Working with Data: Day 1

May 3-4, 2018

Outline

- review data types in R
- data frame manipulation with dplyr
- indexing with base R

Data types in R

- vectors
  - zero to many elements
  - all elements are the same type (and NA)
  - logical, integer, numeric, character, factor
  - 2D matrices and many-D arrays are still vectors, just with the dimensions encoded in the "dim" attribute
- lists
  - aka "generic vectors"
  - any element can be a vector of any type and length
- data.frames
  - just a list with one restriction: all elements have the same length
  - a general representation for a 2D table of data
  - each **list** element is a column in the table

Data frame

- representation of a 2D table of data
- columns are variables (something you measure/observe)
- rows are observations (variable values) of a specific **case**

Data manipulation

Typical tasks:

- view a subset of variables
- choose a subset of observations based on one or more variable values
- create new variables based on existing variables
- order observations by some variable value
- calculate a summary of a set of variable values

Base R can do all of those things, but it can be pretty low-level (focus on coding instead of analyzing data) and awkward, so a number of alternatives have been put forward over time.

Currently, the most popular such alternative solution for data manipulation is the package dplyr. It's so good at what it does, and integrates so well with other popular tools like *ggplot2*, that it has rapidly become the de-facto standard and it is what we will focus on today.

Dplyr has a set of functions, or *verbs* in its terminology, that each deal with one of the above tasks:

- `select`: view only some variables

- `filter`: choose observations by their values
- `arrange`: order observations (rows)
- `mutate`: create new variables
- `summarise`: calculate a summary of many variable values

Each verb works similarly:

- input data frame in the first argument
- other arguments can refer to variables as if they were local objects
- output is another data frame

Select

You can use the `select` function to focus on a subset of variables.

```r
library(dplyr)
select(mtcars,mpg,wt)
```

```
##                      mpg    wt
## Mazda RX4           21.0 2.620
## Mazda RX4 Wag       21.0 2.875
## Datsun 710          22.8 2.320
## Hornet 4 Drive      21.4 3.215
## Hornet Sportabout   18.7 3.440
## Valiant             18.1 3.460
## Duster 360          14.3 3.570
## Merc 240D           24.4 3.190
## Merc 230            22.8 3.150
## Merc 280            19.2 3.440
## Merc 280C           17.8 3.440
## Merc 450SE          16.4 4.070
## Merc 450SL          17.3 3.730
## Merc 450SLC         15.2 3.780
## Cadillac Fleetwood  10.4 5.250
## Lincoln Continental 10.4 5.424
## Chrysler Imperial   14.7 5.345
## Fiat 128            32.4 2.200
## Honda Civic         30.4 1.615
## Toyota Corolla      33.9 1.835
## Toyota Corona       21.5 2.465
## Dodge Challenger    15.5 3.520
## AMC Javelin         15.2 3.435
## Camaro Z28          13.3 3.840
## Pontiac Firebird    19.2 3.845
## Fiat X1-9           27.3 1.935
## Porsche 914-2       26.0 2.140
## Lotus Europa        30.4 1.513
## Ford Pantera L      15.8 3.170
## Ferrari Dino        19.7 2.770
## Maserati Bora       15.0 3.570
## Volvo 142E          21.4 2.780
```

There are many helpful functions that can be used with `select` to describe which variables to keep:

- starts_with(x, ignore.case = TRUE): names starts with x
- ends_with(x, ignore.case = TRUE): names ends in x

- contains(x, ignore.case = TRUE): selects all variables whose name contains x
- matches(x, ignore.case = TRUE): selects all variables whose name matches the regular expression x
- num_range("x", 1:5, width = 2): selects all variables (numerically) from x01 to x05.
- one_of("x", "y", "z"): selects variables provided in a character vector.
- everything(): selects all variables.

```
select(mtcars,starts_with("d"))
```

```
##                      disp drat
## Mazda RX4           160.0 3.90
## Mazda RX4 Wag       160.0 3.90
## Datsun 710          108.0 3.85
## Hornet 4 Drive      258.0 3.08
## Hornet Sportabout   360.0 3.15
## Valiant             225.0 2.76
## Duster 360          360.0 3.21
## Merc 240D           146.7 3.69
## Merc 230            140.8 3.92
## Merc 280            167.6 3.92
## Merc 280C           167.6 3.92
## Merc 450SE          275.8 3.07
## Merc 450SL          275.8 3.07
## Merc 450SLC         275.8 3.07
## Cadillac Fleetwood  472.0 2.93
## Lincoln Continental 460.0 3.00
## Chrysler Imperial   440.0 3.23
## Fiat 128             78.7 4.08
## Honda Civic          75.7 4.93
## Toyota Corolla       71.1 4.22
## Toyota Corona       120.1 3.70
## Dodge Challenger    318.0 2.76
## AMC Javelin         304.0 3.15
## Camaro Z28          350.0 3.73
## Pontiac Firebird    400.0 3.08
## Fiat X1-9            79.0 4.08
## Porsche 914-2       120.3 4.43
## Lotus Europa         95.1 3.77
## Ford Pantera L      351.0 4.22
## Ferrari Dino        145.0 3.62
## Maserati Bora       301.0 3.54
## Volvo 142E          121.0 4.11
```

This trick is handy to reorder the variables so that the ones you're most interested in are at the front, without dropping any:

```
select(mtcars, cyl, everything())
```

```
##                     cyl  mpg  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4             6 21.0 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag         6 21.0 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710            4 22.8 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive        6 21.4 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout     8 18.7 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant               6 18.1 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360            8 14.3 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D             4 24.4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230            4 22.8 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280            6 19.2 167.6 123 3.92 3.440 18.30  1  0    4    4
## Merc 280C           6 17.8 167.6 123 3.92 3.440 18.90  1  0    4    4
## Merc 450SE          8 16.4 275.8 180 3.07 4.070 17.40  0  0    3    3
## Merc 450SL          8 17.3 275.8 180 3.07 3.730 17.60  0  0    3    3
## Merc 450SLC         8 15.2 275.8 180 3.07 3.780 18.00  0  0    3    3
## Cadillac Fleetwood  8 10.4 472.0 205 2.93 5.250 17.98  0  0    3    4
## Lincoln Continental 8 10.4 460.0 215 3.00 5.424 17.82  0  0    3    4
## Chrysler Imperial   8 14.7 440.0 230 3.23 5.345 17.42  0  0    3    4
## Fiat 128            4 32.4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic         4 30.4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla      4 33.9  71.1  65 4.22 1.835 19.90  1  1    4    1
## Toyota Corona       4 21.5 120.1  97 3.70 2.465 20.01  1  0    3    1
## Dodge Challenger    8 15.5 318.0 150 2.76 3.520 16.87  0  0    3    2
## AMC Javelin         8 15.2 304.0 150 3.15 3.435 17.30  0  0    3    2
## Camaro Z28          8 13.3 350.0 245 3.73 3.840 15.41  0  0    3    4
## Pontiac Firebird    8 19.2 400.0 175 3.08 3.845 17.05  0  0    3    2
## Fiat X1-9           4 27.3  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2       4 26.0 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa        4 30.4  95.1 113 3.77 1.513 16.90  1  1    5    2
## Ford Pantera L      8 15.8 351.0 264 4.22 3.170 14.50  0  1    5    4
## Ferrari Dino        6 19.7 145.0 175 3.62 2.770 15.50  0  1    5    6
## Maserati Bora       8 15.0 301.0 335 3.54 3.570 14.60  0  1    5    8
## Volvo 142E          4 21.4 121.0 109 4.11 2.780 18.60  1  1    4    2
```

Using a named argument will rename a variable:

```
select(mtcars, mpg, weight=wt)
```

```
##                      mpg weight
## Mazda RX4           21.0  2.620
## Mazda RX4 Wag       21.0  2.875
## Datsun 710          22.8  2.320
## Hornet 4 Drive      21.4  3.215
## Hornet Sportabout   18.7  3.440
## Valiant             18.1  3.460
## Duster 360          14.3  3.570
## Merc 240D           24.4  3.190
## Merc 230            22.8  3.150
## Merc 280            19.2  3.440
## Merc 280C           17.8  3.440
## Merc 450SE          16.4  4.070
## Merc 450SL          17.3  3.730
## Merc 450SLC         15.2  3.780
## Cadillac Fleetwood  10.4  5.250
## Lincoln Continental 10.4  5.424
## Chrysler Imperial   14.7  5.345
## Fiat 128            32.4  2.200
## Honda Civic         30.4  1.615
## Toyota Corolla      33.9  1.835
## Toyota Corona       21.5  2.465
## Dodge Challenger    15.5  3.520
## AMC Javelin         15.2  3.435
## Camaro Z28          13.3  3.840
## Pontiac Firebird    19.2  3.845
## Fiat X1-9           27.3  1.935
## Porsche 914-2       26.0  2.140
## Lotus Europa        30.4  1.513
## Ford Pantera L      15.8  3.170
## Ferrari Dino        19.7  2.770
## Maserati Bora       15.0  3.570
## Volvo 142E          21.4  2.780
```

(You can also use `rename()` to change variable names while keeping all columns as they were.)

Filter

You can use `filter` to select specific rows based on a logical condition of a variable. To specify more than one condition, just give them as additional arguments. The conditions are joined together as a logical *and*:

```
filter(mtcars, cyl==8)
```

```
##     mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## 1  18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## 2  14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## 3  16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## 4  17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
## 5  15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## 6  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
## 7  10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
## 8  14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
## 9  15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
## 10 15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
## 11 13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## 12 19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
## 13 15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
## 14 15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
```

```
filter(mtcars, cyl==8, carb==3)
```

```
##    mpg cyl  disp  hp drat   wt qsec vs am gear carb
## 1 16.4   8 275.8 180 3.07 4.07 17.4  0  0    3    3
## 2 17.3   8 275.8 180 3.07 3.73 17.6  0  0    3    3
## 3 15.2   8 275.8 180 3.07 3.78 18.0  0  0    3    3
```

To use the logical *or* to join conditions, you must use the `|` operator explicitly:

```
filter(mtcars, cyl==4 | carb==8)
```

```
##     mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## 1  22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## 2  24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## 3  22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## 4  32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## 5  30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## 6  33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## 7  21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## 8  27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## 9  26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## 10 30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## 11 15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
## 12 21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
```

```
filter(mtcars, cyl==8 & carb==8)
```

```
##   mpg cyl disp  hp drat   wt qsec vs am gear carb
## 1  15   8  301 335 3.54 3.57 14.6  0  1    5    8
```

```
mtcars[mtcars$cyl==4 | mtcars$carb==8, ]
```

```
##                 mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Datsun 710     22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Merc 240D      24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230       22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Fiat 128       32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic    30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla 33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## Toyota Corona  21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## Fiat X1-9      27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2  26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa   30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## Maserati Bora  15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
## Volvo 142E     21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
```

If you need to select several conditions on the same variable you can use `%in%`:

```
filter(mtcars,carb==3 | carb==6 | carb==8)
```

```
##    mpg cyl  disp  hp drat   wt qsec vs am gear carb
## 1 16.4   8 275.8 180 3.07 4.07 17.4  0  0    3    3
## 2 17.3   8 275.8 180 3.07 3.73 17.6  0  0    3    3
## 3 15.2   8 275.8 180 3.07 3.78 18.0  0  0    3    3
## 4 19.7   6 145.0 175 3.62 2.77 15.5  0  1    5    6
## 5 15.0   8 301.0 335 3.54 3.57 14.6  0  1    5    8
```

```
filter(mtcars,carb %in% c(3,6,8))
```

```
##    mpg cyl  disp  hp drat   wt qsec vs am gear carb
## 1 16.4   8 275.8 180 3.07 4.07 17.4  0  0    3    3
## 2 17.3   8 275.8 180 3.07 3.73 17.6  0  0    3    3
## 3 15.2   8 275.8 180 3.07 3.78 18.0  0  0    3    3
## 4 19.7   6 145.0 175 3.62 2.77 15.5  0  1    5    6
## 5 15.0   8 301.0 335 3.54 3.57 14.6  0  1    5    8
```

To use numeric indices the *dplyr* function is `slice`.

```
slice(mtcars,c(1,3,5))
```

```
## # A tibble: 3 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  21.0    6.  160.  110.  3.90  2.62  16.5    0.    1.    4.    4.
## 2  22.8    4.  108.   93.  3.85  2.32  18.6    1.    1.    4.    1.
## 3  18.7    8.  360.  175.  3.15  3.44  17.0    0.    0.    3.    2.
```

Note: `slice` and `filter` do not carry the row names with the subset of rows.

If you wish to include the row names you need to add them to the data frame as a variable

```
filter(add_rownames(mtcars), cyl==8, carb==3)
```

```
## Warning: Deprecated, use tibble::rownames_to_column() instead.
```

```
## # A tibble: 3 x 12
##   rowname       mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear
##   <chr>       <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 Merc 450SE   16.4    8.  276.  180.  3.07  4.07  17.4    0.    0.    3.
## 2 Merc 450SL   17.3    8.  276.  180.  3.07  3.73  17.6    0.    0.    3.
## 3 Merc 450SLC  15.2    8.  276.  180.  3.07  3.78  18.0    0.    0.    3.
## # ... with 1 more variable: carb <dbl>
```

Chaining with %>%

when combining several function call together the command can be very hard to read

```
as.data.frame(select(filter(add_rownames(mtcars),mpg>=30),rowname,mpg,cyl,hp))
```

```
## Warning: Deprecated, use tibble::rownames_to_column() instead.
```

```
##          rowname  mpg cyl  hp
## 1       Fiat 128 32.4   4  66
## 2    Honda Civic 30.4   4  52
## 3 Toyota Corolla 33.9   4  65
## 4   Lotus Europa 30.4   4 113
```

You can chain commands together using the `%>%` operator.

`f(x) %>% g(y) is the same as g(f(x),y)`

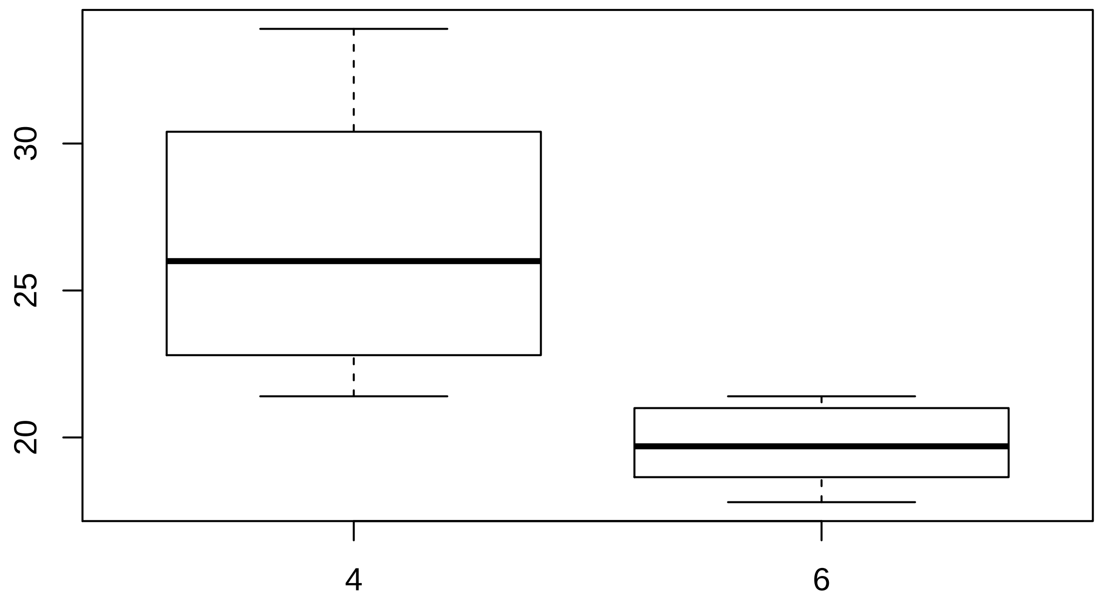
```
add_rownames(mtcars) %>%
  filter(mpg>=30) %>%
  select(rowname,mpg,cyl,hp) %>%
  as.data.frame()
```

```
## Warning: Deprecated, use tibble::rownames_to_column() instead.
```

```
##          rowname  mpg cyl  hp
## 1       Fiat 128 32.4   4  66
## 2    Honda Civic 30.4   4  52
## 3 Toyota Corolla 33.9   4  65
## 4   Lotus Europa 30.4   4 113
```

The default is to put the left hand side as the first argument of the right hand side. You can use `.` as a placeholder to change this behaviour

```
filter(mtcars, cyl<8) %>%
  boxplot(mpg~cyl, data=.)
boxplot(mpg~cyl, data=mtcars, subset= cyl<8)
```



Arrange

Use `arrange` to sort rows by value of a variable:

```
mtcars %>%
  arrange(mpg)
```

```
##     mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## 1  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
## 2  10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
## 3  13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## 4  14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## 5  14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
## 6  15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
## 7  15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## 8  15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
## 9  15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
## 10 15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
## 11 16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## 12 17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
## 13 17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
## 14 18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## 15 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## 16 19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
## 17 19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
## 18 19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
## 19 21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## 20 21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## 21 21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## 22 21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
## 23 21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## 24 22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## 25 22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## 26 24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## 27 26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## 28 27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## 29 30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## 30 30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## 31 32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## 32 33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
```

To break ties, just add more variables. Each additional variable will be used to break ties in the values of preceding ones:

```
arrange(mtcars,cyl,carb,gear) %>% select(cyl,carb,gear)
```

```
##    cyl carb gear
## 1    4    1    3
## 2    4    1    4
## 3    4    1    4
## 4    4    1    4
## 5    4    1    4
## 6    4    2    4
## 7    4    2    4
## 8    4    2    4
## 9    4    2    4
## 10   4    2    5
## 11   4    2    5
## 12   6    1    3
## 13   6    1    3
## 14   6    4    4
## 15   6    4    4
## 16   6    4    4
## 17   6    4    4
## 18   6    6    5
## 19   8    2    3
## 20   8    2    3
## 21   8    2    3
## 22   8    2    3
## 23   8    3    3
## 24   8    3    3
## 25   8    3    3
## 26   8    4    3
## 27   8    4    3
## 28   8    4    3
## 29   8    4    3
## 30   8    4    3
## 31   8    4    5
## 32   8    8    5
```

Use `desc()` to sort in descending order:

```r
mtcars %>%
  arrange(desc(mpg))
```

```
##     mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## 1  33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## 2  32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## 3  30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## 4  30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## 5  27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## 6  26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## 7  24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## 8  22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## 9  22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## 10 21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## 11 21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## 12 21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
## 13 21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## 14 21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## 15 19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
## 16 19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
## 17 19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
## 18 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## 19 18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## 20 17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
## 21 17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
## 22 16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## 23 15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
## 24 15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
## 25 15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## 26 15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
## 27 15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
## 28 14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
## 29 14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## 30 13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## 31 10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
## 32 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
```

Mutate

To add new variables based on some calculation, possibly using the value of other variables, use `mutate`:

```r
mutate(mtcars, displ_l = disp / 61.0237)
```

```
##     mpg cyl  disp  hp drat    wt  qsec vs am gear carb  displ_l
## 1  21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4 2.621932
## 2  21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4 2.621932
## 3  22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1 1.769804
## 4  21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1 4.227866
## 5  18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2 5.899347
## 6  18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1 3.687092
## 7  14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4 5.899347
## 8  24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2 2.403984
## 9  22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2 2.307300
## 10 19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4 2.746474
## 11 17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4 2.746474
## 12 16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3 4.519556
## 13 17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3 4.519556
## 14 15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3 4.519556
## 15 10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4 7.734700
## 16 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4 7.538055
## 17 14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4 7.210313
## 18 32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1 1.289663
## 19 30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2 1.240502
## 20 33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1 1.165121
## 21 21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1 1.968088
## 22 15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2 5.211090
## 23 15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2 4.981671
## 24 13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4 5.735477
## 25 19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2 6.554830
## 26 27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1 1.294579
## 27 26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2 1.971365
## 28 30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2 1.558411
## 29 15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4 5.751864
## 30 19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6 2.376126
## 31 15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8 4.932510
## 32 21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2 1.982836
```

You can add as many variables as you want, and even base their value on any preceding column:

```
mutate(mtcars, displ_l = disp / 61.0237, wt_kg = wt / 2.2, wt_rt = hp / wt_kg)
```

```
##     mpg cyl  disp  hp drat    wt  qsec vs am gear carb  displ_l     wt_kg
## 1  21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4 2.621932 1.1909091
## 2  21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4 2.621932 1.3068182
## 3  22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1 1.769804 1.0545455
## 4  21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1 4.227866 1.4613636
## 5  18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2 5.899347 1.5636364
## 6  18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1 3.687092 1.5727273
## 7  14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4 5.899347 1.6227273
## 8  24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2 2.403984 1.4500000
## 9  22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2 2.307300 1.4318182
## 10 19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4 2.746474 1.5636364
## 11 17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4 2.746474 1.5636364
## 12 16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3 4.519556 1.8500000
## 13 17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3 4.519556 1.6954545
## 14 15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3 4.519556 1.7181818
## 15 10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4 7.734700 2.3863636
## 16 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4 7.538055 2.4654545
## 17 14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4 7.210313 2.4295455
## 18 32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1 1.289663 1.0000000
## 19 30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2 1.240502 0.7340909
## 20 33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1 1.165121 0.8340909
## 21 21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1 1.968088 1.1204545
## 22 15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2 5.211090 1.6000000
## 23 15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2 4.981671 1.5613636
## 24 13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4 5.735477 1.7454545
## 25 19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2 6.554830 1.7477273
## 26 27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1 1.294579 0.8795455
## 27 26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2 1.971365 0.9727273
## 28 30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2 1.558411 0.6877273
## 29 15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4 5.751864 1.4409091
## 30 19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6 2.376126 1.2590909
## 31 15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8 4.932510 1.6227273
## 32 21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2 1.982836 1.2636364
##        wt_rt
## 1   92.36641
## 2   84.17391
## 3   88.18966
## 4   75.27216
## 5  111.91860
## 6   66.76301
## 7  150.98039
## 8   42.75862
## 9   66.34921
## 10  78.66279
## 11  78.66279
## 12  97.29730
## 13 106.16622
## 14 104.76190
## 15  85.90476
## 16  87.20501
## 17  94.66791
## 18  66.00000
## 19  70.83591
## 20  77.92916
## 21  86.57201
## 22  93.75000
## 23  96.06987
## 24 140.36458
## 25 100.13004
## 26  75.03876
## 27  93.55140
## 28 164.30932
## 29 183.21767
## 30 138.98917
## 31 206.44258
## 32  86.25899
```

To only keep the newly created variables, use `transmute`:

```
transmute(mtcars, displ_l = disp / 61.0237)
```

```
##     displ_l
## 1  2.621932
## 2  2.621932
## 3  1.769804
## 4  4.227866
## 5  5.899347
## 6  3.687092
## 7  5.899347
## 8  2.403984
## 9  2.307300
## 10 2.746474
## 11 2.746474
## 12 4.519556
## 13 4.519556
## 14 4.519556
## 15 7.734700
## 16 7.538055
## 17 7.210313
## 18 1.289663
## 19 1.240502
## 20 1.165121
## 21 1.968088
## 22 5.211090
## 23 4.981671
## 24 5.735477
## 25 6.554830
## 26 1.294579
## 27 1.971365
## 28 1.558411
## 29 5.751864
## 30 2.376126
## 31 4.932510
## 32 1.982836
```

Sumarise

`summarise` calculates a single value using a set of variable values:

```
summarise(mtcars, mean(mpg))
```

```
##   mean(mpg)
## 1  20.09062
```

This is the same as, and more wordy than, just using the same function on a data frame column:

```
mean(mtcars$mpg)
```

```
## [1] 20.09062
```

```
summarise(mtcars, mpg=mean(mpg))
```

```
##        mpg
## 1 20.09062
```

Its benefits get more obvious when calculating multiple summaries or when the calculation is based on more than one column:

```
summarise(mtcars, mpg=mean(mpg), wt=median(wt))
```

```
##        mpg    wt
## 1 20.09062 3.325
```

```
summarise(mtcars, mpg=mean(mpg), wt=median(wt), rat=mpg/wt)
```

```
##        mpg    wt      rat
## 1 20.09062 3.325 6.042293
```

You can use any function that takes in a vector of values and returns a scalar (i.e., “aggregates”): `mean`, `median`, `max`, etc.

But where `summarise` really comes handy is when we want to calculate it for groups of observations. This is done by first applying the `group_by` verb and then feed it into `summarise`. For instance, to calculate the mean gas mileage for each engine size:

```
mtcars %>%
  group_by(cyl) %>%
  summarise(mpg = mean(mpg), wt=median(wt))
```

```
## # A tibble: 3 x 3
##     cyl   mpg    wt
##   <dbl> <dbl> <dbl>
## 1    4.  26.7  2.20
## 2    6.  19.7  3.22
## 3    8.  15.1  3.76
```

Note that the grouping variable are included in the result. Using more than one grouping variable will split by each one in turn:

```
mtcars %>%
  group_by(am, cyl) %>%
  summarise(mpg = mean(mpg), wt=median(wt))
```

```
## # A tibble: 6 x 4
## # Groups:   am [?]
##      am   cyl   mpg    wt
##   <dbl> <dbl> <dbl> <dbl>
## 1    0.    4.  22.9  3.15
## 2    0.    6.  19.1  3.44
## 3    0.    8.  15.0  3.81
## 4    1.    4.  28.1  2.04
## 5    1.    6.  20.6  2.77
## 6    1.    8.  15.4  3.37
```

Each summarizing “rolls up” one grouping, starting from the bottom:

```
cars_am_cyl <- mtcars %>%
  group_by(am, cyl) %>%
  summarise(mpg = mean(mpg), wt=median(wt))
cars_am_cyl
```

```
## # A tibble: 6 x 4
## # Groups:   am [?]
##      am   cyl   mpg    wt
##   <dbl> <dbl> <dbl> <dbl>
## 1    0.    4.  22.9  3.15
## 2    0.    6.  19.1  3.44
## 3    0.    8.  15.0  3.81
## 4    1.    4.  28.1  2.04
## 5    1.    6.  20.6  2.77
## 6    1.    8.  15.4  3.37
```

```
cars_am_cyl %>%
  summarise(mpg = mean(mpg), wt=median(wt))
```

```
## # A tibble: 2 x 3
##      am   mpg    wt
##   <dbl> <dbl> <dbl>
## 1    0.  19.0  3.44
## 2    1.  21.3  2.77
```