Pandas Cheat Sheet

In [1]:

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

Series

Creating a Series

In [87]:

```
labels = ['a','b','c']
my_list = [10,20,30]
arr = np.array([10,20,30])
d = {'a':10,'b':20,'c':30}
```

In [ ]:

```
display(pd.Series(data=my_list)) # using list
display(pd.Series(data=my_list,index=labels)) # adding index
display(pd.Series(arr)) #using array
display(pd.Series(d)) # using series
```

Using index to grab info

In [ ]:

```
ser1 = pd.Series([1,2,3,4],index = ['USA', 'Germany','USSR', 'Japan'])
ser1
```

In [ ]:

```
ser2 = pd.Series([5,6,7,8],index = ['USA', 'Germany','Italy', 'Japan'])
ser2
```

In [ ]:

```
ser1['USA']
```

In [ ]:

```
# addingg results in summation of values with common index
ser1 + ser2
```

Dataframes

In [11]:

```
from numpy.random import randn
np.random.seed(101)
```

In [12]:

```
df = pd.DataFrame(randn(5,4),index='A B C D E'.split(),columns='W X Y Z'.split())
df
```

In [ ]:

```
df
```

Indexing

In [ ]:

```
display(df['W']) # series using []
display(df[["W"]]) # df using [[]]
display(df.W) # series using .

display(df[["W", "Y"]])
```

Checking type

In [ ]:

```
type(df['W'])
```

Creating new column

In [19]:

```
df['new'] = df['W'] + df['Y']
df
```

Dropping columns

In [ ]:

```
df.drop('new',axis=1) #axis=1 is used for cols. use inplace=1 to make permanent changes
```

In [ ]:

```
df
```

Dropping rows

In [ ]:

```
df.drop('E',axis=0)
```

Row Selection

In [ ]:

```
display(df.loc['A'])
display(df.iloc[2])
```

Selecting rows and columns

In [ ]:

```
display(df.loc['B','Y'])

display(df.loc[['A','B'],['W','Y']])

display(df.iloc[:2, [0, 1]])
```

Conditional Selection

In [ ]:

```
display(df>0)

display(df[df>0])

display(df[df['W']>0])

display(df[df['W']>0]['Y'])

display(df[df['W']>0][['Y','X']])

display(df[(df['W']>0) & (df['Y'] > 1)])
```

Indexing

In [ ]:

```
df
```

In [ ]:

```
# Reset to default 0,1...n index
df.reset_index()
```

In [ ]:

```
newind = 'CA NY WY OR CO'.split()
df['States'] = newind
df.set_index('States')
```

In [ ]:

```
display(df)
df.set_index('States',inplace=True)
display(df)
```

Basic Operations

In [ ]:

```
# forst 5 rows
df.head()
```

In [ ]:

```
# last 5 rows
df.tail()
```

In [ ]:

```
# get numbber of rows and columns
print("rows:", df.shape[0])
print("columns:", df.shape[1])
```

In [ ]:

```
# number of unique values
df.nunique()
```

In [ ]:

```
df["W"].value_counts()
```

In [ ]:

```
# checking for null values
df.isnull()
```

In [ ]:

```
df.isna().sum()
```

In [ ]:

```
df.sort_values(by='W') #inplace=False by default
```

In [ ]:

```
# quick stats
df.describe()
```

In [ ]:

```
# querying dataframe
df.query("W >= 0.5")
```

Missing Data

In [51]:

```
df = pd.DataFrame({'A':[1,2,np.nan],
                  'B':[5,np.nan,np.nan],
                  'C':[1,2,3]})

df
```

In [ ]:

```
df
```

In [ ]:

```
df.dropna() # drop the rows with null val|ues
```

In [ ]:

```
df.dropna(axis=1) # drop cols with null values
```

In [ ]:

```
df.dropna(thresh=2) # drop rows below a threshold
```

In [ ]:

```
df.fillna(value='FILL VALUE') # fill nan values with specified value
```

In [ ]:

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

to make permenant changes use inplace=1

In [ ]:

```
df
```

Groupby

In [59]:

```
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)
df.head()
```

In [ ]:

```
df.groupby('Company') # groupby object
```

In [62]:

```
by_comp = df.groupby("Company") # saving the object
by_comp["Sales"].mean()
```

some aggregation examples

In [ ]:

```
display(df.groupby("Company")["Sales"].mean())
display(df.groupby("Company")["Sales"].std())
display(df.groupby("Company")["Sales"].min())
display(df.groupby("Company")["Sales"].count())
```

Merging and Joining

In [68]:

```
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])

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])

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])

display(df1)
display(df2)
display(df3)
```

Concatenation: stacks dataframes together

dimensions should match along the axis you are concatenating

In [ ]:

```
display(pd.concat([df1,df2,df3]))
display(pd.concat([df1,df2,df3],axis=1))
```

Merging: similar to merging SQL tables

In [74]:

```
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']})

display(left)
display(right)
```

In [ ]:

```
pd.merge(left,right,how='inner',on='key')
```

In [78]:

```
left = pd.DataFrame({'key1': ['K0', 'K0', 'K1', 'K2'],
                     'key2': ['K0', 'K1', 'K0', 'K1'],
                        'A': ['A0', 'A1', 'A2', 'A3'],
                        'B': ['B0', 'B1', 'B2', 'B3']})

right = pd.DataFrame({'key1': ['K0', 'K1', 'K1', 'K2'],
                               'key2': ['K0', 'K0', 'K0', 'K0'],
                                  'C': ['C0', 'C1', 'C2', 'C3'],
                                  'D': ['D0', 'D1', 'D2', 'D3']})
```

In [ ]:

```
display(pd.merge(left, right, how="outer", on=["key1", "key2"]))
display(pd.merge(left, right, on=['key1', 'key2']))
```

Join

In [82]:

```
left = pd.DataFrame({'A': ['A0', 'A1', 'A2'],
                     'B': ['B0', 'B1', 'B2']},
                      index=['K0', 'K1', 'K2'])

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

In [ ]:

```
display(left.join(right))
display(left.join(right, how='outer'))
display(right.join(left))
display(right.join(left, how='outer'))
```