# Data and Information Quality Report

Daniele Ferracuti 10580840
Samuele Pietro Galli 10710025

Dataset n.13

## 1.1. Project Goal

The goal of this project is to implement a Data Preparation Pipeline on the assigned dataset (HOTELS) to address and resolve key Data Quality issues. The pipeline is designed to follow these steps:

1. Data Profiling and Quality Assessment: Understanding the dataset structure and identifying potential quality issues.
2. Data Cleaning: Transforming, standardizing, detecting, and correcting errors, including:
   - Data Transformation and Standardization: Bringing all data to a consistent format and correcting typographical errors.
   - Error Detection and Correction: Managing missing values and detecting and addressing potential outliers.
   - Data Deduplication: Identifying and resolving non-exact duplicates.
3. Data Quality Assessment (Post-Cleaning): Verifying the effectiveness of the cleaning process.

The dataset assigned to our group represents structured information about hotel facilities in Milan, including details such as location, capacity, and categorization.

For this project, we mainly used the libraries pandas, numpy, and re. Pandas was essential for working with tabular data, allowing us to manage columns, transform datasets, and filter data efficiently. Numpy was used for numerical operations, ensuring faster computations on larger datasets. Finally, the RE library helped with text normalization and cleaning, which was crucial for handling unstructured data like addresses and descriptions.

## PIPELINE

### 2.1. Preliminary assessment

We used head() to preview the first few rows, shape() to check its dimensions, and dtypes() to examine the data types of each column. To ensure data consistency, we applied duplicated() to detect duplicate rows and confirmed there were none. Columns were divided into numerical and categorical groups using select_dtypes(), facilitating targeted analysis. For numerical columns, we plotted histograms with hist() to explore the cardinality and identify potential patterns or irregularities in the data distribution, and to primarily see outliers.
The data assessment began with an evaluation of distinctness, constancy, completeness, and uniqueness. Then we moved to analyze the accuracy ensuring that the data adhered to expected formats and logical constraints.

### 2.2 Accuracy:

Functions designed to ensure the syntactic correctness of the data by validating specific columns: for example, the check_ubicazione function verifies that addresses follow a consistent pattern, starting with a valid prefix such as "VIA" or "PZA" and including other required details. The check_isdigit function ensures that numeric values, such as house numbers or street codes, are valid integers or properly formatted floating-point numbers within a reasonable range. This validation guarantees that columns like Civico or Codice via contain logical and consistent values. For categorical fields, the check_whitelist function confirms that the values belong to a predefined set of acceptable options, as applied to fields like Tipo via or Tipo attività. check_numeric_list and check_list, validate columns that include lists of values, ensuring they adhere to a structured pattern, such as semicolon-separated numbers or alphanumeric codes.

2.3 Consistency:
We assessed the consistency of the dataset by implementing a series of logical checks to verify the relationships between various columns. The safe_sum_split function was a key component in this process, designed to handle cases where numerical values were stored as semicolon-separated strings (e.g., "10;15;20"). It splits the string, converts the parts to numbers, and calculates their sum while accounting for potential missing or invalid values. This ensured that the total values recorded in columns like Camere and Posti letto matched the corresponding semicolon-separated details in Camere piano and Posti letto per piano. Verifica_corrispondenza, was used to validate address information. By reconstructing the full address from its component, Tipo via, Descrizione via, Civico, and ZD and comparing it with the recorded Ubicazione, this function ensured that the address fields were internally consistent and aligned.
The rule verified whether Piani totali matched the structure indicated by related columns Piano piano Camere piano and Posti letto per piano. It also checked if the summed values in Camere piano and Posti letto per piano equaled their respective totals, Camere and Posti letto. Additionally, it validated the alignment of address information by reconstructing the full address from its components and comparing it to the recorded Ubicazione. Rows that satisfied all these conditions were marked as consistent. The dataset has a consistency of 9.5%.

2.4 Data Cleaning (Data transformation)
We started calculating duplicates, and as a result that were absent. We then renamed columns for more clarity. Next, we calculated the number of missing values. Normalize function was used to standardize text fields by converting them to uppercase, removing extra spaces, and handling decimal points. The verifica_corrispondenza function then reconstructed the full address from its components (Tipo via, Descrizione via, Civico, and ZD) and compared it with the recorded Ubicazione. Rows where the reconstructed address did not match the recorded one were flagged for further inspection. estrai_valori function was employed to extract missing or inconsistent address components directly from the Ubicazione field. Using regular expressions, the function identifies elements such as the street type (e.g., "CSO"), street name, house number, and zone code (z.d.). These extracted values were organized into a new DataFrame (valori_df). update_df_by_location iterates through the rows of valori_df and checks for matches (HOTELS) based on the Ubicazione field; for each match found, it updates Tipo via, Descrizione via, Civico, and ZD with the corrected values from valori_df.
We decided to standardize values: in the Tipo via column, abbreviations such as "CSO" and "PZA" were replaced with their full forms ("CORSO" and "PIAZZA"), while other abbreviations like "ALZ" and "VLE" were standardized to "ALZAIA" and "VIALE,". In the Categoria column, ambiguous or numeric entries like "1" and "I" were replaced with descriptive labels such as "1 STELLA," while other categories ("2," "3") were transformed into their full descriptive equivalents ("2 STELLE," "3 STELLE"). In the Tipo attività struture extra column, the term "Albergo" was replaced with "Hotel," and missing values in this and other columns like Insegna were filled with "Non Specificato". After standardization rows with missing values in Tipo via, Descrizione via, Civico, ZD, and Categoria were dropped together with the Ubicazione column.
We moved on to address missing values in the Piani totali. Using the calcola_piani_totali function, we calculated the total number of floors by counting the separators (;) and adding 1; for rows where Piani totali was missing but Piano piano had a value, this method filled the

missing values.
Next, we addressed Piani totali was still missing but information was available in the Camere piano and Posti letto per piano.  For rows where both columns had consistent values, we used the same calcola_piani_totali.
Lastly, the costruisci_piano_piano function was implemented to reconstruct the Piano piano column for rows where it was missing but Piani totali was available. This function generates a semicolon-separated string representing the floor structure (e.g., "1;2;3" for three floors). Using this approach, we filled in the missing values for Piano piano, ensuring it aligned with the total number of floors for each property.
We focused on refining the Camere piano (rooms per floor) and Posti letto per piano (beds per floor) columns to ensure they were logically consistent with the total values (Camere and Posti letto) and the number of floors (Piani totali).
For hotels with a single floor (Piani totali = 1), the Camere piano and Posti letto per piano columns were directly filled with the total values from the Camere and Posti letto columns, respectively. This ensured that single-floor hotels had consistent data. The adjust_dataframe function was introduced to address inconsistencies in the Camere piano and Posti letto per piano columns. This function ensures that the values in these columns meet two critical conditions: the sum of the semicolon-separated values must equal the corresponding totals (Camere or Posti letto), and the number of entries in the semicolon-separated string must match the total number of floors (Piani totali). If the sum of the semicolon-separated values exceeds the total, the column is marked as "Non specificato" to highlight the inconsistency. In cases where the sum is less than the total or the number of values does not align with the number of floors, the function recalculates the distribution of values, either filling missing entries or redistributing excess values to achieve consistency with the totals and floor structure.

## 2.5 Data Cleaning (Error Detection & Correction for Missing Values):

We filled the missing values in critical columns. For example, the Codice via column was filled with "Non specificato" to indicate unavailable values explicitly. For Piani totali, where the total number of floors was missing, we assumed that these hotels had only one floor, filling the column with a value of 1. Similarly, the Piano piano column was set to "1" for hotels assumed to have a single floor. We used the Camere and Posti letto, respectively to address missing data in Camere piano and Posti letto per piano.

## 2.6 Data Cleaning (Error Detection & Correction for Outliers) phase:

We applied a Z-score method (ZS) to the Posti letto column with a threshold of 5, identifying extreme values that deviated significantly from the mean. The detected outliers included unusually high bed counts, such as 864, and 922.

## 2.7 Brief conclusive DQ Assessment

The data quality assessment shows good overall consistency (88.3% of tuples conform to the defined rules), while results for "Category" and "Activity Type" are low, partly due to the assumptions made during the analysis. In the absence of complete information, we applied criteria such as setting the total floors to 1 when missing or considering the total number of rooms valid over the room distribution per floor. While we cannot confirm these choices as entirely accurate, we followed a logical approach that allowed us to focus on the most significant data points, which demonstrate high accuracy levels and provide a solid foundation for the main analyses.