Project Phase 2

CS 513: THEORY AND PRACTICE OF DATA CLEANING

CLEANING AND EXPLORATION OF FARMER'S MARKET DATA

Team Members:
Goutam Debnath(goutamd2@illinois.edu)
Debabrata Biswas(dbiswas3@illinois.edu)
Rohit Narula(rnarula2@illinois.edu)

Table of Contents

Introduction

The project team has identified the Farmer's Market Data as a candidate for CS-513 Theory and Practice of Data Cleaning project submission. This dataset contains information on the location, hours, social media presence, and inventory for 8,677 local farmers markets across America, including Puerto Rico and the U.S. Virgin Islands. Indicator variables allow for easy grouping of markets by what payments they accept and what types of goods they offer. The following part of the document elaborates the procedure, steps, and exploratory data analysis to reach the objective of the project.

Dataset Schema

The cleaned csv file for the dataset was imported in DB Browser for SQLITE to create the following table below:

Tables (1)

Name	Type	Schema
farmersmarkets_clean_openrefine_python		CREATE TABLE "farmersmarkets_clean_openrefine_python" ("field1" INTEGER, "FMID" INTEGER, "MarketName" TEXT, "Website" TEXT, "Facebook" TEXT, "Twitter" TEXT, "Youtube" TEXT, "OtherMedia" TEXT, "street" TEXT, "city" TEXT, "County" TEXT, "State" TEXT, "zip" INTEGER, "Season1StartDate" TEXT, "Season1EndDate" TEXT, "Season1Time" TEXT, "longitude" REAL, "latitude" REAL, "Location" TEXT, "Credit" TEXT, "WIC" TEXT, "WICcash" TEXT, "SFMNP" TEXT, "SNAP" TEXT, "Organic" TEXT, "Bakedgoods" TEXT, "Cheese" TEXT, "Crafts" TEXT, "Flowers" TEXT, "Eggs" TEXT, "Seafood" TEXT, "Herbs" TEXT, "Vegetables" TEXT, "Honey" TEXT, "Jams" TEXT, "Maple" TEXT, "Meat" TEXT, "Nursery" TEXT, "Nuts" TEXT, "Plants" TEXT, "Poultry" TEXT, "Prepared" TEXT, "Soap" TEXT, "Trees" TEXT, "Wine" TEXT, "Coffee" TEXT, "Beans" TEXT, "Fruits" TEXT, "Grains" TEXT, "Juices" TEXT, "Mushrooms" TEXT, "PetFood" TEXT, "Tofu" TEXT, "WildHarvested" TEXT, "updateTime" TEXT)

Use Case – UI

Find the city, state and zip code of all farmers markets that meet the criteria below:

1. Are within a 20-mile radius from a particular city by using the city's latitude and longitude information from the database and applying the Haversine formula which can be used to determine the great circle distance between 2 points on a sphere given their latitude and longitude.
2. Have a valid social media presence (Facebook page) which can be used to check market's rating and reviews.
    a. The URL must start with "https://"
    b. Followed by "www.facebook.com"
3. Sells organic food.
4. Accepts credit cards as a mode of payment.
5. Are operational in a particular month on weekends.

Data Quality Issues

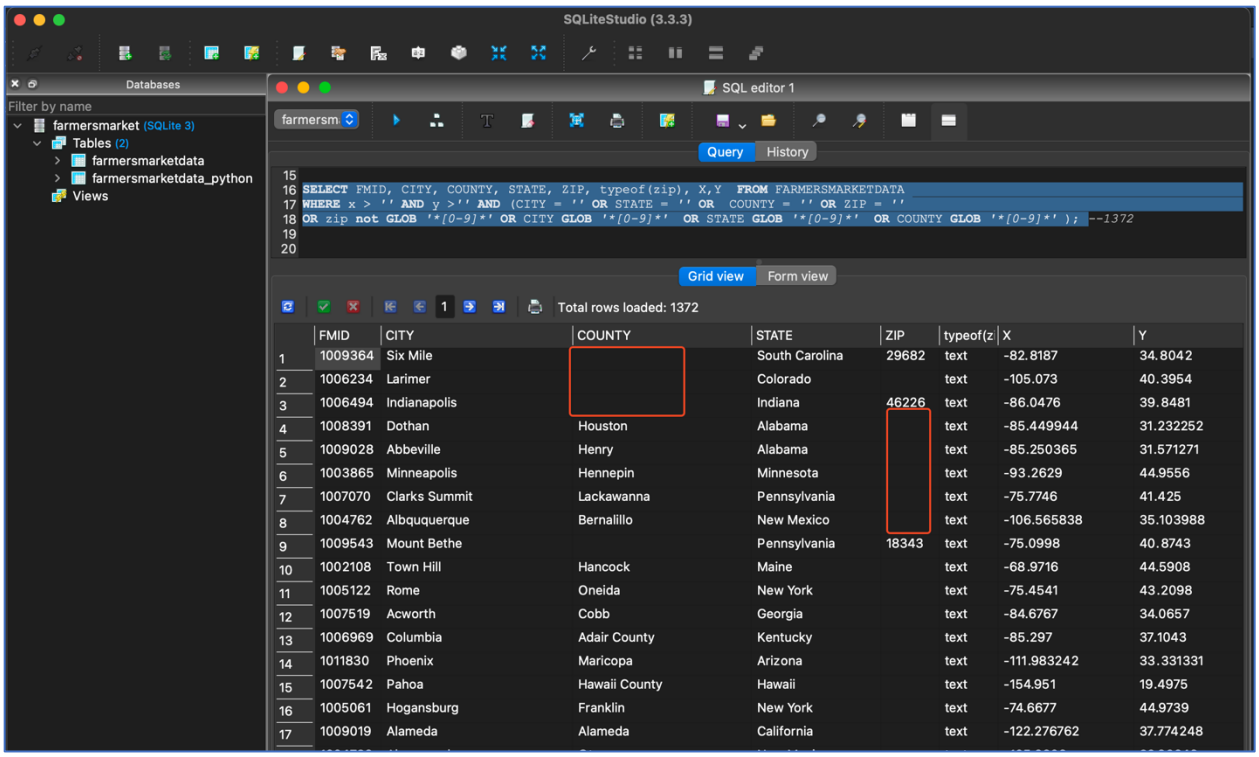
1. Season1Date column values do not follow a consistent format. Most columns follow the format 'YYYY/MM/DD but some columns contain inconsistent values.

	FMID	MarketName	Season1StartDate
1	1008935	water canyon farmers market	Start Date 1/1/13
2	1011963	Wauconda Farmers Market	June 23

2. The Facebook column in the database mostly contains null, empty or invalid URL values.

	FMID	MarketName	Facebook
1	1008935	water canyon farmers market	*NULL*
2	1011963	Wauconda Farmers Market	Wauconda Farmers MArket

3. Rows with a valid latitude and longitude are missing a corresponding value for either city, state or zip code.
    a. Either City, County, State, Zip is Blank.
    b. Either City, County, State has a Number in it.
    c. The Zip has alphabets in it.
    d. The Zip codes are not in DDDDD or DDDDD-DDDD format.



Use Case – U0

1. Find the first 5 farmers markets that are in a particular city and state and accept credit cards as a mode of payment. City, State, and Credit columns contain mostly valid values and querying against them is achievable without any additional data cleaning.

Use Case – U2

1. Find the farmers markets closest to a user location at a particular time and date of the year which are operational, sell a particular food item, have a valid Facebook page and accept Credit Cards as a mode of payment. The dataset is not updated and the latest year in the dataset is 2017 which prevents real time queries against the dataset. 95% of the SeasonDate columns other than Season1Date column have missing or null values which limits the query for farmers markets that are operational at a particular season of the year. No checks can be performed to validate the Facebook column from the dataset as most links are null or invalid and hence no credible information can be extracted regarding the price of goods or ratings / reviews of a particular farmer's market.

Data Cleaning Process Detail:

The farmer's market dataset has been provided as a csv/text format. The following major steps have been performed:

1. Use OpenRefine tool to clean important and obvious data issues visible naturally. OpenRefine is the go-to tool for its ease of use and simplicity and can be used on a huge dataset out of the box.
2. Once the data has been cleaned satisfactorily, it's the time for populating missing us Zip code data. We have identified Python's uszipcode package as a free tool to accomplish the task.
3. Once the data is further cleaned, we rely on SQL to verify the sanctity of the data and run through various SQL integrity constraint checks to ensure the data is clean and trustworthy to answer our base use case. SQL is further used to manually update certain data using information from the web which cannot be cleaned using OpenRefine and Python.

Data Cleaning with OpenRefine

With OpenRefine, data will be clustered if they are in similar text or reformatted to keep consistency of data.
Firstly, all column data should be trimmed and collapsed if they have consecutive white spaces.
Next, county, city, states names are inconsistent. Some of them are in uppercase while others are not. They need to be converted into same format by clustering function.
Date fields are saved as String so we need to convert them to date type so they are all converted to proper date format.

MarketName

- Trim leading and trailing white space.

- Collapse consecutive white spaces.

- Use text facet and cluster by using key collision method and fingerprint keying function.

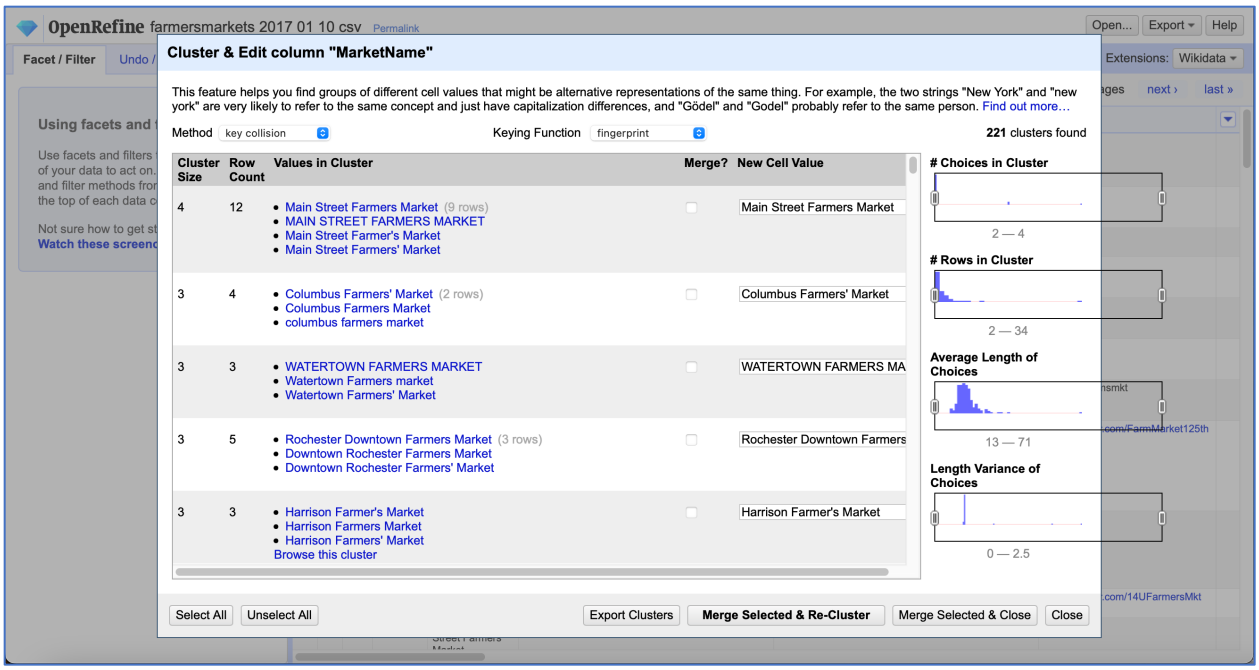


Figure 1 OpenRefine: Cleaning MarketName data

- Use text facet and cluster again by using key collision method and ngram-fingerprint, except those have distinct different names such as Nashville F.A.R.M. II and Nashville F.A.R.M. III.

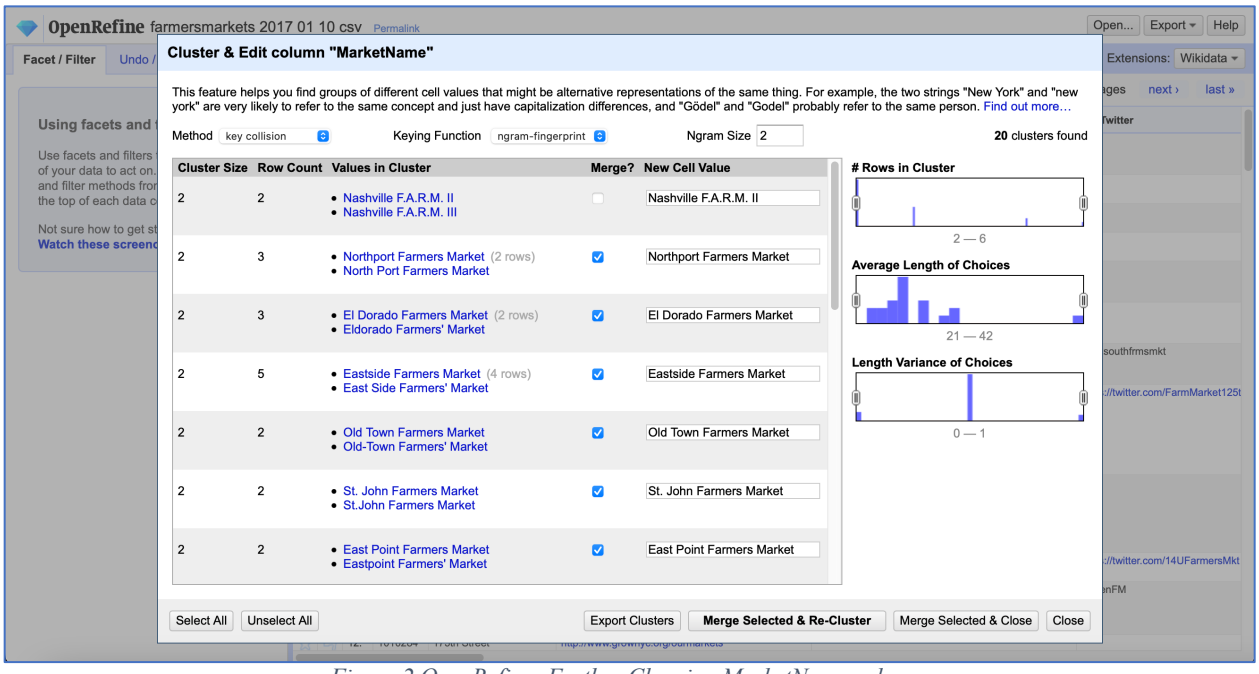


Figure 2 OpenRefine: Further Cleaning MarketName column

Street:

- Trim leading and trailing white space.
- Collapse consecutive white spaces.
- Use text facet and cluster by using key collision method and fingerprint keying function.

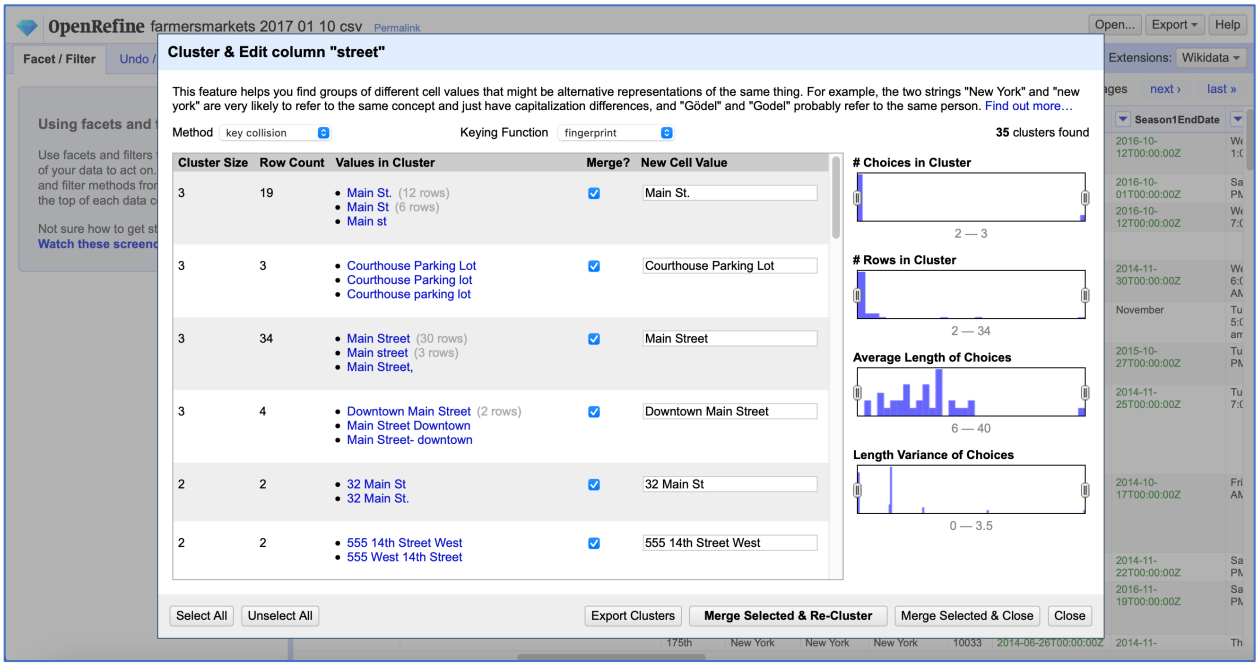


Figure 3 OpenRefine: Cluster & Edit Street Column

- Use text facet and cluster again by using key collision method and ngram-fingerprint except values like Community Center and Unity Community Center.

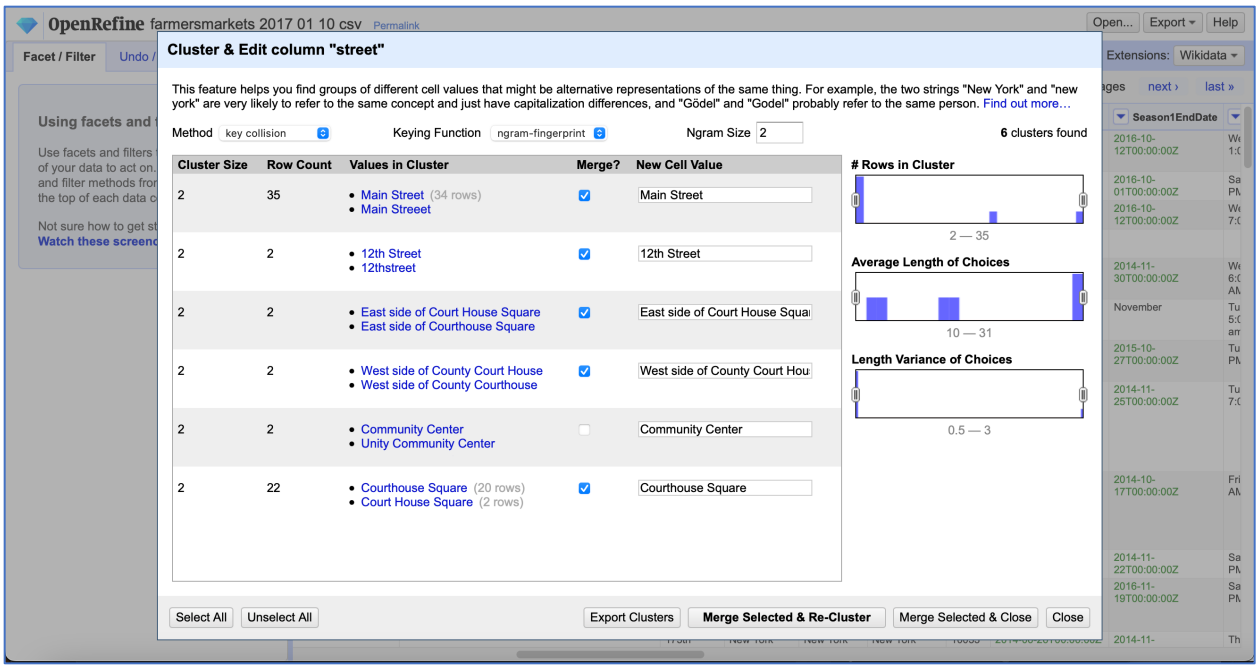


Figure 4 OpenRefine: Cleaning Street column

City:

- Trim leading and trailing white space.
- Collapse consecutive white spaces.
- Use text facet and cluster by using key collision method and fingerprint keying function.

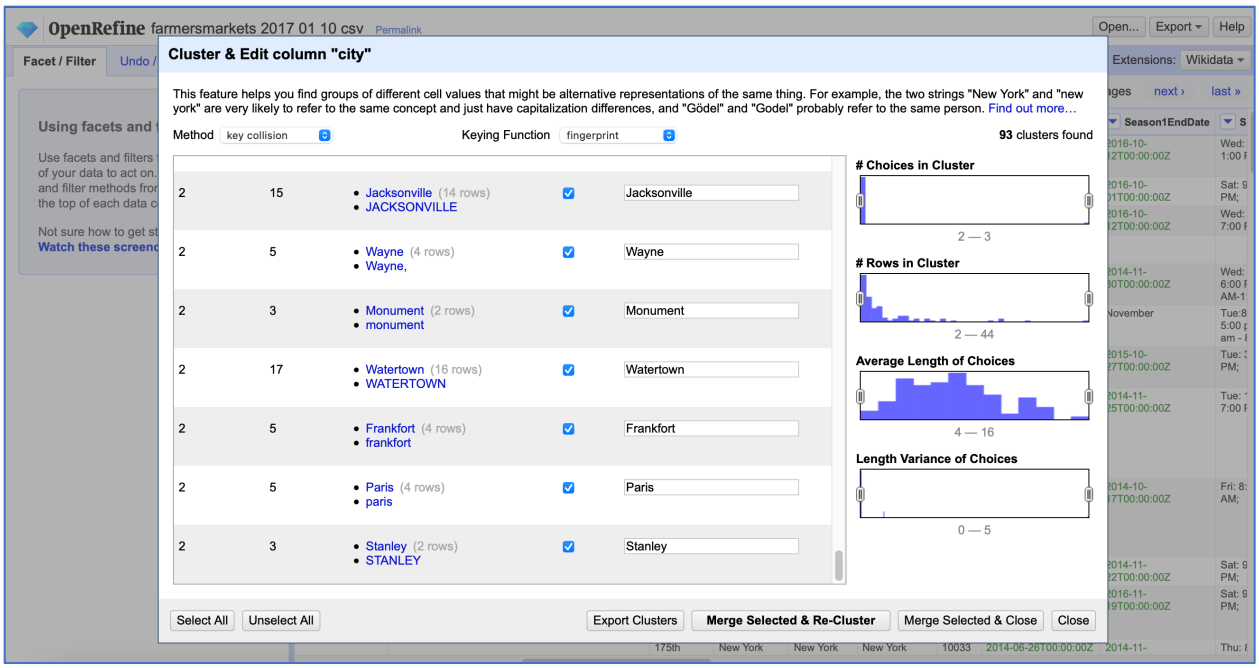


Figure 5 OpenRefine: Cleaning City

- Use text facet and cluster again by using key collision method and ngram-fingerprint

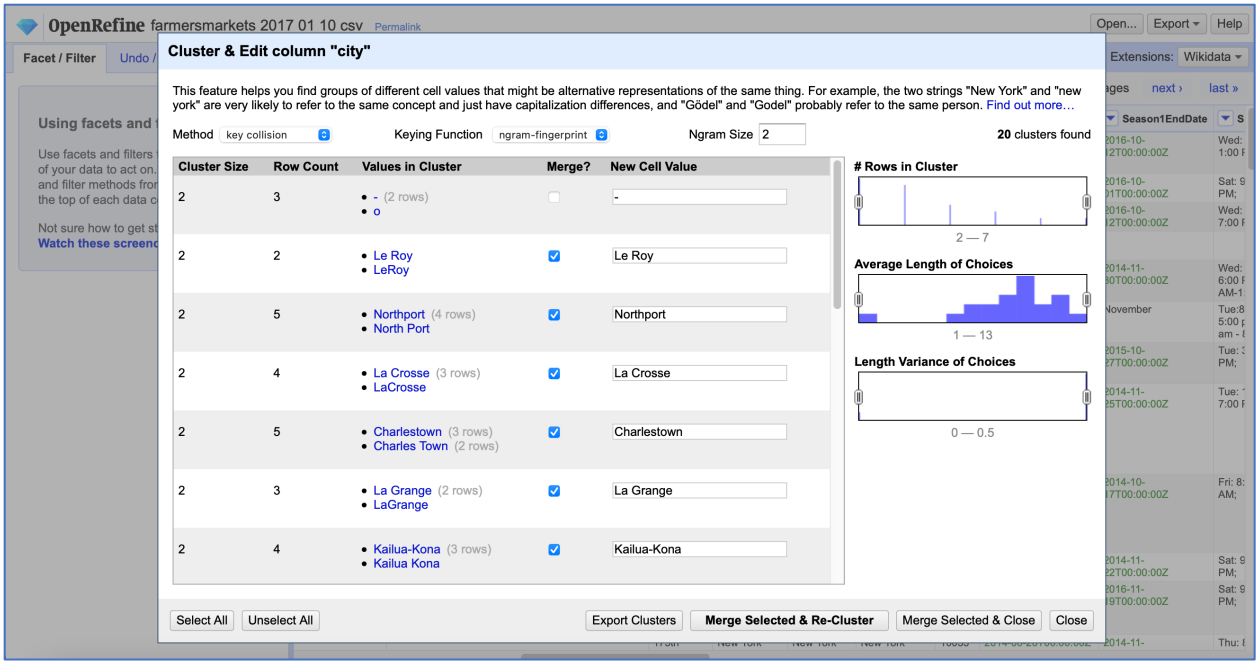


Figure 6 OpenRefine: Cleaning City column

County:

- Trim leading and trailing white space.
- Collapse consecutive white spaces.
- Use text facet and cluster by using key collision method and fingerprint keying function.
- Use text facet and cluster again by using key collision method and ngram-fingerprint

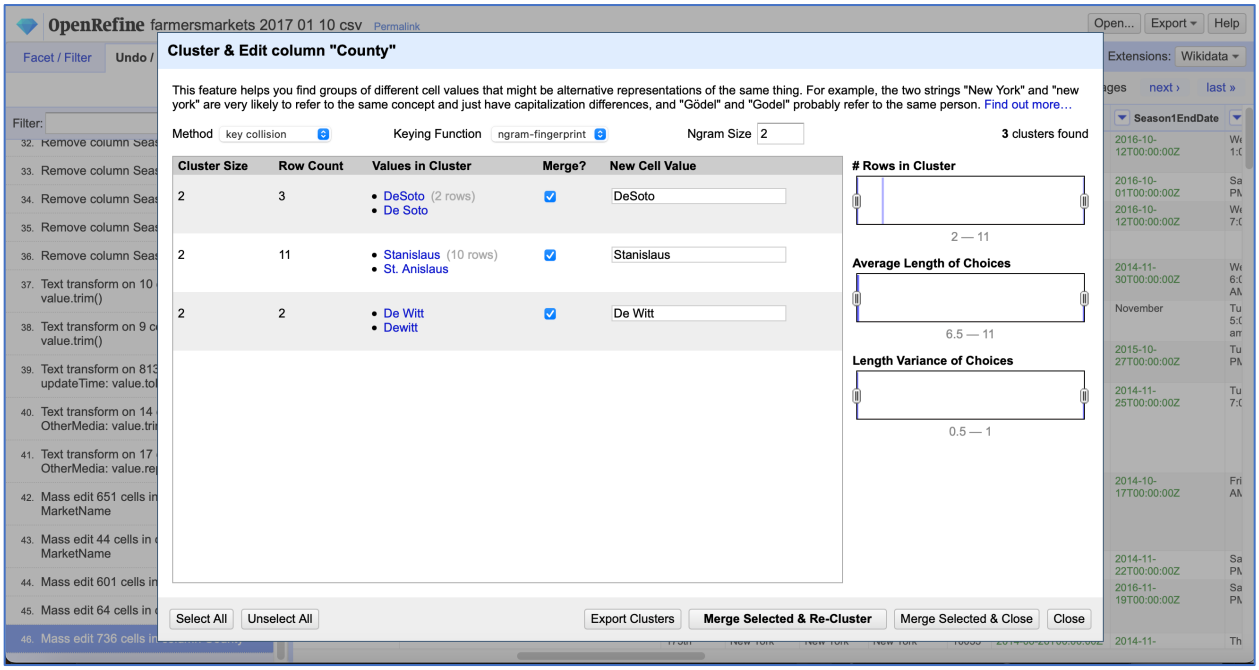


Figure 7 OpenRefine: Cleaning County column

Zip Code:

- Trim leading and trailing white space.
- Collapse consecutive white spaces.
- Convert any string values to blank using GREL.

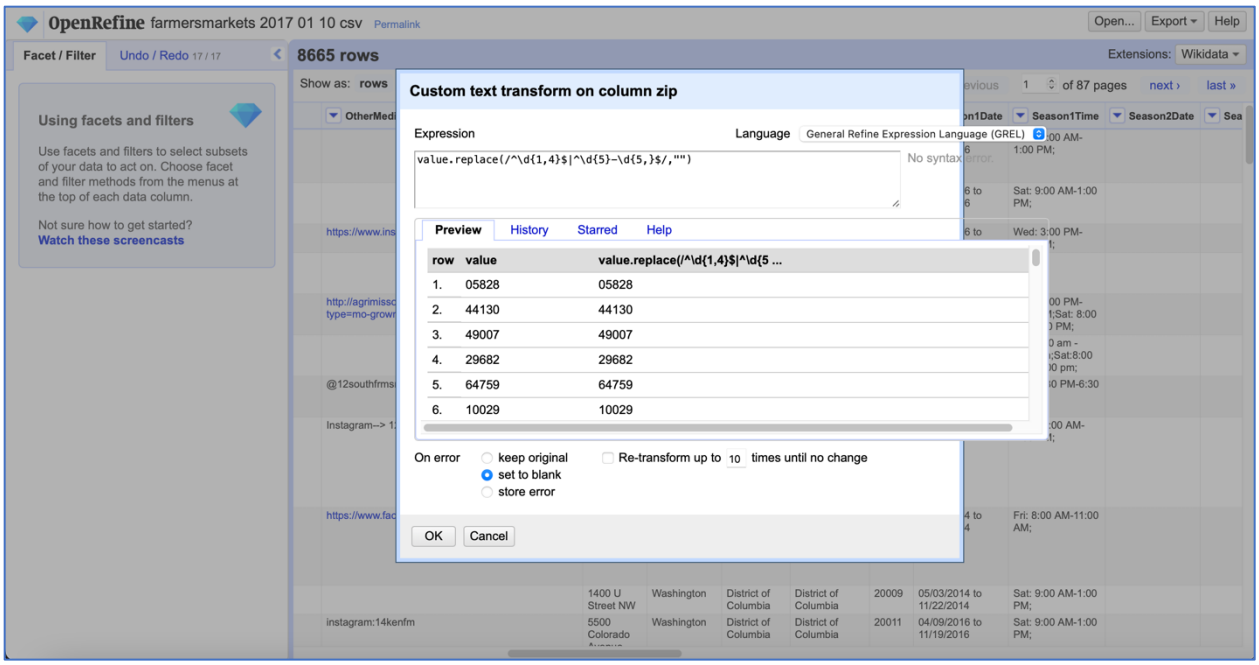


Figure 8 OpenRefine: GREL to fix the zip column

Website

- Trim leading and trailing white space.
- Use text facet and cluster by using key collision method and fingerprint keying function.
- Use text facet and cluster again by using key collision method and ngram-fingerprint

Facebook

- Trim leading and trailing white space.
- Collapse consecutive white spaces.
- Eliminate ‘N/A’ or ‘none’ string values by using clustering function.
- Use text facet and cluster by using key collision method and fingerprint keying function.
- Use text facet and cluster again by using key collision method and ngram-fingerprint

Twitter

- Trim leading and trailing white space.
- Eliminate ‘N/A’ or ‘none’ string values by using clustering function.
- Use text facet and cluster by using key collision method and fingerprint keying function.

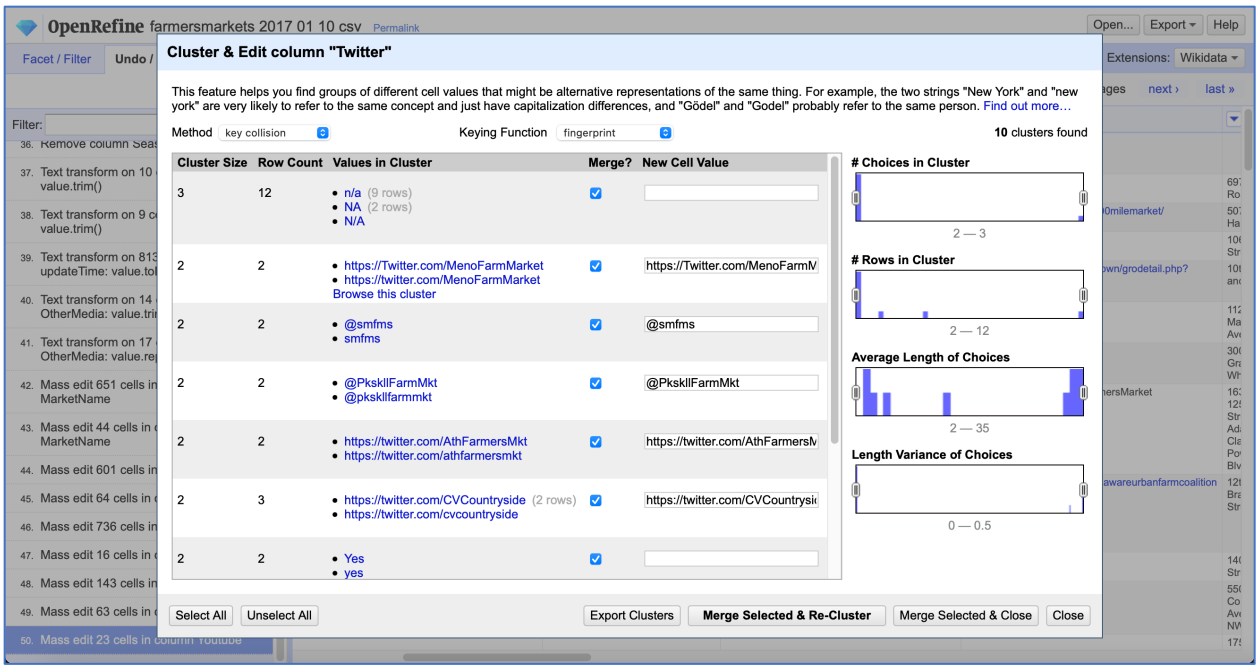


Figure 9 OpenRefine cleaning Twitter handle

Youtube

- Trim leading and trailing white space.
- Eliminate ‘N/A’ or ‘none’ string values by using clustering function.

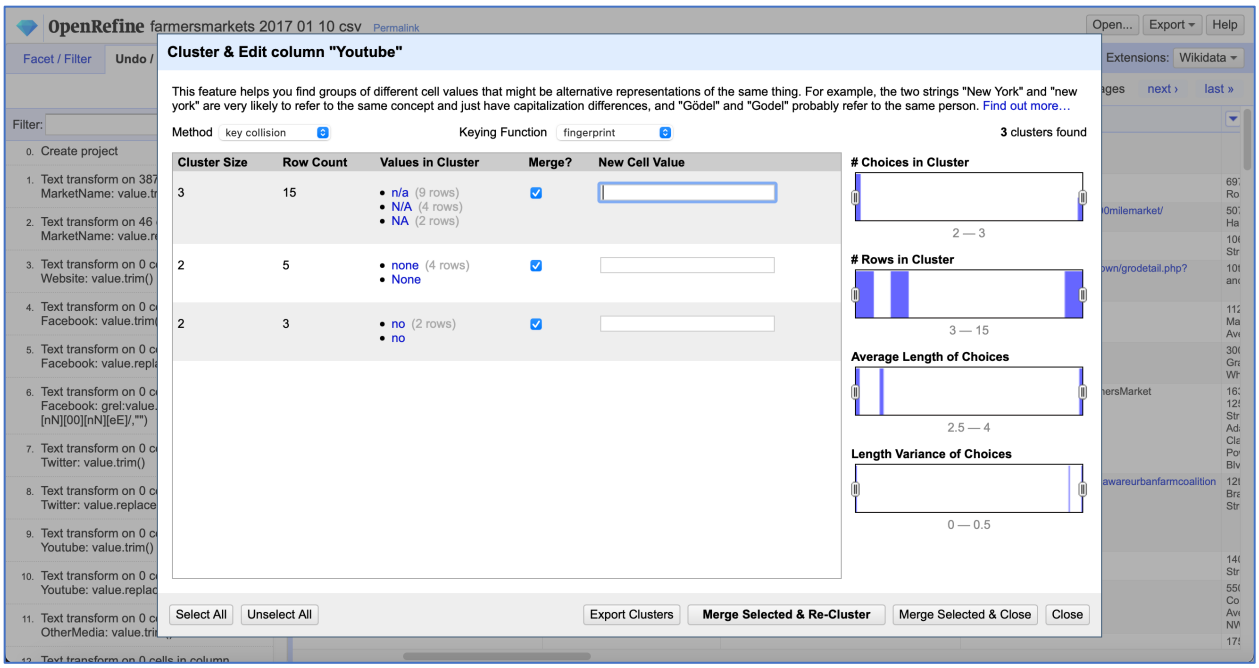


Figure 10 OpenRefine: cleaning youtube column

OtherMedia

- Trim leading and trailing white space.
- Collapse consecutive white spaces.

Season1Date

- Trim leading and trailing white space.
- Collapse consecutive white spaces.
- For better use and clear view, split into two columns: one is start date “Season1StartDate”, and the another is end date “Season1EndDate”.

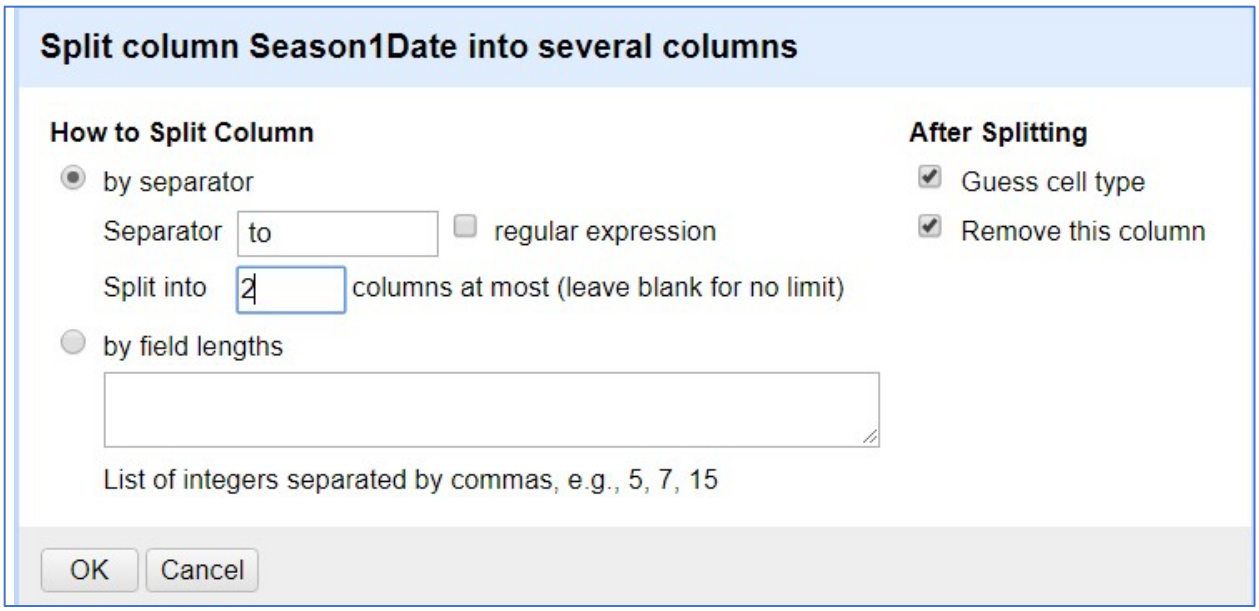


Figure 11 OpenRefine cleaning Season1Date column

Season1StartDate

- Change the date type to date

Season1EndDate

- Change the date type to date

Season2,3,4Date and Season2,3,4Time

- Considering that about 95% of these columns are blank, drop the columns for decreasing sparsity of data.

x
- Trim leading and trailing white space.

y
- Trim leading and trailing white space.

updatedTime

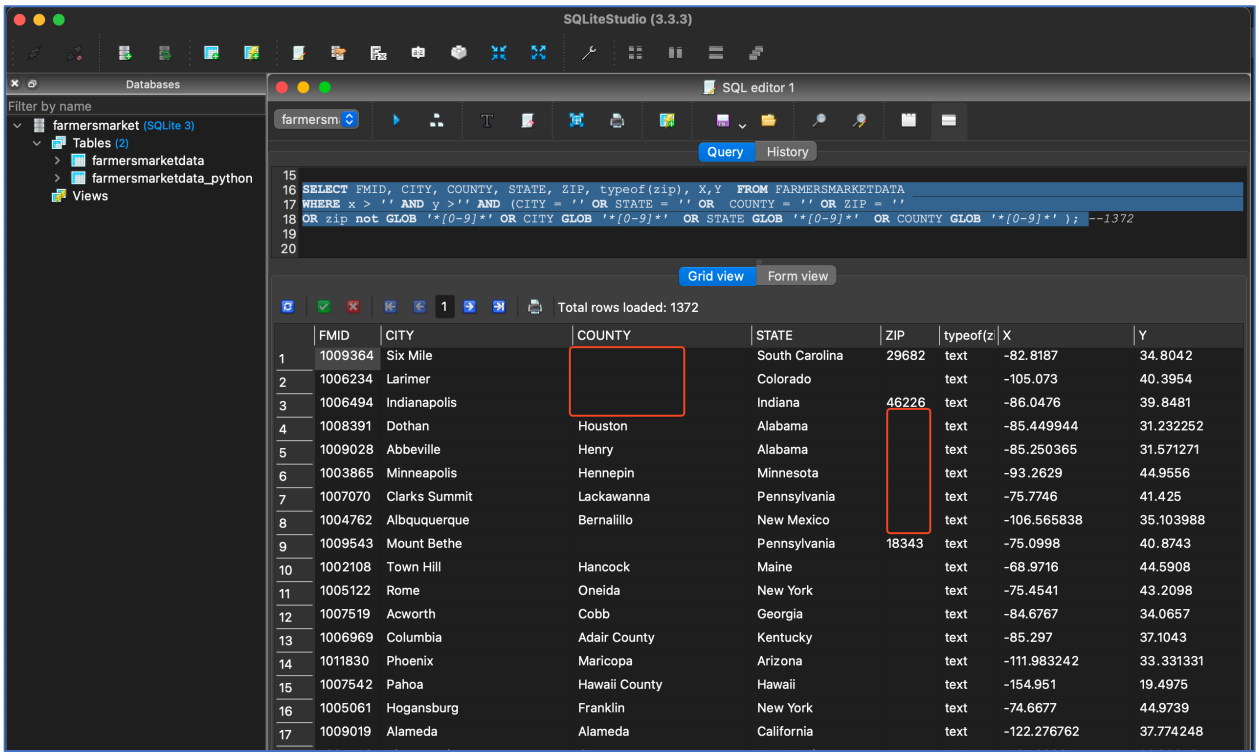
Change the date type to date

Data Correction using Python

OpenRefine does a great job on fixing the incorrect data which are captured due to typo, incorrect formats, and any other human negligence. We still need populate the missing data which can be inferred from other columns.

The project team noted that several records are missing zipcode, city and county information. To answer the use case to find the location of farmers market, we must have state, county, city and zipcode available in the clean dataset.

Below is the raw dataset we target to clean. Note the dataset has missing zipcode and county information. Interestingly we can see that the Longitude(x) and Latitude(y) information is available.



	FMID	CITY	COUNTY	STATE	ZIP	typeof(z	X	Y
1	1009364	Six Mile		South Carolina	29682	text	-82.8187	34.8042
2	1006234	Larimer		Colorado		text	-105.073	40.3954
3	1006494	Indianapolis		Indiana	46226	text	-86.0476	39.8481
4	1008391	Dothan	Houston	Alabama		text	-85.449944	31.232252
5	1009028	Abbeville	Henry	Alabama		text	-85.250365	31.571271
6	1003865	Minneapolis	Hennepin	Minnesota		text	-93.2629	44.9556
7	1007070	Clarks Summit	Lackawanna	Pennsylvania		text	-75.7746	41.425
8	1004762	Albququerque	Bernalillo	New Mexico		text	-106.565838	35.103988
9	1009543	Mount Bethe		Pennsylvania	18343	text	-75.0998	40.8743
10	1002108	Town Hill	Hancock	Maine		text	-68.9716	44.5908
11	1005122	Rome	Oneida	New York		text	-75.4541	43.2098
12	1007519	Acworth	Cobb	Georgia		text	-84.6767	34.0657
13	1006969	Columbia	Adair County	Kentucky		text	-85.297	37.1043
14	1011830	Phoenix	Maricopa	Arizona		text	-111.983242	33.331331
15	1007542	Pahoa	Hawaii County	Hawaii		text	-154.951	19.4975
16	1005061	Hogansburg	Franklin	New York		text	-74.6677	44.9739
17	1009019	Alameda	Alameda	California		text	-122.276762	37.774248

Figure 12 Dataset showing missing Zip and County

The above missing data will be cleaned using Python and below are the major steps involved:

1. Find farmer's market missing key address fields missing but have Longitude(x) and Latitude(y) populated based on the following criteria:
    - Either City, County, State, Zip is Blank.
    - Either City, County, State has a Number in it.
    - The Zip has alphabets in it.
    - The Zip codes are not in DDDDD or DDDDD-DDDD format.

The logical construct of the populating missing zip code using Python's uszipcode/Searchengine is as below:

1. Populate the missing zipcodes based on the Longitude(x) and Latitude(y).
2. Populate the missing City, County and State based on the Longitude(x) and Latitude(y).
3. Rescan the data if there still exists any missing State, City, Zip.
    1. Use the populated zip code and use it to find and populate County.
    2. Use the populated City to populate the zipcode which were not available through using Longitude and Latitude(Farmer's market data is Alaska showing missing information).

Below is the YesWorkflow for Python's uszipcode/searchengine data cleaning script.

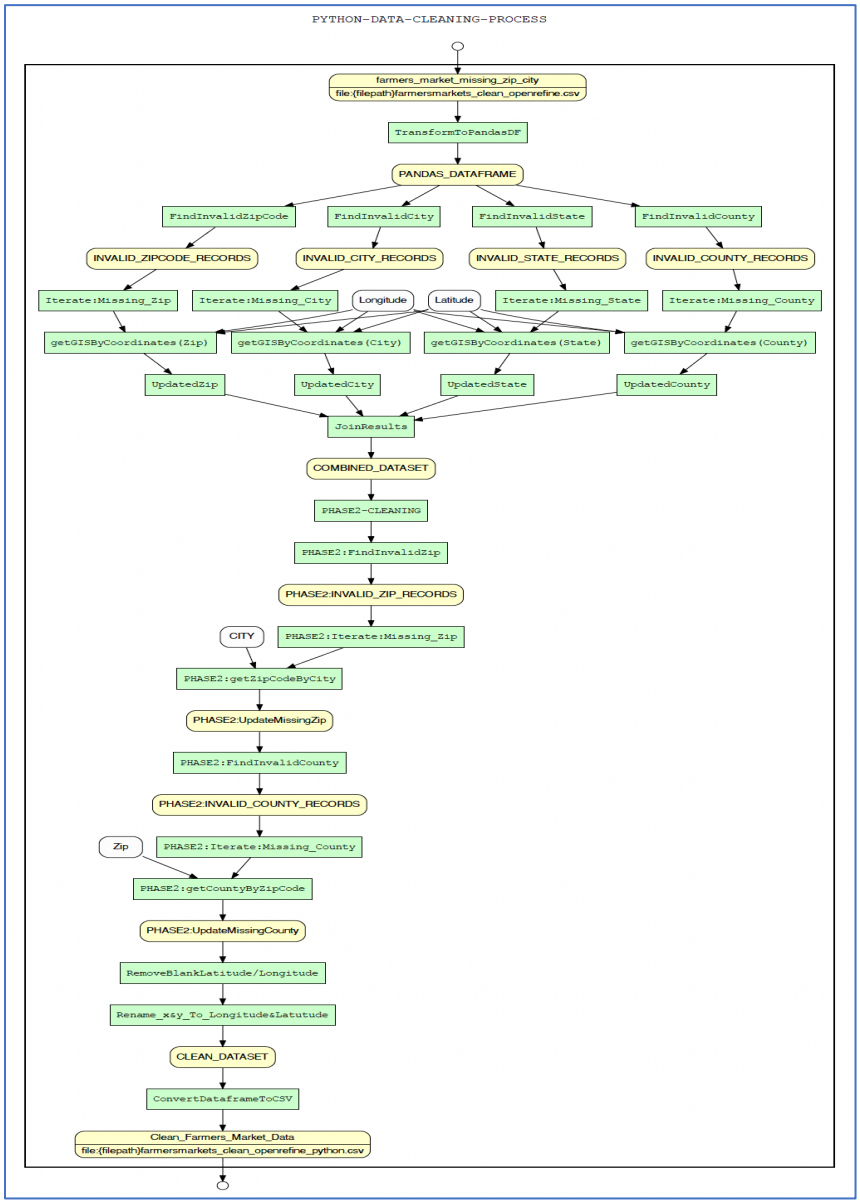
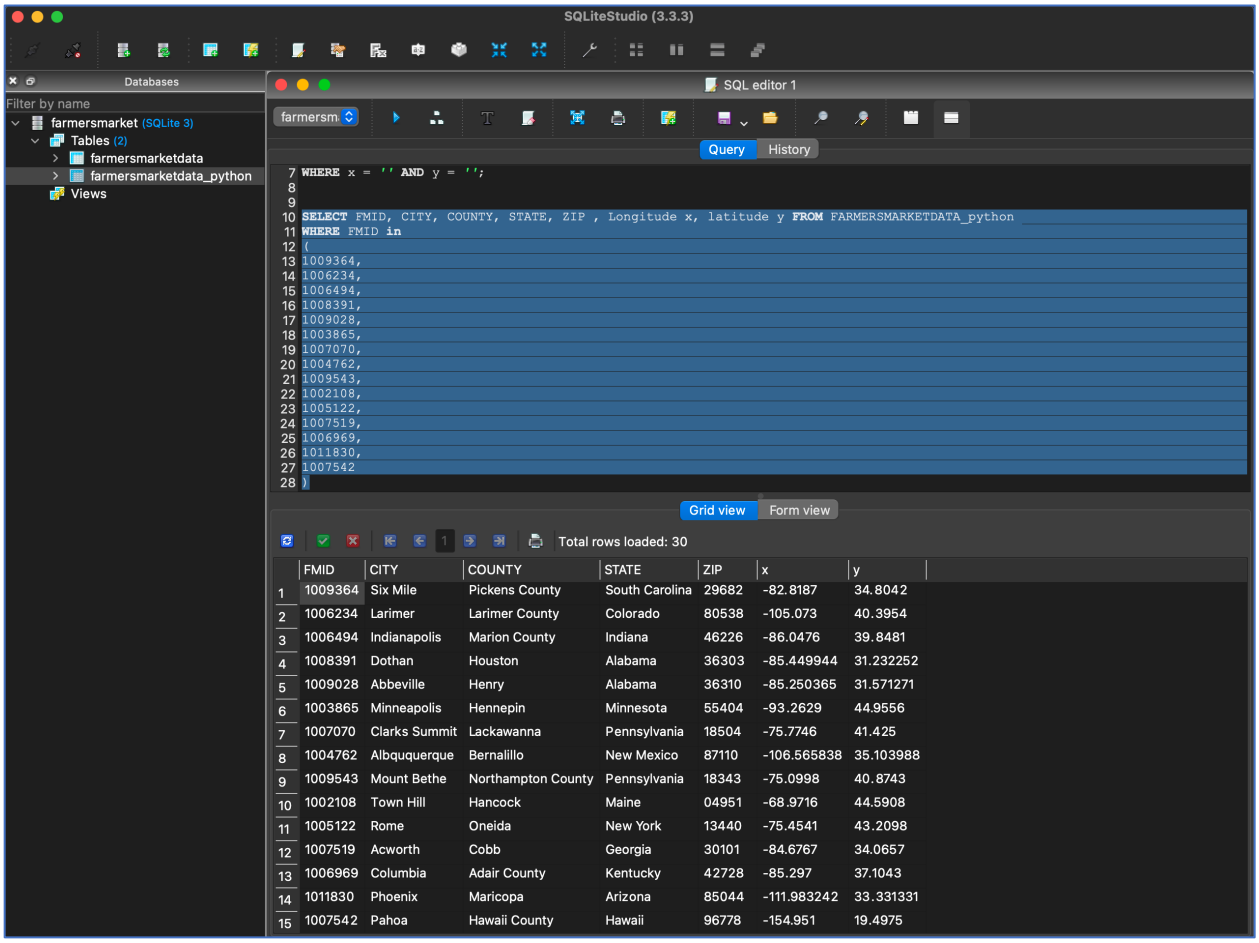


Figure 13 YesWorkFlow showing how Python script is used to populate missing zip, city and county.

The result of data correction is as below.



	FMID	CITY	COUNTY	STATE	ZIP	x	y
1	1009364	Six Mile	Pickens County	South Carolina	29682	-82.8187	34.8042
2	1006234	Larimer	Larimer County	Colorado	80538	-105.073	40.3954
3	1006494	Indianapolis	Marion County	Indiana	46226	-86.0476	39.8481
4	1008391	Dothan	Houston	Alabama	36303	-85.449944	31.232252
5	1009028	Abbeville	Henry	Alabama	36310	-85.250365	31.571271
6	1003865	Minneapolis	Hennepin	Minnesota	55404	-93.2629	44.9556
7	1007070	Clarks Summit	Lackawanna	Pennsylvania	18504	-75.7746	41.425
8	1004762	Albququerque	Bernalillo	New Mexico	87110	-106.565838	35.103988
9	1009543	Mount Bethe	Northampton County	Pennsylvania	18343	-75.0998	40.8743
10	1002108	Town Hill	Hancock	Maine	04951	-68.9716	44.5908
11	1005122	Rome	Oneida	New York	13440	-75.4541	43.2098
12	1007519	Acworth	Cobb	Georgia	30101	-84.6767	34.0657
13	1006969	Columbia	Adair County	Kentucky	42728	-85.297	37.1043
14	1011830	Phoenix	Maricopa	Arizona	85044	-111.983242	33.331331
15	1007542	Pahoa	Hawaii County	Hawaii	96778	-154.951	19.4975

Figure 14 Populated Corrected sample set

Further, the farmer's market data missing Longitude(x) and Latitude(y) are removed.

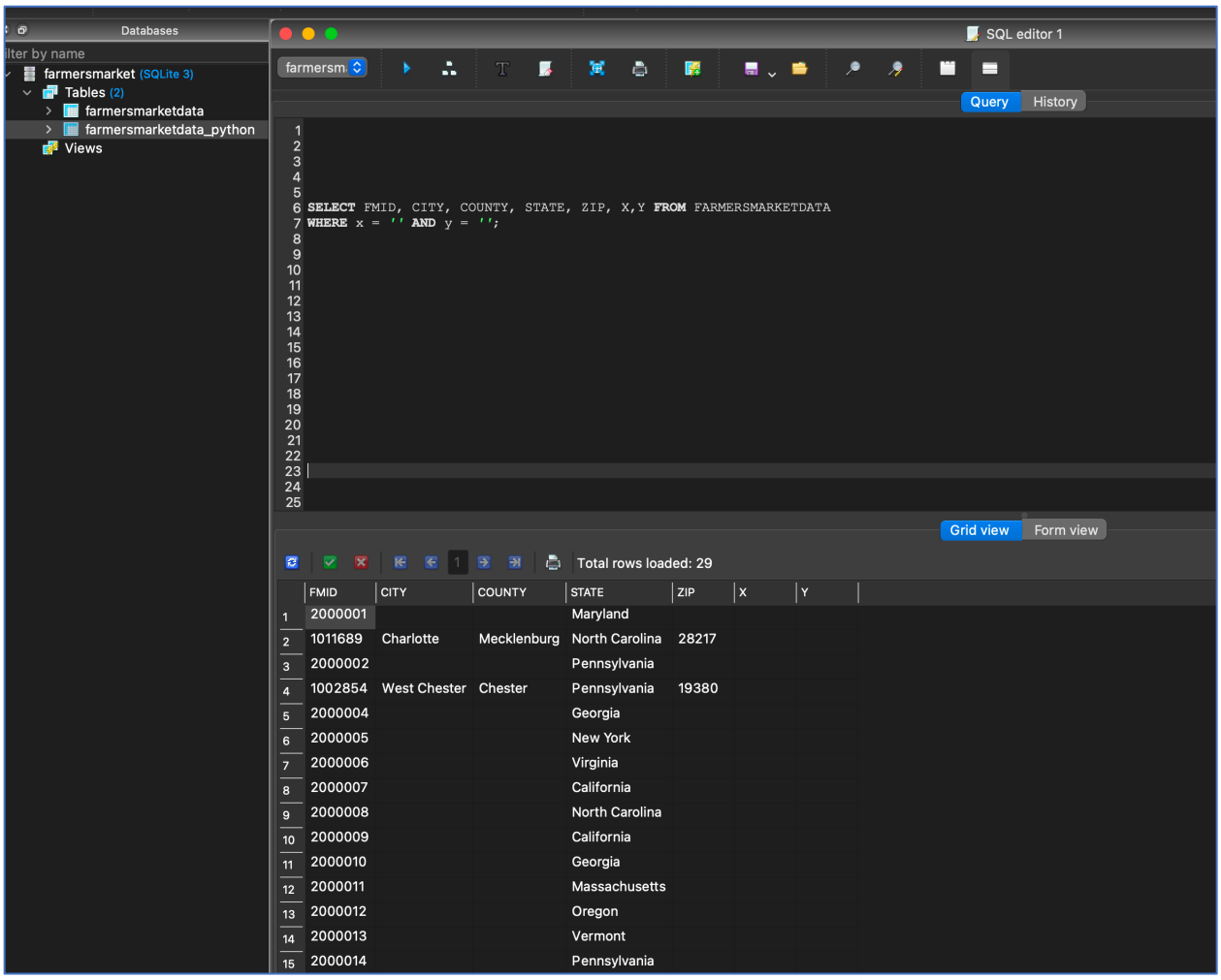


Figure 15 The Missing GeoLocation (x,y) are removed.

SQL – Integrity Constraints Check

1. FMID column which uniquely identifies a farmer's market should contain all unique values in the database.

```
sqlite> SELECT COUNT(distinct FMID) - COUNT(*) FROM farmersmarkets_clean_openrefine_python;
0
```

2. Latitude and Longitude columns should contain values in the range of –90 to +90 for latitude and –180 to +180 for longitude in the database.

```
sqlite> SELECT COUNT(*)
FROM   farmersmarkets_clean_openrefine_python m
WHERE  ( Cast(m.longitude AS FLOAT) > 180
           OR Cast(m.longitude AS FLOAT) <= 180
           OR Cast(m.latitude AS FLOAT) < -90
           OR Cast(m.latitude AS FLOAT) > 90 )
         OR ( m.latitude IS NULL
              OR m.longitude IS NULL
              OR m.latitude = ''
              OR m.longitude = '' );
0
```

3. Non-empty values for latitude and longitude, should have non-empty values for the corresponding city, state, county and zip in the database. Three invalid rows were found and were updated manually using the SQL Update command.

```
sqlite> SELECT m.FMID, m.MarketName, m.city, m.State, m.County, m.zip, m.latitude, m.longitude
FROM farmersmarkets_clean_openrefine_python m
WHERE (m.latitude != "" AND m.longitude!="") AND (m.zip ="" OR m.State==""  or m.County="" OR m.city=""
OR m.State is NULL or m.city is NULL OR m.County is NULL OR m.zip is NULL );

1001904 Denali Farmers Market        Anchorage    Alaska        NULL    NULL    62.3163 -150.234
1002348 Dillingham Farmers Market    Dillingham   Alaska        NULL    99576   59.0397 -158.458
1001903 Trapper Creek Farmers Market TrapperCreek Alaska        NULL    99683   53.8748 -166.54

UPDATE farmersmarkets_clean_openrefine_python
SET County = 'Anchorage',
    zip = '99645'
WHERE
    FMID='1001904';

UPDATE farmersmarkets_clean_openrefine_python
SET County = 'Dillingham'
WHERE
    FMID='1002348';

UPDATE farmersmarkets_clean_openrefine_python
SET County = 'Matanuska-Susitna'
WHERE
    FMID='1001903';
```

4. For all rows which have non-empty values for Season1StartDate and Season1EndDate, Season1EndDate should be greater than Season1EndDate. One invalid row was found and was updated using the SQL Update command.

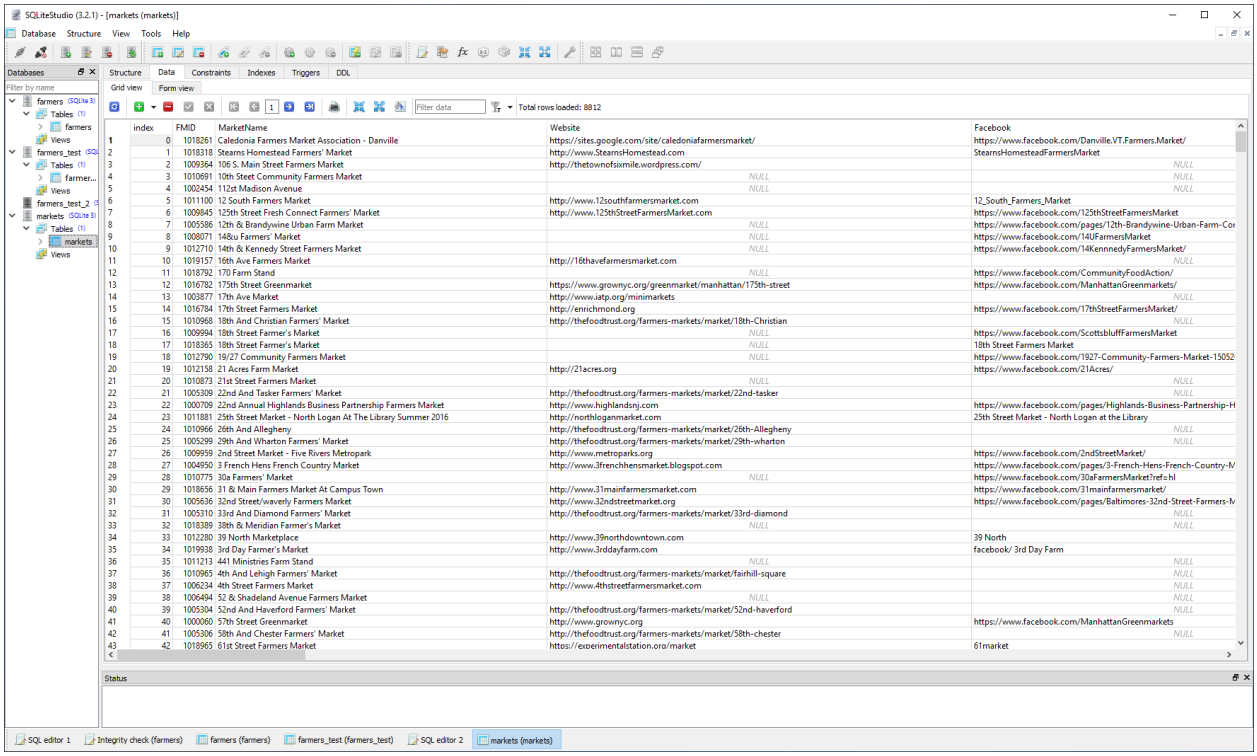
```
sqlite> SELECT m.FMID, m.MarketName, m.Season1StartDate, m.Season1EndDate,
JULIANDAY(m.Season1EndDate) - JULIANDAY(m.Season1StartDate) AS date_difference,
JULIANDAY(m.Season1EndDate) - JULIANDAY(m.Season1StartDate) + 1 AS days_inclusive
FROM farmersmarkets_clean_openrefine_python m
WHERE  (m.Season1StartDate != "" AND m.Season1EndDate!="") AND (date_difference < 0);

1011959 Clark Park Farmer's Market      2016-10-01      2016-05-07      -147.0  -146.0

UPDATE farmersmarkets_clean_openrefine_python
SET Season1StartDate =  Season1EndDate,
Season1EndDate = Season1StartDate
WHERE
    FMID='1011959'
```

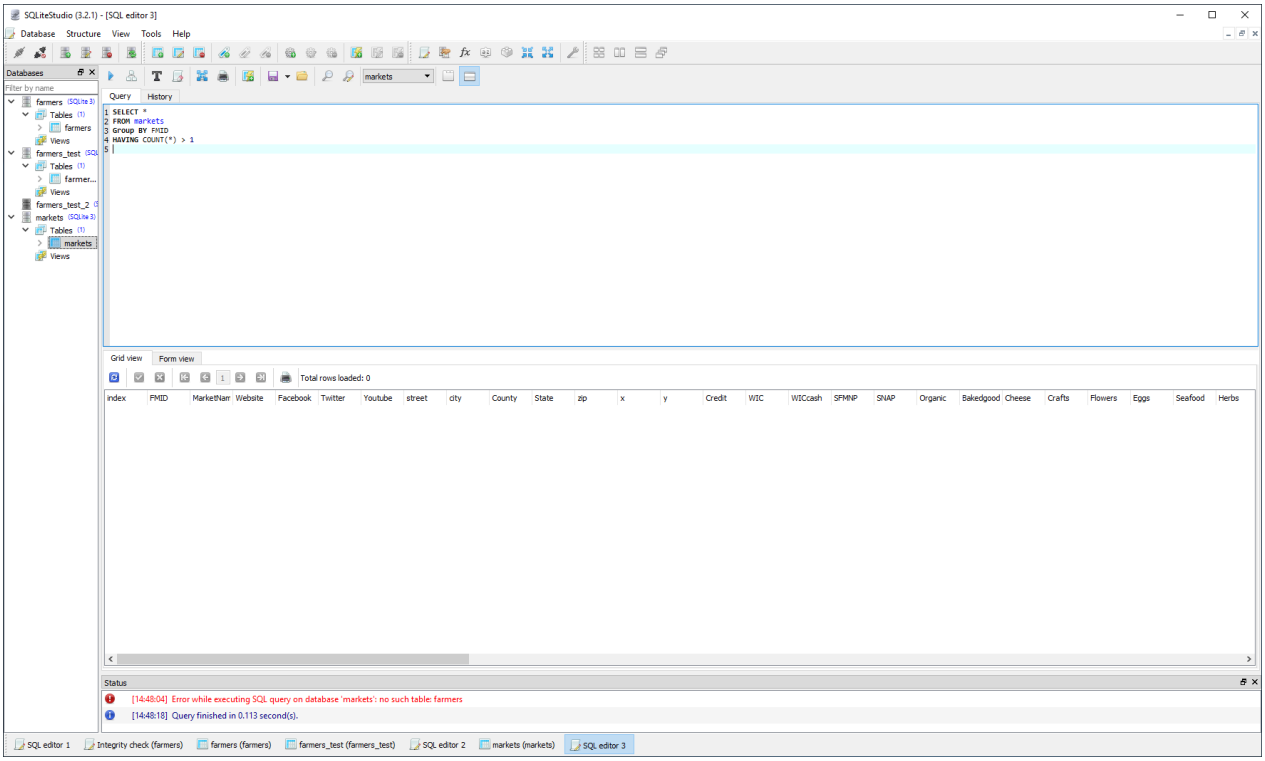
SQL – Query Result for Target Use Case U1

```
sqlite> SELECT
    m.FMID, m.MarketName, m.city,m.zip,(
      ROUND(3959 * acos (
      cos ( radians(37.783062) )
      * cos( radians( latitude ) )
      * cos( radians( longitude ) - radians(-122.460989) )
      + sin ( radians(37.783072) )
      * sin( radians( latitude ) )
    ),2)
) AS distance, m.Facebook, m.Organic, m.Credit, m.Season1StartDate, m.Season1EndDate, m.Season1Time
FROM farmersmarkets_clean_openrefine_python m
WHERE distance < 20 AND m.Facebook LIKE 'https://www.facebook.com%' AND m.Organic = 'Y' AND m.Credit = 'Y'
AND (strftime('%m',m.Season1StartDate) = '12' or strftime('%m',m.Season1EndDate) = '12') AND
(m.Season1Time LIKE '%Sat%' or m.Season1Time LIKE '%sat%' or m.Season1Time LIKE '%Sun%'
or m.Season1Time LIKE '%sun%')  ORDER BY distance ASC;
FMID     MarketName                                          city          zip    distance  Facebook                                                                                  Organic  Credit
-------  --------------------------------------------------  ------------  -----  --------  ----------------------------------------------------------------------------------------  -------  ------
1002741  Fort Mason Center Certified Farmers' Market         San Francisco  94123  1.59      https://www.Facebook.com/FMCFarmersMkt                                                    Y        Y
1005759  Heart of the City Farmers' Market                   San Francisco  94102  2.0       https://www.facebook.com/pages/Heart-of-the-City-Farmers-Market/164898116861011?ref=hl  Y        Y
1011656  Ferry Plaza Farmers Market                          San Francisco  94111  3.42      https://www.facebook.com/CUESA?fref=ts                                                    Y        Y
1012534  Stonestown Farmers Market                           San Francisco  94132  3.43      https://www.facebook.com/StonestownFarmersMarket/?fref=ts                                 Y        Y
1012074  CUESA Jack London Square Farmers Market             Oakland        94607  9.94      https://www.facebook.com/jacklondonsqfm/                                                  Y        Y
1009019  Alameda Farmers' Market                             Alameda        94501  9.95      https://www.facebook.com/AlamedaFarmersMarket                                             Y        Y
1012529  Grand Lake-Oakland Farmers Market                   Oakland        94610  11.68     https://www.facebook.com/GrandLakeFarmersMarket/                                          Y        Y
1012530  Sunday Civic Center Farmers Market                  San Rafael     94903  15.25     https://www.facebook.com/MarinFarmersMarkets/?fref=ts                                     Y        Y
1011348  Pinole Farmers' Market                              Pinole         94564  17.9      https://www.facebook.com/PinoleFarmersMarket                                              Y        Y
1002743  Moraga Certified Farmers Market                     Moraga         94556  18.47     https://www.facebook.com/MHFarmersMkt                                                     Y        Y
1002736  San Leandro Certified Farmers' Market at Bayfair Center  San Leandro  95678  19.11     https://www.facebook.com/BayfairFarmersMkt                                                Y        Y
sqlite>
```

SQL – Query Result for Target Use Case U0

```
sqlite> SELECT
 m.FMID, m.MarketName, m.City, m.State, m.Credit
FROM
        farmersmarkets_clean_openrefine_python m
WHERE m.city = 'San Francisco' AND m.State = 'California' AND m.Credit = 'Y'
LIMIT 5;
FMID     MarketName                                  city           State       Credit
-------  ------------------------------------------  -------------  ----------  ------
1011460  Castro Certified Farmers' Market            San Francisco  California  Y
1012527  Clement St. Farmers Market                  San Francisco  California  Y
1011426  Divisadero Certified Farmers' Market        San Francisco  California  Y
1011656  Ferry Plaza Farmers Market                  San Francisco  California  Y
1002741  Fort Mason Center Certified Farmers' Market  San Francisco  California  Y
sqlite>
```


SQLiteStudio (3.2.1) - [markets (markets)]
Database Structure View Tools Help
Structure Data Constraints Indexes Triggers DDL
Grid view Form view
Total rows loaded: 8812
index FMID MarketName Website Facebook
0 1018261 Caledonia Farmers Market Association - Danville https://sites.google.com/site/caledoniafarmersmarket/ https://www.facebook.com/Danville.VT.Farmers.Market/
1 1018318 Stearns Homestead Farmers' Market http://www.StearnsHomestead.com SteamsHomesteadFarmersMarket
2 1009364 106 S. Main Street Farmers Market http://thetownofsixmile.wordpress.com/ NULL
3 1010691 10th Steet Community Farmers Market NULL NULL
4 1002454 112st Madison Avenue NULL NULL
5 1011100 12 South Farmers Market http://www.12southfarmersmarket.com 12_South_Farmers_Market
6 1009845 125th Street Fresh Connect Farmers' Market http://www.125thStreetFarmersMarket.com https://www.facebook.com/125thStreetFarmersMarket
7 1005586 12th & Brandywine Urban Farm Market NULL https://www.facebook.com/pages/12th-Brandywine-Urban-Farm-Co
8 1008071 148cu Farmers' Market NULL https://www.facebook.com/14UFarmersMarket
9 1012710 14th & Kennedy Street Farmers Market NULL https://www.facebook.com/14KennnedyFarmersMarket/
10 1019157 16th Ave Farmers Market http://16thavefarmersmarket.com NULL
11 1018792 170 Farm Stand NULL https://www.facebook.com/CommunityFoodAction/
12 1016782 175th Street Greenmarket https://www.grownyc.org/greenmarket/manhattan/175th-street https://www.facebook.com/ManhattanGreenmarkets/
13 1003877 17th Ave Market http://www.iatp.org/minimarkets NULL
14 1016784 17th Street Farmers Market http://enrichmond.org https://www.facebook.com/17thStreetFarmersMarket/
15 1010968 18th And Christian Farmers' Market http://thefoodtrust.org/farmers-markets/market/18th-Christian NULL
16 1009994 18th Street Farmer's Market NULL https://www.facebook.com/ScottsbluffFarmersMarket
17 1018365 18th Street Farmer's Market NULL 18th Street Farmers Market
18 1012790 19/27 Community Farmers Market NULL https://www.facebook.com/1927-Community-Farmers-Market-15052
19 1012158 21 Acres Farm Market http://21acres.org https://www.facebook.com/21Acres/
20 1010873 21st Street Farmers Market NULL NULL
21 1005309 22nd And Tasker Farmers' Market http://thefoodtrust.org/farmers-markets/market/22nd-tasker NULL
22 1000709 22nd Annual Highlands Business Partnership Farmers Market http://www.highlandsnj.com https://www.facebook.com/pages/Highlands-Business-Partnership-H
23 1011881 25th Street Market - North Logan At The Library Summer 2016 http://northloganmarket.com 25th Street Market - North Logan at the Library
24 1010966 26th And Allegheny http://thefoodtrust.org/farmers-markets/market/26th-Allegheny NULL
25 1005299 29th And Wharton Farmers' Market http://thefoodtrust.org/farmers-markets/market/29th-wharton NULL
26 1009959 2nd Street Market - Five Rivers Metropark http://www.metroparks.org https://www.facebook.com/2ndStreetMarket/
27 1004950 3 French Hens French Country Market http://www.3frenchhensmarket.blogspot.com https://www.facebook.com/pages/3-French-Hens-French-Country-M
28 1010775 30a Farmers' Market NULL https://www.facebook.com/30aFarmersMarket?ref=hl
29 1018656 31 & Main Farmers Market At Campus Town http://www.31mainfarmersmarket.com https://www.facebook.com/31mainfarmersmarket/
30 1005636 32nd Street/waverly Farmers Market http://www.32ndstreetmarket.org https://www.facebook.com/pages/Baltimores-32nd-Street-Farmers-M
31 1005310 33rd And Diamond Farmers' Market http://thefoodtrust.org/farmers-markets/market/33rd-diamond NULL
32 1018389 38th & Meridian Farmer's Market NULL NULL
33 1012280 39 North Marketplace http://www.39northdowntown.com 39 North
34 1019938 3rd Day Farmer's Market http://www.3rddayfarm.com facebook/ 3rd Day Farm
35 1011213 441 Ministries Farm Stand NULL NULL
36 1010965 4th And Lehigh Farmers' Market http://thefoodtrust.org/farmers-markets/market/fairhill-square NULL
37 1006234 4th Street Farmers Market http://www.4thstreetfarmersmarket.com NULL
38 1006494 52 & Shadeland Avenue Farmers Market NULL NULL
39 1005304 52nd And Haverford Farmers' Market http://thefoodtrust.org/farmers-markets/market/52nd-haverford NULL
40 1000060 57th Street Greenmarket http://www.grownyc.org https://www.facebook.com/ManhattanGreenmarkets
41 1005306 58th And Chester Farmers' Market http://thefoodtrust.org/farmers-markets/market/58th-chester NULL
42 1018965 61st Street Farmers Market https://experimentalstation.org/market 61market
Status
SQL editor 1 Integrity check (farmers) farmers (farmers) farmers_test (farmers_test) SQL editor 2 markets (markets)


SQLiteStudio (3.2.1) - [SQL editor 3]
Database Structure View Tools Help
Query History
1 SELECT *
2 FROM markets
3 Group BY FMID
4 HAVING COUNT(*) > 1
5
Grid view Form view
Total rows loaded: 0
index FMID MarketNam Website Facebook Twitter Youtube street city County State zip x y Credit WIC WICcash SFMNP SNAP Organic Bakedgood Cheese Crafts Flowers Eggs Seafood Herbs
Status
[14:48:04] Error while executing SQL query on database 'markets': no such table: farmers
[14:48:18] Query finished in 0.113 second(s).
SQL editor 1 Integrity check (farmers) farmers (farmers) farmers_test (farmers_test) SQL editor 2 markets (markets) SQL editor 3

Workflow Diagram

We have translated openrefine JSON recipe to YesWorkflow diagram using OR2YWTool.

Here are the steps:

(base) Goutams-MBP:ProjectWork gdebnath$ **pip install or2ywtool**
….
….
Successfully installed or2ywtool-0.0.16

(base) Goutams-MBP:ProjectWork gdebnath$ **or2yw**
usage: or2yw [-h] [-i INPUT] [-o OUTPUT] [-t TYPE] [-ot OUTPUTTYPE]
[-java JAVA] [-dot DOT] [-title TITLE] [-desc DESCRIPTION]

OR2YW v0.0.16

optional arguments:
-h, --help show this help message and exit
-i INPUT, --input INPUT
openrefine json file
-o OUTPUT, --output OUTPUT
yesworkflow output file
-t TYPE, --type TYPE Workflow Type, Produce [serial,parallel,merge]
workflow, Default: serial
-ot OUTPUTTYPE, --outputtype OUTPUTTYPE
Output Type, Produce output [yw,gv,png,svg,pdf],
Default: yw
-java JAVA, --java JAVA
Java Path, if not initialized will use the java
installation environment path
-dot DOT, --dot DOT Dot Path, if not initialized will use the dot
installation environment path
-title TITLE, --title TITLE
Title for the Workflow
-desc DESCRIPTION, --description DESCRIPTION
Description for the Workflow

(base) Goutams-MBP:ProjectWork gdebnath$ **or2yw -i**
Data-Cleaning-cs513-Phase-2-Jul_12.docx
Data-Cleaning-cs513-Phase-2.docx
FarmersMarketDataCleaning.py
OpenRefineHistory.json

farmersmarkets-2017-01-10.csv
farmersmarkets_clean_openrefine.csv
farmersmarkets_clean_openrefine_python.csv
yw-editor-app-0.2.1.2-SNAPSHOT.jar

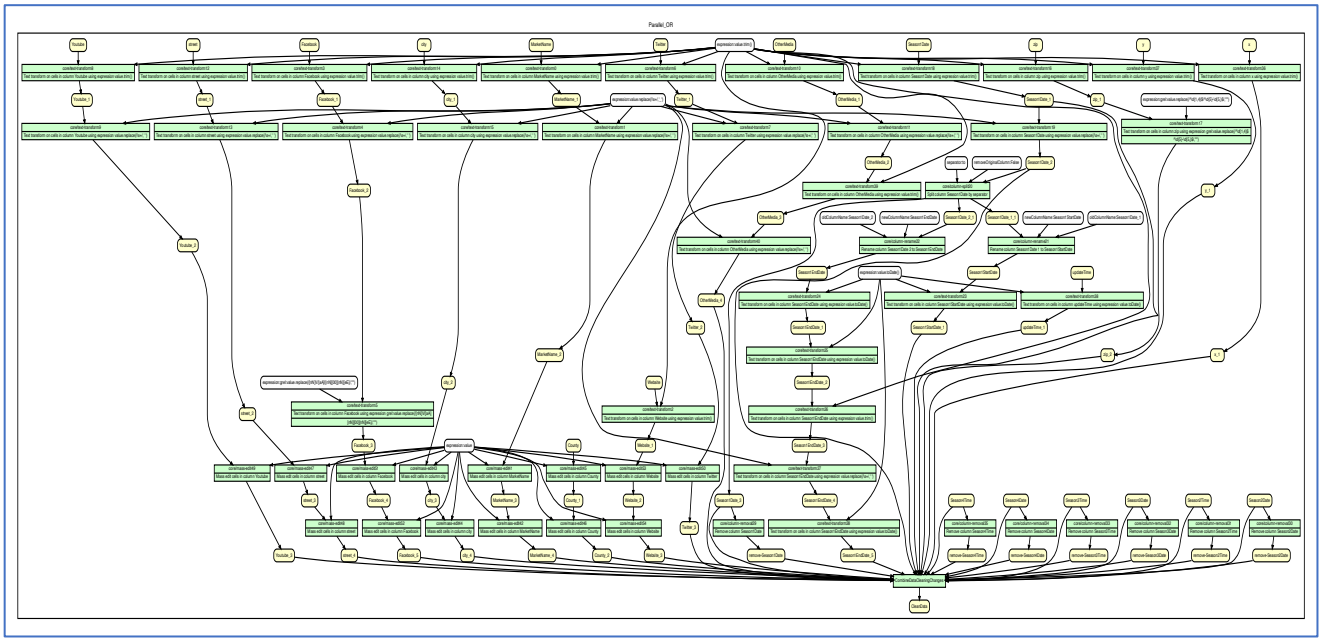
(base) Goutams-MBP:ProjectWork gdebnath$ **or2yw -i OpenRefineHistory.json -o** OpenRefineHistory.yw -t parallel
File OpenRefineHistory.yw generated.

(base) Goutams-MBP:ProjectWork gdebnath$ **dot -V**
dot - graphviz version 3.0.0 (20220226.1711)

(base) Goutams-MBP:ProjectWork gdebnath$ **or2yw -i OpenRefineHistory.json -ot pdf -o OpenRefineHistory.pdf -t parallel**
java found: java
dot found: dot
File OpenRefineHistory.pdf generated.

(base) Goutams-MBP:ProjectWork gdebnath$ **or2yw -i OpenRefineHistory.json -ot png -o OpenRefineHistory.png -t parallel**
java found: java
dot found: dot
File OpenRefineHistory.png generated.

Please find the Generated Workflow Diagram as below:

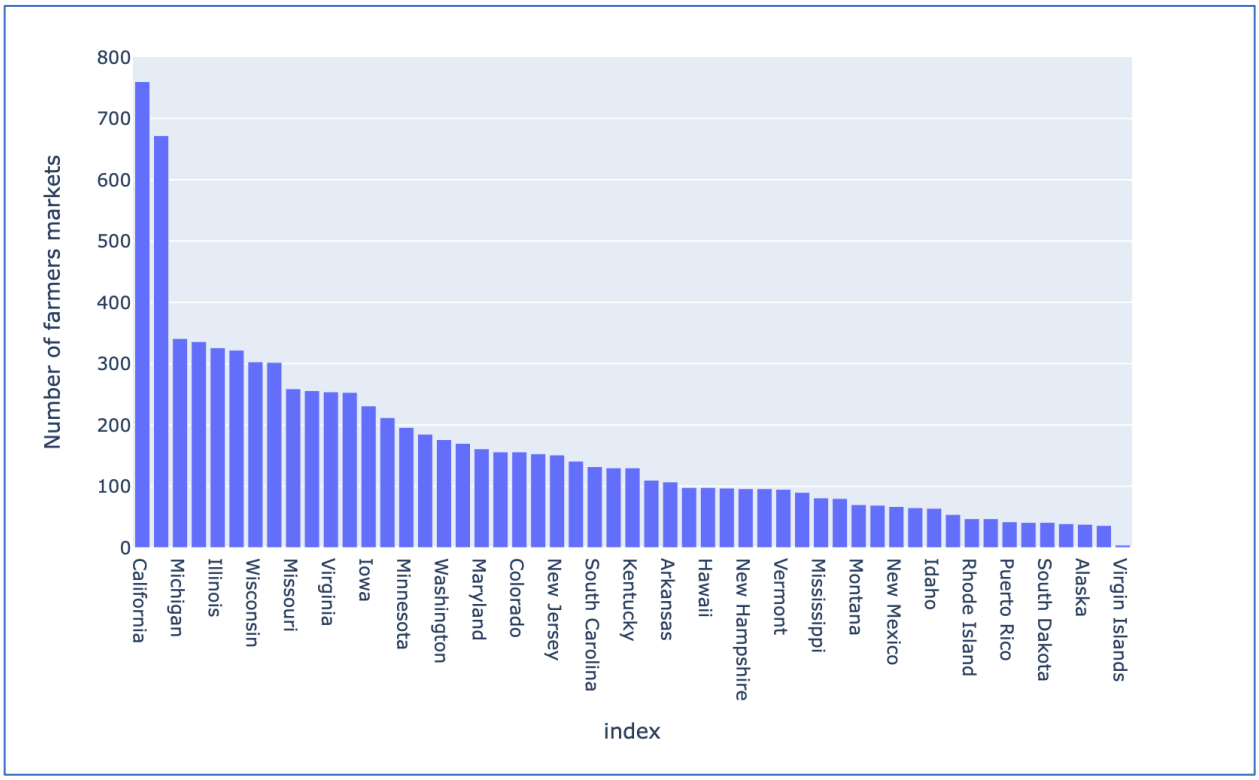


We will share the generated yw/pdf/png files as part of the supplemental package.

Integrity Constraints Check

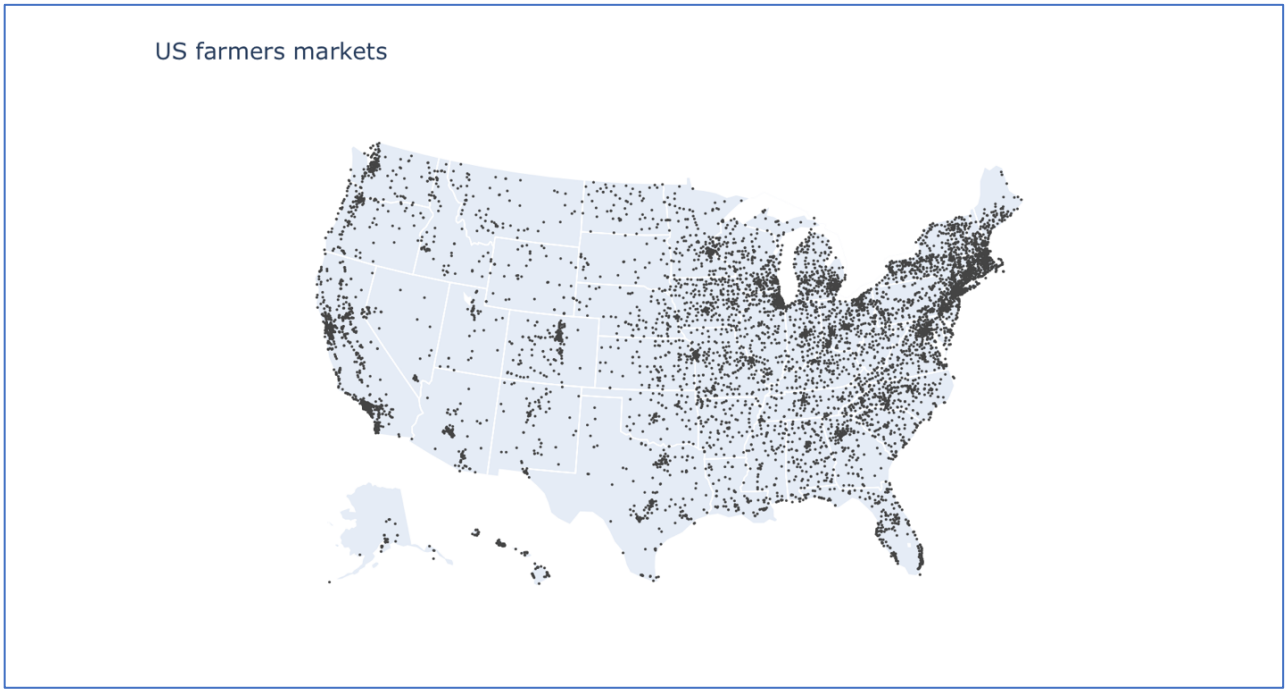
Exploring the distribution of farmers markets in the US.

Used Python express library from Plotly on the cleaned dataset and below bar plot shows **Number of farmers market per State**



California and Michigan has most number of farmers markets whereas Virgin island has the least. This gives us a good indication of market availability for the states we are interested in.

Used Python graph_objects library from Plotly to plot the farmers market locations in US map.



Observations:

- We can see that the southern region has a relatively moderate density of farmers markets throughout, with the states further to the east having a higher density.
- The mid-west states have a relatively low density of farmers markets throughout, however the states further to the east increase in density.
- The farmers markets of the western states are concentrated along the coastal states, with quite a low density across the rest of the region.

- The north-eastern states have the highest density of farmers markets throughout all of the states.