Introduction and initial assessment of dataset

The farmersmarkets.csv file contains data including markets' name, type of products available, time of operation, market location and websites.
The first row of the csv file is the column name and the first column is the FMID of each market, followed by market name which ideally should only contain strings without special characters. Then it lists all the websites and Facebook, twitter and other media sites available, however, these columns contain values which are not consistent. Since some of them only include user name without valid URLs while others contain URLs.
The next 5 columns contains the location information for the specific market. The quality issue for city and states is that some values are in lowercase while others might be in uppercase. Another issue is that special characters also presents in those columns. Needless to say there are some extra white spaces before and after these strings.
Next 8 columns include data for operation dates. The seasonal dates are not all in the same standard date format which makes it hard for user to query or select data following certain rules. Some entries are in date/mm/year format while others enter the period in month to month format(e.g. July to November, 06/25/2014 to 09/30/2014). The next 2 columns include information for X and Y corresponding to latitude and longitude values. The rest columns representing the availability of type of food provided in the market. Each entry is marked either Y or N. But Null values are also present in those columns. The last column in this csv file is the updated time for the data set. Most rows contain data formatted in mm/dd/yr h:m:s format, but some rows only have year.

The major issue of this data set is inconsistency. Most values are entered without following specific rules such as ISO standard formats. This could make query data harder than expected and the results of these queries will not be conclusive enough for the user.

Use case can be achieved from this dataset

Even though the dataset has lots of flaws, user can still get correct result for questions like: Which Farmer's market sell Seafood and accepts credit card payment? People these days might only bring their credit card with them and if they don't bother to visit the bank and get cash before go buy seafood, they might want to know where is the that could let them purchase seafood with credit card.

Use case that can be achieved after cleaning this dataset

After cleaning the dataset, we hope to be able to answer questions like where is the closest market that I can go to buy flower and seafood with only credit card. By filtering data through its location and operation hour, we could get the closest opening market and then we can further select the data with flower and seafood entries to be 'Y' in order to answer the above question.

Use case that can never be achieved from this dataset

Among all the markets available, which market sells the cheapest flower? Since the dataset itself does not contain market prices for their products, we cannot get the price comparisons directly from this dataset.

Data cleaning methods and processes Using Openrefine

Provenenance.json file contains all the cleaning steps used in openrefine.

Trim extra white spaces from all the columns, this step is performed both at the start and at the end of the whole process to make sure no extra spaces are present.

1. Text transform on 392 cells in column MarketName: value.trim()
2. Text transform on 43 cells in column MarketName: value.replace(/\s+/,' ')
3. Text transform on 0 cells in column FMID: value.trim()
4. Text transform on 0 cells in column FMID: value.replace(/\s+/,' ')
5. Text transform on 21 cells in column Website: value.trim()
6. Text transform on 0 cells in column Website: value.replace(/\s+/,' ')
7. Text transform on 32 cells in column Facebook: value.trim()
8. Text transform on 3 cells in column Facebook: value.replace(/\s+/,' ')
9. Text transform on 11 cells in column Twitter: value.trim()
10. Text transform on 0 cells in column Twitter: value.replace(/\s+/,' ')

Cluster identical names together (website, city, state, country)

30. Mass edit 160 cells in column Website
31. Mass edit 134 cells in column street
32. Mass edit 552 cells in column city
33. Mass edit 734 cells in column County

Remove special characters (city, street,county)

40. Text transform on 6 cells in column city: grel:value.replace(/[\(\)!@#$%]/,"")
41. Text transform on 0 cells in column County: grel:value.replace(/[\(\)!@#$%]/,"")
42. Text transform on 0 cells in column State: grel:value.replace(/[\(\)!@#$%]/,"")

Convert to ISO standard format (Update time)

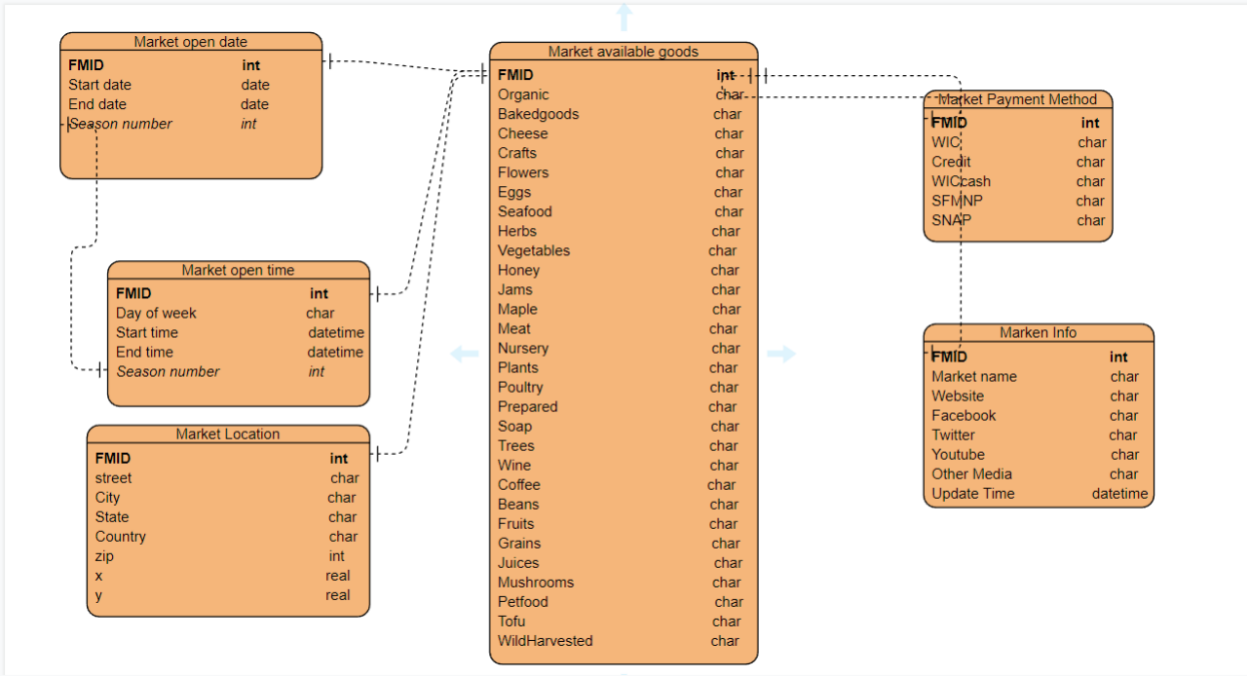
39. Text transform on 8384 cells in column updateTime: grel:value.toDate('d/m/y h:m:s')

Split into two columns(start date and end date for season date 1, season date 2, season date 3, season date 4)

34. Text transform on 235 cells in column street: grel:value.replace(/[\(\)!@#$%]/,"")
35. Text transform on 5479 cells in column Season1Date: grel:value.replace(/to/,"~")
36. Split 5479 cell(s) in column Season1Date into several columns by separator
37. Rename column Season1Date 1 to Season1Date Start
38. Rename column Season1Date 2 to Season1Date End
39. Text transform on 8384 cells in column updateTime: grel:value.toDate('d/m/y h:m:s')
40. Text transform on 6 cells in column city: grel:value.replace(/[\(\)!@#$%]/,"")
41. Text transform on 0 cells in column County: grel:value.replace(/[\(\)!@#$%]/,"")
42. Text transform on 0 cells in column State: grel:value.replace(/[\(\)!@#$%]/,"")

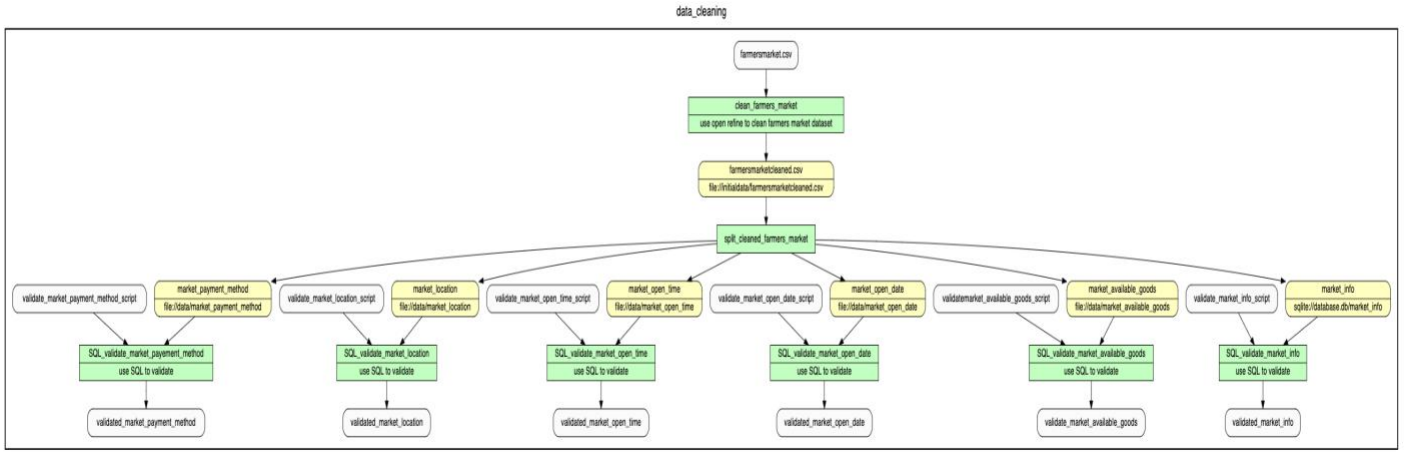
The dataset after cleaning is saved as farmersmarket_cleaned.csv in the initial_assessment_and_data_cleaning folder. After the cleaning of original data, I can then split the csv file into several files to create tables for next step – SQL queries.

Develop a relational database schema



Details for sample queries and integrity constraints are included in the IC_queries file in the SQL folder along with ER_diagram.

Workflow model



Shown as an image in workflow_image in the folder with workflow showing all the workflow steps in the workflow folder.