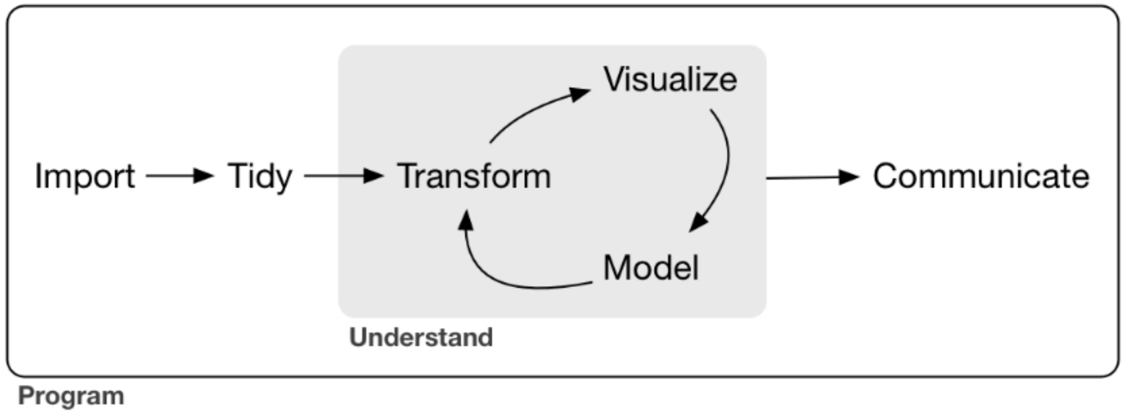


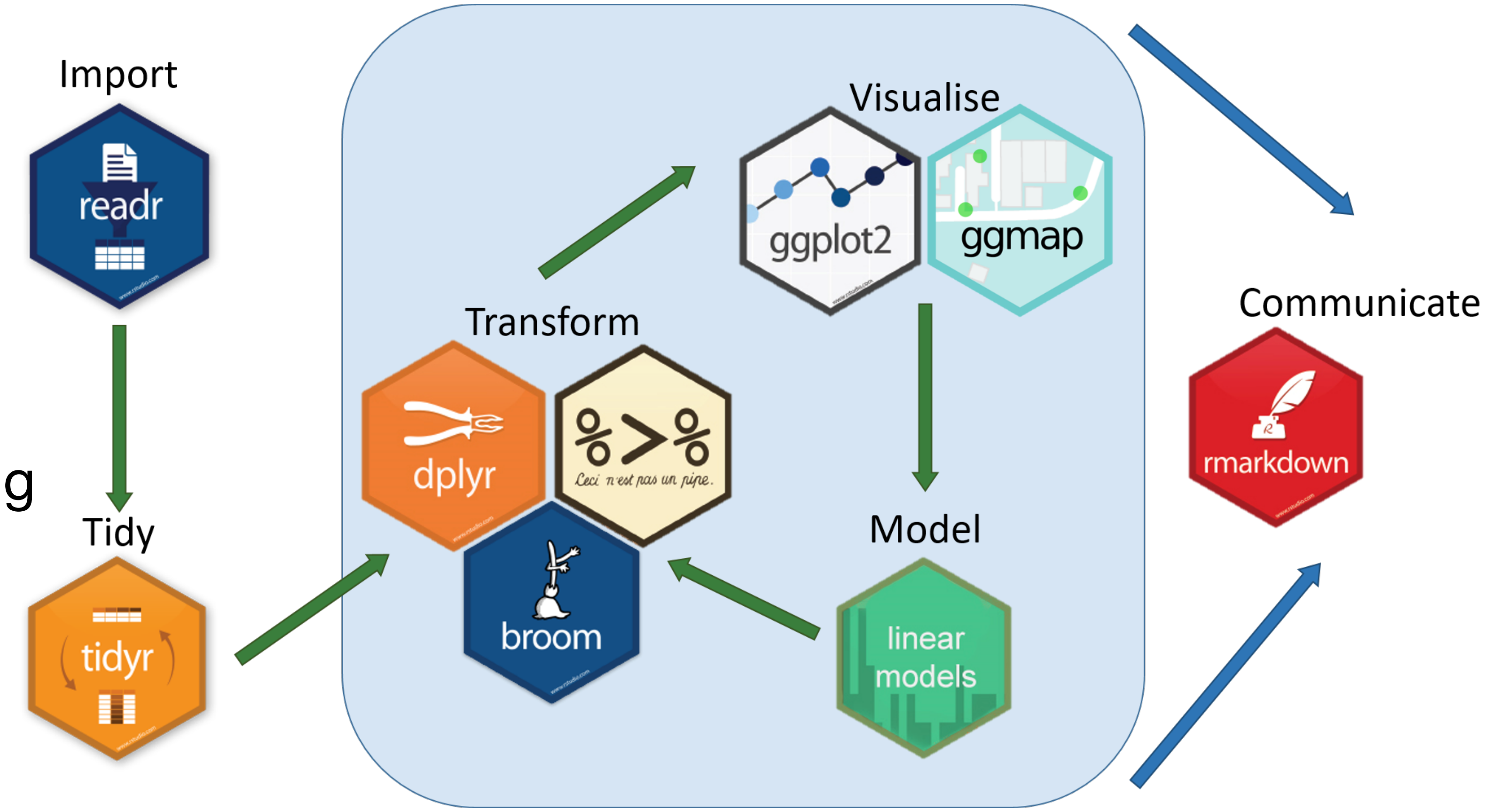
Data Science: Tidyverse

Alex Di Genova

16/05/2024

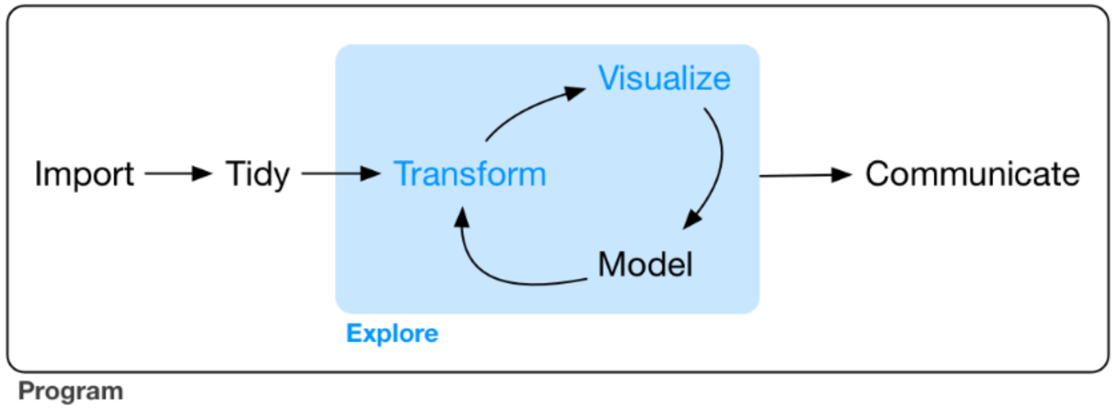
What is Tidyverse?

- A collection of R packages for data science
  - ggplot2 — data visualization
  - dplyr — data manipulation
  - tidyr — data tidying
  - readr — data import
  - purrr — functional programming
  - tibble — modern dataframes
  - stringr — string manipulation
  - forcats — factor handling
- Data science workflow (import, clean, transform, visualize, model)



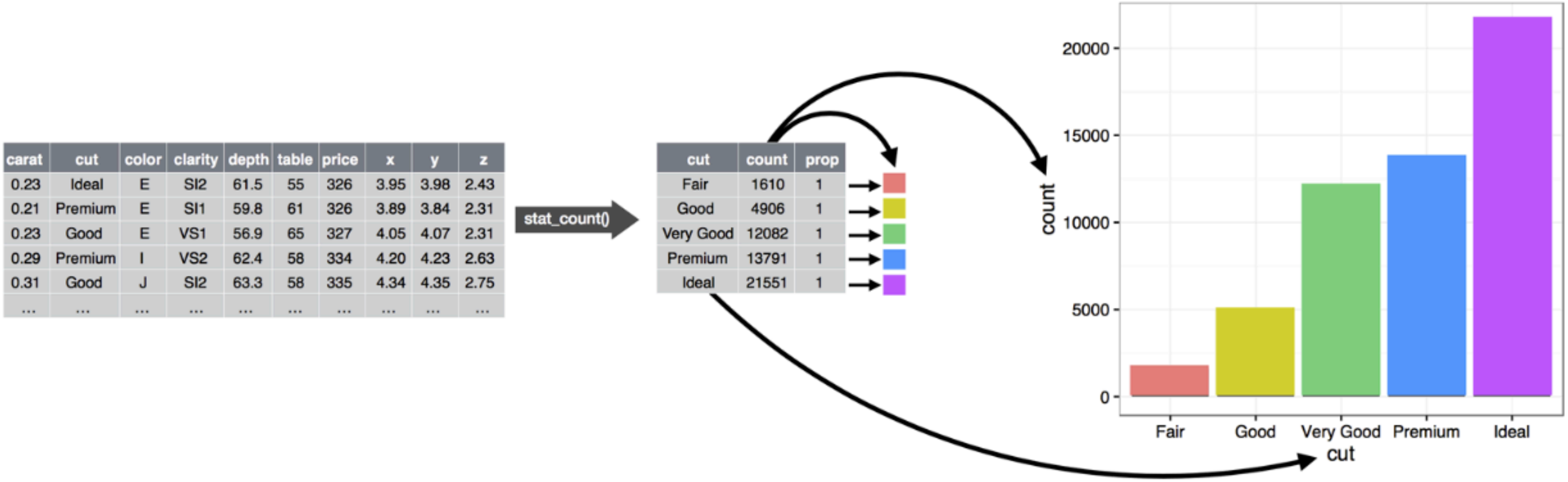
Tidyverse

Explore



5. Place geoms in a cartesian coordinate system.

6. Map the y values to `..count..` and the x values to `cut`.



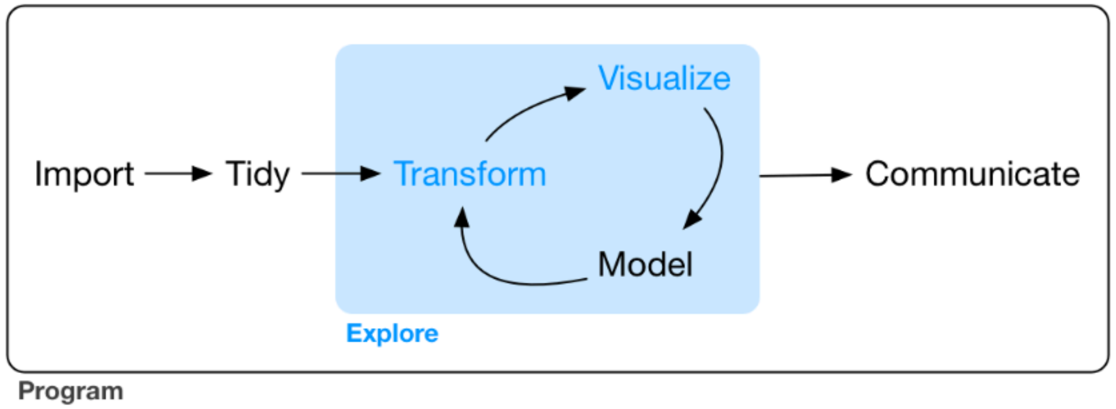
carat	cut	color	clarity	depth	table	price	x	y	z
0.23	Ideal	E	SI2	61.5	55	326	3.95	3.98	2.43
0.21	Premium	E	SI1	59.8	61	326	3.89	3.84	2.31
0.23	Good	E	VS1	56.9	65	327	4.05	4.07	2.31
0.29	Premium	I	VS2	62.4	58	334	4.20	4.23	2.63
0.31	Good	J	SI2	63.3	58	335	4.34	4.35	2.75
...	...	...	...	...	...	...	...	...	...

stat_count()

cut	count	prop
Fair	1610	1
Good	4906	1
Very Good	12082	1
Premium	13791	1
Ideal	21551	1

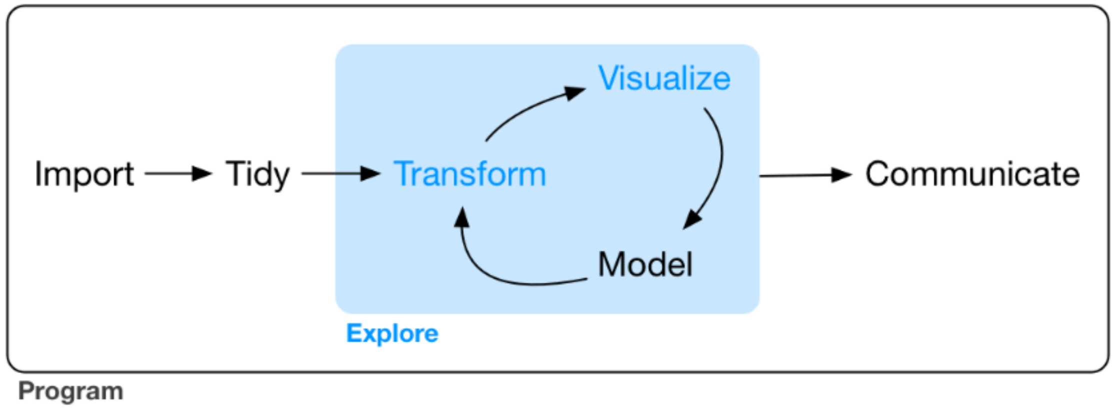
Tidyverse

Dpylr — transform



- Data manipulation challenges:
  - Pick observations by their values (filter())
  - Reorder the rows (arrange())
  - Pick variables by their names (select())
  - Create new variables (mutate())
  - Collapse many values to a single summary (summarize())
  - Group rows (group_by())
- All dplyr verbs expect a data.frame and produce a new data.frame

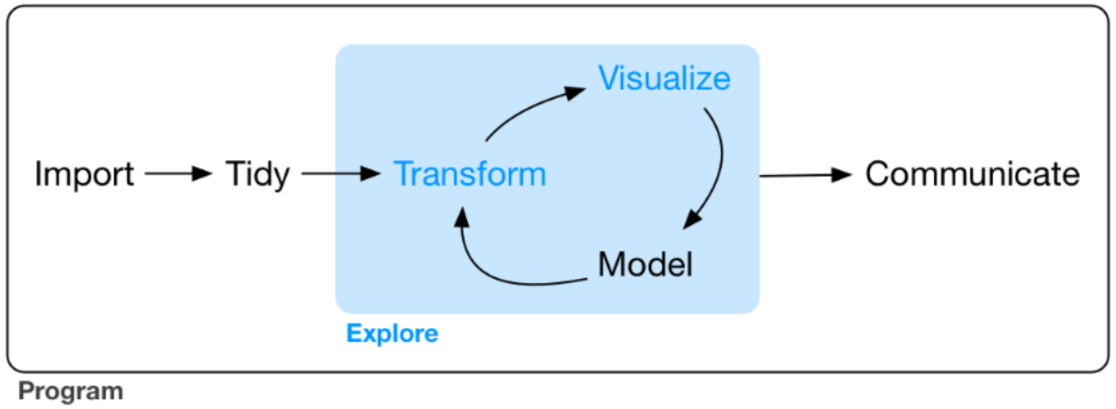
Dpylr — transform



Select

- Select columns with select()
  - mpg %>% select(model,manufacturer,cyl)
  - mpg %>% select(model:year)
  - mpg %>% select(-(model:year))
- Helper functions
  - starts_with("abc") matches names that begin with “abc”.
  - ends_with("xyz") matches names that end with “xyz”.
  - contains("ijk") matches names that contain “ijk”.
  - num_range("x", 1:3) matches x1, x2, and x3
- rename(modelo = model)
- mpg %>% select(trans, drv, **everything**())

Dpylr — transform

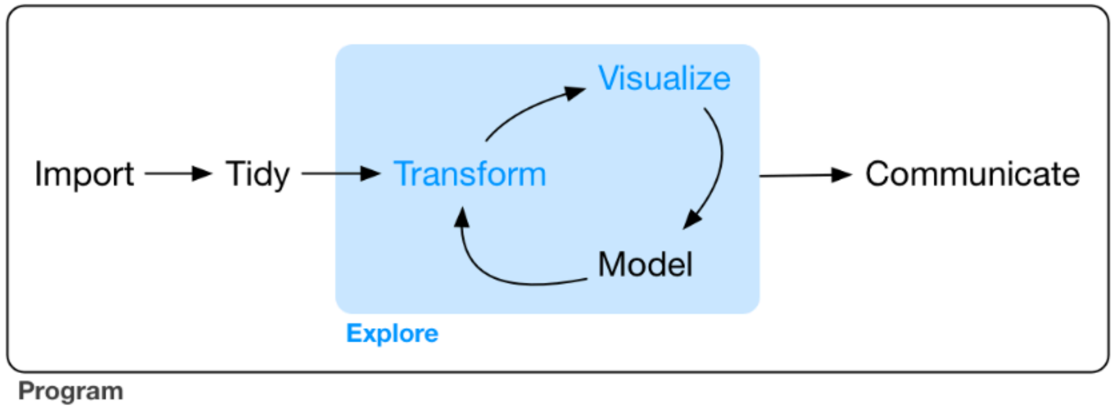


Mutate

- New Variables with mutate()
- mutate() always adds new columns at the end of data frame
  - mpg %>% mutate(delta=hwy-cty)
  - ?transmutate()
- Helper functions
  - Arithmetic operators +, -, *, /, ^
  - Logs log(), log2(), log10()
  - Cumulative R provides cumsum(), cumprod(), cummin(), cummax(); and dplyr provides cummean()
  - Logical comparisons <, <=, >, >=, !=

Dpylr — transform

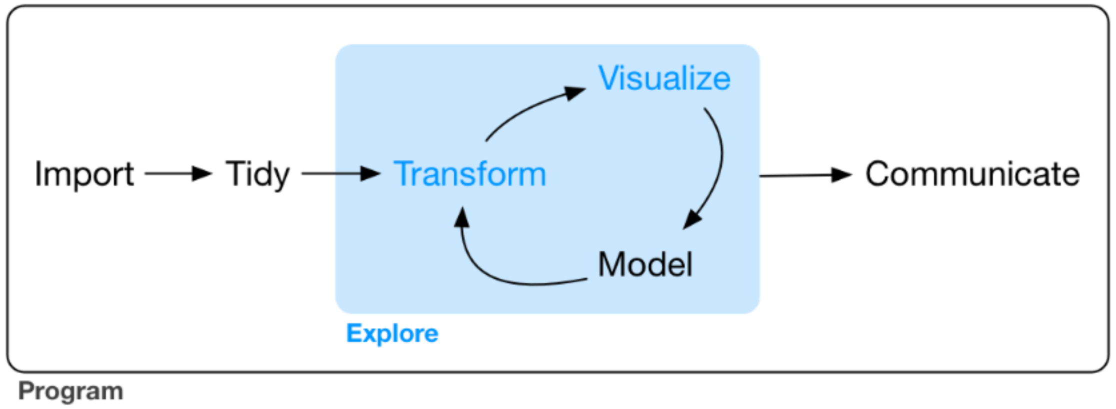
Summarise



- Grouped Summaries with summarize()
- collapses a data frame to a single row
  - mpg %>% summarise(mean(hwy))
- summarize() and group_by() a perfect couple
  - mpg %>% group_by(manufacturer) %>% summarize(m=mean(hwy)) %>% arrange(desc(m))
- Helper functions
  - Measures of location: mean(x), median(x)
  - Measures of spread: sd(x), IQR(x), mad(x)
  - Measures of rank: min(x), quantile(x, 0.25), max(x)
  - Counts: n(), n_distinct(x)

Dpylr — transform

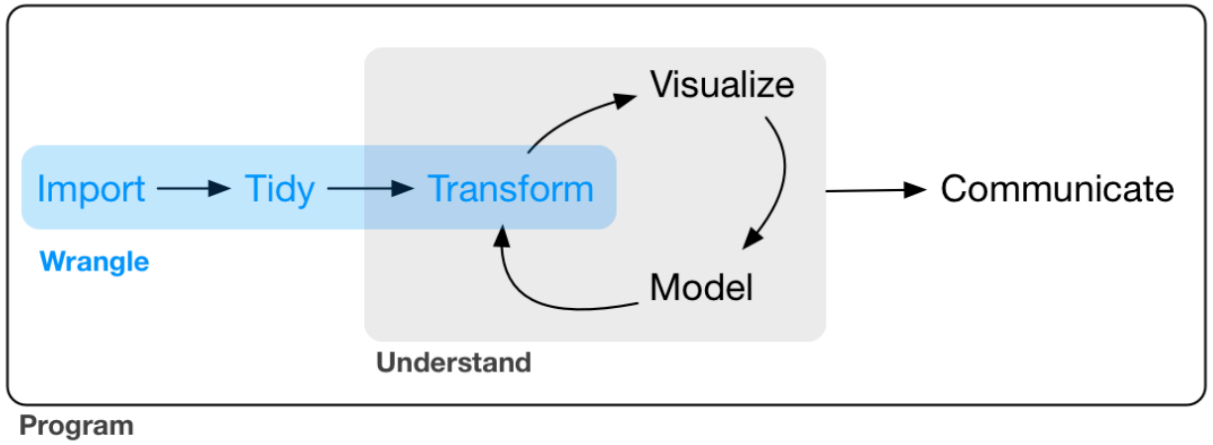
Summarise



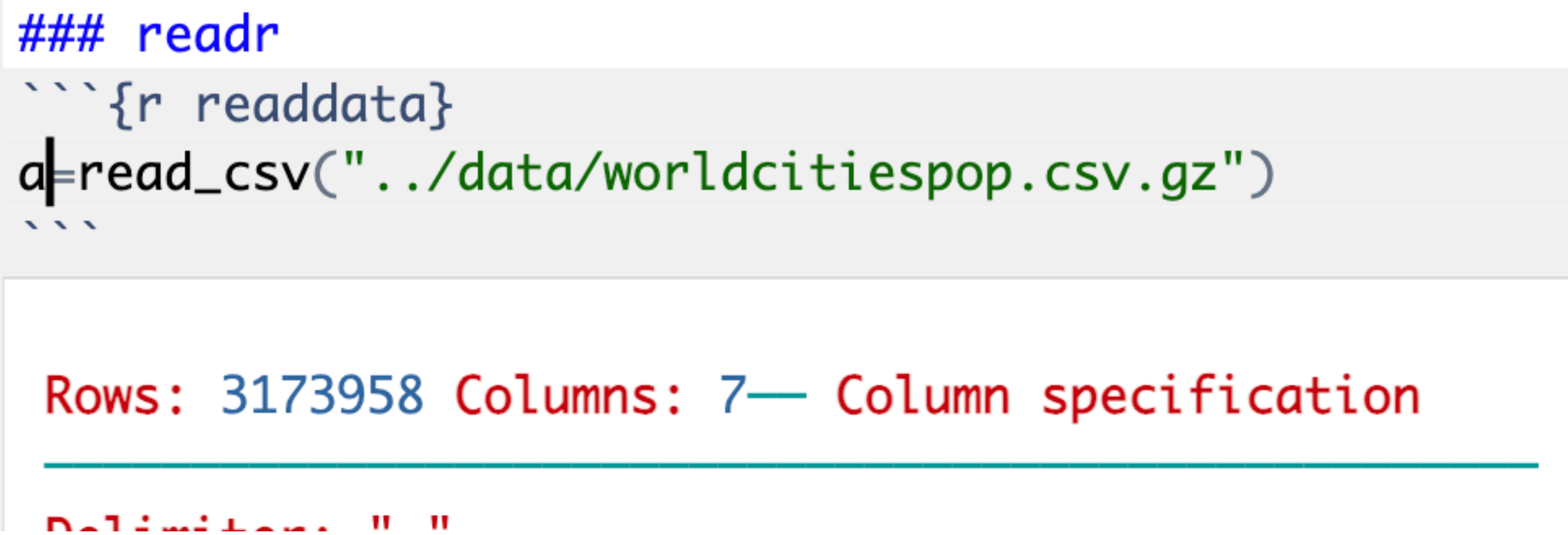
- Grouped Summaries with summarize()
- collapses a data frame to a single row
    - mpg %>% summarise(mean(hwy))
- summarize() and group_by() a perfect couple
    - mpg %>% group_by(manufacturer) %>% summarize(m=mean(hwy)) %>% arrange(desc(m))
- Helper functions
    - Measures of location: mean(x), median(x)
    - Measures of spread: sd(x), IQR(x), mad(x)
    - Measures of rank: min(x), quantile(x, 0.25), max(x)
    - Counts: n(), n_distinct(x)

Tidyverse

Wrangle

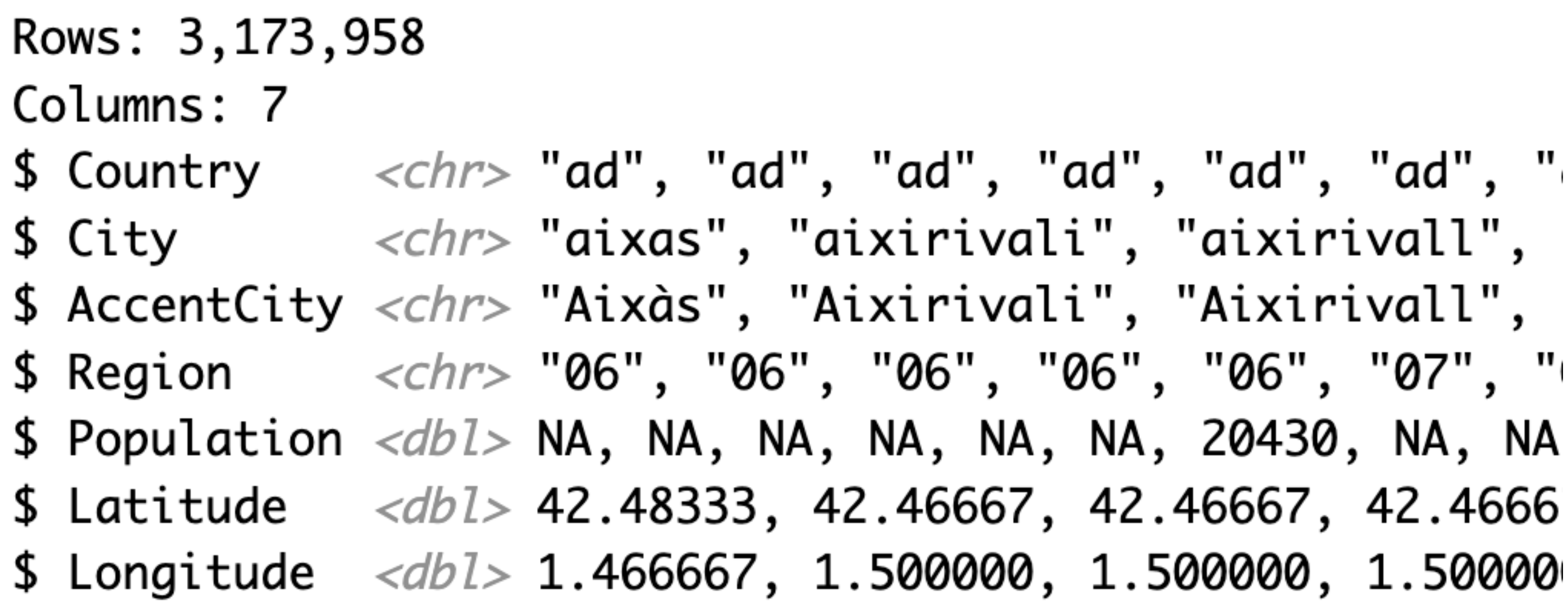


- Getting your data in a helpful form for visualization and modeling
- data.frames and Tibb
- Data import with read
    - read plain-text rect
    - From a file.txt to a
    - Common functions
        - read_csv(). read_ read_delim(), rea



~~~
### readr
```{r readdata}
a=read_csv("../data/worldcitiespop.csv.gz")
```

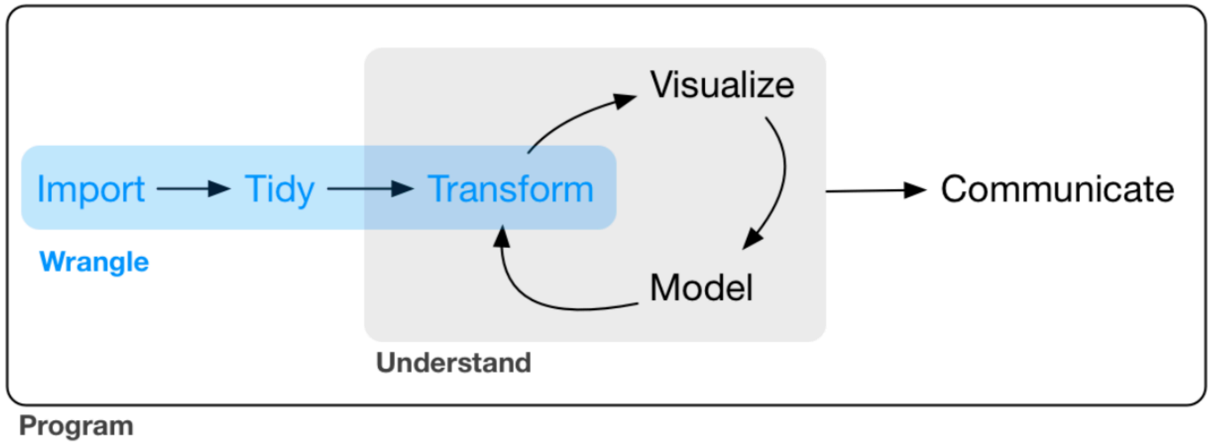
Rows: 3173958 Columns: 7— Column specification
────────────────────────────────────────────────
Delimiter: ","
~~~

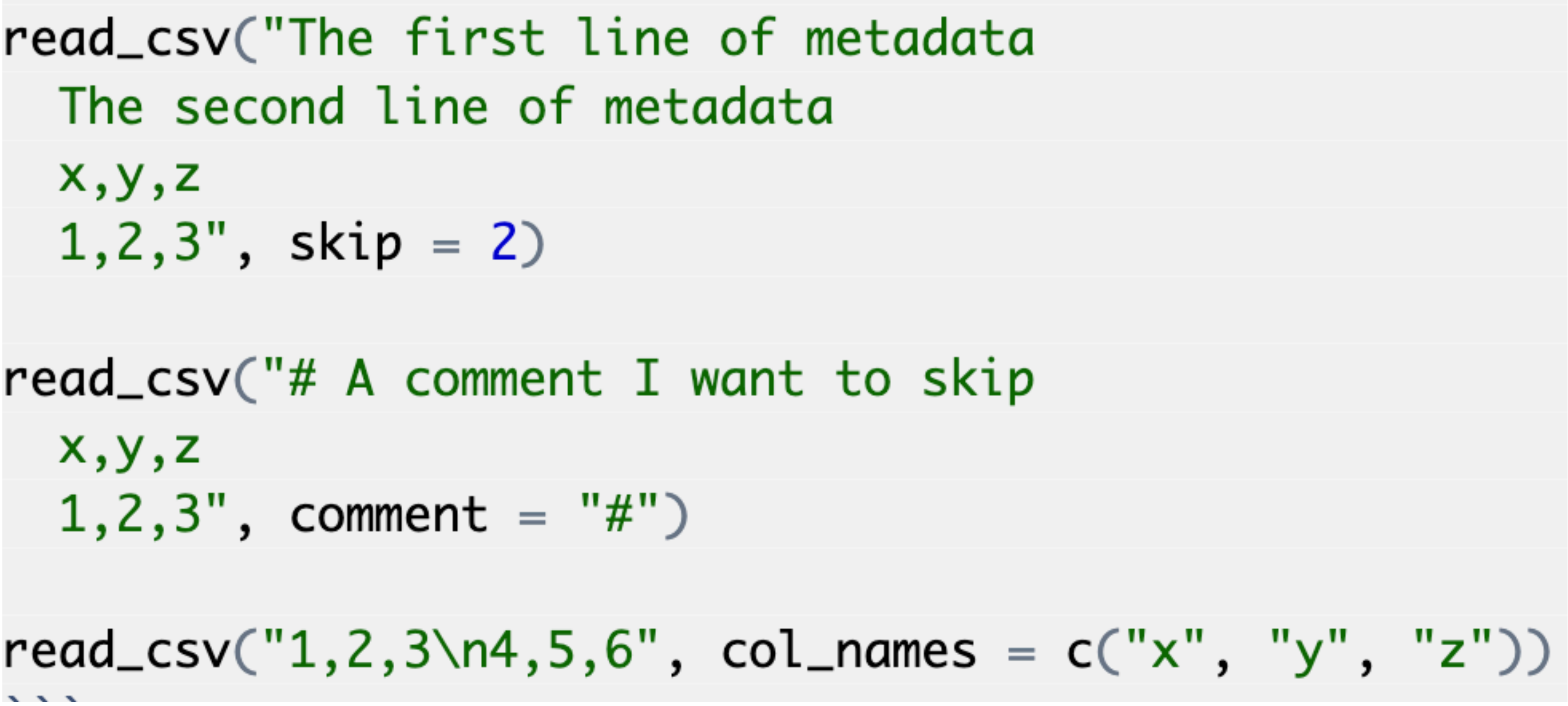


```
Rows: 3,173,958
Columns: 7
$ Country    <chr> "ad", "ad", "ad", "ad", "ad", "ad", "
$ City       <chr> "aixas", "aixirivali", "aixirivall",
$ AccentCity <chr> "Aixàs", "Aixirivali", "Aixirivall",
$ Region     <chr> "06", "06", "06", "06", "06", "07", "
$ Population <dbl> NA, NA, NA, NA, NA, NA, 20430, NA, NA
$ Latitude   <dbl> 42.48333, 42.46667, 42.46667, 42.4666
$ Longitude  <dbl> 1.466667, 1.500000, 1.500000, 1.50000
```

Tidyverse

Readr

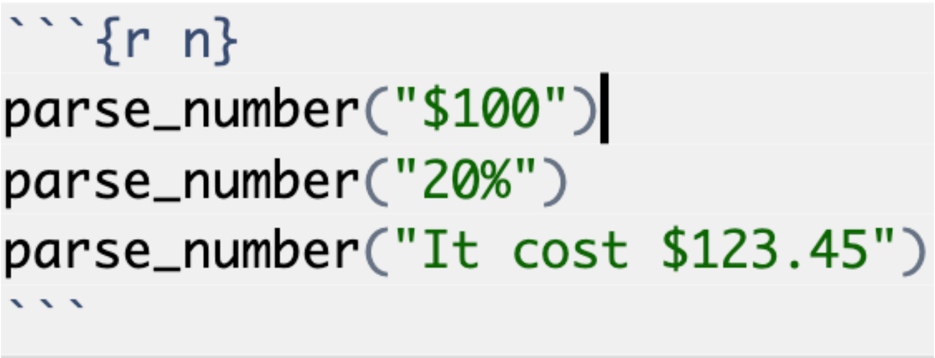




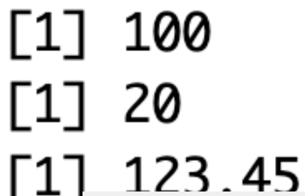
~~~~
read_csv("The first line of metadata
  The second line of metadata
  x,y,z
  1,2,3", skip = 2)

read_csv("# A comment I want to skip
  x,y,z
  1,2,3", comment = "#")

read_csv("1,2,3\n4,5,6", col_names = c("x", "y", "z"))
```
~~~~

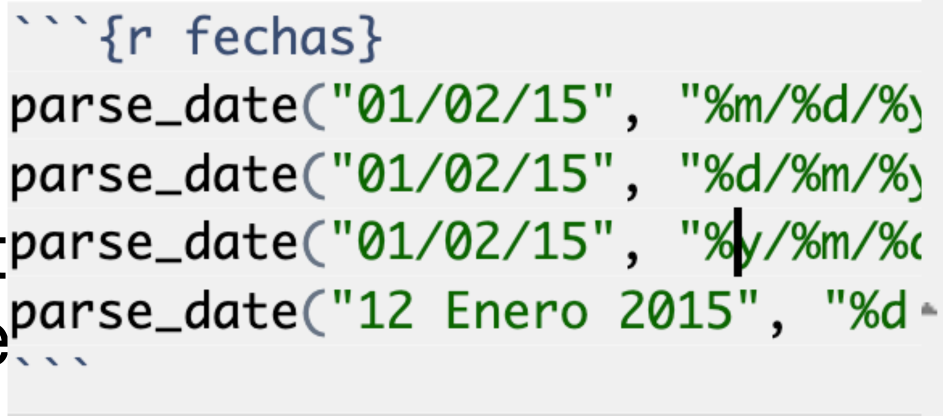


~~~~
```{r n}
parse_number("$100")
parse_number("20%")
parse_number("It cost $123.45")
```
~~~~



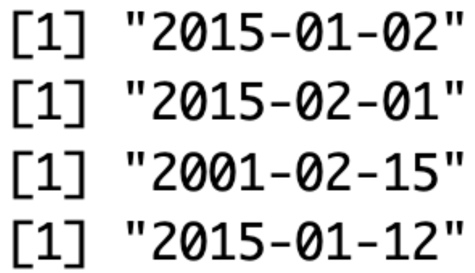
```
[1] 100
[1] 20
[1] 123.45
```

- parse_*() functions:
- From a character vect specialized vector like or date.
- parse_logical(), parse_ parse_date()

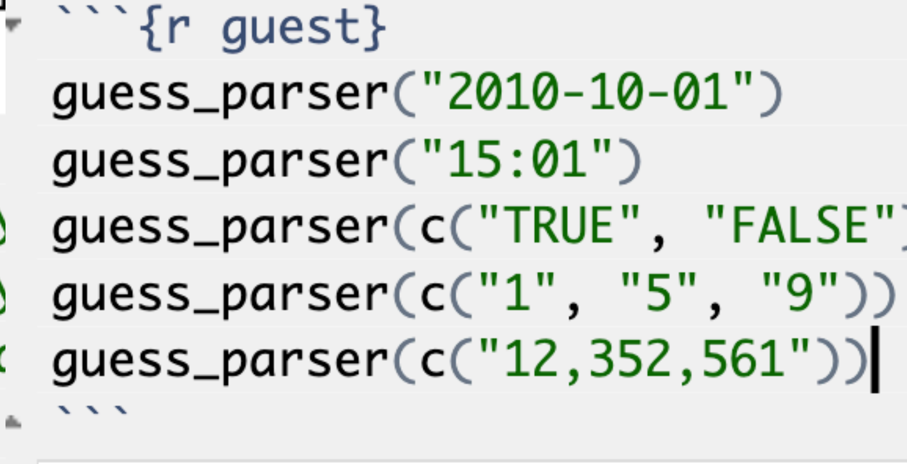


~~~~
```{r fechas}
parse_date("01/02/15", "%m/%d/%y
parse_date("01/02/15", "%d/%m/%y
parse_date("01/02/15", "%y/%m/%d
parse_date("12 Enero 2015", "%d
```
~~~~

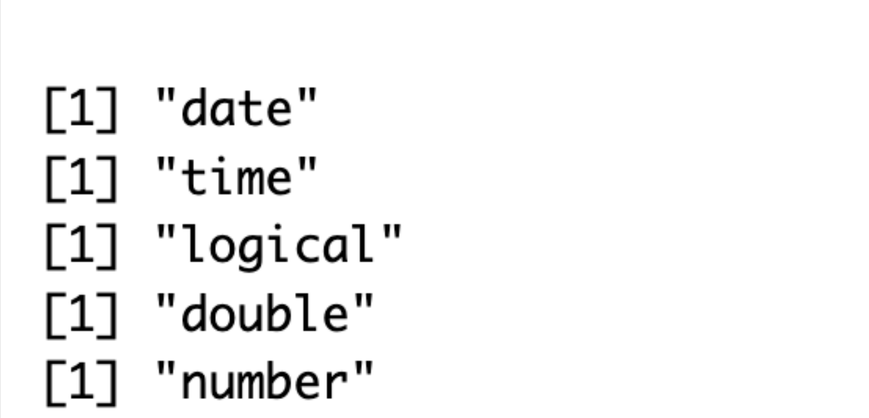
How readr automatically guess



```
[1] "2015-01-02"
[1] "2015-02-01"
[1] "2001-02-15"
[1] "2015-01-12"
```



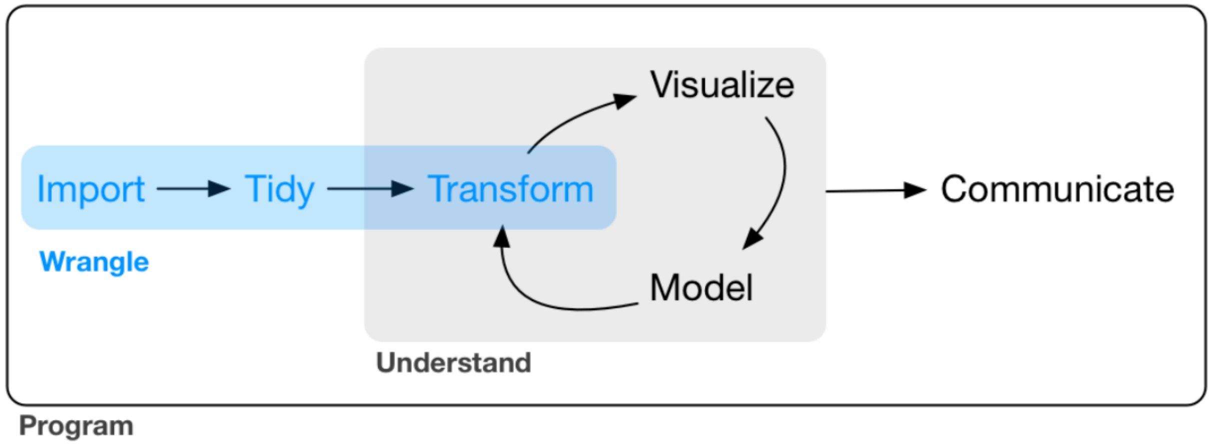
~~~~
```{r guest}
guess_parser("2010-10-01")
guess_parser("15:01")
guess_parser(c("TRUE", "FALSE"
guess_parser(c("1", "5", "9"))
guess_parser(c("12,352,561"))
```
~~~~



```
[1] "date"
[1] "time"
[1] "logical"
[1] "double"
[1] "number"
```

Tidyverse

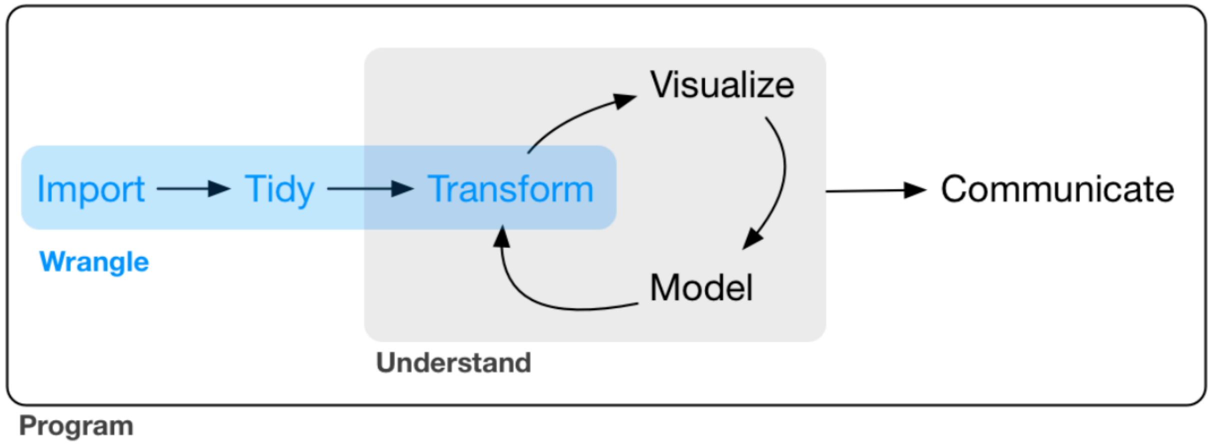
Readr



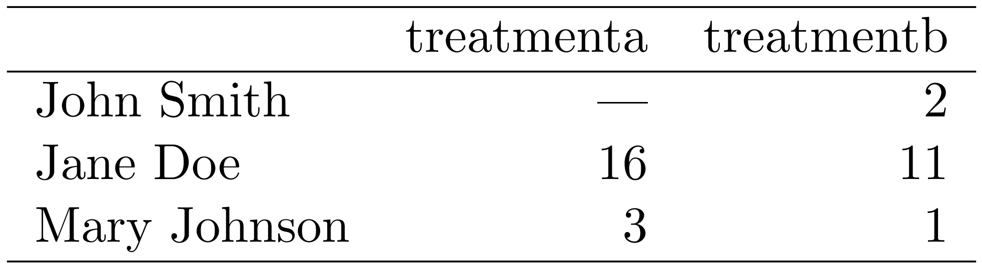
- write_csv() and write_tsv()
  - write_csv(challenge, “challenge.csv")
- write_rds() and read_rds(), store data in R’s custom binary format called RDS
  - write_rds(challenge, "challenge.rds")
  - read_rds("challenge.rds")

Tidyverse

Tidy

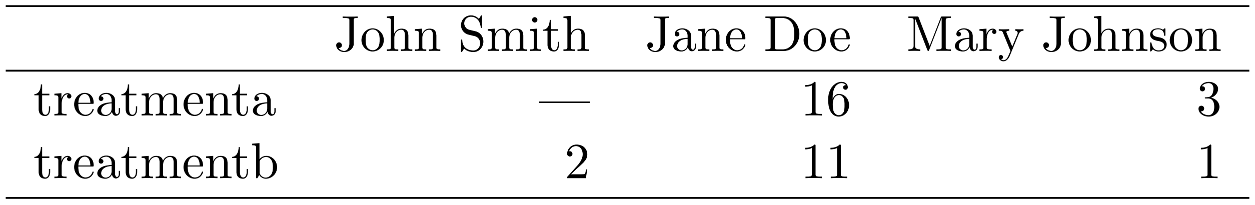


Journal of Statistical Software



	treatmenta	treatmentb
John Smith	—	2
Jane Doe	16	11
Mary Johnson	3	1

Table 1: Typical presentation dataset.



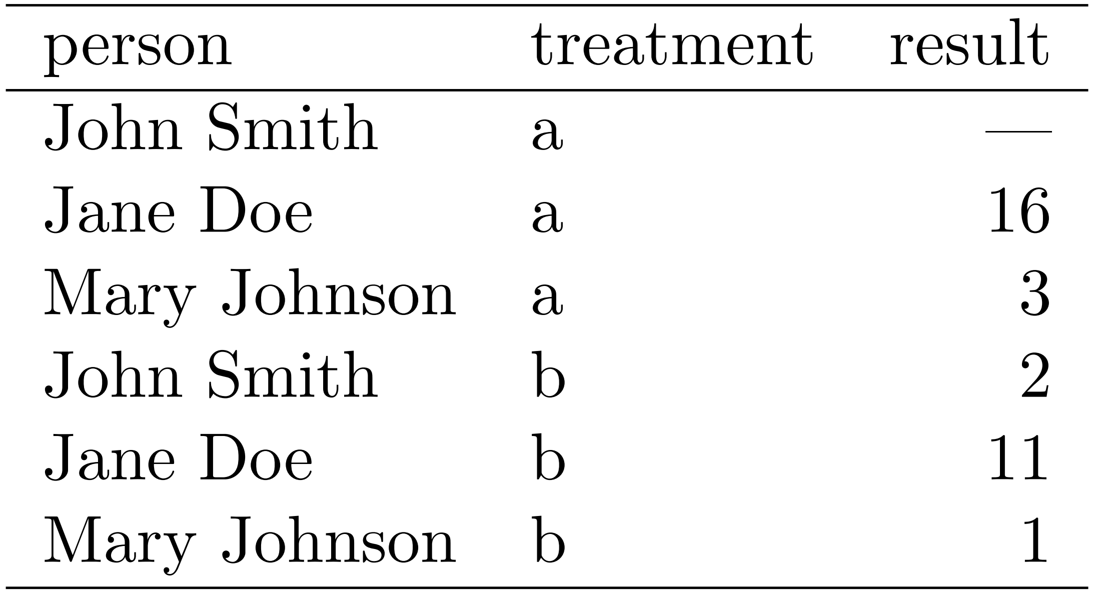
	John Smith	Jane Doe	Mary Johnson
treatmenta	—	16	3
treatmentb	2	11	1

Table 2: The same data as in Table 1 but structured differently.

Tidy data

1. Each variable forms a column.
2. Each observation forms a row.
3. Each type of observational unit forms a table.

Tidy Data

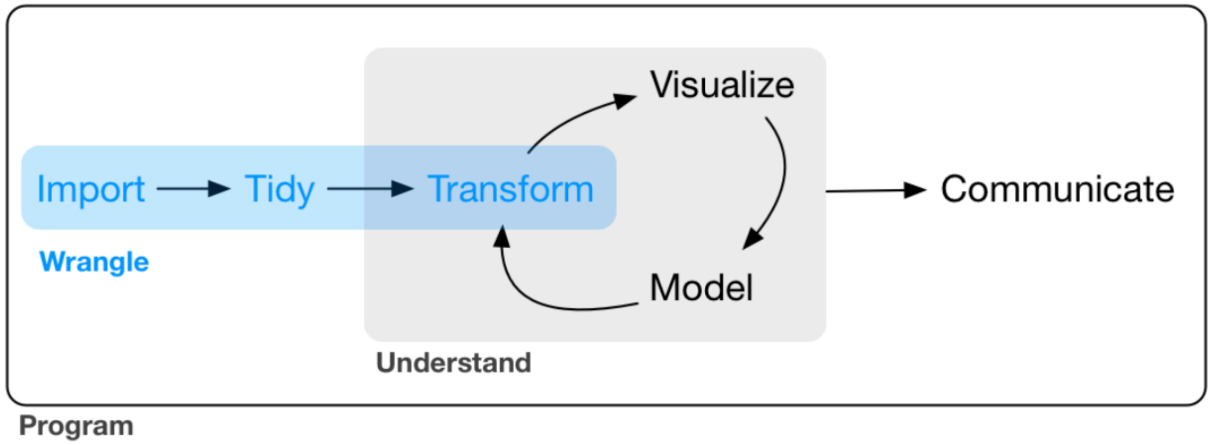


person	treatment	result
John Smith	a	—
Jane Doe	a	16
Mary Johnson	a	3
John Smith	b	2
Jane Doe	b	11
Mary Johnson	b	1

- Column headers are values, not variable names.
- Multiple variables are stored in one column.
- Variables are stored in both rows and columns.
- Multiple types of observational units are stored in the same table.
- A single observational unit is stored in multiple tables.

Tidyverse

Tidy



Import → Tidy → Transform

Wrangle

Visualize

Model

Understand

→ Communicate

Program

religion	<$10k	$10–20k	$20–30k	$30–40k	$40–50k	$50–75k
Agnostic	27	34	60	81	76	137
Atheist	12	27	37	52	35	70
Buddhist	27	21	30	34	33	58
Catholic	418	617	732	670	638	1116
Don't know/refused	15	14	15	11	10	35
Evangelical Prot	575	869	1064	982	881	1486
Hindu	1	9	7	9	11	34
Historically Black Prot	228	244	236	238	197	223
Jehovah's Witness	20	27	24	24	21	30
Jewish	19	19	25	25	30	95

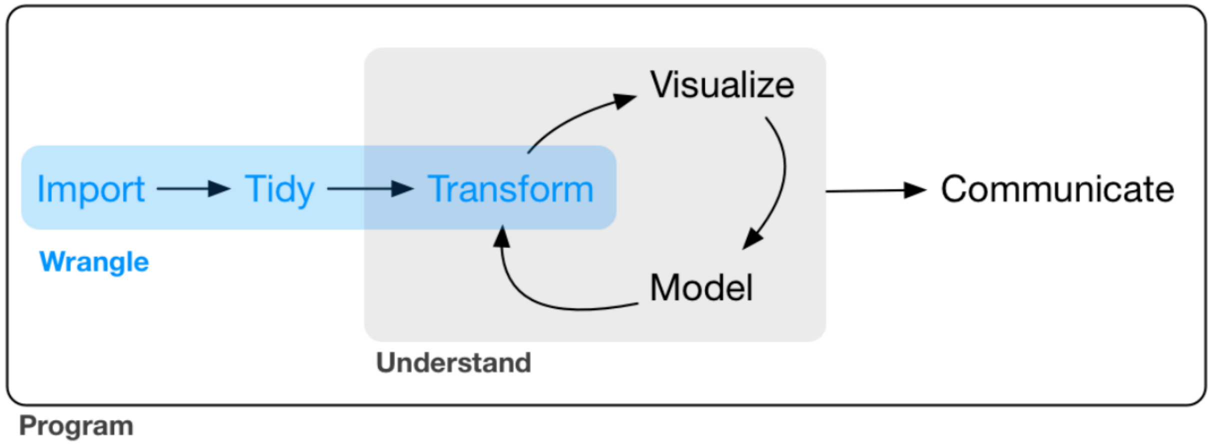
religion	income	freq
Agnostic	<$10k	27
Agnostic	$10–20k	34
Agnostic	$20–30k	60
Agnostic	$30–40k	81
Agnostic	$40–50k	76
Agnostic	$50–75k	137
Agnostic	$75–100k	122
Agnostic	$100–150k	109
Agnostic	>150k	84
Agnostic	Don't know/refused	96

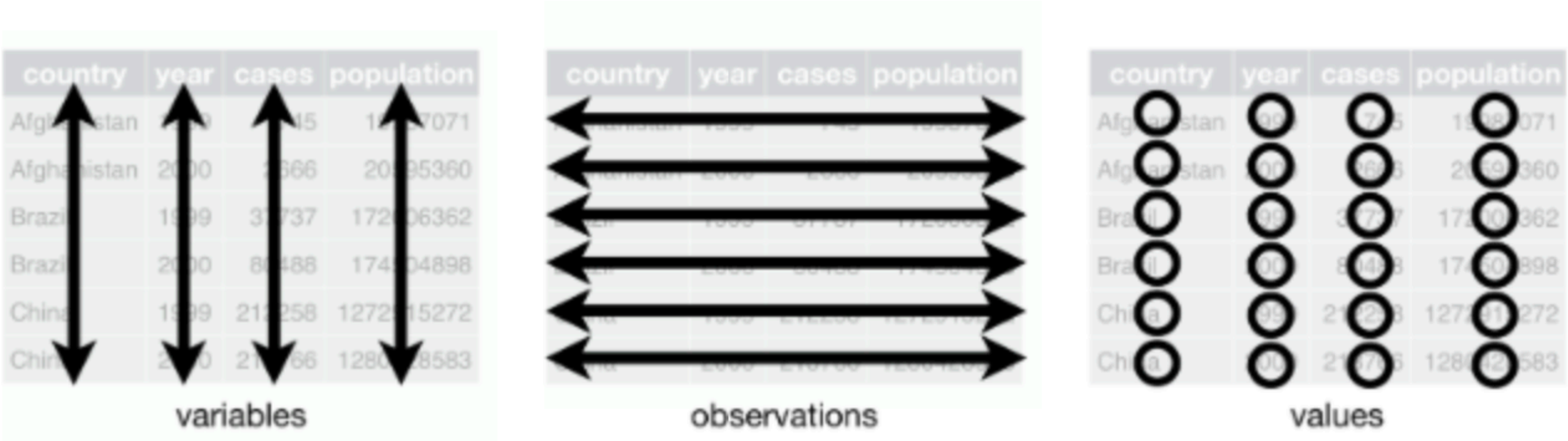
id	year	month	element	d1	d2	d3	d4	d5	d6	d7	d8
MX17004	2010	1	tmax	—	—	—	—	—	—	—	—
MX17004	2010	1	tmin	—	—	—	—	—	—	—	—
MX17004	2010	2	tmax	—	27.3	24.1	—	—	—	—	—
MX17004	2010	2	tmin	—	14.4	14.4	—	—	—	—	—
MX17004	2010	3	tmax	—	—	—	—	32.1	—	—	—
MX17004	2010	3	tmin	—	—	—	—	14.2	—	—	—
MX17004	2010	4	tmax	—	—	—	—	—	—	—	—
MX17004	2010	4	tmin	—	—	—	—	—	—	—	—
MX17004	2010	5	tmax	—	—	—	—	—	—	—	—
MX17004	2010	5	tmin	—	—	—	—	—	—	—	—

id	date	tmax	tmin
MX17004	2010-01-30	27.8	14.5
MX17004	2010-02-02	27.3	14.4
MX17004	2010-02-03	24.1	14.4
MX17004	2010-02-11	29.7	13.4
MX17004	2010-02-23	29.9	10.7
MX17004	2010-03-05	32.1	14.2
MX17004	2010-03-10	34.5	16.8
MX17004	2010-03-16	31.1	17.6
MX17004	2010-04-27	36.3	16.7
MX17004	2010-05-27	33.2	18.2

Tidyverse

Tidy

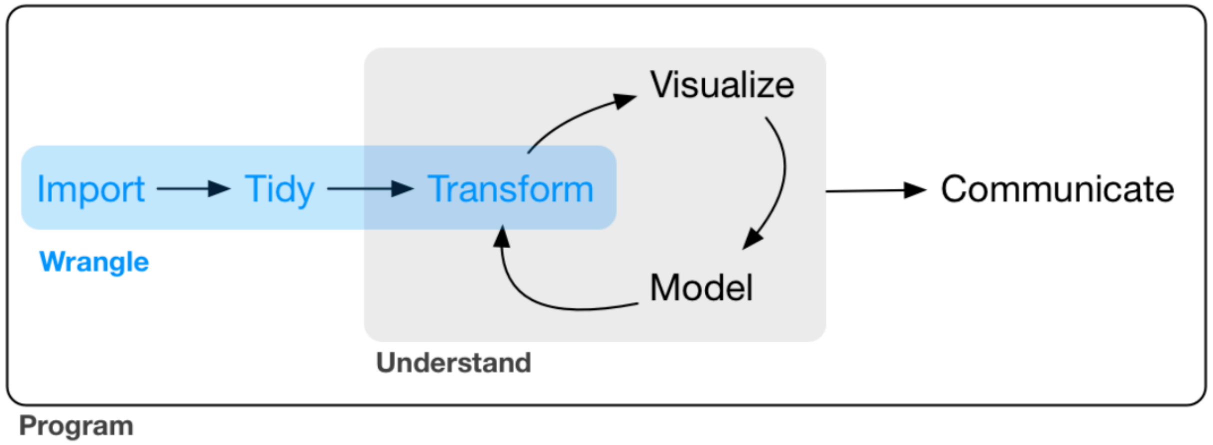




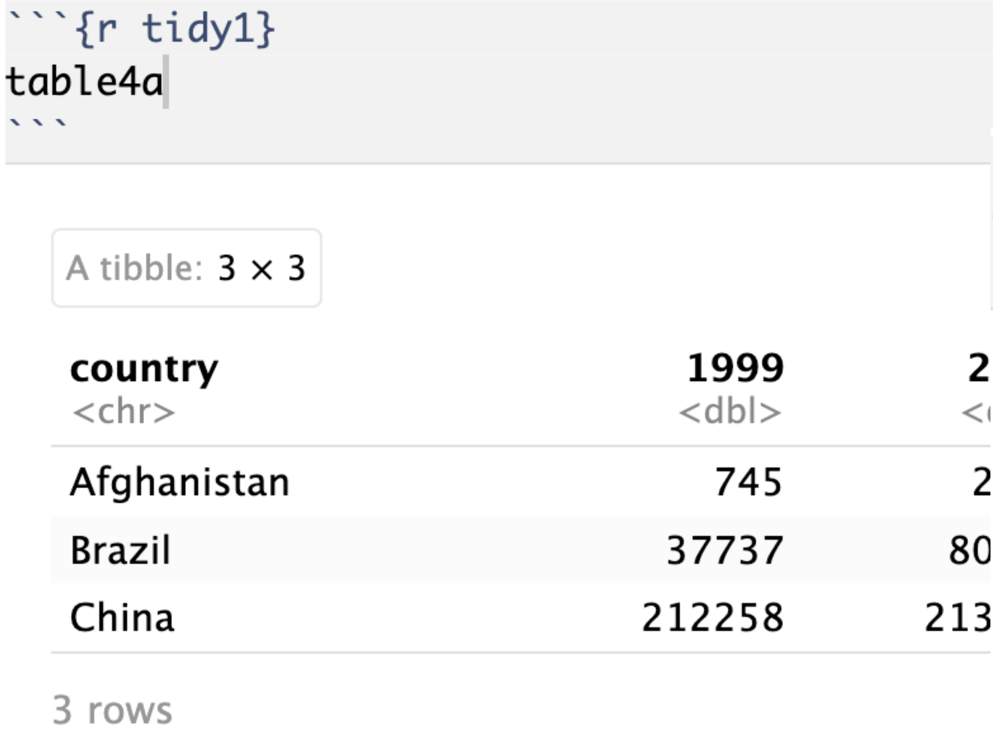
- Tidy dataset:
  - Variables are in columns
  - Observations in rows
  - Values in cells

Tidyverse

Tidy



- Spreading and Gathering
- Dateset where columns are values of a variable

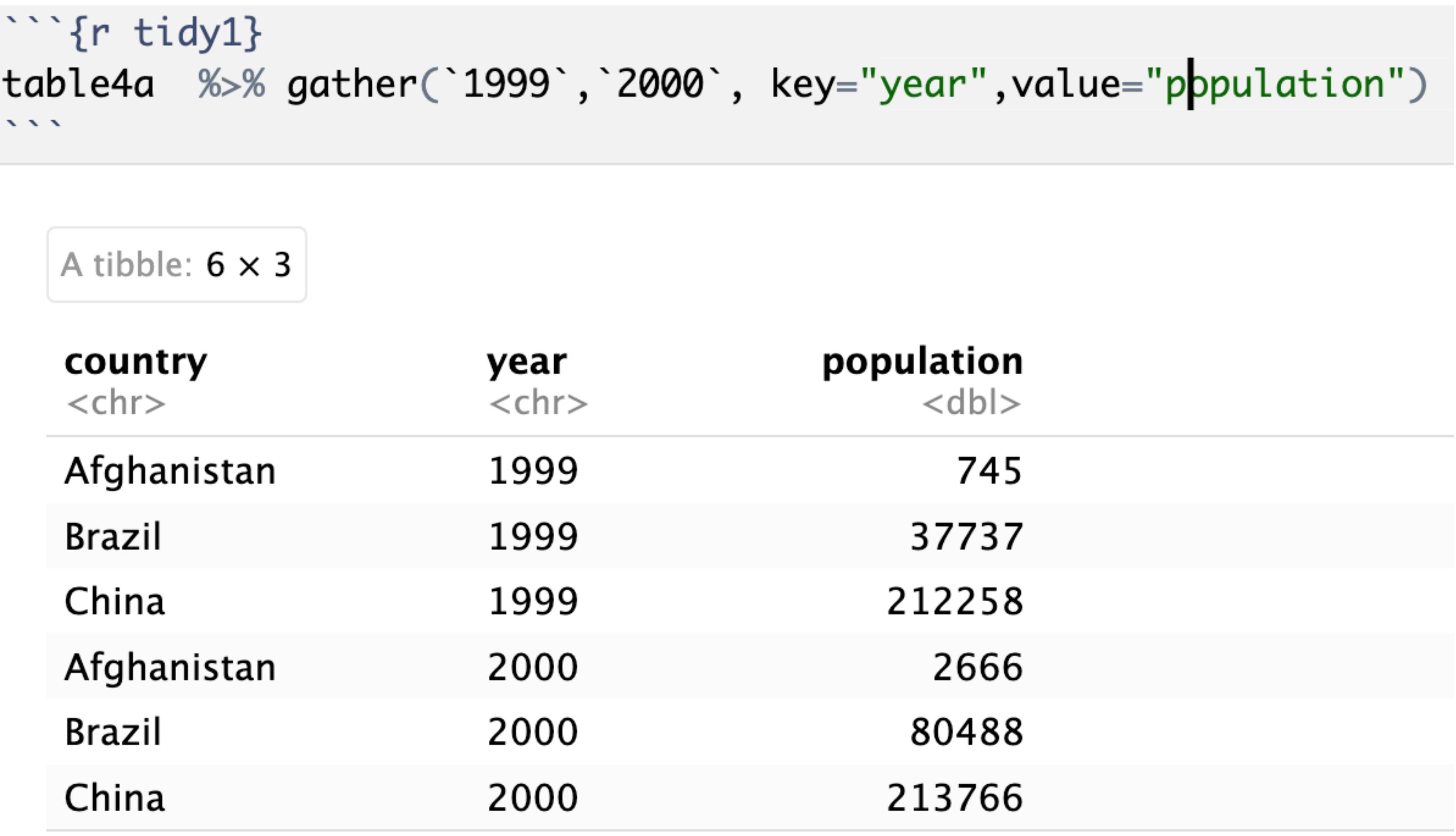


~~~
```{r tidy1}
table4a
```
~~~

A tibble: 3 × 3

country <chr>	1999 <dbl>	2 <
Afghanistan	745	2
Brazil	37737	80
China	212258	213

3 rows



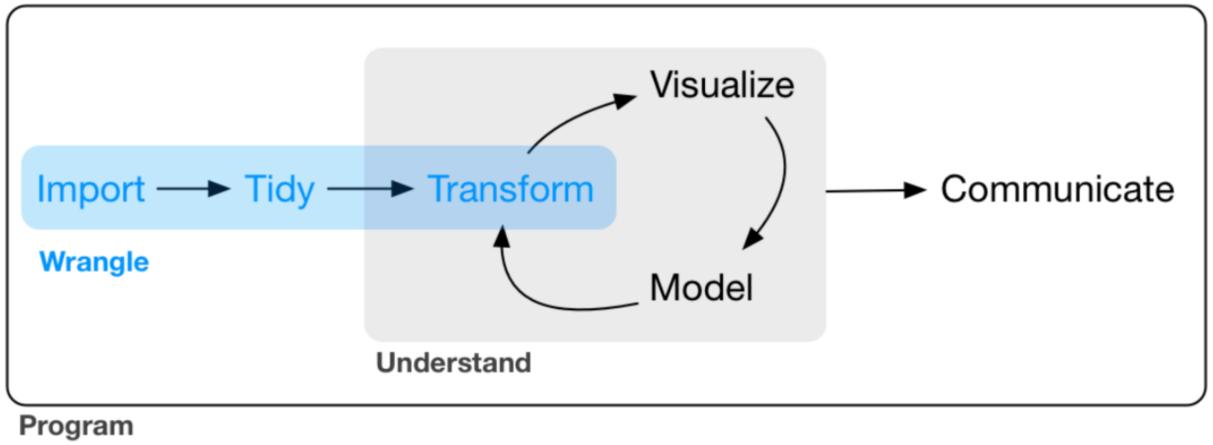
~~~
```{r tidy1}
table4a  %>% gather(`1999`,`2000`, key="year",value="population")
```
~~~

A tibble: 6 × 3

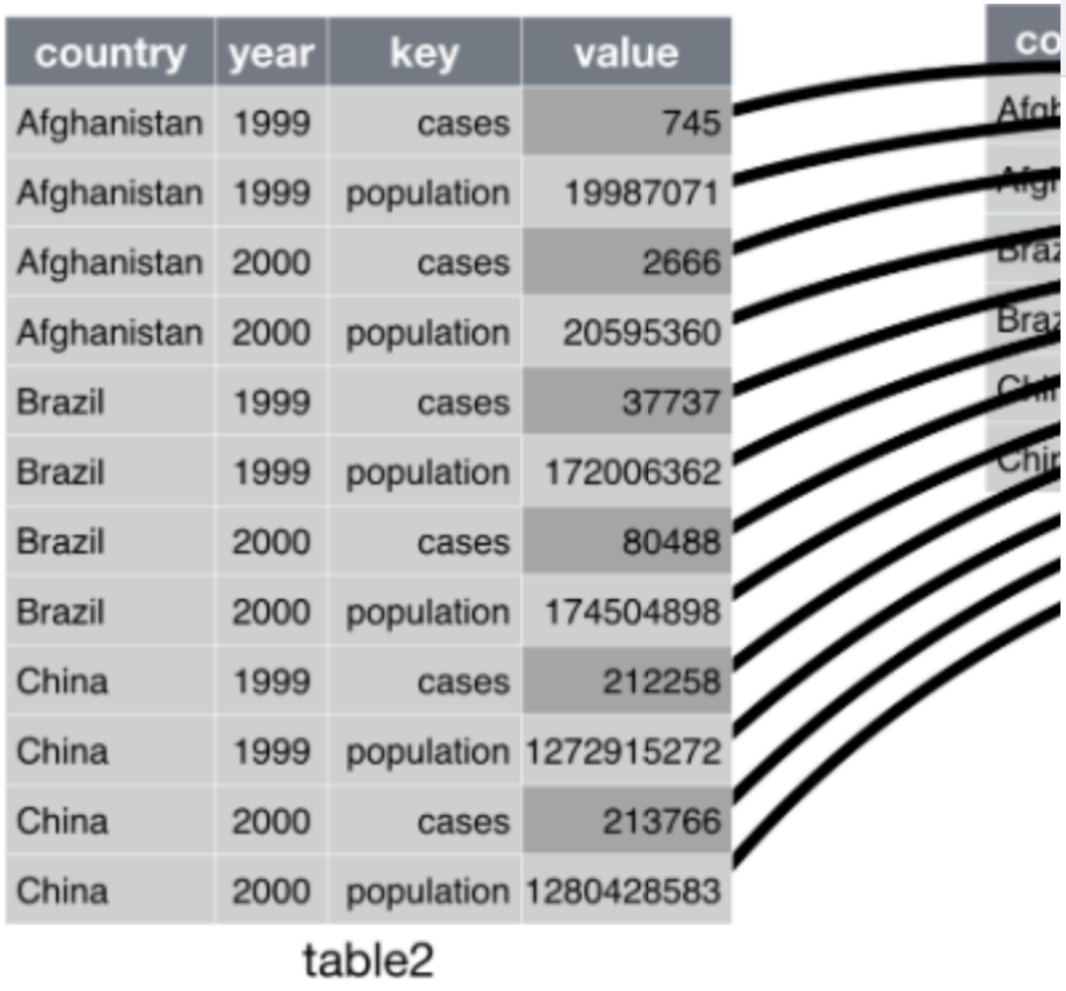
country <chr>	year <chr>	population <dbl>
Afghanistan	1999	745
Brazil	1999	37737
China	1999	212258
Afghanistan	2000	2666
Brazil	2000	80488
China	2000	213766

Tidyverse

Tidy

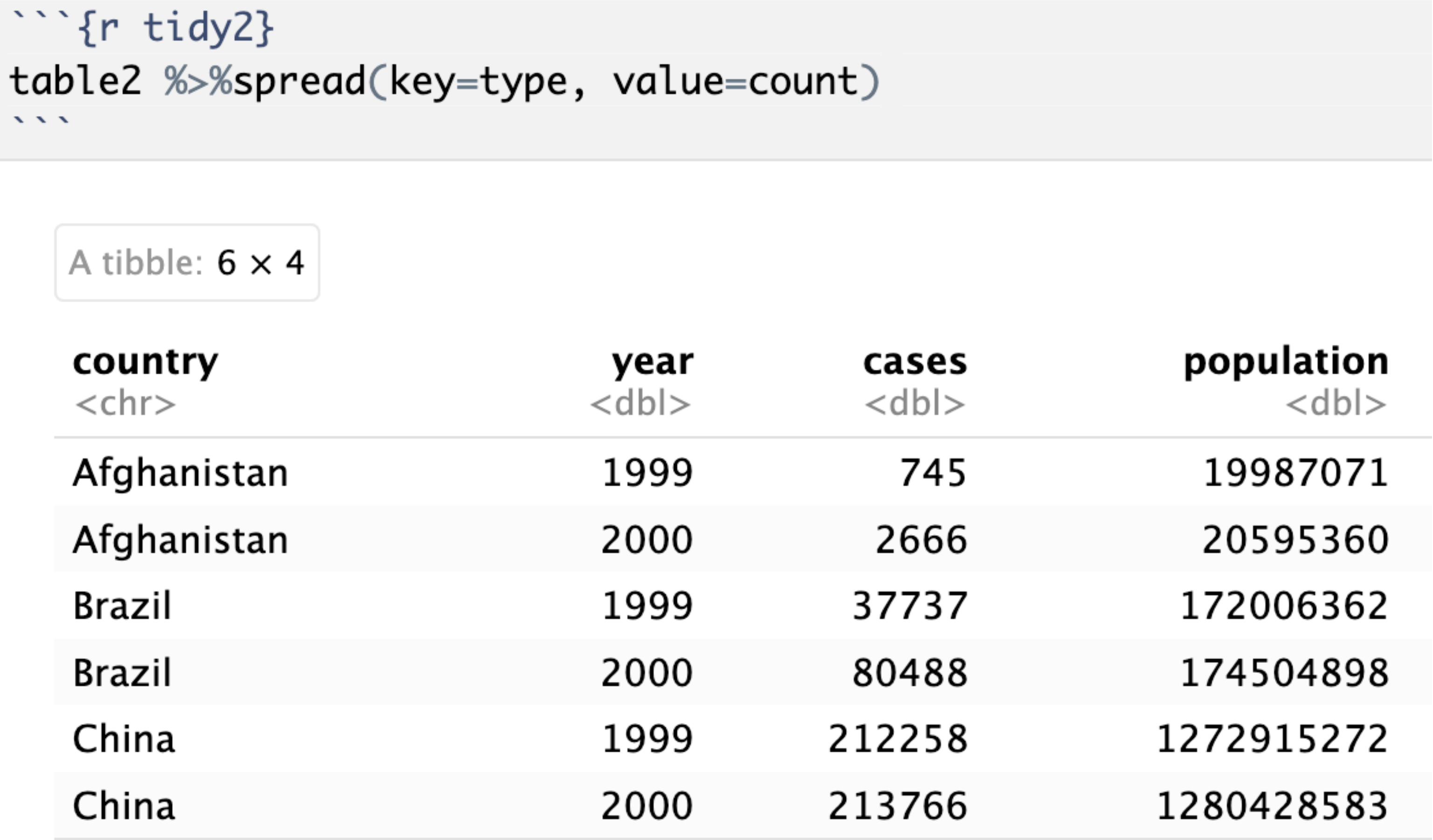


- Spreading
  - Opposite of gathering



country	year	key	value
Afghanistan	1999	cases	745
Afghanistan	1999	population	19987071
Afghanistan	2000	cases	2666
Afghanistan	2000	population	20595360
Brazil	1999	cases	37737
Brazil	1999	population	172006362
Brazil	2000	cases	80488
Brazil	2000	population	174504898
China	1999	cases	212258
China	1999	population	1272915272
China	2000	cases	213766
China	2000	population	1280428583

table2



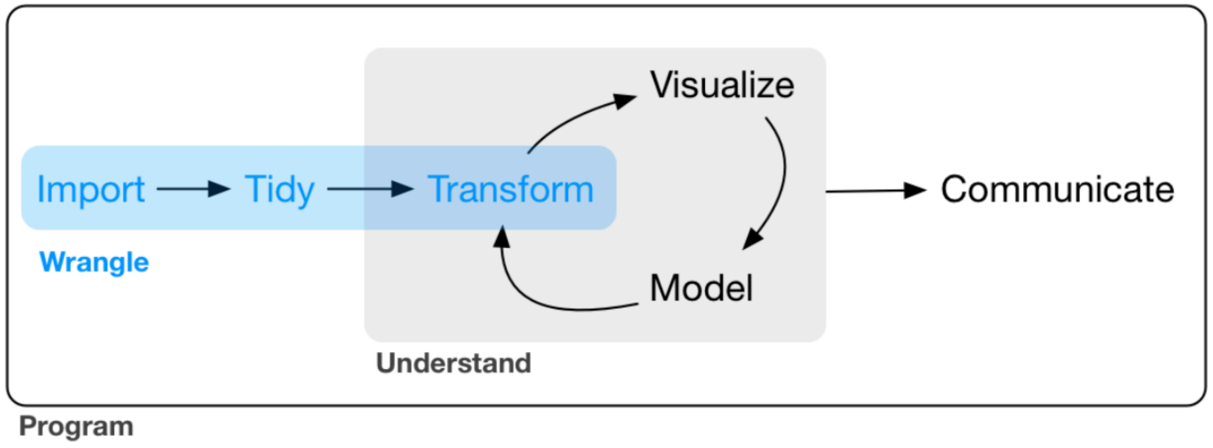
```{r tidy2}
table2 %>%spread(key=type, value=count)
```

A tibble: 6 × 4

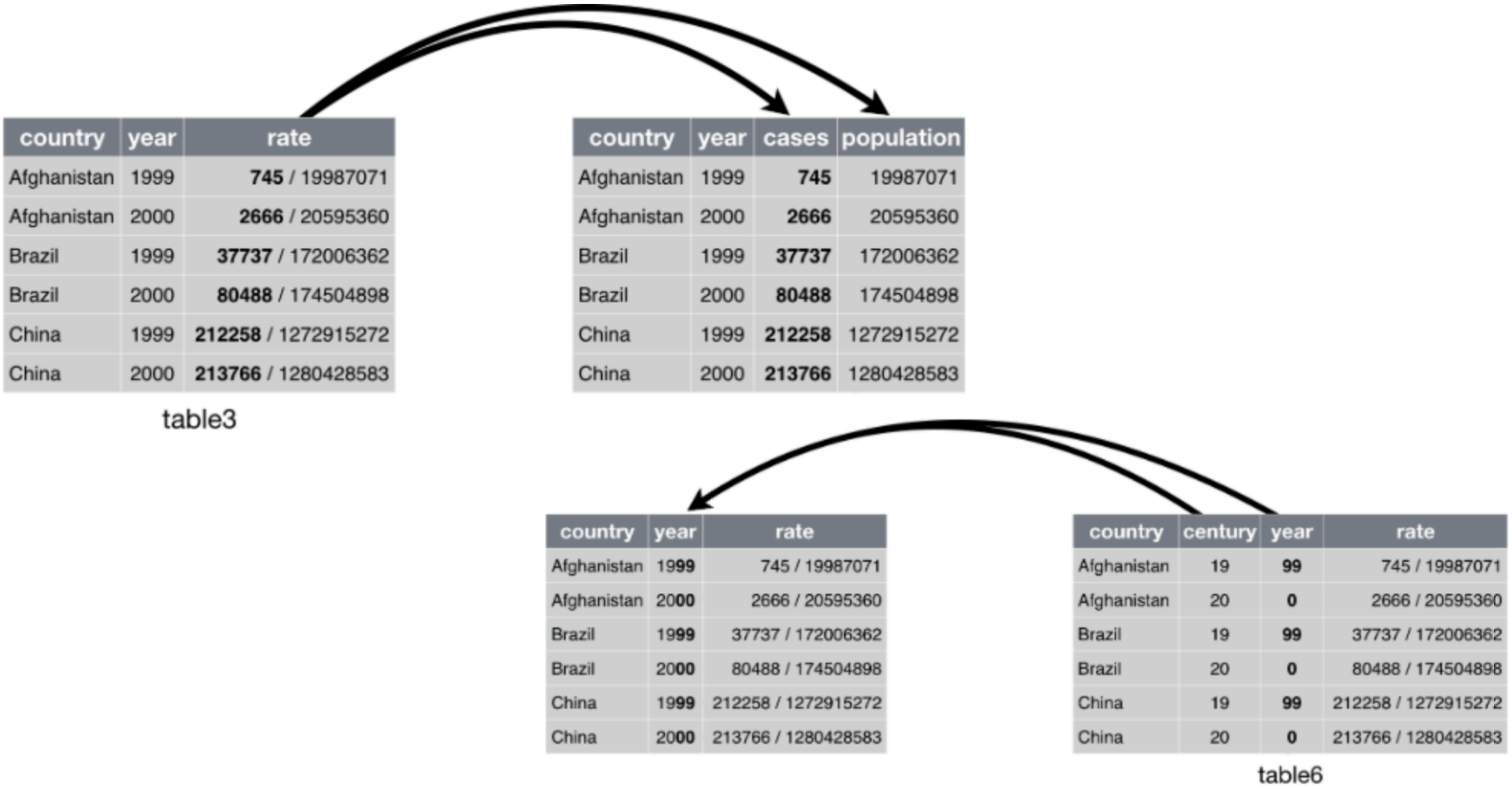
country <chr>	year <dbl>	cases <dbl>	population <dbl>
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	213766	1280428583

Tidyverse

Tidy



- Separate() and unite()



country	year	rate
Afghanistan	1999	**745** / 19987071
Afghanistan	2000	**2666** / 20595360
Brazil	1999	**37737** / 172006362
Brazil	2000	**80488** / 174504898
China	1999	**212258** / 1272915272
China	2000	**213766** / 1280428583

table3

country	year	cases	population
Afghanistan	1999	**745**	19987071
Afghanistan	2000	**2666**	20595360
Brazil	1999	**37737**	172006362
Brazil	2000	**80488**	174504898
China	1999	**212258**	1272915272
China	2000	**213766**	1280428583

country	year	rate
Afghanistan	1999	745 / 19987071
Afghanistan	2000	2666 / 20595360
Brazil	1999	37737 / 172006362
Brazil	2000	80488 / 174504898
China	1999	212258 / 1272915272
China	2000	213766 / 1280428583

country	century	year	rate
Afghanistan	19	**99**	745 / 19987071
Afghanistan	20	**0**	2666 / 20595360
Brazil	19	**99**	37737 / 172006362
Brazil	20	**0**	80488 / 174504898
China	19	**99**	212258 / 1272915272
China	20	**0**	213766 / 1280428583

table6

Questions?
Practice!!!