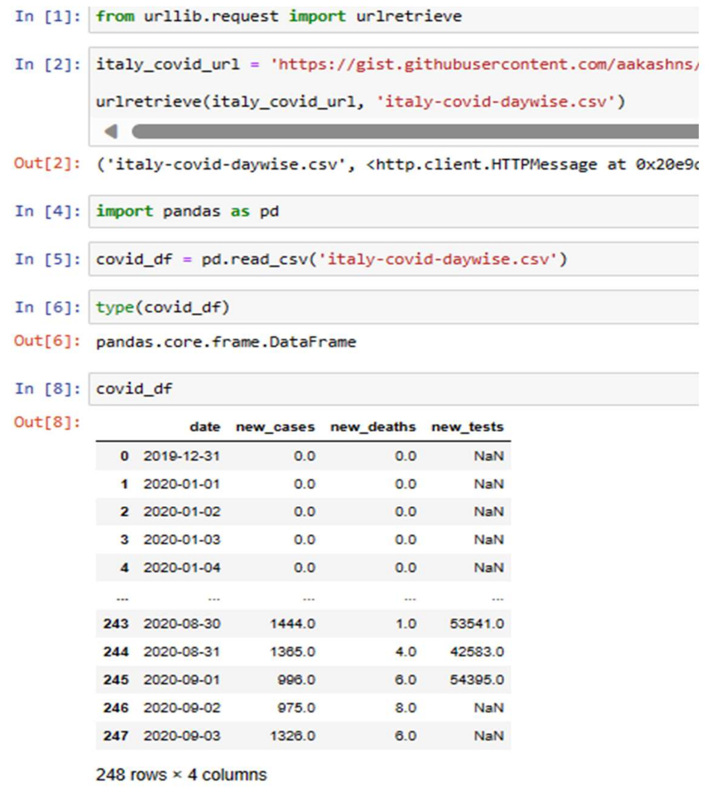


```
In [1]: from urllib.request import urlretrieve
```

```
In [2]: italy_covid_url = 'https://gist.githubusercontent.com/aakashns/

        urlretrieve(italy_covid_url, 'italy-covid-daywise.csv')
```

```
Out[2]: ('italy-covid-daywise.csv', <http.client.HTTPMessage at 0x20e9c
```

```
In [4]: import pandas as pd
```

```
In [5]: covid_df = pd.read_csv('italy-covid-daywise.csv')
```

```
In [6]: type(covid_df)
```

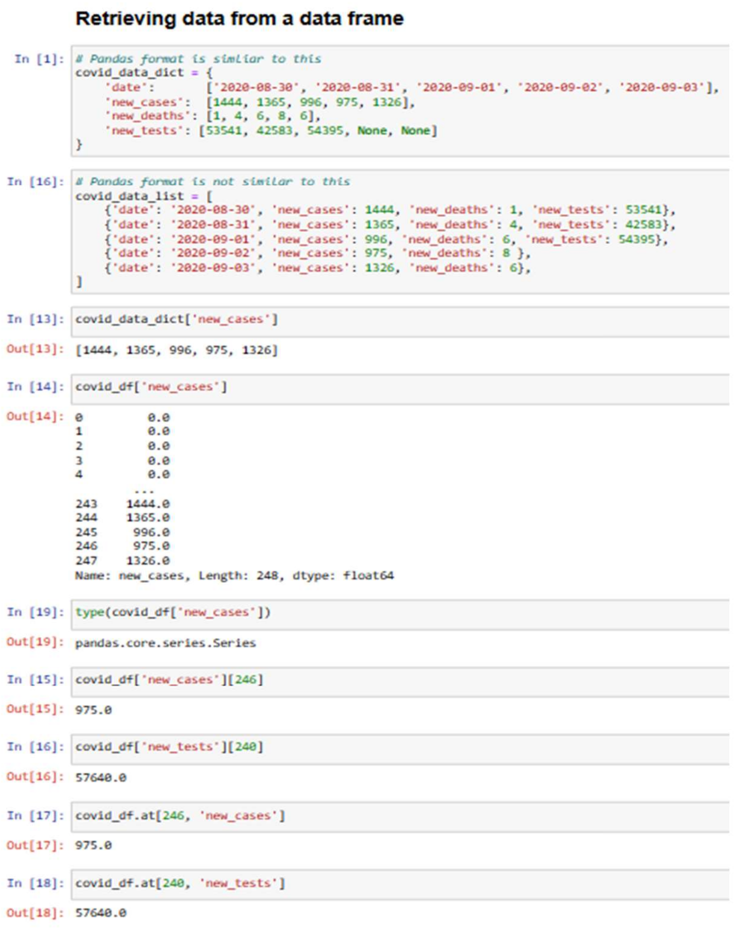
```
Out[6]: pandas.core.frame.DataFrame
```

```
In [8]: covid_df
```

Out[8]:

	date	new_cases	new_deaths	new_tests
0	2019-12-31	0.0	0.0	NaN
1	2020-01-01	0.0	0.0	NaN
2	2020-01-02	0.0	0.0	NaN
3	2020-01-03	0.0	0.0	NaN
4	2020-01-04	0.0	0.0	NaN
...	...	...	...	...
243	2020-08-30	1444.0	1.0	53541.0
244	2020-08-31	1365.0	4.0	42583.0
245	2020-09-01	996.0	6.0	54395.0
246	2020-09-02	975.0	8.0	NaN
247	2020-09-03	1326.0	6.0	NaN

248 rows × 4 columns



Retrieving data from a data frame

```
In [1]: # Pandas format is simliar to this
        covid_data_dict = {
            'date':       ['2020-08-30', '2020-08-31', '2020-09-01', '2020-09-02', '2020-09-03'],
            'new_cases':  [1444, 1365, 996, 975, 1326],
            'new_deaths': [1, 4, 6, 8, 6],
            'new_tests': [53541, 42583, 54395, None, None]
        }
```

```
In [16]: # Pandas format is not similar to this
         covid_data_list = [
             {'date': '2020-08-30', 'new_cases': 1444, 'new_deaths': 1, 'new_tests': 53541},
             {'date': '2020-08-31', 'new_cases': 1365, 'new_deaths': 4, 'new_tests': 42583},
             {'date': '2020-09-01', 'new_cases': 996, 'new_deaths': 6, 'new_tests': 54395},
             {'date': '2020-09-02', 'new_cases': 975, 'new_deaths': 8 },
             {'date': '2020-09-03', 'new_cases': 1326, 'new_deaths': 6},
         ]
```

```
In [13]: covid_data_dict['new_cases']

Out[13]: [1444, 1365, 996, 975, 1326]
```

```
In [14]: covid_df['new_cases']

Out[14]: 0         0.0
         1         0.0
         2         0.0
         3         0.0
         4         0.0
                 ...
         243    1444.0
         244    1365.0
         245     996.0
         246     975.0
         247    1326.0
         Name: new_cases, Length: 248, dtype: float64
```

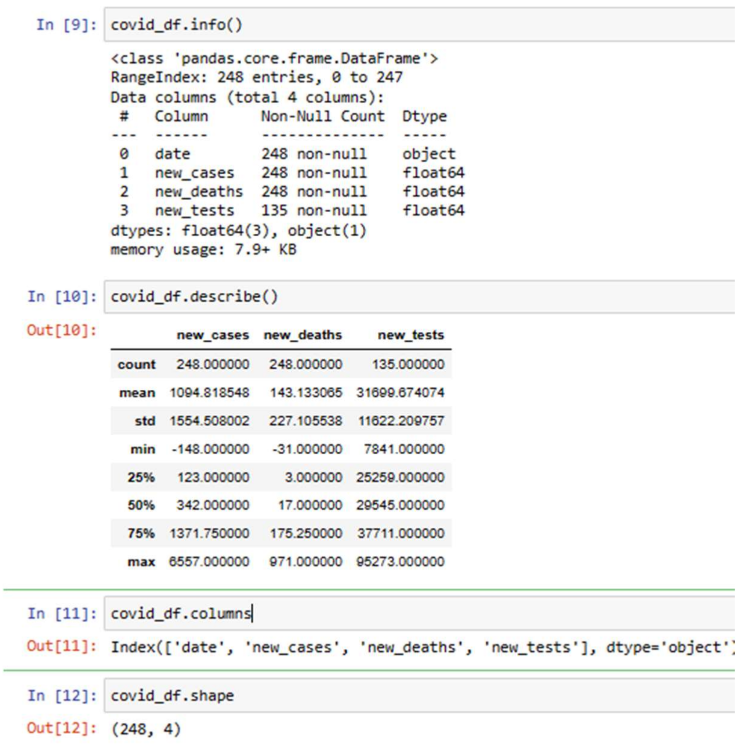
```
In [19]: type(covid_df['new_cases'])

Out[19]: pandas.core.series.Series
```

```
In [15]: covid_df['new_cases'][246]

Out[15]: 975.0
```

```
In [16]: covid_df['new_tests'][240]

Out[16]: 57640.0
```

```
In [17]: covid_df.at[246, 'new_cases']

Out[17]: 975.0
```

```
In [18]: covid_df.at[240, 'new_tests']

Out[18]: 57640.0
```



```
In [9]: covid_df.info()

        <class 'pandas.core.frame.DataFrame'>
        RangeIndex: 248 entries, 0 to 247
        Data columns (total 4 columns):
         #   Column      Non-Null Count  Dtype
        ---  ------      --------------  -----
         0   date        248 non-null    object
         1   new_cases   248 non-null    float64
         2   new_deaths  248 non-null    float64
         3   new_tests   135 non-null    float64
        dtypes: float64(3), object(1)
        memory usage: 7.9+ KB
```

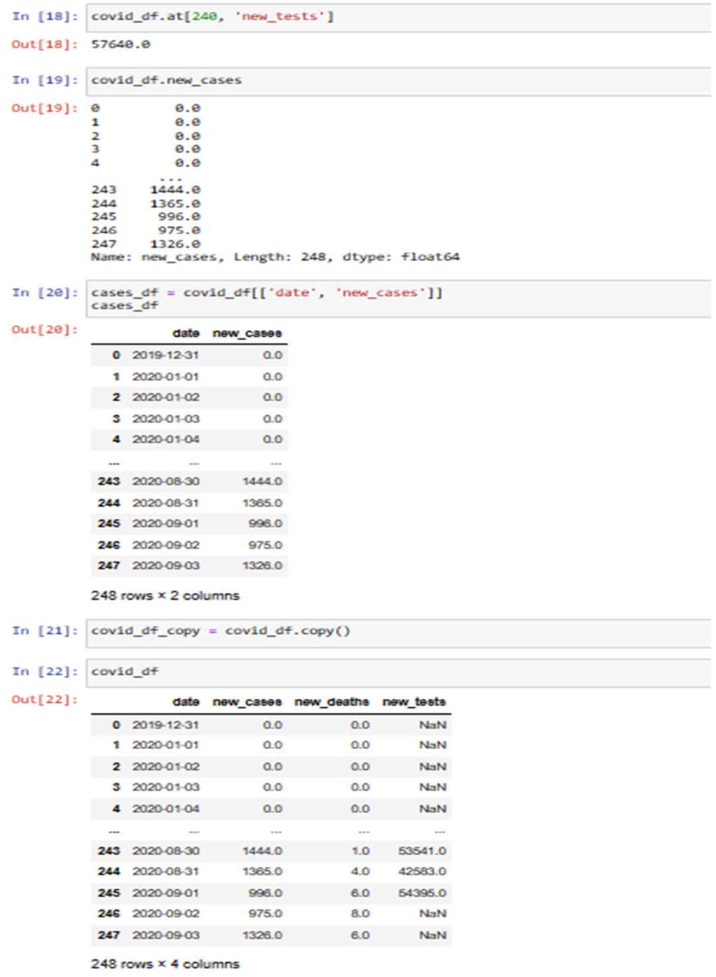
```
In [10]: covid_df.describe()
```

Out[10]:

	new_cases	new_deaths	new_tests
count	248.000000	248.000000	135.000000
mean	1094.818548	143.133065	31699.674074
std	1554.508002	227.105538	11622.209757
min	-148.000000	-31.000000	7841.000000
25%	123.000000	3.000000	25259.000000
50%	342.000000	17.000000	29545.000000
75%	1371.750000	175.250000	37711.000000
max	6557.000000	971.000000	95273.000000

```
In [11]: covid_df.columns

Out[11]: Index(['date', 'new_cases', 'new_deaths', 'new_tests'], dtype='object')
```

```
In [12]: covid_df.shape

Out[12]: (248, 4)
```



```
In [18]: covid_df.at[240, 'new_tests']

Out[18]: 57640.0
```

```
In [19]: covid_df.new_cases

Out[19]: 0         0.0
         1         0.0
         2         0.0
         3         0.0
         4         0.0
                 ...
         243    1444.0
         244    1365.0
         245     996.0
         246     975.0
         247    1326.0
         Name: new_cases, Length: 248, dtype: float64
```

```
In [20]: cases_df = covid_df[['date', 'new_cases']]
         cases_df
```

Out[20]:

	date	new_cases
0	2019-12-31	0.0
1	2020-01-01	0.0
2	2020-01-02	0.0
3	2020-01-03	0.0
4	2020-01-04	0.0
...	...	...
243	2020-08-30	1444.0
244	2020-08-31	1365.0
245	2020-09-01	996.0
246	2020-09-02	975.0
247	2020-09-03	1326.0

248 rows × 2 columns

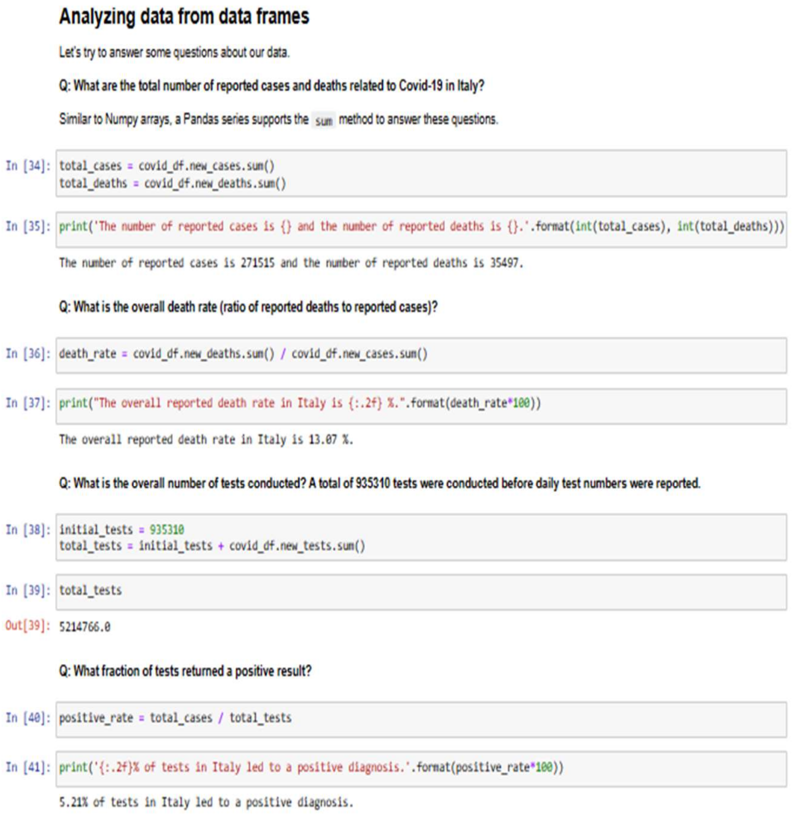
```
In [21]: covid_df_copy = covid_df.copy()
```

```
In [22]: covid_df
```

Out[22]:

	date	new_cases	new_deaths	new_tests
0	2019-12-31	0.0	0.0	NaN
1	2020-01-01	0.0	0.0	NaN
2	2020-01-02	0.0	0.0	NaN
3	2020-01-03	0.0	0.0	NaN
4	2020-01-04	0.0	0.0	NaN
...	...	...	...	...
243	2020-08-30	1444.0	1.0	53541.0
244	2020-08-31	1365.0	4.0	42583.0
245	2020-09-01	996.0	6.0	54395.0
246	2020-09-02	975.0	8.0	NaN
247	2020-09-03	1326.0	6.0	NaN

248 rows × 4 columns



Analyzing data from data frames

Let's try to answer some questions about our data.

Q: What are the total number of reported cases and deaths related to Covid-19 in Italy?

Similar to Numpy arrays, a Pandas series supports the `sum` method to answer these questions.

```
In [34]: total_cases = covid_df.new_cases.sum()
         total_deaths = covid_df.new_deaths.sum()
```

```
In [35]: print('The number of reported cases is {} and the number of reported deaths is {}.'.format(int(total_cases), int(total_deaths)))

         The number of reported cases is 271515 and the number of reported deaths is 35497.
```

Q: What is the overall death rate (ratio of reported deaths to reported cases)?

```
In [36]: death_rate = covid_df.new_deaths.sum() / covid_df.new_cases.sum()
```

```
In [37]: print("The overall reported death rate in Italy is {:.2f} %.".format(death_rate*100))

         The overall reported death rate in Italy is 13.07 %.
```

Q: What is the overall number of tests conducted? A total of 935310 tests were conducted before daily test numbers were reported.

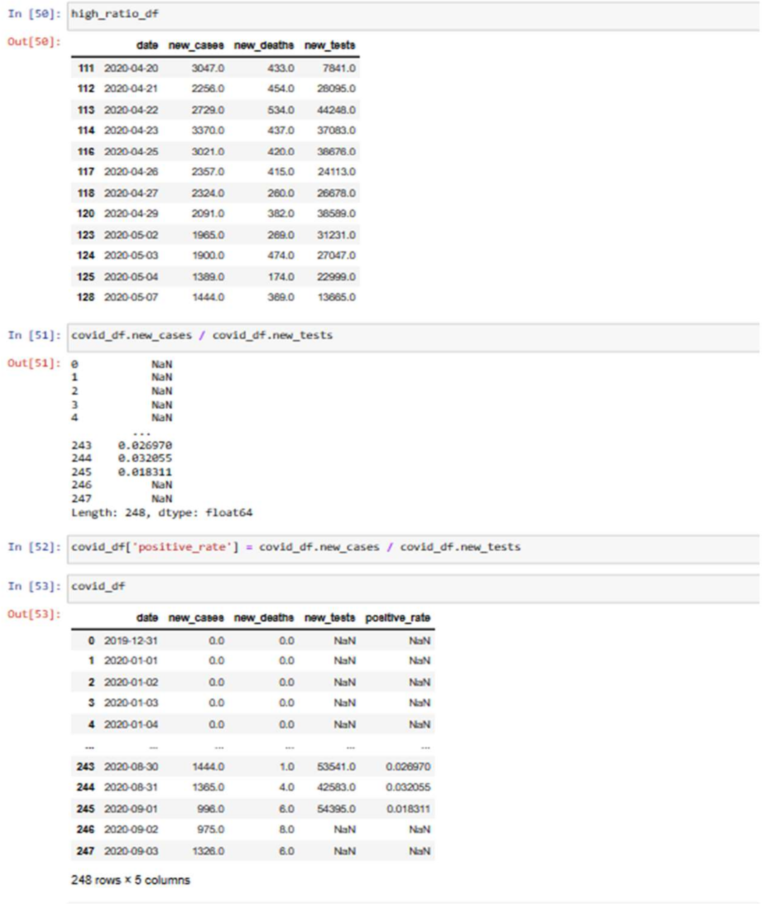
```
In [38]: initial_tests = 935310
         total_tests = initial_tests + covid_df.new_tests.sum()
```

```
In [39]: total_tests

Out[39]: 5214766.0
```

Q: What fraction of tests returned a positive result?

```
In [40]: positive_rate = total_cases / total_tests
```

```
In [41]: print('{:.2f}% of tests in Italy led to a positive diagnosis.'.format(positive_rate*100))

         5.21% of tests in Italy led to a positive diagnosis.
```



```
In [50]: high_ratio_df
```

Out[50]:

	date	new_cases	new_deaths	new_tests
111	2020-04-20	3047.0	433.0	7841.0
112	2020-04-21	2256.0	454.0	28095.0
113	2020-04-22	2729.0	534.0	44248.0
114	2020-04-23	3370.0	437.0	37083.0
116	2020-04-25	3021.0	420.0	38676.0
117	2020-04-26	2357.0	415.0	24113.0
118	2020-04-27	2324.0	260.0	26678.0
120	2020-04-29	2091.0	382.0	38589.0
123	2020-05-02	1965.0	269.0	31231.0
124	2020-05-03	1900.0	474.0	27047.0
125	2020-05-04	1389.0	174.0	22999.0
128	2020-05-07	1444.0	369.0	13665.0

```
In [51]: covid_df.new_cases / covid_df.new_tests

Out[51]: 0           NaN
         1           NaN
         2           NaN
         3           NaN
         4           NaN
                  ...
         243    0.026970
         244    0.032055
         245    0.018311
         246         NaN
         247         NaN
         Length: 248, dtype: float64
```

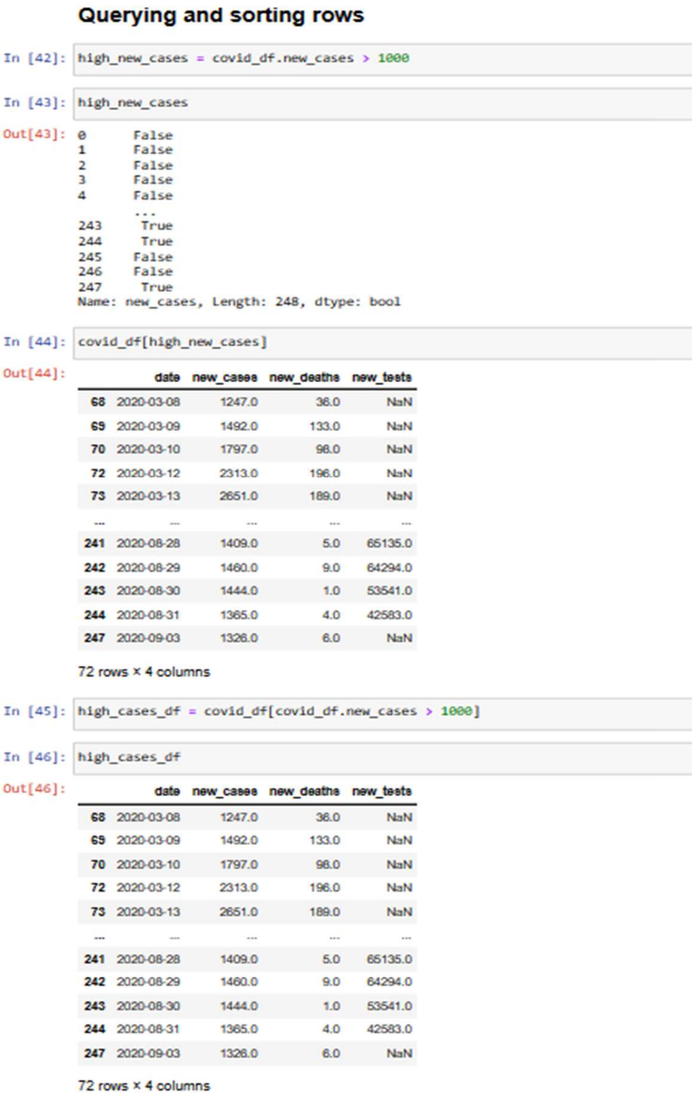
```
In [52]: covid_df['positive_rate'] = covid_df.new_cases / covid_df.new_tests
```

```
In [53]: covid_df
```

Out[53]:

	date	new_cases	new_deaths	new_tests	positive_rate
0	2019-12-31	0.0	0.0	NaN	NaN
1	2020-01-01	0.0	0.0	NaN	NaN
2	2020-01-02	0.0	0.0	NaN	NaN
3	2020-01-03	0.0	0.0	NaN	NaN
4	2020-01-04	0.0	0.0	NaN	NaN
...	...	...	...	...	...
243	2020-08-30	1444.0	1.0	53541.0	0.026970
244	2020-08-31	1365.0	4.0	42583.0	0.032055
245	2020-09-01	996.0	6.0	54395.0	0.018311
246	2020-09-02	975.0	8.0	NaN	NaN
247	2020-09-03	1326.0	6.0	NaN	NaN

248 rows × 5 columns



Querying and sorting rows

```
In [42]: high_new_cases = covid_df.new_cases > 1000
```

```
In [43]: high_new_cases

Out[43]: 0      False
         1      False
         2      False
         3      False
         4      False
                ...
         243     True
         244     True
         245    False
         246    False
         247     True
         Name: new_cases, Length: 248, dtype: bool
```

```
In [44]: covid_df[high_new_cases]
```

Out[44]:

	date	new_cases	new_deaths	new_tests
68	2020-03-08	1247.0	36.0	NaN
69	2020-03-09	1492.0	133.0	NaN
70	2020-03-10	1797.0	98.0	NaN
72	2020-03-12	2313.0	196.0	NaN
73	2020-03-13	2651.0	189.0	NaN
...	...	...	...	...
241	2020-08-28	1409.0	5.0	65135.0
242	2020-08-29	1460.0	9.0	64294.0
243	2020-08-30	1444.0	1.0	53541.0
244	2020-08-31	1365.0	4.0	42583.0
247	2020-09-03	1326.0	6.0	NaN

72 rows × 4 columns

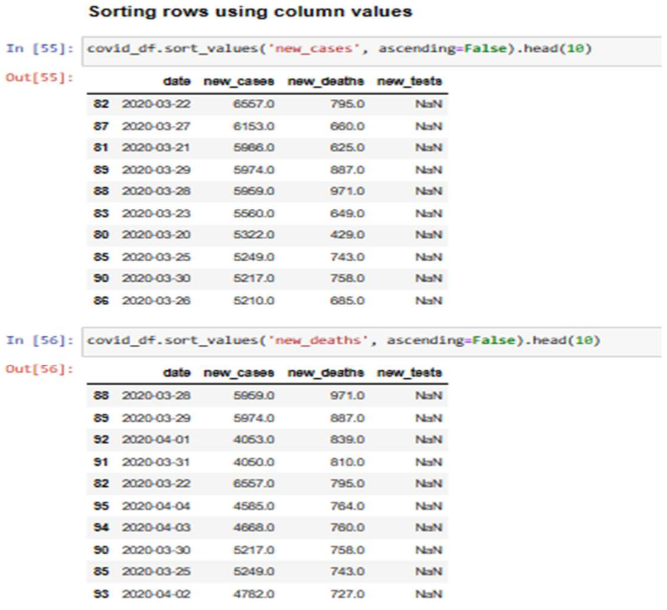
```
In [45]: high_cases_df = covid_df[covid_df.new_cases > 1000]
```

```
In [46]: high_cases_df
```

Out[46]:

	date	new_cases	new_deaths	new_tests
68	2020-03-08	1247.0	36.0	NaN
69	2020-03-09	1492.0	133.0	NaN
70	2020-03-10	1797.0	98.0	NaN
72	2020-03-12	2313.0	196.0	NaN
73	2020-03-13	2651.0	189.0	NaN
...	...	...	...	...
241	2020-08-28	1409.0	5.0	65135.0
242	2020-08-29	1460.0	9.0	64294.0
243	2020-08-30	1444.0	1.0	53541.0
244	2020-08-31	1365.0	4.0	42583.0
247	2020-09-03	1326.0	6.0	NaN

72 rows × 4 columns



Sorting rows using column values

```
In [55]: covid_df.sort_values('new_cases', ascending=False).head(10)
```

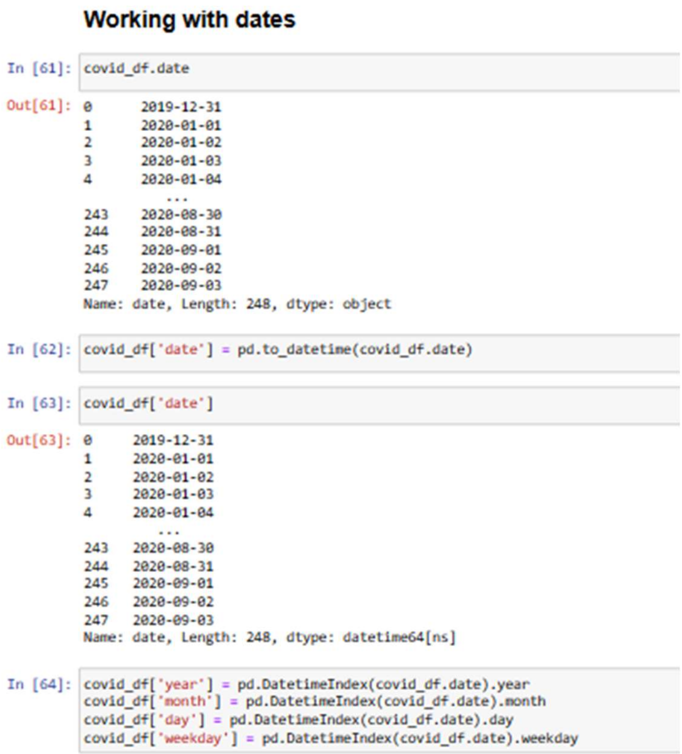
Out[55]:

	date	new_cases	new_deaths	new_tests
82	2020-03-22	6557.0	795.0	NaN
87	2020-03-27	6153.0	660.0	NaN
81	2020-03-21	5986.0	625.0	NaN
89	2020-03-29	5974.0	887.0	NaN
88	2020-03-28	5959.0	971.0	NaN
83	2020-03-23	5560.0	649.0	NaN
80	2020-03-20	5322.0	429.0	NaN
85	2020-03-25	5249.0	743.0	NaN
90	2020-03-30	5217.0	758.0	NaN
86	2020-03-26	5210.0	685.0	NaN

```
In [56]: covid_df.sort_values('new_deaths', ascending=False).head(10)
```

Out[56]:

	date	new_cases	new_deaths	new_tests
88	2020-03-28	5959.0	971.0	NaN
89	2020-03-29	5974.0	887.0	NaN
92	2020-04-01	4053.0	839.0	NaN
91	2020-03-31	4050.0	810.0	NaN
82	2020-03-22	6557.0	795.0	NaN
95	2020-04-04	4585.0	764.0	NaN
94	2020-04-03	4668.0	760.0	NaN
90	2020-03-30	5217.0	758.0	NaN
85	2020-03-25	5249.0	743.0	NaN
93	2020-04-02	4782.0	727.0	NaN



Working with dates

```
In [61]: covid_df.date
```

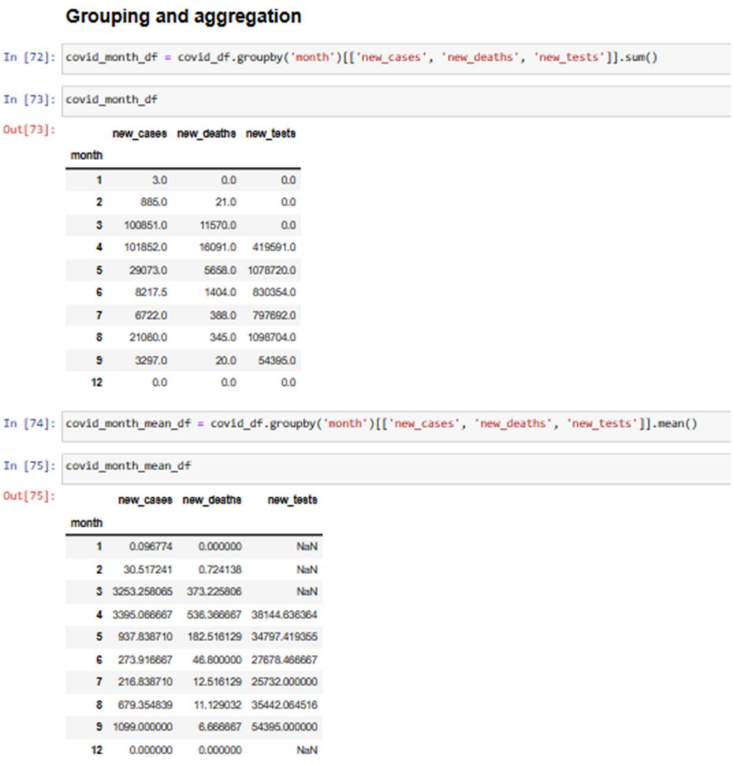
```
Out[61]: 0      2019-12-31
         1      2020-01-01
         2      2020-01-02
         3      2020-01-03
         4      2020-01-04
                   ...
         243    2020-08-30
         244    2020-08-31
         245    2020-09-01
         246    2020-09-02
         247    2020-09-03
         Name: date, Length: 248, dtype: object
```

```
In [62]: covid_df['date'] = pd.to_datetime(covid_df.date)
```

```
In [63]: covid_df['date']
```

```
Out[63]: 0     2019-12-31
         1     2020-01-01
         2     2020-01-02
         3     2020-01-03
         4     2020-01-04
                  ...
         243   2020-08-30
         244   2020-08-31
         245   2020-09-01
         246   2020-09-02
         247   2020-09-03
         Name: date, Length: 248, dtype: datetime64[ns]
```

```
In [64]: covid_df['year'] = pd.DatetimeIndex(covid_df.date).year
         covid_df['month'] = pd.DatetimeIndex(covid_df.date).month
         covid_df['day'] = pd.DatetimeIndex(covid_df.date).day
         covid_df['weekday'] = pd.DatetimeIndex(covid_df.date).weekday
```



Grouping and aggregation

```
In [72]: covid_month_df = covid_df.groupby('month')[['new_cases', 'new_deaths', 'new_tests']].sum()
```

```
In [73]: covid_month_df
```

Out[73]:

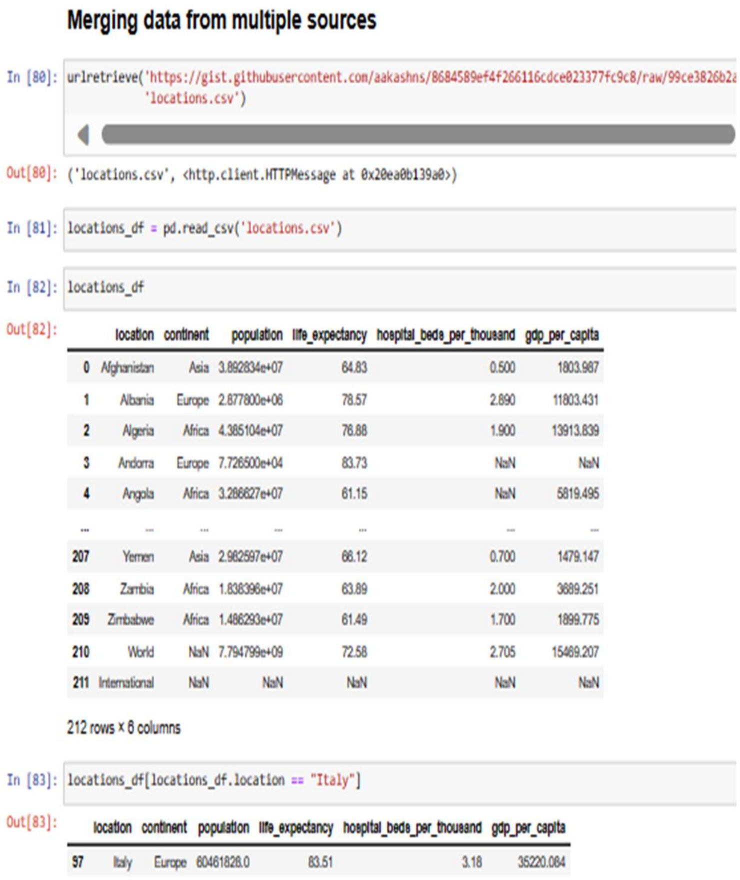
	new_cases	new_deaths	new_tests
month			
1	3.0	0.0	0.0
2	885.0	21.0	0.0
3	100851.0	11570.0	0.0
4	101852.0	16091.0	419591.0
5	29073.0	5658.0	1078720.0
6	8217.5	1404.0	830354.0
7	6722.0	388.0	797692.0
8	21060.0	345.0	1098704.0
9	3297.0	20.0	54395.0
12	0.0	0.0	0.0

```
In [74]: covid_month_mean_df = covid_df.groupby('month')[['new_cases', 'new_deaths', 'new_tests']].mean()
```

```
In [75]: covid_month_mean_df
```

Out[75]:

	new_cases	new_deaths	new_tests
month			
1	0.096774	0.000000	NaN
2	30.517241	0.724138	NaN
3	3253.258065	373.225806	NaN
4	3395.066667	536.366667	38144.636364
5	937.838710	182.516129	34797.419355
6	273.916667	46.800000	27678.466667
7	216.838710	12.516129	25732.000000
8	679.354839	11.129032	35442.064516
9	1099.000000	6.666667	54395.000000
12	0.000000	0.000000	NaN



Merging data from multiple sources

```
In [80]: urlretrieve('https://gist.githubusercontent.com/aakashns/8684589ef4f266116cdce023377fc9c8/raw/99ce3826b2a
                     'locations.csv')
```

```
Out[80]: ('locations.csv', <http.client.HTTPMessage at 0x20ea0b139a0>)
```

```
In [81]: locations_df = pd.read_csv('locations.csv')
```

```
In [82]: locations_df
```

Out[82]:

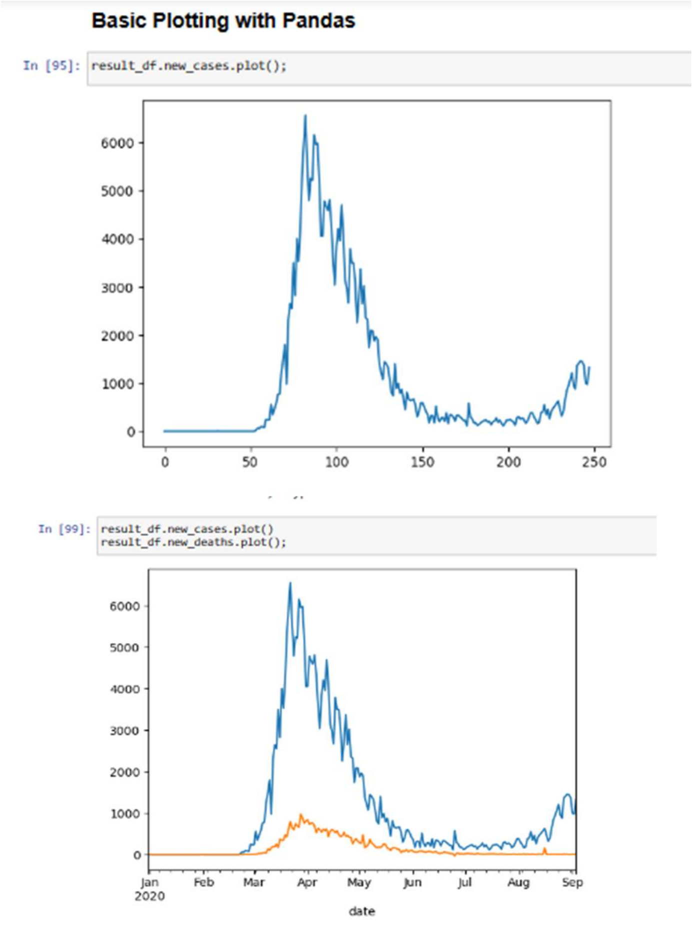
	location	continent	population	life_expectancy	hospital_beds_per_thousand	gdp_per_capita
0	Afghanistan	Asia	3.892834e+07	64.83	0.500	1803.987
1	Albania	Europe	2.877800e+06	78.57	2.890	11803.431
2	Algeria	Africa	4.385104e+07	76.88	1.900	13913.839
3	Andorra	Europe	7.726500e+04	83.73	NaN	NaN
4	Angola	Africa	3.288627e+07	61.15	NaN	5819.495
...	...	...	...	...	...	...
207	Yemen	Asia	2.982597e+07	66.12	0.700	1479.147
208	Zambia	Africa	1.838396e+07	63.89	2.000	3689.251
209	Zimbabwe	Africa	1.486293e+07	61.49	1.700	1899.775
210	World	NaN	7.794799e+09	72.58	2.705	15469.207
211	International	NaN	NaN	NaN	NaN	NaN

212 rows × 6 columns

```
In [83]: locations_df[locations_df.location == "Italy"]
```

Out[83]:

	location	continent	population	life_expectancy	hospital_beds_per_thousand	gdp_per_capita
97	Italy	Europe	60461826.0	83.51	3.18	35220.084



Basic Plotting with Pandas

```
In [95]: result_df.new_cases.plot();
```

```
In [99]: result_df.new_cases.plot()
         result_df.new_deaths.plot();
```