FARMERS MARKET DATA CLEANING

Balaji H S Rajaputra
Department of Computer Science
University of Illinois at Urbana-Champaign
Champaign, IL, USA
bhr4@illinois.edu

***Abstract*—This document is an exploration of farmers market data analysis. The report summarizes an end to end process of data preparation, data cleaning and provenance. Used tools and techniques learned in CS 513 theory and practice of data cleaning course.**

I. Overview and Initial Assesment

The farmers market data set contains more than 8 thousand records and 59 columns. The data includes different market names, their web site, social media information, address, market seasons and the timings, the market longitude and latitude values, items that are sold in the markets, examples – groceries, dairy products, meat, vegetables, drinks, fruits and more.

Initial data set is not good, it's messy with lots of white spaces, special symbols, and useless data. It is not easy to use for any practical analysis. To make the data meaningful and ease of use, we need to perform data cleaning activities.

II. Proposal

By performing data cleaning activities and make the dataset useful. The data set can be used to analyze farmer market details like few mentioned below

- Analysis can be performed to understand the markets by location using state, city or longitude and latitude values.
- Seasons of farmers markets, the products sold by each market.
- Can categorize the markets by type of products sold in the farmers market.
- Categorize most and least markets by city, state or by regions
- The provenance and workflow of data cleaning activities can be referenced to get the history and be used for future projects.

III. Initial Data Exploration

Understanding of the current data set is most important before starting any data cleaning activities. Data quality is crucial for data user to take an advantage of it, the quality data brings meaning and productivity that gives value for future analysis and/or critical decisions.

During my initial review, I found the data set has several issues that need to be addressed by performing data cleaning activities. Open source tools like Open Refine, datalog, SQLite are few that can help in achieving our goals for this project.

IV. OpenRefine – Data Wrangling

As part of the data wrangling activities, I used very powerful Open Refine tool to detect and clean data errors, splitting, transforming and linking data, remove white spaces and special characters.

Used Open Refine tools common transforms features like trim trailing, leading white spaces and collapse consecutive whitespace, used letter case conversions. GREL to replace special symbols. changing the date formats as desired.

Clustering is one of a very impressive and useful functionalities that is used to cluster similar text/data to replace the values with one decided value that is closure and more meaningful.

After successfully performing step by step cleaning activities, exported the refined data set into 4 different .csv files. Used custom tabular exporter feature to select required fields and downloaded different files for next step.

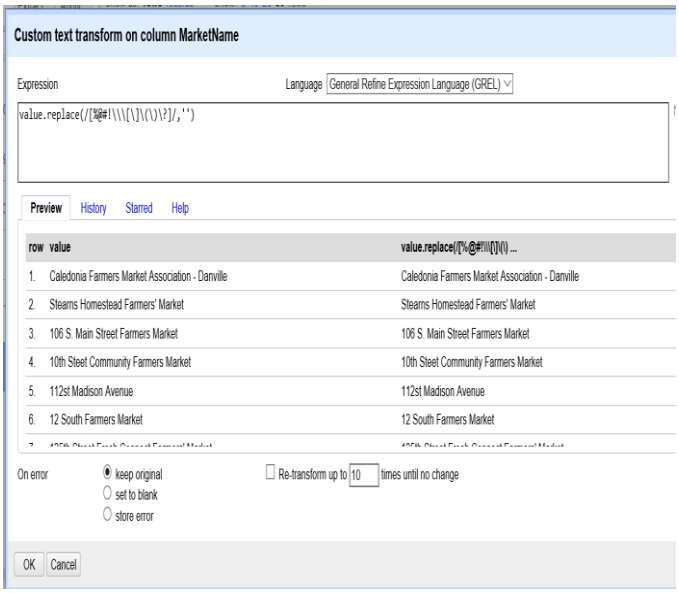
Finally, extracted parts of Open Refine wrangling operations history as JSON file that can apply to this or other projects in the future.

Below is the list of data cleaning activities performed

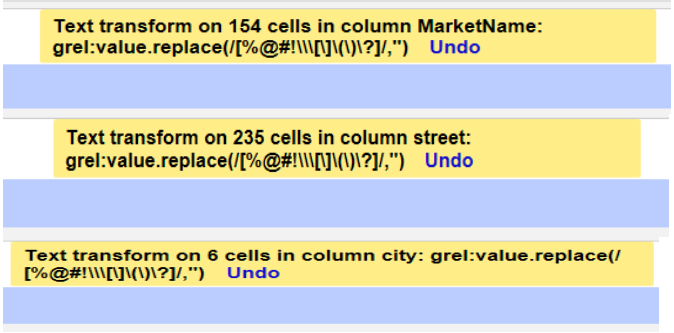
- Trim leading and trailing whitespaces
- Collapse consecutive whitespaces
- Cluster operations using key-collusion method and fingerprint keying function

- Cluster operations using key-collusion method & ngram fingerprint keying function
- Deleted unusual characters using one regular expression to replace with desired value.
- Performed common transforms, to date on date columns in the data set.

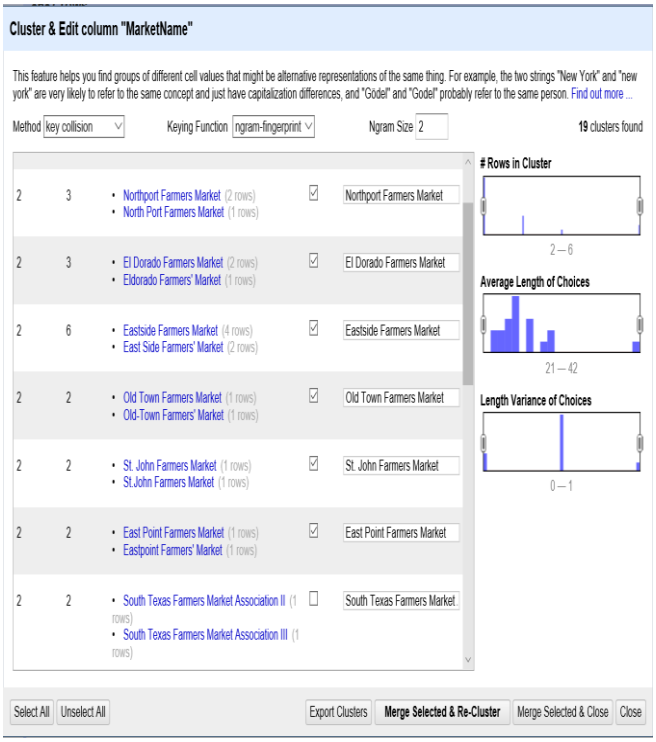
Below are some screen shots of GREL on MarketName column



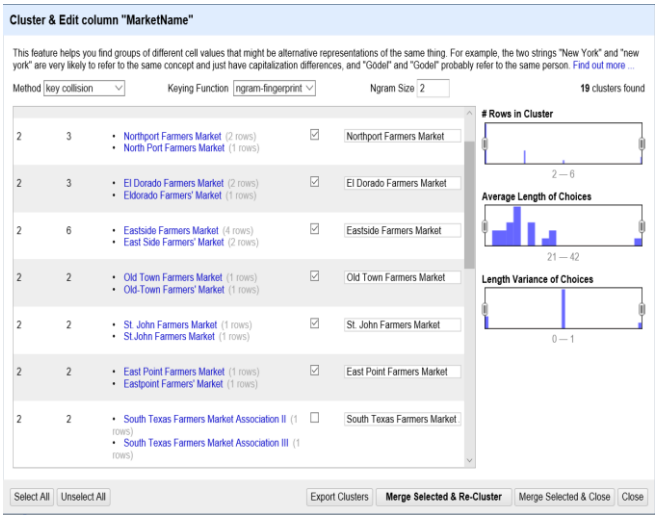
Result of GREL on columns like Market Name, Street and City



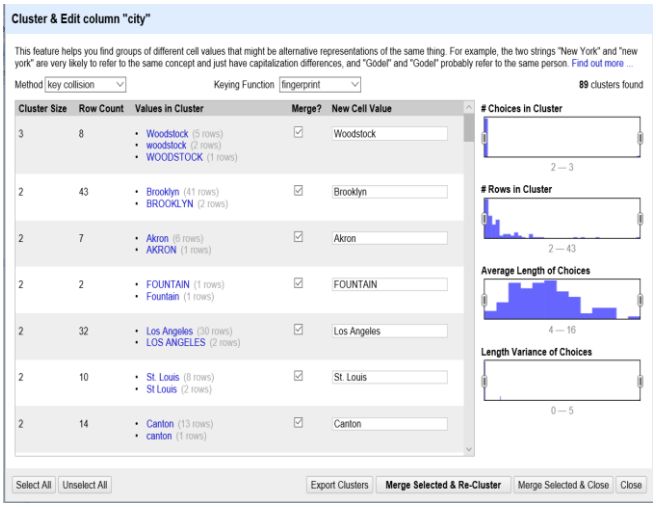
- Seasons date information was in the text format and cannot be used to perform any date range activities. Splitting and converting into date format is required on these columns
- Split the season date column into multiple columns using substring function. Created 2 columns like season1 start date and season1 end date.
- For start date used GREL functions substring(value,0,10) & for end date substring(value,13,24)
- Removed leading, trailing white space and consecutive white spaces.
- Converted the newly created date columns into proper format using Common Transform to date option.
- Same steps repeated to split and created start and end dates for season2, Season3 and season4 date columns.
- Next, using the text facet, performed the cluster operations using key collision method and fingerprint keying functionality. (7) The finger printing method is very fast & simple, works relatively well in variety of conditions and it is the least likely to produce false positives.



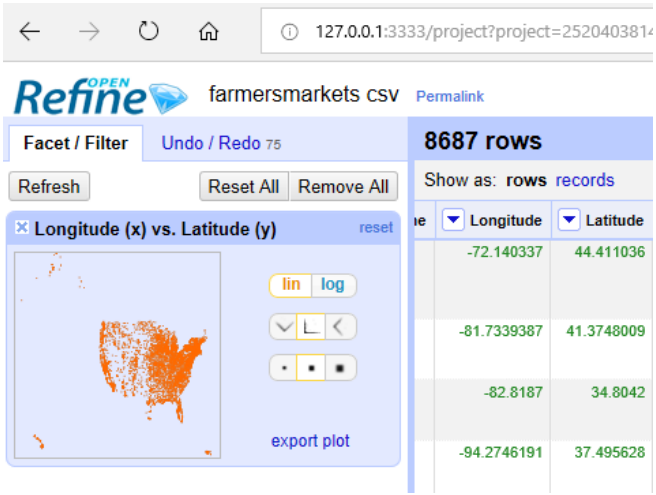
- Performed clustering operations on MarketName, City and address columns, used key collision and fingerprint function first then key collision and ngram-fingerprint function



City Column



- Renamed X and Y columns to latitude and longitude, then text transformed to number format.



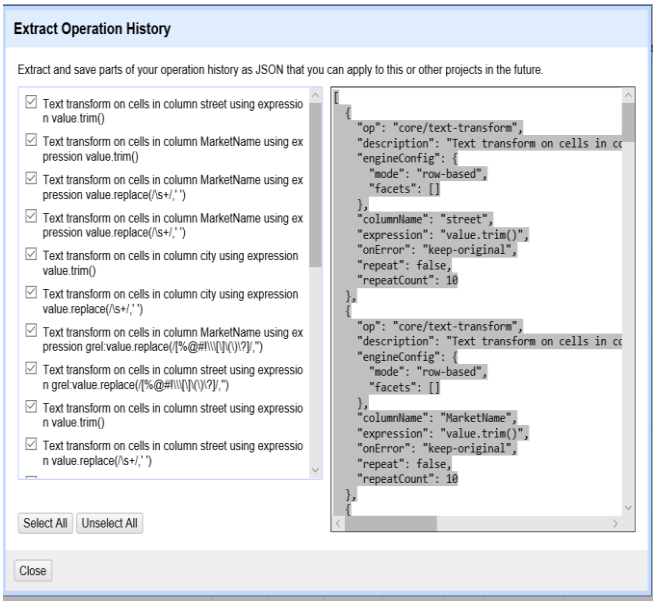
The farmers market dataset is ready to export for further integrity, constraints and cleaning activities in SQLite database.

Further to maintain relational data model, farmers markets single large data set has been exported in to into 4 files.

1. Farmers Market Address –file is exported with FMID (PK), Street, City, State, County, Zip, Latitude, Longitude and Location details.
2. Farmers Market Social Media – this file has FMID, Market Name, Website, Facebook, Twitter, YouTube and OtherMedia columns.
3. Farmers Market Products – this file has columns like FMID, Credit, WIC, WICCash, SFMNP, SNAP and all other fields related to the products that are sold in the farmers market.
4. Farmers_Markets_Seasons – this file exported with FMID, Season1_Start_Date, Season1_End_Date, Season2_Start_Date, Season2_End_Date, Season3_Start_Date, Season3_End_Date, Season4_Start_Date, Season4_End_Date, Season1Time, Season2Time, Season3Time, Season4Time.

FMID column can be used as key field to join tables. Splitting one large file into multiple tables will eliminate unnecessary process time/load on the database process engine, improve performance, focus on portion of data and relatively retrieve subset of the data as needed.

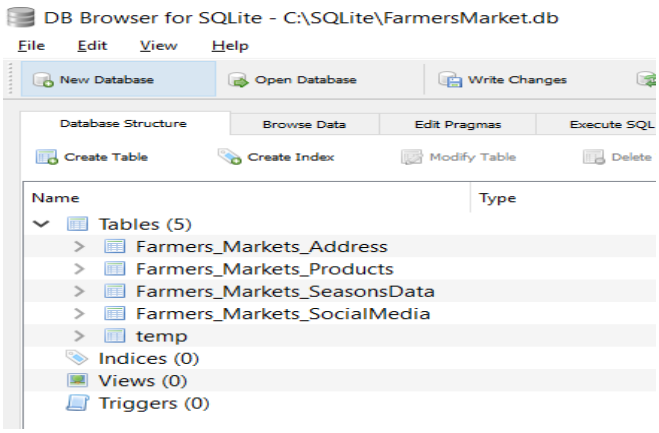
Finally, extracted all activities in JSON format for future usage.



V. SQLITE - RELATIONAL SCHEMA & INTEGRITY CONSTRAINTS

SQLite is in process library that implements a self-contained, zero-configuration, serverless, transactional SQL database engine. SQLite is an embedded database engine. (5) Unlike other databases, SQLite does not have a separate server process.

Imported above 4 .csv files into database in SQLite and then performed integrity constrain checks.

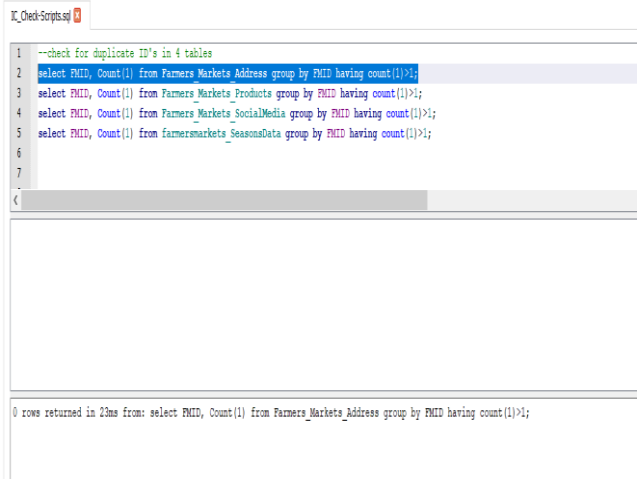


Verified data in each table, identified dirty/null data and flagged those records.

The flagged records can be deleted OR excluded for future analysis.

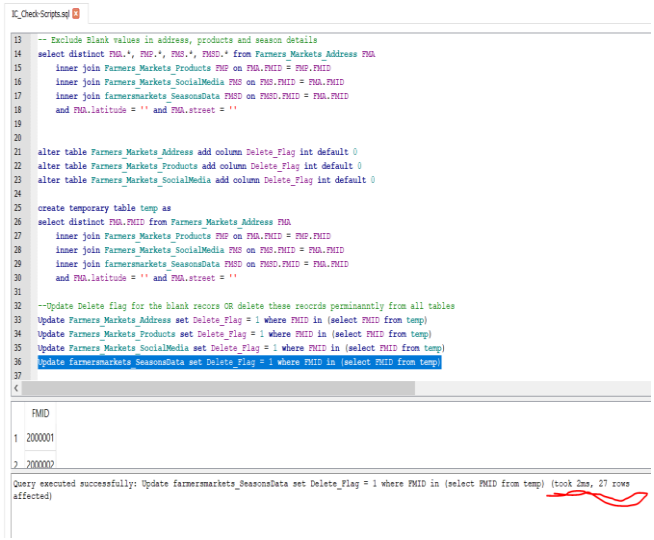
Below integrity constraints checked on farmers market address, products, social media and season data tables

1. Checked for any duplicate FMID in all 4 tables. The FMID is key to join tables so checking for duplicates and correcting the values is important. (Found no duplicates)

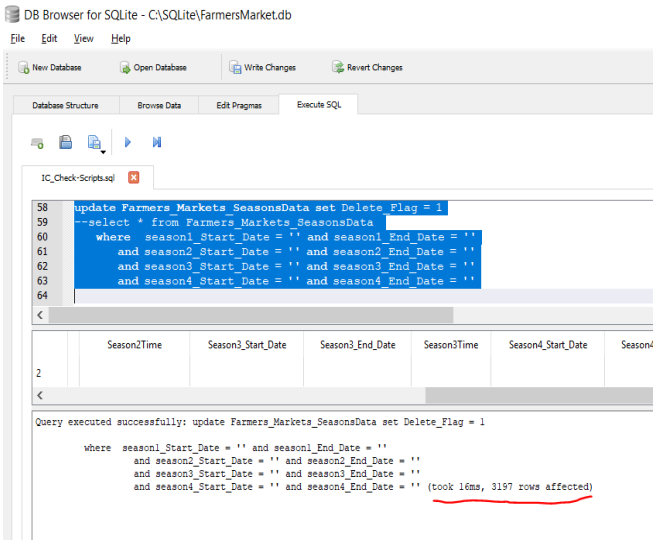


2. Verified columns street and latitude values for null values in the address table. Found 27 records that are blank, with no address, Longitude and Latitude coordinates details. No data means, these 27 records cannot be used for location analysis.

3. Joined results from above step with remaining 3 tables to find any additional details in those table that may be used for further analysis. Identified 27 records have no details in all 4 tables. So, these records are not useful. Updated 27 records with delete flag.



4. Performed similar operation on Famers markets Seasons table. Found 3197 records with no seasons date information. So, this set cannot be used for any season related analysis. Updated the delete flag on these records as well.



5. Checked the seasonal data filtering by date range

```
select season1_Start_Date,  strftime('%Y', season1_Start_Date) as "Year"
       from Farmers_Markets_SeasonsData
      where season1_Start_Date <> ''

select * from Farmers_Markets_SeasonsData
   where strftime('%Y', season1_Start_Date) > '2014'
```

	FMID	Season1_Start_Date	Season1_End_Date	Season1Time	Season2
14	1011881	2016-05-07T04:00:00Z	2016-10-15T04:00:00Z	Sun: 9:00 AM-1:00 PM;	
15	1010966	2015-06-01T04:00:00Z	2015-11-15T05:00:00Z	Wed: 1:00 PM-5:00 PM;	
16	1005299	2015-06-01T04:00:00Z	2015-11-15T05:00:00Z	Tue: 2:00 PM-6:00 PM;	
17	1010994	2016-05-05T04:00:00Z	2016-09-01T04:00:00Z	Thu: 3:30 PM-6:30 PM;	
18	1009959	2016-01-01T05:00:00Z	2016-12-31T05:00:00Z	Thu: 11:00 AM-3:00 PM;Fri: 11:00 AM-3:00 ...	
19	1010775	2015-01-18T05:00:00Z	2020-01-05T05:00:00Z	Sun: 9:00 AM-1:00 PM;	
20	1018656	2017-06-11T04:00:00Z	2017-10-29T04:00:00Z	Sun: 10:00 AM-2:00 PM;	
21	1005310	2015-06-01T04:00:00Z	2015-11-01T04:00:00Z	Thu: 2:00 PM-6:00 PM;	

```
2705 rows returned in 10ms from: select * from Farmers_Markets_SeasonsData
      where strftime('%Y', season1_Start_Date) > '2014'
```

6. Checked the seasonal dataset to identified number of valid records per year. This gives us an overview of number of records exist for each season by year and what season/ year data can be used to perform accurate analysis.

```
select strftime('%Y', season1_Start_Date) as "Year", count(1)
       from Farmers_Markets_SeasonsData
      where season1_Start_Date is not null  group by "Year"  order by 2 desc
```

	Year	count(1)
2	2014	1274
3	2016	1198
4	2015	802
5	2017	693
6	2013	641
7	2012	52
8	2002	11
9	2020	11
10	2011	6
11	0201	3

```
13 rows returned in 30ms from: select strftime('%Y', season1_Start_Date) as "Year", count(1)
       from Farmers_Markets_SeasonsData
      where season1_Start_Date is not null  group by "Year"  order by 2 desc
```

7. Checked integrity constraints on Farmers markets Social Media set. Ran scripts to see any records without market name and verified data count for each column to identify any columns that are completely blank and can be deleted.

```
select count(*) [TotalRows],
       sum(case when MarketName = '' then 0 else 1 end) [MarketName],
       sum(case when Website = '' then 0 else 1 end) [Website],
       sum(case when Facebook = '' then 0 else 1 end) [Facebook],
       sum(case when Twitter = '' then 0 else 1 end) [Twitter],
       sum(case when Youtube = '' then 0 else 1 end) [Youtube],
       sum(case when OtherMedia = '' then 0 else 1 end) [OtherMedia]
       from Farmers_Markets_SocialMedia
```

	TotalRows	MarketName	Website	Facebook	Twitter	Youtube	OtherMedia
1	8687	8687	5212	3931	1018	185	730

8. Verified farmers markets product table, ran scripts to find number of records that do not have any product related information, these records are useless, so updated the delete_flag.

```
select * from Farmers_Markets_Products where
                        Credit= 'N'
                    and WIC='N'
                    and WICcash='N'
                    and SFMNP='N'
                    and SNAP='N'
                    and Organic='-'
                    and Bakedgoods=''
                    and Cheese=''
                    and Crafts=''
                    and Flowers=''
                    and Eggs=''
                    and Seafood=''
                    and Herbs=''
                    and Vegetables=''
                    and Honey=''
                    and Jams=''
                    and Maple=''
```

	FMID	Credit	WIC	WICcash	SFMNP	SNAP	Organic	Bakedgoods	Cheese	Crafts
1	1006234	N	N	N	N	N	-			

```
2439 rows returned in 32ms from: select * from Farmers_Markets_Products where
                                                        Credit= 'N'
and WIC='N'
and WICcash='N'
and SFMNP='N'
and SNAP='N'
and Organic='-'
```

9. After all updates, only 6221 records may be used for analysis and the rest 2466 records are blank.

10. Performed sanity check to see for any possible dirty data out of 6221 records and concluded that the data set is good to go for analysis.

IC_Check-Scripts.sql | IC_Check-Scripts1.sql

```
select Delete_Flag, count(1)
   from Farmers_Markets_Products
   group by Delete_Flag
```

	Delete_flag	count(1)
1	0	6221
2	1	2466

11. Finally, removed records that has delete flag, verified all 4 tables to check any further possible cleaning activities, and concluded that the data is fully cleaned and ready to use.

VI. YesWorkFlow – Create Work Flow Model

Created work flows of the entire end to end data cleaning process using YesWorkFlow online editor tool.

Below diagram shows the flow of cleaning activities performed in Open refine.

VII. Conclusion

Data cleaning is the most essential to produce quality data for accurate data analysis. Low quality data can cost huge loss to companies, institutes, government OR who ever rely on the data analysis and decision made based on the analysis reports.

For this project, I used Open Refine & SQLite. Open Refine a powerful tool, has many key features that are useful for cleaning activities. Even with its limitations, Open Refine would be one good choice for data cleaning activities. SQLite is a self-contained, serverless and transactional DB engine. Very useful for integrity constrains check and cleaning relational data models.

This end to end data cleaning project gave me excellent experience. Now, I have greater understanding and confidence of performing data cleaning activities on a huge and messy data sets from beginning to the end.

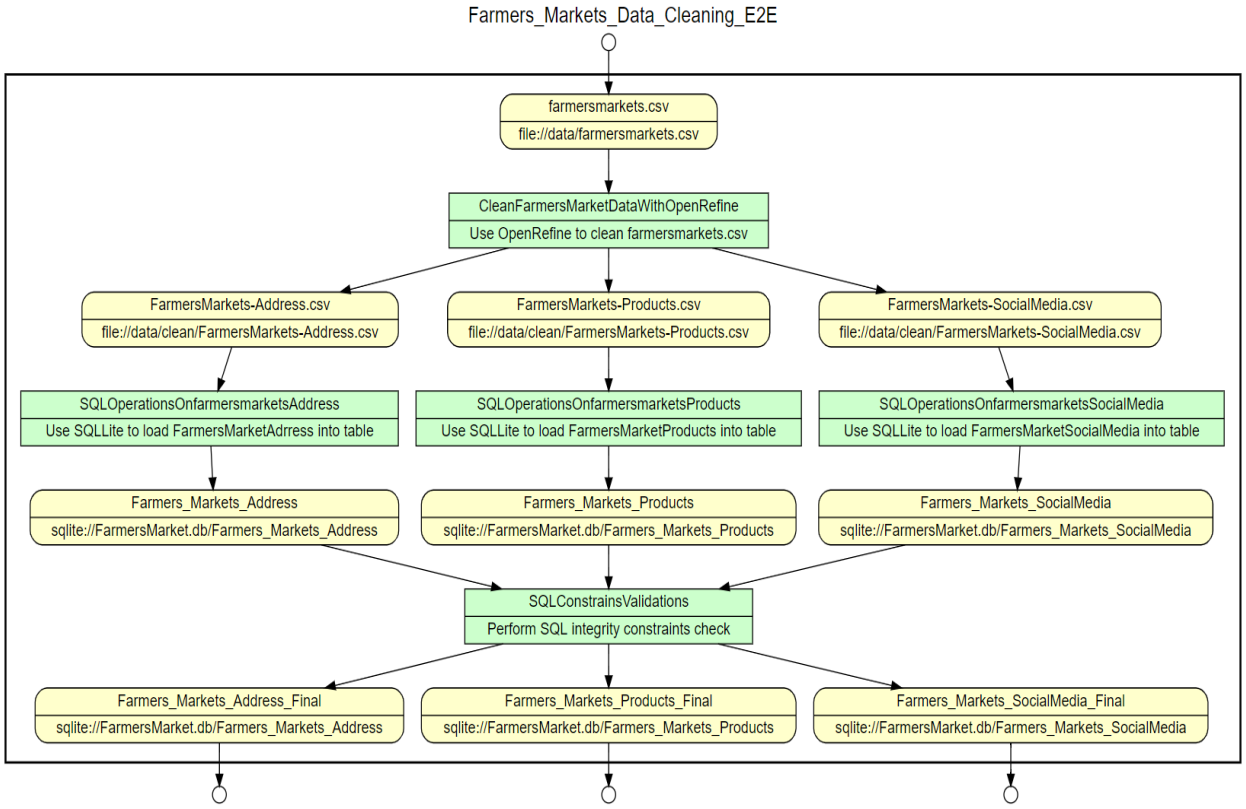
Complete project deliverables are available at GitHub - Link

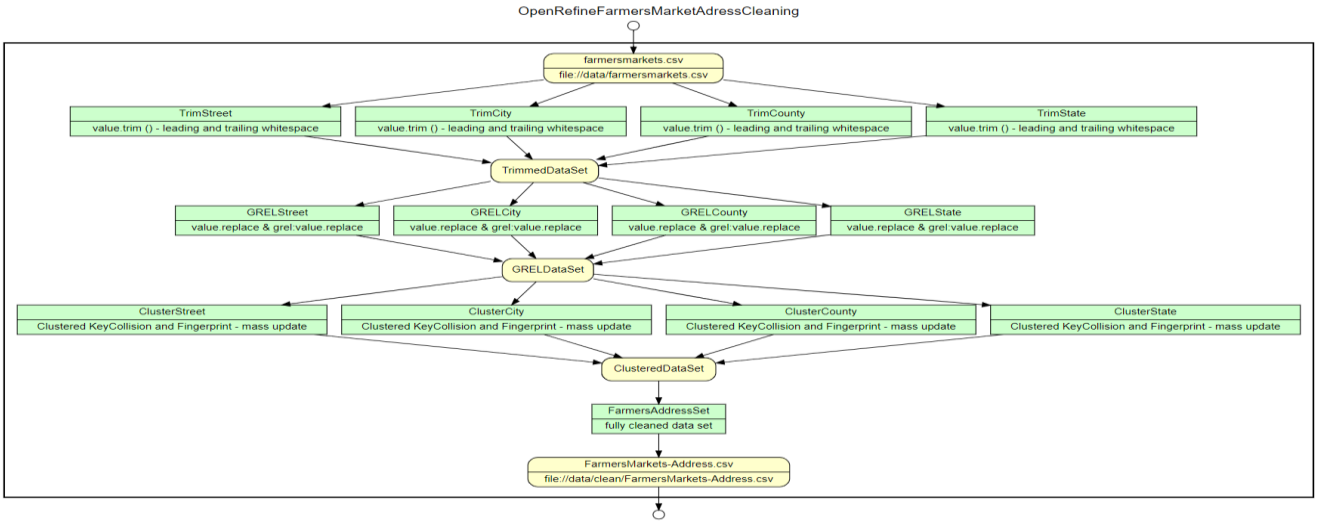
VIII. Acknowledgement

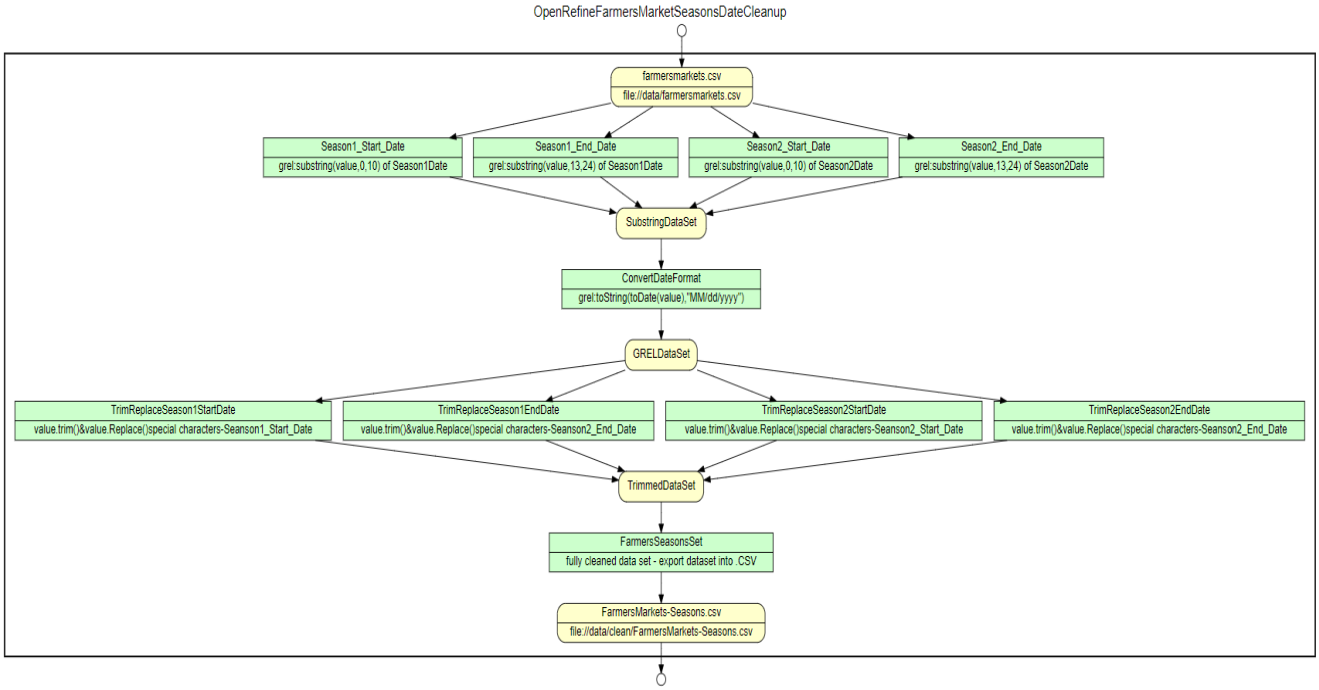
I thank the University of Illinois at Urbana Champaign, Professor, Teaching Assistants for giving me the opportunity, inspiration and valuable guidance for this course and the project.

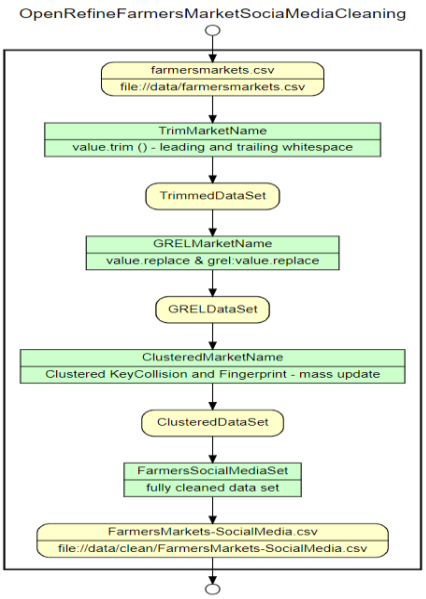
IX. References

1. US Farmers Market - https://www.ams.usda.gov/local-food-directories/farmersmarkets
2. Open Refine - http://openrefine.org/
3. YesWorkFlow - https://github.com/yesworkflow-org
4. YesWorkFlow Online - http://try.yesworkflow.org/
5. SQLIte - https://www.sqlite.org/index.html
6. Regular Expressions - https://regexr.com/
7. Open Refine examples Recipes - https://github.com/OpenRefine/OpenRefine/wiki/Recipes.




OpenRefineFarmersMarketAdressCleaning
farmersmarkets.csv
file://data/farmersmarkets.csv
TrimStreet
value.trim () - leading and trailing whitespace
TrimCity
value.trim () - leading and trailing whitespace
TrimCounty
value.trim () - leading and trailing whitespace
TrimState
value.trim () - leading and trailing whitespace
TrimmedDataSet
GRELStreet
value.replace & grel:value.replace
GRELCity
value.replace & grel:value.replace
GRELCounty
value.replace & grel:value.replace
GRELState
value.replace & grel:value.replace
GRELDataSet
ClusterStreet
Clustered KeyCollision and Fingerprint - mass update
ClusterCity
Clustered KeyCollision and Fingerprint - mass update
ClusterCounty
Clustered KeyCollision and Fingerprint - mass update
ClusterState
Clustered KeyCollision and Fingerprint - mass update
ClusteredDataSet
FarmersAddressSet
fully cleaned data set
FarmersMarkets-Address.csv
file://data/clean/FarmersMarkets-Address.csv


OpenRefineFarmersMarketSeasonsDateCleanup
farmersmarkets.csv
file://data/farmersmarkets.csv
Season1_Start_Date
grel:substring(value,0,10) of Season1Date
Season1_End_Date
grel:substring(value,13,24) of Season1Date
Season2_Start_Date
grel:substring(value,0,10) of Season2Date
Season2_End_Date
grel:substring(value,13,24) of Season2Date
SubstringDataSet
ConvertDateFormat
grel:toString(toDate(value),"MM/dd/yyyy")
GRELDataSet
TrimReplaceSeason1StartDate
value.trim()&value.Replace()special characters-Seanson1_Start_Date
TrimReplaceSeason1EndDate
value.trim()&value.Replace()special characters-Seanson2_End_Date
TrimReplaceSeason2StartDate
value.trim()&value.Replace()special characters-Seanson2_Start_Date
TrimReplaceSeason2EndDate
value.trim()&value.Replace()special characters-Seanson2_End_Date
TrimmedDataSet
FarmersSeasonsSet
fully cleaned data set - export dataset into .CSV
FarmersMarkets-Seasons.csv
file://data/clean/FarmersMarkets-Seasons.csv


OpenRefineFarmersMarketSociaMediaCleaning
farmersmarkets.csv
file://data/farmersmarkets.csv
TrimMarketName
value.trim () - leading and trailing whitespace
TrimmedDataSet
GRELMarketName
value.replace & grel:value.replace
GRELDataSet
ClusteredMarketName
Clustered KeyCollision and Fingerprint - mass update
ClusteredDataSet
FarmersSocialMediaSet
fully cleaned data set
FarmersMarkets-SocialMedia.csv
file://data/clean/FarmersMarkets-SocialMedia.csv