GROUP 01

Project Surveillance

CIS 4400

Section CMWA

Radhika Kalani (radhika.kalani@baruchmail.cuny.edu)
Sophia Tsilerides (sophia.tsilerides@baruchmail.cuny.edu)

Table of Contents

Narrative Description

Project Surveillance is an independent project that works with two datasets. The first is the NYPD Complaint Data. This dataset includes all valid felony, misdemeanor, and violation crimes reported to the NYPD. The second dataset is the Open Parking and Camera Violations Data. This dataset contains open parking and camera violations issued by the City of New York.

We wanted to build a data warehouse to create an integrated view of the data from these two databases. The purpose of this data warehouse would be to understand if the crime in NYC boroughs is correlated to or a reflection of the city's surveillance cameras.

Information Needs

A "complaint" refers to a reported incident to the police department. Each complaint was reported on a single day to a precinct in a particular borough of NYC. Each complaint describes a particular offense such as harassment, larsency, dangerous drugs, assault, robbery, etc.

A "summons" is the result of an open camera violation on a citizen. Each summons was given to one and only one vehicle for a particular violation on a certain date by a precinct in an NYC borough. Violations include, no standing, expired meter, front or back plate missing, fire hydrant, double parking, fire lane, etc.

Because both data sets provide day-to-day data in NYC, we will perform our analysis using these attributes in our analysis.

Datasets

We downloaded the Crime Dataset and the Open Parking and Camera Violations Dataset from the NYC Open Data website and created seven CSV files, for every table in our schema except the Date Dimension and the two fact tables which we built ourselves. The CSVs were used to load the data into Pentaho in the input step. We are using data from the FY 2016.

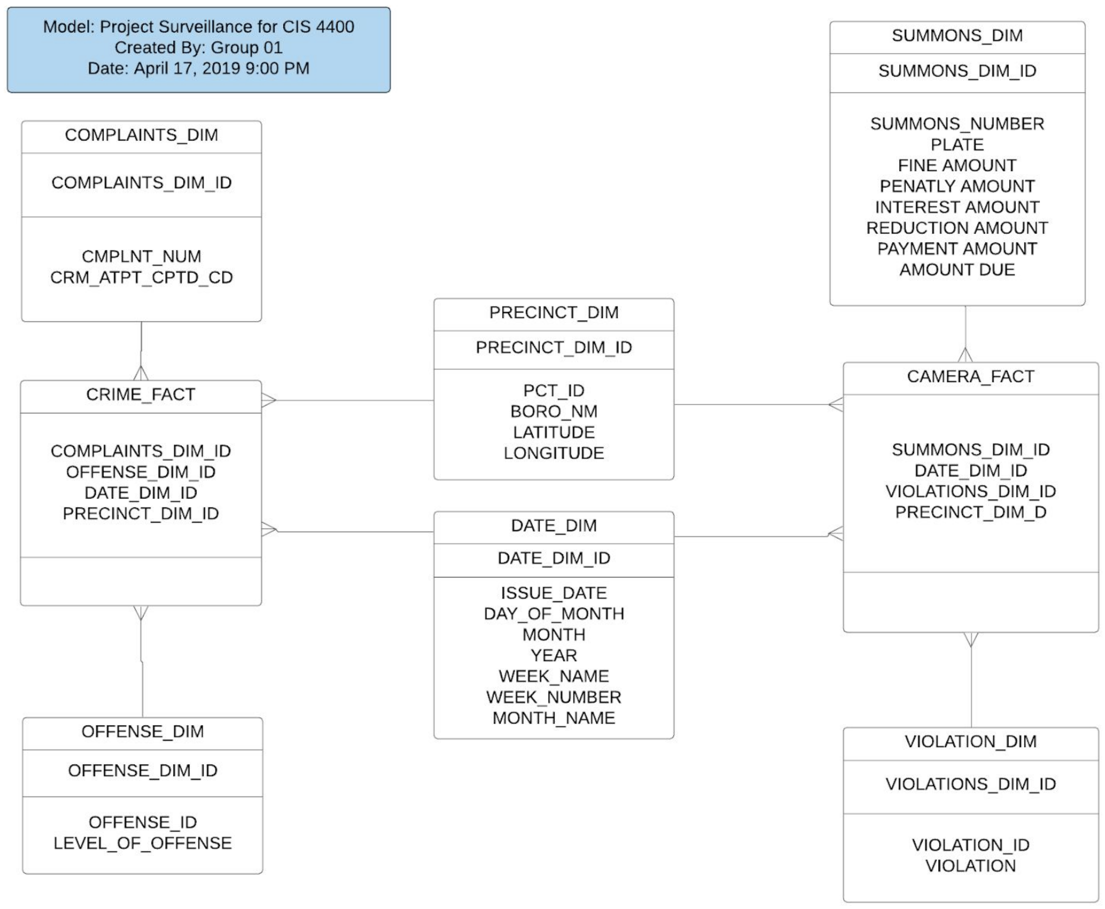
Crime Dataset source:

https://data.cityofnewyork.us/Public-Safety/NYPD-Complaint-Data-Current-Year-To-Date-/5uac-w243/data

Open Parking and Camera Violations Dataset source:
https://data.cityofnewyork.us/City-Government/Open-Parking-and-Camera-Violations/nc67-uf89/data

A Revised E-R Diagram

This is a revised E-R Diagram, different from our original in that it has the tables, keys, and attributes names that match the names used during the ETL process. We originally had embellishments in our fact tables, however because the grain for the fact tables is transactional we decided the embellishments were better off being displayed in the dashboard after querying and drilling up in our data warehouse.



Dimensions Description

Complaints Dimension: Includes a native key for each complaint that was reported to the precinct and an indicator of whether the crime was successfully completed or attempted, but failed or was interrupted prematurely.

Offenses Dimension: Contains a three digit offense classification code, level of offense, and description of the offense. The offense dimension has a relationship with the complaint dimension in that every complaint is associated with one and only one type of offense.
Date Dimension: Includes day, month, and year. Each complaint and each summons occurs on one and only one day. This dimension was embellished to include the day of the month, the month, the year, the week name, the week number and the month name.

Summons Dimension: A summons is the result of a camera violation to a vehicle. The dimension has a unique ID, license plate number, and license type (commercial, taxi, NYC, etc.). The dimension also includes the penalty amount, interest amount, reduction amount, payment amount and amount due.

Violations Dimension: Includes the violation description and fine amount associated with the violation. The dimension needed to be embellished with a violation ID for each violation. Each summons is associated with one and only one violation.

Precinct Dimension: Each complaint and violation was given by one and only one precinct. Each precinct belongs to a borough in NYC where the complaint or violation occurred. Each borough has a latitude and longitude location that correlates to the center of the borough's location, not individual precincts or violation locations.

Fact Table Descriptions

Crime Fact Table: Contains the Complaint ID, Date ID, Offense ID, Precinct ID, Borough ID. Number of Complaints is the fact of the fact table. The grain is periodic by day.

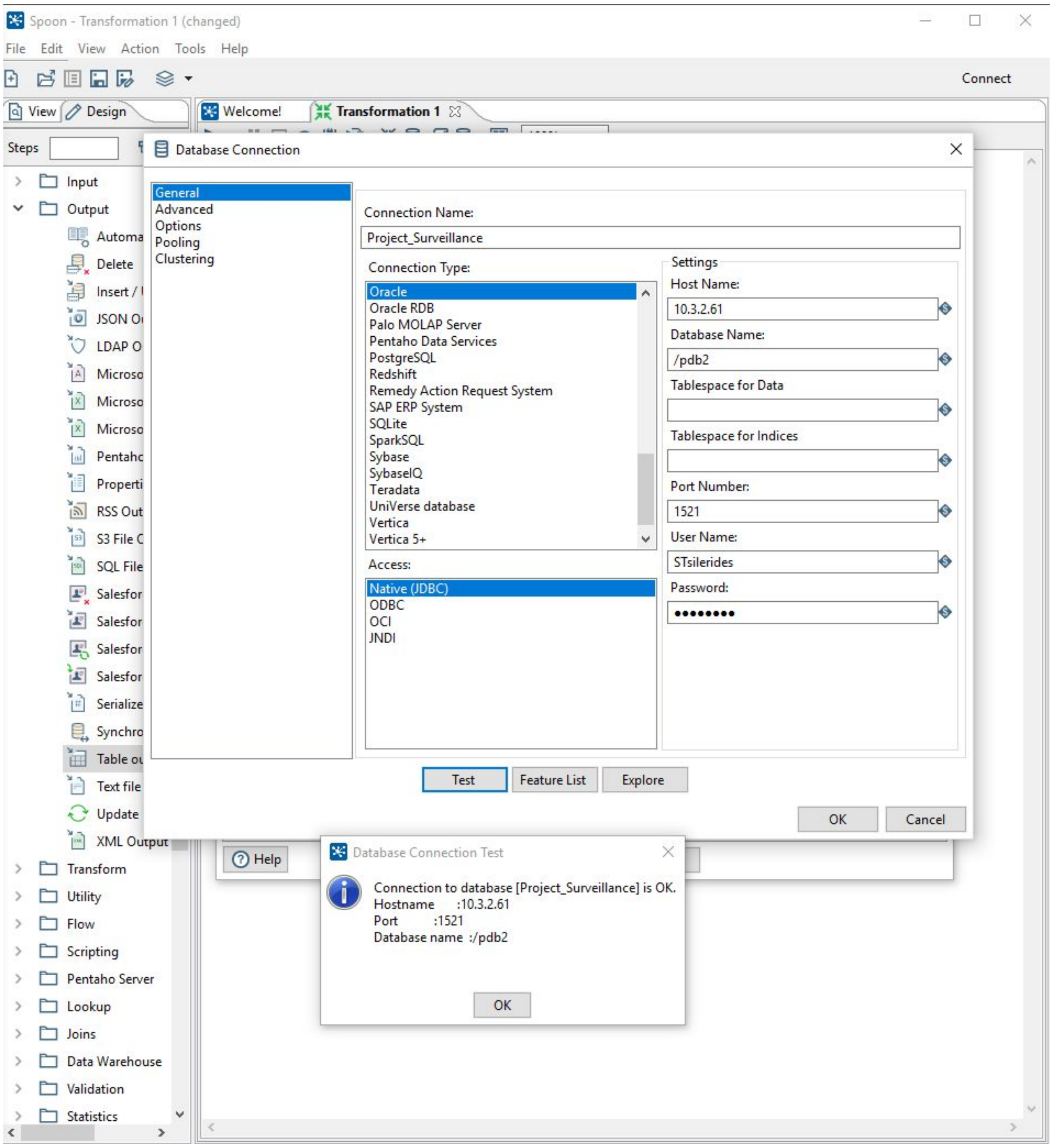
CameraViolations Fact Table: Contains the SummonsID, DateID, ViolationID, PrecinctID, BoroughID. Number of Violations and Outstanding Payments are the facts of the fact table. The grain is periodic by day.

KPIs

One of our KPI's is the ratio of the number of complaints and number of violations within the NYC boroughs. Another KPI was a time series analysis of complaints and violations to see how they fluctuate throughout the year and what months have the highest amounts and why. Our last KPI was the number of complaints and violations by precinct.

Target Schema

The target database we used for our data warehouse is Oracle. The schema is STsilerides. We used Oracle because we were comfortable with using SQL Developer from the first homework assignment and setting up an Oracle database connection with the account information provided to us. We used the table output step to establish the connection between Oracle and Pentaho.



Connection Name: Project_Surveillance

Connection Type: Oracle

Access: Native (JDBC)

Settings

Host Name: 10.3.2.61

Database Name: /pdb2

Tablespace for Data:

Tablespace for Indices:

Port Number: 1521

User Name: STsilerides

Password: ••••••••

```
Connection to database [Project_Surveillance] is OK.
Hostname      :10.3.2.61
Port          :1521
Database name :/pdb2
```

ETL

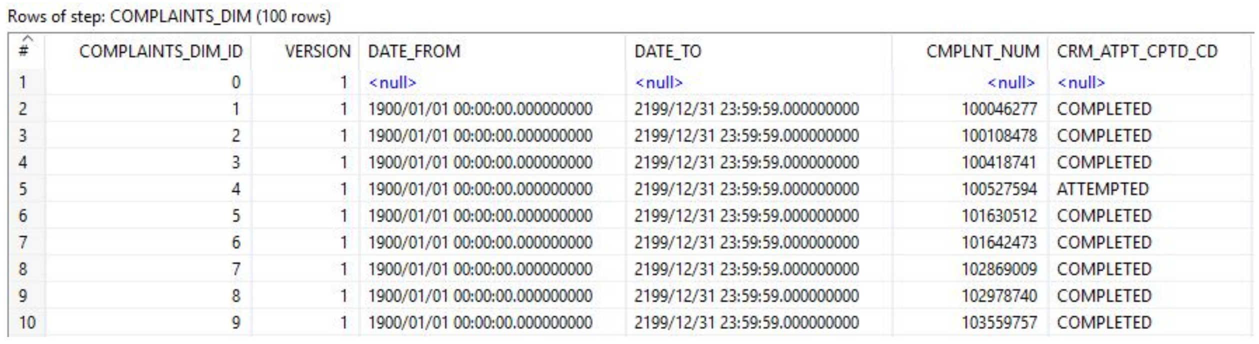
We used Pentaho's graphical interface Spoon to create the transformations and jobs for our ETL stage of the project. Next, we will go through all the transformations.

Complaint Dimension

The complaint dimension has a native key CMPLNT_NUM and a surrogate key COMPLAINTS_DIM_ID. We can see from the database screenshot that the surrogate key is much smaller than our business key. The SCD type for the complaint dimension is SCD Type 2. This is because when there is a change to a record for a complaint, the business people may want to reference why the change occurred and what it was updated to. Having a SCD Type 2 will allow the data warehouse to keep a historical record of these changes.

Naming conventions in the Complaint Dimension:

CSV file: `Complaint_Table`
CSV file input step name: `Complaints CSV File Import`
Dimensional lookup step name: `Complaints Dimension Update/Lookup`
target table name: `complaints_dim`
Dimension field: `CMPLNT_NUM`
field in stream: `CMPLNT_NUM`
technical key: `complaints_dim_id`
SCD type: `insert`



Rows of step: COMPLAINTS_DIM (100 rows)

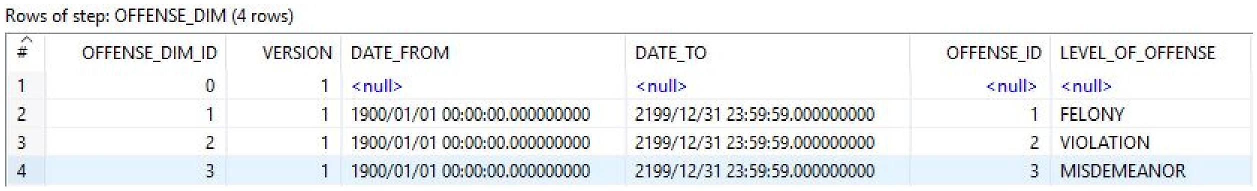
#	COMPLAINTS_DIM_ID	VERSION	DATE_FROM	DATE_TO	CMPLNT_NUM	CRM_ATPT_CPTD_CD
1	0	1	<null>	<null>	<null>	<null>
2	1	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	100046277	COMPLETED
3	2	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	100108478	COMPLETED
4	3	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	100418741	COMPLETED
5	4	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	100527594	ATTEMPTED
6	5	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	101630512	COMPLETED
7	6	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	101642473	COMPLETED
8	7	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	102869009	COMPLETED
9	8	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	102978740	COMPLETED
10	9	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	103559757	COMPLETED

Offense Dimension

The offense dimension has a SCD type of Type 1. This is because the Offense dimension holds the names of the types of offenses that a complaint can be labeled as. In our dataset, there are three types: felony, violation and misdemeanor. If for some reason the name of the offense is changed, than it can just be written over the original name and changed everywhere. This does not retain any history, but for this dimension, history is not needed for analysis.

Naming conventions in the Offense Dimension:

CSV file: `Offense_Table`
CSV file input step name: `Offense CSV File Import`
Dimensional lookup step name: `Offense Dimension Update/Lookup`
target table name: `offense_dim`
Dimension field: `Offense_ID`
field in stream: `Offense_ID`
technical key: `Offense_dim_id`
SCD type: `punch through`


Rows of step: OFFENSE_DIM (4 rows)

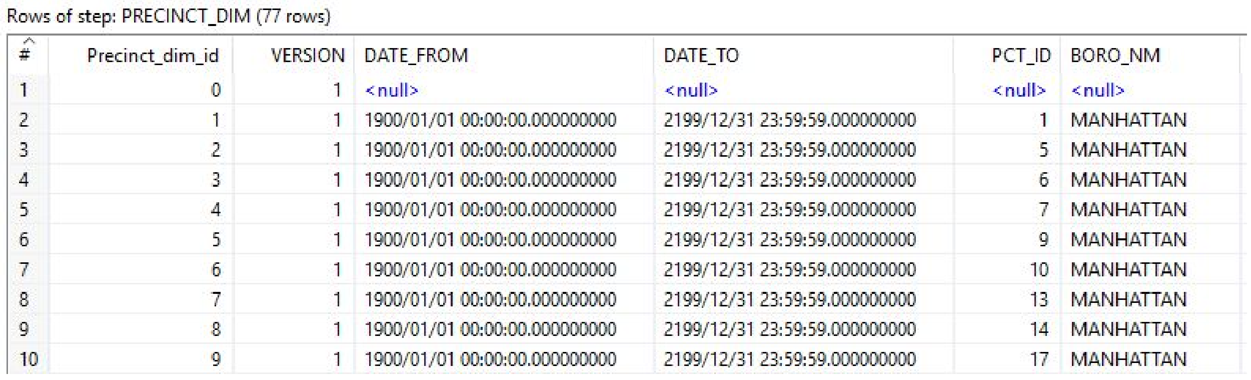
#	OFFENSE_DIM_ID	VERSION	DATE_FROM	DATE_TO	OFFENSE_ID	LEVEL_OF_OFFENSE
1	0	1	<null>	<null>	<null>	<null>
2	1	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	1	FELONY
3	2	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	2	VIOLATION
4	3	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	3	MISDEMEANOR

Precinct Dimension

The precinct dimension has a SCD type of Type 2. If for some reason the borough of a precinct changes, SCD Type 2 will allow the dimension to keep the historical borough and the new borough. This way, during analysis, the current and historical records are available.

Naming conventions in the Precinct Dimension:

CSV file: `Precinct_Table`
CSV file input step name: `Precinct CSV File Import`
Dimensional lookup step name: `Precinct Dimension Update/Lookup`
target table name: `offense_dim`
Dimension field: `PCT_ID`
field in stream: `PCT_ID`
technical key: `Precinct_dim_id`
SCD type: `insert`


Rows of step: PRECINCT_DIM (77 rows)

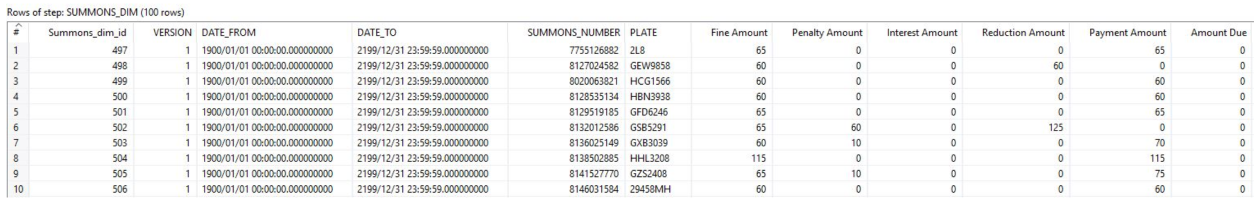
#	Precinct_dim_id	VERSION	DATE_FROM	DATE_TO	PCT_ID	BORO_NM
1	0	1	<null>	<null>	<null>	<null>
2	1	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	1	MANHATTAN
3	2	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	5	MANHATTAN
4	3	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	6	MANHATTAN
5	4	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	7	MANHATTAN
6	5	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	9	MANHATTAN
7	6	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	10	MANHATTAN
8	7	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	13	MANHATTAN
9	8	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	14	MANHATTAN
10	9	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	17	MANHATTAN

Summons Dimension

The source of our summons and violations dimensions had over 500 million rows. In order to keep our goals focused, and our processing time fast, we pruned the rows to only contain violations that are considered “dangerous” by our standards. This left us with a manageable number of records. The summons dimension has a SCD of Type 1. This is because we are only keeping track of the financials in this dimension. Therefore, if a summons has been paid, the amount can just be flipped to 0. The original dataset does not keep track of when the payment was PIFed, so there would be no way to analyze it regardless.

Naming conventions in the Summons Dimension:

CSV file: `Summons_Table`
CSV file input step name: `Summons CSV File Import`
Dimensional lookup step name: `Summons Dimension Update/Lookup`
target table name: `summons_dim`
Dimension field: `Summons_Number`
field in stream: `Summons_Number`
technical key: `Summons_dim_id`
SCD type: `punch through`


Rows of step: SUMMONS_DIM (100 rows)

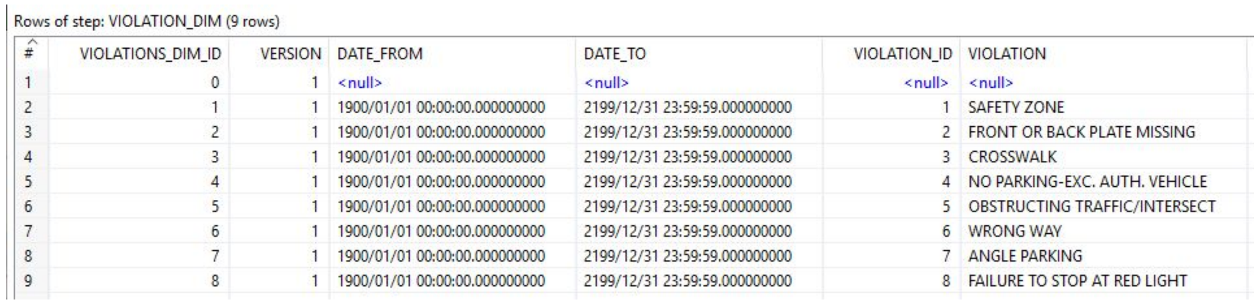
#	Summons_dim_id	VERSION	DATE_FROM	DATE_TO	SUMMONS_NUMBER	PLATE	Fine Amount	Penalty Amount	Interest Amount	Reduction Amount	Payment Amount	Amount Due
1	497	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	7755126882	2L8	65	0	0	0	65	0
2	498	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	8127024582	GEW9858	60	0	0	60	0	0
3	499	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	8020063821	HCG1566	60	0	0	0	60	0
4	500	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	8128535134	HBN3938	60	0	0	0	60	0
5	501	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	8129519185	GFD6246	65	0	0	0	65	0
6	502	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	8132012586	GSB5291	65	60	0	125	0	0
7	503	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	8136025149	GXB3039	60	10	0	0	70	0
8	504	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	8138502885	HHL3208	115	0	0	0	115	0
9	505	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	8141527770	GZS2408	65	10	0	0	75	0
10	506	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	8146031584	29458MH	60	0	0	0	60	0

Violations Dimension

The violations that were left after pruning are stored in this violations dimension. The violations dimension has a SCD of Type 1. If for some reason the name of the violation is changed, than it can just be written over the original name and changed everywhere. This does not retain any history, but for this dimension, history is not needed for analysis.

Naming conventions in the Violations Dimension:

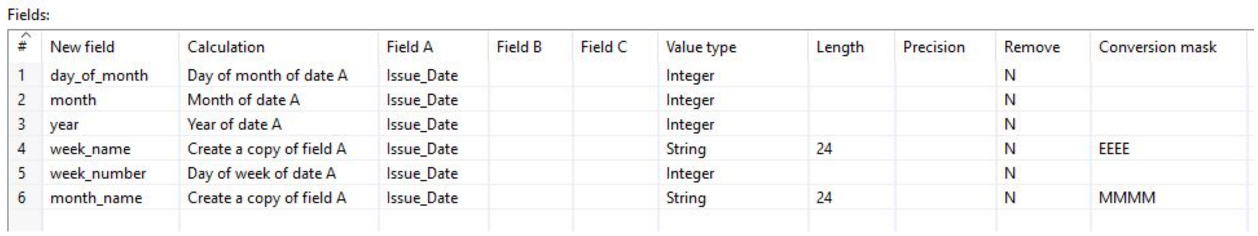
CSV file: `Violations_Table`
CSV file input step name: `Violations CSV File Import`
Dimensional lookup step name: `Violations Dimension Update/Lookup`
target table name: `Violation_dim`
Dimension field: `Violations_ID`
field in stream: `Violations_ID`
technical key: `Violations_dim_id`
SCD type: `punch through`


Rows of step: VIOLATION_DIM (9 rows)

#	VIOLATIONS_DIM_ID	VERSION	DATE_FROM	DATE_TO	VIOLATION_ID	VIOLATION
1	0	1	<null>	<null>	<null>	<null>
2	1	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	1	SAFETY ZONE
3	2	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	2	FRONT OR BACK PLATE MISSING
4	3	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	3	CROSSWALK
5	4	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	4	NO PARKING-EXC. AUTH. VEHICLE
6	5	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	5	OBSTRUCTING TRAFFIC/INTERSECT
7	6	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	6	WRONG WAY
8	7	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	7	ANGLE PARKING
9	8	1	1900/01/01 00:00:00.000000000	2199/12/31 23:59:59.000000000	8	FAILURE TO STOP AT RED LIGHT

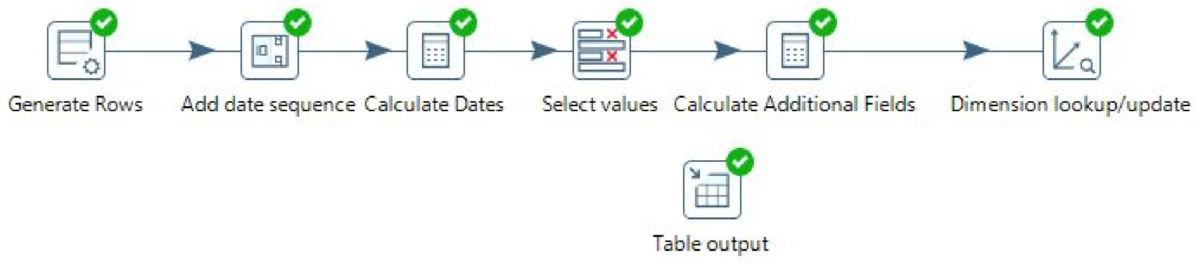
Date Dimension

The date dimension was built to store the dates and their IDs for FY 2016 as our data represents. Therefore, 800 rows were generated starting from 12/31/2015 and it goes into 2017 for a few days in January. Using the calculator step, some embellished columns were also included in the date dimension to make analysis after ETL easier. They are shown here:


Fields:

#	New field	Calculation	Field A	Field B	Field C	Value type	Length	Precision	Remove	Conversion mask
1	day_of_month	Day of month of date A	Issue_Date			Integer			N	
2	month	Month of date A	Issue_Date			Integer			N	
3	year	Year of date A	Issue_Date			Integer			N	
4	week_name	Create a copy of field A	Issue_Date			String	24		N	EEEE
5	week_number	Day of week of date A	Issue_Date			Integer			N	
6	month_name	Create a copy of field A	Issue_Date			String	24		N	MMMM

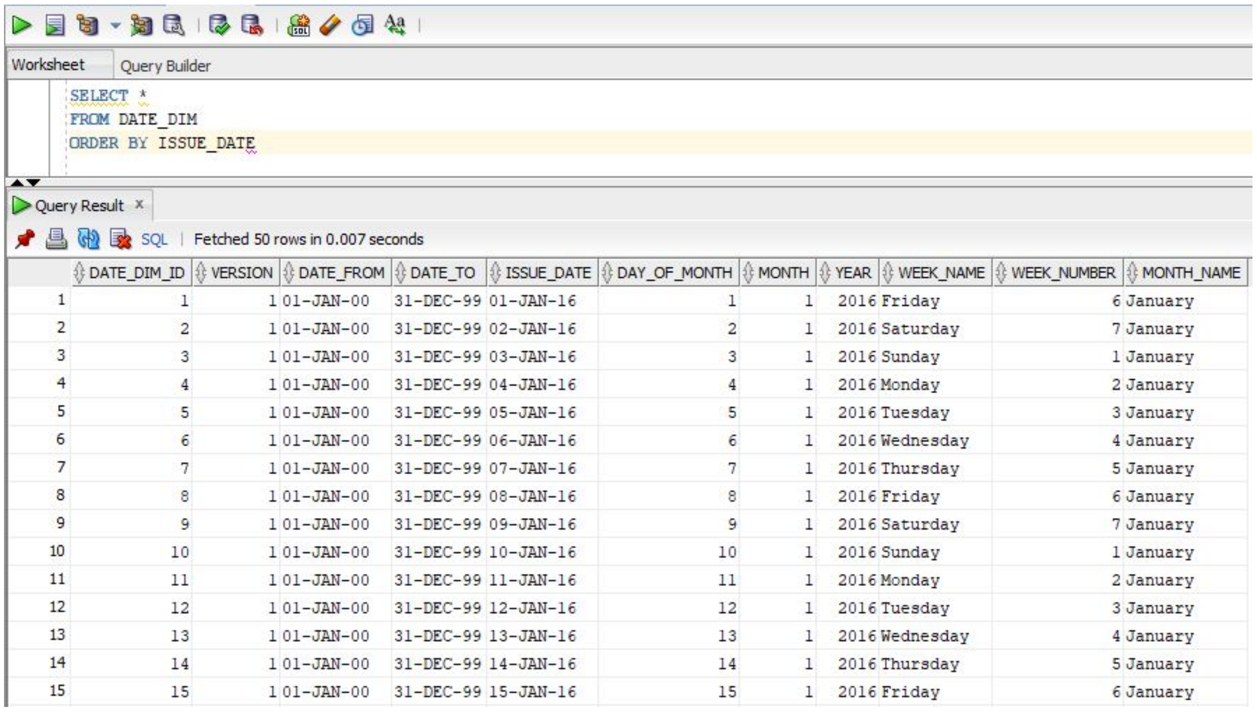
The transformation of our date dimension looked like the below:



Naming conventions in the Date Dimension:

New field name: `Issue_date`
Table name: `date_dim`
Dimension field: `Issue_Date`
field in stream: `Issue_Date`
technical key: `date_dim_id`

To check whether our transformation build out the date dimension correctly, we viewed the records in oracle and ordered the records by the Issue_date. This showed that every day for this year had a record in order and in the year's entirety.



```
SELECT *
FROM DATE_DIM
ORDER BY ISSUE_DATE
```

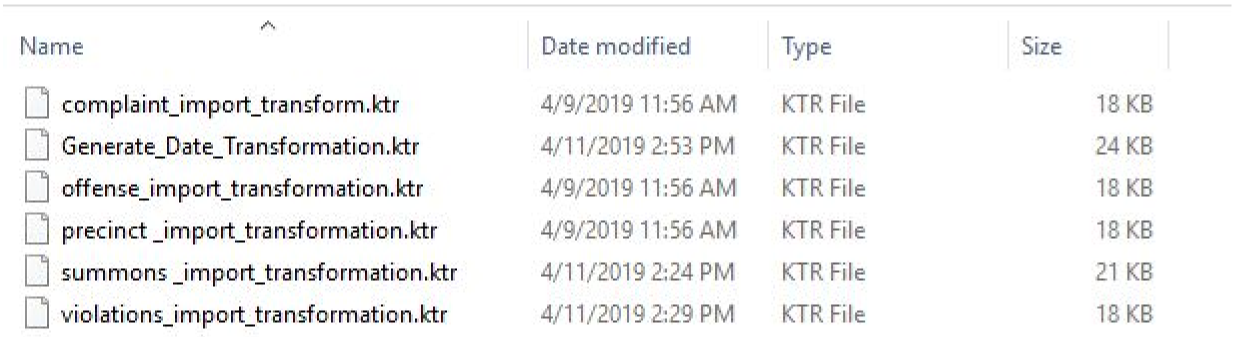
Query Result

Fetched 50 rows in 0.007 seconds

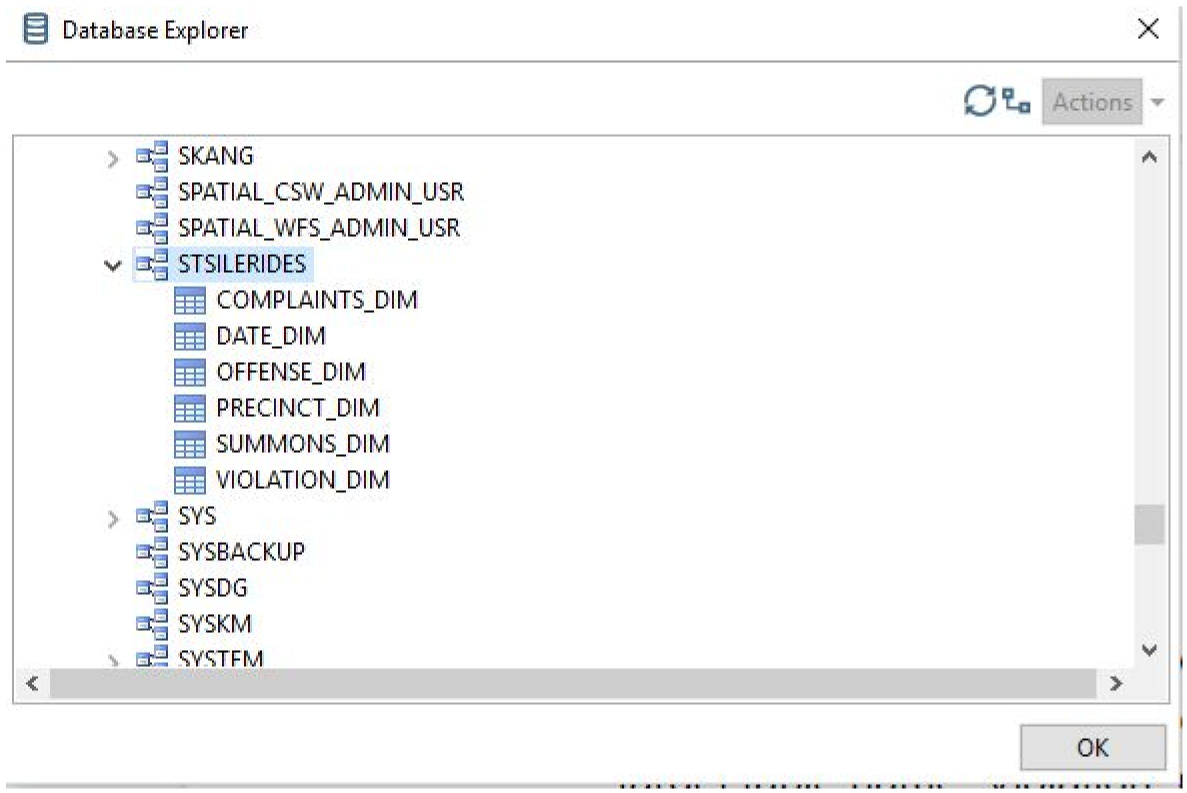
	DATE_DIM_ID	VERSION	DATE_FROM	DATE_TO	ISSUE_DATE	DAY_OF_MONTH	MONTH	YEAR	WEEK_NAME	WEEK_NUMBER	MONTH_NAME
1	1	1	01-JAN-00	31-DEC-99	01-JAN-16	1	1	2016	Friday	6	January
2	2	1	01-JAN-00	31-DEC-99	02-JAN-16	2	1	2016	Saturday	7	January
3	3	1	01-JAN-00	31-DEC-99	03-JAN-16	3	1	2016	Sunday	1	January
4	4	1	01-JAN-00	31-DEC-99	04-JAN-16	4	1	2016	Monday	2	January
5	5	1	01-JAN-00	31-DEC-99	05-JAN-16	5	1	2016	Tuesday	3	January
6	6	1	01-JAN-00	31-DEC-99	06-JAN-16	6	1	2016	Wednesday	4	January
7	7	1	01-JAN-00	31-DEC-99	07-JAN-16	7	1	2016	Thursday	5	January
8	8	1	01-JAN-00	31-DEC-99	08-JAN-16	8	1	2016	Friday	6	January
9	9	1	01-JAN-00	31-DEC-99	09-JAN-16	9	1	2016	Saturday	7	January
10	10	1	01-JAN-00	31-DEC-99	10-JAN-16	10	1	2016	Sunday	1	January
11	11	1	01-JAN-00	31-DEC-99	11-JAN-16	11	1	2016	Monday	2	January
12	12	1	01-JAN-00	31-DEC-99	12-JAN-16	12	1	2016	Tuesday	3	January
13	13	1	01-JAN-00	31-DEC-99	13-JAN-16	13	1	2016	Wednesday	4	January
14	14	1	01-JAN-00	31-DEC-99	14-JAN-16	14	1	2016	Thursday	5	January
15	15	1	01-JAN-00	31-DEC-99	15-JAN-16	15	1	2016	Friday	6	January

ETL Milestone

At this point, all the dimensions surrounding our two fact tables have been transformed and loaded into the target. The transformations were also been saved to a USB. visuals of both these milestones are shown below.



Name	Date modified	Type	Size
complaint_import_transform.ktr	4/9/2019 11:56 AM	KTR File	18 KB
Generate_Date_Transformation.ktr	4/11/2019 2:53 PM	KTR File	24 KB
offense_import_transformation.ktr	4/9/2019 11:56 AM	KTR File	18 KB
precinct _import_transformation.ktr	4/9/2019 11:56 AM	KTR File	18 KB
summons _import_transformation.ktr	4/11/2019 2:24 PM	KTR File	21 KB
violations_import_transformation.ktr	4/11/2019 2:29 PM	KTR File	18 KB



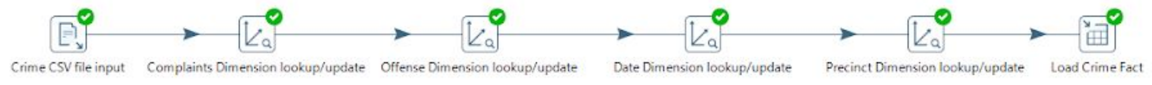
SKANG
SPATIAL_CSW_ADMIN_USR
SPATIAL_WFS_ADMIN_USR
STSILERIDES
COMPLAINTS_DIM
DATE_DIM
OFFENSE_DIM
PRECINCT_DIM
SUMMONS_DIM
VIOLATION_DIM
SYS
SYSBACKUP
SYSDG
SYSKM
SYSTEM

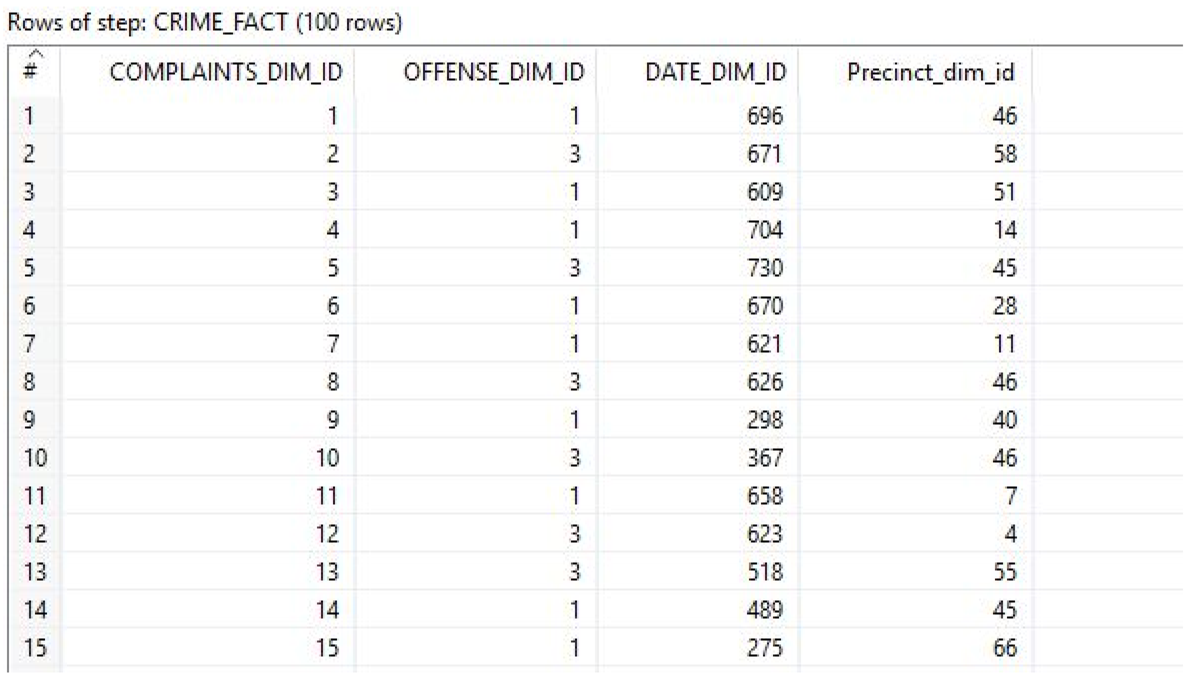
Fact Tables

Our fact tables are all-key with no measurements associated with each row. Therefore, they help us to keep track of the events that have taken place: a summons or a complaint. Eventually, we used our fact tables to see if there is any correlation between the crime in NYC boroughs and the city's surveillance cameras.

Crime Fact Table

Our first fact table is for our crime data. It has the dimension IDs for the complaints dimension, the offense dimension, the date dimension, and the precinct dimension. It has a transactional grain, so each record represents a complaint made to a precinct in NYC in 2016.


Crime CSV file input
Complaints Dimension lookup/update
Offense Dimension lookup/update
Date Dimension lookup/update
Precinct Dimension lookup/update
Load Crime Fact

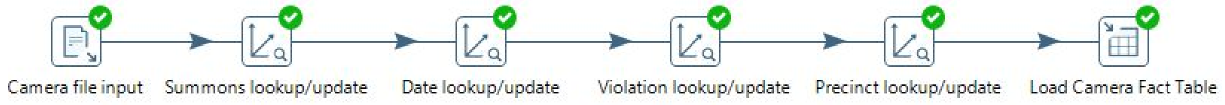


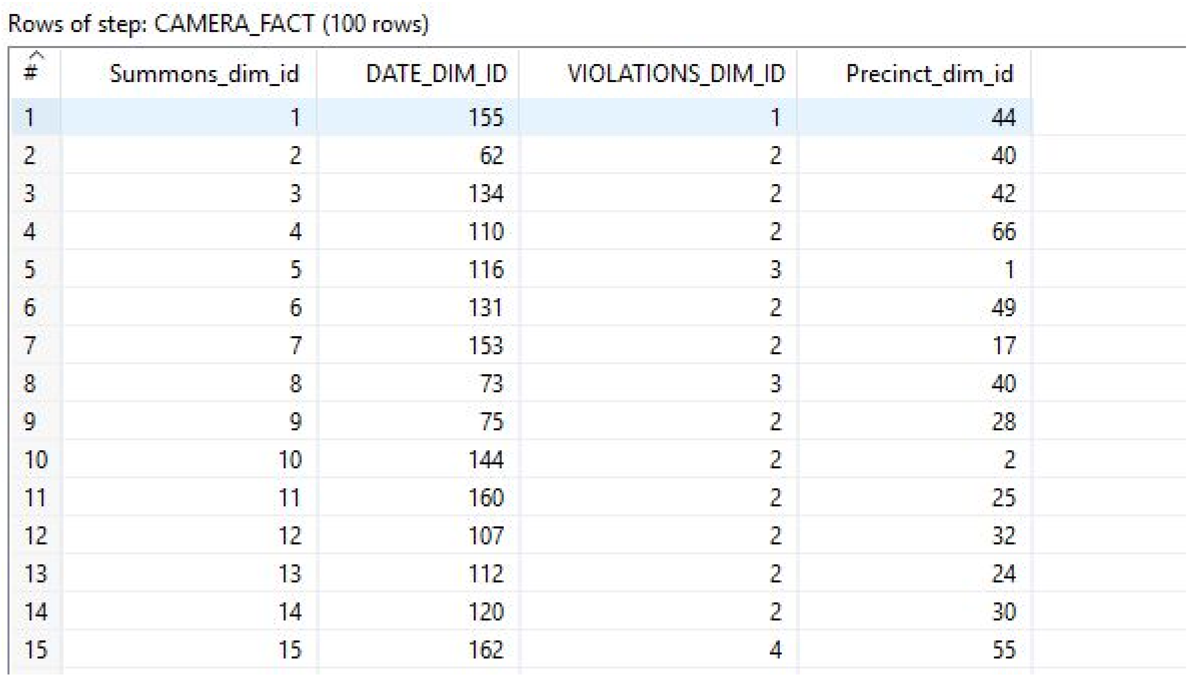
Rows of step: CRIME_FACT (100 rows)

#	COMPLAINTS_DIM_ID	OFFENSE_DIM_ID	DATE_DIM_ID	Precinct_dim_id
1	1	1	696	46
2	2	3	671	58
3	3	1	609	51
4	4	1	704	14
5	5	3	730	45
6	6	1	670	28
7	7	1	621	11
8	8	3	626	46
9	9	1	298	40
10	10	3	367	46
11	11	1	658	7
12	12	3	623	4
13	13	3	518	55
14	14	1	489	45
15	15	1	275	66

Camera Fact Table

Our second fact table is for our camera data. It has the dimension IDs for the summons dimension, the date dimension, the violations dimension, and the precinct dimension. It has a transactional grain, so each record represents a summons given by precinct in NYC in 2016 for certain violations we consider "dangerous".

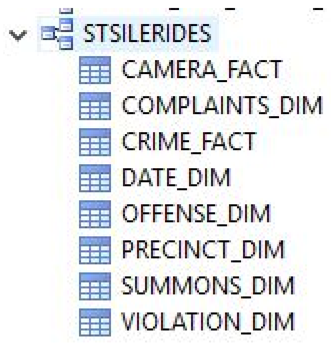




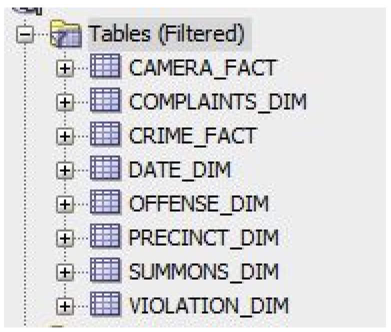
Rows of step: CAMERA_FACT (100 rows)

#	Summons_dim_id	DATE_DIM_ID	VIOLATIONS_DIM_ID	Precinct_dim_id
1	1	155	1	44
2	2	62	2	40
3	3	134	2	42
4	4	110	2	66
5	5	116	3	1
6	6	131	2	49
7	7	153	2	17
8	8	73	3	40
9	9	75	2	28
10	10	144	2	2
11	11	160	2	25
12	12	107	2	32
13	13	112	2	24
14	14	120	2	30
15	15	162	4	55

Fact Tables Milestone
All the tables for our data warehouse are now saved in our target database.



STSILERIDES
CAMERA_FACT
COMPLAINTS_DIM
CRIME_FACT
DATE_DIM
OFFENSE_DIM
PRECINCT_DIM
SUMMONS_DIM
VIOLATION_DIM



Tables (Filtered)
CAMERA_FACT
COMPLAINTS_DIM
CRIME_FACT
DATE_DIM
OFFENSE_DIM
PRECINCT_DIM
SUMMONS_DIM
VIOLATION_DIM

Foreign Key SQL Constraints

The following SQL was used with SQL Developer to add foreign key constraints to tie our dimension tables and fact tables in our target DBMS after the ETL process..

```
ALTER TABLE CRIME_FACT
     ADD CONSTRAINT FK_ComplaintsID
```

```
          FOREIGN KEY (COMPLAINTS_DIM_ID)
               REFERENCES COMPLAINTS_DIM(COMPAINTS_DIM_ID)

ALTER TABLE CRIME_FACT
     ADD CONSTRAINT FK_OffenseID
          FOREIGN KEY (OFFENSE_DIM_ID)
               REFERENCES OFFENSE_DIM(OFFENSE_DIM_ID)

ALTER TABLE CRIME_FACT
     ADD CONSTRAINT FK_DateID
          FOREIGN KEY (DATE_DIM_ID)
               REFERENCES DATE_DIM(DATE_DIM_ID)

ALTER TABLE CRIME_FACT
     ADD CONSTRAINT FK_PrecinctID
          FOREIGN KEY ("Precinct_dim_id ")
               REFERENCES PRECINCT_DIM("Precinct_dim_id ")

ALTER TABLE CAMERA_FACT
     ADD CONSTRAINT FK_Summons_ID
          FOREIGN KEY ("Summons_dim_id ")
               REFERENCES SUMMONS_DIM("Summons_dim_id ")

ALTER TABLE CAMERA_FACT
     ADD CONSTRAINT FK_Date_ID
          FOREIGN KEY (DATE_DIM_ID)
               REFERENCES DATE_DIM(DATE_DIM_ID)

ALTER TABLE CAMERA_FACT
     ADD CONSTRAINT FK_Precinct_ID
          FOREIGN KEY ("Precinct_dim_id ")
               REFERENCES PRECINCT_DIM("Precinct_dim_id ")

ALTER TABLE CAMERA_FACT
     ADD CONSTRAINT FK_ViolationsID
          FOREIGN KEY (VIOLATIONS_DIM_ID)
               REFERENCES VIOLATION_DIM(VIOLATIONS_DIM_ID)
```

Physical Level Diagram

After ETL and tying our dimension tables to the fact tables with foreign key constraints, we used the Oracle SQL Developer Data Model tools to reverse engineer the database schema created through the ETL process into the below physical level diagram:

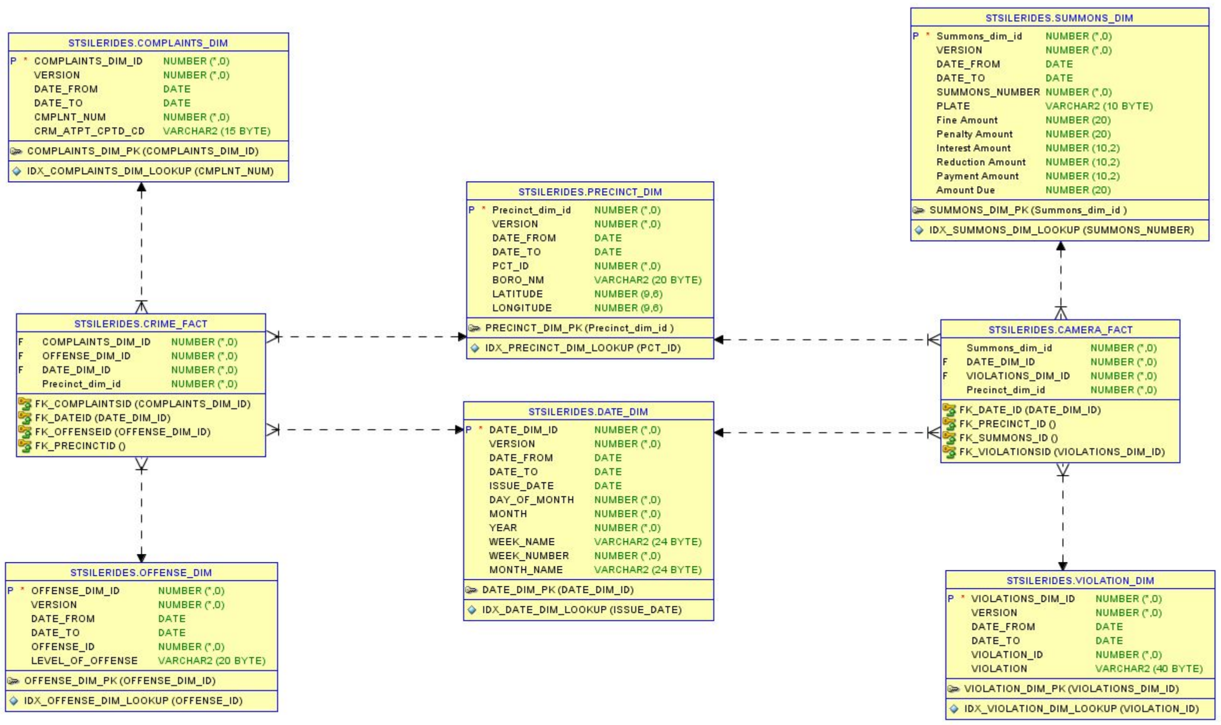
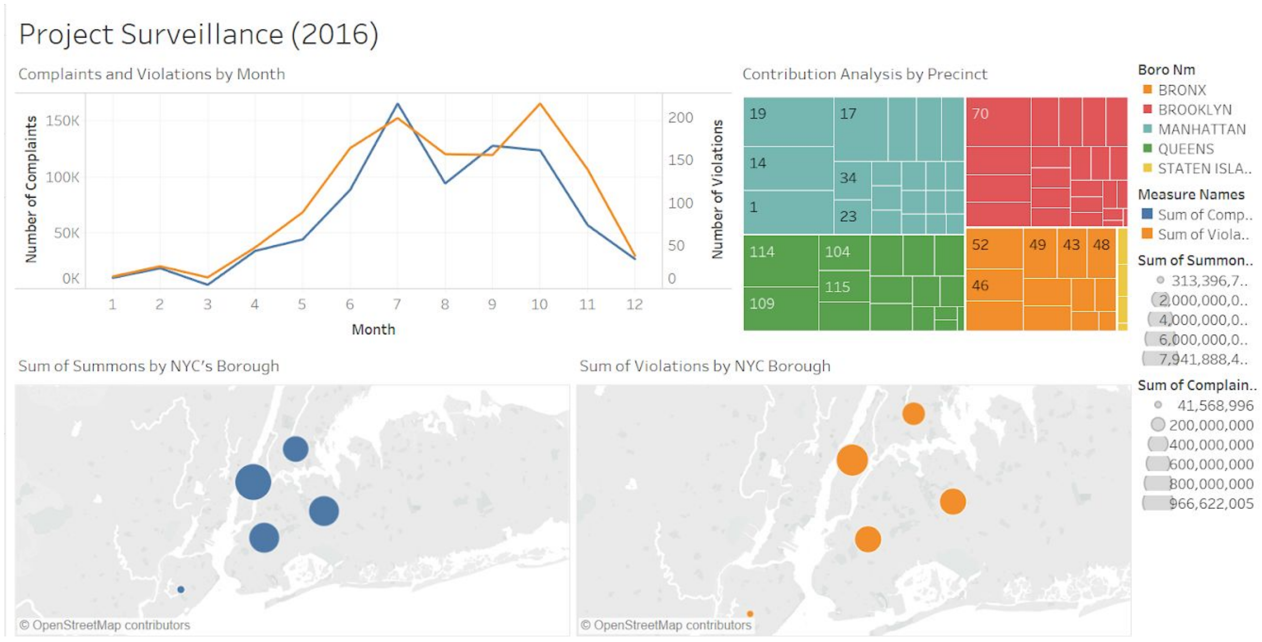

STSILERIDES.COMPLAINTS_DIM
P * COMPLAINTS_DIM_ID NUMBER (*,0)
VERSION NUMBER (*,0)
DATE_FROM DATE
DATE_TO DATE
CMPLNT_NUM NUMBER (*,0)
CRM_ATPT_CPTD_CD VARCHAR2 (15 BYTE)
COMPLAINTS_DIM_PK (COMPLAINTS_DIM_ID)
IDX_COMPLAINTS_DIM_LOOKUP (CMPLNT_NUM)
STSILERIDES.PRECINCT_DIM
P * Precinct_dim_id NUMBER (*,0)
VERSION NUMBER (*,0)
DATE_FROM DATE
DATE_TO DATE
PCT_ID NUMBER (*,0)
BORO_NM VARCHAR2 (20 BYTE)
LATITUDE NUMBER (9,6)
LONGITUDE NUMBER (9,6)
PRECINCT_DIM_PK (Precinct_dim_id)
IDX_PRECINCT_DIM_LOOKUP (PCT_ID)
STSILERIDES.SUMMONS_DIM
P * Summons_dim_id NUMBER (*,0)
VERSION NUMBER (*,0)
DATE_FROM DATE
DATE_TO DATE
SUMMONS_NUMBER NUMBER (*,0)
PLATE VARCHAR2 (10 BYTE)
Fine Amount NUMBER (20)
Penalty Amount NUMBER (20)
Interest Amount NUMBER (10,2)
Reduction Amount NUMBER (10,2)
Payment Amount NUMBER (10,2)
Amount Due NUMBER (20)
SUMMONS_DIM_PK (Summons_dim_id)
IDX_SUMMONS_DIM_LOOKUP (SUMMONS_NUMBER)
STSILERIDES.CRIME_FACT
F COMPLAINTS_DIM_ID NUMBER (*,0)
F OFFENSE_DIM_ID NUMBER (*,0)
F DATE_DIM_ID NUMBER (*,0)
Precinct_dim_id NUMBER (*,0)
FK_COMPLAINTSID (COMPLAINTS_DIM_ID)
FK_DATEID (DATE_DIM_ID)
FK_OFFENSEID (OFFENSE_DIM_ID)
FK_PRECINCTID ()
STSILERIDES.CAMERA_FACT
Summons_dim_id NUMBER (*,0)
F DATE_DIM_ID NUMBER (*,0)
F VIOLATIONS_DIM_ID NUMBER (*,0)
Precinct_dim_id NUMBER (*,0)
FK_DATE_ID (DATE_DIM_ID)
FK_PRECINCT_ID ()
FK_SUMMONS_ID ()
FK_VIOLATIONSID (VIOLATIONS_DIM_ID)
STSILERIDES.DATE_DIM
P * DATE_DIM_ID NUMBER (*,0)
VERSION NUMBER (*,0)
DATE_FROM DATE
DATE_TO DATE
ISSUE_DATE DATE
DAY_OF_MONTH NUMBER (*,0)
MONTH NUMBER (*,0)
YEAR NUMBER (*,0)
WEEK_NAME VARCHAR2 (24 BYTE)
WEEK_NUMBER NUMBER (*,0)
MONTH_NAME VARCHAR2 (24 BYTE)
DATE_DIM_PK (DATE_DIM_ID)
IDX_DATE_DIM_LOOKUP (ISSUE_DATE)
STSILERIDES.OFFENSE_DIM
P * OFFENSE_DIM_ID NUMBER (*,0)
VERSION NUMBER (*,0)
DATE_FROM DATE
DATE_TO DATE
OFFENSE_ID NUMBER (*,0)
LEVEL_OF_OFFENSE VARCHAR2 (20 BYTE)
OFFENSE_DIM_PK (OFFENSE_DIM_ID)
IDX_OFFENSE_DIM_LOOKUP (OFFENSE_ID)
STSILERIDES.VIOLATION_DIM
P * VIOLATIONS_DIM_ID NUMBER (*,0)
VERSION NUMBER (*,0)
DATE_FROM DATE
DATE_TO DATE
VIOLATION_ID NUMBER (*,0)
VIOLATION VARCHAR2 (40 BYTE)
VIOLATION_DIM_PK (VIOLATIONS_DIM_ID)
IDX_VIOLATION_DIM_LOOKUP (VIOLATION_ID)

Tableau Dashboard

For the dashboard portion of the project, we decided to use Tableau. Tableau is easy to use, simple, and has a intuitive user interface with common layouts and visual type standards automatically generated. Also, we were able to directly connect our Oracle Database to Tableau. This way, we can customize and filter our data to what we want to see and drag significant attributes. Using the key performance indicators from our planning stage of the project, we ensured that our dashboards would help us solve the question: is there a relationship between the number of violations per day and complaints received in the NYC boroughs in 2016.

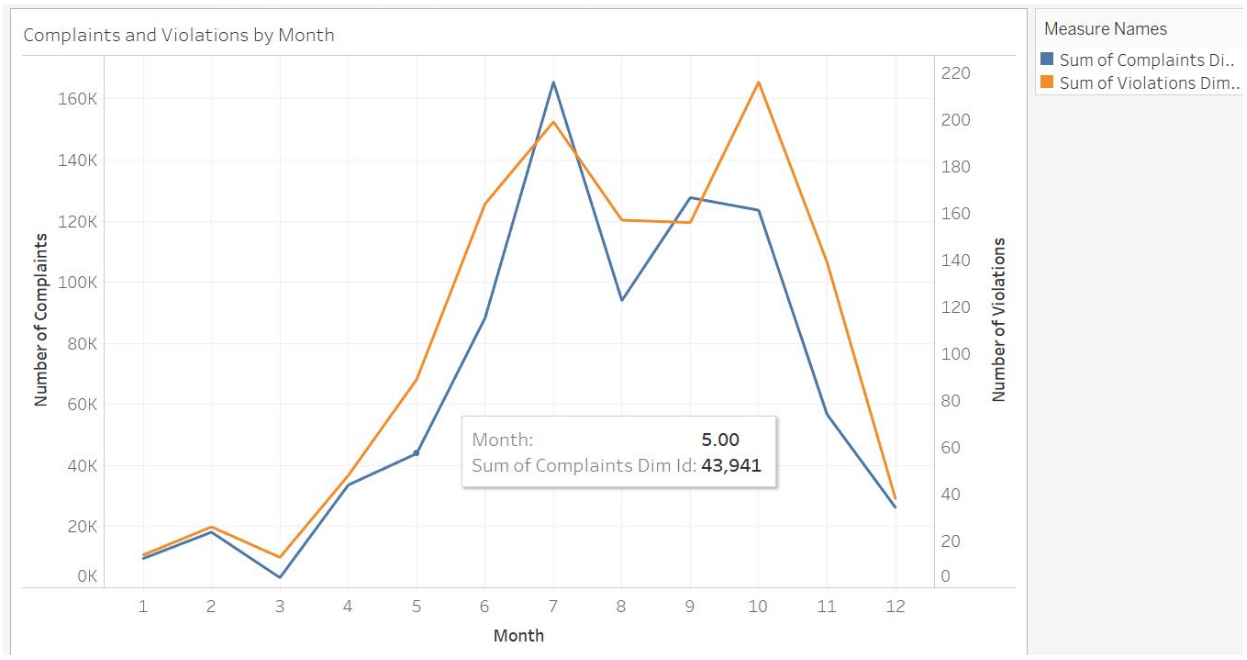
Finalized Dashboard

This is our final dashboard. We represented a subset of the data used in our data warehouse to explain our KPIs through different forms. Tableau allowed us to keep a consistent theme through the same color palette. The different colors are defined in the legend on the right-hand side for all the visuals. There is also a hover capability that allows you to get the exact data of each number, point, and circle. We arranged our dashboard this way to make it intuitive for end users to analyze. A simple line graph shows the correlation between complaints and violations easily. Next we have a tree map on the top right for contribution analysis. Lastly, we the two maps on the bottom showcase summons and violations by borough. We placed these next to each other for side-by-side comparison.



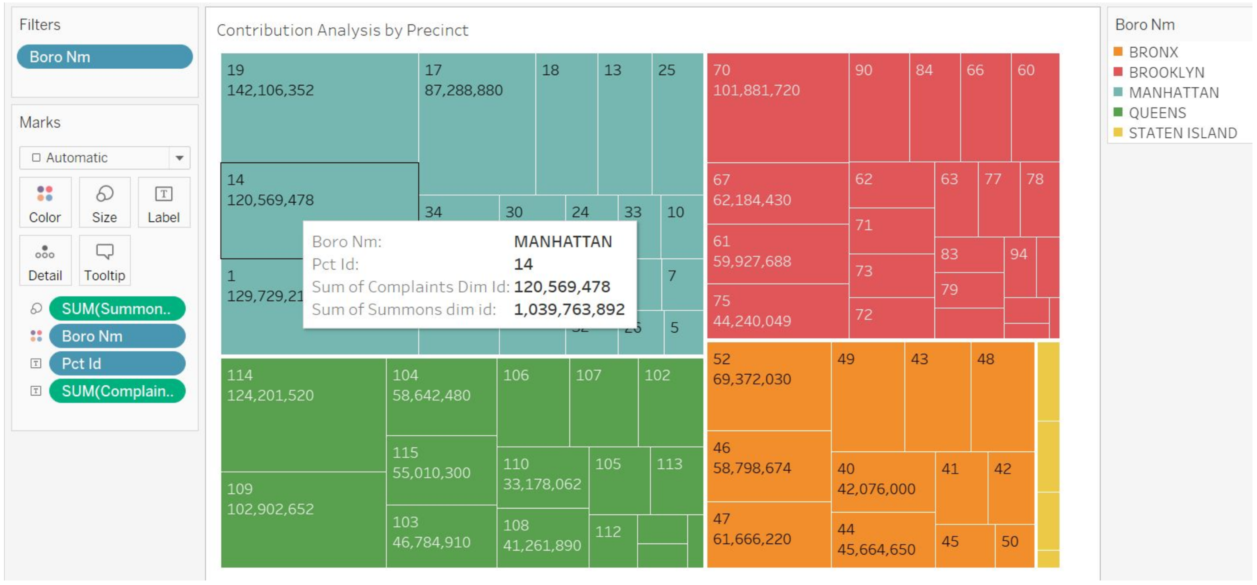
Trending Analysis - Complaints and Violations by Month

The best way to showcase the relationship between the sum of complaints and violations in 2016 was through a time-series analysis. The start of the year begins with 0 complaints and 0 violations. The first few months have low amount of complaints and violations relative to the other months. As the year progresses, we reach a pinnacle around July. This suggests that more crimes and violations happen around this summer month. The sum drops, but then increases again around October. Maybe this increase is related to students returning to college and people starting their jobs again after summer break. Lastly, the year ends strong with a decrease in crime in December. This trend analysis clearly shows there is a correlation between the sum of complaints and violations since both are increase and decrease in sync.



Contribution Analysis - Sum of Summons and Complaints by Precinct

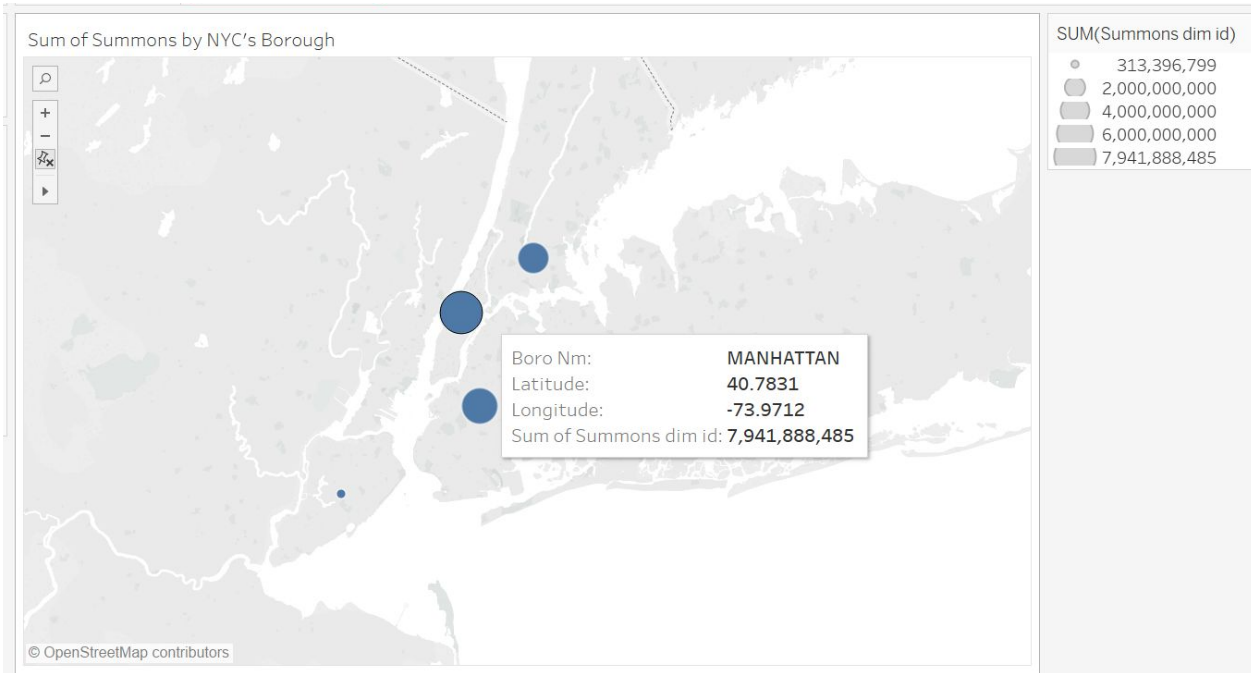
In order to see how the number of summons and complaints contributed to the total amount received in the 5 boroughs, a heatmap was used. This contribution analysis denotes that Manhattan has the highest number of complaints and summons, which may be a result of its high population. When you hover over a box in the heatmap, you can see the exact number of complaints and summons as well. The summons is represented by the size of the box and the number in each box represents the complaints. As the box's size increases, so does the number written inside of it. implying a direct correlation between summons and complaints.



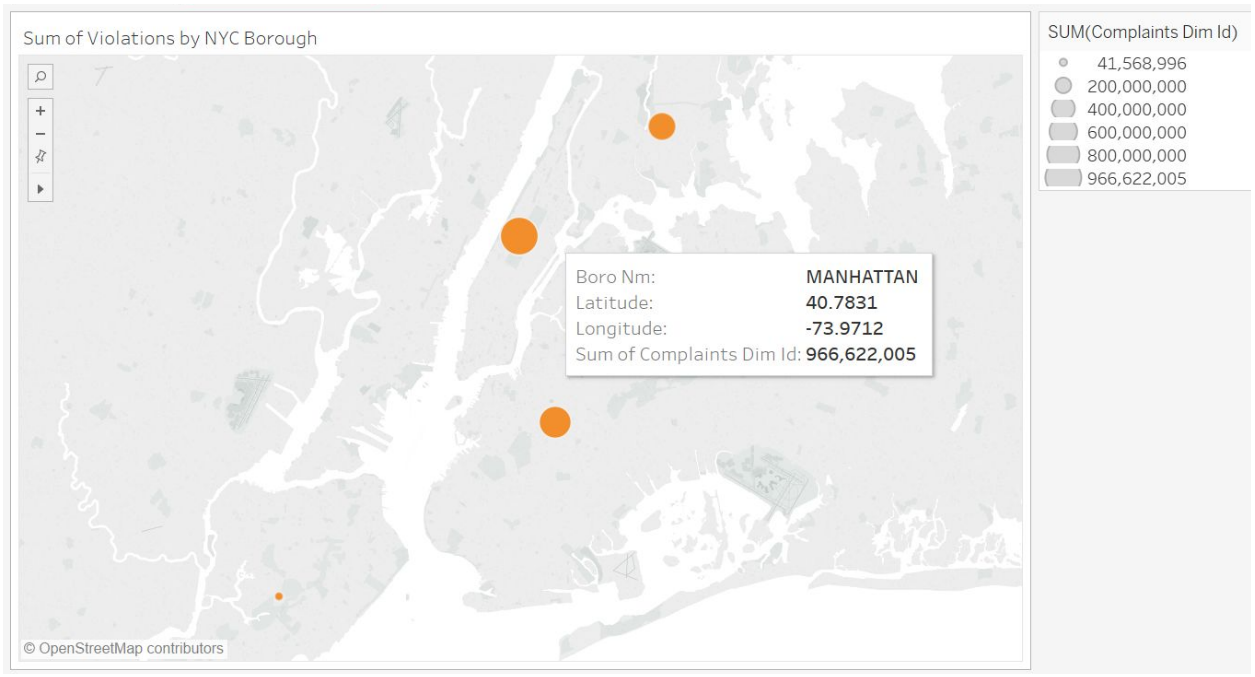
Boro Nm:	MANHATTAN
Pct Id:	14
Sum of Complaints Dim Id:	120,569,478
Sum of Summons dim id:	1,039,763,892

Geographic Analysis - Summons and Violations by NYC Borough

The heatmap gave us a good visual of how each precinct contributed the the total amount of complaints and violations, but a geographic analysis helped with visualizing how each borough contributed to the total number. Instead of graphing all the violations and summons by the precinct location, we used a central point to represent each borough for easier analysis. Once again, we see that the number of summons and violations are correlated with each other since the sizes of the circles are relatively similar and in the same rank in both maps.



Boro Nm: MANHATTAN
Latitude: 40.7831
Longitude: -73.9712
Sum of Summons dim id: 7,941,888,485



Boro Nm: MANHATTAN
Latitude: 40.7831
Longitude: -73.9712
Sum of Complaints Dim Id: 966,622,005

Conclusion

During this project we developed our experience in working with a data warehouse and using commercial database management systems and development tools while following the Kimball Lifecycle. It helped us appreciate all that goes into creating a data warehouse. We learned a lot about data cleaning and accurately set up dimensions for analytical purposes. This helped us through the design stages, conforming our data and creating analytic charts.

Figuring out meaningful KPIs from the original datasets was a challenging part that required research in what the industry considered important indicators. The ETL process was also a difficult step because we had to be very detail-oriented. Looking at each attribute carefully we were trying to predict what would have an effect on each other. We went through the ETL process many times, each time taking less time than the last because we were becoming more familiar with the process and tools. As we moved up through the data warehouse lifecycle, we forgot to incorporate grains in our fact tables and had to go back to the ER diagram for revisions many times as well.

We think by integrating these two sources of data, we were able to gain a more holistic view of different facts relating to crime in New York City. This data can now be used to better understand where and when crime poses a bigger threat. Location was also important in determining future trends and predictions in boroughs. Of course this was interesting for us, but it can really help the actually NYPD in the future too if they use these methodologies and diagrams. One benefit to this system would be the incorporation the violations description and payment amounts to the BI application to provide more color to the data. If we were to do it again, we would add more of these to the KPIs.

The Tableau dashboard was a nice way of tying all our work together and providing closure to our data warehousing project. We wouldn't be able to see it in colorful graphics showing clear percentages without the heat maps and contribution analysis. Additionally, this BI tool was very useful in ensuring our data was sourced correctly in the marts. Then we gained some predictions on our owns by designing them over and over again. These maps, diagrams, percentages, show us which boroughs are problematic based on read new data. This can possibly be used to be compared to other cities, be targeted by one category at a time, stored to see trends, or find values like averages, minimums, maximums, and much more over time.

Overall, we were able to handle the project in an extremely efficient way. We divided the work early and kept a clear focus of our goals throughout the semester and worked well together as a team.