# Chapter 5

Data Transformation

Icaro Franco Hernandes

2022-09-26

Remember that to view a whole data set we can execute, for example, `view(nycflights13::flights)`. This is a tibble, and tibble $\neq$ table. Tibbles work better for the tidyverse. To check what kind of variable we are working with, we can use the following command:

```
typeof(nycflights13::flights$time_hour)
```

```
## [1] "double"
```

## Filtering

Selecting all flights from January first:

```
nycflights13::flights%>%
  dplyr::filter(month==1,day==1)->jan1
#remeber that dplyr does not change the original dataset (always try to be as pure as possible).
```

If we want to also print the new data set, just put between parenthesis:

```
(nycflights13::flights%>%
  dplyr::filter(month==1,day==1)->jan1)
```

```
## # A tibble: 842 x 19
##     year month   day dep_time sched_de~1 dep_d~2 arr_t~3 sched~4 arr_d~5 carrier
##    <int> <int> <int>    <int>      <int>   <dbl>   <int>   <int>   <dbl> <chr>
##  1  2013     1     1      517        515       2     830     819      11 UA
##  2  2013     1     1      533        529       4     850     830      20 UA
##  3  2013     1     1      542        540       2     923     850      33 AA
##  4  2013     1     1      544        545      -1    1004    1022     -18 B6
##  5  2013     1     1      554        600      -6     812     837     -25 DL
##  6  2013     1     1      554        558      -4     740     728      12 UA
##  7  2013     1     1      555        600      -5     913     854      19 B6
##  8  2013     1     1      557        600      -3     709     723     -14 EV
##  9  2013     1     1      557        600      -3     838     846      -8 B6
## 10  2013     1     1      558        600      -2     753     745       8 AA
## # ... with 832 more rows, 9 more variables: flight <int>, tailnum <chr>,
## #   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,
## #   minute <dbl>, time_hour <dttm>, and abbreviated variable names
## #   1: sched_dep_time, 2: dep_delay, 3: arr_time, 4: sched_arr_time,
## #   5: arr_delay
```

Boolean operators are: `&` for **and**, `|` for **or** and `!` for **is not**.

Inclusion operator: `%in%`. For example:

```
nycflights13::flights%>%
  dplyr::filter(month %in% c(11,12))
```

```
## # A tibble: 55,403 x 19
##     year month   day dep_time sched_de~1 dep_d~2 arr_t~3 sched~4 arr_d~5 carrier
##    <int> <int> <int>    <int>      <int>   <dbl>   <int>   <int>   <dbl> <chr>
##  1  2013    11     1        5       2359       6     352     345       7 B6
##  2  2013    11     1       35       2250     105     123    2356      87 B6
##  3  2013    11     1      455        500      -5     641     651     -10 US
##  4  2013    11     1      539        545      -6     856     827      29 UA
##  5  2013    11     1      542        545      -3     831     855     -24 AA
##  6  2013    11     1      549        600     -11     912     923     -11 UA
##  7  2013    11     1      550        600     -10     705     659       6 US
##  8  2013    11     1      554        600      -6     659     701      -2 US
##  9  2013    11     1      554        600      -6     826     827      -1 DL
## 10  2013    11     1      554        600      -6     749     751      -2 DL
## # ... with 55,393 more rows, 9 more variables: flight <int>, tailnum <chr>,
## #   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,
## #   minute <dbl>, time_hour <dttm>, and abbreviated variable names
## #   1: sched_dep_time, 2: dep_delay, 3: arr_time, 4: sched_arr_time,
## #   5: arr_delay
```

This will filter all flights that happened in november **or** december. The filter already excludes `NA` values.

## Exercises 5.2.4

### 1

Find all flights that

- Had an arrival delay of two or more hours

```
nycflights13::flights%>%
  dplyr::filter(arr_delay>=120)->f1
```

- Flew to Houston (`IAH` or `HOU`)

```
nycflights13::flights%>%
  dplyr::filter(dest %in% c("IAH", "HOU"))->f2
```

- Were operated by United, American, or Delta

```
nycflights13::flights%>%
  dplyr::filter(carrier %in% c("UA", "AA", "DL"))->f3
```

- Departed in Summer (July, August, and September)

```
nycflights13::flights%>%
  dplyr::filter(month %in% c(7, 8, 9))->fsummer
```

- Arrived more than two hours late, but did not leave late

```
nycflights13::flights%>%
  dplyr::filter(arr_delay>120,dep_time<=sched_dep_time)->f5
```

- Were delayed by at least an hour, but made up over 30 minutes in flight

```
nycflights13::flights%>%
  dplyr::filter(arr_delay>=60,air_time>30)->f6
```

- Departed between midnight and 6am (inclusive)

```
nycflights13::flights%>%
  dplyr::filter(hour %in% c(seq(0,6)))->f7

or
```

```
## function (e1, e2)  .Primitive("|")
```

```
nycflights13::flights%>%
  dplyr::filter(hour >= 0 & hour<= 6)->f71

# nycflights13::flights%>%
#   dplyr::filter(0 <= hour <= 6) -> this does not work!
```

**2**

Another useful `dplyr` filter helper is `between()`. What does it do? Can you use it to simplify the code needed to answer the precious questions?

According to the documentation, `between()` let us pick any values between to boundaries, and it is a shortcut for `x>= & x<=`. They would be useful in the cases where we had to filter for the summer months and the flights between midnight and 6 a.m.:

```
nycflights13::flights%>%
  dplyr::filter(between(month, 7, 9))->f8

nycflights13::flights%>%
  dplyr::filter(between(hour, 0, 6))->f9
```

**3**

How many flights have a missing `dep_time`? What other variables are missing? What might these rows represent?

```
nycflights13::flights%>%
  dplyr::filter(is.na(dep_time))%>%
  dplyr::summarise(n = dplyr::n())->na

na
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1  8255
```

Using the count operator from `dplyr` we can see that 8255 flights have missing values for the departure time. This means that theses flights were canceled. If we do not have the departure time, we also cannot check the airtime, the departure delay and the arrival delay. **Remember this count operator (within the summarise function) from dplyr**.

**4**

Why is `NA^0`not missing? Why is `NA|TRUE` not missing? Whys is `FALSE & NA` not missing? Can you figure out the general rule? (`NA*0` is a tricky counterexample!)

```
NA^0
```

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

```
NA|TRUE
```

```
## [1] TRUE
```

```
FALSE&NA
```

```
## [1] FALSE
```

Since we are working with boolean operators here, the general rule is that R avoids the NA values and does let them contaminate the operation. It is different from the case if we calculate the average of some values with an NA (in that case it does contaminate the average).

```
v1<-c(1,1, NA)
mean(v1)
```

```
## [1] NA
```

```
mean(v1, na.rm = T)
```

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

The command `na.rm=TRUE` discards the `NA` values from the calculation!

# Arranging

## Exercises 5.3.1

### 1

How could you use `arrange()` to sort all missing values to the start? (Hint: use `is.na()`)

```
v<-tibble::tibble(
  values = c(rnorm(3),NA)
)

v%>%
  dplyr::arrange(desc(is.na(v)))
```

```
## # A tibble: 4 x 1
##    values
##     <dbl>
## 1 NA
## 2 -0.0235
## 3  2.34
## 4 -1.12
```

### 2

Sort `flights` to find the most delayed flights. Find the flights that left earliest.

For the most delayed flights we just need to arrange in descending format:

```
nycflights13::flights%>%
  dplyr::arrange(desc(dep_delay))
```

```
## # A tibble: 336,776 x 19
##     year month   day dep_time sched_de~1 dep_d~2 arr_t~3 sched~4 arr_d~5 carrier
##    <int> <int> <int>    <int>      <int>   <dbl>   <int>   <int>   <dbl> <chr>
##  1  2013     1     9      641        900    1301    1242    1530    1272 HA
##  2  2013     6    15     1432       1935    1137    1607    2120    1127 MQ
```

```
##  3  2013     1    10     1121       1635    1126    1239    1810    1109 MQ
##  4  2013     9    20     1139       1845    1014    1457    2210    1007 AA
##  5  2013     7    22      845       1600    1005    1044    1815     989 MQ
##  6  2013     4    10     1100       1900     960    1342    2211     931 DL
##  7  2013     3    17     2321        810     911     135    1020     915 DL
##  8  2013     6    27      959       1900     899    1236    2226     850 DL
##  9  2013     7    22     2257        759     898     121    1026     895 DL
## 10  2013    12     5      756       1700     896    1058    2020     878 AA
## # ... with 336,766 more rows, 9 more variables: flight <int>, tailnum <chr>,
## #   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,
## #   minute <dbl>, time_hour <dttm>, and abbreviated variable names
## #   1: sched_dep_time, 2: dep_delay, 3: arr_time, 4: sched_arr_time,
## #   5: arr_delay
```

And for the earliest departures we just need to arrange them:

```
nycflights13::flights%>%
  dplyr::arrange(dep_delay)
```

```
## # A tibble: 336,776 x 19
##     year month   day dep_time sched_de~1 dep_d~2 arr_t~3 sched~4 arr_d~5 carrier
##    <int> <int> <int>    <int>      <int>   <dbl>   <int>   <int>   <dbl> <chr>
##  1  2013    12     7     2040       2123     -43      40    2352      48 B6
##  2  2013     2     3     2022       2055     -33    2240    2338     -58 DL
##  3  2013    11    10     1408       1440     -32    1549    1559     -10 EV
##  4  2013     1    11     1900       1930     -30    2233    2243     -10 DL
##  5  2013     1    29     1703       1730     -27    1947    1957     -10 F9
##  6  2013     8     9      729        755     -26    1002     955       7 MQ
##  7  2013    10    23     1907       1932     -25    2143    2143       0 EV
##  8  2013     3    30     2030       2055     -25    2213    2250     -37 MQ
##  9  2013     3     2     1431       1455     -24    1601    1631     -30 9E
## 10  2013     5     5      934        958     -24    1225    1309     -44 B6
## # ... with 336,766 more rows, 9 more variables: flight <int>, tailnum <chr>,
## #   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,
## #   minute <dbl>, time_hour <dttm>, and abbreviated variable names
## #   1: sched_dep_time, 2: dep_delay, 3: arr_time, 4: sched_arr_time,
## #   5: arr_delay
```

**3**

Sort `flights` to find the fastest (highest speed) flights

```
nycflights13::flights%>%
  dplyr::select(air_time,
                dplyr::everything())%>%
  dplyr::arrange(air_time)
```

```
## # A tibble: 336,776 x 19
##    air_time  year month   day dep_time sched_d~1 dep_d~2 arr_t~3 sched~4 arr_d~5
##       <dbl> <int> <int> <int>    <int>     <int>   <dbl>   <int>   <int>   <dbl>
##  1       20  2013     1    16     1355      1315      40    1442    1411      31
##  2       20  2013     4    13      537       527      10     622     628      -6
##  3       21  2013    12     6      922       851      31    1021     954      27
##  4       21  2013     2     3     2153      2129      24    2247    2224      23
##  5       21  2013     2     5     1303      1315     -12    1342    1411     -29
##  6       21  2013     2    12     2123      2130      -7    2211    2225     -14
```

```
##  7       21  2013     3     2     1450      1500     -10    1547    1608     -21
##  8       21  2013     3     8     2026      1935      51    2131    2056      35
##  9       21  2013     3    18     1456      1329      87    1533    1426      67
## 10       21  2013     3    19     2226      2145      41    2305    2246      19
## # ... with 336,766 more rows, 9 more variables: carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, distance <dbl>, hour <dbl>,
## #   minute <dbl>, time_hour <dttm>, and abbreviated variable names
## #   1: sched_dep_time, 2: dep_delay, 3: arr_time, 4: sched_arr_time,
## #   5: arr_delay
```

**4**

Which flights traveled the farthest? Which traveled the shortest?

The ones travelled the fartherst:

```
nycflights13::flights%>%
  dplyr::select(distance,
                dplyr::everything())%>%
  dplyr::arrange(desc(distance))
```

```
## # A tibble: 336,776 x 19
##    distance  year month   day dep_time sched_d~1 dep_d~2 arr_t~3 sched~4 arr_d~5
##       <dbl> <int> <int> <int>    <int>     <int>   <dbl>   <int>   <int>   <dbl>
##  1     4983  2013     1     1      857       900      -3    1516    1530     -14
##  2     4983  2013     1     2      909       900       9    1525    1530      -5
##  3     4983  2013     1     3      914       900      14    1504    1530     -26
##  4     4983  2013     1     4      900       900       0    1516    1530     -14
##  5     4983  2013     1     5      858       900      -2    1519    1530     -11
##  6     4983  2013     1     6     1019       900      79    1558    1530      28
##  7     4983  2013     1     7     1042       900     102    1620    1530      50
##  8     4983  2013     1     8      901       900       1    1504    1530     -26
##  9     4983  2013     1     9      641       900    1301    1242    1530    1272
## 10     4983  2013     1    10      859       900      -1    1449    1530     -41
## # ... with 336,766 more rows, 9 more variables: carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, hour <dbl>,
## #   minute <dbl>, time_hour <dttm>, and abbreviated variable names
## #   1: sched_dep_time, 2: dep_delay, 3: arr_time, 4: sched_arr_time,
## #   5: arr_delay
```

The ones that traveled the shortest:

```
nycflights13::flights%>%
  dplyr::select(distance,
                dplyr::everything())%>%
  dplyr::arrange(distance)
```

```
## # A tibble: 336,776 x 19
##    distance  year month   day dep_time sched_d~1 dep_d~2 arr_t~3 sched~4 arr_d~5
##       <dbl> <int> <int> <int>    <int>     <int>   <dbl>   <int>   <int>   <dbl>
##  1       17  2013     7    27       NA       106      NA      NA     245      NA
##  2       80  2013     1     3     2127      2129      -2    2222    2224      -2
##  3       80  2013     1     4     1240      1200      40    1333    1306      27
##  4       80  2013     1     4     1829      1615     134    1937    1721     136
##  5       80  2013     1     4     2128      2129      -1    2218    2224      -6
##  6       80  2013     1     5     1155      1200      -5    1241    1306     -25
##  7       80  2013     1     6     2125      2129      -4    2224    2224       0
```

```
##  8       80  2013     1     7     2124      2129      -5    2212    2224     -12
##  9       80  2013     1     8     2127      2130      -3    2304    2225      39
## 10       80  2013     1     9     2126      2129      -3    2217    2224      -7
## # ... with 336,766 more rows, 9 more variables: carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, hour <dbl>,
## #   minute <dbl>, time_hour <dttm>, and abbreviated variable names
## #   1: sched_dep_time, 2: dep_delay, 3: arr_time, 4: sched_arr_time,
## #   5: arr_delay
```