# VERİ AKTARMA :: REFERANS KAĞIDI

R programının içinde bulunan **tidyverse** kütüphanesi, gelişmiş veri çerçeveleri olan **tibbles** biçiminde depolanan **düzenli veriler** etrafında oluşturulmuştur.

Bu referans kağıdının önyüzü metin dosyalarının **readr** ile R'ye nasıl aktarılacağını gösterir.

Arka yüzü ise, **tibble** ile nasıl tibble oluşturulacağını ve **tidyr** ile düzenli verilerin nasıl düzenleneceğini gösterir.

### DİĞER VERİ TÜRLERİ

Diğer dosya türlerini R'a aktarmak için aşağıdaki paketlerden biri kullanılabilir.

- **haven** - SPSS, Stata, ve SAS dosyaları
- **readxl** - excel dosyaları (.xls ve .xlsx)
- **DBI** - veritabanları
- **jsonlite** - json
- **xml2** - XML
- **httr** - Web APIs
- **rvest** - HTML (Web Scraping)

## Veriyi Kaydetme

Bir R nesnesi olan x'i , bir **dosya yoluna** şu şekilde kaydedebilirsiniz:

**Virgülle ayrılmış dosya**
**write_csv(**x, path, na ="NA", append =FALSE, col_names =!append**)**

**Rasgele bir ayraç içeren dosya**
**write_delim(**x, path, delim =" ", na ="NA", append =FALSE, col_names =!append**)**

**Excel için CSV**
**write_excel_csv(**x, path, na ="NA", append = FALSE, col_names =!append**)**

**Dizeden dosyaya**
**write_file(**x, path, append =FALSE**)**

**Dize vektöründen dosyaya, satır başına bir öğe**
**write_lines(**x,path, na ="NA", append =FALSE**)**

**RDS dosyasına nesne olarak**
**write_rds(**x, path, compress =c("none", "gz", "bz2", "xz"), ...**)**

**Sekmeyle ayrılmış dosya**
**write_tsv(**x, path, na ="NA", append =FALSE, col_names =!append**)**

## Tablo Halinde Verileri Okuma – Bu fonksiyonlar ortak argümanları paylaşır:

**read_*(**file, col_names =TRUE, col_types =NULL, locale =default_locale(), na =c("", "NA"), quoted_na =TRUE, comment ="", trim_ws =TRUE, skip =0, n_max =Inf, guess_max =min(1000, n_max), progress =interactive()**)**

```
a,b,c
1,2,3
4,5,NA
```

| A | B | C |
|---|---|---|
| 1 | 2 | 3 |
| 4 | 5 | NA |

**Virgülle Ayrılmış Dosyalar**
**read_csv(**"file.csv"**)**
Yapabilirsiniz file.csvrun:
write_file(x = "a,b,c\n1,2,3\n4,5,NA", path = "file.csv")

```
a;b;c
1;2;3
4;5;NA
```

| A | B | C |
|---|---|---|
| 1 | 2 | 3 |
| 4 | 5 | NA |

**Noktalı Virgülle Ayrılmış Dosyalar**
**read_csv2(**"file2.csv"**)**
write_file(x = "a;b;c\n1;2;3\n4;5;NA", path = "file2.csv")

```
a|b|c
1|2|3
4|5|NA
```

| A | B | C |
|---|---|---|
| 1 | 2 | 3 |
| 4 | 5 | NA |

**Herhangi Bir Ayraç İçeren Dosyalar**
**read_delim(**"file.txt", delim ="|"**)**
write_file(x = "a|b|c\n1|2|3\n4|5|NA", path ="file.txt")

```
a b c
1 2 3
4 5 NA
```

| A | B | C |
|---|---|---|
| 1 | 2 | 3 |
| 4 | 5 | NA |

**Sabit Genişlikli Dosyalar**
**read_fwf(**"file.fwf", col_positions =c(1, 3, 5)**)**
write_file(x = "a b c\n1 2 3\n4 5 NA", path = "file.fwf")

**Sekmeyle Ayrılmış Dosyalar**
**read_tsv(**"file.tsv"**)** Ayrıca **read_table().**
write_file(x = "a\tb\tc\n1\t2\t3\n4\t5\tNA", path = "file.tsv")

### FAYDALI ARGÜMANLAR

```
a,b,c
1,2,3
4,5,NA
```

**Örnek Dosya**
write_file("a,b,c\n1,2,3\n4,5,NA","file.csv")
f <-"file.csv"

| A | B | C |
|---|---|---|
| 1 | 2 | 3 |
| 4 | 5 | NA |

**Başlık Yok**
read_csv(f, **col_names = FALSE**)

| x | y | z |
|---|---|---|
| A | B | C |
| 1 | 2 | 3 |
| 4 | 5 | NA |

**Başlık Var**
read_csv(f, **col_names = c("x", "y", "z")**)

| 1 | 2 | 3 |
|---|---|---|
| 4 | 5 | NA |

**Satırları atlama**
read_csv(f, **skip =1**)

| A | B | C |
|---|---|---|
| 1 | 2 | 3 |

**Bir alt küme olarak veriyi alma**
read_csv(f, **n_max =1**)

| A | B | C |
|---|---|---|
| NA | 2 | 3 |
| 4 | 5 | NA |

**Kayıp Değerler**
read_csv(f, **na = c("1", ".")**)

## Tablo Halinde Olmayan Verileri Okuma

**Bir dosyayı tek bir kod parçası olarak okuma**
**read_file(**file, locale =default_locale()**)**

**Her satırı kendi kod parçasına okuma**
**read_lines(**file, skip = 0, n_max = -1L, na = character(), locale = default_locale(), progress = interactive()**)**

**Apache log dosyalarını okuma**
**read_log(**file, col_names = FALSE, col_types = NULL, skip = 0, n_max = -1, progress = interactive()**)**

**Bir dosyayı bir satır vektörü olarak okuma**
**read_file_raw(**file**)**

**Her satırı bir satır vektörü olarak okuma**
**read_lines_raw(**file, skip = 0, n_max = -1L, progress = interactive()**)**

## Veri Tipleri

readr fonksiyonları her bir sütunun nasıl bir tür olduğunu anlar ve uygun şekile dönüştürür (ancak dizeleri otomatik olarak faktörlere DÖNÜŞTÜRMEZ).

Aşağıdaki mesaj, çıktıda yer alan her sütunun türünü gösterir.

```
## Parsed with column specification:  ## cols(
##   age = col_integer(),
##   sex = col_character(),
##   earn = col_double()
## )
```

age bir tamsayıdır
sex bir karakterdir
earn bir nümerik değerdir

1. Sorunları teşhis etmek için **problems()** fonksiyonu kullanılır.
*x <- read_csv("file.csv"); problems(x)*

2. Ayrıştırmak için col_function fonksiyonu kullanılır.
- **col_guess()** - varsayılan
- **col_character()**
- **col_double()**, **col_euro_double()**
- **col_datetime(**format = ""**)** Ayrıca **col_date(**format = ""**)**, **col_time(**format = ""**)**
- **col_factor(**levels, ordered =FALSE**)**
- **col_integer()**
- **col_logical()**
- **col_number()**, **col_numeric()**
- **col_skip()**

```
x <- read_csv("file.csv", col_types = cols(
  A = col_double(),
  B = col_logical(),
  C = col_factor()))
```

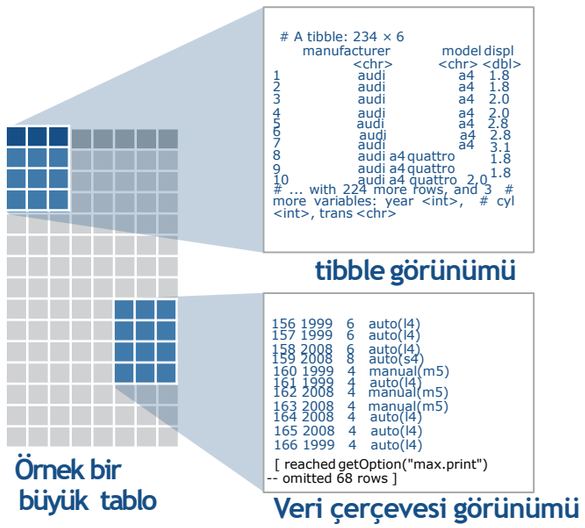
3. Aksi takdirde, karakter vektörleri gibi okunur ve ardından parse_ function fonksiyonuyla incelenebilir.
- **parse_guess()**
- **parse_character()**
- **parse_datetime()** Ayrıca **parse_date()** ve **parse_time()**
- **parse_double()**
- **parse_factor()**
- **parse_integer()**
- **parse_logical()**
- **parse_number()**

```
x$A <- parse_number(x$A)
```

# Tibbles – gelişmiş bir veri çerçevesi

**tibble** paketi, tablo halinde verilen depolamak için yeni bir S3 sınıfı, tibble sağlar. Tibbles, veri çerçevesi gibidir ancak iyileştirilmiş üç özelliği vardır:

- **Alt küme** - [ her zaman yeni bir tibble verir, [[ ve $ her zaman bir vektör döndürür.
- **Kısmi eşleşme yapılamaz** - Alt küme oluştururken sütun adlarını kullanmalısınız.
- **Görünüm** – Bir tibble yazdırdığınızda, R size verileri tek parça olacak şekilde görüntüler.




```
# A tibble: 234 × 6
   manufacturer       model displ
          <chr>       <chr> <dbl>
1          audi          a4   1.8
2          audi          a4   1.8
3          audi          a4   2.0
4          audi          a4   2.0
5          audi          a4   2.8
6          audi          a4   2.8
7          audi          a4   3.1
8          audi a4 quattro     1.8
9          audi a4 quattro     1.8
10         audi a4 quattro     2.0
# ... with 224 more rows, and 3 more variables: year <int>, cyl <int>, trans <chr>
```

```
156 1999   6   auto(l4)
157 1999   6   auto(l4)
158 2008   6   auto(l4)
159 2008   8   auto(s4)
160 1999   4   manual(m5)
161 1999   4   auto(l4)
162 2008   4   manual(m5)
163 2008   4   manual(m5)
164 2008   4   auto(l4)
165 2008   4   auto(l4)
166 1999   4   auto(l4)
 [ reached getOption("max.print")
-- omitted 68 rows ]
```

- Varsayılan görünümü seçeneklerle kontrol edin:
  **options(**tibble.print_max = n, tibble.print_min = m, tibble.width = Inf**)**
- Tüm veri setini **View()** veya **glimpse()** ile görüntüleyin
- Veri çerçevesine **as.data.frame()** ile geri döndürün

### İKİ ŞEKİLDE BİR TIBBLE OLUŞTURUN

**tibble(...)**
Sütunlara göre yapın.

```
tibble(x = 1:3, y = c("a", "b", "c"))
```

Her ikisi de bu tibble'ı çıkarır.

```
A tibble: 3 × 2
      x     y
  <int> <chr>
1     1     a
2     2     b
3     3     c
```

**tribble(...)** .

```
tribble( ~x,  ~y,
         1,  "a",
         2,  "b",
         3,  "c")
```

**as_tibble(**x, ...**)** Veri çerçevesinden tibble' a dönüştürün.

**enframe(**x, name = "name", value = "value"**)**
İsimleştirilmiş vektörleri tibble' a dönüştürün.

**is_tibble(**x**)** x değişkeninin tibble olup-olmadığını test edin.

RStudio

# tidyr ile Veri Düzenleme

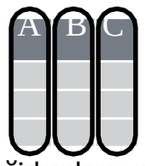
**Düzenli veriler(tidy data),** tablo şeklindeki verileri düzenlemenin bir yoludur. Paketler arasında tutarlı bir veri yapısı sağlar.

Eğer bir tablo düzenli ise:

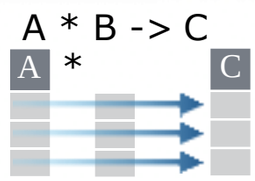

Her **değişken** kendi **sütunundadır**

Her **gözlem** veya **durum**, kendi **satırındadır**

Düzenli veri:




Değişkenlere vektör olarak erişmeyi kolaylaştırır




Vektörize işlemler sırasında durumları korur

# Verileri Yeniden Şekillendirme - bir tablodaki değerlerin düzenini değiştirin

Bir tablonun değerlerini yeni bir düzende yeniden düzenlemek için **gather()** ve **spread()** kullanın.

**gather(**data, key, value, ..., na.rm =FALSE, convert =FALSE, factor_key =FALSE**)**

gather() sütun adlarını bir **anahtar** sütuna taşır, sütun **değerlerini** tek bir değer sütununda toplar.

table4a

| country | 1999 | 2000 |
|---|---|---|
| A | 0.7K | 2K |
| B | 37K | 80K |
| C | 212K | 213K |

→

| country | year | cases |
|---|---|---|
| A | 1999 | 0.7K |
| B | 1999 | 37K |
| C | 1999 | 212K |
| A | 2000 | 2K |
| B | 2000 | 80K |
| C | 2000 | 213K |

Anahtar Değer

*gather(table4a, `1999`, `2000`, key ="year", value ="cases")*

**spread(**data, key, value, fill =NA, convert =FALSE, drop =TRUE, sep =NULL**)**

spread() bir **anahtar** sütunun tekil değerlerini sütun adlarına taşır, bir **değer** sütununun **değerlerini** yeni sütuna yayar.

table2

| country | year | type | count |
|---|---|---|---|
| A | 1999 | cases | 0.7K |
| A | 1999 | pop | 19M |
| A | 2000 | cases | 2K |
| A | 2000 | pop | 20M |
| B | 1999 | cases | 37K |
| B | 1999 | pop | 172M |
| B | 2000 | cases | 80K |
| B | 2000 | pop | 174M |
| C | 1999 | cases | 212K |
| C | 1999 | pop | 1T |
| C | 2000 | cases | 213K |
| C | 2000 | pop | 1T |

→

| country | year | cases | pop |
|---|---|---|---|
| A | 1999 | 0.7K | 19M |
| A | 2000 | 2K | 20M |
| B | 1999 | 37K | 172M |
| B | 2000 | 80K | 174M |
| C | 1999 | 212K | 1T |
| C | 2000 | 213K | 1T |

Anahtar Değer

*spread(table2, type, count)*

# Kayıp Verileri Yönetmek

**drop_na(**data, ...**)**
NA içeren satır ve sütunları çıkarın.

| x1 | x2 |
|---|---|
| A. | 1 |
| B. | NA |
| C. | NA |
| D. | 3 |
| E. | NA |

→

| x1 | x2 |
|---|---|
| A | 1 |
| D | 3 |

*drop_na(x, x2)*

**fill(**data, ..., .direction =c("down", "up"**)**
Satırları NA olmayan bir önceki değer ile doldurun.

| x1 | x2 |
|---|---|
| A | 1 |
| B | NA |
| C | NA |
| D | 3 |
| E | NA |

→

| x1 | x2 |
|---|---|
| A | 1 |
| B | 1 |
| C | 1 |
| D | 3 |
| E | 3 |

*fill(x, x2)*

**replace_na(**data, replace =list(), ...**)**
NA' ları belli bir değere göre değiştirir.

| x1 | x2 |
|---|---|
| A | 1 |
| B | NA |
| C | NA |
| D | 3 |
| E | NA |

→

| x1 | x2 |
|---|---|
| A | 1 |
| B | 2 |
| C | 2 |
| D | 3 |
| E | 2 |

*replace_na(x, list(x2 =2))*

# Tabloları Genişletme - değer kombinasyonları ile tabloları hızlı bir şekilde oluşturun

**complete(**data, ..., fill =list()**)**
Listelenen değişkenlerin değerlerinin eksik kombinasyonlarını verilere ekler ...
*complete(mtcars, cyl, gear, carb)*

**expand(**data, ...**)**
Listelenen değişkenlerin değerlerinin tüm olası kombinasyonlarını içeren yeni bir tablo oluşturur ...
*expand(mtcars, cyl, gear, carb)*

# Hücrelere Bölme

Hücreleri ayrı ayrı, izole değerlere bölmek veya birleştirmek için bu fonksiyon kullanılır .

**separate(**data, col, into, sep ="[^[:alnum:]]+", remove =TRUE, convert =FALSE, extra ="warn", fill ="warn", ...**)**

Birkaç sütun yapmak için bir sütundaki her bir hücreyi bölebilirsiniz.

table3

| country | year | rate |
|---|---|---|
| A | 1999 | 0.7K/19M |
| A | 2000 | 2K/20M |
| B | 1999 | 37K/172M |
| B | 2000 | 80K/174M |
| C | 1999 | 212K/1T |
| C | 2000 | 213K/1T |

→

| country | year | cases | pop |
|---|---|---|---|
| A | 1999 | 0.7K | 19M |
| A | 2000 | 2K | 20M |
| B | 1999 | 37K | 172 |
| B | 2000 | 80K | 174 |
| C | 1999 | 212K | 1T |
| C | 2000 | 213K | 1T |

*separate(table3, rate, sep ="/", into =c("cases", "pop"))*

**separate_rows(**data, ..., sep ="[^[:alnum:].]+", convert =FALSE**)**

Birkaç satır yapmak için bir sütundaki her bir hücreyi bölebilirsiniz.

table3

| country | year | rate |
|---|---|---|
| A | 1999 | 0.7K/19M |
| A | 2000 | 2K/20M |
| B | 1999 | 37K/172M |
| B | 2000 | 80K/174M |
| C | 1999 | 212K/1T |
| C | 2000 | 213K/1T |

→

| country | year | rate |
|---|---|---|
| A | 1999 | 0.7K |
| A | 1999 | 19M |
| A | 2000 | 2K |
| A | 2000 | 20M |
| B | 1999 | 37K |
| B | 1999 | 172M |
| B | 2000 | 80K |
| B | 2000 | 174M |
| C | 1999 | 212K |
| C | 1999 | 1T |
| C | 2000 | 213K |
| C | 2000 | 1T |

*separate_rows(table3, rate, sep ="/")*

**unite(**data, col, ..., sep ="_", remove =TRUE**)**

Tek bir sütun oluşturmak için farklı hücreleri bir sütundan birleştirebilirsiniz...

table5

| country | century | year |
|---|---|---|
| Afghan | 19 | 99 |
| Afghan | 20 | 00 |
| Brazil | 19 | 99 |
| Brazil | 20 | 00 |
| China | 19 | 99 |
| China | 20 | 00 |

→

| country | year |
|---|---|
| Afghan | 1999 |
| Afghan | 2000 |
| Brazil | 1999 |
| Brazil | 2000 |
| China | 1999 |
| China | 2000 |

*unite(table5, century, year, col ="year", sep ="")*