BRP Report

Data Quality Analyst

Kallil de Araujo Bezerra

October 21, 2023

Contents

1 Task 1

The first part of the assignment is to read the data sent and do a quick analysis, to understand what this data is about. There are two files, both in *.csv* format.

- Date fixes: the data in both files present some date inconsistencies. For example, they use '-' and '/' to separate month, day, and year. This may cause some confusion when a script tries to determine what type of data the file has.

- Special Character fixes: in order to keep the data consistent, it's important to change some special characters. When sending data between different systems, so from a Microsoft Server to a Power BI dashboard, or to a .csv file it's important to have the data to be shown the same way, this cannot be guaranteed if special characters are involved. Therefore, a special character removal is an important step to keep consistency.

- Data validation: it is important to check if all our data is valid. For example, checking if all cities in 'CITY_DEALER' are cities that do exist and also are cities in which BRP has a dealer. The same should be applied to to 'STATE_CODE'. Another step is to make sure that the cities follow a standard and are not duplicated. We can find 'Alexandria Bay' and 'Alexandria Bay,', despite being the same city, it could be considered 2 different cities without a data cleansing.

- Check for duplicated records: duplicated records can affect how managers will take their decisions because they may represent the same event twice. This can come in two different ways, either by having 2 identical records on the dataset, or by having 2 different records that represent the same thing. The last one can be caused by problems like cities with different names but that represent the same place, similar to what was described in the previous topic.

- Check for missing values in critical columns: check if there aren't any missing values in ID columns for example. In the files under analysis I believe that 'REG_DEALER_NUMBER' and 'MODEL_NUMBER' are important columns that can't have null or empty values, therefore it's important to check if they are complete.

Date Standardization: Ensure consistent date formats by converting all

date entries to a standardized format (e.g., YYYY-MM-DD). This standardization prevents format-related issues when processing the data.

Special Character Cleanup: Remove special characters from data entries to maintain consistency when transferring data across different systems, such as from a Microsoft Server to Power BI or a CSV file.

Data Validation and Standardization: Perform data validation to verify the accuracy and consistency of location-related data. Ensure that cities and state codes are valid, adhere to standard naming conventions, and eliminate duplicated or similar but distinct entries (e.g., 'Alexandria Bay' and 'Alexandria Bay,') to avoid data duplication.

Duplicate Record Detection: Detect and handle duplicated records, which can distort analysis and decision-making. This can involve identifying identical records or reconciling records representing the same entity with variations in data entry.

Missing Value Assessment: Check critical columns for missing values. Columns such as 'REG_DEALER_NUMBER' and 'MODEL_NUMBER' should be complete, as they are important identifiers. Address any null or empty values to maintain data integrity.

- item 1
- item 2
- item 3
  - sub item 1
  - sub item 2
  - sub item 3
- item 4
  1. passo 1
  2. passo 2
  3. passo 3

1.1 Most expensive products at the company

To analyze the most expensive products in the schema, it was necessary to order them by their prices. From most to less expensive. The query used can be seen below. The **LIMIT 11** was used because the last products cost the same (315.90), so I considered both as the 10^{th} position.

```
SELECT PRODUCT_NAME, PRODUCT_VAL
FROM looqbox_challenge.data_product
ORDER BY PRODUCT_VAL DESC
LIMIT 11;
```

The result can be seen in the image 1 or in the table **??**.

Figure 1: Data from task 1

PRODUCT_NAME	PRODUCT_VAL
Whisky Escoces THE MACALLAN Ruby Garrafa 700ml c...	741.99
Whisky Escoces JOHNNIE WALKER Blue Label Garrafa ...	735.90
Cafeteira Expresso 3 CORACOES Tres Modo Vermelho	499.00
Vinho Portugues Tinto Vintage QUINTA DO CRASTO G...	445.90
Escova Dental Eletrica ORAL B D34 Professional Care 5...	399.90
Champagne Rose VEUVE CLICQUOT PONSARDIM Garr...	366.90
Champagne Frances Brut Imperial MOET Rose Garrafa ...	359.90
Conjunto de Panelas Allegra em Inox TRAMONTINA 5 ...	359.00
Whisky Escoces CHIVAS REGAL 18 Anos Garrafa 750ml	329.90
Champagne Frances Brut Imperial MOET & CHANDON ...	315.90
Champagne Frances Demi Sec Nectar Imperial MOET &...	315.90

I tried to execute another query, using **RANK**. However, it did not work because of a server version issue. I am not sure why. The *better* query is the one below, it is more elegant than the one I presented.

```
SELECT RANK() OVER(ORDER BY PRODUCT_VAL DESC) ranked_products,
       PRODUCT_NAME,
       PRODUCT_VAL
FROM looqbox_challenge.data_product
WHERE ranked_products < 10;
```

1.2 Sections from selected departments

In the next task it is asked to analyze which sections the departments **BEBIDAS** and **PADARIA** have. To do this, the following query was written.

```
SELECT DISTINCT SECTION_NAME, SECTION_COD, DEP_NAME
FROM looqbox_challenge.data_product
WHERE (DEP_NAME LIKE 'BEBIDAS%' OR DEP_NAME LIKE 'PADARIA%')
ORDER BY DEP_NAME;
```

The result can be seen in the table 1.

# SECTION_NAME	SECTION_COD	DEP_NAME
BEBIDAS	4	BEBIDAS
CERVEJAS	29	BEBIDAS
VINHOS	30	BEBIDAS
REFRESCOS	31	BEBIDAS
DOCES-E-SOBREMESAS	8	PADARIA
PADARIA	19	PADARIA
QUEIJOS-E-FRIOS	22	PADARIA
GESTANTE	27	PADARIA

Table 1: Department and section analysis

1.3 Total sales in 2019

In this analysis it was assumed that Business Area could be interpreted as **BUSINESS_NAME**.

```
SELECT BUSINESS_NAME, SUM(SALES_VALUE)
FROM looqbox_challenge.data_store_sales d_sales
INNER JOIN looqbox_challenge.data_store_cad d_cad
           ON d_cad.STORE_CODE = d_sales.STORE_CODE
WHERE DATE BETWEEN '2019-01-01' AND '2019-04-01'
GROUP BY BUSINESS_NAME;
```

The result of this query can be seen in table 2. This option is ordered by the business' name. Another way to see the results is shown in the table 3, which is ordered by **SALES_VALUE**, therefore the ones with highest sales value will come first.

# BUSINESS_NAME	SUM(SALES_VALUE)
Atacado	81079295.20
Farma	82462460.37
Posto	32338509.96
Proximidade	80863761.30
Varejo	81733342.62

Table 2: Total sales by business area in the first quarter of 2019

# BUSINESS_NAME	SUM(SALES_VALUE)
Posto	32338509.96
Proximidade	80863761.30
Atacado	81079295.20
Varejo	81733342.62
Farma	82462460.37

Table 3: Total sales by business area in the first quarter of 2019 - ordered by sales value

2 Case 1 - Dynamic Function

3 Case 2 - Join queries

Two different queries were given, and I was asked to not modify the queries. The result must be in the following format: **Loja**, **Categoria**, and **TM**.

```
SELECT store_cad.STORE_NAME AS Loja,
       store_cad.BUSINESS_NAME AS Categoria,
       ROUND((store_sales.SALES_VALUE/store_sales.SALES_QTY),2) AS TM
FROM(
SELECT
      STORE_CODE,
      STORE_NAME,
      START_DATE,
      END_DATE,
      BUSINESS_NAME,
      BUSINESS_CODE
FROM looqbox_challenge.data_store_cad
) AS store_cad
JOIN (
SELECT
        STORE_CODE,
        DATE,
        SALES_VALUE,
        SALES_QTY
FROM looqbox_challenge.data_store_sales
WHERE DATE BETWEEN '2019-01-01' AND '2019-12-31'
) AS store_sales ON store_sales.STORE_CODE = store_cad.STORE_CODE
GROUP BY store_cad.STORE_NAME
ORDER BY store_cad.STORE_NAME;
```

4 Case 3 - Data visualization