Groupby

The groupby method allows you to group rows of data together and call aggregate functions

In [1]:

```
import pandas as pd
# Create dataframe
data = {'Company':['GOOG','GOOG','MSFT','MSFT','FB','FB'],
       'Person':['Sam','Charlie','Amy','Vanessa','Carl','Sarah'],
       'Sales':[200,120,340,124,243,350]}
```

In [2]:

```
print(data)
```

```
{'Company': ['GOOG', 'GOOG', 'MSFT', 'MSFT', 'FB', 'FB'], 'Person':
['Sam', 'Charlie', 'Amy', 'Vanessa', 'Carl', 'Sarah'], 'Sales': [200,
120, 340, 124, 243, 350]}
```

In [3]:

```
df = pd.DataFrame(data)
```

In [5]:

```
df
```

Out[5]:

	Company	Person	Sales
0	GOOG	Sam	200
1	GOOG	Charlie	120
2	MSFT	Amy	340
3	MSFT	Vanessa	124
4	FB	Carl	243
5	FB	Sarah	350

In [10]:

```
df.groupby('Company')


res = df.groupby('Company')

res
```

Out[10]:

```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x7f721f49d0b
8>
```

In [12]:

```
1 200+120
```

Out[12]:

320

In [14]:

```
1 320/2
```

Out[14]:

160.0

In [16]:

```
1 (340+124)/2
```

Out[16]:

232.0

In [15]:

```
1 (243+350)/2
```

Out[15]:

296.5

In [9]:

```
1 res.mean()
```

Out[9]:

	Sales
Company	
FB	296.5
GOOG	160.0
MSFT	232.0

** Now you can use the .groupby() method to group rows together based off of a column name. For instance let's group based off of Company. This will create a DataFrameGroupBy object:**

In [34]:

```
1 df.groupby('Company')
```

Out[34]:

```
<pandas.core.groupby.DataFrameGroupBy object at 0x113014128>
```

You can save this object as a new variable:

In [18]:

```
by_comp = df.groupby("Company")
```

And then call aggregate methods off the object:

In [19]:

```
by_comp.mean()
```

Out[19]:

Company	Sales
FB	296.5
GOOG	160.0
MSFT	232.0

In [20]:

```
df.groupby('Company').mean()
```

Out[20]:

Company	Sales
FB	296.5
GOOG	160.0
MSFT	232.0

In [21]:

```
df.groupby('Company').mean()
```

Out[21]:

Company	Sales
FB	296.5
GOOG	160.0
MSFT	232.0

In [24]:

```
1 res.std()
```

Out[24]:

	Sales
Company	
FB	75.660426
GOOG	56.568542
MSFT	152.735065

In [28]:

```
1 res.max()
```

Out[28]:

	Person	Sales
Company		
FB	Sarah	350
GOOG	Sam	200
MSFT	Vanessa	340

In [29]:

```
1 df
```

Out[29]:

	Company	Person	Sales
0	GOOG	Sam	200
1	GOOG	Charlie	120
2	MSFT	Amy	340
3	MSFT	Vanessa	124
4	FB	Carl	243
5	FB	Sarah	350

In [27]:

```
1 res.min()
```

Out[27]:

Company	Person	Sales
FB	Carl	243
GOOG	Charlie	120
MSFT	Amy	124

In [30]:

```
1 res.count()
```

Out[30]:

Company	Person	Sales
FB	2	2
GOOG	2	2
MSFT	2	2

In [31]:

```
1 res.describe()
```

Out[31]:

	Sales							
	count	mean	std	min	25%	50%	75%	max
Company								
FB	2.0	296.5	75.660426	243.0	269.75	296.5	323.25	350.0
GOOG	2.0	160.0	56.568542	120.0	140.00	160.0	180.00	200.0
MSFT	2.0	232.0	152.735065	124.0	178.00	232.0	286.00	340.0

More examples of aggregate methods:

In [38]:

```
1 by_comp.std()
```

Out[38]:

Company	Sales
FB	75.660426
GOOG	56.568542
MSFT	152.735065

In [39]:

```
1 by_comp.min()
```

Out[39]:

Company	Person	Sales
FB	Carl	243
GOOG	Charlie	120
MSFT	Amy	124

In [40]:

```
1 by_comp.max()
```

Out[40]:

Company	Person	Sales
FB	Sarah	350
GOOG	Sam	200
MSFT	Vanessa	340

In [41]:

```
1 by_comp.count()
```

Out[41]:

Company	Person	Sales
FB	2	2
GOOG	2	2
MSFT	2	2

In [42]:

```
1 by_comp.describe()
```

Out[42]:

		Sales
Company		
FB	**count**	2.000000
	mean	296.500000
	std	75.660426
	min	243.000000
	25%	269.750000
	50%	296.500000
	75%	323.250000
	max	350.000000
GOOG	**count**	2.000000
	mean	160.000000
	std	56.568542
	min	120.000000
	25%	140.000000
	50%	160.000000
	75%	180.000000
	max	200.000000
MSFT	**count**	2.000000
	mean	232.000000
	std	152.735065
	min	124.000000
	25%	178.000000
	50%	232.000000
	75%	286.000000
	max	340.000000

In [43]:

```
1 by_comp.describe().transpose()
```

Out[43]:

Company	FB								GOOG			
	count	mean	std	min	25%	50%	75%	max	count	mean	...	75
Sales	2.0	296.5	75.660426	243.0	269.75	296.5	323.25	350.0	2.0	160.0	...	180

1 rows × 24 columns

In [44]:

```
1 by_comp.describe().transpose()['GOOG']
```

Out[44]:

	count	mean	std	min	25%	50%	75%	max
Sales	2.0	160.0	56.568542	120.0	140.0	160.0	180.0	200.0

Type *Markdown* and LaTeX: α^2

In [32]:

```
1 res.describe()
```

Out[32]:

	Sales							
	count	mean	std	min	25%	50%	75%	max
Company								
FB	2.0	296.5	75.660426	243.0	269.75	296.5	323.25	350.0
GOOG	2.0	160.0	56.568542	120.0	140.00	160.0	180.00	200.0
MSFT	2.0	232.0	152.735065	124.0	178.00	232.0	286.00	340.0

In [33]:

```
1 res.describe().transpose()
```

Out[33]:

	Company	FB	GOOG	MSFT
Sales	**count**	2.000000	2.000000	2.000000
	mean	296.500000	160.000000	232.000000
	std	75.660426	56.568542	152.735065
	min	243.000000	120.000000	124.000000
	25%	269.750000	140.000000	178.000000
	50%	296.500000	160.000000	232.000000
	75%	323.250000	180.000000	286.000000
	max	350.000000	200.000000	340.000000

In [36]:

```
1 df
```

Out[36]:

	Company	Person	Sales
0	GOOG	Sam	200
1	GOOG	Charlie	120
2	MSFT	Amy	340
3	MSFT	Vanessa	124
4	FB	Carl	243
5	FB	Sarah	350

In [39]:

```
1 res.describe()
```

Out[39]:

	Sales							
	count	**mean**	**std**	**min**	**25%**	**50%**	**75%**	**max**
Company								
FB	2.0	296.5	75.660426	243.0	269.75	296.5	323.25	350.0
GOOG	2.0	160.0	56.568542	120.0	140.00	160.0	180.00	200.0
MSFT	2.0	232.0	152.735065	124.0	178.00	232.0	286.00	340.0

In [40]:

```
1 res.describe().transpose()['GOOG']
```

Out[40]:

```
Sales  count      2.000000
       mean     160.000000
       std       56.568542
       min      120.000000
       25%      140.000000
       50%      160.000000
       75%      180.000000
       max      200.000000
Name: GOOG, dtype: float64
```

In [41]:

```
1 res.describe().transpose()['FB']
```

Out[41]:

```
Sales  count      2.000000
       mean     296.500000
       std       75.660426
       min      243.000000
       25%      269.750000
       50%      296.500000
       75%      323.250000
       max      350.000000
Name: FB, dtype: float64
```

In [42]:

```
1 res.describe().transpose()['MSFT']
```

Out[42]:

```
Sales  count      2.000000
       mean     232.000000
       std      152.735065
       min      124.000000
       25%      178.000000
       50%      232.000000
       75%      286.000000
       max      340.000000
Name: MSFT, dtype: float64
```

In [ ]:

```
1
```