Missing Data

In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
d = {'A':[1,np.nan,2],'B':[2,np.nan,np.nan],'C':[4,5,6]}
```

In [3]:
```
df = pd.DataFrame(d)
```

In [4]:
```
df
```

Out[4]:

	A	B	C
0	1.0	2.0	4
1	NaN	NaN	5
2	2.0	NaN	6

In [5]:
```
df.dropna()
```

Out[5]:

	A	B	C
0	1.0	2.0	4

In [6]:
```
df
```

Out[6]:

	A	B	C
0	1.0	2.0	4
1	NaN	NaN	5
2	2.0	NaN	6

In [7]:
```
df.dropna(axis=1)
```

Out[7]:

	C
0	4
1	5
2	6

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

Out[8]:

	A	B	C
0	1.0	2.0	4
1	NaN	NaN	5
2	2.0	NaN	6

In [9]:
```
df.dropna(thresh=2)
```

Out[9]:

	A	B	C
0	1.0	2.0	4
2	2.0	NaN	6

In [10]:
```
df
```

Out[10]:

	A	B	C
0	1.0	2.0	4
1	NaN	NaN	5
2	2.0	NaN	6

In [11]:
```
df.fillna(value='fill')
```

Out[11]:

	A	B	C
0	1	2	4
1	fill	fill	5
2	2	fill	6

In [12]:
```
df['A'].fillna(value=df['A'].mean())
```

Out[12]:
```
0    1.0
1    1.5
2    2.0
Name: A, dtype: float64
```

In [13]:
```
df
```

Out[13]:

	A	B	C
0	1.0	2.0	4
1	NaN	NaN	5
2	2.0	NaN	6

Group By

Group By allows you to group together rows based off a column and perform an aggregate function on them

In [14]:
```
data = {'Company':['GOOG','GOOG','MSFT','MSFT','FB','FB'],
        'Person':['Sam','Charlie','Amy','Vanessa','Carl','Sarah'],
        'Sales':[200,120,340,124,243,350]}
```

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

In [15]:

In [16]:
```
df
```

Out[16]:

	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 [17]:
```
byComp = df.groupby('Company')
```

In [18]:
```
byComp.mean()
```

Out[18]:

Company	Sales
FB	296.5
GOOG	160.0
MSFT	232.0

In [19]:
```
byComp.std()
```

Out[19]:

Company	Sales
FB	75.660426
GOOG	56.568542
MSFT	152.735065

In [20]:
```
byComp.sum().loc['FB']
```

Out[20]:
```
Sales    593
Name: FB, dtype: int64
```

In [23]:
```
df.groupby('Sales').count()
```

Out[23]:

Sales	Company	Person
120	1	1
124	1	1
200	1	1

Sales	Company	Person
243	1	1
340	1	1
350	1	1

In [24]:
```
df.groupby('Company').count()
```

Out[24]:

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

In [25]:
```
df.groupby('Company').min()
```

Out[25]:

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

In [26]:
```
df.groupby('Company').max()
```

Out[26]:

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

In [27]:
```
df.groupby('Company').describe()
```

Out[27]:

	Sales							
Company	count	mean	std	min	25%	50%	75%	max
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 [28]:
```
df.groupby('Company').describe().transpose()
```

Out[28]:

	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

Merging, Joining & Concatinating

In [37]:
```
df1 = pd.DataFrame({'A': ['A0', 'A1', 'A2', 'A3'],
                    'B': ['B0', 'B1', 'B2', 'B3'],
                    'C': ['C0', 'C1', 'C2', 'C3'],
                    'D': ['D0', 'D1', 'D2', 'D3']},
                   index=[0, 1, 2, 3])
```

In [38]:
```
df1
```

Out[38]:

	A	B	C	D
0	A0	B0	C0	D0
1	A1	B1	C1	D1
2	A2	B2	C2	D2
3	A3	B3	C3	D3

In [39]:
```
df2 = pd.DataFrame({'A': ['A4', 'A5', 'A6', 'A7'],
                    'B': ['B4', 'B5', 'B6', 'B7'],
                    'C': ['C4', 'C5', 'C6', 'C7'],
                    'D': ['D4', 'D5', 'D6', 'D7']},
                     index=[4, 5, 6, 7])
```

In [42]:
```
df3 = pd.DataFrame({'A': ['A8', 'A9', 'A10', 'A11'],
                    'B': ['B8', 'B9', 'B10', 'B11'],
                    'C': ['C8', 'C9', 'C10', 'C11'],
                    'D': ['D8', 'D9', 'D10', 'D11']},
                   index=[8, 9, 10, 11])
```

In [43]:
```
df3
```

Out[43]:

	A	B	C	D
8	A8	B8	C8	D8

	A	B	C	D
9	A9	B9	C9	D9
10	A10	B10	C10	D10
11	A11	B11	C11	D11

In [45]:
```
pd.concat([df1,df2,df3])
```

Out[45]:

	A	B	C	D
0	A0	B0	C0	D0
1	A1	B1	C1	D1
2	A2	B2	C2	D2
3	A3	B3	C3	D3
4	A4	B4	C4	D4
5	A5	B5	C5	D5
6	A6	B6	C6	D6
7	A7	B7	C7	D7
8	A8	B8	C8	D8
9	A9	B9	C9	D9
10	A10	B10	C10	D10
11	A11	B11	C11	D11

In [46]:
```
pd.concat([df1,df2,df3],axis=1)
```

Out[46]:

	A	B	C	D	A	B	C	D	A	B	C	D
0	A0	B0	C0	D0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	A1	B1	C1	D1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	A2	B2	C2	D2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	A3	B3	C3	D3	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN	A4	B4	C4	D4	NaN	NaN	NaN	NaN
5	NaN	NaN	NaN	NaN	A5	B5	C5	D5	NaN	NaN	NaN	NaN
6	NaN	NaN	NaN	NaN	A6	B6	C6	D6	NaN	NaN	NaN	NaN
7	NaN	NaN	NaN	NaN	A7	B7	C7	D7	NaN	NaN	NaN	NaN
8	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	A8	B8	C8	D8
9	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	A9	B9	C9	D9
10	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	A10	B10	C10	D10
11	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	A11	B11	C11	D11

In [47]:
```
left = pd.DataFrame({'key': ['K0', 'K1', 'K2', 'K3'],
                     'A': ['A0', 'A1', 'A2', 'A3'],
                     'B': ['B0', 'B1', 'B2', 'B3']})

right = pd.DataFrame({'key': ['K0', 'K1', 'K2', 'K3'],
                          'C': ['C0', 'C1', 'C2', 'C3'],
                          'D': ['D0', 'D1', 'D2', 'D3']})
```

In [48]:
```
left
```

Out[48]:

	key	A	B
0	K0	A0	B0
1	K1	A1	B1
2	K2	A2	B2
3	K3	A3	B3

In [49]:
```
right
```

Out[49]:

	key	C	D
0	K0	C0	D0
1	K1	C1	D1
2	K2	C2	D2
3	K3	C3	D3

Merge function allows to merge DataFrames together using a similar logic as merging SQL tables together

In [50]:
```
pd.merge(left,right,how='inner',on='key')
```

Out[50]:

	key	A	B	C	D
0	K0	A0	B0	C0	D0
1	K1	A1	B1	C1	D1
2	K2	A2	B2	C2	D2
3	K3	A3	B3	C3	D3

In [52]:
```
right['C'].unique()
```

Out[52]:
```
array(['C0', 'C1', 'C2', 'C3'], dtype=object)
```

In [53]:
```
right['C'].nunique()
```

Out[53]:
```
4
```

In [54]:
```
right['C'].value_counts()
```

```
C0    1
```

Out[54]:
```
C1    1
C2    1
C3    1
Name: C, dtype: int64
```

In [55]:
```
new_df = df.groupby('Company').describe()
```

In [56]:
```
new_df
```

Out[56]:

	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 [62]:
```
#new_df[new_df['count']>120.0]
```

In [64]:
```
def times_two(x):
    return x*x
```

In [66]:
```
df
```

Out[66]:

	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 [67]:
```
df['Sales'].apply(times_two)
```

Out[67]:
```
0     40000
1     14400
2    115600
3     15376
4     59049
5    122500
Name: Sales, dtype: int64
```

In [ ]:
```

```