

IE 5390 – Assignment - 07

Name: Simran Abhay Sinha

Lab 15

This lab consists of 3 questions that have a few sub questions. You are required to create an R Notebook that answers all the questions. Add explanations and comments wherever required.

I. Use the data in the file “femaleMiceWeights.csv” taken from a study of diabetes. The body weight of mice (in grams) was measured after around two weeks on one of two diets (chow or high fat). These data are in comma separated variable (csv) format.

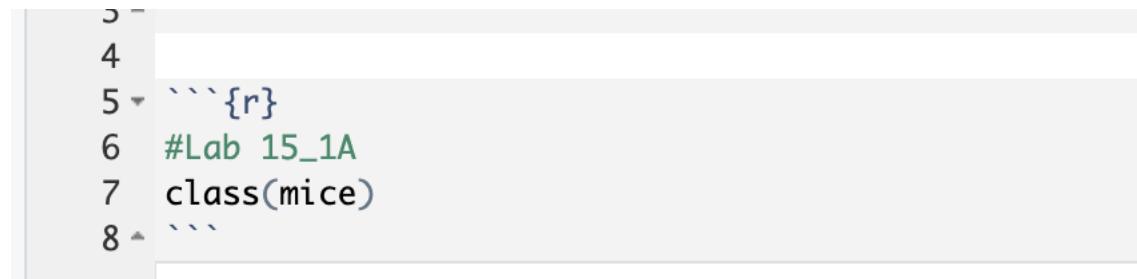
Task:

1. Load the data into RStudio and answer the following:



```
source visual
1 ~ ``{r}
2 mice <- read.csv("femaleMiceWeights.csv")
3 ~ ```
4
```

(a) Is mice an R object of class data.frame or matrix, and how can you tell?



```
5 ~ ````{r}
6 #Lab 15_1A
7 class(mice)
8 ~ ````
```

The screenshot shows the RStudio interface with the following components:

- Source View:** Displays the R script `Lab15.Rmd` containing the following code:


```

1 ~ ``{r}
2 mice <- read.csv("femaleMiceWeights.csv")
3 ~ ``
4
5 ~ ``{r}
6 #Lab 15_1A
7 class(mice)
8 ~ ``
9
10 ~ ``{r}
11 #Lab 15_1C
12 dim(mice)
13 ~ ``
14
15
      
```
- Environment View:** Shows the `mice` dataset with 24 observations and 2 variables.
- Files View:** Lists files in the current directory, including `Assignment 8_Simran Abhay Sinha.docx` and `Lab 15`.
- Console View:** Displays the R session history with the same commands and their outputs.

(b) What is the difference between a dataframe and a matrix in R?

- A matrix in R must contain elements of the same data type (homogeneous) and is primarily used for mathematical operations. A dataframe can store different data types across columns (heterogeneous) and is typically used for structured datasets with named variables.

(c) What is the dimension of the mice dataset?

```

9
10 ~ ``{r}
11 #Lab 15_1C
12 dim(mice)
13 ~ ``
      
```

The screenshot shows the RStudio interface with the following components:

- Source View:** Contains the R Markdown code for Lab15.Rmd. The code reads a CSV file "femaleMiceWeights.csv", creates a variable "mice", and prints its dimensions.
- Environment View:** Shows the variable "mice" as a data frame with 24 observations and 2 variables.
- Files View:** Displays the file structure: Home > Documents > Semester 3 > SDA > Assignments > Week 8. It lists three files: a folder named "Lab 15" and two Microsoft Word documents: "Assignment 8_Simran Abhay Sinha.docx" (162 B, modified Feb 25, 2025, 3:33 PM) and "Assignment 8_Simran Abhay Sinha.docx" (18.6 KB, modified Feb 25, 2025, 3:33 PM).
- Console View:** Shows the R session history with the same commands as the Source view.

(d) What are the units, and what are the variables?

- Variables: Diet (categorical) and Bodyweight (numerical)

Units: Bodyweight is measured in grams.

2. What is the exact name of the column containing the weights? Use the function colnames() to report code that extracts this.

```

14
15 ~ ``{r}
16 #Lab 15_2
17 colnames(mice)
18 ~ ```

```

The screenshot shows the RStudio interface with the following components:

- Source View:** Displays the R code from the previous block. The code is highlighted in green, indicating it is R code. The code reads a CSV file, prints its class, dimensions, and column names.
- Environment View:** Shows the global environment with one object: `mice` (24 obs. of 2 variables).
- Data View:** Shows the `mice` data frame with two columns: "Diet" and "Bodyweight".
- Files View:** Shows the project structure: Home > Documents > Semester 3 > SDA > Assignments > Week 8. It lists files: Assignment 8_Simran Abhay Sinha.docx (162 B, Feb 25, 2025, 3:33 PM) and Assignment 8_Simran Abhay Sinha.doc (18.6 KB, Feb 25, 2025, 3:33 PM). A folder named "Lab 15" is also present.
- Console View:** Displays the R session output, which is identical to the code in the Source view, showing the loading of the `mice` data frame and its properties.

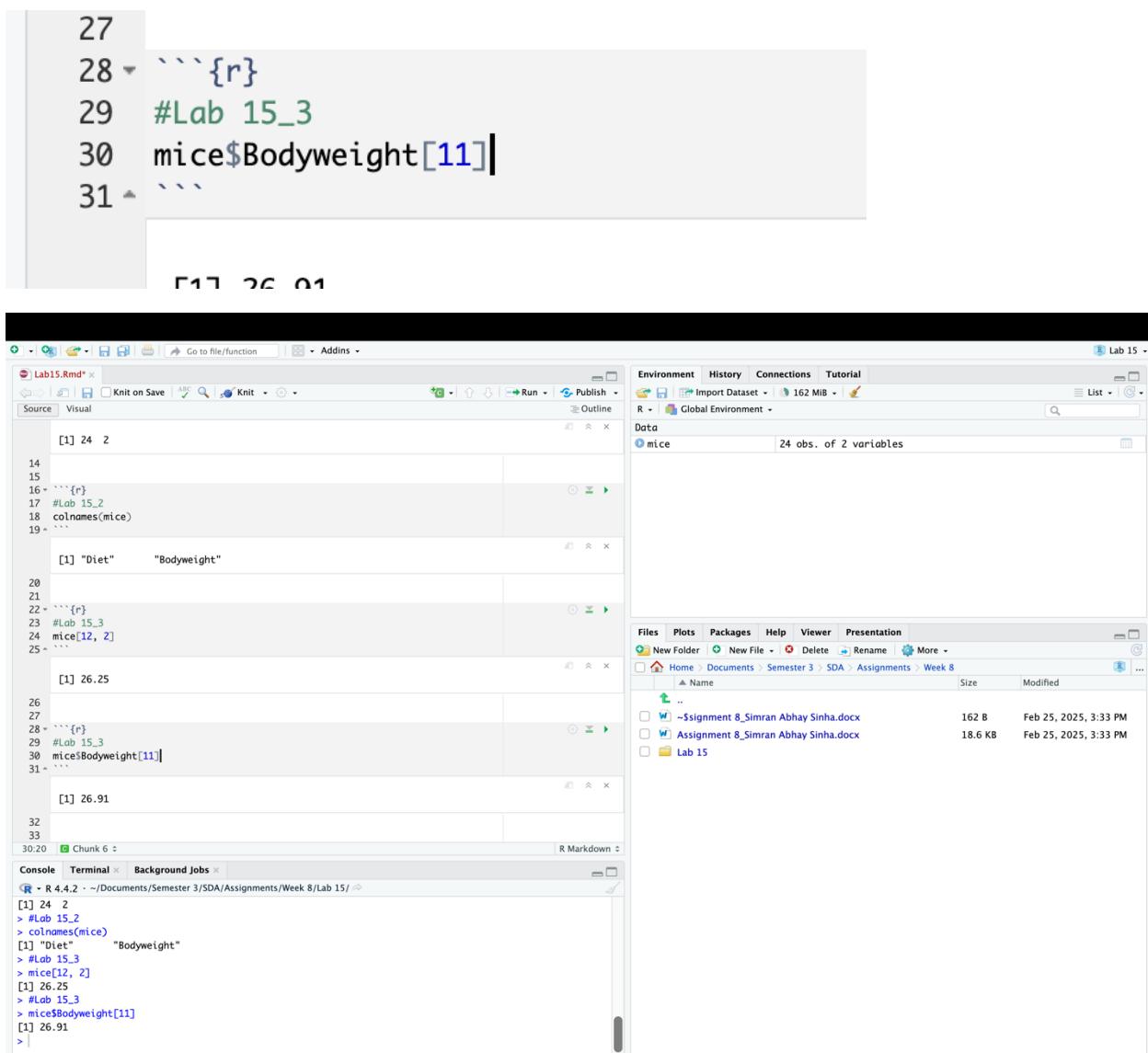
3. What is the entry in the 12th row and second column? Give code to extract it.

```
-- 
21
22 ~ ``{r}
23 #Lab 15_3
24 mice[12, 2]
25 ~ ````
```

The screenshot shows the RStudio interface with the following sections:

- Code Editor:** Displays the R code from the previous block. The line `mice[12, 2]` is highlighted.
- Environment:** Shows the variable `mice` as a data frame with 24 observations and 2 variables.
- Data View:** Provides a detailed view of the `mice` data frame, showing columns "Diet" and "Bodyweight".
- File Browser:** Shows the project structure under "Week 8" with files like "Assignment 8_Simran Abhay Sinha.docx" and "Lab 15".
- Console:** Displays the R session history, including the execution of the code and the output `26.25`.

4. Use '\$' to extract the weight column and report the weight of the mouse in the 11th row and the code used to obtain it.



The screenshot shows the RStudio interface with the following details:

- Code Editor:** Shows the R code being run. The code includes comments like `#Lab 15_3` and `#Lab 15_2` to identify specific parts of the assignment.
- Console:** Displays the output of the code execution. It shows the creation of a data frame `mice` from the `mice` dataset, and then extracts the 11th row's bodyweight, which is 26.91.
- Environment:** Shows the `mice` dataset in the global environment, which contains 24 observations and 2 variables.
- File Explorer:** Shows the project structure with files like `Assignment 8_Simran Abhay Sinha.docx` and `Lab 15`.

```

27
28 - ``{r}
29 #Lab 15_3
30 mice$Bodyweight[11]
31 ```

[1] 26.91

```

5. The length function returns the number of elements in a vector. Use length() to obtain the number of mice included in our dataset. Report your code.

```
32
33
34 - ``{r}
35 #Lab 15_3
36 length(mice$Bodyweight)
37 - ````
```

The screenshot shows the RStudio interface with the following components:

- Source:** Contains the R code from the previous block.
- Environment:** Shows the variable `mice` defined as a data frame with 24 observations and 2 variables.
- Files:** Displays the file structure: Home > Documents > Semester 3 > SDA > Assignments > Week 8. It lists three files: `~$Assignment 8_Simran Abhay Sinha.docx` (162 B, modified Feb 25, 2025, 3:33 PM), `Assignment 8_Simran Abhay Sinha.docx` (18.6 KB, modified Feb 25, 2025, 3:33 PM), and `Lab 15`.
- Console:** Shows the R session history with the command `length(mice$Bodyweight)` and its output [1] 24.

6. To create a vector with the numbers 3 to 7, we can use seq(3,7) or, because they are consecutive, 3:7. View the data and determine what rows are associated with the high fat diet (coded as hf). Use a sequence of indices to extract the vector of weights for the corresponding animals and then use the mean() function to compute their average weight. Report this mean weight and the code used to get it.

```

39
40 ``{r}
41 #Lab 15_6
42 hf_weights <- mice$Bodyweight[mice$Diet == "hf"]
43 mean(hf_weights)
44 ```

[1] 26.83417
45

```

The screenshot shows the RStudio interface with the following components:

- Source View:** Displays the R Markdown code for 'Lab15.Rmd'. It includes several code chunks (e.g., lines 26, 32, 34, 38, 41) and their corresponding results (e.g., [1] 26.25, [1] 26.91, [1] 24, [1] 26.83417).
- Environment View:** Shows the global environment with the 'mice' dataset loaded. The 'mice' dataset has 24 observations and 2 variables. The 'hf_weights' variable contains the values [1:12] 25.7 26.4 22.8 25.3 25 ...
- Files View:** Displays the file structure: Home > Documents > Semester 3 > SDA > Assignments > Week 8. It lists files: ~\$Assignment 8_Simran Abhay Sinha.docx (162 B, Feb 25, 2025, 3:33 PM), Assignment 8_Simran Abhay Sinha.docx (18.6 KB, Feb 25, 2025, 3:33 PM), and Lab 15.
- Console View:** Shows the R session history from R 4.4.2. The user runs code to create 'hf_weights' and calculate its mean, resulting in [1] 26.83417.

II. Working with Vectors in R

Task:

1. Create a vector of 100 elements such that the first 20 elements are 1, 2, ..., 20, the next 10 elements are 10, 20, 30, ..., 100, and the last 70 elements are 31, 32, ..., 100.

Hint: You will need to use the function seq().

```
45
46
47 - ``{r}
48 #Lab 15_II 1
49 # First 20 elements: 1 to 20
50 part1 <- seq(1, 20)
51
52 # Next 10 elements: 10, 20, ..., 100
53 part2 <- seq(10, 100, by=10)
54
55 # Last 70 elements: 31 to 100
56 part3 <- seq(31, 100)
57
58 # Combine all parts into one vector
59 my_vector <- c(part1, part2, part3)
60
61 - ````
```

2. Show the contents of the vector.

```
62
63 - ``{r}
64 #Lab 15_II 2
65 print(my_vector)
66 - ````
```

The screenshot shows the RStudio interface with the following components:

- Code pane (Source tab):** Contains R code for generating a vector from three parts. The code is as follows:

```

46
47 - ``{r}
48 #Lab 15_II 1
49 # First 20 elements: 1 to 20
50 part1 <- seq(1, 20)
51
52 # Next 10 elements: 10, 20, ..., 100
53 part2 <- seq(10, 100, by=10)
54
55 # Last 70 elements: 31 to 100
56 part3 <- seq(31, 100)
57
58 # Combine all parts into one vector
59 my_vector <- c(part1, part2, part3)
60
61 ``````{r}
62 #Lab 15_II 2
63 print(my_vector)
64 ``````{r}
65
66
67
68
69

```

- Environment pane:** Shows the global environment with objects like `mice` (24 obs. of 2 variables), `hf_weights` (num [1:12] 25.7 26.4 22.8 25.3 25 ...), `my_vector` (int [1:100] 1 2 3 4 5 6 7 8 9 10 ...), `part1` (int [1:20] 1 2 3 4 5 6 7 8 9 10 ...), `part2` (int [1:10] 10 20 30 40 50 60 70 80 90 100), and `part3` (int [1:70] 31 32 33 34 35 36 37 38 39 40 ...).
- Files pane:** Shows the file structure: Home > Documents > Semester 3 > SDA > Assignments > Week 8. It lists two files: `--\$assignment_8_Simran_Abhay_Sinha.docx` (162 B, Feb 25, 2025, 3:33 PM) and `Assignment_8_Simran_Abhay_Sinha.docx` (18.6 KB, Feb 25, 2025, 3:33 PM). A folder named `Lab 15` is also shown.
- Console pane:** Displays the R session output, which is identical to the code in the Source pane.

III. Cleaning and Transformation

The HERs data are a simple random sample of n=2763 observations from the HERs study. The HERs study was a randomized clinical trial of hormone therapy (estrogen plus progestin)

Task:

1. Load the hersdata.Rdata in RStudio

```

68
69 ~ ``{r}
70 # Lab 15_III
71 load("hersdata.Rdata") # Ensure the file is in your working directory
72 ~ ```
73

```

The screenshot shows the RStudio interface with the following sections:

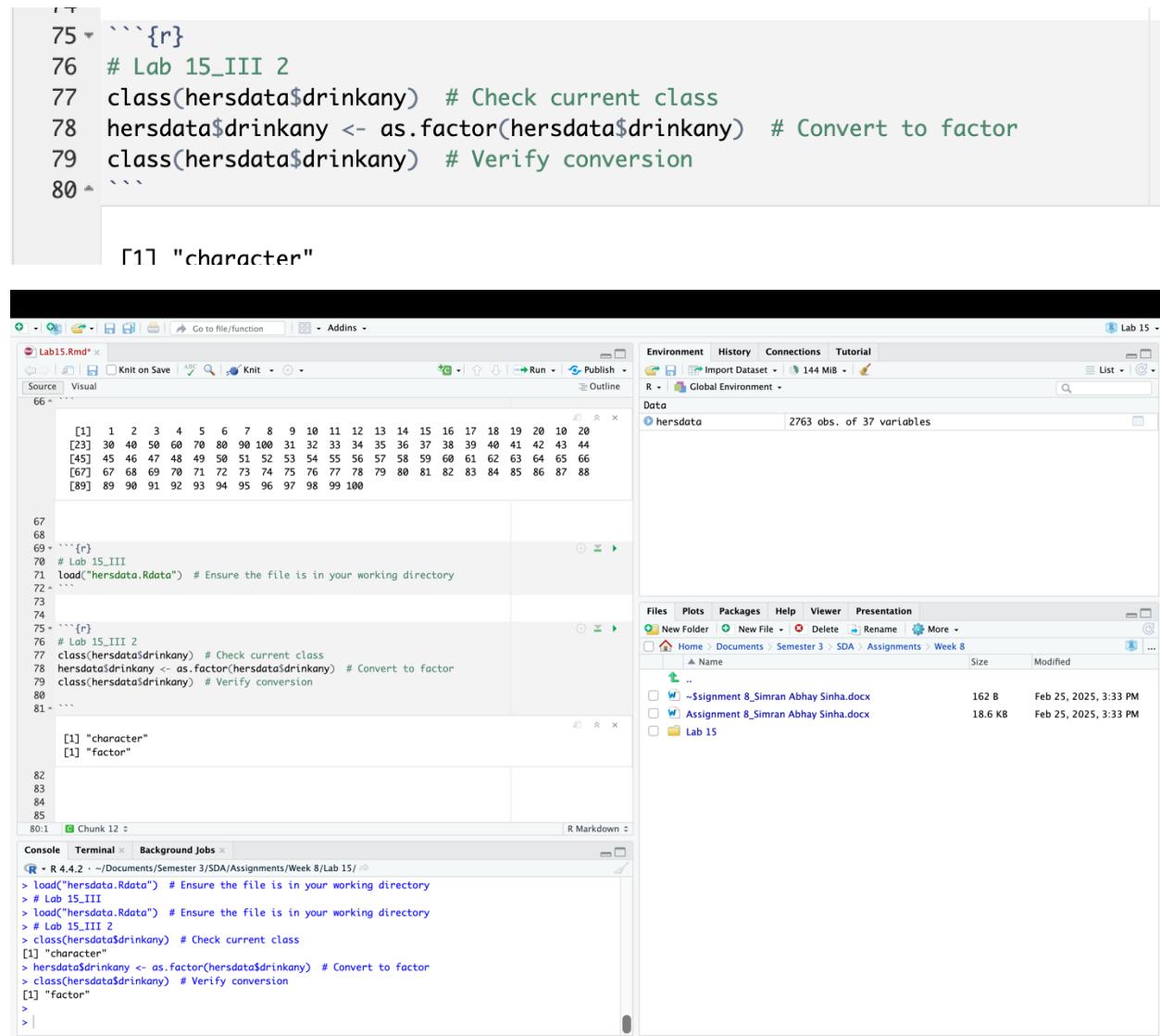
- Code Editor:** Displays the R code for loading the 'hersdata.Rdata' file.
- Environment:** Shows the 'hersdata' dataset with 2763 observations and 37 variables.
- Files:** Lists files in the current directory, including 'Assignment 8_Simran Abhay Sinha.docx' and 'Assignment 8_Simran Abhay Sinha.docx' (modified).
- Console:** Displays the output of the R code, showing the contents of the 'my_vector' variable.

```

[1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 10 20
[23] 30 40 50 60 70 80 90 100 31 32 33 34 35 36 37 38 39 40 41 42 43 44
[45] 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66
[67] 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88
[89] 89 90 91 92 93 94 95 96 97 98 99 100

```

2. Issue the command `class(hersdata$drinkany)`. You should see that it is a character variable. Convert this variable to factor.



The screenshot shows the RStudio interface with the following components:

- Code Editor:** Displays the R code for Lab 15_III 2. It includes a snippet of data from the hersdata dataset and the output of the `class()` function showing the variable is a character.
- Data View:** Shows the `hersdata` dataset with 2763 observations and 37 variables.
- File Explorer:** Shows the file structure for Week 8, including files like `Assignment 8_Simran Abhay Sinha.docx` and `Lab 15`.
- Console:** Displays the R session history, including the loading of the dataset and the execution of the code to check the class and convert it to a factor.

```

75 - ``{r}
76 # Lab 15_III 2
77 class(hersdata$drinkany) # Check current class
78 hersdata$drinkany <- as.factor(hersdata$drinkany) # Convert to factor
79 class(hersdata$drinkany) # Verify conversion
80 ```

```

[1] "character"

```

[1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 10 20
[23] 30 40 50 60 70 80 90 100 31 32 33 34 35 36 37 38 39 40 41 42 43 44
[45] 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66
[67] 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88
[89] 89 90 91 92 93 94 95 96 97 98 99 100

[1] "character"
[1] "factor"

```

```

> load("hersdata.Rdata") # Ensure the file is in your working directory
> # Lab 15_III 2
> load("hersdata.Rdata") # Ensure the file is in your working directory
> # Lab 15_III 2
> class(hersdata$drinkany) # Check current class
[1] "character"
> hersdata$drinkany <- as.factor(hersdata$drinkany) # Convert to factor
> class(hersdata$drinkany) # Verify conversion
[1] "factor"
>

```

3. Install the `summarytools` package and load the library. Next, produce a one-way frequency table of the `drinkany` variable. How many missing values are there?

```

81
82
83 - ``{r}
84 # Lab 15_III 3
85 install.packages("summarytools") # Install package |
86 library(summarytools) # Load package
87
88 # Create a one-way frequency table
89 freq(hersdata$drinkany)
90
91 # Count missing values
92 sum(is.na(hersdata$drinkany))
93

```

The RStudio interface shows the following details:

- Source View:** Displays the R code from the 'Lab 15.Rmd' file.
- Environment View:** Shows the 'hersdata' dataset with 2763 observations and 37 variables.
- Packages View:** Shows the 'System Library' with various packages listed, such as 'base', 'grid', 'lattice', etc., along with their versions and descriptions.
- Console View:** Displays the output of the R code, including the download of the 'summarytools' package, the frequency table for 'drinkany', and the count of missing values.

```

trying URL 'https://cran.rstudio.com/bin/macosx/big-sur-arm64/contrib/4.4/summarytools_1.1.0.tgz'
Content type 'application/x-gzip' length 1671939 bytes (1.6 MB)
downloaded 1.6 MB

The downloaded binary packages are in
  /var/folders/9m/f5j18xnd4ls2_3yknkt1_rpm000gn/T//RtmpXMdvnR/downloaded_packages
Frequencies
hersdata$drinkany
Type: Factor

      Freq % Valid % Valid Cum. % Total % Total Cum.
no     1680 60.848   60.848 60.803   60.803
yes    1081 39.152   100.000 39.124   99.928
<NA>      2          0.072   100.000
Total   2763 100.000 100.000 100.000 100.000

```

```

> # Count missing values
> sum(is.na(hersdata$drinkany))
[1] 2
>

```

4. Create a 0/1 numeric variable drinkany01 that is coded as follows: = 0 if drinkany == "no"
 1 if drinkany == "yes" and NA if drinkany == NA

```

95
96 ````{r}
97 # Lab 15_ III 4
98 hersdata$drinkany01 <- ifelse(hersdata$drinkany == "no", 0,
99                               ifelse(hersdata$drinkany == "yes", 1, NA))
100
101 # Check the conversion
102 table(hersdata$drinkany, hersdata$drinkany01, useNA = "always")
103 ````
```

The screenshot shows the RStudio interface with the following components:

- Code Editor:** Displays the R code from the previous block.
- Environment Pane:** Shows the dataset `hersdata` with 2763 observations and 38 variables.
- Library Pane:** Shows the `System Library` with a list of installed R packages and their versions.
- Console:** Displays the R session history, including the execution of the code and the resulting table output.

5. Create a subset of hersdata called mytiny as follows:

- Include only the following variables: HT, LDL, and SBP
- Include only the observations with: weight > 125

```
106
107 ``{r}
108 # Lab 15_III 5
109 library(tidyverse) # Load package
110
111 # Now try the corrected approach
112 mytiny <- hersdata %>%
113   # First filter for observations where weight > 125
114   filter(weight > 125) %>%
115   # Then select only the specified variables
116   select(HT, LDL, SBP)
117
118 # Lab 15_III 6 |
119 print(mytiny)
120 ````
```

6. Print the mytiny data.

```
106
107 ``{r}
108 # Lab 15_III 5
109 library(tidyverse) # Load package
110
111 # Now try the corrected approach
112 mytiny <- hersdata %>%
113   # First filter for observations where weight > 125
114   filter(weight > 125) %>%
115   # Then select only the specified variables
116   select(HT, LDL, SBP)
117
118 # Lab 15_III 6 |
119 print(mytiny)
120 ````
```

The screenshot shows the RStudio interface with the following components:

- Source View:** Displays the R code in `Lab15.Rmd`. The code installs `tidyverse`, loads `herdsdata`, filters for weight > 125, and selects variables HT, LDL, and SBP.
- Environment View:** Shows the global environment with two datasets: `herdsdata` (2763 obs. of 38 variables) and `mytiny` (5 obs. of 3 variables).
- Plots View:** Displays a data frame titled "A tibble: 5 x 3" with columns HT, LDL, and SBP. The data is as follows:

| HT | LDL | SBP |
|-----------------|-------|-----|
| hormone therapy | 122.2 | 129 |
| placebo | 204.6 | 133 |
| placebo | 161.2 | 112 |
| placebo | 137.0 | 130 |
| placebo | 148.4 | 139 |

- Console View:** Shows the R session history with the same commands run at the prompt.
- Packages View:** Shows the installed packages in the current session, including `tidyverse` (version 2.0.0), `broom`, `rematch2`, and `rlang`.

IV. Gen AI Prompt Engineering:

You are a data analyst working for an e-commerce company. Your manager provides you with a dataset containing customer transaction records. Using sample_transactions.xlsx, your task is to analyze the dataset and generate insights by performing the following tasks:

Bot Used: Claudio.AI

Prompt:

This is the given dataset
Analyze the dataset and generate insights by performing the following tasks

1. Data Structures:
 - a. Store the dataset using an appropriate data structure (list, dictionary, DataFrame, etc.).
 - b. Write a function to find the top 3 most frequently purchased product categories.
1. Control Structures: a. Write a loop to calculate the total revenue generated from each product category. b. Use conditional statements to analyze the most preferred payment method.

Give me R code

Edit | < 2 / 2 >

Gen AI:

I'll analyze the dataset and generate insights using R. Let me write code for each of the requested tasks.

R Analysis of Transaction Data
Click to open code

Here's what the code does for each task:

1. Data Structures

- **a. Storing the dataset:** I've used R's `read.csv()` function to load the data into a data frame, which is R's standard structure for tabular data.
- **b. Top 3 product categories:** The `find_top_categories()` function counts occurrences of each product category and returns the top 3 most frequently purchased categories.

2. Control Structures

- **a. Revenue by category:** The `calculate_revenue_by_category()` function uses a for-loop to iterate through each unique product category, calculating the total revenue ($\text{Quantity} \times \text{Price}$) for each.
- **b. Payment method analysis:** The `analyze_payment_methods()` function uses conditional statements to provide insights about the most preferred payment method.

Results you'll get from running this code:

```
# Create a data frame from the CSV string
transactions <- read.csv(text = transactions_data, stringsAsFactors = FALSE)

# Display the data frame structure
str(transactions)

# b. Write a function to find the top 3 most frequently purchased product categories
find_top_categories <- function(data) {
  # Calculate the frequency of each product category
  category_counts <- table(data$Product.Category)

  # Sort in descending order
  sorted_categories <- sort(category_counts, decreasing = TRUE)

  # Return the top 3 (or less if there are fewer categories)
  top_n <- min(3, length(sorted_categories))
  return(head(sorted_categories, top_n))
}

# Call the function to find top 3 categories
top_categories <- find_top_categories(transactions)
print("Top 3 most frequently purchased product categories:")
print(top_categories)

# 2. Control Structures

# a. Calculate the total revenue from each product category using a loop
calculate_revenue_by_category <- function(data) {
  # Get unique product categories
  unique_categories <- unique(data$Product.Category)
  total_revenue <- 0

  for (category in unique_categories) {
    quantity <- sum(data$Quantity[data$Product.Category == category])
    price <- mean(data$Price[data$Product.Category == category])
    revenue <- quantity * price
    total_revenue <- total_revenue + revenue
  }

  return(total_revenue)
}

# Call the function to calculate total revenue
total_revenue <- calculate_revenue_by_category(transactions)
print("Total revenue generated from all product categories: $", total_revenue)
```

You are out of free messages until 9:00 PM

Reply to Claude... Subscribe

Claude 3.7 Sonnet Choose style

Last edited 9 minutes ago Edit Publish

Claude Analyzing Product Sales and Payment Trends

I'll analyze the dataset and generate insights using R. Let me write code for each of the requested tasks.

`</> R Analysis of Transaction Data`

Click to open code

Here's what the code does for each task:

1. Data Structures

- a. **Storing the dataset:** I've used R's `read.csv()` function to load the data into a data frame, which is R's standard structure for tabular data.
- b. **Top 3 product categories:** The `find_top_categories()` function counts occurrences of each product category and returns the top 3 most frequently purchased categories.

2. Control Structures

- a. **Revenue by category:** The `calculate_revenue_by_category()` function uses a for-loop to iterate through each unique product category, calculating the total revenue ($\text{Quantity} \times \text{Price}$) for each.
- b. **Payment method analysis:** The `analyze_payment_methods()` function uses conditional statements to provide insights about the most preferred payment method.

Results you'll get from running this code:

You are out of free messages until 9:00 PM

Reply to Claude... Subscribe

Claude 3.7 Sonnet Choose style

`revenue_by_category <- list()`

```
# Loop through each category
for (category in categories) {
  # Filter transactions for the current category
  category_data <- data[data$Product.Category == category, ]

  # Calculate revenue for each transaction (Quantity * Price)
  transaction_revenue <- category_data$Quantity.Purchased * category_data$Price.per.Unit

  # Sum the revenue for the category
  total_revenue <- sum(transaction_revenue)

  # Store in the list
  revenue_by_category[[category]] <- total_revenue
}

return(revenue_by_category)
```

Call the function to calculate revenue by category

```
revenue_by_category <- calculate_revenue_by_category(transactions)
print("Total revenue generated from each product category:")
print(revenue_by_category)
```

b. Analyze the most preferred payment method using conditional statements

```
analyze_payment_methods <- function(data) {
  # Calculate the frequency of each payment method
  payment_counts <- table(data$Payment.Method)
```

Last edited 9 minutes ago

Share Publish

Claude Analyzing Product Sales and Payment Trends

I'll analyze the dataset and generate insights using R. Let me write code for each of the requested tasks.

`</> R Analysis of Transaction Data`

Click to open code

Here's what the code does for each task:

1. Data Structures

- a. **Storing the dataset:** I've used R's `read.csv()` function to load the data into a data frame, which is R's standard structure for tabular data.
- b. **Top 3 product categories:** The `find_top_categories()` function counts occurrences of each product category and returns the top 3 most frequently purchased categories.

2. Control Structures

- a. **Revenue by category:** The `calculate_revenue_by_category()` function uses a for-loop to iterate through each unique product category, calculating the total revenue ($\text{Quantity} \times \text{Price}$) for each.
- b. **Payment method analysis:** The `analyze_payment_methods()` function uses conditional statements to provide insights about the most preferred payment method.

Results you'll get from running this code:

You are out of free messages until 9:00 PM

Reply to Claude... Subscribe

Claude 3.7 Sonnet Choose style

```
# else if (most_preferred == "Debit Card") {
  result$analysis <- "Debit Card is the most popular payment method, suggesting customers prefer convenience over security or cost."}
else if (most_preferred == "Crypto") {
  result$analysis <- "Cryptocurrency is the most popular payment method, indicating a younger demographic or a niche market."}
else {
  result$analysis <- paste(most_preferred, "is the most popular payment method.")}
```

return(result)

Call the function to analyze payment methods

```
payment_analysis <- analyze_payment_methods(transactions)
print("Payment method analysis:")
print(payment_analysis$counts)
print(paste("Most preferred payment method:", payment_analysis$most_preferred))
print(payment_analysis$analysis)
```

Additional summary statistics for the dataset

```
print("Summary of transaction data:")
summary_stats <- list()
total_transactions = nrow(transactions),
total_revenue = sum(transactions$Quantity.Purchased * transactions$Price.per.Unit)
average_order_value = mean(transactions$Quantity.Purchased * transactions$Price.per.Unit)
total_items_sold = sum(transactions$Quantity.Purchased)
)
print(summary_stats)
```

Last edited 9 minutes ago

Share Publish

1. Data Structures:

- a. Store the dataset using an appropriate data structure (list, dictionary, DataFrame, etc.).

The screenshot shows the RStudio interface with the following components:

- Left Panel (Code Editor):** An R script titled "Lab15.Rmd". The code includes comments like "# Assuming the file is named 'sample_transactions.csv' and is in your working directory" and "# Display the data frame structure". It also contains a `str(transactions)` command which outputs the structure of the data frame.
- Top Bar:** Contains icons for file operations (New, Open, Save, Print, etc.), Go to file/function, Addins, and Lab 15.
- Environment Tab:** Shows the global environment with objects like `payment_analysis` (List of 3), `revenue_by_category` (List of 4), `summary_stats` (List of 4), and `transactions` (10 obs. of 6 variables). It also lists functions: `analyze_payment_metho...`, `calculate_revenue_by...`, and `find_top_categories`.
- File Browser:** A sidebar showing the project structure under "Lab 15". It includes files like ".RData", ".RHistory", "Basic.Rmd", "femaleMiceWeights.csv", "hersdata.Rdata", "Lab 15.kproj", "Lab15.Rmd", and "sample_transactions.csv".
- Bottom Panel (Console):** Displays R session output:
 - Total_revenue: [1] 2749.76
 - Average_order_value: [1] 274.976
 - Total_items_sold: [1] 24

- b. Write a function to find the top 3 most frequently purchased product categories.

The screenshot shows the RStudio interface with the following sections:

- Script Editor (left):** Displays the `Lab15.Rmd` file containing R code for finding top categories and calculating revenue.
- Environment Browser (top right):** Shows objects like `payment_analysis`, `revenue_by_category`, `summary_stats`, and `transactions`.
- File Browser (right):** Lists files in the project directory, including `.RData`, `.History`, `Basic.Rmd`, `femaleMiceWeights.csv`, `herdsdata.Rdata`, `Lab 15.Rproj`, `Lab15.Rmd`, and `sample_transactions.csv`.
- Console (bottom left):** Displays the output of the R code, including the top categories and total revenue.

2. Control Structures:

- a. Write a loop to calculate the total revenue generated from each product category.

The screenshot shows the RStudio interface with the following details:

- Source View:** Contains R code to calculate revenue by category. The code uses a loop to iterate through categories, calculates total revenue for each, and prints the results.
- Console View:** Shows the output of the R code, displaying the total revenue for four categories: Electronics (\$2099.92), Clothing (\$339.91), Home & Kitchen (\$259.96), and Beauty (\$49.97).
- Environment View:** Shows the global environment with objects like payment_analysis, revenue_by_category, summary_stats, and transactions.
- Files View:** Shows the project structure under Lab 15, including files like .RData, .Rhistory, Basic.Rmd, femaleMiceWeights.csv, hersdata.Rdata, Lab 15.Rproj, Lab15.Rmd, and sample_transactions.csv.

- b. Use conditional statements to analyze the most preferred payment method.

The screenshot shows the RStudio interface with the following details:

- Source View:** Contains R code for payment method analysis. It calculates total transactions, average order value, and total items sold, then prints summary statistics.
- Console View:** Shows the output of the R code, including a summary of transaction data and the most preferred payment method (Credit Card).
- Environment View:** Shows the global environment with objects like payment_analysis, revenue_by_category, summary_stats, and transactions.
- Files View:** Shows the project structure under Lab 15, including files like .RData, .Rhistory, Basic.Rmd, femaleMiceWeights.csv, hersdata.Rdata, Lab 15.Rproj, Lab15.Rmd, and sample_transactions.csv.

Lab 16

I. Use the countries.csv data. Create a Shiny App with the following:

1. Read the countries.csv file, save it in a data frame called “countries”



```

1 # Lab 16_Question I
2
3 # Read the CSV file
4 df <- read.csv("countries.csv")
5

```

2. Use fluidPage() for the UI, with the Title Panel “Country Data”.



```

6 # Define UI
7 ui <- fluidPage(
8   titlePanel("Country Data"),
9   sidebarLayout(
10     sidebarPanel(
11       selectInput(

```

3. Next, change the input type to selectInput and name the input “country”, make the label say “Countries”, and include the options to select by adding c(countries\$Country) because in the csv file, the name of the countries are under the Country column. Preselect China, by adding selected = “China”, and we won’t allow multiple selections by adding multiple = FALSE.



```

6 # Define UI
7 ui <- fluidPage(
8   titlePanel("Country Data"),
9   sidebarLayout(
10     sidebarPanel(
11       selectInput(
12         inputId = "country",
13         label = "Countries",
14         choices = df$Country,
15         selected = "China",
16         multiple = FALSE
17       )
18     ),
19     mainPanel(
20       plotOutput("countryPlot", width = "800px", height = "600px")
21     )
22   )
23 )
24

```

4. Create a plot (mainPanel) where the x axis is the population of the country and the y axis is the per capita GDP of the country. The title of the plot should be “Population and GDP”

```

18   ),
19   mainPanel(
20     plotOutput("countryPlot", width = "800px", height = "600px")
21   )
22 )
23 )
24
25 # Define server
26 server <- function(input, output) {
27   output$countryPlot <- renderPlot({
28     plot(
29       df$Population, df$GDP_per_capita,
30       log = "xy", # Use logarithmic scale
31       col = ifelse(df$Country == input$country, "red", "black"),
32       pch = 16,
33       cex = 1.2,
34       xlab = "Population",
35       ylab = "Per Capita GDP",
36       main = "Population and GDP"
37     )
38     grid()
39   })
40 }
41

```

Run the Shiny application

5. Use this function to highlight the country selected:

```
col=ifelse(countries$Country==country, "red", "black")
```

```

24
25 # Define server
26 server <- function(input, output) {
27   output$countryPlot <- renderPlot({
28     plot(
29       df$Population, df$GDP_per_capita,
30       log = "xy", # Use logarithmic scale
31       col = ifelse(df$Country == input$country, "red", "black"),
32       pch = 16,
33       cex = 1.2,
34       xlab = "Population",
35       ylab = "Per Capita GDP",
36       main = "Population and GDP"
37     )
38     grid()
39   })
40 }
41
42

```

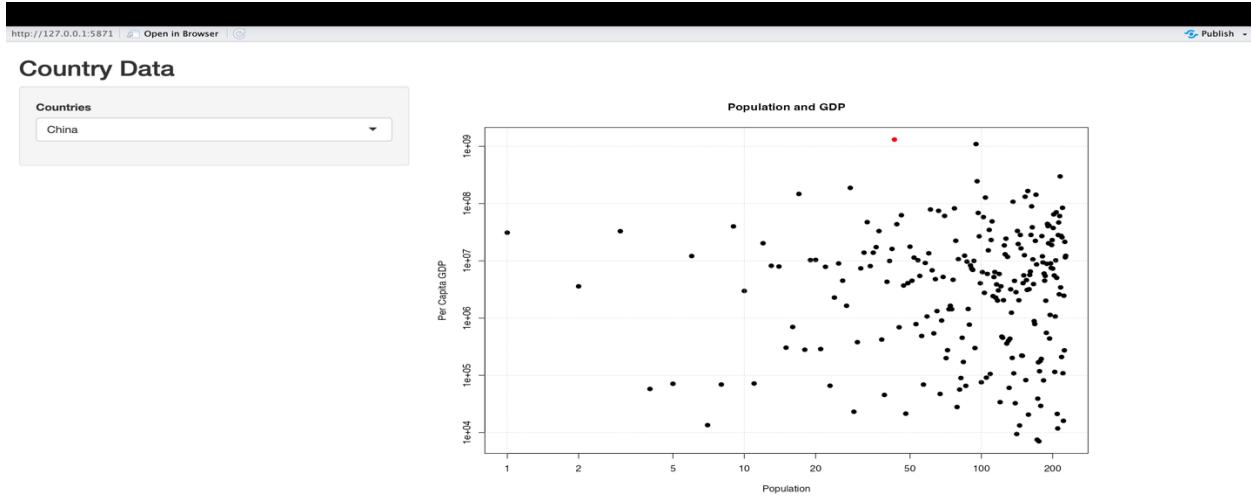
Run the Shiny application

The screenshot shows the RStudio interface with the following components:

- Code Editor:** Displays the R script for "Lab 16_Question 1". The script defines a UI using `fluidPage` and `mainPanel` to create a dropdown menu for countries, and a plot output for a scatter plot.
- Environment View:** Shows the global environment with a dataset named "countries" containing 227 observations and 20 variables.
- File Explorer:** Shows the file structure for "Lab 15", including files like ".Data", "Rhistory", "Basic.Rmd", "femaleMiceWeights.csv", "hersdata.Rdata", "Lab 15.Rproj", "Lab15.Rmd", "Lab16_Week8", and "sample_transactions.csv".
- Console:** Displays the command `runApp("Lab16_Week8/Lab16_1.R")` being run, followed by multiple "Listening on http://127.0.0.1:5871" messages.

The screenshot shows the RStudio interface with the following components:

- Code Editor:** Displays the modified R script for "Lab 16_Question 1". The changes include adding a logarithmic scale to the plot and changing the main title to "Population and GDP".
- Environment View:** Shows the global environment with a dataset named "countries" containing 227 observations and 20 variables.
- File Explorer:** Shows the file structure for "Lab 15", including files like ".Data", "Rhistory", "Basic.Rmd", "femaleMiceWeights.csv", "hersdata.Rdata", "Lab 15.Rproj", "Lab15.Rmd", "Lab16_Week8", and "sample_transactions.csv".
- Console:** Displays the command `runApp("Lab16_Week8/Lab16_1.R")` being run, followed by multiple "Listening on http://127.0.0.1:5871" messages.



II. Use the france.csv data. The goal is to create a dashboard using Flexdashboard

a) Title and Output Format

```

1 ---  

2 title: "France Economic Report"  

3 output:  

4   flexdashboard::flex_dashboard:  

5     orientation: rows  

6     social: menu  

7     theme: lumen  

8     vertical_layout: fill  

9 ---
```

b) Required Packages and Libraries

```

10
11 ````{r setup, include=FALSE}
12 library(flexdashboard)
13 library(ggplot2)
14 library(readr)
15 library(WDI)
16 library(dplyr)
17 library(stats)
18 library(visreg)
19 library(RColorBrewer)
20 library(stargazer)
21 library(kableExtra)
22 library(maps)
23 ````
```

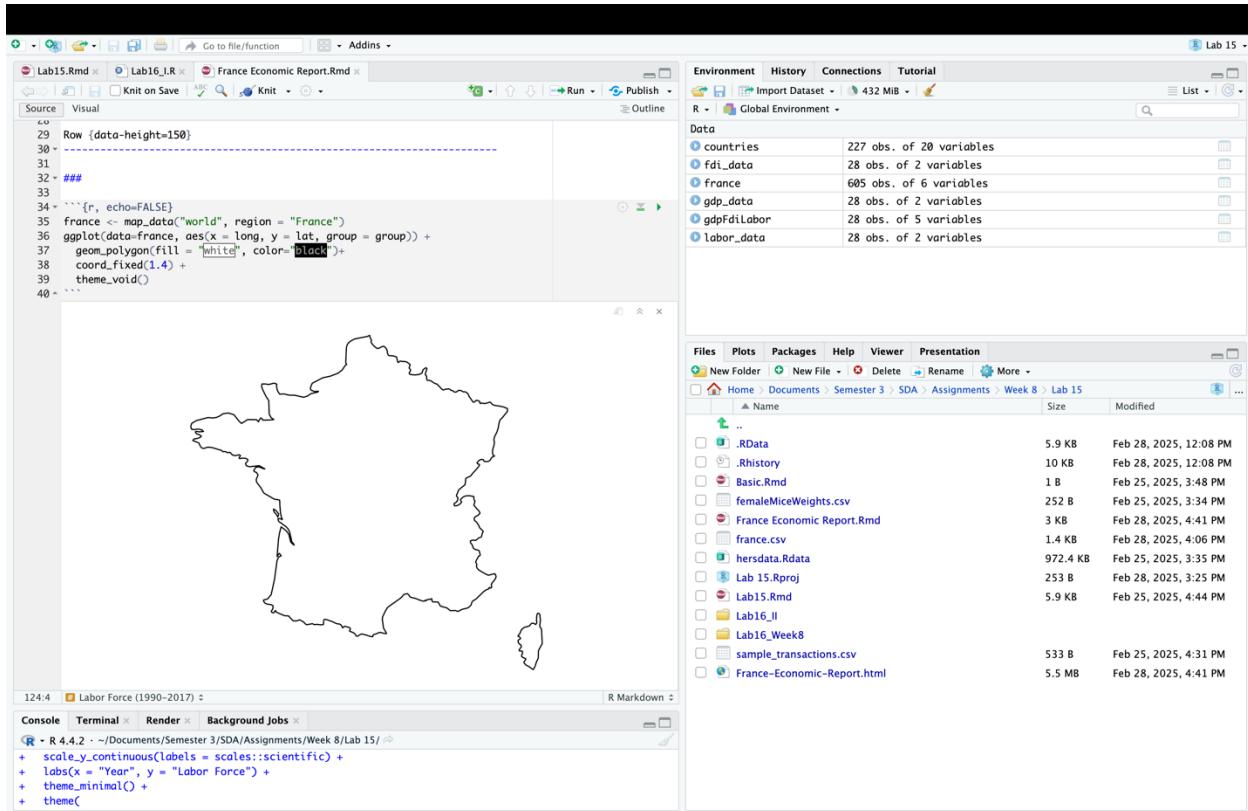
c) Loading the data

```

25 ````{r, echo=FALSE}
26 gdpFdLabor <- read_csv("france.csv", show_col_types = FALSE)
27 ``
28

```

d) First row with 4 columns showing map of France, GDP Numbers, Foreign Direct Investment and Labor Force in Millions respectively.



The screenshot shows the RStudio interface with the following details:

- Left Panel (Code Editor):** Displays the R code for the `Lab16_I.R` file. The code includes a `valueBox` call with the value `2860`.
- Right Panel (Environment View):** Shows the global environment with objects like `countries`, `fdi_data`, `france`, `gdp_data`, `gdpfdlabor`, and `labor_data`.
- Bottom Panel (File Explorer):** Shows the file structure for Week 8, Lab 15, including files like `.RData`, `.Rhistory`, `Basic.Rmd`, `femaleMiceWeights.csv`, `france.Rmd`, `france.csv`, `hersdata.Rdata`, `Lab 15.Rproj`, `Lab15.Rmd`, `Lab16_II`, `Lab16_Week8`, `sample_transactions.csv`, and `France-Economic-Report.html`.

The screenshot shows the RStudio interface with the following details:

- Left Panel (Code Editor):** Displays the R code for the `Lab16_I.R` file. The code includes a `valueBox` call with the value `37`.
- Right Panel (Environment View):** Shows the global environment with objects like `countries`, `fdi_data`, `france`, `gdp_data`, `gdpfdlabor`, and `labor_data`.
- Bottom Panel (File Explorer):** Shows the file structure for Week 8, Lab 15, including files like `.RData`, `.Rhistory`, `Basic.Rmd`, `femaleMiceWeights.csv`, `france.Rmd`, `france.csv`, `hersdata.Rdata`, `Lab 15.Rproj`, `Lab15.Rmd`, `Lab16_II`, `Lab16_Week8`, `sample_transactions.csv`, and `France-Economic-Report.html`.

Screenshot of RStudio showing the environment for Lab 15.

Source View:

```

54 ## Labor Force (Millions) in 2017
55
56 ````{r}
57 valueBox(30, icon = "fa-user-tie", color = "primary")
58 ````
```

Console View:

```

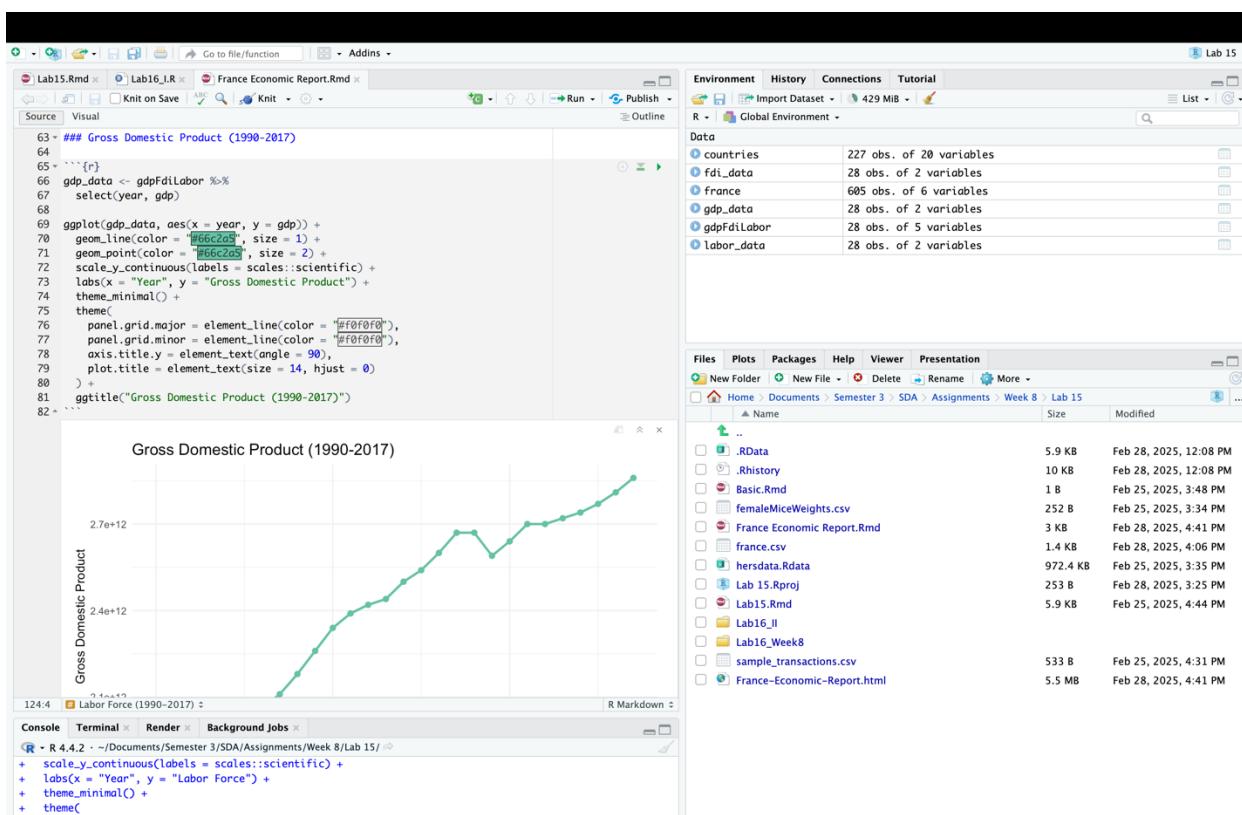
R > R 4.4.2 - ~/Documents/Semester 3/SDA/Assignments/Week 8/Lab 15/ 
+ scale_y_continuous(labels = scales::scientific) +
+ labs(x = "Year", y = "Labor Force") +
+ theme_minimal() +
+ theme(
```

Environment View:

- countries: 227 obs. of 20 variables
- fdi_data: 28 obs. of 2 variables
- france: 605 obs. of 6 variables
- gdp_data: 28 obs. of 2 variables
- gdpfdilabor: 28 obs. of 5 variables
- labor_data: 28 obs. of 2 variables

Files View:

| Name | Size | Modified |
|-----------------------------|----------|------------------------|
| .. | 5.9 KB | Feb 28, 2025, 12:08 PM |
| .RData | 10 KB | Feb 28, 2025, 12:08 PM |
| .Rhistory | 1 B | Feb 25, 2025, 3:48 PM |
| Basic.Rmd | 252 B | Feb 25, 2025, 3:34 PM |
| femaleMiceWeights.csv | 3 KB | Feb 28, 2025, 4:41 PM |
| France Economic Report.Rmd | 1.4 KB | Feb 28, 2025, 4:06 PM |
| france.csv | 972.4 KB | Feb 25, 2025, 3:35 PM |
| hersdata.Rdata | 253 B | Feb 28, 2025, 3:25 PM |
| Lab 15.Rproj | 5.9 KB | Feb 25, 2025, 4:44 PM |
| Lab15.Rmd | 533 B | Feb 25, 2025, 4:31 PM |
| Lab16_II | 5.5 MB | Feb 28, 2025, 4:41 PM |
| Lab16_Week8 | | |
| sample_transactions.csv | | |
| France-Economic-Report.html | | |



Two screenshots of RStudio environments showing the analysis of France's economic report.

Screenshot 1: Foreign Direct Investment (1990-2017)

```

85
86 - ``{r}
87   fdi_data <- gdpFdilabor %>%
88   select(year, fdi)
89
90   ggplot(fdi_data, aes(x = year, y = fdi)) +
91     geom_line(color = "#80b1d3", size = 1) +
92     geom_point(color = "#80b1d3", size = 2) +
93     scale_y_continuous(labels = scales::scientific) +
94     labs(x = "Year", y = "Foreign Direct Investments") +
95     theme(
96       panel.grid.major = element_line(color = "#f0f0f0"),
97       panel.grid.minor = element_line(color = "#f0f0f0"),
98       axis.title.y = element_text(angle = 90),
99       plot.title = element_text(size = 14, hjust = 0)
100    ) +
101  ggtitle("Foreign Direct Investment (1990-2017)")
103

```

Screenshot 2: Labor Force (1990-2017)

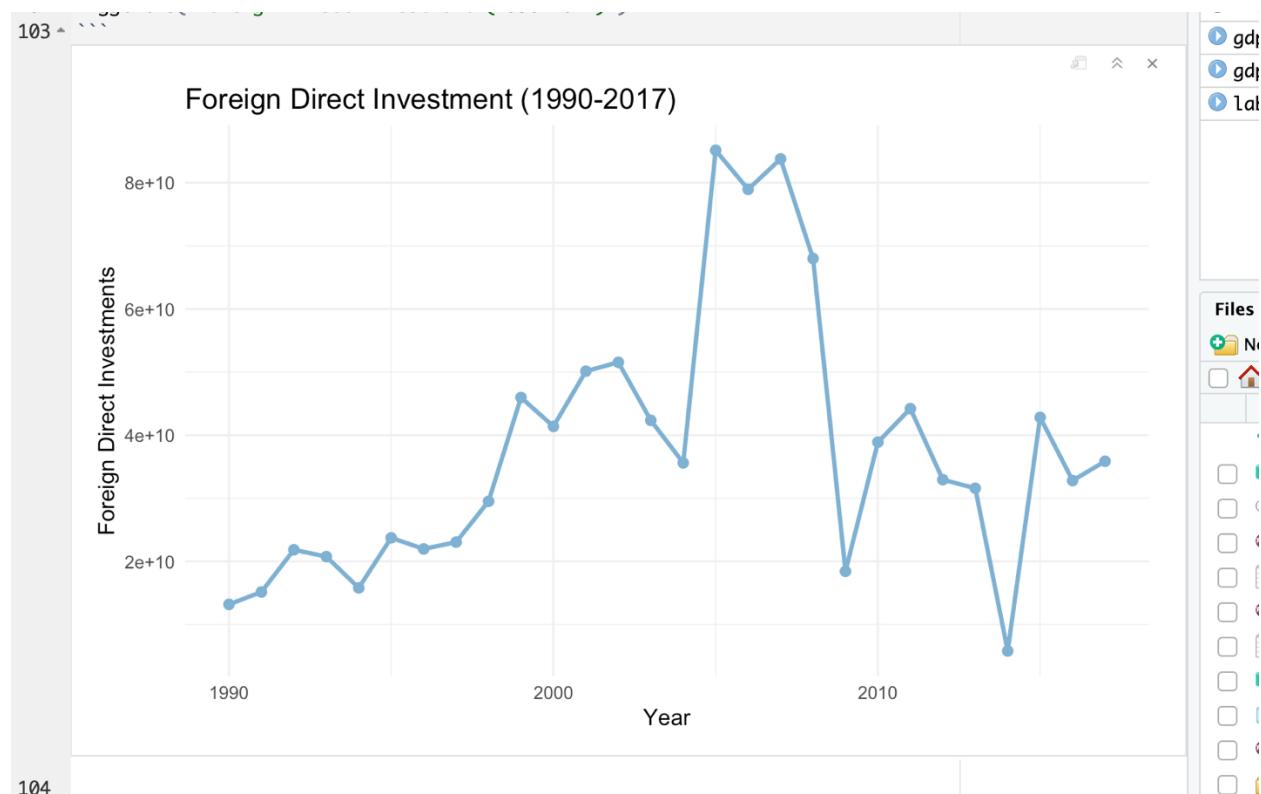
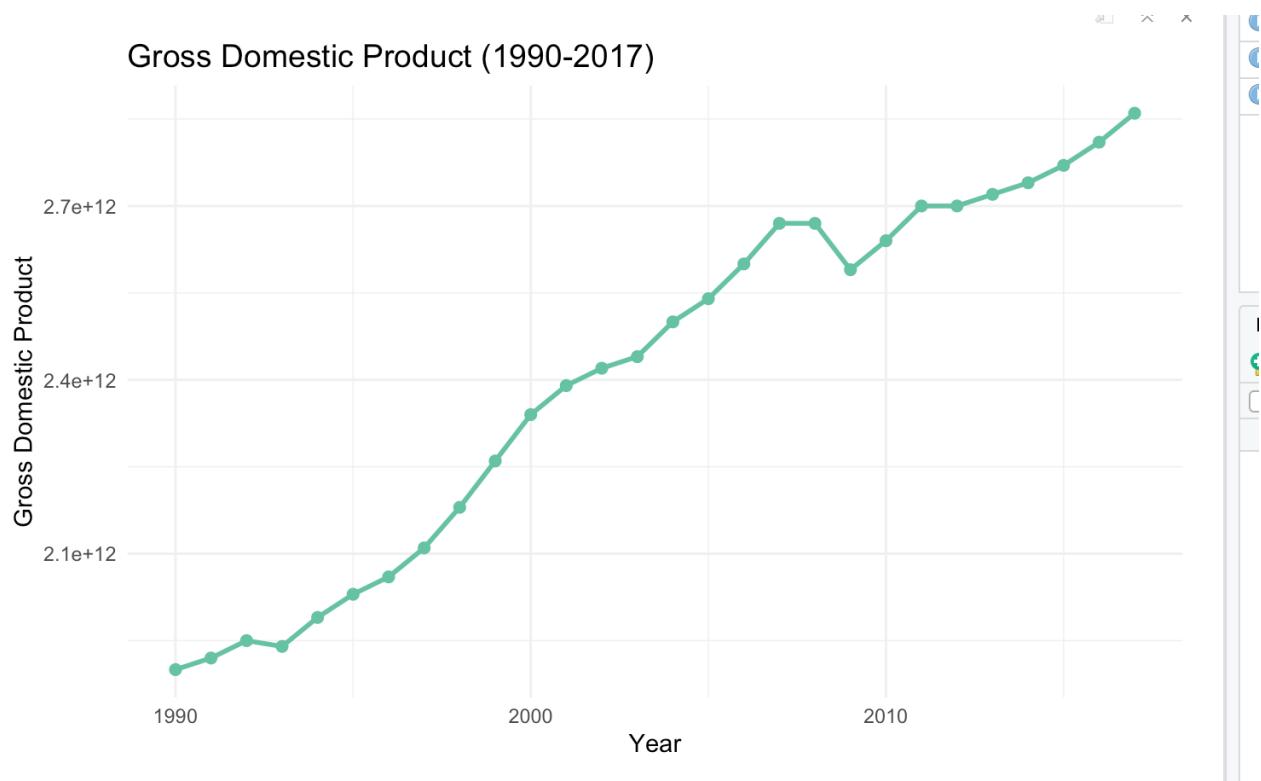
```

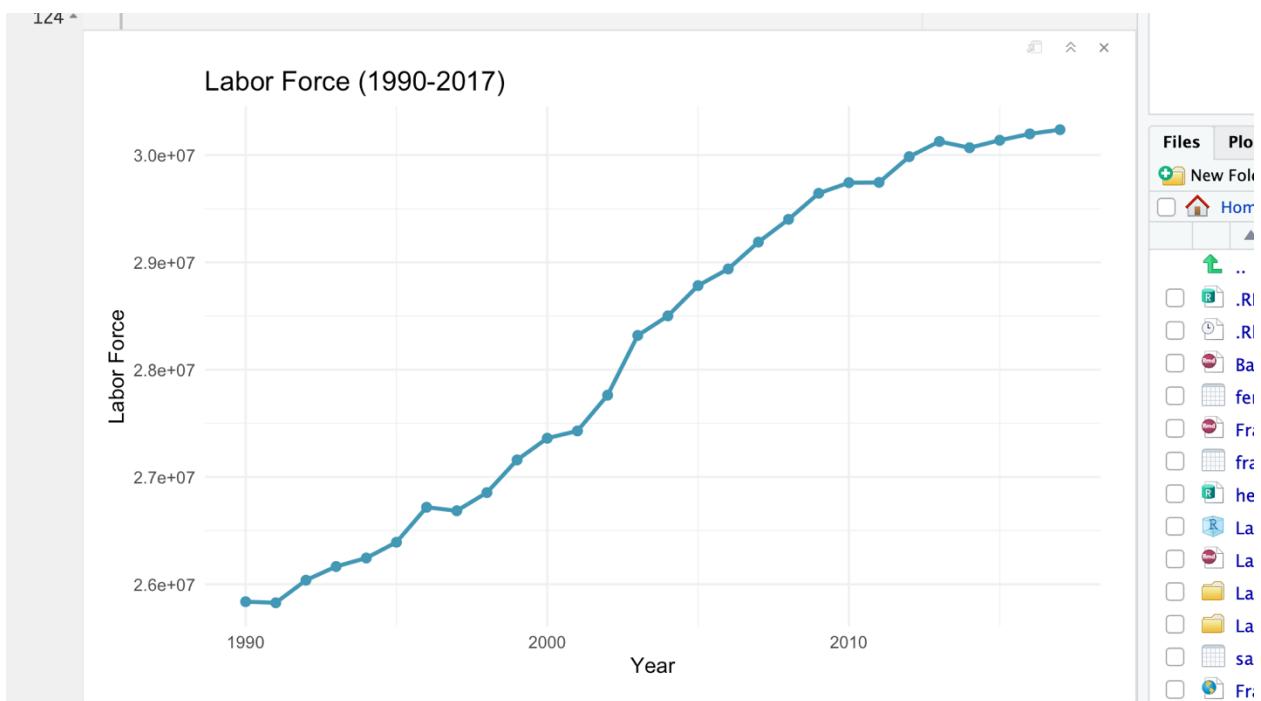
104
105 - ## Labor Force (1990-2017)
106
107 - ``{r}
108   labor_data <- gdpFdilabor %>%
109   select(year, laborForce)
110
111   ggplot(labor_data, aes(x = year, y = laborForce)) +
112     geom_line(color = "#449966", size = 1) +
113     geom_point(color = "#449966", size = 2) +
114     scale_y_continuous(labels = scales::scientific) +
115     labs(x = "Year", y = "Labor Force") +
116     theme_minimal() +
117     theme(
118       panel.grid.major = element_line(color = "#f0f0f0"),
119       panel.grid.minor = element_line(color = "#f0f0f0"),
120       axis.title.y = element_text(angle = 90),
121       plot.title = element_text(size = 14, hjust = 0)
122    ) +
123  ggtitle("Labor Force (1990-2017)")
124

```

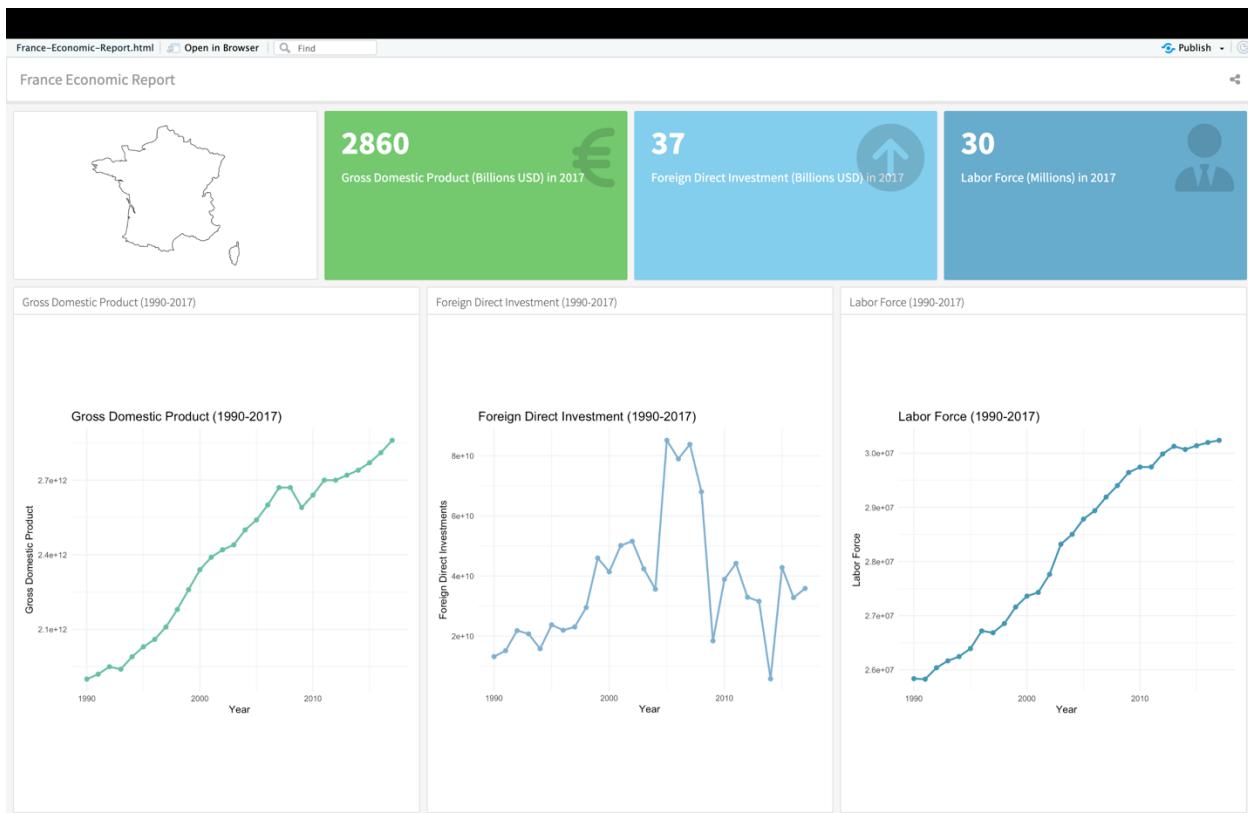
The RStudio interface includes the following components:

- Top Bar:** Shows tabs for Lab15.Rmd, Lab16.R, and France Economic Report.Rmd, along with file operations like Go to file/function, Knit on Save, Knit, Run, Publish, and Addins.
- Environment Tab:** Displays the Global Environment with objects like countries, fdi_data, france, gdp_data, gdpFdilabor, and labor_data.
- Files Tab:** Shows the project structure under Home > Documents > Semester 3 > SDA > Assignments > Week 8 > Lab 15, including files like .RData, .Rhistory, Basic.Rmd, femaleMiceWeights.csv, France Economic Report.Rmd, france.csv, hersdata.Rdata, Lab 15.Rproj, Lab15.Rmd, Lab16_II, Lab16_Week8, sample_transactions.csv, and France-Economic-Report.html.
- Console Tab:** Displays R code and its output.
- Terminal Tab:** Displays R code and its output.
- Render Tab:** Displays R code and its output.
- Background Jobs Tab:** Displays R code and its output.





Dashboard:



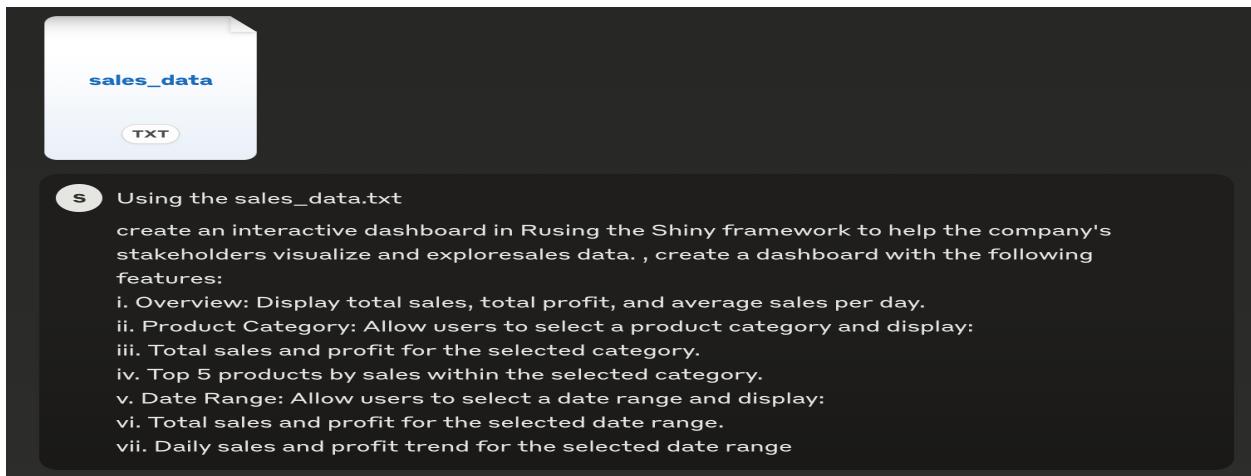
6. Gen AI Prompt Engineering:

You are a data analyst for a company that sells products through an e-commerce platform. Your task is to create an interactive dashboard in R using the Shiny framework to help the company's stakeholders visualize and explore sales data. Using the sales_data.txt, create a dashboard with the following features:

- i. Overview: Display total sales, total profit, and average sales per day.
- ii. Product Category: Allow users to select a product category and display:
- iii. Total sales and profit for the selected category.
- iv. Top 5 products by sales within the selected category.
- v. Date Range: Allow users to select a date range and display:
- vi. Total sales and profit for the selected date range.
- vii. Daily sales and profit trend for the selected date range

Bot Used: Claude.ai

Prompt:



GenAI:

Claude

I'll create an interactive Shiny dashboard using R to visualize the sales data. Here's a complete implementation that includes all the features you requested:

R Shiny Sales Dashboard
Click to open code

This R Shiny dashboard provides a comprehensive visualization of your sales data with three main tabs:

1. Overview Tab

- Shows key metrics with value boxes: total sales, total profit, and average sales per day
- Displays a bar chart of sales by product category
- Shows a line chart of daily sales and profit trends
- Includes a data table with all sales records

2. Product Category Tab

- Dropdown to select a specific product category
- Displays total sales and profit for the selected category
- Shows the top 5 products (subcategories) by sales within the selected category
- Includes a scatter plot comparing sales vs. profit for each subcategory

3. Date Range Tab

- Date range picker to select a specific time period
- Shows total sales and profit for the selected period
- Displays a detailed trend line with points showing daily sales and profit
- Includes a pie chart showing the category breakdown for the selected period

How to Use the Dashboard:

- Save the code as `app.R`
- Make sure to place your `sales_data.txt` file in the same directory

Corrected R Shiny Sales Dashboard

```

corrected_code <- sum(sales_data()$Sales)
valueBox(
  paste0("$$", format(total_profit, big.mark = ",")),
  "Total Profit",
  icon = icon("chart-line"),
  color = "blue"
)

# Average Sales Box
output$avg_sales_box <- renderValueBox({
  unique_days <- length(unique(sales_data()$Date))
  avg_sales <- sum(sales_data()$Sales) / unique_days
  valueBox(
    paste0("$$", format(round(avg_sales, 2), big.mark = ",")),
    "Avg. Sales per Day",
    icon = icon("calculator"),
    color = "purple"
  )
})

# Overview Category Plot
output$overview_category_plot <- renderPlot({
  data <- sales_data() %>%
    group_by(Product.Category) %>%
    ...
})

```

Last edited 2 hours ago

Claude

I'll create an interactive Shiny dashboard using R to visualize the sales data. Here's a complete implementation that includes all the features you requested:

R Shiny Sales Dashboard
Click to open code

This R Shiny dashboard provides a comprehensive visualization of your sales data with three main tabs:

1. Overview Tab

- Displays a bar chart of sales by product category
- Shows a line chart of daily sales and profit trends
- Includes a data table with all sales records

2. Product Category Tab

- Dropdown to select a specific product category
- Displays total sales and profit for the selected category
- Shows the top 5 products (subcategories) by sales within the selected category
- Includes a scatter plot comparing sales vs. profit for each subcategory

3. Date Range Tab

- Date range picker to select a specific time period
- Shows total sales and profit for the selected period
- Displays a detailed trend line with points showing daily sales and profit
- Includes a pie chart showing the category breakdown for the selected period

How to Use the Dashboard:

- Save the code as `app.R`
- Make sure to place your `sales_data.txt` file in the same directory

Corrected R Shiny Sales Dashboard

```

ggplot(data, aes(x = reorder(Product.Category, -Sales), y = Sales, fill = Product.Category)) +
  geom_bar(stat = "identity") +
  theme_minimal() +
  labs(x = "Category", y = "Total Sales ($)") +
  theme(legend.position = "none", axis.text.x = element_text(angle = 45, hjust = 1))

# Overview Daily Trend
output$overview_daily_trend <- renderPlot({
  data <- sales_data() %>%
    group_by(Date) %>%
    summarise(Sales = sum(Sales), Profit = sum(Profit))

  ggplot(data, aes(x = Date)) +
    geom_line(aes(y = Sales, color = "Sales"), size = 1) +
    geom_line(aes(y = Profit, color = "Profit"), size = 1) +
    scale_color_manual(values = c("Sales" = "blue", "Profit" = "red")) +
    theme_minimal() +
    labs(x = "Date", y = "Amount ($)", color = "Metric") +
    theme(legend.position = "bottom")
})

# Overview Table
output$overview_table <- renderDT({
  ...
})

```

Last edited 2 hours ago

Claude Q Interactive Sales Data Dashboard ↴ Shows total sales and profit for the selected period

- Displays a detailed trend line with points showing daily sales and profit
- Includes a pie chart showing the category breakdown for the selected period

How to Use the Dashboard:

1. Save the code as `app.R`
2. Make sure to place your `sales_data.txt` file in the same directory
3. Install the required packages if you haven't already:

```
r
install.packages(c("shiny", "shinydashboard", "ggplot2", "dplyr"))
```

4. Run the application:

```
r
shiny::runApp()
```

The dashboard is fully interactive - users can filter by product category or date range to drill down into specific aspects of the sales data. The visualizations automatically update to reflect the selected filters.

ⓘ Claude does not have the ability to run the code it generates yet.

Last edited 2 hours ago

Share ⚡ 1 ⚡ 🔍

S US Claude 3.7 Sonnet Choose style

Claude Q Interactive Sales Data Dashboard ↴ Shows total sales and profit for the selected period

- Displays a detailed trend line with points showing daily sales and profit
- Includes a pie chart showing the category breakdown for the selected period

How to Use the Dashboard:

1. Save the code as `app.R`
2. Make sure to place your `sales_data.txt` file in the same directory
3. Install the required packages if you haven't already:

```
r
install.packages(c("shiny", "shinydashboard", "ggplot2", "dplyr"))
```

4. Run the application:

```
r
shiny::runApp()
```

The dashboard is fully interactive - users can filter by product category or date range to drill down into specific aspects of the sales data. The visualizations automatically update to reflect the selected filters.

ⓘ Claude does not have the ability to run the code it generates yet.

Last edited 2 hours ago

Share ⚡ 1 ⚡ 🔍

S US Claude 3.7 Sonnet Choose style

The image shows a Mac desktop with two browser windows open. The top window is a Chrome instance displaying an R script for a shiny dashboard. The script includes code for a bar chart (# Corrected R Shiny Sales Dashboard), a scatter plot (# Subcategory Plot), and a date range tab output (# DATE RANGE TAB OUTPUTS). The bottom window is also a Chrome instance showing the generated shiny app interface. The app has a sidebar with tabs like 'US' and 'Claude 3.7 Sonnet'. The main area displays a sales dashboard with various charts and filters. A status bar at the bottom indicates the date as Fri Feb 28 7:18PM.

Code:

The screenshot shows the RStudio interface with two main panes. The left pane displays an R Markdown file named 'Lab16.Rmd' containing R code for creating a dashboard. The right pane shows the preview of the dashboard, which includes a sidebar menu with tabs for Overview, Category, and Date Range, and two main content boxes showing overall sales and profit by category.

```
1 #> # Lub16_Problem GENAI
2
3 # Load and preprocess the data
4 sales_data <- read.csv("sales_data.txt")
5
6 # Load necessary libraries
7 library(shiny)
8 library(shinydashboard)
9 library(ggplot2)
10 library(dplyr)
11 library(lubridate)
12 library(DT)
13
14 # UI
15 ui <- dashboardPage(
16   dashboardHeader(title = "Sales Data Dashboard"),
17   dashboardSidebar(
18     sidebarMenu(
19       menuItem("Overview", tabName = "overview", icon = icon("dashboard")),
20       menuItem("Product Category", tabName = "category", icon = icon("tag")),
21       menuItem("Date Range", tabName = "date", icon = icon("calendar"))
22     )
23   ),
24   dashboardBody(
25     tabItems(
26       # Overview Tab
27       tabItem(tabName = "overview",
28         fluidRow(
29           valueBoxOutput("total_sales_box", width = 4),
30           valueBoxOutput("total_profit_box", width = 4),
31           valueBoxOutput("avg_sales_box", width = 4)
32         ),
33         fluidRow(
34           box(
35             title = "Overall Sales by Category",
36             status = "primary",
37             solidHeader = TRUE,
38             plotOutput("overview_category_plot", height = 250)
39           ),
40           box(
41             title = "Daily Sales Trend",
42             status = "primary",
43             solidHeader = TRUE,
44             plotOutput("overview_daily_trend", height = 250)
45           )
46         ),
47       )
48     )
49   )
50 )
```

IE5390 Structured Data Analytics

The screenshot shows the RStudio interface with several tabs open:

- Lab15.Rmd
- Genal.R
- sales_data.txt
- Lab16_I.R
- France Economic Report.Rmd

The main pane displays the code for the `Lab15.Rmd` file, which contains R code for creating a dashboard. The code includes:

- A tab item for "Category Tab" with a fluid row containing a box for filtering by category.
- A fluid row with two value box outputs for "category_sales_box" and "category_profit_box".
- A fluid row with a box for "Top 5 Products by Sales" with a height of 300.
- A fluid row with a box for "Sales by Subcategory" with a height of 300.
- A tab item for "Date Range Tab" with a fluid row containing a box for selecting a date range.
- A fluid row with two value box outputs for "date_sales_box" and "date_profit_box".

The top right corner shows the "Environment" tab selected, displaying the global environment. The bottom right corner shows the file browser with the following structure:

- Home > Documents > Semester 3 > SDA > Assignments > Week 8 > GenAI_Q3
- Genal.R (11.4 KB, Feb 28, 2025, 7:14 PM)
- sales_data.txt (726 B, Feb 28, 2025, 4:49 PM)

The screenshot shows the RStudio interface with several panes:

- Script Editor:** Displays an R script named `Genai.R`. The code reads a CSV file `sales_data.txt`, processes it, and creates a plot named `date_category_plot`.
- Environment:** Shows the global environment with a single object `server` of type `function`.
- File Browser:** Shows the file structure: `Home > Documents > Semester 3 > SDA > Assignments > Week 8 > GenAI_Q3`. It lists two files: `Genai.R` (11.4 KB, modified Feb 28, 2025, 7:14 PM) and `sales_data.txt` (726 B, modified Feb 28, 2025, 4:49 PM).
- Console:** Shows the command `source("Genai.R")` being run.

```

160 date_range <- range(sales_data()$Date)
161 updateDateRangeInput(session, "date_range",
162   start = date_range[1],
163   end = date_range[2])
164 })
165
166 # OVERVIEW TAB OUTPUTS
167
168 # Total Sales Box
169 output$total_sales_box <- renderValueBox({
170   total_sales <- sum(sales_data()$Sales)
171   valueBox(
172     paste0("$", format(total_sales, big.mark = "")),
173     "Total Sales",
174     icon = icon("dollar-sign"),
175     color = "#00A0A0"
176   )
177 })
178
179 # Total Profit Box
180 output$total_profit_box <- renderValueBox({
181   total_profit <- sum(sales_data()$Profit)
182   valueBox(
183     paste0("$", format(total_profit, big.mark = "")),
184     "Total Profit",
185     icon = icon("chart-line"),
186     color = "#00A0A0"
187   )
188 })
189
190 # Average Sales Box
191 output$avg_sales_box <- renderValueBox({
192   unique_days <- length(unique(sales_data()$Date))
193   avg_sales <- sum(sales_data()$Sales) / unique.days
194   valueBox(
195     paste0("$", format(round(avg_sales, 2), big.mark = "")),
196     "Avg. Sales per Day",
197     icon = icon("calculator"),
198     color = "#A0A0D0"
199   )
200 })
201
202 # Overview Category Plot
203 output$overview_category_plot <- renderPlot({
204   data <- sales_data() %>%
205     group_by(Product.Category) %>%
206     summarise(Sales = sum(Sales))
207
208 ggplot(data, aes(x = reorder(Product.Category, -Sales), y = Sales, fill = Product.Category)) +
209   geom_bar(stat = "identity") +
210   theme_minimal() +
211   labs(x = "Category", y = "Total Sales ($)") +
212   theme(legend.position = "none", axis.text.x = element_text(angle = 45, hjust = 1))
213 })
214
215 # Overview Daily Trend
216 output$overview_daily_trend <- renderPlot({
217   data <- sales_data() %>%
218     group_by(Date) %>%
219     summarise(Sales = sum(Sales), Profit = sum(Profit))
220
221 ggplot(data, aes(x = Date)) +
222   geom_line(aes(y = Sales, color = "Sales"), size = 1) +
223   geom_line(aes(y = Profit, color = "Profit"), size = 1) +
224   scale_color_manual(values = c("Sales" = "#00A0A0", "Profit" = "#FF0000")) +
225   theme_minimal() +
226   labs(x = "Date", y = "Amount ($)", color = "Metric") +
227   theme(legend.position = "bottom")
228 })
229
230 # Overview Table
231 output$overview_table <- renderDT({
232   datatable(sales_data(), options = list(pageLength = 5))
233 })
234
235 # CATEGORY TAB OUTPUTS
236
237 # Filtered data by category
238 filtered_category_data <- reactive({
239   req(input$category_filter)
240   sales_data() %>%
241     filter(Product.Category == input$category_filter)
242 })
243
244 # Category Sales Box
245 output$category_sales_box <- renderValueBox({
246   req(input$category_filter)
247   category_sales <- sum(filtered_category_data()$Sales)
248 })
249
250 
```

```

251   valueBox(
252     paste0("$", format(category_sales, big.mark = "")),
253     "Category Sales",
254     icon = icon("dollar-sign"),
255     color = "#00A0A0"
256   )
257 })
258
259 
```

The screenshot shows an RStudio interface with several panes:

- Code Editor:** Displays an R script with code for generating reports. The script includes sections for "CATEGORY TAB OUTPUTS", "Category Sales Box", "Category Profit Box", and "Top Products Plot". It uses reactive programming with `reactive` and `req` functions, and performs data summarization and plotting.
- Environment:** Shows the global environment with a single object named "server".
- Files:** Shows the project structure: Home > Documents > Semester 3 > SDA > Assignments > Week 8 > GenAI_Q3. Inside, there are files "Genai.R" (11.4 KB, modified Feb 28, 2025, 7:14 PM) and "sales_data.txt" (726 B, modified Feb 28, 2025, 4:49 PM).
- Console:** Shows the command "(Top Level) :".

The screenshot shows an RStudio interface with several tabs open. The main code editor contains an R script for generating reports from sales data. The script includes functions for creating bar charts for top products and subcategory plots, filtering data by date range, and creating a sales box visualization.

```
269 + output$top_products_plot <- renderPlot({
270   req(input$category_filter)
271   data <- filtered_category_data()
272   group_by(Product, Subcategory) %>%
273   summarise(Sales = sum(Sales)) %>%
274   arrange(desc(Sales)) %>%
275   slice_head(n = 5)
276 
277   ggplot(data, aes(x = reorder(Product.Subcategory, -Sales), y = Sales, fill = Product.Subcategory))
278   geom_bar(stat = "identity") +
279   theme_minimal() +
280   labs(x = "Product", y = "Total Sales ($)", title = paste("Top 5 Products in", input$category_filter))
281   theme(legend.position = "none", axis.text.x = element_text(angle = 45, hjust = 1))
282 })
283 
284 # Subcategory Plot
285 output$subcategory_plot <- renderPlot({
286   req(input$category_filter)
287   data <- filtered_category_data()
288   group_by(Product, Subcategory) %>%
289   summarise(Sales = sum(Sales), Profit = sum(Profit))
290 
291   ggplot(data, aes(x = Sales, y = Profit, color = Product.Subcategory)) +
292   geom_point(size = 5, alpha = 0.7) +
293   theme_minimal() +
294   labs(x = "Sales ($)", y = "Profit ($)", color = "Subcategory") +
295   theme(legend.position = "bottom")
296 })
297 
298 # DATE RANGE TAB OUTPUTS
299 
300 # Filtered data by date range
301 filtered_date_data <- reactive({
302   req(input$date_range)
303   sales_data() %>%
304   filter(Date >= input$date_range[1] & Date <= input$date_range[2])
305 })
306 
307 # Date Sales Box
308 output$date_sales_box <- renderValueBox({
309   req(input$date_range)
310   date_sales <- sum(filtered_date_data()$Sales)
311   valueBox(
312     paste0("$", format(date_sales, big.mark = ",")),
313     "Total Sales for Selected Period",
314     icon = icon("dollar-sign"),
315   )
316 })
```

```

318 # Date Profit Box
319 output$date_profit_box <- renderValueBox({
320   req(inputs$date_range)
321   date_profit <- sum(filtered_date_data()$Profit)
322   valueBox(
323     paste0("$", format(date_profit, big.mark = ",")),
324     "Total Profit for Selected Period",
325     icon = icon("chart-line"),
326     color = "blue"
327   )
328 })
329
330 # Date Trend Plot
331 output$date_trend_plot <- renderPlot({
332   req(inputs$date_range)
333   data <- filtered_date_data() %>%
334     group_by(Date) %>%
335     summarise(Sales = sum(Sales), Profit = sum(Profit))
336
337 ggplot(data, aes(x = Date)) +
338   geom_line(aes(y = Sales, color = "Sales"), size = 1) +
339   geom_point(aes(y = Sales, color = "Sales"), size = 3) +
340   geom_line(aes(y = Profit, color = "Profit"), size = 1) +
341   geom_point(aes(y = Profit, color = "Profit"), size = 3) +
342   scale_color_manual(values = c("Sales" = "blue", "Profit" = "red")) +
343   theme_minimal() +
344   labs(x = "Date", y = "Amount ($)", color = "Metric") +
345   theme(legend.position = "bottom")
346 })
347
348 # Date Category Plot
349 output$date_category_plot <- renderPlot({
350   req(inputs$date_range)
351   data <- filtered_date_data() %>%
352     group_by(Product.Category) %>%
353     summarise(Sales = sum(Sales))
354
355 ggplot(data, aes(x = "", y = Sales, fill = Product.Category)) +
356   geom_bar(stat = "identity", width = 1) +
357   coord_polar("y", start = 0) +
358   theme_minimal() +
359   labs(fill = "Category", title = "Sales by Category") +
360   theme(axis.title = element_blank(),
361         axis.text = element_blank(),
362         axis.ticks = element_blank())
363 })
364
365 # Run the application
366 shinyApp(ui = ui, server = server)
8.17 (Top Level) : R Script

```

```

323 valueBox(
324   paste0("$", format(date_profit, big.mark = ",")),
325   "Total Profit for Selected Period",
326   icon = icon("chart-line"),
327   color = "blue"
328 )
329
330 # Date Trend Plot
331 output$date_trend_plot <- renderPlot({
332   req(inputs$date_range)
333   data <- filtered_date_data() %>%
334     group_by(Date) %>%
335     summarise(Sales = sum(Sales), Profit = sum(Profit))
336
337 ggplot(data, aes(x = Date)) +
338   geom_line(aes(y = Sales, color = "Sales"), size = 1) +
339   geom_point(aes(y = Sales, color = "Sales"), size = 3) +
340   geom_line(aes(y = Profit, color = "Profit"), size = 1) +
341   geom_point(aes(y = Profit, color = "Profit"), size = 3) +
342   scale_color_manual(values = c("Sales" = "blue", "Profit" = "red")) +
343   theme_minimal() +
344   labs(x = "Date", y = "Amount ($)", color = "Metric") +
345   theme(legend.position = "bottom")
346 })
347
348 # Date Category Plot
349 output$date_category_plot <- renderPlot({
350   req(inputs$date_range)
351   data <- filtered_date_data() %>%
352     group_by(Product.Category) %>%
353     summarise(Sales = sum(Sales))
354
355 ggplot(data, aes(x = "", y = Sales, fill = Product.Category)) +
356   geom_bar(stat = "identity", width = 1) +
357   coord_polar("y", start = 0) +
358   theme_minimal() +
359   labs(fill = "Category", title = "Sales by Category") +
360   theme(axis.title = element_blank(),
361         axis.text = element_blank(),
362         axis.ticks = element_blank())
363 })
364
365 # Run the application
366 shinyApp(ui = ui, server = server)
8.17 (Top Level) : R Script

```

