# برگه تقلب بسته تایدی ورس:: وارد کردن داده ها

مشارکت در ترجمه : وحید فرجی جبه دار، رضا مظلومی

بسته تایدی ورس R بر اساس نسخه تایدی دیتا ایجاد شده که در تیبلز (tibbles)ذخیره می شوند و چارچوب های داده را بهبود می دهد.

صفحه اول این برگه به شما نشان می دهد که چگونه متن ها در نرم افزار R با اسفاده از بسته readr خوانده می شوند.

در صفحه دوم نحوه ایجاد نوارها با کمک tibble و چارچوب مرتب سازی داده با tider نشان داده می شود

انواع دیگر داده :
واز بسته های زیر برای وارد کردن سایر فایل ها می توانید استفاده کنید:

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

## ذخیره داده :

برای ذخیره کردن فایل در مسیر با فرمت ایکس می توانید از دستورهای زیر استفاده کنید :

**کاما جداکننده فایل:**

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

**فایل با جداکننده دلخواه:**

**write_delim(**x, path, delim = " ", na = "NA", append = FALSE, col_names = !append**)**

**CSVبرای اکسل**

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

**رشته به فایل:**

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

بردار رشته به فایل، یک عضو در هرخط

**write_lines(**x,path, na = "NA", append = FALSE**)**

**شی به فایل RDS**

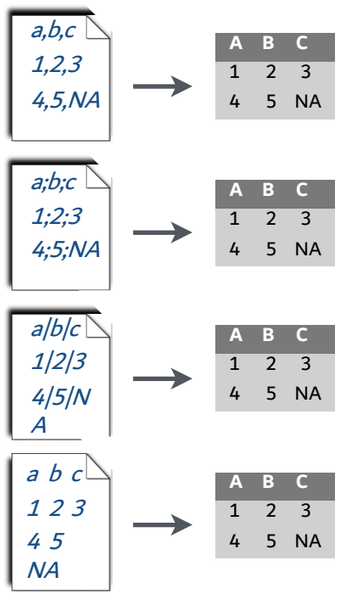
**write_rds(**x, path, compress = c("none", "gz", "bz2", "xz"), ...**)**

**تب جداکننده فایلها:**

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

خواندن داده جدولی - این توابع گزاره های متداول را به اشتراک می گذارند:

```
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())
```




**کاما جداکننده فایل:**

**read_csv(**"file.csv"**)**
To make file.csv run:
write_file(x = "a,b,c\n1,2,3\n4,5,NA", path = "file.csv")

**جدا کننده فایل توسط نقطه ویرگول**

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

**فایل به همراه نوع جداکننده**

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

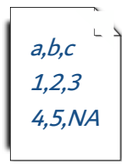
**فایل با عرض ثابت**

**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")

**تب جدا کننده فایل:**

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

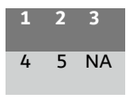
### گزاره های کاربردی




**فایل نمونه:**

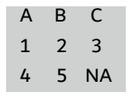
write_file("a,b,c\n1,2,3\n4,5,NA","file.csv")

f <- "file.csv"



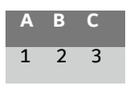

**خطوط پرش**

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




**بدون سر برگ**

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

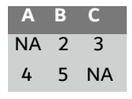



**خوانش از زیرمجموعه**

read_csv(f, **n_max = 1**)


**سرصفحه ارائه دهنده**

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




**داده های گمشده**

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

## خواندن داده غیر جدولی

**خواندن فایل در یک رشته**

read_file(file, locale = default_locale())

**خواندن فایل درون بردار سطری**

read_file_raw(file)

**خواندن هر خط درون رشته خودش**

read_lines(file, skip = 0, n_max = -1L, na = character(), locale = default_locale(), progress = interactive())

**خواندن هر خطر در بردار سطری**

read_lines_raw(file, skip = 0, n_max = -1L, progress = interactive())

**آپاچی با استفاده از فرمت لاگ زیر فایل را می خواند**

read_log(file, col_names = FALSE, col_types = NULL, skip = 0, n_max = -1, progress = interactive())

## انواع داده :

توابع خوانش آر یا readr نوع هر ستون را حدس میزند و در صورتی که صحیح باشند تبدیل را انجام می دهد.( اما به صورت خودکار رشته ها را به عوامل تبدیل نمی کنند.)

در بخش نتیجه نوع هر ستون را مشخص می کند.

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

سن
متغیر عدد صحیح

جنسیت متغیر کاراکتر

درآمد ترکیبی از هر دو(numeric

1.از **problems()** برای شناسایی مساله استفاده کنید

***x <- read_csv("file.csv"); problems(x)***

2. از col_ function برای هدایت تجزیه و تقسیم استفاده کنید

- **col_guess()** - the default
- **col_character()**
- **col_double()**, **col_euro_double()**
- **col_datetime(**format = ""**)** Also **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.در غیر این صورت مشخصه بردار را میخواند سپس با parse_ function آن را تجزیه می کند.

- **parse_guess()**
- **parse_character()**
- **parse_datetime()** Also **parse_date()** and **parse_time()**
- **parse_double()**
- **parse_factor()**
- **parse_integer()**
- **parse_logical()**
- **parse_number()**

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



**@Persian CheetSheet**

**Translated by Vahid Faraji Jobehdar, Reza Mazloomi**
**vfaraji89@gmail.com** **rezamazloumikia@gmail.com**

# تقسیم سلول ها

از این توابع برای تقسیم یا ترکیب سلولها به مقادیر جدا شده جداگانه استفاده میکنیم.

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

برای ساختن چندین ستون ، هر سلول را در یک ستون جدا میکند.

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**)**

برای ساختن چندین سطر ، هر سلول را در یک ستون جدا کنید

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**)**

چند ستون را به یک ستون ادغام می کند

table5

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

→

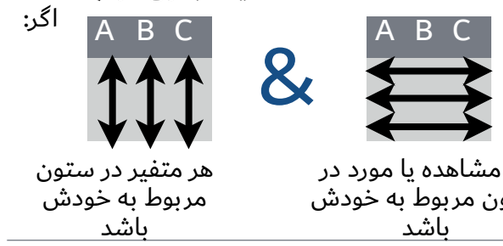
| country | year |
|---|---|
| Afghan | 1999 |
| Afghan | 2000 |
| Brazil | 1999 |
| Brazil | 2000 |
| China | 1999 |
| China | 2000 |

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

## مرتب کردن داده ها به وسیله tidyr

مرتب کردن دادها روشی برای سازماندهی داده های جدولی است. این یک ساختار داده ثابت را در بین بسته ها فراهم می کند.

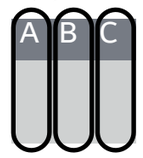
یک جدول مرتب است اگر:



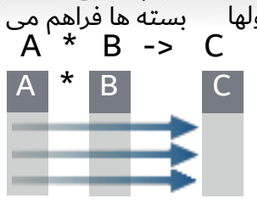

هر متغیر در ستون مربوط به خودش باشد

هر مشاهده یا مورد در ستون مربوط به خودش باشد

داده مرتب شده




دسترسی به متغیرها




نمونه ها در طول عملیات برداری حفظ می شود

# تغییر شکل داده- تغیر طرح و چیدمان متغیرها در جدول

از ()gather و ()spread برای شناسایی مقادیر جدول به یک طرح جدید استفاده کرد.

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

()spread متغیر یکتایی از ستون کلیدی را به اسامی ستون های جابجای می کند ، مقادیر ارزش یک ستون را در ستونهای جدید پخش می کند

| | | | |
|---|---|---|---|
| A | 1999 | cases | 0.7K |
| A | 1999 | pop | 19M |
| A | 200 | 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 |

کلید ارزش

→

| 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 |

*spread(table2, type, count)*

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

()gather اسامی ستون ها را به سمت ستون های کلیدی جابجا خواهد کرد، جمع آوری ارزش های هریک از ستون ها به یک ستون را انجام می دهد.

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 |

کلید ارزش

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

## مقادیر گمشده را کنترل کنید

**drop_na(**data, ...**)**

مقادیر ناموجود در ستون ها را شناسایی می کند

x

| 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")**)**

مقادیر خالی را در ستون ها را با استفاده از الگوی مقادیر کامل می کند.

x

| 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(), ...**)**

جایگزینی مقادیر خالی با ستون ها.

| 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))*

## جداول صفحه گسترده : به سرعت جداول با ترکیبی از مقادیر ایجاد کنید

**complete(**data, ..., fill = list()**)**

به داده های ترکیبی از دست رفته ارزش های متغیر لیست شده را اضافه می کند.

*complete(mtcars, cyl, gear, carb)*

**expand(**data, ...**)**

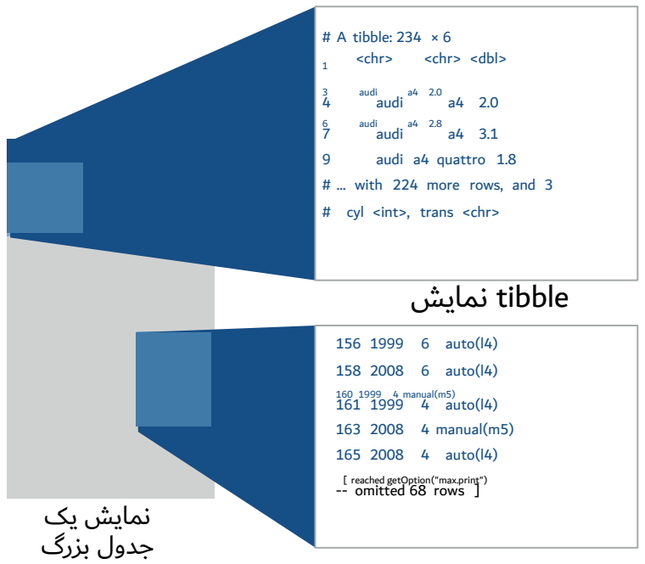
Tibble جدیدی از تمامی ترکیبات ارزش های متغیرهای لیست شده را ایجاد می کند.

*expand(mtcars, cyl, gear, carb)*

# چارچوب داده پیشرفته: Tibble

بسته Tibbleنوع S3جدیدی را برای ذخیره داده های تابلویی ایجاد می کند. Tibbleنوع چارچوب داده را ازبین می برد اما 3 رفتار را بهبود می بخشد:

زیرمجموعه : [ همیشه به یک مضمون جدید برمی گرداند، [ و $ همیشه به یک بردار برمی گرداند.
عدم تطبیق جزیی : باید از نام کل ستون زمانی که زیرمجموعه میدهید استفاده می کنید.
نمایش دادن- زمانی که از یک Tibble پرینت میگیرید، R نمای مختصری از داده که متناسب با یک صفحه است فراهم می کند.




نمایش tibble

نمایش یک جدول بزرگ

- ظاهر پیش فرض را با گزینه ها کنترل کنید:

options(tibble.print_max = n, tibble.print_min = m, tibble.width = Inf)

- مجموعه داده ها را با View() یا glimpse() ببینید.
- با as.data.frame() به چارچوب داده برمیگردید.

## یک tibble را به دو روش می توان ایجاد کرد:

tibble(...)
ساخت ستون ها :
*tibble(x = 1:3, y = c("a", "b", "c"))*

tribble(...)
ساخت سطرها
*tribble( ~x, ~y, 1, "a", 2, "b", 3, "c")*

هر دو این tibble را می سازند

A tibble: 3 × 2

| x | y |
|---|---|
| 1 | a |
| 2 | b |
| 3 | c |

as_tibble(x, ...) چارچوب را به tibbleتبدیل می کند.

enframe(x, name = "name", value = "value")
تبدیل اسامی بردار به tibble

is_tibble(x)تست می کند که آیا xیک tibbleاست.


**مشارکت در ترجمه :** وحید فرجی جبه دار، رضا مظلومی





**@Persian CheetSheet**

**Translated by Vahid Faraji Jobehdar, Reza Mazloomi**
**vfaraji89@gmail.com rezamazloumikia@gmail.com**