

# PANDAS İLE TEMEL İŞLEMLER



**Bir liste üzerinde Series oluşturma:** içerisine listemizi ekledik. İlk sütundaki sayılar Series'in index'idir. Diğer sütunda listemizin elemanlarını içerir.

```
import pandas as pd
import numpy as np

pd.Series([10,20,30,40,50])
```

```
0    10
1    20
2    30
3    40
4    50
dtype: int64
```

Numpy arraylerinden farkı index bulundurmasıdır. **Numpy array kullanarak Series oluşturma:** index parametresiyle değerlerimizi değiştirdik.

```
data = np.array([10,20,30,40,50])
seri = pd.Series(data, index = [100,101,102,103,104])
seri
```

```
100    10
101    20
102    30
103    40
104    50
dtype: int64
```

Dikkat edilmesi gereken nokta, data uzunluğu ile index uzunluğunun aynı olması.

Sözlük yapısını kullanarak Series oluşturma:

```
data ={'a':0, 'b':1, 'c':2 }  
  
[ ] seri =pd.Series(data)  
seri  
  
a    0  
b    1  
c    2  
dtype: int64
```

Series yapısında verilerimize ulaşma:

İlk iki elemana ulaşma:

```
seri[0:2]  
  
a    0  
b    1  
dtype: int64
```

Aynı anda birden fazla index'e erişmek:

```
[18] seri[['a','b','c']]  
  
a    0  
b    1  
c    2  
dtype: int64  
  
seri['a':'d']  
  
a    0  
b    1  
c    2  
dtype: int64
```

Series yapısında verilerimizi güncelleme:

```
[20] data = np.array([10,20,30,40,50])
      seri = pd.Series(data, index = ['a','b','c','d','e'])
      seri

a    10
b    20
c    30
d    40
e    50
dtype: int64
```

```
seri['a']

10
```

```
[22] seri['a'] = 90
```

```
seri

a    90
b    20
c    30
d    40
e    50
dtype: int64
```

Series yapısında sum(toplama), max(en büyük değer), mean(ortalama), median(medyan) gibi fonksiyonları kullanma:

```
[24] seri

a    90
b    20
c    30
d    40
e    50
dtype: int64
```

```
[25] seri.sum()

230
```

```
[26] seri.max()

90
```

```
seri.mean()

46.0
```

```
seri.median()

40.0
```

## DataFrame oluşturma:

**Liste üzerinden DataFrame oluşturma:** columns argümanı değişkenimizi isimlendirmemizi sağlıyor.

```
[29] isimler = ["Ali", "Buse", "Ceren", "Deniz", "Emre"]
```

```
df = pd.DataFrame(isimler, columns = ["İsimler"])  
df
```

```
İsimler
```

```
0      Ali
```

```
1      Buse
```

```
2      Ceren
```

```
3      Deniz
```

```
4      Emre
```

## Sözlük üzerinden DataFrame oluşturma:

```
[31] data = {'Şehirler': ['Ankara', 'Bursa', 'Denizli', 'Eskişehir'],  
         'Plakalar': [6, 16, 20, 26]}
```

```
data
```

```
{'Plakalar': [6, 16, 20, 26],  
 'Şehirler': ['Ankara', 'Bursa', 'Denizli', 'Eskişehir']}
```

```
df = pd.DataFrame(data, index = ['a', 'b', 'c', 'd'])  
df
```

```
Şehirler Plakalar
```

```
a      Ankara      6
```

```
b       Bursa     16
```

```
c     Denizli     20
```

```
d     Eskişehir    26
```

### DataFrame üzerinde ekleme, silme ve seçme işlemleri:

*Ekleme yapmak için;* df eklemek istediğimiz değişkenin adı ve ardından değişkenin değerlerini giriyoruz. “havadurumu” kolonu eklendi.

```
df["havadurumu"] = pd.Series([33,26,40,36], index=['a', 'b', 'c', 'd'])
df
```

	Sehirler	Plakalar	havadurumu
a	Ankara	6	33
b	Bursa	16	26
c	Denizli	20	40
d	Eskişehir	26	36

*Silme işlemi için;* drop() kullanıyoruz. axis=1 sütun olarak silmemizi sağlıyor. Argüman olarak inplace=True eklersek ana yapısından siliniyor.

```
[34] df.drop("havadurumu",axis = 1, inplace = True)
```

```
[35] df
```

	Sehirler	Plakalar
a	Ankara	6
b	Bursa	16
c	Denizli	20
d	Eskişehir	26

*Satır silmek:* index ismini vererek silebiliriz.

```
[36] df.drop('a', axis = 0, inplace = True)
```

df

	Sehirler	Plakalar
b	Bursa	16
c	Denizli	20
d	Eskişehir	26

### Elemanlara erişme işlemleri:

*LOC:* Tanımlamış olduğumuz şekilde kullanmamızı sağlar. Yani indexlememiz a, b ve c'den oluşuyorsa onların isimlerini kullanarak seçim yapmamızı sağlar. Aynı işlemi sütun bazlıda gerçekleştirebiliriz.

```
df.loc["b":"d"]
```

	Sehirler	Plakalar
b	Bursa	16
c	Denizli	20
d	Eskişehir	26

*ILOC:* Normalde kullandığımız indexleme mantığı ile seçim yapar. Yani belirttiğimiz değerden başlayarak belirttiğimiz uzunluğa kadar olan sayılardan oluşan bir indexleme kullanır.

```
df.iloc[1:3]
```

	Sehirler	Plakalar
c	Denizli	20
d	Eskişehir	26

## Bazı fonksiyonların DataFrame üzerinde çalışması:

```
[42] data={'isimler': ["Ali", "Buse", "Ceren", "Deniz", "Emre", "Furkan", "Gizem", "Hakan"],  
        'Ogr_No': pd.Series([100, 101, 102, 103, 104, 105, 106, 107]),  
        'Fizik': pd.Series([90, 88, 75, 92, 100, 81, 77, 63]),  
        'Matematik': pd.Series([100, 92, 82, 79, 95, 91, 83, 80])}
```

```
[43] df = pd.DataFrame(data)
```

df

	isimler	Ogr_No	Fizik	Matematik
0	Ali	100	90	100
1	Buse	101	88	92
2	Ceren	102	75	82
3	Deniz	103	92	79
4	Emre	104	100	95
5	Furkan	105	81	91
6	Gizem	106	77	83
7	Hakan	107	63	80

```
[46] df['Fizik'].sum()
```

666

```
[48] df['Matematik'].mean()
```

87.75

```
[49] df['Fizik'].min()
```

63

```
[50] df['Matematik'].max()
```

100

***describe ()*** ile hepsini bir arada hesaplayabiliyoruz. Nümerik kolonların her biri için bu bilgileri hesaplayabiliriz.

```
df.describe()
```

	Ogr_No	Fizik	Matematik
count	8.00000	8.000000	8.000000
mean	103.50000	83.250000	87.750000
std	2.44949	11.634309	7.778175
min	100.00000	63.000000	79.000000
25%	101.75000	76.500000	81.500000
50%	103.50000	84.500000	87.000000
75%	105.25000	90.500000	92.750000
max	107.00000	100.000000	100.000000

**Koşullu eleman işlemleri:**

```
df[df.Matematik > 90]["isimler"]
```

```
0      Ali  
1      Buse  
4      Emre  
5      Furkan  
Name: isimler, dtype: object
```

## KAYNAKÇA

Bilgeiř “Herkes için Yapay Zekâ I” eğitimi.