Workshop 4: Extracting and Exporting Data Subsets

Author's name here

July 2024

Introduction

Welcome!

For this week's assignment, you will be carrying out a new task as a data analyst: preparing data subsets for **someone else to use**. It is a hands-on approach to using the `select()` and `filter()` verbs. This will build on the following lessons from the pre-work (linked below):

- **Select & rename**
- **Filter**

The lesson notes can be very helpful for completing the exercise below, so do refer to them during the workshop!

The due date for the assignment is **Friday, August 2nd at 23:59 GMT**. The assignment should be submitted individually, but you are encouraged to brainstorm with partners.

Set Up

To get started, you should download, unzip, and look through the assignment folder.

1. **Open the RStudio Project** by clicking on the relevant *.Rproj* file in the unzipped workshop folder. Make sure you have **opened the week_04_workshop project in RStudio**.
2. In RStudio, navigate to the Files tab and open the *"rmd"* folder. The file called **week_04_exercise.Rmd** contains both the instructions and the workshop assignment in one file (these are the same instructions you see here).
3. In the *"data"* folder, open the metadata (data dictionary) file, "**rabies_metadata.pdf.**" This data dictionary was provided to define the variables in **"rabies_dataset.csv."** It is necessary because the data for today is numerically encoded; that is, all variables are stored as numbers which correspond to categories. Without the variable dictionary, you wouldn't know what the numeric codes correspond to!

---

1. Load and Clean the Data

Now that you understand the structure of the repo, you can load in and clean your dataset.

In the code section below, **load in the needed packages** (hint: load {tidyverse} and {here}).

Pro tip: Use `p_load()` to load in your packages, since it both loads and installs packages as needed.

```
# Load packages
if(!require(pacman)) install.packages("pacman")
```

```
## Loading required package: pacman
```

```
"WRITE_YOUR_CODE_HERE"
```

```
## [1] "WRITE_YOUR_CODE_HERE"
```

Now, **read the dataset into R**. The data frame you import should have 1466 rows and 23 columns. Remember to use the `here()` function to allow your Rmd to use project-relative paths.

```
# Import data
rabies_raw <- "WRITE_YOUR_CODE_HERE"
```

The dataset comes from a study aimed to assess the extent of knowledge and understanding of rabies disease in rural and urban communities of Pakistan.

Explore the data frame with functions like `glimpse()`, `summary()`, `names()`, or `view()`.

Next, perform the following two cleaning tasks on the imported dataset, then store the cleaned dataset in a new object.

- **Bring the respondent ID to the front**. The respondent ID is the 23rd column of your dataset. It should be more visible so that someone who opens up the CSV knows immediately that each row corresponds to a respondent. Move it to the first position in the data frame. (Hint: remember the `everything()` function which we taught you. You can also use `relocate()`)
- **Remove the `Education` variable, which has not been properly encoded.** (If you want to see what the encoding issues are, you can look at both the data frame and the metadata file). We are considering this variable unusable for now, and are therefore removing it.

```
# Clean data
rabies <- "WRITE_YOUR_CODE_HERE"
```

2. Create and Export Data Subsets

In each "Data subset" section below, you should:

1) Determine whether to use the `filter()` or `select()` function, then apply that function to create the required extract of the dataset.
2) Export each data subset into an appropriately-named CSV file in the outputs folder.

Data Subset 1: Extract demographic information and vaccination indicators

Create and export a data subset containing the variables related to the following topics: 1. Respondents' demographic information — their age, gender and geographic background 2. Variables related to vaccination (Hint: use `contains()` or `ends_with()` to select the 7 variables about vaccination).

```
# Subset the data
demog <- "WRITE_YOUR_CODE_HERE"

# Export the data
"WRITE_YOUR_CODE_HERE"
```

```
## [1] "WRITE_YOUR_CODE_HERE"
```

Remember to use the `here()` function to allow your Rmd to use project-relative paths.

Data Subset 2: Extract the "knowledge-evaluation" question variables

From looking through the metadata file, you will see that some variables correspond to "knowledge-evaluation" questions about rabies. Create and export a subset that includes just these variables.

```
# Subset the data
"WRITE_YOUR_CODE_HERE"
```

```
## [1] "WRITE_YOUR_CODE_HERE"
```

```
# Export the data
"WRITE_YOUR_CODE_HERE"
```

```
## [1] "WRITE_YOUR_CODE_HERE"
```

Data Subset 3: Extract all male adults

Create and export a data subset containing only records for males aged over 18.

Hint: this is a row-filtering question, so there is no need to select or drop columns.

```
# Subset the data
"WRITE_YOUR_CODE_HERE"
```

```
## [1] "WRITE_YOUR_CODE_HERE"
```

```
# Export the data
"WRITE_YOUR_CODE_HERE"
```

```
## [1] "WRITE_YOUR_CODE_HERE"
```

Data Subset 4: Extract at-risk individuals

Create and export a subset with "at-risk" individuals. We define "at-risk" as people who meet any of the following criteria:

1. have a pet at home,
2. have no health facility in their area, OR
3. consider that the rabies vaccine is not affordable for them.

```
# Subset the data
"WRITE_YOUR_CODE_HERE"
```

```
## [1] "WRITE_YOUR_CODE_HERE"
```

```
# Export the data
"WRITE_YOUR_CODE_HERE"
```

```
## [1] "WRITE_YOUR_CODE_HERE"
```

Data Subset 5: Extract respondents with "ideal" knowledge, attitudes and practices (KAPs) towards rabies

We will define people with *ideal* KAPs as people who answered that they meet all of the below criteria:

- know that dog bites transmit rabies;
- know the clinical signs for rabies; and
- would visit a doctor after being bitten by an animal

Create and export a data subset that includes just these individuals.

```
# Subset the data
"WRITE_YOUR_CODE_HERE"
```

```
## [1] "WRITE_YOUR_CODE_HERE"
```

```
# Export the data
"WRITE_YOUR_CODE_HERE"
```

```
## [1] "WRITE_YOUR_CODE_HERE"
```

3. Submission: Upload your Rmd file to GRAPH Courses website

Once you have finished the tasks above, you should upload just the Rmd file to the assignment page in the GRAPH Courses website. You do not need to upload the data exports; **just the Rmd file is enough**. We will be able to tell whether your exports were correct by just looking at your Rmd file.

***NOTE:** It is recommended that you remove code and text that is not relevant to your assignment from the Rmd before submitting. For example, if you did not complete the challenge assignment outlined below, you can remove that.

Challenge Assignment

If you finish your main task on time, you can tackle this optional challenge section.

Your task for this challenge is to export the same data subsets created above into an **Excel workbook**, with each subset in a separate worksheet.

Why is this useful? Excel is a widely-used and familiar tool for many professionals including public health officials. As a data analyst, you often need to present your findings in a format that is accessible and understandable to these individuals. Excel is such a tool.

Moreover, Excel workbooks allow you to organize multiple related tables into separate sheets within a single file, enhancing the clarity and navigability of your output.

We have not formally taught you how to export to Excel files yet, so we've provided a tutorial at the bottom of this document, and a link to the documentation for the package you will be using.

CSV vs Excel files

Not sure about the difference between a CSV (.csv) and an Excel (.xls or .xlsx) file? Here is a quick summary:

CSV Files (.csv):

- CSV stands for Comma-Separated Values. These files are plain text files where each piece of data is separated by a comma.
- Here are some of their main characteristics:
  - **Simplicity**: CSV files can be opened with any text editor, making them easy to view and manipulate.
  - **Compatibility**: They are widely supported across different platforms and programming languages. You can import and export CSV files in most data processing software, including Microsoft Excel, Google Sheets, and any database management system.
  - **No Formatting**: CSV files do not support any text formatting or styles. This means they can't store information such as font styles, colors, or sizes.
  - **No Multiple Sheets**: Unlike Excel files, CSV files can't contain multiple sheets. Each CSV file holds a single table of data.

Excel Files (.xls or .xlsx):

- Excel files are Microsoft's proprietary binary format for storing data in workbooks, which can contain one or more worksheets.
- Some key features of Excel files include:
  - **Multiple Sheets**: Excel workbooks can contain multiple sheets, allowing you to organize related data sets within a single file.
  - **Rich Formatting**: You can format text in Excel in a variety of ways, from changing the font, color, and size to more advanced styling options like conditional formatting. These formats can be set from within R itself, as you will see below.

Here is a short video that explains some of this: https://youtu.be/hlbRgI45_90

Challenge Instructions

Now, here are the step-by-step instructions for the challenge assignment:

1. You will start by creating a custom data frame that will describe the content of each sheet in the workbook. You can do this using the `data.frame()` function. This will serve as a table of contents for your Excel file and will be placed in the first sheet.
2. Using the `openxlsx` package, you will then export each data subset into its own sheet in a single Excel workbook. Ensure that the first sheet in the workbook is the descriptive table you created in the first step.
3. Each column header should be in bold font. This can be achieved by defining a custom style using the `createStyle()` function and applying it during the export process.

For a detailed tutorial on how to write to Excel using the `openxlsx` package, see the official documentation of `openxlsx` package here.

Challenge Example

To guide you, here's a reproducible example using the built-in `iris` dataset. The example creates subsets of data for each species of iris and exports them into separate sheets of an Excel workbook. It also creates a table describing each subset:

```
# Load required packages
if(!require(pacman)) install.packages("pacman")
pacman::p_load(tidyverse, here, openxlsx)

# Create subsets of the iris data for each species
iris_setosa <- iris %>% filter(Species == "setosa")
iris_virginica <- iris %>% filter(Species == "virginica")
iris_versicolor <- iris %>% filter(Species == "versicolor")

# Create a data frame that describes what each excel sheet will contain
# This description data frame will be stored in the first sheet of the Excel file
description_df <- data.frame(
  Sheet_Name = c("Iris Setosa", "Iris Virginica", "Iris Versicolor"),
  Description = c("This sheet contains a data subset for Iris Setosa",
                  "This sheet contains a data subset for Iris Virginica",
                  "This sheet contains a data subset for Iris Versicolor")
)

# Print description_df to the console for verification
description_df
```

```
##        Sheet_Name                                          Description
## 1     Iris Setosa     This sheet contains a data subset for Iris Setosa
## 2  Iris Virginica  This sheet contains a data subset for Iris Virginica
## 3 Iris Versicolor This sheet contains a data subset for Iris Versicolor
```

```r
# Create a custom style for the file headers using 'createStyle()' from the 'openxlsx' package
# Style is: bold text, white font color, and blue fill
my_header_style <- createStyle(
  textDecoration = "BOLD", fontColour = "#FFFFFF", fgFill = "#4F80BD"
)

# Create a list 'iris_subsets' containing the table we will be storing in Excel
# These are the descrition data frame all three iris data subsets
# Each element in the list is given a name: "Description", "Iris Setosa", "Iris Virginica" and "Iris Ve
iris_subsets <- list("Description" = description_df,
                     "Iris Setosa" = iris_setosa,
                     "Iris Virginica" = iris_virginica,
                     "Iris Versicolor" = iris_versicolor)

# Write the 'iris_subsets' list into an Excel workbook using 'write.xlsx' function from 'openxlsx' pack
# Specify the file path using 'here' function for portability
# 'headerStyle = my_header_style' applies the previously defined custom style to the headers
write.xlsx(
  x = iris_subsets,
  file = here("outputs/iris_subsets_challenge_assignment_example.xlsx"),
  headerStyle = my_header_style
)
```

Your turn

Now, try exporting the rabies data subsets you created in the assignment to an Excel workbook. Make sure to write a short description of each subset in the first sheet of the workbook. At the end, you should have 6 sheets in your workbook. Good luck!

```r
"YOUR CODE FOR THE CHALLENGE QUESTION HERE"
```

```
## [1] "YOUR CODE FOR THE CHALLENGE QUESTION HERE"
```