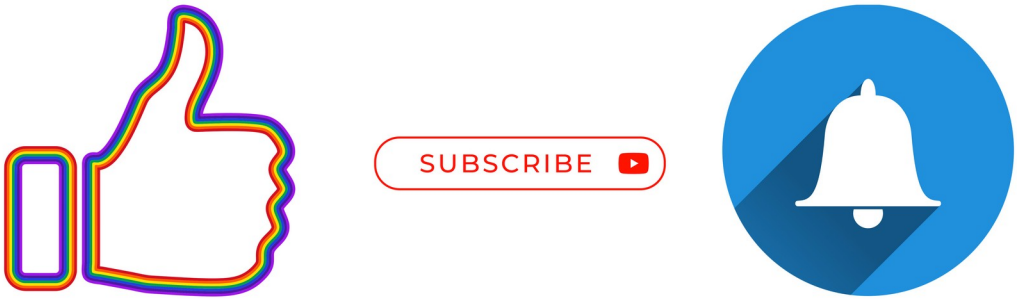
 # Python Pandas Input/Output



LIKE SUBSCRIBE PRESS

#Python Pandas Input (Import)

```
1.  pandas.read_pickle
2.  pandas.read_table
3.  pandas.read_csv
4.  pandas.read_fwf
5.  pandas.read_clipboard
6.  pandas.read_excel
7.  pandas.read_json
8.  pandas.read_html
9.  pandas.read_xml
10.  pandas.read_hdf
11.  pandas.read_feather
12.  pandas.read_parquet
13.  pandas.read_orc
```

14. pandas.read_sas
15. pandas.read_spss
16. pandas.read_sql_table
17. pandas.read_sql_query
18. pandas.read_sql
19. pandas.read_gbq
20. pandas.read_stata

Input/output

Pickling

read_pickle(filepath_or_buffer[, ...])	Load pickled pandas object (or any object) from file.
DataFrame.to_pickle(path[, compression, ...])	Pickle (serialize) object to file.

Flat file

read_table(filepath_or_buffer[, sep, ...])	Read general delimited file into DataFrame.
read_csv(filepath_or_buffer[, sep, ...])	Read a comma-separated values (csv) file into DataFrame.
DataFrame.to_csv([path_or_buf, sep, na_rep, ...])	Write object to a comma-separated values (csv) file.
read_fwf(filepath_or_buffer[, colspecs, ...])	Read a table of fixed-width formatted lines into DataFrame.

Clipboard

read_clipboard([sep])	Read text from clipboard and pass to read_csv.
DataFrame.to_clipboard([excel, sep])	Copy object to the system clipboard.

Excel

read_excel(io[, sheet_name, header, names, ...])	Read an Excel file into a pandas DataFrame.
DataFrame.to_excel(excel_writer[, ...])	Write object to an Excel sheet.
ExcelFile.parse([sheet_name, header, names, ...])	Parse specified sheet(s) into a DataFrame.
Styler.to_excel(excel_writer[, sheet_name, ...])	Write Styler to an Excel sheet.
ExcelWriter(path[, engine, date_format, ...])	Class for writing DataFrame objects into excel sheets.

JSON

read_json([path_or_buf, orient, typ, dtype, ...])	Convert a JSON string to pandas object.
json_normalize(data[, record_path, meta, ...])	Normalize semi-structured JSON data into a flat table.

`DataFrame.to_json`([path_or_buf, orient, ...])	Convert the object to a JSON string.
`build_table_schema`(data[, index, ...])	Create a Table schema from `data`.

HTML

`read_html`(io[, match, flavor, header, ...])	Read HTML tables into a `list` of `DataFrame` objects.
`DataFrame.to_html`([buf, columns, col_space, ...])	Render a DataFrame as an HTML table.
`Styler.to_html`([buf, table_uuid, ...])	Write Styler to a file, buffer or string in HTML-CSS format.

XML

`read_xml`(path_or_buffer[, xpath, ...])	Read XML document into a `DataFrame` object.
`DataFrame.to_xml`([path_or_buffer, index, ...])	Render a DataFrame to an XML document.

Latex

`DataFrame.to_latex`([buf, columns, ...])	Render object to a LaTeX tabular, longtable, or nested table.
`Styler.to_latex`([buf, column_format, ...])	Write Styler to a file, buffer or string in LaTeX format.

HDFStore: PyTables (HDF5)

`read_hdf`(path_or_buf[, key, mode, errors, ...])	Read from the store, close it if we opened it.
`HDFStore.put`(key, value[, format, index, ...])	Store object in HDFStore.
`HDFStore.append`(key, value[, format, axes, ...])	Append to Table in file.
`HDFStore.get`(key)	Retrieve pandas object stored in file.
`HDFStore.select`(key[, where, start, stop, ...])	Retrieve pandas object stored in file, optionally based on where criteria.
`HDFStore.info`()	Print detailed information on the store.
`HDFStore.keys`([include])	Return a list of keys corresponding to objects stored in HDFStore.
`HDFStore.groups`()	Return a list of all the top-level nodes.
`HDFStore.walk`([where])	Walk the pytables group hierarchy for pandas objects.

Warning

One can store a subclass of `DataFrame` or `Series` to HDF5, but the type of the subclass is lost upon storing.

Feather

`read_feather`(path[, columns, use_threads, ...])	Load a feather-format object from the file path.
`DataFrame.to_feather`(path, **kwargs)	Write a DataFrame to the binary Feather format.

Parquet

`read_parquet`(path[, engine, columns, ...])	Load a parquet object from the file path, returning a DataFrame.
`DataFrame.to_parquet`([path, engine, ...])	Write a DataFrame to the binary parquet format.

ORC

`read_orc`(path[, columns])	Load an ORC object from the file path, returning a DataFrame.

SAS

`read_sas`(filepath_or_buffer[, format, ...])	Read SAS files stored as either XPORT or SAS7BDAT format files.

SPSS

`read_spss`(path[, usecols, convert_categoricals])	Load an SPSS file from the file path, returning a DataFrame.

SQL

`read_sql_table`(table_name, con[, schema, ...])	Read SQL database table into a DataFrame.
`read_sql_query`(sql, con[, index_col, ...])	Read SQL query into a DataFrame.
`read_sql`(sql, con[, index_col, ...])	Read SQL query or database table into a DataFrame.
`DataFrame.to_sql`(name, con[, schema, ...])	Write records stored in a DataFrame to a SQL database.

Google BigQuery

`read_gbq`(query[, project_id, index_col, ...])	Load data from Google BigQuery.

STATA

`read_stata`(filepath_or_buffer[, ...])	Read Stata file into DataFrame.
`DataFrame.to_stata`(path[, convert_dates, ...])	Export DataFrame object to Stata dta format.
`StataReader.data_label`	Return data label of Stata file.
`StataReader.value_labels`()	Return a dict, associating each variable name a dict, associating each value its corresponding label.
`StataReader.variable_labels`()	Return variable labels as a dict, associating each variable name with corresponding label.
`StataWriter.write_file`()	Export DataFrame object to Stata dta format.

```
import pandas as pd
url =
'https://raw.githubusercontent.com/justmarkham/DAT8/master/data/chipot
le.tsv'
```

```
df_chipotle = pd.read_csv(url, sep = '\t')
df_chipotle
```

```
      order_id  ...  item_price
0            1  ...      $2.39
1            1  ...      $3.39
2            1  ...      $3.39
3            1  ...      $2.39
4            2  ...     $16.98
...        ...  ...         ...
4617      1833  ...     $11.75
4618      1833  ...     $11.75
4619      1834  ...     $11.25
4620      1834  ...      $8.75
4621      1834  ...      $8.75

[4622 rows x 5 columns]
```

```
import pandas as pd
url =
'https://raw.githubusercontent.com/justmarkham/DAT8/master/data/u.user
'
users = pd.read_table(url,sep='|', index_col='user_id')
users
```

```
         age gender     occupation zip_code
user_id
1         24      M     technician    85711
2         53      F          other    94043
3         23      M         writer    32067
4         24      M     technician    43537
5         33      F          other    15213
...      ...    ...            ...      ...
939       26      F        student    33319
940       32      M  administrator    02215
941       20      M        student    97229
942       48      F      librarian    78209
943       22      M        student    77841

[943 rows x 4 columns]
```

```
import pandas as pd
euro12 = pd.read_csv('Euro2012TEAM.csv')
euro12
```

```
                  Team  Goals  ...  Subs off  Players Used
0              Croatia      4  ...         9            16
1       Czech Republic      4  ...        11            19
2              Denmark      4  ...         7            15
3              England      5  ...        11            16
4               France      3  ...        11            19
```

```
5                Germany     10  ...        15            17
6                 Greece      5  ...        12            20
7                  Italy      6  ...        18            19
8            Netherlands      2  ...         7            15
9                 Poland      2  ...         7            17
10              Portugal      6  ...        14            16
11  Republic of Ireland      1  ...        10            17
12                Russia      5  ...         7            16
13                 Spain     12  ...        17            18
14                Sweden      5  ...         9            18
15               Ukraine      2  ...         9            18

[16 rows x 35 columns]
```

```
# importing packages
import pandas as pd

# dictionary of data
dct = {'ID': {0: 23, 1: 43, 2: 12,
                  3: 13, 4: 67, 5: 89,
                  6: 90, 7: 56, 8: 34},
      'StuName': {0: 'Ram', 1: 'Deep',
                       2: 'Yash', 3: 'Aman',
                       4: 'Arjun', 5: 'Aditya',
                       6: 'Divya', 7: 'Chalsea',
                       8: 'Akash' },
      'Percent': {0: 89, 1: 97, 2: 45, 3: 78,
                       4: 56, 5: 76, 6: 100, 7: 87,
                       8: 81},
      'Rank': {0: 'B', 1: 'A', 2: 'F', 3: 'C',
                       4: 'E', 5: 'C', 6: 'A', 7: 'B',
                       8: 'B'}
      }

# forming dataframe
data = pd.DataFrame(dct)

# using to_pickle function to form file
# with name 'pickle_file'
pd.to_pickle(data,'./pickle_file.pkl')

# unpickled the data by using the
# pd.read_pickle method
unpickled_data = pd.read_pickle("./pickle_file.pkl")
print(unpickled_data)
```

```
   ID  StuName  Percent Rank
0  23      Ram       89    B
1  43     Deep       97    A
2  12     Yash       45    F
```

```
3  13     Aman       78    C
4  67    Arjun       56    E
5  89   Aditya       76    C
6  90    Divya      100    A
7  56  Chalsea       87    B
8  34    Akash       81    B
```

```
df_euro_table = pd.read_table('Euro2012TEAM.csv',delimiter=',')
df_euro_table
```

```
                   Team  Goals  ...  Subs off  Players Used
0               Croatia      4  ...         9            16
1        Czech Republic      4  ...        11            19
2               Denmark      4  ...         7            15
3               England      5  ...        11            16
4                France      3  ...        11            19
5               Germany     10  ...        15            17
6                Greece      5  ...        12            20
7                 Italy      6  ...        18            19
8           Netherlands      2  ...         7            15
9                Poland      2  ...         7            17
10             Portugal      6  ...        14            16
11  Republic of Ireland      1  ...        10            17
12               Russia      5  ...         7            16
13                Spain     12  ...        17            18
14               Sweden      5  ...         9            18
15              Ukraine      2  ...         9            18

[16 rows x 35 columns]
```

```
df_foodstore_xml = pd.read_xml("foodstore.xml")
df_foodstore_xml
```

```
                 Msg  slNo foodItem  price   quantity discount
0  Food Store items.   NaN     None    NaN       None     None
1               None   1.0     meat  200.0        1kg       7%
2               None   2.0     fish  150.0        1kg       5%
3               None   3.0      egg  100.0  50 pieces       5%
4               None   4.0     milk   50.0    1 litre       3%
```

```
df_html = pd.read_html("Euro2012TeamHTML.html")
df_html
```

```
[    Unnamed: 0                Team  Goals  ...  Subs on  Subs off
Players Used
 0            0              Croatia      4  ...        9         9
16
 1            1       Czech Republic      4  ...       11        11
19
 2            2              Denmark      4  ...        7         7
15
```

```
 3            3              England      5  ...       11        11
16
 4            4               France      3  ...       11        11
19
 5            5              Germany     10  ...       15        15
17
 6            6               Greece      5  ...       12        12
20
 7            7                Italy      6  ...       18        18
19
 8            8          Netherlands      2  ...        7         7
15
 9            9               Poland      2  ...        7         7
17
 10          10             Portugal      6  ...       14        14
16
 11          11  Republic of Ireland      1  ...       10        10
17
 12          12               Russia      5  ...        7         7
16
 13          13                Spain     12  ...       17        17
18
 14          14               Sweden      5  ...        9         9
18
 15          15              Ukraine      2  ...        9         9
18

 [16 rows x 36 columns]]
```

```
df_json = pd.read_json("Euro2012TeamJSON.json")
df_json
```

```
                   Team  Goals  ...  Subs off  Players Used
0               Croatia      4  ...         9            16
1        Czech Republic      4  ...        11            19
2               Denmark      4  ...         7            15
3               England      5  ...        11            16
4                France      3  ...        11            19
5               Germany     10  ...        15            17
6                Greece      5  ...        12            20
7                 Italy      6  ...        18            19
8           Netherlands      2  ...         7            15
9                Poland      2  ...         7            17
10             Portugal      6  ...        14            16
11  Republic of Ireland      1  ...        10            17
12               Russia      5  ...         7            16
13                Spain     12  ...        17            18
14               Sweden      5  ...         9            18
15              Ukraine      2  ...         9            18
```

```
[16 rows x 35 columns]
```

```
df_xlsx = pd.read_excel("Euro2012TeamXLSX.xlsx")
df_xlsx
```

```
    Unnamed: 0                Team  Goals  ...  Subs on  Subs off
Players Used
0            0             Croatia      4  ...        9         9
16
1            1      Czech Republic      4  ...       11        11
19
2            2             Denmark      4  ...        7         7
15
3            3             England      5  ...       11        11
16
4            4              France      3  ...       11        11
19
5            5             Germany     10  ...       15        15
17
6            6              Greece      5  ...       12        12
20
7            7               Italy      6  ...       18        18
19
8            8         Netherlands      2  ...        7         7
15
9            9              Poland      2  ...        7         7
17
10          10            Portugal      6  ...       14        14
16
11          11  Republic of Ireland      1  ...       10        10
17
12          12              Russia      5  ...        7         7
16
13          13               Spain     12  ...       17        17
18
14          14              Sweden      5  ...        9         9
18
15          15             Ukraine      2  ...        9         9
18

[16 rows x 36 columns]
```

```
df_xlsx.to_csv("example_csv.csv")
```

```
df_xlsx.to_html("example_html.html")
```

```
df_xlsx.to_json("example_json.json")
```

```
df_xlsx.to_csv("example_csv.csv")
```