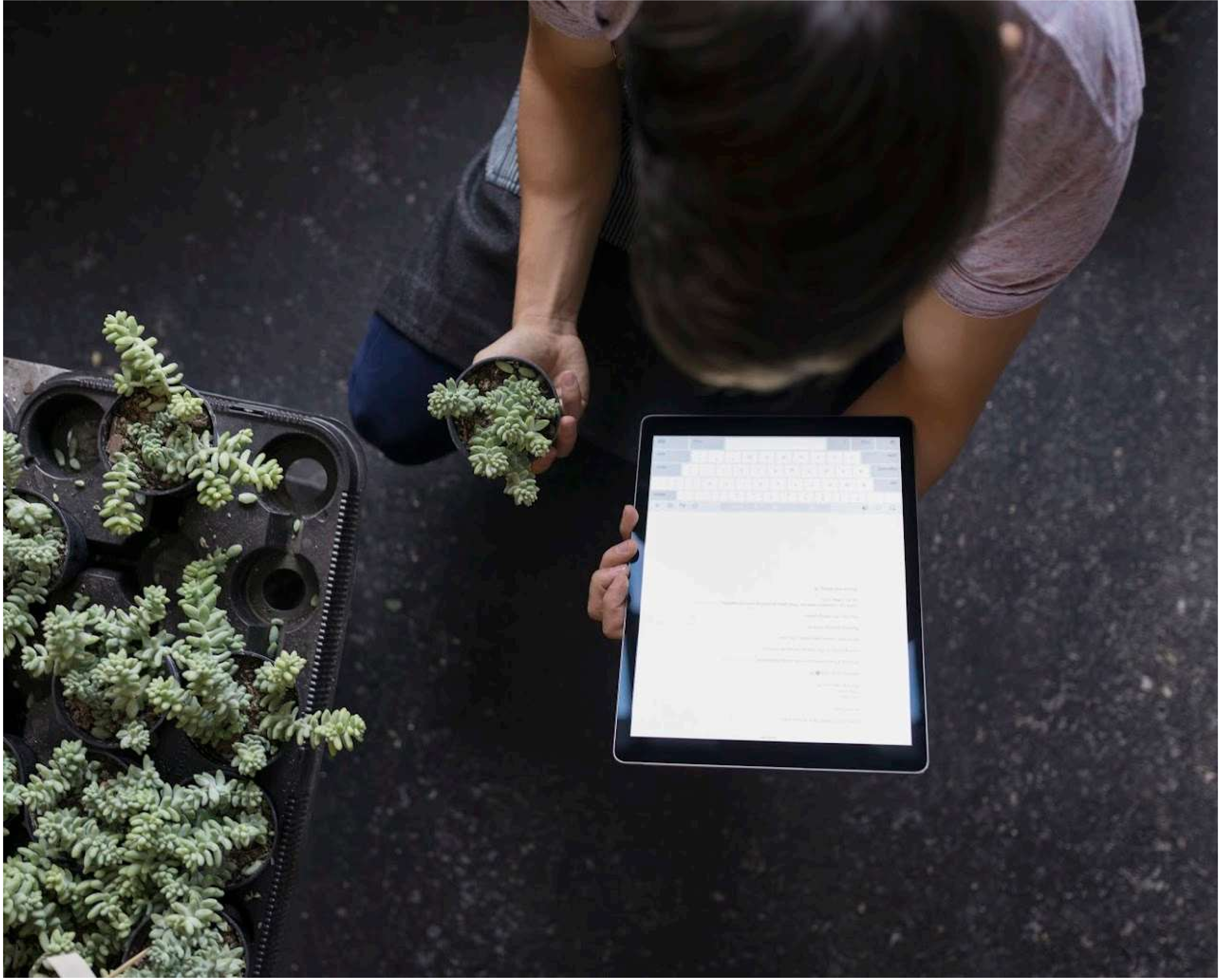
Walmart Sales Analysis

A brief analysis of the Walmart sales using R

Customer ratings, revenue and profit analysis of Walmart for the last 10 years

Visualization



Milestone 1

Data Upload

Milestone 2

Data Cleaning

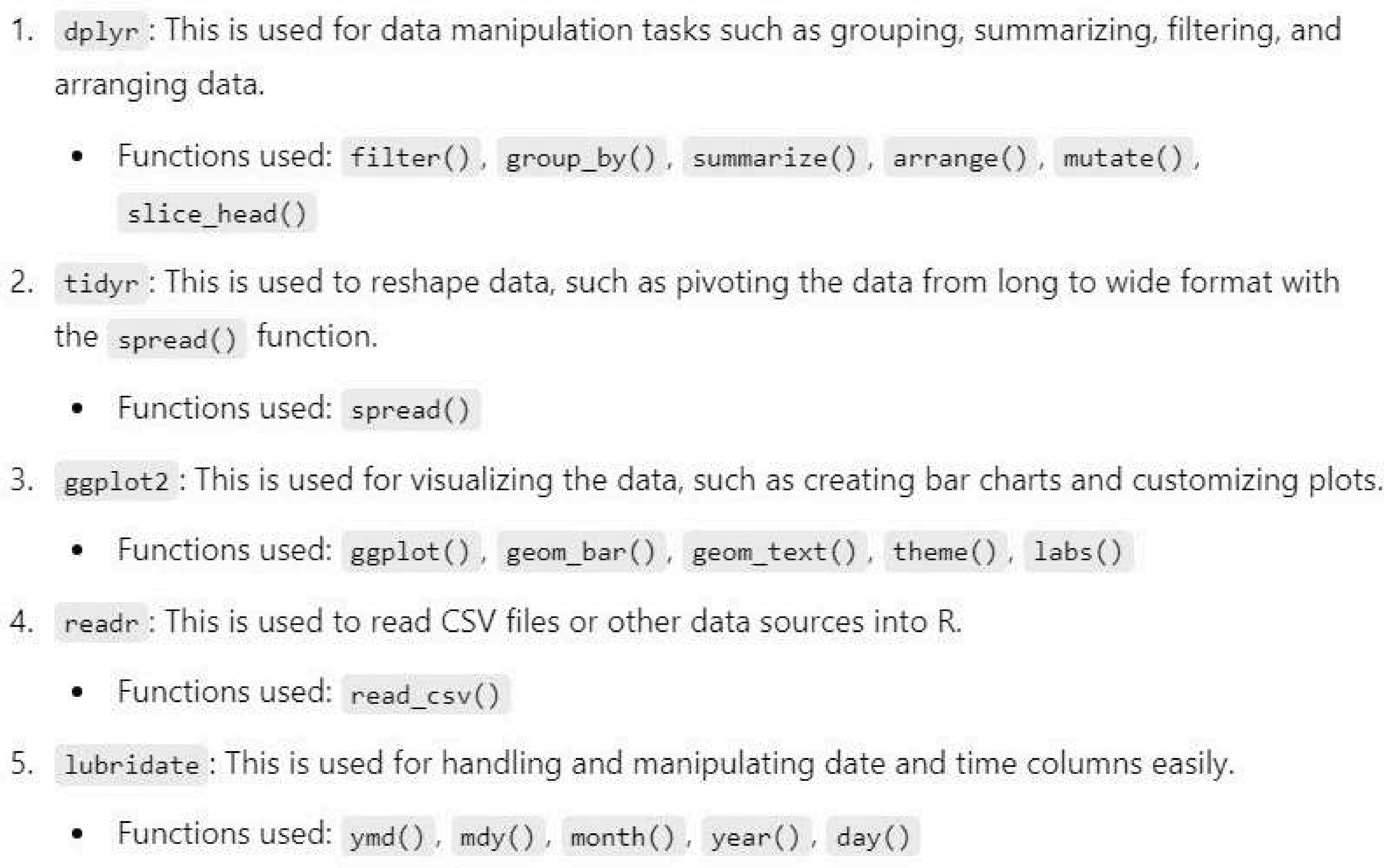
Milestone 3

Data Analysis

Milestone 2

Data Presenting

I. Dependencies used in the project



1. `dplyr` : This is used for data manipulation tasks such as grouping, summarizing, filtering, and arranging data.
   - Functions used: `filter()` , `group_by()` , `summarize()` , `arrange()` , `mutate()` , `slice_head()`
2. `tidyr` : This is used to reshape data, such as pivoting the data from long to wide format with the `spread()` function.
   - Functions used: `spread()`
3. `ggplot2` : This is used for visualizing the data, such as creating bar charts and customizing plots.
   - Functions used: `ggplot()` , `geom_bar()` , `geom_text()` , `theme()` , `labs()`
4. `readr` : This is used to read CSV files or other data sources into R.
   - Functions used: `read_csv()`
5. `lubridate` : This is used for handling and manipulating date and time columns easily.
   - Functions used: `ymd()` , `mdy()` , `month()` , `year()` , `day()`

R Dependencies : Libraries & Packages

Following libraries and packages have been used:

- dplyr
- tidyr
- readr
- lubridate

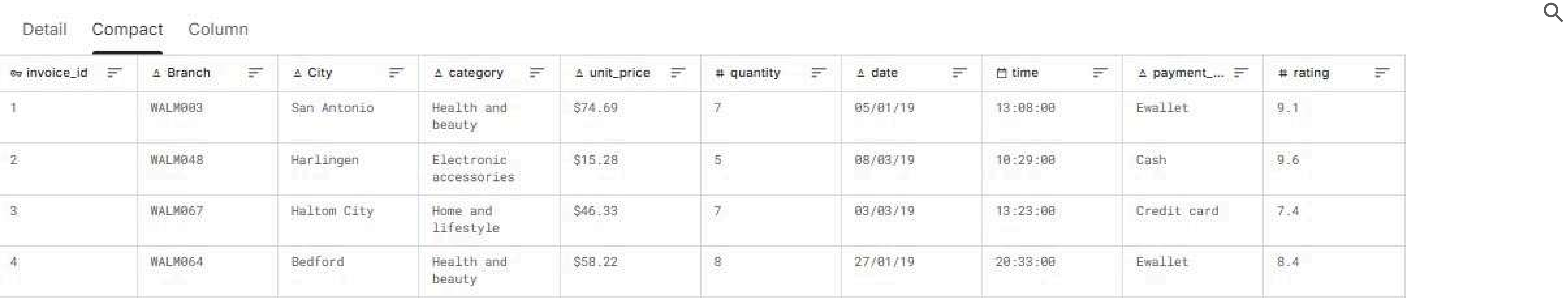
II. Data structure & summary of the dataset used for analysis purpose

Dataset : Wallmart Sales

You can see 10051 entries with 11 columns and data type of each columns

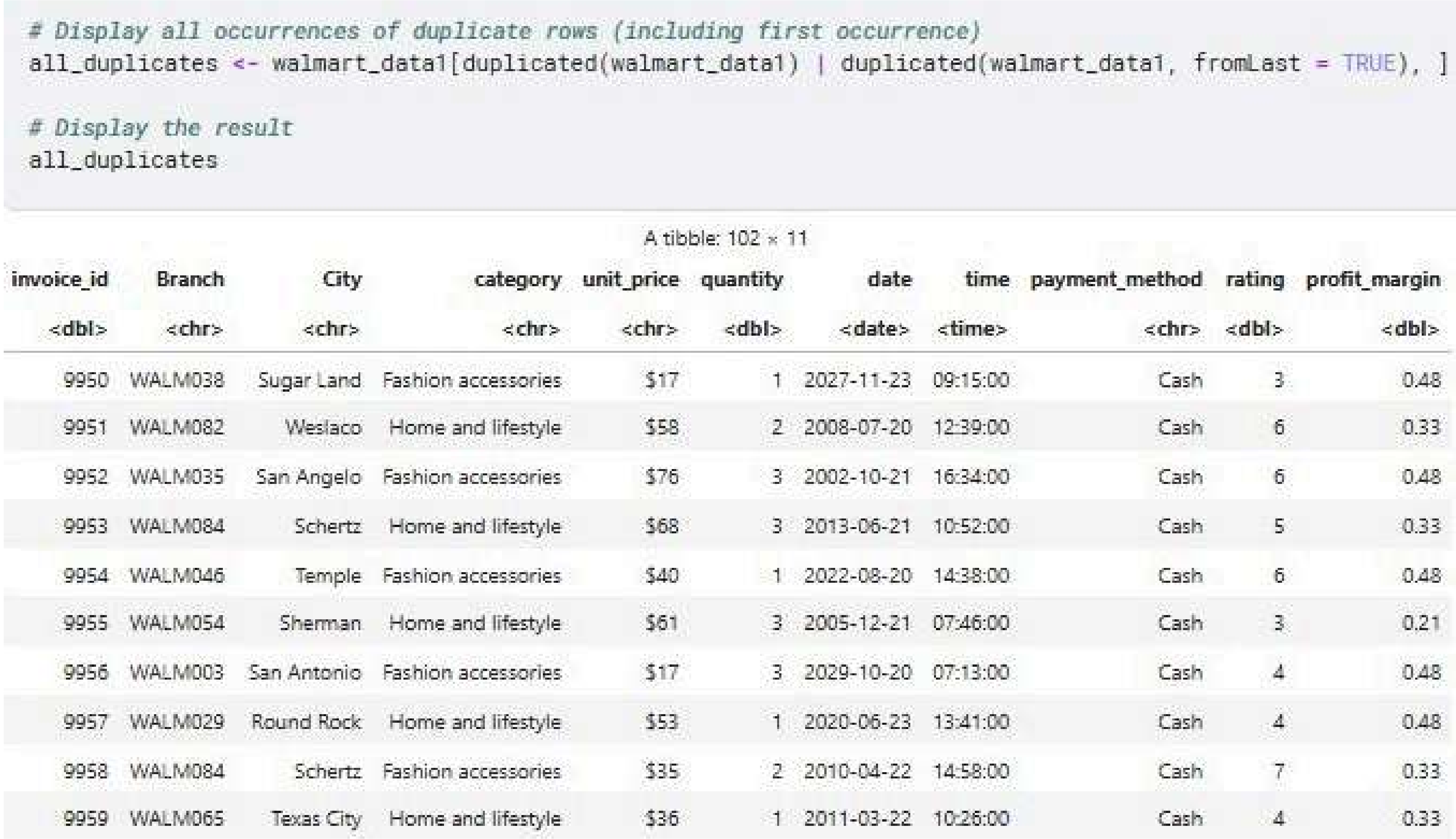
Observe the unit price column structure to understand the need of the cleaning

III. Data View



invoice_id	Branch	City	category	unit_price	quantity	date	time	payment_...	rating
1	WALM003	San Antonio	Health and beauty	$74.69	7	05/01/19	13:08:00	Ewallet	9.1
2	WALM048	Harlingen	Electronic accessories	$15.28	5	08/03/19	10:29:00	Cash	9.6
3	WALM067	Haltom City	Home and lifestyle	$46.33	7	03/03/19	13:23:00	Credit card	7.4
4	WALM064	Bedford	Health and beauty	$58.22	8	27/01/19	20:33:00	Ewallet	8.4

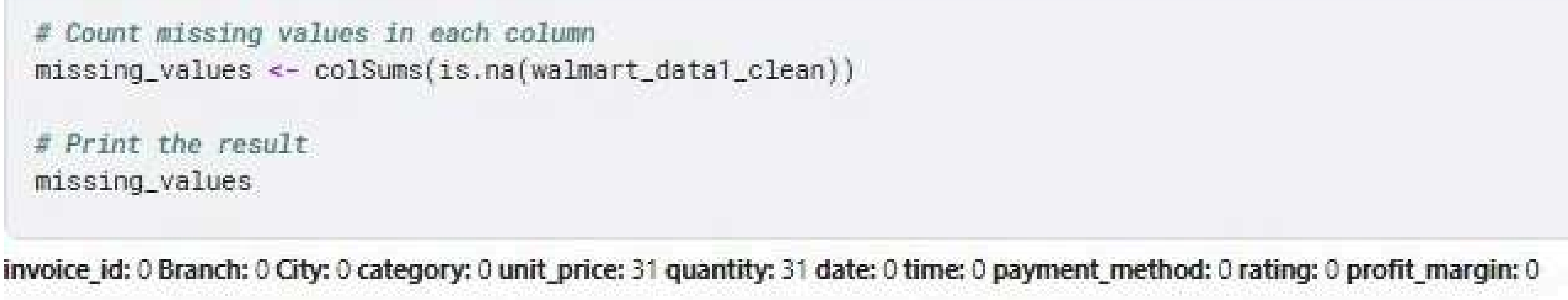
IV. Cleaning & Data Preparation : missing values and duplicates



```r
# Display all occurrences of duplicate rows (including first occurrence)
all_duplicates <- walmart_data1[duplicated(walmart_data1) | duplicated(walmart_data1, fromLast = TRUE), ]

# Display the result
all_duplicates
```

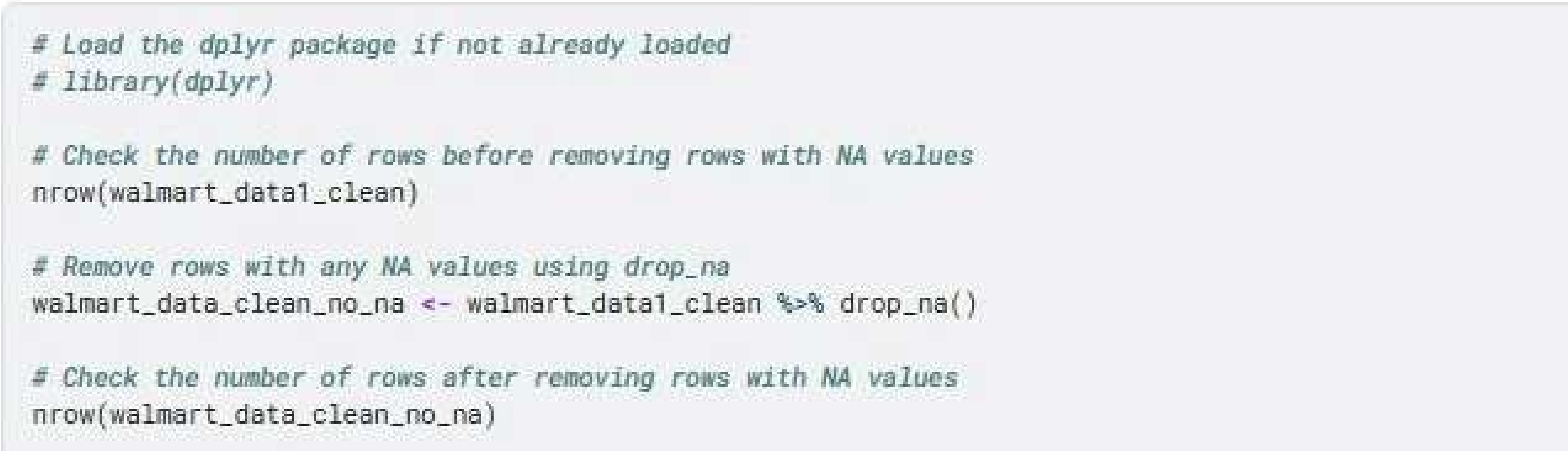
A tibble: 102 × 11

invoice_id	Branch	City	category	unit_price	quantity	date	time	payment_method	rating	profit_margin
<dbl>	<chr>	<chr>	<chr>	<chr>	<dbl>	<date>	<time>	<chr>	<dbl>	<dbl>
9950	WALM038	Sugar Land	Fashion accessories	$17	1	2027-11-23	09:15:00	Cash	3	0.48
9951	WALM082	Weslaco	Home and lifestyle	$58	2	2008-07-20	12:39:00	Cash	6	0.33
9952	WALM035	San Angelo	Fashion accessories	$76	3	2002-10-21	16:34:00	Cash	6	0.48
9953	WALM084	Schertz	Home and lifestyle	$68	3	2013-06-21	10:52:00	Cash	5	0.33
9954	WALM046	Temple	Fashion accessories	$40	1	2022-08-20	14:38:00	Cash	6	0.48
9955	WALM054	Sherman	Home and lifestyle	$61	3	2005-12-21	07:46:00	Cash	3	0.21
9956	WALM003	San Antonio	Fashion accessories	$17	3	2029-10-20	07:13:00	Cash	4	0.48
9957	WALM029	Round Rock	Home and lifestyle	$53	1	2020-06-23	13:41:00	Cash	4	0.48
9958	WALM084	Schertz	Fashion accessories	$35	2	2010-04-22	14:58:00	Cash	7	0.33
9959	WALM065	Texas City	Home and lifestyle	$36	1	2011-03-22	10:26:00	Cash	4	0.33

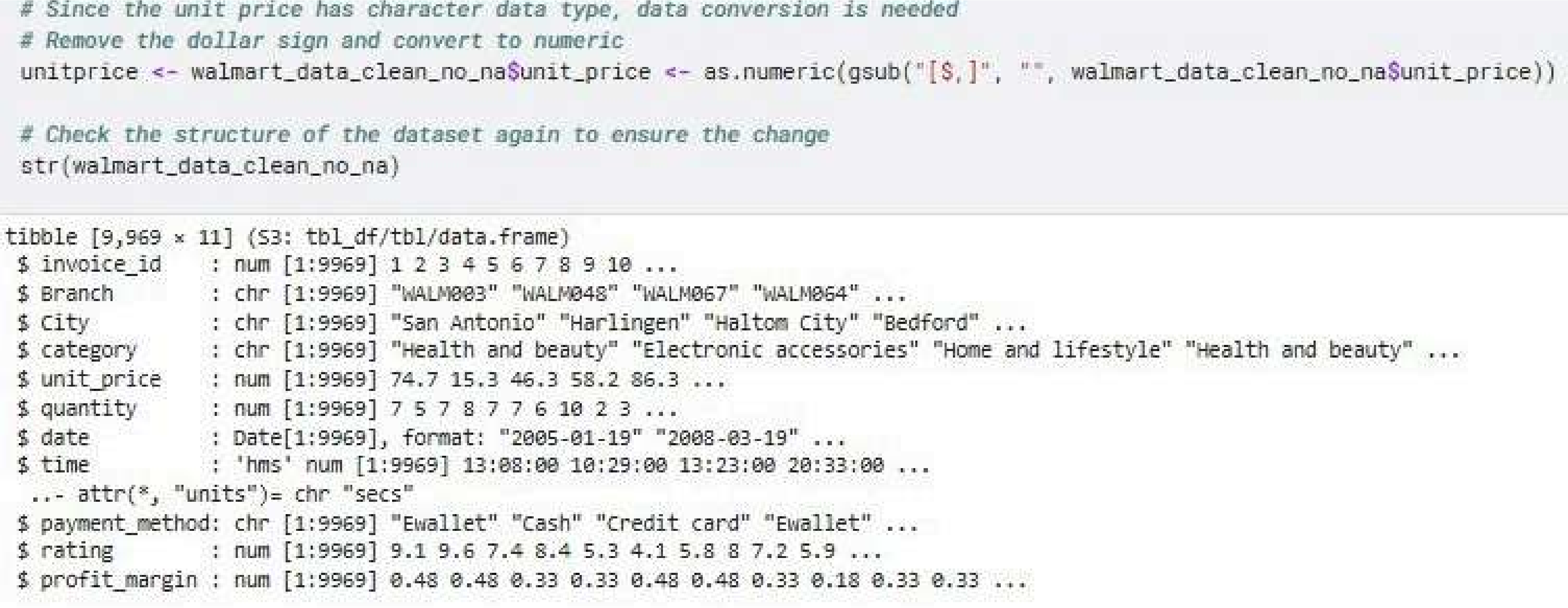


```r
# Count missing values in each column
missing_values <- colSums(is.na(walmart_data1_clean))

# Print the result
missing_values
```

```
invoice_id: 0 Branch: 0 City: 0 category: 0 unit_price: 31 quantity: 31 date: 0 time: 0 payment_method: 0 rating: 0 profit_margin: 0
```



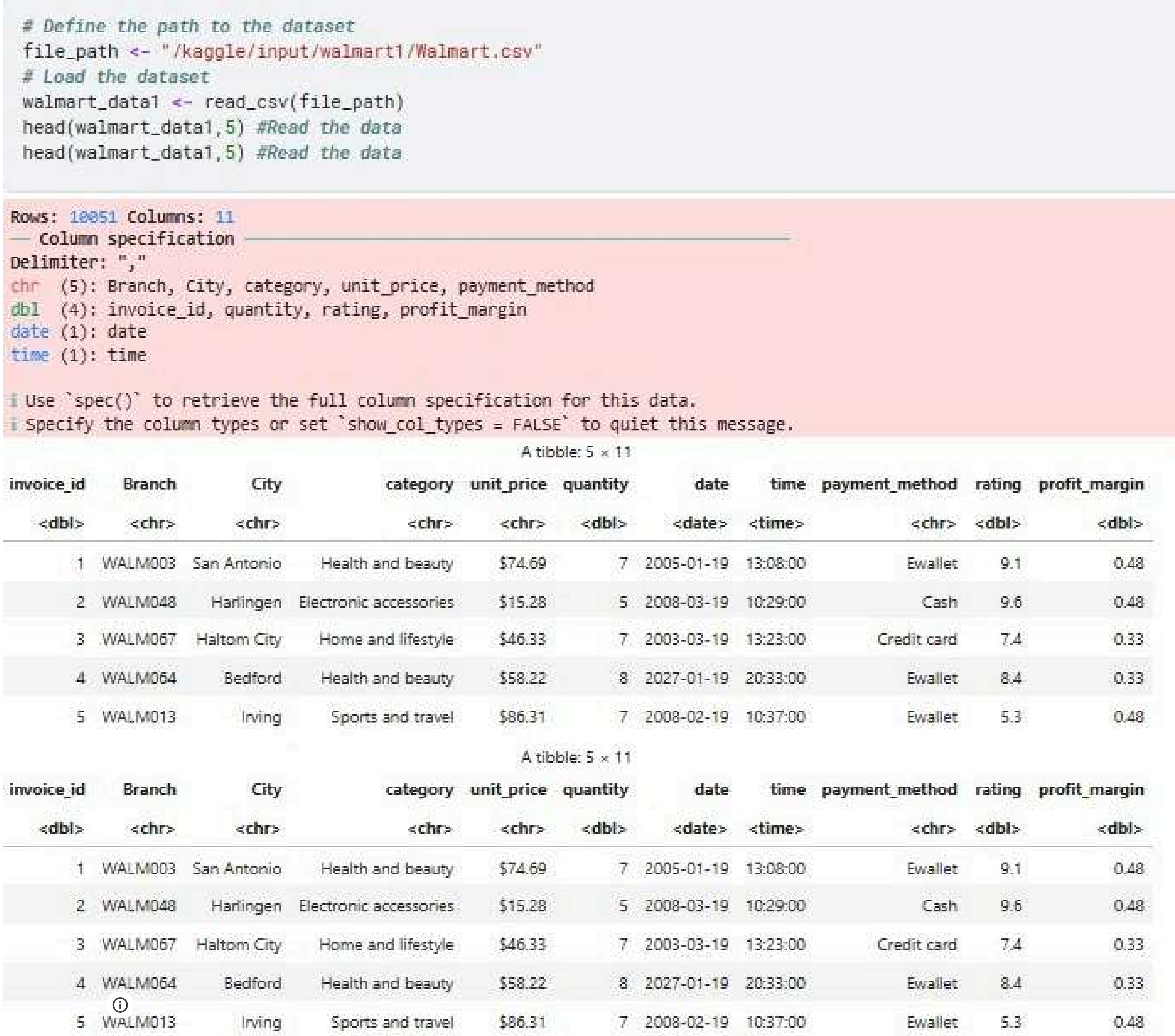
```r
# Load the dplyr package if not already loaded
# library(dplyr)

# Check the number of rows before removing rows with NA values
nrow(walmart_data1_clean)

# Remove rows with any NA values using drop_na
walmart_data_clean_no_na <- walmart_data1_clean %>% drop_na()

# Check the number of rows after removing rows with NA values
nrow(walmart_data_clean_no_na)
```



```r
# Since the unit price has character data type, data conversion is needed
# Remove the dollar sign and convert to numeric
unitprice <- walmart_data_clean_no_na$unit_price <- as.numeric(gsub("[$,]", "", walmart_data_clean_no_na$unit_price))

# Check the structure of the dataset again to ensure the change
str(walmart_data_clean_no_na)
```

```
tibble [9,969 × 11] (S3: tbl_df/tbl/data.frame)
 $ invoice_id    : num [1:9969] 1 2 3 4 5 6 7 8 9 10 ...
 $ Branch        : chr [1:9969] "WALM003" "WALM048" "WALM067" "WALM064" ...
 $ City          : chr [1:9969] "San Antonio" "Harlingen" "Haltom City" "Bedford" ...
 $ category      : chr [1:9969] "Health and beauty" "Electronic accessories" "Home and lifestyle" "Health and beauty" ...
 $ unit_price    : num [1:9969] 74.7 15.3 46.3 58.2 86.3 ...
 $ quantity      : num [1:9969] 7 5 7 8 7 7 6 10 2 3 ...
 $ date          : Date[1:9969], format: "2005-01-19" "2008-03-19" ...
 $ time          : 'hms' num [1:9969] 13:08:00 10:29:00 13:23:00 20:33:00 ...
  ..- attr(*, "units")= chr "secs"
 $ payment_method: chr [1:9969] "Ewallet" "Cash" "Credit card" "Ewallet" ...
 $ rating        : num [1:9969] 9.1 9.6 7.4 8.4 5.3 4.1 5.8 8 7.2 5.9 ...
 $ profit_margin : num [1:9969] 0.48 0.48 0.33 0.33 0.48 0.48 0.33 0.18 0.33 0.33 ...
```

V. Cleaned Data Upload



```
# Define the path to the dataset
file_path <- "/kaggle/input/walmart1/Walmart.csv"
# Load the dataset
walmart_data1 <- read_csv(file_path)
head(walmart_data1,5) #Read the data
head(walmart_data1,5) #Read the data
```

```
Rows: 10051 Columns: 11
── Column specification ────────────────────────────────────────────────────────
Delimiter: ","
chr  (5): Branch, City, category, unit_price, payment_method
dbl  (4): invoice_id, quantity, rating, profit_margin
date (1): date
time (1): time

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

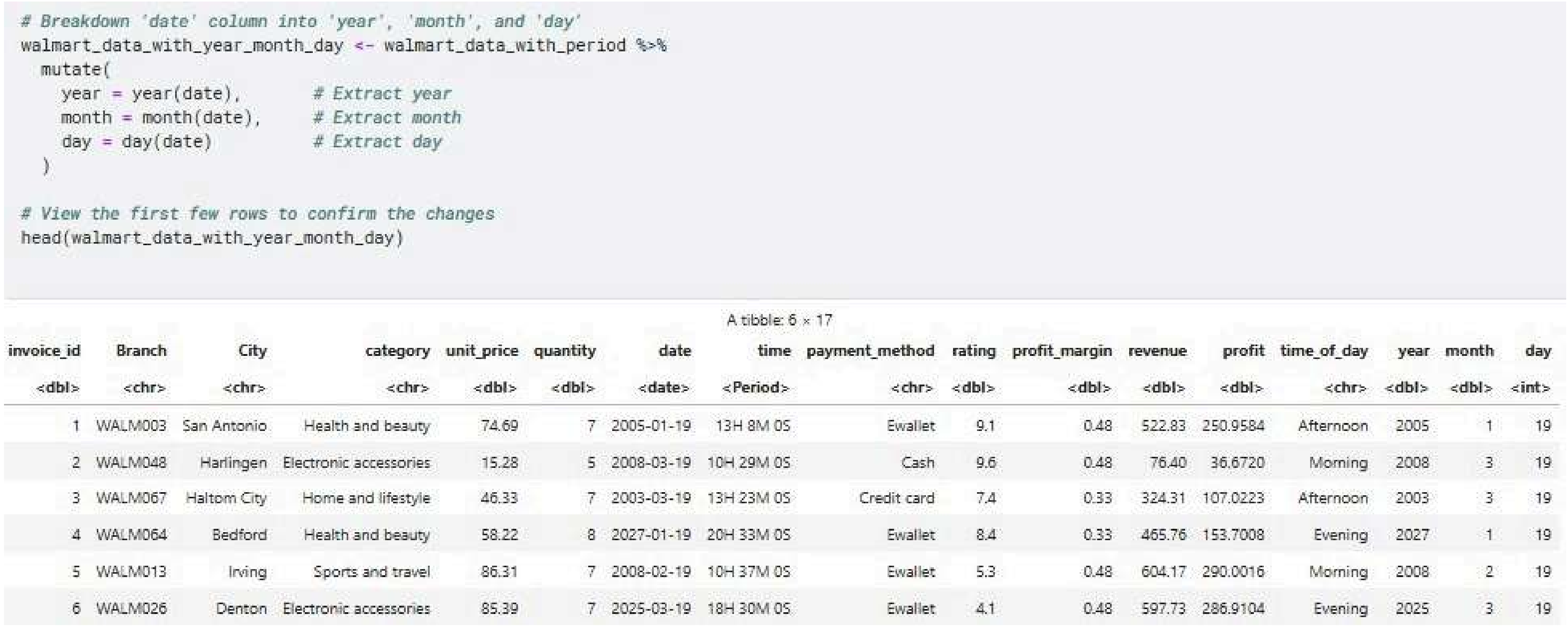
A tibble: 5 × 11

invoice_id	Branch	City	category	unit_price	quantity	date	time	payment_method	rating	profit_margin
<dbl>	<chr>	<chr>	<chr>	<chr>	<dbl>	<date>	<time>	<chr>	<dbl>	<dbl>
1	WALM003	San Antonio	Health and beauty	$74.69	7	2005-01-19	13:08:00	Ewallet	9.1	0.48
2	WALM048	Harlingen	Electronic accessories	$15.28	5	2008-03-19	10:29:00	Cash	9.6	0.48
3	WALM067	Haltom City	Home and lifestyle	$46.33	7	2003-03-19	13:23:00	Credit card	7.4	0.33
4	WALM064	Bedford	Health and beauty	$58.22	8	2027-01-19	20:33:00	Ewallet	8.4	0.33
5	WALM013	Irving	Sports and travel	$86.31	7	2008-02-19	10:37:00	Ewallet	5.3	0.48

A tibble: 5 × 11

invoice_id	Branch	City	category	unit_price	quantity	date	time	payment_method	rating	profit_margin
<dbl>	<chr>	<chr>	<chr>	<chr>	<dbl>	<date>	<time>	<chr>	<dbl>	<dbl>
1	WALM003	San Antonio	Health and beauty	$74.69	7	2005-01-19	13:08:00	Ewallet	9.1	0.48
2	WALM048	Harlingen	Electronic accessories	$15.28	5	2008-03-19	10:29:00	Cash	9.6	0.48
3	WALM067	Haltom City	Home and lifestyle	$46.33	7	2003-03-19	13:23:00	Credit card	7.4	0.33
4	WALM064	Bedford	Health and beauty	$58.22	8	2027-01-19	20:33:00	Ewallet	8.4	0.33
5	WALM013	Irving	Sports and travel	$86.31	7	2008-02-19	10:37:00	Ewallet	5.3	0.48

VI. Data Preparation for Time-bound Analysis

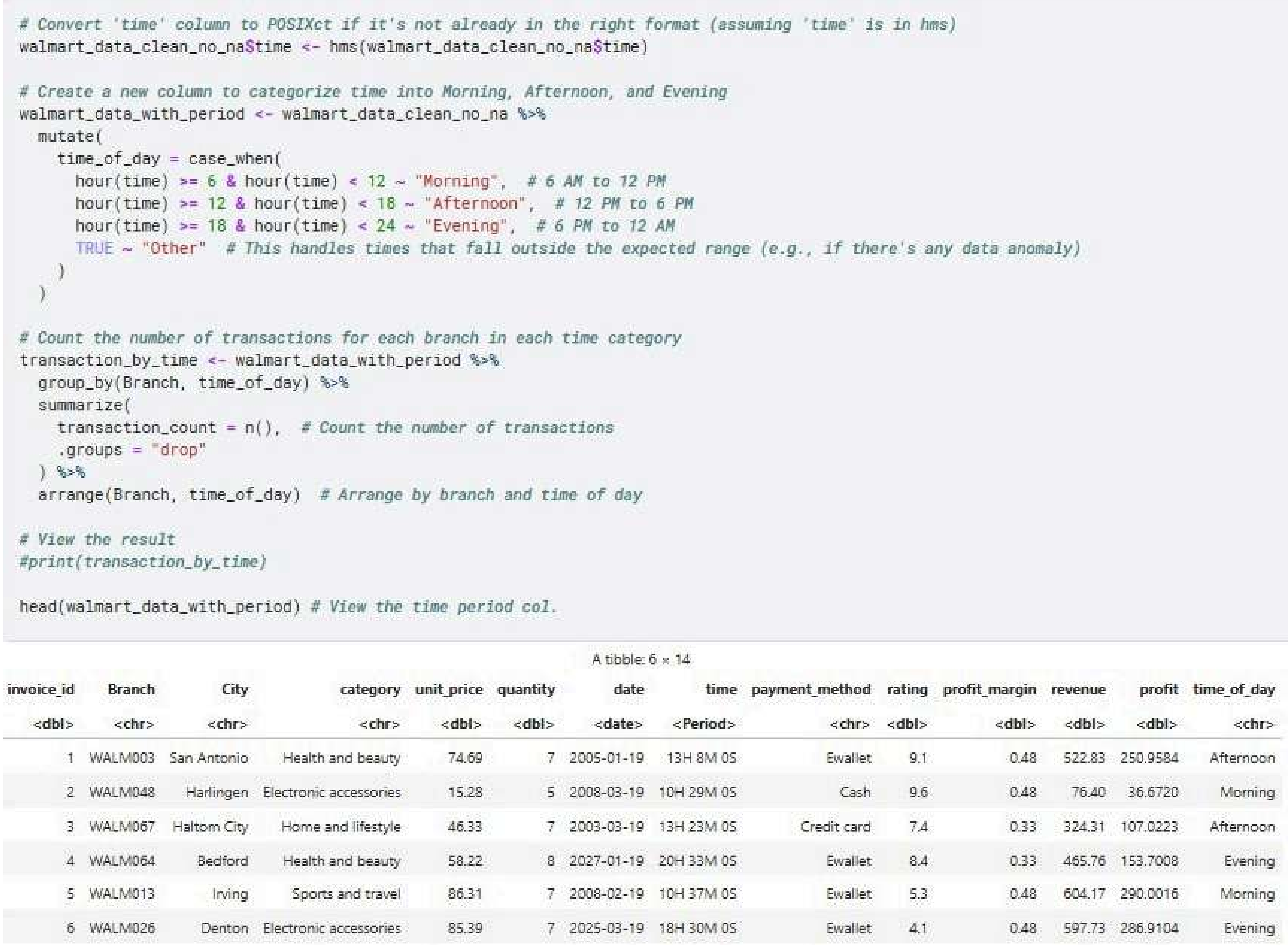


```
# Breakdown 'date' column into 'year', 'month', and 'day'
walmart_data_with_year_month_day <- walmart_data_with_period %>%
  mutate(
    year = year(date),       # Extract year
    month = month(date),     # Extract month
    day = day(date)          # Extract day
  )

# View the first few rows to confirm the changes
head(walmart_data_with_year_month_day)
```

A tibble: 6 × 17

invoice_id	Branch	City	category	unit_price	quantity	date	time	payment_method	rating	profit_margin	revenue	profit	time_of_day	year	month	day
<dbl>	<chr>	<chr>	<chr>	<dbl>	<dbl>	<date>	<Period>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<chr>	<dbl>	<dbl>	<int>
1	WALM003	San Antonio	Health and beauty	74.69	7	2005-01-19	13H 8M 0S	Ewallet	9.1	0.48	522.83	250.9584	Afternoon	2005	1	19
2	WALM048	Harlingen	Electronic accessories	15.28	5	2008-03-19	10H 29M 0S	Cash	9.6	0.48	76.40	36.6720	Morning	2008	3	19
3	WALM067	Haltom City	Home and lifestyle	46.33	7	2003-03-19	13H 23M 0S	Credit card	7.4	0.33	324.31	107.0223	Afternoon	2003	3	19
4	WALM064	Bedford	Health and beauty	58.22	8	2027-01-19	20H 33M 0S	Ewallet	8.4	0.33	465.76	153.7008	Evening	2027	1	19
5	WALM013	Irving	Sports and travel	86.31	7	2008-02-19	10H 37M 0S	Ewallet	5.3	0.48	604.17	290.0016	Morning	2008	2	19
6	WALM026	Denton	Electronic accessories	85.39	7	2025-03-19	18H 30M 0S	Ewallet	4.1	0.48	597.73	286.9104	Evening	2025	3	19

Time (Year Month Day) Breakdown



```r
# Convert 'time' column to POSIXct if it's not already in the right format (assuming 'time' is in hms)
walmart_data_clean_no_na$time <- hms(walmart_data_clean_no_na$time)

# Create a new column to categorize time into Morning, Afternoon, and Evening
walmart_data_with_period <- walmart_data_clean_no_na %>%
  mutate(
    time_of_day = case_when(
      hour(time) >= 6 & hour(time) < 12 ~ "Morning",  # 6 AM to 12 PM
      hour(time) >= 12 & hour(time) < 18 ~ "Afternoon",  # 12 PM to 6 PM
      hour(time) >= 18 & hour(time) < 24 ~ "Evening",  # 6 PM to 12 AM
      TRUE ~ "Other"  # This handles times that fall outside the expected range (e.g., if there's any data anomaly)
    )
  )

# Count the number of transactions for each branch in each time category
transaction_by_time <- walmart_data_with_period %>%
  group_by(Branch, time_of_day) %>%
  summarize(
    transaction_count = n(),  # Count the number of transactions
    .groups = "drop"
  ) %>%
  arrange(Branch, time_of_day)  # Arrange by branch and time of day

# View the result
#print(transaction_by_time)

head(walmart_data_with_period) # View the time period col.
```

A tibble: 6 × 14

invoice_id	Branch	City	category	unit_price	quantity	date	time	payment_method	rating	profit_margin	revenue	profit	time_of_day
<dbl>	<chr>	<chr>	<chr>	<dbl>	<dbl>	<date>	<Period>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<chr>
1	WALM003	San Antonio	Health and beauty	74.69	7	2005-01-19	13H 8M 0S	Ewallet	9.1	0.48	522.83	250.9584	Afternoon
2	WALM048	Harlingen	Electronic accessories	15.28	5	2008-03-19	10H 29M 0S	Cash	9.6	0.48	76.40	36.6720	Morning
3	WALM067	Haltom City	Home and lifestyle	46.33	7	2003-03-19	13H 23M 0S	Credit card	7.4	0.33	324.31	107.0223	Afternoon
4	WALM064	Bedford	Health and beauty	58.22	8	2027-01-19	20H 33M 0S	Ewallet	8.4	0.33	465.76	153.7008	Evening
5	WALM013	Irving	Sports and travel	86.31	7	2008-02-19	10H 37M 0S	Ewallet	5.3	0.48	604.17	290.0016	Morning
6	WALM026	Denton	Electronic accessories	85.39	7	2025-03-19	18H 30M 0S	Ewallet	4.1	0.48	597.73	286.9104	Evening

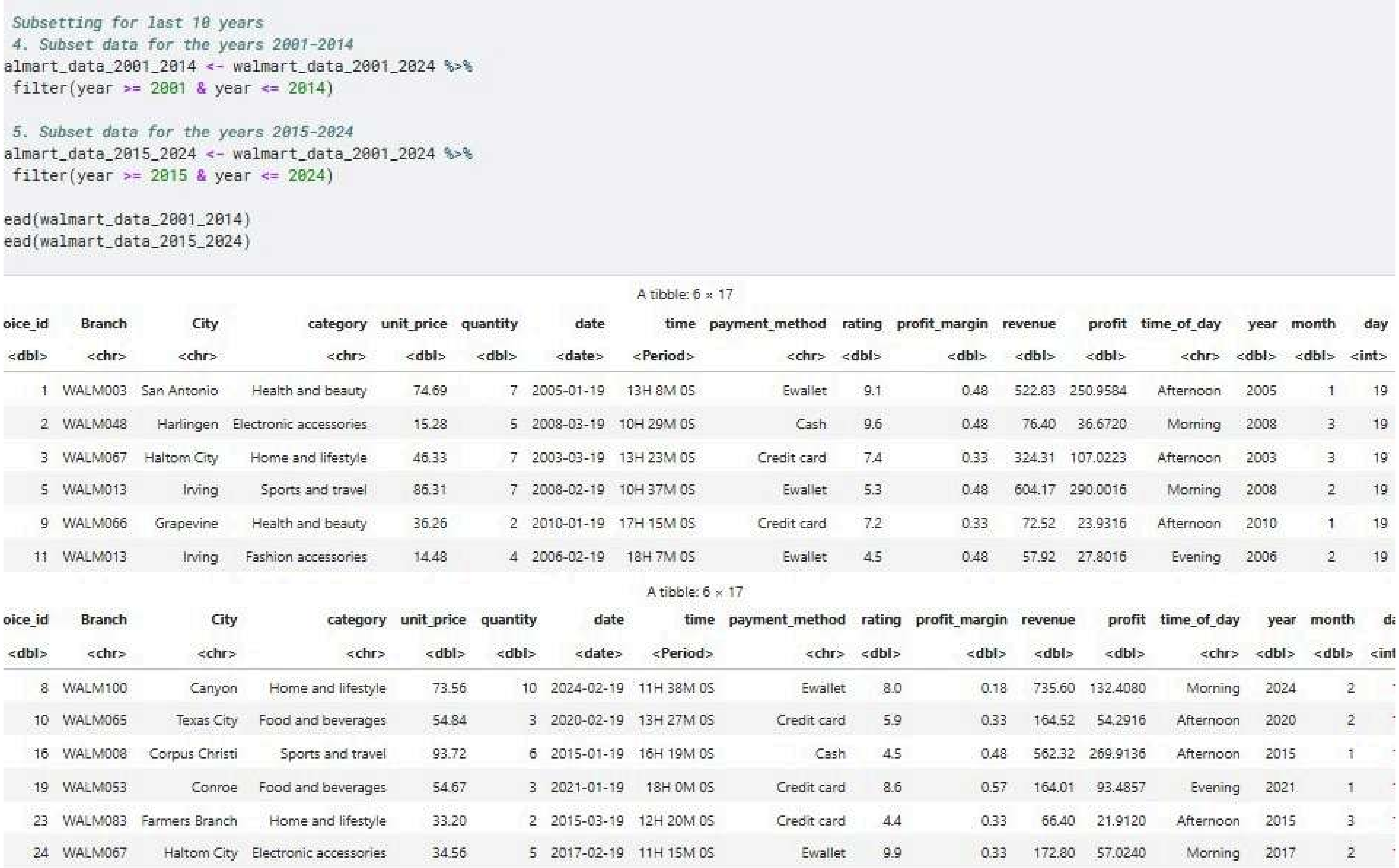
Shift Breakdown:

Morning

Afternoon

Evening

VII. Subsetting for preparing the last 10 years' data



```
Subsetting for last 10 years
 4. Subset data for the years 2001-2014
almart_data_2001_2014 <- walmart_data_2001_2024 %>%
 filter(year >= 2001 & year <= 2014)

 5. Subset data for the years 2015-2024
almart_data_2015_2024 <- walmart_data_2001_2024 %>%
 filter(year >= 2015 & year <= 2024)

ead(walmart_data_2001_2014)
ead(walmart_data_2015_2024)
```

A tibble: 6 × 17

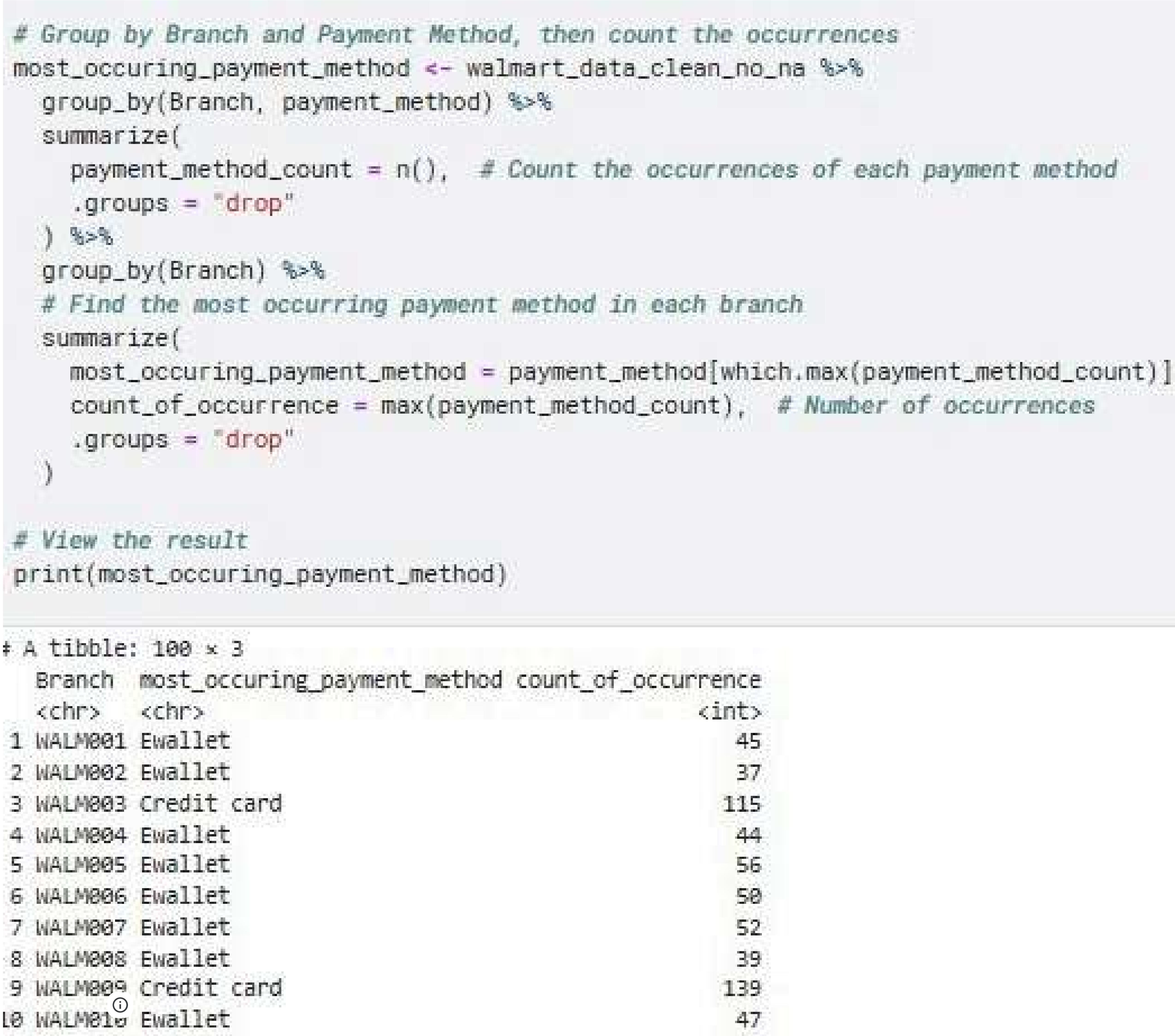
oice_id	Branch	City	category	unit_price	quantity	date	time	payment_method	rating	profit_margin	revenue	profit	time_of_day	year	month	day
<dbl>	<chr>	<chr>	<chr>	<dbl>	<dbl>	<date>	<Period>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<chr>	<dbl>	<dbl>	<int>
1	WALM003	San Antonio	Health and beauty	74.69	7	2005-01-19	13H 8M 0S	Ewallet	9.1	0.48	522.83	250.9584	Afternoon	2005	1	19
2	WALM048	Harlingen	Electronic accessories	15.28	5	2008-03-19	10H 29M 0S	Cash	9.6	0.48	76.40	36.6720	Morning	2008	3	19
3	WALM067	Haltom City	Home and lifestyle	46.33	7	2003-03-19	13H 23M 0S	Credit card	7.4	0.33	324.31	107.0223	Afternoon	2003	3	19
5	WALM013	Irving	Sports and travel	86.31	7	2008-02-19	10H 37M 0S	Ewallet	5.3	0.48	604.17	290.0016	Morning	2008	2	19
9	WALM066	Grapevine	Health and beauty	36.26	2	2010-01-19	17H 15M 0S	Credit card	7.2	0.33	72.52	23.9316	Afternoon	2010	1	19
11	WALM013	Irving	Fashion accessories	14.48	4	2006-02-19	18H 7M 0S	Ewallet	4.5	0.48	57.92	27.8016	Evening	2006	2	19

A tibble: 6 × 17

oice_id	Branch	City	category	unit_price	quantity	date	time	payment_method	rating	profit_margin	revenue	profit	time_of_day	year	month	da
<dbl>	<chr>	<chr>	<chr>	<dbl>	<dbl>	<date>	<Period>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<chr>	<dbl>	<dbl>	<int
8	WALM100	Canyon	Home and lifestyle	73.56	10	2024-02-19	11H 38M 0S	Ewallet	8.0	0.18	735.60	132.4080	Morning	2024	2	1
10	WALM065	Texas City	Food and beverages	54.84	3	2020-02-19	13H 27M 0S	Credit card	5.9	0.33	164.52	54.2916	Afternoon	2020	2	1
16	WALM008	Corpus Christi	Sports and travel	93.72	6	2015-01-19	16H 19M 0S	Cash	4.5	0.48	562.32	269.9136	Afternoon	2015	1	1
19	WALM053	Conroe	Food and beverages	54.67	3	2021-01-19	18H 0M 0S	Credit card	8.6	0.57	164.01	93.4857	Evening	2021	1	1
23	WALM083	Farmers Branch	Home and lifestyle	33.20	2	2015-03-19	12H 20M 0S	Credit card	4.4	0.33	66.40	21.9120	Afternoon	2015	3	1
24	WALM067	Haltom City	Electronic accessories	34.56	5	2017-02-19	11H 15M 0S	Ewallet	9.9	0.33	172.80	57.0240	Morning	2017	2	1

VIII. Presentation

A. Branches General Scenario

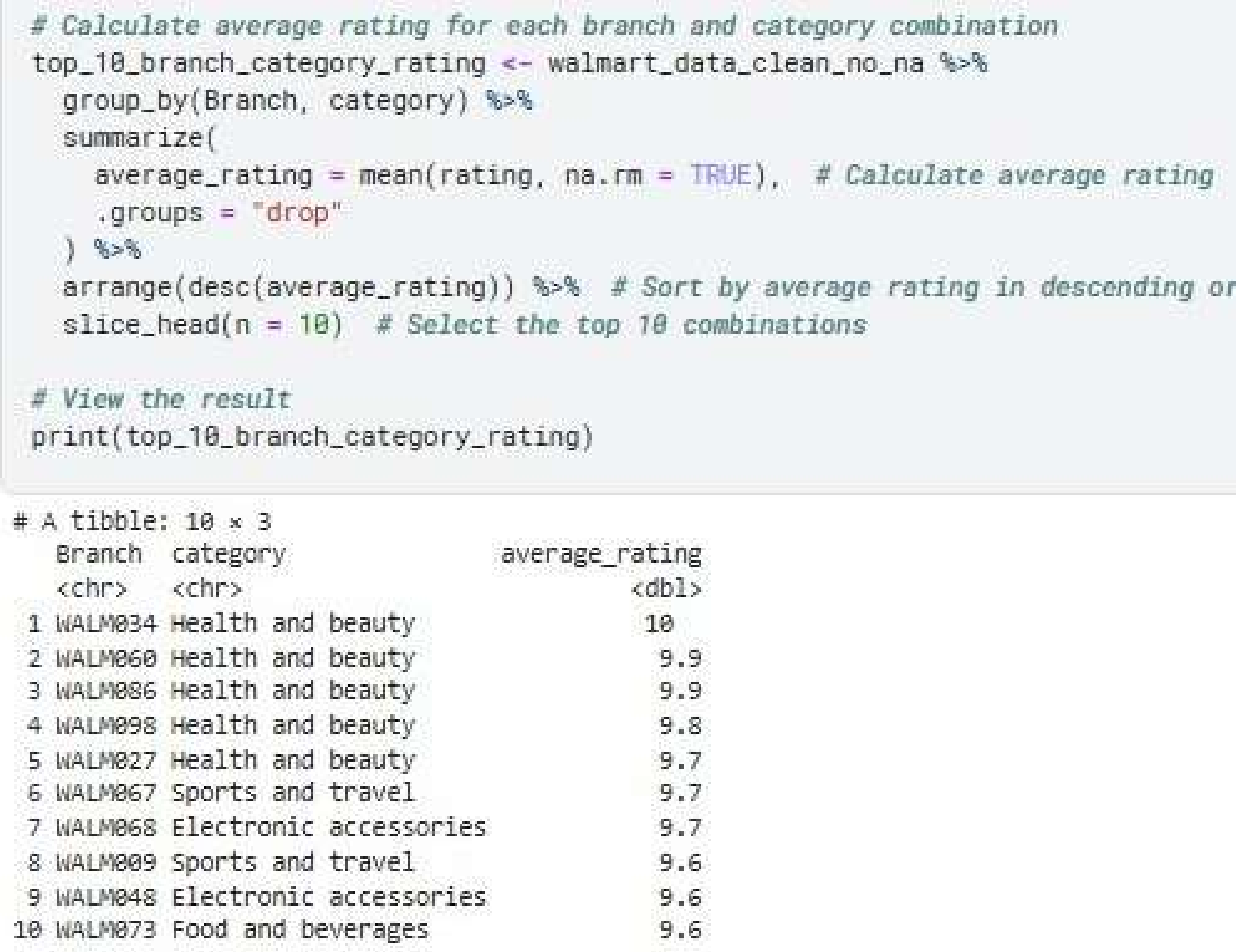


```
# Group by Branch and Payment Method, then count the occurrences
most_occuring_payment_method <- walmart_data_clean_no_na %>%
  group_by(Branch, payment_method) %>%
  summarize(
    payment_method_count = n(),  # Count the occurrences of each payment method
    .groups = "drop"
  ) %>%
  group_by(Branch) %>%
  # Find the most occurring payment method in each branch
  summarize(
    most_occuring_payment_method = payment_method[which.max(payment_method_count)]
    count_of_occurrence = max(payment_method_count),  # Number of occurrences
    .groups = "drop"
  )

# View the result
print(most_occuring_payment_method)
```

```
# A tibble: 100 × 3
   Branch  most_occuring_payment_method count_of_occurrence
   <chr>   <chr>                                      <int>
 1 WALM001 Ewallet                                       45
 2 WALM002 Ewallet                                       37
 3 WALM003 Credit card                                  115
 4 WALM004 Ewallet                                       44
 5 WALM005 Ewallet                                       56
 6 WALM006 Ewallet                                       50
 7 WALM007 Ewallet                                       52
 8 WALM008 Ewallet                                       39
 9 WALM009 Credit card                                  139
10 WALM010 Ewallet                                       47
```

```
# i 90 more rows
```

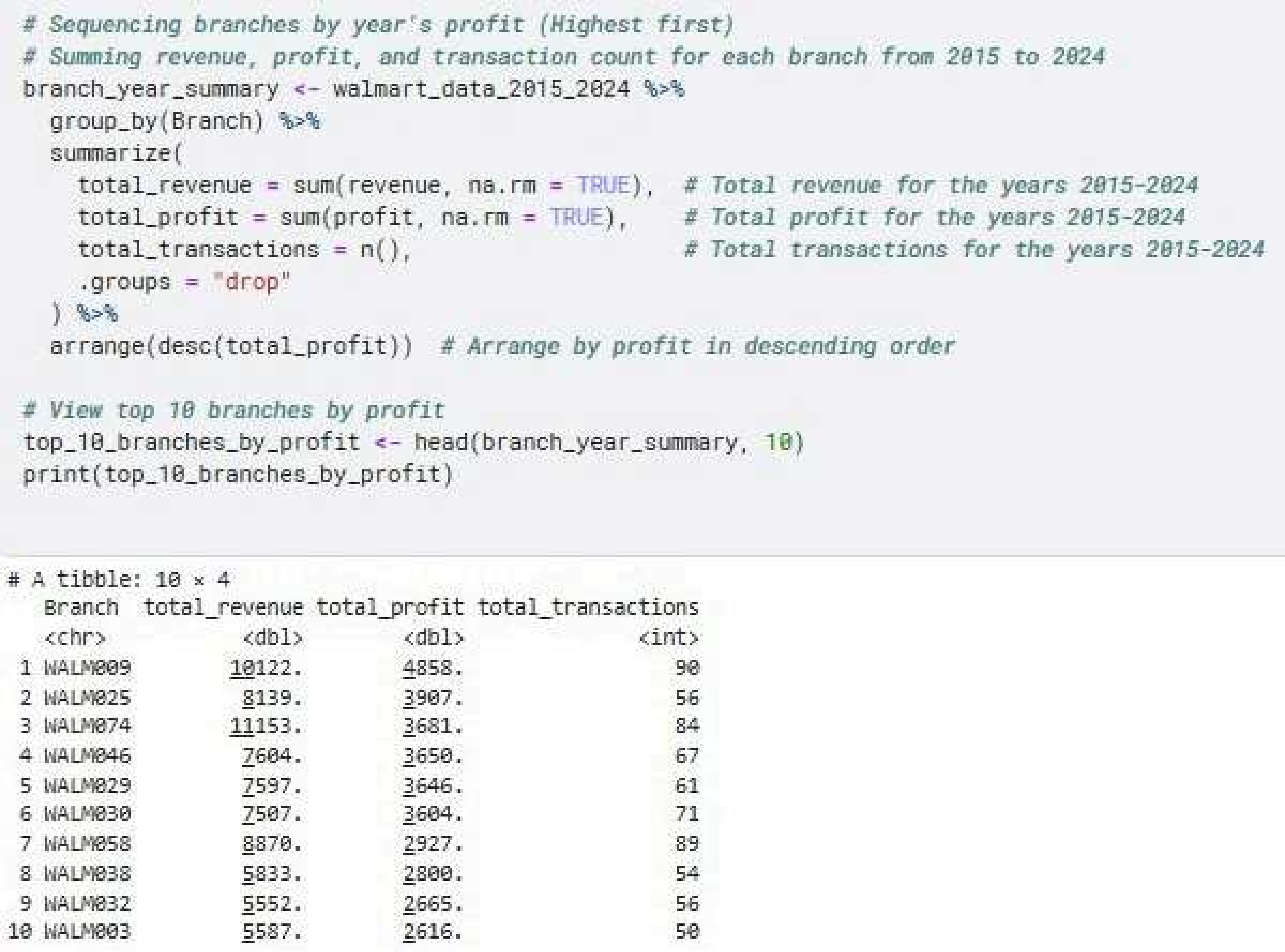
Payment Method Per Branches



```
# Calculate average rating for each branch and category combination
top_10_branch_category_rating <- walmart_data_clean_no_na %>%
  group_by(Branch, category) %>%
  summarize(
    average_rating = mean(rating, na.rm = TRUE),  # Calculate average rating
    .groups = "drop"
  ) %>%
  arrange(desc(average_rating)) %>%  # Sort by average rating in descending or
  slice_head(n = 10)  # Select the top 10 combinations

# View the result
print(top_10_branch_category_rating)
```

```
# A tibble: 10 × 3
   Branch  category               average_rating
   <chr>   <chr>                           <dbl>
 1 WALM034 Health and beauty                10  
 2 WALM060 Health and beauty                 9.9
 3 WALM086 Health and beauty                 9.9
 4 WALM098 Health and beauty                 9.8
 5 WALM027 Health and beauty                 9.7
 6 WALM067 Sports and travel                 9.7
 7 WALM068 Electronic accessories            9.7
 8 WALM009 Sports and travel                 9.6
 9 WALM048 Electronic accessories            9.6
10 WALM073 Food and beverages                9.6
```

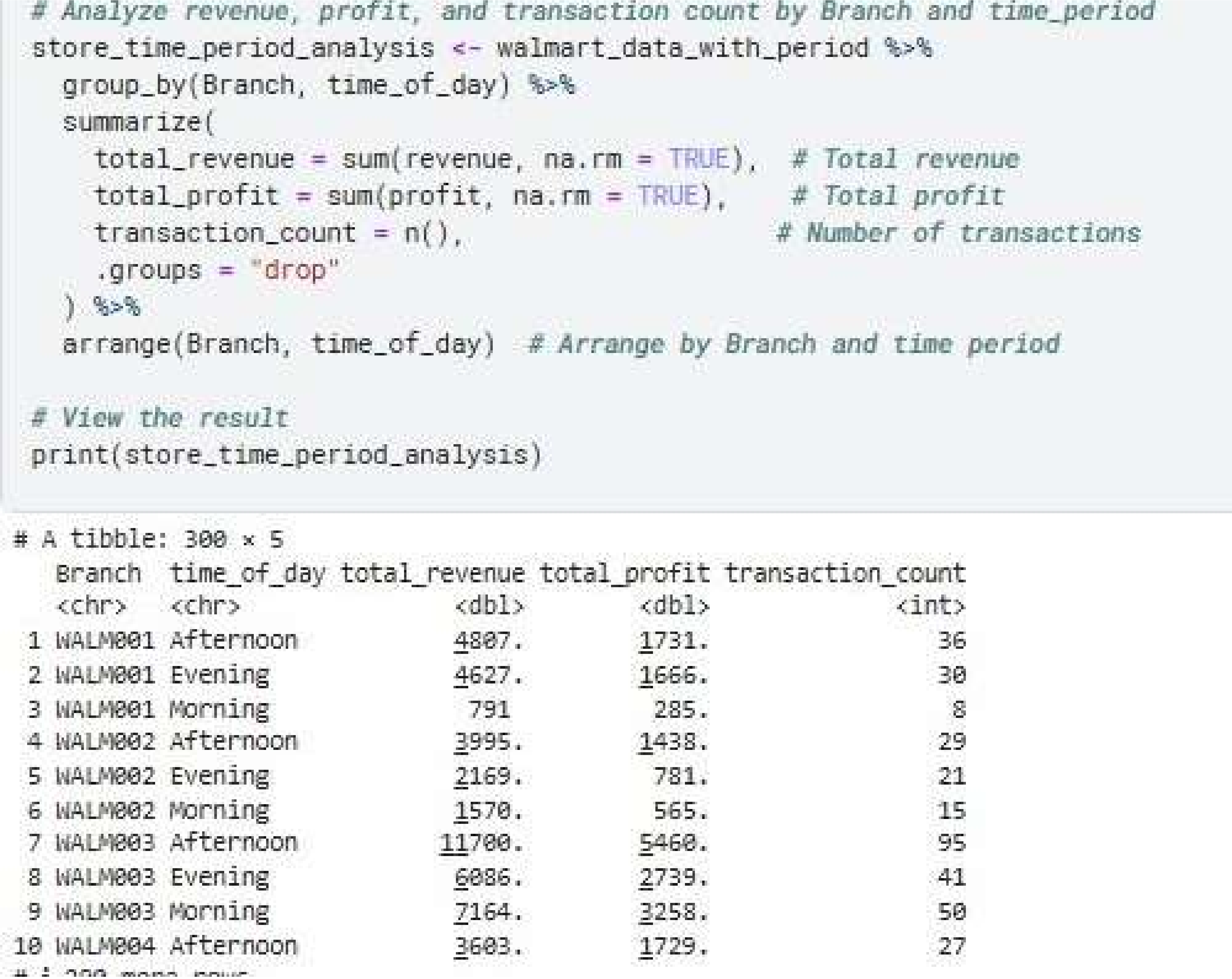
Top Rated Branches



```r
# Sequencing branches by year's profit (Highest first)
# Summing revenue, profit, and transaction count for each branch from 2015 to 2024
branch_year_summary <- walmart_data_2015_2024 %>%
  group_by(Branch) %>%
  summarize(
    total_revenue = sum(revenue, na.rm = TRUE),  # Total revenue for the years 2015-2024
    total_profit = sum(profit, na.rm = TRUE),    # Total profit for the years 2015-2024
    total_transactions = n(),                    # Total transactions for the years 2015-2024
    .groups = "drop"
  ) %>%
  arrange(desc(total_profit))  # Arrange by profit in descending order

# View top 10 branches by profit
top_10_branches_by_profit <- head(branch_year_summary, 10)
print(top_10_branches_by_profit)
```

```
# A tibble: 10 × 4
   Branch  total_revenue total_profit total_transactions
   <chr>           <dbl>        <dbl>              <int>
 1 WALM009        10122.        4858.                 90
 2 WALM025         8139.        3907.                 56
 3 WALM074        11153.        3681.                 84
 4 WALM046         7604.        3650.                 67
 5 WALM029         7597.        3646.                 61
 6 WALM030         7507.        3604.                 71
 7 WALM058         8870.        2927.                 89
 8 WALM038         5833.        2800.                 54
 9 WALM032         5552.        2665.                 56
10 WALM003         5587.        2616.                 50
```

Top 10 Earners in Last 10 Years

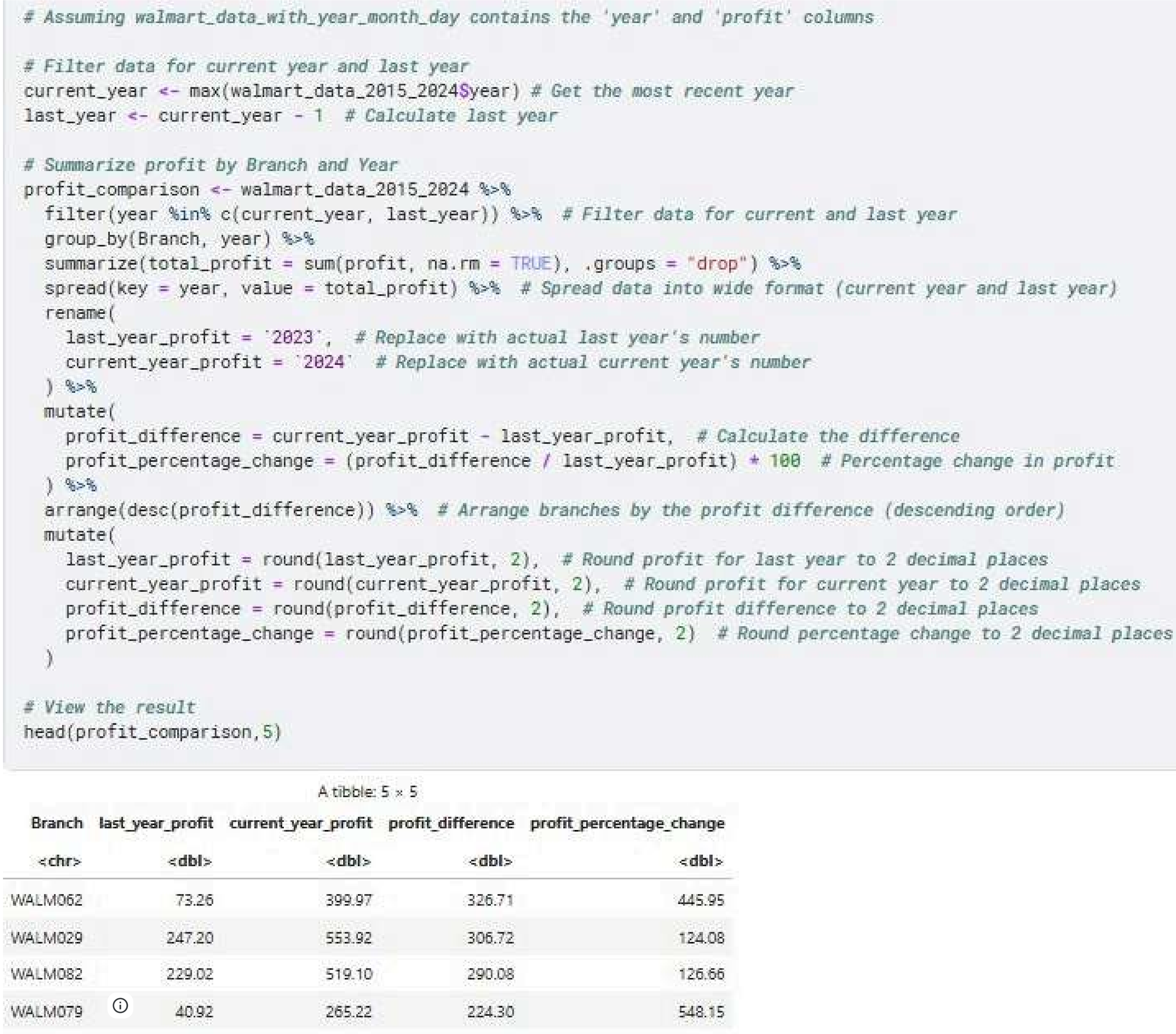


```r
# Analyze revenue, profit, and transaction count by Branch and time_period
store_time_period_analysis <- walmart_data_with_period %>%
  group_by(Branch, time_of_day) %>%
  summarize(
    total_revenue = sum(revenue, na.rm = TRUE),  # Total revenue
    total_profit = sum(profit, na.rm = TRUE),    # Total profit
    transaction_count = n(),                    # Number of transactions
    .groups = "drop"
  ) %>%
  arrange(Branch, time_of_day)  # Arrange by Branch and time period

# View the result
print(store_time_period_analysis)
```

```
# A tibble: 300 × 5
   Branch  time_of_day total_revenue total_profit transaction_count
   <chr>   <chr>               <dbl>        <dbl>             <int>
 1 WALM001 Afternoon           4807.        1731.                36
 2 WALM001 Evening             4627.        1666.                30
 3 WALM001 Morning              791          285.                 8
 4 WALM002 Afternoon           3995.        1438.                29
 5 WALM002 Evening             2169.         781.                21
 6 WALM002 Morning             1570.         565.                15
 7 WALM003 Afternoon          11700.        5460.                95
 8 WALM003 Evening             6086.        2739.                41
 9 WALM003 Morning             7164.        3258.                50
10 WALM004 Afternoon           3603.        1729.                27
# ℹ 290 more rows
```

Earning Per Shift

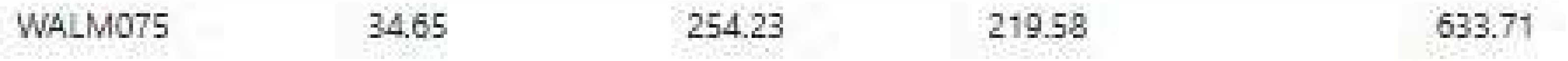
C. Branches E ing Scenario : 10 Year Period



```
# Assuming walmart_data_with_year_month_day contains the 'year' and 'profit' columns

# Filter data for current year and last year
current_year <- max(walmart_data_2015_2024$year) # Get the most recent year
last_year <- current_year - 1  # Calculate last year

# Summarize profit by Branch and Year
profit_comparison <- walmart_data_2015_2024 %>%
  filter(year %in% c(current_year, last_year)) %>%  # Filter data for current and last year
  group_by(Branch, year) %>%
  summarize(total_profit = sum(profit, na.rm = TRUE), .groups = "drop") %>%
  spread(key = year, value = total_profit) %>%  # Spread data into wide format (current year and last year)
  rename(
    last_year_profit = `2023`,  # Replace with actual last year's number
    current_year_profit = `2024`  # Replace with actual current year's number
  ) %>%
  mutate(
    profit_difference = current_year_profit - last_year_profit,  # Calculate the difference
    profit_percentage_change = (profit_difference / last_year_profit) * 100  # Percentage change in profit
  ) %>%
  arrange(desc(profit_difference)) %>%  # Arrange branches by the profit difference (descending order)
  mutate(
    last_year_profit = round(last_year_profit, 2),  # Round profit for last year to 2 decimal places
    current_year_profit = round(current_year_profit, 2),  # Round profit for current year to 2 decimal places
    profit_difference = round(profit_difference, 2),  # Round profit difference to 2 decimal places
    profit_percentage_change = round(profit_percentage_change, 2)  # Round percentage change to 2 decimal places
  )

# View the result
head(profit_comparison,5)
```

A tibble: 5 × 5

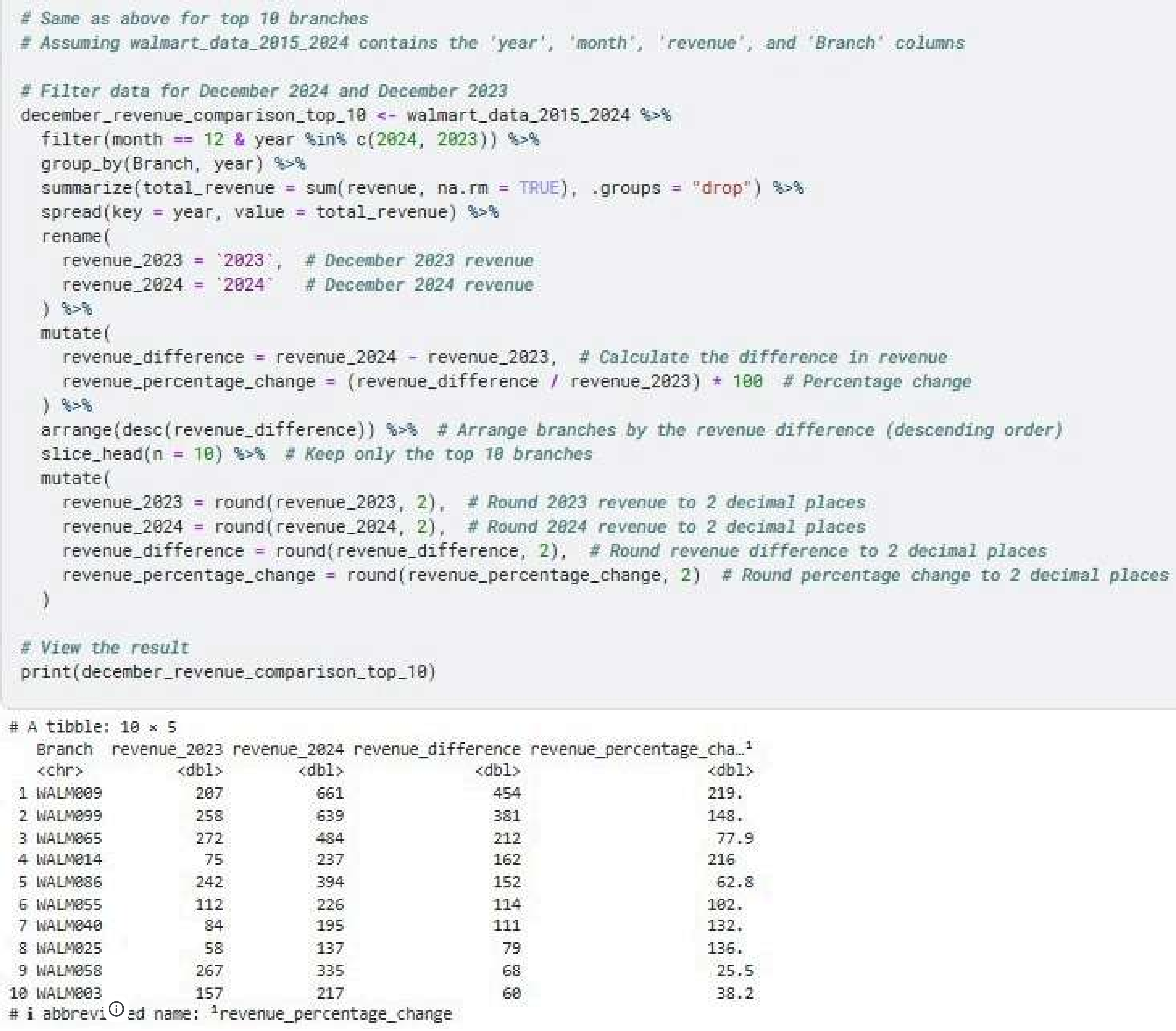
Branch	last_year_profit	current_year_profit	profit_difference	profit_percentage_change
<chr>	<dbl>	<dbl>	<dbl>	<dbl>
WALM062	73.26	399.97	326.71	445.95
WALM029	247.20	553.92	306.72	124.08
WALM082	229.02	519.10	290.08	126.66
WALM079	40.92	265.22	224.30	548.15



WALM075	34.65	254.23	219.58	633.71

i. Year Over Year:

Comparing Profit in relation to last year



```
# Same as above for top 10 branches
# Assuming walmart_data_2015_2024 contains the 'year', 'month', 'revenue', and 'Branch' columns

# Filter data for December 2024 and December 2023
december_revenue_comparison_top_10 <- walmart_data_2015_2024 %>%
  filter(month == 12 & year %in% c(2024, 2023)) %>%
  group_by(Branch, year) %>%
  summarize(total_revenue = sum(revenue, na.rm = TRUE), .groups = "drop") %>%
  spread(key = year, value = total_revenue) %>%
  rename(
    revenue_2023 = `2023`,  # December 2023 revenue
    revenue_2024 = `2024`   # December 2024 revenue
  ) %>%
  mutate(
    revenue_difference = revenue_2024 - revenue_2023,  # Calculate the difference in revenue
    revenue_percentage_change = (revenue_difference / revenue_2023) * 100  # Percentage change
  ) %>%
  arrange(desc(revenue_difference)) %>%  # Arrange branches by the revenue difference (descending order)
  slice_head(n = 10) %>%  # Keep only the top 10 branches
  mutate(
    revenue_2023 = round(revenue_2023, 2),  # Round 2023 revenue to 2 decimal places
    revenue_2024 = round(revenue_2024, 2),  # Round 2024 revenue to 2 decimal places
    revenue_difference = round(revenue_difference, 2),  # Round revenue difference to 2 decimal places
    revenue_percentage_change = round(revenue_percentage_change, 2)  # Round percentage change to 2 decimal places
  )

# View the result
print(december_revenue_comparison_top_10)
```

```
# A tibble: 10 × 5
   Branch  revenue_2023 revenue_2024 revenue_difference revenue_percentage_cha…¹
   <chr>          <dbl>        <dbl>              <dbl>                    <dbl>
 1 WALM009          207          661                454                    219.
 2 WALM099          258          639                381                    148.
 3 WALM065          272          484                212                     77.9
 4 WALM014           75          237                162                    216
 5 WALM086          242          394                152                     62.8
 6 WALM055          112          226                114                    102.
 7 WALM040           84          195                111                    132.
 8 WALM025           58          137                 79                    136.
 9 WALM058          267          335                 68                     25.5
10 WALM003          157          217                 60                     38.2
# ℹ abbrevi ed name: ¹revenue_percentage_change
```

ii. Month Over Month:

Comparing Revenue in relation to December last year

For feedback: *kiran1.paudel2@gmail.com* | [Address] | [Phone number]