Lab 05: Fonts & Tables

CS631

Alison Hill

Contents

Goals for Lab 05 **1**

TL;DR **2**

`knitr::kable` **2**
- `kable` all tables everywhere . . . 2
- `kable` table in a chunk . . . 2
- Styled `kable` tables in a chunk . . . 9
- `kable` + `kableExtra` + `formattable` . . . 17
- `tibble` + `kable` + `kableExtra` . . . 19

Markdown Tables **21**
- Simple table . . . 21
- Multi-line tables . . . 21
- Grid tables . . . 22
- Pipe tables . . . 23

Making tables in R **23**
- `dplyr` . . . 23
- `tidyr` . . . 31
- `broom` . . . 33

Specialized Packages **34**
- `huxtable` . . . 34
- `tableone` . . . 34
- The `DT` package . . . 37
- `xtable` (best for html) . . . 37
- `pixiedust` (best for PDF) . . . 39

Finally, fonts! **39**

Goals for Lab 05

```
mazes <- read_csv("http://bit.ly/mazes-gist") %>%
  clean_names()  #janitor package
```

TL;DR

The workhorse for making tables in R Markdown documents is the `knitr` package's `kable` function. This function is really versatile, but also free of fancy formatting options, for better or worse.

`knitr::kable`

`kable` all tables everywhere

Update the YAML of your document. For HTML:

```
---
title: "My Awesome Data Vis Lab"
output:
  html_document:
    df_print: kable
---
```

You can also define the html format in the global options.

```
# If you don't define format here, you'll need put `format = "html"` in every kable function.
options(knitr.table.format = "html")
# You may also wish to set this option
options(scipen = 1, digits = 2)
```

`kable` table in a chunk

For HTML:

```
head(mazes) %>%
  kable(format = "html")
```

study_id

ca

viq

dx

activity

content

filler

rep

rev

fs

cued

not_cued

CSLU-001

5.6667

124

TD

Conversation

24

31

2

5

17

36

50

CSLU-001

5.6667

124

TD

Picture Description

1

2

0

0

1

2

3

CSLU-001

5.6667

124

TD

Play

21

6

3

8

10

6

27

CSLU-001

5.6667

124

TD

Wordless Picture Book

8

2

0

4

4

2

10

CSLU-002

6.5000

124

TD

Conversation

3

10

3

0

0

10

13

CSLU-002

6.5000

124

TD

Picture Description

5

3

2

1

2

3

8

```
head(mazes) %>%
  kable(format = "html", digits = 2, caption = "A table produced by kable.")
```

A table produced by kable.

study_id

ca

viq

dx

activity

content

filler

rep

rev

fs

cued

not_cued

CSLU-001

5.67

124

TD

Conversation

24

31

2

5

17

36

50

CSLU-001

5.67

124

TD

Picture Description

1

2

0

0

1

2

3

CSLU-001

5.67

124

TD

Play

21

6

3

8

10

6

27

CSLU-001

5.67

124

TD

Wordless Picture Book

8

2

0

4

4

2

10

CSLU-002

6.50

124

TD

Conversation

3

10

3

0

0

10

13

CSLU-002

6.50

124

TD

Picture Description

5

3

2

1

2

3

8

```
my_maze_names <- c("Participant", "Age", "Verbal\nIQ", "Group", "Activity", "Content\nMaze", "Filler\nMa
head(mazes) %>%
  kable(format = "html", digits = 2, caption = "A table produced by kable.",
        col.names = my_maze_names)
```

A table produced by kable.

Participant

Age

Verbal IQ

Group

Activity

Content Maze

Filler Maze

Repetition

Revision

False Start

Cued

Not Cued

CSLU-001

5.67

124

TD

Conversation

24

31

2

5

17

36

50

CSLU-001

5.67

124

TD

Picture Description

1

2

0

0

1

2

3

CSLU-001

5.67

124

TD

Play

21

6

3

8

10

6

27

CSLU-001

5.67

124

TD

Wordless Picture Book

8

2

0

4

4

2

10

CSLU-002

6.50

124

TD

Conversation

3

10

3

0

0

10

13

CSLU-002

6.50

124

TD

Picture Description

5

3

2

1

2

3

8

Styled `kable` tables in a chunk

Solution: apply some Bootstrap CSS styling using the `kableExtra` package.

```
head(mazes) %>%
  kable(format = "html", digits = 2, caption = "A styled kable table.",
        col.names = my_maze_names) %>%
  kable_styling()
```

A styled kable table.

Participant

Age

Verbal IQ

Group

Activity

Content Maze

Filler Maze

Repetition

Revision

False Start

Cued

Not Cued

CSLU-001

5.67

124

TD

Conversation

24

31

2

5

17

36

50

CSLU-001

5.67

124

TD

Picture Description

1

2

0

0

1

2

3

CSLU-001

5.67

124

TD

Play

21

6

3

8

10

6

27

CSLU-001

5.67

124

TD

Wordless Picture Book

8

2

0

4

4

2

10

CSLU-002

6.50

124

TD

Conversation

3

10

3

0

0

10

13

CSLU-002

6.50

124

TD

Picture Description

5

3

2

1

2

3

8

Lots of printing options: https://haozhu233.github.io/kableExtra/awesome_table_in_html.html

```
head(mazes) %>%
  kable(format = "html", digits = 2, caption = "A non-full width zebra kable table.") %>%
  kable_styling(bootstrap_options = "striped", full_width = F)
```

A non-full width zebra kable table.

study_id

ca

viq

dx

activity

content

filler

rep

rev

fs

cued

not_cued

CSLU-001

5.67

124

TD

Conversation

24

31

2

5

17

36

50

CSLU-001

5.67

124

TD

Picture Description

1

2

0

0

1

2

3

CSLU-001

5.67

124

TD

Play

21

6

3

8

10

6

27

CSLU-001

5.67

124

TD

Wordless Picture Book

8

2

0

4

4

2

10

CSLU-002

6.50

124

TD

Conversation

3

10

3

0

0

10

13

CSLU-002

6.50

124

TD

Picture Description

5

3

2

1

2

3

8

```
head(mazes) %>%
  kable(format = "html", digits = 2, caption = "Over here!") %>%
  kable_styling(bootstrap_options = "striped", full_width = F, position = "left")
```

Over here!

study_id

ca

viq

dx

activity

content

filler

rep

rev

fs

cued

not_cued

CSLU-001

5.67

124

TD

Conversation

24

31

2

5

17

36

50

CSLU-001

5.67

124

TD

Picture Description

1

2

0

0

1

2

3

CSLU-001

5.67

124

TD

Play

21

6

3

8

10

6

27

CSLU-001

5.67

124

TD

Wordless Picture Book

8

2

0

4

4

2

10

CSLU-002

6.50

124

TD

Conversation

3

10

3

0

0

10

13

CSLU-002

6.50

124

TD

Picture Description

5

3

2

1

2

3

8

kable + kableExtra + formattable

`color_tile` and `color_bar` are neat extras if used wisely!

http://haozhu233.github.io/kableExtra/use_kableExtra_with_formattable.html

```
library(formattable)
head(mazes) %>%
  mutate(ca = color_tile("transparent", "lightpink")(ca),
         viq = color_bar("lightseagreen")(viq)) %>%
  kable("html", escape = F, caption = 'This table is colored.') %>%
  kable_styling(position = "center") %>%
  column_spec(4, width = "3cm")
```

This table is colored.

study_id

ca

viq

dx

activity

content

filler

rep

rev

fs

cued

not_cued

CSLU-001

5.6667

124

TD

Conversation

24

31

2

5

17

36

50

CSLU-001

5.6667

124

TD

Picture Description

1

2

0

0

1

2

3

CSLU-001

5.6667

124

TD

Play

21

6

3

8

10

6

27

CSLU-001

5.6667

124

TD

Wordless Picture Book

8

2

0

4

4

2

10

CSLU-002

6.5000

124

TD

Conversation

3

10

3

0

0

10

13

CSLU-002

6.5000

124

TD

Picture Description

5

3

2

1

2

3

8

`tibble` + `kable` + `kableExtra`

You can also use any of these tools with plain text tables using the `tibble` package to create a table. Two main functions:

- `tribble`: enter tibble by rows
- `tibble`: enter tibble by columns

For example, I used `tribble` to make this table in our slide decks:

```
math_table <- tibble::tribble(
  ~Operator, ~Description, ~Usage,
  "\\+", "addition", "x + y",
  "\\-", "subtraction", "x - y",
```

```
  "\\*", "multiplication", "x * y",
  "/", "division", "x / y",
  "^", "raised to the power of", "x ^ y",
  "abs", "absolute value", "abs(x)",
  "%/%", "integer division", "x %/% y",
  "%%", "remainder after division", "x %% y"
)
```

Then I used this chunk to print it:

```
```{r, results = 'asis'}
knitr::kable(math_table, format = "html", caption = "Helpful mutate functions") %>%
  kable_styling(bootstrap_options = "striped", full_width = F, position = "left")
```
```

```
knitr::kable(math_table, format = "html", caption = "Helpful mutate functions") %>%
  kable_styling(bootstrap_options = "striped", full_width = F, position = "left")
```

Helpful mutate functions

Operator

Description

Usage

\+

addition

x + y

\-

subtraction

x - y

*

multiplication

x * y

/

division

x / y

^

raised to the power of

x ^ y

abs

absolute value

abs(x)

%/%

integer division

x %/% y

%%

remainder after division

x %% y

Markdown Tables

Sometimes you may just want to type in a table in Markdown and ignore R. Four kinds of tables may be used. The first three kinds presuppose the use of a fixed-width font, such as Courier. The fourth kind can be used with proportionally spaced fonts, as it does not require lining up columns. All of the below will render when typed *outside* of an R code chunk since these are based on `pandoc` being used to render your markdown document. Note that these should all work whether you are knitting to either html or PDF.

Simple table

This code for a simple table:

```
  Right     Left     Center     Default
-------     ------ ----------   -------
     12     12        12            12
    123     123       123          123
      1     1          1             1

Table:  Demonstration of simple table syntax.
```

Produces this simple table:

Table 1: Demonstration of simple table syntax.

Right	Left	Center	Default
12	12	12	12
123	123	123	123
1	1	1	1

The headers and table rows must each fit on one line. Column alignments are determined by the position of the header text relative to the dashed line below it:3

- If the dashed line is flush with the header text on the right side but extends beyond it on the left, the column is right-aligned.
- If the dashed line is flush with the header text on the left side but extends beyond it on the right, the column is left-aligned.
- If the dashed line extends beyond the header text on both sides, the column is centered.
- If the dashed line is flush with the header text on both sides, the default alignment is used (in most cases, this will be left).
- The table must end with a blank line, or a line of dashes followed by a blank line.

The column headers may be omitted, provided a dashed line is used to end the table.

Multi-line tables

This code for a multi-line table:

```
-------------------------------------------------------------
 Centered   Default           Right Left
  Header    Aligned         Aligned Aligned
```

```
---------- ------- --------------- -------------------------
   First    row                12.0 Example of a row that
                                    spans multiple lines.

  Second    row                 5.0 Here's another one. Note
                                    the blank line between
                                    rows.
-------------------------------------------------------------

Table: Here's the caption. It, too, may span
multiple lines.
```

Produces this multi-line table:

Table 2: Here's the caption. It, too, may span multiple lines.

Centered Header	Default Aligned	Right Aligned	Left Aligned
First	row	12.0	Example of a row that spans multiple lines.
Second	row	5.0	Here's another one. Note the blank line between rows.

Grid tables

This code for a grid table:

```
: Sample grid table.

+---------------+---------------+--------------------+
| Fruit         | Price         | Advantages         |
+===============+===============+====================+
| Bananas       | $1.34         | - built-in wrapper |
|               |               | - bright color     |
+---------------+---------------+--------------------+
| Oranges       | $2.10         | - cures scurvy     |
|               |               | - tasty            |
+---------------+---------------+--------------------+
```

Produces this grid table:

Table 3: Sample grid table.

Fruit	Price	Advantages
Bananas	$1.34	• built-in wrapper • bright color
Oranges	$2.10	• cures scurvy • tasty

Alignments are not supported, nor are cells that span multiple columns or rows.

Pipe tables

This code for a pipe table:

```
| Right | Left | Default | Center |
|------:|:-----|---------|:------:|
|   12  |  12  |    12   |    12  |
|  123  |  123 |   123   |   123  |
|    1  |    1 |     1   |     1  |

  : Demonstration of pipe table syntax.
```

Produces this pipe table:

Table 4: Demonstration of pipe table syntax.

Right	Left	Default	Center
12	12	12	12
123	123	123	123
1	1	1	1

Making tables in R

If you want to make tables that include R output (like output from functions like means, variances, or output from models), there are two steps:

1. Get the numbers you need in tabular format; then
2. Render that information in an aesthetically-pleasing way.

This section covers (1). But, although there are some nice options for (2) within R Markdown via various packages, I am not dogmatic about doing *everything* in R Markdown, especially things like (2).

`dplyr`

We'll use the `pnwflights14` package to practice our `dplyr` skills. We need to download the package from github using `devtools`.

```
# once per machine
install.packages("devtools")
devtools::install_github("ismayc/pnwflights14")
```

Now, we need to load the `flights` dataset from the `pnwflights14` package.

```
# once per work session
data("flights", package = "pnwflights14")
```

`dplyr::select`

Use select to specify which columns in a dataframe you'd like to keep **by name**. Heretofore, this was not possible in base R! In base R, this can only be achieved using numeric variable positions. But most of the time, you keep track of your variables by name (like `carrier`) rather than position (the 8th column).

```
# keep these 2 cols
mini_flights <- flights %>%
  select(carrier, flight)
glimpse(mini_flights)
```

```
Observations: 162,049
Variables: 2
$ carrier <chr> "AS", "US", "UA", "US", "AS", "DL", "UA", "UA", "UA", "UA",...
$ flight  <int> 145, 1830, 1609, 466, 121, 1823, 1481, 229, 1576, 478, 1569...
```

```
# keep first five cols
first_five <- flights %>%
  select(year, month, day, dep_time, dep_delay)
glimpse(first_five)
```

```
Observations: 162,049
Variables: 5
$ year      <int> 2014, 2014, 2014, 2014, 2014, 2014, 2014, 2014, 2014, 201...
$ 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> 1, 4, 8, 28, 34, 37, 346, 526, 527, 536, 541, 549, 550, 5...
$ dep_delay <dbl> 96, -6, 13, -2, 44, 82, 227, -4, 7, 1, 1, 24, 0, -3, -3, ...
```

```
# alternatively, specify range
first_five <- flights %>%
  select(year:dep_delay)
glimpse(first_five)
```

```
Observations: 162,049
Variables: 5
$ year      <int> 2014, 2014, 2014, 2014, 2014, 2014, 2014, 2014, 2014, 201...
$ 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> 1, 4, 8, 28, 34, 37, 346, 526, 527, 536, 541, 549, 550, 5...
$ dep_delay <dbl> 96, -6, 13, -2, 44, 82, 227, -4, 7, 1, 1, 24, 0, -3, -3, ...
```

We can also choose the columns we want by negation, that is, you can specify which columns to drop instead of keep. This way, all variables **not** listed are kept.

```
# we can also use negation
all_but_year <- flights %>%
  select(-year)
glimpse(all_but_year)
```

```
Observations: 162,049
Variables: 15
$ 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> 1, 4, 8, 28, 34, 37, 346, 526, 527, 536, 541, 549, 550, 5...
$ dep_delay <dbl> 96, -6, 13, -2, 44, 82, 227, -4, 7, 1, 1, 24, 0, -3, -3, ...
$ arr_time  <int> 235, 738, 548, 800, 325, 747, 936, 1148, 917, 1334, 911, ...
$ arr_delay <dbl> 70, -23, -4, -23, 43, 88, 219, 15, 24, -6, 4, 12, -12, -1...
$ carrier   <chr> "AS", "US", "UA", "US", "AS", "DL", "UA", "UA", "UA", "UA...
$ tailnum   <chr> "N508AS", "N195UW", "N37422", "N547UW", "N762AS", "N806DN...
$ flight    <int> 145, 1830, 1609, 466, 121, 1823, 1481, 229, 1576, 478, 15...
$ origin    <chr> "PDX", "SEA", "PDX", "PDX", "SEA", "SEA", "SEA", "PDX", "...
$ dest      <chr> "ANC", "CLT", "IAH", "CLT", "ANC", "DTW", "ORD", "IAH", "...
$ air_time  <dbl> 194, 252, 201, 251, 201, 224, 202, 217, 136, 268, 130, 12...
$ distance  <dbl> 1542, 2279, 1825, 2282, 1448, 1927, 1721, 1825, 1024, 240...
$ hour      <dbl> 0, 0, 0, 0, 0, 0, 3, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, ...
$ minute    <dbl> 1, 4, 8, 28, 34, 37, 46, 26, 27, 36, 41, 49, 50, 57, 57, ...
```

`dplyr::select` comes with several other helper functions...

```
depart <- flights %>%
  select(starts_with("dep_"))
glimpse(depart)
```

```
Observations: 162,049
Variables: 2
$ dep_time  <int> 1, 4, 8, 28, 34, 37, 346, 526, 527, 536, 541, 549, 550, 5...
$ dep_delay <dbl> 96, -6, 13, -2, 44, 82, 227, -4, 7, 1, 1, 24, 0, -3, -3, ...
```

```
times <- flights %>%
  select(contains("time"))
glimpse(times)
```

```
Observations: 162,049
Variables: 3
$ dep_time <int> 1, 4, 8, 28, 34, 37, 346, 526, 527, 536, 541, 549, 550, 55...
$ arr_time <int> 235, 738, 548, 800, 325, 747, 936, 1148, 917, 1334, 911, 9...
$ air_time <dbl> 194, 252, 201, 251, 201, 224, 202, 217, 136, 268, 130, 122...
```

```
# note that we are not actually saving the new dataframe here
flights %>%
  select(-contains("time")) %>% head()
```

year	month	day	dep_delay	arr_delay	carrier	tailnum	flight	origin	dest	distance	hour
2014	1	1	96	70	AS	N508AS	145	PDX	ANC	1.54e+03	0
2014	1	1	-6	-23	US	N195UW	1830	SEA	CLT	2.28e+03	0
2014	1	1	13	-4	UA	N37422	1609	PDX	IAH	1.82e+03	0
2014	1	1	-2	-23	US	N547UW	466	PDX	CLT	2.28e+03	0
2014	1	1	44	43	AS	N762AS	121	SEA	ANC	1.45e+03	0
2014	1	1	82	88	DL	N806DN	1823	SEA	DTW	1.93e+03	0

```
delays <- flights %>%
  select(ends_with("delay"))
glimpse(delays)
```

```
Observations: 162,049
Variables: 2
$ dep_delay <dbl> 96, -6, 13, -2, 44, 82, 227, -4, 7, 1, 1, 24, 0, -3, -3, ...
$ arr_delay <dbl> 70, -23, -4, -23, 43, 88, 219, 15, 24, -6, 4, 12, -12, -1...
```

One of my favorite select helper functions is `everything()`, which allows you to use select to keep **all** your variables, but easily rearrange the columns without having to list all the variables to keep/drop.

```
new_order <- flights %>%
  select(origin, dest, everything())
head(new_order)
```

origin	dest	year	month	day	dep_time	dep_delay	arr_time	arr_delay	carrier	tailnum	fl
PDX	ANC	2014	1	1	1	96	235	70	AS	N508AS	
SEA	CLT	2014	1	1	4	-6	738	-23	US	N195UW	
PDX	IAH	2014	1	1	8	13	548	-4	UA	N37422	
PDX	CLT	2014	1	1	28	-2	800	-23	US	N547UW	
SEA	ANC	2014	1	1	34	44	325	43	AS	N762AS	
SEA	DTW	2014	1	1	37	82	747	88	DL	N806DN	

```
# with negation
new_order2 <- flights %>%
  select(origin, dest, everything(), -year)
head(new_order2)
```

origin	dest	month	day	dep_time	dep_delay	arr_time	arr_delay	carrier	tailnum	flight	a
PDX	ANC	1	1	1	96	235	70	AS	N508AS	145	
SEA	CLT	1	1	4	-6	738	-23	US	N195UW	1830	
PDX	IAH	1	1	8	13	548	-4	UA	N37422	1609	
PDX	CLT	1	1	28	-2	800	-23	US	N547UW	466	
SEA	ANC	1	1	34	44	325	43	AS	N762AS	121	
SEA	DTW	1	1	37	82	747	88	DL	N806DN	1823	

We can also rename variables within select.

```
flights2 <- flights %>%
  select(tail_num = tailnum, everything())
head(flights2)
```

tail_num	year	month	day	dep_time	dep_delay	arr_time	arr_delay	carrier	flight	origin
N508AS	2014	1	1	1	96	235	70	AS	145	PDX
N195UW	2014	1	1	4	-6	738	-23	US	1830	SEA
N37422	2014	1	1	8	13	548	-4	UA	1609	PDX
N547UW	2014	1	1	28	-2	800	-23	US	466	PDX
N762AS	2014	1	1	34	44	325	43	AS	121	SEA
N806DN	2014	1	1	37	82	747	88	DL	1823	SEA

If you don't want to move the renamed variables within your dataframe, you can use the `rename` function.

```
flights3 <- flights %>%
  rename(tail_num = tailnum)
```

```
Error in rename(., tail_num = tailnum): unused argument (tail_num = tailnum)
```

```
glimpse(flights3)
```

```
Error in glimpse(flights3): object 'flights3' not found
```

dplyr::filter

```
# flights taking off from PDX
pdx <- flights %>%
  filter(origin == "PDX")
head(pdx)
```

year	month	day	dep_time	dep_delay	arr_time	arr_delay	carrier	tailnum	flight	origin	d
2014	1	1	1	96	235	70	AS	N508AS	145	PDX	A
2014	1	1	8	13	548	-4	UA	N37422	1609	PDX	L
2014	1	1	28	-2	800	-23	US	N547UW	466	PDX	C
2014	1	1	526	-4	1148	15	UA	N813UA	229	PDX	L
2014	1	1	541	1	911	4	UA	N36476	1569	PDX	D
2014	1	1	549	24	907	12	US	N548UW	649	PDX	P

```r
# january flights from PDX
pdx_jan <- flights %>%
  filter(origin == "PDX", month == 1) # the comma is an "and"
head(pdx_jan)
```

year	month	day	dep_time	dep_delay	arr_time	arr_delay	carrier	tailnum	flight	origin	d
2014	1	1	1	96	235	70	AS	N508AS	145	PDX	A
2014	1	1	8	13	548	-4	UA	N37422	1609	PDX	L
2014	1	1	28	-2	800	-23	US	N547UW	466	PDX	C
2014	1	1	526	-4	1148	15	UA	N813UA	229	PDX	L
2014	1	1	541	1	911	4	UA	N36476	1569	PDX	D
2014	1	1	549	24	907	12	US	N548UW	649	PDX	P

```r
# flights to ATL (Atlanta) or BNA (Nashville)
to_south <- flights %>%
  filter(dest == "ATL" | dest == "BNA") %>% # | is "or"
  select(origin, dest, everything())
head(to_south)
```

origin	dest	year	month	day	dep_time	dep_delay	arr_time	arr_delay	carrier	tailnum	flig
SEA	ATL	2014	1	1	624	-6	1401	-6	DL	N617DL	9
SEA	ATL	2014	1	1	802	-3	1533	-17	AS	N532AS	1
SEA	ATL	2014	1	1	824	-1	1546	-14	DL	N633DL	1
PDX	ATL	2014	1	1	944	-6	1727	-8	AS	N548AS	1
PDX	ATL	2014	1	1	1054	94	1807	84	DL	N377DA	15
SEA	ATL	2014	1	1	1158	6	1915	-14	DL	N6712B	19

```r
# flights from PDX to ATL (Atlanta) or BNA (Nashville)
pdx_to_south <- flights %>%
  filter(origin == "PDX", dest == "ATL" | dest == "BNA") %>% # | is "or"
  select(origin, dest, everything())
head(pdx_to_south)
```

origin	dest	year	month	day	dep_time	dep_delay	arr_time	arr_delay	carrier	tailnum	flig
PDX	ATL	2014	1	1	944	-6	1727	-8	AS	N548AS	1
PDX	ATL	2014	1	1	1054	94	1807	84	DL	N377DA	15
PDX	ATL	2014	1	1	1323	-2	2038	-15	DL	N393DA	1
PDX	ATL	2014	1	1	2253	8	611	4	DL	N371DA	5
PDX	ATL	2014	1	2	627	-3	1350	-7	DL	N3746H	11
PDX	ATL	2014	1	2	918	-2	1643	-2	DL	N3756	19

```r
# alternatively, using group membership
south_dests <- c("ATL", "BNA")
pdx_to_south2 <- flights %>%
  filter(origin == "PDX", dest %in% south_dests) %>%
  select(origin, dest, everything())
head(pdx_to_south2)
```

```r
# flights delayed by 1 hour or more
delay_1plus <- flights %>%
  filter(dep_delay >= 60)
head(delay_1plus)
```

origin	dest	year	month	day	dep_time	dep_delay	arr_time	arr_delay	carrier	tailnum	flig
PDX	ATL	2014	1	1	944	-6	1727	-8	AS	N548AS	
PDX	ATL	2014	1	1	1054	94	1807	84	DL	N377DA	15
PDX	ATL	2014	1	1	1323	-2	2038	-15	DL	N393DA	
PDX	ATL	2014	1	1	2253	8	611	4	DL	N371DA	5
PDX	ATL	2014	1	2	627	-3	1350	-7	DL	N3746H	1
PDX	ATL	2014	1	2	918	-2	1643	-2	DL	N3756	19

year	month	day	dep_time	dep_delay	arr_time	arr_delay	carrier	tailnum	flight	origin	d
2014	1	1	1	96	235	70	AS	N508AS	145	PDX	A
2014	1	1	37	82	747	88	DL	N806DN	1823	SEA	D
2014	1	1	346	227	936	219	UA	N14219	1481	SEA	O
2014	1	1	650	90	1037	91	US	N626AW	460	SEA	P
2014	1	1	959	164	1137	157	AS	N534AS	805	SEA	S
2014	1	1	1008	68	1242	64	AS	N788AS	456	SEA	L

```
# flights delayed by 1 hour, but not more than 2 hours
delay_1hr <- flights %>%
  filter(dep_delay >= 60, dep_delay < 120)
head(delay_1hr)
```

year	month	day	dep_time	dep_delay	arr_time	arr_delay	carrier	tailnum	flight	origin	d
2014	1	1	1	96	235	70	AS	N508AS	145	PDX	A
2014	1	1	37	82	747	88	DL	N806DN	1823	SEA	D
2014	1	1	650	90	1037	91	US	N626AW	460	SEA	P
2014	1	1	1008	68	1242	64	AS	N788AS	456	SEA	L
2014	1	1	1014	75	1613	81	UA	N37408	1444	SEA	O
2014	1	1	1036	81	1408	63	OO	N218AG	3466	PDX	T

```
range(delay_1hr$dep_delay, na.rm = TRUE)
```

```
[1]  60 119
```

```
# even more efficient using between (always inclusive)
delay_bwn <- flights %>%
  filter(between(dep_delay, 60, 119))
head(delay_bwn)
```

```
range(delay_bwn$dep_delay, na.rm = TRUE)
```

```
[1]  60 119
```

dplyr::arrange

```
# default is ascending order
flights %>%
  arrange(year, month, day) %>% head(n=20)
```

```
# descending order
flights %>%
  arrange(desc(year), desc(month), desc(day)) %>% head(n=20)
```

year	month	day	dep_time	dep_delay	arr_time	arr_delay	carrier	tailnum	flight	origin	d
2014	1	1	1	96	235	70	AS	N508AS	145	PDX	A
2014	1	1	37	82	747	88	DL	N806DN	1823	SEA	D
2014	1	1	650	90	1037	91	US	N626AW	460	SEA	P
2014	1	1	1008	68	1242	64	AS	N788AS	456	SEA	L
2014	1	1	1014	75	1613	81	UA	N37408	1444	SEA	O
2014	1	1	1036	81	1408	63	OO	N218AG	3466	PDX	T

year	month	day	dep_time	dep_delay	arr_time	arr_delay	carrier	tailnum	flight	origin	d
2014	1	1	1	96	235	70	AS	N508AS	145	PDX	A
2014	1	1	4	-6	738	-23	US	N195UW	1830	SEA	C
2014	1	1	8	13	548	-4	UA	N37422	1609	PDX	L
2014	1	1	28	-2	800	-23	US	N547UW	466	PDX	C
2014	1	1	34	44	325	43	AS	N762AS	121	SEA	A
2014	1	1	37	82	747	88	DL	N806DN	1823	SEA	D
2014	1	1	346	227	936	219	UA	N14219	1481	SEA	C
2014	1	1	526	-4	1148	15	UA	N813UA	229	PDX	L
2014	1	1	527	7	917	24	UA	N75433	1576	SEA	D
2014	1	1	536	1	1334	-6	UA	N574UA	478	SEA	E
2014	1	1	541	1	911	4	UA	N36476	1569	PDX	D
2014	1	1	549	24	907	12	US	N548UW	649	PDX	P
2014	1	1	550	0	837	-12	DL	N660DL	1634	SEA	S
2014	1	1	557	-3	1134	-16	AA	N3JLAA	1094	SEA	D
2014	1	1	557	-3	825	-25	AS	N562AS	81	SEA	A
2014	1	1	558	-2	801	-2	AS	N402AS	200	SEA	S
2014	1	1	559	-1	916	-9	F9	N210FR	796	PDX	D
2014	1	1	600	0	1151	-19	AA	N3JFAA	2240	SEA	C
2014	1	1	600	-10	842	-8	AS	N786AS	426	SEA	L
2014	1	1	602	-3	943	5	F9	N201FR	144	SEA	D

dplyr::distinct

```
# all unique origin-dest combinations
flights %>%
  select(origin, dest) %>%
  distinct %>% head(n=100)
```

```
# all unique destinations from PDX (there are 49)
from_pdx <- flights %>%
  filter(origin == "PDX") %>%
  select(origin, dest) %>%
  distinct(dest)
head(from_pdx)
```

dplyr::mutate

```
# add total delay variable
flights %>%
  mutate(tot_delay = dep_delay + arr_delay) %>%
  select(origin, dest, ends_with("delay"), everything()) %>%
  head(n=100)
```

```
# flights that were delayed at departure had on time or early arrivals?
arrivals <- flights %>%
```

year	month	day	dep_time	dep_delay	arr_time	arr_delay	carrier	tailnum	flight	origin	
2014	12	31	2	12	601	31	AA	N3JKAA	1230	SEA	
2014	12	31	27	-3	623	3	AA	N3EWAA	1431	SEA	
2014	12	31	39	14	324	4	AS	N762AS	135	SEA	
2014	12	31	40	0	549	0	DL	N757AT	2440	SEA	
2014	12	31	52	-8	917	-21	AA	N3JFAA	371	SEA	
2014	12	31	54	4	621	17	DL	N128DL	1670	PDX	
2014	12	31	56	61	848	80	DL	N655DL	929	SEA	
2014	12	31	512	-3	904	4	US	N653AW	480	SEA	
2014	12	31	515	-5	855	5	US	N580UW	425	PDX	
2014	12	31	534	4	859	7	UA	N34460	1075	PDX	
2014	12	31	546	1	916	-4	WN	N8323C	757	PDX	
2014	12	31	548	-2	1351	-13	UA	N461UA	665	PDX	
2014	12	31	549	4	1208	12	UA	N68807	1457	SEA	
2014	12	31	550	0	922	2	WN	N797MX	2121	PDX	
2014	12	31	551	-4	1202	12	AA	N3HXAA	1094	SEA	
2014	12	31	551	-9	744	-15	AS	N570AS	342	SEA	
2014	12	31	555	-10	824	-1	AS	N548AS	602	SEA	
2014	12	31	558	-2	849	0	DL	N668DN	1831	PDX	
2014	12	31	558	-2	1149	4	AA	N436AA	1534	PDX	
2014	12	31	558	-2	738	-4	AS	N585AS	406	PDX	

```
  mutate(arr_ok = ifelse(dep_delay > 0 & arr_delay <= 0, 1, 0)) %>%
  select(origin, dest, ends_with("delay"), carrier, arr_ok)

# peek at it
arrivals %>%
  filter(arr_ok == 1) %>%
  head
```

dplyr::summarise (or dplyr::summarize)

```
flights %>%
  summarise(mean(dep_delay, na.rm = TRUE))
```

```
# we can also name that variable, and summarise multiple variables
flights %>%
  summarise(mean_delay = mean(dep_delay, na.rm = TRUE),
            sd_delay = sd(dep_delay, na.rm = TRUE),
            median_delay = median(dep_delay, na.rm = TRUE))
```

But this can get tedious with multiple summaries...

```
flights %>%
  filter(!is.na(dep_delay)) %>%
  select(dep_delay) %>%
  summarise_each(lst(mean, sd, median))
```

```
# same thing
flights %>%
  filter(!is.na(dep_delay)) %>%
  summarise_each(lst(mean, sd, median), dep_delay)
```

```
# combine with gather, change names too
flights %>%
  filter(!is.na(dep_delay)) %>%
  summarise_each(lst(mean, stdev = sd, median), dep_delay) %>%
  gather(delay_stat, value)
```

Using aggregating functions in `summarise`

```
# how many unique destinations?
summary_table <- flights %>%
  summarise(tot_flights = n(),
            tot_planes = n_distinct(tailnum),
            tot_carriers = n_distinct(carrier),
            tot_dests = n_distinct(dest),
            tot_origins = n_distinct(origin))
```

```
Error: n() should only be called in a data context
```

```
summary_table
```

```
Error in eval(expr, envir, enclos): object 'summary_table' not found
```

```
# chain with tidyr functions
summary_table %>%
  gather(key, value) %>%
  separate(key, into = c("tot", "entity")) %>%
  select(-tot, total = value)
```

```
Error in eval(lhs, parent, parent): object 'summary_table' not found
```

tidyr

We'll work with a made up dataframe:

```
df <- data.frame(
  id = 1:10,
  date = as.Date('2015-01-01') + 0:9,
  q1_m1_w1 = rnorm(10, 0, 1),
  q1_m1_w2 = rnorm(10, 0, 1),
  q1_m2_w3 = rnorm(10, 0, 1),
  q2_m1_w1 = rnorm(10, 0, 1),
  q2_m2_w1 = rnorm(10, 0, 1),
  q2_m2_w2 = rnorm(10, 0, 1)
)
```

```
# HLO
head(df)
```

```
glimpse(df)
```

```
Observations: 10
Variables: 8
$ id       <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
$ date     <date> 2015-01-01, 2015-01-02, 2015-01-03, 2015-01-04, 2015-01-0...
$ q1_m1_w1 <dbl> 1.3776917, -0.6916492, 1.1723681, -2.0609808, -0.5307731, ...
$ q1_m1_w2 <dbl> -0.51370367, -0.86622778, 0.06480501, 2.09739481, -1.02144...
$ q1_m2_w3 <dbl> -0.52359457, 0.29724294, -1.12298499, -1.13920406, 0.12061...
$ q2_m1_w1 <dbl> 1.6370159, -1.2295158, -0.6781920, 0.7309601, -1.4894654, ...
```

```
$ q2_m2_w1 <dbl> -0.8632530, -0.9046099, -2.0944336, -1.8465203, -0.1512347...
$ q2_m2_w2 <dbl> -1.42694403, 0.94049436, -0.43794666, -0.17760237, -0.1098...
```

tidyr::gather

First, let's gather...

```
df_tidy <- df %>%
  gather(key, value, q1_m1_w1:q2_m2_w2)
head(df_tidy)
```

Now let's gather using subtraction...

```
df_tidy <- df %>%
  gather(key, value, -id, -date)
head(df_tidy)
```

tidyr::separate

```
# separate 1 col into 3 cols
df_sep <- df_tidy %>%
  separate(key, into = c("quarter", "month", "week"))
head(df_sep)
```

```
# separate 1 col into 2 cols
df_sep2 <- df_tidy %>%
  separate(key, into = c("quarter", "period"), extra = "merge")
head(df_sep2)
```

stringr vs. tidyr separate by regular expression

tidyr::extract

`Extract` is essentially the same as `separate`, let's see how...

```
# extract
df_ext <- df_sep2 %>%
  extract(period, into = "month")
head(df_ext)
```

```
# this gives us same output as separate
df_ext <- df_sep2 %>%
  extract(period, into = c("month", "week"),
          regex = "([[:alnum:]]+)_([[:alnum:]]+)")
head(df_ext)
```

tidyr::unite

```
# let's say we want to combine quarter and month with an underscore
df_uni <- df_sep %>%
  unite(period, quarter:month) # sep = "_" is the default arg
head(df_uni)
```

```
# let's say we want to combine quarter and month with nothing
df_uni <- df_sep %>%
  unite(period, quarter:month, sep = "")
head(df_uni)
```

`tidyr::spread`

```
# finally let's spread
df_spread <- df_uni %>%
  spread(week, value) # fill = NA is default arg
head(df_spread)
```

Gather multiple sets of columns (`gather() %>% separate() %>% spread()`)

Gather multiple sets of columns

All in one, if we had wanted to essentially "gather" three sets of columns (here, one for each week)...

```
df_tidiest <- df %>%
  gather(key, value, -id, -date) %>%
  separate(key, into = c("quarter", "month", "week")) %>%
  spread(week, value)
head(df_tidiest)
```

`broom`

"The broom package takes the messy output of built-in functions in R, such as `lm`, `nls`, or `t.test`, and turns them into tidy data frames." So, broom tidies output from other R functions that are un-tidy.

See here for list of functions: https://github.com/dgrtwo/broom

Vignette: ftp://cran.r-project.org/pub/R/web/packages/broom/vignettes/broom.html

```
fit <- lm(mpg ~ qsec + factor(am) + wt + factor(gear),
          data = mtcars)
```

Un-tidy output from `lm`

```
summary(fit)
```

```
Call:
lm(formula = mpg ~ qsec + factor(am) + wt + factor(gear), data = mtcars)

Residuals:
    Min      1Q  Median      3Q     Max
-3.5064 -1.5220 -0.7517  1.3841  4.6345

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)    9.3650     8.3730   1.118  0.27359
qsec           1.2449     0.3828   3.252  0.00317 **
factor(am)1    3.1505     1.9405   1.624  0.11654
wt            -3.9263     0.7428  -5.286 1.58e-05 ***
factor(gear)4 -0.2682     1.6555  -0.162  0.87257
factor(gear)5 -0.2697     2.0632  -0.131  0.89698
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.55 on 26 degrees of freedom
Multiple R-squared:  0.8498,    Adjusted R-squared:  0.8209
F-statistic: 29.43 on 5 and 26 DF,  p-value: 6.379e-10
```

Tidy output from `broom`

```
tidy(fit)
```

Specialized Packages

huxtable

tableone

Vignette: https://cran.r-project.org/web/packages/tableone/vignettes/introduction.html

```
library(tableone)
```

```
CreateTableOne(data = mazes)
```

```
                         Overall
  n                         381
  study_id (%)
     CSLU-001                 4 ( 1.0)
     CSLU-002                 4 ( 1.0)
     CSLU-007                 4 ( 1.0)
     CSLU-010                 4 ( 1.0)
     CSLU-020                 4 ( 1.0)
     CSLU-024                 4 ( 1.0)
     CSLU-027                 4 ( 1.0)
     CSLU-031                 4 ( 1.0)
     CSLU-036                 3 ( 0.8)
     CSLU-046                 4 ( 1.0)
     CSLU-053                 4 ( 1.0)
     CSLU-054                 4 ( 1.0)
     CSLU-059                 4 ( 1.0)
     CSLU-062                 4 ( 1.0)
     CSLU-066                 4 ( 1.0)
     CSLU-073                 4 ( 1.0)
     CSLU-077                 4 ( 1.0)
     CSLU-080                 4 ( 1.0)
     CSLU-082                 3 ( 0.8)
     CSLU-084                 4 ( 1.0)
     CSLU-089                 4 ( 1.0)
     CSLU-095                 3 ( 0.8)
     CSLU-096                 4 ( 1.0)
     CSLU-101                 4 ( 1.0)
     CSLU-104                 4 ( 1.0)
     CSLU-112                 4 ( 1.0)
     CSLU-117                 4 ( 1.0)
     CSLU-119                 4 ( 1.0)
     CSLU-122                 4 ( 1.0)
     CSLU-124                 3 ( 0.8)
     CSLU-142                 4 ( 1.0)
     CSLU-144                 4 ( 1.0)
     CSLU-146                 4 ( 1.0)
     CSLU-154                 4 ( 1.0)
     CSLU-156                 4 ( 1.0)
     CSLU-161                 4 ( 1.0)
```

CSLU-163	4 (1.0)
CSLU-165	4 (1.0)
CSLU-167	4 (1.0)
CSLU-180	4 (1.0)
CSLU-191	4 (1.0)
CSLU-203	4 (1.0)
CSLU-204	4 (1.0)
CSLU-213	4 (1.0)
CSLU-216	4 (1.0)
CSLU-220	4 (1.0)
CSLU-226	4 (1.0)
CSLU-233	4 (1.0)
CSLU-238	4 (1.0)
CSLU-245	4 (1.0)
CSLU-258	4 (1.0)
CSLU-259	4 (1.0)
CSLU-263	4 (1.0)
CSLU-269	4 (1.0)
CSLU-274	4 (1.0)
CSLU-275	4 (1.0)
CSLU-277	4 (1.0)
CSLU-284	4 (1.0)
CSLU-290	4 (1.0)
CSLU-303	4 (1.0)
CSLU-306	4 (1.0)
CSLU-312	4 (1.0)
CSLU-315	4 (1.0)
CSLU-316	4 (1.0)
CSLU-320	4 (1.0)
CSLU-324	4 (1.0)
CSLU-332	4 (1.0)
CSLU-335	4 (1.0)
CSLU-339	4 (1.0)
CSLU-348	4 (1.0)
CSLU-349	4 (1.0)
CSLU-355	4 (1.0)
CSLU-359	4 (1.0)
CSLU-372	4 (1.0)
CSLU-373	4 (1.0)
CSLU-375	4 (1.0)
CSLU-379	4 (1.0)
CSLU-388	2 (0.5)
CSLU-389	4 (1.0)
CSLU-393	4 (1.0)
CSLU-395	4 (1.0)
CSLU-417	4 (1.0)
CSLU-419	4 (1.0)
CSLU-427	4 (1.0)
CSLU-432	3 (0.8)
CSLU-435	4 (1.0)
CSLU-441	4 (1.0)
CSLU-442	4 (1.0)
CSLU-447	4 (1.0)
CSLU-454	4 (1.0)

```
    CSLU-460                   4 ( 1.0)
    CSLU-470                   4 ( 1.0)
    CSLU-472                   4 ( 1.0)
    CSLU-477                   4 ( 1.0)
    CSLU-482                   4 ( 1.0)
    CSLU-486                   4 ( 1.0)
    CSLU-499                   4 ( 1.0)
  ca (mean (SD))             6.83 (1.06)
  viq (mean (SD))          100.82 (18.74)
  dx (%)
    ASD                      183 (48.0)
    SLI                       71 (18.6)
    TD                       127 (33.3)
  activity (%)
    Conversation              94 (24.7)
    Picture Description       94 (24.7)
    Play                      96 (25.2)
    Wordless Picture Book     97 (25.5)
  content (mean (SD))       18.73 (24.84)
  filler (mean (SD))        11.20 (17.59)
  rep (mean (SD))            6.24 (9.45)
  rev (mean (SD))            3.79 (4.31)
  fs (mean (SD))             8.70 (12.76)
  cued (mean (SD))          14.36 (24.22)
  not_cued (mean (SD))      26.77 (31.73)
```

```
my_maze_names <- c("Participant", "Age", "Verbal\nIQ", "Group", "Activity", "Content\nMaze", "Filler\nM
## Vector of variables to summarize
my_num_vars <- c("ca", "viq", "content", "filler", "rep", "rev", "fs", "cued", "not_cued")
## Vector of categorical variables that need transformation
my_cat_vars <- c("dx", "activity")
## Create a TableOne object
tab2 <- CreateTableOne(vars = my_num_vars, data = mazes, factorVars = my_cat_vars)
print(tab2, showAllLevels = TRUE)
```

```
                       level Overall
  n                              381
  ca (mean (SD))                6.83 (1.06)
  viq (mean (SD))             100.82 (18.74)
  content (mean (SD))          18.73 (24.84)
  filler (mean (SD))           11.20 (17.59)
  rep (mean (SD))               6.24 (9.45)
  rev (mean (SD))               3.79 (4.31)
  fs (mean (SD))                8.70 (12.76)
  cued (mean (SD))             14.36 (24.22)
  not_cued (mean (SD))         26.77 (31.73)
```

```
tab3 <- CreateTableOne(vars = my_num_vars, strata = "dx" , data = mazes)
tab3
```

```
                     Stratified by dx
                      ASD           SLI           TD            p      test
  n                     183            71           127
  ca (mean (SD))       6.74 (1.11)   7.15 (1.00)   6.76 (0.97)   0.015
  viq (mean (SD))     95.29 (17.62) 86.24 (5.95)  116.94 (12.82) <0.001
```

```
  content (mean (SD))  20.46 (29.73) 17.34 (24.35)  17.00 (15.67)  0.422
  filler (mean (SD))    7.86 (13.54) 10.56 (16.35)  16.38 (21.84) <0.001
  rep (mean (SD))       7.25 (11.82)  5.45 (6.86)    5.23 (6.21)   0.134
  rev (mean (SD))       3.87 (4.85)   3.25 (3.55)    3.98 (3.85)   0.498
  fs (mean (SD))        9.35 (14.60)  8.63 (15.00)   7.80 (7.55)   0.574
  cued (mean (SD))     10.66 (21.94) 13.21 (22.54)  20.35 (27.10)  0.002
  not_cued (mean (SD)) 25.52 (33.49) 25.25 (31.84)  29.41 (29.04)  0.517
```

The DT package

An excellent tutorial on DT is available at https://rstudio.github.io/DT/.

```
datatable(mazes)
```

xtable (best for html)

The xtable is a solution that delivers both HTML and LaTeX. The syntax is very similar to kable:

```
output <-
  matrix(sprintf("Content %s", LETTERS[1:4]),
         ncol=2, byrow=TRUE)
colnames(output) <-
  c("1st header", "2nd header")
rownames(output) <-
  c("1st row", "2nd row")

print(xtable(output,
             caption="A test table",
             align = c("l", "c", "r")),
      type="html")
```

```
<!-- html table generated in R 3.6.3 by xtable 1.8-4 package -->
<!-- Sat Mar 28 14:10:33 2020 -->
<table border=1>
<caption align="bottom"> A test table </caption>
<tr> <th>  </th> <th> 1st header </th> <th> 2nd header </th>  </tr>
  <tr> <td> 1st row </td> <td align="center"> Content A </td> <td align="right"> Content B </td> </tr>
  <tr> <td> 2nd row </td> <td align="center"> Content C </td> <td align="right"> Content D </td> </tr>
   </table>
```

Note that to make it `knit`, you need to specify a chunk option: `results = 'asis'`

```
print(xtable(output,
             caption="A test table",
             align = c("l", "c", "r")),
      type="html")
```

A test table

1st header

2nd header

1st row

Content A

Content B

2nd row

Content C

Content D

```
print(xtable(head(iris)), type = 'html', html.table.attributes = '')
```

Sepal.Length

Sepal.Width

Petal.Length

Petal.Width

Species

1

5.10

3.50

1.40

0.20

setosa

2

4.90

3.00

1.40

0.20

setosa

3

4.70

3.20

1.30

0.20

setosa

4

4.60

3.10

1.50

0.20

setosa

5

5.00

3.60

1.40

0.20

setosa

6

5.40

3.90

1.70

0.40

setosa

`pixiedust` (best for PDF)

Remember that `broom` package we used earlier? We can make this table better...

```
tidy(fit)
```

https://cran.r-project.org/web/packages/pixiedust/vignettes/pixiedust.html

http://www.suchanutter.net/pixiedust/index.html

```
dust(fit) %>%
  sprinkle(cols = "term",
           replace = c("Intercept", "Quarter Mile Time", "Automatic vs. Manual",
                       "Weight", "Gears: 4 vs. 3", "Gears: 5 vs 3")) %>%
  sprinkle(cols = c("estimate", "std.error", "statistic"),
           round = 3) %>%
  sprinkle(cols = "p.value", fn = quote(pvalString(value))) %>%
  sprinkle_colnames("Term", "Coefficient", "SE", "T-statistic", "P-value")
```

Finally, fonts!

https://github.com/wch/extrafont

Follow all installation instructions from `github`

origin	dest
PDX	ANC
SEA	CLT
PDX	IAH
PDX	CLT
SEA	ANC
SEA	DTW
SEA	ORD
SEA	DEN
SEA	EWR
PDX	DEN
PDX	PHX
SEA	SLC
SEA	DFW
SEA	SJC
SEA	LAX
PDX	ORD
SEA	OAK
SEA	SFO
PDX	SJC
SEA	SNA
SEA	SAN
PDX	DFW
PDX	EWR
SEA	IAH
SEA	ATL
PDX	BUR
SEA	MDW
PDX	LAX
PDX	SAN
PDX	IAD
SEA	PSP
PDX	MDW
SEA	MSP
PDX	SNA
SEA	PHX
PDX	HNL
SEA	JFK
PDX	MSP
PDX	SFO
SEA	LAS
SEA	LGB
PDX	ONT
PDX	LGB
PDX	JFK
PDX	KOA
PDX	LAS
SEA	ONT
SEA	MKE
SEA	BUR
SEA	KTN
PDX	OAK
SEA	DCA
SEA	ABQ
SEA	IAD
SEA	MCO
SEA	OGG
SEA	MCI
SEA	PHL
PDX	SLC
SEA	LIH

dest
ANC
IAH
CLT
DEN
PHX
ORD

origin	dest	dep_delay	arr_delay	tot_delay	year	month	day	dep_time	arr_time	carrier
PDX	ANC	96	70	166	2014	1	1	1	235	AS
SEA	CLT	-6	-23	-29	2014	1	1	4	738	US
PDX	IAH	13	-4	9	2014	1	1	8	548	UA
PDX	CLT	-2	-23	-25	2014	1	1	28	800	US
SEA	ANC	44	43	87	2014	1	1	34	325	AS
SEA	DTW	82	88	170	2014	1	1	37	747	DL
SEA	ORD	227	219	446	2014	1	1	346	936	UA
PDX	IAH	-4	15	11	2014	1	1	526	1148	UA
SEA	DEN	7	24	31	2014	1	1	527	917	UA
SEA	EWR	1	-6	-5	2014	1	1	536	1334	UA
PDX	DEN	1	4	5	2014	1	1	541	911	UA
PDX	PHX	24	12	36	2014	1	1	549	907	US
SEA	SLC	0	-12	-12	2014	1	1	550	837	DL
SEA	DFW	-3	-16	-19	2014	1	1	557	1134	AA
SEA	ANC	-3	-25	-28	2014	1	1	557	825	AS
SEA	SJC	-2	-2	-4	2014	1	1	558	801	AS
PDX	DEN	-1	-9	-10	2014	1	1	559	916	F9
SEA	ORD	0	-19	-19	2014	1	1	600	1151	AA
SEA	LAX	-10	-8	-18	2014	1	1	600	842	AS
SEA	DEN	-3	5	2	2014	1	1	602	943	F9
PDX	ORD	-3	7	4	2014	1	1	602	1204	UA
SEA	OAK	-2	-17	-19	2014	1	1	603	755	AS
SEA	ORD	-3	-2	-5	2014	1	1	603	1202	UA
SEA	SFO	-4	-19	-23	2014	1	1	606	806	AS
PDX	SJC	6	3	9	2014	1	1	606	746	AS
SEA	SNA	-1	-2	-3	2014	1	1	614	850	AS
SEA	SAN	2	-12	-10	2014	1	1	617	850	AS
PDX	DFW	-2	-30	-32	2014	1	1	618	1135	AA
SEA	SFO	-6	-7	-13	2014	1	1	619	822	VX
SEA	LAX	-2	0	-2	2014	1	1	620	905	OO
PDX	EWR	2	-19	-17	2014	1	1	622	1412	UA
SEA	IAH	13	-4	9	2014	1	1	623	1218	UA
SEA	ATL	-6	-6	-12	2014	1	1	624	1401	DL
SEA	DEN	-9	-1	-10	2014	1	1	629	1014	UA
PDX	BUR	-10	-14	-24	2014	1	1	630	834	OO
SEA	MDW	-3	5	2	2014	1	1	632	1235	WN
PDX	LAX	-8	-7	-15	2014	1	1	637	858	AS
PDX	SAN	-3	-6	-9	2014	1	1	637	854	AS
PDX	PHX	-2	-5	-7	2014	1	1	638	1003	AS
PDX	IAD	10	-4	6	2014	1	1	638	1408	UA
SEA	PSP	-1	3	2	2014	1	1	639	918	AS
SEA	DFW	-1	-10	-11	2014	1	1	639	1216	AS
PDX	MDW	4	-11	-7	2014	1	1	639	1219	WN
SEA	LAX	-3	-2	-5	2014	1	1	647	923	AS
SEA	DEN	-3	19	16	2014	1	1	647	1038	AS
SEA	MSP	-7	-9	-16	2014	1	1	648	1203	AS
PDX	SNA	-12	-11	-23	2014	1	1	648	910	AS
SEA	PHX	90	91	181	2014	1	1	650	1037	US
PDX	HNL	-6	-33	-39	2014	1	1	654	1047	AS
SEA	JFK	-6	-10	-16	2014	1	1	654	1455	DL
PDX	MSP	-5	-7	-12	2014	1	1	655	1210	DL
PDX	ORD	-4	-28	-32	2014	1	1	656	1242	AA
PDX	SFO	-4	8	4	2014	1	1	656	853	VX
PDX	SFO	0	-1	-1	2014	1	1	700	844	UA
SEA	LAS	-4	-6	-10	2014	1	1	701	918	AS
SEA	PHX	1	-9	-8	2014	1	1	701	1036	WN
SEA	LAX	-3	-8	-11	2014	1	1	702	932	VX
SEA	LGB	-8	-2	-10	2014	1	1	702	940	OO
SEA	DFW	-6	-20	-26	2014	1	1	704	1245	AA
PDX	ONT	-1	-9	-10	2014	1	1	704	910	OO

origin	dest	dep_delay	arr_delay	carrier	arr_ok
PDX	IAH	13	-4	UA	1
SEA	EWR	1	-6	UA	1
SEA	SAN	2	-12	AS	1
PDX	EWR	2	-19	UA	1
SEA	IAH	13	-4	UA	1
PDX	IAD	10	-4	UA	1

mean(dep_delay, na.rm = TRUE)
6.13

mean_delay	sd_delay	median_delay
6.13	29.1	-2

	mean	sd	median
	6.13	29.1	-2

	mean	sd	median
	6.13	29.1	-2

delay_stat	value
mean	6.13
stdev	29.1
median	-2

id	date	q1_m1_w1	q1_m1_w2	q1_m2_w3	q2_m1_w1	q2_m2_w1	q2_m2_w2
1	2015-01-01	1.38	-0.514	-0.524	1.64	-0.863	-1.43
2	2015-01-02	-0.692	-0.866	0.297	-1.23	-0.905	0.94
3	2015-01-03	1.17	0.0648	-1.12	-0.678	-2.09	-0.438
4	2015-01-04	-2.06	2.1	-1.14	0.731	-1.85	-0.178
5	2015-01-05	-0.531	-1.02	0.121	-1.49	-0.151	-0.11
6	2015-01-06	-0.371	-0.837	-0.172	-0.762	-0.835	-0.594

id	date	key	value
1	2015-01-01	q1_m1_w1	1.38
2	2015-01-02	q1_m1_w1	-0.692
3	2015-01-03	q1_m1_w1	1.17
4	2015-01-04	q1_m1_w1	-2.06
5	2015-01-05	q1_m1_w1	-0.531
6	2015-01-06	q1_m1_w1	-0.371

id	date	key	value
1	2015-01-01	q1_m1_w1	1.38
2	2015-01-02	q1_m1_w1	-0.692
3	2015-01-03	q1_m1_w1	1.17
4	2015-01-04	q1_m1_w1	-2.06
5	2015-01-05	q1_m1_w1	-0.531
6	2015-01-06	q1_m1_w1	-0.371

id	date	quarter	month	week	value
1	2015-01-01	q1	m1	w1	1.38
2	2015-01-02	q1	m1	w1	-0.692
3	2015-01-03	q1	m1	w1	1.17
4	2015-01-04	q1	m1	w1	-2.06
5	2015-01-05	q1	m1	w1	-0.531
6	2015-01-06	q1	m1	w1	-0.371

id	date	quarter	period	value
1	2015-01-01	q1	m1_w1	1.38
2	2015-01-02	q1	m1_w1	-0.692
3	2015-01-03	q1	m1_w1	1.17
4	2015-01-04	q1	m1_w1	-2.06
5	2015-01-05	q1	m1_w1	-0.531
6	2015-01-06	q1	m1_w1	-0.371

id	date	quarter	month	value
1	2015-01-01	q1	m1	1.38
2	2015-01-02	q1	m1	-0.692
3	2015-01-03	q1	m1	1.17
4	2015-01-04	q1	m1	-2.06
5	2015-01-05	q1	m1	-0.531
6	2015-01-06	q1	m1	-0.371

id	date	quarter	month	week	value
1	2015-01-01	q1	m1	w1	1.38
2	2015-01-02	q1	m1	w1	-0.692
3	2015-01-03	q1	m1	w1	1.17
4	2015-01-04	q1	m1	w1	-2.06
5	2015-01-05	q1	m1	w1	-0.531
6	2015-01-06	q1	m1	w1	-0.371

id	date	period	week	value
1	2015-01-01	q1_m1	w1	1.38
2	2015-01-02	q1_m1	w1	-0.692
3	2015-01-03	q1_m1	w1	1.17
4	2015-01-04	q1_m1	w1	-2.06
5	2015-01-05	q1_m1	w1	-0.531
6	2015-01-06	q1_m1	w1	-0.371

id	date	period	week	value
1	2015-01-01	q1m1	w1	1.38
2	2015-01-02	q1m1	w1	-0.692
3	2015-01-03	q1m1	w1	1.17
4	2015-01-04	q1m1	w1	-2.06
5	2015-01-05	q1m1	w1	-0.531
6	2015-01-06	q1m1	w1	-0.371

id	date	period	w1	w2	w3
1	2015-01-01	q1m1	1.38	-0.514	
1	2015-01-01	q1m2			-0.524
1	2015-01-01	q2m1	1.64		
1	2015-01-01	q2m2	-0.863	-1.43	
2	2015-01-02	q1m1	-0.692	-0.866	
2	2015-01-02	q1m2			0.297

id	date	quarter	month	w1	w2	w3
1	2015-01-01	q1	m1	1.38	-0.514	
1	2015-01-01	q1	m2			-0.524
1	2015-01-01	q2	m1	1.64		
1	2015-01-01	q2	m2	-0.863	-1.43	
2	2015-01-02	q1	m1	-0.692	-0.866	
2	2015-01-02	q1	m2			0.297

term	estimate	std.error	statistic	p.value
(Intercept)	9.37	8.37	1.12	0.274
qsec	1.24	0.383	3.25	0.00317
factor(am)1	3.15	1.94	1.62	0.117
wt	-3.93	0.743	-5.29	1.58e-05
factor(gear)4	-0.268	1.66	-0.162	0.873
factor(gear)5	-0.27	2.06	-0.131	0.897

term	estimate	std.error	statistic	p.value
(Intercept)	9.37	8.37	1.12	0.274
qsec	1.24	0.383	3.25	0.00317
factor(am)1	3.15	1.94	1.62	0.117
wt	-3.93	0.743	-5.29	1.58e-05
factor(gear)4	-0.268	1.66	-0.162	0.873
factor(gear)5	-0.27	2.06	-0.131	0.897

Term	Coefficient	SE	T-statistic	P-value
Intercept	9.365	8.373	1.118	0.27
Quarter Mile Time	1.245	0.383	3.252	0.003
Automatic vs. Manual	3.151	1.941	1.624	0.12
Weight	-3.926	0.743	-5.286	< 0.001
Gears: 4 vs. 3	-0.268	1.655	-0.162	0.87
Gears: 5 vs 3	-0.27	2.063	-0.131	0.9