MAHARAJA SURAJMAL INSTITUTE

C-4, Janakpuri, New-Delhi, 110058

Department of Computer Applications

DATA SCIENCE PRACTICAL FILE

Course Code: **BCAP 212**

Course Name: **Introduction to Data Science**

Submitted By:

Name: AKANKSHA DANGRI
Enrolment Number: 08914902022
Semester and Section: IV-B

Submitted To:

Ms. Tarunim Sharma
Asst. Professor

(Affiliated to GGSIP University)

2022-25

INDEX

<table>
<tr><th>S.No.</th><th>Question</th></tr>
<tr><td>1</td><td>Write all the ways to create a dataframe</td></tr>
<tr><td>2</td><td>Given a dataset, print the following:
1. Records of index 1 & 3
2. Records where age >= 15
3. Records where age >= 12 and gender = Male
4. City and gender of people with age >= 12</td></tr>
<tr><td>3</td><td>Create a dataframe to store data of 10 students, with the columns being "Name", "Age", "Semester I marks out of 600", "Semester II marks out of 500", and "Attendance"
1. Display details of students who scored more than 560 marks in sem I
2. Display details of students who scored less than 250 marks in sem II
3. Display details of student who scored minimum marks in sem II
4. Display details of student who scored maximum marks in sem II
5. Display details of students whose attendance is more than 75
6. Display details of students whose attendance is less than 50
7. Insert 2 new records
8. Add a column corresponding to percentage of marks of both semesters
9. Add a new column corresponding to grades:
<table>
<tr><th>Percentage</th><th>Grade</th></tr>
<tr><td>>=90</td><td>O</td></tr>
<tr><td>>=75 and <90</td><td>A+</td></tr>
<tr><td>>=60 and <75</td><td>A</td></tr>
<tr><td>>=50 and <60</td><td>B+</td></tr>
<tr><td>>=40 and <50</td><td>B</td></tr>
<tr><td>>40</td><td>F</td></tr>
</table>
</td></tr>
<tr><td>4</td><td>Create a DataFrame based on E-Commerce data and generate mean, mode, and median</td></tr>
<tr><td>5</td><td>Write a program to implement pivot() and pivot-table() on a DataFrame</td></tr>
<tr><td>6</td><td>Write a Program to read a CSV file and create its dataframe</td></tr>
<tr><td>7</td><td>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</td></tr>
<tr><td>8</td><td>Create a dataframe having age, name, weight of five students. Write a program to</td></tr>
</table>

<table>
<tr><td></td><td>display the details of first and fourth rows</td></tr>
<tr><td>9</td><td>Write a program to create a dataframe to store weight, age and name of three people. Print the DataFrame and its transpose</td></tr>
<tr><td>10</td><td>Create a pandas series from a dictionary of values and an ndarray</td></tr>
<tr><td>11</td><td>Perform sorting on series data and dataframes</td></tr>
<tr><td>12</td><td>Two series objects, Population stores the details of four metro cities of India and another object, and AvgIncome stores the total average income reported in four years in these cities. Calculate income per capita for each of these metro cities</td></tr>
<tr><td>13</td><td>Series objects Temp1, Temp2, Temp3, and Temp4 store the temperature of days of week 1, week 2, week 3, week 4. Write a script to:
1. Print average temperature per week
2. Print average temperature of entire month</td></tr>
<tr><td>14</td><td>Write a pandas program to convert a series of lists to one series</td></tr>
<tr><td>15</td><td>Write a pandas program to compare elements of two series</td></tr>
<tr><td>16</td><td>Write a pandas program to create a subset of a given series based on values and condition</td></tr>
<tr><td>17</td><td>Given the dataset Toyota.csv, do the following operations:
1. Upload Toyota.csv in the dataframe df.
2. What is the data type of MetColor?
3. How many null values are there in the KM field?
4. Which column has 7 unique values?
5. How many records are there?
6. Replace three, four, five value in Doors column to 3,4,5 respectively
7. Change the datatype of Doors to int64
8. Impute the value of Price with median
9. Replace “????” in the HP field with mean
10. Impute blank values in FuelType with mode.
11. Delete the rows with MetColor and Age as blank
12. Replace “??” value in KM with mean
13. What is the mean, median and mode of the KM field?
14. Create a new column “Category” based on the value of the column “Age” according to the following table:
<table><tr><th>Value</th><th>Category</th></tr><tr><td>0-10</td><td>Old</td></tr><tr><td>11-20</td><td>Medium</td></tr><tr><td>20+</td><td>New</td></tr></table>15. Create dummy fields for the FuelType column</td></tr>
</table>

<table>
<tr><td>18</td><td>Write a pandas program to change the order of index of a given series</td></tr>
<tr><td>19</td><td>Write a pandas program to get the items of a given series not present in another given series</td></tr>
<tr><td>20</td><td>Write a pandas program to calculate the frequency counts of each unique value of a given series</td></tr>
<tr><td>21</td><td>Create a series and print all the elements that are above 75th percentile</td></tr>
<tr><td>22</td><td>Write a program to find the MAD (mean absolute deviation) of all columns in a dataframe</td></tr>
<tr><td>23</td><td>Create a dataframe based on employee data and generate quartile and variance</td></tr>
<tr><td>24</td><td>Write a program to implement skewness on random data</td></tr>
<tr><td>25</td><td>Create a dataframe of any data and compute the kurtosis</td></tr>
<tr><td>26</td><td>Give the code or syntax to perform the following operation on two 2D numpy array array1 and array2 and 1D array array3:
1) Add array1 and array2
2) Find sum of array1 elements over a given axis
3) Find product of array2 elements over a given axis
4) Change the dimension of array3 to 2D
5) Transpose the array created in part 4
6) Display 2 rows and the third column of the 2D array
7) Join two 2D arrays along row
8) Convert array2 to a 1D array
9) Split array1 into multiple subarrays</td></tr>
<tr><td>27</td><td>Write python code to create the following series:
<table><tr><td>101</td><td>Harsh</td></tr><tr><td>102</td><td>Arun</td></tr><tr><td>103</td><td>Ankur</td></tr><tr><td>104</td><td>Harpal</td></tr><tr><td>105</td><td>Divya</td></tr><tr><td>106</td><td>Jeet</td></tr></table>
1) Show the details of the first 3 employees using the head method
2) Show the details of the last 3 employees using the tail method
3) Show the details of the first 3 employees without using the head method
4) Show the details of the last 3 employees without using the tail method
5) Show the value of index number 102
6) Show 2nd to 4th records (inclusive)
7) Show the values at indexes 101, 103, and 105</td></tr>
</table>

8) Show details of “Arun”

28 Create a dataframe for the below given data:

SNO	Batsman	Test	ODI	T20
1	Virat Kohli	3543	2245	1925
2	Ajinkya Rehane	2578	2165	1853
3	Rohit Sharma	2280	2080	1522
4	Shikhar Dhawan	2158	1957	1020
5	Hardik Pandya	1879	1856	980

1) Print the batsman name along with runs scored in Test and T20 using column names and dot notation
2) Display the batsman name along with runs scored in the ODI using the loc method
3) Display the batsman details who scored runs:
   - More than 2000 in ODI
   - Less than 2500 in Test
   - More than 1500 in T20
4) Display the columns using the column index number like 0, 2, and 4
5) Display the alternate rows
6) Reindex the dataframe created above with batsman name and delete the data of Hardik Pandya and Shikhar Dhawan by their index from the original dataframe
7) Insert two rows in the dataframe and delete rows whose index is 1 and 4
8) Delete the column Test, and add one more column “total”, ie the total of ODI and T20 runs
9) Rename the column “T20” to “T20I Runs”
10) Print the dataframe without headers

29 Create the following dataframe “Sales” containing year-wise sales figures for five salespersons in INR. Use the years as column labels and the salesperson names as indexes

	2014	2015	2016	2017
Madhu	100.5	12000	2000	50000
Kusum	150.8	18000	5000	60000
Kinshuk	200.9	22000	70000	70000
Ankit	30000	30000	1000	80000
Shruti	40000	45000	1250	90000

1. Display the indexes
2. Display the names of the columns

<table>
<tr><td></td><td>
3. Display the dimensions, shape, size, and values
4. Display the last two rows
5. Display the first two columns
6. Change the dataframe Sales such that it becomes its transpose
7. Add data to Sales for the salesman “Sumeet” where the sales made are [196.2, 37800, 52000, 78438] in the years [2014, 2015, 2016, 2017] respectively
8. Delete the data for the the year 2014
9. Update the sale made by Shruti in 2017 to 100000
10. Export the dataframe Sales to a comma separated file “SalesFigures.csv” on the disk. Do not export the indexes or column names
11. Change the name of the salesperson “Ankit” to “Vivaan” and “Kinshuk” to “Shailesh”
12. Delete the data for the salesman “Madhu”
</td></tr>
<tr><td>30</td><td>
Write code to create the following dataframe “Patient” and write code to perform the following operations:
<table>
<tr><th>PatientID</th><th>Treatment_Starts</th><th>Drug</th><th>Dosage</th></tr>
<tr><td>PT1</td><td>1/14/16</td><td>CISPLATIN</td><td>200</td></tr>
<tr><td>PT20</td><td>1/2/16</td><td>NIVOLUNAB</td><td>140</td></tr>
<tr><td>PT2</td><td>1/10/16</td><td>CISPLATIN</td><td>180</td></tr>
<tr><td>PT5</td><td>1/24/16</td><td>CISPLATIN</td><td>140</td></tr>
<tr><td>PT8</td><td>2/14/16</td><td>CISPLATIN</td><td>190</td></tr>
</table>
1. Check for the number of rows in the dataframe
2. Show the datatype of each column
3. Display the first and third column
4. List the number of unique drugs
5. Show the record of the patient with the ID of “PT5” and the “CISPLATIN” drug
6. Display all the rows where the dosage is greater than 180
7. Sort the original dataframe in ascending order of “PatientID” and in descending order of “Treatment_Starts”
8. Show all the drugs and how many patients received those drugs
9. Create a bar chart in seaborn to compare the counts for the two drugs
10. Display the average dosage of each drug
</td></tr>
<tr><td>31</td><td>Given a csv file named milknew.csv, write a program to implement a linear regression model to investigate the relationship between the dependent and independent variables present in the dataset. Evaluate the model's performance using two key metrics: the coefficient of determination (R^2) and the Mean Squared Error (MSE)</td></tr>
</table>

Ques 1: Write all the ways to create a dataframe.

[2]:
```
import pandas as pd

# From a list of lists
data = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
df = pd.DataFrame(data)
print('From a list of lists:')
df
```

```
From a list of lists:
```

[2]:

	0	1	2
0	1	2	3
1	4	5	6
2	7	8	9

[3]:
```
import pandas as pd

# From a dictionary
data = {'a': [1, 4, 7], 'b': [2, 5, 8], 'c': [3, 6, 9]}
df = pd.DataFrame(data)
print('\nFrom a dictionary:')
df
```

```
From a dictionary:
```

[3]:

	a	b	c
0	1	2	3
1	4	5	6
2	7	8	9

[4]:
```
import pandas as pd

# From a list of dictionaries
data = [{'a': 1, 'b': 2, 'c': 3}, {'a': 4, 'b': 5, 'c': 6}, {'a': 7, 'b': 8, 'c': 9}]
df = pd.DataFrame(data)
print('\nFrom a list of dictionaries:')
df
```

```
From a list of dictionaries:
```

[4]:

	a	b	c
0	1	2	3
1	4	5	6
2	7	8	9

Ques 2: Given a dataset, print the following:

1. Records of index 1 & 3
2. Records where age >= 15
3. Records where age >= 12 and gender = Male
4. City and gender of people with age >= 12.

[43]:
```
import pandas as pd

d = {
    'Name':['Akanksha','Mallika','Drishti','Satyam'],
    'Age':[11,12,15,19],
    'Gender':['F','F','F','M'],
    'City':['Delhi','Mumbai','Chennai','Kolkata']
}
df = pd.DataFrame(d)
df
```

[43]:

	Name	Age	Gender	City
0	Akanksha	11	F	Delhi
1	Mallika	12	F	Mumbai
2	Drishti	15	F	Chennai
3	Satyam	19	M	Kolkata

[44]:
```
df.iloc[[1,3],:]
```

[44]:

	Name	Age	Gender	City
1	Mallika	12	F	Mumbai
3	Satyam	19	M	Kolkata

[45]:
```
f = df[df['Age']>=15]
f
```

[45]:

	Name	Age	Gender	City
2	Drishti	15	F	Chennai
3	Satyam	19	M	Kolkata

[55]:
```
s = df.query('Age>=12 and Gender=="M"')
s
```

[55]:

	Name	Age	Gender	City
3	Satyam	19	M	Kolkata

[56]:
```
t = df.query('Age >= 12')[['City','Gender']]
t
```

[56]:

	City	Gender
1	Mumbai	F
2	Chennai	F
3	Kolkata	M

Ques 3: Create a dataframe to store data of 10 students, with the columns being "Name", "Age", "Semester I marks out of 600", "Semester II marks out of 500", and "Attendance"

1. Display details of students who scored more than 560 marks in sem I
2. Display details of students who scored less than 250 marks in sem II
3. Display details of student who scored minimum marks in sem II
4. Display details of student who scored maximum marks in sem II
5. Display details of students whose attendance is more than 75
6. Display details of students whose attendance is less than 50
7. Insert 2 new records
8. Add a column corresponding to percentage of marks of both semesters
9. Add a new column corresponding to grades.

[13]:
```
q8={'Name':['Aditya','Hardik','Jiya','Manas','Yash'],'Age':[23,24,25,22,26],'Sem I marks out of 600':[545,567,589,500,578],
    'Sem II marks out of 500':[456,478,468,488,497],'Attendance':[56,45,36,78,86]}
df=pd.DataFrame(q8)
df
```

[13]:

	Name	Age	Sem I marks out of 600	Sem II marks out of 500	Attendance
0	Aditya	23	545	456	56
1	Hardik	24	567	478	45
2	Jiya	25	589	468	36
3	Manas	22	500	488	78
4	Yash	26	578	497	86

[24]:
```
df[df['Sem I marks out of 600']>560]
```

[24]:

	Name	Age	Sem I marks out of 600	Sem II marks out of 500	Attendance
1	Hardik	24	567	478	45
2	Jiya	25	589	468	36
4	Yash	26	578	497	86

[25]:
```
df[df['Sem II marks out of 500']<250]
```

[25]:

	Name	Age	Sem I marks out of 600	Sem II marks out of 500	Attendance

[29]:
```
df[df['Sem II marks out of 500'] == df['Sem II marks out of 500'].min()]
```

[29]:

	Name	Age	Sem I marks out of 600	Sem II marks out of 500	Attendance
0	Aditya	23	545	456	56

[30]:
```
df[df['Sem II marks out of 500'] == df['Sem II marks out of 500'].max()]
```

[30]:

	Name	Age	Sem I marks out of 600	Sem II marks out of 500	Attendance
4	Yash	26	578	497	86

[31]:
```
df[df['Attendance']>75]
```

[31]:

	Name	Age	Sem I marks out of 600	Sem II marks out of 500	Attendance
3	Manas	22	500	488	78
4	Yash	26	578	497	86

[33]:
```
df[df['Attendance']<50]
```

[33]:

	Name	Age	Sem I marks out of 600	Sem II marks out of 500	Attendance
1	Hardik	24	567	478	45
2	Jiya	25	589	468	36

[13]:
```
new_data = {
    'Name':['Shashwat', 'Dhruv'],
    'Age':[22,  23],
    'Sem I marks out of 600':  [300, 400],
    'Sem II marks out of 500': [400, 300],
    'Attendance':[80,  40]
}
new_df = pd.DataFrame(new_data)
df     = pd.concat([df,new_df], ignore_index=True)
df
```

[13]:

	Name	Age	Sem I marks out of 600	Sem II marks out of 500	Attendance
0	Aditya	23	545	456	56
1	Hardik	24	567	478	45
2	Jiya	25	589	468	36
3	Manas	22	500	488	78
4	Yash	26	578	497	86
5	Shashwat	22	300	400	80
6	Dhruv	23	400	300	40

[14]:
```
df['Percentage']=(df['Sem I marks out of 600']+df['Sem II marks out of 500'])//11
df
```

[14]:

	Name	Age	Sem I marks out of 600	Sem II marks out of 500	Attendance	Percentage
0	Aditya	23	545	456	56	91
1	Hardik	24	567	478	45	95
2	Jiya	25	589	468	36	96
3	Manas	22	500	488	78	89
4	Yash	26	578	497	86	97
5	Shashwat	22	300	400	80	63
6	Dhruv	23	400	300	40	63

[15]:
```
def get_grade(x):
    if   x >= 90: return 'O'
    elif x >= 75: return 'A+'
    elif x >= 60: return 'A'
    elif x >= 50: return 'B+'
    elif x >= 40: return 'B'
    else:         return 'F'
df['Grade'] = df['Percentage'].apply(get_grade)
df
```

[15]:

	Name	Age	Sem I marks out of 600	Sem II marks out of 500	Attendance	Percentage	Grade
0	Aditya	23	545	456	56	91	O
1	Hardik	24	567	478	45	95	O
2	Jiya	25	589	468	36	96	O
3	Manas	22	500	488	78	89	A+
4	Yash	26	578	497	86	97	O
5	Shashwat	22	300	400	80	63	A
6	Dhruv	23	400	300	40	63	A

Ques 4: Create a DataFrame based on E-Commerce data and generate mean, mode, and median.

[7]:
```
e_comm = {
    'ItemID':[101,102,103,104],
    'Name':['Mouse','Keyboard','Headphones','Monitor'],
    'Price':[199,299,199,499]
}
e_df = pd.DataFrame(e_comm)
e_df
```

[7]:

	ItemID	Name	Price
0	101	Mouse	199
1	102	Keyboard	299
2	103	Headphones	199
3	104	Monitor	499

[10]:
```
print("Average Price is: ")
e_df['Price'].mean()
```

```
Average Price is:
```

[10]:
```
299.0
```

[11]:
```
print("Median Price is: ")
e_df['Price'].median()
```

```
Median Price is:
```

[11]:
```
249.0
```

[12]:
```
print("Mode Price is: ")
e_df['Price'].mode()
```

```
Mode Price is:
```

[12]:
```
0    199
Name: Price, dtype: int64
```

Ques 5: Write a program to implement pivot() and pivot-table() on a DataFrame.

[12]:
```
import pandas as pd
data = {
    'Day':         ['Monday', 'Monday', 'Tuesday', 'Tuesday', 'Wednesday', 'Wednesday'],
    'City':        ['Delhi', 'Mumbai', 'Delhi', 'Mumbai', 'Delhi', 'Mumbai'],
    'Temperature': [32, 34, 33, 35, 34, 36],
}
df = pd.DataFrame(data)
df
```

[12]:

	Day	City	Temperature
0	Monday	Delhi	32
1	Monday	Mumbai	34
2	Tuesday	Delhi	33
3	Tuesday	Mumbai	35
4	Wednesday	Delhi	34
5	Wednesday	Mumbai	36

[13]:
```
pivot_df = df.pivot(index='Day', columns='City', values='Temperature')
pivot_df
```

[13]:

City	Delhi	Mumbai
Day		
Monday	32	34
Tuesday	33	35
Wednesday	34	36

[14]:
```
import pandas as pd
data = {
    'Day':         ['Monday', 'Monday', 'Monday', 'Tuesday', 'Tuesday', 'Tuesday'],
    'City':        ['Delhi', 'Delhi', 'Mumbai', 'Delhi', 'Mumbai', 'Mumbai'],
    'Temperature': [32, 33, 36, 33, 36, 37],
}
df1 = pd.DataFrame(data)
df1
```

[14]:

	Day	City	Temperature
0	Monday	Delhi	32
1	Monday	Delhi	33
2	Monday	Mumbai	36
3	Tuesday	Delhi	33
4	Tuesday	Mumbai	36
5	Tuesday	Mumbai	37

[15]:
```
pivot_table_df = df1.pivot_table(index='Day', columns='City', values='Temperature', aggfunc='count')
pivot_table_df
```

[15]:

City	Delhi	Mumbai
Day		
Monday	2	1
Tuesday	1	2

Ques 6: Write a Program to read a CSV file and create its dataframe.

[16]:
```
import pandas as pd
filename = 'data.csv'
csv_df = pd.read_csv(filename)
csv_df
```

[16]:

	Name	Age	Gender
0	Ram	16	M
1	Manish	18	M
2	Sahil	15	M
3	Amrit	20	F
4	Mark	19	M

Ques 7: 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.

[13]:
```
sales = {
    'Item_Category':['Food','Stationery','Electronics','Stationery','Food','Electronics'],
    'Item_Name':['Chips','Pen','TV','Eraser','Juice','Radio'],
    'Expenditure':[20,50,400,25,60,100]
}
QtrSales =  pd.DataFrame(sales)
QtrSales
```

[13]:

	Item_Category	Item_Name	Expenditure
0	Food	Chips	20
1	Stationery	Pen	50
2	Electronics	TV	400
3	Stationery	Eraser	25
4	Food	Juice	60
5	Electronics	Radio	100

[16]:
```
avg_expenditure = QtrSales['Expenditure'].groupby(QtrSales['Item_Category'])
print(avg_expenditure.mean())
```

```
Item_Category
Electronics    250.0
Food            40.0
Stationery      37.5
Name: Expenditure, dtype: float64
```

Ques 8: Create a dataframe having age, name, weight of five students. Write a program to display the details of first and fourth rows.

[17]:
```
student = {
    'Name':['Shashwat','Khushi','Bhumika','Aditya','Jiya'],
    'Age':[20,19,21,18,22],
    'Weight':[56,78,43,55,69]
}
stu_df =  pd.DataFrame(student)
stu_df
```

[17]:

	Name	Age	Weight
0	Shashwat	20	56
1	Khushi	19	78
2	Bhumika	21	43
3	Aditya	18	55
4	Jiya	22	69

[20]:
```
stu_df.iloc[[0,3]]
```

[20]:

	Name	Age	Weight
0	Shashwat	20	56
3	Aditya	18	55

Ques 9: Write a program to create a dataframe to store weight, age and name of three people. Print the DataFrame and its transpose.

[21]:
```
people = {
    'Name':['Shashwat','Khushi','Bhumika'],
    'Age':[20,19,21],
    'Weight':[56,78,43]
}
p_df =  pd.DataFrame(people)
p_df
```

[21]:

	Name	Age	Weight
0	Shashwat	20	56
1	Khushi	19	78
2	Bhumika	21	43

[22]:
```
p_df.T
```

[22]:

	0	1	2
Name	Shashwat	Khushi	Bhumika
Age	20	19	21
Weight	56	78	43

Ques 10: Create a Panda series from dictionary of values and an ndarray.

[19]:
```
import pandas as pd

dict1 = {1:'A', 2:'B', 3:'C'}
series1 = pd.Series(dict1)
series1
```

[19]:
```
1    A
2    B
3    C
dtype: object
```

[21]:
```
import pandas as pd
import numpy as np

arr = np.array([1,2,3,4])
series2 = pd.Series(arr)
series2
```

[21]:
```
0    1
1    2
2    3
3    4
dtype: int32
```

Ques 11: Perform sorting on series data and dataframes.

[26]:
```
import pandas as pd
import numpy as np

arr = np.array([5,6,3,1,7,4,2])
series = pd.Series(arr)
sorted_s = series.sort_values()
sorted_s
```

[26]:
```
3    1
6    2
2    3
5    4
0    5
1    6
4    7
dtype: int32
```

[27]:
```
import pandas as pd
data = {
    'Day':         ['Monday', 'Monday', 'Tuesday', 'Tuesday', 'Wednesday', 'Wednesday'],
    'City':        ['Delhi', 'Mumbai', 'Delhi', 'Mumbai', 'Delhi', 'Mumbai'],
    'Temperature': [32, 34, 33, 35, 34, 36],
}
df = pd.DataFrame(data)
sorted_df = df.sort_values(by = 'Temperature')
sorted_df
```

[27]:

	Day	City	Temperature
0	Monday	Delhi	32
2	Tuesday	Delhi	33
1	Monday	Mumbai	34
4	Wednesday	Delhi	34
3	Tuesday	Mumbai	35
5	Wednesday	Mumbai	36

Ques 12: Two series objects, Population stores the details of four metro cities of India and another object, and AvgIncome stores the total average income reported in four years in these cities. Calculate income per capita for each of these metro cities.

[28]:
```
import pandas as pd

pop = {'Delhi':5000, 'Chennai':3000, 'Kolkata':2500, 'Mumbai':4000}
population = pd.Series(pop)
inc = {'Delhi':45000, 'Chennai':44000, 'Kolkata':15000, 'Mumbai':33000}
avg_inc = pd.Series(inc)

inc_per_capita = avg_inc/population
inc_per_capita
```

[28]:
```
Delhi       9.000000
Chennai    14.666667
Kolkata     6.000000
Mumbai      8.250000
dtype: float64
```

Ques 13: Series objects Temp1, Temp2, Temp3, and Temp4 store the temperature of days of week 1, week 2, week 3, week 4. Write a script to:

1. Print average temperature per week
2. Print average temperature of entire month

[2]:
```
import pandas as pd

temp1 = pd.Series([23,22,21,24,25], index = ['Mon','Tue','Wed','Thurs','Fri'])
temp2 = pd.Series([20,19,23,22,27], index = ['Mon','Tue','Wed','Thurs','Fri'])
temp3 = pd.Series([22,21,16,20,24], index = ['Mon','Tue','Wed','Thurs','Fri'])
temp4 = pd.Series([28,20,21,24,23], index = ['Mon','Tue','Wed','Thurs','Fri'])
dict1 = {'Week1':temp1, 'Week2':temp2, 'Week3':temp3, 'Week4':temp4}
df = pd.DataFrame(dict1)
df
```

[2]:

	Week1	Week2	Week3	Week4
Mon	23	20	22	28
Tue	22	19	21	20
Wed	21	23	16	21
Thurs	24	22	20	24
Fri	25	27	24	23

[3]:
```
df.mean()
```

[3]:
```
Week1    23.0
Week2    22.2
Week3    20.6
Week4    23.2
dtype: float64
```

[4]:
```
print("Average Temperatue of Entire Month: ",df.values.mean())
```

```
Average Temperatue of Entire Month:  22.25
```

Ques 14: Write a pandas program to convert a series of lists to one series.

[29]:
```
import pandas as pd

s = pd.Series([[1,2],[3,4],[5,6]])
s
```

[29]:
```
0    [1, 2]
1    [3, 4]
2    [5, 6]
dtype: object
```

[30]:
```
l = []
for i in s:
    l.extend(i)
new_s = pd.Series(l)
new_s
```

[30]:
```
0    1
1    2
2    3
3    4
4    5
5    6
dtype: int64
```

Ques 15: Write a pandas program to compare elements of two series.

```
]: import pandas as pd

s1 = pd.Series([1,2,6,8])
s2 = pd.Series([3,2,4,9])
print("Equal : ")
print(s1 == s2)
print("\nNot Equal : ")
print(s1 != s2)
print("\nGreater than : ")
print(s1 > s2)
print("\nLess than : ")
print(s1 < s2)
```

```
Equal :
0    False
1     True
2    False
3    False
dtype: bool

Not Equal :
0     True
1    False
2     True
3     True
dtype: bool

Greater than :
0    False
1    False
2     True
3    False
dtype: bool

Less than :
0     True
1    False
2    False
3     True
dtype: bool
```

Ques 16: Write a pandas program to create a subset of a given series based on values and condition.

[34]:
```
import pandas as pd

s = pd.Series([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
print("Subset of elements greater than 5")
new_s = s[s > 5]
new_s
```

```
Subset of elements greater than 5
```

[34]:
```
6    6
7    7
8    8
9    9
dtype: int64
```

Ques 17: Given the dataset Toyota.csv, do the following operations:

1. Upload Toyota.csv in the dataframe df.
2. What is the data type of MetColor?
3. How many null values are there in the KM field?
4. Which column has 7 unique values?
5. How many records are there?
6. Replace three, four, five value in Doors column to 3,4,5 respectively
7. Change the datatype of Doors to int64
8. Impute the value of Price with median
9. Replace "????" in the HP field with mean
10. Impute blank values in FuelType with mode.
11. Delete the rows with MetColor and Age as blank
12. Replace "??" value in KM with mean
13. What is the mean, median and mode of the KM field?
14. Create a new column "Category" based on the value of the column "Age".
15. Create dummy fields for the FuelType column.

[1]:
```
import pandas as pd

df = pd.read_csv('Toyota.csv')
df
```

[1]:

	Unnamed: 0	Price	Age	KM	FuelType	HP	MetColor	Automatic	CC	Doors	Weight
0	0	13500	23.0	46986	Diesel	90	1.0	0	2000	three	1165
1	1	13750	23.0	72937	Diesel	90	1.0	0	2000	3	1165
2	2	13950	24.0	41711	Diesel	90	NaN	0	2000	3	1165
3	3	14950	26.0	48000	Diesel	90	0.0	0	2000	3	1165
4	4	13750	30.0	38500	Diesel	90	0.0	0	2000	3	1170
...	...	...	...	...	...	...	...	...	...	...	...
1431	1431	7500	NaN	20544	Petrol	86	1.0	0	1300	3	1025
1432	1432	10845	72.0	??	Petrol	86	0.0	0	1300	3	1015
1433	1433	8500	NaN	17016	Petrol	86	0.0	0	1300	3	1015
1434	1434	7250	70.0	??	NaN	86	1.0	0	1300	3	1015
1435	1435	6950	76.0	1	Petrol	110	0.0	0	1600	5	1114

1436 rows × 11 columns

[2]:
```
print('Datatype of the MetColor column is ', df['MetColor'].dtype)
```

```
Datatype of the MetColor column is  float64
```

[3]:
```
print('Number of null values in KM Field ', df['KM'].isnull().sum())
```

```
Number of null values in KM Field  0
```

[5]:
```
for col in df.columns:
    if df[col].nunique() == 7:
        print(f'The {col} column has 7 unique values')
```

```
The Doors column has 7 unique values
```

[6]:
```
print('Number of records: ',len(df))
```

```
Number of records:  1436
```

[7]:
```
replace_dict = {'three': 3, 'four': 4, 'five': 5}
df['Doors'] = df['Doors'].replace(replace_dict)
df['Doors']
```

[7]:
```
0       3
1       3
2       3
3       3
4       3
       ..
1431    3
1432    3
1433    3
1434    3
1435    5
Name: Doors, Length: 1436, dtype: object
```

[8]:
```
df['Doors'] = df['Doors'].astype('int64')
df['Doors'].dtype
```

[8]:
```
dtype('int64')
```

[9]:
```
median_price = df['Price'].median()
df['Price']  = df['Price'].fillna(median_price)
df['Price']
```

[9]:
```
0       13500
1       13750
2       13950
3       14950
4       13750
        ...
1431     7500
1432    10845
1433     8500
1434     7250
1435     6950
Name: Price, Length: 1436, dtype: int64
```

[10]:
```
temp     = df['HP']
temp     = temp[temp != '????']
temp     = temp.astype('int64')
mean_hp  = temp.mean()
df['HP'] = df['HP'].replace('????', mean_hp)
df['HP']
```

[10]:
```
0       90
1       90
2       90
3       90
4       90
       ...
1431    86
1432    86
1433    86
1434    86
1435   110
Name: HP, Length: 1436, dtype: object
```

[13]:
```
mode_f  = df['FuelType'].mode()[0]
df['FuelType'] = df['FuelType'].fillna(mode_f)
df['FuelType']
```

[13]:
```
0       Diesel
1       Diesel
3       Diesel
4       Diesel
5       Diesel
         ...
1429    Petrol
1430    Petrol
1432    Petrol
1434    Petrol
1435    Petrol
Name: FuelType, Length: 1192, dtype: object
```

[12]:
```
df.dropna(subset=['MetColor', 'Age'], inplace=True)
df
```

[12]:

	Unnamed: 0	Price	Age	KM	FuelType	HP	MetColor	Automatic	CC	Doors	Weight
0	0	13500	23.0	46986	Diesel	90	1.0	0	2000	3	1165
1	1	13750	23.0	72937	Diesel	90	1.0	0	2000	3	1165
3	3	14950	26.0	48000	Diesel	90	0.0	0	2000	3	1165
4	4	13750	30.0	38500	Diesel	90	0.0	0	2000	3	1170
5	5	12950	32.0	61000	Diesel	90	0.0	0	2000	3	1170
...	...	...	...	...	...	...	...	...	...	...	...
1429	1429	8950	78.0	24000	Petrol	86	1.0	1	1300	5	1065
1430	1430	8450	80.0	23000	Petrol	86	0.0	0	1300	3	1015
1432	1432	10845	72.0	??	Petrol	86	0.0	0	1300	3	1015
1434	1434	7250	70.0	??	Petrol	86	1.0	0	1300	3	1015
1435	1435	6950	76.0	1	Petrol	110	0.0	0	1600	5	1114

1192 rows × 11 columns

[14]:
```
temp     = df['KM']
temp     = temp[temp != '??']
temp     = temp.astype('int64')
mean_km  = temp.mean()
df['KM'] = df['KM'].replace('??', mean_km)
df['KM']
```

[14]:
```
0              46986
1              72937
3              48000
4              38500
5              61000
             ...
1429           24000
1430           23000
1432    69006.620017
1434    69006.620017
1435               1
Name: KM, Length: 1192, dtype: object
```

[16]:
```
km = df['KM'].astype('int64')
print('Mean of KM: ',km.mean())
print('Median of KM: ',km.median())
print('Mode of KM: ',km.mode()[0])
```

```
Mean of KM:  69006.61325503356
Median of KM:  63875.5
Mode of KM:  69006
```

[17]:
```
def func(val):
    if   val <= 10: return 'New'
    elif val <= 20: return 'Medium'
    else:           return 'Old'
df['Category'] = df['Age'].apply(func)
df
```

[17]:

	Unnamed: 0	Price	Age	KM	FuelType	HP	MetColor	Automatic	CC	Doors	Weight	Category
0	0	13500	23.0	46986	Diesel	90	1.0	0	2000	3	1165	Old
1	1	13750	23.0	72937	Diesel	90	1.0	0	2000	3	1165	Old
3	3	14950	26.0	48000	Diesel	90	0.0	0	2000	3	1165	Old
4	4	13750	30.0	38500	Diesel	90	0.0	0	2000	3	1170	Old
5	5	12950	32.0	61000	Diesel	90	0.0	0	2000	3	1170	Old
...	...	...	...	...	...	...	...	...	...	...	...	...
1429	1429	8950	78.0	24000	Petrol	86	1.0	1	1300	5	1065	Old
1430	1430	8450	80.0	23000	Petrol	86	0.0	0	1300	3	1015	Old
1432	1432	10845	72.0	69006.620017	Petrol	86	0.0	0	1300	3	1015	Old
1434	1434	7250	70.0	69006.620017	Petrol	86	1.0	0	1300	3	1015	Old
1435	1435	6950	76.0	1	Petrol	110	0.0	0	1600	5	1114	Old

1192 rows × 12 columns

[ ]:
```
df = pd.get_dummies(df, columns=['FuelType'])
```

Ques 18: Write a pandas program to change the order of index of a given series.

[21]:
```
series1 = pd.Series(data = [1, 2, 3, 4, 5], index = ['A', 'B', 'C', 'D', 'E'])
print("Initial Index: ")
print(series1)

series2 = series1.reindex(index = ['E','A','B','D','C'])
print("Reordered Index: ")
print(series2)
```

```
Initial Index:
A    1
B    2
C    3
D    4
E    5
dtype: int64
Reordered Index:
E    5
A    1
B    2
D    4
C    3
dtype: int64
```

Ques 19: Write a pandas program to get the items of a given series not present in another given series.

[25]:
```
series1 = pd.Series([1, 2, 3, 4, 5])
series2 = pd.Series([2, 3, 5, 6, 7])
print('Elements of Series1 not in Series2: ')
print(series1[~series1.isin(series2)])
```

```
Elements of Series1 not in Series2:
0    1
3    4
dtype: int64
```

Ques 20: Write a pandas program to calculate the frequency counts of each unique value of a given series.

[26]:
```
import pandas as pd

s1 = pd.Series([1,2,3,1,2,2,3,1,1])
freq = s1.value_counts()
print(freq)
```

```
1    4
2    3
3    2
dtype: int64
```

Ques 21: Create a series and print all the elements that are above 75th percentile.

[27]:
```
import pandas as pd

s1 = pd.Series([20,21,23,45,67,89])
per75 = s1[s1>s1.quantile(0.75)]
print(per75)
```

```
4    67
5    89
dtype: int64
```

Ques 22: Write a program to find the MAD (mean absolute deviation) of all columns in a dataframe.

[30]:
```
import pandas as pd

df = pd.DataFrame({'A':[20,23,57,78],'B':[31,90,64,72]})
df

ans = pd.Series(dtype='float64')
for col in df:
    mean = df[col].mean()
    sub = (df[col]-mean).abs()
    mad = sub.mean()
    ans[col] = mad

print("Mean Absolute Deviation is: ")
print(ans)
```

```
Mean Absolute Deviation is:
A    23.00
B    16.75
dtype: float64
```

Ques 23: Create a dataframe based on employee data and generate quartile and variance.

[32]:
```
import pandas as pd

df = pd.DataFrame({'Name':['A', 'B', 'C', 'D', 'E'],'Salary':[1000, 2000, 3000, 4000, 5000],'Age':[20, 30, 40, 50, 60]})
quartiles = df.quantile([0.25, 0.5, 0.75], numeric_only=True)
variances = df.var(numeric_only=True)
print('Quartiles: ')
print(quartiles)
print('Variances: ')
print(variances)
```

```
Quartiles:
      Salary   Age
0.25  2000.0  30.0
0.50  3000.0  40.0
0.75  4000.0  50.0
Variances:
Salary    2500000.0
Age           250.0
dtype: float64
```

Ques 24: Write a program to implement skewness on random data.

[34]:
```
import numpy as np
import pandas as pd
import random

df = pd.DataFrame({'randint': np.random.randint(1, 100, 1000),'random':  np.random.random(1000)})
skew_df = df.skew()
print("Skewness on random data:")
print(skew_df)
```

```
Skewness on random data:
randint   -0.078593
random    -0.042288
dtype: float64
```

Ques 25: Create a dataframe of any data and compute the kurtosis.

[35]:
```
import numpy as np
import pandas as pd
import random

df = pd.DataFrame({'randint': np.random.randint(1, 100, 1000),'random':  np.random.random(1000)})
kurt_df = df.kurtosis()
print("Kurtosis on random data:")
print(kurt_df)
```

```
Kurtosis on random data:
randint   -1.202750
random    -1.181024
dtype: float64
```

Ques 26: Give the code or syntax to perform the following operation on two 2D numpy array array1 and array2 and 1D array array3:

1) Add array1 and array2
2) Find sum of array1 elements over a given axis
3) Find product of array2 elements over a given axis
4) Change the dimension of array3 to 2D
5) Transpose the array created in part 4
6) Display 2 rows and the third column of the 2D array
7) Join two 2D arrays along row
8) Convert array2 to a 1D array
9) Split array1 into multiple subarrays

[36]:
```
import numpy as np

array1 = np.array([[1,2,3], [4,5,6]])
array2 = np.array([[7,8,9], [10,11,12]])
array3 = np.array([13,14,15,16])

print(array1+array2)
```

```
[[ 8 10 12]
 [14 16 18]]
```

[37]:
```
print(np.sum(array1,axis=0))
```

```
[5 7 9]
```

[38]:
```
print(np.prod(array2, axis=0))
```

```
[ 70  88 108]
```

[40]:
```
new = array3.reshape((2,2))
print(new)
```

```
[[13 14]
 [15 16]]
```

[41]:
```
print(new.T)
```

```
[[13 15]
 [14 16]]
```

[43]:
```
print(array1[:2,2])
```

```
[3 6]
```

[45]:
```
print(np.concatenate((array1,array2), axis=1))
```

```
[[ 1  2  3  7  8  9]
 [ 4  5  6 10 11 12]]
```

[46]:
```
print(array2.flatten())
```

```
[ 7  8  9 10 11 12]
```

[47]:
```
print(np.array_split(array2,2))
```

```
[array([[7, 8, 9]]), array([[10, 11, 12]])]
```

Ques 27: Write python code to create the following series:

101	Harsh
102	Arun
103	Ankur
104	Harpal
105	Divya
106	Jeet

1) Show the details of the first 3 employees using the head method
2) Show the details of the last 3 employees using the tail method
3) Show the details of the first 3 employees without using the head method
4) Show the details of the last 3 employees without using the tail method
5) Show the value of index number 102
6) Show 2nd to 4th records (inclusive)
7) Show the values at indexes 101, 103, and 105
8) Show details of "Arun"

[48]:
```
import pandas as pd

series = pd.Series(['Harsh', 'Arun', 'Ankur', 'Harpal', 'Divya', 'Jeet'],index=[101, 102, 103, 104, 105, 106])
print(series)
```

```
101     Harsh
102      Arun
103     Ankur
104    Harpal
105     Divya
106      Jeet
dtype: object
```

[49]:
```
print(series.head(3))
```

```
101    Harsh
102     Arun
103    Ankur
dtype: object
```

[50]:
```
print(series.tail(3))
```

```
104    Harpal
105     Divya
106      Jeet
dtype: object
```

[51]:
```
print(series[0:3])
```

```
101    Harsh
102     Arun
103    Ankur
dtype: object
```

[53]:
```
print(series[-3:])
```

```
104    Harpal
105     Divya
106      Jeet
dtype: object
```

[54]:
```
print(series[102])
```

```
Arun
```

[55]:
```
print(series[2:5])
```

```
103     Ankur
104    Harpal
105     Divya
dtype: object
```

[57]:
```
print(series[[101,103,105]])
```

```
101    Harsh
103    Ankur
105    Divya
dtype: object
```

[58]:
```
print(series[series=='Arun'])
```

```
102    Arun
dtype: object
```

Ques 28: Create a dataframe for the below given data:

SNO	Batsman	Test	ODI	T20
1	Virat Kohli	3543	2245	1925
2	Ajinkya Rehane	2578	2165	1853
3	Rohit Sharma	2280	2080	1522
4	Shikhar Dhawan	2158	1957	1020
5	Hardik Pandya	1879	1856	980

1. Print the batsman name along with runs scored in Test and T20 using column names and dot notation
2. Display the batsman name along with runs scored in the ODI using the loc method
3. Display the batsman details who scored runs:
    - More than 2000 in ODI
    - Less than 2500 in Test
    - More than 1500 in T20
4. Display the columns using the column index number like 0, 2, and 4
5. Display the alternate rows
6. Reindex the dataframe created above with batsman name and delete the data of Hardik Pandya and Shikhar Dhawan by their index from the original dataframe
7. Insert two rows in the dataframe and delete rows whose index is 1 and 4
8. Delete the column Test, and add one more column “total”, ie the total of ODI and T20 runs
9. Rename the column “T20” to “T20I Runs”
10. Print the dataframe without headers

[59]:
```
import pandas as pd

data = {'SNO':     [1, 2, 3, 4, 5],
    'Batsman': ['Virat Kohli', 'Ajinkya Rehane', 'Rohit Sharma', 'Shikhar Dhawan', 'Hardik Pandya'],
    'Test':[3543, 2578, 2280, 2158, 1879],'ODI':[2245, 2165, 2080, 1957, 1856],'T20':[1925, 1853, 1522, 1020, 980]}

df = pd.DataFrame(data)
df
```

[59]:

	SNO	Batsman	Test	ODI	T20
0	1	Virat Kohli	3543	2245	1925
1	2	Ajinkya Rehane	2578	2165	1853
2	3	Rohit Sharma	2280	2080	1522
3	4	Shikhar Dhawan	2158	1957	1020
4	5	Hardik Pandya	1879	1856	980

[60]:
```
new_df = pd.DataFrame({'Batsman': df.Batsman, 'Test': df.Test, 'T20': df.T20})
new_df
```

[60]:

	Batsman	Test	T20
0	Virat Kohli	3543	1925
1	Ajinkya Rehane	2578	1853
2	Rohit Sharma	2280	1522
3	Shikhar Dhawan	2158	1020
4	Hardik Pandya	1879	980

[61]:
```
df.loc[:,['Batsman','ODI']]
```

[61]:

	Batsman	ODI
0	Virat Kohli	2245
1	Ajinkya Rehane	2165
2	Rohit Sharma	2080
3	Shikhar Dhawan	1957
4	Hardik Pandya	1856

[62]:
```
df[df['ODI']>2000]
```

[62]:

	SNO	Batsman	Test	ODI	T20
0	1	Virat Kohli	3543	2245	1925
1	2	Ajinkya Rehane	2578	2165	1853
2	3	Rohit Sharma	2280	2080	1522

[63]:
```
df[df['Test']<2500]
```

[63]:

	SNO	Batsman	Test	ODI	T20
2	3	Rohit Sharma	2280	2080	1522
3	4	Shikhar Dhawan	2158	1957	1020
4	5	Hardik Pandya	1879	1856	980

[64]:
```
df[df['T20']>1500]
```

[64]:

	SNO	Batsman	Test	ODI	T20
0	1	Virat Kohli	3543	2245	1925
1	2	Ajinkya Rehane	2578	2165	1853
2	3	Rohit Sharma	2280	2080	1522

[65]:
```
df.iloc[:,[0,2,4]]
```

[65]:

	SNO	Test	T20
0	1	3543	1925
1	2	2578	1853
2	3	2280	1522
3	4	2158	1020
4	5	1879	980

[66]:
```
df[::2]
```

[66]:

	SNO	Batsman	Test	ODI	T20
0	1	Virat Kohli	3543	2245	1925
2	3	Rohit Sharma	2280	2080	1522
4	5	Hardik Pandya	1879	1856	980

[69]:
```
new_df.index = new_df['Batsman']
new_df = new_df.drop(columns='Batsman')
drop_indices = df.query('Batsman == "Hardik Pandya" or Batsman == "Shikhar Dhawan"').index
new_df = df.drop(index=drop_indices)
new_df
```

[69]:

	SNO	Batsman	Test	ODI	T20
0	1	Virat Kohli	3543	2245	1925
1	2	Ajinkya Rehane	2578	2165	1853
2	3	Rohit Sharma	2280	2080	1522

[71]:
```
to_insert = pd.DataFrame({'SNO':[6, 7],'Batsman': ['Rishabh Pant', 'KL Rahul'],
    'Test':[1500, 1200],'ODI':[1000, 900],'T20':[800, 700]})
new_df = pd.concat([df, to_insert], ignore_index=True)
new_df.drop(index=[1, 4], inplace=True)
new_df
```

[71]:

	SNO	Batsman	Test	ODI	T20
0	1	Virat Kohli	3543	2245	1925
2	3	Rohit Sharma	2280	2080	1522
3	4	Shikhar Dhawan	2158	1957	1020
5	6	Rishabh Pant	1500	1000	800
6	7	KL Rahul	1200	900	700

[72]:
```
new_df= df.drop(columns='Test')
new_df['total'] = new_df['ODI'] + new_df['T20']
new_df
```

[72]:

	SNO	Batsman	ODI	T20	total
0	1	Virat Kohli	2245	1925	4170
1	2	Ajinkya Rehane	2165	1853	4018
2	3	Rohit Sharma	2080	1522	3602
3	4	Shikhar Dhawan	1957	1020	2977
4	5	Hardik Pandya	1856	980	2836

[73]:
```
new_df = df.rename(columns={'T20': 'T20I Runs'})
new_df
```

[73]:

	SNO	Batsman	Test	ODI	T20I Runs
0	1	Virat Kohli	3543	2245	1925
1	2	Ajinkya Rehane	2578	2165	1853
2	3	Rohit Sharma	2280	2080	1522
3	4	Shikhar Dhawan	2158	1957	1020
4	5	Hardik Pandya	1879	1856	980

[76]:
```
print(new_df.to_string(header=False, index=False))
```

```
1    Virat Kohli 3543 2245 1925
2 Ajinkya Rehane 2578 2165 1853
3   Rohit Sharma 2280 2080 1522
4 Shikhar Dhawan 2158 1957 1020
5  Hardik Pandya 1879 1856  980
```

Ques 29: Create the following dataframe "Sales" containing year-wise sales figures for five salespersons in INR. Use the years as column labels and the salesperson names as indexes.

	2014	**2015**	**2016**	**2017**
Madhu	100.5	12000	2000	50000
Kusum	150.8	18000	5000	60000
Kinshuk	200.9	22000	70000	70000
Ankit	30000	30000	1000	80000
Shruti	40000	45000	1250	90000

1. Display the indexes
2. Display the names of the columns
3. Display the dimensions, shape, size, and values
4. Display the last two rows
5. Display the first two columns
6. Change the dataframe Sales such that it becomes its transpose
7. Add data to Sales for the salesman "Sumeet" where the sales made are [196.2, 37800, 52000, 78438] in the years [2014, 2015, 2016, 2017] respectively
8. Delete the data for the the year 2014
9. Update the sale made by Shruti in 2017 to 100000
10. Export the dataframe Sales to a comma separated file "SalesFigures.csv" on the disk. Do not export the indexes or column names
11. Change the name of the salesperson "Ankit" to "Vivaan" and "Kinshuk" to "Shailesh"
12. Delete the data for the salesman "Madhu"

[77]:
```
import pandas as pd
data = {'2014': [100.5, 150.8, 200.9, 30000, 40000],'2015': [12000, 18000, 22000, 30000, 45000],
    '2016': [2000,  5000,  70000, 1000,  1250],'2017': [50000, 60000, 70000, 80000, 90000]}
Sales = pd.DataFrame(data,index=['Madhu', 'Kusum', 'Kinshuk', 'Ankit', 'Shruti'])
Sales
```

[77]:

	2014	2015	2016	2017
Madhu	100.5	12000	2000	50000
Kusum	150.8	18000	5000	60000
Kinshuk	200.9	22000	70000	70000
Ankit	30000.0	30000	1000	80000
Shruti	40000.0	45000	1250	90000

[78]:
```
print(Sales.index)
```
```
Index(['Madhu', 'Kusum', 'Kinshuk', 'Ankit', 'Shruti'], dtype='object')
```

[79]:
```
print(Sales.columns)
```
```
Index(['2014', '2015', '2016', '2017'], dtype='object')
```

[80]:
```
print(Sales.ndim)
```
```
2
```

[81]:
```
print(Sales.shape)
```
```
(5, 4)
```

[82]:
```
print(Sales.size)
```
```
20
```

[83]:
```
print(Sales.values)
```
```
[[  100.5 12000.   2000.  50000. ]
 [  150.8 18000.   5000.  60000. ]
 [  200.9 22000.  70000.  70000. ]
 [30000.  30000.   1000.  80000. ]
 [40000.  45000.   1250.  90000. ]]
```

[84]:
```
print(Sales.tail(2))
```
```
           2014   2015  2016   2017
Ankit   30000.0  30000  1000  80000
Shruti  40000.0  45000  1250  90000
```

[86]:
```
Sales.iloc[:,0:2]
```

[86]:

	2014	2015
Madhu	100.5	12000
Kusum	150.8	18000
Kinshuk	200.9	22000
Ankit	30000.0	30000
Shruti	40000.0	45000

[87]:
```
Sales.T
```

[87]:

	Madhu	Kusum	Kinshuk	Ankit	Shruti
2014	100.5	150.8	200.9	30000.0	40000.0
2015	12000.0	18000.0	22000.0	30000.0	45000.0
2016	2000.0	5000.0	70000.0	1000.0	1250.0
2017	50000.0	60000.0	70000.0	80000.0	90000.0

[88]:
```
to_add = pd.DataFrame([[196.2, 37800, 52000, 78438]], columns=Sales.columns, index=['Sumeet'])
new_df = pd.concat([Sales, to_add])
new_df
```

[88]:

	2014	2015	2016	2017
Madhu	100.5	12000	2000	50000
Kusum	150.8	18000	5000	60000
Kinshuk	200.9	22000	70000	70000
Ankit	30000.0	30000	1000	80000
Shruti	40000.0	45000	1250	90000
Sumeet	196.2	37800	52000	78438

[94]:
```
new = Sales.drop(['2014'],axis=1)
new
```

[94]:

	2015	2016	2017
Madhu	12000	2000	50000
Kusum	18000	5000	60000
Kinshuk	22000	70000	70000
Ankit	30000	1000	80000
Shruti	45000	1250	90000

[95]:
```
Sales.loc['Shruti','2017']=100000
Sales
```

[95]:

	2014	2015	2016	2017
Madhu	100.5	12000	2000	50000
Kusum	150.8	18000	5000	60000
Kinshuk	200.9	22000	70000	70000
Ankit	30000.0	30000	1000	80000
Shruti	40000.0	45000	1250	100000

```
Sales.to_csv('SalesFigures.csv', index=False, header=False)
print('Successfully exported the Sales dataframe to SalesFigures.csv without indexes and column names')
```

[4]:
```
to_rename = {'Ankit': 'Vivaan', 'Kinshuk': 'Shailesh'}
new_df = Sales.rename(index=to_rename)
new_df
```

[4]:

	2014	2015	2016	2017
Madhu	100.5	12000	2000	50000
Kusum	150.8	18000	5000	60000
Shailesh	200.9	22000	70000	70000
Vivaan	30000.0	30000	1000	80000
Shruti	40000.0	45000	1250	90000

[5]:
```
new_df = Sales.drop(index='Madhu')
new_df
```

[5]:

	2014	2015	2016	2017
Kusum	150.8	18000	5000	60000
Kinshuk	200.9	22000	70000	70000
Ankit	30000.0	30000	1000	80000
Shruti	40000.0	45000	1250	90000

Ques 30: Write code to create the following dataframe "Patient" and write code to perform the following operations:

PatientID	Treatment_Starts	Drug	Dosage
PT1	1/14/16	CISPLATIN	200
PT20	1/2/16	NIVOLUNAB	140
PT2	1/10/16	CISPLATIN	180
PT5	1/24/16	CISPLATIN	140
PT8	2/14/16	CISPLATIN	190

1. Check for the number of rows in the dataframe
2. Show the datatype of each column
3. Display the first and third column
4. List the number of unique drugs
5. Show the record of the patient with the ID of "PT5" and the "CISPLATIN" drug
6. Display all the rows where the dosage is greater than 180
7. Sort the original dataframe in ascending order of "PatientID" and in descending order of "Treatment_Starts"
8. Show all the drugs and how many patients received those drugs
9. Create a bar chart in seaborn to compare the counts for the two drugs
10. Display the average dosage of each drug

[7]:
```
import pandas as pd
import matplotlib.pyplot as plt
data = {
    'PatientID':        ['PT1', 'PT20', 'PT2', 'PT5', 'PT8'],
    'Treatment_Starts': ['1/14/16', '1/2/16', '1/10/16', '1/24/16', '2/14/16'],
    'Drug':             ['CISPLATIN', 'NIVOLUNAB', 'CISPLATIN', 'CISPLATIN', 'CISPLATIN'],
    'Dosage':           [200, 140, 180, 140, 190]
}

Patient = pd.DataFrame(data)
Patient
```

[7]:

	PatientID	Treatment_Starts	Drug	Dosage
0	PT1	1/14/16	CISPLATIN	200
1	PT20	1/2/16	NIVOLUNAB	140
2	PT2	1/10/16	CISPLATIN	180
3	PT5	1/24/16	CISPLATIN	140
4	PT8	2/14/16	CISPLATIN	190

[8]:
```
print('Number of rows in the dataframe: ',Patient.shape[0])
```

```
Number of rows in the dataframe:  5
```

[9]:
```
print(Patient.dtypes)
```

```
PatientID           object
Treatment_Starts    object
Drug                object
Dosage               int64
dtype: object
```

[10]:
```
Patient.iloc[:, [0,2]]
```

[10]:

	PatientID	Drug
0	PT1	CISPLATIN
1	PT20	NIVOLUNAB
2	PT2	CISPLATIN
3	PT5	CISPLATIN
4	PT8	CISPLATIN

[11]:
```
print('Number of unique drugs: ',Patient["Drug"].nunique())
```

```
Number of unique drugs:  2
```

[12]:
```
new_df = Patient.query('PatientID == "PT5" and Drug == "CISPLATIN"')
new_df
```

[12]:

	PatientID	Treatment_Starts	Drug	Dosage
3	PT5	1/24/16	CISPLATIN	140

[13]:
```
new_df = Patient.query('Dosage > 180')
new_df
```

[13]:

	PatientID	Treatment_Starts	Drug	Dosage
0	PT1	1/14/16	CISPLATIN	200
4	PT8	2/14/16	CISPLATIN	190

[14]:
```
new_df = Patient.sort_values(by=['PatientID', 'Treatment_Starts'], ascending=[True, False])
new_df
```

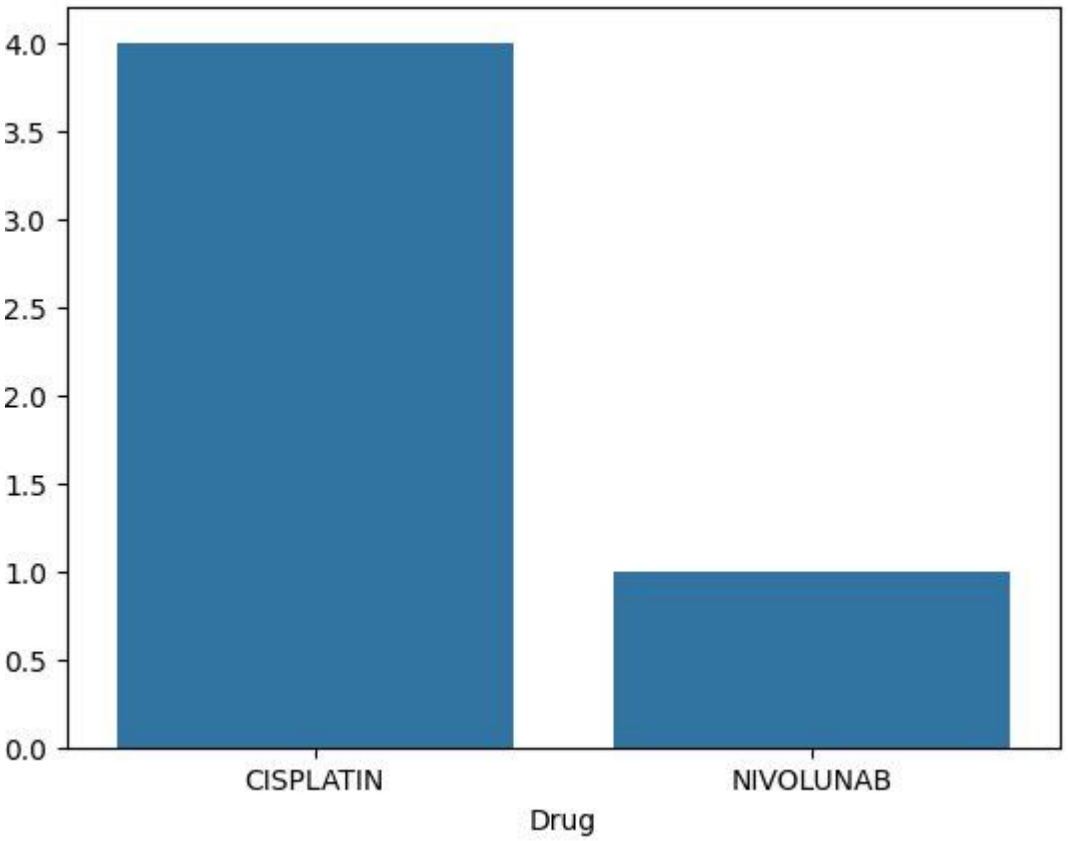
[14]:

	PatientID	Treatment_Starts	Drug	Dosage
0	PT1	1/14/16	CISPLATIN	200
2	PT2	1/10/16	CISPLATIN	180
1	PT20	1/2/16	NIVOLUNAB	140
3	PT5	1/24/16	CISPLATIN	140
4	PT8	2/14/16	CISPLATIN	190

[15]:
```
new_df = Patient['Drug'].value_counts()
new_df
```

[15]:
```
CISPLATIN    4
NIVOLUNAB    1
Name: Drug, dtype: int64
```

[ ]:
```
import seaborn as sns
counts = Patient['Drug'].value_counts()
sns.barplot(x=counts.index, y=counts.values)
plt.show()
```



[16]:
```
new_df = Patient.pivot_table(index='Drug', values='Dosage', aggfunc='mean')
new_df
```

[16]:

Drug	Dosage
CISPLATIN	177.5
NIVOLUNAB	140.0

Ques 31: Given a csv file named milknew.csv, write a program to implement a linear regression model to investigate the relationship between the dependent and independent variables present in the dataset. Evaluate the model's performance using two key metrics: the coefficient of determination (R^2) and the Mean Squared Error (MSE).

```
[*]: import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
df = pd.read_csv('milknew.csv')
to_replace = {'low':    '1','medium': '2', 'high':   '3'}
df['Grade'] = df['Grade'].replace(to_replace).astype(int)
ATTRS = ['pH', 'Temprature', 'Taste', 'Odor', 'Fat', 'Turbidity', 'Colour']
LABEL = 'Grade'
X = df[ATTRS]
Y = df[LABEL]
model = LinearRegression()
model.fit(X, Y)
print('\nThe relationship between the dependent and independent variables:')
print('Coefficients:', model.coef_)
print('Intercept:',    model.intercept_)
r2  = r2_score(Y, model.predict(X))
mse = mean_squared_error(Y, model.predict(X))
print('\nPerformance Metrics:')
print('R²: ', r2)
print('MSE:', mse)
```

```
The relationship between the dependent and independent variables:
Coefficients: [ 0.09557826 -0.03288794 -0.11862257  0.28412461  0.37620376 -0.36523753
 -0.00471413]
Intercept: 3.7134139858866346

Performance Metrics:
R²:  0.2768402608210946
MSE: 0.4484672271421274
```