ABS Transactions Analysis — Fully Annotated

Arash Nateghian

1) Load Libraries & Dataset

```r
# Core data and wrangling
library(dplyr)       # verbs: filter, select, group_by, summarise, mutate
```

```
## Warning: package 'dplyr' was built under R version 4.5.1
```

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

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

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

```r
library(tidyverse)   # brings ggplot2, readr, tidyr, etc.
```

```
## Warning: package 'tidyverse' was built under R version 4.5.1
```

```
## Warning: package 'ggplot2' was built under R version 4.5.1
```

```
## Warning: package 'tibble' was built under R version 4.5.1
```

```
## Warning: package 'tidyr' was built under R version 4.5.1
```

```
## Warning: package 'readr' was built under R version 4.5.1
```

```
## Warning: package 'purrr' was built under R version 4.5.1
```

```
## Warning: package 'stringr' was built under R version 4.5.1
## Warning: package 'forcats' was built under R version 4.5.1
```

```
## Warning: package 'lubridate' was built under R version 4.5.1
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ forcats   1.0.0     ✔ readr     2.1.5
## ✔ ggplot2   3.5.2     ✔ stringr   1.5.1
## ✔ lubridate 1.9.4     ✔ tibble    3.3.0
## ✔ purrr     1.1.0     ✔ tidyr     1.3.1
```

```
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
e errors
```

```
library(stringr)     # robust string/regex helpers
library(lubridate)   # date parsing, wday/month/year helpers
library(ggplot2)     # plots
library(purrr)       # functional map/reduce style helpers
library(arules)      # market basket analysis (Apriori)
```

```
## Warning: package 'arules' was built under R version 4.5.1
```

```
## Loading required package: Matrix
##
## Attaching package: 'Matrix'
##
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
##
##
## Attaching package: 'arules'
##
## The following object is masked from 'package:dplyr':
##
##     recode
##
## The following objects are masked from 'package:base':
##
##     abbreviate, write
```

```
transactions <- read.csv(
  "TransactionData.csv",
  header = TRUE,
  sep = ",",
  encoding = "UTF-8",
  quote = "\""
)
```

2) Filter out unwanted rows and handle missing/invalid data

```
transactions <- transactions |>
  filter(
    STORENAME != "Westwood",
    CLASSIFICATIONDEPARTMENT != "DONATIONS",
    CLASSIFICATIONDEPARTMENT != "NON INVENTORY",
    SIZE != "UNIT",
    (SIZE != "" & !is.na(SIZE)),
    (TAGDESC != "NULL" & !is.na(TAGDESC)),
    (TAGDESC == "STOCK")
  )
```

3) Clean and normalize text columns

```
transactions <- transactions |>
  mutate(
    CLASSIFICATIONDEPARTMENT = trimws(CLASSIFICATIONDEPARTMENT),
    STORENAME = trimws(STORENAME),
    STORENAME = ifelse(STORENAME == "White Oak", "White Oak Town Center", STORENAME),
    SIZE = ifelse(SIZE == "5LTR", "5L", SIZE),
    SIZE = ifelse(SIZE == "3LTR", "3L", SIZE),
    SIZE = ifelse(SIZE == "1LTR", "1L", SIZE),
    TRANSDATE = as.Date(TRANSDATE, format = "%Y-%m-%d"),
    YEAR = year(TRANSDATE),
    DESCRIPTION = DESCRIPTION %>% str_trim() %>% str_squish() %>% str_to_upper()
  )
```

4) Normalize PACKUNIT values

```
transactions <- transactions %>%
  mutate(
    PACKUNIT = case_when(
      grepl("^[0-9]+$", PACKUNIT) & PACKUNIT == "1" ~ "Btl",
      grepl("^[0-9]+$", PACKUNIT) ~ paste0(PACKUNIT, "pk"),
      TRUE ~ PACKUNIT
    )
  )
```

5) Fix inconsistent pack unit labels

```
transactions <- transactions %>%
  mutate(
    PACKUNIT = case_when(
      PACKUNIT == "08pk" ~ "8pk",
      PACKUNIT == "06pk" ~ "6pk",
      PACKUNIT == "05pk" ~ "5pk",
      PACKUNIT == "04pk" ~ "4pk",
      PACKUNIT == "03pk" ~ "3pk",
      PACKUNIT == "02pk" ~ "2pk",
      PACKUNIT == "01pk" ~ "Btl",
      TRUE ~ PACKUNIT
    )
  )
```

6) Fix missing BEER classification type

```
transactions$CLASSIFICATIONTYPE <- ifelse(
  (transactions$CLASSIFICATIONTYPE == "" |
     transactions$CLASSIFICATIONTYPE == "NULL" |
     is.na(transactions$CLASSIFICATIONTYPE)) &
    transactions$CLASSIFICATIONDEPARTMENT == "BEER",
  transactions$CLASSIFICATIONDEPARTMENT,
  transactions$CLASSIFICATIONTYPE
)
```

7) # Standardize Product Descriptions

```r
# Identify items with conflicting descriptions

item_desc_conflicts <- transactions %>%
  select(ITEMID, DESCRIPTION) %>%
  distinct() %>%
  group_by(ITEMID) %>%
  filter(n_distinct(DESCRIPTION) > 1) %>%
  arrange(ITEMID)

# Choose preferred description (the one that includes size)

dict <- item_desc_conflicts %>%
  mutate(has_size = str_detect(DESCRIPTION, "\\b[0-9.]+(ML|L|OZ|Z|LTR)\\b")) %>%
  group_by(ITEMID) %>%
  mutate(
    preferred_desc = DESCRIPTION[has_size][1]  # choose version with size
  ) %>%
  filter(!has_size) %>%  # only the short ones need replacing
  ungroup()

# Output suggested case_when rules for manual review

dict %>%
  mutate(
    rule = paste0(
      'DESCRIPTION == "', DESCRIPTION, '" ~ "', preferred_desc, '"'
    )
  ) %>%
  pull(rule) %>%
  cat(sep = ",\n")
```

```
# Automated cleaning method

transactions_clean <- transactions %>%
  group_by(ITEMID) %>%
  mutate(
    has_size = str_detect(DESCRIPTION, "\\b[0-9.]+(ML|L|OZ|Z)\\b"),
    DESCRIPTION = if (any(has_size)) DESCRIPTION[has_size][1] else DESCRIPTION
  ) %>%
  ungroup() %>%
  select(-has_size)

transactions <- transactions_clean

# Manual override rules for edge cases

transactions <- transactions %>%
  mutate(
    DESCRIPTION = case_when(
      DESCRIPTION == "BOWMANS PET VODKA" ~ "BOWMANS PET VODKA 200ML",
      DESCRIPTION == "NEW AMSTERDAM APPLE" ~ "NEW AMSTERDAM APPLE - 50ML",
      DESCRIPTION == "NEW AMSTERDAM PEACH" ~ "NEW AMSTERDAM PEACH - 50ML",
      DESCRIPTION == "NEW AMSTERDAM- PASSION F" ~ "NEW AMSTERDAM- PASSION F - 50ML",
      DESCRIPTION == "NEW AMSTERDAM GRAPEFRUIT" ~ "NEW AMSTERDAM GRAPEFRUIT - 50ML",
      DESCRIPTION == "L MARCA PRO -ROSE" ~ "L MARCA PRO -ROSE - 750ML",
      DESCRIPTION == "MR BOSTON TRIPLE SEC" ~ "MR BOSTON TRIPLE SEC - 1LTR",
      DESCRIPTION == "MR BOSTON PEACH SCHNAPPS 1" ~ "MR BOSTON PEACH SCHNAPPS 1 - 1LTR",

      TRUE ~ DESCRIPTION
    )
  )
```

8) Data Quality & Structure Diagnostics

```
cat("\n=== NA Value Check ===\n")
```

```
##
## === NA Value Check ===
na_counts <- colSums(is.na(transactions))
print(na_counts)
```

```
##                TRANSDATE            TRANSACTIONID                    STORE
##                        0                        0                        0
##                STORENAME                   ITEMID              DESCRIPTION
##                        0                        0                        0
##                NETAMOUNT                  LINEQTY                 PACKUNIT
##                        0                        0                        0
##                 TOTALQTY                  ITEMTAG                  TAGDESC
##                        0                        0                        0
## CLASSIFICATIONDEPARTMENT       CLASSIFICATIONTYPE   CLASSIFICATIONCATEGORY
##                        0                        0                        0
##           BOTTLESPERCASE                     SIZE              CUSTACCOUNT
##                        0                        0                        0
##                     YEAR
##                        0
```

```
if(any(na_counts > 0)) {
  cat("\nColumns with missing values:\n")
  print(na_counts[na_counts > 0])
} else {
  cat("No missing values found.\n")
}
```

```
## No missing values found.
```

9) Product Mix & Size Performance

```
# Sales & quantity by package size

transactions %>%
  group_by(SIZE) %>%
  summarise(
    TotalSales = sum(NETAMOUNT, na.rm = TRUE),
    TotalQty   = sum(LINEQTY,  na.rm = TRUE)
  ) %>%
  arrange(desc(TotalSales))
```

```
## # A tibble: 36 × 3
##    SIZE  TotalSales TotalQty
##    <chr>      <dbl>    <dbl>
##  1 750ML 131020386.  6039756
##  2 1.75L  43902274.  1444640
##  3 12.0Z  13699709.   844580
##  4 1L     11097585.   480518
##  5 375ML   8352566.   811572
##  6 1.5L    5794310.   423448
##  7 50ML    4438538.  2836377
##  8 3L      3481740.   169073
##  9 355ML   2340103.   148296
## 10 200ML   2331356.   350634
## # ℹ 26 more rows
```

```
# Sales by category type (e.g., VODKA, PROSECCO, etc.)

transactions %>%
  group_by(CLASSIFICATIONTYPE) %>%
  summarise(TotalSales = sum(NETAMOUNT, na.rm = TRUE))
```

```
## # A tibble: 102 × 2
##    CLASSIFICATIONTYPE TotalSales
##    <chr>                   <dbl>
##  1 ALES                 2070524.
##  2 APERITIF              115210.
##  3 BAROLO                 40554.
##  4 BEAUJOLAIS             52585.
##  5 BEAUJOLAIS VILLAGE     87642.
##  6 BIANCO                  1649.
##  7 BLANC                  28187.
##  8 BLUSH                 148868.
##  9 BOURBON             17299433.
## 10 BRANDY               1947305.
## # ℹ 92 more rows
```

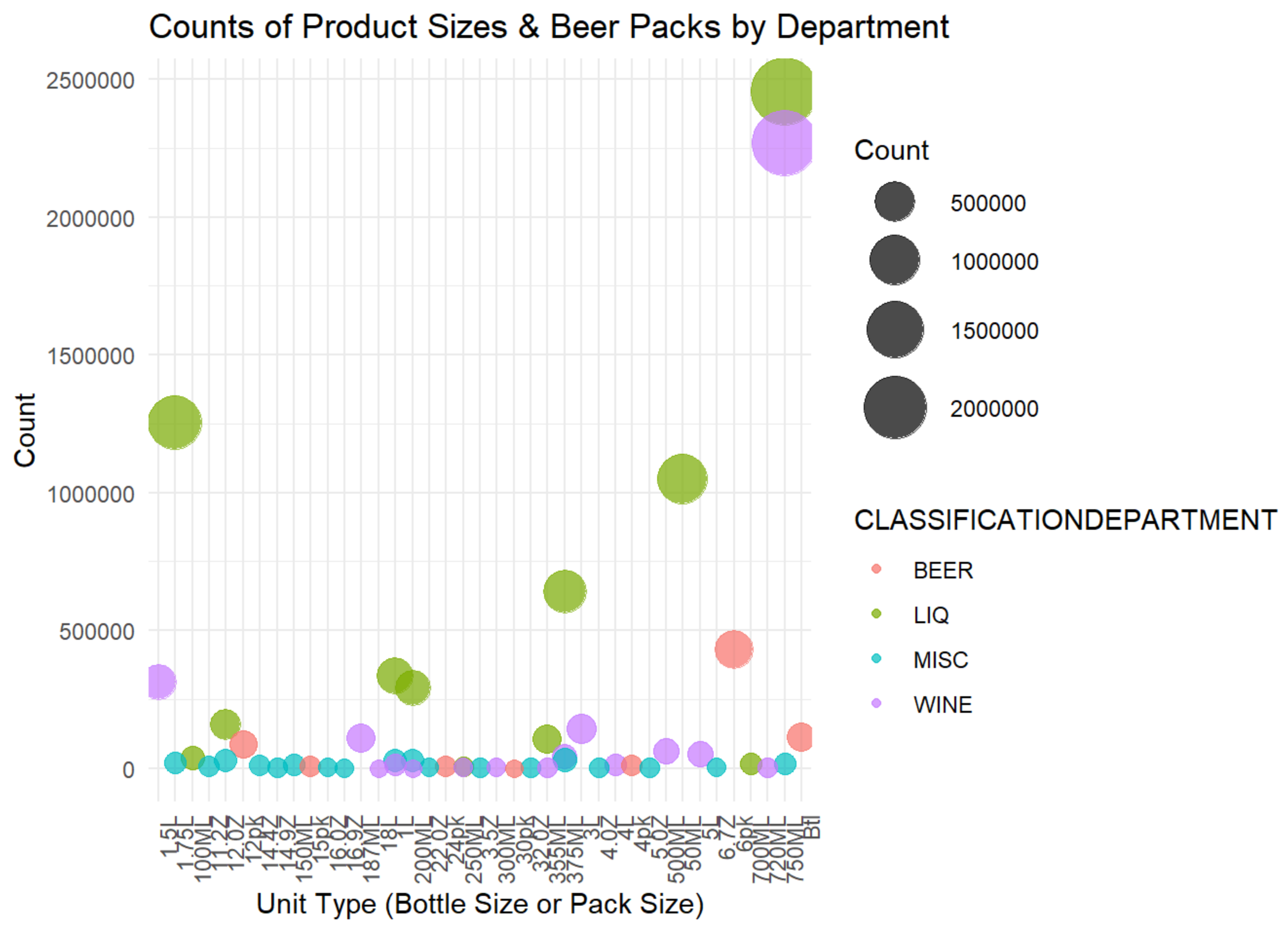
10) High-Value Transactions (> $200)

```r
# Count & sales of high-value receipts per store

top_high_value_stores <- transactions %>%
  filter(NETAMOUNT > 200) %>%
  group_by(STORENAME) %>%
  summarise(
    HighValueCount      = n(),
    TotalHighValueSales = sum(NETAMOUNT),
    .groups = "drop"
  ) %>%
  arrange(desc(HighValueCount))

# % of receipts that are > $200 per store

store_high_value_share <- transactions %>%
  group_by(STORENAME) %>%
  summarise(
    TotalTransactions      = n(),
    HighValueTransactions  = sum(NETAMOUNT > 200, na.rm = TRUE),
    HighValueShare         = round(100 * HighValueTransactions / TotalTransactions, 2),
    .groups = "drop"
  ) %>%
  arrange(desc(HighValueShare))
```

11) Bottle / Pack Size Popularity

```r
transactionsnew <- transactions %>%
  mutate(UnitType = ifelse(CLASSIFICATIONDEPARTMENT == "BEER", PACKUNIT, SIZE))

size_counts <- transactionsnew %>%
  group_by(CLASSIFICATIONDEPARTMENT, UnitType) %>%
  summarise(Count = n(), .groups = "drop") %>%
  arrange(desc(Count))
```

```r
ggplot(size_counts,
       aes(x = UnitType, y = Count, size = Count, color = CLASSIFICATIONDEPARTMENT)) +
  geom_point(alpha = 0.7) +
  scale_size(range = c(3, 12)) +
  labs(
    title = "Counts of Product Sizes & Beer Packs by Department",
    x = "Unit Type (Bottle Size or Pack Size)",
    y = "Count"
  ) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



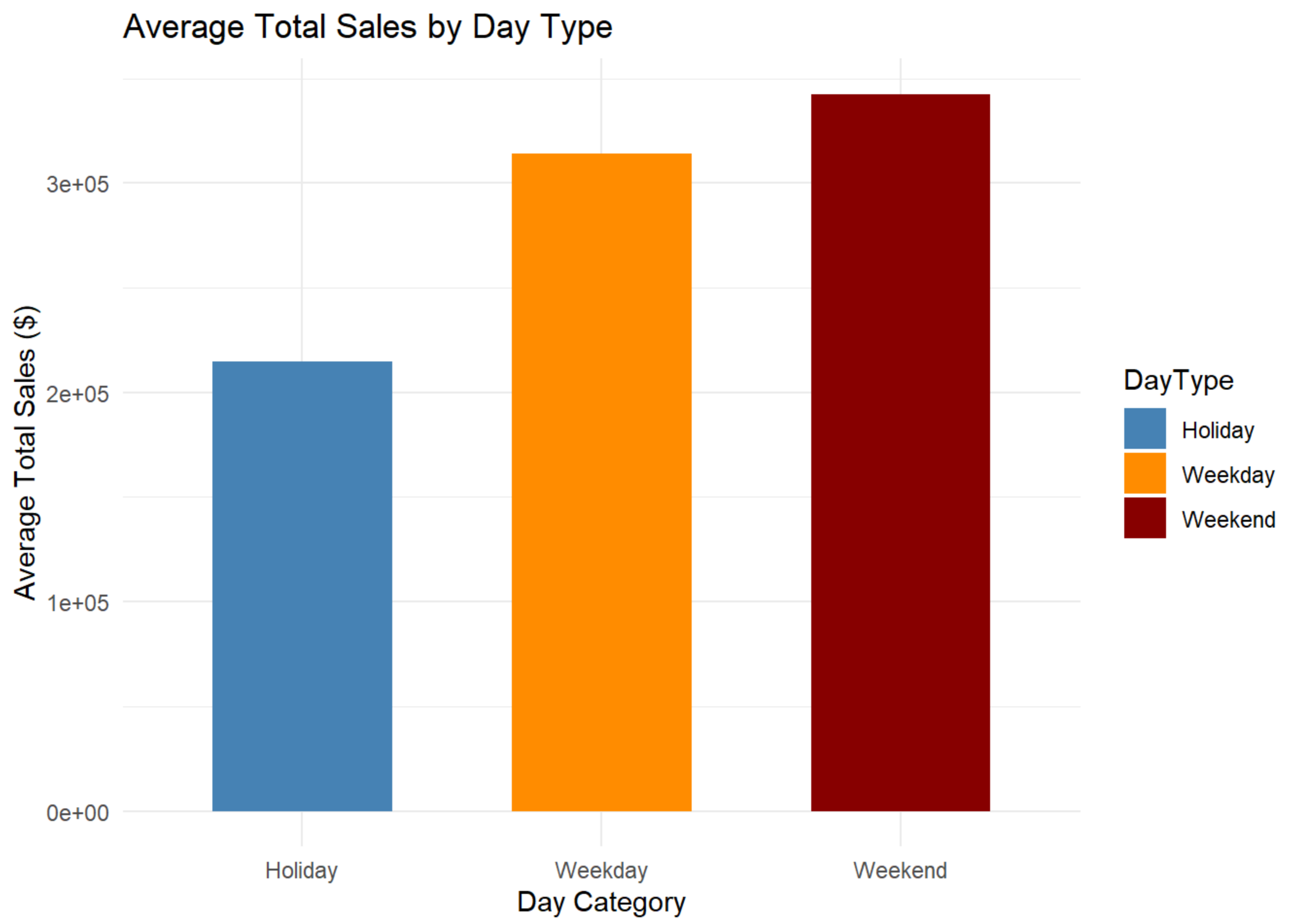
12) Category Mix by Store (Pivot)

```r
# Receipt counts by department x store

dept_by_store <- transactions %>%
  group_by(STORENAME, CLASSIFICATIONDEPARTMENT) %>%
  summarise(Count = n(), .groups = "drop") %>%
  pivot_wider(names_from = CLASSIFICATIONDEPARTMENT, values_from = Count, values_fill = 0)


# Sales by department x store

dept_sales_by_store <- transactions %>%
  group_by(STORENAME, CLASSIFICATIONDEPARTMENT) %>%
  summarise(TotalSales = sum(NETAMOUNT, na.rm = TRUE), .groups = "drop") %>%
  pivot_wider(names_from = CLASSIFICATIONDEPARTMENT, values_from = TotalSales, values_fill = 0)
```

13) Weekday / Weekend / Holiday Sales

```r
transactions$TRANSDATE <- as.Date(transactions$TRANSDATE)

ny_range  <- seq(as.Date("2023-12-21"), as.Date("2024-12-24"), by="day")
ny_range1 <- seq(as.Date("2024-12-21"), as.Date("2025-12-24"), by="day")
ny_range2 <- seq(as.Date("2023-12-29"), as.Date("2024-01-02"), by="day")
ny_range3 <- seq(as.Date("2024-12-29"), as.Date("2025-01-02"), by="day")

us_holidays <- as.Date(c(
  "2023-07-04","2023-09-04","2023-11-23","2023-12-25",
  ny_range, ny_range1,
  "2024-05-27","2024-07-04","2024-09-02","2024-11-28","2024-12-25",
  ny_range2, ny_range3,
  "2025-05-26"
))

transactions <- transactions %>%
  mutate(
    DayOfWeek = wday(TRANSDATE, label = TRUE),
    DayType   = case_when(
      TRANSDATE %in% us_holidays         ~ "Holiday",
      DayOfWeek %in% c("Sat","Sun")      ~ "Weekend",
      TRUE                               ~ "Weekday"
    )
  )

# Aggregate daily by DayType

daily_sales <- transactions %>%
  group_by(TRANSDATE, DayType) %>%
  summarise(DailySales = sum(NETAMOUNT, na.rm = TRUE), .groups = "drop")

# Mean per DayType

avg_totals <- daily_sales %>%
  group_by(DayType) %>%
  summarise(AverageDailySales = mean(DailySales), .groups = "drop")
```

```r
ggplot(avg_totals, aes(x = DayType, y = AverageDailySales, fill = DayType)) +
  geom_col(width = 0.6) +
  labs(
    title = "Average Total Sales by Day Type",
    x = "Day Category",
    y = "Average Total Sales ($)"
  ) +
  theme_minimal() +
  scale_fill_manual(values = c("steelblue", "darkorange", "darkred"))
```

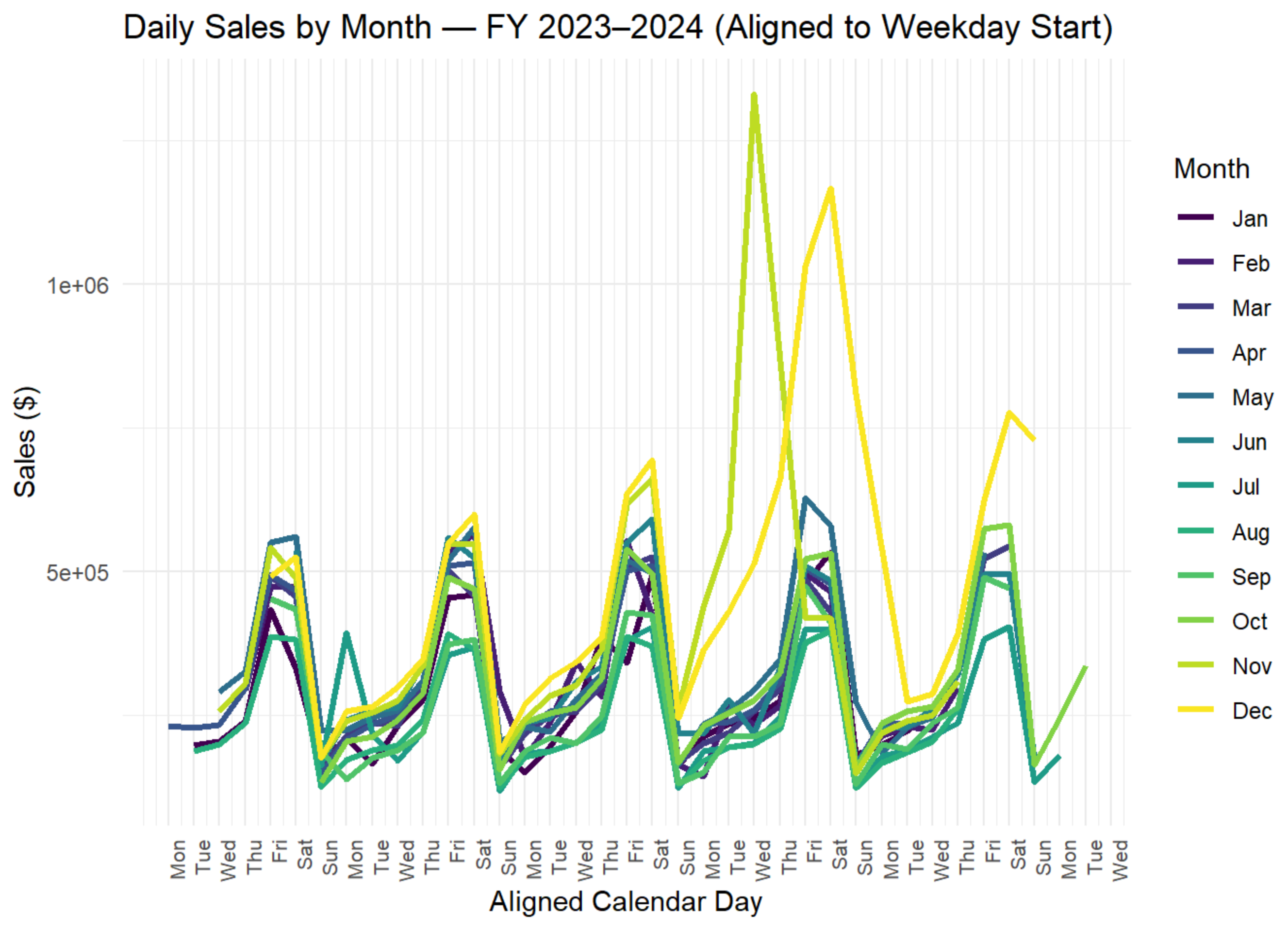


14) Daily Sales — FY23-24 vs FY24-25

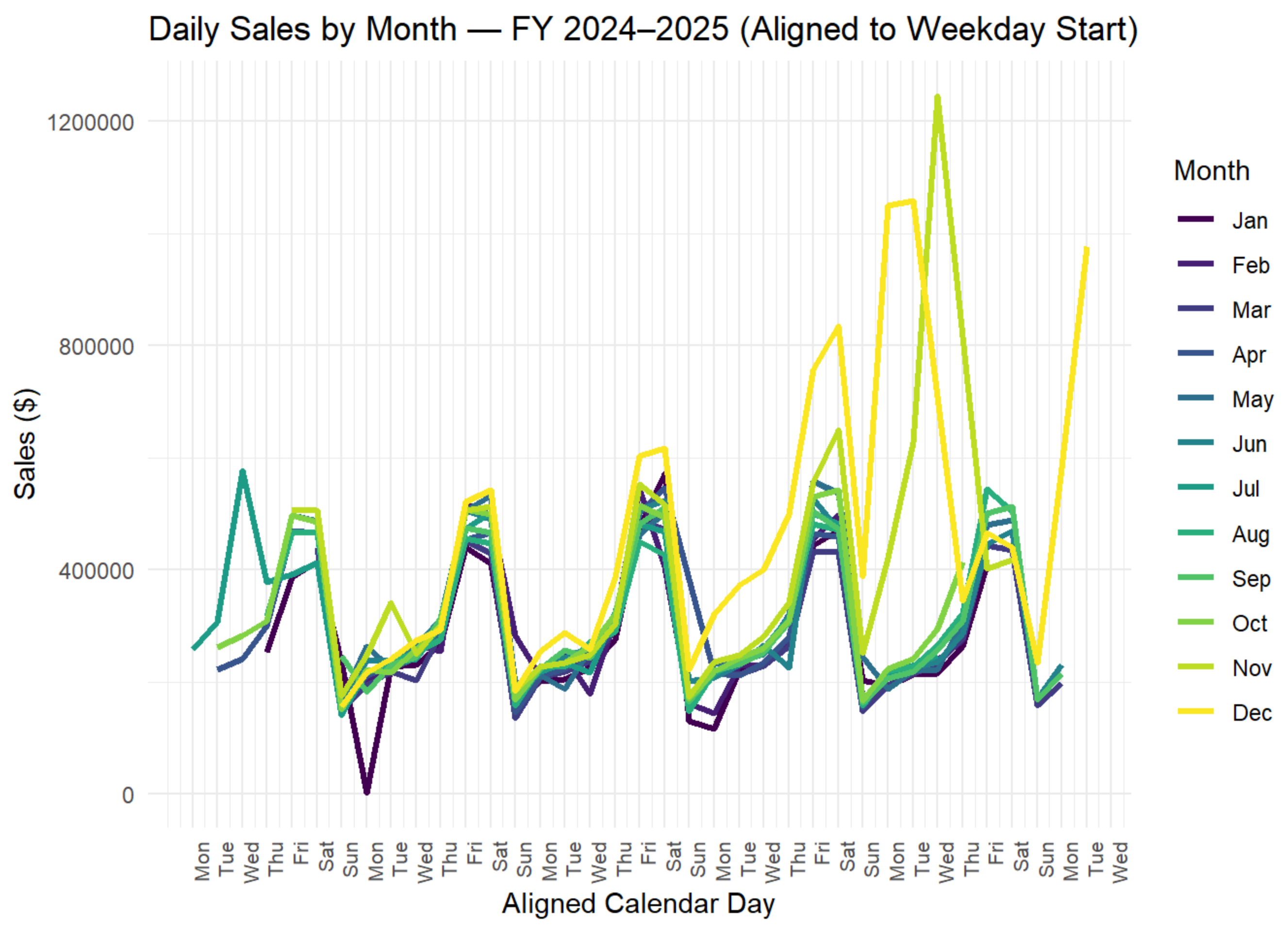
```
daily_sales <- transactions %>%
  group_by(TRANSDATE) %>%
  summarise(TotalSales = sum(NETAMOUNT, na.rm = TRUE), .groups = "drop") %>%
  mutate(
    Day   = day(TRANSDATE),
    Month = month(TRANSDATE, label = TRUE, abbr = TRUE),
    Year  = year(TRANSDATE)
  )

fy23_24 <- daily_sales %>% filter(TRANSDATE >= as.Date("2023-07-01") & TRANSDATE <= as.Date("202
4-06-30"))
fy24_25 <- daily_sales %>% filter(TRANSDATE >= as.Date("2024-07-01") & TRANSDATE <= as.Date("202
5-06-30"))

fy23_24_aligned <- fy23_24 %>% mutate(FirstOfMonth=floor_date(TRANSDATE,"month"),
                                      WeekdayOffset=wday(FirstOfMonth,week_start=1)-1,
                                      DayAligned=Day+WeekdayOffset)

fy24_25_aligned <- fy24_25 %>% mutate(FirstOfMonth=floor_date(TRANSDATE,"month"),
                                      WeekdayOffset=wday(FirstOfMonth,week_start=1)-1,
                                      DayAligned=Day+WeekdayOffset)
```

```r
weekday_labels <- rep(c("Mon","Tue","Wed","Thu","Fri","Sat","Sun"), length.out = 40)

ggplot(fy23_24_aligned, aes(x = DayAligned, y = TotalSales, color = Month, group = Month)) +
  geom_line(size = 1.1) +
  labs(
    title = "Daily Sales by Month — FY 2023-2024 (Aligned to Weekday Start)",
    x = "Aligned Calendar Day",
    y = "Sales ($)"
  ) +
  scale_x_continuous(
    breaks = seq(1, 40, by = 1),
    labels = weekday_labels
  ) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, size = 8))
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## ℹ Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



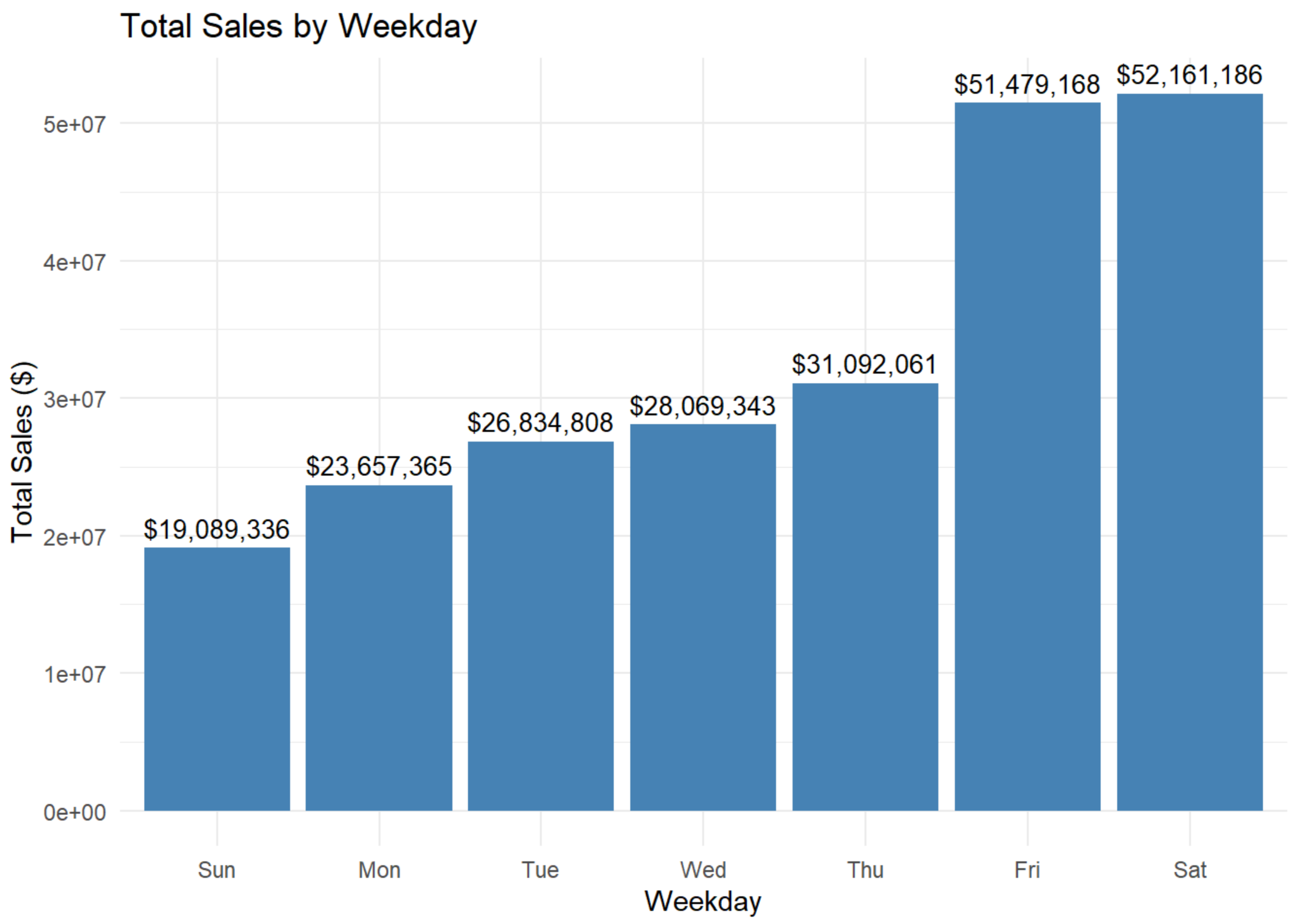
```
ggplot(fy24_25_aligned, aes(x = DayAligned, y = TotalSales, color = Month, group = Month)) +
  geom_line(size = 1.1) +
  labs(
    title = "Daily Sales by Month — FY 2024-2025 (Aligned to Weekday Start)",
    x = "Aligned Calendar Day",
    y = "Sales ($)"
  ) +
  scale_x_continuous(
    breaks = seq(1, 40, by = 1),
    labels = weekday_labels
  ) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, size = 8))
```



15) Weekday Sales Ranking

```
weekday_sales <- transactions %>%
  mutate(Weekday = wday(TRANSDATE, label = TRUE, abbr = TRUE)) %>%
  group_by(Weekday) %>%
  summarise(TotalSales = sum(NETAMOUNT, na.rm = TRUE), .groups = "drop") %>%
  arrange(TotalSales)
```

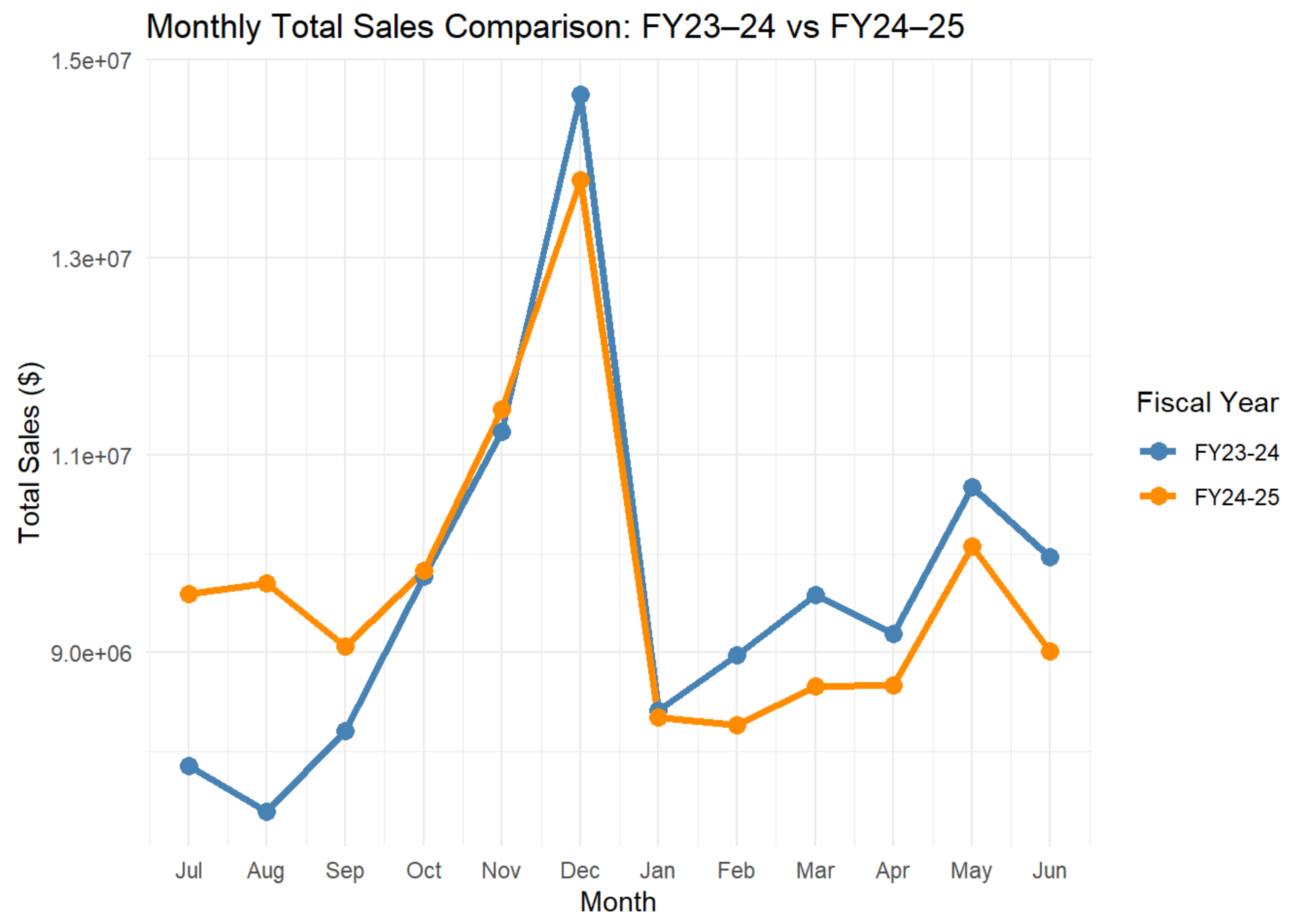
```
ggplot(weekday_sales, aes(x = reorder(Weekday, TotalSales), y = TotalSales)) +
  geom_col(fill = "steelblue") +
  geom_text(aes(label = scales::dollar_format()(round(TotalSales,0))),
            vjust = -0.5, size = 3.5) +
  labs(
    title = "Total Sales by Weekday",
    x = "Weekday", y = "Total Sales ($)"
  ) +
  theme_minimal()
```



16) Monthly Sales — Compare Fiscal Years

```r
transactions <- transactions %>%
  mutate(YearMonth = format(TRANSDATE, "%Y-%m"))

fy23_24_monthly <- transactions %>%
  filter(TRANSDATE >= "2023-07-01" & TRANSDATE <= "2024-06-30") %>%
  group_by(YearMonth) %>%
  summarise(TotalSales = sum(NETAMOUNT), .groups = "drop") %>%
  arrange(YearMonth) %>%
  mutate(FiscalYear="FY23-24", MonthIndex=row_number())

fy24_25_monthly <- transactions %>%
  filter(TRANSDATE >= "2024-07-01" & TRANSDATE <= "2025-06-30") %>%
  group_by(YearMonth) %>%
  summarise(TotalSales = sum(NETAMOUNT), .groups = "drop") %>%
  arrange(YearMonth) %>%
  mutate(FiscalYear="FY24-25", MonthIndex=row_number())

combined_monthly <- rbind(fy23_24_monthly, fy24_25_monthly)
```

```r
ggplot(combined_monthly, aes(x = MonthIndex, y = TotalSales, color = FiscalYear, group = FiscalY
ear)) +
  geom_line(linewidth = 1.3) +
  geom_point(size = 3) +
  scale_x_continuous(
    breaks = 1:12,
    labels = c("Jul","Aug","Sep","Oct","Nov","Dec","Jan","Feb","Mar","Apr","May","Jun")
  ) +
  scale_color_manual(values = c("FY23-24"="steelblue", "FY24-25"="darkorange")) +
  labs(
    title = "Monthly Total Sales Comparison: FY23–24 vs FY24–25",
    x = "Month", y = "Total Sales ($)", color = "Fiscal Year"
  ) +
  theme_minimal()
```



17) Store Summary — Sales, Transactions, Baskets (FY24-25)

```
fy24_25 <- transactions %>%
  filter(TRANSDATE >= "2024-07-01" & TRANSDATE <= "2025-06-30")

store_fy24_25_summary <- fy24_25 %>%
  group_by(STORENAME) %>%
  summarise(
    TotalSales        = sum(NETAMOUNT, na.rm=TRUE),
    TotalTransactions = n(),
    .groups = "drop"
  )

baskets_fy24_25 <- fy24_25 %>%
  group_by(STORENAME) %>%
  summarise(TotalBaskets = n_distinct(TRANSACTIONID), .groups = "drop")

store_fy24_25_final <- store_fy24_25_summary %>%
  left_join(baskets_fy24_25, by="STORENAME") %>%
  arrange(STORENAME)
```

18) Add Square Footage & Efficiency Metrics

```r
store_sqft <- readr::read_csv("Designation.csv") # expect columns: STORENAME, SquareFootage
```

```
## Rows: 27 Columns: 6
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## chr (4): STORENAME, Current Designation, SY 2023, SY 2024
## dbl (2): STORE, SquareFootage
##
## 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.
```

```r
store_fy24_25_final <- store_fy24_25_final %>%
  left_join(store_sqft %>% select(STORENAME, SquareFootage), by="STORENAME") %>%
  mutate(
    TotalSalesPerSqFt        = TotalSales        / SquareFootage,
    TotalTransactionsPerSqFt = TotalTransactions / SquareFootage,
    TotalBasketsPerSqFt      = TotalBaskets      / SquareFootage
  ) %>%
  arrange(STORENAME)

traffic_median <- median(store_fy24_25_final$TotalTransactionsPerSqFt, na.rm = TRUE)
sales_median   <- median(store_fy24_25_final$TotalSalesPerSqFt,        na.rm = TRUE)
basket_median  <- median(store_fy24_25_final$TotalBasketsPerSqFt,      na.rm = TRUE)
```

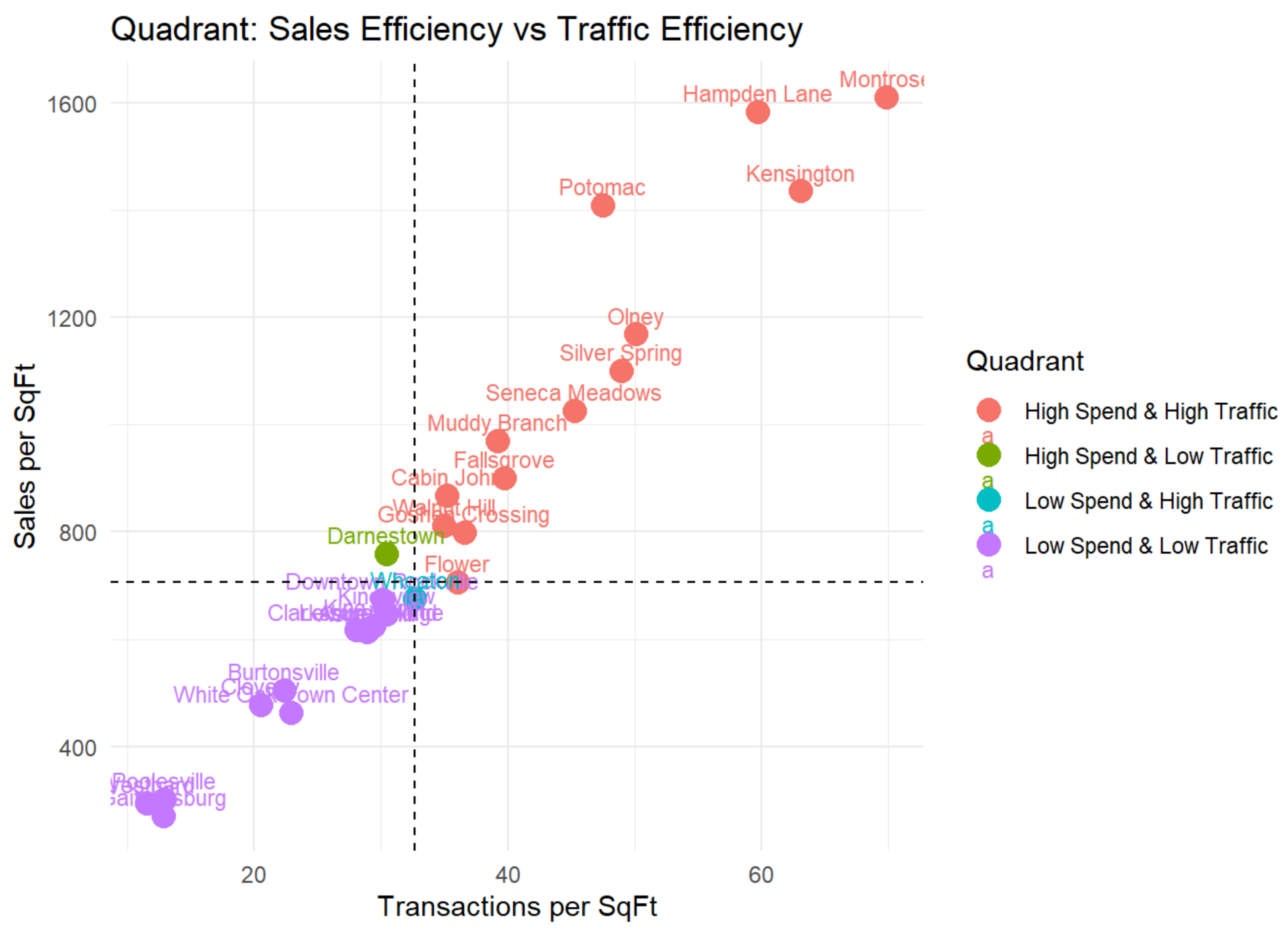
19) Basket Structure — Avg Basket $ and Items

```r
basket_metrics <- fy24_25 %>%
  group_by(STORENAME, TRANSACTIONID) %>%
  summarise(
    BasketTotal = sum(NETAMOUNT, na.rm = TRUE),
    Items       = sum(LINEQTY,  na.rm = TRUE),
    .groups = "drop"
  )

basket_summary <- basket_metrics %>%
  group_by(STORENAME) %>%
  summarise(
    AvgBasketValue    = mean(BasketTotal, na.rm = TRUE),
    AvgItemsPerBasket = mean(Items,       na.rm = TRUE),
    .groups = "drop"
  )

store_fy24_25_final <- store_fy24_25_final %>%
  left_join(basket_summary, by = "STORENAME") %>%
  arrange(STORENAME)
```

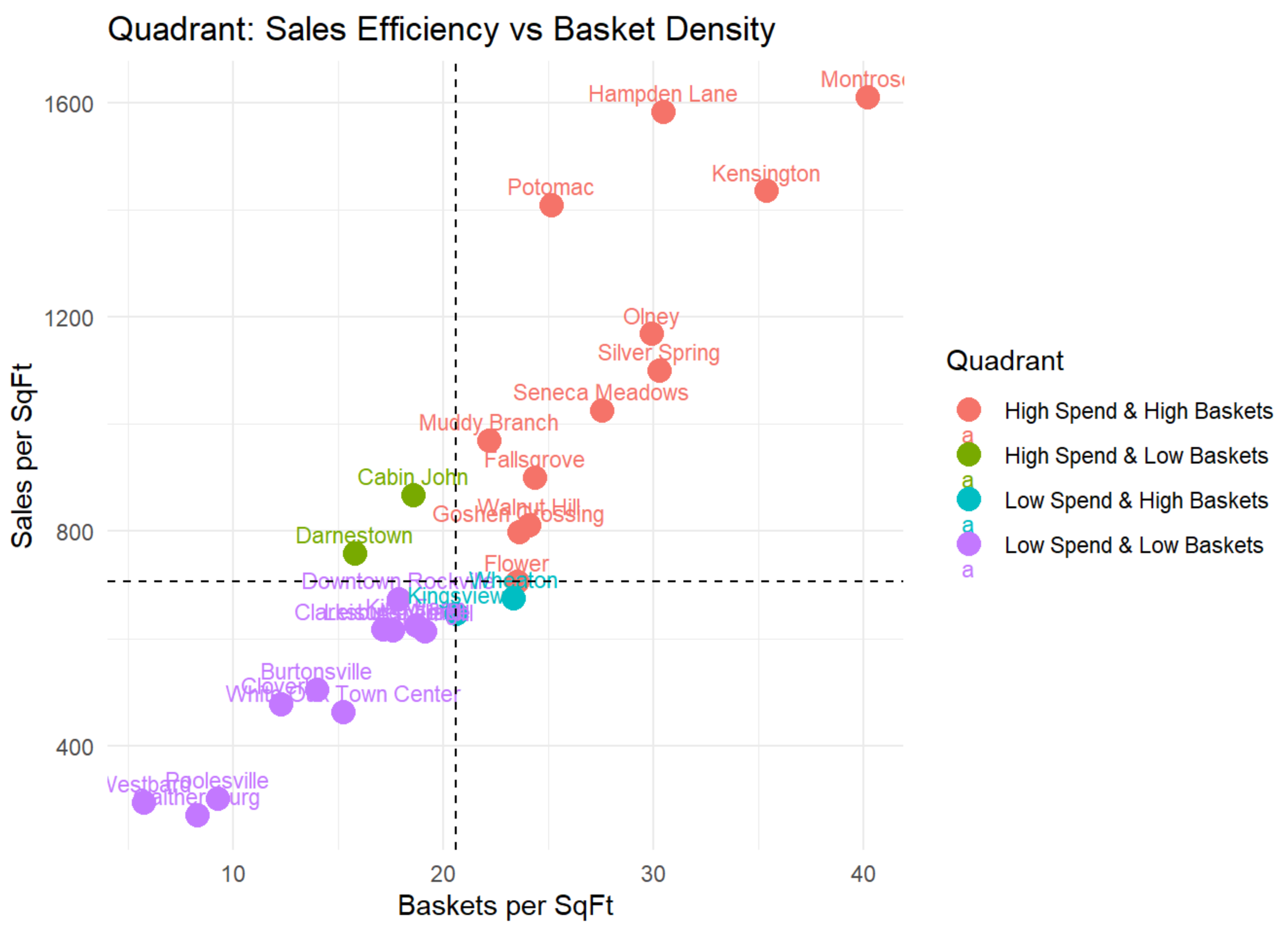
20) Quadrant Classification: Sales, Traffic & Basket Density

```
store_fy24_25_quad <- store_fy24_25_final %>%
  mutate(
    Quad_Sales_Traffic = case_when(
      TotalSalesPerSqFt >= sales_median  & TotalTransactionsPerSqFt >= traffic_median ~ "High Sp
end & High Traffic",
      TotalSalesPerSqFt >= sales_median  & TotalTransactionsPerSqFt <  traffic_median ~ "High Sp
end & Low Traffic",
      TotalSalesPerSqFt <  sales_median  & TotalTransactionsPerSqFt >= traffic_median ~ "Low Spe
nd & High Traffic",
      TRUE ~ "Low Spend & Low Traffic"
    ),
    Quad_Sales_Baskets = case_when(
      TotalSalesPerSqFt >= sales_median  & TotalBasketsPerSqFt >= basket_median ~ "High Spend &
High Baskets",
      TotalSalesPerSqFt >= sales_median  & TotalBasketsPerSqFt <  basket_median ~ "High Spend &
Low Baskets",
      TotalSalesPerSqFt <  sales_median  & TotalBasketsPerSqFt >= basket_median ~ "Low Spend & H
igh Baskets",
      TRUE ~ "Low Spend & Low Baskets"
    ),
    Quad_Traffic_Baskets = case_when(
      TotalTransactionsPerSqFt >= traffic_median & TotalBasketsPerSqFt >= basket_median ~ "High
Traffic & High Baskets",
      TotalTransactionsPerSqFt >= traffic_median & TotalBasketsPerSqFt <  basket_median ~ "High
Traffic & Low Baskets",
      TotalTransactionsPerSqFt <  traffic_median & TotalBasketsPerSqFt >= basket_median ~ "Low T
raffic & High Baskets",
      TRUE ~ "Low Traffic & Low Baskets"
    )
  )
```

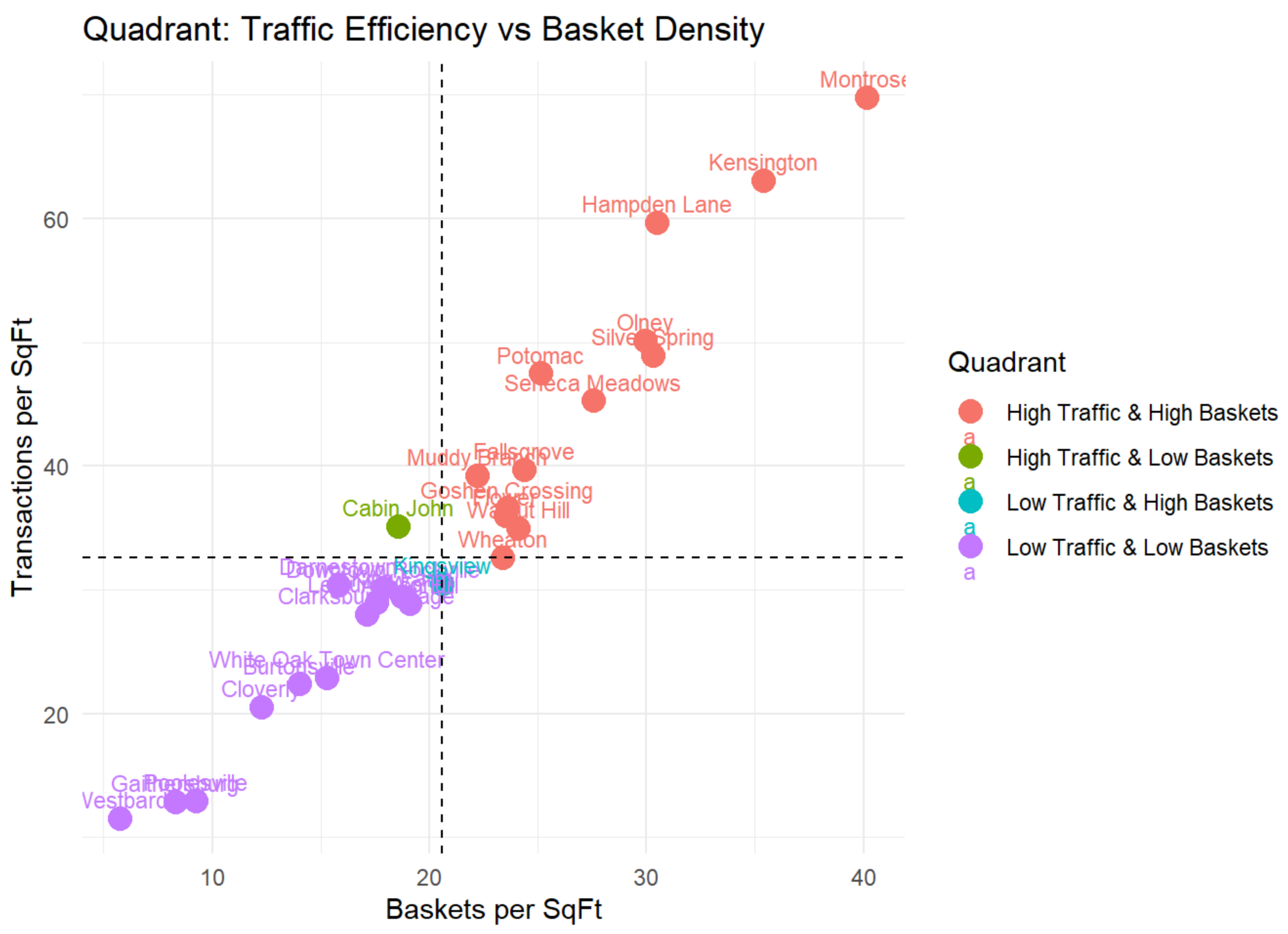
```
# Sales vs Traffic

ggplot(store_fy24_25_quad, aes(x = TotalTransactionsPerSqFt,
                               y = TotalSalesPerSqFt,
                               color = Quad_Sales_Traffic,
                               label = STORENAME)) +
  geom_point(size = 4) +
  geom_text(vjust = -0.6, size = 3) +
  geom_vline(xintercept = traffic_median, linetype = "dashed") +
  geom_hline(yintercept = sales_median, linetype = "dashed") +
  labs(
    title = "Quadrant: Sales Efficiency vs Traffic Efficiency",
    x = "Transactions per SqFt",
    y = "Sales per SqFt",
    color = "Quadrant"
  ) +
  theme_minimal()
```



```
# Sales vs Basket

ggplot(store_fy24_25_quad, aes(x = TotalBasketsPerSqFt,
                               y = TotalSalesPerSqFt,
                               color = Quad_Sales_Baskets,
                               label = STORENAME)) +
  geom_point(size = 4) +
  geom_text(vjust = -0.6, size = 3) +
  geom_vline(xintercept = basket_median, linetype = "dashed") +
  geom_hline(yintercept = sales_median, linetype = "dashed") +
  labs(
    title = "Quadrant: Sales Efficiency vs Basket Density",
    x = "Baskets per SqFt",
    y = "Sales per SqFt",
    color = "Quadrant"
  ) +
  theme_minimal()
```



```
# Traffic vs Basket

ggplot(store_fy24_25_quad, aes(x = TotalBasketsPerSqFt,
                                y = TotalTransactionsPerSqFt,
                                color = Quad_Traffic_Baskets,
                                label = STORENAME)) +
  geom_point(size = 4) +
  geom_text(vjust = -0.6, size = 3) +
  geom_vline(xintercept = basket_median, linetype = "dashed") +
  geom_hline(yintercept = traffic_median, linetype = "dashed") +
  labs(
    title = "Quadrant: Traffic Efficiency vs Basket Density",
    x = "Baskets per SqFt",
    y = "Transactions per SqFt",
    color = "Quadrant"
  ) +
  theme_minimal()
```



21) Demographics vs Store Performance

```
demographics <- read.csv(
  "Demographics.csv",
  header = TRUE, sep = ",", encoding = "UTF-8", quote = "\""
)

demographics_selected <- demographics %>%
  select(STORENAME, SquareFootage, TotalPopulation, White, Black, Hispanic,
         X25YearsOld, PovertyLevel, TwicePovertyLevel)

store_fy24_25_summary <- store_fy24_25_summary %>%
  left_join(demographics_selected, by = "STORENAME")

# 15a) Size vs Sales

cor(store_fy24_25_summary$TotalSales, store_fy24_25_summary$SquareFootage, use = "complete.obs")
```

```
## [1] 0.2980989
```

```
summary(lm(TotalSales ~ SquareFootage, data = store_fy24_25_summary))
```

```
##
## Call:
## lm(formula = TotalSales ~ SquareFootage, data = store_fy24_25_summary)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3101902 -1229319  -248714  1016217  3906812
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2296413.7  1340801.1   1.713   0.0991 .
## SquareFootage     370.2      237.1   1.561   0.1310
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1866000 on 25 degrees of freedom
## Multiple R-squared:  0.08886,    Adjusted R-squared:  0.05242
## F-statistic: 2.438 on 1 and 25 DF,  p-value: 0.131
```

```
# 15b) Population vs Sales

cor(store_fy24_25_summary$TotalSales, store_fy24_25_summary$TotalPopulation, use = "complete.ob
s")
```

```
## [1] 0.1063146
```

```
summary(lm(TotalSales ~ TotalPopulation, data = store_fy24_25_summary))
```

```
##
## Call:
## lm(formula = TotalSales ~ TotalPopulation, data = store_fy24_25_summary)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3387751 -1449593   253560  1245434  3590859
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     3.898e+06  8.626e+05   4.519  0.00013 ***
## TotalPopulation 1.619e+01  3.028e+01   0.535  0.59765
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1944000 on 25 degrees of freedom
## Multiple R-squared:  0.0113, Adjusted R-squared:  -0.02825
## F-statistic: 0.2858 on 1 and 25 DF,  p-value: 0.5976
```

```
# 15c) 25+ vs Sales

cor(store_fy24_25_summary$TotalSales, store_fy24_25_summary$X25YearsOld, use = "complete.obs")
```

```
## [1] 0.1362926
```

```
summary(lm(TotalSales ~ X25YearsOld, data = store_fy24_25_summary))
```

```
##
## Call:
## lm(formula = TotalSales ~ X25YearsOld, data = store_fy24_25_summary)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3304986 -1390813   160750  1286241  3519849
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 3.786e+06  8.528e+05   4.439 0.000159 ***
## X25YearsOld 2.950e+01  4.288e+01   0.688 0.497860
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1937000 on 25 degrees of freedom
## Multiple R-squared:  0.01858,    Adjusted R-squared:  -0.02068
## F-statistic: 0.4732 on 1 and 25 DF,  p-value: 0.4979
```

```
# 15d) Poverty vs Sales

cor(store_fy24_25_summary$TotalSales, store_fy24_25_summary$PovertyLevel, use = "complete.obs")
```

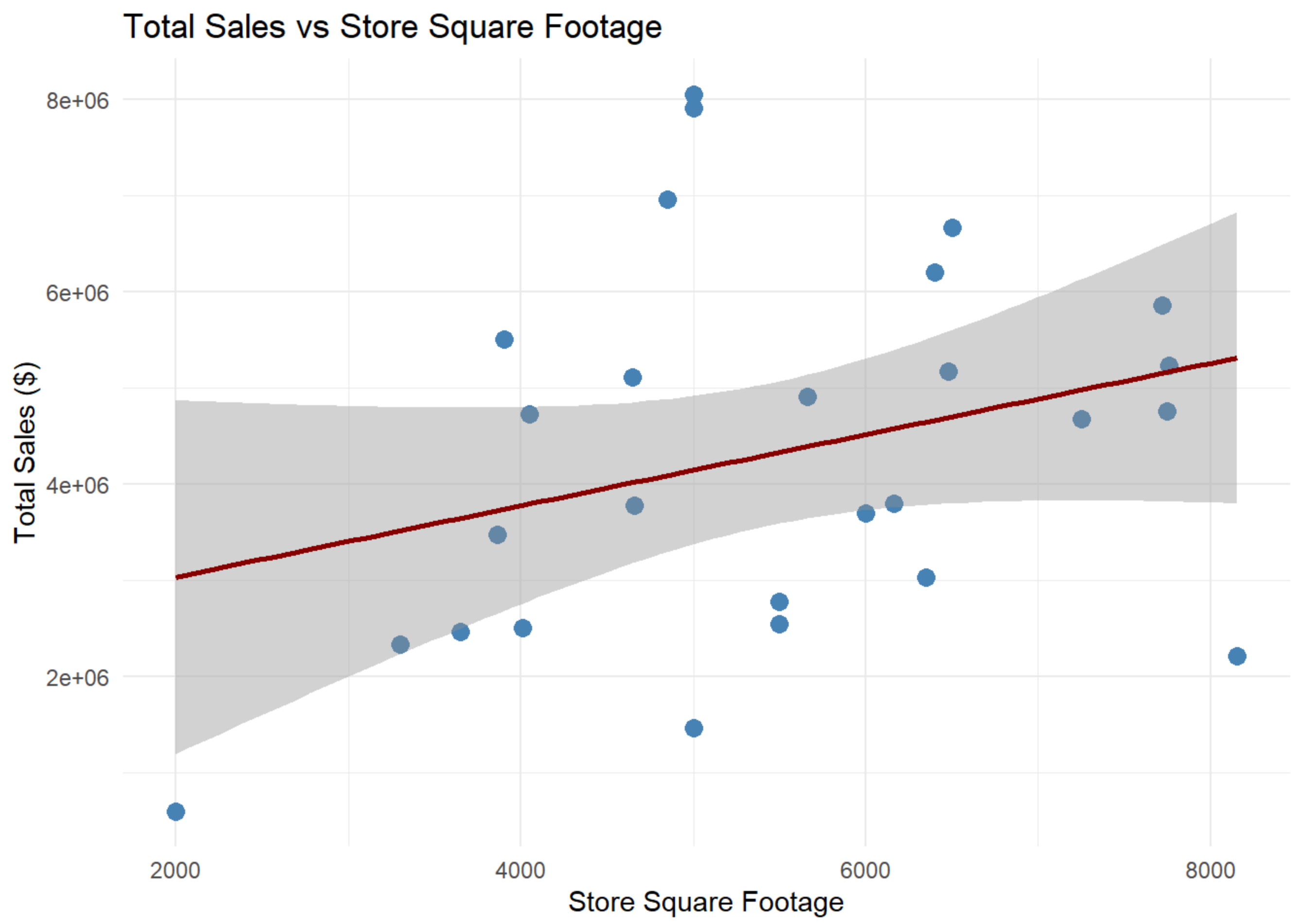
```
## [1] -0.1449147
```

```
summary(lm(TotalSales ~ PovertyLevel, data = store_fy24_25_summary))
```

```
##
## Call:
## lm(formula = TotalSales ~ PovertyLevel, data = store_fy24_25_summary)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3977253 -1489550   216847   984044  3953313
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4643187.6   584163.0   7.948 2.64e-08 ***
## PovertyLevel    -155.4      212.2  -0.732    0.471
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1935000 on 25 degrees of freedom
## Multiple R-squared:  0.021,  Adjusted R-squared:  -0.01816
## F-statistic: 0.5363 on 1 and 25 DF,  p-value: 0.4708
```

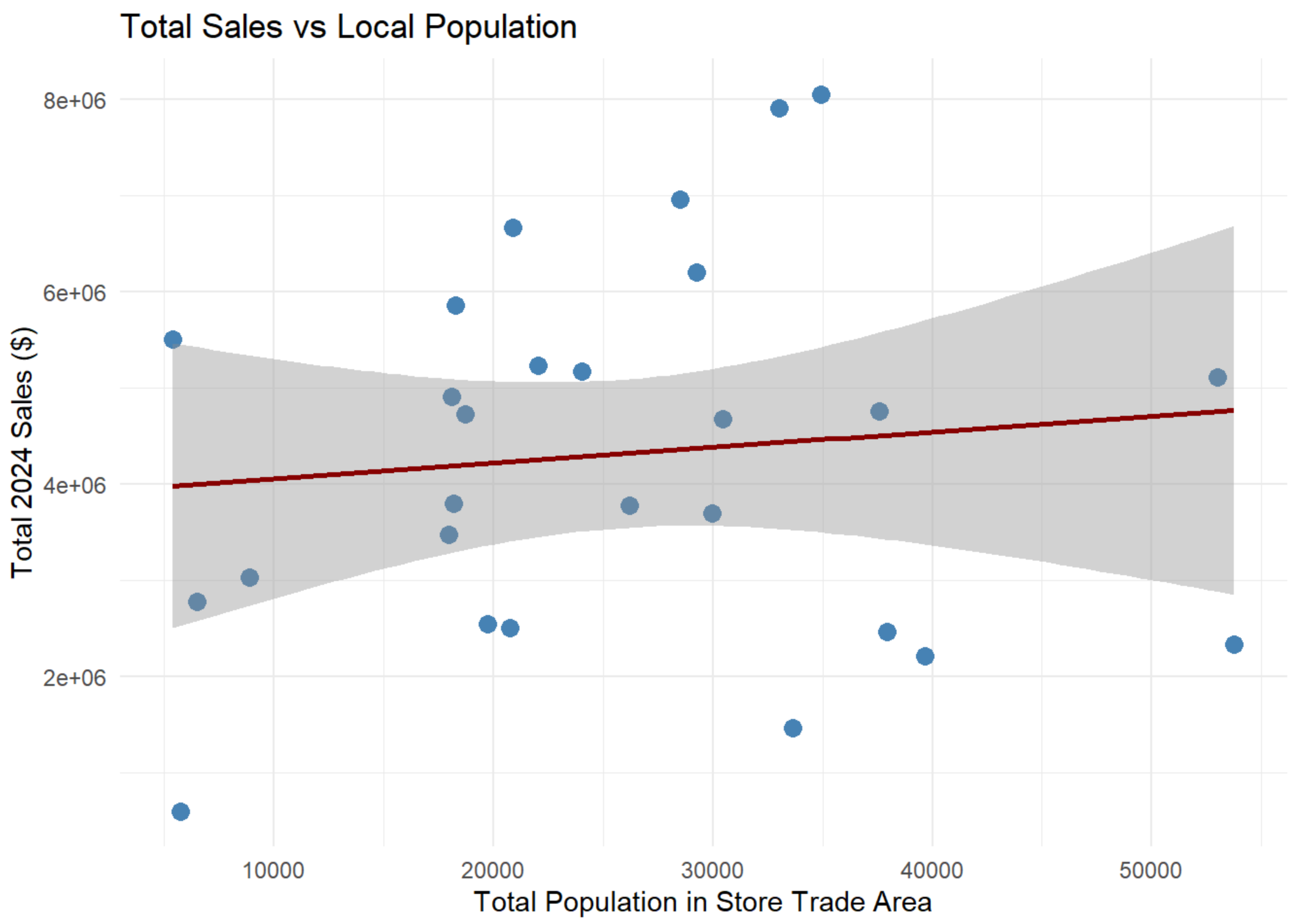
```
ggplot(store_fy24_25_summary,
       aes(x = SquareFootage, y = TotalSales)) +
  geom_point(size = 3, color = "steelblue") +
  geom_smooth(method = "lm", se = TRUE, color = "darkred") +
  labs(
    title = "Total Sales vs Store Square Footage",
    x = "Store Square Footage",
    y = "Total Sales ($)"
  ) +
  theme_minimal()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



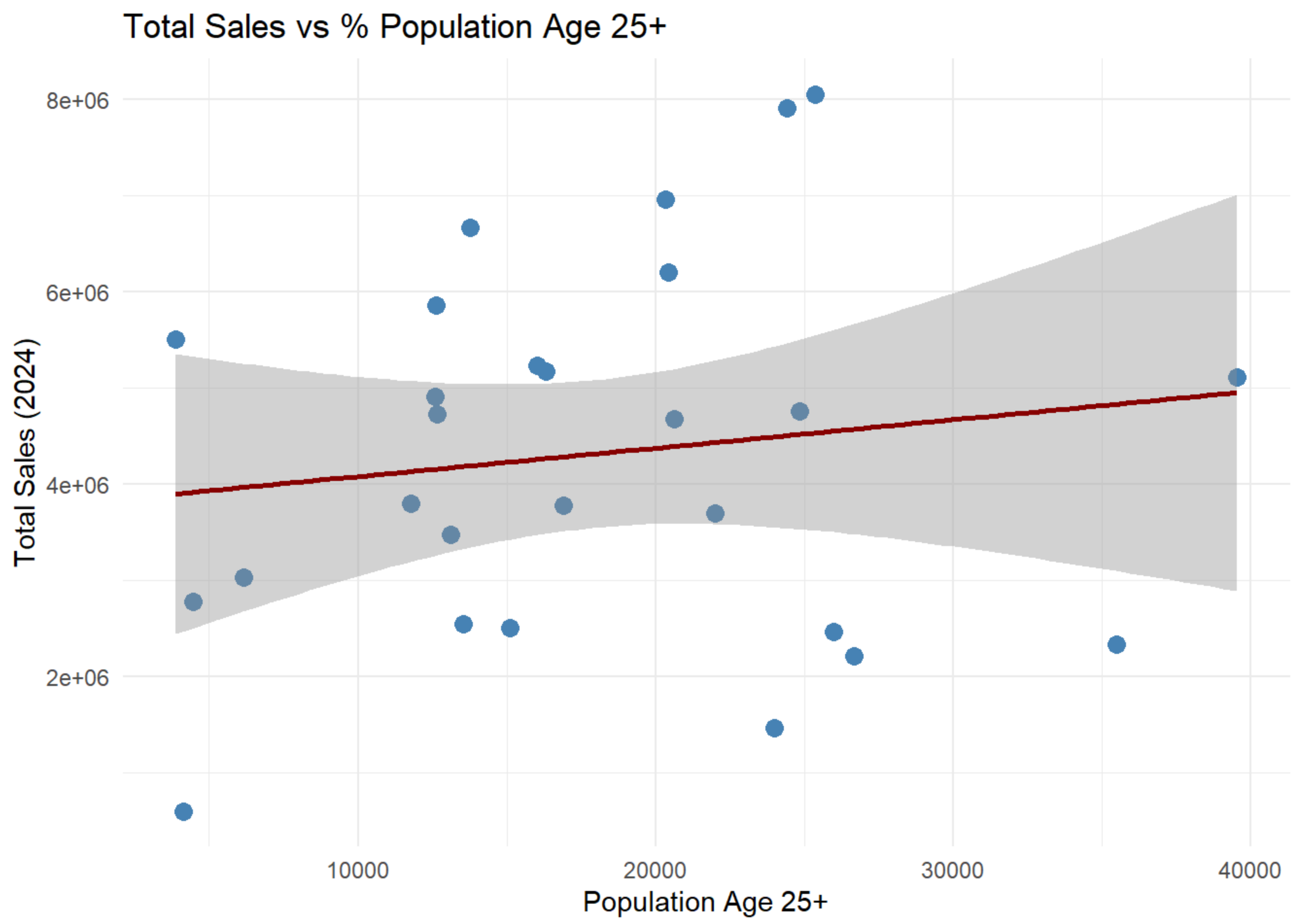
```
ggplot(store_fy24_25_summary,
       aes(x = TotalPopulation, y = TotalSales)) +
  geom_point(size = 3, color = "steelblue") +
  geom_smooth(method = "lm", se = TRUE, color = "darkred") +
  labs(
    title = "Total Sales vs Local Population",
    x = "Total Population in Store Trade Area",
    y = "Total 2024 Sales ($)"
  ) +
  theme_minimal()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



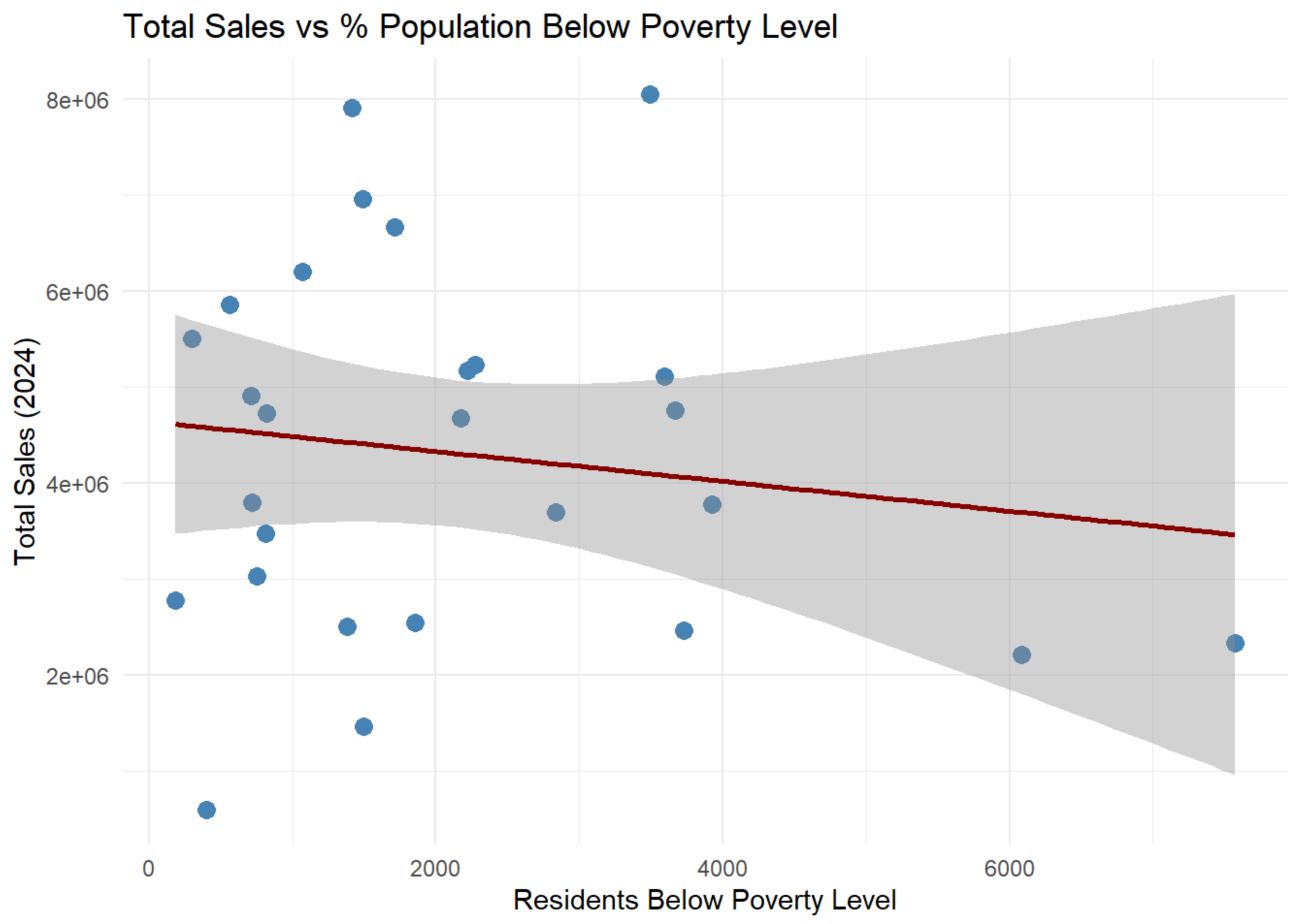
```
ggplot(store_fy24_25_summary,
       aes(x = X25YearsOld, y = TotalSales)) +
  geom_point(size = 3, color = "steelblue") +
  geom_smooth(method = "lm", se = TRUE, color = "darkred") +
  labs(
    title = "Total Sales vs % Population Age 25+",
    x = "Population Age 25+",
    y = "Total Sales (2024)"
  ) +
  theme_minimal()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



```
ggplot(store_fy24_25_summary,
       aes(x = PovertyLevel, y = TotalSales)) +
  geom_point(size = 3, color = "steelblue") +
  geom_smooth(method = "lm", se = TRUE, color = "darkred") +
  labs(
    title = "Total Sales vs % Population Below Poverty Level",
    x = "Residents Below Poverty Level",
    y = "Total Sales (2024)"
  ) +
  theme_minimal()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```


Total Sales vs % Population Below Poverty Level
Total Sales (2024)
8e+06
6e+06
4e+06
2e+06
0
2000
4000
6000
Residents Below Poverty Level

22) Market Basket Analysis (Apriori)

```
# Aggregate items per basket

baskets <- transactions %>%
  group_by(TRANSACTIONID) %>%
  summarise(
    Items      = paste(unique(ITEMID), collapse = ", "),
    Item_Count = n_distinct(ITEMID),
    .groups = "drop"
  ) %>%
  filter(Item_Count > 1) # keep multi-item baskets only

# Convert to list-of-vectors for 'arules'

basket_list <- baskets %>%
  mutate(Items = str_split(Items, ",\\s*")) %>%
  pull(Items)

basket_trans <- as(basket_list, "transactions")

# Apriori parameters: tune these by business tolerance

rules <- apriori(
  basket_trans,
  parameter = list(supp = 0.001, conf = 0.2, minlen = 2)
)
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.2    0.1    1 none FALSE            TRUE       5   0.001      2
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 2247
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[3454 item(s), 2247333 transaction(s)] done [2.48s].
## sorting and recoding items ... [836 item(s)] done [0.09s].
## creating transaction tree ... done [3.60s].
## checking subsets of size 1 2 3 done [0.07s].
## writing ... [14 rule(s)] done [0.00s].
## creating S4 object  ... done [0.16s].
```

```
# Inspect top rules by lift

inspect(head(sort(rules, by = "lift"), 20))
```

```
##      lhs         rhs      support     confidence coverage    lift     count
## [1]  {241462} => {241470} 0.001038564 0.2410410  0.004308663 54.57928 2334
## [2]  {241470} => {241462} 0.001038564 0.2351637  0.004416346 54.57928 2334
## [3]  {70488}  => {35866}  0.001234352 0.3464036  0.003563335 52.73212 2774
## [4]  {76300}  => {76290}  0.001952982 0.3376933  0.005783300 48.71048 4389
## [5]  {76290}  => {76300}  0.001952982 0.2817073  0.006932662 48.71048 4389
## [6]  {76300}  => {72664}  0.001497330 0.2589059  0.005783300 42.52030 3365
## [7]  {72664}  => {76300}  0.001497330 0.2459076  0.006088995 42.52030 3365
## [8]  {76293}  => {72664}  0.001102195 0.2344090  0.004702018 38.49716 2477
## [9]  {76290}  => {72664}  0.001588105 0.2290757  0.006932662 37.62127 3569
## [10] {72664}  => {76290}  0.001588105 0.2608156  0.006088995 37.62127 3569
## [11] {76293}  => {76290}  0.001124444 0.2391407  0.004702018 34.49479 2527
## [12] {88590}  => {82889}  0.001402996 0.3012612  0.004657076 15.83187 3153
## [13] {35289}  => {35211}  0.001349155 0.2126227  0.006345299 14.80050 3032
## [14] {44431}  => {82889}  0.002041976 0.2071316  0.009858352 10.88517 4589
```

23) Top/Bottom SKUs — Store & District

```
# Scope: FY24-25 window

fy24_25 <- transactions %>%
  filter(TRANSDATE >= "2024-07-01" & TRANSDATE <= "2025-06-30")

# Top 5 by qty per store

top5_items_per_store <- fy24_25 %>%
  group_by(STORENAME, ITEMID, DESCRIPTION) %>%
  summarise(
    total_qty = sum(TOTALQTY, na.rm = TRUE),
    total_sales = sum(NETAMOUNT, na.rm = TRUE)
  ) %>%
  ungroup() %>%
  group_by(STORENAME) %>%
  slice_max(total_qty, n = 5, with_ties = FALSE) %>%
  arrange(STORENAME, desc(total_qty))
```

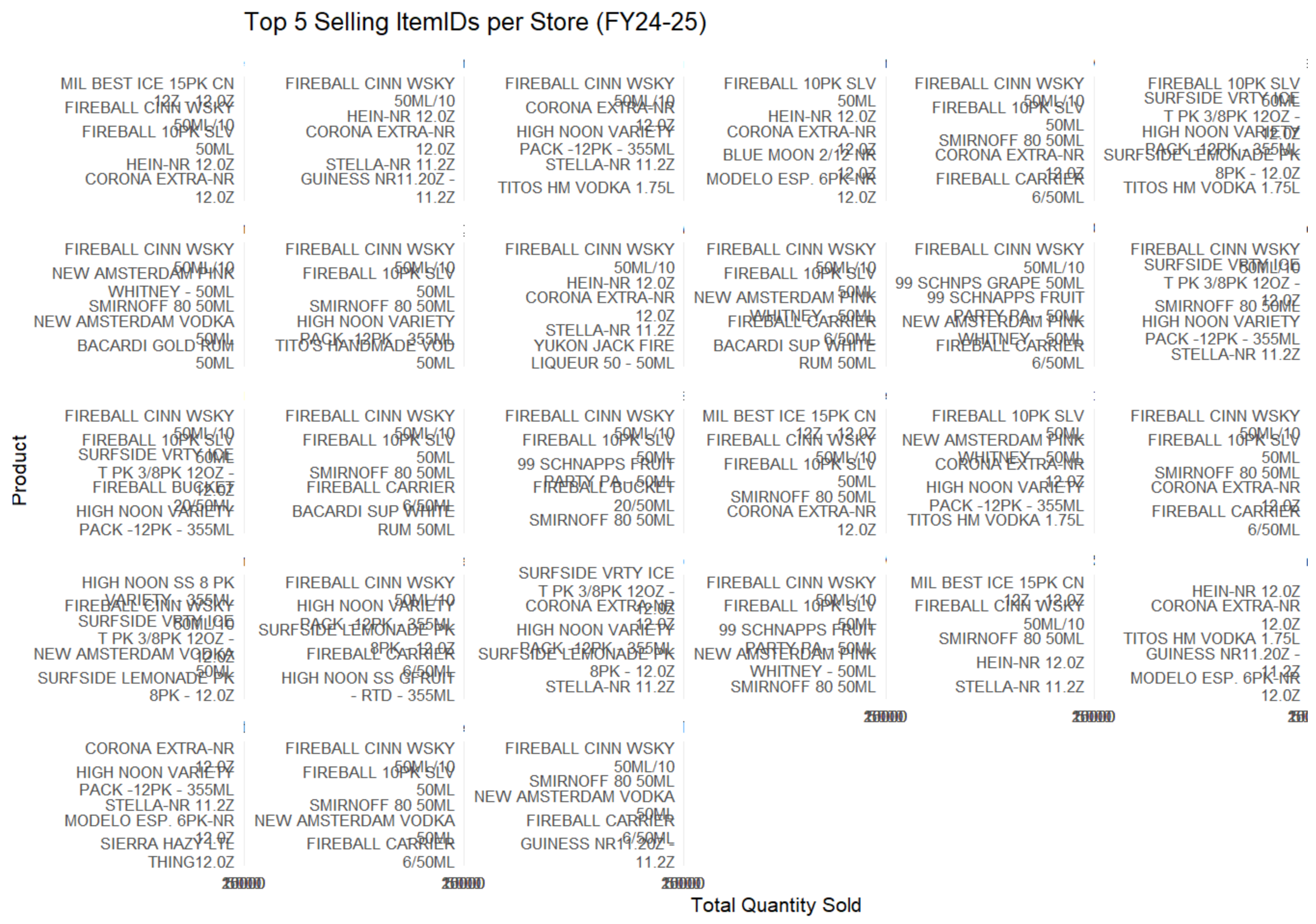
```
## `summarise()` has grouped output by 'STORENAME', 'ITEMID'. You can override
## using the `.groups` argument.
```

```
# Top 30 overall by qty

top30_items_all_stores <- fy24_25 %>%
  group_by(ITEMID, DESCRIPTION) %>%
  summarise(
    total_qty   = sum(TOTALQTY,  na.rm = TRUE),
    total_sales = sum(NETAMOUNT, na.rm = TRUE),
    .groups = "drop"
  ) %>%
  slice_max(total_qty, n = 30, with_ties = FALSE) %>%
  arrange(desc(total_qty))

# Bottom 5 by qty per store (stock only)

bottom5_items_per_stores <- fy24_25 %>%
  filter(ITEMTAG == "ST") %>%
  group_by(STORENAME, ITEMID, DESCRIPTION) %>%
  summarise(
    total_qty = sum(TOTALQTY, na.rm = TRUE),
    total_sales = sum(NETAMOUNT, na.rm = TRUE)
  ) %>%
  ungroup() %>%
  group_by(STORENAME) %>%
  slice_min(total_qty, n = 5, with_ties = FALSE) %>%
  arrange(STORENAME, total_qty)
```

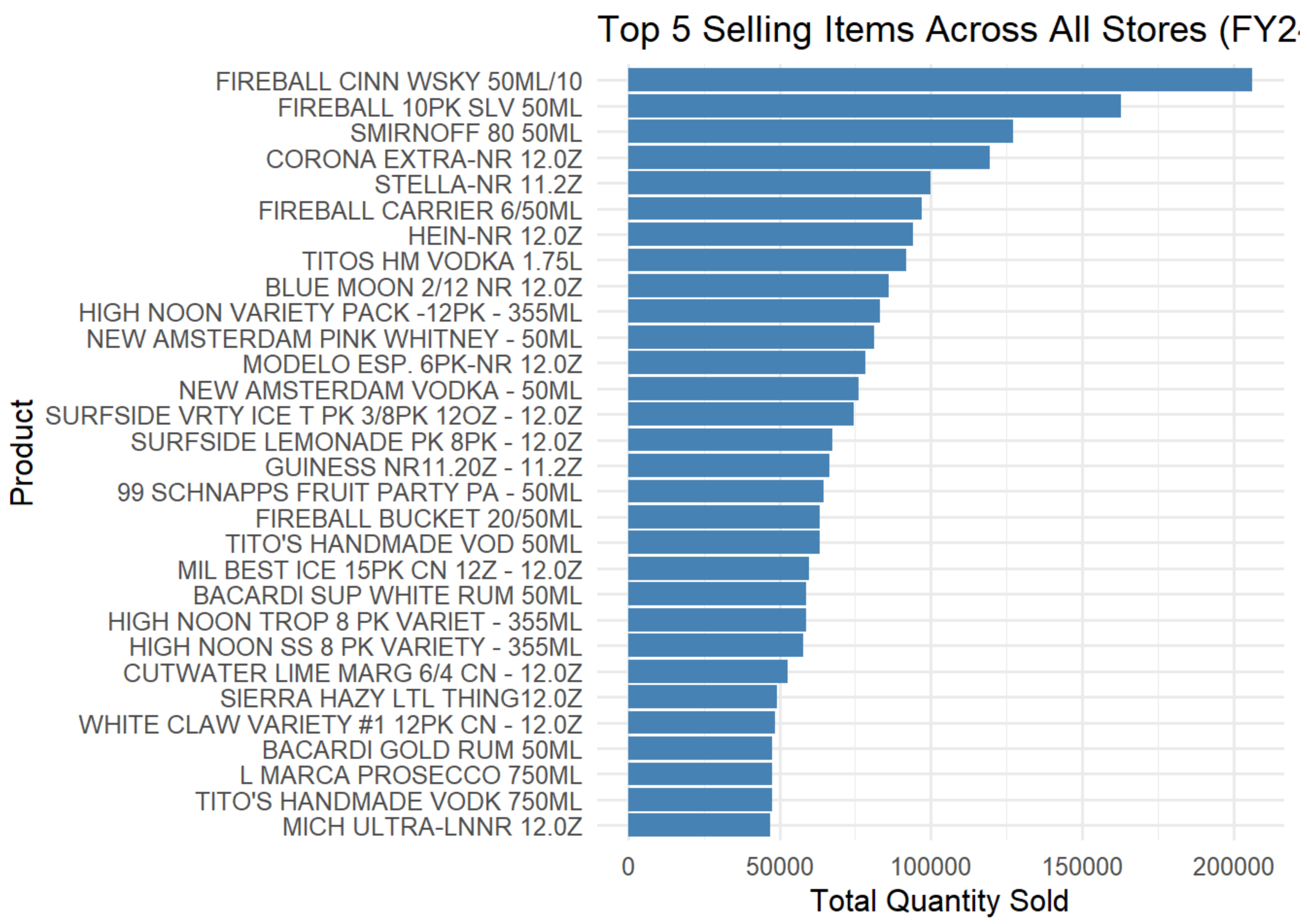
```
## `summarise()` has grouped output by 'STORENAME', 'ITEMID'. You can override
## using the `.groups` argument.
```

```
# Bottom 30 overall (stock only)

bottom30_items_all_stores <- fy24_25 %>%
  filter(ITEMTAG == "ST") %>%
  group_by(ITEMID, DESCRIPTION) %>%
  summarise(
    total_qty   = sum(TOTALQTY,  na.rm = TRUE),
    total_sales = sum(NETAMOUNT, na.rm = TRUE),
    .groups = "drop"
  ) %>%
  slice_min(total_qty, n = 30, with_ties = FALSE) %>%
  arrange(total_qty)
```

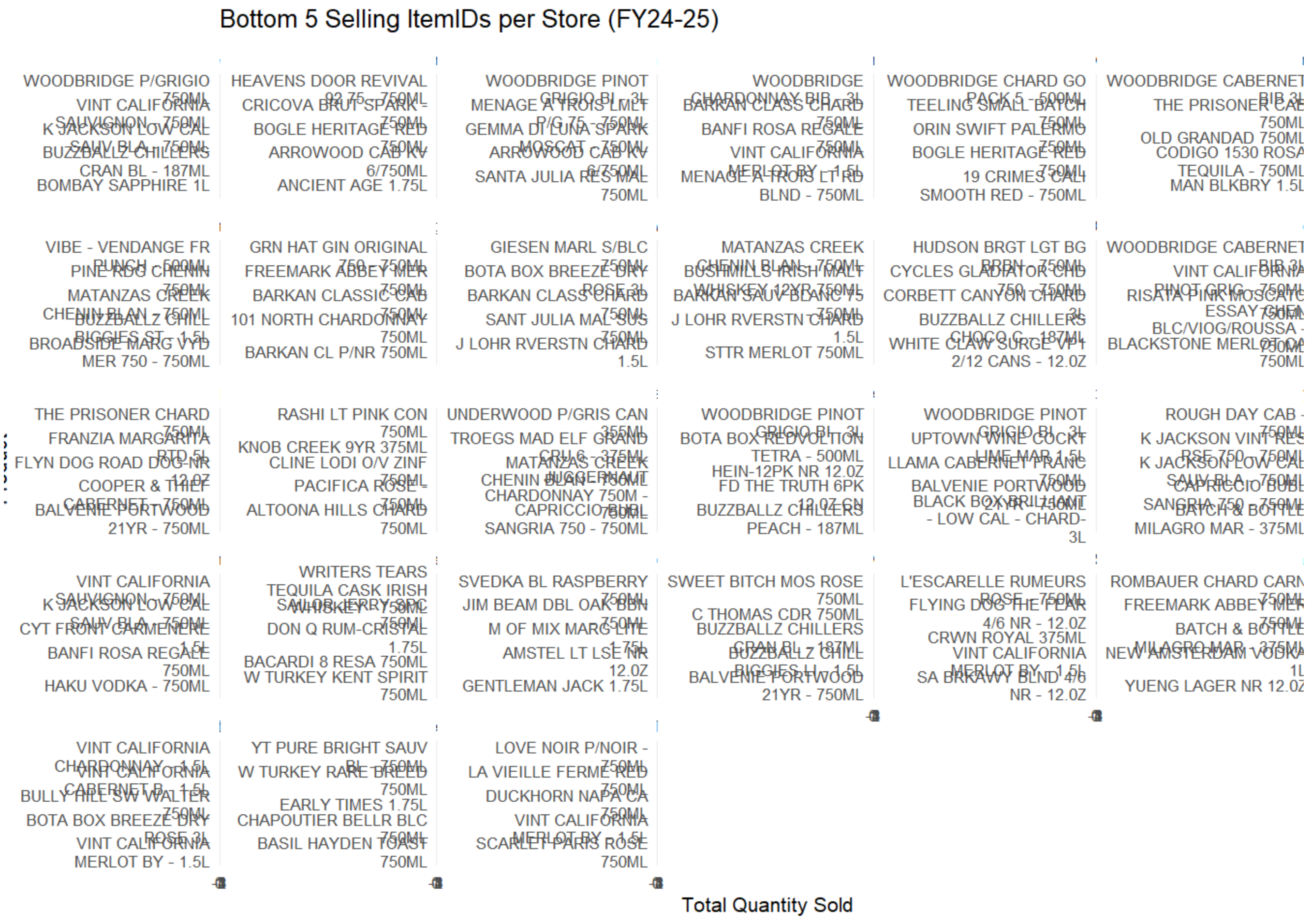
```
ggplot(top5_items_per_store,
       aes(x = reorder(DESCRIPTION, total_qty),
           y = total_qty)) +
  geom_col(fill = "#2E86C1") +
  coord_flip() +
  facet_wrap(~ STORENAME, scales = "free_y") +
  labs(
    title = "Top 5 Selling ItemIDs per Store (FY24-25)",
    x = "Product",
    y = "Total Quantity Sold"
  ) +
  scale_x_discrete(labels = function(x) str_wrap(x, width = 20)) +
  theme_minimal(base_size = 8)
```



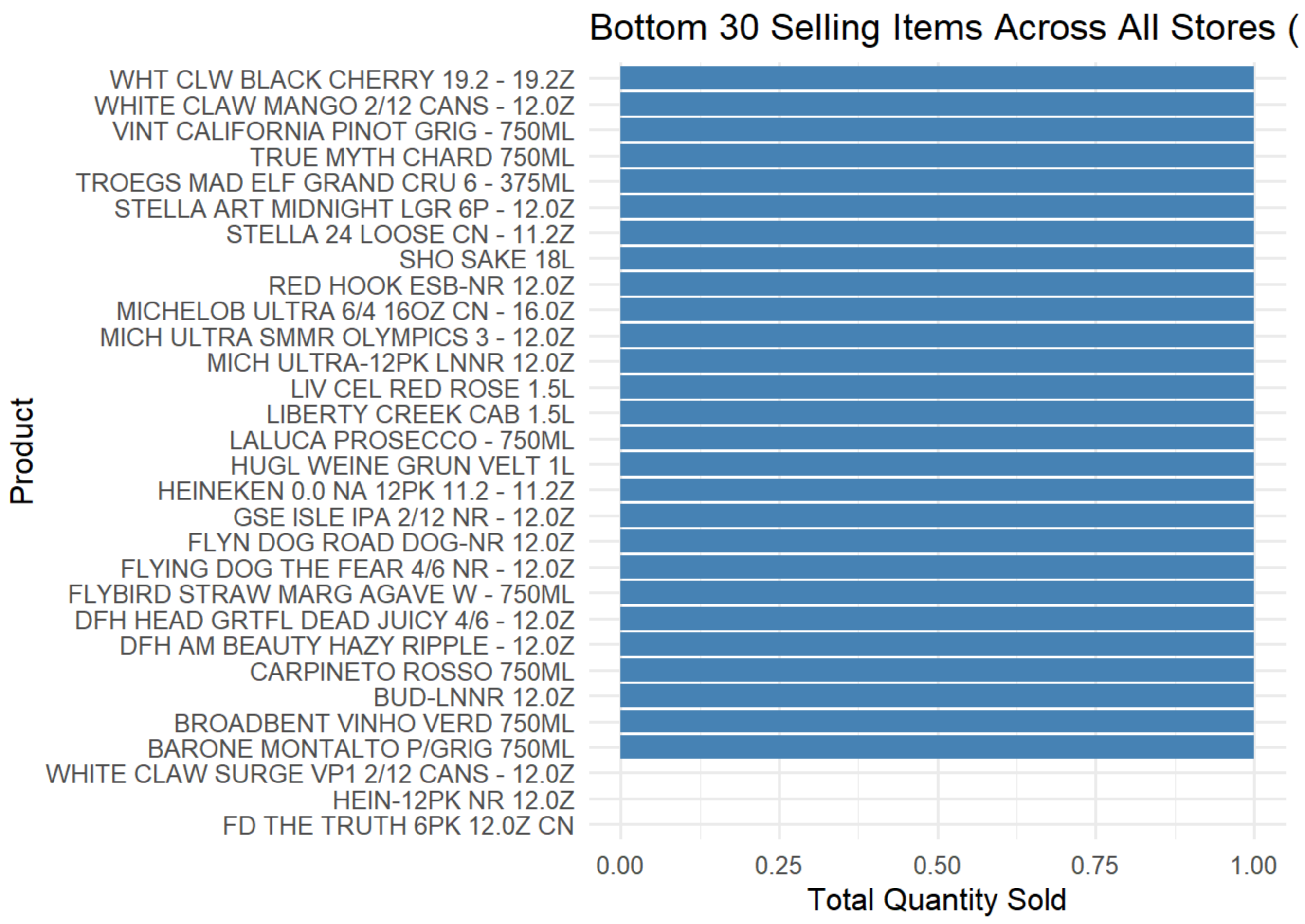
```
ggplot(top30_items_all_stores,
       aes(x = reorder(DESCRIPTION, total_qty),
           y = total_qty)) +
  geom_col(fill = "steelblue") +
  coord_flip() +
  labs(
    title = "Top 5 Selling Items Across All Stores (FY24-25)",
    x = "Product",
    y = "Total Quantity Sold"
  ) +
  theme_minimal(base_size = 12)
```



```r
ggplot(bottom5_items_per_stores,
       aes(x = reorder(DESCRIPTION, total_qty),
           y = total_qty)) +
  geom_col(fill = "#2E86C1") +
  coord_flip() +
  facet_wrap(~ STORENAME, scales = "free_y") +
  labs(
    title = "Bottom 5 Selling ItemIDs per Store (FY24-25)",
    x = "Product",
    y = "Total Quantity Sold"
  ) +
  scale_x_discrete(labels = function(x) str_wrap(x, width = 20)) +
  theme_minimal(base_size = 8)
```



```
ggplot(bottom30_items_all_stores,
       aes(x = reorder(DESCRIPTION, total_qty),
           y = total_qty)) +
  geom_col(fill = "steelblue") +
  coord_flip() +
  labs(
    title = "Bottom 30 Selling Items Across All Stores (FY24-25)",
    x = "Product",
    y = "Total Quantity Sold"
  ) +
  theme_minimal(base_size = 12)
```



Data Processing Flowchart

```
Clean CSV (TransactionData_Clean.csv)
   ↓
Load & Validate
   ↓
Size/Mix/Category KPIs
   ↓
Weekday/Weekend/Holiday Effects
   ↓
Time Series FY vs FY
   ↓
Store KPIs + SqFt Efficiency
   ↓
Demographics Correlations
   ↓
MBA (Apriori) — Cross-Sell
   ↓
Top/Bottom SKU Reports
```