Notes_on_Ch3-DataTransformation

Norman LIm

2025-06-28

Prerequisites

```
library(nycflights13)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.2     v tibble    3.3.0
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.0.4
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

Exploring `nycflights` data

```
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
## # i 336,766 more rows
## # i 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 <dttm>
```

We can see that `flights` is a tibble.

Another way to view the dataset using `glimpse()`:

```
glimpse(flights)
```

```
## Rows: 336,776
## Columns: 19
## $ year           <int> 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2~
## $ month          <int> 1, 1, 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, 1, 1~
## $ dep_time       <int> 517, 533, 542, 544, 554, 554, 555, 557, 557, 558, 558, ~
## $ sched_dep_time <int> 515, 529, 540, 545, 600, 558, 600, 600, 600, 600, 600, ~
## $ dep_delay      <dbl> 2, 4, 2, -1, -6, -4, -5, -3, -3, -2, -2, -2, -2, -2, -1~
## $ arr_time       <int> 830, 850, 923, 1004, 812, 740, 913, 709, 838, 753, 849,~
## $ sched_arr_time <int> 819, 830, 850, 1022, 837, 728, 854, 723, 846, 745, 851,~
## $ arr_delay      <dbl> 11, 20, 33, -18, -25, 12, 19, -14, -8, 8, -2, -3, 7, -1~
## $ carrier        <chr> "UA", "UA", "AA", "B6", "DL", "UA", "B6", "EV", "B6", "~
## $ flight         <int> 1545, 1714, 1141, 725, 461, 1696, 507, 5708, 79, 301, 4~
## $ tailnum        <chr> "N14228", "N24211", "N619AA", "N804JB", "N668DN", "N394~
## $ origin         <chr> "EWR", "LGA", "JFK", "JFK", "LGA", "EWR", "EWR", "LGA",~
## $ dest           <chr> "IAH", "IAH", "MIA", "BQN", "ATL", "ORD", "FLL", "IAD",~
## $ air_time       <dbl> 227, 227, 160, 183, 116, 150, 158, 53, 140, 138, 149, 1~
## $ distance       <dbl> 1400, 1416, 1089, 1576, 762, 719, 1065, 229, 944, 733, ~
## $ hour           <dbl> 5, 5, 5, 5, 6, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 5, 6, 6, 6~
## $ minute         <dbl> 15, 29, 40, 45, 0, 58, 0, 0, 0, 0, 0, 0, 0, 0, 0, 59, 0~
## $ time_hour      <dttm> 2013-01-01 05:00:00, 2013-01-01 05:00:00, 2013-01-01 0~
```

dplyr basics

Common theme for `dplyr` verbs - first argument is always the data frame - subsequent arguments typically describe which columns to operate on - output is always a new data frame - verbs are organized onto four groups based on what they operate on: **rows, columns, groups,** or **tables**

Example:

```
flights |>
  filter(dest == "IAH") |>
  group_by(year, month, day) |>
  summarize(arr_delay = mean(arr_delay, na.rm = TRUE))
```

```
## `summarise()` has grouped output by 'year', 'month'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 365 x 4
## # Groups:   year, month [12]
##     year month   day arr_delay
##    <int> <int> <int>     <dbl>
##  1  2013     1     1     17.8
##  2  2013     1     2      7
##  3  2013     1     3     18.3
##  4  2013     1     4     -3.2
##  5  2013     1     5     20.2
##  6  2013     1     6      9.28
##  7  2013     1     7     -7.74
##  8  2013     1     8      7.79
```

```
##  9  2013     1     9     18.1
## 10  2013     1    10      6.68
## # i 355 more rows
```

Rows

`filter()`

Example: find all flights that departed more than 120 minutes late:

```
flights |>
  filter(dep_delay > 120)
```

```
## # A tibble: 9,723 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      848           1835       853     1001           1950
##  2  2013     1     1      957            733       144     1056            853
##  3  2013     1     1     1114            900       134     1447           1222
##  4  2013     1     1     1540           1338       122     2020           1825
##  5  2013     1     1     1815           1325       290     2120           1542
##  6  2013     1     1     1842           1422       260     1958           1535
##  7  2013     1     1     1856           1645       131     2212           2005
##  8  2013     1     1     1934           1725       129     2126           1855
##  9  2013     1     1     1938           1703       155     2109           1823
## 10  2013     1     1     1942           1705       157     2124           1830
## # i 9,713 more rows
## # i 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 <dttm>
```

Logical conditions can also be used:

```
flights |>
  filter(month == 1, day == 1)
```

```
## # A tibble: 842 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
## # i 832 more rows
## # i 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 <dttm>
```

```
flights |>
  filter(month == 1 | month == 2)
```

```
## # A tibble: 51,955 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
## # i 51,945 more rows
## # i 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 <dttm>
```

Tip: a useful shorcut for comibing | and == is '%in%. For example, the expression above can be written as:

```
flights |>
  filter(month %in% c(1, 2))
```

```
## # A tibble: 51,955 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
## # i 51,945 more rows
## # i 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 <dttm>
```

To save the result, we simply use the assignment operator <-:

```
jan1 <- flights |>
  filter(month == 1 & day == 1)
```

arrange()

- `arrange()` changes the order or the rows based on the value of the columns.

```
flights |>
  arrange(year, month, day, dep_time)
```

```
## # 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
## # i 336,766 more rows
## # i 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 <dttm>
```

- the `desc()` function, when combined with `arrange()`, will show the results in descending order:

```
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
## # i 336,766 more rows
## # i 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 <dttm>
```

‘distinct()

- finds all unique rows in a dataset

```
flights |>
  distinct()
```

```
## # 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
## # i 336,766 more rows
## # i 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 <dttm>
```

```
flights |>
  distinct(origin, dest)
```

```
## # A tibble: 224 x 2
##    origin dest
##    <chr>  <chr>
##  1 EWR    IAH
##  2 LGA    IAH
##  3 JFK    MIA
##  4 JFK    BQN
##  5 LGA    ATL
##  6 EWR    ORD
##  7 EWR    FLL
##  8 LGA    IAD
##  9 JFK    MCO
## 10 LGA    ORD
## # i 214 more rows
```

- Filtering for unique rows while keeping other columns:

```
flights |>
  distinct(origin, dest, .keep_all = TRUE)
```

```
## # A tibble: 224 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
## # i 214 more rows
## # i 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 <dttm>
```

Columns

Four important columns that affect the columns without changing the rows: - `mutate()` - `select()` - `rename()` - `relocate()`

Adding new column/s that are calculated from existing columns:

```
flights |>
  mutate(
    gain = dep_delay - arr_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
## # i 336,766 more rows
## # i 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 <dttm>, gain <dbl>, speed <dbl>
```

- by default, mutate adds new columns on the right-hand-side of the dataset, making it difficult to see.
- we can use the `before()` argument to instead add the new column/s on the left-hand-side:

```
flights |>
  mutate(
    gain = dep_delay - arr_delay,
    speed = distance / air_time * 60,
    .before = 1
  )
```

```
## # A tibble: 336,776 x 21
##     gain speed  year month   day dep_time sched_dep_time dep_delay arr_time
##    <dbl> <dbl> <int> <int> <int>    <int>          <int>     <dbl>    <int>
##  1    -9  370.  2013     1     1      517            515         2      830
```

```
##  2   -16  374.  2013     1     1      533            529         4      850
##  3   -31  408.  2013     1     1      542            540         2      923
##  4    17  517.  2013     1     1      544            545        -1     1004
##  5    19  394.  2013     1     1      554            600        -6      812
##  6   -16  288.  2013     1     1      554            558        -4      740
##  7   -24  404.  2013     1     1      555            600        -5      913
##  8    11  259.  2013     1     1      557            600        -3      709
##  9     5  405.  2013     1     1      557            600        -3      838
## 10   -10  319.  2013     1     1      558            600        -2      753
## # i 336,766 more rows
## # i 12 more variables: sched_arr_time <int>, arr_delay <dbl>, carrier <chr>,
## #   flight <int>, tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>,
## #   distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>
```

- we can also specify where to insert the new column/s with `.after()`:

```
flights |>
  mutate(
    gain = dep_delay - arr_delay,
    speed = distance / air_time * 60,
    .after = day
  )
```

```
## # A tibble: 336,776 x 21
##     year month   day  gain speed dep_time sched_dep_time dep_delay arr_time
##    <int> <int> <int> <dbl> <dbl>    <int>          <int>     <dbl>    <int>
##  1  2013     1     1    -9  370.      517            515         2      830
##  2  2013     1     1   -16  374.      533            529         4      850
##  3  2013     1     1   -31  408.      542            540         2      923
##  4  2013     1     1    17  517.      544            545        -1     1004
##  5  2013     1     1    19  394.      554            600        -6      812
##  6  2013     1     1   -16  288.      554            558        -4      740
##  7  2013     1     1   -24  404.      555            600        -5      913
##  8  2013     1     1    11  259.      557            600        -3      709
##  9  2013     1     1     5  405.      557            600        -3      838
## 10  2013     1     1   -10  319.      558            600        -2      753
## # i 336,766 more rows
## # i 12 more variables: sched_arr_time <int>, arr_delay <dbl>, carrier <chr>,
## #   flight <int>, tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>,
## #   distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>
```

- keeping only the newly-created columns and the other columns used in creating it/them:

```
flights |>
  mutate(
    gain = dep_delay - arr_time,
    speed = distance / air_time * 60,
    .keep = "used"
  )
```

```
## # A tibble: 336,776 x 6
##    dep_delay arr_time air_time distance  gain speed
```

```
##        <dbl>    <int>    <dbl>    <dbl> <dbl> <dbl>
##  1         2      830      227     1400  -828  370.
##  2         4      850      227     1416  -846  374.
##  3         2      923      160     1089  -921  408.
##  4        -1     1004      183     1576 -1005  517.
##  5        -6      812      116      762  -818  394.
##  6        -4      740      150      719  -744  288.
##  7        -5      913      158     1065  -918  404.
##  8        -3      709       53      229  -712  259.
##  9        -3      838      140      944  -841  405.
## 10        -2      753      138      733  -755  319.
## # i 336,766 more rows
```

select()

`select()` is used to 'select' columns.

- selecting columns by name:

```
flights |>
  select(year, month, day)
```

```
## # A tibble: 336,776 x 3
##     year month   day
##    <int> <int> <int>
##  1  2013     1     1
##  2  2013     1     1
##  3  2013     1     1
##  4  2013     1     1
##  5  2013     1     1
##  6  2013     1     1
##  7  2013     1     1
##  8  2013     1     1
##  9  2013     1     1
## 10  2013     1     1
## # i 336,766 more rows
```

- selecting all columns between two column names (inclusive):

```
flights |>
  select(year:day)
```

```
## # A tibble: 336,776 x 3
##     year month   day
##    <int> <int> <int>
##  1  2013     1     1
##  2  2013     1     1
##  3  2013     1     1
##  4  2013     1     1
##  5  2013     1     1
##  6  2013     1     1
```

```
##  7  2013     1     1
##  8  2013     1     1
##  9  2013     1     1
## 10  2013     1     1
## # i 336,766 more rows
```

- selecting all columns exept those you specify (inclusive):

```
flights |>
  select(!year:day)
```

```
## # A tibble: 336,776 x 16
##    dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier
##       <int>          <int>     <dbl>    <int>          <int>     <dbl> <chr>
##  1      517            515         2      830            819        11 UA
##  2      533            529         4      850            830        20 UA
##  3      542            540         2      923            850        33 AA
##  4      544            545        -1     1004           1022       -18 B6
##  5      554            600        -6      812            837       -25 DL
##  6      554            558        -4      740            728        12 UA
##  7      555            600        -5      913            854        19 B6
##  8      557            600        -3      709            723       -14 EV
##  9      557            600        -3      838            846        -8 B6
## 10      558            600        -2      753            745         8 AA
## # i 336,766 more rows
## # i 9 more variables: flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>
```

- note: in older versions of R use `-` instead of `!` for this.
- other useful options with
  - `select()`
  - `starts_with()`
  - `ends_with()`
  - `contains()`
  - `num_range()`: `num_range("x", 1:3)` matches `x1`, `x2`, and `x3`
- renaming as you select using the `=` operator:

```
flights |>
  select(tail_num = tailnum)
```

```
## # A tibble: 336,776 x 1
##    tail_num
##    <chr>
##  1 N14228
##  2 N24211
##  3 N619AA
##  4 N804JB
##  5 N668DN
##  6 N39463
##  7 N516JB
```

```
##  8 N829AS
##  9 N593JB
## 10 N3ALAA
## # i 336,766 more rows
```

rename()

- keep all existing variables and just rename a few:

```
flights |>
  rename(tail_num = tailnum)
```

```
## # 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
## # i 336,766 more rows
## # i 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tail_num <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>
```

relocate()

- used to move variables around to improve visibility or collect related variables

```
flights |>
  relocate(time_hour, air_time)
```

```
## # A tibble: 336,776 x 19
##    time_hour           air_time  year month   day dep_time sched_dep_time
##    <dttm>                 <dbl> <int> <int> <int>    <int>          <int>
##  1 2013-01-01 05:00:00      227  2013     1     1      517            515
##  2 2013-01-01 05:00:00      227  2013     1     1      533            529
##  3 2013-01-01 05:00:00      160  2013     1     1      542            540
##  4 2013-01-01 05:00:00      183  2013     1     1      544            545
##  5 2013-01-01 06:00:00      116  2013     1     1      554            600
##  6 2013-01-01 05:00:00      150  2013     1     1      554            558
##  7 2013-01-01 06:00:00      158  2013     1     1      555            600
##  8 2013-01-01 06:00:00       53  2013     1     1      557            600
##  9 2013-01-01 06:00:00      140  2013     1     1      557            600
## 10 2013-01-01 06:00:00      138  2013     1     1      558            600
## # i 336,766 more rows
```

```
## # i 12 more variables: dep_delay <dbl>, arr_time <int>, sched_arr_time <int>,
## #   arr_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>, origin <chr>,
## #   dest <chr>, distance <dbl>, hour <dbl>, minute <dbl>
```

- the `.before()` and `.after()` arguments works with `relocate()` too:

```
flights |>
  relocate(year:dep_time, .after = time_hour)
```

```
## # A tibble: 336,776 x 19
##    sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier flight
##             <int>     <dbl>    <int>          <int>     <dbl> <chr>    <int>
##  1            515         2      830            819        11 UA        1545
##  2            529         4      850            830        20 UA        1714
##  3            540         2      923            850        33 AA        1141
##  4            545        -1     1004           1022       -18 B6         725
##  5            600        -6      812            837       -25 DL         461
##  6            558        -4      740            728        12 UA        1696
##  7            600        -5      913            854        19 B6         507
##  8            600        -3      709            723       -14 EV        5708
##  9            600        -3      838            846        -8 B6          79
## 10            600        -2      753            745         8 AA         301
## # i 336,766 more rows
## # i 12 more variables: tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>,
## #   distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>, year <int>,
## #   month <int>, day <int>, dep_time <int>
```

```
flights |>
  relocate(starts_with("arr"), .before = dep_time)
```

```
## # A tibble: 336,776 x 19
##     year month   day arr_time arr_delay dep_time sched_dep_time dep_delay
##    <int> <int> <int>    <int>     <dbl>    <int>          <int>     <dbl>
##  1  2013     1     1      830        11      517            515         2
##  2  2013     1     1      850        20      533            529         4
##  3  2013     1     1      923        33      542            540         2
##  4  2013     1     1     1004       -18      544            545        -1
##  5  2013     1     1      812       -25      554            600        -6
##  6  2013     1     1      740        12      554            558        -4
##  7  2013     1     1      913        19      555            600        -5
##  8  2013     1     1      709       -14      557            600        -3
##  9  2013     1     1      838        -8      557            600        -3
## 10  2013     1     1      753         8      558            600        -2
## # i 336,766 more rows
## # i 11 more variables: sched_arr_time <int>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>
```

The pipe (|> or %>%)

- useful when combining "verbs"

- avoids having to nest one function call inside another
- makes the code readable

```
flights |>
  filter(dest == "IAH") |>
  mutate(speed = distance / air_time * 60) |>
  select(year:day, dep_time, carrier, flight, speed) |>
  arrange(desc(speed))
```

```
## # A tibble: 7,198 x 7
##     year month   day dep_time carrier flight speed
##    <int> <int> <int>    <int> <chr>    <int> <dbl>
##  1  2013     7     9      707 UA         226  522.
##  2  2013     8    27     1850 UA        1128  521.
##  3  2013     8    28      902 UA        1711  519.
##  4  2013     8    28     2122 UA        1022  519.
##  5  2013     6    11     1628 UA        1178  515.
##  6  2013     8    27     1017 UA         333  515.
##  7  2013     8    27     1205 UA        1421  515.
##  8  2013     8    27     1758 UA         302  515.
##  9  2013     9    27      521 UA         252  515.
## 10  2013     8    28      625 UA         559  515.
## # i 7,188 more rows
```

Groups

- most important functions are `group_by()`, `summarize()`, and the slice family of functions

`group_by()`

- divides the dataset into groups:

```
flights |>
  group_by(month)
```

```
## # A tibble: 336,776 x 19
## # Groups:   month [12]
##     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
## # i 336,766 more rows
## # i 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 <dttm>
```

`summarize()`

- calculates a single summary statistic
- reduces the data frame into 1 row of summary statistic per group

```
flights |>
  group_by(month) |>
  summarize(
    ave_delay = mean(dep_delay)
  )
```

```
## # A tibble: 12 x 2
##    month ave_delay
##    <int>     <dbl>
##  1     1        NA
##  2     2        NA
##  3     3        NA
##  4     4        NA
##  5     5        NA
##  6     6        NA
##  7     7        NA
##  8     8        NA
##  9     9        NA
## 10    10        NA
## 11    11        NA
## 12    12        NA
```

- using `summarize()` when there are `NA`s in the data:

```
flights |>
  group_by(month) |>
  summarize(
    ave_delay = mean(dep_delay, na.rm = TRUE)
  )
```

```
## # A tibble: 12 x 2
##    month ave_delay
##    <int>     <dbl>
##  1     1     10.0
##  2     2     10.8
##  3     3     13.2
##  4     4     13.9
##  5     5     13.0
##  6     6     20.8
##  7     7     21.7
##  8     8     12.6
##  9     9      6.72
## 10    10      6.24
## 11    11      5.44
## 12    12     16.6
```

- using `summarize()` and show the count per group:

```
flights |>
  group_by(month) |>
  summarize(
    ave_delay = dep_delay, na.rm = TRUE,
    n = n()
  )
```

```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

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

```
## # A tibble: 336,776 x 4
## # Groups:   month [12]
##    month ave_delay na.rm     n
##    <int>     <dbl> <lgl> <int>
##  1     1         2 TRUE  27004
##  2     1         4 TRUE  27004
##  3     1         2 TRUE  27004
##  4     1        -1 TRUE  27004
##  5     1        -6 TRUE  27004
##  6     1        -4 TRUE  27004
##  7     1        -5 TRUE  27004
##  8     1        -3 TRUE  27004
##  9     1        -3 TRUE  27004
## 10     1        -2 TRUE  27004
## # i 336,766 more rows
```

- according to the warnings, the `summarize()` was deprecated in dplyr 1.1.0.
- we can use `reframe()` instead:

```
flights |>
  group_by(month) |>
  reframe(
    ave_delay = dep_delay, na.rm = TRUE,
    n = n()
  )
```

```
## # A tibble: 336,776 x 4
##    month ave_delay na.rm     n
##    <int>     <dbl> <lgl> <int>
##  1     1         2 TRUE  27004
##  2     1         4 TRUE  27004
##  3     1         2 TRUE  27004
##  4     1        -1 TRUE  27004
##  5     1        -6 TRUE  27004
```

```
##  6     1        -4 TRUE  27004
##  7     1        -5 TRUE  27004
##  8     1        -3 TRUE  27004
##  9     1        -3 TRUE  27004
## 10     1        -2 TRUE  27004
## # i 336,766 more rows
```

the `slice()_` functions

- used to extract specific rows within each group

```
flights |>
  group_by(dest) |>
  slice_max(arr_delay, n = 1) |>
  relocate(dest)
```

```
## # A tibble: 108 x 19
## # Groups:   dest [105]
##    dest   year month   day dep_time sched_dep_time dep_delay arr_time
##    <chr> <int> <int> <int>    <int>          <int>     <dbl>    <int>
##  1 ABQ    2013     7    22     2145           2007        98      132
##  2 ACK    2013     7    23     1139            800       219     1250
##  3 ALB    2013     1    25      123           2000       323      229
##  4 ANC    2013     8    17     1740           1625        75     2042
##  5 ATL    2013     7    22     2257            759       898      121
##  6 AUS    2013     7    10     2056           1505       351     2347
##  7 AVL    2013     8    13     1156            832       204     1417
##  8 BDL    2013     2    21     1728           1316       252     1839
##  9 BGR    2013    12     1     1504           1056       248     1628
## 10 BHM    2013     4    10       25           1900       325      136
## # i 98 more rows
## # i 11 more variables: sched_arr_time <int>, arr_delay <dbl>, carrier <chr>,
## #   flight <int>, tailnum <chr>, origin <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>
```

- instead of specifying the number of item to extract using the argument `n`, we can specify the proportion instead, using `prop`

```
flights |>
  group_by(dest) |>
  slice_max(prop = 0.1, order_by = dest) |>
  relocate(dest)
```

```
## # A tibble: 336,766 x 19
## # Groups:   dest [102]
##    dest   year month   day dep_time sched_dep_time dep_delay arr_time
##    <chr> <int> <int> <int>    <int>          <int>     <dbl>    <int>
##  1 ABQ    2013    10     1     1955           2001        -6     2213
##  2 ABQ    2013    10     2     2010           2001         9     2230
##  3 ABQ    2013    10     3     1955           2001        -6     2232
##  4 ABQ    2013    10     4     2017           2001        16     2304
##  5 ABQ    2013    10     5     1959           1959         0     2226
```

```
##  6 ABQ    2013    10     6     1959           2001        -2     2234
##  7 ABQ    2013    10     7     2002           2001         1     2233
##  8 ABQ    2013    10     8     1957           2001        -4     2216
##  9 ABQ    2013    10     9     1957           2001        -4     2220
## 10 ABQ    2013    10    10     2011           2001        10     2235
## # i 336,756 more rows
## # i 11 more variables: sched_arr_time <int>, arr_delay <dbl>, carrier <chr>,
## #   flight <int>, tailnum <chr>, origin <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>
```

Grouping multiple variables

```
daily <- flights |>
  group_by(year, month, day)

daily
```

```
## # A tibble: 336,776 x 19
## # Groups:   year, month, day [365]
##     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
## # i 336,766 more rows
## # i 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 <dttm>
```

- when we summarize a tibble grouped by more than 1 variable, each summary "peels" off the last group by defailt:

```
daily_flights <- daily |>
  summarize(n = n())
```

```
## `summarise()` has grouped output by 'year', 'month'. You can override using the
## `.groups` argument.
```

```
daily_flights
```

```
## # A tibble: 365 x 4
## # Groups:   year, month [12]
##     year month   day     n
```

```
##    <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
## # i 355 more rows
```

- but we can also specify to drop or keep the grouping layers:

```
daily |>
  summarize(n = n(), .groups = "drop_last")
```

```
## # A tibble: 365 x 4
## # Groups:   year, month [12]
##     year month   day     n
##    <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
## # i 355 more rows
```

```
daily |>
  summarize(n = n(), .groups = "keep")
```

```
## # A tibble: 365 x 4
## # Groups:   year, month, day [365]
##     year month   day     n
##    <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
## # i 355 more rows
```

ungrouping

- we can also remove a grouping from a data frame using `ungroup()` instead of `summarize()`

```
daily |>
  ungroup()
```

```
## # 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
## # i 336,766 more rows
## # i 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 <dttm>
```

- to satisfy our curiosity, we can try summarizing an ungrouped data frame:

```
daily |>
  ungroup() |>
  summarize(
    avg_delay = mean(dep_delay, na.rm = TRUE),
    flights = n()
  )
```

```
## # A tibble: 1 x 2
##   avg_delay flights
##       <dbl>   <int>
## 1      12.6  336776
```

- in this case, we got a single row because dplyr treats all rows of an ungrouped data frame as belonging to one group (or a supergroup?)

the `.by()` argument in `summarize()`

- new and experimental argument
- allows us to use the `summarize()` function without using `group_by()` first

```
flights |>
  summarize(
    delay = mean(dep_delay, na.rm = TRUE),
    n = n(),
    .by = month
  )
```

```
## # A tibble: 12 x 3
##    month delay     n
##    <int> <dbl> <int>
##  1     1 10.0  27004
##  2    10  6.24 28889
##  3    11  5.44 27268
##  4    12 16.6  28135
##  5     2 10.8  24951
##  6     3 13.2  28834
##  7     4 13.9  28330
##  8     5 13.0  28796
##  9     6 20.8  28243
## 10     7 21.7  29425
## 11     8 12.6  29327
## 12     9  6.72 27574
```

```
flights |>
  summarize(
    delay = mean(dep_delay, na.rm = TRUE),
    n = n(),
    .by = c(origin, dest)
  )
```

```
## # A tibble: 224 x 4
##    origin dest  delay     n
##    <chr>  <chr> <dbl> <int>
##  1 EWR    IAH   11.8   3973
##  2 LGA    IAH    9.06  2951
##  3 JFK    MIA    9.34  3314
##  4 JFK    BQN    6.67   599
##  5 LGA    ATL   11.4  10263
##  6 EWR    ORD   14.6   6100
##  7 EWR    FLL   13.5   3793
##  8 LGA    IAD   16.7   1803
##  9 JFK    MCO   10.6   5464
## 10 LGA    ORD   10.7   8857
## # i 214 more rows
```

Exercises using data from the Lahman package

```
batters <- Lahman::Batting |>
  group_by(playerID) |>
  summarize(
    performance = sum(H, na.rm = TRUE) / sum(AB, na.rm = TRUE),
    n = sum(AB, na.rm = TRUE)
  )

batters
```

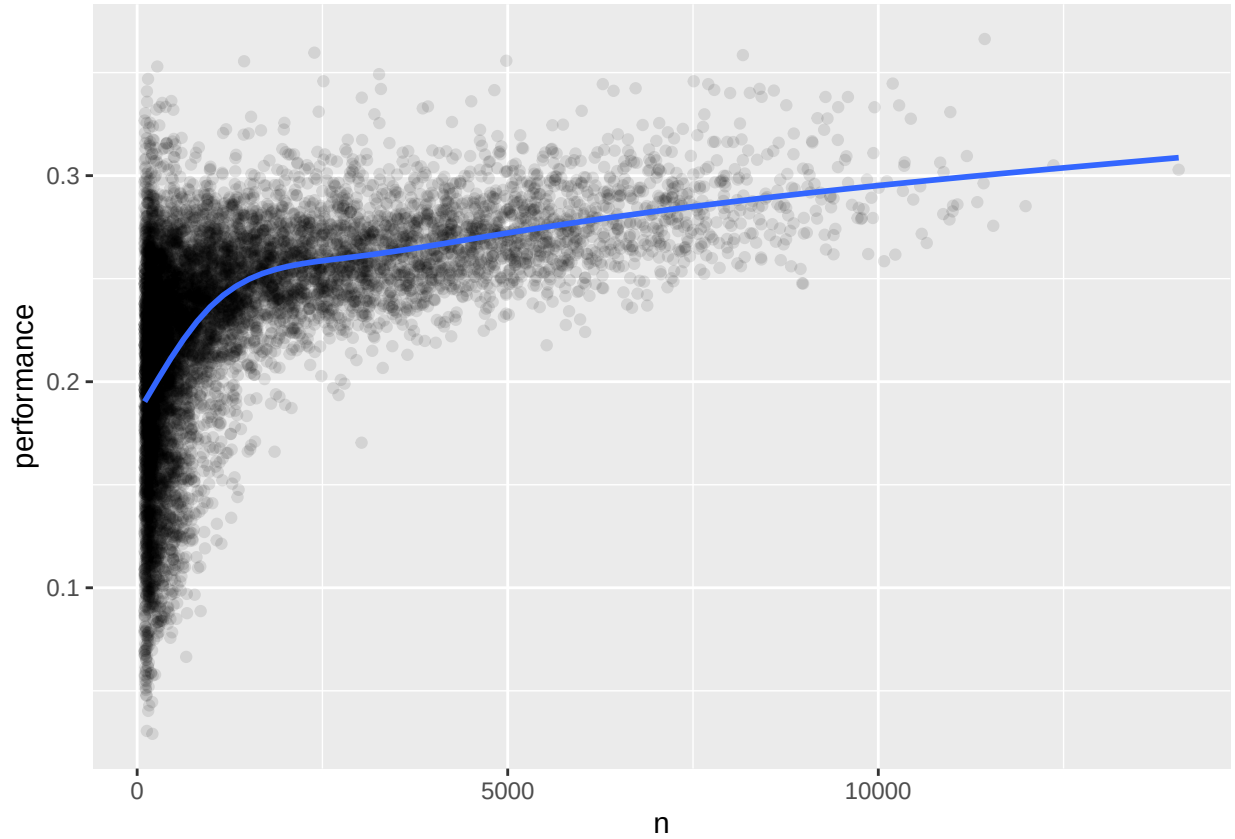
```
## # A tibble: 20,730 x 3
##    playerID  performance     n
```

```
##    <chr>           <dbl> <int>
##  1 aardsda01      0         4
##  2 aaronha01      0.305  12364
##  3 aaronto01      0.229    944
##  4 aasedo01       0          5
##  5 abadan01       0.0952    21
##  6 abadfe01       0.111      9
##  7 abadijo01      0.224     49
##  8 abbated01      0.254   3044
##  9 abbeybe01      0.169    225
## 10 abbeych01      0.281   1756
## # i 20,720 more rows
```

- plotting the skill of the batter (batting average, performance) against the number of opportunities to hit the ball (times at bat, `n`):

```
batters |>
  filter(n > 100) |>
  ggplot(aes(x = n, y = performance)) +
  geom_point(alpha = 0.10) +
  geom_smooth(se = FALSE)
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```



- from the plot, we can see that:

1. the variation in performance among players with fewer at-bats. This is commonly observed when computing summary statistics: the variation decreases as the sample size increases.
2. There is a positive correlation between skill (performance) and opportunities to hit the ball — of course teams want to give their best batters the most opportunities to hit the ball.

- this makes it tricky to rank the performance of batters.
- Naively getting the ranking based on batting averages will yield a different result:

```
batters |>
  arrange(desc(performance))
```

```
## # A tibble: 20,730 x 3
##    playerID  performance     n
##    <chr>           <dbl> <int>
##  1 abramge01           1     1
##  2 alberan01           1     1
##  3 banisje01           1     1
##  4 bartocl01           1     1
##  5 bassdo01            1     1
##  6 birasst01           1     2
##  7 bruneju01           1     1
##  8 burnscb01           1     1
##  9 cammaer01           1     1
## 10 campsh01            1     1
## # i 20,720 more rows
```

- this reminds me of school rankings based on board exam results. If we naively rank based on passing percentage alone, many schools who only had one board exam taker will land on top simply because they got a "perfect" passing rate (1 out of 1).