

# R Programming For Natural Resource Professionals



```
library(tidyverse)
```



## Lecture 3: Introduction to the Tidyverse

# Paper discussions

Log in and pull up the course website:

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

Discussion Google Doc:

<https://bit.ly/3GVPwuV>

Assign groups

# Previewing the week

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

# Revised lecture strategy

## More emphasis on guided coding

- You still will not receive all answers for the homework

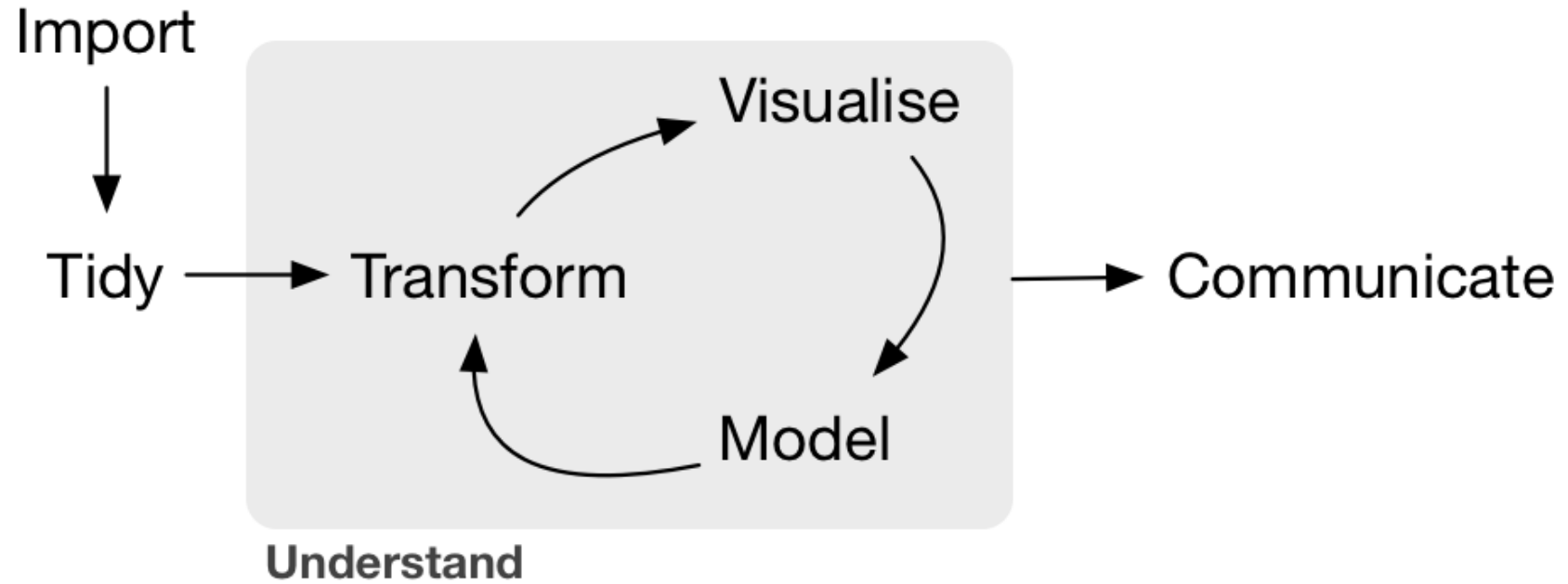
## Clear statement of learning objectives

- Don't feel like you've met them? Reach out.

# This week's learning objectives

1. Understand the idea behind the Tidyverse coding approach
2. Differentiate “tidy” and “messy” data
3. Develop competency with tools to tidy up messy data

# Tidyverse: What?



# Tidyverse: Who?

## Member of Tidyverse Development Team



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



Mara Averick  
@dataandme



Claus Wilke  
@ClausWilke



Davis Vaughan  
@dvaughan32



Romain François  
@romain\_francois

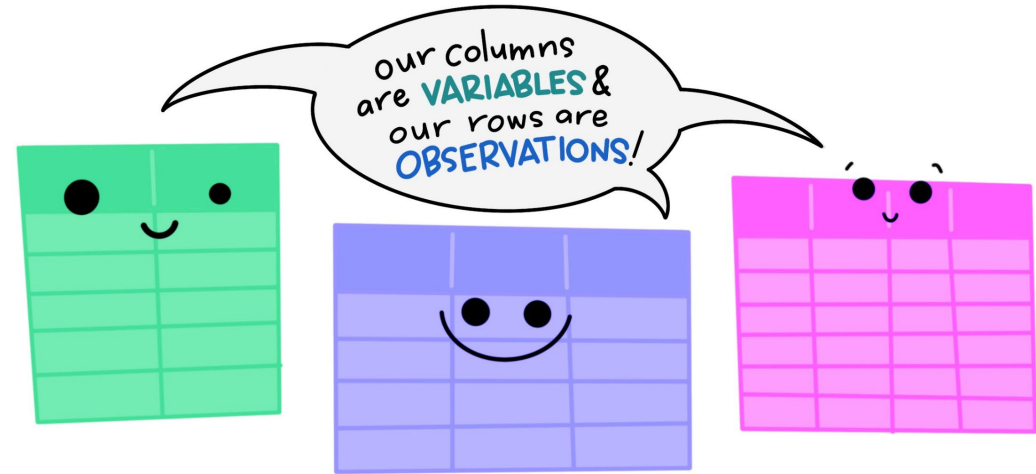


Winston Chang  
@winston\_chang

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

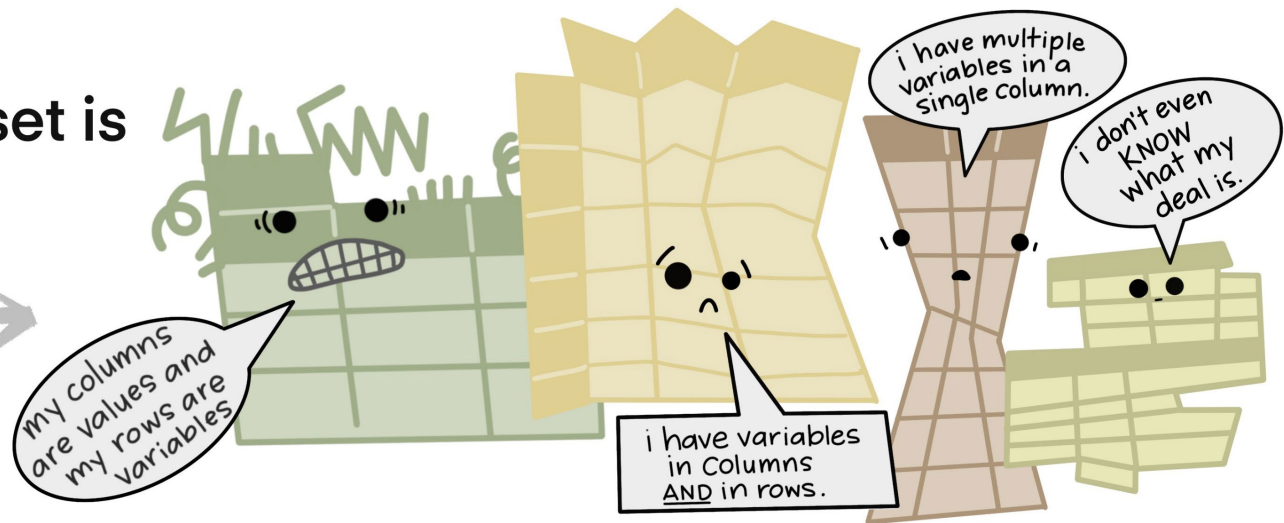
# Tidyverse: Why?

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



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

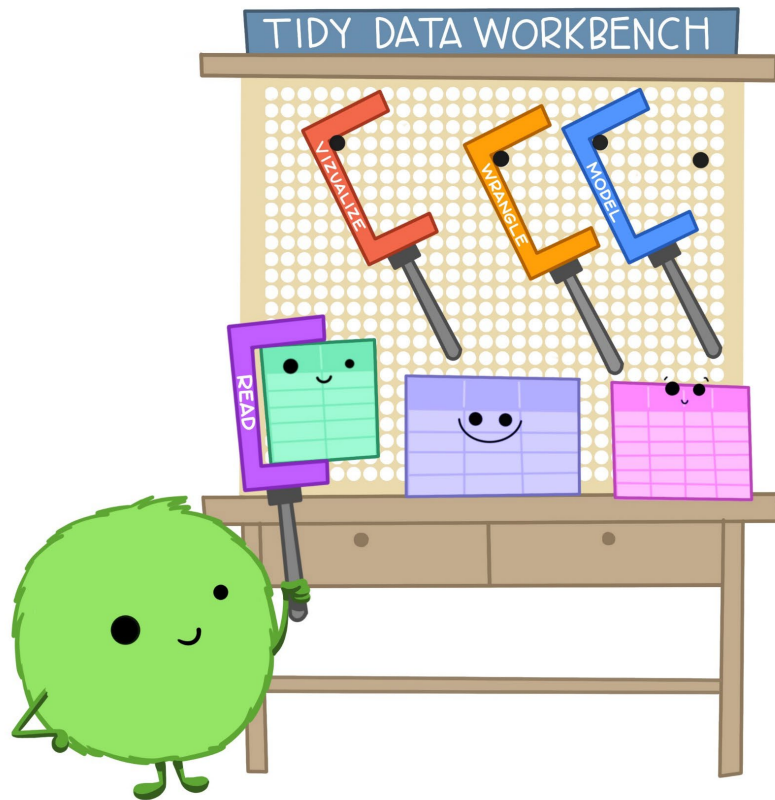
—HADLEY WICKHAM



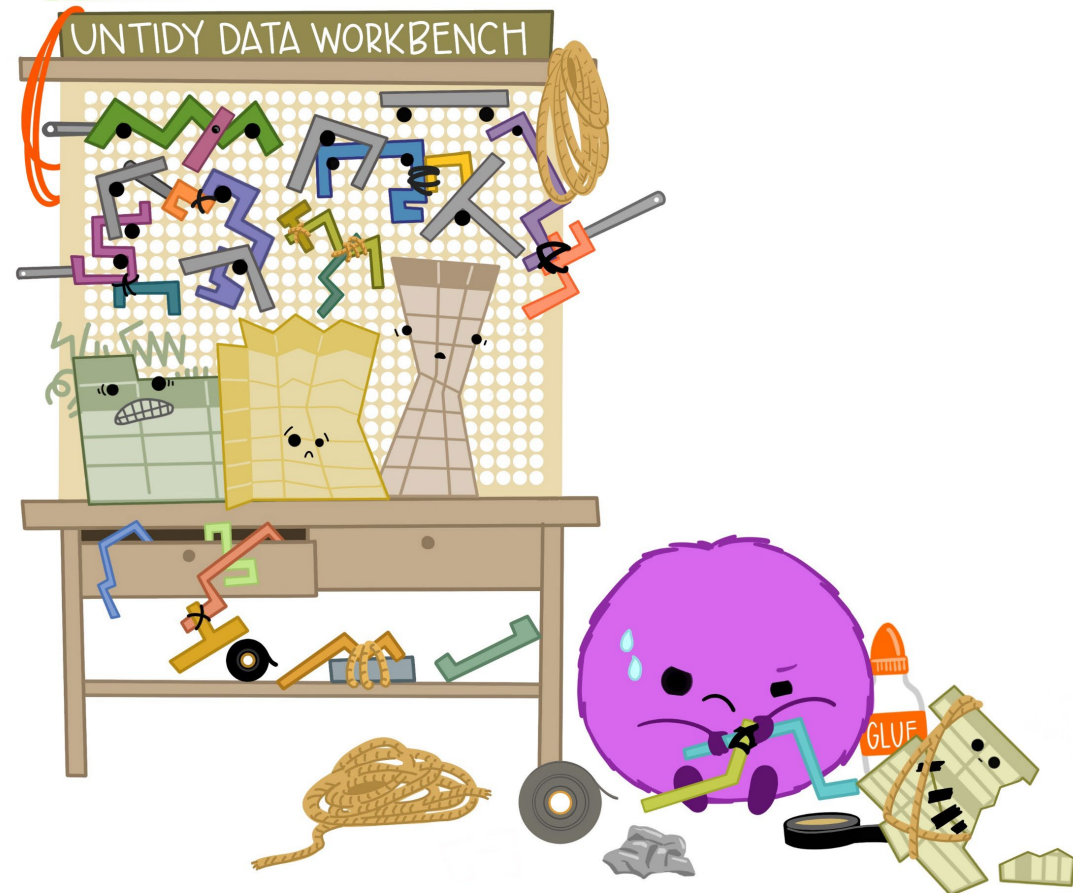


# Tidyverse: Why?

When working with tidy data, we can use the same tools in similar ways for different datasets...

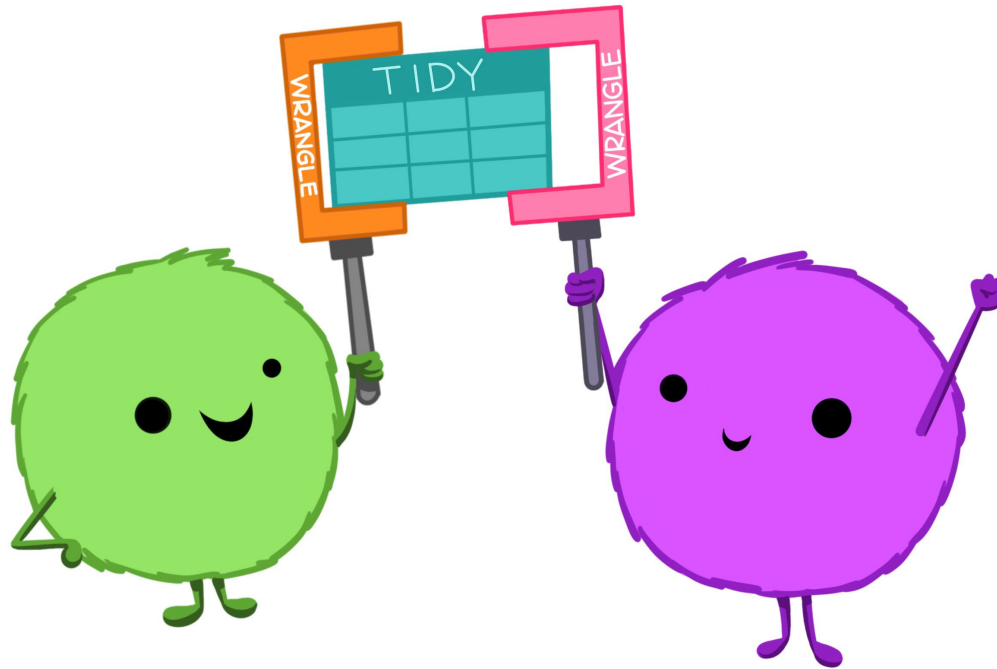


...but working with untidy data often means reinventing the wheel with one-time approaches that are hard to iterate or reuse.



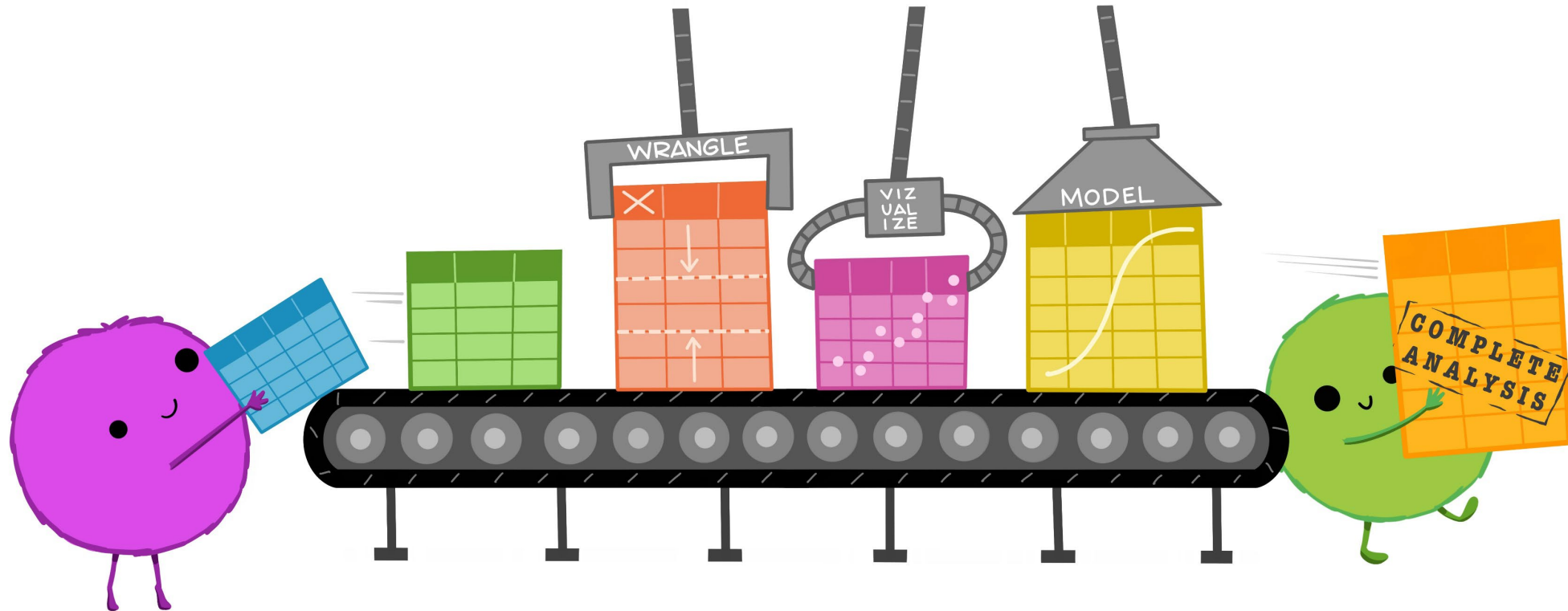
# Tidyverse: Why?

**Tidy data permits easier  
collaboration!**



# Tidyverse: Why?

**Tidy data allows for easier pipeline  
and data re-use**



# The Pipe Operator %>%

```
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



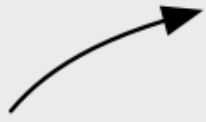
Import



Tidy



Transform



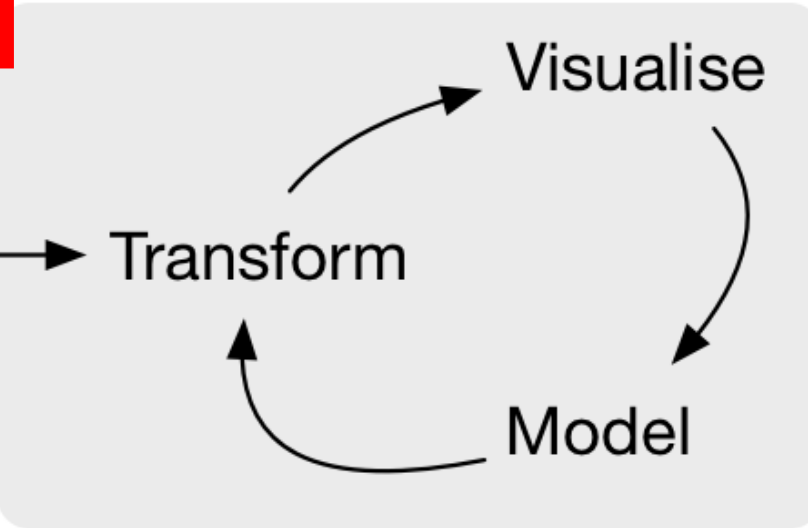
Visualise



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

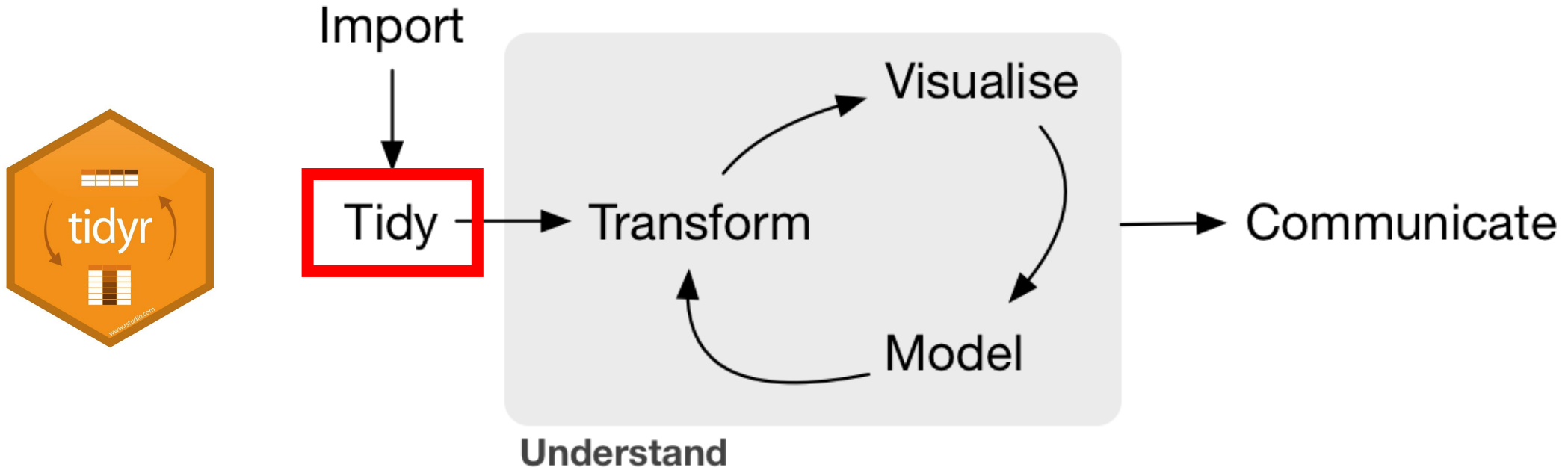
Read in Microsoft Excel data using `readxl`

```
read_excel(filePath)
```

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



# Tidyverse



# 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



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

each row  
an  
observation



# Tidy data structures

country	year	cases	population
Afghanistan	1999	2666	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	213766	1280425583

variables

country	year	cases	population
Afghanistan	1999	2666	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	213766	1280425583

observations

country	year	cases	population
Afghanistan	99	745	19987071
Afghanistan	00	666	20595360
Brazil	99	737	172006362
Brazil	00	488	174504898
China	99	2258	1272915272
China	00	3766	1280425583

values

# 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

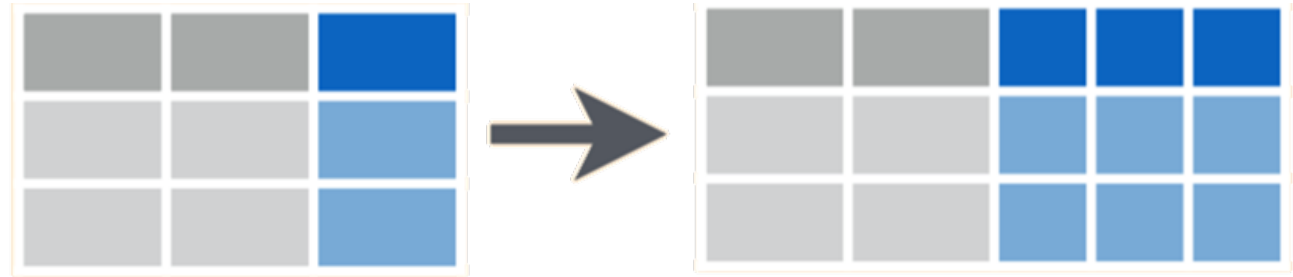
id	x	y	z
1	a	c	e
2	b	d	f

# 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).

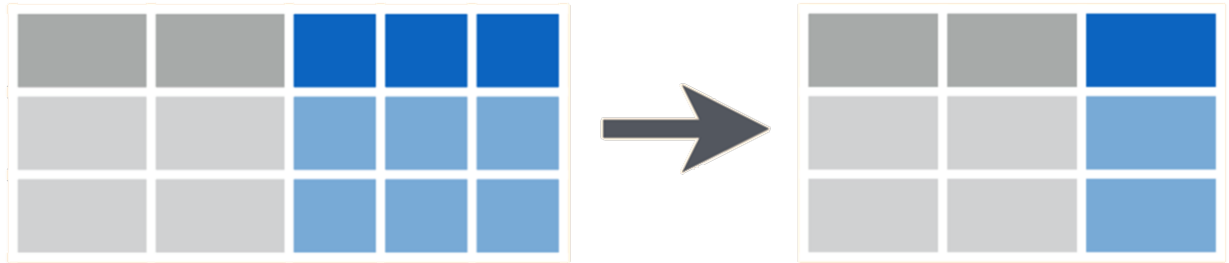


# Tidying data

## **`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("."))`

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