Food Inspections (Dallas and Chicago)

After performing Data Profiling on both the datasets: We saw for missing values, inconsistent values and data types to match up with the dimension tables:

First we made data dictionary separately for Dallas and Chicago datasets, we listed down column names and data types that were coming from staging tables so it was easier while loading into dim and facts

**Dallas Food Inspection : Data Catalog**

|  |  |  |
| --- | --- | --- |
| Restaurant Name | Varchar(65) Null |  |
| Inspection Type | Char(9) Null | Code indicating the inspection type, such as **Routine, Follow-up, Complaint, Temporary and Mobile.** • Routine Inspections – are conducted at least once every six months • Follow-up Inspections – are conducted as a result of poor sanitation issues, low scores • Complaints Inspections – General Sanitation/Hygienic Practices /Illness Investigation, Smoking and Other • Temporary – the City of Dallas Office of Special Events provides a listing of public events being held involving food and the Consumer Health Division provides guidance and inspects • Mobile – the various mobile food units are inspected annually with random inspections conducted during the year |
| Inspection\_Date | Date |  |
| Inspection score | Smallint | The aggregate score from the inspection violations. Please note not every violation will reflect a point deduction as establishments are allowed to correct violations during the inspection process, and therefore no reduction in the overall score is reflected for the violation |
| Street Number | Int null | Eg 123 |
| Street Name | Varchar(25) | Eg Name |
| Street Direction | Char(1) | Eg N, E |
| Street Type | Char(4) | Eg LD |
| Street Unit | Char(5) | Eg Unit of shop |
| Zip Code | Char(10) | Eg |
| Violation Description |  | (Violation Number + Description) |
| Violation points |  |  |
| Violation Detail |  |  |
| Inspection Month | Char(8) |  |
| Inspection FY | Char(6) |  |
| Latitude | Float |  |
| Longitude | Float |  |

**Chicago Food Inspection : Data Catalog**

Inspection ID, DBA Name, AKA Name, License #, Facility Type, Risk, Address, City, State, Zip, Inspection Date, Inspection Type, Results, Latitude, Longitude, Location, Inspection\_Date, Violations, Violation Code, Violation Description, Violation Comments

**Business\_License**

|  |  |  |
| --- | --- | --- |
| ID | Char(16) |  |
| LICENSE ID | Int | An internal database ID for each record. Each license can have multiple records as it goes through renewals and other transactions. See the LICENSE NUMBER field for the number generally known to the public and used in most other data sources that refer to the license.  License ID is used from Business\_License Dataset of Chicago to connect with FoodInspection Dataset of Chicago |

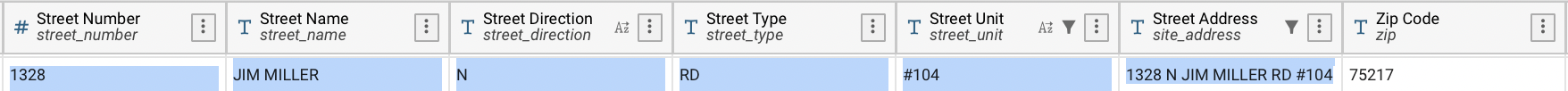
**FoodInspection\_Chiacgo**

|  |  |  |
| --- | --- | --- |
| Inspection ID | Int |  |
| DBA Name | Varchar(79) |  |
| AKA Name | Varchar(79) |  |
| License # | Int |  |
| Facility Type | Varchar(47) |  |
| Risk | Char(15) |  |
| Address | Varchar(51) |  |
| City | Varchar(20) |  |
| State | char() |  |
| Zip | int |  |
| Inspection Date | date |  |
| Inspection Type | varchar() |  |
| Results | varchar() |  |
| Latitude | char() |  |
| Longitude | char() |  |
| Inspection\_Date | date |  |
| Location | char() |  |
| Violations | varchar() |  |
| Violation Code | int |  |
| Violation Description | varchar() |  |
| Violation Comments | varchar() |  |

**General Insights for Chicago and Dallas Dataset:**

1. In Dallas, had to Spli the lat long location column into latitude and longitude using regex, data cleansing, text to column connectors
2. In Dallas, Since the violation attributes in Dallas had it type of data that is in the wide format(there were 25 violation attributes), we took the 25 unique combination of Violation Description -1, Violation Points -1, Violation Detail-1, Violation Memo -1, similarly for 2 till 25 and then we performed union to poulate this data into a single common column for each of the violation attributes.  
     
   Graphical user interface, application

   Description automatically generated
3. Fact 2 Loading:  
     
   Graphical user interface, Word

   Description automatically generated
4. In dallas we found that the address attributes like the column Street Address was a combination of (Street Number, Street Name, Street Type, Street Direction, Street Unit). So initially while loading the food inspection fact we joined on all this attributes from staging tables with the dim dallas geo. But after studying the dataset we found some discrepancies like here you can see for the first row: the combination works but for the second row although   
     
   

Discrepancies example: here although the street unit is null, still the street address shows some kind of value for street unit as #334. So when we join with all the address attributes the valid addresses were not being captured in the inner join output due this descrepancies, so we decided to join on only street address and zipcode which are up to date to populate the fact table to get the correct row count.

****

1. We also found that the number before the description had some standard meaning to it which is actually a violation code which we used further in our combined food inspection.

**Table

Description automatically generated with low confidence**

1. In chicago, we had to split the into latitude and longitude using regex, data cleansing, text to column connectors
2. In chicago, the violation column also had to be split into 3 columns: Violation code, violation description and violation comments, we did that by using regex:  
     
   Graphical user interface, text

   Description automatically generated
3. We also found that DBA corresponds to legal name, so we joined according to the business document standard definition

Text, letter

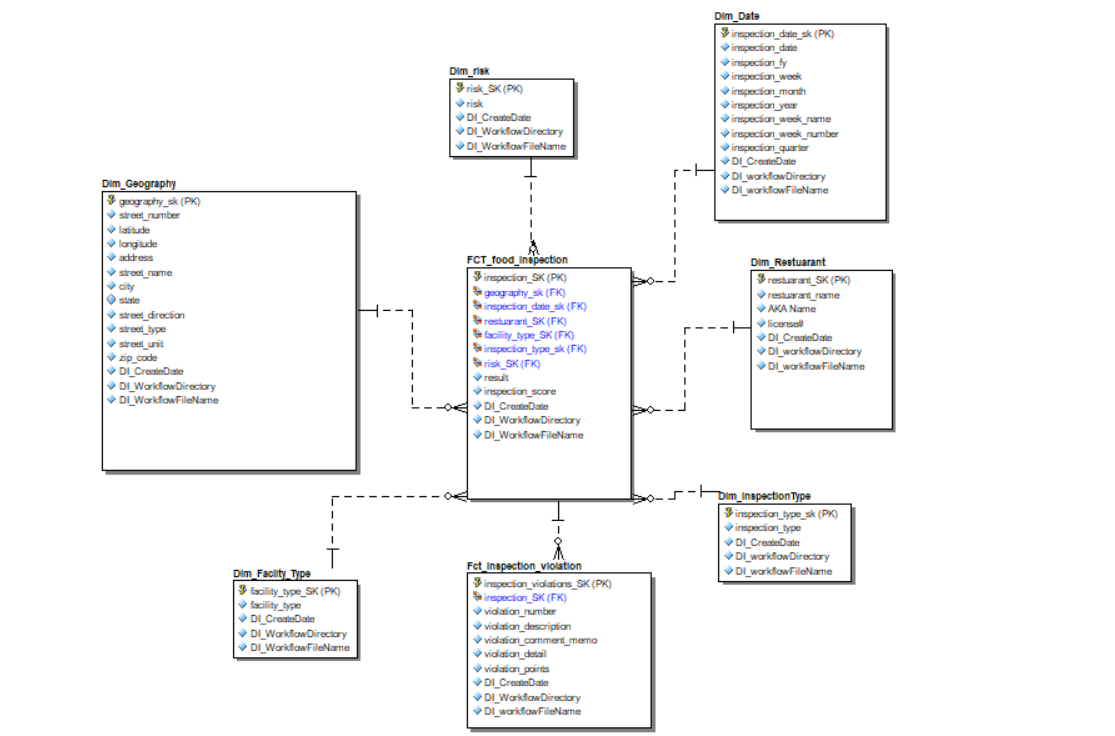
Description automatically generated

Graphical user interface, text, application, chat or text message

Description automatically generated

1. Here we joined Chicago inspections dataset and business licenses on the license number, restaurant name and address to get inner joined output as for this all the restaurants that had inspections had license number available.  
     
   Dim Restaurant Reload: Now there were many restaurants in Chicago who had the inspections but did not have the license number attributes present in the business license dataset, so it was not getting captured in the inner join output. For this we performed union on the inner join output and left join output to get all the records that had food inspections recorded for chicago.

**Combined Dimensional Model:** This is the combined dimensional model that we designed.



We the insights we got from the individual chicago and dallas dataset, used same cleaning process and techniques while loading the dim and facts

**Reasons why we choose to make these dimensions:**

After analyzing the dataset provided for both the Dallas and Chicago food inspections, we have identified some common entities and attributes that can be modeled in a single data model.

**Dim\_Restaurant:** This entity will contain information about the restaurants where the inspections are being conducted.

**Inspection Type:** This entity will contain information about the type of inspections conducted.

**Dim\_Date:** This entity will contain inspection dates along with different grain elements that are useful for analyzing the trends (weekly, quarterly etc.)

**Dim\_Geography**: This entity will contain the address attributes associated with the restaurants present in both data set.

**Reason for Dim\_risk and Dim\_Facility\_Type as a separate dimension:**

This is because in the future there is a possibility of the expansion of these values and it will be better to make changes in the dimension rather than making changes in fact. (Currently this attributes has values only for chicago food dataset)  
  
**For the inspection\_violation\_fact:**We studied the dataset and found that some of the violation attributes can be combined together for both chicago and dallas, (color coded are combined together to represent one column for both dallas and chicago)

Violation Description (Dallas) = (Violation Number + Description)

Violation points (Dallas)

Violation Detail (Dallas)

Violation Memo(Dallas)

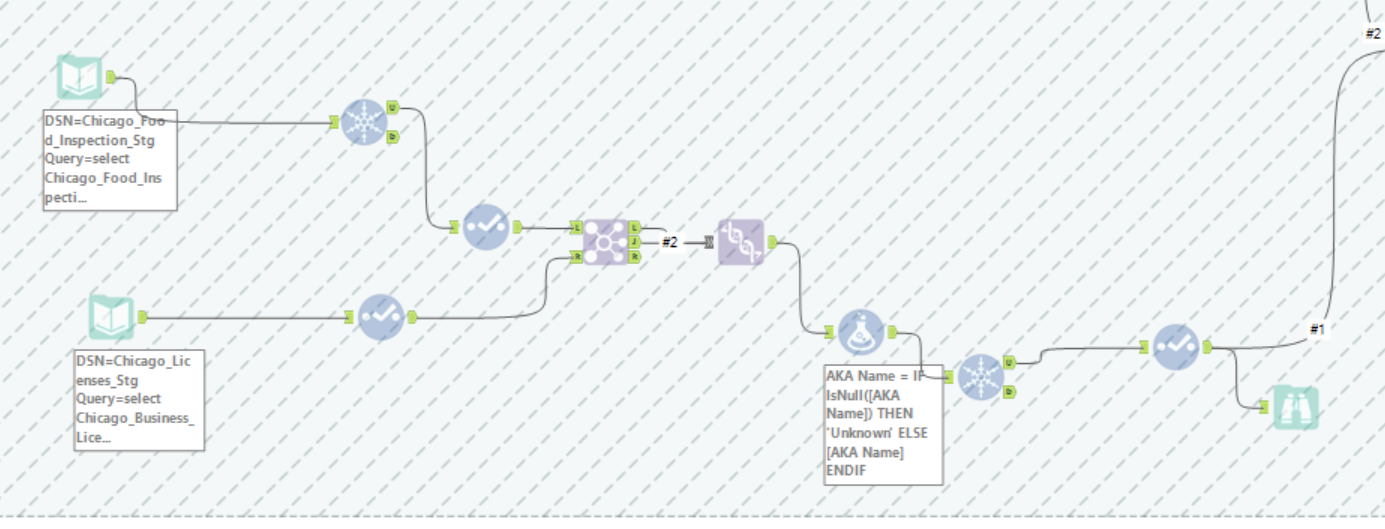
Violations(Chicago): which we splitted into the 3 columns belows

Violation Code (Chicago)

Violation Description(Chicago)

Violation Comments(Chicago)

**Reason for excluding the business licenses attributes that are present for the Chicago:**Since the business licenses staging table had licenses for all the entities( food, hospital, school etc), we though of eliminating the unwanted one and only storing the required one.

In dim\_restaurant, along with restaurant\_name and aka\_name, we took the only license number from the business licenses staging. Here also the license number was only available for chicago so had to define default values for the license number attribute as -9999 for dallas records since we were combining it into a common dim.   
  
While loading the restaurant dimension for the chicago in alteryx:  


**DISCREPANCIES:**

There are some discrepancies that were observed while loading the data into the common data model:

**Data Types**: Some of the attributes in the original data sets had distinct data types. The Inspection Date attribute, for instance, was of type datetime in the Dallas data set while it was of type date in the Chicago data set. In the shared data model, we had to make sure that the attribute data types were consistent between the two data collections.

**Attribute Names**: The attribute names in the original data sets were not consistent. For example, in the Dallas dataset there was an Inspection Score attribute, whereas in the Chicago dataset there were Results.

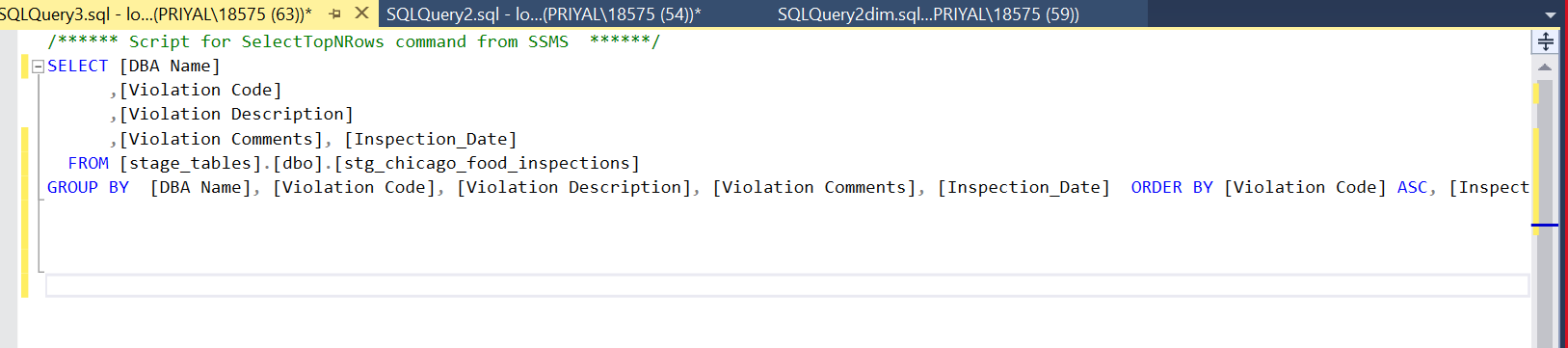
Attributes like City and State were missing from the Dallas dataset, hence we imputation them with default values, City - Dallas & State - TX.

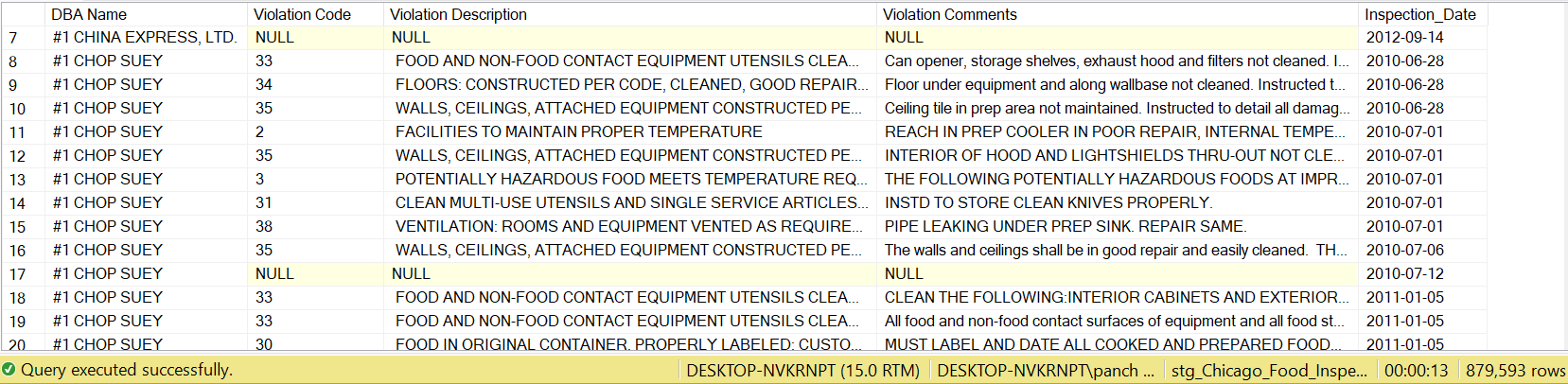
Similarly, there were other attributes that were missing in the Dallas dataset but were essential to the data model, hence we imputed them with default values. We had to ensure that the attribute names in the common data model were uniform across both data sets.

**Primary Keys**: The primary keys in the original data sets were not consistent. For example, the primary key for the FCT\_Inspection table in the Dallas data set was named InspectionSK, whereas the primary key for the FCT\_Inspection table in the Chicago data set was named InspectionID. We had to make sure that the primary keys were consistent across both data sets in the common data model.

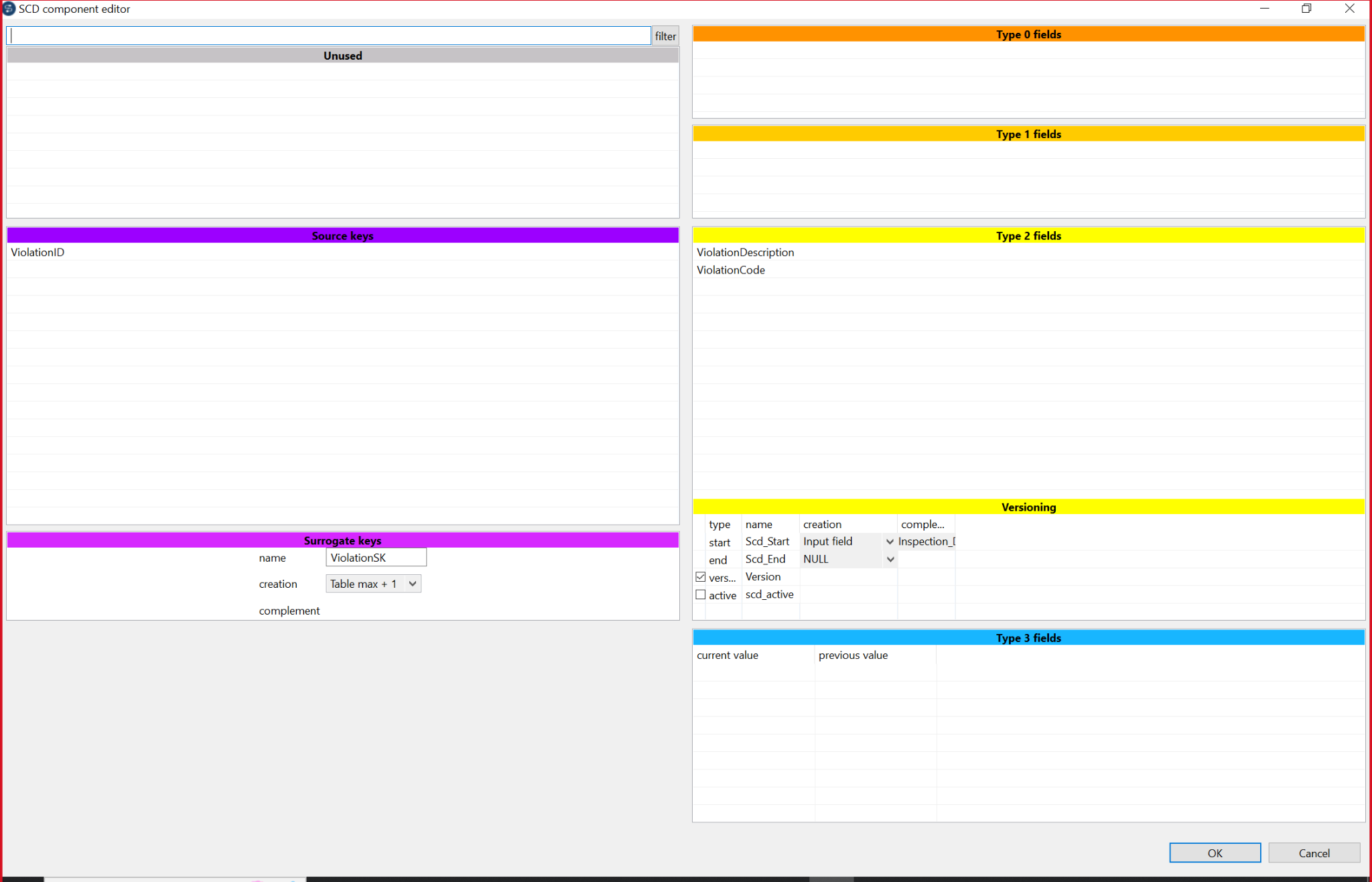
**Data Quality**: We observed some issues with the data quality in both data sets. For example, some of the restaurants had missing or incorrect address information. We had to perform some data cleaning and validation to ensure the quality of the data before loading it into the common data model.

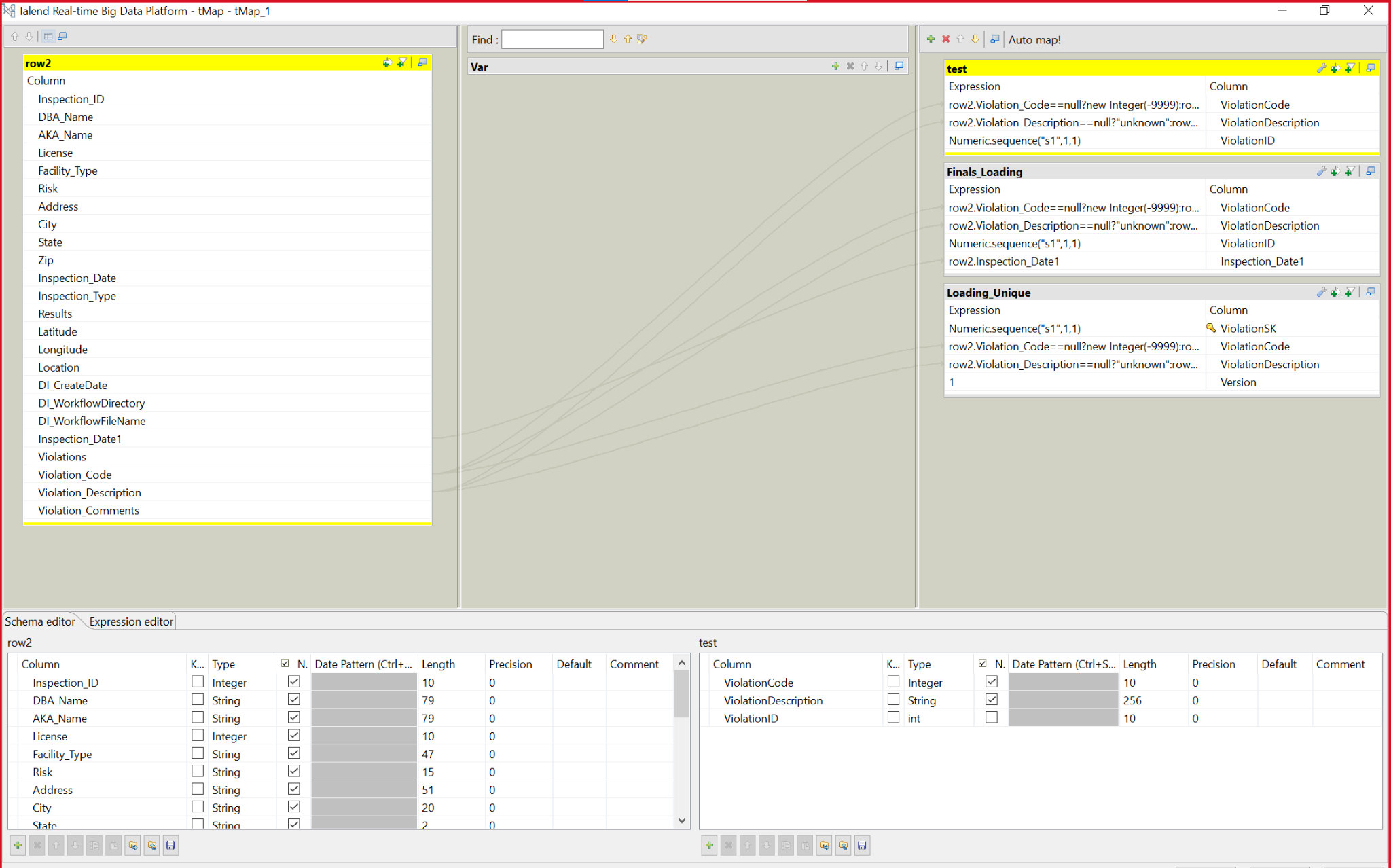
**SCD:**

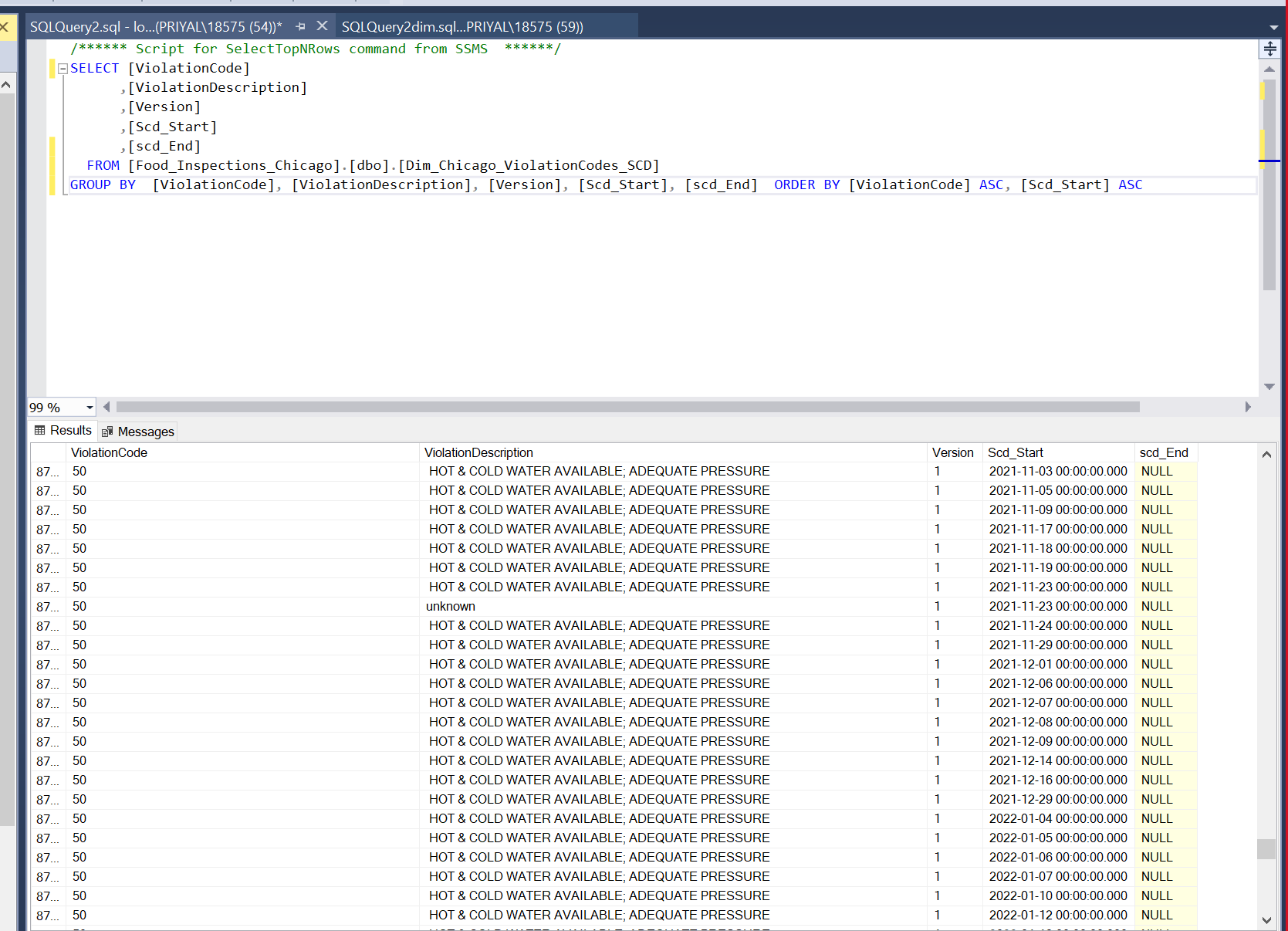
SCD for violations (Query done on stage table): ****



The scd table we implemented using Prof Rick Sherman’s data model:







Previously, In the data model which was provided to us there was an scd for Violation\_Code as in the above SS we can see that for a DBA Name there is a Violation code change at every inspection\_date. But there was an error in the Loading of the SCD in data model as in the first SS attach we can see that the Version column is coming out to be same for all the rows and as there was no end date provided to us in the data set and in the data model the End\_date column is NULL for all the rows.

Now, we have not Shown SCD in our version of the data model because in the original dataset of chicago we just have inspection date but no where the end\_date is mentioned of the inspection date in the dataset. So, if we want to implement scd correctly then first we have to track all the inspection dates of the DBA and make a new column where we will store when the inspection is ended. This column will help us populate the End\_date column in the scd and also gives us the correct version.

**Chicago Dimensional Model:**

Diagram

Description automatically generated

**Dallas Dimensional Model:**

Diagram

Description automatically generated

Dallas food inspections staging row count: 69502  
Graphical user interface, application, table

Description automatically generated

Dallas Dimensional model row count:  
  
A computer screen capture

Description automatically generated with medium confidence

Chicago business licenses staging row count: 1092416  
  
A screenshot of a computer

Description automatically generated with medium confidence

Chicago food inspection staging row count: 887227  
Graphical user interface, text, application

Description automatically generated

Chicago Dimensional model row count:  
Graphical user interface, text

Description automatically generated

Combined Dimensional model row count:

Graphical user interface, table

Description automatically generated with medium confidence