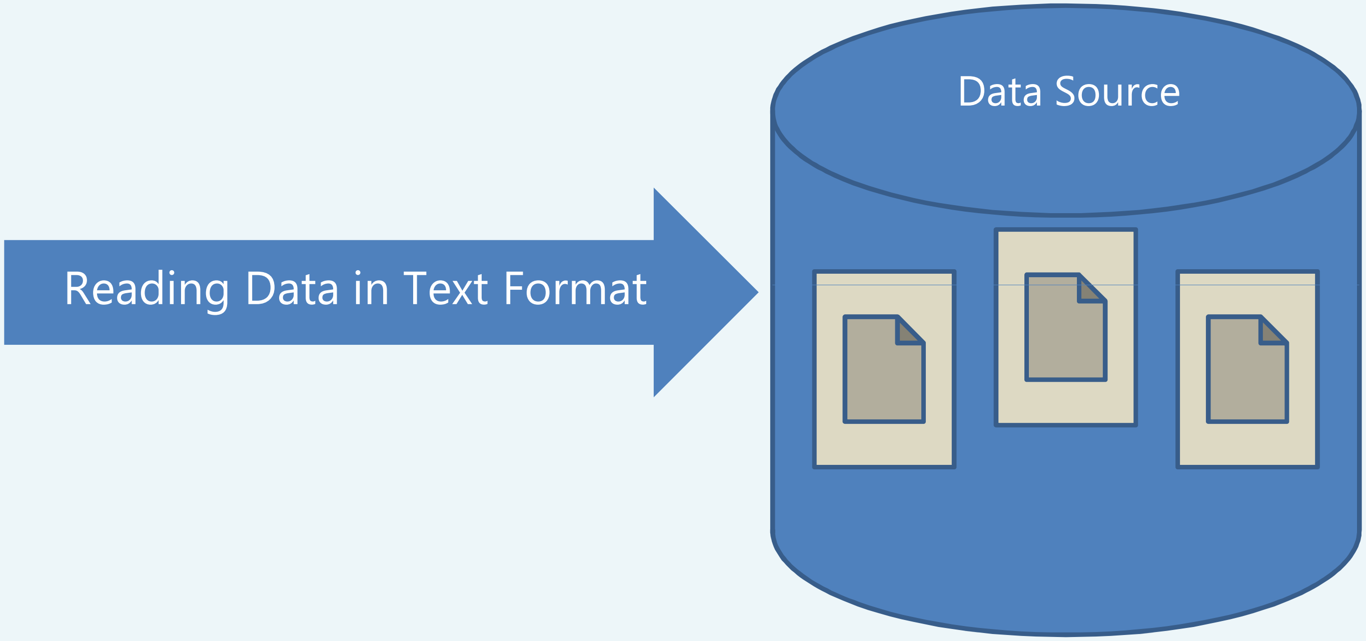
Data Handling

Data Handling

Reading Data in Text Format

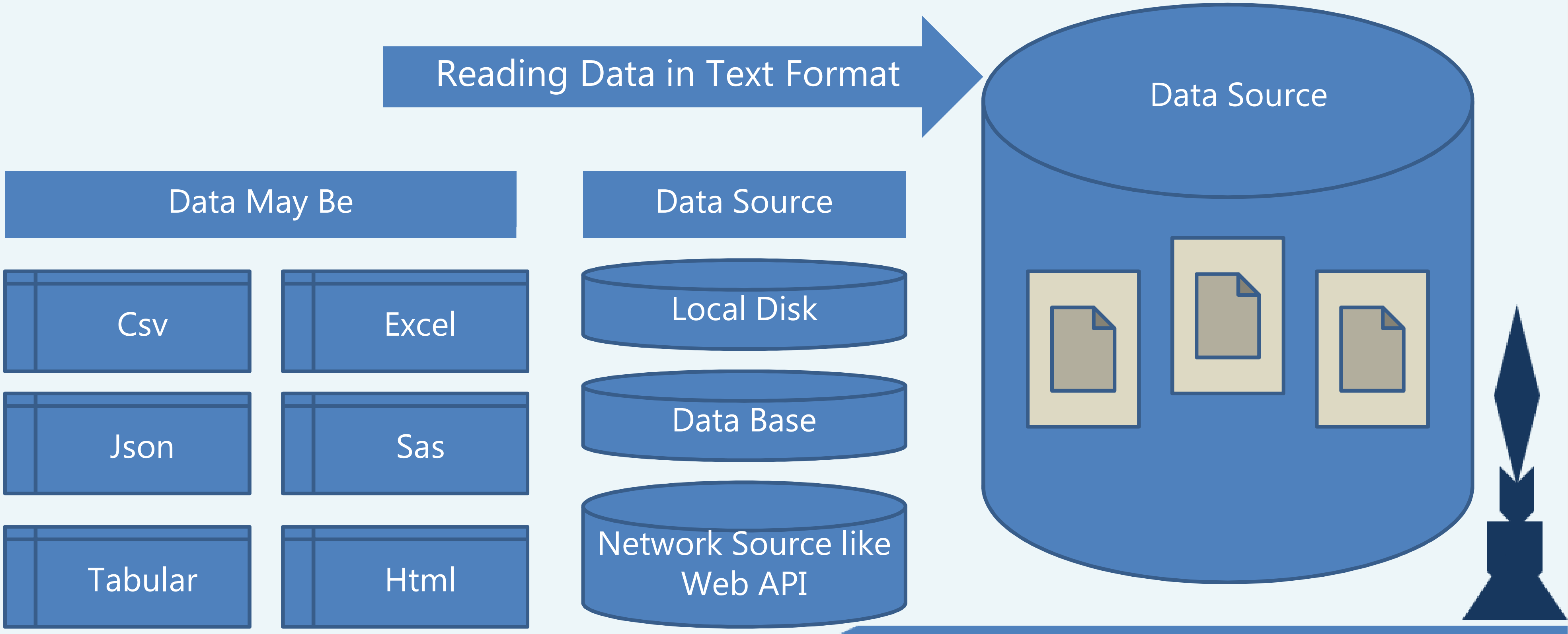


Reading Data in Text Format

Data Source

Data Handling

Reading Data in Text Format



Reading Data in Text Format

Data Source

Data May Be

Csv

Excel

Json

Sas

Tabular

Html

Data Source

Local Disk

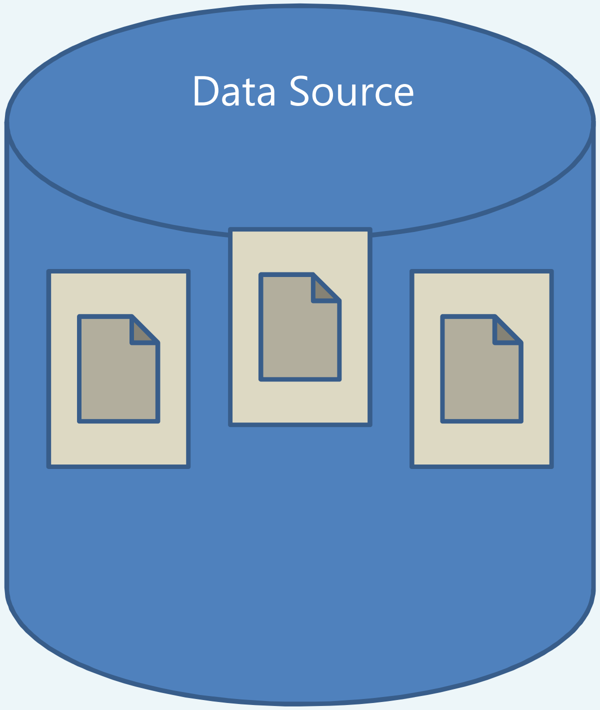
Data Base

Network Source like Web API

Data Handling

Reading Data in Text Format

Reading Data in Text Format



Functions For Reading Tabular Data as a DataFrame

read_csv	read_table	read_sql
read_excel	read_sas	read_pickle
read_json	read_html	read_clipbord

Data Handling

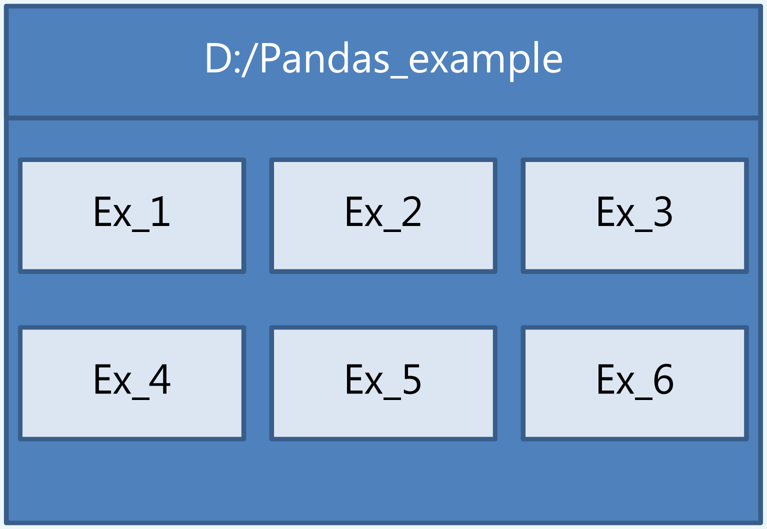
Reading Data in Text Format

To print the raw contents of the file From Local Source (Like PCs D drive) to the screen.

```
!type D:/Pandas_example/Ex_1
```

```
!cat D:/Pandas_example/Ex_1
```

Data Source



D:/Pandas_example

Ex_1 Ex_2 Ex_3

Ex_4 Ex_5 Ex_6

Data Handling

Reading Data in Text Format

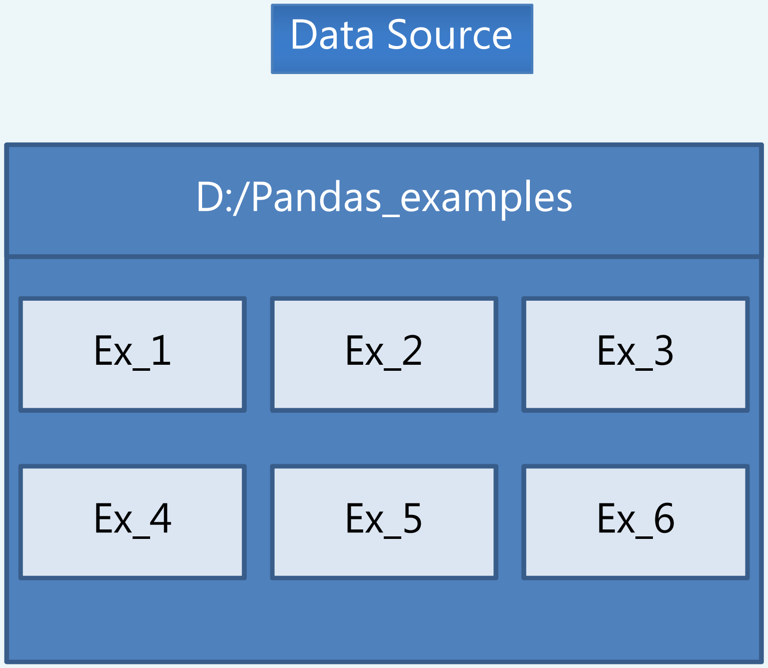
To read the ‘csv files’, ‘text files with variable amount of whitespace’ into the Pandas DataFrame.

```
Df = pd.read_csv(‘File Path’)
```

```
Df = pd.read_csv(‘D:/Pandas_examples/Ex_1.csv’)
```

```
Df = read_table(‘D:/Pandas_examples/Ex_2.csv’, sep=‘ , ’)
```

```
Df = pd.read_table(' D:/Pandas_examples/Ex_3.txt ', sep='\s+')
```



Data Source

D:/Pandas_examples

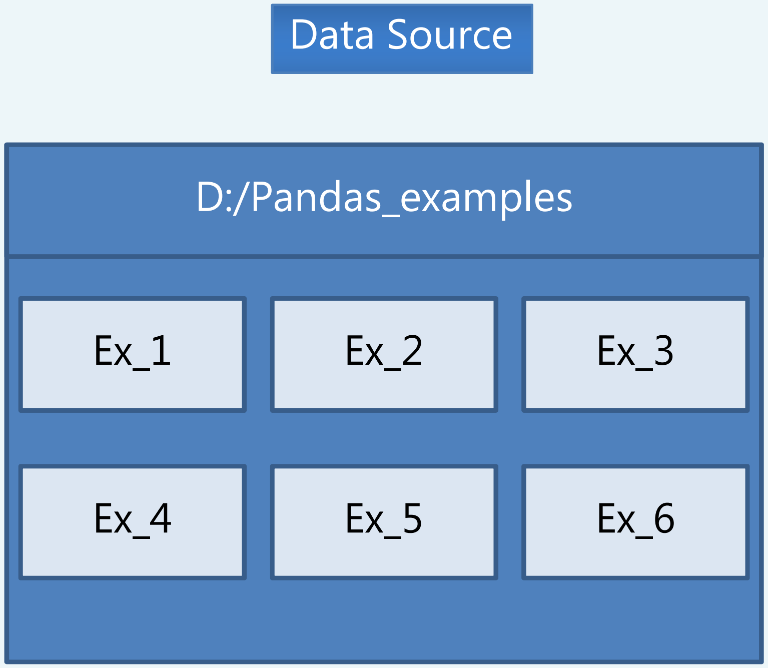
Ex_1	Ex_2	Ex_3
Ex_4	Ex_5	Ex_6

Data Handling

Reading Data in Text Format

Pandas provides many functional arguments to read different types of files.

Argument	Argument
path	skiprows
sep	na_values
header	encoding
index_col	nrows
names	skip_footer



Data Source

D:/Pandas_examples

Ex_1 Ex_2 Ex_3

Ex_4 Ex_5 Ex_6

Data Handling

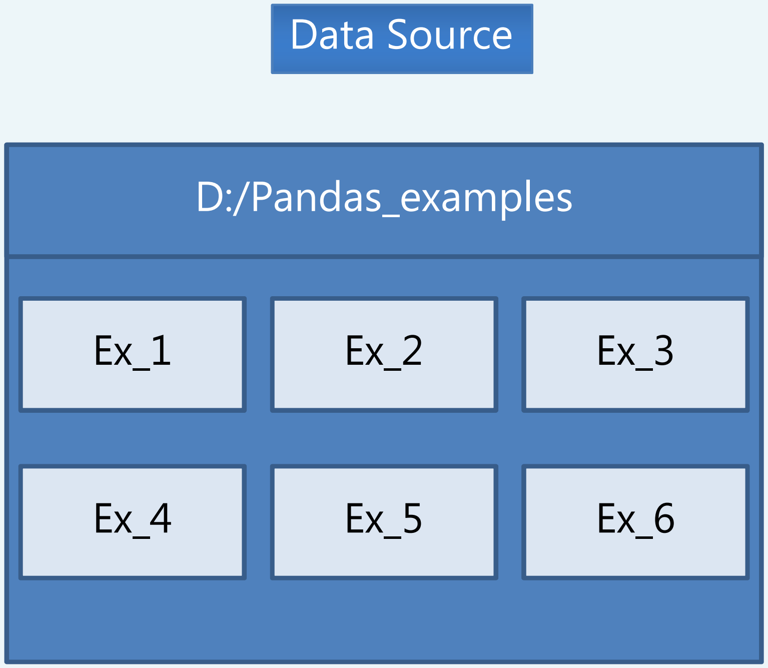
Reading Text Files in Pieces

To read small number of Rows

- `pd.read_csv('D:/Pandas_examples/Ex_4.csv', nrows=5)`

To read a file in pieces, We can specify a chunk size as a Number of Rows

- `pd.read_csv('D:/Pandas_examples/Ex_4.csv', chunksize=1000)`



Data Source

D:/Pandas_examples

Ex_1	Ex_2	Ex_3
Ex_4	Ex_5	Ex_6

Data Handling

Writing Data To A Text File

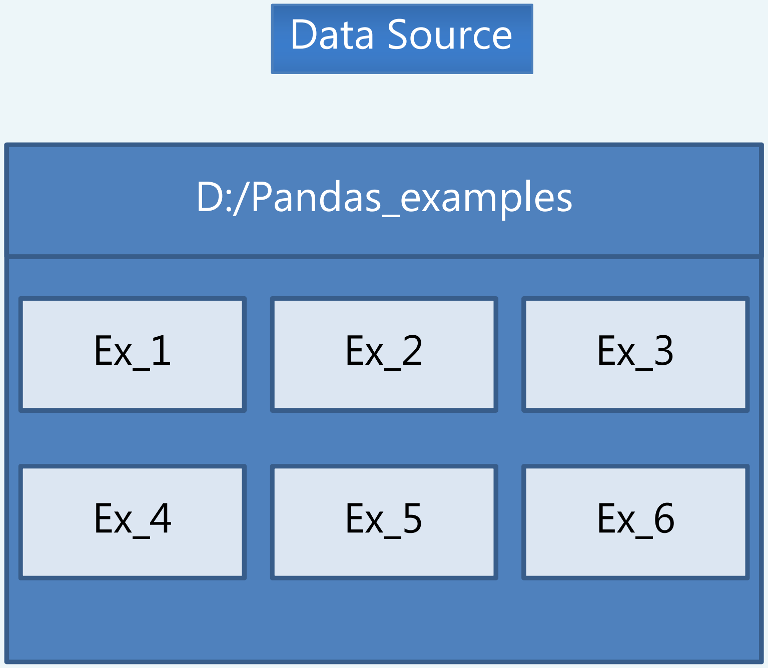
To Export data from a comma separated text file to csv file

Pandas provides ‘to_csv’ method to export the data

- `Ex_5.to_csv('D:/Pandas_examples/Ex_6.csv‘, sep=‘,’)`

It Provides many other Optional Arguments To Control How We can export the Data with different possibilities

- `Ex.: we can use sep=‘|’ instead of sep=‘,’`



Data Source

D:/Pandas_examples

Ex_1 Ex_2 Ex_3

Ex_4 Ex_5 Ex_6

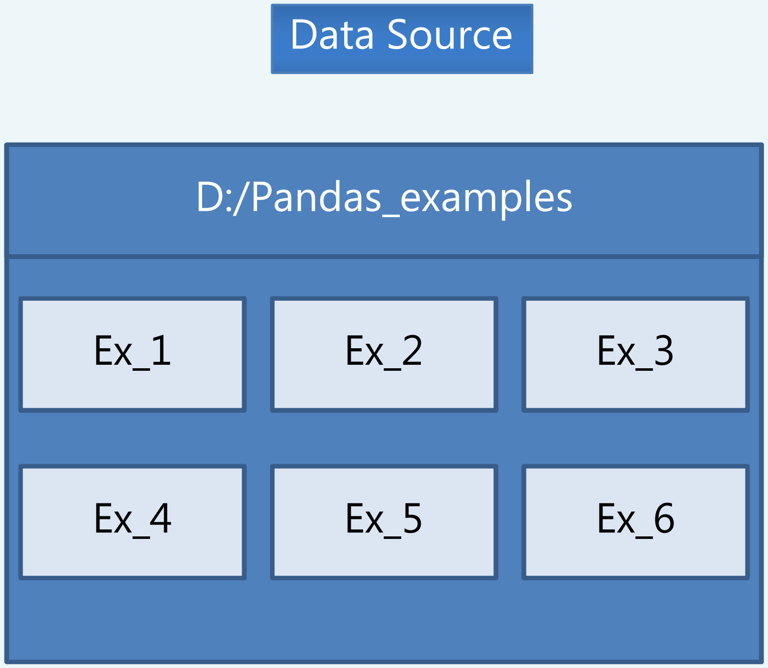
Data Handling

Working With Delimited Data

In order to read a csv file use Python's built-in csv module

'csv.reader'

To write delimited files manually, you can use 'csv.writer'



Data Source

D:/Pandas_examples

Ex_1 Ex_2 Ex_3

Ex_4 Ex_5 Ex_6

Data Handling

Other File Handling Methods

JSON

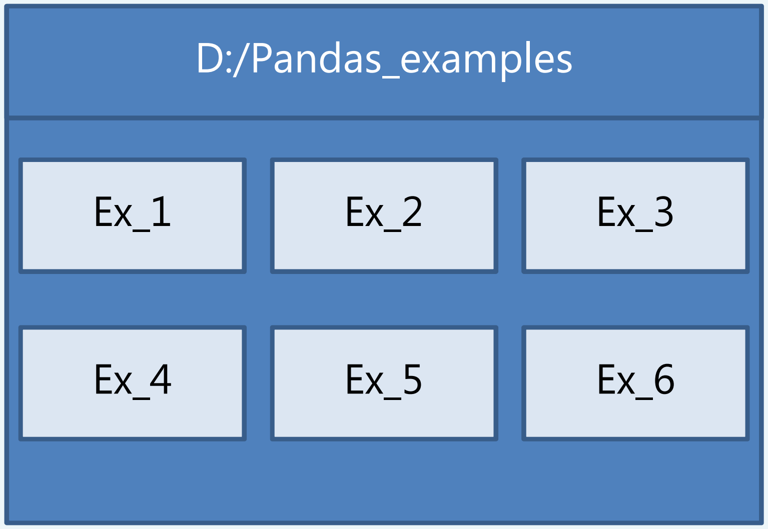
XML & HTML

Binary Data Formats

HDF5 FORMAT

EXCEL FILE

Data Source



D:/Pandas_examples

Ex_1	Ex_2	Ex_3
Ex_4	Ex_5	Ex_6

Data Handling