# S.G.T.B.I.M.I.T

## BCA - BCAP - 212 - DATA SCIENCE

## PRACTICAL FILE QUESTIONS

| S. No. | Detailed Statement |
|---|---|
| 1 | Create 2 numpy arrays with 5 elements using arange() and linspace() and display the implement the concept of slicing on them. |
| 2 | Create two 2-d arrays and perform addition, subtraction and multiplication on these arrays. Print the value of bordered elements of both matrices. |
| 3 | Create two pandas series from a dictionary of values and an ndarray and display the values from 2nd to 5th index. Print the index, minimum and maximum values in the first series. |
| 4 | Create a Series and print all the elements that are above 75th percentile. Print minimum, maximum, and sum of series using aggregate() |
| 5 | Series objects Temp1, temp2, temp3, temp 4 stores the temperature of days of week 1, week 2, week 3, week 4. Write a script to:-<br>a. Print average temperature per week<br>b. Print the average temperature of the entire month |
| 6 | Create a series containing 2 NaN values. Perform check for null values and then replace the null values with 0. Perform any 5 statistical functions on the series. |
| 7 | Two Series object, Population stores the details of four metro cities of India and another object AvgIncome store the total average income reported in four years in these cities. Calculate income per capita for each of these metro cities. |
| 8 | Given two series S1 and S2.<br>S1 S2<br>A 39 A 10<br>B 41 B 10<br>C 42 D 10<br>D 44 F 10<br>Find the output for following python pandas statements?<br>a. S1[ : 2]*100<br>b. S1 * S2<br>c. S2[ : : -1]*10 |
| 9 | Write a program to create a Series having 10 random numbers in the range of 10 and 20 |
| 10 | Consider a series object s10 that stores the number of students in each section of class 12 as shown below. First two sections have been given task for selling tickets @ Rs.100/- per ticket as a part of social experiment. Write code to create the series and display how much section A and B have collected. A-39, B- 31, C- 32, D- 34, E- 35 |
| 11 | Write a program to create a DataFrame to store weight, age and name of |

Nikhil Kumar
BCA 4D
09091102022

| | |
|---|---|
| | three people. Print the DataFrame and its transpose. Add 5 rows in the dataframe through code. Rename the Weight column as Wgt. And then print the index names, column names and total amount of data. Sort the dataframe on the basis of age. |
| 12 | Create a DataFrame having age, name, weight of five students. Print the dataframe using head(). Modify the weight of student in first and 4th row. Display only the weight of first and fourth rows before and after modification. |
| 13 | Create a DataFrame based on E-Commerce data and generate mean, mode, median. |
| 14 | Write a Program to create a CSV file with student data containing 10 rows. create its DataFrame and use describe() to display its statistics. Write all the steps and definition of all the statistical functions displayed. |
| 15 | Consider the DataFrame QtrSales where each row contains the item category, item name and expenditure and group the rows by category, and print the average expenditure per category |
| 16 | Write a program to implement pivot() and pivot-table() on a DataFrame |
| 17 | Write a program to find mean absolute deviation on a DataFrame. |
| 18 | Create a DataFrame based on employee data and generate quartile and variance. |
| 19 | Program to implement Skewness on Random data. |
| 20 | Create a DateFrame on any Data and compute statistical function of Kurtosis. |
| 21 | CASE STUDY ON :CRIME_BY_STATE_RT.CSV<br>crime_by_state_rt.zip<br>21.1: Find the 75% percentile for data<br>21.2: Apply aggregate function to find count, min , max and sum .<br>21.3: Perform group by on State and Year<br>21.4: Calculate the number of entries year wise<br>21.5: Create a pie chart for all states<br>21.6: Plot the no. of murders state-wise<br>21.7: Check the data type of all column<br>21.8: Check the robbery column for NULL values.<br>if exist replace with 0<br>21.9: Select the data set where robbery greater than 10 and year= 2001<br>21.10: describe () the dataset<br>21.11: Perform sorting on Arson and Robbery<br>21.12: Perform renaming on two columns |
| 22 | CASE STUDY ON: investigating clinical data<br>investigating clinical data.csv<br>22.1: describe () the dataset<br>22.2: Perform renaming on two columns<br>22.3: Is there any relation between the age of the patient and having a delay on the date of appointment.<br>22.4: Is there any relation between gender and delaying appointments? (Show by plotting bar graph) |

Nikhil Kumar
BCA 4D
09091102022

| | |
|---|---|
| | 22.5: Do people delay their appointments when they have no scholarship? (Show by plotting)<br>22.6: Is there any relationship between gender having scholarships and delaying the appointment? (show by plotting)<br>22.7: Which gender has more appointments in which illness? (Show by plotting)<br>22.8: Does a person suffering from alcoholism tend to delay appointments? (Show by plotting)<br>22.9: Perform group by and count on neighborhood<br>22.10: Perform agg() function on dataset. |
| 23 | CASE STUDY ON: Credit card transaction<br>Credit card transactions - India - Simple.csv<br>23.1: Show the head and visualize the basic stats for all columns.<br>23.2: Display pivot table to create a summary for the total amount spent<br>a) month b) city<br>23.3: Analyze the impact of gender to study consumer behavior<br>23.4: Check for NULL values in columns.If they exist replace them with appropriate values.<br>23.5: Analyze the relationship type between expense type and amount.<br>23.6: Analyze the spending habits by city and gender.<br>23.7: Implement Skewness and kurtosis on data<br>Draw plots along with an appropriate question |
| 24 | CASE STUDY ON PROJECT TOPIC |
| 25 | RESEARCH PAPER |

Nikhil Kumar
BCA 4D
09091102022

Q1

```
import numpy as np

# Using arange() to create an array with 5 elements
array_arange = np.arange(1, 10, 2)  # Start at 1, stop before 10, step by 2
print("Array created using arange():", array_arange)

# Using linspace() to create an array with 5 elements
array_linspace = np.linspace(1, 10, 5)  # Start at 1, stop at 10, with 5 elements
print("Array created using linspace():", array_linspace)

# Slicing the arrays
slice_arange = array_arange[1:4]  # Get elements from index 1 to index 3 (exclusive)
print("Sliced array using arange():", slice_arange)

slice_linspace = array_linspace[2:]  # Get elements from index 2 to the end
print("S")
```

OUTPUT

```
========== RESTART: C:/Users/natur/OneDrive/Desktop/array.py =
Array created using arange(): [1 3 5 7 9]
Array created using linspace(): [ 1.    3.25  5.5   7.75 10.  ]
Sliced array using arange(): [3 5 7]
S
```

Nikhil Kumar
BCA 4D
09091102022

Q2

```
import numpy as np

array1 = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
array2 = np.array([[9, 8, 7], [6, 5, 4], [3, 2, 1]])

addition_result = array1 + array2
subtraction_result = array1 - array2
multiplication_result = array1 * array2

bordered_elements_array1 = array1[[0, -1], :]
bordered_elements_array2 = array2[:, [0, -1]]

print("Array 1:")
print(array1)
print("Array 2:")
print(array2)
print("Addition result:")
print(addition_result)
print("Subtraction result:")
print(subtraction_result)
print("Multiplication result:")
print(multiplication_result)
print("Bordered elements of Array 1:")
print(bordered_elements_array1)
print("Bordered elements of Array 2:")
```

```
=============== RESTART: C:/Use
Array 1:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Array 2:
[[9 8 7]
 [6 5 4]
 [3 2 1]]
Addition result:
[[10 10 10]
 [10 10 10]
 [10 10 10]]
Subtraction result:
[[-8 -6 -4]
 [-2  0  2]
 [ 4  6  8]]
Multiplication result:
[[ 9 16 21]
 [24 25 24]
 [21 16  9]]
Bordered elements of Array 1:
[[1 2 3]
 [7 8 9]]
Bordered elements of Array 2:
[[9 7]
 [6 4]
 [3 1]]
```

Nikhil Kumar
BCA 4D
09091102022

```
print(bordered_elements_array2)
```

Q3

```
import pandas as pd

import numpy as np


dict_data = {'A': 10, 'B': 20, 'C': 30, 'D': 40, 'E': 50}


series_dict = pd.Series(dict_data)

print("Series from dictionary:")

print(series_dict)


ndarray_data = np.array([1, 2, 3, 4, 5])


series_ndarray = pd.Series(ndarray_data)

print("\nSeries from ndarray:")

print(series_ndarray)


print("\nValues from 2nd to 5th index of Series from dictionary:")

print(series_dict[1:5])  # Indexing starts from 0, so 1 corresponds to the 2nd index and 5 corresponds to the 6th index

print("Values from 2nd to 5th index of Series from ndarray:")

print(series_ndarray[1:5])


print("\nIndex, minimum, and maximum values of the Series from dictionary:")

print("Index:", series_dict.index.tolist())

print("Minimum value:", series_dict.min())

print("Maximum value:", series_dict.max())
```

Nikhil Kumar
BCA 4D
09091102022

```
========== RESTART: C:/Users/natur/OneDrive/Desktop/array.py
Series from dictionary:
A    10
B    20
C    30
D    40
E    50
dtype: int64

Series from ndarray:
0    1
1    2
2    3
3    4
4    5
dtype: int32

Values from 2nd to 5th index of Series from dictionary:
B    20
C    30
D    40
E    50
dtype: int64
Values from 2nd to 5th index of Series from ndarray:
1    2
2    3
3    4
4    5
dtype: int32

Index, minimum, and maximum values of the Series from dictionary:
Index: ['A', 'B', 'C', 'D', 'E']
Minimum value: 10
Maximum value: 50
```

Nikhil Kumar
BCA 4D
09091102022

Q4

```
import pandas as pd

import numpy as np


data = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]

series = pd.Series(data)

print("Original Series:")

print(series)


percentile_75 = series.quantile(0.75)

elements_above_75th_percentile = series[series > percentile_75]

print("\nElements above the 75th percentile:")

print(elements_above_75th_percentile)


aggregates = series.agg(['min', 'max', 'sum'])

print("\nAggregate results - Minimum, Maximum, Sum:")

print(aggregates)
```

```
---------- RESTART: C:/Users/nakul/Onl
Original Series:
0      10
1      20
2      30
3      40
4      50
5      60
6      70
7      80
8      90
9     100
dtype: int64

Elements above the 75th percentile:
7      80
8      90
9     100
dtype: int64

Aggregate results - Minimum, Maximum, Sum:
min      10
max     100
sum     550
dtype: int64
```

Nikhil Kumar
BCA 4D
09091102022

Q5

```
import pandas as pd

temp_data = {
    'Week 1': [22, 25, 23, 24, 21],  # Sample temperatures for Week 1
    'Week 2': [26, 27, 28, 25, 24],  # Sample temperatures for Week 2
    'Week 3': [20, 22, 21, 23, 24],  # Sample temperatures for Week 3
    'Week 4': [24, 25, 26, 27, 28]   # Sample temperatures for Week 4
}

Temp1 = pd.Series(temp_data['Week 1'])
Temp2 = pd.Series(temp_data['Week 2'])
Temp3 = pd.Series(temp_data['Week 3'])
Temp4 = pd.Series(temp_data['Week 4'])

avg_temp_per_week = {
    'Week 1': Temp1.mean(),
    'Week 2': Temp2.mean(),
    'Week 3': Temp3.mean(),
    'Week 4': Temp4.mean()
}
print("Average temperature per week:")
for week, avg_temp in avg_temp_per_week.items():
    print(f"{week}: {avg_temp}")

all_temps = pd.concat([Temp1, Temp2, Temp3, Temp4])
avg_temp_month = all_temps.mean()
```

Nikhil Kumar
BCA 4D
09091102022

```
print("\nAverage temperature of the entire month:", avg_temp_month)
```

```
Average temperature per week:
Week 1: 23.0
Week 2: 26.0
Week 3: 22.0
Week 4: 26.0

Average temperature of the entire month: 24.25
```

Q6

```
import pandas as pd

import numpy as np


data = [10, np.nan, 30, np.nan, 50]

series = pd.Series(data)

print("Original Series:")

print(series)


null_check = series.isnull()

print("\nNull check:")

print(null_check)


series_filled = series.fillna(0)

print("\nSeries with null values replaced by 0:")

print(series_filled)


mean_value = series_filled.mean()

median_value = series_filled.median()

std_deviation = series_filled.std()

min_value = series_filled.min()
```

Nikhil Kumar
BCA 4D
09091102022

```
max_value = series_filled.max()


print("\nStatistical functions on the Series:")
print("Mean:", mean_value)
print("Median:", median_value)
print("Standard Deviation:", std_deviation)
print("Minimum Value:", min_value)
print("Maximum Value:", max_value)
```

```
Original Series:
0    10.0
1     NaN
2    30.0
3     NaN
4    50.0
dtype: float64

Null check:
0    False
1     True
2    False
3     True
4    False
dtype: bool

Series with null values replaced by 0:
0    10.0
1     0.0
2    30.0
3     0.0
4    50.0
dtype: float64

Statistical functions on the Series:
Mean: 18.0
Median: 10.0
Standard Deviation: 21.6794833886788
Minimum Value: 0.0
Maximum Value: 50.0
```

Nikhil Kumar
BCA 4D
09091102022

Q7

```
import pandas as pd


population_data = {'Delhi': 20000000, 'Mumbai': 22000000, 'Kolkata': 15000000, 'Chennai': 12000000}

avg_income_data = {'Delhi': 800000, 'Mumbai': 900000, 'Kolkata': 700000, 'Chennai': 600000}


Population = pd.Series(population_data)

AvgIncome = pd.Series(avg_income_data)


income_per_capita = AvgIncome / Population

income_per_capita = income_per_capita.fillna(0)  # Replace NaN values with 0 if any


print("Income per capita for each metro city:")

print(income_per_capita)
```

```
========== RESTART: C:/Users/natur/Or
Income per capita for each metro city:
Delhi      0.040000
Mumbai     0.040909
Kolkata    0.046667
Chennai    0.050000
dtype: float64
```

Nikhil Kumar
BCA 4D
09091102022

Q8

```
import pandas as pd

data_S1 = {'A': 39, 'B': 41, 'C': 42, 'D': 44}
data_S2 = {'A': 10, 'B': 10, 'D': 10, 'F': 10}

S1 = pd.Series(data_S1)
S2 = pd.Series(data_S2)

output_a = S1[:2] * 100

output_b = S1 * S2

output_c = S2[::-1] * 10

print("Output of S1[:2]*100:")
print(output_a)
print("\nOutput of S1 * S2:")
print(output_b)
print("\nOutput of S2[::-1]*10:")
print(output_c)
```

```
Output of S1[:2]*100:
A    3900
B    4100
dtype: int64

Output of S1 * S2:
A    390.0
B    410.0
C      NaN
D    440.0
F      NaN
dtype: float64

Output of S2[::-1]*10:
F    100
D    100
B    100
A    100
dtype: int64
```

Nikhil Kumar
BCA 4D
09091102022

Q9

```
import pandas as pd

import numpy as np


# Generate 10 random numbers in the range of 10 and 20

random_numbers = np.random.randint(10, 21, size=10)


# Create a Pandas Series from the random numbers

random_series = pd.Series(random_numbers)


print("Random Series:")

print(random_series)
```

```
Random Series:
0    17
1    14
2    19
3    12
4    11
5    13
6    12
7    11
8    13
9    17
dtype: int32
```

Nikhil Kumar
BCA 4D
09091102022

Q10

```
import pandas as pd

# Create the Series with the number of students in each section
data_s10 = {'A': 39, 'B': 31, 'C': 32, 'D': 34, 'E': 35}
s10 = pd.Series(data_s10)

# Calculate the amount collected by sections A and B
amount_collected_A = s10['A'] * 100
amount_collected_B = s10['B'] * 100

print("Amount collected by Section A:", amount_collected_A)
print("Amount collected by Section B:", amount_collected_B)
```

```
========== RESTART: C:/Users/natur/
Amount collected by Section A: 3900
Amount collected by Section B: 3100
```

Q11

```
import pandas as pd

data = {'Name': ['Alice', 'Bob', 'Charlie'], 'Weight': [65, 70, 75], 'Age': [25, 30, 35]}
df = pd.DataFrame(data)

print("Original DataFrame:")
print(df)

print("\nDataFrame Transpose:")
print(df.T)
```

Nikhil Kumar
BCA 4D
09091102022

```
new_data = {'Name': ['David', 'Emily', 'Frank', 'Grace', 'Henry'], 'Weight': [80, 85, 90, 95, 100], 'Age': [40, 45, 50, 55, 60]}

df = df.append(pd.DataFrame(new_data), ignore_index=True)


df.rename(columns={'Weight': 'Wgt'}, inplace=True)


print("\nIndex Names:")

print(df.index)

print("\nColumn Names:")

print(df.columns)


print("\nTotal Amount of Data:")

print(df.size)


df.sort_values(by='Age', inplace=True)

print("\nSorted DataFrame based on Age:")

print(df)
```

```
========== RESTART: C:/User
Original DataFrame:
      Name  Weight  Age
0    Alice      65   25
1      Bob      70   30
2  Charlie      75   35

DataFrame Transpose:
            0    1        2
Name    Alice  Bob  Charlie
Weight     65   70       75
Age        25   30       35
```

Nikhil Kumar
BCA 4D
09091102022

Q12

```
import pandas as pd

data = {'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Emily'],
        'Age': [20, 21, 22, 23, 24],
        'Weight': [55, 60, 65, 70, 75]}
df = pd.DataFrame(data)

print("Initial DataFrame:")
print(df.head())

print("\nWeight of first and fourth rows before modification:")
print("First Row Weight:", df.loc[0, 'Weight'])
print("Fourth Row Weight:", df.loc[3, 'Weight'])

df.loc[0, 'Weight'] = 58  # Modify weight of student in the first row
df.loc[3, 'Weight'] = 73  # Modify weight of student in the fourth row

print("\nWeight of first and fourth rows after modification:")
print("First Row Weight:", df.loc[0, 'Weight'])
print("Fourth Row Weight:", df.loc[3, 'Weight'])
```

```
Initial DataFrame:
      Name  Age  Weight
0    Alice   20      55
1      Bob   21      60
2  Charlie   22      65
3    David   23      70
4    Emily   24      75

Weight of first and fourth rows before modification:
First Row Weight: 55
Fourth Row Weight: 70

Weight of first and fourth rows after modification:
First Row Weight: 58
Fourth Row Weight: 73
```

Nikhil Kumar
BCA 4D
09091102022

Q13

```
import pandas as pd

import numpy as np


np.random.seed(0)  # For reproducibility

num_orders = 1000

order_numbers = np.arange(1, num_orders + 1)

order_amounts = np.random.randint(10, 500, size=num_orders)

payment_methods = np.random.choice(['Credit Card', 'Debit Card', 'Net Banking', 'Cash on Delivery'], size=num_orders)


ecommerce_df = pd.DataFrame({'Order Number': order_numbers, 'Order Amount': order_amounts, 'Payment Method': payment_methods})


print("Sample E-Commerce DataFrame:")

print(ecommerce_df.head())


mean_order_amount = ecommerce_df['Order Amount'].mean()

mode_order_amount = ecommerce_df['Order Amount'].mode()[0]  # Mode can have multiple values, so we choose the first one

median_order_amount = ecommerce_df['Order Amount'].median()


print("\nMean Order Amount:", mean_order_amount)

print("Mode Order Amount:", mode_order_amount)

print("Median Order Amount:", median_order_amount)
```

Nikhil Kumar
BCA 4D
09091102022

```
Sample E-Commerce DataFrame:
   Order Number  Order Amount Payment Method
0             1           182     Debit Card
1             2            57    Net Banking
2             3           127    Net Banking
3             4           202     Debit Card
4             5           333    Net Banking

Mean Order Amount: 253.067
Mode Order Amount: 207
Median Order Amount: 249.5
```

Q14

```
import pandas as pd


csv_data = '''StudentID,Name,Age,Grade
1,Alice,20,A
2,Bob,21,B
3,Charlie,22,C
4,David,23,A
5,Emily,24,B
6,Frank,25,C
7,Grace,26,A
8,Henry,27,B
9,Ivy,28,C
10,Jack,29,A'''
with open('student_data.csv', 'w') as f:
    f.write(csv_data)


df = pd.read_csv('student_data.csv')


print("Student Data DataFrame:")
print(df)
```

Nikhil Kumar
BCA 4D
09091102022

```
print("\nStatistics of Student Data:")
print(df.describe())
```

```
Student Data DataFrame:
   StudentID     Name  Age Grade
0          1    Alice   20     A
1          2      Bob   21     B
2          3  Charlie   22     C
3          4    David   23     A
4          5    Emily   24     B
5          6    Frank   25     C
6          7    Grace   26     A
7          8    Henry   27     B
8          9      Ivy   28     C
9         10     Jack   29     A

Statistics of Student Data:
       StudentID       Age
count   10.00000  10.00000
mean     5.50000  24.50000
std      3.02765   3.02765
min      1.00000  20.00000
25%      3.25000  22.25000
50%      5.50000  24.50000
75%      7.75000  26.75000
max     10.00000  29.00000
```

Q15

```
import pandas as pd

# Sample DataFrame QtrSales (replace this with your actual DataFrame)
data = {'Category': ['Electronics', 'Clothing', 'Electronics', 'Clothing', 'Grocery', 'Grocery'],
        'Item Name': ['Laptop', 'T-Shirt', 'Smartphone', 'Jeans', 'Fruits', 'Vegetables'],
        'Expenditure': [1500, 200, 1000, 300, 150, 200]}
QtrSales = pd.DataFrame(data)

# Group by Category and calculate average expenditure
```

Nikhil Kumar
BCA 4D
09091102022

```
avg_expenditure_per_category = QtrSales.groupby('Category')['Expenditure'].mean()


# Print average expenditure per category

print("Average Expenditure per Category:")

print(avg_expenditure_per_category)
```

```
Average Expenditure per Category:
Category
Clothing        250.0
Electronics    1250.0
Grocery         175.0
Name: Expenditure, dtype: float64
```

Q16

```
import pandas as pd


data = {'Category': ['Electronics', 'Electronics', 'Clothing', 'Clothing', 'Electronics', 'Clothing'],
        'Item Name': ['Laptop', 'Smartphone', 'T-Shirt', 'Jeans', 'Tablet', 'Dress'],
        'Price': [1500, 1000, 200, 300, 800, 400],
        'Quantity': [5, 3, 10, 7, 2, 5]}
df = pd.DataFrame(data)


pivot_df = df.pivot(index='Category', columns='Item Name', values='Quantity')


pivot_table_df = df.pivot_table(index='Category', columns='Item Name', values='Price',
aggfunc='mean')


print("DataFrame after pivot():")

print(pivot_df)
```

Nikhil Kumar
BCA 4D
09091102022

```
print("\nDataFrame after pivot_table():")
print(pivot_table_df)
```

```
DataFrame after pivot():
Item Name    Dress  Jeans  Laptop  Smartphone  T-Shirt  Tablet
Category
Clothing       5.0    7.0     NaN         NaN     10.0     NaN
Electronics    NaN    NaN     5.0         3.0      NaN     2.0

DataFrame after pivot_table():
Item Name    Dress  Jeans  Laptop  Smartphone  T-Shirt  Tablet
Category
Clothing     400.0  300.0     NaN         NaN    200.0     NaN
Electronics    NaN    NaN  1500.0      1000.0      NaN   800.0
```

Q17

```
import pandas as pd

# Sample DataFrame
data = {'Category': ['Electronics', 'Electronics', 'Clothing', 'Clothing', 'Electronics', 'Clothing'],
        'Item Name': ['Laptop', 'Smartphone', 'T-Shirt', 'Jeans', 'Tablet', 'Dress'],
        'Price': [1500, 1000, 200, 300, 800, 400],
        'Quantity': [5, 3, 10, 7, 2, 5]}
df = pd.DataFrame(data)

# Calculate mean absolute deviation (MAD) using apply() and lambda function
mad = df['Price'].apply(lambda x: abs(x - df['Price'].mean())).mean()

# Display the MAD
print("Mean Absolute Deviation (MAD):", mad)
```

```
Mean Absolute Deviation (MAD): 400.0
```

Nikhil Kumar
BCA 4D
09091102022

Q18

```
import pandas as pd
import numpy as np

data = {'EmployeeID': [101, 102, 103, 104, 105],
        'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Emily'],
        'Salary': [50000, 60000, 70000, 80000, 90000],
        'Experience': [3, 5, 8, 4, 6]}
df = pd.DataFrame(data)

quartiles = np.percentile(df['Salary'], [25, 50, 75])

variance = df['Experience'].var()

print("Employee Data DataFrame:")
print(df)

print("\nQuartiles of Salary:")
print("25th Percentile (Q1):", quartiles[0])
print("50th Percentile (Median):", quartiles[1])
print("75th Percentile (Q3):", quartiles[2])
print("\nVariance of Experience:", variance)
```

```
Employee Data DataFrame:
   EmployeeID     Name  Salary  Experience
0         101    Alice   50000           3
1         102      Bob   60000           5
2         103  Charlie   70000           8
3         104    David   80000           4
4         105    Emily   90000           6

Quartiles of Salary:
25th Percentile (Q1): 60000.0
50th Percentile (Median): 70000.0
75th Percentile (Q3): 80000.0

Variance of Experience: 3.7
```

Q19

```
import pandas as pd

import numpy as np


# Generate random data

np.random.seed(0)  # For reproducibility

data = np.random.normal(loc=0, scale=1, size=1000)  # Generate 1000 random numbers from a normal distribution


# Create a Pandas Series from the random data

random_series = pd.Series(data)


# Calculate skewness

skewness = random_series.skew()


# Display the skewness

print("Skewness of Random Data:", skewness)
```

```
========== RESTART: C:/Users/natur/OneDriv
Skewness of Random Data: 0.03390983920295854
```

Q20

```
import pandas as pd

import numpy as np


# Generate random data

np.random.seed(0)  # For reproducibility
```

Nikhil Kumar
BCA 4D
09091102022

```
data = np.random.normal(loc=0, scale=1, size=1000)  # Generate 1000 random numbers from a normal distribution


# Create a DataFrame from the random data

df = pd.DataFrame({'RandomData': data})


# Compute kurtosis

kurtosis = df['RandomData'].kurtosis()


# Display the kurtosis

print("Kurtosis of Random Data:", kurtosis)
```

```
=============== RESTART: C:/Users/natur/OneDriv
Kurtosis of Random Data: -0.04097691643266943
```

Q21

Nikhil Kumar
BCA 4D
09091102022

**21.2: Apply aggregate function to find count, min , max and sum .**

➢ ***SOURCE CODE:-***

```
import pandas as pd
df=pd.read_csv("/content/crime.csv")
#print(df)
aggregate_functions = {
   'Year': ['count'],
   'Murder': ['min', 'max', 'sum'],
   'Robbery': ['min', 'max', 'sum'],
   'Arson': ['min', 'max', 'sum']
}
aggregate_result = df.agg(aggregate_functions)
print("\nAggregate function results:")
print(aggregate_result)
```

➢ ***OUTPUT***

```
Aggregate function results:
        Year  Murder  Robbery   Arson
count  420.0     NaN      NaN     NaN
min      NaN     0.0      0.0     0.0
max      NaN   423.0     83.0   178.0
sum      NaN  7900.0    953.0  2717.0
```

## 21.2: Apply aggregate function to find count, min , max and sum .

➢ ***SOURCE CODE:-***

```
import pandas as pd
df=pd.read_csv("/content/crime.csv")
#print(df)
aggregate_functions = {
   'Year': ['count'],
   'Murder': ['min', 'max', 'sum'],
   'Robbery': ['min', 'max', 'sum'],
   'Arson': ['min', 'max', 'sum']
}
aggregate_result = df.agg(aggregate_functions)
print("\nAggregate function results:")
print(aggregate_result)
```

➢ ***OUTPUT***

```
Aggregate function results:
        Year  Murder  Robbery   Arson
count  420.0     NaN      NaN     NaN
min      NaN     0.0      0.0     0.0
max      NaN   423.0     83.0   178.0
sum      NaN  7900.0    953.0  2717.0
```

Nikhil Kumar
BCA 4D
09091102022

## 21.5: Create a pie chart for all states

➢ ***SOURCE CODE:-***

```
import pandas as pd
import matplotlib.pyplot as plt
df=pd.read_csv("/content/crime.csv")
#print(df)
state_counts = df['State'].value_counts()
plt.figure(figsize=(10, 8))
state_counts.plot(kind='pie', autopct='%1.1f%%')
plt.title('Crime distribution by state')
plt.ylabel('')
plt.show()
```

➢ ***OUTPUT***

## 21.6: Plot the no. of murders state-wise

➢ ***SOURCE CODE:-***

```
import pandas as pd
import matplotlib.pyplot as plt
df=pd.read_csv("/content/crime.csv")
#print(df)
murder_by_state = df.groupby('STATE/UT')['Murder'].sum()
murder_by_state.plot(kind='bar', figsize=(10, 6))
plt.title('Number of murders by state')
plt.xlabel('State')
plt.ylabel('Number of murders')
plt.show()
```

➢ ***OUTPUT***

Nikhil Kumar
BCA 4D
09091102022

## 21.7: Check the data type of all column

➢ ***SOURCE CODE:-***

```
import pandas as pd
import matplotlib.pyplot as plt
df=pd.read_csv("/content/crime.csv")
#print(df)
# Check data type of all columns
print("\nData types of all columns:")
print(df.dtypes)
```

➢ ***OUTPUT***

```
Data types of all columns:
STATE/UT                               object
Year                                  float64
Murder                                float64
Assault on women                      float64
Kidnapping and Abduction              float64
Dacoity                               float64
Robbery                               float64
Arson                                 float64
Hurt                                  float64
Prevention of atrocities (POA) Act    float64
Protection of Civil Rights (PCR) Act  float64
Other Crimes Against SCs              float64
dtype: object
```

## 21.8: Check the robbery column for NULL values.
## if exist replace with 0

## 21.9: Select the data set where robbery greater than 10 and year= 2001

Nikhil Kumar
BCA 4D
09091102022

➢ ***SOURCE CODE:-***

```
import pandas as pd
df=pd.read_csv("/content/crime.csv")
#print(df)
print(df.describe())
```

➢ ***OUTPUT***

```
              Year      Murder  Assault on women  Kidnapping and Abduction  \
count   420.000000  420.000000        420.000000                420.000000
mean   2006.500000   18.809524         37.897619                 11.138095
std       3.456169   52.543442         79.435608                 34.946327
min    2001.000000    0.000000          0.000000                  0.000000
25%    2003.750000    0.000000          0.000000                  0.000000
50%    2006.500000    1.000000          3.000000                  0.000000
75%    2009.250000   14.000000         34.250000                  8.000000
max    2012.000000  423.000000        412.000000                363.000000

          Dacoity     Robbery       Arson         Hurt  \
count  420.000000  420.000000  420.000000   420.000000
mean     0.940476    2.269048    6.469048   117.033333
std      2.703976    6.100314   15.771988   206.612233
min      0.000000    0.000000    0.000000     0.000000
25%      0.000000    0.000000    0.000000     0.000000
50%      0.000000    0.000000    0.000000     3.500000
75%      0.000000    1.000000    5.000000   168.250000
max     22.000000   83.000000  178.000000  1252.000000

       Prevention of atrocities (POA) Act  \
count                          420.000000
mean                           296.566667
std                            636.605911
min                              0.000000
25%                              0.000000
50%                              8.000000
75%                            260.250000
max                           4885.000000

       Protection of Civil Rights (PCR) Act  Other Crimes Against SCs
count                            420.000000                420.000000
mean                              10.166667                380.219048
std                               35.968971                864.165129
min                                0.000000                  0.000000
25%                                0.000000                  0.000000
50%                                0.000000                  6.000000
75%                                2.000000                288.000000
max                              459.000000               4771.000000
```

**21.11: Perform sorting on Arson and Robbery**

**21.12: Perform renaming on two columns**

Nikhil Kumar
BCA 4D
09091102022

Q22

➢ ***SOURCE CODE:-***

```
import pandas as pd
df = pd.read_csv("/content/investigating clinical data.csv")
print(df)
```

➢ ***OUTPUT:-***

```
          PatientId  AppointmentID Gender          ScheduledDay  \
0      2.987250e+13        5642903      F  2016-04-29T18:38:08Z
1      5.589978e+14        5642503      M  2016-04-29T16:08:27Z
2      4.262962e+12        5642549      F  2016-04-29T16:19:04Z
3      8.679512e+11        5642828      F  2016-04-29T17:29:31Z
4      8.841186e+12        5642494      F  2016-04-29T16:07:23Z
...             ...            ...    ...                   ...
110522 2.572134e+12        5651768      F  2016-05-03T09:15:35Z
110523 3.596266e+12        5650093      F  2016-05-03T07:27:33Z
110524 1.557663e+13        5630692      F  2016-04-27T16:03:52Z
110525 9.213493e+13        5630323      F  2016-04-27T15:09:23Z
110526 3.775115e+14        5629448      F  2016-04-27T13:30:56Z
```

```
            AppointmentDay  Age      Neighbourhood  Scholarship  \
0      2016-04-29T00:00:00Z   62    JARDIM DA PENHA            0
1      2016-04-29T00:00:00Z   56    JARDIM DA PENHA            0
2      2016-04-29T00:00:00Z   62      MATA DA PRAIA            0
3      2016-04-29T00:00:00Z    8  PONTAL DE CAMBURI            0
4      2016-04-29T00:00:00Z   56    JARDIM DA PENHA            0
...                     ...  ...                ...          ...
110522 2016-06-07T00:00:00Z   56        MARIA ORTIZ            0
110523 2016-06-07T00:00:00Z   51        MARIA ORTIZ            0
110524 2016-06-07T00:00:00Z   21        MARIA ORTIZ            0
110525 2016-06-07T00:00:00Z   38        MARIA ORTIZ            0
110526 2016-06-07T00:00:00Z   54        MARIA ORTIZ            0

        Hipertension  Diabetes  Alcoholism  Handcap  SMS_received No-show
0                  1         0           0        0             0      No
1                  0         0           0        0             0      No
2                  0         0           0        0             0      No
3                  0         0           0        0             0      No
4                  1         1           0        0             0      No
...              ...       ...         ...      ...           ...     ...
110522             0         0           0        0             1      No
110523             0         0           0        0             1      No
110524             0         0           0        0             1      No
110525             0         0           0        0             1      No
110526             0         0           0        0             1      No
```

**22.1: describe() the dataset**

Nikhil Kumar
BCA 4D
09091102022

```
import pandas as pd
df = pd.read_csv("/content/investigating clinical data.csv")
df.describe()
```

➢ ***OUTPUT:-***

| | PatientId | AppointmentID | Age | Scholarship | Hipertension | Diabetes | Alcoholism | Handcap | SMS_received |
|---|---|---|---|---|---|---|---|---|---|
| count | 1.105270e+05 | 1.105270e+05 | 110527.000000 | 110527.000000 | 110527.000000 | 110527.000000 | 110527.000000 | 110527.000000 | 110527.000000 |
| mean | 1.474963e+14 | 5.675305e+06 | 37.088874 | 0.098266 | 0.197246 | 0.071865 | 0.030400 | 0.022248 | 0.321026 |
| std | 2.560949e+14 | 7.129575e+04 | 23.110205 | 0.297675 | 0.397921 | 0.258265 | 0.171686 | 0.161543 | 0.466873 |
| min | 3.921784e+04 | 5.030230e+06 | -1.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| 25% | 4.172614e+12 | 5.640286e+06 | 18.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| 50% | 3.173184e+13 | 5.680573e+06 | 37.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| 75% | 9.439172e+13 | 5.725524e+06 | 55.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 1.000000 |
| max | 9.999816e+14 | 5.790484e+06 | 115.000000 | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 4.000000 | 1.000000 |

## 22.2: Perform renaming on two columns

➢ ***SOURCE CODE:-***

```
import pandas as pd
df = pd.read_csv("/content/investigating clinical data.csv")
df.rename(columns={"PatientId": "PatientID_NUM", "Neighbourhood": "Address"},
inplace=True)
print(df.columns)
```

➢ ***OUTPUT:-***

```
Index(['PatientId', 'AppointmentID', 'Gender', 'ScheduledDay',
       'AppointmentDay', 'Age', 'Neighbourhood', 'Scholarship', 'Hipertension',
       'Diabetes', 'Alcoholism', 'Handcap', 'SMS_received', 'No-show'],
      dtype='object')
```

```
Index(['PatientID_NUM', 'AppointmentID', 'Gender', 'ScheduledDay',
       'AppointmentDay', 'Age', 'Address', 'Scholarship', 'Hipertension',
       'Diabetes', 'Alcoholism', 'Handcap', 'SMS_received', 'No-show'],
      dtype='object')
```

Nikhil Kumar
BCA 4D
09091102022

## ➢ ***SOURCE CODE:-***

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

df = pd.read_csv("/content/investigating clinical data.csv")

df['AppointmentDay'] = pd.to_datetime(df['AppointmentDay'])
df['ScheduledDay'] = pd.to_datetime(df['ScheduledDay'])

df['AppointmentDelayDays'] = (df['AppointmentDay'] - df['ScheduledDay']).dt.days

age_delay = df.groupby('Age')['AppointmentDelayDays'].mean()

plt.figure(figsize=(10, 6))
plt.plot(age_delay.index, age_delay.values, marker='o', linestyle='-', color='b')
plt.xlabel('Age')
plt.ylabel('Average Appointment Delay (Days)')
plt.title('Age vs. Appointment Delay')
plt.grid(True)
plt.show()
```

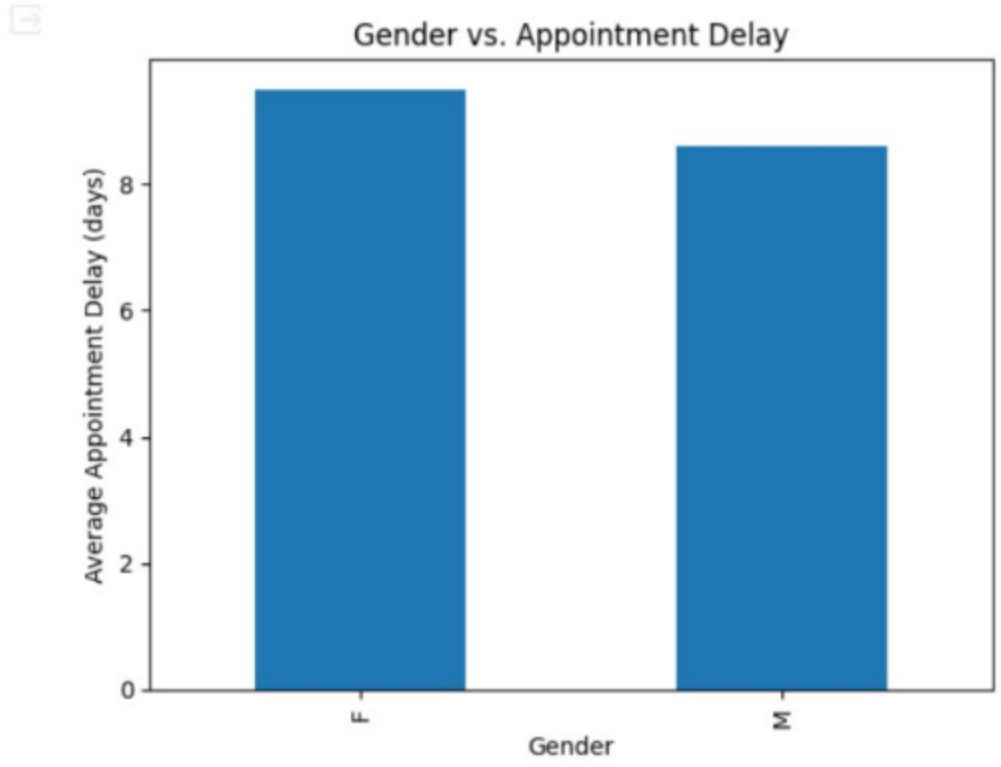
## ➢ ***OUTPUT:-***


Nikhil Kumar
BCA 4D
09091102022

➢ ***SOURCE CODE:-***

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

df = pd.read_csv("/content/investigating clinical data.csv")

df['AppointmentDay'] = pd.to_datetime(df['AppointmentDay'])
df['ScheduledDay'] = pd.to_datetime(df['ScheduledDay'])

df['AppointmentDelayDays'] = (df['AppointmentDay'] - df['ScheduledDay']).dt.days


df.groupby("Gender")["AppointmentDelayDays"].mean().plot(kind="bar")
plt.xlabel("Gender")
plt.ylabel("Average Appointment Delay (days)")
plt.title("Gender vs. Appointment Delay")
plt.show()
```

➢ ***OUTPUT:-***



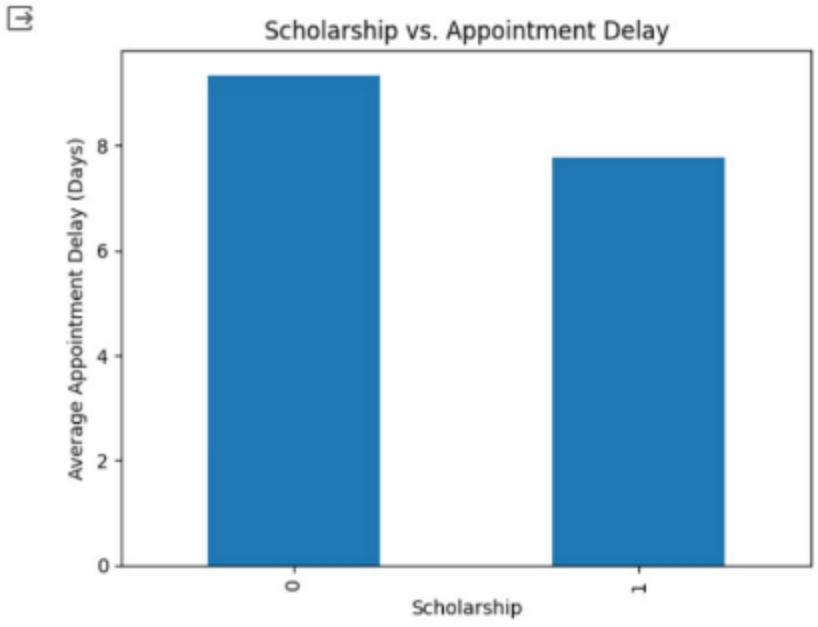

Nikhil Kumar
BCA 4D
09091102022

**22.5: Do people delay their appointments when they have no scholarship? (Show by plotting)**

➢ ***SOURCE CODE:-***

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

df = pd.read_csv("/content/investigating clinical data.csv")

df['AppointmentDay'] = pd.to_datetime(df['AppointmentDay'])
df['ScheduledDay'] = pd.to_datetime(df['ScheduledDay'])

df['AppointmentDelayDays'] = (df['AppointmentDay'] - df['ScheduledDay']).dt.days

df.groupby("Scholarship")["AppointmentDelayDays"].mean().plot(kind="bar")
plt.xlabel("Scholarship")
plt.ylabel("Average Appointment Delay (Days)")
plt.title("Scholarship vs. Appointment Delay")
plt.show()
```

➢ ***OUTPUT:-***



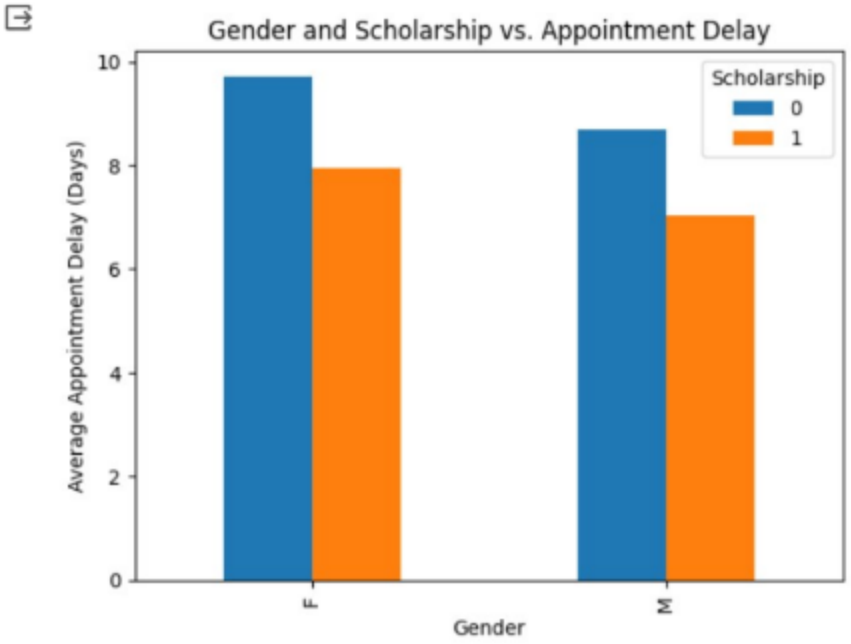

Nikhil Kumar
BCA 4D
09091102022

**22.6: Is there any relationship between gender having scholarships and delaying the appointment? (show by plotting)**

➢ ***SOURCE CODE:-***

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

df = pd.read_csv("/content/investigating clinical data.csv")

df['AppointmentDay'] = pd.to_datetime(df['AppointmentDay'])
df['ScheduledDay'] = pd.to_datetime(df['ScheduledDay'])

df['AppointmentDelayDays'] = (df['AppointmentDay'] - df['ScheduledDay']).dt.days
df.groupby(["Gender","Scholarship"])
["AppointmentDelayDays"].mean().unstack().plot(kind="bar")
plt.xlabel("Gender")
plt.ylabel("Average Appointment Delay (Days)")
plt.title("Gender and Scholarship vs. Appointment Delay")
plt.show()
```

➢ ***OUTPUT:-***



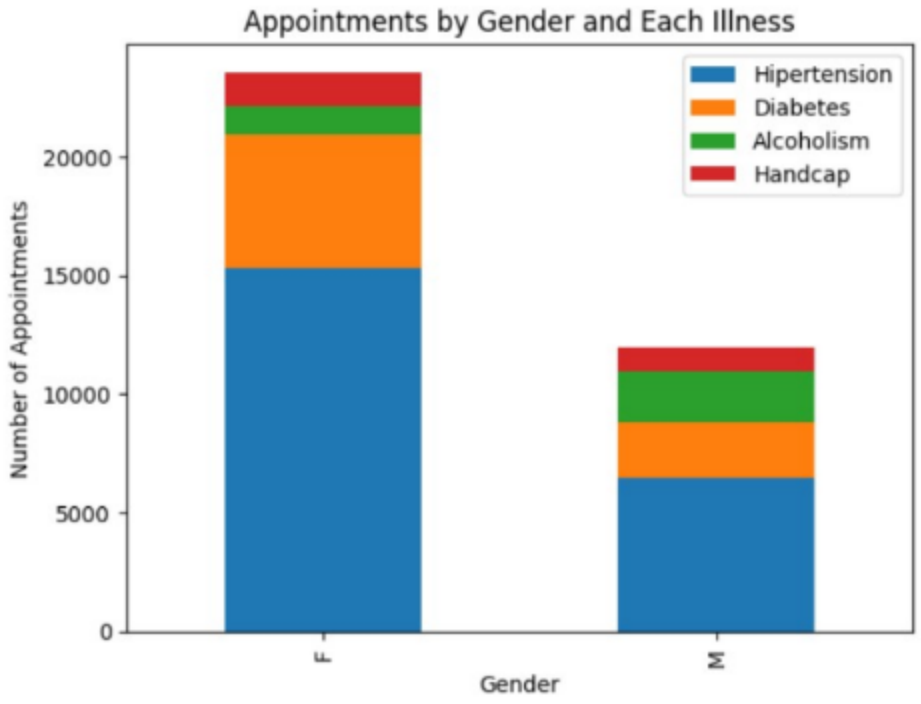

Nikhil Kumar
BCA 4D
09091102022

**22.7: Which gender has more appointments in which illness? (Show by plotting)**

➢ ***SOURCE CODE:-***

```
import pandas as pd
import matplotlib.pyplot as plt
 df = pd.read_csv("/content/investigating clinical data.csv")

grouped_df = df.groupby("Gender")[illness_columns].sum()

grouped_df.plot(kind="bar", stacked=True)
plt.xlabel("Gender")
plt.ylabel("Number of Appointments")
plt.title("Appointments by Gender and Each Illness")
plt.show()
```

➢ ***OUTPUT:-***



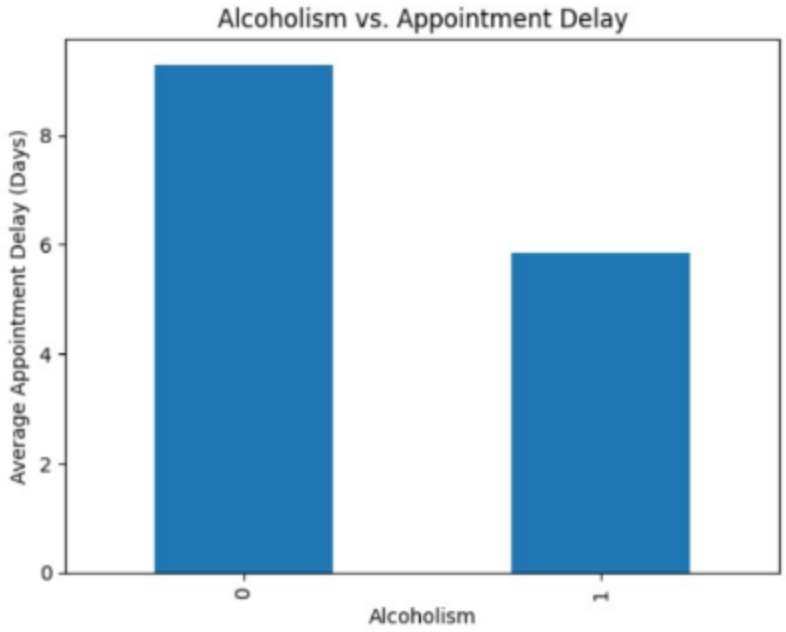

Nikhil Kumar
BCA 4D
09091102022

**22.8: Does a person suffering from alcoholism tend to delay appointments? (Show by plotting)**

- ***SOURCE CODE:-***

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

df = pd.read_csv("/content/investigating clinical data.csv")

df['AppointmentDay'] = pd.to_datetime(df['AppointmentDay'])
df['ScheduledDay'] = pd.to_datetime(df['ScheduledDay'])

df['AppointmentDelayDays'] = (df['AppointmentDay'] - df['ScheduledDay']).dt.days
df.groupby("Alcoholism")["AppointmentDelayDays"].mean().plot(kind="bar")
plt.xlabel("Alcoholism")
plt.ylabel("Average Appointment Delay (Days)")
plt.title("Alcoholism vs. Appointment Delay")
plt.show()
```

- ***OUTPUT:-***




Nikhil Kumar
BCA 4D
09091102022

## 22.9: Perform group by and count on neighborhood

➢ ***SOURCE CODE:-***

```
import pandas as pd
df = pd.read_csv("/content/investigating clinical data.csv")
neighborhood_counts = df.groupby("Neighbourhood").size()
print(neighborhood_counts)
```

➢ ***OUTPUT:-***

```
Neighbourhood
AEROPORTO                8
ANDORINHAS            2262
ANTÔNIO HONÓRIO        271
ARIOVALDO FAVALESSA    282
BARRO VERMELHO         423
                       ...
SÃO JOSÉ              1977
SÃO PEDRO             2448
TABUAZEIRO            3132
UNIVERSITÁRIO          152
VILA RUBIM             851
Length: 81, dtype: int64
```

## 22.10: Perform agg() function on dataset.

➢ ***SOURCE CODE:-***

```
import pandas as pd
df = pd.read_csv("/content/investigating clinical data.csv")
df.agg({"PatientId": "median" , "Age": "sum", "AppointmentID": "mean" , "Scholarship":
"min" , "Hipertension": "max"})
```

➢ ***OUTPUT:-***

```
PatientId        3.173184e+13
Age              4.099322e+06
AppointmentID    5.675305e+06
Scholarship      0.000000e+00
Hipertension     1.000000e+00
dtype: float64
```

Nikhil Kumar
BCA 4D
09091102022

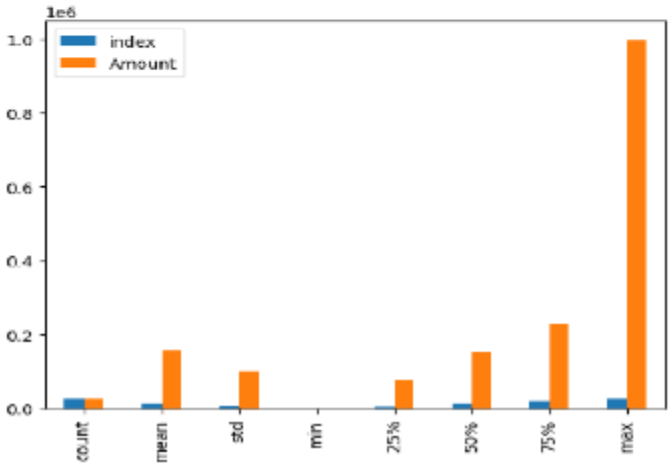
Q23

➢ ***SOURCE CODE:-***

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
data=pd.read_csv("/content/credit.csv")
#print(data)
print(data.head())
data.describe().plot(kind='bar')
plt.show()
```

➢ ***OUTPUT***

```
   index                 City       Date  Card Type Exp Type Gender  Amount
0      0          Delhi, India  29-Oct-14        Gold    Bills      F   82475
1      1  Greater Mumbai, India  22-Aug-14    Platinum    Bills      F   32555
2      2       Bengaluru, India  27-Aug-14      Silver    Bills      F  101738
3      3  Greater Mumbai, India  12-Apr-14   Signature    Bills      F  123424
4      4       Bengaluru, India  05-May-15        Gold    Bills      F  171574
```




Nikhil Kumar
BCA 4D
09091102022

**23.2: Display pivot table to create a summary for the total amount spent a) month b) city**

**(b) City**

➢ ***SOURCE CODE:-***

```
import pandas as pd
import numpy as np
data=pd.read_csv("/content/credit.csv")
#print(data)
pivot_city = data.pivot_table(index='City', values='Amount', aggfunc='sum')
print("\nTotal amount spent by city:")
print(pivot_city)
```

➢ ***OUTPUT***

```
Total amount spent by city:
                  Amount
City
Achalpur, India   1606641
Adilabad, India   1769464
Adityapur, India   963993
Adoni, India      1575355
Adoor, India       647725
...                   ...
Zaidpur, India     723818
Zamania, India     865634
Zira, India       1640547
Zirakpur, India    549663
Zunheboto, India   466429

[986 rows x 1 columns]
```

Nikhil Kumar
BCA 4D
09091102022

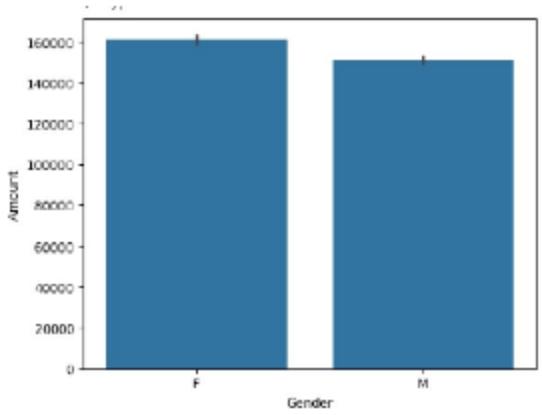
## 23.3: Analyze the impact of gender to study consumer behavior

### ➢ *SOURCE CODE:-*

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
data=pd.read_csv("/content/credit.csv")
#print(data)
gender_analysis = data.groupby('Gender')['Amount'].sum()

print("\nTotal amount spent by gender:")
print(gender_analysis)
sns.barplot(x='Gender', y='Amount', data=data)
plt.show()
```

### ➢ *OUTPUT*

```
Total amount spent by gender:
Gender
F    2205311030
M    1869522343
Name: Amount, dtype: int64
```




Nikhil Kumar
BCA 4D
09091102022

**23.4: Check for NULL values in columns.If they exist replace them with appropriate values.**

➢ ***SOURCE CODE:-***

```
import pandas as pd
import numpy as np
data=pd.read_csv("/content/credit.csv")
#print(data)
print("\nNULL values in columns:")
print(data.isnull().sum())
data['Amount'].fillna(0, inplace=True)
print("\nAfter replacing NULL values:")
print(data.isnull().sum())
```

➢ ***OUTPUT***

```
NULL values in columns:
index         0
City          0
Date          0
Card Type     0
Exp Type      0
Gender        0
Amount        0
dtype: int64

After replacing NULL values:
index         0
City          0
Date          0
Card Type     0
Exp Type      0
Gender        0
Amount        0
dtype: int64
```

Nikhil Kumar
BCA 4D
09091102022

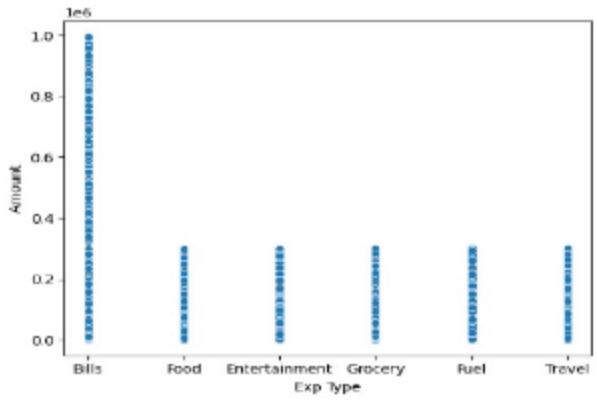
**23.5: Analyze the relationship type between expense type and amount.**

➢ ***SOURCE CODE:-***

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
data=pd.read_csv("/content/credit.csv")
#print(data)
expense_groups = data.groupby('Exp Type')
print(expense_groups['Amount'].describe())
sns.scatterplot(x='Exp Type', y='Amount', data=data)
plt.show()
```

➢ ***OUTPUT***

```
               count           mean            std     min       25%
Exp Type
Bills         5078.0  178627.899370  151893.760199  1026.0  78831.75
Entertainment 4762.0  152548.831583   86628.587025  1061.0  77749.25
Food          5463.0  150965.405272   86492.589170  1018.0  75862.50
Fuel          5257.0  150111.436371   85929.342308  1038.0  77405.00
Grocery       4754.0  151074.447413   86440.356165  1005.0  75648.25
Travel         738.0  148042.833333   86627.487531  1070.0  73909.00

                50%       75%       max
Exp Type
Bills        160551.0  237895.50  998077.0
Entertainment 153994.5 228414.75 299936.0
Food         151679.0  225387.50  299837.0
Fuel         149620.0  224480.00  299985.0
Grocery      152157.5  225550.50  299920.0
Travel       146947.0  221778.75  299618.0
```




Nikhil Kumar
BCA 4D
09091102022

## 23.6: Analyze the spending habits by city and gender.

➢ ***SOURCE CODE:-***

```
import pandas as pd
import numpy as np
data=pd.read_csv("/content/credit.csv")
#print(data)
city_gender_groups = data.groupby(['City', 'Gender'])
print(city_gender_groups['Amount'].describe())
```

➢ ***OUTPUT***

```
                          count           mean             std       min  \
City             Gender
Achalpur, India  F          5.0  172195.200000   66849.453788   55347.0
                 M          4.0  186416.250000   43066.249789  138246.0
Adilabad, India  F          5.0  232886.000000   65219.130767  122336.0
                 M          5.0  121006.800000   61118.426986   49020.0
Adityapur, India F          2.0  253386.500000   61599.607243  209829.0
...                         ...            ...            ...       ...
Zira, India      M          7.0  178522.714286   81757.430940   48909.0
Zirakpur, India  F          1.0  289172.000000            NaN  289172.0
                 M          2.0  130245.500000   64092.865754   84925.0
Zunheboto, India F          1.0   78591.000000            NaN   78591.0
                 M          2.0  193919.000000  147994.620875   89271.0

                           25%       50%       75%       max
City             Gender
Achalpur, India  F      178612.00  201032.0  209805.00  216180.0
                 M      164563.50  182957.5  204810.25  241504.0
Adilabad, India  F      227325.00  258225.0  278085.00  278459.0
                 M       80193.00  108313.0  178349.00  189159.0
Adityapur, India F      231607.75  253386.5  275165.25  296944.0
...                           ...       ...       ...       ...
Zira, India      M      138824.00  190358.0  219226.50  294291.0
Zirakpur, India  F      289172.00  289172.0  289172.00  289172.0
                 M      107585.25  130245.5  152905.75  175566.0
Zunheboto, India F       78591.00   78591.0   78591.00   78591.0
                 M      141595.00  193919.0  246243.00  298567.0

[1876 rows x 8 columns]
```

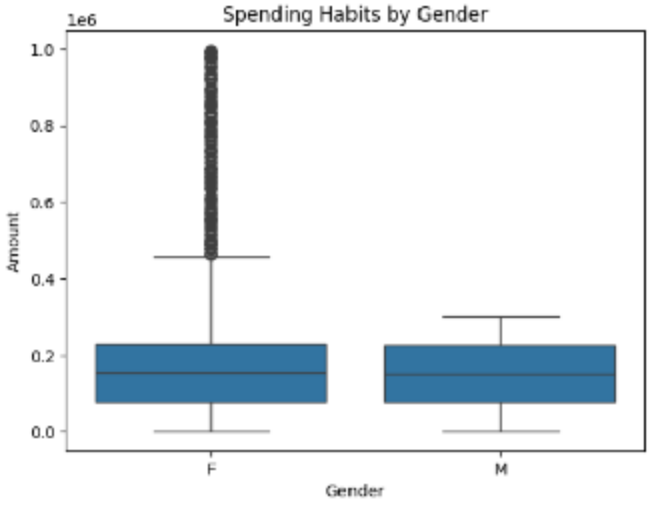
Nikhil Kumar
BCA 4D
09091102022

## ➢ *SOURCE CODE:-*

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from scipy.stats import skew, kurt import pandas as pd
import matplotlib.pyplot as plt
df=pd.read_csv("/content/crime.csv")
#print(df)
state_counts = df['STATE/UT'].value_counts()
plt.figure(figsize=(10, 8))
state_counts.plot(kind='pie', autopct='%1.1f%%')
plt.title('Crime distribution by state')
plt.ylabel('')
plt.show()osis
data=pd.read_csv("/content/credit.csv")
#print(data)
data_skewness = skew(data['Amount'])
data_kurtosis = kurtosis(data['Amount'])
print(f'Skewness: {data_skewness}')
print(f'Kurtosis: {data_kurtosis}')
sns.boxplot(x='Gender', y='Amount', data=data)
plt.title('Spending Habits by Gender')
plt.show()
```

## ➢ *OUTPUT*

```
Skewness: 1.7550705661419843
Kurtosis: 10.312972126257737
```




Nikhil Kumar
BCA 4D
09091102022

Nikhil Kumar
BCA 4D
09091102022