RAJALAKSHMI INSTITUTE OF TECHNOLOGY

Approved by AICTE, New Delhi and Affiliated to Anna University, Chennai.

Kuthambakkam, Chennai-124

BONAFIDE CERTIFICATE

DEPARTMENT: …….....

REGISTER NO: | | | | | | | | | | | |

This is to certify that bonafide record of work done by

……. *in the ……...Laboratory of this Institution for the……year / semester during the academic ……*

Staff In-Charge **HOD**

Submitted for the University Practical Examination to be held on……..

Internal Examiner **External Examiner**

INDEX

Sl. No	Date	NAME OF THE EXPERIMENT	Page no	Marks	Initials
1		Install the data Analysis and Visualization tool: R/ Python /Tableau, Public/ Power BI			
2		Perform exploratory data analysis (EDA) with datasets like emaildata set. Export all your emails as a dataset, import them inside a pandas data frame, visualize them and get different insights from the data.			
3		Working with numpy arrays, pandas data frames, basicplots using matplotlib			
4		Explore various variable and row filters in R for cleaning data			
5		Perform time series analysis and apply the various visualization techniques			
6		Perform data analysis and representation on A map using various map datasets with mouse Rollover effect ,user interaction			
7		Build cartographic visualization for multiple datasets involving various countries of the world, states and districts in India			
8		Perform EDA on wine quality data set			
9		Use a case study on a data set and apply the variousE DA and visualization techniques and present an analysis report			

Ex. No:1	Install the data Analysis and Visualization tool: R/ Python /Tableau Public/ Power BI
Date:	

AIM

To install the data analysis and visualization tools like R, PYTHON, TABLEAU PUBLIC, POWER BI.

ALGORITM

STEP 1: Install required libraries

STEP 2: Create a DataFrame from the data
STEP 3: Display the DataFrame
STEP 4: Data Analysis
STEP 5: Data Visualization

PROGRAM

```
import pandas as pd
import matplotlib.pyplot as plt
# Sample data (you can replace this with your own dataset)data
= {
   'Name': ['John', 'Jane', 'Mike', 'Emily', 'Alex'],
   'Age': [25, 30, 22, 28, 35],
   'Score': [90, 85, 78, 95, 88]
}
# Create a DataFrame from the data
df = pd.DataFrame(data)
# Display the DataFrame
print("DataFrame:")
print(df)
# Data Analysis
print("\nData Analysis:")
print("Mean Age:", df['Age'].mean())
print("Maximum Score:", df['Score'].max())
print("Minimum Score:", df['Score'].min())
# Data Visualization
plt.figure(figsize=(8, 4))
plt.bar(df['Name'], df['Score'])
plt.xlabel('Name')
plt.ylabel('Score')
plt.title('Scores of Students')
plt.show()
```

OUTPUT

DATAFRAME

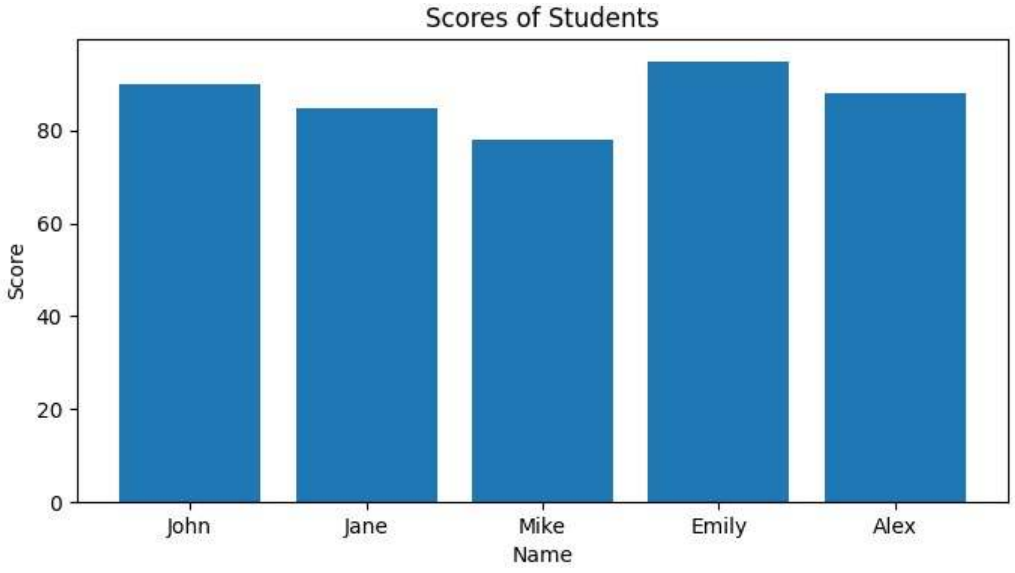
S.no	Name	Age	Score
1	John	25	90
2	Jane	30	85
3	Mike	22	78
4	Emily	28	95
5	Alex	35	88

DATA ANALYSIS

Mean Age: 28.0

Maximum Score: 95

Minimum Score: 78



RESULT

Thus, the installation of analysis and visualization tool is done successfully.

Ex. No:2	PERFORM EXPLORATORY DATA ANALYSIS (EDA) WITH DATASETS LIKE EMAIL DATA SET.
Date:	

AIM

To perform Exploratory data Analysis (EDA) with Datasets like Email Dataset in python using PyCharm.

ALGORITHM

STEP 1: Import necessary libraries

STEP 2: Sample data for the email dataset

STEP 3: Create a DataFrame from the sample data STEP 4: Convert timestamp to datetime format STEP 5: Save the DataFrame to a CSV file
STEP 6: Load the data from CSV into a DataFrame
STEP 7: Display basic information about the DataFrame
STEP 8: Convert timestamp to datetime format
STEP 9: Create a new column for email length

STEP 10: Perform additional analysis and visualizations as needed based on the specific characteristics of your dataset.

PROGRAM

```
import pandas as
pddata = {
  'sender': ['alice@example.com', 'bob@example.com', 'alice@example.com'],
  'receiver': ['bob@example.com', 'alice@example.com', 'carol@example.com'],
  'subject': ['Hello', 'Meeting Reminder', 'Project Update'],
  'timestamp': ['2023-08-01 10:00:00', '2023-08-02 14:30:00', '2023-08-03 09:15:00'],
  'content': ['Hi Bob,\n\nHow are you?', 'Hi Alice,\n\nDon\'t forget the meeting at 3 PM.',
'HiCarol,\n\nHere\'s the latest project update.']
}
df = pd.DataFrame(data)
df['timestamp'] = pd.to_datetime(df['timestamp'])
df.to_csv('emails.csv', index=False)
print("CSV file created
successfully.")import pandas as pd
import matplotlib.pyplot as
pltimport seaborn as sns
df = pd.read_csv("emails.csv")
print(df.info())
```

df['timestamp'] = pd.to_datetime(df['timestamp'])

```
df.dropna(inplace=True)
df['email_length'] =
df['content'].apply(len)
plt.figure(figsize=(10, 6))
sns.histplot(data=df, x='email_length', bins=30, kde=True)
plt.xlabel('Email Length')
plt.ylabel('Count')
plt.title('Distribution of Email Lengths')
plt.show()
top_senders = df['sender'].value_counts()[:10]
top_receivers = df['receiver'].value_counts()[:10]
plt.figure(figsize=(12, 6))
sns.barplot(x=top_senders.index, y=top_senders.values)
plt.xticks(rotation=45)
plt.xlabel('Sender')
plt.ylabel('Number of Emails')
plt.title('Top 10 Email Senders')
plt.tight_layout()
plt.show()
df['year_month'] =
df['timestamp'].dt.to_period('M') email_activity =
df.groupby('year_month').size()
plt.figure(figsize=(12, 6))
email_activity.plot(kind='line')
plt.xlabel('Year-Month')
plt.ylabel('Number of Emails')
plt.title('Email Activity Over Time')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

OUTPUT

CSV file created successfully.

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3 entries, 0 to 2

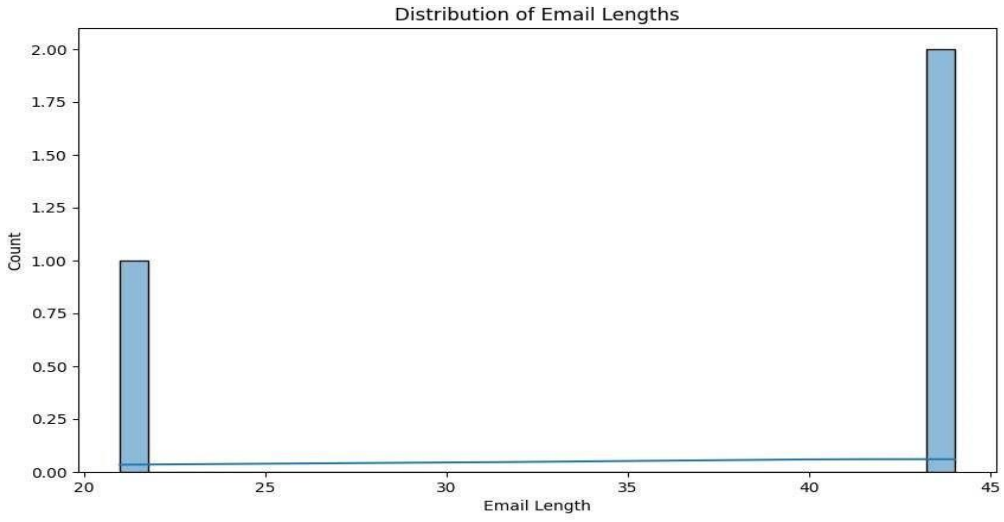
Data columns (total 5 columns):

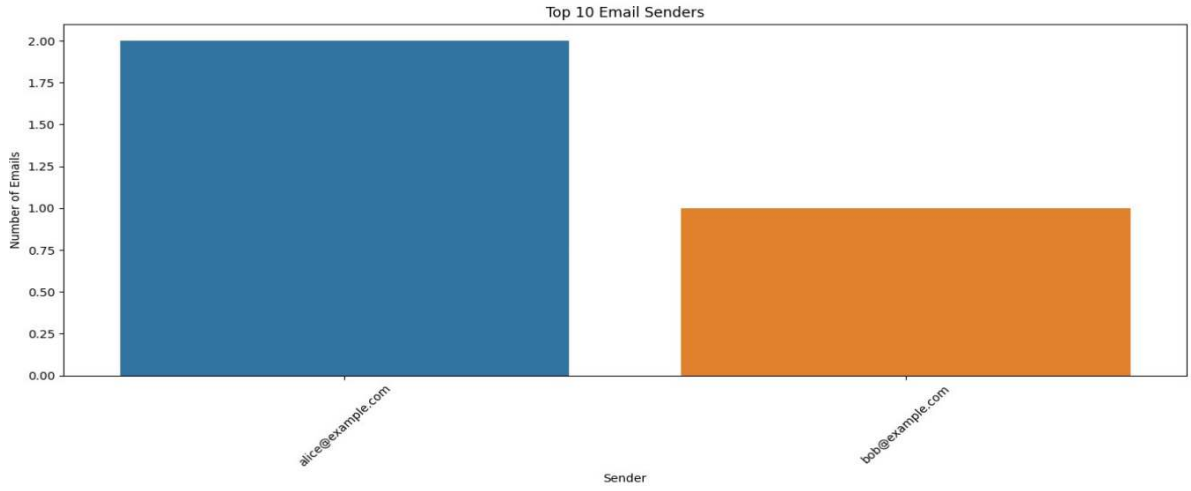
#	Column	Non-Null	Dtype

		Count	
----	--------	----------	-------
0	sender	3 non-null	object
1	receiver	3 non-null	object
2	subject	3 non-null	object
3	timestamp	3 non-null	object
4	content	3 non-null	object

dtypes: object(5)

memory usage: 252.0+ bytesNone





RESULT

Thus, to perform Exploratory data Analysis (EDA) with Datasets like Email Dataset in pythonusing PyCharm is done successfully.

Ex. No:3	WORKING WITH NUMPY ARRAYS, PANDAS DATA

Date:	FRAMES,BASIC PLOTS USING MATPLOTLIB

AIM

To execute with NumPy Arrays, Pandas DataFrame, Basic Plots using Matplotlib in pythonusing PyCharm.

ALGORITHM

STEP 1: Install required libraries

STEP 2: Create a DataFrame from the
dataSTEP 3: Display the DataFrame
STEP 4: Perform basic operations with NumPy Arrays, Basic plots using Matplotlib
STEP 5: Data Analysis
STEP 6: Data Visualization

PROGRAM

1)NUMPY ARRAYS

```
import numpy as np
arr=np.array([[1,2,3],[4,2,5]])
print("Array is of type:",type(arr))
print("No of dimensions:",arr.ndim)
print("Shape of array:",arr.shape)
print("Size of array:",arr.size)
print("Array stores elements of type:",arr.dtype)
```

OUTPUT

```
Array is of type: <class
'numpy.ndarray'>No of dimensions: 2
Shape of array: (2,
3)Size of array: 6
Array stores elements of type: int32
```

```
import numpy as np
a=np.array([[1,2,3],[3,4,5],[4,5,6]])
print(a)
print("After
Slicing")print(a[1:])
```

OUTPUT

[[1 2 3]

[3 4 5]

[4 5 6]]

After Slicing

[[3 4 5]

[4 5 6]]

```
import numpy as np
a=np.array([[1,2,3],[3,4,5],[4,5,6]])
print('Our array is:')print(a)
print('The items in the second column are:')print(a[...,1])
print('\n')
print('The items in the second row are:')
print(a[1,...])
print('\n')
print('The items column 1 onwards are:')
print(a[...,1])
```

OUTPUT

Our array is:

[[1 2 3]

[3 4 5]

[4 5 6]]

The items in the second column are:

[2 4 5]

The items in the second row are:

[3 4 5]

The items column 1 onwards are:

[2 4 5]

2) PANDAS DATAFRAMES

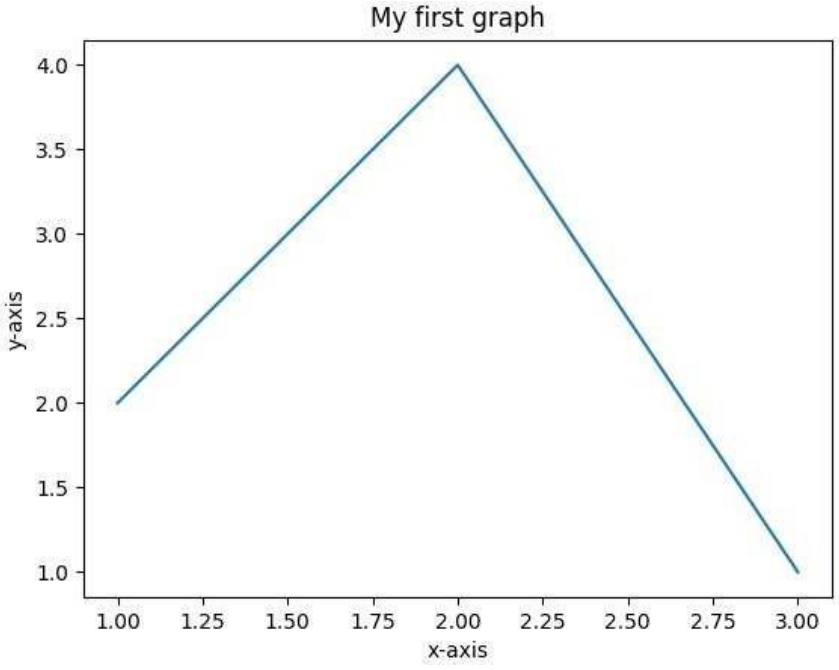
```
import numpy as npimport pandas as pd
data=np.array([['Col1','Col2'],['Row1',1,2],['Row2',3,4]])
print(pd.DataFrame(data=data[1:,1:],index=data[1:,0],columns=data[0,1:]))
```

```
my_2darray=np.array([[1,2,3],[4,5,6]])
print(pd.DataFrame(my_2darray))
my_dict={1:['1','3'],2:['1','2'],3:['2','4']}
print(pd.DataFrame(my_dict))
my_df=pd.DataFrame(data=[4,5,6,7],index=range(0,4),columns=['A'])
print(pd.DataFrame(my_df))
my_series=pd.Series({"UnitedKingdom":"London","India":"NewDelhi","United
States":"Washington","Belgium":"Brussels"})
print(pd.DataFrame(my_series))
df=pd.DataFrame(np.array([[1,2,3],[4,5,6]]))
print(df.shape)
print(len(df.index))
```

3) BASIC PLOT USING MATPLOTLIB

```
import matplotlib.pyplot as
pltx=[1,2,3]
y=[2,4,1]
plt.plot(x,y)
plt.xlabel('x-axis')
plt.ylabel('y-axis')
plt.title('My first
graph')plt.show()
```

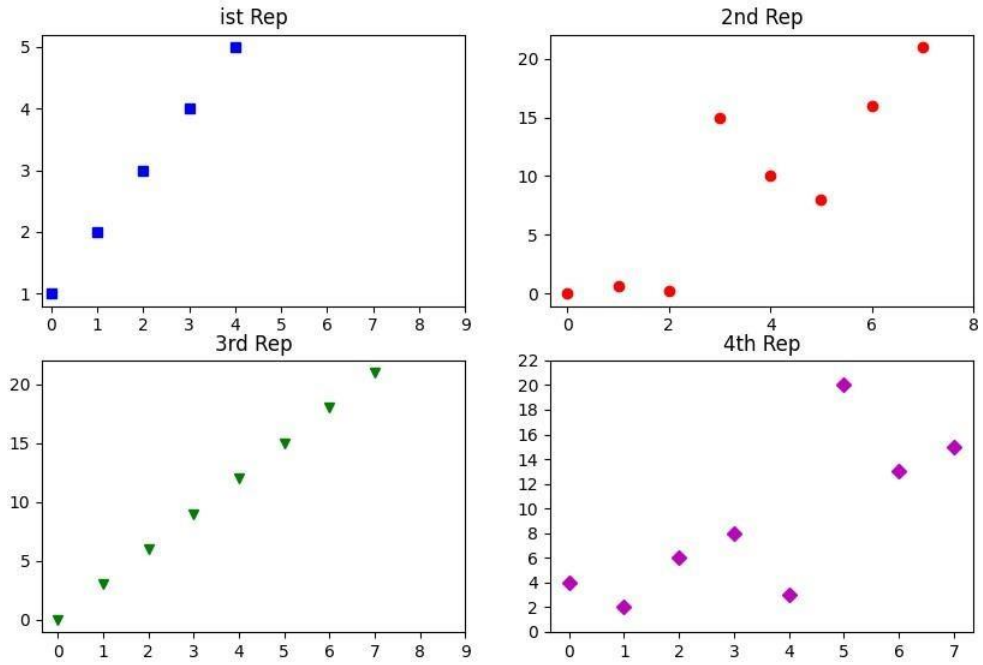
OUTPUT



```
import matplotlib.pyplot as
plt a=[1,2,3,4,5]
b=[0,0.6,0.2,15,10,8,16,21]
plt.plot(a)
plt.plot(b,"or")
plt.plot(list(range(0,22,3))
) plt.xlabel('Day->')
plt.ylabel('Temp->')
c=[4,2,6,8,3,20,13,15]
plt.plot(c,label='4th Rep')
ax=plt.gca()
ax.spines['right'].set_visible(False
)
ax.spines['top'].set_visible(False
) ax.spines['left'].set_bounds(-
3,40) plt.xticks(list(range(-
3,10))) plt.yticks(list(range(-
3,20,3)))
```

```
import matplotlib.pyplot as
plt a=[1,2,3,4,5]
b=[0,0.6,0.2,15,10,8,16,21]
c=[4,2,6,8,3,20,13,15]
fig=plt.figure(figsize=(10,10))
sub1=plt.subplot(2,2,1)
sub2=plt.subplot(2,2,2)
sub3=plt.subplot(2,2,3)
sub4=plt.subplot(2,2,4)
sub1.plot(a,'sb')
sub1.set_xticks(list(range(0,10,1)))
sub1.set_title('ist Rep')
sub2.plot(b,'or')
sub2.set_xticks(list(range(0,10,2)))
sub2.set_title('2nd Rep')
sub3.plot(list(range(0,22,3)),'vg')
sub3.set_xticks(list(range(0,10,1)))
sub3.set_title('3rd Rep')
sub4.plot(c,'Dm')
sub4.set_yticks(list(range(0,24,2)))
sub4.set_title('4th Rep')
plt.show()
```

OUTPUT



RESULT

Thus, to execute with NumPy Arrays, Pandas DataFrame, Basic Plots using Matplotlib inpython using PyCharm is executed successfully.

Ex. No:4	EXPLORE VARIOUS VARIABLE AND ROW FILTERS IN R FOR CLEANING DATA
Date:	

AIM

To explore various variable and Row Filters in R For cleaning data and also to apply various plot features in Ron sample data sets and visualize in python using PyCharm.

ALGORITHM

STEP 1: Create a Sample Dataset and Import Libraries

1.1. Create a sample dataset or use an existing one.

1.2. Import the necessary Python libraries, including Pandas and Matplotlib.STEP 2: Load and Explore the Dataset

2.1. Load your dataset into a Pandas DataFrame.

2.2. Explore the dataset to understand its structure and content.STEP 3: Data Cleaning

3.1. Handle missing values by either removing rows with missing values or imputingthem.

3.2. Remove duplicate rows if necessary. STEP 4: Apply Variable Filters

4.1. Select specific columns or variables of interest.

STEP 5: Apply Row Filters

5.1. Filter rows based on specific conditions.STEP 6: Data Visualization

6.1. Visualize the data using Matplotlib or other plotting libraries.

PROGRAM

```
import pandas as
pdimport numpy as
npdata = {
   'ID': range(1, 11),
   'Age': np.random.randint(18, 65, size=10),
   'Income': np.random.randint(30000, 90000, size=10),
   'Gender': ['Male', 'Female', 'Male', 'Female', 'Male', 'Female', 'Male', 'Female', 'Male', 'Male'],
   'Education': ['High School', 'Bachelor', 'Master', 'PhD', 'Bachelor', 'Master', 'Bachelor',
'PhD','High School', 'Master']
}
df = pd.DataFrame(data)
print(df.head())
print(df.describe())
```

```
print(df.isnull().sum())
print(df['Gender'].unique())
print(df['Education'].unique())
selected_columns = df[['Age', 'Income']]
print(selected_columns.head())
filtered_data = df[df['Age'] > 30]
print(filtered_data.head())
filtered_rows = df[(df['Gender'] == 'Male') & (df['Education'] == 'Master')]
print(filtered_rows.head())
import matplotlib.pyplot as plt
plt.hist(df['Age'], bins=5,
edgecolor='black') plt.title('Age
Distribution')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.show()
plt.boxplot(df['Income'])
plt.title('Income
Distribution')
plt.ylabel('Income')
plt.show()
gender_counts = df['Gender'].value_counts()
gender_counts.plot(kind='bar', color='skyblue')
plt.title('Gender Distribution')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.show()
education_counts = df['Education'].value_counts()
education_counts.plot(kind='pie', autopct='%1.1f%%', colors=['gold', 'lightcoral', 'lightgreen',
'lightskyblue'])
plt.title('Education Distribution')
plt.ylabel('')
plt.show()
```

OUTPUT

	ID	Age	Income	Gender	Education
0	1	23	63477	Male	High School
1	2	18	89372	Female	Bachelor
2	3	33	67140	Male	Master
3	4	22	80944	Female	PhD
4	5	44	33050	Male	Bachelor

	ID	Age	Income
count	10.00000	10.000000	10.000000
mean	5.50000	32.600000	59744.800000
std	3.02765	11.644741	20306.181121
min	1.00000	18.000000	30151.000000
25%	3.25000	22.250000	43115.250000
50%	5.50000	32.500000	63191.500000
75%	7.75000	43.500000	73913.250000
max	10.00000	`49.000000	89372.000000

ID	0
Age	0
Income	0
Gender	0
Education	0

dtype: int64

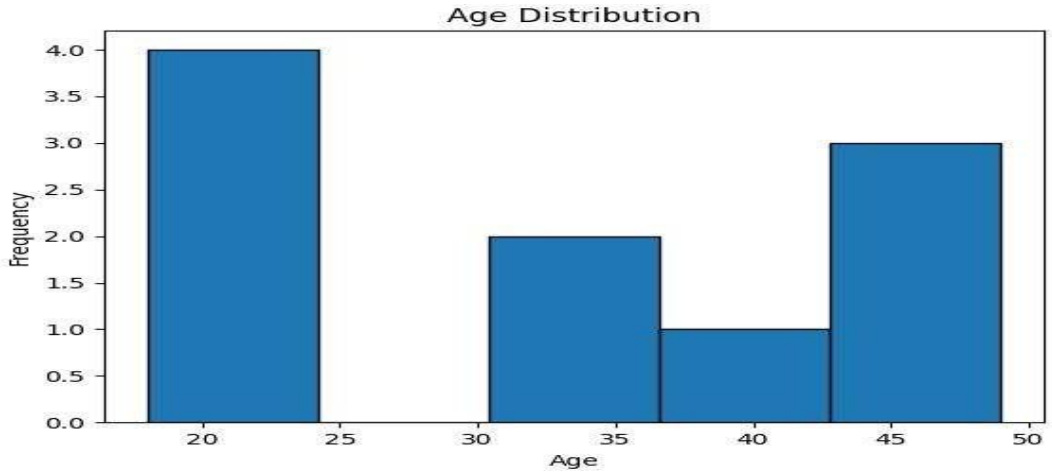
['Male' 'Female']

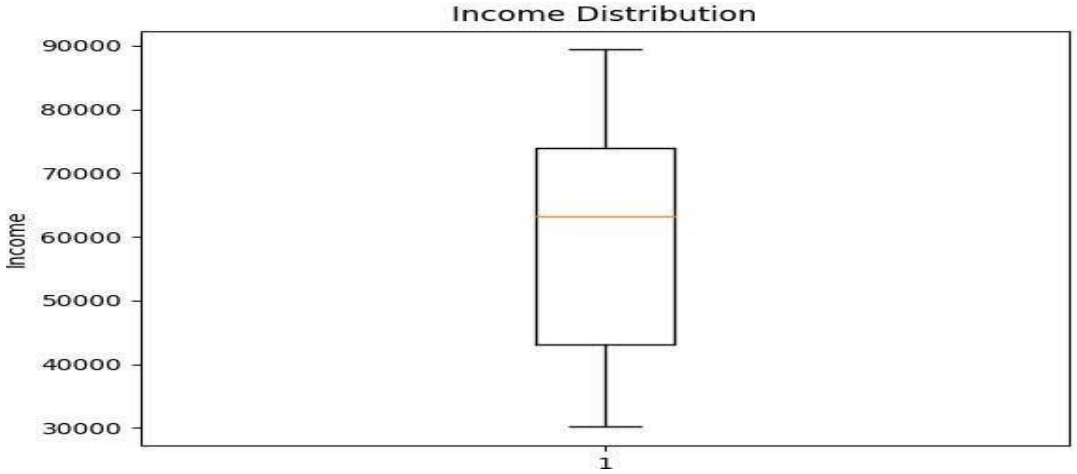
['High School' 'Bachelor' 'Master' 'PhD']

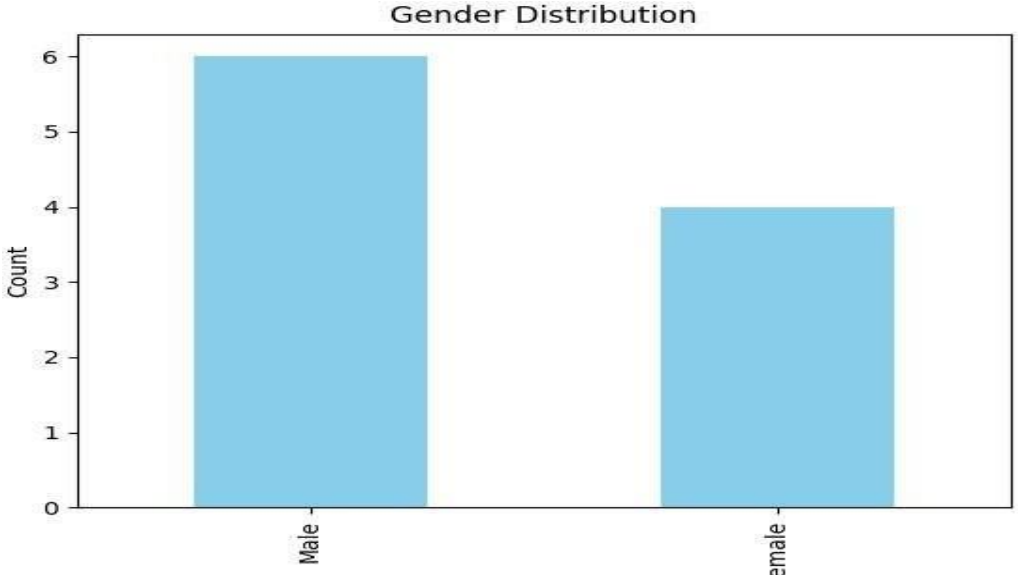
	Age	Income
0	23	63477
1	18	89372
2	33	67140
3	22	80944
4	44	33050

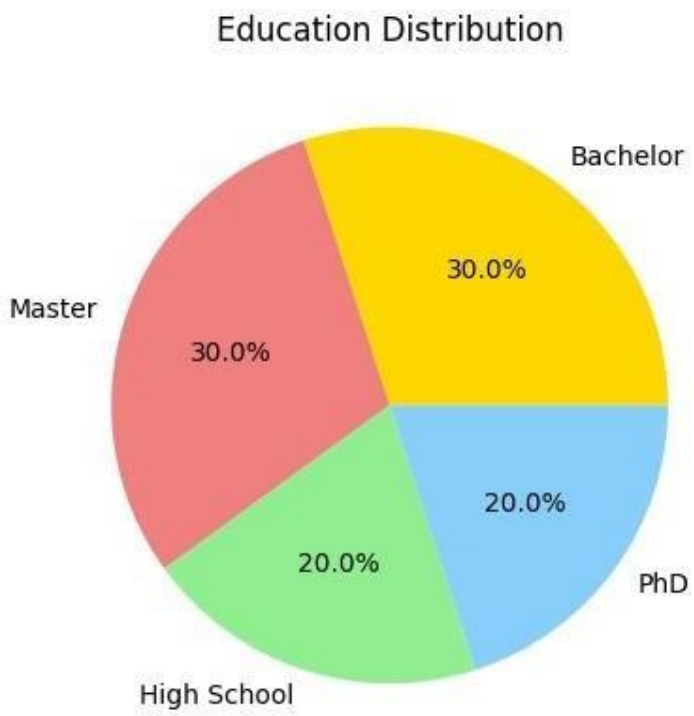
	ID	Age	Income	Gender	Education
2	3	33	67140	Male	Master
4	5	44	33050	Male	Bachelor
6	7	49	76171	Male	Bachelor
7	8	32	30151	Female	PhD
8	9	42	62906	Male	High School

	ID	Age	Income	Gender	Education
2	3	33	67140	Male	Master
9	10	44	39112	Male	Master


Age Distribution
Frequency
Age
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
20
25
30
35
40
45
50


Income Distribution
Income
90000
80000
70000
60000
50000
40000
30000
1


Gender Distribution
Count
0
1
2
3
4
5
6
Male
emale



RESULT:

Thus, the program to explore various variable and row filters in Python for cleaning data on sample data sets is done and visualized using PyCharm.

Ex. No:5	PERFORM TIME SERIES ANALYSIS AND APPLY THE VARIOUS VISUALIZATION TECHNIQUES
Date:	

AIM

To perform Time Series Analysis and apply the various visualization techniques in pythonusing PyCharm.

ALGORITHM

STEP 1: Data Preparation
Import necessary libraries (e.g., pandas, matplotlib, statsmodels).
Load your time series data into a panda DataFrame.
Set the date column as the index and ensure it's in date format.STEP 2: Data Exploration
Display basic statistics of your time series data using df. describe ().
Visualize the time series data using line plots to observe trends and patterns.
STEP 3: Decomposition
Decompose your time series data into trend, seasonality, and residuals using a method like seasonal decomposition.
STEP 4: Visualization of Decomposed Components
Plot the original time series data, trend, seasonality, and residuals.
STEP 5: Autocorrelation Plot
Create an autocorrelation plot to understand the temporal relationships in the data.
STEP 6: Rolling Statistics
Calculate rolling statistics (e.g., rolling mean and rolling standard deviation) and plot them.
STEP 7: Interpretation and Further Analysis
Run the program and visualizations for each above step is obtained in form of graphs.
STEP 8: Report and Visualization Export (Optional)
Create a report summarizing your findings and insights.
Save visualizations as image files if needed.

PROGRAM

```
import pandas as
pdimport numpy as
np
import matplotlib.pyplot as
pltimport statsmodels.api as
sm np.random.seed(0)
start_date = '2020-01-01'
end_date = '2023-12-31'
date_range = pd.date_range(start=start_date, end=end_date, freq='M') sales_data = np.random.randint(10000, 50000, size=len(date_range)) df = pd.DataFrame({'Date': date_range, 'Sales': sales_data}) df.set_index('Date', inplace=True)
print(df.describe())
```

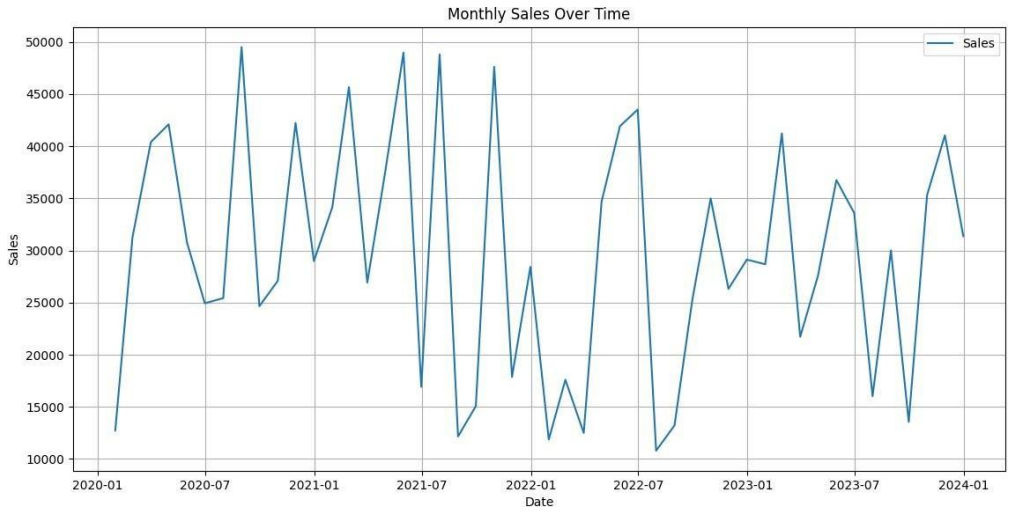
```
plt.figure(figsize=(12, 6))
plt.title('Monthly Sales Over Time')
plt.xlabel('Date')
plt.ylabel('Sales')
plt.plot(df.index, df['Sales'], label='Sales')
plt.legend()
plt.grid(True)
plt.show()
from statsmodels.tsa.seasonal import seasonal_decompose
decomposition = seasonal_decompose(df['Sales'], model='additive',
period=12) trend = decomposition.trend
seasonal = decomposition.seasonal
residual = decomposition.resid
plt.figure(figsize=(12, 10))
plt.subplots_adjust(hspace=0.4)
plt.subplot(4, 1, 1)
plt.title('Original Sales Data')
plt.plot(df.index, df['Sales'], label='Sales')
plt.legend()
plt.subplot(4, 1, 2)
plt.title('Trend')
plt.plot(trend,
label='Trend')plt.legend()
plt.subplot(4, 1, 3)
plt.title('Seasonality')
plt.plot(seasonal, label='Seasonality')
plt.legend()
plt.subplot(4, 1, 4)
plt.title('Residuals')
plt.plot(residual, label='Residuals')
plt.legend()
plt.show()
from statsmodels.graphics.tsaplots import plot_acf
plt.figure(figsize=(12, 4))
plot_acf(df['Sales'], lags=24) plt.title('Autocorrelation Plot') plt.xlabel('Lags')
plt.ylabel('ACF') plt.grid(True)
plt.show()
rolling_mean = df['Sales'].rolling(window=12).mean()
rolling_std = df['Sales'].rolling(window=12).std()
plt.figure(figsize=(12, 6))
plt.title('Rolling Mean and Standard Deviation')
plt.xlabel('Date')
plt.plot(df.index, df['Sales'], label='Sales')
```

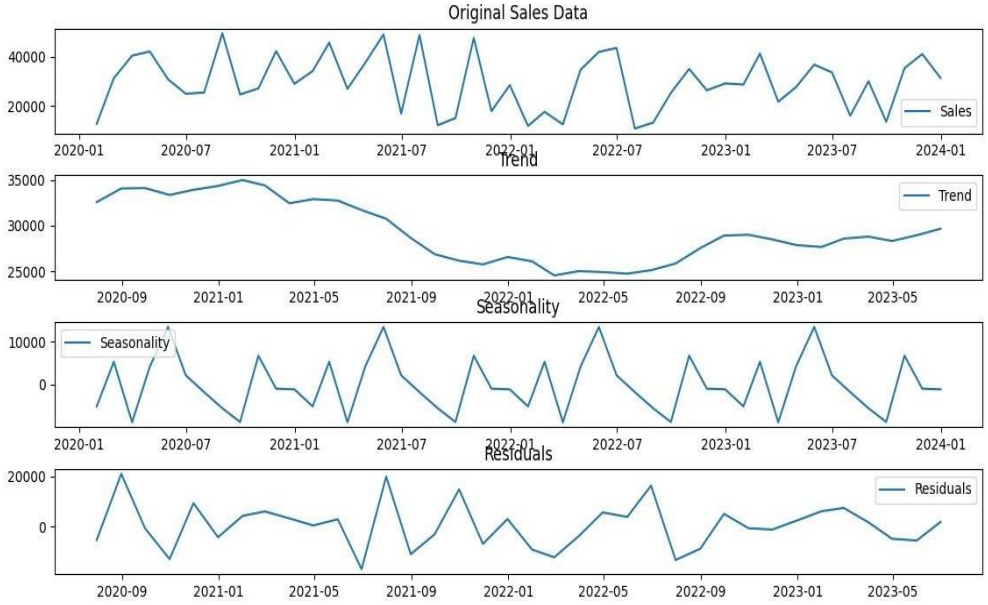
```
plt.plot(rolling_mean, label='Rolling Mean (12 months)', color='red')plt.plot(rolling_std, label='Rolling Std (12 months)', color='green') plt.legend()
plt.grid(True)
plt.show()
```

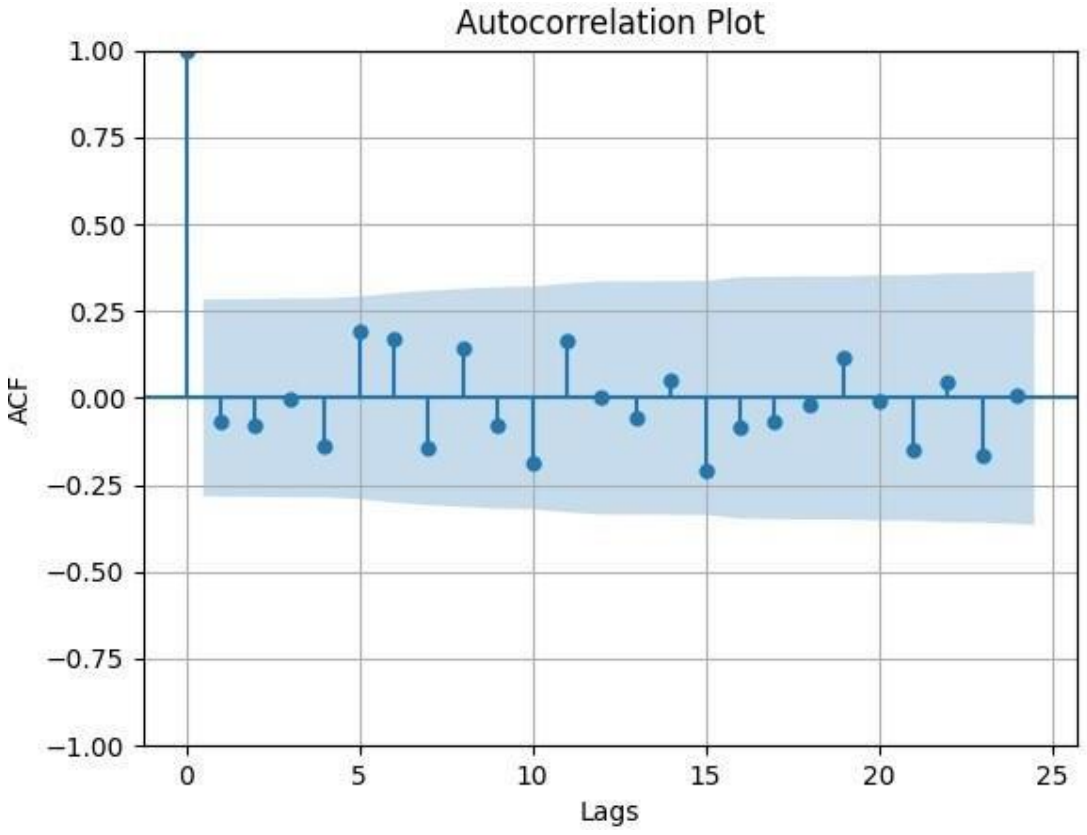
OUTPUT

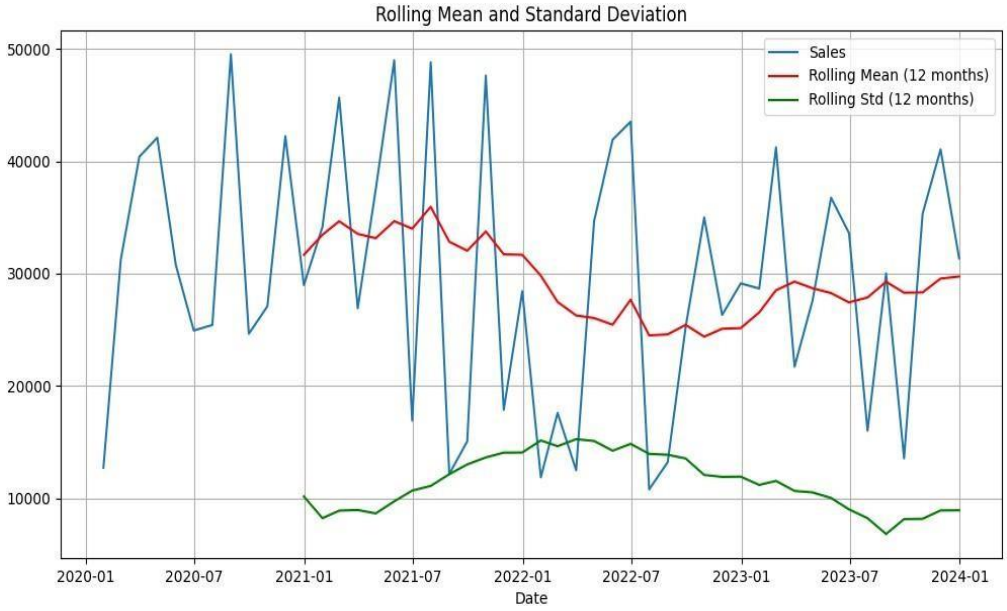
Sales

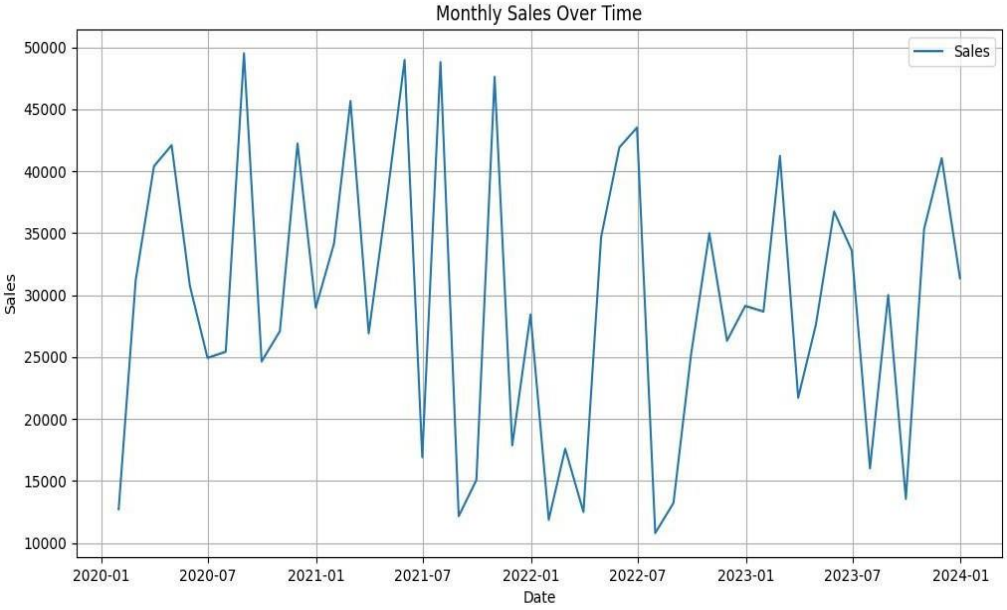
count	48.000000
mean	29559.562500
std	11393.671539
min	10797.000000
25%	20761.500000
50%	29056.000000
75%	38202.500000
max	49512.000000

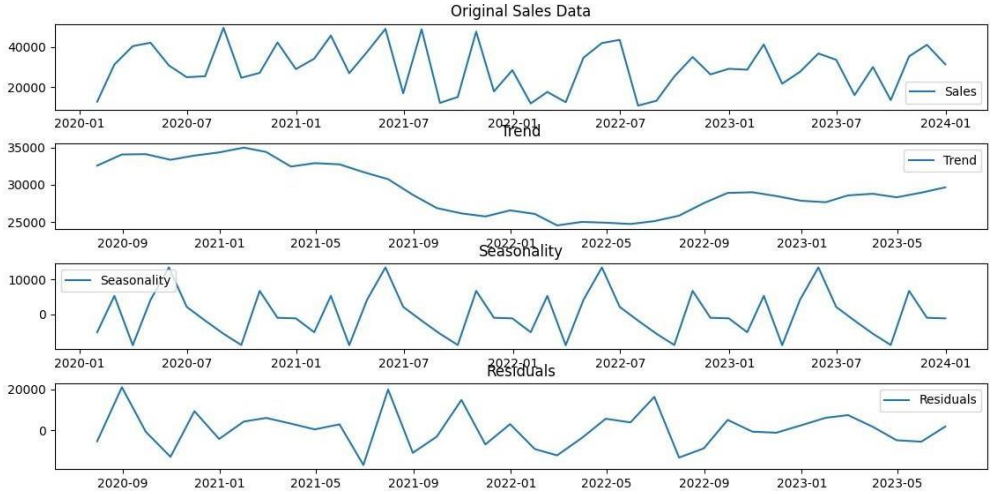



Original Sales Data
Sales
Trend
Seasonality
Residuals


Autocorrelation Plot
ACF
Lags


Rolling Mean and Standard Deviation
Sales
Rolling Mean (12 months)
Rolling Std (12 months)
50000
40000
30000
20000
10000
2020-01
2020-07
2021-01
2021-07
2022-01
2022-07
2023-01
2023-07
2024-01
Date


Monthly Sales Over Time
Sales
50000
45000
40000
35000
30000
25000
20000
15000
10000
2020-01
2020-07
2021-01
2021-07
2022-01
2022-07
2023-01
2023-07
2024-01
Date



RESULT:

Thus, to perform Time Series Analysis and apply the various visualization techniques in python using PyCharm is done and executed successfully.

Ex.No: 6	PERFORM DATA ANALYSIS AND REPRESENTATION ON A MAP USING VARIOUS MAP DATASETS WITH MOUSE ROLLOVER EFFECT ,USER INTERACTION
Date:	

AIM

To perform data analysis and representation on a map using various map datasets with mouserollover effect ,user interaction in python using PyCharm.

ALGORITHM

STEP 1: Import required libraries.
STEP 2: Generate random data.
STEP 3:Create a Dataframe..

STEP 4: Save the DataFrame to a CSV fileSTEP 5: Load the generated CSV file
STEP 6:Create a base map
STEP 7:Add data points to the map
STEP 8:Save the map as an HTML file

PROGRAM

```
import pandas as
pdimport numpy as
np
# Generate random
data
np.random.seed(42)
num_points = 100
latitude = np.random.uniform(37.5, 38.5, num_points)
longitude = np.random.uniform(-123, -121, num_points)
value = np.random.randint(1, 100, num_points)
# Create a DataFrame
df = pd.DataFrame({'Latitude': latitude, 'Longitude': longitude, 'Value': value})# Save the DataFrame to a CSV file
df.to_csv('map_data.csv', index=False)
import folium
# Load the generated CSV file
df =
pd.read_csv('map_data.csv') #
Create a base map
base_map = folium.Map(location=[df['Latitude'].mean(), df['Longitude'].mean()], zoom_start=10)

# Add data points to the map for index, row in df.iterrows():
   popup_text = f"Value: {row['Value']}"


   folium.Marker([row['Latitude'], row['Longitude']],
```

```
popup=popup_text).add_to(base_map) # Save the map as an HTML file
base_map.save('interactive_map.html')
# Import necessary libraries import pandas as pd
import folium
# Load the data
df = pd.read_csv('map_data.csv') # Display basic statistics
print("Basic Statistics:")
print(df.describe())
# Create a base map
base_map = folium.Map(location=[df['Latitude'].mean(), df['Longitude'].mean()], zoom_start=10)
# Add data points to the map
for index, row in df.iterrows():
   popup_text = f"Value: {row['Value']}"
   folium.Marker([row['Latitude'], row['Longitude']], popup=popup_text).add_to(base_map) # Display the map
base_map
```

OUTPUT

Basic Statistics:

	LATITUDE	**LONGITUDE**	**VALUE**
count	100.000000	100.000000	100.000000
mean	37.970181	-122.004337	50.670000
std	0.297489	0.586223	28.640247
min	37.505522	-122.986096	2.000000
25%	37.693201	-122.515991	27.000000
50%	37.964142	-121.988750	47.500000
75%	38.230203	-121.467633	72.750000
max	38.486887	-121.028699	99.000000

RESULT

Thus,to perform data analysis and representation on a map using various map datasets with mouse rollover effect ,user interaction in python using PyCharm is done and executed successfully.

Ex. No:7	BUILD CARTOGRAPHIC VISUALIZATION FOR MULTIPLE DATASETS INVOLVING VARIOUS COUNTRIES OF THE WORLD; STATES AND DISTRICTS IN INDIA
Date:	

AIM

To build cartographic visualization for multiple datasets involving various countriesof theworld; states and districts in India etc. in pandas using PyCharm.

ALGORITHM

STEP 1: Import required libraries.

STEP 2: Replace with actual data paths and datasets.STEP 3: Function to load and merge data with maps.STEP 4: Load world and India maps.
STEP 5: Merge data with maps.

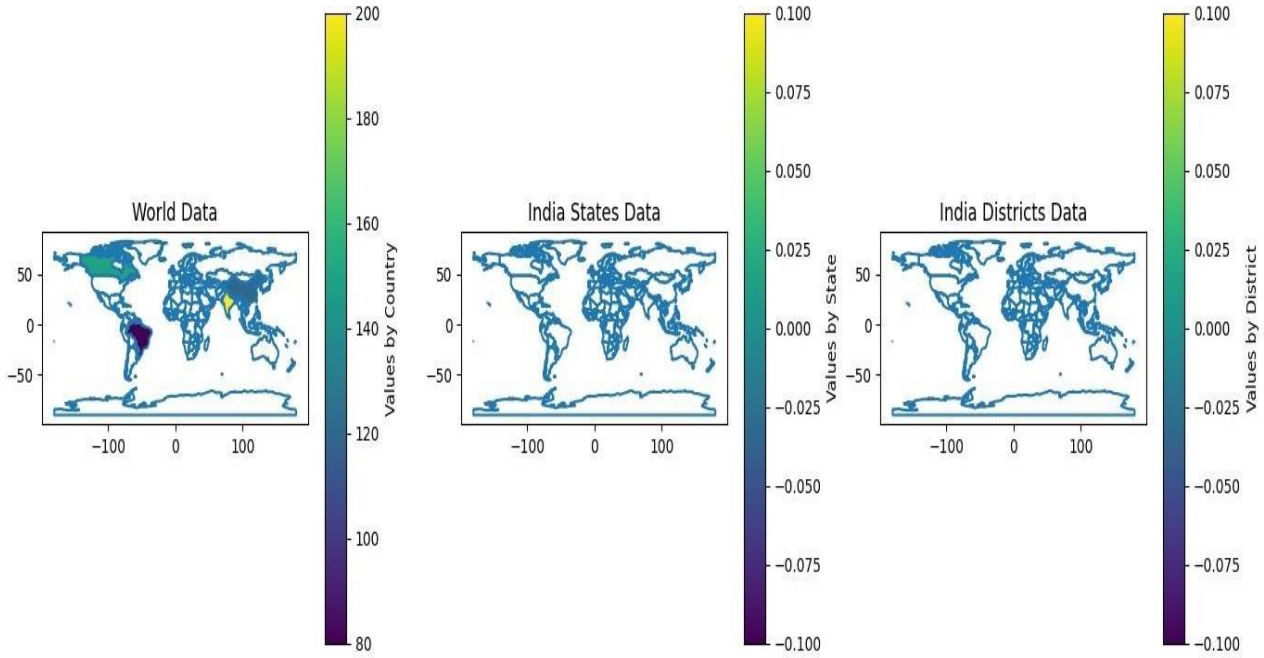
STEP 6: Function to plot cartographic visualizations for World Map, India States Map and India Districts Map.

PROGRAM

```
import pandas as pd
import geopandas as
gpd
import matplotlib.pyplot as
plt world_data =
pd.DataFrame({
  'Country': ['USA', 'Canada', 'India', 'Brazil',
  'China'],'Value': [100, 150, 200, 80, 120]
})
india_states_data = pd.DataFrame({
  'State': ['Maharashtra', 'Karnataka', 'Tamil Nadu', 'Uttar Pradesh',
  'Gujarat'],'Value': [50, 75, 60, 40, 30]
})
india_districts_data = pd.DataFrame({
  'District': ['Mumbai', 'Bengaluru', 'Chennai', 'Lucknow', 'Ahmedabad'],
  'Value': [20, 30, 25, 15, 10]
})
world_map = gpd.read_file(gpd.datasets.get_path('naturalearth_lowres'))
india_states_map = gpd.read_file(gpd.datasets.get_path('naturalearth_lowres'))
india_districts_map = gpd.read_file(gpd.datasets.get_path('naturalearth_lowres'))
world_data_geo = world_map.merge(world_data, how='left', left_on='name',
right_on='Country')
india_states_data_geo = india_states_map.merge(india_states_data, how='left',
left_on='name', right_on='State')
```

```
india_districts_data_geo = india_districts_map.merge(india_districts_data, how='left', left_on='name', right_on='District')
fig, axs = plt.subplots(1, 3, figsize=(15, 5))
axs[0].set_title('World Data')
world_data_geo.boundary.plot(ax=axs[0])
world_data_geo.plot(column='Value', ax=axs[0], legend=True, legend_kwds={'label': "Values by Country"})
axs[1].set_title('India States Data')
india_states_data_geo.boundary.plot(ax=axs[1])
india_states_data_geo.plot(column='Value', ax=axs[1], legend=True, legend_kwds={'label': "Values by State"})
axs[2].set_title('India Districts Data')
india_districts_data_geo.boundary.plot(ax=axs[2])
india_districts_data_geo.plot(column='Value', ax=axs[2], legend=True, legend_kwds={'label': "Values by District"})
plt.tight_layout()
plt.show()
```

OUTPUT



RESULT

Thus, the python program to perform to build cartographic visualization for multiple datasets involving various countries of theworld; states and districts in India etc. inpython using PyCharm is executed successfully.

Ex. No:8	PERFORM EDA ON WINE QUALITY DATA SET
Date:	

AIM

To perform EDA on Wine Quality dataset in pandas using PyCharm.

ALGORITHM

STEP 1**:** Generate a Sample Wine Quality Dataset.
STEP 2: Perform EDA on the Generated Dataset.
STEP 3: Display basic information about the dataset.
STEP 4: Display summary statistics.
STEP 5: Check for missing values.
STEP 6: Plot the distribution of wine quality.
STEP 7: Plot the correlation matrix.
STEP 8: Plot pair plot for selected features.
STEP 9: Plot boxplot for wine quality and alcohol content.

PROGRAM

```
import pandas as
pdimport numpy as
np
np.random.seed(42)
data = {
   'fixed acidity': np.random.uniform(5, 15, 100),
   'volatile acidity': np.random.uniform(0.1, 1.5, 100),
   'citric acid': np.random.uniform(0, 1, 100),
   'residual sugar': np.random.uniform(0, 10, 100),
   'alcohol': np.random.uniform(8, 15, 100),
   'quality': np.random.randint(1, 11, 100)
}
wine_df = pd.DataFrame(data)
wine_df.to_csv('wine_quality_dataset.csv', index=False)
import matplotlib.pyplot as plt
import seaborn as
sns # Load the
dataset
wine_df = pd.read_csv('wine_quality_dataset.csv')
print(wine_df.info())
print(wine_df.describe())
print(wine_df.isnull().sum())
sns.countplot(x='quality', data=wine_df)
plt.title('Distribution of Wine Quality')
plt.show()
```

```
correlation_matrix = wine_df.corr()
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Matrix')
plt.show()
selected_features = ['alcohol', 'volatile acidity', 'citric acid', 'residual sugar', 'quality']
sns.pairplot(wine_df[selected_features], hue='quality')
plt.title('Pairplot of Selected Features')plt.show()
plt.figure(figsize=(10, 6))
sns.boxplot(x='quality', y='alcohol', data=wine_df)
plt.title('Boxplot of Alcohol Content by Wine Quality')
plt.show()
```

OUTPUT

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 6 columns):
```

#	Column	Non-Null Count	Dtype
----	--------	----------	-------
0	fixed acidity	100 non-null	float64
1	volatile acidity	100 non-null	float64
2	citric acid	100 non-null	float64
3	residual sugar	100 non-null	float64
4	alcohol	100 non-null	float64
5	quality	100 non-null	int64

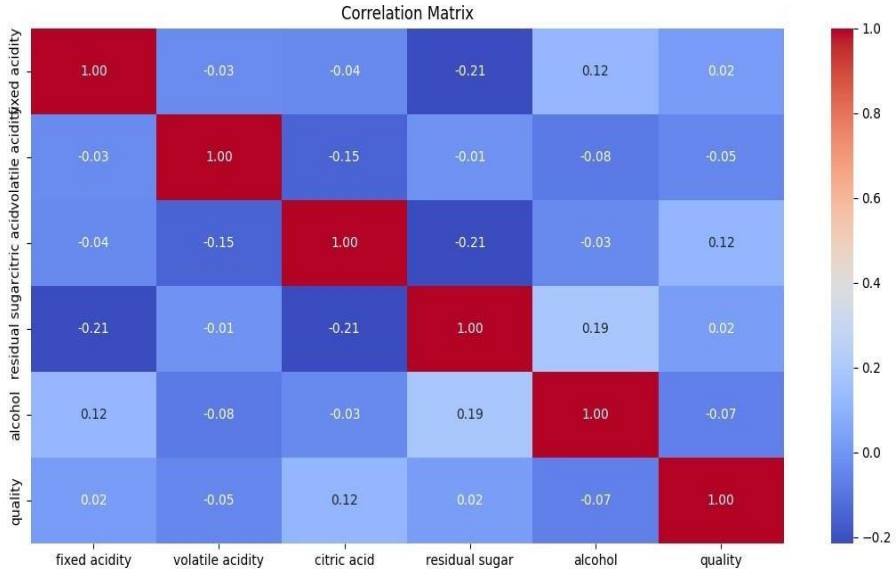
```
dtypes: float64(5), int64(1)
memory usage: 4.8 KB
None
```

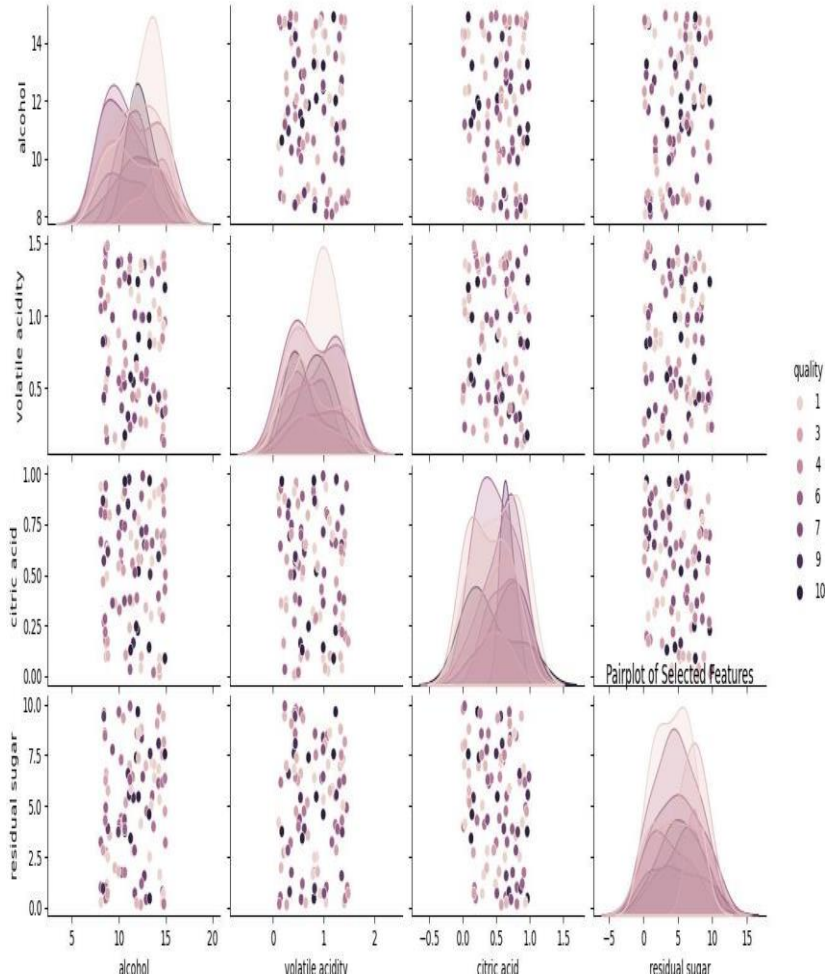
	fixed acidity	volatile acidity	...	alcohol	quality
count	100.000000	100.000000	...	100.000000	100.000000
mean	9.701807	0.796964	...	11.612321	4.750000
std	2.974894	0.410356	...	2.230206	2.900279
min	5.055221	0.109733	...	8.075864	1.000000
25%	6.932008	0.438806	...	9.847544	2.000000
50%	9.641425	0.807875	...	11.677796	5.000000
75%	12.302031	1.172657	...	13.578189	7.000000
max	14.868869	1.479911	...	14.950754	10.000000

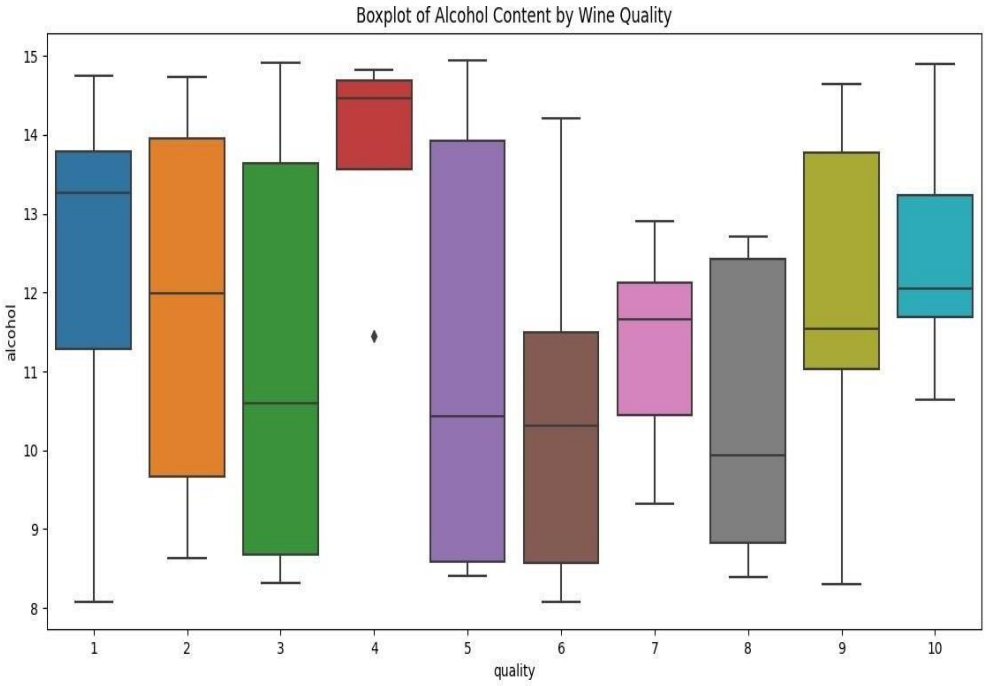
[8 rows x 6 columns]

fixed acidity	0
volatile acidity	0
citric acid	0
residual sugar	0
alcohol	0
quality	0

dtype: int64


Correlation Matrix
fixed acidity
volatile acidity
citric acid
residual sugar
alcohol
quality
1.00 -0.03 -0.04 -0.21 0.12 0.02
-0.03 1.00 -0.15 -0.01 -0.08 -0.05
-0.04 -0.15 1.00 -0.21 -0.03 0.12
-0.21 -0.01 -0.21 1.00 0.19 0.02
0.12 -0.08 -0.03 0.19 1.00 -0.07
0.02 -0.05 0.12 0.02 -0.07 1.00
1.0
0.8
0.6
0.4
0.2
0.0
−0.2


alcohol
volatile acidity
citric acid
residual sugar
quality
1
3
4
6
7
9
10
Pairplot of Selected Features



RESULT

Thus, the program to perform EDA on Wine Quality dataset in python using PyCharm has been executed successfully.

Ex. No:9	USE A CASE STUDY ON A DATA SET AND APPLY THE VARIOUS EDA AND VISUALIZATION TECHNIQUESAND PRESENT AN ANALYSISREPORT
Date:	

AIM

Perform a case study on a data set and apply the various EDA and visualization techniques and present an analysis report in pandas using PyCharm.

ALGORITHM

STEP 1: Create CSV Dataset:
STEP 2: Load the Dataset
STEP 3: Exploratory Data Analysis (EDA)
a. Basic Dataset Exploration
b. Handle Missing Values
c. Explore Categorical Variables
d. STEP 4: Visualizations
a. Distribution of Age:
b. Performance Ratings by Department:
STEP 5: Analysis Report

a. Demographic Overview

The age of employees in the company is normally distributed, with the majority falling between 22 and 60 years.

The workforce is well-distributed among different departments, with the highest number in Sales.

b. Performance Analysis

The performance ratings are generally high across all departments.
There is a positive correlation between years of experience and performance ratings.

c. Salary Analysis

Explore salary distribution and identify any anomalies or disparities.
STEP 6: Insights and Recommendations

a. Identify High-Performing Employees

Recognize and reward high-performing employees based on performance ratings.

b. Training Opportunities

Identify departments with lower performance ratings and provide targeted training programs.

c. Address Salary Disparities

Investigate and address any salary disparities to ensure fairness.

PROGRAM

```
import pandas as pd import
numpy as np import
seaborn as sns
import matplotlib.pyplot as plt
np.random.seed(42) employee_id =
range(1, 101)
age = np.random.randint(22, 60, size=100)
gender = np.random.choice(['Male', 'Female'], size=100)
department = np.random.choice(['HR', 'Sales', 'IT', 'Marketing'], size=100)
years_of_experience = np.random.randint(1, 15, size=100) performance_rating =
np.random.randint(1, 6, size=100)
salary = np.random.randint(40000, 120000, size=100)
employee_data = pd.DataFrame({
  'EmployeeID': employee_id,'Age':
  age,
  'Gender': gender, 'Department':
  department,
  'YearsOfExperience': years_of_experience,
  'PerformanceRating': performance_rating, 'Salary':
  salary
})
employee_data.to_csv('employee_data.csv', index=False)
employee_data = pd.read_csv('employee_data.csv')
print(employee_data.info()) print(employee_data.describe())
print(employee_data.isnull().sum())
department_distribution = employee_data['Department'].value_counts()
print(department_distribution)
import matplotlib.pyplot as pltimport seaborn as sns
sns.histplot(employee_data['Age'], bins=20, kde=True)
```

```
plt.title('Distribution of Employee Ages')
plt.show()
sns.boxplot(x='Department', y='PerformanceRating', data=employee_data)
plt.title('Performance Ratings by Department')
plt.show()
```

OUTPUT

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99

Data columns (total 7 columns):
```

#	Column	Non-Null Count	Dtype
0	EmployeeID	100 non-null	int64
1	Age	100 non-null	int64
2	Gender	100 non-null	object
3	Department	100 non-null	object
4	Years Of Experience	100 non-null	int64
5	Performance Rating	100 non-null	int64
6	Salary	100 non-null	int64

```
dtypes: int64(5), object(2)
memory usage: 5.6+ KB None
```

	EMPLOYEE ID	AGE	PERFORMANCE	...	SALARY
count	100.000000	100.000000	100.000000	...	100.000000
mean	50.500000	40.060000	2.96000	...	81430.86000
std	29.011492	10.688255	1.44194	...	22497.541898
min	1.000000	22.000000	1.00000	...	40412.000000
25%	25.750000	30.000000	2.00000	...	62624.750000
50%	50.500000	41.500000	3.00000	...	80856.500000
75%	75.250000	45.000000	4.00000	...	100741.500000
max	100.000000	59.000000	5.00000	...	119605.000000

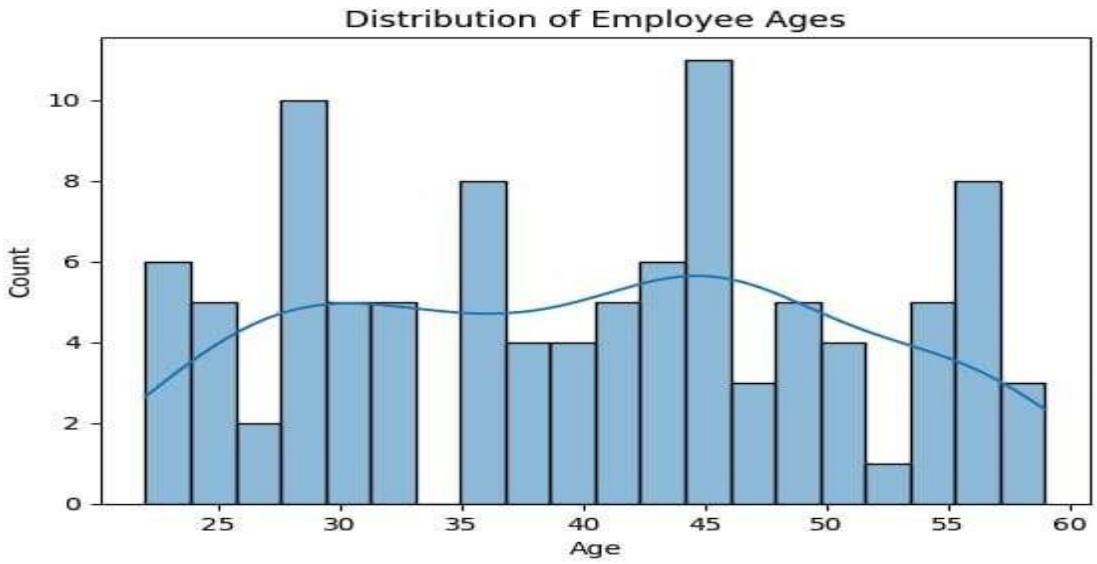
```
[8 rows x 5 columns]
```

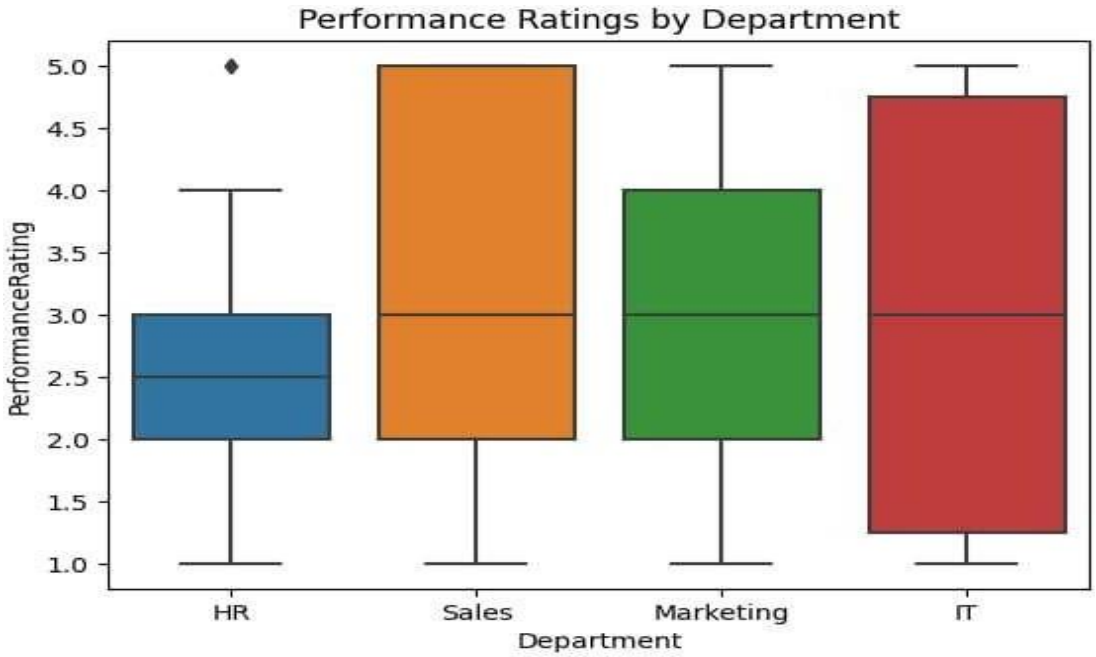
EmployeeID	0
Age	0
Gender	0
Department	0
Years Of Experience	0
Performance Rating	0
Salary	0

dtype: int64

Department	
Marketing	31
IT	26
HR	22
Sales	21

Name: count, dtype: int64





RESULT

Thus, a case study on a data set and applying the various EDA and visualization techniques and present an analysis report in python using PyCharm is done and executed successfully.