

dplyr 1.0.0

2022-08-26



# Contents

		<b>5</b>
	.....	5
	.....	5
	.....	5
	.....	6
	.....	6
	.....	6
<b>1</b>	<b>select(), rename_with() relocate()</b>	<b>7</b>
1.1	.....	7
1.2	.....	7
1.3	.....	7
1.4	.....	9
<b>2</b>	<b>across()</b>	<b>11</b>
2.1	.....	11
2.2	.....	11
2.3	.....	11
2.4	.....	14
<b>3</b>	<b>rowwise()</b>	<b>15</b>
3.1	.....	15
3.2	.....	15
3.3	.....	15
3.4	.....	17

<b>4</b>	<b>summarise()</b>	<b>19</b>
4.1	. . . . .	19
4.2	. . . . .	19
4.3	. . . . .	19
4.4	. . . . .	21
<b>5</b>	<b>, rows_*()</b>	<b>23</b>
5.1	. . . . .	23
5.2	. . . . .	23
5.3	. . . . .	23
5.4	. . . . .	26
		<b>29</b>
	. . . . .	29
	. . . . .	31
e	. . . . .	31
e	. . . . .	32
e	. . . . .	33
		<b>35</b>

---

dplyr - [dplyr 1.0.0](#), [tidyverse](#), [R](#).  
[dplyr 1.0.0](#), [tidyverse](#), [R](#).  
[dplyr 1.0.0](#), [tidyverse](#), [R](#).  
[dplyr 1.0.0](#), [tidyverse](#), [R](#).

- dplyr 1.0.0: select, rename, relocate
- dplyr 1.0.0: working across columns
- dplyr 1.0.0: working within rows
- dplyr 1.0.0: new summarise() features
- dplyr 1.0.0: last minute additions

“dplyr 1.0.0” [tidyverse](#), [R](#). ..  
“R Excel”, [tidyverse](#), [R](#). ..

5 1 2 .  
, , .  
, ,  
. “ ”, 5 - 10 .

, 2008 .  
 - Netpeak.  
 R : ryandexdirect, rfacebookstat, timeperiodsR,  
 rvkstat . CRAN  
 130 000 .  
 “ R - ”.  
 Telegram YouTube R4marketing.  
 , Netpeak Journal.  
 , Go-  
 Analytics, Analyze, eCommerce, 8P .  
 2016 R .

1. select(), rename\_\_with() relocate()
2. across()
3. rowwise()
4. summarise()
5. , rows\_\_\*()

, , .  
 :

# Chapter 1

## select(), rename\_with() relocate()

### 1.1

‘select’, ‘rename’, ‘relocate’  
: select(), rename\_with(), relocate(),  
any\_of(), all\_of().  
“dplyr 1.0.0: select, rename, relocate”.

### 1.2

### 1.3

```
#devtools::install_github("tidyverse/dplyr")
library(dplyr, warn.conflicts = FALSE)

# rename
#
df1 <- tibble(a = 1:5, a = 5:1, .name_repair = "minimal")
df1

df1 %>% rename(b = 2)
```

```

# select
#
df2 <- tibble(x1 = 1, x2 = "a", x3 = 2, y1 = "b", y2 = 3, y3 = "c", y4 = 4)

#
df2 %>% select(is.numeric)
#
df2 %>% select(!is.character)

#
#           ,           X
df2 %>% select(starts_with("x") & is.numeric)

#
#           any_of  all_of
vars <- c("x1", "x2", "y1", "z")
df2 %>% select(any_of(vars))

df2 %>% select(all_of(vars))

#           rename_with
df2 %>% rename_with(toupper)

df2 %>% rename_with(toupper, starts_with("x"))

df2 %>% rename_with(toupper, is.numeric)

# relocate
df3 <- tibble(w = 0, x = 1, y = "a", z = "b")
#           y, z
df3 %>% relocate(y, z)
#
df3 %>% relocate(is.character)

#           w      y
df3 %>% relocate(w, .after = y)
#           w      y
df3 %>% relocate(w, .before = y)
#           w
df3 %>% relocate(w, .after = last_col())

```



1.4

```
iris,
:

1.      Width.
2.      relocate()
3.      rename_with()
```



## Chapter 2

# across()

### 2.1

```
mutate() summarise() across(),
*_at() , *_if(), *_all(). across()
```

“dplyr 1.0.0: working across columns”.

### 2.2

### 2.3

```
# devtools::install_github("tidyverse/dplyr")
library(dplyr, warn.conflicts = FALSE)

#
df <- tibble(g1 = as.factor(sample(letters[1:4],size = 10, replace = T )),
             g2 = as.factor(sample(LETTERS[1:3],size = 10, replace = T )),
             a = runif(10, 1, 10),
             b = runif(10, 10, 20),
             c = runif(10, 15, 30),
             d = runif(10, 1, 50))

#
##
##
```

```

df %>%
  group_by(g1, g2) %>%
  summarise(a = mean(a), b = mean(b), c = mean(c), d = mean(c))

#
##
##                                select()
###                                a    d
df %>%
  group_by(g1, g2) %>%
  summarise(across(a:d, mean))

###
df %>%
  group_by(g1, g2) %>%
  summarise(across(is.numeric, mean))

# #####
#
#          accros

## .cols -          ,      ,      ,      ,
## .fns -      ,

##
starwars %>%
  summarise(across(is.character, n_distinct))

##
starwars %>%
  group_by(species) %>%
  filter(n() > 1) %>%
  summarise(across(c(sex, gender, homeworld), n_distinct))

##          accross
starwars %>%
  group_by(homeworld) %>%
  filter(n() > 1) %>%
  summarise(across(is.numeric, mean, na.rm = TRUE),
            n = n())

# #####
#          accross                                _at, _if, _all

## 1. accross                                summarise

```

```
##
df %>%
  group_by(g1, g2) %>%
  summarise(
    across(is.numeric, mean),
    across(is.factor, nlevels),
    n = n(),
  )

##
starwars %>%
  group_by(species) %>%
  summarise(across(is.character, n_distinct),
            across(is.numeric, mean),
            across(is.list, length),
            n = n()
  )

## 2.                                dplyr,
## 3.                                if_, at_,
## 4. accross                        vars

# #####
#                                accross

##                                _at, _if, _all
### accross                      :
### *_if()                      .
### *_at()                      vars().
### *_all(),                    everything()

##
df <- tibble(y_a = runif(10, 1, 10),
             y_b = runif(10, 10, 20),
             x   = runif(10, 15, 30),
             d   = runif(10, 1, 50))

### _if accross
df %>% mutate_if(is.numeric, mean, na.rm = TRUE)
# ->
df %>% mutate(across(is.numeric, mean, na.rm = TRUE))

### _at accross
df %>% mutate_at(vars(c(x, starts_with("y"))), mean, na.rm = TRUE)
# ->
```

```
df %>% mutate(across(c(x, starts_with("y")), mean, na.rm = TRUE))

### _all accross
df %>% mutate_all(mean, na.rm = TRUE)
# ->
df %>% mutate(across(everything(), mean, na.rm = TRUE))
```

## 2.4

iris.

1. across(), Length
  2. , Sepal
- Species.

## Chapter 3

# rowwise()

### 3.1

`rowwise()`, `dplyr`.

`apply`

“dplyr 1.0.0: working within rows”.

### 3.2

### 3.3

```
# devtools::install_github("tidyverse/dplyr")
library(dplyr)

# test data
df <- tibble(
  student_id = 1:4,
  test1 = 10:13,
  test2 = 20:23,
  test3 = 30:33,
  test4 = 40:43
)

df
```

```

#
df %>% mutate(avg = mean(c(test1, test2, test3, test4)))

#           rowwise
rf <- rowwise(df, student_id)
rf

rf %>% mutate(avg = mean(c(test1, test2, test3, test4)))

#           c_across
rf %>% mutate(avg = mean(c_across(starts_with("test"))))

# #####
#
df %>% mutate(total = test1 + test2 + test3 + test4)

#
df %>% mutate(avg = (test1 + test2 + test3 + test4) / 4)

#
#
df %>% mutate(
  min = pmin(test1, test2, test3, test4),
  max = pmax(test1, test2, test3, test4),
  string = paste(test1, test2, test3, test4, sep = "-")
)
#           rowwise
#   rowwise

# #####
#
df <- tibble(
  x = list(1, 2:3, 4:6),
  y = list(TRUE, 1, "a"),
  z = list(sum, mean, sd)
)

#
df %>%
  rowwise() %>%
  summarise(
    x_length = length(x),
    y_type = typeof(y),
    z_call = z(1:5)
  )

```



```

# #####
#
df <- tribble(
  ~id, ~ n, ~ min, ~ max,
  1,   3,   0,   1,
  2,   2,  10,  100,
  3,   2, 100, 1000,
)

#           rowwise
df %>%
  rowwise(id) %>%
  mutate(data = list(runif(n, min, max)))

df %>%
  rowwise(id) %>%
  summarise(x = runif(n, min, max))

# #####
#           nest_by
by_cyl <- mtcars %>% nest_by(cyl)
by_cyl

#
#           mutate
by_cyl <- by_cyl %>% mutate(model = list(lm(mpg ~ wt, data = data)))
by_cyl

#           summarise
#
by_cyl %>% summarise(broom::glance(model))
by_cyl %>% summarise(broom::tidy(model))

```

## 3.4

```

#
set.seed(400)
year <- 2000:2005
sales <- sapply(
  month.abb,
  FUN = function(x) round(runif(n = 6, min = 100, max = 400), 0)
)

```

```

)

sales <- as.data.frame(sales, row.names = year)
sales$year <- year
sales
#>      Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec year
#> 2000 145 174 342 185 117 313 314 102 382 220 226 297 2000
#> 2001 156 251 286 280 179 176 317 323 247 194 233 263 2001
#> 2002 319 182 329 155 240 177 146 244 370 300 197 187 2002
#> 2003 209 187 238 296 393 234 366 314 198 213 206 234 2003
#> 2004 379 126 263 261 136 201 352 351 362 203 304 183 2004
#> 2005 221 275 374 318 127 376 257 193 340 190 225 273 2005

      ,
      ,

      .

      ,      4      :

      • winter_avg_sales -      ;
      • spring_avg_sales -      ;
      • summer_avg_sales -      ;
      • autumn_avg_sales -      ;

      ,

      :

# A tibble: 6 x 5
# Rowwise:
  year winter_avg_sales spring_avg_sales summer_avg_sales autumn_avg_sales
<int>      <dbl>      <dbl>      <dbl>      <dbl>
1  2000          322          226          145          227
2  2001          174          192.          179.          295.
3  2002          106          352.          215.          258.
4  2003          105          260.          334.          206.
5  2004          167          192.          239          254.
6  2005          210          191.          271.          235

```

## Chapter 4

# summarise()

### 4.1

`summarise()`.

“dplyr 1.0.0: new summarise() features”.

### 4.2

### 4.3

```
#devtools::install_github("tidyverse/dplyr")
library(dplyr)

#
##      summarise
###
###

# #####
#
# #####
df <- tibble(
  grp = rep(1:2, each = 5),
  x = c(rnorm(5, -0.25, 1), rnorm(5, 0, 1.5)),
  y = c(rnorm(5, 0.25, 1), rnorm(5, 0, 0.5)),
)
```

```

df

#
#
df %>%
  group_by(grp) %>%
  summarise(rng = range(x))

##      range      2
range(df$x)
##      summarise      ,
##

#      ,
df %>%
  group_by(grp) %>%
  summarise(tibble(min = min(x), mean = mean(x)))

# #####
#
# #####
df %>%
  group_by(grp) %>%
  summarise(x = quantile(x, c(0.25, 0.5, 0.75)), q = c(0.25, 0.5, 0.75))

#
quibble <- function(x, q = c(0.25, 0.5, 0.75)) {
  tibble(x = quantile(x, q), q = q)
}
#      summarise
df %>%
  group_by(grp) %>%
  summarise(quibble(x, c(0.25, 0.5, 0.75)))

#
#
quibble2 <- function(x, q = c(0.25, 0.5, 0.75)) {
  tibble("{ x }" := quantile(x, q), "{ x }_q" := q)
}

df %>%
  group_by(grp) %>%
  summarise(quibble2(x, c(0.25, 0.5, 0.75)))

```

```

#                                     summarise
#
#
out <- df %>%
  group_by(grp) %>%
  summarise(quantile = quibble2(y, c(0.25, 0.75)))

str(out)

#
out$y

#
#
out$quantile$y_q

# summarise + rowwise
#                                     purrrr   apply
tibble(path = dir(pattern = "\\*.csv$")) %>%
  rowwise(path) %>%
  summarise(readr::read_csv(path))

# #####
#
# #####
#
df %>%
  group_by(grp) %>%
  summarise(y = list(quibble(y, c(0.25, 0.75)))) %>%
  tidyr::unnest(y)

df %>%
  group_by(grp) %>%
  do(quibble(.$y, c(0.25, 0.75)))

```

## 4.4

```

(
  rnorm()).
  :

```

```
library(dplyr)
```

```
params <- tribble(
  ~sim, ~n, ~mean, ~sd,
  1, 4, 1, 5,
  2, 7, 2, 10,
  3, 10, -1, 25
)
```

```

      val                                     sim
      400 (set.seed(400)).                    :

# A tibble: 21 x 2
# Groups:   sim [3]
   sim    val
  <dbl> <dbl>
1     1 -1.75
2     1 -3.70
3     1  2.27
4     1 13.2
5     2  9.87
6     2 -5.14
7     2 -8.09
8     2 -3.57
9     2  6.77
10    2 29.4
# ... with 11 more rows
```

## Chapter 5

,  
**rows\_\***()

### 5.1

SQL INSERT, UPDATE DELETE,  
dplyr 1.0.0  
R.  
: - rows\_insert() - rows\_update() -  
rows\_upsert() - rows\_patch() - rows\_delete()  
summarise(), .groups,  
.  
“dplyr 1.0.0: last minute additions”.

### 5.2

### 5.3

```
#devtools::install_github("tidyverse/dplyr")  
library(dplyr)  
  
# summarise + .groups  
starwars %>%  
  group_by(homeworld, species) %>%
```

```

summarise(n = n())

##      .groups
### .groups = "drop_last"
### .groups = "drop"
### .groups = "keep"
### .groups = "rowwise"                                rowwise()

# rows_*()
## rows_update(x, y)                                     x      y
## rows_patch(x, y)                                     rows_update()      NA
## rows_insert(x, y)                                    x      y
## rows_upsert(x, y)                                    x      y
## rows_delete(x, y)                                    x      y.

#
df <- tibble(a = 1:3, b = letters[c(1:2, NA)], c = 0.5 + 0:2)
df

new <- tibble(a = c(4, 5), b = c("d", "e"), c = c(3.5, 4.5))
new

#
##
df %>% rows_insert(new)

## row_insert
df %>% rows_insert(tibble(a = 3, b = "c"))

##
df %>% rows_update(tibble(a = 3, b = "c"))                row_update

## rows_update
df %>% rows_update(tibble(a = 4, b = "d"))

## rows_patch
df %>%
  rows_patch(tibble(a = 2:3, b = "B"))

## rows_upsert
## rows_upsert
df %>%
  rows_upsert(tibble(a = 3, b = "c")) %>%
  rows_upsert(tibble(a = 4, b = "d"))

```



```

# #####
#
set.seed(555)

#
managers <- c("Paul", "Alex", "Tim", "Bill", "John")
#
products <- tibble(name = paste0("product_", LETTERS),
                    price = round(runif(n = length(LETTERS), 100, 400), 0))

#
prod_list <- function(prod_list, size_min, size_max) {

  prod <- tibble(product = sample(prod_list,
                                  size = round(runif(1, size_min, size_max), 0) ,
                                  replace = F))

  return(prod)
}

#
sales <- tibble(id = 1:200,
                manager_id = sample(managers, size = 200, replace = T),
                refund = FALSE,
                refund_sum = 0)

#
sale_proucts <-
  sales %>%
  rowwise(id) %>%
  summarise(prod_list(products$name, 1, 6)) %>%
  left_join(products, by = c("product" = "name"))

#
sales <- left_join(sales, sale_proucts, by = "id")

#
refund <- sample_n(sales, 25) %>%
  mutate( refund = TRUE,
          refund_sum = price * 0.9) %>%
  select(-price, -manager_id)

#
sales %>%
  rows_update(refund)

```

```
#      by
result <-
  sales %>%
    rows_update(refund, by = c("id", "product"))
```

## 5.4

6

.

:

```
library(dplyr)

#
salary <- tibble(
  employee_id = 1:5,
  rate        = c(1000, 1200, 700, 1500, 2000),
  bonus       = rep(0, 5),
  penalty     = rep(0, 5)
)

#
bonus <- tibble(
  employee_id = c(3, 5),
  bonus       = c(100, 500)
)

#
penalty <- tibble(
  employee_id = c(1, 4, 5),
  penalty     = c(150, 320, 80)
)

#
new <- tibble(
  employee_id = 6,
  rate        = 500,
  bonus       = 0,
  penalty     = 0
)

#
time_rate <- tibble(
```

```
employee_id = 1:6,  
time_rate = c(1, 1, 1, 0.8, 1, 0.5)  
)
```

5 :

- salary - , ;
- bonus - , ;
- penalty - , ;
- new - 6 , ;
- time\_rate - .

total = rate \* time\_rate +

bonus - penalty.

:

# A tibble: 6 x 6

	employee_id	rate	bonus	penalty	time_rate	total
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	1	1000	0	150	1	850
2	2	1200	0	0	1	1200
3	3	700	100	0	1	800
4	4	1500	0	320	0.8	880
5	5	2000	500	80	1	2420
6	6	500	0	0	0.5	250



1. , Width.

```
library(dplyr)
#>
#> Attaching package: 'dplyr'
#> The following objects are masked from 'package:stats':
#>
#>   filter, lag
#> The following objects are masked from 'package:base':
#>
#>   intersect, setdiff, setequal, union

select(iris, ends_with('Width')) %>%
  tibble()
#> # A tibble: 150 x 2
#>   Sepal.Width Petal.Width
#>   <dbl>         <dbl>
#> 1     3.5         0.2
#> 2     3           0.2
#> 3     3.2         0.2
#> 4     3.1         0.2
#> 5     3.6         0.2
#> 6     3.9         0.4
#> 7     3.4         0.3
#> 8     3.4         0.2
#> 9     2.9         0.2
#> 10    3.1         0.1
#> # ... with 140 more rows
#> # i Use `print(n = ...)` to see more rows
```

2. relocate() .

```
relocate(iris, where(is.factor)) %>%
  tibble()
#> # A tibble: 150 x 5
#>   Species Sepal.Length Sepal.Width Petal.Length Petal.Width
#>   <fct>      <dbl>      <dbl>      <dbl>      <dbl>
#> 1 setosa      5.1        3.5        1.4        0.2
#> 2 setosa      4.9         3         1.4        0.2
#> 3 setosa      4.7        3.2        1.3        0.2
#> 4 setosa      4.6        3.1        1.5        0.2
#> 5 setosa      5         3.6        1.4        0.2
#> 6 setosa      5.4        3.9        1.7        0.4
#> 7 setosa      4.6        3.4        1.4        0.3
#> 8 setosa      5         3.4        1.5        0.2
#> 9 setosa      4.4        2.9        1.4        0.2
#> 10 setosa     4.9        3.1        1.5        0.1
#> # ... with 140 more rows
#> # i Use `print(n = ...)` to see more rows
```

### 3. rename\_with()

```
renamer <- function(x) gsub('\\.', '\\_', x) %>% tolower()
rename_with(iris, renamer) %>%
  tibble()
#> # A tibble: 150 x 5
#>   sepal_length sepal_width petal_length petal_width species
#>   <dbl>      <dbl>      <dbl>      <dbl> <fct>
#> 1      5.1        3.5        1.4        0.2 setosa
#> 2      4.9         3         1.4        0.2 setosa
#> 3      4.7        3.2        1.3        0.2 setosa
#> 4      4.6        3.1        1.5        0.2 setosa
#> 5      5         3.6        1.4        0.2 setosa
#> 6      5.4        3.9        1.7        0.4 setosa
#> 7      4.6        3.4        1.4        0.3 setosa
#> 8      5         3.4        1.5        0.2 setosa
#> 9      4.4        2.9        1.4        0.2 setosa
#> 10     4.9        3.1        1.5        0.1 setosa
#> # ... with 140 more rows
#> # i Use `print(n = ...)` to see more rows
```

1. `across()`, `Length`

```
library(dplyr)

mutate(iris, across(ends_with('Length'), ~ . / mean(.))) %>%
  tibble()
#> # A tibble: 150 x 5
#>   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
#>   <dbl>         <dbl>         <dbl>         <dbl> <fct>
#> 1      0.873         3.5         0.373         0.2 setosa
#> 2      0.839         3         0.373         0.2 setosa
#> 3      0.804         3.2         0.346         0.2 setosa
#> 4      0.787         3.1         0.399         0.2 setosa
#> 5      0.856         3.6         0.373         0.2 setosa
#> 6      0.924         3.9         0.452         0.4 setosa
#> 7      0.787         3.4         0.373         0.3 setosa
#> 8      0.856         3.4         0.399         0.2 setosa
#> 9      0.753         2.9         0.373         0.2 setosa
#> 10     0.839         3.1         0.399         0.1 setosa
#> # ... with 140 more rows
#> # i Use `print(n = ...)` to see more rows
```

2. `Species`, `Sepal`

```
group_by(iris, Species) %>%
  summarise(across(starts_with('Sepal'), mean))
#> # A tibble: 3 x 3
#>   Species   Sepal.Length Sepal.Width
#>   <fct>         <dbl>         <dbl>
#> 1 setosa         5.01         3.43
#> 2 versicolor     5.94         2.77
#> 3 virginica      6.59         2.97
```

e

1. `4` :

- `winter_avg_sales -` ;

- `spring_avg_sales` - ;
- `summer_avg_sales` - ;
- `autumn_avg_sales` - ;

```
library(dplyr)

rowwise(sales) %>%
  mutate(
    winter_avg_sales = mean(Dec, Jan, Feb),
    spring_avg_sales = mean(c_across(Mar:May)),
    summer_avg_sales = mean(c_across(Jun:Aug)),
    autumn_avg_sales = mean(c_across(Sep:Nov))
  ) %>%
  select(year, matches('avg'))
#> # A tibble: 6 x 5
#> # Rowwise:
#>   year winter_avg_sales spring_avg_sales summer_a~1 autum~2
#>   <int>          <dbl>          <dbl>      <dbl>    <dbl>
#> 1  2000             297             215.      243     276
#> 2  2001             263             248.      272     225.
#> 3  2002             187             241.      189     289
#> 4  2003             234             309      305.     206.
#> 5  2004             183             220      301.     290.
#> 6  2005             273             273      275.     252.
#> # ... with abbreviated variable names 1: summer_avg_sales,
#> #   2: autumn_avg_sales
```

e

1. `val` `sim`,  
400 (`set.seed(400)`).  
:  
:

```
library(dplyr)
set.seed(400)

params %>%
```



```

rowwise(sim) %>%
  summarise(val = rnorm(n, mean, sd))
#> `summarise()` has grouped output by 'sim'. You can override
#> using the ``.groups` argument.
#> # A tibble: 21 x 2
#> # Groups:   sim [3]
#>       sim    val
#>   <dbl> <dbl>
#> 1     1 -4.18
#> 2     1  4.08
#> 3     1  8.36
#> 4     1 -2.41
#> 5     2 -4.02
#> 6     2 -11.5
#> 7     2 10.6
#> 8     2  9.20
#> 9     2  3.08
#> 10    2 -3.75
#> # ... with 11 more rows
#> # Use `print(n = ...)` to see more rows

```

e

```

1.                                     total = rate *
   time_rate + bonus - penalty.

:

```

```

library(dplyr)

rows_update(salary, bonus, by = 'employee_id') %>%
  rows_update(penalty, by = 'employee_id') %>%
  rows_insert(new, by = 'employee_id') %>%
  left_join(time_rate, by = 'employee_id') %>%
  mutate(total = rate * time_rate + bonus - penalty)
#> # A tibble: 6 x 6
#>   employee_id rate bonus penalty time_rate total
#>       <int> <dbl> <dbl>   <dbl>   <dbl> <dbl>
#> 1         1  1000     0    150     1    850
#> 2         2  1200     0     0     1   1200
#> 3         3   700   100     0     1    800
#> 4         4  1500     0    320   0.8   880

```

#> 5	5	2000	500	80	1	2420
#> 6	6	500	0	0	0.5	250

dplyr 1.0.0 ,  
.  
Telegram YouTube .