Data Management Assignment 1

# Assumptions and domains

**Date Fields**

“Created date”, “Closed date”, and “Due date” are all datetime values in the format “yyyy-mm—dd hh:mm:ss”

Upon the creation of a service request entity, I am assuming the following:

* the created date refers to the datetime that refers to when the customer made the call request. Every Request, by definition, requires one of these.
* the closed date refers to the datetime when the call center representative resolves a request. Naturally, a service request that has a NULL closed date has not been resolved.
* Due date is possibly a datetime generated by either the system or manually input by the call agent which refers to an estimated time of resolution for the service request. Every request must be associated with a due date.

**Status**

Status refers to the stage of resolution process in which the service request sits. The assumption behind this is that when a representative closes a request, they mark it as ‘Closed.’ Any other value refers to the stage at which the current service request resides during the process of fixing. The possible values for this are text strings “Open”,” Assigned”,” Pending”,” In Progress” or” Closed.”

**Incident zip fields**

Incident Zip is stored as a character of length five. However, necessarily includes character representations of integers. Each service request has one assigned. Incident Zip refers to the ZIP code of the location at which the service request occurs. Throughout the dataset, we see this field with the alternative names “Incident Zip ID” or “Zip.”

**Complaint Type**

Complaint type refers to the category of the service request and serves to group request of the same or similar nature together. This takes the format of text (readable characters). In the reference data we have a base level of twenty-six categories for this under ‘complaint type’.

**Problem Area**

Arguably, ‘Complaint type’ and ‘problem area’ are interchangeable as they have the same function and domain. ‘Problem area’ has fifty-seven distinct values which one could think of as categories. Therefore, with four common categories forming a slight overlap it could be said that there are seventy-nine base categories to accommodate service requests.

**Borough**

Borough takes the format of text and refers to the name of the district, in New York, from which a customer makes a service request. Note the alternative name ‘Park Borough.’

# Data Quality Metrics

1. **% Missing Due Dates**

This metric refers to the Percentage of NULL values within the “due date” column. We would s would attribute said percentage to Completeness since we are concerned with the fullness of rows.

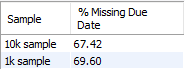


Figure . Missing Due dates

1. **% Created Date After Closed Date**

This metric captures the number of rows wherein the ‘created date’ field occurs after the ‘closed date’ field. It does not make logical sense for a representative to close a request even before it is made. Such an action violates certain semantic rules. And so, this must be measuring Consistency.

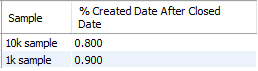


Figure . Created Date After Closed Date

1. **% Due date before Created Date**

This is a metric that measures the consistency of the fields ‘Due Date’ and ‘Created Date’ by counting the number of rows in the dataset where a service request is due before the instance is created. Once again, this does not make follow logically from the definitions of the fields in question.

Table

Description automatically generated

Figure 3 Due date before Created Date

1. **% Status not Match Closed Date**

This metric quantifies the Consistency between the field “Status” and the field “Closed Date.” Intuitively, we should expect that when the status shifts to ‘Closed’ then there must be a closed date. Conversely, we should also expect that when the status is open that there is not a Closed Date. Due to the ambiguity of not knowing which of these terms is the correct one, I have decided to place this under the category of Consistency rather than Accuracy.

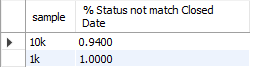


Figure . Status not matching Closed Date

1. **% Zip Code Missingness**

Here we are attempting to measure the number of service requests in the dataset that do not have an assigned zip code. Since each customer makes a service request is from a location it follows that each row should have an assigned ZIP Code (See Assumptions). Hence, we are measuring data completeness.



Figure . Missing Zips

1. **% Zip no Borough**

The no borough measure seeks to quantify the number of rows where a zip code does not correspond to a borough. For completeness, we should expect every zip code to have an assigned borough and vice versa. It does not follow logically that a request occurs within a zip code area but not a borough.



Figure 6.Zip no Assigned Borough

And so, we see the measure corresponds to %5.7 of the rows in the complete database. This falls under the category of moderate quality.

1. **% Invalid Zip Codes**

Here we are aiming to correctly quantify the percentage of rows that do not follow the 5-digit format of a zip code. Since we are assessing how well the value is represented by its domain, this falls under the category of validity.



Figure . invalid zip codes

1. **Complaint Types not in reference Data**

This metric quantifies the consistency of the “complaint type” field when measured against the reference data table “ref\_sr\_type\_nyc311\_open\_data\_twenty-six”. It shows a lack of structure consistency across entity types.

Text

Description automatically generated

Figure 8. No Reference Match

1. **Borough Zip Consistency**

This metric calculates the number of rows in the sample dataset where the ‘Borough’ of the service request does not refer to the same ‘Borough’ as it’s respective zip code in the ‘zip\_code\_nyc\_borough’ reference table. Discrepancies among the same entities tell us that we are measuring Consistency.

**Graphical user interface, application, table

Description automatically generated**

Figure 9. Unmatched Boroughs

1. **Inconsistent Park Boroughs**

Inconsistent Park Boroughs refers to the service requests that do not have ‘Park Borough’ fields that correspond to Boroughs in the table ‘zip\_code\_nyc\_borough’. In this case, the data has a different domain definition for ‘Park Borough’ and for ‘Borough.’ Namely, the presence of ‘Unknown’ values.

Text

Description automatically generated with medium confidence

Figure 10.Park Boroughs no map

1. **Missing Park Boroughs**

This is a metric that calculates the number of service requests which do not have an associated borough. Since we would expect every request to take place in a park, the absence of an assigned value suggests the presence of data artefacts. And so, we are assessing data quality Completeness.

Graphical user interface, text, application, table

Description automatically generated

Figure 11. Missing Park Boroughs

1. **Complete Created Date**

Here we have a metric that measures the number of rows that are missing ‘Created Date’. We are dealing with Completeness as every service request is expected to have a created date on data creation.

****

Figure 12 Created Date Complete

# Results and Conclusion

**Overall Data Quality**

Overall, the data has moderate data quality. While some fields have large proportions of data that have poor data quality, most of the negative measures were insignificant, often hovering around One Percent or less. In addition, ‘Created Date’ showed a high level of Completeness.

With regards to the measures that reported substantial quality issues, Complaint type Inconsistency and Missing Due dates corresponded to 37 and 67 percent of their respective fields. It appears that for ‘Complaint type’ the existing categories for service requests do not account for the large variety in request types. Naturally, these will need to be redefined. Missing due dates could either mean that my assumption is incorrect or that there are drastic underlying problems in the data collection process. These would need to be addressed.

Furthermore, we have a metrics that measure Validity of Zip codes at 5 Percent, Park Borough Inconsistency at 4 Percent and Park Borough Missingness at 2 Percent. Keeping in mind that 5 Percent equates to 221,890 rows, this compounds to a sizeable number of rows with data quality issues. This means that we will need to address these fields as well. It is possible that these were human errors from filling out input forms or that the program which brought the data into the database had bugs. All of which contributing to a reduction to the overall data quality.