

ICP 3

```

value_column = df['Value']
print(value_column)

0      0.579471
1      0.365414
2      0.128926
3      0.137030
4      0.920268
...
999995  0.926394
999996  0.426359
999997  0.904199
999998  0.019513
999999  0.260912
Name: Value, Length: 1000000, dtype: float64

[ ] df_renamed = df.rename(columns={'ID': 'ID number', 'Value': 'Random value', 'Category': 'Choice'})
print(df_renamed.head(5))

ID number  Random value  Choice
0          1      0.579471      B
1          2      0.365414      D
2          3      0.128926      B
3          4      0.137030      B
4          5      0.920268      A

[ ] import pandas as pd

# Optional setting for display purposes (not a bug)
pd.set_option('display.max_rows', None)

```

```
import pandas as pd

# Optional setting for display purposes (not a bug)
pd.set_option('display.max_rows', None)

# Corrected data with matching lengths for all columns
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', 'David Parkes', 'Praveen'],
    'date_of_Birth': ['15/05/2002', '17/05/2002', '16/02/1999', '25/09/1998', '15/09/1997', '10/08/2000'],
    'age': [12, 12, 13, 13, 12, 14],
    '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	address
S1	S001	V	Alberto Franco	15/05/2002	12	173	35

```
print(group)
```

Original DataFrame:

school_code	class	name	date_of_Birth	age	height	weight	address
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	13	167	30
S5	S002	V	David Parkes	15/09/1997	12	151	31
S6	S004	VI	Praveen	10/08/2000	14	159	32

address

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

Split the said data on school_code, class wise:

Group:
(('S001', 'V'))

school_code	class	name	date_of_Birth	age	height	weight	address
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	13	167	30 street1

```
+ Code + Text

[20] from google.colab import drive
drive.mount('/content/drive')

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).

[22] import pandas as pd

[24] dff=pd.read_csv('/content/drive/My Drive/data.csv')

[11] des=dff.describe()
print(des)
```

0s completed at 9:00 PM

```
[24] dff=pd.read_csv('/content/drive/My Drive/data.csv')

des=dff.describe()
print(des)
```

	Duration	Pulse	Maxpulse	Calories
count	169.000000	169.000000	169.000000	164.000000
mean	63.846154	107.461538	134.047337	375.790244
std	42.209949	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

```
[14] dff.isnull().values.any()

True

mean_v=dff.mean()

#Replace null values with the mean of the respective column
dff.fillna(mean_v,inplace=True)
#Display the DataFrame after replacing null values
print("\n dataframe after replacing ")
print(dff.head())
```

dataframe after replacing				
	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

```
[ ] dataframe after replacing
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

aggregation = dff.agg({
    'Pulse': ['min', 'max', 'count', 'mean'],
    'Calories': ['min', 'max', 'count', 'mean']
})
print("\n The Aggregation of Pulse and Calories are:")
print(aggregation)
```

	Pulse	Calories
min	80.000000	50.300000
max	159.000000	1860.400000
count	169.000000	164.000000
mean	107.461538	375.790244

```
[ ] filtered_dff_500_1000 = dff[(dff['Calories'] >= 500) & (dff['Calories'] <= 1000)]
print("\n The Rows with Calories between 500 and 1000:")
print(filtered_dff_500_1000)
```

The Rows with Calories between 500 and 1000:				
	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

```
The Rows with Calories between 500 and 1000:
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

filtered_df_calories_pulse = dff[(dff['Calories'] > 500) & (dff['Pulse'] < 100)]
print("\n Rows of calories > 500 and Rows of calories Pulse < 100:")
print(filtered_df_calories_pulse)
```

Rows of calories > 500 and Rows of calories Pulse < 100:				
	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

My video link: