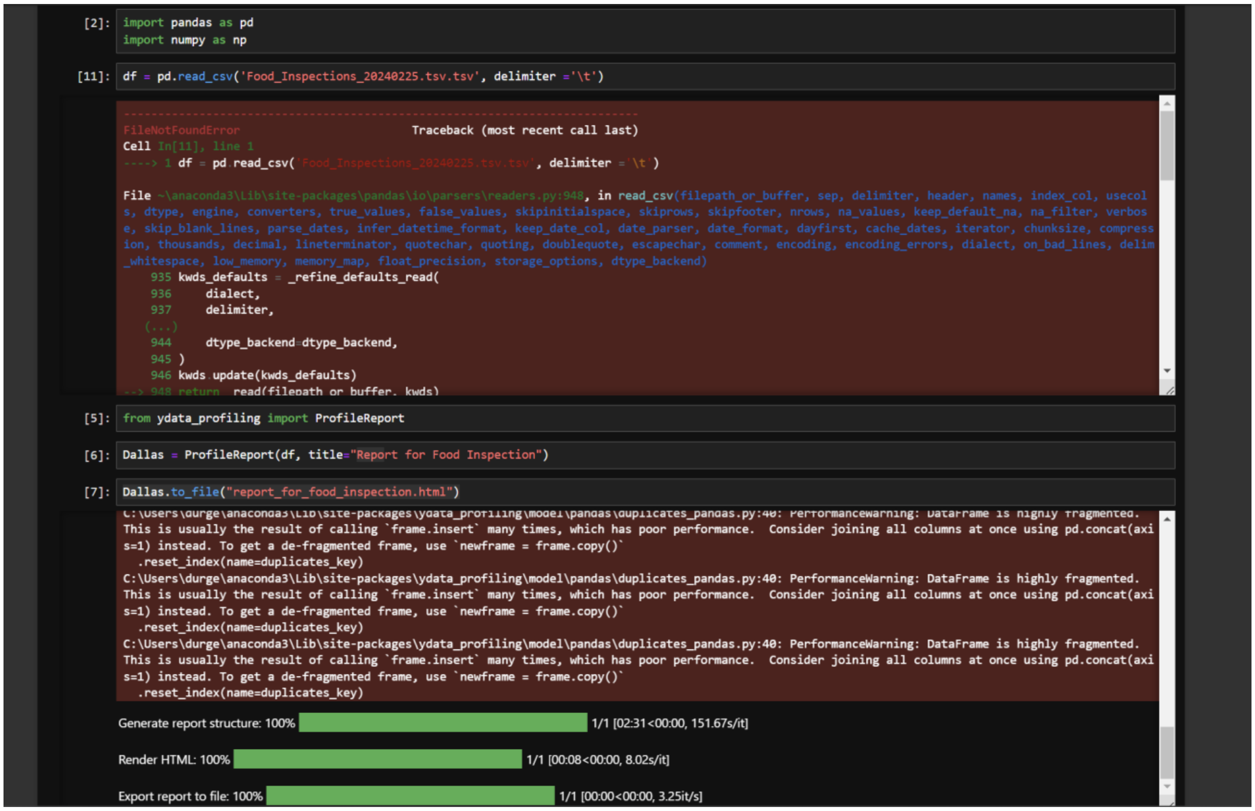
Data Profiling

Dallas

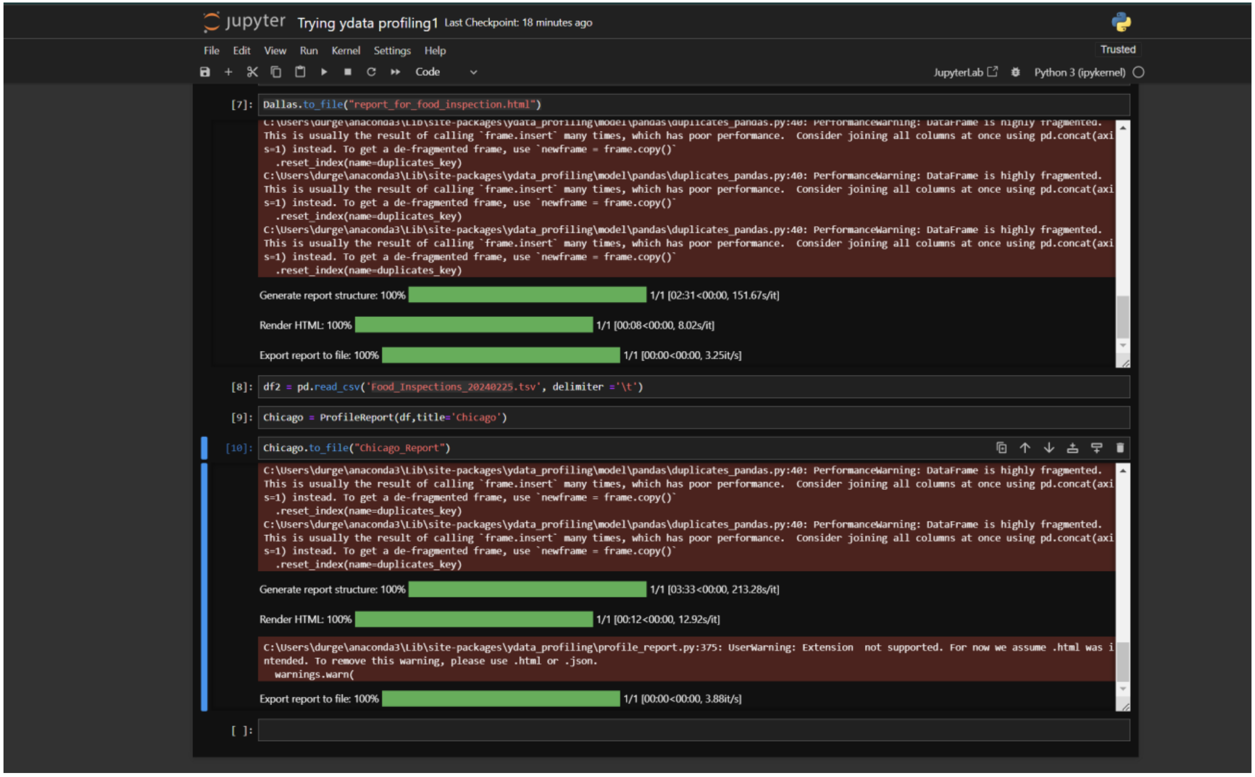


```
[2]: import pandas as pd
     import numpy as np

[11]: df = pd.read_csv('Food_Inspections_20240225.tsv.tsv', delimiter ='\t')

---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
Cell In[11], line 1
----> 1 df = pd.read_csv('Food_Inspections_20240225.tsv.tsv', delimiter ='\t')

File ~\anaconda3\Lib\site-packages\pandas\io\parsers\readers.py:948, in read_csv(filepath_or_buffer, sep, delimiter, header, names, index_col, usecol
s, dtype, engine, converters, true_values, false_values, skipinitialspace, skiprows, skipfooter, nrows, na_values, keep_default_na, na_filter, verbos
e, skip_blank_lines, parse_dates, infer_datetime_format, keep_date_col, date_parser, date_format, dayfirst, cache_dates, iterator, chunksize, compress
ion, thousands, decimal, lineterminator, quotechar, quoting, doublequote, escapechar, comment, encoding, encoding_errors, dialect, on_bad_lines, delim
_whitespace, low_memory, memory_map, float_precision, storage_options, dtype_backend)
    935 kwds_defaults = _refine_defaults_read(
    936     dialect,
    937     delimiter,
   (...)
    944     dtype_backend=dtype_backend,
    945 )
    946 kwds.update(kwds_defaults)
--> 948 return _read(filepath_or_buffer, kwds)

[5]: from ydata_profiling import ProfileReport

[6]: Dallas = ProfileReport(df, title="Report for Food Inspection")

[7]: Dallas.to_file("report_for_food_inspection.html")

C:\Users\durge\anaconda3\Lib\site-packages\ydata_profiling\model\pandas\duplicates_pandas.py:40: PerformanceWarning: DataFrame is highly fragmented.
This is usually the result of calling `frame.insert` many times, which has poor performance.  Consider joining all columns at once using pd.concat(axi
s=1) instead. To get a de-fragmented frame, use `newframe = frame.copy()`
  .reset_index(name=duplicates_key)
C:\Users\durge\anaconda3\Lib\site-packages\ydata_profiling\model\pandas\duplicates_pandas.py:40: PerformanceWarning: DataFrame is highly fragmented.
This is usually the result of calling `frame.insert` many times, which has poor performance.  Consider joining all columns at once using pd.concat(axi
s=1) instead. To get a de-fragmented frame, use `newframe = frame.copy()`
  .reset_index(name=duplicates_key)
C:\Users\durge\anaconda3\Lib\site-packages\ydata_profiling\model\pandas\duplicates_pandas.py:40: PerformanceWarning: DataFrame is highly fragmented.
This is usually the result of calling `frame.insert` many times, which has poor performance.  Consider joining all columns at once using pd.concat(axi
s=1) instead. To get a de-fragmented frame, use `newframe = frame.copy()`
  .reset_index(name=duplicates_key)

Generate report structure: 100% 1/1 [02:31<00:00, 151.67s/it]

Render HTML: 100% 1/1 [00:08<00:00, 8.02s/it]

Export report to file: 100% 1/1 [00:00<00:00, 3.25it/s]
```

Here we used ydata_profiling for profiling of **“Restaurant and Food Establishment Inspections (October 2016 to Present) | Dallas OpenDataLinks to an external site”**.

We stored the file to a dataframe and then using ydata profiling generated a “Profile Report” in html.

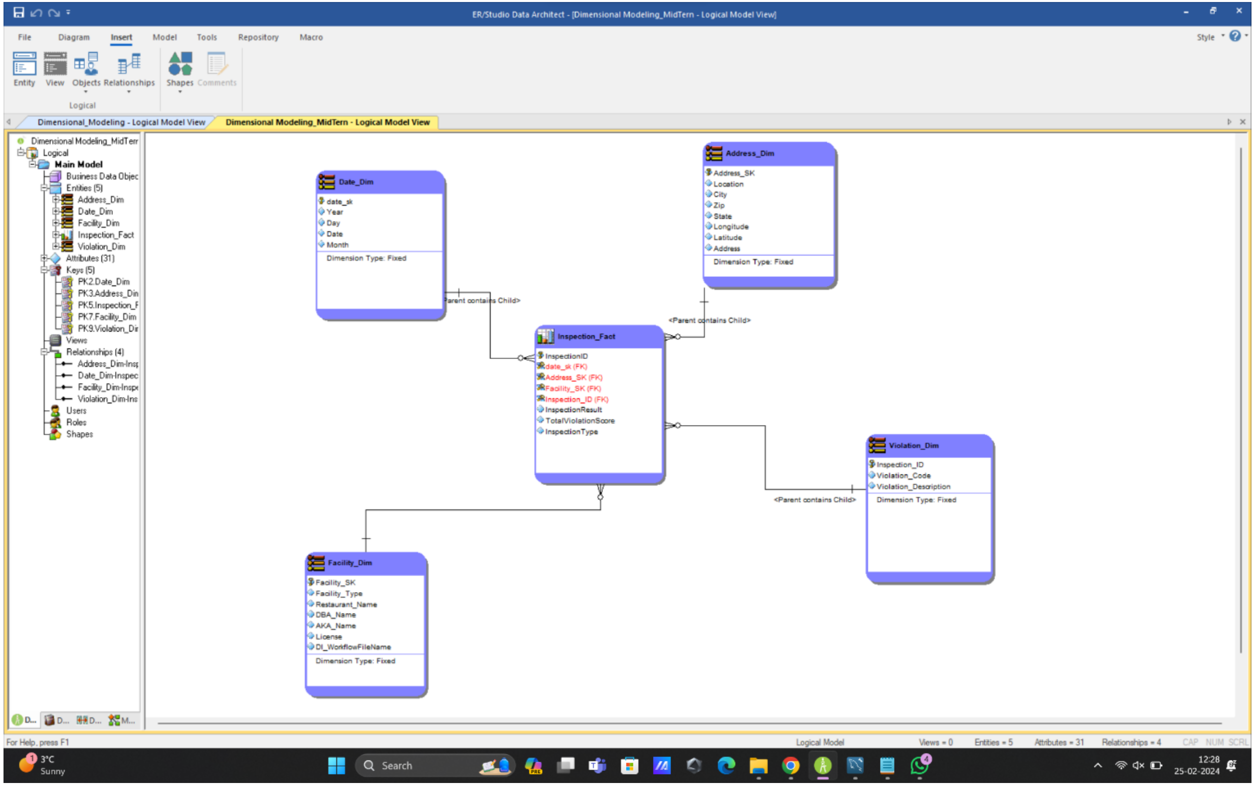
Chicago


jupyter Trying ydata profiling1 Last Checkpoint: 18 minutes ago
Trusted
File Edit View Run Kernel Settings Help
Code
JupyterLab
Python 3 (ipykernel)

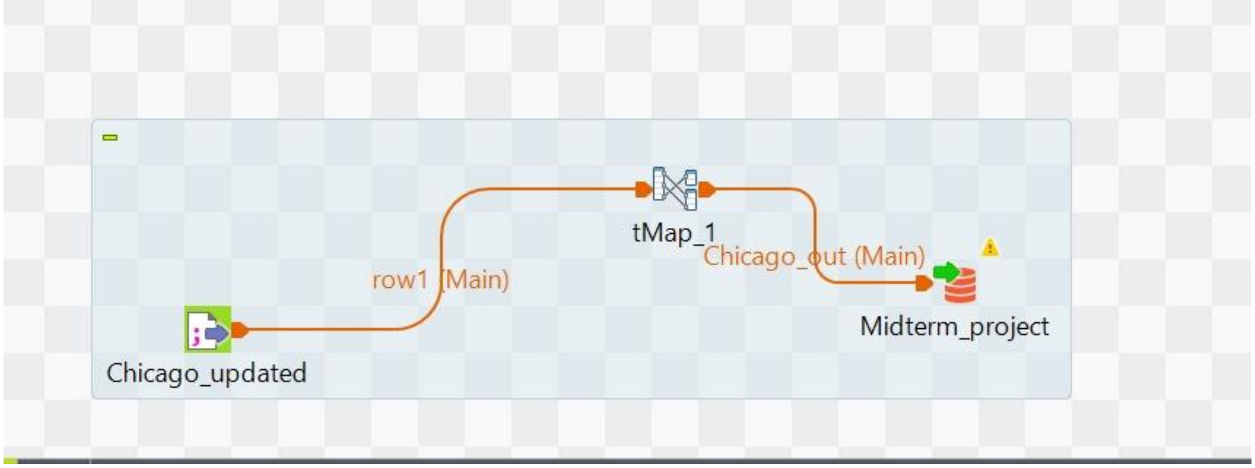
```
[7]: Dallas.to_file("report_for_food_inspection.html")
C:\Users\durge\anaconda3\Lib\site-packages\ydata_profiling\model\pandas\duplicates_pandas.py:40: PerformanceWarning: DataFrame is highly fragmented.
This is usually the result of calling `frame.insert` many times, which has poor performance.  Consider joining all columns at once using pd.concat(axi
s=1) instead. To get a de-fragmented frame, use `newframe = frame.copy()`
  .reset_index(name=duplicates_key)
C:\Users\durge\anaconda3\Lib\site-packages\ydata_profiling\model\pandas\duplicates_pandas.py:40: PerformanceWarning: DataFrame is highly fragmented.
This is usually the result of calling `frame.insert` many times, which has poor performance.  Consider joining all columns at once using pd.concat(axi
s=1) instead. To get a de-fragmented frame, use `newframe = frame.copy()`
  .reset_index(name=duplicates_key)
C:\Users\durge\anaconda3\Lib\site-packages\ydata_profiling\model\pandas\duplicates_pandas.py:40: PerformanceWarning: DataFrame is highly fragmented.
This is usually the result of calling `frame.insert` many times, which has poor performance.  Consider joining all columns at once using pd.concat(axi
s=1) instead. To get a de-fragmented frame, use `newframe = frame.copy()`
  .reset_index(name=duplicates_key)
Generate report structure: 100% 1/1 [02:31<00:00, 151.67s/it]
Render HTML: 100% 1/1 [00:08<00:00, 8.02s/it]
Export report to file: 100% 1/1 [00:00<00:00, 3.25it/s]

[8]: df2 = pd.read_csv('Food_Inspections_20240225.tsv', delimiter ='\t')

[9]: Chicago = ProfileReport(df,title='Chicago')

[10]: Chicago.to_file("Chicago_Report")
C:\Users\durge\anaconda3\Lib\site-packages\ydata_profiling\model\pandas\duplicates_pandas.py:40: PerformanceWarning: DataFrame is highly fragmented.
This is usually the result of calling `frame.insert` many times, which has poor performance.  Consider joining all columns at once using pd.concat(axi
s=1) instead. To get a de-fragmented frame, use `newframe = frame.copy()`
  .reset_index(name=duplicates_key)
C:\Users\durge\anaconda3\Lib\site-packages\ydata_profiling\model\pandas\duplicates_pandas.py:40: PerformanceWarning: DataFrame is highly fragmented.
This is usually the result of calling `frame.insert` many times, which has poor performance.  Consider joining all columns at once using pd.concat(axi
s=1) instead. To get a de-fragmented frame, use `newframe = frame.copy()`
  .reset_index(name=duplicates_key)
Generate report structure: 100% 1/1 [03:33<00:00, 213.28s/it]
Render HTML: 100% 1/1 [00:12<00:00, 12.92s/it]
C:\Users\durge\anaconda3\Lib\site-packages\ydata_profiling\profile_report.py:375: UserWarning: Extension  not supported. For now we assume .html was i
ntended. To remove this warning, please use .html or .json.
  warnings.warn(
Export report to file: 100% 1/1 [00:00<00:00, 3.88it/s]

[ ]:
```

Just like dallas, we have used YdataProfiling for Chicago as well to generate a data report for Chicago.

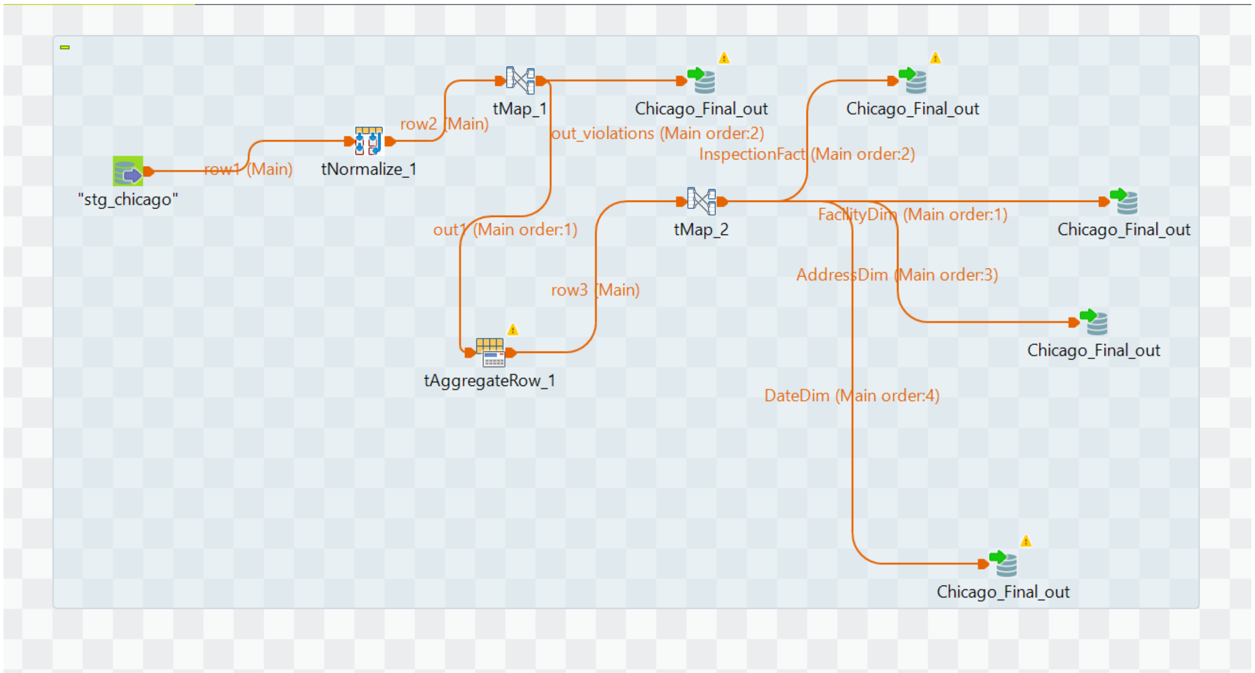
The following is the Dimensional Model that was created in ER Studio:



Workflow for Chicago Dataset.



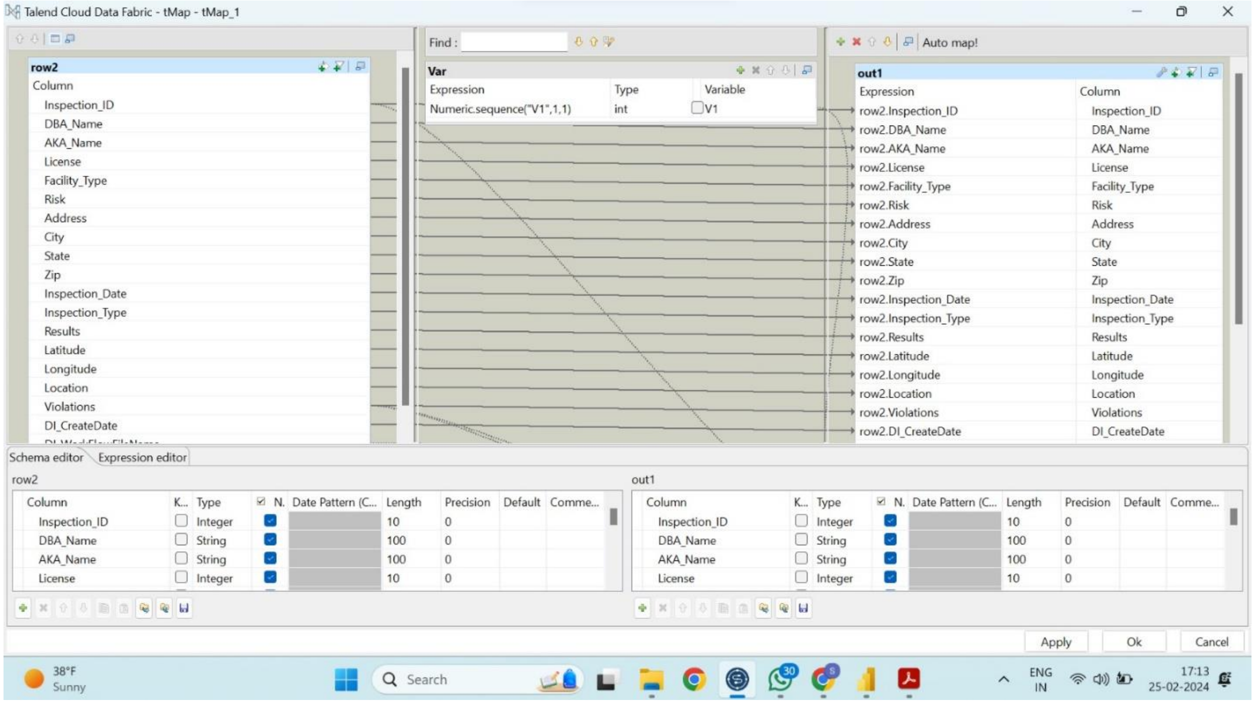
After data profiling, we used the file input delimited component to get the file with tmap component and database out as the component to stage the data in MySql.



Later we created the above ETL pipeline. Here we took the staged table as an input and used tnormalize to normalize the data then mapped with tmap and used it to map out the violations.

Later we used tAggregateRow and then used tmap to get the output for “Inspection Fact”, “Facility Dimension”, “Address Dimension” and “Date Dimension”.

The following is the tmap configuration used to map the normalized data which was used to sperate the violations



Talend Cloud Data Fabric - tMap - tMap_1

row2

Column
Inspection_ID
DBA_Name
AKA_Name
License
Facility_Type
Risk
Address
City
State
Zip
Inspection_Date
Inspection_Type
Results
Latitude
Longitude
Location
Violations
DI_CreateDate

Var

Expression	Type	Variable
Numeric.sequence("V1",1,1)	int	□ V1

out1

Expression	Column
row2.Inspection_ID	Inspection_ID
row2.DBA_Name	DBA_Name
row2.AKA_Name	AKA_Name
row2.License	License
row2.Facility_Type	Facility_Type
row2.Risk	Risk
row2.Address	Address
row2.City	City
row2.State	State
row2.Zip	Zip
row2.Inspection_Date	Inspection_Date
row2.Inspection_Type	Inspection_Type
row2.Results	Results
row2.Latitude	Latitude
row2.Longitude	Longitude
row2.Location	Location
row2.Violations	Violations
row2.DI_CreateDate	DI_CreateDate

Schema editor | Expression editor

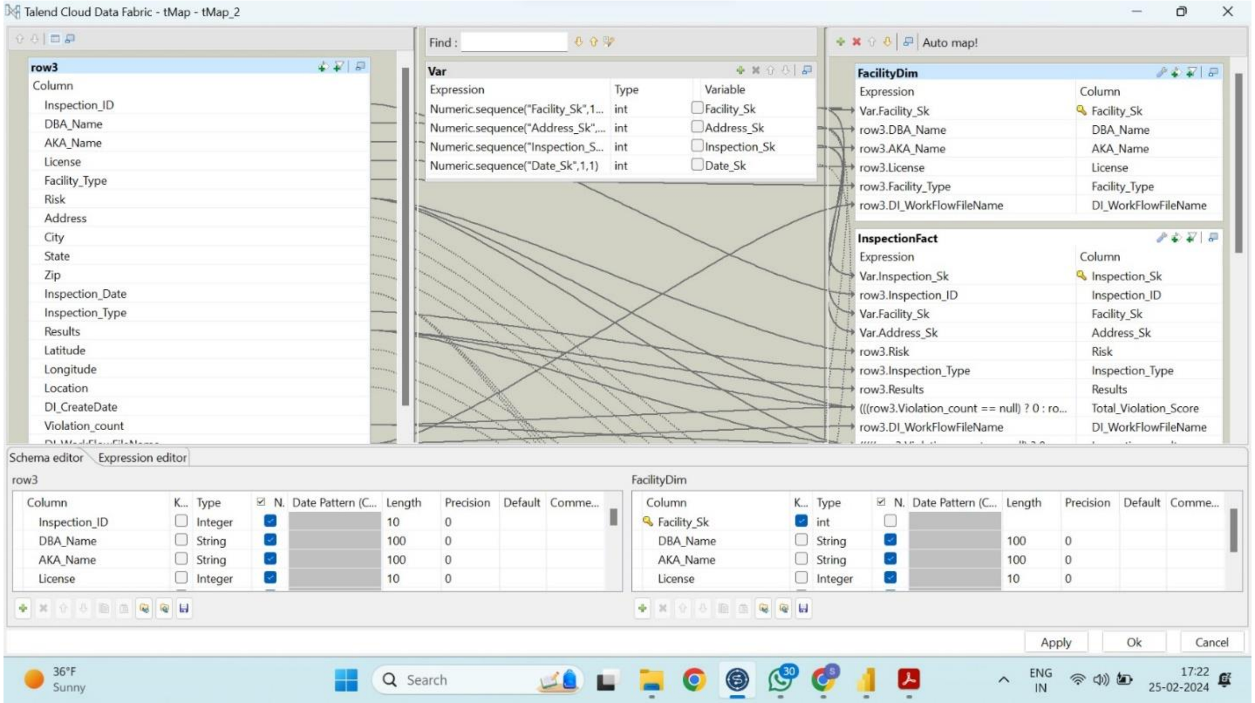
row2

Column	K...	Type	☑ N.	Date Pattern (C...	Length	Precision	Default	Comme...
Inspection_ID	□	Integer	☑		10	0		
DBA_Name	□	String	☑		100	0		
AKA_Name	□	String	☑		100	0		
License	□	Integer	☑		10	0		

out1

Column	K...	Type	☑ N.	Date Pattern (C...	Length	Precision	Default	Comme...
Inspection_ID	□	Integer	☑		10	0		
DBA_Name	□	String	☑		100	0		
AKA_Name	□	String	☑		100	0		
License	□	Integer	☑		10	0		

The following is the tmap configuration used to Aggregate the data and then populated in different dimensional and one fact table.



Talend Cloud Data Fabric - tMap - tMap_2

row3

Column
Inspection_ID
DBA_Name
AKA_Name
License
Facility_Type
Risk
Address
City
State
Zip
Inspection_Date
Inspection_Type
Results
Latitude
Longitude
Location
DI_CreateDate
Violation_count

Var

Expression	Type	Variable
Numeric.sequence("Facility_Sk",1...	int	□ Facility_Sk
Numeric.sequence("Address_Sk",...	int	□ Address_Sk
Numeric.sequence("Inspection_S...	int	□ Inspection_Sk
Numeric.sequence("Date_Sk",1,1)	int	□ Date_Sk

FacilityDim

Expression	Column
Var.Facility_Sk	Facility_Sk
row3.DBA_Name	DBA_Name
row3.AKA_Name	AKA_Name
row3.License	License
row3.Facility_Type	Facility_Type
row3.DI_WorkFlowFileName	DI_WorkFlowFileName

InspectionFact

Expression	Column
Var.Inspection_Sk	Inspection_Sk
row3.Inspection_ID	Inspection_ID
Var.Facility_Sk	Facility_Sk
Var.Address_Sk	Address_Sk
row3.Risk	Risk
row3.Inspection_Type	Inspection_Type
row3.Results	Results
(((row3.Violation_count == null) ? 0 : ro...	Total_Violation_Score
row3.DI_WorkFlowFileName	DI_WorkFlowFileName

Schema editor | Expression editor

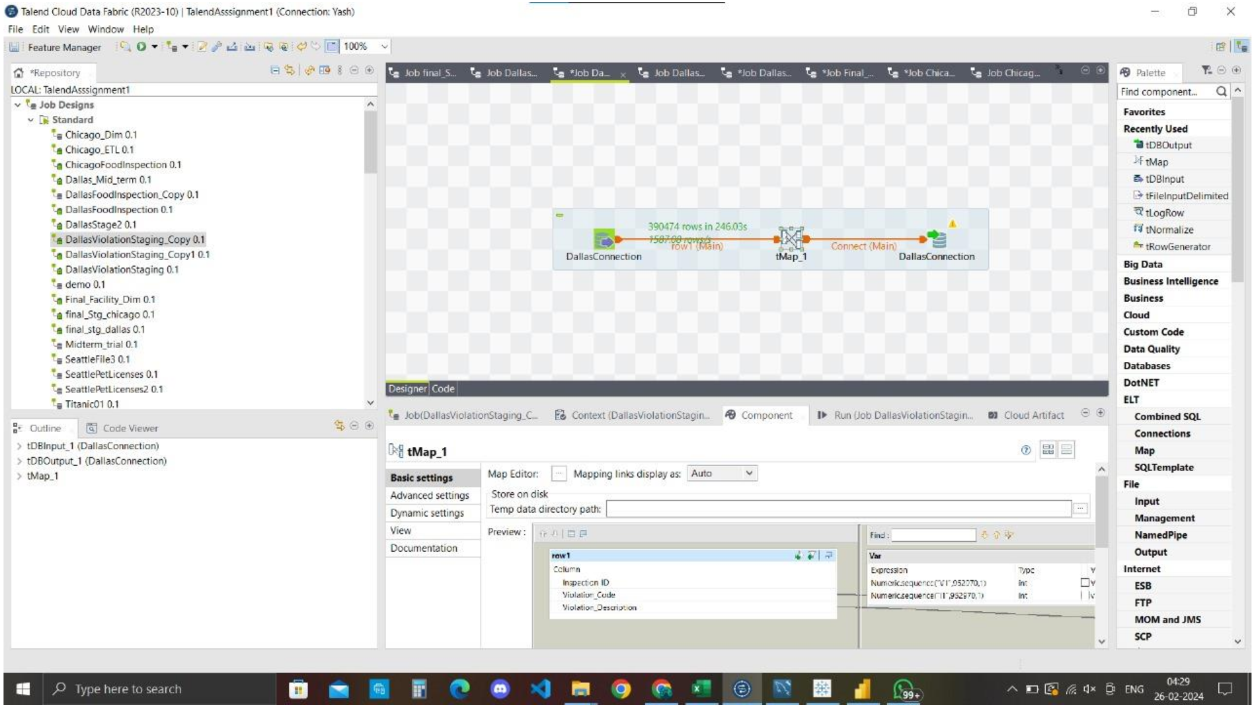
row3

Column	K...	Type	☑ N.	Date Pattern (C...	Length	Precision	Default	Comme...
Inspection_ID	□	Integer	☑		10	0		
DBA_Name	□	String	☑		100	0		
AKA_Name	□	String	☑		100	0		
License	□	Integer	☑		10	0		

FacilityDim

Column	K...	Type	☑ N.	Date Pattern (C...	Length	Precision	Default	Comme...
Facility_Sk	☑	int	□					
DBA_Name	□	String	☑		100	0		
AKA_Name	□	String	☑		100	0		
License	□	Integer	☑		10	0		

Staging of Dallas dataset



Map Editor: ... Mapping links display as: Auto

Store on disk

Temp data directory path:

Preview :

row1

Column
Inspection ID
Violation_Code
Violation_Description

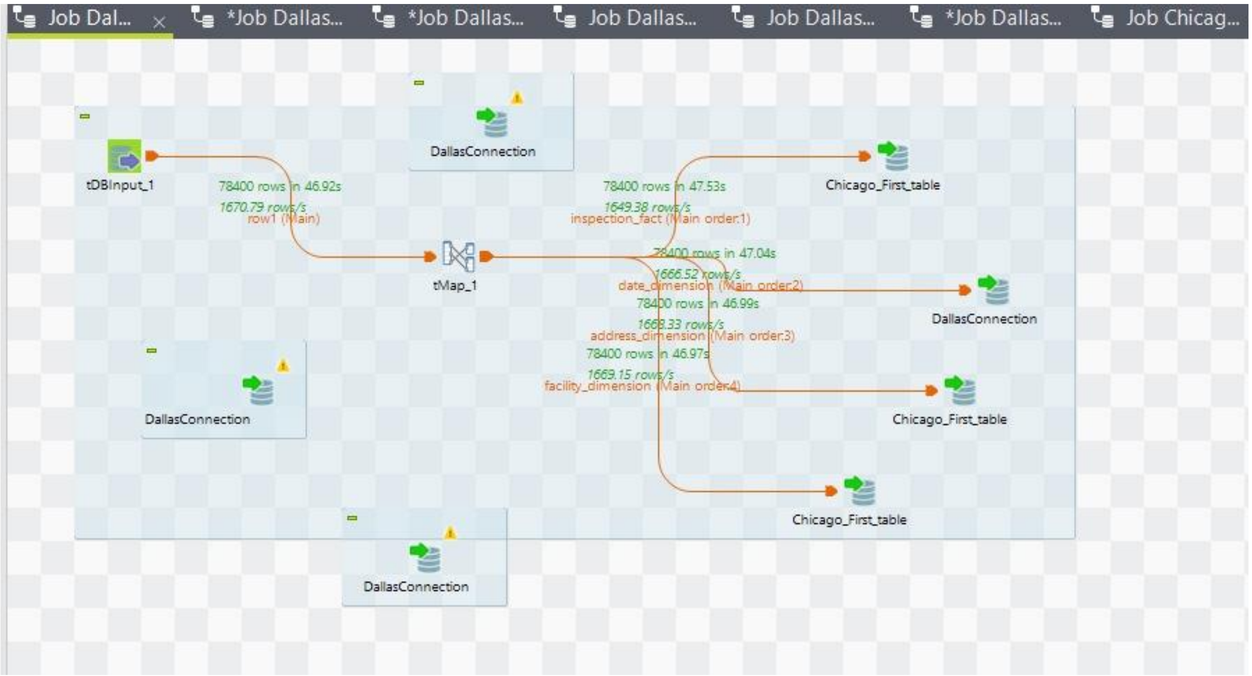
Find :

Var

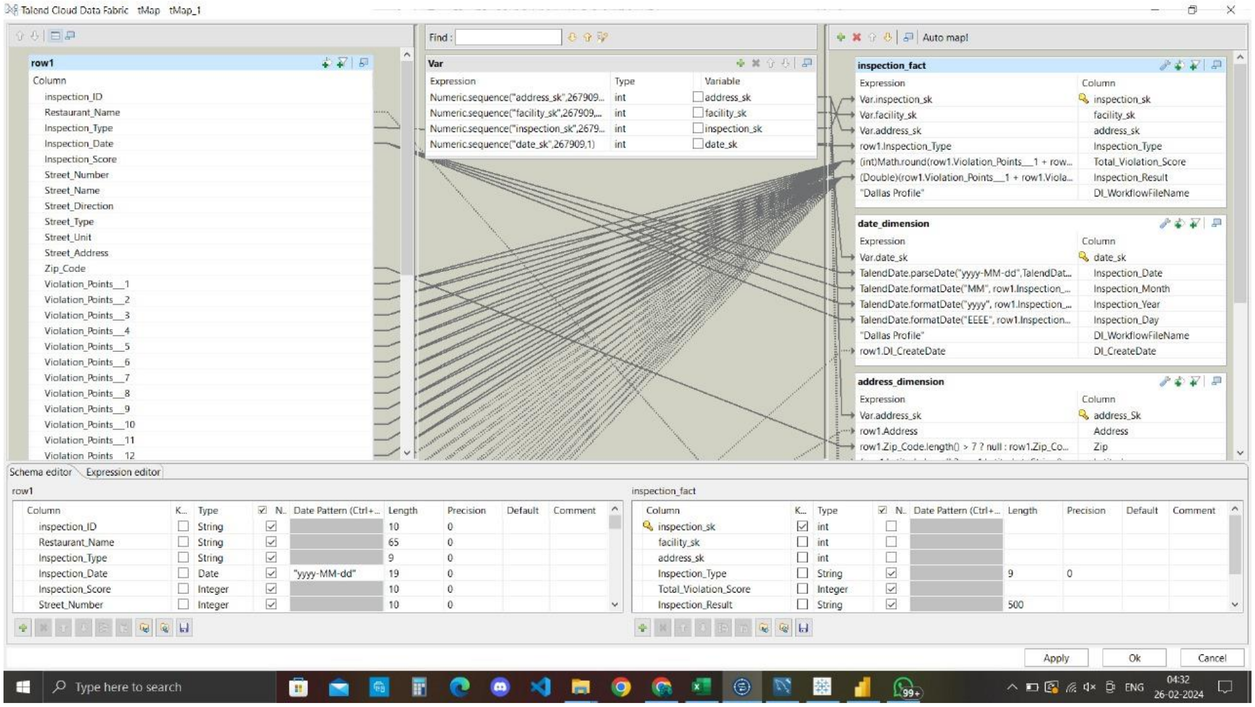
Expression	Type
Numeric.sequence("V1",952970,1)	int
Numeric.sequence("I1",952970,1)	int

Here we took an input mapped using tmap and then the mapped output is written to database.

s



the above is the ETL pipeline that was used where the input is the staged data of dallas and then this was mapped and the output was passed to different dimensions and one fact table.



Talend Cloud Data Fabric tMap tMap_1

row1

Column
inspection_ID
Restaurant_Name
Inspection_Type
Inspection_Date
Inspection_Score
Street_Number
Street_Name
Street_Direction
Street_Type
Street_Unit
Street_Address
Zip_Code
Violation_Points__1
Violation_Points__2
Violation_Points__3
Violation_Points__4
Violation_Points__5
Violation_Points__6
Violation_Points__7
Violation_Points__8
Violation_Points__9
Violation_Points__10
Violation_Points__11
Violation_Points__12

Var

Expression	Type	Variable
Numeric.sequence("address_sk",267909...	int	□ address_sk
Numeric.sequence("facility_sk",267909,...	int	□ facility_sk
Numeric.sequence("inspection_sk",2679...	int	□ inspection_sk
Numeric.sequence("date_sk",267909,1)	int	□ date_sk

inspection_fact

Expression	Column
Var.inspection_sk	inspection_sk
Var.facility_sk	facility_sk
Var.address_sk	address_sk
row1.Inspection_Type	Inspection_Type
(int)Math.round(row1.Violation_Points__1 + row...	Total_Violation_Score
(Double)(row1.Violation_Points__1 + row1.Viola...	Inspection_Result
"Dallas Profile"	DI_WorkflowFileName

date_dimension

Expression	Column
Var.date_sk	date_sk
TalendDate.parseDate("yyyy-MM-dd",TalendDat...	Inspection_Date
TalendDate.formatDate("MM", row1.Inspection_...	Inspection_Month
TalendDate.formatDate("yyyy", row1.Inspection_...	Inspection_Year
TalendDate.formatDate("EEEE", row1.Inspection...	Inspection_Day
"Dallas Profile"	DI_WorkflowFileName
row1.DI_CreateDate	DI_CreateDate

address_dimension

Expression	Column
Var.address_sk	address_Sk
row1.Address	Address
row1.Zip_Code.length() > 7 ? null : row1.Zip_Co...	Zip

Schema editor | Expression editor

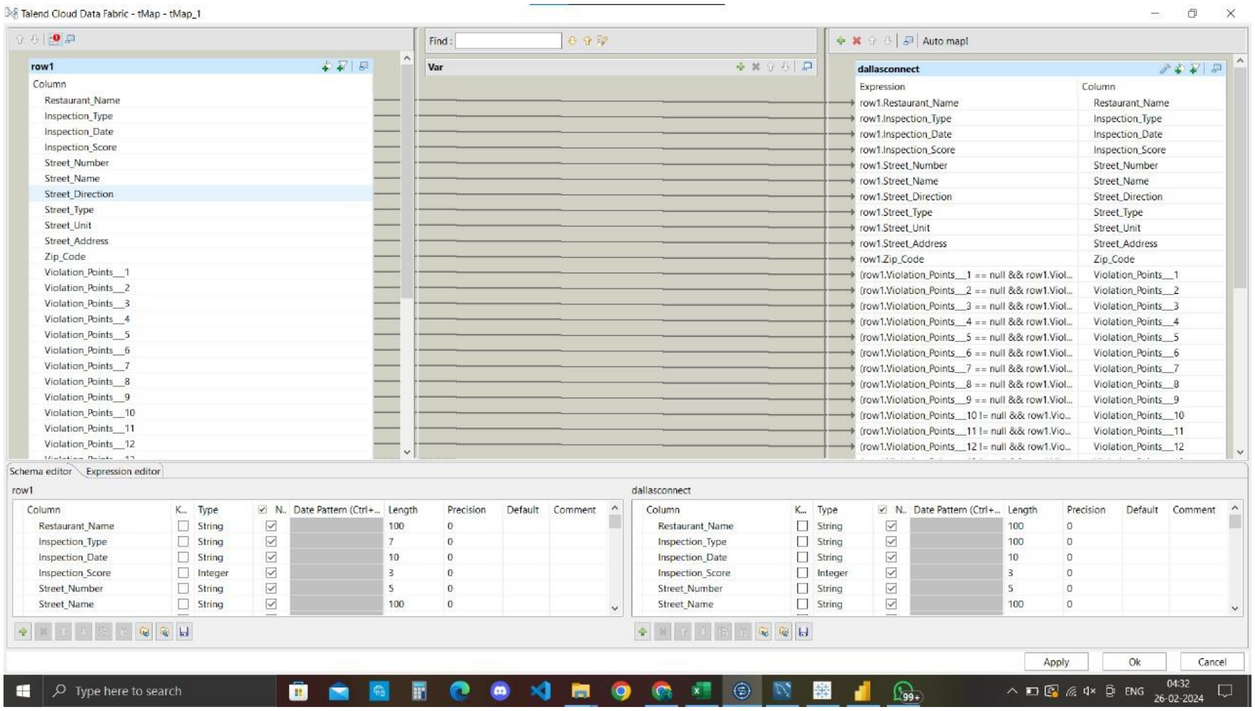
row1

Column	K...	Type	☑ N..	Date Pattern (Ctrl+...	Length	Precision	Default	Comment
inspection_ID	□	String	☑		10	0		
Restaurant_Name	□	String	☑		65	0		
Inspection_Type	□	String	☑		9	0		
Inspection_Date	□	Date	☑	"yyyy-MM-dd"	19	0		
Inspection_Score	□	Integer	☑		10	0		
Street_Number	□	Integer	☑		10	0		

inspection_fact

Column	K...	Type	☑ N..	Date Pattern (Ctrl+...	Length	Precision	Default	Comment
inspection_sk	☑	int	□					
facility_sk	□	int	□					
address_sk	□	int	□					
Inspection_Type	□	String	☑		9	0		
Total_Violation_Score	□	Integer	☑					
Inspection_Result	□	String	☑		500			

The above is the tmap used for putting the data in different dimensions and the fact table.



Talend Cloud Data Fabric - tMap - tMap_1

row1

Column
Restaurant_Name
Inspection_Type
Inspection_Date
Inspection_Score
Street_Number
Street_Name
Street_Direction
Street_Type
Street_Unit
Street_Address
Zip_Code
Violation_Points__1
Violation_Points__2
Violation_Points__3
Violation_Points__4
Violation_Points__5
Violation_Points__6
Violation_Points__7
Violation_Points__8
Violation_Points__9
Violation_Points__10
Violation_Points__11
Violation_Points__12

dallasconnect

Expression	Column
row1.Restaurant_Name	Restaurant_Name
row1.Inspection_Type	Inspection_Type
row1.Inspection_Date	Inspection_Date
row1.Inspection_Score	Inspection_Score
row1.Street_Number	Street_Number
row1.Street_Name	Street_Name
row1.Street_Direction	Street_Direction
row1.Street_Type	Street_Type
row1.Street_Unit	Street_Unit
row1.Street_Address	Street_Address
row1.Zip_Code	Zip_Code
(row1.Violation_Points__1 == null && row1.Viol...	Violation_Points__1
(row1.Violation_Points__2 == null && row1.Viol...	Violation_Points__2
(row1.Violation_Points__3 == null && row1.Viol...	Violation_Points__3
(row1.Violation_Points__4 == null && row1.Viol...	Violation_Points__4
(row1.Violation_Points__5 == null && row1.Viol...	Violation_Points__5
(row1.Violation_Points__6 == null && row1.Viol...	Violation_Points__6
(row1.Violation_Points__7 == null && row1.Viol...	Violation_Points__7
(row1.Violation_Points__8 == null && row1.Viol...	Violation_Points__8
(row1.Violation_Points__9 == null && row1.Viol...	Violation_Points__9
(row1.Violation_Points__10 != null && row1.Vio...	Violation_Points__10
(row1.Violation_Points__11 != null && row1.Vio...	Violation_Points__11
(row1.Violation_Points__12 != null && row1.Vio...	Violation_Points__12

Schema editor | Expression editor

row1

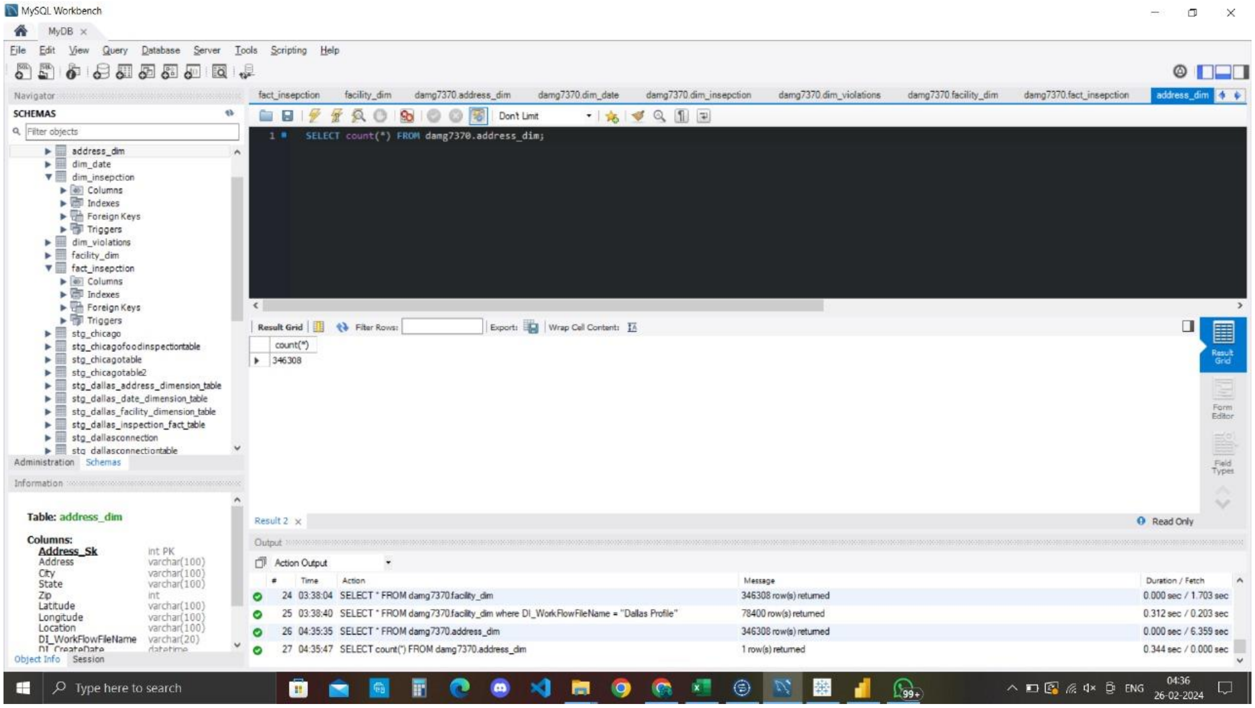
Column	K...	Type	☑ N..	Date Pattern (Ctrl+...	Length	Precision	Default	Comment
Restaurant_Name	□	String	☑		100	0		
Inspection_Type	□	String	☑		7	0		
Inspection_Date	□	String	☑		10	0		
Inspection_Score	□	Integer	☑		3	0		
Street_Number	□	String	☑		5	0		
Street_Name	□	String	☑		100	0		

dallasconnect

Column	K...	Type	☑ N..	Date Pattern (Ctrl+...	Length	Precision	Default	Comment
Restaurant_Name	□	String	☑		100	0		
Inspection_Type	□	String	☑		100	0		
Inspection_Date	□	String	☑		10	0		
Inspection_Score	□	Integer	☑		3	0		
Street_Number	□	String	☑		5	0		
Street_Name	□	String	☑		100	0		

This is the tmap configuration used to stage the dallas dataset.

The following are the count row queries for all the dimensions and one fact table



```
SELECT count(*) FROM damg7370.address_dim;
```

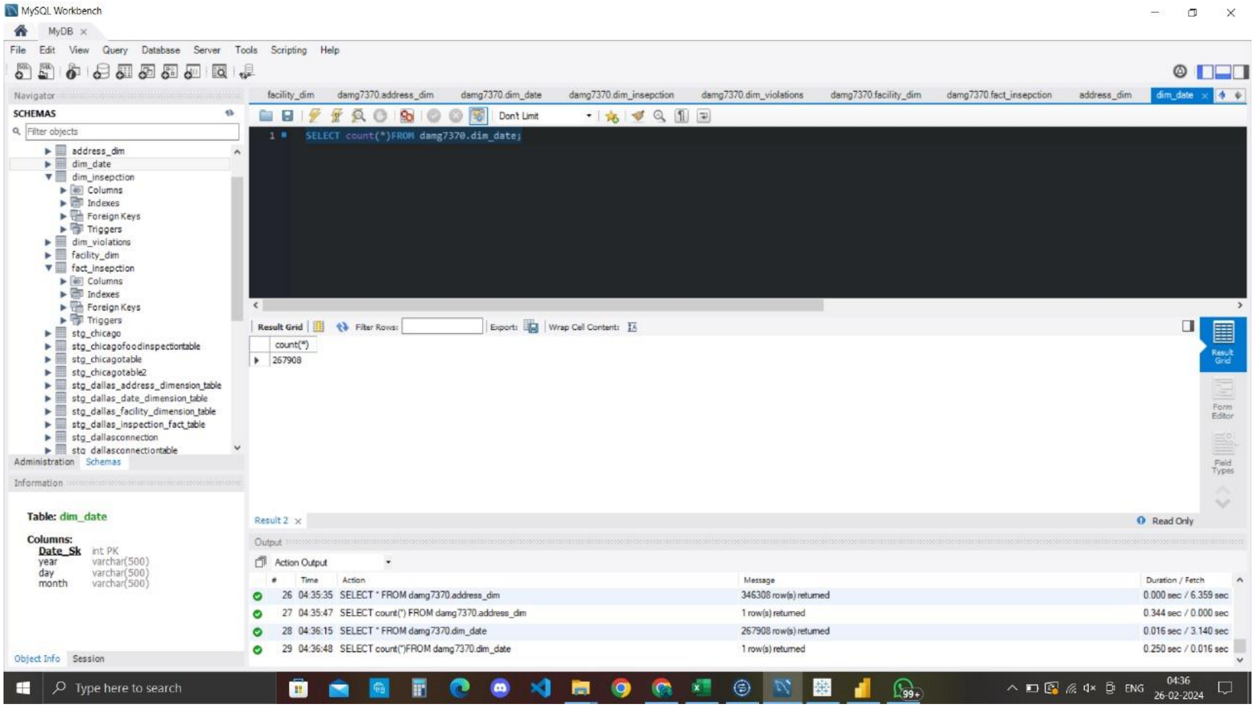
count(*)
346308

Table: address_dim

Columns:

Address_Sk	int PK
Address	varchar(100)
City	varchar(100)
State	varchar(100)
Zip	int
Latitude	varchar(100)
Longitude	varchar(100)
Location	varchar(100)
DI_WorkFlowFileName	varchar(20)
DI_CreateDate	datetime

```
#   Time      Action                                                                       Message                   Duration / Fetch
24  03:38:04  SELECT * FROM damg7370.facility_dim                                          346308 row(s) returned    0.000 sec / 1.703 sec
25  03:38:40  SELECT * FROM damg7370.facility_dim where DI_WorkFlowFileName = "Dallas Profile"  78400 row(s) returned  0.312 sec / 0.203 sec
26  04:35:35  SELECT * FROM damg7370.address_dim                                           346308 row(s) returned    0.000 sec / 6.359 sec
27  04:35:47  SELECT count(*) FROM damg7370.address_dim                                    1 row(s) returned         0.344 sec / 0.000 sec
```



```
1  SELECT count(*)FROM damg7370.dim_date;
```

count(*)
267908

Table: dim_date

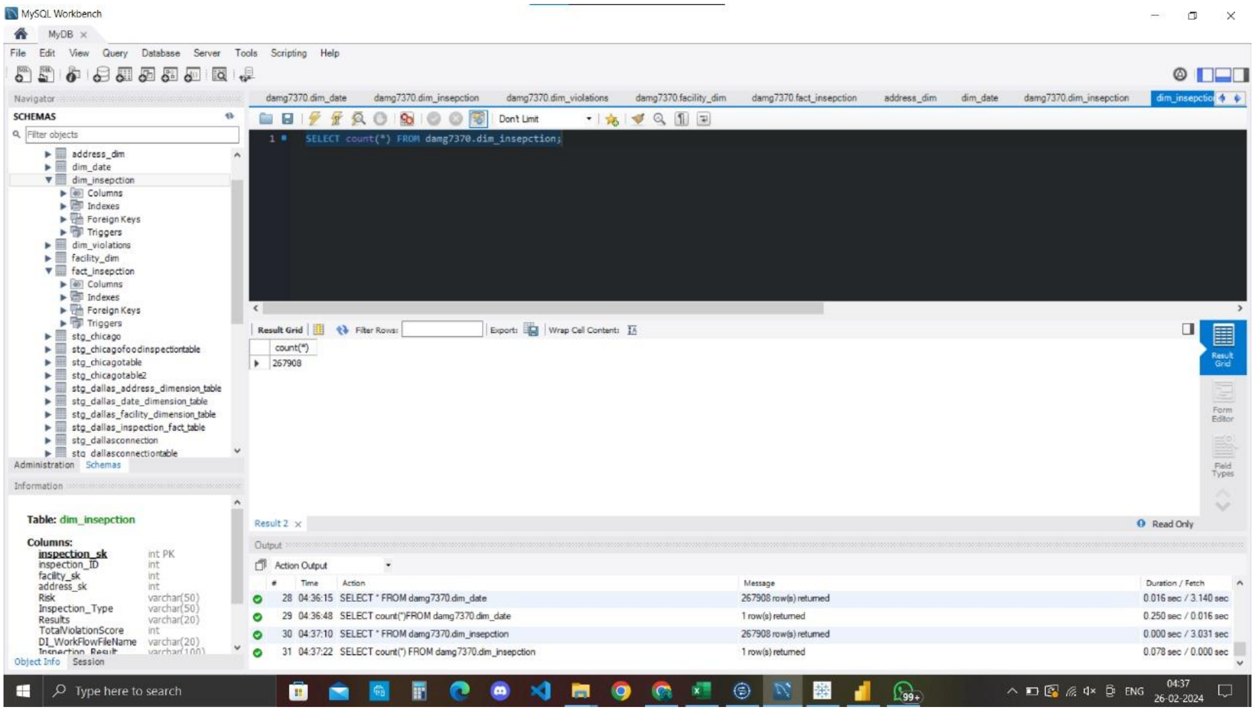
Columns:

```
Date_Sk   int PK
year      varchar(500)
day       varchar(500)
month     varchar(500)
```

Output

Action Output

```
#   Time      Action                                        Message                  Duration / Fetch
26  04:35:35  SELECT * FROM damg7370.address_dim            346308 row(s) returned   0.000 sec / 6.359 sec
27  04:35:47  SELECT count(*) FROM damg7370.address_dim     1 row(s) returned        0.344 sec / 0.000 sec
28  04:36:15  SELECT * FROM damg7370.dim_date               267908 row(s) returned   0.016 sec / 3.140 sec
29  04:36:48  SELECT count(*)FROM damg7370.dim_date         1 row(s) returned        0.250 sec / 0.016 sec
```



```
1  SELECT count(*) FROM damg7370.dim_insepction;
```

count(*)
267908

Table: dim_insepction

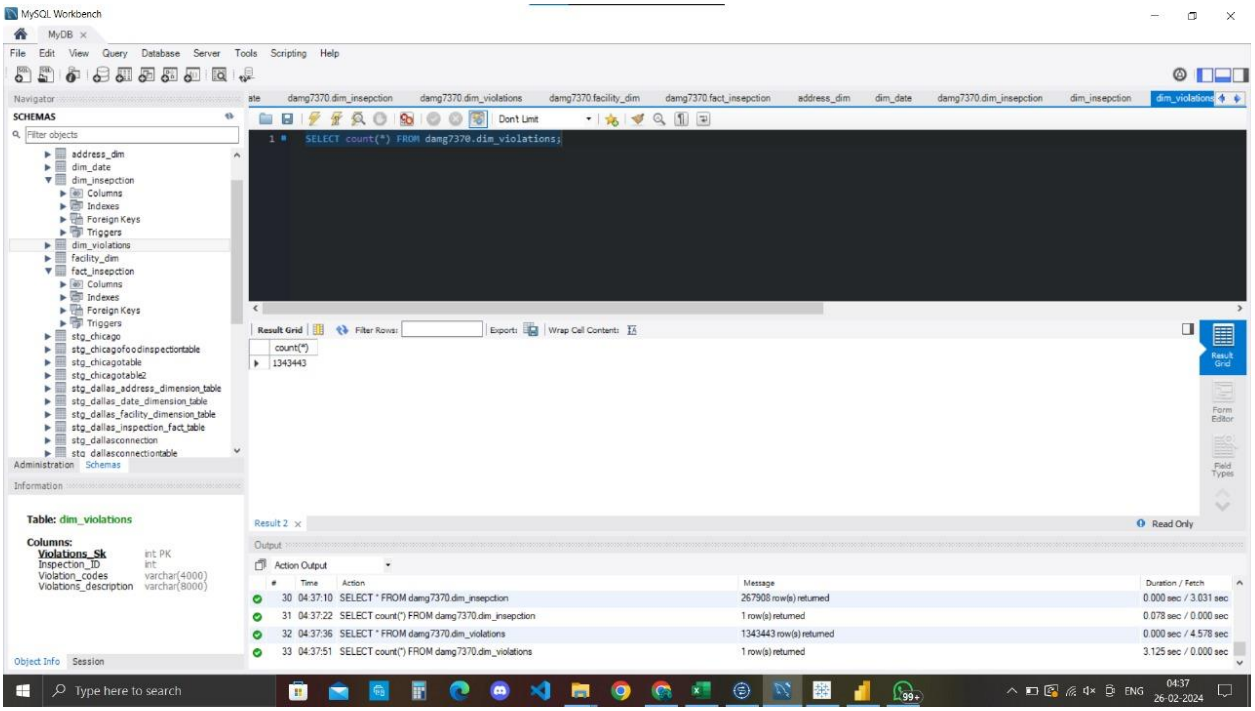
Columns:

```
inspection_sk          int PK
inspection_ID          int
facility_sk            int
address_sk             int
Risk                   varchar(50)
Inspection_Type        varchar(50)
Results                varchar(20)
TotalViolationScore    int
DI_WorkFlowFileName    varchar(20)
Inspection_Result      varchar(100)
```

Output

Action Output

```
#   Time      Action                                          Message                  Duration / Fetch
28  04:36:15  SELECT * FROM damg7370.dim_date                 267908 row(s) returned   0.016 sec / 3.140 sec
29  04:36:48  SELECT count(*)FROM damg7370.dim_date           1 row(s) returned        0.250 sec / 0.016 sec
30  04:37:10  SELECT * FROM damg7370.dim_insepction           267908 row(s) returned   0.000 sec / 3.031 sec
31  04:37:22  SELECT count(*) FROM damg7370.dim_insepction    1 row(s) returned        0.078 sec / 0.000 sec
```



```sql
SELECT count(*) FROM damg7370.dim_violations;
```

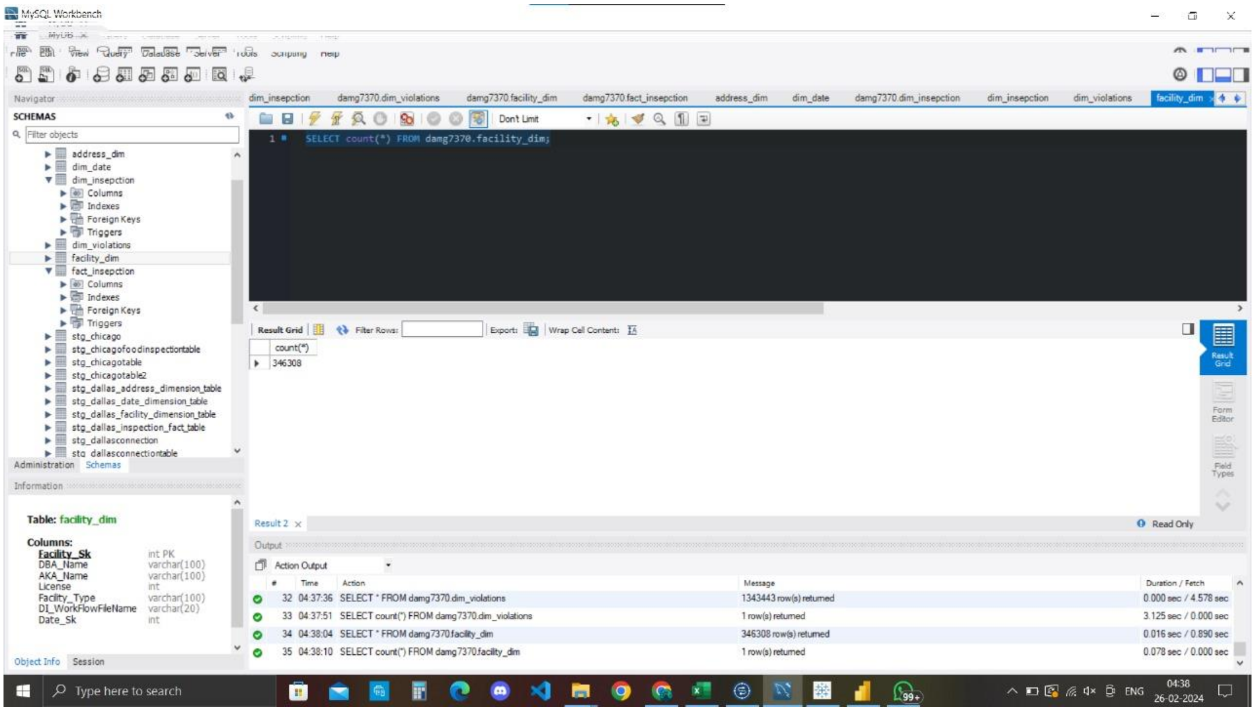
```
count(*)
1343443
```

Table: dim_violations

Columns:

```
Violations_Sk          int PK
Inspection_ID          int
Violation_codes        varchar(4000)
Violations_description varchar(8000)
```

```
#   Time      Action                                           Message                    Duration / Fetch
30  04:37:10  SELECT * FROM damg7370.dim_insepction            267908 row(s) returned     0.000 sec / 3.031 sec
31  04:37:22  SELECT count(*) FROM damg7370.dim_insepction     1 row(s) returned          0.078 sec / 0.000 sec
32  04:37:36  SELECT * FROM damg7370.dim_violations            1343443 row(s) returned    0.000 sec / 4.578 sec
33  04:37:51  SELECT count(*) FROM damg7370.dim_violations     1 row(s) returned          3.125 sec / 0.000 sec
```



```sql
SELECT count(*) FROM damg7370.facility_dim;
```

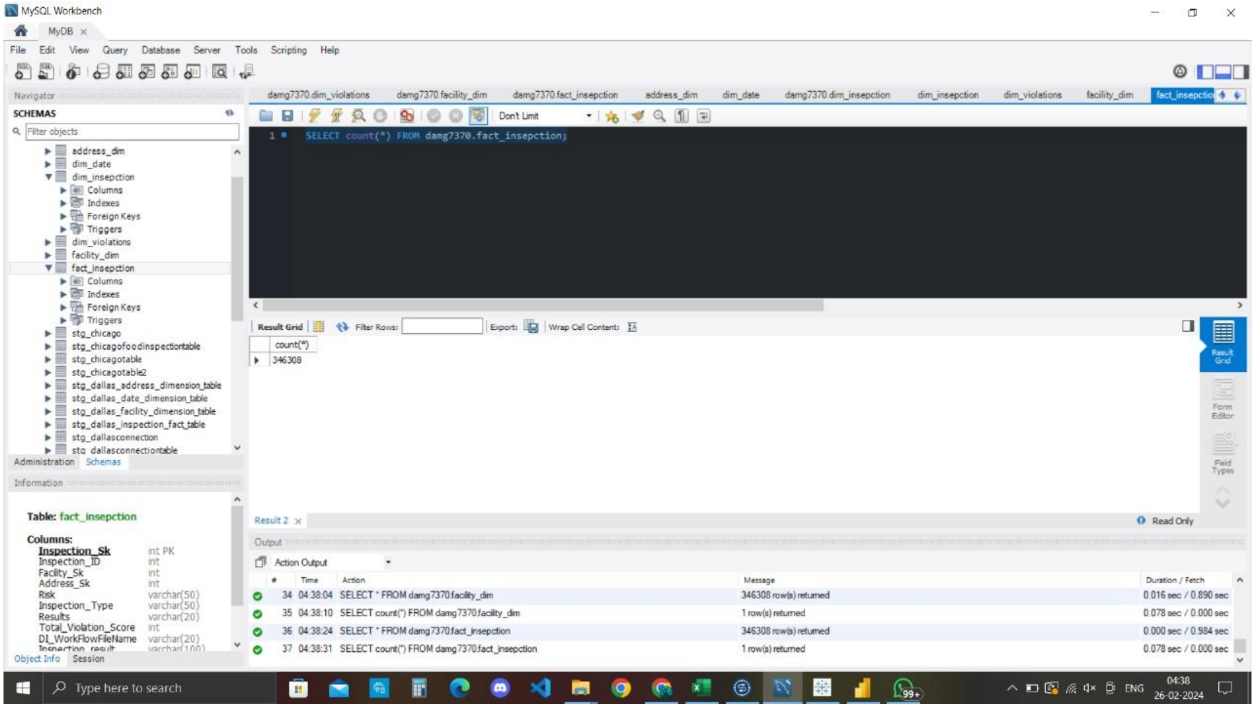
```
count(*)
346308
```

Table: facility_dim

Columns:

```
Facility_Sk          int PK
DBA_Name             varchar(100)
AKA_Name             varchar(100)
License              int
Facility_Type        varchar(100)
DI_WorkFlowFileName  varchar(20)
Date_Sk              int
```

```
#   Time      Action                                           Message                    Duration / Fetch
32  04:37:36  SELECT * FROM damg7370.dim_violations            1343443 row(s) returned    0.000 sec / 4.578 sec
33  04:37:51  SELECT count(*) FROM damg7370.dim_violations     1 row(s) returned          3.125 sec / 0.000 sec
34  04:38:04  SELECT * FROM damg7370.facility_dim              346308 row(s) returned     0.016 sec / 0.890 sec
35  04:38:10  SELECT count(*) FROM damg7370.facility_dim       1 row(s) returned          0.078 sec / 0.000 sec
```



```
SELECT count(*) FROM damg7370.fact_insepction;
```

```
count(*)
346308
```

Table: fact_insepction

Columns:

```
Inspection_Sk           int PK
Inspection_ID           int
Facility_Sk             int
Address_Sk              int
Risk                    varchar(50)
Inspection_Type         varchar(50)
Results                 varchar(20)
Total_Violation_Score   int
DI_WorkFlowFileName     varchar(20)
Inspection_result       varchar(100)
```

```
#   Time      Action                                          Message                  Duration / Fetch
34  04:38:04  SELECT * FROM damg7370.facility_dim             346308 row(s) returned   0.016 sec / 0.890 sec
35  04:38:10  SELECT count(*) FROM damg7370.facility_dim      1 row(s) returned        0.078 sec / 0.000 sec
36  04:38:24  SELECT * FROM damg7370.fact_insepction          346308 row(s) returned   0.000 sec / 0.984 sec
37  04:38:31  SELECT count(*) FROM damg7370.fact_insepction   1 row(s) returned        0.078 sec / 0.000 sec
```

The following are the DDL scripts for the above tables:

Address_Dim

```
CREATE TABLE `address_dim` (
  `Address_Sk` int NOT NULL,
  `Address` varchar(100) DEFAULT NULL,
  `City` varchar(100) DEFAULT NULL,
  `State` varchar(100) DEFAULT NULL,
  `Zip` int DEFAULT NULL,
  `Latitude` varchar(100) DEFAULT NULL,
  `Longitude` varchar(100) DEFAULT NULL,
  `Location` varchar(100) DEFAULT NULL,
  `DI_WorkFlowFileName` varchar(20) DEFAULT NULL,
  `DI_CreateDate` datetime DEFAULT NULL,
  PRIMARY KEY (`Address_Sk`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

dim_Date

```
CREATE TABLE `dim_date` (
  `Date_Sk` int NOT NULL,
  `year` varchar(500) DEFAULT NULL,
  `day` varchar(500) DEFAULT NULL,
  `month` varchar(500) DEFAULT NULL,
  PRIMARY KEY (`Date_Sk`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

dim_Inspection

```
CREATE TABLE `dim_insepction` (
  `inspection_sk` int NOT NULL,
  `inspection_ID` int DEFAULT NULL,
  `facility_sk` int NOT NULL,
  `address_sk` int NOT NULL,
  `Risk` varchar(50) DEFAULT NULL,
  `Inspection_Type` varchar(50) DEFAULT NULL,
  `Results` varchar(20) DEFAULT NULL,
  `TotalViolationScore` int DEFAULT NULL,
  `DI_WorkFlowFileName` varchar(20) DEFAULT NULL,
  `Inspection_Result` varchar(100) DEFAULT NULL,
  `Date_Sk` int NOT NULL,
  PRIMARY KEY (`inspection_sk`)
) ENGINE=InnoDB DEFAULT CHARFSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

dim_violations

```
CREATE TABLE `dim_violations` (
  `Violations_Sk` int NOT NULL,
  `Inspection_ID` int DEFAULT NULL,
  `Violation_codes` varchar(4000) DEFAULT NULL,
  `Violations_description` varchar(8000) DEFAULT NULL,
  `DI_WorkFlowFileName` varchar(100) DEFAULT NULL,
  PRIMARY KEY (`Violations_Sk`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

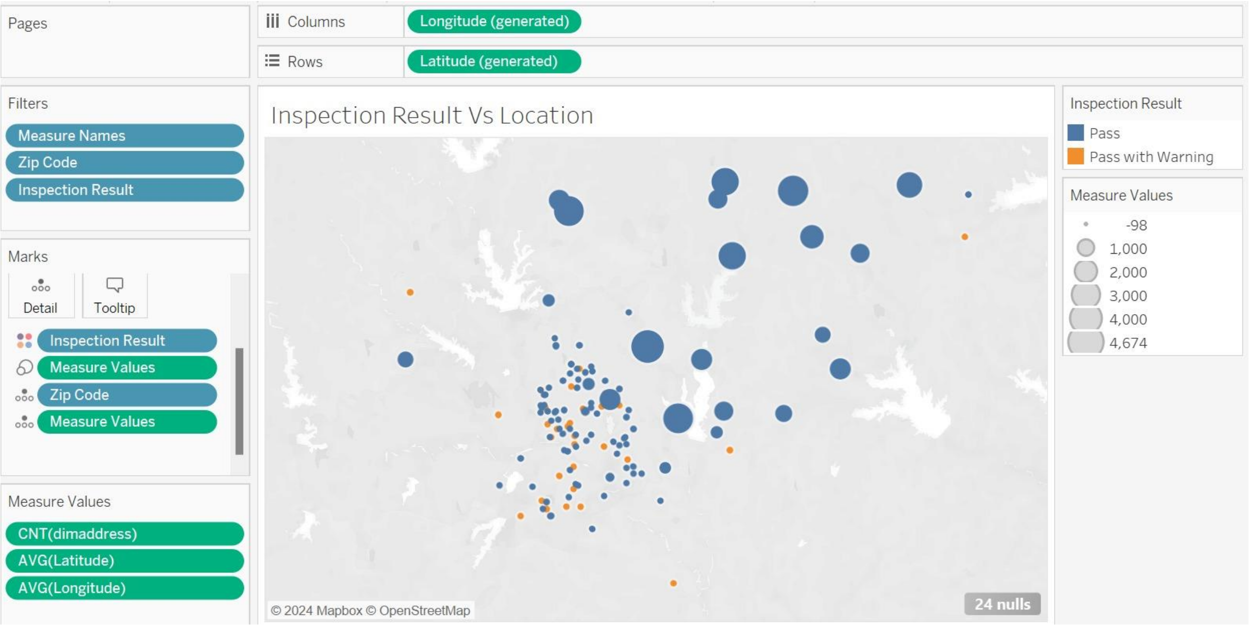
Facility_Dim

```
CREATE TABLE `facility_dim` (
  `Facility_Sk` int NOT NULL,
  `DBA_Name` varchar(100) DEFAULT NULL,
  `AKA_Name` varchar(100) DEFAULT NULL,
  `License` int DEFAULT NULL,
  `Facility_Type` varchar(100) DEFAULT NULL,
  `DI_WorkFlowFileName` varchar(20) DEFAULT NULL,
  `Date_Sk` int NOT NULL,
  `DI_CreateDate` datetime DEFAULT NULL,
  PRIMARY KEY (`Facility_Sk`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Fact_Inspection

```
CREATE TABLE `fact_insepction` (
  `Inspection_Sk` int NOT NULL,
  `Inspection_ID` int DEFAULT NULL,
  `Facility_Sk` int NOT NULL,
  `Address_Sk` int NOT NULL,
  `Risk` varchar(50) DEFAULT NULL,
  `Inspection_Type` varchar(50) DEFAULT NULL,
  `Results` varchar(20) DEFAULT NULL,
  `Total_Violation_Score` int DEFAULT NULL,
  `DI_WorkFlowFileName` varchar(20) DEFAULT NULL,
  `Inspection_result` varchar(100) DEFAULT NULL,
  PRIMARY KEY (`Inspection_Sk`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Dashboard Screenshots



Pages

Columns: Longitude (generated)

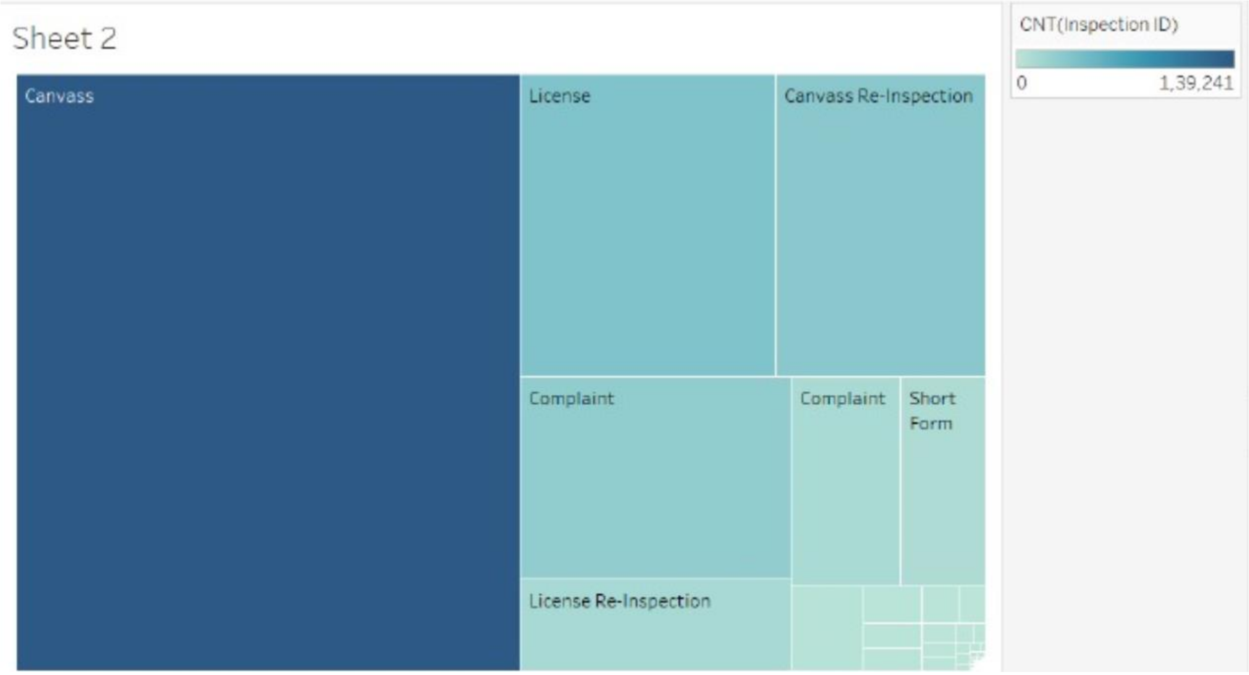
Rows: Latitude (generated)

Filters
- Measure Names
- Zip Code
- Inspection Result

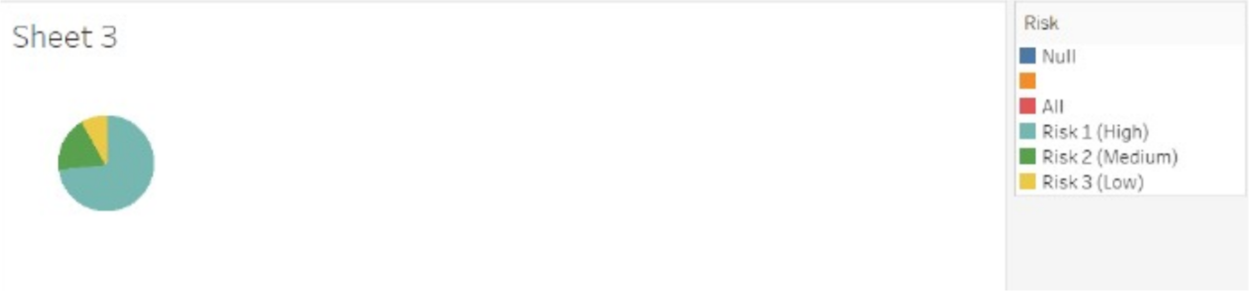
Marks
- Detail
- Tooltip
- Inspection Result
- Measure Values
- Zip Code
- Measure Values

Measure Values
- CNT(dimaddress)
- AVG(Latitude)
- AVG(Longitude)

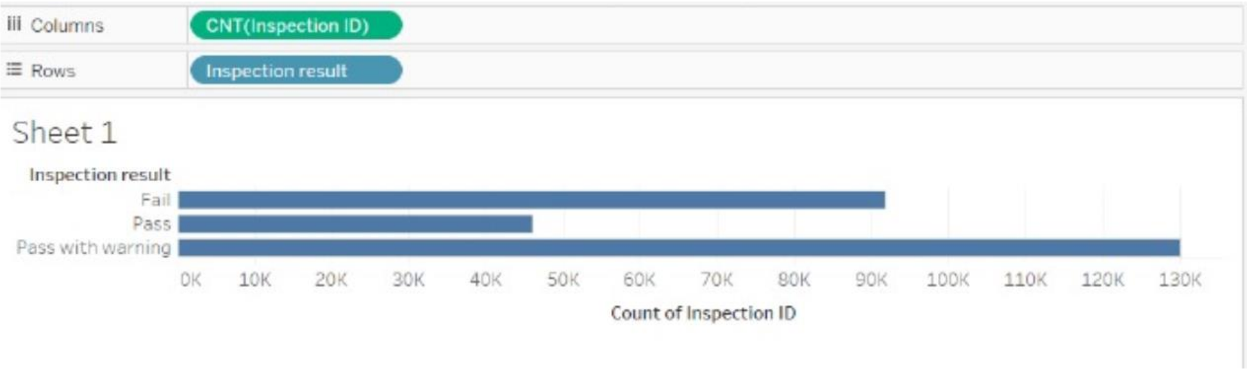
Inspection Result Vs Location



Sheet 2

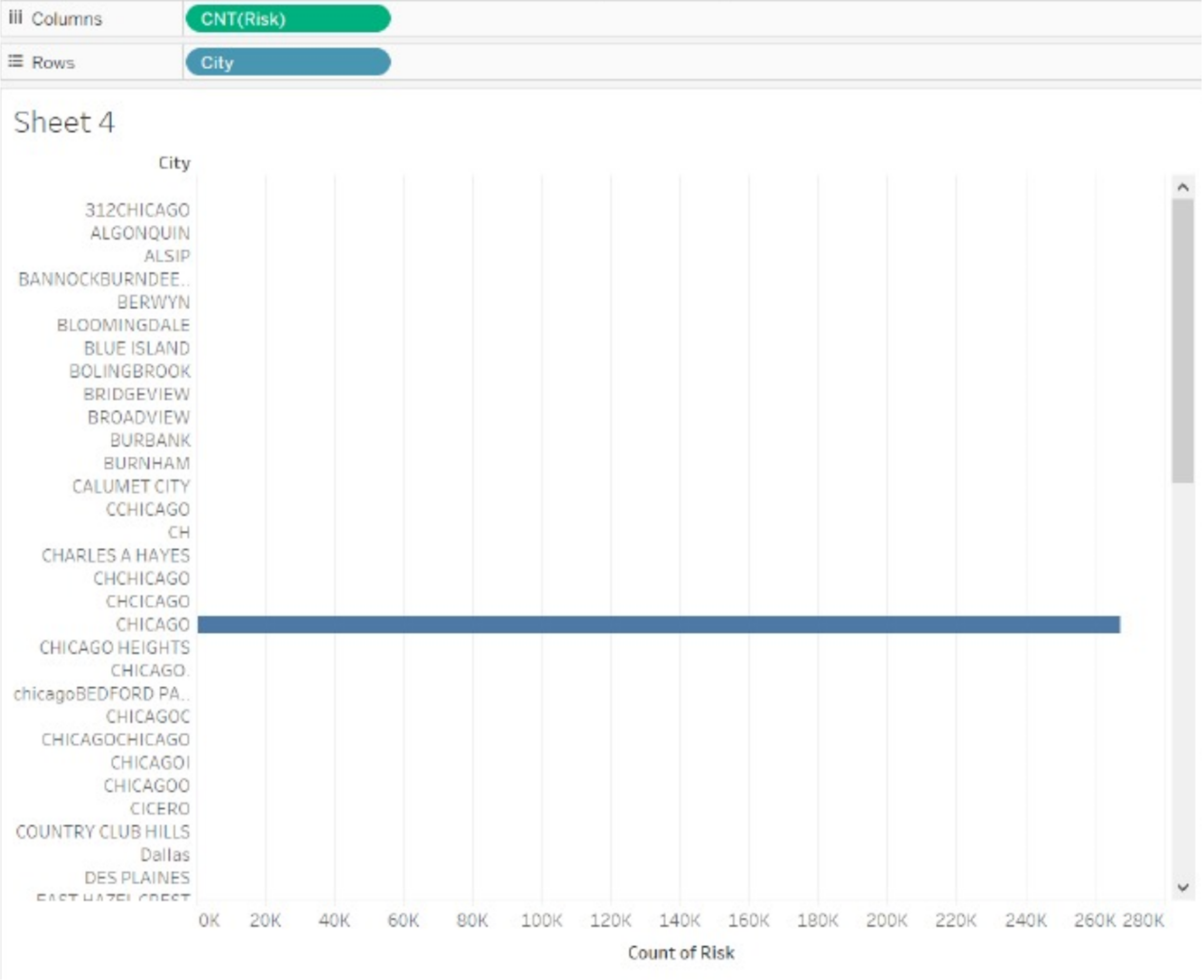


Sheet 3



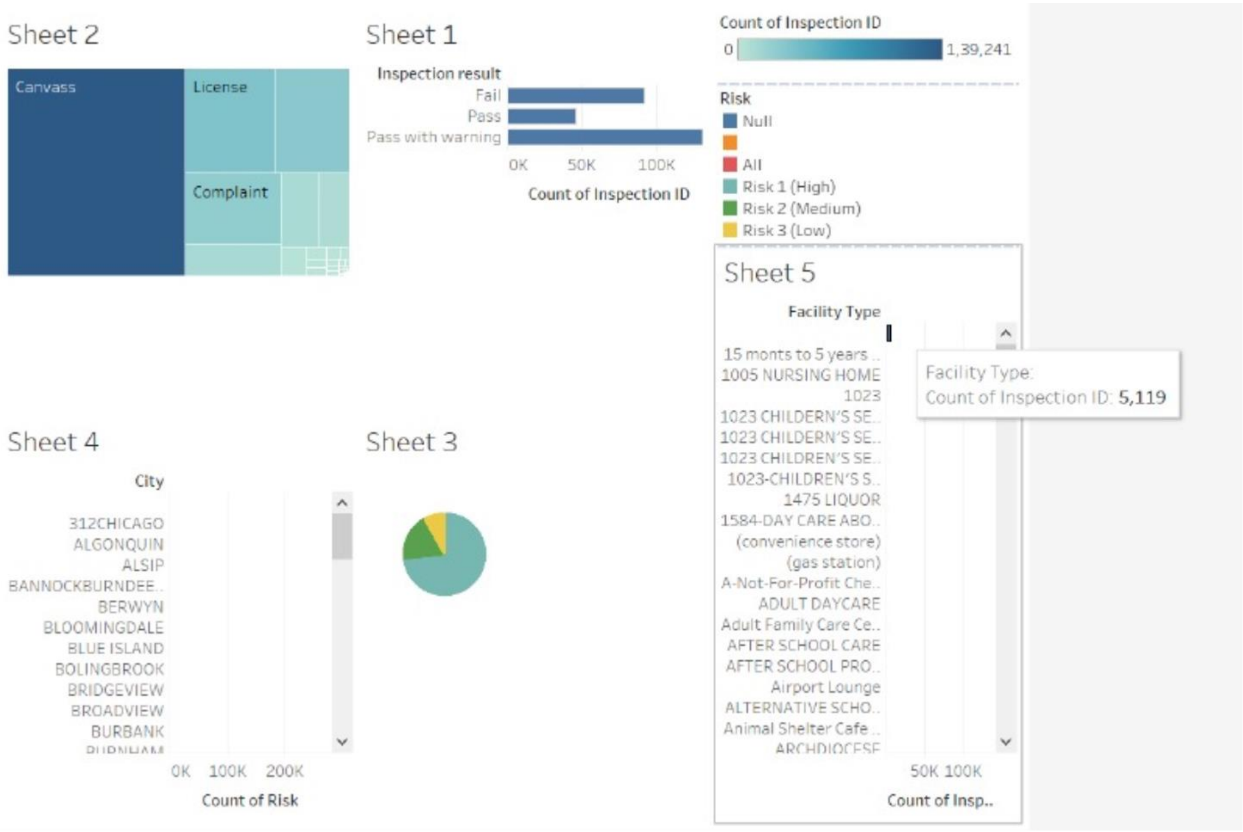
Columns: CNT(Inspection ID)

Rows: Inspection result

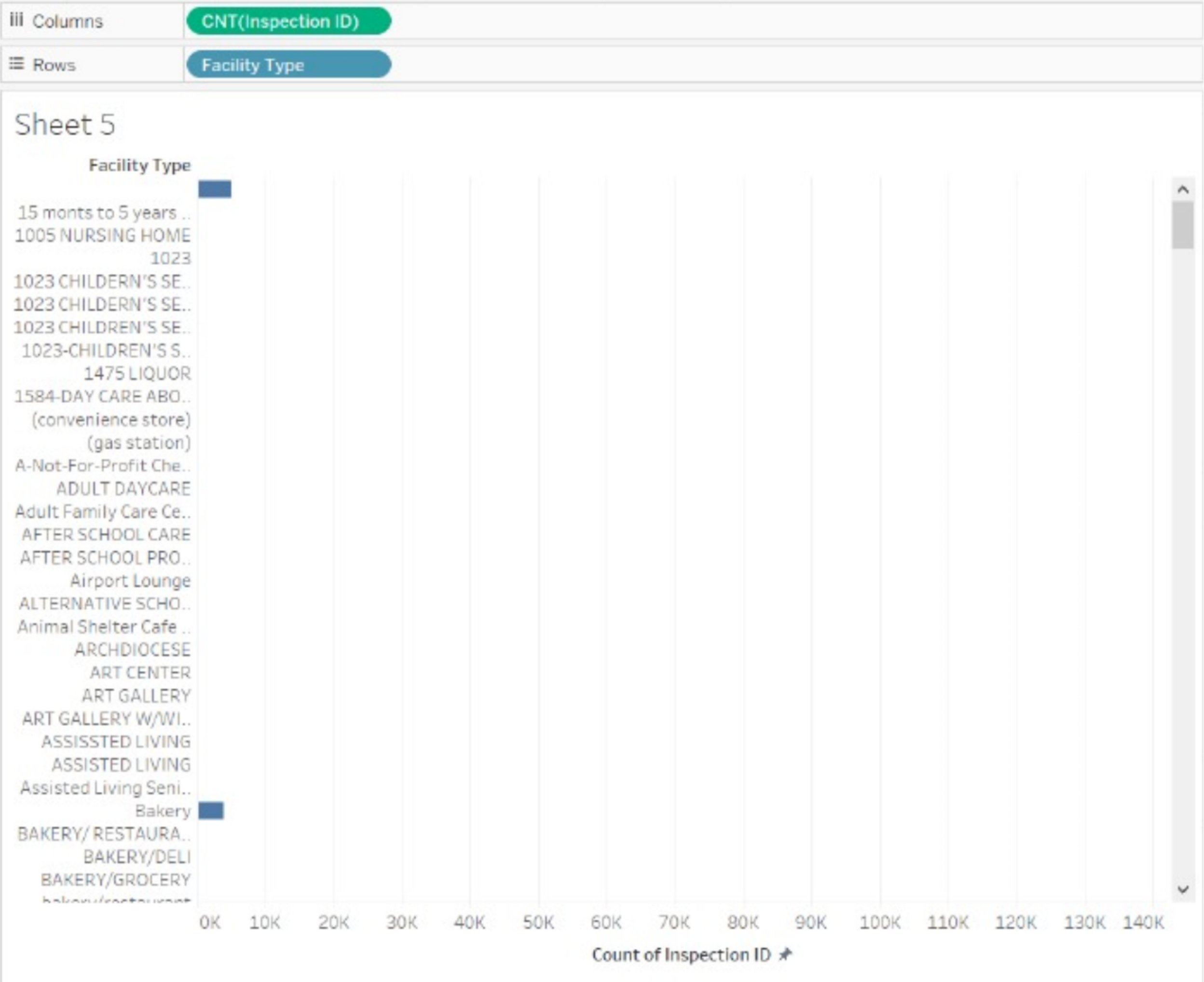


Columns: CNT(Risk)

Rows: City



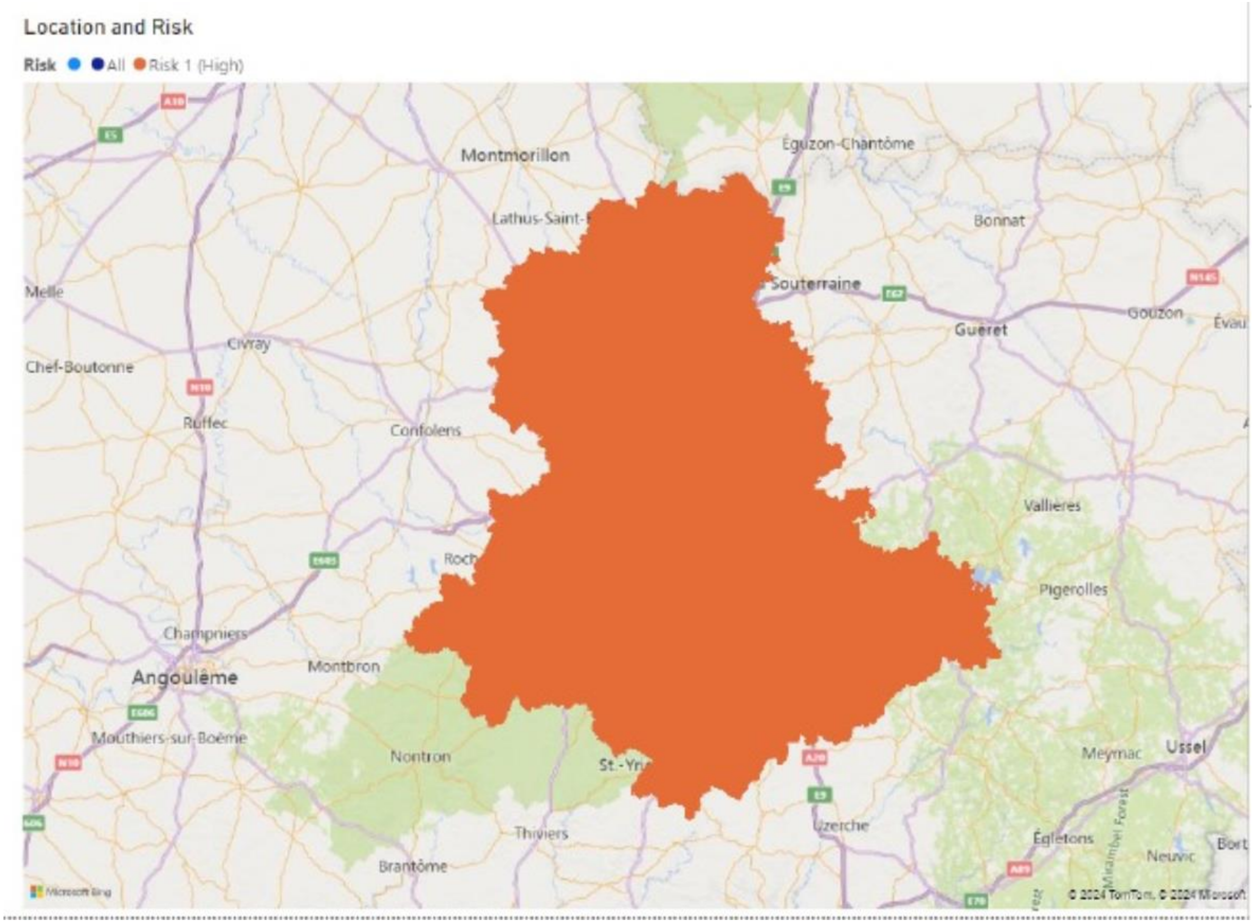
Facility Type:
Count of Inspection ID: **5,119**



Columns CNT(Inspection ID)

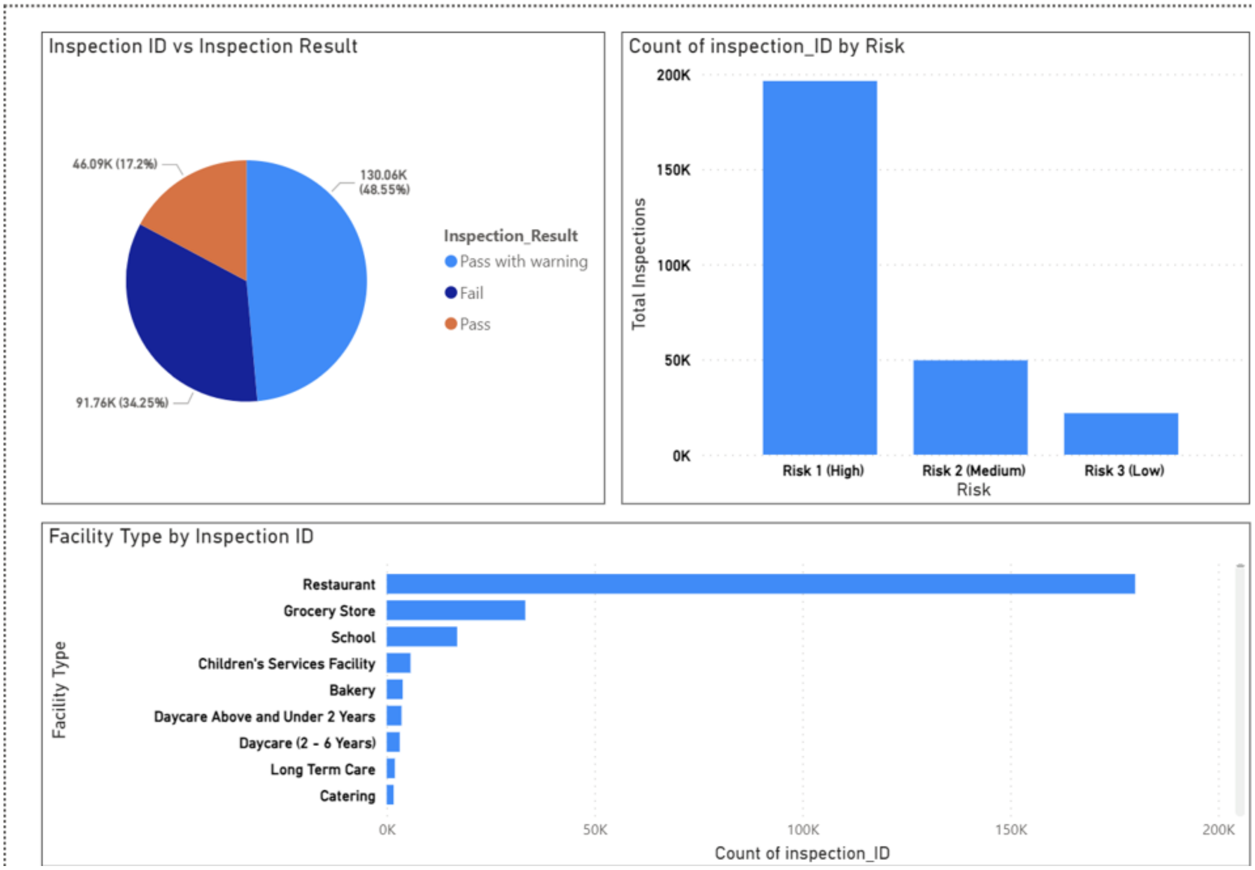
Rows Facility Type

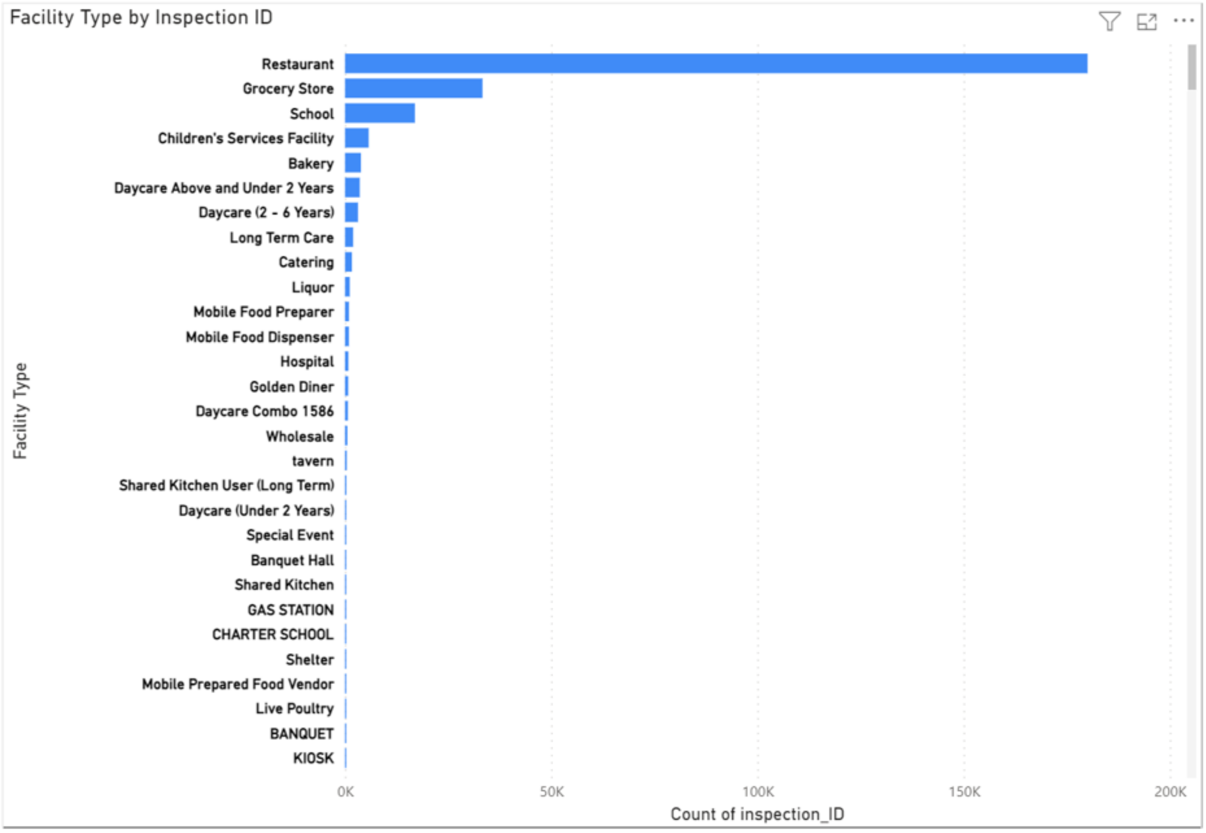
PowerBI Screenshots

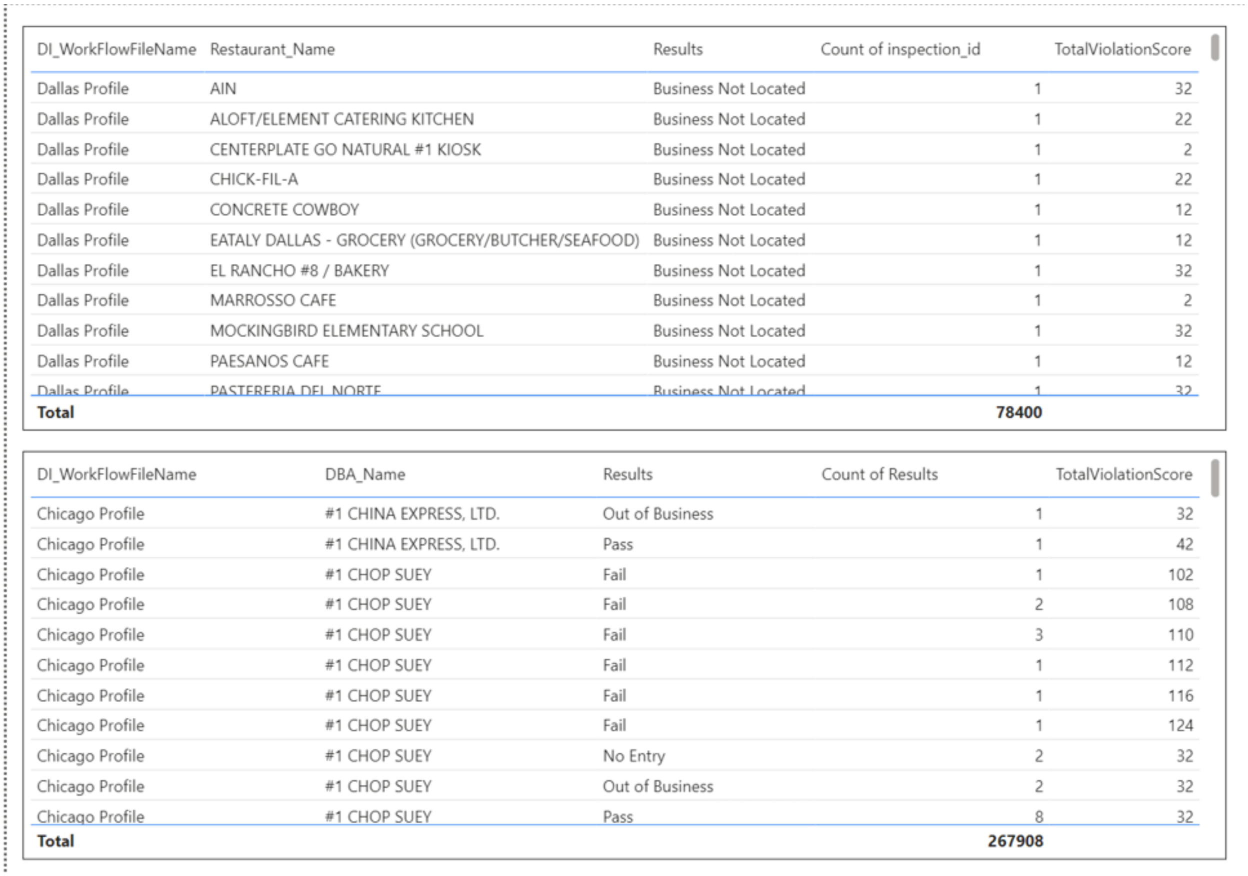


Location and Risk

Risk ● ●All ●Risk 1 (High)


Inspection ID vs Inspection Result
46.09K (17.2%)
130.06K (48.55%)
91.76K (34.25%)
Inspection_Result
● Pass with warning
● Fail
● Pass
Count of inspection_ID by Risk
200K
150K
100K
50K
0K
Total Inspections
Risk 1 (High)
Risk 2 (Medium)
Risk 3 (Low)
Risk
Facility Type by Inspection ID
Facility Type
Restaurant
Grocery Store
School
Children's Services Facility
Bakery
Daycare Above and Under 2 Years
Daycare (2 - 6 Years)
Long Term Care
Catering
0K
50K
100K
150K
200K
Count of inspection_ID


Facility Type by Inspection ID
Restaurant
Grocery Store
School
Children's Services Facility
Bakery
Daycare Above and Under 2 Years
Daycare (2 - 6 Years)
Long Term Care
Catering
Liquor
Mobile Food Preparer
Mobile Food Dispenser
Hospital
Golden Diner
Daycare Combo 1586
Wholesale
tavern
Shared Kitchen User (Long Term)
Daycare (Under 2 Years)
Special Event
Banquet Hall
Shared Kitchen
GAS STATION
CHARTER SCHOOL
Shelter
Mobile Prepared Food Vendor
Live Poultry
BANQUET
KIOSK
Facility Type
0K
50K
100K
150K
200K
Count of inspection_ID



DI_WorkFlowFileName	Restaurant_Name	Results	Count of inspection_id	TotalViolationScore
Dallas Profile	AIN	Business Not Located	1	32
Dallas Profile	ALOFT/ELEMENT CATERING KITCHEN	Business Not Located	1	22
Dallas Profile	CENTERPLATE GO NATURAL #1 KIOSK	Business Not Located	1	2
Dallas Profile	CHICK-FIL-A	Business Not Located	1	22
Dallas Profile	CONCRETE COWBOY	Business Not Located	1	12
Dallas Profile	EATALY DALLAS - GROCERY (GROCERY/BUTCHER/SEAFOOD)	Business Not Located	1	12
Dallas Profile	EL RANCHO #8 / BAKERY	Business Not Located	1	32
Dallas Profile	MARROSSO CAFE	Business Not Located	1	2
Dallas Profile	MOCKINGBIRD ELEMENTARY SCHOOL	Business Not Located	1	32
Dallas Profile	PAESANOS CAFE	Business Not Located	1	12
Dallas Profile	PASTERERIA DEL NORTE	Business Not Located	1	32
Total			**78400**	

DI_WorkFlowFileName	DBA_Name	Results	Count of Results	TotalViolationScore
Chicago Profile	#1 CHINA EXPRESS, LTD.	Out of Business	1	32
Chicago Profile	#1 CHINA EXPRESS, LTD.	Pass	1	42
Chicago Profile	#1 CHOP SUEY	Fail	1	102
Chicago Profile	#1 CHOP SUEY	Fail	2	108
Chicago Profile	#1 CHOP SUEY	Fail	3	110
Chicago Profile	#1 CHOP SUEY	Fail	1	112
Chicago Profile	#1 CHOP SUEY	Fail	1	116
Chicago Profile	#1 CHOP SUEY	Fail	1	124
Chicago Profile	#1 CHOP SUEY	No Entry	2	32
Chicago Profile	#1 CHOP SUEY	Out of Business	2	32
Chicago Profile	#1 CHOP SUEY	Pass	8	32
Total			**267908**	