Enhancing Data Quality: Customer Data Cleansing and Unification ETL in a Brazilian Healthcare Company.

Diogenes Vaz de Melo Oliveira [a]

[a] Institute of Exact Sciences (ICEx), Federal University of Minas Gerais (UFMG), Belo Horizonte, Minas Gerais, Brazil, diogenesvazmelo@gmail.com.

Abstract. This paper presents a detailed account of the data cleansing and unification process implemented in the customer database of a prominent Brazilian healthcare company. Focused on enhancing data quality, the study begins with an overview of data cleansing principles and the significance of accurate customer data in the healthcare industry. Leveraging a structured ETL (Extraction, Transformation, Loading) approach, the methodology involved comprehensive analysis of the customer dimension table, revealing numerous inconsistencies and errors. Utilizing tools such as SQL Server Management Studio (SSMS), SQL Server Integration Services (SSIS), and Apache Spark, cleansing transformations were applied to address issues ranging from special characters to data validation errors. Additionally, a Python algorithm was developed to address the unification of customer records, aiming to mitigate duplicate entries. The outcomes include the creation of a refined customer dimension table and a table containing suggested unifications, which serve as valuable resources for the company's data management and decision-making. Furthermore, the study discusses opportunities for improvement, particularly in the scalability and performance of the data processing environment, suggesting migration to a cloud computing service for enhanced efficiency. This study underscores the importance of systematic data cleansing and unification processes in improving data integrity, ultimately enhancing customer experiences in the healthcare sector.

Keywords. ETL, Customer Data, Data Cleansing, Data Unification.

1. Introduction

Data cleansing, also called data cleaning or scrubbing, deals with detecting and removing errors and inconsistencies from data in order to improve the quality of data [1]. Through this process, redundant, inaccurate, or incomplete data can be identified and corrected, ensuring that the database contains accurate and reliable information.

Implementing a data cleansing process in a healthcare company's customer database can significantly enhance data quality. In the healthcare industry, ensuring the integrity of customer data is crucial for providing high-quality patient care, streamlining operations, and maintaining regulatory compliance. By cleansing the data, the company can minimize errors in patient records, improve decision-making processes, and enhance overall efficiency in healthcare personalized customer experiences.

This paper outlines the cleansing of the customer dimension table of a large Brazilian healthcare company through the steps of data Extraction, Transformation, and Loading (ETL). Lucas, Raja and Ishfaq [2] define an ETL as a process in which multiple software tools are utilized for the extraction of data from several sources, their cleansing, customization and insertion into a data warehouse. In this instance, the primary software tools utilized encompassed the SQL Server Management Studio (SSMS), the SQL Server Integration Services (SSIS) and Apache Spark for large-scale data processing.

The present study aimed to achieve two primary objectives: firstly, to construct a refined customer dimension table characterized by accurate data types, validated information, and enhanced derived columns based on the original dataset. Secondly, the study sought to identify and address potential instances of duplicated customer entries within the database

2. Methodology

2.1 Data source

The initial step involved identifying and comprehensively analyzing the dimension table containing the company's customer data source. Dimension tables are usually wide, flat denormalized tables with many low-cardinality text attributes which are the primary target of constraints and grouping specifications from queries and BI applications [3]. The company's customer dimension table comprised over 10 million rows and 16 columns. Details regarding these 16 columns are provided in Table 1.

The examination of the data within the table unveiled numerous inconsistencies, including the presence of special characters and numerical values within fields such as FullName, City, and State. Additionally, non-numeric characters were found in fields like CustomerId, CPF (Brazilian individual taxpayer registry), and CEP (Brazilian ZIP code). Further irregularities comprised the improper usage of lowercase or uppercase letters, discrepancies in date formats, and deviations from standard "M" (for male) or "F" (for female) values in the Sex field. Invalid entries were also identified, such as incorrect CPF, CEP, and phone numbers, as well as invalid email addresses (e.g., lacking the "@" symbol), among other discrepancies.

Tab. 1 - Customer table column description.

Column	Data Type	Description
Customer Id	*varchar*	A sequence consisting of 1 to 18 digits that serves as an identifier for customer registration
FullName	*varchar*	Customer's full name
DateOf Birth	*varchar*	Customer's date of birth
Sex	*varchar*	Customer's sex, "M" for male and "F" for female (F)
Address	*text*	Customer's home address
CEP	*varchar*	Brazilian ZIP code (customer's address)
City	*varchar*	Brazilian city (customer's address)
Health Insurance Provider	*varchar*	If the customer has no health insurance, then "PART". Else, insurance company name
Registra tionDate	*varchar*	Registration date of the customer into the company's database
Last Modifica tionDate	*varchar*	The date of the most recent modification to any customer information
State	*varchar*	Brazilian state (customer's address)
CPF	*varchar*	11-digit Brazilian individual taxpayer number issued by the government
Email	*varchar*	Customer's e-mail
Phone Number	*varchar*	Customer's Brazilian phone number. It can store a maximum of two numbers separated by a slash
Source	*int*	System source indicator (unchanged during the cleansing process)

The source table has undergone daily updates, encompassing the insertion of new data and the updating of existing records. On a daily basis, all data from the source table has been extracted and subjected to the cleansing transformation process.

2.2 Cleansing transformation

To organize the collection of connections, control flow elements, data flow elements, event handlers, variables, parameters, and configurations, an SSIS project and package were created. The SSIS is a tool that facilitates data extraction, consolidation, and loading options (ETL), SQL Server coding enhancements, data warehousing, and customizations [4]. Within SSIS, the following transformations have been executed:

- All blank cells have been converted to *null*;
- All special characters including "\t", "\n", "\v", "\f", "\r", "¬", backslashes, as well as leading and trailing spaces, have been eliminated from all string fields;
- All consecutive sequences of space characters have been condensed to a single space;
- In order to standardize and facilitate comparisons, all strings have been converted to uppercase;
- All values other than "M" and "F" in the Sex field have been converted to *null*;
- All non-numeric characters have been removed from the columns CustomerId, CPF, and CEP;
- All non-alphanumeric characters, excluding spaces, have been removed from the fields FullName, City, and State;
- The PhoneNumber field has been split into two separate columns, PhoneNumber1 and PhoneNumber2;

- The Address field has been divided into the columns StreetName, StreetType, and StreetNumber;
- The FullName column has been split into FirstName and Surname;
- The CPF number and the e-mail have been validated;
- All the string columns have been converted to *nvarchar* (unicode) data type, to mitigate encoding erros;
- Boolean (SQL Server *bit* data type) columns have been added to indicate whether the values in the CustomerId, FullName, CPF, and Email fields were duplicated in the database.

The outcome has been stored in a new customer dimension table, maintaining an identical row count to the original.

2.3 Customer unification

After the cleansing process, the database still contained multiple records seemingly associated with the same customer. To resolve this issue, a Python algorithm was developed to compare similarities between each record in the table.

The algorithm utilized the Python API for Apache Spark, known as PySpark. PySpark is the Python API for Apache Spark, an open source, distributed computing framework and set of libraries for real-time, large-scale data processing [5]. For the purpose of reading the data in the algorithm, the entire content of the new customer dimension table has been automatically extracted and saved into a comma-separated values (CSV) file, the simplest and widest-spread format for the prevailing tabular datasets [6].

The CSV file has been stored in an on-premises server folder, and the file location was specified in the algorithm to enable reading and loading into a cluster distributed Spark dataframe.

In total, the algorithm comprised 15 categories, each written as a function with the purpose of comparing values between column rows. The categories and their respective criteria are detailed in Table 2.

Some of the categories, such as Categorie2 and Categorie3, utilize a similarity distance measure between two words described by Levenshtein [7]. The Levenshtein distance is a string comparison metric that counts the number of edit operations (replacements, insertions, and deletions) required to transform one string into another [8]. The implementation of the Levenshtein distance algorithm has been provided as a PySpark SQL built-in standard function.

Tab. 2 - Unification algorithm categories and criteria.

Category	Unification criteria
Category1	CPF, DateOfBirth and FullName are identical
Category2	CPF and DateOfBirth are identical, and both FullName entries have a maximum Levenshtein distance of 1
Category3	CPF, DateOfBirth and FirstName are identical, and both Surname entries have a maximum Levenshtein distance of 2
Category4	CPF, DateOfBirth and FirstName are identical
Category5	FullName, DateOfBirth, Email and PhoneNumber1 are identical
Category6	FullName, DateOfBirth, Email and PhoneNumber2 are identical
Category7	FullName, DateOfBirth, and Email are identical, and the PhoneNumber1 of the first record matches the PhoneNumber2 of the second
Category8	FullName, DateOfBirth, and Email are identical, and the PhoneNumber2 of the first record matches the PhoneNumber1 of the second
Category9	FullName, DateOfBirth, Email and are identical
Category10	FullName, DateOfBirth and PhoneNumber1 are identical
Category11	FullName, DateOfBirth and PhoneNumber2 are identical
Category12	FullName and DateOfBirth are identical, and the PhoneNumber1 of the first record matches the PhoneNumber2 of the second
Category13	FullName and DateOfBirth are identical, and the PhoneNumber2 of the first record matches the PhoneNumber1 of the second
Category14	FullName, DateOfBirth, Street Name and StreetNumber are identical
Category15	FullName, DateOfBirth and StreetNumber are identical

In addition to the criteria outlined in Table 2, all records undergoing unification must also meet the following conditions:

- The evaluated field values must not be *null*;

- The FirstName must not be equal to “RN”. When “RN” appears as the first name, it indicates a newborn. In such cases, the surname was filled with the full mother’s name, along with other relevant personal information pertaining to the mother;
- The CustomerId must be valid.

Each category has been defined to execute and return the pair original CustomerId and suggested unification primary CustomerId through a Spark dataframe as the result of the function.

To ensure each CustomerId appears only once in the category returned dataframe, if a CustomerId is duplicated with multiple other CustomerIds within the same category, priority is given to unifying the CustomerId with the most recent registration modification date record. In cases where the modification dates are identical, the CustomerId with the higher numerical value takes precedence for unification. This process ensures the uniqueness of each CustomerId in the returned dataframe for each category.

Subsequently, all category results have been combined into a single Spark dataframe containing the original CustomerId and the suggested correlated primary one, along with information about the unification category used.

In cases where a CustomerId appeared in multiple unification categories, preference was given to the lower category over the higher one. For instance, if CustomerId 1234 was paired with CustomerId 6789 across categories 1, 5, and 10, only the record “1234, 6789, 1” would appear in the final combined Spark dataframe, indicating that CustomerId 1234 is suggested to be unified with CustomerId 6789 through category 1.

This decision was informed by the structured nature of the categories, where a lower category indicates stronger criteria. For example, category 1 incorporates unchangeable personal information such as the CPF number and date of birth, as well as relatively stable details like the customer’s full name. In contrast, category 5 utilizes information that, while unique, may change over time, such as the customer’s e-mail or phone number. The last categories, like categories 14 and 15, employ criteria that are neither unique nor static, such as the customer’s address, and are subject to frequent change.

To ensure the intended success of the unification process, an additional verification step was incorporated into the final Spark dataframe. This step involved the development of an extra algorithmic function tasked with evaluating the presence of circular references. A circular reference is a chain of references where one object in the chain refers to the next object, and the last object refers to the first object again [9], as illustrated in Figure 1. This verification process, aims to guarantee the presence of a primary record for each unified customer. The identification of a circular reference during this verification step would signify a flaw in the algorithm's logic.

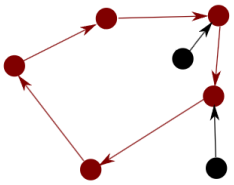

Fig. 1 - Circular reference in a graph, represented in red.

At the conclusion of the algorithm, the resulting Spark dataframe was exported to a CSV file, which was then utilized as a flat file source in SSIS and loaded into a SQL Server table.

3. Results

The first outcome was the creation of the new customer dimension table. This new dimension comprised 33 columns derived from the original ones after the cleansing process, and the row count remained unchanged between the old and new customer tables. Loading was scheduled to occur daily, ensuring the table remained up-to-date and synchronized with the source table using a full load approach. In this method, the table is truncated, removing all records, and then reloaded with updated information [10], thus preserving integrity between the source and destination.

The second outcome was the table containing the original CustomerId, the suggested correlated primary one, and the unification category. Loading was scheduled to occur weekly using an incremental load approach, wherein only the new or changed data is loaded. In the most recent execution, the table contained approximately 600 thousand rows, which represents 6% of the total customer dimension table row count. The distribution of unifications per category is depicted in Figure 2.

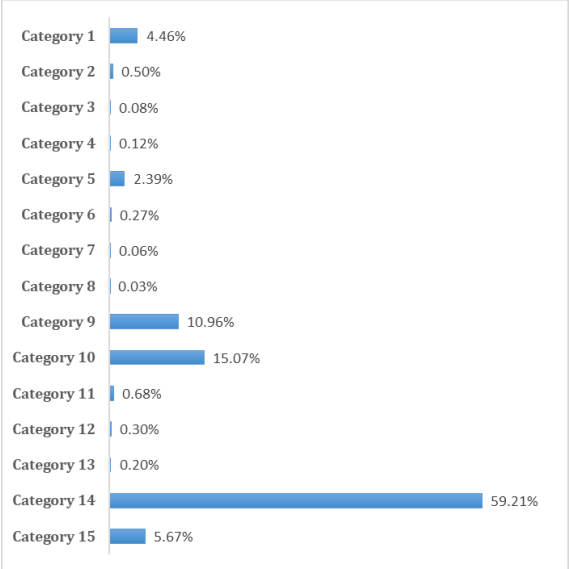

Fig. 2 - Bar chart: distribution of records across unification categories.

4. Discussion

The successful implementation of the ETL process in this study demonstrates its effectiveness in cleaning and preparing data for analysis. However, it is important to acknowledge that the current execution environment on-premises server presents some limitations in terms of scalability and performance. There is a clear opportunity for improvement by considering migration to a cloud computing service. Utilizing a cloud platform would offer significant advantages, especially for complex algorithms, such as those executed with PySpark, where distributed computing can accelerate execution time and improve overall process efficiency. Migration to the cloud would also provide additional flexibility and enable dynamic resource adjustments according to system demands.

Furthermore, the resulting table containing the data of suggested unifications provides a valuable source of information for the company's registration team. These data can be used to generate reports that allow for careful evaluation and precise validation of the proposed unifications.

Regarding the presented distribution of records across unifications categories, it is noteworthy how Category 14 stands out compared to the rest: almost 60% of the unifications belong to this category. This significant representation could be attributed to the lower proportion of *null* values in the columns used as merging criteria within this category, as depicted in Figure 3.

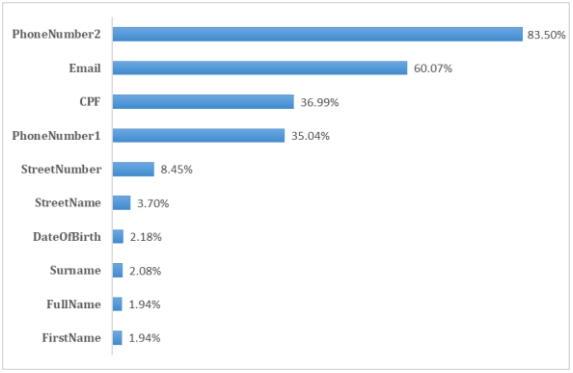

Fig. 3 - Bar chart: proportion of *null* values in fields utilized across unification categories.

5. Conclusion

This paper has outlined a comprehensive approach to enhancing data quality through the cleansing and unification of customer data in a Brazilian healthcare company. By employing a structured ETL process, significant improvements were achieved in the accuracy and reliability of the company's customer dimension table.

The methodology involved meticulous analysis of the data source, identification of inconsistencies, and the implementation of cleansing transformations using tools such as SSIS and Apache Spark. The cleansing process addressed various issues, including special characters, formatting discrepancies, and data validation errors, resulting in a refined customer dimension table with improved data integrity.

Moreover, the development of a Python algorithm facilitated the unification of customer records, proposing to resolve instances of duplicate entries. The algorithm provided a systematic approach to identifying and correlating similar customer records, further enhancing data accuracy.

The outcomes of this study include the creation of a new customer dimension table and a table containing suggested unifications, both of which serve as valuable resources for the company's data management and decision-making processes. The distribution of unifications per category offers insights into the prevalence of data inconsistencies and informs corrective actions by the user registration team.

Looking ahead, there are opportunities for further improvement, particularly in the scalability and performance of the data processing environment. Migration to a cloud computing service could unlock additional benefits, such as enhanced processing power and flexibility, thereby optimizing the efficiency of complex data cleansing and unification algorithms.

6. References

[1] Erhard E., Do H. Data Cleaning: Problems and Current Approaches. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*. 2000; 23(4):3-13.

[2] Lucas J., Raja U., Ishfaq R. How Clean is Clean Enough? Determining the Most Effective Use of Resources in the Data Cleansing Process. *International Conference on Information Systems Proceedings*. 2014; 35(1):1-10.

[3] Kimball R., Ross M. *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling*. John Wiley & Sons, Indianapolis; 2013; 564 p.

[4] Cote C, Lah M., Sarka D. *SQL Server 2017 Integration Services Cookbook*. Packt, Birmingham; 2017; 558 p.

[5] Kudale H., Phadnis M., Chittar P., Zarkar K., Bodhke B. A Review Of Data Analysis And Visualization Of Olympics Using Pyspark And Dash-Plotly. *International Research Journal of Modernization in Engineering Technology and Science*. 2022; 4(6):2093–2097.

[6] Carvalho P., Hitzelberger P., Otjacques B., Bouali F., Venturini G. Information Visualization for CSV Open Data Files Structure Analysis. *International Conference on Information Visualization Theory and Applications*. 2015; 6(1):101–108.

[7] Levenshtein V. Binary codes capable of correcting

deletions, insertions, and reversals. *Doklady Akademii Nauk SSSR*. 1965; 163(4), 845–848.

[8] Kruskal B. An overview of sequence comparison: Time warps, string edits, and macromolecules. *SIAM Review*. 1983; 25(2):201–237.

[9] Shukla C., Dreamtech Software India. *ASP.NET 2.0 Black Book*. Paraglyph Press, Scottsdale; 2006; 1167 p.

[10] Bogza R., Zaharie D., Avasilcai S., Bacali L. Architecture Models and Data Flows in Local and Group Datawarehouses. *Innovations in Computing Sciences and Software Engineering*. 2010; 1(1):627-362.