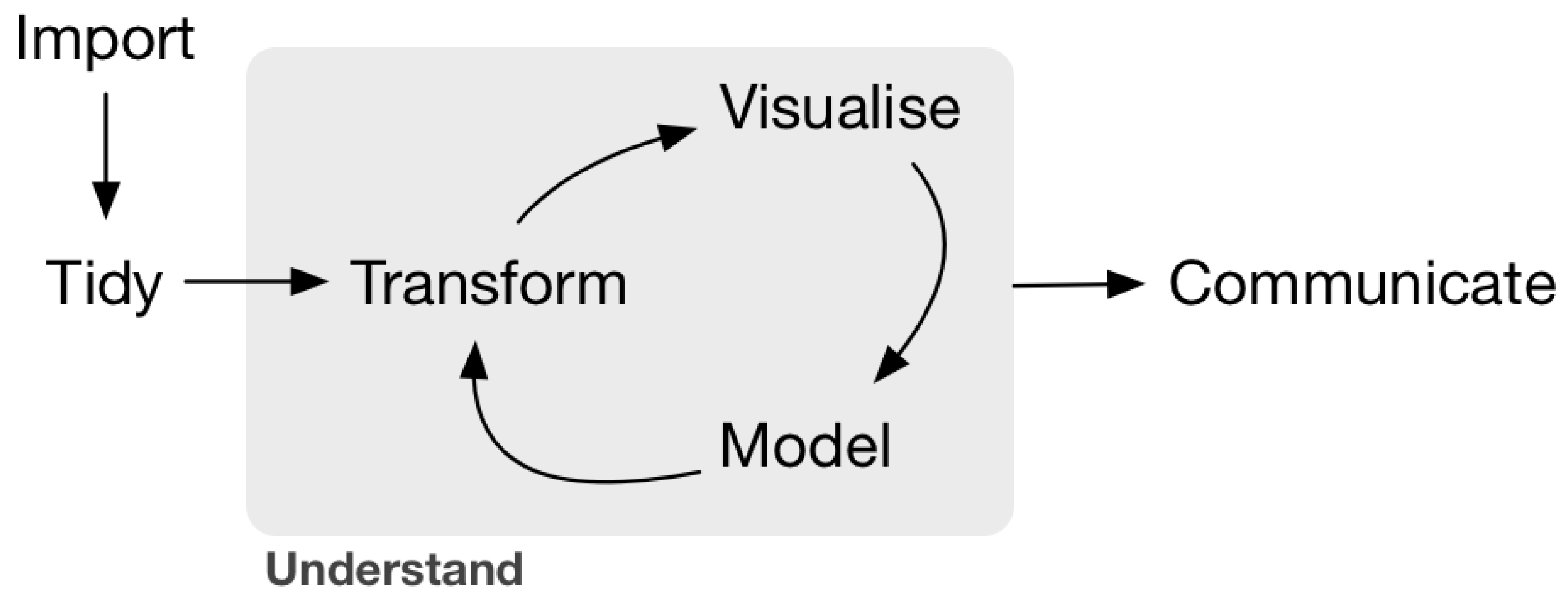
R Programming For Natural Resource Professionals

Lecture 3: Introduction to the Tidyverse

Previewing the week

https://jaredhomola.github.io/RforNatRes/

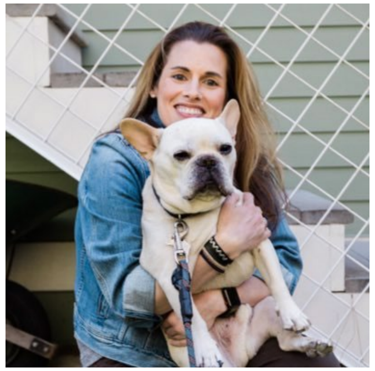
Tidyverse: What?



Tidyverse: Who?

Hadley Wickham
Chief Scientist at R Studio
Lead Developer of Tidyverse
@hadleywickham

Member of Tidyverse Development Team



Mara Averick
@dataandme

Claus Wilke
@ClausWilke

Davis Vaughan
@dvaughan32

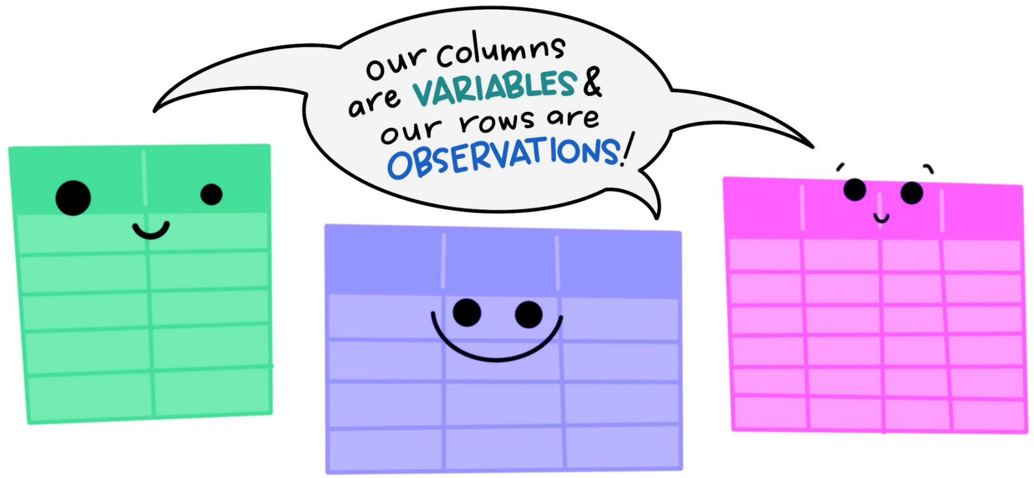
Romain François
@romain_francois

Winston Chang
@winston_chang

Others listed at https://github.com/orgs/tidyverse/people

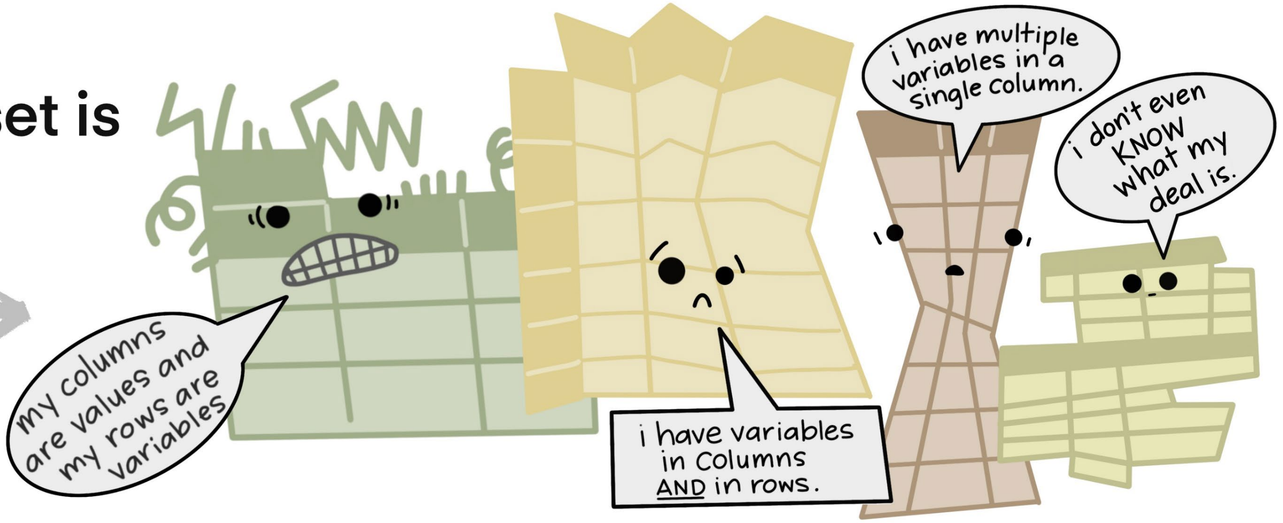
Tidyverse: Why?

The standard structure of tidy data means that "tidy datasets are all alike..."



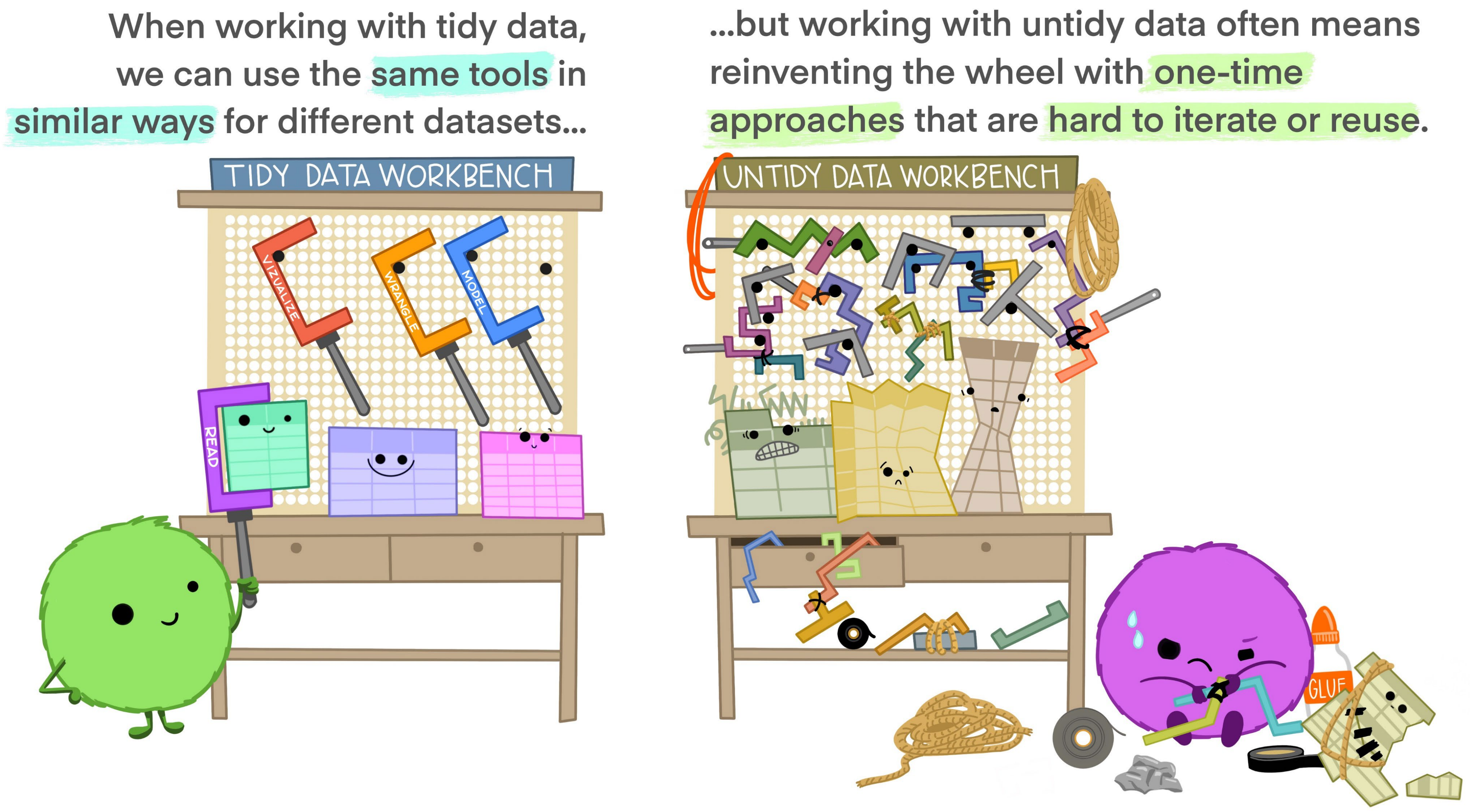
"...but every messy dataset is messy in its own way."

—HADLEY WICKHAM


my columns are values and my rows are variables
i have variables in columns AND in rows.
i have multiple variables in a single column.
i don't even KNOW what my deal is.

Allison Horst

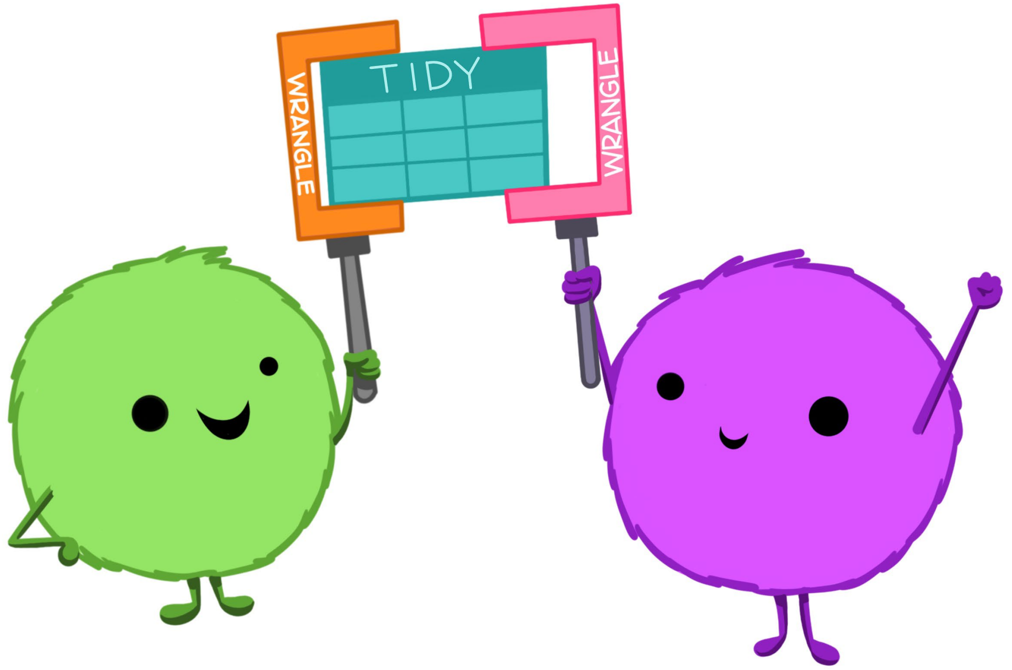
Tidyverse: Why?


When working with tidy data, we can use the same tools in similar ways for different datasets...
...but working with untidy data often means reinventing the wheel with one-time approaches that are hard to iterate or reuse.
TIDY DATA WORKBENCH
VIZUALIZE
WRANGLE
MODEL
READ
UNTIDY DATA WORKBENCH
GLUE

Allison Horst

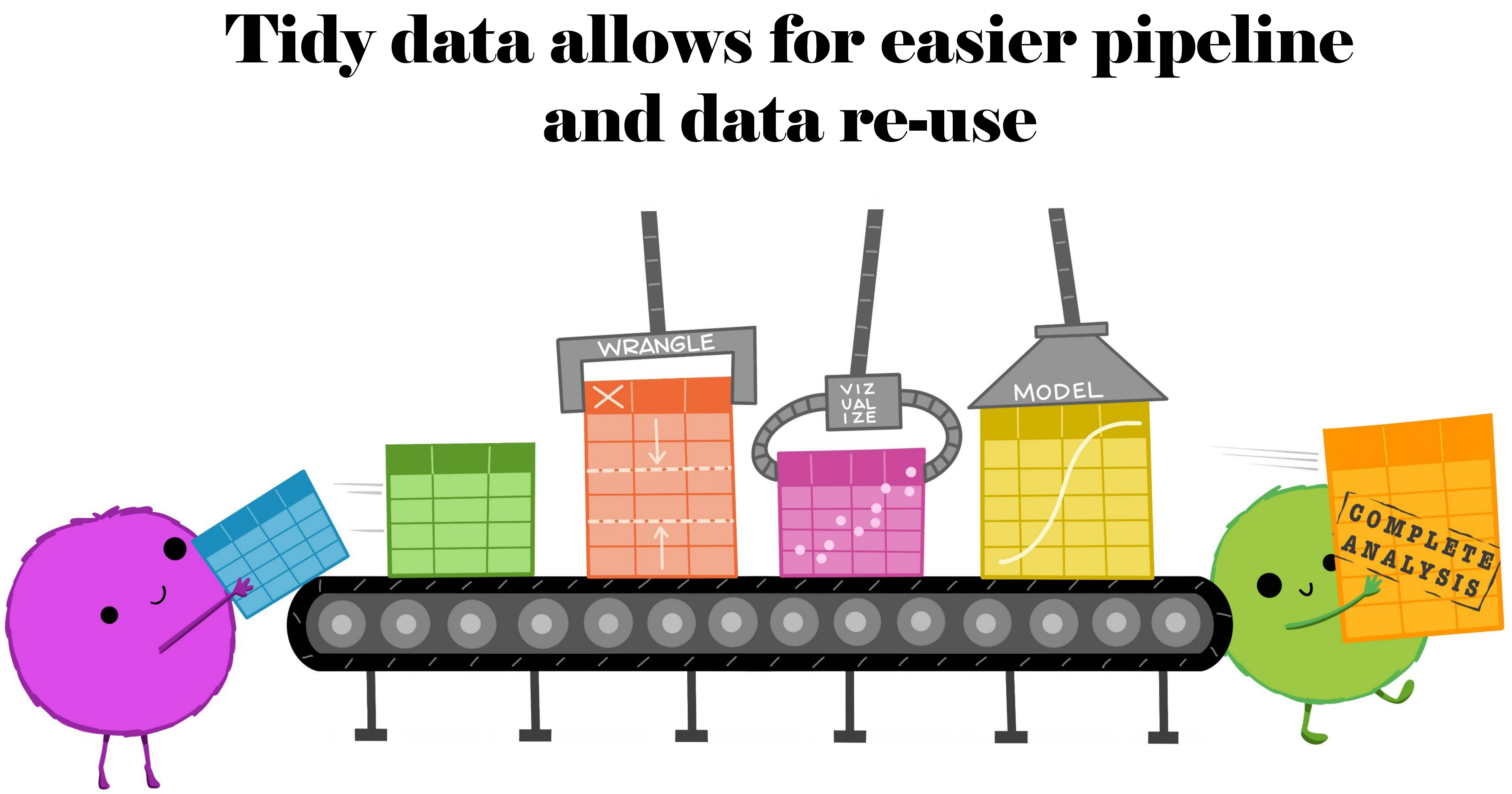
Tidyverse: Why?

Tidy data permits easier collaboration!



Allison Horst

Tidyverse: Why?



Allison Horst

Tidyverse: Why?

Tidy data allows for using a standardized set of packages

tidyverse
ggplot2
dplyr
tidyr
%>%
Ceci n'est pas un pipe.
forcats
stringr
lubridate
TIBBLE
readr
hms
broom
rsample
purrr
modelr
rmarkdown
knitr
bookdown
reprex

The Pipe Operator %>% and |>

```
Data |>
    Operation1 |>
    Operation2 |>
    Operation3
```

The pipe operator allows code products to be passed through a series of functions

- Fewer nested function calls
- Minimize creation of temporary variables
- Minimize overwriting of original data

Shortcut to create the pipe: ctrl+shift+m

Tidy coding syntax

Base R syntax:

```
aggregate(airquality[, “Ozone”], list(Month =
airquality[, “Month”]), mean)
```

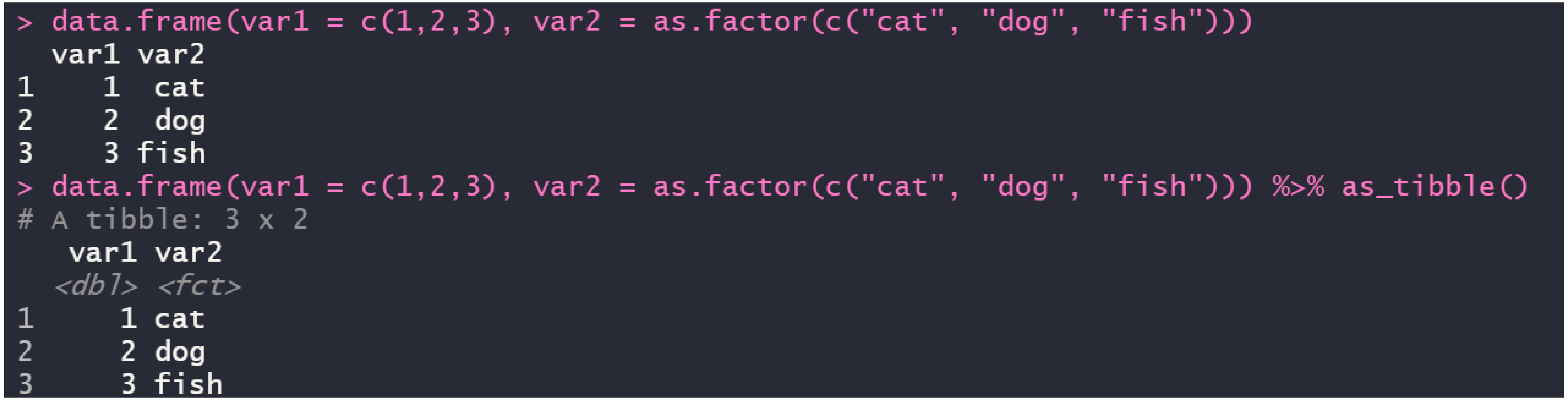
TidyR syntax:

```
airquality %>%
   group_by(Month) %>%
   summarize(mean_o3 = mean(Ozone))
```

Tibble: The Tidy data structure

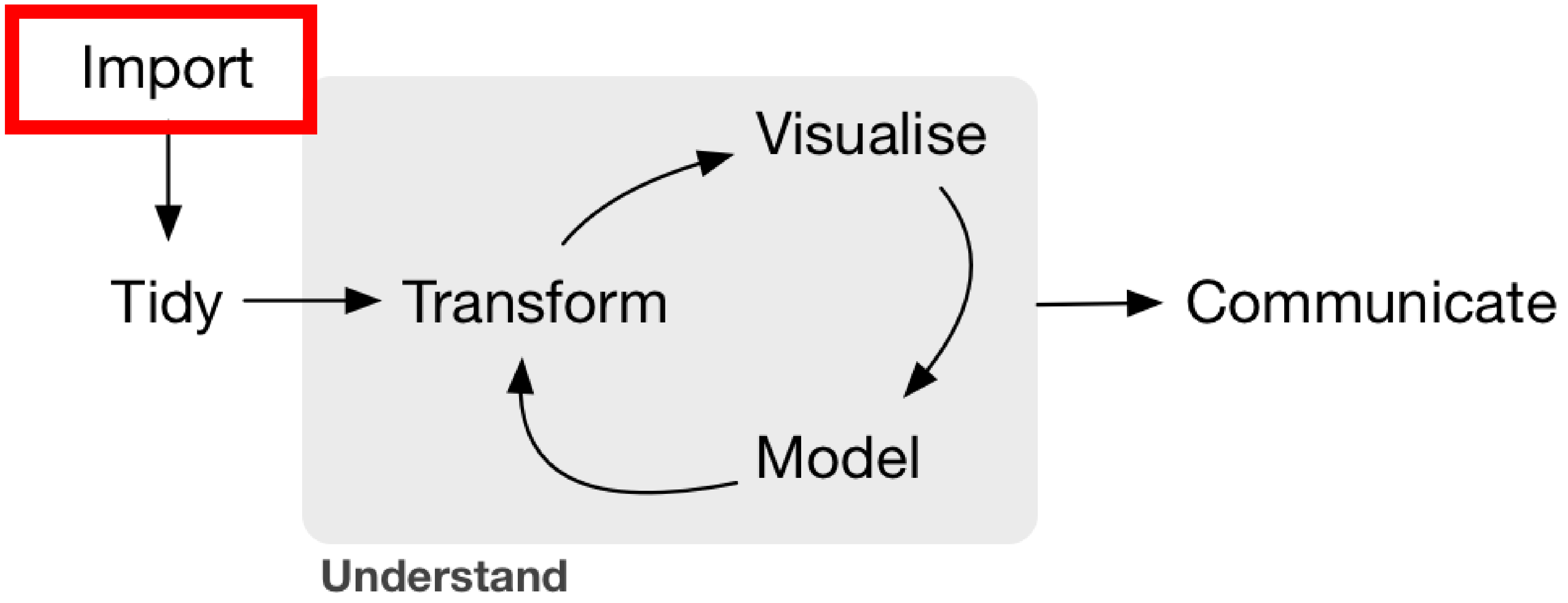
Tibble

- "A modern re-imagining of the data frame"
- Build-in ability to identify issues with variables (e.g., invalid names)
- Retain variable types better than data.frame()



```
> data.frame(var1 = c(1,2,3), var2 = as.factor(c("cat", "dog", "fish")))
  var1 var2
1    1  cat
2    2  dog
3    3 fish
> data.frame(var1 = c(1,2,3), var2 = as.factor(c("cat", "dog", "fish"))) %>% as_tibble()
# A tibble: 3 x 2
   var1 var2
  <dbl> <fct>
1     1 cat
2     2 dog
3     3 fish
```

Tidyverse


Import
Tidy
Transform
Visualise
Model
Communicate
Understand

Reading in non-Excel data

Read in rectangular data using `readr`

- Common data types are .txt and .csv

See R Studio ‘Import Data’ cheat sheet for more details

Read Tabular Data with readr

tidyr::read_*

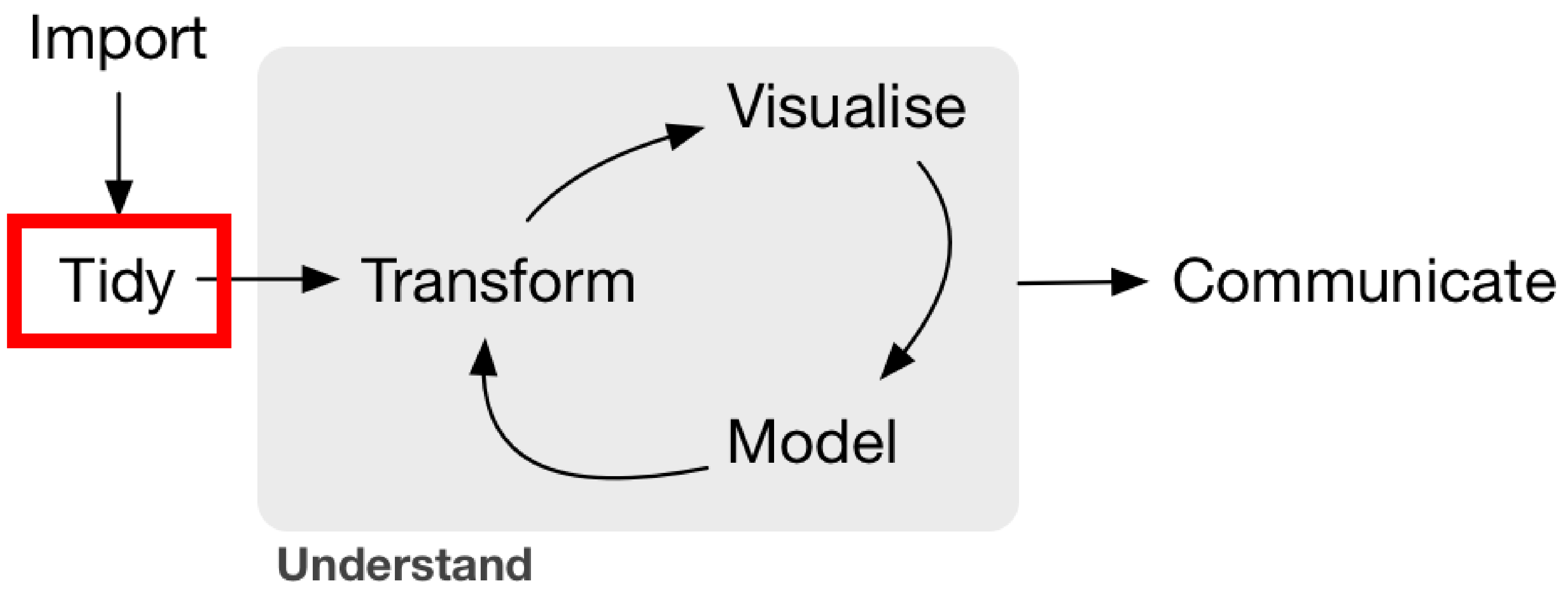
Reading in Excel data

Read in Microsoft Excel data using `readxl`

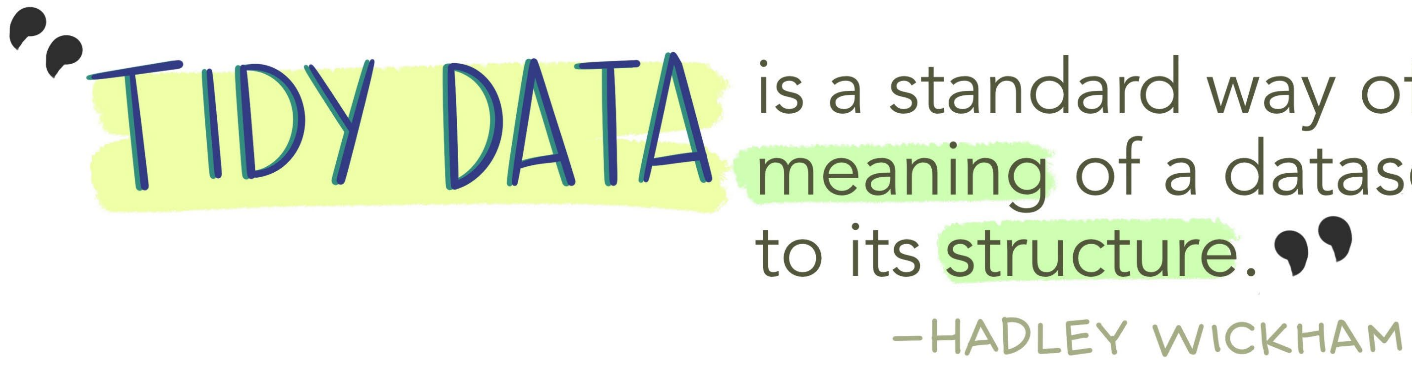
```
read_excel(filePath)
```

See R Studio 'Import Data' cheat sheet for more details

Tidyverse


Import
Tidy
Transform
Visualise
Model
Communicate
Understand

Tidy data structures

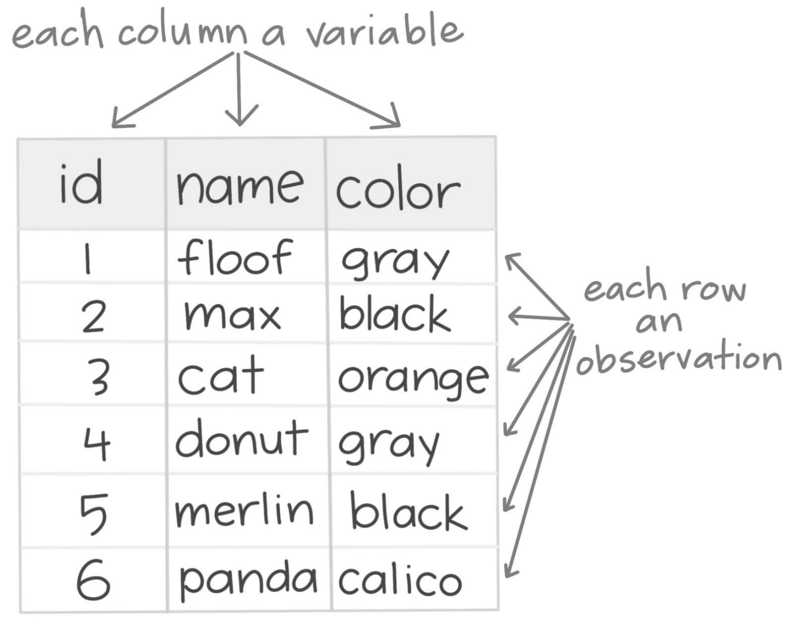


"TIDY DATA is a standard way of mapping the meaning of a dataset to its structure."

—HADLEY WICKHAM

In tidy data:

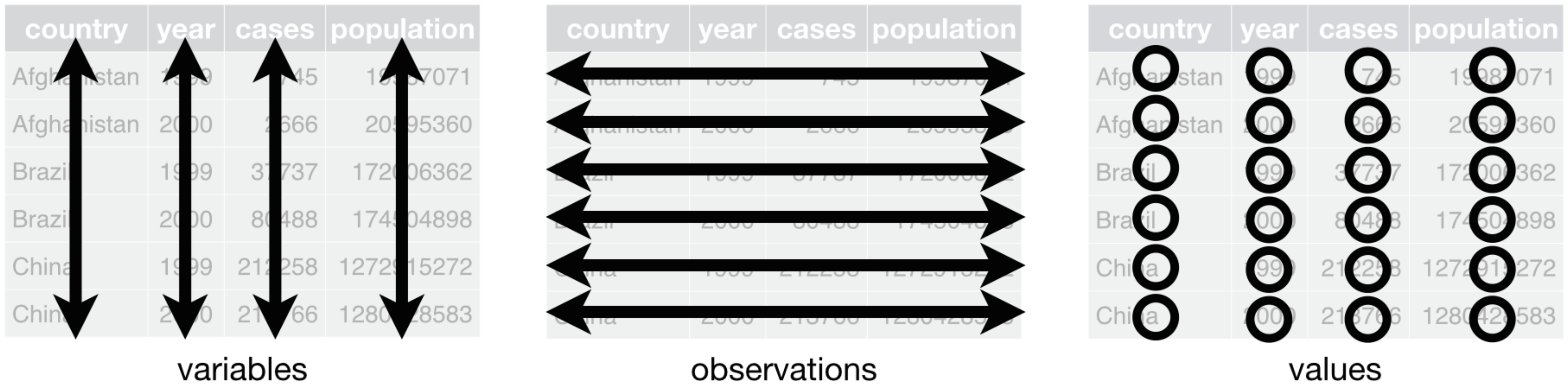
- each variable forms a column
- each observation forms a row
- each cell is a single measurement


each column a variable
each row an observation

id	name	color
1	floof	gray
2	max	black
3	cat	orange
4	donut	gray
5	merlin	black
6	panda	calico

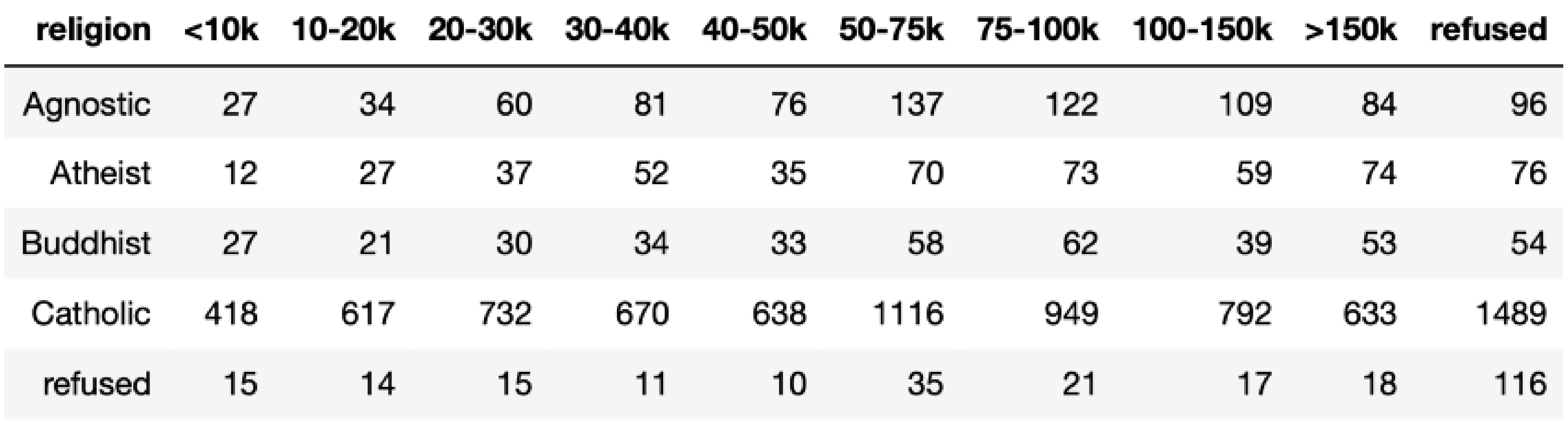
Allison Horst

Tidy data structures



Tidy data?

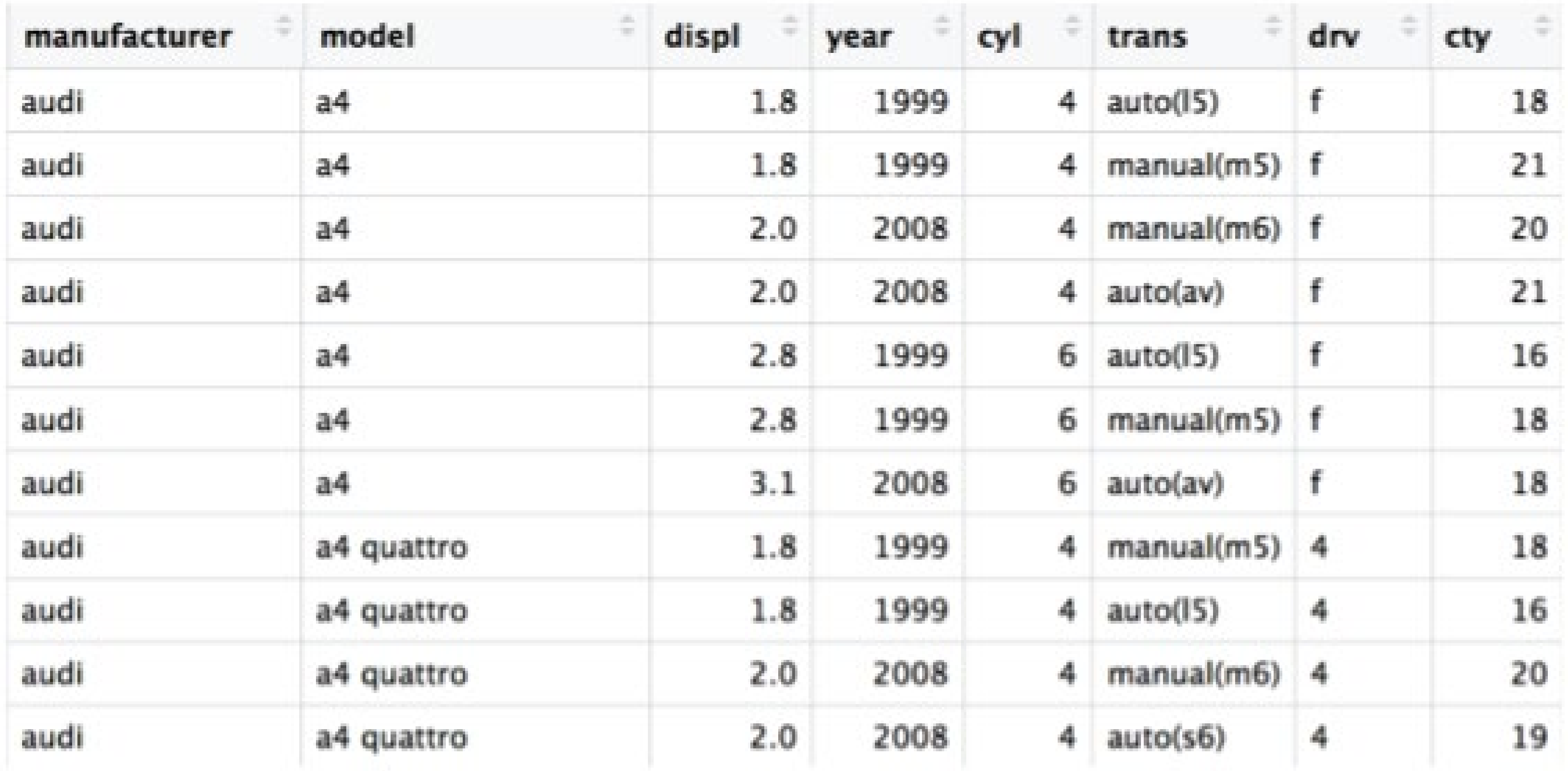
Data set: Religious affiliation for various income brackets.
Column headers: Income ranges



religion	<10k	10-20k	20-30k	30-40k	40-50k	50-75k	75-100k	100-150k	>150k	refused
Agnostic	27	34	60	81	76	137	122	109	84	96
Atheist	12	27	37	52	35	70	73	59	74	76
Buddhist	27	21	30	34	33	58	62	39	53	54
Catholic	418	617	732	670	638	1116	949	792	633	1489
refused	15	14	15	11	10	35	21	17	18	116

Tidy data?

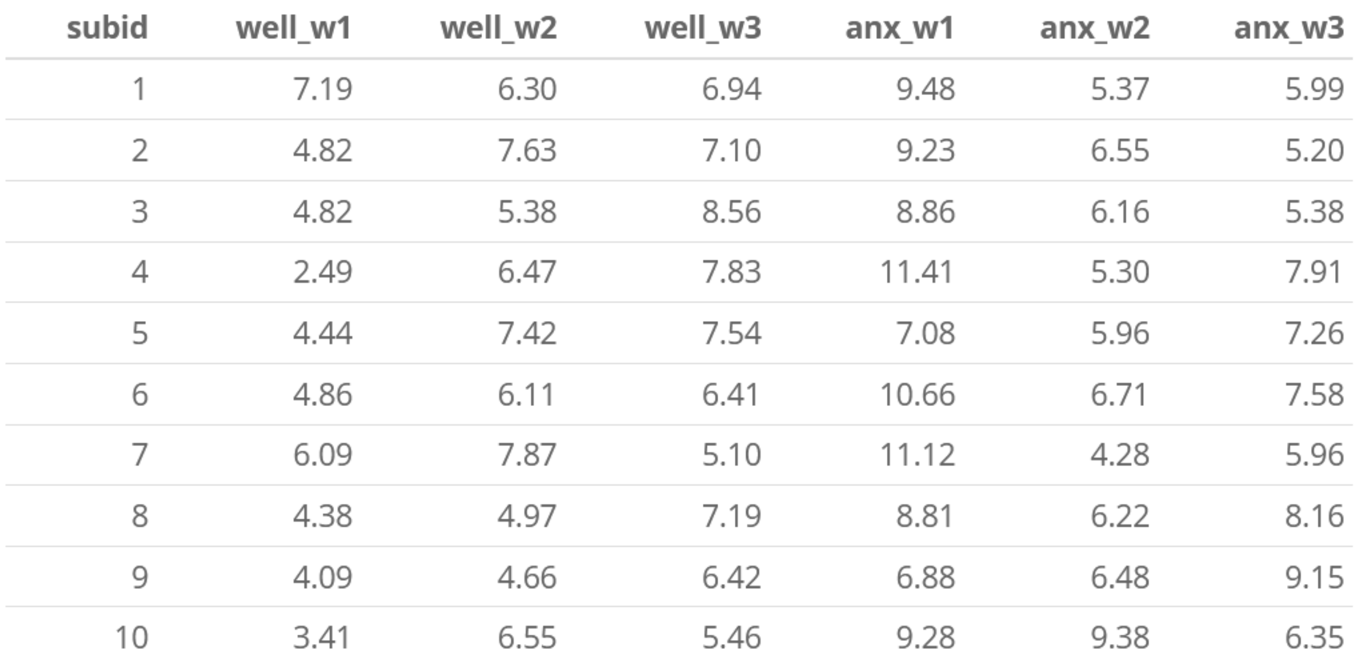
Data set: Characteristics of various cars.
Column headers: Car characteristics



manufacturer	model	displ	year	cyl	trans	drv	cty
audi	a4	1.8	1999	4	auto(l5)	f	18
audi	a4	1.8	1999	4	manual(m5)	f	21
audi	a4	2.0	2008	4	manual(m6)	f	20
audi	a4	2.0	2008	4	auto(av)	f	21
audi	a4	2.8	1999	6	auto(l5)	f	16
audi	a4	2.8	1999	6	manual(m5)	f	18
audi	a4	3.1	2008	6	auto(av)	f	18
audi	a4 quattro	1.8	1999	4	manual(m5)	4	18
audi	a4 quattro	1.8	1999	4	auto(l5)	4	16
audi	a4 quattro	2.0	2008	4	manual(m6)	4	20
audi	a4 quattro	2.0	2008	4	auto(s6)	4	19

Tidy data?

Data set: People asked to provide a “wellness” and “anxiety” score for each of 3 weeks.
Column headers: Wellness or anxiety paired with week of recording (w*)



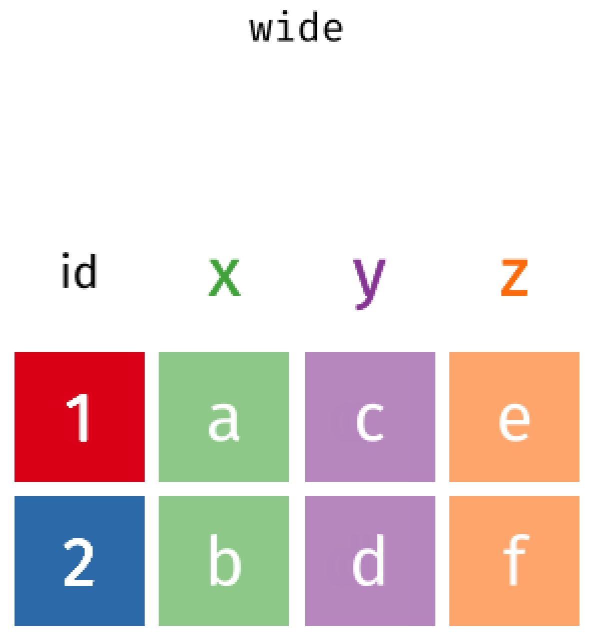
subid	well_w1	well_w2	well_w3	anx_w1	anx_w2	anx_w3
1	7.19	6.30	6.94	9.48	5.37	5.99
2	4.82	7.63	7.10	9.23	6.55	5.20
3	4.82	5.38	8.56	8.86	6.16	5.38
4	2.49	6.47	7.83	11.41	5.30	7.91
5	4.44	7.42	7.54	7.08	5.96	7.26
6	4.86	6.11	6.41	10.66	6.71	7.58
7	6.09	7.87	5.10	11.12	4.28	5.96
8	4.38	4.97	7.19	8.81	6.22	8.16
9	4.09	4.66	6.42	6.88	6.48	9.15
10	3.41	6.55	5.46	9.28	9.38	6.35

Set up for in class exercises

Tidying data

tidyr::pivot_longer()

- `cols` = Columns you'd like to pivot to longer format
- `names_to` = The name of a column to create that will contain the current column names.
- `values_to` = The name of the column to create where the data stored in cell values will move to.



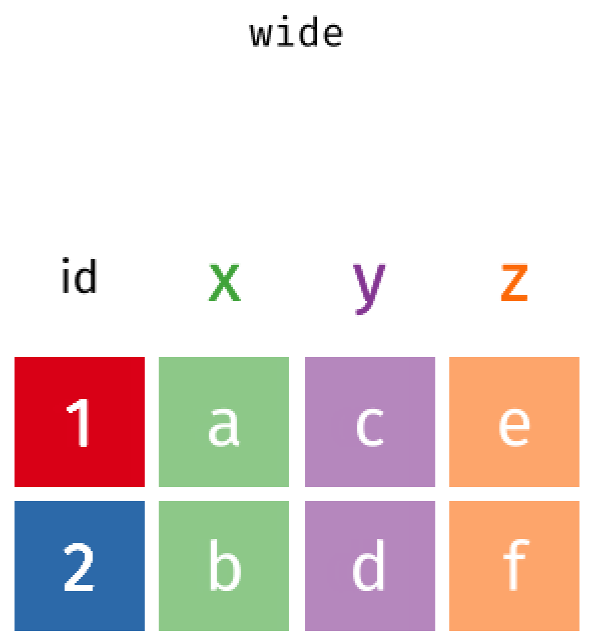
wide

id	x	y	z
1	a	c	e
2	b	d	f

Tidying data

tidyr::pivot_wider()

- `names_from` = Which column (or columns) containing the name of the output column (names_from).
- `values_from` = Which column (or columns) to get the cell values from (values_from).


wide

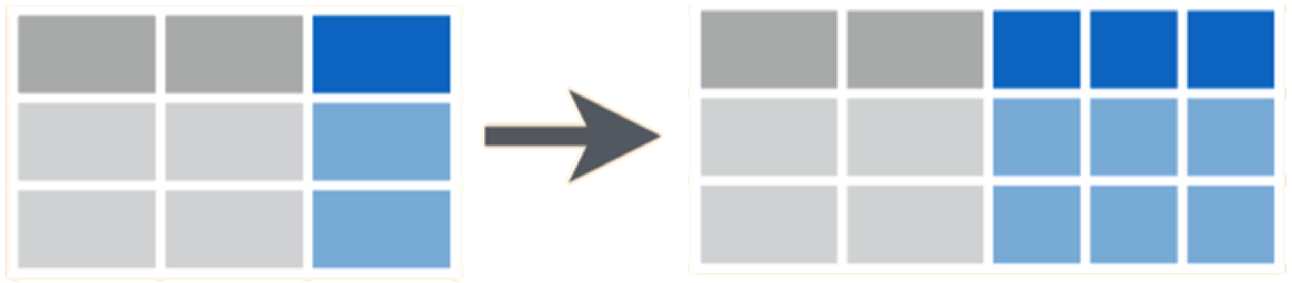
id	x	y	z
1	a	c	e
2	b	d	f

Tidying data

tidyr::separate()

Goal: Split a columns into multiple.

- `col` = The column to split.
- `into` = A vector of new column names to receive the split values.
- `sep` = The symbol that indicates where to split the values (delimiter).

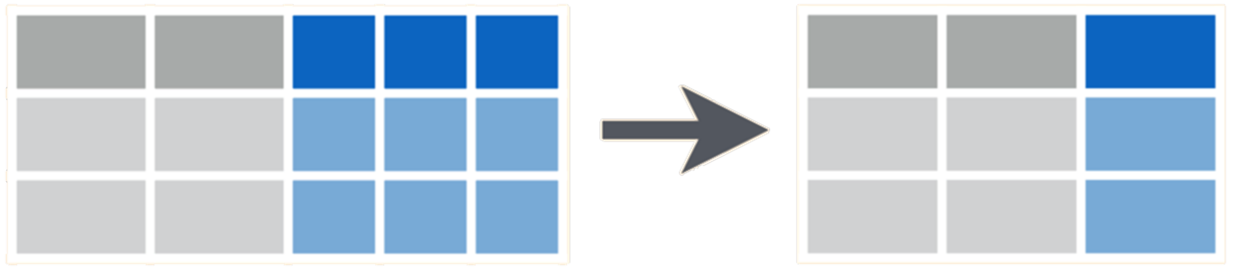


Tidying data

tidyr::unite()

Goal: Join multiple columns together.

- `col` = Name of new column.
- The columns to unite.
- `sep` = a separator, if desired



Tidying data

Dealing with missing data (NAs)

Option 1: `drop_na()`

- No column specification = Drop all rows with NAs anywhere
- Specifying a column = Drop rows with NA in that column

Option 2: `fill()`

- Replace the NA with an adjacent value `(.direction = “down”)`

Helper functions

Shortcuts for selecting variables with certain attributes

Format: `tibble %>% select(contains(“wk”))`

- `contains()`
- `ends_with()`
- `everything()`
- `matches()`
- `num_range()`
- `any_of()`
- `starts_with()`

Pair programming

- Driver: person typing on the keyboard
- Navigator: describes code to be entered; reviews it after being entered

- Driver should open the Thursday in class exercises R script on course website

Pair programming: Exercise 1

- Part 1: Data loading
- Load the data indicated using:
  - read_csv()
  - read.csv()
  - read_delim()
  - Examine the differences

Pair programming: Exercise 1

- Part 2: Start tidying the tibble by pivoting it
  - What function will we use?
  - Which columns to pivot?
  - What to name the new column that will contain current column <u>names</u>?
  - What to name the new column that will contain current column <u>values</u>?

- Pipe the data into the function. Do not assign any new variables.

Pair programming: Exercise 1

- Part 3: Separate column with combined variables
  - Which column needs to be split?
  - What should the new columns it splits into be named?

Pair programming: Exercise 2

- Part 1: Subset the data
  - Use dplyr::select() to subset the billboard tibble to include only artist, track, date.entered, wk1, wk2, wk3, wk4, and wk5 variables

- Part 2: Drop observations that contain an NA in any column

Pair programming: Exercise 2

- Part 3: Tidy up the data set by switching to a long format
  - Use a helper function to achieve this

- Part 4: Separate the date.entered variable into separate columns for year, month, and day
  - Add an argument to keep the original column

kableExtra

```r
head(mtcars)
```

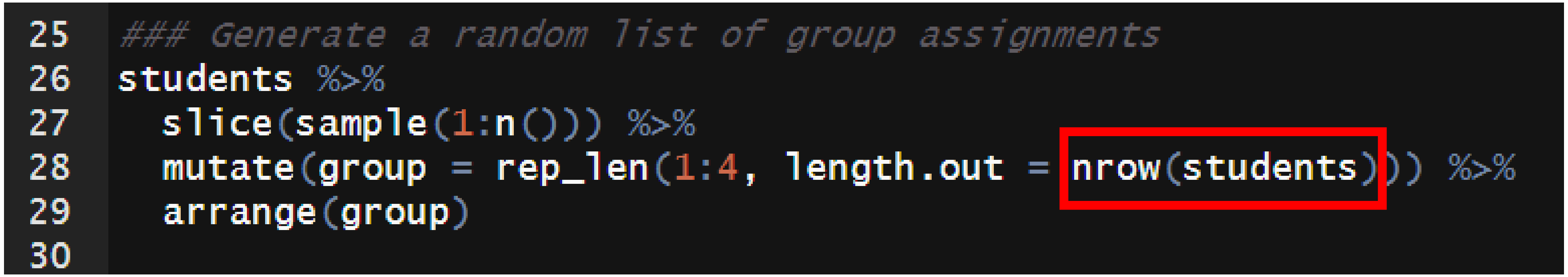
```
##                    mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

```r
kbl(head(mtcars)) %>%
  kable_styling()
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	3	1

Programmatic coding

- When possible, use code to state your arguments within functions



```
### Generate a random list of group assignments
students %>%
  slice(sample(1:n())) %>%
  mutate(group = rep_len(1:4, length.out = nrow(students))) %>%
  arrange(group)

```

Tidy pipelines

- Always pass an entire tibble into tidyverse pipelines

```
tibble |>
    Function1() |>
    Function2() |> …
```