Data tidying

Luis Francisco Gomez Lopez

2023-08-23

Contents

Tidy data

The concept of tidy data is describe in (Wickham, 2014):

- Each type of observational unit forms a table
- Each variable in a table is a column
- Each observation in a table is a row
- Each value in a table is a cell

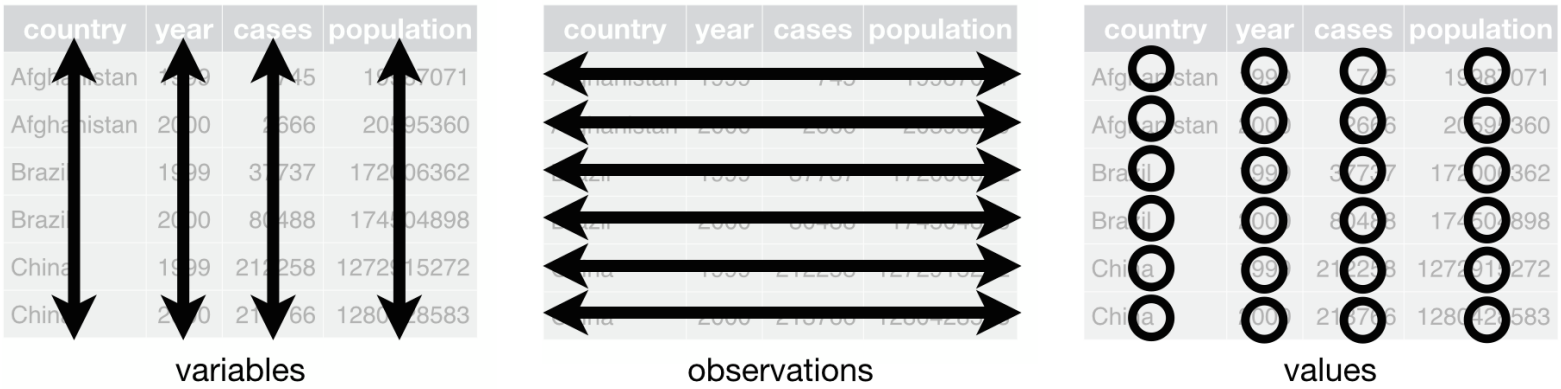


Figure 1: Tidy data table (Wickham et al., 2023, Chapter 6, Figure 6.1)

Tidy data

- **Table 1**: observational unit (country, year)
  - **Id**: country, year and **Variables**: continent, lifeExp, pop, gdpPercap

```
gapminder::gapminder |> head(n = 3) # data set from gapminder package
```

```
# A tibble: 3 x 6
  country     continent  year lifeExp      pop gdpPercap
  <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
1 Afghanistan Asia       1952    28.8  8425333      779.
2 Afghanistan Asia       1957    30.3  9240934      821.
3 Afghanistan Asia       1962    32.0 10267083      853.
```

- **Table 2**: observational unit (country)
  - **Id**: iso_alpha and **Variables**: country, iso_num

```
gapminder::country_codes |> head(n = 3)
```

```
# A tibble: 3 x 3
  country     iso_alpha iso_num
  <chr>       <chr>       <int>
1 Afghanistan AFG             4
2 Albania     ALB             8
3 Algeria     DZA            12
```

Lengthening data

- Why data is not tidy?
  - Data is often organized to facilitate some goal other than analysis like structuring it to make data entry easy
  - People are not familiar with the principles of tidy data
- Some tools to make data tidy[1]
  - `tidyr::pivot_longer`
  - `tidyr::pivot_wider`

[1] The package `tidyr`, that is included in the tidyverse, include more tools

Lengthening data

- Data in column names
  - Each row is related to a song (there is no id for each song)
  - `artist`, `track` and `date.entered` are variables that describe the song
  - `wk1` to `wk76` describe the rank of the song in each week
    - The cell values are the `rank` of the song in each week

```
tidyr::billboard |> head(n=3) # data set from tidyr package
```

```
# A tibble: 3 x 79
  artist      track date.entered   wk1   wk2   wk3   wk4   wk5   wk6   wk7   wk8
  <chr>       <chr> <date>       <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
1 2 Pac       Baby~ 2000-02-26      87    82    72    77    87    94    99    NA
2 2Ge+her     The ~ 2000-09-02      91    87    92    NA    NA    NA    NA    NA
3 3 Doors Do~ Kryp~ 2000-04-08      81    70    68    67    66    57    54    53
# i 68 more variables: wk9 <dbl>, wk10 <dbl>, wk11 <dbl>, wk12 <dbl>,
#   wk13 <dbl>, wk14 <dbl>, wk15 <dbl>, wk16 <dbl>, wk17 <dbl>, wk18 <dbl>,
#   wk19 <dbl>, wk20 <dbl>, wk21 <dbl>, wk22 <dbl>, wk23 <dbl>, wk24 <dbl>,
#   wk25 <dbl>, wk26 <dbl>, wk27 <dbl>, wk28 <dbl>, wk29 <dbl>, wk30 <dbl>,
#   wk31 <dbl>, wk32 <dbl>, wk33 <dbl>, wk34 <dbl>, wk35 <dbl>, wk36 <dbl>,
#   wk37 <dbl>, wk38 <dbl>, wk39 <dbl>, wk40 <dbl>, wk41 <dbl>, wk42 <dbl>,
#   wk43 <dbl>, wk44 <dbl>, wk45 <dbl>, wk46 <dbl>, wk47 <dbl>, wk48 <dbl>, ...
```

Lengthening data

- Solution to make data tidy
  - Not all songs stay in the top 100 so `NA` values are not drop but you can drop them with the option `values_drop_na = TRUE`

```
billboard |>
  pivot_longer(
    cols = starts_with(match = "wk"), # columns to pivot
    names_to = 'week', # variable name related to the column names
    values_to = 'rank' # variable name related to data stored in cell values
  )
```

```
# A tibble: 24,092 x 5
   artist track                   date.entered week   rank
   <chr>  <chr>                   <date>       <chr> <dbl>
 1 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk1      87
 2 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk2      82
 3 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk3      72
 4 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk4      77
 5 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk5      87
 6 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk6      94
 7 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk7      99
 8 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk8      NA
 9 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk9      NA
10 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk10     NA
# i 24,082 more rows
```

Lengthening data

- Dropping `NA` values and parsing values that are not of correct type
  - `week` is really a numeric variable

```
billboard |>
  pivot_longer(
    cols = starts_with(match = "wk"), # columns to pivot
    names_to = 'week', # variable name related to the column names
    values_to = 'rank', # variable name related to data stored in cell values
    values_drop_na = TRUE,
  ) |>
  mutate(week = readr::parse_number(week)) # parse the first number it finds
```

```
# A tibble: 5,307 x 5
   artist  track                   date.entered  week  rank
   <chr>   <chr>                   <date>       <dbl> <dbl>
 1 2 Pac   Baby Don't Cry (Keep... 2000-02-26       1    87
 2 2 Pac   Baby Don't Cry (Keep... 2000-02-26       2    82
 3 2 Pac   Baby Don't Cry (Keep... 2000-02-26       3    72
 4 2 Pac   Baby Don't Cry (Keep... 2000-02-26       4    77
 5 2 Pac   Baby Don't Cry (Keep... 2000-02-26       5    87
 6 2 Pac   Baby Don't Cry (Keep... 2000-02-26       6    94
 7 2 Pac   Baby Don't Cry (Keep... 2000-02-26       7    99
 8 2Ge+her The Hardest Part Of ... 2000-09-02       1    91
 9 2Ge+her The Hardest Part Of ... 2000-09-02       2    87
10 2Ge+her The Hardest Part Of ... 2000-09-02       3    92
# i 5,297 more rows
```

Lengthening data

- How pivoting works?

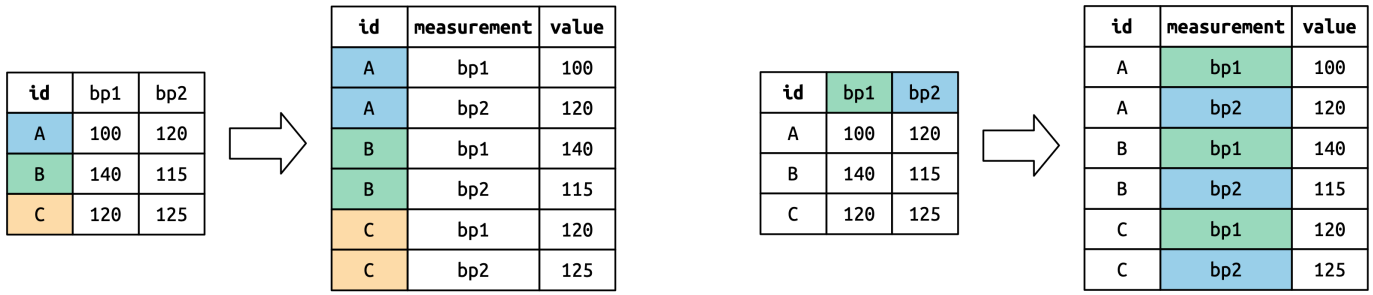
```
df <- tibble::tribble(
  ~id,  ~bp1, ~bp2,
   "A",  100,  120,
   "B",  140,  115,
   "C",  120,  125
)

df |>
  pivot_longer(
    cols = bp1:bp2,
    names_to = "measurement",
    values_to = "value"
  )
```

```
# A tibble: 6 x 3
  id    measurement value
  <chr> <chr>       <dbl>
1 A     bp1           100
2 A     bp2           120
3 B     bp1           140
4 B     bp2           115
5 C     bp1           120
6 C     bp2           125
```

Lengthening data

- How pivoting works?



(a) ID

(b) Name

(c) Value

Figure 2: Pivoting

Lengthening data

- Many variables in column names: multiple pieces of information included into the column names
  - `sp_m_014 - rel_f_65`: 3 variables included into the column names (see `?tidyr::who2`)
    - Method of diagnosis, Gender and Age group

```
who2 |> head(n=5)
```

```
# A tibble: 5 x 58
  country      year sp_m_014 sp_m_1524 sp_m_2534 sp_m_3544 sp_m_4554 sp_m_5564
  <chr>       <dbl>    <dbl>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>
1 Afghanistan  1980       NA        NA        NA        NA        NA        NA
2 Afghanistan  1981       NA        NA        NA        NA        NA        NA
3 Afghanistan  1982       NA        NA        NA        NA        NA        NA
4 Afghanistan  1983       NA        NA        NA        NA        NA        NA
5 Afghanistan  1984       NA        NA        NA        NA        NA        NA
# i 50 more variables: sp_m_65 <dbl>, sp_f_014 <dbl>, sp_f_1524 <dbl>,
#   sp_f_2534 <dbl>, sp_f_3544 <dbl>, sp_f_4554 <dbl>, sp_f_5564 <dbl>,
#   sp_f_65 <dbl>, sn_m_014 <dbl>, sn_m_1524 <dbl>, sn_m_2534 <dbl>,
#   sn_m_3544 <dbl>, sn_m_4554 <dbl>, sn_m_5564 <dbl>, sn_m_65 <dbl>,
#   sn_f_014 <dbl>, sn_f_1524 <dbl>, sn_f_2534 <dbl>, sn_f_3544 <dbl>,
#   sn_f_4554 <dbl>, sn_f_5564 <dbl>, sn_f_65 <dbl>, ep_m_014 <dbl>,
#   ep_m_1524 <dbl>, ep_m_2534 <dbl>, ep_m_3544 <dbl>, ep_m_4554 <dbl>, ...
```

Lengthening data

- Solution to many variables in column names

```
who2 |>
  pivot_longer(
    cols = !c(country, year), # an alternative to select the columns to pivot
    names_to = c('diagnosis', 'gender', 'age_group'), # we can improve the label for the age_group
    names_sep = '_',
    values_to = 'count'
  )
```

```
# A tibble: 405,440 x 6
   country      year diagnosis gender age_group count
   <chr>       <dbl> <chr>     <chr>  <chr>     <dbl>
 1 Afghanistan  1980 sp        m      014          NA
 2 Afghanistan  1980 sp        m      1524         NA
 3 Afghanistan  1980 sp        m      2534         NA
 4 Afghanistan  1980 sp        m      3544         NA
 5 Afghanistan  1980 sp        m      4554         NA
 6 Afghanistan  1980 sp        m      5564         NA
 7 Afghanistan  1980 sp        m      65           NA
 8 Afghanistan  1980 sp        f      014          NA
 9 Afghanistan  1980 sp        f      1524         NA
10 Afghanistan  1980 sp        f      2534         NA
# i 405,430 more rows
```

Widening data

- An observation is spread across multiple rows

```
cms_patient_experience |> head(n=5)
```

```
# A tibble: 5 x 5
  org_pac_id org_nm                     measure_cd  measure_title       prf_rate
  <chr>      <chr>                      <chr>       <chr>                  <dbl>
1 0446157747 USC CARE MEDICAL GROUP INC CAHPS_GRP_1 CAHPS for MIPS SSM~       63
2 0446157747 USC CARE MEDICAL GROUP INC CAHPS_GRP_2 CAHPS for MIPS SSM~       87
3 0446157747 USC CARE MEDICAL GROUP INC CAHPS_GRP_3 CAHPS for MIPS SSM~       86
4 0446157747 USC CARE MEDICAL GROUP INC CAHPS_GRP_5 CAHPS for MIPS SSM~       57
5 0446157747 USC CARE MEDICAL GROUP INC CAHPS_GRP_8 CAHPS for MIPS SSM~       85
```

Widening data

- Solution to make data tidy

```
cms_patient_experience |>
  pivot_wider(id_cols = starts_with(match='org'),
              names_from = measure_cd,
              values_from = prf_rate)
```

```
# A tibble: 95 x 8
   org_pac_id org_nm CAHPS_GRP_1 CAHPS_GRP_2 CAHPS_GRP_3 CAHPS_GRP_5 CAHPS_GRP_8
   <chr>      <chr>        <dbl>       <dbl>       <dbl>       <dbl>       <dbl>
 1 0446157747 USC C~          63          87          86          57          85
 2 0446162697 ASSOC~          59          85          83          63          88
 3 0547164295 BEAVE~          49          NA          75          44          73
 4 0749333730 CAPE ~          67          84          85          65          82
 5 0840104360 ALLIA~          66          87          87          64          87
 6 0840109864 REX H~          73          87          84          67          91
 7 0840513552 SCL H~          58          83          76          58          78
 8 0941545784 GRITM~          46          86          81          54          NA
 9 1052612785 COMMU~          65          84          80          58          87
10 1254237779 OUR L~          61          NA          NA          65          NA
# i 85 more rows
# i 1 more variable: CAHPS_GRP_12 <dbl>
```

Widening data

- How pivoting works?

```
df <- tribble(
  ~id, ~measurement, ~value,
  "A",        "bp1",    100,
  "B",        "bp1",    140,
  "B",        "bp2",    115,
  "A",        "bp2",    120,
  "A",        "bp3",    105
)

df |>
  pivot_wider(
    id_cols = id, # columns that uniquely identify each observation
    names_from = measurement,
    values_from = value # in this case NA values are created
  )
```

```
# A tibble: 2 x 4
  id      bp1   bp2   bp3
  <chr> <dbl> <dbl> <dbl>
1 A       100   120   105
2 B       140   115    NA
```

Widening data

- How pivoting works?
  - Case where an observation has 2 values

```
df <- tribble(
  ~id, ~measurement, ~value,
  "A",        "bp1",    100, # A measure apply to A generate 2 values
  "A",        "bp1",    102,
  "A",        "bp2",    120,
  "B",        "bp1",    140,
  "B",        "bp2",    115
)
```

Widening data

- Attempting to wide the data when an observation has 2 values

```
df |>
  pivot_wider(
    id_cols = id,
    names_from = measurement,
    values_from = value
  )
```

```
Warning: Values from `value` are not uniquely identified; output will contain list-cols.
* Use `values_fn = list` to suppress this warning.
* Use `values_fn = {summary_fun}` to summarise duplicates.
* Use the following dplyr code to identify duplicates.
  {data} %>%
  dplyr::group_by(id, measurement) %>%
  dplyr::summarise(n = dplyr::n(), .groups = "drop") %>%
  dplyr::filter(n > 1L)
```

```
# A tibble: 2 x 3
  id    bp1       bp2
  <chr> <list>    <list>
1 A     <dbl [2]> <dbl [1]>
2 B     <dbl [1]> <dbl [1]>
```

Widening data

- Apply the recommendation pointed out in the warning
  - Using the recommendation we identify that patient A has to values when applying measurement bp1

```
df %>%
  dplyr::group_by(id, measurement) %>%
  dplyr::summarise(n = dplyr::n(), .groups = "drop") %>%
  dplyr::filter(n > 1L)
```

```
# A tibble: 1 x 3
  id    measurement     n
  <chr> <chr>       <int>
1 A     bp1             2
```

References I

Wickham, H. (2014). Tidy Data. *Journal of Statistical Software*, *59*(10). https://doi.org/10.18637/jss.v059.i10

Wickham, H., Çetinkaya-Rundel, M., & Grolemund, G. (2023). *R for data science: Import, tidy, transform, visualize, and model data* (2nd edition). O'Reilly Media, Inc. https://r4ds.hadley.nz/