Lab 3 - Data tidying and joining

Introduction

In this lab you'll build the data wrangling and visualization skills you've developed so far and data tidying and joining to your repertoire.

> **i Note**
>
> This lab assumes you've completed the labs so far and doesn't repeat setup and overview content from those labs. If you have not yet done those, you should go back and review the previous labs before starting on this one.

Learning objectives

By the end of the lab, you will…

- Be able to pivot/reshape data using `tidyr`
- Continue developing your data wrangling skills using `dplyr`
- Build on your mastery of data visualizations using `ggplot2`
- Get more experience with data science workflow using R, RStudio, Git, and GitHub
- Further your reproducible authoring skills with Quarto
- Improve your familiarity with version control using Git and GitHub

Getting started

Log in to RStudio, clone your `lab-3` repo from GitHub, open your `lab-3.qmd` document, and get started!

Click here if you prefer to see step-by-step instructions

Log in to RStudio

- Go to https://cmgr.oit.duke.edu/containers and log in with your Duke NetID and Password.
- Click `STA198-199` under My reservations to log into your container. You should now see the RStudio environment.

Clone the repo & start new RStudio project

- Go to the course organization at github.com/sta199-s24 organization on GitHub. Click on the repo with the prefix **lab-3**. It contains the starter documents you need to complete the lab.

- Click on the green **CODE** button, select **Use SSH** (this might already be selected by default, and if it is, you'll see the text **Clone with SSH**). Click on the clipboard icon to copy the repo URL.

- In RStudio, go to *File* *New Project* *Version Control* *Git*.

- Copy and paste the URL of your assignment repo into the dialog box *Repository URL*. Again, please make sure to have *SSH* highlighted under *Clone* when you copy the address.

- Click *Create Project*, and the files from your GitHub repo will be displayed in the *Files* pane in RStudio.

- Click *lab-3.qmd* to open the template Quarto file. This is where you will write up your code and narrative for the lab.

First steps

In `lab-3.qmd`, update the `author` field to your name, render your document and examine the changes. Then, in the Git pane, click on **Diff** to view your changes, add a commit message (e.g., "Added author name"), and click **Commit**. Then, push the changes to your GitHub repository, and in your browser confirm that these changes have indeed propagated to your repository.

! Important

If you run into any issues with the first steps outlined above, flag a TA for help before proceeding.

Packages

In this lab we will work with the **tidyverse** package, which is a collection of packages for doing data analysis in a "tidy" way.

```r
library(tidyverse)
```

Render the document which loads this package with the `library()` function.

Guidelines

As we've discussed in lecture, your plots should include an informative title, axes should be labeled, and careful consideration should be given to aesthetic choices.

In addition, the code should all the code should be be able to be read (not run off the page) when you render to PDF. Make sure that is the case, and add line breaks where the code is running off the page.[1]

> Note
>
> Continuing to develop a sound workflow for reproducible data analysis is important as you complete the lab and other assignments in this course. There will be periodic reminders in this assignment to remind you to **render, commit, and push** your changes to GitHub. You should have at least 3 commits with meaningful commit messages by the end of the assignment.

Questions

Part 1

Inflation across the world

For this part of the analysis you will work with inflation data from various countries in the world over the last 30 years.

```r
country_inflation <- read_csv("data/country-inflation.csv")
```

[1] Remember, haikus not novellas when writing code!

Question 1

Get to know the data.

a. `glimpse()` at the `country_inflation` data frame and answer the following questions based on the output. How many rows does `country_inflation` have and what does each row represent? How many columns does `country_inflation` have and what does each column represent?

b. Display a list of the countries included in the dataset.

> Tip
>
> A function that can be useful for part (b) is `pull()`. Check out its documentation for examples of usage.

Question 2

Which countries had the top three highest inflation rates in 2021? Your output should be a data frame with two columns, `country` and `2021`, with inflation rates in descending order, and three rows for the top three countries. Briefly comment on how the inflation rates for these countries compare to the inflation rate for United States in that year.

> Tip
>
> Column names that are numbers are not considered "proper" in R, therefore to select them you'll need to surround them with backticks, e.g. `` select( ` 1993 ` ) ``.

Question 3

In a single pipeline,

- calculate the ratio of the inflation in 2021 and inflation in 1993 for each country and store this information in a new column called `inf_ratio`,
- arrange the data frame in decreasing order of `inf_ratio`, and
- select the variables `country` and `inf_ratio` to display as the result of the pipeline.

Do not save this new variable in `inf_ratio`, only calculate and display it so you can answer the following question based on the output of the pipeline.

Which country's inflation change is the largest over this time period? Did inflation increase of decrease between 1993 and 2021 in this country?

Tip

For this question you'll once again need to use variables whose names are numbers (years) in your pipeline. Make sure to surround the names of such variables with backticks (`).

Question 4

Reshape (pivot) `country_inflation` such that each row represents a country/year combination, with columns `country`, `year`, and `annual_inflation`. Then, display the resulting data frame and state how many rows and columns it has.

Requirements:

- Your code must use one of `pivot_longer()` or `pivot_wider()`. There are other ways you can do this reshaping move in R, but this question requires solving this problem by pivoting.
- In your `pivot_*()` function, you must use `names_transform = as.numeric` as an argument to transform the variable type to numeric as you pivot the data so that in the resulting data frame the year variable is numeric.
- The resulting data frame must be saved as something other than `country_inflation` so you (1) can refer to this data frame later in your analysis and (2) do not overwrite `country_inflation`. Use a short but informative name.

! Important

The remaining questions in Part 1 require the use of the pivoted data frame from Question 4.

Question 5

Use a separate, single pipeline to answer each of the following questions.

Requirement: Your code must use the `filter()` function for each part, not `arrange()`.

a. What is the highest inflation rate observed between 1993 and 2021? The output of the pipeline should be a data frame with one row and three columns. In addition to code and output, your response should include a single sentence stating the country and year.

b. What is the lowest inflation rate observed between 1993 and 2021? The output of the pipeline should be a data frame with one row and three columns. In addition to code and output, your response should include a single sentence stating the country and year.

c. Putting (a) and (b) together: What are the highest and the lowest inflation rates observed between 1993 and 2021? The output of the pipeline should be a data frame with two rows and three columns.

Question 6

a. Create a vector called `countries_of_interest` which contains the names of up tp five countries you want to visualize the inflation rates for over the years. For example, if these countries are Türkiye and United States, you can express this as follows:

```r
countries_of_interest <- c("Türkiye", "United States")
```

If they are Türkiye, United States, and China, you can express this as follows:

```r
countries_of_interest <- c(
  "Türkiye", "United States", "China (People's Republic of)"
)
```

So on and so forth… Then, in 1-2 sentences, state why you chose these countries.

> ℹ Note
>
> Your `countries_of_interest` should consist of no more than five countries. Make sure that the spelling of your countries matches how they appear in the dataset.

b. In a single pipeline, filter your reshaped dataset to include only the `countries_of_interest` from part (a), and save the resulting data frame with a new name so you (1) can refer to this data frame later in your analysis and (2) do not overwrite the data frame you're starting with. Use a short but informative name. Then, in a new pipeline, find the `distinct()` countries in the data frame you created.

> Tip
>
> The number of distinct countries in the filtered data frame you created in part (b) should equal the number of countries you chose in part (a). If it doesn't, you might have misspelled a country name or made a mistake in how to filter for these countries. Go back and check your code.

Question 7

Using your data frame from the previous question, create a plot of annual inflation vs. year for these countries. Then, in a few sentences, describe the patterns you observe in the plot, particularly focusing on anything you find surprising or not surprising, based on your knowledge (or lack thereof) of these countries economies.

Requirements for the plot:

- Data should be represented with points as well as lines connecting the points for each country.
- Each country should be represented by a different color line and different color and shape points.
- Axes and legend should be properly labeled.
- The plot should have an appropriate title (and optionally a subtitle).
- Plot should be customized in at least one way – you could use a different than default color scale, or different than default theme, or some other customization.

If you haven't yet done so, now is a good time to render, commit, and push. Make sure that you commit and push all changed documents and your Git pane is completely empty before proceeding.

Part 2

Inflation in the US

The OECD defines inflation as follows:

> Inflation is a rise in the general level of prices of goods and services that households acquire for the purpose of consumption in an economy over a period of time.
>
> The main measure of inflation is the annual inflation rate which is the movement of the Consumer Price Index (CPI) from one month/period to the same month/period of the previous year expressed as percentage over time.
>
> Source: OECD CPI FAQ

CPI is broken down into 12 divisions such as food, housing, health, etc. Your goal in this part is to create another time series plot of annual inflation, this time for US only.

The data you will need to create this visualization is spread across two files:

- `us-inflation.csv`: Annual inflation rate for the US for 12 CPI divisions. Each division is identified by an ID number.
- `cpi-divisions.csv`: A "lookup table" of CPI division ID numbers and their descriptions.

Let's load both of these files.

```r
us_inflation <- read_csv("data/us-inflation.csv")
cpi_divisions <- read_csv("data/cpi-divisions.csv")
```

Question 8

a. How many columns and how many rows does the `us_inflation` dataset have? What are the variables in it? Add a brief (1-2 sentences) narrative summarizing this information.

b. How many columns and how many rows does the `cpi_divisions` dataset have? What are the variables in it? Add a brief (1-2 sentences) narrative summarizing this information.

c. Create a new dataset by joining the `us_inflation` dataset with the `cpi_division_id` dataset.

- Determine which type of join is the most appropriate one and use that.
- Note that the two datasets don't have a common variable. Review the help for the join functions to determine how to use the `by` argument when the names of the variables that the datasets should be joined by are different.
- Use a short but informative name for the joined dataset, and do not overwrite either of the datasets that go into creating it.

Then, find the number of rows and columns of the resulting dataset and report the names of its columns. Add a brief (1-2 sentences) narrative summarizing this information.

Question 9

a. Create a vector called `divisions_of_interest` which contains the descriptions or IDs of CPI divisions you want to visualize. Your `divisions_of_interest` should consist of no more than five divisions. If you're using descriptions, make sure that the spelling of your divisions matches how they appear in the dataset. Then, in 1-2 sentences, state why you chose these divisions.

> Tip
>
> Refer back to the guidance provided in Question 6 if you're not sure how to create this vector.

b. In a single pipeline, filter your reshaped dataset to include only the `divisions_of_interest` from part (a), and save the resulting data frame with a new name so you (1) can refer to this data frame later in your analysis and (2) do not overwrite the data frame you're starting with.

Use a short but informative name. Then, in a new pipeline, find the `distinct()` divisions in the data frame you created.

Question 10

Using your data frame from the previous question, create a plot of annual inflation vs. year for these divisions. Then, in a few sentences, describe the patterns you observe in the plot, particularly focusing on anything you find surprising or not surprising, based on your knowledge (or lack thereof) of inflation rates in the US over the last decade.

- Data should be represented with points as well as lines connecting the points for each division.
- Each division should be represented by a different color line and different color and shape points.
- Axes and legend should be properly labeled.
- The plot should have an appropriate title (and optionally a subtitle).
- Plot should be customized in at least one way – you could use a different than default color scale, or different than default theme, or some other customization.
- If your legend has labels that are too long, you can try moving the legend to the bottom and stack the labels vertically. *Hint:* The `legend.position` and `legend.direction` arguments of the `theme()` functions will be useful.
- Edit the code chunk options so the code chunk is evaluated!

```r
ggplot(...) +
  ... +
  theme(
    legend.position = "bottom",
    legend.direction = "vertical"
  )
```

If you haven't yet done so since Part 1, now is a good time to render, commit, and push. Make sure that you commit and push all changed documents and your Git pane is completely empty before proceeding.

Wrap-up

Submission

Once you are finished with the lab, you will submit your final PDF document to Gradescope.

⚠ Warning

Before you wrap up the assignment, make sure all of your documents are updated on your GitHub repo. We will be checking these to make sure you have been practicing how to commit and push changes.
You must turn in a PDF file to the Gradescope page by the submission deadline to be considered "on time".

To submit your assignment:

- Go to http://www.gradescope.com and click *Log in* in the top right corner.
- Click *School Credentials* → *Duke NetID* and log in using your NetID credentials.
- Click on your *STA 199* course.
- Click on the assignment, and you'll be prompted to submit it.
- Mark all the pages associated with question. All the pages of your lab should be associated with at least one question (i.e., should be "checked").

❗ Checklist

Make sure you have:

- attempted all questions
- rendered your Quarto document
- committed and pushed everything to your GitHub repository such that the Git pane in RStudio is empty
- uploaded your PDF to Gradescope
- selected pages associated with each question on Gradescope

Grading

The lab is graded out of a total of 50 points.

You can earn up to 5 points on each question:

- 5: Response shows excellent understanding and addresses all or almost all of the rubric items.
- 4: Response shows good understanding and addresses most of the rubric items.
- 3: Response shows understanding and addresses a majority of the rubric items.
- 2: Response shows effort and misses many of the rubric items.
- 1: Response does not show sufficient effort or understanding and/or is largely incomplete.

- 0: No attempt.