Group 4: Final Project

311 Complaint Data vs. NYC Collision Data

CIS 4400 CMWA
Group 4
Alan Anthony Fridburg (alan.fridburg@baruchmail.cuny.edu)
Wen Bi (wen.bi@baruchmail.cuny.edu)
Steven Le (steven.le@baruchmail.cuny.edu)
William D Perez (william.perez2@baruchmail.cuny.edu)
Lorena Madelin Vasquez(lorena.vasquez@baruchmail.cuny.edu)

Type of 311 Complaint: Illegal Parking

Business Problem: Does Illegal Parking cause or correlate with the rising rate of Car Accidents?

Narrative Description:

As New Yorkers begin to return to their normal pre-covid routines, car accidents are becoming more frequent. In doing so this has been recently increasing car insurance rates. Whether people are heading back to school or work, the same people returning to normal life are looking for areas to park—sometimes illegal. It is possible these frequent car accidents are caused by illegal parking.

To create a solution that protects their citizens, the mayor, city council, and other public servants decided to investigate these claims and reports.

A significant part of their research is investigating recent 311 complaints and whether illegal parking in certain areas of the city is linked or correlated with recent cases of car accidents. During their research, the team will be focusing on critical KPIs that will help create the correlation between illegal parking complaints and recent car accidents.

Finalized KPI's the team will be analyzing are:

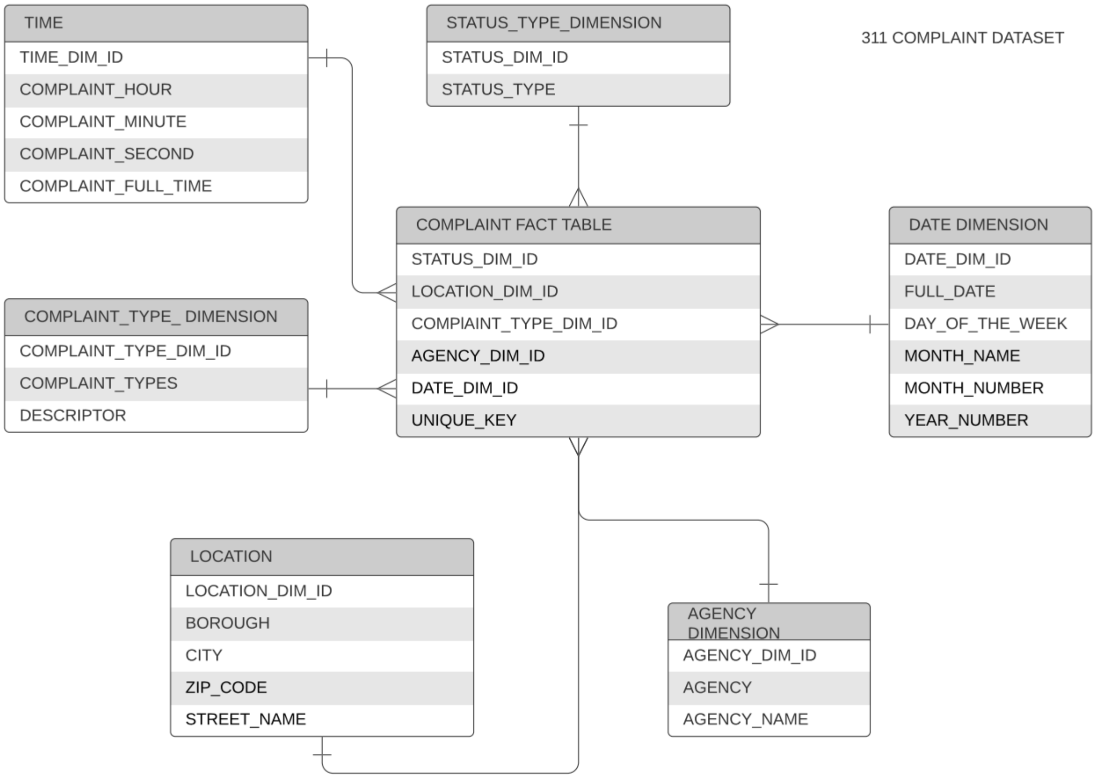
- Numbers of 311 complaints per Borough
- Numbers of illegal parking tickets per Day
- Numbers of Car accidents per Borough (city wide)
- Numbers of illegal parking tickets per Borough
- Numbers of Car Accidents per Borough
- Numbers of Mortality per Month
- Numbers of Mortality per Borough
- Numbers of Cars involved per Accident
- Numbers of [complaint type] per Day
    - **Finalized complaint types:**
        - Illegal Parking
        - Street Condition
        - Street light condition

Finalize Dataset:

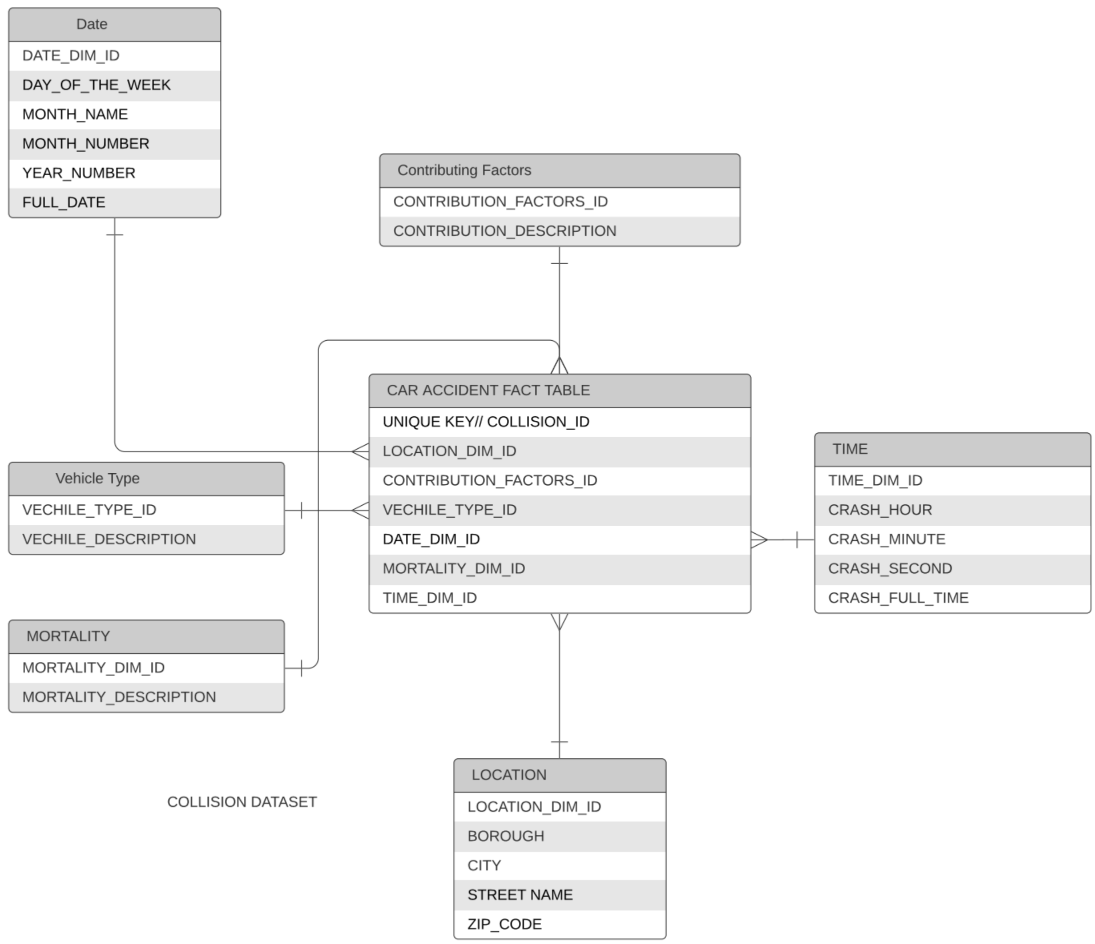
- 311 Dataset (NYC OpenData)
    - Transaction Grain
- Motor Vehicle Collisions - Crashes (NYC OpenData)
    - Transaction Grain

Insights: Before creating our dimension models for our datasets, we chose transaction grain as the best way to view our business operations.

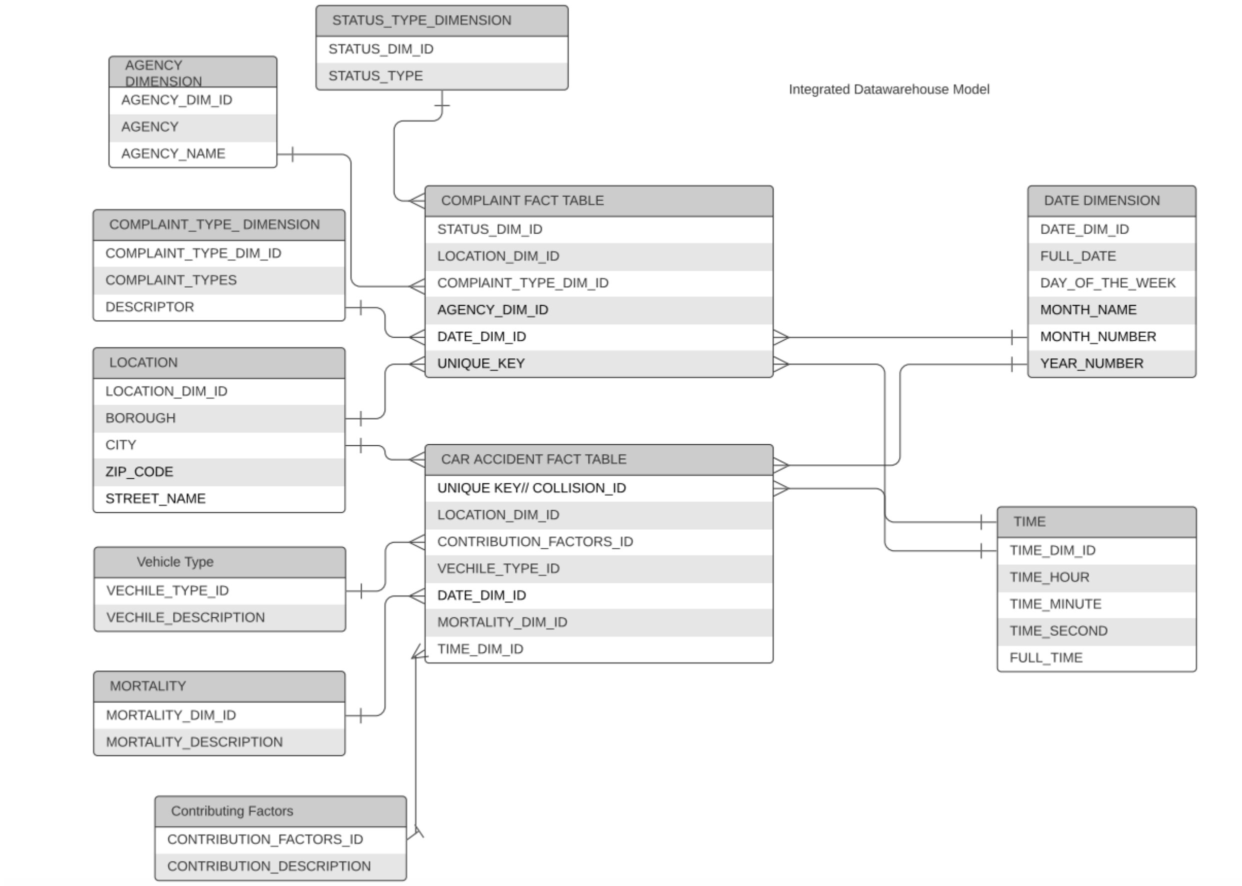
Dimensional Modeling: 311 Complaint Dataset



Dimensional Modeling: New York Collision Dataset



Dimensional Modeling: Integrated Datawarehouse Model



Schema Summary:

Our final dimensional schema, which is our integrated model, shows how the fact tables and dimensions we created are connected. Our final schema will be used as a point of reference throughout the rest of the project. The fact table contains values that will help calculate our KPIs. For example, for our business problem, we have two fact tables necessary for both our datasets: "COMPLAIINT_FACT_TABLE" and "CAR_ACCIDENT_FACT_TABLE." Both fact tables are connected by the location_dimension, the time_dimension, and the date_location.

Data Profiling, ETL Tools, and DBMS

We have chosen to use SQL as our primary ETL tool to help with our ETL programming. In addition, we will be using dbt to help create our dimensional model according to our integrated schema. We are using BigQuery on the Google Cloud Platform as our target DBMS.

We used both python and R to clean our dataset from unwanted data entry and generate data summary reports, which provided valuable insights on our dataset, variables, and potential KPIs.

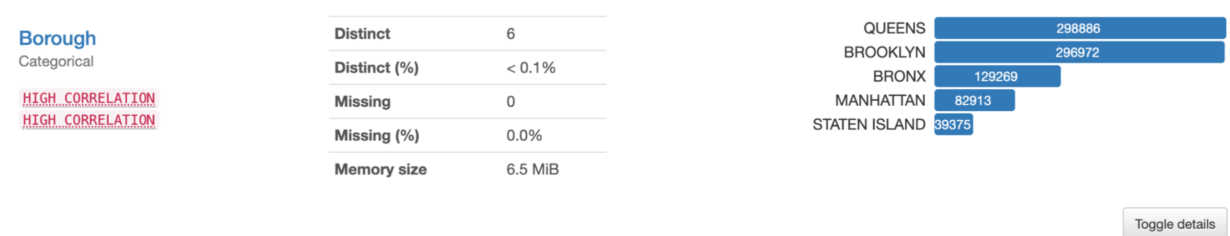
Chosen ETL Tool:

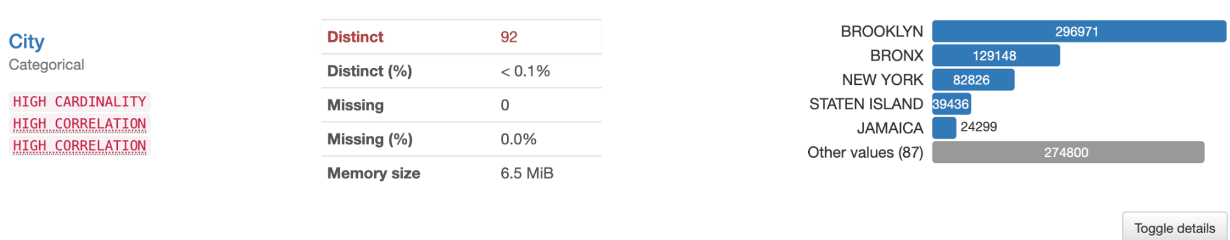
- SQL

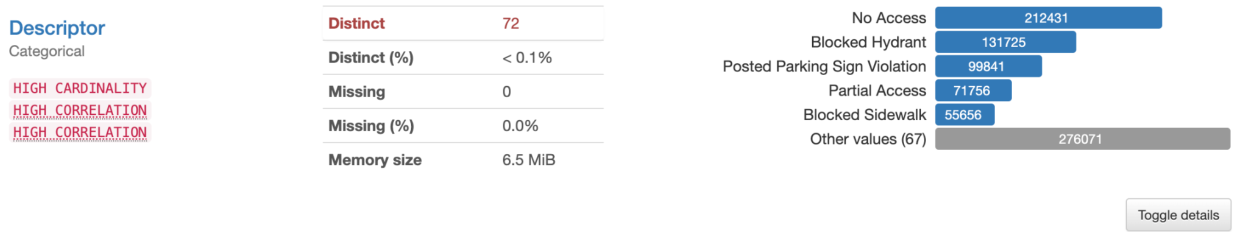
Chosen DBMS:

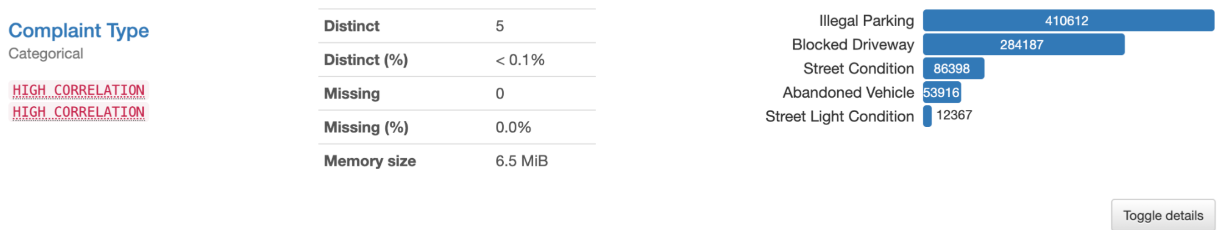
- Big Query

Data Profiling for [311 Dataset](#) (NYC OpenData)







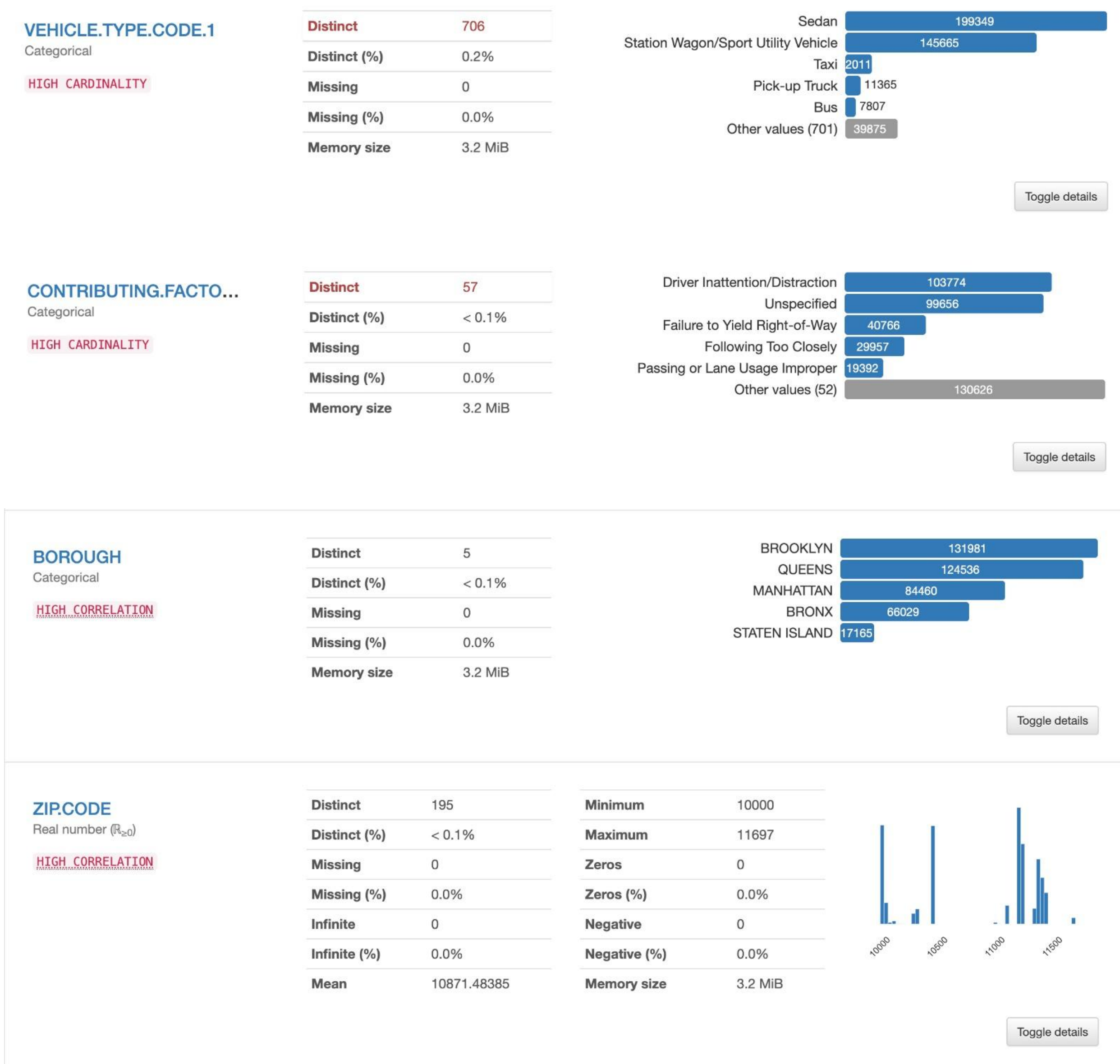


311 Dataset - Data Profile Summary:

After generating the summary data report from our 311 dataset, we were able to draw some initial insights into our complaint data, leading us one step closer to drawing a reasonable conclusion about whether illegal parking, blocked driveways, street condition, and other complaint types coordinate with recent accidents from our collision dataset.

From our 311 dataset, we can conclude that both Brooklyn and Queens had the most complaints that are related to illegal parking, followed by blocked driveawy, street conditions, abandoned vehicles, and flawed street light.

Data Profiling for [Motor Vehicle Collisions - Crashes](#) (NYC OpenData)



VEHICLE.TYPE.CODE.1
Categorical
HIGH CARDINALITY

Distinct	706
Distinct (%)	0.2%
Missing	0
Missing (%)	0.0%
Memory size	3.2 MiB

Sedan	199349
Station Wagon/Sport Utility Vehicle	145665
Taxi	2011
Pick-up Truck	11365
Bus	7807
Other values (701)	39875

Toggle details

CONTRIBUTING.FACTO...
Categorical
HIGH CARDINALITY

Distinct	57
Distinct (%)	< 0.1%
Missing	0
Missing (%)	0.0%
Memory size	3.2 MiB

Driver Inattention/Distraction	103774
Unspecified	99656
Failure to Yield Right-of-Way	40766
Following Too Closely	29957
Passing or Lane Usage Improper	19392
Other values (52)	130626

Toggle details

BOROUGH
Categorical
HIGH CORRELATION

Distinct	5
Distinct (%)	< 0.1%
Missing	0
Missing (%)	0.0%
Memory size	3.2 MiB

BROOKLYN	131981
QUEENS	124536
MANHATTAN	84460
BRONX	66029
STATEN ISLAND	17165

Toggle details

ZIP.CODE
Real number ($\mathbb{R}_{\geq 0}$)
HIGH CORRELATION

Distinct	195	Minimum	10000
Distinct (%)	< 0.1%	Maximum	11697
Missing	0	Zeros	0
Missing (%)	0.0%	Zeros (%)	0.0%
Infinite	0	Negative	0
Infinite (%)	0.0%	Negative (%)	0.0%
Mean	10871.48385	Memory size	3.2 MiB

Toggle details

Motor Vehicle Collision Dataset - Data Profile Summary:

After generating the summary data report from our Motor Vehicle Collision dataset, we were able to draw some initial insights that Brooklyn had the most accidents out of all the 5 boroughs. The leading cause of NYC accidents is Driver inattention to the road or distracted drivers. And lastly, the car involved in the most accidents are sedans.

Data Profiling Code for 311 Dataset:

In Python:

```
import pandas as pd
import numpy as np
df=pd.read_csv('311 Updated Dataset.csv')
df=df.dropna(subset=['Closed Date','Street Name','Borough','Incident Zip','City'])
df2=df.drop(columns=['Unnamed: 13','Unnamed: 14', 'Unnamed: 15','Resolution Description'])
df2.isnull().sum()
pip install pandas-profiling
import pandas_profiling
from pandas_profiling import ProfileReport
profile = ProfileReport(df2, title='311Data', html={'style':{'full_width':True}})
profile.to_notebook_iframe()
profile.to_file(output_file='311_NYC_Data_Report')
```

Data Profiling Code for Motor Vehicle Collision Dataset:

In R:

```
library(readr)
collision_data <- read.csv("/Users/stevenle/Downloads/Motor_Vehicle_Collisions_Master.csv")
View(collision_data)
str(collision_data)
collision_data[collision_data == ""] <- NA
View(collision_data)
collision_data <- na.omit(collision_data)
View(collision_data)
#EDA Data Profiling
write.csv(collision_data, "/Users/stevenle/Downloads/Motor_Vehicle_Collisions_Data.csv", row.names =
FALSE)
```

In Python:

```
import pandas_profiling
import numpy as np
import pandas as pd

df = pd.read_csv('collision_dataset_update.csv')
data_report = pandas_profiling.ProfileReport(df)
data_report.to_file('Collision_Data_Report.html')
```

Programming Code & Results for Dimensional Models based on our Schema:

From DBT - Agency Dimension

```
{{
  config(
    materialized='table'
  )
}}

SELECT
row_number() OVER () AS AGENCY_ID, Agency, Agency_Name
FROM
( SELECT DISTINCT Agency, Agency_Name
FROM `311_Data.New_Complaint_Table`

)
```

Preview | Compile | Query Results | Compiled SQL | Lineage

6.0 sec —Returned 3 rows.

AGENCY_ID	Agency	Agency_Name
1	NYPD	New York City Police Department
2	NYPD	Traffic Management Center
3	DOT	Department of Transportation

From DBT - Complaint Dimension

```
{{
  config(
    materialized='table'
  )
}}

SELECT
row_number() OVER () AS COMPLAINT_ID, Complaint_Type, Descriptor
FROM
( SELECT DISTINCT Complaint_Type, Descriptor
FROM `311_Data.New_Complaint_Table`

)
```

Preview | Compile | Query Results | Compiled SQL | Lineage

4.5 sec —Returned 72 rows.

COMPLAINT_ID	Complaint_Type	Descriptor
1	Illegal Parking	Blocked Sidewalk
2	Illegal Parking	Blocked Bike Lane
3	Illegal Parking	Blocked Crosswalk
4	Illegal Parking	Parking Permit Improper Use
5	Illegal Parking	Posted Parking Sign Violation
6	Illegal Parking	Double Parked Blocking Traffic

From DBT - Status Dimension

```
{{
  config(
    materialized='table'
  )
}}

SELECT
row_number() OVER () AS STATUS_ID, Status,
FROM
( SELECT DISTINCT Status
FROM `311_Data.New_Complaint_Table`

)
```

Preview | Compile | Query Results | Cor

4.6 sec —Returned 3 rows.

STATUS_ID	Status
1	Closed
2	Pending
3	Open

From DBT - Contributing Factors Dimension

```
{{
  config(
    materialized='table'
  )
}}

SELECT
row_number() OVER () AS contributing_id,
CONTRIBUTING_FACTOR_VEHICLE_1, CONTRIBUTING_FACTOR_VEHICLE_2,
FROM
( SELECT DISTINCT CONTRIBUTING_FACTOR_VEHICLE_1, CONTRIBUTING_FACTOR_VEHICLE_2,
FROM `collision_data.Accident_Table`
)
```

Preview | Compile | Query Results | Compiled SQL | Lineage

5.0 sec —Results limited to 500 rows.

contributing_id	CONTRIBUTING_FACTOR_VEHICLE_1	CONTRIBUTING_FACTOR_VEHICLE_2
1	Unspecified	Unspecified
2	Driver Inattention/Distraction	Unspecified
3	Driver Inexperience	Unspecified
4	Driver Inattention/Distraction	Driver Inattention/Distraction
5	Passing or Lane Usage Improper	Unspecified
6	Failure to Yield Right-of-Way	Unspecified

From DBT - Date Dimension

```
{{
  config(
    materialized='table'
  )
}}

SELECT

 ROW_NUMBER() OVER() as date_dim_id,

 FORMAT_DATE("%Y%m%d",d)  as date_integer,

 d AS full_date
```

Preview | Compile | Query Results | Compiled SQL | Lineage

4.2 sec —Results limited to 500 rows.

date_dim_id	date_integer	full_date	year	year_week	year_day	fiscal_year	fiscal_qtr	month	month_name	week_day
1	20170101	2017-01-…	2017	1	1	2017	1	1	January	0
2	20170102	2017-01-…	2017	1	2	2017	1	1	January	1
3	20170103	2017-01-…	2017	1	3	2017	1	1	January	2
4	20170104	2017-01-…	2017	1	4	2017	1	1	January	3
5	20170105	2017-01-…	2017	1	5	2017	1	1	January	4
6	20170106	2017-01-…	2017	1	6	2017	1	1	January	5

From DBT - Location Dimension

```
{{
  config(
    materialized='table'
  )
}}

SELECT
row_number() OVER () AS location_dim_id, Zip_Code, Borough,
FROM
( SELECT DISTINCT ZIP_CODE, BOROUGH,
FROM `collision_data.Accident_Table`

)
```

Preview | Compile | Query Results | Compiled SQL | Lineage

4.7 sec —Returned 199 rows.

location_dim_id	Zip_Code	Borough
1	10000	MANHATTAN
2	10001	MANHATTAN
3	10002	MANHATTAN
4	10003	MANHATTAN
5	10004	MANHATTAN
6	10005	MANHATTAN

From DBT - Mortality Dimension

```
{{
  config(
    materialized='table'
  )
}}

SELECT
row_number() OVER () AS mortality_dim_id,
NUMBER_OF_PERSONS_INJURED, NUMBER_OF_PERSONS_KILLED,
FROM
( SELECT DISTINCT NUMBER_OF_PERSONS_INJURED,NUMBER_OF_PERSONS_KILLED,
FROM `collision_data.Accident_Table`
)
```

Preview | Compile | Query Results | Compiled SQL | Lineage

4.3 sec —Returned 36 rows.

mortality_dim_id	NUMBER_OF_PERSONS_INJURED	NUMBER_OF_PERSONS_KILLED
1	0	0
2	1	0
3	2	0
4	4	0
5	3	0
6	0	1

From DBT - Mortality Dimension

```
{{
  config(
    materialized='table'
  )
}}

SELECT
row_number() OVER () AS vehicle_id,
VEHICLE_TYPE_CODE_1, VEHICLE_TYPE_CODE_2,
FROM
( SELECT DISTINCT VEHICLE_TYPE_CODE_1, VEHICLE_TYPE_CODE_2,
FROM `collision_data.Accident_Table`
)
```

Preview | Compile | Query Results | Compiled SQL | Lineage

4.8 sec —Results limited to 500 rows.

vehicle_id	VEHICLE_TYPE_CODE_1	VEHICLE_TYPE_CODE_2
1	Sedan	Van
2	Bike	Bike
3	Van	Van
4	Sedan	Sedan
5	Station Wagon/Sport Utility Vehicle	Station Wagon/Sport Utility Vehicle
6	Bus	Sedan

From Big Query - Time Dimension (Upload Time Dimension Spreadsheet)

SCHEMA DETAILS PREVIEW

Row	time_dim_id	fulltime	hours	minutes	seconds	date_from	date_to	version
1	1	00:00:00	0	0	0	1990-01-01	2030-01-01	1
2	2	00:00:01	0	0	1	1990-01-01	2030-01-01	1
3	3	00:00:02	0	0	2	1990-01-01	2030-01-01	1
4	4	00:00:03	0	0	3	1990-01-01	2030-01-01	1
5	5	00:00:04	0	0	4	1990-01-01	2030-01-01	1
6	6	00:00:05	0	0	5	1990-01-01	2030-01-01	1
7	7	00:00:06	0	0	6	1990-01-01	2030-01-01	1
8	8	00:00:07	0	0	7	1990-01-01	2030-01-01	1
9	9	00:00:08	0	0	8	1990-01-01	2030-01-01	1
10	10	00:00:09	0	0	9	1990-01-01	2030-01-01	1
11	11	00:00:10	0	0	10	1990-01-01	2030-01-01	1
12	12	00:00:11	0	0	11	1990-01-01	2030-01-01	1
13	13	00:00:12	0	0	12	1990-01-01	2030-01-01	1
14	14	00:00:13	0	0	13	1990-01-01	2030-01-01	1
15	15	00:00:14	0	0	14	1990-01-01	2030-01-01	1
16	16	00:00:15	0	0	15	1990-01-01	2030-01-01	1
17	17	00:00:16	0	0	16	1990-01-01	2030-01-01	1

From Big Query - Complaint_Fact_Table

RUN SAVE SHARE SCHEDULE MORE — This query will process 61.4 MB when run.

```
SELECT AGENCY_ID, Unique_Key,COMPLAINT_ID, location_dim_id, date_dim_id,
FROM `311_Data.New_Complaint_Table` AS MAIN_TABLE
  INNER JOIN dbt_stevenle999.AGENCY_DIM USING(agency)
  INNER JOIN dbt_stevenle999.COMPLAINT_DIM AS complaint_dim ON COMPLAINT_DIM.complaint_type = MAIN_TABLE.Complaint_Type AND complain
  INNER JOIN dbt_stevenle999.DATE_DIM AS date_dimension ON date_dimension.full_date = EXTRACT(DATE FROM MAIN_TABLE.created_date)
  INNER JOIN dbt_stevenle999.LOCATION_DIM AS location_dim ON location_dim.borough = MAIN_TABLE.borough AND location_dim.ZIP_CODE = M
```

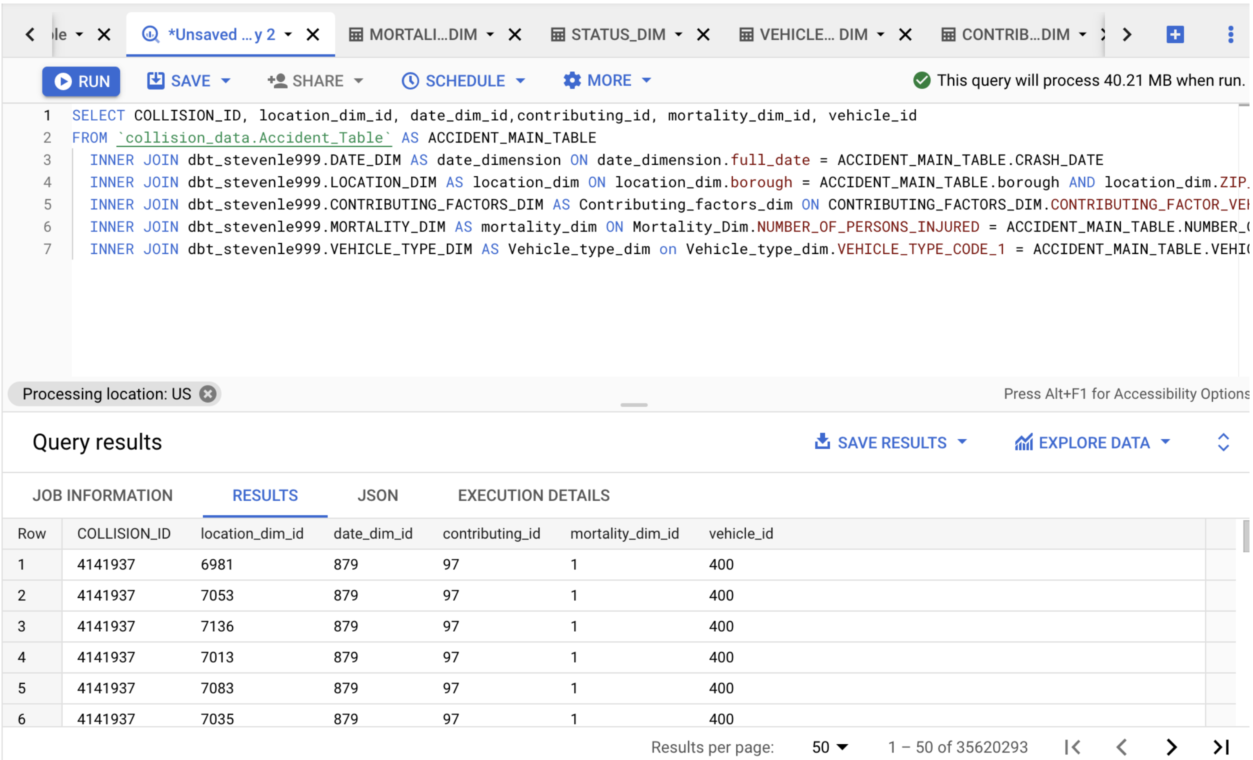
Press Alt+F1 for Accessibility Option

Query results — SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS

Row	AGENCY_ID	Unique_Key	COMPLAINT_ID	location_dim_id	date_dim_id
43	1	43004219	14	5584	899
44	1	43137270	14	5572	908
45	1	52777043	14	5545	1807
46	1	52777043	14	5564	1807
47	1	52828979	14	5552	1812
48	1	47621720	14	5580	1357
49	1	47569263	14	5590	1353
50	1	47817459	14	5552	1377

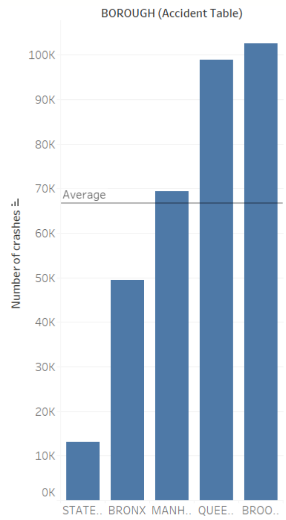
From Big Query - Collision_Fact_Table



Queries for KPI's

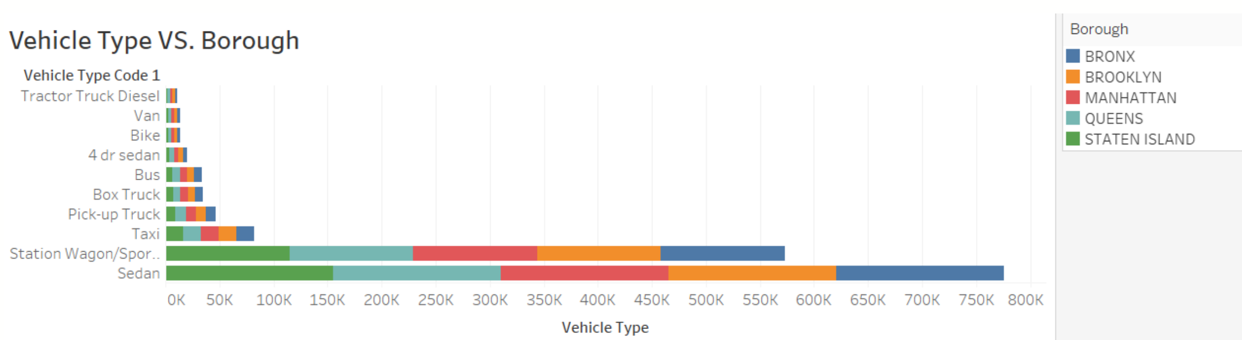
Google Big Query Code:

```
SELECT  BOROUGH, COUNT(COLLISION_ID) AS Number_of_crashes
FROM `cis-4400-group-4.collision_data.Accident_Table`
GROUP BY BOROUGH
ORDER BY Number_of_crashes
```



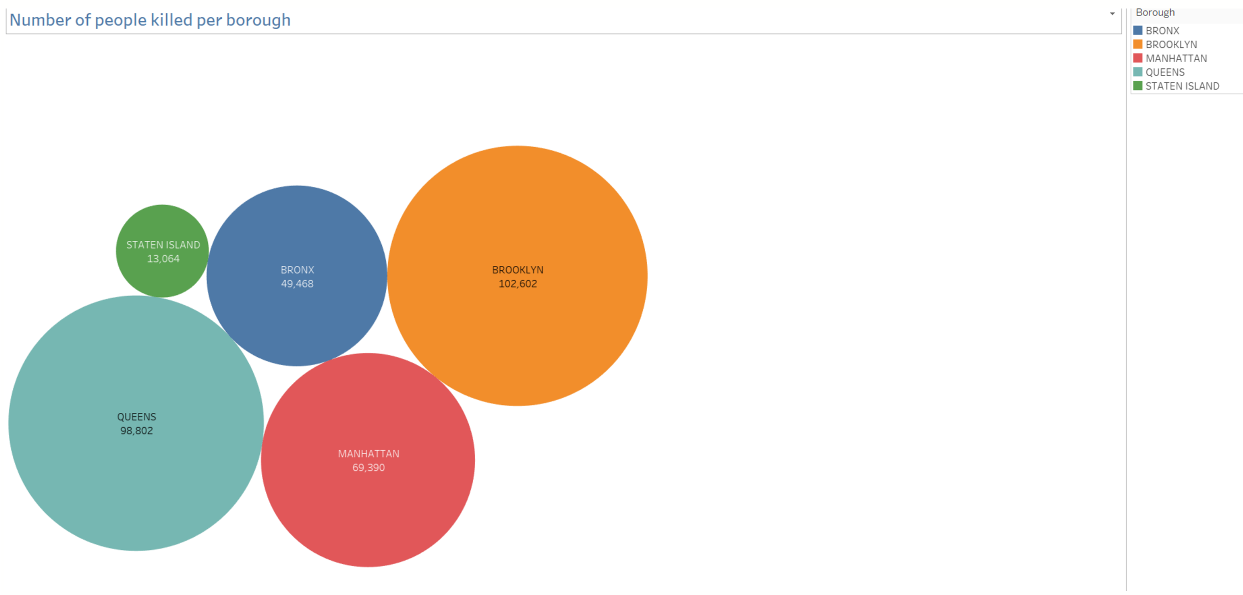
Google Big Query Code:

```
SELECT DISTINCT COUNT(VEHICLE_TYPE_CODE_1) AS Vehicle_Type, VEHICLE_TYPE_CODE_1
FROM `cis-4400-group-4.collision_data.Accident_Table`
GROUP BY VEHICLE_TYPE_CODE_1
ORDER BY Vehicle_Type DESC
LIMIT 10
```



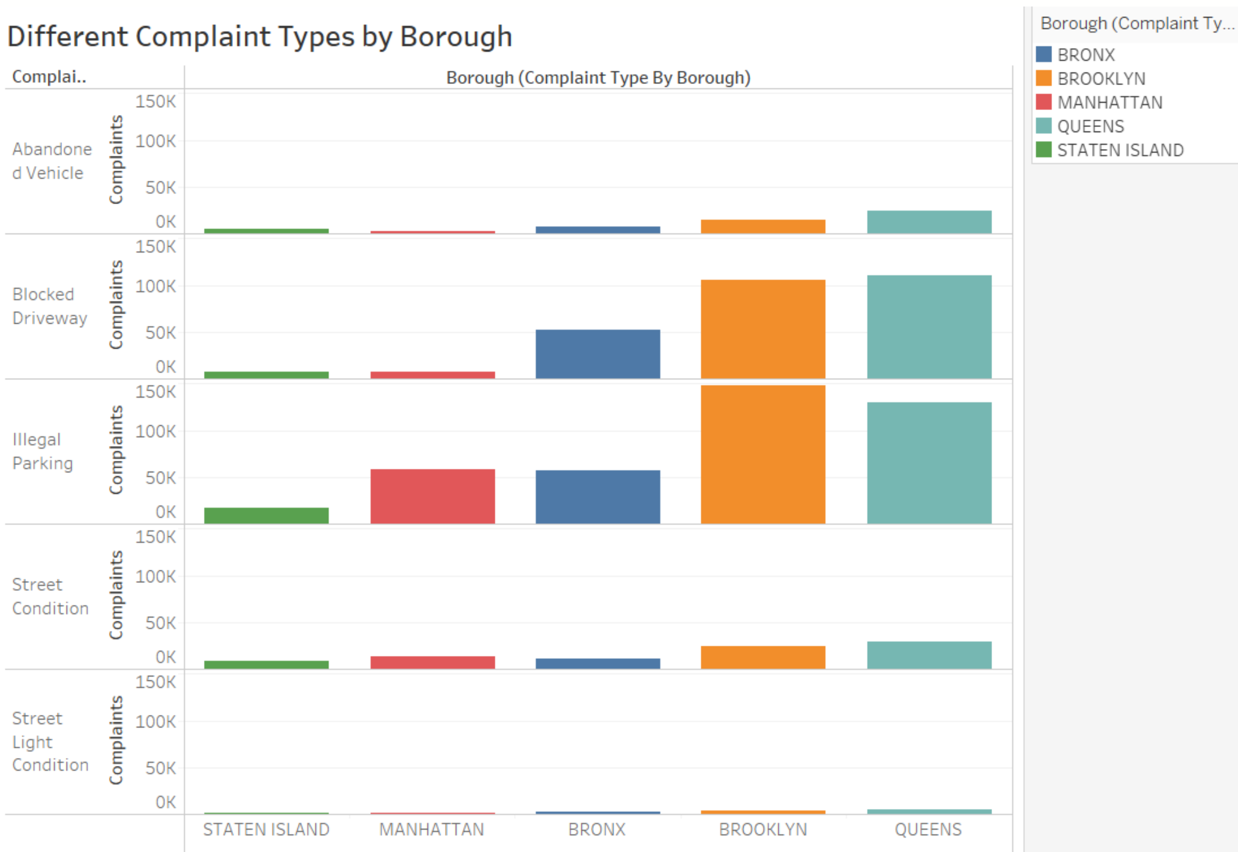
Google Big Query Code:

```
SELECT COUNT('NUMBER_OF_PERSONS_INJURED') AS number_of_persons_injured, BOROUGH
FROM `cis-4400-group-4.collision_data.Accident_Table`
GROUP BY BOROUGH
ORDER BY number_of_persons_injured DESC
LIMIT 10
```



Google Big Query Code:

```
SELECT DISTINCT COUNT(Complaint_Type) AS Complaints, Complaint_Type, Borough
FROM `cis-4400-group-4.311_Data.New_Complaint_Table`
GROUP BY Complaint_Type, Borough
ORDER BY Complaint_Type DESC
```



Conclusion:

Some of the tools we used to help us assist our group project are:

- Jupyter Notebook (We used Python for our Data Profiling)
- RStudio (We used R to help clean our dataset0
- BigQuery (Our DBMS)
- Dbt (To transform our data and create our dimensional models)

Our group met regularly, intending to accomplish our project milestones before or on the deadline. Doing this allowed us to take whatever spare time we had to ask for professional help, feedback, or criticism to improve our project to ensure that there were fewer obstacles in the later milestones. Our group faces several challenges and obstacles throughout the entire group project, especially when creating our dimensional models and fact tables during the ETL process. When we initially tried to develop our dimensional model, we struggled to upload our dataset due to its large size. As a result, we had to clean our dataset using Python and R to slim down the data.

We decided to use dbt as a primary tool to help create our dimensional models during the ETL process. We have followed and executed a tutorial from Homework 3, providing us with valuable insights on managing and analyzing data on cloud services. Using our final database schema as our point of reference, we created 9 dimensional models and two fact tables necessary to execute our queries, allowing us to analyze our KPIs better. We were initially having some issues creating our fact table. However, our solution was to rename our tables to prevent any "name ambiguous error." If we would have to repeat this assignment, we probably withhold from using dbt and only use BigQuery since we created some of our dimensions on the platform. We would also probably split our large dataset by year to upload every single data, allowing us to have a more accurate analysis on our data.

From our dataset, we learned that Brooklyn had the highest number of crashes. Although Sedans took the lead in vehicles being involved in an accident, we found that majority of Sedans were involved in an accident in the Bronx. For the amount of individuals injured from a car accident, Brooklyn took the lead followed by Queens, Manhattan, the Bronx, and lastly Staten Island. From our data profiling, we found that majority of the complaints came from road block or no road access, indicating the road infrastructure is needed to improve to prevent future car crashes. Better driving policy needs to be reinforced to ensure that drivers are paying attention and not distracted, since drivers' inattention is the leading caused in car accident.

Group Meeting Log Sheet

Meeting #1
02/11/2022 8:00PM - 8:52PM
Discussed each of our 311 ideas, possible KPIs, and eliminated topics that do not have a good narrative.
Attendees: Wen Bi, Alan Anthony Fridburg, Steven Le, William D Perez, Lorena Madelin Vasquez

Meeting #2
02/16/2022 6:00-6:20PM
Discussed the optimal choice for our project and shared links to the data set.
Attendees: Wen Bi, Alan Anthony Fridburg, Steven Le, William D Perez, Lorena Madelin Vasquez

Meeting #3
02/24/2022 8:30-9:30PM
Discussed the KPIs and started the dimensional modeling.
Attendees: Wen Bi, Alan Anthony Fridburg, Steven Le, William D Perez, Lorena Madelin Vasquez

Meeting #4
03/2/2022 7:00-9:00 PM
Finalized draft of the dimensional model.
Attendees: Wen Bi, Alan Anthony Fridburg, Steven Le, William D Perez, Lorena Madelin Vasquez

Meeting #4
03/13/2022 8:00-8:20 PM
Finalized The dimensional model.
Attendees: Wen Bi, Alan Anthony Fridburg, Steven Le, William D Perez, Lorena Madelin Vasquez

Meeting #5
4/10/22
ETL / DBMS / Data Profiling
Attendees: Wen Bi, Alan Anthony Fridburg, Steven Le, William D Perez, Lorena Madelin Vasquez

Meeting #6
4/12/22

ETL / DBMS / Data Profiling
Attendees: Wen Bi, Alan Anthony Fridburg, Steven Le, William D Perez, Lorena Madelin Vasquez
Meeting #6
04/20/2022
ETL / DBMS / Data Profiling
Attendees: Wen Bi, Alan Anthony Fridburg, Steven Le, William D Perez, Lorena Madelin Vasquez

Meeting #7
04/29/2022
ETL / DBMS / Data Profiling
Attendees: Wen Bi, Alan Anthony Fridburg, Steven Le, William D Perez, Lorena Madelin Vasquez

Meeting #8
05/04/2022
ETL / DBMS / Data Profiling.
Attendees: Alan Anthony Fridburg, Steven Le, Lorena Madelin Vasquez

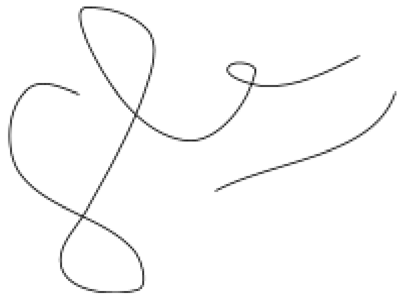
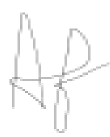
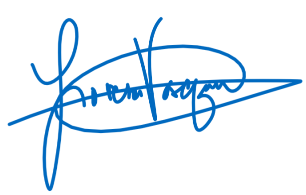
Meeting #9
05/07/2022 7:00-9:00 PM
Programming
Attendees: Alan Anthony Fridburg, Steven Le, Lorena Madelin Vasquez

Meeting #10
05/11/2022 8:00-8:20 PM
Programming
Attendees: Alan Anthony Fridburg, Steven Le, Lorena Madelin Vasquez

Meeting #11
5/15/22
Programming
Attendees: Alan Anthony Fridburg, Steven Le, Lorena Madelin Vasquez

Meeting #12
5/19/22
Finalize Programming and Dimensional Model on Dbt and BigQuery
Attendees: Alan Anthony Fridburg, Steven Le, Lorena Madelin Vasquez

Performance Appraisal & Sign-off

Team Member Name(print)	Signature	Weekly Contribution
Steven Le		26.67%
Wen Bi		10%
Alan Anthony Fridburg		26.67%
William D Perez	William Perez	10%
Lorena Madelin Vasquez		26.67%