R Notebook

```
# Load required libraries
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.0.4
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(openxlsx)
library(lubridate)
```

```
# 1. Read the CSV file
customer_data <- read.csv("customer_sales.csv")
```

```
# 2. Clean and analyze the data
# Convert purchase_date to Date type
customer_data$purchase_date <- as.Date(customer_data$purchase_date)

# Check for missing values
missing_values <- colSums(is.na(customer_data))
print("Missing values in each column:")
```

```
## [1] "Missing values in each column:"
```

```
print(missing_values)
```

```
##      customer_id    purchase_date           amount product_category
##                0                0                0                0
##     customer_age           gender         location
##                0                0                0
```

```
# Basic data summary
summary_stats <- summary(customer_data)
print("Data summary:")
```

```
## [1] "Data summary:"
```

```
print(summary_stats)
```

```
##   customer_id   purchase_date            amount       product_category
##  Min.   :1001   Min.   :2023-01-05   Min.   : 22.50   Length:30
##  1st Qu.:1003   1st Qu.:2023-01-25   1st Qu.: 48.38   Class :character
##  Median :1006   Median :2023-02-18   Median : 79.25   Mode  :character
##  Mean   :1006   Mean   :2023-02-16   Mean   : 90.41
```

```
##  3rd Qu.:1008   3rd Qu.:2023-03-07   3rd Qu.:119.25
##  Max.   :1010   Max.   :2023-03-30   Max.   :220.00
##   customer_age     gender             location
##  Min.   :22.0   Length:30          Length:30
##  1st Qu.:31.0   Class :character   Class :character
##  Median :39.0   Mode  :character   Mode  :character
##  Mean   :40.8
##  3rd Qu.:49.0
##  Max.   :65.0
```

```r
# 3. Segment customers based on spending levels
# Calculate total spending per customer
customer_spending <- customer_data %>%
  group_by(customer_id) %>%
  summarise(
    total_spending = sum(amount),
    average_purchase = mean(amount),
    num_purchases = n(),
    last_purchase = max(purchase_date)
  ) %>%
  mutate(
    spending_category = case_when(
      total_spending >= 300 ~ "High Spender",
      total_spending >= 150 & total_spending < 300 ~ "Medium Spender",
      TRUE ~ "Low Spender"
    )
  )

# Add demographic information to customer spending
customer_demographics <- customer_data %>%
  group_by(customer_id) %>%
  summarise(
    age = first(customer_age),
    gender = first(gender),
    location = first(location)
  )

customer_spending <- customer_spending %>%
  left_join(customer_demographics, by = "customer_id")

# Define age groups
customer_spending <- customer_spending %>%
  mutate(
    age_group = case_when(
      age < 30 ~ "18-29",
      age >= 30 & age < 45 ~ "30-44",
      age >= 45 & age < 60 ~ "45-59",
      TRUE ~ "60+"
    )
  )
```

```r
# 4. Create multiple analysis perspectives
# Product category analysis
product_analysis <- customer_data %>%
  group_by(product_category) %>%
```

```r
  summarise(
    total_revenue = sum(amount),
    num_transactions = n(),
    avg_transaction = mean(amount),
    unique_customers = n_distinct(customer_id)
  ) %>%
  arrange(desc(total_revenue))

# Age group analysis
age_analysis <- customer_data %>%
  mutate(age_group = case_when(
    customer_age < 30 ~ "18-29",
    customer_age >= 30 & customer_age < 45 ~ "30-44",
    customer_age >= 45 & customer_age < 60 ~ "45-59",
    TRUE ~ "60+"
  )) %>%
  group_by(age_group, gender) %>%
  summarise(
    total_revenue = sum(amount),
    num_transactions = n(),
    avg_transaction = mean(amount),
    unique_customers = n_distinct(customer_id)
  )
```

```
## `summarise()` has grouped output by 'age_group'. You can override using the
## `.groups` argument.
```

```r
# Location analysis
location_analysis <- customer_data %>%
  group_by(location) %>%
  summarise(
    total_revenue = sum(amount),
    num_transactions = n(),
    avg_transaction = mean(amount),
    unique_customers = n_distinct(customer_id)
  )
```

```r
# 5. Track monthly sales trend
monthly_sales <- customer_data %>%
  mutate(month = floor_date(purchase_date, "month")) %>%
  group_by(month) %>%
  summarise(
    total_revenue = sum(amount),
    num_transactions = n(),
    avg_transaction = mean(amount),
    unique_customers = n_distinct(customer_id)
  )

# Create marketing impact analysis - simplified approach
marketing_impact <- data.frame(
  spending_category = c("High Spender", "Medium Spender", "Low Spender")
)

# Calculate metrics for each spending category
for (category in marketing_impact$spending_category) {
```

```
  subset_data <- customer_spending[customer_spending$spending_category == category, ]
  marketing_impact[marketing_impact$spending_category == category, "num_customers"] <- nrow(subset_data)
  marketing_impact[marketing_impact$spending_category == category, "total_revenue"] <- sum(subset_data$
  marketing_impact[marketing_impact$spending_category == category, "avg_spending"] <- mean(subset_data$
}

# Add conversion rates and potential revenue
marketing_impact$conversion_rate <- ifelse(marketing_impact$spending_category == "High Spender", 0.15,
                                    ifelse(marketing_impact$spending_category == "Medium Spender", 0.10,

marketing_impact$avg_additional_purchase <- ifelse(marketing_impact$spending_category == "High Spender"
                                            ifelse(marketing_impact$spending_category == "Medium Spender

marketing_impact$potential_revenue <- marketing_impact$num_customers *
                              marketing_impact$conversion_rate *
                              marketing_impact$avg_additional_purchase

# Create customer recommendations - simplified approach
customer_recommendations <- customer_spending %>%
  select(customer_id, total_spending, spending_category, num_purchases, age_group, gender, location)

# Add recommendation column
customer_recommendations$recommendation <- NA
for (i in 1:nrow(customer_recommendations)) {
  if (customer_recommendations$spending_category[i] == "High Spender") {
    customer_recommendations$recommendation[i] <- "Premium loyalty program"
  } else if (customer_recommendations$spending_category[i] == "Medium Spender" &&
             customer_recommendations$num_purchases[i] > 2) {
    customer_recommendations$recommendation[i] <- "Targeted promotions for most purchased categories"
  } else if (customer_recommendations$spending_category[i] == "Medium Spender") {
    customer_recommendations$recommendation[i] <- "Encourage repeat purchases with discounts"
  } else if (customer_recommendations$spending_category[i] == "Low Spender" &&
             customer_recommendations$num_purchases[i] > 1) {
    customer_recommendations$recommendation[i] <- "Value-based offers"
  } else {
    customer_recommendations$recommendation[i] <- "Re-engagement campaign"
  }
}

# Debug print to verify data frames
print("Sample of marketing_impact data:")
```

```
## [1] "Sample of marketing_impact data:"
```

```
print(head(marketing_impact))
```

```
##   spending_category num_customers total_revenue avg_spending conversion_rate
## 1      High Spender             3       1036.23      345.410            0.15
## 2    Medium Spender             6       1557.69      259.615            0.10
## 3       Low Spender             1        118.24      118.240            0.05
##   avg_additional_purchase potential_revenue
## 1                     200             90.0
## 2                     100             60.0
## 3                      50              2.5
```

```r
print("Sample of customer_recommendations data:")
```

```
## [1] "Sample of customer_recommendations data:"
```

```r
print(head(customer_recommendations))
```

```
## # A tibble: 6 x 8
##   customer_id total_spending spending_category num_purchases age_group gender
##         <int>          <dbl> <chr>                     <int> <chr>     <chr>
## 1        1001           361. High Spender                  3 30-44     M
## 2        1002           287. Medium Spender                3 18-29     F
## 3        1003           324  High Spender                  3 45-59     F
## 4        1004           212. Medium Spender                3 45-59     F
## 5        1005           118. Low Spender                   3 18-29     M
## 6        1006           274. Medium Spender                3 30-44     M
## # i 2 more variables: location <chr>, recommendation <chr>
```

```r
# 6. Export all analyses into a single Excel file with multiple sheets
# Create workbook
wb <- createWorkbook()

# Add worksheets
addWorksheet(wb, "Customer Spending")
addWorksheet(wb, "Product Categories")
addWorksheet(wb, "Age Group Analysis")
addWorksheet(wb, "Location Analysis")
addWorksheet(wb, "Monthly Sales")
addWorksheet(wb, "Marketing Impact")
addWorksheet(wb, "Customer Recommendations")

# Write data to worksheets - with explicit error handling
tryCatch({
  writeData(wb, "Customer Spending", customer_spending)
  print("Wrote Customer Spending worksheet successfully")
}, error = function(e) {
  print(paste("Error writing Customer Spending:", e$message))
})
```

```
## [1] "Wrote Customer Spending worksheet successfully"
```

```r
tryCatch({
  writeData(wb, "Product Categories", product_analysis)
  print("Wrote Product Categories worksheet successfully")
}, error = function(e) {
  print(paste("Error writing Product Categories:", e$message))
})
```

```
## [1] "Wrote Product Categories worksheet successfully"
```

```r
tryCatch({
  writeData(wb, "Age Group Analysis", age_analysis)
  print("Wrote Age Group Analysis worksheet successfully")
}, error = function(e) {
  print(paste("Error writing Age Group Analysis:", e$message))
})
```

```
## [1] "Wrote Age Group Analysis worksheet successfully"
```

```
tryCatch({
  writeData(wb, "Location Analysis", location_analysis)
  print("Wrote Location Analysis worksheet successfully")
}, error = function(e) {
  print(paste("Error writing Location Analysis:", e$message))
})
```

```
## [1] "Wrote Location Analysis worksheet successfully"
```

```
tryCatch({
  writeData(wb, "Monthly Sales", monthly_sales)
  print("Wrote Monthly Sales worksheet successfully")
}, error = function(e) {
  print(paste("Error writing Monthly Sales:", e$message))
})
```

```
## [1] "Wrote Monthly Sales worksheet successfully"
```

```
tryCatch({
  writeData(wb, "Marketing Impact", marketing_impact)
  print("Wrote Marketing Impact worksheet successfully")
}, error = function(e) {
  print(paste("Error writing Marketing Impact:", e$message))
})
```

```
## [1] "Wrote Marketing Impact worksheet successfully"
```

```
tryCatch({
  writeData(wb, "Customer Recommendations", customer_recommendations)
  print("Wrote Customer Recommendations worksheet successfully")
}, error = function(e) {
  print(paste("Error writing Customer Recommendations:", e$message))
})
```

```
## [1] "Wrote Customer Recommendations worksheet successfully"
```

```
# Style the workbook - using a safer approach
for (sheet in names(wb)) {
  # Get number of columns in the sheet's data frame
  cols_count <- switch(sheet,
                       "Customer Spending" = ncol(customer_spending),
                       "Product Categories" = ncol(product_analysis),
                       "Age Group Analysis" = ncol(age_analysis),
                       "Location Analysis" = ncol(location_analysis),
                       "Monthly Sales" = ncol(monthly_sales),
                       "Marketing Impact" = ncol(marketing_impact),
                       "Customer Recommendations" = ncol(customer_recommendations),
                       1) # Default to 1 if sheet name doesn't match

  # Style headers
  addStyle(wb, sheet, style = createStyle(textDecoration = "bold"), rows = 1, cols = 1:cols_count)

  # Auto-size columns (safely)
  setColWidths(wb, sheet, cols = 1:cols_count, widths = "auto")
}

# Print worksheet names for verification
```

```r
print("Worksheets in workbook:")
```

```
## [1] "Worksheets in workbook:"
```

```r
print(names(wb))
```

```
## [1] "Customer Spending"        "Product Categories"
## [3] "Age Group Analysis"       "Location Analysis"
## [5] "Monthly Sales"            "Marketing Impact"
## [7] "Customer Recommendations"
```

```r
# Save workbook
saveWorkbook(wb, "customer_analysis_results.xlsx", overwrite = TRUE)

# Print exact file location
file_path <- file.path(getwd(), "customer_analysis_results.xlsx")
print(paste("Analysis complete! Results exported to:", file_path))
```

```
## [1] "Analysis complete! Results exported to: /Users/simransinha/Documents/Semester 3/SDA/Assignments/
```