Pivot and Pivot tables

A pivot table in Pandas is a quantitive table that summarizes a large DataFrame, such as a large dataset. It is a component of data processing. In pivot table, the report may include average, mode, summation or other statistical elements.

In ~~Pivot~~ Pivot table, can take both catagorical (latter) or numerical variable ~~are taken~~

- Pivot tables in pandas are popularly seen in MS Excel files. In python, Pivot tables of pandas dataframes can be created using the command: pivot_table
- You can aggregate a numeric column as a cross tabulation against two categorical columns.
- Pivot allows to reshape the data frame.

```
import pandas as pd
datas = pd.read_csv('weather.csv')
```

call ⇒

datas

	date	city	temperature	humidity	wind speed	weather condition
0	2024-04-1	New York	70	50	10	Sunny
1	2024-04-1	Los Angeles	75	55	12	Cloudy
2	2024-04-02	New York	72	48	9	Partly cloudy
3	2024-04-02	Los A	77	60	11	Rainy
4	2024-4-3	New Yo	68	52	8	Sunny
5	2024-4-3	Los An	73	57	13	Thunderstorm

datas. pivot(index = 'date', columns = 'city')

city / date	temp		hum.		w-speed		w-condi	
	Los Ang	New York	New York / Los Ang	N Y	L A	N Y	L A	N Y
2024-4-1	75	70	55	50	12	10	Cloudy	Sunny
2024-4-2	77	72	60	48	11	9	Rainy	Partly Clo
2024-4-3	73	68	57	52	13	8	Thunderstom	Sunny

datas. pivot(index = 'date', columns = 'city', values = 'temper

city / date	Los Angeles	New York
2024-4-1	75	70
2024-4-2	77	72
2024-4-3	73	68

⊙ Pivot Table is used to summarize or aggregate data with in a dataframe.

data21 = pd. read_csv ('weather2.csv')
data1 . pivot_table (index = 'city', columns = 'date')

date / city	humidity			temp			wind-spe		
	2024-4-1	2024-4-2	2024-4-3	2024-4-1	24-4-2	24-4-3	24-4-1	~~24-4-2~~ 24-4-2	24-4-3
Los Angeles	55	60	57	75	77	73	12	11	13
New York	50	48	52	70	72	68	10	9	8

data21. pivot_table (index = 'city', columns = 'date', aggfunc = 'sum')
same output

datas. pivot_table (index = 'city, columns = 'date', aggfunc = 'sum', margins = True)

date / City	humidity				temperature				wind-speed			
	24-4-1	24-4-2	24-4-3	All	24-4-1	24-4-2	24-4-2	All	24-4-1	24-4-2	24-4-3	All
Los Angeles	55	60	57	172	75	77	73	225	12	11	13	36
New York	50	48	52	150	70	72	68	210	10	9	8	27
All	105	106	109	322	145	149	141	435	22	20	21	68

⊙ Grouper

datas2 = pd.read_csv ('weather2.csv')

call

datas 2

	date	city	temperature	humidity	wind_speed
0	2024-4-1	New York	70	50	10
1	2024-4-1	Los Angeles	75	55	12
2	2024-4-2	New York	72	48	9
3	2024-4-2	Los Angeles	77	60	11
4	2024-4-3	New York	68	52	8
5	2024-4-3	Los Angeles	73	57	13

(In this case date is string)

⊙ datas2 ['date'] = pd.to_datetime (datas2['date'])

The new outur will be same as previous output but date is change to datestamp

⊙ datas2.pivot_table (index=pd.Grouper(freq='M', key='date'), columns='city')

city	humidity		temperature		wind_speed	
date	Los Angeles	New York	Los Angeles	New York	Los Angeles	New York
2024-4-30	57.333333	50.0	75	70	12	9

datas2.pivot_table (index = pd.Grouper (freq='M', key='date'), columns='city', aggfunc='sur

city	humidity		temperature		wind_speed	
date	Los Angeles	New York	Los Angeles	New York	Los Angeles	New York
2024-4-30	172	150	225	210	36	27

melt..

① Melt allows to transform or reshape the dataframe

datas = pd.read-csv('weather.csv')

datas

	date	city	temperature	humidity	wind-speed	weather-condition
0	2024-4-1	NewYork	70	50	10	Sunny
1	2024-4-1	LosAngeles	75	55	12	Cloudy
2	2024-4-2	NewYork	72	48	9	Partly Cloudy
3	2024-4-2	LosAngeles	77	60	11	Rainy
4	2024-4-3	NewYork	68	52	8	Sunny
5	2024-4-3	LosAngeles	73	57	13	Thunderstorms

pd.melt(datas, id-vars=['date'])

	date	variable	value
0	2024-4-1	city	NewYork
1	"	city	LosAngeles
2	2024-4-2	city	NewYork
3	"	"	Los Ang
4	2024-4-3	"	
5	"	"	
6	2024-4-1	temperature	
7	2024-4-1	"	
......		"	

→ All data will seperate

pd.melt(datas, id-vars=['date'], value_vars='temperature', var_name = 'city')

	date	city	value
0	2024-4-1	temperature	70
1	"	"	75
2	2024-4-2	"	72
3	"	"	77
4	2024-4-3	"	68
5	"	"	73