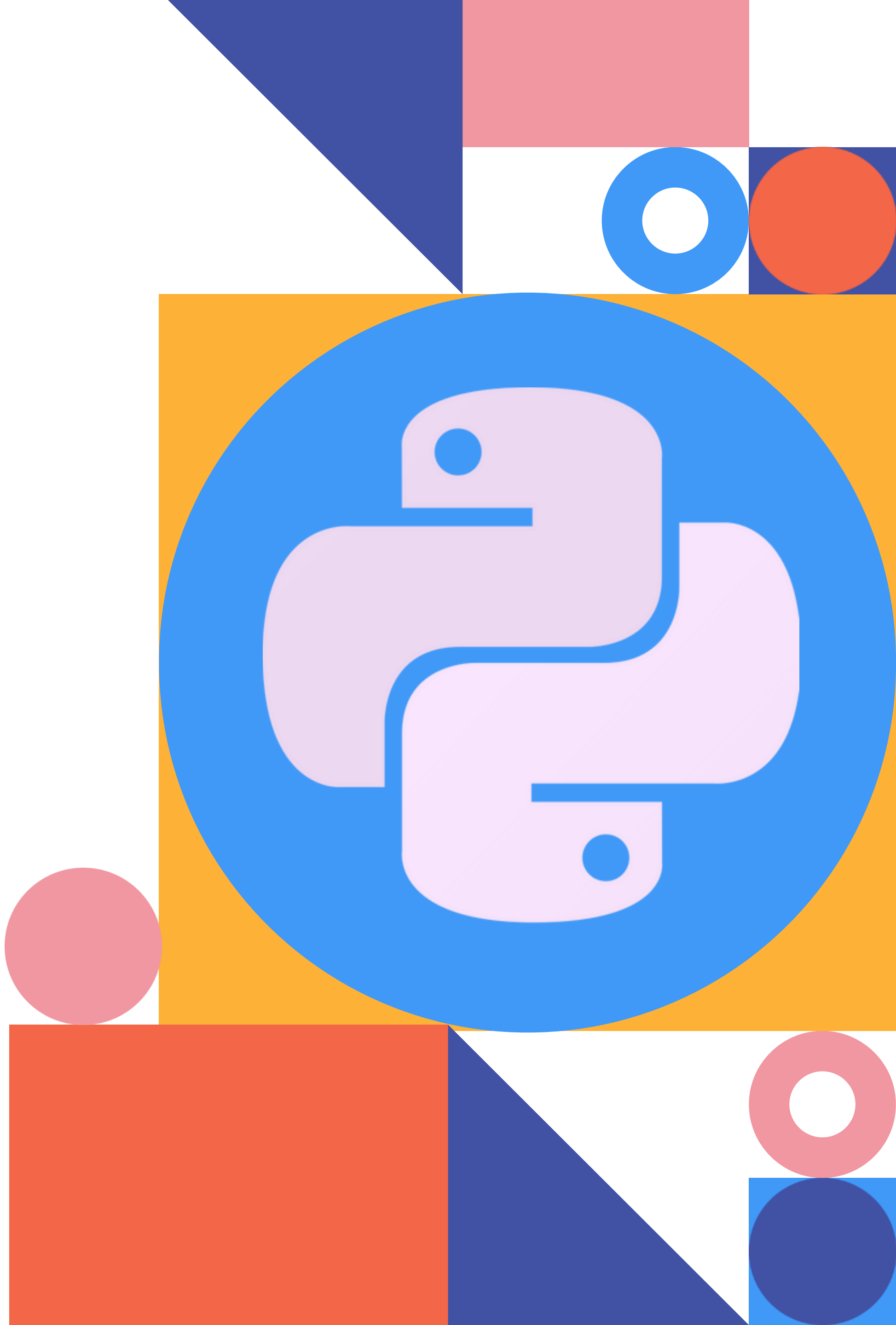
Data Analysis II

Yan-Fu Kuo

Dept. of Biomechatronics Engineering
National Taiwan University

→



Contents

01 Exploratory Data Analysis

02 Data Analysis Packages

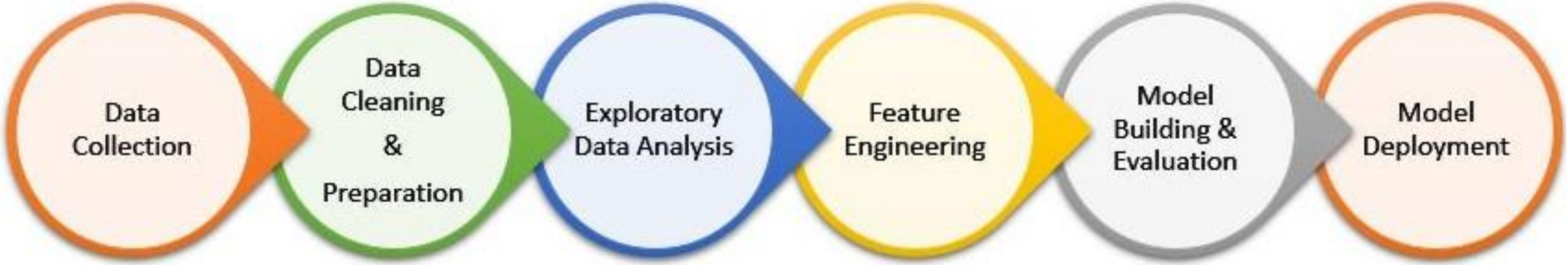
Pandas

01 Exploratory Data Analysis

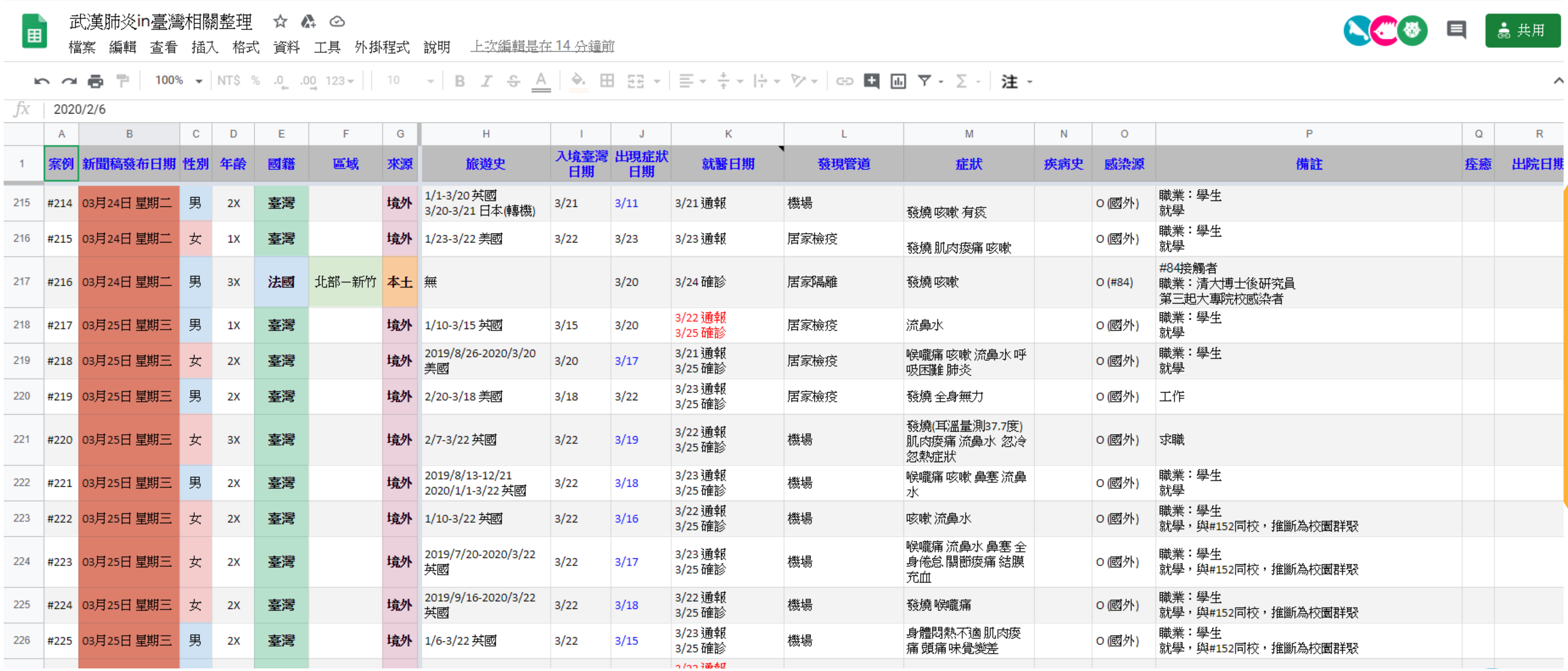
Exploratory Data Analysis (EDA)

An approach to analyzing data sets to summarize their main characteristics, often with visual methods

1. Understand your variables
2. Clean your dataset
3. Analyze relationships between variables


Data Collection
Data Cleaning & Preparation
Exploratory Data Analysis
Feature Engineering
Model Building & Evaluation
Model Deployment

COVID-19 in Taiwan

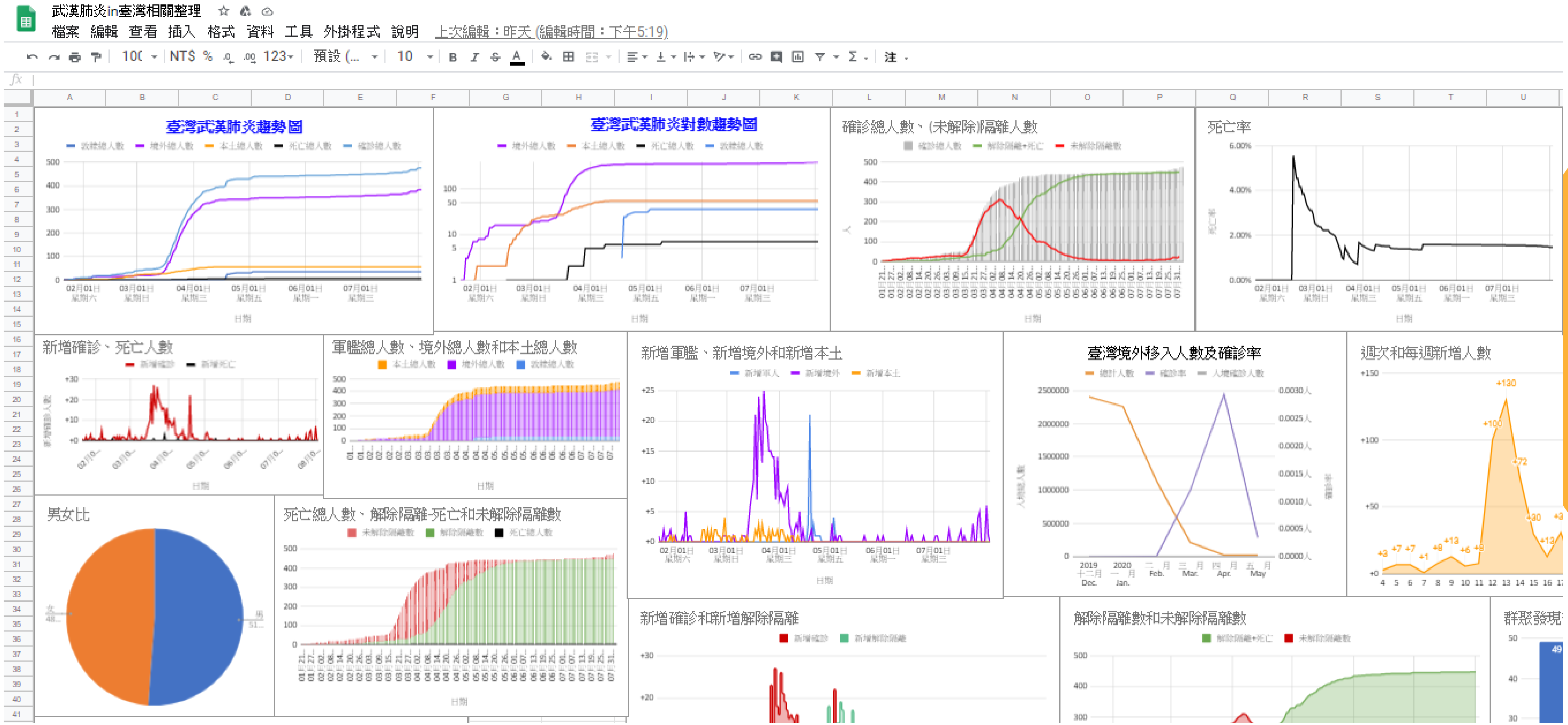


fx 2020/2/6

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	案例	新聞稿發布日期	性別	年齡	國籍	區域	來源	旅遊史	入境臺灣日期	出現症狀日期	就醫日期	發現管道	症狀	疾病史	感染源	備註	痊癒	出院日期
215	#214	03月24日 星期二	男	2X	臺灣		境外	1/1-3/20 英國 3/20-3/21 日本(轉機)	3/21	3/11	3/21 通報	機場	發燒 咳嗽 有痰		O (國外)	職業：學生 就學		
216	#215	03月24日 星期二	女	1X	臺灣		境外	1/23-3/22 美國	3/22	3/23	3/23 通報	居家檢疫	發燒 肌肉痠痛 咳嗽		O (國外)	職業：學生 就學		
217	#216	03月24日 星期二	男	3X	法國	北部ー新竹	本土	無		3/20	3/24 確診	居家隔離	發燒 咳嗽		O (#84)	#84接觸者 職業：清大博士後研究員 第三起大專院校感染者		
218	#217	03月25日 星期三	男	1X	臺灣		境外	1/10-3/15 英國	3/15	3/20	3/22 通報 3/25 確診	居家檢疫	流鼻水		O (國外)	職業：學生 就學		
219	#218	03月25日 星期三	女	2X	臺灣		境外	2019/8/26-2020/3/20 美國	3/20	3/17	3/21 通報 3/25 確診	居家檢疫	喉嚨痛 咳嗽 流鼻水 呼吸困難 肺炎		O (國外)	職業：學生 就學		
220	#219	03月25日 星期三	男	2X	臺灣		境外	2/20-3/18 美國	3/18	3/22	3/23 通報 3/25 確診	居家檢疫	發燒 全身無力		O (國外)	工作		
221	#220	03月25日 星期三	女	3X	臺灣		境外	2/7-3/22 英國	3/22	3/19	3/22 通報 3/25 確診	機場	發燒(耳溫量測37.7度) 肌肉痠痛 流鼻水 忽冷忽熱症狀		O (國外)	求職		
222	#221	03月25日 星期三	男	2X	臺灣		境外	2019/8/13-12/21 2020/1/1-3/22 英國	3/22	3/18	3/23 通報 3/25 確診	機場	喉嚨痛 咳嗽 鼻塞 流鼻水		O (國外)	職業：學生 就學		
223	#222	03月25日 星期三	女	2X	臺灣		境外	1/10-3/22 英國	3/22	3/16	3/22 通報 3/25 確診	機場	咳嗽 流鼻水		O (國外)	職業：學生 就學，與#152同校，推斷為校園群聚		
224	#223	03月25日 星期三	女	2X	臺灣		境外	2019/7/20-2020/3/22 英國	3/22	3/17	3/23 通報 3/25 確診	機場	喉嚨痛 流鼻水 鼻塞 全身倦怠 關節痠痛 結膜充血		O (國外)	職業：學生 就學，與#152同校，推斷為校園群聚		
225	#224	03月25日 星期三	女	2X	臺灣		境外	2019/9/16-2020/3/22 英國	3/22	3/18	3/22 通報 3/25 確診	機場	發燒 喉嚨痛		O (國外)	職業：學生 就學，與#152同校，推斷為校園群聚		
226	#225	03月25日 星期三	男	2X	臺灣		境外	1/6-3/22 英國	3/22	3/15	3/23 通報 3/25 確診	機場	身體悶熱不適 肌肉痠痛 頭痛 味覺變差		O (國外)	職業：學生 就學，與#152同校，推斷為校園群聚		

https://reurl.cc/QdjjaM

COVID-19 in Taiwan



武漢肺炎in臺灣相關整理
檔案 編輯 查看 插入 格式 資料 工具 外掛程式 說明
上次編輯：昨天 (編輯時間：下午5:19)
100 NT$ % .0 .00 123 預設 (... 10
臺灣武漢肺炎趨勢圖
敦睦總人數 境外總人數 本土總人數 死亡總人數 確診總人數
0 100 200 300 400 500
02月01日 星期六 03月01日 星期日 04月01日 星期三 05月01日 星期五 06月01日 星期一 07月01日 星期三
日期
臺灣武漢肺炎對數趨勢圖
境外總人數 本土總人數 死亡總人數 敦睦總人數
1 5 10 50 100
確診總人數、(未解除)隔離人數
確診總人數 解除隔離+死亡 未解除隔離數
死亡率
0.00% 2.00% 4.00% 6.00%
新增確診、死亡人數
新增確診 新增死亡
+0 +10 +20 +30
軍艦總人數、境外總人數和本土總人數
本土總人數 境外總人數 敦睦總人數
新增軍艦、新增境外和新增本土
新增軍人 新增境外 新增本土
+0 +5 +10 +15 +20 +25
臺灣境外移入人數及確診率
總計人數 確診率 入境確診人數
0 500000 1000000 1500000 2000000 2500000
0.0000人 0.0005人 0.0010人 0.0015人 0.0020人 0.0025人 0.0030人
2019 十二月 Dec. 2020 一月 Jan. 二月 Feb. 三月 Mar. 四月 Apr. 五月 May
週次和每週新增人數
+0 +50 +100 +150
+130 +100 +72 +30 +13 +3 +7 +7 +1 +8 +13 +6 +8
4 5 6 7 8 9 10 11 12 13 14 15 16
男女比
女 48 男 51
死亡總人數、解除隔離-死亡和未解除隔離數
未解除隔離數 解除隔離數 死亡總人數
新增確診和新增解除隔離
新增確診 新增解除隔離
解除隔離數和未解除隔離數
解除隔離+死亡 未解除隔離數
群聚發現
49

02 Pandas

Pandas

pandas

A fast and efficient **DataFrame** object for data manipulation with integrated indexing

Tools for **reading and writing data** between in-memory data structures and different formats: CSV and text files, Microsoft Excel, SQL databases, and the fast HDF5 format

...

Pandas Objects

Series

One-dimensional array of indexed data, or a specialization of a Python dictionary

Dataframe *

A generalization of a NumPy array, or an extended Python dictionary

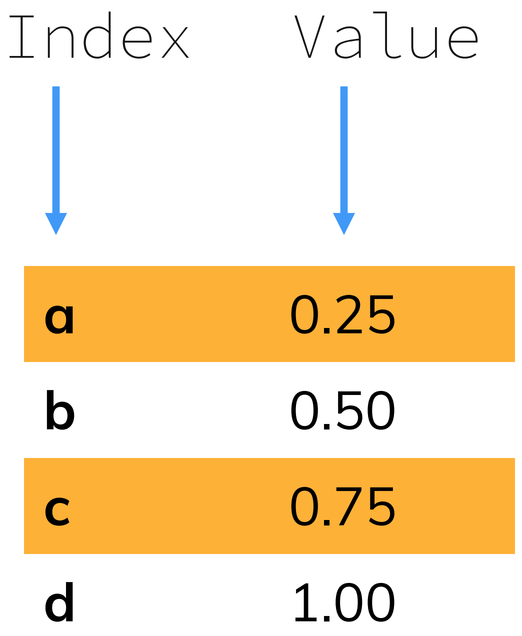
Index

An immutable array or as an ordered set

Series

```
import pandas as pd

data = pd.Series([0.25, 0.5, 0.75, 1.0],
                 index=['a', 'b', 'c', 'd'])
data['b'] #0.5
```

```
data = pd.Series([0.25, 0.5, 0.75, 1.0],
                 index=[2, 5, 3, 7])
data[3]
```



Index	Value
a	0.25
b	0.50
c	0.75
d	1.00

What is the value of data[3] ?

Series as Specialized Dictionary

```
population_dict = {'California': 38332521,
                   'Texas': 26448193,
                   'New York': 19651127,
                   'Florida': 19552860,
                   'Illinois': 12882135}
population = pd.Series(population_dict)

population_dict['California'] #38332521
population['California'] #38332521
```

```
population.sort_values(ascending=True)
```

California	38332521
Texas	16448193
New York	19651127
Florida	19552860
Illinois	12882135

Illinois	12882135
Texas	16448193
Florida	19552860
New York	19651127
California	38332521

Creating a `DataFrame`

- From a single Series object
- From a list of dictionaries
- From clipboard
- From a two-dimensional NumPy array
- From a NumPy structured array

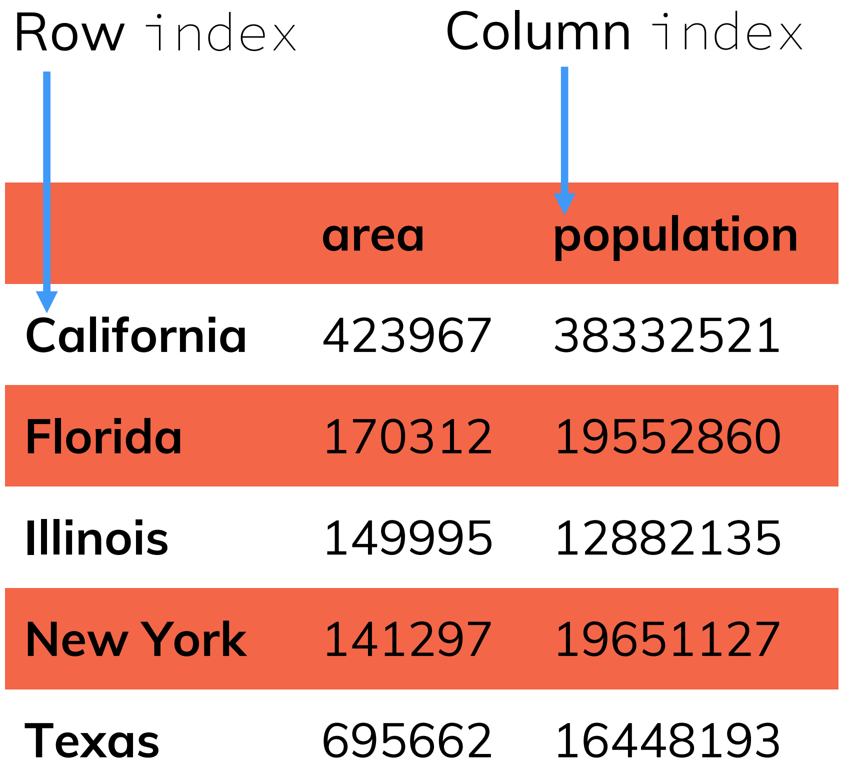
DataFrame from Series

```
area_dict = {'California': 423967, 'Texas': 695662,
             'New York': 141297, 'Florida': 170312,
             'Illinois': 149995}
area = pd.Series(area_dict)
```

```
states = pd.DataFrame({'population': population,
                       'area': area})
```

Series 2 Series 1

```
States['area']
```

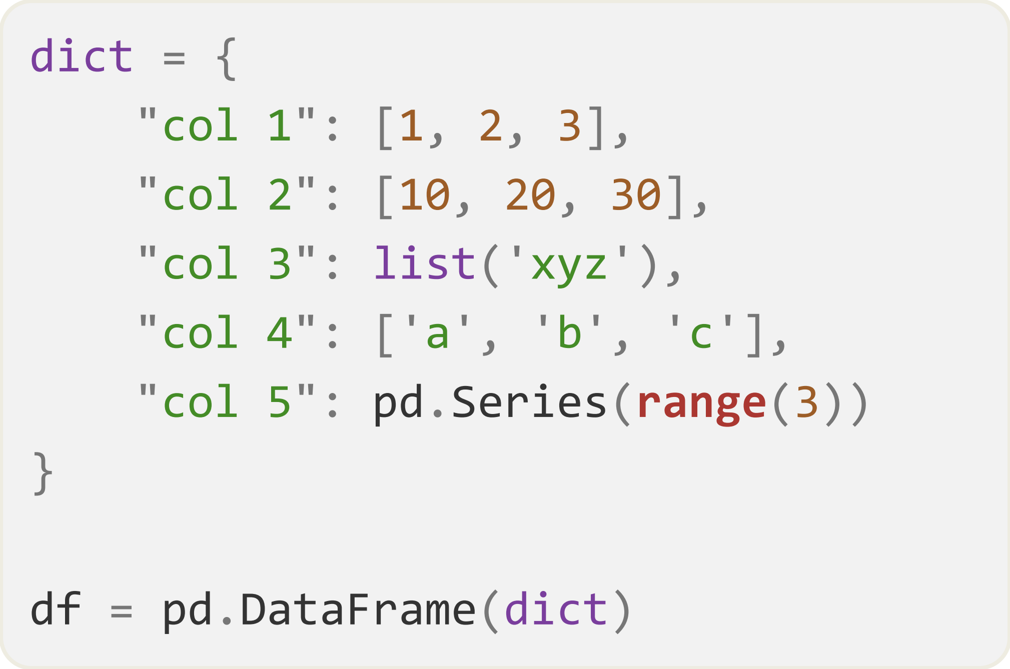


Row index

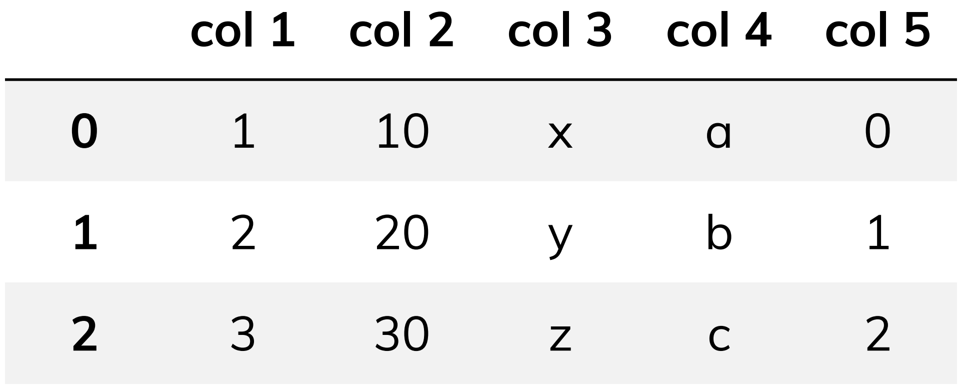
Column index

	area	population
California	423967	38332521
Florida	170312	19552860
Illinois	149995	12882135
New York	141297	19651127
Texas	695662	16448193

From a List of Dictionaries



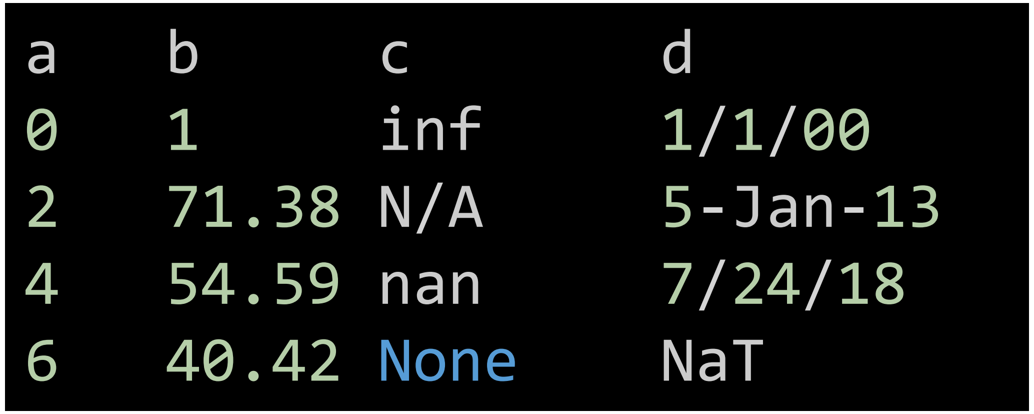
```python
dict = {
    "col 1": [1, 2, 3],
    "col 2": [10, 20, 30],
    "col 3": list('xyz'),
    "col 4": ['a', 'b', 'c'],
    "col 5": pd.Series(range(3))
}

df = pd.DataFrame(dict)
```



	col 1	col 2	col 3	col 4	col 5
0	1	10	x	a	0
1	2	20	y	b	1
2	3	30	z	c	2

From Clipboard

Copy the text:



```
a    b     c       d
0    1     inf     1/1/00
2    71.38 N/A     5-Jan-13
4    54.59 nan     7/24/18
6    40.42 None    NaT
```

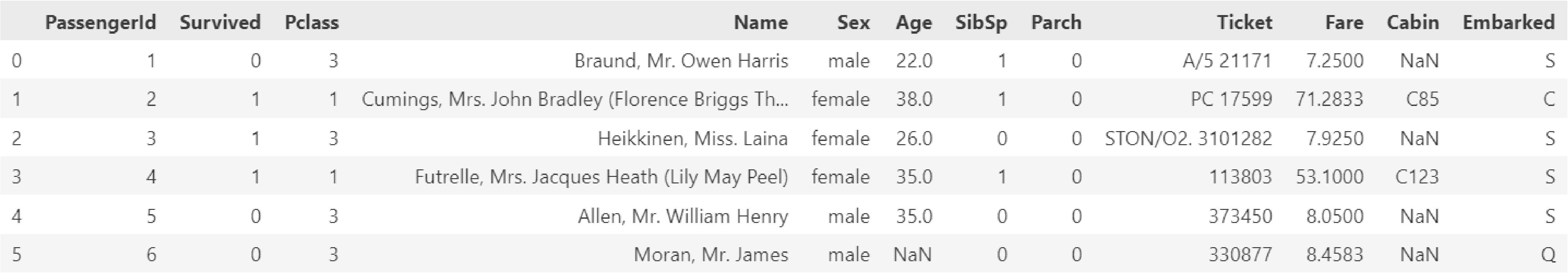
```
pd.read_clipboard()
```

From CSV

```
df = pd.read_csv('titanic.csv')
df.head(6)
```

Also check:

```
df.tail(6)
```



	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q

Titanic Machine Learning Competition

Titanic Dataset

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr.	male	22.0	1	0	A/5 21171	7.2500	**NaN**	S
1	2	1	1	Cumings, Mrs. John...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen,	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr.	male	35.0	0	0	373450	8.0500	**NaN**	S
...	...	...	...	...	...	...	...	...	...	...	...	...
888	889	0	3	Johnston, Miss. "	female	NaN	1	2	W./C. 6607	23.4500	**NaN**	S
889	890	1	1	Behr, Mr.	male	26.0	0	0	111369	30.0000	C148	C
890	891	0	3	Dooley, Mr. k	male	32.0	0	0	370376	7.7500	**NaN**	Q

df = read_csv()

```
pandas.read_csv(filepath_or_buffer, sep=',', delimiter=None, header='infer',
names=None, index_col=None, usecols=None, squeeze=False, prefix=None,
mangle_dupe_cols=True, dtype=None, engine=None, converters=None,
true_values=None, false_values=None, skipinitialspace=False, skiprows=None,
skipfooter=0, nrows=None, na_values=None, keep_default_na=True,
na_filter=True, verbose=False, skip_blank_lines=True, parse_dates=False,
infer_datetime_format=False, keep_date_col=False, date_parser=None,
dayfirst=False, cache_dates=True, iterator=False, chunksize=None,
compression='infer', thousands=None, decimal='.', lineterminator=None,
quotechar='"', quoting=0, doublequote=True, escapechar=None, comment=None,
encoding=None, dialect=None, error_bad_lines=True, warn_bad_lines=True,
delim_whitespace=False, low_memory=True, memory_map=False,
float_precision=None)
```

Memorize? Try SHIFT + TAB in jupyter

df.info()

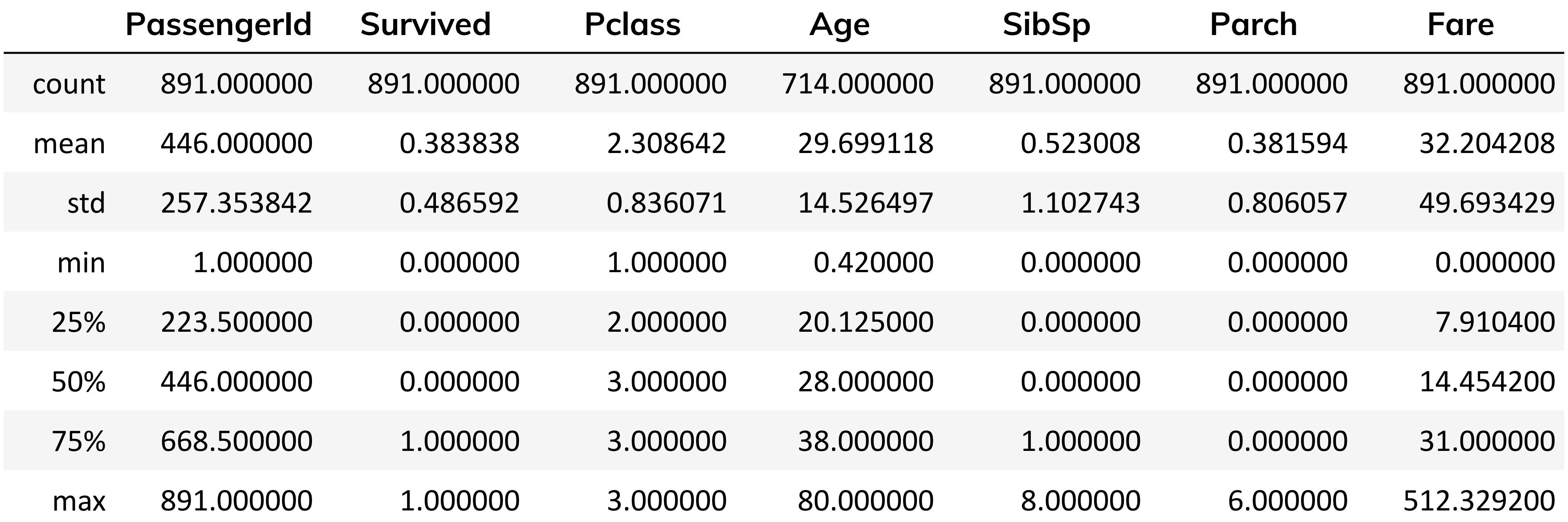
Q: What are types of the objects?

```
df.info(memory_usage="deep")
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
#     Column        Non-Null Count   Dtype
---   --------      ------------     -----
0     PassengerId   891 non-null     int64
1     Survived      891 non-null     int64
2     Pclass        891 non-null     int64
3     Name          891 non-null     object
4     Sex           891 non-null     object
5     Age           714 non-null     float64
6     SibSp         891 non-null     int64
7     Parch         891 non-null     int64
8     Ticket        891 non-null     object
9     Fare          891 non-null     float64
10    Cabin         204 non-null     object
11    Embarked      889 non-null     object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

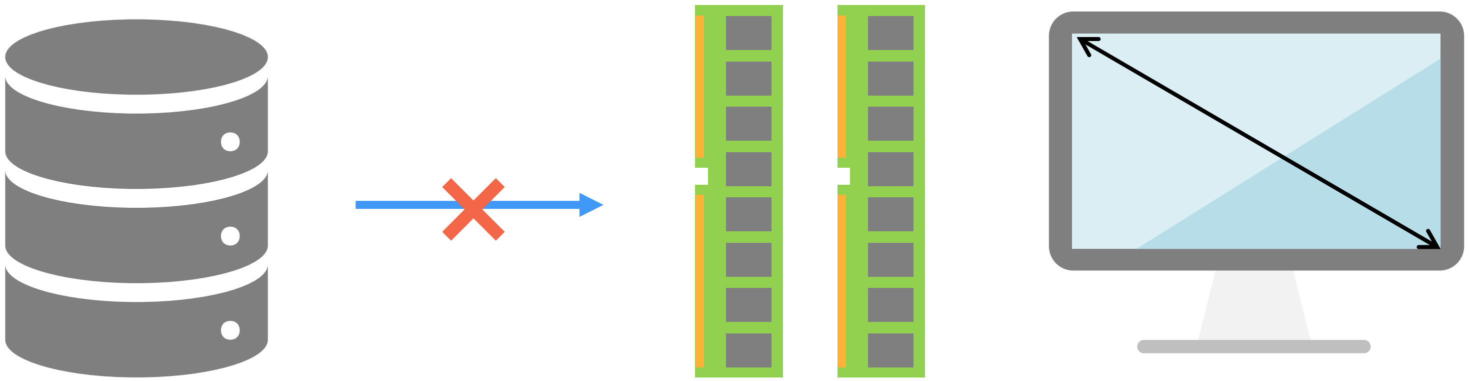
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
#     Column        Non-Null Count   Dtype
---   --------      ------------     -----
0     PassengerId   891 non-null     int64
1     Survived      891 non-null     int64
2     Pclass        891 non-null     int64
3     Name          891 non-null     object
4     Sex           891 non-null     object
5     Age           714 non-null     float64
6     SibSp         891 non-null     int64
7     Parch         891 non-null     int64
8     Ticket        891 non-null     object
9     Fare          891 non-null     float64
10    Cabin         204 non-null     object
11    Embarked      889 non-null     object
dtypes: float64(2), int64(5), object(5)
memory usage: 315.0 KB
```

df.describe()



	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

Dealing with Big Data

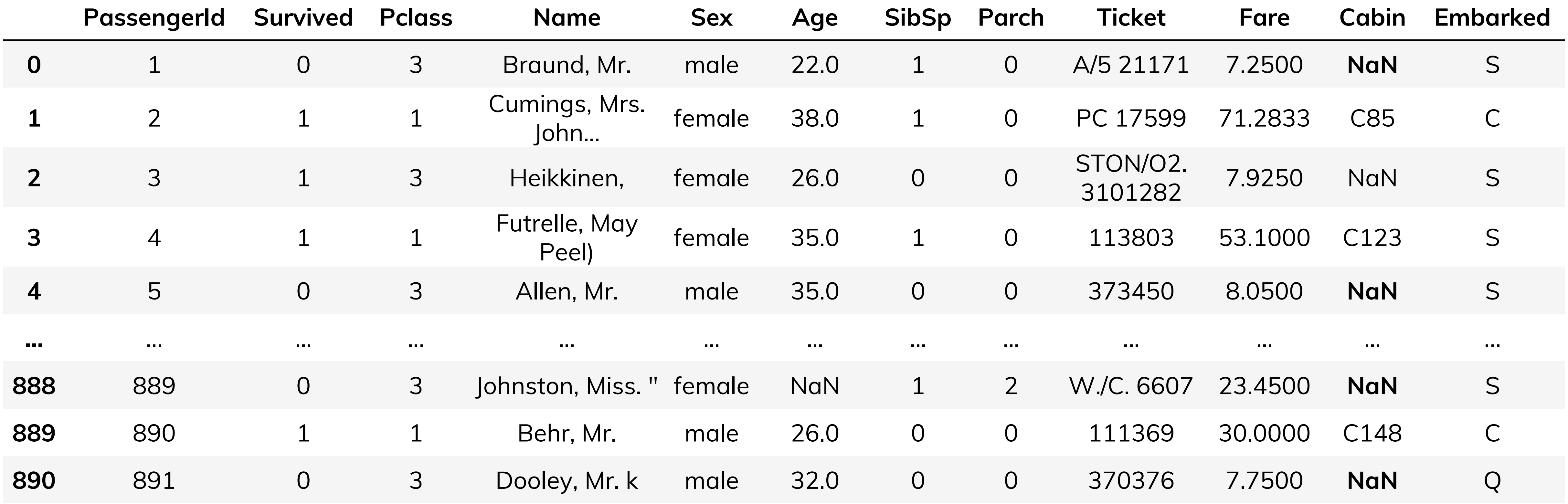


1PB Dataset

64GB ram

27” monitor

Titanic Dataset (Review)



	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr.	male	22.0	1	0	A/5 21171	7.2500	**NaN**	S
1	2	1	1	Cumings, Mrs. John...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen,	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr.	male	35.0	0	0	373450	8.0500	**NaN**	S
...	...	...	...	...	...	...	...	...	...	...	...	...
888	889	0	3	Johnston, Miss. "	female	NaN	1	2	W./C. 6607	23.4500	**NaN**	S
889	890	1	1	Behr, Mr.	male	26.0	0	0	111369	30.0000	C148	C
890	891	0	3	Dooley, Mr. k	male	32.0	0	0	370376	7.7500	**NaN**	Q

Reducing Memory Usage – 1

Using the Pandas **Category** data type

```
dtypes = {"Embarked": "category"}
cols = ['PassengerId', 'Name', 'Sex', 'Embarked']
df = pd.read_csv('titanic.csv',
                 dtype=dtypes, usecols=cols)
df.info(memory_usage="deep")
```

```
...
10    Cabin                204 non-null
11    Embarked             889 non-null
dtypes: float64(2), int64(5), object(5)
memory usage: 315.0 KB
```

↓

```
...
10    Cabin                204 non-null
11    Embarked             889 non-null
dtypes: float64(2), int64(5), object(5)
memory usage: 135.0 KB
```

https://pandas.pydata.org/docs/reference/api/pandas.read_csv.html

Reducing Memory Usage – 2

Batch processing

```
reader = pd.read_csv('titanic.csv', chunksize=4)

for _, df_partial in zip(range(2), reader):
    display(df_partial)
```

Q: How do you determine the batch size?

https://docs.python.org/3/library/functions.html#zip

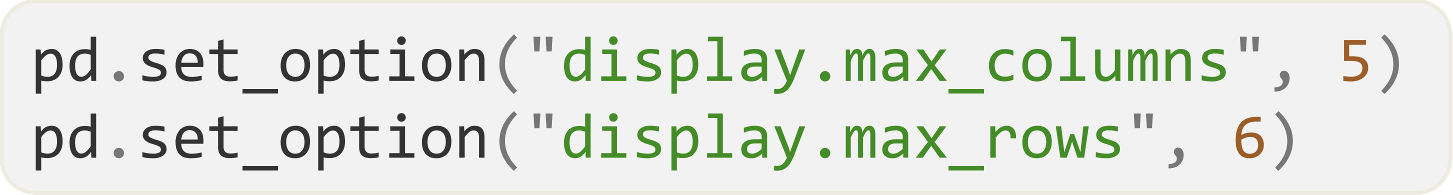
Customizing DataFrame Display

```python
print("Default display.max_colwidth:",
      pd.get_option("display.max_colwidth"))
```

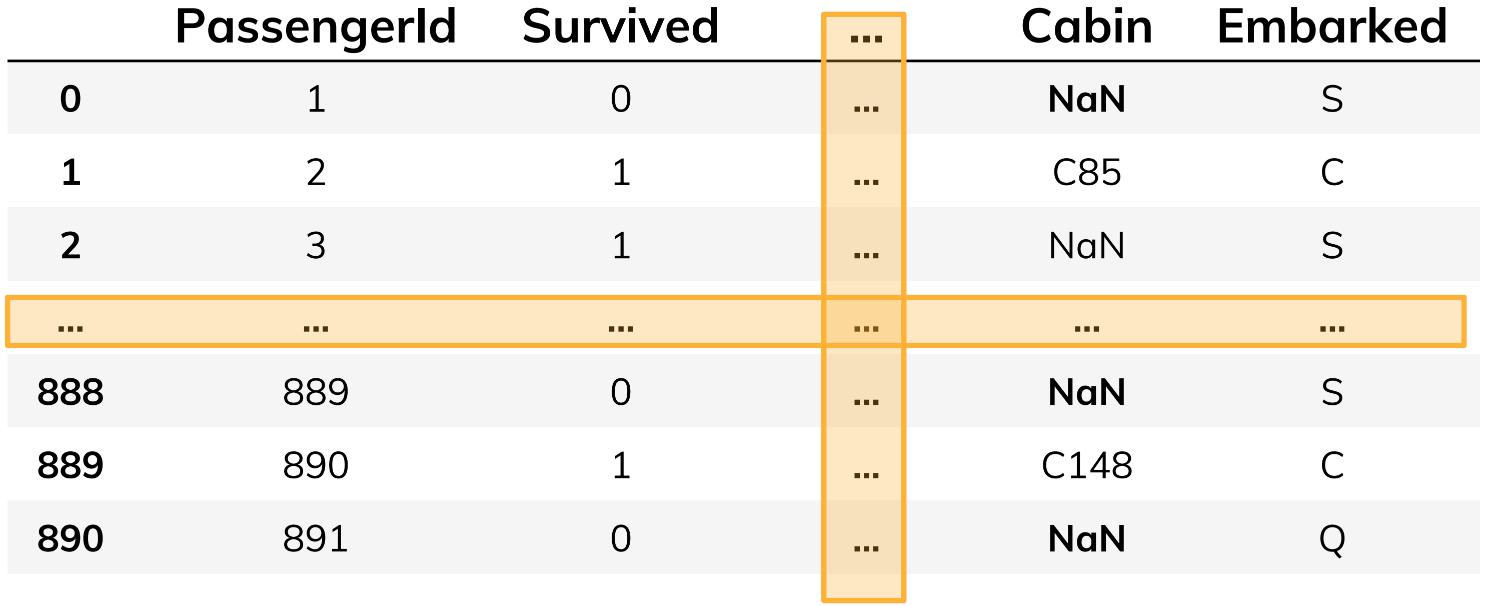
```
Default display.max_colwidth: 10
```

```python
pd.set_option("display.max_colwidth", 5)
```

Customizing DataFrame Display

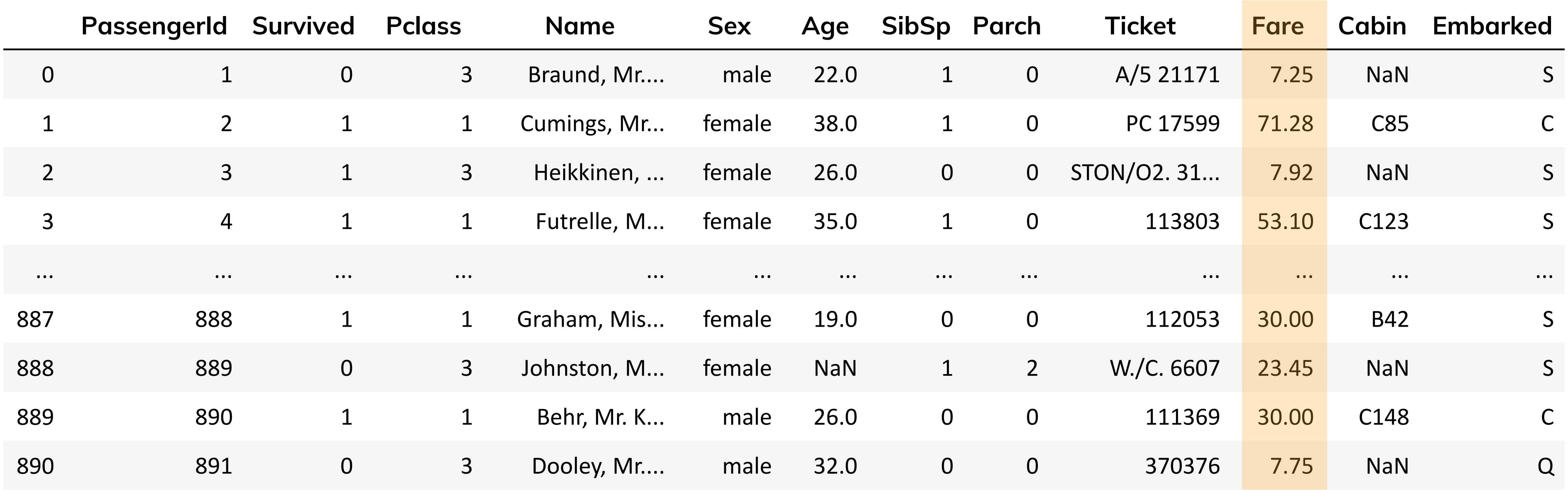


```
pd.set_option("display.max_columns", 5)
pd.set_option("display.max_rows", 6)
```



	PassengerId	Survived	...	Cabin	Embarked
0	1	0	...	NaN	S
1	2	1	...	C85	C
2	3	1	...	NaN	S
...	...	...	...	...	...
888	889	0	...	NaN	S
889	890	1	...	C148	C
890	891	0	...	NaN	Q

Exercise (5 mins)

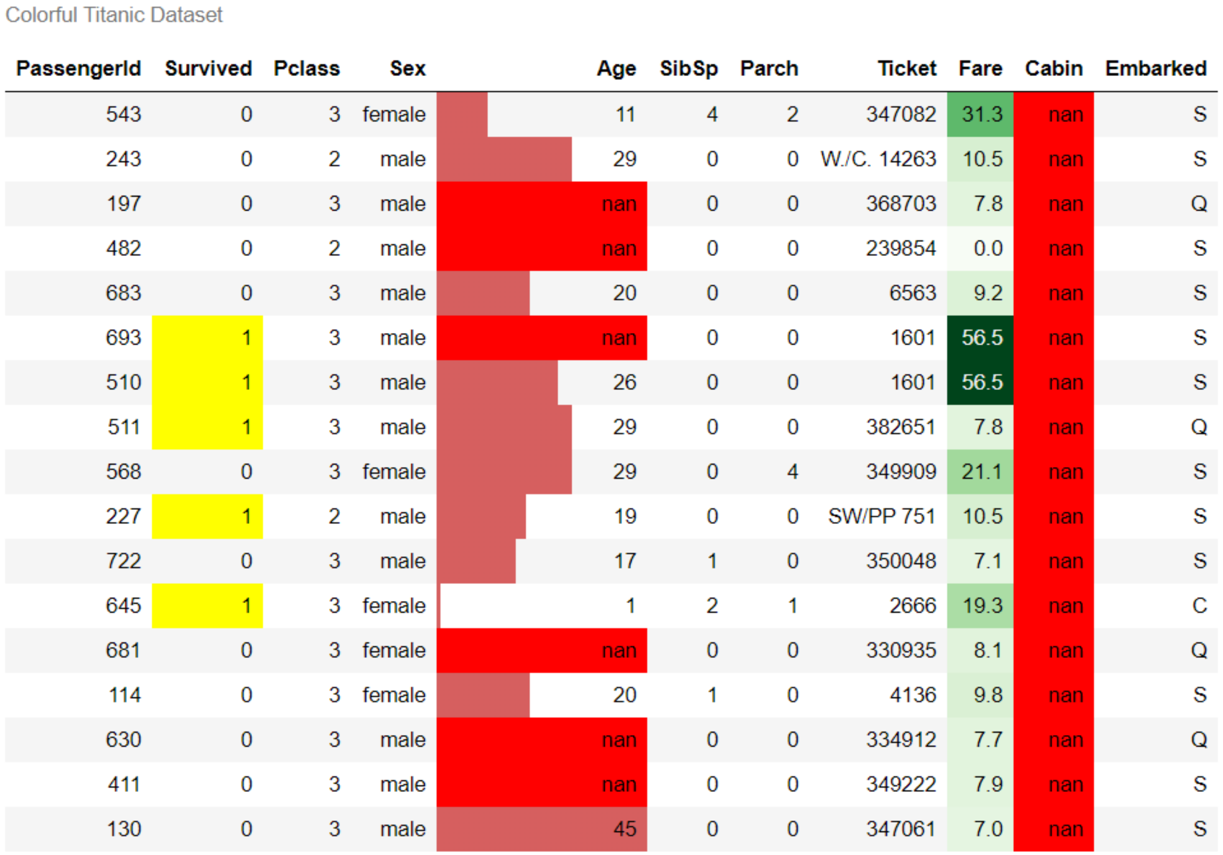


	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr....	male	22.0	1	0	A/5 21171	7.25	NaN	S
1	2	1	1	Cumings, Mr...	female	38.0	1	0	PC 17599	71.28	C85	C
2	3	1	3	Heikkinen, ...	female	26.0	0	0	STON/O2. 31...	7.92	NaN	S
3	4	1	1	Futrelle, M...	female	35.0	1	0	113803	53.10	C123	S
...	...	...	...	...	...	...	...	...	...	...	...	...
887	888	1	1	Graham, Mis...	female	19.0	0	0	112053	30.00	B42	S
888	889	0	3	Johnston, M...	female	NaN	1	2	W./C. 6607	23.45	NaN	S
889	890	1	1	Behr, Mr. K...	male	26.0	0	0	111369	30.00	C148	C
890	891	0	3	Dooley, Mr....	male	32.0	0	0	370376	7.75	NaN	Q

https://pandas.pydata.org/pandas-docs/stable/user_guide/options.html

DataFrame Styling

```
df_sample.style\
    .format('{:.1f}', subset='Fare')\
    .set_caption('Colorful Titanic Dataset')\
    .hide_index()\
    .bar('Age', vmin=0)\
    .highlight_max('Survived')\
    .background_gradient('Greens',
                         subset='Fare')\
    .highlight_null()
```

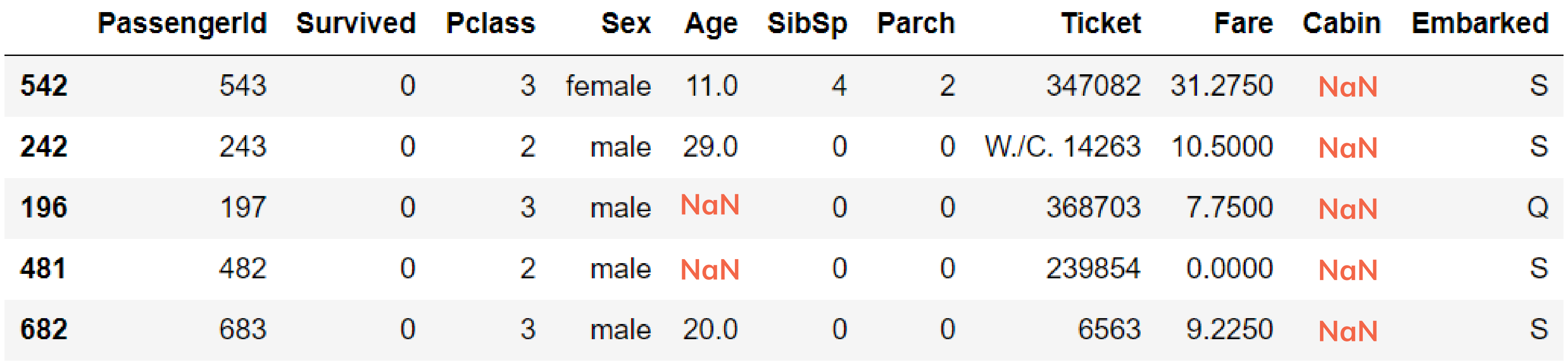


Colorful Titanic Dataset

PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
543	0	3	female	11	4	2	347082	31.3	nan	S
243	0	2	male	29	0	0	W./C. 14263	10.5	nan	S
197	0	3	male	nan	0	0	368703	7.8	nan	Q
482	0	2	male	nan	0	0	239854	0.0	nan	S
683	0	3	male	20	0	0	6563	9.2	nan	S
693	1	3	male	nan	0	0	1601	56.5	nan	S
510	1	3	male	26	0	0	1601	56.5	nan	S
511	1	3	male	29	0	0	382651	7.8	nan	Q
568	0	3	female	29	0	4	349909	21.1	nan	S
227	1	2	male	19	0	0	SW/PP 751	10.5	nan	S
722	0	3	male	17	1	0	350048	7.1	nan	S
645	1	3	female	1	2	1	2666	19.3	nan	C
681	0	3	female	nan	0	0	330935	8.1	nan	Q
114	0	3	female	20	1	0	4136	9.8	nan	S
630	0	3	male	nan	0	0	334912	7.7	nan	Q
411	0	3	male	nan	0	0	349222	7.9	nan	S
130	0	3	male	45	0	0	347061	7.0	nan	S

https://pandas.pydata.org/pandas-docs/stable/user_guide/style.html

Exercise – Data Cleansing (5 mins)



	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
542	543	0	3	female	11.0	4	2	347082	31.2750	NaN	S
242	243	0	2	male	29.0	0	0	W./C. 14263	10.5000	NaN	S
196	197	0	3	male	NaN	0	0	368703	7.7500	NaN	Q
481	482	0	2	male	NaN	0	0	239854	0.0000	NaN	S
682	683	0	3	male	20.0	0	0	6563	9.2250	NaN	S

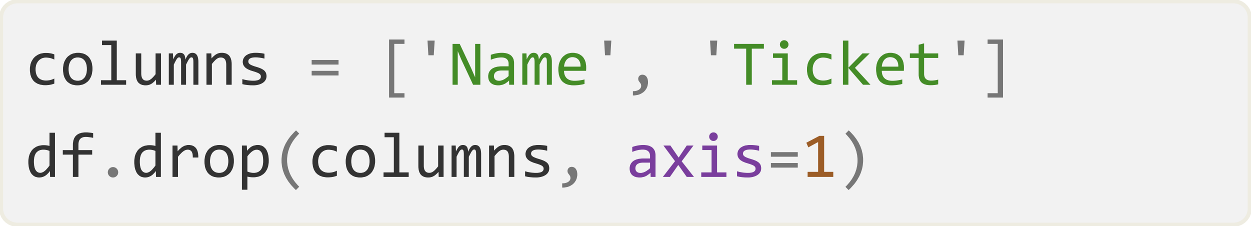
Try:

```
df.fillna(0)
df.fillna("Unknown")
df.Age.fillna(int(df.Age.mean()))
```

Q: Which on is better?

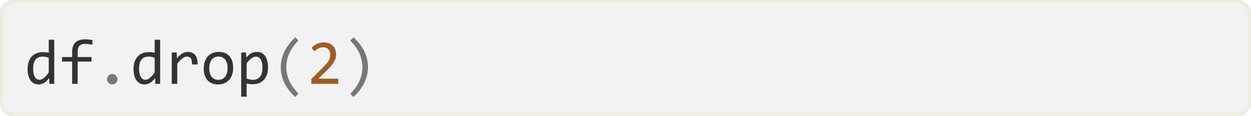
Drop (Useless) Data

Drop column data



```
columns = ['Name', 'Ticket']
df.drop(columns, axis=1)
```

Drop row data



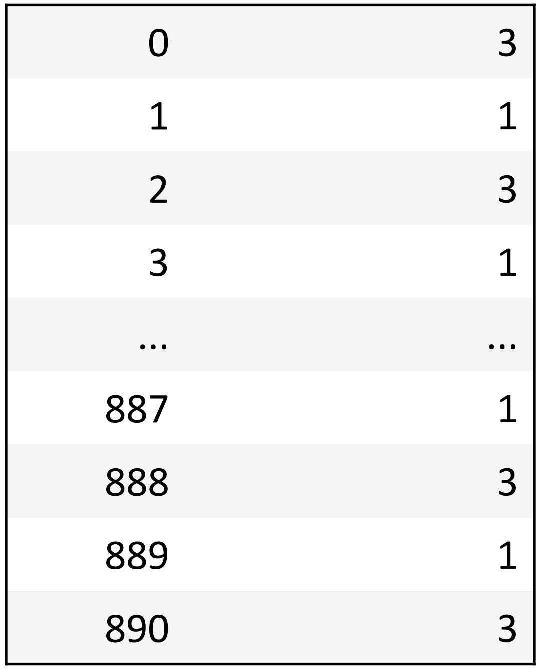
```
df.drop(2)
```

DataFrame Indexing

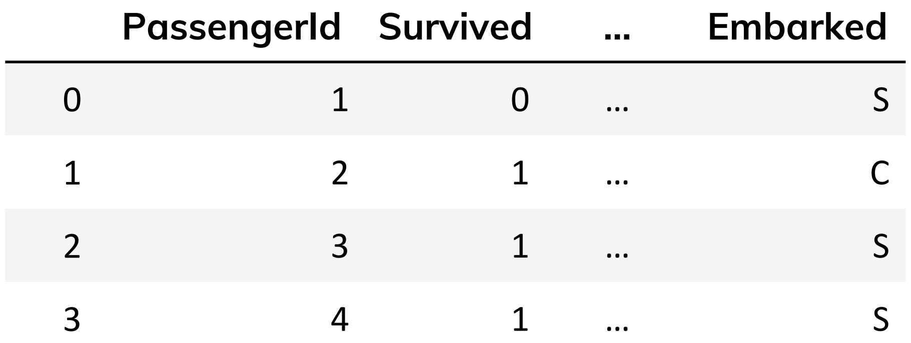
```
df.Pclass
```

```
df['Pclass']
```

```
df[0:4]
```



0	3
1	1
2	3
3	1
…	…
887	1
888	3
889	1
890	3



	PassengerId	Survived	…	Embarked
0	1	0	…	S
1	2	1	…	C
2	3	1	…	S
3	4	1	…	S

- Index refers to columns and slicing refers to rows
- Cannot do indexing on both columns and rows together!

Indexer

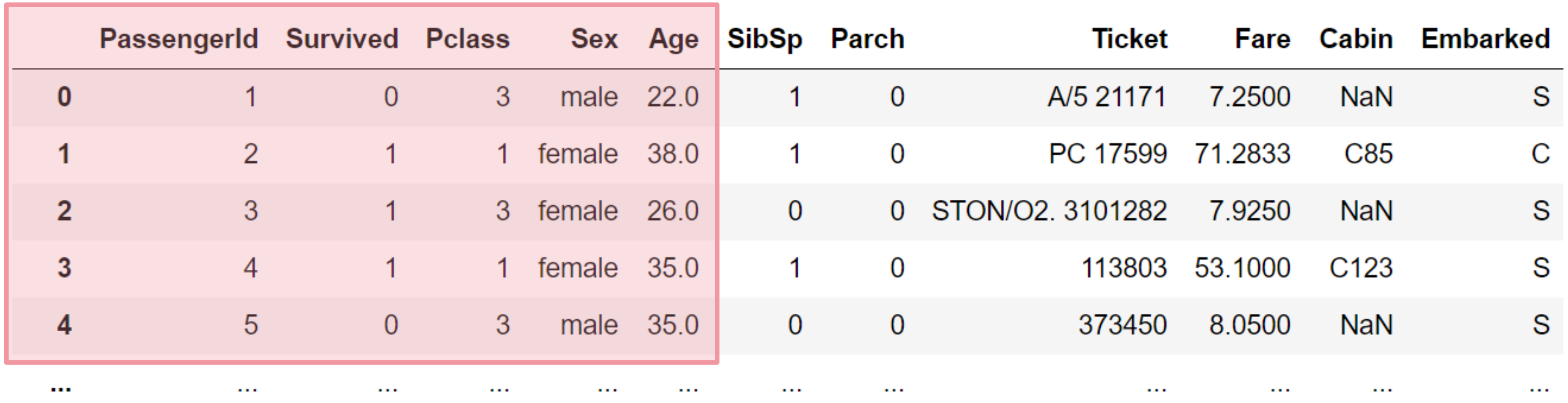
Allow indexing DataFrame as if it is a simple NumPy array.

- `df.loc` is primarily label based, but may also be used with a boolean array.
- `df.iloc` is primarily integer position based (from 0 to length-1 of the axis), but may also be used with a boolean array.

Data Selection with Indexer

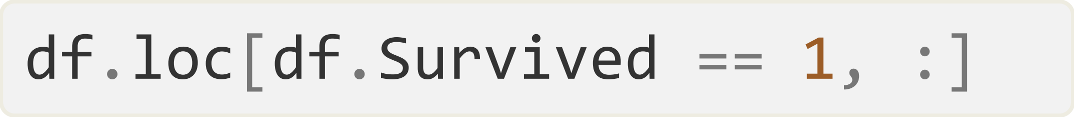
```
df.loc[0:4, 'PassengerId': 'Age']
```

```
df.iloc[0:5, 0:5]
```

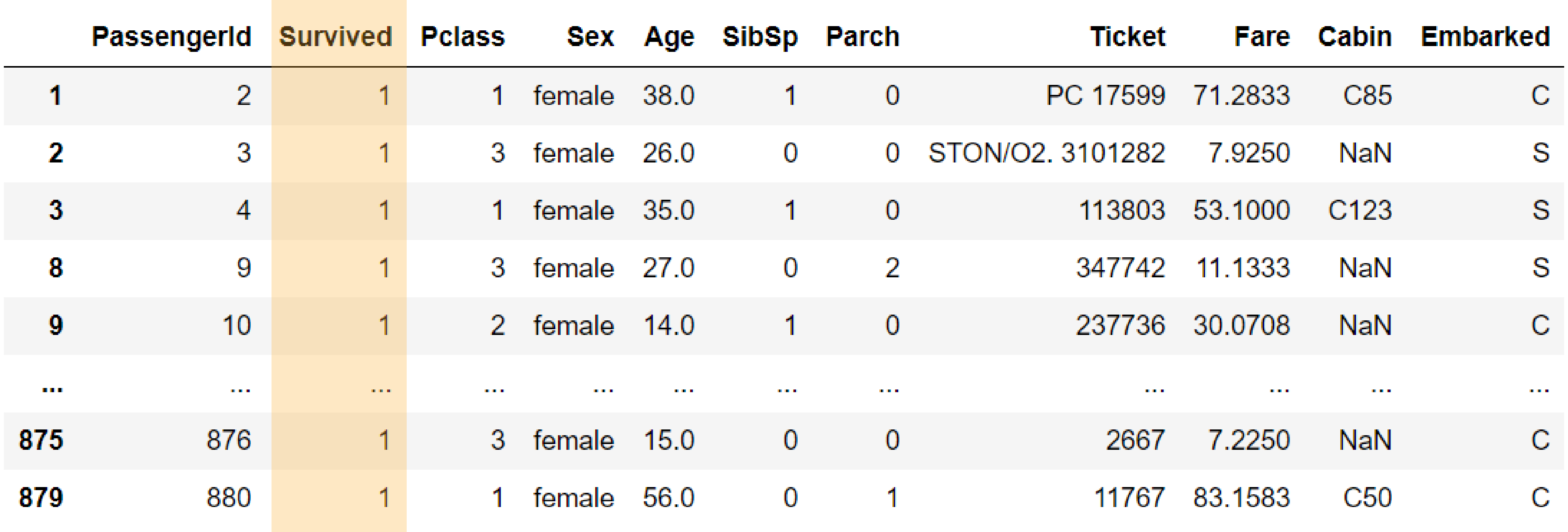


	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	male	35.0	0	0	373450	8.0500	NaN	S
...	...	...	...	...	...	...	...	...	...	...	...

Data Selection with Indexer



```
df.loc[df.Survived == 1, :]
```



	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
1	2	1	1	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	female	35.0	1	0	113803	53.1000	C123	S
8	9	1	3	female	27.0	0	2	347742	11.1333	NaN	S
9	10	1	2	female	14.0	1	0	237736	30.0708	NaN	C
...	...	...	...	...	...	...	...	...	...	...	...
875	876	1	3	female	15.0	0	0	2667	7.2250	NaN	C
879	880	1	1	female	56.0	0	1	11767	83.1583	C50	C

Data Selection – String and Num

Masking

```
female_and_age_under_20 = (df.Sex == 'female') & (df.Age < 20)
df[female_and_age_under_20]
```

Query

```
age = 70
df.query("Age > @age & Sex == 'male'")
```

Data Selection – String

```
df[df.Name.str.contains("Mr\.")]
```

```
tickets = ["SC/Paris 2123", "PC 17475"]
df[df.Ticket.isin(tickets)]
```

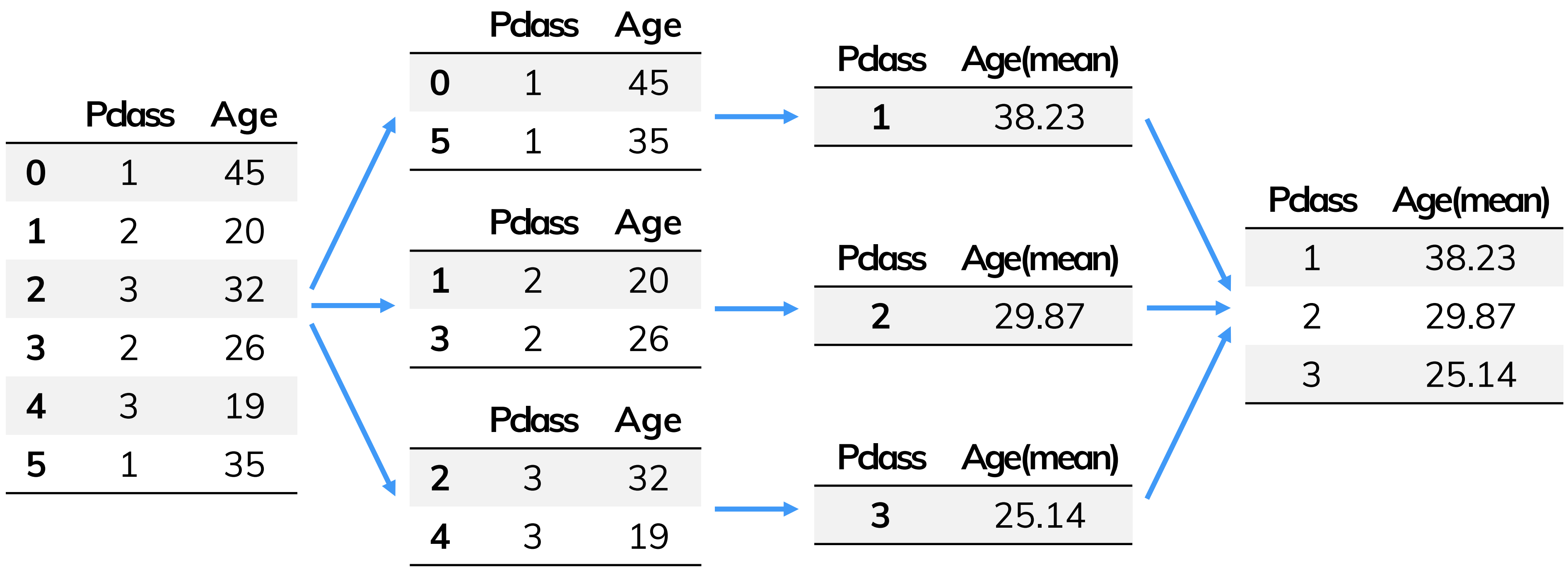
```
top_k = 3
top_tickets = df.Ticket.value_counts()[:top_k]
top_tickets.index
```

Exercise

Show the passenger info of the most frequently purchased class of tickets

GroupBy: Split, Apply, Combine

```
df.groupby("Pclass").Age.mean()
```



	Pclass	Age
0	1	45
1	2	20
2	3	32
3	2	26
4	3	19
5	1	35

	Pclass	Age
0	1	45
5	1	35

	Pclass	Age
1	2	20
3	2	26

	Pclass	Age
2	3	32
4	3	19

Pclass	Age(mean)
1	38.23

Pclass	Age(mean)
2	29.87

Pclass	Age(mean)
3	25.14

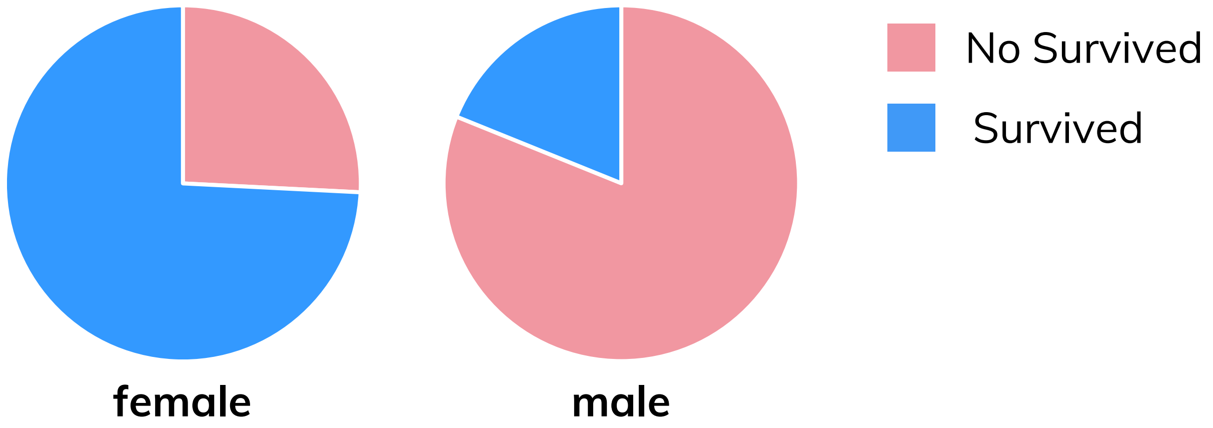
Pclass	Age(mean)
1	38.23
2	29.87
3	25.14

Insights

Does sex related to the survival rate in that accident?

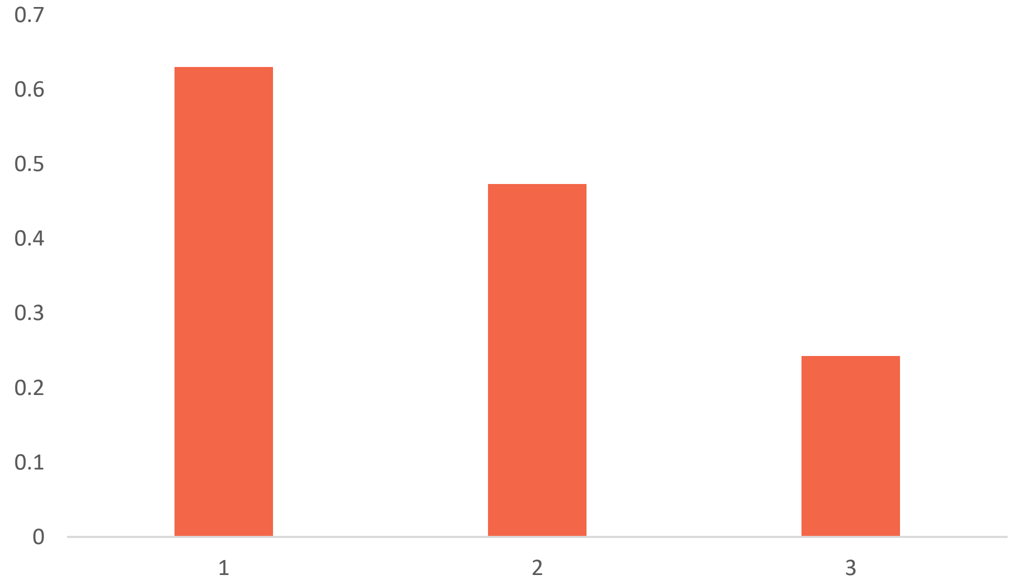
```
df.groupby(['Sex','Survived']).Sex.count()
```

```
Sex     Survived
female  0            81
        1           233
male    0           468
        1           109
Name: Sex, dtype: int64
```



Exercise

Does ticket class (`pclass`) related to the survival rate in that accident?



thank you!