1. Each variable forms a column
2. Each observation forms a row
3. Each value in its own cell

```
install.packages("gapminder")
gap <- gapminder::gapminder
head(gap)
```

```
# A tibble: 6 x 6
       country continent  year lifeExp      pop gdpPercap
        <fctr>    <fctr> <int>   <dbl>    <int>     <dbl>
1 Afghanistan      Asia  1952  28.801  8425333  779.4453
2 Afghanistan      Asia  1957  30.332  9240934  820.8530
3 Afghanistan      Asia  1962  31.997 10267083  853.1007
4 Afghanistan      Asia  1967  34.020 11537966  836.1971
5 Afghanistan      Asia  1972  36.088 13079460  739.9811
6 Afghanistan      Asia  1977  38.438 14880372  786.1134
```

```
install.packages("gapminder")
gap <- gapminder::gapminder
head(gap)
```

```
# A tibble: 6 x 6
       country continent  year lifeExp      pop gdpPercap
        <fctr>    <fctr> <int>   <dbl>    <int>     <dbl>
1 Afghanistan      Asia  1952  28.801  8425333  779.4453
2 Afghanistan      Asia  1957  30.332  9240934  820.8530
3 Afghanistan      Asia  1962  31.997 10267083  853.1007
4 Afghanistan      Asia  1967  34.020 11537966  836.1971
5 Afghanistan      Asia  1972  36.088 13079460  739.9811
6 Afghanistan      Asia  1977  38.438 14880372  786.1134
```

what is the advantage of this

1. Consistency for consistency sake

2. Can take advantage of R tricks to make new variables that are derivatives of original variables

```
#> # A tibble: 12 × 4
#>       country  year       type      count
#>         <chr> <int>      <chr>      <int>
#> 1 Afghanistan  1999      cases        745
#> 2 Afghanistan  1999 population   19987071
#> 3 Afghanistan  2000      cases       2666
#> 4 Afghanistan  2000 population   20595360
#> 5      Brazil  1999      cases      37737
#> 6      Brazil  1999 population  172006362
#> # ... with 6 more rows
```

```
#> # A tibble: 6 × 3
#>       country  year              rate
#> *       <chr> <int>             <chr>
#> 1 Afghanistan  1999       745/19987071
#> 2 Afghanistan  2000      2666/20595360
#> 3      Brazil  1999    37737/172006362
#> 4      Brazil  2000    80488/174504898
#> 5       China  1999 212258/1272915272
#> 6       China  2000 213766/1280428583
```

```
#> # A tibble: 6 × 4
#>       country  year  cases population
#>         <chr> <int>  <int>      <int>
#> 1 Afghanistan  1999    745   19987071
#> 2 Afghanistan  2000   2666   20595360
#> 3      Brazil  1999  37737  172006362
#> 4      Brazil  2000  80488  174504898
#> 5       China  1999 212258 1272915272
#> 6       China  2000 213766 1280428583
```

LONG vs WIDE data

LONG

	country	year	avgtemp
1	Sweden	1994	6
2	Denmark	1994	6
3	Norway	1994	3
4	Sweden	1995	5
5	Denmark	1995	8
6	Norway	1995	11
7	Sweden	1996	7
8	Denmark	1996	8
9	Norway	1996	7

WIDE

	country	avgtemp.1994	avgtemp.1995	avgtemp.1996
1	Sweden	6	5	7
2	Denmark	6	8	8
3	Norway	3	11	7

LONG

	country	year	avgtemp
1	Sweden	1994	6
2	Denmark	1994	6
3	Norway	1994	3
4	Sweden	1995	5
5	Denmark	1995	8
6	Norway	1995	11
7	Sweden	1996	7
8	Denmark	1996	8
9	Norway	1996	7

WIDE

	country	avgtemp.1994	avgtemp.1995	avgtemp.1996
1	Sweden	6	5	7
2	Denmark	6	8	8
3	Norway	3	11	7

Why might we prefer one over the other?

LONG

	country	year	avgtemp
1	Sweden	1994	6
2	Denmark	1994	6
3	Norway	1994	3
4	Sweden	1995	5
5	Denmark	1995	8
6	Norway	1995	11
7	Sweden	1996	7
8	Denmark	1996	8
9	Norway	1996	7

WIDE

	country	avgtemp.1994	avgtemp.1995	avgtemp.1996
1	Sweden	6	5	7
2	Denmark	6	8	8
3	Norway	3	11	7

Why might we prefer one over the other?

The limits of long data

	variable	value
1	age	10
2	age	8
3	sex	M
4	sex	M
5	weight	2.2
6	weight	2.3

.

With this in mind
Let’s take a look at our data

Two principles

NEVER TOUCH THE RAW DATA

IF A MACHINE CAN’T READ IT,
IT DOESN’T EXIST

A quick note about project organization

```
install.packages("tidyverse")
library(tidyr)
```

gather()

takes multiple columns, and gathers them into key-value pairs: it makes "wide" data longer

Your data.frame or tibble

Name of variable whose values fill out cells

```
gather(data, key, value, columns)
```

Name of variable whose values form column names

Columns that you want to wrangle

```
too_wide <- tibble(
  country=c("Afghanistan", "Brazil", "China"),
  `1999`=c(745, 37737, 212258),
  `2000`=c(2666, 80488, 213766)
)
```

```
too_wide <- tibble(
  country=c("Afghanistan", "Brazil", "China"),
  `1999`=c(745, 37737, 212258),
  `2000`=c(2666, 80488, 213766)
)

gather(too_wide, key="year", value="cases",
`1999`, `2000`)
```

country	year	cases
Afghanistan	1999	745
Afghanistan	2000	2666
Brazil	1999	37737
Brazil	2000	80488
China	1999	212258
China	2000	213766

country	1999	2000
Afghanistan	745	2666
Brazil	37737	80488
China	212258	213766

table4

```
install.packages("tidyverse")
library(tidyr)
```

spread()

takes two columns (key & value) and spreads in to multiple columns, it makes "long" data wider

Your data.frame or tibble

Name of variable whose values **you want to** fill out cells

```
spread(data, key, value)
```

Name of variable whose values **you want to** form column names

```
too_long <- gather(too_wide, key="year",
value="cases", `1999`, `2000`)
```

```
spread(too_long, key="year", value=“cases")
```