Tidy data
Tidy data

“Tidy datasets are all alike, but every messy dataset is messy in its own way.”

— Hadley Wickham
Acknowledgements

Material for this session (and the next on joins) draws heavily on Chapters 9 and 10 of R4DS:
# Tidy data

Each variable has its own column.

<table>
<thead>
<tr>
<th>country</th>
<th>year</th>
<th>cases</th>
<th>population</th>
</tr>
</thead>
<tbody>
<tr>
<td>Afghanistan</td>
<td>1999</td>
<td>66</td>
<td>17,107,123</td>
</tr>
<tr>
<td>Afghanistan</td>
<td>2000</td>
<td>3566</td>
<td>20,553,600</td>
</tr>
<tr>
<td>Brazil</td>
<td>1999</td>
<td>3937</td>
<td>172,663,622</td>
</tr>
<tr>
<td>Brazil</td>
<td>2000</td>
<td>84688</td>
<td>174,548,984</td>
</tr>
<tr>
<td>China</td>
<td>1999</td>
<td>215598</td>
<td>1272,334,392</td>
</tr>
<tr>
<td>China</td>
<td>2000</td>
<td>215566</td>
<td>1288,858,833</td>
</tr>
</tbody>
</table>

Each observation has its own row.

Each value has its own cell.
Why tidy data?

- Consistent Data Structure
- Tools work in uniform way
- Tools are easier to use
- Exploits R’s vectorised nature
- Thinking process is simpler
- Easier wrangling, viz, modelling...
Tidy data

Let’s look at the data sets we loaded in the “Importing data” session.

Untidy or tidy?
Tidying data
```r
df %>%
gather(new1, new2, old_1, ... old_n)
```

- **Give name to variable which currently forms column names**
- **Give name to variable which is spread over cells**
- **Current set of columns that represent values NOT VARIABLES**
```r
gather

tb_cases %>>%
gather(year, cases, 2:5)
```

- **Give name to variable which currently forms column names**
- **Give name to variable which is spread over cells**
- **Current set of columns that represent values NOT VARIABLES**
Your turn: Tidy “pop”

\[
\text{pop} \ %>>% \\
\text{gather}(\text{new1, new2, old_1, \ldots old_n})
\]

*Give name to variable which currently forms column names*

*Give name to variable which is spread over cells*

*Current set of columns that represent values NOT VARIABLES*
Import Excel workbooks

Import `data_spread.xlsx`. Assign to object “table2”

for Excel workbooks (will require package: `readxl`)

for csvs
**spread**

does the opposite of gather:

```
table2 %>%
  spread(key = type, value = count)
```
Real life example:

You’ve found ONS population projections that you’d like to use for forecasting future healthcare utilisation. Load into R and tidy!
This work is licensed as

Creative Commons Attribution-ShareAlike 4.0 International

To view a copy of this license, visit

https://creativecommons.org/licenses/by-sa/4.0/
End
### Why tidy data?

```r
# A tibble: 6 x 6
  manufacturer model displ year cty hwy
  <chr>    <chr>   <dbl> <int> <int> <int>
1 audi     a4 quattro 2.00  2008  19   27
2 dodge    durango 4wd 3.90  1999  13   17
3 dodge    ram 1500 pickup 4wd 4.70  2008  12   16
4 ford     f150 pickup 4wd 4.60  2008  13   17
5 nissan   pathfinder 4wd 3.30  1999  15   17
6 subaru   forester awd 2.50  2008  18   23
```
Why tidy data?

# A tibble: 12 x 6

<table>
<thead>
<tr>
<th>manufacturer</th>
<th>model</th>
<th>displ</th>
<th>year</th>
<th>environment</th>
<th>mpg</th>
</tr>
</thead>
<tbody>
<tr>
<td>audi</td>
<td>a4 quattro</td>
<td>2.00</td>
<td>2008</td>
<td>city</td>
<td>19</td>
</tr>
<tr>
<td>audi</td>
<td>a4 quattro</td>
<td>2.00</td>
<td>2008</td>
<td>highway</td>
<td>27</td>
</tr>
<tr>
<td>dodge</td>
<td>durango 4wd</td>
<td>3.90</td>
<td>1999</td>
<td>city</td>
<td>13</td>
</tr>
<tr>
<td>dodge</td>
<td>ram 1500 pickup 4wd</td>
<td>4.70</td>
<td>2008</td>
<td>city</td>
<td>12</td>
</tr>
<tr>
<td>dodge</td>
<td>durango 4wd</td>
<td>3.90</td>
<td>1999</td>
<td>highway</td>
<td>17</td>
</tr>
<tr>
<td>dodge</td>
<td>ram 1500 pickup 4wd</td>
<td>4.70</td>
<td>2008</td>
<td>highway</td>
<td>16</td>
</tr>
<tr>
<td>ford</td>
<td>f150 pickup 4wd</td>
<td>4.60</td>
<td>2008</td>
<td>city</td>
<td>13</td>
</tr>
<tr>
<td>ford</td>
<td>f150 pickup 4wd</td>
<td>4.60</td>
<td>2008</td>
<td>highway</td>
<td>17</td>
</tr>
</tbody>
</table>