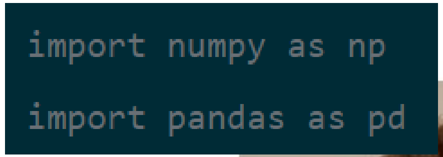
таблицы PanDas

Работа с БД и таблицами

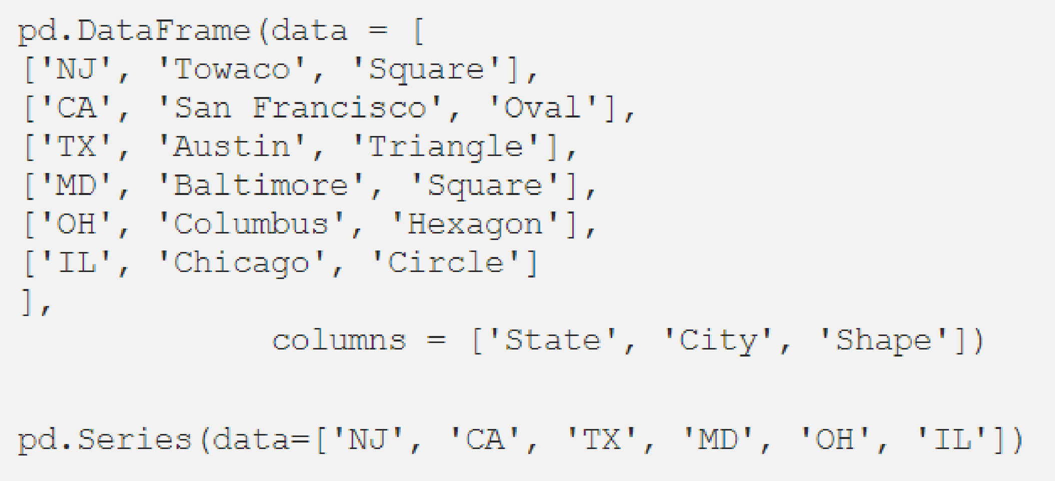
Microsoft SQL Server
ORACLE
PostgreSQL
MySQL
SYBASE
IBM DB2.
Microsoft Access
ГАЗПРОМ НЕФТЬ
Excel

Pandas = panel data set



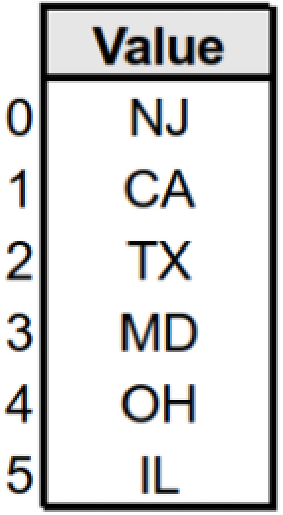
```
import numpy as np

import pandas as pd
```



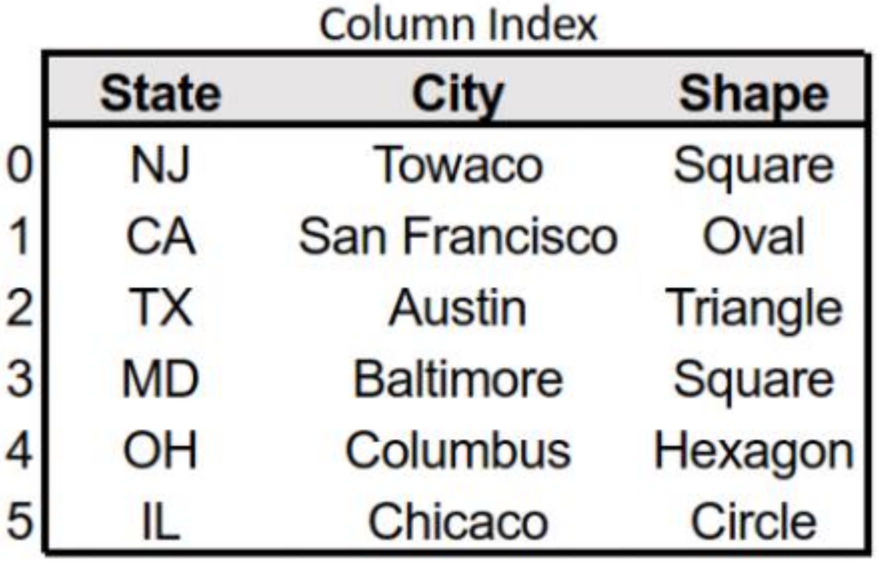
```
pd.DataFrame(data = [
['NJ', 'Towaco', 'Square'],
['CA', 'San Francisco', 'Oval'],
['TX', 'Austin', 'Triangle'],
['MD', 'Baltimore', 'Square'],
['OH', 'Columbus', 'Hexagon'],
['IL', 'Chicago', 'Circle']
],
                 columns = ['State', 'City', 'Shape'])


pd.Series(data=['NJ', 'CA', 'TX', 'MD', 'OH', 'IL'])
```



	Value
0	NJ
1	CA
2	TX
3	MD
4	OH
5	IL

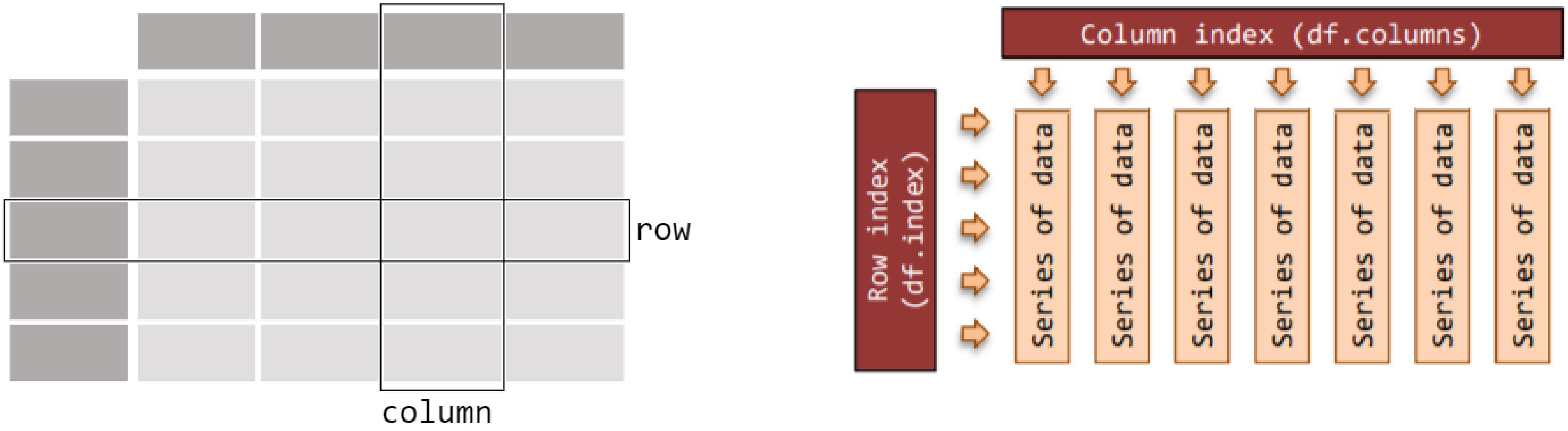
Series



	Column Index		
	State	City	Shape
0	NJ	Towaco	Square
1	CA	San Francisco	Oval
2	TX	Austin	Triangle
3	MD	Baltimore	Square
4	OH	Columbus	Hexagon
5	IL	Chicaco	Circle

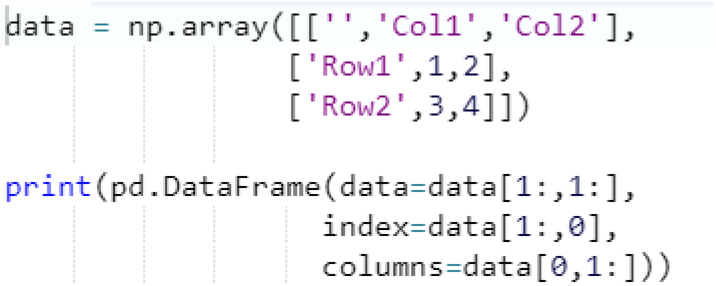
Dataframe

Dataframe состоит из Series



Dataframe & Series

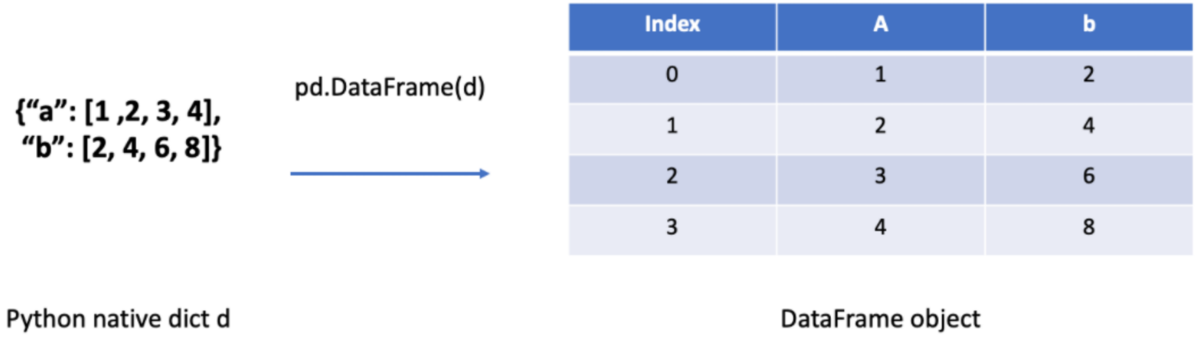
Из чего их создавать?



```
data = np.array([['','Col1','Col2'],
                 ['Row1',1,2],
                 ['Row2',3,4]])

print(pd.DataFrame(data=data[1:,1:],
                   index=data[1:,0],
                   columns=data[0,1:]))
```

массивы



{"a": [1 ,2, 3, 4],
"b": [2, 4, 6, 8]}

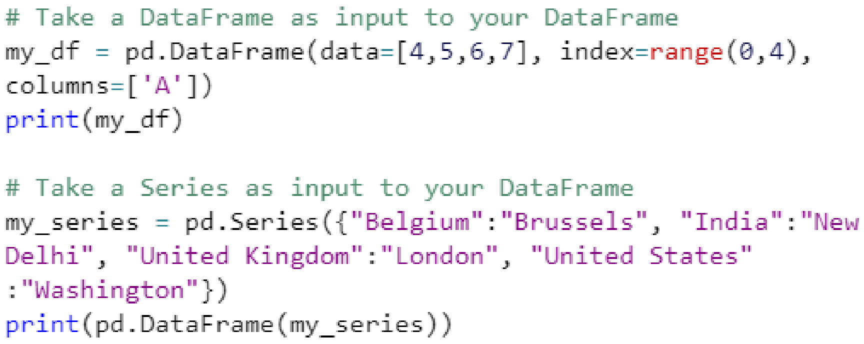
pd.DataFrame(d)

Index	A	b
0	1	2
1	2	4
2	3	6
3	4	8

Python native dict d

DataFrame object

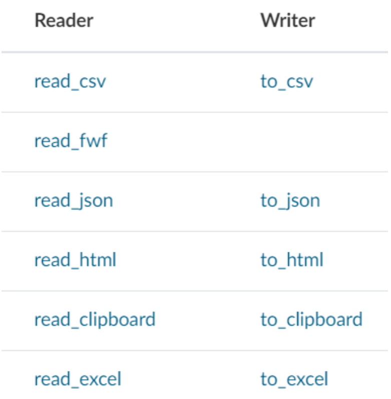
словари



```
# Take a DataFrame as input to your DataFrame
my_df = pd.DataFrame(data=[4,5,6,7], index=range(0,4),
columns=['A'])
print(my_df)

# Take a Series as input to your DataFrame
my_series = pd.Series({"Belgium":"Brussels", "India":"New
Delhi", "United Kingdom":"London", "United States"
:"Washington"})
print(pd.DataFrame(my_series))
```

Dataframe & Series



Reader	Writer
read_csv	to_csv
read_fwf	
read_json	to_json
read_html	to_html
read_clipboard	to_clipboard
read_excel	to_excel

файлы! https://pandas.pydata.org/pandas-docs/stable/user_guide/io.html

Как теперь с этим работать?

Доступ к элементам, столбцам, строкам, срезы, map, конкатенация…

- нужно почитать маны

https://pandas.pydata.org/pandas-docs/stable/user_guide/10min.html

- попробовать поработать руками

https://www.datacamp.com/community/tutorials/pandas-tutorial-dataframe-python#question1
https://habr.com/ru/company/ruvds/blog/494720/

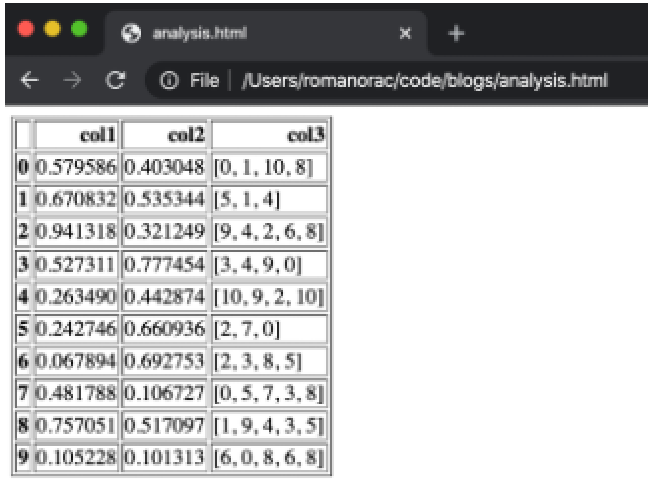
- поисследовать хиросты

https://habr.com/ru/company/ruvds/blog/492220/

- и распечатать cheatsheet

https://www.webpages.uidaho.edu/~stevel/cheatsheets/Pandas%20DataFrame%20Notes_12pages.pdf

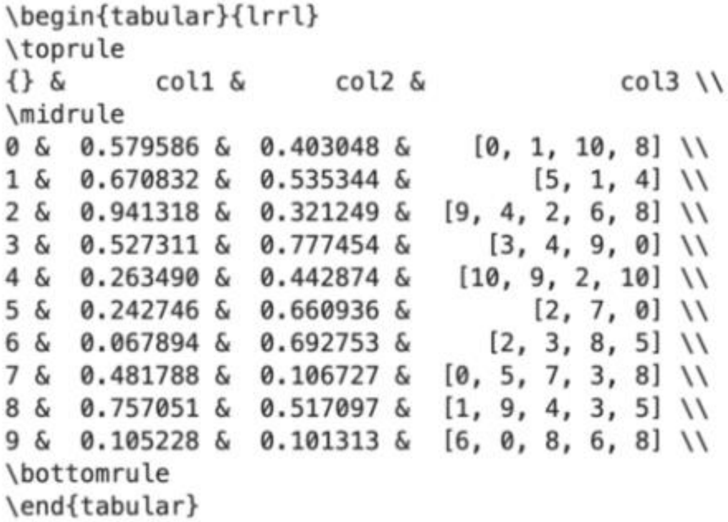
Html



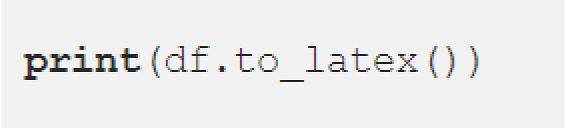
	col1	col2	col3
0	0.579586	0.403048	[0, 1, 10, 8]
1	0.670832	0.535344	[5, 1, 4]
2	0.941318	0.321249	[9, 4, 2, 6, 8]
3	0.527311	0.777454	[3, 4, 9, 0]
4	0.263490	0.442874	[10, 9, 2, 10]
5	0.242746	0.660936	[2, 7, 0]
6	0.067894	0.692753	[2, 3, 8, 5]
7	0.481788	0.106727	[0, 5, 7, 3, 8]
8	0.757051	0.517097	[1, 9, 4, 3, 5]
9	0.105228	0.101313	[6, 0, 8, 6, 8]

```
df_html = df.to_html()with open('analysis.html', 'w') as f:
    f.write(df_html)
```

LaTex

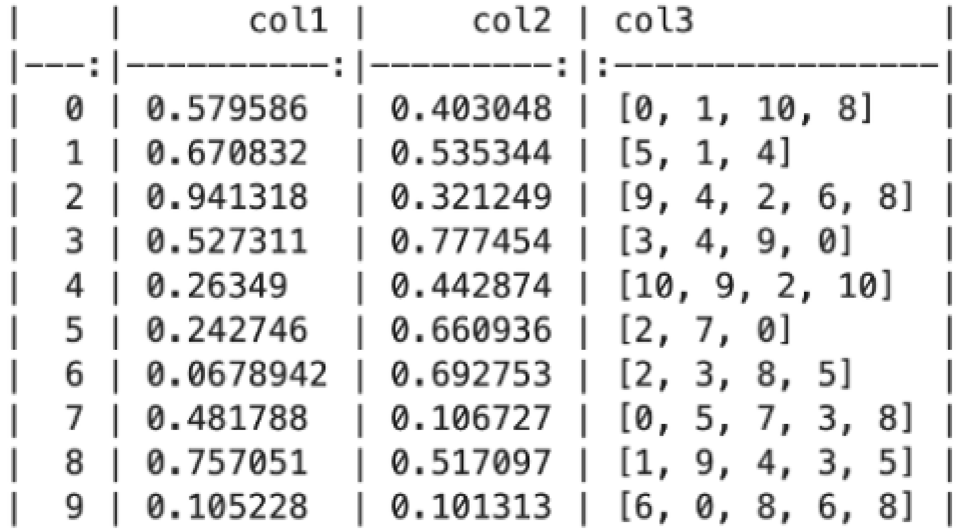


```
\begin{tabular}{lrrl}
\toprule
{} &      col1 &      col2 &             col3 \\
\midrule
0 &  0.579586 &  0.403048 &    [0, 1, 10, 8] \\
1 &  0.670832 &  0.535344 &        [5, 1, 4] \\
2 &  0.941318 &  0.321249 &  [9, 4, 2, 6, 8] \\
3 &  0.527311 &  0.777454 &     [3, 4, 9, 0] \\
4 &  0.263490 &  0.442874 &   [10, 9, 2, 10] \\
5 &  0.242746 &  0.660936 &        [2, 7, 0] \\
6 &  0.067894 &  0.692753 &     [2, 3, 8, 5] \\
7 &  0.481788 &  0.106727 &  [0, 5, 7, 3, 8] \\
8 &  0.757051 &  0.517097 &  [1, 9, 4, 3, 5] \\
9 &  0.105228 &  0.101313 &  [6, 0, 8, 6, 8] \\
\bottomrule
\end{tabular}
```

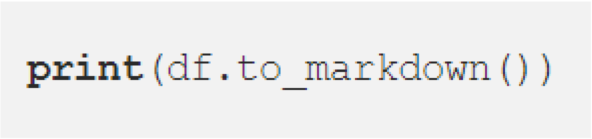


```
print(df.to_latex())
```

Markdown

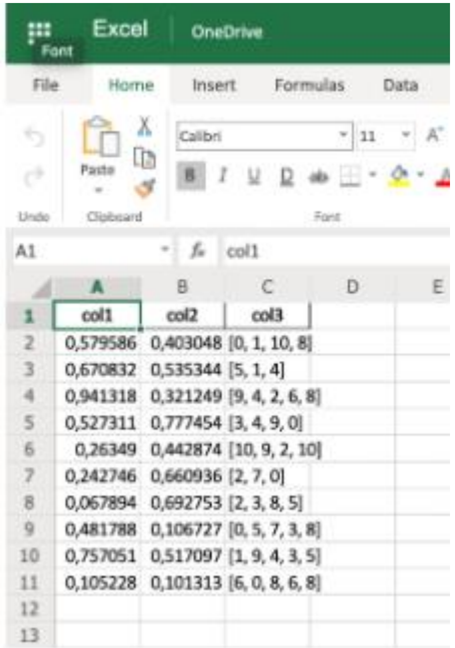


```
|    |      col1 |     col2 | col3             |
|---:|----------:|---------:|:-----------------|
|  0 | 0.579586  | 0.403048 | [0, 1, 10, 8]    |
|  1 | 0.670832  | 0.535344 | [5, 1, 4]        |
|  2 | 0.941318  | 0.321249 | [9, 4, 2, 6, 8]  |
|  3 | 0.527311  | 0.777454 | [3, 4, 9, 0]     |
|  4 | 0.26349   | 0.442874 | [10, 9, 2, 10]   |
|  5 | 0.242746  | 0.660936 | [2, 7, 0]        |
|  6 | 0.0678942 | 0.692753 | [2, 3, 8, 5]     |
|  7 | 0.481788  | 0.106727 | [0, 5, 7, 3, 8]  |
|  8 | 0.757051  | 0.517097 | [1, 9, 4, 3, 5]  |
|  9 | 0.105228  | 0.101313 | [6, 0, 8, 6, 8]  |
```



```
print(df.to_markdown())
```

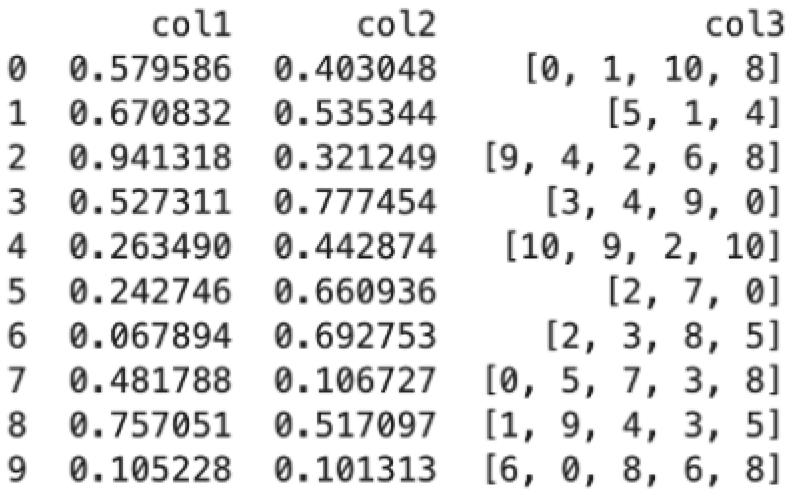
excel



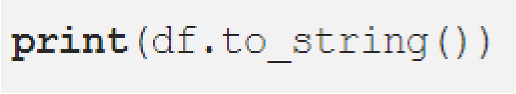
col1	col2	col3
0,579586	0,403048	[0, 1, 10, 8]
0,670832	0,535344	[5, 1, 4]
0,941318	0,321249	[9, 4, 2, 6, 8]
0,527311	0,777454	[3, 4, 9, 0]
0,26349	0,442874	[10, 9, 2, 10]
0,242746	0,660936	[2, 7, 0]
0,067894	0,692753	[2, 3, 8, 5]
0,481788	0,106727	[0, 5, 7, 3, 8]
0,757051	0,517097	[1, 9, 4, 3, 5]
0,105228	0,101313	[6, 0, 8, 6, 8]

```
df.to_excel('analysis.xlsx', index=False)
```

string



```
       col1      col2              col3
0  0.579586  0.403048    [0, 1, 10, 8]
1  0.670832  0.535344        [5, 1, 4]
2  0.941318  0.321249  [9, 4, 2, 6, 8]
3  0.527311  0.777454     [3, 4, 9, 0]
4  0.263490  0.442874   [10, 9, 2, 10]
5  0.242746  0.660936        [2, 7, 0]
6  0.067894  0.692753     [2, 3, 8, 5]
7  0.481788  0.106727  [0, 5, 7, 3, 8]
8  0.757051  0.517097  [1, 9, 4, 3, 5]
9  0.105228  0.101313  [6, 0, 8, 6, 8]
```



```
print(df.to_string())
```