dplyr

Data Manipulation

Changing the variables, values, and
units of analysis contained in the data set.

Data Tidying

Changing the layout of tabular data to make it
suitable for a particular piece of software (R).

Data Visualization

Transforming the data to a visual format that
reveals visual patterns.

Data sets contain more information than they display

tb

Tuberculosis cases by country collected by the WHO for the *Global Tuberculosis Report*

```
library(EDAWR)
?tb
```

World Health Organization

tb

(3,800 x 6)

Number of cases reported by *country*, *year*, *sex* and *age* group

```
View(tb)
```

country	year	sex	child	adult	elderly
Afghanistan	1995	female	NA	NA	NA
Afghanistan	1995	male	NA	NA	NA
Afghanistan	1996	female	NA	NA	NA
Afghanistan	1996	male	NA	NA	NA
Afghanistan	1997	female	5	96	1
Afghanistan	1997	male	0	26	0

Goal

Number of cases reported by *country* and *year*

(3,800 x 6)

country	year	sex	child	adult	elderly
Afghanistan	1995	female	NA	NA	NA
Afghanistan	1995	male	NA	NA	NA
Afghanistan	1996	female	NA	NA	NA
Afghanistan	1996	male	NA	NA	NA
Afghanistan	1997	female	5	96	1
Afghanistan	1997	male	0	26	0

→

(1,691 x 3)

country	year	cases
Afghanistan	1997	128
Afghanistan	1998	1778
Afghanistan	1999	745
Afghanistan	2000	2666
Afghanistan	2001	4639
Afghanistan	2002	6509

dplyr

A package that helps transform tabular data.

```
# install.packages("dplyr")
library(dplyr)          ?select
?left_join              ?filter
?inner_join             ?mutate
?semi_join              ?summarise
?anti_join              ?group_by
```

Ways to access information

1	**Extract** existing variables.	select()
2	**Extract** existing observations.	filter()
3	**Derive** new variables (from existing variables)	mutate()
4	**Derive** new observations (from existing observations)	summarise()

Select variables

select()

storms

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

→

storm	pressure
Alberto	1007
Alex	1009
Allison	1005
Ana	1013
Arlene	1010
Arthur	1010

```
select(storms, storm, pressure)
```

select()

storms

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

→

wind	pressure	date
110	1007	2000-08-12
45	1009	1998-07-30
65	1005	1995-06-04
40	1013	1997-07-01
50	1010	1999-06-13
45	1010	1996-06-21

```
select(storms, -storm)
# see ?select for more
```

select()

storms

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

→

wind	pressure	date
110	1007	2000-08-12
45	1009	1998-07-30
65	1005	1995-06-04
40	1013	1997-07-01
50	1010	1999-06-13
45	1010	1996-06-21

```
select(storms, wind:date)
# see ?select for more
```

Useful select functions

* Blue functions come in dplyr

-	Select everything but
:	Select range
contains()	Select columns whose name contains a character string
ends_with()	Select columns whose name ends with a string
everything()	Select every column
matches()	Select columns whose name matches a regular expression
num_range()	Select columns named x1, x2, x3, x4, x5
one_of()	Select columns whose names are in a group of names
starts_with()	Select columns whose name starts with a character string

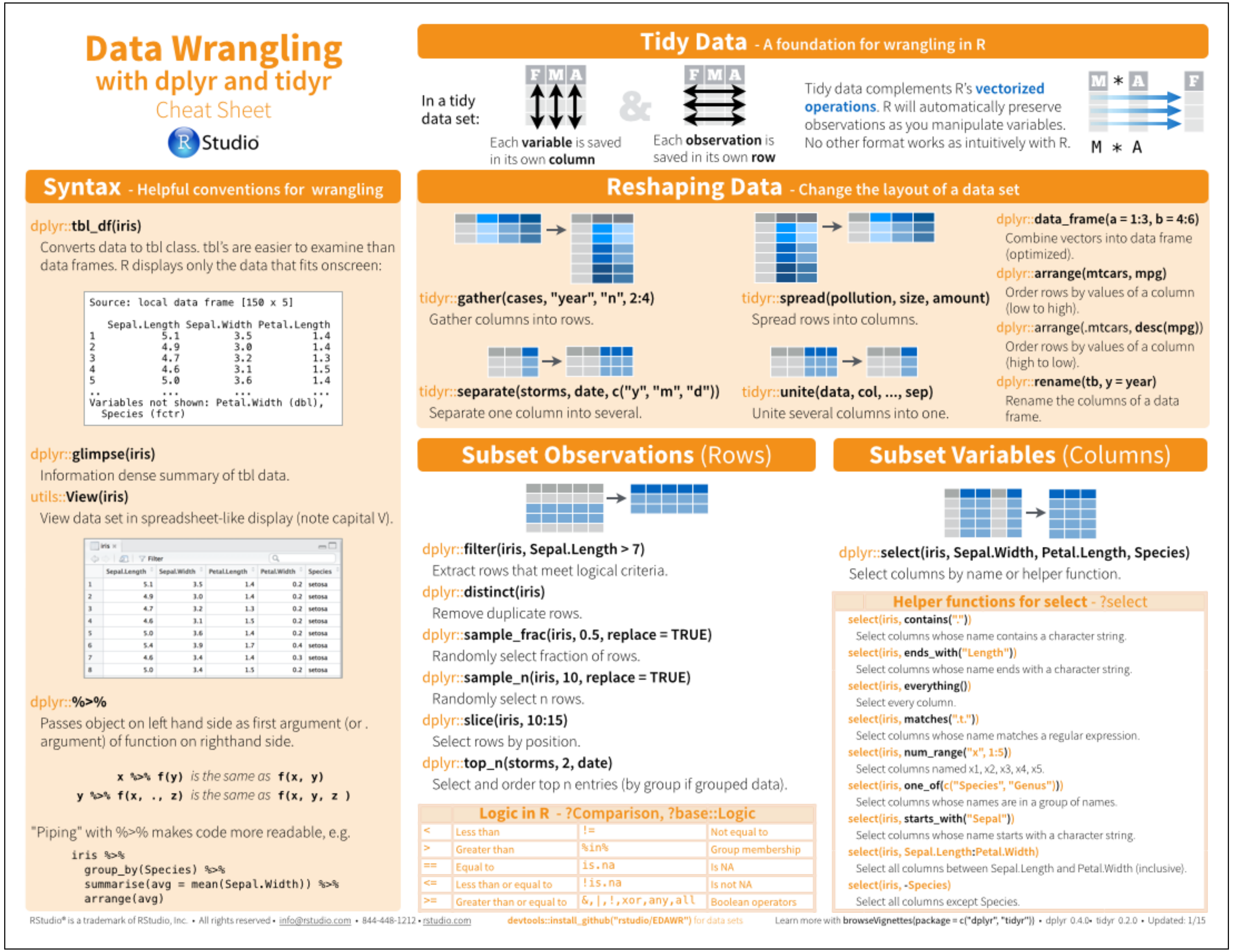
Your Turn

Use select to return just these columns from flights:

1. **dep_delay** and **dep_time**
2. **dep_time**, **arr_time**, and **air_time**
3. **dep_time**, **dep_delay**, **arr_time**, and **arr_delay**

flights (336,776 x 16)

year	month	day	dep_time	dep_delay	arr_time	arr_delay	carrier	tailnum	flight	origin	dest	air_time	distance	hour	minute
2013	1	1	517	2	830	11	UA	N14228	1545	EWR	IAH	227	1400	5	17
2013	1	1	533	4	850	20	UA	N24211	1714	LGA	IAH	227	1416	5	33



Data Wrangling with dplyr and tidyr

Cheat Sheet

RStudio

Syntax - Helpful conventions for wrangling

dplyr::**tbl_df(iris)**

Converts data to tbl class. tbl's are easier to examine than data frames. R displays only the data that fits onscreen:

```
Source: local data frame [150 x 5]

   Sepal.Length Sepal.Width Petal.Length
1           5.1         3.5          1.4
2           4.9         3.0          1.4
3           4.7         3.2          1.3
4           4.6         3.1          1.5
5           5.0         3.6          1.4
..          ...         ...          ...
Variables not shown: Petal.Width (dbl),
  Species (fctr)
```

dplyr::**glimpse(iris)**

Information dense summary of tbl data.

utils::**View(iris)**

View data set in spreadsheet-like display (note capital V).

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5.0	3.4	1.5	0.2	setosa

dplyr::**%>%**

Passes object on left hand side as first argument (or . argument) of function on righthand side.

x %>% f(y) *is the same as* **f(x, y)**

y %>% f(x, ., z) *is the same as* **f(x, y, z)**

"Piping" with %>% makes code more readable, e.g.

```
iris %>%
  group_by(Species) %>%
  summarise(avg = mean(Sepal.Width)) %>%
  arrange(avg)
```

Tidy Data - A foundation for wrangling in R

In a tidy data set:

Each **variable** is saved in its own **column**

Each **observation** is saved in its own **row**

Tidy data complements R's **vectorized operations**. R will automatically preserve observations as you manipulate variables. No other format works as intuitively with R.

Reshaping Data - Change the layout of a data set

tidyr::**gather(cases, "year", "n", 2:4)**

Gather columns into rows.

tidyr::**spread(pollution, size, amount)**

Spread rows into columns.

tidyr::**separate(storms, date, c("y", "m", "d"))**

Separate one column into several.

tidyr::**unite(data, col, ..., sep)**

Unite several columns into one.

dplyr::**data_frame(a = 1:3, b = 4:6)**

Combine vectors into data frame (optimized).

dplyr::**arrange(mtcars, mpg)**

Order rows by values of a column (low to high).

dplyr::**arrange(.mtcars, desc(mpg))**

Order rows by values of a column (high to low).

dplyr::**rename(tb, y = year)**

Rename the columns of a data frame.

Subset Observations (Rows)

dplyr::**filter(iris, Sepal.Length > 7)**

Extract rows that meet logical criteria.

dplyr::**distinct(iris)**

Remove duplicate rows.

dplyr::**sample_frac(iris, 0.5, replace = TRUE)**

Randomly select fraction of rows.

dplyr::**sample_n(iris, 10, replace = TRUE)**

Randomly select n rows.

dplyr::**slice(iris, 10:15)**

Select rows by position.

dplyr::**top_n(storms, 2, date)**

Select and order top n entries (by group if grouped data).

Logic in R - ?Comparison, ?base::Logic

<	Less than	!=	Not equal to
>	Greater than	%in%	Group membership
==	Equal to	is.na	Is NA
<=	Less than or equal to	!is.na	Is not NA
>=	Greater than or equal to	&,|,!,xor,any,all	Boolean operators

Subset Variables (Columns)

dplyr::**select(iris, Sepal.Width, Petal.Length, Species)**

Select columns by name or helper function.

Helper functions for select - ?select

select(iris, **contains(".")**)

Select columns whose name contains a character string.

select(iris, **ends_with("Length")**)

Select columns whose name ends with a character string.

select(iris, **everything()**)

Select every column.

select(iris, **matches(".t.")**)

Select columns whose name matches a regular expression.

select(iris, **num_range("x", 1:5)**)

Select columns named x1, x2, x3, x4, x5.

select(iris, **one_of(c("Species", "Genus"))**)

Select columns whose names are in a group of names.

select(iris, **starts_with("Sepal")**)

Select columns whose name starts with a character string.

select(iris, Sepal.Length:Petal.Width)

Select all columns between Sepal.Length and Petal.Width (inclusive).

select(iris, -Species)

Select all columns except Species.

RStudio® is a trademark of RStudio, Inc. • All rights reserved • info@rstudio.com • 844-448-1212 • rstudio.com devtools::install_github("rstudio/EDAWR") for data sets Learn more with **browseVignettes(package = c("dplyr", "tidyr"))** • dplyr 0.4.0• tidyr 0.2.0 • Updated: 1/15

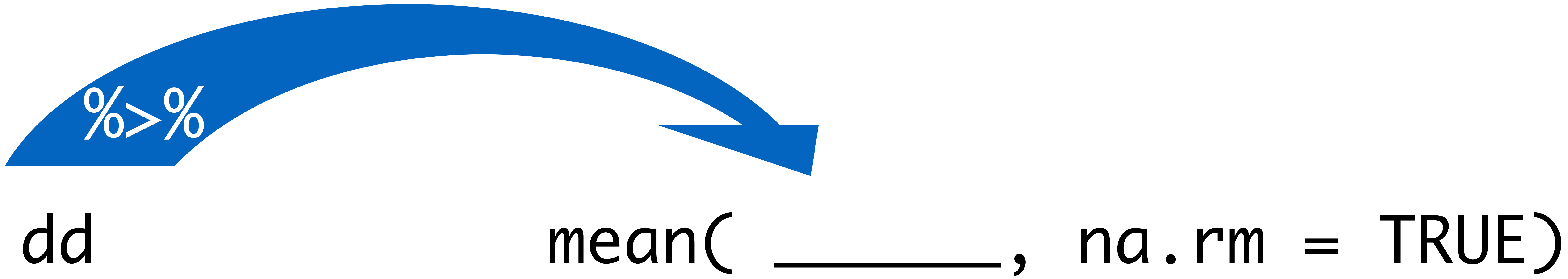
http://www.rstudio.com/resources/cheatsheets/

```
select(flights, starts_with("dep"))
select(flights, ends_with("time"))
select(flights, dep_time:arr_delay)
```

The pipe operator %>%

```
dd <- flights$dep_delay
mean(dd, na.rm = TRUE)
dd %>% mean(na.rm = TRUE)
```

These do the same thing
Try it!



%>%

```
dd                mean( ______, na.rm = TRUE)
```

```
flights %>% select(starts_with("dep"))
flights %>% select(ends_with("time"))
flights %>% select(dep_time:arr_delay)
```

Shortcut to type %>%

Cmd + Shift + M (Mac)

Ctrl + Shift + M (Windows)

Filter observations

filter()

storms

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

→

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Allison	65	1005	1995-06-04
Arlene	50	1010	1999-06-13

```
storms %>% filter(wind >= 50)
```

filter()

storms

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

→

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Ana	40	1013	1997-07-01

```
storms %>% filter(storm %in% c("Alberto", "Ana"))
```

filter()

storms

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

→

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Allison	65	1005	1995-06-04

```
storms %>% filter(wind >= 50,
  storm %in% c("Alberto", "Alex", "Allison"))
```

logical tests in R

?Comparison

<	Less than
>	Greater than
==	Equal to
<=	Less than or equal to
>=	Greater than or equal to
!=	Not equal to
%in%	Group membership
is.na	Is NA
!is.na	Is not NA

?base::Logic

&	boolean and
|	boolean or
xor	exactly or
!	not
any	any true
all	all true

Your Turn

Return just the rows of flights where **arr_delay** does not equal **NA**.

flights (336,776 x 16)

year	month	day	dep_time	dep_delay	arr_time	arr_delay	carrier	tailnum	flight	origin	dest	air_time	distance	hour	minute
2013	1	1	517	2	830	11	UA	N14228	1545	EWR	IAH	227	1400	5	17
2013	1	1	533	4	850	20	UA	N24211	1714	LGA	IAH	227	1416	5	33
2013	1	1	542	2	923	33	AA	N619AA	1141	JFK	MIA	160	1089	5	42
2013	1	1	544	-1	1004	-18	B6	N804JB	725	JFK	BQN	183	1576	5	44
2013	1	1	554	-6	812	-25	DL	N668DN	461	LGA	ATL	116	762	5	54
2013	1	1	554	-4	740	12	UA	N39463	1696	EWR	ORD	150	1065	5	54

```
flights %>% filter(!is.na(arr_delay))
```

(327,346 x 16)

year	month	day	dep_time	dep_delay	arr_time	arr_delay	carrier	tailnum	flight	origin	dest	air_time	distance	hour	minute
2013	1	1	517	2	830	11	UA	N14228	1545	EWR	IAH	227	1400	5	17
2013	1	1	533	4	850	20	UA	N24211	1714	LGA	IAH	227	1416	5	33
2013	1	1	542	2	923	33	AA	N619AA	1141	JFK	MIA	160	1089	5	42
2013	1	1	544	-1	1004	-18	B6	N804JB	725	JFK	BQN	183	1576	5	44
2013	1	1	554	-6	812	-25	DL	N668DN	461	LGA	ATL	116	762	5	54

storms

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

→

storm	pressure
Alberto	1007
Allison	1005
Arlene	1010

```
storms %>%
  filter(wind >= 50) %>%
  select(storm, pressure)
```

Your Turn

Filter flights to the rows where **arr_delay** != **NA.**

Then select just the **carrier** and **arr_delay** variables from the results.

flights (336,776 x 16)

year	month	day	dep_time	dep_delay	arr_time	arr_delay	carrier	tailnum	flight	origin	dest	air_time	distance	hour	minute
2013	1	1	517	2	830	11	UA	N14228	1545	EWR	IAH	227	1400	5	17
2013	1	1	533	4	850	20	UA	N24211	1714	LGA	IAH	227	1416	5	33
2013	1	1	542	2	923	33	AA	N619AA	1141	JFK	MIA	160	1089	5	42
2013	1	1	544	-1	1004	-18	B6	N804JB	725	JFK	BQN	183	1576	5	44
2013	1	1	554	-6	812	-25	DL	N668DN	461	LGA	ATL	116	762	5	54
2013	1	1	554	-4	740	12	UA	N39463	1696	EWR	ORD	150	1065	5	54

```
flights %>%
  filter(!is.na(arr_delay)) %>%
  select(carrier, arr_delay)
```

(327,346 x 2)

carrier	arr_delay
UA	11
UA	20
AA	33
B6	-18
DL	-25
UA	12

Derive variables

Ways to access information

1	**Extract** existing variables.	select()	✓
2	**Extract** existing observations.	filter()	✓
3	**Derive** new variables (from existing variables)	mutate()	
4	**Derive** new observations (from existing observations)	summarise()	

mutate()

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

→

storm	wind	pressure	date	ratio
Alberto	110	1007	2000-08-12	9.15
Alex	45	1009	1998-07-30	22.42
Allison	65	1005	1995-06-04	15.46
Ana	40	1013	1997-07-01	25.32
Arlene	50	1010	1999-06-13	20.20
Arthur	45	1010	1996-06-21	22.44

```
storms %>% mutate(ratio = pressure / wind)
```

mutate()

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

→

storm	wind	pressure	date	ratio	inverse
Alberto	110	1007	2000-08-12	9.15	0.11
Alex	45	1009	1998-07-30	22.42	0.04
Allison	65	1005	1995-06-04	15.46	0.06
Ana	40	1013	1997-07-01	25.32	0.04
Arlene	50	1010	1999-06-13	20.20	0.05
Arthur	45	1010	1996-06-21	22.44	0.04

```
storms %>% mutate(ratio = pressure / wind, inverse = ratio^-1)
```

mutate()

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

? →

storm	ratio
Alberto	9.15
Alex	22.42
Allison	15.46
Ana	25.32
Arlene	20.20
Arthur	22.44

```
storms %>%
  mutate(ratio = pressure / wind) %>%
  select(storm, ratio)
```

Useful mutate functions

* All take a vector of values and return a vector of values

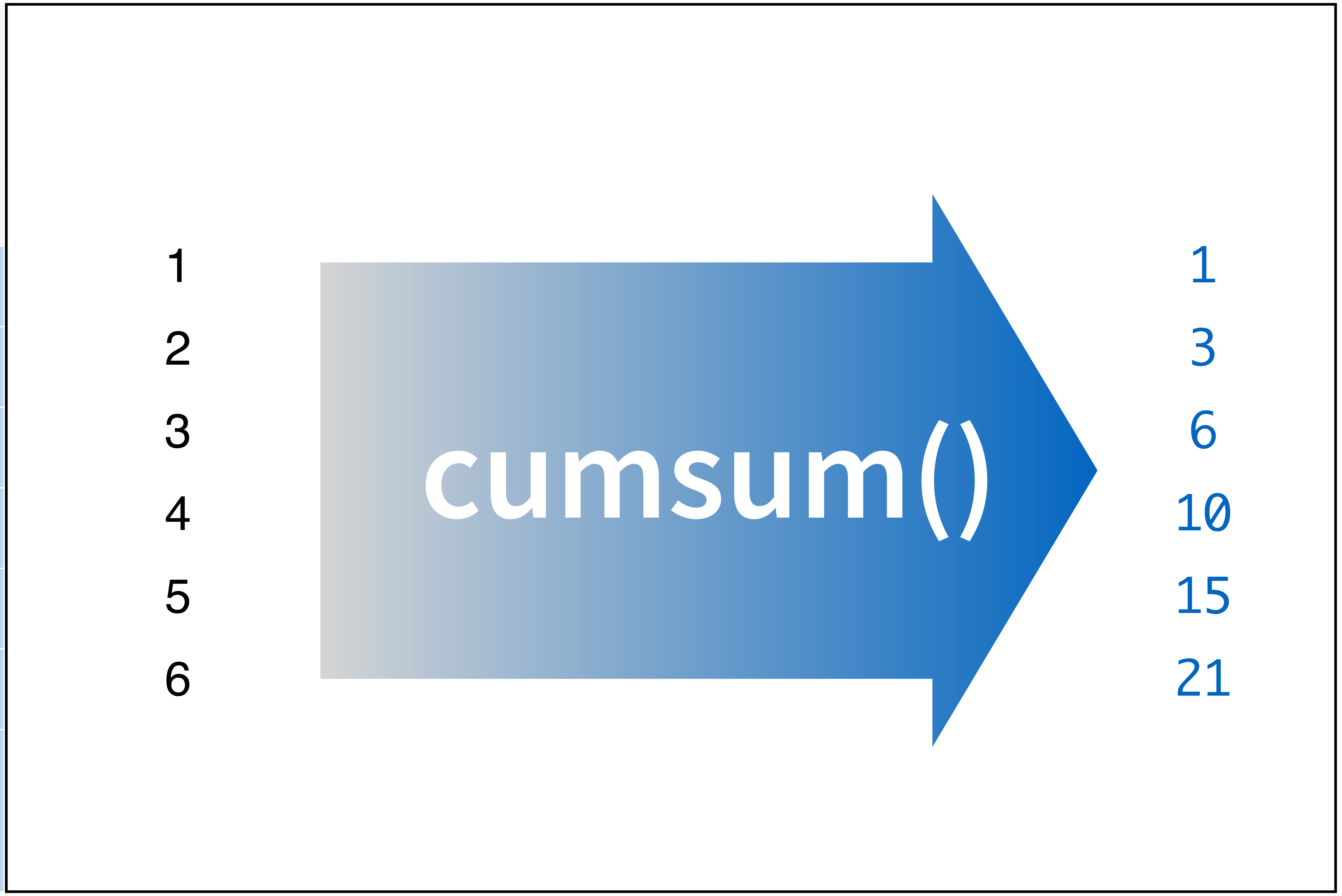
** Blue functions come in dplyr

pmin(), pmax()	Element-wise min and max
cummin(), cummax()	Cumulative min and max
cumsum(), cumprod()	Cumulative sum and product
between()	Are values between a and b?
cume_dist()	Cumulative distribution of values
cumall(), cumany()	Cumulative all and any
cummean()	Cumulative mean
lead(), lag()	Copy with values one position
ntile()	Bin vector into n buckets
dense_rank(), min_rank(), percent_rank(), row_number()	Various ranking methods

"Window" functions

* All take a vector of values and return a vector of values

- pmin(), pmax()
- cummin(), cummax()
- cumsum(), cumprod()
- between()
- cume_dist()
- cumall(), cumany()
- cummean()
- lead(), lag()
- ntile()
- dense_rank(), min_rank(), percent_rank(), row_number()



Your Turn

Use `mutate()`, `select()`, and %>% to make a data set with three variables: **carrier**, **arr_delay**, and **speed** (e.g., **distance** / **air_time** * 60)

flights (336,776 x 16)

year	month	day	dep_time	dep_delay	arr_time	arr_delay	carrier	tailnum	flight	origin	dest	air_time	distance	hour	minute
2013	1	1	517	2	830	11	UA	N14228	1545	EWR	IAH	227	1400	5	17
2013	1	1	533	4	850	20	UA	N24211	1714	LGA	IAH	227	1416	5	33
2013	1	1	542	2	923	33	AA	N619AA	1141	JFK	MIA	160	1089	5	42
2013	1	1	544	-1	1004	-18	B6	N804JB	725	JFK	BQN	183	1576	5	44
2013	1	1	554	-6	812	-25	DL	N668DN	461	LGA	ATL	116	762	5	54
2013	1	1	554	-4	740	12	UA	N39463	1696	EWR	ORD	150	1065	5	54

```
flights %>%
  mutate(speed = distance / air_time * 60) %>%
  select(carrier, arr_delay, speed)
```

(336,776 x 3)

carrier	arr_delay	speed
UA	11	370.0
UA	20	374.3
AA	33	408.4
B6	-18	516.7
DL	-25	394.1
UA	12	287.6

Summarise observations

Ways to access information

1	**Extract** existing variables.	select()	✓
2	**Extract** existing observations.	filter()	✓
3	**Derive** new variables (from existing variables)	mutate()	✓
4	**Derive** new observations (from existing observations)	summarise()	

summarise()

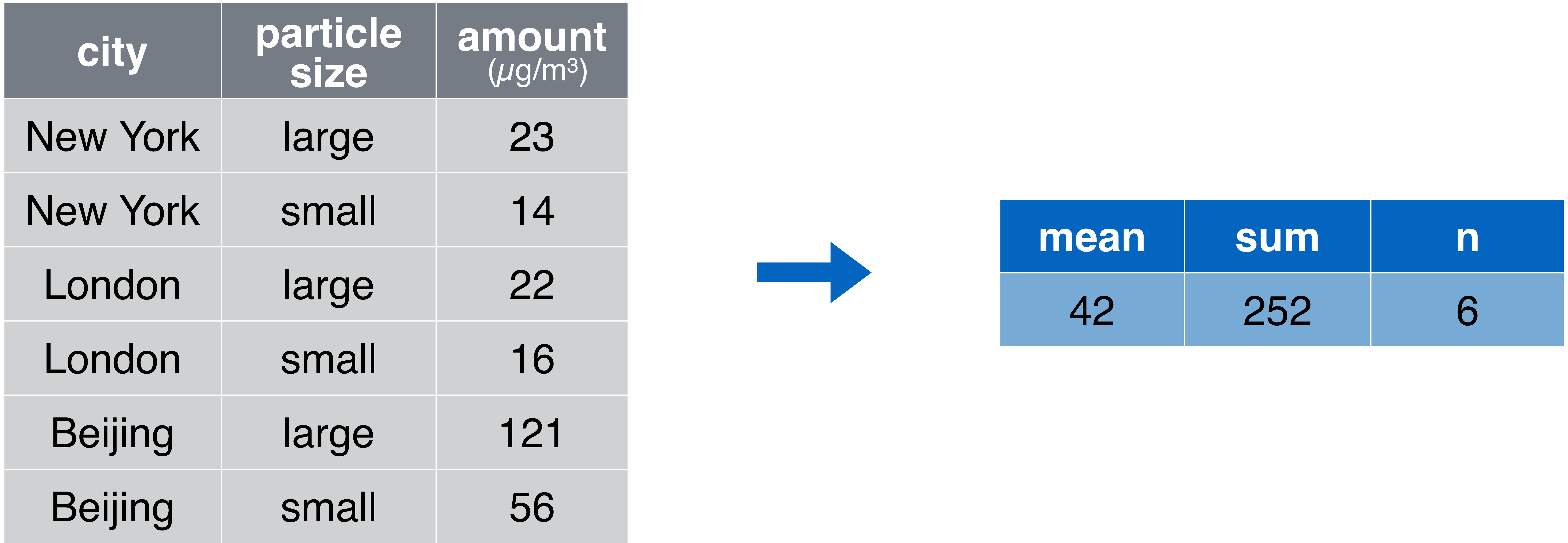
city	particle size	amount (μg/m^3)
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

→

median	variance
22.5	1731.6

```
pollution %>% summarise(median = median(amount), variance = var(amount))
```

summarise()



city	particle size	amount ($\mu g/m^3$)
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

mean	sum	n
42	252	6

```
pollution %>% summarise(mean = mean(amount), sum = sum(amount), n = n())
```

Useful summary functions

* All take a vector of values and return a single value
** Blue functions come in dplyr

min(), max()	Minimum and maximum values
mean()	Mean value
median()	Median value
sum()	Sum of values
var, sd()	Variance and standard deviation of a vector
first()	First value in a vector
last()	Last value in a vector
nth()	Nth value in a vector
n()	The number of values in a vector
n_distinct()	The number of distinct values in a vector

"Summary" functions

* All take a vector of values and return a single value

min(), max()

mean()

median()

sum()

var, sd()

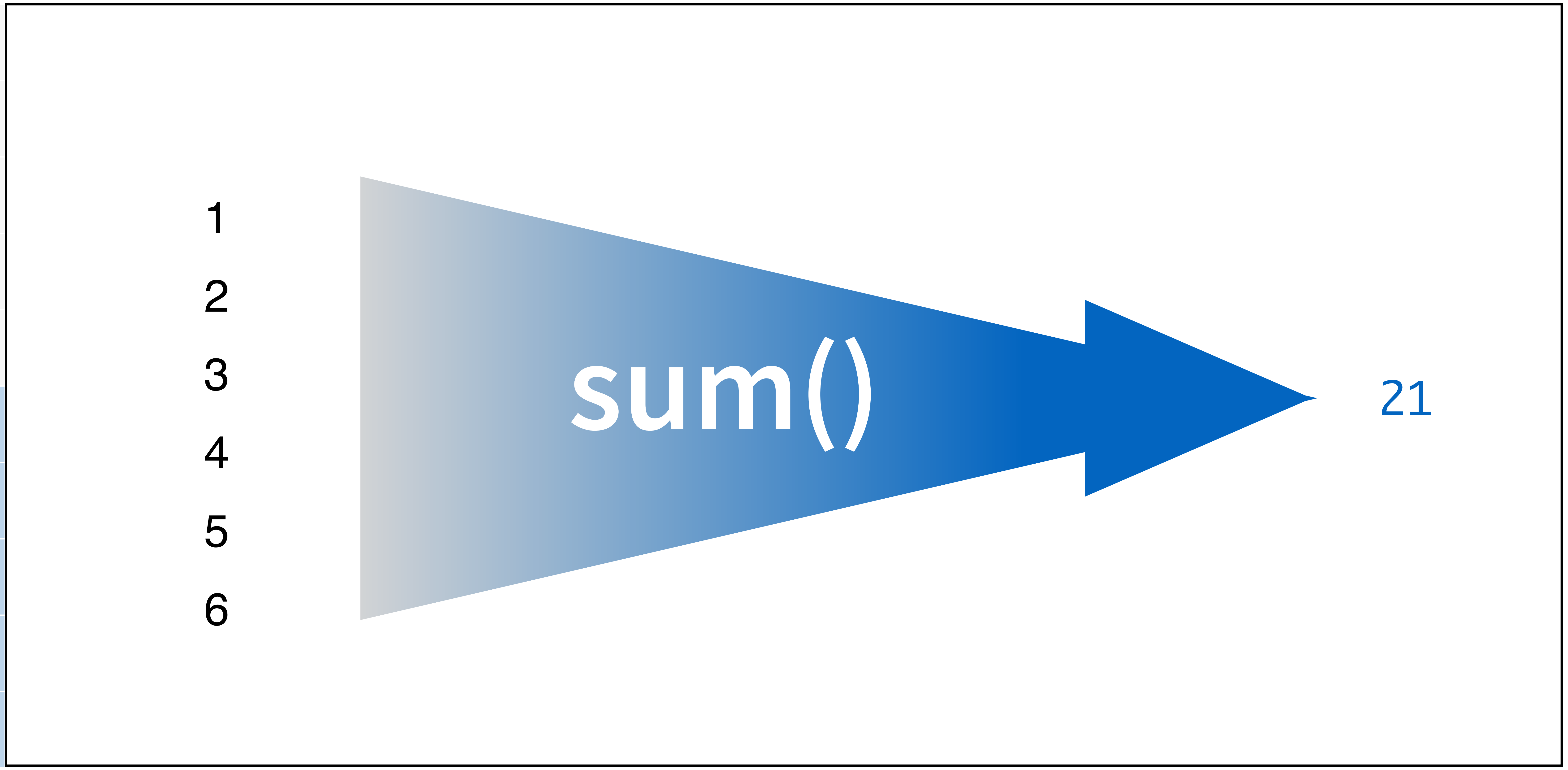
first()

last()

nth()

n()

n_distinct()



Your Turn

`filter()` out observations where **air_time** and **distance** equal NA. Then create a summary that shows:

- **n** - the total number of flights (e.g. rows) in the data set
- **n_carriers** - the number of distinct airlines in the data set
- **total_time** - the total number of minutes planes in the data set spent in the air
- **total_dist** - the total distance travelled by planes in the data set

flights (336,776 x 16)

year	month	day	dep_time	dep_delay	arr_time	arr_delay	carrier	tailnum	flight	origin	dest	air_time	distance	hour	minute
2013	1	1	517	2	830	11	UA	N14228	1545	EWR	IAH	227	1400	5	17
2013	1	1	533	4	850	20	UA	N24211	1714	LGA	IAH	227	1416	5	33

```
flights %>%
  filter(!is.na(air_time), !is.na(distance)) %>%
  summarise(n = n(), n_carriers = n_distinct(carrier),
    total_time = sum(air_time), total_dist = sum(distance))
```

(1 x 4)

n	n_carriers	total_time	total_dist
327346	16	49326610	343180156

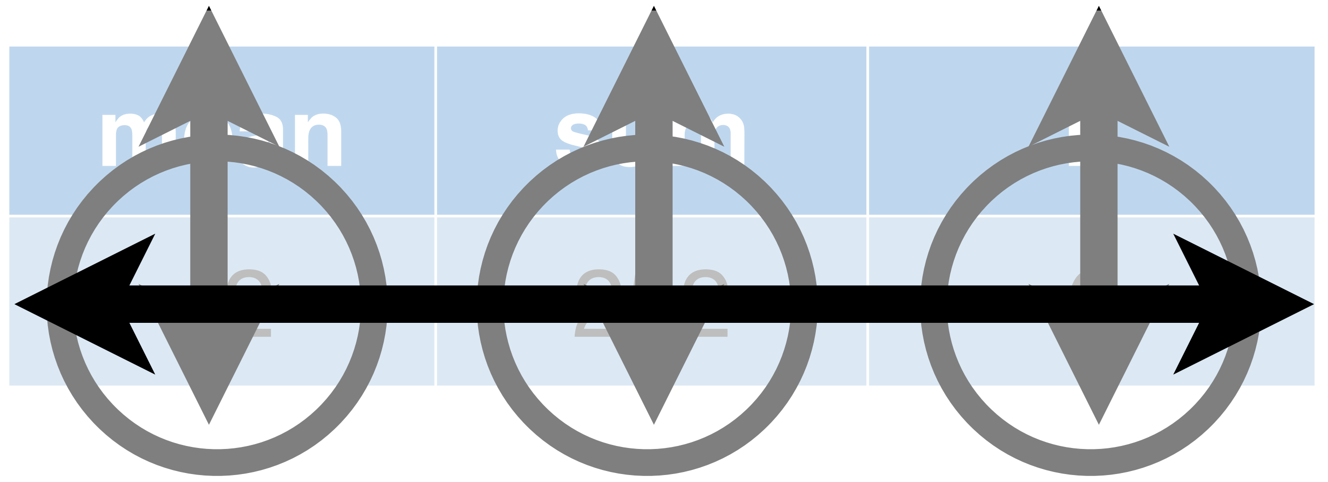
Group observations

summarise()

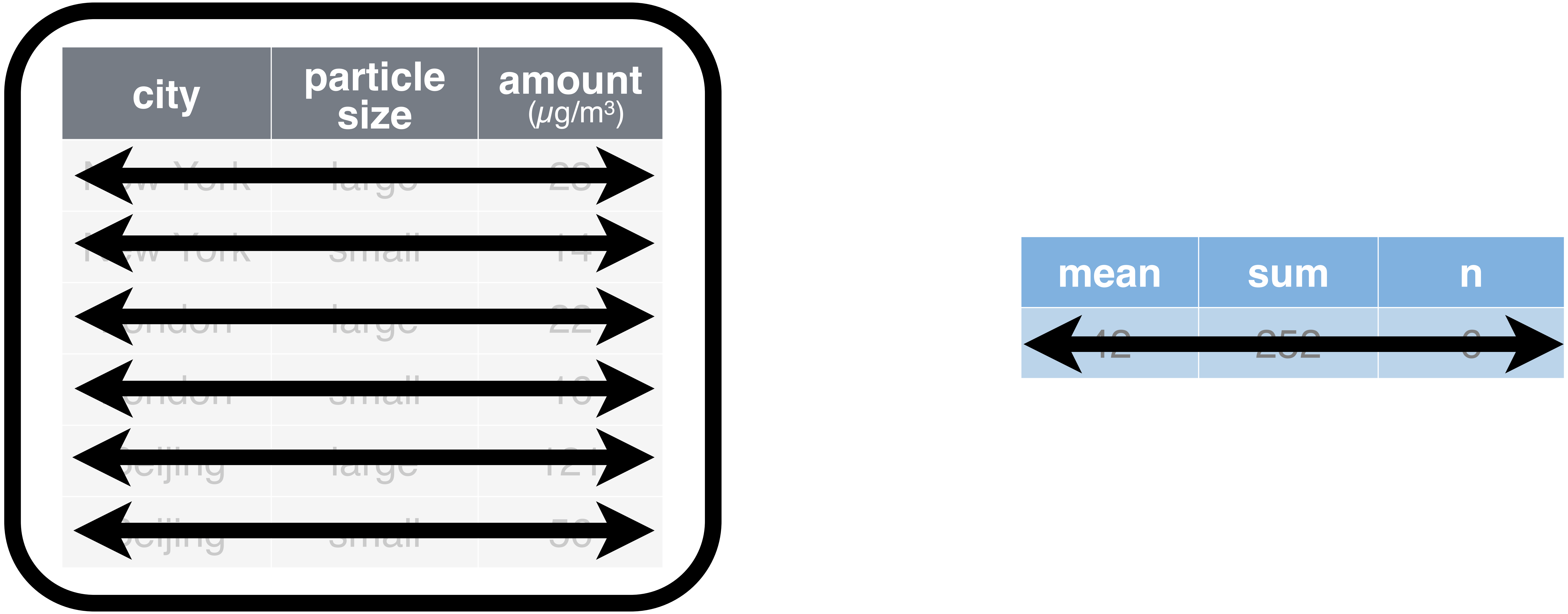
city	particle size	amount (μg/m³)
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

Mean = Sum / N

42 = 252 / 6



```
pollution %>% summarise(mean = mean(amount), sum = sum(amount), n = n())
```

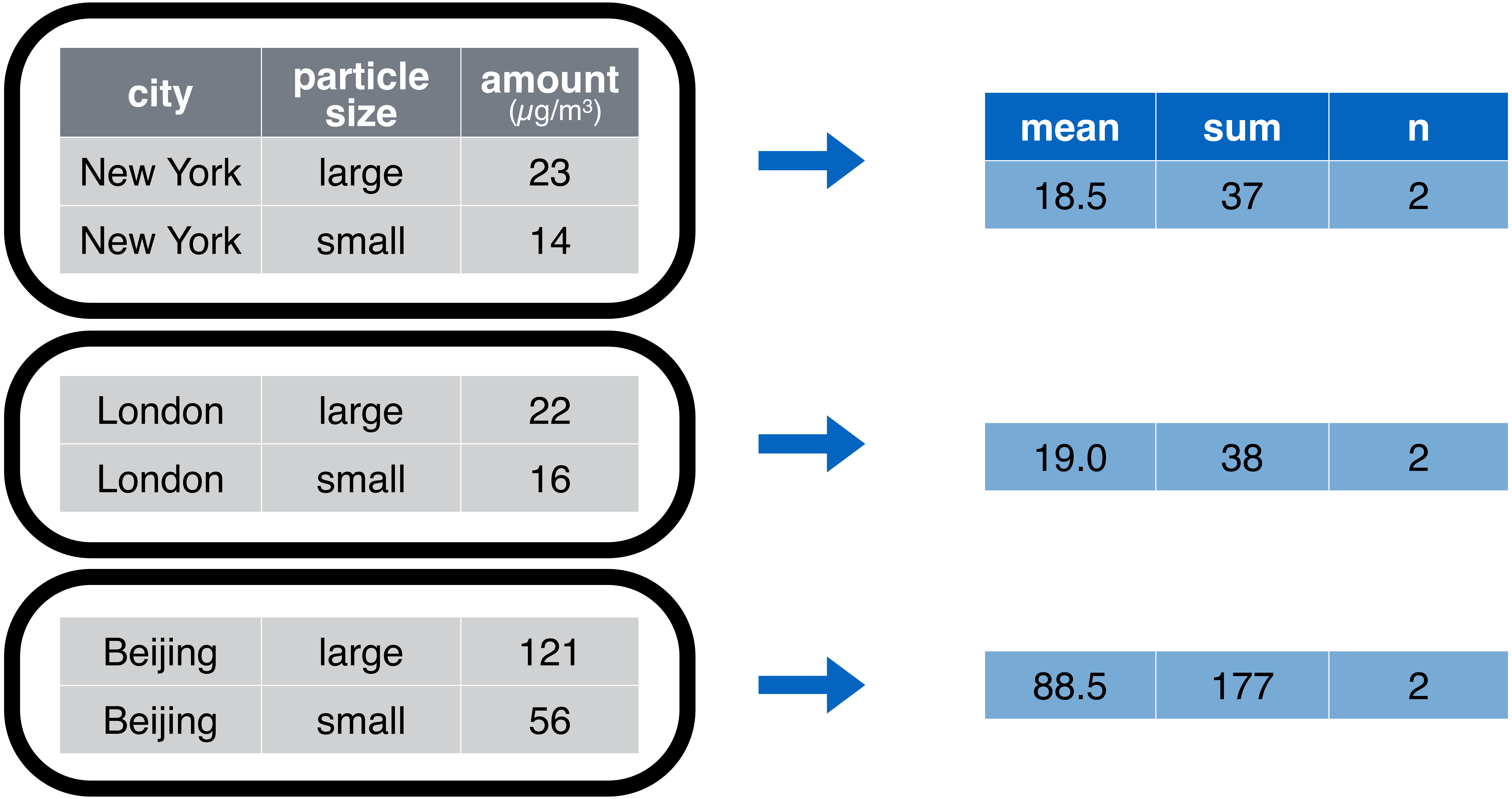


city	particle size	amount (μg/m³)
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

mean	sum	n
42	252	6

city	particle size	amount (μg/m^3)
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

mean	sum	n
42	252	6

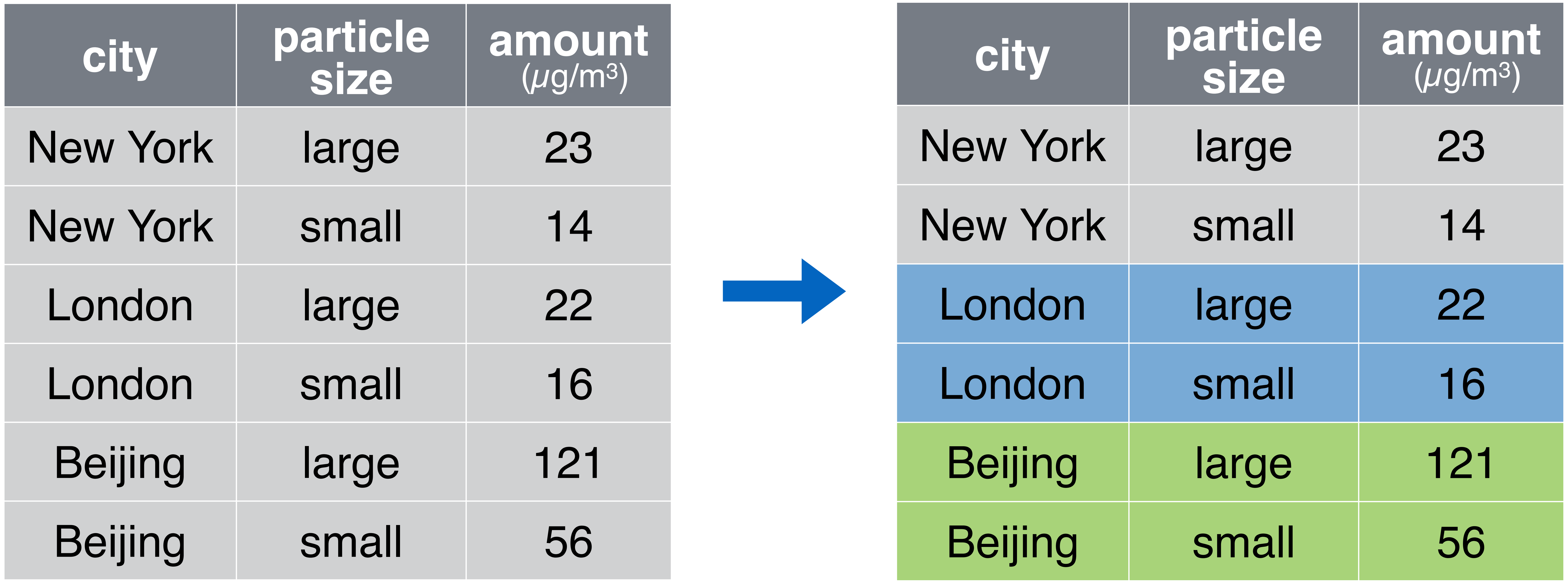


city	particle size	amount (μg/m^3)
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

mean	sum	n
18.5	37	2
19.0	38	2
88.5	177	2

group_by() + summarise()

group_by()



city	particle size	amount (μg/m^3)
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

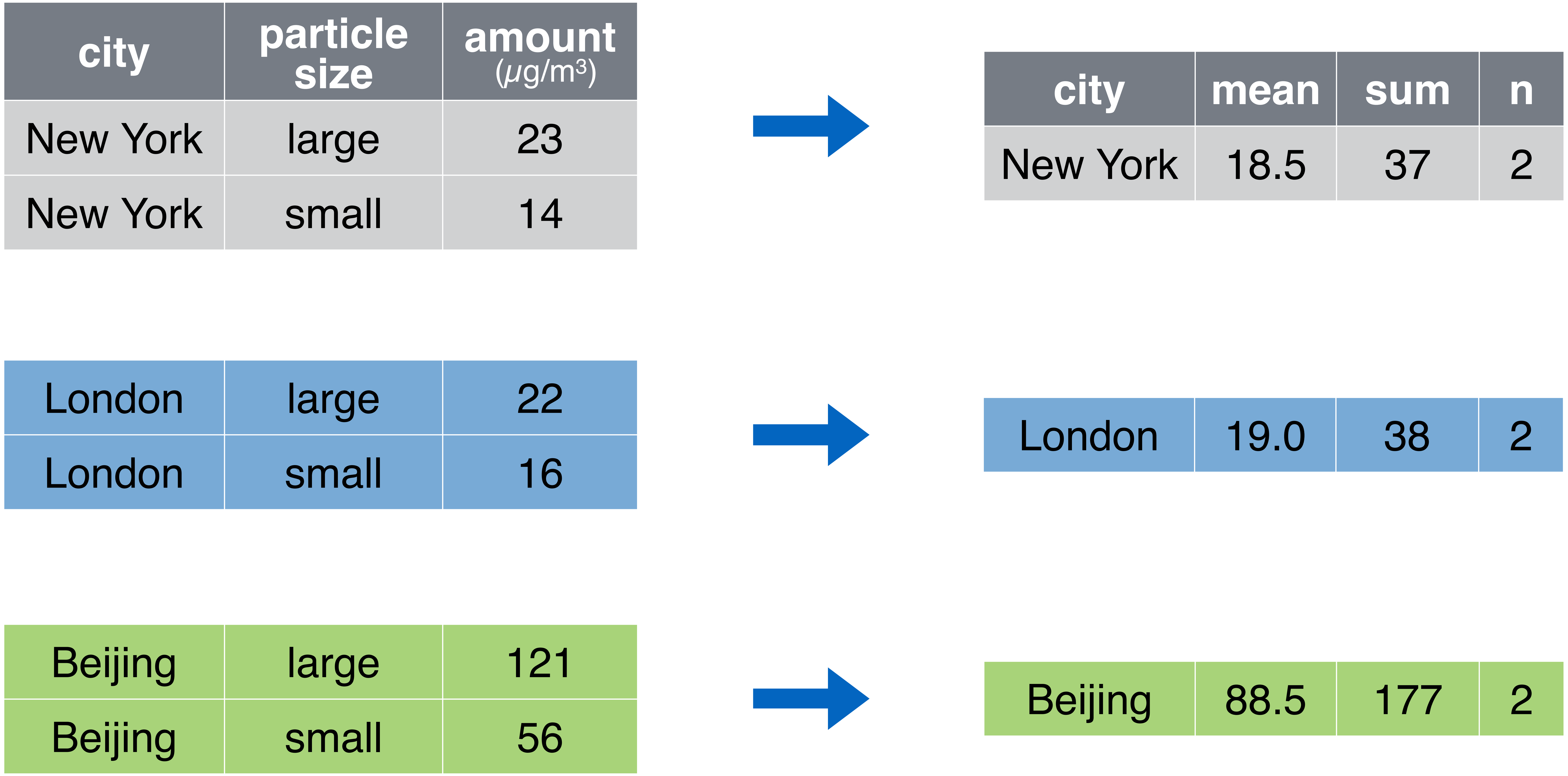
city	particle size	amount (μg/m^3)
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

```
pollution %>% group_by(city)
pollution %>% group_by(city)
## Source: local data frame [6 x 3]
## Groups: city
##
##       city  size amount
## 1 New York large     23
## 2 New York small     14
## 3   London large     22
## 4   London small     16
## 5  Beijing large    121
## 6  Beijing small     56
```

group_by() + summarise()

city	particle size	amount ($\mu g/m^3$)
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

```
pollution %>% group_by(city) %>%
  summarise(mean = mean(amount), sum = sum(amount), n = n())
```



city	particle size	amount (μg/m³)
New York	large	23
New York	small	14

city	mean	sum	n
New York	18.5	37	2

London	large	22
London	small	16

London	19.0	38	2

Beijing	large	121
Beijing	small	56

Beijing	88.5	177	2

```
pollution %>% group_by(city) %>%
  summarise(mean = mean(amount), sum = sum(amount), n = n())
```

city	particle size	amount (μg/m^3)
New York	large	23
New York	small	14

London	large	22
London	small	16

Beijing	large	121
Beijing	small	56

city	mean	sum	n
New York	18.5	37	2

city	mean	sum	n
New York	18.5	37	2
London	19.0	38	2
Beijing	88.5	177	2

Beijing	88.5	177	2

```
pollution %>% group_by(city) %>%
  summarise(mean = mean(amount), sum = sum(amount), n = n())
```

city	particle size	amount ($\mu g/m^3$)
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

city	mean	sum	n
New York	18.5	37	2
London	19.0	38	2
Beijing	88.5	177	2

```
pollution %>% group_by(city) %>%
  summarise(mean = mean(amount), sum = sum(amount), n = n())
```

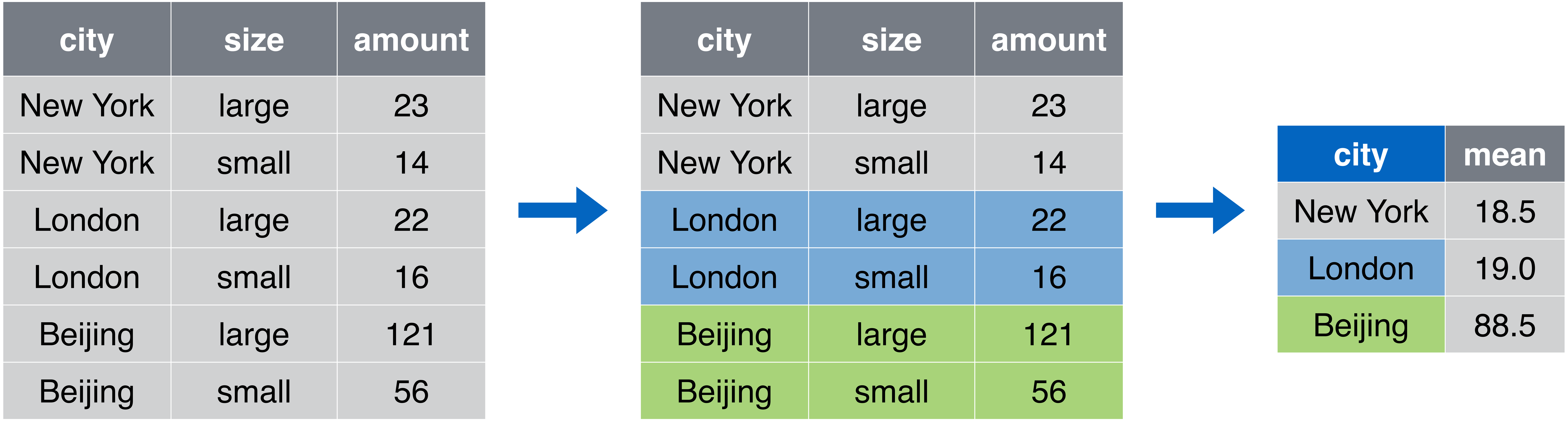
city	particle size	amount (μg/m^3)
New York	large	23
New York	small	14

London	large	22
London	small	16

Beijing	large	121
Beijing	small	56

city	mean	sum	n
New York	18.5	37	2
London	19.0	38	2
Beijing	88.5	177	2

```
pollution %>% group_by(city) %>%
  summarise(mean = mean(amount), sum = sum(amount), n = n())
```

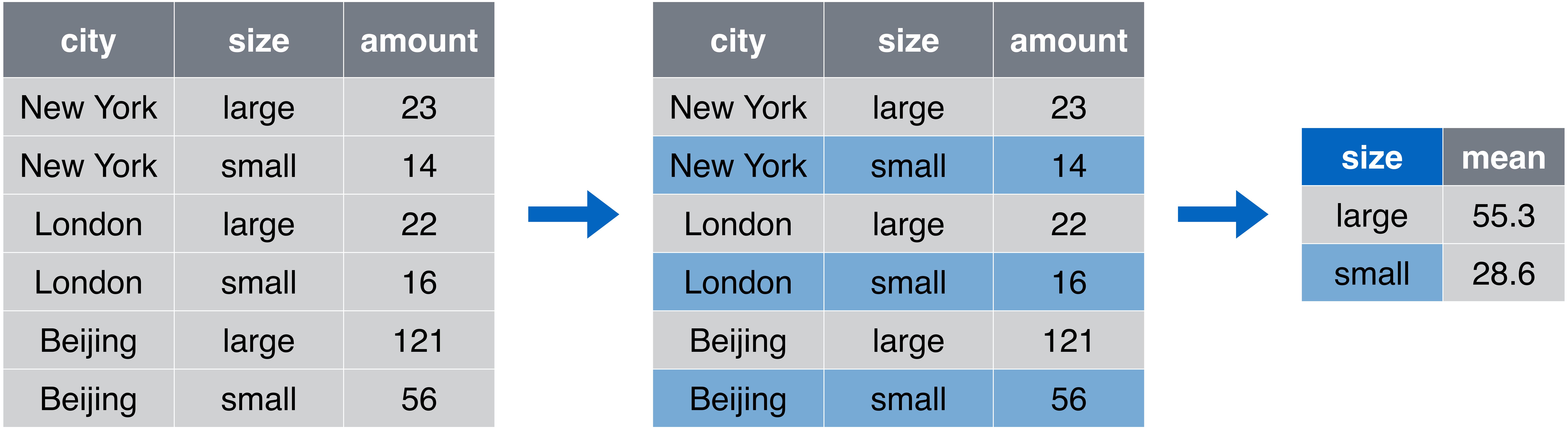


city	size	amount
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

city	size	amount
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

city	mean
New York	18.5
London	19.0
Beijing	88.5

```
pollution %>% group_by(city) %>%  summarise(mean = mean(amount))
```



city	size	amount
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

city	size	amount
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

size	mean
large	55.3
small	28.6

```
pollution %>% group_by(size) %>%  summarise(mean = mean(amount))
```

Your Turn

Filter out observations where **arr_delay** equals NA. Then use `group_by()` and `summarise()` to calculate avg_delay, the mean **arr_delay** by **carrier**.

Save your new data as **delays**. We will use it again soon.

flights (336,776 x 16)

year	month	day	dep_time	dep_delay	arr_time	arr_delay	carrier	tailnum	flight	origin	dest	air_time	distance	hour	minute
2013	1	1	517	2	830	11	UA	N14228	1545	EWR	IAH	227	1400	5	17
2013	1	1	533	4	850	20	UA	N24211	1714	LGA	IAH	227	1416	5	33

```
delays <- flights %>%
  filter(!is.na(arr_delay)) %>%
  group_by(carrier) %>%
  summarise(avg_delay = mean(arr_delay))
```

(16 x 2)

carrier	avg_delay
9E	7.38
AA	0.36
AS	-9.93
B6	9.95
DL	1.64

ungroup()

city	particle size	amount (μg/m^3)
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

→

city	particle size	amount (μg/m^3)
New York	large	23
New York	small	14
London	large	22
London	small	16
Beijing	large	121
Beijing	small	56

```
pollution %>% ungroup()
```

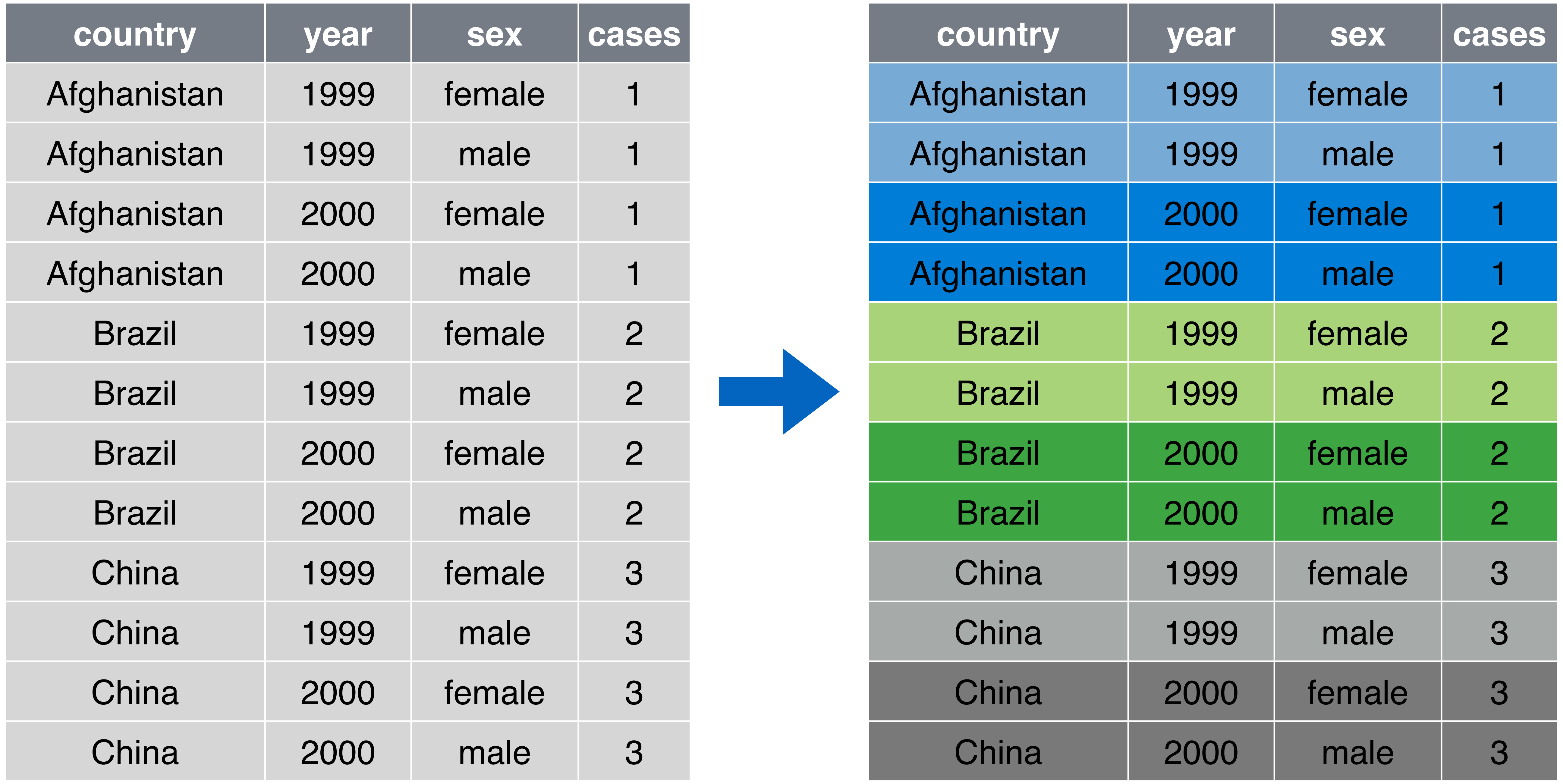
Combinations

1 Pass **group_by()** multiple variables to group by combinations of values

2 **Summarise()** will remove the rightmost grouping variable

country	year	sex	cases
Afghanistan	1999	female	1
Afghanistan	1999	male	1
Afghanistan	2000	female	1
Afghanistan	2000	male	1
Brazil	1999	female	2
Brazil	1999	male	2
Brazil	2000	female	2
Brazil	2000	male	2
China	1999	female	3
China	1999	male	3
China	2000	female	3
China	2000	male	3

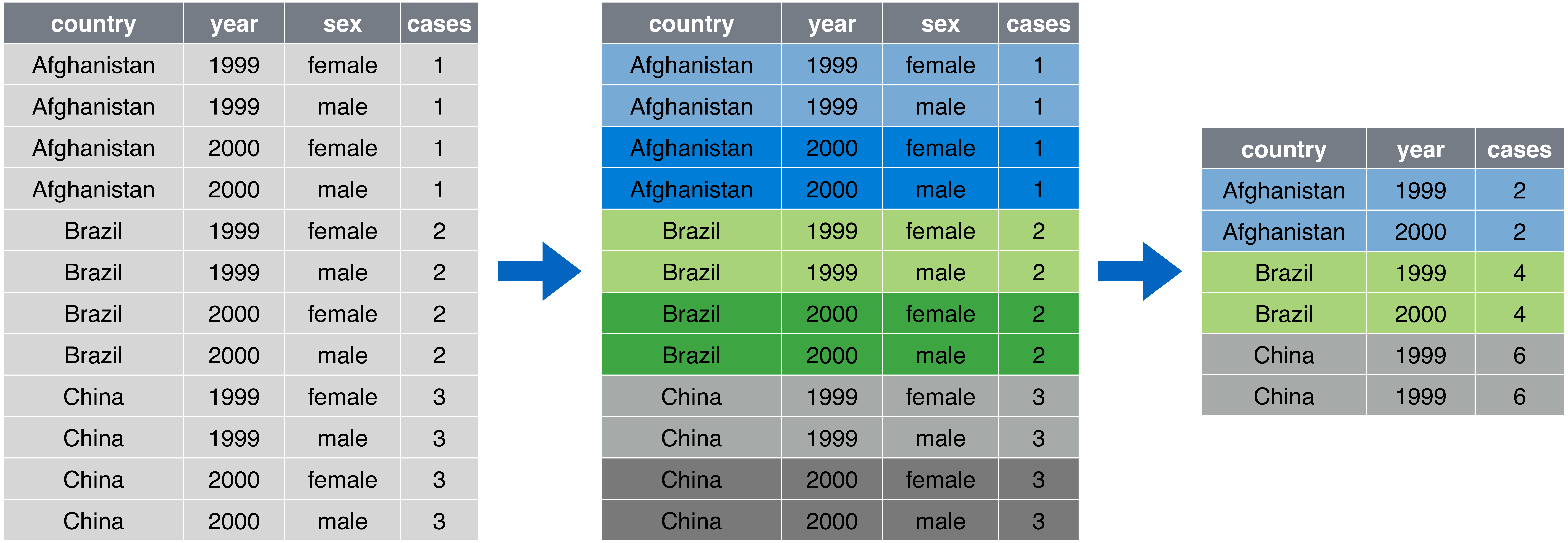
toyb



country	year	sex	cases
Afghanistan	1999	female	1
Afghanistan	1999	male	1
Afghanistan	2000	female	1
Afghanistan	2000	male	1
Brazil	1999	female	2
Brazil	1999	male	2
Brazil	2000	female	2
Brazil	2000	male	2
China	1999	female	3
China	1999	male	3
China	2000	female	3
China	2000	male	3

country	year	sex	cases
Afghanistan	1999	female	1
Afghanistan	1999	male	1
Afghanistan	2000	female	1
Afghanistan	2000	male	1
Brazil	1999	female	2
Brazil	1999	male	2
Brazil	2000	female	2
Brazil	2000	male	2
China	1999	female	3
China	1999	male	3
China	2000	female	3
China	2000	male	3

```
toyb %>%
  group_by(country, year)
```

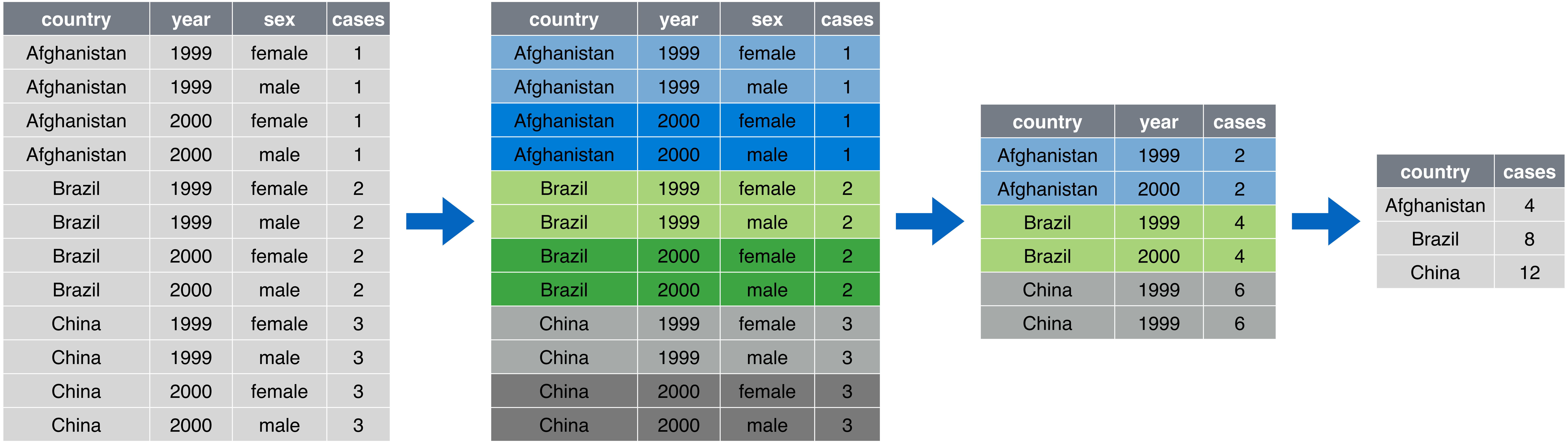


country	year	sex	cases
Afghanistan	1999	female	1
Afghanistan	1999	male	1
Afghanistan	2000	female	1
Afghanistan	2000	male	1
Brazil	1999	female	2
Brazil	1999	male	2
Brazil	2000	female	2
Brazil	2000	male	2
China	1999	female	3
China	1999	male	3
China	2000	female	3
China	2000	male	3

country	year	sex	cases
Afghanistan	1999	female	1
Afghanistan	1999	male	1
Afghanistan	2000	female	1
Afghanistan	2000	male	1
Brazil	1999	female	2
Brazil	1999	male	2
Brazil	2000	female	2
Brazil	2000	male	2
China	1999	female	3
China	1999	male	3
China	2000	female	3
China	2000	male	3

country	year	cases
Afghanistan	1999	2
Afghanistan	2000	2
Brazil	1999	4
Brazil	2000	4
China	1999	6
China	1999	6

```
toyb %>%
  group_by(country, year) %>%
  summarise(cases = sum(cases))
```



country	year	sex	cases
Afghanistan	1999	female	1
Afghanistan	1999	male	1
Afghanistan	2000	female	1
Afghanistan	2000	male	1
Brazil	1999	female	2
Brazil	1999	male	2
Brazil	2000	female	2
Brazil	2000	male	2
China	1999	female	3
China	1999	male	3
China	2000	female	3
China	2000	male	3

country	year	sex	cases
Afghanistan	1999	female	1
Afghanistan	1999	male	1
Afghanistan	2000	female	1
Afghanistan	2000	male	1
Brazil	1999	female	2
Brazil	1999	male	2
Brazil	2000	female	2
Brazil	2000	male	2
China	1999	female	3
China	1999	male	3
China	2000	female	3
China	2000	male	3

country	year	cases
Afghanistan	1999	2
Afghanistan	2000	2
Brazil	1999	4
Brazil	2000	4
China	1999	6
China	1999	6

country	cases
Afghanistan	4
Brazil	8
China	12

```
toyb %>%
  group_by(country, year) %>%
  summarise(cases = sum(cases)) %>%
  summarise(cases = sum(cases))
```

Your Turn

For each **origin**, calculate the total number of flights to each **destination**.

Will the results be "grouped?" How can you check?

Which variable(s) will they be grouped on?

How can you ensure that the results are *not* grouped?

flights (336,776 x 16)

year	month	day	dep_time	dep_delay	arr_time	arr_delay	carrier	tailnum	flight	origin	dest	air_time	distance	hour	minute
2013	1	1	517	2	830	11	UA	N14228	1545	EWR	IAH	227	1400	5	17
2013	1	1	533	4	850	20	UA	N24211	1714	LGA	IAH	227	1416	5	33

```
flights %>%
  filter(!is.na(arr_delay)) %>%
  group_by(origin, dest) %>%
  summarise(n = n())
```

(223 x 3)

origin	dest	n
EWR	ALB	418
EWR	ANC	8
EWR	ATL	4876
EWR	AUS	957
EWR	AVL	251

```
flights %>%
  filter(!is.na(arr_delay)) %>%
  group_by(origin, dest) %>%
  summarise(n = n())
```

```
## Source: local data frame [223 x 3]
## Groups: origin

##    origin dest    n
## 1     EWR  ALB  418
## 2     EWR  ANC    8
## 3     EWR  ATL 4876
## 4     EWR  AUS  957
```

```
flights %>%
  filter(!is.na(arr_delay)) %>%
  group_by(origin, dest) %>%
  summarise(n = n()) %>%
  ungroup()
```

```
## Source: local data frame [223 x 3]

##    origin dest    n
## 1     EWR  ALB  418
## 2     EWR  ANC    8
## 3     EWR  ATL 4876
## 4     EWR  AUS  957
```

Re-arrange observations

arrange()

storms

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

→

storm	wind	pressure	date
Ana	40	1013	1997-07-01
Alex	45	1009	1998-07-30
Arthur	45	1010	1996-06-21
Arlene	50	1010	1999-06-13
Allison	65	1005	1995-06-04
Alberto	110	1007	2000-08-12

```
storms %>% arrange(wind)
```

arrange()

storms

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

storm	wind	pressure	date
Ana	40	1013	1997-07-01
Alex	45	1009	1998-07-30
Arthur	45	1010	1996-06-21
Arlene	50	1010	1999-06-13
Allison	65	1005	1995-06-04
Alberto	110	1007	2000-08-12

```
storms %>% arrange(wind)
```

arrange()

storms

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Allison	65	1005	1995-06-04
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21
Alex	45	1009	1998-07-30
Ana	40	1013	1997-07-01

```
storms %>% arrange(desc(wind))
```

arrange()

storms

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

→

storm	wind	pressure	date
Ana	40	1013	1997-07-01
Alex	45	1009	1998-07-30
Arthur	45	1010	1996-06-21
Arlene	50	1010	1999-06-13
Allison	65	1005	1995-06-04
Alberto	110	1007	2000-08-12

```
storms %>% arrange(wind)
```

arrange()

storms

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

→

storm	wind	pressure	date
Ana	40	1013	1997-07-01
Arthur	45	1010	1996-06-21
Alex	45	1009	1998-07-30
Arlene	50	1010	1999-06-13
Allison	65	1005	1995-06-04
Alberto	110	1007	2000-08-12

```
storms %>% arrange(wind, date)
```

Your Turn

Rearrange your **delays** data set so it lists carriers from the carrier with the largest average delay to the carrier with the smallest average delay.

delays (16 x 2)

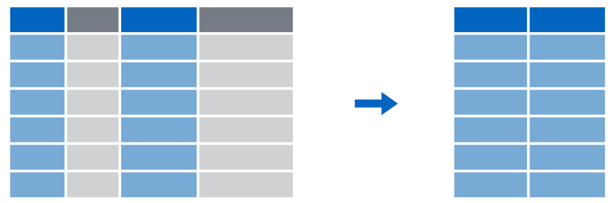
carrier	avg_delay
9E	7.38
AA	0.36
AS	-9.93
B6	9.95
DL	1.64
EV	15.80
F9	21.92

```
delays %>% arrange(desc(avg_delay))
```

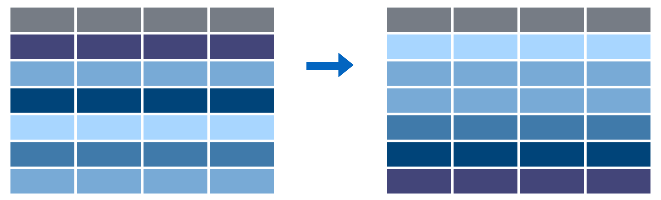
(16 x 2)

carrier	avg_delay
F9	21.92
FL	20.12
EV	15.80
YV	15.56
OO	11.93

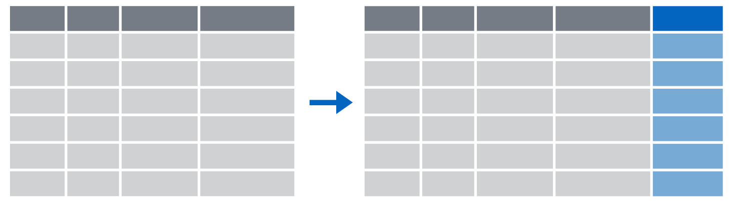
Recap: Information



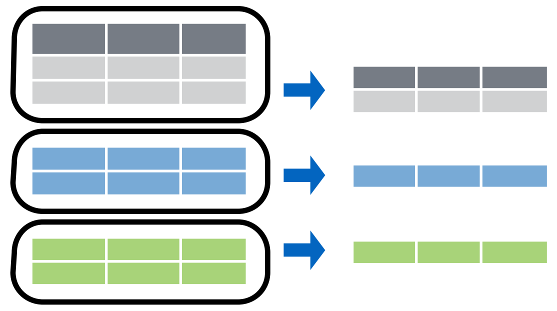
Extract variables and observations with **select()** and **filter()**



Arrange observations, with **arrange()**.



Make new variables, with **mutate()**.



Make groupwise observations with **group_by()** and **summarise()**.

Case Study 1:
TB counts

tb

Tuberculosis cases by country collected by the WHO for the *Global Tuberculosis Report*

```
library(EDAWR)
?tb
```

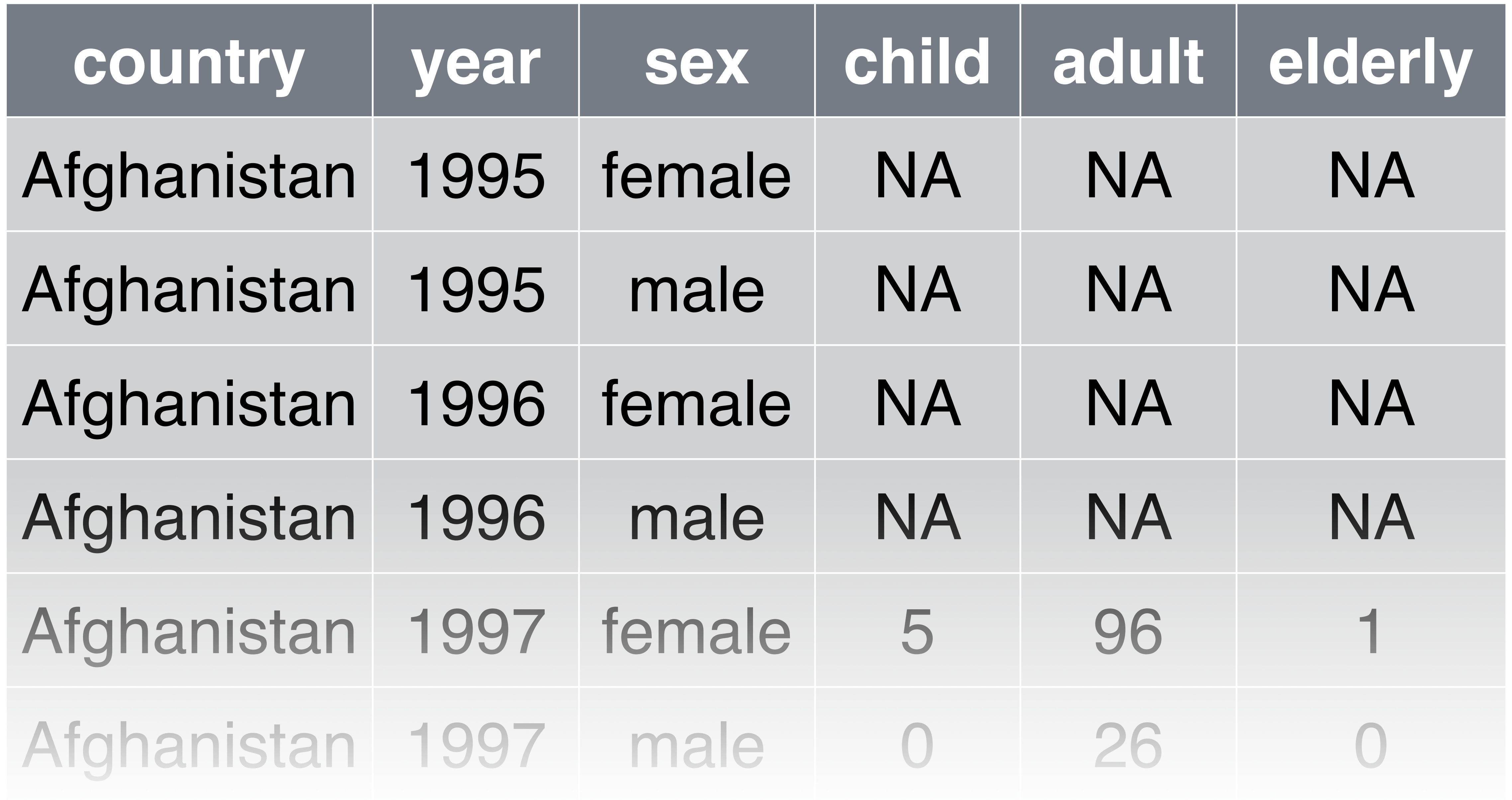
World Health Organization

tb

(3,800 x 6)

Number of cases reported by *country, year, sex* and *age* group

```
View(tb)
```



country	year	sex	child	adult	elderly
Afghanistan	1995	female	NA	NA	NA
Afghanistan	1995	male	NA	NA	NA
Afghanistan	1996	female	NA	NA	NA
Afghanistan	1996	male	NA	NA	NA
Afghanistan	1997	female	5	96	1
Afghanistan	1997	male	0	26	0

Goal

Number of cases reported by *country* and *year*

(3,800 x 6)

country	year	sex	child	adult	elderly
Afghanistan	1995	female	NA	NA	NA
Afghanistan	1995	male	NA	NA	NA
Afghanistan	1996	female	NA	NA	NA
Afghanistan	1996	male	NA	NA	NA
Afghanistan	1997	female	5	96	1
Afghanistan	1997	male	0	26	0

→

(1,691 x 3)

country	year	cases
Afghanistan	1997	128
Afghanistan	1998	1778
Afghanistan	1999	745
Afghanistan	2000	2666
Afghanistan	2001	4639
Afghanistan	2002	6509

Your Turn

Use some or all of:

filter()
select()
mutate()
summarise()
group_by()
arrange()

to calculate the total number of cases per country per year. Remove rows where the cases column contains an NA.

(3,800 x 6)

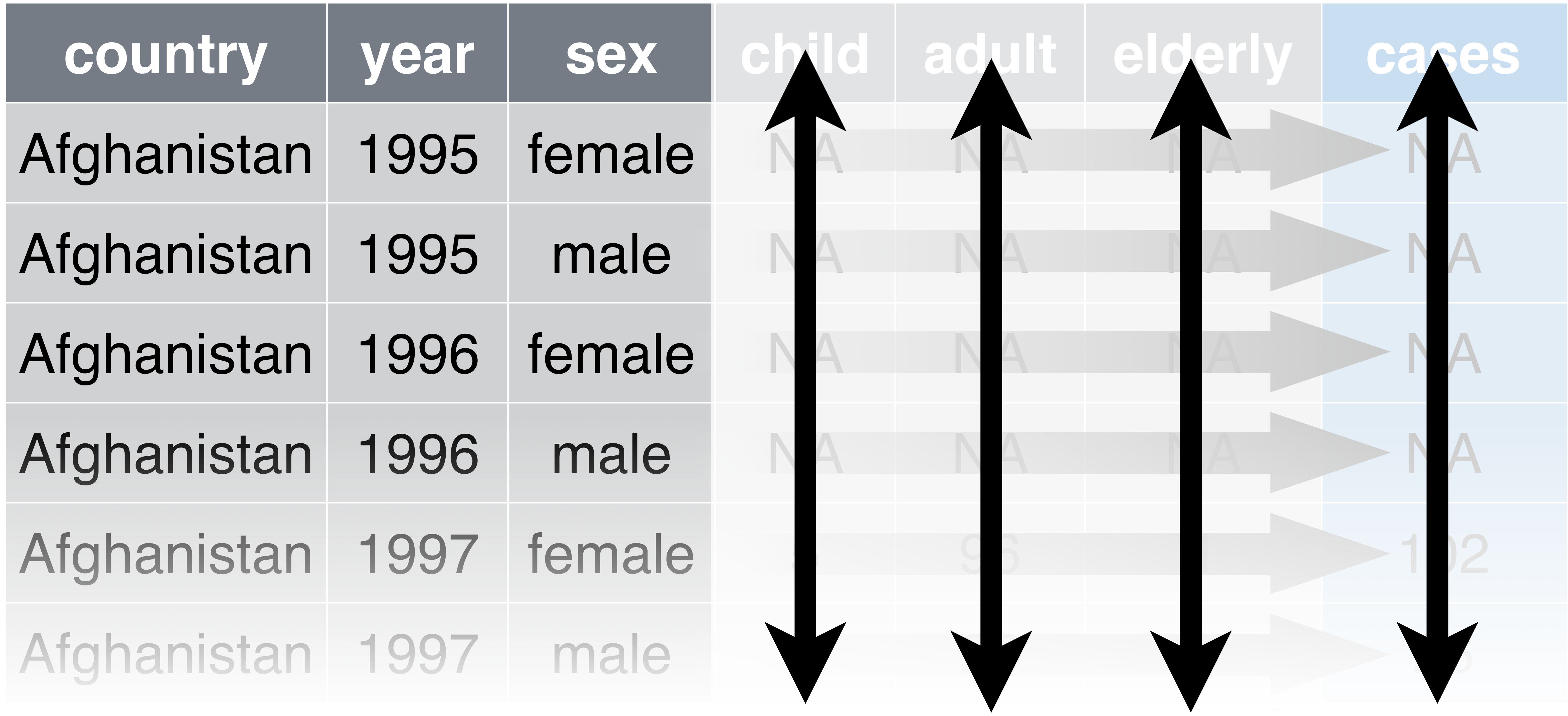
country	year	sex	child	adult	elderly
Afghanistan	1995	female	NA	NA	NA
Afghanistan	1995	male	NA	NA	NA
Afghanistan	1996	female	NA	NA	NA

→

(1,691 x 3)

country	year	cases
Afghanistan	1997	128
Afghanistan	1998	1778
Afghanistan	1999	745

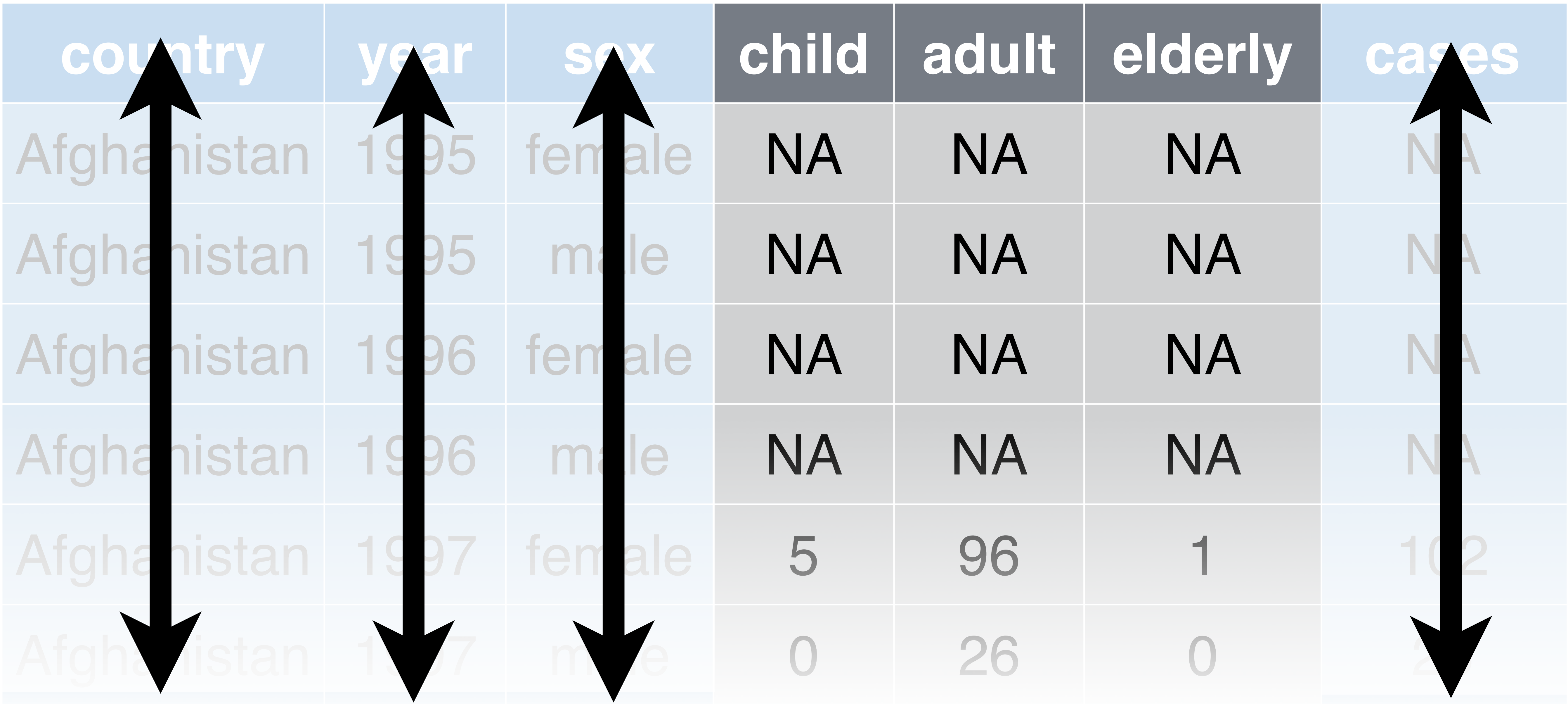
Step 1 - Combine child, adult, and elderly



country	year	sex	child	adult	elderly	cases
Afghanistan	1995	female	NA	NA	NA	NA
Afghanistan	1995	male	NA	NA	NA	NA
Afghanistan	1996	female	NA	NA	NA	NA
Afghanistan	1996	male	NA	NA	NA	NA
Afghanistan	1997	female	[illegible]	96	1	102
Afghanistan	1997	male	[illegible]	[illegible]	[illegible]	[illegible]

```
tb %>%
  mutate(cases = child + adult +
            elderly)
```

Step 2 - Select relevant variables



country	year	sex	child	adult	elderly	cases
Afghanistan	1995	female	NA	NA	NA	NA
Afghanistan	1995	male	NA	NA	NA	NA
Afghanistan	1996	female	NA	NA	NA	NA
Afghanistan	1996	male	NA	NA	NA	NA
Afghanistan	1997	female	5	96	1	102
Afghanistan	1997	male	0	26	0	26

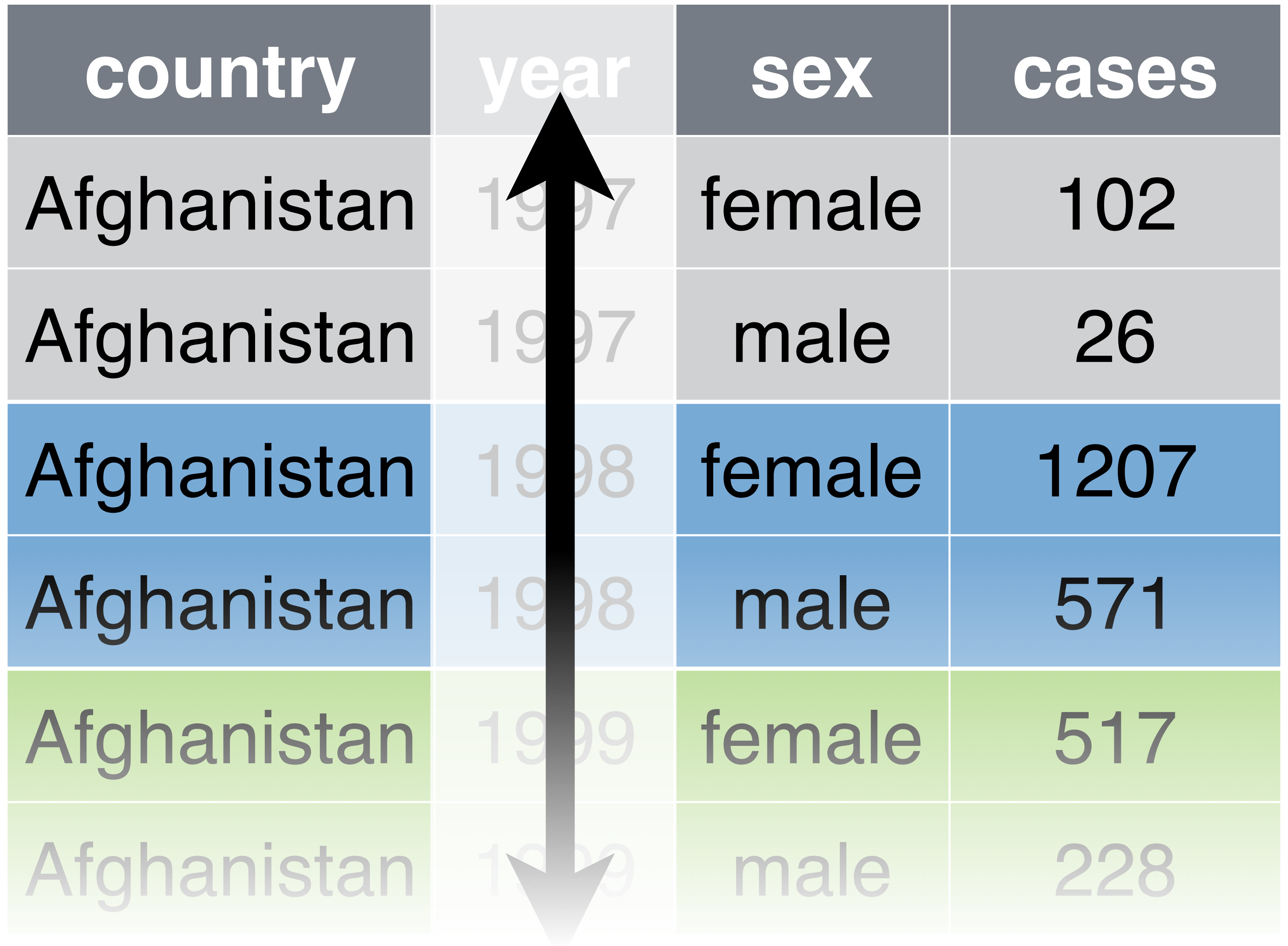
```
tb %>%
  mutate(cases = child + adult +
           elderly) %>%
  select(country:sex, cases)
```

Step 3 - Remove observations with NA's

country	year	sex	cases
Afghanistan	1995	female	NA
Afghanistan	1995	male	NA
Afghanistan	1996	female	NA
Afghanistan	1996	male	NA
Afghanistan	1997	female	102
Afghanistan	1997	male	26

```
tb %>%
  mutate(cases = child + adult +
           elderly) %>%
  select(country:sex, cases) %>%
  filter(!is.na(cases))
```

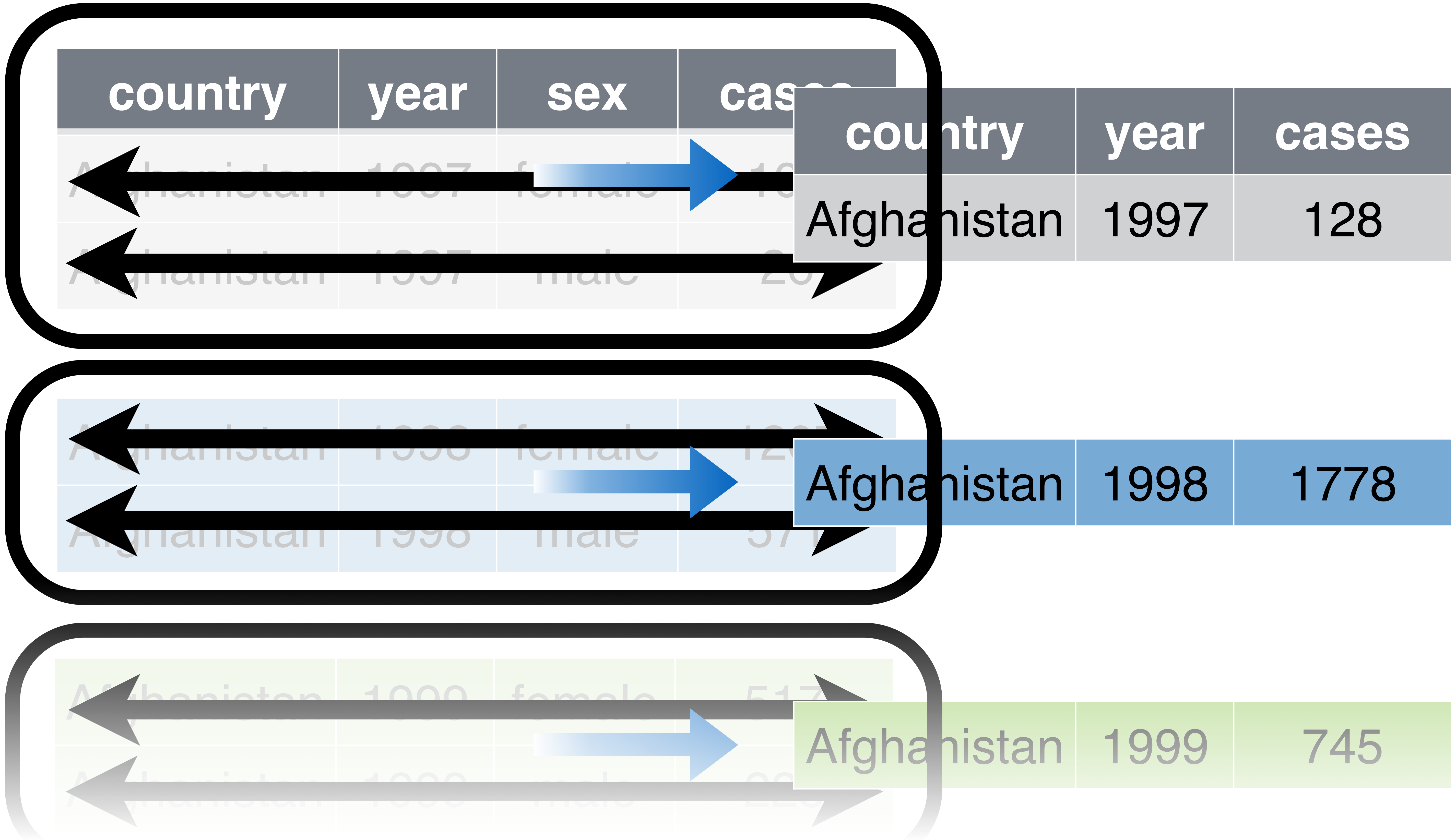
Step 4 - Group observations by year



country	year	sex	cases
Afghanistan	1997	female	102
Afghanistan	1997	male	26
Afghanistan	1998	female	1207
Afghanistan	1998	male	571
Afghanistan	1999	female	517
Afghanistan	1999	male	228

```
tb %>%
  mutate(cases = child + adult +
           elderly) %>%
  select(country:sex, cases) %>%
  filter(!is.na(cases)) %>%
  group_by(country, year)
```

Step 5 - Summarise total cases by year



country	year	cases
Afghanistan	1997	128
Afghanistan	1998	1778
Afghanistan	1999	745

```
tb %>%
  mutate(cases = child + adult +
           elderly) %>%
  select(country:sex, cases) %>%
  filter(!is.na(cases)) %>%
  group_by(country, year) %>%
  summarise(cases = sum(cases))
```

Step 6 - Ungroup results

country	year	cases
Afghanistan	1997	128
Afghanistan	1998	1778
Afghanistan	1999	745

```
tb %>%
  mutate(cases = child + adult +
                 elderly) %>%
  select(country:sex, cases) %>%
  filter(!is.na(cases)) %>%
  group_by(country, year) %>%
  summarise(cases = sum(cases)) %>%
  ungroup()
```

Data Science for Data Wranglers Part 2:

Units of analysis

$$F = MA$$

$$\begin{array}{|ccc|}\hline f_1 & = & m_1 \cdot a_1 \\ \hline f_2 & = & m_2 \cdot a_2 \\ \hline f_3 & = & m_3 \cdot a_3 \\ \hline\end{array}$$

Unit of Analysis - The combination of conditions that define an observation.

Observation - The values of several variables measured under similar conditions.

$$F = MA$$

particle 1	$f_1 = m_1 \cdot a_1$	particle 1 at time 1
particle 2	$f_2 = m_2 \cdot a_2$	particle 1 at time 2
particle 3	$f_3 = m_3 \cdot a_3$	particle 1 at time 3

Unit of Analysis - The combination of conditions that define an observation.

The original tb data set?

```
View(rawtb)
```

rawtb (949,316 x 5)

country	year	sex	age	n
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1

Practice: units of analysis

What is the unit of analysis?

rawtb (949,316 x 5)

country	year	sex	age	n
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1

00:10

Practice: units of analysis

What is the unit of analysis?

Individual people

rawtb (949,316 x 5)

country	year	sex	age	n
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1

Practice: units of analysis

What is the unit of analysis?

(895 x 5)

country	year	sex	age	n
Afghanistan	1997	female	adult	96
Afghanistan	1997	female	child	5
Afghanistan	1997	female	elderly	1
Afghanistan	1997	male	adult	25
Afghanistan	1997	male	child	1
Afghanistan	1998	female	elderly	1142
Afghanistan	1998	female	adult	45
Afghanistan	1998	female	child	20

00:10

Practice: units of analysis

What is the unit of analysis?

Groups of people, grouped by:
age, sex, year, and country

(895 x 5)

country	year	sex	age	n
Afghanistan	1997	female	adult	96
Afghanistan	1997	female	child	5
Afghanistan	1997	female	elderly	1
Afghanistan	1997	male	adult	25
Afghanistan	1997	male	child	1
Afghanistan	1998	female	elderly	1142
Afghanistan	1998	female	adult	45
Afghanistan	1998	female	child	20

Practice: units of analysis

What is the unit of analysis?

(306 x 4)

country	year	sex	n
Afghanistan	1997	female	102
Afghanistan	1997	male	26
Afghanistan	1998	female	1207
Afghanistan	1998	male	571
Afghanistan	1999	female	517
Afghanistan	1999	male	228
Afghanistan	2000	female	1751
Afghanistan	2000	male	915

00:10

Practice: units of analysis

What is the unit of analysis?

Groups of people, grouped by:
sex, year, and country

(306 x 4)

country	year	sex	n
Afghanistan	1997	female	102
Afghanistan	1997	male	26
Afghanistan	1998	female	1207
Afghanistan	1998	male	571
Afghanistan	1999	female	517
Afghanistan	1999	male	228
Afghanistan	2000	female	1751
Afghanistan	2000	male	915

Practice: units of analysis

What is the unit of analysis?

(153 x 3)

country	year	n
Afghanistan	1997	128
Afghanistan	1998	1778
Afghanistan	1999	745
Afghanistan	2000	2666
Afghanistan	2001	4639
Afghanistan	2002	6509
Afghanistan	2003	6528
Afghanistan	2004	8245

00:10

Practice: units of analysis

What is the unit of analysis?

Groups of people, grouped by:
year, and country

(153 x 3)

country	year	n
Afghanistan	1997	128
Afghanistan	1998	1778
Afghanistan	1999	745
Afghanistan	2000	2666
Afghanistan	2001	4639
Afghanistan	2002	6509
Afghanistan	2003	6528
Afghanistan	2004	8245

Practice: units of analysis

What is the unit of analysis?

(9 x 2)

country	n
Afghanistan	140225
Algeria	128119
Angola	308365
Argentina	117156
Azerbaijan	29965
Belarus	37185
Benin	48821
Botswana	71470

00:10

Practice: units of analysis

What is the unit of analysis?

Groups of people by country

(9 x 2)

country	n
Afghanistan	140225
Algeria	128119
Angola	308365
Argentina	117156
Azerbaijan	29965
Belarus	37185
Benin	48821
Botswana	71470

Practice: units of analysis

What is the unit of analysis?

(1 x 1)

n
949316

00:10

Practice: units of analysis

What is the unit of analysis?

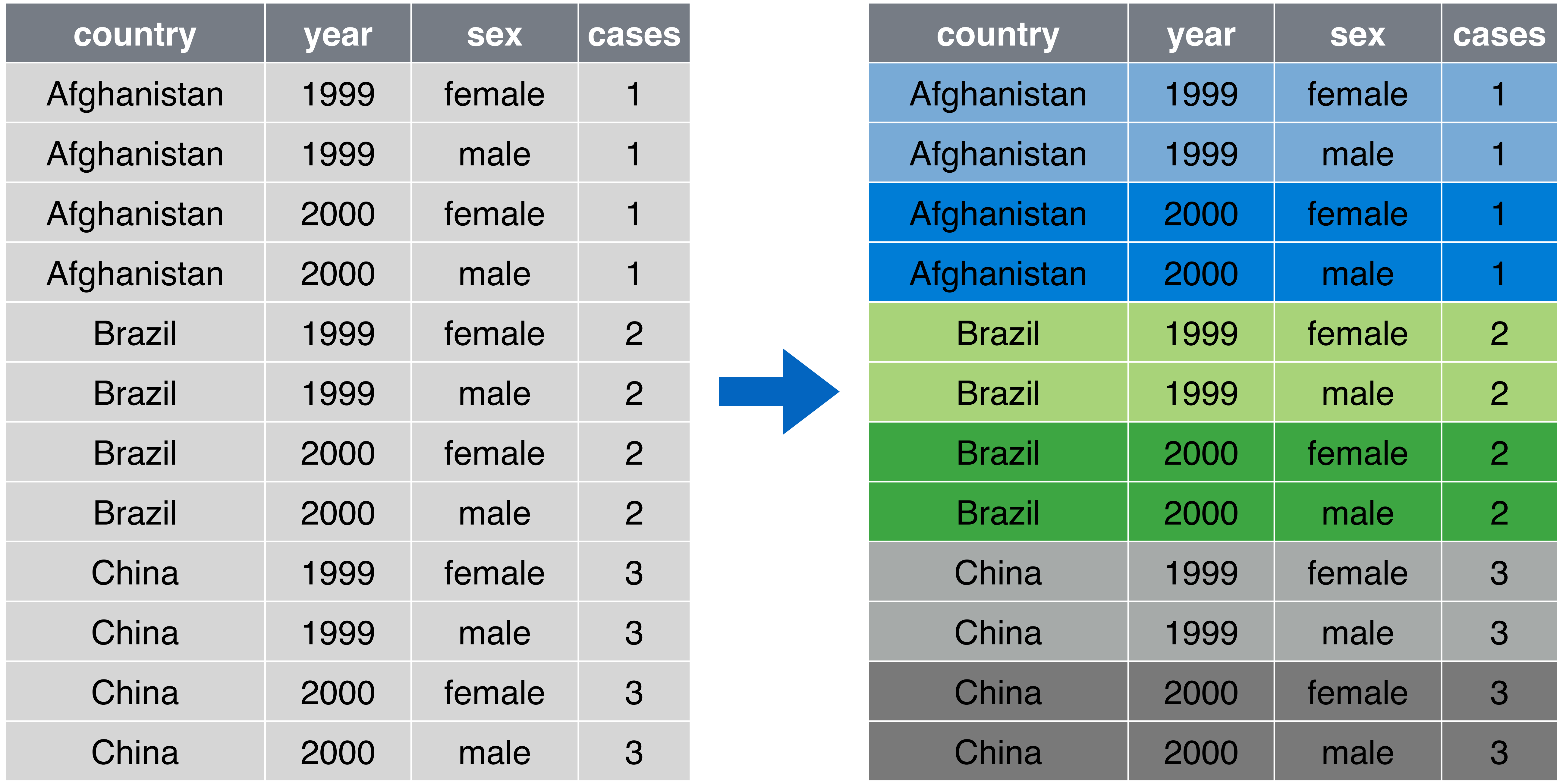
The group of all cases

(1 x 1)

n
949316

country	year	sex	cases
Afghanistan	1999	female	1
Afghanistan	1999	male	1
Afghanistan	2000	female	1
Afghanistan	2000	male	1
Brazil	1999	female	2
Brazil	1999	male	2
Brazil	2000	female	2
Brazil	2000	male	2
China	1999	female	3
China	1999	male	3
China	2000	female	3
China	2000	male	3

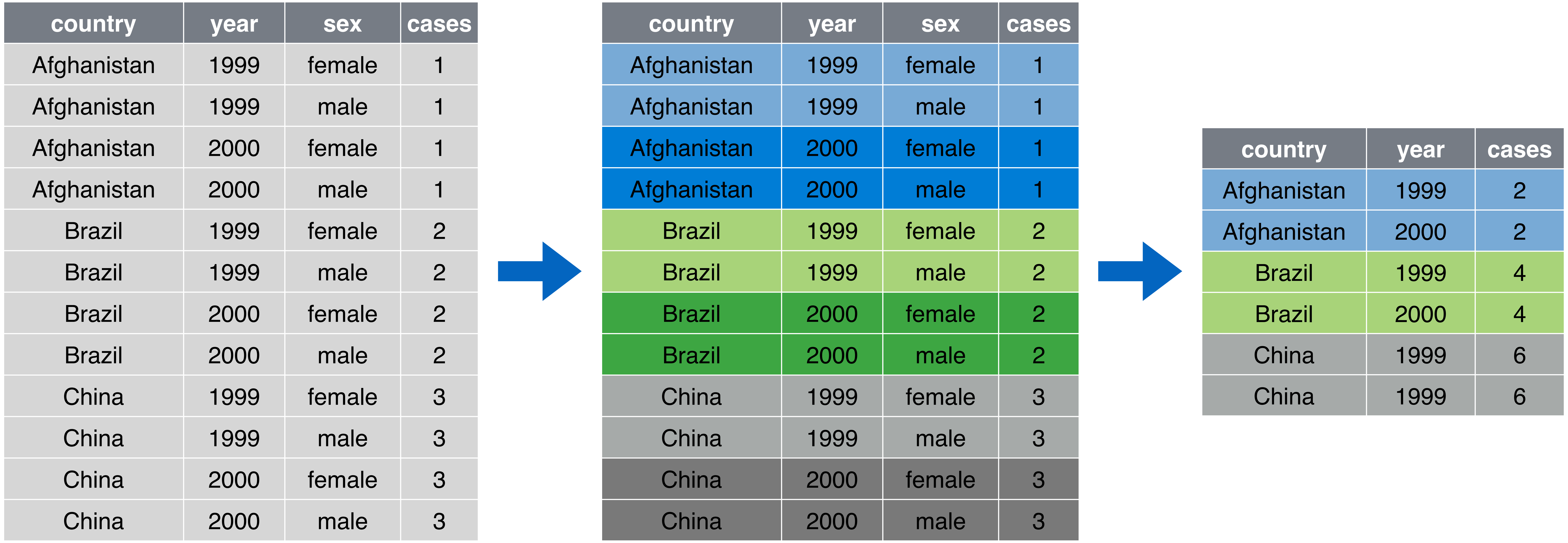
`toyb`



country	year	sex	cases
Afghanistan	1999	female	1
Afghanistan	1999	male	1
Afghanistan	2000	female	1
Afghanistan	2000	male	1
Brazil	1999	female	2
Brazil	1999	male	2
Brazil	2000	female	2
Brazil	2000	male	2
China	1999	female	3
China	1999	male	3
China	2000	female	3
China	2000	male	3

country	year	sex	cases
Afghanistan	1999	female	1
Afghanistan	1999	male	1
Afghanistan	2000	female	1
Afghanistan	2000	male	1
Brazil	1999	female	2
Brazil	1999	male	2
Brazil	2000	female	2
Brazil	2000	male	2
China	1999	female	3
China	1999	male	3
China	2000	female	3
China	2000	male	3

```
toyb %>%
  group_by(country, year)
```

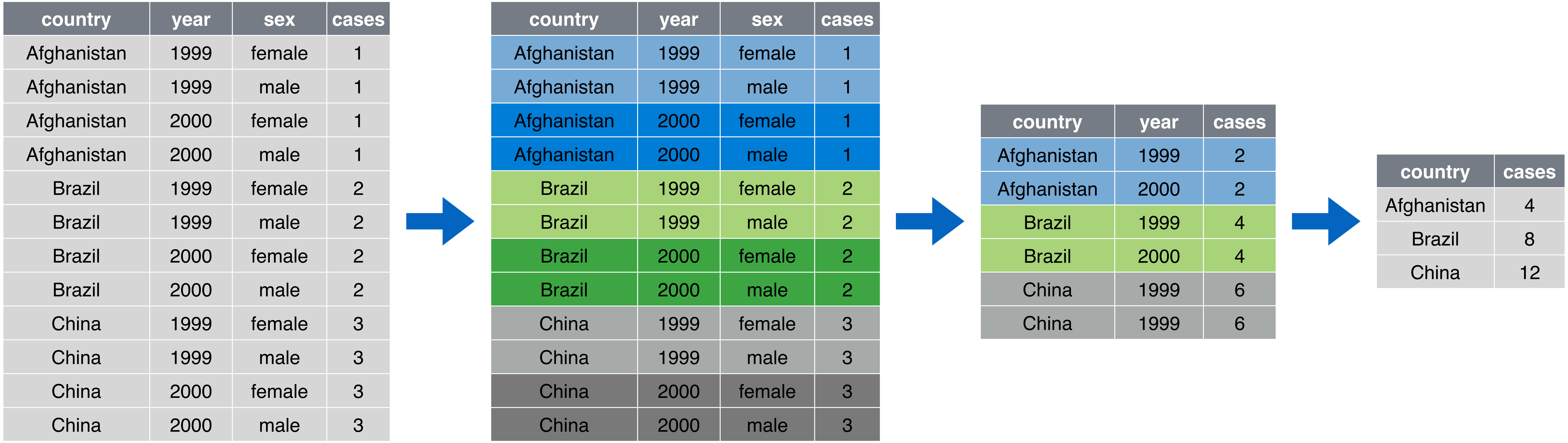


country	year	sex	cases
Afghanistan	1999	female	1
Afghanistan	1999	male	1
Afghanistan	2000	female	1
Afghanistan	2000	male	1
Brazil	1999	female	2
Brazil	1999	male	2
Brazil	2000	female	2
Brazil	2000	male	2
China	1999	female	3
China	1999	male	3
China	2000	female	3
China	2000	male	3

country	year	sex	cases
Afghanistan	1999	female	1
Afghanistan	1999	male	1
Afghanistan	2000	female	1
Afghanistan	2000	male	1
Brazil	1999	female	2
Brazil	1999	male	2
Brazil	2000	female	2
Brazil	2000	male	2
China	1999	female	3
China	1999	male	3
China	2000	female	3
China	2000	male	3

country	year	cases
Afghanistan	1999	2
Afghanistan	2000	2
Brazil	1999	4
Brazil	2000	4
China	1999	6
China	1999	6

```
toyb %>%
  group_by(country, year) %>%
  summarise(cases = sum(cases))
```



country	year	sex	cases
Afghanistan	1999	female	1
Afghanistan	1999	male	1
Afghanistan	2000	female	1
Afghanistan	2000	male	1
Brazil	1999	female	2
Brazil	1999	male	2
Brazil	2000	female	2
Brazil	2000	male	2
China	1999	female	3
China	1999	male	3
China	2000	female	3
China	2000	male	3

country	year	sex	cases
Afghanistan	1999	female	1
Afghanistan	1999	male	1
Afghanistan	2000	female	1
Afghanistan	2000	male	1
Brazil	1999	female	2
Brazil	1999	male	2
Brazil	2000	female	2
Brazil	2000	male	2
China	1999	female	3
China	1999	male	3
China	2000	female	3
China	2000	male	3

country	year	cases
Afghanistan	1999	2
Afghanistan	2000	2
Brazil	1999	4
Brazil	2000	4
China	1999	6
China	1999	6

country	cases
Afghanistan	4
Brazil	8
China	12

```
toyb %>%
  group_by(country, year) %>%
  summarise(cases = sum(cases)) %>%
  summarise(cases = sum(cases))
```

Hierarchy of information

country	year	sex	cases
Afghanistan	1999	female	1
Afghanistan	1999	male	1
Afghanistan	2000	female	1
Afghanistan	2000	male	1
Brazil	1999	female	2
Brazil	1999	male	2
Brazil	2000	female	2
Brazil	2000	male	2
China	1999	female	3
China	1999	male	3
China	2000	female	3
China	2000	male	3

country	year	cases
Afghanistan	1999	2
Afghanistan	2000	2
Brazil	1999	4
Brazil	2000	4
China	1999	6
China	2000	6

country	cases
Afghanistan	4
Brazil	8
China	12

cases
24

Larger units of analysis

Your Turn

Use dplyr functions to transform rawtb to the data set on the right.

Hint: Groups of people, grouped by: age, sex, year, and country

rawtb (949,316 x 5)

country	year	sex	age	n
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1

(895 x 5)

country	year	sex	age	n
Afghanistan	1997	female	adult	96
Afghanistan	1997	female	child	5
Afghanistan	1997	female	elderly	1
Afghanistan	1997	male	adult	25
Afghanistan	1997	male	child	1
Afghanistan	1998	female	elderly	1142
Afghanistan	1998	female	adult	45
Afghanistan	1998	female	child	20
Afghanistan	1998	male	elderly	500

```
rawtb %>%
  group_by(country, year, sex, age) %>%
  summarise(n = sum(n))
```

Your Turn

Use dplyr functions to transform rawtb to the data set on the right.

Hint: Groups of people, grouped by: age, sex, and year

rawtb (949,316 x 5)

country	year	sex	age	n
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1

→

(306 x 4)

country	year	sex	n
Afghanistan	1997	female	102
Afghanistan	1997	male	26
Afghanistan	1998	female	1207
Afghanistan	1998	male	571
Afghanistan	1999	female	517
Afghanistan	1999	male	228
Afghanistan	2000	female	1751
Afghanistan	2000	male	915
Afghanistan	2001	female	3062

```
rawtb %>%
  group_by(country, year, sex, age) %>%
  summarise(n = sum(n))
```

```
rawtb %>%
  group_by(country, year, sex) %>%
  summarise(n = sum(n))
```

Your Turn

Use dplyr functions to transform rawtb to the data set on the right.

rawtb (949,316 x 5)

country	year	sex	age	n
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1

→

(153 x 3)

country	year	n
Afghanistan	1997	128
Afghanistan	1998	1778
Afghanistan	1999	745
Afghanistan	2000	2666
Afghanistan	2001	4639
Afghanistan	2002	6509
Afghanistan	2003	6528
Afghanistan	2004	8245
Afghanistan	2005	9949

```
rawtb %>%
  group_by(country, year, sex, age) %>%
  summarise(n = sum(n))
```

```
rawtb %>%
  group_by(country, year, sex) %>%
  summarise(n = sum(n))
```

```
rawtb %>%
  group_by(country, year) %>%
  summarise(n = sum(n))
```

Your Turn

Use dplyr functions to transform rawtb to the data set on the right.

rawtb (949,316 x 5)

country	year	sex	age	n
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1

→

(9 x 2)

country	n
Afghanistan	140225
Algeria	128119
Angola	308365
Argentina	117156
Azerbaijan	29965
Belarus	37185
Benin	48821
Botswana	71470
Burundi	68010

```
rawtb %>%
  group_by(country, year, sex, age) %>%
  summarise(n = sum(n))

rawtb %>%
  group_by(country, year, sex) %>%
  summarise(n = sum(n))

rawtb %>%
  group_by(country, year) %>%
  summarise(n = sum(n))

rawtb %>%
  group_by(country) %>%
  summarise(n = sum(n))
```

Your Turn

Use dplyr functions to transform rawtb to the data set on the right.

rawtb (949,316 x 5)

country	year	sex	age	n
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1
Afghanistan	1997	female	adult	1

(1 x 1)

n
949316

```
rawtb %>%
  group_by(country, year, sex, age) %>%
  summarise(n = sum(n))

rawtb %>%
  group_by(country, year, sex) %>%
  summarise(n = sum(n))

rawtb %>%
  group_by(country, year) %>%
  summarise(n = sum(n))

rawtb %>%
  group_by(country) %>%
  summarise(n = sum(n))

rawtb %>%
  summarise(n = sum(n))
```

Data sets contain more information than they display

mutate()

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

→

storm	wind	pressure	date	ratio
Alberto	110	1007	2000-08-12	9.15
Alex	45	1009	1998-07-30	22.42
Allison	65	1005	1995-06-04	15.46
Ana	40	1013	1997-07-01	25.32
Arlene	50	1010	1999-06-13	20.20
Arthur	45	1010	1996-06-21	22.44

```
storms %>% mutate(ratio = pressure / wind)
```

Hierarchy of information

country	year	sex	cases
Afghanistan	1999	female	1
Afghanistan	1999	male	1
Afghanistan	2000	female	1
Afghanistan	2000	male	1
Brazil	1999	female	2
Brazil	1999	male	2
Brazil	2000	female	2
Brazil	2000	male	2
China	1999	female	3
China	1999	male	3
China	2000	female	3
China	2000	male	3

country	year	cases
Afghanistan	1999	2
Afghanistan	2000	2
Brazil	1999	4
Brazil	2000	4
China	1999	6
China	2000	6

country	cases
Afghanistan	4
Brazil	8
China	12

cases
24

Larger units of analysis