Import pandas library

```
import pandas as pd
```

Load Csv file

```
df=pd.read_csv("/content/01.Data Cleaning and Preprocessing.csv")
```

Dataset has 324 columns with different null valuese

```
#summary of data
df.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
```

```
df.describe()
```

	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	T-lowerExt-2
count	324.000000	319.000000	307.000000	308.000000	323.000000	322.000000	322.000000
mean	20.635370	14.347937	87.464456	1237.837614	258.164483	356.904295	324.020180
std	3.070036	1.499095	7.995012	100.593735	87.987452	9.209290	7.621402
min	12.170000	9.983000	68.645000	0.000000	0.000000	339.168000	284.633000
25%	18.382500	13.358000	81.823000	1193.215250	213.527000	350.241250	321.420000
50%	20.845000	14.308000	86.739000	1273.138500	271.792000	356.843000	325.669000
75%	23.032500	15.517000	92.372000	1289.196000	321.680000	362.242250	329.175000
max	27.600000	16.958000	121.717000	1351.240000	419.014000	399.135000	337.012000

8 rows × 22 columns

Data cleaning

```
#delete all duplicates
df=df.drop_duplicates()
df
```

	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
...	...	...	...	...	...	...	...	...
298	12-09:00	20.90	15.167	84.640	1283.706	339.440	354.803	311.041
299	12-10:00	24.98	NaN	85.034	1278.345	368.564	357.723	321.387
300	12-11:00	21.00	NaN	88.013	1307.722	278.842	357.438	323.757
301	12-12:00	21.40	NaN	85.490	1255.986	273.484	361.365	322.689
307	31-05:00	20.89	14.308	94.172	1327.832	251.120	351.263	332.485

301 rows × 23 columns

```
#check whether null values exists or not
#returns true if null else False
df.isnull()
```

	Observation	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	T-lowerExt-2
0	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...
298	False	False	False	False	False	False	False	False
299	False	False	False	False	False	False	False	False
300	False	False	False	False	False	False	False	False
301	False	False	False	False	False	False	False	False
307	False	False	False	False	False	False	False	False

301 rows × 23 columns

```
#Find no. of null values in each column
df.isnull().sum()
#Total null values in entire dataset
df.isnull().sum().sum()
```

```
352
```

Ways to fill null values 1)Any value 2)ffill 3)bfill

```
#1)filling null values with any value
df1=df.fillna(value='Nan')
df1
```

	Observation	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	T-lowerExt-2
0	31-00:00	23.10	16.52	121.717	1177.607	169.805	358.282	329.545
1	31-01:00	27.60	16.81	79.022	1328.36	341.327	351.05	329.067
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	331.142
4	31-04:00	22.90	15.618	93.244	1334.168	243.131	351.64	332.709
...	...	...	...	...	...	...	...	...
298	12-09:00	20.90	15.167	84.64	1283.706	339.44	354.803	311.041
299	12-10:00	24.98	Nan	85.034	1278.345	368.564	357.723	321.387
300	12-11:00	21.00	Nan	88.013	1307.722	278.842	357.438	323.757
301	12-12:00	21.40	Nan	85.49	1255.986	273.484	361.365	322.689
307	31-05:00	20.89	14.308	94.172	1327.832	251.12	351.263	332.485

301 rows × 23 columns

```
df1.isnull().sum().sum()
```

```
0
```

```
#2)Filling null values with forward fill
df2=df.fillna(method="ffill")
df2
```

	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
...	...	...	...	...	...	...	...	...
298	12-09:00	20.90	15.167	84.640	1283.706	339.440	354.803	311.041
299	12-10:00	24.98	15.167	85.034	1278.345	368.564	357.723	321.387
300	12-11:00	21.00	15.167	88.013	1307.722	278.842	357.438	323.757
301	12-12:00	21.40	15.167	85.490	1255.986	273.484	361.365	322.689
307	31-05:00	20.89	14.308	94.172	1327.832	251.120	351.263	332.485

301 rows × 23 columns

```
#3)filling null values with backward fill
df3=df.fillna(method="bfill")
df3
```

	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
...	...	...	...	...	...	...	...	...
298	12-09:00	20.90	15.167	84.640	1283.706	339.440	354.803	311.041
299	12-10:00	24.98	14.308	85.034	1278.345	368.564	357.723	321.387
300	12-11:00	21.00	14.308	88.013	1307.722	278.842	357.438	323.757
301	12-12:00	21.40	14.308	85.490	1255.986	273.484	361.365	322.689
307	31-05:00	20.89	14.308	94.172	1327.832	251.120	351.263	332.485

301 rows × 23 columns

Preprocessing

```
#import necessary libraries
import numpy as np
import matplotlib.pyplot as plt
import scipy as sp
```

```
df2.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')
```

```
#drop observation column
df2.drop(['Observation'],axis=1,inplace=True)
df2.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')
```

```
#Find quantiles of dataframe to remove outliers
Q1=df2.quantile(00.25)
Q3=df2.quantile(0.75)
IQR=Q3-Q1
IQR
```

```
Y-Kappa              4.55000
ChipRate             2.12500
BF-CMratio          10.13800
BlowFlow            96.76600
ChipLevel4         105.86800
T-upperExt-2        12.05500
T-lowerExt-2         7.58900
UCZAA                0.13900
WhiteFlow-4        100.09800
AAWhiteSt-4          0.10550
AA-Wood-4            1.47800
ChipMoisture-4       2.18600
SteamFlow-4          8.84000
Lower-HeatT-3        8.57600
Upper-HeatT-3        7.78500
```

```
ChipMass-4          19.34700
WeakLiquorF        180.61300
BlackFlow-2        280.07500
WeakWashF          263.33500
SteamHeatF-3         6.90100
T-Top-Chips-4        2.02900
SulphidityL-4        0.69525
dtype: float64
```

```
df2=df2[~((df2<(Q1-1.5*IQR))|(df2>(Q3+1.5*IQR))).any(axis=1)]
df2
```

	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	T-lowerExt-2	UCZAA	WhiteFlow-4	AAWhiteSt-4	...	SteamFlow-4	Lower-HeatT-3	Upper HeatT
1	27.60	16.810	79.022	1328.360	341.327	351.050	329.067	1.549	537.201	6.076	...	60.012	330.823	304.87
2	23.19	16.709	79.562	1329.407	239.161	350.022	329.260	1.600	549.611	6.076	...	61.304	329.140	303.38
5	14.23	15.350	85.518	1171.604	198.538	344.014	325.195	1.436	628.245	6.020	...	65.225	322.103	298.51
6	13.49	13.700	98.186	1243.688	116.275	346.208	326.982	1.434	696.766	6.020	...	72.989	322.982	296.08
7	22.65	14.100	91.887	1307.852	288.989	352.321	331.162	1.468	625.549	6.143	...	71.298	329.662	301.53
...	...	...	...	...	...	...	...	...	...	...	...	...	...	.
298	20.90	15.167	84.640	1283.706	339.440	354.803	311.041	1.635	532.419	6.340	...	65.561	332.924	307.62
299	24.98	15.167	85.034	1278.345	368.564	357.723	321.387	1.635	520.365	6.220	...	65.729	332.523	307.16
300	21.00	15.167	88.013	1307.722	278.842	357.438	323.757	1.635	553.070	6.220	...	65.795	331.263	306.40
301	21.40	15.167	85.490	1255.986	273.484	361.365	322.689	1.635	590.199	6.230	...	71.456	333.032	308.73
307	20.89	14.308	94.172	1327.832	251.120	351.263	332.485	1.522	631.514	6.230	...	71.286	328.699	300.70

235 rows × 22 columns

Now dataset is cleaned completely

```
#Description of data is
df.describe()
```

	Y-Kappa	ChipRate	BF-CMratio	BlowFlow	ChipLevel4	T-upperExt-2	T-lowerExt-2	UCZAA	WhiteFlow-4	AAWhiteSt-4	...	SteamFl
count	301.000000	297.000000	287.000000	288.000000	300.000000	300.000000	300.000000	277.000000	300.000000	160.000000	...	300.000
mean	20.568605	14.338670	87.271564	1235.537278	259.365993	356.907033	324.010783	1.490588	593.148090	6.143012	...	66.834
std	2.990751	1.490121	7.839690	102.023065	85.643939	8.954457	7.650502	0.108138	66.949732	0.082396	...	5.643
min	12.170000	9.983000	68.645000	0.000000	0.000000	339.168000	284.633000	1.182000	405.111000	5.890000	...	48.568
25%	18.450000	13.358000	81.753500	1194.047250	215.369250	350.317000	321.534000	1.429000	543.137750	6.093000	...	62.847
50%	20.740000	14.417000	86.705000	1267.130000	271.845500	357.178500	325.638500	1.498000	594.294000	6.140000	...	67.751
75%	23.000000	15.492000	92.153500	1288.390000	321.285000	362.326250	329.129250	1.561000	643.567000	6.200000	...	71.650
max	27.600000	16.958000	121.717000	1351.240000	419.014000	399.135000	337.012000	1.747000	731.394000	6.340000	...	76.147

8 rows × 22 columns

Start coding or generate with AI.