MBIS623 Case Study Report

Since the information telephone line was launched in 2003, 311 has been used very frequently. In New York City alone, it has fielded millions of calls. Each of these calls are citizens either requesting information, assistance, filing complaints, or reporting civic issues and these calls are then stored in the New York City 311 (NYC311) database. This report focuses on this database and the quality of the data stored within it, called 'Service Requests'. These service requests will be analysed against the terms of some common dimensions of data quality and automated data quality metrics will be developed to assist in displaying the level of quality present in the database. Data quality is defined as "the measure of how well suited a data set is to serve its specific purpose." (Heavy AI) and the metrics used will address this. The data in the service requests that will be investigated are the borough, incident zip, complaint type, and the full dates fields.

Data Quality Dimensions:

Some of the common dimensions are accuracy, timeliness, relevance, completeness, understandability, validity, uniqueness, integrity, and trustworthiness. In this report the dimensions that will be referred to with the metrics are accuracy, completeness, and validity. Further dimensions will be discussed to provide further clarification. The meanings for the chosen metrics are as follows; Accuracy will be measured against how accurate the values are in relation to their real-world values, completeness is judged against how many null or empty values are present in the database columns and if the data present "is sufficient to deliver meaningful inferences and decisions." (Gupta) Finally. validity is a measure of whether the data replicates the format of the typical input and how any discrepancies could cause problems for interpretation of the data.

Dates:

For the 'Date' data fields, the biggest indicators for good data quality would fall to validity, accuracy, and completeness. All these dimensions are relevant because dates have a very strict structure for how they are represented and how they work; the assumption is that the fields should only have one of two outcomes, to either have the date or not. The 'not' can come in many forms such as a null value, an incorrect value and/or the default value. For the service requests the 'Created Date' is the date the request was made, 'Closed Date' is when the request was closed by the responding agent and the 'Due Date' is when the responding agency is expected to update the request, all these elements are of the type 'Floating Timestamp',

For completeness, Query 1 returns percentages of null values against the entire column relevant to either the 'Closed Date', 'Created Date', or 'Due Date' fields. From this query it is clear that the 'Created Date' holds priority as it returns zero percent. This might mean that it is an automatic system that generates the created date rather than it being user input based. This comment can be made upon comparing the 'Closed Date' and 'Due Date' percentages to it as they hold 2.8% and 68.7% respectively which may indicate that they are values that are driven by user input and therefore might be because of user error. The high value of null returns in the 'Due Date' criteria might also indicate that some incidents don't require a due date for resolution, and this might be a reflection over whether this input is entirely necessary. In terms of data quality, it is suggested to try and enter as much data through automatic means and the 'Created Date' field proves how this method can work quite effectively.

In the cases of validity and accuracy, Query 2 returns the error percentage of 0.97%, which means this percentage of the 'Closed Date' column returns as occurring earlier than 'Created Date', this is an important check as a request should not be able to be closed before it was created. When looking at the overall data table it is also clear that some of these returned values are the SQL default value for the 'DATETIME' data type. Query 2 also returns this default percentage out of all the incorrect date fields as described earlier, 2.13%. Although a low percentage, this indicates more user input

error in association with the dates fields and therefore more cleaning needed for the data to correctly serve its purpose. A potential fix for both of these issues could be setting the default to always being after the created time and date.

Overall, the 'Created Date' column best suits the meaning of good data quality, and the 'Closed Date' column needs some minor revision to improve the data that it holds to elevate how well it can serve its purpose. At this stage the 'Due Date' column doesn't meet a high enough standard and seems to be only required for some service requests.

Incident Zip:

The 'Incident Zip' column holds fairly straight-forward information. This column should store the zip codes for where an incident has occurred, the assumption in this field is that the zip codes should be for New York City and that they all have a matching format. Looking at the service request table, the majority of the zip codes are represented by five digits and so this will be assumed to be the valid layout. This zip code is also provided by geo validation and is of the data type 'Text'.

Because all incidents must have a location, a check for completeness would be appropriate. Query 3 returns 4.98% of values either have a null value or an empty space, although this is a relatively small value it does bring up questions for the process of filling in this data. As it is a location it should be pulled from an address database in order to populate the text fields to prevent any null values being present in the final data set.

Testing the accuracy of the zip codes is also an important measure as zip codes are used widely and should be regulated within set values as the 'Incident Zip' field from NYC311 should only contain those which are located in New York City. Query 4 addresses this issue in relation to a reference dataset. 94.42% of the zip codes located in the complete dataset are located within the reference data which means that 5.58% are zip codes either invalid or located outside of New York City. If this information is incorrect, it can cause issues for those trying to use the zip codes for location services and may cause delays for the whole chain of events. The incorrect values clearly represent the issues surrounding user-based input methods as it is typical for manual input to have mistaken.

Finally, Query 5 checks the validity of the 'Incident Zip' summary table. Validity in this case is also very important as it is a principle of good data inference that the data has consistency. This validity query draws out the values that don't match the five digits that are typically seen. This includes the values such as "0", "?", "000000" and "00000-0000". It is important to consider that the nine-digit zip code is a valid zip in New York City, but in this context, it falls outside the common requirement and thus can cause confusion when trying to collate this data. The query returns 5.71% invalid zip codes within the summary table,

Complaint Type:

The 'Complaint Type' column holds a lot more variable content, and it seems that there isn't a conventional way of naming the complaints. This column should consist of a string of varying length that gives a brief description of the type of complaint. Viewing the summary of complaints, it is clear to see that this is a field that is based on manual input with some preference to previous complaints as there are 'Complaint Types' that do not contain words or consist primarily of symbols. Because of how many variations this field can contain it would be appropriate to map each complaint to a specific ID type for ease of access for the employees entering the information. This could turn the text field into a drop-down box that only accepts the prefilled ID's. Reviewing the map file of the 'Complain Type', it is clear that this process has started but it still consists of many null values and therefore does not contain good data quality. This field has a data type of 'Text' and can have further information supplemented about an event with a corresponding descriptor column, but this is not necessary.

Query 6 focuses on checking the validity of the 'Complaint Type' summary table; it also tries to reduce the number of potential duplications present in the data field from poor input by using the 'REGEXP_REPLACE' function which replaces the given text with a blank space. This check is necessary to analyse how many complaints do not contain a proper value for fulfilling the requirement to meet the definition of good data quality. The query returns that 61.73% of the complaints match the reference table and the rest are not a valid input for data usage.

In the case of the 'Complaint Type' column, it is a very difficult field to manage. This is because the information that the database employees are receiving are from the public and it may not be worded in a typical way. This would mean that it is left up to the employee to interpret what someone from the public is meaning or implying and because the input isn't restricted this can allow for duplications, bogus inputs, or questionable data.

Borough:

The 'Borough' data field is a very direct field that has strict requirements to meet a good data quality. The data in this field should only consist of a borough from New York City, like the zip codes. Completeness and accuracy would once again be very prominent in this column. Further dimensions that could be considered could be uniqueness and relevance. Uniqueness could be presented as data in a heatmap for where queries are sourced from and relevance in comparison to zip codes - is it explicitly needed to have both the borough and zip code. An answer to that question could be that the database is trying to make up for any incompleteness within itself and a missing borough could be assuaged by having a zip code or vice versa. The 'Borough' column has a data type of 'Text' and is confirmed by geo validation.

Completeness in terms of the 'Borough' column is interpreted to mean containing a string of text that names a borough of New York City. This means that any null inputs would go against the meaning of completeness. Query 7 addresses this and returns 0.14%, this gives a good insight that the data is well populated and has a value for nearly 100% of its values. To further expand on this Query 8 addresses the use of 'Unspecified' to replace the actual borough and returns 4.12%. This means that approximately 4% of the boroughs in the dataset are considered unspecified, although this doesn't meet the meaning for good data quality, it does improve it substantially as it gives a value as to the reasons behind it and doesn't leave the value incomplete.

Because there are real life values to measure this data against, accuracy is another important dimension. Query 9 returns two percentages, 95.74% of the boroughs in the data set match the New York City borough reference data set and 4.12% fall outside of a match. This is a high result for a match, yet it still shows that an automated option would be preferable to remove any chance of mismatches.

To sum up, the 'Borough' column is a well populated field that has a set of methods in place to address when information is not presented with a request.

Conclusion

In conclusion, the NYC311 database has a lot to work on to improve their data quality. Their focus is on addressing as many requests as possible rather than having completely accurate requests as cleaning the data would take time away from being able to address their customers. Focusing on automating response and text fields would help reduce the amount of poor data within the system as user input is the primary issue whether this stems from low levels of training or mistakes. This does not mean that the NYC311 database is full of poor data though, as it still meets its specific purpose and addresses New York City and its people's concerns. It is also a database full of incredibly interesting information that can help further database systems and research queries into how New York City can function.

Citations:

Heavy AI. "Data Quality." *What Is Data Quality? Definition and FAQs | HEAVY.AI*, https://www.heavy.ai/technical-glossary/data-quality#:~:text=Data%20quality%20is%20the%20measure,validity%2C%20uniqueness%2C%20and%20timeliness.

Gupta, Ankur. "The 6 Dimensions of Data Quality." *Collibra*, 14 Sept. 2021, https://www.collibra.com/us/en/blog/the-6-dimensions-of-data-quality.

DoITT, 311. "311 Service Requests from 2010 to Present: NYC Open Data." *311 Service Requests from 2010 to Present | NYC Open Data*, 9 May 2022, https://data.cityofnewyork.us/Social-Services/311-Service-Requests-from-2010-to-Present/erm2-nwe9.