tidyverse优雅的处理数据

R数据分析指南-闫俊安

2021-10-19

目录

引言

本文档主要介绍tidyverse体系在数据分析中的一些常见的使用方法,使用tidyverse可以很轻松的完成从数据清洗到数据可视化一整套的分析流程，即数据导入、格式转化、数据清洗、数据可视化、统计建模以及使用Rmarkdown生成分析报告，整个数据分析及可视乎都在一个程序代码中完成，这种方式使的数据分析过程变得异常简洁

> 欢迎扫描下方二维码关注我的公众号，下回更新不迷路

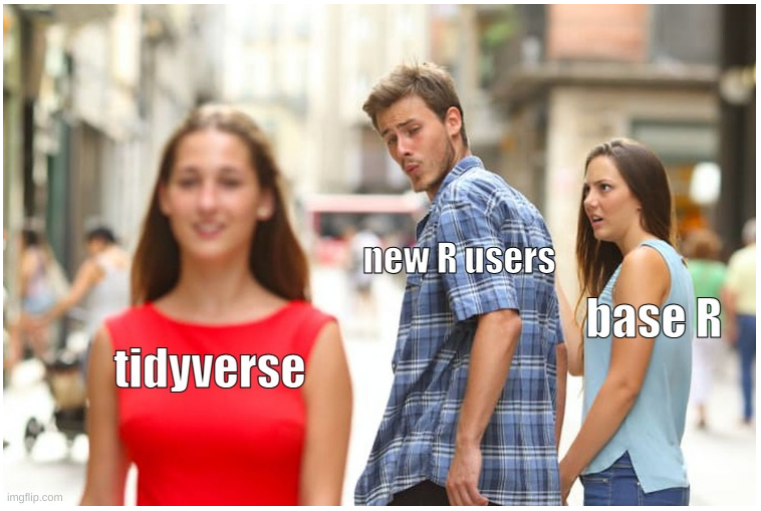
tidyverse家族

tidyverse家族核心成员

功能	R包
数据可视化	ggplot2
数据处理	dplyr
格式转换	tidyr
数据加载	readr
循环加速	purrr
数据框强化	tibble
字符串处理	stringr
因子处理	forcats

数据处理

现在我们正式开始数据处理的历程,主要介绍tidyverse中的一些常用函数,并通过一个个小案例来进行系统性的展示



1. 宽表转长表

1.1 pivot_longer函数

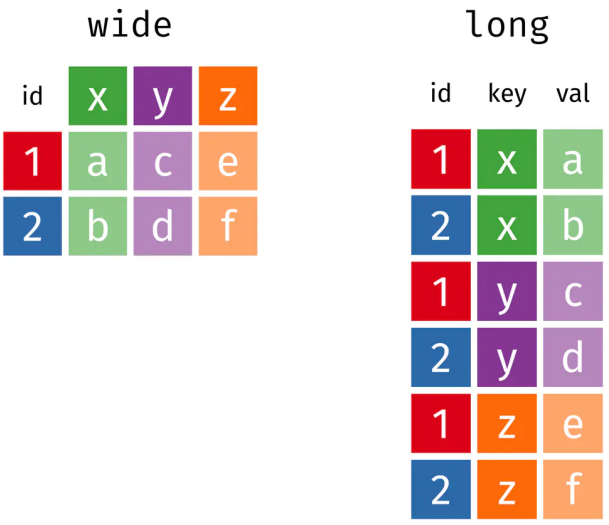
如果您是R语言的初学者,在开始进行数据格式转换前请先阅读以下2篇文档

配置R与Rstudio

ggplot2中的一些关键概念

通过上面2篇文档的学习,相信您已经了解了如何安装`R`包,设置路径以及对`ggplot2`的一些关键概念有了了解,下面就该读入自己的数据来进行可视化了;那么在画图之前我们还需要对数据来进行格式处理,是否还记得在`ggplot2`进行可视化时需要给`aes`传入2个美学参数,一个作为X另一个为Y,但是我们通常手里边的数据有很多列,在R中将其定义为宽表,而`ggplot`函数需要的数据格式在R中我们定义为长表,因此第一步即宽表转长表

如下图所示



下面我们创建数据集来演示一下

```
df <- data.frame(state = c("Maine", "Massachusetts",
                           "New Hampshire", "Vermont"),
                 male_fulltime = c(50329,66066, 59962, 50530),
                 male_other = c(18099, 18574, 20274, 17709),
                 female_fulltime = c(40054, 53841, 46178, 42198),
                 female_other = c(13781, 14981, 15121, 14422))

df
```

```
##           state male_fulltime male_other female_fulltime female_other
## 1         Maine         50329      18099           40054        13781
## 2 Massachusetts         66066      18574           53841        14981
## 3 New Hampshire         59962      20274           46178        15121
## 4       Vermont         50530      17709           42198        14422
```

```
df2.long <- pivot_longer(df,cols = -state,
                         names_to = c("sex","work"),
                         names_sep = "_",
                         values_to = "income")
df2.long %>% head()
```

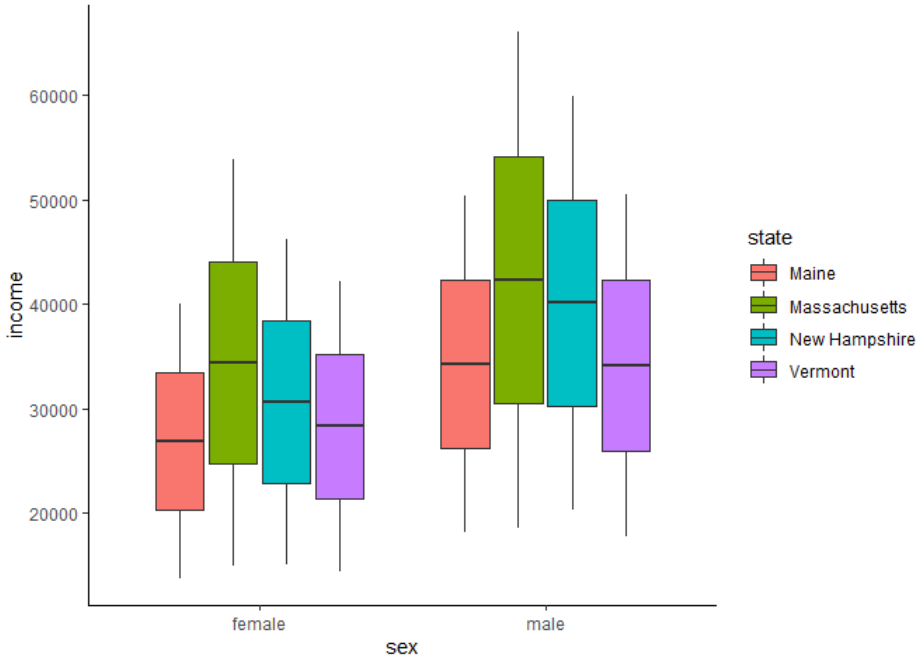
```
## # A tibble: 6 x 4
##   state         sex    work     income
##   <chr>         <chr>  <chr>     <dbl>
## 1 Maine         male   fulltime  50329
## 2 Maine         male   other     18099
## 3 Maine         female fulltime  40054
## 4 Maine         female other     13781
## 5 Massachusetts male   fulltime  66066
## 6 Massachusetts male   other     18574
```

可以看到通过pivot_longer( )函数很轻松完成了宽表转长表

- pivot_longer()函数有三个主要的参数
- cols,表示哪些列需要转换
- names_to,表示cols选取的这些列的名字,构成了新的一列或多列;需要给定名称
- values_to,表示cols选取的这些列的值，构成了新的一列，同样要给定名称

经过上面的格式转换我们得到了需要的数据,接下来进行一个简单的可视化

```
df2.long %>% ggplot(aes(sex,income,fill=state))+
  geom_boxplot()+theme_classic()
```



在某些时刻我们可能还需要进行长表转宽表,让我们通过下面的代码来学习

2. 长表转宽表

2.1 pivot_wider函数

```
pivot_wider(df2.long,
            names_from = c(sex,work),
            values_from = income,
            names_sep = ".")
```

```
## # A tibble: 4 x 5
##   state         male.fulltime male.other female.fulltime female.other
##   <chr>                 <dbl>      <dbl>           <dbl>        <dbl>
## 1 Maine                 50329      18099           40054        13781
## 2 Massachusetts         66066      18574           53841        14981
## 3 New Hampshire         59962      20274           46178        15121
## 4 Vermont               50530      17709           42198        14422
```

3. select选择列的方法

3.1 select选择列基础

要选择几列，只需在select函数中添加其名称即可。添加它们的顺序将确定它们在输出中出现的顺序

```
msleep %>% select(name, genus, sleep_total, awake) %>% head()
```

```
## # A tibble: 6 x 4
##   name                       genus      sleep_total awake
##   <chr>                      <chr>            <dbl> <dbl>
## 1 Cheetah                    Acinonyx          12.1  11.9
## 2 Owl monkey                 Aotus             17     7
## 3 Mountain beaver            Aplodontia        14.4   9.6
## 4 Greater short-tailed shrew Blarina           14.9   9.1
## 5 Cow                        Bos                4    20
## 6 Three-toed sloth           Bradypus          14.4   9.6
```

如果要添加大量的列，可以使用start_col:end_col的语句：

```
msleep %>% select(name:order,sleep_cycle:brainwt) %>% head()
```

```
## # A tibble: 6 x 7
##   name                       genus      vore  order   sleep_cycle awake  brainwt
##   <chr>                      <chr>      <chr> <chr>         <dbl> <dbl>    <dbl>
## 1 Cheetah                    Acinonyx   carni Carniv~      NA      11.9 NA
## 2 Owl monkey                 Aotus      omni  Primat~      NA       7    0.0155
## 3 Mountain beaver            Aplodontia herbi Rodent~      NA       9.6 NA
## 4 Greater short-tailed shrew Blarina    omni  Sorico~       0.133   9.1  0.00029
## 5 Cow                        Bos        herbi Artiod~       0.667  20    0.423
## 6 Three-toed sloth           Bradypus   herbi Pilosa        0.767   9.6 NA
```

还可以通过在列名称前面添加减号来取消列

```
msleep %>% select(-conservation, -(sleep_total:awake)) %>% head()
```

```
## # A tibble: 6 x 6
##   name                       genus      vore  order         brainwt  bodywt
##   <chr>                      <chr>      <chr> <chr>           <dbl>   <dbl>
## 1 Cheetah                    Acinonyx   carni Carnivora    NA        50
## 2 Owl monkey                 Aotus      omni  Primates      0.0155    0.48
## 3 Mountain beaver            Aplodontia herbi Rodentia     NA         1.35
```

```
## 4 Greater short-tailed shrew Blarina    omni  Soricomorpha  0.00029   0.019
## 5 Cow                       Bos        herbi Artiodactyla  0.423   600
## 6 Three-toed sloth          Bradypus   herbi Pilosa       NA         3.85
```

3.2 根据部分列名选择列

如果有很多的列具有相似的结构,可以通过starts_with(),ends_with()或contains( )来进行选择

```
msleep %>% select(name, starts_with("sleep")) %>% head()
```

```
## # A tibble: 6 x 4
##   name                        sleep_total sleep_rem sleep_cycle
##   <chr>                             <dbl>     <dbl>       <dbl>
## 1 Cheetah                            12.1      NA         NA
## 2 Owl monkey                         17         1.8       NA
## 3 Mountain beaver                    14.4       2.4       NA
## 4 Greater short-tailed shrew         14.9       2.3        0.133
## 5 Cow                                 4         0.7        0.667
## 6 Three-toed sloth                   14.4       2.2        0.767
```

```
msleep %>% select(contains("eep"), ends_with("wt")) %>% head()
```

```
## # A tibble: 6 x 5
##   sleep_total sleep_rem sleep_cycle  brainwt  bodywt
##         <dbl>     <dbl>       <dbl>    <dbl>   <dbl>
## 1        12.1      NA         NA     NA        50
## 2        17         1.8       NA      0.0155    0.48
## 3        14.4       2.4       NA     NA         1.35
## 4        14.9       2.3        0.133  0.00029   0.019
## 5         4         0.7        0.667  0.423   600
## 6        14.4       2.2        0.767 NA         3.85
```

3.3 根据正则表达式选择列

如果列名没有相似性,则可以使用matches()来进行选择;以下示例代码将添加包含“o”后跟一个或多个其他字母和“er”的列

```
msleep %>% select(matches("o.+er")) %>% head()
```

```
## # A tibble: 6 x 2
##   order        conservation
##   <chr>        <chr>
## 1 Carnivora    lc
## 2 Primates     <NA>
## 3 Rodentia     nt
## 4 Soricomorpha lc
## 5 Artiodactyla domesticated
## 6 Pilosa       <NA>
```

3.4 根据数据集来选择列

```
class <- c("name", "genus", "vore", "order", "conservation")

msleep %>% select(!!class) %>% head()
```

```
## # A tibble: 6 x 5
##   name                        genus      vore  order        conservation
```

```
##   <chr>                      <chr>      <chr> <chr>        <chr>
## 1 Cheetah                    Acinonyx   carni Carnivora    lc
## 2 Owl monkey                 Aotus      omni  Primates     <NA>
## 3 Mountain beaver            Aplodontia herbi Rodentia     nt
## 4 Greater short-tailed shrew Blarina    omni  Soricomorpha lc
## 5 Cow                        Bos        herbi Artiodactyla domesticated
## 6 Three-toed sloth           Bradypus   herbi Pilosa       <NA>
```

3.5 按数据类型选择列 (重点)

select_if( )函数来判断数据类型，可以使用其来选择所有字符串列select_if(is.character)。同样也可以添加is.numeric,is.integer,is.double,is.logical,is.factor等列；如果有日期列,则可以加载lubridate包,然后使用 is.POSIXt或is.Date来进行格式转换

```
msleep %>% select_if(is.numeric) %>% head()
```

```
## # A tibble: 6 x 6
##   sleep_total sleep_rem sleep_cycle awake  brainwt  bodywt
##         <dbl>     <dbl>       <dbl> <dbl>    <dbl>   <dbl>
## 1        12.1      NA        NA      11.9 NA        50
## 2        17         1.8      NA       7    0.0155    0.48
## 3        14.4       2.4      NA       9.6 NA         1.35
## 4        14.9       2.3       0.133   9.1  0.00029   0.019
## 5         4         0.7       0.667  20    0.423   600
## 6        14.4       2.2       0.767   9.6 NA         3.85
```

同样也可以取反，选择不需要那种数据类型的列

```
msleep %>% select_if(~!is.numeric(.)) %>% head()
```

```
## # A tibble: 6 x 5
##   name                       genus      vore  order        conservation
##   <chr>                      <chr>      <chr> <chr>        <chr>
## 1 Cheetah                    Acinonyx   carni Carnivora    lc
## 2 Owl monkey                 Aotus      omni  Primates     <NA>
## 3 Mountain beaver            Aplodontia herbi Rodentia     nt
## 4 Greater short-tailed shrew Blarina    omni  Soricomorpha lc
## 5 Cow                        Bos        herbi Artiodactyla domesticated
## 6 Three-toed sloth           Bradypus   herbi Pilosa       <NA>
```

3.6 通过逻辑表达式选择列 (重点)

select_if( )不仅仅是基于数据类型来进行选择。还可以选择所有列平均值大于10的列。mean > 10它本身不是函数，因此需要在前面添加波浪号，或使用funs()将语句转换为函数

```
msleep %>% select_if(is.numeric) %>%
  select_if(~mean(., na.rm=TRUE) > 10) %>% head()
```

```
## # A tibble: 6 x 3
##   sleep_total awake  bodywt
##         <dbl> <dbl>   <dbl>
## 1        12.1  11.9  50
## 2        17     7     0.48
## 3        14.4   9.6   1.35
## 4        14.9   9.1   0.019
## 5         4    20   600
## 6        14.4   9.6   3.85
```

也可以这样写

```
msleep %>%
  select_if(~is.numeric(.) & mean(., na.rm=TRUE) > 10) %>% head()
```

```
## # A tibble: 6 x 3
##   sleep_total awake  bodywt
##         <dbl> <dbl>   <dbl>
## 1        12.1  11.9  50
## 2        17     7     0.48
## 3        14.4   9.6   1.35
## 4        14.9   9.1   0.019
## 5         4    20   600
## 6        14.4   9.6   3.85
```

另一个有用的select_if参数是n_distinct()，它可以在列中找到的不同值出现的数量

```
msleep %>% select_if(~n_distinct(.) < 10) %>% head()
```

```
## # A tibble: 6 x 2
##   vore  conservation
##   <chr> <chr>
## 1 carni lc
## 2 omni  <NA>
## 3 herbi nt
## 4 omni  lc
## 5 herbi domesticated
## 6 herbi <NA>
```

3.7 对列进行重新排序

everything()函数可将选择的列移至表格最前

```
msleep %>% select(conservation, sleep_total, everything()) %>% head()
```

```
## # A tibble: 6 x 11
##   conservation sleep_total name    genus vore  order sleep_rem sleep_cycle awake
##   <chr>              <dbl> <chr>   <chr> <chr> <chr>     <dbl>       <dbl> <dbl>
## 1 lc                  12.1 Cheetah Acin~ carni Carn~      NA        NA      11.9
## 2 <NA>                17   Owl mo~ Aotus omni  Prim~       1.8      NA       7
## 3 nt                  14.4 Mounta~ Aplo~ herbi Rode~       2.4      NA       9.6
## 4 lc                  14.9 Greate~ Blar~ omni  Sori~       2.3       0.133   9.1
## 5 domesticated         4   Cow     Bos   herbi Arti~       0.7       0.667  20
## 6 <NA>                14.4 Three-~ Brad~ herbi Pilo~       2.2       0.767   9.6
## # ... with 2 more variables: brainwt <dbl>, bodywt <dbl>
```

3.8 更改列名

```
msleep %>%
  select(animal = name, sleep_total, extinction_threat = conservation)
```

```
## # A tibble: 83 x 3
##    animal                    sleep_total extinction_threat
##    <chr>                           <dbl> <chr>
##  1 Cheetah                          12.1 lc
##  2 Owl monkey                       17   <NA>
##  3 Mountain beaver                  14.4 nt
```

```
##  4 Greater short-tailed shrew       14.9 lc
##  5 Cow                               4   domesticated
##  6 Three-toed sloth                 14.4 <NA>
##  7 Northern fur seal                 8.7 vu
##  8 Vesper mouse                      7   <NA>
##  9 Dog                              10.1 domesticated
## 10 Roe deer                          3   lc
## # ... with 73 more rows
```

也可以通过rename()函数来重命名

```
msleep %>%
  rename(animal = name, extinction_threat = conservation)
```

```
## # A tibble: 83 x 11
##    animal   genus vore  order extinction_thre~ sleep_total sleep_rem sleep_cycle
##    <chr>    <chr> <chr> <chr> <chr>                  <dbl>     <dbl>       <dbl>
##  1 Cheetah  Acin~ carni Carn~ lc                      12.1      NA        NA
##  2 Owl mon~ Aotus omni  Prim~ <NA>                    17         1.8      NA
##  3 Mountai~ Aplo~ herbi Rode~ nt                      14.4       2.4      NA
##  4 Greater~ Blar~ omni  Sori~ lc                      14.9       2.3       0.133
##  5 Cow      Bos   herbi Arti~ domesticated            4         0.7       0.667
##  6 Three-t~ Brad~ herbi Pilo~ <NA>                    14.4       2.2       0.767
##  7 Norther~ Call~ carni Carn~ vu                      8.7       1.4       0.383
##  8 Vesper ~ Calo~ <NA>  Rode~ <NA>                    7        NA        NA
##  9 Dog      Canis carni Carn~ domesticated           10.1       2.9       0.333
## 10 Roe deer Capr~ herbi Arti~ lc                      3        NA        NA
## # ... with 73 more rows, and 3 more variables: awake <dbl>, brainwt <dbl>,
## #   bodywt <dbl>
```

3.9 重新格式化所有列名

select_all()函数允许更改所有列，并以一个函数作为参数。要以大写形式获取所有列名称,可以使用toupper()，也可以使用tolower()将其全部转化为小写

```
msleep %>% select_all(toupper) %>% head()
```

```
## # A tibble: 6 x 11
##   NAME    GENUS VORE  ORDER CONSERVATION SLEEP_TOTAL SLEEP_REM SLEEP_CYCLE AWAKE
##   <chr>   <chr> <chr> <chr> <chr>              <dbl>     <dbl>       <dbl> <dbl>
## 1 Cheetah Acin~ carni Carn~ lc                  12.1      NA        NA      11.9
## 2 Owl mo~ Aotus omni  Prim~ <NA>                17         1.8      NA       7
## 3 Mounta~ Aplo~ herbi Rode~ nt                  14.4       2.4      NA       9.6
## 4 Greate~ Blar~ omni  Sori~ lc                  14.9       2.3       0.133   9.1
## 5 Cow     Bos   herbi Arti~ domesticated         4         0.7       0.667  20
## 6 Three-~ Brad~ herbi Pilo~ <NA>                14.4       2.2       0.767   9.6
## # ... with 2 more variables: BRAINWT <dbl>, BODYWT <dbl>
```

3.10 自主创建函数(重点)

将列名中的空格替换为下划线

```
msleep2 <- select(msleep, name, sleep_total, brainwt)
colnames(msleep2) <- c("name", "sleep total", "brain weight")

msleep2 %>% select_all(~str_replace(., " ", "_"))
```

```
## # A tibble: 83 x 3
##    name                      sleep_total brain_weight
```

```
##    <chr>                          <dbl>        <dbl>
##  1 Cheetah                         12.1     NA
##  2 Owl monkey                      17        0.0155
##  3 Mountain beaver                 14.4     NA
##  4 Greater short-tailed shrew      14.9      0.00029
##  5 Cow                              4        0.423
##  6 Three-toed sloth                14.4     NA
##  7 Northern fur seal                8.7     NA
##  8 Vesper mouse                     7       NA
##  9 Dog                             10.1      0.07
## 10 Roe deer                         3        0.0982
## # ... with 73 more rows
```

还可以使用select_all与str_replace来消除多余的字符

```
msleep2 <- select(msleep, name, sleep_total, brainwt)
colnames(msleep2) <- c("Q1 name", "Q2 sleep total", "Q3 brain weight")
msleep2[1:3,]
```

```
## # A tibble: 3 x 3
##   `Q1 name`       `Q2 sleep total` `Q3 brain weight`
##   <chr>                      <dbl>             <dbl>
## 1 Cheetah                     12.1           NA
## 2 Owl monkey                  17              0.0155
## 3 Mountain beaver             14.4           NA
```

```
msleep2 %>%
  select_all(~str_replace(., "Q[0-9]+", "")) %>%
  select_all(~str_replace(., " ", "_"))
```

```
## # A tibble: 83 x 3
##    `_name`                    `_sleep total` `_brain weight`
##    <chr>                               <dbl>           <dbl>
##  1 Cheetah                              12.1        NA
##  2 Owl monkey                           17           0.0155
##  3 Mountain beaver                      14.4        NA
##  4 Greater short-tailed shrew           14.9         0.00029
##  5 Cow                                   4           0.423
##  6 Three-toed sloth                     14.4        NA
##  7 Northern fur seal                     8.7        NA
##  8 Vesper mouse                          7          NA
##  9 Dog                                  10.1         0.07
## 10 Roe deer                              3           0.0982
## # ... with 73 more rows
```

3.11 从行到列

某些数据框的行名本身实际上并不是一列，例如mtcars数据集

```
                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

如果希望此列为实际列，则可以使用该 rownames_to_column()函数，并指定新的列名称

```
mtcars %>%
  tibble::rownames_to_column("car_model") %>% head
```

```
##           car_model  mpg cyl disp  hp drat    wt  qsec vs am gear carb
## 1         Mazda RX4 21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## 2     Mazda RX4 Wag 21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## 3        Datsun 710 22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## 4    Hornet 4 Drive 21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## 5 Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## 6           Valiant 18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

4. filter按行处理数据

还是使用我们熟悉的iris(鸢尾花)数据集，可以看到数据有5列，150行，数据类型为数据框；分别表示Sepal.Length(花萼长度)，Sepal.Width(花萼宽度) 、Petal.Length(花瓣长度)，Petal.Width(花瓣宽度) 、Species(花的类型)，其中花有3种类型(setosa、versicolor、virginica)

```
iris %>% as_tibble() %>% head()
```

```
## # A tibble: 6 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
```

4.1 基本行过滤

筛选出含有setosa的行，注意是 `==`

```
iris %>% as_tibble() %>% filter(Species=="setosa") %>% head()
```

```
## # A tibble: 6 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
```

筛选出不含有setosa的行，R中 `!=` 代表不等于

```
iris %>% as_tibble() %>% filter(Species !="setosa") %>% head()
```

```
## # A tibble: 6 x 5
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##          <dbl>       <dbl>        <dbl>       <dbl> <fct>
## 1          7           3.2          4.7         1.4 versicolor
## 2          6.4         3.2          4.5         1.5 versicolor
## 3          6.9         3.1          4.9         1.5 versicolor
## 4          5.5         2.3          4           1.3 versicolor
```

```
## 5          6.5         2.8          4.6         1.5 versicolor
## 6          5.7         2.8          4.5         1.3 versicolor
```

```
# 此2种方法结果一致，处理复杂数据时推荐第二种
iris %>% as_tibble() %>% filter(!Species %in% "setosa") %>% head()
```

```
## # A tibble: 6 x 5
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##          <dbl>       <dbl>        <dbl>       <dbl> <fct>
## 1          7           3.2          4.7         1.4 versicolor
## 2          6.4         3.2          4.5         1.5 versicolor
## 3          6.9         3.1          4.9         1.5 versicolor
## 4          5.5         2.3          4           1.3 versicolor
## 5          6.5         2.8          4.6         1.5 versicolor
## 6          5.7         2.8          4.5         1.3 versicolor
```

根据2个关键词进行筛选 通过`%in%`进行判断

```
iris %>% as_tibble() %>%
  filter(.,Species %in% c("setosa","virginica")) %>% head()
```

```
## # A tibble: 6 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
```

`!Species`表示不包含在其中，此处注意前面的`.,`

```
iris %>% as_tibble() %>%
  filter(.,!Species %in% c("setosa","virginica")) %>% head()
```

```
## # A tibble: 6 x 5
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##          <dbl>       <dbl>        <dbl>       <dbl> <fct>
## 1          7           3.2          4.7         1.4 versicolor
## 2          6.4         3.2          4.5         1.5 versicolor
## 3          6.9         3.1          4.9         1.5 versicolor
## 4          5.5         2.3          4           1.3 versicolor
## 5          6.5         2.8          4.6         1.5 versicolor
## 6          5.7         2.8          4.5         1.3 versicolor
```

4.2 根据多个条件进行筛选

- filter(condition1, condition2) 将返回同时满足两个条件的行
- filter(condition1, !condition2) 将返回条件一为真但条件二为不成立的所有行
- filter(condition1 | condition2) 将返回满足条件1和/或条件2的行
- filter(xor(condition1, condition2)将返回仅满足一个条件的所有行，而不是同时满足两个条件的所有行

```
iris %>% as_tibble() %>%
  filter(Species=="setosa",Sepal.Length >= 5) %>% head()
```

```
## # A tibble: 6 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          5           3.6          1.4         0.2 setosa
## 3          5.4         3.9          1.7         0.4 setosa
## 4          5           3.4          1.5         0.2 setosa
## 5          5.4         3.7          1.5         0.2 setosa
## 6          5.8         4            1.2         0.2 setosa
```

`&` 在R中表示和的意思与`,`作用一致；`|`或的意思

```
iris %>% as_tibble() %>%
  filter(.,(Species=="setosa" & !Sepal.Length >= 5)) %>% head()
```

```
## # A tibble: 6 x 5
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##          <dbl>       <dbl>        <dbl>       <dbl> <fct>
## 1          4.9         3            1.4         0.2 setosa
## 2          4.7         3.2          1.3         0.2 setosa
## 3          4.6         3.1          1.5         0.2 setosa
## 4          4.6         3.4          1.4         0.3 setosa
## 5          4.4         2.9          1.4         0.2 setosa
## 6          4.9         3.1          1.5         0.1 setosa
```

```
iris %>% as_tibble() %>%
  filter(.,(Species=="setosa" | Sepal.Length >= 5)) %>% head()
```

```
## # A tibble: 6 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
```

4.3 重点xor

xor只返回仅满足一个条件的所有行，而不是同时满足两个条件的所有行即`c(T,F)` | `c(F,T)`2种情况

```
iris %>% as_tibble() %>%
  filter(.,xor(Species=="setosa",Sepal.Length >= 5)) %>% head()
```

```
## # A tibble: 6 x 5
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##          <dbl>       <dbl>        <dbl>       <dbl> <fct>
## 1          4.9         3            1.4         0.2 setosa
## 2          4.7         3.2          1.3         0.2 setosa
## 3          4.6         3.1          1.5         0.2 setosa
## 4          4.6         3.4          1.4         0.3 setosa
## 5          4.4         2.9          1.4         0.2 setosa
## 6          4.9         3.1          1.5         0.1 setosa
```

可通过以下代码验证上面的结果

```
iris %>% as_tibble() %>%
  filter(.,(Species=="setosa" & !Sepal.Length >= 5)) %>% head()
```

```
## # A tibble: 6 x 5
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##          <dbl>       <dbl>        <dbl>       <dbl> <fct>
## 1          4.9         3            1.4         0.2 setosa
## 2          4.7         3.2          1.3         0.2 setosa
## 3          4.6         3.1          1.5         0.2 setosa
## 4          4.6         3.4          1.4         0.3 setosa
## 5          4.4         2.9          1.4         0.2 setosa
## 6          4.9         3.1          1.5         0.1 setosa
```

```
iris %>% as_tibble() %>%
  filter(.,Species!= "setosa",Sepal.Length >= 5) %>% head()
```

```
## # A tibble: 6 x 5
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##          <dbl>       <dbl>        <dbl>       <dbl> <fct>
## 1          7           3.2          4.7         1.4 versicolor
## 2          6.4         3.2          4.5         1.5 versicolor
## 3          6.9         3.1          4.9         1.5 versicolor
## 4          5.5         2.3          4           1.3 versicolor
## 5          6.5         2.8          4.6         1.5 versicolor
## 6          5.7         2.8          4.5         1.3 versicolor
```

4.4 多条件嵌套过滤

此代码将首先提取出含有`setosa`的行，之后根据`Sepal.Length >= 5`这一条件对其进行过滤，最后将含有`versicolor","virginica"`的数据追加上去

```
iris %>% as_tibble() %>%
  filter(.,(Species=="setosa" & Sepal.Length >= 5)|(Species %in% c("versicolor","virginica")))
```

```
## # A tibble: 130 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          5           3.6          1.4         0.2 setosa
##  3          5.4         3.9          1.7         0.4 setosa
##  4          5           3.4          1.5         0.2 setosa
##  5          5.4         3.7          1.5         0.2 setosa
##  6          5.8         4            1.2         0.2 setosa
##  7          5.7         4.4          1.5         0.4 setosa
##  8          5.4         3.9          1.3         0.4 setosa
##  9          5.1         3.5          1.4         0.3 setosa
## 10          5.7         3.8          1.7         0.3 setosa
## # ... with 120 more rows
```

删除Species中含有`NA`的行

```
iris %>% filter(!is.na(Species)) %>% head()
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          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          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

4.5 跨多列过滤

- filter_all() 过滤所有列
- filter_if()需要一个返回的布尔值以指示要过滤列的类型。如果是符合条件则将遵循这些列进行过滤
- filter_at()要求在vars() 参数中指定要进行过滤的列

4.5.1 filter_all( )

对数值执行全部筛选：此代码将保留任何值等于5的行

```
iris %>% filter_all(any_vars(. == 5)) %>% head()
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1            5         3.6          1.4         0.2  setosa
## 2            5         3.4          1.5         0.2  setosa
## 3            5         3.0          1.6         0.2  setosa
## 4            5         3.4          1.6         0.4  setosa
## 5            5         3.2          1.2         0.2  setosa
## 6            5         3.5          1.3         0.3  setosa
```

对字符串进行过滤，在所有列中检索含有“Ca”的字符串，需要将条件包装在any_vars()中

```
msleep %>%
  select(name:order, sleep_total, -vore) %>%
  filter_all(any_vars(str_detect(., pattern = "Ca"))) %>% head()
```

```
## # A tibble: 6 x 4
##   name              genus       order        sleep_total
##   <chr>             <chr>       <chr>              <dbl>
## 1 Cheetah           Acinonyx    Carnivora           12.1
## 2 Northern fur seal Callorhinus Carnivora            8.7
## 3 Vesper mouse      Calomys     Rodentia             7
## 4 Dog               Canis       Carnivora           10.1
## 5 Roe deer          Capreolus   Artiodactyla         3
## 6 Goat              Capri       Artiodactyla         5.3
```

4.5.2 filter_if( )

下面这段代码首先对列的类型进行判断，再在字符列中筛选`NA`

```
msleep %>%
  select(name:order, sleep_total:sleep_rem) %>%
  filter_if(is.character, any_vars(is.na(.))) %>% head()
```

```
## # A tibble: 6 x 6
##   name            genus       vore  order          sleep_total sleep_rem
##   <chr>           <chr>       <chr> <chr>                <dbl>     <dbl>
## 1 Vesper mouse    Calomys     <NA>  Rodentia               7        NA
## 2 Desert hedgehog Paraechinus <NA>  Erinaceomorpha        10.3       2.7
## 3 Deer mouse      Peromyscus  <NA>  Rodentia              11.5      NA
## 4 Phalanger       Phalanger   <NA>  Diprotodontia         13.7       1.8
## 5 Rock hyrax      Procavia    <NA>  Hyracoidea             5.4       0.5
## 6 Mole rat        Spalax      <NA>  Rodentia              10.6       2.4
```

4.5.3 filter_at( )

filter_at( )它不筛选所有列，也不需要您指定列的类型,可以通过vars() 参数选择要对那些列进行筛选

```
iris %>%
  filter_at(vars(Sepal.Length,Petal.Length),all_vars(. >=6)) %>% head()
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
## 1          6.3         3.3          6.0         2.5 virginica
## 2          7.6         3.0          6.6         2.1 virginica
## 3          7.3         2.9          6.3         1.8 virginica
## 4          7.2         3.6          6.1         2.5 virginica
## 5          7.7         3.8          6.7         2.2 virginica
## 6          7.7         2.6          6.9         2.3 virginica
```

5. mutate创建新列

mutate函数其基本功能为创建新列；**mutate**中的选项几乎是无穷无尽的，可以通过各种函数之间的组合来对数据集做任意的处理，下面通过具体的案例来进行演示

这次我们使用R内置的数据集msleep，其中包括哺乳动物的睡眠时间。让我们首先加载包并查看数据：

```
msleep %>% as_tibble()
```

```
## # A tibble: 83 x 11
##    name   genus vore  order conservation sleep_total sleep_rem sleep_cycle awake
##    <chr>  <chr> <chr> <chr> <chr>              <dbl>     <dbl>       <dbl> <dbl>
##  1 Cheet~ Acin~ carni Carn~ lc                  12.1      NA        NA      11.9
##  2 Owl m~ Aotus omni  Prim~ <NA>                17         1.8      NA       7
##  3 Mount~ Aplo~ herbi Rode~ nt                  14.4       2.4      NA       9.6
##  4 Great~ Blar~ omni  Sori~ lc                  14.9       2.3       0.133   9.1
##  5 Cow    Bos   herbi Arti~ domesticated         4         0.7       0.667  20
##  6 Three~ Brad~ herbi Pilo~ <NA>                14.4       2.2       0.767   9.6
##  7 North~ Call~ carni Carn~ vu                   8.7       1.4       0.383  15.3
##  8 Vespe~ Calo~ <NA>  Rode~ <NA>                 7        NA        NA      17
##  9 Dog    Canis carni Carn~ domesticated        10.1       2.9       0.333  13.9
## 10 Roe d~ Capr~ herbi Arti~ lc                   3        NA        NA      21
## # ... with 73 more rows, and 2 more variables: brainwt <dbl>, bodywt <dbl>
```

5.1 mutate 基础操作

最简单的的操作就是根据其他列中的值进行计算。在示例代码中，我们将睡眠数据从以小时为单位更改为分钟为单位

```
msleep %>%
  select(name,sleep_total) %>%
  mutate(sleep_total_min = sleep_total * 60) %>% head()
```

```
## # A tibble: 6 x 3
##   name                        sleep_total sleep_total_min
##   <chr>                             <dbl>           <dbl>
## 1 Cheetah                            12.1             726
## 2 Owl monkey                         17              1020
## 3 Mountain beaver                    14.4             864
## 4 Greater short-tailed shrew         14.9             894
## 5 Cow                                 4               240
## 6 Three-toed sloth                   14.4             864
```

下列代码创建了两列新列：一列显示了睡眠时间与平均睡眠时间的差异，另一列显示了与睡眠时间最少的动物之间的差异；**round**( )对数据进行四舍五入操作

```
msleep %>%
  select(name, sleep_total) %>%
  mutate(AVG = sleep_total - round(mean(sleep_total), 1),
         MIN = sleep_total - min(sleep_total)) %>% head()
```

```
## # A tibble: 6 x 4
##   name                        sleep_total   AVG   MIN
##   <chr>                             <dbl> <dbl> <dbl>
## 1 Cheetah                            12.1   1.7  10.2
## 2 Owl monkey                         17     6.6  15.1
## 3 Mountain beaver                    14.4   4    12.5
## 4 Greater short-tailed shrew         14.9   4.5  13
## 5 Cow                                 4    -6.4   2.1
## 6 Three-toed sloth                   14.4   4    12.5
```

选择特定列按行求均值，rowwise()说明按行进行操作

```
msleep %>%
  select(name, contains("sleep")) %>%
  rowwise() %>%
  mutate(avg = mean(c(sleep_rem,sleep_cycle))) %>% head()
```

```
## # A tibble: 6 x 5
## # Rowwise:
##   name                        sleep_total sleep_rem sleep_cycle    avg
##   <chr>                             <dbl>     <dbl>       <dbl>  <dbl>
## 1 Cheetah                            12.1      NA         NA     NA
## 2 Owl monkey                         17         1.8       NA     NA
## 3 Mountain beaver                    14.4       2.4       NA     NA
## 4 Greater short-tailed shrew         14.9       2.3        0.133  1.22
## 5 Cow                                 4         0.7        0.667  0.683
## 6 Three-toed sloth                   14.4       2.2        0.767  1.48
```

通过**ifelse**判断语句对数据进行操作，如果`brainwt > 4`返回`NA`,不满足此条件返回原值

```
msleep %>%
  select(name, brainwt) %>%
  mutate(brainwt2 = ifelse(brainwt > 4, NA, brainwt)) %>%
  arrange(desc(brainwt)) %>% head()
```

```
## # A tibble: 6 x 3
##   name             brainwt brainwt2
##   <chr>              <dbl>    <dbl>
## 1 African elephant   5.71    NA
## 2 Asian elephant     4.60    NA
## 3 Human              1.32     1.32
## 4 Horse              0.655    0.655
## 5 Chimpanzee         0.44     0.44
## 6 Cow                0.423    0.423
```

也可以结合使用stringr的功能或正则表达式来对字符串列进行操作:示例代码将返回动物名称的最后一个单词，并使其小写

```
msleep %>%
  select(name) %>%
```

```
  mutate(name_last_word = tolower(str_extract(name, pattern = "\\w+$"))) %>% head()
```

```
## # A tibble: 6 x 2
##   name                        name_last_word
##   <chr>                       <chr>
## 1 Cheetah                     cheetah
## 2 Owl monkey                  monkey
## 3 Mountain beaver             beaver
## 4 Greater short-tailed shrew shrew
## 5 Cow                         cow
## 6 Three-toed sloth            sloth
```

5.2 对多列同时进行操作

- mutate_all() 将对所有列进行操作
- mutate_if()首先需要一个返回布尔值，如果是T，则将在这些变量上执行mutate指令
- mutate_at()要求在vars() 参数内指定要进行改变的列

5.2.1 将所有数据转换为小写：

```
msleep %>% mutate_all(tolower) %>% head
```

```
## # A tibble: 6 x 11
##   name    genus vore  order conservation sleep_total sleep_rem sleep_cycle awake
##   <chr>   <chr> <chr> <chr> <chr>        <chr>       <chr>     <chr>       <chr>
## 1 cheetah acin~ carni carn~ lc           12.1        <NA>      <NA>        11.9
## 2 owl mo~ aotus omni  prim~ <NA>         17          1.8       <NA>        7
## 3 mounta~ aplo~ herbi rode~ nt           14.4        2.4       <NA>        9.6
## 4 greate~ blar~ omni  sori~ lc           14.9        2.3       0.133333333 9.1
## 5 cow     bos   herbi arti~ domesticated 4           0.7       0.666666667 20
## 6 three-~ brad~ herbi pilo~ <NA>         14.4        2.2       0.766666667 9.6
## # ... with 2 more variables: brainwt <chr>, bodywt <chr>
```

5.2.2 所有列添加`" /n "`

```
msleep %>% mutate_all(~paste(., "  /n  ")) %>% head()
```

```
## # A tibble: 6 x 11
##   name   genus  vore  order conservation sleep_total sleep_rem sleep_cycle awake
##   <chr>  <chr>  <chr> <chr> <chr>        <chr>       <chr>     <chr>       <chr>
## 1 "Chee~ "Acin~ "car~ "Car~ "lc   /n  "  "12.1   /n~ "NA   /n~ "NA   /n  " "11.~
## 2 "Owl ~ "Aotu~ "omn~ "Pri~ "NA   /n  "  "17   /n  " "1.8   /~ "NA   /n  " "7  ~
## 3 "Moun~ "Aplo~ "her~ "Rod~ "nt   /n  "  "14.4   /n~ "2.4   /~ "NA   /n  " "9.6~
## 4 "Grea~ "Blar~ "omn~ "Sor~ "lc   /n  "  "14.9   /n~ "2.3   /~ "0.1333333~ "9.1~
## 5 "Cow ~ "Bos ~ "her~ "Art~ "domesticat~ "4   /n  "  "0.7   /~ "0.6666666~ "20 ~
## 6 "Thre~ "Brad~ "her~ "Pil~ "NA   /n  "  "14.4   /n~ "2.2   /~ "0.7666666~ "9.6~
## # ... with 2 more variables: brainwt <chr>, bodywt <chr>
```

5.2.3 将`"  /n  "`全部替换为空

```
msleep_ohno <- msleep %>% mutate_all(~paste(., "  /n  ")) %>% head()

msleep_ohno %>%
  mutate_all(~str_replace_all(., "/n", "")) %>%
  mutate_all(str_trim) %>% head()
```

```
## # A tibble: 6 x 11
##   name    genus vore  order conservation sleep_total sleep_rem sleep_cycle awake
##   <chr>   <chr> <chr> <chr> <chr>        <chr>       <chr>     <chr>       <chr>
## 1 Cheetah Acin~ carni Carn~ lc           12.1        NA        NA          11.9
## 2 Owl mo~ Aotus omni  Prim~ NA           17          1.8       NA          7
## 3 Mounta~ Aplo~ herbi Rode~ nt           14.4        2.4       NA          9.6
## 4 Greate~ Blar~ omni  Sori~ lc           14.9        2.3       0.133333333 9.1
## 5 Cow     Bos   herbi Arti~ domesticated 4           0.7       0.666666667 20
## 6 Three-~ Brad~ herbi Pilo~ NA           14.4        2.2       0.766666667 9.6
## # ... with 2 more variables: brainwt <chr>, bodywt <chr>
```

5.3 mutate_if()对数据进行判断

如果数据类型是数值，对其进行四舍五入操作

```
msleep %>%
  select(name, sleep_total:bodywt) %>%
  mutate_if(is.numeric, round) %>% head()
```

```
## # A tibble: 6 x 7
##   name                    sleep_total sleep_rem sleep_cycle awake brainwt bodywt
##   <chr>                         <dbl>     <dbl>       <dbl> <dbl>   <dbl>  <dbl>
## 1 Cheetah                          12        NA          NA    12      NA     50
## 2 Owl monkey                       17         2          NA     7       0      0
## 3 Mountain beaver                  14         2          NA    10      NA      1
## 4 Greater short-tailed s~          15         2           0     9       0      0
## 5 Cow                               4         1           1    20       0    600
## 6 Three-toed sloth                 14         2           1    10      NA      4
```

5.4 mutate_at( )对特定列进行操作

对列名含有`sleep`的进行操作

```
msleep %>%
  select(name, sleep_total:awake) %>%
  mutate_at(vars(contains("sleep")), ~(.*60)) %>% head
```

```
## # A tibble: 6 x 5
##   name                       sleep_total sleep_rem sleep_cycle awake
##   <chr>                            <dbl>     <dbl>       <dbl> <dbl>
## 1 Cheetah                            726        NA          NA  11.9
## 2 Owl monkey                        1020       108          NA   7
## 3 Mountain beaver                    864       144          NA   9.6
## 4 Greater short-tailed shrew         894       138        8.00   9.1
## 5 Cow                                240        42        40.0  20
## 6 Three-toed sloth                   864       132        46.0   9.6
```

5.5 更改列名

```
msleep %>%
  select(name, sleep_total:awake) %>%
  mutate_at(vars(contains("sleep")), ~(.*60)) %>%
  rename_at(vars(contains("sleep")), ~paste0(.,"_min")) %>% head
```

```
## # A tibble: 6 x 5
##   name                       sleep_total_min sleep_rem_min sleep_cycle_min awake
##   <chr>                                <dbl>         <dbl>           <dbl> <dbl>
## 1 Cheetah                                726            NA              NA  11.9
```

```
## 2 Owl monkey                          1020          108          NA      7
## 3 Mountain beaver                      864          144          NA      9.6
## 4 Greater short-tailed shrew           894          138           8.00   9.1
## 5 Cow                                  240           42          40.0   20
## 6 Three-toed sloth                     864          132          46.0    9.6
```

保留原始数据

```
msleep %>%
  select(name, sleep_total:awake) %>%
  mutate_at(vars(contains("sleep")), funs(min = .*60)) %>% head()
```

```
## # A tibble: 6 x 8
##   name     sleep_total sleep_rem sleep_cycle awake sleep_total_min sleep_rem_min
##   <chr>          <dbl>     <dbl>       <dbl> <dbl>           <dbl>         <dbl>
## 1 Cheetah         12.1      NA        NA      11.9             726            NA
## 2 Owl mon~        17         1.8      NA       7               1020           108
## 3 Mountai~        14.4       2.4      NA       9.6             864            144
## 4 Greater~        14.9       2.3       0.133   9.1             894            138
## 5 Cow              4         0.7       0.667  20               240             42
## 6 Three-t~        14.4       2.2       0.767   9.6             864            132
## # ... with 1 more variable: sleep_cycle_min <dbl>
```

5.6 ifelse创建2个级别的离散列

```
msleep %>%
  select(name, sleep_total) %>%
  mutate(sleep_time = ifelse(sleep_total > 10, "long", "short")) %>% head()
```

```
## # A tibble: 6 x 3
##   name                        sleep_total sleep_time
##   <chr>                             <dbl> <chr>
## 1 Cheetah                            12.1 long
## 2 Owl monkey                         17   long
## 3 Mountain beaver                    14.4 long
## 4 Greater short-tailed shrew         14.9 long
## 5 Cow                                 4   short
## 6 Three-toed sloth                   14.4 long
```

5.7 case_when创建多级离散列

此函数在后续数据清洗中有大有，需要多多练习

```
msleep %>%
  select(name, sleep_total) %>%
  mutate(sleep_total_discr = case_when(
    sleep_total > 13 ~ "very long",
    sleep_total > 10 ~ "long",
    sleep_total > 7 ~ "limited",
    TRUE ~ "short")) %>% head()
```

```
## # A tibble: 6 x 3
##   name                        sleep_total sleep_total_discr
##   <chr>                             <dbl> <chr>
## 1 Cheetah                            12.1 long
## 2 Owl monkey                         17   very long
## 3 Mountain beaver                    14.4 very long
## 4 Greater short-tailed shrew         14.9 very long
```

```
## 5 Cow                              4   short
## 6 Three-toed sloth                14.4 very long
```

5.8 将数据转化为NA

```
msleep %>%
  select(name:order) %>%
  na_if("omni") %>% head()
```

```
## # A tibble: 6 x 4
##   name                        genus      vore  order
##   <chr>                       <chr>      <chr> <chr>
## 1 Cheetah                     Acinonyx   carni Carnivora
## 2 Owl monkey                  Aotus      <NA>  Primates
## 3 Mountain beaver             Aplodontia herbi Rodentia
## 4 Greater short-tailed shrew Blarina    <NA>  Soricomorpha
## 5 Cow                         Bos        herbi Artiodactyla
## 6 Three-toed sloth            Bradypus   herbi Pilosa
```

6. rowwise行处理函数

依然还是使用我们熟悉的**iris**数据集

```
iris %>% as_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
```

可以看到有5列其中一列为因子其余四列为数值，进行计算时只需要数值列

6.1 按行计算均值

方法1

```
iris %>%  as_tibble() %>% select(-Species) %>%
  mutate(.,mean=rowMeans(.)) %>% head()
```

```
## # A tibble: 6 x 5
##   Sepal.Length Sepal.Width Petal.Length Petal.Width  mean
##          <dbl>       <dbl>        <dbl>       <dbl> <dbl>
## 1          5.1         3.5          1.4         0.2  2.55
## 2          4.9         3            1.4         0.2  2.38
## 3          4.7         3.2          1.3         0.2  2.35
## 4          4.6         3.1          1.5         0.2  2.35
## 5          5           3.6          1.4         0.2  2.55
## 6          5.4         3.9          1.7         0.4  2.85
```

方法2

根据逻辑判断只选择了数值列

```
iris %>% as_tibble() %>% select_if(is.numeric) %>%
  mutate(.,mean=rowMeans(.)) %>% head()
```

```
## # A tibble: 6 x 5
##   Sepal.Length Sepal.Width Petal.Length Petal.Width  mean
##          <dbl>       <dbl>        <dbl>       <dbl> <dbl>
## 1          5.1         3.5          1.4         0.2  2.55
## 2          4.9         3            1.4         0.2  2.38
## 3          4.7         3.2          1.3         0.2  2.35
## 4          4.6         3.1          1.5         0.2  2.35
## 5          5           3.6          1.4         0.2  2.55
## 6          5.4         3.9          1.7         0.4  2.85
```

方法3

```
iris %>% as_tibble() %>% select_if(is.numeric) %>%
  rowwise() %>%
  mutate(mean = mean(c_across(Sepal.Length:Petal.Width))) %>% head()
```

```
## # A tibble: 6 x 5
## # Rowwise:
##   Sepal.Length Sepal.Width Petal.Length Petal.Width  mean
##          <dbl>       <dbl>        <dbl>       <dbl> <dbl>
## 1          5.1         3.5          1.4         0.2  2.55
## 2          4.9         3            1.4         0.2  2.38
## 3          4.7         3.2          1.3         0.2  2.35
## 4          4.6         3.1          1.5         0.2  2.35
## 5          5           3.6          1.4         0.2  2.55
## 6          5.4         3.9          1.7         0.4  2.85
```

通过**rowwise**函数说明对数据按行进行处理，c_across选择多列

方法4

```
iris %>% as_tibble() %>%
  rowwise() %>%
  mutate(mean = rowMeans(across(where(is.numeric)))) %>% head
```

```
## # A tibble: 6 x 6
## # Rowwise:
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species  mean
##          <dbl>       <dbl>        <dbl>       <dbl> <fct>   <dbl>
## 1          5.1         3.5          1.4         0.2 setosa   2.55
## 2          4.9         3            1.4         0.2 setosa   2.38
## 3          4.7         3.2          1.3         0.2 setosa   2.35
## 4          4.6         3.1          1.5         0.2 setosa   2.35
## 5          5           3.6          1.4         0.2 setosa   2.55
## 6          5.4         3.9          1.7         0.4 setosa   2.85
```

通过**across**函数只选择了数值列，此函数异常强大，灵活使用能使代码简洁无比

6.2 按行统计最小值

```
iris %>% as_tibble() %>%
  rowwise() %>%
```

```
  mutate(min = min(across(where(is.numeric)))) %>% head()
```

```
## # A tibble: 6 x 6
## # Rowwise:
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species   min
##          <dbl>       <dbl>        <dbl>       <dbl> <fct>   <dbl>
## 1          5.1         3.5          1.4         0.2 setosa    0.2
## 2          4.9         3            1.4         0.2 setosa    0.2
## 3          4.7         3.2          1.3         0.2 setosa    0.2
## 4          4.6         3.1          1.5         0.2 setosa    0.2
## 5          5           3.6          1.4         0.2 setosa    0.2
## 6          5.4         3.9          1.7         0.4 setosa    0.4
```

6.3 按行统计最大值

```
iris %>% as_tibble() %>%
  rowwise() %>%
  mutate(max = max(across(where(is.numeric)))) %>% head()
```

```
## # A tibble: 6 x 6
## # Rowwise:
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species   max
##          <dbl>       <dbl>        <dbl>       <dbl> <fct>   <dbl>
## 1          5.1         3.5          1.4         0.2 setosa    5.1
## 2          4.9         3            1.4         0.2 setosa    4.9
## 3          4.7         3.2          1.3         0.2 setosa    4.7
## 4          4.6         3.1          1.5         0.2 setosa    4.6
## 5          5           3.6          1.4         0.2 setosa    5
## 6          5.4         3.9          1.7         0.4 setosa    5.4
```

6.4 按行求和

```
iris %>% as_tibble() %>%
  rowwise() %>%
  mutate(sum = sum(across(where(is.numeric)))) %>% head()
```

```
## # A tibble: 6 x 6
## # Rowwise:
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species   sum
##          <dbl>       <dbl>        <dbl>       <dbl> <fct>   <dbl>
## 1          5.1         3.5          1.4         0.2 setosa   10.2
## 2          4.9         3            1.4         0.2 setosa    9.5
## 3          4.7         3.2          1.3         0.2 setosa    9.4
## 4          4.6         3.1          1.5         0.2 setosa    9.4
## 5          5           3.6          1.4         0.2 setosa   10.2
## 6          5.4         3.9          1.7         0.4 setosa   11.4
```

6.5 按行计算标准差

```
iris %>% as_tibble() %>%
  rowwise() %>%
  mutate(sd = sd(across(where(is.numeric)))) %>% head()
```

```
## # A tibble: 6 x 6
## # Rowwise:
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species    sd
##          <dbl>       <dbl>        <dbl>       <dbl> <fct>   <dbl>
## 1          5.1         3.5          1.4         0.2 setosa   2.18
```

```
## 2          4.9         3           1.4         0.2 setosa   2.04
## 3          4.7         3.2         1.3         0.2 setosa   2.00
## 4          4.6         3.1         1.5         0.2 setosa   1.91
## 5          5           3.6         1.4         0.2 setosa   2.16
## 6          5.4         3.9         1.7         0.4 setosa   2.23
```

6.6 统计每行中某值出现的次数

```
iris %>%  as_tibble() %>% select(-Species) %>%
  mutate(.,n=rowSums(. > 3)) %>% head()
```

```
## # A tibble: 6 x 5
##   Sepal.Length Sepal.Width Petal.Length Petal.Width     n
##          <dbl>       <dbl>        <dbl>       <dbl> <dbl>
## 1          5.1         3.5          1.4         0.2     2
## 2          4.9         3            1.4         0.2     1
## 3          4.7         3.2          1.3         0.2     2
## 4          4.6         3.1          1.5         0.2     2
## 5          5           3.6          1.4         0.2     2
## 6          5.4         3.9          1.7         0.4     2
```

7. tidyverse中的专职函数(上)

7.1 arrange()排序

- arrange 对列进行排序

```
arrange(mtcars,mpg) %>% as_tibble() %>% head()
```

```
## # A tibble: 6 x 11
##     mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
##   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1  10.4     8   472   205  2.93  5.25  18.0     0     0     3     4
## 2  10.4     8   460   215  3     5.42  17.8     0     0     3     4
## 3  13.3     8   350   245  3.73  3.84  15.4     0     0     3     4
## 4  14.3     8   360   245  3.21  3.57  15.8     0     0     3     4
## 5  14.7     8   440   230  3.23  5.34  17.4     0     0     3     4
## 6  15       8   301   335  3.54  3.57  14.6     0     1     5     8
```

arrange() 通过选定的列进行排序，默认为升序

7.2 desc()数据降序

- desc 降序排列

arrange()结合desc()对数据进行降序

```
arrange(mtcars,desc(mpg)) %>% as_tibble() %>% head()
```

```
## # A tibble: 6 x 11
##     mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
##   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1  33.9     4  71.1    65  4.22  1.84  19.9     1     1     4     1
## 2  32.4     4  78.7    66  4.08  2.2   19.5     1     1     4     1
## 3  30.4     4  75.7    52  4.93  1.62  18.5     1     1     4     2
## 4  30.4     4  95.1   113  3.77  1.51  16.9     1     1     5     2
## 5  27.3     4  79      66  4.08  1.94  18.9     1     1     4     1
## 6  26       4 120.     91  4.43  2.14  16.7     0     1     5     2
```

7.3 distinct()数据去重

- distinct 去重复行

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

df %>% distinct() %>% head()
```

```
## # A tibble: 6 x 2
##       x     y
##   <int> <int>
## 1     8     1
## 2     9     3
## 3     6     6
## 4     7     5
## 5     6     4
## 6     8     5
```

distinct() 针对数据框进行去重，unique针对向量进行去重

7.4 rename()更改列名

- raname 更改列名

```
rename(iris,petal_length=Petal.Length) %>% as_tibble()
```

7.5 relocate()更改列顺序

- relocate 更改列的顺序

```
iris %>% as_tibble() %>% relocate(Species) %>% head()
```

```
## # A tibble: 6 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
```

```
#下述方法也可以实现，但是较为麻烦
iris %>% as_tibble() %>% select(Species,everything()) %>% head()
```

```
## # A tibble: 6 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
```

```r
df <- tibble(a = 1, b = 1, c = 1, d = "a", e = "a", f = "a")

df
```

```
## # A tibble: 1 x 6
##       a     b     c d     e     f
##   <dbl> <dbl> <dbl> <chr> <chr> <chr>
## 1     1     1     1 a     a     a
```

7.5.1 指定列的顺序

```r
df %>% relocate(a, .after = c) %>% head()
```

```
## # A tibble: 1 x 6
##       b     c     a d     e     f
##   <dbl> <dbl> <dbl> <chr> <chr> <chr>
## 1     1     1     1 a     a     a
```

```r
df %>% relocate(f, .before = b) %>% head()
```

```
## # A tibble: 1 x 6
##       a f         b     c d     e
##   <dbl> <chr> <dbl> <dbl> <chr> <chr>
## 1     1 a         1     1 a     a
```

7.5.2 移至最后一列

```r
df %>% relocate(a, .after = last_col()) %>% head()
```

```
## # A tibble: 1 x 6
##       b     c d     e     f         a
##   <dbl> <dbl> <chr> <chr> <chr> <dbl>
## 1     1     1 a     a     a         1
```

```r
df %>% relocate(ff = f) #更改列名
```

7.5.3 选择所有字符列

```r
df %>% relocate(where(is.character)) %>% head()
```

```
## # A tibble: 1 x 6
##   d     e     f         a     b     c
##   <chr> <chr> <chr> <dbl> <dbl> <dbl>
## 1 a     a     a         1     1     1
```

7.5.4 选择所有数字列

```r
df %>% relocate(where(is.numeric), .after = last_col()) %>% head()
```

```
## # A tibble: 1 x 6
##   d     e     f         a     b     c
```

```
##   <chr> <chr> <chr> <dbl> <dbl> <dbl>
## 1 a     a     a         1     1     1
```

7.6 dorp_na()删除缺失值

- drop_na

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

df
```

```
## # A tibble: 3 x 2
##       x y
##   <dbl> <chr>
## 1     1 a
## 2     2 <NA>
## 3    NA b
```

```
df %>% drop_na() %>% head()
```

```
## # A tibble: 1 x 2
##       x y
##   <dbl> <chr>
## 1     1 a
```

```
df %>% drop_na(x) %>% head()
```

```
## # A tibble: 2 x 2
##       x y
##   <dbl> <chr>
## 1     1 a
## 2     2 <NA>
```

7.7 pull()提取单列

- pull

pull( )与$相似，在管道中使用pull更加优雅

```
iris %>% as_tibble() %>%
  mutate(mean = rowMeans(across(where(is.numeric)))) %>%
  pull(mean) %>% head()
```

```
## [1] 2.550 2.375 2.350 2.350 2.550 2.850
```

7.7.1 点过滤

不使用pull函数称为点过滤

```
iris %>% as_tibble() %>%
  mutate(mean = rowMeans(across(where(is.numeric)))) %>%
  .$mean %>% head()
```

```
## [1] 2.550 2.375 2.350 2.350 2.550 2.850
```

8. tidyverse中的专职函数(中)

8.1 unite多列合并

- unite 多列合并为1列

unite 函数有5个参数

- col 设置合并后列的名称
- sep 指定分隔符
- remove 如果为TRUE，则从输出数据框中删除输入列
- na.rm 如果为TRUE，则在合并数据之前删除缺失值

```
iris %>% as_tibble() %>%
  unite(.,col="Sepal",Sepal.Length:Sepal.Width,sep="|",
        remove = T,na.rm = F) %>% head()
```

```
## # A tibble: 6 x 4
##   Sepal   Petal.Length Petal.Width Species
##   <chr>          <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
```

上面的案例我们将2列合并成了1列，那么如果我们的数据有20列也需要2列合并为一列，此时该如何操作，请继续往下看

8.1.1 unite结合for循环合并多列

```
data <- iris %>% select_if(is.numeric)

result <- list(); b <- 2
for (i in 1:(data %>% ncol() / b)) {
  result[[i]] <- data %>%
    select((b * i - b + 1):(b * i) %>% all_of()) %>%
    unite(.,A,sep="_",remove = T,na.rm = F)
}

df <- result %>% as.data.frame() %>%
  set_colnames(c("Sepal","Petal")) %>% as_tibble()

df %>% head()
```

```
## # A tibble: 6 x 2
##   Sepal   Petal
##   <chr>   <chr>
## 1 5.1_3.5 1.4_0.2
## 2 4.9_3   1.4_0.2
## 3 4.7_3.2 1.3_0.2
## 4 4.6_3.1 1.5_0.2
## 5 5_3.6   1.4_0.2
## 6 5.4_3.9 1.7_0.4
```

通过上面的for循环操作我们完成了数据的合并，这段代码还有别的用处，以后再一一介绍

8.2 sperate拆分列

- separate 1列变多列

```
df %>% separate(`Sepal`,into=c("A","B"),sep="_") %>% head()
```

```
## # A tibble: 6 x 3
##   A     B     Petal
##   <chr> <chr> <chr>
## 1 5.1   3.5   1.4_0.2
## 2 4.9   3     1.4_0.2
## 3 4.7   3.2   1.3_0.2
## 4 4.6   3.1   1.5_0.2
## 5 5     3.6   1.4_0.2
## 6 5.4   3.9   1.7_0.4
```

8.2.1 只取分隔符前的列

```
df %>% separate(`Sepal`,into="Sepal",sep="_") %>% head()
```

```
## Warning: Expected 1 pieces. Additional pieces discarded in 150 rows [1, 2, 3, 4,
## 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, ...].
```

```
## # A tibble: 6 x 2
##   Sepal Petal
##   <chr> <chr>
## 1 5.1   1.4_0.2
## 2 4.9   1.4_0.2
## 3 4.7   1.3_0.2
## 4 4.6   1.5_0.2
## 5 5     1.4_0.2
## 6 5.4   1.7_0.4
```

上面的只是一般的情况，实际中肯定有更复杂的需求，试想如果有多列需要拆分那该如果处理，请继续往下看

8.2.2 多列进行拆分

```
dt <- data.frame(Group = c(rep("A",5),rep("B",5)),
                 A1 = c(rep("A1_B",5),rep("B1_C",5)),
                 A2 = c(rep("A2_D",5),rep("B2_A",5)))

dt %>% as_tibble() %>% head()
```

```
## # A tibble: 6 x 3
##   Group A1    A2
##   <chr> <chr> <chr>
## 1 A     A1_B  A2_D
## 2 A     A1_B  A2_D
## 3 A     A1_B  A2_D
## 4 A     A1_B  A2_D
## 5 A     A1_B  A2_D
## 6 B     B1_C  B2_A
```

```
dt %>%
  mutate_at(vars(-Group),~str_split(.,"_",simplify=TRUE)[,2]) %>% head()
```

```
##   Group A1 A2
## 1     A  B  D
```

```
## 2     A  B  D
## 3     A  B  D
## 4     A  B  D
## 5     A  B  D
## 6     B  C  A
```

8.3 sparate_rows将折叠的行展开

- separate_rows 将折叠行展开

案例1

```
df <- tibble(
  x = 1:3,y = c("a", "d_e_f", "g_h"),
  z = c("1", "2", "5"))

df
```

```
## # A tibble: 3 x 3
##       x y     z
##   <int> <chr> <chr>
## 1     1 a     1
## 2     2 d_e_f 2
## 3     3 g_h   5
```

```
separate_rows(df,y, z,convert=TRUE,sep="_")
```

```
## # A tibble: 6 x 3
##       x y         z
##   <int> <chr> <int>
## 1     1 a         1
## 2     2 d         2
## 3     2 e         2
## 4     2 f         2
## 5     3 g         5
## 6     3 h         5
```

案例2

```
d <- data.frame(a=c(1:3),
                b=c("name1,name2,name3","name4","name5,name6"),
                c=c("name7","name8,name9","name10"))
d
```

```
##   a                b           c
## 1 1 name1,name2,name3       name7
## 2 2             name4 name8,name9
## 3 3       name5,name6      name10
```

```
d %>% separate_rows(b) %>% separate_rows(c)
```

```
## # A tibble: 7 x 3
##       a b     c
##   <int> <chr> <chr>
## 1     1 name1 name7
## 2     1 name2 name7
## 3     1 name3 name7
```

```
## 4     2 name4 name8
## 5     2 name4 name9
## 6     3 name5 name10
## 7     3 name6 name10
```

8.3.1 for循环展开多列

```
cols <- c("b", "c")
for(col in cols) {
  d <- separate_rows_(d, col)
}

d
```

```
## # A tibble: 7 x 3
##       a b     c
##   <int> <chr> <chr>
## 1     1 name1 name7
## 2     1 name2 name7
## 3     1 name3 name7
## 4     2 name4 name8
## 5     2 name4 name9
## 6     3 name5 name10
## 7     3 name6 name10
```

另外一种展开折叠行的方法

```
df %>% rowwise() %>%
  mutate(y= y %>% strsplit(.,split = "_")) %>%
  unnest_longer(col = y) %>% ungroup()
```

```
## # A tibble: 6 x 3
##       x y     z
##   <int> <chr> <chr>
## 1     1 a     1
## 2     2 d     2
## 3     2 e     2
## 4     2 f     2
## 5     3 g     5
## 6     3 h     5
```

9. tidyverse中的专职函数(下)

9.1 bind_rows数据框纵向合并

- bind_rows 纵向合并数据框

```
one <- iris[1:4, ]
two <- iris[9:12, ]
two %>% bind_rows(one) %>% as_tibble()
```

```
## # A tibble: 8 x 5
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##          <dbl>       <dbl>        <dbl>       <dbl> <fct>
## 1          4.4         2.9          1.4         0.2 setosa
## 2          4.9         3.1          1.5         0.1 setosa
## 3          5.4         3.7          1.5         0.2 setosa
## 4          4.8         3.4          1.6         0.2 setosa
## 5          5.1         3.5          1.4         0.2 setosa
```

```
## 6          4.9         3           1.4         0.2 setosa
## 7          4.7         3.2         1.3         0.2 setosa
## 8          4.6         3.1         1.5         0.2 setosa
```

```
bind_rows(list(one, two), .id = "id")
```

```
##   id Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1  1          5.1         3.5          1.4         0.2  setosa
## 2  1          4.9         3.0          1.4         0.2  setosa
## 3  1          4.7         3.2          1.3         0.2  setosa
## 4  1          4.6         3.1          1.5         0.2  setosa
## 5  2          4.4         2.9          1.4         0.2  setosa
## 6  2          4.9         3.1          1.5         0.1  setosa
## 7  2          5.4         3.7          1.5         0.2  setosa
## 8  2          4.8         3.4          1.6         0.2  setosa
```

9.2 bind_cols横向合并数据框

- bind_cols 横向合并数据框

```
bind_cols(a = 1:3, b = 4:6)
```

```
tab_1 <- mtcars[,1:3]
tab_2 <- mtcars[,4:6]
tab_3 <- mtcars[,7:8]
bind_cols(tab_1, tab_2, tab_3)
```

9.3 inner_join 内部连接

- left_join 数据框连接

合并两个数据集的变量，但仅保留具有共同ID的行

```
tab1 <- mpg %>% select(2,3)

tab1
```

```
## # A tibble: 234 x 2
##    model      displ
##    <chr>      <dbl>
##  1 a4           1.8
##  2 a4           1.8
##  3 a4           2
##  4 a4           2
##  5 a4           2.8
##  6 a4           2.8
##  7 a4           3.1
##  8 a4 quattro   1.8
##  9 a4 quattro   1.8
## 10 a4 quattro   2
## # ... with 224 more rows
```

```
tab2 <- mpg %>% select(2,4)

tab2
```

```
## # A tibble: 234 x 2
##    model       year
##    <chr>      <int>
##  1 a4          1999
##  2 a4          1999
##  3 a4          2008
##  4 a4          2008
##  5 a4          1999
##  6 a4          1999
##  7 a4          2008
##  8 a4 quattro  1999
##  9 a4 quattro  1999
## 10 a4 quattro  2008
## # ... with 224 more rows
```

```
inner_join(tab1,tab2,by="model")
```

```
## # A tibble: 1,612 x 3
##    model displ  year
##    <chr> <dbl> <int>
##  1 a4      1.8  1999
##  2 a4      1.8  1999
##  3 a4      1.8  2008
##  4 a4      1.8  2008
##  5 a4      1.8  1999
##  6 a4      1.8  1999
##  7 a4      1.8  2008
##  8 a4      1.8  1999
##  9 a4      1.8  1999
## 10 a4      1.8  2008
## # ... with 1,602 more rows
```

left_join保留与左数据表匹配的行

```
left_join(tab1,tab2,by="model")
```

```
## # A tibble: 1,612 x 3
##    model displ  year
##    <chr> <dbl> <int>
##  1 a4      1.8  1999
##  2 a4      1.8  1999
##  3 a4      1.8  2008
##  4 a4      1.8  2008
##  5 a4      1.8  1999
##  6 a4      1.8  1999
##  7 a4      1.8  2008
##  8 a4      1.8  1999
##  9 a4      1.8  1999
## 10 a4      1.8  2008
## # ... with 1,602 more rows
```

setequal

比较2组数据是否相同

```
setequal(1:5, 1:6)
```

```
## [1] FALSE
```

```
setequal(tab1,tab2) %>% head()
```

```
## [1] FALSE
```

10. across()列处理函数

across作为`tidyverse`中的新生函数，需要更新至 **dplyr-1-0-0**才能使用

across()它可以轻松地对多列执行相同的操作

across() 有两个主要参数： * 第一个参数.cols选择要操作的列 * 第二个参数.fns是要应用于每一列的一个函数或函数列表

创建数据

```
gdf <- tibble(g = c(1,1,2,3),v1 = 10:13,v2 = 20:23,v3=1:4)
gdf %>% head()
```

```
## # A tibble: 4 x 4
##       g    v1    v2    v3
##   <dbl> <int> <int> <int>
## 1     1    10    20     1
## 2     1    11    21     2
## 3     2    12    22     3
## 4     3    13    23     4
```

10.1 给每一列加1

```
gdf %>% mutate(across(v1:v3, ~ .x +1)) %>% head()
```

```
## # A tibble: 4 x 4
##       g    v1    v2    v3
##   <dbl> <dbl> <dbl> <dbl>
## 1     1    11    21     2
## 2     1    12    22     3
## 3     2    13    23     4
## 4     3    14    24     5
```

10.2 前两列四舍五入

```
iris %>% as_tibble() %>%
  mutate(across(c(1,2),round)) %>% head()
```

```
## # A tibble: 6 x 5
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##          <dbl>       <dbl>        <dbl>       <dbl> <fct>
## 1            5           4          1.4         0.2 setosa
## 2            5           3          1.4         0.2 setosa
## 3            5           3          1.3         0.2 setosa
## 4            5           3          1.5         0.2 setosa
## 5            5           4          1.4         0.2 setosa
## 6            5           4          1.7         0.4 setosa
```

还有如下2种写法

```
iris %>% as_tibble() %>%
  mutate(across(1:Sepal.Width, round)) %>% head()
```

```
## # A tibble: 6 x 5
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##          <dbl>       <dbl>        <dbl>       <dbl> <fct>
## 1            5           4          1.4         0.2 setosa
## 2            5           3          1.4         0.2 setosa
## 3            5           3          1.3         0.2 setosa
## 4            5           3          1.5         0.2 setosa
## 5            5           4          1.4         0.2 setosa
## 6            5           4          1.7         0.4 setosa
```

```
iris %>% as_tibble() %>%
  mutate(across(where(is.double) & !c(Petal.Length, Petal.Width), round))
```

```
## # A tibble: 150 x 5
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##           <dbl>       <dbl>        <dbl>       <dbl> <fct>
##  1            5           4          1.4         0.2 setosa
##  2            5           3          1.4         0.2 setosa
##  3            5           3          1.3         0.2 setosa
##  4            5           3          1.5         0.2 setosa
##  5            5           4          1.4         0.2 setosa
##  6            5           4          1.7         0.4 setosa
##  7            5           3          1.4         0.3 setosa
##  8            5           3          1.5         0.2 setosa
##  9            4           3          1.4         0.2 setosa
## 10            5           3          1.5         0.1 setosa
## # ... with 140 more rows
```

10.3 按列求均值

```
iris %>% summarize(across(is.numeric,mean)) %>% head()
```

```
## Warning: Predicate functions must be wrapped in `where()`.
##
##   # Bad
##   data %>% select(is.numeric)
##
##   # Good
##   data %>% select(where(is.numeric))
##
## i Please update your code.
## This message is displayed once per session.
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1     5.843333    3.057333        3.758    1.199333
```

10.4 按行求和

```
iris %>% as_tibble() %>% rowwise() %>%
  mutate(mean = sum(across(where(is.numeric)))) %>% head()
```

```
## # A tibble: 6 x 6
## # Rowwise:
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species  mean
##          <dbl>       <dbl>        <dbl>       <dbl> <fct>   <dbl>
## 1          5.1         3.5          1.4         0.2 setosa   10.2
## 2          4.9         3            1.4         0.2 setosa    9.5
## 3          4.7         3.2          1.3         0.2 setosa    9.4
## 4          4.6         3.1          1.5         0.2 setosa    9.4
## 5          5           3.6          1.4         0.2 setosa   10.2
## 6          5.4         3.9          1.7         0.4 setosa   11.4
```

10.5 分组求均值

```
iris %>%
  group_by(Species) %>%
  summarise(across(starts_with("Sepal"), ~ mean(.x, na.rm = TRUE))) %>% head()
```

```
## # 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
```

10.6 分组求和

```
iris %>%
  group_by(Species) %>%
  summarise(across(starts_with("Sepal"), ~ sum(.x, na.rm=TRUE))) %>% head()
```

```
## # A tibble: 3 x 3
##   Species    Sepal.Length Sepal.Width
##   <fct>             <dbl>       <dbl>
## 1 setosa             250.        171.
## 2 versicolor         297.        138.
## 3 virginica          329.        149.
```

```
iris %>%
  group_by(Species) %>%
  summarise(across(starts_with("Sepal"), list(mean = mean, sd = sd))) %>% head()
```

```
## # A tibble: 3 x 5
##   Species    Sepal.Length_mean Sepal.Length_sd Sepal.Width_mean Sepal.Width_sd
##   <fct>                  <dbl>           <dbl>            <dbl>          <dbl>
## 1 setosa                  5.01           0.352             3.43          0.379
## 2 versicolor              5.94           0.516             2.77          0.314
## 3 virginica               6.59           0.636             2.97          0.322
```

10.7 使用.name参数控制输出名

```
iris %>%
  group_by(Species) %>%
  summarise(across(starts_with("Sepal"), mean, .names = "mean_{.col}")) %>% head()
```

```
## # A tibble: 3 x 3
##   Species    mean_Sepal.Length mean_Sepal.Width
##   <fct>                  <dbl>            <dbl>
## 1 setosa                  5.01             3.43
```

```
## 2 versicolor              5.94            2.77
## 3 virginica               6.59            2.97
```

10.8 筛选没有缺失值的行

```
starwars %>% filter(across(everything(), ~ !is.na(.x))) %>% head()
```

```
## # A tibble: 6 x 14
##   name     height  mass hair_color  skin_color eye_color birth_year sex   gender
##   <chr>     <int> <dbl> <chr>       <chr>      <chr>          <dbl> <chr> <chr>
## 1 Luke Sk~    172    77 blond       fair       blue            19   male  mascu~
## 2 Darth V~    202   136 none        white      yellow          41.9 male  mascu~
## 3 Leia Or~    150    49 brown       light      brown           19   fema~ femin~
## 4 Owen La~    178   120 brown, grey light      blue            52   male  mascu~
## 5 Beru Wh~    165    75 brown       light      blue            47   fema~ femin~
## 6 Biggs D~    183    84 black       light      brown           24   male  mascu~
## # ... with 5 more variables: homeworld <chr>, species <chr>, films <list>,
## #   vehicles <list>, starships <list>
```

使用时mutate()，所有转换across()都将立即应用

```
df <- tibble(x = 2, y = 4, z = 8)
df %>% mutate(across(everything(), ~ .x / y))
```

10.9 统计字符长度

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

```
## # A tibble: 1 x 8
##    name hair_color skin_color eye_color   sex gender homeworld species
##   <int>      <int>      <int>     <int> <int>  <int>     <int>   <int>
## 1    87         13         31        15     5      3        49      38
```

10.10 统计列最小/最大值

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

```
##   Sepal.Length_min Sepal.Length_max Sepal.Width_min Sepal.Width_max
## 1              4.3              7.9               2             4.4
##   Petal.Length_min Petal.Length_max Petal.Width_min Petal.Width_max
## 1                1              6.9             0.1             2.5
```

11. summarise汇总数据

11.1 count()

count 统计观察次数

```
msleep %>% count(order, sort = TRUE) %>% head()
```

```
## # A tibble: 6 x 2
##   order           n
##   <chr>       <int>
## 1 Rodentia       22
## 2 Carnivora      12
## 3 Primates       12
## 4 Artiodactyla    6
## 5 Soricomorpha    5
## 6 Cetacea         3
```

也可以在一个count()语句中添加多个变量

```
msleep %>% count(order, vore, sort = TRUE) %>% head()
```

```
## # A tibble: 6 x 3
##   order          vore      n
##   <chr>          <chr> <int>
## 1 Rodentia       herbi    16
## 2 Carnivora      carni    12
## 3 Primates       omni     10
## 4 Artiodactyla   herbi     5
## 5 Cetacea        carni     3
## 6 Perissodactyla herbi     3
```

11.2 summarize()

dplyr 中的summarize函数使用直观易读的代码对统计数据进行汇总

```
msleep %>%
  summarise(n = n(), average = mean(sleep_total), maximum = max(sleep_total))
```

```
## # A tibble: 1 x 3
##       n average maximum
##   <int>   <dbl>   <dbl>
## 1    83    10.4    19.9
```

11.3 group_by( )按分组进行汇总

```
msleep %>%
  group_by(vore) %>%
  summarise(n = n(), average = mean(sleep_total), maximum = max(sleep_total))
```

summarise( )几乎适用于任何聚合函数，并允许进行额外的算术运算： * n() - 给出观察次数 * n_distinct(var) - 给出唯一值的数量 var * sum(var), max(var), min(var), ... * mean(var), median(var), sd(var), IQR(var)

将平均 sleep_total 并除以 24，以获得一天的睡眠量

```
msleep %>%
  group_by(vore) %>%
  summarise(avg_sleep_day = mean(sleep_total)/24)
```

```
## # A tibble: 5 x 2
##   vore    avg_sleep_day
##   <chr>           <dbl>
## 1 carni           0.432
## 2 herbi           0.396
## 3 insecti         0.622
```

```
## 4 omni            0.455
## 5 <NA>            0.424
```

11.4 summarise_all()

summarise_all()需要一个函数作为参数,它将应用于所有列；示例代码计算每列的平均值

```
msleep %>%
  group_by(vore) %>%
  summarise_all(mean, na.rm=TRUE)
```

```
## # A tibble: 5 x 11
##   vore     name genus order conservation sleep_total sleep_rem sleep_cycle awake
##   <chr>   <dbl> <dbl> <dbl>        <dbl>       <dbl>     <dbl>       <dbl> <dbl>
## 1 carni      NA    NA    NA           NA       10.4       2.29       0.373 13.6
## 2 herbi      NA    NA    NA           NA        9.51      1.37       0.418 14.5
## 3 insecti    NA    NA    NA           NA       14.9       3.52       0.161  9.06
## 4 omni       NA    NA    NA           NA       10.9       1.96       0.592 13.1
## 5 <NA>       NA    NA    NA           NA       10.2       1.88       0.183 13.8
## # ... with 2 more variables: brainwt <dbl>, bodywt <dbl>
```

给每列的值加5

```
msleep %>%
  group_by(vore) %>%
  summarise_all(~mean(., na.rm = TRUE) + 5)
```

```
## # A tibble: 5 x 11
##   vore     name genus order conservation sleep_total sleep_rem sleep_cycle awake
##   <chr>   <dbl> <dbl> <dbl>        <dbl>       <dbl>     <dbl>       <dbl> <dbl>
## 1 carni      NA    NA    NA           NA        15.4       7.29        5.37  18.6
## 2 herbi      NA    NA    NA           NA        14.5       6.37        5.42  19.5
## 3 insecti    NA    NA    NA           NA        19.9       8.52        5.16  14.1
## 4 omni       NA    NA    NA           NA        15.9       6.96        5.59  18.1
## 5 <NA>       NA    NA    NA           NA        15.2       6.88        5.18  18.8
## # ... with 2 more variables: brainwt <dbl>, bodywt <dbl>
```

11.5 summarise_if()

计算所有数字列的平均值

```
msleep %>%
  group_by(vore) %>%
  summarise_if(is.numeric, mean, na.rm=TRUE)
```

```
## # A tibble: 5 x 7
##   vore    sleep_total sleep_rem sleep_cycle awake brainwt  bodywt
##   <chr>         <dbl>     <dbl>       <dbl> <dbl>   <dbl>   <dbl>
## 1 carni         10.4       2.29       0.373 13.6  0.0793   90.8
## 2 herbi          9.51      1.37       0.418 14.5  0.622   367.
## 3 insecti       14.9       3.52       0.161  9.06 0.0216   12.9
## 4 omni          10.9       1.96       0.592 13.1  0.146    12.7
## 5 <NA>          10.2       1.88       0.183 13.8  0.00763   0.858
```

11.6 rename_if( )对列进行重命名

```
msleep %>%
  group_by(vore) %>%
```

```
  summarise_if(is.numeric, mean, na.rm=TRUE) %>%
  rename_if(is.numeric, ~paste0("avg_", .))
```

```
## # A tibble: 5 x 7
##   vore    avg_sleep_total avg_sleep_rem avg_sleep_cycle avg_awake avg_brainwt
##   <chr>             <dbl>         <dbl>           <dbl>     <dbl>       <dbl>
## 1 carni              10.4          2.29           0.373     13.6      0.0793
## 2 herbi               9.51         1.37           0.418     14.5      0.622
## 3 insecti            14.9          3.52           0.161      9.06     0.0216
## 4 omni               10.9          1.96           0.592     13.1      0.146
## 5 <NA>               10.2          1.88           0.183     13.8      0.00763
## # ... with 1 more variable: avg_bodywt <dbl>
```

11.7 summarise_at()

下面的代码将返回平均含有单词"睡眠"的所有列，并且还它们重命名为"AVG_ VAR"

```
msleep %>%
  group_by(vore) %>%
  summarise_at(vars(contains("sleep")), mean, na.rm=TRUE) %>%
  rename_at(vars(contains("sleep")), ~paste0("avg_", .))
```

```
## # A tibble: 5 x 4
##   vore    avg_sleep_total avg_sleep_rem avg_sleep_cycle
##   <chr>             <dbl>         <dbl>           <dbl>
## 1 carni              10.4          2.29           0.373
## 2 herbi               9.51         1.37           0.418
## 3 insecti            14.9          3.52           0.161
## 4 omni               10.9          1.96           0.592
## 5 <NA>               10.2          1.88           0.183
```

11.8 top_n( )

保留值最高的5个

```
msleep %>%
  group_by(order) %>%
  summarise(average = mean(sleep_total)) %>%
  top_n(5)
```

```
## Selecting by average
```

```
## # A tibble: 5 x 2
##   order           average
##   <chr>             <dbl>
## 1 Afrosoricida       15.6
## 2 Chiroptera         19.8
## 3 Cingulata          17.8
## 4 Didelphimorphia    18.7
## 5 Pilosa             14.4
```

保留值最低的5个

```
msleep %>%
  group_by(order) %>%
  summarise(average = mean(sleep_total)) %>%
  top_n(-5)
```

```
## Selecting by average
```

```
## # A tibble: 5 x 2
##   order          average
##   <chr>            <dbl>
## 1 Artiodactyla      4.52
## 2 Cetacea           4.5
## 3 Hyracoidea        5.67
## 4 Perissodactyla    3.47
## 5 Proboscidea       3.6
```

示例代码将保留average_sleep 的5 个最高值

```
msleep %>%
  group_by(order) %>%
  summarise(average_sleep = mean(sleep_total), max_sleep = max(sleep_total)) %>%
  top_n(5, average_sleep)
```

```
## # A tibble: 5 x 3
##   order           average_sleep max_sleep
##   <chr>                   <dbl>     <dbl>
## 1 Afrosoricida             15.6      15.6
## 2 Chiroptera               19.8      19.9
## 3 Cingulata                17.8      18.1
## 4 Didelphimorphia          18.7      19.4
## 5 Pilosa                   14.4      14.4
```

11.9 sample_frac()

sample_frac()允许随机选择一部分行（此处为 10%）

```
msleep %>% sample_frac(.1)
```

```
## # A tibble: 8 x 11
##   name    genus vore  order conservation sleep_total sleep_rem sleep_cycle awake
##   <chr>   <chr> <chr> <chr> <chr>              <dbl>     <dbl>       <dbl> <dbl>
## 1 Big br~ Epte~ inse~ Chir~ lc                  19.7       3.9       0.117   4.3
## 2 Africa~ Loxo~ herbi Prob~ vu                   3.3      NA        NA      20.7
## 3 Potto   Pero~ omni  Prim~ lc                  11        NA        NA      13
## 4 Easter~ Tami~ herbi Rode~ <NA>                15.8      NA        NA       8.2
## 5 Human   Homo  omni  Prim~ <NA>                 8         1.9       1.5    16
## 6 North ~ Dide~ omni  Dide~ lc                  18         4.9       0.333   6
## 7 Africa~ Rhab~ omni  Rode~ <NA>                 8.7      NA        NA      15.3
## 8 Tenrec  Tenr~ omni  Afro~ <NA>                15.6       2.3      NA       8.4
## # ... with 2 more variables: brainwt <dbl>, bodywt <dbl>
```