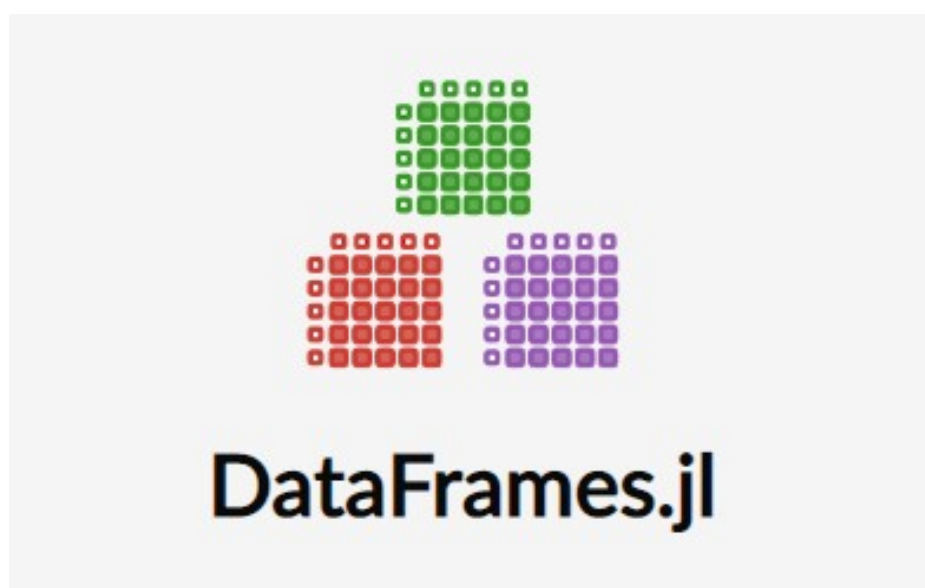
The Julia logo features the word "julia" in a bold, black, lowercase sans-serif font. Above the letters are four colored circles: a blue circle above the 'j', a green circle above the 'i', a red circle above the 'l', and a purple circle above the 'a'.

julia



DataFrames.jl ile Data Wrangling

Bu bölümde veri keşfinden sonra DataFrames.jl ile Data Wrangling aşamalarının adımlarını gerçekleştireceğiz.

Tidy Data -Data Wrangling'in temeli

Tidy veri, veri analizini daha kolay ve daha sezgisel yapmamızı sağlar.DataFrames.jl, verilerimizi düzenlememize yardımcı olabilir.

Dönüşüm Fonksiyonlarının Temel Kullanımı

DataFrames.jl'de bir data frame nesnesinin değişkenlerinin dönüşümlerini gerçekleştirmek için kullanılabilecek beş ana fonksiyonumuz var:

combine:kaynak data frame değişkenlerine uygulanan ve potansiyel olarak gözlemleri birleştiren dönüşümün sonuçları olan değişkenlerle doldurulmuş yeni bir data frame oluşturur;
Summarize Data bölümünün Aggregating variables alt başlığına aittir.

select: Data Frame nesnesinde belirtilen değişkenlerde seçim işlemi yapar.

select!:select ile aynıdır ancak data frame nesnesini kalıcı olarak günceller.

transform: select ile aynıdır ancak data frame'de zaten mevcut olan değişkenleri tutar.Ekstradan hesaplanmış yeni değişken yaratır.

Transform! : transform fonksiyonuyla aynıdır tek fark yapılan işlem kalıcıdır.

Bir dönüşümü(transform) belirtmenin temel yolları şunlardır:

-source_column => transform => target_column_name; Bu senaryoda source_column, transform fonksiyonuna bir argüman olarak iletilir ve target_column_name değişkeninde saklanır.

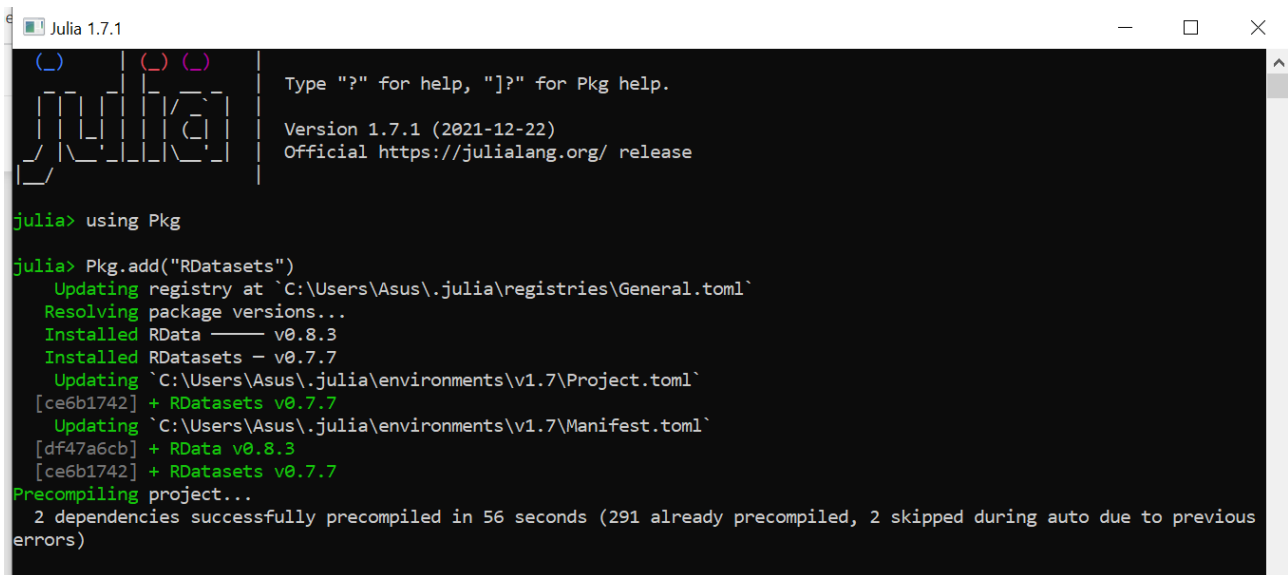
-source_column => transform; Bu senaryoda, source_column'a transform fonksiyonunu uygularız ve hedef değişken adları otomatik olarak oluşturulur.

-source_column => target_column_name, source_column'u target_column_name olarak yeniden adlandırır.

-source_column herhangi bir transform yapmadan kaynak sütunu sonuçta olduğu gibi tutar;

Describe DataFrame

Öncelikle R altındaki veri setlerinden yararlanmak için R Dataset paketi yüklenir.



```
Julia 1.7.1
julia> using Pkg

julia> Pkg.add("RDatasets")
  Updating registry at `C:\Users\Asus\.julia\registries\General.toml`
  Resolving package versions...
  Installed RData v0.8.3
  Installed RDatasets v0.7.7
  Updating `C:\Users\Asus\.julia\environments\v1.7\Project.toml`
 [ce6b1742] + RDatasets v0.7.7
  Updating `C:\Users\Asus\.julia\environments\v1.7\Manifest.toml`
 [df47a6cb] + RData v0.8.3
 [ce6b1742] + RDatasets v0.7.7
  Precompiling project...
  2 dependencies successfully precompiled in 56 seconds (291 already precompiled, 2 skipped during auto due to previous errors)
```

```
Julia 1.7.1
errors)

julia> Pkg.status()
Status `C:\Users\Asus\.julia\environments\v1.7\Project.toml`
[c52e3926] Atom v0.12.36
[336ed68f] CSV v0.9.11
[324d7699] CategoricalArrays v0.10.5
[8f4d0f93] Conda v1.6.0
[717857b8] DSP v0.7.4
[1b08a953] Dash v1.1.1
[a93c6f00] DataFrames v1.3.1
[54a5dec1] DataSkimmer v0.4.1
[dcc97b0b] GeoStats v0.27.0
[7073ff75] IJulia v1.23.2
[e5e0dc1b] Juno v0.8.4
[eadc2687] Pandas v1.5.3
[91a5bcd] Plots v1.25.4
[c3e4b0f8] Pluto v0.17.6
[438e738f] PyCall v1.93.0
[6f40c342] RCall v0.13.12
[ce6b1742] RDatasets v0.7.7
[6d360f66] Tables v1.7.0
[44d3d7a6] Weave v0.10.10
[fdbf4ff8] XLSX v0.7.8
[10745b16] Statistics

julia>
```

İlgili kütüphaneler import edilir.

```
In [1]: using DataFrames
using CSV
using RDatasets
```

R veri setlerinden “mtcars” isimli veri seti import edilir.

```
In [2]: mtcars=dataset("datasets", "mtcars")
```

Out[2]: 32 rows × 12 columns (omitted printing of 4 columns)

	Model	MPG	Cyl	Disp	HP	DRat	WT	QSec
	String31	Float64	Int64	Float64	Int64	Float64	Float64	Float64
1	Mazda RX4	21.0	6	160.0	110	3.9	2.62	16.46
2	Mazda RX4 Wag	21.0	6	160.0	110	3.9	2.875	17.02
3	Datsun 710	22.8	4	108.0	93	3.85	2.32	18.61
4	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44
5	Hornet Sportabout	18.7	8	360.0	175	3.15	3.44	17.02
6	Valiant	18.1	6	225.0	105	2.76	3.46	20.22
7	Duster 360	14.3	8	360.0	245	3.21	3.57	15.84
8	Merc 240D	24.4	4	146.7	62	3.69	3.19	20.0
9	Merc 230	22.8	4	140.8	95	3.92	3.15	22.9
10	Merc 280	19.2	6	167.6	123	3.92	3.44	18.3
11	Merc 280C	17.8	6	167.6	123	3.92	3.44	18.9
12	Merc 450SE	16.4	8	275.8	180	3.07	4.07	17.4
13	Merc 450SL	17.3	8	275.8	180	3.07	3.73	17.6
14	Merc 450SLC	15.2	8	275.8	180	3.07	3.78	18.0
15	Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.25	17.98
16	Lincoln Continental	10.4	8	460.0	215	3.0	5.424	17.82

Şimdi veri keşif ve ön işleme eylemlerine geçebiliriz;

describe:Tüm değişkenler için özet istatistikler sunan fonksiyondur.

```
describe(mtcars)
```

12 rows × 7 columns

	variable	mean	min	median	max	nmissing	eltype
	Symbol	Union...	Any	Union...	Any	Int64	DataType
1	Model	AMC Javelin			Volvo 142E	0	String31
2	MPG	20.0906	10.4	19.2	33.9	0	Float64
3	Cyl	6.1875	4	6.0	8	0	Int64
4	Disp	230.722	71.1	196.3	472.0	0	Float64
5	HP	146.688	52	123.0	335	0	Int64
6	DRat	3.59656	2.76	3.695	4.93	0	Float64
7	WT	3.21725	1.513	3.325	5.424	0	Float64
8	QSec	17.8487	14.5	17.71	22.9	0	Float64
9	VS	0.4375	0	0.0	1	0	Int64
10	AM	0.40625	0	0.0	1	0	Int64
11	Gear	3.6875	3	4.0	5	0	Int64
12	Carb	2.8125	1	2.0	8	0	Int64

Sort Data

sort fonksiyonu belirlenen değişkene göre sıralama yapar.

```
In [9]: ► sort(mtcars,"Cyl")
```

Out[9]: 32 rows × 12 columns (omitted printing of 4 columns)

	Model	MPG	Cyl	Disp	HP	DRat	WT	QSec
	String31	Float64	Int64	Float64	Int64	Float64	Float64	Float64
1	Datsun 710	22.8	4	108.0	93	3.85	2.32	18.61
2	Merc 240D	24.4	4	146.7	62	3.69	3.19	20.0
3	Merc 230	22.8	4	140.8	95	3.92	3.15	22.9
4	Fiat 128	32.4	4	78.7	66	4.08	2.2	19.47
5	Honda Civic	30.4	4	75.7	52	4.93	1.615	18.52
6	Toyota Corolla	33.9	4	71.1	65	4.22	1.835	19.9
7	Toyota Corona	21.5	4	120.1	97	3.7	2.465	20.01
8	Fiat X1-9	27.3	4	79.0	66	4.08	1.935	18.9
9	Porsche 914-2	26.0	4	120.3	91	4.43	2.14	16.7
10	Lotus Europa	30.4	4	95.1	113	3.77	1.513	16.9

Aşağıdaki örnek ise sıralamayı azalana göre yapar.

```
In [10]: sort(mtcars, "Cyl", rev=true)
```

Out[10]: 32 rows × 12 columns (omitted printing of 4 columns)

	Model	MPG	Cyl	Disp	HP	DRat	WT	QSec
	String31	Float64	Int64	Float64	Int64	Float64	Float64	Float64
1	Hornet Sportabout	18.7	8	360.0	175	3.15	3.44	17.02
2	Duster 360	14.3	8	360.0	245	3.21	3.57	15.84
3	Merc 450SE	16.4	8	275.8	180	3.07	4.07	17.4
4	Merc 450SL	17.3	8	275.8	180	3.07	3.73	17.6
5	Merc 450SLC	15.2	8	275.8	180	3.07	3.78	18.0
6	Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.25	17.98
7	Lincoln Continental	10.4	8	460.0	215	3.0	5.424	17.82
8	Chrysler Imperial	14.7	8	440.0	230	3.23	5.345	17.42
9	Dodge Challenger	15.5	8	318.0	150	2.76	3.52	16.87
10	AMC Javelin	15.2	8	304.0	150	3.15	3.435	17.3

Kümülatif ve İstatistiki İşlemler

Aşağıdaki örnekte belli bir değişkene göre kümülatif toplam hesaplatılmıştır.

```
In [13]: select(mtcars, :Cyl => cumsum)
```

Out[13]: 32 rows × 1 columns

	Cyl_cumsum
	Int64
1	6
2	12
3	16
4	22
5	30

Veri Özeti

Değişkenleri Toplama

MPG ve Cyl değişkenlerinin toplam değerleri özet halinde aşağıdaki gibidir.

```
In [15]: ► combine(mtcars, "Cyl"=>sum, "MPG"=>sum)
```

Out[15]: 1 rows × 2 columns

	Cyl_sum	MPG_sum
	Int64	Float64
1	198	642.9

```
In [19]: ► combine(mtcars, [:Cyl, :MPG] .=> maximum)
```

Out[19]: 1 rows × 2 columns

	Cyl_maximum	MPG_maximum
	Int64	Float64
1	8	33.9

Toplama sonuçlarıyla değişkenler ekleme

```
In [30]: ► transform(mtcars, :MPG => mean => :average_MPG)  
select(mtcars, :MPG, :Cyl, :MPG => mean => :average_MPG)
```

Out[30]: 32 rows × 3 columns

	MPG	Cyl	average_MPG
	Float64	Int64	Float64
1	21.0	6	20.0906
2	21.0	6	20.0906
3	22.8	4	20.0906
4	21.4	6	20.0906

Değişkenleri Gruplama

```
In [31]: > groupby(mtcars, :Model)
```

Out[31]: GroupedDataFrame with 32 groups based on key: Model

First Group (1 row): Model = "Mazda RX4"

	Model	MPG	Cyl	Disp	HP	DRat	WT	QSec	VS	AM
	String31	Float64	Int64	Float64	Int64	Float64	Float64	Float64	Int64	Int64
1	Mazda RX4	21.0	6	160.0	110	3.9	2.62	16.46	0	1

:

Last Group (1 row): Model = "Volvo 142E"

	Model	MPG	Cyl	Disp	HP	DRat	WT	QSec	VS	AM
	String31	Float64	Int64	Float64	Int64	Float64	Float64	Float64	Int64	Int64
1	Volvo 142E	21.4	4	121.0	109	4.11	2.78	18.6	1	1

Julia'da Pipe Operatörü

R'da olan pipe operatörü seçeneğini Julia'da da kullanabiliriz. Aşağıda bu kullanıma özgü örneği görebiliyoruz.

```
In [40]: > using Pipe
```

```
In [41]: > @pipe mtcars |>
  filter(:Cyl == 8, _) |>
  groupby(_, :Model) |>
  combine(_, :MPG => mean)
```

Out[41]: 14 rows × 2 columns

	Model	MPG_mean
	String31	Float64
1	Hornet Sportabout	18.7
2	Duster 360	14.3
3	Merc 450SE	16.4
4	Merc 450SL	17.3
5	Merc 450SLC	15.2
6	Cadillac Fleetwood	10.4
7	Lincoln Continental	10.4
8	Chrysler Imperial	14.7
9	Dodge Challenger	15.5
10	AMC Javelin	15.2
11	Camaro Z28	13.3
12	Pontiac Firebird	19.2
13	Ford Pantera L	15.8
14	Maserati Bora	15.0