

Tidy Data

01-09-2020

Quote from the book “R for Data Science”, the author said

R is an old language, and some things that were useful 10 or 20 years ago now get in your way. It's difficult to change base R without breaking existing code, so most innovation occurs in packages.

What is tidyverse?

- base R's functions are often slow and the implementations are often not consistent
- writing code in **tidyverse** style usually is more elegant
- the operations would be easily chained together using piping (more below)

So what is tidyverse?

- It is a collection of R packages which are designed to be used together.
 - **ggplot2**, for data visualisation
 - **dplyr**, for data manipulation
 - **tidyr**, for data tidying
 - **readr**, for data import
 - **purrr**, for functional programming
 - **tibble**, for tibbles, a modern re-imagining of data frames
 - **stringr**, for strings
 - **forcats**, for factors

dplyr basics

- It offers five basic verbs
 - **select**: picks variables based on their names
 - **filter**: picks cases based on their values
 - **mutate**: adds new variables that are functions of existing variables
 - **arrange**: changes the ordering of the rows
 - **summarize** or **summarise**: reduces multiple values down to a single summary
- These all combine naturally with **group_by** which allows you to perform any operation “by group”.

Obtain some Data

First of all, we need some data to work with. If the data is stored in a **csv**,

```
flights <- read_csv("flights.csv")
```

We are using the tidyverse function **read_csv** to import the **flights.csv** instead of the obsolete base function **read.csv**. - **read_csv** imports data as **tibble** which has better output - **read_csv** is often faster than **read.csv** - **read_csv** handles unicode characters better

The datasets are actually obtained from the R package **nycflights13**

```
# Airline on-time data for all flights departing NYC in 2013.
library(nycflights13)
flights
```

```
## # A tibble: 336,776 x 19
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>       <dbl>   <int>         <int>
## 1  2013     1     1     517           515         2     830           819
## 2  2013     1     1     533           529         4     850           830
## 3  2013     1     1     542           540         2     923           850
## 4  2013     1     1     544           545        -1    1004          1022
## 5  2013     1     1     554           600        -6     812           837
## 6  2013     1     1     554           558        -4     740           728
## 7  2013     1     1     555           600        -5     913           854
## 8  2013     1     1     557           600        -3     709           723
## 9  2013     1     1     557           600        -3     838           846
## 10 2013     1     1     558           600        -2     753           745
## # ... with 336,766 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
```

select: picks variables based on their names.

To select arrival and departure times,

```
# old way to do it
# flights[, c("arr_time", "dep_time")]
select(flights, arr_time, dep_time)
```

```
## # A tibble: 336,776 x 2
##   arr_time dep_time
##   <int>    <int>
## 1     830      517
## 2     850      533
## 3     923      542
## 4    1004      544
## 5     812      554
## 6     740      554
## 7     913      555
## 8     709      557
## 9     838      557
## 10     753      558
## # ... with 336,766 more rows
```

I don't see why it's useful

dplyr provides a lot of helper functions,

```
# colon `:` specifies all the variables between the columns of `dep_time` and `arr_time`
select(flights, dep_time:arr_time)
```

```
## # A tibble: 336,776 x 4
##   dep_time sched_dep_time dep_delay arr_time
##   <int>         <int>         <dbl>    <int>
## 1     517           515           2      830
## 2     533           529           4      850
## 3     542           540           2      923
## 4     544           545          -1     1004
## 5     554           600          -6      812
## 6     554           558          -4      740
## 7     555           600          -5      913
## 8     557           600          -3      709
## 9     557           600          -3      838
## 10    558           600          -2      753
## # ... with 336,766 more rows
```

```
# all the columns start with arr_
select(flights, starts_with("arr_"))
```

```
## # A tibble: 336,776 x 2
##   arr_time arr_delay
##   <int>     <dbl>
## 1     830         11
## 2     850         20
## 3     923         33
## 4    1004        -18
## 5     812        -25
## 6     740         12
## 7     913         19
## 8     709        -14
## 9     838         -8
## 10    753          8
## # ... with 336,766 more rows
```

```
# all the columns end with _time
select(flights, ends_with("_time"))
```

```
## # A tibble: 336,776 x 5
##   dep_time sched_dep_time arr_time sched_arr_time air_time
##   <int>         <int>     <int>         <int>     <dbl>
## 1     517           515      830           819       227
## 2     533           529      850           830       227
## 3     542           540      923           850       160
## 4     544           545     1004          1022       183
## 5     554           600      812           837       116
## 6     554           558      740           728       150
## 7     555           600      913           854       158
## 8     557           600      709           723         53
## 9     557           600      838           846       140
## 10    558           600      753           745       138
## # ... with 336,766 more rows
```

```
# all the columns contain dep
select(flights, contains("dep"))
```

```
## # A tibble: 336,776 x 3
##   dep_time sched_dep_time dep_delay
##   <int>      <int>      <dbl>
## 1     517         515         2
## 2     533         529         4
## 3     542         540         2
## 4     544         545        -1
## 5     554         600        -6
## 6     554         558        -4
## 7     555         600        -5
## 8     557         600        -3
## 9     557         600        -3
## 10    558         600        -2
## # ... with 336,766 more rows
```

```
# all the columns do not contain dep
select(flights, -contains("dep"))
```

```
## # A tibble: 336,776 x 16
##   year month   day arr_time sched_arr_time arr_delay carrier flight tailnum
##   <int> <int> <int>   <int>       <int>       <dbl> <chr>   <int> <chr>
## 1  2013     1     1     830         819         11 UA      1545 N14228
## 2  2013     1     1     850         830         20 UA      1714 N24211
## 3  2013     1     1     923         850         33 AA      1141 N619AA
## 4  2013     1     1    1004        1022        -18 B6       725 N804JB
## 5  2013     1     1     812         837        -25 DL       461 N668DN
## 6  2013     1     1     740         728         12 UA      1696 N39463
## 7  2013     1     1     913         854         19 B6       507 N516JB
## 8  2013     1     1     709         723        -14 EV      5708 N829AS
## 9  2013     1     1     838         846         -8 B6        79 N593JB
## 10 2013     1     1     753         745          8 AA       301 N3ALAA
## # ... with 336,766 more rows, and 7 more variables: origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
```

```
# using regular expression
select(flights, matches("(arr|dep)_""))
```

```
## # A tibble: 336,776 x 4
##   dep_time dep_delay arr_time arr_delay
##   <int>      <dbl>   <int>      <dbl>
## 1     517         2     830         11
## 2     533         4     850         20
## 3     542         2     923         33
## 4     544        -1    1004        -18
## 5     554        -6     812        -25
## 6     554        -4     740         12
## 7     555        -5     913         19
## 8     557        -3     709        -14
## 9     557        -3     838         -8
```

```
## 10      558      -2      753      8
## # ... with 336,766 more rows
```

```
# of course, we could select everything
select(flights, everything())
```

```
## # A tibble: 336,776 x 19
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>         <dbl>   <int>         <int>
## 1  2013     1     1     517           515           2     830           819
## 2  2013     1     1     533           529           4     850           830
## 3  2013     1     1     542           540           2     923           850
## 4  2013     1     1     544           545          -1    1004          1022
## 5  2013     1     1     554           600          -6     812           837
## 6  2013     1     1     554           558          -4     740           728
## 7  2013     1     1     555           600          -5     913           854
## 8  2013     1     1     557           600          -3     709           723
## 9  2013     1     1     557           600          -3     838           846
## 10 2013     1     1     558           600          -2     753           745
## # ... with 336,766 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
```

```
# move air_time to the front
select(flights, air_time, everything())
```

```
## # A tibble: 336,776 x 19
##   air_time year month   day dep_time sched_dep_time dep_delay arr_time
##   <dbl> <int> <int> <int>   <int>         <int>         <dbl>   <int>
## 1    227  2013     1     1     517           515           2     830
## 2    227  2013     1     1     533           529           4     850
## 3    160  2013     1     1     542           540           2     923
## 4    183  2013     1     1     544           545          -1    1004
## 5    116  2013     1     1     554           600          -6     812
## 6    150  2013     1     1     554           558          -4     740
## 7    158  2013     1     1     555           600          -5     913
## 8     53  2013     1     1     557           600          -3     709
## 9    140  2013     1     1     557           600          -3     838
## 10   138  2013     1     1     558           600          -2     753
## # ... with 336,766 more rows, and 11 more variables: sched_arr_time <int>,
## #   arr_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>, origin <chr>,
## #   dest <chr>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
```

Remarks: - if you just need a single variable, you could use `pull`. - you could use `rename` to rename columns

filter: picks cases based on their values

```
filter(flights, origin == "JFK")
```

```
## # A tibble: 111,279 x 19
```

```
##      year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##      <int> <int> <int>   <int>         <int>      <dbl>   <int>         <int>
##  1  2013     1     1     542           540         2      923           850
##  2  2013     1     1     544           545        -1     1004          1022
##  3  2013     1     1     557           600        -3      838           846
##  4  2013     1     1     558           600        -2      849           851
##  5  2013     1     1     558           600        -2      853           856
##  6  2013     1     1     558           600        -2      924           917
##  7  2013     1     1     559           559         0      702           706
##  8  2013     1     1     606           610        -4      837           845
##  9  2013     1     1     611           600        11      945           931
## 10  2013     1     1     613           610         3      925           921
## # ... with 111,269 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
```

```
filter(flights, distance > 1000)
```

```
## # A tibble: 147,105 x 19
##      year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##      <int> <int> <int>   <int>         <int>      <dbl>   <int>         <int>
##  1  2013     1     1     517           515         2      830           819
##  2  2013     1     1     533           529         4      850           830
##  3  2013     1     1     542           540         2      923           850
##  4  2013     1     1     544           545        -1     1004          1022
##  5  2013     1     1     555           600        -5      913           854
##  6  2013     1     1     558           600        -2      849           851
##  7  2013     1     1     558           600        -2      853           856
##  8  2013     1     1     558           600        -2      924           917
##  9  2013     1     1     558           600        -2      923           937
## 10  2013     1     1     559           600        -1      941           910
## # ... with 147,095 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
```

```
# note that we are using a single `&` instead of `&&` as in base R
filter(flights, origin == "JFK" & distance > 1000)
```

```
## # A tibble: 62,071 x 19
##      year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##      <int> <int> <int>   <int>         <int>      <dbl>   <int>         <int>
##  1  2013     1     1     542           540         2      923           850
##  2  2013     1     1     544           545        -1     1004          1022
##  3  2013     1     1     558           600        -2      849           851
##  4  2013     1     1     558           600        -2      853           856
##  5  2013     1     1     558           600        -2      924           917
##  6  2013     1     1     611           600        11      945           931
##  7  2013     1     1     613           610         3      925           921
##  8  2013     1     1     615           615         0     1039          1100
##  9  2013     1     1     627           630        -3     1018          1018
## 10  2013     1     1     628           630        -2     1137          1140
## # ... with 62,061 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
```

```
filter(flights, distance < 500 | distance > 1000)
```

```
## # A tibble: 227,322 x 19
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>         <dbl>   <int>         <int>
## 1  2013     1     1     517           515           2     830           819
## 2  2013     1     1     533           529           4     850           830
## 3  2013     1     1     542           540           2     923           850
## 4  2013     1     1     544           545          -1    1004          1022
## 5  2013     1     1     555           600          -5     913           854
## 6  2013     1     1     557           600          -3     709           723
## 7  2013     1     1     558           600          -2     849           851
## 8  2013     1     1     558           600          -2     853           856
## 9  2013     1     1     558           600          -2     924           917
## 10 2013     1     1     558           600          -2     923           937
## # ... with 227,312 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
```

```
filter(flights, !between(distance, 500, 1000))
```

```
## # A tibble: 227,322 x 19
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>         <dbl>   <int>         <int>
## 1  2013     1     1     517           515           2     830           819
## 2  2013     1     1     533           529           4     850           830
## 3  2013     1     1     542           540           2     923           850
## 4  2013     1     1     544           545          -1    1004          1022
## 5  2013     1     1     555           600          -5     913           854
## 6  2013     1     1     557           600          -3     709           723
## 7  2013     1     1     558           600          -2     849           851
## 8  2013     1     1     558           600          -2     853           856
## 9  2013     1     1     558           600          -2     924           917
## 10 2013     1     1     558           600          -2     923           937
## # ... with 227,312 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
```

```
# only keep the complete cases
filter(flights, complete.cases(flights))
```

```
## # A tibble: 327,346 x 19
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>         <dbl>   <int>         <int>
## 1  2013     1     1     517           515           2     830           819
## 2  2013     1     1     533           529           4     850           830
## 3  2013     1     1     542           540           2     923           850
## 4  2013     1     1     544           545          -1    1004          1022
## 5  2013     1     1     554           600          -6     812           837
## 6  2013     1     1     554           558          -4     740           728
## 7  2013     1     1     555           600          -5     913           854
## 8  2013     1     1     557           600          -3     709           723
```

```
## 9 2013 1 1 557 600 -3 838 846
## 10 2013 1 1 558 600 -2 753 745
## # ... with 327,336 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
```

Chaining and piping

Very often, we will need to use multiple `dplyr` verbs, for example

```
filter(select(flights, origin, arr_time), origin == "JFK")
```

```
## # A tibble: 111,279 x 2
##   origin arr_time
##   <chr>    <int>
## 1 JFK      923
## 2 JFK     1004
## 3 JFK      838
## 4 JFK      849
## 5 JFK      853
## 6 JFK      924
## 7 JFK      702
## 8 JFK      837
## 9 JFK      945
## 10 JFK     925
## # ... with 111,269 more rows
```

```
# the pipe operator %>% increases readability
flights %>%
  select(origin, air_time) %>%
  filter(origin == "JFK") %>%
  filter(air_time < 500) %>%
  rename(airtime = air_time)
```

```
## # A tibble: 108,737 x 2
##   origin airtime
##   <chr>    <dbl>
## 1 JFK      160
## 2 JFK      183
## 3 JFK      140
## 4 JFK      149
## 5 JFK      158
## 6 JFK      345
## 7 JFK       44
## 8 JFK      128
## 9 JFK      366
## 10 JFK      175
## # ... with 108,727 more rows
```

```
# a few more examples
flights %>%
  select(origin, air_time) %>%
  filter(origin == "JFK", air_time < mean(air_time, na.rm = TRUE))
```



```
## # A tibble: 55,521 x 2
##   origin air_time
##   <chr>     <dbl>
## 1 JFK         140
## 2 JFK         149
## 3 JFK          44
## 4 JFK         128
## 5 JFK          41
## 6 JFK          63
## 7 JFK         142
## 8 JFK         147
## 9 JFK          64
## 10 JFK         54
## # ... with 55,511 more rows
```

```
mean_air_time <- flights %>%
  pull(air_time) %>%
  mean(na.rm = TRUE)
flights %>%
  select(origin, air_time) %>%
  filter(origin == "JFK", air_time > mean_air_time)
```

```
## # A tibble: 53,558 x 2
##   origin air_time
##   <chr>     <dbl>
## 1 JFK         160
## 2 JFK         183
## 3 JFK         158
## 4 JFK         345
## 5 JFK         366
## 6 JFK         175
## 7 JFK         182
## 8 JFK         330
## 9 JFK         192
## 10 JFK        323
## # ... with 53,548 more rows
```

```
# what if there is a name colision?
air_time <- flights %>%
  pull(air_time) %>%
  mean(na.rm = TRUE)
flights %>%
  select(origin, air_time) %>%
  filter(origin == "JFK", air_time > {{ air_time }})
```

```
## # A tibble: 53,558 x 2
##   origin air_time
##   <chr>     <dbl>
## 1 JFK         160
## 2 JFK         183
## 3 JFK         158
## 4 JFK         345
## 5 JFK         366
```

```
## 6 JFK 175
## 7 JFK 182
## 8 JFK 330
## 9 JFK 192
## 10 JFK 323
## # ... with 53,548 more rows
```

```
# you could use . to represent the working data frame
flights %>%
  filter(complete.cases(.))
```

```
## # A tibble: 327,346 x 19
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>         <dbl>   <int>         <int>
## 1  2013     1     1     517             515           2     830             819
## 2  2013     1     1     533             529           4     850             830
## 3  2013     1     1     542             540           2     923             850
## 4  2013     1     1     544             545          -1    1004            1022
## 5  2013     1     1     554             600          -6     812             837
## 6  2013     1     1     554             558          -4     740             728
## 7  2013     1     1     555             600          -5     913             854
## 8  2013     1     1     557             600          -3     709             723
## 9  2013     1     1     557             600          -3     838             846
## 10 2013     1     1     558             600          -2     753             745
## # ... with 327,336 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
```

In fact, this pipe operator is exported from the package `magrittr` and could be used solely

```
x1 <- 1:5
x2 <- 2:6
sqrt(sum((x2 - x1)^2))
```

```
## [1] 2.236068
```

```
(x2 - x1)^2 %>%
  sum() %>%
  sqrt()
```

```
## [1] 2.236068
```

```
# computer the binomial coefficients
n <- 5
x <- 0:5
choose(n, x)
```

```
## [1] 1 5 10 10 5 1
```

```
n %>% choose(x)
```

```
## [1] 1 5 10 10 5 1
```

```
x %>% choose(n, .)
```

```
## [1] 1 5 10 10 5 1
```

```
list(n = 5, x = 0:5) %>% {  
  choose(.$n, .$x)  
}
```

```
## [1] 1 5 10 10 5 1
```

PS: use `slice` if you want particular rows

```
flights %>% slice(100:105)
```

```
## # A tibble: 6 x 19  
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time  
##   <int> <int> <int>   <int>         <int>      <dbl>    <int>         <int>  
## 1  2013     1     1     752           759        -7      955           959  
## 2  2013     1     1     753           755        -2     1056          1110  
## 3  2013     1     1     754           759        -5     1039          1041  
## 4  2013     1     1     754           755        -1     1103          1030  
## 5  2013     1     1     758           800        -2     1053          1054  
## 6  2013     1     1     759           800        -1     1057          1127  
## # ... with 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,  
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,  
## #   hour <dbl>, minute <dbl>, time_hour <dtm>
```

mutate: adds new variables that are functions of existing variables

```
flights %>% mutate(  
  gain = arr_delay - dep_delay,  
  speed = distance / air_time * 60  
)
```

```
## # A tibble: 336,776 x 21  
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time  
##   <int> <int> <int>   <int>         <int>      <dbl>    <int>         <int>  
## 1  2013     1     1     517           515         2      830           819  
## 2  2013     1     1     533           529         4      850           830  
## 3  2013     1     1     542           540         2      923           850  
## 4  2013     1     1     544           545        -1     1004          1022  
## 5  2013     1     1     554           600        -6      812           837  
## 6  2013     1     1     554           558        -4      740           728  
## 7  2013     1     1     555           600        -5      913           854
```

```
## 8 2013 1 1 557 600 -3 709 723
## 9 2013 1 1 557 600 -3 838 846
## 10 2013 1 1 558 600 -2 753 745
## # ... with 336,766 more rows, and 13 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>,
## #   gain <dbl>, speed <dbl>
```

```
# we could refer to the columns just created
flights %>% mutate(
  gain = arr_delay - dep_delay,
  gain_per_hour = gain / (air_time / 60)
)
```

```
## # A tibble: 336,776 x 21
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>         <dbl>   <int>         <int>
## 1 2013     1     1     517           515           2     830           819
## 2 2013     1     1     533           529           4     850           830
## 3 2013     1     1     542           540           2     923           850
## 4 2013     1     1     544           545          -1    1004          1022
## 5 2013     1     1     554           600          -6     812           837
## 6 2013     1     1     554           558          -4     740           728
## 7 2013     1     1     555           600          -5     913           854
## 8 2013     1     1     557           600          -3     709           723
## 9 2013     1     1     557           600          -3     838           846
## 10 2013     1     1     558           600          -2     753           745
## # ... with 336,766 more rows, and 13 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>,
## #   gain <dbl>, gain_per_hour <dbl>
```

```
# `transmute` only keep the new variables
flights %>% transmute(
  gain = arr_delay - dep_delay,
  gain_per_hour = gain / (air_time / 60)
)
```

```
## # A tibble: 336,776 x 2
##   gain gain_per_hour
##   <dbl>         <dbl>
## 1     9         2.38
## 2    16         4.23
## 3    31        11.6
## 4   -17        -5.57
## 5   -19       -9.83
## 6    16         6.4
## 7    24         9.11
## 8   -11       -12.5
## 9    -5        -2.14
## 10   10         4.35
## # ... with 336,766 more rows
```

Six variations on ranking functions

- `row_number`: equivalent to `rank(ties.method = "first")`
- `min_rank`: equivalent to `rank(ties.method = "min")`
- `dense_rank`: like `min_rank()`, but with no gaps between ranks
- `percent_rank`: a number between 0 and 1 computed by rescaling `min_rank` to $[0, 1]$
- `cume_dist`: a cumulative distribution function. Proportion of all values less than or equal to the current rank.
- `ntile`: a rough rank, which breaks the input vector into `n` buckets

```
some_data <- tibble(  
  x = c(3, 4, 1, 3, 1)  
)  
some_data %>% mutate(row_number(), row_number(x), min_rank(x), percent_rank(x))
```

```
## # A tibble: 5 x 5  
##       x `row_number()` `row_number(x)` `min_rank(x)` `percent_rank(x)`  
##   <dbl>      <int>      <int>      <int>      <dbl>  
## 1     3          1          3          3          0.5  
## 2     4          2          5          5          1  
## 3     1          3          1          1          0  
## 4     3          4          4          3          0.5  
## 5     1          5          2          1          0
```

lead and lag

```
some_data2 <- tibble(  
  time = 1:5,  
  value = c(3, 4, 1, 3, 1)  
)  
some_data2 %>% mutate(diff1 = value - lag(value), diff2 = lead(value) - value)
```

```
## # A tibble: 5 x 4  
##   time value diff1 diff2  
##   <int> <dbl> <dbl> <dbl>  
## 1     1     3    NA     1  
## 2     2     4     1    -3  
## 3     3     1    -3     2  
## 4     4     3     2    -2  
## 5     5     1    -2    NA
```

Conditional mutation

```
flights %>% transmute(  
  arr_delay = arr_delay,  
  status = if_else(arr_delay > 0, "delayed", "on time")  
)
```

```
## # A tibble: 336,776 x 2
##   arr_delay status
##   <dbl> <chr>
## 1      11 delayed
## 2      20 delayed
## 3      33 delayed
## 4     -18 on time
## 5     -25 on time
## 6      12 delayed
## 7      19 delayed
## 8     -14 on time
## 9      -8 on time
## 10      8 delayed
## # ... with 336,766 more rows
```

```
(flight_distances <- flights %>%
  transmute(
    distance,
    distance_type = case_when(
      distance < 500 ~ "short",
      distance < 1000 ~ "mid",
      TRUE ~ "long"
    )
  ))
```

```
## # A tibble: 336,776 x 2
##   distance distance_type
##   <dbl> <chr>
## 1    1400 long
## 2    1416 long
## 3    1089 long
## 4    1576 long
## 5     762 mid
## 6     719 mid
## 7    1065 long
## 8     229 short
## 9     944 mid
## 10     733 mid
## # ... with 336,766 more rows
```

recode values

```
flight_distances %>% mutate(distance_type = recode(distance_type,
  long = "long-distance",
  mid = "mid-distance",
  short = "short-distance"
))
```

```
## # A tibble: 336,776 x 2
##   distance distance_type
##   <dbl> <chr>
```

```
## 1      1400 long-distance
## 2      1416 long-distance
## 3      1089 long-distance
## 4      1576 long-distance
## 5        762 mid-distance
## 6        719 mid-distance
## 7      1065 long-distance
## 8        229 short-distance
## 9        944 mid-distance
## 10       733 mid-distance
## # ... with 336,766 more rows
```

```
some_data %>% mutate(
  y = recode(x,
    `1` = "one",
    `3` = "three",
    `4` = "four"
  )
)
```

```
## # A tibble: 5 x 2
##       x y
##   <dbl> <chr>
## 1     3 three
## 2     4 four
## 3     1 one
## 4     3 three
## 5     1 one
```

arrange: changes the ordering of the rows

```
flights %>% arrange(year, month, day)
```

```
## # A tibble: 336,776 x 19
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>         <dbl>     <int>         <int>
## 1  2013     1     1     517             515           2         830           819
## 2  2013     1     1     533             529           4         850           830
## 3  2013     1     1     542             540           2         923           850
## 4  2013     1     1     544             545          -1        1004          1022
## 5  2013     1     1     554             600          -6         812           837
## 6  2013     1     1     554             558          -4         740           728
## 7  2013     1     1     555             600          -5         913           854
## 8  2013     1     1     557             600          -3         709           723
## 9  2013     1     1     557             600          -3         838           846
## 10 2013     1     1     558             600          -2         753           745
## # ... with 336,766 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
```

```
flights %>% arrange(desc(dep_delay))
```

```
## # A tibble: 336,776 x 19
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>      <dbl>    <int>         <int>
## 1  2013     1     9     641             900      1301     1242         1530
## 2  2013     6    15    1432            1935      1137     1607         2120
## 3  2013     1    10    1121            1635      1126     1239         1810
## 4  2013     9    20    1139            1845      1014     1457         2210
## 5  2013     7    22     845            1600      1005     1044         1815
## 6  2013     4    10    1100            1900       960     1342         2211
## 7  2013     3    17    2321             810       911      135         1020
## 8  2013     6    27     959            1900       899     1236         2226
## 9  2013     7    22    2257             759       898      121         1026
## 10 2013    12     5     756            1700       896     1058         2020
## # ... with 336,766 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
```

summarize and group_by operations

```
flights %>%
  group_by(tailnum) %>%
  summarize(n = n())
```

```
## # A tibble: 4,044 x 2
##   tailnum      n
##   <chr>    <int>
## 1 D942DN         4
## 2 NOEGMQ       371
## 3 N10156       153
## 4 N102UW        48
## 5 N103US        46
## 6 N104UW        47
## 7 N10575       289
## 8 N105UW        45
## 9 N107US        41
## 10 N108UW        60
## # ... with 4,034 more rows
```

```
flights %>%
  group_by(tailnum) %>%
  tally() # shorthand
```

```
## # A tibble: 4,044 x 2
##   tailnum      n
##   <chr>    <int>
## 1 D942DN         4
## 2 NOEGMQ       371
## 3 N10156       153
```



```
## 4 N102UW      48
## 5 N103US      46
## 6 N104UW      47
## 7 N10575     289
## 8 N105UW      45
## 9 N107US      41
## 10 N108UW     60
## # ... with 4,034 more rows
```

```
flights %>% count(tailnum) # another shorthand
```

```
## # A tibble: 4,044 x 2
##   tailnum      n
##   <chr>   <int>
## 1 D942DN      4
## 2 NOEGMQ    371
## 3 N10156    153
## 4 N102UW     48
## 5 N103US     46
## 6 N104UW     47
## 7 N10575    289
## 8 N105UW     45
## 9 N107US     41
## 10 N108UW    60
## # ... with 4,034 more rows
```

```
flights %>%
  group_by(tailnum) %>%
  summarize(
    count = n(),
    dist = mean(distance, na.rm = TRUE),
    delay = mean(arr_delay, na.rm = TRUE)
  )
```

```
## # A tibble: 4,044 x 4
##   tailnum count  dist  delay
##   <chr>   <int> <dbl> <dbl>
## 1 D942DN      4  854.  31.5
## 2 NOEGMQ    371  676.   9.98
## 3 N10156    153  758.  12.7
## 4 N102UW     48  536.   2.94
## 5 N103US     46  535.  -6.93
## 6 N104UW     47  535.   1.80
## 7 N10575    289  520.  20.7
## 8 N105UW     45  525.  -0.267
## 9 N107US     41  529.  -5.73
## 10 N108UW    60  534.  -1.25
## # ... with 4,034 more rows
```

```
flights %>%
  group_by(dest) %>%
  summarize(
```

```

planes = n_distinct(tailnum),
flights = n()
)

```

```

## # A tibble: 105 x 3
##   dest planes flights
##   <chr>   <int>   <int>
## 1 ABQ     108     254
## 2 ACK      58     265
## 3 ALB     172     439
## 4 ANC       6       8
## 5 ATL    1180    17215
## 6 AUS     993    2439
## 7 AVL     159     275
## 8 BDL     186     443
## 9 BGR      46     375
## 10 BHM      45     297
## # ... with 95 more rows

```

```

# group multiple variables

```

```

(per_day <- flights %>%
  group_by(year, month, day) %>%
  summarize(flights = n()))

```

```

## # A tibble: 365 x 4
## # Groups:   year, month [12]
##   year month   day flights
##   <int> <int> <int>   <int>
## 1  2013     1     1     842
## 2  2013     1     2     943
## 3  2013     1     3     914
## 4  2013     1     4     915
## 5  2013     1     5     720
## 6  2013     1     6     832
## 7  2013     1     7     933
## 8  2013     1     8     899
## 9  2013     1     9     902
## 10 2013     1    10     932
## # ... with 355 more rows

```

```

(per_month <- per_day %>%
  summarize(flights = sum(flights)))

```

```

## # A tibble: 12 x 3
## # Groups:   year [1]
##   year month flights
##   <int> <int>   <int>
## 1  2013     1  27004
## 2  2013     2  24951
## 3  2013     3  28834
## 4  2013     4  28330

```

```
## 5 2013 5 28796
## 6 2013 6 28243
## 7 2013 7 29425
## 8 2013 8 29327
## 9 2013 9 27574
## 10 2013 10 28889
## 11 2013 11 27268
## 12 2013 12 28135
```

```
(per_year <- per_month %>%
  summarize(flights = sum(flights)))
```

```
## # A tibble: 1 x 2
##   year flights
##   <int>   <int>
## 1  2013  336776
```

Other useful functions

```
flights %>% glimpse()
```

```
## Observations: 336,776
## Variables: 19
## $ year      <int> 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2013...
## $ month     <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
## $ day       <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
## $ dep_time  <int> 517, 533, 542, 544, 554, 554, 555, 557, 557, 558, 55...
## $ sched_dep_time <int> 515, 529, 540, 545, 600, 558, 600, 600, 600, 600, 60...
## $ dep_delay <dbl> 2, 4, 2, -1, -6, -4, -5, -3, -3, -2, -2, -2, -2, -2,...
## $ arr_time  <int> 830, 850, 923, 1004, 812, 740, 913, 709, 838, 753, 8...
## $ sched_arr_time <int> 819, 830, 850, 1022, 837, 728, 854, 723, 846, 745, 8...
## $ arr_delay <dbl> 11, 20, 33, -18, -25, 12, 19, -14, -8, 8, -2, -3, 7,...
## $ carrier   <chr> "UA", "UA", "AA", "B6", "DL", "UA", "B6", "EV", "B6"...
## $ flight    <int> 1545, 1714, 1141, 725, 461, 1696, 507, 5708, 79, 301...
## $ tailnum   <chr> "N14228", "N24211", "N619AA", "N804JB", "N668DN", "N...
## $ origin    <chr> "EWR", "LGA", "JFK", "JFK", "LGA", "EWR", "EWR", "LG...
## $ dest      <chr> "IAH", "IAH", "MIA", "BQN", "ATL", "ORD", "FLL", "IA...
## $ air_time  <dbl> 227, 227, 160, 183, 116, 150, 158, 53, 140, 138, 149...
## $ distance  <dbl> 1400, 1416, 1089, 1576, 762, 719, 1065, 229, 944, 73...
## $ hour      <dbl> 5, 5, 5, 5, 6, 5, 6, 6, 6, 6, 6, 6, 6, 6, 5, 6, 6...
## $ minute    <dbl> 15, 29, 40, 45, 0, 58, 0, 0, 0, 0, 0, 0, 0, 0, 59...
## $ time_hour <dtm> 2013-01-01 05:00:00, 2013-01-01 05:00:00, 2013-01-0...
```

```
flights %>% sample_n(5)
```

```
## # A tibble: 5 x 19
##   year month day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>      <dbl>      <int>         <int>
## 1  2013     2  25     811           815        -4       1011          1015
## 2  2013     2  20     601           600         1       916           920
```

```
## 3 2013 7 12 2112 1940 92 2241 2150
## 4 2013 9 20 1311 1315 -4 1458 1516
## 5 2013 10 13 1330 1330 0 1636 1632
## # ... with 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dtm>
```

```
## rows with smallest values of air_time with the original order preserved
flights %>% top_n(3, air_time)
```

```
## # A tibble: 4 x 19
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>      <dbl>    <int>         <int>
## 1 2013     2     6     853           900        -7     1542         1540
## 2 2013     3    15    1001          1000         1     1551         1530
## 3 2013     3    17    1006          1000         6     1607         1530
## 4 2013     3    17    1337          1335         2     1937         1836
## # ... with 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dtm>
```

Some variations of verbs

tidyverse also ships with some variations of verbs which could be useful.

```
# only select columns which are numerical
flights %>% select_if(is.numeric)
```

```
## # A tibble: 336,776 x 14
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>      <dbl>    <int>         <int>
## 1 2013     1     1     517           515         2      830         819
## 2 2013     1     1     533           529         4      850         830
## 3 2013     1     1     542           540         2      923         850
## 4 2013     1     1     544           545        -1     1004         1022
## 5 2013     1     1     554           600        -6      812         837
## 6 2013     1     1     554           558        -4      740         728
## 7 2013     1     1     555           600        -5      913         854
## 8 2013     1     1     557           600        -3      709         723
## 9 2013     1     1     557           600        -3      838         846
## 10 2013     1     1     558           600        -2      753         745
## # ... with 336,766 more rows, and 6 more variables: arr_delay <dbl>,
## #   flight <int>, air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>
```

```
flights %>% select_if(~ is.numeric(.))
```

```
## # A tibble: 336,776 x 14
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>      <dbl>    <int>         <int>
## 1 2013     1     1     517           515         2      830         819
## 2 2013     1     1     533           529         4      850         830
```

```
## 3 2013 1 1 542 540 2 923 850
## 4 2013 1 1 544 545 -1 1004 1022
## 5 2013 1 1 554 600 -6 812 837
## 6 2013 1 1 554 558 -4 740 728
## 7 2013 1 1 555 600 -5 913 854
## 8 2013 1 1 557 600 -3 709 723
## 9 2013 1 1 557 600 -3 838 846
## 10 2013 1 1 558 600 -2 753 745
## # ... with 336,766 more rows, and 6 more variables: arr_delay <dbl>,
## # flight <int>, air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>
```

```
flights %>% select_if(~ !is.numeric(.))
```

```
## # A tibble: 336,776 x 5
##   carrier tailnum origin dest time_hour
##   <chr>    <chr>   <chr> <chr> <dtm>
## 1 UA      N14228   EWR  IAH  2013-01-01 05:00:00
## 2 UA      N24211   LGA  IAH  2013-01-01 05:00:00
## 3 AA      N619AA    JFK  MIA  2013-01-01 05:00:00
## 4 B6      N804JB    JFK  BQN  2013-01-01 05:00:00
## 5 DL      N668DN    LGA  ATL  2013-01-01 06:00:00
## 6 UA      N39463    EWR  ORD  2013-01-01 05:00:00
## 7 B6      N516JB    EWR  FLL  2013-01-01 06:00:00
## 8 EV      N829AS    LGA  IAD  2013-01-01 06:00:00
## 9 B6      N593JB    JFK  MCO  2013-01-01 06:00:00
## 10 AA     N3ALAA    LGA  ORD  2013-01-01 06:00:00
## # ... with 336,766 more rows
```

```
flights %>% rename_if(is.numeric, toupper)
```

```
## # A tibble: 336,776 x 19
##   YEAR MONTH DAY DEP_TIME SCHED_DEP_TIME DEP_DELAY ARR_TIME SCHED_ARR_TIME
##   <int> <int> <int>   <int>         <int>     <dbl>   <int>         <int>
## 1 2013 1 1 517 515 2 830 819
## 2 2013 1 1 533 529 4 850 830
## 3 2013 1 1 542 540 2 923 850
## 4 2013 1 1 544 545 -1 1004 1022
## 5 2013 1 1 554 600 -6 812 837
## 6 2013 1 1 554 558 -4 740 728
## 7 2013 1 1 555 600 -5 913 854
## 8 2013 1 1 557 600 -3 709 723
## 9 2013 1 1 557 600 -3 838 846
## 10 2013 1 1 558 600 -2 753 745
## # ... with 336,766 more rows, and 11 more variables: ARR_DELAY <dbl>,
## # carrier <chr>, FLIGHT <int>, tailnum <chr>, origin <chr>, dest <chr>,
## # AIR_TIME <dbl>, DISTANCE <dbl>, HOUR <dbl>, MINUTE <dbl>, time_hour <dtm>
```

```
flights %>% mutate_if(
  ~ is.numeric(.) && is.double(.),
  round
)
```

```
## # A tibble: 336,776 x 19
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>       <dbl>   <int>         <int>
## 1  2013     1     1     517           515         2     830           819
## 2  2013     1     1     533           529         4     850           830
## 3  2013     1     1     542           540         2     923           850
## 4  2013     1     1     544           545        -1    1004          1022
## 5  2013     1     1     554           600        -6     812           837
## 6  2013     1     1     554           558        -4     740           728
## 7  2013     1     1     555           600        -5     913           854
## 8  2013     1     1     557           600        -3     709           723
## 9  2013     1     1     557           600        -3     838           846
## 10 2013     1     1     558           600        -2     753           745
## # ... with 336,766 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
```

```
flights %>% filter_if(
  ~ is.numeric(.),
  all_vars(. > 0)
)
```

```
## # A tibble: 79,090 x 19
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>       <dbl>   <int>         <int>
## 1  2013     1     1     517           515         2     830           819
## 2  2013     1     1     533           529         4     850           830
## 3  2013     1     1     542           540         2     923           850
## 4  2013     1     1     613           610         3     925           921
## 5  2013     1     1     623           610        13     920           915
## 6  2013     1     1     632           608        24     740           728
## 7  2013     1     1     715           713         2     911           850
## 8  2013     1     1     732           729         3    1041          1039
## 9  2013     1     1     732           645        47    1011           941
## 10 2013     1     1     743           730        13    1107          1100
## # ... with 79,080 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
```

```
flights %>% rename_at(
  vars(starts_with("arr_")),
  ~ str_replace(., "arr_", "arrival_")
)
```

```
## # A tibble: 336,776 x 19
##   year month   day dep_time sched_dep_time dep_delay arrival_time
##   <int> <int> <int>   <int>         <int>       <dbl>       <int>
## 1  2013     1     1     517           515         2         830
## 2  2013     1     1     533           529         4         850
## 3  2013     1     1     542           540         2         923
## 4  2013     1     1     544           545        -1        1004
## 5  2013     1     1     554           600        -6         812
## 6  2013     1     1     554           558        -4         740
```

```
## 7 2013 1 1 555 600 -5 913
## 8 2013 1 1 557 600 -3 709
## 9 2013 1 1 557 600 -3 838
## 10 2013 1 1 558 600 -2 753
## # ... with 336,766 more rows, and 12 more variables: sched_arr_time <int>,
## #   arrival_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,
## #   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,
## #   minute <dbl>, time_hour <dtm>
```

```
flights %>% filter_at(
  vars(ends_with("_time"), -air_time),
  all_vars(. >= 1200)
)
```

```
## # A tibble: 183,159 x 19
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>         <dbl>   <int>         <int>
## 1 2013     1     1    1200           1200           0    1408           1356
## 2 2013     1     1    1202           1207          -5    1318           1314
## 3 2013     1     1    1203           1205          -2    1501           1437
## 4 2013     1     1    1203           1200           3    1519           1545
## 5 2013     1     1    1204           1200           4    1500           1448
## 6 2013     1     1    1205           1200           5    1503           1505
## 7 2013     1     1    1206           1209          -3    1325           1328
## 8 2013     1     1    1211           1215          -4    1423           1413
## 9 2013     1     1    1217           1220          -3    1414           1350
## 10 2013     1     1    1217           1218          -1    1525           1529
## # ... with 183,149 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
```

Two-table verbs

There are 4 types of joins.

- `inner_join(x, y)` only includes observations that match in both x and y
- `left_join(x, y)` includes all observations in x, regardless of whether they match or not.
- `right_join(x, y)` equivalent to `left_join(y, x)`
- `full_join(x, y)` includes all observations from x and y

```
(df1 <- tibble(id = c(1, 2), v = 2:1))
```

```
## # A tibble: 2 x 2
##   id     v
##   <dbl> <int>
## 1     1     2
## 2     2     1
```

```
(df2 <- tibble(id = c(1, 3), a = 10, b = "a"))
```

```
## # A tibble: 2 x 3
##   id      a b
##   <dbl> <dbl> <chr>
## 1     1     10 a
## 2     3     10 a
```

```
df1 %>% inner_join(df2)
```

```
## Joining, by = "id"
```

```
## # A tibble: 1 x 4
##   id      v      a b
##   <dbl> <int> <dbl> <chr>
## 1     1     2     10 a
```

```
df1 %>% left_join(df2)
```

```
## Joining, by = "id"
```

```
## # A tibble: 2 x 4
##   id      v      a b
##   <dbl> <int> <dbl> <chr>
## 1     1     2     10 a
## 2     2     1     NA <NA>
```

```
df1 %>% right_join(df2)
```

```
## Joining, by = "id"
```

```
## # A tibble: 2 x 4
##   id      v      a b
##   <dbl> <int> <dbl> <chr>
## 1     1     2     10 a
## 2     3     NA     10 a
```

```
df1 %>% full_join(df2)
```

```
## Joining, by = "id"
```

```
## # A tibble: 3 x 4
##   id      v      a b
##   <dbl> <int> <dbl> <chr>
## 1     1     2     10 a
## 2     2     1     NA <NA>
## 3     3     NA     10 a
```


Tidy Data

There are three interrelated rules which make a dataset tidy

- Each variable must have its own column (long format).
- Each observation must have its own row (wide format).
- Each value must have its own cell.

Using the datasets from R for Data Science to show that the same data could be organised in different ways.

```
# make sure you have tidyr 1.0
library(tidyr)
```

`pivot_longer` is the replacement for `gather()` and `pivot_wider()` is the replacement for `spread()`. Both are designed to be simpler and can handle more cases than `gather` and `spread`.

Pivot longer

```
relig_income %>%
  pivot_longer(-religion, names_to = "income", values_to = "count")
```

```
## # A tibble: 180 x 3
##   religion income      count
##   <chr>    <chr>    <dbl>
## 1 Agnostic <$10k      27
## 2 Agnostic $10-20k     34
## 3 Agnostic $20-30k     60
## 4 Agnostic $30-40k     81
## 5 Agnostic $40-50k     76
## 6 Agnostic $50-75k    137
## 7 Agnostic $75-100k   122
## 8 Agnostic $100-150k  109
## 9 Agnostic >150k     84
## 10 Agnostic Don't know/refused 96
## # ... with 170 more rows
```

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

```
## # A tibble: 5,307 x 5
##   artist track      date.entered week rank
##   <chr>   <chr>      <date>         <int> <dbl>
## 1 2 Pac   Baby Don't Cry (Keep... 2000-02-26      1    87
```

```
## 2 2 Pac Baby Don't Cry (Keep... 2000-02-26 2 82
## 3 2 Pac Baby Don't Cry (Keep... 2000-02-26 3 72
## 4 2 Pac Baby Don't Cry (Keep... 2000-02-26 4 77
## 5 2 Pac Baby Don't Cry (Keep... 2000-02-26 5 87
## 6 2 Pac Baby Don't Cry (Keep... 2000-02-26 6 94
## 7 2 Pac Baby Don't Cry (Keep... 2000-02-26 7 99
## 8 2Ge+her The Hardest Part Of ... 2000-09-02 1 91
## 9 2Ge+her The Hardest Part Of ... 2000-09-02 2 87
## 10 2Ge+her The Hardest Part Of ... 2000-09-02 3 92
## # ... with 5,297 more rows
```

Pivot wider

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

```
## # A tibble: 19 x 12
##   fish Release I80_1 Lisbon Rstr Base_TD BCE BCW BCE2 BCW2 MAE MAW
##   <fct>   <int> <int>   <int> <int>   <int> <int> <int> <int> <int> <int> <int>
## 1 4842     1     1     1     1     1     1     1     1     1     1     1
## 2 4843     1     1     1     1     1     1     1     1     1     1     1
## 3 4844     1     1     1     1     1     1     1     1     1     1     1
## 4 4845     1     1     1     1     1     0     0     0     0     0     0
## 5 4847     1     1     1     0     0     0     0     0     0     0     0
## 6 4848     1     1     1     1     0     0     0     0     0     0     0
## 7 4849     1     1     0     0     0     0     0     0     0     0     0
## 8 4850     1     1     0     1     1     1     1     0     0     0     0
## 9 4851     1     1     0     0     0     0     0     0     0     0     0
## 10 4854     1     1     0     0     0     0     0     0     0     0     0
## 11 4855     1     1     1     1     1     0     0     0     0     0     0
## 12 4857     1     1     1     1     1     1     1     1     1     0     0
## 13 4858     1     1     1     1     1     1     1     1     1     1     1
## 14 4859     1     1     1     1     1     1     0     0     0     0     0
## 15 4861     1     1     1     1     1     1     1     1     1     1     1
## 16 4862     1     1     1     1     1     1     1     1     1     0     0
## 17 4863     1     1     0     0     0     0     0     0     0     0     0
## 18 4864     1     1     0     0     0     0     0     0     0     0     0
## 19 4865     1     1     1     0     0     0     0     0     0     0     0
```

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

```
## # A tibble: 52 x 6
##   GEOID NAME estimate_income estimate_rent moe_income moe_rent
##   <chr> <chr>           <dbl>           <dbl>       <dbl>   <dbl>
## 1 01 Alabama      24476           747         136     3
## 2 02 Alaska      32940          1200         508    13
## 3 04 Arizona      27517           972         148     4
```

##	4	05	Arkansas	23789	709	165	5
##	5	06	California	29454	1358	109	3
##	6	08	Colorado	32401	1125	109	5
##	7	09	Connecticut	35326	1123	195	5
##	8	10	Delaware	31560	1076	247	10
##	9	11	District of Columbia	43198	1424	681	17
##	10	12	Florida	25952	1077	70	3
##	#		... with 42 more rows				

References

- Documentation of dplyr <https://dplyr.tidyverse.org/>
- R for Data Science <http://r4ds.had.co.nz/tidy-data.html>