



[Return to Classroom](#)

Real World Data Wrangling with Python

REVIEW

HISTORY

Meets Specifications

Hi there,

Congratulations for passing the Data Wrangling Project 🎉, you have nailed this at all crucial parts and fulfilled the notebook and process.

You can further deep dive into Webscraping by combining [requests](#) library with [Beautiful Soup](#) and if need be with [Selenium's webdriver](#) if you need to scrape from javascript-implemented buttons in html with a webdriver.

Good luck for your upcoming projects! 

Code Quality and Submission Phase

All code is functional (i.e. no errors are thrown by the code). Warnings are okay, as long as they are not a result of poor coding practices.

[Awesome] All code cells can be run without error, great job!




For future jupyter notebooks, in which you may work in more complex projects have a look at this [debugging toolkit](#) that allows to modify potential error messages according to need


Students' .zip folder must contain the following:

- A Jupyter Notebook with completed code
- Source datasets or a .txt file containing the links to the datasets, if data is downloaded manually
- the data gathering code in the notebook, if data is gathered programmatically,
- the raw datasets that are stored locally
- the final cleaned dataset (after variables are combined) that is stored locally.

Thank you for successfully submitting the worked through notebooks, further it is suggested to add areports in form of a .pdf or .html version.

See [here](#) for further elaborations on why this is important for cross-platform specific readability/functionality of the project.

The submission contains all base datasets and reference to the sources as links or txt-files 

Cleaned and modified base datasets as well as merged master datasets added to submission 

For each stage of the process, students must explain:

- Their justifications behind the method used for gathering, assessment, cleaning, storing, answering the research question.
- Write code comments for code readability

All code and text blocks in the notebook denoted "FILL IN" should be filled in by the student.

In the last section of the notebook, "Answering the Research Question", there must be at least 2 plots generated.

Nicely done! 🙌, code comments are an essential part of data analysis projects. They serve multiple purposes, including:

- **Documentation:** Comments provide explanations and descriptions of the code, making it easier for others (including yourself in the future) to understand the purpose and functionality of different parts of the code.
- **Clarity and Readability:** Comments help improve the readability of the code by providing additional context and explanations. They make the code more understandable and maintainable, especially for complex data analysis tasks.
- **Collaboration:** When working in a team, comments help facilitate collaboration by allowing team members to understand and contribute to the codebase more effectively. They provide a way to communicate ideas, suggestions, and improvements within the code itself.

- **Debugging and Troubleshooting:** Comments can be helpful during the debugging process. By adding comments that explain the logic or reasoning behind certain code sections, it becomes easier to identify and fix any issues or errors.

💡 Here are a few resources that provide more information on the importance of code comments in data analysis:

[The Importance of Code Comments in Data Analysis](#) - This article discusses the significance of code comments in data analysis and provides examples of good commenting practices.

[Best Practices for Writing Code Comments](#) - This Real Python article covers best practices for writing code comments in Python, including guidelines for data analysis projects.

Remember, comments should be clear, concise, and relevant. They should explain the why behind the code, not just the what. Do you have any further questions?

Gathering, Assessment, and Cleaning

Students write 2-4 full sentences explaining the kind of problem they will want to look at and the datasets they will be wrangling for this project.

Good job! ✓ Writing an introduction in data analysis is important for providing context, engaging the reader, guiding the analysis, and ensuring reproducibility. It helps set the stage, capture interest, and outline the steps of the analysis.

Here are two links for further reading:

1. [The Importance of Writing an Introduction in Data Analysis](#)
2. [Tips for Writing an Effective Introduction in Data Analysis](#)

Students should pick at least two different gathering methods from the list below:

- Download data manually
- Programmatically downloading files
- Gather data by accessing APIs
- Gather and extract data from HTML files using BeautifulSoup
- Extract data from a SQL database

Students must then gather at least two datasets. Each dataset must have at least two variables and have greater than 500 data samples within each dataset.

For each dataset, students must describe in 2-3 full sentences:

- Why they picked the dataset
- The gathering method

- The names and significance of the variables in the dataset.

For applicable data gathering methods, such as parsing HTML, extracting SQL data, and programmatic API access, students must show their work (e.g., if using an API to download the data, please include a snippet of your code in the notebook).

Students must load the dataset programmatically into this notebook.

Excellent! 🙌

You have utilised 2 very effective ways of gathering data, commonly if available, an API is preferred as most stable gathering methodology.

To summarize methodologies that stand to your disposal:

📄 Manual Data Download: Manually downloading data files from a source, see [Python docs](#)

📡 Programmatic Data Download: Programmatically downloading data files using Python libraries like [requests](#) or [urllib](#)

🌐 API Access: Retrieving data through APIs, APIs are different per webpage, partly in-house or third party solutions, see this [API tutorial](#)

🔍 HTML Parsing: Extracting data from HTML files using libraries like [BeautifulSoup](#)

📄 SQL Database: Extracting data from a SQL database using libraries like sqlite3 or pymysql, see this [tutorial about sql-libraries](#)

Students should list two data quality issues and two tidiness issues with the datasets they have selected. For each data issue, they should:

- Briefly describe the issue they find.
- Assess the issue visually and programmatically using the methods discussed in the course (e.g., `df.head()`, `df.describe()`).
- Justify the assessment methods.

List of data quality issues:

- Completeness
- Validity
- Accuracy
- Consistency
- Uniqueness

Excellent, now you have correctly identified 2 Quality Issues and 2 Tidiness Issues,

💡 Just for review, here's a summary of the characteristics of **quality issues** and **tidiness issues** in data:

Quality Issues:

Inaccurate data: Data that contains errors, mistakes, or inconsistencies.

Missing data: Data that is incomplete or has missing values.

Duplicate data: Data that is repeated or duplicated within the dataset.

Outliers: Data points that are significantly different from other data points.

Inconsistent data: Data that is inconsistent in format, units, or naming conventions.

Tidiness Issues:

Untidy data structure: Data that is not organized or structured in a consistent and logical manner.

Columns containing multiple variables: Data that has multiple variables stored in a single column.

Rows containing multiple observations: Data that has multiple observations stored in a single row.

Inconsistent data types: Data that has inconsistent data types within the same column.

Redundant information: Data that contains unnecessary or duplicate information.

These issues can affect the reliability and usability of the data, and it's important to address them during the data cleaning process.

Students should clean the dataset to solve the 4 issues corresponding to data quality and tidiness in the assessment step. For each issue cleaned, they must:

- Include justifications for the cleaning method used and cleaning decisions.
- Use either the visual or programmatic method to validate that the cleaning was successful

Great job, you have cleaned the listed 4 issues like a good citizen 🙌



Here are some useful Pandas methods for cleaning quality and tidiness issues in data:

Quality Issues:

dropna: Drops rows or columns with missing values.

fillna: Fills missing values with specified values or methods.

drop_duplicates: Drops duplicate rows from the dataset.

replace: Replaces specific values in the dataset with new values.

query: Filters the dataset based on specific conditions.

Tidiness Issues:

melt: Unpivots columns into rows, creating a tidy structure.

pivot: Pivots rows into columns, reshaping the data.

merge: Combines two or more datasets based on a common column.

str.split: Splits a string column into multiple columns based on a delimiter.

astype: Converts the data type of a column to a specified type.

These methods can help you address and clean up various quality and tidiness issues in your data.

Students must remove unnecessary variables for their analysis and combine their datasets. After combining the data, the final dataset must have at least 4 variables.

Unnecessary variables of cleaned datasets removed prior merging ✓

Afterwards you have combined the dataframes correctly, have a look at the [different pandas methods](#) you can use 💡

Data Storage and Answering the Research Question

Update your database/data store with the cleaned data

- Students must maintain different instances/versions of data (raw data and cleaned data)
- Students must name their dataset files informatively
- Students must ensure both the raw and cleaned data are saved to their database/data store

Note: Students are not required to use a relational/non-relational database store.

Store updated, combined dataset saved with [pandas.to_csv\(\)](#) ✓

[Suggestion] If your dataframe has a usable **identifier-column**, it is beneficial to not to store the dataset with an additional index, which is why the argument `index=False` is often good practice, see this [discussion](#).

💡 Certainly for datasets as such **storage to .csv file is the simplest** and in this case sufficient mode of storage, however for advanced or larger data, you will need to consider using data-storage solutions like Microsoft Azure. To give a quick overview, these are the storage possibilities over there:

Azure SQL Database: Azure SQL Database is a fully managed relational database service that offers high performance, scalability, and security. It is compatible with SQL Server and provides features such as automatic backups, built-in intelligence, and advanced security capabilities.

Azure Cosmos DB: Azure Cosmos DB is a globally distributed, multi-model database service that supports multiple data models, including document, key-value, graph, and columnar. It offers low latency, automatic scaling, and global distribution, making it suitable for applications that require high throughput and low latency access to structured data.


Azure Data Lake Storage: Azure Data Lake Storage is a scalable and secure data lake solution that allows you to store and analyze large amounts of structured, semi-structured, and unstructured data. It provides a hierarchical file system and integrates well with other Azure services such as Azure Databricks and Azure Data Factory.

Azure Table Storage: Azure Table Storage is a NoSQL key-value store that is ideal for storing large amounts of structured data. It offers high availability, scalability, and durability, and can be accessed using simple REST APIs. Azure Table Storage is a cost-effective option for scenarios that require massive scale and fast

access to structured data.


Students must use the final cleaned data to answer the question they raised from the problem statement in Step 1.

Students must produce at least two visualizations using the cleaned data and explain how they help the student answer the question (in 1-2 full sentences).


Orderly your plots have labels and titles, and can be interpreted with ease 

Now you will learn later on that in Data Visualizations that we distinguish between **explorative and explanatory visualisations**. In this notebook we generally perform **explorative**, in which we plot to gain insights and interpret, while in **explanatory analysis** we plot to communicate our found insights. The latter must be very polished and representable plots, while the former still needs titles and labels to follow through as author and reader.

In 2-4 sentences, students must describe what actions they would take if they had more time to complete the project.

In this submission you have additionally explained what next steps to take if more time was available, which is sound 

The minimal requirement really is to make a statement in regards to the mentioned, what is also of high interest is whether you see any limitations of your analysis so far, as this is a crucial step in analytics, see more on this topic in this [blogpost](#).

 Also have a look at the resource about how to effectively [write a conclusion](#) and how a conclusion is more than a summary.

 [DOWNLOAD PROJECT](#)

[RETURN TO PATH](#)