R1 Exercises: Basic Data Wrangling

Topór Karol

2023-11-19

Table of contents

Packages used in this notebook:

```
suppressPackageStartupMessages({
  library(tidyverse)
})
```

---

1 Create tibble friends

Create a tibble `friends` using the commands `as_tibble()`, `tibble()` and `tribble()`, respectively, with the following variables: `name` (Susan, Walter, Tim, Ann), `height` in cm (180, 185, 190, 172) and `weight` in kg (70, 85, 100, 75). Additionally add a variable `sex` with entries (Male and Female) that corresponds to the sex of the `name` entry.

1.1 Create `friends` using `as_tibble()` (1P)

```
df_friends <- data.frame(
  name = c("Susan", "Walter", "Tim", "Ann"),
  height = c(180, 185, 190, 172),
  weight = c(70, 85, 100, 75),
  sex = c("Female", "Male", "Male", "Female")
)

tb_friends <- as_tibble(df_friends)
tb_friends
```

```
# A tibble: 4 x 4
  name   height weight sex
  <chr>   <dbl>  <dbl> <chr>
1 Susan     180     70 Female
2 Walter    185     85 Male
3 Tim       190    100 Male
4 Ann       172     75 Female
```

1.2 Create `friends` using `tibble()` (1P)

```
tb_friends <- tibble(
  name = c("Susan", "Walter", "Tim", "Ann"),
  height = c(180, 185, 190, 172),
  weight = c(70, 85, 100, 75),
  sex = c("Female", "Male", "Male", "Female")
)
tb_friends
```

```
# A tibble: 4 x 4
  name   height weight sex
  <chr>   <dbl>  <dbl> <chr>
1 Susan     180     70 Female
2 Walter    185     85 Male
3 Tim       190    100 Male
4 Ann       172     75 Female
```

1.3 Create `friends` using `tribble()` (1P)

```
tb_friends <- friends_tribble <- tribble(
  ~name,   ~height, ~weight, ~sex,
  "Susan", 180,     70,      "Female",
  "Walter", 185,    85,      "Male",
  "Tim",   190,     100,     "Male",
  "Ann",   172,     75,      "Female"
)
tb_friends
```

```
# A tibble: 4 x 4
  name   height weight sex
  <chr>   <dbl>  <dbl> <chr>
1 Susan     180     70 Female
2 Walter    185     85 Male
3 Tim       190    100 Male
4 Ann       172     75 Female
```

1.4 Tidy data format (1P)

Is the data in the tibble `friends` in the format of a tidy data set? Explain your reasoning.

- Every column is a variable
- Every row is an observation

2 Basic data manipulation

2.1 Transform variable `sex` into a factor (1P)

Change the variable `sex` in `friends` into a factor (use command `as.factor()`).

```
tb_friends <- tb_friends %>% mutate(sex = as.factor(sex))
tb_friends
```

```
# A tibble: 4 x 4
  name   height weight sex
  <chr>   <dbl>  <dbl> <fct>
1 Susan     180     70 Female
2 Walter    185     85 Male
3 Tim       190    100 Male
4 Ann       172     75 Female
```

2.2 Sorting `friends` (1P)

Sort the data in `friends` such that Male entries come before Female entries, subsequently the names in ascending order, the height in ascending and finally the weight in descending order. Explain the ordering of the output.

```
tb_friends %>% arrange(desc(sex), name, height, desc(weight))
```

```
# A tibble: 4 x 4
  name   height weight sex
  <chr>   <dbl>  <dbl> <fct>
1 Tim       190    100 Male
2 Walter    185     85 Male
3 Ann       172     75 Female
4 Susan     180     70 Female
```

First the dataframe is sorted in descending order by sex and since 'Male' comes after 'Female' alphabetically 'Male' comes first. In every sex category the names are sorted alphabetically. In every names category the heights are sorted from biggest to smallest and in every height category the weights are sorted from smalles to biggest.

However because there are not more entires which have the same sex and name the height is not sorted or better it appears as not sorted. The same goes for weight. There are no two rows with the same sex, name and height which could be sorted. This is due to the logic of the arrange function

2.3 Add variable `bmi` (1P)

Add an additional variable `bmi` (body mass index) **after the variable `name`** to the friends data. The `bmi` entry is the weight of a person in kg divided through the squared height in meter of that person.

```
tb_friends <-  mutate(tb_friends, bmi = weight/((height/100)^2), .after = "name")
tb_friends
```

```
# A tibble: 4 x 5
  name     bmi height weight sex
  <chr>  <dbl>  <dbl>  <dbl> <fct>
1 Susan   21.6    180     70 Female
2 Walter  24.8    185     85 Male
3 Tim     27.7    190    100 Male
4 Ann     25.4    172     75 Female
```

2.4 Add variable `overweight` (1P)

Add to `friends` before column 3 a variable `overweight` that is a factor with entry `yes` for persons with a `bmi` larger than 25 and `no` otherwise.

```
tb_friends <-  mutate(tb_friends, overweight = ifelse(bmi > 25, "yes", "No"), .before = 3)
tb_friends
```

```
# A tibble: 4 x 6
  name     bmi overweight height weight sex
  <chr>  <dbl> <chr>       <dbl>  <dbl> <fct>
1 Susan   21.6 No            180     70 Female
2 Walter  24.8 No            185     85 Male
```

```
3 Tim     27.7 yes           190    100 Male
4 Ann     25.4 yes           172     75 Female
```

2.5 Summarize friends (1P)

Summarize the `friends` data by showing the mean of the heights and weight.

```
summarise(tb_friends,
          mean_height = mean(height),
          mean_weight = mean(weight)
          )
```

```
# A tibble: 1 x 2
  mean_height mean_weight
        <dbl>       <dbl>
1        182.        82.5
```

2.6 Summarize friends separated by `sex` (1P)

Summarize the `friends` data by showing the mean of the heights and weight separated by `sex`.

```
summarise(group_by(tb_friends, sex),
          mean_height = mean(height),
          mean_weight = mean(weight)
          )
```

```
# A tibble: 2 x 3
  sex    mean_height mean_weight
  <fct>        <dbl>       <dbl>
1 Female         176         72.5
2 Male           188.        92.5
```

2.7 Summarize `bmi` by `overweight` (1P)

Summarize the `bmi` in `friends` by showing the mean, min and max of `bmi` separated by overweight.

```
summarise(group_by(tb_friends, overweight),
          bmi_mean = mean(bmi),
          bmi_max = max(bmi),
          bmi_min = min(bmi)
          )
```

```
# A tibble: 2 x 4
  overweight bmi_mean bmi_max bmi_min
  <chr>         <dbl>   <dbl>   <dbl>
1 No             23.2    24.8    21.6
2 yes            26.5    27.7    25.4
```

2.8 Summarize `bmi` separated by `sex` and `overweight` (1P)

Summarize the `bmi` in `friends` by showing the mean, min and max of `bmi` separated by `sex` and using the `%>%` operator.

```
group_by(tb_friends, sex, overweight) %>% summarise(
  bmi_mean = mean(bmi),
  bmi_max = max(bmi),
  bmi_min = min(bmi)
  )
```

```
# A tibble: 4 x 5
# Groups:   sex [2]
  sex    overweight bmi_mean bmi_max bmi_min
  <fct>  <chr>         <dbl>   <dbl>   <dbl>
1 Female No             21.6    21.6    21.6
2 Female yes            25.4    25.4    25.4
3 Male   No             24.8    24.8    24.8
4 Male   yes            27.7    27.7    27.7
```

2.9 Add `mean` of `bmi` to the `friends` data (1P)

Add the mean of the `bmi` of all friends permanently to the `friends` data right after the `bmi` variable.

```
tb_friends <- mutate(tb_friends, bmi_mean = mean(bmi), .after = "bmi")
tb_friends
```

```
# A tibble: 4 x 7
  name     bmi bmi_mean overweight height weight sex
  <chr>  <dbl>    <dbl> <chr>       <dbl>  <dbl> <fct>
1 Susan   21.6     24.9 No            180     70 Female
2 Walter  24.8     24.9 No            185     85 Male
3 Tim     27.7     24.9 yes           190    100 Male
4 Ann     25.4     24.9 yes           172     75 Female
```

2.10 Filter on two rows (1P)

Filter all friends with a height between 172 and 180 cm OR with a weight exceeding 90 kg.

```
filter(tb_friends, between(height, 172, 180) | weight > 90)
```

```
# A tibble: 3 x 7
  name    bmi bmi_mean overweight height weight sex
  <chr> <dbl>    <dbl> <chr>       <dbl>  <dbl> <fct>
1 Susan  21.6     24.9 No            180     70 Female
2 Tim    27.7     24.9 yes           190    100 Male
3 Ann    25.4     24.9 yes           172     75 Female
```

2.11 Filter data on bmi (1P)

Show only those entries in `friends` that have a `bmi` larger than the average `bmi` of all entries in `friends`.

```
filter(tb_friends, bmi > mean(bmi))
```

```
# A tibble: 2 x 7
  name    bmi bmi_mean overweight height weight sex
  <chr> <dbl>    <dbl> <chr>       <dbl>  <dbl> <fct>
1 Tim    27.7     24.9 yes           190    100 Male
2 Ann    25.4     24.9 yes           172     75 Female
```

2.12 Select data on `bmi` (1P)

Show only the names of the persons in `friends` that have a `bmi` larger than the average of the `bmi`

```r
filter(tb_friends, bmi > mean(bmi)) %>% select(name)
```

```
# A tibble: 2 x 1
  name
  <chr>
1 Tim
2 Ann
```

2.13 Select data on `bmi` and show it as a vector (1P)

Show the names of the persons in `friends` that have a `bmi` larger than the average of the `bmi` as a vector (Hint: a tibble is still a data frame and a data frame is a list, so you can extract the names form a tibble the way you would extract it from a list).

```r
filter(tb_friends, bmi > mean(bmi)) %>% pull(name)
```

```
[1] "Tim" "Ann"
```

3 The `state.x77` data

3.1 Transform the data (4P)

In the `state.x77` data, create a new variable `Risk` with the values `high` (`Murder` $>$ 10), `low` (`Murder` $<$ 4) and `average`.

- Show how to create this new variable `Risk` with `ifelse()` and with `case_when()`
- Transform `Area` into square kilometers. Replace the old variable. One square miles is equals to 2.58998811 square kilometers.
- Remove the variable `Frost`

3.1.1 Risk via `ifelse()`

```r
# Risk = ifelse()
tb_state.x77 <- state.x77 %>%
  as_tibble() %>%
  mutate(Risk = ifelse(
                        Murder > 10, "high",
                        ifelse(Murder < 4, "low", "avarage")
```

```
                        )
        )

  tb_state.x77
```

```
# A tibble: 50 x 9
   Population Income Illiteracy `Life Exp` Murder `HS Grad` Frost   Area Risk
        <dbl>  <dbl>      <dbl>      <dbl>  <dbl>     <dbl> <dbl>  <dbl> <chr>
 1       3615   3624        2.1       69.0   15.1      41.3    20  50708 high
 2        365   6315        1.5       69.3   11.3      66.7   152 566432 high
 3       2212   4530        1.8       70.6    7.8      58.1    15 113417 avarage
 4       2110   3378        1.9       70.7   10.1      39.9    65  51945 high
 5      21198   5114        1.1       71.7   10.3      62.6    20 156361 high
 6       2541   4884        0.7       72.1    6.8      63.9   166 103766 avarage
 7       3100   5348        1.1       72.5    3.1      56     139   4862 low
 8        579   4809        0.9       70.1    6.2      54.6   103   1982 avarage
 9       8277   4815        1.3       70.7   10.7      52.6    11  54090 high
10       4931   4091        2         68.5   13.9      40.6    60  58073 high
# i 40 more rows
```

3.1.2 Risk via `case_when()`

```
  # Risk = case_when()
  tb_state.x77 <- state.x77 %>%
    as_tibble() %>%
    mutate(
      Risk = case_when(
        Murder > 10  ~ "high",
        Murder < 4   ~ "low",
        TRUE         ~ "avarage"
      )
    )
  tb_state.x77
```

```
# A tibble: 50 x 9
   Population Income Illiteracy `Life Exp` Murder `HS Grad` Frost   Area Risk
        <dbl>  <dbl>      <dbl>      <dbl>  <dbl>     <dbl> <dbl>  <dbl> <chr>
 1       3615   3624        2.1       69.0   15.1      41.3    20  50708 high
 2        365   6315        1.5       69.3   11.3      66.7   152 566432 high
 3       2212   4530        1.8       70.6    7.8      58.1    15 113417 avarage
```

```
 4       2110   3378        1.9       70.7   10.1      39.9    65  51945 high
 5      21198   5114        1.1       71.7   10.3      62.6    20 156361 high
 6       2541   4884        0.7       72.1    6.8      63.9   166 103766 avarage
 7       3100   5348        1.1       72.5    3.1      56     139   4862 low
 8        579   4809        0.9       70.1    6.2      54.6   103   1982 avarage
 9       8277   4815        1.3       70.7   10.7      52.6    11  54090 high
10       4931   4091        2         68.5   13.9      40.6    60  58073 high
# i 40 more rows
```

3.2 Transforming and Summarizing (4P)

Use the `state.x77` data with the added `Risk` variable from above. For each risk group, compute mean, median, minimum, maximum income and count. Filter out the group with highest average income.

```
summary_stats <- tb_state.x77 %>%
    group_by(Risk) %>%
    summarise(
          mean = round(mean(Income), 0),
          median = round(median(Income)),
          min = min(Income),
          max = max(Income),
          N = n()
          )
summary_stats
```

```
# A tibble: 3 x 6
  Risk     mean median   min   max     N
  <chr>   <dbl>  <dbl> <dbl> <dbl> <int>
1 avarage  4477   4546  3601  5299    22
2 high     4301   4091  3098  6315    17
3 low      4561   4558  3694  5348    11
```

```
summary_stats %>% filter(mean == max(mean))
```

```
# A tibble: 1 x 6
  Risk   mean median   min   max     N
  <chr> <dbl>  <dbl> <dbl> <dbl> <int>
1 low    4561   4558  3694  5348    11
```