R

Yufei Zhong

2021-04-29

Contents

Chapter 1

R R shiny R , R

:598253220@qq.com

:

```
#
sessionInfo()
#> R version 4.0.5 (2021-03-31)
#> Platform: x86_64-w64-mingw32/x64 (64-bit)
#> Running under: Windows 10 x64 (build 19041)
#>
#> Matrix products: default
#>
#> locale:
#> [1] LC_COLLATE=Chinese (Simplified)_China.936
#> [2] LC_CTYPE=Chinese (Simplified)_China.936
#> [3] LC_MONETARY=Chinese (Simplified)_China.936
#> [4] LC_NUMERIC=C
#> [5] LC_TIME=Chinese (Simplified)_China.936
#>
#> attached base packages:
#> [1] stats     graphics  grDevices utils     datasets  methods   base
#>
#> loaded via a namespace (and not attached):
#>  [1] compiler_4.0.5    magrittr_2.0.1    bookdown_0.21     htmltools_0.5.1.1
#>  [5] tools_4.0.5       rstudioapi_0.13   yaml_2.2.1        stringi_1.5.3
#>  [9] rmarkdown_2.7     knitr_1.32        stringr_1.4.0     digest_0.6.27
#> [13] xfun_0.22         rlang_0.4.10      evaluate_0.14
```

Chapter 2

data.table

data.table R python julia

data.table tidyverse “ ” “ ” “ ” “ ” R

Python ing python

data.table :

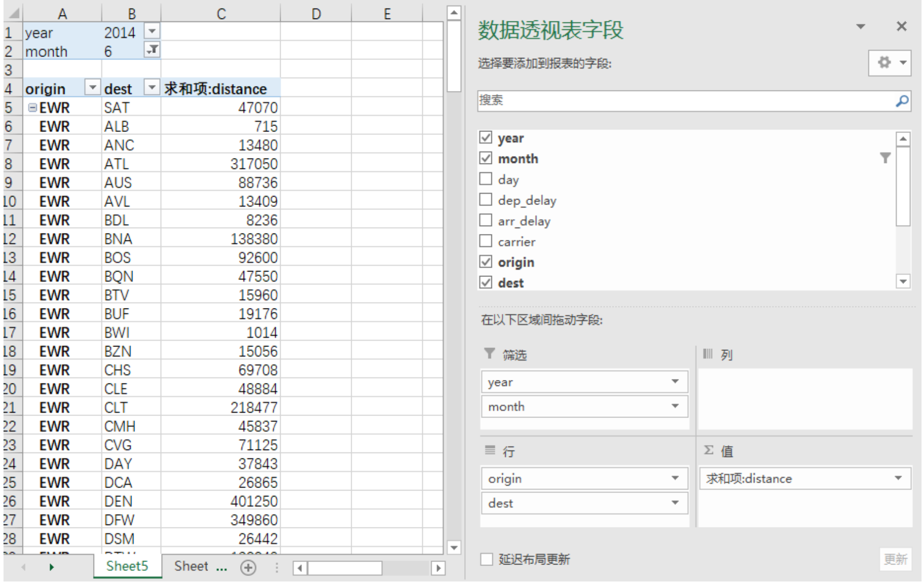
https://cran.r-project.org/web/packages/data.table/vignettes/datatable-intro.html

2.1

2.1.1

```
DT[i, j, by]
##   R:                 i                 j         by
## SQL:  where | order by   select | update  group by
```

data.table ,i ,j ,by . Excel ,i ,by ,j , :



	A	B	C
1	year	2014	
2	month	6	
3			
4	origin	dest	求和项:distance
5	⊟EWR	SAT	47070
6	EWR	ALB	715
7	EWR	ANC	13480
8	EWR	ATL	317050
9	EWR	AUS	88736
10	EWR	AVL	13409
11	EWR	BDL	8236
12	EWR	BNA	138380
13	EWR	BOS	92600
14	EWR	BQN	47550
15	EWR	BTV	15960
16	EWR	BUF	19176
17	EWR	BWI	1014
18	EWR	BZN	15056
19	EWR	CHS	69708
20	EWR	CLE	48884
21	EWR	CLT	218477
22	EWR	CMH	45837
23	EWR	CVG	71125
24	EWR	DAY	37843
25	EWR	DCA	26865
26	EWR	DEN	401250
27	EWR	DFW	349860
28	EWR	DSM	26442

Sheet5 Sheet ...

数据透视表字段

选择要添加到报表的字段:

搜索

☑ year
☑ month
□ day
□ dep_delay
□ arr_delay
□ carrier
☑ origin
☑ dest

在以下区域间拖动字段:

筛选: year, month

列

行: origin, dest

Σ 值: 求和项:distance

□ 延迟布局更新 更新

1.

2014 6 , .

```
library(data.table)
flights <- fread("./data/flights.csv")
flights[year==2014 & month==6,.( distance=sum(distance)),by=.(origin,dest)]
#>      origin dest distance
#>   1:    JFK  LAX  2663100
#>   2:    JFK  DFW    82069
#>   3:    JFK  LAS   795792
#>   4:    JFK  SFO  1967946
#>   5:    JFK  SAN   349778
#>  ---
#> 191:    EWR  ANC    13480
#> 192:    EWR  BZN    15056
#> 193:    LGA  TVC     7205
#> 194:    LGA  BZN     3788
#> 195:    JFK  HYA      980
```

2.

i year==2014 month==6 ;

j distance=sum(distance) .() list()

by .(origin,dest), .() , Excel

.() “ ” data.table Python

2.2 i j by

data.table data.table fread i,j,by

2.2.1

data.table fread , ,csv,Excel .fread CSV , .
data.table demo.

```
library(data.table)
input <- if (file.exists("./data/flights.csv")) {
   "./data/flights.csv" #
} else {
  "https://raw.githubusercontent.com/Rdatatable/data.table/master/vignettes/flights.csv" #
}
flights <- fread(input) #             encoding ,  encoding='UTF-8'

head(flights)
#>    year month day dep_delay arr_delay carrier origin dest air_time distance
#> 1: 2014     1   1        14        13      AA    JFK  LAX      359     2475
#> 2: 2014     1   1        -3        13      AA    JFK  LAX      363     2475
#> 3: 2014     1   1         2         9      AA    JFK  LAX      351     2475
#> 4: 2014     1   1        -8       -26      AA    LGA  PBI      157     1035
#> 5: 2014     1   1         2         1      AA    JFK  LAX      350     2475
#> 6: 2014     1   1         4         0      AA    EWR  LAX      339     2454
#>    hour
#> 1:    9
#> 2:   11
#> 3:   19
#> 4:    7
#> 5:   13
#> 6:   18
```

, , (http://www.zhongyufei.com/datatable/data/flights.csv)

```
flights <- fread("http://www.zhongyufei.com/datatable/data/flights.csv")
```

2014 , 3 (:JFK LGA , EWR)
( )

2.2.2

Excel R >
< == != >= <= & |

```
#
filghts[year == 2014] # year==2014
#     &
flights[ year == 2014 & month == 6]
# |
flights[ month == 5 | month == 6]
# %in%  sql in
flights[month %in% c(1,3,5,7,9)]
# %between%  sql between and
flights[month %between% c(1,7)]
```

2.2.3

sql select .() list() data.table

```
#   . .()
flights[,.(year,month,day,dep_delay,carrier,origin)]
#>         year month day dep_delay carrier origin
#>      1: 2014     1   1        14      AA    JFK
#>      2: 2014     1   1        -3      AA    JFK
#>      3: 2014     1   1         2      AA    JFK
#>      4: 2014     1   1        -8      AA    LGA
#>      5: 2014     1   1         2      AA    JFK
#>     ---
#> 253312: 2014    10  31         1      UA    LGA
#> 253313: 2014    10  31        -5      UA    EWR
#> 253314: 2014    10  31        -8      MQ    LGA
#> 253315: 2014    10  31        -4      MQ    LGA
#> 253316: 2014    10  31        -5      MQ    LGA
# flights[,list(year,month,day,dep_delay,carrier,origin)]  same above

# not run
# flights[,1:3]

# not run
# flights[,c('year','month','day')]
```

setcolorder

```
# not run
# setcolorder(x = flights,neworder = c( "month","day","dep_delay" ,"arr_delay","carrie
#           ,    ,    flights
```

2.2.4

i , j ;data.table j i data.table i,j

```
dt <- flights[ year == 2014 & month == 6 & day >=15,.(year,month,day,dep_delay,carrier,origin)]
head(dt)
#>    year month day dep_delay carrier origin
#> 1: 2014     6  15        -4      AA    JFK
#> 2: 2014     6  15        -8      AA    JFK
#> 3: 2014     6  15       -12      AA    JFK
#> 4: 2014     6  15        -4      AA    LGA
#> 5: 2014     6  15        -3      AA    JFK
#> 6: 2014     6  15         5      AA    JFK
```

2.2.5 j

Excel

```
flights[year==2014 & month==6,.(  distance=sum(distance),   =mean(distance)),by=.(origin,dest)]
```

i j by j R

```
myfun <- function(x){
    x^2/2
}
flights[year==2014 & month==6,.(myfun(distance)),by=.(origin,dest)]
#>        origin dest      V1
#>     1:    JFK  LAX 3062813
#>     2:    JFK  LAX 3062813
#>     3:    JFK  LAX 3062813
#>     4:    JFK  LAX 3062813
#>     5:    JFK  LAX 3062813
#>    ---
#> 26484:    JFK  HYA   19208
#> 26485:    JFK  HYA   19208
#> 26486:    JFK  HYA   19208
#> 26487:    JFK  HYA   19208
#> 26488:    JFK  HYA   19208
```

2.2.6 by

1.

```
flights[,.(sum(distance)),by=.(month)]
#>     month       V1
#>  1:     1 25112563
#>  2:     2 22840391
#>  3:     3 28716598
#>  4:     4 27816797
#>  5:     5 28030020
#>  6:     6 29093557
#>  7:     7 30059175
#>  8:     8 30322047
#>  9:     9 27615097
#> 10:    10 28900834
```

2.

```
dt <- flights[,.(sum(distance)),by=.(carrier,origin)]
head(dt)
#>    carrier origin       V1
#> 1:      AA    JFK 20492213
#> 2:      AA    LGA 12365282
#> 3:      AA    EWR  3550217
#> 4:      AS    EWR  1378748
#> 5:      B6    JFK 38117662
#> 6:      B6    EWR  4508574
#
dt <- flights[,.(sum(distance)),by=.(newcol1 = carrier,newcol2 = origin)]
head(dt)
#>    newcol1 newcol2       V1
#> 1:      AA     JFK 20492213
#> 2:      AA     LGA 12365282
#> 3:      AA     EWR  3550217
#> 4:      AS     EWR  1378748
#> 5:      B6     JFK 38117662
#> 6:      B6     EWR  4508574
```

3. 6

6

```
dt <- flights[,.(sum(distance)),by=.(month>6)] #by
head(dt)
#>    month        V1
#> 1: FALSE 161609926
#> 2:  TRUE 116897153
```

2.3

2.3.1

data.table :=

-

: addcol [] ,

```
#data.table()  data.table
dt <- data.table(col1=1:10,col2=letters[1:10],col3=LETTERS[1:10],col4=1:10)
#    :=
dt[,addcol:=rep(' ',10)][] #  []       ,
#>      col1 col2 col3 col4 addcol
#>  1:     1    a    A    1       
#>  2:     2    b    B    2       
#>  3:     3    c    C    3       
#>  4:     4    d    D    4       
#>  5:     5    e    E    5       
#>  6:     6    f    F    6       
#>  7:     7    g    G    7       
#>  8:     8    h    H    8       
#>  9:     9    i    I    9       
#> 10:    10    j    J   10       
#dt[,addcol:=rep(' ',10)]      , []
#
dt[,`:=`(newcol1=rep('newcol1',10),newcol2=rep('newcol2',10))][]
#>      col1 col2 col3 col4 addcol newcol1 newcol2
#>  1:     1    a    A    1        newcol1 newcol2
#>  2:     2    b    B    2        newcol1 newcol2
#>  3:     3    c    C    3        newcol1 newcol2
#>  4:     4    d    D    4        newcol1 newcol2
#>  5:     5    e    E    5        newcol1 newcol2
#>  6:     6    f    F    6        newcol1 newcol2
#>  7:     7    g    G    7        newcol1 newcol2
#>  8:     8    h    H    8        newcol1 newcol2
#>  9:     9    i    I    9        newcol1 newcol2
#> 10:    10    j    J   10        newcol1 newcol2
```

-

  NULL

```
#
dt[,col1:=NULL][]
#>     col2 col3 col4 addcol newcol1 newcol2
#>  1:    a    A    1     newcol1 newcol2
#>  2:    b    B    2     newcol1 newcol2
#>  3:    c    C    3     newcol1 newcol2
#>  4:    d    D    4     newcol1 newcol2
#>  5:    e    E    5     newcol1 newcol2
#>  6:    f    F    6     newcol1 newcol2
#>  7:    g    G    7     newcol1 newcol2
#>  8:    h    H    8     newcol1 newcol2
#>  9:    i    I    9     newcol1 newcol2
#> 10:    j    J   10     newcol1 newcol2
#
dt[,c('newcol1','newcol2'):=NULL][]
#>     col2 col3 col4 addcol
#>  1:    a    A    1
#>  2:    b    B    2
#>  3:    c    C    3
#>  4:    d    D    4
#>  5:    e    E    5
#>  6:    f    F    6
#>  7:    g    G    7
#>  8:    h    H    8
#>  9:    i    I    9
#> 10:    j    J   10
```

-

```
#
dt[,col1:=11:20][]
#>     col2 col3 col4 addcol col1
#>  1:    a    A    1        11
#>  2:    b    B    2        12
#>  3:    c    C    3        13
#>  4:    d    D    4        14
#>  5:    e    E    5        15
#>  6:    f    F    6        16
#>  7:    g    G    7        17
#>  8:    h    H    8        18
#>  9:    i    I    9        19
#> 10:    j    J   10        20
```

```
# not run
#
dt[,newcol:=col1/col4]
```

2.3.2

`setorder` `setorderv` `data.table` base R `order` Note that queries like x[order(.)] are optimised internally to use data.table's fast order x[order(.)] data.table

```
set.seed(45L)
DT = data.table(A=sample(3, 10, TRUE),
         B=sample(letters[1:3], 10, TRUE), C=sample(10))

setorder(DT, A, -B) # DT A B  A  ,-B

#          setorderv
setorderv(DT, c("A", "B"), c(1, -1))
```

2.3.3

data.table

-

%in% sql in

```
#   %in%
flights[ hour %in% seq(1,24,2) ]
```

-

%chin% %in%

```
#
flights[ origin %chin% c('JFK','LGA')]
# not run   %chin%
#flights[ origin %in% c('JFK','LGA')]
```

- between

```
#between
#between(x, lower, upper, incbounds=TRUE, NAbounds=TRUE, check=FALSE)
X <-  data.table(a=1:5, b=6:10, c=c(5:1))
X[b %between% c(7,9)]
#>    a b c
#> 1: 2 7 4
#> 2: 3 8 3
#> 3: 4 9 2
X[between(b, 7, 9)] #
#>    a b c
#> 1: 2 7 4
#> 2: 3 8 3
#> 3: 4 9 2
X[c %between% list(a,b)] #
#>    a b c
#> 1: 1 6 5
#> 2: 2 7 4
#> 3: 3 8 3
```

- like

%like%   SQL like

```
# %like%   SQL  like
DT = data.table(Name=c("Mary","George","Martha"), Salary=c(2,3,4))
DT[Name %like% "^Mar"]
#>      Name Salary
#> 1:   Mary      2
#> 2: Martha      4
```

2.3.4

.SD,.BY,.N,.I,.NGRP .GRP,.SDcols ,   j   ,.N   i   .

```
DT = data.table(x=rep(c("b","a","c"),each=3), v=c(1,1,1,2,2,1,1,2,2), y=c(1,3,6), a=1:
DT
#>    x v y a b
#> 1: b 1 1 1 9
#> 2: b 1 3 2 8
#> 3: b 1 6 3 7
#> 4: a 2 1 4 6
#> 5: a 2 3 5 5
#> 6: a 1 6 6 4
#> 7: c 1 1 7 3
#> 8: c 2 3 8 2
#> 9: c 2 6 9 1
X = data.table(x=c("c","b"), v=8:7, foo=c(4,2))
X
#>    x v foo
#> 1: c 8   4
#> 2: b 7   2

#  i
DT[.N] # DT  ,.N
#>    x v y a b
#> 1: c 2 6 9 1
DT[,.N] #DT
#> [1] 9
DT[, .N, by=x]  #
#>    x N
#> 1: b 3
#> 2: a 3
#> 3: c 3
DT[, .SD, .SDcols=x:y]  #  x  y
#>    x v y
#> 1: b 1 1
#> 2: b 1 3
#> 3: b 1 6
#> 4: a 2 1
#> 5: a 2 3
#> 6: a 1 6
#> 7: c 1 1
#> 8: c 2 3
#> 9: c 2 6
#DT[, .SD, .SDcols=c("x","y")]

DT[, .SD[1]] #
#>    x v y a b
#> 1: b 1 1 1 9
DT[, .SD[1], by=x] # x
#>    x v y a b
#> 1: b 1 1 1 9
#> 2: a 2 1 4 6
#> 3: c 1 1 7 3
DT[, c(.N, lapply(.SD, sum)), by=x] # x
#>    x N v  y  a  b
#> 1: b 3 3 10  6 24
#> 2: a 3 5 10 15 15
#> 3: c 3 5 10 24  6
```

2.4

2.4.1 frank

frank frankv

```
frank(x, ..., na.last=TRUE, ties.method=c("average",
  "first", "last", "random", "max", "min", "dense"))

frankv(x, cols=seq_along(x), order=1L, na.last=TRUE,
      ties.method=c("average", "first", "random",
        "max", "min", "dense"))
```

, :

```
# on vectors
x = c(4, 1, 4, NA, 1, NA, 4)
# NAs are considered identical (unlike base R)
# default is average
frankv(x) # na.last=TRUE
#> [1] 4.0 1.5 4.0 6.5 1.5 6.5 4.0
frankv(x, na.last=FALSE)
#> [1] 6.0 3.5 6.0 1.5 3.5 1.5 6.0

# on data.table
DT = data.table(x, y=c(1, 1, 1, 0, NA, 0, 2))
frankv(DT, cols="x") # same as frankv(x) from before
#> [1] 4.0 1.5 4.0 6.5 1.5 6.5 4.0
frankv(DT, cols="x", na.last="keep")
#> [1] 4.0 1.5 4.0  NA 1.5  NA 4.0
frankv(DT, cols="x", ties.method="dense", na.last=NA)
#> [1] 2 1 2 1 2
frank(DT, x, ties.method="dense", na.last=NA) # equivalent of above using frank
#> [1] 2 1 2 1 2
```

- frankv ,NA , base R .

```
x <-  c(4, 1, 4, NA, 1, NA, 4)
frankv(x)
#> [1] 4.0 1.5 4.0 6.5 1.5 6.5 4.0
```

```
rank(x)
#> [1] 4.0 1.5 4.0 6.0 1.5 7.0 4.0
```

-

order 1 -1. 1

```
frankv(x,order = 1L)
#> [1] 4.0 1.5 4.0 6.5 1.5 6.5 4.0
frankv(x,order = -1L)
#> [1] 2.0 4.5 2.0 6.5 4.5 6.5 2.0
```

-

average, dense,random,first,last,max,min dense random

```
x <- c(1,1,1,2,3)
frankv(x)  #     ,    2
frankv(x,ties.method = 'min')  #      ,
frankv(x,ties.method = 'max')  #      ,
frankv(x,ties.method = 'first') #
frankv(x,ties.method = 'dense')
frankv(x,ties.method = 'random')
```

- NA

NA ,NAs base R

na.last TRUE FALSE, NA “keep”, NA.

```
frankv(c(NA,NA,1,2,3), na.last = TRUE,ties.method = 'first')
#> [1] 4 5 1 2 3
frankv(c(NA,NA,1,2,3), na.last = FALSE,ties.method = 'first')
#> [1] 1 2 3 4 5
frankv(c(NA,NA,1,2,3), na.last = NA,ties.method = 'first')
#> [1] 1 2 3
frankv(c(NA,NA,1,2,3), na.last = 'keep',ties.method = 'first')
#> [1] NA NA  1  2  3
```

2.4.2

- fifelse

fifelse() `dplyr::if_else()` , base::ifelse()

```
x <-  c(1:4, 3:2, 1:4,5)
fifelse(x > 2L, x, x - 1L)
#>  [1] 0 1 3 4 3 1 0 1 3 4 5

fifelse(x > 2L,fifelse(x >= 4L,x + 1L,x),x-1L)
#>  [1] 0 1 3 5 3 1 0 1 3 5 6
```

- fcase

sql case when dplyr `case_when()` fifelse

```
x = 1:10
fcase(
    x < 5L, 1L,
    x > 5L, 3L
)
#>  [1]  1  1  1  1 NA  3  3  3  3  3

# not run
fifelse(x > 5,fifelse(x >8,2,1),0)
#>  [1] 0 0 0 0 0 1 1 1 2 2
fcase(
  x > 8,2,
  x > 5,1,
  default = 0
)
#>  [1] 0 0 0 0 0 1 1 1 2 2
```

2.4.3

base R union(),intersect(),setdiff() setequal() .all , SQL ,data.table .

```
fintersect(x, y, all = FALSE)
fsetdiff(x, y, all = FALSE)
funion(x, y, all = FALSE)
fsetequal(x, y, all = TRUE)

x <-  data.table(c(1,2,2,2,3,4,4))
x2 <-  data.table(c(1,2,3,4)) # same set of rows as x
y <-  data.table(c(2,3,4,4,4,5))
```

```
fintersect(x, y)            # intersect
fintersect(x, y, all=TRUE)  # intersect all

fsetdiff(x, y)              # except
fsetdiff(x, y, all=TRUE)    # except all
funion(x, y)                # union
funion(x, y, all=TRUE)      # union all
fsetequal(x, x2, all=FALSE) # setequal
fsetequal(x, x2)            # setequal all
```

2.4.4

, , left_join right_join,inner_join .

?merge() ,data.table base R merge , data.table data.table::merge().

```
?merge()
merge(x, y, by = NULL, by.x = NULL, by.y = NULL, all = FALSE,
all.x = all, all.y = all, sort = TRUE, suffixes = c(".x", ".y"), no.dups = TRUE,
allow.cartesian=getOption("datatable.allow.cartesian"),  # default FALSE
...)
```

x.y , , by=c(",") , ,by.x=,by.y= ,all,all.x,all.y ,sort , ,allow.cartesian= , , TUR

2.4.5

`dcast melt` Excel

- dcast

fun.aggregate value.var formula x+y~z x,y z

```
dcast(data, formula, fun.aggregate = NULL, sep = "_",
    ..., margins = NULL, subset = NULL, fill = NULL,
    drop = TRUE, value.var = guess(data),
    verbose = getOption("datatable.verbose"))
```

```
dt <- data.table(  =rep(c(' ',' ',' ',' '),1000),
               =rep(c('  ','  ','  ','  '),1000),
                =sample(100:200,4000,replace = TRUE))
```

```
dcast(dt,  ~ ,value.var = "  ",fun.aggregate = sum)
#>
#> 1:      149135       0       0       0
#> 2:           0       0       0  150585
#> 3:           0       0  149451       0
#> 4:           0  150649       0       0
```

V1.9.6

fun fun.aggregate

```
dt <-  data.table(x=sample(5,20,TRUE), y=sample(2,20,TRUE),
                z=sample(letters[1:2], 20,TRUE), d1 = runif(20), d2=1L)
dcast(dt, x + y ~ z, fun=list(sum,mean), value.var=c("d1","d2"))
#>    x y d1_sum_a d1_sum_b d2_sum_a d2_sum_b d1_mean_a d1_mean_b d2_mean_a
#> 1: 1 1    0.000   0.3141        0        1       NaN    0.3141       NaN
#> 2: 1 2    0.675   0.7524        1        1     0.675    0.7524         1
#> 3: 2 1    0.722   1.9725        1        3     0.722    0.6575         1
#> 4: 2 2    1.062   0.0657        2        1     0.531    0.0657         1
#> 5: 3 2    0.329   0.0000        1        0     0.329       NaN         1
#> 6: 4 1    1.934   0.3536        3        1     0.645    0.3536         1
#> 7: 4 2    1.968   0.0000        3        0     0.656       NaN         1
#> 8: 5 2    0.404   0.8995        1        1     0.404    0.8995         1
#>    d2_mean_b
#> 1:         1
#> 2:         1
#> 3:         1
#> 4:         1
#> 5:       NaN
#> 6:         1
#> 7:       NaN
#> 8:         1
dcast(dt, x + y ~ z, fun=list(sum,mean), value.var=list("d1","d2")) # value.var
#>    x y d1_sum_a d1_sum_b d2_mean_a d2_mean_b
#> 1: 1 1    0.000   0.3141       NaN         1
#> 2: 1 2    0.675   0.7524         1         1
#> 3: 2 1    0.722   1.9725         1         1
#> 4: 2 2    1.062   0.0657         1         1
#> 5: 3 2    0.329   0.0000         1       NaN
#> 6: 4 1    1.934   0.3536         1         1
#> 7: 4 2    1.968   0.0000         1       NaN
#> 8: 5 2    0.404   0.8995         1         1
```

- melt

```
melt(data, id.vars, measure.vars,
    variable.name = "variable", value.name = "value",
    ..., na.rm = FALSE, variable.factor = TRUE,
    value.factor = FALSE,
    verbose = getOption("datatable.verbose"))
```

:

```
ChickWeight = as.data.table(ChickWeight)
setnames(ChickWeight, tolower(names(ChickWeight)))
DT <- melt(as.data.table(ChickWeight), id=2:4) # calls melt.data.table
DT
#>      time chick diet variable value
#>   1:    0     1    1   weight    42
#>   2:    2     1    1   weight    51
#>   3:    4     1    1   weight    59
#>   4:    6     1    1   weight    64
#>   5:    8     1    1   weight    76
#>  ---
#> 574:   14    50    4   weight   175
#> 575:   16    50    4   weight   205
#> 576:   18    50    4   weight   234
#> 577:   20    50    4   weight   264
#> 578:   21    50    4   weight   264
```

2.4.6

uniqueN length(unique(x)),

```
x <-sample(1:10,50,replace = TRUE)
uniqueN(x)
#> [1] 10

DT <- data.table(A = rep(1:3, each=4), B = rep(1:4, each=3),
                 C = rep(1:2, 6), key = "A,B")

uniqueN(DT, by = key(DT))
#> [1] 6
uniqueN(DT)
#> [1] 10
```

2.4.7 rleid

0011001110111101 1 1 2 2 3 3 4 4 4 5 6 6 6 6 7 8

```
rleid(c(0,0,1,1,0,0,1,1,1,0,1,1,1,1,0,1))
#>  [1] 1 1 2 2 3 3 4 4 4 5 6 6 6 6 7 8
```

```
rleid(..., prefix=NULL)
rleidv(x, cols=seq_along(x), prefix=NULL)
```

```
DT = data.table(grp=rep(c("A", "B", "C", "A", "B"), c(2,2,3,1,2)), value=1:10)
rleid(DT$grp) # get run-length ids
#>  [1] 1 1 2 2 3 3 3 4 5 5
rleidv(DT, "grp") # same as above
#>  [1] 1 1 2 2 3 3 3 4 5 5
rleid(DT$grp, prefix="grp") # prefix with 'grp'
#>  [1] "grp1" "grp1" "grp2" "grp2" "grp3" "grp3" "grp3" "grp4" "grp5" "grp5"
```

2.4.8 shift

,

```
x = 1:5
# lag with n=1 and pad with NA (returns vector)
shift(x, n=1, fill=NA, type="lag")
#> [1] NA  1  2  3  4
```

n n type , n=-1 and type=‘lead’ n=1 and type=’lag’

data.table

```
DT = data.table(year=2010:2014, v1=runif(5), v2=1:5, v3=letters[1:5])
cols = c("v1","v2","v3")
anscols = paste("lead", cols, sep="_")
DT[, (anscols) := shift(.SD, 1, 0, "lead"), .SDcols=cols]
```

2.5

2.5.1

1.

```
#

fun <- function(x){
  x <- x^2+1
}

DT <-  data.table(x=rep(c("b","a","c"),each=3), v=c(1,1,1,2,2,1,1,2,2), y=c(1,3,6), a=1:9, b=9:1)

DT[,.(newcol=fun(y)),by=.(x)]
#>    x newcol
#> 1: b      2
#> 2: b     10
#> 3: b     37
#> 4: a      2
#> 5: a     10
#> 6: a     37
#> 7: c      2
#> 8: c     10
#> 9: c     37

#Not run
#DT[,lapply(.SD,fun),.SDcols=c('y','a'),by=.(x)] #


#
#Not run

# myfun <- function(x){
#   return(x)
# }
#
# dt <- dt[,colnames(dt):=lapply(.SD[,1:ncol(dt)],myfun)] #
```

2.5.2

by .

1. rollup

id=TRUE , by . .

```
#Usage
#rollup(x, j, by, .SDcols, id = FALSE, ...)
n = 24L
```

```
set.seed(25)
DT <- data.table(
    color = sample(c("green","yellow","red"), n, TRUE),
    year = as.Date(sample(paste0(2011:2015,"-01-01"), n, TRUE)),
    status = as.factor(sample(c("removed","active","inactive","archived"), n, TRUE)),
    amount = sample(1:5, n, TRUE),
    value = sample(c(3, 3.5, 2.5, 2), n, TRUE)
)
rollup(DT, j = sum(value), by = c("color","year","status")) # default id=FALSE
#>      color       year   status   V1
#>  1:    red 2015-01-01   active  3.5
#>  2:  green 2015-01-01 inactive  5.5
#>  3:  green 2014-01-01 archived  3.5
#>  4:  green 2015-01-01 archived  2.0
#>  5: yellow 2014-01-01   active  4.5
#>  6:    red 2013-01-01 inactive  2.0
#>  7:  green 2011-01-01   active  6.0
#>  8:    red 2014-01-01 inactive  2.5
#>  9:  green 2011-01-01 archived  2.5
#> 10: yellow 2015-01-01   active  2.0
#> 11:    red 2012-01-01 archived  2.0
#> 12:    red 2011-01-01  removed  3.5
#> 13:  green 2014-01-01 inactive  8.0
#> 14:  green 2011-01-01  removed  2.0
#> 15: yellow 2012-01-01 archived  2.5
#> 16:    red 2013-01-01  removed  3.5
#> 17:  green 2013-01-01   active  3.0
#> 18:  green 2014-01-01  removed  2.5
#> 19:    red 2011-01-01 archived  3.0
#> 20:    red 2015-01-01     <NA>  3.5
#> 21:  green 2015-01-01     <NA>  7.5
#> 22:  green 2014-01-01     <NA> 14.0
#> 23: yellow 2014-01-01     <NA>  4.5
#> 24:    red 2013-01-01     <NA>  5.5
#> 25:  green 2011-01-01     <NA> 10.5
#> 26:    red 2014-01-01     <NA>  2.5
#> 27: yellow 2015-01-01     <NA>  2.0
#> 28:    red 2012-01-01     <NA>  2.0
#> 29:    red 2011-01-01     <NA>  6.5
#> 30: yellow 2012-01-01     <NA>  2.5
#> 31:  green 2013-01-01     <NA>  3.0
#> 32:    red       <NA>     <NA> 20.0
#> 33:  green       <NA>     <NA> 35.0
#> 34: yellow       <NA>     <NA>  9.0
#> 35:   <NA>       <NA>     <NA> 64.0
#>      color       year   status   V1
#rollup(DT, j = sum(value), by = c("color","year","status"), id=TRUE)
```

, , Excel , R , , .

- rollup

```
set.seed(25)
N <- 1000
dt <- data.table(col1=sample(LETTERS[1:5],N,replace = T),col2=sample(letters[1:5],N,replace = T),

rollup(dt,j=c(list(sum(num))),by=c('col1','col2'))
#>     col1 col2     V1
#>  1:    E    a  19926
#>  2:    D    a  20966
#>  3:    A    d  12927
#>  4:    A    b  20862
#>  5:    A    c  15331
#>  6:    B    d  15414
#>  7:    C    e  20794
#>  8:    D    e  16110
#>  9:    C    d  22152
#> 10:    A    a  18378
#> 11:    C    c  19474
#> 12:    E    d  18831
#> 13:    B    b  19941
#> 14:    C    a  19652
#> 15:    E    c  16734
#> 16:    E    e  24137
#> 17:    E    b  21988
#> 18:    D    b  16607
#> 19:    B    c  25720
#> 20:    B    a  22109
#> 21:    A    e  18724
#> 22:    C    b  24323
#> 23:    D    d  20508
#> 24:    D    c  19668
#> 25:    B    e  29224
#> 26:    E <NA> 101616
#> 27:    D <NA>  93859
#> 28:    A <NA>  86222
#> 29:    B <NA> 112408
#> 30:    C <NA> 106395
#> 31: <NA> <NA> 500500
#>     col1 col2     V1
#       total
#rollup(dt,j=c(list(total=sum(num))),by=c('col1','col2'))
# id=TRUE ,  grouping
#rollup(dt,j=c(list(total=sum(num))),by=c('col1','col2'),id=TRUE)
```

2.groupingsets

. SQL GROUPING SETS . postgresql

```
res <- groupingsets(DT, j = c(list(count=.N), lapply(.SD, sum)), by = c("color","year"
             sets = list("color", c("year","status"), character()), id=TRUE)
head(res)
#>    grouping  color       year   status count amount value
#> 1:        3    red       <NA>     <NA>     7     19  20.0
#> 2:        3  green       <NA>     <NA>    13     43  35.0
#> 3:        3 yellow       <NA>     <NA>     4     10   9.0
#> 4:        4   <NA> 2015-01-01   active     2      8   5.5
#> 5:        4   <NA> 2015-01-01 inactive     2      5   5.5
#> 6:        4   <NA> 2014-01-01 archived     1      3   3.5
```

groupingsets sets , list() , character(), . by , . sql “()”.

sql.

```
select color ,year, status,count(*) count,sum(amount) amount,sum(value) value
FROM dbo.DT
GROUP BY
GROUPING SETS(
(color),
(year,status),
() ----   character()
)
```

cube() , ?cube

2.5.3

-

tstrsplit()

```
n <- 10
dt <- data.table(name=LETTERS[1:n],char=rep(' - -R- - '),n)
res <- dt[,.(newcol=tstrsplit(char,'-')),by=.(name)]
head(res)
#>    name newcol
#> 1:    A
#> 2:    A
#> 3:    A      R
#> 4:    A
#> 5:    A
#> 6:    B
```

-

```
res[,.(char=paste0(newcol,collapse = '-')),by=.(name)]
#>     name          char
#>  1:    A  - -R- -
#>  2:    B  - -R- -
#>  3:    C  - -R- -
#>  4:    D  - -R- -
#>  5:    E  - -R- -
#>  6:    F  - -R- -
#>  7:    G  - -R- -
#>  8:    H  - -R- -
#>  9:    I  - -R- -
#> 10:    J  - -R- -
#
#res[,.(char=stringr::str_c(newcol,collapse = '-')),by=.(name)]
# A  - -R- -
# B  - -R- -
# C  - -R- -
# D  - -R- -
# E  - -R- -
# F  - -R- -
# G  - -R- -
# H  - -R- -
# I  - -R- -
# J  - -R- -
```

Chapter 3

database

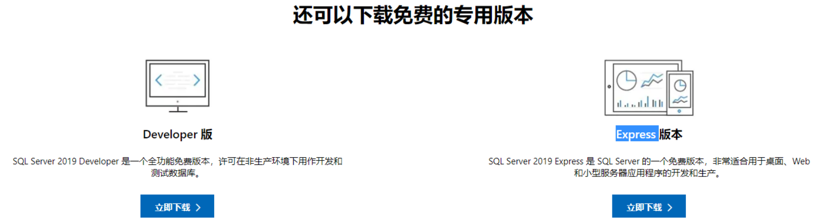
R , MSSQL,Oracle,mysql , “R ”
R DBI,RODBC,RMySQL,ROracle,odbc DBI RODBC , , ,
; Excel, 50 vlookup Excel

3.1

Windows R ETL
MS SQL Server

- Win

MS (Developer Express)



还可以下载免费的专用版本

Developer 版

SQL Server 2019 Developer 是一个全功能免费版本，许可在非生产环境下用作开发和测试数据库。

Express 版本

SQL Server 2019 Express 是 SQL Server 的一个免费版本，非常适合用于桌面、Web 和小型服务器应用程序的开发和生产。

Figure 3.1:

`SSMS` MS SQL SERVER

- Linux

SQL Server 2019 Ubuntu 20.04 Ubuntu 18.04 16.04 /ubuntu/18.04/ /ubuntu/16.04/ /ubuntu/20.04/

```
#
wget -qO- https://packages.microsoft.com/keys/microsoft.asc | sudo apt-key add -

#   SQL Server 2019   Microsoft SQL Server Ubuntu
sudo add-apt-repository "$(wget -qO- https://packages.microsoft.com/config/ubuntu/20.04

# sudo add-apt-repository "$(wget -qO- https://packages.microsoft.com/config/ubuntu/18

#   SQL Server
sudo apt-get update
sudo apt-get install -y mssql-server

#
systemctl status mssql-server --no-pager
```

sql server

R

3.2 DBI

3.2.1

```
install.packages('DBI')
```

3.2.2

- MS SQL SERVER

172.16.88.2(IP)

```
library(DBI)
con <- dbConnect(
  drv = odbc::odbc(), Driver = "SQL Server", server = "172.16.88.2",database = "spb", u
)
```

windows DBI encoding win sqlserver encoding = “GBK”

```
library(DBI)
#      encoding
con <- dbConnect(
  drv = odbc::odbc(), Driver = "SQL Server", server = "172.16.88.2",
  database = "spb", uid = "zhongyf", pwd = "Zyf123456", encoding = "GBK"
)
#            ODBC Driver 17 for SQL Server

Drivers_tbl <- odbc::odbcListDrivers()
head(Drivers_tbl)
```

,

```
con <- dbConnect(
  drv = odbc::odbc(), Driver = "ODBC Driver 17 for SQL Server",
  server = "172.16.88.2", database = "spb", uid = "zhongyf", pwd = "Zyf123456"
)

#   936
sql <- "SELECT COLLATIONPROPERTY( 'chinese_prc_ci_as', 'codepage' )"

dbGetQuery(con,sql)

# same above
# dbExecute(con,sql)

#
DBI::dbDisconnect(con)
```

- mysql

`MySQL()` `RMySQL` `<MySQLDriver>` MySQL

```
library(RMySQL)
con <- dbConnect(MySQL(),
  dbname = "test", user = "test_admin", password = "30HL1234M7# lD6gxjB",
  host = "prd-public-mypersonal.mysql.test.zhangjiabei.rds.aliyuncs.com"
)
```

```
con <- DBI::dbConnect(odbc::odbc(),
  Driver = "MySQL ODBC 8.0 Unicode Driver",
  Server = "localhost", UID = "root", PWD = "123456", Database = "mysql",
  Port = 3306
)
```

mysql 3306, 3306

3.2.3 sql

dbGetQuery() DBI con ,dbExecute()

```
# dbGetQuery
res_table <- dbGetQuery(con,'select * from table') #  sql

#dbReadTable
dbReadTable(con,'tbl_name') #

# dbSendQuery
res <- dbSendQuery(conn = con,statement = 'select * FROM tab')
dbFetch(res)
dbClearResult(res)

# dbExecute
dbExecute(con,'delete from table where num <=1000') #

# dbWriteTable()
#   ,  ,    df,overwrite  ,append
dbWriteTable(conn = con,name = ' ',value = df,overwrite=TURE,append=FALSE)
```

3.2.4

, , .

```
con <- dbConnect(
  drv = odbc::odbc(),
  Driver = "ODBC Driver 17 for SQL Server", server = "172.16.88.2",
  database = "spb", uid = "zhongyf", pwd = "Zyf123456", encoding = "GBK"
)

#
dbGetInfo(con)
```

```
#
dbListTables(con) #win

#
dbRemoveTable(con,'tbl_name')

#
dbDisconnect(con)
```

3.3 odbc

Connect to ODBC databases (using the DBI interface)

odbc DBI

odbc (SQL Server, Oracle, MySQL,PostgreSQL,SQLite) odbc DBI DBI

1.

```
#
install.packages('odbc')
```

2.

Win Sql Server encoding

linux odbc SqlServer, charset=zh_CN.GBK gbk

```
library(odbc)
con <- odbc::dbConnect(odbc(),
  Driver = "SQL Server", Server = "Vega", Database = "ghzy",
  Trusted_Connection = "True"
) # windows
# con <- dbConnect(odbc::odbc(), .connection_string = "Driver={SQL Server};
#                                  server=Vega;database=ghzy;uid=zhongyf;pwd=Zyf123456;", timeout
con
## Not run
# Win
con_spb <- dbConnect(odbc(), .connection_string = "driver={ODBC Driver 17 for SQL Server};server=
                     timeout = 10, timezone = "Asia/Shanghai",encoding = 'gbk')
#Linux
con_dd <- dbConnect(odbc::odbc(), .connection_string = "driver={ODBC Driver 17 for SQL Server};se
                 database=aojo_dd;uid=wj;pwd=12qw#$ER;charset=zh_CN.GBK", timeout = 10)
```

3.

```
dt <- odbc::dbGetQuery(con,'select * from DT')
head(dt)
```

4.

```
odbc::dbWriteTable(con,name = ' ',value = dt,overwrite = T ) #
odbc::dbWriteTable(con,name = ' ',value = dt,append = T ) #
```

3.4 RODBC

RODBC R ODBC , ODBC .

1.

```
install.packages('RODBC')
```

2.SQL SERVER

```
library(RODBC)
con <- odbcDriverConnect("driver={SQL Server};server=192.168.2.62;database=dbname;uid=
con
RODBC::sqlQuery(con,'select * from test')
```

WINDOWS , .

-

```
odbc::odbcListDrivers()
```

-

ODBC for sql server driver

3.

```
#ODBC Driver 17 for SQL Server
cn <- odbcDriverConnect("Driver={ODBC Driver 17 for SQL Server};Server=localhost;Database=name;UI
```

sql server sql server

3.5 ROracle

oracle R Oracle Oracle Instant Client

1.

oracle 32 64

2.

```
OCI_INC='D:\app\zhongyf\product\11.2.0\client_1\oci\include'
OCI_LIB64='D:\app\zhongyf\product\11.2.0\client_1\BIN'
```

linxu `Roracle`

3.

Roracle Rtools oracle ,

ROracle Oracle Instant Client,

```
install.packages('ROracle')
```

4.

`Roracle` `DBI`

```
library(ROracle)
drv <-dbDriver("Oracle")
connect.string <- '(DESCRIPTION =
                    (ADDRESS = (PROTOCOL = TCP)(HOST = 192.16.88.129)(PORT = 1521))
                  (CONNECT_DATA =
                      (SERVER = DEDICATED)
                    (SERVICE_NAME = bidev)
                  ))' #

con <- dbConnect(drv,username = "query", password = "query",dbname = connect.string)
```

5.

oracle

linux Renviron [/opt/R/4.0.2/lib/R/etc/Renviron]

```
#
select userenv('language') from dual
Sys.setenv(NLS_LANG="SIMPLIFIED CHINESE_CHINA.AL32UTF8")
```

3.6 RMySQL

RMySQL mysql mysql . RMariaDB

3.6.1

Win

```
#On recent Debian or Ubuntu install libmariadbclient-dev

sudo apt-get install -y libmariadbclient-dev
#On Fedora, CentOS or RHEL we need mariadb-devel:

sudo yum install mariadb-devel
#On OS-X use mariadb-connector-c from Homebrew:

brew install mariadb-connector-c
```

```
install.packages('RMySQL')
```

3.6.2

```
library(RMySQL)
con <- RMySQL::dbConnect(drv = RMySQL::MySQL(),host='localhost',dbname="mysql",username="root",pa
```

`RMariaDB` `RMySQL`

```
install.packages('RMariaDB')
library(RMariaDB)
con <- RMySQL::dbConnect(drv = RMariaDB::MariaDB() ,host='localhost',dbname="dbtest",username="ro
```

3.7

R

3.7.1

R win

- MS SQL SERVER

encoding win RODBC odbc encoding

```
# win
con_spb <- dbConnect(odbc(),
  .connection_string =
    "driver={ SQLServer};server=172.16.88.2;database=spb;uid=zhongyf;pwd=Zyf123456",
  timeout = 10, timezone = "Asia/Shanghai", encoding = "gbk"
)

# linux
con_spb <- dbConnect(odbc(),
                     .connection_string =
                       "driver={ODBC Driver 17 for SQL Server};server=172.16.88.2;database=spb;ui
                     timeout = 10, timezone = "Asia/Shanghai", encoding = "utf8"
)
```

- MySQL

1.

```
#
dbSendQuery(con,'SET NAMES gbk')
```

2.ODBC

ODBC , ,

3.7.2

mysql , RMySQL mysql, Navicat , Authentication plugin 'caching_sha2_password' cannot be loaded

mysql8 mysql_native_password, mysql8 , caching_sha2_password,

```
--cmd
mysql -u root -p
--
password:

--
ALTER USER 'root'@'localhost' IDENTIFIED BY 'password' PASSWORD EXPIRE NEVER;   #
---ALTER USER 'root'@'%' IDENTIFIED BY 'password' PASSWORD EXPIRE NEVER;

ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY 'password'; #
```

3.7.3

, IT MS SQL SERVER

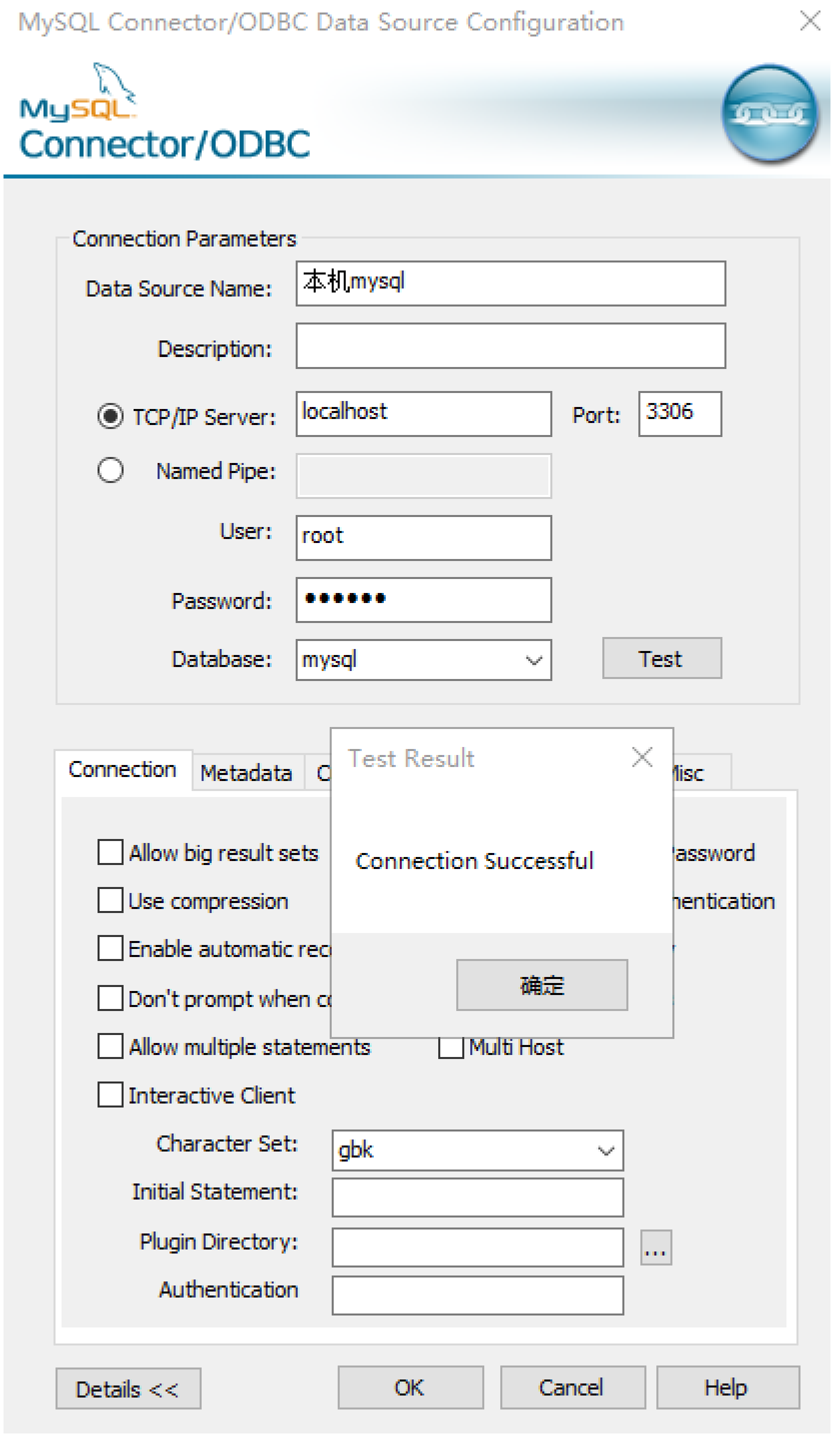
, Mysql 3306; Rds , DBA

3.8 dbplyr

dbplyr dplyr SQL

dbplyr SQL,

- dbplyr dplyr
-
- R dbplyr
- dplyr dbplyr



MySQL Connector/ODBC Data Source Configuration

Connection Parameters

Data Source Name: 本机mysql

Description:

● TCP/IP Server: localhost Port: 3306

○ Named Pipe:

User: root

Password: ••••••

Database: mysql

Test Result

Connection Successful

□ Allow big result sets

□ Use compression

□ Enable automatic rec

□ Don't prompt when c

□ Allow multiple statements

□ Multi Host

□ Interactive Client

Character Set: gbk

Initial Statement:

Plugin Directory:

Authentication

Figure 3.2: ODBC

3.8.1

```r
library(dplyr)
library(dbplyr)

mf <- memdb_frame(x = 1, y = 2)

mf %>%
  mutate(
    a = y * x,
    b = a ^ 2,
  ) %>%
  show_query()
```

```r
library(dplyr)
#connect database
con <- DBI::dbConnect(RSQLite::SQLite(), path = ":memory:")
#
copy_to(con, nycflights13::flights, "flights",
  temporary = FALSE,
  indexes = list(
    c("year", "month", "day"),
    "carrier",
    "tailnum",
    "dest"
  )
)

#
#dbListTables(con)

#tbl()  flights

flights_db <- tbl(con, "flights")
flights_db

#
flights_db %>% select(year:day, dep_delay, arr_delay)
flights_db %>% filter(dep_delay > 240)
flights_db %>%
  group_by(dest) %>%
  summarise(delay = mean(dep_time))
```

sql dplyr .

```
tailnum_delay_db <- flights_db %>%
  group_by(tailnum) %>%
  summarise(
    delay = mean(arr_delay,na.rm = T),
    n = n()
  ) %>%
  arrange(desc(delay)) %>%
  filter(n > 100)
tailnum_delay_db
tailnum_delay_db %>% show_query()
tailnum_delay <- tailnum_delay_db %>% collect() #     R
```

3.8.2

lubridate dbplyr

Oracle to_date

group_by ,

- date

```
#
get_sales_data <- function(con,...,start_date,end_date,brand_name,channel_type = NULL ,area_name

  store_table <- store(con,brand_name = brand_name,channel_type = channel_type ,area_name = area_

  sku_table <- sku(con,category_name =  category_name ) #

  tbl(con, in_schema("DW", "DW_SALE_SHOP_F")) %>% #DW
    select(BILL_DATE1, SKU_NO, SHOP_NO, BILL_QTY, BILL_MONEY2, PRICE) %>%
    filter(between(
      BILL_DATE1, to_date(start_date, "yyyy-mm-dd"),
      to_date(end_date, "yyyy-mm-dd")
    )) %>%
    mutate(  = year(BILL_DATE1),   = month(BILL_DATE1)) %>%
    inner_join(store_table) %>%
    inner_join(sku_table) %>%
    group_by(...) %>%
    summarise(
        = sum(BILL_MONEY2, na.rm = TRUE),
        = sum(BILL_QTY, na.rm = TRUE),
         = sum(BILL_QTY * PRICE, na.rm = TRUE)) %>%
    collect() %>%
    mutate(  :=   /   ) %>%
    arrange(...)


  # return(res)
}
```

- like

```
mf %>%
  filter(x %LIKE% "%foo%") %>%
  show_query()
```

-

  sql()

```
mf %>%
  transmute(factorial = sql("x!")) %>%
  show_query()
```

3.9

DBI https://dbi.r-dbi.org/reference/

dbplyr https://dbplyr.tidyverse.org/

rstudio https://db.rstudio.com/databases

https://www.connectionstrings.com/

Roracle http://www.zhongyufei.com/2020/07/25/roracle-install/

https://www.r-consortium.org/blog/2017/05/15/improving-dbi-a-retrospect

Chapter 4

stringr

, .R stringr ,

R for Data Science

Excle : `left`,`len`,`mid`,`find`,`Proper`,`rept`,`trim`,`upper`,`substitute`,`concatenate`, `Excle2019` `concat`,`TEXTJOIN`

`stringr`

- https://cran.r-project.org/web/packages/stringr/vignettes/stringr.html

4.1

4.1.1

`R`

-
- R
- R sql

4.1.2

```
#install.packages('stringr')
library(stringr)
char <- "  \'  \'"   #      ,
char
```

, writeLines() cat()

```
x <- c("\"", "\\")
x
#> [1] "\"" "\\"
writeLines(x)
cat(char)
#> "
#> \
```

, “||”

```
str_remove(string = 'a||b',pattern = "\\|\\|")
```

\n, \t , , , .

4.1.3

```
char <- " R   "
str_length(char)
#
str_length(c("a", "R for data science", NA))
```

4.1.4

R python , :

R

```
#base R
paste0('a','b')

#stringr
str_c("a","b")
str_c("a", "b", sep = ", ") #sep
```

Python

```
'a' + 'b'
```

,stringr

```
#base R
paste0(c('a','b','d','e'),collapse = ',')
#stringr
str_c(c('a','b','d','e'),collapse = ',')  #collapse
```

-

```
library(data.table)
dt <- data.table(col=rep('a',10),letters=letters[1:10])
dt[,newcol:=str_c(letters,collapse = '|'),by=.(col)][]
```

-

```
#             -  - ,
dt <- data.table(col='a',letters=str_c(letters[1:10],collapse = '|'))

my_str_split <- function(x){

  str_split(x,pattern = "\\|") %>% unlist()  #str_split
}

dt[,list(newcol=my_str_split(letters)),by=.(col)]
```

4.1.5 R4.0

```
char <- r"(\\a\ab\d\e\f)" #windows   ,
char
```

```
char <- "  \'  \'"
cat(char)

char <- r"(  'R '  )"
cat(char)
```

4.2

4.2.1

Excle left,mid,right

str_sub() :

string:

start: 1L,

end: -1L,

```
# end 3   -3
str_sub(string = ' R   ',start = 2,end = 3)
str_sub(string = ' R   ',start = 2,end = -3)
```

4.2.2

,str_match() pattern( ) . .

```
?str_match()
?str_match_all()
?str_extract()
?str_extract_all()
```

str_extract() ,str_match() .

```
#     < >
strings <- c('       ,  ,   ,   ,  ,  .')
str_extract(strings,'  ')
str_match(strings,'  ')
```

-

$$4e00 - 9fa5$$

```
str_extract_all(strings,'[\u4e00-\u9fa5]') # list
```

-

[0-9]; :[a-zA-Z]

```
strings <- c('00123545','LOL league of legends')
str_extract_all(strings,'[0-9]')
str_extract_all(strings,'[a-zA-Z]')
```

4.2.3

str_pad()

,1,2,3,4,5,6,7,8,9,10,11,12 01,02,03,04,05,06,07,08,09,10,11,12. , :

```
str_pad(string = 1:12,width = 2,side = 'left',pad = '0')
```

4.2.4

excel trim

```
# side   both  left right
str_trim(' ab af ',side = 'both')
```

4.2.5

str_split()

```
#    ,
str_split("a,b,d,e",pattern = ',')

str_split('ab||cd','\\|\\|') %>% unlist()
# same above
#str_split('ab||cd','\\|\\|') %>% purrr::as_vector()
```

```
fruits <- c(
  "apples and oranges and pears and bananas",
  "pineapples and mangos and guavas"
)

str_split(fruits, " and ")
```

4.2.6

str_replace() str_replace_all()

```
fruits <- c("one apple", "two pears", "three bananas")
str_replace(fruits, "[aeiou]", "-")
str_replace_all(fruits, "[aeiou]", "-")
```

4.2.7

str_remove(),str_remove_all()

```
fruits <- c("one apple", "two pears", "three bananas")
str_remove(fruits, "[aeiou]")
str_remove_all(fruits, "[aeiou]")
```

```
str_replace_all(string = ' d a  b ',pattern = ' ',replacement = '')
```

4.2.8

- str_subset() str_which()

```
fruit <- c("apple", "banana", "pear", "pinapple")
str_subset(fruit, "a")
str_which(fruit, "a") #
```

```
#str_which  which(str_detect(x,pattern))
#str_which()

#str_subset x[str_detect(x,pattern)]
#str_subset()

#
set.seed(24)
dt <- data.table::data.table(col=sample(c(letters,1:10),100,replace = T))
head(dt[str_which(col,pattern = '[a-z]')])
```

- str_dup()

```
fruit <- c("apple", "pear", "banana")
str_dup(fruit, 2)
str_dup(fruit, 1:3)
str_c("ba", str_dup("na", 0:5))
```

- str_starts() str_ends()

str_detect() .

```
str_starts('abd','a')
str_detect('abd','^a')

str_ends('abd','d')
str_detect('abd','a$')
```

-

```
dog <- "The quick brown dog"
str_to_upper(dog)
str_to_lower(dog)
str_to_title(dog)
str_to_sentence("the quick brown dog")
```

4.3 R Excel

stringr Rcpp

- left

```
r_left <- function(str,num){
  str_sub(string = str,start = 1,end = num)
}
r_left(' R   ',3)
```

- right

```
r_right <- function(str,num){
  str_sub(string = str,start = str_length(str) - num + 1)
}
r_right(' R   ',3)
```

- mid

```r
r_mid <- function(str,start,num){
  str_sub(string = str,start = start,end = start + num -1)
}
r_mid(' R   ',3,3)
```

Chapter 5

lubridate

, R `lubridate`

`lubridate`

`Excel` `Power Pivot` DAX

-

`date`,`datediff`,`datevalue`,`edate`,`eomonth`,`quarter`,`TIMEVALUE`

-

`dateadd`,`DATESBETWEEN`,`DATESMTD`,`TOTALMTD`,`TOTALQTD`,`TOTALYTD`

`Excel`

DAX `R` DAX R

`R` `1970-01-01`,Excel `1900-01-01`, 25568 R Excel

R : 2021-04-29 : 18746, Excel 2021-04-29 :44314, 25568.

5.1

`lubridate`

5.1.1

```
install.packages("tidyverse")
#    lubridate
install.packages('lubridate')
#
devtools::install_github("tidyverse/lubridate")
```

```
#
library(lubridate,warn.conflicts = FALSE)
```

5.1.2

- now

```
now(tzone = 'Asia/Shanghai')
#base R
base::Sys.time()
```

CST

```
Sys.timezone()
# windows
# linux  "Asia/Shanghai"
```

- today

```
today(tzone = 'Asia/Shanghai')
#base R
base::Sys.Date()
```

5.1.3

```
#
year(now())
#
month(now())
#
```

```
yday(now())
#
mday(now())
#
wday(now(),label = TRUE,week_start = 1)
#
hour(now())
#
minute(now())
#
second(now())
```

5.2

with_tz() force_tz()

```
time <- ymd_hms("2020-12-13 15:30:30")
time

# Changes printing
with_tz(time, "Asia/Shanghai")
# Changes time
force_tz(time, "Asia/Shanghai")
```

5.3

BI

```
#
ymd(20200604)
ymd('20200604')
mdy(06042020)
dmy(04062020)
```

unix .POSIXct() .

unix

```
.POSIXct(1591709615)
ymd_hms(.POSIXct(1591709615))
```

unix mysql RDS lubridate tz

```
ymd_hms(.POSIXct(1591709615),tz = 'asia/shanghai')
```

CST : ;UTC : (UTC)

: UTC 0 CST 8 8 .

- https://home.kpn.nl/vanadovv/time/TZworld.html#asi

```
lubridate::now()
as_datetime(now()) #  UTC
as_datetime(now(),tz = 'asia/shanghai')
```

5.4

`make_date` `make_datetime` “UTC”

```
make_date(year = year(today()), month = month(today()), day = day(today()), tz = "asia/
make_datetime(
  year = year(today()),
  month = month(today()),
  day = day(today()),
  hour = hour(now()),
  min = minute(now()),
  sec = second(now()),
  tz = "asia/shanghai"
)
```

```
as_datetime('2020-01-09 09:15:40',tz='asia/shanghai')
as_date('2020-01-09') #ymd
# same above
#as_date('2020/01/09')
#as_date('20200109')
```

5.5

lubridate interveal

```r
arrive <- ymd_hms("2020-12-04 12:00:00", tz = "asia/shanghai")
arrive

leave <- ymd_hms("2020-12-10 14:00:00", tz = "asia/shanghai")
leave

res <- interval(arrive, leave)
# same above
res <- arrive %--% leave
```

```r
jsm <- interval(ymd(20201020, tz = "asia/shanghai"), ymd(20201231, tz = "asia/shanghai"))
jsm
int_overlaps(jsm, res)
```

?interveal

```r
interval(start = NULL, end = NULL, tzone = tz(start))

start %--% end

is.interval(x)

int_start(int)

int_start(int) <- value

int_end(int)

int_end(int) <- value

int_length(int)

int_flip(int)

int_shift(int, by)

int_overlaps(int1, int2)
```

```
int_standardize(int)

int_aligns(int1, int2)

int_diff(times)
```

5.6

number line Because the timeline is not as reliable as the number line

```
minutes(2)
dminutes(2)
dhours(2)
```

```
leap_year(2019)
ymd(20190101) + dyears(1)
ymd(20190101) + years(1)

leap_year(2020)
ymd(20200101) + dyears(1)   #
ymd(20200101) + years(1)
```

lubridate

```
meeting <- ymd_hms("2020-12-01 09:00:00", tz = "asia/shanghai")
meeting <- meeting + weeks(0:5)
meeting %within% jsm
```

```
res / ddays(1)
res / dminutes(1)


res %/% months(1)
res %% months(1)
```

as.period

```
as.period(res %% months(1))
```

```
jan31 <- ymd("2020-01-31")
jan31 + months(0:11)
```

lubridate NA

%m+% %m-%

```
jan31 %m+% months(0:11)
jan31 %m-% months(0:11)
```

5.7

5.7.1

floor_date()

```
floor_date(today(),unit = 'year')
floor_date(today(),unit = 'month') # rollback
floor_date(today(),unit = 'week')
```

-

```
n <- 1
date <- today()
# current
current_start_date <-  floor_date(date,unit = 'year')
current_start_date
date
# last year
last_start_date <- floor_date(date,unit = 'year') %m-% years(n)
last_start_date
last_end_date <- date %m-% years(n)
last_end_date
```

“month”

-

%m+% %m-%

```
as_date('2020-03-30') %m-% months(1)
today()
today() %m-% months(1)
```

```
#
bill_date <- as_date((as_date('2019-01-01'):as_date('2020-12-01')))
area <-  sample(c(' ',' ',' ',' '),size = length(bill_date),replace = TRUE)
dt <- tibble::tibble(bill_date = bill_date ,money = sample(80:150,size = length(bill_da
head(dt)
```

```
library(dplyr,warn.conflicts = FALSE)

y_to_y <- function(.dt,date,n = 1,...){

  date <- ymd(date)

  if(is.na(date)){
    stop('      20200101')
  }

  # current
 current_start_date <-  floor_date(date,unit = 'year')

 # last year
 last_start_date <- floor_date(date,unit = 'year') %m-% years(n)
 last_end_date <- date %m-% years(n)

 .dt %>% mutate(   = case_when(between(bill_date,current_start_date,date) ~ " ",
              between(bill_date,last_start_date,last_end_date) ~ " ",
              TRUE ~ " ")) %>%
   filter(  != " ") %>%
   group_by(...) %>%
   summarise(  = sum(money,na.rm = TRUE)) %>%
   ungroup()

 #%>% pivot_wider(names_from = ' ',values_from = ' ')

}
```

```
y_to_y(dt,date = '20200101',n = 1,area, )
```

5.7.2

c('2001/2/13 10:33','1/24/13 11:16') ;

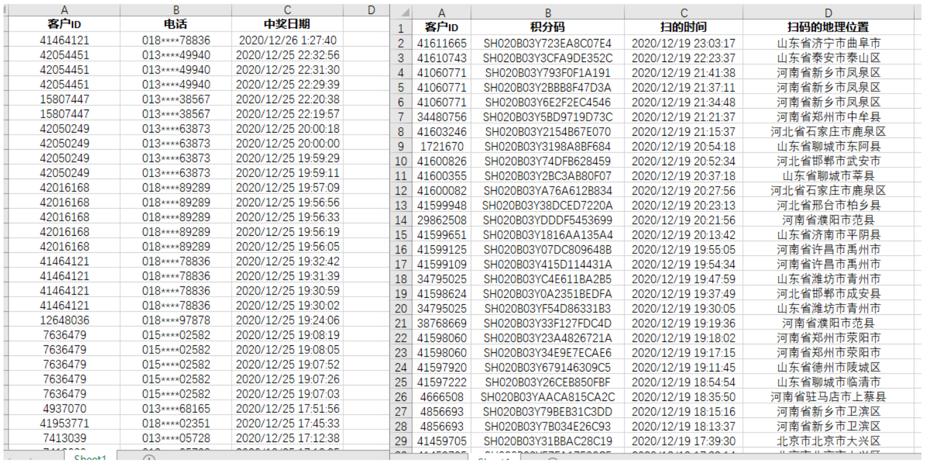
```
library(lubridate)
library(tidyverse)

date1 <- c('2001/2/13 10:33','1/24/13 11:16')

myfun <- function(x){

  n_length <- length(x)
  res <- vector(length = n_length)

  for(i in 1:n_length){
    n <- strsplit(x[i],'/') %>% `[[`(1) %>% `[[`(1)
    if(str_length(n)==4){
      res[i] <- ymd_hm(x[i],tz = 'Asia/Shanghai')
    } else {
      res[i] <- mdy_hm(x[i],tz = 'Asia/Shanghai')
    }
  }
  as_datetime(res,tz = 'Asia/Shanghai')
}

myfun(date1)
```

5.7.3

ID ,

“ ”

```
testfun <- function(x,y){
  result <- data.frame() #
  n  <-  length(x)
  for( i in 1:n){
    res <- x[i]-y
```



客户ID	电话	中奖日期
41464121	018****78836	2020/12/26 1:27:40
42054451	013****49940	2020/12/25 22:32:56
42054451	013****49940	2020/12/25 22:31:30
42054451	013****49940	2020/12/25 22:29:39
15807447	013****38567	2020/12/25 22:20:38
15807447	013****38567	2020/12/25 22:19:57
42050249	013****63873	2020/12/25 20:00:18
42050249	013****63873	2020/12/25 20:00:00
42050249	013****63873	2020/12/25 19:59:29
42050249	013****63873	2020/12/25 19:59:11
42016168	018****89289	2020/12/25 19:57:09
42016168	018****89289	2020/12/25 19:56:56
42016168	018****89289	2020/12/25 19:56:33
42016168	018****89289	2020/12/25 19:56:19
42016168	018****89289	2020/12/25 19:56:05
41464121	018****78836	2020/12/25 19:32:42
41464121	018****78836	2020/12/25 19:31:39
41464121	018****78836	2020/12/25 19:30:59
41464121	018****78836	2020/12/25 19:30:02
12648036	018****97878	2020/12/25 19:24:06
7636479	015****02582	2020/12/25 19:08:19
7636479	015****02582	2020/12/25 19:08:05
7636479	015****02582	2020/12/25 19:07:52
7636479	015****02582	2020/12/25 19:07:26
7636479	015****02582	2020/12/25 19:07:03
4937070	013****68165	2020/12/25 17:51:56
41953771	018****02351	2020/12/25 17:45:33
7413039	013****05728	2020/12/25 17:12:38

1	客户ID	积分码	扫的时间	扫码的地理位置
2	41611665	SH020B03Y723EA8C07E4	2020/12/19 23:03:17	山东省济宁市曲阜市
3	41610743	SH020B03Y3CFA9DE352C	2020/12/19 22:23:37	山东省泰安市泰山区
4	41060771	SH020B03Y793F0F1A191	2020/12/19 21:41:38	河南省新乡市凤泉区
5	41060771	SH020B03Y2BBB8F47D3A	2020/12/19 21:37:11	河南省新乡市凤泉区
6	41060771	SH020B03Y6E2F2EC4546	2020/12/19 21:34:48	河南省新乡市凤泉区
7	34480756	SH020B03Y5BD9719D73C	2020/12/19 21:21:37	河南省郑州市中牟县
8	41603246	SH020B03Y2154B67E070	2020/12/19 21:15:37	河北省石家庄市鹿泉区
9	1721670	SH020B03Y3198A8BF684	2020/12/19 20:54:18	山东省聊城市东阿县
10	41600826	SH020B03Y74DFB628459	2020/12/19 20:52:34	河北省邯郸市武安市
11	41600355	SH020B03Y2BC3AB80F07	2020/12/19 20:37:18	山东省聊城市莘县
12	41600082	SH020B03YA76A612B834	2020/12/19 20:27:56	河北省石家庄市鹿泉区
13	41599948	SH020B03Y38DCED7220A	2020/12/19 20:23:13	河北省邢台市柏乡县
14	29862508	SH020B03YDDDF5453699	2020/12/19 20:21:56	河南省濮阳市范县
15	41599651	SH020B03Y1816AA135A4	2020/12/19 20:13:42	山东省济南市平阴县
16	41599125	SH020B03Y07DC809648B	2020/12/19 19:55:05	河南省许昌市禹州市
17	41599109	SH020B03Y415D114431A	2020/12/19 19:54:34	河南省许昌市禹州市
18	34795025	SH020B03YC4E611BA2B5	2020/12/19 19:47:59	山东省潍坊市青州市
19	41598624	SH020B03Y0A2351BEDFA	2020/12/19 19:37:49	河北省邯郸市成安县
20	34795025	SH020B03YF54D86331B3	2020/12/19 19:30:05	山东省潍坊市青州市
21	38768669	SH020B03Y33F127FDC4D	2020/12/19 19:19:36	河南省濮阳市范县
22	41598060	SH020B03Y23A4826721A	2020/12/19 19:18:02	河南省郑州市荥阳市
23	41598060	SH020B03Y34E9E7ECAE6	2020/12/19 19:17:15	河南省郑州市荥阳市
24	41597920	SH020B03Y679146309C5	2020/12/19 19:11:45	山东省德州市陵城区
25	41597222	SH020B03Y26CEB850FBF	2020/12/19 18:54:54	山东省聊城市临清市
26	4666508	SH020B03YAACA815CA2C	2020/12/19 18:35:50	河南省驻马店市上蔡县
27	4856693	SH020B03Y79BEB31C3DD	2020/12/19 18:15:16	河南省新乡市卫滨区
28	4856693	SH020B03Y7B034E26C93	2020/12/19 18:13:37	河南省新乡市卫滨区
29	41459705	SH020B03Y31BBAC28C19	2020/12/19 17:39:30	北京市北京市大兴区

Figure 5.1:

```
    res <- abs(res) %>% which.min() #     res 0
    kong <- data.frame(    = x[i],    = y[res])
    result <- rbind(kong,result)

  }
  return(result)
}
res <- testfun(dt$ ,scan_dt$ )
```

```
testfun <- function(x,y){
  n  <-  length(x)
  result <- list()

  for( i in 1:n){
    y <- y[x>y]
    res <- x[i]-y
    res <- res %>% which.min()
    kong <- data.frame(    = x[i],    = y[res])
    result[[i]] <- kong
  }
  return(result)
}

res <- testfun(dt$ ,scan_dt$ )
```

ID

```
testfun <- function(dt){

  x <- dt$
  y <- dt$
  n  <-  length(x)
  result <- list()

  for( i in 1:n){
    y <- y[x>y]
    res <- x[i]-y
    res <- res %>% which.min()
    kong <- data.frame(   = x[i],    = y[res])
    result[[i]] <- kong
  }
  result <- dplyr::bind_rows(result)
  return(result)
}
dtlist <- split(alldt,' ID')
purrr::map_dfr(dtlist,testfun)
```

5.8

- https://cran.r-project.org/web/packages/lubridate/vignettes/lubridate.html
- https://www.rdocumentation.org/packages/lubridate/versions/1.7.8
- pdf https://rawgit.com/rstudio/cheatsheets/master/lubridate.pdf
- Excle dax https://docs.microsoft.com/en-us/dax/time-intelligence-functions-dax

Chapter 6

forcats

,forcats , tidyverse .

R . . . “ ” “ ” True False .

,R4.0 . https://r4ds.had.co.nz/factors.html

```r
object.size(rep(letters,100000))
object.size(rep(forcats::as_factor(letters),100000))
```

6.1

, , .

```r
library(forcats)
vec1 <- c(' a',' b',' d',' f')
sort(vec1)
vec2 <- as_factor(c(' f',' d',' a',' b'))
sort(vec2)
```

: , , X .

Chapter 7

tidyr

tidyr tidyverse   ,tidyr

-
-
-

7.1

```
##       tidyverse
install.packages('tidyverse')

##      tidyr:
install.packages('tidyr')

##   github
## install.packages("devtools")
devtools::install_github("tidyverse/tidyr")

# CTEST CODE
```

7.2

```
library(tidyr)
```

tidyr 5

- pivot_longer() pivot_wider()
- unnest_longer() unnest_wider(),hoist()
- nest()
- separate(),extract() ,
- replace_na()

7.2.1

vignette("pivot"),

7.2.1.1

EXcel tidyr

Excel ,

col1	col2	col3	col4	col5	col6	col7
v1	v2	v3	v4	v5	v6	v7
vb1	vb2	vb3	vb4	vb5	vb6	vb7

“ ”

```
library(tidyr)
library(dplyr)
library(readr)
```

```
relig_income %>%
  pivot_longer(cols = !religion,names_to = 'income',values_to = "count")
```

-
- religion
- names_to
- values_to

7.2.1.2

```r
billboard %>%
  pivot_longer(
    cols = starts_with("wk"),
    names_to = "week",
    values_to = "rank",
    values_drop_na = TRUE
  )
```

names_prefix names_transform

```r
billboard %>%
  pivot_longer(
    cols = starts_with("wk"),
    names_to = "week",
    names_prefix = "wk",
    names_transform = list(week = as.integer),
    values_to = "rank",
    values_drop_na = TRUE,
  )
```

week

```r
library(tidyverse,warn.conflicts = TRUE)

# method 1
billboard %>%
  pivot_longer(
    cols = starts_with("wk"),
    names_to = "week",
    names_transform = list(week = readr::parse_number),
    values_to = "rank",
    values_drop_na = TRUE,
)

# method 2
billboard %>%
  pivot_longer(
    cols = starts_with("wk"),
    names_to = "week",
    values_to = "rank",
    values_drop_na = TRUE,
  ) %>%
  mutate(week = str_remove(week, "wk") %>% as.integer())
```

7.2.1.3

,new_?(.*)_(.)(.*) new_? new new_ (.*) 0

```
who %>% pivot_longer(
  cols = new_sp_m014:newrel_f65,
  names_to = c("diagnosis", "gender", "age"),
  names_pattern = "new_?(.*)_(.)(.*)",
  values_to = "count"
)
```

gender age

```
who %>% pivot_longer(
  cols = new_sp_m014:newrel_f65,
  names_to = c("diagnosis", "gender", "age"),
  names_pattern = "new_?(.*)_(.)(.*)",
  names_transform = list(
    gender = ~ readr::parse_factor(.x, levels = c("f", "m")),
    age = ~ readr::parse_factor(
      .x,
      levels = c("014", "1524", "2534", "3544", "4554", "5564", "65"),
      ordered = TRUE
    )
  ),
  values_to = "count",
)
```

7.2.1.4

```
family <- tribble(
  ~family, ~dob_child1, ~dob_child2, ~gender_child1, ~gender_child2,
  1L, "1998-11-26", "2000-01-29", 1L, 2L,
  2L, "1996-06-22", NA, 2L, NA,
  3L, "2002-07-11", "2004-04-05", 2L, 2L,
  4L, "2004-10-10", "2009-08-27", 1L, 1L,
  5L, "2000-12-05", "2005-02-28", 2L, 1L,
)
family <- family %>% mutate_at(vars(starts_with("dob")), parse_date)
family
```

```
family %>%
  pivot_longer(
    !family,
    names_to = c(".value", "child"),
    names_sep = "_",
    values_drop_na = TRUE
  )
```

```
anscombe %>%
  pivot_longer(everything(),
    names_to = c(".value", "set"),
    names_pattern = "(.)(.)"
  ) %>%
  arrange(set)
```

```
pnl <- tibble(
  x = 1:4,
  a = c(1, 1,0, 0),
  b = c(0, 1, 1, 1),
  y1 = rnorm(4),
  y2 = rnorm(4),
  z1 = rep(3, 4),
  z2 = rep(-2, 4),
)

pnl %>%
  pivot_longer(
    !c(x, a, b),
    names_to = c(".value", "time"),
    names_pattern = "(.)(.)"
  )
```

7.2.1.5

```
df <- tibble(id = 1:3, y = 4:6, y = 5:7, y = 7:9, .name_repair = "minimal")
df %>% pivot_longer(!id, names_to = "name", values_to = "value")
```

7.2.2

`pivot_wider()` `pivot_longer()` Excel

7.2.2.1

```
fish_encounters %>% pivot_wider(names_from = station, values_from = seen)
```

```
fish_encounters %>% pivot_wider(
  names_from = station,
  values_from = seen,
  values_fill = 0
)
```

7.2.2.2

```
warpbreaks <- warpbreaks %>% as_tibble()
warpbreaks %>% count(wool, tension)
```

values_fn

```
warpbreaks %>% pivot_wider(names_from = wool, values_from = breaks,values_fn= list(brea
```

7.2.2.3

```
production <- expand_grid(
    product = c("A", "B"),
    country = c("AI", "EI"),
    year = 2000:2014
  ) %>%
  filter((product == "A" & country == "AI") | product == "B") %>%
  mutate(production = rnorm(nrow(.)))
production
```

```
production %>% pivot_wider(
  names_from = c(product, country),
  values_from = production
)
```

names_sep names_prefix names_glue

```
production %>% pivot_wider(
  names_from = c(product, country),
  values_from = production,
  names_sep = ".",
  names_prefix = "prod."
)
```

```
production %>% pivot_wider(
  names_from = c(product, country),
  values_from = production,
  names_glue = "prod_{product}_{country}"
)
```

7.2.2.4

```
us_rent_income %>%
  pivot_wider(names_from = variable, values_from = c(estimate, moe))
```

7.2.3 json,html

jsonlite

vignette("rectangle")

```
library(tidyr)
library(dplyr)
library(repurrrsive)
```

```
users <- tibble(user = gh_users)
users
users %>% unnest_wider(user)
```

7.2.4

```
library(tidyr)
library(dplyr)
library(purrr)
```

7.2.4.1

```
df1 <- tibble(
  g = c(1, 2, 3),
  data = list(
    tibble(x = 1, y = 2),
    tibble(x = 4:5, y = 6:7),
    tibble(x = 10)
  )
)
df1
```

data.frame()

```
df2 <- tribble(
  ~g, ~x, ~y,
   1,  1,  2,
   2,  4,  6,
   2,  5,  7,
   3, 10,  NA
)
df2 %>% nest(data = c(x, y))

#sample above
#df2 %>% group_by(g) %>% nest()
```

nest unnest

```
df1 %>% unnest(data)
```

7.2.5

```
mtcars_nested <- mtcars %>%
  group_by(cyl) %>%
  nest()

mtcars_nested
```

```
mtcars_nested <- mtcars_nested %>%
  mutate(model = map(data, function(df) lm(mpg ~ wt, data = df)))
mtcars_nested
```

```
mtcars_nested <- mtcars_nested %>%
  mutate(model = map(model, predict))
mtcars_nested
```

7.2.6

7.2.6.1

```
library(tidyr)
df <- data.frame(x = c(NA, "a.b", "a.d", "b.c"))
df %>% separate(x, c("A", "B"))
```

NA

```
df <- data.frame(x = c("a", "a b", "a b c", NA))
df %>% separate(x, c("a", "b"))
```

```
# The same behaviour as previous, but drops the c without warnings:
df %>% separate(x, c("a", "b"), extra = "drop", fill = "right")
```

```
df %>% separate(x, c("a", "b"), extra = "merge", fill = "left")
```

```
df %>% separate(x, c("a", "b", "c"))
```

```
df %>% separate(x, c("key", "value"), sep = ": ", extra = "merge")
```

```
# Use regular expressions to separate on multiple characters:
df <- data.frame(x = c(NA, "a?b", "a.d", "b:c"))
df %>% separate(x, c("A","B"), sep = "([.?:])")
```

7.2.6.2

```
df <- data.frame(x = c(NA, "a-b", "a-d", "b-c", "d-e"))
df %>% extract(x, "A")
df %>% extract(x, c("A", "B"), "([[:alnum:]]+)-([[:alnum:]]+)")
# [:alnum:]
```

7.2.6.3

```
df <- expand_grid(x = c("a", NA), y = c("b", NA))
df
df %>% unite("z", x:y, remove = FALSE)
# expand_grid
```

```
df %>% unite("z", x:y, na.rm = TRUE, remove = FALSE)
```

```
df %>%
  unite("xy", x:y) %>%
  separate(xy, c("x", "y"))
```

7.2.7

replace_na()

```
df <- tibble(x = c(1, 2, NA), y = c("a", NA, "b"))
df %>% replace_na(list(x = 0, y = "unknown"))
```

```
df %>% dplyr::mutate(x = replace_na(x, 0))
```

Chapter 8

dplyr

R Excel sql R
sql R

-
- R
-
- dbplyr sql

8.1

dplyr tidyverse ,dplyr

- mutate() ,
- select() ,
- filter()
- summarise()
- arrange()

8.2

```
##       tidyverse
install.packages('tidyverse')

##     tidyr:
install.packages('dplyr')

##   github
## install.packages("devtools")
devtools::install_github("tidyverse/dplyr")

# CTEST CODE
```

8.3

```
library(dplyr)
```

8.3.1 filter

-

Excel `species == "Droid"`

```
starwars %>%
  filter(species == "Droid")
```

-

```
starwars %>%
  filter(species == "Droid",skin_color == "gold")

# same above
# starwars %>%
#   filter(species == "Droid" & skin_color == "white")
```

-

`SQL in` Excel “ ”

```
starwars %>%
  filter(species %in%  c("Droid",'Clawdite'))
```

-

| ,&,! , | , & ! ,

```
library(nycflights13)
filter(flights, !(arr_delay > 120 | dep_delay > 120))
filter(flights, arr_delay <= 120, dep_delay <= 120)
# same above
filter(flights, arr_delay <= 120 & dep_delay <= 120)
# %in%
starwars %>%
  filter(!species %in%  c("Droid",'Clawdite'))
```

8.3.2 select

`select()`

-

```
starwars %>%
  select(name,height,mass,hair_color,skin_color,eye_color)
```

-

```
starwars %>%
  select(name : eye_color)
#same above
starwars %>%
  select(1:6)
# starwars %>%
#   select(c(1,2,4,5,7))
```

8.3.3 rename

rename()

```
starwars %>% rename(home_world = homeworld)
#
starwars %>% rename(home_world = homeworld,skincolor = skin_color)
```

8.3.4 relocate

select()

```
# sex:homeworld height
starwars %>% relocate(sex:homeworld, .before = height)
```

8.3.5 mutate

-

```
starwars %>%
  mutate(bmi = mass / ((height / 100)  ^ 2)) %>%
  select(name:mass,bmi)
```

-

```
starwars %>%
  mutate(bmi = mass / ((height / 100)  ^ 2),newbmi = bmi *2) %>%
  select(name:mass,bmi,newbmi)
```

-

```
starwars %>% mutate(height = NULL)
```

8.3.6 arrange

- desc()

```
starwars %>%
  arrange(desc(mass))
```

-

```
starwars %>%
  arrange(height,desc(mass))
```

8.3.7 group_by

group_by() SQL group by ···

8.3.8 summarise

```
starwars %>%
  group_by(species) %>%
  summarise(
    n = n(),
    mass = mean(mass, na.rm = TRUE)
  )
```

8.4

1. sql left join,inner join Excel Power Piovt
2.
3. Excle ,Excle R tidyverse rowwise()

8.4.1

left_join(),full_join,inner_join() vignette("two-table")

left_join() Excel VLOOKUP left join “ ” “ ” “ ”

-

left_join(),right_join(),full_join(),inner_join() ( )

```
library("nycflights13")
# Drop unimportant variables so it's easier to understand the join results.
flights2 <- flights %>% select(year:day, hour, origin, dest, tailnum, carrier)

flights2 %>%
  left_join(airlines)
```

```
on a.column = b.column
```

```
flights2 %>% left_join(planes, by = "tailnum")
```

-

`left_join(x,y,by = c("a" = "b", "c" = "d"))` x*atoy*b x*ctoy*d

```
#
flights2 %>% left_join(airports, by = c("dest" = "faa"))
#flights2 %>% left_join(airports, c("origin" = "faa"))
#           c("dest" = "faa","cola" = "colb"))
```

-

`anti_join()`

`semi_join()`

```
df1 <- tibble(a=letters[1:20],b=1:20)
df2 <- tibble(a=letters,b=1:26)

df1 %>% semi_join(df2)
df2 %>% anti_join(df1)
```

-

1. `intersect(x,y)` x,y
2. `union(x,y)` x,y
3. `setdiff(x,y)` x y

```
(df1 <- tibble(x = 1:2, y = c(1L, 1L)))
(df2 <- tibble(x = 1:2, y = 1:2))
intersect(df1, df2)
union(df1, df2)
setdiff(df1, df2)
setdiff(df2, df1)
```

8.4.2

purrr::reduce(),

```
dt1 <- data.frame(x = letters)
dt2 <- data.frame(x = letters,cola = 1:26)
dt3 <- data.frame(x = letters,colb = 1:26)
dt4 <- data.frame(x = letters,cold = 1:26)
dt5 <- data.frame(x = letters,cole = 1:26)

dtlist <- list(dt1,dt2,dt3,dt4,dt5)
purrr::reduce(dtlist,left_join,by='x')
```

8.5

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

across()

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

8.5.1

across()

- .cols tidyr select()
- .fns purrr ~ .x / 2

```
starwars %>%
  summarise(across(where(is.character), ~ length(unique(.x))))

#
# starwars %>%
#   summarise(length(unique(name)))
# starwars %>%
#   summarise(length(unique(hair_color)))

starwars %>%
  group_by(species) %>%
  filter(n() > 1) %>%
  summarise(across(c(sex, gender, homeworld), ~ length(unique(.x))))

starwars %>%
  group_by(homeworld) %>%
  filter(n() > 1) %>%
  summarise(across(where(is.numeric), ~ mean(.x, na.rm = TRUE)))
```

across()

```
df <- data.frame(g = c(1, 1, 2), x = c(-1, 1, 3), y = c(-1, -4, -9))
df %>%
  group_by(g) %>%
  summarise(across(where(is.numeric), sum))
```

8.5.2

lambda

```
min_max <- list(
  min = ~min(.x, na.rm = TRUE),
  max = ~max(.x, na.rm = TRUE)
)
starwars %>% summarise(across(where(is.numeric), min_max))
```

.names

NB:

```
starwars %>% summarise(across(where(is.numeric), min_max, .names = "{.fn}.{.col}"))
```

```
starwars %>% summarise(across(where(is.numeric), min_max, .names = "{fn}.{col}"))
```

.names .

```
starwars %>% summarise(across(where(is.numeric), min_max, .names = "{fn}--{col}"))
```

8.5.3

“ ” cur_column()

```
df <- tibble(x = 1:3, y = 3:5, z = 5:7)
mult <- list(x = 1, y = 10, z = 100)

df %>% mutate(across(all_of(names(mult)), ~ .x * mult[[cur_column()]]))
```

8.6

dplyr

8.6.1

```
df <- tibble(x = 1:2, y = 3:4, z = 5:6)
df %>% rowwise()
```

group_by(),rowwise()

```
df %>% mutate(m = mean(c(x, y, z)))
df %>% rowwise() %>% mutate(m = mean(c(x, y, z)))
```

data.table :

```
library(data.table)

dt <- data.table(x = 1:2, y = 3:4, z = 5:6)
dt[,m:=mean(c(x,y,z))][]
dt[,m:=mean(c(x,y,z)),by=.(x)][]
```

“ ” rowwise() summarise() group_by()

```
df <- tibble(name = c("Mara", "Hadley"), x = 1:2, y = 3:4, z = 5:6)

df %>%
  rowwise() %>%
  summarise(m = mean(c(x, y, z)))

df %>%
  rowwise(name) %>%
  summarise(m = mean(c(x, y, z)))
```

8.6.2

dplyr::summarise() rowwise()

```
df <- tibble(id = 1:6, w = 10:15, x = 20:25, y = 30:35, z = 40:45)
rf <- df %>% rowwise(id)
rf %>% mutate(total = sum(c(w, x, y, z)))
rf %>% summarise(total = sum(c(w, x, y, z)))
```

c_across()

```
rf %>% mutate(total = sum(c_across(w:z)))
rf %>% mutate(total = sum(c_across(where(is.numeric))))

rf %>%
  mutate(total = sum(c_across(w:z))) %>%
  ungroup() %>%
  mutate(across(w:z, ~ . / total))
```

8.7

: https://cloud.r-project.org/web/packages/dplyr/vignettes/grouping.html

group_by() ,

8.7.1

```
by_species <- starwars %>% group_by(species)
by_sex_gender <- starwars %>% group_by(sex, gender)
```

mutate() group_by:

```
bmi_breaks <- c(0, 18.5, 25, 30, Inf)
starwars %>%
  group_by(bmi_cat = cut(mass/(height/100)^2, breaks=bmi_breaks)) %>%
  tally()
```

8.7.2

ungroup():

```
by_species %>%
  ungroup() %>%
  tally()
```

8.7.3

summarise() group_keys

```
by_species %>%
  summarise(
    n = n(),
    height = mean(height, na.rm = TRUE)
  )
```

.groups= .groups = “drop_last” .groups = NULL

1.0.0 (.groups = "keep") (.groups = 'drop)

```
a <- by_species %>%
  summarise(
    n = n(),
    height = mean(height, na.rm = TRUE),.groups='drop') %>%
  group_vars()

b <- by_species %>%
  summarise(
    n = n(),
    height = mean(height, na.rm = TRUE),.groups='keep') %>%
  group_vars()

object.size(a)
object.size(b)
```

8.8

8.8.1

base::ifelse,if_else TRUE FALSE data.table::fifelse()

```
if_else(condition, true, false, missing = NULL)
```

ifelse if_else

```
x <- factor(sample(letters[1:5], 10, replace = TRUE))
ifelse(x %in% c("a", "b", "c"), x, factor(NA))
if_else(x %in% c("a", "b", "c"), x, factor(NA))
```

8.8.2 case_when

case_when, sql case when

```
Dates <- as.Date(c('2018-10-01', '2018-10-02', '2018-10-03'))
case_when(
  Dates == '2018-10-01' ~ Dates - 1,
  Dates == '2018-10-02' ~ Dates + 1,
  Dates == '2018-10-03' ~ Dates + 2,
  TRUE ~ Dates
)
```

8.8.3

-

count()

```
df %>% count(a, b)
# same above
df %>% group_by(a, b) %>% summarise(n = n())
```

```
starwars %>% count(species)
# same above
starwars %>% group_by(species) %>% summarise(n = n())
```

-

n_distinct() length(unique(x))

```
x <- sample(1:10, 1e5, rep = TRUE)
length(unique(x))
n_distinct(x)
```

8.8.4

dplyr SQL2003

- row_number(): rank(ties.method = “first”)
- min_rank(): rank(ties.method = “min”)
- dense_rank(): min_rank() ,
- percent_rank(): 0 1 min_rank() [0,1]

```
x <- c(5, 1, 3, 2, 2, NA)
row_number(x)
min_rank(x)
dense_rank(x)
percent_rank(x)
cume_dist(x)
```

8.8.5

[[

```
nth(x, n, order_by = NULL, default = default_missing(x))
first(x, order_by = NULL, default = default_missing(x))
last(x, order_by = NULL, default = default_missing(x))
```

```
x <- 1:10
y <- 10:1
first(x)
last(y)
nth(x, 1)
nth(x, 5)
```

8.8.6 group

group_by(),group_map(), group_nest(), group_split(), group_trim()

group_by(),group_split() group_by() group_by()

- group_by()

```
#group_by()
by_cyl <- mtcars %>% group_by(cyl)
by_cyl
# It changes how it acts with the other dplyr verbs:
by_cyl %>% summarise(
  disp = mean(disp),
  hp = mean(hp)
)
# group_by       mutate
mtcars %>% group_by(vsam = vs + am) %>%
  group_vars()
```

- group_map()

group_map group_modify,group_walk   purrr

```
# return a list
#
mtcars %>%
  group_by(cyl) %>%
  group_map(~ head(.x, 2L))
```

```
iris %>%
  group_by(Species) %>%
  group_modify(~ {
    .x %>%
      purrr::map_dfc(fivenum) %>%
      mutate(nms = c("min", "Q1", "median", "Q3", "max"))
  })
```

```
# group_walk
dir.create(temp <- tempfile())
iris %>%
  group_by(Species) %>%
  group_walk(~ write.csv(.x, file = file.path(temp, paste0(.y$Species, ".csv"))))
list.files(temp, pattern = "csv$")
unlink(temp, recursive = TRUE)
```

- group_cols()

```
gdf <- iris %>% group_by(Species)
gdf %>% select(group_cols())
```

8.8.7

- between
- cummean cumsum cumall cumany

```
x <- c(1, 3, 5, 2, 2)
cummean(x)
cumsum(x) / seq_along(x)

cumall(x < 5)
cumany(x == 3)
```

- distinct

```
df <- tibble(
  x = sample(10, 100, rep = TRUE),
  y = sample(10, 100, rep = TRUE)
)

distinct(df, x)
distinct(df, x, .keep_all = TRUE)
distinct(df, diff = abs(x - y))
```

8.9 dplyr

Programming with dplyr:

https://cloud.r-project.org/web/packages/dplyr/vignettes/programming.html

- When you have the data-variable in a function argument (i.e. an env-variable that holds a promise2), you need to ** embrace ** the argument by surrounding it in doubled braces, like `filter(df, {{ var }})`.

The following function uses embracing to create a wrapper around `summarise()` that computes the minimum and maximum values of a variable, as well as the number of observations that were summarised:

```
var_summary <- function(data, var) {
  data %>%
    summarise(n = n(), min = min({{ var }}), max = max({{ var }}))
}
mtcars %>%
  group_by(cyl) %>%
  var_summary(mpg)
```

- When you have an env-variable that is a character vector, you need to index into the .data pronoun with [[, like summarise(df, mean = mean(.data[[var]])).

The following example uses .data to count the number of unique values in each variable of mtcars:

```
for (var in names(mtcars)) {
  mtcars %>% count(.data[[var]]) %>% print()
}
```

Note that .data is not a data frame; it's a special construct, a pronoun, that allows you to access the current variables either directly, with `.data$x` or indirectly with `.data[[var]]`. Don't expect other functions to work with it.

8.9.1

```
my_summarise <- function(data, group_var) {
  data %>%
    group_by({{ group_var }}) %>%
    summarise(mean = mean(mass))
}
```

```
my_summarise2 <- function(data, expr) {
  data %>% summarise(
    mean = mean({{ expr }}),
    sum = sum({{ expr }}),
    n = n()
  )
}
```

```
my_summarise3 <- function(data, mean_var, sd_var) {
  data %>%
    summarise(mean = mean({{ mean_var }}), sd = mean({{ sd_var }}))
}
```

```
my_summarise4 <- function(data, expr) {
  data %>% summarise(
    "mean_{{expr}}" := mean({{ expr }}),
    "sum_{{expr}}" := sum({{ expr }}),
    "n_{{expr}}" := n()
  )
}
my_summarise5 <- function(data, mean_var, sd_var) {
  data %>%
    summarise(
      "mean_{{mean_var}}" := mean({{ mean_var }}),
      "sd_{{sd_var}}" := mean({{ sd_var }})
    )
}
```

```
my_summarise <- function(.data, ...) {
  .data %>%
    group_by(...) %>%
    summarise(mass = mean(mass, na.rm = TRUE), height = mean(height, na.rm = TRUE))
}
starwars %>% my_summarise(homeworld)
starwars %>% my_summarise(sex, gender)
```

Chapter 9

Loop structure

, ,

9.1

1 100

```
total <- 0
for(i in 1:100){
  total <- total+i
}
print(paste0('1 100    :',total))

# loop structure
# for (var in seq) {expr}
```

9.2

9.2.1

R

- Repeat

```
i <- 1
total <- 0
repeat{
  total <- total+i
  i <- i+1
  if(i > 100){
    print(paste0('     :',total))
    break
  }
}
```

- while

```
i <- 1
total <- 0
while(i <= 1000){
  total <- total+i
  i <- i+1
}
print(paste0('1 1000     :',total))
# not run
# sum(1:1000)
```

- for

```
library(tidyverse)
df <- tibble(
  a = rnorm(10),
  b = rnorm(10),
  c = rnorm(10),
  d = rnorm(10)
)

output <- vector("double", ncol(df))  # 1. output
for (i in seq_along(df)) {            # 2. sequence
  output[[i]] <- median(df[[i]])      # 3. body
}
output
```

R , output , , .

vector (‘ ’,‘ ’,‘ ’,’ ’) vector(length=5), .

`seq_along` `?seq` .

hadely :

You might not have seen seq_along() before. It's a safe version of the familiar 1:length(l), with an important difference: if you have a zero-length vector, seq_along() does the right thing:

```
#wrong
seq_along(c())
1:length(c())

# generates the integer sequence 1, 2, ..., length(along.with). (along.with is usually abbreviate
```

9.2.2 next break

- next

```
for(i in letters[1:6] ){
  if(i == "d"){
  next
  }
  print(i)
}
```

- break

repeat

9.2.3

```
# not run
v <- vector(length = 100)
for(i in 1:10){
  for(j in 1:10){
    v[i*j] = i * j
  }
}
```

9.3

9.3.1

```
res <- 1:100
for(i in seq_along(res)){
  res[i] <- res[i] * i
}
str(res)
```

9.3.2

, for (i in seq_along(xs)), x[[i]]. :

-

for(i in xs), ,

-

for (nm in names(xs)), x[[nm]] . .

```
results <- vector("list", length(x))
names(results) <- names(x)
```

, .

```
for (i in seq_along(x)) {
  name <- names(x)[[i]]
  value <- x[[i]]
}
```

9.3.3

. ,

```
means <- c(0, 1, 2)

output <- double()
for (i in seq_along(means)) {
  n <- sample(100, 1)
  output <- c(output, rnorm(n, means[[i]]))
}
str(output)
```

. $(O(n^2))$. , :

```
out <- vector("list", length(means))
for (i in seq_along(means)) {
  n <- sample(100, 1)
  out[[i]] <- rnorm(n, means[[i]])
}
str(out)
str(unlist(out)) #unlist
```

Chapter 10

Iteration

purrr ,

https://purrr.tidyverse.org/

10.1

- map

map ,map list

```r
library(tidyverse)

# define function
addTen <- function(.x) {
  return(.x + 10)
}

map(.x = c(1, 4, 7), .f = addTen)
# not run
# map(c(1, 4, 7), addTen) # same above
```

- map_dbl

map_dbl map_dbl vector

```
#library(purrr)
add1 <- function(x) {
  (x+1)*x
}
result1 <- map_dbl(1:1000,add1) # maP_dbl

#for
result2 <- vector(length = 1000)
for(i in 1:1000){
  result2[i] <- (i+1) * i
}
# test
#not run
#table(result1 == result2)
# all equal
identical(result1,result2)
```

10.2 map

- map_chr

`map_chr(.x, .f)` ,map_chr

- map_dbl

`map_dbl(.x, .f)` ,map_dbl ( )

- map_df

`map_df(.x, .f)`,map_df , `map_dfr(.x,.f)`,`map_dfc(.x,.f)`

- map_gl

`map_lgl(.x, .f)`

- map_int

`map_int(.x, .f, ...)`

map_df()

```
#
map_df(c(1, 4, 7), function(.x) {
  return(data.frame(old_number = .x,
                    new_number = addTen(.x)))
})

#
#step1
make_dataframe <- function(x){
  data.frame(old_number = x,new_number = addTen(x))
}
#step2
map_df(c(1,4,7),make_dataframe)
```

10.3

reduce accumulate() .

- reduce

, reduce merge()

```
reduce(1:100,`+`)
reduce(100:1,`-`)
```

list

```
n <- 10
dt1 <- data.frame(a=letters[n],b1=rnorm(n))
dt2 <- data.frame(a=letters[n],b2=rnorm(n))
dt3 <- data.frame(a=letters[n],b3=rnorm(n))
dt4 <- data.frame(a=letters[n],b4=rnorm(n))

reduce(list(dt1,dt2,dt3,dt4),merge)
# not run
# reduce(list(dt1,dt2,dt3,dt4),merge,by='a') same above
```

- accumulate

```
1:5 %>% accumulate(`+`)
accumulate(letters[1:5], paste, sep = ".")
```

10.4

possibly() safely(), ,

```
l <- list(1,2,3,4,'5')
map(l,function(.x) .x+1)
```

,

```
l <- list(1,2,3,4,'5')
test_fun <- safely(function(.x) .x+1)
map(l,test_fun)
```

safely() function , , , ,

10.5

map2 pmap

```
li1 <- list(1,3,5)
li2 <- list(2,4,6)
map2(li1,li2,`+`)
```

map2_dbl,map2_chr,map2_dfr

```
li1 <- list(1,3,5)
li2 <- list(2,4,6)
li3 <- list(2,4,6)
li1 <- c(1,3,5)
li2 <- c(2,4,6)
li3 <- c(2,3,4)
li <- list(li1,li2,li3)
pmap(li,sum)
```

pmap_int,pmap_dbl,pmap_dfr

10.6

- flatten

flatten() purrr package Examples

```
x <- rerun(2, sample(4))
x
x %>% flatten()
x %>% flatten_int()
# You can use flatten in conjunction with map
x %>% map(1L) %>% flatten_int()
# But it's more efficient to use the typed map instead.
x %>% map_int(1L)
```

- imap

imap()

imap_xxx(x, …), an indexed map, is short hand for map2(x, names(x), …) if x has names, or map2(x, seq_along(x), …) if it does not. This is useful if you need to compute on both the value and the position of an element.

imap, x names(x) seq_along(x) ,imap map2

, (.x), / (.y)

:?imap

1

```
imap_chr(sample(10), ~ paste0(.y, ": ", .x))
```

sample(10), names(), map2 :

```
#same above

map2_chr(sample(10),1:10,~paste0(.y,": ",.x)) #   list    .
```

Chapter 11

define function

, , .

SKU , 5 , .

:

```
library(tidyverse)
num <- sample(1:1000,1000)
res1 <- if_else(num <= 50,"1-50",
                if_else(num <= 100,"51-100",
                        if_else(num <= 150,"101-150",
                                if_else(num <= 200 ,"151-200",
                                        if_else(num >200,"200 ",'  ')))))



# same above
# case_when(num <= 50 ~ '1-50',
#           num <= 100 ~ '51-100',
#           num <= 150 ~ '101-150',
#           num <= 200 ~ '151-200',
#           num > 100 ~ '200 '
#           )

#   data.table
# data.table::fifelse()
# data.table::fcase()  sql case when
```

:

, , , .

```
#
#library(tidyverse)
cut_function <- function(vecto,x,n){
  vec <- c(0)
  for(i in 1:n){
    kong <-  i*x
    vec <- c(vec,kong)
  }
  vec <- c(vec,Inf)
  labels <- c()
  j <- 1

  while (j<=n) {
    labels[j] <- str_c(vec[j]+1,"-",vec[j+1])
    j <- j+1
  }
  labels <- c(labels,paste0(vec[j],' '))
  res <- cut(x = vecto,breaks = vec,labels = labels) %>% as.character()
}

res2 <- cut_function(num,50,4)

# identical(res1,res2)
# > TRUE
```

11.1

, .

```
add_ten <- function(x){
  res <- x+10
  return(res) #
}
add_ten(1)
```

, , .

```
add_ten <- function(x){
  if(is.numeric(x)==TRUE){
    x+10
  } else {
    print('Error,   ')
  }
}
```

11.2

```
has_name <- function(x) {
  nms <- names(x)
  if (is.null(nms)) {
    rep(FALSE, length(x))
  } else {
    !is.na(nms) & nms != ""
  }
}
```

11.2.1

```
if (this) {
  # do that
} else if (that) {
  # do something else
} else {
  #
}
```

if switch()

```
function(x, y, op) {
   switch(op,
     plus = x + y,
     minus = x - y,
     times = x * y,
     divide = x / y,
     stop("Unknown op!")
   )
 }
```

11.3

, , .

```
mean_ci <- function(x, conf = 0.95) {
  se <- sd(x) / sqrt(length(x))
  alpha <- 1 - conf
  mean(x) + se * qnorm(c(alpha / 2, 1 - alpha / 2))
}
x <- runif(100)
mean_ci(x)
mean_ci(x, conf = 0.99)
```

11.3.1

, , .

- x, y, z: vectors.
- w: a vector of weights.
- df: a data frame.
- i, j: numeric indices (typically rows and columns).
- n: length, or number of rows.
- p: number of columns.

11.3.2

, , .

```
wt_mean <- function(x, w) {
  if (length(x) != length(w)) {
    stop("`x` and `w` must be the same length", call. = FALSE)
  }
  sum(w * x) / sum(w)
}
```

11.3.3 ...

R

```
sum(1,2,3,4,5,6,7,8,9,10)
stringr::str_c('a','b','d','e','f','g','h')
```

```
commas <- function(...) stringr::str_c(..., collapse = ", ")
commas(letters[1:10])
#> [1] "a, b, c, d, e, f, g, h, i, j"

rule <- function(..., pad = "-") {
  title <- paste0(...)
  width <- getOption("width") - nchar(title) - 5
  cat(title, " ", stringr::str_dup(pad, width), "\n", sep = "")
}
rule("Important output")
```

11.4

11.4.1

, return() R for Data Science : ‘ return() , ’

- A common reason to do this is because the inputs are empty:

```
complicated_function <- function(x, y, z) {
  if (length(x) == 0 || length(y) == 0) {
    return(0)
  }
  # Complicated code here
}
```

- Another reason is because you have a if statement with one complex block and one simple block. For example, you might write an if statement like this:

```
f <- function() {
  if (x) {
    # Do
    # something
    # that
    # takes
    # many
    # lines
    # to
    # express
  } else {
    # return something short
  }
}
```

11.4.2

: transformations and side-effects transformations side-effects , “ ”

R for Data Science

```
show_missings <- function(df) {
  n <- sum(is.na(df))
  cat("Missing values: ", n, "\n", sep = "")

  invisible(df)
}
```

invisible() , df :

```
show_missings(mtcars)
```

:

```
x <- show_missings(mtcars)
class(x)
dim(x)
```

```
mtcars %>%
  show_missings() %>%
  mutate(mpg = ifelse(mpg < 20, NA, mpg)) %>%
  show_missings()
```

11.5

, .

The last component of a function is its environment. This is not something you need to understand deeply when you first start writing functions. However, it’s important to know a little bit about environments because they are crucial to how functions work. The environment of a function controls how R finds the value associated with a name. For example, take this function:

```
f <- function(x) {
  x + y
}
```

y. R , , R lexical scoping . y, y:

```
y <- 100
f(10)

y <- 1000
f(10)
```

https://r4ds.had.co.nz/functions.html#environment

http://adv-r.had.co.nz/

11.6

, , group_by() ,

https://dplyr.tidyverse.org/articles/programming.html

```
#library(tidyverse)
mean_mpg = function(data, group_col) {
  data %>%
    group_by(group_col) %>%
    summarize(mean_mpg = mean(mpg))
}
mtcars %>% mean_mpg(cyl)
mtcars %>% mean_mpg(gear)
```

,

```
#
my_summarise3 <- function(data, group_var,mean_var, sd_var) {
  data %>%
    group_by({{ group_var }}) %>%
    summarise(mean = mean({{ mean_var }}), sd = mean({{ sd_var }}))
}

res1 <- my_summarise3(data = mtcars,group_var = cyl,mean_var = carb,sd_var = gear)
my_summarise3(data = mtcars,group_var = am,mean_var = carb,sd_var = gear)
```

```
#
res2 <- mtcars %>%
  group_by(cyl) %>%
  summarise(mean=mean(carb),sd=mean(gear))

identical(res1,res2)

#res1  res2
```

my_summarise3()