Week 6

Melanie Beebe

apply() family of functions

call in Lahman Battting data

```
#code in lecture but doesn't work
library(Lahman)
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.1     v tibble    3.2.1
v lubridate 1.9.3     v tidyr     1.3.1
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becor
```

```
my_batting <-
  Batting[, c("playerID", "teamID", "G", "AB", "R", "H", "X2B", "X3B", "HR")] |>
  as_tibble()
my_batting
```

```
# A tibble: 112,184 x 9
   playerID  teamID     G    AB     R     H   X2B   X3B    HR
   <chr>     <fct>  <int> <int> <int> <int> <int> <int> <int>
 1 abercda01 TRO        1     4     0     0     0     0     0
 2 addybo01  RC1       25   118    30    32     6     0     0
```

```
 3 allisar01 CL1       29   137    28    40     4     5     0
 4 allisdo01 WS3       27   133    28    44    10     2     2
 5 ansonca01 RC1       25   120    29    39    11     3     0
 6 armstbo01 FW1       12    49     9    11     2     1     0
 7 barkeal01 RC1        1     4     0     1     0     0     0
 8 barnero01 BS1       31   157    66    63    10     9     0
 9 barrebi01 FW1        1     5     1     1     1     0     0
10 barrofr01 BS1       18    86    13    13     2     1     0
# i 112,174 more rows
```

```
library(Lahman)
library(tidyverse)
my_batting <-
  Batting |>
  select (playerID, teamID, G, AB, R, H, X2B, X3B, HR) |>
  as_tibble()
my_batting
```

```
# A tibble: 112,184 x 9
   playerID  teamID     G    AB     R     H   X2B   X3B    HR
   <chr>     <fct>  <int> <int> <int> <int> <int> <int> <int>
 1 abercda01 TRO        1     4     0     0     0     0     0
 2 addybo01  RC1       25   118    30    32     6     0     0
 3 allisar01 CL1       29   137    28    40     4     5     0
 4 allisdo01 WS3       27   133    28    44    10     2     2
 5 ansonca01 RC1       25   120    29    39    11     3     0
 6 armstbo01 FW1       12    49     9    11     2     1     0
 7 barkeal01 RC1        1     4     0     1     0     0     0
 8 barnero01 BS1       31   157    66    63    10     9     0
 9 barrebi01 FW1        1     5     1     1     1     0     0
10 barrofr01 BS1       18    86    13    13     2     1     0
# i 112,174 more rows
```

use apply

```
#NOTE the capital X
apply(X = my_batting,
      MARGIN = 2,
      FUN = summary,
      na.rm = TRUE)
```

```
       playerID    teamID      G           AB          R           H
Length "112184"    "112184"    "112184"    "112184"    "112184"    "112184"
Class  "character" "character" "character" "character" "character" "character"
Mode   "character" "character" "character" "character" "character" "character"
       X2B         X3B         HR
Length "112184"    "112184"    "112184"
Class  "character" "character" "character"
Mode   "character" "character" "character"
```

Above is not useful because we have multiple types of data, so R coerces all into character. Instead, use as.numeric to get a meaningful summary.

```
batting_summary <- apply(X = my_batting |>
                           select(where(is.numeric)),
                         MARGIN = 2,
                         FUN = summary,
                         na.rm = TRUE)
batting_summary
```

```
                G       AB         R       H     X2B       X3B         HR
Min.      1.00000   0.0000   0.00000   0.000  0.0000  0.000000  0.000000
1st Qu.  12.00000   3.0000   0.00000   0.000  0.0000  0.000000  0.000000
Median   34.00000  45.0000   4.00000   8.000  1.0000  0.000000  0.000000
Mean     50.47547 137.9281  18.30589  35.993  6.1552  1.221048  2.863367
3rd Qu.  78.00000 221.0000  26.00000  55.000  9.0000  1.000000  2.000000
Max.    165.00000 716.0000 198.00000 262.000 67.0000 36.000000 73.000000
```

custom functions with the apply family (anonymous/lambda functions)

```
custom_batting_summary <- apply(X = my_batting |>
 select(where(is.numeric)),
 MARGIN = 2,
 FUN = function(x){
 temp <- c(mean(x), sd(x))
 names(temp) <- c("mean", "sd")
 temp
 }
)
custom_batting_summary
```

```
            G       AB        R        H      X2B      X3B       HR
mean 50.47547 137.9281 18.30589 35.99300 6.155200 1.221048 2.863367
sd   46.77938 183.1120 27.92615 51.95348 9.604293 2.566017 6.391064
```

other arguments can be provided in anonymous functions

Here, trim = 0.1 is passed as an argument to the anonynmous function through apply

```
custom_batting_summary <- apply(X = my_batting |>
 select(where(is.numeric)),
 MARGIN = 2,
 FUN = function(x, trim){
 temp <- c(mean(x, trim), sd(x))
 names(temp) <- c("mean", "sd")
return(temp)        #can use return or just say temp
 },
 trim = 0.1
)
custom_batting_summary
```

```
            G       AB        R        H      X2B       X3B       HR
mean 44.43402 103.6474 12.12291 25.27238 4.008490 0.5717342 1.175737
sd   46.77938 183.1120 27.92615 51.95348 9.604293 2.5660168 6.391064
```

lapply

```
# generate list
set.seed(10)
my_list <- list(rnorm(100), runif(10), rgamma(40, shape = 1, rate = 1))
my_list
```

```
[[1]]
  [1]  0.01874617 -0.18425254 -1.37133055 -0.59916772  0.29454513  0.38979430
  [7] -1.20807618 -0.36367602 -1.62667268 -0.25647839  1.10177950  0.75578151
 [13] -0.23823356  0.98744470  0.74139013  0.08934727 -0.95494386 -0.19515038
 [19]  0.92552126  0.48297852 -0.59631064 -2.18528684 -0.67486594 -2.11906119
 [25] -1.26519802 -0.37366156 -0.68755543 -0.87215883 -0.10176101 -0.25378053
 [31] -1.85374045 -0.07794607  0.96856634  0.18492596 -1.37994358 -1.43551436
 [37]  0.36208723 -1.75908675 -0.32454401 -0.65156299  1.08655140 -0.76254488
```

```
 [43] -0.82866254  0.83447390 -0.96765199 -0.02881534  0.23252515 -0.30120868
 [49] -0.67761458  0.65522764 -0.40063755 -0.33455657  1.36795395  2.13776710
 [55]  0.50581926  0.78634238 -0.90221194  0.53289699 -0.64589425  0.29098749
 [61] -1.23759447 -0.45617628 -0.83032265  0.34011564  1.06637640  1.21612584
 [67]  0.73569066 -0.48120862  0.56274476 -1.24631971  0.38092221 -1.43042725
 [73] -1.04844550 -0.21850355 -1.48993624  1.17270628 -1.47982702 -0.43038782
 [79] -1.05163864  1.52258634  0.59282805 -0.22266151  0.71289428  0.71660083
 [85]  0.44024186  0.15883062  0.65976414  2.22051966 -1.18394507 -0.07395583
 [91] -0.41635467 -0.19148234  0.06954478  1.15534832  0.59495735 -1.41964511
 [97] -1.60667725  0.89292590  0.14816796  1.22702839

[[2]]
 [1] 0.2230884 0.5358950 0.6625291 0.8480705 0.1491831 0.6700994 0.7616357
 [8] 0.9986345 0.2632973 0.8851860

[[3]]
 [1] 0.97751605 1.07661402 0.53668935 0.30164994 0.31950172 1.50569731
 [7] 1.16219155 1.78190251 1.39846297 0.21755133 0.41221634 0.01073490
[13] 0.19436361 1.97454198 1.65336544 2.67144283 3.83048886 0.41298794
[19] 0.56358677 0.26168819 0.67036627 0.96368153 0.30989614 1.52032718
[25] 2.58631303 0.65429193 0.08615201 0.63838299 1.01594682 2.52592172
[31] 0.30547757 1.31577612 1.02134303 0.18546398 4.16151197 2.45387334
[37] 0.29014960 0.71864545 1.10831509 0.53331983
```

```
lapply(X = my_list, FUN = mean) #takes mean of each list
```

```
[[1]]
[1] -0.1365489

[[2]]
[1] 0.5997619

[[3]]
[1] 1.108209
```

```
lapply(X = my_list, FUN = mean)
```

```
[[1]]
[1] -0.1365489

[[2]]
```

```
[1] 0.5997619

[[3]]
[1] 1.108209
```

can pass additional arguments

```
lapply(X = my_list, FUN = mean, trim = 0.1, na.rm = TRUE)
```

```
[[1]]
[1] -0.1359629

[[2]]
[1] 0.6062252

[[3]]
[1] 0.9563087
```

sapply (simplify the result)

```
#simplifies list to vector
sapply(X = my_list, FUN = mean)
```

```
[1] -0.1365489  0.5997619  1.1082087
```

```
is.vector(sapply(X = my_list, FUN = mean))
```

```
[1] TRUE
```

purrr, a cleaner version of apply

map() is an lapply() type function so it returns a list

use map to get means

```
map(my_list, mean)  #equivalent to lapply(X = my_list, FUN = mean, trim = 0.1, na.rm = TRUE)
```

```
[[1]]
[1] -0.1365489

[[2]]
[1] 0.5997619

[[3]]
[1] 1.108209
```

other ways to use map

```
#grap second element from each list element (see my_list printing)
map(my_list, 2)
```

```
[[1]]
[1] -0.1842525

[[2]]
[1] 0.535895

[[3]]
[1] 1.076614
```

```
#alt code using lapply
lapply(my_list, function(x) x[[2]])
```

```
[[1]]
[1] -0.1842525

[[2]]
[1] 0.535895

[[3]]
[1] 1.076614
```

```r
#OR
lapply(my_list, '[[' , 2)
```

```
[[1]]
[1] -0.1842525

[[2]]
[1] 0.535895

[[3]]
[1] 1.076614
```

Think of it like this: my_list <- list(list(1, “a”, TRUE), list(2, “b”, FALSE), list(3, “c”, TRUE)) result <- lapply(my_list, function(x) x[[2]])

- For the first list list(1, “a”, TRUE), x[[2]] is “a”.
- For the second list list(2, “b”, FALSE), x[[2]] is “b”.
- For the third list list(3, “c”, TRUE), x[[2]] is “c”.

purr allows us to use shorthand ways to make anonymous functions

\(x) = function(x), so \(x0) mean(x) = function(x) mean(x)

for example, result <- map(my_list, \(x) mean(x, trim = 0.1))

List columns

used to add a list to a list, ie iris has 150 observations, can add a column of additional data of 150 observations

```r
iris |>
 as_tibble() |>
 mutate(diffs = pmap(list(Sepal.Length, Sepal.Width, Petal.Length, Petal.Width),
 \(x, y, z, w) list(x-y, x-z, x-w))) |>
 select(diffs, everything())
```

```
# A tibble: 150 x 6
   diffs      Sepal.Length Sepal.Width Petal.Length Petal.Width Species
   <list>            <dbl>       <dbl>        <dbl>       <dbl> <fct>
```

```
 1 <list [3]>          5.1         3.5          1.4         0.2 setosa
 2 <list [3]>          4.9         3            1.4         0.2 setosa
 3 <list [3]>          4.7         3.2          1.3         0.2 setosa
 4 <list [3]>          4.6         3.1          1.5         0.2 setosa
 5 <list [3]>          5           3.6          1.4         0.2 setosa
 6 <list [3]>          5.4         3.9          1.7         0.4 setosa
 7 <list [3]>          4.6         3.4          1.4         0.3 setosa
 8 <list [3]>          5           3.4          1.5         0.2 setosa
 9 <list [3]>          4.4         2.9          1.4         0.2 setosa
10 <list [3]>          4.9         3.1          1.5         0.1 setosa
# i 140 more rows
```

view the diffs:

```
iris |>
 as_tibble() |>
 mutate(diffs = pmap(list(Sepal.Length, Sepal.Width, Petal.Length, Petal.Width),
 \(x, y, z, w) list(x-y, x-z, x-w))) |>
pull(diffs)
```

```
[[1]]
[[1]][[1]]
[1] 1.6

[[1]][[2]]
[1] 3.7

[[1]][[3]]
[1] 4.9


[[2]]
[[2]][[1]]
[1] 1.9

[[2]][[2]]
[1] 3.5

[[2]][[3]]
[1] 4.7
```

```
[[3]]
[[3]][[1]]
[1] 1.5

[[3]][[2]]
[1] 3.4

[[3]][[3]]
[1] 4.5


[[4]]
[[4]][[1]]
[1] 1.5

[[4]][[2]]
[1] 3.1

[[4]][[3]]
[1] 4.4


[[5]]
[[5]][[1]]
[1] 1.4

[[5]][[2]]
[1] 3.6

[[5]][[3]]
[1] 4.8


[[6]]
[[6]][[1]]
[1] 1.5

[[6]][[2]]
[1] 3.7

[[6]][[3]]
[1] 5
```

```
[[7]]
[[7]][[1]]
[1] 1.2

[[7]][[2]]
[1] 3.2

[[7]][[3]]
[1] 4.3


[[8]]
[[8]][[1]]
[1] 1.6

[[8]][[2]]
[1] 3.5

[[8]][[3]]
[1] 4.8


[[9]]
[[9]][[1]]
[1] 1.5

[[9]][[2]]
[1] 3

[[9]][[3]]
[1] 4.2


[[10]]
[[10]][[1]]
[1] 1.8

[[10]][[2]]
[1] 3.4

[[10]][[3]]
[1] 4.8
```

```
[[11]]
[[11]][[1]]
[1] 1.7

[[11]][[2]]
[1] 3.9

[[11]][[3]]
[1] 5.2


[[12]]
[[12]][[1]]
[1] 1.4

[[12]][[2]]
[1] 3.2

[[12]][[3]]
[1] 4.6


[[13]]
[[13]][[1]]
[1] 1.8

[[13]][[2]]
[1] 3.4

[[13]][[3]]
[1] 4.7


[[14]]
[[14]][[1]]
[1] 1.3

[[14]][[2]]
[1] 3.2

[[14]][[3]]
```

```
[1] 4.2


[[15]]
[[15]][[1]]
[1] 1.8

[[15]][[2]]
[1] 4.6

[[15]][[3]]
[1] 5.6


[[16]]
[[16]][[1]]
[1] 1.3

[[16]][[2]]
[1] 4.2

[[16]][[3]]
[1] 5.3


[[17]]
[[17]][[1]]
[1] 1.5

[[17]][[2]]
[1] 4.1

[[17]][[3]]
[1] 5


[[18]]
[[18]][[1]]
[1] 1.6

[[18]][[2]]
[1] 3.7
```

```
[[18]][[3]]
[1] 4.8


[[19]]
[[19]][[1]]
[1] 1.9

[[19]][[2]]
[1] 4

[[19]][[3]]
[1] 5.4


[[20]]
[[20]][[1]]
[1] 1.3

[[20]][[2]]
[1] 3.6

[[20]][[3]]
[1] 4.8


[[21]]
[[21]][[1]]
[1] 2

[[21]][[2]]
[1] 3.7

[[21]][[3]]
[1] 5.2


[[22]]
[[22]][[1]]
[1] 1.4

[[22]][[2]]
[1] 3.6
```

```
[[22]][[3]]
[1] 4.7


[[23]]
[[23]][[1]]
[1] 1

[[23]][[2]]
[1] 3.6

[[23]][[3]]
[1] 4.4


[[24]]
[[24]][[1]]
[1] 1.8

[[24]][[2]]
[1] 3.4

[[24]][[3]]
[1] 4.6


[[25]]
[[25]][[1]]
[1] 1.4

[[25]][[2]]
[1] 2.9

[[25]][[3]]
[1] 4.6


[[26]]
[[26]][[1]]
[1] 2

[[26]][[2]]
```

```
[1] 3.4

[[26]][[3]]
[1] 4.8


[[27]]
[[27]][[1]]
[1] 1.6

[[27]][[2]]
[1] 3.4

[[27]][[3]]
[1] 4.6


[[28]]
[[28]][[1]]
[1] 1.7

[[28]][[2]]
[1] 3.7

[[28]][[3]]
[1] 5


[[29]]
[[29]][[1]]
[1] 1.8

[[29]][[2]]
[1] 3.8

[[29]][[3]]
[1] 5


[[30]]
[[30]][[1]]
[1] 1.5
```

```
[[30]][[2]]
[1] 3.1

[[30]][[3]]
[1] 4.5


[[31]]
[[31]][[1]]
[1] 1.7

[[31]][[2]]
[1] 3.2

[[31]][[3]]
[1] 4.6


[[32]]
[[32]][[1]]
[1] 2

[[32]][[2]]
[1] 3.9

[[32]][[3]]
[1] 5


[[33]]
[[33]][[1]]
[1] 1.1

[[33]][[2]]
[1] 3.7

[[33]][[3]]
[1] 5.1


[[34]]
[[34]][[1]]
[1] 1.3
```

```
[[34]][[2]]
[1] 4.1

[[34]][[3]]
[1] 5.3


[[35]]
[[35]][[1]]
[1] 1.8

[[35]][[2]]
[1] 3.4

[[35]][[3]]
[1] 4.7


[[36]]
[[36]][[1]]
[1] 1.8

[[36]][[2]]
[1] 3.8

[[36]][[3]]
[1] 4.8


[[37]]
[[37]][[1]]
[1] 2

[[37]][[2]]
[1] 4.2

[[37]][[3]]
[1] 5.3


[[38]]
[[38]][[1]]
```

```
[1] 1.3

[[38]][[2]]
[1] 3.5

[[38]][[3]]
[1] 4.8


[[39]]
[[39]][[1]]
[1] 1.4

[[39]][[2]]
[1] 3.1

[[39]][[3]]
[1] 4.2


[[40]]
[[40]][[1]]
[1] 1.7

[[40]][[2]]
[1] 3.6

[[40]][[3]]
[1] 4.9


[[41]]
[[41]][[1]]
[1] 1.5

[[41]][[2]]
[1] 3.7

[[41]][[3]]
[1] 4.7


[[42]]
```

```
[[42]][[1]]
[1] 2.2

[[42]][[2]]
[1] 3.2

[[42]][[3]]
[1] 4.2


[[43]]
[[43]][[1]]
[1] 1.2

[[43]][[2]]
[1] 3.1

[[43]][[3]]
[1] 4.2


[[44]]
[[44]][[1]]
[1] 1.5

[[44]][[2]]
[1] 3.4

[[44]][[3]]
[1] 4.4


[[45]]
[[45]][[1]]
[1] 1.3

[[45]][[2]]
[1] 3.2

[[45]][[3]]
[1] 4.7
```

```
[[46]]
[[46]][[1]]
[1] 1.8

[[46]][[2]]
[1] 3.4

[[46]][[3]]
[1] 4.5


[[47]]
[[47]][[1]]
[1] 1.3

[[47]][[2]]
[1] 3.5

[[47]][[3]]
[1] 4.9


[[48]]
[[48]][[1]]
[1] 1.4

[[48]][[2]]
[1] 3.2

[[48]][[3]]
[1] 4.4


[[49]]
[[49]][[1]]
[1] 1.6

[[49]][[2]]
[1] 3.8

[[49]][[3]]
[1] 5.1
```

```
[[50]]
[[50]][[1]]
[1] 1.7

[[50]][[2]]
[1] 3.6

[[50]][[3]]
[1] 4.8


[[51]]
[[51]][[1]]
[1] 3.8

[[51]][[2]]
[1] 2.3

[[51]][[3]]
[1] 5.6


[[52]]
[[52]][[1]]
[1] 3.2

[[52]][[2]]
[1] 1.9

[[52]][[3]]
[1] 4.9


[[53]]
[[53]][[1]]
[1] 3.8

[[53]][[2]]
[1] 2

[[53]][[3]]
[1] 5.4
```

```
[[54]]
[[54]][[1]]
[1] 3.2

[[54]][[2]]
[1] 1.5

[[54]][[3]]
[1] 4.2


[[55]]
[[55]][[1]]
[1] 3.7

[[55]][[2]]
[1] 1.9

[[55]][[3]]
[1] 5


[[56]]
[[56]][[1]]
[1] 2.9

[[56]][[2]]
[1] 1.2

[[56]][[3]]
[1] 4.4


[[57]]
[[57]][[1]]
[1] 3

[[57]][[2]]
[1] 1.6

[[57]][[3]]
```

```
[1] 4.7


[[58]]
[[58]][[1]]
[1] 2.5

[[58]][[2]]
[1] 1.6

[[58]][[3]]
[1] 3.9


[[59]]
[[59]][[1]]
[1] 3.7

[[59]][[2]]
[1] 2

[[59]][[3]]
[1] 5.3


[[60]]
[[60]][[1]]
[1] 2.5

[[60]][[2]]
[1] 1.3

[[60]][[3]]
[1] 3.8


[[61]]
[[61]][[1]]
[1] 3

[[61]][[2]]
[1] 1.5
```

```
[[61]][[3]]
[1] 4


[[62]]
[[62]][[1]]
[1] 2.9

[[62]][[2]]
[1] 1.7

[[62]][[3]]
[1] 4.4


[[63]]
[[63]][[1]]
[1] 3.8

[[63]][[2]]
[1] 2

[[63]][[3]]
[1] 5


[[64]]
[[64]][[1]]
[1] 3.2

[[64]][[2]]
[1] 1.4

[[64]][[3]]
[1] 4.7


[[65]]
[[65]][[1]]
[1] 2.7

[[65]][[2]]
[1] 2
```

```
[[65]][[3]]
[1] 4.3


[[66]]
[[66]][[1]]
[1] 3.6

[[66]][[2]]
[1] 2.3

[[66]][[3]]
[1] 5.3


[[67]]
[[67]][[1]]
[1] 2.6

[[67]][[2]]
[1] 1.1

[[67]][[3]]
[1] 4.1


[[68]]
[[68]][[1]]
[1] 3.1

[[68]][[2]]
[1] 1.7

[[68]][[3]]
[1] 4.8


[[69]]
[[69]][[1]]
[1] 4

[[69]][[2]]
```

```
[1] 1.7

[[69]][[3]]
[1] 4.7


[[70]]
[[70]][[1]]
[1] 3.1

[[70]][[2]]
[1] 1.7

[[70]][[3]]
[1] 4.5


[[71]]
[[71]][[1]]
[1] 2.7

[[71]][[2]]
[1] 1.1

[[71]][[3]]
[1] 4.1


[[72]]
[[72]][[1]]
[1] 3.3

[[72]][[2]]
[1] 2.1

[[72]][[3]]
[1] 4.8


[[73]]
[[73]][[1]]
[1] 3.8
```

```
[[73]][[2]]
[1] 1.4

[[73]][[3]]
[1] 4.8


[[74]]
[[74]][[1]]
[1] 3.3

[[74]][[2]]
[1] 1.4

[[74]][[3]]
[1] 4.9


[[75]]
[[75]][[1]]
[1] 3.5

[[75]][[2]]
[1] 2.1

[[75]][[3]]
[1] 5.1


[[76]]
[[76]][[1]]
[1] 3.6

[[76]][[2]]
[1] 2.2

[[76]][[3]]
[1] 5.2


[[77]]
[[77]][[1]]
[1] 4
```

```
[[77]][[2]]
[1] 2

[[77]][[3]]
[1] 5.4


[[78]]
[[78]][[1]]
[1] 3.7

[[78]][[2]]
[1] 1.7

[[78]][[3]]
[1] 5


[[79]]
[[79]][[1]]
[1] 3.1

[[79]][[2]]
[1] 1.5

[[79]][[3]]
[1] 4.5


[[80]]
[[80]][[1]]
[1] 3.1

[[80]][[2]]
[1] 2.2

[[80]][[3]]
[1] 4.7


[[81]]
[[81]][[1]]
```

```
[1] 3.1

[[81]][[2]]
[1] 1.7

[[81]][[3]]
[1] 4.4


[[82]]
[[82]][[1]]
[1] 3.1

[[82]][[2]]
[1] 1.8

[[82]][[3]]
[1] 4.5


[[83]]
[[83]][[1]]
[1] 3.1

[[83]][[2]]
[1] 1.9

[[83]][[3]]
[1] 4.6


[[84]]
[[84]][[1]]
[1] 3.3

[[84]][[2]]
[1] 0.9

[[84]][[3]]
[1] 4.4


[[85]]
```

```
[[85]][[1]]
[1] 2.4

[[85]][[2]]
[1] 0.9

[[85]][[3]]
[1] 3.9


[[86]]
[[86]][[1]]
[1] 2.6

[[86]][[2]]
[1] 1.5

[[86]][[3]]
[1] 4.4


[[87]]
[[87]][[1]]
[1] 3.6

[[87]][[2]]
[1] 2

[[87]][[3]]
[1] 5.2


[[88]]
[[88]][[1]]
[1] 4

[[88]][[2]]
[1] 1.9

[[88]][[3]]
[1] 5
```

```
[[89]]
[[89]][[1]]
[1] 2.6

[[89]][[2]]
[1] 1.5

[[89]][[3]]
[1] 4.3


[[90]]
[[90]][[1]]
[1] 3

[[90]][[2]]
[1] 1.5

[[90]][[3]]
[1] 4.2


[[91]]
[[91]][[1]]
[1] 2.9

[[91]][[2]]
[1] 1.1

[[91]][[3]]
[1] 4.3


[[92]]
[[92]][[1]]
[1] 3.1

[[92]][[2]]
[1] 1.5

[[92]][[3]]
[1] 4.7
```

```
[[93]]
[[93]][[1]]
[1] 3.2

[[93]][[2]]
[1] 1.8

[[93]][[3]]
[1] 4.6


[[94]]
[[94]][[1]]
[1] 2.7

[[94]][[2]]
[1] 1.7

[[94]][[3]]
[1] 4


[[95]]
[[95]][[1]]
[1] 2.9

[[95]][[2]]
[1] 1.4

[[95]][[3]]
[1] 4.3


[[96]]
[[96]][[1]]
[1] 2.7

[[96]][[2]]
[1] 1.5

[[96]][[3]]
[1] 4.5
```

```
[[97]]
[[97]][[1]]
[1] 2.8

[[97]][[2]]
[1] 1.5

[[97]][[3]]
[1] 4.4


[[98]]
[[98]][[1]]
[1] 3.3

[[98]][[2]]
[1] 1.9

[[98]][[3]]
[1] 4.9


[[99]]
[[99]][[1]]
[1] 2.6

[[99]][[2]]
[1] 2.1

[[99]][[3]]
[1] 4


[[100]]
[[100]][[1]]
[1] 2.9

[[100]][[2]]
[1] 1.6

[[100]][[3]]
```

```
[1] 4.4


[[101]]
[[101]][[1]]
[1] 3

[[101]][[2]]
[1] 0.3

[[101]][[3]]
[1] 3.8


[[102]]
[[102]][[1]]
[1] 3.1

[[102]][[2]]
[1] 0.7

[[102]][[3]]
[1] 3.9


[[103]]
[[103]][[1]]
[1] 4.1

[[103]][[2]]
[1] 1.2

[[103]][[3]]
[1] 5


[[104]]
[[104]][[1]]
[1] 3.4

[[104]][[2]]
[1] 0.7
```

```
[[104]][[3]]
[1] 4.5


[[105]]
[[105]][[1]]
[1] 3.5

[[105]][[2]]
[1] 0.7

[[105]][[3]]
[1] 4.3


[[106]]
[[106]][[1]]
[1] 4.6

[[106]][[2]]
[1] 1

[[106]][[3]]
[1] 5.5


[[107]]
[[107]][[1]]
[1] 2.4

[[107]][[2]]
[1] 0.4

[[107]][[3]]
[1] 3.2


[[108]]
[[108]][[1]]
[1] 4.4

[[108]][[2]]
[1] 1
```

```
[[108]][[3]]
[1] 5.5


[[109]]
[[109]][[1]]
[1] 4.2

[[109]][[2]]
[1] 0.9

[[109]][[3]]
[1] 4.9


[[110]]
[[110]][[1]]
[1] 3.6

[[110]][[2]]
[1] 1.1

[[110]][[3]]
[1] 4.7


[[111]]
[[111]][[1]]
[1] 3.3

[[111]][[2]]
[1] 1.4

[[111]][[3]]
[1] 4.5


[[112]]
[[112]][[1]]
[1] 3.7

[[112]][[2]]
```

```
[1] 1.1

[[112]][[3]]
[1] 4.5


[[113]]
[[113]][[1]]
[1] 3.8

[[113]][[2]]
[1] 1.3

[[113]][[3]]
[1] 4.7


[[114]]
[[114]][[1]]
[1] 3.2

[[114]][[2]]
[1] 0.7

[[114]][[3]]
[1] 3.7


[[115]]
[[115]][[1]]
[1] 3

[[115]][[2]]
[1] 0.7

[[115]][[3]]
[1] 3.4


[[116]]
[[116]][[1]]
[1] 3.2
```

```
[[116]][[2]]
[1] 1.1

[[116]][[3]]
[1] 4.1


[[117]]
[[117]][[1]]
[1] 3.5

[[117]][[2]]
[1] 1

[[117]][[3]]
[1] 4.7


[[118]]
[[118]][[1]]
[1] 3.9

[[118]][[2]]
[1] 1

[[118]][[3]]
[1] 5.5


[[119]]
[[119]][[1]]
[1] 5.1

[[119]][[2]]
[1] 0.8

[[119]][[3]]
[1] 5.4


[[120]]
[[120]][[1]]
[1] 3.8
```

```
[[120]][[2]]
[1] 1

[[120]][[3]]
[1] 4.5


[[121]]
[[121]][[1]]
[1] 3.7

[[121]][[2]]
[1] 1.2

[[121]][[3]]
[1] 4.6


[[122]]
[[122]][[1]]
[1] 2.8

[[122]][[2]]
[1] 0.7

[[122]][[3]]
[1] 3.6


[[123]]
[[123]][[1]]
[1] 4.9

[[123]][[2]]
[1] 1

[[123]][[3]]
[1] 5.7


[[124]]
[[124]][[1]]
```

```
[1] 3.6

[[124]][[2]]
[1] 1.4

[[124]][[3]]
[1] 4.5


[[125]]
[[125]][[1]]
[1] 3.4

[[125]][[2]]
[1] 1

[[125]][[3]]
[1] 4.6


[[126]]
[[126]][[1]]
[1] 4

[[126]][[2]]
[1] 1.2

[[126]][[3]]
[1] 5.4


[[127]]
[[127]][[1]]
[1] 3.4

[[127]][[2]]
[1] 1.4

[[127]][[3]]
[1] 4.4


[[128]]
```

```
[[128]][[1]]
[1] 3.1

[[128]][[2]]
[1] 1.2

[[128]][[3]]
[1] 4.3


[[129]]
[[129]][[1]]
[1] 3.6

[[129]][[2]]
[1] 0.8

[[129]][[3]]
[1] 4.3


[[130]]
[[130]][[1]]
[1] 4.2

[[130]][[2]]
[1] 1.4

[[130]][[3]]
[1] 5.6


[[131]]
[[131]][[1]]
[1] 4.6

[[131]][[2]]
[1] 1.3

[[131]][[3]]
[1] 5.5
```

```
[[132]]
[[132]][[1]]
[1] 4.1

[[132]][[2]]
[1] 1.5

[[132]][[3]]
[1] 5.9


[[133]]
[[133]][[1]]
[1] 3.6

[[133]][[2]]
[1] 0.8

[[133]][[3]]
[1] 4.2


[[134]]
[[134]][[1]]
[1] 3.5

[[134]][[2]]
[1] 1.2

[[134]][[3]]
[1] 4.8


[[135]]
[[135]][[1]]
[1] 3.5

[[135]][[2]]
[1] 0.5

[[135]][[3]]
[1] 4.7
```

```
[[136]]
[[136]][[1]]
[1] 4.7

[[136]][[2]]
[1] 1.6

[[136]][[3]]
[1] 5.4


[[137]]
[[137]][[1]]
[1] 2.9

[[137]][[2]]
[1] 0.7

[[137]][[3]]
[1] 3.9


[[138]]
[[138]][[1]]
[1] 3.3

[[138]][[2]]
[1] 0.9

[[138]][[3]]
[1] 4.6


[[139]]
[[139]][[1]]
[1] 3

[[139]][[2]]
[1] 1.2

[[139]][[3]]
[1] 4.2
```

```
[[140]]
[[140]][[1]]
[1] 3.8

[[140]][[2]]
[1] 1.5

[[140]][[3]]
[1] 4.8


[[141]]
[[141]][[1]]
[1] 3.6

[[141]][[2]]
[1] 1.1

[[141]][[3]]
[1] 4.3


[[142]]
[[142]][[1]]
[1] 3.8

[[142]][[2]]
[1] 1.8

[[142]][[3]]
[1] 4.6


[[143]]
[[143]][[1]]
[1] 3.1

[[143]][[2]]
[1] 0.7

[[143]][[3]]
```

```
[1] 3.9


[[144]]
[[144]][[1]]
[1] 3.6

[[144]][[2]]
[1] 0.9

[[144]][[3]]
[1] 4.5


[[145]]
[[145]][[1]]
[1] 3.4

[[145]][[2]]
[1] 1

[[145]][[3]]
[1] 4.2


[[146]]
[[146]][[1]]
[1] 3.7

[[146]][[2]]
[1] 1.5

[[146]][[3]]
[1] 4.4


[[147]]
[[147]][[1]]
[1] 3.8

[[147]][[2]]
[1] 1.3
```

```
[[147]][[3]]
[1] 4.4


[[148]]
[[148]][[1]]
[1] 3.5

[[148]][[2]]
[1] 1.3

[[148]][[3]]
[1] 4.5


[[149]]
[[149]][[1]]
[1] 2.8

[[149]][[2]]
[1] 0.8

[[149]][[3]]
[1] 3.9


[[150]]
[[150]][[1]]
[1] 2.9

[[150]][[2]]
[1] 0.8

[[150]][[3]]
[1] 4.1
```

Advanced Function Writing

Unnamed arguments

Recall standardize function

```
standardize <- function(vector, center = TRUE, scale = TRUE) {
 mean <- mean(vector)
 stdev <- sd(vector)
 if (center) {
 vector <- vector - mean
 }
 if (scale) {
 vector <- vector / stdev
 }
 return(list(result = vector, mean = mean, sd = stdev))
 }
```

- sd() and mean() have as defaults na.rm = FALSE, so if there are NAs, the standardize function will return an error
- can use … so na.rm can be changed however desired

```
standardize <- function(vector, center = TRUE, scale = TRUE, ...) {
 mean <- mean(vector, ...)
 stdev <- sd(vector, ...)
 if (center) {
 vector <- vector - mean
 }
 if (scale) {
 vector <- vector / stdev
 }
 return(list(result = vector, mean = mean, sd = stdev))
}
standard_Ozone <- standardize(airquality$Ozone, na.rm = TRUE)
standard_Ozone$mean
```

```
[1] 42.12931
```

function to find group means

one group using enquo

```
library(rlang)
```

```
Attaching package: 'rlang'

The following objects are masked from 'package:purrr':

    %@%, flatten, flatten_chr, flatten_dbl, flatten_int, flatten_lgl,
    flatten_raw, invoke, splice
```

```
find_group_mean <- function(.df, group){
  group_name <- enquo(group)
  .df |>
    group_by(!!group_name) |>
    summarize(across(where(is.numeric),
                     list("mean" = mean),
                     .names = "{.fn}_{.col}"))
}
find_group_mean(iris, Species)
```

```
# A tibble: 3 x 5
  Species  mean_Sepal.Length mean_Sepal.Width mean_Petal.Length mean_Petal.Width
  <fct>                <dbl>            <dbl>             <dbl>            <dbl>
1 setosa                5.01             3.43              1.46            0.246
2 versico~              5.94             2.77              4.26            1.33
3 virgini~              6.59             2.97              5.55            2.03
```

one group using {{}}

```
find_group_mean <- function(.df, group){
  .df |>
    group_by({{group}}) |>
    summarize(across(where(is.numeric),
                     list("mean" = mean),
```

```
                    .names = "{.fn}_{.col}"))
}
find_group_mean(iris, Species)
```

```
# A tibble: 3 x 5
  Species  mean_Sepal.Length mean_Sepal.Width mean_Petal.Length mean_Petal.Width
  <fct>                <dbl>            <dbl>             <dbl>            <dbl>
1 setosa                5.01             3.43              1.46            0.246
2 versico~              5.94             2.77              4.26            1.33
3 virgini~              6.59             2.97              5.55            2.03
```

more than one group using quos

```
find_group_mean <- function(.df, ...){
  group_vars <- quos(...)
  .df |>
    group_by(!!!group_vars) |>
    summarize(across(where(is.numeric),
                     list("mean" = mean),
                     .names = "{.fn}_{.col}"))
}
find_group_mean(CO2, Type, Treatment)
```

```
`summarise()` has grouped output by 'Type'. You can override using the
`.groups` argument.
```

```
# A tibble: 4 x 4
# Groups:   Type [2]
  Type        Treatment  mean_conc mean_uptake
  <fct>       <fct>          <dbl>       <dbl>
1 Quebec      nonchilled       435        35.3
2 Quebec      chilled          435        31.8
3 Mississippi nonchilled       435        26.0
4 Mississippi chilled          435        15.8
```

as_label for tidyverse style functions

```r
find_group_mean <- function(.df, group, column){
  group_name <- enquo(group)
  column_name <- enquo(column)
  column_label <- paste0("mean_", as_label(column_name))
  .df |>
    group_by(!!group_name) |>
    summarize(!!(column_label) := mean(!!column_name))
}
find_group_mean(iris, Species, Sepal.Length)
```

```
# A tibble: 3 x 2
  Species    mean_Sepal.Length
  <fct>                  <dbl>
1 setosa                  5.01
2 versicolor              5.94
3 virginica               6.59
```

Pipeable Functions

Below need invisible(.df) because you can't pipe number of observations into summarize, the data is no longer accessible. invisible() returns data frame so you can keep using it.

```r
print_num_obs <- function(.df) {
 cat("The number of observations in the data set is ",
     nrow(.df),
     "\n",
     sep = "")
  invisible(.df)
}
iris |>
 print_num_obs() |>
 summarize(mean = mean(Sepal.Length))
```

```
The number of observations in the data set is 150
```

```
      mean
1 5.843333
```

Querying APIs

API: Application Programming Interface, a communication protocol for 2 computers or 2 softwares communication

```
library(tidycensus)
rent <- "DP04_0142PE" #PE means percentage
rent_data <- get_acs(variables = rent,
        geography = "county",
        geometry = TRUE, #returns the polygon data and allows for maps easily
        survey = "acs5",
        show_call = TRUE) #can add state and other things
```

```
Getting data from the 2018-2022 5-year ACS
```

```
Warning: * You have not set a Census API key. Users without a key are limited to 500
queries per day and may experience performance limitations.
i For best results, get a Census API key at
http://api.census.gov/data/key_signup.html and then supply the key to the
`census_api_key()` function to use it throughout your tidycensus session.
This warning is displayed once per session.
```

```
Downloading feature geometry from the Census website.  To cache shapefiles for use in future
```

```
Using the ACS Data Profile
```

```
Census API call: https://api.census.gov/data/2022/acs/acs5/profile?get=DP04_0142PE%2CDP04_014
```

```
  |
  |                                                                      |   0%
  |
  |                                                                      |   1%
  |
  |=                                                                     |   1%
  |
  |=                                                                     |   2%
  |
  |==                                                                    |   2%
  |
```

```
|==                                                                |   3%
|
|===                                                               |   4%
|
|===                                                               |   5%
|
|====                                                              |   5%
|
|====                                                              |   6%
|
|=====                                                             |   6%
|
|=====                                                             |   7%
|
|=====                                                             |   8%
|
|======                                                            |   8%
|
|======                                                            |   9%
|
|=======                                                           |   9%
|
|=======                                                           |  10%
|
|=======                                                           |  11%
|
|========                                                          |  11%
|
|========                                                          |  12%
|
|=========                                                         |  12%
|
|=========                                                         |  13%
|
|==========                                                        |  14%
|
|==========                                                        |  15%
|
|===========                                                       |  15%
|
|===========                                                       |  16%
|
|============                                                      |  17%
```

```
|
|============                                                     |  18%
|
|=============                                                    |  18%
|
|=============                                                    |  19%
|
|===============                                                  |  19%
|
|===============                                                  |  20%
|
|===============                                                  |  21%
|
|================                                                 |  21%
|
|================                                                 |  22%
|
|================                                                 |  22%
|
|=================                                                |  23%
|
|=================                                                |  24%
|
|==================                                               |  24%
|
|==================                                               |  25%
|
|==================                                               |  25%
|
|===================                                              |  26%
|
|===================                                              |  26%
|
|====================                                             |  27%
|
|====================                                             |  28%
|
|====================                                             |  28%
|
|=====================                                            |  29%
|
|=====================                                            |  29%
|
```

```
|=====================                                                 |  30%
|
|=====================                                                 |  31%
|
|======================                                                |  31%
|
|======================                                                |  32%
|
|=======================                                               |  32%
|
|=======================                                               |  33%
|
|========================                                              |  34%
|
|========================                                              |  35%
|
|=========================                                             |  35%
|
|=========================                                             |  36%
|
|==========================                                            |  37%
|
|==========================                                            |  38%
|
|===========================                                           |  38%
|
|===========================                                           |  39%
|
|============================                                          |  39%
|
|============================                                          |  40%
|
|============================                                          |  41%
|
|=============================                                         |  41%
|
|=============================                                         |  42%
|
|==============================                                        |  42%
|
|==============================                                        |  43%
|
|===============================                                       |  44%
```

```
|
|================================                                 |  45%
|
|================================                                 |  45%
|
|================================                                 |  46%
|
|=================================                                |  46%
|
|=================================                                |  47%
|
|=================================                                |  48%
|
|=================================                                |  48%
|
|=================================                                |  49%
|
|=================================                                |  49%
|
|=================================                                |  50%
|
|=================================                                |  51%
|
|=================================                                |  51%
|
|=================================                                |  52%
|
|=================================                                |  52%
|
|=================================                                |  53%
|
|=================================                                |  54%
|
|=================================                                |  54%
|
|=================================                                |  55%
|
|=================================                                |  55%
|
|=================================                                |  56%
|
|=================================                                |  57%
|
```

```
|=======================================                               |  58%
|
|=======================================                               |  58%
|
|========================================                              |  59%
|
|========================================                              |  60%
|
|=========================================                             |  61%
|
|=========================================                             |  62%
|
|=========================================                             |  62%
|
|=========================================                             |  63%
|
|==========================================                            |  64%
|
|==========================================                            |  64%
|
|==========================================                            |  65%
|
|==========================================                            |  65%
|
|===========================================                           |  66%
|
|===========================================                           |  66%
|
|===========================================                           |  67%
|
|============================================                          |  68%
|
|============================================                          |  69%
|
|============================================                          |  69%
|
|============================================                          |  70%
|
|=============================================                         |  71%
|
|=============================================                         |  72%
|
|=============================================                         |  72%
```

```
|
|================================================                      |  73%
|
|=================================================                     |  74%
|
|==================================================                    |  75%
|
|===================================================                   |  76%
|
|====================================================                  |  77%
|
|=====================================================                 |  78%
|
|======================================================                |  78%
|
|=======================================================               |  79%
|
|========================================================              |  79%
|
|=========================================================             |  80%
|
|==========================================================            |  81%
|
|===========================================================           |  81%
|
|============================================================          |  82%
|
|=============================================================         |  82%
|
|==============================================================        |  83%
|
|===============================================================       |  84%
|
|================================================================      |  85%
|
|=================================================================     |  85%
|
|==================================================================    |  86%
|
|===================================================================   |  87%
|
|====================================================================  |  88%
|
```

```
  |=============================================================         |  88%
  |
  |==============================================================        |  89%
  |
  |==============================================================        |  90%
  |
  |==============================================================        |  91%
  |
  |===============================================================       |  91%
  |
  |===============================================================       |  92%
  |
  |================================================================      |  92%
  |
  |================================================================      |  93%
  |
  |=================================================================     |  94%
  |
  |=================================================================     |  95%
  |
  |==================================================================    |  95%
  |
  |==================================================================    |  96%
  |
  |===================================================================   |  96%
  |
  |===================================================================   |  97%
  |
  |===================================================================   |  98%
  |
  |==================================================================== |  98%
  |
  |==================================================================== |  99%
  |
  |=====================================================================|  99%
  |
  |=====================================================================| 100%
```

```
rent_data
```

```
Simple feature collection with 3222 features and 5 fields
Geometry type: MULTIPOLYGON
```

```
Dimension:     XY
Bounding box:  xmin: -179.1467 ymin: 17.88328 xmax: 179.7785 ymax: 71.38782
Geodetic CRS:  NAD83
First 10 features:
   GEOID                            NAME   variable estimate  moe
1  01069        Houston County, Alabama DP04_0142P     36.5  2.2
2  01023        Choctaw County, Alabama DP04_0142P     37.8 13.2
3  01005        Barbour County, Alabama DP04_0142P     35.9  7.7
4  01107        Pickens County, Alabama DP04_0142P     35.5  8.6
5  01033        Colbert County, Alabama DP04_0142P     37.0  5.0
6  04012         La Paz County, Arizona DP04_0142P     20.1  6.3
7  04001         Apache County, Arizona DP04_0142P     17.8  4.2
8  05081  Little River County, Arkansas DP04_0142P     22.8 12.1
9  05121      Randolph County, Arkansas DP04_0142P     26.1  8.0
10 06037 Los Angeles County, California DP04_0142P     47.9  0.4
                         geometry
1  MULTIPOLYGON (((-85.71209 3...
2  MULTIPOLYGON (((-88.47323 3...
3  MULTIPOLYGON (((-85.74803 3...
4  MULTIPOLYGON (((-88.34043 3...
5  MULTIPOLYGON (((-88.13925 3...
6  MULTIPOLYGON (((-114.7312 3...
7  MULTIPOLYGON (((-110.0007 3...
8  MULTIPOLYGON (((-94.48558 3...
9  MULTIPOLYGON (((-91.40687 3...
10 MULTIPOLYGON (((-118.6044 3...
```

a plot

```
#install mapview
library(mapview)
library(sf)
```

```
Linking to GEOS 3.12.1, GDAL 3.8.4, PROJ 9.3.1; sf_use_s2() is TRUE
```

```
library(webshot)
map <- rent_data |>
  mapview::mapview(zcol = "estimate",
                   layer.name = "Median rent as a % of gross income")
```

build our own API

```
URL_ids <- "https://api.nhle.com/stats/rest/en/team"
id_info <- httr::GET(URL_ids)
str(id_info, max.level = 1)
```

```
List of 10
 $ url        : chr "https://api.nhle.com/stats/rest/en/team"
 $ status_code: int 200
 $ headers    :List of 10
  ..- attr(*, "class")= chr [1:2] "insensitive" "list"
 $ all_headers:List of 1
 $ cookies    :'data.frame':    0 obs. of  7 variables:
 $ content    : raw [1:6561] 7b 22 64 61 ...
 $ date       : POSIXct[1:1], format: "2024-06-24 19:56:09"
 $ times      : Named num [1:6] 0 0.0252 0.0401 0.0826 0.1367 ...
  ..- attr(*, "names")= chr [1:6] "redirect" "namelookup" "connect" "pretransfer" ...
 $ request    :List of 7
  ..- attr(*, "class")= chr "request"
 $ handle     :Class 'curl_handle' <externalptr>
 - attr(*, "class")= chr "response"
```

parse the data

```
library(jsonlite)
```

```
Attaching package: 'jsonlite'

The following objects are masked from 'package:rlang':

    flatten, unbox

The following object is masked from 'package:purrr':

    flatten
```

```
parsed <- fromJSON(rawToChar(id_info$content))
parsed
```

```
$data
   id franchiseId                fullName leagueId rawTricode triCode
1  11          35       Atlanta Thrashers      133        ATL     ATL
2  34          26        Hartford Whalers      133        HFD     HFD
3  31          15   Minnesota North Stars      133        MNS     MNS
4  32          27        Quebec Nordiques      133        QUE     QUE
5  33          28    Winnipeg Jets (1979)      133        WIN     WIN
6  35          23        Colorado Rockies      133        CLR     CLR
7  36           3  Ottawa Senators (1917)      133        SEN     SEN
8  37           4         Hamilton Tigers      133        HAM     HAM
9  38           9     Pittsburgh Pirates      133        PIR     PIR
10 39           9    Philadelphia Quakers      133        QUA     QUA
11 40          12         Detroit Cougars      133        DCG     DCG
12 41           2      Montreal Wanderers      133        MWN     MWN
13 42           4         Quebec Bulldogs      133        QBD     QBD
14 43           7        Montreal Maroons      133        MMR     MMR
15 44           8      New York Americans      133        NYA     NYA
16 45           3        St. Louis Eagles      133        SLE     SLE
17 46          13           Oakland Seals      133        OAK     OAK
18 47          21          Atlanta Flames      133        AFM     AFM
19 48          23      Kansas City Scouts      133        KCS     KCS
20 49          13        Cleveland Barons      133        CLE     CLE
21 50          12         Detroit Falcons      133        DFL     DFL
22 51           8      Brooklyn Americans      133        BRK     BRK
23 56          13 California Golden Seals      133        CGS     CGS
24 57           5          Toronto Arenas      133        TAN     TAN
25 58           5    Toronto St. Patricks      133        TSP     TSP
26 99          NA                     NHL      133        NHL     NHL
27 17          12       Detroit Red Wings      133        DET     DET
28  6           6           Boston Bruins      133        BOS     BOS
29  5          17     Pittsburgh Penguins      133        PIT     PIT
30 14          31     Tampa Bay Lightning      133        TBL     TBL
31  4          16     Philadelphia Flyers      133        PHI     PHI
32 12          26     Carolina Hurricanes      133        CAR     CAR
33 20          21          Calgary Flames      133        CGY     CGY
34  8           1      Montréal Canadiens      133        MTL     MTL
35 15          24     Washington Capitals      133        WSH     WSH
36 23          20       Vancouver Canucks      133        VAN     VAN
37 21          27      Colorado Avalanche      133        COL     COL
```

```
38 18          34     Nashville Predators      133        NSH     NSH
39 24          32           Anaheim Ducks      133        ANA     ANA
40 54          38    Vegas Golden Knights      133        VGK     VGK
41 25          15            Dallas Stars      133        DAL     DAL
42 27          28         Phoenix Coyotes      133        PHX     PHX
43 16          11      Chicago Blackhawks      133        CHI     CHI
44  3          10        New York Rangers      133        NYR     NYR
45 13          33        Florida Panthers      133        FLA     FLA
46 22          25         Edmonton Oilers      133        EDM     EDM
47 30          37          Minnesota Wild      133        MIN     MIN
48 19          18         St. Louis Blues      133        STL     STL
49  2          22      New York Islanders      133        NYI     NYI
50 26          14       Los Angeles Kings      133        LAK     LAK
51 70          NA        To be determined      133        TBD     TBD
52  7          19          Buffalo Sabres      133        BUF     BUF
53  9          30         Ottawa Senators      133        OTT     OTT
54 10           5     Toronto Maple Leafs      133        TOR     TOR
55  1          23       New Jersey Devils      133        NJD     NJD
56 52          35           Winnipeg Jets      133        WPG     WPG
57 55          39          Seattle Kraken      133        SEA     SEA
58 28          29         San Jose Sharks      133        SJS     SJS
59 53          28         Arizona Coyotes      133        ARI     ARI
60 59          40        Utah Hockey Club      133        UTA     UTA
61 29          36   Columbus Blue Jackets      133        CBJ     CBJ

$total
[1] 61
```

```
team_info <- as_tibble(parsed$data)
team_info
```

```
# A tibble: 61 x 6
      id franchiseId fullName               leagueId rawTricode triCode
   <int>       <int> <chr>                     <int> <chr>      <chr>
 1    11          35 Atlanta Thrashers           133 ATL        ATL
 2    34          26 Hartford Whalers            133 HFD        HFD
 3    31          15 Minnesota North Stars       133 MNS        MNS
 4    32          27 Quebec Nordiques            133 QUE        QUE
 5    33          28 Winnipeg Jets (1979)        133 WIN        WIN
 6    35          23 Colorado Rockies            133 CLR        CLR
 7    36           3 Ottawa Senators (1917)      133 SEN        SEN
 8    37           4 Hamilton Tigers             133 HAM        HAM
```

```
 9    38           9 Pittsburgh Pirates          133 PIR        PIR
10    39           9 Philadelphia Quakers        133 QUA        QUA
# i 51 more rows
```

another example

```
URL_team_stats <-
"https://api.nhle.com/stats/rest/en/team/summary?sort=wins&cayenneExp=seasonId=20232024%20and
team_stats_return <- httr::GET(URL_team_stats)
parsed_team_stats <- fromJSON(rawToChar(team_stats_return$content))
team_stats <- as_tibble(parsed_team_stats$data)
team_stats |>
 select(teamId, teamFullName, everything())
```

```
# A tibble: 32 x 24
   teamId teamFullName          faceoffWinPct gamesPlayed goalsAgainst
    <int> <chr>                         <dbl>       <int>        <int>
 1     28 San Jose Sharks               0.490          82          326
 2     16 Chicago Blackhawks            0.463          82          289
 3     29 Columbus Blue Jackets         0.472          82          298
 4     24 Anaheim Ducks                 0.466          82          293
 5      8 Montréal Canadiens            0.515          82          281
 6     55 Seattle Kraken                0.475          82          232
 7     53 Arizona Coyotes               0.451          82          274
 8      9 Ottawa Senators               0.510          82          281
 9     20 Calgary Flames                0.496          82          267
10      1 New Jersey Devils             0.535          82          281
# i 22 more rows
# i 19 more variables: goalsAgainstPerGame <dbl>, goalsFor <int>,
#   goalsForPerGame <dbl>, losses <int>, otLosses <int>,
#   penaltyKillNetPct <dbl>, penaltyKillPct <dbl>, pointPct <dbl>,
#   points <int>, powerPlayNetPct <dbl>, powerPlayPct <dbl>,
#   regulationAndOtWins <int>, seasonId <int>, shotsAgainstPerGame <dbl>,
#   shotsForPerGame <dbl>, ties <lgl>, wins <int>, winsInRegulation <int>, ...
```