R Workshop Part 2: Data manipulation

Rose Driscoll

2/22/2019

Data manipulation basics

Topics

1. The tidyverse and tidy data
2. String manipulation with `stringr`
3. Data manipulation with `dplyr`

PART 1: The Tidyverse and Tidy Data

The tidyverse (https://www.tidyverse.org/) is a group of R packages that are all based on a common philosophy and grammar. `dplyr` and `ggplot2` are part of the tidyverse, as are `tidyr`, `stringr`, and `tibble` (which we will use today) and `readr`, `purrr`, and `forcats` (which we won't be using).

1. Tidyverse syntax

All tidyverse functions have the same basic syntax:

`function(data, other_arguments)`

Many tidyverse functions are named with a verb describing what you are doing to the data (i.e., gathering, filtering, summarizing, etc.) The first argument is always the data frame, followed by other arguments.

Sometimes, these arguments will be the names of columns in the data frame that you want the function to work with. When you refer to columns by name in a tidyverse function, you don't ever have to use `$` the way you do in base R. To refer to a column named `temperature` in a data frame named `weather`, you would do `weather$temperature` in base R. Within a tidyverse function, it's just

`function(weather, temperature)`

This will become more intuitive once we start using tidyverse functions. But first, let's talk about tidy data...

2. Tidy data

All tidyverse packages run on tidy data. "Tidy" data means data that is in the long form, with one column for each variable and one row for each observation.

```
# some tidy data
data(airquality)
head(airquality)
```

```
##   Ozone Solar.R Wind Temp Month Day
## 1    41     190  7.4   67     5   1
## 2    36     118  8.0   72     5   2
## 3    12     149 12.6   74     5   3
## 4    18     313 11.5   62     5   4
## 5    NA      NA 14.3   56     5   5
```

```
## 6    28      NA 14.9   66     5   6
```

```
# some untidy data
le <- read.csv("le_mess.csv")
head(le)
```

```
##               country X1951 X1952 X1953 X1954 X1955 X1956 X1957 X1958
## 1         Afghanistan 27.13 27.67 28.19 28.73 29.27 29.80 30.34 30.86
## 2             Albania 54.72 55.23 55.85 56.59 57.45 58.42 59.48 60.60
## 3             Algeria 43.03 43.50 43.96 44.44 44.93 45.44 45.94 46.45
## 4              Angola 31.05 31.59 32.14 32.69 33.24 33.78 34.33 34.88
## 5 Antigua and Barbuda 58.26 58.80 59.34 59.87 60.41 60.93 61.45 61.97
## 6           Argentina 61.93 62.54 63.10 63.59 64.03 64.41 64.73 65.00
##   X1959 X1960 X1961 X1962 X1963 X1964 X1965 X1966 X1967 X1968 X1969 X1970
## 1 31.40 31.94 32.47 33.01 33.53 34.07 34.60 35.13 35.66 36.17 36.69 37.20
## 2 61.75 62.87 63.92 64.84 65.60 66.18 66.59 66.88 67.11 67.32 67.55 67.83
## 3 46.97 47.50 48.02 48.55 49.07 49.58 50.09 50.58 51.05 51.49 51.95 52.41
## 4 35.43 35.98 36.53 37.08 37.63 38.18 38.74 39.28 39.84 40.39 40.95 41.50
## 5 62.48 62.97 63.46 63.93 64.38 64.81 65.23 65.63 66.03 66.41 66.81 67.19
## 6 65.22 65.39 65.53 65.64 65.74 65.84 65.95 66.08 66.26 66.47 66.72 67.01
##   X1971 X1972 X1973 X1974 X1975 X1976 X1977 X1978 X1979 X1980 X1981 X1982
## 1 37.70 38.19 38.67 39.14 39.61 40.07 40.53 40.98 41.46 41.96 42.51 43.11
## 2 68.16 68.53 68.93 69.35 69.77 70.17 70.54 70.86 71.14 71.39 71.63 71.88
## 3 52.88 53.38 53.91 54.52 55.24 56.11 57.13 58.28 59.56 60.92 62.31 63.69
## 4 42.06 42.62 43.17 43.71 44.22 44.68 45.12 45.50 45.84 46.14 46.42 46.69
## 5 67.56 67.94 68.30 68.64 68.99 69.32 69.64 69.96 70.28 70.59 70.90 71.22
## 6 67.32 67.64 67.96 68.28 68.60 68.92 69.24 69.57 69.89 70.20 70.51 70.78
##   X1983 X1984 X1985 X1986 X1987 X1988 X1989 X1990 X1991 X1992 X1993 X1994
## 1 43.75 44.45 45.21 46.02 46.87 47.74 48.62  49.5  49.3  49.4  49.5  48.9
## 2 72.15 72.42 72.71 72.96 73.14 73.25 73.30  73.3  73.4  73.6  73.6  73.6
## 3 64.97 66.15 67.18 68.04 68.75 69.33 69.81  70.2  70.5  70.9  71.2  71.4
## 4 46.96 47.23 47.50 47.75 47.99 48.20 48.40  48.6  49.3  49.6  48.4  50.0
## 5 71.52 71.82 72.13 72.42 72.70 72.97 73.24  73.5  73.6  73.5  73.4  73.4
## 6 71.04 71.26 71.46 71.66 71.84 72.05 72.26  72.5  72.7  72.8  73.1  73.4
##   X1995 X1996 X1997 X1998 X1999 X2000 X2001 X2002 X2003 X2004 X2005 X2006
## 1  49.4  49.7  49.5  48.6  50.0  50.1  50.4  51.0  51.4  51.8  52.0  52.1
## 2  73.7  73.8  74.1  74.2  74.2  74.7  75.1  75.5  75.7  75.9  76.2  76.4
## 3  71.6  72.1  72.4  72.6  73.0  73.3  73.5  73.8  73.9  74.4  74.8  75.0
## 4  50.9  51.3  51.7  51.8  51.8  52.3  52.5  53.3  53.9  54.5  55.2  55.7
## 5  73.5  73.5  73.9  74.1  74.0  73.8  74.1  74.3  74.5  74.6  74.9  74.9
## 6  73.5  73.5  73.6  73.8  73.9  74.2  74.3  74.3  74.5  75.0  75.3  75.3
##   X2007 X2008 X2009 X2010 X2011 X2012 X2013 X2014 X2015 X2016
## 1  52.4  52.8  53.3  53.6  54.0  54.4  54.8  54.9  53.8 52.72
## 2  76.6  76.8  77.0  77.2  77.4  77.5  77.7  77.9  78.0 78.10
## 3  75.3  75.5  75.7  76.0  76.1  76.2  76.3  76.3  76.4 76.50
## 4  56.2  56.7  57.1  57.6  58.1  58.5  58.8  59.2  59.6 60.00
## 5  75.3  75.5  75.7  75.8  75.9  76.1  76.2  76.3  76.4 76.50
## 6  75.2  75.4  75.6  75.8  76.0  76.1  76.2  76.3  76.5 76.70
```

3. Putting data in tidy (long) format

A lot of data that you encounter out in the world doesn't come nicely pre-tidied, so it's important to know how to tidy data for yourself. For this, use the `gather()` function from the `tidyr` package.

```
# gather(data, key, value, ...)
# All of the old column names will be gathered in a column named with the key
# All of the old cells will be gathered into a column named with the value
# Use - to exclude columns from gathering
tidy_le <- gather(le, year, life_expectancy, -country)
head(tidy_le)
```

```
##               country  year life_expectancy
## 1         Afghanistan X1951           27.13
## 2             Albania X1951           54.72
## 3             Algeria X1951           43.03
## 4              Angola X1951           31.05
## 5 Antigua and Barbuda X1951           58.26
## 6           Argentina X1951           61.93
```

Some base R functions do **not** work with tidy data (`prcomp()` is a great example of this) so if you are going back and forth between base `R` and (for example) `dplyr` you will also need to know how to take data out of tidy format.

```
# spread(data, key, value)
untidied_le <- spread(tidy_le, year, life_expectancy)
head(untidied_le)
```

```
##               country X1951 X1952 X1953 X1954 X1955 X1956 X1957 X1958
## 1         Afghanistan 27.13 27.67 28.19 28.73 29.27 29.80 30.34 30.86
## 2             Albania 54.72 55.23 55.85 56.59 57.45 58.42 59.48 60.60
## 3             Algeria 43.03 43.50 43.96 44.44 44.93 45.44 45.94 46.45
## 4              Angola 31.05 31.59 32.14 32.69 33.24 33.78 34.33 34.88
## 5 Antigua and Barbuda 58.26 58.80 59.34 59.87 60.41 60.93 61.45 61.97
## 6           Argentina 61.93 62.54 63.10 63.59 64.03 64.41 64.73 65.00
##   X1959 X1960 X1961 X1962 X1963 X1964 X1965 X1966 X1967 X1968 X1969 X1970
## 1 31.40 31.94 32.47 33.01 33.53 34.07 34.60 35.13 35.66 36.17 36.69 37.20
## 2 61.75 62.87 63.92 64.84 65.60 66.18 66.59 66.88 67.11 67.32 67.55 67.83
## 3 46.97 47.50 48.02 48.55 49.07 49.58 50.09 50.58 51.05 51.49 51.95 52.41
## 4 35.43 35.98 36.53 37.08 37.63 38.18 38.74 39.28 39.84 40.39 40.95 41.50
## 5 62.48 62.97 63.46 63.93 64.38 64.81 65.23 65.63 66.03 66.41 66.81 67.19
## 6 65.22 65.39 65.53 65.64 65.74 65.84 65.95 66.08 66.26 66.47 66.72 67.01
##   X1971 X1972 X1973 X1974 X1975 X1976 X1977 X1978 X1979 X1980 X1981 X1982
## 1 37.70 38.19 38.67 39.14 39.61 40.07 40.53 40.98 41.46 41.96 42.51 43.11
## 2 68.16 68.53 68.93 69.35 69.77 70.17 70.54 70.86 71.14 71.39 71.63 71.88
## 3 52.88 53.38 53.91 54.52 55.24 56.11 57.13 58.28 59.56 60.92 62.31 63.69
## 4 42.06 42.62 43.17 43.71 44.22 44.68 45.12 45.50 45.84 46.14 46.42 46.69
## 5 67.56 67.94 68.30 68.64 68.99 69.32 69.64 69.96 70.28 70.59 70.90 71.22
## 6 67.32 67.64 67.96 68.28 68.60 68.92 69.24 69.57 69.89 70.20 70.51 70.78
##   X1983 X1984 X1985 X1986 X1987 X1988 X1989 X1990 X1991 X1992 X1993 X1994
## 1 43.75 44.45 45.21 46.02 46.87 47.74 48.62  49.5  49.3  49.4  49.5  48.9
## 2 72.15 72.42 72.71 72.96 73.14 73.25 73.30  73.3  73.4  73.6  73.6  73.6
## 3 64.97 66.15 67.18 68.04 68.75 69.33 69.81  70.2  70.5  70.9  71.2  71.4
## 4 46.96 47.23 47.50 47.75 47.99 48.20 48.40  48.6  49.3  49.6  48.4  50.0
## 5 71.52 71.82 72.13 72.42 72.70 72.97 73.24  73.5  73.6  73.5  73.4  73.4
## 6 71.04 71.26 71.46 71.66 71.84 72.05 72.26  72.5  72.7  72.8  73.1  73.4
##   X1995 X1996 X1997 X1998 X1999 X2000 X2001 X2002 X2003 X2004 X2005 X2006
## 1  49.4  49.7  49.5  48.6  50.0  50.1  50.4  51.0  51.4  51.8  52.0  52.1
## 2  73.7  73.8  74.1  74.2  74.2  74.7  75.1  75.5  75.7  75.9  76.2  76.4
## 3  71.6  72.1  72.4  72.6  73.0  73.3  73.5  73.8  73.9  74.4  74.8  75.0
```

```
## 4  50.9  51.3  51.7  51.8  51.8  52.3  52.5  53.3  53.9  54.5  55.2  55.7
## 5  73.5  73.5  73.9  74.1  74.0  73.8  74.1  74.3  74.5  74.6  74.9  74.9
## 6  73.5  73.5  73.6  73.8  73.9  74.2  74.3  74.3  74.5  75.0  75.3  75.3
##   X2007 X2008 X2009 X2010 X2011 X2012 X2013 X2014 X2015 X2016
## 1  52.4  52.8  53.3  53.6  54.0  54.4  54.8  54.9  53.8 52.72
## 2  76.6  76.8  77.0  77.2  77.4  77.5  77.7  77.9  78.0 78.10
## 3  75.3  75.5  75.7  76.0  76.1  76.2  76.3  76.3  76.4 76.50
## 4  56.2  56.7  57.1  57.6  58.1  58.5  58.8  59.2  59.6 60.00
## 5  75.3  75.5  75.7  75.8  75.9  76.1  76.2  76.3  76.4 76.50
## 6  75.2  75.4  75.6  75.8  76.0  76.1  76.2  76.3  76.5 76.70
```

4. Dealing with rownames

Another problem you might encounter is data with rownames. Tidyverse functions don't work well with rownames - they like to have all of the data in (named) columns so that it can all be handled in the same way. The `tibble` package has a great function for converting rownames into a column...

```
data("mtcars")
head(mtcars)
```

```
##                    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
```

```
rownames(mtcars)
```

```
##  [1] "Mazda RX4"           "Mazda RX4 Wag"       "Datsun 710"
##  [4] "Hornet 4 Drive"      "Hornet Sportabout"   "Valiant"
##  [7] "Duster 360"          "Merc 240D"           "Merc 230"
## [10] "Merc 280"            "Merc 280C"           "Merc 450SE"
## [13] "Merc 450SL"          "Merc 450SLC"         "Cadillac Fleetwood"
## [16] "Lincoln Continental" "Chrysler Imperial"   "Fiat 128"
## [19] "Honda Civic"         "Toyota Corolla"      "Toyota Corona"
## [22] "Dodge Challenger"    "AMC Javelin"         "Camaro Z28"
## [25] "Pontiac Firebird"    "Fiat X1-9"           "Porsche 914-2"
## [28] "Lotus Europa"        "Ford Pantera L"      "Ferrari Dino"
## [31] "Maserati Bora"       "Volvo 142E"
```

```
mtcars_models <- rownames_to_column(mtcars, "model")
head(mtcars_models)
```

```
##               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
```

5. Dealing with column names

Because of the way tidyverse syntax works, you will frequently need to refer to columns in a data frame by name. R doesn't care what these column names are,* but it's a good idea to use column names that are meaningful and easy for you to interpret and remember. `dplyr` provides a simple function for renaming one, several, or all columns in a data frame:

```
head(mtcars_models)
```

```
##               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
```

```
mtcars_tidy <- rename(mtcars_models, cylinders = cyl, horsepower = hp, weight_1000 = wt, transmission =
head(mtcars_tidy)
```

```
##               model  mpg cylinders disp horsepower drat weight_1000  qsec
## 1         Mazda RX4 21.0         6  160        110 3.90       2.620 16.46
## 2     Mazda RX4 Wag 21.0         6  160        110 3.90       2.875 17.02
## 3        Datsun 710 22.8         4  108         93 3.85       2.320 18.61
## 4    Hornet 4 Drive 21.4         6  258        110 3.08       3.215 19.44
## 5 Hornet Sportabout 18.7         8  360        175 3.15       3.440 17.02
## 6           Valiant 18.1         6  225        105 2.76       3.460 20.22
##   vs transmission gear carburetors
## 1  0            1    4           4
## 2  0            1    4           4
## 3  1            1    4           1
## 4  1            0    3           1
## 5  0            0    3           2
## 6  1            0    3           1
```

NOTE: R doesn't care what the column names are... unless they start with a number or have spaces in them. If column names start with a number or have spaces in them, you have to enclose them in backticks when you refer to them:

```
test <- data.frame(a=c(1,2), b=c(1,2))
colnames(test) <- c("1", "2")
# test$1 # this throws an error
test$`1` # this works though
```

```
## [1] 1 2
```

```
colnames(test) <- c("column 1", "column 2")
#test$column 1 # this throws an error
test$`column 1` # this works though
```

```
## [1] 1 2
```

```
# the backtick rule applies whether you are working in base R or dplyr.
```

In general, it's better to avoid column names like this. R tries to help you avoid it by adding an X to the start of any column name that starts with a number when you read in data with `read.csv()` or `read.table()` (it did this above with the `le` data.)

PART 2: String Manipulation with stringr

R added Xs to all of the years in the life expectancy (`le`) data when we read it in, and these didn't go away when we converted the data to long format...

```
head(tidy_le)
```

```
##               country  year life_expectancy
## 1         Afghanistan X1951           27.13
## 2             Albania X1951           54.72
## 3             Algeria X1951           43.03
## 4              Angola X1951           31.05
## 5 Antigua and Barbuda X1951           58.26
## 6           Argentina X1951           61.93
```

In order to get rid of the Xs, we need to manupulate strings with `stringr`.

```
# extract `year` column so that we are just working with a vector of strings for the example
year <- tidy_le$year
head(year)
```

```
## [1] "X1951" "X1951" "X1951" "X1951" "X1951" "X1951"
```

```
# use str_sub to extract just the part of the string after the X
# str_sub(string, start = 1L, end = -1L)
year_clean <- str_sub(year, 2, 5)
head(year_clean)
```

```
## [1] "1951" "1951" "1951" "1951" "1951" "1951"
```

```
# default start is first character and default end is last character, so this does the same thing:
year_clean <- str_sub(year, 2)
head(year_clean)
```

```
## [1] "1951" "1951" "1951" "1951" "1951" "1951"
```

Some more `stringr` functions:

```
data(fruit)
str_detect(fruit, "berry")
```

```
##  [1] FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE  TRUE  TRUE
## [12] FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE FALSE
## [23] FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE  TRUE  TRUE
## [34] FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
## [45] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE
## [56] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [67] FALSE FALSE FALSE  TRUE FALSE FALSE  TRUE FALSE FALSE  TRUE FALSE
## [78] FALSE FALSE FALSE
```

```
str_which(fruit, "berry")
```

```
##  [1]  6  7 10 11 19 21 29 32 33 38 50 70 73 76
```

```
str_subset(fruit, "berry")
```

```
##  [1] "bilberry"    "blackberry"  "blueberry"   "boysenberry" "cloudberry"
##  [6] "cranberry"   "elderberry"  "goji berry"  "gooseberry"  "huckleberry"
## [11] "mulberry"    "raspberry"   "salal berry" "strawberry"
```

```r
str_split(fruit, " ")
```

```
## [[1]]
## [1] "apple"
##
## [[2]]
## [1] "apricot"
##
## [[3]]
## [1] "avocado"
##
## [[4]]
## [1] "banana"
##
## [[5]]
## [1] "bell"   "pepper"
##
## [[6]]
## [1] "bilberry"
##
## [[7]]
## [1] "blackberry"
##
## [[8]]
## [1] "blackcurrant"
##
## [[9]]
## [1] "blood"  "orange"
##
## [[10]]
## [1] "blueberry"
##
## [[11]]
## [1] "boysenberry"
##
## [[12]]
## [1] "breadfruit"
##
## [[13]]
## [1] "canary" "melon"
##
## [[14]]
## [1] "cantaloupe"
##
## [[15]]
## [1] "cherimoya"
##
## [[16]]
## [1] "cherry"
##
## [[17]]
## [1] "chili"  "pepper"
##
## [[18]]
```

```
## [1] "clementine"
##
## [[19]]
## [1] "cloudberry"
##
## [[20]]
## [1] "coconut"
##
## [[21]]
## [1] "cranberry"
##
## [[22]]
## [1] "cucumber"
##
## [[23]]
## [1] "currant"
##
## [[24]]
## [1] "damson"
##
## [[25]]
## [1] "date"
##
## [[26]]
## [1] "dragonfruit"
##
## [[27]]
## [1] "durian"
##
## [[28]]
## [1] "eggplant"
##
## [[29]]
## [1] "elderberry"
##
## [[30]]
## [1] "feijoa"
##
## [[31]]
## [1] "fig"
##
## [[32]]
## [1] "goji"  "berry"
##
## [[33]]
## [1] "gooseberry"
##
## [[34]]
## [1] "grape"
##
## [[35]]
## [1] "grapefruit"
##
## [[36]]
```

```
## [1] "guava"
##
## [[37]]
## [1] "honeydew"
##
## [[38]]
## [1] "huckleberry"
##
## [[39]]
## [1] "jackfruit"
##
## [[40]]
## [1] "jambul"
##
## [[41]]
## [1] "jujube"
##
## [[42]]
## [1] "kiwi"  "fruit"
##
## [[43]]
## [1] "kumquat"
##
## [[44]]
## [1] "lemon"
##
## [[45]]
## [1] "lime"
##
## [[46]]
## [1] "loquat"
##
## [[47]]
## [1] "lychee"
##
## [[48]]
## [1] "mandarine"
##
## [[49]]
## [1] "mango"
##
## [[50]]
## [1] "mulberry"
##
## [[51]]
## [1] "nectarine"
##
## [[52]]
## [1] "nut"
##
## [[53]]
## [1] "olive"
##
## [[54]]
```

```
## [1] "orange"
##
## [[55]]
## [1] "pamelo"
##
## [[56]]
## [1] "papaya"
##
## [[57]]
## [1] "passionfruit"
##
## [[58]]
## [1] "peach"
##
## [[59]]
## [1] "pear"
##
## [[60]]
## [1] "persimmon"
##
## [[61]]
## [1] "physalis"
##
## [[62]]
## [1] "pineapple"
##
## [[63]]
## [1] "plum"
##
## [[64]]
## [1] "pomegranate"
##
## [[65]]
## [1] "pomelo"
##
## [[66]]
## [1] "purple"     "mangosteen"
##
## [[67]]
## [1] "quince"
##
## [[68]]
## [1] "raisin"
##
## [[69]]
## [1] "rambutan"
##
## [[70]]
## [1] "raspberry"
##
## [[71]]
## [1] "redcurrant"
##
## [[72]]
```

```
## [1] "rock"  "melon"
##
## [[73]]
## [1] "salal" "berry"
##
## [[74]]
## [1] "satsuma"
##
## [[75]]
## [1] "star"  "fruit"
##
## [[76]]
## [1] "strawberry"
##
## [[77]]
## [1] "tamarillo"
##
## [[78]]
## [1] "tangerine"
##
## [[79]]
## [1] "ugli"  "fruit"
##
## [[80]]
## [1] "watermelon"
```

You can also use regular expressions with `stringr` functions:

```
str_subset(fruit, "[ap][pe][pa][lr]")
```

```
## [1] "apple"     "pear"      "pineapple"
```

However, a word of caution: while `stringr` is very powerful, it may not always be the best tool for pure string manipulation work. Check the `stringr` tidyverse page (https://stringr.tidyverse.org/) and `stringr` cheatsheet to see what sorts of functionality `stringr` has to offer. Python might be a better choice for some kinds of string manipulation problems.

PART 3: Data manipulation with dplyr

dplyr is what I would consider to be the "meat" of the tidyverse's data manipulation tools.

1. Single-table operations

These are sometimes known as the "seven verbs of dplyr", although there are actually at least ten basic functions...

Select

`select()` allows you to select columns from a data frame, dropping columns that aren't mentioned.

```
select(mtcars_tidy, model, mpg, cylinders)
```

```
##                 model  mpg cylinders
## 1           Mazda RX4 21.0         6
## 2       Mazda RX4 Wag 21.0         6
## 3          Datsun 710 22.8         4
## 4      Hornet 4 Drive 21.4         6
## 5   Hornet Sportabout 18.7         8
## 6             Valiant 18.1         6
## 7          Duster 360 14.3         8
## 8           Merc 240D 24.4         4
## 9            Merc 230 22.8         4
## 10           Merc 280 19.2         6
## 11           Merc 280C 17.8         6
## 12          Merc 450SE 16.4         8
## 13          Merc 450SL 17.3         8
## 14         Merc 450SLC 15.2         8
## 15  Cadillac Fleetwood 10.4         8
## 16 Lincoln Continental 10.4         8
## 17   Chrysler Imperial 14.7         8
## 18            Fiat 128 32.4         4
## 19         Honda Civic 30.4         4
## 20      Toyota Corolla 33.9         4
## 21       Toyota Corona 21.5         4
## 22    Dodge Challenger 15.5         8
## 23         AMC Javelin 15.2         8
## 24          Camaro Z28 13.3         8
## 25    Pontiac Firebird 19.2         8
## 26           Fiat X1-9 27.3         4
## 27       Porsche 914-2 26.0         4
## 28        Lotus Europa 30.4         4
## 29      Ford Pantera L 15.8         8
## 30        Ferrari Dino 19.7         6
## 31       Maserati Bora 15.0         8
## 32          Volvo 142E 21.4         4
```

You can also use - to drop named columns:

```
select(mtcars_tidy, -qsec, -vs)
```

```
##                 model  mpg cylinders  disp horsepower drat weight_1000
## 1           Mazda RX4 21.0         6 160.0        110 3.90       2.620
## 2       Mazda RX4 Wag 21.0         6 160.0        110 3.90       2.875
## 3          Datsun 710 22.8         4 108.0         93 3.85       2.320
## 4      Hornet 4 Drive 21.4         6 258.0        110 3.08       3.215
## 5   Hornet Sportabout 18.7         8 360.0        175 3.15       3.440
## 6             Valiant 18.1         6 225.0        105 2.76       3.460
## 7          Duster 360 14.3         8 360.0        245 3.21       3.570
## 8           Merc 240D 24.4         4 146.7         62 3.69       3.190
## 9            Merc 230 22.8         4 140.8         95 3.92       3.150
## 10           Merc 280 19.2         6 167.6        123 3.92       3.440
## 11           Merc 280C 17.8         6 167.6        123 3.92       3.440
## 12          Merc 450SE 16.4         8 275.8        180 3.07       4.070
## 13          Merc 450SL 17.3         8 275.8        180 3.07       3.730
## 14         Merc 450SLC 15.2         8 275.8        180 3.07       3.780
## 15  Cadillac Fleetwood 10.4         8 472.0        205 2.93       5.250
## 16 Lincoln Continental 10.4         8 460.0        215 3.00       5.424
```

```
## 17   Chrysler Imperial 14.7        8 440.0        230 3.23       5.345
## 18            Fiat 128 32.4        4  78.7         66 4.08       2.200
## 19         Honda Civic 30.4        4  75.7         52 4.93       1.615
## 20      Toyota Corolla 33.9        4  71.1         65 4.22       1.835
## 21       Toyota Corona 21.5        4 120.1         97 3.70       2.465
## 22    Dodge Challenger 15.5        8 318.0        150 2.76       3.520
## 23         AMC Javelin 15.2        8 304.0        150 3.15       3.435
## 24          Camaro Z28 13.3        8 350.0        245 3.73       3.840
## 25    Pontiac Firebird 19.2        8 400.0        175 3.08       3.845
## 26           Fiat X1-9 27.3        4  79.0         66 4.08       1.935
## 27       Porsche 914-2 26.0        4 120.3         91 4.43       2.140
## 28        Lotus Europa 30.4        4  95.1        113 3.77       1.513
## 29      Ford Pantera L 15.8        8 351.0        264 4.22       3.170
## 30        Ferrari Dino 19.7        6 145.0        175 3.62       2.770
## 31       Maserati Bora 15.0        8 301.0        335 3.54       3.570
## 32          Volvo 142E 21.4        4 121.0        109 4.11       2.780
##    transmission gear carburetors
## 1             1    4           4
## 2             1    4           4
## 3             1    4           1
## 4             0    3           1
## 5             0    3           2
## 6             0    3           1
## 7             0    3           4
## 8             0    4           2
## 9             0    4           2
## 10            0    4           4
## 11            0    4           4
## 12            0    3           3
## 13            0    3           3
## 14            0    3           3
## 15            0    3           4
## 16            0    3           4
## 17            0    3           4
## 18            1    4           1
## 19            1    4           2
## 20            1    4           1
## 21            0    3           1
## 22            0    3           2
## 23            0    3           2
## 24            0    3           4
## 25            0    3           2
## 26            1    4           1
## 27            1    5           2
## 28            1    5           2
## 29            1    5           4
## 30            1    5           6
## 31            1    5           8
## 32            1    4           2
```

If you want to change the order of your columns, select can do that too!

```
select(mtcars_tidy, cylinders, vs, mpg, model)
```

```
##    cylinders vs  mpg              model
```

```
## 1          6  0 21.0           Mazda RX4
## 2          6  0 21.0       Mazda RX4 Wag
## 3          4  1 22.8          Datsun 710
## 4          6  1 21.4      Hornet 4 Drive
## 5          8  0 18.7   Hornet Sportabout
## 6          6  1 18.1             Valiant
## 7          8  0 14.3          Duster 360
## 8          4  1 24.4           Merc 240D
## 9          4  1 22.8            Merc 230
## 10         6  1 19.2            Merc 280
## 11         6  1 17.8           Merc 280C
## 12         8  0 16.4          Merc 450SE
## 13         8  0 17.3          Merc 450SL
## 14         8  0 15.2         Merc 450SLC
## 15         8  0 10.4  Cadillac Fleetwood
## 16         8  0 10.4 Lincoln Continental
## 17         8  0 14.7   Chrysler Imperial
## 18         4  1 32.4            Fiat 128
## 19         4  1 30.4         Honda Civic
## 20         4  1 33.9      Toyota Corolla
## 21         4  1 21.5       Toyota Corona
## 22         8  0 15.5    Dodge Challenger
## 23         8  0 15.2         AMC Javelin
## 24         8  0 13.3          Camaro Z28
## 25         8  0 19.2    Pontiac Firebird
## 26         4  1 27.3           Fiat X1-9
## 27         4  0 26.0       Porsche 914-2
## 28         4  1 30.4        Lotus Europa
## 29         8  0 15.8      Ford Pantera L
## 30         6  0 19.7        Ferrari Dino
## 31         8  0 15.0       Maserati Bora
## 32         4  1 21.4          Volvo 142E
```

```
# now model is on the far right instead of the far left
```

Filter

`filter()` is used to filter the rows of a data frame based on one or more conditions.

```
filter(mtcars_tidy, cylinders > 6, transmission == 1)
```

```
##            model  mpg cylinders disp horsepower drat weight_1000 qsec vs
## 1 Ford Pantera L 15.8         8  351        264 4.22        3.17 14.5  0
## 2  Maserati Bora 15.0         8  301        335 3.54        3.57 14.6  0
##   transmission gear carburetors
## 1            1    5           4
## 2            1    5           8
```

```
filter(mtcars_tidy, model == "Valiant")
```

```
##     model  mpg cylinders disp horsepower drat weight_1000  qsec vs
## 1 Valiant 18.1         6  225        105 2.76        3.46 20.22  1
##   transmission gear carburetors
## 1            0    3           1
```

If you want to apply more than one condition to a single variable, boolean operators can come in handy:

```r
filter(mtcars_tidy, mpg > 20 & mpg < 25)
```

```
##            model  mpg cylinders  disp horsepower drat weight_1000  qsec vs
## 1      Mazda RX4 21.0         6 160.0        110 3.90       2.620 16.46  0
## 2  Mazda RX4 Wag 21.0         6 160.0        110 3.90       2.875 17.02  0
## 3     Datsun 710 22.8         4 108.0         93 3.85       2.320 18.61  1
## 4 Hornet 4 Drive 21.4         6 258.0        110 3.08       3.215 19.44  1
## 5      Merc 240D 24.4         4 146.7         62 3.69       3.190 20.00  1
## 6       Merc 230 22.8         4 140.8         95 3.92       3.150 22.90  1
## 7  Toyota Corona 21.5         4 120.1         97 3.70       2.465 20.01  1
## 8     Volvo 142E 21.4         4 121.0        109 4.11       2.780 18.60  1
##   transmission gear carburetors
## 1            1    4           4
## 2            1    4           4
## 3            1    4           1
## 4            0    3           1
## 5            0    4           2
## 6            0    4           2
## 7            0    3           1
## 8            1    4           2
```

```r
# for & simply supplying the two conditions separately does the same thing
filter(mtcars_tidy, mpg > 20, mpg < 25)
```

```
##            model  mpg cylinders  disp horsepower drat weight_1000  qsec vs
## 1      Mazda RX4 21.0         6 160.0        110 3.90       2.620 16.46  0
## 2  Mazda RX4 Wag 21.0         6 160.0        110 3.90       2.875 17.02  0
## 3     Datsun 710 22.8         4 108.0         93 3.85       2.320 18.61  1
## 4 Hornet 4 Drive 21.4         6 258.0        110 3.08       3.215 19.44  1
## 5      Merc 240D 24.4         4 146.7         62 3.69       3.190 20.00  1
## 6       Merc 230 22.8         4 140.8         95 3.92       3.150 22.90  1
## 7  Toyota Corona 21.5         4 120.1         97 3.70       2.465 20.01  1
## 8     Volvo 142E 21.4         4 121.0        109 4.11       2.780 18.60  1
##   transmission gear carburetors
## 1            1    4           4
## 2            1    4           4
## 3            1    4           1
## 4            0    3           1
## 5            0    4           2
## 6            0    4           2
## 7            0    3           1
## 8            1    4           2
```

```r
# but this doesn't work for any other operators
filter(mtcars_tidy, gear == 3 | gear == 5)
```

```
##                  model  mpg cylinders  disp horsepower drat weight_1000
## 1       Hornet 4 Drive 21.4         6 258.0        110 3.08       3.215
## 2    Hornet Sportabout 18.7         8 360.0        175 3.15       3.440
## 3              Valiant 18.1         6 225.0        105 2.76       3.460
## 4           Duster 360 14.3         8 360.0        245 3.21       3.570
## 5           Merc 450SE 16.4         8 275.8        180 3.07       4.070
## 6           Merc 450SL 17.3         8 275.8        180 3.07       3.730
## 7          Merc 450SLC 15.2         8 275.8        180 3.07       3.780
## 8   Cadillac Fleetwood 10.4         8 472.0        205 2.93       5.250
```

```
## 9  Lincoln Continental 10.4         8 460.0        215 3.00       5.424
## 10   Chrysler Imperial 14.7         8 440.0        230 3.23       5.345
## 11        Toyota Corona 21.5         4 120.1         97 3.70       2.465
## 12    Dodge Challenger 15.5         8 318.0        150 2.76       3.520
## 13         AMC Javelin 15.2         8 304.0        150 3.15       3.435
## 14          Camaro Z28 13.3         8 350.0        245 3.73       3.840
## 15    Pontiac Firebird 19.2         8 400.0        175 3.08       3.845
## 16       Porsche 914-2 26.0         4 120.3         91 4.43       2.140
## 17        Lotus Europa 30.4         4  95.1        113 3.77       1.513
## 18      Ford Pantera L 15.8         8 351.0        264 4.22       3.170
## 19        Ferrari Dino 19.7         6 145.0        175 3.62       2.770
## 20       Maserati Bora 15.0         8 301.0        335 3.54       3.570
##     qsec vs transmission gear carburetors
## 1  19.44  1            0    3           1
## 2  17.02  0            0    3           2
## 3  20.22  1            0    3           1
## 4  15.84  0            0    3           4
## 5  17.40  0            0    3           3
## 6  17.60  0            0    3           3
## 7  18.00  0            0    3           3
## 8  17.98  0            0    3           4
## 9  17.82  0            0    3           4
## 10 17.42  0            0    3           4
## 11 20.01  1            0    3           1
## 12 16.87  0            0    3           2
## 13 17.30  0            0    3           2
## 14 15.41  0            0    3           4
## 15 17.05  0            0    3           2
## 16 16.70  0            1    5           2
## 17 16.90  1            1    5           2
## 18 14.50  0            1    5           4
## 19 15.50  0            1    5           6
## 20 14.60  0            1    5           8
```

Sample_n and sample_frac

If you want a random subset of the rows of a table, use `sample_n()` to pull out a set number of rows or `sample_frac()` to pull out a fraction of the rows.

```
sample_n(mtcars_tidy, 5)
```

```
##               model  mpg cylinders  disp horsepower drat weight_1000
## 5  Hornet Sportabout 18.7         8 360.0        175 3.15       3.440
## 4     Hornet 4 Drive 21.4         6 258.0        110 3.08       3.215
## 1          Mazda RX4 21.0         6 160.0        110 3.90       2.620
## 2      Mazda RX4 Wag 21.0         6 160.0        110 3.90       2.875
## 21     Toyota Corona 21.5         4 120.1         97 3.70       2.465
##     qsec vs transmission gear carburetors
## 5  17.02  0            0    3           2
## 4  19.44  1            0    3           1
## 1  16.46  0            1    4           4
## 2  17.02  0            1    4           4
## 21 20.01  1            0    3           1
```

```r
sample_frac(mtcars_tidy, 0.5) # sample half of the rows
```

```
##                model  mpg cylinders  disp horsepower drat weight_1000
## 26         Fiat X1-9 27.3         4  79.0         66 4.08       1.935
## 9           Merc 230 22.8         4 140.8         95 3.92       3.150
## 32        Volvo 142E 21.4         4 121.0        109 4.11       2.780
## 1          Mazda RX4 21.0         6 160.0        110 3.90       2.620
## 29    Ford Pantera L 15.8         8 351.0        264 4.22       3.170
## 22  Dodge Challenger 15.5         8 318.0        150 2.76       3.520
## 8           Merc 240D 24.4         4 146.7         62 3.69       3.190
## 23       AMC Javelin 15.2         8 304.0        150 3.15       3.435
## 5  Hornet Sportabout 18.7         8 360.0        175 3.15       3.440
## 30      Ferrari Dino 19.7         6 145.0        175 3.62       2.770
## 27     Porsche 914-2 26.0         4 120.3         91 4.43       2.140
## 7         Duster 360 14.3         8 360.0        245 3.21       3.570
## 21     Toyota Corona 21.5         4 120.1         97 3.70       2.465
## 3         Datsun 710 22.8         4 108.0         93 3.85       2.320
## 18          Fiat 128 32.4         4  78.7         66 4.08       2.200
## 19       Honda Civic 30.4         4  75.7         52 4.93       1.615
##     qsec vs transmission gear carburetors
## 26 18.90  1            1    4           1
## 9  22.90  1            0    4           2
## 32 18.60  1            1    4           2
## 1  16.46  0            1    4           4
## 29 14.50  0            1    5           4
## 22 16.87  0            0    3           2
## 8  20.00  1            0    4           2
## 23 17.30  0            0    3           2
## 5  17.02  0            0    3           2
## 30 15.50  0            1    5           6
## 27 16.70  0            1    5           2
## 7  15.84  0            0    3           4
## 21 20.01  1            0    3           1
## 3  18.61  1            1    4           1
## 18 19.47  1            1    4           1
## 19 18.52  1            1    4           2
```

```r
sample_frac(mtcars_tidy) # if you don't supply a fraction, it gives you all of the rows back in random
```

```
##                  model  mpg cylinders  disp horsepower drat weight_1000
## 18            Fiat 128 32.4         4  78.7         66 4.08       2.200
## 21       Toyota Corona 21.5         4 120.1         97 3.70       2.465
## 23         AMC Javelin 15.2         8 304.0        150 3.15       3.435
## 12          Merc 450SE 16.4         8 275.8        180 3.07       4.070
## 32          Volvo 142E 21.4         4 121.0        109 4.11       2.780
## 1            Mazda RX4 21.0         6 160.0        110 3.90       2.620
## 26           Fiat X1-9 27.3         4  79.0         66 4.08       1.935
## 15  Cadillac Fleetwood 10.4         8 472.0        205 2.93       5.250
## 2        Mazda RX4 Wag 21.0         6 160.0        110 3.90       2.875
## 4       Hornet 4 Drive 21.4         6 258.0        110 3.08       3.215
## 28        Lotus Europa 30.4         4  95.1        113 3.77       1.513
## 7           Duster 360 14.3         8 360.0        245 3.21       3.570
## 8            Merc 240D 24.4         4 146.7         62 3.69       3.190
## 10            Merc 280 19.2         6 167.6        123 3.92       3.440
```

```
## 16 Lincoln Continental 10.4         8 460.0        215 3.00       5.424
## 22    Dodge Challenger 15.5         8 318.0        150 2.76       3.520
## 13          Merc 450SL 17.3         8 275.8        180 3.07       3.730
## 19         Honda Civic 30.4         4  75.7         52 4.93       1.615
## 11           Merc 280C 17.8         6 167.6        123 3.92       3.440
## 17   Chrysler Imperial 14.7         8 440.0        230 3.23       5.345
## 25    Pontiac Firebird 19.2         8 400.0        175 3.08       3.845
## 30        Ferrari Dino 19.7         6 145.0        175 3.62       2.770
## 3           Datsun 710 22.8         4 108.0         93 3.85       2.320
## 9             Merc 230 22.8         4 140.8         95 3.92       3.150
## 14         Merc 450SLC 15.2         8 275.8        180 3.07       3.780
## 27       Porsche 914-2 26.0         4 120.3         91 4.43       2.140
## 5    Hornet Sportabout 18.7         8 360.0        175 3.15       3.440
## 24          Camaro Z28 13.3         8 350.0        245 3.73       3.840
## 29      Ford Pantera L 15.8         8 351.0        264 4.22       3.170
## 20      Toyota Corolla 33.9         4  71.1         65 4.22       1.835
## 31       Maserati Bora 15.0         8 301.0        335 3.54       3.570
## 6              Valiant 18.1         6 225.0        105 2.76       3.460
##     qsec vs transmission gear carburetors
## 18 19.47  1            1    4           1
## 21 20.01  1            0    3           1
## 23 17.30  0            0    3           2
## 12 17.40  0            0    3           3
## 32 18.60  1            1    4           2
## 1  16.46  0            1    4           4
## 26 18.90  1            1    4           1
## 15 17.98  0            0    3           4
## 2  17.02  0            1    4           4
## 4  19.44  1            0    3           1
## 28 16.90  1            1    5           2
## 7  15.84  0            0    3           4
## 8  20.00  1            0    4           2
## 10 18.30  1            0    4           4
## 16 17.82  0            0    3           4
## 22 16.87  0            0    3           2
## 13 17.60  0            0    3           3
## 19 18.52  1            1    4           2
## 11 18.90  1            0    4           4
## 17 17.42  0            0    3           4
## 25 17.05  0            0    3           2
## 30 15.50  0            1    5           6
## 3  18.61  1            1    4           1
## 9  22.90  1            0    4           2
## 14 18.00  0            0    3           3
## 27 16.70  0            1    5           2
## 5  17.02  0            0    3           2
## 24 15.41  0            0    3           4
## 29 14.50  0            1    5           4
## 20 19.90  1            1    4           1
## 31 14.60  0            1    5           8
## 6  20.22  1            0    3           1
```

Mutate and transmute

`mutate()` is used to create new columns in a table by applying some sort of a function or rule. `transmute()` does the same, but drops all of the old columns and only returns the new one.

```
mutate(mtcars_tidy, weight = weight_1000*1000)
```

```
##                  model  mpg cylinders  disp horsepower drat weight_1000
## 1            Mazda RX4 21.0         6 160.0        110 3.90       2.620
## 2        Mazda RX4 Wag 21.0         6 160.0        110 3.90       2.875
## 3           Datsun 710 22.8         4 108.0         93 3.85       2.320
## 4       Hornet 4 Drive 21.4         6 258.0        110 3.08       3.215
## 5    Hornet Sportabout 18.7         8 360.0        175 3.15       3.440
## 6              Valiant 18.1         6 225.0        105 2.76       3.460
## 7           Duster 360 14.3         8 360.0        245 3.21       3.570
## 8            Merc 240D 24.4         4 146.7         62 3.69       3.190
## 9             Merc 230 22.8         4 140.8         95 3.92       3.150
## 10            Merc 280 19.2         6 167.6        123 3.92       3.440
## 11           Merc 280C 17.8         6 167.6        123 3.92       3.440
## 12          Merc 450SE 16.4         8 275.8        180 3.07       4.070
## 13          Merc 450SL 17.3         8 275.8        180 3.07       3.730
## 14         Merc 450SLC 15.2         8 275.8        180 3.07       3.780
## 15  Cadillac Fleetwood 10.4         8 472.0        205 2.93       5.250
## 16 Lincoln Continental 10.4         8 460.0        215 3.00       5.424
## 17   Chrysler Imperial 14.7         8 440.0        230 3.23       5.345
## 18            Fiat 128 32.4         4  78.7         66 4.08       2.200
## 19         Honda Civic 30.4         4  75.7         52 4.93       1.615
## 20      Toyota Corolla 33.9         4  71.1         65 4.22       1.835
## 21       Toyota Corona 21.5         4 120.1         97 3.70       2.465
## 22    Dodge Challenger 15.5         8 318.0        150 2.76       3.520
## 23         AMC Javelin 15.2         8 304.0        150 3.15       3.435
## 24          Camaro Z28 13.3         8 350.0        245 3.73       3.840
## 25    Pontiac Firebird 19.2         8 400.0        175 3.08       3.845
## 26           Fiat X1-9 27.3         4  79.0         66 4.08       1.935
## 27       Porsche 914-2 26.0         4 120.3         91 4.43       2.140
## 28        Lotus Europa 30.4         4  95.1        113 3.77       1.513
## 29      Ford Pantera L 15.8         8 351.0        264 4.22       3.170
## 30        Ferrari Dino 19.7         6 145.0        175 3.62       2.770
## 31       Maserati Bora 15.0         8 301.0        335 3.54       3.570
## 32          Volvo 142E 21.4         4 121.0        109 4.11       2.780
##     qsec vs transmission gear carburetors weight
## 1  16.46  0            1    4           4   2620
## 2  17.02  0            1    4           4   2875
## 3  18.61  1            1    4           1   2320
## 4  19.44  1            0    3           1   3215
## 5  17.02  0            0    3           2   3440
## 6  20.22  1            0    3           1   3460
## 7  15.84  0            0    3           4   3570
## 8  20.00  1            0    4           2   3190
## 9  22.90  1            0    4           2   3150
## 10 18.30  1            0    4           4   3440
## 11 18.90  1            0    4           4   3440
## 12 17.40  0            0    3           3   4070
## 13 17.60  0            0    3           3   3730
## 14 18.00  0            0    3           3   3780
```

```
## 15 17.98  0           0    3          4   5250
## 16 17.82  0           0    3          4   5424
## 17 17.42  0           0    3          4   5345
## 18 19.47  1           1    4          1   2200
## 19 18.52  1           1    4          2   1615
## 20 19.90  1           1    4          1   1835
## 21 20.01  1           0    3          1   2465
## 22 16.87  0           0    3          2   3520
## 23 17.30  0           0    3          2   3435
## 24 15.41  0           0    3          4   3840
## 25 17.05  0           0    3          2   3845
## 26 18.90  1           1    4          1   1935
## 27 16.70  0           1    5          2   2140
## 28 16.90  1           1    5          2   1513
## 29 14.50  0           1    5          4   3170
## 30 15.50  0           1    5          6   2770
## 31 14.60  0           1    5          8   3570
## 32 18.60  1           1    4          2   2780
```

```
# can use 2 variables (though this isn't super meaningful for this particular dataset)
mutate(mtcars_tidy, cyl_x_gear = cylinders*gear)
```

```
##                  model  mpg cylinders  disp horsepower drat weight_1000
## 1            Mazda RX4 21.0         6 160.0        110 3.90       2.620
## 2        Mazda RX4 Wag 21.0         6 160.0        110 3.90       2.875
## 3           Datsun 710 22.8         4 108.0         93 3.85       2.320
## 4       Hornet 4 Drive 21.4         6 258.0        110 3.08       3.215
## 5    Hornet Sportabout 18.7         8 360.0        175 3.15       3.440
## 6              Valiant 18.1         6 225.0        105 2.76       3.460
## 7           Duster 360 14.3         8 360.0        245 3.21       3.570
## 8            Merc 240D 24.4         4 146.7         62 3.69       3.190
## 9             Merc 230 22.8         4 140.8         95 3.92       3.150
## 10            Merc 280 19.2         6 167.6        123 3.92       3.440
## 11           Merc 280C 17.8         6 167.6        123 3.92       3.440
## 12          Merc 450SE 16.4         8 275.8        180 3.07       4.070
## 13          Merc 450SL 17.3         8 275.8        180 3.07       3.730
## 14         Merc 450SLC 15.2         8 275.8        180 3.07       3.780
## 15  Cadillac Fleetwood 10.4         8 472.0        205 2.93       5.250
## 16 Lincoln Continental 10.4         8 460.0        215 3.00       5.424
## 17   Chrysler Imperial 14.7         8 440.0        230 3.23       5.345
## 18            Fiat 128 32.4         4  78.7         66 4.08       2.200
## 19         Honda Civic 30.4         4  75.7         52 4.93       1.615
## 20      Toyota Corolla 33.9         4  71.1         65 4.22       1.835
## 21       Toyota Corona 21.5         4 120.1         97 3.70       2.465
## 22    Dodge Challenger 15.5         8 318.0        150 2.76       3.520
## 23         AMC Javelin 15.2         8 304.0        150 3.15       3.435
## 24          Camaro Z28 13.3         8 350.0        245 3.73       3.840
## 25    Pontiac Firebird 19.2         8 400.0        175 3.08       3.845
## 26            Fiat X1-9 27.3         4  79.0         66 4.08       1.935
## 27       Porsche 914-2 26.0         4 120.3         91 4.43       2.140
## 28        Lotus Europa 30.4         4  95.1        113 3.77       1.513
## 29      Ford Pantera L 15.8         8 351.0        264 4.22       3.170
## 30        Ferrari Dino 19.7         6 145.0        175 3.62       2.770
## 31       Maserati Bora 15.0         8 301.0        335 3.54       3.570
## 32          Volvo 142E 21.4         4 121.0        109 4.11       2.780
```

```
##     qsec vs transmission gear carburetors cyl_x_gear
## 1  16.46  0            1    4           4         24
## 2  17.02  0            1    4           4         24
## 3  18.61  1            1    4           1         16
## 4  19.44  1            0    3           1         18
## 5  17.02  0            0    3           2         24
## 6  20.22  1            0    3           1         18
## 7  15.84  0            0    3           4         24
## 8  20.00  1            0    4           2         16
## 9  22.90  1            0    4           2         16
## 10 18.30  1            0    4           4         24
## 11 18.90  1            0    4           4         24
## 12 17.40  0            0    3           3         24
## 13 17.60  0            0    3           3         24
## 14 18.00  0            0    3           3         24
## 15 17.98  0            0    3           4         24
## 16 17.82  0            0    3           4         24
## 17 17.42  0            0    3           4         24
## 18 19.47  1            1    4           1         16
## 19 18.52  1            1    4           2         16
## 20 19.90  1            1    4           1         16
## 21 20.01  1            0    3           1         12
## 22 16.87  0            0    3           2         24
## 23 17.30  0            0    3           2         24
## 24 15.41  0            0    3           4         24
## 25 17.05  0            0    3           2         24
## 26 18.90  1            1    4           1         16
## 27 16.70  0            1    5           2         20
## 28 16.90  1            1    5           2         20
## 29 14.50  0            1    5           4         40
## 30 15.50  0            1    5           6         30
## 31 14.60  0            1    5           8         40
## 32 18.60  1            1    4           2         16
```

```
# can combine this with ifelse()
mutate(mtcars_tidy, engine = ifelse(vs==0, "V", "S"))
```

```
##                 model  mpg cylinders  disp horsepower drat weight_1000
## 1           Mazda RX4 21.0         6 160.0        110 3.90       2.620
## 2       Mazda RX4 Wag 21.0         6 160.0        110 3.90       2.875
## 3          Datsun 710 22.8         4 108.0         93 3.85       2.320
## 4      Hornet 4 Drive 21.4         6 258.0        110 3.08       3.215
## 5   Hornet Sportabout 18.7         8 360.0        175 3.15       3.440
## 6             Valiant 18.1         6 225.0        105 2.76       3.460
## 7          Duster 360 14.3         8 360.0        245 3.21       3.570
## 8           Merc 240D 24.4         4 146.7         62 3.69       3.190
## 9            Merc 230 22.8         4 140.8         95 3.92       3.150
## 10           Merc 280 19.2         6 167.6        123 3.92       3.440
## 11          Merc 280C 17.8         6 167.6        123 3.92       3.440
## 12         Merc 450SE 16.4         8 275.8        180 3.07       4.070
## 13         Merc 450SL 17.3         8 275.8        180 3.07       3.730
## 14        Merc 450SLC 15.2         8 275.8        180 3.07       3.780
## 15 Cadillac Fleetwood 10.4         8 472.0        205 2.93       5.250
## 16 Lincoln Continental 10.4        8 460.0        215 3.00       5.424
## 17  Chrysler Imperial 14.7         8 440.0        230 3.23       5.345
```

```
## 18            Fiat 128 32.4         4  78.7         66 4.08       2.200
## 19         Honda Civic 30.4         4  75.7         52 4.93       1.615
## 20      Toyota Corolla 33.9         4  71.1         65 4.22       1.835
## 21       Toyota Corona 21.5         4 120.1         97 3.70       2.465
## 22    Dodge Challenger 15.5         8 318.0        150 2.76       3.520
## 23         AMC Javelin 15.2         8 304.0        150 3.15       3.435
## 24          Camaro Z28 13.3         8 350.0        245 3.73       3.840
## 25    Pontiac Firebird 19.2         8 400.0        175 3.08       3.845
## 26           Fiat X1-9 27.3         4  79.0         66 4.08       1.935
## 27       Porsche 914-2 26.0         4 120.3         91 4.43       2.140
## 28        Lotus Europa 30.4         4  95.1        113 3.77       1.513
## 29      Ford Pantera L 15.8         8 351.0        264 4.22       3.170
## 30        Ferrari Dino 19.7         6 145.0        175 3.62       2.770
## 31       Maserati Bora 15.0         8 301.0        335 3.54       3.570
## 32          Volvo 142E 21.4         4 121.0        109 4.11       2.780
##     qsec vs transmission gear carburetors engine
## 1  16.46  0            1    4           4      V
## 2  17.02  0            1    4           4      V
## 3  18.61  1            1    4           1      S
## 4  19.44  1            0    3           1      S
## 5  17.02  0            0    3           2      V
## 6  20.22  1            0    3           1      S
## 7  15.84  0            0    3           4      V
## 8  20.00  1            0    4           2      S
## 9  22.90  1            0    4           2      S
## 10 18.30  1            0    4           4      S
## 11 18.90  1            0    4           4      S
## 12 17.40  0            0    3           3      V
## 13 17.60  0            0    3           3      V
## 14 18.00  0            0    3           3      V
## 15 17.98  0            0    3           4      V
## 16 17.82  0            0    3           4      V
## 17 17.42  0            0    3           4      V
## 18 19.47  1            1    4           1      S
## 19 18.52  1            1    4           2      S
## 20 19.90  1            1    4           1      S
## 21 20.01  1            0    3           1      S
## 22 16.87  0            0    3           2      V
## 23 17.30  0            0    3           2      V
## 24 15.41  0            0    3           4      V
## 25 17.05  0            0    3           2      V
## 26 18.90  1            1    4           1      S
## 27 16.70  0            1    5           2      V
## 28 16.90  1            1    5           2      S
## 29 14.50  0            1    5           4      V
## 30 15.50  0            1    5           6      V
## 31 14.60  0            1    5           8      V
## 32 18.60  1            1    4           2      S
```

```
transmute(mtcars_tidy, weight = weight_1000*1000)
```

```
##    weight
## 1    2620
## 2    2875
## 3    2320
```

```
## 4    3215
## 5    3440
## 6    3460
## 7    3570
## 8    3190
## 9    3150
## 10   3440
## 11   3440
## 12   4070
## 13   3730
## 14   3780
## 15   5250
## 16   5424
## 17   5345
## 18   2200
## 19   1615
## 20   1835
## 21   2465
## 22   3520
## 23   3435
## 24   3840
## 25   3845
## 26   1935
## 27   2140
## 28   1513
## 29   3170
## 30   2770
## 31   3570
## 32   2780
```

You can create more than one column at at time, and even use columns you've just created to create even more columns in the same command:

```
mutate(mtcars_tidy, weight = weight_1000*1000, engine = ifelse(vs==0, "V", "S"))
```

```
##                  model  mpg cylinders  disp horsepower drat weight_1000
## 1            Mazda RX4 21.0         6 160.0        110 3.90       2.620
## 2        Mazda RX4 Wag 21.0         6 160.0        110 3.90       2.875
## 3           Datsun 710 22.8         4 108.0         93 3.85       2.320
## 4       Hornet 4 Drive 21.4         6 258.0        110 3.08       3.215
## 5    Hornet Sportabout 18.7         8 360.0        175 3.15       3.440
## 6              Valiant 18.1         6 225.0        105 2.76       3.460
## 7           Duster 360 14.3         8 360.0        245 3.21       3.570
## 8            Merc 240D 24.4         4 146.7         62 3.69       3.190
## 9             Merc 230 22.8         4 140.8         95 3.92       3.150
## 10            Merc 280 19.2         6 167.6        123 3.92       3.440
## 11           Merc 280C 17.8         6 167.6        123 3.92       3.440
## 12          Merc 450SE 16.4         8 275.8        180 3.07       4.070
## 13          Merc 450SL 17.3         8 275.8        180 3.07       3.730
## 14         Merc 450SLC 15.2         8 275.8        180 3.07       3.780
## 15  Cadillac Fleetwood 10.4         8 472.0        205 2.93       5.250
## 16 Lincoln Continental 10.4         8 460.0        215 3.00       5.424
## 17   Chrysler Imperial 14.7         8 440.0        230 3.23       5.345
## 18            Fiat 128 32.4         4  78.7         66 4.08       2.200
## 19         Honda Civic 30.4         4  75.7         52 4.93       1.615
```

```
## 20      Toyota Corolla 33.9         4  71.1         65 4.22       1.835
## 21       Toyota Corona 21.5         4 120.1         97 3.70       2.465
## 22    Dodge Challenger 15.5         8 318.0        150 2.76       3.520
## 23         AMC Javelin 15.2         8 304.0        150 3.15       3.435
## 24          Camaro Z28 13.3         8 350.0        245 3.73       3.840
## 25    Pontiac Firebird 19.2         8 400.0        175 3.08       3.845
## 26           Fiat X1-9 27.3         4  79.0         66 4.08       1.935
## 27       Porsche 914-2 26.0         4 120.3         91 4.43       2.140
## 28        Lotus Europa 30.4         4  95.1        113 3.77       1.513
## 29      Ford Pantera L 15.8         8 351.0        264 4.22       3.170
## 30        Ferrari Dino 19.7         6 145.0        175 3.62       2.770
## 31       Maserati Bora 15.0         8 301.0        335 3.54       3.570
## 32          Volvo 142E 21.4         4 121.0        109 4.11       2.780
##     qsec vs transmission gear carburetors weight engine
## 1  16.46  0            1    4           4   2620      V
## 2  17.02  0            1    4           4   2875      V
## 3  18.61  1            1    4           1   2320      S
## 4  19.44  1            0    3           1   3215      S
## 5  17.02  0            0    3           2   3440      V
## 6  20.22  1            0    3           1   3460      S
## 7  15.84  0            0    3           4   3570      V
## 8  20.00  1            0    4           2   3190      S
## 9  22.90  1            0    4           2   3150      S
## 10 18.30  1            0    4           4   3440      S
## 11 18.90  1            0    4           4   3440      S
## 12 17.40  0            0    3           3   4070      V
## 13 17.60  0            0    3           3   3730      V
## 14 18.00  0            0    3           3   3780      V
## 15 17.98  0            0    3           4   5250      V
## 16 17.82  0            0    3           4   5424      V
## 17 17.42  0            0    3           4   5345      V
## 18 19.47  1            1    4           1   2200      S
## 19 18.52  1            1    4           2   1615      S
## 20 19.90  1            1    4           1   1835      S
## 21 20.01  1            0    3           1   2465      S
## 22 16.87  0            0    3           2   3520      V
## 23 17.30  0            0    3           2   3435      V
## 24 15.41  0            0    3           4   3840      V
## 25 17.05  0            0    3           2   3845      V
## 26 18.90  1            1    4           1   1935      S
## 27 16.70  0            1    5           2   2140      V
## 28 16.90  1            1    5           2   1513      S
## 29 14.50  0            1    5           4   3170      V
## 30 15.50  0            1    5           6   2770      V
## 31 14.60  0            1    5           8   3570      V
## 32 18.60  1            1    4           2   2780      S
```

```
mutate(mtcars_tidy, engine = ifelse(vs==0, "V", "S"), engine_config = paste(engine, cylinders, sep = ""
```

```
##                 model  mpg cylinders  disp horsepower drat weight_1000
## 1           Mazda RX4 21.0         6 160.0        110 3.90       2.620
## 2       Mazda RX4 Wag 21.0         6 160.0        110 3.90       2.875
## 3          Datsun 710 22.8         4 108.0         93 3.85       2.320
## 4      Hornet 4 Drive 21.4         6 258.0        110 3.08       3.215
## 5   Hornet Sportabout 18.7         8 360.0        175 3.15       3.440
```

```
## 6              Valiant 18.1         6 225.0        105 2.76       3.460
## 7           Duster 360 14.3         8 360.0        245 3.21       3.570
## 8            Merc 240D 24.4         4 146.7         62 3.69       3.190
## 9             Merc 230 22.8         4 140.8         95 3.92       3.150
## 10            Merc 280 19.2         6 167.6        123 3.92       3.440
## 11           Merc 280C 17.8         6 167.6        123 3.92       3.440
## 12          Merc 450SE 16.4         8 275.8        180 3.07       4.070
## 13          Merc 450SL 17.3         8 275.8        180 3.07       3.730
## 14         Merc 450SLC 15.2         8 275.8        180 3.07       3.780
## 15  Cadillac Fleetwood 10.4         8 472.0        205 2.93       5.250
## 16 Lincoln Continental 10.4         8 460.0        215 3.00       5.424
## 17   Chrysler Imperial 14.7         8 440.0        230 3.23       5.345
## 18            Fiat 128 32.4         4  78.7         66 4.08       2.200
## 19         Honda Civic 30.4         4  75.7         52 4.93       1.615
## 20      Toyota Corolla 33.9         4  71.1         65 4.22       1.835
## 21       Toyota Corona 21.5         4 120.1         97 3.70       2.465
## 22    Dodge Challenger 15.5         8 318.0        150 2.76       3.520
## 23         AMC Javelin 15.2         8 304.0        150 3.15       3.435
## 24          Camaro Z28 13.3         8 350.0        245 3.73       3.840
## 25    Pontiac Firebird 19.2         8 400.0        175 3.08       3.845
## 26           Fiat X1-9 27.3         4  79.0         66 4.08       1.935
## 27       Porsche 914-2 26.0         4 120.3         91 4.43       2.140
## 28        Lotus Europa 30.4         4  95.1        113 3.77       1.513
## 29      Ford Pantera L 15.8         8 351.0        264 4.22       3.170
## 30        Ferrari Dino 19.7         6 145.0        175 3.62       2.770
## 31       Maserati Bora 15.0         8 301.0        335 3.54       3.570
## 32          Volvo 142E 21.4         4 121.0        109 4.11       2.780
##     qsec vs transmission gear carburetors engine engine_config
## 1  16.46  0            1    4           4      V            V6
## 2  17.02  0            1    4           4      V            V6
## 3  18.61  1            1    4           1      S            S4
## 4  19.44  1            0    3           1      S            S6
## 5  17.02  0            0    3           2      V            V8
## 6  20.22  1            0    3           1      S            S6
## 7  15.84  0            0    3           4      V            V8
## 8  20.00  1            0    4           2      S            S4
## 9  22.90  1            0    4           2      S            S4
## 10 18.30  1            0    4           4      S            S6
## 11 18.90  1            0    4           4      S            S6
## 12 17.40  0            0    3           3      V            V8
## 13 17.60  0            0    3           3      V            V8
## 14 18.00  0            0    3           3      V            V8
## 15 17.98  0            0    3           4      V            V8
## 16 17.82  0            0    3           4      V            V8
## 17 17.42  0            0    3           4      V            V8
## 18 19.47  1            1    4           1      S            S4
## 19 18.52  1            1    4           2      S            S4
## 20 19.90  1            1    4           1      S            S4
## 21 20.01  1            0    3           1      S            S4
## 22 16.87  0            0    3           2      V            V8
## 23 17.30  0            0    3           2      V            V8
## 24 15.41  0            0    3           4      V            V8
## 25 17.05  0            0    3           2      V            V8
## 26 18.90  1            1    4           1      S            S4
```

```
## 27 16.70  0           1    5          2      V           V4
## 28 16.90  1           1    5          2      S           S4
## 29 14.50  0           1    5          4      V           V8
## 30 15.50  0           1    5          6      V           V6
## 31 14.60  0           1    5          8      V           V8
## 32 18.60  1           1    4          2      S           S4
```

Arrange

`arrange()` is used to order the rows in a data frame according to the values of one (or more) variables. Use `desc()` to get values in descending order (highest to lowest or Z to A).

```
arrange(mtcars_tidy, model)
```

```
##                  model  mpg cylinders  disp horsepower drat weight_1000
## 1          AMC Javelin 15.2         8 304.0        150 3.15       3.435
## 2   Cadillac Fleetwood 10.4         8 472.0        205 2.93       5.250
## 3           Camaro Z28 13.3         8 350.0        245 3.73       3.840
## 4    Chrysler Imperial 14.7         8 440.0        230 3.23       5.345
## 5           Datsun 710 22.8         4 108.0         93 3.85       2.320
## 6     Dodge Challenger 15.5         8 318.0        150 2.76       3.520
## 7           Duster 360 14.3         8 360.0        245 3.21       3.570
## 8         Ferrari Dino 19.7         6 145.0        175 3.62       2.770
## 9             Fiat 128 32.4         4  78.7         66 4.08       2.200
## 10           Fiat X1-9 27.3         4  79.0         66 4.08       1.935
## 11      Ford Pantera L 15.8         8 351.0        264 4.22       3.170
## 12         Honda Civic 30.4         4  75.7         52 4.93       1.615
## 13      Hornet 4 Drive 21.4         6 258.0        110 3.08       3.215
## 14   Hornet Sportabout 18.7         8 360.0        175 3.15       3.440
## 15 Lincoln Continental 10.4         8 460.0        215 3.00       5.424
## 16        Lotus Europa 30.4         4  95.1        113 3.77       1.513
## 17       Maserati Bora 15.0         8 301.0        335 3.54       3.570
## 18           Mazda RX4 21.0         6 160.0        110 3.90       2.620
## 19       Mazda RX4 Wag 21.0         6 160.0        110 3.90       2.875
## 20            Merc 230 22.8         4 140.8         95 3.92       3.150
## 21           Merc 240D 24.4         4 146.7         62 3.69       3.190
## 22            Merc 280 19.2         6 167.6        123 3.92       3.440
## 23           Merc 280C 17.8         6 167.6        123 3.92       3.440
## 24          Merc 450SE 16.4         8 275.8        180 3.07       4.070
## 25          Merc 450SL 17.3         8 275.8        180 3.07       3.730
## 26         Merc 450SLC 15.2         8 275.8        180 3.07       3.780
## 27    Pontiac Firebird 19.2         8 400.0        175 3.08       3.845
## 28       Porsche 914-2 26.0         4 120.3         91 4.43       2.140
## 29      Toyota Corolla 33.9         4  71.1         65 4.22       1.835
## 30       Toyota Corona 21.5         4 120.1         97 3.70       2.465
## 31             Valiant 18.1         6 225.0        105 2.76       3.460
## 32          Volvo 142E 21.4         4 121.0        109 4.11       2.780
##     qsec vs transmission gear carburetors
## 1  17.30  0            0    3           2
## 2  17.98  0            0    3           4
## 3  15.41  0            0    3           4
## 4  17.42  0            0    3           4
## 5  18.61  1            1    4           1
## 6  16.87  0            0    3           2
```

```
## 7  15.84  0           0    3          4
## 8  15.50  0           1    5          6
## 9  19.47  1           1    4          1
## 10 18.90  1           1    4          1
## 11 14.50  0           1    5          4
## 12 18.52  1           1    4          2
## 13 19.44  1           0    3          1
## 14 17.02  0           0    3          2
## 15 17.82  0           0    3          4
## 16 16.90  1           1    5          2
## 17 14.60  0           1    5          8
## 18 16.46  0           1    4          4
## 19 17.02  0           1    4          4
## 20 22.90  1           0    4          2
## 21 20.00  1           0    4          2
## 22 18.30  1           0    4          4
## 23 18.90  1           0    4          4
## 24 17.40  0           0    3          3
## 25 17.60  0           0    3          3
## 26 18.00  0           0    3          3
## 27 17.05  0           0    3          2
## 28 16.70  0           1    5          2
## 29 19.90  1           1    4          1
## 30 20.01  1           0    3          1
## 31 20.22  1           0    3          1
## 32 18.60  1           1    4          2
```

```
arrange(mtcars_tidy, cylinders, desc(mpg))
```

```
##                  model  mpg cylinders  disp horsepower drat weight_1000
## 1       Toyota Corolla 33.9         4  71.1         65 4.22       1.835
## 2            Fiat 128 32.4         4  78.7         66 4.08       2.200
## 3          Honda Civic 30.4         4  75.7         52 4.93       1.615
## 4        Lotus Europa 30.4         4  95.1        113 3.77       1.513
## 5           Fiat X1-9 27.3         4  79.0         66 4.08       1.935
## 6       Porsche 914-2 26.0         4 120.3         91 4.43       2.140
## 7           Merc 240D 24.4         4 146.7         62 3.69       3.190
## 8          Datsun 710 22.8         4 108.0         93 3.85       2.320
## 9            Merc 230 22.8         4 140.8         95 3.92       3.150
## 10      Toyota Corona 21.5         4 120.1         97 3.70       2.465
## 11         Volvo 142E 21.4         4 121.0        109 4.11       2.780
## 12     Hornet 4 Drive 21.4         6 258.0        110 3.08       3.215
## 13          Mazda RX4 21.0         6 160.0        110 3.90       2.620
## 14      Mazda RX4 Wag 21.0         6 160.0        110 3.90       2.875
## 15       Ferrari Dino 19.7         6 145.0        175 3.62       2.770
## 16           Merc 280 19.2         6 167.6        123 3.92       3.440
## 17            Valiant 18.1         6 225.0        105 2.76       3.460
## 18          Merc 280C 17.8         6 167.6        123 3.92       3.440
## 19   Pontiac Firebird 19.2         8 400.0        175 3.08       3.845
## 20  Hornet Sportabout 18.7         8 360.0        175 3.15       3.440
## 21         Merc 450SL 17.3         8 275.8        180 3.07       3.730
## 22         Merc 450SE 16.4         8 275.8        180 3.07       4.070
## 23     Ford Pantera L 15.8         8 351.0        264 4.22       3.170
## 24   Dodge Challenger 15.5         8 318.0        150 2.76       3.520
## 25        Merc 450SLC 15.2         8 275.8        180 3.07       3.780
```

```
## 26         AMC Javelin 15.2         8 304.0        150 3.15       3.435
## 27       Maserati Bora 15.0         8 301.0        335 3.54       3.570
## 28   Chrysler Imperial 14.7         8 440.0        230 3.23       5.345
## 29          Duster 360 14.3         8 360.0        245 3.21       3.570
## 30          Camaro Z28 13.3         8 350.0        245 3.73       3.840
## 31  Cadillac Fleetwood 10.4         8 472.0        205 2.93       5.250
## 32 Lincoln Continental 10.4         8 460.0        215 3.00       5.424
##     qsec vs transmission gear carburetors
## 1  19.90  1            1    4           1
## 2  19.47  1            1    4           1
## 3  18.52  1            1    4           2
## 4  16.90  1            1    5           2
## 5  18.90  1            1    4           1
## 6  16.70  0            1    5           2
## 7  20.00  1            0    4           2
## 8  18.61  1            1    4           1
## 9  22.90  1            0    4           2
## 10 20.01  1            0    3           1
## 11 18.60  1            1    4           2
## 12 19.44  1            0    3           1
## 13 16.46  0            1    4           4
## 14 17.02  0            1    4           4
## 15 15.50  0            1    5           6
## 16 18.30  1            0    4           4
## 17 20.22  1            0    3           1
## 18 18.90  1            0    4           4
## 19 17.05  0            0    3           2
## 20 17.02  0            0    3           2
## 21 17.60  0            0    3           3
## 22 17.40  0            0    3           3
## 23 14.50  0            1    5           4
## 24 16.87  0            0    3           2
## 25 18.00  0            0    3           3
## 26 17.30  0            0    3           2
## 27 14.60  0            1    5           8
## 28 17.42  0            0    3           4
## 29 15.84  0            0    3           4
## 30 15.41  0            0    3           4
## 31 17.98  0            0    3           4
## 32 17.82  0            0    3           4
```

Distinct

`distinct()` returns all unique rows of a data frame, dropping duplicates.

```
redundant_data <- data.frame(a=c(1,1,1,2,2,2), b=c(1,1,2,3,3,3))
head(redundant_data)
```

```
##   a b
## 1 1 1
## 2 1 1
## 3 1 2
## 4 2 3
## 5 2 3
```

```
## 6 2 3
```

```
distinct(redundant_data)
```

```
##   a b
## 1 1 1
## 2 1 2
## 3 2 3
```

By itself, `distinct()` isn't all that useful.

Summarize

`summarize()` applies one or more summary functions to a table (`min()`, `max()`, `mean()`, etc.)

```
summarize(mtcars_tidy, mean_mpg = mean(mpg), max_horsepower = max(horsepower))
```

```
##   mean_mpg max_horsepower
## 1 20.09062            335
```

By itself, `summarize()` can be useful but isn't all that exciting.

Group_by

`group_by()` invisibly groups a table by one or more variables; if you look at the output, the rows are in the exact same order, but R knows that they are now in groups.

```
group_by(mtcars_tidy, vs, transmission)
```

```
## # A tibble: 32 x 12
## # Groups:   vs, transmission [4]
##                model   mpg cylinders  disp horsepower  drat weight_1000
##                <chr> <dbl>     <dbl> <dbl>      <dbl> <dbl>       <dbl>
##  1         Mazda RX4  21.0         6 160.0        110  3.90       2.620
##  2     Mazda RX4 Wag  21.0         6 160.0        110  3.90       2.875
##  3        Datsun 710  22.8         4 108.0         93  3.85       2.320
##  4    Hornet 4 Drive  21.4         6 258.0        110  3.08       3.215
##  5 Hornet Sportabout  18.7         8 360.0        175  3.15       3.440
##  6           Valiant  18.1         6 225.0        105  2.76       3.460
##  7        Duster 360  14.3         8 360.0        245  3.21       3.570
##  8         Merc 240D  24.4         4 146.7         62  3.69       3.190
##  9          Merc 230  22.8         4 140.8         95  3.92       3.150
## 10          Merc 280  19.2         6 167.6        123  3.92       3.440
## # ... with 22 more rows, and 5 more variables: qsec <dbl>, vs <dbl>,
## #   transmission <dbl>, gear <dbl>, carburetors <dbl>
```

```
# compare to arrange()
arrange(mtcars_tidy, vs, transmission)
```

```
##                 model  mpg cylinders  disp horsepower drat weight_1000
## 1    Hornet Sportabout 18.7         8 360.0        175 3.15       3.440
## 2           Duster 360 14.3         8 360.0        245 3.21       3.570
## 3           Merc 450SE 16.4         8 275.8        180 3.07       4.070
## 4           Merc 450SL 17.3         8 275.8        180 3.07       3.730
## 5          Merc 450SLC 15.2         8 275.8        180 3.07       3.780
## 6   Cadillac Fleetwood 10.4         8 472.0        205 2.93       5.250
## 7  Lincoln Continental 10.4         8 460.0        215 3.00       5.424
```

```
## 8    Chrysler Imperial 14.7         8 440.0        230 3.23       5.345
## 9     Dodge Challenger 15.5         8 318.0        150 2.76       3.520
## 10         AMC Javelin 15.2         8 304.0        150 3.15       3.435
## 11          Camaro Z28 13.3         8 350.0        245 3.73       3.840
## 12    Pontiac Firebird 19.2         8 400.0        175 3.08       3.845
## 13           Mazda RX4 21.0         6 160.0        110 3.90       2.620
## 14       Mazda RX4 Wag 21.0         6 160.0        110 3.90       2.875
## 15       Porsche 914-2 26.0         4 120.3         91 4.43       2.140
## 16      Ford Pantera L 15.8         8 351.0        264 4.22       3.170
## 17        Ferrari Dino 19.7         6 145.0        175 3.62       2.770
## 18       Maserati Bora 15.0         8 301.0        335 3.54       3.570
## 19      Hornet 4 Drive 21.4         6 258.0        110 3.08       3.215
## 20             Valiant 18.1         6 225.0        105 2.76       3.460
## 21           Merc 240D 24.4         4 146.7         62 3.69       3.190
## 22            Merc 230 22.8         4 140.8         95 3.92       3.150
## 23            Merc 280 19.2         6 167.6        123 3.92       3.440
## 24           Merc 280C 17.8         6 167.6        123 3.92       3.440
## 25       Toyota Corona 21.5         4 120.1         97 3.70       2.465
## 26          Datsun 710 22.8         4 108.0         93 3.85       2.320
## 27            Fiat 128 32.4         4  78.7         66 4.08       2.200
## 28         Honda Civic 30.4         4  75.7         52 4.93       1.615
## 29      Toyota Corolla 33.9         4  71.1         65 4.22       1.835
## 30           Fiat X1-9 27.3         4  79.0         66 4.08       1.935
## 31       Lotus Europa 30.4         4  95.1        113 3.77       1.513
## 32         Volvo 142E 21.4         4 121.0        109 4.11       2.780
##     qsec vs transmission gear carburetors
## 1  17.02  0            0    3           2
## 2  15.84  0            0    3           4
## 3  17.40  0            0    3           3
## 4  17.60  0            0    3           3
## 5  18.00  0            0    3           3
## 6  17.98  0            0    3           4
## 7  17.82  0            0    3           4
## 8  17.42  0            0    3           4
## 9  16.87  0            0    3           2
## 10 17.30  0            0    3           2
## 11 15.41  0            0    3           4
## 12 17.05  0            0    3           2
## 13 16.46  0            1    4           4
## 14 17.02  0            1    4           4
## 15 16.70  0            1    5           2
## 16 14.50  0            1    5           4
## 17 15.50  0            1    5           6
## 18 14.60  0            1    5           8
## 19 19.44  1            0    3           1
## 20 20.22  1            0    3           1
## 21 20.00  1            0    4           2
## 22 22.90  1            0    4           2
## 23 18.30  1            0    4           4
## 24 18.90  1            0    4           4
## 25 20.01  1            0    3           1
## 26 18.61  1            1    4           1
## 27 19.47  1            1    4           1
## 28 18.52  1            1    4           2
```

```
## 29 19.90  1            1    4           1
## 30 18.90  1            1    4           1
## 31 16.90  1            1    5           2
## 32 18.60  1            1    4           2
```

`group_by()` is completely useless on its own, which is why we need to talk about piping...

2. The pipe %>%

The pipe (%>%) is an operator that performs a very simple action: it takes the output of the preceding function and inserts it as the first argument of the following function.

```
filter(mtcars_tidy, cylinders == 8) %>%
  select(model, mpg, cylinders) # no need to supply the data frame as it is piped from the previous lin
```

```
##                  model  mpg cylinders
## 1    Hornet Sportabout 18.7         8
## 2              Duster 360 14.3         8
## 3              Merc 450SE 16.4         8
## 4              Merc 450SL 17.3         8
## 5             Merc 450SLC 15.2         8
## 6   Cadillac Fleetwood 10.4         8
## 7  Lincoln Continental 10.4         8
## 8    Chrysler Imperial 14.7         8
## 9     Dodge Challenger 15.5         8
## 10         AMC Javelin 15.2         8
## 11          Camaro Z28 13.3         8
## 12    Pontiac Firebird 19.2         8
## 13      Ford Pantera L 15.8         8
## 14       Maserati Bora 15.0         8
```

Since all of the tidyverse functions have the data frame as their first argument, it's easy to combine them with the pipe. You can link together as many commands as you like and run them all together, without creating any intermediate variables or wrapping functions around each other (`filter(select(mutate(...)))` would get old real fast...)

```
select(mtcars_tidy, -disp, -drat, -qsec, -carburetors) %>%
  mutate(engine = ifelse(vs==0, "V", "S"), weight = weight_1000*1000) %>%
  filter(engine == "V", horsepower > 200) %>%
  arrange(desc(mpg))
```

```
##                 model  mpg cylinders horsepower weight_1000 vs
## 1      Ford Pantera L 15.8         8        264       3.170  0
## 2       Maserati Bora 15.0         8        335       3.570  0
## 3   Chrysler Imperial 14.7         8        230       5.345  0
## 4            Duster 360 14.3         8        245       3.570  0
## 5            Camaro Z28 13.3         8        245       3.840  0
## 6  Cadillac Fleetwood 10.4         8        205       5.250  0
## 7 Lincoln Continental 10.4         8        215       5.424  0
##   transmission gear engine weight
## 1            1    5      V   3170
## 2            1    5      V   3570
## 3            0    3      V   5345
## 4            0    3      V   3570
## 5            0    3      V   3840
## 6            0    3      V   5250
```

```
## 7            0    3      V   5424
```

`group_by()` becomes an incredibly powerful tool when you pipe the output to `summarize()` or `distinct()` - instead of being applied to the whole table, these functions are instead applied to each group separately.

```
group_by(mtcars_tidy, cylinders) %>%
  summarize(mean_mpg = mean(mpg))
```

```
## # A tibble: 3 x 2
##   cylinders mean_mpg
##       <dbl>    <dbl>
## 1         4 26.66364
## 2         6 19.74286
## 3         8 15.10000
```

```
# gives you the mean for each group

group_by(mtcars_tidy, cylinders, vs) %>%
  distinct(transmission)
```

```
## # A tibble: 7 x 3
## # Groups:   cylinders, vs [5]
##   cylinders    vs transmission
##       <dbl> <dbl>        <dbl>
## 1         6     0            1
## 2         4     1            1
## 3         6     1            0
## 4         8     0            0
## 5         4     1            0
## 6         4     0            1
## 7         8     0            1
```

```
# gives you all the unique combinations of cylinder number, engine configuration, and transmission type
```

And the pipe isn't just for use with `dplyr` functions - it's easy to use with anything in the tidyverse. Here, I'm piping the output of `gather()` into a bunch of `dplyr` functions, but this will become particularly useful when combined with ggplot.

```
gather(le, raw_year, life_expectancy, -country) %>%
  mutate(year = as.numeric(str_sub(raw_year, 2)), country = as.character(country)) %>%
  select(country, year, life_expectancy) %>%
  filter(year < 2000) -> le_50s_thru_90s
sample_n(le_50s_thru_90s, 15)
```

```
##                     country year life_expectancy
## 2355                Nigeria 1962           41.61
## 9530                  Chile 1998           76.60
## 1866                Denmark 1960           72.28
## 6771           Macao, China 1984           73.75
## 152  Sao Tome and Principe 1951           46.10
## 673                   Ghana 1954           43.30
## 9833               Pakistan 1999           62.10
## 8290                  Aruba 1992           73.54
## 4249                Armenia 1972           72.02
## 7093               Botswana 1986           62.70
## 8001             Mozambique 1990           51.50
## 9445           Saudi Arabia 1997           76.00
## 4154             Mauritania 1971           50.25
```

```
## 1885                 Ghana 1960           46.34
## 9022           North Korea 1995           58.60
```

```
# you can use the assignment operator at the start or at the end
# I find putting it at the end more readable, but here's an example of putting it at the start:
le_50s_thru_90s <- gather(le, raw_year, life_expectancy, -country) %>%
  mutate(year = as.numeric(str_sub(raw_year, 2))) %>%
  select(country, year, life_expectancy) %>%
  filter(year < 2000)
```

You can also use the pipe with functions from base R and other packages, even ones that don't have data as the first argument. Just use . to tell R where to put the data it's piping.

```
data.frame(a=c(1,2), b=c(1,2)) %>%
  list("a", .)
```

```
## [[1]]
## [1] "a"
##
## [[2]]
##   a b
## 1 1 1
## 2 2 2
```

3. Working with multiple tables

In addition to its single-table verbs, `dplyr()` also has a set of functions that are used for combining tables in different ways. These come in a few different types:

1. Functions that combine columns
2. Functions that combine or otherwise work with rows

Functions that combine columns

`dplyr` supplies four row-aware functions that combine columns from two tables: `left_join()`, `right_join()`, `inner_join()`, and `full_join()`. The four functions differ in how they handle rows that do not match between tables. Run the following code to see the differences:

```
mice_color <- data.frame(name = c("Mouse 1", "Mouse 2", "Mouse 3", "Mouse 4"), color = c("black", "brow
mice_weight <- data.frame(name = c("Mouse 1", "Mouse 2", "Mouse 3", "Mouse 5"), weight = c(20, 25, 18, 1

# exception to dplyr's "refer to columns by name" rule: when you tell any of the following functions wh

# left_join
left_join(mice_color, mice_weight, by = "name")
```

```
##      name color weight
## 1 Mouse 1 black     20
## 2 Mouse 2 brown     25
## 3 Mouse 3 white     18
## 4 Mouse 4 brown     NA
```

```
# right_join
right_join(mice_color, mice_weight, by = "name")
```

```
##      name color weight
```

```
## 1 Mouse 1 black     20
## 2 Mouse 2 brown     25
## 3 Mouse 3 white     18
## 4 Mouse 5  <NA>     22
```

```
# inner_join
inner_join(mice_color, mice_weight, by = "name")
```

```
##      name color weight
## 1 Mouse 1 black     20
## 2 Mouse 2 brown     25
## 3 Mouse 3 white     18
```

```
# full_join
full_join(mice_color, mice_weight, by = "name")
```

```
##      name color weight
## 1 Mouse 1 black     20
## 2 Mouse 2 brown     25
## 3 Mouse 3 white     18
## 4 Mouse 4 brown     NA
## 5 Mouse 5  <NA>     22
```

```
# these also work if your tables are different lengths
mini_mice <- data.frame(name = c("Mouse 3", "Mouse 5"), weight = c(18, 22), stringsAsFactors = FALSE)
left_join(mice_color, mini_mice, by = "name")
```

```
##      name color weight
## 1 Mouse 1 black     NA
## 2 Mouse 2 brown     NA
## 3 Mouse 3 white     18
## 4 Mouse 4 brown     NA
```

`dplyr` also provides a non-row-aware function that combines columns from two tables, `bind_cols()`. Use `bind_cols()` with caution, as it will match rows solely by position - if your two tables don't have the same row order, `bind_cols()` can get you in trouble!

```
mice_age <- data.frame(name = c("Mouse 1", "Mouse 2", "Mouse 3", "Mouse 4"), age = c(5, 6, 4, 5), string

# bind_cols
bind_cols(mice_color, mice_age)
```

```
##      name color   name1 age
## 1 Mouse 1 black Mouse 1   5
## 2 Mouse 2 brown Mouse 2   6
## 3 Mouse 3 white Mouse 3   4
## 4 Mouse 4 brown Mouse 4   5
```

```
# I'm keeping both `name` columns in order to show that we got the rows right

# here's an example that gets the rows wrong
bind_cols(mice_color, mice_weight)
```

```
##      name color   name1 weight
## 1 Mouse 1 black Mouse 1     20
## 2 Mouse 2 brown Mouse 2     25
## 3 Mouse 3 white Mouse 3     18
## 4 Mouse 4 brown Mouse 5     22
```

```
# bind_cols fails if your tables aren't the exact same length
#filter(mice_color, name != "Mouse 4") %>%
#  bind_cols(mice_weight)
```

Usually, one of the row-aware functions will be a better choice than `bind_cols()`.

Functions that combine or otherwise work with rows

`dplyr` functions that combine rows from two tables come in three broad categories. First, there are functions that filter the rows of one table based on the rows of another table: `semi_join()` and `anti_join()`. These can be helpful for figuring out what rows will be kept/dropped if you use `left_join()` or `right_join()` on the tables.

```
mice_color2 <- data.frame(name = c("Mouse 1", "Mouse 2", "Mouse 3", "Mouse 5"), color = c("black", "bro

# semi_join
semi_join(mice_color, mice_color2, by = "name")
```

```
##      name color
## 1 Mouse 1 black
## 2 Mouse 2 brown
## 3 Mouse 3 white
```

```
# anti_join
anti_join(mice_color, mice_color2, by = "name")
```

```
##      name color
## 1 Mouse 4 brown
```

```
# these also work on tables that have different sets of columns
semi_join(mice_color, mice_weight, by = "name")
```

```
##      name color
## 1 Mouse 1 black
## 2 Mouse 2 brown
## 3 Mouse 3 white
```

```
# row-joining functions don't care if the tables have different numbers of rows
```

Second, there are functions that combine sets of rows from two tables based on whether they appear in one or both tables: `intersect()`, `union()`, and `setdiff()`.

```
# intersect
intersect(mice_color, mice_color2)
```

```
##      name color
## 1 Mouse 1 black
## 2 Mouse 2 brown
## 3 Mouse 3 white
```

```
# union
union(mice_color, mice_color2)
```

```
##      name color
## 1 Mouse 5 white
## 2 Mouse 4 brown
## 3 Mouse 3 white
## 4 Mouse 2 brown
```

```
## 5 Mouse 1 black
```

```
# setdiff
setdiff(mice_color, mice_color2)
```

```
##      name color
## 1 Mouse 4 brown
```

`intersect()` is similar to `semi_join()`, and `setdiff()` is similar to `anti_join()`. However, these functions only work on tables that have all the same columns.

```
# this throws an error:
#intersect(mice_color, mice_weight)
```

Lastly, there is a non-column-aware option, `bind_rows()`. Just like `bind_cols()` matches rows by position, `bind_rows()` matches columns solely by position, so you need to be sure that the columns in your tables are in the exact same order.

```
more_mice <- data.frame(name = c("Mouse 5", "Mouse 6", "Mouse 7", "Mouse 8"), color = c("white", "brown

# bind_rows
bind_rows(mice_color, more_mice)
```

```
##      name color
## 1 Mouse 1 black
## 2 Mouse 2 brown
## 3 Mouse 3 white
## 4 Mouse 4 brown
## 5 Mouse 5 white
## 6 Mouse 6 brown
## 7 Mouse 7 black
## 8 Mouse 8 black
```