DATA PRE PROCESSING AND VISUALIZATION

PROFESSORA JOANA NEVES

UNIVERSIDADE NOVA DE LISBOA
INFORMATION MANAGEMENT SCHOOL
BACHELOR DEGREE IN DATA SCIENCE

GROUP M:

Francisco Batista | 20221847
Joel Mendes | 20221825
Lourenço Martins | 20222053
Vicente Miranda | 20221845

Index

The Mega Market project entailed analyzing customer transaction data to extract actionable insights. It involved **data cleaning**, constructing an Customer Enhanced-Signature Table (**ABT**), and visualizing data through **Power BI dashboards**. The data preprocessing phase addressed outliers and missing values, enhancing data reliability. The ABT phase involved crafting detailed customer profiles using a calculated RFM model, prioritizing customer engagement metrics over simple monetary value. The final phase involved creating Power BI dashboards, offering a clear view of customer trends and business performance.

Background

In the project background, we delve into advanced *SQL* and *SAS* functions not covered in regular coursework. The **'INTCK'** function was instrumental in calculating the **'Recency'** aspect of our RFM score, providing the means to measure the time between the most recent purchase and a set reference date. '**COALESCE'** allowed us to select the first non-null value from a list, ensuring completeness in our data analysis. This function is pivotal in time series analysis for quantifying time intervals critical to assessing customer purchase recency.

For determining customers' preferred payment type, especially in tie situations, we utilized the **'RETAIN'** function. The **'RETAIN'** function, helped in preserving the most frequent payment method across data iterations until a new dominant method emerged, ensuring the accuracy of our preferred payment method analysis. These functions were key to enhancing the analytical robustness of the project.

Introduction

In the increasingly competitive retail landscape, ***Mega Market*** stands at a critical juncture where customer service and satisfaction are pivotal for differentiation and growth. This project centers on leveraging Mega Market's extensive data resources, housed in its information systems, to gain a deeper understanding of customer behaviors and preferences.

Faced with a challenge of limited insights into business activities and customer purchasing patterns, Mega Market has engaged a specialized data science team, with a dedicated Data Preparation (DP) Team focused on data pre-processing. This team's primary role is to lay the groundwork for advanced analytics by conducting exploratory data analysis and developing an **Enhanced Customer-Signature Table** (ABT). The **ABT** will serve as the foundation for customer segmentation and descriptive analysis. The overarching aim is to equip ***Mega Market*** with enriched data insights, enabling more strategic business decisions, enhancing customer satisfaction, and ensuring improved business performance. This initiative marks a pivotal step in ***Mega Market's*** journey towards data-driven excellence in the retail sector.

Methodology:

The methodology for the ***Mega Market*** project was systematically executed in three key phases, utilizing specialized tools for each stage to ensure a thorough analysis and insightful outcomes.

- **Phase 1:** Data Preprocessing Initial efforts involved data preprocessing using **SAS Enterprise Miner**, focusing on outlier treatment, data cleaning, and missing value imputation to ensure data integrity and readiness for deeper analysis.

- **Phase 2:** Data Transformation and ABT Creation with **SAS Guide** The project then transitioned to SAS Guide for further data refinement. Here, SQL was employed to address data inconsistencies and to construct the **Enhanced Customer-Signature Table** (ABT), shaping the data into a structured format for analytical purposes.

- **Phase 3:** Data Visualization and BI with **Power BI** In the final phase, it was leveraged for data visualization, transforming the refined data into interactive dashboards. These visualizations provided key business insights, aiding ***Mega Market*** in strategic decision-making and market analysis.

This structured approach, from detailed data preparation to dynamic visual storytelling, was pivotal in unlocking actionable business insights for ***Mega Market***.

Original Dataset

The initial dataset provided by Mega Market consisted of individual product purchase records, each linked to a Product Category ID and a Transaction ID. This dataset was unique as it lacked a primary key, which is essential for efficient data management and analysis. This absence necessitated extensive data pre-treatment and pre-processing. Each transaction had to be carefully analyzed to establish connections and patterns, considering various dimensions such as customer behavior and product preferences. It has *99999* rows.

Variable	Description
transactionNo	Transaction ID
date	Transaction's date
Product Id	Product ID
Product category Id	Product category ID
Product category name	Name of the product category
Product Name	Name of the product
Unit Price	Product unit price
Quantity	Number of items bought
total_payed	Amount spent by the customer
customerNo	Customer ID
Nationality	Customer nationality
Gender	Customer gender (M/F/O)
Monthly Income	Customer monthly income
Age	Customer age
Kids	Customer have kids (1-yes;0-no)
Reviews	Customer let a review about the product (1-yes;0-no)
Payment	Customer payment type
Channel	Sales channel name

fig 1

Class Variable Summary Statistics

Class Variable Summary Statistics
(maximum 500 observations printed)

Data Role=TRAIN

Data Role	Variable Name	Role	Number of Levels	Missing	Mode	Mode Percentage	Mode2	Mode2 Percentage
TRAIN	Channel	INPUT	2	0	Store	91.72	Online	8.28
TRAIN	Gender	INPUT	3	0	M	61.95	F	36.61
TRAIN	Kids	INPUT	3	2266	1	75.15	0	22.59
TRAIN	Nationality	INPUT	31	0	United Kingdom	90.81	Germany	2.42
TRAIN	Payment	INPUT	3	0	Paypal	40.07	Credit Card	34.98
TRAIN	ProductName	INPUT	513	0	Alarm Clock Bakelike Red	1.06	Alarm Clock Bakelike Ivory	0.79
TRAIN	Product_Category_Name	INPUT	9	0	Miscellaneous	87.12	Decorative items	8.32
TRAIN	Reviews	INPUT	3	57701	.	57.70	0	21.26

fig 11

We can understand, by these statistics, several points:

- **'Reviews'** has **57.70%** of **missing values**, so it will be dropped;
- **~62%** of customers are **man**;
- Just **8%** of the original data was performed in ***Online*** Shopping;
- **90.81%** of the Customers are from **UK**;
- **'ProductName'** has several different values, as we can see by the Mode Percentage; over the overall data is just **~1%** (513 different values);
- **'Kids'** has 2266 missing values, that are going to be imputed later.

Interval Variable Summary Statistics

Interval Variable Summary Statistics
(maximum 500 observations printed)

Data Role=TRAIN

Variable	Role	Mean	Standard Deviation	Non Missing	Missing	Minimum	Median	Maximum	Skewness	Kurtosis
Age	INPUT	54.36123	15.11825	97733	2266	29	51	299	-0.09631	-0.28342
Monthly_Income	INPUT	2506.403	1443.184	97733	2266	900	3000	90000	15.90465	582.5151
Quantity	INPUT	3.0133	1.743847	99999	0	1	3	100	18.90988	1050.286
Total_payed	INPUT	22.91824	17.8784	99999	0	1	18	77	0.981472	0.184304
Unit_Price	INPUT	7.628366	4.291977	99999	0	1	7	15	0.155836	-1.1938

fig 12

We can understand, by these statistics several Points:

- The **MAX** value for **'Age'** is **299 years**, which is an error; It also has missing values.
- **'Monthly_Income'** has an very **high skewness**, which indicates several outliers; It also has missing values;
- **'Quantity'** has a median of '3' and a maximum of '100' which explains the high skewness;
- '**Total_Payed'** skewness has not a relationship with **'Quantity'** skewness.

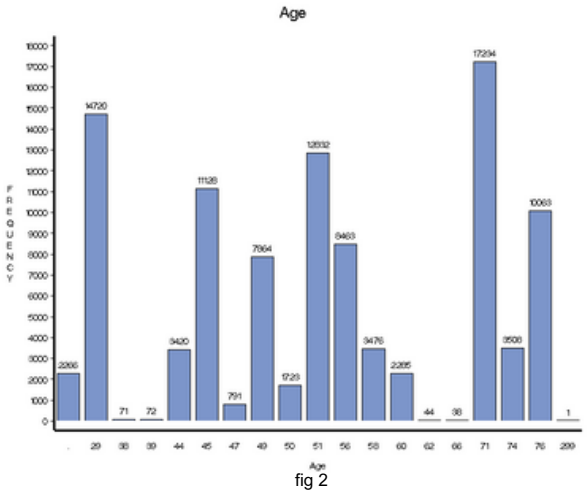


fig 2

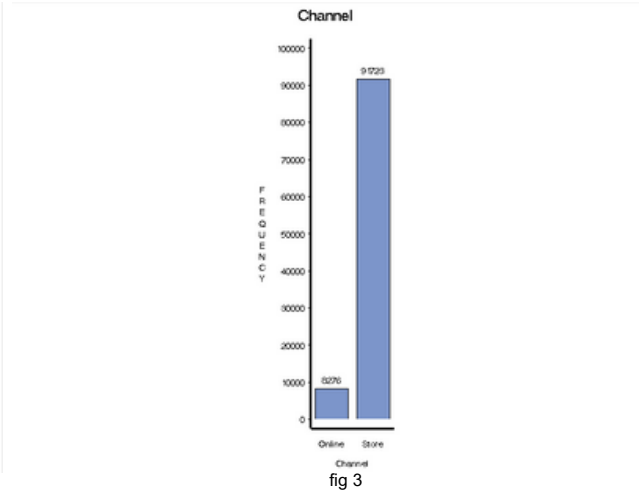


fig 3

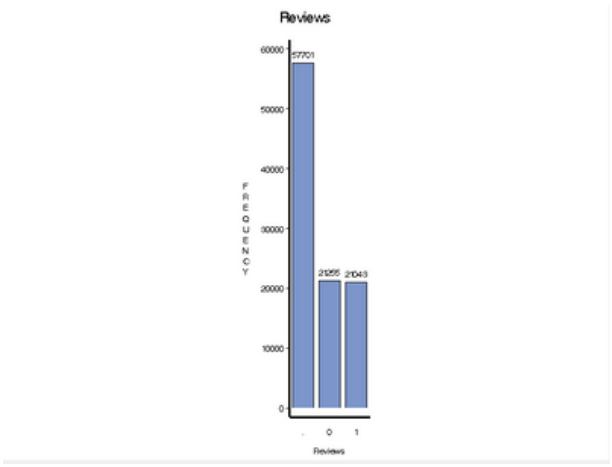
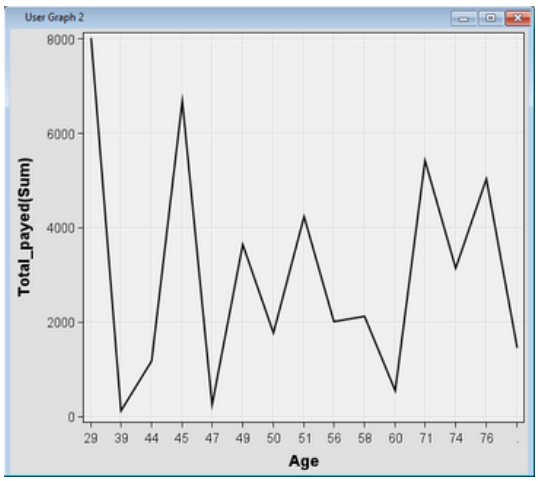


fig 5



We can´t really take conclusions with the untreated data
fig 6

Phase 1: *Data Cleaning in SAS MINER*

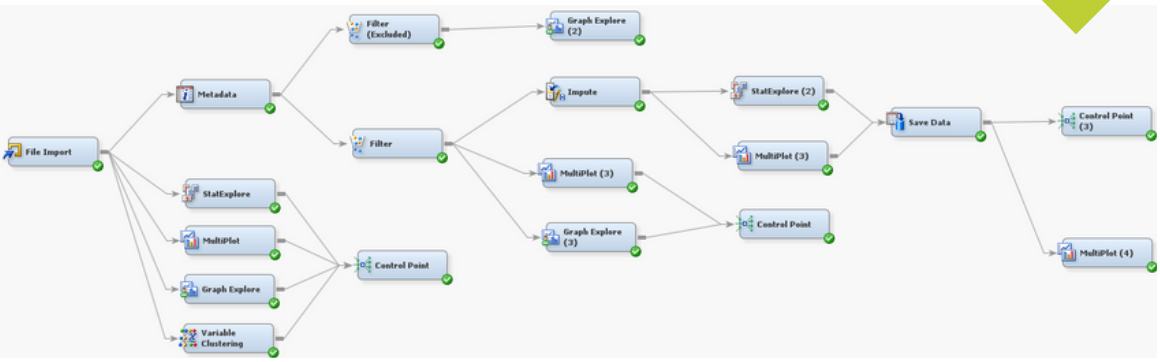


fig 7

Outlier Treatment:

To address the presence of outliers and enhance the dataset's statistical reliability, filters were applied to the interval variables. This approach was essential to mitigate issues related to skewness and unrealistic values, such as implausibly high ages (e.g., 299 years) and extreme income levels akin to the '**Bill Gates effect'**. The filtering process involved setting specific ranges for key variables, thereby ensuring more realistic and statistically meaningful data. The defined filter ranges were:

1. **Age:** Limited between 17 and 85 years, to exclude unrealistic age records and focus on the most relevant consumer age group.
2. **Monthly Income:** Capped at €5,460.61, addressing extreme income outliers that could skew the income distribution and affect the analysis.
3. **Quantity:** Set to a default maximum of 6, ensuring the dataset reflects typical purchase quantities and excludes abnormal bulk transactions.
4. **Unit Price:** Ranged from €0 to €15.5 as a default, which excludes extreme pricing outliers, ensuring a focus on the most common price range for products.

This filtering approach not only enhanced the quality of the dataset but also ensured that subsequent analyses, such as consumer behavior patterns and spending trends, were grounded in realistic and representative data. It was a crucial step in preparing the dataset for deeper analytical explorations, allowing for more accurate and scientifically sound insights.

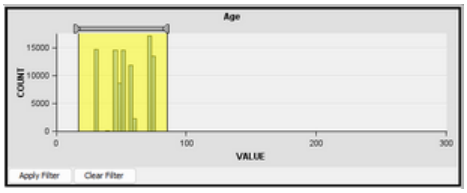


fig 8

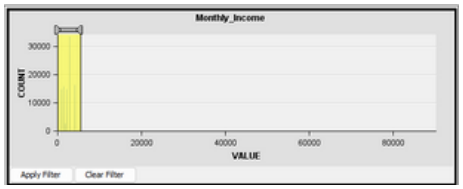


fig 10

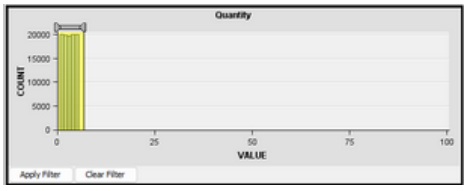


fig 9

Missing Values Imputation:

In addressing missing values in the dataset:

1. Categorical Trees for **'Kids'**: This method forecasts missing categories by analyzing data patterns, ensuring logical imputation.
2. Regression Trees for **'Age'** and **'Monthly_Income'**: This technique, ideal for continuous data, segments the dataset and fits a model to each subset, efficiently predicting missing values and enhancing data reliability.

These imputation methods are not only instrumental in maintaining the integrity of the dataset but also crucial for ensuring that subsequent analyses are based on complete and reliable data.

After PHASE 1 outputs:

Class Variable Summary Statistics
(maximum 500 observations printed)

Data Role=TRAIN

Data Role	Variable Name	Role	Number of Levels	Missing	Mode	Mode Percentage	Mode2	Mode2 Percentage
TRAIN	Channel	INPUT	2	0	Store	91.73	Online	8.27
TRAIN	Gender	INPUT	3	0	M	61.95	F	36.62
TRAIN	IMP_Kids	INPUT	2	0	1	77.42	0	22.58
TRAIN	Nationality	INPUT	31	0	United Kingdom	90.81	Germany	2.42
TRAIN	Payment	INPUT	3	0	Paypal	40.08	Credit Card	34.97
TRAIN	ProductName	INPUT	513	0	Alarm Clock Bakelike Red	1.06	Alarm Clock Bakelike Ivory	0.79
TRAIN	Product_Category_Name	INPUT	9	0	Miscellaneous	87.11	Decorative items	8.32

Interval Variable Summary Statistics
(maximum 500 observations printed)

Data Role=TRAIN

Variable	Role	Mean	Standard Deviation	Non Missing	Missing	Minimum	Median	Maximum	Skewness	Kurtosis
IMP_Age	INPUT	54.61781	15.02377	99949	0	29	51	76	-0.17959	-0.96213
IMP_Monthly_Income	INPUT	2520.677	1080.46	99949	0	900	3000	4200	0.065507	-1.0562
Quantity	INPUT	3.002571	1.416477	99949	0	1	3	5	-0.00433	-1.3055
Total_payed	INPUT	22.91844	17.87856	99949	0	1	18	77	0.98141	0.184254
Unit_Price	INPUT	7.628661	4.292189	99949	0	1	7	15	0.155612	-1.19397

fig 13

After implementing outlier treatment and imputation methods, the dataset's skewness and missing value issues were effectively resolved. Filters applied to interval variables eliminated unrealistic data, helping normalize the distribution. The use of categorical and regression trees for imputation accurately filled missing values in both categorical ('Kids') and continuous ('Age', 'Monthly_Income') variables, maintaining the dataset's integrity for further analysis.

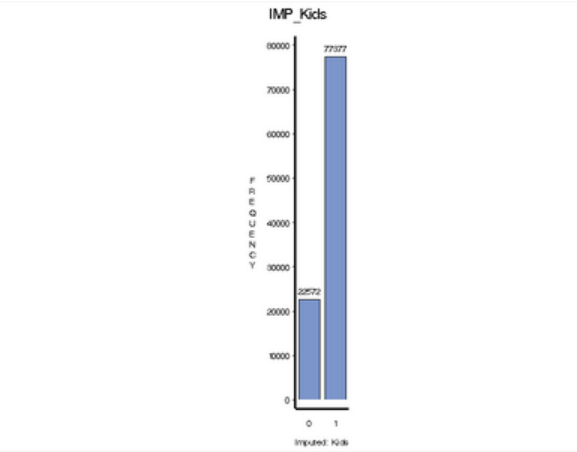


fig 14

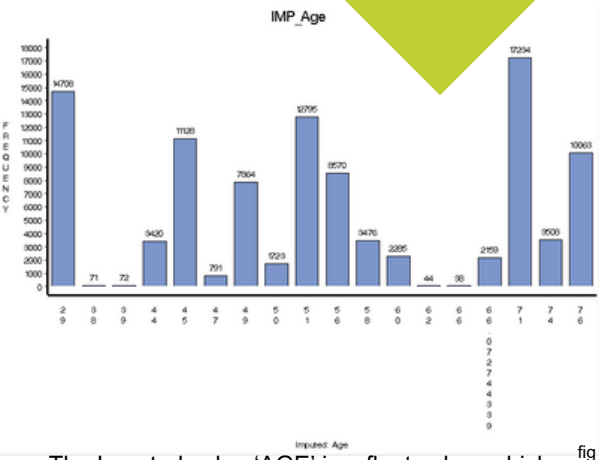


fig 15

The Imputed value 'AGE' is a float value, which will be rounded at the PHASE 2

Quantity before and after the PHASE 1 treatment:

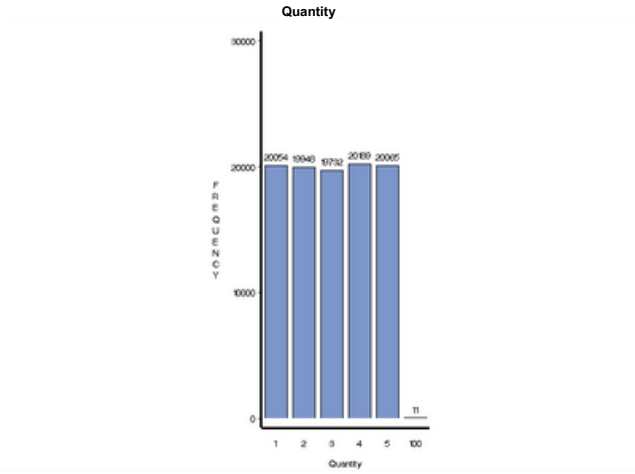


fig 17

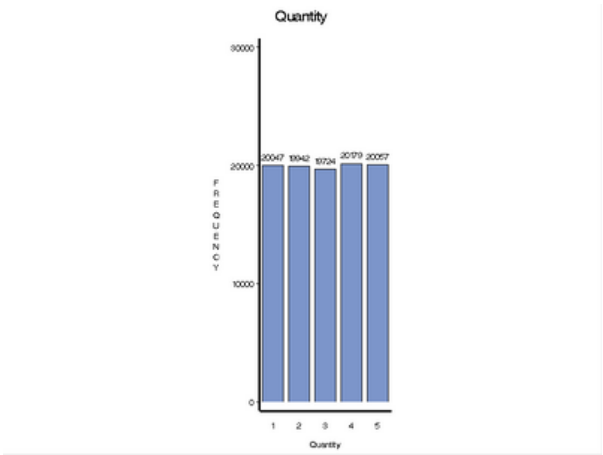


fig 16

Monthly income before and after the PHASE 1 treatment:

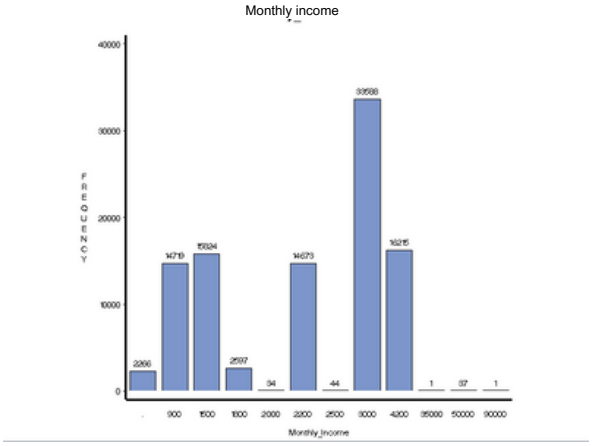


fig 18

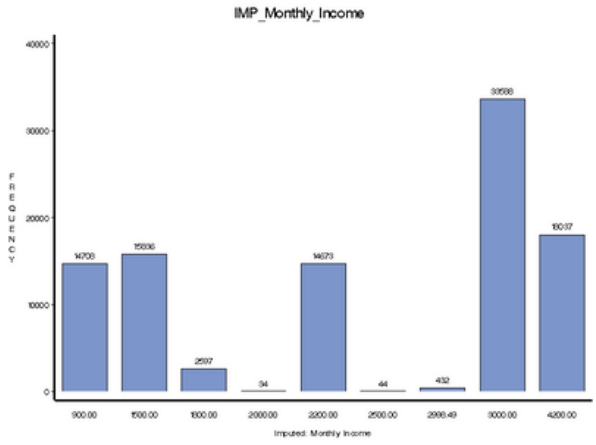


fig 19

Phase 2: Incoherence Treatment and ABT *in SAS GUIDE*

In **Phase 2** of the project, we initiated by rounding **imputed** values such as **'Age'** and **'Monthly Income'** to ensure data consistency. Following this, we addressed various incoherences within the dataset. Some of these inconsistencies were rectified directly within the primary dataset, while others required the creation of auxiliary tables for a more detailed examination and correction of incoherences. This process was essential to cleanse and prepare the data for the construction of the Enhanced Customer-Signature Table (ABT), setting a solid foundation for accurate and reliable data analysis.

Incoherence Treatment:

1. **Paying Cash in Online Shopping:** Transactions indicating cash payments for online purchases are removed, as they contradict the nature of online transactions.
2. **Kids or No Kids?:** Inconsistencies in the 'Kids' field are addressed. Records where 'Kids' are neither 0 nor 1 (indicating the presence or absence of kids) are excluded.
3. **Invalid Gender:** Records with genders other than 'F', 'M', or 'O' are considered inaccurate and removed.
4. **Paying Cash Online:** Similar to the first point, this incoherence targets records where the 'Channel' is online but the 'Payment' method is cash, which is logically inconsistent.
5. **Invalid Product Category:** This incoherence filters out records with product categories that don't match the predefined list of valid categories.
6. Invalid Channel: Records with channels other than 'Online' or 'Store' are removed for data accuracy.
7. **Invalid Payment Type:** Transactions that do not have 'Cash', 'Credit Card', or 'Paypal' as the payment method are considered incoherent and excluded.
8. **Invalid Nationality:** Entries with 'Unspecified' nationality are removed for lacking specific demographic information.
9. **Invalid Unit's Price:** Transactions with a unit price less than 0 are considered invalid and removed.
10. **Inconsistency in Total Payment Calculation:** Transactions where the total paid does not equal the product of unit price and quantity are considered erroneous and are corrected.
11. I**nvalid Quantity:** Records with quantities less than 1 are removed as they are not logical for a completed transaction.
12. **Invalid Monthly Income:** Entries with a monthly income less than 0 are excluded for being unrealistic.
13. **Invalid Age:** Age values outside the range of 16 to 99 years are considered invalid and removed.
14. **Invalid Date:** Transactions dated beyond the year 2019 are excluded as they are outside the study period.
15. **More Than One Gender and Nationality:** Customers with more than one gender or nationality in the records are removed to maintain demographic consistency.
16. **Transaction Made by Multiple Customers:** Transactions linked to multiple customers are removed for data integrity.
17. **Multiple Dates in One Transaction:** Transactions with more than one date are excluded for logical consistency.
18. **Multiple Channels in One Transaction:** Transactions recorded in multiple sales channels are removed for accuracy.
19. **Multiple Payment Types in One Transaction:** Transactions showing multiple payment methods are considered incoherent and excluded.

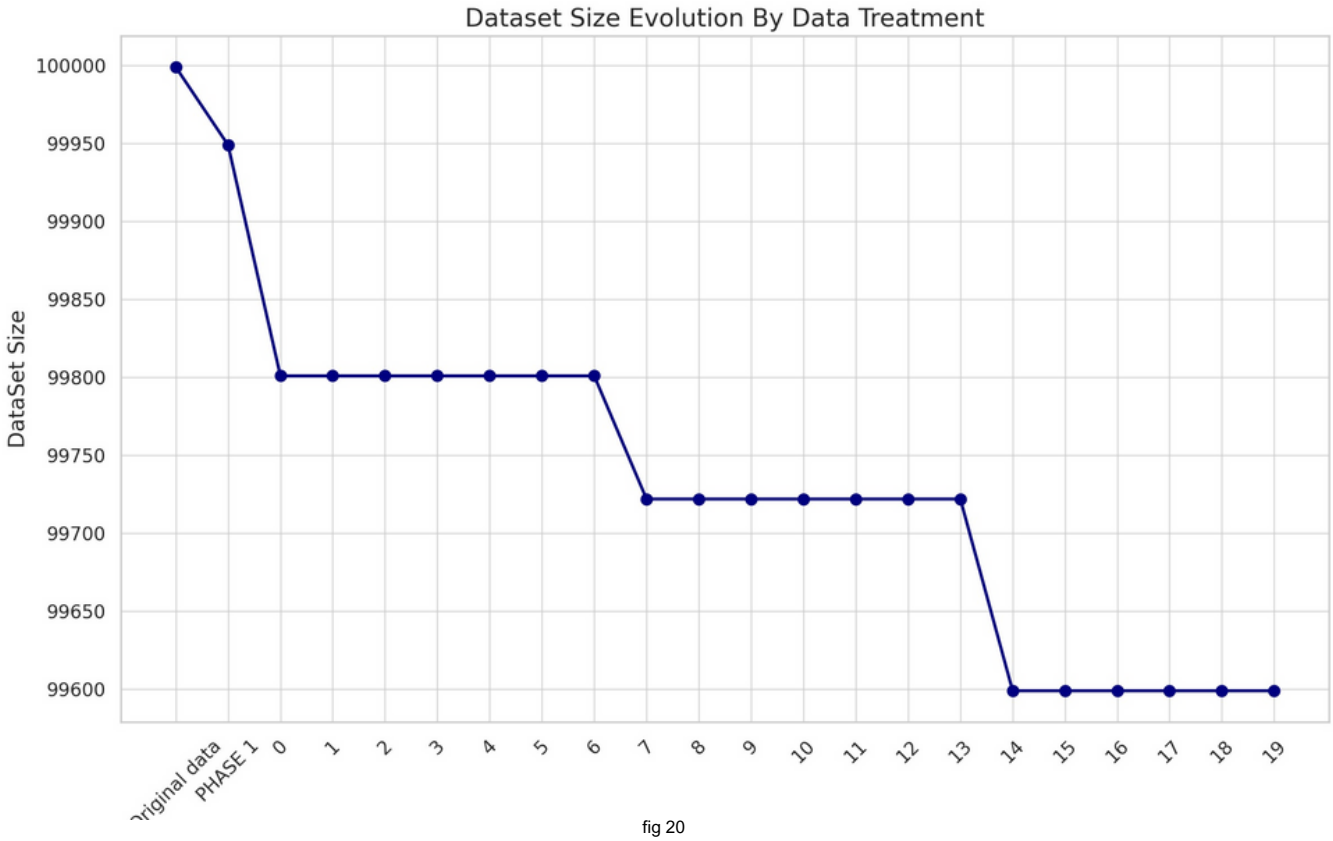


fig 20

In the 10th incoherence we did not deleted the incoherences, we solved them by replacing by the real value.

```
/* Incoherence NR 10
- Inconsistency in Total Payment Calculation */
if Total_payed ne (Unit_Price * Quantity) then do;
    Total_Payed = Unit_Price * Quantity;
end;
```

Which affected ~500 rows

fig 21

Customer 12414 - a curious case:

Customer 12414 is a curious case: he had **2 different monthly incomes in the dataset**. In the original dataset **C12414** had transaction records with monthly income recorded and others with monthly income NOT recorded. So the 2nd different value was **imputed** by SAS Miner at the Imputation Phase.

If this was the only issue, C12414 would not be deleted, but as he also inserted 2 genders we assumed it was an error. So he/she/it was **manually deleted**.

```
/*deleting customers' incoherences
16584: inserted 2 genders
13841: inserted 2 genders
12414: inserted 2 genders
12414 has 2 monthly incomes. One is 1500 euros, another one is 2998 euros (rounded)
this second value was imputed by SAS MINER. If this was the only issue, he/she/it
would be not eliminated, but has he/she/it has 2 genders in the database, Customer 12441 will be eliminated*/
data WORK.TABLE2;
    set WORK.TABLE2;
    if CustomerNo in (16584, 13841, 12414) then delete;
run;
```

fig 22

New Variables:

Primary Key	Customer ID
Demographics	Age
	Kids
	Nationality
RFM	DaysSinceLastPurchase
	Freq
	Monetary
	RecencyRank
	FrequencyRank
	MonetaryRank
	RFMScore
	Cluster
DateStats	CustomerSince
	LastPurchase
	DurationCustomer
StoreStats	FreqStore
	MonetaryStore
	MonetaryOnline
	FreqOnline
PaymentStats	PaypalFrequency
	CashFrequency
	CreditCardFrequency
	FavoritePayment
	PaypalMonetary
	CashMonetary
	CreditCardMonetary
CategoryStats	Monetary_+'CategoryName'
	FavoriteCategory
Average	
	AvgSpent
CustomerBehavior	NrCategories

fig 23

ABT CONSTRUCTION:

The construction of the Enhanced Customer-Signature Table (ABT) in Phase 2 of the project entailed a meticulous process of data transformation and enhancement, leveraging the capabilities of SAS Guide and SQL.

The ABT construction began with the rounding of imputed values such as 'Age' and 'Monthly_Income', ensuring more realistic and consistent data. This was followed by addressing various data incoherencies, some of which required straightforward fixes, while others necessitated the creation of auxiliary tables for a detailed examination of the inconsistencies.

Key steps in the ABT construction included:

- Cleaning and Preparing Data: Removal of anomalies and rectification of data inconsistencies to ensure data integrity and reliability.
- Enhancing Data Quality: Application of various techniques like imputation and outlier treatment to refine the dataset for more accurate analysis.
- Variable Transformation: Conversion and renaming of variables to create a more understandable and analytically useful dataset.
- Creation of New Variables: Introduction of new variables to capture deeper insights and facilitate a more comprehensive analysis.
- SQL Queries and Joins: Extensive use of SQL for data manipulation and to join multiple tables, enriching the dataset with a wide range of information.
- Final Compilation: The culmination of these efforts resulted in a comprehensive ABT, serving as a foundation for subsequent phase.
- The ABT thus created was instrumental in enabling a more nuanced understanding of customer behavior, product performance, and overall business dynamics, significantly contributing to the project's success.

Demographic:

In the initial phase of our analysis, we concentrated on developing demographic variables, including age, gender, and nationality. This approach aimed to understand our customer base better and facilitate targeted marketing strategies and product alignment with diverse customer needs. This demographic profiling was crucial for enhancing customer insights and informed further analytical steps.

RFM Score:

In the pursuit of a more nuanced and comprehensive customer segmentation, we recognized that relying solely on monetary value to rank and understand customers was insufficient. To address this, we devised a more holistic approach through the creation of an **RFM** *(Recency, Frequency, Monetary)* model. This model allowed us to categorize customers not just based on their spending but also considering how recently and how frequently they engaged with the business.

The RFM model provides a multi-dimensional view of the customer base, acknowledging that:

- **Recency (R):** The time since a customer's last purchase is a strong indicator of their engagement level. More recent interactions suggest a higher likelihood of future engagement.

- **Frequency (F):** The number of transactions a customer has over a period demonstrates their loyalty and habitual engagement with the brand.

- **Monetary (M):** While still important, the monetary value is considered in conjunction with recency and frequency, providing a balanced view of a customer's value to the business.

By segmenting customers based on these three dimensions, we were able to create more targeted and effective customer engagement strategies. This RFM-based segmentation is instrumental in identifying high-value customers, developing retention strategies, and tailoring marketing efforts to different customer segments, ultimately leading to a more strategic and informed approach in customer relationship management.

1. Calculating absolute R, F, M values
2. Attributing Ranks for each one:
    a. Frequency:
        i. 1: [1,10[
        ii. 2: [10, 20[
        iii. 3: >= 20
    b. Recency:
        i. 3: <=73
        ii. 2:]73, 110[
        iii. 1: else
    c. Monetary:
        i. 1 <1000
        ii. 2 [1000,3500[
        iii. 3 >= 3500
3. Combining RFM ranks, like: 131, 312, 213
4. Clusters created summing -> RFM: 331 = 3+3+1 = 7 SCORE = GOLD

The RFM-based customer clusters significantly aid the marketing and BI teams. By segmenting customers, marketing can create targeted campaigns, boosting engagement and conversions. Simultaneously, the BI team gains insights into customer preferences, guiding strategic decisions and trend predictions. This approach leads to more effective strategies and a better understanding of customer dynamics.

Type of Clients:

BRONZE	RFM SUM = 3 (MIN)
SILVER	RFM SUM = 4 OR 5
Gold	RFM SUM = 6 OR 7
Platina	RFM SUM = 8
Prestige	RFM SUM = 9 (MAX)

{There is no prestige clients at the moment)

fig 24 -> example of a
Customer Prestige Plan Card

DateStats:

Specifically focusing on each customer's first and last purchase dates. This data allowed the calculation of the **'Duration Customer'** variable, representing the total period a customer has been shopping with us. These insights are instrumental in understanding customer loyalty and shopping patterns.

StoreStats:

Calculated **'Freq Online'** and **'Freq Store'** to quantify the number of purchases made online and in-store, respectively. Additionally, **'Monetary Online'** and **'Monetary Store'** provided insights into the total spending in each channel.

PaymentStats:

Calculating frequencies and monetary values associated with PayPal, cash, and credit card transactions, we gained insights into the popularity and financial impact of each method.

Important Note: *FreqPaypal + FreqCreditCard + FreqCash* != *freq* (RFM)

'Freq' was calculated by distinct and unique dates, freq'payment' was not, because sometimes the same bill ,at the same day, by the same customer was paid in different payment methods.

CategoryStats:

The category statistics section of the analysis focused on evaluating customer spending across various product categories. This analysis calculated the total expenditure per customer in distinct categories such as *Beauty & Accessories*, *Kitchenware*, and *Decorative Items*. This information is crucial for understanding consumer preferences, aiding in inventory management, and facilitating targeted marketing strategies. Additionally, it supports the development of new product lines aligned with prevailing customer interests and spending behaviors.

Was also calculated the number of categories each customer purchased.

AverageStats:

The average statistics in the project focus on two key aspects: AvgSpent and '**AvgDaysBetweenPurchases'**. **'AvgSpent'** represents the average amount spent by customers per transaction, providing insights into spending habits and financial engagement. **'AvgDaysBetweenPurchases'**, on the other hand, measures the average interval between transactions, offering a perspective on customer visit frequency.

Phase 3: Data Analysis in PowerBI

In the third phase of the project, both the refined original data and the enhanced customer-signature table (ABT) were employed, utilizing a **one-to-many relationship**. The integration of these datasets offered a layered understanding of customer behaviors and purchasing trends, crucial for generating precise visualizations and insights. This comprehensive data synthesis was pivotal for strategic decision-making and developing targeted business strategies.

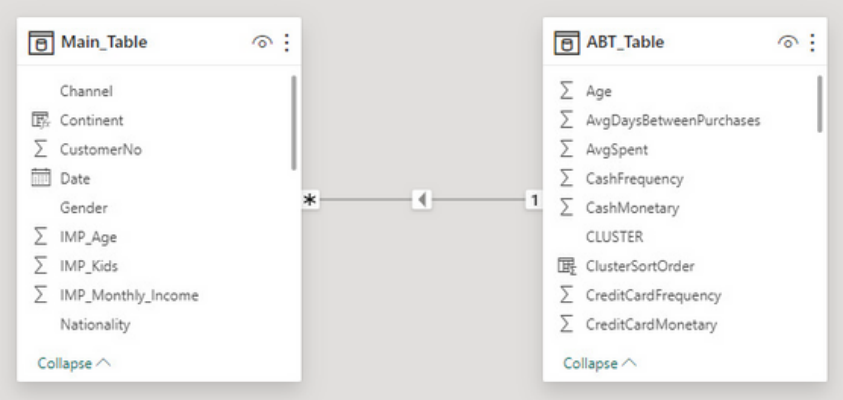


fig 25

Financial Overview:

During the second half of 2019, ***Mega Market*** observed a significant trend in its daily total revenue. Starting from **August 11th**, there was a noticeable upward trajectory, with revenue doubling by the end of November. This period saw a remarkable peak of **€42,334** on **November 20th, 2019**. However, as the year drew to a close, revenue experienced a decline, falling back to around **€7,000**, close to the initial values observed at the start of the observation period.

Interestingly, the analysis did not reveal any distinct purchasing trends associated with seasonal sales events, holidays, or festive seasons such as summer breaks or Thanks Giving Day. The lack of spikes during these traditionally high-spending periods suggests that customer purchasing behavior at Mega Market is not heavily influenced by these events.

Despite the end-of-year dip, forecasts indicate a potential recovery in revenue. This projection is based on current data trends and market analysis, suggesting a positive outlook for ***Mega Market*** in the upcoming periods.

Total revenue in this period is **2M€**.

PayPal is not used in *Online* Shopping, just in *Stores*. In online Shopping the only payment method used is CreditCard.

Regarding the *sales channels* at **Mega Market**, there is a notable disparity in revenue generation. *Physical stores* account for a significant majority, contributing **92%** to the total revenue. In contrast, online sales constitute only **8%**, highlighting a strong customer preference for in-store shopping experiences.

When it comes to payment methods, *PayPal* emerges as the most popular option, being used in **40%** of transactions. This preference is closely followed by *credit card* payments, which account for **35%** of sales. *Cash* transactions, while still substantial, represent **25%** of the total, indicating a lesser but still relevant preference for traditional payment methods.

These insights into channel preferences and payment methods can be instrumental for Mega Market in strategizing their sales and marketing efforts, optimizing both online and offline customer experiences, and tailoring payment options to suit customer preferences.

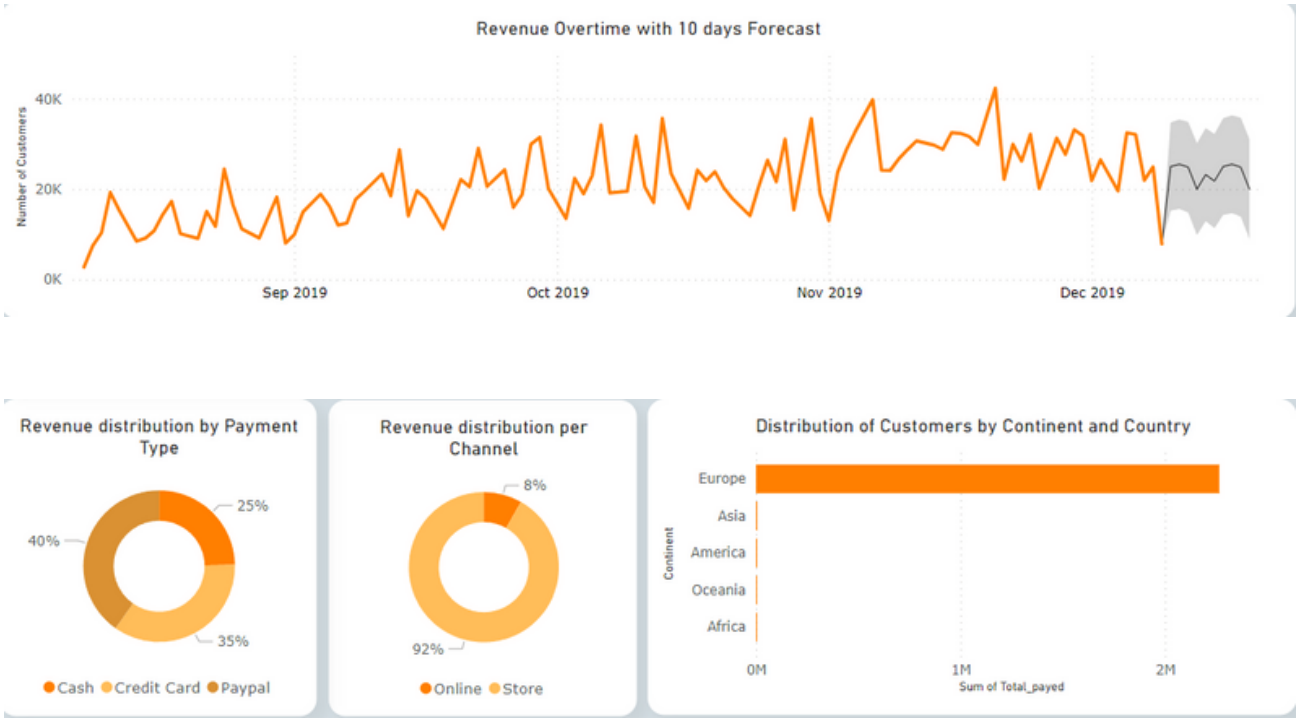


fig 4

Customer Analysis:

In the customer analysis segment of Mega Market's retail analytics, distinct patterns emerge across different customer segments based on the RFM (Recency, Frequency, Monetary) model.

Customer Segment Revenue Contribution: The **'Gold'** segment contributes the most to Mega Market's revenue, while the **'Bronze'** segment contributes the least. This indicates that customers who shop more frequently and spend more ('Gold') are the most valuable to the company. In contrast, 'Silver' customers, although the most numerous with **1789** individuals, contribute less on a per-customer basis. The 'Prestige' category has no customers, and 'Platina' has a minimal presence with only **2** customers.

Geographical Distribution: A significant **90%** of ***Mega Market***'s customers are from the **UK**, with the remaining customers spread across various continents. This suggests a strong local market presence but also indicates potential for international expansion.

Income and Age Demographics: The analysis shows that customers with lower incomes predominantly fall into the **'Silver'** category. Interestingly, younger customers (around 29 years old), with an average monthly income of approximately **900€**, tend to be in the 'Silver' or 'Gold' segments but rarely in 'Bronze'. They are mainly from Canada, Malta, the UK, and EIRE. Middle-aged customers, despite earning about four times more, exhibit similar shopping trends. The older demographic, with a retirement income of around **3000€**, has a larger representation in the 'Gold' segment and is exclusively **male**.

In ***Mega Market***'s customer base, there is a distinct gender distribution which also correlates with their **RFM** segments and demographics:

1. Gender Distribution: The customer base comprises **63%** men, **35%** women, and **2%** identifying as 'other'. This distribution suggests a male-dominated customer demographic;
2. RFM Segmentation by Gender: Both men and women predominantly fall into the 'Silver' and 'Gold' RFM segments. Notably, men constitute over half of the 'Gold' segment customers, indicating their significant contribution to Mega Market's revenue;
3. Income and Age: The analysis reveals that men generally have a higher monthly income and are older compared to other gender groups. This could imply that older male customers with higher disposable incomes are more likely to be frequent and high-spending patrons ('Gold' segment);
4. 'Other' Gender Category: Customers who identify as **'other'** are, on average, the **youngest group** and have the **lowest income**.

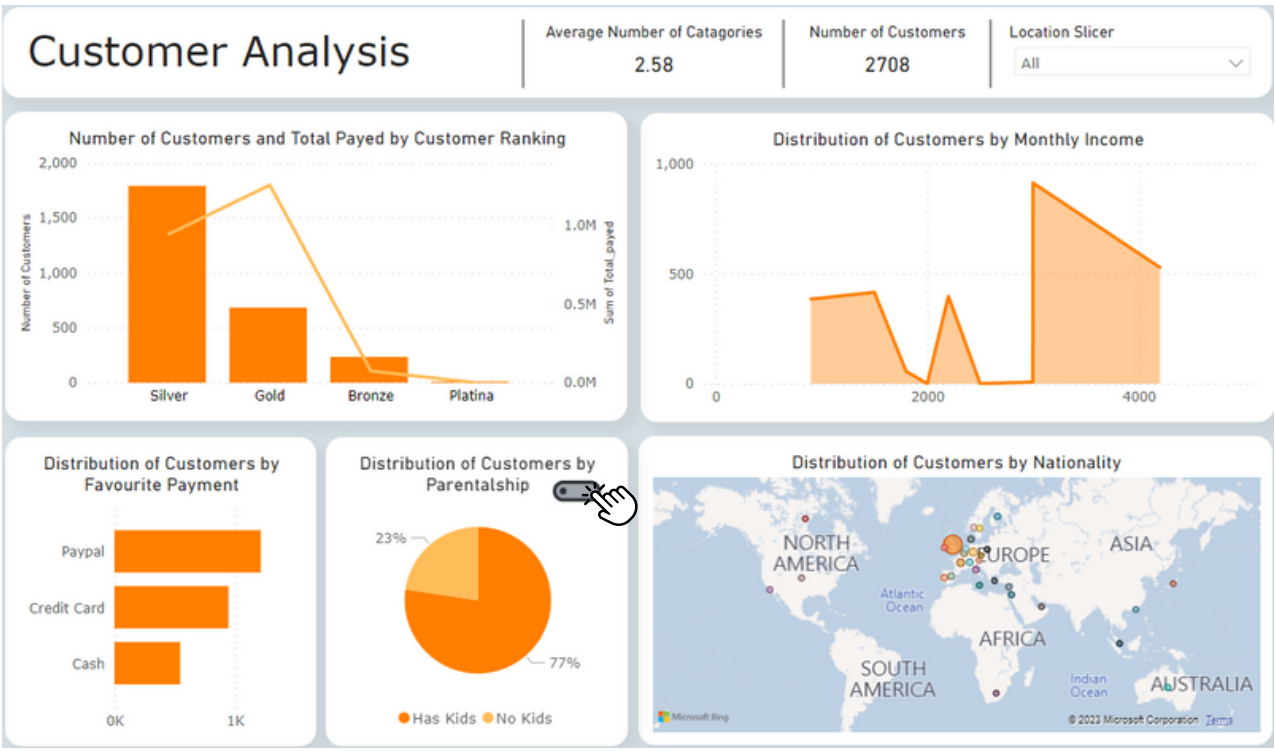


fig 26

*this is a toggle

Product Analysis:

1. Sombrero Category: Although the **'Sombrero'** category records the **least number of purchases**, it stands out as the **most expensive** category on average. This suggests that while these items are not frequently bought, they contribute significantly to revenue per sale.
2. Most Popular Category: The **'Miscellaneous'** category dominates in terms of purchase frequency, holding a substantial share of **87.11%** of all purchases. This high percentage indicates that customers favor this category for its variety or utility.
3. Sales Pattern Across Categories: Interestingly, sales in all categories show fluctuations with peaks and troughs occurring every 3 to 4 days. Despite these variations, purchase patterns remain relatively consistent, indicating steady customer interest.
4. Top-Selling and Highest Revenue Products: The 'Paper Chain Kit 50'S Christmas' emerges as the most sold product, indicating its popularity among customers. In terms of revenue generation, the 'Hot Water Bottle Keep Calm' leads, contributing significantly to ***Mega Market***'s earnings despite not being the top-selling item.

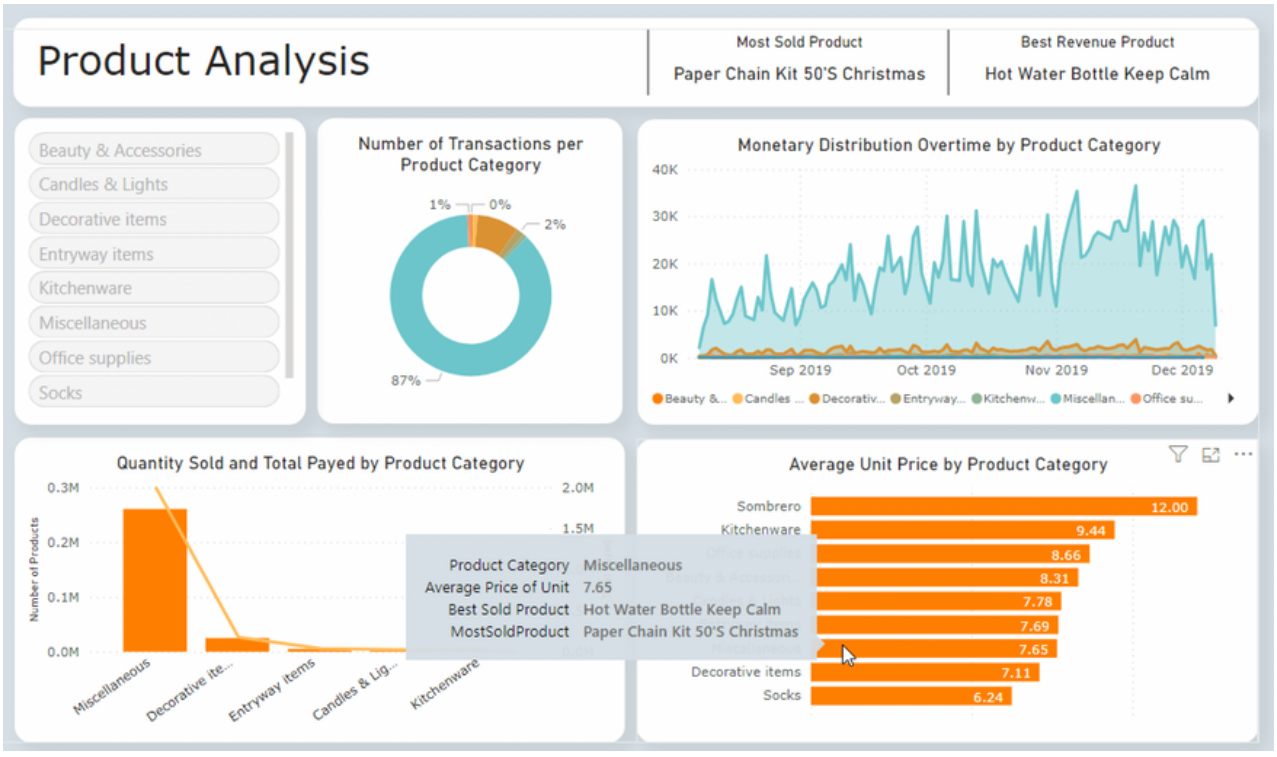


fig 27

Conclusion

The ***Mega Market*** project, spanning data preprocessing, ABT construction, and Power BI dashboard creation, marked a significant advancement in retail analytics. Initially, the team focused on data integrity, addressing outliers and imputing missing values to stabilize the dataset. In phase two, the raw data was transformed into an enriched ABT, with the application of an RFM model for customer segmentation. This model provided a deeper understanding of customer value beyond mere monetary metrics. The final phase involved creating interactive Power BI dashboards, offering dynamic insights into customer demographics, purchasing patterns, and sales trends. This comprehensive approach has poised Mega Market to make data-driven decisions, enhance customer engagement, and maintain a competitive edge in the retail sector.

In future endeavors, the Customer Enhanced-Signature Table (ABT) developed during this project holds potential as a foundational dataset for constructing predictive machine learning models. These models can further enhance the strategic decision-making process by forecasting trends and customer behaviors.

Softwares Used

- SAS Enterprise Miner Client 15.2
- SAS Studio 2023
- Canva App for Mac 1.79.0
- Microsoft Power BI
- Parallels Desktop for Mac
- Microsoft Office Excel 2024

Appendix:

```
/*rounding imputed monthly income values*/

data table1;
set table;
IMP_Monthly_Income = round(IMP_Monthly_Income);
IMP_Age = round(IMP_Age);
run;
```

```
/* droping _WARN_*/
drop _WARN_;

/* Incoherence NR 1
- paying cash in online shopping */
if (Payment = 'Cash' and Channel = 'Online') then do;
delete;
end;

/* Incoherence NR 2
- kids or no kids? */
if (IMP_Kids ne 0 and IMP_Kids ne 1) then do;
delete;
end;

/* Incoherence NR 3
- invalid gender */
if (Gender ne 'F' and Gender ne 'M' and Gender ne 'O') then do;
delete;
end;

/*Incoherence NR 4
- paying cash online */
if (Channel = 'Online' and 'Payment' = 'Online') then do;
delete;
end;
```

```
/* Incoherence NR 5
-invalid product category*/
if (Product_Category_Name ne 'Beauty & Accessories' and Product_Category_Name ne 'Candles & Lights'
and Product_Category_Name ne 'Decorative items' and Product_Category_Name ne 'Entryway items' and
Product_Category_Name ne 'Kitchenware' and Product_Category_Name ne 'Miscellaneous'  and
Product_Category_Name ne 'Office supplies' and Product_Category_Name ne 'Socks' and
Product_Category_Name ne 'Sombrero') then do;
delete;
end;

/*Incoherence NR 6
- invalid Channel*/
if (Channel ne 'Online' and Channel ne 'Store') then do;
delete;
end;


/*Incoherence NR 7
-invalid Payment Type*/
if (Payment ne 'Cash' and Payment ne 'Credit Card' and Payment ne 'Paypal') then do;
delete;
end;

/*Incoherence NR 8
-invalid nationality */
if (Nationality = 'Unspecified') then do;
delete;
end;
```

```
/* Incoherence NR 9
-invalid Unit's Price*/
IF (Unit_Price <0) then do;
delete;
end;

/* Incoherence NR 10
- Inconsistency in Total Payment Calculation */
if Total_payed ne (Unit_Price * Quantity) then do;
    Total_Payed = Unit_Price * Quantity;
end;

/* Incoherence NR 11
- invalid quantity*/
if (Quantity < 1) then do;
delete;
end;



/* Incoherence NR 12
-Invalid Monthly Income*/
if (IMP_Monthly_Income < 0) then do;
delete;
end;

/* Incoherence NR 13
-Invalid IMP_Age*/
if (IMP_Age <16 or IMP_Age > 99) then do;
delete;
end;

/* Incoherence NR 14
- Invalid Date*/
if (Year(Date) >= 2020) then do;
delete;
end;
```

```
/* Incoherence NR 15
-More than 1 gender and nationality*/

proc sql;
    create table WORK.CustomerSummary as
    select CustomerNo,
           count(distinct Nationality) as NumNationalities,
           count(distinct Gender) as NumGenders,
           count(distinct IMP_Kids) as Numkids,
           count(distinct IMP_Monthly_Income) as  MonthlyIncome
    from WORK.TABLE2
    group by CustomerNo;
quit;
proc sql;
    create table inc_transactions_id_customer as
    select CustomerNo as Customers, TransactionNo as Transactions, IMP_Monthly_Income
    from WORK.TABLE2
    where CustomerNo = 12414
    order by CustomerNo, TransactionNo;
run;


/*deleting customers' incoherences
16584: inserted 2 genders
13841: inserted 2 genders
12414: inserted 2 genders
12414 has 2 monthly incomes. One is 1500 euros, another one is 2998 euros (rounded)
this second value was imputed by SAS MINER. If this was the only issue, he/she/it
would be not eliminated, but has he/she/it has 2 genders in the database, Customer 12441 will be eliminated*/
data WORK.TABLE2;
    set WORK.TABLE2;
    if CustomerNo in (16584, 13841, 12414) then delete;
run;
```

```sql
/*Incoherence NR 16
-Transaction made by multiple customers*/
proc sql;
    create table transaction_id_cust_id as
    select TransactionNo, count(distinct(CustomerNO)) as diff_customer
    from WORK.TABLE2
    group by TransactionNO
    having count(distinct(CustomerNo)) ne 1;
run;

/*Incoherence NR 17
-Multiple dates in one transaction*/
proc sql;
    create table inc_transaction_diff_dates as
    select TransactionNo, count(distinct(Date)) as diff_transaction_date
    from WORK.TABLE2
    group by transactionNo
    having count(distinct(Date)) ne 1;
run;

/*Incoherence NR 18
-Multiple Channel in one transaction*/
proc sql;
    create table inc_transaction_channel as
    select TransactionNo, count(distinct(Channel)) as diff_channel_number
    from WORK.IMPORT
    group by TransactionNo
    having count(distinct(channel)) ne 1;
run;

/*Incoherence NR 19
-multiple payment types*/
proc sql;
    create table inc_transaction_payment as
    select TransactionNo, count(distinct(Payment)) as diff_payment_type_number
    from WORK.TABLE2
    group by TransactionNo
    having count(distinct(Payment)) ne 1;
run;
```

```sql
/*CREATING ABT*/


/*DEMOGRAPHIC INFO ABOUT CUSTOMERS
-GENDER
-AGE
-NATIONALITY*/

/*Gender*/
/*Creating a table 'Gender' that contains unique customer numbers and their corresponding gender, sorted by customer number and gender.*/
proc sql;
    create table gender as
    select distinct CustomerNo, Gender
    from WORK.TABLE2
    order by CustomerNo, Gender;
run;

/*Age*/
/*Creating a table 'Age' that contains unique customer numbers and their corresponding age, sorted by customer number.*/
proc sql;
    create table Age as
    select distinct CustomerNo, IMP_Age
    from WORK.TABLE2
    order by CustomerNo;
run;

/*Creating a table 'Kids' that contains unique customer numbers and a flag or number indicating kids, sorted by customer number and the kids indicator.*/
proc sql;
    create table Kids as
    select distinct CustomerNo, IMP_Kids
    from WORK.TABLE2
    order by CustomerNo, IMP_Kids;
run;

/*Creating a table 'Nationality' that contains unique customer numbers and their corresponding nationality, sorted by customer number and nationality.*/
proc sql;
    create table Nationality as
    select distinct CustomerNo, Nationality
    from WORK.TABLE2
    order by CustomerNo, Nationality;
run;
/*DEMOGRAPHIC INFO IS OVER*/
```

```
/*Static Information*/

/*RFM SCORE
-RECENCY
-FREQUENCY
-MONETARY*/

/*Calculating Recency*/
/* Definir a data de referência para 1 de janeiro de 2020 */
%let referenceDate = '1jan2020'd;

/* Calcular Recency com base na data de referência */
proc sql;
    create table Recency as
    select CustomerNo,
           /* Calcular o número de dias entre a última compra e a data de referência */
           intck('day', max(Date), &referenceDate) as DaysSinceLastPurchase
    from WORK.TABLE2
    group by CustomerNo;
run;


/*Creating a table 'Frequency' that counts the total number of transactions per customer.*/
proc sql;
    create table Frequency as
    select CustomerNo, count(distinct(TransactionNo)) as freq
    from WORK.TABLE2
    group by CustomerNo;
run;

/*Creating a table 'Monetary' that calculates the total amount spent by each customer.*/
proc sql;
    create table Monetary as
    select distinct CustomerNo, sum(Unit_Price*Quantity) as Monetary
    from WORK.TABLE2
    group by CustomerNo
    order by CustomerNo;
run;

/*CREATING RFM SCORE*/

/* Assigns ranking for Frequency based on specified limits */
proc sql;
    create table Frequency_Rank as
    select CustomerNo,
           case
               when freq >= 1 and freq < 10 then 1
               when freq >= 10 and freq < 20 then 2
               when freq >= 20 then 3
           end as FrequencyRank
    from Frequency;
quit;

/* Assigns ranking for Recency based on specified limits */
proc sql;
    create table Recency_Rank as
    select CustomerNo,
           case
               when DaysSinceLastPurchase <= 73 then 3
               when DaysSinceLastPurchase > 73 and DaysSinceLastPurchase < 110 then 2
               else 1
           end as RecencyRank
    from Recency;
quit;


/* Assigns ranking for Monetary based on specified limits */
proc sql;
    create table Monetary_Rank as
    select CustomerNo,
           case
               when Monetary < 1000 then 1
               when Monetary >= 1000 and Monetary < 3500 then 2
               when Monetary >= 3500 then 3
           end as MonetaryRank
    from Monetary;
quit;
```

```sql
/* Combines the rankings into a single Score table */
proc sql;
    create table Score as
    select a.CustomerNo,
           a.RecencyRank,
           b.FrequencyRank,
           c.MonetaryRank,
           cats(a.RecencyRank, b.FrequencyRank, c.MonetaryRank) as RFMScore
    from Recency_Rank as a
    join Frequency_Rank as b on a.CustomerNo = b.CustomerNo
    join Monetary_Rank as c on a.CustomerNo = c.CustomerNo;
quit;
/* Assigns ranking for Frequency based on specified limits */
proc sql;
    create table Frequency_Rank as
    select CustomerNo,
           case
               when freq >= 1 and freq < 10 then 1
               when freq >= 10 and freq < 20 then 2
               when freq >= 20 then 3
           end as FrequencyRank
    from Frequency;
quit;

/* Assigns ranking for Recency based on specified limits */
proc sql;
    create table Recency_Rank as
    select CustomerNo,
           case
               when DaysSinceLastPurchase <= 73 then 3
               when DaysSinceLastPurchase > 73 and DaysSinceLastPurchase < 110 then 2
               else 1
           end as RecencyRank
    from Recency;
quit;

/* Assigns ranking for Monetary based on specified limits */
proc sql;
    create table Monetary_Rank as
    select CustomerNo,
           case
               when Monetary < 1000 then 1
               when Monetary >= 1000 and Monetary < 3500 then 2
               when Monetary >= 3500 then 3
           end as MonetaryRank
    from Monetary;
quit;

/* Combines the rankings into a single Score table */
proc sql;
    create table Score as
    select a.CustomerNo,
           a.RecencyRank,
           b.FrequencyRank,
           c.MonetaryRank,
           cats(a.RecencyRank, b.FrequencyRank, c.MonetaryRank) as RFMScore
    from Recency_Rank as a
    join Frequency_Rank as b on a.CustomerNo = b.CustomerNo
    join Monetary_Rank as c on a.CustomerNo = c.CustomerNo;
quit;

/* Adding a column 'cluster' with null val */
proc sql;
    alter table Score
    add CLUSTER char(10);
quit;
```

```
/* updating 'cluster' with RFM */
proc sql;
    update Score
    set CLUSTER =
        case
            when (coalesce(RecencyRank,0) + coalesce(FrequencyRank,0) + coalesce(MonetaryRank,0)) = 3 then 'Bronze'
            when (coalesce(RecencyRank,0) + coalesce(FrequencyRank,0) + coalesce(MonetaryRank,0)) between 4 and 5 then 'Silver'
            when (coalesce(RecencyRank,0) + coalesce(FrequencyRank,0) + coalesce(MonetaryRank,0)) between 6 and 7 then 'Gold'
            when (coalesce(RecencyRank,0) + coalesce(FrequencyRank,0) + coalesce(MonetaryRank,0)) = 8 then 'Platina'
            when (coalesce(RecencyRank,0) + coalesce(FrequencyRank,0) + coalesce(MonetaryRank,0)) = 9 then 'Prestige'
            else 'Unassigned'
        end;
quit;

/* RFM FINISHED*/

/*Creating a table 'CategoryFrequency' that shows how often each customer purchased in each product category.*/
proc sql;
    create table CategoryFrequency as
    select CustomerNo, Product_Category_Name, count(distinct TransactionNo) as freq
    from WORK.TABLE2
    Group by CustomerNo, Product_Category_Name;
run;

/*Sorting 'CategoryFrequency' by customer number.*/
proc sort data=CategoryFrequency;
    by CustomerNo;
run;


/*Creating a table 'MonetaryCategory' that calculates the total amount spent by each customer in each product category.*/
proc sql;
    create table MonetaryCategory as
    SELECT CustomerNo,
           SUM(CASE WHEN Product_Category_Name = 'Beauty & Accessories' THEN Unit_Price ELSE 0 END) AS Beauty_Accessories,
           SUM(CASE WHEN Product_Category_Name = 'Candles & Lights' THEN Unit_Price ELSE 0 END) AS Candles_Lights,
           SUM(CASE WHEN Product_Category_Name = 'Decorative items' THEN Unit_Price ELSE 0 END) AS Decorative_items,
           SUM(CASE WHEN Product_Category_Name = 'Entryway items' THEN Unit_Price ELSE 0 END) AS Entryway_items,
           SUM(CASE WHEN Product_Category_Name = 'Kitchenware' THEN Unit_Price ELSE 0 END) AS Kitchenware,
           SUM(CASE WHEN Product_Category_Name = 'Miscellaneous' THEN Unit_Price ELSE 0 END) AS Miscellaneous,
           SUM(CASE WHEN Product_Category_Name = 'Office supplies' THEN Unit_Price ELSE 0 END) AS Office_supplies,
           SUM(CASE WHEN Product_Category_Name = 'Socks' THEN Unit_Price ELSE 0 END) AS Socks,
           SUM(CASE WHEN Product_Category_Name = 'Sombrero' THEN Unit_Price ELSE 0 END) AS Sombrero
    from WORK.TABLE2
    GROUP BY CustomerNo
    ORDER BY CustomerNo;
run;


/*Creating a table 'FirstPurchase' that records the earliest purchase date for each customer.*/
proc sql;
    create table FirstPurchase as
    select CustomerNo, min(Date) as CustomerSince format=date9.
    from WORK.TABLE2
    group by CustomerNo;
run;

/*Creating a table 'LastPurchase' that records the most recent purchase date for each customer.*/
proc sql;
    create table LastPurchase as
    select CustomerNo, max(Date) as LastPurchase format=date9.
    from WORK.TABLE2
    group by CustomerNo;
run;


/*Creating a table 'MonetaryPerChannel' that calculates the total amount spent by each customer in each channel.*/
proc sql;
    create table MonetaryPerChannel as
    select CustomerNo,
           SUM(CASE WHEN Channel = 'Store' THEN Unit_Price*Quantity ELSE 0 END) AS MonetaryStore,
           SUM(CASE WHEN Channel = 'Online' THEN Unit_Price*Quantity ELSE 0 END) AS MonetaryOnline
    from WORK.TABLE2
    group by CustomerNo;
run;


/*Creating a table 'FreqChannel' that counts the number of purchases each customer made in each channel.*/
proc sql;
    create table FreqChannel as
    select CustomerNo,
           sum(CASE WHEN Channel = 'Store' THEN 1 ELSE 0 END) AS FreqStore,
           sum(CASE WHEN Channel = 'Online' THEN 1 ELSE 0 END) AS FreqOnline
    from WORK.TABLE2
    group by CustomerNo;
run;
```

```sql
/*Creating FavoriteCategory*/
/* Passo 1: Count Purchases per Category Name */
proc sql;
    create table CategoryCounts as
    select CustomerNo, Product_Category_Name, count(*) as PurchaseCount
    from WORK.TABLE2
    group by CustomerNo, Product_Category_Name;
quit;

/* Passo 2: Identify the most purchased category per client */
proc sql;
    create table FavoriteCategory as
    select a.CustomerNo, a.Product_Category_Name
    from CategoryCounts as a
    where a.PurchaseCount = (select max(b.PurchaseCount) from CategoryCounts as b where a.CustomerNo = b.CustomerNo)
    group by a.CustomerNo, a.Product_Category_Name;
quit;
/*Favorite Category OVER*/

/*Creating a table with Customer Duration*/
/* Joining first and last purchase */
proc sql;
    create table FirstLastPurchase as
    select CustomerNo,
           min(Date) as FirstPurchase,
           max(Date) as LastPurchase
    from WORK.TABLE2
    group by CustomerNo;
quit;

/* Calculating days between dates */
proc sql;
    create table CustomerDuration as
    select CustomerNo,
           intck('day', FirstPurchase, LastPurchase) as DurationDays
    from FirstLastPurchase;
quit;
/*Customer Duration OVER*/
```

```sql
/*Creating table with how many categories did Customer bought*/
proc sql;
    create table CategoryCountPerCustomer as
    select CustomerNo,
           count(distinct Product_Category_Name) as NumCategories
    from WORK.TABLE2
    group by CustomerNo;
quit;

/*Frequency on PaymentType*/
proc sql;
    create table PaymentFrequency as
    select CustomerNo,
           sum(case when Payment = 'Paypal' then 1 else 0 end) as PayPalFrequency,
           sum(case when Payment = 'Cash' then 1 else 0 end) as CashFrequency,
           sum(case when Payment = 'Credit Card' then 1 else 0 end) as CreditCardFrequency
    from WORK.TABLE2
    group by CustomerNo;
quit;


/*Creating Favorite Payment Type*/
proc sql;
    /* Criação de uma tabela temporária para contar a frequência de pagamentos */
    create table PaymentCounts as
    select CustomerNo, Payment, count(*) as Frequency
    from WORK.TABLE2
    group by CustomerNo, Payment;
quit;
```

```sql
/*Creating Favorite Payment Type*/
proc sql;
    /* Criação de uma tabela temporária para contar a frequência de pagamentos */
    create table PaymentCounts as
    select CustomerNo, Payment, count(*) as Frequency
    from WORK.TABLE2
    group by CustomerNo, Payment;
quit;

/*Creating most used payment method*/
proc sql;
    /* Identification of the most used */
    create table PaymentFavouriteIntermediate as
    select a.CustomerNo,
           a.Payment,
           a.Frequency,
           count(*) as NumOfModes
    from PaymentCounts as a
    inner join
        (select CustomerNo, max(Frequency) as MaxFreq
         from PaymentCounts
         group by CustomerNo) as b
    on a.CustomerNo = b.CustomerNo and a.Frequency = b.MaxFreq
    group by a.CustomerNo, a.Frequency;
quit;

/* Set ties as 'No Favourite' */
data PaymentFavourite;
    set PaymentFavouriteIntermediate;
    by CustomerNo;
    if first.CustomerNo and last.CustomerNo and NumOfModes = 1 then FavoritePayment = Payment;
    else FavoritePayment = 'No Favorite';
run;
/*Favourite payment type OVER*/



/*Creating table with the total amount spent in each payment method*/
proc sql;
    create table MonetaryPayment as
    select CustomerNo,
           sum(case when Payment = 'Paypal' then Unit_Price * Quantity else 0 end) as PayPalMonetary,
           sum(case when Payment = 'Cash' then Unit_Price * Quantity else 0 end) as CashMonetary,
           sum(case when Payment = 'Credit Card' then Unit_Price * Quantity else 0 end) as CreditCardMonetary
    from WORK.TABLE2
    group by CustomerNo;
quit;

/*Creating table with the total amount spent in each Product Category*/
proc sql;
    create table MonetaryPerCategory as
    select CustomerNo,
           SUM(CASE WHEN Product_Category_Name = 'Beauty & Accessories' THEN Unit_Price * Quantity ELSE 0 END) AS Monetary_Beauty_Accessories,
           SUM(CASE WHEN Product_Category_Name = 'Candles & Lights' THEN Unit_Price * Quantity ELSE 0 END) AS Monetary_Candles_Lights,
           SUM(CASE WHEN Product_Category_Name = 'Decorative items' THEN Unit_Price * Quantity ELSE 0 END) AS Monetary_Decorative_items,
           SUM(CASE WHEN Product_Category_Name = 'Entryway items' THEN Unit_Price * Quantity ELSE 0 END) AS Monetary_Entryway_items,
           SUM(CASE WHEN Product_Category_Name = 'Kitchenware' THEN Unit_Price * Quantity ELSE 0 END) AS Monetary_Kitchenware,
           SUM(CASE WHEN Product_Category_Name = 'Miscellaneous' THEN Unit_Price * Quantity ELSE 0 END) AS Monetary_Miscellaneous,
           SUM(CASE WHEN Product_Category_Name = 'Office supplies' THEN Unit_Price * Quantity ELSE 0 END) AS Monetary_Office_supplies,
           SUM(CASE WHEN Product_Category_Name = 'Socks' THEN Unit_Price * Quantity ELSE 0 END) AS Monetary_Socks,
           SUM(CASE WHEN Product_Category_Name = 'Sombrero' THEN Unit_Price * Quantity ELSE 0 END) AS Monetary_Sombrero
    from WORK.TABLE2
    group by CustomerNo;
quit;
```

```
/*Average Variables*/


/*Creating Average Spent per Purchase*/
/* Passo 1: Calculate the total_paid per client per purchase */
proc sql;
    create table TotalSpentPerTransaction as
    select CustomerNo, TransactionNo, sum(Unit_Price * Quantity) as TotalSpent
    from WORK.TABLE2
    group by CustomerNo, TransactionNo;
quit;

/* Passo 2: Calculate the average */
proc sql;
    create table AvgSpentPerPurchase as
    select CustomerNo, mean(TotalSpent) as AvgSpent
    from TotalSpentPerTransaction
    group by CustomerNo;
quit;



/*Create table with average days between purchases*/
/* Criar uma cópia da TABLE2 ordenada por cliente e data */
proc sort data=WORK.TABLE2 out=SortedPurchases;
    by CustomerNo Date;
run;

/* Calculating the difference of days between 2 consecutive purchases */
data Diffs;
    set SortedPurchases;
    by CustomerNo;
    retain prevDate;
    if first.CustomerNo then do;
        prevDate = Date;
        DaysDiff = .; /* Não calcular diferença para a primeira compra de cada cliente */
    end;
    else do;
        DaysDiff = Date - prevDate; /* Calcular a diferença para compras subsequentes */
        prevDate = Date;
    end;
    if not first.CustomerNo;
run;
/*Average days between Purchases OVER*/


/*Merging to ABT*/


data final;
merge Age kids nationality recency frequency monetary
score firstpurchase lastpurchase CustomerDuration
freqchannel  monetaryperchannel PaymentFrequency PaymentFavourite MonetaryPayment
MonetaryPerCategory favoritecategory
AvgSpentPerPurchase AvgDaysBetweenPurchases
CategoryCountPerCustomer ;
by CustomerNo;
run;
```

Bug Solving:

```
data WORK.final_copy;
    set WORK.final_copy;

    /* Converter para inteiro */
    CustomerNo = input(put(CustomerNo, best32.), best32.);
    Age = input(put(Age, best32.), best32.);
    DaysSinceLastPurchase = input(put(DaysSinceLastPurchase, best32.), best32.);
    freq = input(put(freq, best32.), best32.);
    RecencyRank = input(put(RecencyRank, best32.), best32.);
    FrequencyRank = input(put(FrequencyRank, best32.), best32.);
    MonetaryRank = input(put(MonetaryRank, best32.), best32.);
    RFMScore = input(put(RFMScore, best32.), best32.);
    DurationCustomer = input(put(DurationCustomer + 1, best32.), best32.); /* Adicionar +1 Ass. Francisco Batista */
    FreqStore = input(put(FreqStore, best32.), best32.);
    FreqOnline = input(put(FreqOnline, best32.), best32.);
    PaypalFrequency = input(put(PaypalFrequency, best32.), best32.);
    CashFrequency = input(put(CashFrequency, best32.), best32.);
    CreditCardFrequency = input(put(CreditCardFrequency, best32.), best32.);
    AvgDaysBetweenPurchase = input(put(AvgDaysBetweenPurchase, best32.), best32.);
    NumCategories = input(put(NumCategories, best32.), best32.);
    monetary = input(put(monetary, best32.), best32.);
    MonetaryStore = input(put(MonetaryStore, best32.), best32.);
    MonetaryOnline = input(put(MonetaryOnline, best32.), best32.);
    PaypalMonetary = input(put(PaypalMonetary, best32.), best32.);
    CashMonetary = input(put(CashMonetary, best32.), best32.);
    CreditCardMonetary = input(put(CreditCardMonetary, best32.), best32.);
    Monetary_Beauty_Accessories = input(put(Monetary_Beauty_Accessories, best32.), best32.);
    Monetary_Candles_Lights = input(put(Monetary_Candles_Lights, best32.), best32.);
    Monetary_Decorative_Items = input(put(Monetary_Decorative_Items, best32.), best32.);
    Monetary_Entryway_Items = input(put(Monetary_Entryway_Items, best32.), best32.);
    Monetary_Kitchenware = input(put(Monetary_Kitchenware, best32.), best32.);
    Monetary_Miscellaneous = input(put(Monetary_Miscellaneous, best32.), best32.);
    Monetary_Office_Supplies = input(put(Monetary_Office_Supplies, best32.), best32.);
    Monetary_Socks = input(put(Monetary_Socks, best32.), best32.);
    Monetary_Sombrero = input(put(Monetary_Sombrero, best32.), best32.);

    /* Converter datas para formato date9. */
    CustomerSince = input(put(CustomerSince, date9.), date9.);
    LastPurchase = input(put(LastPurchase, date9.), date9.);

    /* Variáveis varchar (string) não precisam de conversão */
run;
```

‘Customer Behaviour Analysis using SQL - Portfolio Project - A Case Study’ **, Her Data Project**, 1st December, https://www.youtube.com/watch?v=lHR1_j8DYFA

Customer, Product and Sales Analysis with T-SQL (Part 1) - Medium - https://nguy3387.medium.com/customer-product-and-sales-analysis-with-t-sql-part-1-79874b8e8320

SAS Miner Guide – Data Exploration - Data Pre-Processing and Visualization - Joana Neves

‘SAS Miner Guide – Outliers Treatment’- Data Pre-Processing and Visualization - Joana Neves

‘SAS Miner Guide – Misssing Values Treatment Guide’- Data Pre-Processing and Visualization - Joana Neves

‘SAS Miner Guide – Data Transformation Guide’- Data Pre-Processing and Visualization - Joana Neves