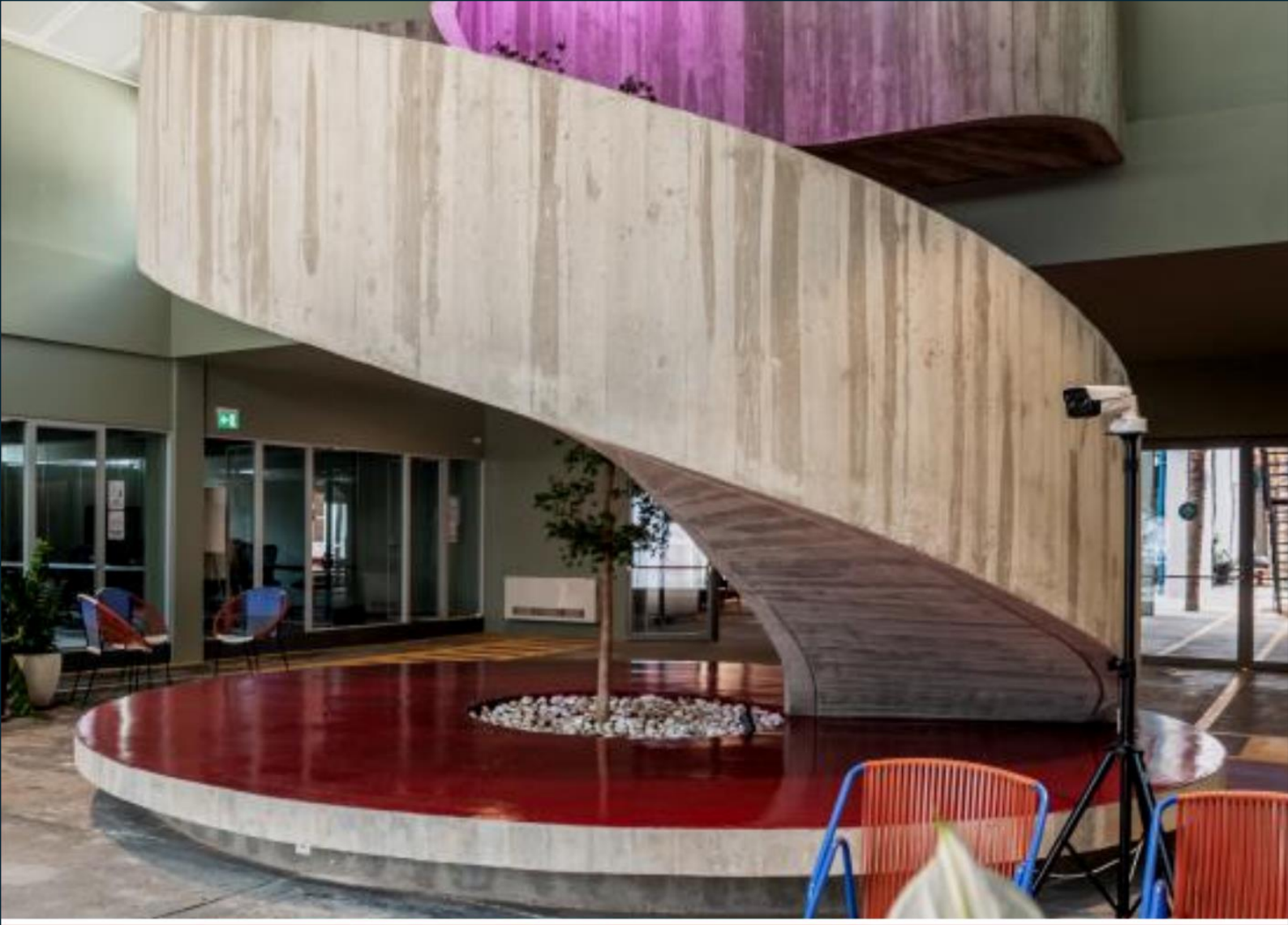

# BUSINESS INTELLIGENCE

## GROUP 25 REPORT

NOVA IMS Information Management School

Retail4all

# BUSINESS INTELLIGENCE

## MASTER DEGREE PROGRAM IN DATA SCIENCE AND ADVANCED ANALYTICS – SPECIALIZATION IN BUSINESS ANALYTICS

## RETAIL4ALL

A Group 25 Project by:

Alexandre Spagnol, 20230434

Gonçalo Ferreira, 20230492

Hugo Alves, 20230438

Sebastião Oliveira, 20220558

June 2024

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

# TABLE OF CONTENTS

# INDEX OF FIGURES

# 1. BUSINESS PRESENTATION

## 1.1. RETAIL4ALL INTRODUCTION

Retail4all is a Portuguese retail company dedicated to delivering top quality products and services across the entire country. Our commitment to innovation and excellence made it possible to be established as a trusted name in the retail industry. Retail4all is focused on selling different types of products; from fashion and beauty to electronics and home essentials, Retail4all offers a diverse range of products to cater to every lifestyle and budget. A schematic representation of the company's daily business process can be summarized as follows, with the larger arrows representing the normal flow of products:


**Figure 1** - Retail4all's Normal Workflow

Retail4all embraces innovation and aims to stay ahead of evolving markets to anticipate and meet changing behaviours of the customers. By leveraging technology and data-driven insights, we can keep up the exceptional work we are known for. That is why it is so important to have report and dashboarding tools to improve our analytical output and decision-making processes.

Retail4all can be found all around Portugal, with over 30 stores, and close to 90 different products. Our sales have been increasing for the past 3 years and, in 2023, the prediction is to finish the year with 4.6 million euros in sales (Figure 1). From 2020 to 2022, we were able to process almost 900 thousand sales, with a total of 3 million products sold resulting in a total just shy of the 11 million euros. This means that we are already considered as a small and medium-sized enterprise.

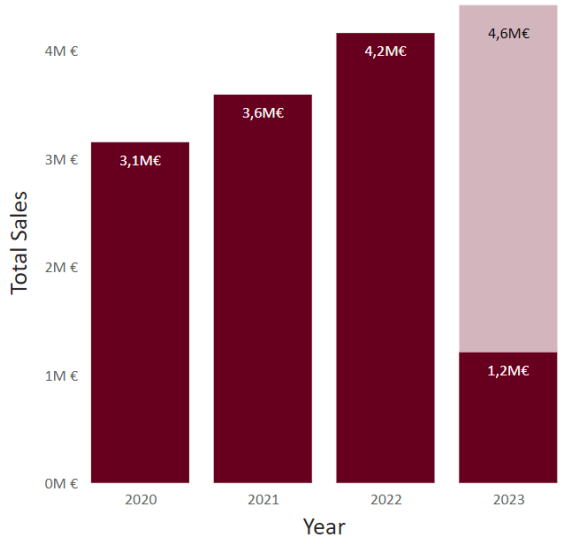


**Figure 2** - Retail4all Yearly Sales (and 2023 prediction)

Retail4all mainly sells electronic products, resulting in approximately 68% of all the sales in the available data. From the plots below, we can also gather that, overall, sales quantity and sales amount tend to follow similar distributions regarding product category.

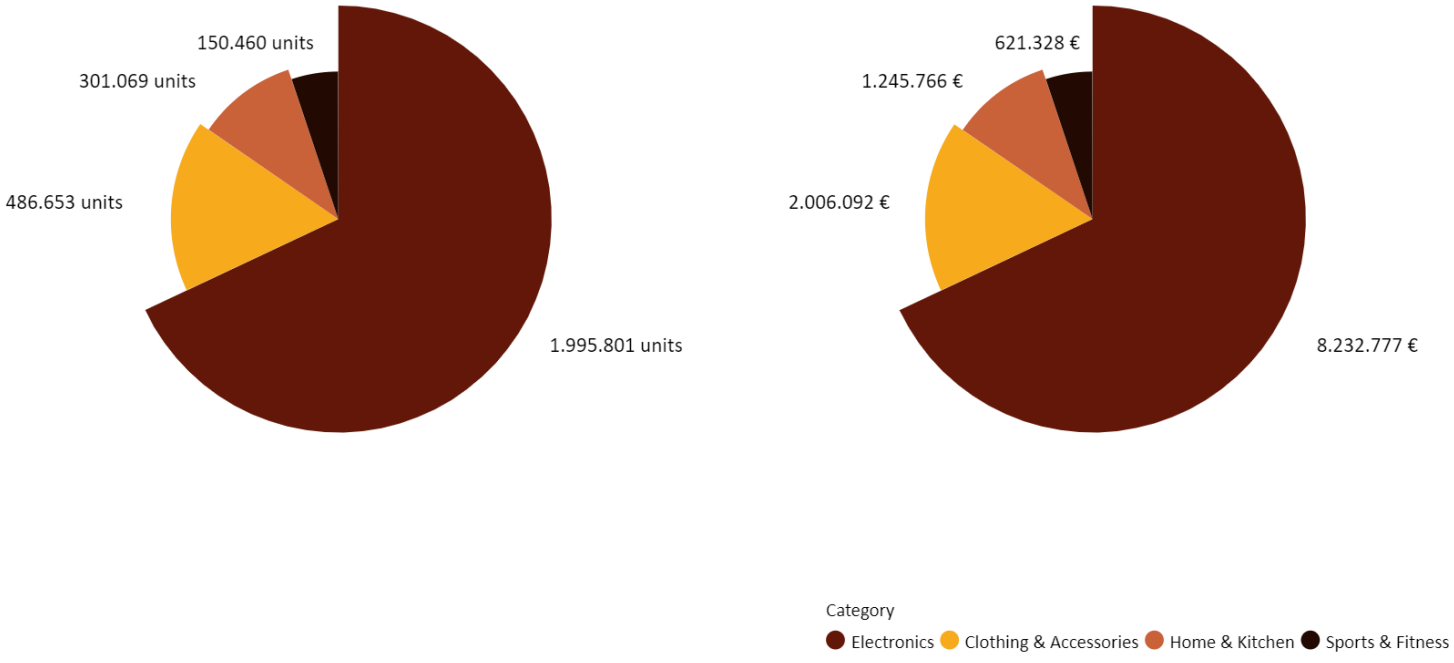


**Figure 3** - Sales by Category (by quantity and amount)

Our sales are mostly located on the north region of Portugal, with Aguiar da Beira – a town in the Viseu district – being the one where the sales amount is the highest. Considering that Retail4all has 13 stores in this location (more than in any other place), it would be natural to assume that most of our revenue would be originated there.

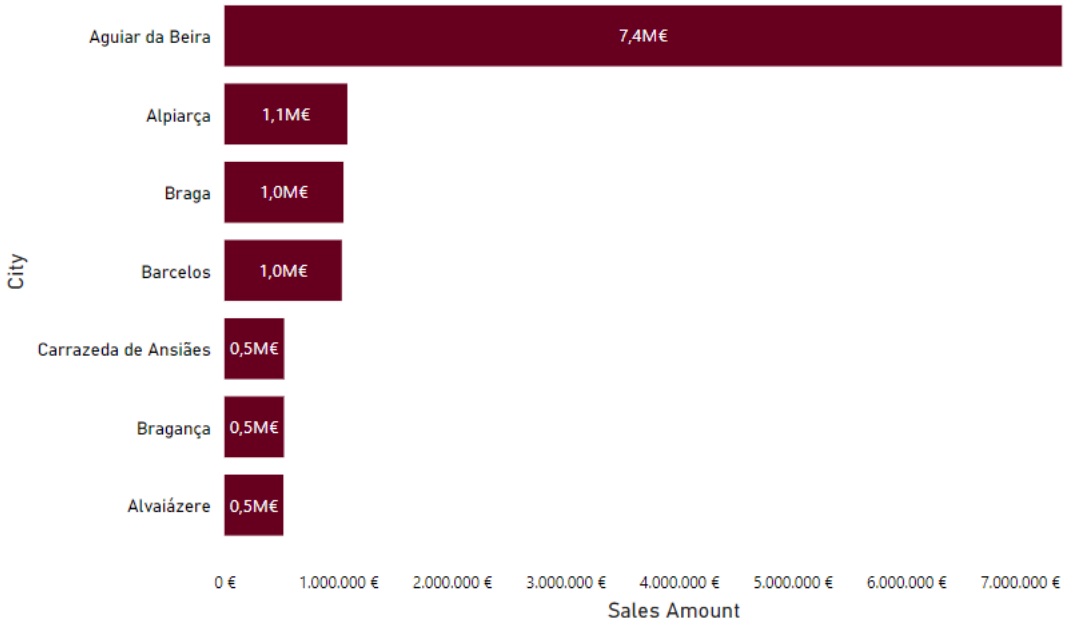


**Figure 4** - Sales Amount by Location

## 1.2. BUSINESS PROBLEM

It seems that, at the moment, the analytical tools that Retail4all has access to are not optimized. As such, the company's administration has set the goal to evolve from the present rudimentary business intelligence solution to one that allows for data-driven decision-making and has assigned a specialized team to achieve that goal.

Based on approximately three years of historical data that has been made available, we will aim to analyse the current state of sales and create a dashboard as complete as needed to make this process as efficient as possible. This dashboard should be able to give valuable insight into sales, including quantity and amount of sales. We should never forget the past, so Year-Over-Year (YOY) and Year-to-Date (YTD) analysis will also need to be allowed. Additionally, we want to understand our customers better, so profiling will need to be available for analysis. We should be able to easily see the category and subcategory of product sales, as well as what each store sells.

### 1.2.1. BUSINESS QUESTIONS

In order to identify the business needs for the project under development, we have to consider and start by understanding what are the different interests of the stakeholders involved. Considering the importance given by Retail4all's administration to the analysis of sales (in quantity and amount) and how they are evolving over different time periods and locations, we will list below the business questions we find to be the most relevant:

1. What is our number of sales across distinct time periods (month, quarter, and year)?
2. How is our accumulated performance in sales to date?
3. How have we evolved compared to the previous year's sales performance?
4. How much money are we making every month?
5. What products are our customers buying more? And when?
6. Who are the employees responsible for the most sales?
7. What categories of products are most sold in each location (by sales quantity)?
8. What are the locations where we have sold the most?
9. What channel has a bigger weight in our sales (online vs face-to-face)? And how is it evolving?

Over the following chapters (in specific, when we identify the fact table for our dimensional model), we will detail how these business questions translate into different measures and how they will be integrated within the data warehouse.

## 2. DATA SOURCES

For this project, we were given a total of 7 datasets from different areas of the business. These were:

- **Locations** – A text file containing Portuguese geographical information. This included the following columns:
  - Location ID: An integer value that uniquely identifies a location (city or town).
  - City: The name of the location (which could be either a city or a town).
  - District: The district where the city previously mentioned is located in. In almost all cases, the value for this attribute was missing.
  - Country: The name of the country where the city previously mentioned is located in. For all towns, the corresponding country was Portugal.
- **Stores A** and **Stores B** – These two text files were grouped into a single table, as they contained information regarding the same variables and Stores B was solely a continuation of Stores A. The features included were:
  - Store ID: An integer value that uniquely identifies the store to whom Retail4all sells their products.
  - Nome (Name): The name of the store.
  - Localização (Location): An integer that identifies the location in which the store is in. This variable is a foreign key, being directly related to "Location ID", the unique identifier of a location in the "Locations" table.
  - Online: An attribute that is either "Y" (Yes) or "S" (Sim) if the store sells the products to its clients through an online platform, or "N" (No) otherwise. If values were missing, we assumed that these stores also didn't utilize the online method to sell products.
- **Products** – A text file with data related to the products sold by Retail4all. This dataset included the following attributes:
  - SKU: A code (business key) that uniquely identifies the product in question.

- Product Name: The name of the product. In general, this variable included the name of the brand in the product name (which would be the sensible thing to do, considering the absence of a "Brand" column, but could not be the case).
- Category: The broader category to which the product belongs to, according to Retail4all's classification. This could either be "Electronics", "Clothing & Accessories", "Sports & Fitness" or "Home & Kitchen".
- Subcategory: Subdivision of the product category which more specifically describes the type of product in question. Naturally, the options available for this feature will vary depending on the "Category" assigned to the product.

- **Operators** – This text file included information about Retail4all's staff, mainly regarding:
  - Operator id: An integer value that uniquely identifies each company collaborator.
  - first_name: The first name of the employee in question.
  - last_name: The last name of the employee in question.
  - email: The email (if provided) of the employee.
  - gender: The gender of the employee.
  - role: The function developed by the employee in Retail4all.
  - team: An integer value identifying the team each employee was assigned to. From the analysis of the dataset, we learned that "1" was related to employees in higher positions (for example, directors), "2" was assigned to collaborators in "floor" positions (such as Floor Manager or Floor Staff) and "3" was the identifier of Retail4all's sales staff (which also included a checkout manager).
- **Point of Supply** – Another text file, this time related to data about Retail4all's suppliers. The attributes present in this data source were**:**
  - POS id: An integer value that uniquely identifies each company's supplier.
  - name: The name of the supplier.
  - email: The email address of the supplier.
- **Sales** – This text file was, by far, the biggest dataset provided in terms of length, as it included information about each sale made by Retail4all. This information was detailed in terms of:
  - Sale ID: An integer value uniquely identifying each sale.
  - Datetime: The date and time in which the sale transaction occurred. This feature was expressed in the format DD/MM/YYYY HH:MM.
  - SKU: The code related with the product sold. This variable is a foreign key, directly associated to the business key identified in the "Products" dataset.
  - Store: An integer value informing about the location in which the sale occurred. This feature is also a foreign key, directly associated with the "Store ID" column of the "Stores A" and "Stores B" dataset.
  - POS: An integer value identifying who supplied the product that was sold. This attribute is a foreign key, directly related to the "POS id" variable of the "Point of Supply" dataset.
  - Localização (Location): An integer value informing about the location in which the sale was made. This is also a foreign key, being directly related to the "Location ID" attribute of the "Locations" dataset.

- Currency: The identifier of the currency in which the transaction was paid. All sales were paid using Euro ("EUR"), which makes sense considering that Retail4all only operates in Portugal and only has Portuguese clients (as seen in the "Stores" table).
- Quantity: The number of items sold of the product in that specific sale.
- Amount: The total amount of the sale.
- Dt_Cri: An auxiliary datetime variable whose values were always one minute after the shown in the "Datetime" variable. Similarly, this feature was expressed in the format DD/MM/YYYY HH:MM.
- operator_id: An integer value identifying the employee who made the sale. Like the previous three variables, this one is also a foreign key, this time associated with the "Operator id" feature of the "Operators" dataset.

# 3. DIMENSIONAL MODELLING

For the design of our data warehouse, following a defined methodology is essential to allow for a clear definition of the steps that should be taken. As such, we opted for the Kimball methodology, consisting of the following steps:

1. Identify the business process – At this first step, we should focus on identifying and understanding the business process that we wish our dimensional model to represent.
2. Identify the grain – Here, we decide on the level of detail of our dimensional model, that is, what a row in our fact table actually represents.
3. Identify the dimensions – Next, we should group the variables that we have access to in different dimensions (similar to entities). These should be descriptive and consider the grain previously defined.
4. Identify the facts – Finally, we should be able to create one or more fact tables, containing the measures (presented in a numeric format) that we identify as relevant and also respecting the granularity defined in step 2.

We will dedicate each of the following four subchapters to each of the steps presented in the Kimball Methodology. In the fifth, we will present the final dimensional model design and what was the selected schema to build it.

## 3.1. IDENTIFY THE BUSINESS PROCESS

We already introduced a few steps into the business process. We previously provided a context for the current situation of the company and the primary business problem, defined the business questions we want to answer, as well as the data we have available to get to them. The focus of this project will be around sales related measures and problems. From the data sources provided, we have several good dimensions to create the fact table.

We did our best to get all stakeholders involved in this process, by asking for feedback about the questions defined previously, while also making sure we have data available to answer those questions. They play a vital role in providing valuable insights into the business process.

The main KPI's we will use are sales related, such as gross sales, number of sales, quantities sold, best performing products, and best performing stores. We also desire to keep every stakeholder

that was involved at the starting point for this project, in a feedback loop, to make sure we are on the right direction and the needs are satisfied.

## 3.2. IDENTIFY THE GRAIN

Granularity is an important part of the chosen methodology, as here we determine the level of detail we can store and analyse. In other words, we define what will be the meaning of each single data point.

After a thorough analysis and a lot of discussion with the stakeholder involved, we decided that the ideal level of granularity for our sales would be sales at a daily level, product level, and city level, and it should also consider the operator who made it, the store in which it was made and the supplier who sold the product to Retail4all in the first place. As such, each sale was defined as "the daily transaction of a product furnished by a certain supplier, sold in a specific store of a certain city by a determined operator". Each sale can involve than a transaction of more than one product unit and will have an amount associated with it, as we will later see when creating the fact table.

### 3.2.1 IDENTIFY HIERARCHIES

From the data that was provided, there are four main hierarchies that we are able to identify – concerning dates, products, locations and sellers -, which are directly related to the level of detail desired and were reflected in the definition of the grain.

Regarding temporal granularity, we can say that technically, the available data allowed us to analyse the sales at a deeper granular level, such as sales per minute, but we found that day would provide the best balance between detail and usability for the needed analytical tasks. Daily granularity will enable us to capture meaningful trends and patterns in sales, without overwhelming the user.

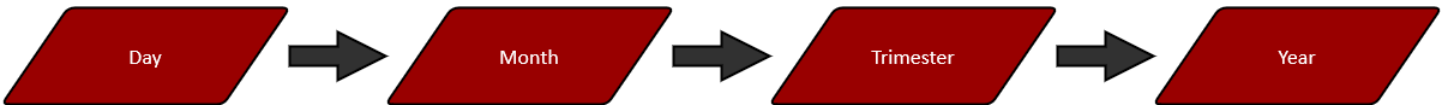


**Figure 5** - Dates Hierarchy

In terms of product granularity, we opted by capturing sales at a product level. This is the lowest granularity possible, following the order bellow. This level of detail will allow us to analyse the best performing product, the main product we sell as well as the least sold products.

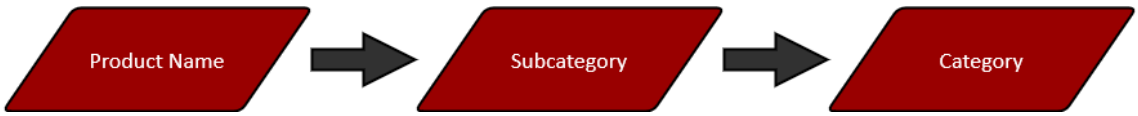


**Figure 6** - Products Hierarchy

Additionally, location granularity will be at city-level, enabling us to analyse sales performance across the cities we are inserted. Also, consumer behaviour can be analysed using this level of granularity. Finally, for the future, it leads to efficient data management where only the needed city can be analysed.


**Figure 7** - Locations Hierarchy

Finally, the sellers granularity will be at the operator level, allowing us to identify the individual vendors who performed the most sales, as well as their roles and the name of the team they belong to. We should note that, at this point, we don't yet have the team name of the employee explicitly defined. This is something that we will later calculate upon the ETL process.


**Figure 8** - Sellers Hierarchy

## 3.3. IDENTIFY THE DIMENSIONS

After defining the granularity level intended for this project, we can now proceed, respecting Kimball's methodology, to create the dimension tables that will allow for the representation of the desired grain level. We identified six different dimensions that were then grouped into six tables, as described over the following subchapters. In each one, a small description of the dimension will be provided, as well as a table containing the variables that are present in it. In each table, the surrogate key (that is, the unique identifier of a table – which differs from the traditional "primary key" definition in the sense that it doesn't have any contextual or business meaning) will be represented with "SK" between parenthesis, after the name of the variable.

### 3.3.1. DIM_LOCATION

This dimension, as the name suggests, contains geographic information that will later support our Sales fact table. It is equivalent to the "Locations" dataset described in the chapter related to the data sources. The only addition to this table is the new "City_Country" column, that will be created with the purpose of helping maps identify locations later in the dashboard stage. It will be the name of the city (from the "City" column) followed by a comma, a space, and the name of the country (from the "Country" column).

| Dim_Location | |
|---|---|
| **Variable** | **Data Type** |
| SK_Location (SK) | INT (10) |
| City | VARCHAR (100) |
| District | VARCHAR (100) |
| Country | VARCHAR (100) |
| City_Country | VARCHAR (150) |

### 3.3.2. DIM_STORE

This dimension describes the different stores to which Retail4all sells their products to.

| Dim_Store | |
|---|---|
| **Variable** | **Data Type** |
| SK_Store (SK) | INT (10) |
| Store_Name | VARCHAR (100) |
| Online | VARCHAR (1) |

### 3.3.3. DIM_POINT_OF_SUPPLY

This dimension includes data regarding the suppliers from which Retail4all buys the products they later sell. It is equivalent to the "Point of Supply" dataset described in the data sources chapter.

| Dim_Point_of_Supply | |
|---|---|
| **Variable** | **Data Type** |
| SK_POS (SK) | INT (10) |
| Supplier_Name | VARCHAR (100) |

### 3.3.4. DIM_CALENDAR

This dimension was manually created, considering the importance of dates to further analyse sales evolution overtime. Therefore, its level of detail is also deeply connected to the business questions proposed and with the desired level of granularity. Here, even though the "Full_Date" variable was already unique considering the desired granularity, we created a surrogate key containing the date in the format YYYYMMDD, as it is good practice to create this type of key when handling dates and it may be more practical going forward.

| Dim_Calendar | |
|---|---|
| **Variable** | **Data Type** |
| SK_Date (SK) | INT (8) |
| Full_Date | DATE |
| Year | INT (4) |
| Quarter_Number | INT (1) |
| Quarter_Name | VARCHAR (10) |
| Quarter_Name_Short | VARCHAR (5) |
| Month_Number | INT (2) |
| Month_Name | VARCHAR (10) |
| Month_Name_Short | VARCHAR (3) |
| Day_Number | INT (2) |

### 3.3.5. DIM_OPERATOR

This dimension concerns Retail4all's employees and grabs its information from the "Operators" dataset, earlier described in the context of the identification of the different data sources.

| Dim_Operator | |
|---|---|
| **Variable** | **Data Type** |
| SK_Operator (SK) | INT (10) |
| Full_Name | VARCHAR (100) |
| Gender | VARCHAR (20) |
| Operator_Role | VARCHAR (100) |
| Team_Name | VARCHAR (100) |

We could have created a separate dimension for the team to which the operator belongs and linked it with the operators' dimension through the fact table. However, in context of the problem and of the solution that we were asked for, we did not find it relevant to have the team as more than an attribute of an operator, with that being the main reason for not creating a new dimension in our dimensional model.

### 3.3.6. DIM_PRODUCT

Finally, the last dimension is related to the products that are part of Retail4all's inventory. Although the base source for this information comes from the "Products" dataset (described in the data sources chapter), a Product ID ("SK Product") was added to act as a surrogate key, replacing "SKU" (which was mostly a business key) as the primary unique identifier of a product.

| Dim_Product | |
|---|---|
| **Variable** | **Data Type** |
| SK_Product (SK) | INT (10) |
| SKU | VARCHAR (10) |
| Product_Name | VARCHAR (100) |
| Category | VARCHAR (100) |
| Subcategory | VARCHAR (100) |

## 3.4. IDENTIFY THE FACTS

Finally, having identified the dimensions, the final step in Kimball's methodology consists of identifying the facts that will take part in our dimensional model. For this project, a single fact table is required, which will be related to the sales made by Retail4all. It is important to enhance that, now, the definition of a "sale" will be dependent on the level of granularity defined. As such, and as was mentioned above, each sale will be considered as "the daily transaction of a product furnished by a certain supplier, sold in a specific store of a certain city by a determined operator".

The fact table will contain all surrogate keys from the dimension tables initially created (which will now act as foreign keys), but also measures created with the objective of answering the business questions earlier defined. The following two tables explain the process of identifying the two measures that will be used, related to sales quantity and amount:

| ID | Business Question (Need) | Measures |
|---|---|---|
| 1 | What is our number of sales across distinct time periods (month, quarter, and year)? | Number of sales |
| 2 | How is our accumulated performance in sales to date? | Accumulated sales |
| 3 | How have we evolved compared to the previous year's sales performance? | Sales (YTD) |
| 4 | How much money are we making every month? | Sales amount |
| 5 | What products are our customers buying more? And when? | Sales quantity |
| 6 | Who are the employees responsible for the most sales? | Gross sales |
| 7 | What categories of products are most sold in each location (by sales quantity)? | Sales quantity |
| 8 | What are the locations where we have sold the most? | Sales amount |
| 9 | What channel has a bigger weight in our sales (online vs face-to-face)? And how is it evolving? | Sales amount |

| ID | DW Raw (Primitive) Measures | Original OLTP Field |
|---|---|---|
| 1 | Quantity | Qty |
| 2 | Amount | Amount |

Having defined these measures, we can now summarize all variables of the sales fact table and their corresponding data types as follows:

| Fact_Sales | |
|---|---|
| **Variable** | **Data Type** |
| FK_Date | INT (8) |
| FK_Location | INT (3) |
| FK_Product | INT (3) |
| FK_Store | INT (3) |
| FK_Operator | INT (3) |
| FK_POS | INT (3) |
| Quantity | INT (2) |
| Amount | DECIMAL (18, 2) |

## 3.5. FINAL DIMENSIONAL MODEL

Having successfully defined the dimensions and fact that will be part of our dimensional model, we are now able to integrate them and propose a final design. This design will respect the Star Schema variant, in which the fact table will be connected to all dimension tables through one-to-many relationships (from the dimension to the fact table).

Despite employing more redundancy, this type of schema is also faster to query, less complex (and thus, more understandable) and more scalable, being in a better position to answer to an expansion need than, for instance, a Snowflake schema. All things considered, the following schema represents the design of our dimensional model:

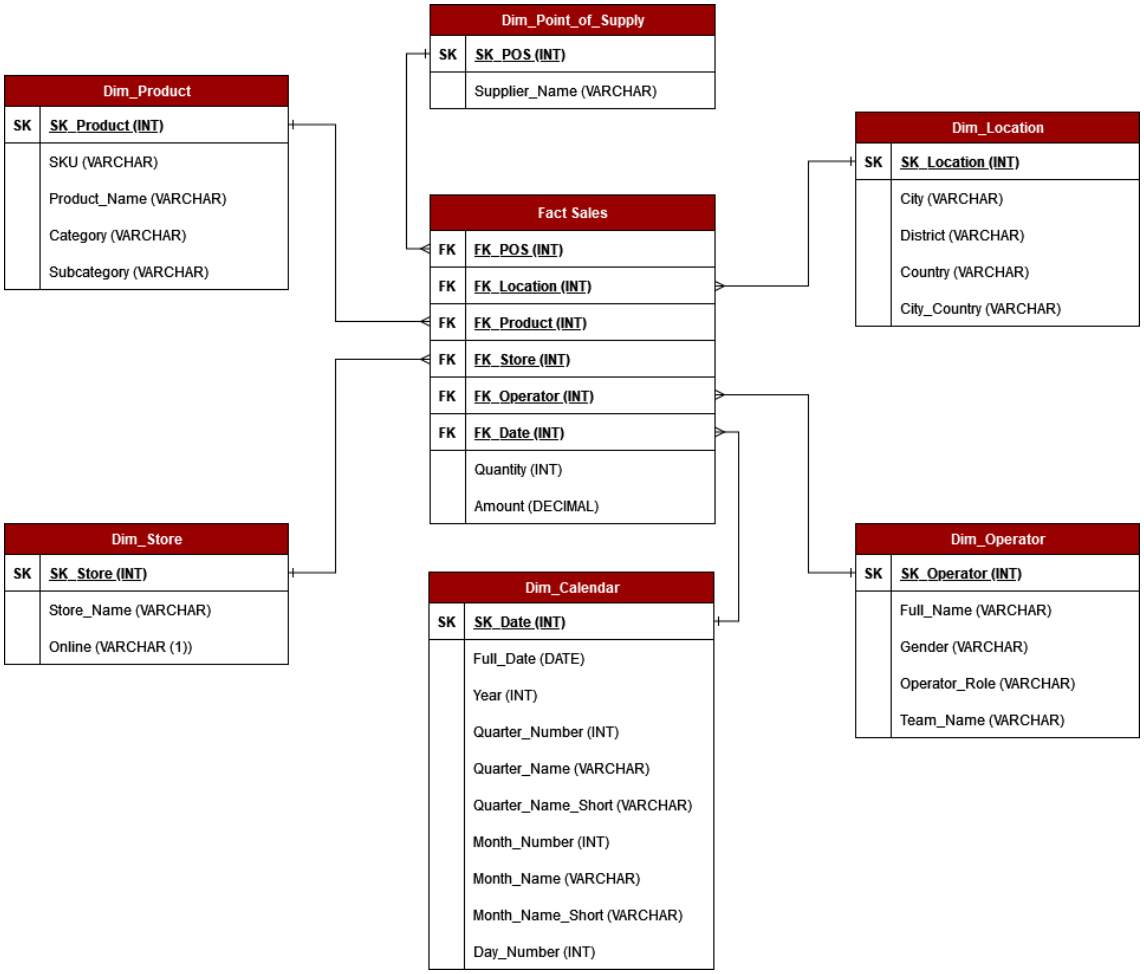


**Figure 9** - Retail4all's Dimensional Model

The proposed dimensional model shows all dimensions connected to the sales fact table through their surrogate keys, which are represented as foreign keys in the fact table. We believe that this schema reflects the four universally accepted characteristics of a data warehouse, as proposed by Inmon: it is subject-oriented (organized, in this case, by sales); integrated (gathers data from a variety of sources in a coherent fashion); time-variant (focused on change over time, as reflected by the presence of a calendar dimension); and non-volatile (having entered the dimensional model, the collected data will no longer be altered). Therefore, having set-up the base for the dimensional model, we can now proceed with the following steps for our project.

# 4. ETL PROCESS

## 4.1. LOADING DATA TO THE LAKEHOUSE

The first step in the ETL process, which marked the start of the use of Microsoft Fabric for this project, was to create an environment dedicated to it. We named it "BI MAA 2024 Group 25". Next, on Data Factory, we created a Lakehouse to serve as the repository of our raw, untransformed data, which we named "LH_RETAIL4ALL_SOURCES". Here, all we had to do was to upload the (CSV) files that were provided to us and previously described in the data sources' chapter.

## 4.2. CREATING DATAFLOWS

After the creation of the Lakehouse, we proceeded with the creation of the data warehouse, named "DW_RETAIL4ALL", which would store the organized data to support the decision-making process. In here, we wrote an SQL script (visible in the Fabric environment) to create the tables that we previously defined as part of our dimensional model, as well as their respective columns and data types.

Regarding variables that we created as strings (VARCHAR) or non-whole numbers (DECIMAL), we had to define the maximum size allowed for a value on each column. In these cases, we defined very big numbers compared to what we would most likely be needing, which was something that was consciously done in order to guarantee that we would never have problems introducing data into these columns due to size constraints.

Another aspect we took into account upon creating the tables was not allowing the existence of missing values (NULL data) in any of our surrogate keys. This was done because, if one value was missing, we would have no way of identifying who or what was the concrete object of our analysis. Moreover, for the "Dim_Calendar" and "Fact_Sales" tables, we decided not to allow missing values in any of the remaining variables. For "Dim_ Calendar", we did it because we would most likely be needing all information in the future, but also because, since we created this dimension manually (as explained in the "Dim_Calendar" Dataflow chapter ahead), the presence of a missing value would mean some kind of error from our side when creating it, something we wanted to control and avoid. For the "Fact_Sales" table, we did not allow for the presence of any missing data in the variables that were not surrogate keys (Quantity and Amount) because this was the central table for our analysis, that would connect all dimensions and include the associated measures. As such, we would only want to work with values that were not missing.

Finally, we also added commented code to drop the tables if needed. This was done because of the possible need (which actually occurred) of redefining certain aspects in the tables and in the script, which would be much more complicated if we were unable to fully delete the tables we already had defined for the data warehouse and create them again.

Having successfully created the tables, the following step was to populate them. To do this, we created several dataflows (one for each table of our dimensional model) that would get the data that was stored in the Lakehouse, transform it according to our needs (and in accordance with

the SQL script created), and then connect it to the respective data warehouse table, still empty up to this point (and still empty until we ran the pipeline). The following subchapters detail how each dataflow was developed.

### 4.2.1. DIM_LOCATION DATAFLOW

For this first dataflow, we started by getting the data from the Lakehouse and, in specific, from the "Location" CSV file that we had previously uploaded. From here, the first step was to filter out the rows where the surrogate key was missing. This was done because the last rows of the file were just empty rows, and we wanted our data to be as clean as possible. Next, we changed the first column's ("SK_Location") data type, from text (which had been automatically assigned by Fabric) to a whole number. Then, we created a new column, that was simply an aggregation of two others that we already had: "City" and "Country". As we were exploring preliminary possibilities for creating a complete dashboard, we noticed that some maps required the location to be passed in the form "City, Country" in order to identify the correct place. As such, we created the column to fill this requirement, naming it "City_Country". Lastly, we renamed the columns to match the field names defined in the SQL script. This was done for every dataflow to facilitate adding the connection.

One step we opted not to take was the removal of the "District" column, which had almost every value missing. We decided to keep it in case there is the future need to elaborate a detailed analysis at a district level and this field is required. As filling these values is, despite burdensome, relatively straightforward (having the city, we can easily find its district), we believed that it would be worth keeping this variable and just disregard while it is not necessary.

Before publishing the dataflow, the final step was to add a data destination, which, in our case, was the data warehouse (this is where we want to store our transformed data) we had just created. To do this, we looked for the table relative to this dimension ("Dim_Location") and simply confirmed that all variables were matching, confirming the connection, and moving on to the following dataflow.

### 4.2.2. DIM_STORE DATAFLOW

Fairly like the first dataflow, for "Dim_Store" we started by getting data from the Lakehouse. However, in this case we had 2 CSV files, one for "Stores A" and another for "Stores B". Despite being 2 different files, with information about different stores on each file, these correspond to the same dimension. As such, the only solution was to merge them. Before performing the merge, we needed to address some situations.

Firstly, we loaded the 2 CSV's and promoted first row to headers on both "Stores"' tables. Then, we realized that we had a duplicated ID, which was "20", in Stores A and B. However, the one on "Stores B" seemed like a typo, because it appeared after ID "29" and before ID "31". We assumed this ought to be actually "30", so we made the necessary adjustment by replacing this duplicated ID with the value "30".

Moreover, we found 2 missing values, for "Stores A", in the column "Online" and one for "Stores B" in the same column. We assumed that, if a value was missing, it was because the store does not have online presence. Hence, these were replaced with "N". Also in the same column, we

had “S” and “Y”. Both these mean that there is Online Presence, however one is in Portuguese and the other in English. To uniformize, we replaced “S” with “Y”.

Next, we found out that the 2 bottom rows for “Stores B” were nulls, so we removed them. In addition, we verified that “Store ID” in “Stores B” had a decimal data type, which should be whole number. With this in mind, we made this necessary update.

After all these transformations were performed, we finally appended “Stores B” to “Stores A”. Lastly, we added a data destination, which, once again, was the data warehouse. The changes were into the existing corresponding Dimension (“Dim_Store”) and simply mapped all variables accordingly.

### 4.2.3. DIM_POINT_OF_SUPPLY DATAFLOW

The “Dim_Point_of_Supply” dataflow started similarly to the others, by importing the corresponding table from the Lakehouse. Then, since we got our data from a CSV file, we had to define the first row as the header to our table. This allowed Fabric to correct the data types and assign each column to the correct one. Next, we removed the email variable, which we had already identified as irrelevant for analysis, and renamed the columns according to the names defined upon the creation of the tables in the data warehouse. Finally, we connected the dataflow to its corresponding table in the warehouse, confirmed that each column was matching the right one in the destination, and published the dataflow.

### 4.2.4. DIM_CALENDAR DATAFLOW

This was the most unique dataflow, since it was created entirely from scratch. Here, we created a table with a date interval, starting with the first record on our fact table and ending with the last record. In our specific case, we started in 01/01/2023 and ended in 31/12/2023, progressing at a daily level (with each row representing a day). After this, we converted the data type to date type. Throughout this stage, we always made sure that the data types were correct and aligned with the SQL script.

Then, we started creating the rest of the needed columns we defined on our dimensional model. We extracted the day, month, year, and quarter of each date. This was simply done since Power Query already enables us to do this automatically by extracting this from the date.

Next, we created the columns with the written names and abbreviations of each month and quarter. This was done in two ways. For months, Power Query can detect automatically and simply add a column with the month name, and, for the abbreviation, we just added a column with the “first characters” option. For quarters, this was done with conditional column option, were based on the quarter number, we assign a name to it, and the same for the abbreviation.

Finally, we created the “SK_Date” column using a custom function, which we specified to build a surrogate key with the year, month, and day, combining them into a unique YYYYMMDD value.

In order to make things simpler, we also named each column according to their corresponding name in the SQL script that created the data warehouse. This facilitated our last step, which was to add a connection between the dataflow we had created and the “Dim_Date” table of the data warehouse.

### 4.2.5. DIM_OPERATOR DATAFLOW

For this dataflow, we started by getting the data from the "Operators" CSV in the Lakehouse and promoted the first row to the heading of our table, which simultaneously allowed for the correct identification of the data type of each column. Next, we grouped the "First_Name" and "Last_Name" columns into a single one (which we named "Full_Name"), as it was very unlikely that, at any point of our analysis, we would be interested in having only the first or the last name of an employee. Afterwards, we removed the name columns that were no longer needed ("First_Name" and "Last_Name"), as well as the email, for the same reason as pointed in the "Dim_Point_of_Supply" dataflow.

Then, we added a new conditional column to our data, which we called "Team_Name". For the employees whose "Team_ID" was 1, we assigned them to the team "Directors"; for "Team_ID" 2, we assigned them to the value "Floor Level"; finally, the operators whose "Team_ID" was 3 were assigned to the "Store Staff" team name. This classification was based on our analysis of the values in the "Operator_Role" column. We then discarded the "Team_ID" column, as we found more useful to use the team name rather than its unique identifier.

The last steps were about renaming the columns to match the names in the table of the data warehouse and reordering the columns (simply to have it more organized, we moved the "Full_Name" column to immediately after "SK_Operator", instead of having it as the last column). We added the connection to the "Dim_Operator"'s table in the data warehouse, published the dataflow, and moved to the following one.

### 4.2.6. DIM_PRODUCT DATAFLOW

For the last dimension, we started by grabbing the data from the Lakehouse and setting the first row as the header to our table, as we had done for most of the previous dataflows. We noticed that there were duplicated rows in the dataset (including the last one, which was repeated 6 times), so we handled this situation by selecting all four columns and filtering out the duplicate values. It is important to note that we only removed rows where all four values, as a combination, were duplicated. Next, we filtered out rows where the "SKU" was missing. Like in the "Dim_Location" dataflow, this was done because the last row of the file was just an empty row, and we wanted our data to be as clean as possible. After this, we had to create the "SK_Product" column to uniquely identify each row, as we had already concluded that, despite also being unique, "SKU" should just be used as a business key. Then, we reordered the columns, to place the newly created "SK_Product" as the first column of the table and renamed all fields to match their correspondents in the table from the data warehouse. Finally, we added the destination to connect the dataflow and the warehouse, verified that all columns were correctly assigned, and published the dataflow.

### 4.2.7. FACT_SALES DATAFLOW

Due to data integration, granularity, and modelling purposes, the fact table must be the last one to be done.

Firstly, we imported our sales data from the Sales CSV stored in or Lakehouse, and also from our "Dim_Calendar" and "Dim_Product" dataflows created before. Then, we promoted the first row

to headers. We were having some trouble with dates, so we had to make some small adjustments so that, whenever we wanted to merge the "SK_Date" column from "Dim_Calendar", it would be possible.

The next step was to perform a left outer join between Sales and "Dim_Calendar". This join was made with the Date column, and we selected "SK_Date" as our desired column to add. We did a similar process with "Dim_Products", performing again a left outer join, this time using the column "SKU", and selected the "SKU" column to be added to the fact table. However, in this case, the merge was unsuccessful in one row, meaning that we had a missing value on our Sales CSV, so we decided to drop that row, since it would not be very impactful.

The following step was to align the fact table with the changes made on the "Dim_Store" dataflow, where if the "SK_Store" was 20 and "SK_Location" was 5, then we would need to change the value of "SK_Store" to 30. This was done using a custom conditional value replacement function made by us, since Power Query didn't have a direct way of doing it.

Finally, we identified a row with an extreme value of 9 000 005 for the "Amount" variable. We assumed this to be an error entering the data and, as such, we removed it. Next, we made sure all data types were correct and aligned with our SQL query and also deleted the columns that would not be needed, to make things cleaner. We made sure the column names were the same as our SQL query for the "Fact_Sales", added the data destination, and with this we finalized the creation of all the dataflows in our project.

We were now prepared to move to the next phase, the creation of the data pipeline.

## 4.3 CREATING THE PIPELINE

Our final step was to create a data pipeline. Before explaining what we did step by step, we will give a small introduction on what data pipelines are.

A data pipeline defines a sequence of activities that translates into a process. It is a method where raw data is ingested from different data sources, transformed in some way, and finally ported to a data lake or a data warehouse, with the main goal of analysis. Pipelines are commonly used to perform processes automatically (extract, transform and load). Inside of the pipeline, we incorporate dataflows. There are several types of data pipelines – in our case, we are working with a data integration pipeline, often related to ETL, where processes that enrich, clean, and modify data are performed before storage in a centralized space.

Now, returning to our project, the pipeline called "RETAIL4ALL_PIPELINE" started with two scripts, for cleaning data from both facts and dimensions. The script used was similar to the one presented during classes, which performs a delete command to our warehouse, dropping all data.

Next, we introduced a control flow called "Wait" with a 1 second wait time before proceeding to the data loading phase. It is good practice to introduce a wait flow between each main phase of the pipeline, to ensure that the all the previous steps are finalized before continuing to the next one.

Then, we inserted all the dimension dataflows created previously. After testing, we came across some problems related to the software, whereby simply retrying those issues would be resolved, so we defined that in case of failure, the pipeline would retry 2 times to reload the dataflows. After success, another wait flow was introduced, for the same reasons as before.

Dimensions need to come before the facts because we need the data to be already inside our pipeline to retrieve it to build our fact table. This is the natural order of the process and makes everything run in a smother and organized way.

The final step was to load the data to the fact table, by using another dataflow, this time the "Fact_Sales" dataflow. Here, we finalized the data load and were ready for the next steps of this project.

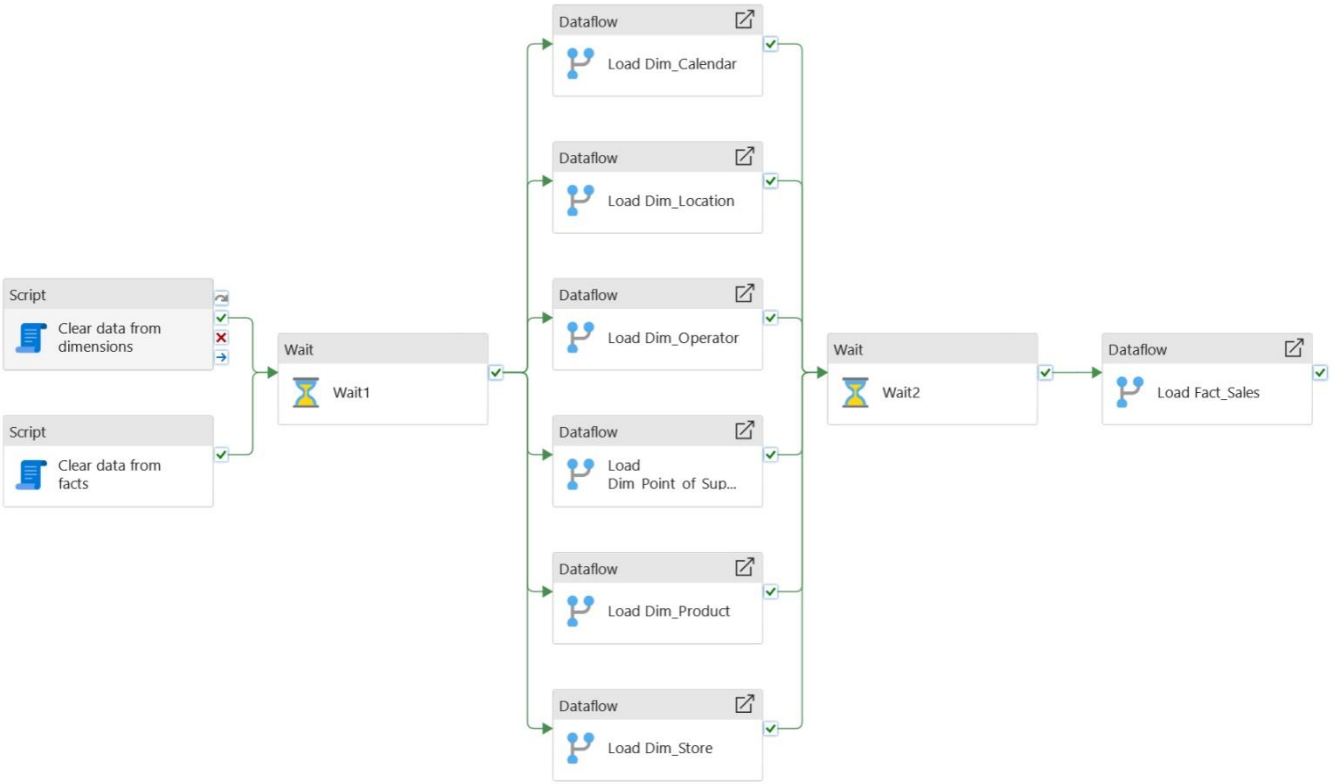


**Figure 10** - Final Data Pipeline

# 5. MODEL OPTIMIZATION

Before proceeding to the reporting task itself, we needed to optimize the tables we had at our disposal. This was done with the help of the semantic model, which, in short, would be the "single source of truth" for our report, containing all the necessary relationships to connect the tables that resulted from the previous steps, as well as hierarchies and other necessary transformations to make our analysis more accurate, complete, and appealing to the user. However, before using the semantic model, we had to create it.

In the data warehouse, we started by creating two semantic models. The first one, the default semantic model, was a more general model that included all tables with business data, excluding only those that were related to log events and other administrative tasks. In the second, which we named "SM Sales", we only included the tables that were relevant to the analysis in the context of this project and the defined business questions. In this situation, since we were working in an academic environment with data provided for that purpose, all the business tables that we had were important to consider, meaning that both the default and the "SM Sales" semantic models contained the same information (tables). Therefore, both had the "Dim_Location", "Dim_Store", "Dim_Point_of_Supply", "Dim_Calendar", "Dim_Operator", "Dim_Product" and "Fact_Sales" tables. However, if we were, for instance, to develop a future project using the same data, but whose focus was solely on the relationship between Retail4all's staff and sales, we could create a new semantic model that would include only the "Dim_Operator" and "Fact_Sales" tables.

Having created the semantic model on which our report would be based, we proceeded to transform and optimize it. Opening the data model in "SM Sales", the first step we took was to finally define the relationships between our dimensions and the fact table, which, up to this point, were not yet connected. This was the step that ensured that, from the fact table, we would be able to directly access more detailed information regarding each dimension. The process of creating the relationships was relatively straightforward: all we had to do was drag a foreign key in the fact table to its corresponding surrogate key in its corresponding dimension (for example, dragging the "FK_Location" in "Fact_Sales" to "SK_Location" in the "Dim_Location" table). This approach automatically created a one-to-many relationship from the dimension tables to the fact, exactly as we intended. This is because a dimension should have each instance appearing only once, but that same instance could appear multiple times in the fact table – using again the example for locations, each should appear only once in the "Dim_Location" table, but we wanted to allow for it to appear many times in "Fact_Sales", as we can have multiple sales happening in the same location.

After the relationships, we created the hierarchies in the semantic model, respecting those that we had previously defined. This step enabled us to more easily drilldown our analysis later in the report. In the "Dim_Calendar" table, we defined our first hierarchy with the variables "Year", "Quarter_Name", "Month_Name" and "Day_of_Month", which we named "Date Hierarchy". Regarding the "Dim_Product" table, our "Product Hierarchy" contained the columns "Category", "Subcategory" and "Product_Name". In the "Dim_Location" table, the created "Location Hierarchy" included the "Country", "District" and "City" attributes. Finally, we added our "Sellers Hierarchy" to the "Dim_Operator" table, using the "Team_Name", "Operator_Role" and "Full_Name" columns.

The following task was to hide columns that would be irrelevant for analysis, in order to simplify the variables to be shown when creating the report. Thus, we decided not to display none of the surrogate (for dimension tables) or foreign keys (for the fact table), as their purpose was to define instances and create relationships between tables, but not to be analysed individually. Additionally, we also chose to hide the "SKU" column in the "Dim_Product" table, which was also representing a (business) key, as well as the "District" column in the "Dim_Location" table. This latter variable was hidden because, as we previously stated, it mostly consists of missing values and was kept for an eventual future case where a district analysis is relevant and its values are filled, which doesn't match the current situation. Moreover, we opted to hide the columns that we had already included in the hierarchies previously created. Even though it was not strictly necessary, we decided to do it to avoid unnecessarily "duplicating" these attributes, which would otherwise appear both inside the hierarchy and on their own.

We then proceeded to mark our "Dim_Calendar" table as a date table – which would allow us to later create and apply time intelligence measures on it – before moving to the phase of renaming the columns and tables in our data. Until this moment, they were presented in a more computerized and mechanical format, but we wanted to display them in a more human-like shape. Therefore, besides replacing the underscore symbol with a space, we made the following changes in our variables' names:

| ID | (New) Table Name | DW Alias | Is Key? | Friendly Name |
|---|---|---|---|---|
| 1 | Dim Location | SK_Location | Key | |
| 2 | Dim Location | City | | City |
| 3 | Dim Location | District | | District |
| 4 | Dim Location | Country | | Country |
| 5 | Dim Location | City_Country | | City & Country |
| 6 | Dim Store | SK_Store | Key | |
| 7 | Dim Store | Store_Name | | Store |
| 8 | Dim Store | Online | | Online |
| 9 | Dim Point of Supply | SK_POS | Key | |
| 10 | Dim Point of Supply | Supplier_Name | | Supplier |
| 11 | Dim Calendar | SK_Date | Key | |
| 12 | Dim Calendar | Full_Date | | Date |
| 13 | Dim Calendar | Year | | Year |
| 14 | Dim Calendar | Quarter_Number | | Quarter Number |
| 15 | Dim Calendar | Quarter_Name | | Quarter |
| 16 | Dim Calendar | Quarter_Name_Short | | Abrev. Quarter |
| 17 | Dim Calendar | Month_Number | | Month Number |
| 18 | Dim Calendar | Month_Name | | Month |
| 19 | Dim Calendar | Month_Name_Short | | Abrev. Month |
| 20 | Dim Calendar | Day_Number | | Day of Month |
| 21 | Dim Operator | SK_Operator | Key | |
| 22 | Dim Operator | Full_Name | | Operator Full Name |
| 23 | Dim Operator | Gender | | Gender |
| 24 | Dim Operator | Operator_Role | | Operator Role |
| 25 | Dim Operator | Team_Name | | Team Name |

| ID | (New) Table Name | DW Alias | Is Key? | Friendly Name |
|---|---|---|---|---|
| **26** | Dim Product | SK_Product | Key | |
| **27** | Dim Product | SKU | | SKU |
| **28** | Dim Product | Product_Name | | Product |
| **29** | Dim Product | Category | | Category |
| **30** | Dim Product | Subcategory | | Subcategory |
| **31** | Fact Sales | FK_Date | Key | |
| **32** | Fact Sales | FK_Location | Key | |
| **33** | Fact Sales | FK_Product | Key | |
| **34** | Fact Sales | FK_Store | Key | |
| **35** | Fact Sales | FK_Operator | Key | |
| **36** | Fact Sales | FK_POS | Key | |
| **37** | Fact Sales | Quantity | | Sales Quantity |
| **38** | Fact Sales | Amount | | Sales Amount |

After this step, we performed a minor background change in two variables of the "Dim Calendar" table, "Month" and "Quarter". By default, these (and all other) columns were being sorted based on their own values. Being text columns, this meant that they were being sorted alphabetically. However, this should not be the case for these two specific attributes, as they have a sequence that doesn't necessarily match the alphabetical order. Therefore, making use of the column properties, we defined "Month" to be sorted based on the "Month Number" variable, and "Quarter" to be ordered according to the values in "Quarter Number". Besides this transformation, we also defined the "Sales Amount" column in our "Fact Sales" table to be a currency-like decimal number, to match its "real-life" characteristics. This enforced the presence of two decimal places for the values for this column, and allowed us to specify the currency format, which we defined to be "€ Euro (€ 123)".

Following these steps, we were finally ready to move to Power BI, in order to start building our report.

## 6. BUILDING THE REPORT

After all the preparatory steps we took up to this point, we arrived at the most crucial one in this project. With a well prepared and presented report, we would allow business users to make data-driven decisions and answer the business questions that they are interested in finding out more about, pushing the organization forward by providing adequate analytical tools. Therefore, this stage was crucial to define the value that our work can provide to Retail4all.

The initial step was, naturally, to add the data to be analysed in the report. To achieve this, all we had to do was to connect our previously created semantic model to the report document. With this, we will only have to refresh the connection if we want to incorporate more data into our analysis in the future (as long as this data is related to the same variables we currently have).

We should also note that, throughout our ETL process, we had created calculated columns according to our needs for the project, which can be seen in more detail in the chapter dedicated to the creation of the dataflows. Nonetheless, in our case, these involved the aggregation of two columns in some form, and can be summarized as follows:

| Original Name | New Name | Explanation | Justification |
|---|---|---|---|
| City_Country | City & Country | Joins the "City" and "Country" columns, representing them in a "City, Country" format. | Some map visuals we attempted could not assign the correct location unless data was passed in this format. |
| Full_Name | Operator Full Name | Joins the "First_Name" with the "Last_Name" of an operator. | As earlier mentioned, we are not interested in the first or last name of an operator, but rather on its full name. Besides being more correct and respectful for the worker, this also reduces the risk of having duplicated information (for instance, two employees with the same surname). |
| Team_Name | Team Name | We assigned each value in the "Team_ID" column to a corresponding team name, based on our understanding from the data. | We wanted to analyse sales at the level of the teams who performed them, but it was more difficult to see this if the team was represented by a number. Therefore, we assigned "Team_ID" 1 to "Directors", 2 to "Floor Level", and 3 to "Store Staff". |

Moreover, we created several measures to integrate in our report, according to what we found relevant to analyse and, sometimes, according to the needs of the visualizations we intended to perform. In essence, they represent dynamic calculation formulas, built using the DAX language, whose output is dependent on the context surrounding it. Despite most measures being associated with the fact sales table, we also created some related to the operators' dimension, which can be described as follows (its formulas can be found in Annex 1):

| Measure Name | Explanation |
|---|---|
| Number of Operators | This measure counts the number of different operators that there are in Retail4all and was used for building one of the 3 KPI's in the operators' dashboard of the report. |
| Number of Roles | This measure counts the number of different roles that there are in Retail4all and was used for building one of the 3 KPI's in the operators' dashboard of the report. |
| Number of Teams | This measure counts the number of different teams that there are in Retail4all and was used for building one of the 3 KPI's in the operators' dashboard of the report. |

On the other hand, regarding the measures that were incorporated in the sales' fact table, they can be explained as shown below. The corresponding formulas can be found in Annex 1:

| Measure Name | Explanation |
|---|---|
| Filter | This measure is used to create a dynamic table displaying the filters applied to the current page. |
| Filter Header | This measure is used to create a header where we display the header of the applied filters. This way the user has a better understanding of what is filtering the page. |
| Measure – CY Quantity | This measure calculates the quantity of items sold in the current year. It's used in several visualizations to compare current year vs previous year. |
| Measure – CY Sales | This measure calculates the sales amount of the items sold in the current year. It's used in several visualizations to compare current year vs previous year. |

| Measure Name | Explanation |
|---|---|
| Measure – PY Quantity | This measure calculates the quantity of items sold in the previous year. It's used in several visualizations to compare current year vs previous year. |
| Measure – PY Sales | This measure calculates the sales amount of the items sold in the previous year. It's used in several visualizations to compare current year vs previous year. |
| Measure – Qty YTD | This measure calculates the year-to-date quantity of items sold, enabling us to see how the quantity of items sold has evolved since the start of the year. |
| Measure – Sales YTD | This measure calculates the year-to-date sales amount of items sold, enabling us to see how the amount of items sold has evolved since the start of the year. |
| Measure – Total Quantity Data | This measure calculates the total sum of the quantity of items sold during the period of the analysis. This measure was needed to create current year and previous year measures, as well as some visuals. |
| Measure – Total Sales Data | This measure calculates the total sum of the sales amount of items sold during the period of the analysis. This measure was needed to create current year and previous year measures, as well as some visuals. |
| Measure – YoY Quantity Variance | This measure calculates the difference between the current year and the previous year, in this case for quantity of items sold. |
| Measure – YoY Sales Variance | This measure calculates the difference between the current year and the previous year, in this case for the sales amount of items sold. |
| Measure – Quantity Amount YoY% | This measure calculates the difference between the current year and the previous year, in this case for quantity of items sold. For this measure the result is returned in percentage. |
| Measure – Sales Amount YoY% | This measure calculates the difference between the current year and the previous year, in this case for the sales amount of items sold. For this measure the result is returned in percentage. |
| G1 Measure – Label pos left | This measure is used to display the correct label on the left side of the graph. This is a measure made specially for a visual and to help the user understand better the data. It will check if CY Sales is smaller than PY Sales to then display the data on the correct side. |
| G1 Measure – Label pos right | This measure is used to display the correct label on the right side of the graph. This is a measure made specially for a visual and to help the user understand better the data. It will check if CY Sales is bigger than PY Sales to then display the data on the correct side. |
| G1 Measure – Line Size Negative | This measure is purely cosmetic and is only used to increase the size of the line, by multiplying the variation by a constant. It does not affect any analysis, only enhances the visual. |
| G1 Measure – Line Size Positive | This measure is purely cosmetic and is only used to increase the size of the line, by multiplying the variation by a constant. It does not affect any analysis, only enhances the visual. |
| G1 Measure – Sales Variance Centre | This measure is purely cosmetic and only used to set a dynamic value so that it draws a reference line on a regular vertical histogram enabling the rest of the visual to be built. |
| G1 Measure – Sales Variance Centre Dummy | This measure is purely cosmetic and copies the measure "G1 Measure – Sales Variance Centre", enabling the rest of the visual to be built. |
| G1 Measure – YoY Sales Variance neg | This measure is the base for the "G1 Measure – Line Size Negative" measure, and it returns the year-over-year sales variance, for the values where current year sales are smaller than previous year sales. |

| Measure Name | Explanation |
|---|---|
| G1 Measure – YoY Sales Variance pos | This measure is the base for the "G1 Measure – Line Size Positive" measure, and it returns the year-over-year sales variance, for the values where current year sales are bigger than previous year sales. |
| G2 Measure – Label pos left | This measure was created with the same goal as "G1 Measure – Label pos left" but for quantity of items sold. |
| G2 Measure – Label pos right | This measure was created with the same goal as "G1 Measure – Label pos right" but for quantity of items sold. |
| G2 Measure – Line Size Negative | This measure was created with the same goal as "G1 Measure – Line Size Negative" but for quantity of items sold. |
| G2 Measure – Line Size Positive | This measure was created with the same goal as "G1 Measure – Line Size Positive" but for quantity of items sold. |
| G2 Measure – Quantity Variance Centre | This measure was created with the same goal as "G1 Measure – Sales Variance Centre" but for quantity of items sold. |
| G2 Measure – Quantity Variance Centre Dummy | This measure was created with the same goal as "G1 Measure – Sales Variance Centre Dummy" but for quantity of items sold. |
| G2 Measure – YoY Quantity Variance neg | This measure was created with the same goal as "G1 Measure – Sales Variance neg" but for quantity of items sold. |
| G2 Measure – YoY Quantity Variance pos | This measure was created with the same goal as "G1 Measure – Sales Variance pos" but for quantity of items sold. |
| G3 Measure – Data Bar Sales vs Target | This measure will draw a progress bar, showing our status regarding the completeness of out target sales. It's used to provide the user a visual representation of the progress, for a quick and intuitive analysis. |
| G3 Measure – Sales vs Target | This measure calculates how far away our current sales are from the target defined on the "G3 Measure – Target Sales". |
| G3 Measure – Target Sales | This measure defines the hypothetical target sales. In our case we didn't have a target, but we wanted to show the ability to display it, so we created a constant target of 10% more than the value of sales. In this specific case, it's a display of ability rather than analysis, but can easily be a valuable tool if adapted for the data provided. |
| G4 Measure – Data Bar Qty vs Target | This measure was created with the same goal as "G3 Measure – Data Bar Sales vs Target" but for quantity of items sold. |
| G4 Measure – Qty vs Target | This measure was created with the same goal as "G3 Measure – Qty vs Target" but for quantity of items sold. |
| G4 Measure – Target Qty | This measure was created with the same goal as "G3 Measure – Target Qty" but for quantity of items sold, but in this case the goal is 15%. |

With these measures, we were able deepen our ability to convey the most relevant information in the most adequate way for the business user to gain practical insights from the data. Over the next sub-chapters, we will explain our rationale behind the creation of each dashboard (page) of the report, as well as some of the most valuable conclusions they allowed us to take. Then, we will aggregate what we were able to observe from our visuals and answer the business questions earlier presented as the challenge for Retail4all's business. Our ability to do this in a direct, complete, and accurate manner will ultimately be the determinant of the success of our project.

## 6.1. HOME PAGE

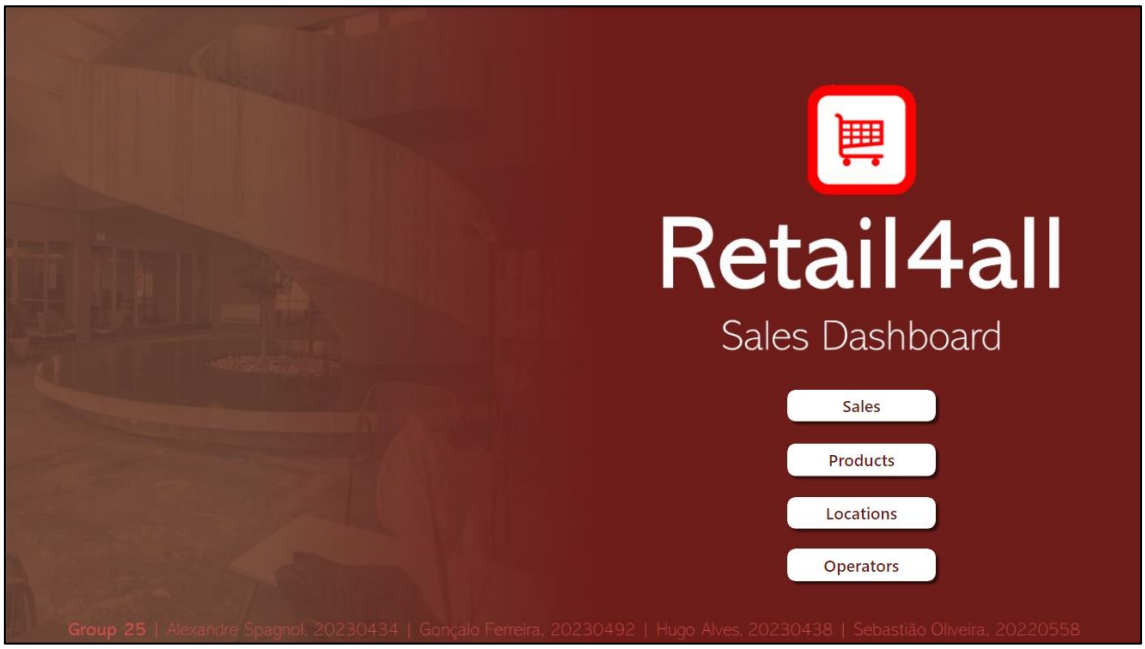


**Figure 11** - Report's Home Page

To answer the business questions we defined for this project, we decided to create four different pages (dashboards) in our report, related to the sales, the products being sold, the locations Retail4all sells in, and the operators who make these sales.

In addition to the pages we will be exploring, we created a simple home page to be the starting point of the report, and to allow for an easier navigation through the file, as we can see in the figure above. From here, the users learn what are the pages that they can explore and can click on the button referring to the page that they want to see, being then immediately directed towards it.

## 6.2. SALES' PAGE

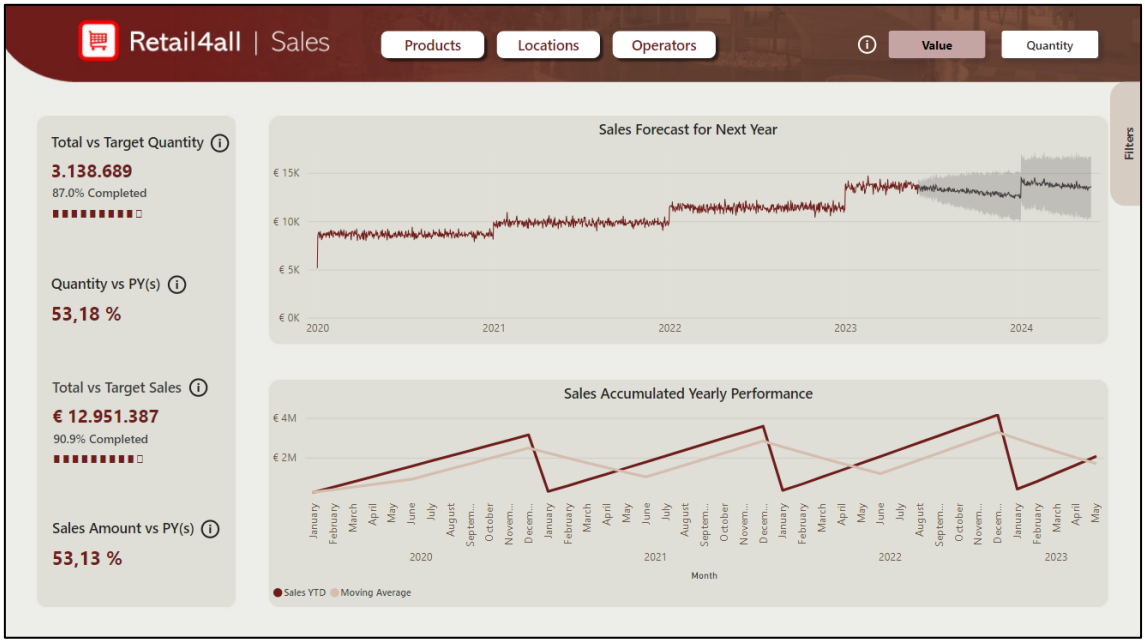


**Figure 12** - Report's Sales Page

Starting from the top, the header of this dashboard displays the buttons that allow the user to be immediately directed to the remaining pages of the report. Additionally, by clicking on Retail4all's logo, the user will be conducted to the home page shown above. This was done with the objective of simplifying the navigation throughout the document and was done for every page of the report (expect for the home page, already displayed earlier).

The header also displays two buttons to switch the focus of analysis of the page between sales quantity and amount. Where applicable, this will change the visualizations according to the intended measure, respecting the necessities of the user. A small information tooltip is present next to the buttons to shortly explain how to interact with them.

In addition, each page displays a hidden filter tab on the top-right corner. By clicking on in, the user can filter the data shown in that dashboard according to "Calendar" (year, quarter, and month), "Stores" (if it is an online seller or not, its location, and the name of the store) and "Product" (by product category, subcategory, and the product itself). This allows for a more detailed analysis if the user desires so, and is a much more direct way to control the information shown in its visuals. On top of these filters, a small window displays all filters currently selected for that page, along with a button to immediately clear all filters that are active. This pane required the use of both "Filter" measures described earlier.

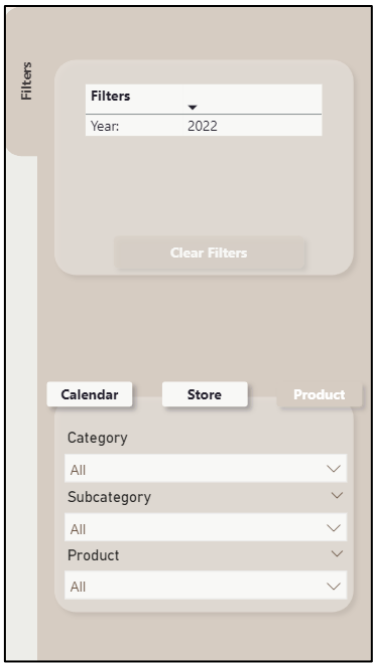


**Figure 13** - Filters Tab

Analysing specifically the sales' page, it starts by showing 4 Key Performance Indicators (KPI's) on the left-hand side. Two of them are dedicated at showing the total sales in quantity and amount, and how they compare against a fictional target that we defined (in this case, we set as a target 15% and 10% above any value that was presented, respectively, simply to demonstrate that the visual would work). An information icon was added to let the user know what this target is for each case. Below the main number of the KPI, highlighted in red, we added a percentage of completion towards the target, as well as a customized progress bar to make evident how far off completion Retail4all is in these two metrics. This required the use of some of our created measures, namely those whose name start with either "G3" or "G4", but also "Quantity Amount YoY%" and "Quantity Amount YoY%". By looking at these visuals, users can immediately see that Retail4all achieved overall sales of more than 3 million units worthing almost 13 million euros. In both cases, the numbers represent approximately 90% of the target sales, meaning that the company is on a good path in this matter. Naturally, these numbers can change according to the filters the user chooses to select.

The 2 remaining KPI's are intended to showcase the quantity and value of sales compared to the previous year. If no year is selected by the user, then the comparison will be between the most and least recent moments of data. An information icon was also placed next to the visual's title to explain the user how to interpret it. From here, we can see that, between the first and the latest year of available data, sales have increased a little over 53%, in both quantity and amount.

To the right of the KPI's, a line chart was added to reflect how sales have behaved over time in either value or quantity, depending on the selection of the user in the header of the page. This graphic was chosen as it is the most adequate to reflect the evolution of a measure in a time perspective. In addition, we added a forecast for the following year of analysis, defining a yearly seasonality since, in the beginning of every year, sales register a considerable jump in both quantity and amount. We also added a shadow around the forecast, representing the variability of the 95% confidence interval we defined for our prediction. The forecast line was also chosen

to be in a different, less vibrant colour than the known past sales, to make it clear that they do not represent actual data but rather a prediction that was made for future times. In addition, we added a light horizontal gridline to the graphic, so that the user could more easily compare sales against the reference points in the y-axis. We should also note that we made this graphic independent from any selected filters, in order to avoid showcasing forecasts for periods of known data (otherwise, if we selected the year of 2021, for example, we would see predictions for the year of 2022, which is not intended as we have the actual values for this moment). From this chart, we can directly see that sales constantly increase inter-years (increasing more as time moves forward), but are relatively steady within each calendar year.

To the bottom, another line chart was added, this time representing sales yearly accumulated performance (either in quantity or amount, depending on the selection in the header of the dashboard). To do this, we utilized the "YTD Qty" and "YTD Sales" measures earlier described. Moreover, a moving average line was added in a lighter colour tone to represent the average sales of the previous 6 months, at a certain point of the chart. A legend was added to the bottom left of the visual, so as to make sure the users can understand what each line represents. From here, we can take similar conclusions to the previous visualization, making evident a very steady growth within each year with no major variations between months or quarters, which the moving average line also reflects.

## 6.3. PRODUCTS' PAGE

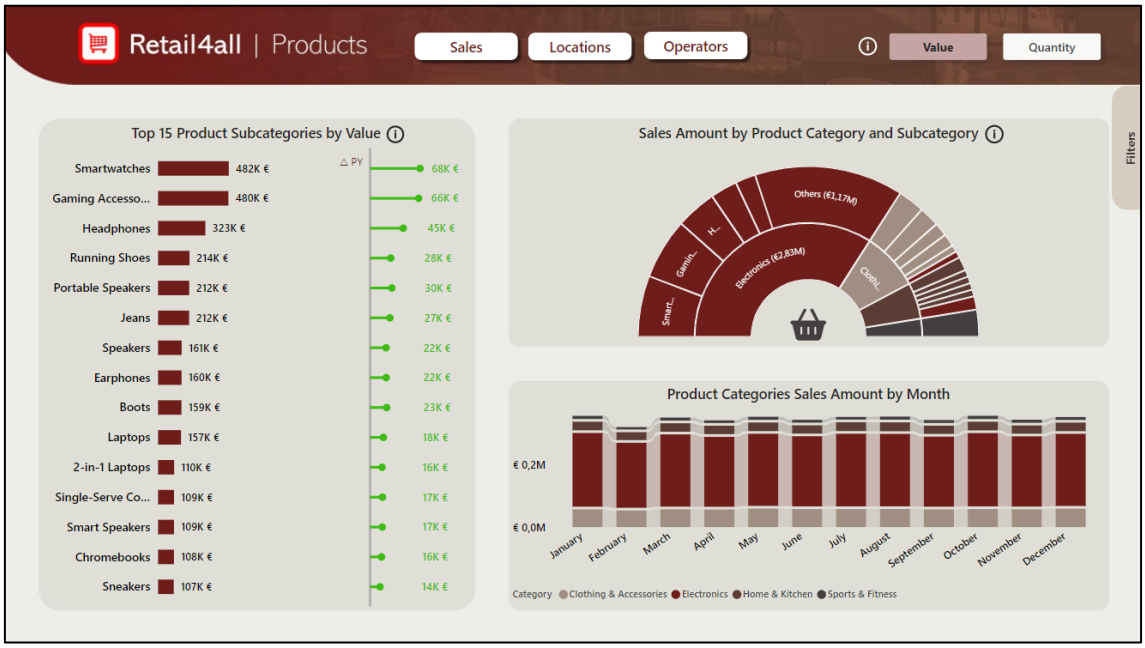


**Figure 14** - Report's Products Page

On the products' dashboard, we started by developing an advanced clustered bar chart that returned the 15 most popular product subcategories according to sales quantity or amount (depending on the selected button on the right of the header). To complement it, we also created a display of comparison between the selected year (which was 2022 as the default for this page) and the previous one. If the variation was positive (which means Retail4all sold more than in the previous year), a green bar would show the difference in absolute terms. Otherwise, the bar

would turn red (and would be to the left side of the vertical grey line), displaying the absolute value of the decrease in sales (in either euros or units sold). An information tooltip was placed next to the title to indicate how to view and comprehend the information in this visualization. In order to build this chart, we employed a vast group of measures out of those that were described earlier, in specific those started with "G1" and "G2", but also "CY Quantity", "CY Sales", "PY Quantity" and "PY Sales". Moreover, we made this visualization independent from the remaining visuals in the same page, so as not to have the selection of other categories affecting this one. In terms of practical conclusions, this visual makes evident that the most popular product subcategories have grown in 2022 in comparison to 2021, with smartwatches, gaming accessories and headphones being the ones that registered the most sales (both in quantity and amount), but also those that grew the most in comparison to the previous year.

Next, we built a sunburst chart, an external Power BI visualization displaying the breakdown of the sales per product category and subcategory. Additionally, as the information icon also explains, by hovering over each group, a personalized tooltip will appear, showcasing the total sales quantity and value of that same group (either category or subcategory), as well as their 5 most popular products and their sales quantity and amount. We opted for a chart covering only 180 degrees rather than the usual 360 of a sunburst chart, as it was more practical to analyse, but also better to fit in our dashboard and don't compromise space utilization. Moreover, the colours employed for this graphic were carefully chosen to match the report's palette, while also being easily distinguishable among themselves and from the background. A basket was added for aesthetic purposes. This chart allows the user to see the predominance of electronics as the most popular product category in the year of 2022, with its 5 most sold products being very similar to each other in terms of sales quantity and value.

Lastly, we wanted to assess how the structure of sales for Retail4all changed across the 12 months of the year, in terms of product category. For that purpose, we chose a stacked column chart, placing the month of the selected year in the x-axis to show the distribution of sales amount over that period. Additionally, the product categories were displayed in the same colours as the sunburst chart above, to make it clear that we were dealing with the same entities (even though a legend was still added to make this even more evident). From the visual, we see that electronics account always for the biggest percentage of sales, not being significantly affected by the month of the year.

## 6.4. LOCATIONS' PAGE

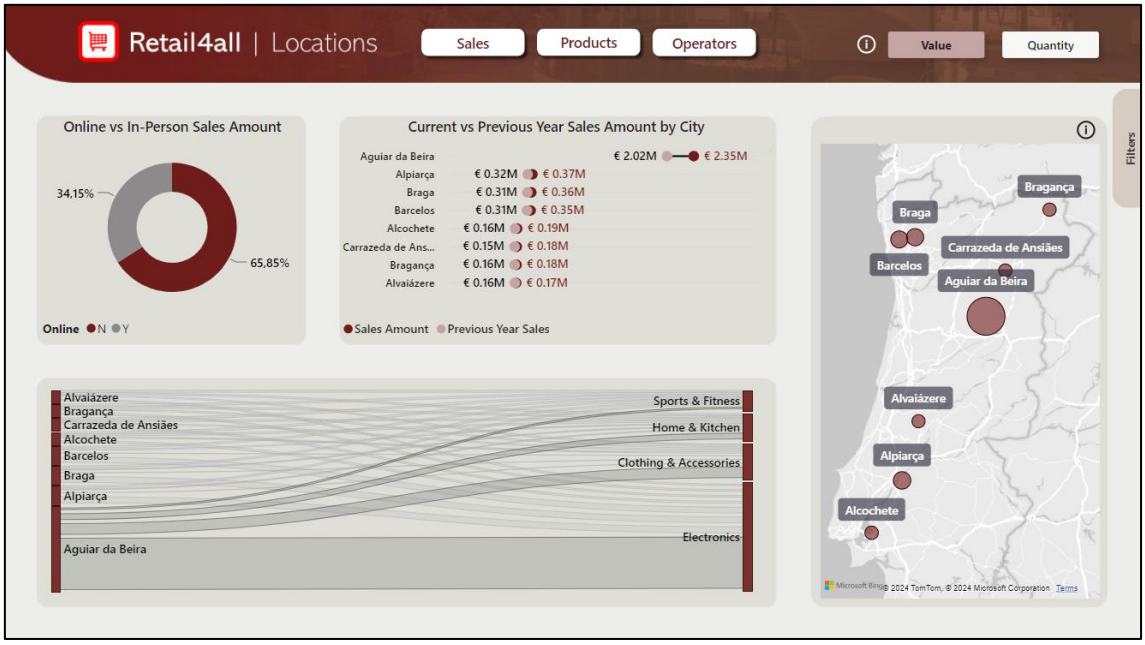


**Figure 15** - Report's Locations Page

Moving to the locations' page, we started by using a donut chart to showcase the percentage of sales (in amount) that were made online against those that were made in-person. By playing with the filters on the tab on the right-hand side of the dashboard, we are able to see that this distribution was relatively steady throughout the years of data that we had available, with almost two thirds of sales being made in-person.

To the right of the donut chart, a dumbbell visualization was created to show the variation of sales (in either amount or quantity, depending on the selection in the header) throughout the different locations where Retail4all is present. While the circles related to the previous year were coloured in lighter tones, the ones related to the currently selected year (by default, 2022) were coloured in a darker tone of red, as they represented the most recent and up-to-date information. Nonetheless, a legend was displayed on the bottom left of the chart, so that the user could clearly understand how to look at this visual. For this graphic, the "PY Quantity" and "PY Sales" measures were necessary to display the information related to the previous year. From here, the user can see that Aguiar da Beira is, by far, the most popular location for Retail4all's sales (both in value and quantity), while also being the most growing one in absolute terms throughout the years.

To the bottom, we represented the relationship between a location and its most sold product categories (in quantity or amount, depending on the selection in the header) with the help of a Sankey diagram where the weights of the flow between two entities was dictated by the selected measure. By hovering over each flow, we enabled the user to see how much (or how many units) had been sold of a product category in a certain location. We used a darker tone of grey to highlight the most popular location, Aguiar da Beira, due to its relevance in the total sales of Retail4all. From this visualization, we can this town accounts for around half of the sales of electronics (the most popular product category), meaning that, perhaps, the people from this location are more interested in new technologies than other customers.

Finally, we used a map to show the distribution of sales throughout the Portuguese territory, with the bubble size being defined in accordance with the sales amount of a specific location. In addition, a tooltip was added to complement the analysis – by hovering over a location, the total sales quantity and amount would be displayed, as well as its 5 most popular stores and their contribution to the total sales value and units sold in each place. From this graphic, we enabled the users to see that most sales occur in the north region of Portugal, but also that there is a lack of presence in the south regions of the country. Moreover, we can also see that all stores tend to have very similar sales, meaning that a possible justification for Aguiar da Beira being the most popular location can simply be due to the fact that it actually has more store than the other cities.

## 6.5. OPERATORS' PAGE

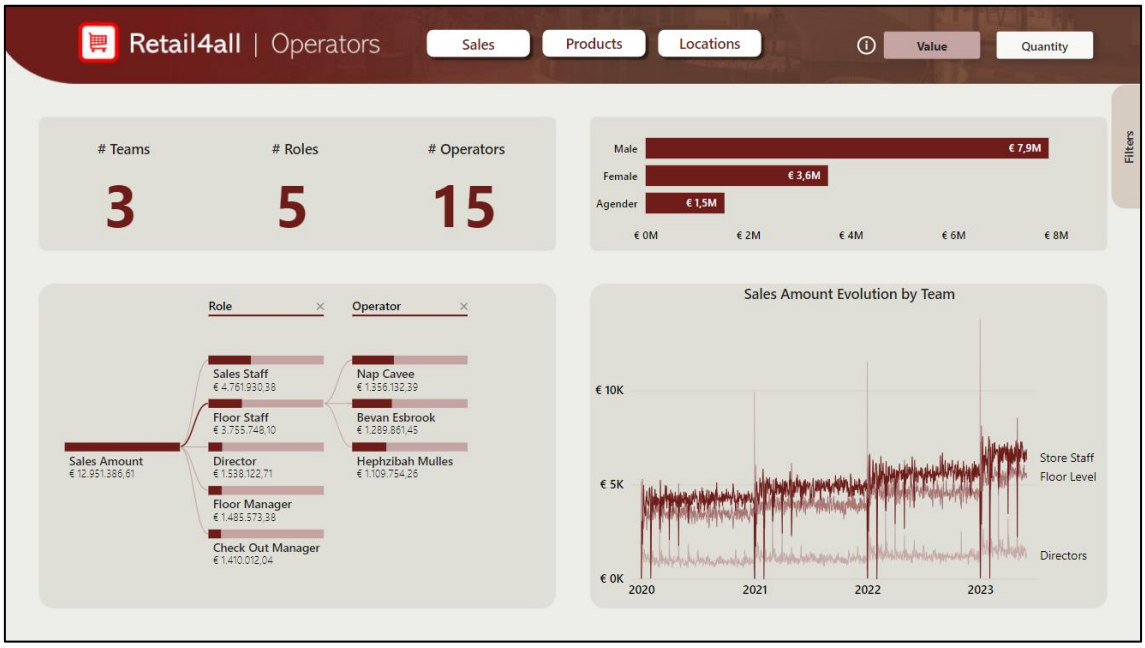


**Figure 16** - Report's Operators Page

Finally, for the operators' page, we started by displaying 3 KPI's related to the number of teams, roles, and employees in Retail4all. These KPI's make use of the 3 measures created regarding the operators, earlier described. To the right, a bar chart was developed to represent the sales by gender (in either quantity or value, depending on the selection in the header). We added the labels of each bar for improved readability and placed them inside the bar it related to. This graphic showed that most sales were made by men – however, by clicking on this bar, we saw in the KPI's visual that there were 7 male operators out of a total of 15, meaning that, most likely, people tend to sell the same independently of their gender.

Next, we used a decomposition tree to enable the analysis of the structure of sales across the sellers' hierarchy, in either value or quantity terms. Here, each bar is filled according to how much each instance represents in the total of the branch they originated from, and it is a very direct way to compare different groups in the same level of the detail. Nonetheless, we should note that this visualization isn't ideal if many new roles are added or if many new operators are hired by Retail4all, as this would complicate the analysis by making more difficult to convey this

information in a single view. However, for the time being, it is more than suitable for the company's purposes. As for practical conclusions is concerned, this tree shows that Sales Staff are the role that sells the most, which is justifiable considering the characteristics of this role.

To the right of the decomposition tree, we added a line chart displaying the evolution of sales quantity or amount (depending on the selection in the header page) and how they variate for each team in the organization. This graphic is independent from the others in this page, remaining unchanged if the users select something specific in other visuals – this was done with the purpose of avoiding losing the ability to use this graphic to compare performance between teams, which could occur if only one was being shown. In this visual, we see that the evolution seems to follow similar patterns throughout the years for each team, but we can denote 2 significant increases every year in Store Staff, on one of the first days of January and the other on one of the first days of February. Additionally, directors record an abnormal value and quantity for sales on the on the 28$^{th}$ of December of every year.

For this dashboard, we should note that we considered adding a sub-page to analyse in more detail the performance of each individual operator. However, in this case, the intended objective and subsequent business questions arise more from a strategic and tactical context rather than from an operational perspective. Therefore, we decided to not follow this way so as to not deviate from the purpose of our report, which is to improve the overall decision-making process at Retail4all. If this analysis of individual employees becomes a necessity, a separate report could be built to facilitate this analysis, using, for example, a dedicated semantic model fitted to that intent.

## 6.6. ANSWERING THE BUSINESS QUESTIONS

After building the dashboard, we are finally able to answer the business questions that resulted from the challenge and necessities of Retail4all. We can summarize them in the following table:

| ID | Business Question | Answer | Main Source Visual |
|---|---|---|---|
| 1 | What is our number of sales across distinct time periods (month, quarter, and year)? | While a summary answer cannot be provided to the nature of the question (we would have to explicit the sales for each month, quarter, and year of data), the report allows the user to analyse this information. Overall, we can say that the quantity of sales increases every year but is very steady within quarters and months of the same year. | Line charts in the sales' page, when analysing data by quantity. |
| 2 | How is our accumulated performance in sales to date? | € 12 950 000 | "Total vs Target Sales" KPI in the sales' dashboard. |
| 3 | How have we evolved compared to the previous year's sales performance? | In 2022 (the most recent year with full available data), sales increased 16.02% in quantity and 15.86% in amount in comparison to 2021. If we take a look at the most and least recent years of data, we see that sales increased around 53%, in both quantity and value. | "Quantity vs PY(s)" and "Sales Amount vs PY(s)" KPIs in the sales' dashboard. |

| ID | Business Question | Answer | Main Source Visual |
|---|---|---|---|
| 4 | How much money are we making every month? | Similarly to the first question, it is not possible to summarize an answer to this question, but the report allows the user to visualize this information. Overall, sales amount follows similar patterns to the sales quantity. | Line charts in the sales' page, when analysing data by value. |
| 5 | What products are our customers buying more? And when? | Electronics, with no particular seasonality. Within these, smartwatches, gaming accessories and headphones are the ones that registered the most sales and that are growing the most, but all products tend to have similar weights in the total of sales. | Sunburst and stacked column chart in the products' page, when analysing data by quantity. |
| 6 | Who are the employees responsible for the most sales? | Consistently, sales staff are the role that sells the most (with its employees having similar weights in the total sales). However, the director has a peak on the 28th of December of every year, vastly surpassing any other role. | Decomposition tree and line chart in the operators' page. |
| 7 | What categories of products are most sold in each location (by sales quantity)? | Since we have many locations and 4 product categories, it is difficult to give a summary answer to this question. However, taking Aguiar da Beira as an example, we see that Electronics are, by far, the predominant category of products, with Sports & Fitness having a very residual weight. | Sankey diagram in the locations' page, when analysing by quantity. |
| 8 | What are the locations where we have sold the most? | Aguiar da Beira. | Map in the location's page. |
| 9 | What channel has a bigger weight in our sales (online vs face-to-face)? And how is it evolving? | In-person sales are the most common scenario, representing almost two thirds of the sales. Interestingly, this percentage is almost constant throughout the years. | Donut chart in the location's page of the report. |

From our report, Retail4all's administration can increase the quality and confidence of its decision-making process, and can implement strategies that aim to address the opportunities and obstacles that the analysis enables to visualize. To give some examples, special marketing techniques can be employed to each location, aiming to increase sales according to the shopping patterns of each place – for instance, knowing that Aguiar da Beira is the most popular town in terms of sales, and that is mostly sells electronic products, special campaigns can be developed, even though they don't necessarily apply to other places with different customer behaviours. Additionally, Retail4all can try to focus on the ever-growing online shopping area, in order to increase its weight in total sales and be able to reach more clients that are not necessarily those that buy in store.

# 7. CONCLUSION

This project arose from the necessity of an analytical tool to be at Retail4all's disposal, in order for its administrators to improve the organization's analytical output and, consequently, solidifying its decision-making process. Making use of the information made available, related to about 3 years of historic data, we started by learning more about the business and, from there, propose the business questions that our work should enable to answer. Next, we designed our dimensional model in accordance with the data available and the defined business questions – we identified 4 different hierarchies and the 6 main dimensions on our data, but we also created the sales fact table, on which the main measures for analysis were presented. Then, we were ready to start the ETL process, using Microsoft Fabric and its applications for that purpose. We started by loading our raw data to a Lakehouse, before creating the warehouse that would store our structured and transformed tables. The dataflows were created to transform the tables according to our needs, by changing or removing the initial columns and adding others more relevant for our project, and were later integrated into a pipeline to automate the ETL process and allow for the data to be loaded correctly. The following step involved the optimization of the model, creating a semantic model for that intent. Here, we changed data types and column names to a more human-like format, identifying our calendar dimension as a date table, and created the hierarchies and relationships that, up to this moment, had not yet been passed into the data. We also hid some attributes that would have no purpose to be analysed, and changed the way some columns were sorted (in specific, those related to the month and quarter of the year). Finally, we built the report that materialized the possibility for analytical decisions. By building relevant visualizations in 4 different dashboards (plus a home page), integrated into an easy to navigate and well-structured report, we were able to successfully and clearly answer the business questions, deeming our project as a beneficial tool for its stakeholders at Retail4all.

In the future, and as was previously mentioned, other reports can be built to analyse other aspects at a more operational level, which was not the main purpose of this project. In addition, more data could be integrated, either coming from the organization itself or from external sources, with the goal of allowing for an even more complete analysis and more relevant insights for its users. Nonetheless, we are very confident that our proposed solution is a strong start for Retail4all's business intelligence practices, with a solid project from the early stages of understanding the business until finally presenting a report that users can understand and interact with, pushing the company forward to be a powerful competitor in the Portuguese retail market.

# 8. ANNEXES

## 8.1. ANNEX 1 | MEASURES' FORMULAS IN DAX

| Measure Name | Formula |
|---|---|
| Number of Operators | `DISTINCTCOUNT('Dim Operator'[Operator Full Name])` |
| Number of Roles | `DISTINCTCOUNT('Dim Operator'[Operator Role])` |
| Number of Teams | `DISTINCTCOUNT('Dim Operator'[Team Name])` |
| Filter | `Filter =`<br>`-- Online Shop`<br>`IF(`<br>`ISFILTERED('Dim Store'[Online]),`<br>`VAR items = VALUES('Dim Store'[Online])`<br>`VAR itemscombined = CONCATENATEX(items, 'Dim Store'[Online], UNICHAR(10))`<br>`RETURN itemscombined & UNICHAR(10)`<br>`) &`<br>`-- Store Name`<br>`IF(`<br>`ISFILTERED('Dim Store'[Store]),`<br>`VAR items = VALUES('Dim Store'[Store])`<br>`VAR itemscombined = CONCATENATEX(items, 'Dim Store'[Store], UNICHAR(10))`<br>`RETURN itemscombined & UNICHAR(10)`<br>`) &`<br>`-- Store Location`<br>`IF(`<br>`ISFILTERED('Dim Location'[City]),`<br>`VAR items = VALUES('Dim Location'[City])`<br>`VAR itemscombined = CONCATENATEX(items, 'Dim Location'[City], UNICHAR(10))`<br>`RETURN itemscombined & UNICHAR(10)`<br>`) &`<br>`-- Year`<br>`IF(`<br>`ISFILTERED('Dim Calendar'[Year]),`<br>`VAR items = VALUES('Dim Calendar'[Year])`<br>`VAR itemscombined = CONCATENATEX(items, 'Dim Calendar'[Year], UNICHAR(10))`<br>`RETURN itemscombined & UNICHAR(10)`<br>`) &`<br>`-- Quarter`<br>`IF(`<br>`ISFILTERED('Dim Calendar'[Quarter]),`<br>`VAR items = VALUES('Dim Calendar'[Quarter])`<br>`VAR itemscombined = CONCATENATEX(items, 'Dim Calendar'[Quarter], UNICHAR(10))`<br>`RETURN itemscombined & UNICHAR(10)`<br>`) &`<br>`-- Month`<br>`IF(`<br>`ISFILTERED('Dim Calendar'[Month]),`<br>`VAR items = VALUES('Dim Calendar'[Month])` |

| | |
|---|---|
| | VAR itemscombined = CONCATENATEX(items, 'Dim Calendar'[Month], UNICHAR(10))<br>RETURN itemscombined & UNICHAR(10)<br>) &<br>-- Product Category<br>IF(<br>ISFILTERED('Dim Product'[Category]),<br>VAR items = VALUES('Dim Product'[Category])<br>VAR itemscombined = CONCATENATEX(items, 'Dim Product'[Category], UNICHAR(10))<br>RETURN itemscombined & UNICHAR(10)<br>) &<br>-- Product Subcategory<br>IF(<br>ISFILTERED('Dim Product'[Subcategory]),<br>VAR items = VALUES('Dim Product'[Subcategory])<br>VAR itemscombined = CONCATENATEX(items, 'Dim Product'[Subcategory], UNICHAR(10))<br>RETURN itemscombined & UNICHAR(10)<br>) &<br>-- Product<br>IF(<br>ISFILTERED('Dim Product'[Product]),<br>VAR items = VALUES('Dim Product'[Product])<br>VAR itemscombined = CONCATENATEX(items, 'Dim Product'[Product], UNICHAR(10))<br>RETURN itemscombined & UNICHAR(10)<br>) |
| Filter Header | Filter Header =<br>-- Online Shop<br>IF(<br>ISFILTERED('Dim Store'[Online]),<br>"Online: " & REPT(UNICHAR(10),COUNTROWS(VALUES('Dim Store'[Online])))<br>) &<br>-- Store Name<br>IF(<br>ISFILTERED('Dim Store'[Store]),<br>"Store: " & REPT(UNICHAR(10), COUNTROWS(VALUES('Dim Store'[Store])))<br>) &<br>-- Store Location<br>IF(<br>ISFILTERED('Dim Location'[City]),<br>"City: " & REPT(UNICHAR(10), COUNTROWS(VALUES('Dim Location'[City])))<br>) &<br>-- Year<br>IF(<br>ISFILTERED('Dim Calendar'[Year]),<br>"Year: " & REPT(UNICHAR(10), COUNTROWS(VALUES('Dim Calendar'[Year])))<br>) &<br>-- Quarter<br>IF(<br>ISFILTERED('Dim Calendar'[Quarter]),<br>"Quarter: " & REPT(UNICHAR(10), COUNTROWS(VALUES('Dim Calendar'[Quarter])))<br>) &<br>-- Month |

| | |
|---|---|
| | IF(<br>ISFILTERED('Dim Calendar'[Month]),<br>"Month: " & REPT(UNICHAR(10), COUNTROWS(VALUES('Dim Calendar'[Month])))<br>) &<br>-- Product Category<br>IF(<br>ISFILTERED('Dim Product'[Category]),<br>"Category: " & REPT(UNICHAR(10), COUNTROWS(VALUES('Dim Product'[Category])))<br>) &<br>-- Product Subcategory<br>IF(<br>ISFILTERED('Dim Product'[Subcategory]),<br>"Subcategory: " & REPT(UNICHAR(10), COUNTROWS(VALUES('Dim Product'[Subcategory])))<br>) &<br>-- Product<br>IF(<br>ISFILTERED('Dim Product'[Product]),<br>"Product: " & REPT(UNICHAR(10), COUNTROWS(VALUES('Dim Product'[Product])))<br>) |
| Measure – CY Quantity | Measure - CY Quantity =<br><br>Var CY = MAX('Dim Calendar'[Year])<br><br>RETURN<br>CALCULATE([Measure - Total Quantity Data], 'Dim Calendar'[Year]=CY) |
| Measure – CY Sales | Measure - CY Sales =<br><br>Var CY = MAX('Dim Calendar'[Year])<br><br>RETURN<br>CALCULATE([Measure - Total Sales Data], 'Dim Calendar'[Year]=CY) |
| Measure – PY Quantity | Measure - PY Quantity =<br><br>CALCULATE([Measure - Total Quantity Data], SAMEPERIODLASTYEAR('Dim Calendar'[Date])) |
| Measure – PY Sales | Measure - PY Sales =<br><br>CALCULATE([Measure - Total Sales Data], SAMEPERIODLASTYEAR('Dim Calendar'[Date])) |
| Measure – Qty YTD | Measure - Qty YTD =<br>TOTALYTD([Measure - CY Quantity], 'Dim Calendar'[Date]) |
| Measure – Sales YTD | Measure - Sales YTD =<br>TOTALYTD([Measure - CY Sales], 'Dim Calendar'[Date]) |

| | |
|---|---|
| Measure – Total Quantity Data | `Measure - Total Quantity Data = SUM('Fact Sales'[Sales Quantity])` |
| Measure – Total Sales Data | `Measure - Total Sales Data = SUM('Fact Sales'[Sales Amount])` |
| Measure – YoY Quantity Variance | `Measure - YoY Quantity Variance =`<br>`[Measure - CY Quantity] - [Measure - PY Quantity]` |
| Measure – YoY Sales Variance | `Measure - YoY Sales Variance =`<br>`[Measure - CY Sales] - [Measure - PY Sales]` |
| Measure – Quantity Amount YoY% | `Measure - Quantity Amount YoY% =`<br>`VAR __PREV_YEAR =`<br>`CALCULATE(`<br>`SUM('Fact Sales'[Sales Quantity]),`<br>`DATEADD('Dim Calendar'[Date], -1, YEAR)`<br>`)`<br>`RETURN`<br>`DIVIDE(SUM('Fact Sales'[Sales Quantity]) - __PREV_YEAR, __PREV_YEAR)` |
| Measure – Sales Amount YoY% | `Measure - Sales Amount YoY% =`<br>`VAR __PREV_YEAR =`<br>`CALCULATE(`<br>`SUM('Fact Sales'[Sales Amount]),`<br>`DATEADD('Dim Calendar'[Date], -1, YEAR)`<br>`)`<br>`RETURN`<br>`DIVIDE(SUM('Fact Sales'[Sales Amount]) - __PREV_YEAR, __PREV_YEAR)` |
| G1 Measure – Label pos left | `G1 Measure - Label pos left =`<br>`IF (`<br>`[Measure - CY Sales] < [Measure - PY Sales],`<br>`[G1 Measure - Sales Variance Centre] + [Measure - YoY Sales Variance]`<br>`)` |
| G1 Measure – Label pos right | `G1 Measure - Label pos right =`<br>`IF (`<br>`[Measure - CY Sales] > [Measure - PY Sales],`<br>`([G1 Measure - Sales Variance Centre] + [Measure - YoY Sales Variance])*1.2`<br>`)` |
| G1 Measure – Line Size Negative | `G1 Measure - Line Size Negative = [G1 Measure - YoY Sales Variance neg]*1` |
| G1 Measure – Line Size Positive | `G1 Measure - Line Size Positive = [G1 Measure - YoY Sales Variance pos]*5` |
| G1 Measure – Sales Variance Centre | `G1 Measure - Sales Variance Centre =`<br>`MAXX(ALLSELECTED('Dim Product'[Subcategory]), [Measure - CY Sales])*3` |

| | |
|---|---|
| G1 Measure – Sales Variance Centre Dummy | `G1 Measure - Sales Variance Centre Dummy = [G1 Measure - Sales Variance Centre]` |
| G1 Measure – YoY Sales Variance neg | `G1 Measure - YoY Sales Variance neg =`<br>`if(`<br>`[Measure - CY Sales] < [Measure - PY Sales],`<br>`[Measure - YoY Sales Variance]`<br>`)` |
| G1 Measure – YoY Sales Variance pos | `G1 Measure - YoY Sales Variance pos =`<br>`if(`<br>`[Measure - CY Sales] > [Measure - PY Sales],`<br>`[Measure - YoY Sales Variance]`<br>`)` |
| G2 Measure – Label pos left | `G2 Measure - Label pos left =`<br>`IF (`<br>`[Measure - CY Quantity] < [Measure - PY Quantity],`<br>`[G2 Measure - Quantity Variance Centre] + [Measure - YoY Quantity Variance]`<br>`)` |
| G2 Measure – Label pos right | `G2 Measure - Label pos right =`<br>`IF (`<br>`[Measure - CY Quantity] > [Measure - PY Quantity],`<br>`([G2 Measure - Quantity Variance Centre] + [Measure - YoY Quantity Variance])*1.4`<br>`)` |
| G2 Measure – Line Size Negative | `G2 Measure - Line Size Negative = [G2 Measure - YoY Quantity Variance neg]*1` |
| G2 Measure – Line Size Positive | `G2 Measure - Line Size Positive = [G2 Measure - YoY Quantity Variance pos]*5` |
| G2 Measure – Quantity Variance Centre | `G2 Measure - Quantity Variance Centre =`<br>`MAXX(ALLSELECTED('Dim Product'[Subcategory]), [Measure - CY Quantity])*2.6` |
| G2 Measure – Quantity Variance Centre Dummy | `G2 Measure - Quantiry Variance Centre Dummy = [G2 Measure - Quantity Variance Centre]` |
| G2 Measure – YoY Quantity Variance neg | `G2 Measure - YoY Quantity Variance neg =`<br>`if(`<br>`[Measure - CY Quantity] < [Measure - PY Quantity],`<br>`[Measure - YoY Quantity Variance]`<br>`)` |
| G2 Measure – YoY Quantity Variance pos | `G2 Measure - YoY Quantity Variance pos =`<br>`if(`<br>`[Measure - CY Quantity] > [Measure - PY Quantity],`<br>`[Measure - YoY Quantity Variance]`<br>`)` |
| G3 Measure – Data Bar Sales vs Target | `G3 Measure - Data Bar Sales vs Target =`<br>`VAR _NrIcons_Total = 10`<br>`VAR _PctOfTarget = MIN(DIVIDE([Measure - Total Sales Data], [G3 Measure - Target Sales]),1)`<br>`VAR _NrIcons_Fill = _PctOfTarget * _NrIcons_Total`<br>`VAR _NrIcons_Empty = _NrIcons_Total - _NrIcons_Fill` |

| | |
|---|---|
| | ```<br>    Var _IconFill = "■"<br>    VAR _IconEmpty = "□"<br><br>    VAR _DataBar =<br>        REPT(_IconFill, _NrIcons_Fill) & REPT(_IconEmpty, _NrIcons_Empty)<br><br>RETURN<br>    _DataBar<br>``` |
| G3 Measure – Sales vs Target | ```<br>G3 Measure - Sales vs Target =<br>VAR _Value = FORMAT(DIVIDE([Measure - Total Sales Data], [G3 Measure - Target Sales]), "0.0%")<br>VAR _Pretex = "Completed"<br>VAR _Space = " "<br>VAR _Label =<br>    _Space & _Value & _Space & _Pretex<br>RETURN<br>    _Label<br>``` |
| G3 Measure – Target Sales | ```<br>G3 Measure - Target Sales = SUM('Fact Sales'[Sales Amount])*1.10<br>``` |
| G4 Measure – Data Bar Qty vs Target | ```<br>G4 Measure - Data Bar Qty vs Target =<br>    VAR _NrIcons_Total = 10<br>    VAR _PctOfTarget = MIN(DIVIDE([Measure - Total Quantity Data], [G4 Measure - Target Qty]),1)<br>    VAR _NrIcons_Fill = _PctOfTarget * _NrIcons_Total<br>    VAR _NrIcons_Empty = _NrIcons_Total - _NrIcons_Fill<br><br>    Var _IconFill = "■"<br>    VAR _IconEmpty = "□"<br><br>    VAR _DataBar =<br>        REPT(_IconFill, _NrIcons_Fill) & REPT(_IconEmpty, _NrIcons_Empty)<br><br>RETURN<br>    _DataBar<br>``` |
| G4 Measure – Qty vs Target | ```<br>G4 Measure - Qty vs Target =<br>VAR _Value = FORMAT(DIVIDE([Measure - Total Quantity Data], [G4 Measure - Target Qty]), "0.0%")<br>VAR _Pretex = "Completed"<br>VAR _Space = " "<br>VAR _Label =<br>    _Space & _Value & _Space & _Pretex<br>RETURN<br>    _Label<br>``` |

| | |
|---|---|
| G4 Measure – Target Qty | `G4 Measure - Target Qty = SUM('Fact Sales'[Sales Quantity])*1.15` |