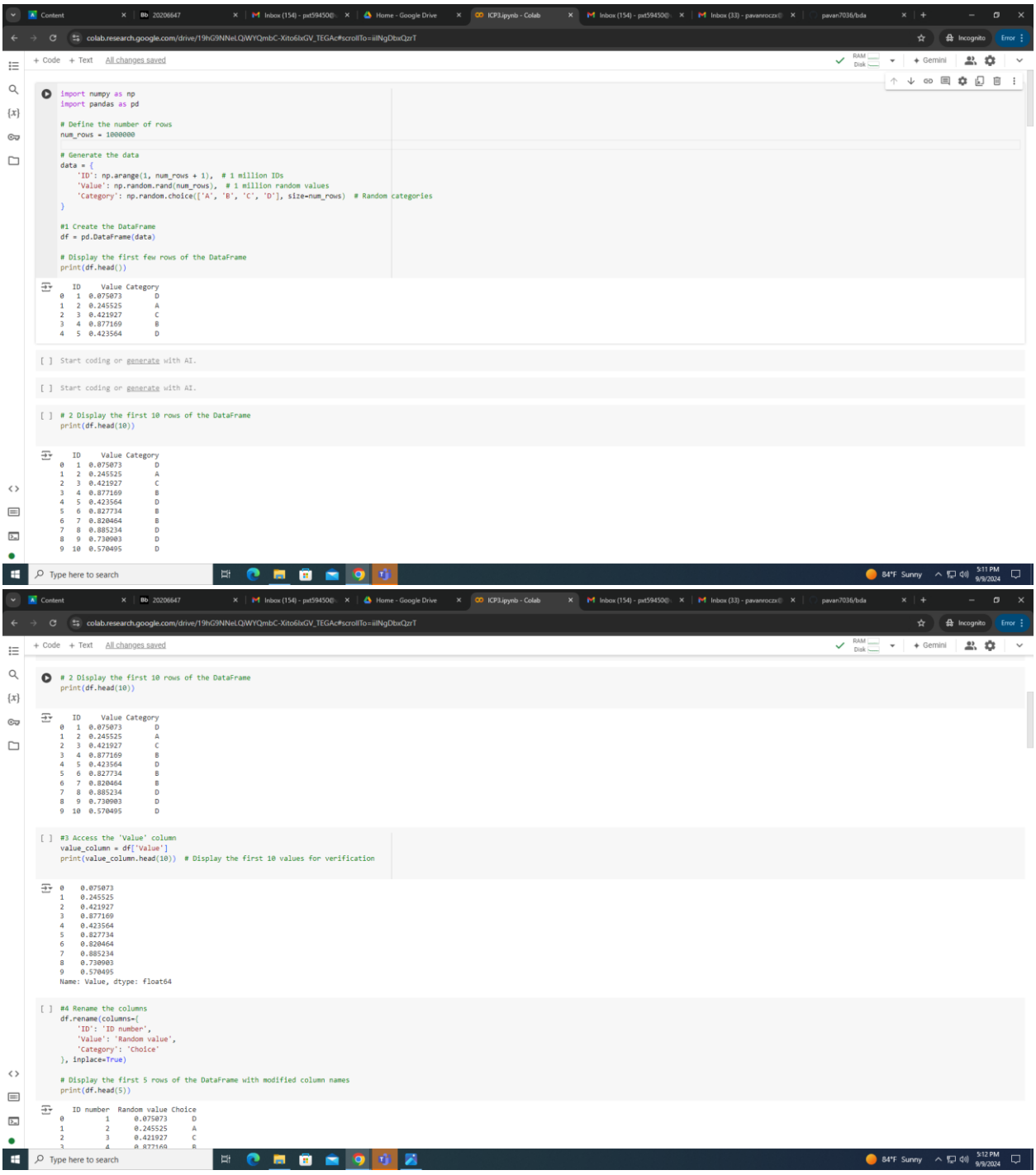
Icp3 assignment



```
import numpy as np
import pandas as pd

# Define the number of rows
num_rows = 1000000

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

#1 Create the DataFrame
df = pd.DataFrame(data)

# Display the first few rows of the DataFrame
print(df.head())
```

```
   ID     Value Category
0   1  0.075073        D
1   2  0.245525        A
2   3  0.421927        C
3   4  0.877169        B
4   5  0.423564        D
```

```
[ ] Start coding or generate with AI.
```

```
[ ] Start coding or generate with AI.
```

```
[ ] # 2 Display the first 10 rows of the DataFrame
    print(df.head(10))
```

```
   ID     Value Category
0   1  0.075073        D
1   2  0.245525        A
2   3  0.421927        C
3   4  0.877169        B
4   5  0.423564        D
5   6  0.827734        B
6   7  0.820464        B
7   8  0.885234        D
8   9  0.730903        D
9  10  0.570495        D
```

```
# 2 Display the first 10 rows of the DataFrame
print(df.head(10))
```

```
   ID     Value Category
0   1  0.075073        D
1   2  0.245525        A
2   3  0.421927        C
3   4  0.877169        B
4   5  0.423564        D
5   6  0.827734        B
6   7  0.820464        B
7   8  0.885234        D
8   9  0.730903        D
9  10  0.570495        D
```

```
[ ] #3 Access the 'Value' column
    value_column = df['Value']
    print(value_column.head(10))  # Display the first 10 values for verification
```

```
0    0.075073
1    0.245525
2    0.421927
3    0.877169
4    0.423564
5    0.827734
6    0.820464
7    0.885234
8    0.730903
9    0.570495
Name: Value, dtype: float64
```

```
[ ] #4 Rename the columns
    df.rename(columns={
        'ID': 'ID number',
        'Value': 'Random value',
        'Category': 'Choice'
    }, inplace=True)

    # Display the first 5 rows of the DataFrame with modified column names
    print(df.head(5))
```

```
   ID number  Random value Choice
0          1      0.075073      D
1          2      0.245525      A
2          3      0.421927      C
3          4      0.877169      B
```

```
9    0.570495
Name: Value, dtype: float64
```

```
#4 Rename the columns
df.rename(columns={
    'ID': 'ID number',
    'Value': 'Random value',
    'Category': 'Choice'
}, inplace=True)

# Display the first 5 rows of the DataFrame with modified column names
print(df.head(5))
```

```
   ID number  Random value Choice
0          1      0.075073      D
1          2      0.245525      A
2          3      0.421927      C
3          4      0.877169      B
4          5      0.423564      D
```

```
#5 import pandas as pd

# Set display options
pd.set_option('display.max_rows', None)

# Create DataFrame with corrected syntax
student_data = pd.DataFrame({
    'school_code': [S001, S002, S003, S004, S005, S006],
    '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, 13, 14, 12],
    'height': [173, 192, 186, 167, 151, 159],
    'weight': [35, 32, 33, 30, 31, 32],
    'address': ['Street1', 'Street2', 'Street3', 'Street4', 'Street5', 'Street6']
}, index=['S1', 'S2', 'S3', 'S4', 'S5', 'S6'])

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

# Split the data on school_code and class
result = student_data.groupby(['school_code', 'class'])

# Print the grouped data
print('\nSplit the data on school_code and class:')
for name, group in result:
    print("\nGroup:")
    print(f"Name: {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
```


```
3    4      0.877169      B
4    5      0.423564      D
```

```
#5 import pandas as pd

# Set display options
pd.set_option('display.max_rows', None)

# Create DataFrame with corrected syntax
student_data = pd.DataFrame({
    'school_code': [S001, S002, S003, S004, S005, S006],
    '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, 13, 14, 12],
    'height': [173, 192, 186, 167, 151, 159],
    'weight': [35, 32, 33, 30, 31, 32],
    'address': ['Street1', 'Street2', 'Street3', 'Street4', 'Street5', 'Street6']
}, index=['S1', 'S2', 'S3', 'S4', 'S5', 'S6'])

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

# Split the data on school_code and class
result = student_data.groupby(['school_code', 'class'])

# Print the grouped data
print('\nSplit the data on school_code and class:')
for name, group in result:
    print("\nGroup:")
    print(f"Name: {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         S004    VI    Eesha Hinton    25/09/1998   13     167      30
S5         S005     V    Gino Mcneill    11/05/2002   14     151      31
S6         S006    VI    David Parkes    15/09/1997   12     159      32

    address
S1  Street1
S2  Street2
S3  Street3
S4  Street4
S5  Street5
S6  Street6

Split the data on school_code and class:

Group:
Name: (S001, 'V')
    school_code class            name date_of_birth  age  height  weight  \
S1         S001     V  Alberto Franco    15/05/2002   12     173      35
```

```
Split the data on school_code and class:
```

```
Group:
Name: (5001, 'V')
    school_code class           name date_of_birth  age  height  weight  \
S1         5001     V  Alberto Franco    15/05/2002   12     173      35

    address
S1  Street1

Group:
Name: (5002, 'V')
    school_code class          name date_of_birth  age  height  weight  \
S2         5002     V  Gino Mcneill    17/05/2002   12     192      32

    address
S2  Street2

Group:
Name: (5003, 'VI')
    school_code class         name date_of_birth  age  height  weight  address
S3         5003    VI  Ryan Parkes    16/02/1999   13     186      33  Street3

Group:
Name: (5004, 'VI')
    school_code class          name date_of_birth  age  height  weight  \
S4         5004    VI  Eesha Hinton    25/09/1998   13     167      30

    address
S4  Street4

Group:
Name: (5005, 'V')
    school_code class          name date_of_birth  age  height  weight  \
S5         5005     V  Gino Mcneill    11/05/2002   14     151      31

    address
S5  Street5

Group:
Name: (5006, 'VI')
    school_code class          name date_of_birth  age  height  weight  \
S6         5006    VI  David Parkes    15/09/1997   12     159      32

    address
S6  Street6
```

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

df.head()
```

	Duration	Pulse	Maxpulse	Calories
0	60	110	130	409.1
1	60	117	145	479.0

+ Code + Text All changes saved
RAM Disk Gemini

Group:
Name: (5006, 'VI')
    school_code class   name date_of_birth age height weight \
S6   5006   VI David Parkes   15/09/1997   12   159   32

    address
S6 Street6

#6
file_path = '/content/data.csv'
df = pd.read_csv(file_path)

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

```
print("Basic Statistical Description:") # 7 answer
print(student_data.describe(include='all'))
```

```
Basic Statistical Description:
       school_code class          name date_of_birth        age      height  \
count     6.000000     6             6             6   6.000000    6.000000
unique         NaN     2             5             6        NaN         NaN
top            NaN     V  Gino Mcneill    15/05/2002        NaN         NaN
freq           NaN     3             2             1        NaN         NaN
mean    5003.500000   NaN           NaN           NaN  12.666667  171.333333
std        1.870829   NaN           NaN           NaN   0.816497   15.680136
min     5001.000000   NaN           NaN           NaN  12.000000  151.000000
25%     5002.250000   NaN           NaN           NaN  12.000000  161.000000
50%     5003.500000   NaN           NaN           NaN  12.500000  170.000000
75%     5004.750000   NaN           NaN           NaN  13.000000  182.750000
max     5006.000000   NaN           NaN           NaN  14.000000  192.000000

          weight  address
count   6.000000        6
unique       NaN        6
top          NaN  Street1
freq         NaN        1
mean   32.166667      NaN
std     1.722401      NaN
min    30.000000      NaN
25%    31.250000      NaN
50%    32.000000      NaN
75%    32.750000      NaN
max    35.000000      NaN
```

```
#8 Check for null values
```

```
max      35.000000      NaN
```

```
#8 Check for null values
print("\nCheck for null values:")
print(student_data.isnull().sum())

# Replace null values with the mean of the respective column
student_data.fillna(student_data.mean(numeric_only=True), inplace=True)
```

```
Check for null values:
school_code      0
class            0
name             0
date_of_birth    0
age              0
height           0
weight           0
address          0
dtype: int64
```

```
#9 Aggregate data
aggregation = student_data[['height', 'weight']].agg(['min', 'max', 'count', 'mean'])
print("\nAggregated Data:")
print(aggregation)
```

```
Aggregated Data:
            height     weight
min     151.000000  30.000000
max     192.000000  35.000000
count     6.000000   6.000000
mean    171.333333  32.166667
```

```
[1] import pandas as pd

    # 10 Sample DataFrame
    data = {
        'food': ['apple', 'banana', 'burger', 'pizza', 'salad'],
        'calories': [95, 105, 800, 1200, 150]
    }
    df = pd.DataFrame(data)

    # Filter rows where calories are between 500 and 1000
    filtered_df = df[(df['calories'] >= 500) & (df['calories'] <= 1000)]

    print(filtered_df)
```

```
     food  calories
2  burger       800
```

```
[2] import pandas as pd
```

+ Code + Text All changes saved RAM Disk Gemini

2 burger   800

```
[2] import pandas as pd

    # 11 Sample DataFrame
    data = {
        'food': ['apple', 'banana', 'burger', 'pizza', 'salad'],
        'calories': [95, 105, 800, 1200, 150],
        'pulse': [80, 90, 110, 95, 85]
    }
    df = pd.DataFrame(data)

    # Filter rows where calories > 500 and pulse < 100
    filtered_df = df[(df['calories'] > 500) & (df['pulse'] < 100)]

    print(filtered_df)
```

```
     food  calories  pulse
3  pizza      1200     95
```

```
import pandas as pd

#12 Sample DataFrame
data = {
    'food': ['apple', 'banana', 'burger', 'pizza', 'salad'],
    'calories': [95, 105, 800, 1200, 150],
    'Maxpulse': [80, 90, 110, 95, 85]
}
df = pd.DataFrame(data)

# Create a new DataFrame df_modified that excludes the "Maxpulse" column
df_modified = df.drop(columns=['Maxpulse'])

print(df_modified)
```

```
     food  calories
0   apple        95
1  banana       105
2  burger       800
3   pizza      1200
4   salad       150
```

```
[4] import pandas as pd

    #13 Sample DataFrame
    data = {
        'food': ['apple', 'banana', 'burger', 'pizza', 'salad'],
        'calories': [95, 105, 800, 1200, 150],
        'Maxpulse': [80, 90, 110, 95, 85]
    }
    df = pd.DataFrame(data)

    # Delete the "Maxpulse" column
    df = df.drop(columns=['Maxpulse'])
```

Gethub:https://github.com/pavan7036/bda