Data 607- Week 7 Assignment

shri Tripathi

2024-10-17

Handling and Importing Various Data Formats

Overview:

This week's assignment focused on creating four files in different formats—JSON, HTML, XML, and Parquet. Each file contained a structured table of information including attributes such as title, author, language, year of publication, and more. The goal was to demonstrate how to create data tables in these formats and practice importing them into R as data frames. By working with multiple formats, we gained insights into the versatility of each format and their use cases in data storage and analysis. This task also helped us strengthen our skills in reading and handling diverse data formats in R

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.0     v readr     2.1.5
## v ggplot2   3.4.4     v stringr   1.5.1
## v lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

HTML

To import the HTML data, I first created and uploaded the HTML file containing the information to GitHub. I used the `rvest` package in R to handle the extraction process. I passed the GitHub URL through the `read_html()` function, which retrieves the raw HTML data and stores it as an `xml_document` in the form of an `xml_node`. Then, using the `html_nodes()` function from the `dplyr` package, I located the table within the HTML data by specifying the "table" argument. This method efficiently extracts the table from the HTML structure for further analysis.

After extracting the data from the table, I place it in the data frame html_data.

```r
# Load the necessary library for web scraping
library(rvest)
```

```
## Warning: package 'rvest' was built under R version 4.3.3

##
## Attaching package: 'rvest'

## The following object is masked from 'package:readr':
##
##     guess_encoding
```

```r
# URL to the HTML file on GitHub containing the table of data
html_url = "https://raw.githubusercontent.com/Shriyanshh/Week-7-Assignment/refs/heads/main/data.html"

# Read the HTML content from the GitHub URL
git_html = read_html(html_url)

# Check the class of the object to ensure it has been read correctly as HTML
class(git_html)
```

```
## [1] "xml_document" "xml_node"
```

```r
# Extract the first table from the HTML document into a data frame
html_data = git_html %>%
  html_nodes("table") %>%  # Find all the tables in the HTML document
  .[[1]] %>%               # Select the first table (if there are multiple tables)
  html_table()             # Convert the HTML table into a data frame

# Display the extracted data frame
html_data
```

```
## # A tibble: 20 x 7
##    Category         `Item Name`       `Item ID` Brand      Price `Variation ID`
##    <chr>            <chr>                 <int> <chr>      <dbl> <chr>
##  1 Electronics      Smartphone              101 TechBrand   700.  101-A
##  2 Electronics      Smartphone              101 TechBrand   700.  101-B
##  3 Electronics      Laptop                  102 CompuBrand 1100.  102-A
##  4 Electronics      Laptop                  102 CompuBrand 1100.  102-B
##  5 Home Appliances  Refrigerator            201 HomeCool    900.  201-A
##  6 Home Appliances  Refrigerator            201 HomeCool    900.  201-B
##  7 Home Appliances  Washing Machine        202 CleanTech   500.  202-A
##  8 Home Appliances  Washing Machine        202 CleanTech   500.  202-B
##  9 Clothing         T-Shirt                301 FashionCo    20.0 301-A
## 10 Clothing         T-Shirt                301 FashionCo    20.0 301-B
## 11 Clothing         T-Shirt                301 FashionCo    20.0 301-C
## 12 Clothing         Jeans                  302 DenimWorks   50.0 302-A
## 13 Clothing         Jeans                  302 DenimWorks   50.0 302-B
## 14 Books            Fiction Novel          401 -            15.0 401-A
## 15 Books            Fiction Novel          401 -            15.0 401-B
## 16 Books            Non-Fiction Guide      402 -            25.0 402-A
## 17 Books            Non-Fiction Guide      402 -            25.0 402-B
## 18 Sports Equipment Basketball             501 SportsGear   30.0 501-A
## 19 Sports Equipment Tennis Racket          502 RacketPro    90.0 502-A
## 20 Sports Equipment Tennis Racket          502 RacketPro    90.0 502-B
## # i 1 more variable: ‘Variation Details‘ <chr>
```

JSON

To import the JSON data, I utilized both the `jsonlite` and `httr` packages. First, I used `httr`'s `GET()` function to retrieve the JSON data from the web page, storing the result in `git_json`, which holds the data as a response object. Next, I applied `httr`'s `content()` function to extract the raw data from the response and store it as a character vector. Following that, I used `jsonlite`'s `fromJSON()` function to parse the JSON file and convert it into a list. By setting `flatten=TRUE`, I ensured that the list was transformed into a wide data frame format. Finally, I converted the list into a data frame for further analysis.

```r
# Load the necessary libraries for working with JSON data and making web requests
library(jsonlite)  # For parsing JSON data
```

```
## Warning: package 'jsonlite' was built under R version 4.3.3

##
## Attaching package: 'jsonlite'

## The following object is masked from 'package:purrr':
##
##     flatten
```

```r
library(httr)      # For handling HTTP requests
```

```
## Warning: package 'httr' was built under R version 4.3.3
```

```r
# JSON file URL from GitHub
json_url = "https://raw.githubusercontent.com/Shriyanshh/Week-7-Assignment/refs/heads/main/data.json"

# Retrieve the JSON data from the GitHub URL using GET request
git_json = GET(json_url)

# Check the class of the response to confirm it's an HTTP response object
class(git_json)
```

```
## [1] "response"
```

```r
# Extract the content from the HTTP response and convert it into a text format
json_content = content(git_json, "text")

# Check the class of the extracted content (it should be a character vector)
class(json_content)
```

```
## [1] "character"
```

```r
# Parse the JSON content into an R object (list by default) and flatten it into a wide structure
json_data = fromJSON(json_content, flatten = TRUE)

# Check the class of the parsed JSON data (it should now be a list or data frame)
class(json_data)
```

```
## [1] "list"
```

```r
# Convert the JSON data (list) into a data frame for easier analysis
json_data = as.data.frame(json_data)

# Display the JSON data frame
json_data
```

```
##    Inventory.Category Inventory.ItemName Inventory.ItemID Inventory.Brand
## 1         Electronics         Smartphone              101       TechBrand
## 2         Electronics         Smartphone              101       TechBrand
## 3         Electronics             Laptop              102      CompuBrand
## 4         Electronics             Laptop              102      CompuBrand
## 5     Home Appliances       Refrigerator              201        HomeCool
## 6     Home Appliances       Refrigerator              201        HomeCool
## 7     Home Appliances    Washing Machine              202       CleanTech
## 8     Home Appliances    Washing Machine              202       CleanTech
## 9            Clothing            T-Shirt              301       FashionCo
## 10           Clothing            T-Shirt              301       FashionCo
## 11           Clothing            T-Shirt              301       FashionCo
## 12           Clothing              Jeans              302      DenimWorks
## 13           Clothing              Jeans              302      DenimWorks
## 14              Books      Fiction Novel              401               -
## 15              Books      Fiction Novel              401               -
## 16              Books  Non-Fiction Guide              402               -
## 17              Books  Non-Fiction Guide              402               -
## 18   Sports Equipment         Basketball              501      SportsGear
## 19   Sports Equipment      Tennis Racket              502       RacketPro
## 20   Sports Equipment      Tennis Racket              502       RacketPro
##    Inventory.Price Inventory.VariationID Inventory.VariationDetails.Color
## 1           699.99                 101-A                            Black
## 2           699.99                 101-B                            White
## 3          1099.99                 102-A                           Silver
## 4          1099.99                 102-B                       Space Gray
## 5           899.99                 201-A                  Stainless Steel
## 6           899.99                 201-B                            White
## 7           499.99                 202-A                             <NA>
## 8           499.99                 202-B                             <NA>
## 9           19.99               301-A                           Blue
## 10          19.99               301-B                            Red
## 11          19.99               301-C                          Green
## 12          49.99               302-A                      Dark Blue
## 13          49.99               302-B                     Light Blue
## 14          14.99               401-A                           <NA>
## 15          14.99               401-B                           <NA>
## 16          24.99               402-A                           <NA>
## 17          24.99               402-B                           <NA>
## 18          29.99               501-A                         Orange
## 19          89.99               502-A                          Black
## 20          89.99               502-B                         Silver
##    Inventory.VariationDetails.Storage Inventory.VariationDetails.Capacity
## 1                                64GB                                <NA>
## 2                               128GB                                <NA>
## 3                               256GB                                <NA>
## 4                               512GB                                <NA>
## 5                                <NA>                           20 cu ft
## 6                                <NA>                           18 cu ft
## 7                                <NA>                          4.5 cu ft
## 8                                <NA>                          5.0 cu ft
## 9                                <NA>                                <NA>
## 10                               <NA>                                <NA>
## 11                               <NA>                                <NA>
## 12                               <NA>                                <NA>
## 13                               <NA>                                <NA>
## 14                               <NA>                                <NA>
## 15                               <NA>                                <NA>
## 16                               <NA>                                <NA>
## 17                               <NA>                                <NA>
## 18                               <NA>                                <NA>
## 19                               <NA>                                <NA>
## 20                               <NA>                                <NA>
##    Inventory.VariationDetails.Type Inventory.VariationDetails.Size
## 1                             <NA>                            <NA>
## 2                             <NA>                            <NA>
## 3                             <NA>                            <NA>
## 4                             <NA>                            <NA>
## 5                             <NA>                            <NA>
## 6                             <NA>                            <NA>
## 7                       Front Load                            <NA>
## 8                         Top Load                            <NA>
## 9                             <NA>                               S
## 10                            <NA>                               M
## 11                            <NA>                               L
## 12                            <NA>                              32
## 13                            <NA>                              34
## 14                            <NA>                            <NA>
## 15                            <NA>                            <NA>
## 16                            <NA>                            <NA>
## 17                            <NA>                            <NA>
## 18                            <NA>                          Size 7
## 19                            <NA>                            <NA>
## 20                            <NA>                            <NA>
##    Inventory.VariationDetails.Format Inventory.VariationDetails.Language
## 1                               <NA>                                <NA>
## 2                               <NA>                                <NA>
## 3                               <NA>                                <NA>
## 4                               <NA>                                <NA>
## 5                               <NA>                                <NA>
## 6                               <NA>                                <NA>
## 7                               <NA>                                <NA>
## 8                               <NA>                                <NA>
## 9                               <NA>                                <NA>
## 10                              <NA>                                <NA>
## 11                              <NA>                                <NA>
## 12                              <NA>                                <NA>
## 13                              <NA>                                <NA>
## 14                         Hardcover                             English
## 15                         Paperback                             Spanish
## 16                             eBook                             English
## 17                         Paperback                              French
## 18                              <NA>                                <NA>
## 19                              <NA>                                <NA>
## 20                              <NA>                                <NA>
##    Inventory.VariationDetails.Material
## 1                                 <NA>
## 2                                 <NA>
## 3                                 <NA>
## 4                                 <NA>
## 5                                 <NA>
## 6                                 <NA>
## 7                                 <NA>
## 8                                 <NA>
## 9                                 <NA>
## 10                                <NA>
## 11                                <NA>
## 12                                <NA>
## 13                                <NA>
## 14                                <NA>
## 15                                <NA>
## 16                                <NA>
## 17                                <NA>
## 18                                <NA>
## 19                            Graphite
## 20                            Aluminum
```

XML

To import the XML data, I used the `xml2` package. I utilized the `read_xml()` function to read the XML data from the URL, which stores it as an `xml_document` in the form of `xml_nodes`. After retrieving the XML data, I converted it into a list and then transformed the list into a data frame for further analysis.

```r
# Set the CRAN mirror and install the arrow package
install.packages("arrow", repos = "https://cloud.r-project.org")
```

```
## Installing package into 'C:/Users/16462/AppData/Local/R/win-library/4.3'
## (as 'lib' is unspecified)

## package 'arrow' successfully unpacked and MD5 sums checked

## Warning: cannot remove prior installation of package 'arrow'

## Warning in file.copy(savedcopy, lib, recursive = TRUE): problem copying
## C:\Users\16462\AppData\Local\R\win-library\4.3\00LOCK\arrow\libs\x64\arrow.dll
## to C:\Users\16462\AppData\Local\R\win-library\4.3\arrow\libs\x64\arrow.dll:
## Permission denied

## Warning: restored 'arrow'

##
## The downloaded binary packages are in
##  C:\Users\16462\AppData\Local\Temp\RtmpyokIs9\downloaded_packages
```

```r
# Load the arrow package for reading and writing Parquet files
library(arrow)
```

```
## Warning: package 'arrow' was built under R version 4.3.3

##
## Attaching package: 'arrow'

## The following object is masked from 'package:lubridate':
##
##     duration

## The following object is masked from 'package:utils':
##
##     timestamp
```

```r
# URL to the Parquet file stored on GitHub
parquet_url <- "https://raw.githubusercontent.com/Shriyanshh/Week-7-Assignment/main/data.parquet"

# Download the Parquet file temporarily and save it locally as "data.parquet"
download.file(parquet_url, destfile = "data.parquet", mode = "wb")

# Read the downloaded Parquet file into R as a data frame
git_parquet <- read_parquet("data.parquet")

# Display the contents of the Parquet file, now stored as a data frame
print(git_parquet)
```

```
## # A tibble: 20 x 7
##    Category        Item_Name Item_ID Brand  Price Variation_ID Variation_Details
##    <chr>           <chr>       <dbl> <chr>  <dbl> <chr>        <chr>
##  1 Electronics     Smartpho~     101 Tech~  700.  101-A        Color: Black, St~
##  2 Electronics     Smartpho~     101 Tech~  700.  101-B        Color: White, St~
##  3 Electronics     Laptop        102 Comp~ 1100.  102-A        Color: Silver, S~
##  4 Electronics     Laptop        102 Comp~ 1100.  102-B        Color: Space Gra~
##  5 Home Appliances Refriger~     201 Home~  900.  201-A        Color: Stainless~
##  6 Home Appliances Refriger~     201 Home~  900.  201-B        Color: White, Ca~
##  7 Home Appliances Washing ~     202 Clea~  500.  202-A        Type: Front Load~
##  8 Home Appliances Washing ~     202 Clea~  500.  202-B        Type: Top Load, ~
##  9 Clothing        T-Shirt       301 Fash~   20.0 301-A        Color: Blue, Siz~
## 10 Clothing        T-Shirt       301 Fash~   20.0 301-B        Color: Red, Size~
## 11 Clothing        T-Shirt       301 Fash~   20.0 301-C        Color: Green, Si~
## 12 Clothing        Jeans         302 Deni~   50.0 302-A        Color: Dark Blue~
## 13 Clothing        Jeans         302 Deni~   50.0 302-B        Color: Light Blu~
## 14 Books           Fiction ~     401 -       15.0 401-A        Format: Hardcove~
## 15 Books           Fiction ~     401 -       15.0 401-B        Format: Paperbac~
## 16 Books           Non-Fict~     402 -       25.0 402-A        Format: eBook, L~
## 17 Books           Non-Fict~     402 -       25.0 402-B        Format: Paperbac~
## 18 Sports Equipme~ Basketba~     501 Spor~   30.0 501-A        Size: Size 7, Co~
## 19 Sports Equipme~ Tennis R~     502 Rack~   90.0 502-A        Material: Graphi~
## 20 Sports Equipme~ Tennis R~     502 Rack~   90.0 502-B        Material: Alumin~
```

Parquet

To import the Parquet data, I used the `arrow` package in R. First, I specified the URL to the Parquet file that I uploaded to GitHub. Since Parquet files cannot be read directly from a URL, I used `download.file()` to temporarily download the file to my local system. The downloaded file was saved as `"data.parquet"`. After downloading the file, I used the `read_parquet()` function from the `arrow` package to load the Parquet data into R and store it as a data frame. Finally, I printed the data frame to inspect the imported data.

```r
# Install and load the arrow package, which is used for reading and writing Parquet files
setRepositories(ind = 1) # Choose a CRAN mirror
install.packages("arrow")
```

```
## Warning: package 'arrow' is in use and will not be installed
```

```r
install.packages("arrow", repos = "https://cloud.r-project.org")
```

```
## Warning: package 'arrow' is in use and will not be installed
```

```r
library(arrow)


# URL to the Parquet file stored on GitHub
parquet_url <- "https://raw.githubusercontent.com/Shriyanshh/Week-7-Assignment/main/data.parquet"

# Download the Parquet file temporarily and save it locally as "data.parquet"
# Parquet files cannot be directly read from a URL, so we need to download it first
download.file(parquet_url, destfile = "data.parquet", mode = "wb")
```

```
## Warning in download.file(parquet_url, destfile = "data.parquet", mode = "wb"):
## URL
## https://raw.githubusercontent.com/Shriyanshh/Week-7-Assignment/main/data.parquet:
## cannot open destfile 'data.parquet', reason 'Invalid argument'
## Warning in download.file(parquet_url, destfile = "data.parquet", mode = "wb"):
## download had nonzero exit status
```

```r
# Read the downloaded Parquet file into R as a data frame
git_parquet <- read_parquet("data.parquet")

# Display the contents of the Parquet file, now stored as a data frame
print(git_parquet)
```

```
## # A tibble: 20 x 7
##    Category        Item_Name Item_ID Brand  Price Variation_ID Variation_Details
##    <chr>           <chr>       <dbl> <chr>  <dbl> <chr>        <chr>
##  1 Electronics     Smartpho~     101 Tech~  700.  101-A        Color: Black, St~
##  2 Electronics     Smartpho~     101 Tech~  700.  101-B        Color: White, St~
##  3 Electronics     Laptop        102 Comp~ 1100.  102-A        Color: Silver, S~
##  4 Electronics     Laptop        102 Comp~ 1100.  102-B        Color: Space Gra~
##  5 Home Appliances Refriger~     201 Home~  900.  201-A        Color: Stainless~
##  6 Home Appliances Refriger~     201 Home~  900.  201-B        Color: White, Ca~
##  7 Home Appliances Washing ~     202 Clea~  500.  202-A        Type: Front Load~
##  8 Home Appliances Washing ~     202 Clea~  500.  202-B        Type: Top Load, ~
##  9 Clothing        T-Shirt       301 Fash~   20.0 301-A        Color: Blue, Siz~
## 10 Clothing        T-Shirt       301 Fash~   20.0 301-B        Color: Red, Size~
## 11 Clothing        T-Shirt       301 Fash~   20.0 301-C        Color: Green, Si~
## 12 Clothing        Jeans         302 Deni~   50.0 302-A        Color: Dark Blue~
## 13 Clothing        Jeans         302 Deni~   50.0 302-B        Color: Light Blu~
## 14 Books           Fiction ~     401 -       15.0 401-A        Format: Hardcove~
## 15 Books           Fiction ~     401 -       15.0 401-B        Format: Paperbac~
## 16 Books           Non-Fict~     402 -       25.0 402-A        Format: eBook, L~
## 17 Books           Non-Fict~     402 -       25.0 402-B        Format: Paperbac~
## 18 Sports Equipme~ Basketba~     501 Spor~   30.0 501-A        Size: Size 7, Co~
## 19 Sports Equipme~ Tennis R~     502 Rack~   90.0 502-A        Material: Graphi~
## 20 Sports Equipme~ Tennis R~     502 Rack~   90.0 502-B        Material: Alumin~
```

Pros and Cons:

Each format has its own advantages and disadvantages depending on how the data is stored and analyzed. Below is an overview of each format along with its pros and cons.

Data Formats

1. JSON (JavaScript Object Notation)

Pros:
- Simple and human-readable format. - Supports nested structures, which makes it versatile. - Commonly used in APIs and web development for data exchange.

Cons:
- May not be ideal for very large datasets. - Slower than binary formats like Parquet for large-scale data processing.

2. HTML (HyperText Markup Language)

Pros:
- Easily viewable in web browsers, making it accessible for visual inspection. - Great for displaying tabular data on web pages.

Cons:
- Not optimized for analytical or large-scale data manipulation. - Requires parsing when converting to data frames in R.

3. XML (eXtensible Markup Language)

Pros:
- Structured and self-descriptive, ideal for storing hierarchical information. - Used widely in web services for data interchange.

Cons:
- More verbose than JSON and less efficient for large datasets. - Parsing XML can be slower and more complex.

4. Parquet

Pros:
- Efficient columnar storage format that is excellent for big data analytics. - Supports compression and optimized for queries, reducing both storage and processing time.

Cons:
- Not human-readable and requires specialized tools like `arrow` to access and manipulate. - Primarily useful for large datasets rather than small data exchange.

Conclusion

In this assignment, we explored various file formats—JSON, HTML, XML, and Parquet—by creating and importing data tables for each format into R. Through this process, we gained practical experience in handling different data formats, uploading files to GitHub, and utilizing R packages to read and convert the data into data frames for analysis.

For **JSON**, we utilized the `jsonlite` and `httr` packages to retrieve the file from GitHub, parse it into a list, and convert it to a wide data frame. For **HTML**, we used the `rvest` package to extract the table data from the raw HTML content stored on GitHub and converted it into a data frame. For **XML**, the `xml2` package was used to read the XML data, convert it into a list, and further transform it into a data frame. Lastly, for **Parquet**, we leveraged the `arrow` package, downloading the file temporarily from GitHub and reading it into R using the `read_parquet()` function.

Overall, this assignment provided us with valuable insights into how different file formats can be used to store structured data and how we can efficiently work with them in R. It also reinforced our understanding of integrating data from different sources, such as GitHub, into a consistent data analysis workflow.