Swire Coca Cola Exploratory Data Analysis

Varun Selvam

2025-23-01

EDA Overview

Background:

SCCU(Swire Coca-Cola United States) tries to optimize logistics by transitioning customers selling below a specific annual volume to an Alternate Route to Market (ARTM). There is an annual 400 gallons volume threshold used to distinguish between the direct delivery route and ARTM. However, SCCU is looking for a more cost-efficient strategy to decide new threshold for optimizing logistics which is driving better operational efficiency and more revenues.

Requirement:

1. The analysis will focus on classifying which customers must be included in ARTM or Direct route, and which volume threshold would be optimal to decide for the classification.
2. The analysis will focus on two key customer segments.

- 1st Group: Local Market Partners that buy fountains only: Customers who buy only fountain drinks and no CO2, cans, or bottles.
- 2nd Group: This group includes all customers, regardless of whether they are local market partners or not, and includes those purchasing CO2, cans, bottles, or fountain drinks.

Questions:

- What factors or characteristics distinguish customers with annual sales exceeding the determined volume threshold from those below this threshold?
- How can SCCU uses historical sales data, or other Customer Characteristics to predict which ARTM customers have the potential to grow beyond the volume threshold annually?
- How can these insights be integrated into the routing strategy to support long-term growth while maintaining logistical efficiency?
- What levers can be employed to accelerate volume and share growth at growth-ready, high-potential customers?

Import libraries

```
# import libraries
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.4     ✔ readr     2.1.5
## ✔ forcats   1.0.0     ✔ stringr   1.5.1
## ✔ ggplot2   3.5.1     ✔ tibble    3.2.1
## ✔ lubridate 1.9.4     ✔ tidyr     1.3.1
## ✔ purrr     1.0.2
## ── 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 become errors
```

```
library(janitor)
```

```
##
## Attaching package: 'janitor'
##
## The following objects are masked from 'package:stats':
##
##     chisq.test, fisher.test
```

```
library(skimr)
library(psych)
```

```
##
## Attaching package: 'psych'
##
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```
library(glue)
library(here)
```

```
## here() starts at C:/Users/varun/Box Sync/Business Analytics Degree/Semesters/Spring Semester 2025/IS 6813/EDA/Solo EDA/Sucessful_
State_Code
```

```
library(readxl)
library(zipcodeR)
```

Import Datasets

- There are 4 datasets used for the analysis, which contains address, customer profile, delivery cost, and transaction history.

```r
# Create a variable that contains all of the data files
directory = "C:\\Users\\varun\\Box Sync\\Business Analytics Degree\\Semesters\\Spring Semester 2025\\IS 6813\\Data"

# Get the directory for all the files
files <- list.files(directory,full.names = TRUE)

# Create Empty List to store all the files
data <- list()

# Loop through each file
for (i in files) {
  # Check if the file is a CSV
  if (grepl("\\.csv$", i)) {
    # Read the CSV file
    a <- read_csv(i)
    # Process the data by only extracting the name of the file and not the full file path
    file_name <- basename(i)
    data[[file_name]] <- a
  }
  # Check if the file is an excel file
  else if (grepl("\\.xlsx$",i)) {
    b <- read_excel(i)
    # Process the data by only extracting the name of the file and not the full file path
    file_name_01 <- basename(i)
    data[[file_name_01]] <- b
  }
  # If the file is neither csv or excel, exit the loop
  else {
    # Ignore the file if it's not a CSV
    next
  }
}

# Extract the dataframes from the "data" list
address_df <- data[["customer_address_and_zip_mapping.csv"]]
profile_df <- data[["customer_profile.csv"]]
trans_df <- data[["transactional_data.csv"]]
delivery_cost_df <- data[["delivery_cost_data.xlsx"]]

# Remove intermediate variables used when reading in the functions
rm(a)
rm(b)
rm(directory)
rm(file_name)
rm(file_name_01)
rm(files)
rm(i)
rm(data)
```

Dataset Profiling & Exploration

1. Address Dataset Profile

Variables can be described as below.

- Zip: ZIP code for the location.
- Full address: Full address information seperated by , including city, state, county, region, and latitude/longitude.
- Full address is listed in the order of zipcode, city, state full name, state acronym, county, FIPS codes, latitude, longitude

```r
sample_n(address_df, 10)
```

zip	full address
<dbl>	<chr>
42603	42603,Alpha,Kentucky,KY,Clinton,53,36.7824,-85.0275
67543	67543,Haven,Kansas,KS,Reno,155,37.8989,-97.7828
42413	42413,Hanson,Kentucky,KY,Hopkins,107,37.4382,-87.4751
1831	01831,Haverhill,Massachusetts,MA,Essex,9,42.7711,-71.1221
1253	01253,Otis,Massachusetts,MA,Berkshire,3,42.1931,-73.0918

zip	full address
<dbl>	<chr>
1742	01742,Concord,Massachusetts,MA,Middlesex,17,42.4567,-71.3747
66667	66667,Topeka,Kansas,KS,Shawnee,177,39.0429,-95.7697
21776	21776,New Windsor,Maryland,MD,Carroll,13,39.5162,-77.1034
67634	67634,Dorrance,Kansas,KS,Russell,167,38.8348,-98.5695
1431	01431,Ashby,Massachusetts,MA,Middlesex,17,42.6745,-71.8174

1-10 of 10 rows

2. Customer Profile Dataset Profile

Variables can be described as below.

- Customer Number: Unique identifying number of customer
- Primary Group Number: The group number of which customer mainly belongs to
- Frequent Order Type: The order type that customer mainly uses
- First Delivery Date: The date that first delivery was made
- On Boarding Date: The date that first transaction was made
- Cold Drink Channel: General channel category for cold drink purchases (e.g., “DINING”)
- Trade Channel: Detailed channel classification (e.g., “OTHER DINING & BEVERAGE”)
- Sub Trade Channel: Sub-classification within the trade channel (e.g., “OTHER DINING”)
- Local Market Partner: Whether customer is local market partner (True or False)
- CO2 Customer: Whether customer purchases CO2 product or not (True or False)
- Zip Code: customer address zip code which is connected with Zip variable in `address_df`

```
sample_n(profile_df,10)
```

CUSTOMER_NUMBER	PRIMARY_GROUP_NUMBER	FREQUENT_ORDER_TYPE	FIRST_DELIVERY_DATE
<dbl>	<dbl>	<chr>	<chr>
600574266	1194	SALES REP	5/22/2017
501243238	NA	SALES REP	11/11/2021
600574528	1274	SALES REP	3/2/2017
501475072	8521	SALES REP	1/5/2023
501647301	NA	MYCOKE360	5/10/2024
501026298	1194	SALES REP	12/6/2019
501502756	1971	SALES REP	6/1/2023
501602448	NA	MYCOKE360	2/2/2024
501444050	NA	CALL CENTER	4/11/2023
600057127	366	EDI	3/7/2018

1-10 of 10 rows | 1-4 of 11 columns

3. Delivery Cost Dataset Profile

Variables can be described as below.

- Cold Drink Channel: The main functional category of commerce
- Vol Range: The annual volume range of products
- Applicable to: which category of products that volumes apply to
- Median Delivery Cost: Median cost of delivery per cost type
- Cost type: the unit by measuring the cost
- Fountain → Measured in gallons (Per Gallon)
- Bottles and Cans → Measured in cases (Per Case).

```
delivery_cost_df
```

Cold Drink Channel	Vol Range	Applicable To	Median Delivery Cost	Cost Type
<chr>	<chr>	<chr>	<dbl>	<chr>
WORKPLACE	0 - 149	Bottles and Cans	8.0649504	Per Case
WORKPLACE	150 - 299	Bottles and Cans	4.1656458	Per Case

Cold Drink Channel	Vol Range	Applicable To	Median Delivery Cost	Cost Type
<chr>	<chr>	<chr>	<dbl>	<chr>
WORKPLACE	300 - 449	Bottles and Cans	2.9915579	Per Case
WORKPLACE	450 - 599	Bottles and Cans	2.5242219	Per Case
WORKPLACE	600 - 749	Bottles and Cans	2.0568859	Per Case
WORKPLACE	750 - 899	Bottles and Cans	1.9995638	Per Case
WORKPLACE	900 - 1049	Bottles and Cans	1.9422418	Per Case
WORKPLACE	1050 - 1199	Bottles and Cans	1.8849198	Per Case
WORKPLACE	1200 - 1349	Bottles and Cans	0.6666636	Per Case
WORKPLACE	1350+	Bottles and Cans	0.3716757	Per Case

1-10 of 160 rows Previous **1** 2 3 4 5 6 ... 16 Next

4. Transaction Dataset Profile

Variables can be described as below.

- Transaction Date: Date of the transaction (YYYY-MM-DD format).
- Week: Week number of the year when the transaction occurred.
- Year: Year of the transaction occurred.
- Customer Number: Unique identifier for the customer.
- Order Type: Type of order placed
- Ordered Cases: The amount of cases that ordered
- Loaded Cases: The amount of cases that loaded in the truck
- Delivered Cases: The amount of cases that delivered to the customer
- Ordered Gallons: The amount of gallons that ordered
- Loaded Gallons: The amount of gallons that loaded in the truck
- Delivered Gallons: The amount of gallons that delivered to the customer
- **Information 1**: One standard physical case equating to one gallon, allowing for a direct summation of cases and gallons.
- **Information 2**: Negative delivered volume must be considered as a return.

```
sample_n(trans_df,10)
```

TRANSACTION_DATE	WEEK	YEAR	CUSTOMER_NUMBER	ORDER_TYPE	ORDERED_CASES
<chr>	<dbl>	<dbl>	<dbl>	<chr>	<dbl>
8/16/2023	33	2023	600263480	CALL CENTER	805.0
10/20/2023	42	2023	501346848	MYCOKE LEGACY	0.0
4/6/2023	14	2023	600053933	CALL CENTER	4.5
11/14/2024	46	2024	600566637	MYCOKE360	15.5
10/3/2024	40	2024	501027831	CALL CENTER	30.0
10/8/2024	41	2024	600685959	MYCOKE360	14.0
11/20/2024	47	2024	501583366	SALES REP	54.0
8/12/2024	33	2024	600249340	MYCOKE360	17.0
6/21/2024	25	2024	600055269	CALL CENTER	5.0
3/22/2024	12	2024	501214327	EDI	9.0

1-10 of 10 rows | 1-6 of 11 columns

Skimming of Dataset

```
skim(address_df)
```

Data summary

Name	address_df
Number of rows	1801
Number of columns	2

Column type frequency:	
character	1
numeric	1
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
full address	0	1	45	73	0	1801	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
zip	0	1	28919.81	25588.64	1001	2153	21634	42440	71483	

```
skim(profile_df)
```

Data summary

Name	profile_df
Number of rows	30478
Number of columns	11
Column type frequency:	
character	6
logical	2
numeric	3
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
FREQUENT_ORDER_TYPE	0	1	3	13	0	6	0
FIRST_DELIVERY_DATE	0	1	8	10	0	2401	0
ON_BOARDING_DATE	0	1	8	10	0	6487	0
COLD_DRINK_CHANNEL	0	1	5	13	0	9	0
TRADE_CHANNEL	0	1	6	28	0	26	0
SUB_TRADE_CHANNEL	0	1	4	27	0	48	0

Variable type: logical

skim_variable	n_missing	complete_rate	mean	count
LOCAL_MARKET_PARTNER	0	1	0.90	TRU: 27355, FAL: 3123
CO2_CUSTOMER	0	1	0.39	FAL: 18496, TRU: 11982

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
CUSTOMER_NUMBER	0	1.0	538301800.92	47950644.47	500245678	501164306	501573995	600075795	600975408	
PRIMARY_GROUP_NUMBER	18196	0.4	2779.85	2608.64	4	444	1892	4488	9999	
ZIP_CODE	0	1.0	30252.25	25953.08	1001	2155	21771	42762	71483	

```
skim(delivery_cost_df)
```

Data summary

Name	delivery_cost_df
Number of rows	160
Number of columns	5
_______________________	
Column type frequency:	
character	4
numeric	1
________________________	
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
Cold Drink Channel	0	1	5	13	0	8	0
Vol Range	0	1	5	11	0	10	0
Applicable To	0	1	8	16	0	2	0
Cost Type	0	1	8	10	0	2	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
Median Delivery Cost	0	1	2.6	1.71	0.37	1.33	2.24	3.47	8.59	▇▅▂▁▁

```
skim(trans_df)
```

Data summary

Name	trans_df
Number of rows	1045540
Number of columns	11
_______________________	
Column type frequency:	
character	2
numeric	9
________________________	
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
TRANSACTION_DATE	0	1	8	10	0	723	0
ORDER_TYPE	0	1	3	13	0	7	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
WEEK	0	1	26.23	14.52	1.0	14	26	38.00	52.00	▇▇▇
YEAR	0	1	2023.50	0.50	2023.0	2023	2023	2024.00	2024.00	▇▁
CUSTOMER_NUMBER	0	1	546643776.32	49426585.56	500245678.0	501091920	501548213	600080939.00	600975408.00	▇▁
ORDERED_CASES	0	1	26.85	126.76	0.0	0	7	18.50	8479.89	▇▁

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
LOADED_CASES	0	1	25.92	122.79	0.0	0	7	18.00	8171.56	▇▁▁▁▁
DELIVERED_CASES	0	1	25.13	121.52	-3132.0	0	6	17.33	8069.48	▁▇▁▁▁
ORDERED_GALLONS	0	1	9.87	26.47	0.0	0	0	12.50	2562.50	▇▁▁▁▁
LOADED_GALLONS	0	1	9.60	25.65	0.0	0	0	12.50	2562.50	▇▁▁▁▁
DELIVERED_GALLONS	0	1	9.21	25.18	-1792.5	0	0	12.50	2292.50	▁▁▁▇▁

Checking NA per variable

```
colSums(is.na(address_df))
```

```
##         zip full address
##           0            0
```

```
colSums(is.na(profile_df))
```

```
##      CUSTOMER_NUMBER PRIMARY_GROUP_NUMBER  FREQUENT_ORDER_TYPE
##                    0                18196                    0
##  FIRST_DELIVERY_DATE     ON_BOARDING_DATE   COLD_DRINK_CHANNEL
##                    0                    0                    0
##        TRADE_CHANNEL    SUB_TRADE_CHANNEL LOCAL_MARKET_PARTNER
##                    0                    0                    0
##         CO2_CUSTOMER             ZIP_CODE
##                    0                    0
```

```
colSums(is.na(delivery_cost_df))
```

```
##   Cold Drink Channel            Vol Range        Applicable To
##                    0                    0                    0
## Median Delivery Cost            Cost Type
##                    0                    0
```

```
colSums(is.na(trans_df))
```

```
##  TRANSACTION_DATE              WEEK              YEAR   CUSTOMER_NUMBER
##                 0                 0                 0                 0
##        ORDER_TYPE     ORDERED_CASES      LOADED_CASES   DELIVERED_CASES
##                 0                 0                 0                 0
##   ORDERED_GALLONS    LOADED_GALLONS DELIVERED_GALLONS
##                 0                 0                 0
```

```
colSums(is.na(address_df)) / nrow(address_df) * 100
```

```
##         zip full address
##           0            0
```

```
colSums(is.na(profile_df)) / nrow(profile_df) * 100
```

```
##      CUSTOMER_NUMBER PRIMARY_GROUP_NUMBER  FREQUENT_ORDER_TYPE
##              0.00000             59.70208              0.00000
##  FIRST_DELIVERY_DATE     ON_BOARDING_DATE   COLD_DRINK_CHANNEL
##              0.00000              0.00000              0.00000
##        TRADE_CHANNEL    SUB_TRADE_CHANNEL LOCAL_MARKET_PARTNER
##              0.00000              0.00000              0.00000
##         CO2_CUSTOMER             ZIP_CODE
##              0.00000              0.00000
```

```
colSums(is.na(delivery_cost_df)) / nrow(delivery_cost_df) * 100
```

```
##   Cold Drink Channel            Vol Range        Applicable To
##                    0                    0                    0
## Median Delivery Cost            Cost Type
##                    0                    0
```

```
colSums(is.na(trans_df)) / nrow(trans_df) * 100
```

```
##  TRANSACTION_DATE              WEEK              YEAR   CUSTOMER_NUMBER
##                 0                 0                 0                 0
##        ORDER_TYPE     ORDERED_CASES      LOADED_CASES   DELIVERED_CASES
##                 0                 0                 0                 0
##   ORDERED_GALLONS    LOADED_GALLONS DELIVERED_GALLONS
##                 0                 0                 0
```

- `PRIMARY_GROUP_NUMBER` has a 18196 missing values, which takes up 60% of `profile_df` dataset.

The list of EDA questions

- How many customers are partnered with Local Market Partners out of the entire customers?
- How many customers are purchasing C02 products out of entire customers?
- Which number can we extract out of transaction history?
- How many customers belongs to the direct route based on the original volume threshold? And how many customers belong to the ARTM based on the original volume threshold?
- Which customer characteristics have brought more profits from given transaction data?
  - CO2 vs Non-CO2
  - Local Market Partners vs Non-Local Market Partners
  - Cold Drink Channel
  - Frequent Order Type
- How many customers belongs to the Local Market Partners that buy fountains only? (Group Segment 1)
-

The summary table of Local Market Partner Customer

```
# the distribution of local market partner customers out of entire customers
table(profile_df$LOCAL_MARKET_PARTNER)
```

```
## 
## FALSE  TRUE 
##  3123 27355
```

```
round(prop.table(table(profile_df$LOCAL_MARKET_PARTNER)),2)
```

```
## 
## FALSE  TRUE 
##   0.1   0.9
```

Approximately, 90% of listed customers belong to the local market partners, which indicates that they are smaller, regionally focused customers who serve their local communities. They tend to show their reliance on local market dynamics and consistent purchasing patterns.

The summary table of of CO2 customer

```
# the distribution of CO2 customers out of entire customers
table(profile_df$CO2_CUSTOMER)
```

```
## 
## FALSE  TRUE 
## 18496 11982
```

```
round(prop.table(table(profile_df$CO2_CUSTOMER)),2)
```

```
## 
## FALSE  TRUE 
##  0.61  0.39
```

Approximately, 40% of listed customer belongs to the CO2 customer, which represents that they have purchased carbon dioxide materials.

Total number of transaction

- Total number of customer
- Total volume of cases
- Total volume of gallons
- Total transaction period

```
trans_df %>%
  summarise(customer_n = n_distinct(CUSTOMER_NUMBER))
```

customer_n
<int>
30322

1 row

```
trans_df %>%
  summarise(case_volume = sum(ORDERED_CASES),
            gallon_volume = sum(ORDERED_GALLONS),
            total_volume = case_volume + gallon_volume)
```

case_volume	gallon_volume	total_volume
<dbl>	<dbl>	<dbl>
28074470	10323337	38397807

1 row

```
max(as.Date(trans_df$TRANSACTION_DATE, format="%m/%d/%Y"))
```

```
## [1] "2024-12-31"
```

```
min(as.Date(trans_df$TRANSACTION_DATE, format="%m/%d/%Y"))
```

```
## [1] "2023-01-01"
```

30322 customers have transacted 28,074,470 cases and 10,323,337 gallons (total 38,397,807 units) with SCCU from 1/1/2023 to 12/31/2024. (2 years)

```
trans_history <-
trans_df %>%
  mutate(TRANSACTION_DATE = as.Date(TRANSACTION_DATE, format="%m/%d/%Y")) %>%
  group_by(CUSTOMER_NUMBER) %>%
  summarise(
            FIRST_TRANSACTION_DATE = min(TRANSACTION_DATE),
            LAST_TRANSACTION_DATE = max(TRANSACTION_DATE),
            TRANS_DAYS = LAST_TRANSACTION_DATE - FIRST_TRANSACTION_DATE + 1,
            TRANS_COUNT = n(),
            TRANS_COUNT_2023 = sum((year(TRANSACTION_DATE) == 2023)),
            TRANS_COUNT_2024 = sum((year(TRANSACTION_DATE) == 2024)),
            ANNUAL_VOLUME_CASES_2023 = sum((year(TRANSACTION_DATE) == 2023) * ORDERED_CASES, na.rm = TRUE),
            ANNUAL_VOLUME_GALLON_2023 = sum((year(TRANSACTION_DATE) == 2023) * ORDERED_GALLONS, na.rm = TRUE),
            ANNUAL_VOLUME_CASES_2024 = sum((year(TRANSACTION_DATE) == 2024) * ORDERED_CASES, na.rm = TRUE),
            ANNUAL_VOLUME_GALLON_2024 = sum((year(TRANSACTION_DATE) == 2024) * ORDERED_GALLONS, na.rm = TRUE),
            ANNUAL_VOLUME_2023 = sum((year(TRANSACTION_DATE) == 2023) * (ORDERED_CASES + ORDERED_GALLONS), na.rm = TRUE),
            AVG_ORDER_VOLUME_2023 = ANNUAL_VOLUME_2023 / TRANS_COUNT_2023,
            ANNUAL_VOLUME_2024 = sum((year(TRANSACTION_DATE) == 2024) * (ORDERED_CASES + ORDERED_GALLONS), na.rm = TRUE),
            AVG_ORDER_VOLUME_2024 = ANNUAL_VOLUME_2024 / TRANS_COUNT_2024,
            CHANGED_VOLUME = ANNUAL_VOLUME_2024 - ANNUAL_VOLUME_2023,
            PERCENT_CHANGE = round(CHANGED_VOLUME/ANNUAL_VOLUME_2023,2) * 100,
            THRESHOLD_2023 = ifelse(ANNUAL_VOLUME_2023 >= 400, 'above', 'below'),
            THRESHOLD_2024 = ifelse(ANNUAL_VOLUME_2024 >= 400, 'above', 'below'),
  ) %>%
  ungroup()

trans_history
```

CUSTOMER_NUMBER <dbl>	FIRST_TRANSACTION_DATE <date>	LAST_TRANSACTION_DATE <date>	TRANS_DAYS <drtn>
500245678	2023-01-09	2024-11-20	682 days
500245685	2023-01-06	2024-08-16	589 days
500245686	2023-03-07	2024-12-17	652 days
500245687	2023-02-06	2024-10-28	631 days
500245689	2023-01-13	2024-12-26	714 days
500245690	2023-01-26	2024-12-23	698 days
500245695	2023-01-04	2024-12-04	701 days
500245698	2023-01-13	2024-12-23	711 days
500245701	2023-01-03	2024-05-13	497 days
500245704	2023-01-10	2024-12-26	717 days

1-10 of 10,000 rows | 1-4 of 19 columns Previous **1** 2 3 4 5 6 ... 1000 Next

```
colSums(is.na(trans_history))
```

```
##          CUSTOMER_NUMBER    FIRST_TRANSACTION_DATE     LAST_TRANSACTION_DATE
##                        0                         0                         0
##               TRANS_DAYS               TRANS_COUNT          TRANS_COUNT_2023
##                        0                         0                         0
##          TRANS_COUNT_2024  ANNUAL_VOLUME_CASES_2023 ANNUAL_VOLUME_GALLON_2023
##                        0                         0                         0
##  ANNUAL_VOLUME_CASES_2024 ANNUAL_VOLUME_GALLON_2024        ANNUAL_VOLUME_2023
##                        0                         0                         0
##     AVG_ORDER_VOLUME_2023        ANNUAL_VOLUME_2024     AVG_ORDER_VOLUME_2024
##                     4270                         0                       721
##            CHANGED_VOLUME            PERCENT_CHANGE            THRESHOLD_2023
##                        0                       137                         0
##            THRESHOLD_2024
##                        0
```

- calculation of ANNUAL_VOLUME = AVG_ORDER_VOLUME (Order Volume) * TRANS_COUNT (Frequency) for certain year (2023 vs 2024)

```
# 2023 above vs below threshold
table(trans_history$THRESHOLD_2023)
```

```
##
## above below
##  7745 22577
```

```
prop.table(table(trans_history$THRESHOLD_2023))
```

```
##
##     above     below
## 0.2554251 0.7445749
```

```
# 2024 above vs below threshold
table(trans_history$THRESHOLD_2024)
```

```
##
## above below
##  7867 22455
```

```
prop.table(table(trans_history$THRESHOLD_2024))
```

```
##
##     above     below
## 0.2594486 0.7405514
```

- approximately, 25% of customers are above the original volume threshold (400 annual volume), whereas 75% remains below the threshold in both 2023 and 2024. It appears that the proportion of customer group haven't changed much.

```
thres_change_customer <-
trans_history %>%
  filter(THRESHOLD_2023 != THRESHOLD_2024)

thres_change_customer
```

CUSTOMER_NUMBER <dbl>	FIRST_TRANSACTION_DATE <date>	LAST_TRANSACTION_DATE <date>	TRANS_DAYS <drtn>
500245698	2023-01-13	2024-12-23	711 days
500245791	2023-01-10	2024-12-24	715 days
500245851	2023-10-11	2023-10-17	7 days
500245864	2023-02-23	2024-08-23	548 days
500246054	2023-01-13	2023-12-29	351 days
500249461	2023-01-10	2024-12-17	708 days
500263851	2023-03-03	2024-12-20	659 days
500264574	2023-01-06	2024-12-27	722 days
500264805	2023-01-12	2024-12-19	708 days
500266407	2023-01-11	2024-12-18	708 days

1-10 of 2,378 rows | 1-4 of 19 columns Previous **1** 2 3 4 5 6 ... 238 Next

```
table(thres_change_customer$THRESHOLD_2023, thres_change_customer$THRESHOLD_2024)
```

```
##
##         above below
##   above     0  1128
##   below  1250     0
```

```
round(prop.table(table(thres_change_customer$THRESHOLD_2023, thres_change_customer$THRESHOLD_2024)),2)
```

```
##
##         above below
##   above  0.00  0.47
##   below  0.53  0.00
```

However, when we get into the depth of data, 2,378 (8%) customers experienced a change in volume based on the original volume threshold from 2023 to 2024 out of 30,322 customers. Among them, 1,250 customers (around 4%) exceeded the threshold in 2024 from below threshold status, whereas 1,128 (around 4%) customers drops below the threshold.

Volume changes comparison

Changed volume statistics

```
# total customer growth statistics
trans_history %>%
  summarise(AVG_CHANGE_VOL = mean(CHANGED_VOLUME),
            MED_CHANGE_VOL = median(CHANGED_VOLUME),
            MIN_CHANGE_VOL = min(CHANGED_VOLUME),
            MAX_CHANGE_VOL = max(CHANGED_VOLUME))
```

AVG_CHANGE_VOL <dbl>	MED_CHANGE_VOL <dbl>	MIN_CHANGE_VOL <dbl>	MAX_CHANGE_VOL <dbl>
32.51572	0	-132830	86977

1 row

```r
# below in both year growth statistics

trans_history %>%
  filter(THRESHOLD_2023 == 'below' & THRESHOLD_2024 == 'below') %>%
  summarise(AVG_CHANGE_VOL = mean(CHANGED_VOLUME),
            MED_CHANGE_VOL = median(CHANGED_VOLUME),
            MIN_CHANGE_VOL = min(CHANGED_VOLUME),
            MAX_CHANGE_VOL = max(CHANGED_VOLUME))
```

AVG_CHANGE_VOL	MED_CHANGE_VOL	MIN_CHANGE_VOL	MAX_CHANGE_VOL
<dbl>	<dbl>	<dbl>	<dbl>
6.849459	1.5	-393	399.009

1 row

```r
# above in both year growth statistics

trans_history %>%
  filter(THRESHOLD_2023 == 'above' & THRESHOLD_2024 == 'above') %>%
  summarise(AVG_CHANGE_VOL = mean(CHANGED_VOLUME),
            MED_CHANGE_VOL = median(CHANGED_VOLUME),
            MIN_CHANGE_VOL = min(CHANGED_VOLUME),
            MAX_CHANGE_VOL = max(CHANGED_VOLUME))
```

AVG_CHANGE_VOL	MED_CHANGE_VOL	MIN_CHANGE_VOL	MAX_CHANGE_VOL
<dbl>	<dbl>	<dbl>	<dbl>
5.785284	-17	-132830	82637.21

1 row

```r
# potential growth customer statistics
trans_history %>%
  filter(THRESHOLD_2023 == 'below' & THRESHOLD_2024 == 'above') %>%
  summarise(AVG_CHANGE_VOL = mean(CHANGED_VOLUME),
            MED_CHANGE_VOL = median(CHANGED_VOLUME),
            MIN_CHANGE_VOL = min(CHANGED_VOLUME),
            MAX_CHANGE_VOL = max(CHANGED_VOLUME))
```

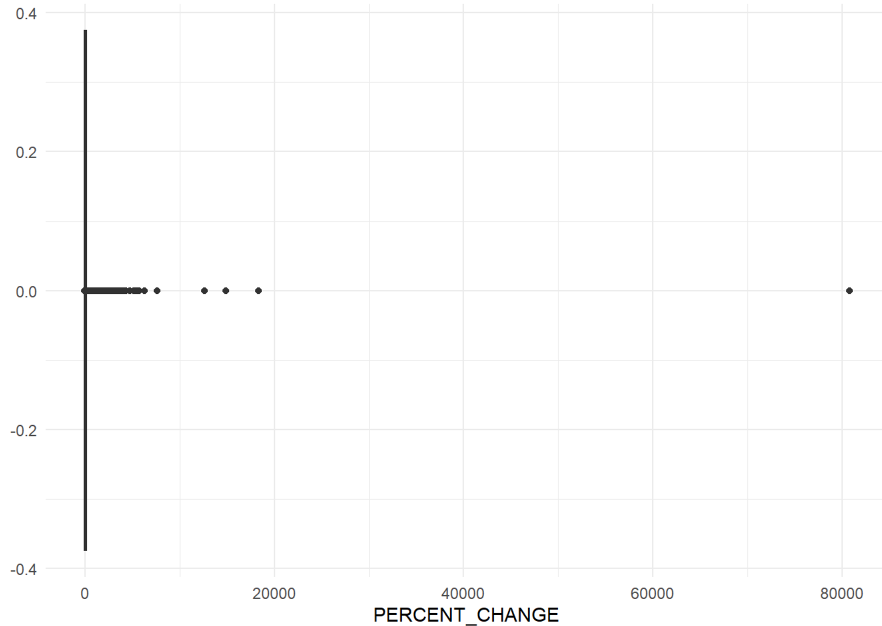
AVG_CHANGE_VOL	MED_CHANGE_VOL	MIN_CHANGE_VOL	MAX_CHANGE_VOL
<dbl>	<dbl>	<dbl>	<dbl>
1035.36	418	8.5	86977

1 row

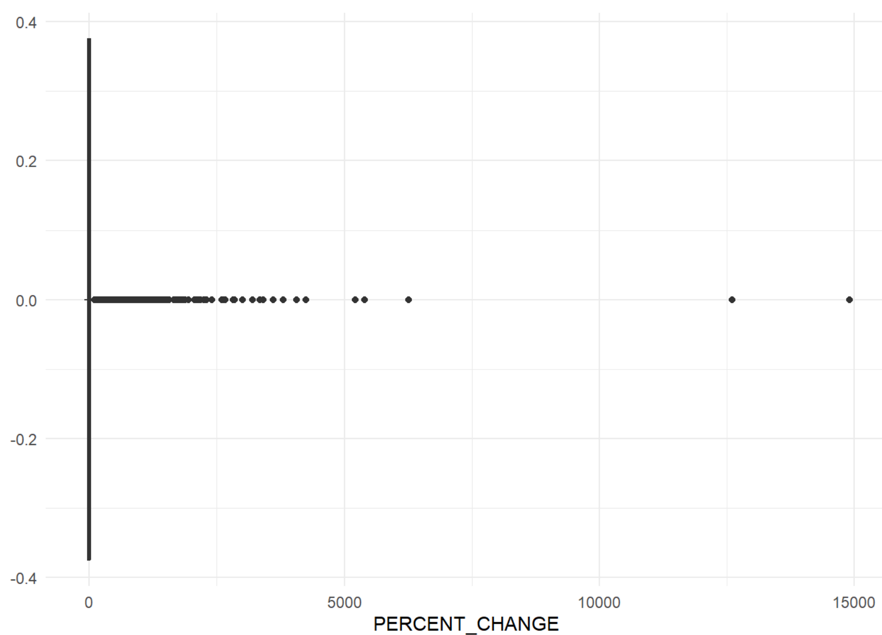
Changes in volume percent distribution

```r
# total customer
trans_history %>%
  ggplot() +
  geom_boxplot(aes(x = PERCENT_CHANGE)) +
  theme_minimal()
```

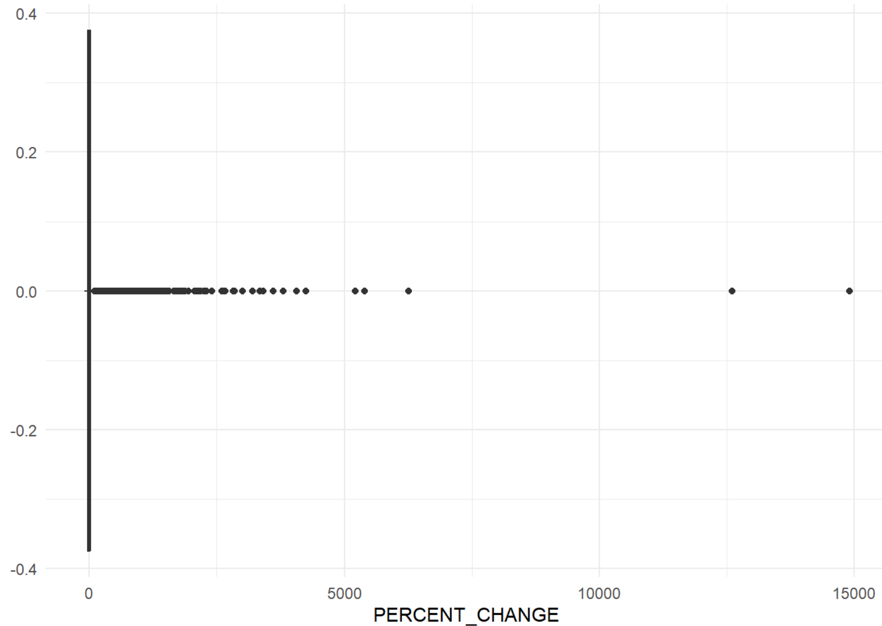
```
## Warning: Removed 4413 rows containing non-finite outside the scale range
## (`stat_boxplot()`).
```




```
# both below customer
trans_history %>%
  filter(THRESHOLD_2023 == 'below' & THRESHOLD_2024 == 'below') %>%
  ggplot() +
  geom_boxplot(aes(x = PERCENT_CHANGE), na.rm = TRUE) +
  theme_minimal()
```

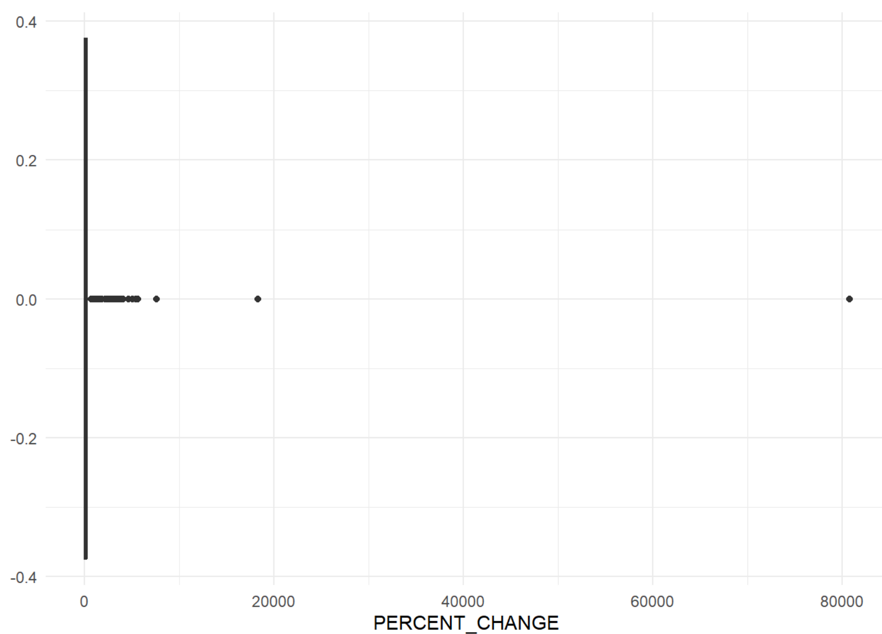



```
# both above customer
trans_history %>%
  filter(THRESHOLD_2023 == 'below' & THRESHOLD_2024 == 'below') %>%
  ggplot() +
  geom_boxplot(aes(x = PERCENT_CHANGE), na.rm = TRUE) +
  theme_minimal()
```




```
# potential growth customer
trans_history %>%
  filter(THRESHOLD_2023 == 'below' & THRESHOLD_2024 == 'above') %>%
  ggplot() +
  geom_boxplot(aes(x = PERCENT_CHANGE)) +
  theme_minimal()
```

```
## Warning: Removed 397 rows containing non-finite outside the scale range
## (`stat_boxplot()`).
```



Combining the Dataset (Data Modeling)

In order to take in-depth analysis per each of customer's attributes, we've combined the customer profile `profile_df` data with `trans_history` , joined by `CUSTOMER_NUMBER` variable.

```
trans_profile_df <- left_join(trans_history, profile_df, by = 'CUSTOMER_NUMBER')
sample_n(trans_profile_df,10)
```

CUSTOMER_NUMBER <dbl>	FIRST_TRANSACTION_DATE <date>	LAST_TRANSACTION_DATE <date>	TRANS_DAYS <drtn>
501308175	2023-01-13	2024-12-23	711 days

CUSTOMER_NUMBER <dbl>	FIRST_TRANSACTION_DATE <date>	LAST_TRANSACTION_DATE <date>	TRANS_DAYS <drtn>
501640760	2024-04-18	2024-04-18	1 days
500996453	2023-01-09	2024-12-27	719 days
500945262	2023-01-11	2024-12-04	694 days
500592664	2023-03-28	2024-12-17	631 days
501020632	2023-02-08	2024-12-18	680 days
600581294	2023-01-13	2024-12-20	708 days
501590513	2023-12-28	2024-11-14	323 days
501279610	2023-01-04	2024-12-26	723 days
500873944	2023-08-10	2024-07-15	341 days

1-10 of 10 rows | 1-4 of 29 columns

Local Market Partner Comparison

```
volume_2023 <- sum(trans_profile_df$ANNUAL_VOLUME_2023, na.rm = TRUE)
volume_2024 <- sum(trans_profile_df$ANNUAL_VOLUME_2024, na.rm = TRUE)

trans_profile_df %>%
  group_by(LOCAL_MARKET_PARTNER) %>%
  summarise(TOTAL_VOL_2023 = sum(ANNUAL_VOLUME_2023),
            TOTAL_VOL_2024 = sum(ANNUAL_VOLUME_2024),
            PERCENT_2023 = (TOTAL_VOL_2023 / volume_2023) * 100,
            PERCENT_2024 = (TOTAL_VOL_2024 / volume_2024) * 100,
            AVG_VOL_2023 = mean(ANNUAL_VOLUME_2023),
            AVG_VOL_2024 = mean(ANNUAL_VOLUME_2024),
            MED_VOL_2023 = median(ANNUAL_VOLUME_2023),
            MED_VOL_2024 = median(ANNUAL_VOLUME_2024),
            COUNT_2023 = sum(TRANS_COUNT_2023),
            COUNT_2024 = sum(TRANS_COUNT_2024),
            ABOVE_THRES_2023 = sum(THRESHOLD_2023 == 'above'),
            ABOVE_THRES_2024 = sum(THRESHOLD_2024 == 'above')
  )
```

LOCAL_MARKET_PARTNER <lgl>	TOTAL_VOL_2023 <dbl>	TOTAL_VOL_2024 <dbl>	PERCENT_2023 <dbl>	PERCENT_2024 <dbl>
FALSE	5332519	5310790	28.5071	26.96945
TRUE	13373414	14381084	71.4929	73.03055

2 rows | 1-5 of 13 columns

C02 customer Comparison

```
trans_profile_df %>%
  group_by(CO2_CUSTOMER) %>%
  summarise(TOTAL_VOL_2023 = sum(ANNUAL_VOLUME_2023),
            TOTAL_VOL_2024 = sum(ANNUAL_VOLUME_2024),
            PERCENT_2023 = (TOTAL_VOL_2023 / volume_2023) * 100,
            PERCENT_2024 = (TOTAL_VOL_2024 / volume_2024) * 100,
            AVG_VOL_2023 = mean(ANNUAL_VOLUME_2023),
            AVG_VOL_2024 = mean(ANNUAL_VOLUME_2024),
            MED_VOL_2023 = median(ANNUAL_VOLUME_2023),
            MED_VOL_2024 = median(ANNUAL_VOLUME_2024),
            COUNT_2023 = sum(TRANS_COUNT_2023),
            COUNT_2024 = sum(TRANS_COUNT_2024),
            ABOVE_THRES_2023 = sum(THRESHOLD_2023 == 'above'),
            ABOVE_THRES_2024 = sum(THRESHOLD_2024 == 'above')
  )
```

CO2_CUSTOMER <lgl>	TOTAL_VOL_2023 <dbl>	TOTAL_VOL_2024 <dbl>	PERCENT_2023 <dbl>	PERCENT_2024 <dbl>
FALSE	12304118	12919326	65.77655	65.6074

CO2_CUSTOMER	TOTAL_VOL_2023	TOTAL_VOL_2024	PERCENT_2023	PERCENT_2024
<lgl>	<dbl>	<dbl>	<dbl>	<dbl>
TRUE	6401815	6772548	34.22345	34.3926

2 rows | 1-5 of 13 columns

Frequent order type Comparison

```
trans_profile_df %>%
  group_by(FREQUENT_ORDER_TYPE) %>%
  summarise(TOTAL_VOL_2023 = sum(ANNUAL_VOLUME_2023),
            TOTAL_VOL_2024 = sum(ANNUAL_VOLUME_2024),
            PERCENT_2023 = (TOTAL_VOL_2023 / volume_2023) * 100,
            PERCENT_2024 = (TOTAL_VOL_2024 / volume_2024) * 100,
            AVG_VOL_2023 = mean(ANNUAL_VOLUME_2023),
            AVG_VOL_2024 = mean(ANNUAL_VOLUME_2024),
            MED_VOL_2023 = median(ANNUAL_VOLUME_2023),
            MED_VOL_2024 = median(ANNUAL_VOLUME_2024),
            COUNT_2023 = sum(TRANS_COUNT_2023),
            COUNT_2024 = sum(TRANS_COUNT_2024),
            ABOVE_THRES_2023 = sum(THRESHOLD_2023 == 'above'),
            ABOVE_THRES_2024 = sum(THRESHOLD_2024 == 'above')
  )
```

FREQUENT_ORDER_TYPE	TOTAL_VOL_2023	TOTAL_VOL_2024	PERCENT_2023	PERCENT_2024
<chr>	<dbl>	<dbl>	<dbl>	<dbl>
CALL CENTER	179514.0	186631.8	0.9596635	0.9477604
EDI	149081.2	305437.8	0.7969731	1.5510854
MYCOKE LEGACY	246564.9	244420.9	1.3181106	1.2412271
MYCOKE360	381316.7	581339.1	2.0384802	2.9521774
OTHER	3753564.5	3612092.6	20.0661713	18.3430614
SALES REP	13995891.3	14761952.2	74.8206014	74.9646883

6 rows | 1-5 of 13 columns

Cold Drink Channel Comparison

```
trans_profile_df %>%
  group_by(COLD_DRINK_CHANNEL) %>%
  summarise(TOTAL_VOL_2023 = sum(ANNUAL_VOLUME_2023),
            TOTAL_VOL_2024 = sum(ANNUAL_VOLUME_2024),
            PERCENT_2023 = (TOTAL_VOL_2023 / volume_2023) * 100,
            PERCENT_2024 = (TOTAL_VOL_2024 / volume_2024) * 100,
            AVG_VOL_2023 = mean(ANNUAL_VOLUME_2023),
            AVG_VOL_2024 = mean(ANNUAL_VOLUME_2024),
            MED_VOL_2023 = median(ANNUAL_VOLUME_2023),
            MED_VOL_2024 = median(ANNUAL_VOLUME_2024),
            COUNT_2023 = sum(TRANS_COUNT_2023),
            COUNT_2024 = sum(TRANS_COUNT_2024),
            ABOVE_THRES_2023 = sum(THRESHOLD_2023 == 'above'),
            ABOVE_THRES_2024 = sum(THRESHOLD_2024 == 'above')
  )
```

COLD_DRINK_CHANNEL	TOTAL_VOL_2023	TOTAL_VOL_2024	PERCENT_2023	PERCENT_2024
<chr>	<dbl>	<dbl>	<dbl>	<dbl>
ACCOMMODATION	476384.4	483019.35	2.54670235	2.45288662
BULK TRADE	4877746.7	5109930.39	26.07593428	25.94943632
CONVENTIONAL	5569.5	6052.25	0.02977398	0.03073476
DINING	5178051.2	5262747.86	27.68133134	26.72547961
EVENT	2377010.9	2448306.34	12.70725685	12.43307921
GOODS	1705056.7	2194385.48	9.11505824	11.14360898
PUBLIC SECTOR	999364.4	1027559.74	5.34249950	5.21819164

COLD_DRINK_CHANNEL <chr>	TOTAL_VOL_2023 <dbl>	TOTAL_VOL_2024 <dbl>	PERCENT_2023 <dbl>	PERCENT_2024 <dbl>
WELLNESS	622871.2	609083.30	3.32980584	3.09306918
WORKPLACE	2463877.7	2550789.64	13.17163762	12.95351368

9 rows | 1-5 of 13 columns

Zip Code Insights:

Varun EDA:

- How many customers are there in each State?
- How many states does Swire Coca Cola cover?
- What are the transactions per State? (Transactions are all of the orders that companies place in a state.)
- What is overall volume per state?
- What is the average volume per state?
- What is the breakdown of **Local Market Partners** vs everyone else in each state?

Extracting States from the Zip Codes

The addresses are anonymized to protect the identities of the clients. Swire Coca Cola however has provided the actual zip codes, which means that we can extract the state information from the zip codes. The code block below will extract the state information from the zip codes.

```
# Rename the zip column in address_df to ZIP_CODE for left join
 address_df <- address_df %>%
   rename(ZIP_CODE = zip)

# Do a left join and join the trans_profile_df with the address_df.
trans_profile_address_df <- left_join(trans_profile_df,address_df,by = "ZIP_CODE")


# Check to make sure that there are no missing values
sum(is.na(trans_profile_address_df$`full address`))
```

```
## [1] 0
```

```
# Extract all the 4 number zip codes from the dataframe
four_digit_zipcodes <- trans_profile_address_df %>%
  filter(nchar(as.character(ZIP_CODE)) == 4)

# Get the count of MA
MA = sum(grepl("Massachusetts",four_digit_zipcodes$`full address`))

# Compare the count of MA to the four_digit_zipcdes df
nrow(four_digit_zipcodes) == MA
```

```
## [1] TRUE
```

```
# Add leading zero for 4-digit ZIP codes
trans_profile_address_df <- trans_profile_address_df %>%
  mutate(ZIP_CODE = if_else(nchar(as.character(ZIP_CODE)) == 4, paste0("0", as.character(ZIP_CODE)), as.character(ZIP_CODE)))

# Create a vector of Zip Codes
Zip_Codes <- trans_profile_address_df %>%
  select(ZIP_CODE) %>% pull()

# Create an Empty Vector which still store the state names
state_names <- vector()

# Use for loop to get state names for each zip code
for (i in 1:length(Zip_Codes)) {
   # Get the state for the current ZIP code
  a <- tryCatch(reverse_zipcode(as.character(Zip_Codes[i]))$state, error = function(e) NA)  # Handle errors by assigning NA

  # Store the state in the vector
  state_names[i] <- a
}

# Add the state vector to the dataframe
trans_profile_address_df$State <- state_names
```

The states have now been successfully extracted from the zip codes and added to `trans_profile_df`

How many Customers per State

```
# See how many unique States are in this profile
length(unique(trans_profile_address_df$State))
```

```
## [1] 5
```

```
# See how many unique Customers are there for each state
trans_profile_address_df %>%
  group_by(State) %>%
  summarise(n = n_distinct(CUSTOMER_NUMBER)) %>%
  arrange(desc(n))
```

State <chr>	n <int>
MA	10970
KS	7133
KY	6957
MD	4876
LA	386

5 rows

```
length(unique(trans_profile_df$CUSTOMER_NUMBER))
```

```
## [1] 30322
```

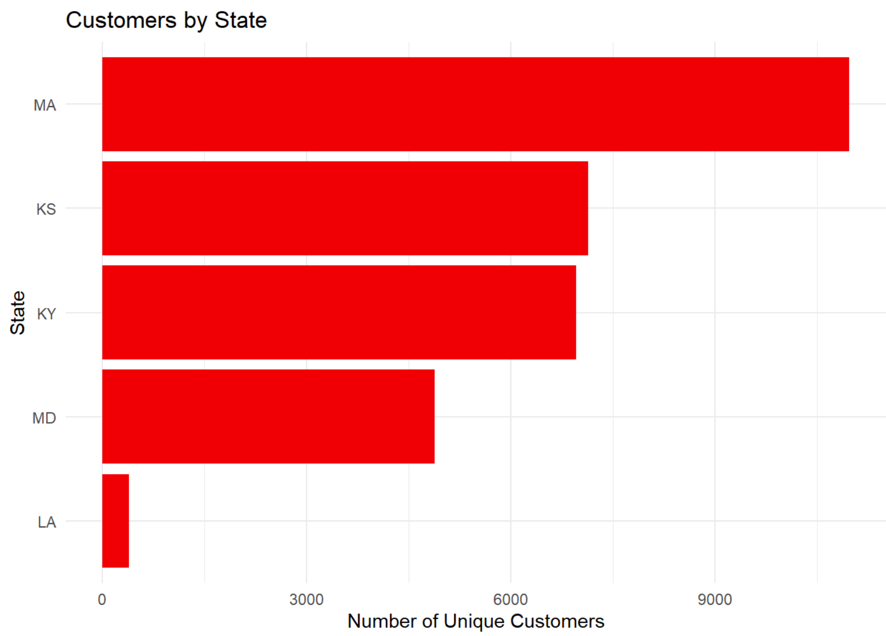
This dataset shows that SCCU serves 5 states which is

- Massachusetts
- Kansas
- Kentucky
- Maryland
- Louisiana

There are 30,322 customers overall and the customers per state adds up to this as well. A

Customers per State Graph

```
trans_profile_address_df %>%
  group_by(State) %>%
  summarise(n = n_distinct(CUSTOMER_NUMBER)) %>%
  arrange(desc(n)) %>%  # Ensures ordering before plotting
  ggplot(aes(x = fct_reorder(State, n), y = n, fill = State)) +  # Orders bars
  geom_col(show.legend = FALSE,fill = "#F40009") +  # Hides legend if not needed
  theme_minimal() +
  coord_flip() +  # Flips for better readability
  labs(x = "State", y = "Number of Unique Customers", title = "Customers by State")
```



Visualization shows the number of unique customers in each state. Massachusetts has the highest number of customers followed by Kansas and Kentucky. Kansas and Kentucky are very close in the number of customers that are served.

Finally Lousiana is last and the number of customers served in Lousiana is quite small compared to the other states.

Transaction by States

```
trans_profile_address_df %>%
  group_by(State) %>%
  summarise(transactions = sum(TRANS_COUNT),
            trans_2023 = sum(TRANS_COUNT_2023),
            trans_2024 = sum(TRANS_COUNT_2024),
            difference = trans_2024-trans_2023,
            pctg_change = round(((trans_2024-trans_2023)/trans_2023) * 100,2)) %>%
  arrange(desc(transactions))
```

State <chr>	transactions <int>	trans_2023 <int>	trans_2024 <int>	difference <int>	pctg_change <dbl>
MA	377139	189024	188115	-909	-0.48
KS	250843	126705	124138	-2567	-2.03
KY	239711	121270	118441	-2829	-2.33
MD	165257	83173	82084	-1089	-1.31
LA	12590	6222	6368	146	2.35

5 rows

This table shows the total transactions for each year per state. `trans_2023` is all the transactions that occurred in 2023 while `trans_2024` is all the transactions that occured in 2024.

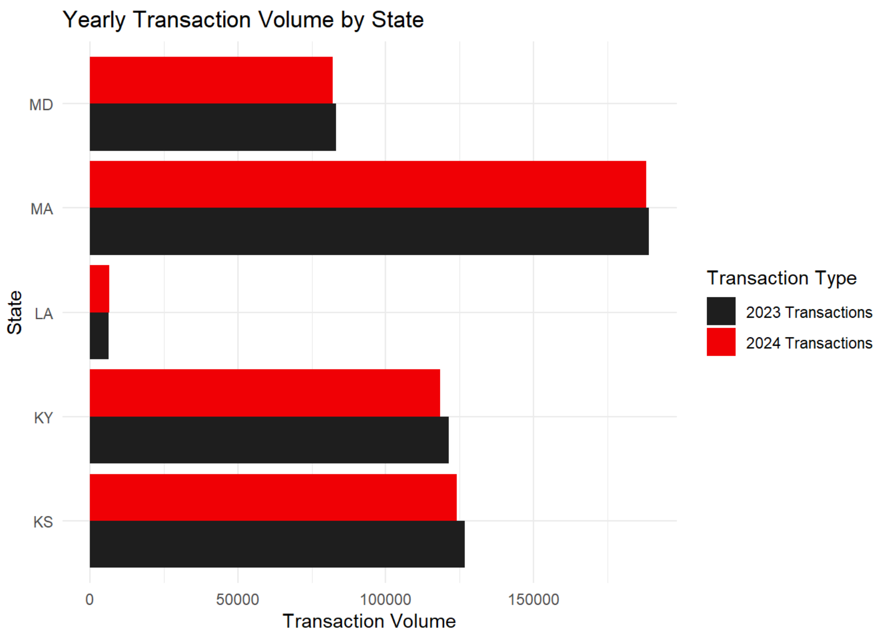
The `difference` column represents the change in transactions from 2023 to 2024 while `pctg_change` represents this difference as percentages.

Transactions by State Visualizations

Yearly Transactions by State

```r
# Assign changes from table to new dataframe
trans_profile_address_viz <- trans_profile_address_df %>%
  group_by(State) %>%
  summarise(transactions = sum(TRANS_COUNT),
            trans_2023 = sum(TRANS_COUNT_2023),
            trans_2024 = sum(TRANS_COUNT_2024),
            difference = trans_2024 - trans_2023,
            pctg_change = round(((trans_2024 - trans_2023) / trans_2023) * 100, 2)) %>%
  arrange(desc(transactions)) %>%
  pivot_longer(cols = c(trans_2023, trans_2024), # Pivot the dataframe for easier plotting
               names_to = "Metric", values_to = "Value")


ggplot(trans_profile_address_viz, aes(x = State, y = Value, fill = Metric)) +
  geom_col(position = "dodge") +  # Dodge to separate bars
  theme_minimal() +
  labs(title = "Yearly Transaction Volume by State",
       y = "Transaction Volume",
       x = "State",
       fill = "Transaction Type") +
  coord_flip() +
  scale_fill_manual(values = c("trans_2023" = "#1E1E1E", "trans_2024" = "#F40009"),
                    labels = c("2023 Transactions","2024 Transactions"))
```

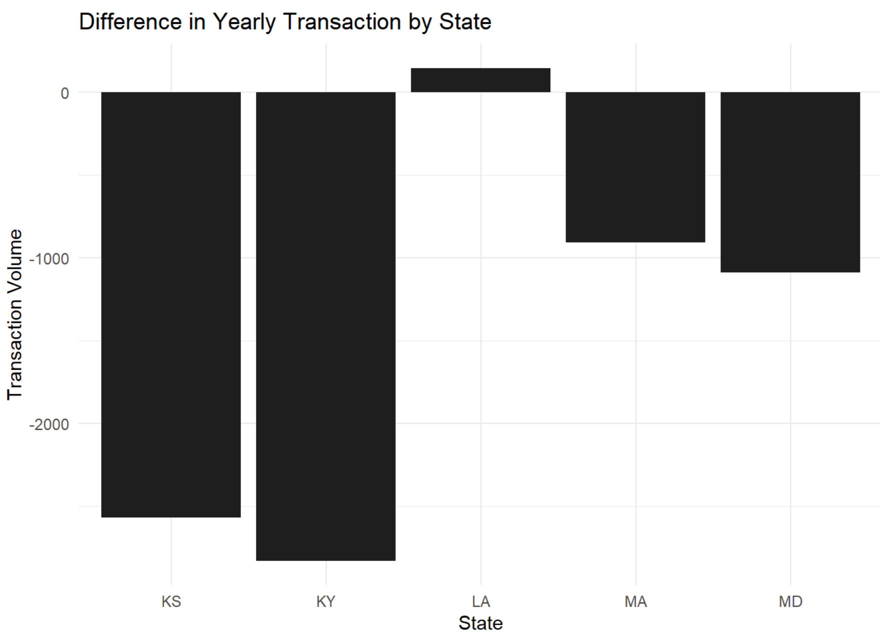


This graph shows the Transaction Volume by State for 2023 and 2024.

Transactions by State

```
# Assign changes from table to new dataframe
trans_profile_address_viz <- trans_profile_address_df %>%
  group_by(State) %>%
  summarise(transactions = sum(TRANS_COUNT),
            trans_2023 = sum(TRANS_COUNT_2023),
            trans_2024 = sum(TRANS_COUNT_2024),
            difference = trans_2024 - trans_2023,
            pctg_change = round(((trans_2024 - trans_2023) / trans_2023) * 100, 2)) %>%
  arrange(desc(transactions)) %>%
  pivot_longer(cols = c(difference), # Pivot the dataframe for easier plotting
               names_to = "Metric", values_to = "Value")


ggplot(trans_profile_address_viz, aes(x = State, y = Value, fill = Metric)) +
  geom_col(position = "dodge") +  # Dodge to separate bars
  theme_minimal() +
  labs(title = "Difference in Yearly Transaction by State",
       y = "Transaction Volume",
       x = "State",
       fill = "Transaction Type") +
  scale_fill_manual(values = c("difference" = "#1E1E1E")) +
  theme(legend.position = "none")
```

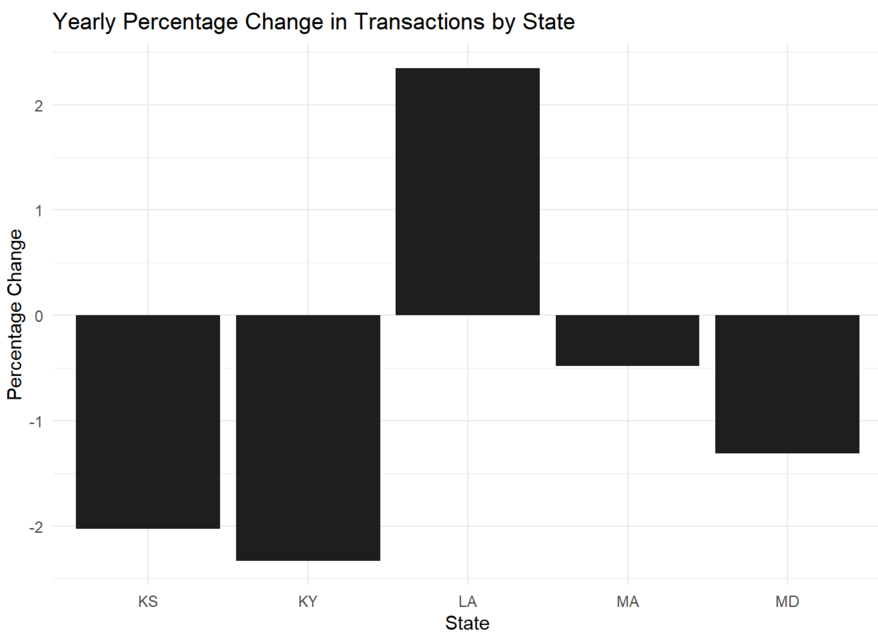


The bar graph shows the difference in yearly transactions between 2023 and 2024. 4 out of the 5 states saw a decline in the total amount of transactions except for Louisiana. This indicates that Louisiana may have growth potential.

Percentage Change in Transactions by Year

```
# Assign changes from table to new dataframe
trans_profile_address_viz <- trans_profile_address_df %>%
  group_by(State) %>%
  summarise(transactions = sum(TRANS_COUNT),
            trans_2023 = sum(TRANS_COUNT_2023),
            trans_2024 = sum(TRANS_COUNT_2024),
            difference = trans_2024 - trans_2023,
            pctg_change = round(((trans_2024 - trans_2023) / trans_2023) * 100, 2)) %>%
  arrange(desc(transactions)) %>%
  pivot_longer(cols = c(pctg_change), # Pivot the dataframe for easier plotting
               names_to = "Metric", values_to = "Value")


ggplot(trans_profile_address_viz, aes(x = State, y = Value, fill = Metric)) +
  geom_col(position = "dodge") +  # Dodge to separate bars
  theme_minimal() +
  labs(title = "Yearly Percentage Change in Transactions by State",
       y = "Percentage Change",
       x = "State",
       fill = "Transaction Type") +
  scale_fill_manual(values = c("pctg_change" = "#1E1E1E")) +
  theme(legend.position = "none")
```



The bar graph shows the percentage change in yearly transactions between 2023 and 2024. 4 out of the 5 states saw a decline except for Louisiana. As previously mentioned, this shows that Louisiana may have potential for growth in the future.

Volume by States

```
trans_profile_address_df %>%
  group_by(State) %>%
  summarise(volume_2023 = sum(ANNUAL_VOLUME_2023),
            gal_ordered_2023 = sum(ANNUAL_VOLUME_GALLON_2023),
            cases_ordered_2023 = sum(ANNUAL_VOLUME_CASES_2023),
            volume_2024 = sum(ANNUAL_VOLUME_2024),
            gal_ordered_2024 = sum(ANNUAL_VOLUME_GALLON_2024),
            cases_ordered_2024 = sum(ANNUAL_VOLUME_CASES_2024),
            pctg_change = round(((volume_2024-volume_2023)/volume_2023)*100,2)) %>%
  arrange(desc(volume_2023))
```

State <chr>	volume_2023 <dbl>	gal_ordered_2023 <dbl>	cases_ordered_2023 <dbl>	volume_2024 <dbl>
MA	7612212.8	1829253.01	5782959.8	8181960.4
KS	4422923.9	1228806.35	3194117.6	4488791.8
KY	3966826.0	1154511.18	2812314.8	4190178.2

State	volume_2023	gal_ordered_2023	cases_ordered_2023	volume_2024
<chr>	<dbl>	<dbl>	<dbl>	<dbl>
MD	2547778.0	789762.87	1758015.1	2659844.3
LA	156191.8	52821.45	103370.4	171099.6

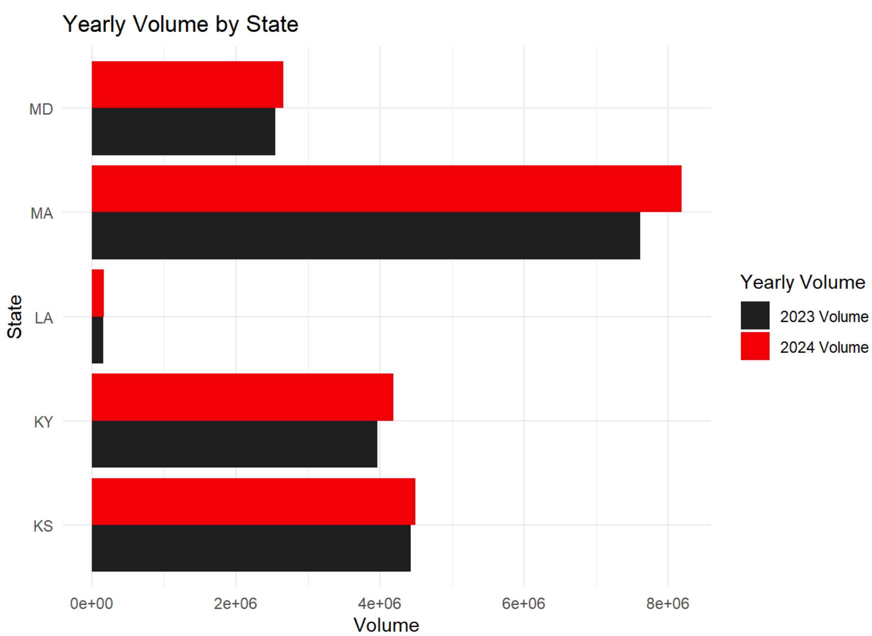
5 rows | 1-5 of 8 columns

Volume by State Visualizations

Yearly Volume by State

```
# Assign changes from table to new dataframe
trans_profile_address_viz <- trans_profile_address_df %>%
  group_by(State) %>%
  summarise(volume_2023 = sum(ANNUAL_VOLUME_2023),
            gal_ordered_2023 = sum(ANNUAL_VOLUME_GALLON_2023),
            cases_ordered_2023 = sum(ANNUAL_VOLUME_CASES_2023),
            volume_2024 = sum(ANNUAL_VOLUME_2024),
            gal_ordered_2024 = sum(ANNUAL_VOLUME_GALLON_2024),
            cases_ordered_2024 = sum(ANNUAL_VOLUME_CASES_2024),
            pctg_change = round(((volume_2024-volume_2023)/volume_2023)*100,2)) %>%
  pivot_longer(cols = c(volume_2023, volume_2024), # Pivot the dataframe for easier plotting
               names_to = "Metric", values_to = "Value")


ggplot(trans_profile_address_viz, aes(x = State, y = Value, fill = Metric)) +
  geom_col(position = "dodge") +  # Dodge to separate bars
  theme_minimal() +
  labs(title = "Yearly Volume by State",
       y = "Volume",
       x = "State",
       fill = "Yearly Volume") +
  coord_flip() +
  scale_fill_manual(values = c("volume_2023" = "#1E1E1E", "volume_2024" = "#F40009"),
                    labels = c("2023 Volume ","2024 Volume"))
```

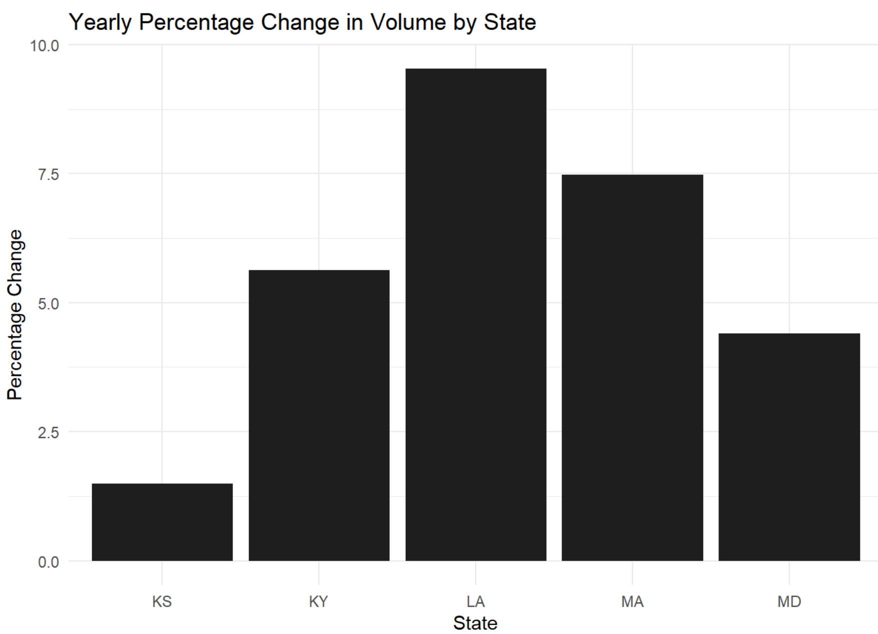


The graph shows the yearly change in the annual volume of gallons ordered for 2023 and 2024. All of the states saw increases in the volume of gallons ordered from 2023 to 2024.

Yearly Volume by State

```
# Assign changes from table to new dataframe
trans_profile_address_viz <- trans_profile_address_df %>%
  group_by(State) %>%
  summarise(volume_2023 = sum(ANNUAL_VOLUME_2023),
            gal_ordered_2023 = sum(ANNUAL_VOLUME_GALLON_2023),
            cases_ordered_2023 = sum(ANNUAL_VOLUME_CASES_2023),
            volume_2024 = sum(ANNUAL_VOLUME_2024),
            gal_ordered_2024 = sum(ANNUAL_VOLUME_GALLON_2024),
            cases_ordered_2024 = sum(ANNUAL_VOLUME_CASES_2024),
            pctg_change = round(((volume_2024-volume_2023)/volume_2023)*100,2)) %>%
  pivot_longer(cols = c(pctg_change), # Pivot the dataframe for easier plotting
               names_to = "Metric", values_to = "Value")


ggplot(trans_profile_address_viz, aes(x = State, y = Value, fill = Metric)) +
  geom_col(position = "dodge") +  # Dodge to separate bars
  theme_minimal() +
  labs(title = "Yearly Percentage Change in Volume by State",
       y = "Percentage Change",
       x = "State",
       fill = "Transaction Type") +
  scale_fill_manual(values = c("pctg_change" = "#1E1E1E")) +
  theme(legend.position = "none")
```



This graph shows the yearly percentage change in volume for each of the states. Lousiana has the largest increase in volume from 2023 to 2024.

Average Volume by States

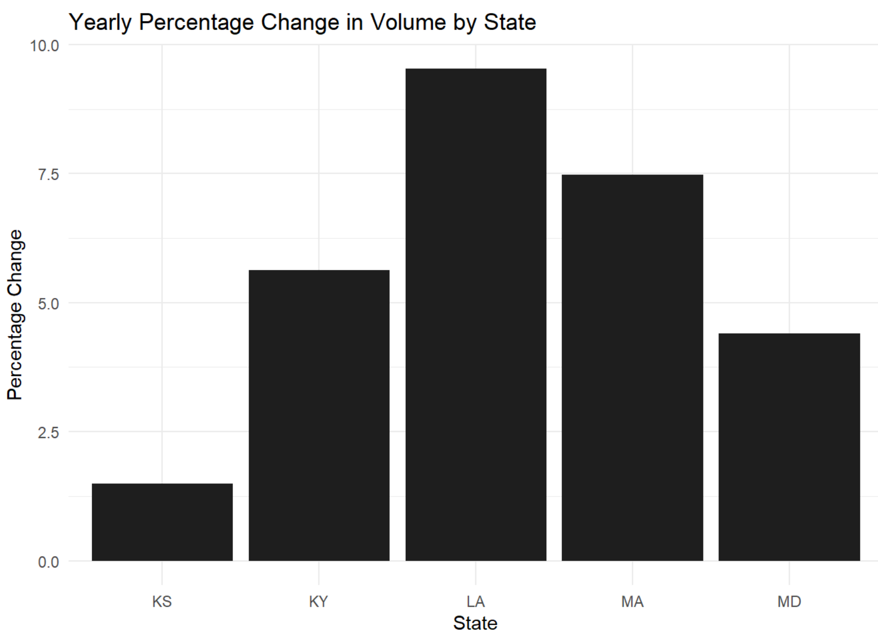
```
trans_profile_address_df %>%
  group_by(State) %>%
  summarise(avg_vol = sum(ANNUAL_VOLUME_2023 + ANNUAL_VOLUME_2024)/n(),
            avg_vol_2023 = sum(ANNUAL_VOLUME_2023)/sum(TRANS_COUNT_2023),
            avg_vol_2024 = sum(ANNUAL_VOLUME_2024)/sum(TRANS_COUNT_2024),
            pctg_change = round(((avg_vol_2024 - avg_vol_2023)/avg_vol_2023)*100,2)) %>%
  arrange(desc(avg_vol))
```

State <chr>	avg_vol <dbl>	avg_vol_2023 <dbl>	avg_vol_2024 <dbl>	pctg_change <dbl>
MA	1439.7606	40.27114	43.49446	8.00
KS	1249.3643	34.90726	36.15969	3.59
KY	1172.4887	32.71070	35.37777	8.15
MD	1068.0111	30.63227	32.40393	5.78
LA	847.9054	25.10316	26.86866	7.03

5 rows

```
# Assign changes from table to new dataframe
trans_profile_address_viz <- trans_profile_address_df %>%
  group_by(State) %>%
  summarise(volume_2023 = sum(ANNUAL_VOLUME_2023),
            gal_ordered_2023 = sum(ANNUAL_VOLUME_GALLON_2023),
            cases_ordered_2023 = sum(ANNUAL_VOLUME_CASES_2023),
            volume_2024 = sum(ANNUAL_VOLUME_2024),
            gal_ordered_2024 = sum(ANNUAL_VOLUME_GALLON_2024),
            cases_ordered_2024 = sum(ANNUAL_VOLUME_CASES_2024),
            pctg_change = round(((volume_2024-volume_2023)/volume_2023)*100,2)) %>%
  pivot_longer(cols = c(pctg_change), # Pivot the dataframe for easier plotting
               names_to = "Metric", values_to = "Value")


ggplot(trans_profile_address_viz, aes(x = State, y = Value, fill = Metric)) +
  geom_col(position = "dodge") +  # Dodge to separate bars
  theme_minimal() +
  labs(title = "Yearly Percentage Change in Volume by State",
       y = "Percentage Change",
       x = "State",
       fill = "Transaction Type") +
  scale_fill_manual(values = c("pctg_change" = "#1E1E1E")) +
  theme(legend.position = "none")
```



Local Transaction Partner per State Count

```
trans_profile_address_df %>%
  filter(CO2_CUSTOMER != FALSE) %>%
  group_by(State,LOCAL_MARKET_PARTNER) %>%
  summarise(n = n())
```

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

State <chr>	LOCAL_MARKET_PARTNER <lgl>	n <int>
KS	FALSE	331
KS	TRUE	2473
KY	FALSE	316
KY	TRUE	2457

State <chr>	LOCAL_MARKET_PARTNER <lgl>	n <int>
LA	FALSE	16
LA	TRUE	108
MA	FALSE	477
MA	TRUE	3762
MD	FALSE	228
MD	TRUE	1675

1-10 of 10 rows

In every state, Local Market Partners are the majority. `True` represents Local Market Partners while `False` means that they are **not** Local Market Partners.