Challenge_3: Joining Relational Data, Writing Your Own Functions, and String Operations

AUTHOR
Muskan Dhar

PUBLISHED
June 21, 2024

Make sure you change the author's name in the above YAML header. Name the html file as Lastname_Firstname_Challenge3.html

Setup

If you have not installed the following packages, please install them before loading them.

```
library(tidyverse)
library(readxl)
library(haven) #for loading other datafiles (SAS, STATA, SPSS, etc.)
library(stringr) # if you have not installed this package, please install it.
library(lubridate)
library(stringr)
```

Challenge Overview

In this challenge, we will practice `join()` with relational data, use string functions to process, extract information, and mutate and clean data. We will also practice wring own functions.

There will be coding components and writing components. Please read the instructions for each part and complete your challenges.

Datasets

There are four datasets provided in this challenge. Please download the following dataset files from Google Classroom and save them to a folder within your project working directory (i.e.: "DACSS601_data"). If you don't have a folder to store the datasets, please create one.

- Part 1 and 2: ESS_5.dta and p5v2018.sav (used in Challenge#1) ⭐⭐
- Part 3: babynames.csv (used in Challenge#1) ⭐

Find the `_data` folder, then use the correct R command to read the datasets.

Part 1. Joining Individual-level and Country-Level Data

We have been working with ESS and Polity datasets in the previous two challenges and should be familiar with them. Suppose we have a research project that studies European citizens' social behaviors and public opinions, and we are interested in **how the countries that respondents live**

in influence their behavior and opinion. In this case, we will need to combine the two data for future analysis.

1. **Read the two raw datasets.**

   For ESS_5: (1) keep only the following columns: *idno, essround, male, age, edu, eth_major, income_10, cntry, vote.* **(2)** recode *essround* to 2010, and rename it as *year*.

   For Polity V, keep the first 10 columns.

```
#Type your code here

data1 <- read_dta("ESS_5.dta")
data1 <- data1[, c("idno", "essround", "male", "age", "edu", "eth_major", "income_10
data1 <- data1 %>%
  mutate(year = ifelse(essround == 5, 2010, essround)) %>%
  select(idno, year, everything())
data1 <- data1 %>%
  select(-essround)
head(data1)
```

```
# A tibble: 6 × 9
   idno  year  male   age   edu eth_major income_10 cntry vote
  <dbl> <dbl> <dbl> <dbl> <dbl>     <dbl>     <dbl> <chr> <dbl+lbl>
1 15906  2010     0    14     1         1         2 GR    3 [Not eligible to vo…
2 21168  2010     0    14     1         1         2 IE    3 [Not eligible to vo…
3    40  2010     0    14     1        NA         8 LT    3 [Not eligible to vo…
4  2108  2010     0    14     1         1        NA RU    3 [Not eligible to vo…
5   519  2010     0    14     1         1        NA IL    2 [No]
6  2304  2010     0    14     1         1        NA ES    3 [Not eligible to vo…
```

```
data2 <- read_sav("p5v2018.sav")
data2 <- data2[, 1:10]
head(data2)
```

```
# A tibble: 6 × 10
     p5   cyear ccode scode country      year  flag fragment democ autoc
  <dbl>   <dbl> <dbl> <chr> <chr>       <dbl> <dbl>    <dbl> <dbl> <dbl>
1     0 7001800   700 AFG   Afghanistan  1800     0       NA     1     7
2     0 7001801   700 AFG   Afghanistan  1801     0       NA     1     7
3     0 7001802   700 AFG   Afghanistan  1802     0       NA     1     7
4     0 7001803   700 AFG   Afghanistan  1803     0       NA     1     7
5     0 7001804   700 AFG   Afghanistan  1804     0       NA     1     7
6     0 7001805   700 AFG   Afghanistan  1805     0       NA     1     7
```

2. **Answer the following questions:**

   (1) In this project, what is our unit of analysis? Which is the primary data, and which is the foreign data In ESS_5.dta the unit of analysis is idno and in p5v2018.sav the unit of analysis is country. The primary data is ESS_5.dta and the foreign data is p5v2018.sav

   (2) What is(are) the key(s) for the two data The keys are idno and country.

3. **Suppose we have a theory that a country's level of democracy (*democ* in Polity V) affects an individual's electoral participation (*vote* in ESS 5). We must first conduct some necessary data transformation before merging the two data.**

   (1) Countries in ESS_5 are coded with their 2-digit codes (ISO-3166-1) in the *cntry* column. It is difficult to identify from these two-letter abbreviations. Let's first transform the *cntry* column by changing it from the abbreviations to the full country names and renaming the column as *country.*

   Please refer to this website for the list of countries with their 2-letter abbreviations. Read the country list (csv) file, into RStudio, and merge it with the ESS_5 data. By doing so, you add a new "country" column to the existing ESS_5 data.

```
data3 <- read.csv("country_abb.csv")
head(data3)
```

```
            Name Code
1    Afghanistan   AF
2        Albania   AL
3        Algeria   DZ
4 American Samoa   AS
5        Andorra   AD
6         Angola   AO
```

```
data3 <- data3|>
  rename(cntry = Code)|>
  rename(country = Name)
data1<-data1 |> left_join(data3, by = "cntry")
head(data1)
```

```
# A tibble: 6 × 10
   idno  year  male   age   edu eth_major income_10 cntry vote           country
  <dbl> <dbl> <dbl> <dbl> <dbl>     <dbl>     <dbl> <chr> <dbl+lbl>      <chr>
1 15906  2010     0    14     1         1         2 GR    3 [Not eligib… Greece
2 21168  2010     0    14     1         1         2 IE    3 [Not eligib… Ireland
3    40  2010     0    14     1        NA         8 LT    3 [Not eligib… Lithua…
4  2108  2010     0    14     1         1        NA RU    3 [Not eligib… Russia…
5   519  2010     0    14     1         1        NA IL    2 [No]         Israel
6  2304  2010     0    14     1         1        NA ES    3 [Not eligib… Spain
```

   (2) What column(s) will we use as a matching key(s) for combining the updated ESS_5 dataset and Polity V dataset? Note: you can use multiple matching strategies, but I suggest we create a common matching key for both data if there are none.

```
data2<-filter(data2, year == 2010)
head(data2)
```

```
# A tibble: 6 × 10
     p5   cyear ccode scode country      year  flag fragment democ autoc
  <dbl>   <dbl> <dbl> <chr> <chr>       <dbl> <dbl>    <dbl> <dbl> <dbl>
```

```
1     0 7002010   700 AFG   Afghanistan  2010     0        2   -66   -66
2     1 3392010   339 ALB   Albania      2010     0        0     9     0
3     1 6152010   615 ALG   Algeria      2010     0        0     3     1
4     1 5402010   540 ANG   Angola       2010     0        0     2     4
5     1 1602010   160 ARG   Argentina    2010     0        0     8     0
6     1 3712010   371 ARM   Armenia      2010     0        0     5     0
```

\(3\) Join the two data (updated ESS_5 and Polity V). Please print the first few entries as a sanity check. Name the joined data as "ESS_Polity"

::: {.cell}

```{.r .cell-code}
ESS_Polity <- left_join(data1, data2, by = c("country"))
head(ESS_Polity)
```

::: {.cell-output .cell-output-stdout}
```
# A tibble: 6 × 19
   idno year.x  male   age   edu eth_major income_10 cntry vote    country    p5
  <dbl>  <dbl> <dbl> <dbl> <dbl>     <dbl>     <dbl> <chr> <dbl+l> <chr>   <dbl>
1 15906   2010     0    14     1         1         2 GR    3 [Not… Greece      0
2 21168   2010     0    14     1         1         2 IE    3 [Not… Ireland     0
3    40   2010     0    14     1        NA         8 LT    3 [Not… Lithua…     0
4  2108   2010     0    14     1         1        NA RU    3 [Not… Russia…    NA
5   519   2010     0    14     1         1        NA IL    2 [No]  Israel      0
6  2304   2010     0    14     1         1        NA ES    3 [Not… Spain       0
# ℹ 8 more variables: cyear <dbl>, ccode <dbl>, scode <chr>, year.y <dbl>,
#   flag <dbl>, fragment <dbl>, democ <dbl>, autoc <dbl>
```
:::
:::

\(4\) Save the joined data *ESS_Polity* to your local directory using the following code. We will be using this joined data to explore visualization in future challenges. (This is for future usage. No need to submit the saved joined data.)

::: {.cell}

```{.r .cell-code}
write_csv(ESS_Polity, "ESS_Polity.csv")
```
:::

4. **Describe the data structure of the newly joined data *ESS_Polity*. What is its dimension (# of rows and # of columns)? What is its unit of observation? Compared to the original ESS_5 data, does the above data combination change the dimension and unit of observation?**

```
dim(ESS_Polity)
```

```
[1] 52458    19
```

The newly joined data “ESS_Polity” is a tibble. The dimension of ESS_Polity is 52458 rows and 19 columns. The unit of observation is a row where every row represents the details of a country for the year 2010 and the details of the different people who participated in a survey with responses measuring democracy or autocracy. Yes compared to the original ESS_5 data the dimension of the new ESS_Polity data changes (number of rows remain the same but the number of column increase due to the join operation) but the unit of observation does not change.

Part 2. Writing Your Own Functions

Please use the joined data ***ESS_Polity*** in Part 1 and write **ONE** function to complete all the following tasks:

(1) Estimate the range, average, standard deviation, number of NAs, and the number of unique values of any given numeric-type (double or integer) columns.

(2) Test your function with any four columns of your choice.

```
math <- function(data, coln) {
  col <- data[[coln]]
  col_range <- range(col,na.rm = TRUE)
  col_mean <- mean(col, na.rm = TRUE)
  col_sd <- sd(col, na.rm = TRUE)
  col_na <- sum(is.na(col))
  col_unique <- length(unique(na.omit(col)))
    print(paste("Range: ", paste(col_range, collapse = " to "),", Average: ", col_me
}
print("column: income_10")
```

```
[1] "column: income_10"
```

```
math(ESS_Polity, "income_10")
```

```
[1] "Range:  1 to 10 , Average:  5.04862191877092 , Standard Deviation:
2.78753199032184 , No. of NAs:  12620 , No. of unique values:  10"
```

```
print("age")
```

```
[1] "age"
```

```
math(ESS_Polity, "age")
```

```
[1] "Range:  14 to 101 , Average:  47.9152921389154 , Standard Deviation:
18.7957345258898 , No. of NAs:  137 , No. of unique values:  87"
```

```
print("edu")
```

```
[1] "edu"
```

```
math(ESS_Polity, "edu")
```

```
[1] "Range:  1 to 4 , Average:  2.76753077923072 , Standard Deviation:
0.918133378351045 , No. of NAs:  150 , No. of unique values:  4"
```

```
print("eth_major")
```

```
[1] "eth_major"
```

```
math(ESS_Polity, "eth_major")
```

```
[1] "Range:  0 to 1 , Average:  0.936986783451944 , Standard Deviation:
0.24298910563668 , No. of NAs:  1310 , No. of unique values:  2"
```

Part 3. Practicing Stringr Package with Babynames

1. **Import the babynames data:**

```
#Type your code here
data_bn <- read.csv("babynames.csv")
head(data_bn)
```

```
       Name    Sex Occurrences Year
1      Mary Female        7065 1880
2      Anna Female        2604 1880
3      Emma Female        2003 1880
4 Elizabeth Female        1939 1880
5    Minnie Female        1746 1880
6  Margaret Female        1578 1880
```

2. **Use different string functions from stringr package to answer the following questions:**

   (1) Find and list the longest names using count() and a string function.

```
names_long <- data_bn %>%
  count(Name) %>%
  mutate(length = str_length(Name)) %>%
  filter(length == max(length)) %>%
  select(Name)

print(names_long)
```

```
             Name
1  Ashleyelizabeth
```

```
2  Christianalexan
3  Christiananthon
4  Christiandaniel
5  Christianjoseph
6  Christianjoshua
7  Christianmichae
8  Christopheranth
9  Christopherdavi
10 Christopherjame
11 Christopherjohn
12 Christopherjose
13 Christophermich
14 Christopherpaul
15 Christopherryan
16 Davidchristophe
17 Franciscojavier
18 Gabrielalexande
19 Hannahelizabeth
20 Jaydenalexander
21 Johnchristopher
22 Jonathanmichael
23 Jordanalexander
24 Jordanchristoph
25 Joshuaalexander
26 Kevinchristophe
27 Laurenelizabeth
28 Mariadelosangel
29 Mariadelrosario
30 Markchristopher
31 Matthewalexande
32 Michaelchristop
33 Muhammadibrahim
34 Muhammadmustafa
35 Ryanchristopher
36 Seanchristopher
37 Sophiaelizabeth
```

\(2\) Use a string function to detect if the following names are present in the data:

"Ronaldo", "Messi", "Wayne", "Clarck", "Rick", and "Morty".

::: {.cell}

```{.r .cell-code}
#Use the Anchoring (a way of regular expression), "^name$", to specify the name that you search for. By using Anchoring ("^"and"$"), you can search the exact name by specifying the beginning and the ending letters.
#Use the Anchoring (a way of regular expression), "^name$", to specify the name that you search for. By using Anchoring ("^"and"$"), you can search the exact name by specifying the beginning and the ending letters.

names <- c("Ronaldo", "Messi", "Wayne", "Clarck", "Rick", "Morty")
```

```
for (name in names) {
  if (any(str_detect(data_bn$Name, paste0("^", name, "$")))) {
cat(name, ": Present\n")
  } else {
cat(name, ": Not Present\n")
  }
}
```

::: {.cell-output .cell-output-stdout}
```
Ronaldo : Present
Messi : Present
Wayne : Present
Clarck : Not Present
Rick : Present
Morty : Present
```
:::
:::

\(3\) Create a column *LastName* with just one value, "LastName". Next, create another column *FullName,* by combing the strings of columns *name* and *LastName*, separating by a period. For example, a value in this new column should be like "Jacky.LastName".

```
data_bn$LastName <- "LastName"
data_bn$FullName <- str_c(data_bn$Name, data_bn$LastName, sep = ".")
head(data_bn$FullName)
```

```
[1] "Mary.LastName"      "Anna.LastName"      "Emma.LastName"
[4] "Elizabeth.LastName" "Minnie.LastName"    "Margaret.LastName"
```

\(4\) Find all "Elizabeth" in the data and replace "Elizabeth" with "Liz".

```
#Type your code here
data_bn <- data_bn %>%
  mutate(Name = gsub("^Elizabeth$", "Liz", Name))
head(data_bn$Name)
```

```
[1] "Mary"     "Anna"     "Emma"     "Liz"      "Minnie"   "Margaret"
```