* **Work with grouping and aggregate functions**
  + df.groupby(‘Col name’)
  + g[‘Col name’].count()
* **Utilize different types of joins (left, inner, outer, etc.)** 
  + df3=pd.merge(df1,df2,how='outer',on='Col name') #outer/inner/left/right
  + want to look for missing data again
* **including when and how to use a self-join**
  + when- you want to get a value from another row of same table
  + df.join(df.drop('m\_ids',1).set\_index('e\_ids'),on='m\_ids',rsuffix='e\_names')
* **Append multiple data sources (concat in Pandas)**
  + df=pd.concat([df1,df2],ignore\_index=True)
* **Filter data by multiple, complex conditions**
  + Use >, <, >=, <=, ==, <!= along with &, | (or)
  + df[df[‘Col name’]==True]
  + df[(df[‘Col name’]>=10)|(df[‘Col name’]<2)]
* **De-duplication**
  + df[df[‘Col name’].duplicated(keep=False)==True] #identify duplicates
  + df.drop([row#,row#],axis=0,inplace=True) #drop specific rows
  + df.drop\_duplicates(subset=’Col name’,keep=’first’,inplace=True) #keep first/last
  + df[‘key’=df1[‘Col Name’].astype(str)+’\_’+df1[‘Col Name’].astype(str) #create key
* **Sorting**
  + df.sort\_values(by=[‘Col Name’,’Col Name’],ascending=[True,False],na\_position=’first’,inplace=True) #sort df, na\_position is only for first column in the sort
  + sorted(“series”) #sort series
* **Handling missing/incomplete data**
  + df[df.isna().any(axis=1)] #find missing values
  + df.dropna(axis=0,subset=['Col name'],inplace=True) #drop rows w/ cols na
  + df['Col name'].fillna(df['Col name'].mean(),inplace=True)