6. Consider any sales training/ weather forecasting dataset

a. Compute mean of a series grouped by another series
b. Fill an intermittent time series to replace all missing dates with values of previous non-missing date.
c. Perform appropriate year-month string to dates conversion.
d. Split a dataset to group by two columns and then sort the aggregated results within the groups.
e. Split a given dataframe into groups with bin counts.

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
ls
```

```
Datasets/           Prac3.ipynb         Prac8 - Sheet1.csv  delete.pdf
PDFS/               Prac4.ipynb         Prac8.ipynb         prac7.csv
PYQ_Q2.csv          Prac5.ipynb         Prac_4_Sheet.csv
PYQ_Q4.csv          Prac6.ipynb         Practicle_1.ipynb
Prac2.ipynb         Prac7.ipynb         delete.ipynb
```

In [3]:
```
df=pd.read_csv("/Users/jatin/Documents/DAV/Practicles/Datasets/weatherHi
```

In [ ]:
```

```

In [4]:
```
df.head()
```

Out[4]:

	Formatted Date	Summary	Precip Type	Temperature (C)	Apparent Temperature (C)	Humidity	Wind Speed (km/h)	Wind Bearing (degrees)	Visib (
0	2006-04-01 00:00:00.000 +0200	Partly Cloudy	rain	9.472222	7.388889	0.89	14.1197	251.0	15.8
1	2006-04-01 01:00:00.000 +0200	Partly Cloudy	rain	9.355556	7.227778	0.86	14.2646	259.0	15.8
2	2006-04-01 02:00:00.000 +0200	Mostly Cloudy	rain	9.377778	9.377778	0.89	3.9284	204.0	14.9
3	2006-04-01 03:00:00.000 +0200	Partly Cloudy	rain	8.288889	5.944444	0.83	14.1036	269.0	15.8
4	2006-04-01 04:00:00.000 +0200	Mostly Cloudy	rain	8.755556	6.977778	0.83	11.0446	259.0	15.8

In [ ]:
```

```

In [ ]:
```

```

In [5]:
```
df=df.sample(frac=0.004)
```

In [6]:
```
df=df.drop(df[df['Summary'].isin(['Windy and Mostly Cloudy','Breezy and
```

In [7]:
```
df['Summary'].value_counts()
```

Out[7]:
```
Partly Cloudy                126
Mostly Cloudy                 89
Overcast                      86
Clear                         46
Foggy                         32
Breezy and Overcast            3
Humid and Mostly Cloudy        1
Breezy and Partly Cloudy       1
Breezy                         1
Breezy and Mostly Cloudy       1
Name: Summary, dtype: int64
```

In [ ]:
```

```

In [8]:
```
df
```

Out[8]:

	Formatted Date	Summary	Precip Type	Temperature (C)	Apparent Temperature (C)	Humidity	Wind Speed (km/h)	Wind Bearing (degrees)
56603	2012-07-22 11:00:00.000 +0200	Partly Cloudy	rain	18.838889	18.838889	0.79	14.9247	7.0
17554	2008-04-01 10:00:00.000 +0200	Mostly Cloudy	rain	10.022222	10.022222	0.58	14.3290	22.0
83563	2015-06-02 22:00:00.000 +0200	Partly Cloudy	rain	19.888889	19.888889	0.69	6.2951	120.0
40697	2010-03-29 18:00:00.000 +0200	Mostly Cloudy	rain	17.844444	17.844444	0.40	11.0446	183.0
48770	2011-06-30 02:00:00.000 +0200	Mostly Cloudy	rain	16.783333	16.783333	0.91	16.2288	300.0
...	...	...	...	...	...	...	...	...
34224	2009-10-04 00:00:00.000 +0200	Clear	rain	5.555556	5.555556	0.86	0.0000	0.0
95975	2016-09-19 02:00:00.000 +0200	Overcast	rain	17.638889	17.638889	0.84	5.7960	16.0
20490	2008-01-10 18:00:00.000 +0100	Foggy	snow	-2.244444	-2.244444	1.00	0.0000	0.0
49731	2011-05-11 04:00:00.000 +0200	Mostly Cloudy	rain	13.677778	13.677778	0.76	6.2790	29.0
36708	2010-12-15 12:00:00.000 +0100	Overcast	snow	-2.222222	-6.644444	0.88	12.7351	309.0

386 rows × 12 columns

In [ ]:

In [ ]:

In [ ]:

a. Compute mean of a series grouped by another series

In [9]:
```
df.groupby(['Precip Type','Summary']).mean(numeric_only=1).drop('Loud Co
```

Out[9]:

Precip Type	Summary	Temperature (C)	Apparent Temperature (C)	Humidity	Wind Speed (km/h)	Wind Bearing (degrees)	Visibility (km)	Pr (mil
rain	Breezy	6.111111	1.605556	0.600000	28.980000	320.000000	9.982000	0.0
	Breezy and Mostly Cloudy	22.222222	22.222222	0.760000	28.980000	280.000000	7.084000	1004.3
	Breezy and Overcast	9.662963	6.981481	0.696667	29.221500	246.333333	9.944433	1007.2
	Breezy and Partly Cloudy	21.111111	21.111111	0.260000	35.420000	150.000000	9.982000	1011.9
	Clear	15.194577	14.704497	0.753333	6.699133	160.071429	11.322517	969.5
	Foggy	6.853216	6.228363	0.960000	5.131663	180.684211	1.718463	1017.1
	Humid and Mostly Cloudy	20.011111	20.011111	0.870000	12.155500	71.000000	11.125100	1010.0
	Mostly Cloudy	13.737468	13.006460	0.719651	11.297707	179.418605	11.139328	1016.4
	Overcast	9.570118	7.948653	0.838636	12.457253	194.909091	9.646095	999.2
	Partly Cloudy	17.745151	17.413324	0.627119	10.713868	201.915254	12.001868	1016.5
snow	Clear	-5.283333	-7.855556	0.867500	7.486500	131.500000	10.082625	1030.6
	Foggy	-4.329915	-5.826923	0.928462	4.634323	174.384615	1.836638	1025.8
	Mostly Cloudy	-5.733333	-9.791667	0.850000	9.893450	180.500000	10.505250	1027.5
	Overcast	-2.819722	-6.812222	0.888500	12.366410	236.450000	6.120415	1017.6
	Partly Cloudy	-3.419841	-8.119841	0.741429	12.983500	168.571429	9.308100	1023.9

In [ ]:

In [ ]:

In [ ]:

b. Fill an intermittent time series to replace all missing dates with values of previous non-missing date.

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

c. Perform appropriate year-month string to dates conversion.

In [10]:
```
from datetime import datetime
```

In [11]:
```
date_format = '%Y-%m-%d %H:%M:%S'
df["Formatted Date"]=df["Formatted Date"].apply(lambda x:datetime.strpti
```

In [12]:
```
df['Formatted Date'].dtypes
```

Out[12]:
```
dtype('<M8[ns]')
```

In [13]:
```
np.dtype('datetime64[ns]') == np.dtype('<M8[ns]')
```

Out[13]:
```
True
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

d. Split a dataset to group by two columns and then sort the aggregated results within the groups.

In [14]:
```
res=df.groupby(['Precip Type','Summary']).mean(numeric_only=1)['Humidity
res
```

Out[14]:
```
Precip Type  Summary
rain         Breezy                        0.600000
             Breezy and Mostly Cloudy      0.760000
             Breezy and Overcast           0.696667
             Breezy and Partly Cloudy      0.260000
             Clear                         0.753333
             Foggy                         0.960000
             Humid and Mostly Cloudy       0.870000
             Mostly Cloudy                 0.719651
             Overcast                      0.838636
             Partly Cloudy                 0.627119
snow         Clear                         0.867500
             Foggy                         0.928462
             Mostly Cloudy                 0.850000
             Overcast                      0.888500
             Partly Cloudy                 0.741429
Name: Humidity, dtype: float64
```

In [ ]:

In [ ]:

In [15]:
```
res.groupby(level=1).sum().sort_values()
```

Out[15]:
```
Summary
Breezy and Partly Cloudy    0.260000
Breezy                      0.600000
Breezy and Overcast         0.696667
Breezy and Mostly Cloudy    0.760000
Humid and Mostly Cloudy     0.870000
Partly Cloudy               1.368547
Mostly Cloudy               1.569651
Clear                       1.620833
Overcast                    1.727136
Foggy                       1.888462
Name: Humidity, dtype: float64
```

In [ ]:

In [ ]:

In [16]:
```
res.groupby(level=0).sum().sort_values()
```

Out[16]:
```
Precip Type
snow    4.275890
rain    7.085406
Name: Humidity, dtype: float64
```

In [ ]:

In [ ]:

e.Split a given dataframe into groups with bin counts.

In [ ]:

In [ ]:

In [17]:
```
grp=df.groupby(['Precip Type',pd.cut(df['Visibility (km)'],5)])
grp.size().unstack()
```

Out[17]:

Visibility (km)	(-0.0161, 3.22]	(3.22, 6.44]	(6.44, 9.66]	(9.66, 12.88]	(12.88, 16.1]
Precip Type					
rain	20	39	24	156	99
snow	13	18	4	8	3

In [ ]: