N

A

Neighborhood Analysis

Neighborhood Analysis

Manipulating Data

Last Session

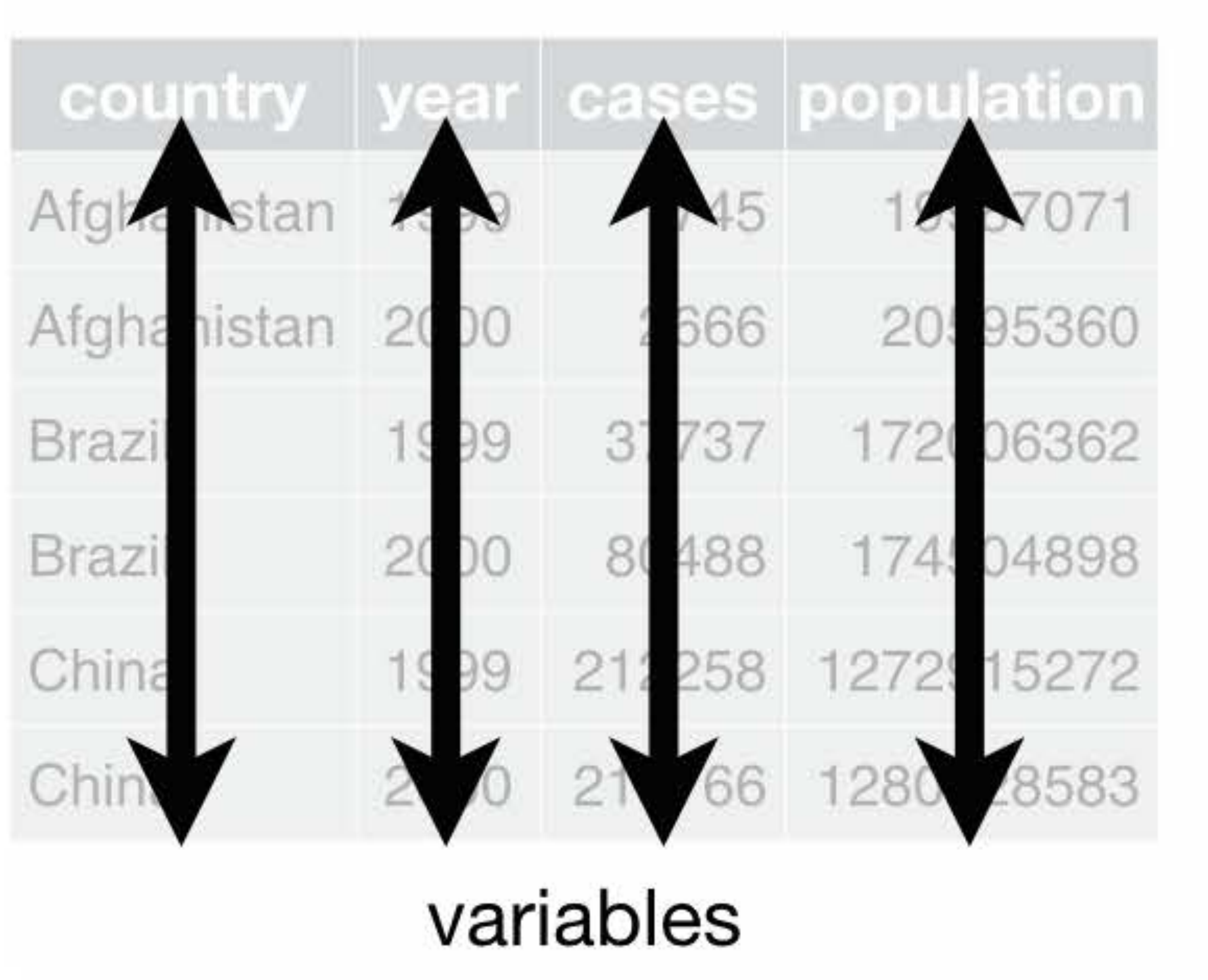
- Introduced data frames
- Data import / export
- Data summarization
- Data formatting

Today's Session

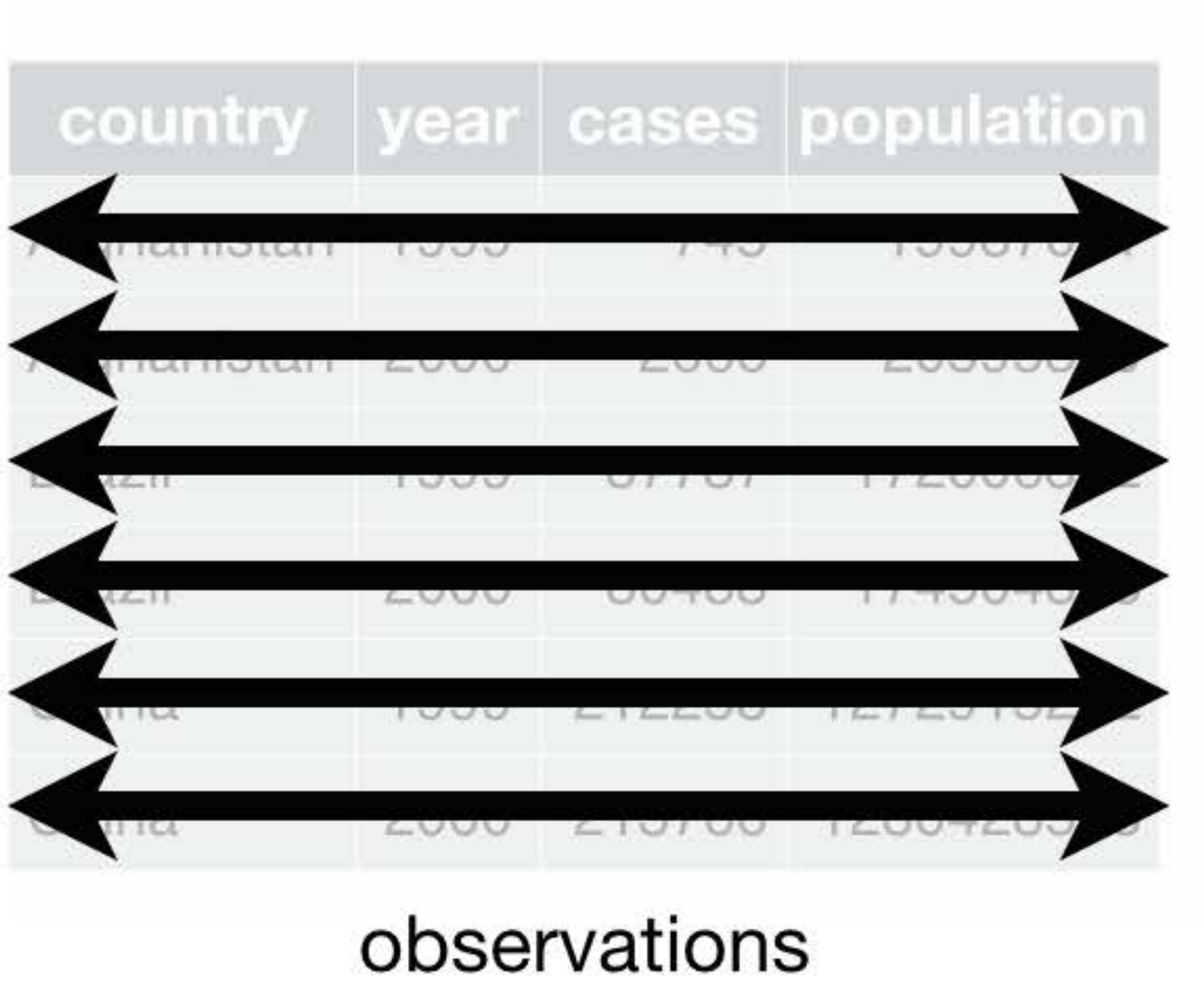
- More advanced tools for data manipulation
- Principles of tidy data
- Applications of dplyr

What is tidy data?

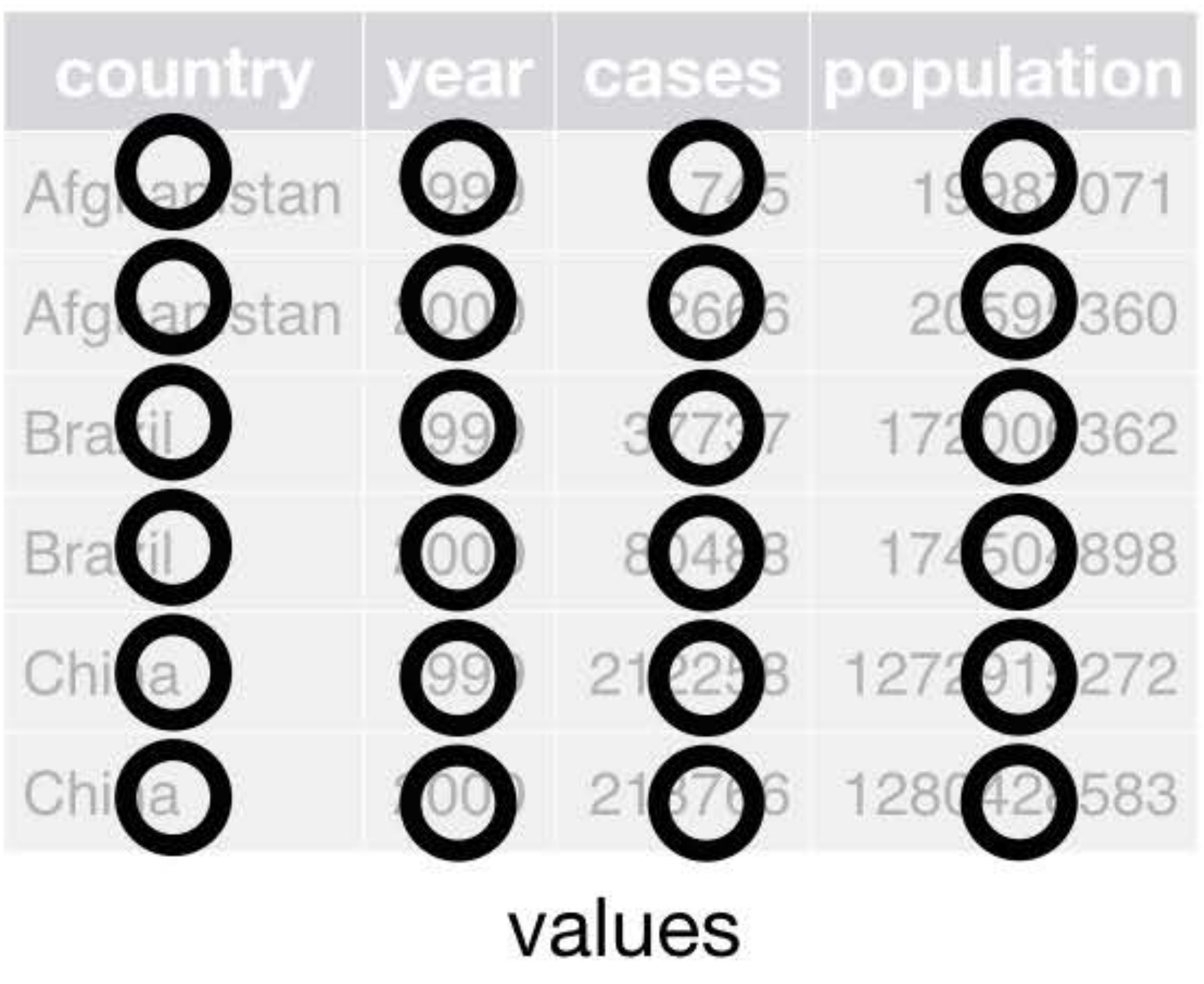
Each variable has it’s own column



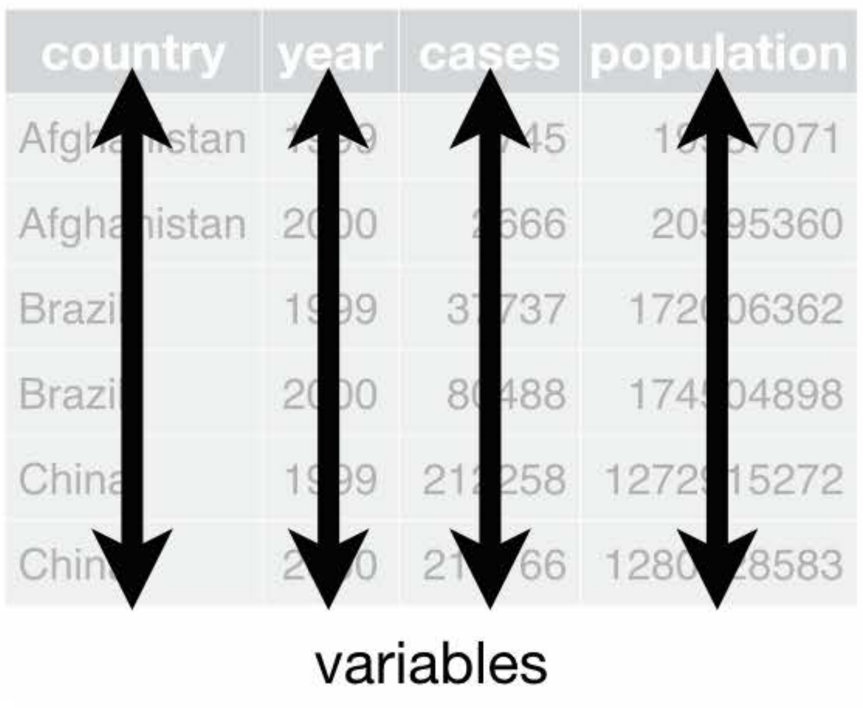
Each observation has it’s own row



Each value has it’s own cell

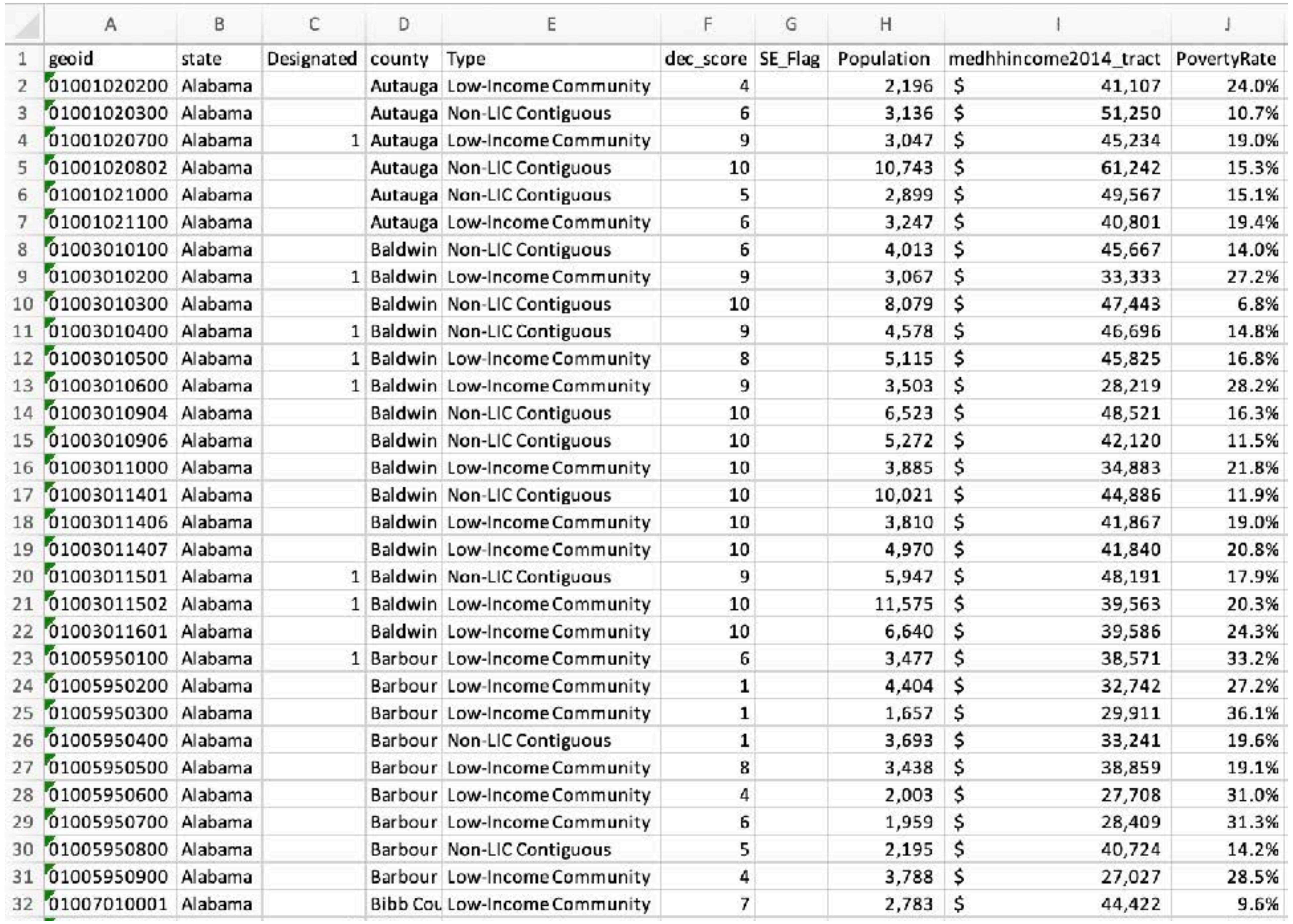


What is tidy data?

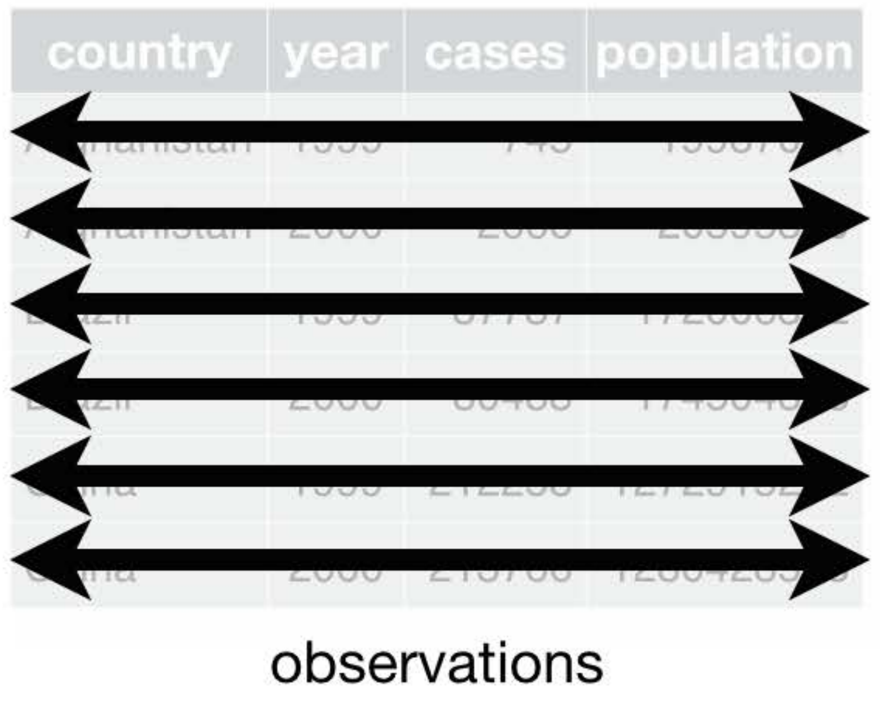


Demographic Indicators

ozs dataset

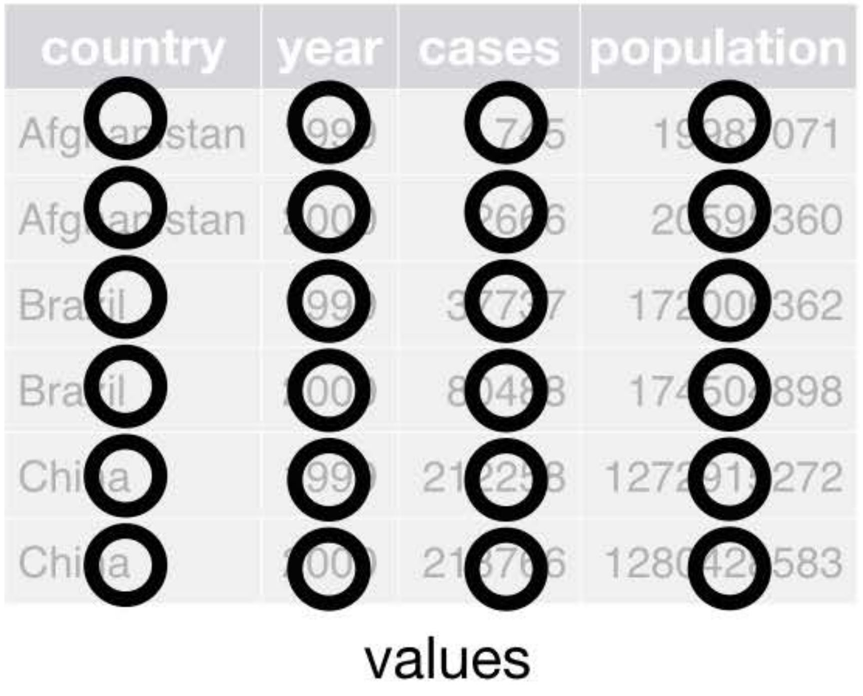


	A	B	C	D	E	F	G	H	I	J
1	geoid	state	Designated	county	Type	dec_score	SE_Flag	Population	medhhincome2014_tract	PovertyRate
2	01001020200	Alabama		Autauga	Low-Income Community	4		2,196	$ 41,107	24.0%
3	01001020300	Alabama		Autauga	Non-LIC Contiguous	6		3,136	$ 51,250	10.7%
4	01001020700	Alabama	1	Autauga	Low-Income Community	9		3,047	$ 45,234	19.0%
5	01001020802	Alabama		Autauga	Non-LIC Contiguous	10		10,743	$ 61,242	15.3%
6	01001021000	Alabama		Autauga	Non-LIC Contiguous	5		2,899	$ 49,567	15.1%
7	01001021100	Alabama		Autauga	Low-Income Community	6		3,247	$ 40,801	19.4%
8	01003010100	Alabama		Baldwin	Non-LIC Contiguous	6		4,013	$ 45,667	14.0%
9	01003010200	Alabama	1	Baldwin	Low-Income Community	9		3,067	$ 33,333	27.2%
10	01003010300	Alabama		Baldwin	Non-LIC Contiguous	10		8,079	$ 47,443	6.8%
11	01003010400	Alabama	1	Baldwin	Non-LIC Contiguous	9		4,578	$ 46,696	14.8%
12	01003010500	Alabama	1	Baldwin	Low-Income Community	8		5,115	$ 45,825	16.8%
13	01003010600	Alabama	1	Baldwin	Low-Income Community	9		3,503	$ 28,219	28.2%
14	01003010904	Alabama		Baldwin	Non-LIC Contiguous	10		6,523	$ 48,521	16.3%
15	01003010906	Alabama		Baldwin	Non-LIC Contiguous	10		5,272	$ 42,120	11.5%
16	01003011000	Alabama		Baldwin	Low-Income Community	10		3,885	$ 34,883	21.8%
17	01003011401	Alabama		Baldwin	Non-LIC Contiguous	10		10,021	$ 44,886	11.9%
18	01003011406	Alabama		Baldwin	Low-Income Community	10		3,810	$ 41,867	19.0%
19	01003011407	Alabama		Baldwin	Low-Income Community	10		4,970	$ 41,840	20.8%
20	01003011501	Alabama	1	Baldwin	Non-LIC Contiguous	9		5,947	$ 48,191	17.9%
21	01003011502	Alabama	1	Baldwin	Low-Income Community	10		11,575	$ 39,563	20.3%
22	01003011601	Alabama		Baldwin	Low-Income Community	10		6,640	$ 39,586	24.3%
23	01005950100	Alabama	1	Barbour	Low-Income Community	6		3,477	$ 38,571	33.2%
24	01005950200	Alabama		Barbour	Low-Income Community	1		4,404	$ 32,742	27.2%
25	01005950300	Alabama		Barbour	Low-Income Community	1		1,657	$ 29,911	36.1%
26	01005950400	Alabama		Barbour	Non-LIC Contiguous	1		3,693	$ 33,241	19.6%
27	01005950500	Alabama		Barbour	Low-Income Community	8		3,438	$ 38,859	19.1%
28	01005950600	Alabama		Barbour	Low-Income Community	4		2,003	$ 27,708	31.0%
29	01005950700	Alabama		Barbour	Low-Income Community	6		1,959	$ 28,409	31.3%
30	01005950800	Alabama		Barbour	Non-LIC Contiguous	5		2,195	$ 40,724	14.2%
31	01005950900	Alabama		Barbour	Low-Income Community	4		3,788	$ 27,027	28.5%
32	01007010001	Alabama		Bibb Cou	Low-Income Community	7		2,783	$ 44,422	9.6%



Census

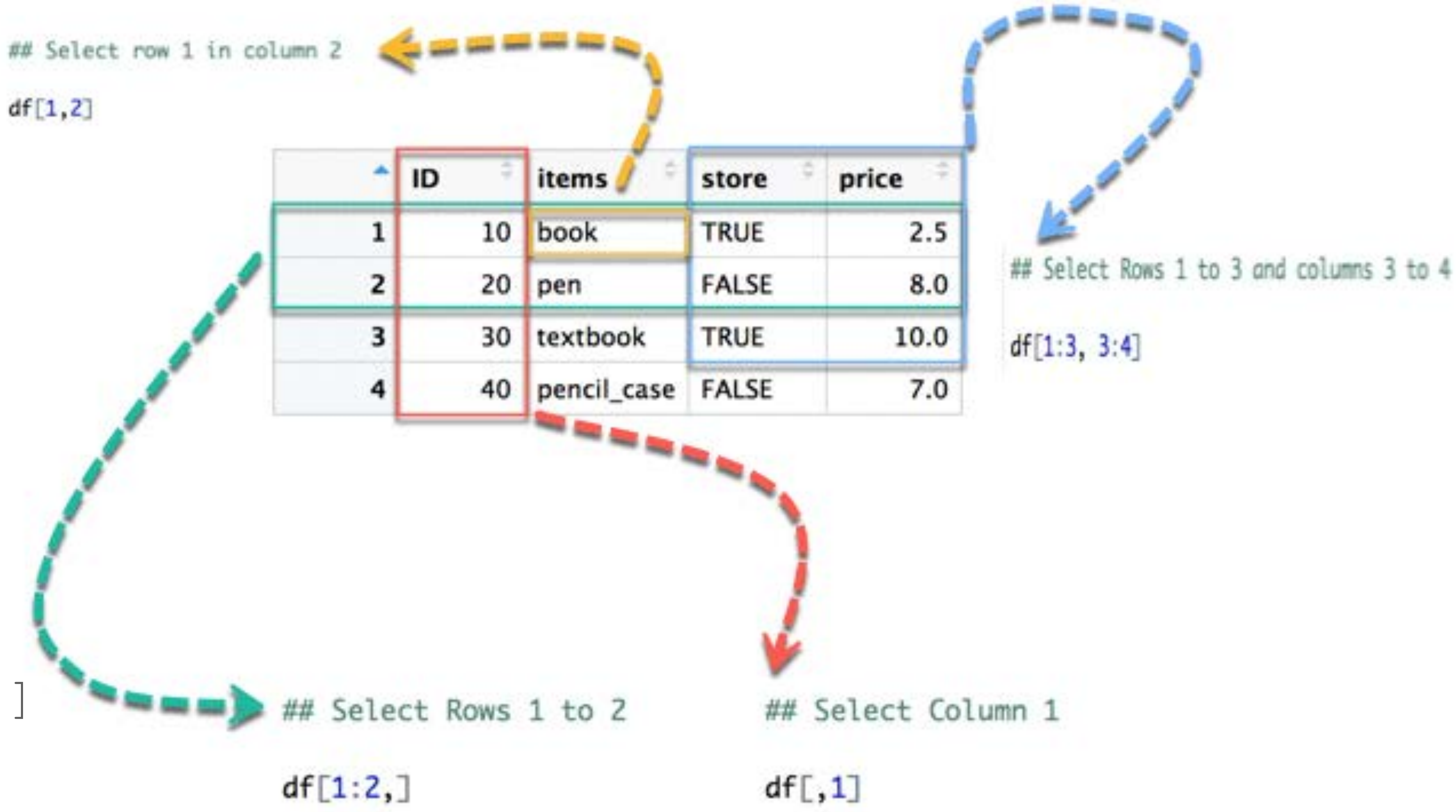
Tracts



Manipulating data

You’ve now practiced extracting data subsets using indexing

```
ozs[3,5]
ozs[ozs$designated == 1,]
ozs[ozs$PovertyRate > 20,]
ozs[,c(“GEOID”, “dec_score“)]
```



```
## Select row 1 in column 2
df[1,2]
```

	ID	items	store	price
1	10	book	TRUE	2.5
2	20	pen	FALSE	8.0
3	30	textbook	TRUE	10.0
4	40	pencil_case	FALSE	7.0

```
## Select Rows 1 to 3 and columns 3 to 4
df[1:3, 3:4]
```

```
## Select Rows 1 to 2
df[1:2,]
```

```
## Select Column 1
df[,1]
```

Manipulating data

We can create summaries from our extracts

```
mean(ozs$PovertyRate, na.rm=TRUE)

mean(ozs$medhhincome2014[ozs$designated == 1,])

max(ozs$PovertyRate[ozs$PovertyRate < 20,])
```

These are tough to read! We need to read from the inside out...

Manipulating data

We can create summaries from our extracts

```
mean(ozs$PovertyRate, na.rm=TRUE)

mean(ozs$medhhincome2014[ozs$designated == 1,])

max(ozs$PovertyRate[ozs$PovertyRate < 20,])
```

These are tough to read! We need to read from the inside out...

```
max(ozs$PovertyRate[ozs$PovertyRate < 20,])
```

From the ozs dataset, select the column poverty rate. Filter the poverty rate to those values where the poverty rate is less than 20. Find the maximum value of poverty rate

Enter dplyr

Enter dplyr

```
max(ozs$PovertyRate[ozs$PovertyRate < 20,])
```

From the ozs dataset, select the column poverty rate. Filter the poverty rate to those values where the poverty rate is less than 20. Find the maximum value of poverty rate

```
ozs %>%
select(PovertyRate) %>%
filter(PovertyRate < 20) %>%
max()
```

These do the same thing - dplyr notation isn't necessarily shorter, but it's much easier to see what's happening.

What’s that squiggly thing?

```
ozs %>%
select(PovertyRate) %>%
filter(PovertyRate < 20) %>%
max()
```

It’s a pipe (%>% or |>)

Pipes allow us to flow data through our code

`ozs %>%`	From the **ozs dataset**
`select(PovertyRate) %>%`	**Select** the column poverty rate.
`filter(PovertyRate < 20) %>%`	**Filter** the poverty rate to those values where the poverty rate is less than 20
`max()`	Find the **maximum** value of poverty rate

Your Lab

- Introduces dplyr verbs
- Revisits data summarization

Questions