Jupyter Book Project

Author Name

Sep 30, 2021

CONTENTS

This is a small sample book to give you a feel for how book content is structured.

Note: Here is a note!

And here is a code block:

```
e = mc^2
```

Check out the content pages bundled with this sample book to see more.

test

CHAPTER

ONE

EXAMPLE

```
import pandas as pd
list1=[3,-5,7,4]
list2=['a','b','c','d']
series=pd.Series(list1,index=list2)
series
```

```
a    3
b   -5
c    7
d    4
dtype: int64
```

```
data_dict = {'Numbers': [1,2,3],
             'Countries': ['DE','FR','IT'],
             'Color': ['Blue','Red','Green']}
df = pd.DataFrame(data=data_dict)
df
```

```
   Numbers Countries  Color
0        1        DE   Blue
1        2        FR    Red
2        3        IT  Green
```

```
import numpy as np
ndarray = np.array([(1,2,3),
                    (4,5,6),
                    (7,8,9)])
df2 = pd.DataFrame(ndarray, columns = ['a','b','c'])
df2
```

```
   a  b  c
0  1  2  3
1  4  5  6
2  7  8  9
```

1.1 common manipulations

Target	Command	Description
adding column (opt.1)	`df['new_col1'] = list`	Adding list to existing DataFrame as a new column at the end
adding column (opt.2)	`df.insert(2, 'new_col2', list)`	Insert list to existing DataFrame as a new column at specific position
filter df	`df.filter()`	Select columns or rows by name, regex
filter df by column values (opt.1)	`df.query("Color == 'Blue'")`	Filter based on a query string
filter df by column values (opt.2)	`df[list_bool]`	Filter by a given list of Booleans, like list_bool = [True, False, True, …]
drop column (opt.1)	`df.drop(columns=['col1', 'col2'])`	Drop column by giving list of column names
drop columns (opt.2)	`df.pop('col_name')`	Drop a column and return the dropped column
drop rows	`df.drop(index=[0,2])`	Drop rows based on list of index
drop NA	`df.dropna()`	Drop the rows where at least one element is missing

```
# will change df!
new_col_list = ['Jan','Feb','Mar']
df['new_col1'] = new_col_list
df
```

```
   Numbers Countries  Color new_col1
0        1        DE   Blue      Jan
1        2        FR    Red      Feb
2        3        IT  Green      Mar
```

```
# will change df!
df.insert(2,'new_col_list2', new_col_list)
df
```

```
   Numbers Countries new_col_list2  Color new_col1
0        1        DE           Jan   Blue      Jan
1        2        FR           Feb    Red      Feb
2        3        IT           Mar  Green      Mar
```

```
# no impact on df, needs to be assigneed
df.filter(items=['Countries','Color'])
```

```
  Countries  Color
0        DE   Blue
1        FR    Red
2        IT  Green
```

```
# no impact on df, needs to be assigneed
df.filter(like='ne')
```

```
  new_col_list2 new_col1
0           Jan      Jan
```

(continues on next page)

(continued from previous page)

```
1           Feb      Feb
2           Mar      Mar
```

```
# no impact on df, needs to be assigneed
df.filter([0,2],axis=0)
```

```
   Numbers Countries new_col_list2  Color new_col1
0        1        DE           Jan   Blue      Jan
2        3        IT           Mar  Green      Mar
```

```
# no impact on df, needs to be assigneed
df.drop(columns=['new_col_list2'])
```

```
   Numbers Countries  Color new_col1
0        1        DE   Blue      Jan
1        2        FR    Red      Feb
2        3        IT  Green      Mar
```

```
# will change df!
df.pop('new_col_list2')
```

```
0    Jan
1    Feb
2    Mar
Name: new_col_list2, dtype: object
```

```
df.drop(index=2)
```

```
   Numbers Countries Color new_col1
0        1        DE  Blue      Jan
1        2        FR   Red      Feb
```