Tidying Data Assignment

Tyler Kephart

2024-08-14

Load the tidyverse library

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

Question 1. The built in billboard dataset is not tidy. Describe why it is not tidy and then tidy the dataset.

```
# First gather all the week entries into a row for each week for each song
# (where there is an entry)
bb <- billboard %>%
    pivot_longer(
        cols = starts_with("wk"),
        names_to = "week",
        values_to = "rank",
        names_prefix = "wk"
        )
bb
```

```
## # 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   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 2 Pac  Baby Don't Cry (Keep... 2000-02-26   8        NA
##  9 2 Pac  Baby Don't Cry (Keep... 2000-02-26   9        NA
## 10 2 Pac  Baby Don't Cry (Keep... 2000-02-26   10       NA
## # i 24,082 more rows
```

```r
# Then, convert the week variable to a number and...
bb$week <- parse_integer(bb$week)
# figure out the date corresponding to each week on the chart
bb <- bb %>%
    # dropped NAs because it meant the song wasn't on the billboard
    drop_na(rank) %>%
    mutate(date = date.entered + (7 * (week - 1)))
bb
```

```
## # A tibble: 5,307 x 6
##    artist  track                   date.entered  week  rank date
##    <chr>   <chr>                   <date>       <int> <dbl> <date>
##  1 2 Pac   Baby Don’t Cry (Keep... 2000-02-26       1    87 2000-02-26
##  2 2 Pac   Baby Don’t Cry (Keep... 2000-02-26       2    82 2000-03-04
##  3 2 Pac   Baby Don’t Cry (Keep... 2000-02-26       3    72 2000-03-11
##  4 2 Pac   Baby Don’t Cry (Keep... 2000-02-26       4    77 2000-03-18
##  5 2 Pac   Baby Don’t Cry (Keep... 2000-02-26       5    87 2000-03-25
##  6 2 Pac   Baby Don’t Cry (Keep... 2000-02-26       6    94 2000-04-01
##  7 2 Pac   Baby Don’t Cry (Keep... 2000-02-26       7    99 2000-04-08
##  8 2Ge+her The Hardest Part Of ... 2000-09-02       1    91 2000-09-02
##  9 2Ge+her The Hardest Part Of ... 2000-09-02       2    87 2000-09-09
## 10 2Ge+her The Hardest Part Of ... 2000-09-02       3    92 2000-09-16
## # i 5,297 more rows
```

```r
# Sort the data by artist, track and week.
# Here are what your first entries should be (formatting can be different):
#>  A tibble: 5,307 x 5
#   artist  track                   date.entered  week  rank   date
 #    <chr>   <chr>                   <date>       <int> <dbl>   <date>
 #  1 2 Pac   Baby Don't Cry (Keep... 2000-02-26       1    87 2000-02-26
 #  2 2 Pac   Baby Don't Cry (Keep... 2000-02-26       2    82 2000-03-04
 #  3 2 Pac   Baby Don't Cry (Keep... 2000-02-26       3    72 2000-03-11
 #  4 2 Pac   Baby Don't Cry (Keep... 2000-02-26       4    77 2000-03-18
 #  5 2 Pac   Baby Don't Cry (Keep... 2000-02-26       5    87 2000-03-25
 #  6 2 Pac   Baby Don't Cry (Keep... 2000-02-26       6    94 2000-04-01
 #  7 2 Pac   Baby Don't Cry (Keep... 2000-02-26       7    99 2000-04-08
 #  8 2Ge+her The Hardest Part Of ... 2000-09-02       1    91 2000-09-02
 #  9 2Ge+her The Hardest Part Of ... 2000-09-02       2    87 2000-09-09
 # 10 2Ge+her The Hardest Part Of ... 2000-09-02       3    92 2000-09-16
 # ... with 5,297 more rows

# arranged to make sure that all the data was sorted properly
bb <- bb %>% arrange(artist, track, week)
bb
```

```
## # A tibble: 5,307 x 6
##    artist  track                   date.entered  week  rank date
##    <chr>   <chr>                   <date>       <int> <dbl> <date>
##  1 2 Pac   Baby Don’t Cry (Keep... 2000-02-26       1    87 2000-02-26
##  2 2 Pac   Baby Don’t Cry (Keep... 2000-02-26       2    82 2000-03-04
##  3 2 Pac   Baby Don’t Cry (Keep... 2000-02-26       3    72 2000-03-11
##  4 2 Pac   Baby Don’t Cry (Keep... 2000-02-26       4    77 2000-03-18
##  5 2 Pac   Baby Don’t Cry (Keep... 2000-02-26       5    87 2000-03-25
##  6 2 Pac   Baby Don’t Cry (Keep... 2000-02-26       6    94 2000-04-01
##  7 2 Pac   Baby Don’t Cry (Keep... 2000-02-26       7    99 2000-04-08
##  8 2Ge+her The Hardest Part Of ... 2000-09-02       1    91 2000-09-02
##  9 2Ge+her The Hardest Part Of ... 2000-09-02       2    87 2000-09-09
## 10 2Ge+her The Hardest Part Of ... 2000-09-02       3    92 2000-09-16
## # i 5,297 more rows
```

Question 2. Tidy the “fish_encounters” dataset of fish spotting by monitoring stations. Make the NA into 0 using the option “values_fill = list(seen = 0)”

```
fish_encounters %>%
    complete(fish, station, fill = list(seen = 0))
```

```
## # A tibble: 209 x 3
##    fish  station  seen
##    <fct> <fct>   <int>
##  1 4842  Release     1
##  2 4842  I80_1       1
##  3 4842  Lisbon      1
##  4 4842  Rstr        1
##  5 4842  Base_TD     1
##  6 4842  BCE         1
##  7 4842  BCW         1
##  8 4842  BCE2        1
##  9 4842  BCW2        1
## 10 4842  MAE         1
## # i 199 more rows
```

```
# you can verify by viewing more of the data like adding "%>% print(n = Inf)"
```

Question 3. Import the flowers1 dataset. Tidy and pivot the data. Hint: use “read_csv2()” to read in the dataset

```
# read in the data from semi-colon separated values file
flowers1 <- read_csv2("flowers1.csv")
```

```
## i Using "’,’" as decimal and "’.’" as grouping mark. Use ‘read_delim()‘ for more control.

## Rows: 48 Columns: 4
## -- Column specification --------------------------------------------------------
## Delimiter: ";"
## chr (1): Variable
## dbl (3): Time, replication, Value
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
# pivot so that the flowers value and intensity value are in each observation
flowers1 %>% pivot_wider(names_from = Variable, values_from = Value)
```

```
## # A tibble: 24 x 4
##     Time replication Flowers Intensity
##    <dbl>       <dbl>   <dbl>     <dbl>
##  1     1           1    62.3       150
##  2     1           2    77.4       150
##  3     1           3    55.3       300
##  4     1           4    54.2       300
##  5     1           5    49.6       450
##  6     1           6    61.9       450
##  7     1           7    39.4       600
##  8     1           8    45.7       600
##  9     1           9    31.3       750
## 10     1          10    44.9       750
## # i 14 more rows
```

Question 4. Import the flowers2 dataset. Tidy the dataset by turning the one column into 3 separate columns

```
# read in the data from semi-colon separated values file
# this will separate the time column
flowers2 <- read_csv2("flowers2.csv")
```

```
## i Using "’,’" as decimal and "’.’" as grouping mark. Use ‘read_delim()‘ for more control.
```

```
## Rows: 24 Columns: 2
## -- Column specification --------------------------------------------------------
## Delimiter: ";"
## chr (1): Flowers/Intensity
## dbl (1): Time
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
# separate the Flowers/Intensity into 2 cols and add replication col
flowers2 %>%
    separate(
        col = `Flowers/Intensity`,
        into = c("Flowers", "Intensity"),
        sep = "/",
        remove = TRUE,
        convert = TRUE
        ) %>%
    mutate(replication = as.numeric(rownames(.)) - (12 * (Time - 1)))
```

```
## # A tibble: 24 x 4
##    Flowers Intensity  Time replication
##      <dbl>     <int> <dbl>       <dbl>
##  1    62.3       150     1           1
##  2    77.4       150     1           2
##  3    55.3       300     1           3
##  4    54.2       300     1           4
##  5    49.6       450     1           5
##  6    61.9       450     1           6
##  7    39.4       600     1           7
##  8    45.7       600     1           8
##  9    31.3       750     1           9
## 10    44.9       750     1          10
## # i 14 more rows
```

Question 5. In the following dataset, turn the implicit missing values to explicit

```
output <- tibble(
        treatment = c("a", "b", "a", "c", "b"),
        gender = factor(c("M", "F", "F", "M", "M"),
        levels = c("M", "F", "O")),
        return = c(1.5, 0.75,  0.5,  1.8,  NA)
    )
output
```

```
## # A tibble: 5 x 3
##   treatment gender return
##   <chr>     <fct>   <dbl>
## 1 a         M        1.5
## 2 b         F        0.75
## 3 a         F        0.5
## 4 c         M        1.8
## 5 b         M       NA
```

```
# used complete to add the missing observations
output %>% complete(treatment, gender)
```

```
## # A tibble: 9 x 3
##   treatment gender return
##   <chr>     <fct>   <dbl>
## 1 a         M        1.5
## 2 a         F        0.5
## 3 a         O       NA
## 4 b         M       NA
## 5 b         F        0.75
## 6 b         O       NA
## 7 c         M        1.8
## 8 c         F       NA
## 9 c         O       NA
```

Question 6. Import the weather dataset as weather. Use "pivot_longer()" to put the days all in one column, then use "pivot_wider" to separate tmax and tmin into separate columns. Print the summary of the final resulting dataset

```
# read in the csv data
weather <- read_csv("weather.csv")
```

```
## Rows: 22 Columns: 35
## -- Column specification -------------------------------------------------------
## Delimiter: ","
## chr  (2): id, element
## dbl (25): year, month, d1, d2, d3, d4, d5, d6, d7, d8, d10, d11, d13, d14, d...
## lgl  (8): d9, d12, d18, d19, d20, d21, d22, d24
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# pivot the days into one col
weather <- weather %>%
    pivot_longer(
        cols = starts_with("d"),
        names_to = "day",
        values_to = "value",
        names_prefix = "d",
        names_transform = list(day = as.integer)
        )
weather
```

```
## # A tibble: 682 x 6
##    id        year month element   day value
##    <chr>    <dbl> <dbl> <chr>   <int> <dbl>
##  1 MX17004   2010     1 tmax        1    NA
##  2 MX17004   2010     1 tmax        2    NA
##  3 MX17004   2010     1 tmax        3    NA
##  4 MX17004   2010     1 tmax        4    NA
##  5 MX17004   2010     1 tmax        5    NA
##  6 MX17004  2010     1 tmax        6    NA
##  7 MX17004  2010     1 tmax        7    NA
##  8 MX17004  2010     1 tmax        8    NA
##  9 MX17004  2010     1 tmax        9    NA
## 10 MX17004  2010     1 tmax       10    NA
## # i 672 more rows
```

```
# pivot the element into tmax and tmin
weather <- weather %>%
    pivot_wider(
        names_from = element,
        values_from = value
        )
weather
```

```
## # A tibble: 341 x 6
##    id       year month   day  tmax  tmin
##    <chr>   <dbl> <dbl> <int> <dbl> <dbl>
##  1 MX17004  2010     1     1    NA    NA
##  2 MX17004  2010     1     2    NA    NA
##  3 MX17004  2010     1     3    NA    NA
##  4 MX17004  2010     1     4    NA    NA
##  5 MX17004  2010     1     5    NA    NA
##  6 MX17004  2010     1     6    NA    NA
##  7 MX17004  2010     1     7    NA    NA
##  8 MX17004  2010     1     8    NA    NA
##  9 MX17004  2010     1     9    NA    NA
## 10 MX17004  2010     1    10    NA    NA
## # i 331 more rows
```

```
# print summary
weather %>% summary()
```

```
##       id                 year          month             day    
##  Length:341         Min.   :2010   Min.   : 1.000   Min.   : 1  
##  Class :character   1st Qu.:2010   1st Qu.: 3.000   1st Qu.: 8  
##  Mode  :character   Median :2010   Median : 6.000   Median :16  
##                     Mean   :2010   Mean   : 6.273   Mean   :16  
##                     3rd Qu.:2010   3rd Qu.:10.000   3rd Qu.:24  
##                     Max.   :2010   Max.   :12.000   Max.   :31  
##                                                                 
##       tmax            tmin      
##  Min.   :24.10   Min.   : 7.90  
##  1st Qu.:27.80   1st Qu.:13.40  
##  Median :29.00   Median :15.00  
##  Mean   :29.19   Mean   :14.65  
##  3rd Qu.:29.90   3rd Qu.:16.50  
##  Max.   :36.30   Max.   :18.20  
##  NA's   :308     NA's   :308    
```

Question 7. Load the built in “anscombe” data frame and use “pivot_longer()” to separate all the x and y columns and categorize them into 4 sets

```
anscombe %>%
    pivot_longer(
        # get the columns to separate
        cols = (starts_with("x") | starts_with("y")),
        # .value will name the cols x or y respective to their values
        names_to = c(".value", "set"),
        # names_pattern will match the column pattern x or y, set 1 to 4
        names_pattern = "(x|y)([1-4])"
        )
```

```
## # A tibble: 44 x 3
##    set       x     y
##    <chr> <dbl> <dbl>
##  1 1        10  8.04
##  2 2        10  9.14
##  3 3        10  7.46
##  4 4         8  6.58
##  5 1         8  6.95
##  6 2         8  8.14
##  7 3         8  6.77
##  8 4         8  5.76
##  9 1        13  7.58
## 10 2        13  8.74
## # i 34 more rows
```