Day 29- Facilitation Guide
(Pandas Pivot Table)

Index

(1.50 hrs) ILT

I. Recap

In our last session we learned:

- **Pandas Data Analysis:** Pandas is a popular Python library for data analysis that provides powerful data structures and data manipulation tools. It is widely used in data science, machine learning, and data analysis projects.

In this session we are going to understand the pandas pivot table for data analysis.

II. Pandas Pivot Table

A pandas pivot table is a data manipulation tool used to reorganize and reshape data in a tabular form, typically used for summarizing, aggregating, and restructuring data for better analysis.
It's a powerful feature in the pandas library that allows you to transform data from long format to wide format or vice versa, making it easier to perform various data operations and generate insights.

Here are some key aspects of pandas pivot tables:

Aggregation: Pivot tables are often used to aggregate data. You can specify one or more columns as the index, one or more columns as columns, and a column to aggregate using a specific function (e.g., sum, mean, count) to create a summary of the data.

Reshaping: Pivot tables can transform data from a "long" format (where each row represents an observation) to a "wide" format (where columns represent categories or groups), and vice versa.

Multi-level Indexing: You can create pivot tables with multi-level index columns, allowing you to organize and group data hierarchically.

Handling Missing Data: Pivot tables can handle missing data efficiently, enabling you to choose how to deal with NaN or None values when aggregating.

Syntax:

pivot_table(data=, values=None, index=None, columns=None, aggfunc='mean', fill_value=None, margins=False, dropna=True, margins_name='All', observed=False, sort=True)

The method takes a DataFrame and then also returns a DataFrame. The table below provides an overview of the different parameters available in the function:

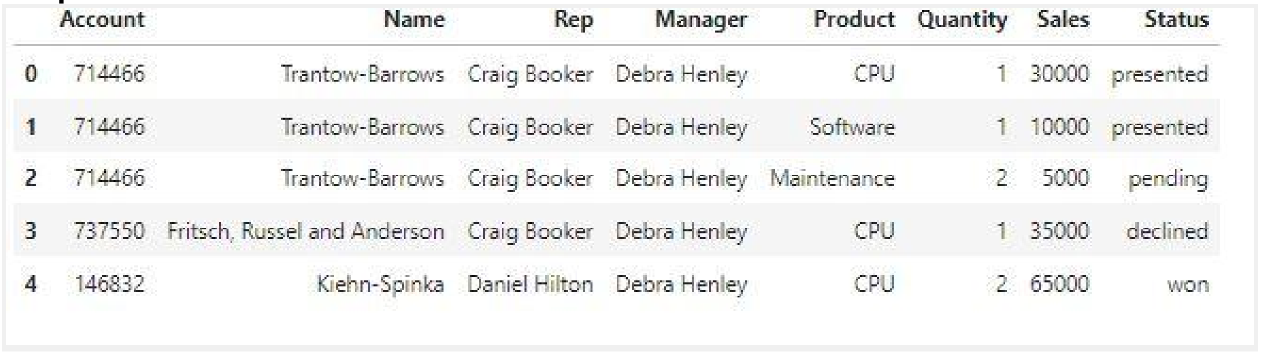
Parameter	**Default Value**	**Description**
`data=`		The DataFrame to pivot
`values=`		The column to aggregate (if blank, will aggregate all numerical values)
`index=`		The column or columns to group data by. A single column can be a string, while multiple columns should be a list of strings
`columns=`		The column or columns to group data by. A single column can be a string, while multiple columns should be a list of strings
`aggfunc=`	'mean'	A function or list of functions to aggregate data by
`fill_value=`		Value to replace missing values with
`margins=`	False	Add a row and column for totals
`dropna=`	True	To choose to not include columns where all entries are NaN
`margins_name=`	'All'	Name of total row/column
`observed=`	False	Only for categorical data – if `True` will only show observed values for categorical groups

`sort=`	True	Whether to sort the resulting values

<u>Quick Examples of Pandas Pivot Table</u>
Read in the data

```
import pandas as pd
#Read in our sales data into our DataFrame
df = pd.read_csv("sales-data.csv")
df.head()
```

Output:



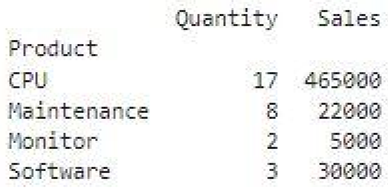
	Account	Name	Rep	Manager	Product	Quantity	Sales	Status
0	714466	Trantow-Barrows	Craig Booker	Debra Henley	CPU	1	30000	presented
1	714466	Trantow-Barrows	Craig Booker	Debra Henley	Software	1	10000	presented
2	714466	Trantow-Barrows	Craig Booker	Debra Henley	Maintenance	2	5000	pending
3	737550	Fritsch, Russel and Anderson	Craig Booker	Debra Henley	CPU	1	35000	declined
4	146832	Kiehn-Spinka	Daniel Hilton	Debra Henley	CPU	2	65000	won

Pivot the data
The simplest pivot table must have a dataframe and an index . In this case, let’s use the Name as our index.
Syntax:
*pd.pivot_table(df,index=["*Product*"])*

Example:

```
import pandas as pd
import numpy as np
#Read in our sales data into our DataFrame
df = pd.read_csv("sales-data.csv")
print(pd.pivot_table(df,index=["Product"],values=['Sales','Quantity'],aggfunc=np.sum))
```

Output:



```
             Quantity   Sales
Product
CPU                17  465000
Maintenance         8   22000
Monitor             2    5000
Software            3   30000
```

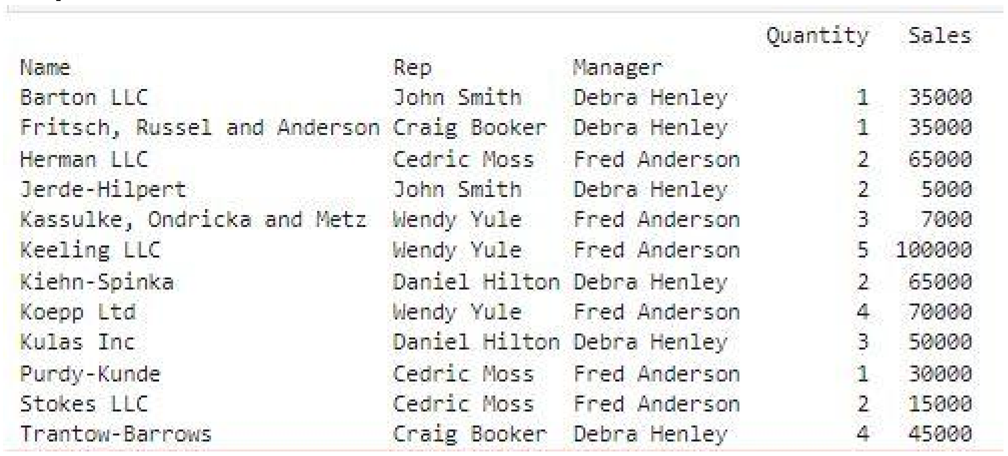
The above output shows the total quantity of products sold and the total sales value.
You can have multiple indexes as well. In fact, most of the pivot_table args can take multiple values via a list.
Syntax:

pd.pivot_table(df,index=["Name","Rep","Manager"])

Example:

```
import pandas as pd
import numpy as np
#Read in our sales data into our DataFrame
df = pd.read_csv("sales-data.csv")
print(pd.pivot_table(df,index=["Name","Rep","Manager"],values=['Sales','Quantity'],ag
gfunc=np.sum))
```

Output:



```
                                                          Quantity   Sales
Name                          Rep           Manager
Barton LLC                    John Smith    Debra Henley         1   35000
Fritsch, Russel and Anderson Craig Booker  Debra Henley         1   35000
Herman LLC                    Cedric Moss   Fred Anderson        2   65000
Jerde-Hilpert                 John Smith    Debra Henley         2    5000
Kassulke, Ondricka and Metz   Wendy Yule    Fred Anderson        3    7000
Keeling LLC                   Wendy Yule    Fred Anderson        5  100000
Kiehn-Spinka                  Daniel Hilton Debra Henley         2   65000
Koepp Ltd                     Wendy Yule    Fred Anderson        4   70000
Kulas Inc                     Daniel Hilton Debra Henley         3   50000
Purdy-Kunde                   Cedric Moss   Fred Anderson        1   30000
Stokes LLC                    Cedric Moss   Fred Anderson        2   15000
Trantow-Barrows               Craig Booker  Debra Henley         4   45000
```

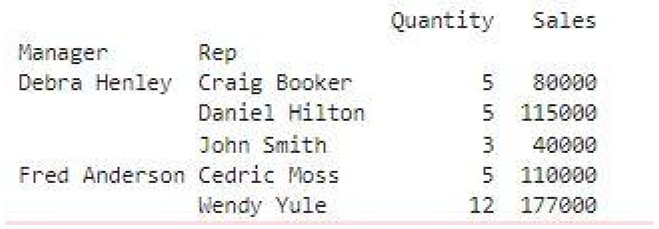
The above output shows the company-wise total sales along with the names of representative and manager
We probably want to look at this by Manager and Rep. It’s easy enough to do by changing the index .

Syntax:

pd.pivot_table(df,index=["Manager","Rep"])
Example:

```
import pandas as pd
import numpy as np
#Read in our sales data into our DataFrame
df = pd.read_csv("sales-data.csv")
print(pd.pivot_table(df,index=["Manager","Rep"],values=['Sales','Quantity'],aggfunc=np
.sum))
```

Output:



```
                              Quantity   Sales
Manager       Rep
Debra Henley  Craig Booker           5   80000
              Daniel Hilton          5  115000
              John Smith             3   40000
Fred Anderson Cedric Moss            5  110000
              Wendy Yule            12  177000
```

So now we can view the same data Manager-wise. This will help to assess the performance of the employees during the year.

<u>Columns vs. Values</u>

One of the confusing points with the pivot_table is the use of columns and values.
Remember, columns are optional - they provide an additional way to segment the actual values you care about.
The aggregation functions are applied to the values you list.

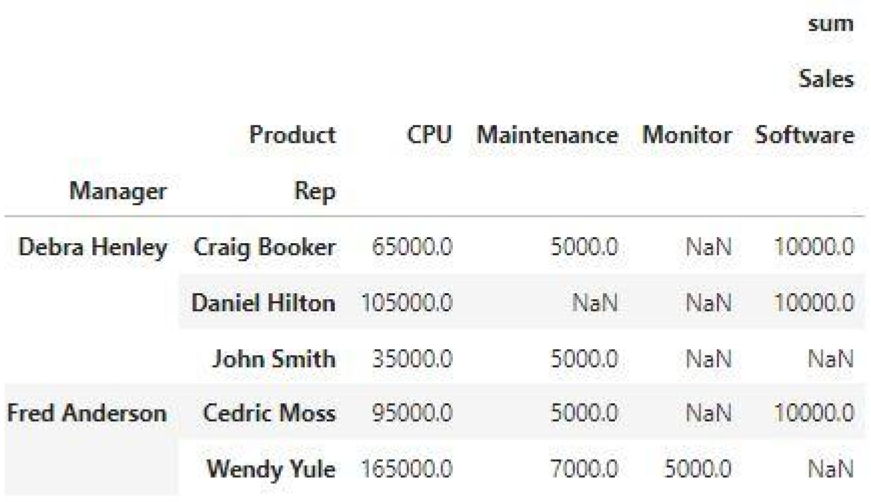
Syntax:

pd.pivot_table(df,index=["Manager","Rep"],values=["Sales"],
columns=["Product"],aggfunc=[np.sum])

Example:

```
import pandas as pd
import numpy as np
#Read in our sales data into our DataFrame
df = pd.read_csv("sales-data.csv")
pd.pivot_table(df,index=["Manager","Rep"],values=["Sales"],
columns=["Product"],aggfunc=[np.sum])
```

Output:



		sum			
		Sales			
	Product	CPU	Maintenance	Monitor	Software
Manager	Rep				
Debra Henley	Craig Booker	65000.0	5000.0	NaN	10000.0
	Daniel Hilton	105000.0	NaN	NaN	10000.0
	John Smith	35000.0	5000.0	NaN	NaN
Fred Anderson	Cedric Moss	95000.0	5000.0	NaN	10000.0
	Wendy Yule	165000.0	7000.0	5000.0	NaN

Here, you can see the total sales of each product for a particular Manager as well as their representatives.
The NaN's are a bit distracting. If we want to remove them, we could use fill_value to set them to 0.

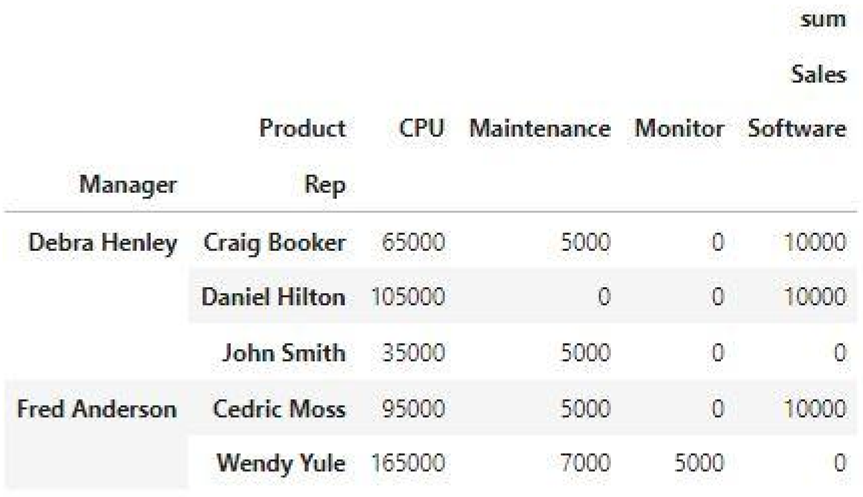
Syntax:

```
pd.pivot_table(df,index=["Manager","Rep"],values=["Sales"],
            columns=["Product"],aggfunc=[np.sum],fill_value=0)
```

Example:

```
import pandas as pd
import numpy as np
#Read in our sales data into our DataFrame
df = pd.read_csv("sales-data.csv")
pd.pivot_table(df,index=["Manager","Rep"],values=["Sales"],
            columns=["Product"],aggfunc=[np.sum],fill_value=0)
```

Output:



		sum			
		Sales			
	Product	CPU	Maintenance	Monitor	Software
Manager	Rep				
Debra Henley	Craig Booker	65000	5000	0	10000
	Daniel Hilton	105000	0	0	10000
	John Smith	35000	5000	0	0
Fred Anderson	Cedric Moss	95000	5000	0	10000
	Wendy Yule	165000	7000	5000	0

It would be useful to add the quantity as well. Add Quantity to the values list.

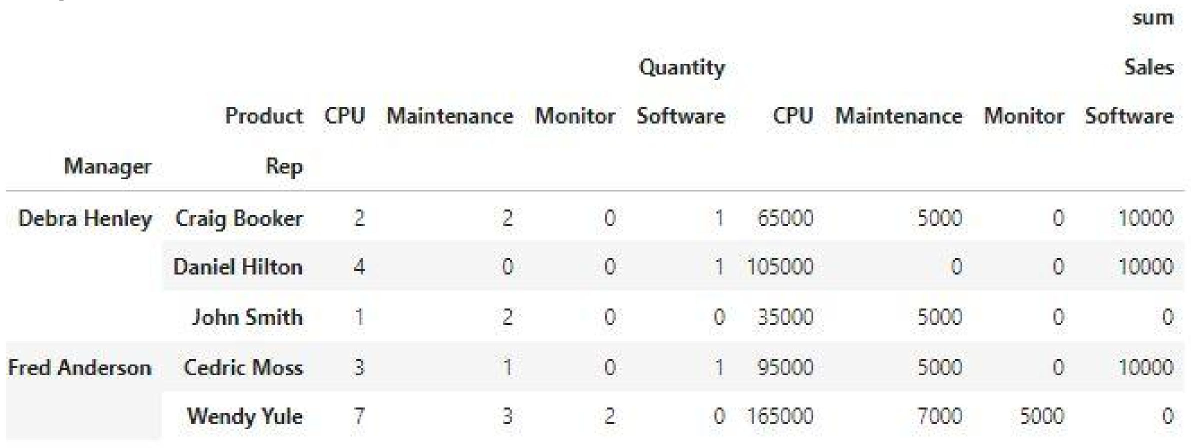
Syntax:

```
pd.pivot_table(df,index=["Manager","Rep"],values=["Sales","Quantity"],
            columns=["Product"],aggfunc=[np.sum],fill_value=0)
```

Example:

```
import pandas as pd
import numpy as np
#Read in our sales data into our DataFrame
df = pd.read_csv("sales-data.csv")
pd.pivot_table(df,index=["Manager","Rep"],values=["Sales","Quantity"],
            columns=["Product"],aggfunc=[np.sum],fill_value=0)
```

Output:



									sum
					Quantity				Sales
	Product	CPU	Maintenance	Monitor	Software	CPU	Maintenance	Monitor	Software
Manager	Rep								
Debra Henley	Craig Booker	2	2	0	1	65000	5000	0	10000
	Daniel Hilton	4	0	0	1	105000	0	0	10000
	John Smith	1	2	0	0	35000	5000	0	0
Fred Anderson	Cedric Moss	3	1	0	1	95000	5000	0	10000
	Wendy Yule	7	3	2	0	165000	7000	5000	0

You can move items to the index to get a different visual representation. Remove Product from the columns and add to the index .

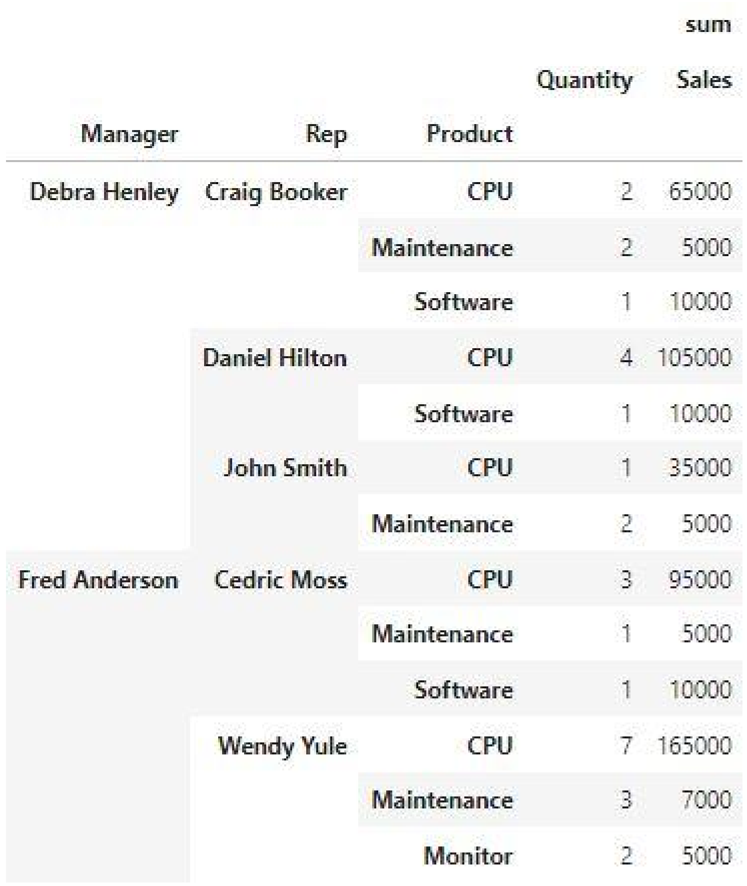
Syntax:

```
pd.pivot_table(df,index=["Manager","Rep","Product"],
            values=["Sales","Quantity"],aggfunc=[np.sum],fill_value=0)
```

Example:

```
import pandas as pd
import numpy as np
#Read in our sales data into our DataFrame
df = pd.read_csv("sales-data.csv")
pd.pivot_table(df,index=["Manager","Rep","Product"],
            values=["Sales","Quantity"],aggfunc=[np.sum],fill_value=0)
```

Output:



			sum	
			Quantity	Sales
Manager	Rep	Product		
Debra Henley	Craig Booker	CPU	2	65000
		Maintenance	2	5000
		Software	1	10000
	Daniel Hilton	CPU	4	105000
		Software	1	10000
	John Smith	CPU	1	35000
		Maintenance	2	5000
Fred Anderson	Cedric Moss	CPU	3	95000
		Maintenance	1	5000
		Software	1	10000
	Wendy Yule	CPU	7	165000
		Maintenance	3	7000
		Monitor	2	5000

For this data set, this representation makes more sense.

Now, what if you want to see some totals? margins=True does that for us.

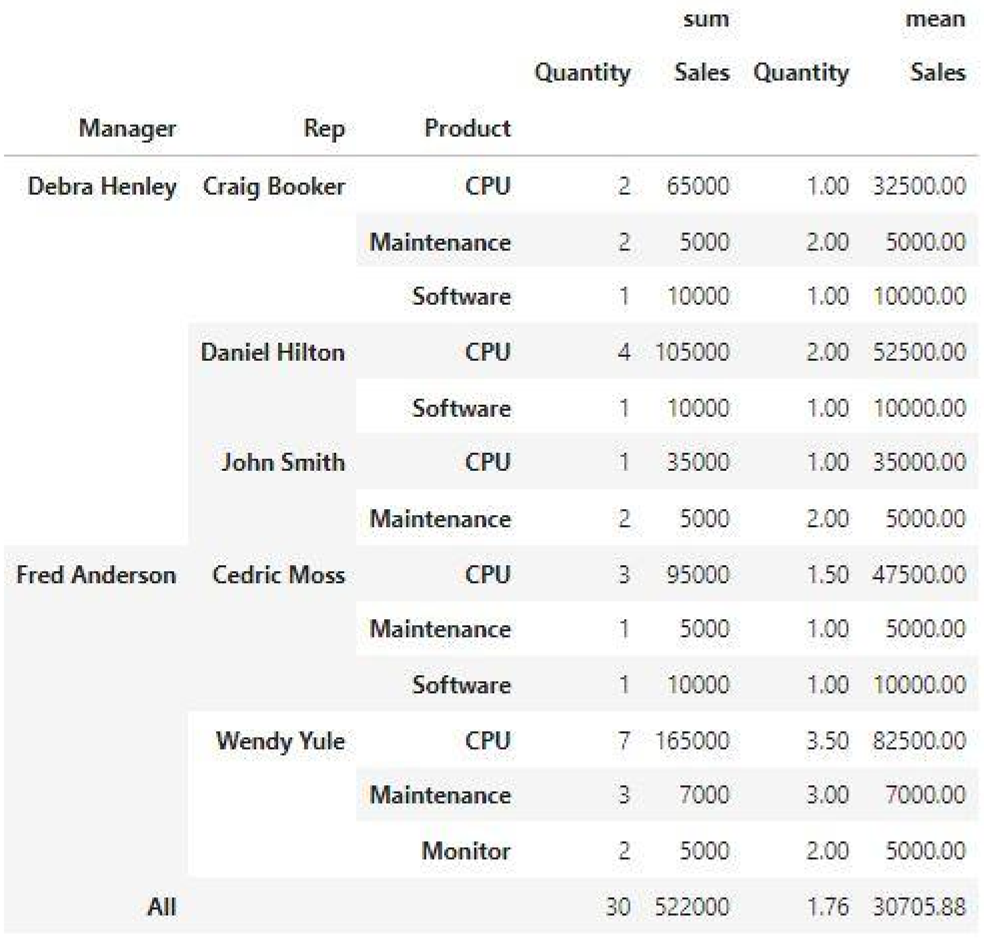
Syntax:

```
pd.pivot_table(df,index=["Manager","Rep","Product"],
            values=["Sales","Quantity"],
            aggfunc=[np.sum,np.mean],fill_value=0,margins=True)
```

Example:

```
import pandas as pd
import numpy as np
#Read in our sales data into our DataFrame
df = pd.read_csv("sales-data.csv")
np.round(pd.pivot_table(df,index=["Manager","Rep","Product"],
            values=["Sales","Quantity"],
            aggfunc=[np.sum,np.mean],fill_value=0,margins=True),2)
```

Output:



```
                                                 sum                 mean
                                          Quantity    Sales  Quantity      Sales
Manager        Rep           Product
Debra Henley   Craig Booker  CPU                 2    65000      1.00   32500.00
                             Maintenance         2     5000      2.00    5000.00
                             Software            1    10000      1.00   10000.00
               Daniel Hilton CPU                 4   105000      2.00   52500.00
                             Software            1    10000      1.00   10000.00
               John Smith    CPU                 1    35000      1.00   35000.00
                             Maintenance         2     5000      2.00    5000.00
Fred Anderson  Cedric Moss   CPU                 3    95000      1.50   47500.00
                             Maintenance         1     5000      1.00    5000.00
                             Software            1    10000      1.00   10000.00
               Wendy Yule    CPU                 7   165000      3.50   82500.00
                             Maintenance         3     7000      3.00    7000.00
                             Monitor             2     5000      2.00    5000.00
All                                             30   522000      1.76   30705.88
```

Let’s move the analysis up a level and look at our pipeline at the manager level. Notice how the status is ordered based on our earlier category definition.

Syntax:

```
pd.pivot_table(df,index=["Manager","Status"],values=["Sales"],
            aggfunc=[np.sum],fill_value=0,margins=True)
```

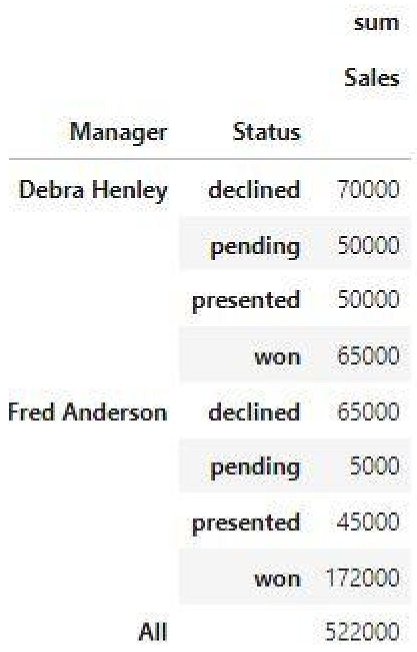
Example:

```
import pandas as pd
import numpy as np

#Read in our sales data into our DataFrame
df = pd.read_csv("sales-data.csv")

pd.pivot_table(df,index=["Manager","Status"],values=["Sales"],
            aggfunc=[np.sum],fill_value=0,margins=True)
```

Output:



```
                                  sum
                                Sales
       Manager      Status
  Debra Henley    declined      70000
                   pending      50000
                 presented      50000
                       won      65000
 Fred Anderson    declined      65000
                   pending       5000
                 presented      45000
                       won     172000
           All                 522000
```

A really handy feature is the ability to pass a dictionary to the aggfunc so you can perform different functions on each of the values you select. This has a side-effect of making the labels a little cleaner.

Syntax:
pd.pivot_table(df,index=["Manager","Status"],columns=["Product"],values=["Quantity","Sales"],aggfunc={"Quantity":len,"Price":np.sum},fill_value=0)

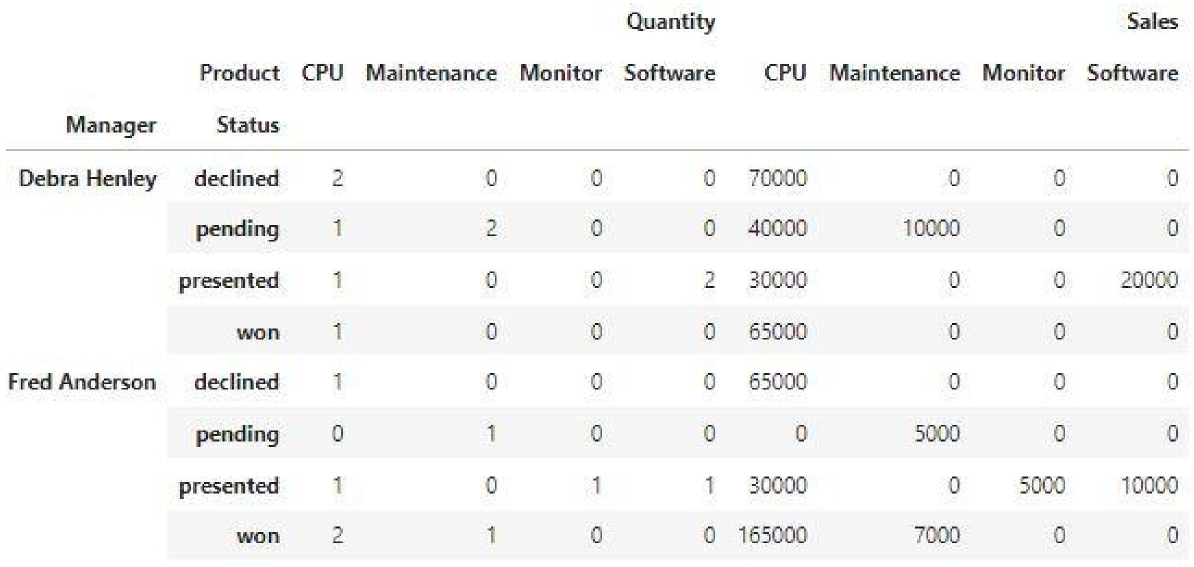
Example:

```
import pandas as pd
import numpy as np
```

```
#Read in our sales data into our DataFrame
df = pd.read_csv("sales-data.csv")

pd.pivot_table(df,index=["Manager","Status"],columns=["Product"],values=["Quantity","
Sales"],aggfunc={"Quantity":len,"Sales":np.sum},fill_value=0)
```

Output:



		Quantity				Sales			
	Product	CPU	Maintenance	Monitor	Software	CPU	Maintenance	Monitor	Software
Manager	Status								
Debra Henley	declined	2	0	0	0	70000	0	0	0
	pending	1	2	0	0	40000	10000	0	0
	presented	1	0	0	2	30000	0	0	20000
	won	1	0	0	0	65000	0	0	0
Fred Anderson	declined	1	0	0	0	65000	0	0	0
	pending	0	1	0	0	0	5000	0	0
	presented	1	0	1	1	30000	0	5000	10000
	won	2	1	0	0	165000	7000	0	0

You can provide a list of agg functions to apply to each value too.

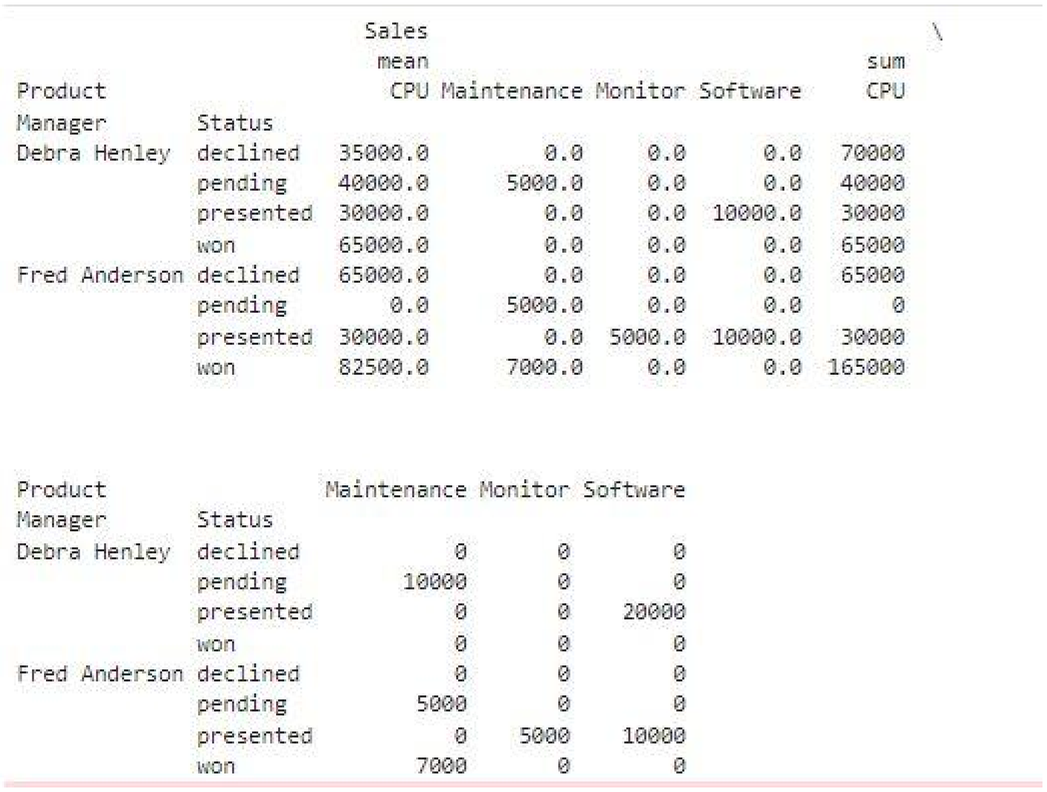
Syntax:
table =
pd.pivot_table(df,index=["Manager","Status"],columns=["Product"],values=["Sales"],
aggfunc={"Sales":[np.sum,np.mean]},fill_value=0)

Example:

```
import pandas as pd
import numpy as np
#Read in our sales data into our DataFrame
df = pd.read_csv("sales-data.csv")

table =
pd.pivot_table(df,index=["Manager","Status"],columns=["Product"],values=["Sales"],
aggfunc={"Sales":[np.sum,np.mean]},fill_value=0)
print(table)
```

Output:



```
                          Sales                                         \
                           mean                                    sum
Product                     CPU Maintenance Monitor Software       CPU
Manager       Status
Debra Henley  declined  35000.0         0.0     0.0      0.0     70000
              pending   40000.0      5000.0     0.0      0.0     40000
              presented 30000.0         0.0     0.0  10000.0     30000
              won       65000.0         0.0     0.0      0.0     65000
Fred Anderson declined  65000.0         0.0     0.0      0.0     65000
              pending       0.0      5000.0     0.0      0.0         0
              presented 30000.0         0.0  5000.0  10000.0     30000
              won       82500.0      7000.0     0.0      0.0    165000


Product                 Maintenance Monitor Software
Manager       Status
Debra Henley  declined            0       0        0
              pending         10000       0        0
              presented           0       0    20000
              won                 0       0        0
Fred Anderson declined            0       0        0
              pending          5000       0        0
              presented           0    5000    10000
              won              7000       0        0
```

Advanced Pivot Table Filtering

Once you have generated your data, it is in a DataFrame so you can filter on it using your standard DataFrame functions.

If you want to look at just one manager:

Syntax:
table.query()

table.query(query_string)

Parameters:

query_string: This is a string containing the query expression that specifies the filtering condition. The query string is used to filter the rows of the table based on conditions specified within the string.

table.query('Manager == ["Debra Henley"]')

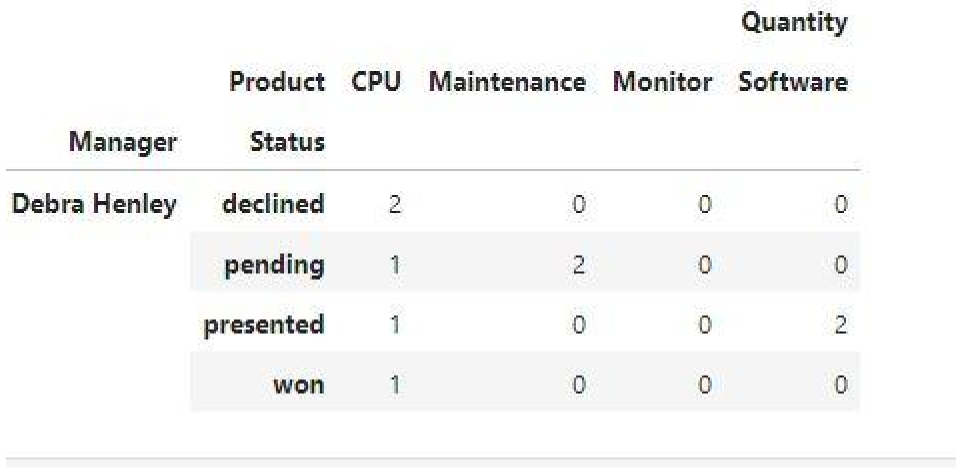
Example:

```
import pandas as pd
import numpy as np

#Read in our sales data into our DataFrame
df = pd.read_csv("sales-data.csv")

table =
pd.pivot_table(df,index=["Manager","Status"],columns=["Product"],values=["Quantity"],
aggfunc={"Quantity":len},fill_value=0)
table.query('Manager == ["Debra Henley"]')
```

Output:



		Quantity			
	Product	CPU	Maintenance	Monitor	Software
Manager	Status				
Debra Henley	declined	2	0	0	0
	pending	1	2	0	0
	presented	1	0	0	2
	won	1	0	0	0

We can look at all of our pending and won deals.
Syntax:

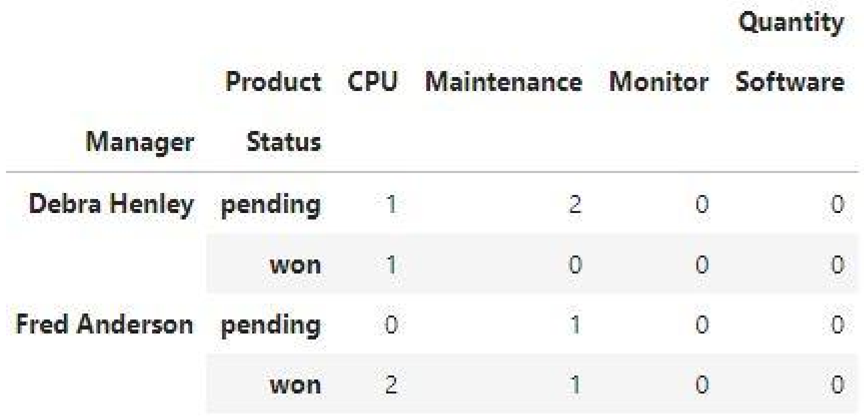
```
table.query('Status == ["pending","won"]')
```

Example:

```
import pandas as pd
import numpy as np
#Read in our sales data into our DataFrame
df = pd.read_csv("sales-data.csv")

table =
pd.pivot_table(df,index=["Manager","Status"],columns=["Product"],values=["Quantity"],
aggfunc={"Quantity":len},fill_value=0)
table.query('Status == ["pending","won"]')
```

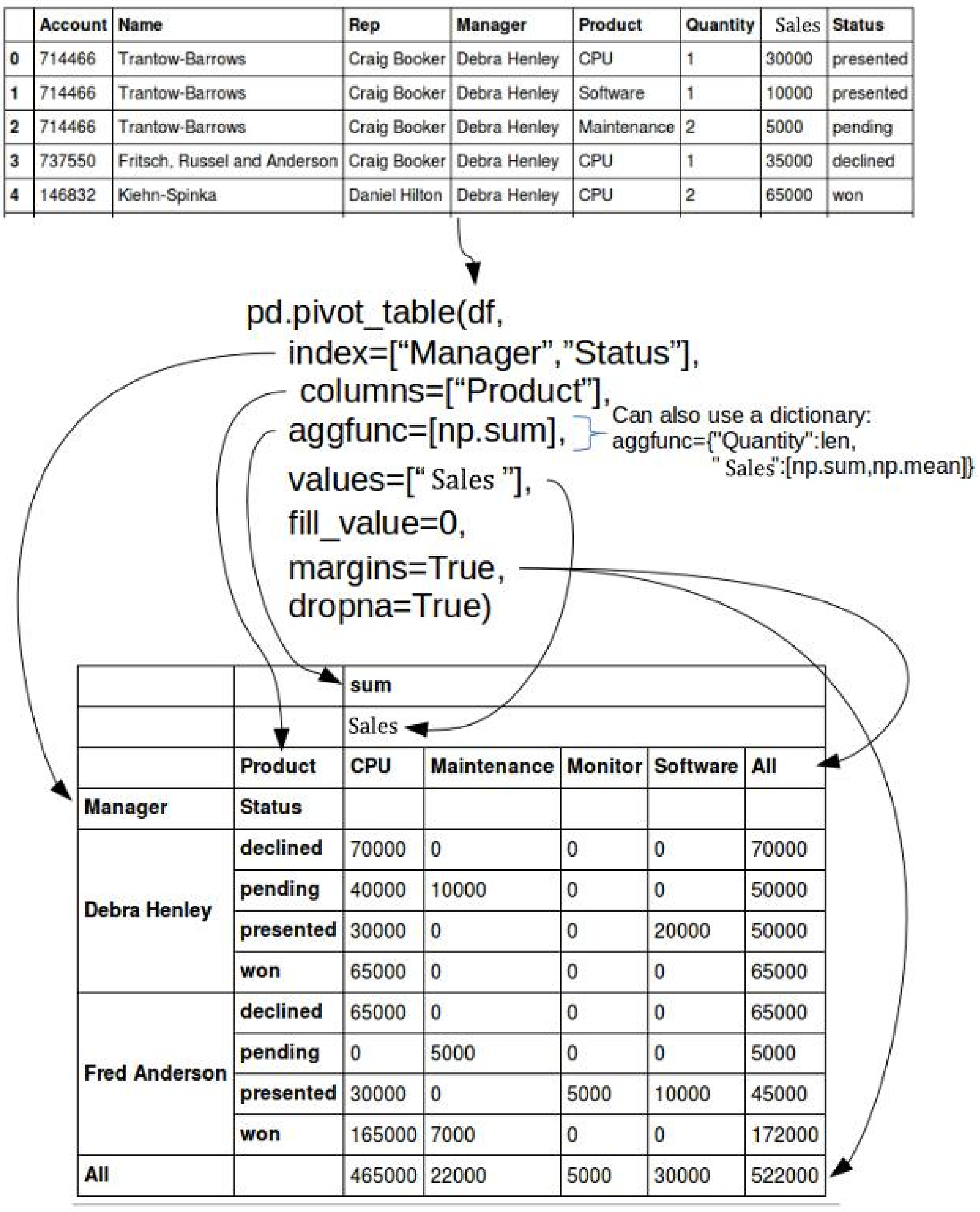
Output:



		Quantity			
	Product	CPU	Maintenance	Monitor	Software
Manager	Status				
Debra Henley	pending	1	2	0	0
	won	1	0	0	0
Fred Anderson	pending	0	1	0	0
	won	2	1	0	0

Cheat Sheet
In order to try to summarize all of this, please see the below cheat sheet that will help you remember how to use the pandas pivot_table.

pandas pivot_table explained



	Account	Name	Rep	Manager	Product	Quantity	Sales	Status
0	714466	Trantow-Barrows	Craig Booker	Debra Henley	CPU	1	30000	presented
1	714466	Trantow-Barrows	Craig Booker	Debra Henley	Software	1	10000	presented
2	714466	Trantow-Barrows	Craig Booker	Debra Henley	Maintenance	2	5000	pending
3	737550	Fritsch, Russel and Anderson	Craig Booker	Debra Henley	CPU	1	35000	declined
4	146832	Kiehn-Spinka	Daniel Hilton	Debra Henley	CPU	2	65000	won

```
pd.pivot_table(df,
  index=[“Manager”,”Status”],
   columns=[“Product”],
  aggfunc=[np.sum],
  values=[“ Sales ”],
  fill_value=0,
  margins=True,
  dropna=True)
```

Can also use a dictionary:
aggfunc={"Quantity":len,
" Sales":[np.sum,np.mean]}

		sum				
		Sales				
	Product	**CPU**	**Maintenance**	**Monitor**	**Software**	**All**
Manager	**Status**					
Debra Henley	**declined**	70000	0	0	0	70000
	pending	40000	10000	0	0	50000
	presented	30000	0	0	20000	50000
	won	65000	0	0	0	65000
Fred Anderson	**declined**	65000	0	0	0	65000
	pending	0	5000	0	0	5000
	presented	30000	0	5000	10000	45000
	won	165000	7000	0	0	172000
All		465000	22000	5000	30000	522000

Exercise

Use GPT to write a program:

1.Hi! I want a complete Python code example with dummy sales data and Pandas pivot table analysis. I want to work with sales data and analyze it to gain insights. I want to display pivot tables and also visualize the data like pie plot and bar plot.