

data_wrangling() && ("manipulation" %in% R)



%>%



%>%



%>%



```
postgraduate_workshop(  
  dept = "Biological Sciences",  > face()  > logos()  
  presenter = c(  
    "Ruan van Mazijk",  
    "MSc candidate"  
)  
)
```



iCWild

Institute for Communities and Wildlife in Africa

> introduce()



> introduce()



- BSc + Hons here at UCT

> introduce()



- BSc + Hons here at UCT
- Ecology & evolution
- (Mostly plant) comparative biology
- Biogeography

> introduce()



- BSc + Hons here at UCT
- Ecology & evolution
- (Mostly plant) comparative biology
- Biogeography
- Been working with R for 4½ years
 - Every major project I've done...

> introduce()



Tetaria ustulata
Marloth NR
R. van Mazijk 2018



Tetraria thermalis
Silvermine, Table Mountain NP
R. van Mazijk 2018



Schoenus compar
Silvermine, Table Mountain NP
R. van Mazijk 2018

```
> workshop$goals
```

> workshop\$goals

- More reproducible science

> workshop\$goals

- More reproducible science
- Save time by:
 - Automating repetitive tasks
 - Eliminating human error

> workshop\$goals

- More reproducible science
- Save time by:
 - Automating repetitive tasks
 - Eliminating human error
- Boost your skills
- Think about your data programmatically

Notes & slides will go up here:

tinyurl.com/r-with-ruan

(But I encourage you to make your own notes!)

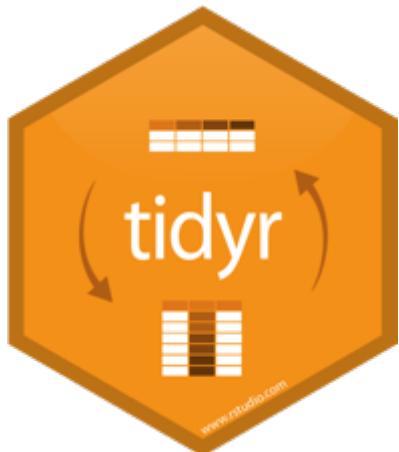
```
> workshop$outline
```

```
> workshop$outline[1:3]
```

```
> workshop$outline[1:3]
```

DAY 1

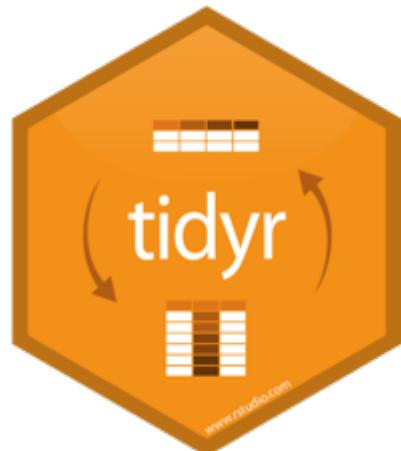
Tidy data principles
& `tidyr`



> workshop\$outline[1:3]

DAY 1

Tidy data principles
& `tidyverse`



DAY 2

Manipulating data
& an intro to `dplyr`



DAY 3

Extending your data
with `mutate()`,
`summarise()`
& friends

```
> workshop$outline[-(1:3)]
```

```
> workshop$outline[-(1:3)]
```

2 dialects of R:

```
> workshop$outline[-(1:3)]
```

2 dialects of R:

base

\$ [] [[]]

apply() which() subset()

```
> workshop$outline[-(1:3)]
```

2 dialects of R:

base

\$ [] [[]]

apply() which() subset()

tidyverse



```
data <- read.csv("my-data.csv")
```



```
data <- read.csv("my-data.csv")
```

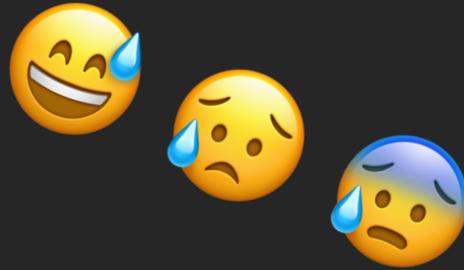
```
data1 <- f(data, arg1 = "something")
```



```
data <- read.csv("my-data.csv")
```

```
data1 <- f(data, arg1 = "something")
```

```
data2 <- g(data1, another.thing = "blah")
```



```
data <- read.csv("my-data.csv")
```

```
data1 <- f(data, arg1 = "something")
```

```
data2 <- g(data1, another.thing = "blah")
```

```
data3 <- h(data2, a.setting = TRUE)
```



```
data <- read.csv("my-data.csv")  
  
data1 <- f(data, arg1 = "something")  
data2 <- g(data1, another.thing = "blah")  
data3 <- h(data2, a.setting = TRUE)  
data4 <- data3[data3$a.column == "cough", ]
```



```
data <- read.csv("my-data.csv")  
  
data1 <- f(data, arg1 = "something")  
data2 <- g(data1, another.thing = "blah")  
data3 <- h(data2, a.setting = TRUE)  
data4 <- data3[data3$a.column == "cough", ]
```

```
data <- read.csv("my-data.csv")
```

```
data <- read.csv("my-data.csv")
```

```
data <-
```

```
data
```

```
data <- read.csv("my-data.csv")  
data <-  
f(data, arg1 = "something")
```

```
data <- read.csv("my-data.csv")  
  
data <-  
  
g(  
  f(data, arg1 = "something"),  
  another.thing = "blah"  
)
```

```
data <- read.csv("my-data.csv")

data <-
h(
g(
  f(data, arg1 = "something"),
  another.thing = "blah"
),
a.setting = TRUE
)
```



```
data <- read.csv("my-data.csv")  
  
data <-  
  h(  
    g(  
      f(data, arg1 = "something"),  
      another.thing = "blah"  
    ),  
    a.setting = TRUE  
)
```



```
data <- read.csv("my-data.csv")
```

```
data <-
h(
  g( ~~~~~
    f(data, arg1 = "something"),
    another.thing = "blah"
  ),
  a.setting = TRUE
)
```



```
data <- read.csv("my-data.csv")
```

```
data <-
h(
  g(
    f(data, arg1 = "something"),
    another.thing = "blah"
  ),
  a.setting = TRUE
)
```



```
data <- read.csv("my-data.csv")
```

```
data <-
h(
  g(
    f(data, arg1 = "something"),
    another.thing = "blah"
  ),
  a.setting = TRUE
)
```



```
data <- read.csv("my-data.csv")
```

```
data <-  
  h(  
    g(  
      f(data, arg1 = "something"),  
      another.thing = "blah"  
    ),  
    a.setting = TRUE  
)
```

data <- data[data\$a.column == "cough",]

&\$!#%

Solution: the pipe!

% > %

Solution: the pipe!

% > %

{ } [] [[]] <- = () , " " . . .

Read: “then”

Solution: the pipe!

% > %

{ } [] [[]] <- = () , " " . . .



```
data <- read.csv("my-data.csv")  
  
data1 <- f(data, arg1 = "something")  
data2 <- g(data1, another.thing = "blah")  
data3 <- h(data2, a.setting = TRUE)  
data4 <- data3[data3$a.column == "cough", ]
```

data

data



$f()$



$g()$



$h()$

data



$f()$



$g()$



$h()$



Some subsetting

data

```
graph TD; data[data] --> f[f()]; f --> g[g()]; g --> h[h()]; h --> sub[Some subsetting]; sub --> new[new data]
```

$f()$

$g()$

$h()$

Some subsetting

new data

f(x)

f(x)

sort(1:10)

f(x)

sort(1:10)

x %>% f()

`f(x)`

`sort(1:10)`

`x %>% f()`

`1:10 %>% sort()`

$f(x, y)$

`t.test(data$x, data$y)`

$f(x, y)$	<code>t.test(data\$x, data\$y)</code>
<code>x %>% f(y)</code>	<code>data\$x %>% t.test(data\$y)</code>



```
data <- read.csv("my-data.csv")  
  
data1 <- f(data, arg1 = "something")  
data2 <- g(data1, another.thing = "blah")  
data3 <- h(data2, a.setting = TRUE)  
data4 <- data3[data3$a.column == "cough", ]
```



```
data <- read.csv("my-data.csv")
```

```
data <-  
  h(  
    g(  
      f(data, arg1 = "something"),  
      another.thing = "blah"  
    ),  
    a.setting = TRUE  
)
```

data <- data[data\$a.column == "cough",]



$h(g(f(x)))$

$h(g(f(x)))$

$x \% > \%$

`h(g(f(x)))`

`x %>%`

`f() %>%`

`h(g(f(x)))`

`x %>%`

`f() %>%`

`g() %>%`

$h(g(f(x)))$

$x \%>%$

$f() \%>%$

$g() \%>%$

$h()$

`data`

\downarrow

$f()$

\downarrow

$g()$

\downarrow

$h()$

Some subsetting

`new data`

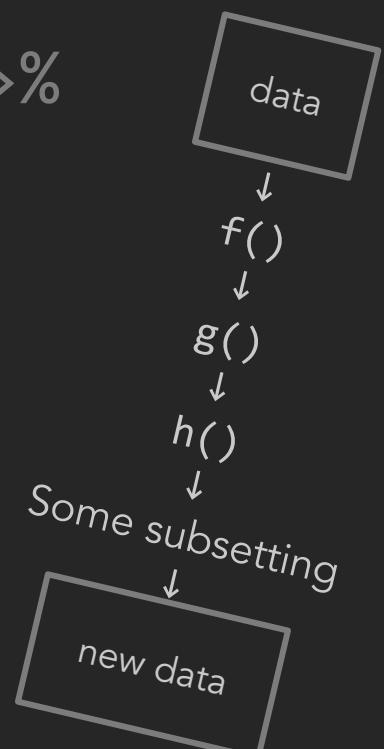
```
data <- read.csv("my-data.csv")  
  
data <-  
  h(  
    g(  
      f(data, arg1 = "something"),  
      another.thing = "blah"  
    ),  
    a.setting = TRUE  
  )
```

```
data <- read.csv("my-data.csv")
```

```
data <- data %>%
  f(arg1 = "something") %>%
  g(another.thing = "blah") %>%
  h(a.setting = TRUE)
```

```
data <- read.csv("my-data.csv")
```

```
data <- data %>%  
  f(arg1 = "something") %>%  
  g(another.thing = "blah") %>%  
  h(a.setting = TRUE)
```



```
data <- read.csv("my-data.csv")
```

```
data <- data %>%  
  f(arg1 = "something") %>%  
  g(another.thing = "blah") %>%  
  h(a.setting = TRUE)
```



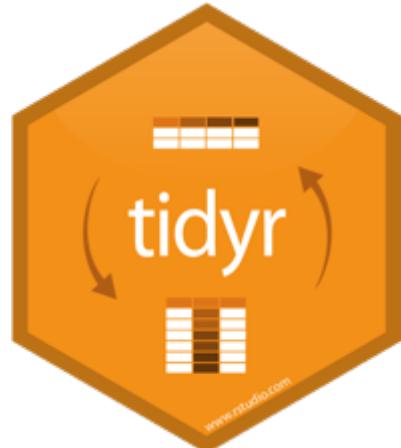
data <- data[data\$a.column == "cough",]

???????

> workshop\$outline[1:3]

DAY 1

Tidy data principles
& `tidyverse`



DAY 2

Manipulating data
& an intro to `dplyr`



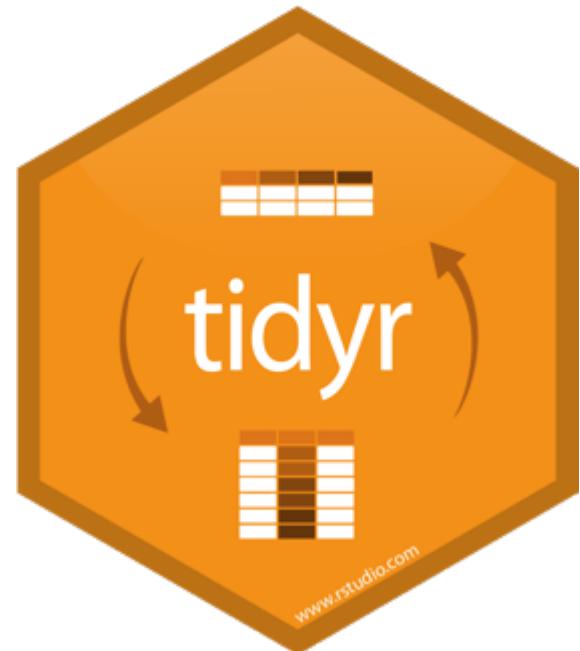
DAY 3

Extending your data
with `mutate()`,
`summarise()`
& friends

```
> workshop$outline[[1]]
```

DAY 1

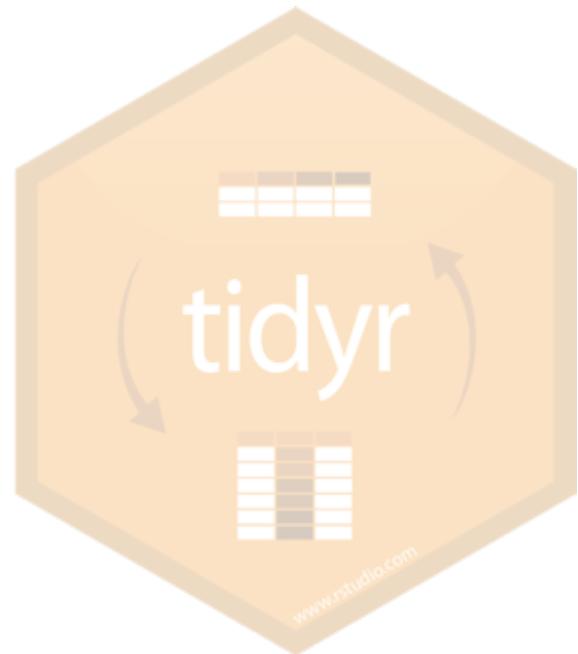
Tidy data principles
& *tidyverse*



```
> workshop$outline[[1]]
```

DAY 1

Tidy **data** principles
& **tidyR**



A motivating example...



Kogelberg NR,
R. van Mazijk
2018



Observation Pk,
R. van Mazijk 2018

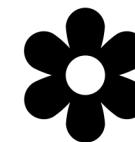
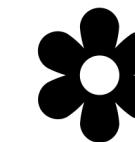
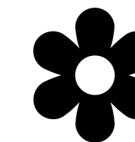
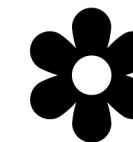
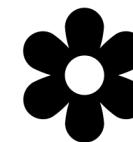
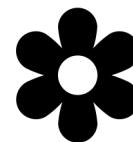
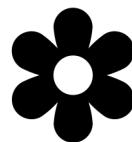
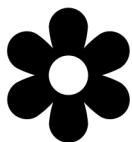
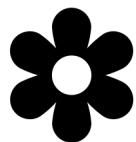


Near Pearly Beach, Agulhas Plains,
R. van Mazijk 2018

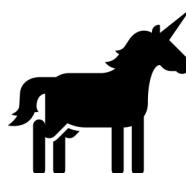
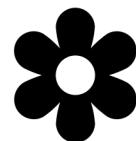
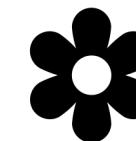
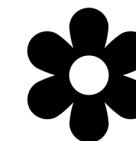
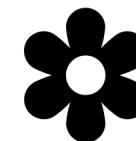
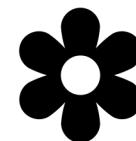
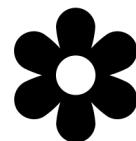
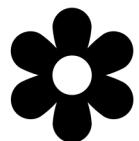
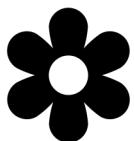
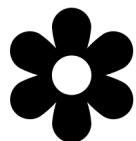
An example data-collection scenario in biology



An example data-collection scenario in biology



An example data-collection scenario in biology



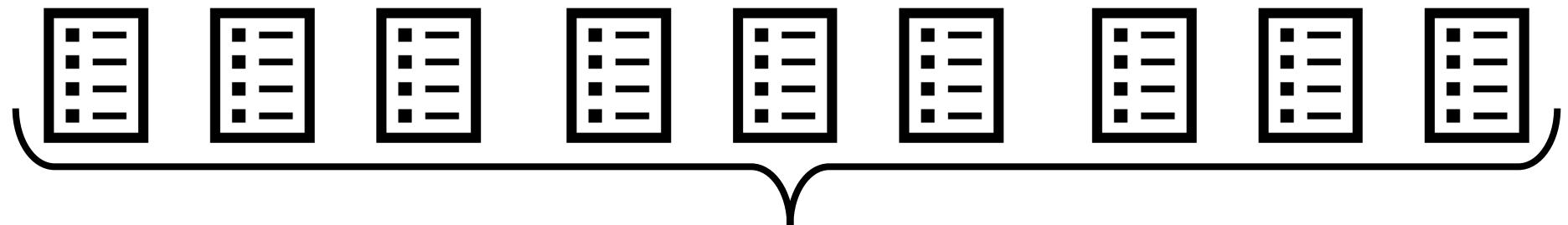
An example data-collection scenario in biology

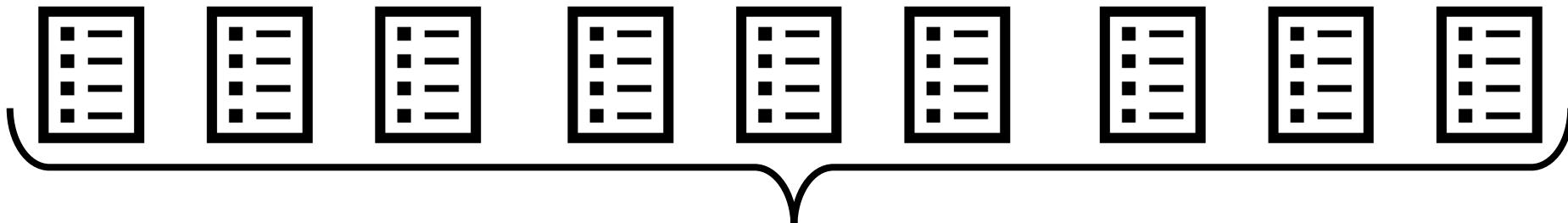


An example data-collection scenario in biology



(A good way to *collect* your data!)







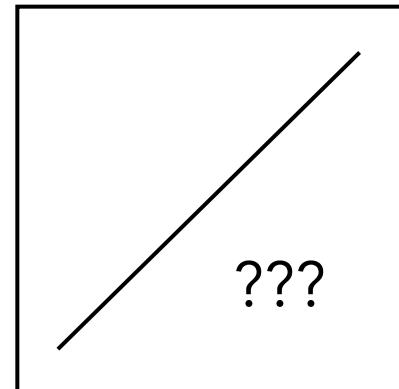
Site 1			Site 2			Site 3		
Sp 1	Sp 2	Sp 3	Sp 1	Sp 2	Sp 3	Sp 1	Sp 2	Sp 3
❖ unicorn								
❖ unicorn								
❖ unicorn								
❖ unicorn								

One way to lay out your collected data...

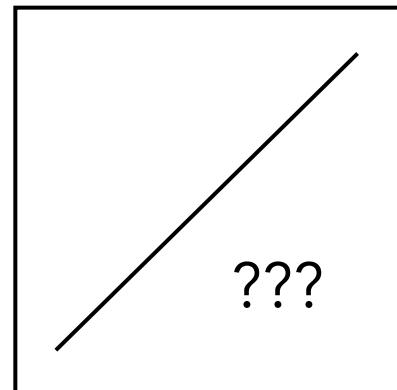


Site 1			Site 2			Site 3		
Sp 1	Sp 2	Sp 3	Sp 1	Sp 2	Sp 3	Sp 1	Sp 2	Sp 3
 	 	 	 	 	 	 	 	 
 	 	 	 	 	 	 	 	 
 	 	 	 	 	 	 	 	 
 	 	 	 	 	 	 	 	 

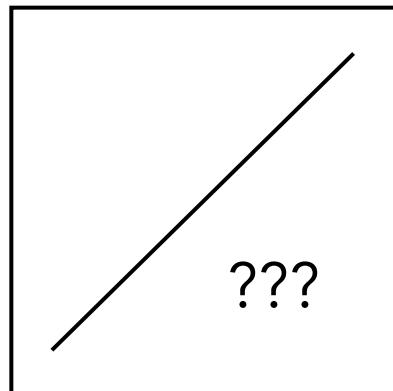
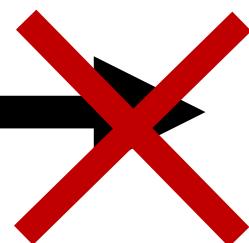
Site 1			Site 2			Site 3		
Sp 1	Sp 2	Sp 3	Sp 1	Sp 2	Sp 3	Sp 1	Sp 2	Sp 3



Site 1			Site 2			Site 3		
Sp 1	Sp 2	Sp 3	Sp 1	Sp 2	Sp 3	Sp 1	Sp 2	Sp 3



Site 1			Site 2			Site 3		
Sp 1	Sp 2	Sp 3	Sp 1	Sp 2	Sp 3	Sp 1	Sp 2	Sp 3



	Site 1		Site 2		Site 3	
Sp						

Another way...



Sp	Site	*	Unicorn



The “best” way.
(Will make your life easiest in the long-term.)

TIDY DATA

Sp	Site	*	🦄



The “best” way.
(Will make your life easiest in the long-term.)

TIDY DATA

TIDY DATA

country	year	cases	population
Afghanistan	1990	1745	1987071
Afghanistan	2000	2666	20595360
Brazil	1999	31737	172006362
Brazil	2000	80488	174504898
China	1999	21258	1272915272
China	2000	21666	128042583

variables

country	year	cases	population
Afghanistan	1990	1745	1987071
Afghanistan	2000	2666	20595360
Brazil	1999	31737	172006362
Brazil	2000	80488	174504898
China	1999	21258	1272915272
China	2000	216766	128042583

observations

country	year	cases	population
Afghanistan	99	1745	1987071
Afghanistan	00	2666	20595360
Brazil	99	31737	172006362
Brazil	00	80488	174504898
China	99	21258	1272915272
China	00	216766	128042583

values

TIDY DATA

country	year	cases	population
Afghanistan	1990	745	1987071
Afghanistan	2000	2666	20595360
Brazil	1999	31737	172006362
Brazil	2000	80488	174504898
China	1999	21258	1272915272
China	2000	216766	128042583

variables

country	year	cases	population
Afghanistan	1990	745	1987071
Afghanistan	2000	2666	20595360
Brazil	1999	31737	172006362
Brazil	2000	80488	174504898
China	1999	21258	1272915272
China	2000	216766	128042583

observations

country	year	cases	population
Afghanistan	99	745	1987071
Afghanistan	00	2666	20595360
Brazil	99	31737	172006362
Brazil	00	80488	174504898
China	99	21258	1272915272
China	00	216766	128042583

values

1. Each **variable** must have its own **column**
2. Each **observation** must have its own **row**
3. Each **value**, therefore, must have its own **cell**

country	year	cases	population
Afghanistan	1990	745	1987071
Afghanistan	2000	2666	20595360
Brazil	1999	31737	172006362
Brazil	2000	80488	174504898
China	1999	21258	1272915272
China	2000	21666	128042583

variables

country	year	cases	population
Afghanistan	1990	745	1987071
Afghanistan	2000	2666	20595360
Brazil	1999	31737	172006362
Brazil	2000	80488	174504898
China	1999	21258	1272915272
China	2000	216766	128042583

observations

country	year	cases	population
Afghanistan	99	745	1987071
Afghanistan	00	2666	20595360
Brazil	99	31737	172006362
Brazil	00	80488	174504898
China	99	21258	1272915272
China	00	216766	128042583

values

tidyverse

An R-package all about getting to this:

Verbs to tidy your data

Verbs to tidy your data

Untidy observations?

gather() # if > 1 observation per row

spread() # if observations live in > 1 row

Verbs to tidy your data

Untidy observations?

gather() # if > 1 observation per row

spread() # if observations live in > 1 row

Untidy variables?

separate() # if > 1 variable per column

unite() # if variables live in > 1 column

Note the following when choosing **tidyverse**-verbs:

Note the following when choosing `tidyverse`-verbs:

- Be clear on what your **observations** are:
 - Like, what **unit** of your study “counts” as an observation
 - E.g. Leaf traits: plant leaf vs plant individual
 - E.g. Reproductive success: egg size vs clutch size

Note the following when choosing `tidyverse`-verbs:

- Be clear on what your **observations** are:
 - Like, what **unit** of your study “counts” as an observation
 - E.g. Leaf traits: plant leaf vs plant individual
 - E.g. Reproductive success: egg size vs clutch size
 - This will depend on your study &/or data!

Note the following when choosing `tidyverse`-verbs:

- Be clear on what your **observations** are:
 - Like, what **unit** of your study “counts” as an observation
 - E.g. Leaf traits: plant leaf vs plant individual
 - E.g. Reproductive success: egg size vs clutch size
 - This will depend on your study &/or data!
- **Variables** are discrete, separate ideas!

Note the following when choosing `tidyverse`-verbs:

- Be clear on what your **observations** are:
 - Like, what **unit** of your study “counts” as an observation
 - E.g. Leaf traits: plant leaf vs plant individual
 - E.g. Reproductive success: egg size vs clutch size
 - This will depend on your study &/or data!
- **Variables** are discrete, separate ideas!
 - But again, this will depend on your study &/or data!

Verbs to tidy your data

Untidy observations?

gather() # if > 1 observation per row

spread() # if observations live in > 1 row

Untidy variables?

separate() # if > 1 variable per column

unite() # if variables live in > 1 column

Untidy observations?

```
# Untidy observations?  
gather()      # if > 1 observation per row
```

```
# Untidy observations?  
gather()      # if > 1 observation per row  
  
data %>%  
  gather(key, value, ...)
```

```
# Untidy observations?  
gather()      # if > 1 observation per row
```

```
data %>%  
  gather(key, value, ...)
```

country	1999	2000
A	0.7K	2K
B	37K	80K
C	212K	213K



country	year	cases
A	1999	0.7K
B	1999	37K
C	1999	212K
A	2000	2K
B	2000	80K
C	2000	213K

key value

```
# Untidy observations?  
gather()      # if > 1 observation per row
```

```
data %>%  
  gather(key, value, ...)
```

country	1999	2000
A	0.7K	2K
B	37K	80K
C	212K	213K



country	year	cases
A	1999	0.7K
B	1999	37K
C	1999	212K
A	2000	2K
B	2000	80K
C	2000	213K

key value

```
# Untidy observations?  
gather()      # if > 1 observation per row
```

```
data %>%  
  gather(year, cases, 1999, 2000)
```

country	1999	2000
A	0.7K	2K
B	37K	80K
C	212K	213K



country	year	cases
A	1999	0.7K
B	1999	37K
C	1999	212K
A	2000	2K
B	2000	80K
C	2000	213K

key value

```
# Untidy observations?  
spread()      # if observations live in > 1 row
```

```
# Untidy observations?  
spread()      # if observations live in > 1 row  
  
data %>%  
  spread(key, value)
```

```
# Untidy observations?  
spread()      # if observations live in > 1 row  
  
data %>%  
  spread(key, value)
```

```
# Untidy observations?  
spread()      # if observations live in > 1 row
```

```
data %>%  
  spread(key, value)
```

The diagram illustrates the process of "tidying" data using the `spread()` function. On the left, an "untidy" data frame is shown as a grid of rows and columns. The columns are labeled `country`, `year`, `type`, and `count`. The data consists of 12 rows, grouped by country (A, B, C) and year (1999, 2000). The `type` column contains two values: `cases` and `pop`. The `count` column contains numerical values representing thousands (K), millions (M), or billions (T). An arrow points from this untidy frame to a "tidy" data frame on the right. In the tidy frame, the `country` and `year` columns remain, but the `type` and `count` columns are split into separate columns: `cases` and `pop`. The data is now organized by country, year, and type, with each observation having its own row.

country	year	type	count
A	1999	cases	0.7K
A	1999		19M
A	2000	cases	2K
A	2000		20M
B	1999	cases	37K
B	1999	pop	172M
B	2000	cases	80K
B	2000	pop	174M
C	1999	cases	212K
C	1999	pop	1T
C	2000	cases	213K
C	2000	pop	1T

key value

country	year	cases	pop
A	1999	0.7K	19M
A	2000	2K	20M
B	1999	37K	172M
B	2000	80K	174M
C	1999	212K	1T
C	2000	213K	1T

```
# Untidy observations?  
spread()      # if observations live in > 1 row
```

```
data %>%  
  spread(type, count)
```

The diagram illustrates the process of "tidying" data using the `spread()` function. On the left, an "untidy" dataset is shown as a wide table with four columns: `country`, `year`, `type`, and `count`. The `type` column contains two types of data: "cases" and "pop". The `count` column contains values like "0.7K", "19M", "2K", etc. An arrow points from this untidy table to a "tidy" version on the right. In the tidy version, the `type` column has been moved into new columns: `cases` and `pop`. The `country` and `year` columns remain as identifiers. The `count` values are now placed under their respective `cases` and `pop` columns, resulting in a long table where each row corresponds to a specific country, year, type, and value.

country	year	type	count
A	1999	cases	0.7K
A	1999		19M
A	2000	cases	2K
A	2000		20M
B	1999	cases	37K
B	1999	pop	172M
B	2000	cases	80K
B	2000	pop	174M
C	1999	cases	212K
C	1999	pop	1T
C	2000	cases	213K
C	2000	pop	1T

key value

country	year	cases	pop
A	1999	0.7K	19M
A	2000	2K	20M
B	1999	37K	172M
B	2000	80K	174M
C	1999	212K	1T
C	2000	213K	1T

Untidy variables?

```
# Untidy variables?  
separate() # if > 1 variable per column
```

```
# Untidy variables?  
separate() # if > 1 variable per column  
  
data %>%  
  separate(col, into, sep)
```

```
# Untidy variables?  
separate() # if > 1 variable per column  
  
data %>%  
  separate(col, into)
```

```
# Untidy variables?  
separate() # if > 1 variable per column
```

```
data %>%  
  separate(col, into)
```

country	year	rate
A	1999	0.7K/19M
A	2000	2K/20M
B	1999	37K/172M
B	2000	80K/174M
C	1999	212K/1T
C	2000	213K/1T



country	year	cases	pop
A	1999	0.7K	19M
A	2000	2K	20M
B	1999	37K	172
B	2000	80K	174
C	1999	212K	1T
C	2000	213K	1T

```
# Untidy variables?  
separate() # if > 1 variable per column
```

```
data %>%  
  separate(rate, c("cases", "pop"))
```

country	year	rate
A	1999	0.7K/19M
A	2000	2K/20M
B	1999	37K/172M
B	2000	80K/174M
C	1999	212K/1T
C	2000	213K/1T



country	year	cases	pop
A	1999	0.7K	19M
A	2000	2K	20M
B	1999	37K	172
B	2000	80K	174
C	1999	212K	1T
C	2000	213K	1T

```
# Untidy variables?  
unite()      # if variables live in > 1 column
```

```
# Untidy variables?  
unite()      # if variables live in > 1 column  
  
data %>%  
  unite(col, ..., sep)
```

```
# Untidy variables?  
unite()      # if variables live in > 1 column
```

```
data %>%  
  unite(col, ...)
```

country	century	year
Afghan	19	99
Afghan	20	0
Brazil	19	99
Brazil	20	0
China	19	99
China	20	0



country	year
Afghan	1999
Afghan	2000
Brazil	1999
Brazil	2000
China	1999
China	2000

```
# Untidy variables?  
unite()      # if variables live in > 1 column  
  
data %>%  
  unite(year, century, year)
```



country	century	year
Afghan	19	99
Afghan	20	0
Brazil	19	99
Brazil	20	0
China	19	99
China	20	0

→

country	year
Afghan	1999
Afghan	2000
Brazil	1999
Brazil	2000
China	1999
China	2000

```
> demo()
```

DATASETS:

> `demo()`

tinyurl.com/unicorns-day-1
tinyurl.com/prepost-day-1
tinyurl.com/lang-day-1