THIS TASK INVOLVES USING THE PANDAS LIBRARY TO MANIPULATE DATA AND TO PERFORM OPERATIONS LIKE FILTERING DATA BASED ON CONDITIONS, HANDLING MISSING VALUES, AND CALCULATING SUMMARY STATISTICS.

- *Reading CSV File*

```
# Importing pandas to perform data manipulation and analysis tasks
import pandas as pd


# Reading the CSV file
data = pd.read_csv('/content/01.Data Cleaning and Preprocessing.csv')
```

- *Initial Exploration:*

```
# Displaying the first few rows of the DataFrame
data.head()
```

	Observation	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	T-lowerExt-2
0	31-00:00	23.10	16.520	121.717	1177.607	169.805	358.282	329.545
1	31-01:00	27.60	16.810	79.022	1328.360	341.327	351.050	329.067
2	31-02:00	23.19	16.709	79.562	1329.407	239.161	350.022	329.260
3	31-03:00	23.60	16.478	81.011	1334.877	213.527	350.938	331.142
4	31-04:00	22.90	15.618	93.244	1334.168	243.131	351.640	332.709

5 rows × 23 columns

```
# Displaying the shape of the DataFrame:
data.shape
```

```
(324, 23)
```

```
# Displaying information about the DataFrame:
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 324 entries, 0 to 323
Data columns (total 23 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   Observation      324 non-null    object 
 1   Y-Kappa          324 non-null    float64
 2   ChipRate         319 non-null    float64
 3   BF-CMratio       307 non-null    float64
 4   BlowFlow         308 non-null    float64
 5   ChipLevel4       323 non-null    float64
 6   T-upperExt-2     322 non-null    float64
 7   T-lowerExt-2     322 non-null    float64
 8   UCZAA            299 non-null    float64
 9   WhiteFlow-4      323 non-null    float64
 10  AAWhiteSt-4      173 non-null    float64
 11  AA-Wood-4        323 non-null    float64
 12  ChipMoisture-4   323 non-null    float64
 13  SteamFlow-4      323 non-null    float64
 14  Lower-HeatT-3    322 non-null    float64
 15  Upper-HeatT-3    322 non-null    float64
 16  ChipMass-4       323 non-null    float64
 17  WeakLiquorF      323 non-null    float64
 18  BlackFlow-2      322 non-null    float64
 19  WeakWashF        323 non-null    float64
 20  SteamHeatF-3     322 non-null    float64
 21  T-Top-Chips-4    323 non-null    float64
 22  SulphidityL-4    173 non-null    float64
dtypes: float64(22), object(1)
memory usage: 58.3+ KB
```

```
# Displaying data types of columns:
data.dtypes
```

```
Observation         object
Y-Kappa            float64
ChipRate           float64
BF-CMratio         float64
```

```
BlowFlow          float64
ChipLevel4        float64
T-upperExt-2      float64
T-lowerExt-2      float64
UCZAA             float64
WhiteFlow-4       float64
AAWhiteSt-4       float64
AA-Wood-4         float64
ChipMoisture-4    float64
SteamFlow-4       float64
Lower-HeatT-3     float64
Upper-HeatT-3     float64
ChipMass-4        float64
WeakLiquorF       float64
BlackFlow-2       float64
WeakWashF         float64
SteamHeatF-3      float64
T-Top-Chips-4     float64
SulphidityL-4     float64
dtype: object
```

```
# Displaying column names:
data.columns
```

```
Index(['Observation', 'Y-Kappa', 'ChipRate', 'BF-CMratio', 'BlowFlow',
       'ChipLevel4 ', 'T-upperExt-2 ', 'T-lowerExt-2  ', 'UCZAA',
       'WhiteFlow-4 ', 'AAWhiteSt-4 ', 'AA-Wood-4  ', 'ChipMoisture-4 ',
       'SteamFlow-4 ', 'Lower-HeatT-3', 'Upper-HeatT-3 ', 'ChipMass-4 ',
       'WeakLiquorF ', 'BlackFlow-2 ', 'WeakWashF ', 'SteamHeatF-3 ',
       'T-Top-Chips-4 ', 'SulphidityL-4 '],
      dtype='object')
```

```
# Displaying summary statistics of the DataFrame:
data.describe()
```

	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	lowerE
count	324.000000	319.000000	307.000000	308.000000	323.000000	322.000000	322.000
mean	20.635370	14.347937	87.464456	1237.837614	258.164483	356.904295	324.020
std	3.070036	1.499095	7.995012	100.593735	87.987452	9.209290	7.621
min	12.170000	9.983000	68.645000	0.000000	0.000000	339.168000	284.633
25%	18.382500	13.358000	81.823000	1193.215250	213.527000	350.241250	321.420
50%	20.845000	14.308000	86.739000	1273.138500	271.792000	356.843000	325.669
75%	23.032500	15.517000	92.372000	1289.196000	321.680000	362.242250	329.175
max	27.600000	16.958000	121.717000	1351.240000	419.014000	399.135000	337.012

8 rows × 22 columns

- *Handling Duplicate Rows:*

```
# Checking for duplicated rows:
data.duplicated()
```

```
0      False
1      False
2      False
3      False
4      False
       ...
319     True
320     True
321     True
322     True
323     True
Length: 324, dtype: bool
```

```
# Sorting the DataFrame by a specific column:
data.sort_values(by='BF-CMratio', ascending=True)
```

	Observation	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	T lowerExt
31	1-06:00	16.10	16.292	68.645	1094.883	245.962	344.478	321.06
112	4-15:00	18.80	16.455	71.679	1191.663	195.725	359.473	326.50
30	1-05:00	24.15	14.533	72.015	1173.241	267.057	347.734	317.65
63	2-14:00	24.70	16.183	72.611	1231.362	332.497	350.938	322.80
15	31-14:00	25.40	16.425	72.924	1197.775	118.821	350.765	329.79
...	...	...	...	...	...	...	...	.
207	8-14:00	16.05	13.114	NaN	NaN	0.000	NaN	Na
208	8-15:00	17.75	11.283	NaN	NaN	41.607	362.545	331.24
314	8-13:00	16.70	NaN	NaN	NaN	0.000	373.867	324.18
315	8-14:00	16.05	13.114	NaN	NaN	0.000	NaN	Na
316	8-15:00	17.75	11.283	NaN	NaN	41.607	362.545	331.24

324 rows × 23 columns

```
# Counting non-null values in each column:
data.notnull().sum()
```

```
Observation        324
Y-Kappa            324
ChipRate           319
BF-CMratio         307
BlowFlow           308
ChipLevel4         323
T-upperExt-2       322
T-lowerExt-2       322
UCZAA              299
WhiteFlow-4        323
AAWhiteSt-4        173
AA-Wood-4          323
ChipMoisture-4     323
SteamFlow-4        323
Lower-HeatT-3      322
Upper-HeatT-3      322
ChipMass-4         323
WeakLiquorF        323
BlackFlow-2        322
WeakWashF          323
SteamHeatF-3       322
T-Top-Chips-4      323
SulphidityL-4      173
dtype: int64
```

```
# Checking for null values in the DataFrame:
data.isnull()
```

	Observation	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	T lowerExt
0	False	False	False	False	False	False	False	Fals
1	False	False	False	False	False	False	False	Fals
2	False	False	False	False	False	False	False	Fals
3	False	False	False	False	False	False	False	Fals
4	False	False	False	False	False	False	False	Fals
...	...	...	...	...	...	...	...	.
319	False	False	False	False	False	False	False	Fals
320	False	False	False	False	False	False	False	Fals
321	False	False	False	False	False	False	False	Fals
322	False	False	False	False	False	False	False	Fals
323	False	False	False	False	False	False	False	Fals

324 rows × 23 columns

```
# Counting null values in each column:
data.isnull().sum()
```

```
Observation          0
Y-Kappa              0
ChipRate             5
BF-CMratio          17
BlowFlow            16
ChipLevel4           1
T-upperExt-2         2
T-lowerExt-2         2
UCZAA               25
WhiteFlow-4          1
AAWhiteSt-4        151
AA-Wood-4            1
ChipMoisture-4       1
SteamFlow-4          1
Lower-HeatT-3        2
Upper-HeatT-3        2
ChipMass-4           1
WeakLiquorF          1
BlackFlow-2          2
WeakWashF            1
SteamHeatF-3         2
T-Top-Chips-4        1
SulphidityL-4      151
dtype: int64
```

```
# Counting total null values in the DataFrame:
data.isnull().sum().sum()
```

```
386
```

- *Handling Missing Values*

```
# Filling null values with 0:
data1 = data.fillna(value=0)
data1
```

	Observation	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	T lowerExt
0	31-00:00	23.10	16.520	121.717	1177.607	169.805	358.282	329.54
1	31-01:00	27.60	16.810	79.022	1328.360	341.327	351.050	329.06
2	31-02:00	23.19	16.709	79.562	1329.407	239.161	350.022	329.26
3	31-03:00	23.60	16.478	81.011	1334.877	213.527	350.938	331.14
4	31-04:00	22.90	15.618	93.244	1334.168	243.131	351.640	332.70
...	...	...	...	...	...	...	...	
319	10-16:00	23.75	12.667	93.450	1178.252	276.955	347.286	310.97
320	9-19:00	19.80	12.558	94.352	1184.119	297.071	399.135	319.57
321	9-20:00	23.01	12.550	90.842	1188.517	289.826	373.633	314.59
322	9-21:00	24.32	13.083	88.910	1192.879	318.006	364.081	308.55
323	9-22:00	25.75	13.417	85.451	1186.342	248.312	356.289	310.48

324 rows × 23 columns

```
# Counting null values in each column after filling null values with 0
data1.isnull().sum()
```

```
Observation        0
Y-Kappa            0
ChipRate           0
BF-CMratio         0
BlowFlow           0
ChipLevel4         0
T-upperExt-2       0
T-lowerExt-2       0
UCZAA              0
WhiteFlow-4        0
AAWhiteSt-4        0
AA-Wood-4          0
ChipMoisture-4     0
SteamFlow-4        0
Lower-HeatT-3      0
Upper-HeatT-3      0
```

```
ChipMass-4         0
WeakLiquorF        0
BlackFlow-2        0
WeakWashF          0
SteamHeatF-3       0
T-Top-Chips-4      0
SulphidityL-4      0
dtype: int64
```

```
# Removing duplicate rows from the DataFrame:
data2 = data.drop_duplicates()
data2
```

	Observation	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	T lowerExt
0	31-00:00	23.10	16.520	121.717	1177.607	169.805	358.282	329.54
1	31-01:00	27.60	16.810	79.022	1328.360	341.327	351.050	329.06
2	31-02:00	23.19	16.709	79.562	1329.407	239.161	350.022	329.26
3	31-03:00	23.60	16.478	81.011	1334.877	213.527	350.938	331.14
4	31-04:00	22.90	15.618	93.244	1334.168	243.131	351.640	332.70
...	...	...	...	...	...	...	...	
298	12-09:00	20.90	15.167	84.640	1283.706	339.440	354.803	311.04
299	12-10:00	24.98	NaN	85.034	1278.345	368.564	357.723	321.38
300	12-11:00	21.00	NaN	88.013	1307.722	278.842	357.438	323.75
301	12-12:00	21.40	NaN	85.490	1255.986	273.484	361.365	322.68
307	31-05:00	20.89	14.308	94.172	1327.832	251.120	351.263	332.48

301 rows × 23 columns

```
# Remove non-numeric columns
numeric_data = data.select_dtypes(include=['number'])
```

```
# Fill null values with the mean of each column
data3 = numeric_data.fillna(numeric_data.mean())
data3
```

	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	T-lowerExt-2	UCZAA	Wl
0	23.10	16.520	121.717	1177.607	169.805	358.282	329.545	1.44300	
1	27.60	16.810	79.022	1328.360	341.327	351.050	329.067	1.54900	
2	23.19	16.709	79.562	1329.407	239.161	350.022	329.260	1.60000	
3	23.60	16.478	81.011	1334.877	213.527	350.938	331.142	1.60400	
4	22.90	15.618	93.244	1334.168	243.131	351.640	332.709	1.49201	
...	...	...	...	...	...	...	...	...	
319	23.75	12.667	93.450	1178.252	276.955	347.286	310.970	1.52300	
320	19.80	12.558	94.352	1184.119	297.071	399.135	319.576	1.45100	
321	23.01	12.550	90.842	1188.517	289.826	373.633	314.591	1.45700	
322	24.32	13.083	88.910	1192.879	318.006	364.081	308.559	1.52300	
323	25.75	13.417	85.451	1186.342	248.312	356.289	310.482	1.47400	

324 rows × 22 columns

```
# Filling null values using backward fill (bfill):
data4 = data.fillna(method = 'bfill')
data4
```

	Observation	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	T lowerExt
0	31-00:00	23.10	16.520	121.717	1177.607	169.805	358.282	329.54
1	31-01:00	27.60	16.810	79.022	1328.360	341.327	351.050	329.06
2	31-02:00	23.19	16.709	79.562	1329.407	239.161	350.022	329.26
3	31-03:00	23.60	16.478	81.011	1334.877	213.527	350.938	331.14
4	31-04:00	22.90	15.618	93.244	1334.168	243.131	351.640	332.70
...	...	...	...	...	...	...	...	.
319	10-16:00	23.75	12.667	93.450	1178.252	276.955	347.286	310.97
320	9-19:00	19.80	12.558	94.352	1184.119	297.071	399.135	319.57
321	9-20:00	23.01	12.550	90.842	1188.517	289.826	373.633	314.59
322	9-21:00	24.32	13.083	88.910	1192.879	318.006	364.081	308.55
323	9-22:00	25.75	13.417	85.451	1186.342	248.312	356.289	310.48

324 rows × 23 columns

```
# Filling null values using forward fill (pad):
data5 = data.fillna(method = 'pad')
data5
```

	Observation	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	T lowerExt
0	31-00:00	23.10	16.520	121.717	1177.607	169.805	358.282	329.54
1	31-01:00	27.60	16.810	79.022	1328.360	341.327	351.050	329.06
2	31-02:00	23.19	16.709	79.562	1329.407	239.161	350.022	329.26
3	31-03:00	23.60	16.478	81.011	1334.877	213.527	350.938	331.14
4	31-04:00	22.90	15.618	93.244	1334.168	243.131	351.640	332.70
...	...	...	...	...	...	...	...	.
319	10-16:00	23.75	12.667	93.450	1178.252	276.955	347.286	310.97
320	9-19:00	19.80	12.558	94.352	1184.119	297.071	399.135	319.57
321	9-20:00	23.01	12.550	90.842	1188.517	289.826	373.633	314.59
322	9-21:00	24.32	13.083	88.910	1192.879	318.006	364.081	308.55
323	9-22:00	25.75	13.417	85.451	1186.342	248.312	356.289	310.48

324 rows × 23 columns

- *Outlier Detection and Removal:*

```
import numpy as np
```

```
data1.columns
```

```
Index(['Observation', 'Y-Kappa', 'ChipRate', 'BF-CMratio', 'BlowFlow',
       'ChipLevel4 ', 'T-upperExt-2 ', 'T-lowerExt-2  ', 'UCZAA',
       'WhiteFlow-4 ', 'AAWhiteSt-4 ', 'AA-Wood-4  ', 'ChipMoisture-4 ',
       'SteamFlow-4 ', 'Lower-HeatT-3', 'Upper-HeatT-3 ', 'ChipMass-4 ',
       'WeakLiquorF ', 'BlackFlow-2 ', 'WeakWashF ', 'SteamHeatF-3 ',
       'T-Top-Chips-4 ', 'SulphidityL-4 '],
      dtype='object')
```

```
# Dropping the 'Observation' column:
data1.drop(['Observation'],axis = 1, inplace = True)
data1.columns
```

```
Index(['Y-Kappa', 'ChipRate', 'BF-CMratio', 'BlowFlow', 'ChipLevel4 ',
       'T-upperExt-2 ', 'T-lowerExt-2  ', 'UCZAA', 'WhiteFlow-4 ',
       'AAWhiteSt-4 ', 'AA-Wood-4  ', 'ChipMoisture-4 ', 'SteamFlow-4 ',
       'Lower-HeatT-3', 'Upper-HeatT-3 ', 'ChipMass-4 ', 'WeakLiquorF ',
       'BlackFlow-2 ', 'WeakWashF ', 'SteamHeatF-3 ', 'T-Top-Chips-4 ',
       'SulphidityL-4 '],
      dtype='object')
```

```
# Calculate the first quartile (Q1)
Quartile1 = data1.quantile(0.25)

# Calculate the third quartile (Q3)
Quartile3 = data1.quantile(0.75)

# Calculate the interquartile range (IQR)
iqr = Quartile3 - Quartile1
print(iqr)
```

```
Y-Kappa              4.65000
ChipRate             2.25625
BF-CMratio          11.11225
BlowFlow            98.43375
ChipLevel4         107.92275
T-upperExt-2        11.96500
T-lowerExt-2         7.82875
UCZAA                0.13925
WhiteFlow-4         98.59525
AAWhiteSt-4          6.14000
AA-Wood-4            1.45900
ChipMoisture-4       2.22000
SteamFlow-4          9.04675
Lower-HeatT-3        8.46750
Upper-HeatT-3        7.77050
ChipMass-4          19.70375
WeakLiquorF        174.05550
BlackFlow-2        276.51675
WeakWashF          271.44325
SteamHeatF-3         6.94975
T-Top-Chips-4        2.01025
SulphidityL-4       30.40250
dtype: float64
```

```
# Determine the lower and upper bounds for outliers
lower_bound = Quartile1 - 1.5 * iqr
upper_bound = Quartile3 + 1.5 * iqr

# Filter outliers
data1 = data1[~((data1 < lower_bound) | (data1 > upper_bound)).any(axis=1)]

data1
```

	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	T-lowerExt-2	UCZAA	Whi
1	27.60	16.810	79.022	1328.360	341.327	351.050	329.067	1.549	
2	23.19	16.709	79.562	1329.407	239.161	350.022	329.260	1.600	
3	23.60	16.478	81.011	1334.877	213.527	350.938	331.142	1.604	
5	14.23	15.350	85.518	1171.604	198.538	344.014	325.195	1.436	
6	13.49	13.700	98.186	1243.688	116.275	346.208	326.982	1.434	
...	...	...	...	...	...	...	...	...	
317	17.80	16.625	78.367	1276.082	202.744	360.127	329.266	1.488	
318	18.20	16.283	83.508	1288.104	234.284	359.412	328.670	1.534	
319	23.75	12.667	93.450	1178.252	276.955	347.286	310.970	1.523	
321	23.01	12.550	90.842	1188.517	289.826	373.633	314.591	1.457	
323	25.75	13.417	85.451	1186.342	248.312	356.289	310.482	1.474	

241 rows × 22 columns

- *Standardizing the Data.*

```
import sklearn
from sklearn import preprocessing
from sklearn.preprocessing import scale


data1.describe()
```

	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	lowerE
count	241.000000	241.000000	241.000000	241.000000	241.000000	241.000000	241.000
mean	20.766349	14.678552	86.089842	1256.920801	265.153842	356.569282	325.276
std	3.098619	1.315662	7.186591	49.689544	74.200658	7.519533	5.676
min	12.480000	10.833000	68.645000	1084.083000	61.783000	340.222000	310.421
25%	18.400000	13.850000	81.011000	1220.750000	220.164000	350.317000	322.272
50%	20.900000	14.700000	85.006000	1280.487000	271.419000	357.163000	326.625
75%	23.190000	15.717000	91.341000	1289.992000	323.560000	361.202000	329.266
max	27.600000	16.958000	108.493000	1351.240000	402.245000	375.047000	337.012

8 rows × 22 columns

```
# Scaling data:
data1.matrix = data1.values.reshape(-1,1)

store= preprocessing.MinMaxScaler(feature_range=(0,10))
stored=store.fit_transform(data1)

data1
```

```
<ipython-input-204-11fc09716ecc>:2: UserWarning: Pandas doesn't allow columns to be c
  data1.matrix = data1.values.reshape(-1,1)
```

	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	T-lowerExt-2	UCZAA	Whi
1	27.60	16.810	79.022	1328.360	341.327	351.050	329.067	1.549	
2	23.19	16.709	79.562	1329.407	239.161	350.022	329.260	1.600	
3	23.60	16.478	81.011	1334.877	213.527	350.938	331.142	1.604	
5	14.23	15.350	85.518	1171.604	198.538	344.014	325.195	1.436	
6	13.49	13.700	98.186	1243.688	116.275	346.208	326.982	1.434	
...	...	...	...	...	...	...	...	...	
317	17.80	16.625	78.367	1276.082	202.744	360.127	329.266	1.488	
318	18.20	16.283	83.508	1288.104	234.284	359.412	328.670	1.534	
319	23.75	12.667	93.450	1178.252	276.955	347.286	310.970	1.523	
321	23.01	12.550	90.842	1188.517	289.826	373.633	314.591	1.457	
323	25.75	13.417	85.451	1186.342	248.312	356.289	310.482	1.474	

241 rows × 22 columns

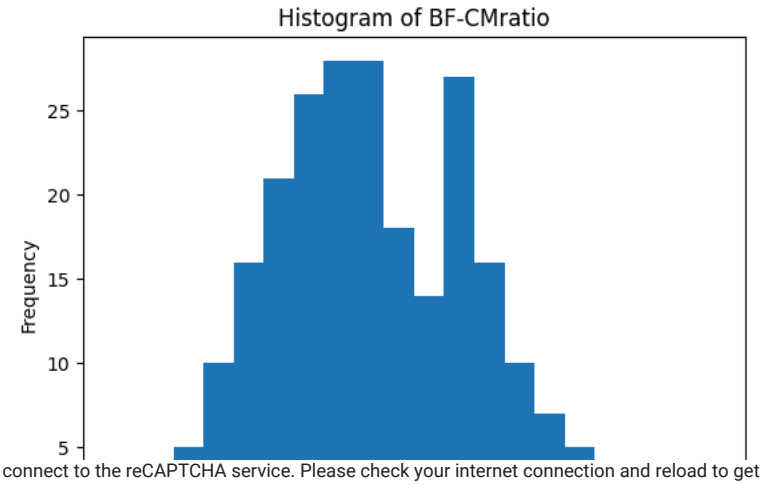
- *Writing to CSV Files*

```
data1.to_csv('output_task2.csv', index=False)
```

- *Data Visualization*

```
# Visual exploration of data distribution
import matplotlib.pyplot as plt

plt.hist(data1['BF-CMratio'], bins=20)
plt.title('Histogram of BF-CMratio')
plt.xlabel('Values')
plt.ylabel('Frequency')
plt.show()
```



Could not connect to the reCAPTCHA service. Please check your internet connection and reload to get a reCAPTCHA challenge.