Data manipulation with dplyr

Programming for Statistical Science

Shawn Santo

Supplementary materials

Full video lecture available in Zoom Cloud Recordings

Additional resources

- `dplyr` cheat sheet
- dplyr vignette
- Chapter 5, R for Data Science

Getting started

```
library(tidyverse)
```

```
── Attaching packages ─────────────────── tidyverse 1.3.0 ──
✓ ggplot2 3.3.2     ✓ purrr   0.3.4
✓ tibble  3.0.3     ✓ dplyr   1.0.0
✓ tidyr   1.1.0     ✓ stringr 1.4.0
✓ readr   1.3.1     ✓ forcats 0.5.0
── Conflicts ────────────────────── tidyverse_conflicts() ──
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
```

Also, load `nycflights13`.

```
library(nycflights13)
```

Pipes

Pipes in R

Infix function `%>%` is a forward-pipe operator. It allows you to pipe an object forward into a function or call expression.

You can think about the following sequence of actions - find keys, unlock car, start car, drive to school, park.

Expressed as a set of nested functions in R pseudo code this would look like:

```
park(drive(start_car(unlock_car(find("keys"))), to = "campus"))
```

Writing it out using pipes give it a more natural (and easier to read) structure:

```
find("keys") %>%
  unlock_car() %>%
  start_car() %>%
  drive(to = "campus") %>%
  park()
```

Approaches

All of the following are fine, it comes down to personal preference.

Nested:

```
h(g(f(x), y = 1), z = 1)
```

Piped:

```
f(x) %>%
  g(y = 1) %>%
  h(z = 1)
```

Intermediate:

```
res <- f(x)
res <- g(res, y = 1)
res <- h(res, z = 1)
```

What about other arguments?

By default, the object on the left-hand side of `%>%` is placed as the value to the first argument in the function on the right-hand side of `%>%`.

To pass the value to other arguments a `.` is used. For example,

```
data.frame(a = 1:3, b = 3:1) %>% lm(a ~ b, data = .)
```

```
#>
#> Call:
#> lm(formula = a ~ b, data = .)
#>
#> Coefficients:
#> (Intercept)            b
#>           4           -1
```

```
data.frame(a = 1:3, b = 3:1) %>% .[[1]]
```

```
#> [1] 1 2 3
```

```
data.frame(a = 1:3, b = 3:1) %>% .[[length(.)]]
```

```
#> [1] 3 2 1
```

Data wrangling with `dplyr`

A grammar of data manipulation

Package `dplyr` is based on the concepts of functions as verbs that manipulate data frames.

Single data frame functions / verbs:

Function	Description
`filter()`	pick rows matching criteria
`slice()`	pick rows using indices
`select()`	pick columns by name
`pull()`	grab a column as a vector
`rename()`	rename specific columns
`arrange()`	reorder rows
`mutate()`	add new variables
`transmute()`	create new data frame with variables
`distinct()`	filter for unique rows
`sample_n()` / `sample_frac()`	randomly sample rows
`summarise()`	reduce variables to values

... many more.

`dplyr` rules

1. First argument is *always* a data frame
2. Subsequent arguments say what to do with that data frame
3. Almost always returns a data frame
4. Doesn't modify in place

Based on rules 1 and 3, it is natural to apply `%>%` in a sequence of `dplyr` functions for data wrangling purposes.

Data

We will demonstrate `dplyr`'s functionality using the `nycflights13` package.

```
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 <dttm>
```

`filter()` - March flights

```
flights %>%
  filter(month == 3)
```

```
#> # A tibble: 28,834 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     3     1        4           2159       125      318             56
#>  2  2013     3     1       50           2358        52      526            438
#>  3  2013     3     1      117           2245       152      223           2354
#>  4  2013     3     1      454            500        -6      633            648
#>  5  2013     3     1      505            515       -10      746            810
#>  6  2013     3     1      521            530        -9      813            827
#>  7  2013     3     1      537            540        -3      856            850
#>  8  2013     3     1      541            545        -4     1014           1023
#>  9  2013     3     1      549            600       -11      639            703
#> 10  2013     3     1      550            600       -10      747            801
#> # … with 28,824 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 <dttm>
```

`filter()` - flights in the first 7 days of March

```
flights %>%
  filter(month == 3, day <= 7)
```

```
#> # A tibble: 6,530 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     3     1        4           2159       125      318             56
#>  2  2013     3     1       50           2358        52      526            438
#>  3  2013     3     1      117           2245       152      223           2354
#>  4  2013     3     1      454            500        -6      633            648
#>  5  2013     3     1      505            515       -10      746            810
#>  6  2013     3     1      521            530        -9      813            827
#>  7  2013     3     1      537            540        -3      856            850
#>  8  2013     3     1      541            545        -4     1014           1023
#>  9  2013     3     1      549            600       -11      639            703
#> 10  2013     3     1      550            600       -10      747            801
#> # … with 6,520 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 <dttm>
```

`filter()` - flights to LAX *or* RDU in March

```
flights %>%
  filter(dest == "LAX" | dest == "RDU", month == 3)
```

```
#> # A tibble: 1,935 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     3     1      607            610        -3      832            925
#>  2  2013     3     1      608            615        -7      737            750
#>  3  2013     3     1      623            630        -7      753            810
#>  4  2013     3     1      629            632        -3      844            952
#>  5  2013     3     1      657            700        -3      953           1034
#>  6  2013     3     1      714            715        -1      939           1037
#>  7  2013     3     1      716            710         6      958           1035
#>  8  2013     3     1      727            730        -3     1007           1100
#>  9  2013     3     1      803            810        -7      923            955
#> 10  2013     3     1      823            824        -1      954           1014
#> # … with 1,925 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 <dttm>
```

`slice()` - first 10 flights

```
flights %>%
  slice(1:10)
```

```
#> # A tibble: 10 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 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>
```

`slice()` - last 5 flights

```
flights %>%
  slice((n() - 4):n())
```

```
#> # 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     9    30       NA           1455        NA       NA           1634
#> 2  2013     9    30       NA           2200        NA       NA           2312
#> 3  2013     9    30       NA           1210        NA       NA           1330
#> 4  2013     9    30       NA           1159        NA       NA           1344
#> 5  2013     9    30       NA            840        NA       NA           1020
#> # … 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 <dttm>
```

`select()` - specific variables

```
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
#> # … with 336,766 more rows
```

`select()` - exclude variables

```
flights %>%
  select(-year, -month, -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
#> # … with 336,766 more rows, and 9 more variables: flight <int>, tailnum <chr>,
#> #   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,
#> #   minute <dbl>, time_hour <dttm>
```

`select()` - ranges

```
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
#> # … with 336,766 more rows
```

`select()` - exclude ranges

```
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
#> # … with 336,766 more rows, and 9 more variables: flight <int>, tailnum <chr>,
#> #   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,
#> #   minute <dbl>, time_hour <dttm>
```

`select()` - matching

```
flights %>%
  select(contains("dep"), contains("arr"))
```

```
#> # A tibble: 336,776 x 7
#>    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
#> # … with 336,766 more rows
```

```
flights %>%
  select(starts_with("dep"), starts_with("arr"))
```

```
#> # 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
```

Other helpers: `ends_with()`, `matches()`, `num_range()`, `one_of()`, `everything()`, `last_col()`

select_if()

```
flights %>%
  select_if(function(x) !is.numeric(x))
```

```
#> # A tibble: 336,776 x 5
#>    carrier tailnum origin dest  time_hour
#>    <chr>   <chr>   <chr>  <chr> <dttm>
#>  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
```

Alternatively,

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

`pull()` - grab a vector

```
names(flights)
```

```
#>  [1] "year"           "month"          "day"            "dep_time"
#>  [5] "sched_dep_time" "dep_delay"      "arr_time"       "sched_arr_time"
#>  [9] "arr_delay"      "carrier"        "flight"         "tailnum"
#> [13] "origin"         "dest"           "air_time"       "distance"
#> [17] "hour"           "minute"         "time_hour"
```

```
flights %>% pull("year") %>% head()
```

```
#> [1] 2013 2013 2013 2013 2013 2013
```

```
flights %>% pull(1) %>% head
```

```
#> [1] 2013 2013 2013 2013 2013 2013
```

```
flights %>% pull(-1) %>% .[1:4]
```

```
#> [1] "2013-01-01 05:00:00 EST" "2013-01-01 05:00:00 EST"
#> [3] "2013-01-01 05:00:00 EST" "2013-01-01 05:00:00 EST"
```

`arrange()` - sort data

```
flights %>%
  filter(month == 3, day == 2) %>%
  arrange(origin, dest)
```

```
#> # A tibble: 765 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     3     2     1336           1329         7     1426           1432
#>  2  2013     3     2      628            629        -1      837            849
#>  3  2013     3     2      637            640        -3      903            915
#>  4  2013     3     2      743            745        -2      945           1010
#>  5  2013     3     2      857            900        -3     1117           1126
#>  6  2013     3     2     1027           1030        -3     1234           1247
#>  7  2013     3     2     1134           1145       -11     1332           1359
#>  8  2013     3     2     1412           1415        -3     1636           1630
#>  9  2013     3     2     1633           1636        -3     1848           1908
#> 10  2013     3     2     1655           1700        -5     1857           1924
#> # … with 755 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 <dttm>
```

`arrange()` & `desc()`

By default, sorting is done in ascending order. To change that, use `desc()`.

```
flights %>%
  filter(month == 3, day == 2) %>%
  arrange(desc(origin), dest) %>%
  select(origin, dest, tailnum)
```

```
#> # A tibble: 765 x 3
#>    origin dest  tailnum
#>    <chr>  <chr> <chr>
#>  1 LGA    ATL   N928AT
#>  2 LGA    ATL   N623DL
#>  3 LGA    ATL   N680DA
#>  4 LGA    ATL   N996AT
#>  5 LGA    ATL   N510MQ
#>  6 LGA    ATL   N663DN
#>  7 LGA    ATL   N942DL
#>  8 LGA    ATL   N511MQ
#>  9 LGA    ATL   N910DE
#> 10 LGA    ATL   N902DE
#> # … with 755 more rows
```

`mutate()` - modify columns

```
flights %>%
  select(year:day) %>%
  mutate(date = paste(year, month, day, sep = "/"))
```

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

`transmute()` - create tibble from existing columns

```
flights %>%
  transmute(date = paste(year, month, day, sep = "/"))
```

```
#> # A tibble: 336,776 x 1
#>    date
#>    <chr>
#>  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
#> # … with 336,766 more rows
```

`distinct()` - find unique rows

```
flights %>%
  select(origin, dest) %>%
  distinct() %>%
  arrange(origin, dest)
```

```
#> # A tibble: 224 x 2
#>    origin dest
#>    <chr>  <chr>
#>  1 EWR    ALB
#>  2 EWR    ANC
#>  3 EWR    ATL
#>  4 EWR    AUS
#>  5 EWR    AVL
#>  6 EWR    BDL
#>  7 EWR    BNA
#>  8 EWR    BOS
#>  9 EWR    BQN
#> 10 EWR    BTV
#> # … with 214 more rows
```

Sampling rows

`sample_n()`

```
flights %>%
  select(year, origin) %>%
  sample_n(10)
```

```
#> # A tibble: 10 x 2
#>     year origin
#>    <int> <chr>
#>  1  2013 EWR
#>  2  2013 LGA
#>  3  2013 EWR
#>  4  2013 LGA
#>  5  2013 LGA
#>  6  2013 EWR
#>  7  2013 LGA
#>  8  2013 EWR
#>  9  2013 EWR
#> 10  2013 LGA
```

`sample_frac()`

```
flights %>%
  select(year, origin) %>%
  sample_frac(.00001)
```

```
#> # A tibble: 3 x 2
#>    year origin
#>   <int> <chr>
#> 1  2013 JFK
#> 2  2013 JFK
#> 3  2013 JFK
```

Exercises

Data: Wake county parcels

Parcel boundaries with address and revenue-related information for properties in Wake County, NC. http://data-wake.opendata.arcgis.com/datasets/parcels

```
wake <- read_csv("https://www2.stat.duke.edu/~sms185/data/econ/parcels.csv")
```

```
wake <-
  janitor::clean_names(wake)

glimpse(wake)
```

```
#> Rows: 378,020
#> Columns: 59
#> $ objectid              <dbl> 31257151, 31257152, 31257153, 31257154, 3125715…
#> $ pin_num               <chr> "1701518493", "0745330365", "0753213162", "1743…
#> $ calc_area             <dbl> 0.59611048, 0.06596296, 0.17740871, 0.24685172,…
#> $ reid                  <chr> "0004217", "0240874", "0337154", "0340605", "03…
#> $ map_name              <chr> "1701 19", "0745 03", "0753 17", "1743 02", "18…
#> $ owner                 <chr> "HAMILTON, HUBERT EARL HAMILTON, PATRICIA Y", "…
#> $ addr1                 <chr> "500 POPLAR DR", "104 MADISON GROVE PL", "3038 …
#> $ addr2                 <chr> "RALEIGH NC 27603-4330", "CARY NC 27519-8159", …
#> $ addr3                 <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,…
#> $ deed_book             <chr> "002618", "015772", "012192", "016213", "014787…
#> $ deed_page             <chr> "00683", "00810", "00556", "00085", "01020", "0…
#> $ deed_date             <dttm> 1978-01-01, 2014-09-04, 2006-09-29, 2015-11-16…
#> $ deed_acres            <dbl> 0.60, 0.07, 0.18, 0.25, 0.26, 0.16, 1.15, 1.01,…
#> $ bldg_val              <dbl> 0, 190388, 319131, 216470, 390623, 204414, 2476…
#> $ land_val              <dbl> 27500, 102200, 90000, 40000, 85000, 40000, 5600…
#> $ total_value_assd      <dbl> 27500, 292588, 409131, 256470, 475623, 244414, …
#> $ billclass             <dbl> 2, 2, 2, 2, 2, 2, 2, 1, 2, 2, 3, 2, 2, 2, 2, 2,…
#> $ billing_class_decode  <chr> "Individual", "Individual", "Individual", "Indi…
#> $ propdesc              <chr> "LO1 ECHO HTS SE7", "LO99 CARPENTER VILLAGE BLL…
#> $ heatedarea            <dbl> NA, 1821, 3460, 2372, 3512, 2275, 2613, 1064, 3…
#> $ stname                <chr> "HICKORY", "MADISON GROVE", "KILARNEY RIDGE", "…
#> $ stype                 <chr> "LN", "PL", "LOOP", "DR", "DR", "DR", "CT", "RD…
#> $ stpre                 <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,…
#> $ stsuf                 <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,…
#> $ stnum                 <dbl> 5626, 104, 3038, 4608, 1328, 4014, 1501, 8241, …
#> $ stmisc                <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,…
#> $ site_address          <chr> "5626 HICKORY LN", "104 MADISON GROVE PL", "303…
#> $ full_street_name      <chr> "HICKORY LN", "MADISON GROVE PL", "KILARNEY RID…
#> $ city                  <chr> NA, "CAR", "CAR", "KNI", "WAK", "KNI", NA, NA, …
#> $ city_decode           <chr> NA, "CARY", "CARY", "KNIGHTDALE", "WAKE FOREST"…
#> $ planning_jurisdiction <chr> "GA", "CA", "CA", "KN", "WF", "KN", "WC", "WC",…
#> $ township              <chr> "16", "05", "04", "17", "19", "17", "15", "15",…
#> $ township_decode       <chr> "St. Mary's", "CEDAR FORK", "CARY", "St. Matthe…
#> $ firedist              <dbl> 23, NA, NA, NA, NA, NA, 23, 23, 23, 23, 23, 23,…
```

Tasks

1. Which city has the fewest land parcels in the dataset? *Hint*: `count()`.

2. Create a tibble that shows the year a parcel was built and the total value, where all parcels are located in Apex and are more than one acre in area. Sort the result in ascending order by year built.

3. Choose a subset of five variables and 10 random rows from `wake` and save it as an object named `wake_mini`. Experiment renaming variables with `select()` and `rename()` on `wake_mini`. What is the difference between the two functions?

summarise()

```
flights %>%
  summarize(n(), min(dep_delay), max(dep_delay))
```

```
#> # A tibble: 1 x 3
#>    `n()` `min(dep_delay)` `max(dep_delay)`
#>    <int>            <dbl>            <dbl>
#> 1 336776               NA               NA
```

```
flights %>%
  summarize(
    n = n(),
    min_dep_delay = min(dep_delay, na.rm = TRUE),
    max_dep_delay = max(dep_delay, na.rm = TRUE)
  )
```

```
#> # A tibble: 1 x 3
#>        n min_dep_delay max_dep_delay
#>    <int>         <dbl>         <dbl>
#> 1 336776           -43          1301
```

Useful `summarise()` functions

- Center: `mean()`, `median()`
- Spread: `sd()`, `IQR()`, `mad()`
- Range: `min()`, `max()`, `quantile()`
- Position: `first()`, `last()`, `nth()`
- Count: `n()`, `n_distinct()`
- Logical: `any()`, `all()`

group_by()

```
flights %>%
  group_by(origin)
```

```
# A tibble: 336,776 x 19
# Groups:   origin [3]
    year month   day dep_time sched_dep_time dep_delay arr_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>, 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>
```

`group_by()` **then** `summarise()`

```
flights %>%
  group_by(origin) %>%
  summarize(
    n = n(),
    min_dep_delay = min(dep_delay, na.rm = TRUE),
    max_dep_delay = max(dep_delay, na.rm = TRUE)
  )
```

```
#> # A tibble: 3 x 4
#>   origin      n min_dep_delay max_dep_delay
#>   <chr>   <int>         <dbl>         <dbl>
#> 1 EWR    120835           -25          1126
#> 2 JFK    111279           -43          1301
#> 3 LGA    104662           -33           911
```

```r
flights %>%
  group_by(origin, carrier) %>%
  summarize(
    n             = n(),
    min_dep_delay = min(dep_delay, na.rm = TRUE),
    max_dep_delay = max(dep_delay, na.rm = TRUE),
    .groups = "drop"
  ) %>%
  filter(n > 10000)
```

```
#> # A tibble: 10 x 5
#>    origin carrier     n min_dep_delay max_dep_delay
#>    <chr>  <chr>   <int>         <dbl>         <dbl>
#>  1 EWR    EV      43939           -25           548
#>  2 EWR    UA      46087           -18           424
#>  3 JFK    9E      14651           -24           747
#>  4 JFK    AA      13783           -15          1014
#>  5 JFK    B6      42076           -43           453
#>  6 JFK    DL      20701           -18           960
#>  7 LGA    AA      15459           -24           803
#>  8 LGA    DL      23067           -33           911
#>  9 LGA    MQ      16928           -26           366
#> 10 LGA    US      13136           -18           500
```

mutate() with group_by()

```
flights %>%
  group_by(origin) %>%
  mutate(n = n()) %>%
  select(origin, n)
```

```
#> # A tibble: 336,776 x 2
#> # Groups:   origin [3]
#>    origin      n
#>    <chr>   <int>
#>  1 EWR    120835
#>  2 LGA    104662
#>  3 JFK    111279
#>  4 JFK    111279
#>  5 LGA    104662
#>  6 EWR    120835
#>  7 EWR    120835
#>  8 LGA    104662
#>  9 JFK    111279
#> 10 LGA    104662
#> # … with 336,766 more rows
```

Example

```
flights %>%
  group_by(origin, month) %>%
  summarise(n = n()) %>%
  slice(1)
```

```
#> # A tibble: 3 x 3
#> # Groups:   origin [3]
#>   origin month     n
#>   <chr>  <int> <int>
#> 1 EWR        1  9893
#> 2 JFK        1  9161
#> 3 LGA        1  7950
```

Why do I have a tibble with three rows?

You may need to `ungroup()`

```
flights %>%
  group_by(origin, month) %>%
  summarise(n = n()) %>%
  ungroup() %>%
  slice(1)
```

```
#> # A tibble: 1 x 3
#>   origin month     n
#>   <chr>  <int> <int>
#> 1 EWR        1  9893
```

Or set the `.groups` argument in `summarise()` to `"drop"`. This is a new feature is `dplyr` version `1.0.0`.

`case_when()` - multi-case `if_else()`

Suppose we want to break the parcel size into three categories: small, medium, large.

```
wake %>%
  mutate(lot_size = case_when(
    deed_acres < .5   ~ "small",
    deed_acres < 1.5  ~ "medium",
    deed_acres >= 1.5 ~ "large"
  )) %>%
  select(deed_acres, lot_size)
```

```
#> # A tibble: 378,020 x 2
#>    deed_acres lot_size
#>         <dbl> <chr>
#>  1       0.6  medium
#>  2       0.07 small
#>  3       0.18 small
#>  4       0.25 small
#>  5       0.26 small
#>  6       0.16 small
#>  7       1.15 medium
#>  8       1.01 medium
#>  9       3.61 large
#> 10       1    medium
#> # … with 378,010 more rows
```

Exercises

Tasks

Continue to use `wake` for the following tasks.

1. Compute the mean area for each design style.
2. Compute the median sale price for each year. *Hint:* `lubridate::year()`
3. Which city with at least 1,000 parcels classified as a "Townhouse" had the highest proportion of parcels as "Townhouse"?

References

1. A Grammar of Data Manipulation. (2020). https://dplyr.tidyverse.org/index.html

2. Parcels. (2020). Data-wake.opendata.arcgis.com. http://data-wake.opendata.arcgis.com/datasets/parcels