ICP4—BIGDATA ANALTICS

Github link :
https://drive.google.com/file/d/1Eup1PRUXnfwBoAEys0tfsiKO1rVXx1bg/view?usp=sharing

SCREEN SHOTS

```python
# 1.Converting into a Data Frame
import numpy as np
import pandas as pd

data = {
    'ID': np.arange(1, 1000001),  # 1 million IDs
    'Value': np.random.rand(1000000),  # 1 million random values
    'Category': np.random.choice(['A', 'B', 'C', 'D'], size=1000000)  # Random categories
}

df = pd.DataFrame(data)
```

```python
# 2.printing the first 10 rows of data
print(df.head(10))
```

```
   ID     Value Category
0   1  0.164877        A
1   2  0.626932        C
2   3  0.850543        B
3   4  0.210900        A
4   5  0.709576        D
5   6  0.669209        A
6   7  0.218002        C
7   8  0.549475        B
8   9  0.534078        D
9  10  0.784112        A
```

```
[ ] # 3.Access the column “Value”
    print(df['Value'])
```

```
0         0.164877
1         0.626932
2         0.850543
3         0.210900
4         0.709576
            ...
999995    0.921747
999996    0.519572
999997    0.465576
999998    0.623257
999999    0.008767
Name: Value, Length: 1000000, dtype: float64
```

```
print(df['Value'].head())
```

```
0    0.164877
1    0.626932
2    0.850543
3    0.210900
4    0.709576
Name: Value, dtype: float64
```

```
[ ] # 4. Modify column names and display first 5 rows
    df.columns = ['ID number', 'Random value', 'Choice']
    print(df.head())
```

```
   ID number  Random value Choice
0          1      0.164877      A
1          2      0.626932      C
2          3      0.850543      B
3          4      0.210900      A
4          5      0.709576      D
```

```python
# 5.Removing the bugs and errors for the given code
import pandas as pd

pd.set_option('display.max_rows', None)
# pd.set_option('display.max_columns', None)

student_data = pd.DataFrame({
    'school_code': ['s001', 's002', 's003', 's001', 's002', 's004'],
    'class': ['V', 'V', 'VI', 'VI', 'V', 'VI'],
    'name': ['Alberto Franco', 'Gino Mcneill', 'Ryan Parkes', 'Eesha Hinton', 'Gino Mcneill', 'David Parkes'],
    'date_Of_Birth': ['15/05/2002', '17/05/2002', '16/02/1999', '25/09/1998', '11/05/2002', '15/09/1997'],
    'age': [12, 12, 13, 14, 12, 12],
    'height': [173, 192, 186, 167, 151, 159],
    'weight': [35, 32, 33, 30, 31, 32],
    'address': ['street1', 'street2', 'street3', 'street1', 'street2', 'street4']
}, index=['S1', 'S2', 'S3', 'S4', 'S5', 'S6'])

print("Original DataFrame:")
print(student_data)

print('\nSplit the said data on school_code, class wise:')
result = student_data.groupby(['school_code', 'class'])

for name, group in result:
    print("\nGroup:")
    print(name)
    print(group)
```

```
Original DataFrame:
   school_code class            name date_Of_Birth  age  height  weight  \
S1        s001     V  Alberto Franco    15/05/2002   12     173      35
S2        s002     V    Gino Mcneill    17/05/2002   12     192      32
S3        s003    VI     Ryan Parkes    16/02/1999   13     186      33
S4        s001    VI    Eesha Hinton    25/09/1998   14     167      30
S5        s002     V    Gino Mcneill    11/05/2002   12     151      31
S6        s004    VI    David Parkes    15/09/1997   12     159      32

    address
S1  street1
S2  street2
S3  street3
S4  street1
S5  street2
S6  street4

Split the said data on school_code, class wise:
```

```
Split the said data on school_code, class wise:

Group:
('s001', 'V')
   school_code class            name date_Of_Birth  age  height  weight  \
S1        s001     V  Alberto Franco    15/05/2002   12     173      35

    address
S1  street1

Group:
('s001', 'VI')
   school_code class          name date_Of_Birth  age  height  weight  address
S4        s001    VI  Eesha Hinton    25/09/1998   14     167      30  street1

Group:
('s002', 'V')
   school_code class          name date_Of_Birth  age  height  weight  address
S2        s002     V  Gino Mcneill    17/05/2002   12     192      32  street2
S5        s002     V  Gino Mcneill    11/05/2002   12     151      31  street2

Group:
('s003', 'VI')
   school_code class         name date_Of_Birth  age  height  weight  address
S3        s003    VI  Ryan Parkes    16/02/1999   13     186      33  street3

Group:
('s004', 'VI')
   school_code class          name date_Of_Birth  age  height  weight  address
S6        s004    VI  David Parkes    15/09/1997   12     159      32  street4
```

```
#6.
df = pd.read_csv("/content/data.csv")
```

```
df.head()
```

	Duration	Pulse	Maxpulse	Calories
0	60	110	130	409.1
1	60	117	145	479.0
2	60	103	135	340.0
3	45	109	175	282.4
4	45	117	148	406.0

```
[ ] # 7.
df.describe()
```

	Duration	Pulse	Maxpulse	Calories
count	169.000000	169.000000	169.000000	164.000000
mean	63.846154	107.461538	134.047337	375.790244
std	42.299949	14.510259	16.450434	266.379919
min	15.000000	80.000000	100.000000	50.300000
25%	45.000000	100.000000	124.000000	250.925000
50%	60.000000	105.000000	131.000000	318.600000
75%	60.000000	111.000000	141.000000	387.600000
max	300.000000	159.000000	184.000000	1860.400000

```
#8.
df.isnull().sum()
```

	0
Duration	0
Pulse	0
Maxpulse	0
Calories	5

dtype: int64

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

	0
Duration	63.846154
Pulse	107.461538
Calories	375.544379

dtype: float64

```python
# Replace nulls in numeric columns with their respective column means
df.fillna(df.mean(numeric_only=True), inplace=True)

# Confirm that nulls are handled
print("\nNull values after replacement:")
df.isnull().sum()
```

```
Null values after replacement:
```

	0
Duration	0
Pulse	0
Maxpulse	0
Calories	0

dtype: int64

```python
#9.
# Select two columns
selected_columns = df[['Duration', 'Calories']]

# Apply aggregation functions
aggregation_result = selected_columns.agg(['min', 'max', 'count', 'mean'])

# Display the result
print(aggregation_result)
```

```
         Duration     Calories
min     15.000000    50.300000
max    300.000000  1860.400000
count  169.000000   169.000000
mean    63.846154   375.790244
```

```
[ ] # 10.
    # Filter rows where Calories are between 500 and 1000 (inclusive)
    filtered_df = df[(df['Calories'] >= 500) & (df['Calories'] <= 1000)]

    # Display the filtered data
    print(filtered_df)
```

```
     Duration  Pulse  Maxpulse  Calories
51         80    123       146     643.1
62        160    109       135     853.0
65        180     90       130     800.4
66        150    105       135     873.4
67        150    107       130     816.0
72         90    100       127     700.0
73        150     97       127     953.2
75         90     98       125     563.2
78        120    100       130     500.4
83        120    100       130     500.0
90        180    101       127     600.1
99         90     93       124     604.1
101        90     90       110     500.0
102        90     90       100     500.0
103        90     90       100     500.4
106       180     90       120     800.3
108        90     90       120     500.3
```

```
# 11.
# Filter rows where Calories > 500 AND Pulse < 100
filtered_df = df[(df['Calories'] > 500) & (df['Pulse'] < 100)]

# Display the filtered rows
print(filtered_df)
```

```
     Duration  Pulse  Maxpulse  Calories
65        180     90       130     800.4
70        150     97       129    1115.0
73        150     97       127     953.2
75         90     98       125     563.2
99         90     93       124     604.1
103        90     90       100     500.4
106       180     90       120     800.3
108        90     90       120     500.3
```

```
[ ] # 12.
    df_modified = df.drop(columns=['Maxpulse'])
    df_modified
```

Show hidden output

```
# 13.
df.drop(columns=['Maxpulse'], inplace=True)
df.columns
```

```
Index(['Duration', 'Pulse', 'Calories'], dtype='object')
```

```
[ ] # 14.
    df['Calories'] = df['Calories'].astype(int)
```

```
[ ] # 15.
    import matplotlib.pyplot as plt

    df.plot.scatter(x='Duration', y='Calories', title='Duration vs Calories')
    plt.show()
```