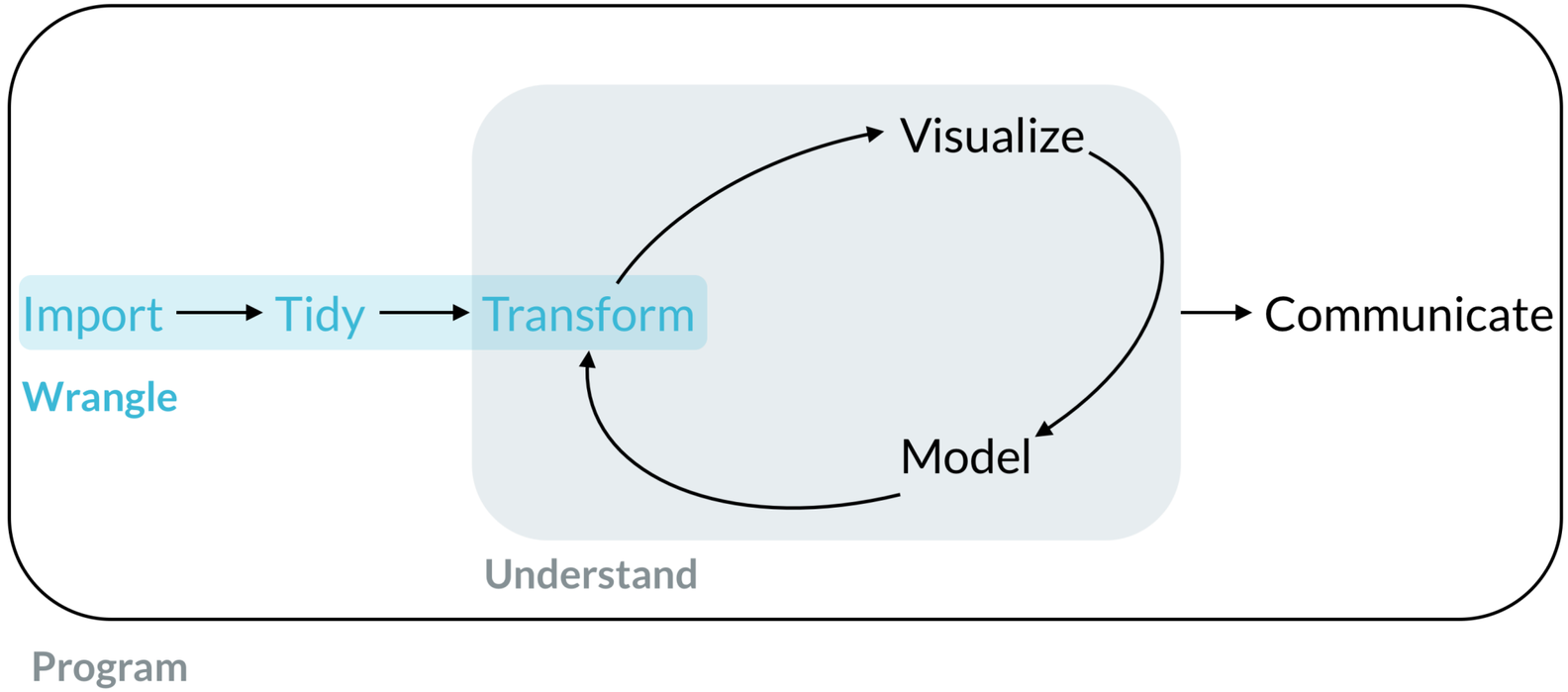
Import Your Data

WORKING WITH DATA IN THE TIDYVERSE

Alison Hill
Professor & Data Scientist



[1] R for Data Science (https://r4ds.had.co.nz/wrangle [2] intro.html)

Rectangular data

Columns hold variables

series	baker	age	num_episodes	aired_us	last_date_uk
3	Natasha	36	1	FALSE	2012-08-14
3	Peter	43	2	FALSE	2012-08-21
3	Victoria	50	3	FALSE	2012-08-28
4	Toby	30	1	TRUE	2013-08-20
4	Lucy	38	2	TRUE	2013-08-27
4	Deborah	51	3	TRUE	2013-09-03
4	Mark	37	3	TRUE	2013-09-03
5	Claire	31	1	TRUE	2014-08-06
5	Enwezor	39	2	TRUE	2014-08-13
5	Jordan	32	3	TRUE	2014-08-20

Rectangular data

series	baker	age	num_episodes	aired_us	last_date_uk
3	Natasha	36	1	FALSE	2012-08-14
3	Peter	43	2	FALSE	2012-08-21
3	Victoria	50	3	FALSE	2012-08-28
4	Toby	30	1	TRUE	2013-08-20
4	Lucy	38	2	TRUE	2013-08-27
4	Deborah	51	3	TRUE	2013-09-03
4	Mark	37	3	TRUE	2013-09-03
5	Claire	31	1	TRUE	2014-08-06
5	Enwezor	39	2	TRUE	2014-08-13
5	Jordan	32	3	TRUE	2014-08-20

Rectangular data in R

```
bakers
```

```
# A tibble: 8 x 6
   series baker         age num_episodes aired_us last_date_uk
   <dbl>  <chr>       <dbl>        <dbl> <lgl>    <date>
 1 3      Natasha      36.           1. FALSE    2012-08-14
 2 3      Sarah-Jane   28.           7. FALSE    2012-09-25
 3 3      Cathryn      27.           8. FALSE    2012-10-02
 4 4      Lucy         38.           2. TRUE     2013-08-27
 5 4      Howard       51.           6. TRUE     2013-09-24
 6 4      Beca         31.           9. TRUE     2013-10-15
 7 4      Kimberley    30.          10. TRUE     2013-10-22
 8 5      Enwezor      39.           2. TRUE     2014-08-13
```

The readr package

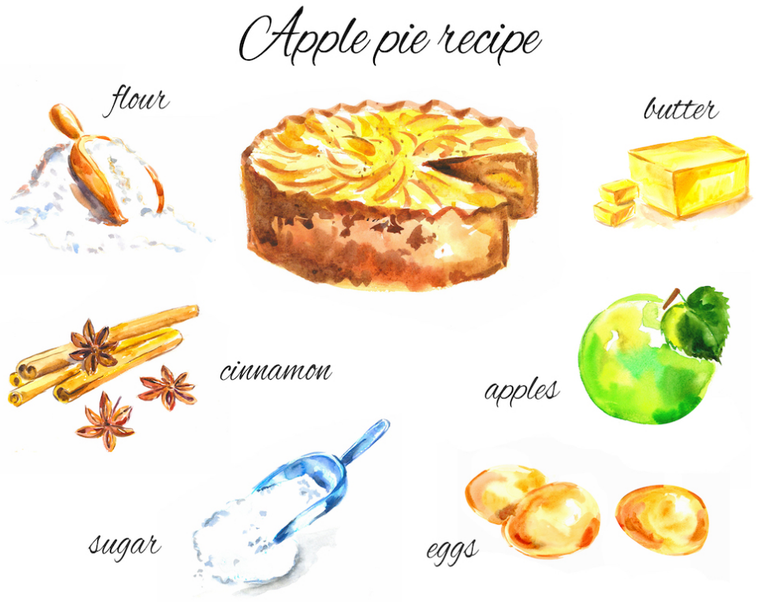
```
library(readr) # once per work session
```

[1] http://readr.tidyverse.org

Functions in R

```
recipe_name(ingredients)
```

```
function_name(arguments)
```



The read_csv function

```
?read_csv
```

Read A Delimited File (Including Csv & Tsv) Into A Tibble

`read_csv()` and `read_tsv()` are special cases of the general `read_delim()`. They're useful for reading the most common types of flat file data, comma separated values and tab separated values, respectively. `read_csv2()` uses `;` for separators, instead of `,`. This is common in European countries which use `,` as the decimal separator.

Usage

```
read_delim(file, delim, quote = "\"", escape_backslash = FALSE,
  escape_double = TRUE, col_names = TRUE, col_types = NULL,
  locale = default_locale(), na = c("", "NA"), quoted_na = TRUE,
  comment = "", trim_ws = FALSE, skip = 0, n_max = Inf,
  guess_max = min(1000, n_max), progress = show_progress())

read_csv(file, col_names = TRUE, col_types = NULL,
  locale = default_locale(), na = c("", "NA"), quoted_na = TRUE,
  quote = "\"", comment = "", trim_ws = TRUE, skip = 0, n_max = Inf,
  guess_max = min(1000, n_max), progress = show_progress())
```

```
?read_csv
```

Usage

```
read_csv(file, col_names = TRUE, col_types = NULL,
  locale = default_locale(), na = c("", "NA"), quoted_na = TRUE,
  quote = "\"", comment = "", trim_ws = TRUE, skip = 0, n_max = Inf,
  guess_max = min(1000, n_max), progress = show_progress())
```

The file argument

```
?read_csv
```

Arguments

file Either a path to a file, a connection, or literal data (either a single string or a raw vector).

Files ending in `.gz`, `.bz2`, `.xz`, or `.zip` will be automatically uncompressed. Files starting with `http://`, `https://`, `ftp://`, or `ftps://` will be automatically downloaded. Remote gz files can also be automatically downloaded and decompressed.

Literal data is most useful for examples and tests. It must contain at least one new line to be recognised as data (instead of a path).

Read the CSV file

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

```
Parsed with column specification:
cols(
  series = col_double(),
  baker = col_character(),
  age = col_double(),
  num_episodes = col_double(),
  aired_us = col_logical(),
  last_date_uk = col_date(format = "")
)
```

Print bakers

```
bakers
```

```
# A tibble: 8 x 6
   series baker         age num_episodes aired_us last_date_uk
   <dbl>  <chr>       <dbl>        <dbl> <lgl>    <date>
 1 3      Natasha      36.           1. FALSE    2012-08-14
 2 3      Sarah-Jane   28.           7. FALSE    2012-09-25
 3 3      Cathryn      27.           8. FALSE    2012-10-02
 4 4      Lucy         38.           2. TRUE     2013-08-27
 5 4      Howard       51.           6. TRUE     2013-09-24
 6 4      Beca         31.           9. TRUE     2013-10-15
 7 4      Kimberley    30.          10. TRUE     2013-10-22
 8 5      Enwezor      39.           2. TRUE     2014-08-13
```

Other functions and packages

Rectangular data

File extensions like: .csv, .tsv, .fwf, .log

SPSS, Stata, & SAS files

File extensions like: .sav, .spss, .dta, .sas

Excel files

File extensions like: .xls, .xlsx

Let's practice!

WORKING WITH DATA IN THE TIDYVERSE

Know Your Data

WORKING WITH DATA IN THE TIDYVERSE

Alison Hill
Professor & Data Scientist

The Great British Bake Off

Signature

Technical

Showstopper

Look at your data

```
bakers_mini
```

```
# A tibble: 8 x 10
   series baker    age num_episodes aired_us last_date_uk
   <fct>  <chr>  <dbl>        <dbl> <lgl>    <date>
 1 3      Natas…   36.           1. FALSE    2012-08-14
 2 3      Sarah…   28.           7. FALSE    2012-09-25
 3 3      Cathr…   27.           8. FALSE    2012-10-02
 4 4      Lucy     38.           2. TRUE     2013-08-27
 5 4      Howard   51.           6. TRUE     2013-09-24
 6 4      Beca     31.           9. TRUE     2013-10-15
 7 4      Kimbe…   30.          10. TRUE     2013-10-22
 8 5      Enwez…   39.           2. TRUE     2014-08-13
# ... with 4 more variables: occupation <chr>,
#   hometown <chr>, star_baker <dbl>,
#   technical_winner <dbl>
```

Use glimpse

```
glimpse(bakers_mini)
```

```
Observations: 10
Variables: 10
$ series           <fct> 3, 3, 3, 4, 4, 4, 4, 5, 5, 5
$ baker            <chr> "Natasha", "Sarah-Jane", "Ca...
$ age              <dbl> 36, 28, 27, 38, 51, 31, 30, ...
$ num_episodes     <dbl> 1, 7, 8, 2, 6, 9, 10, 2, 3, 4
$ aired_us         <lgl> FALSE, FALSE, FALSE, TRUE, T...
$ last_date_uk     <date> 2012-08-14, 2012-09-25, 201...
$ occupation       <chr> "Midwife", "Vicar's wife", "...
$ hometown         <chr> "Tamworth, Staffordshire", "...
$ star_baker       <dbl> 0, 0, 0, 0, 0, 0, 2, 0, 0, 0
$ technical_winner <dbl> 0, 1, 1, 0, 0, 1, 3, 0, 0, 0
```

Use skim

```
library(skimr)
skim(bakers_mini)
```

```
Skim summary statistics
 n obs: 10
 n variables: 10

Variable type: character
    variable missing complete  n min max empty n_unique
1      baker       0       10 10   4  10     0       10
2   hometown       0       10 10   6  26     0       10
3 occupation       0       10 10   7  28     0       10
```

Skim date, factor, and logical variables

```
skim(bakers_mini)
```

```
Variable type: Date
      variable missing complete  n        min        max     median n_unique
1 last_date_uk       0       10 10 2012-08-14 2014-08-27 2013-10-04       10
```

```
Variable type: factor
  variable missing complete  n n_unique             top_counts ordered
1   series       0       10 10        3 4: 4, 3: 3, 5: 3, 1: 0 FALSE
```

```
Variable type: logical
  variable missing complete  n mean                count
1 aired_us       0       10 10  0.7 TRU: 7, FAL: 3, NA: 0
```

Skim numeric variables

```
skim(bakers_mini)
```

```
Variable type: numeric
          variable missing complete  n mean   sd min   p25 median   p75 max
1              age       0       10 10 34.3 7.12  27 30.25   31.5 37.5   51
2     num_episodes       0       10 10  5.2 3.22   1  2.25    5    7.75  10
3       star_baker       0       10 10  0.2 0.63   0  0       0    0      2
4 technical_winner       0       10 10  0.6 0.97   0  0       0    1      3
      hist
1 ????????
2 ????????
3 ????????
4 ????????
```

Let's get to work!

WORKING WITH DATA IN THE TIDYVERSE

Count With Your Data

WORKING WITH DATA IN THE TIDYVERSE

Alison Hill
Professor & Data Scientist

All the bakers

```
glimpse(bakers)
```

```
Observations: 95
Variables: 10
$ series           <fct> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
$ baker            <chr> "Lea", "Mark", "Annetha", "L...
$ age              <dbl> 51, 48, 30, 44, 25, 31, 45, ...
$ num_episodes     <dbl> 1, 1, 2, 2, 3, 4, 5, 6, 6, 6...
$ aired_us         <lgl> FALSE, FALSE, FALSE, FALSE, ...
$ last_date_uk     <date> 2010-08-17, 2010-08-17, 201...
$ occupation       <chr> "Retired", "Bus Driver", "Si...
$ hometown         <chr> "Midlothian, Scotland", "Sou...
$ star_baker       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
$ technical_winner <dbl> 0, 0, 0, 0, 1, 0, 0, 2, 2, 0...
```

Distinct series

```
bakers %>%
  distinct(series)
```

```
# A tibble: 8 x 1
  series
  <fct>
1 1
2 2
3 3
4 4
5 5
6 6
7 7
8 8
```

Count rows by one variable

```
bakers %>%
  count(series)
```

```
# A tibble: 8 x 2
  series     n
  <fct>  <int>
1 1         10
2 2         12
3 3         12
4 4         13
5 5         12
6 6         12
7 7         12
8 8         12
```

Count does group_by and summarize for you

```
bakers %>%
  count(series)
```

```
# A tibble: 8 x 2
  series     n
  <fct>  <int>
1 1         10
2 2         12
3 3         12
4 4         13
5 5         12
6 6         12
7 7         12
8 8         12
```

```
bakers %>%
  group_by(series) %>%
  summarize(n = n())
```

```
# A tibble: 8 x 2
  series     n
  <fct>  <int>
1 1         10
2 2         12
3 3         12
4 4         13
5 5         12
6 6         12
7 7         12
8 8         12
```

Count rows by two variables

```
bakers %>%
  count(aired_us, series)
```

```
# A tibble: 8 x 3
  aired_us series     n
  <lgl>    <fct>  <int>
1 FALSE    1         10
2 FALSE    2         12
3 FALSE    3         12
4 FALSE    8         12
5 TRUE     4         13
6 TRUE     5         12
7 TRUE     6         12
8 TRUE     7         12
```

Count also ungroups for you

```
bakers %>%
  count(aired_us, series) %>%
  mutate(prop_bakers = n/sum(n))
```

```
# A tibble: 8 x 4
  aired_us series     n prop_bakers
  <lgl>    <fct>  <int>       <dbl>
1 FALSE    1         10       0.105
2 FALSE    2         12       0.126
3 FALSE    3         12       0.126
4 FALSE    8         12       0.126
5 TRUE     4         13       0.137
6 TRUE     5         12       0.126
7 TRUE     6         12       0.126
8 TRUE     7         12       0.126
```

```
bakers %>%
  group_by(aired_us, series) %>%
  summarize(n = n()) %>%
  mutate(prop_bakers = n/sum(n))
```

```
# A tibble: 8 x 4
# Groups:   aired_us [2]
  aired_us series     n prop_bakers
  <lgl>    <fct>  <int>       <dbl>
1 FALSE    1         10       0.217
2 FALSE    2         12       0.261
3 FALSE    3         12       0.261
4 FALSE    8         12       0.261
5 TRUE     4         13       0.265
6 TRUE     5         12       0.245
7 TRUE     6         12       0.245
8 TRUE     7         12       0.245
```

Count also ungroups for you

```
bakers %>%
  count(aired_us, series) %>%
  mutate(prop_bakers = n/sum(n))
```

```
# A tibble: 8 x 4
  aired_us series     n prop_bakers
  <lgl>    <fct>  <int>       <dbl>
1 FALSE    1         10       0.105
2 FALSE    2         12       0.126
3 FALSE    3         12       0.126
4 FALSE    8         12       0.126
5 TRUE     4         13       0.137
6 TRUE     5         12       0.126
7 TRUE     6         12       0.126
8 TRUE     7         12       0.126
```

```
bakers %>%
  group_by(aired_us, series) %>%
  summarize(n = n()) %>%
  ungroup() %>%
  mutate(prop_bakers = n/sum(n))
```

```
# A tibble: 8 x 4
  aired_us series     n prop_bakers
  <lgl>    <fct>  <int>       <dbl>
1 FALSE    1         10       0.105
2 FALSE    2         12       0.126
3 FALSE    3         12       0.126
4 FALSE    8         12       0.126
5 TRUE     4         13       0.137
6 TRUE     5         12       0.126
7 TRUE     6         12       0.126
8 TRUE     7         12       0.126
```

Count to roll up a level

```
bakers %>%
  count(aired_us, series)
```

```
# A tibble: 8 x 3
  aired_us series     n
  <lgl>    <fct>  <int>
1 FALSE    1         10
2 FALSE    2         12
3 FALSE    3         12
4 FALSE    8         12
5 TRUE     4         13
6 TRUE     5         12
7 TRUE     6         12
8 TRUE     7         12
```

```
bakers %>%
  count(aired_us, series) %>%
  count(aired_us)
```

```
# A tibble: 2 x 2
  aired_us    nn
  <lgl>    <int>
1 FALSE        4
2 TRUE         4
```

Let's get to work!

WORKING WITH DATA IN THE TIDYVERSE