

Module 02

DataFrames

Data Science Developer

Outline

- Read Data from a CSV file
- Basic functionality
- Missing Value
- Data Aggregating

Read a Data Frame from CSV file

```
import pandas as pd
```

```
df = pd.read_csv('Salaries.csv')
```

Basic Functionality

Basic Functionality

functionality to explore the dataframe

- `.head` : return the first n rows
- `.tail` : return the last n rows
- `.info` : show summary of dataframe's metadata
- `.shape`, `.columns`, `.dtypes` : show some dataframe's metadata

descriptive statistics :

- `.describe` : quick summary of descriptive statistics
- `.mean`, `.median`, `.std`, `.min`, `.max`
- `.unique`, `.nunique`, `.value_counts`

.head and .tail

```
df.head()
```

	Id	EmployeeName	JobTitle	BasePay	OvertimePay	OtherPay	Benefits	TotalPay	TotalPayBenefits	Year	Notes	Agency	Status
0	1	NATHANIEL FORD	GENERAL MANAGER-METROPOLITAN TRANSIT AUTHORITY	167411.18	0.00	400184.25	NaN	567595.43	567595.43	2011	NaN	San Francisco	NaN
1	2	GARY JIMENEZ	CAPTAIN III (POLICE DEPARTMENT)	155966.02	245131.88	137811.38	NaN	538909.28	538909.28	2011	NaN	San Francisco	NaN
2	3	ALBERT PARDINI	CAPTAIN III (POLICE DEPARTMENT)	212739.13	106088.18	16452.60	NaN	335279.91	335279.91	2011	NaN	San Francisco	NaN
3	4	CHRISTOPHER CHONG	WIRE ROPE CABLE MAINTENANCE MECHANIC	77916.00	56120.71	198306.90	NaN	332343.61	332343.61	2011	NaN	San Francisco	NaN
4	5	PATRICK GARDNER	DEPUTY CHIEF OF DEPARTMENT,(FIRE DEPARTMENT)	134401.60	9737.00	182234.59	NaN	326373.19	326373.19	2011	NaN	San Francisco	NaN

```
df.tail()
```

	Id	EmployeeName	JobTitle	BasePay	OvertimePay	OtherPay	Benefits	TotalPay	TotalPayBenefits	Year	Notes	Agency	Status
148649	148650	Roy I Tillery	Custodian	0.0	0.0	0.00	0.0	0.00	0.00	2014	NaN	San Francisco	NaN
148650	148651	Not provided	Not provided	NaN	NaN	NaN	NaN	0.00	0.00	2014	NaN	San Francisco	NaN
148651	148652	Not provided	Not provided	NaN	NaN	NaN	NaN	0.00	0.00	2014	NaN	San Francisco	NaN
148652	148653	Not provided	Not provided	NaN	NaN	NaN	NaN	0.00	0.00	2014	NaN	San Francisco	NaN
148653	148654	Joe Lopez	Counselor, Log Cabin Ranch	0.0	0.0	-618.13	0.0	-618.13	-618.13	2014	NaN	San Francisco	NaN

.info

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 148654 entries, 0 to 148653  
Data columns (total 13 columns):  
Id                148654 non-null int64  
EmployeeName      148654 non-null object  
JobTitle          148654 non-null object  
BasePay           148045 non-null float64  
OvertimePay       148650 non-null float64  
OtherPay          148650 non-null float64  
Benefits          112491 non-null float64  
TotalPay          148654 non-null float64  
TotalPayBenefits  148654 non-null float64  
Year             148654 non-null int64  
Notes            0 non-null float64  
Agency          148654 non-null object  
Status           0 non-null float64  
dtypes: float64(8), int64(2), object(3)  
memory usage: 14.7+ MB
```

.shape, .columns, and .dtypes

```
df.shape
```

```
(148654, 13)
```

```
df.columns
```

```
Index(['Id', 'EmployeeName', 'JobTitle', 'BasePay', 'OvertimePay', 'OtherPay',  
      'Benefits', 'TotalPay', 'TotalPayBenefits', 'Year', 'Notes', 'Agency',  
      'Status'],  
      dtype='object')
```

```
df.dtypes
```

```
Id                int64  
EmployeeName      object  
JobTitle          object  
BasePay          float64  
OvertimePay      float64  
OtherPay         float64  
Benefits         float64  
TotalPay         float64  
TotalPayBenefits float64  
Year             int64  
Notes           float64  
Agency         object  
Status          float64  
dtype: object
```


Quick Summary of Descriptive Statistics

```
df.describe()
```

	Id	BasePay	OvertimePay	OtherPay	Benefits	TotalPay
count	148654.000000	148045.000000	148650.000000	148650.000000	112491.000000	148654.000000
mean	74327.500000	66325.448841	5066.059886	3648.767297	25007.893151	74768.321972
std	42912.857795	42764.635495	11454.380559	8056.601866	15402.215858	50517.005274
min	1.000000	-166.010000	-0.010000	-7058.590000	-33.890000	-618.130000
25%	37164.250000	33588.200000	0.000000	0.000000	11535.395000	36168.995000
50%	74327.500000	65007.450000	0.000000	811.270000	28628.620000	71426.610000
75%	111490.750000	94691.050000	4658.175000	4236.065000	35566.855000	105839.135000
max	148654.000000	319275.010000	245131.880000	400184.250000	96570.660000	567595.430000

```
df.describe(include = object)
```

	EmployeeName	JobTitle	Agency
count	148654	148654	148654
unique	110811	2159	1
top	Kevin Lee	Transit Operator	San Francisco
freq	13	7036	148654

Some Descriptive Statistics

```
print("average:",df['BasePay'].mean())  
print("median:",df['BasePay'].median())  
print("standar deviation:",df['BasePay'].std())  
print("minimum vale:",df['BasePay'].min())  
print("maximum value:",df['BasePay'].max())
```

average: 66325.44884050643

median: 65007.45

standar deviation: 42764.63549525958

minimum vale: -166.01

maximum value: 319275.01

Some Descriptive Statistics

```
df['JobTitle'].unique()
```

```
array(['GENERAL MANAGER-METROPOLITAN TRANSIT AUTHORITY',  
      'CAPTAIN III (POLICE DEPARTMENT)',  
      'WIRE ROPE CABLE MAINTENANCE MECHANIC', ..., 'Conversion',  
      'Cashier 3', 'Not provided'], dtype=object)
```

```
df['JobTitle'].nunique()
```

```
2159
```

```
df['JobTitle'].value_counts()
```

```
Transit Operator          7036  
Special Nurse            4389  
Registered Nurse        3736  
Public Svc Aide-Public Works 2518  
Police Officer 3        2421  
...  
MENTAL HEALTH HEARING OFFICER    1  
FORENSIC TOXICOLOGIST           1  
MAYORAL STAFF V                1  
Special Assistant 13            1  
CONFIDENTIAL SECRETARY TO DISTRICT ATTORNEY 1  
Name: JobTitle, Length: 2159, dtype: int64
```

Missing Value

Create DataFrame with NaN Values

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

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

```
In [3]: df
```

Out[3]:

	A	B	C
0	1.0	5.0	1
1	2.0	NaN	2
2	NaN	NaN	3

Dropna() method

```
In [4]: df.dropna()
```

Out[4]:

	A	B	C
0	1.0	5.0	1

```
In [5]: df.dropna(axis=1)
```

Out[5]:

	C
0	1
1	2
2	3

```
In [6]: df.dropna(thresh=2)
```

Out[6]:

	A	B	C
0	1.0	5.0	1
1	2.0	NaN	2

Fillna() method

```
In [7]: df.fillna(value='FILL VALUE')
```

Out[7]:

	A	B	C
0	1	5	1
1	2	FILL VALUE	2
2	FILL VALUE	FILL VALUE	3

```
In [10]: df['A'].fillna(value=df['A'].mean())
```

Out[10]:

0	1.0
1	2.0
2	1.5

Name: A, dtype: float64

```
In [11]: df
```

Out[11]:

	A	B	C
0	1.0	5.0	1
1	2.0	NaN	2
2	NaN	NaN	3

Data Aggregating

Create DataFrame

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

```
In [3]: df
```

```
Out[3]:
```

	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

Groupby() method

mean()

```
In [4]: df.groupby('Company')
```

```
Out[4]: <pandas.core.groupby.groupby.DataFrameGroupBy object at 0x000002654B5F12E8>
```

You can save this object as a new variable:

```
In [5]: by_comp = df.groupby("Company")
```

And then call aggregate methods off the object:

```
In [6]: by_comp.mean()
```

```
Out[6]:
```

Sales	
Company	
FB	296.5
GOOG	160.0
MSFT	232.0

```
In [7]: df.groupby('Company').mean()
```

```
Out[7]:
```

Sales	
Company	
FB	296.5
GOOG	160.0
MSFT	232.0

More Aggregate Methods

```
In [8]: by_comp.std()
```

```
Out[8]:
```

Sales	
Company	
FB	75.660426
GOOG	56.568542
MSFT	152.735065

```
In [9]: by_comp.min()
```

```
Out[9]:
```

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

```
In [10]: by_comp.max()
```

```
Out[10]:
```

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

```
In [11]: by_comp.count()
```

```
Out[11]:
```

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

More Aggregate Methods

```
In [12]: by_comp.describe()
```

Out[12]:

	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 [13]: by_comp.describe().transpose()
```

Out[13]:

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

More Aggregate Methods

```
In [15]: by_comp.describe().transpose()['GOOG']
```

```
Out[15]: 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 [17]: by_comp.describe().transpose()['GOOG'].loc['Sales'].loc['25%']
```

```
Out[17]: 140.0
```