**Project ETL - Extract, Transform, Load**

**Team members**

Swati Swain

Emi Rivera

Nicholas Prentkowski

Terrence Cummings

**Minneapolis Police Incidents, Use of Force, and Demographics**

Project Objective:

To create a database that will facilitate an analysis of the relationship between police incidents, the use of force by police in those incidents, and the demographic characteristics of the neighborhood in which the incidents occurred such as race and income.

The team will extract data from various sources, clean and normalize the data, and load the data into a database. Such data can then be used by applications to allows analysis of relationships and trends between these factors.

The target data, for the city of Minneapolis, includes:

1. Police incident data by neighborhood
2. Police “use of force” data by neighborhood
3. Income levels by neighborhood
4. Race distribution by neighborhood

Please find following Table of Contents directing to ERD, SQL, and detailed ETL flow for each dataset.

Table of Contents

[Entity Relationship Diagram (ERD) 3](#_Toc42235479)

[SQL for PostgreSQL tables 4](#_Toc42235480)

[Police Incidents 9](#_Toc42235481)

[Description 9](#_Toc42235482)

[Data sources 9](#_Toc42235483)

[Data extraction 9](#_Toc42235484)

[Data transformation 9](#_Toc42235485)

[Data loading 9](#_Toc42235486)

[Police Use of Force 10](#_Toc42235487)

[Description 10](#_Toc42235488)

[Data sources 10](#_Toc42235489)

[Data extraction 10](#_Toc42235490)

[Data transformation 10](#_Toc42235491)

[Data loading 10](#_Toc42235492)

[Neighborhood Race Demographics 11](#_Toc42235493)

[Description 11](#_Toc42235494)

[Data sources 11](#_Toc42235495)

[Data extraction 11](#_Toc42235496)

[Data transformation 11](#_Toc42235497)

[Data loading 12](#_Toc42235498)

[Neighborhood Income Demographics 13](#_Toc42235499)

[Description 13](#_Toc42235500)

[Data sources 13](#_Toc42235501)

[Data extraction 13](#_Toc42235502)

[Data transformation 13](#_Toc42235503)

[Data loading 13](#_Toc42235504)

[Web Application 14](#_Toc42235505)

# Entity Relationship Diagram (ERD)

A screenshot of a cell phone

Description automatically generated

# SQL for PostgreSQL tables

|  |
| --- |
| --COMMUNITY |
|  | DROP TABLE IF EXISTS COMMUNITY CASCADE; |
|  | CREATE TABLE COMMUNITY ( |
|  | community\_id INT NOT NULL, |
|  | name VARCHAR(50) NOT NULL, |
|  | CONSTRAINT pk\_COMMUNITY PRIMARY KEY ( |
|  | community\_id |
|  | ), |
|  | CONSTRAINT uc\_COMMUNITY\_name UNIQUE ( |
|  | name |
|  | ) |
|  | ); |
|  |  |
|  | --NEIGHBORHOOD |
|  | DROP TABLE IF EXISTS NEIGHBORHOOD CASCADE; |
|  | CREATE TABLE NEIGHBORHOOD ( |
|  | neighborhood\_id INT NOT NULL, |
|  | name VARCHAR(50) NOT NULL, |
|  | community\_id INT NOT NULL, |
|  | CONSTRAINT pk\_NEIGHBORHOOD PRIMARY KEY ( |
|  | neighborhood\_id |
|  | ), |
|  | CONSTRAINT uc\_NEIGHBORHOOD\_name UNIQUE ( |
|  | name |
|  | ) |
|  |  |
|  | ); |
|  |  |
|  | DROP TABLE IF EXISTS HOUSEHOLD\_INCOME\_BY\_NEIGHBORHOOD CASCADE; |
|  | CREATE TABLE HOUSEHOLD\_INCOME\_BY\_NEIGHBORHOOD ( |
|  | household\_income\_by\_neighborhood\_id INT NOT NULL, |
|  | neighborhood\_id INT NOT NULL, |
|  | IncomeLess35000\_count INT DEFAULT NULL, |
|  | IncomeLess35000\_percent DECIMAL(10,2) DEFAULT NULL, |
|  | IncomeLess35to49\_count INT DEFAULT NULL, |
|  | IncomeLess35to49\_percent DECIMAL(10,2) DEFAULT NULL, |
|  | IncomeLess50to74\_count INT DEFAULT NULL, |
|  | IncomeLess50to74\_percent DECIMAL(10,2) DEFAULT NULL, |
|  | IncomeLess75to99\_count INT DEFAULT NULL, |
|  | IncomeLess75to99\_percent DECIMAL(10,2) DEFAULT NULL, |
|  | Income100Plus\_count INT DEFAULT NULL, |
|  | Income100Plus\_percent DECIMAL(10,2) DEFAULT NULL, |
|  | Median\_Income\_Total DECIMAL(10,2) DEFAULT NULL, |
|  | CONSTRAINT pk\_HOUSEHOLD\_INCOME\_BY\_NEIGHBORHOOD PRIMARY KEY ( |
|  | household\_income\_by\_neighborhood\_id |
|  | ), |
|  | CONSTRAINT uc\_HOUSEHOLD\_INCOME\_BY\_NEIGHBORHOOD\_neighborhood\_id UNIQUE ( |
|  | neighborhood\_id |
|  | ) |
|  | ); |
|  |  |
|  | DROP TABLE IF EXISTS HOUSEHOLD\_INCOME\_BY\_COMMUNITY CASCADE; |
|  | CREATE TABLE HOUSEHOLD\_INCOME\_BY\_COMMUNITY ( |
|  | household\_income\_by\_community\_id INT NOT NULL, |
|  | community\_id INT NOT NULL, |
|  | IncomeLess35000\_count INT DEFAULT NULL, |
|  | IncomeLess35000\_percent DECIMAL(10,2) DEFAULT NULL, |
|  | IncomeLess35to49\_count INT DEFAULT NULL, |
|  | IncomeLess35to49\_percent DECIMAL(10,2) DEFAULT NULL, |
|  | IncomeLess50to74\_count INT DEFAULT NULL, |
|  | IncomeLess50to74\_percent DECIMAL(10,2) DEFAULT NULL, |
|  | IncomeLess75to99\_count INT DEFAULT NULL, |
|  | IncomeLess75to99\_percent DECIMAL(10,2) DEFAULT NULL, |
|  | Income100Plus\_count INT DEFAULT NULL, |
|  | Income100Plus\_percent DECIMAL(10,2) DEFAULT NULL, |
|  | Median\_Income\_Total DECIMAL(10,2) DEFAULT NULL, |
|  | CONSTRAINT pk\_HOUSEHOLD\_INCOME\_BY\_COMMUNITY PRIMARY KEY ( |
|  | household\_income\_by\_community\_id |
|  | ), |
|  | CONSTRAINT uc\_HOUSEHOLD\_INCOME\_BY\_COMMUNITY\_community\_id UNIQUE ( |
|  | community\_id |
|  | ) |
|  | ); |
|  |  |
|  | DROP TABLE IF EXISTS RACE\_BY\_NEIGHBORHOOD CASCADE; |
|  | CREATE TABLE RACE\_BY\_NEIGHBORHOOD ( |
|  | race\_by\_neighborhood\_id SERIAL NOT NULL, |
|  | neighborhood\_id INT NOT NULL, |
|  | total\_cnt DECIMAL(10,2) DEFAULT NULL, |
|  | white\_pct DECIMAL(10,5) DEFAULT NULL, |
|  | black\_pct DECIMAL(10,5) DEFAULT NULL, |
|  | native\_pct DECIMAL(10,5) DEFAULT NULL, |
|  | asian\_pct DECIMAL(10,5) DEFAULT NULL, |
|  | other\_pct DECIMAL(10,5) DEFAULT NULL, |
|  | two\_or\_more\_pct DECIMAL(10,5) DEFAULT NULL, |
|  | hispanic\_pct DECIMAL(10,5) DEFAULT NULL, |
|  | of\_color\_pct DECIMAL(10,5) DEFAULT NULL, |
|  | CONSTRAINT pk\_RACE\_BY\_NEIGHBORHOOD PRIMARY KEY ( |
|  | race\_by\_neighborhood\_id |
|  | ), |
|  | CONSTRAINT uc\_RACE\_BY\_NEIGHBORHOOD\_neighborhood\_id UNIQUE ( |
|  | neighborhood\_id |
|  | ) |
|  | ); |
|  |  |
|  | DROP TABLE IF EXISTS RACE\_BY\_COMMUNITY CASCADE; |
|  | CREATE TABLE RACE\_BY\_COMMUNITY ( |
|  | race\_by\_community\_id SERIAL NOT NULL, |
|  | community\_id INT NOT NULL, |
|  | total\_cnt DECIMAL(10,2) DEFAULT NULL, |
|  | white\_pct DECIMAL(10,5) DEFAULT NULL, |
|  | black\_pct DECIMAL(10,5) DEFAULT NULL, |
|  | native\_pct DECIMAL(10,5) DEFAULT NULL, |
|  | asian\_pct DECIMAL(10,5) DEFAULT NULL, |
|  | other\_pct DECIMAL(10,5) DEFAULT NULL, |
|  | two\_or\_more\_pct DECIMAL(10,5) DEFAULT NULL, |
|  | hispanic\_pct DECIMAL(10,5) DEFAULT NULL, |
|  | of\_color\_pct DECIMAL(10,5) DEFAULT NULL, |
|  | CONSTRAINT pk\_RACE\_BY\_COMMUNITY PRIMARY KEY ( |
|  | race\_by\_community\_id |
|  | ), |
|  | CONSTRAINT uc\_RACE\_BY\_COMMUNITY\_community\_id UNIQUE ( |
|  | community\_id |
|  | ) |
|  | ); |
|  |  |
|  | DROP TABLE IF EXISTS POLICE\_USE\_OF\_FORCE CASCADE; |
|  | CREATE TABLE POLICE\_USE\_OF\_FORCE ( |
|  | police\_use\_of\_force\_id SERIAL NOT NULL, |
|  | response\_date TIMESTAMP DEFAULT NULL, |
|  | case\_number VARCHAR(50) DEFAULT NULL, |
|  | problem VARCHAR(100) DEFAULT NULL, |
|  | subject\_race VARCHAR(50) DEFAULT NULL, |
|  | subject\_sex VARCHAR(50) DEFAULT NULL, |
|  | subject\_age INT DEFAULT NULL, |
|  | subject\_role VARCHAR(50) DEFAULT NULL, |
|  | primary\_offense VARCHAR(50) DEFAULT NULL, |
|  | type\_of\_resistance VARCHAR(50) DEFAULT NULL, |
|  | police\_use\_of\_force\_type VARCHAR(50) DEFAULT NULL, |
|  | force\_type\_action VARCHAR(50) DEFAULT NULL, |
|  | subject\_injury VARCHAR(50) DEFAULT NULL, |
|  | neighborhood\_id INT DEFAULT NULL, |
|  | neighborhood VARCHAR(50) DEFAULT NULL, |
|  | precinct VARCHAR(10) DEFAULT NULL, |
|  | CONSTRAINT pk\_POLICE\_USE\_OF\_FORCE PRIMARY KEY ( |
|  | police\_use\_of\_force\_id |
|  | ), |
|  | CONSTRAINT uc\_POLICE\_USE\_OF\_FORCE\_case\_number UNIQUE ( |
|  | case\_number |
|  | ) |
|  | ); |
|  |  |
|  | DROP TABLE IF EXISTS POLICE\_INCIDENT CASCADE; |
|  | CREATE TABLE POLICE\_INCIDENT ( |
|  | police\_incident\_id SERIAL NOT NULL, |
|  | casenumber VARCHAR(50) DEFAULT NULL, |
|  | reporteddate TIMESTAMP DEFAULT NULL, |
|  | offense VARCHAR(100) DEFAULT NULL, |
|  | neighborhood VARCHAR(100) DEFAULT NULL, |
|  | incident\_id INT DEFAULT NULL, |
|  | neighborhood\_id INT DEFAULT NULL, |
|  | community\_id INT DEFAULT NULL, |
|  | CONSTRAINT pk\_POLICE\_INCIDENT PRIMARY KEY ( |
|  | police\_incident\_id |
|  | ), |
|  | CONSTRAINT uc\_POLICE\_INCIDENT\_incident\_id UNIQUE ( |
|  | incident\_id |
|  | ) |
|  | ); |
|  |  |
|  |  |
|  | --Foreign Keys |
|  | ALTER TABLE NEIGHBORHOOD ADD CONSTRAINT fk\_NEIGHBORHOOD\_community\_id FOREIGN KEY(community\_id) |
|  | REFERENCES COMMUNITY (community\_id); |
|  |  |
|  |  |
|  | ALTER TABLE HOUSEHOLD\_INCOME\_BY\_NEIGHBORHOOD ADD CONSTRAINT fk\_HOUSEHOLD\_INCOME\_BY\_NEIGHBORHOOD\_neighborhood\_id FOREIGN KEY(neighborhood\_id) |
|  | REFERENCES NEIGHBORHOOD (neighborhood\_id); |
|  |  |
|  | ALTER TABLE HOUSEHOLD\_INCOME\_BY\_COMMUNITY ADD CONSTRAINT fk\_HOUSEHOLD\_INCOME\_BY\_COMMUNITY\_community\_id FOREIGN KEY(community\_id) |
|  | REFERENCES COMMUNITY (community\_id); |
|  |  |
|  | ALTER TABLE HOUSEHOLD\_INCOME\_BY\_NEIGHBORHOOD ADD CONSTRAINT fk\_HOUSEHOLD\_INCOME\_BY\_NEIGHBORHOOD\_neighborhood\_id FOREIGN KEY(neighborhood\_id) |
|  | REFERENCES NEIGHBORHOOD (neighborhood\_id); |
|  |  |
|  | ALTER TABLE HOUSEHOLD\_INCOME\_BY\_COMMUNITY ADD CONSTRAINT fk\_HOUSEHOLD\_INCOME\_BY\_COMMUNITY\_community\_id FOREIGN KEY(community\_id) |
|  | REFERENCES COMMUNITY (community\_id); |
|  |  |
|  | ALTER TABLE RACE\_BY\_COMMUNITY ADD CONSTRAINT fk\_RACE\_BY\_COMMUNITY\_community\_id FOREIGN KEY(community\_id) |
|  | REFERENCES COMMUNITY (community\_id); |
|  |  |
|  | ALTER TABLE POLICE\_USE\_OF\_FORCE ADD CONSTRAINT fk\_POLICE\_USE\_OF\_FORCE\_neighborhood\_id FOREIGN KEY(neighborhood\_id) |
|  | REFERENCES NEIGHBORHOOD (neighborhood\_id); |
|  |  |
|  | ALTER TABLE POLICE\_INCIDENT ADD CONSTRAINT fk\_POLICE\_INCIDENT\_neighborhood\_id FOREIGN KEY(neighborhood\_id) |
|  | REFERENCES NEIGHBORHOOD (neighborhood\_id); |
|  |  |
|  | ALTER TABLE POLICE\_INCIDENT ADD CONSTRAINT fk\_POLICE\_INCIDENT\_community\_id FOREIGN KEY(community\_id) |
|  | REFERENCES COMMUNITY (community\_id); |

# Police Incidents

## Description

## Data sources

## Data extraction

## Data transformation

## Data loading

# Police Use of Force

## Description

The objective is to obtain data regarding the incidents during which Minneapolis police officers deemed the use of force necessary.

## Data sources

The primary source utilized to meet the specified objective was a data frame found on the webpage titled, *Police Use of Force*, which can be found on the website titled, Open Minneapolis. The page can be found via the following hyperlink: <http://opendata.minneapolismn.gov/datasets/police-use-of-force/data?geometry=-103.617%2C-5.468%2C10.289%2C48.789&orderBy=ResponseDate&orderByAsc=false>

## 

## Data extraction

The source data was extracted via a comma-separated values file (.CSV) that was first downloaded locally and finally pushed onto our project team’s repository. It is specifically stored in the folder titled, “source\_file.”

## 

## Data transformation

Transformation (cleaning) involved the following steps:

1. Declaring and assigning a variable to the CSV…
2. Reading in the CSV by using the read\_csv function, which will produce and store a Pandas data frame…
3. Dropping unessential data fields…
4. Renaming remaining fields to match entity-relationship diagram (ERD)…
5. Using .dtypes code to determine type of value held in the response\_date field…
6. Using astype function and Numpy to convert response\_date field to datetime64…
7. Declaring and assigning a variable to the MLS\_Neighborhoods CSV…
8. Changing the field name titled: name, in new data frame to, neighborhood…
9. Using the replace function to match the spelling and punctuation of the ten neighborhoods that conflict with one another when trying to merge the two data frames…
10. Merging the two data frames on the field, neighborhood and via a left join…
11. Use double brackets to rearrange the order of the fields of data frame to match the ERD…
12. Use .dtypes to check, or refer to the last time it was used to see what type of values can be found in the subject\_age field…
13. Change all NaN(s) within subject\_age field to 0 via fillna, which will allow for conversion to int64…
14. Convert subject\_age field to int64 via astype function…
15. Change all NaN(s) within neighborhood\_id field to 0 via fillna…
16. Convert values in neighborhood\_id from float64 to int64 via .astype…
17. Rename final data frame to something more concise and clearer…
18. Export as CSV to the folder, target\_files…

## Data loading

The steps are as follows:

1. Create tables in PostgreSQL using the SQL script based on the ERD presented at the beginning of this document.
2. Use Sqlalchemy (from sqlalchemy import create\_engine) to connect to PostgreSQL database.
3. Use Pandas df.to\_sql to populate PostgreSQL tables with Pandas dataframe values.

# Neighborhood Race Demographics

## Description

The objective is to obtain data regarding the racial mix of Minneapolis neighborhoods and communities.

## Data sources

Data is obtained from MINNESOTA COMPASS (mncompass.org). We need to scrape data from the following endpoints:

1. Scrape links to Minneapolis neighborhood-specific webpage on mncompass.org found on:
   1. <http://www.mncompass.org/profiles/neighborhoods/minneapolis-saint-paul>
2. Scrape links to Minneapolis community-specific webpage on mncompass.org found on:
   1. <http://www.mncompass.org/profiles/neighborhoods/minneapolis-saint-paul>
3. Scrape race data for each Minneapolis neighborhood links obtained in step 1. For example, Armatage neighborhood at:
   1. <http://www.mncompass.org/profiles/neighborhoods/minneapolis/armatage>
4. Scrape race data for each Minneapolis community links obtained in step 2. For example, Camden at:
   1. <http://www.mncompass.org/profiles/communities/minneapolis/camden>

## Data extraction

Selenium webdriver (from selenium import webdriver) was used to scrape data at the URL. This is because the data is populated by Javascript and therefore not accessible by Splinter.

Extraction followed the following process:

1. Scrape the individual neighborhood and community links and store in lists of URLs.
2. Send the webdriver to each link in the lists and scrape the race data from each page.

The neighborhood and community race data is then stored in a Pandas dataframe and written to csv files.

## Data transformation

Transformation (cleaning) involved the following steps:

1. Read in the csv’s from extraction as Pandas dataframes.
2. The scraped data contained the word ‘suppressed’ in some table cells. Replace this with NaN so all missing data is represented by NaN.
3. Convert text-styled numbers into numeric type.
4. Add a ‘total’ column as the sum of the individual race columns.
5. Use pd.merge to bring in neighborhood and community ID’s that will be used in PostgreSQL keys.
6. Delete extraneous columns.
7. Reorder columns for presentability.

## Data loading

Steps:

1. Create tables in PostgreSQL using the SQL script based on the ERD presented at the beginning of this document.
2. Use Sqlalchemy (from sqlalchemy import create\_engine) to connect to PostgreSQL database.
3. Use Pandas df.to\_sql to populate PostgreSQL tables with Pandas dataframe values.

# Neighborhood Income Demographics

## Description

## Data sources

## Data extraction

## Data transformation

## Data loading

# Web Application