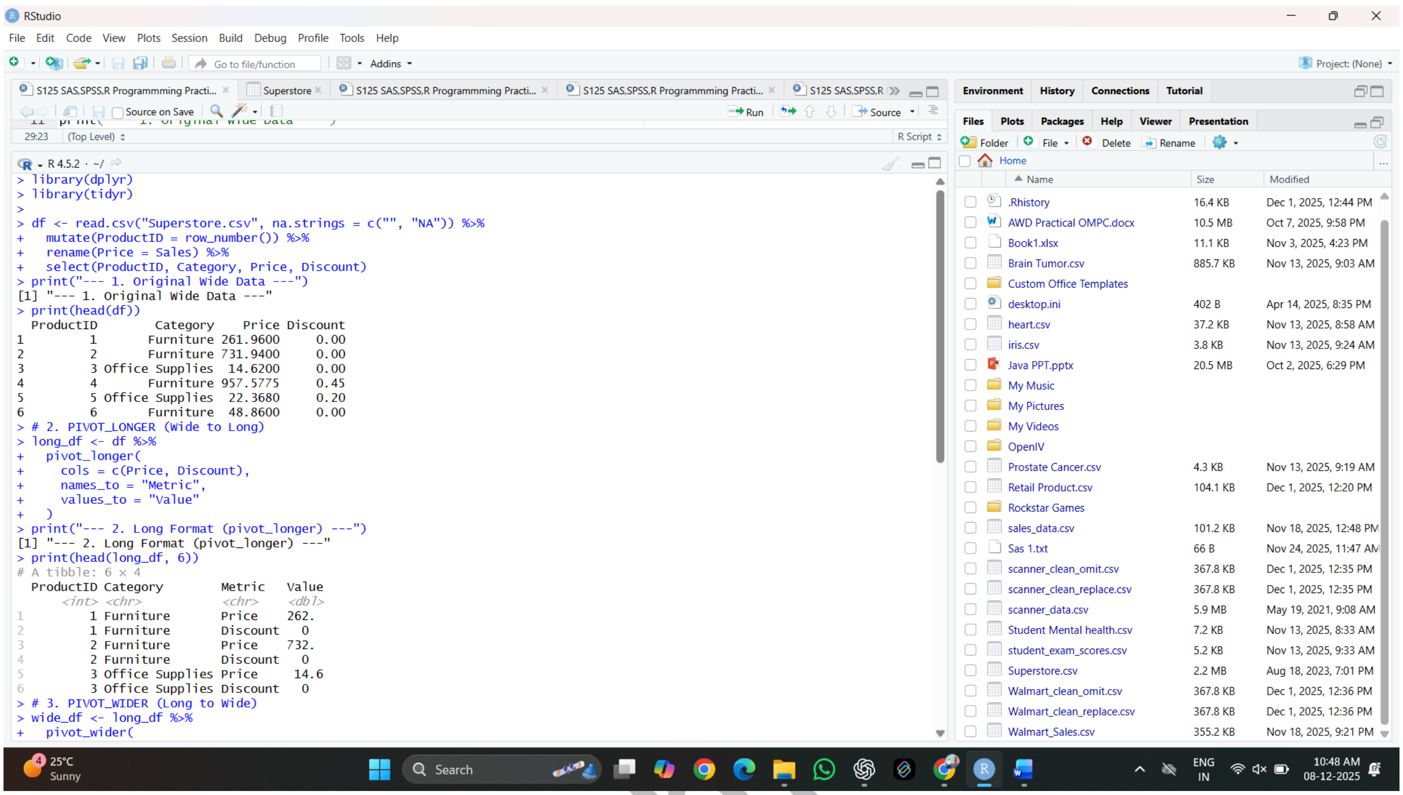
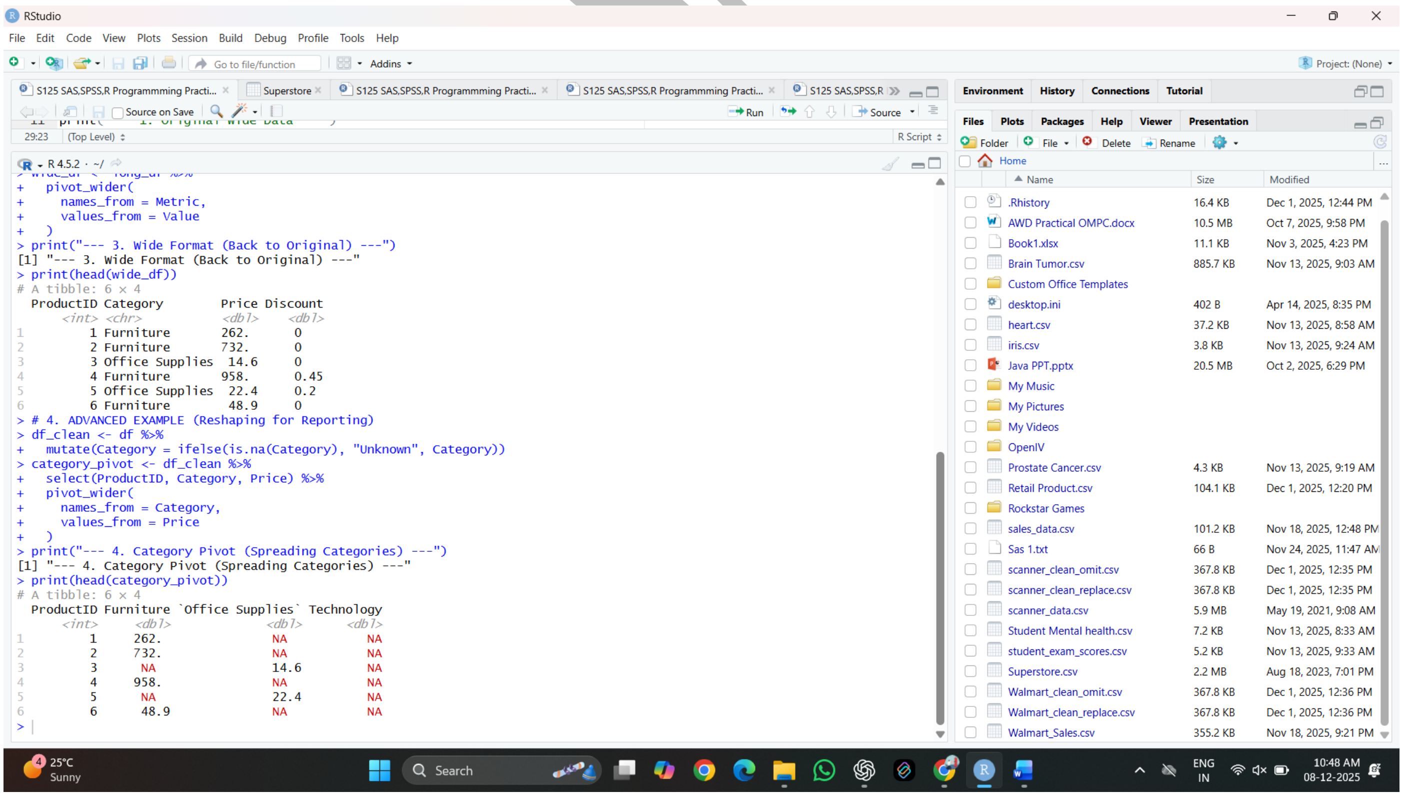
Sheth L.U.J & Sir M.V College
SAS/SPSS/R Programming
Practical No. 11 to 15

Aim: 11. Reshaping data using pivot_longer() and pivot_wider() (R).



```
> library(dplyr)
> library(tidyr)
>
> df <- read.csv("Superstore.csv", na.strings = c("", "NA")) %>%
+   mutate(ProductID = row_number()) %>%
+   rename(Price = Sales) %>%
+   select(ProductID, Category, Price, Discount)
> print("--- 1. Original Wide Data ---")
[1] "--- 1. Original Wide Data ---"
> print(head(df))
  ProductID        Category    Price Discount
1         1       Furniture 261.9600     0.00
2         2       Furniture 731.9400     0.00
3         3 Office Supplies  14.6200     0.00
4         4       Furniture 957.5775     0.45
5         5 Office Supplies  22.3680     0.20
6         6       Furniture  48.8600     0.00
> # 2. PIVOT_LONGER (Wide to Long)
> long_df <- df %>%
+   pivot_longer(
+     cols = c(Price, Discount),
+     names_to = "Metric",
+     values_to = "Value"
+   )
> print("--- 2. Long Format (pivot_longer) ---")
[1] "--- 2. Long Format (pivot_longer) ---"
> print(head(long_df, 6))
# A tibble: 6 × 4
  ProductID Category        Metric   Value
      <int> <chr>           <chr>    <dbl>
1         1 Furniture       Price    262.
2         1 Furniture       Discount   0
3         2 Furniture       Price    732.
4         2 Furniture       Discount   0
5         3 Office Supplies Price     14.6
6         3 Office Supplies Discount   0
> # 3. PIVOT_WIDER (Long to Wide)
> wide_df <- long_df %>%
+   pivot_wider(
```

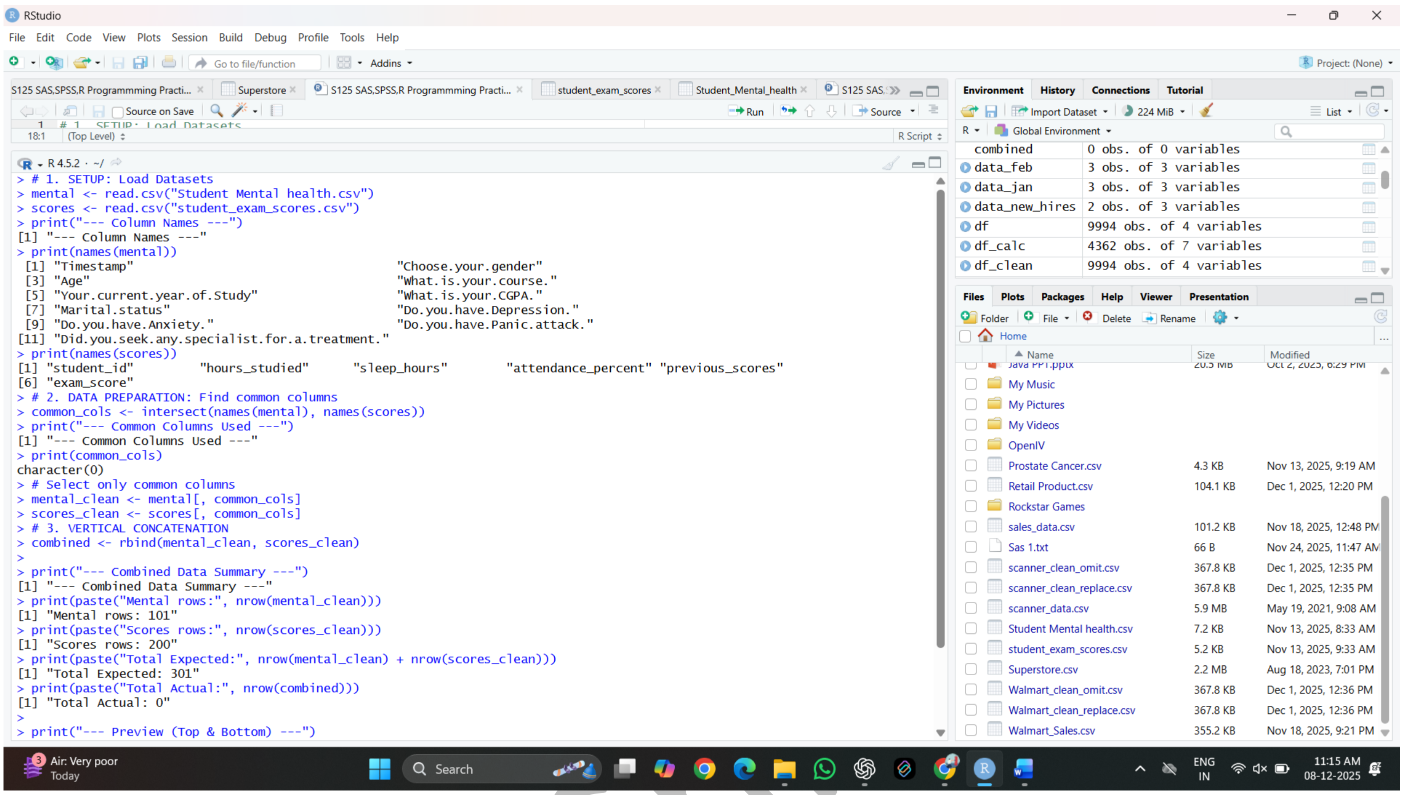


```
+   pivot_wider(
+     names_from = Metric,
+     values_from = Value
+   )
> print("--- 3. Wide Format (Back to Original) ---")
[1] "--- 3. Wide Format (Back to Original) ---"
> print(head(wide_df))
# A tibble: 6 × 4
  ProductID Category        Price Discount
      <int> <chr>           <dbl>    <dbl>
1         1 Furniture       262.      0
2         2 Furniture       732.      0
3         3 Office Supplies  14.6     0
4         4 Furniture       958.      0.45
5         5 Office Supplies  22.4     0.2
6         6 Furniture        48.9     0
> # 4. ADVANCED EXAMPLE (Reshaping for Reporting)
> df_clean <- df %>%
+   mutate(Category = ifelse(is.na(Category), "Unknown", Category))
> category_pivot <- df_clean %>%
+   select(ProductID, Category, Price) %>%
+   pivot_wider(
+     names_from = Category,
+     values_from = Price
+   )
> print("--- 4. Category Pivot (Spreading Categories) ---")
[1] "--- 4. Category Pivot (Spreading Categories) ---"
> print(head(category_pivot))
# A tibble: 6 × 4
  ProductID Furniture `Office Supplies` Technology
      <int>     <dbl>             <dbl>      <dbl>
1         1     262.               NA           NA
2         2     732.               NA           NA
3         3      NA                14.6         NA
4         4     958.               NA           NA
5         5      NA                22.4         NA
6         6      48.9              NA           NA
> |
```

Jeetesh Shirganoor
S125

Sheth L.U.J & Sir M.V College
SAS/SPSS/R Programming
Practical No. 11 to 15

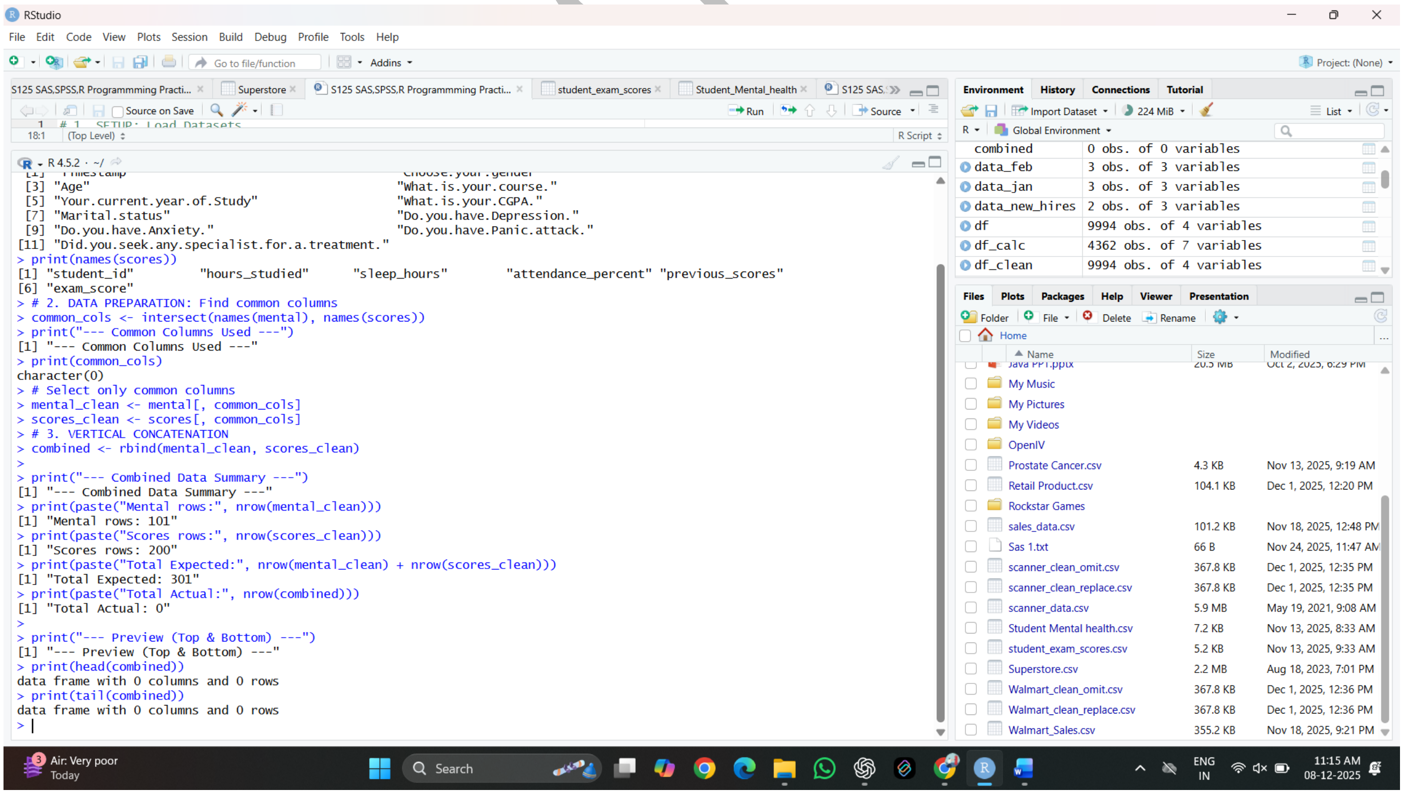
Aim: 12. Combining datasets vertically (concatenation) using rbind() (R).



```
> # 1. SETUP: Load Datasets
> mental <- read.csv("Student Mental health.csv")
> scores <- read.csv("student_exam_scores.csv")
> print("--- Column Names ---")
[1] "--- Column Names ---"
> print(names(mental))
 [1] "Timestamp"                                "Choose.your.gender"
 [3] "Age"                                      "What.is.your.course."
 [5] "Your.current.year.of.Study"               "What.is.your.CGPA."
 [7] "Marital.status"                           "Do.you.have.Depression."
 [9] "Do.you.have.Anxiety."                     "Do.you.have.Panic.attack."
[11] "Did.you.seek.any.specialist.for.a.treatment."
> print(names(scores))
[1] "student_id"         "hours_studied"      "sleep_hours"        "attendance_percent" "previous_scores"
[6] "exam_score"
> # 2. DATA PREPARATION: Find common columns
> common_cols <- intersect(names(mental), names(scores))
> print("--- Common Columns Used ---")
[1] "--- Common Columns Used ---"
> print(common_cols)
character(0)
> # Select only common columns
> mental_clean <- mental[, common_cols]
> scores_clean <- scores[, common_cols]
> # 3. VERTICAL CONCATENATION
> combined <- rbind(mental_clean, scores_clean)
>
> print("--- Combined Data Summary ---")
[1] "--- Combined Data Summary ---"
> print(paste("Mental rows:", nrow(mental_clean)))
[1] "Mental rows: 101"
> print(paste("Scores rows:", nrow(scores_clean)))
[1] "Scores rows: 200"
> print(paste("Total Expected:", nrow(mental_clean) + nrow(scores_clean)))
[1] "Total Expected: 301"
> print(paste("Total Actual:", nrow(combined)))
[1] "Total Actual: 0"
>
> print("--- Preview (Top & Bottom) ---")
```

```
combined        0 obs. of 0 variables
data_feb        3 obs. of 3 variables
data_jan        3 obs. of 3 variables
data_new_hires  2 obs. of 3 variables
df              9994 obs. of 4 variables
df_calc         4362 obs. of 7 variables
df_clean        9994 obs. of 4 variables
```

Name	Size	Modified
My Music		
My Pictures		
My Videos		
OpenIV		
Prostate Cancer.csv	4.3 KB	Nov 13, 2025, 9:19 AM
Retail Product.csv	104.1 KB	Dec 1, 2025, 12:20 PM
Rockstar Games		
sales_data.csv	101.2 KB	Nov 18, 2025, 12:48 PM
Sas 1.txt	66 B	Nov 24, 2025, 11:47 AM
scanner_clean_omit.csv	367.8 KB	Dec 1, 2025, 12:35 PM
scanner_clean_replace.csv	367.8 KB	Dec 1, 2025, 12:35 PM
scanner_data.csv	5.9 MB	May 19, 2021, 9:08 AM
Student Mental health.csv	7.2 KB	Nov 13, 2025, 8:33 AM
student_exam_scores.csv	5.2 KB	Nov 13, 2025, 9:33 AM
Superstore.csv	2.2 MB	Aug 18, 2023, 7:01 PM
Walmart_clean_omit.csv	367.8 KB	Dec 1, 2025, 12:36 PM
Walmart_clean_replace.csv	367.8 KB	Dec 1, 2025, 12:36 PM
Walmart_Sales.csv	355.2 KB	Nov 18, 2025, 9:21 PM



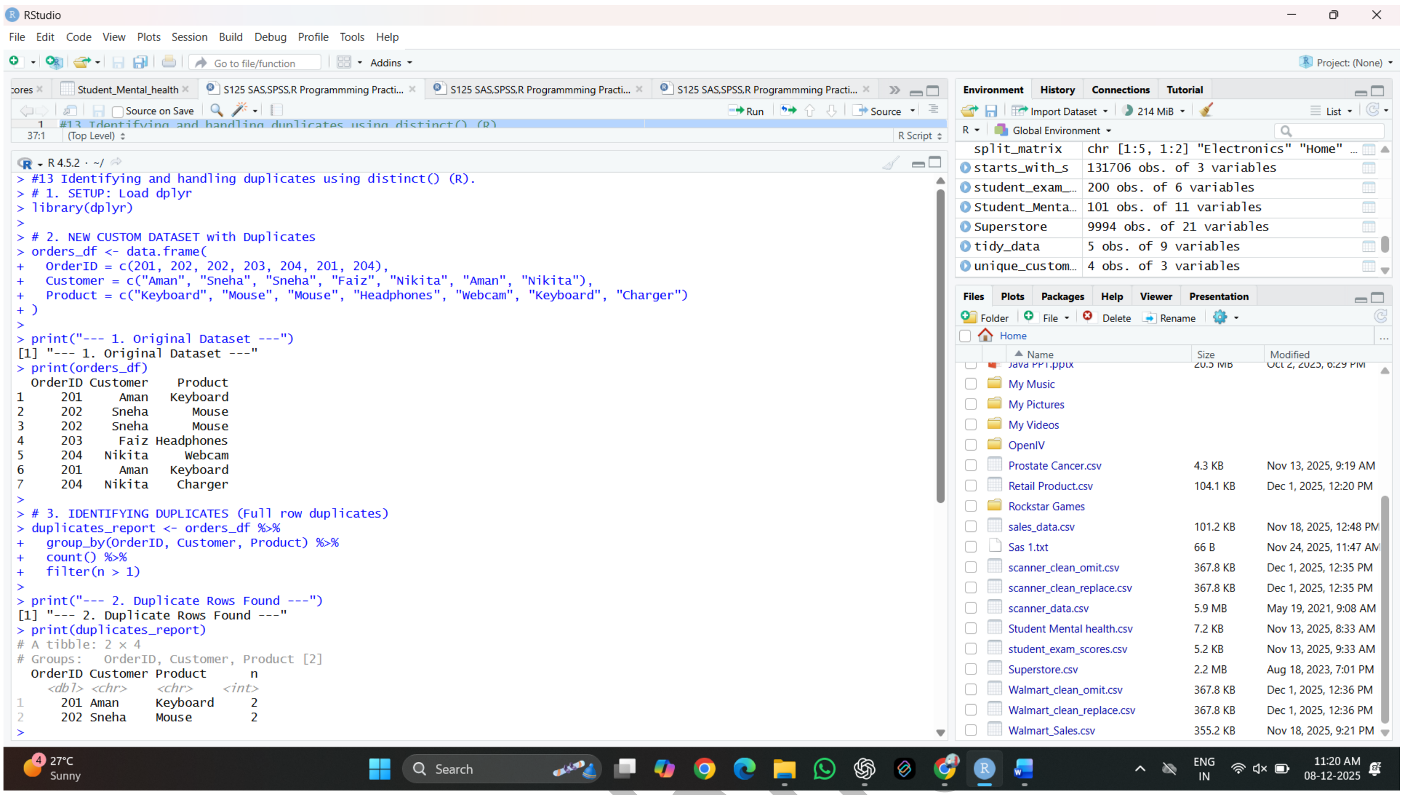
```
 [3] "Age"                                      "What.is.your.course."
 [5] "Your.current.year.of.Study"               "What.is.your.CGPA."
 [7] "Marital.status"                           "Do.you.have.Depression."
 [9] "Do.you.have.Anxiety."                     "Do.you.have.Panic.attack."
[11] "Did.you.seek.any.specialist.for.a.treatment."
> print(names(scores))
[1] "student_id"         "hours_studied"      "sleep_hours"        "attendance_percent" "previous_scores"
[6] "exam_score"
> # 2. DATA PREPARATION: Find common columns
> common_cols <- intersect(names(mental), names(scores))
> print("--- Common Columns Used ---")
[1] "--- Common Columns Used ---"
> print(common_cols)
character(0)
> # Select only common columns
> mental_clean <- mental[, common_cols]
> scores_clean <- scores[, common_cols]
> # 3. VERTICAL CONCATENATION
> combined <- rbind(mental_clean, scores_clean)
>
> print("--- Combined Data Summary ---")
[1] "--- Combined Data Summary ---"
> print(paste("Mental rows:", nrow(mental_clean)))
[1] "Mental rows: 101"
> print(paste("Scores rows:", nrow(scores_clean)))
[1] "Scores rows: 200"
> print(paste("Total Expected:", nrow(mental_clean) + nrow(scores_clean)))
[1] "Total Expected: 301"
> print(paste("Total Actual:", nrow(combined)))
[1] "Total Actual: 0"
>
> print("--- Preview (Top & Bottom) ---")
```

```
[1] "--- Preview (Top & Bottom) ---"
> print(head(combined))
data frame with 0 columns and 0 rows
> print(tail(combined))
data frame with 0 columns and 0 rows
> |
```

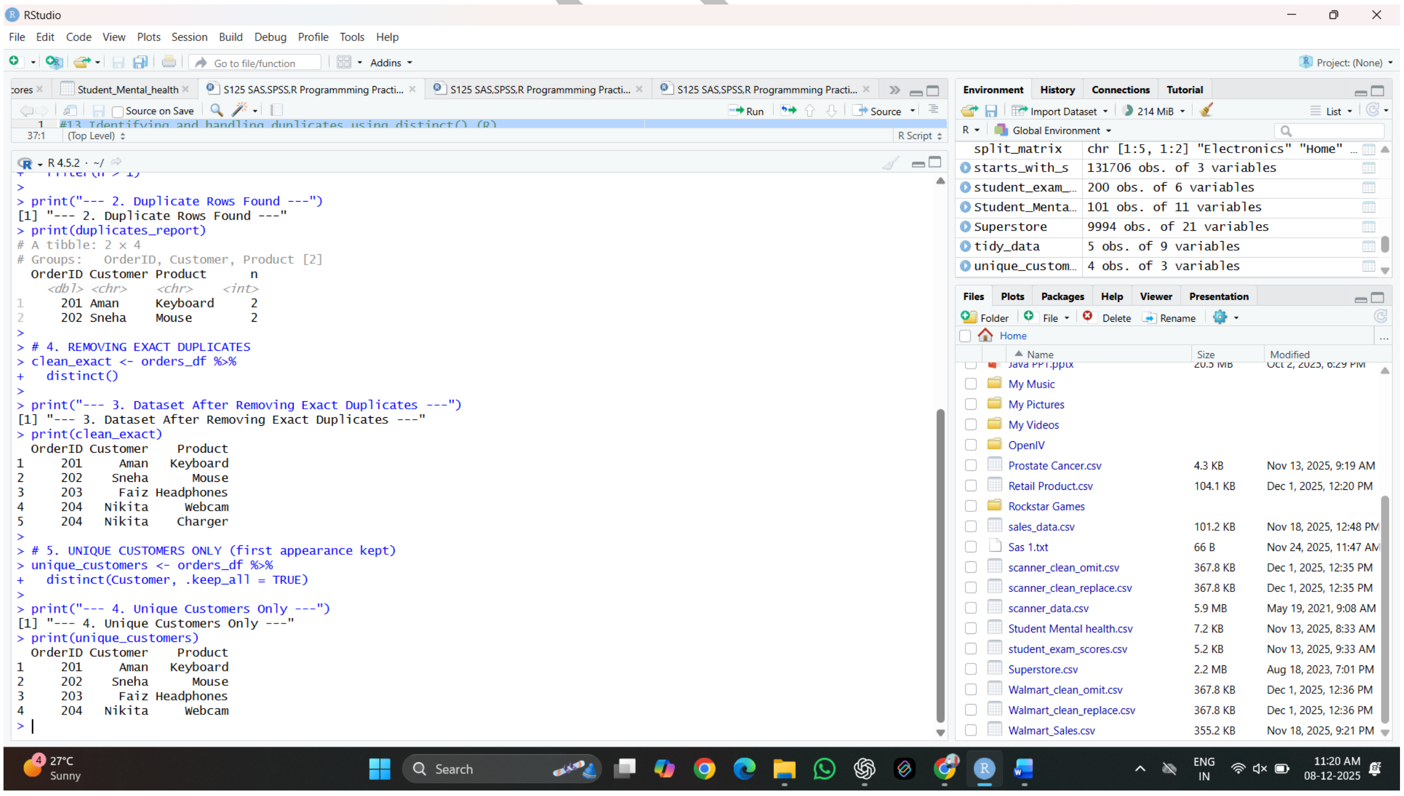
Jeetesh Shriganoor
S125

Sheth L.U.J & Sir M.V College
SAS/SPSS/R Programming
Practical No. 11 to 15

Aim: 13 Identifying and handling duplicates using distinct() (R).



```
> #13 Identifying and handling duplicates using distinct() (R).
> # 1. SETUP: Load dplyr
> library(dplyr)
>
> # 2. NEW CUSTOM DATASET with Duplicates
> orders_df <- data.frame(
+   OrderID = c(201, 202, 202, 203, 204, 201, 204),
+   Customer = c("Aman", "Sneha", "Sneha", "Faiz", "Nikita", "Aman", "Nikita"),
+   Product = c("Keyboard", "Mouse", "Mouse", "Headphones", "Webcam", "Keyboard", "Charger")
+ )
>
> print("--- 1. Original Dataset ---")
[1] "--- 1. Original Dataset ---"
> print(orders_df)
  OrderID Customer    Product
1     201     Aman   Keyboard
2     202    Sneha      Mouse
3     202    Sneha      Mouse
4     203     Faiz Headphones
5     204   Nikita     Webcam
6     201     Aman   Keyboard
7     204   Nikita    Charger
>
> # 3. IDENTIFYING DUPLICATES (Full row duplicates)
> duplicates_report <- orders_df %>%
+   group_by(OrderID, Customer, Product) %>%
+   count() %>%
+   filter(n > 1)
>
> print("--- 2. Duplicate Rows Found ---")
[1] "--- 2. Duplicate Rows Found ---"
> print(duplicates_report)
# A tibble: 2 × 4
# Groups:   OrderID, Customer, Product [2]
  OrderID Customer Product      n
    <dbl> <chr>    <chr>    <int>
1     201 Aman     Keyboard     2
2     202 Sneha    Mouse        2
>
```

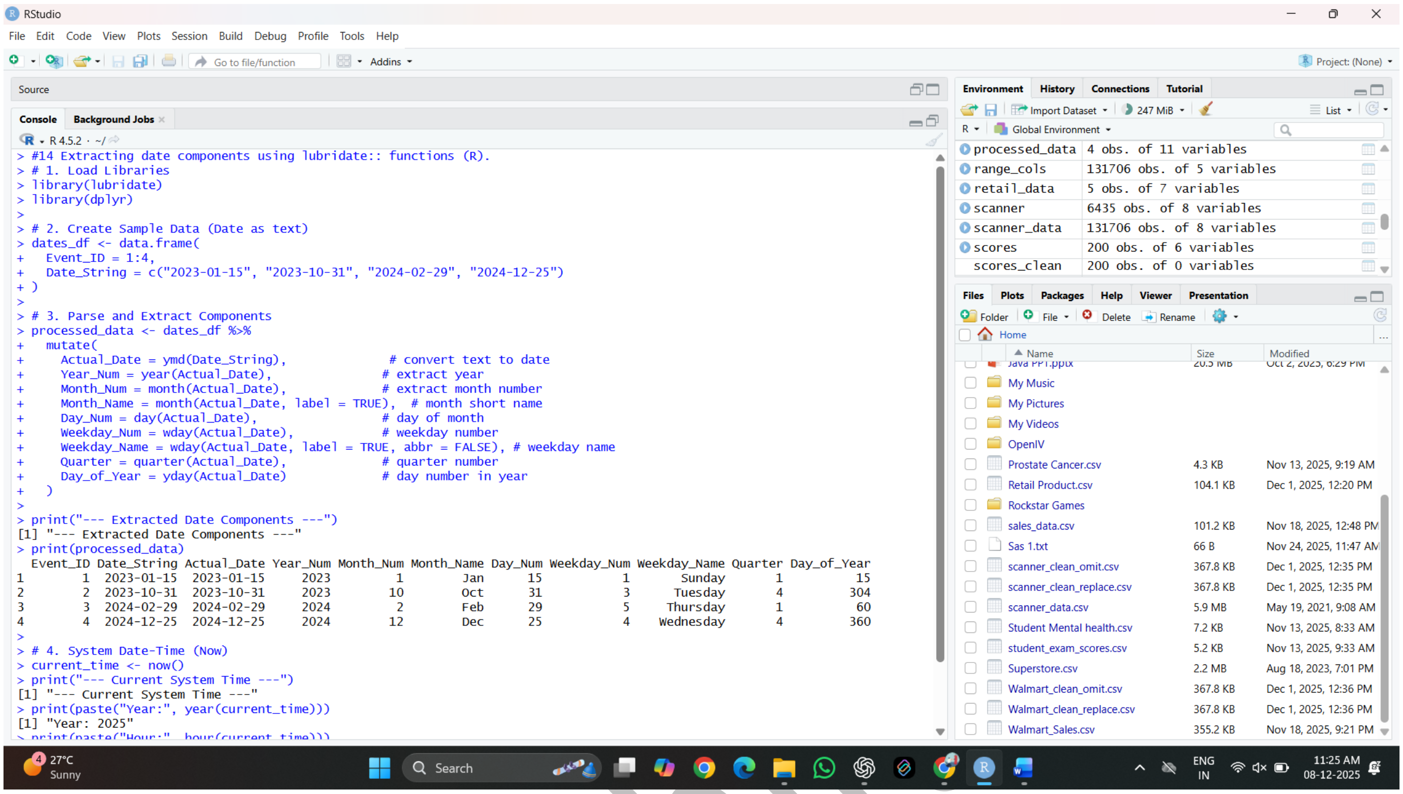


```
  filter(n > 1)
>
> print("--- 2. Duplicate Rows Found ---")
[1] "--- 2. Duplicate Rows Found ---"
> print(duplicates_report)
# A tibble: 2 × 4
# Groups:   OrderID, Customer, Product [2]
  OrderID Customer Product      n
    <dbl> <chr>    <chr>    <int>
1     201 Aman     Keyboard     2
2     202 Sneha    Mouse        2
>
> # 4. REMOVING EXACT DUPLICATES
> clean_exact <- orders_df %>%
+   distinct()
>
> print("--- 3. Dataset After Removing Exact Duplicates ---")
[1] "--- 3. Dataset After Removing Exact Duplicates ---"
> print(clean_exact)
  OrderID Customer    Product
1     201     Aman   Keyboard
2     202    Sneha      Mouse
3     203     Faiz Headphones
4     204   Nikita     Webcam
5     204   Nikita    Charger
>
> # 5. UNIQUE CUSTOMERS ONLY (first appearance kept)
> unique_customers <- orders_df %>%
+   distinct(Customer, .keep_all = TRUE)
>
> print("--- 4. Unique Customers Only ---")
[1] "--- 4. Unique Customers Only ---"
> print(unique_customers)
  OrderID Customer    Product
1     201     Aman   Keyboard
2     202    Sneha      Mouse
3     203     Faiz Headphones
4     204   Nikita     Webcam
> |
```

Jeetesh Shriganoor
S125

Sheth L.U.J & Sir M.V College
SAS/SPSS/R Programming
Practical No. 11 to 15

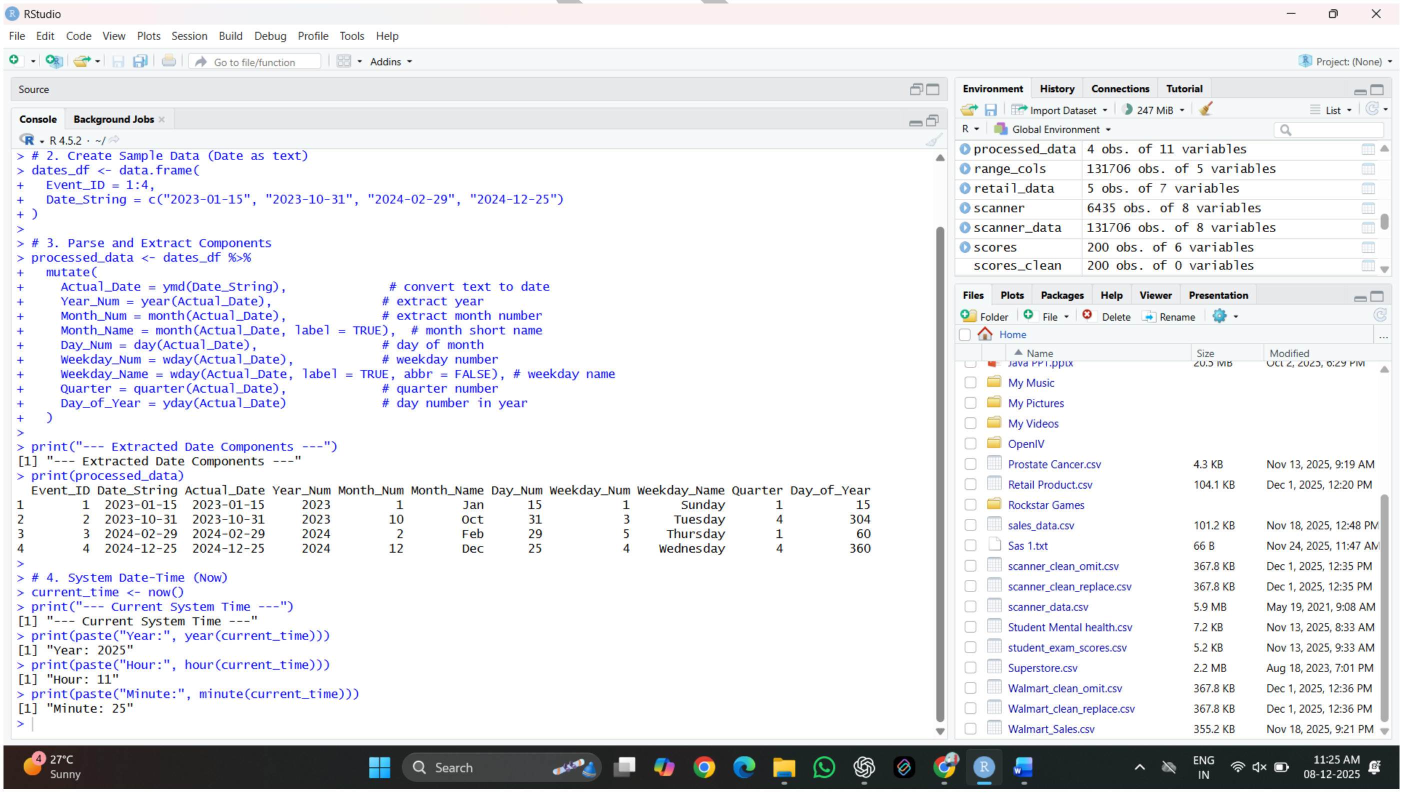
Aim: 14 Extracting date components using lubridate:: functions (R).



```
> #14 Extracting date components using lubridate:: functions (R).
> # 1. Load Libraries
> library(lubridate)
> library(dplyr)
>
> # 2. Create Sample Data (Date as text)
> dates_df <- data.frame(
+   Event_ID = 1:4,
+   Date_String = c("2023-01-15", "2023-10-31", "2024-02-29", "2024-12-25")
+ )
>
> # 3. Parse and Extract Components
> processed_data <- dates_df %>%
+   mutate(
+     Actual_Date = ymd(Date_String),              # convert text to date
+     Year_Num = year(Actual_Date),               # extract year
+     Month_Num = month(Actual_Date),             # extract month number
+     Month_Name = month(Actual_Date, label = TRUE),  # month short name
+     Day_Num = day(Actual_Date),                 # day of month
+     Weekday_Num = wday(Actual_Date),            # weekday number
+     Weekday_Name = wday(Actual_Date, label = TRUE, abbr = FALSE), # weekday name
+     Quarter = quarter(Actual_Date),             # quarter number
+     Day_of_Year = yday(Actual_Date)             # day number in year
+   )
>
> print("--- Extracted Date Components ---")
[1] "--- Extracted Date Components ---"
> print(processed_data)
  Event_ID Date_String Actual_Date Year_Num Month_Num Month_Name Day_Num Weekday_Num Weekday_Name Quarter Day_of_Year
1        1  2023-01-15  2023-01-15     2023         1        Jan      15           1       Sunday       1          15
2        2  2023-10-31  2023-10-31     2023        10        Oct      31           3      Tuesday       4         304
3        3  2024-02-29  2024-02-29     2024         2        Feb      29           5     Thursday       1          60
4        4  2024-12-25  2024-12-25     2024        12        Dec      25           4    Wednesday       4         360
>
> # 4. System Date-Time (Now)
> current_time <- now()
> print("--- Current System Time ---")
[1] "--- Current System Time ---"
> print(paste("Year:", year(current_time)))
[1] "Year: 2025"
```

```
processed_data   4 obs. of 11 variables
range_cols       131706 obs. of 5 variables
retail_data      5 obs. of 7 variables
scanner          6435 obs. of 8 variables
scanner_data     131706 obs. of 8 variables
scores           200 obs. of 6 variables
scores_clean     200 obs. of 0 variables
```

Name	Size	Modified
Java PPT.pptx	20.5 MB	Oct 2, 2025, 6:29 PM
My Music		
My Pictures		
My Videos		
OpenIV		
Prostate Cancer.csv	4.3 KB	Nov 13, 2025, 9:19 AM
Retail Product.csv	104.1 KB	Dec 1, 2025, 12:20 PM
Rockstar Games		
sales_data.csv	101.2 KB	Nov 18, 2025, 12:48 PM
Sas 1.txt	66 B	Nov 24, 2025, 11:47 AM
scanner_clean_omit.csv	367.8 KB	Dec 1, 2025, 12:35 PM
scanner_clean_replace.csv	367.8 KB	Dec 1, 2025, 12:35 PM
scanner_data.csv	5.9 MB	May 19, 2021, 9:08 AM
Student Mental health.csv	7.2 KB	Nov 13, 2025, 8:33 AM
student_exam_scores.csv	5.2 KB	Nov 13, 2025, 9:33 AM
Superstore.csv	2.2 MB	Aug 18, 2023, 7:01 PM
Walmart_clean_omit.csv	367.8 KB	Dec 1, 2025, 12:36 PM
Walmart_clean_replace.csv	367.8 KB	Dec 1, 2025, 12:36 PM
Walmart_Sales.csv	355.2 KB	Nov 18, 2025, 9:21 PM



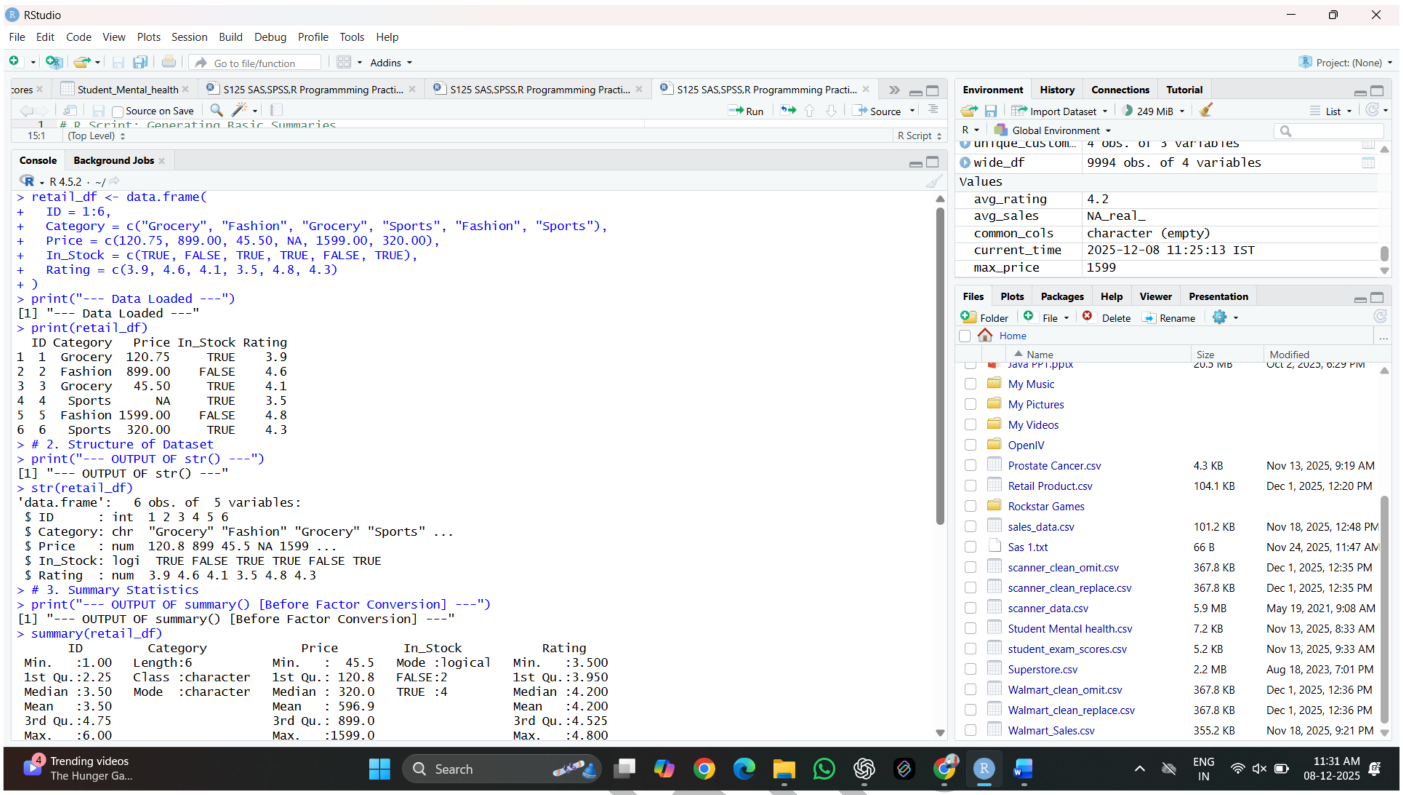
```
> # 2. Create Sample Data (Date as text)
> dates_df <- data.frame(
+   Event_ID = 1:4,
+   Date_String = c("2023-01-15", "2023-10-31", "2024-02-29", "2024-12-25")
+ )
>
> # 3. Parse and Extract Components
> processed_data <- dates_df %>%
+   mutate(
+     Actual_Date = ymd(Date_String),              # convert text to date
+     Year_Num = year(Actual_Date),               # extract year
+     Month_Num = month(Actual_Date),             # extract month number
+     Month_Name = month(Actual_Date, label = TRUE),  # month short name
+     Day_Num = day(Actual_Date),                 # day of month
+     Weekday_Num = wday(Actual_Date),            # weekday number
+     Weekday_Name = wday(Actual_Date, label = TRUE, abbr = FALSE), # weekday name
+     Quarter = quarter(Actual_Date),             # quarter number
+     Day_of_Year = yday(Actual_Date)             # day number in year
+   )
>
> print("--- Extracted Date Components ---")
[1] "--- Extracted Date Components ---"
> print(processed_data)
  Event_ID Date_String Actual_Date Year_Num Month_Num Month_Name Day_Num Weekday_Num Weekday_Name Quarter Day_of_Year
1        1  2023-01-15  2023-01-15     2023         1        Jan      15           1       Sunday       1          15
2        2  2023-10-31  2023-10-31     2023        10        Oct      31           3      Tuesday       4         304
3        3  2024-02-29  2024-02-29     2024         2        Feb      29           5     Thursday       1          60
4        4  2024-12-25  2024-12-25     2024        12        Dec      25           4    Wednesday       4         360
>
> # 4. System Date-Time (Now)
> current_time <- now()
> print("--- Current System Time ---")
[1] "--- Current System Time ---"
> print(paste("Year:", year(current_time)))
[1] "Year: 2025"
```

```
> print(paste("Hour:", hour(current_time)))
[1] "Hour: 11"
> print(paste("Minute:", minute(current_time)))
[1] "Minute: 25"
>
```

Jeetesh Shirganoor
S125

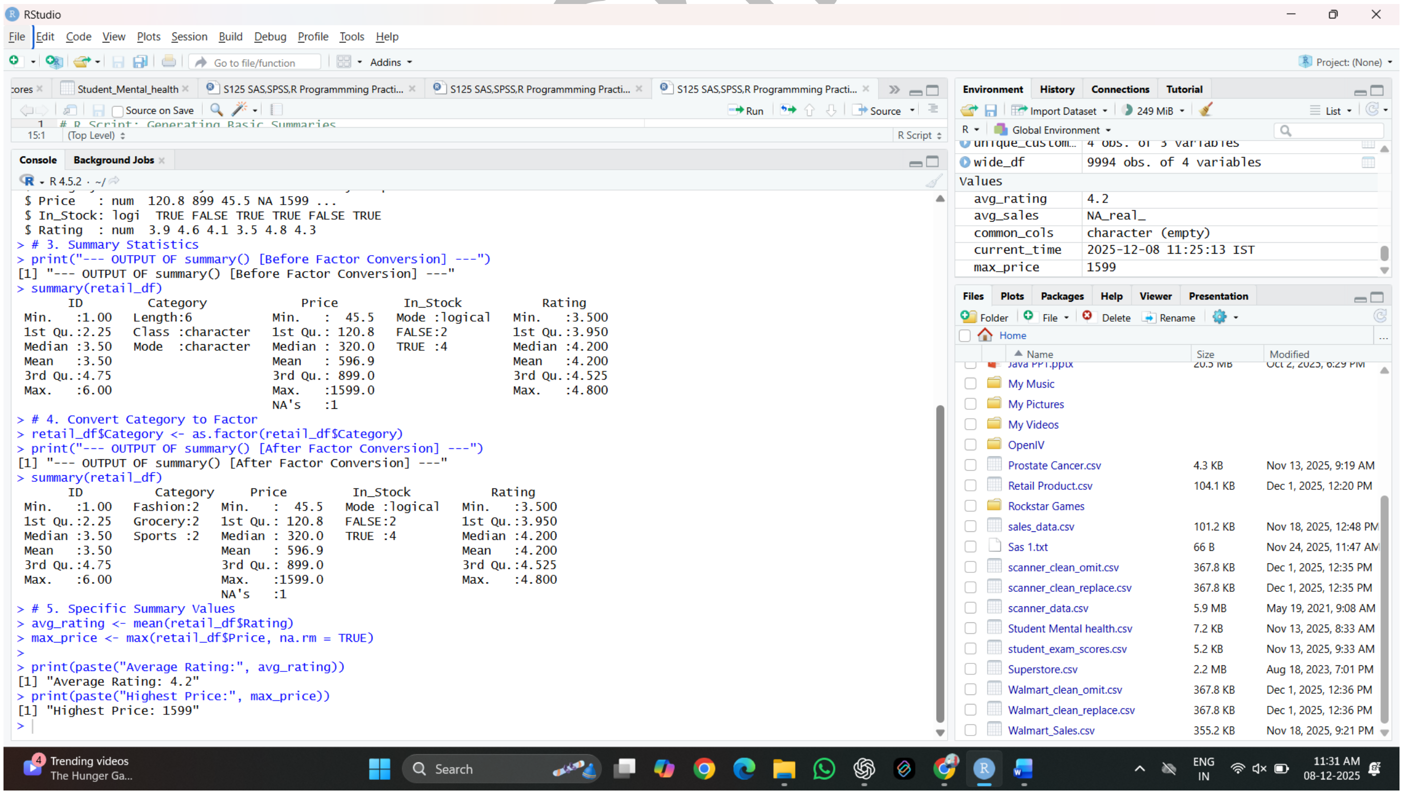
Sheth L.U.J & Sir M.V College
SAS/SPSS/R Programming
Practical No. 11 to 15

Aim: 15 Generating basic summaries using str() or summary() (R).



```
> retail_df <- data.frame(
+   ID = 1:6,
+   Category = c("Grocery", "Fashion", "Grocery", "Sports", "Fashion", "Sports"),
+   Price = c(120.75, 899.00, 45.50, NA, 1599.00, 320.00),
+   In_Stock = c(TRUE, FALSE, TRUE, TRUE, FALSE, TRUE),
+   Rating = c(3.9, 4.6, 4.1, 3.5, 4.8, 4.3)
+ )
> print("--- Data Loaded ---")
[1] "--- Data Loaded ---"
> print(retail_df)
  ID Category   Price In_Stock Rating
1  1  Grocery  120.75     TRUE    3.9
2  2  Fashion  899.00    FALSE    4.6
3  3  Grocery   45.50     TRUE    4.1
4  4   Sports      NA     TRUE    3.5
5  5  Fashion 1599.00    FALSE    4.8
6  6   Sports  320.00     TRUE    4.3
> # 2. Structure of Dataset
> print("--- OUTPUT OF str() ---")
[1] "--- OUTPUT OF str() ---"
> str(retail_df)
'data.frame':	6 obs. of  5 variables:
 $ ID      : int  1 2 3 4 5 6
 $ Category: chr  "Grocery" "Fashion" "Grocery" "Sports" ...
 $ Price   : num  120.8 899 45.5 NA 1599 ...
 $ In_Stock: logi  TRUE FALSE TRUE TRUE FALSE TRUE
 $ Rating  : num  3.9 4.6 4.1 3.5 4.8 4.3
> # 3. Summary Statistics
> print("--- OUTPUT OF summary() [Before Factor Conversion] ---")
[1] "--- OUTPUT OF summary() [Before Factor Conversion] ---"
> summary(retail_df)
       ID         Category             Price          In_Stock           Rating     
 Min.   :1.00   Length:6           Min.   :  45.5   Mode :logical   Min.   :3.500  
 1st Qu.:2.25   Class :character   1st Qu.: 120.8   FALSE:2         1st Qu.:3.950  
 Median :3.50   Mode  :character   Median : 320.0   TRUE :4         Median :4.200  
 Mean   :3.50                      Mean   : 596.9                   Mean   :4.200  
 3rd Qu.:4.75                      3rd Qu.: 899.0                   3rd Qu.:4.525  
 Max.   :6.00                      Max.   :1599.0                   Max.   :4.800  
```

```
wide_df        9994 obs. of 4 variables
Values
  avg_rating     4.2
  avg_sales      NA_real_
  common_cols    character (empty)
  current_time   2025-12-08 11:25:13 IST
  max_price      1599
```



```
 $ Price   : num  120.8 899 45.5 NA 1599 ...
 $ In_Stock: logi  TRUE FALSE TRUE TRUE FALSE TRUE
 $ Rating  : num  3.9 4.6 4.1 3.5 4.8 4.3
> # 3. Summary Statistics
> print("--- OUTPUT OF summary() [Before Factor Conversion] ---")
[1] "--- OUTPUT OF summary() [Before Factor Conversion] ---"
> summary(retail_df)
       ID         Category             Price          In_Stock           Rating     
 Min.   :1.00   Length:6           Min.   :  45.5   Mode :logical   Min.   :3.500  
 1st Qu.:2.25   Class :character   1st Qu.: 120.8   FALSE:2         1st Qu.:3.950  
 Median :3.50   Mode  :character   Median : 320.0   TRUE :4         Median :4.200  
 Mean   :3.50                      Mean   : 596.9                   Mean   :4.200  
 3rd Qu.:4.75                      3rd Qu.: 899.0                   3rd Qu.:4.525  
 Max.   :6.00                      Max.   :1599.0                   Max.   :4.800  
                                   NA's   :1                                       

> # 4. Convert Category to Factor
> retail_df$Category <- as.factor(retail_df$Category)
> print("--- OUTPUT OF summary() [After Factor Conversion] ---")
[1] "--- OUTPUT OF summary() [After Factor Conversion] ---"
> summary(retail_df)
       ID          Category     Price          In_Stock           Rating     
 Min.   :1.00   Fashion:2   Min.   :  45.5   Mode :logical   Min.   :3.500  
 1st Qu.:2.25   Grocery:2   1st Qu.: 120.8   FALSE:2         1st Qu.:3.950  
 Median :3.50   Sports :2   Median : 320.0   TRUE :4         Median :4.200  
 Mean   :3.50               Mean   : 596.9                   Mean   :4.200  
 3rd Qu.:4.75               3rd Qu.: 899.0                   3rd Qu.:4.525  
 Max.   :6.00               Max.   :1599.0                   Max.   :4.800  
                            NA's   :1                                       

> # 5. Specific Summary Values
> avg_rating <- mean(retail_df$Rating)
> max_price <- max(retail_df$Price, na.rm = TRUE)
> 
> print(paste("Average Rating:", avg_rating))
[1] "Average Rating: 4.2"
> print(paste("Highest Price:", max_price))
[1] "Highest Price: 1599"
> 
```

Jeetesh Shirganoor
S125