```
array([ 1,  3,  5,  7,  9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33,
       35, 37, 39, 41, 43, 45, 47, 49, 51, 53, 55, 57, 59, 61, 63, 65, 67,
       69, 71, 73, 75, 77, 79, 81, 83, 85, 87, 89, 91, 93, 95, 97, 99])
```

```
np.zeros((3,3))
```

```
array([[0., 0., 0.],
       [0., 0., 0.],
       [0., 0., 0.]])
```

```
np.ones((3,3))
```

```
array([[1., 1., 1.],
       [1., 1., 1.],
       [1., 1., 1.]])
```

Intro to Pandas

- information of data frame
- select columns
- filter rows
- create new column
- aggregate + summarise
- value counts

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

```
# load csv data
df = pd.read_csv("data/Store.csv")
```

```
# preview dataset
df.tail(3)
```

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City	...
9991	9992	CA-2017-121258	2/26/2017	3/3/2017	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Costa Mesa	...
9992	9993	CA-2017-121258	2/26/2017	3/3/2017	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Costa Mesa	...
9993	9994	CA-2017-119914	5/4/2017	5/9/2017	Second Class	CC-12220	Chris Cortes	Consumer	United States	Westminster	...

3 rows × 21 columns

```
# information of data
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Row ID         9994 non-null   int64
 1   Order ID       9994 non-null   object
 2   Order Date     9994 non-null   object
 3   Ship Date      9994 non-null   object
 4   Ship Mode      9994 non-null   object
 5   Customer ID    9994 non-null   object
 6   Customer Name  9994 non-null   object
 7   Segment        9994 non-null   object
 8   Country        9994 non-null   object
 9   City           9994 non-null   object
 10  State          9994 non-null   object
 11  Postal Code    9994 non-null   int64
 12  Region         9994 non-null   object
 13  Product ID     9994 non-null   object
```

```
 14  Category       9994 non-null   object
 15  Sub-Category   9994 non-null   object
 16  Product Name   9994 non-null   object
 17  Sales          9994 non-null   float64
 18  Quantity       9994 non-null   int64
 19  Discount       9994 non-null   float64
 20  Profit         9994 non-null   float64
dtypes: float64(3), int64(3), object(15)
memory usage: 1.6+ MB
```

```
df.shape # attribute
```

```
(9994, 21)
```

```
df.describe()
```

	Row ID	Postal Code	Sales	Quantity	Discount	Profit
count	9994.000000	9994.000000	9994.000000	9994.000000	9994.000000	9994.000000
mean	4997.500000	55190.379428	229.858001	3.789574	0.156203	28.656896
std	2885.163629	32063.693350	623.245101	2.225110	0.206452	234.260108
min	1.000000	1040.000000	0.444000	1.000000	0.000000	-6599.978000
25%	2499.250000	23223.000000	17.280000	2.000000	0.000000	1.728750
50%	4997.500000	56430.500000	54.490000	3.000000	0.200000	8.666500
75%	7495.750000	90008.000000	209.940000	5.000000	0.200000	29.364000
max	9994.000000	99301.000000	22638.480000	14.000000	0.800000	8399.976000

```
df.columns
```

```
Index(['Row ID', 'Order ID', 'Order Date', 'Ship Date', 'Ship Mode',
       'Customer ID', 'Customer Name', 'Segment', 'Country', 'City', 'State',
       'Postal Code', 'Region', 'Product ID', 'Category', 'Sub-Category',
       'Product Name', 'Sales', 'Quantity', 'Discount', 'Profit'],
      dtype='object')
```

```
# clean column names
cols = df.columns
clean_cols = [col.lower().replace(" ", "_").replace("-","_")  for col in cols]
df.columns = clean_cols
```

```
df.columns
```

```
Index(['row_id', 'order_id', 'order_date', 'ship_date', 'ship_mode',
       'customer_id', 'customer_name', 'segment', 'country', 'city', 'state',
       'postal_code', 'region', 'product_id', 'category', 'sub_category',
       'product_name', 'sales', 'quantity', 'discount', 'profit'],
      dtype='object')
```

```
# select columns
df['segment'].head()
```

```
0     Consumer
1     Consumer
2    Corporate
3     Consumer
4     Consumer
Name: segment, dtype: object
```

```
# create new column
selected_cols = ['order_id', 'segment', 'sales', 'state', 'city']

df2 = df[selected_cols]

df2['tax'] = df2['sales'] * 0.25

df2.head(3)
```

```
<ipython-input-192-89565a0c35e7>:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing,
```

```
# remove columns
df2 = df2.drop(['order_id','city'], axis=1)
df2.head()
```

	segment	sales	state	tax
0	Consumer	261.9600	Kentucky	65.490000
1	Consumer	731.9400	Kentucky	182.985000
2	Corporate	14.6200	California	3.655000
3	Consumer	957.5775	Florida	239.394375
4	Consumer	22.3680	Florida	5.592000

```
# filter data
df[ (df['category'] == 'Furniture') & (df['segment'] == 'Home Office') ][['customer_name', 'segment', 'category']].head(20)
```

	customer_name	segment	category
38	Steve Nguyen	Home Office	Furniture
39	Steve Nguyen	Home Office	Furniture
66	Paul Stevenson	Home Office	Furniture
96	Parhena Norris	Home Office	Furniture
124	Alan Dominguez	Home Office	Furniture
128	Lindsay Shagiari	Home Office	Furniture
129	Lindsay Shagiari	Home Office	Furniture
146	Maureen Gastineau	Home Office	Furniture
189	Mark Packer	Home Office	Furniture
192	Mark Packer	Home Office	Furniture
231	Christopher Schild	Home Office	Furniture
232	Christopher Schild	Home Office	Furniture
234	Christopher Schild	Home Office	Furniture
244	Dianna Wilson	Home Office	Furniture
292	Nick Zandusky	Home Office	Furniture
317	Nathan Mautz	Home Office	Furniture
462	Tanja Norvell	Home Office	Furniture
463	Tanja Norvell	Home Office	Furniture
467	Joni Sundaresam	Home Office	Furniture
485	Michelle Tran	Home Office	Furniture

```
# query() method
result = df.query("category == 'Furniture' and segment == 'Consumer' ")[['customer_name', 'segment', 'category']].tail(10)
```

```
# export csv file
result.to_csv("data/output_store.csv")
```

```
!ls data
```

```
chinook.db  food.txt  hotel.csv  output.csv  output_store.csv  Store.csv
```

```
# value counts
count_segment = df['segment'].value_counts(normalize=True).reset_index()
count_segment.to_csv("data/segment.csv")
```

```
# statistics (aggregate functions)
# numpy statistics
total_sales = df['sales'].sum()
avg_sales = df['sales'].mean()
std_quantity = df['quantity'].std()
```

```
print(f"Total Sales: {round(total_sales,2)}")
print(avg_sales, std_quantity)
```

```
Total Sales: 2297200.86
229.85800083049833 2.2251096911414
```

```
median_sales = np.median(df['sales'])
print(median_sales)
```

```
54.489999999999995
```

```
# groupby + aggregate
df.groupby('segment')['sales'].agg(['sum', 'mean', 'count', 'min', 'max'])
```

segment	sum	mean	count	min	max
Consumer	1.161401e+06	223.733644	5191	0.444	13999.96
Corporate	7.061464e+05	233.823300	3020	0.556	17499.95
Home Office	4.296531e+05	240.972041	1783	0.990	22638.48

```
result = df.groupby(['state','segment'])[['sales', 'profit']]\
    .agg(['sum','mean'])\
    .reset_index()

result.head()
# result.to_csv("data/request_data_18Nov2022.csv")
```

	state	segment	sales		profit	
			sum	mean	sum	mean
0	Alabama	Consumer	7537.540	301.501600	1711.0939	68.443756
1	Alabama	Corporate	10969.380	391.763571	3648.3846	130.299450
2	Alabama	Home Office	1003.720	125.465000	427.3468	53.418350
3	Arizona	Consumer	16424.422	149.312927	-1423.0527	-12.936843
4	Arizona	Corporate	11736.322	170.091623	-788.9158	-11.433562

```
# OKAY : )
```

API

API => Application Programming Interface

Request-Response cycle

import `requests`

```
import requests
import time
import pandas as pd
```

```
url2 = "https://swapi.dev/api/people/2"
response = requests.get(url2)
response.status_code
```

```
200
```

```
result2=response.json()
result2['height']
```

```
'167'
```

```
names = []
heights = []
masses = []

for i in range(1,11):
    url = f"https://swapi.dev/api/people/{i}"
    resp = requests.get(url)
```