Case Study: Coffee Shop Sales

Introduction

This case study is a capstone of the Google Data Analytics Professional Certificate. In this case study, we work for a fictional coffee shop.

We are acting as a junior data analyst working on the marketing analyst team at a coffee shop. The top management wants to understand the sales of the coffee shop.

There is no no defined objective other than an exhaustive analysis of sales trends. This will make it possible to place emphasis on certain products or, on the contrary, remove certain products.

In order to answer the business questions, the steps of the data analysis process: Ask, Prepare, Process, Analyze, Share, and Act, will be followed.

Business Case

The Coffee Shop Company

The company has 3 sales stores located in different areas of the city. These neighborhoods are either family, tourist or business.

Different products, divided into different categories, are sold in the 3 stores. There we could find products for pastries, individual pastries, cakes, lunch, products ready to be taken away, drinks, coffees, teas…

Problematic

The team wants to know what customer habits are in great detail. The current economy encourages users to be more restrained. Every employee has noticed that habits have changed. It is vital for the company to confirm or deny certain habits.

An initial analysis is requested to have an overall overview.

Ask

An initial analysis is recommended to draw initial conclusions. Future analyses will subsequently be commissioned following the findings.

We will have to analyze the Coffee shop stores data to identify trends.

Prepare

Data location

The dataset is coming from the point of sale system. The data are exported in a CSV file.

Good data?

Data is ROCCC compliant as it is:

- **R**eliable: data represents the sales from all users of the 3 stores with no bias. The dataset has only been cleaned to remove some unwanted categories, and to rename some product names. The prices have also been altered with the same constant.
- **O**riginal: The data comes from a direct source, not from an intermediate or a tier source.
- **C**omprehensive:The dataset is comprehensive and has all the information we need.
- **C**urrent: The dataset is up to date and relevant. Data is refreshed at each transaction.
- **C**ited: The data are cited. An extract is on Kaggle. The source is known and reliable.

Data organization

The dataset is split into several dataset, one for each month in a CSV format. It is not required to download them 1 by 1. Extracting 1 full year will create 12 CSV files.

The case study spans the whole year of 2023.

Datasets structure

Each dataset contains a CSV file displaying data by transaction:

Column	Description	Example
Date	*Day of the transaction*	*2023-12-31*
Time	*Time of the transaction*	*6:08:28 PM*
Time Zone	*Time zone where Date and Time correspond*	*America/Vancouver*
Category	*Classification of the product*	*Bread*
Item	The product sold	*Multi-seed Loaf*
Qty	The number of corresponding items sold in the current transaction	*1*
Price Point Name	Detailed information about the item	*Demi Baguette*
SKU	String identifier that gather item and Price Point Name information	*Mul/See/Dem*
Gross Sales	*Amount displayed*	*58.5*
Discounts	Amount off	*5.85*
Net Sales	Amount paid by the user	*52.65*
Tax	Tax amount, already included in the sales amount	*0.48*
Transaction ID	Id of the transaction	*RE3xZhB0nqEgDCDlB5TExt6yACHZY*
Payment ID	Id of the payment - display of information such as Visa, MC, cash, Tips…	*r9SPGB4S129xpbGELxZpc4FvIKHZY*
Location	Location of the coffee shop retail store	*Chelsea*
Unit	Unit of item sold	*ea*

First Data integrity Check

First data integrity for each file is performed with a spreadsheet like Excel or Google Spreadsheet. 3505 transactions have a net sales equal to $0. The discount amount equals the gross amount. It could be free coffee for staff. Other than that, the dataset is very consistent. All fields have values that are logical. The only

thing that we can criticize is that some items are all in capital letters (“AMERICANO”) but it is not dramatic.

Process

In this step, we use R as a tool to check, clean and transform data to be ready for analysis. I would like to use SQL but I could not use Big Queries with the large dataset that we have to work on.

Read CSV files

I will use the tidyverse package:

```
if(!require(tidyverse)){
install.packages("tidyverse",repos = "http://cran.us.r-project.org")
library(tidyverse)
}
```

Then, read all the 12 CSV files

```
setwd("./datasets")
df2301 <- read.csv("items-2023-01-01-2023-02-01.csv")
df2302 <- read.csv("items-2023-02-01-2023-03-01.csv")
df2303 <- read.csv("items-2023-03-01-2023-04-01.csv")
df2304 <- read.csv("items-2023-04-01-2023-05-01.csv")
df2305 <- read.csv("items-2023-05-01-2023-06-01.csv")
df2306 <- read.csv("items-2023-06-01-2023-07-01.csv")
df2307 <- read.csv("items-2023-07-01-2023-08-01.csv")
df2308 <- read.csv("items-2023-08-01-2023-09-01.csv")
df2309 <- read.csv("items-2023-09-01-2023-10-01.csv")
df2310 <- read.csv("items-2023-10-01-2023-11-01.csv")
df2311 <- read.csv("items-2023-11-01-2023-12-01.csv")
df2312 <- read.csv("items-2023-12-01-2024-01-01.csv")
setwd("../")
```

All datasets are merged into one data frame dfSale.

```
# Merge Datasets
dfSale = rbind(df2301,df2302,df2303,df2304,df2305,df2306,df2307,df2308,df2309,df2310,df2311,df2312)
rm(df2301,df2302,df2303,df2304,df2305,df2306,df2307,df2308,df2309,df2310,df2311,df2312)
```

We check the dimension of the dataframe

```
dim(dfSale)
```

```
## [1] 485237     29
```

Let’s check the structure.

```
str(dfSale)
```

```
## 'data.frame':    485237 obs. of  29 variables:
##  $ Date                  : chr  "2023-01-31" "2023-01-31" "2023-01-31" "2023-01-31" ...
##  $ Time                  : chr  "18:36:42" "18:36:42" "18:19:36" "18:18:33" ...
##  $ Time.Zone             : chr  "America/Vancouver" "America/Vancouver" "America/Vancouver"
"America/Vancouver" ...
##  $ Category              : chr  "Macaron" "Bread" "Hot Drinks" "Hot Drinks" ...
##  $ Item                  : chr  "Macaron Each" "Croton/ Crostini" "French Caesar" "Seasonal
Hot Chocolate Bomb" ...
##  $ Qty                   : num  4 1 1 1 1 1 1 1 1 1 ...
##  $ Price.Point.Name      : chr  "Regular" "Regular" "" "" ...
##  $ SKU                   : chr  "Mac/Ind" "" "" "" ...
##  $ Modifiers.Applied     : chr  "" "" "To go (new 2022 by law rule)" "To go (new 2022 by la
w rule)" ...
##  $ Gross.Sales           : chr  "$10.80" "$4.30" "$18.25" "$8.25" ...
##  $ Discounts             : chr  "$0.00" "$0.00" "$0.00" "$0.00" ...
##  $ Net.Sales             : chr  "$10.80" "$4.30" "$18.25" "$8.25" ...
##  $ Tax                   : chr  "$0.54" "$0.00" "$0.91" "$0.41" ...
##  $ Transaction.ID        : chr  "9YjKGHzJCTXfkmupKh8kq70eV" "9YjKGHzJCTXfkmupKh8kq70eV" "LH
TCrrdxCReXezmuNR9hl73eV" "DHZIJjgI0czrebnod7S7RupeV" ...
##  $ Payment.ID            : chr  "dOwISV6O6Mo2Cl2U0TFCf5wZuaB" "dOwISV6O6Mo2Cl2U0TFCf5wZuaB"
"DDbJMfFshVdztTl8p9QtrMSmuaB" "VyAaGUdrjOWkPz3W8zdmvzZruaB" ...
##  $ Device.Name           : chr  "" "" "" "" ...
##  $ Notes                 : chr  "" "" "Dark Hot Chocolate with whipping cream and chocolate
drizzle\n\ntopped with skewer of madeleine, canele, palmie"| __truncated__ "A cup of hot milk
into which is dropped a chocolate sphere filled with coca powder and marshmallow\n\nhot choc
o"| __truncated__ ...
##  $ Details               : chr  "https://app.squareup.com/dashboard/sales/transactions/9YjK
GHzJCTXfkmupKh8kq70eV/by-unit/KA2E2C43551PM" "https://app.squareup.com/dashboard/sales/transa
ctions/9YjKGHzJCTXfkmupKh8kq70eV/by-unit/KA2E2C43551PM" "https://app.squareup.com/dashboard/s
ales/transactions/LHTCrrdxCReXezmuNR9hl73eV/by-unit/KA2E2C43551PM" "https://app.squareup.com/
dashboard/sales/transactions/DHZIJjgI0czrebnod7S7RupeV/by-unit/KA2E2C43551PM" ...
##  $ Event.Type            : chr  "Payment" "Payment" "Payment" "Payment" ...
##  $ Location              : chr  "Forest Hills" "Forest Hills" "Forest Hills" "Forest Hills"
...
##  $ Dining.Option         : chr  NA NA NA NA ...
##  $ Customer.ID           : chr  "VHWWCFT6WD4C13G5XQ0G7VC2AM" "VHWWCFT6WD4C13G5XQ0G7VC2AM" "
" "" ...
##  $ Customer.Name         : chr  "LEIQING Wang" "LEIQING Wang" "" "" ...
##  $ Customer.Reference.ID: chr  "" "" "" "" ...
##  $ Unit                  : chr  "ea" "ea" "ea" "ea" ...
##  $ Count                 : int  4 1 1 1 1 1 1 1 1 1 ...
##  $ Itemization.Type      : chr  "Physical Item" "Physical Item" "Physical Item" "Physical I
tem" ...
##  $ Fulfillment.Note      : chr  "" "" "" "" ...
##  $ Token                 : chr  "7RE3ZLSGHYTH2JYZ5Q3M5W3C" "43XBEHIOK2BXADGUW7QAF3HZ" "DDR6
HLND5MFHBGTQDN5KBENG" "GS5Q7I3ROAIPSMY2EM7INIP2" ...
```

Looking for inconsistencies

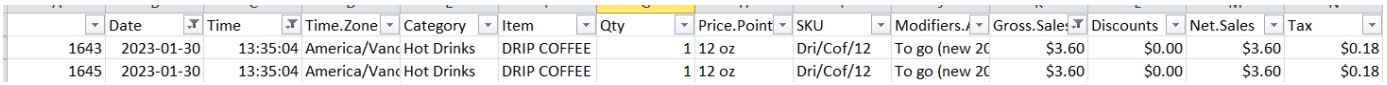
We first check for duplicated rows.

```
dfSale[dfSale %>% duplicated(),] %>% count()
```

```
##     n
## 1 707
```

707 observations are duplicated.

Here is an example:



	Date	Time	Time.Zone	Category	Item	Qty	Price.Point	SKU	Modifiers.A	Gross.Sales	Discounts	Net.Sales	Tax
1643	2023-01-30	13:35:04	America/Vanc	Hot Drinks	DRIP COFFEE	1	12 oz	Dri/Cof/12	To go (new 20	$3.60	$0.00	$3.60	$0.18
1645	2023-01-30	13:35:04	America/Vanc	Hot Drinks	DRIP COFFEE	1	12 oz	Dri/Cof/12	To go (new 20	$3.60	$0.00	$3.60	$0.18

Let us remove the duplicate.

```
# remove duplicated observations
dfSale <- distinct(dfSale)
# check and remove duplicated observations
dfSale[dfSale %>% duplicated(),] %>% count()
```

```
##   n
## 1 0
```

Date

There are no missing values and the range of values correspond to the first and last day of the year 2023.

```
dfSale %>% filter(is.na(Date)) %>% count()
```

```
##   n
## 1 0
```

```
dfSale %>%
  summarise (min_Date = min(as.Date(Date)),
             max_Date = max(as.Date(Date)))
```

```
##     min_Date   max_Date
## 1 2023-01-01 2023-12-31
```

Time

Same as the Date feature, everything is correct. However, it could be surprising to have start and end time around midnight.

```
dfSale %>% filter(is.na(Time)) %>% count()
```

```
##   n
## 1 0
```

```
dfSale %>%
  summarise (min_Time = min(Time),
             max_Time = max(Time))
```

```
##   min_Time max_Time
## 1 00:14:46 23:45:34
```

Category

We notice that we have no missing values. The company split the items into 12 categories. At the end we have the distribution of the transactions per category. The hot drinks seem to be very popular, contrary to the cold drinks.

```
dfSale %>% filter(is.na(Category)) %>% count()
```

```
##   n
## 1 0
```

```
unique(dfSale$Category)
```

```
##  [1] "Macaron"           "Bread"             "Hot Drinks"
##  [4] "Pastry Individual" "Pastry Cake"       "Viennoiserie"
##  [7] "Takeaway"          "Lunch"             "Beverage"
## [10] "Smoothie"          "Crepe"             "Cold Drinks"
```

```
dfSale %>%
  group_by(Category) %>%
  count()
```

```
## # A tibble: 12 × 2
## # Groups:   Category [12]
##    Category               n
##    <chr>              <int>
##  1 Beverage            7256
##  2 Bread              17634
##  3 Cold Drinks         1742
##  4 Crepe              16760
##  5 Hot Drinks        204419
##  6 Lunch              62329
##  7 Macaron            11683
##  8 Pastry Cake         3939
##  9 Pastry Individual  22872
## 10 Smoothie            6066
## 11 Takeaway           32979
## 12 Viennoiserie       96851
```

Item

There are no missing values

```
dfSale %>% filter(is.na(Item)) %>% count()
```

```
##   n
## 1 0
```

Some items are in uppercase (“HOT CHOCOLATE”). Let’s fix it to have camel case (“Hot Chocolate”)

```
dfSale$Item <- str_to_title(dfSale$Item)
```

Qty

All quantities are known and the range is from 1 unit bought to 66. It may be a high volume to buy 66 items but after checking they are mainly macarons or mini viennoiseries.

```
dfSale %>% filter(is.na(Qty)) %>% count()
```

```
##   n
## 1 0
```

```
dfSale %>%
  summarise (min_Qty = min(Qty),
  max_Qty = max(Qty))
```

```
##   min_Qty max_Qty
## 1       1      66
```

That could be an event as the number of high transactions is low. Here below the number of transactions where

more than 10 items have been purchased.

```
dfSale %>%
  filter (Qty > 10) %>%
  count()
```

```
##    n
## 1 67
```

Price.Point.Name

All values are known

```
dfSale %>% filter(is.na(Price.Point.Name)) %>% count()
```

```
##   n
## 1 0
```

Like the Item feature, we update case of Price.Point.Name for harmonization

```
dfSale$Price.Point.Name <- str_to_title(dfSale$Price.Point.Name)
```

Gross.Sales, Discounts, Net.Sales, Tax

Gross sales have no missing values but can be equal to $0. They are removed as it can be a rebate, an error, or a return.

```
dfSale %>% filter(is.na(Gross.Sales)) %>% count()
```

```
##   n
## 1 0
```

```
dfSale %>% filter(Gross.Sales == "$0.00") %>% count()
```

```
##     n
## 1 133
```

```
dfSale <- dfSale %>% filter(!(Gross.Sales == "$0.00"))
```

```
dfSale %>% filter(Gross.Sales == "$0.00") %>% count()
```

```
##   n
## 1 0
```

Discounts is never empty and could be equal to $0, which is normal when no discount is applied.

```
dfSale %>% filter(is.na(Discounts)) %>% count()
```

```
##   n
## 1 0
```

```
dfSale %>% filter(Discounts == "$0.00") %>% count()
```

```
##        n
## 1 444488
```

Net.Sales is not empty and could be equal to $0. It happens when Gross.Sales and Discounts are equals.

```
dfSale %>% filter(is.na(Net.Sales)) %>% count()
```

```
##   n
## 1 0
```

```
dfSale %>% filter(Net.Sales == "$0.00") %>% count()
```

```
##      n
## 1 3369
```

Tax field is always filled and could be equal to $0.

```
dfSale %>% filter(is.na(Tax)) %>% count()
```

```
##   n
## 1 0
```

```
dfSale %>% filter(Tax == "$0.00") %>% count()
```

```
##       n
## 1 26728
```

Location

No values are missing. Only 3 values are present.

```
unique(dfSale$Location)
```

```
## [1] "Forest Hills" "Chelsea"      "FiDi"
```

```
dfSale %>%
  group_by(Location) %>%
  count()
```

```
## # A tibble: 3 × 2
## # Groups:   Location [3]
##   Location          n
##   <chr>         <int>
## 1 Chelsea      202882
## 2 FiDi         103872
## 3 Forest Hills 177643
```

Unit

No Missing values. Everything is sold by each.

```
dfSale %>% filter(is.na(Unit)) %>% count()
```

```
##   n
## 1 0
```

```
unique(dfSale$Unit)
```

```
## [1] "ea"
```

Transform

Let's create a DateTime column that gathers Date and Time. Then add the day of the week, and the month of the transaction. Finally check the minimum and maximum

```
dfSale$TrDatetime <- as_datetime(paste(dfSale$Date, dfSale$Time))
dfSale <- mutate(dfSale, tr_day_of_week = wday(TrDatetime, label = TRUE))
dfSale <- mutate(dfSale, tr_month = month(TrDatetime, label = TRUE))
dfSale %>%
  summarise (min_TrDatetime = min(as.Date(TrDatetime)),
  max_TrDatetime = max(as.Date(TrDatetime)))
```

```
##   min_TrDatetime max_TrDatetime
## 1     2023-01-01     2023-12-31
```

Gross.Sales, Discounts, and Net.Sales are String values. Let's transform them into numbers:

```
dfSale$Gross.Sales <- sub("\\$","",dfSale$Gross.Sales)
dfSale$Gross.Sales <- round(as.numeric(dfSale$Gross.Sales), digits = 2)
dfSale$Net.Sales <- sub("\\$","",dfSale$Net.Sales)
dfSale$Net.Sales <- round(as.numeric(dfSale$Net.Sales), digits = 2)
dfSale$Discounts <- sub("\\$","",dfSale$Discounts)
dfSale$Discounts <- sub("-","",dfSale$Discounts)
dfSale$Discounts <- round(as.numeric(dfSale$Discounts), digits = 2)
```

```
dfSale %>%
  summarise (min_Gross.Sales = min(Gross.Sales),
  max_Gross.Sales = max(Gross.Sales))
```

```
##   min_Gross.Sales max_Gross.Sales
## 1            0.66          229.95
```

```
dfSale %>%
  summarise (min_Discounts = min(Discounts),
             max_Discounts = max(Discounts))
```

```
##   min_Discounts max_Discounts
## 1             0         33.94
```

```
dfSale %>%
  summarise (min_Net.Sales = min(Net.Sales),
             max_Net.Sales = max(Net.Sales))
```

```
##   min_Net.Sales max_Net.Sales
## 1             0        229.95
```

Cleaning

We have already started cleaning as we removed the transactions with Gross.Sales equal to $0 earlier, as well as the duplicated transactions.

Some transactions have been voided. We remove them

```
dfSale %>% filter(str_detect(Item,'Voided'))  %>% count()
```

```
##   n
## 1 0
```

```
dfSale <- dfSale %>% filter(!(str_detect(Item,'Voided')))
dfSale %>% filter(str_detect(Item,'Voided')) %>% count()
```

```
##   n
## 1 0
```

Finally, we will work with 5,743,278 observations.

Analyze

Calculation

We compute the unit price for each transaction (Net.Sales divided by quantity).

```
dfSale <- mutate(dfSale, Unit.price = round((Net.Sales/Qty),2))
dfSale %>%
  summarise (min_Unit.price = min(Unit.price),
             max_Unit.price = max(Unit.price))
```

```
##   min_Unit.price max_Unit.price
## 1              0          116.8
```

As the products can be sold at different prices (regular prices, different discount rates, different rebates amount,…), we add the mean price for each product sold in a transaction. This mean price will be referred as the main price for the product:

```
dfSale <- dfSale %>%
  group_by(Category,Item,Price.Point.Name) %>%
  mutate(Mean_item = mean(Unit.price))
```

Then, we add the product item which is the concatenation of Item and Price.Point.Name

```
dfSale$Item_long <- paste(dfSale$Item, dfSale$Price.Point.Name)
```

Finally, we remove all the columns that we do not need for this case study.

```
dfSale <- select(dfSale, -c('Time.Zone', 'Transaction.ID', 'Payment.ID','Device.Name','Note
s', 'Details','Dining.Option','Customer.ID','Customer.Name','Customer.Reference.ID','Itemizat
ion.Type','Fulfillment.Note','Token','Event.Type','SKU','Modifiers.Applied','Count','Tax'))
```

Output CSV file

After being merged, cleaned and transformed, the dataset will be shared as a CSV file in the following analyses.

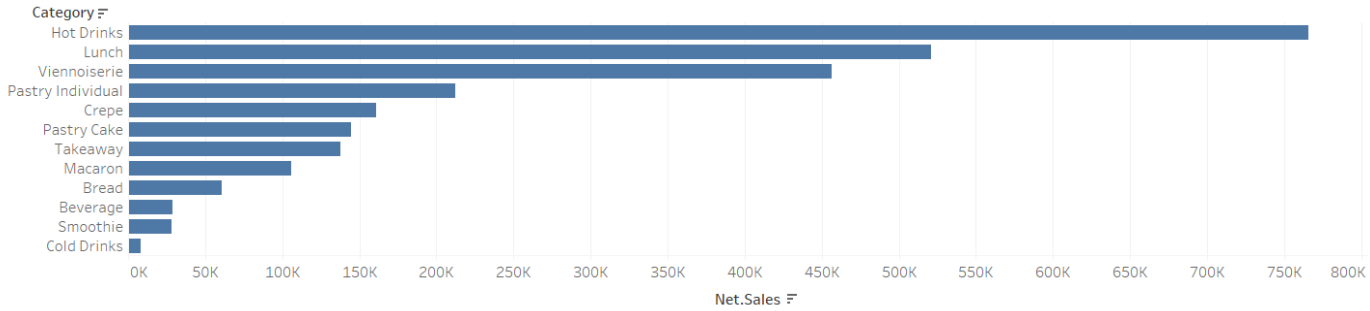
```
write.csv(dfSale, "items-sold-2023.csv")
```

The final CSV file will be used as an entry for the next analysis steps with Tableau.

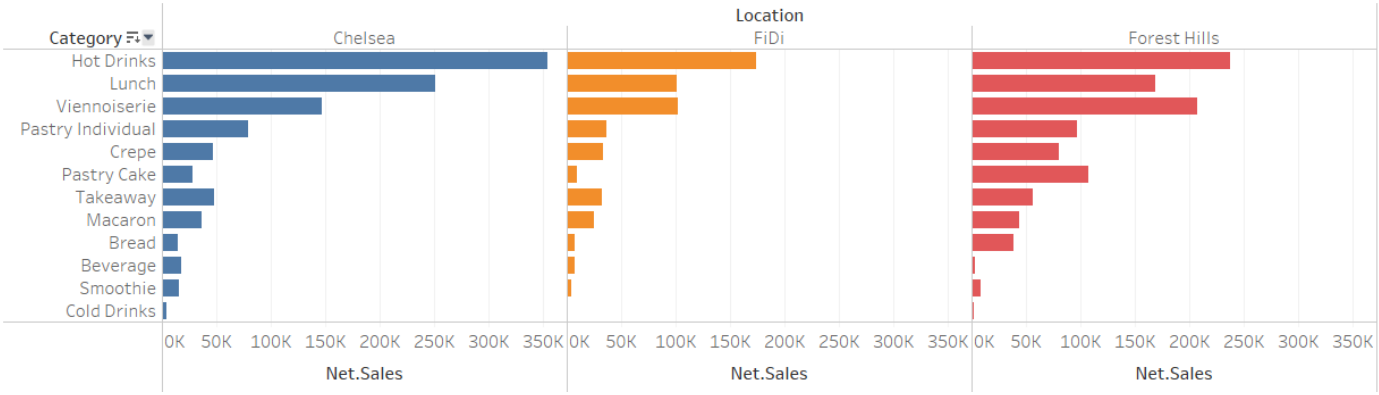
Share

What do the customers buy?

We take a look at what we sell the most.



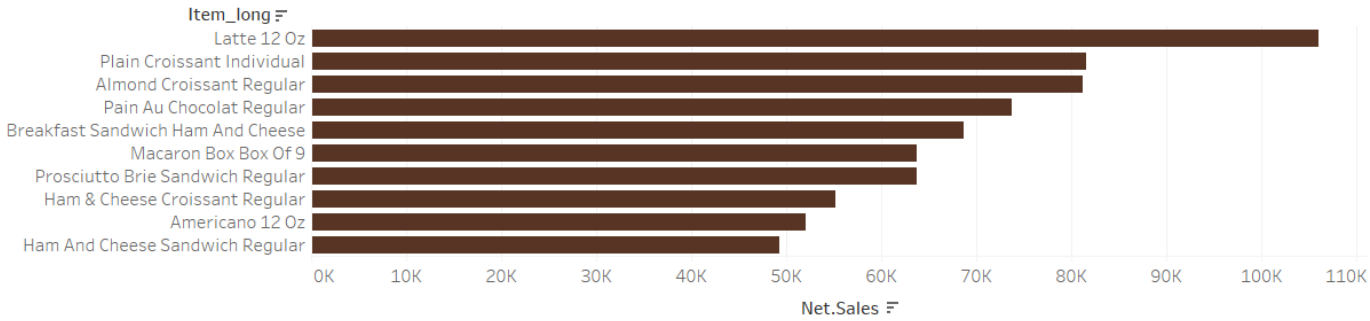
According to this bar graph, the Hot Drinks category generates the most revenue in the company, compared to the Cold Drinks category, which is at the bottom.



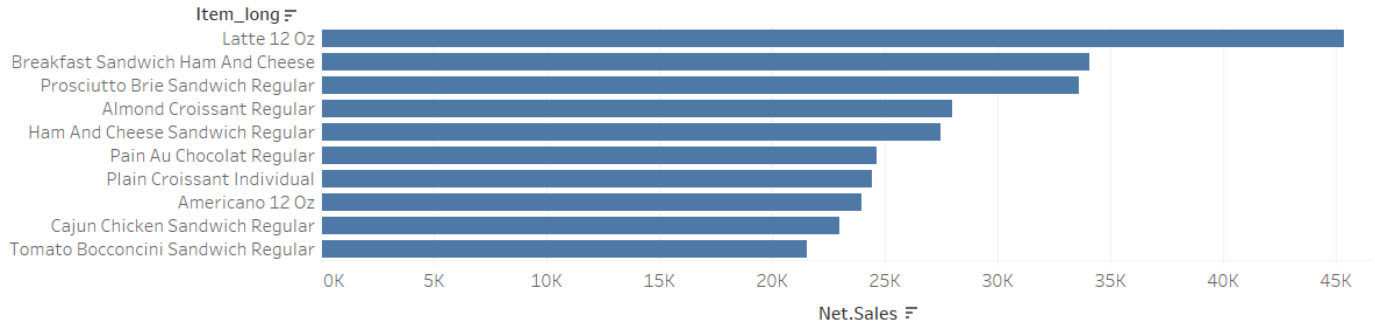
However, when we display this distribution by adding the location, we notice some differences between stores. The overall trend of the company is not necessarily the same at the location level:

- At **Chelsea**, they sell more Takeaway and Macarons than the overall trend.
- At **FiDi**, the Lunch and the Pastry Cake categories are not performant.
- At **Forest Hill**, The Lunch Category is not the best 2nd one, but the Pastry Cake is a specialty for this store.

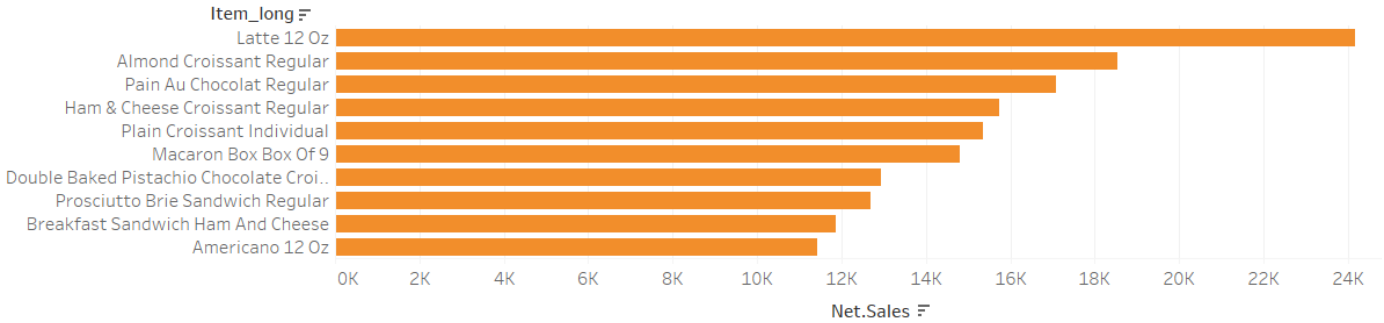
Here are the top 10 items sold at the company. Latte and Americano are indeed the best sellers.



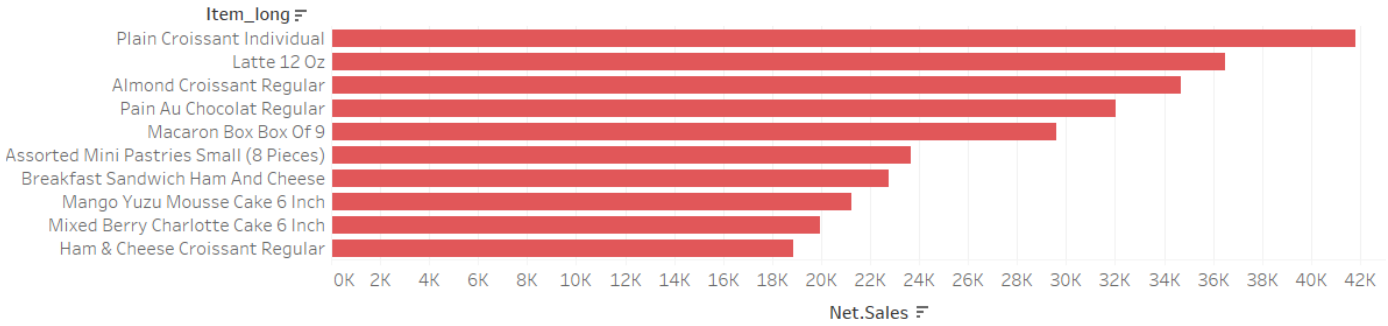
At **Chelsea**, the sandwiches seem to be a specialty.



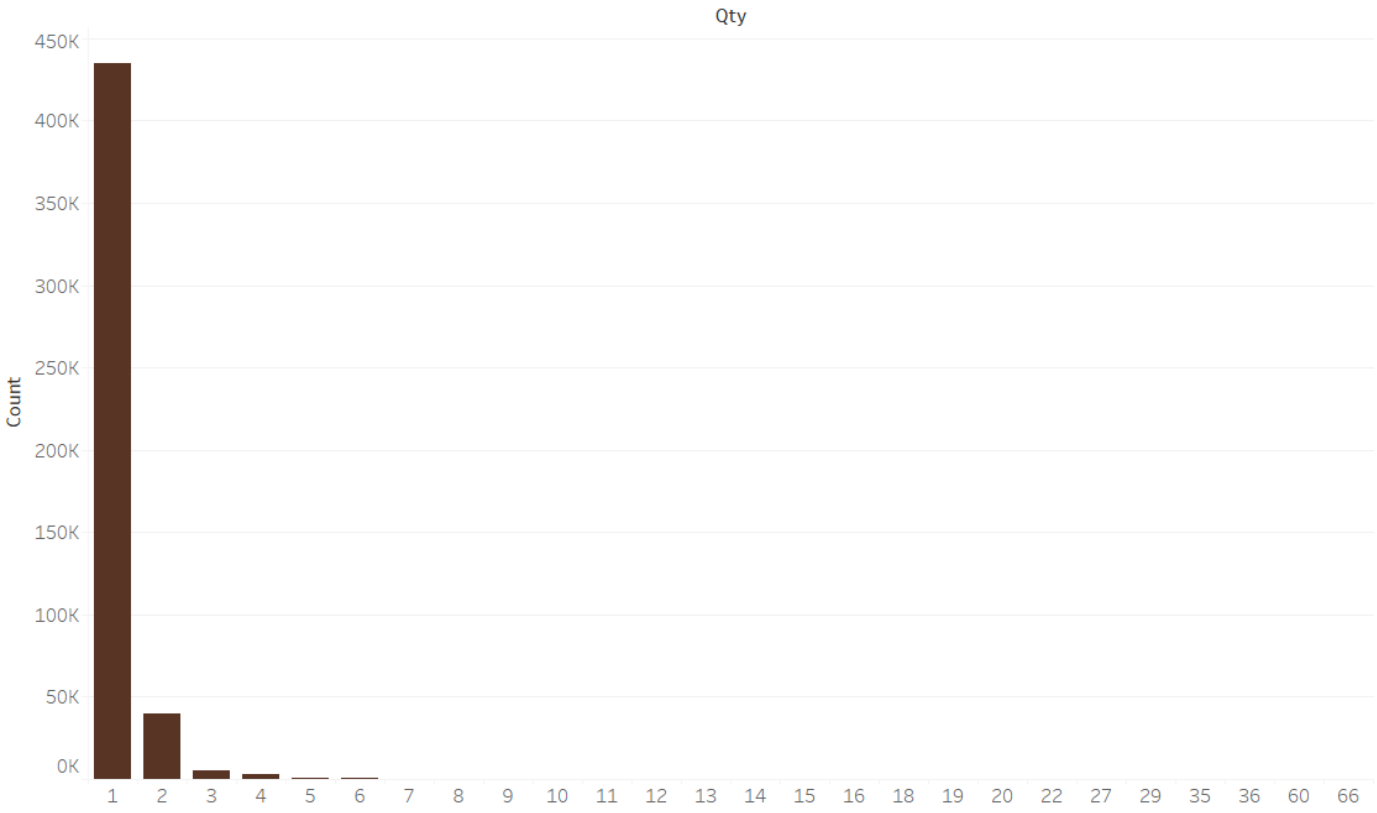
At **FiDi**, the breakdown shows similar items as the global trend. However, they sell twice less than **Chelsea**.



At **Forest Hills**, Latte is not the top item. They sell more cakes than the other stores.

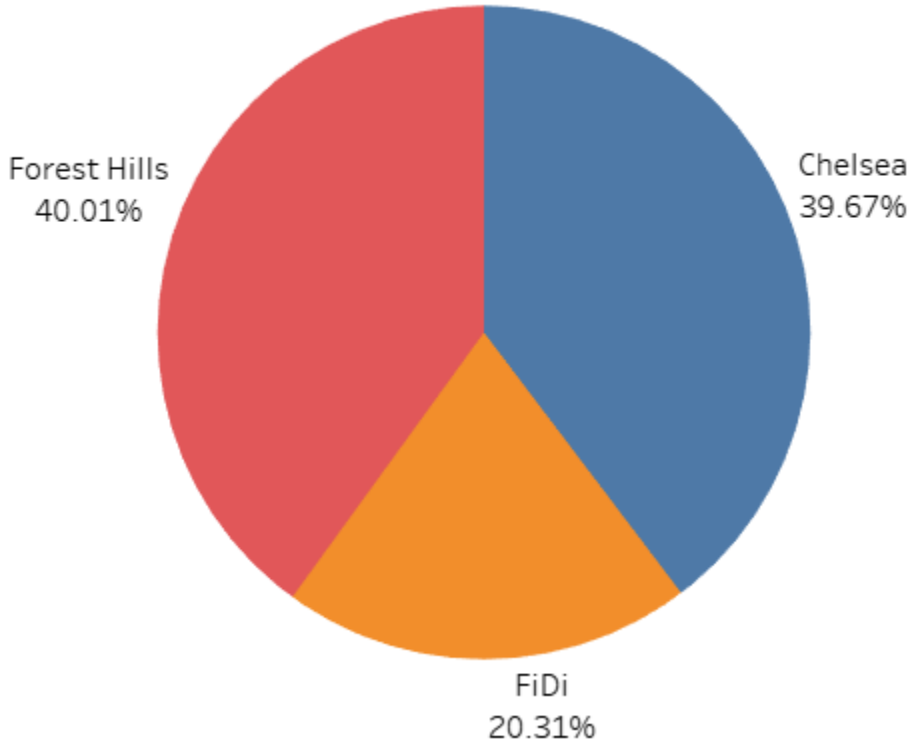


Customers mainly buy one item per transaction. 89.78% of transactions are composed of 1 and only 1 item bought.



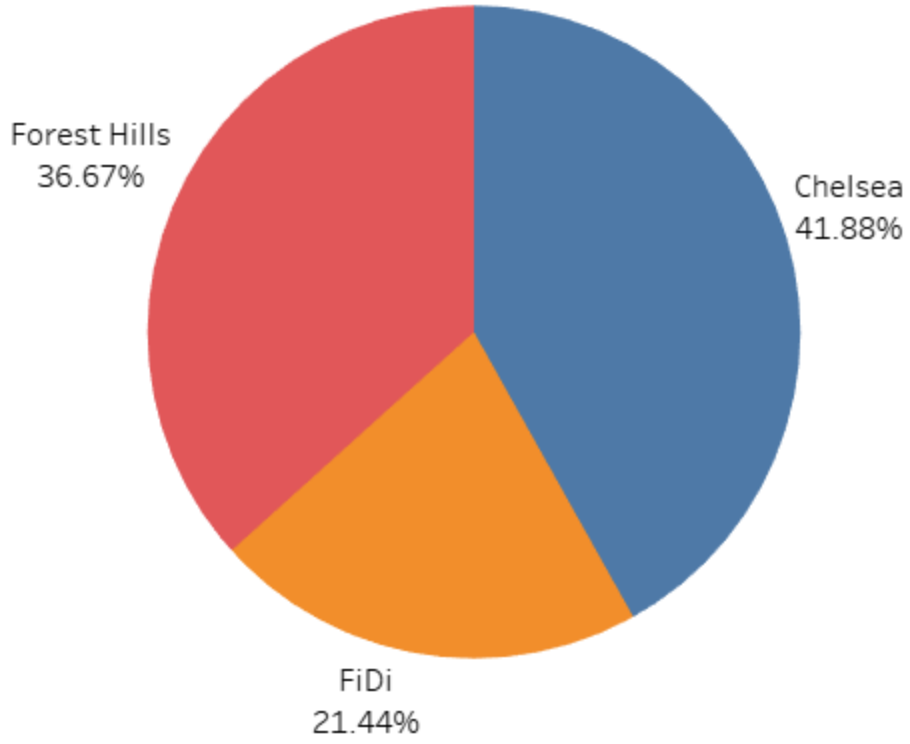
Where do they buy?

Net Sales: The **Chelsea** and **Forest Hills** store generate roughly the same income. The **other** one only generates the half.

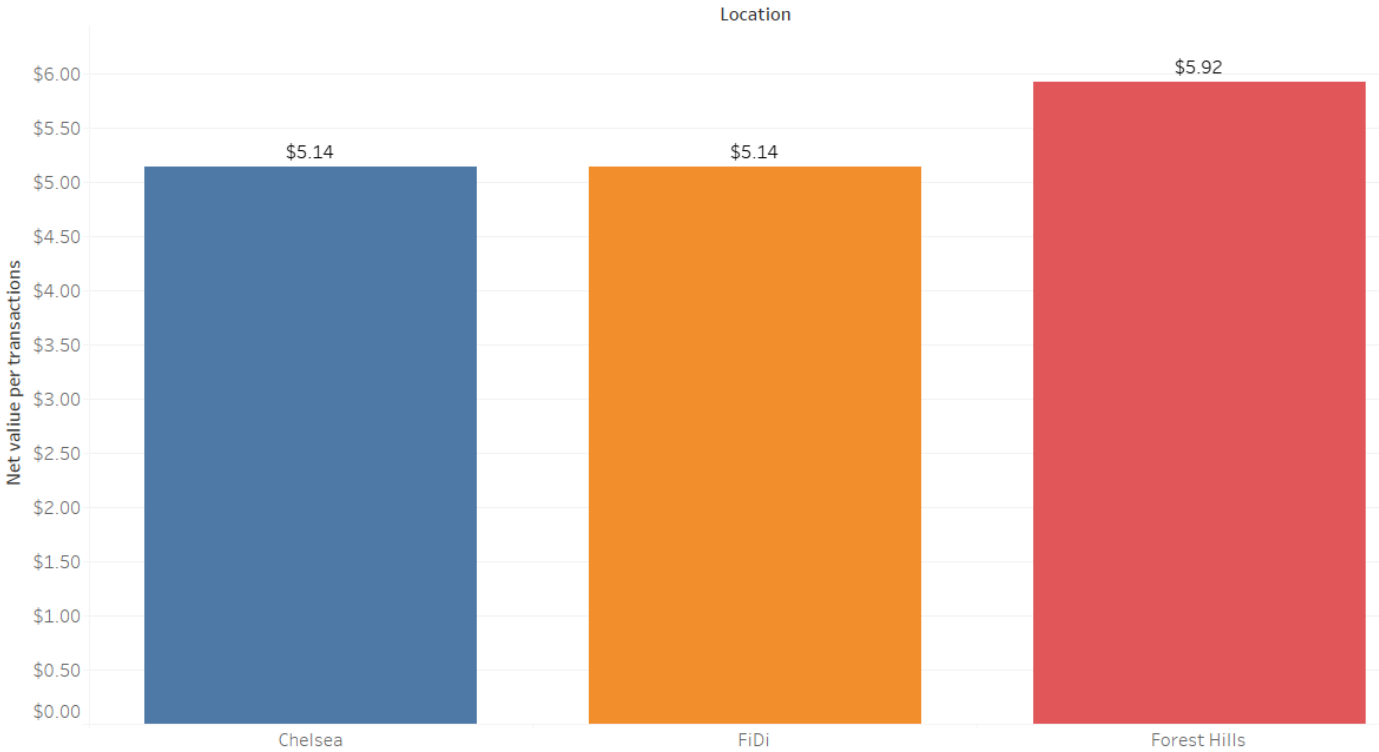


Number of transactions: The **Chelsea** store has more transactions, followed by **Forest Hills** right after. **FiDi** is

at the end with around half of the number of transactions compared to another store.

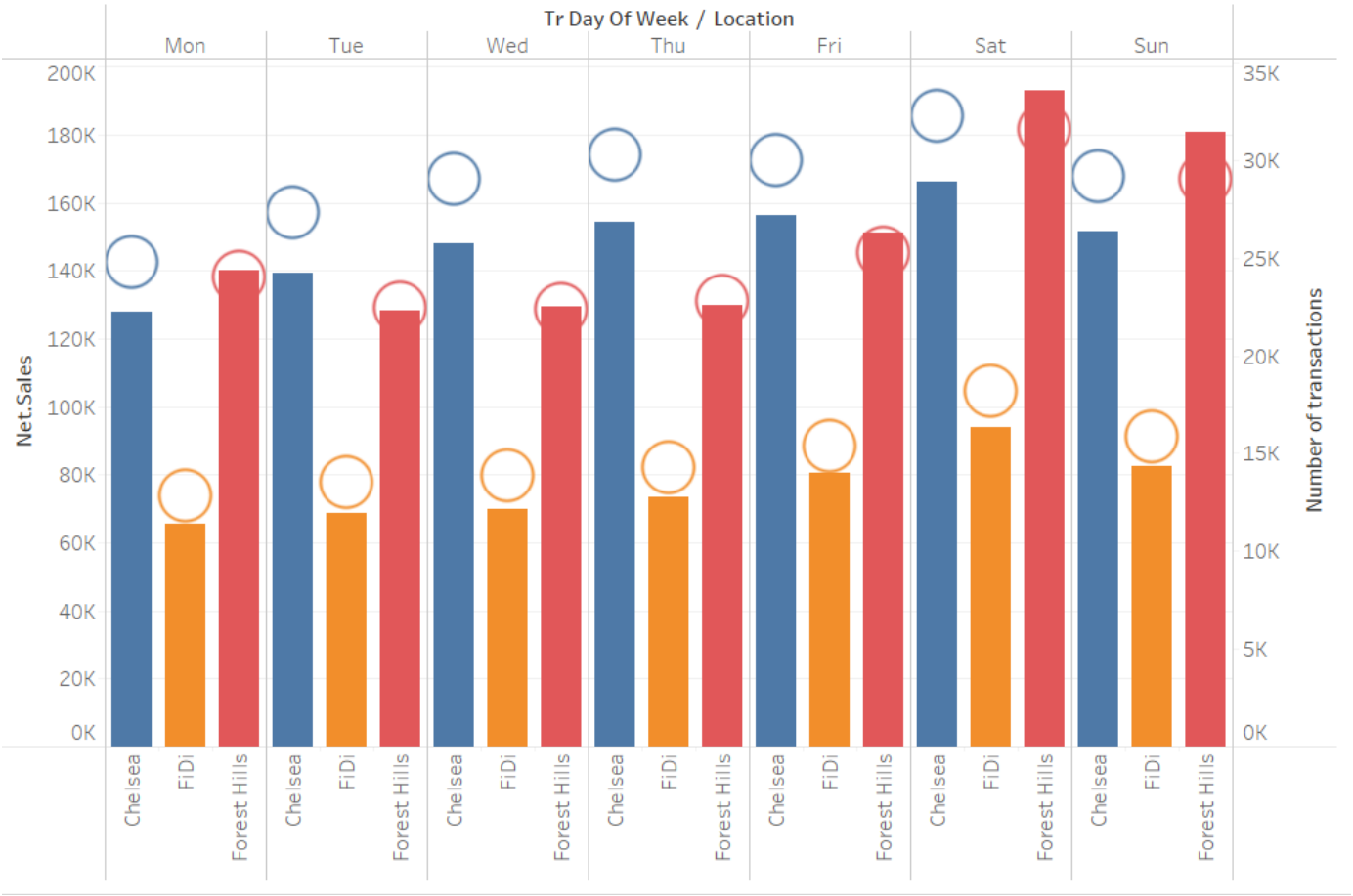


These 2 pie charts start to show a slight difference with **Chelsea** and **Forest Hills** stores. One sells to more customers, but the other one gains more per transaction.



Each transaction at **Forest Hills** brings in more than a transaction in another store. The net sale per transaction is exactly the same at **Chelsea** and at **FiDi**.

When do they buy?

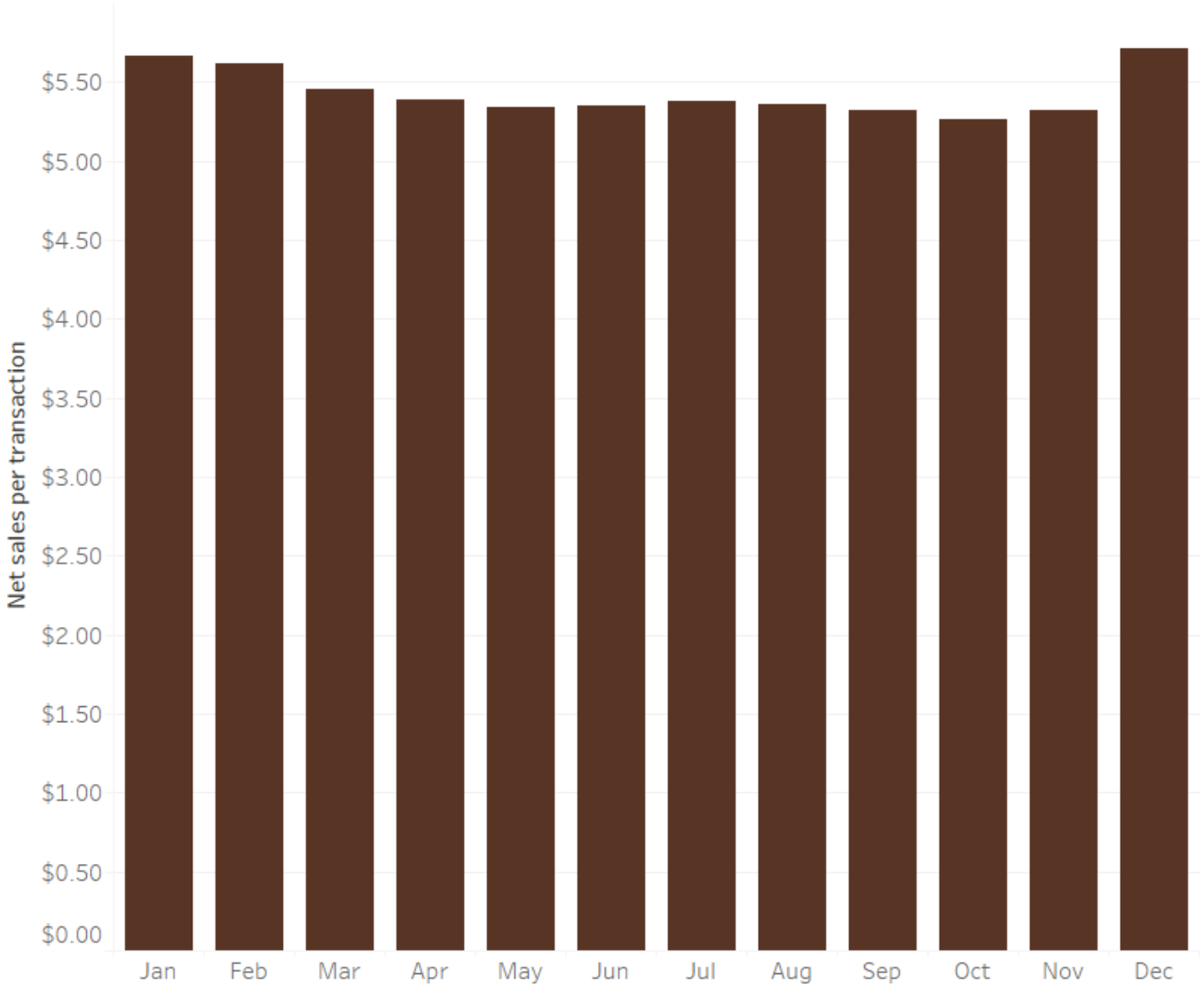


The bars and the left axis display the Net Sales of the store for each day.

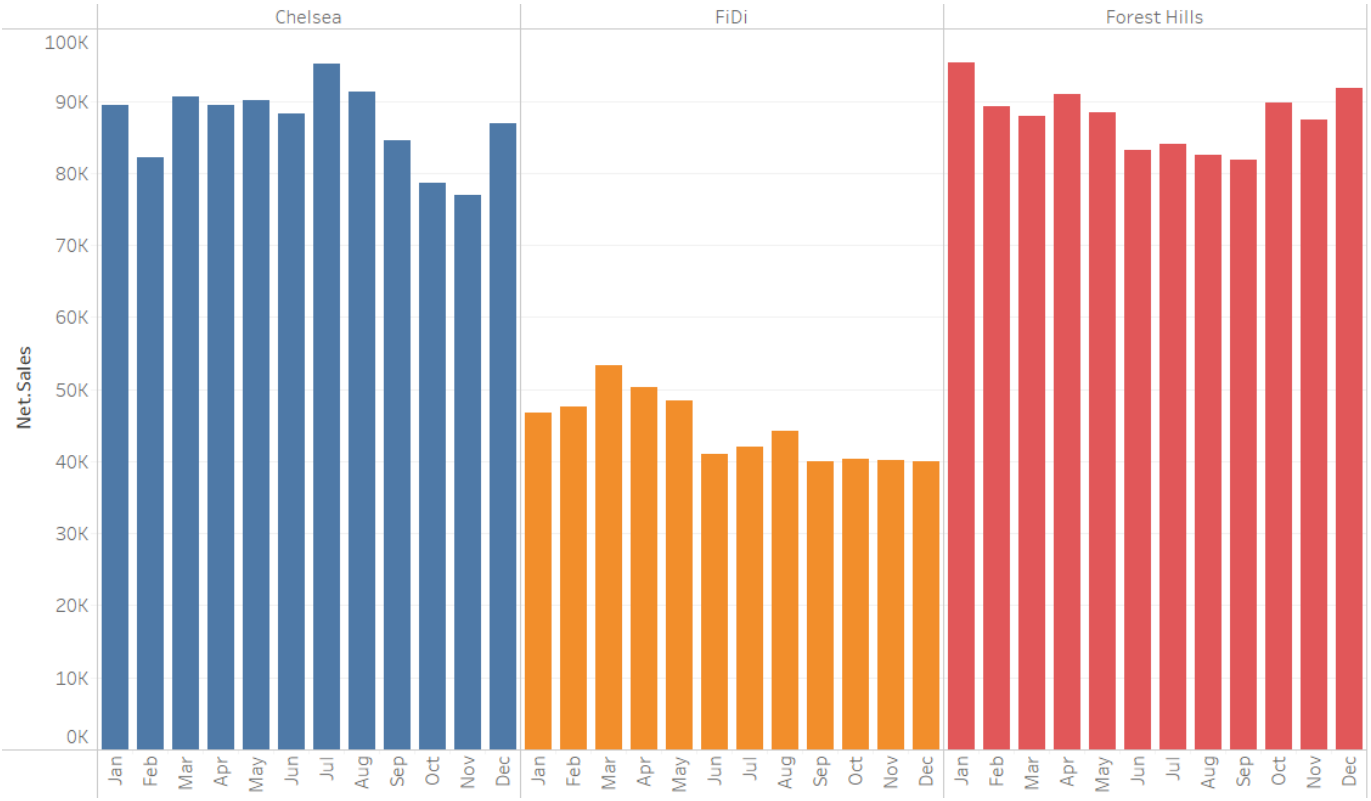
The rounds and the right axis display the number of transactions.

- Saturday is the biggest day for **all stores**
- Monday is the worst day for **Chelsea** and **FiDi**. For both of these stores, the net sale and the number of transactions increase from Monday to Saturday, to decrease on Sunday.
- Tuesday through Wednesday are the worst days for **Forest Hills**.
- We notice that the rounds (i.e. the number of transactions) are behind the bars (i.e. the net sales) only for **Forest Hills**. It means that each transaction at the store generates more revenue than a transaction for all other stores.
- The number of transactions is higher at **Chelsea** on each day. However, the revenue is not higher each day, as it should be expected.

The value generated by each transaction decreases little by little all along the year, with an exception for December due to the end of year celebrations.

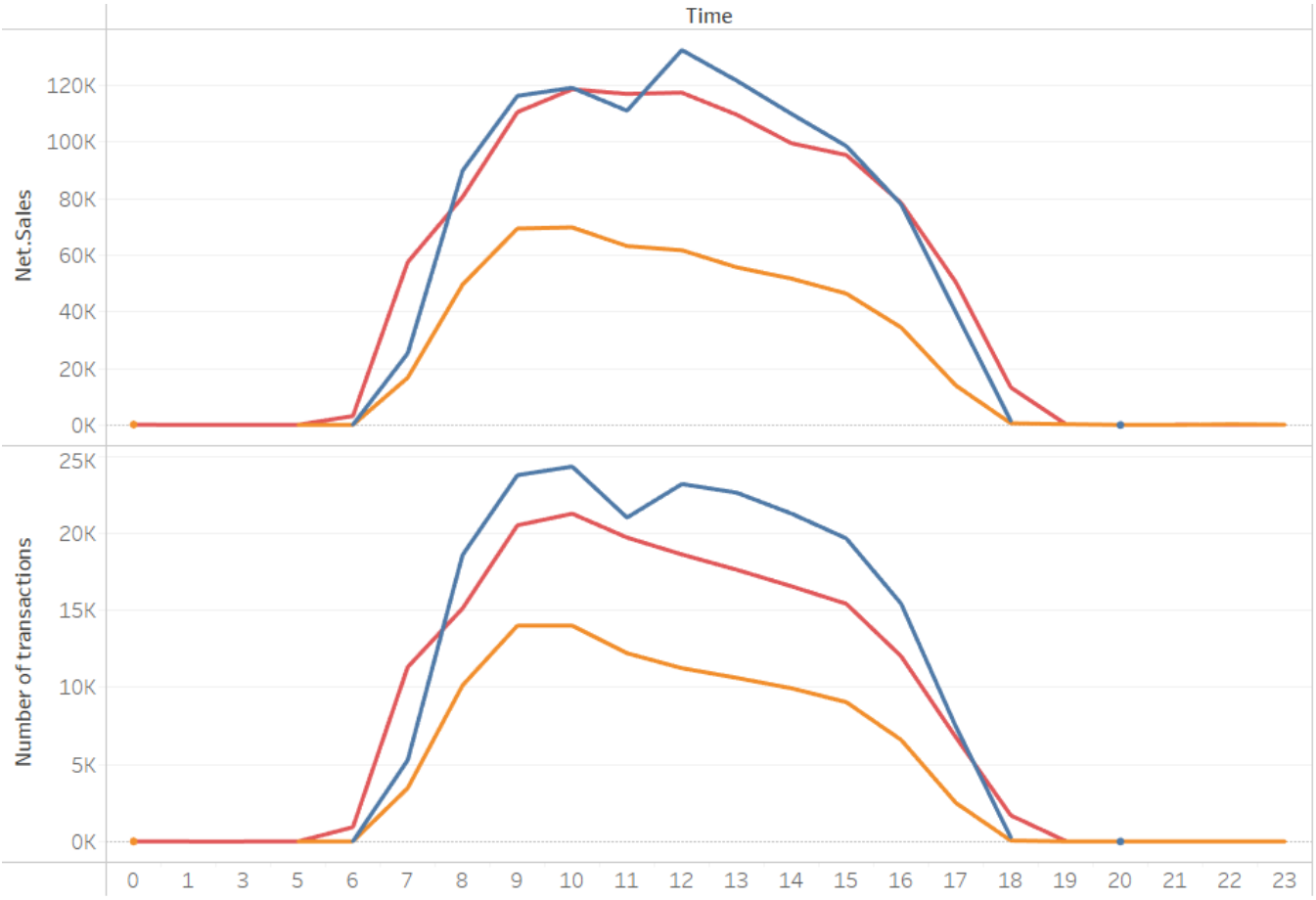


When do they buy?

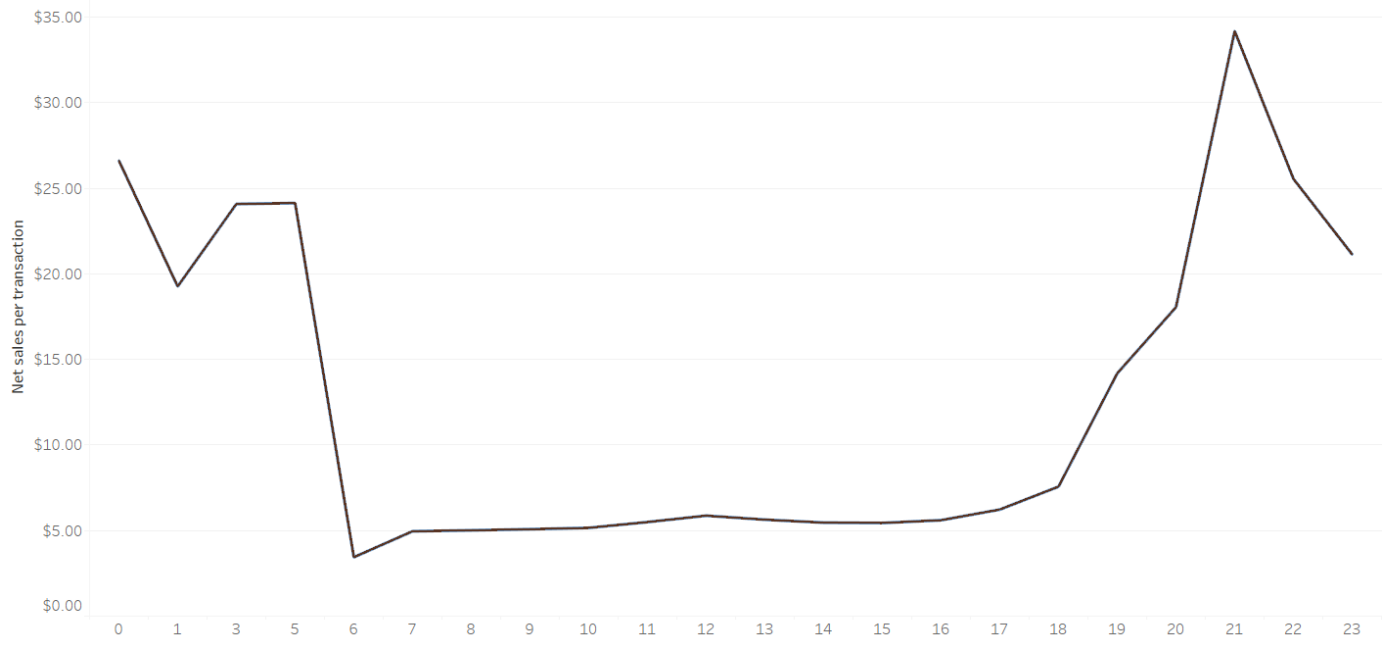


There is no obvious seasonality when we compare the three stores:

- **Chelsea** sells better on sunny days from March to August then it drops. December is probably a good month due to Christmas.
- **FiDi** starts well but stays steady until the end of the year.
- **Forest Hills** is the opposite of **Chelsea**. They do not sell well on sunny days.



If we focus on hours in a day, all the three stores sell a lot from 7am to 10am. There is one pic at **Chelsea** around 12pm. Then everything decreases slowly towards 6pm.

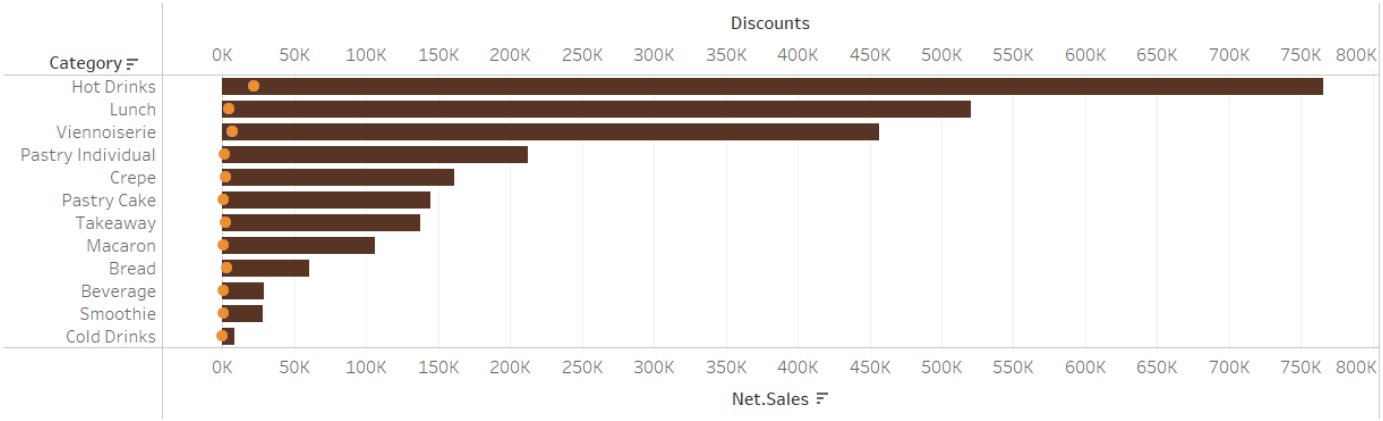


An interesting behavior of the company is highlighted. Some transactions are done outside regular opening hours of a store, i.e. from 19 pm to 6am. An employee is probably dedicated to prepare the orders and charge the transaction. The low net sale per transaction between 6am to 6pm is due to the high volume of transaction with low to medium net sale per transaction.

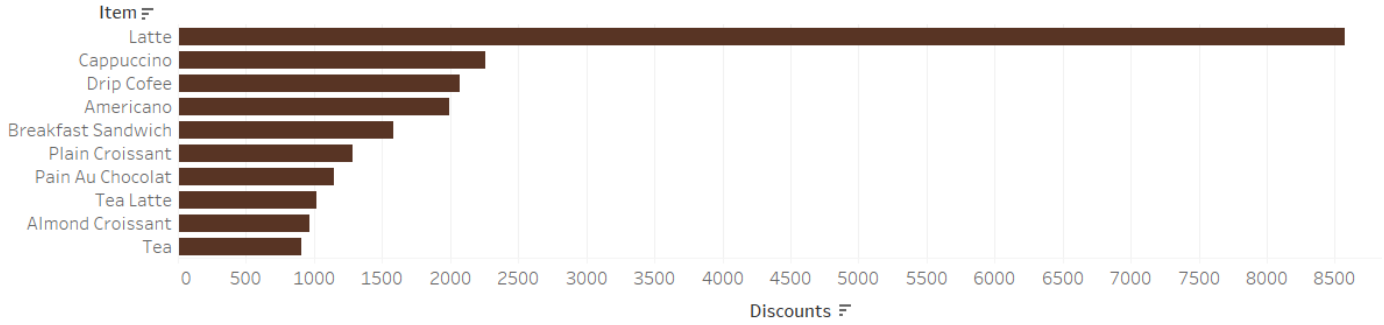
Instead of displaying each category in a graph, here is a table which summarizes the best sales time for each category of product.

Category	Best sales time
Beverage	9am to 11am
Bread	8am to 2pm
Cold Drinks	Around 3pm
Crepe	9am to 1pm
Hot Drinks	8am to 12pm
Lunch	Around 12pm
Macaron	Around 3pm
Cake	Around 4pm
Individual pastry	Around 3pm
Smoothie	1pm to 4pm
Takeaway	1pm to 4pm
Viennoiserie	8am to 11am

How is the client rewarded?

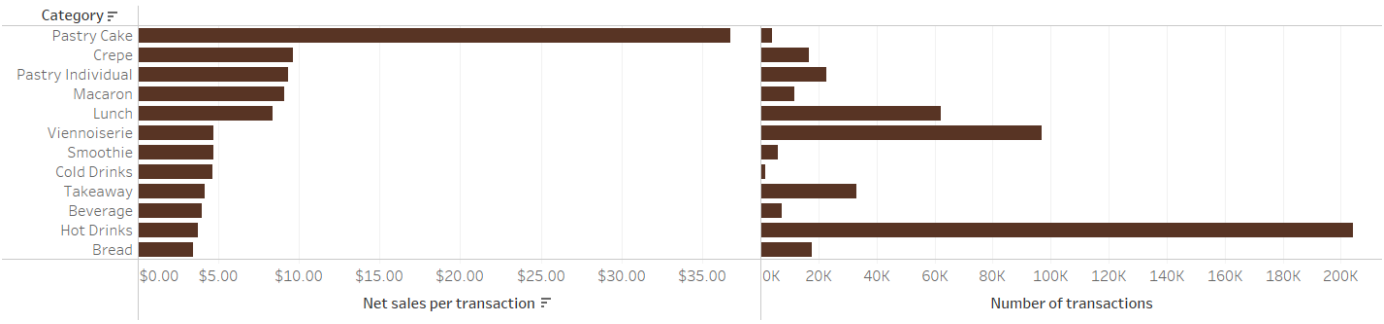


Discounts are mainly applied on Hot Drinks items. The axis on the top shows the applied discounts, compared to the net sales of each category, displayed along the bottom axis. If we change the scale, we will notice that some discounts are applied to the viennoiserie, the lunch and the bread categories.



Latte is indeed by far one the most discounted item in this company.

What are the most profitable items?



Three distinct groups appear:

- The cakes are the most profitable items, but they do not sell a lot.
- Crepe, individual pastries, macaron and lunch categories have a significant profitability, around $9 per transaction
- The last group has an average net sales around $5 per transaction.

Act

The analysis shows how consumers shop at the company's 3 stores.

The products sold in **Forest Hills** are more profitable. This is due to a greater sale of cakes. Although the number of transactions in this store is quite correct, it is not the highest but nevertheless close to the maximum. This store should continue to sell cakes and also increase the number of transactions on less expensive products which could increase its net income significantly.

Chelsea is the most profitable store. Cakes, individual pastries, crepes, or any valuable items should be sold a little more in this store to increase profitability.

FiDi has enormous potential like **Chelsea** but it would be necessary to check whether its geographical location is relevant. It is located in a business district and its sales are very limited. It would take a huge effort to double the number of transactions and/or double the net income to catch up with the other 2 stores. The sales are even steady at the end of the year with no increase during end of the year celebrations.

We notice that the discounts are almost made on a handful of identified products. Seasonal offers, loyalty offers or events should be organized to retain customers and perhaps make them discover other products that they would not have purchased on their own.

Seasonality is not very promoted in this company. They should play on it. For example, it could offer ice creams and smoothies in summer, pumpkin pies in autumn, hot chocolate in winter… However, we do not notice that these products are very successful during their favorite period. Seasonal operations should be considered.