## Data Warehouse Project on Illegal Parking

**Content**

Summary ....................................................................... 2

Data Source ....................................................................... 3

Tools and Languages Used ....................................................................... 3

KPIs ....................................................................... 4

Design of The Dimension Model ........................................................................ 4

Design of The ETL Process ....................................................................... 5 - 12

Design of The Data Warehouse ....................................................................... 13

Visualization & Analysis ....................................................................... 14

Conclusion ....................................................................... 19

Appendix 1 - Meeting Log ....................................................................... 20

Appendix 2 - SQL Script ....................................................................... 21 - 24

## Summary

As the city’s streets become increasingly congested with cars, the phenomenon of illegal parking has become a rising problem in NYC. A large amount of complaint calls are received by the NYC agencies and other affiliates regarding illegal parking, problems which include, but are not limited to:

* Parking on the sidewalk
* Parking in a prohibited space such as a bus lane, a commuter van stop, or in front of a fire hydrant
* Parking in resident spaces
* Parking in excess of the allowed time

Illegal parking emerges as one of the biggest culprits of traffic jams. It can create additional delays, especially on busy roads, and cause temporary bottlenecks that reduce traffic speeds. Imagine if a homeowner cannot park their car in front of their own home because someone has illegally parked along the curb. The problem of illegal parking not only causes traffic congestion but also brings inconvenience to a residents’ daily life.

Our project is aimed to design the ETL process and transform the dataset, obtained from NYC Open Data, into a data warehouse. By analyzing the data, we hope to help the agencies establish a functional parking management solution by efficiently using the existing parking lots and on-street parking throughout the city.

## 2. Data Source

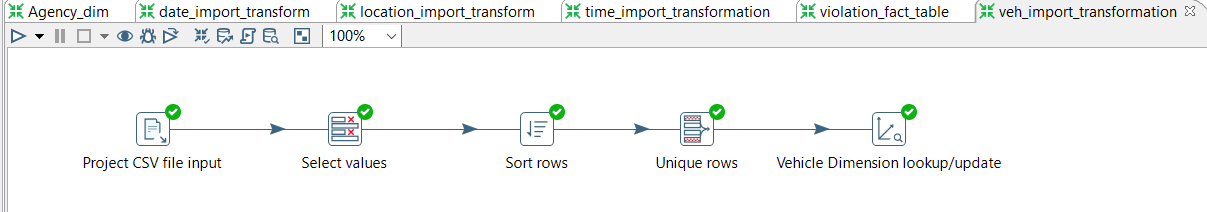
The original dataset was obtained from the NYC Open Data using Python API. The dataset contained over 700,000 records and ranged from 2017 January to December 2020. And the dataset contained more than 20 attributes, such as violation time, summons number, violation code, vehicle plate, violation location, and so on. The dataset is in a Comma Separated Value (.csv) file.

|  |
| --- |
|  |
| 3. Tools & Languages Used  * **Excel:** It was used both for the source data and the view table. We used it to format and review the data we could use for visualizations. * **Anaconda Jupyter Notebook:** In order to download the source data from NYC Open Data more efficiently, we used the Socrata API to retrieve the data and export it in Python code. * **Oracle Cloud:** We need to use the same wallet for the ATP database in order to have the same connection so we could transform each of our ETL into the same database. * **Pentaho Data Integration:** We used PDI to build the transformations for the dimensions and fact table. * **Oracle SQL Developer:** We used it for the final stage of ETL. After all the dimensional and fact tables were loaded into the database, we wrote SQL to check if we got the full columns and also create VIEW in order to import the data into Tableau to create visualizations. * **Tableau:** We used it to create the visualization and dashboard to better present the data and findings.  4. Potential KPIs  * Total number of violations per month * Total tickets in each borough per month * Total number of violations by weekday and hour * Total number of violations by county and quarter  5. Design of The Dimensional Model Diagram The image below is the dimensional model for our project, which is structured of six dimension tables and one fact table. The dimension tables provide the information about the illegal parking claim. The fact table is the numerical part. |
|  |

## 6. Design of the ETL Process

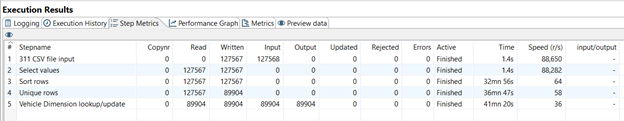
In the ETL process, we extract the source data and transform it into the data warehouse. And our data warehouse has six dimensions and one fact table. Each of them has their own transformation.

**6.1 Vehicle Dimension Table Creation**

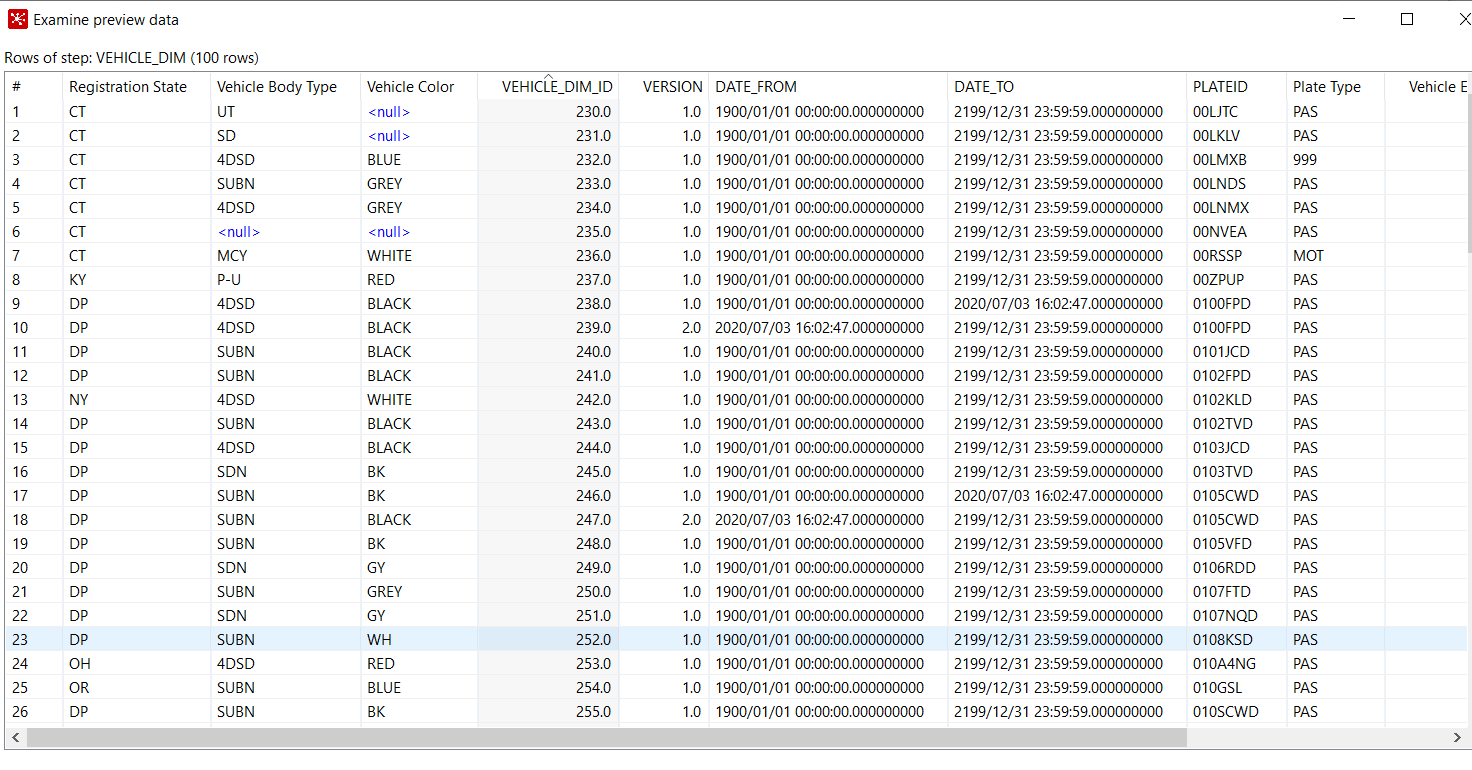


(Figure 1. Vehicle Transformation Process)

As Figure 1 shows, Vehicle Dimension transformation consists of five steps. First, a **CSV File Input** is created to load the Parking\_Violations\_Issued.csv file that was exported from NYC Open Data. Next, **Select Values** is used to select the fields for the Vehicle Dimension. In the **Sort Rows** step, the field PlateID is sorted ascendingly along with other fields. And then, **Unique Rows** is used to remove all duplicates from the input stream. Lastly, we set up a **Dimension lookup/update** step for slowly changing. As a result, a new primary key is created called vehicle\_dim\_id.

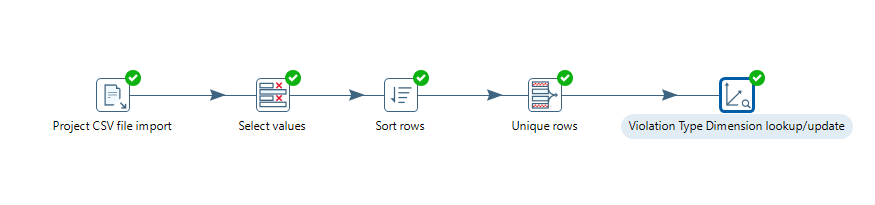


(Figure 2. Step Metrics - Vehicle)



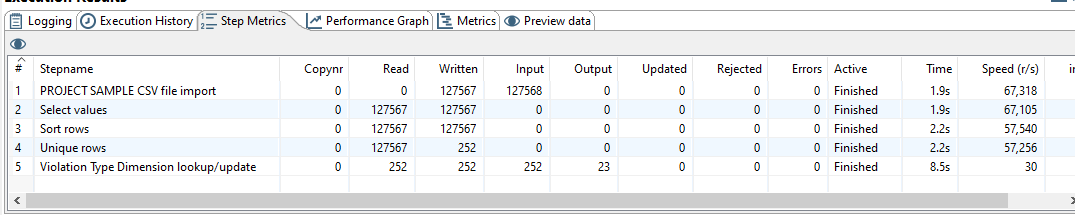
(Figure 3. Sample Output of Vehicle Transformation)

**6.2 Violation Type Dimension Table Creation**

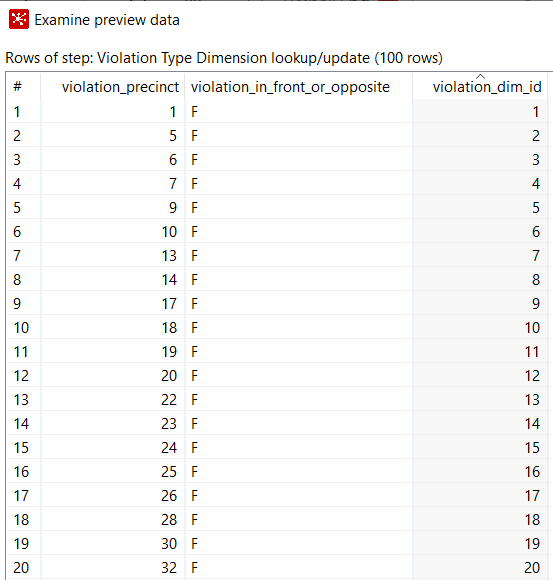


(Figure 4. Violation Transformation Process)

In the transformation process of the violation table, the first step is to load the Parking\_Violations\_Issued.csv file using a **CSV File Input** step**.**  Next, we will select the fields that need to be on Violation Type Dimension. We sort the rows with Violation Precinct and Violation in front of or Opposite fields and remove all duplicates from the input stream using **Unique Rows**. Lastly, we set up a **Dimension lookup/update** step and create a new primary key called violation\_dim\_id.

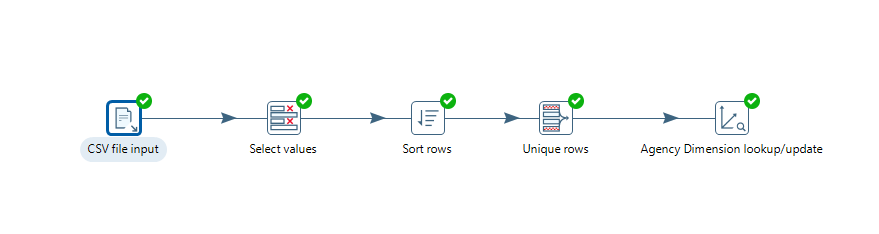


(Figure 5. Step Metrics - Violation)



(Figure 6. Sample Output of Violation Transformation)

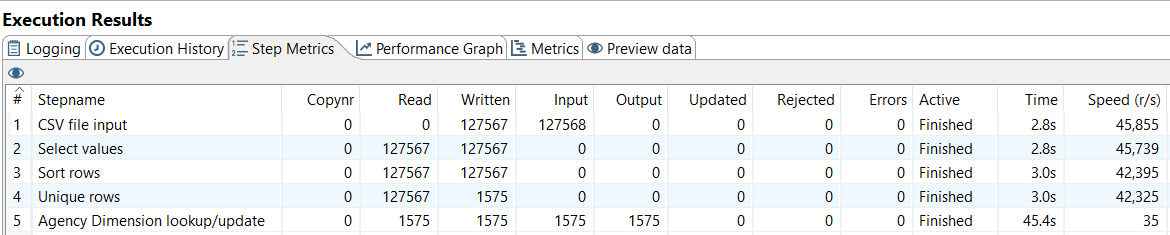
**6.3 Agency Dimension Table Creation**



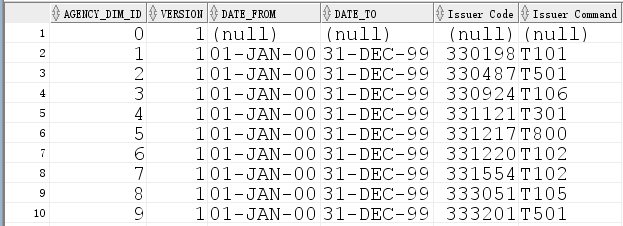
(Figure 7. Agency Transformation Process)

There are several steps to create an Agency dimension table. See below:

First, we create a **CSV File Input** to load the Parking\_Violations\_Issued.csv file that was exported from NYC Open Data. Next, we will select the fields for the Agency Dimension. We sort the rows with Issuer Code and Issuer Command fields. And then, a Unique Rows is used to remove all duplicates from the input stream. Lastly, we set up a **Dimension lookup/update** step for slowly changing. As a result, a new primary key is created called agency\_dim\_id.

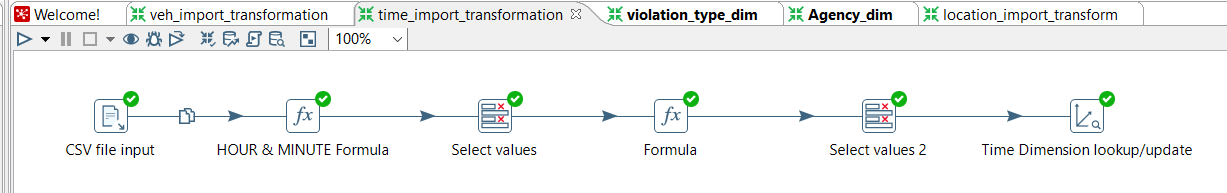


(Figure 8. Step Metrics - Agency)



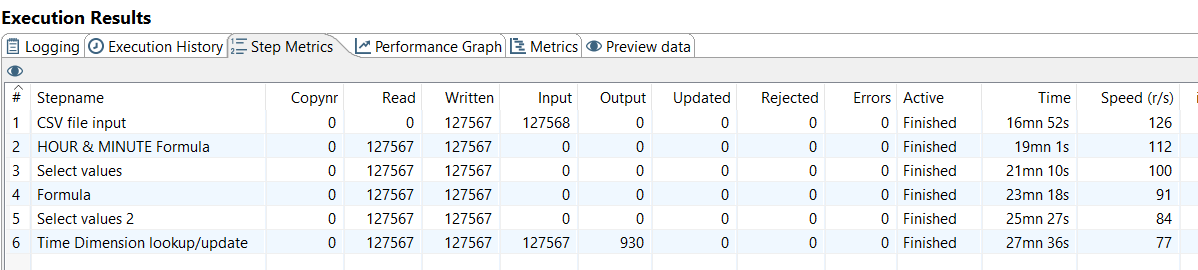
(Figure 9. Sample Output of Agency Transformation)

**6.4 Time Dimension Table Creation**

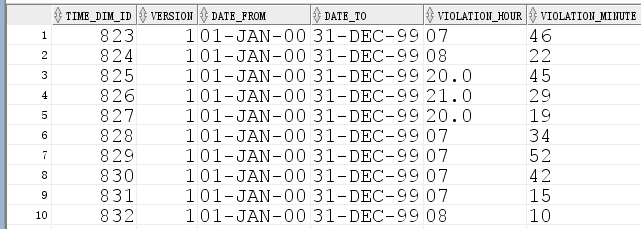


(Figure 10. Time Transformation Process)

In the Time transformation process, there are six steps. We first input the source CSV file Parking\_Violations\_Issued.csv and only keep violation\_time field. After the input step, a **Function** step is used to split up the hours and minutes by using the MID function. In the second **Function** step, We use the IF function to convert AM/PM to a 24-hour clock and replace it with the violation\_hour that is created in the previous step. Next, we select the violation\_hour and violation\_minute and add them to the **Time Dimension lookup/update** step where time\_dim\_id is created.

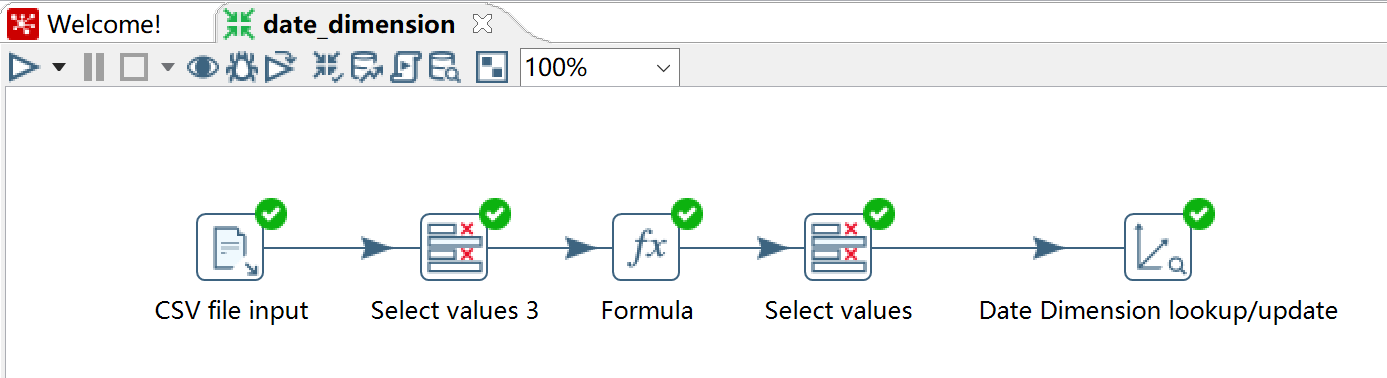


(Figure 11. Step Metrics - Time)



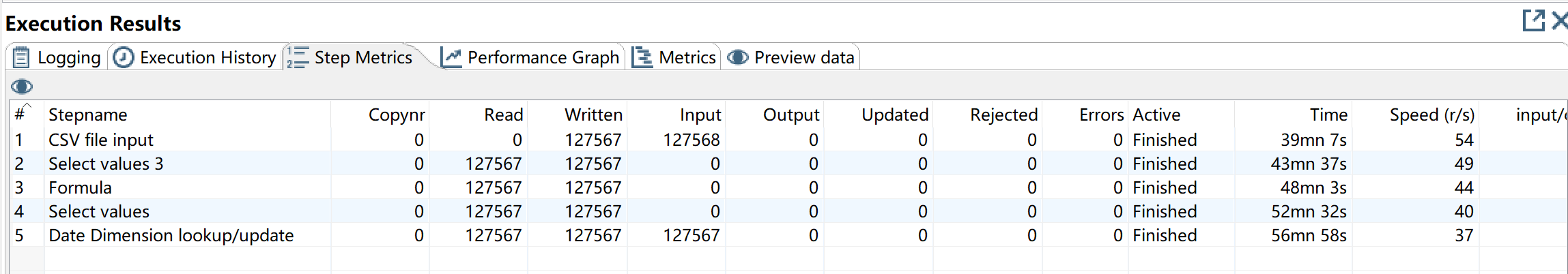
(Figure 12. Sample Output of Time Transformation)

**6.5 Date Dimension Table Creation**

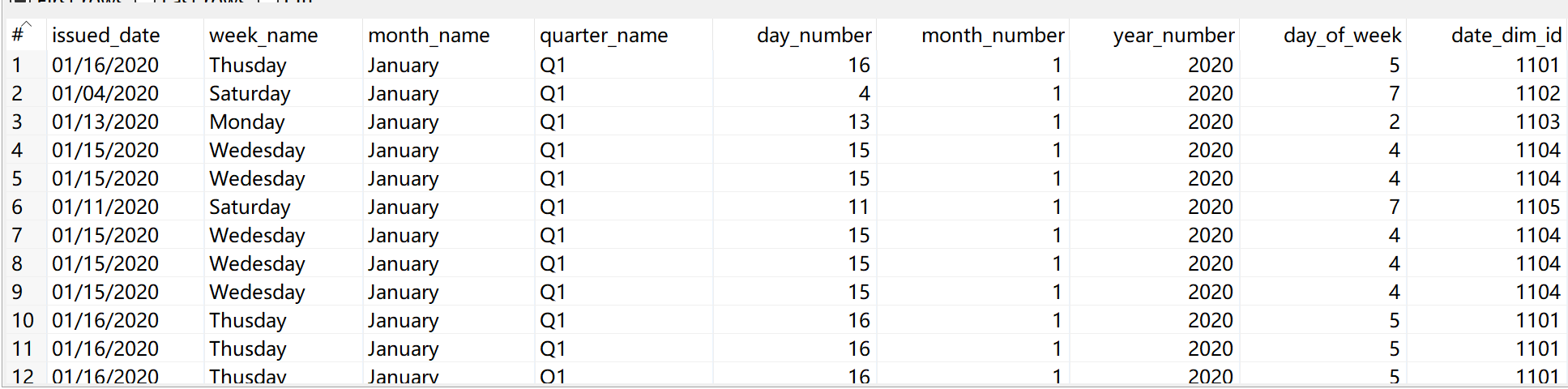
****

(Figure 13. Date Transformation Process)

The first step of creating Date Dimension is to integrate the Parking\_Violations\_Issued.csv file into PDI using the step **CSV File Input**. The **Select Values** is used to select violation\_time from the dataset. The next step is to manually create each field in the Date Dimension using the **FORMULA** step. And then we use another **Select Values** step to select all the new fields that are created in the previous step. A primary key called Date\_dim\_id is created in the **Date Dimension lookup/update** step.



(Figure 14. Step Metrics - Date)



(Figure 15. Sample Output of Vehicle Transformation)

## 6.6 Location Dimension Table Creation

## (Figure 16. Location Transformation Process)

## As Figure 16 shows, Location Transformation consists of five steps. First, a CSV File Input is created to load the Parking\_Violations\_Issued.csv file that was exported from NYC Open Data. Next, Select Values is used to select the fields for the Location Dimension. In the Sort Rows step, the field County is sorted ascendingly along with other fields. And then, Unique Rows is used to remove all duplicates from the input stream. Lastly, we set up a Dimension lookup/update step for slowly changing. As a result, a new primary key is created called location\_dim\_id.

## (Figure 17. Step Metrics - Location)

## (Figure 18. Sample Output of Location Transformation)

## 

## 6.7 Violation Fact Table Creation

## 

## (Figure 19. Violation Fact Transformation Process)

## Now we have all the dimensions loaded into the data warehouse, the Fact table is prepared for loading. We name it “Violation Fact” table and it has six inputs which consists of all records from 2017 to 2019. As the fact table of the star schema, the Violation Fact Second, we create a new Dimension Lookup/Update step to the transformation for every dimension table that we have created. Third, we connect all of the dimension lookup steps. Finally, we use a Table Output step to the end of the transformation and name it “Violation Fact table”.

## 

## (Figure 20. Step Metrics - Violation Fact)

## 

## (Figure 21. Sample Output of Violation Fact Transformation)

## 

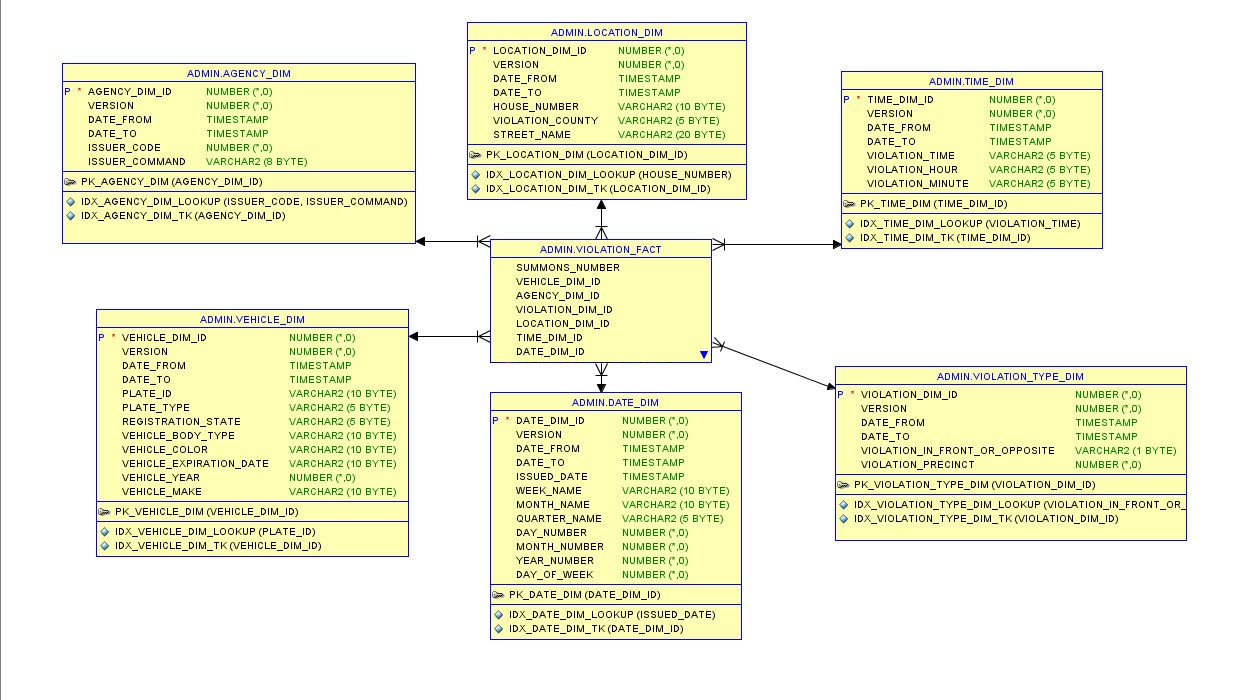
## 

## 

## 

## 7. Design of Data Warehouse

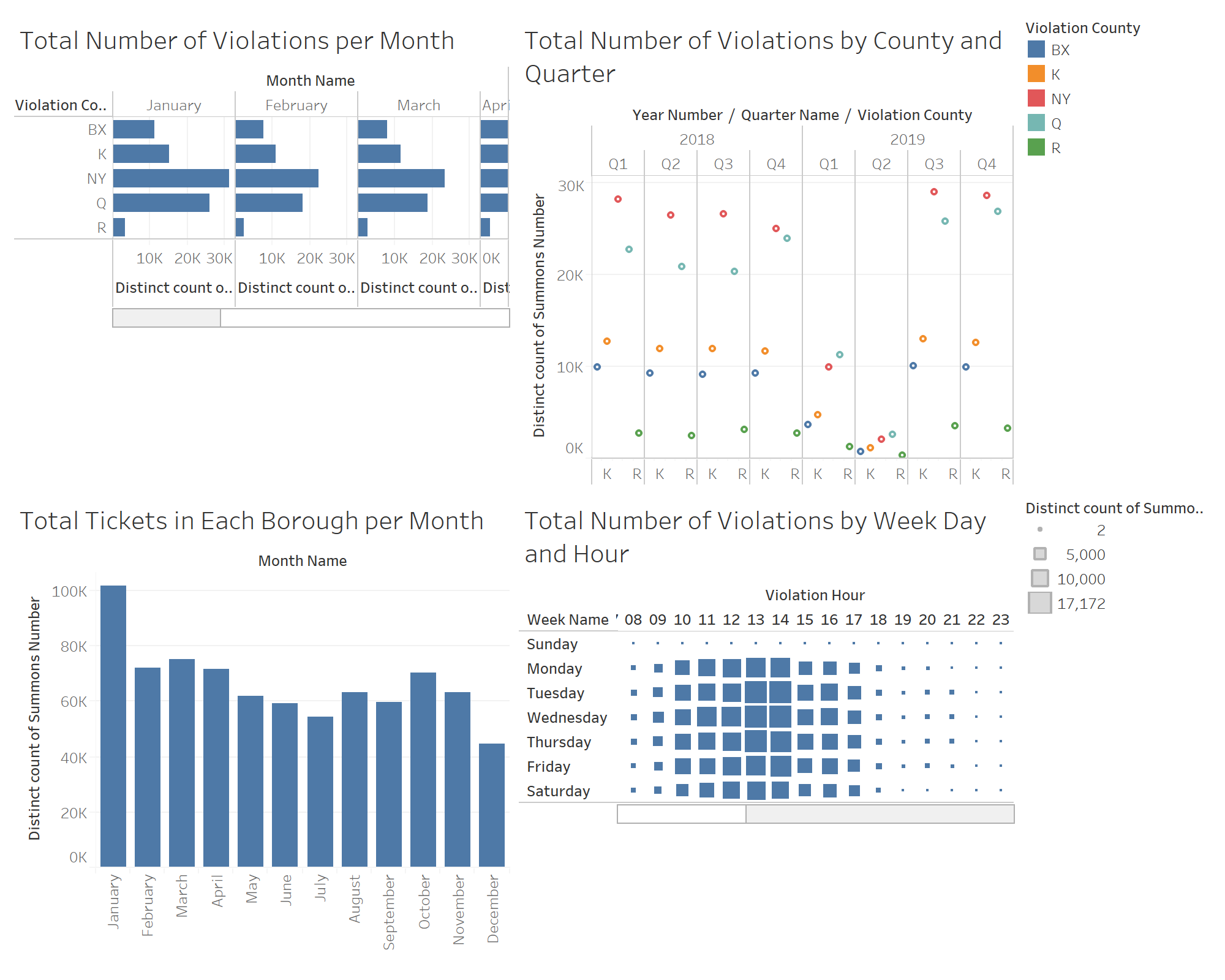
The data warehouse for the fact and dimension tables is modeled based on the star schema. The following diagram shows the details in each dimension and fact table. Each dimension in the star schema is represented with its own table. And the dimension table contains its descriptive attributes and the primary key. The Violation Fact table is in the center and consists of keys to each of six dimensions.

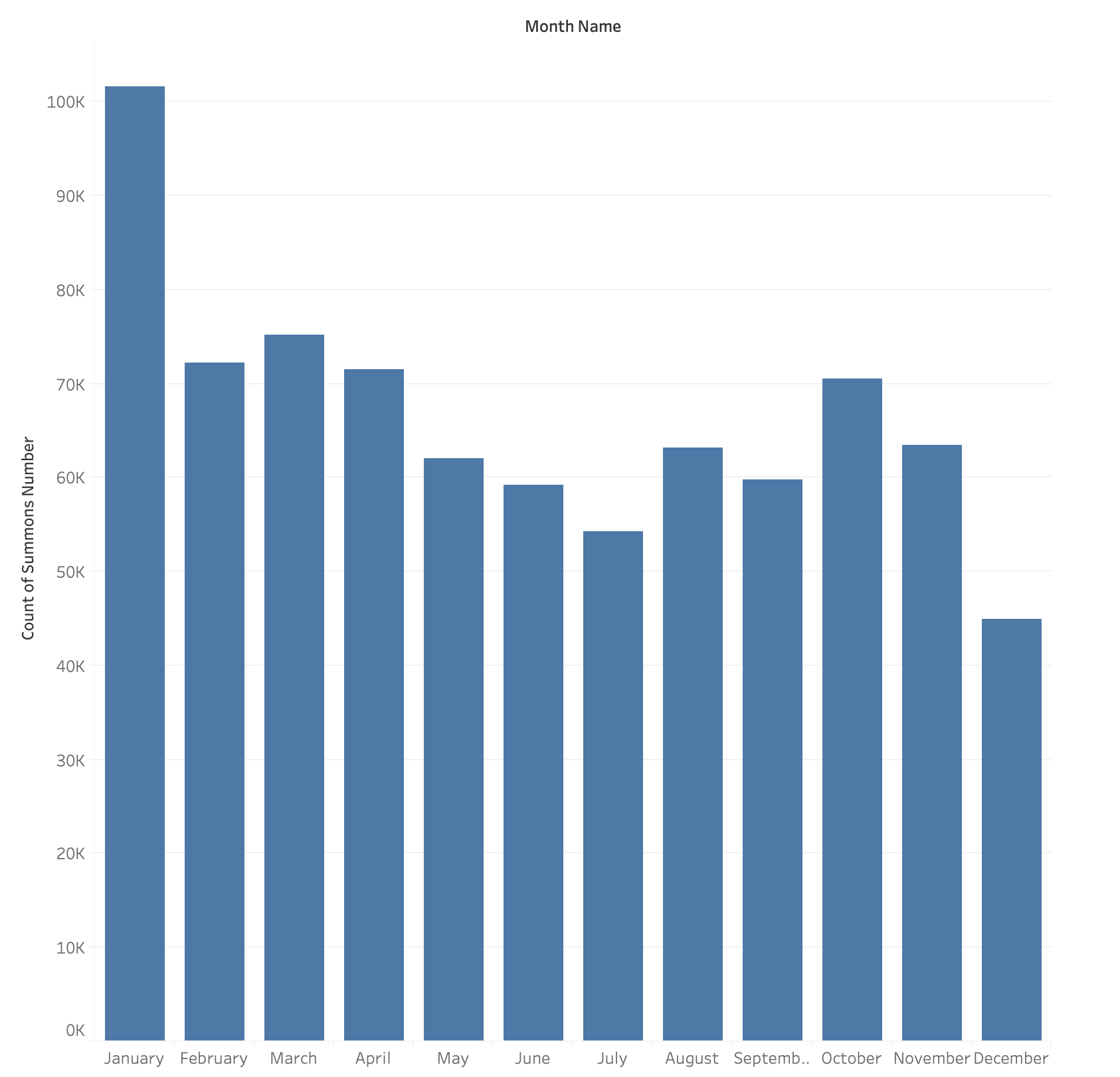


## 9. Visualization & Analysis

We create the data visualization dashboard on Tableau to carry out visualization and analysis for the following KPIs:

* Total number of violations per month
* Total tickets in each borough per month
* Total number of violations by weekday and hour
* Total number of violations by county and quarter



**9.1 <Total number of violations per month>**

As the graph shows, January has the highest number of violations issued. On the other hand, December is the lowest number of violation tickets. From December to January, an approximately 60000 has been increased, which reveals the highest gap within a month.

**9.2 <Total tickets in each borough per month >**

## 

\*NY= New York

Q = Queens

K = Kings

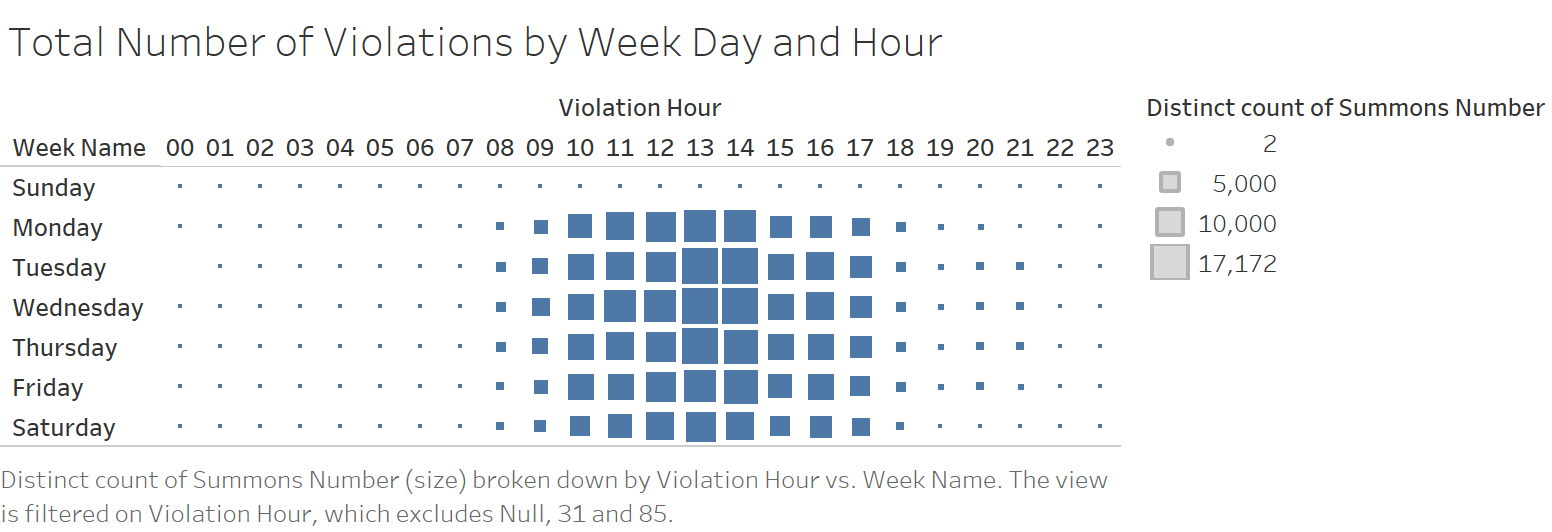
BX =Bronx

R =Richmond

The graph above illustrates that NewYork county has the highest total tickets among all counties. The second highest is Queens, then Kings, Bronx and Richmond accordingly.

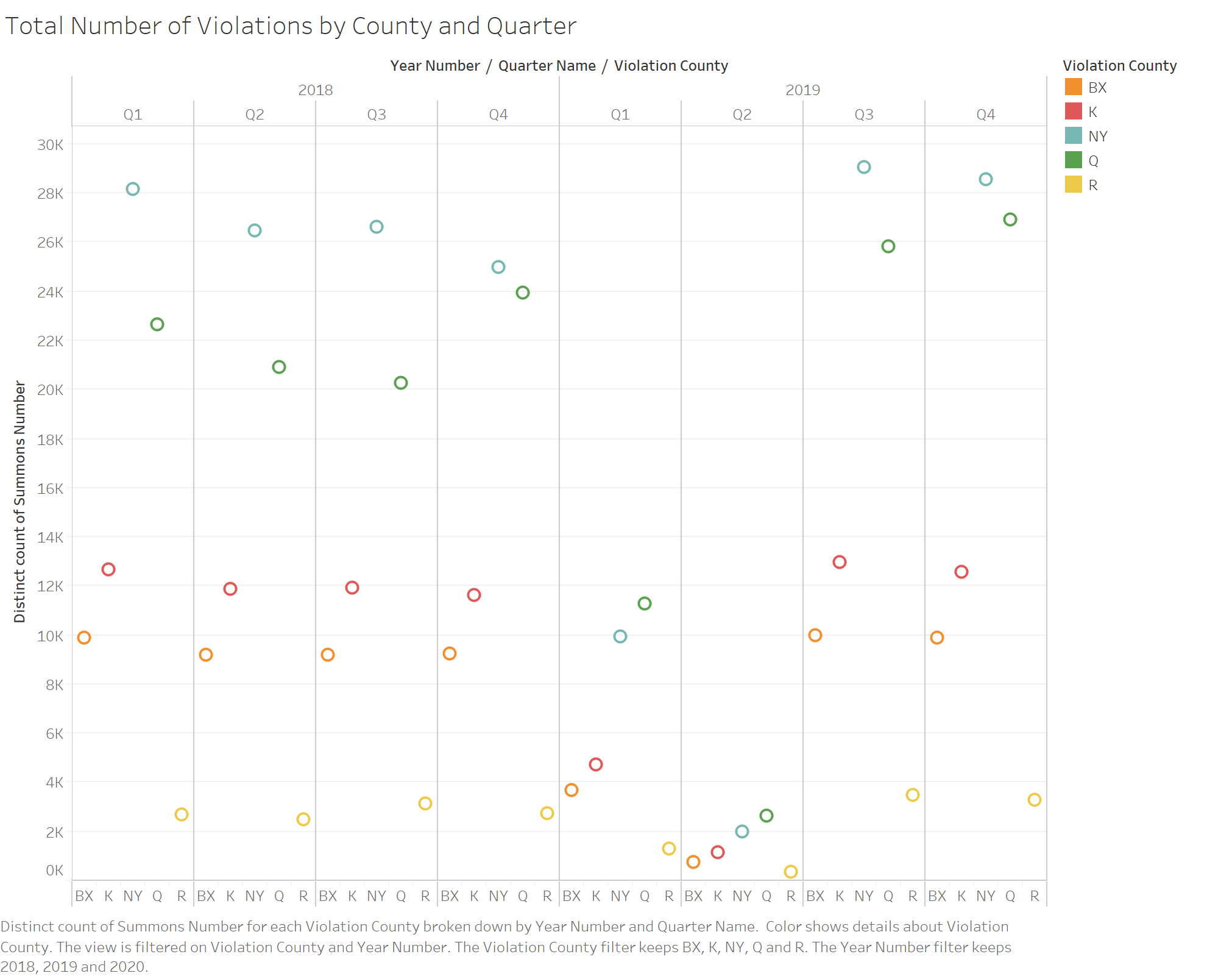
New York counties consist of 62 counties in total. According to New York Demographics, the highest population by counties displays Kings, Queens, New York, Bronx and then Richmond. Therefore, the number of population per county does not influence on where the number of violation tickets were issued.

**9.3 <Total number of violations by weekday and hour >**



From the graph above we know that during the past three years, the summons were issued mostly on weekdays (Monday through Friday), especially Tuesday to Thursday during noon hours. According to such analysis, we could provide the suggestions to the issuing agencies and related departments that they should designate more issuers or agencies during noon working hours/slots since there are more violations. Human resources and operations could also utilize such information to better staff employees and set monitoring cameras to better detect the parking violations on the street.

**9.4 <Total number of violations by county and quarter>**



The graph shows there is a change of the cases of violations during the past two years among the five boroughs. Every county all shows this dramatic decrease of summons during the first and second quarter of 2019. If the source data is relatively accurate, we would wonder what happened during that time that successfully cut off the violations, although the number of cases went higher again since the third quarter. We would suggest the departments or agencies to look up for potential factors that would be conducive to reduce the violations. Necessary initiatives and policies should be published and enforced to improve the parking violations.

## 

## 10. Conclusion

The group used several software and database tools to coordinate and manage the project as well as carry out the programming tasks such as Pentaho, Oracle SQL Developer, and Tableau. The most difficult step in the project was the data source extracting. Our dataset is extremely large, and the data type is inconsistent across different years. Before working on the project, we never imagined how difficult it would be for us to get a data source working. And we know that the data quality could affect our analysis. In order to ensure our dataset is in good quality, we clean and detect the bad data. Handling poor bad is a bit time-consuming. Hence, data cleaning is a challenging step. The easiest step would be designing dimension tables. Once we come up with our KPIs, it is easy to identify the attributes that we will need for each dimension.

We learned numerous things through this project. Starting off, we were concerned about using technologies for a first-time data warehouse builder. The technologies like Pentaho and Tableau are brand new to us before the project. And applying the concepts in real-world practice is a completely new learning experience.

If there is a change to do the project all over again, we would definitely choose two or three smaller size data warehouses rather than an extremely large one. Overall, this project explores the functions of each software as well as enhances our skills of software we utilize.

## 

## Appendix 1 - Meeting Log

|  |  |  |
| --- | --- | --- |
| **Date** | **Attendees** | **Agenda** |
| 05/16 | Gabriella Qi  Shailyn Lin  Shuting Chen  Yukiko Kurashimas | Pick project idea  Identify KPIs and data source  Start project proposal |
| 05/29 | Gabriella Qi  Shailyn Lin  Shuting Chen  Yukiko Kurashimas | Start dimensional modeling  Choose attribute  Design the Dimension tables and the Fact table |
| 06/27 | Gabriella Qi  Shailyn Lin  Shuting Chen  Yukiko Kurashimas | Start ETL process discussion and how to fetch the dataset |
| 07/02  & 07/05 | Gabriella Qi  Shailyn Lin  Shuting Chen  Yukiko Kurashimas | Continue discussing the ETL process |
| 07/11 | Gabriella Qi  Shailyn Lin  Shuting Chen  Yukiko Kurashimas | Create View and finalize the schema  Discuss the graphical analytics  Finalize the report |

## 

## 

## 

## 

## 

## Appendix 2 - SQL Script

1. Codes for Vehicle Dimension

CREATE TABLE ADMIN.vehicle\_dim

(

vehicle\_dim\_id INTEGER

, version INTEGER

, date\_from TIMESTAMP

, date\_to TIMESTAMP

, plate\_id VARCHAR(10)

, plate\_type VARCHAR(5)

, registration\_state VARCHAR(5)

, vehicle\_body\_type VARCHAR(10)

, vehicle\_color VARCHAR(5)

, vehicle\_expiration\_date VARCHAR()

, vehicle\_year INTEGER

, vehicle\_make VARCHAR(10)

)

;CREATE INDEX idx\_vehicle\_dim\_lookup ON ADMIN.vehicle\_dim(plate\_id)

;

CREATE INDEX idx\_vehicle\_dim\_tk ON ADMIN.vehicle\_dim(vehicle\_dim\_id)

;

1. Codes for Agency Dimension

CREATE TABLE ADMIN.Agency\_Dim

(

agency\_dim\_id INTEGER

, version INTEGER

, date\_from DATE

, date\_to DATE

, "Issuer Code" INTEGER

, "Issuer Command" VARCHAR2(4)

, PRIMARY KEY (agency\_dim\_id)

)

;CREATE INDEX idx\_Agency\_Dim\_lookup ON ADMIN.Agency\_Dim("Issuer Code", "Issuer Command")

;

CREATE INDEX idx\_Agency\_Dim\_tk ON ADMIN.Agency\_Dim(agency\_dim\_id)

;

1. Codes for Location Dimension

CREATE TABLE ADMIN.location\_dim

(

location\_dim\_id INTEGER

, version INTEGER

, date\_from DATE

, date\_to DATE

, house\_number VARCHAR2(10)

, County VARCHAR2(5)

, Street\_Name VARCHAR2(20)

, City VARCHAR2(5)

, State VARCHAR2(5)

, PRIMARY KEY (location\_dim\_id)

)

;CREATE INDEX idx\_location\_dim\_lookup ON ADMIN.location\_dim(house\_number)

;

CREATE INDEX idx\_location\_dim\_tk ON ADMIN.location\_dim(location\_dim\_id)

;

1. Codes for Date Dimension

CREATE TABLE ADMIN.date\_dim

(

date\_dim\_id INTEGER

, version INTEGER

, date\_from DATE

, date\_to DATE

, issued\_date DATE

, week\_name VARCHAR2(10)

, month\_name VARCHAR2(10)

, quarter\_name VARCHAR2(5)

, day\_number INTEGER

, month\_number INTEGER

, year\_number INTEGER

, day\_of\_week INTEGER

, PRIMARY KEY (date\_dim\_id)

)

;CREATE INDEX idx\_date\_dim\_lookup ON ADMIN.date\_dim(issued\_date)

;

CREATE INDEX idx\_date\_dim\_tk ON ADMIN.date\_dim(date\_dim\_id)

;

1. Codes for Violation Type Dimension

CREATE TABLE ADMIN.Violation\_Type\_Dim

(

violation\_dim\_id INTEGER

, version INTEGER

, date\_from DATE

, date\_to DATE

, "Violation In Front of Or Opposite" CHAR(1)

, "Violation Precinct" INTEGER

, PRIMARY KEY (violation\_dim\_id)

)

;CREATE INDEX idx\_Violation\_Type\_Dim\_lookup ON ADMIN.Violation\_Type\_Dim("Violation In Front of Or Opposite", "Violation Precinct")

;

CREATE INDEX idx\_Violation\_Type\_Dim\_tk ON ADMIN.Violation\_Type\_Dim(violation\_dim\_id)

;

1. Codes for Violation Fact table

CREATE TABLE ADMIN.violation\_fact

(

"Summons\_Number" INTEGER

, vehicle\_dim\_id INTEGER

, agency\_dim\_id INTEGER

, violation\_dim\_id INTEGER

, location\_dim\_id INTEGER

, time\_dim\_id INTEGER

, date\_dim\_id INTEGER

)

;

1. Codes for the VIEW

CREATE VIEW view1 AS

SELECT

a.AGENCY\_DIM\_ID,

a.ISSUER\_CODE,

a.ISSUER\_COMMAND,

d.DATE\_DIM\_ID,

d.ISSUED\_DATE,

d.WEEK\_NAME,

d.MONTH\_NAME,

d.QUARTER\_NAME,

d.DAY\_NUMBER,

d.MONTH\_NUMBER,

d.YEAR\_NUMBER,

d.DAY\_OF\_WEEK,

l.LOCATION\_DIM\_ID,

l.HOUSE\_NUMBER,

l.VIOLATION\_COUNTY,

l.STREET\_NAME,

t.TIME\_DIM\_ID,

t.VIOLATION\_TIME,

t.VIOLATION\_HOUR,

t.VIOLATION\_MINUTE,

vt.VIOLATION\_PRECINCT,

vt.VIOLATION\_DIM\_ID,

v.VEHICLE\_COLOR,

v.PLATE\_TYPE,

v.VEHICLE\_MAKE,

v.PLATE\_ID,

v.VEHICLE\_BODY\_TYPE,

v.VEHICLE\_DIM\_ID,

v.REGISTRATION\_STATE,

v.VEHICLE\_EXPIRATION\_DATE,

v.VEHICLE\_YEAR,

vf.summons\_number

FROM agency\_dim a INNER JOIN violation\_fact vf ON a.AGENCY\_DIM\_ID = vf.AGENCY\_DIM\_ID

INNER JOIN vehicle\_dim v ON v.VEHICLE\_DIM\_ID = vf.VEHICLE\_DIM\_ID

INNER JOIN violation\_type\_dim vt ON vt.VIOLATION\_DIM\_ID = vf.VIOLATION\_DIM\_ID

INNER JOIN location\_dim l ON l.LOCATION\_DIM\_ID = vf.LOCATION\_DIM\_ID

INNER JOIN date\_dim d ON d.DATE\_DIM\_ID = vf.DATE\_DIM\_ID

INNER JOIN time\_dim t ON t.TIME\_DIM\_ID = vf.TIME\_DIM\_ID;