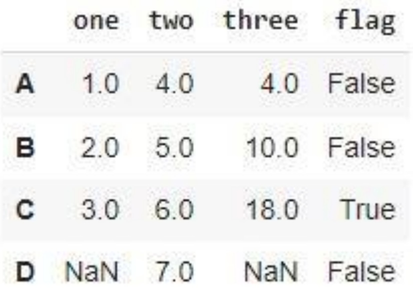
```
import pandas as pd
dictA={
  'one':pd.Series([1. ,2. ,3.], index=['A', 'B', 'C']) ,
  'two':pd.Series([4., 5., 6., 7.], index=['A', 'B', 'C', 'D'])
}

df=pd.DataFrame(dictA)
df


#Add column
df['three'] = df['one'] * df['two']
df['flag'] = df['one'] > 2
df


#Adding a Column Using a Scalar and Assigning to a Data Frame
df['Filler'] = 'HCT'
df['Slic'] = df['one'][:2]
df


# delete columns
del df['one']
df


# another del command
three=df.pop('three')
df

#insert value
```

```
df.insert(1, 'bar', df['two'])
df

#missing value
# returns true for missing values
df.isna()


#summary
df.describe()
```

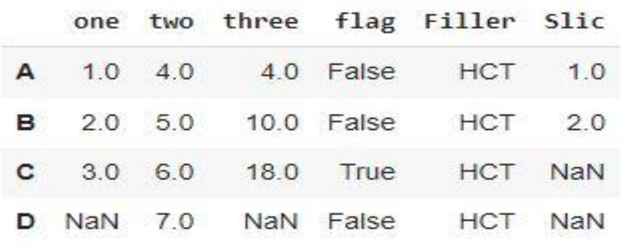
Output:-



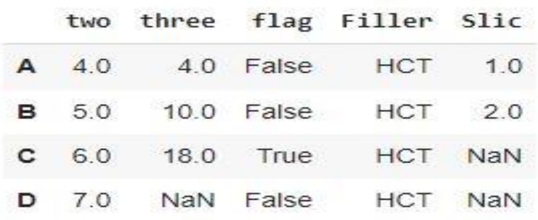
	one	two
A	1.0	4.0
B	2.0	5.0
C	3.0	6.0
D	NaN	7.0



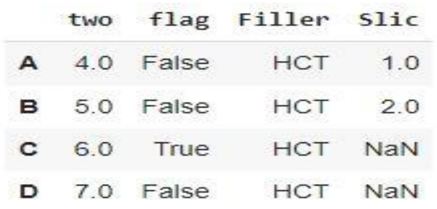
	one	two	three	flag
A	1.0	4.0	4.0	False
B	2.0	5.0	10.0	False
C	3.0	6.0	18.0	True
D	NaN	7.0	NaN	False



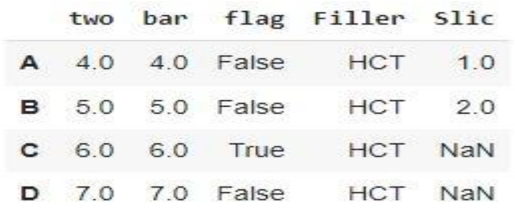
	one	two	three	flag	Filler	Slic
A	1.0	4.0	4.0	False	HCT	1.0
B	2.0	5.0	10.0	False	HCT	2.0
C	3.0	6.0	18.0	True	HCT	NaN
D	NaN	7.0	NaN	False	HCT	NaN



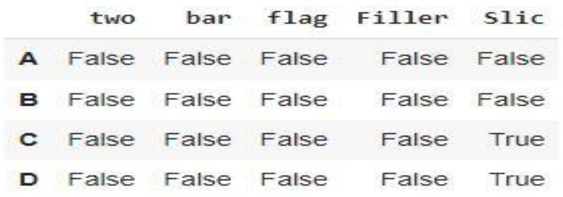
	two	three	flag	Filler	Slic
A	4.0	4.0	False	HCT	1.0
B	5.0	10.0	False	HCT	2.0
C	6.0	18.0	True	HCT	NaN
D	7.0	NaN	False	HCT	NaN



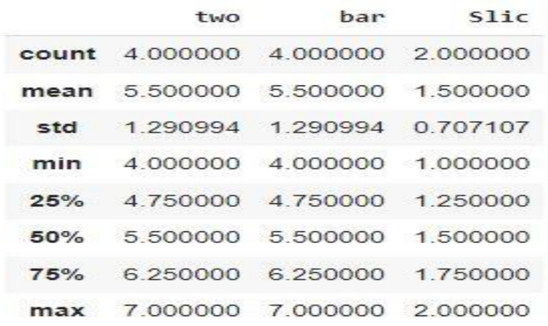
	two	flag	Filler	Slic
A	4.0	False	HCT	1.0
B	5.0	False	HCT	2.0
C	6.0	True	HCT	NaN
D	7.0	False	HCT	NaN



	two	bar	flag	Filler	Slic
A	4.0	4.0	False	HCT	1.0
B	5.0	5.0	False	HCT	2.0
C	6.0	6.0	True	HCT	NaN
D	7.0	7.0	False	HCT	NaN



	two	bar	flag	Filler	Slic
A	False	False	False	False	False
B	False	False	False	False	False
C	False	False	False	False	True
D	False	False	False	False	True



	two	bar	Slic
count	4.000000	4.000000	2.000000
mean	5.500000	5.500000	1.500000
std	1.290994	1.290994	0.707107
min	4.000000	4.000000	1.000000
25%	4.750000	4.750000	1.250000
50%	5.500000	5.500000	1.500000
75%	6.250000	6.250000	1.750000
max	7.000000	7.000000	2.000000

PRACTICAL – 3

Aim: Write a code to read data from the different file formats like JSON, HTML, XML, and CSV files and check for missing data and outlier values and handle them.

Theory:

- Different file format

A CSV (or Comma Separated Value) file is the most common type of file that a data scientist will ever work with. These files use a "," as a delimiter to separate the values and each row in a CSV file is a data record.
XML (eXtensible Markup Language) is a markup language much like HTML and designed to store and transport data and to be self-descriptive
HTML (Hyper Text Markup Language) the standard markup language for creating Web pages. It describes the structure of a Web page and consists of a series of elements. It's elements tell the browser how to display the content.
JSON (JavaScript Object Notation) files are lightweight and human-readable to store and exchange data. It is easy for machines to parse and generate these files and are based on the JavaScript programming language.
JSON files store data within {} similar to how a dictionary stores it in Python.

- **Missing and Outlier value treatment**

Data Cleaning is the process of finding and correcting the inaccurate/incorrect data that are present in the dataset. Missing values are usually represented in the form of Nan or null or None in the dataset. an Outlier is an observation in a given dataset that lies far from the rest of the observations. That means an outlier is vastly larger or smaller than the remaining values in the set. If our dataset is small, we can detect the outlier by just looking at the dataset.

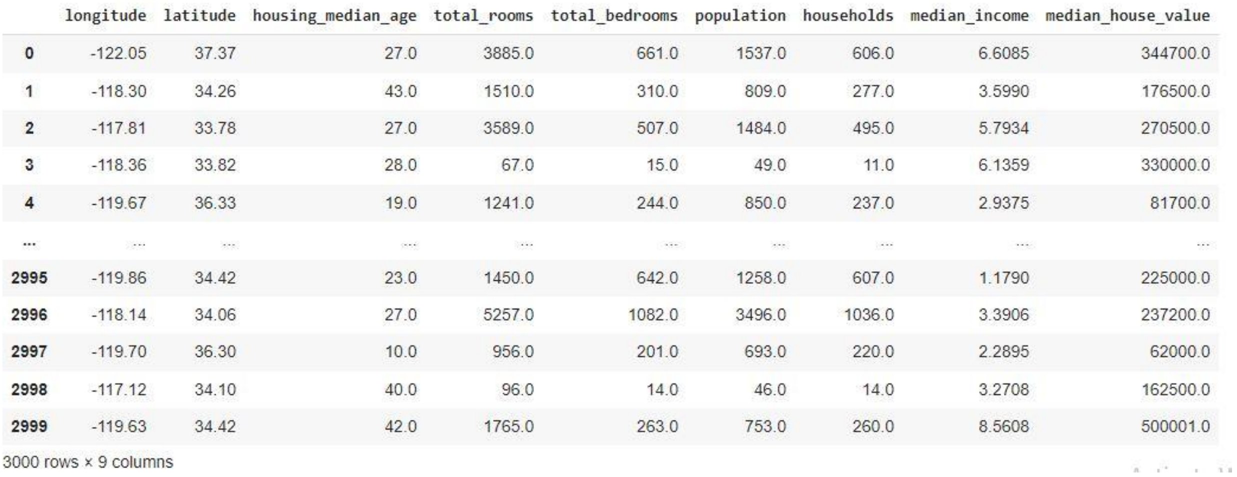
Code: Reading from CSV file.

```
# csv file
import pandas as pd
df = pd.read_csv('/content/sample_data/california_housing_test.csv')
df


#read xml file
df = pd.read_xml('/content/drive/MyDrive/employee.xml')
df.head()

# read Excel file into a DataFrame
df = pd.read_excel('/content/drive/MyDrive/Book1.xlsx')
df
```

```
# read Json file
import json
with open('/content/drive/MyDrive/sample1.json') as data:
    JSONdta = json.load(data)
print(JSONdta)
```

Output-:

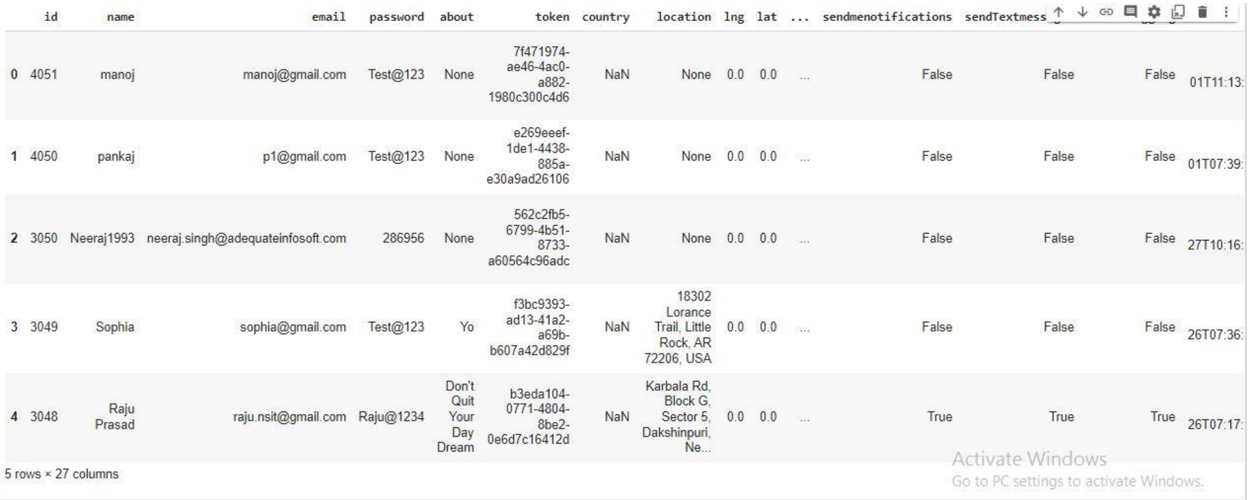
Csv file



	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value
0	-122.05	37.37	27.0	3885.0	661.0	1537.0	606.0	6.6085	344700.0
1	-118.30	34.26	43.0	1510.0	310.0	809.0	277.0	3.5990	176500.0
2	-117.81	33.78	27.0	3589.0	507.0	1484.0	495.0	5.7934	270500.0
3	-118.36	33.82	28.0	67.0	15.0	49.0	11.0	6.1359	330000.0
4	-119.67	36.33	19.0	1241.0	244.0	850.0	237.0	2.9375	81700.0
...	...	...	...	...	...	...	...	...	...
2995	-119.86	34.42	23.0	1450.0	642.0	1258.0	607.0	1.1790	225000.0
2996	-118.14	34.06	27.0	5257.0	1082.0	3496.0	1036.0	3.3906	237200.0
2997	-119.70	36.30	10.0	956.0	201.0	693.0	220.0	2.2895	62000.0
2998	-117.12	34.10	40.0	96.0	14.0	46.0	14.0	3.2708	162500.0
2999	-119.63	34.42	42.0	1765.0	263.0	753.0	260.0	8.5608	500001.0

3000 rows × 9 columns

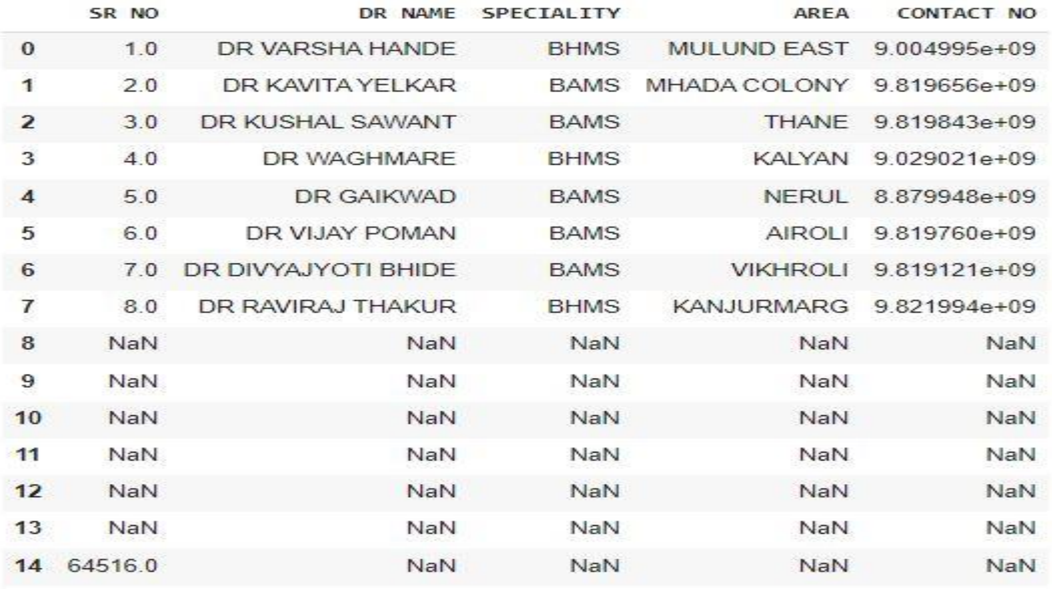
Xml file



	id	name	email	password	about	token	country	location	lng	lat	...	sendmenotifications	sendTextmess		
0	4051	manoj	manoj@gmail.com	Test@123	None	7f471974-ae46-4ac0-a882-1980c300c4d6	NaN	None	0.0	0.0	...	False	False	False	01T11:13:
1	4050	pankaj	p1@gmail.com	Test@123	None	e269eeef-1de1-4438-885a-e30a9ad26106	NaN	None	0.0	0.0	...	False	False	False	01T07:39:
2	3050	Neeraj1993	neeraj.singh@adequateinfosoft.com	286956	None	562c2fb5-6799-4b51-8733-a60564c96adc	NaN	None	0.0	0.0	...	False	False	False	27T10:16:
3	3049	Sophia	sophia@gmail.com	Test@123	Yo	f3bc9393-ad13-41a2-a69b-b607a42d829f	NaN	18302 Lorance Trail, Little Rock, AR 72206, USA	0.0	0.0	...	False	False	False	26T07:36:
4	3048	Raju Prasad	raju.nsit@gmail.com	Raju@1234	Don't Quit Your Day Dream	b3eda104-0771-4804-8be2-0e6d7c16412d	NaN	Karbala Rd, Block G, Sector 5, Dakshinpuri, Ne...	0.0	0.0	...	True	True	True	26T07:17:

5 rows × 27 columns

Excel file



	SR NO	DR NAME	SPECIALITY	AREA	CONTACT NO
0	1.0	DR VARSHA HANDE	BHMS	MULUND EAST	9.004995e+09
1	2.0	DR KAVITA YELKAR	BAMS	MHADA COLONY	9.819656e+09
2	3.0	DR KUSHAL SAWANT	BAMS	THANE	9.819843e+09
3	4.0	DR WAGHMARE	BHMS	KALYAN	9.029021e+09
4	5.0	DR GAIKWAD	BAMS	NERUL	8.879948e+09
5	6.0	DR VIJAY POMAN	BAMS	AIROLI	9.819760e+09
6	7.0	DR DIVYAJYOTI BHIDE	BAMS	VIKHROLI	9.819121e+09
7	8.0	DR RAVIRAJ THAKUR	BHMS	KANJURMARG	9.821994e+09
8	NaN	NaN	NaN	NaN	NaN
9	NaN	NaN	NaN	NaN	NaN
10	NaN	NaN	NaN	NaN	NaN
11	NaN	NaN	NaN	NaN	NaN
12	NaN	NaN	NaN	NaN	NaN
13	NaN	NaN	NaN	NaN	NaN
14	64516.0	NaN	NaN	NaN	NaN

Json file

{'fruit': 'Apple', 'size': 'Large', 'color': 'Red'}

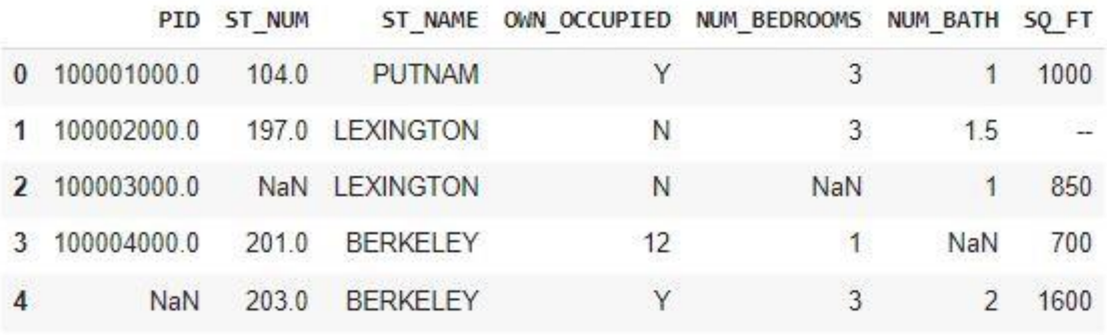
CHECK FOR MISSING DATA AND OUTLIER VALUES AND HANDLE THEM

Code:

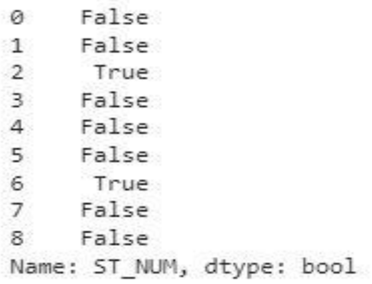
```
import pandas as pd
import numpy as np

# Read csv file into a pandas dataframe
df = pd.read_csv('/content/drive/MyDrive/property.csv')

# Take a look at the first few rows
df.head()
# Standard Missing Values

# Looking at the ST_NUM column
df['ST_NUM']
df['ST_NUM'].isnull()

# Non-Standard Missing Values
# Looking at the NUM_BEDROOMS column
df['NUM_BEDROOMS']
df['NUM_BEDROOMS'].isnull()

# Replace missing values with a number
df['ST_NUM'].fillna(125, inplace=True)
# Location based replacement
```

```
df.loc[2,'ST_NUM'] = 125
df
```

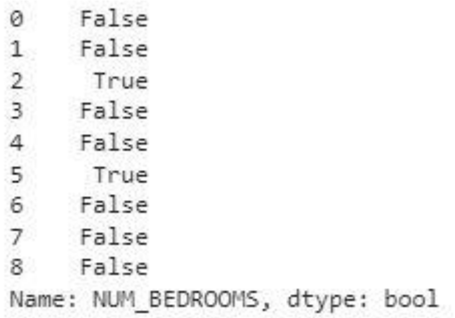
Output:-



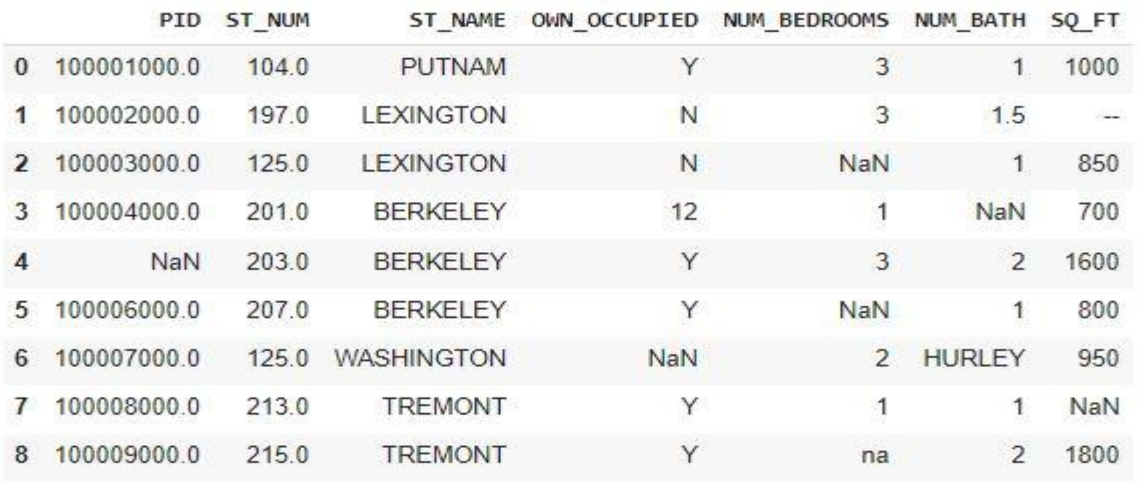
	PID	ST_NUM	ST_NAME	OWN_OCCUPIED	NUM_BEDROOMS	NUM_BATH	SQ_FT
0	100001000.0	104.0	PUTNAM	Y	3	1	1000
1	100002000.0	197.0	LEXINGTON	N	3	1.5	--
2	100003000.0	NaN	LEXINGTON	N	NaN	1	850
3	100004000.0	201.0	BERKELEY	12	1	NaN	700
4	NaN	203.0	BERKELEY	Y	3	2	1600



```
0    False
1    False
2     True
3    False
4    False
5    False
6     True
7    False
8    False
Name: ST_NUM, dtype: bool
```



```
0    False
1    False
2     True
3    False
4    False
5     True
6    False
7    False
8    False
Name: NUM_BEDROOMS, dtype: bool
```



	PID	ST_NUM	ST_NAME	OWN_OCCUPIED	NUM_BEDROOMS	NUM_BATH	SQ_FT
0	100001000.0	104.0	PUTNAM	Y	3	1	1000
1	100002000.0	197.0	LEXINGTON	N	3	1.5	--
2	100003000.0	125.0	LEXINGTON	N	NaN	1	850
3	100004000.0	201.0	BERKELEY	12	1	NaN	700
4	NaN	203.0	BERKELEY	Y	3	2	1600
5	100006000.0	207.0	BERKELEY	Y	NaN	1	800
6	100007000.0	125.0	WASHINGTON	NaN	2	HURLEY	950
7	100008000.0	213.0	TREMONT	Y	1	1	NaN
8	100009000.0	215.0	TREMONT	Y	na	2	1800

HANDLE OUTLIER

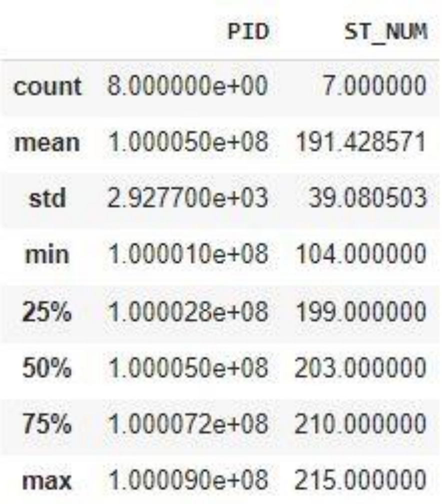
Code:

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

df = pd.read_csv('/content/drive/MyDrive/property.csv')
print(df.shape)
print(df.info())

df.describe()

# Identifying Outliers with Interquartile Range (IQR)
Q1 = df.quantile(0.25)
Q3 = df.quantile(0.75)
IQR = Q3 - Q1
print(IQR)

print(df < (Q1 - 1.5 * IQR)) == (df > (Q3 + 1.5 * IQR));

# Identifying Outliers with Skewness
print(df['ST_NUM'].skew())
df['ST_NUM'].describe()

#Identifying Outliers with Visualization
# 1. Box Plot
plt.boxplot(df['ST_NUM'])
plt.show()
# 2. Histogram
df.ST_NUM.hist()
```

```
# Replacing Outliers with Median Values
print(df['ST_NUM'].quantile(0.50))
print(df['ST_NUM'].quantile(0.95))
df['ST_NUM'] = np.where(df['ST_NUM'] > 325, 140, df['ST_NUM'])
df.describe()
```

Output:-

```
(9, 7)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9 entries, 0 to 8
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   PID           8 non-null      float64
 1   ST_NUM        7 non-null      float64
 2   ST_NAME       9 non-null      object
 3   OWN_OCCUPIED  8 non-null      object
 4   NUM_BEDROOMS  7 non-null      object
 5   NUM_BATH      8 non-null      object
 6   SQ_FT         8 non-null      object
dtypes: float64(2), object(5)
memory usage: 632.0+ bytes
None
```



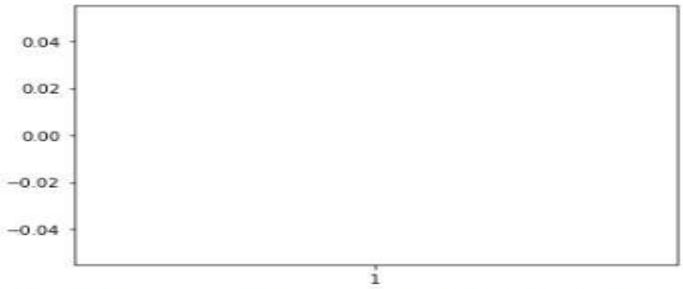
	PID	ST_NUM
count	8.000000e+00	7.000000
mean	1.000050e+08	191.428571
std	2.927700e+03	39.080503
min	1.000010e+08	104.000000
25%	1.000028e+08	199.000000
50%	1.000050e+08	203.000000
75%	1.000072e+08	210.000000
max	1.000090e+08	215.000000

```
PID       4500.0
```

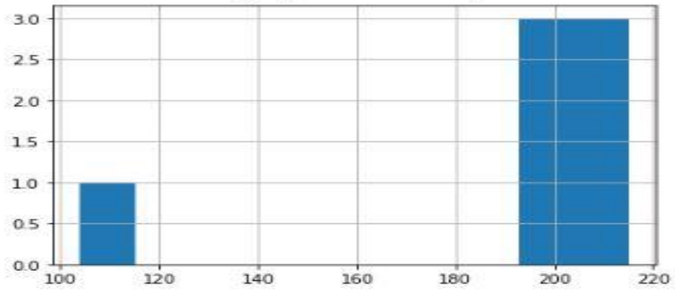
```
ST_NUM      11.0
dtype: float64
```

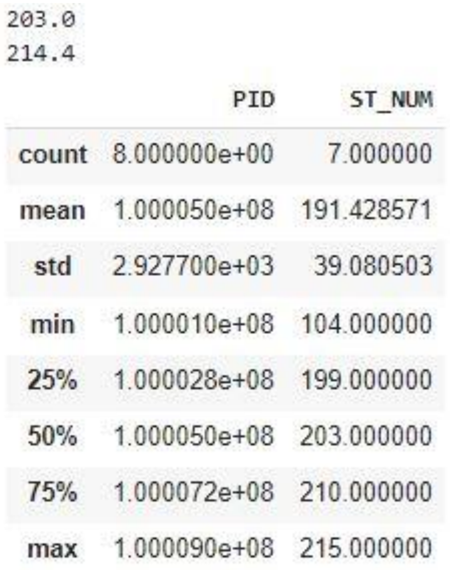
```
  NUM_BATH  NUM_BEDROOMS  OWN_OCCUPIED    PID  SQ_FT  ST_NAME  
ST_NUM
0     False        False        False  False  False    False    True
1     False        False        False  False  False    False   False
2     False        False        False  False  False    False   False
3     False        False        False  False  False    False   False
4     False        False        False  False  False    False   False
5     False        False        False  False  False    False   False
6     False        False        False  False  False    False   False
7     False        False        False  False  False    False   False
8     False        False        False  False  False    False   False
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: FutureWarning: Automatic reindexing on DataFrame vs Series comparisons is deprecated and will raise ValueError in a future version.  Do `left, right = left.align(right, axis=1, copy=False)` before e.g. `left == right`
  """Entry point for launching an IPython kernel.
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: FutureWarning: Automatic reindexing on DataFrame vs Series comparisons is deprecated and will raise ValueError in a future version.  Do `left, right = left.align(right, axis=1, copy=False)` before e.g. `left == right`
  """Entry point for launching an IPython kernel.
```

```
-2.497141895219325
count      7.000000
mean     191.428571
std       39.080503
min      104.000000
25%      199.000000
50%      203.000000
75%      210.000000
max      215.000000
Name: ST_NUM, dtype: float64
```



```
<matplotlib.axes._subplots.AxesSubplot at 0x7fa7a4345890>
```





```
203.0
214.4
```

	PID	ST_NUM
count	8.000000e+00	7.000000
mean	1.000050e+08	191.428571
std	2.927700e+03	39.080503
min	1.000010e+08	104.000000
25%	1.000028e+08	199.000000
50%	1.000050e+08	203.000000
75%	1.000072e+08	210.000000
max	1.000090e+08	215.000000

PRACTICAL – 4

Aim: Perform Reshaping of the hierarchical data and pivoting data frame data.

Theory: In pandas, we can arrange data within the data frame from the existing data frame. For example, we are having the same name with different features, instead of writing the name all time, we can write only once. We can create hierarchical data from the existing data frame using pandas. Stacking a Data Frame means moving (also rotating or pivoting) the innermost column index to become the innermost row index. The inverse operation is called unstacking. It means moving the innermost row index to become the innermost column index.

The pivot function is used to create a new derived table out of a given one. Pivot takes 3 arguements with the following names: index, columns, and values. As a value for each of these parameters you need to specify a column name in the original table. Then the pivot function will create a new table, whose row and column indices are the unique values of the respective parameters.

Code:

```
import pandas as pd
df = pd.read_csv('/content/drive/MyDrive/nba.csv')
print(df.head())

# reshape the dataframe using stack() method
df_stacked = df.stack()
print(df_stacked.head(26))

# reshape the dataframe using unstack() method
df_unstacked = df_stacked.unstack()
print(df_unstacked.head(10))

# reshape the dataframe using melt() method
# it takes two columns "Name" and "Team"
df_melt = df.melt(id_vars =['Name', 'Team'])
print(df_melt.head(10))
```

Output:-

```
            Name            Team  Number Position   Age Height  Weight  \
0  Avery Bradley  Boston Celtics     0.0       PG  25.0    6-2   180.0
1    Jae Crowder  Boston Celtics    99.0       SF  25.0    6-6   235.0
2   John Holland  Boston Celtics    30.0       SG  27.0    6-5   205.0
```

```
3    R.J. Hunter  Boston Celtics    28.0       SG  22.0    6-5   185.0
4  Jonas Jerebko  Boston Celtics     8.0       PF  29.0   6-10   231.0

             College     Salary
0              Texas  7730337.0
1          Marquette  6796117.0
2  Boston University        NaN
3      Georgia State  1148640.0
4                NaN  5000000.0




0  Name           Avery Bradley
   Team          Boston Celtics
   Number                   0.0
   Position                  PG
   Age                     25.0
   Height                   6-2
   Weight                 180.0
   College                Texas
   Salary             7730337.0
1  Name             Jae Crowder
   Team          Boston Celtics
   Number                  99.0
   Position                  SF
   Age                     25.0
   Height                   6-6
   Weight                 235.0
   College            Marquette
   Salary             6796117.0
2  Name            John Holland
   Team          Boston Celtics
   Number                  30.0
   Position                  SG
   Age                     27.0
   Height                   6-5
   Weight                 205.0
   College    Boston University
dtype: object
```

```
            Name            Team  Number Position   Age Height  Weight  \
0  Avery Bradley  Boston Celtics     0.0       PG  25.0    6-2   180.0
1    Jae Crowder  Boston Celtics    99.0       SF  25.0    6-6   235.0
2   John Holland  Boston Celtics    30.0       SG  27.0    6-5   205.0
3    R.J. Hunter  Boston Celtics    28.0       SG  22.0    6-5   185.0
4  Jonas Jerebko  Boston Celtics     8.0       PF  29.0   6-10   231.0
5   Amir Johnson  Boston Celtics    90.0       PF  29.0    6-9   240.0
6  Jordan Mickey  Boston Celtics    55.0       PF  21.0    6-8   235.0
7   Kelly Olynyk  Boston Celtics    41.0        C  25.0    7-0   238.0
8   Terry Rozier  Boston Celtics    12.0       PG  22.0    6-2   190.0
9   Marcus Smart  Boston Celtics    36.0       PG  22.0    6-4   220.0

            College      Salary
0             Texas   7730337.0
1         Marquette   6796117.0
2 Boston University         NaN
3     Georgia State   1148640.0
4               NaN   5000000.0
5               NaN  12000000.0
6               LSU   1170960.0
7          Gonzaga   2165160.0
8        Louisville   1824360.0
9    Oklahoma State   3431040.0
```

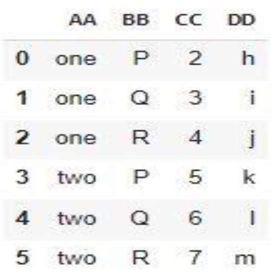
```
            Name            Team variable value
0  Avery Bradley  Boston Celtics   Number   0.0
1    Jae Crowder  Boston Celtics   Number  99.0
2   John Holland  Boston Celtics   Number  30.0
3    R.J. Hunter  Boston Celtics   Number  28.0
4  Jonas Jerebko  Boston Celtics   Number   8.0
5   Amir Johnson  Boston Celtics   Number  90.0
6  Jordan Mickey  Boston Celtics   Number  55.0
7   Kelly Olynyk  Boston Celtics   Number  41.0
8   Terry Rozier  Boston Celtics   Number  12.0
9   Marcus Smart  Boston Celtics   Number  36.0
```

PIVOTING DATA FRAME

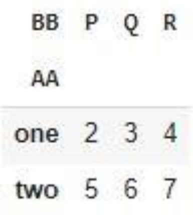
Code:

```
import numpy as np
import pandas as pd

df = pd.DataFrame({'AA': ['one', 'one', 'one', 'two', 'two',
                          'two'],
                   'BB': ['P', 'Q', 'R', 'P', 'Q', 'R'],
                   'CC': [2, 3, 4, 5, 6, 7],
                   'DD': ['h', 'i', 'j', 'k', 'l', 'm']})
df


df.pivot(index='AA', columns='BB', values='CC')


df.pivot(index='AA', columns='BB')['CC']

df.pivot(index='AA', columns='BB', values=['CC', 'DD'])
```

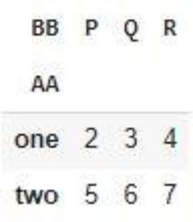
Output:-



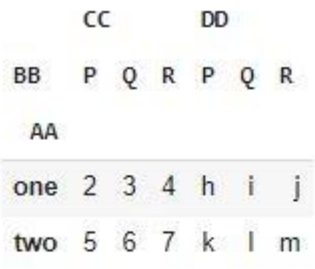
```
    AA   BB  CC  DD
0   one  P   2   h
1   one  Q   3   i
2   one  R   4   j
3   two  P   5   k
4   two  Q   6   l
5   two  R   7   m
```



```
BB   P  Q  R
AA
one  2  3  4
two  5  6  7
```



BB	P	Q	R
AA			
one	2	3	4
two	5	6	7



	CC			DD		
BB	P	Q	R	P	Q	R
AA						
one	2	3	4	h	i	j
two	5	6	7	k	l	m

PRACTICAL – 5

Aim: Connecting and extracting with various data resources in Tableau.

Theory: From the **Connect** pane, connect to an Excel spreadsheet or other connector that supports Data Interpreter such as Text (.csv) files, PDF files or Google sheets.
File-based data includes all sources of data where the data is stored in a file. File-based data sources include the following:

- Extracts: A .hyper or .tde file containing data that was extracted from an original source.
- Microsoft Access: An .mdb or .accdb database file created in Access.
- Microsoft Excel: An .xls , .xlsx , or .xlsm spreadsheet created in Excel. Multiple Excel sheets or sub-tables may be joined or unioned together in a single connection.
- Text file: A delimited text file, most commonly .txt , .csv , or .tab .Multiple text files in a single directory may be joined or unioned together in a single connection.
- Local cube file: A .cub file that contains multi-dimensional data. These files are typically exported from OLAP databases.
- Adobe PDF: A .pdf file that may contain tables of data that can be parsed by Tableau.
- Spatial file: A wide variety of spatial formats are supported such as .kml , .shp , .tab , .mif , spatial JSON, and ESRI database files. These formats contain spatial objects that can be rendered by Tableau.
- Statistical file: A .sav , .sas7bdat , .rda , or .rdata file generated by statistical tools, such as SAS or R.
- JSON file: A .json file that contains data in JSON format.

Data Sets:

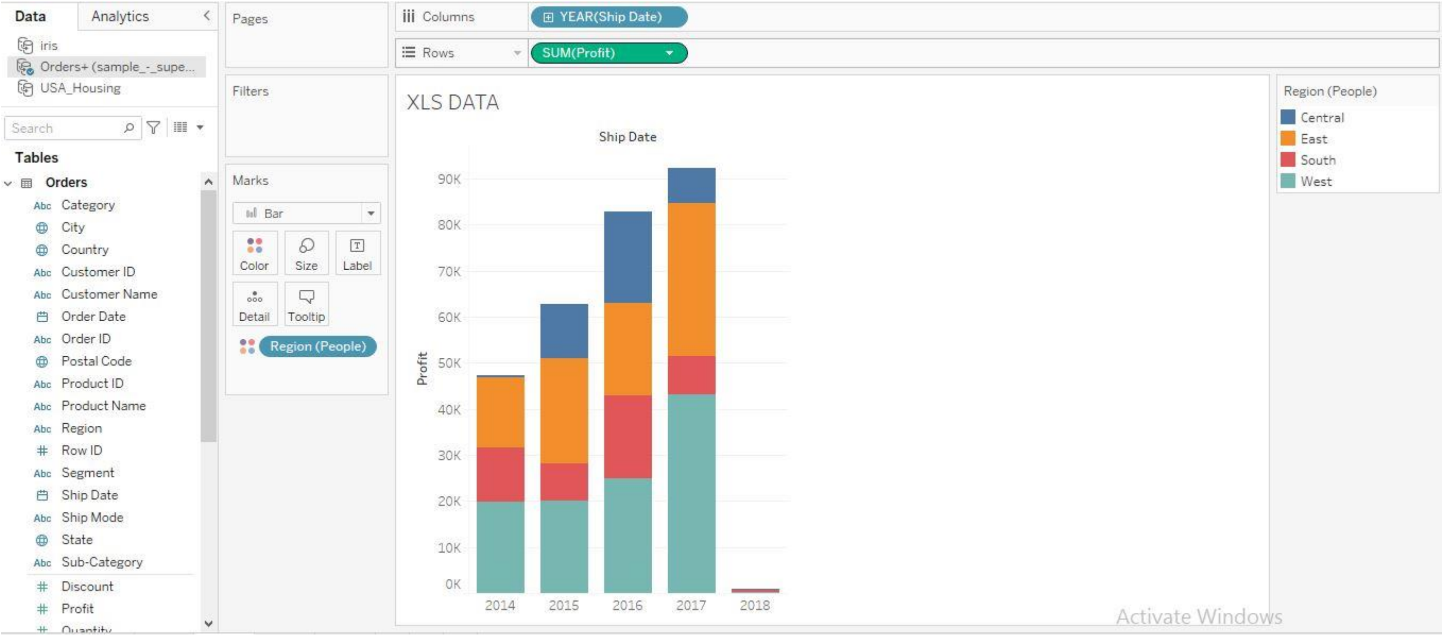
1. For excel: (sample_-superstore.xls) The sample data set for super store is being used for representation of year (ship date) to sum of profit representing the region of people according representing the different color. It has the parameters such as order id, order date, product code, country, region, etc.
2. For csv: (USA_housing.csv) The sample dataset used is of USA housing that represents area according to the population and sum of price in that particular area. It talks about the dataset which can be used when considering buying a house and considers parameter such as number of bedroom, price, etc.
3. For json:(iris.json) This is the iris data set which revolves around the iris of a flower that has parameters such as petal length, width, sepal length, width, etc. This is to show the petal length into the count of petal length.

Steps:

1. Import the data set into the tableau by google drive giving the authentication access.
2. Or we can import the data set into the tableau if we have downloaded the data set to the system.
3. For XLS we can use the above mentioned data set with year of ship date as a column data and sum of profit as the row data. We can include the region of people in the color marks to make the graph appeasing.
4. For CSV we can use the above mentioned data set with area of population as a column and sum of price I.e the price of the house as row data. Including the sum of price as the color marks.
5. For Json we can use the above mentioned data set with petal length in column data and count of total petal length as row data with petal width as the color mark with editing the colors as required.

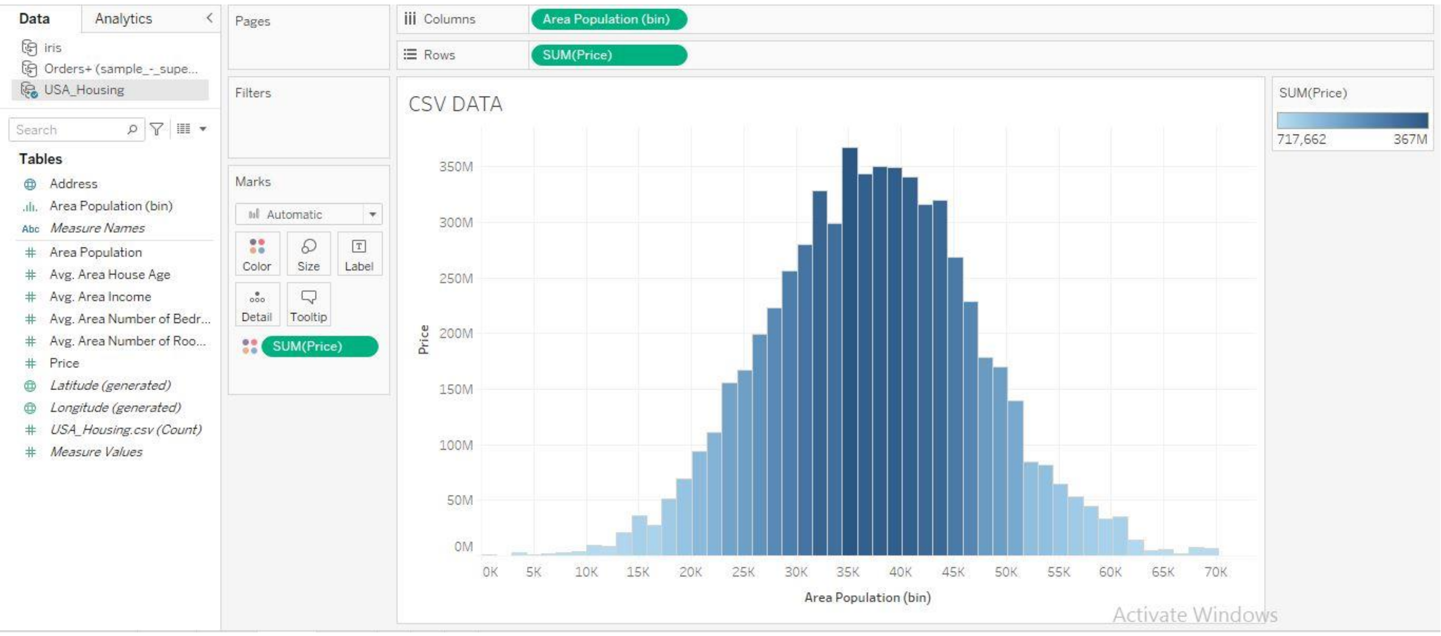
Output:-

XLS Data Set Graph



Columns: YEAR(Ship Date)
Rows: SUM(Profit)
Marks: Bar
Color: Region (People)

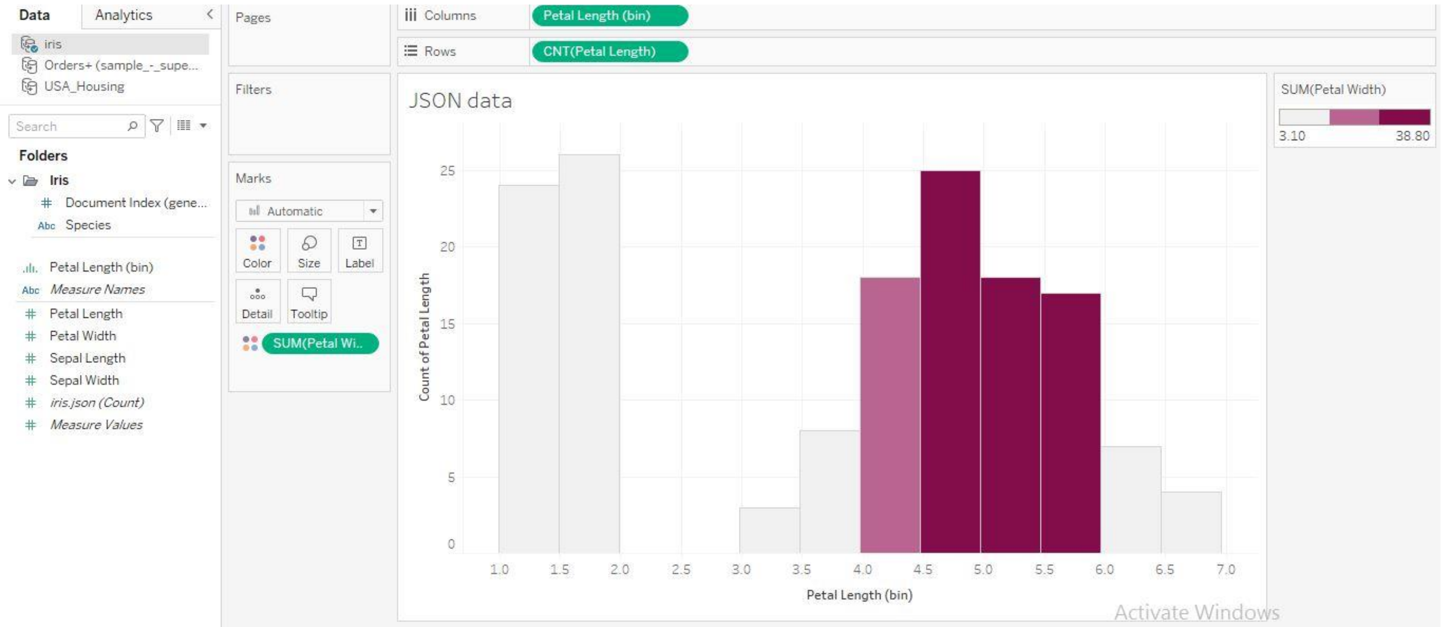
CSV data set graph



Columns: Area Population (bin)

Rows: SUM(Price)

JSON data set graph



Columns: Petal Length (bin)

Rows: CNT(Petal Length)

PRACTICAL – 6

Aim: Performing calculations and creating parameters in Tableau.

Theory: A calculation is often referred to as a Calculated Field in Tableau. Calculations consist of code that's made up of functions, operations, and references to other fields, parameters, constants groups, or sets. This code returns a value.

Types of calculations:

•Row- level Calculations: These calculations are performed for every row of underlying data.

•Aggregate Calculations: These calculations are performed at an aggregate level, which is usually defined by the dimensions used in the view.

•Level of detail Calculations: These special calculations are aggregations that are performed at a specified level of detail, with the results available at row level.

•Table Calculations: These calculations are performed on the table of aggregate data has been returned by the data source to Tableau.

A parameter in Tableau is a placeholder for a single, global value such as a number, date, or string. Parameters may be shown as controls (such as sliders, drop-down lists, or type-in text boxes) to end users of dashboards or views, giving them the ability to change the current value of the parameter. Parameter values may even be changed with actions.

Dataset: Superstore dataset with categories like country, region, subcategory, people, sales, profit etc.

Steps:

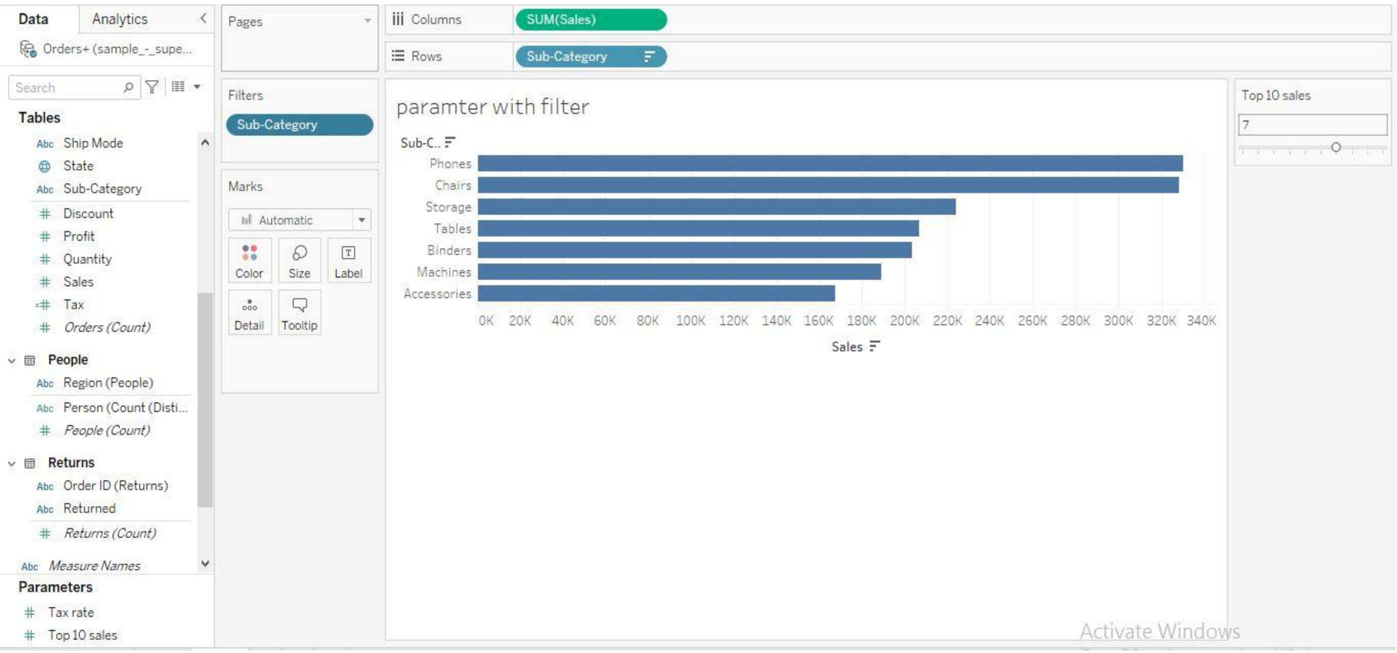
1. Parameter with filter
   a. Import the superstore dataset.
   b. Use sum of sale as columns and sub category with rows shelf, sort the data in descending order.
   c. Drag the sub category from the tables into the filters pane. A dialogue box appear> click on top > click on by field, the default is top 10 sales which can be changed later on if required. Click on apply > and ok.
   d. But if I need to make changes into the dashboard with a particular graph we might need a parameter.
      i. Right click on the sub category> edit filter > top> click on the box with 10 written on it> we get an option of create new parameter.
      ii. Name the parameter as Top 10 Sales> let the current value be 10> change the maximum value to 10(for the user to change the values only upto 10)> apply > and ok.

        iii. The parameter is created in the parameter pane > right click on it> show parameter. The parameter is visible on the right side of the chart.

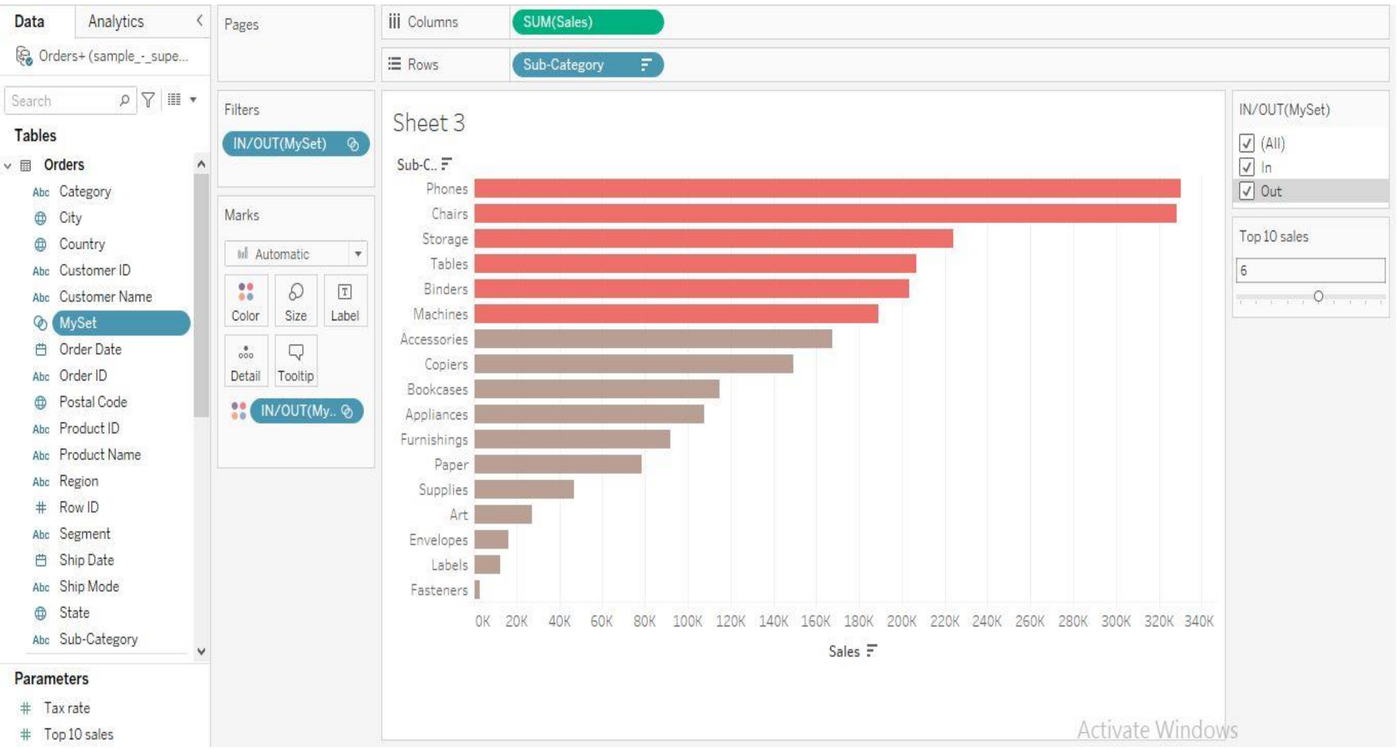
2. Parameter with sets
    a. Import the superstore dataset.
    b. Use sum of sale as columns and sub category with rows shelf, sort the data in descending order.
    c. To highlight the subcategory we have to create a set > click on sub category dropdown > create > set >.
        i. Change the name as MySet > click on Top.
        ii. Click on by field> change the value 10 to the name of parameter with filter (Top 10 Sales)> ok.
        iii. The MySet is created on the left data pane.
        iv. Show the parameter control on the right > right click on Top 10 Sales> show parameter.
        v. Drag the MySet into the color within the marks pane.
        vi. Based on the range in the parameter the sets are created.
3. Calculated fields.
    a. Import superstore dataset.
    b. Double click on subcategory > sales > profit > sort in descending order.
    c. If we need to calculate the cost that is sales- profit we need a measure but that is not available so we follow the below steps:
        i. Right click on any measure > create > calculated field.
        ii. We can name the calculated field and calculations, on the left side there are calculations we can perform.
        iii. Name the field as cost and write SUM([Sales])- SUM([Profit]) > apply > ok.
        iv. We get a cost measure name created. Drag and drop the cost into the sheet with other fields.

<u>Output:-</u>

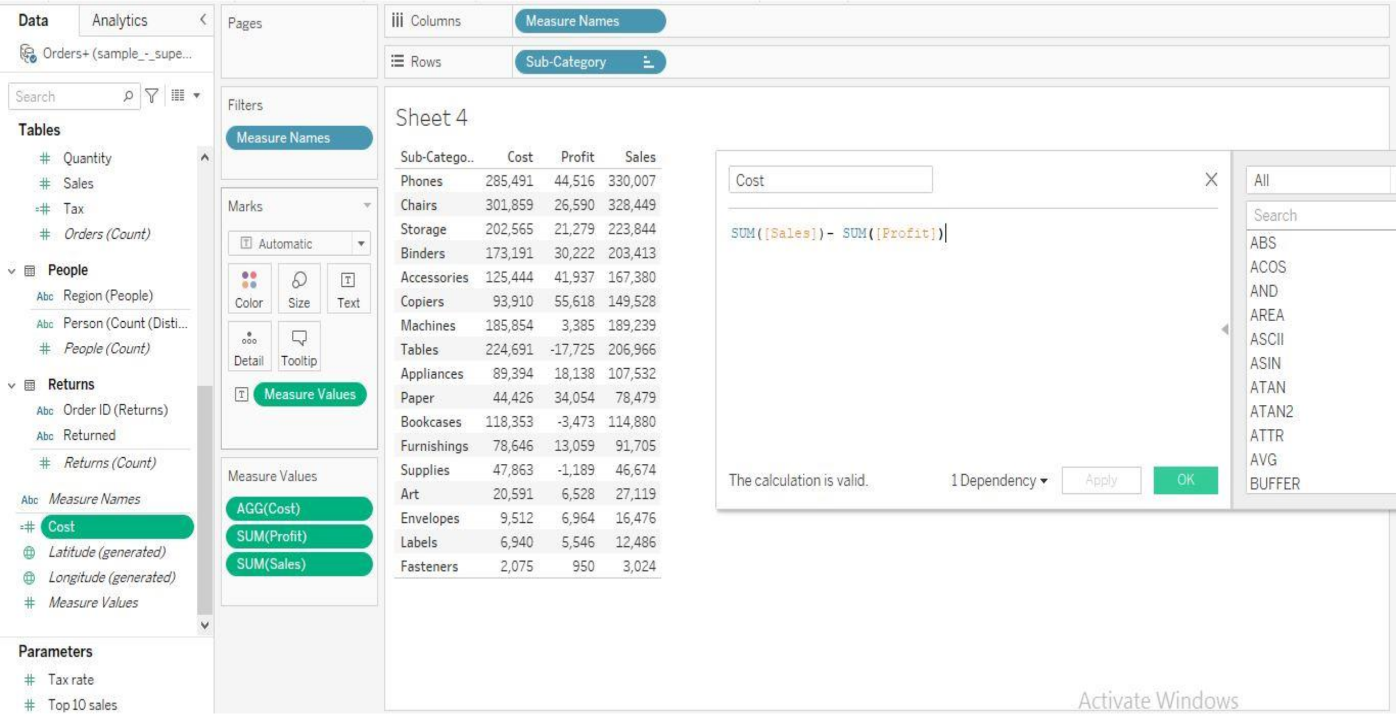
Parameter with filter



Parameter with Sets



Calculated Fields



Sheet 4

Sub-Catego..	Cost	Profit	Sales
Phones	285,491	44,516	330,007
Chairs	301,859	26,590	328,449
Storage	202,565	21,279	223,844
Binders	173,191	30,222	203,413
Accessories	125,444	41,937	167,380
Copiers	93,910	55,618	149,528
Machines	185,854	3,385	189,239
Tables	224,691	-17,725	206,966
Appliances	89,394	18,138	107,532
Paper	44,426	34,054	78,479
Bookcases	118,353	-3,473	114,880
Furnishings	78,646	13,059	91,705
Supplies	47,863	-1,189	46,674
Art	20,591	6,528	27,119
Envelopes	9,512	6,964	16,476
Labels	6,940	5,546	12,486
Fasteners	2,075	950	3,024

Cost

```
SUM([Sales])- SUM([Profit])
```

The calculation is valid.

PRACTICAL – 7

Aim: Designing Tableau Dashboards for different displays and devices.

Theory: Dashboards can include layouts for different types of devices that span a wide range of screen sizes. When you publish these layouts to Tableau Server or Tableau Cloud, people viewing your dashboard experience a design optimized for their phone, tablet, or desktop. As the author, you only have to create a single dashboard and deliver a single URL.

Device layouts appear on the Dashboard tab, under Default. Initially, each device layout contains every item in the Default dashboard and derives its size and layout from Default as well.

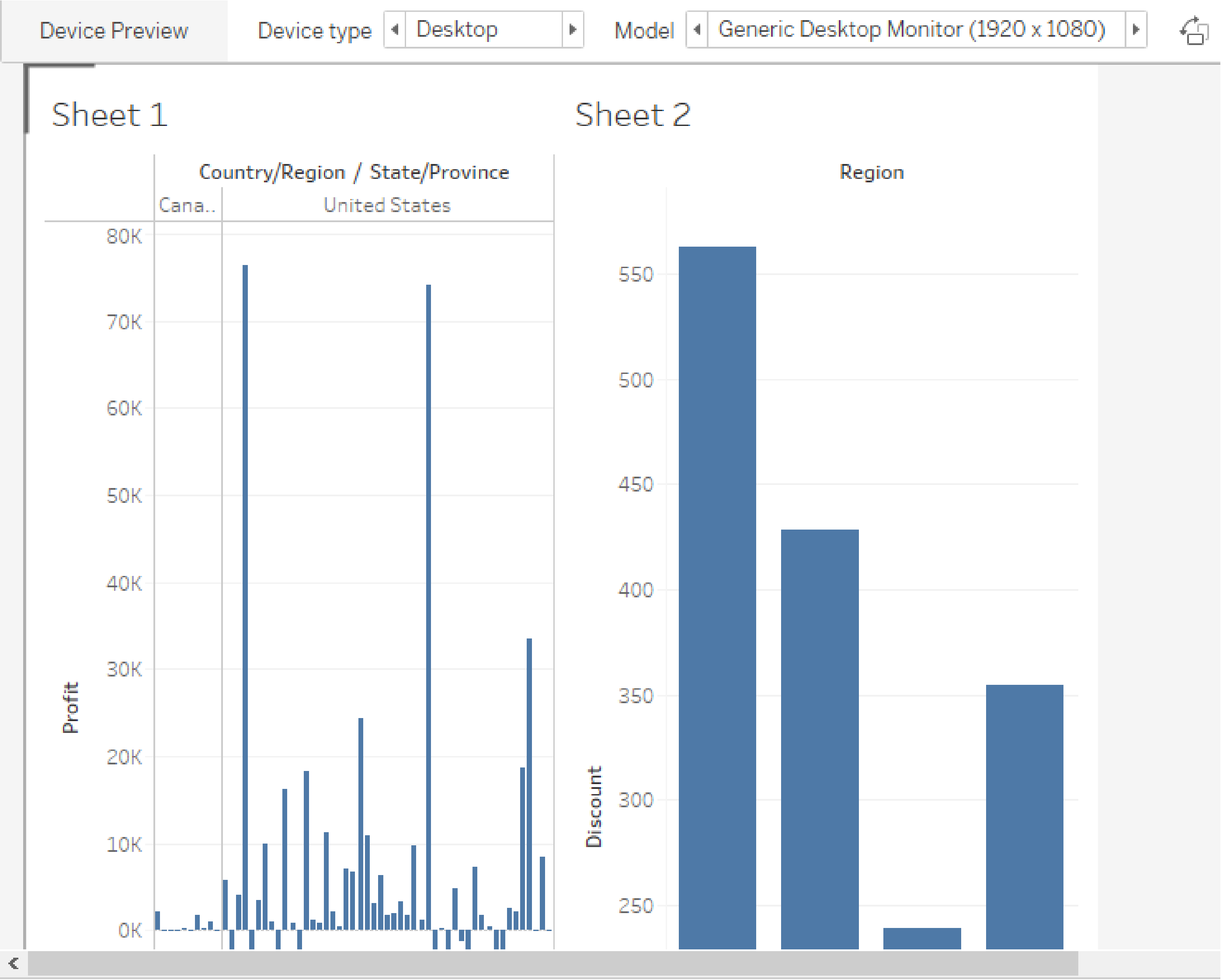
Think of the Default dashboard as the parent, and the device layouts (desktop, tablet, and phone) as its children. Any view, filter, action, legend or parameter that you want to add to a device layout must first exist in the Default dashboard.

Steps:

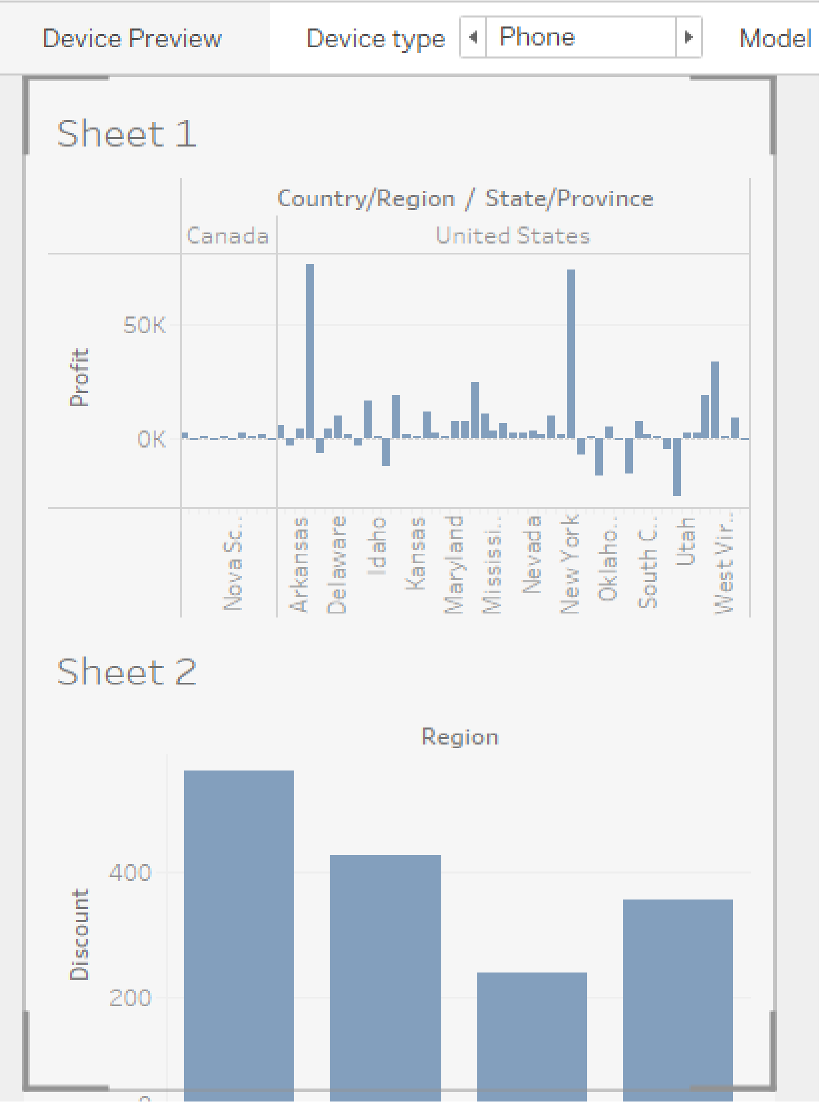
1. Open a dashboard.
2. On the dashboard tab on the left, click Device Preview.
3. Take a moment to click through the device types and models and explore the different screen sizes. Then set these options:
    a. To see how the dashboard will look in landscape vs. portrait mode, click on the option available. Usually, landscape is optimal for tablets and portrait is the best for phones.
    b. Select tableau mobile app to see how the dashboard will look with the app instead of the browser. The option is available for iOS or Android devices and shrinks the dashboard slightly, leaving space for the app controls.
4. Choose a device type, such as tablet.
5. Click through the device model options to see how the layout will appear on different models.
6. At left, explore the options under size.
    a. Default.
    b. Fit all
    c. Fit width

Output:-

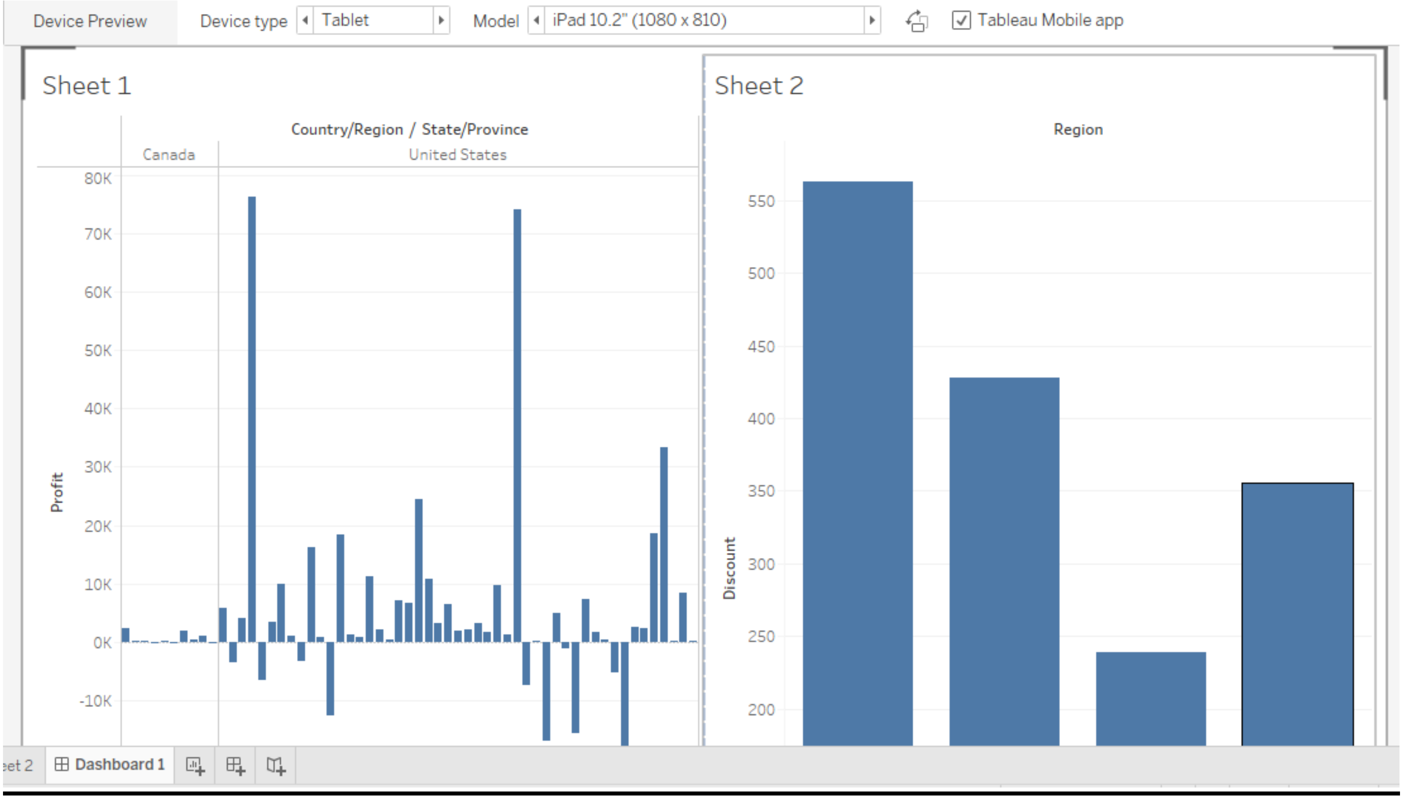
Desktop preview.



Phone preview



Tablet preview


Device Preview
Device type
Tablet
Model
iPad 10.2" (1080 x 810)
☑ Tableau Mobile app
Sheet 1
Country/Region / State/Province
Canada
United States
Profit
80K
70K
60K
50K
40K
30K
20K
10K
0K
-10K
Sheet 2
Region
Discount
550
500
450
400
350
300
250
200
eet 2
Dashboard 1

PRACTICAL – 8

Aim: Create a Trend Model using data, analyze it and use it for forecasting.

Theory: Forecasting in Tableau uses a technique known as exponential smoothing. Forecast algorithms try to find a regular pattern in measures that can be continued into the future. All forecast algorithms are simple models of a real-world data generating process (DGP).

For a high quality forecast, a simple pattern in the DGP must match the pattern described by the model reasonably well.

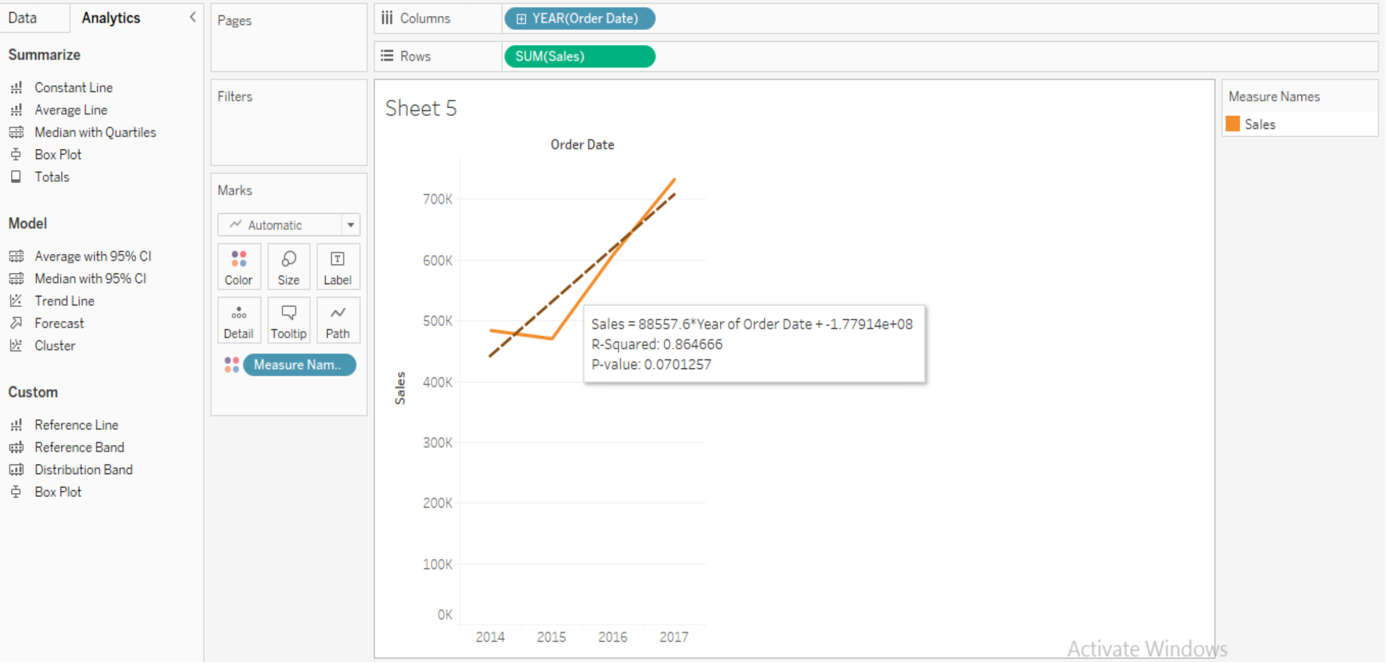
Quality metrics measure how well the model matches the DGP. If the quality is low, the precision measured by the confidence bands is not important because it measures the precision of an inaccurate estimate.

Steps:

1. Connect to data
   a. In tableau desktop, connect to **superstore sample data** provided by tableau.
2. Create the visualization
   a. Create a line chart with ship date(year) in the columns shelf and Sales in the Rows shelf.
   b. Go to the analysis tab and click on forecast under model category.
   c. On completing the above step, we find the options to set various options for forecast.
   d. Choose the forecast length as 2 years and leave the forecast model to automatic and then click ok.
   e. We also get minute details of the forecast model by choosing the option describe forecast. To get this option, right-click on forecast diagram.
   f. For trend model create a line chart with order date (year) in the columns and sales in the row shelf.
   g. After the graph is completed we have to go to analysis tab and click on trend line under model category.
   h. After the completion of above step we get a fine dashed line across the data line.
   i. We can also get minute details of the trend line model by choosing the describe trend line. To get this option, right-click on the trend line.

Output:-

Trend line model chart



Sales = 88557.6*Year of Order Date + -1.77914e+08
R-Squared: 0.864666
P-value: 0.0701257

Describing trend line

Trend Lines Model

A linear trend model is computed for sum of Sales given Order Date Year.

Model formula:	Measure Names*(Year of Order Date + intercept)
Number of modeled observations:	4
Number of filtered observations:	0
Model degrees of freedom:	2
Residual degrees of freedom (DF):	2
SSE (sum squared error):	6.13733e+09
MSE (mean squared error):	3.06866e+09
R-Squared:	0.864666
Standard error:	55395.5
p-value (significance):	0.0701257

Analysis of Variance:

Field	DFSSE	MSE	F	p-value

Measure Names 0 0 N/A N/A N/A

Individual trend lines:

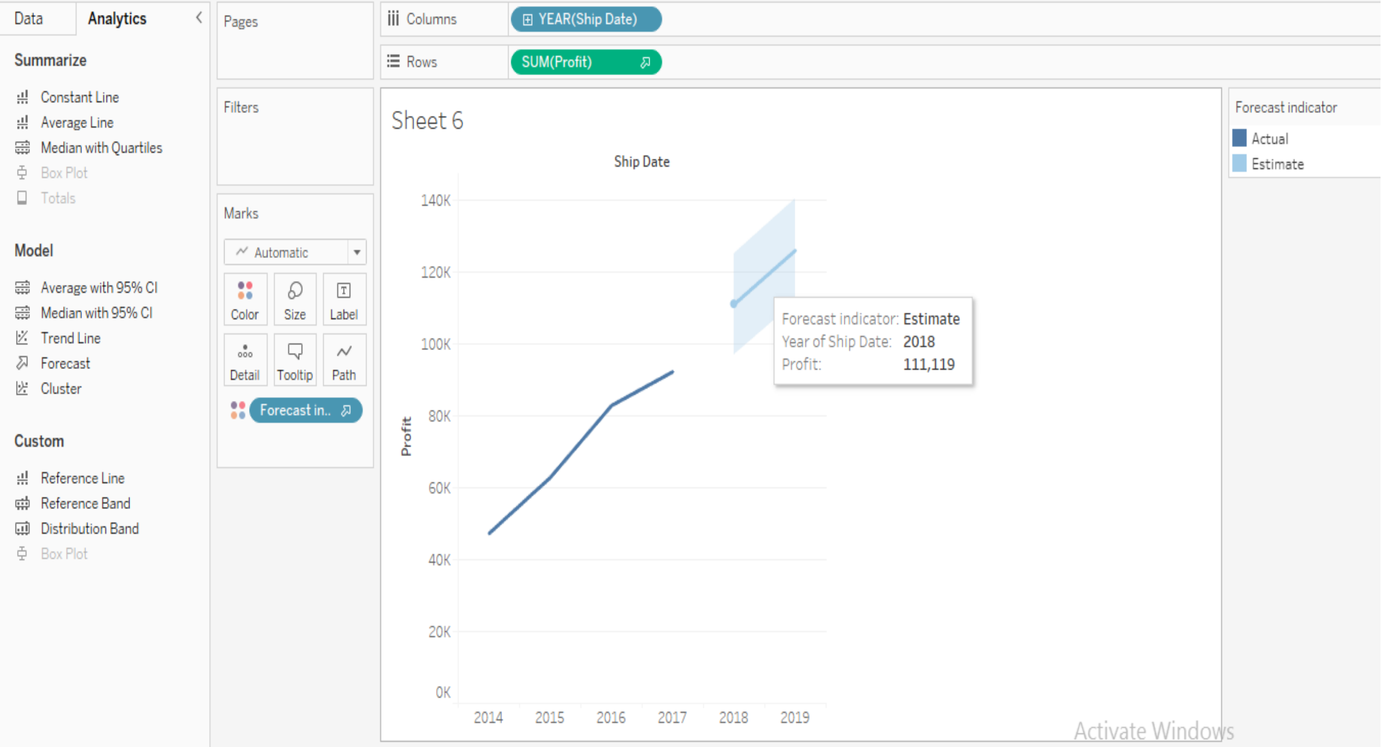
Panes Color Line Coefficients

Row Column Measure Names p-value DF Term Value StdErr t-value p-value

Sales Year of Order Date Sales 0.0701257 2 Year of Order Date 88557.6 24773.6 3.57467 0.0701257

intercept -1.77914e+08 4.99313e+07 -
3.56317 0.0705323

Forecasting chart



Forecast indicator: Estimate
Year of Ship Date: 2018
Profit: 111,119

Describing forecasting

Options Used to Create Forecasts

Time series: Year of Ship Date

Measures: Sum of Profit

Forecast forward: 8 quarters (2018 Q1 – 2019 Q4)

Forecast based on: 2014 Q1 – 2017 Q4

Ignore last: 1 quarter (2018 Q1)

Seasonal pattern: 4 quarter cycle

Sum of Profit

Initial	Change From Initial	Seasonal Effect				Contribution		Quality
2018 Q1	2018 Q1 – 2019 Q4	High		Low		Trend	Season	Quality
21,448 ± 7,570	18,530	2019 Q4	7,505	2019 Q1	-4,455	19.5%	80.5%	Ok

PRACTICAL – 9

Aim: Creating geospatial feature maps in tableau using geospatial data.

Theory: In Tableau Desktop, you can connect to the following spatial file types: Shapefiles, MapInfo tables, KML (Keyhole Markup Language) files, GeoJSON files, TopoJSON files, and Esri File Geodatabases. You can then create point, line, or polygon maps using the data in those files.

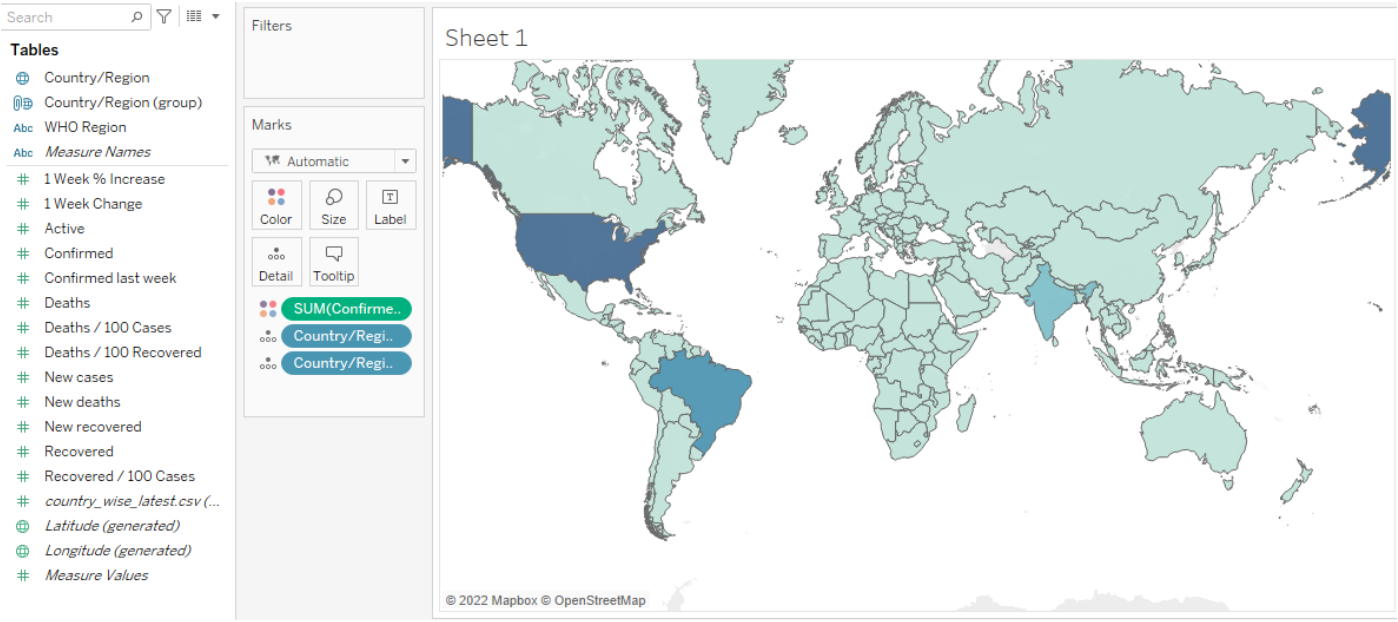
With a Creator license in Tableau Cloud or Tableau Server, you can upload spatial file formats that only require one file (KML, GeoJSON, TopoJSON, Esri shapefiles packaged in a .zip , and Esri File Geodatabases with the extension .gdb.zip) in the Files tab when you create a new workbook and connect to data. In current versions of tableau, you can only connect to point geometrics, linear geometrics, or polygons. You cannot connect to mixed geometry types.

Dataset: The dataset speaks about of Covid-19 cases reported by WHO.

Steps:

1. In tableau desktop: click the new data source icon and select Country wise file. OR. In tableau cloud or tableau server: select create > workbook. Select the files tab.
2. Navigate to the folder that contains the spatial data, select the spatial file you want to connect to, and then click open.
3. Drag and drop the country or state wise data we want to represent in the geospatial chart, into the sheet.
4. In the data pane, under the measures, the longitude and latitude is generated in the columns and rows respectively.
5. Drag and drop sum of happiness into color under the marks date pane.
6. On the right the sum (happiness rank) exists and the color can be changed.

Output:-


Search
Tables
Country/Region
Country/Region (group)
WHO Region
Measure Names
1 Week % Increase
1 Week Change
Active
Confirmed
Confirmed last week
Deaths
Deaths / 100 Cases
Deaths / 100 Recovered
New cases
New deaths
New recovered
Recovered
Recovered / 100 Cases
country_wise_latest.csv (...
Latitude (generated)
Longitude (generated)
Measure Values
Filters
Marks
Automatic
Color
Size
Label
Detail
Tooltip
SUM(Confirme..
Country/Regi..
Country/Regi..
Sheet 1
© 2022 Mapbox © OpenStreetMap

PRACTICAL – 10

Aim: Create dashboard and storytelling using tableau.

Theory: In Tableau, a **story** is a sequence of visualizations that work together to convey information. You can create stories to tell a data narrative, provide context, demonstrate how decisions relate to outcomes, or to simply make a compelling case.

A story is a sheet, so the methods you use to create, name, and manage worksheets and dashboards also apply to stories. At the same time, a story is also a collection of sheets, arranged in a sequence. Each individual sheet in a story is called a **story point**.

When you share a story —for example, by publishing a workbook to Tableau Public, Tableau Server, or Tableau Cloud—users can interact with the story to reveal new findings or ask new questions of the data.

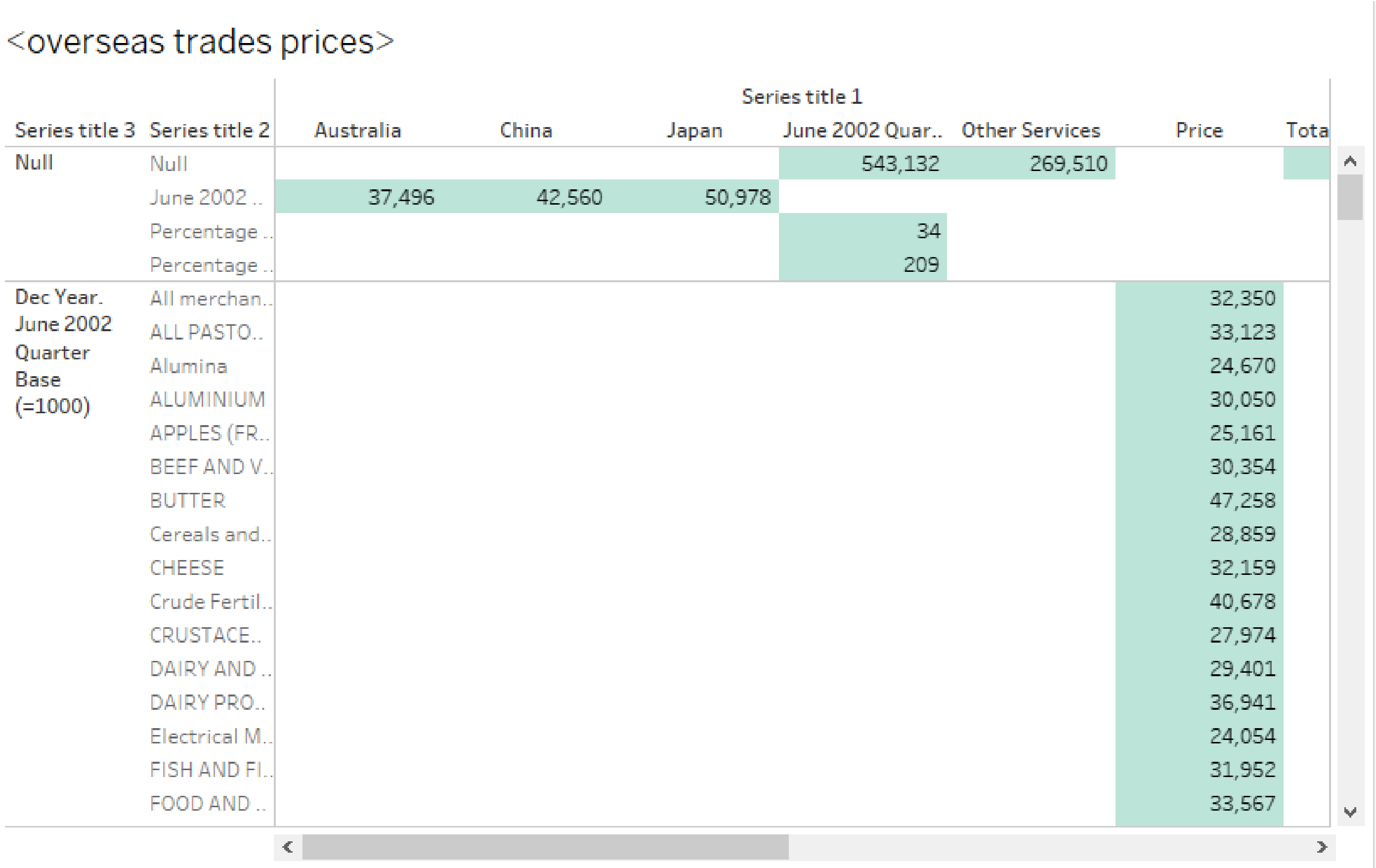
Dataset: The dataset speaks about India Trade with Global country imports and exports.

Steps:

1. Click the new story tab. Tableau opens a new story as your starting point.
2. By default, your story gets its title from the sheet name. To edit it, right-click the sheet tab, and choose rename sheet.
3. To start building your story, double-click a sheet on the left to add it to a story point. In Tableau Desktop, you can also drag sheets into your story point. When you add a sheet to a story point, that sheet remains connected to the original sheet. If you modify the original sheet, your changes will automatically be reflected on the story points that use it.
4. Click Add a caption to summarize the story point.
5. To further highlight the main idea of this story point, you can change a filter or sort on a field in the view. Then save your changes by clicking **Update** on the story toolbar above the navigator box.
6. Add another story point by doing one of the following:
    a. Click **Blank** to use a fresh sheet for the next story point.
    b. Start customizing a story point and click **Save as New** on the toolbar above the navigator box.
    c. Click **Duplicate** to use the current story point as the basis for a new one.
7. You can refine the look of your story using the options on the **Layout** tab.
    a. Click the **Layout** tab.
    b. Choose a navigator style that best suits your story, and show or hide the next and previous arrows.
8. Click the **Size** drop-down menu and select the story you want the dashboard to fit inside.

Output:-

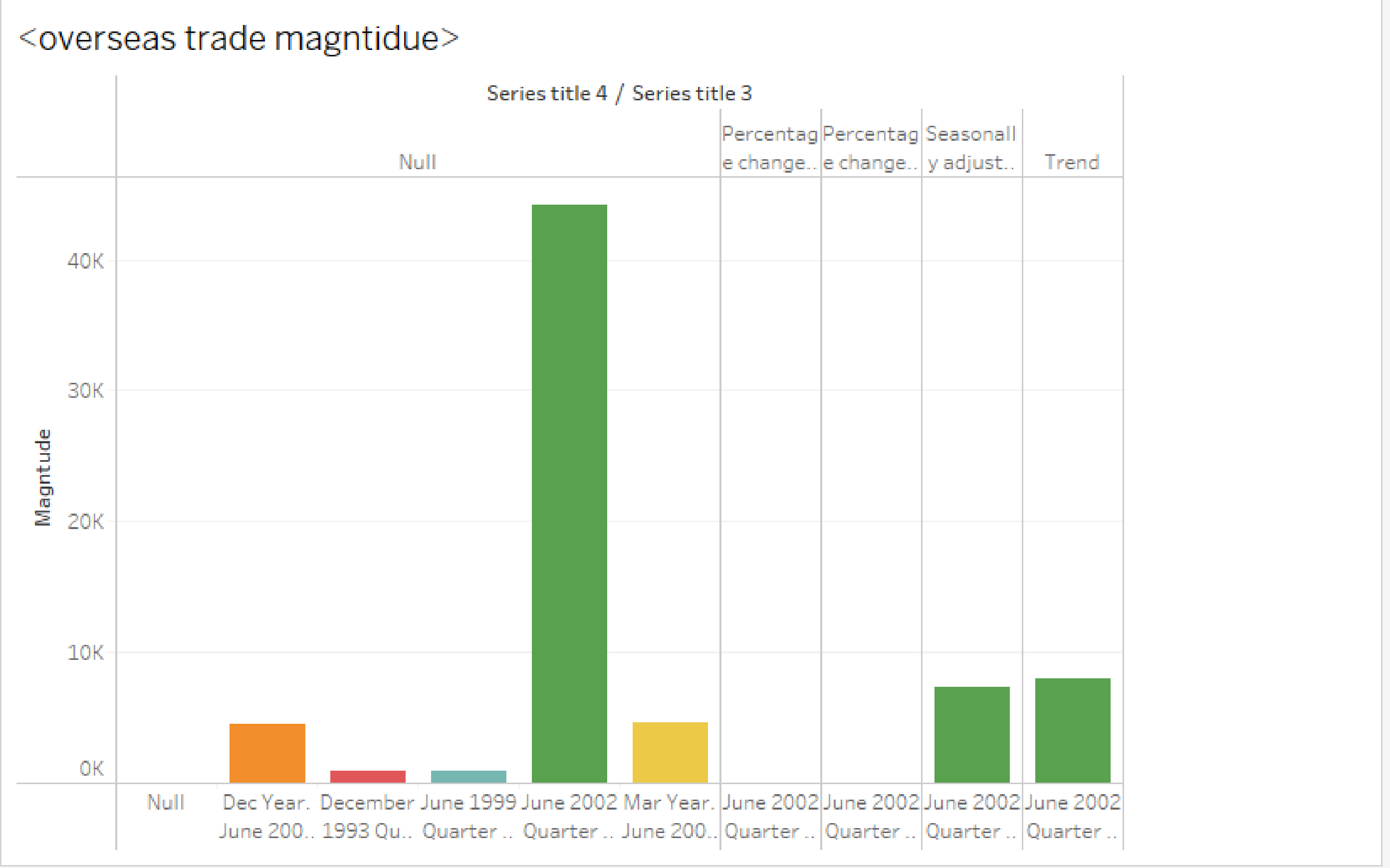
Sheet 1



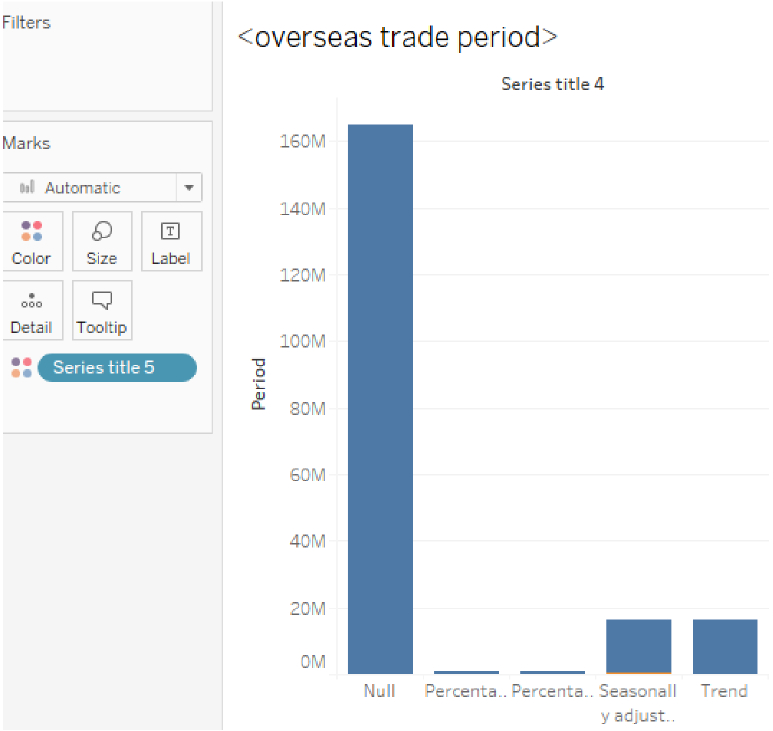
<overseas trades prices>

		Series title 1						
Series title 3	Series title 2	Australia	China	Japan	June 2002 Quar..	Other Services	Price	Tota
Null	Null				543,132	269,510		
	June 2002 ..	37,496	42,560	50,978				
	Percentage ..				34			
	Percentage ..				209			
Dec Year. June 2002 Quarter Base (=1000)	All merchan..						32,350	
	ALL PASTO..						33,123	
	Alumina						24,670	
	ALUMINIUM						30,050	
	APPLES (FR..						25,161	
	BEEF AND V..						30,354	
	BUTTER						47,258	
	Cereals and..						28,859	
	CHEESE						32,159	
	Crude Fertil..						40,678	
	CRUSTACE..						27,974	
	DAIRY AND ..						29,401	
	DAIRY PRO..						36,941	
	Electrical M..						24,054	
	FISH AND FI..						31,952	
	FOOD AND ..						33,567	

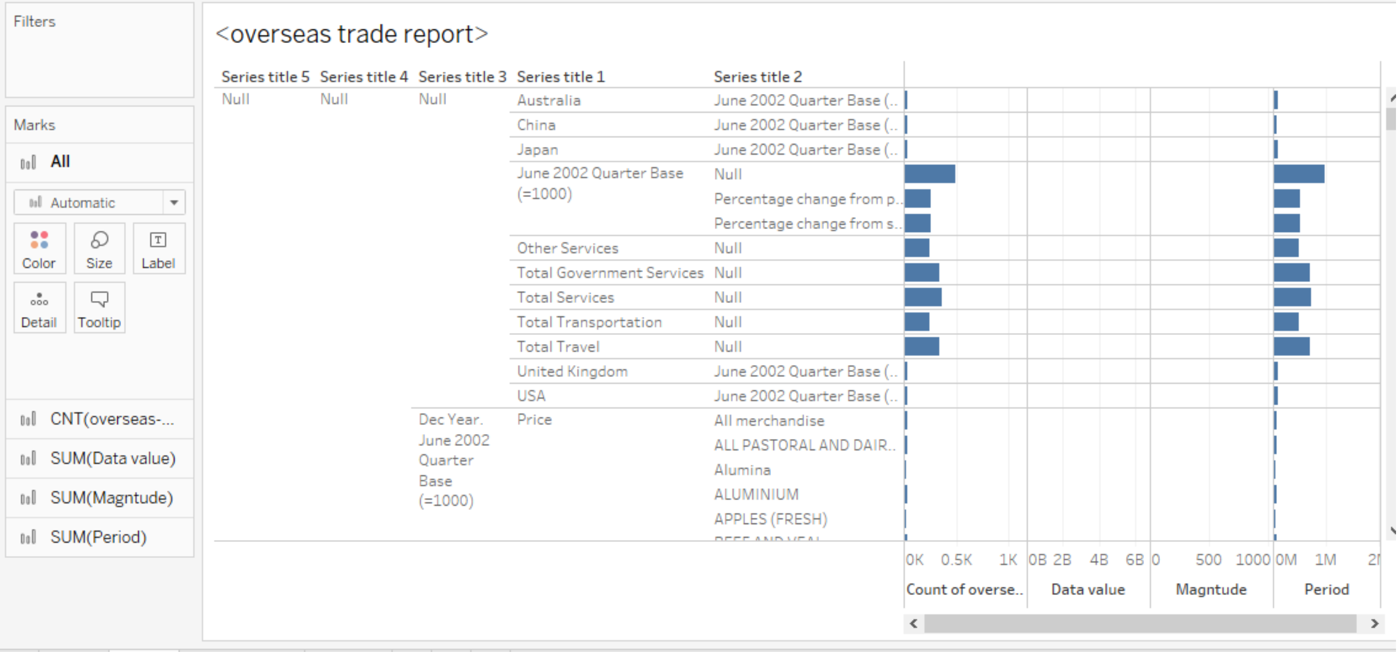
Sheet 2



Sheet 3



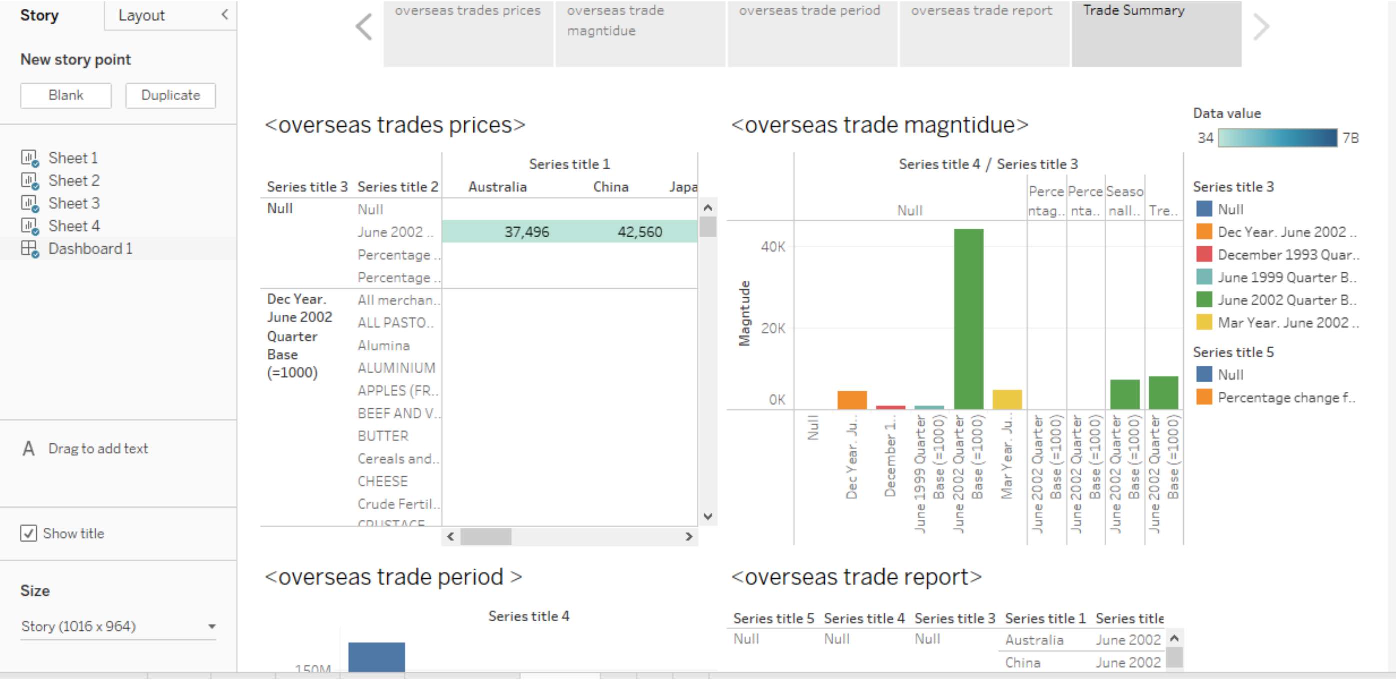
Sheet 4



<overseas trade report>

Series title 5	Series title 4	Series title 3	Series title 1	Series title 2
Null	Null	Null	Australia	June 2002 Quarter Base (..
			China	June 2002 Quarter Base (..
			Japan	June 2002 Quarter Base (..
			June 2002 Quarter Base (=1000)	Null
				Percentage change from p..
				Percentage change from s..
			Other Services	Null
			Total Government Services	Null
			Total Services	Null
			Total Transportation	Null
			Total Travel	Null
			United Kingdom	June 2002 Quarter Base (..
			USA	June 2002 Quarter Base (..
		Dec Year. June 2002 Quarter Base (=1000)	Price	All merchandise
				ALL PASTORAL AND DAIR..
				Alumina
				ALUMINIUM
				APPLES (FRESH)
				BEEF AND VEAL

Sheet 5



<overseas trades prices>

Series title 3	Series title 2	Series title 1 Australia	China	Japa
Null	Null			
	June 2002 ..	37,496	42,560	
	Percentage ..			
	Percentage ..			
Dec Year. June 2002 Quarter Base (=1000)	All merchan..			
	ALL PASTO..			
	Alumina			
	ALUMINIUM			
	APPLES (FR..			
	BEEF AND V..			
	BUTTER			
	Cereals and..			
	CHEESE			
	Crude Fertil..			
	CRUSTACE			

<overseas trade magntidue>

<overseas trade period >

<overseas trade report>

Series title 5	Series title 4	Series title 3	Series title 1	Series title
Null	Null	Null	Australia	June 2002
			China	June 2002