

# Lecture 14: Reshaping data

# So far

- `select`: choose certain columns
- `filter`: choose certain rows
- `summarize`: calculate summary statistics
- `group_by`: group rows together
- `mutate`: create new columns
- `count`: count the number of rows
- `arrange`: re-order the rows

# Do dogs help exam stress?

- Data collected on 284 students at a mid-size Canadian university
- Students randomly assigned to one of three treatment groups: handler-only contact, indirect contact, and direct contact
- Well-being and ill-being measures recorded before and after treatment for each student
- Approach: compare pre/post measures of well-being and ill-being

# Recording well-being and ill-being measures

- Likert items for each well-being / ill-being measure
- Average the likert items to get a score for each measure
- E.g.:
  - Positive affect score is the average of 5 Likert items
  - Social connectedness is the average of 20 Likert items

# Example Likert item for social connectedness

“I am able to relate to my peers.”

- Strongly disagree (1)
- Disagree
- Somewhat disagree
- Somewhat agree
- Agree
- Strongly agree (6)

# The data

```
1 sc_data <- cleaned_data |>
2   select(RID, GroupAssignment, sc_pre, sc_post)
3
4 sc_data
```

	RID	GroupAssignment	sc_pre	sc_post
1	1	Control	3.900000	3.800000
2	2	Direct	5.150000	5.263158
3	3	Indirect	4.100000	4.150000
4	4	Control	4.650000	5.100000
5	5	Direct	3.650000	3.600000
6	6	Indirect	4.350000	4.650000
7	7	Control	4.750000	4.400000
8	8	Direct	4.600000	4.650000
9	9	Indirect	4.200000	4.150000
10	10	Control	5.800000	5.750000
11	11	Direct	4.400000	4.800000
12	12	Indirect	4.100000	4.250000
13	13	Control	5.400000	5.600000

**Question:** What if we want to fit a model with this data?

# Fitting a model

Want code that looks like this:

```
1 lm(score ~ GroupAssignment + stage, data = sc_data)
```

**Problem:** We don't have a column for stage! Or a column for score!

# pivot\_longer

```
1 sc_data |>
2   pivot_longer(cols = c(sc_pre, sc_post),
3                 names_to = "stage",
4                 values_to = "score")
```

# A tibble: 568 × 4

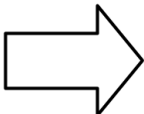
	RID	GroupAssignment	stage	score
	<int>	<chr>	<chr>	<dbl>
1	1	Control	sc_pre	3.9
2	1	Control	sc_post	3.8
3	2	Direct	sc_pre	5.15
4	2	Direct	sc_post	5.26
5	3	Indirect	sc_pre	4.1
6	3	Indirect	sc_post	4.15
7	4	Control	sc_pre	4.65
8	4	Control	sc_post	5.1
9	5	Direct	sc_pre	3.65
10	5	Direct	sc_post	3.6

# i 558 more rows



# pivot\_longer

id	bp1	bp2
A	100	120
B	140	115
C	120	125



id	measurement	value
A	bp1	100
A	bp2	120
B	bp1	140
B	bp2	115
C	bp1	120
C	bp2	125

```
1 df |>
2   pivot_longer(
3     cols = bp1:bp2,
4     names_to = "measurement",
5     values_to = "value"
6   )
```

(Image from *R for Data Science*)

# pivot\_longer

Another example:

```
# A tibble: 260 × 38
  Adult (15+) literacy rate ...1 `1975` `1976` `1977` `1978` `1979`
`1980` `1981`
  <chr>                                <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
<dbl>  <dbl>
1 Afghanistan                        NA      NA      NA      NA    4.99    NA
NA
2 Albania                            NA      NA      NA      NA    NA      NA
NA
3 Algeria                            NA      NA      NA      NA    NA      NA
NA
4 Andorra                            NA      NA      NA      NA    NA      NA
NA
5 Angola                             NA      NA      NA      NA    NA      NA
```

How might we want to restructure this data?

# pivot\_longer

```
# A tibble: 260 × 38
```

```
  Adult (15+) literacy rate ...1 `1975` `1976` `1977` `1978` `1979`  
`1980` `1981`  
  <chr>                <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  
<dbl>  <dbl>  
1 Afghanistan          NA      NA      NA      NA    4.99    NA  
NA  
2 Albania              NA      NA      NA      NA    NA      NA  
NA  
3 Algeria              NA      NA      NA      NA    NA      NA  
NA  
4 Andorra              NA      NA      NA      NA    NA      NA  
NA  
5 Angola              NA      NA      NA      NA    NA      NA
```

```
1 litF |>  
2   rename(country = starts_with("Adult")) |>  
3   pivot_longer(  
4     cols = -country,  
5     names_to = ...,  
6     values_to = ...  
7   )
```

# pivot\_longer

```
1 litF |>
2   rename(country = starts_with("Adult")) |>
3   pivot_longer(
4     cols = -country,
5     names_to = "year",
6     values_to = "literacy_rate"
7   ) |>
8   drop_na(literacy_rate)
```

# A tibble: 571 × 3

	country	year	literacy_rate
	<chr>	<chr>	<dbl>
1	Afghanistan	1979	4.99
2	Afghanistan	2011	13
3	Albania	2001	98.3
4	Albania	2008	94.7
5	Albania	2011	95.7
6	Algeria	1987	35.8
7	Algeria	2002	60.1
8	Algeria	2006	63.9
9	Angola	2001	54.2
10	Angola	2011	58.6

# i 561 more rows

# pivot\_longer

```
1 litF |>
2   rename(country = starts_with("Adult")) |>
3   pivot_longer(
4     cols = -country,
5     names_to = "year",
6     values_to = "literacy_rate",
7     values_drop_na = T
8   )
```

# A tibble: 571 × 3

	country	year	literacy_rate
	<chr>	<chr>	<dbl>
1	Afghanistan	1979	4.99
2	Afghanistan	2011	13
3	Albania	2001	98.3
4	Albania	2008	94.7
5	Albania	2011	95.7
6	Algeria	1987	35.8
7	Algeria	2002	60.1
8	Algeria	2006	63.9
9	Angola	2001	54.2
10	Angola	2011	58.6

# i 561 more rows

# Back to the dog data

```
1 sc_data |>
2   pivot_longer(cols = c(sc_pre, sc_post),
3                 names_to = "stage",
4                 values_to = "score")
```

```
# A tibble: 568 × 4
```

	RID	GroupAssignment	stage	score
	<int>	<chr>	<chr>	<dbl>
1	1	Control	sc_pre	3.9
2	1	Control	sc_post	3.8
3	2	Direct	sc_pre	5.15
4	2	Direct	sc_post	5.26
5	3	Indirect	sc_pre	4.1
6	3	Indirect	sc_post	4.15
7	4	Control	sc_pre	4.65
8	4	Control	sc_post	5.1
9	5	Direct	sc_pre	3.65
10	5	Direct	sc_post	3.6

```
# i 558 more rows
```

Does the `stage` column only contain information about stage?

# Back to the dog data

```
1 sc_data |>
2   pivot_longer(cols = c(sc_pre, sc_post),
3                 names_to = c("measurement", "stage"),
4                 names_sep = "_",
5                 values_to = "score")
```

# A tibble: 568 × 5

	RID	GroupAssignment	measurement	stage	score
	<int>	<chr>	<chr>	<chr>	<dbl>
1	1	Control	sc	pre	3.9
2	1	Control	sc	post	3.8
3	2	Direct	sc	pre	5.15
4	2	Direct	sc	post	5.26
5	3	Indirect	sc	pre	4.1
6	3	Indirect	sc	post	4.15
7	4	Control	sc	pre	4.65
8	4	Control	sc	post	5.1
9	5	Direct	sc	pre	3.65
10	5	Direct	sc	post	3.6

# i 558 more rows

# Working with all the measurements

```
1 cleaned_data |>
2   pivot_longer(cols = -c(RID, GroupAssignment),
3               names_to = c("measurement", "stage"),
4               names_sep = "_",
5               values_to = "score")
```

# A tibble: 4,544 × 5

	RID	GroupAssignment	measurement	stage	score
	<int>	<chr>	<chr>	<chr>	<dbl>
1	1	Control	pa	pre	3.2
2	1	Control	pa	post	3.8
3	1	Control	happiness	pre	2.33
4	1	Control	happiness	post	3.33
5	1	Control	sc	pre	3.9
6	1	Control	sc	post	3.8
7	1	Control	fs	pre	6.12
8	1	Control	fs	post	6
9	1	Control	stress	pre	2
10	1	Control	stress	post	2

# i 4,534 more rows



# Fitting a model

```
1 long_data <- cleaned_data |>
2   pivot_longer(cols = -c(RID, GroupAssignment),
3                 names_to = c("measurement", "stage"),
4                 names_sep = "_",
5                 values_to = "score")
6
7 lm(score ~ GroupAssignment + stage, data = long_data)
```

Call:

```
lm(formula = score ~ GroupAssignment + stage, data = long_data)
```

Coefficients:

(Intercept)	GroupAssignmentDirect
GroupAssignmentIndirect	
3.16307	-0.10118
-0.04836	
stagepre	
0.13805	

# Lengthing data in Python

```
1 import pandas as pd
2
3 df1 = pd.DataFrame({
4     'id' : ['A', 'B', 'C'],
5     'bp1' : [100, 140, 120],
6     'bp2' : [120, 115, 125]
7 })
8
9 df1
```

	id	bp1	bp2
0	A	100	120
1	B	140	115
2	C	120	125

```
1 df1.melt(id_vars = 'id', var_name = 'measurement',
2          value_name = 'value')
```

	id	measurement	value
0	A	bp1	100
1	B	bp1	140
2	C	bp1	120
3	A	bp2	120
4	B	bp2	115
5	C	bp2	125



# Reshaping data in R

```
1 litF |>
2   rename(country = starts_with("Adult")) |>
3   pivot_longer(
4     cols = -country,
5     names_to = "year",
6     values_to = "literacy_rate"
7   ) |>
8   drop_na(literacy_rate)
```

# A tibble: 571 × 3

	country	year	literacy_rate
	<chr>	<chr>	<dbl>
1	Afghanistan	1979	4.99
2	Afghanistan	2011	13
3	Albania	2001	98.3
4	Albania	2008	94.7
5	Albania	2011	95.7
6	Algeria	1987	35.8
7	Algeria	2002	60.1
8	Algeria	2006	63.9
9	Angola	2001	54.2
10	Angola	2011	58.6

# i 561 more rows

What would the corresponding Python code look like?



# Reshaping data in Python

```
1 litF = r.litF
2
3 # first, need to rename the first column
4 litF.rename(columns={litF.columns[0]: 'Country'})
```

	Country	1975	1976	...	2009	2010	2011
0	Afghanistan	NaN	NaN	...	NaN	NaN	13.00000
1	Albania	NaN	NaN	...	NaN	NaN	95.69148
2	Algeria	NaN	NaN	...	NaN	NaN	NaN
3	Andorra	NaN	NaN	...	NaN	NaN	NaN
4	Angola	NaN	NaN	...	NaN	NaN	58.60846
..	...	...	...	...	...	...	...
255	Virgin Islands (U.S.)	NaN	NaN	...	NaN	NaN	NaN
256	Yemen Arab Republic (Former)	NaN	NaN	...	NaN	NaN	NaN
257	Yemen Democratic (Former)	NaN	NaN	...	NaN	NaN	NaN
258	Yugoslavia	NaN	NaN	...	NaN	NaN	NaN
259	Åland	NaN	NaN	...	NaN	NaN	NaN

[260 rows x 38 columns]

# Reshaping data in Python

```
1 litF = r.litF
2
3 # rename the first column
4 # then melt to make the data longer
5 (litF.rename(columns={litF.columns[0]: 'Country'}))
6     .melt(id_vars = 'Country',
7           var_name = 'year',
8           value_name = 'literacy_rate'))
```

	Country	year	literacy_rate
0	Afghanistan	1975	NaN
1	Albania	1975	NaN
2	Algeria	1975	NaN
3	Andorra	1975	NaN
4	Angola	1975	NaN
...	...	...	...
9615	Virgin Islands (U.S.)	2011	NaN
9616	Yemen Arab Republic (Former)	2011	NaN
9617	Yemen Democratic (Former)	2011	NaN
9618	Yugoslavia	2011	NaN
9619	Åland	2011	NaN

[9620 rows x 3 columns]

# Reshaping data in Python

```
1 litF = r.litF
2
3 # rename the first column
4 # then melt to make the data longer
5 (litF.rename(columns={litF.columns[0]: 'Country'}))
6     .melt(id_vars = 'Country',
7           var_name = 'year',
8           value_name = 'literacy_rate')
9     .dropna())
```

	Country	year	literacy_rate
30	Burkina Faso	1975	3.182766
37	Central African Rep.	1975	8.399576
99	Kuwait	1975	48.015214
191	Turkey	1975	45.098921
197	United Arab Emirates	1975	38.124870
...	...	...	...
9562	Vanuatu	2011	81.553540
9564	West Bank and Gaza	2011	92.616180
9565	Vietnam	2011	91.383460
9566	Yemen, Rep.	2011	48.539050
9568	Zimbabwe	2011	80.065659

[571 rows x 3 columns]



What if I want to arrange the rows alphabetically by country?

# Reshaping data in Python

```
1 litF = r.litF
2
3 # rename the first column
4 # then melt to make the data longer
5 (litF.rename(columns={litF.columns[0]: 'Country'}))
6     .melt(id_vars = 'Country',
7           var_name = 'year',
8           value_name = 'literacy_rate')
9     .dropna()
10    .sort_values(by = ['Country', 'year']))
```

	Country	year	literacy_rate
1040	Afghanistan	1979	4.987460
9360	Afghanistan	2011	13.000000
6761	Albania	2001	98.252274
8581	Albania	2008	94.681814
9361	Albania	2011	95.691480
...	...	...	...
7227	Zambia	2002	61.839278
8527	Zambia	2007	51.786967
2028	Zimbabwe	1982	71.853928
4628	Zimbabwe	1992	78.517018
9568	Zimbabwe	2011	80.065659

[571 rows x 3 columns]



# Class activity

[https://sta279-s24.github.io/class\\_activities/ca\\_lecture\\_14.html](https://sta279-s24.github.io/class_activities/ca_lecture_14.html)

