

Lab 2 - Data Wrangling

- Due Nov 21, 2024 by 11:59pm
- Points 100
- Submitting a file upload

Assignment Overview:

In this lab, you will collect, integrate, clean, and load data from four data sources to create a data warehouse used by analysts at a real estate investment firm. These analysts use the data warehouse to understand market trends and price estimates for real estate parcels in New York City.

Update and clarification log:

- Monday, Nov 18, 2024 - Added Block and Lot columns to the header of the required format for the TreeCensus.csv file. You should derive the Block# and Lot# from the bbl field in the source data. "BBL" stands for Borough, Block, Lot, and it uses the same encoding as the ParcelID field in the PropertySale table.
- Wednesday, Nov 20, 2024
 - Changed the header format for TreeCensus.csv for NTA column - from NeighborhoodTabulationArea to NTA_name, to match the data warehouse field name. You can use either of these column names for the TreeCensus.csv output file for the lab submission.
 - Removed the ProblemDescription field from the TreeCensus.csv output file header because the Data Warehouse TreeCensus table does not include this field.
 - Noted that the TreeCensus.Latitutde column in the data warehouse has a spelling error. Noted that you can use either the misspelled name in your TreeCensus.csv output file, or you can correctly spell "Latitude" in the header for that file and map to the data warehouse filename manually when importing the .csv file. No points will be deducted for using either option.

Source data:

Source Dataset 1: Your first data source is a Microsoft Excel workbook containing a collection of every real estate transaction recorded in New York City from 2019 through 2023 (inclusive):

[NYC Real Estate Combined Dataset 2019-2023 v1.0.xlsx](https://canvas.cmu.edu/courses/42925/files/11850105?wrap=1) (<https://canvas.cmu.edu/courses/42925/files/11850105?wrap=1>)
 ↓ (https://canvas.cmu.edu/courses/42925/files/11850105/download?download_frd=1)

This dataset is derived from public real estate transaction records published by the New York City government at

<http://www1.nyc.gov/site/finance/taxes/property-annualized-sales-update.page>
 (<http://www1.nyc.gov/site/finance/taxes/property-annualized-sales-update.page>)

To better understand what this file contains, review the information available at this site, especially the descriptions and details about the data at the following two links:

- [Glossary and Excel File Use Information](http://www1.nyc.gov/assets/finance/downloads/pdf/07pdf/glossary_rsf071607.pdf)
 (http://www1.nyc.gov/assets/finance/downloads/pdf/07pdf/glossary_rsf071607.pdf)
- [NYC Building Class Code Descriptions](http://www1.nyc.gov/assets/finance/jump/hlpbldgcode.html) (<http://www1.nyc.gov/assets/finance/jump/hlpbldgcode.html>)

This five-year data set will give the investment firm a strong baseline for analysis while still being a manageable size for data wrangling and loading into a relational database.

To explore property transactions further and get additional detail about individual properties, you may **optionally** use the interactive, map-based property query tool provided by the NYC government at

<https://propertyinformationportal.nyc.gov/parcels/> ↗ (<https://propertyinformationportal.nyc.gov/parcels/>) to explore and understand where different Blocks, Lots, and individual Parcels are located. This data source also includes additional data about

each block/lot/parcel. Use of this tool is completely optional. You should not need to use it to complete the lab, but it may be helpful if you need to look up specific property information or better understand how the transactional records relate to how data is managed in the city's geographic information systems (GIS's).

Source Dataset 2: The second data source is much smaller and much more slowly changing. The data warehouse contains a table called BuildingClass, which is a lookup table mapping NYC Building Class Codes to descriptions of those codes. You need to load this lookup table with all of the building class codes used in New York City and their descriptions. New York City's website has a page with that mapping - [NYC Building Class Code Descriptions](http://www1.nyc.gov/assets/finance/jump/hlpbldgcode.html) (<http://www1.nyc.gov/assets/finance/jump/hlpbldgcode.html>). Do some screen-scraping to generate a .csv file that you can load into the data warehouse. I suggest simply copying and pasting from the web into MS Excel, then cleaning the HTML data by hand and saving the table as a simple .csv file for import into your target SQLite database.

Source Dataset 3: The third dataset you will use is tax assessment data. This data has already been loaded into the data warehouse for the 2025 tax year. As discussed in the lab details section below, you will use the source data already in the data warehouse's TaxAssessment table to create two *materialized view* tables summarizing information about property owners and city blocks based on individual tax assessments. The [NYC Property Assessment Data Dictionary.xlsx](https://canvas.cmu.edu/courses/42925/files/11850103?wrap=1) (<https://canvas.cmu.edu/courses/42925/files/11850103?wrap=1>). [↓](https://canvas.cmu.edu/courses/42925/files/11850103/download?download_frd=1) (https://canvas.cmu.edu/courses/42925/files/11850103/download?download_frd=1) should help you understand the TaxAssessment table's contents. In addition to using the data dictionary, you are strongly encouraged to explore and profile the existing source dataset and to use external resources to understand precisely what each row and column represents.

Source Dataset 4: The fourth source dataset differs somewhat from the other two. Every ten years (in years ending in '5'), New York City sponsors a volunteer census effort to identify every "street tree" in the city. A "street tree" grows next to a street that the city maintains. This thin strip of property is considered 'right of way', and the city is responsible for planting and maintaining trees planted in these locations. Trees planted on private property away from the right of way (such as in somebody's garden or backyard, or in public parks) are not considered 'street trees' and are thus not included in this dataset. One of the partners at the real estate investment firm has a hypothesis that neighborhoods and blocks with a higher concentration of healthy street trees are more appealing, likely better maintained by the city and the people who live there, and should, in general, command a premium in the real estate market. To test this hypothesis, you need to retrieve the data from the 2015 Tree Census from the NYC website and figure out how to clean, reformat, and otherwise wrangle the tree census data provided on this website into the TreeCensus table of the data warehouse. Further details for doing so are provided in the lab details section below.

Tree census references:

- New York City Open Data website page for 2015 tree census (download source data from here):
 - https://data.cityofnewyork.us/Environment/2015-Street-Tree-Census-Tree-Data/uvpi-gqnh/about_data [↗](https://data.cityofnewyork.us/Environment/2015-Street-Tree-Census-Tree-Data/uvpi-gqnh/about_data)
- Data dictionary for tree census source data (link available on NYC Open Data website page referenced above):
 - <https://data.cityofnewyork.us/api/views/uvpi-gqnh/files/8705bfd6-993c-40c5-8620-0c81191c7e25?download=true&filename=StreetTreeCensus2015TreesDataDictionary20161102.pdf> [↗](https://data.cityofnewyork.us/api/views/uvpi-gqnh/files/8705bfd6-993c-40c5-8620-0c81191c7e25?download=true&filename=StreetTreeCensus2015TreesDataDictionary20161102.pdf)
- Data dictionary for investment firm data warehouse (destination table):
 - [Street Tree Census Data Dictionary \(Data Warehouse - destination\) v1.1.xlsx](https://canvas.cmu.edu/courses/42925/files/11952762?wrap=1) [↓](https://canvas.cmu.edu/courses/42925/files/11952762?wrap=1)
 - https://canvas.cmu.edu/courses/42925/files/11952762/download?download_frd=1
- NYC CityParks website dedicated to the tree census (background information, not a raw data source)
 - <https://www.nycgovparks.org/trees/treescount> [↗](https://www.nycgovparks.org/trees/treescount)

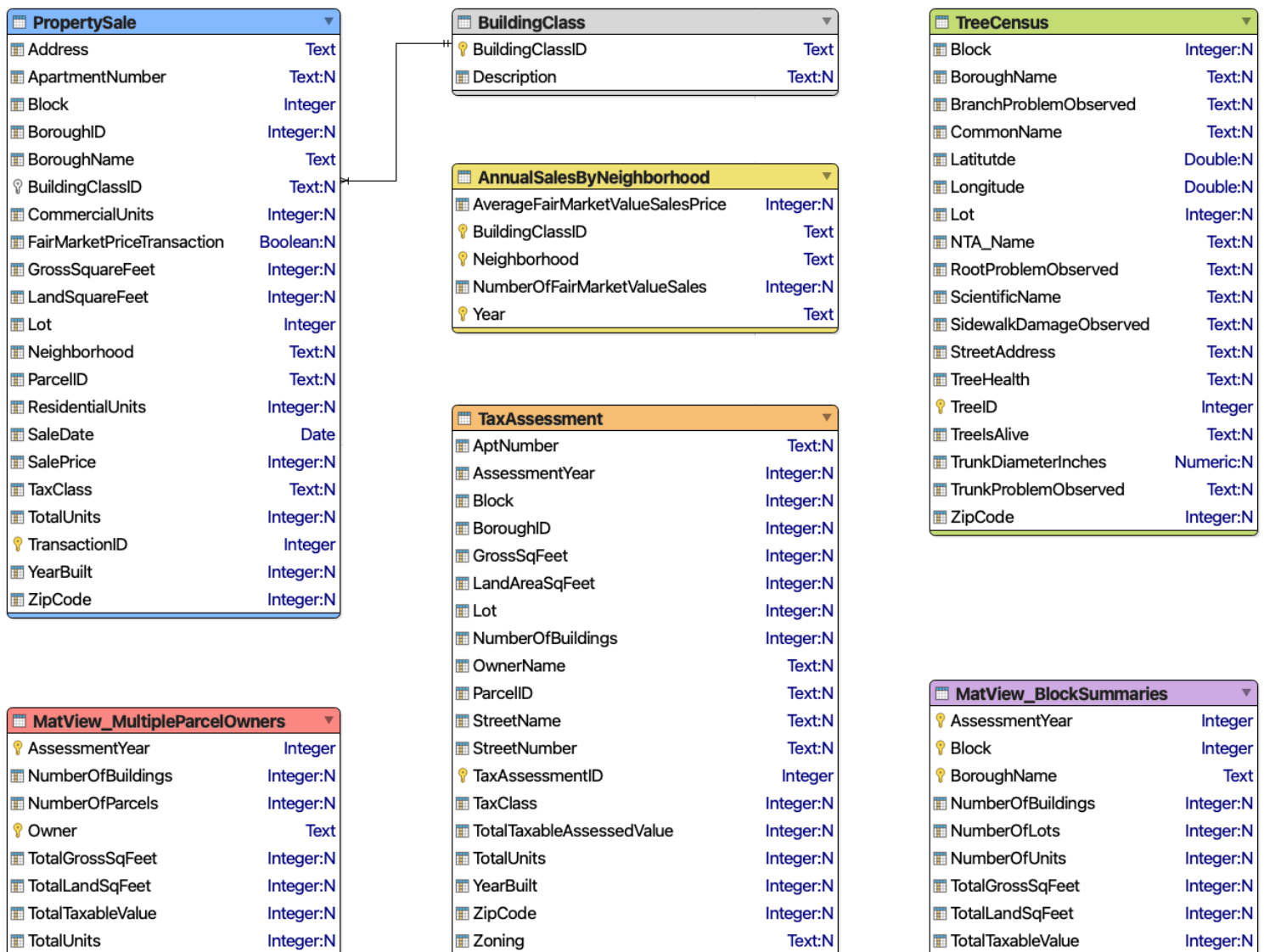
Data Warehouse Table Diagram:

As mentioned above, the investment firm you are working for has a data warehouse that it uses for various analytic purposes. You need to clean, integrate, and shape/transform the specified source data using Tableau Data Prep and/or SQL as specified in the table-by-table instructions that follow and then load it into the company's data warehouse. The provided data warehouse has six empty tables you need to fill with appropriately cleaned and wrangled data. The seventh table (TaxAssessment) is already loaded with data about the assessed value for every piece of real estate in New York City. Further details on what you are expected to do with each table are provided in the following sections.

MDM M2-24 - Lab 2 NYC Real Estate Data Warehouse (v1_0).db (<https://canvas.cmu.edu/courses/42925/files/11855553?wrap=1>) [↓ \(https://canvas.cmu.edu/courses/42925/files/11855553/download?download_frd=1\)](https://canvas.cmu.edu/courses/42925/files/11855553/download?download_frd=1)

Assume that the structure of the company's data warehouse is fixed and that you may not modify its structure. As you will often find in corporate data management situations, the data warehouse is shared by multiple applications used throughout the organization and, thus, not something readily modified to simplify data upload. You typically must adapt your incoming data to match the target data warehouse structure rather than vice-versa. Do so here.

Database table diagram for target SQLite database:



Required data cleaning and shaping steps:

You must complete the following tasks to integrate, clean, shape, and load the requested data into the target database structure. Instructions are given for each of the tables in the target database.

BuildingClass table: To fill the BuildingClass table, you need to retrieve and wrangle the data contained on the website [NYC Building Class Code Descriptions \(http://www1.nyc.gov/assets/finance/jump/hlpbldgcode.html\)](http://www1.nyc.gov/assets/finance/jump/hlpbldgcode.html) to create a file named BuildingClass.csv that has the following structure - exactly these heading names, in exactly this order:

BuildingClassID	Description
-----------------	-------------

Once you have that data in the BuildingClass.csv file, import the data from the file into your target database.

PropertySale table: You will find that the property transactions dataset is far from perfect. The data is scattered across multiple Excel worksheets. There are records with missing data, values that seem wildly incorrect, and values that contradict each other. Figuring out how to integrate this data from multiple source worksheets, format it, clean it, deal with missing or seemingly incorrect data, etc., in a reasonable and consistent way is one of the core tasks of this assignment. There is not only one appropriate way to do so. You are expected to make judgment calls where there is not one (or only one) obviously correct way to proceed with cleaning and integrating the source data. Part of your assessment is on your willingness and ability to apply good and consistent judgment in the face of uncertainty and required trade-offs.

You need to complete *at least* the following data shaping and cleansing steps to load data into the PropertySale table:

- Deal appropriately with duplicate records. This includes both exact duplicates and near duplicates.
- Handle data quality problems with missing fields and fields so far out of their expected value range that they are likely incorrect. This includes, but is not limited to, issues with transactions with unrealistically low transaction prices.
 - *Note - there are many transaction records for transactions with 0 gross square feet and/or 0 land square feet. This should catch your eye as likely data errors. These are not, however, necessarily errors. Property ownership in New York City has some unusual legal structures, such that a person purchasing a condominium or a co-op apartment may not actually have any legal right to the specific building and/or the land on which the building sits. They are basically just purchasing the right to live in a specific unit within a building. Those transactions are then recorded as having 0 sq feet of land or building changing hands. Since these are very common types of real estate holdings in New York City, many transactions show 0 for these fields. Further, the data is not consistently recorded across the years. For some years, these fields are null when there is no data available for the field, for other years they are set to 0 when there's no data available. Likewise, the number of commercial, residential, or total units recorded in the sale are null for some years if there are 0 such units included in the sale. Decide how to handle these values to make their formats consistent across the years and act accordingly.*
- Develop one or more rules to flag transactions that are likely not indicative of the fair market price for the property. These are typically transactions with very low (or zero) prices. Please review the data dictionary carefully for an explanation of what transactions recorded as \$0 mean. Then decide on a rule, or set of rules, to identify transactions that likely do not represent the fair market value of a property (FMV). You should add a flag for each transaction to indicate whether you believe it to be a fair market value (FMV) transaction or a non-market value transaction. Flag transactions that are likely not market-value transactions by setting the PropertySale.FairMarketPriceTransaction field to false.
 - *Note - there is no One Right Rule to determine which transactions are fair market value and which are not. It is clear that transactions with a \$0 sales price are not at fair market value. What you need to do with your rule is to look for a heuristic (rule) that makes a reasonable guess about whether a property transferred at fair market value. I don't expect you to come up with a precise and 100% correct rule. Just one that flags most values that are likely well below market rate. This exercise requires you to look at the available data, choose a reasonable metric and/or threshold, make a (hopefully simple) rule to implement that heuristic, and then test that rule to see how many (likely) false positives and false negatives it creates. Choose a rule that seems to minimize each of those errors, recognizing that you are unlikely to create a single simple rule that eliminates all of them. Developing good judgment about when you've cleaned the data to an appropriate*

level of quality for your analytic and record-keeping purposes (even though some errors likely still exist in the dataset) is an important part of developing your skills as an analyst and data scientist, and an important task in this lab.

- Remove fields that add no information (i.e., are null for all records) or are not required in the target database structure.
- Convert the borough identifier for each property sales record (stored as an integer) into the appropriate name for that borough. There are five boroughs in New York City - Manhattan, Bronx, Brooklyn, Queens, and Staten Island. Determine which number corresponds to which borough (check the data dictionary) and add a step in your Tableau Data Prep flow to convert the ID to a borough name. Store the name in the PropertySale.BoroughName field and the BoroughCode in the original transaction file as an integer in PropertySale.BoroughID.
- Decide which fields to use to for tax class and building class codes for each transaction.
 - There should be only one value for building class and tax class for each transaction in the target database, but multiple columns in the source data might be appropriate sources. Figure out which columns (one for building class and one for tax class) make the most sense for the transactions and eliminate the others.
- Properly separate street addresses and apartment numbers. You will likely find that some PropertySale records have the apartment number appended to the end of the street address field, and others have it listed correctly in the apartment_number column. To the extent possible, you should ensure that all transactions with an apartment number have that apartment number listed in the apartment_number column and that they do not have a redundant apartment number appended to the address field.
- Generate a ParcelID for each transaction. The ParcelID notes column of the NYC Property Assessment Data Dictionary.xlsx describes the formula for doing so. The ParcelID is a 10-digit code generated as follows: The first digit is the Borough ID (1-5), the next five digits are the Block. For Block values with less than five digits, pad the characters to the left of the Block value with leading 0's. The last four digits are the Lot value, also padded to the left with 0's for values below 1000. For example, the ParcelID for Lot 35 in Block 975 of Manhattan (BoroughID = 1) would be 1009750035. As a side note, the BBL column in the Tree Census dataset uses the same encoding strategy.

The output of your Tableau Flow for the transactions dataset should be a .csv file called PropertySales.csv that contains the cleaned, integrated data ready to be loaded into your PropertySales table in SQLite. The output file needs to be properly formatted in Comma Separated Value (.csv) format using exactly the following field headers, listed with exactly these heading names, in exactly this order:

```
Address, ApartmentNumber, Block, BoroughID, BoroughName,
BuildingClassID, CommercialUnits, FairMarketPriceTransaction,
GrossSquareFeet, LandSquareFeet, Lot, Neighborhood, ParcelID,
ResidentialUnits, SaleDate, SalePrice, TaxClass, TotalUnits,
YearBuilt, ZipCode
```

You do not need to provide a TransactionID field for each property sale in this file. Since no TransactionIDs are provided in the source dataset, the data warehouse's schema is configured to automatically generate a unique transaction ID for each sale record inserted into the table.

AnnualSalesByNeighborhood table: You need to create a table to store aggregated summary data about market conditions in various neighborhoods for each year of data in the source dataset in your Tableau Flow. To do so, you must filter out all non-market-price transactions and then build pipeline steps in your Tableau Flow to calculate the total number of market-price transactions organized by year, neighborhood, and building classification, along with the average transaction price for every one of those (Year, Neighborhood, BuildingClassID) combination.

Your AnnualSalesByNeighborhood.csv file needs to be properly formatted in Comma Separated Value (.csv) format. Your file should have the following field headers, listed with exactly these heading names, in exactly this order:

```
Year, Neighborhood, BuildingClassID, NumberOfFairMarketValueSales,
AverageFairMarketValueSalesPrice
```


TaxAssessment table: The TaxAssessment table contains tax assessment data for all physical real estate parcels in New York City. Each assessment record is for a Parcel, which (roughly) corresponds to a physical plot of land with one or more buildings on it. There appear to be some exceptions to this, such as individual apartments listed as lots within a single building, but for the most part, each assessment is tied to a specific plot of land in the city. Unlike the PropertySale table, this table is supposed to be a comprehensive dataset of all taxed real estate parcels in the city. The PropertySale table, on the other hand, only lists properties that changed ownership during the covered time period.

The TaxAssessment table contains data that has been mostly cleaned. You do not need to modify the data in this table for this lab. Instead, it is provided in the database as source data that you can use to generate the MatView_BlockSummaries and MatView_MultipleParcelOwner tables. The data dictionary in this Excel sheet describes the data in this table - [NYC Property Assessment Data Dictionary.xlsx \(\\$CANVAS COURSE REFERENCE\\$/file_ref/gbcdf2dc589409ad16994f9fd29a3fd66?wrap=1\)](#). That said, there are some cleaning steps you are expected to do when you create the materialized views. Just don't modify the data that already exists in the TaxAssessment table.

MatView_BlockSummaries table: The MatViewBlockSummaries table summarizes the data found in the TaxAssessment table and organizes it by *Block*. Each numbered Block in this dataset corresponds (roughly) to one city block. This table, then, describes the total number of Lots, Buildings, and Units built on each Block. A *Lot* is a subdivision of a block that typically contains one building, though it may contain no buildings or more than one building. A *Unit* is a residential or commercial built space on a Lot.


Your task for this lab is to write a SQL script that fills the MatViewBlockSummaries table with calculated values for each Block in the city for 2025. Use the TaxAssessment table as your source data to create this summary table. The meaning of each field in the target database table should be straightforward to interpret. You need to generate one row in that table with the summarized data for all parcels on that block, for every block listed in the TaxAssessment table for tax assessment year 2025.


Your SQL script needs to do the following: (a) empty the current table of all entries for 2025, (b) generate a summary for each block in 2025, and (c) load the generated summary values into the MatViewBlockSummaries table. If you write such a script, it needs to run from beginning to end without human intervention beyond loading the script into Valentina Studio and hitting the execute button. It should also clean up and free any temporary storage you use to generate the table.

MatView_MultipleParcelOwners table: The general instructions for this table are the same as for the MatView_BlockSummaries table, but you will generate a different summary view. Your SQL script needs to generate one row in the MatView_MultipleParcelOwners table for each entity listed as the Owner of more than one parcel in the TaxAssessment table for 2025. You should not include Owners in this table who are only listed as owning a single parcel.


Although you can treat the data in the TaxAssessment table as basically clean and correct, your MatView_MultipleParcelOwners table should not include "Owners" where the name of the owner clearly indicates that the owner of the parcel is unknown. There are multiple terms used in the dataset to indicate as much. When you do the aggregations, the largest property owner in New York should not be listed as "Unknown" or something similar. Use your judgment about which owner names are likely indicative of missing data rather than an actual single owner. As noted earlier, this cleaning process should only be applied to the MatView_MultipleParcelOwners table. Do not modify any data in the TaxAssessment table.

TreeCensus table: As discussed in the data source section, one of the partners in the real estate advisory firm would like to assess the prevalence and health of street trees in various parts of the city to understand if and how that affects property values. To that end, you need to retrieve the NYC 2015 Tree Census dataset at:

- New York City Open Data website page for 2015 tree census (download source data from here):
 - https://data.cityofnewyork.us/Environment/2015-Street-Tree-Census-Tree-Data/uvpi-gqnh/about_data 
 - (https://data.cityofnewyork.us/Environment/2015-Street-Tree-Census-Tree-Data/uvpi-gqnh/about_data)
- Source Data dictionary for tree census source data (link available on NYC Open Data website page referenced above):

- <https://data.cityofnewyork.us/api/views/uvpi-gqnh/files/8705bfd6-993c-40c5-8620-0c81191c7e25?download=true&filename=StreetTreeCensus2015TreesDataDictionary20161102.pdf> 
 (<https://data.cityofnewyork.us/api/views/uvpi-gqnh/files/8705bfd6-993c-40c5-8620-0c81191c7e25?download=true&filename=StreetTreeCensus2015TreesDataDictionary20161102.pdf>)

Once you have downloaded that source data, you need to create a Tableau Flow that will clean and shape the dataset to conform to the specification provided by the following data dictionary and the TreeCensus table in the data warehouse:

- Data dictionary for investment firm data warehouse (destination table):
 - [Street Tree Census Data Dictionary \(Data Warehouse - destination\) v1.1.xlsx](#) (<https://canvas.cmu.edu/courses/42925/files/11952762?wrap=1>)  (https://canvas.cmu.edu/courses/42925/files/11952762/download?download_frd=1)

In addition to transforming the source data to the field structure required by the data warehouse data dictionary, the company has requested that you remove all data about "stumps" from the dataset before loading it into the data warehouse. They would like you to keep records of trees that are dead but still standing, but not data about trees that have been cut down so that only stumps remain.

Your TreeCensus.csv file needs to be properly formatted in Comma Separated Value (.csv) format. You can extract Block and Lot fields from the source files "bbl" field. "bbl" stands for "Borough, Block, Lot" and uses the same encoding described for PropertySale.ParcelID above. Your file should have the following field headers, listed with exactly these heading names, in exactly this order:

```
TreeID, BoroughName, Block, Lot, StreetAddress, ZipCode, NTA_Name,
TreeIsAlive, TreeHealth, TrunkDiameterInches, RootProblemObserved,
TrunkProblemObserved, BranchProblemObserved, SidewalkDamageObserved,
ScientificName, CommonName, Latitude, Longitude
```

This list includes many of the most obvious data quality issues with the dataset, but it does not identify all possible data quality issues with the dataset or all of the transformations you may need to do to fill your SQLite database with high-quality data in the format requested. However, it outlines the primary steps to complete the lab. As you identify additional data quality concerns, you should decide how best to handle them and then do so.

Deliverables:

You need to submit the following for your lab:

- One or two Tableau Data Prep packaged data flow files (.tflx). When downloaded and opened by the grading team, the files need to read in, clean, and shape the provided datasets to create the cleaned .csv files specified ready for loading into your SQLite data warehouse. If desired, you may break this into two files - one for the PropertySale.csv and AnnualSalesByNeighborhood.csv files, and the other for the TreeCensus.csv file. Alternatively, you may generate all three .csv files with a single .tflx flow file — your call.
 - Packaging your entire flow (flow + data) in the .tflx format is important to ensure that the graders can open and test your flows and review the source data you are using for the TreeCensus table.
 - We will test your .tflx file(s) by downloading them from Canvas, running the flows, and reviewing the .csv files they generate.
 - Name these files as follows (change the XX to your lab group number):
 - One .tflx file option - *Lab 2 Flow - Group XX.tflx*
 - Two .tflx files option - *Lab 2 Property Flow - Group XX.tflx* and *Lab 2 Tree Census Flow - Group XX.tflx*.
 - Failure to use these file names will cause problems with our automated testing system, so we will deduct points for incorrect naming. This is consistent with "real world" data pipelines, where the loading system typically expects input file names to use a specified naming convention.

- One file with your SQL script to load the MatView_* tables. This should be a single script that the grading team can load and directly execute for testing. The script needs to empty the existing MatView_* tables of all records from the 2025 tax assessment year and then generate the data for the MatView tables and load it into them. Finally, your SQL script should clean up and free any temporary storage used to generate the MatView tables.
 - Name this file *Lab 2 MatView - Group XX.sql*.
- A SQLite .db file named *Lab 2 Data Warehouse - Group XX.db* containing your entire data warehouse, with all tables loaded with their specified data.

Groups: This lab is a group assignment. It must be completed by the teams assigned for this course. You are welcome to organize the work within your team however you choose, but you are expected to work directly only with your group members to complete the assignment.

Grading criteria: We will use the attached Canvas rubric to grade your submission.

DMF M2-24 Lab 2 Rubric (Data Wrangling)

Criteria	Ratings	Pts
<p>Quality and correctness of data loaded into SQLite database</p> <p>Student has successfully loaded cleaned data output from Tableau Data Prep into SQLite. Database tables each contain the correct (or very close to correct) number of rows. Entries missing data are properly labeled as null, no duplicate records have made their way into the database tables. SQL "audit" queries return expected values. This criterion assesses the correctness of the data loading process, not the data flow steps leading to the load.</p>		10 pts
<p>TDP flow correctly creates the AnnualSalesByNeighborhood table data such that every combination of Year, BuildingID, and Neighborhood present in the source data has exactly one entry in the AnnualSalesByNeighborhood table. Each row in the AnnualSalesByNeighborhood table properly calculates the number of market price transactions, and average transaction value for that Neighborhood for that year.</p>		10 pts
<p>TDP Cleaning Flow - appropriate removal and merging of duplicate sales records.</p>		10 pts
<p>TDP Cleaning Flow properly cleans addresses and apartment numbers, and sets the PropertySale.ParcelID field.</p>		8 pts
<p>TDP Cleaning Flow sets and implements a reasonable rule for estimating whether a property sale was completed at fair market value, updates each sales record accordingly, and carries that flag all the way through the flow to the final database load.</p>		7 pts
<p>Building codes have been appropriately scraped from NYC website and loaded into SQLite BuildingClass table without creating any data quality problems (such as duplicate records). BuildingClassCodes in other tables properly reference the BuildingClass table such that the description associated with each BuildingClassCode can be retrieved and Joined as needed when queries need to find the description for a specific property sale or hotel.</p>		5 pts
<p>Lab 2 MatView - Group XX.sql SQL script generates correct and complete summary data and loads it into the MatView_BlockSummaries table.</p>		10 pts
<p>Lab 2 MatView - Group XX.sql SQL script generates correct and complete summary data and loads it into the MatView_MultipleParcelOwners table. All properties listed with an Owner name that indicates the actual owner is unknown (e.g. "UNKNOWN OWNER", "OWNER UNKNOWN", or similar) are excluded from the load into the MatView_MultiplePropertyOwners table.</p>		10 pts
<p>TreeCensus data has been properly collected from the online source and used as source data in a Tableau Prep flow. The submitted flow file loads, cleans, and transforms the source data precisely into the specified format (.csv file) to load the data into the target data warehouse. The Tableau Prep packaged flow file is packaged correctly and runs to produce the target .csv file without error.</p>		15 pts
<p>TreeCensus data loaded into the data warehouse has approximately the correct number of rows. Data warehouse structure has not been altered. There are no material data quality issues apparent in the loaded data.</p>		5 pts
<p>All files and data are properly packaged, named, and submitted according to specifications in the assignment. Naming conventions have been followed precisely to allow for automated loading and grading of submitted work.</p>		5 pts
<p>Neither the TDP Cleaning flow nor the SQL scripts generate or otherwise cause other data quality problems at any point throughout the cleaning, integration, shaping, or upload processes. TDP Cleaning Flow appropriately addresses all other data quality problems related to property sales that were identified either in the assignment spec as required cleaning/shaping/integration steps, or identified and discussed during class sessions. This specifically does _not_ create an expectation to repair all subtle data quality problems identified; just the ones that have been specifically flagged as needing repair.</p>		5 pts
Total Points: 100		