```
import pandas as pd
df=pd.read_csv('household_power_consumption.csv')
df.head()
```

	Date	Time	Global_active_power	Global_reactive_power	Voltage	Global_ir
0	16/12/2006	17:24:00	4.216	0.418	234.84	
1	16/12/2006	17:25:00	5.36	0.436	233.63	
2	16/12/2006	17:26:00	5.374	0.498	233.29	
3	16/12/2006	17:27:00	5.388	0.502	233.74	
4	16/12/2006	17:28:00	3.666	0.528	235.68	

```
df.dtypes
```

```
Date                      object
Time                      object
Global_active_power       object
Global_reactive_power     object
Voltage                   object
Global_intensity          object
Sub_metering_1            object
Sub_metering_2            object
Sub_metering_3           float64
dtype: object
```

```
df.shape #(rows,columns)
```

```
(1048575, 9)
```

```
df['Sub_metering_3'].astype('object')
```

```
0          17.0
1          16.0
2          17.0
3          17.0
4          17.0
           ...
1048570     0.0
1048571     0.0
1048572     0.0
1048573     0.0
1048574     0.0
Name: Sub_metering_3, Length: 1048575, dtype: object
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1048575 entries, 0 to 1048574
Data columns (total 9 columns):
 #   Column                 Non-Null Count    Dtype
---  ------                 --------------    -----
 0   Date                   1048575 non-null  object
 1   Time                   1048575 non-null  object
 2   Global_active_power    1048575 non-null  object
 3   Global_reactive_power  1048575 non-null  object
 4   Voltage                1048575 non-null  object
 5   Global_intensity       1048575 non-null  object
 6   Sub_metering_1         1048575 non-null  object
 7   Sub_metering_2         1048575 non-null  object
 8   Sub_metering_3         1044506 non-null  float64
dtypes: float64(1), object(8)
memory usage: 72.0+ MB
```

```
df.dtypes
```

```
Date                      object
Time                      object
Global_active_power       object
Global_reactive_power     object
Voltage                   object
Global_intensity          object
Sub_metering_1            object
Sub_metering_2            object
Sub_metering_3           float64
dtype: object
```

```
df.values
```

```
array([['16/12/2006', '17:24:00', '4.216', ..., '0', '1', 17.0],
       ['16/12/2006', '17:25:00', '5.36', ..., '0', '1', 16.0],
       ['16/12/2006', '17:26:00', '5.374', ..., '0', '2', 17.0],
       ...,
       ['13/12/2008', '21:36:00', '0.422', ..., '0', '0', 0.0],
       ['13/12/2008', '21:37:00', '0.422', ..., '0', '0', 0.0],
       ['13/12/2008', '21:38:00', '0.422', ..., '0', '0', 0.0]],
      dtype=object)
```

```
df1=df.sort_values('Global_active_power',ascending=True)
df1.head(1000)
```

	Date	Time	Global_active_power	Global_reactive_power	Voltage	Globa
894447	28/8/2008	20:51:00	0.076	0	234.03	
894448	28/8/2008	20:52:00	0.076	0	233.92	
894446	28/8/2008	20:50:00	0.076	0	234.06	
894445	28/8/2008	20:49:00	0.076	0	234.34	
894444	28/8/2008	20:48:00	0.076	0	234.88	
...	...	...	...	...	...	
883702	21/8/2008	9:46:00	0.08	0	240.71	
867366	10/8/2008	1:30:00	0.08	0	241.05	
867367	10/8/2008	1:31:00	0.08	0	241.01	
884885	22/8/2008	5:29:00	0.08	0	239.63	
884886	22/8/2008	5:30:00	0.08	0	239.47	

1000 rows × 9 columns

```
df1=df.sort_values('Global_active_power',ascending=False)#descending order
df1.head(4000)
```

	Date	Time	Global_active_power	Global_reactive_power	Voltage	Globa
191263	28/4/2007	13:07:00	?	?	?	
193623	30/4/2007	4:27:00	?	?	?	
193619	30/4/2007	4:23:00	?	?	?	
193620	30/4/2007	4:24:00	?	?	?	
191975	29/4/2007	0:59:00	?	?	?	
...	...	...	...	...	...	
190742	28/4/2007	4:26:00	?	?	?	
190499	28/4/2007	0:23:00	?	?	?	
190498	28/4/2007	0:22:00	?	?	?	
191202	28/4/2007	12:06:00	?	?	?	
190497	28/4/2007	0:21:00	?	?	?	

4000 rows × 9 columns

```
#sort acc. to index in dec order as asc =false
df.sort_index(ascending=False)
```

	Date	Time	Global_active_power	Global_reactive_power	Voltage	Glo
1048574	13/12/2008	21:38:00	0.422	0.078	242.61	
1048573	13/12/2008	21:37:00	0.422	0.078	242.56	
1048572	13/12/2008	21:36:00	0.422	0.076	241.73	
1048571	13/12/2008	21:35:00	0.424	0.076	242.1	
1048570	13/12/2008	21:34:00	0.426	0.076	242.27	
...	...	...	...	...	...	
4	16/12/2006	17:28:00	3.666	0.528	235.68	
3	16/12/2006	17:27:00	5.388	0.502	233.74	
2	16/12/2006	17:26:00	5.374	0.498	233.29	
1	16/12/2006	17:25:00	5.36	0.436	233.63	
0	16/12/2006	17:24:00	4.216	0.418	234.84	

1048575 rows × 9 columns

```
df1=df.drop(columns=['Global_active_power'],axis=1)
df1.head()
```

	Date	Time	Global_reactive_power	Voltage	Global_intensity	Sub_metering
0	16/12/2006	17:24:00	0.418	234.84	18.4	
1	16/12/2006	17:25:00	0.436	233.63	23	
2	16/12/2006	17:26:00	0.498	233.29	23	
3	16/12/2006	17:27:00	0.502	233.74	23	
4	16/12/2006	17:28:00	0.528	235.68	15.8	

```
df2=df.drop([1,3],axis=0) #axis=0 for row
df2.head()
```

	Date	Time	Global_active_power	Global_reactive_power	Voltage	Global_ir
0	16/12/2006	17:24:00	4.216	0.418	234.84	
2	16/12/2006	17:26:00	5.374	0.498	233.29	
4	16/12/2006	17:28:00	3.666	0.528	235.68	
5	16/12/2006	17:29:00	3.52	0.522	235.02	
6	16/12/2006	17:30:00	3.702	0.52	235.09	

```
df.sample(10) #selesct randon 10 rows and print it
```

	Date	Time	Global_active_power	Global_reactive_power	Voltage	Glob
552835	4/1/2008	15:19:00	4.92	0	239.92	
619653	20/2/2008	0:57:00	0.24	0	241.15	
178361	19/4/2007	14:05:00	2.572	0.2	236.81	
851393	29/7/2008	23:17:00	0.45	0.132	241.61	
856680	2/8/2008	15:24:00	0.144	0	243.04	
911356	9/9/2008	14:40:00	0.188	0	241.06	
161892	8/4/2007	3:36:00	2.45	0.216	240	
971418	21/10/2008	7:42:00	2.05	0.246	237.05	
265580	19/6/2007	3:44:00	0.16	0.1	239.54	
903605	4/9/2008	5:29:00	0.31	0.19	238.84	

```
print(df.to_string())
```

```
IOPub data rate exceeded.
The notebook server will temporarily stop sending output
to the client in order to avoid crashing it.
To change this limit, set the config variable
`--NotebookApp.iopub_data_rate_limit`.
```

```
Current values:
NotebookApp.iopub_data_rate_limit=1000000.0 (bytes/sec)
NotebookApp.rate_limit_window=3.0 (secs)
```

```
# myvar=pd.DataFrame(df)
# myvar.head()
```

```
print(df[0])
```

```
234.84
```

```
df
```

	Date	Time	Global_active_power	Global_reactive_power	Voltage	Glo
0	16/12/2006	17:24:00	4.216	0.418	234.84	
1	16/12/2006	17:25:00	5.36	0.436	233.63	
2	16/12/2006	17:26:00	5.374	0.498	233.29	
3	16/12/2006	17:27:00	5.388	0.502	233.74	
4	16/12/2006	17:28:00	3.666	0.528	235.68	
...	...	...	...	...	...	
1048570	13/12/2008	21:34:00	0.426	0.076	242.27	
1048571	13/12/2008	21:35:00	0.424	0.076	242.1	
1048572	13/12/2008	21:36:00	0.422	0.076	241.73	
1048573	13/12/2008	21:37:00	0.422	0.078	242.56	
1048574	13/12/2008	21:38:00	0.422	0.078	242.61	

1048575 rows × 9 columns

```
#to remove empty cells
df.dropna(inplace=True) #to makes changes perm.
df.head()
```

	Date	Time	Global_active_power	Global_reactive_power	Voltage	Global_in
0	16/12/2006	17:24:00	4.216	0.418	234.84	
1	16/12/2006	17:25:00	5.36	0.436	233.63	
2	16/12/2006	17:26:00	5.374	0.498	233.29	
3	16/12/2006	17:27:00	5.388	0.502	233.74	
4	16/12/2006	17:28:00	3.666	0.528	235.68	

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1044506 entries, 0 to 1048574
Data columns (total 9 columns):
 #   Column                 Non-Null Count    Dtype
---  ------                 --------------    -----
 0   Date                   1044506 non-null  object
 1   Time                   1044506 non-null  object
 2   Global_active_power    1044506 non-null  object
 3   Global_reactive_power  1044506 non-null  object
 4   Voltage                1044506 non-null  object
 5   Global_intensity       1044506 non-null  object
 6   Sub_metering_1         1044506 non-null  object
 7   Sub_metering_2         1044506 non-null  object
 8   Sub_metering_3         1044506 non-null  float64
dtypes: float64(1), object(8)
memory usage: 79.7+ MB
```

```
#replace empty values
df.fillna(145,inplace=True)
df.head(100)
```

```
          Date      Time  Global_active_power  Global_reactive_power  Voltage  Global_i
0   16/12/2006  17:24:00                4.216                  0.418   234.84
1   16/12/2006  17:25:00                 5.36                  0.436   233.63
2   16/12/2006  17:26:00                5.374                  0.498   233.29
3   16/12/2006  17:27:00                5.388                  0.502   233.74
4   16/12/2006  17:28:00                3.666                  0.528   235.68
...        ...       ...                  ...                    ...      ...
95  16/12/2006  18:59:00                4.224                   0.09   231.96
96  16/12/2006  19:00:00                 4.07                  0.088   231.99
97  16/12/2006  19:01:00                3.612                   0.09   232.36
98  16/12/2006  19:02:00                3.458                   0.09   232.71
99  16/12/2006  19:03:00                3.434                   0.09   232.01

100 rows × 9 columns
```

```
#repalce null values for a specified coulmn
df['Sub_metering_3'].fillna(145,inplace=True)
df.head()
```

```
         Date      Time  Global_active_power  Global_reactive_power  Voltage  Global_ir
0  16/12/2006  17:24:00                4.216                  0.418   234.84
1  16/12/2006  17:25:00                 5.36                  0.436   233.63
2  16/12/2006  17:26:00                5.374                  0.498   233.29
3  16/12/2006  17:27:00                5.388                  0.502   233.74
4  16/12/2006  17:28:00                3.666                  0.528   235.68
```

```
#pandas use mean median amd mode to replace the empty values
x=df['Sub_metering_3'].mean()
df['Sub_metering_3'].fillna(x,inplace=True)
df.head()
```

```
         Date      Time  Global_active_power  Global_reactive_power  Voltage  Global_ir
0  16/12/2006  17:24:00                4.216                  0.418   234.84
1  16/12/2006  17:25:00                 5.36                  0.436   233.63
2  16/12/2006  17:26:00                5.374                  0.498   233.29
3  16/12/2006  17:27:00                5.388                  0.502   233.74
4  16/12/2006  17:28:00                3.666                  0.528   235.68
```

```
x=df['Sub_metering_3'].median()
df['Sub_metering_3'].fillna(x,inplace=True)
df.head()
```

```
         Date      Time  Global_active_power  Global_reactive_power  Voltage  Global_ir
0  16/12/2006  17:24:00                4.216                  0.418   234.84
1  16/12/2006  17:25:00                 5.36                  0.436   233.63
2  16/12/2006  17:26:00                5.374                  0.498   233.29
3  16/12/2006  17:27:00                5.388                  0.502   233.74
4  16/12/2006  17:28:00                3.666                  0.528   235.68
```

```
x=df['Sub_metering_3'].mode()
df['Sub_metering_3'].fillna(x,inplace=True)
df.head()
```

	Date	Time	Global_active_power	Global_reactive_power	Voltage	Global_ir
0	16/12/2006	17:24:00	4.216	0.418	234.84	
1	16/12/2006	17:25:00	5.36	0.436	233.63	
2	16/12/2006	17:26:00	5.374	0.498	233.29	
3	16/12/2006	17:27:00	5.388	0.502	233.74	
4	16/12/2006	17:28:00	3.666	0.528	235.68	

```
#coverting date column into dates
df.dtypes
```

```
Date                      object
Time                      object
Global_active_power       object
Global_reactive_power     object
Voltage                   object
Global_intensity          object
Sub_metering_1            object
Sub_metering_2            object
Sub_metering_3           float64
dtype: object
```

```
df['Date']=pd.to_datetime(df['Date'])
df.head()
```

	Date	Time	Global_active_power	Global_reactive_power	Voltage	Global_intens
0	2006-12-16	17:24:00	4.216	0.418	234.84	1
1	2006-12-16	17:25:00	5.36	0.436	233.63	
2	2006-12-16	17:26:00	5.374	0.498	233.29	

```
df.dtypes
```

```
Date                     datetime64[ns]
Time                             object
Global_active_power              object
Global_reactive_power            object
Voltage                          object
Global_intensity                 object
Sub_metering_1                   object
Sub_metering_2                   object
Sub_metering_3                  float64
dtype: object
```

```
df['Voltage']=pd.to_numeric(df['Voltage'])
df.head()
df.dtypes
```

```
Date                     datetime64[ns]
Time                             object
Global_active_power              object
Global_reactive_power            object
Voltage                         float64
Global_intensity                 object
Sub_metering_1                   object
Sub_metering_2                   object
Sub_metering_3                  float64
dtype: object
```

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

```
#replacing wrong values with right
df.loc[1.0,'Global_active_power']=45
df.head()
```

	Date	Time	Global_active_power	Global_reactive_power	Voltage	Global_inter
0.0	2006-12-16	17:24:00	4.216	0.418	234.84	
1.0	2006-12-16	17:25:00	45	0.436	233.63	
2.0	2006-12-16	17:26:00	5.374	0.498	233.29	

```
#finding and removing duplicates true if duplicates false if no dupicates
print(df.duplicated())
```

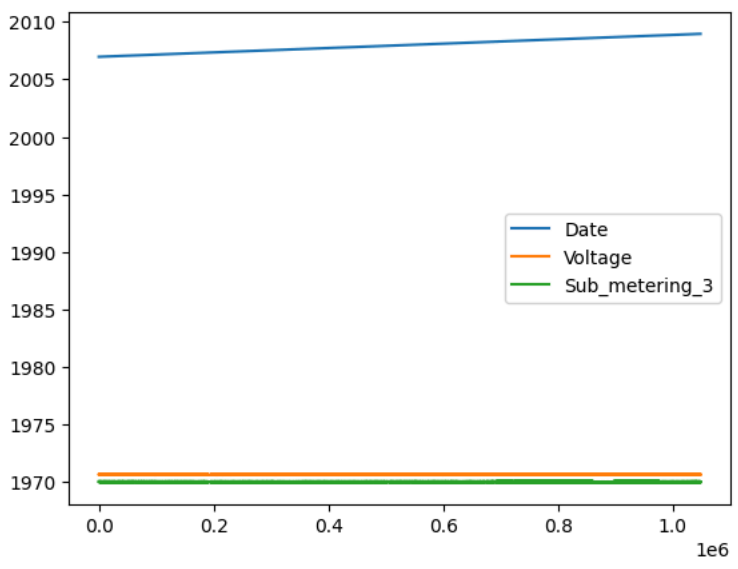
```
0.00          False
1.00          False
2.00          False
3.00          False
4.00          False
              ...
1048571.00    False
1048572.00    False
1048573.00    False
1048574.00    False
5.36          False
Length: 1044507, dtype: bool
```

```
#removing dupicates
df.drop_duplicates(inplace=True)
df.head()
```

	Date	Time	Global_active_power	Global_reactive_power	Voltage	Global_inter
0.0	2006-12-16	17:24:00	4.216	0.418	234.84	
1.0	2006-12-16	17:25:00	45	0.436	233.63	
2.0	2006-12-16	17:26:00	5.374	0.498	233.29	

```
import pandas as pd
import matplotlib.pyplot as plt
df.plot()

plt.show()
```



```
df.head()
df.dtypes
```

```
Date                     datetime64[ns]
Time                             object
Global_active_power              object
Global_reactive_power            object
```

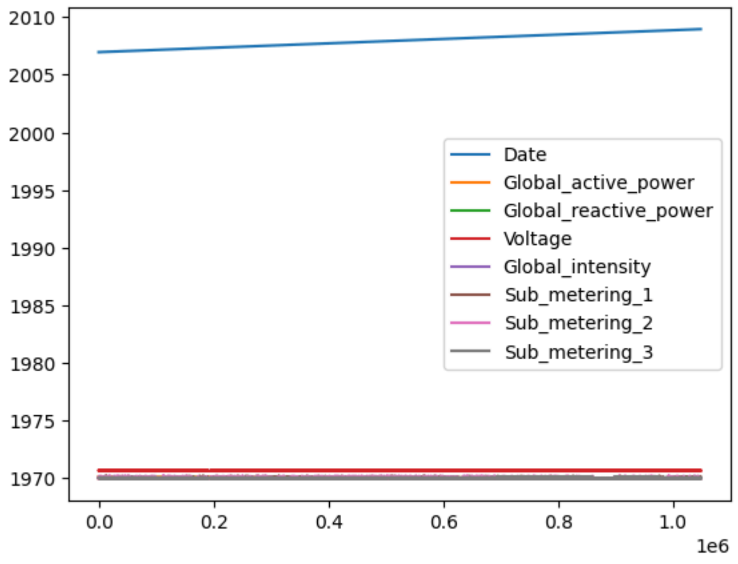
```
Voltage                         float64
Global_intensity                 object
Sub_metering_1                   object
Sub_metering_2                   object
Sub_metering_3                  float64
dtype: object
```

```
df['Sub_metering_2']=pd.to_numeric(df['Sub_metering_2'])
df['Sub_metering_1']=pd.to_numeric(df['Sub_metering_1'])
df['Global_active_power']=pd.to_numeric(df['Global_active_power'])
df['Global_reactive_power']=pd.to_numeric(df['Global_reactive_power'])
df['Global_intensity']=pd.to_numeric(df['Global_intensity'])
df.dtypes
```

```
Date                     datetime64[ns]
Time                             object
Global_active_power             float64
Global_reactive_power           float64
Voltage                         float64
Global_intensity                float64
Sub_metering_1                  float64
Sub_metering_2                  float64
Sub_metering_3                  float64
dtype: object
```

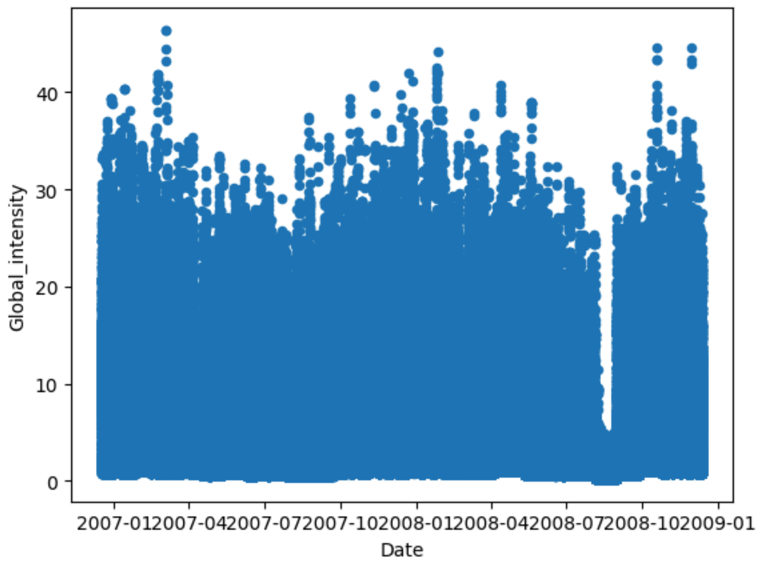
```
import pandas as pd
import matplotlib.pyplot as plt
df.plot()
```

```
<Axes: >
```

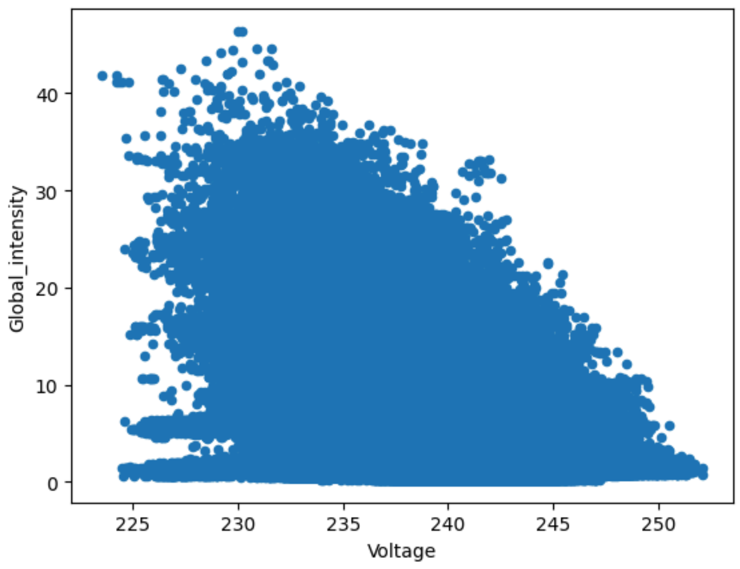


```
df.plot(kind='scatter',x='Date',y='Global_intensity')
```

```
<Axes: xlabel='Date', ylabel='Global_intensity'>
```

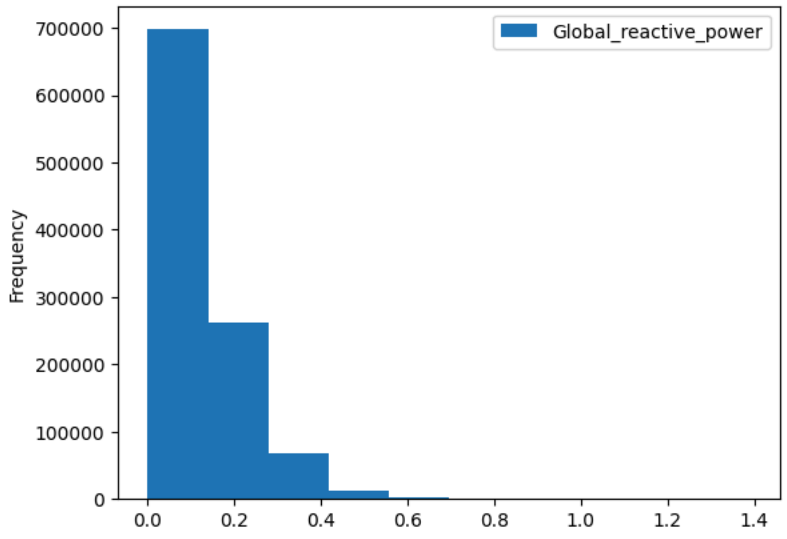


```
df.plot(kind='scatter',x='Voltage',y='Global_intensity')
plt.show()
```



```
df.plot(kind='hist',y='Global_reactive_power',x='Voltage')
```

```
<Axes: ylabel='Frequency'>
```



```
df.shape
```

```
(1044507, 9)
```

```
#no.of rows in series if 2D X with no. of columns
df.size
```

```
9400563
```

```
df.ndim
```

```
2
```

```
df.describe() #retruns mean median mode Q1 and Q3,etc
```

	Date	Global_active_power	Global_reactive_power	Voltage
count	1044506	1.044507e+06	1.044506e+06	1.044506e+06
mean	2007-12-16 03:01:45.445445120	1.108282e+00	1.182732e-01	2.399598e+02