Pandas

Reading the csv or xls/xlsx

`pd.read_csv()` -> returns **DataFrame**

`pd.read_excel("excel file")` install package `#pip3 install xlrd`

Exporting to csv or xls

`df.to_csv('file.csv', index=False)`

`df.to_excel('excel.xlsx')` install package `#pip3 install openpyxl`

`xldf = pd.read_excel("excel.xlsx")`

`xldf = pd.read_excel("excel.xlsx", index_col=0, sheet_name='Sheet1')`

Create DataFrame

- `df = pd.DataFrame(data={array of tuples}, columns=['','',''])`
- `df = pd.DataFrame({"col1":["val1", "val2", "val3"], "col2":[val1,val2,val3], "col3":[val1, val2, val3]})`

DataFrame's Attributes & Functions

- `df.shape` - returns the shape in the `tuple`
- `df.columns` - returns the columns array
- `df.index` - returns the index
- `df.head()` or `df.head(5)` -> returns the top rows
- `df.tail()` or `df.tail(5)` -> returns the bottom rows
- `df.describe()` - returns the statistical variables (mean/std/median & so on) of the numerical columnns
- `df.iloc[]` - indexing similar to the array indexinng
- `df.loc[]` - indexing done based on the **custom index** passed
- `g = df.group_by('col1')` - returns the iterable `col1` and it's `df`
- `g.max()` , `g.mean()` , `g.describe()` - operations on the numerical columns

Functions on DataFrame's Column

- `df['col'].mean()` - find the average of that column
- `df['col'].max()` - max value of that column

Filtering

- `df[df[col1] == df[col1].max()]['col2']` - fetch the `col2` data where `col1` has the max value

Operations

- `pd.concat([df1, df2], ignore_index=True)` - concate 2 dataframes

- `pd.concat([df1, df2], ignore_index=True, axis=1)` - concate 2 dataframes horizontally
- `pd.merge(df1, df2, on="col")` - inner join performed on 2 dataframes
- `pd.merge(df1, df2, on="col", how="right/left/outer/inner")` - other join options

Prepopulate

[ ] ↳ *2 cells hidden*

import

```
import pandas as pd
```

```
df = pd.read_csv("nyc_weather.csv")
df.head()
```

	EST	Temperature	DewPoint	Humidity	Sea Level PressureIn	VisibilityMiles	WindSpeedMPH	Pr
0	1/1/2016	38	23	52	30.03	10	8.0	
1	1/2/2016	36	18	46	30.02	10	7.0	
2	1/3/2016	40	21	47	29.86	10	8.0	
3	1/4/2016	25	9	44	30.05	10	9.0	
4	1/5/2016	20	-3	41	30.57	10	5.0	

```
#get max temp of the month
df['Temperature'].max()
```

```
50
```

```
#to know which day it rains
df[df['Events'] == 'Rain']['EST']
```

```
8      1/9/2016
9     1/10/2016
15    1/16/2016
26    1/27/2016
Name: EST, dtype: object
```

```
#average wind speed
df['WindSpeedMPH'].mean()
```

```
6.892857142857143
```

Data Frame

Like a data table (sheet)

```
#preloading
data="""day,temperature,windspeed,event
1/1/2017,32,6,Rain
1/2/2017,35,7,Sunny
1/3/2017,28,2,Snow
1/4/2017,24,7,Snow
1/5/2017,32,4,Rain
1/6/2017,31,2,Sunny"""
f = open("weather_data.csv", "w")
f.write(data)
f.close()
```

```
df = pd.read_csv("weather_data.csv")
#df = pd.read_excel("excel file") #pip3 install xlrd
df
```

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow
3	1/4/2017	24	7	Snow
4	1/5/2017	32	4	Rain
5	1/6/2017	31	2	Sunny

```
#construct using list of tuples
tuples = [("1/1/2017",32,6,"Rain"),
("1/2/2017",35,7,"Sunny"),
("1/3/2017",28,2,"Snow"),
("1/4/2017",24,7,"Snow"),
("1/5/2017",32,4,"Rain"),
("1/6/2017",31,2,"Sunny")]
df = pd.DataFrame(data=tuples, columns=['day', 'temperature', 'windspeed', 'event'])
df
```

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow
3	1/4/2017	24	7	Snow
4	1/5/2017	32	4	Rain
5	1/6/2017	31	2	Sunny

```
df.shape
df.shape
```

```
(6, 4)
```

```
df.head(5)
```

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow
3	1/4/2017	24	7	Snow
4	1/5/2017	32	4	Rain

```
df.tail(2)
```

	day	temperature	windspeed	event
4	1/5/2017	32	4	Rain
5	1/6/2017	31	2	Sunny

```
df.describe()
```

	temperature	windspeed
count	6.000000	6.000000
mean	30.333333	4.666667
std	3.829708	2.338090
min	24.000000	2.000000
25%	28.750000	2.500000
50%	31.500000	5.000000
75%	32.000000	6.750000
max	35.000000	7.000000

```
df[1:4]
```

	day	temperature	windspeed	event
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow
3	1/4/2017	24	7	Snow

```
df.columns
```

```
Index(['day', 'temperature', 'windspeed', 'event'], dtype='object')
```

```
df.index
```

```
RangeIndex(start=0, stop=6, step=1)
```

```
df[1:4][['day', 'temperature']]
```

	day	temperature
1	1/2/2017	35
2	1/3/2017	28
3	1/4/2017	24

```
df['temperature'].max()
```

```
35
```

```
# Get the data with max temperature
df[df['temperature'] == df['temperature'].max()]
```

	day	temperature	windspeed	event
1	1/2/2017	35	7	Sunny

```
df[df['temperature'] == df['temperature'].max()]['day'] == '1/2/2017'
```

```
1    True
Name: day, dtype: bool
```

```
df.to_csv('new_csv.csv', index=False)
!cat new_csv.csv
```

```
day,temperature,windspeed,event
1/1/2017,32,6,Rain
1/2/2017,35,7,Sunny
1/3/2017,28,2,Snow
1/4/2017,24,7,Snow
1/5/2017,32,4,Rain
1/6/2017,31,2,Sunny
```

```
df.to_excel('excel.xlsx') #pip3 install openpyxl
xldf = pd.read_excel("excel.xlsx")
print(xldf)
xldf = pd.read_excel("excel.xlsx", index_col=0, sheet_name='Sheet1')
print(xldf)
```

```
   Unnamed: 0       day  temperature  windspeed  event
0           0  1/1/2017           32          6   Rain
```

```
    1            1  1/2/2017          35          7  Sunny
    2            2  1/3/2017          28          2   Snow
    3            3  1/4/2017          24          7   Snow
    4            4  1/5/2017          32          4   Rain
    5            5  1/6/2017          31          2  Sunny
            day  temperature  windspeed  event
    0  1/1/2017           32          6   Rain
    1  1/2/2017           35          7  Sunny
    2  1/3/2017           28          2   Snow
    3  1/4/2017           24          7   Snow
    4  1/5/2017           32          4   Rain
    5  1/6/2017           31          2  Sunny
```

GroupBy

```
data = """day,city,temperature,windspeed,event
1/1/2017,new york,32,6,Rain
1/2/2017,new york,36,7,Sunny
1/3/2017,new york,28,12,Snow
1/4/2017,new york,33,7,Sunny
1/1/2017,mumbai,90,5,Sunny
1/2/2017,mumbai,85,12,Fog
1/3/2017,mumbai,87,15,Fog
1/4/2017,mumbai,92,5,Rain
1/1/2017,paris,45,20,Sunny
1/2/2017,paris,50,13,Cloudy
1/3/2017,paris,54,8,Cloudy
1/4/2017,paris,42,10,Cloudy"""
f = open("weather_data_cities.csv", "w")
f.write(data)
f.close()
```

```
df = pd.read_csv("weather_data_cities.csv")
df
```

	day	city	temperature	windspeed	event
0	1/1/2017	new york	32	6	Rain

```
group = df.groupby("city")
group
```

```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x7ff5aa30e7d0>
```

4 1/1/2017 mumbai 90 5 Sunny

```
for g, df in group:
  print(g)
  print(df)
  print("----------------------------------------------------------")
```

```
mumbai
        day    city  temperature  windspeed  event
4  1/1/2017  mumbai           90          5  Sunny
5  1/2/2017  mumbai           85         12    Fog
6  1/3/2017  mumbai           87         15    Fog
7  1/4/2017  mumbai           92          5   Rain
----------------------------------------------------------
new york
        day      city  temperature  windspeed  event
0  1/1/2017  new york           32          6   Rain
1  1/2/2017  new york           36          7  Sunny
2  1/3/2017  new york           28         12   Snow
3  1/4/2017  new york           33          7  Sunny
----------------------------------------------------------
paris
         day   city  temperature  windspeed   event
8   1/1/2017  paris           45         20   Sunny
9   1/2/2017  paris           50         13  Cloudy
10  1/3/2017  paris           54          8  Cloudy
11  1/4/2017  paris           42         10  Cloudy
----------------------------------------------------------
```

```
group.get_group('new york')
```

	day	city	temperature	windspeed	event
0	1/1/2017	new york	32	6	Rain
1	1/2/2017	new york	36	7	Sunny
2	1/3/2017	new york	28	12	Snow
3	1/4/2017	new york	33	7	Sunny

```
group.max()
```

	day	temperature	windspeed	event

```
group.mean()
```

	temperature	windspeed
city		
mumbai	88.50	9.25
new york	32.25	8.00
paris	47.75	12.75

```
group.describe()
```

	temperature								windspeed				
	count	mean	std	min	25%	50%	75%	max	count	mean	std	min	25%
city													
mumbai	4.0	88.50	3.109126	85.0	86.50	88.5	90.50	92.0	4.0	9.25	5.057997	5.0	5.00
new york	4.0	32.25	3.304038	28.0	31.00	32.5	33.75	36.0	4.0	8.00	2.708013	6.0	6.75
paris	4.0	47.75	5.315073	42.0	44.25	47.5	51.00	54.0	4.0	12.75	5.251984	8.0	9.50

Concatenate dataframes

```
india_weather = pd.DataFrame({"city":["Mumbai", "Delhi", "Bangalore"], "temperature":[32,45,30
india_weather
```

	city	temperature	humidity
0	Mumbai	32	80
1	Delhi	45	60
2	Bangalore	30	78

```
us_weather_tuples = [("new york", 68, 21), ("chicago", 65, 14), ("orlando", 75, 35)]
us_weather = pd.DataFrame(data=us_weather_tuples, columns=["city", "temperature", "humidity"])
us_weather
```

	city	temperature	humidity
0	new york	68	21
1	chicago	65	14
2	orlando	75	35

```
df = pd.concat([india_weather, us_weather], ignore_index=True)
df
```

	city	temperature	humidity
0	Mumbai	32	80
1	Delhi	45	60
2	Bangalore	30	78
3	new york	68	21
4	chicago	65	14
5	orlando	75	35

```
df = pd.concat([india_weather, us_weather], axis=1)
df
```

	city	temperature	humidity	city	temperature	humidity
0	Mumbai	32	80	new york	68	21
1	Delhi	45	60	chicago	65	14
2	Bangalore	30	78	orlando	75	35

Merging

```
temperature_df = pd.DataFrame({"city":["Mumbai", "Delhi", "Bangalore", "Hyderabad"], 'temperat
print(temperature_df)
humidity_df = pd.DataFrame({"city":["Mumbai", "Delhi", "Bangalore", "Chennai"], 'humidity':[68
print(humidity_df)
```

```
        city  temperature
0     Mumbai           32
1      Delhi           45
2  Bangalore           30
3  Hyderabad           40
        city  humidity
0     Mumbai        68
1      Delhi        65
2  Bangalore        75
3    Chennai        80
```

```
df = pd.merge(temperature_df, humidity_df, on="city")
df
```

	city	temperature	humidity

```
df = pd.merge(temperature_df, humidity_df, on="city", how="outer")
df
```

	city	temperature	humidity
0	Mumbai	32.0	68.0
1	Delhi	45.0	65.0
2	Bangalore	30.0	75.0
3	Hyderabad	40.0	NaN
4	Chennai	NaN	80.0

```
df = pd.merge(temperature_df, humidity_df, on="city", how="left")
df
```

	city	temperature	humidity
0	Mumbai	32	68.0
1	Delhi	45	65.0
2	Bangalore	30	75.0
3	Hyderabad	40	NaN

```
df = pd.merge(temperature_df, humidity_df, on="city", how="right")
df
```

	city	temperature	humidity
0	Mumbai	32.0	68
1	Delhi	45.0	65
2	Bangalore	30.0	75
3	Chennai	NaN	80

Numerical Indexing

```
df = pd.DataFrame(data=[1,2,3,4,5,6,7,8,9,10], index=[50,49,48,47,46,45,44,43,42,41], columns=
df
```

	num
50	1
49	2
48	3
47	4
46	5
45	6
44	7

```
df.loc[44] #by our given index
```

```
num    7
Name: 44, dtype: int64
```

```
df.iloc[3] #by the row
```

```
num    4
Name: 47, dtype: int64
```

```
df.iloc[:3] #row 0, 1, 2 excluding 3
```

	num
50	1
49	2
48	3

```
df.loc[:44] #all rows till index 44, iterating from top to bottom
```

	num
50	1
49	2
48	3
47	4
46	5
45	6
44	7

```
df.loc[44:42] #from index '44' to '42'
```

	num
44	7
43	8

```
df.iloc[:2] #all rows till row no 2 (exclusive)
```

	num
50	1
49	2

```
df.iloc[1:4] #from 1st to excluding 4th row
```

	num
49	2
48	3
47	4

✓ 0s completed at 1:06 PM