Importing tabular text data

Contents

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.4     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

1 read_delim() and friends

1.1 Data with delimiters

In general, `read_delim()` will do the trick (or one of the wrappers `read_csv()` for commas, `read_csv2()` for semi-colons, `read_tsv()` for tabulator, and `read_table()` for white space as separators):

```
read_delim("data/text1.csv", delim = ",")
```

```
## Rows: 150 Columns: 5
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Species
## dbl (4): Sepal.Length, Sepal.Width, Petal.Length, Petal.Width
```

```
##
## 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.
```

```
## # A tibble: 150 x 5
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##           <dbl>       <dbl>        <dbl>       <dbl> <chr>
##  1          5.1         3.5          1.4         0.2 setosa
##  2          4.9         3            1.4         0.2 setosa
##  3          4.7         3.2          1.3         0.2 setosa
##  4          4.6         3.1          1.5         0.2 setosa
##  5          5           3.6          1.4         0.2 setosa
##  6          5.4         3.9          1.7         0.4 setosa
##  7          4.6         3.4          1.4         0.3 setosa
##  8          5           3.4          1.5         0.2 setosa
##  9          4.4         2.9          1.4         0.2 setosa
## 10          4.9         3.1          1.5         0.1 setosa
## # ... with 140 more rows
```

```r
read_csv("data/text1.csv")
```

```
## Rows: 150 Columns: 5
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Species
## dbl (4): Sepal.Length, Sepal.Width, Petal.Length, Petal.Width
```

```
##
## 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.
```

```
## # A tibble: 150 x 5
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##           <dbl>       <dbl>        <dbl>       <dbl> <chr>
##  1          5.1         3.5          1.4         0.2 setosa
##  2          4.9         3            1.4         0.2 setosa
##  3          4.7         3.2          1.3         0.2 setosa
##  4          4.6         3.1          1.5         0.2 setosa
##  5          5           3.6          1.4         0.2 setosa
##  6          5.4         3.9          1.7         0.4 setosa
##  7          4.6         3.4          1.4         0.3 setosa
##  8          5           3.4          1.5         0.2 setosa
##  9          4.4         2.9          1.4         0.2 setosa
## 10          4.9         3.1          1.5         0.1 setosa
## # ... with 140 more rows
```

These functions try to guess the data types. If this does not work automagically, they can be specified:

```
x = read_csv("data/text1.csv", col_types = "cdi-l")
```

```
## Warning: One or more parsing issues, see `problems()` for details
```

```
x
```

```
## # A tibble: 150 x 4
##    Sepal.Length Sepal.Width Petal.Length Species
##    <chr>              <dbl>        <int> <lgl>
##  1 5.1                  3.5           NA NA
##  2 4.9                  3             NA NA
##  3 4.7                  3.2           NA NA
##  4 4.6                  3.1           NA NA
##  5 5                    3.6           NA NA
##  6 5.4                  3.9           NA NA
##  7 4.6                  3.4           NA NA
##  8 5                    3.4           NA NA
##  9 4.4                  2.9           NA NA
## 10 4.9                  3.1           NA NA
## # ... with 140 more rows
```

Use `problems(x)` to diagnose issues:

```
problems(x)
```

```
## # A tibble: 290 x 5
##      row   col expected           actual file
##    <int> <int> <chr>              <chr>  <chr>
##  1     2     3 an integer         1.4    C:/Users/exploFH/Nextcloud/BWI/05_SEM/~
##  2     2     3 an integer         1.3    C:/Users/exploFH/Nextcloud/BWI/05_SEM/~
##  3     2     3 an integer         1.5    C:/Users/exploFH/Nextcloud/BWI/05_SEM/~
##  4     2     3 an integer         1.7    C:/Users/exploFH/Nextcloud/BWI/05_SEM/~
##  5     2     5 1/0/T/F/TRUE/FALSE setosa C:/Users/exploFH/Nextcloud/BWI/05_SEM/~
##  6     3     3 an integer         1.4    C:/Users/exploFH/Nextcloud/BWI/05_SEM/~
##  7     3     5 1/0/T/F/TRUE/FALSE setosa C:/Users/exploFH/Nextcloud/BWI/05_SEM/~
##  8     4     3 an integer         1.3    C:/Users/exploFH/Nextcloud/BWI/05_SEM/~
##  9     4     5 1/0/T/F/TRUE/FALSE setosa C:/Users/exploFH/Nextcloud/BWI/05_SEM/~
## 10     5     3 an integer         1.5    C:/Users/exploFH/Nextcloud/BWI/05_SEM/~
## # ... with 280 more rows
```

Alternatively, character columns can be transformed with `mutate()`:

```
x = read_csv("data/text1.csv", col_types = "ccddc")
x
```

```
## # A tibble: 150 x 5
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##    <chr>        <chr>              <dbl>       <dbl> <chr>
##  1 5.1          3.5                  1.4         0.2 setosa
##  2 4.9          3                    1.4         0.2 setosa
##  3 4.7          3.2                  1.3         0.2 setosa
##  4 4.6          3.1                  1.5         0.2 setosa
##  5 5            3.6                  1.4         0.2 setosa
##  6 5.4          3.9                  1.7         0.4 setosa
##  7 4.6          3.4                  1.4         0.3 setosa
##  8 5            3.4                  1.5         0.2 setosa
##  9 4.4          2.9                  1.4         0.2 setosa
## 10 4.9          3.1                  1.5         0.1 setosa
## # ... with 140 more rows
```

```
x %>% mutate(Sepal.Length = parse_double(Sepal.Length),
             Sepal.Width = parse_double(Sepal.Width))
```

```
## # A tibble: 150 x 5
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##           <dbl>       <dbl>        <dbl>       <dbl> <chr>
##  1          5.1         3.5          1.4         0.2 setosa
##  2          4.9         3            1.4         0.2 setosa
##  3          4.7         3.2          1.3         0.2 setosa
##  4          4.6         3.1          1.5         0.2 setosa
##  5          5           3.6          1.4         0.2 setosa
##  6          5.4         3.9          1.7         0.4 setosa
##  7          4.6         3.4          1.4         0.3 setosa
##  8          5           3.4          1.5         0.2 setosa
##  9          4.4         2.9          1.4         0.2 setosa
## 10          4.9         3.1          1.5         0.1 setosa
## # ... with 140 more rows
```

1.2 Fixed-width data

Text columns can be a mess if not properly quoted.

```
read_delim("data/text2.txt", delim = " ")
```

```
## New names:
## * `` -> ...3
## * `` -> ...8
## * `` -> ...9
## * `` -> ...10
## * `` -> ...11
## * ...

## Warning: One or more parsing issues, see `problems()` for details

## Rows: 1 Columns: 18

## -- Column specification ---------------------------------------------------------
## Delimiter: " "
## chr (9): David, Meyer, ...3, Höchstädtplatz, 6,, 1200, Wien, ...8, ...10
## dbl (4): ...9, ...11, ...12, ...13
## lgl (5): ...14, 0699, 12345674, ...17, ...18
```

```
##
## 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.
```

```
## # A tibble: 1 x 18
##   David Meyer ...3  Höchstädtplatz `6,`  `1200` Wien   ...8   ...9 ...10 ...11
##   <chr> <chr> <chr> <chr>          <chr> <chr>  <chr>  <chr> <dbl> <chr> <dbl>
## 1 Hugo  H.    Wolf  An             den   langen Lüssen 47;    1190 Wien     43
## # ... with 7 more variables: ...12 <dbl>, ...13 <dbl>, ...14 <lgl>, 0699 <lgl>,
## #   12345674 <lgl>, ...17 <lgl>, ...18 <lgl>
```

read.fwf() can help here:

```
data <- read_fwf("data/text2.txt",
                 fwf_cols(Name = 13, Address = 35, Tel = 15))
```

```
## Rows: 2 Columns: 3
```

```
## -- Column specification --------------------------------------------------------
##
## chr (3): Name, Address, Tel
```

```
##
## 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.
```

```
data
```

```
## # A tibble: 2 x 3
##   Name         Address                            Tel
##   <chr>        <chr>                              <chr>
## 1 David Meyer  Höchstädtplatz 6, 1200 Wien        0699 12345674
## 2 Hugo H. Wolf An den langen Lüssen 47; 1190 Wien +43 4545 45454
```

Exercise:
Try to read in the data sets `dataXX.txt` provided in the file `data.zip`

```
read_delim("data/data1.txt") # Data is shit -> sep and decimal
```

```
## Rows: 93 Columns: 28
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (14): Manufacturer, Model, Type, AirBags, DriveTrain, Cylinders, EngineS...
## dbl (14): ID, Min.Price, Price, Max.Price, MPG.city, MPG.highway, Rev.per.mi...
```

```
##
## 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.
## Warning: One or more parsing issues, see `problems()` for details

## # A tibble: 93 x 28
##       ID Manufacturer Model      Type  Min.Price Price Max.Price MPG.city MPG.highway
##    <dbl> <chr>        <chr>      <chr>     <dbl> <dbl>     <dbl>    <dbl>       <dbl>
##  1     1 Acura        Integra    Small        12     9        15        9          18
##  2     2 Acura        Legend     Mids~        29     2        33        9          38
##  3     3 Audi         90         Comp~        25     9        29        1          32
##  4     4 Audi         100        Mids~        30     8        37        7          44
##  5     5 BMW          535i       Mids~        23     7        30       36           2
##  6     6 Buick        Century    Mids~        14     2        15        7          17
##  7     7 Buick        LeSabre    Large        19     9        20        8          21
##  8     8 Buick        Roadmaster Large        22     6        23        7          24
##  9     9 Buick        Riviera    Mids~        26     3        26        3          26
## 10    10 Cadillac     DeVille    Large        33    34         7       36           3
## # ... with 83 more rows, and 19 more variables: AirBags <chr>,
## #   DriveTrain <chr>, Cylinders <chr>, EngineSize <chr>, Horsepower <chr>,
## #   RPM <chr>, Rev.per.mile <dbl>, Man.trans.avail <dbl>,
## #   Fuel.tank.capacity <chr>, Passengers <chr>, Length <chr>, Wheelbase <chr>,
## #   Width <dbl>, Turn.circle <dbl>, Rear.seat.room <dbl>, Luggage.room <dbl>,
## #   Weight <dbl>, Origin <dbl>, Make <chr>
```

```
read_delim("data/data2.txt", col_names = FALSE) # No Header
```

```
## Rows: 93 Columns: 27

## -- Column specification --------------------------------------------------------
## Delimiter: ";"
## chr  (9): X1, X2, X3, X9, X10, X11, X16, X26, X27
## dbl (12): X7, X8, X13, X14, X15, X18, X19, X20, X21, X22, X24, X25

##
## 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.

## # A tibble: 93 x 27
##    X1    X2    X3       X4    X5    X6    X7    X8 X9    X10   X11     X12   X13
##    <chr> <chr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <chr> <chr> <chr> <dbl> <dbl>
##  1 Acura Inte~ Small   129   159   188    25    31 None  Front 4        18   140
##  2 Acura Lege~ Mids~   292   339   387    18    25 Driv~ Front 6        32   200
##  3 Audi  90    Comp~   259   291   323    20    26 Driv~ Front 6        28   172
##  4 Audi  100   Mids~   308   377   446    19    26 Driv~ Front 6        28   172
##  5 BMW   535i  Mids~   237    30   362    22    30 Driv~ Rear  4        35   208
##  6 Buick Cent~ Mids~   142   157   173    22    31 Driv~ Front 4        22   110
##  7 Buick LeSa~ Large   199   208   217    19    28 Driv~ Front 6        38   170
##  8 Buick Road~ Large   226   237   249    16    25 Driv~ Rear  6        57   180
##  9 Buick Rivi~ Mids~   263   263   263    19    27 Driv~ Front 6        38   170
## 10 Cadi~ DeVi~ Large    33   347   363    16    25 Driv~ Front 8        49   200
## # ... with 83 more rows, and 14 more variables: X14 <dbl>, X15 <dbl>,
## #   X16 <chr>, X17 <dbl>, X18 <dbl>, X19 <dbl>, X20 <dbl>, X21 <dbl>,
## #   X22 <dbl>, X23 <dbl>, X24 <dbl>, X25 <dbl>, X26 <chr>, X27 <chr>
```

```r
d3 <- read_fwf("data/data3.txt", col_positions = fwf_widths(c(15,15,8,10,6,10,9,12,19,18))) # Read with
```

```
## Rows: 94 Columns: 10
```

```
## -- Column specification --------------------------------------------------------
##
## chr (10): X1, X2, X3, X4, X5, X6, X7, X8, X9, X10
```

```
##
## 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.
```

```r
names(d3) <- d3[1,] # Add headers (from first data line)
```

```
## Warning: The `value` argument of `names<-` must be a character vector as of
## tibble 3.0.0.
```

```r
d3 <- x[-1,] # (remove first data line -> header)
d3
```

```
## # A tibble: 149 x 5
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##    <chr>        <chr>              <dbl>       <dbl> <chr>  
##  1 4.9          3                    1.4         0.2 setosa 
##  2 4.7          3.2                  1.3         0.2 setosa 
##  3 4.6          3.1                  1.5         0.2 setosa 
##  4 5            3.6                  1.4         0.2 setosa 
##  5 5.4          3.9                  1.7         0.4 setosa 
##  6 4.6          3.4                  1.4         0.3 setosa 
##  7 5            3.4                  1.5         0.2 setosa 
##  8 4.4          2.9                  1.4         0.2 setosa 
##  9 4.9          3.1                  1.5         0.1 setosa 
## 10 5.4          3.7                  1.5         0.2 setosa 
## # ... with 139 more rows
```

```r
d4 <- read_table("data/data4.txt")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   .default = col_character(),
##   `"Price"` = col_double(),
##   `"Max.Price"` = col_double(),
##   `"MPG.city"` = col_double(),
##   `"MPG.highway"` = col_double(),
##   `"AirBags"` = col_double(),
##   `"Rev.per.mile"` = col_double(),
##   `"Man.trans.avail"` = col_double(),
##   `"Wheelbase"` = col_double(),
##   `"Width"` = col_double(),
##   `"Turn.circle"` = col_double(),
##   `"Rear.seat.room"` = col_double(),
##   `"Luggage.room"` = col_double()
## )
## i Use `spec()` for the full column specifications.
```

```
## Warning: 93 parsing failures.
## row col   expected     actual             file
##   1  -- 27 columns 29 columns 'data/data4.txt'
##   2  -- 27 columns 31 columns 'data/data4.txt'
##   3  -- 27 columns 30 columns 'data/data4.txt'
##   4  -- 27 columns 31 columns 'data/data4.txt'
##   5  -- 27 columns 30 columns 'data/data4.txt'
## ... ... .......... .......... ................
## See problems(...) for more details.
```

```
d4
```

```
## # A tibble: 93 x 27
##    `"Manufacturer"` `"Model"`      `"Type"` `"Min.Price"` `"Price"` `"Max.Price"`
##    <chr>            <chr>          <chr>    <chr>             <dbl>         <dbl>
##  1 "\"1\""          "\"Acura\""    "\"Inte~ "\"Small\""        12.9          15.9
##  2 "\"2\""          "\"Acura\""    "\"Lege~ "\"Midsize\""      29.2          33.9
##  3 "\"3\""          "\"Audi\""     "\"90\"" "\"Compact\""      25.9          29.1
##  4 "\"4\""          "\"Audi\""     "\"100\~ "\"Midsize\""      30.8          37.7
##  5 "\"5\""          "\"BMW\""      "\"535i~ "\"Midsize\""      23.7          30  
##  6 "\"6\""          "\"Buick\""    "\"Cent~ "\"Midsize\""      14.2          15.7
##  7 "\"7\""          "\"Buick\""    "\"LeSa~ "\"Large\""        19.9          20.8
##  8 "\"8\""          "\"Buick\""    "\"Road~ "\"Large\""        22.6          23.7
##  9 "\"9\""          "\"Buick\""    "\"Rivi~ "\"Midsize\""      26.3          26.3
## 10 "\"10\""         "\"Cadillac\"" "\"DeVi~ "\"Large\""        33            34.7
## # ... with 83 more rows, and 21 more variables: "MPG.city" <dbl>,
## #   "MPG.highway" <dbl>, "AirBags" <dbl>, "DriveTrain" <chr>,
## #   "Cylinders" <chr>, "EngineSize" <chr>, "Horsepower" <chr>, "RPM" <chr>,
## #   "Rev.per.mile" <dbl>, "Man.trans.avail" <dbl>, "Fuel.tank.capacity" <chr>,
## #   "Passengers" <chr>, "Length" <chr>, "Wheelbase" <dbl>, "Width" <dbl>,
## #   "Turn.circle" <dbl>, "Rear.seat.room" <dbl>, "Luggage.room" <dbl>,
## #   "Weight" <chr>, "Origin" <chr>, "Make" <chr>
```

2 Separating/joining columns

Separate:

```
data2 <- data %>%
  mutate(Address = str_replace(Address, ",", ";")) %>%
  separate(Address, c("Street", "ZIPCity"), sep = "; ") %>%
  separate(ZIPCity, c("ZIP", "City"), sep = " ")
data2
```

```
## # A tibble: 2 x 5
##   Name         Street                   ZIP   City  Tel
##   <chr>        <chr>                   <chr> <chr> <chr>
## 1 David Meyer  Höchstädtplatz 6        1200  Wien  0699 12345674
## 2 Hugo H. Wolf An den langen Lüssen 47 1190  Wien  +43 4545 45454
```

… and join again:

```
data2 %>%
  unite(ZIPCity, ZIP, City, sep = " ") %>%
  unite(Address, ZIPCity, Street, sep = ", ")
```

```
## # A tibble: 2 x 3
##   Name         Address                            Tel
##   <chr>        <chr>                              <chr>
## 1 David Meyer  1200 Wien, Höchstädtplatz 6        0699 12345674
## 2 Hugo H. Wolf 1190 Wien, An den langen Lüssen 47 +43 4545 45454
```

Exercise:
Using the data below, transform the birth date into the format `YYYY-MM-DD`. Try to pad days and months with a leading 0, so that, e.g., `1.1.1988` becomes `1988-01-01`. (Hint: use `mutate()` with `str_pad()`).

```
tribble(~Name, ~Birthdate,
        "Susan", "29.10.1966",
        "Will", "1.1.1988",
        "Chris", "10.10.1977")
```

```
## # A tibble: 3 x 2
##   Name  Birthdate
##   <chr> <chr>
## 1 Susan 29.10.1966
## 2 Will  1.1.1988
## 3 Chris 10.10.1977
```

3 Wide and long format

Sometimes, values of one variable are "pivoted" into columns:

```
head(USArrests)
```

```
##            Murder Assault UrbanPop Rape
## Alabama      13.2     236       58 21.2
## Alaska       10.0     263       48 44.5
## Arizona       8.1     294       80 31.0
## Arkansas      8.8     190       50 19.5
## California    9.0     276       91 40.6
## Colorado      7.9     204       78 38.7
```

Use `gather()` to transform the data into "long" format:

```
arrests_long <- USArrests %>%
  rownames_to_column("State") %>%  ## to keep info -- gather() will remove rownames
  gather(key = "Crime", value = "Arrests",
         Murder, Assault, Rape)
head(arrests_long)
```

```
##        State UrbanPop  Crime Arrests
## 1    Alabama       58 Murder    13.2
## 2     Alaska       48 Murder    10.0
## 3    Arizona       80 Murder     8.1
## 4   Arkansas       50 Murder     8.8
## 5 California       91 Murder     9.0
## 6   Colorado       78 Murder     7.9
```

… and spread() for transforming "long" into "wide" format:

```
arrests_long %>% spread(Crime, Arrests) %>% head()
```

```
##        State UrbanPop Assault Murder Rape
## 1    Alabama       58     236   13.2 21.2
## 2     Alaska       48     263   10.0 44.5
## 3    Arizona       80     294    8.1 31.0
## 4   Arkansas       50     190    8.8 19.5
## 5 California       91     276    9.0 40.6
## 6   Colorado       78     204    7.9 38.7
```

Exercise:
The sleep data in R is about extra sleep time of 10 students caused by two drugs (group). Transform the data into wide format, so that the timings for the two drugs are represented in two separate columns. Compute, for each student, the difference in extra sleep time and add this to the data.

4 Missing data

```
data = read_table("data/text3.txt", col_names = TRUE, na = "??")
```

```
## Warning: Missing column names filled in: 'X6' [6]
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   Class = col_character(),
##   Sex = col_character(),
##   Age = col_character(),
##   Died = col_double(),
##   Survived = col_double(),
##   X6 = col_character()
## )
## Warning: 16 parsing failures.
## row col  expected    actual             file
##   1  -- 6 columns 5 columns 'data/text3.txt'
##   2  -- 6 columns 3 columns 'data/text3.txt'
##   3  -- 6 columns 5 columns 'data/text3.txt'
##   4  -- 6 columns 3 columns 'data/text3.txt'
##   5  -- 6 columns 5 columns 'data/text3.txt'
## ... ... ......... ......... ................
## See problems(...) for more details.
```

```
data <- data %>%
  mutate_all(na_if, "") %>%
  fill(Class, Sex, .direction = "down")
```

Regular NA handling:

```
data %>% filter(!complete.cases(.)) # find all rows with missings
```

```
## # A tibble: 16 x 6
##    Class  Sex   Age    Died Survived X6
##    <chr>  <chr> <chr> <dbl>    <dbl> <chr>
##  1 1st    Male  Child     0        5 <NA>
##  2 Adult  118   57       NA       NA <NA>
##  3 Female Child 0         1       NA <NA>
##  4 Adult  4     140      NA       NA <NA>
##  5 2nd    Male  Child     0       11 <NA>
##  6 Adult  154   14       NA       NA <NA>
##  7 Female Child 0        13       NA <NA>
##  8 Adult  13    80       NA       NA <NA>
##  9 3rd    Male  Child    35       13 <NA>
## 10 Adult  387   75       NA       NA <NA>
## 11 Female Child 17       14       NA <NA>
## 12 Adult  89    76       NA       NA <NA>
## 13 Crew   Male  Child     0       NA <NA>
## 14 Adult  670   192      NA       NA <NA>
## 15 Female Child 0        NA       NA <NA>
## 16 Adult  3     20       NA       NA <NA>
```

```
data %>% drop_na() ## either drop them ...
```

```
## # A tibble: 0 x 6
## # ... with 6 variables: Class <chr>, Sex <chr>, Age <chr>, Died <dbl>,
## #   Survived <dbl>, X6 <chr>
```

```
data %>% mutate(Survived = replace_na(Survived, 0)) ## ... or replace them
```

```
## # A tibble: 16 x 6
##    Class  Sex   Age    Died Survived X6
##    <chr>  <chr> <chr> <dbl>    <dbl> <chr>
##  1 1st    Male  Child     0        5 <NA>
##  2 Adult  118   57       NA        0 <NA>
##  3 Female Child 0         1        0 <NA>
##  4 Adult  4     140       NA        0 <NA>
##  5 2nd    Male  Child     0        11 <NA>
##  6 Adult  154   14        NA        0 <NA>
##  7 Female Child 0        13        0 <NA>
##  8 Adult  13    80        NA        0 <NA>
##  9 3rd    Male  Child    35        13 <NA>
## 10 Adult  387   75        NA        0 <NA>
## 11 Female Child 17       14        0 <NA>
## 12 Adult  89    76        NA        0 <NA>
## 13 Crew   Male  Child     0        0 <NA>
## 14 Adult  670   192       NA        0 <NA>
## 15 Female Child 0        NA        0 <NA>
## 16 Adult  3     20        NA        0 <NA>
```

Exercise:
Using the data below, first find out all rows with missing data. Impute missing invitations with 0, and missing ages with the average age. Remove all rows with other missings.

```
tribble(~Name, ~Age, ~Invitations, ~Phone,
        "Tim", 20, 0, "123 345",
        "Mary", 30, 12, "321 999",
        "Chris", 25, NA, "444 324",
        "Lilly", NA, 0, "453 424",
        "Will", 20, 0, NA
)
```

```
## # A tibble: 5 x 4
##   Name    Age Invitations Phone
##   <chr> <dbl>       <dbl> <chr>
## 1 Tim      20           0 123 345
## 2 Mary     30          12 321 999
## 3 Chris    25          NA 444 324
## 4 Lilly    NA           0 453 424
## 5 Will     20           0 <NA>
```