Team 140 Final Report

Table of Contents

Phase I Report 2
- Dataset 2
- Target Use Cases 2
  - $U1$ - Main Use Case 2
  - $U0$ - No Data Cleaning Required 2
  - $U2$ - Data Clearning not Sufficient 2
- Dataset Details 2
- Data Quality Problems 3
- Initial Plan 7
  - Cleaning Steps 7

Phase II Report 8
- Data cleaning performed 8
  - Cleaning Menu.csv file 8
  - Cleaning Dish.csv file 18
- Data quality changes 22
  - Summary of changes 22
  - Before and After 24
  - ICV Queries 25
- Workflow Model 29
  - Workflow W0: 29
  - Workflow W1: Cleaning Menu.csv file 30
  - Workflow W2: Cleaning Dish.csv file 30
- Conclusions & Summary 31

Vikas Seth (**Team 140**)

vseth3@illinois.edu

1. Points received for initial Phase-1 submission: 96 (out of 100)
2. Our team chooses the following Phase-1 option:
   - [X] (A) No change to Phase-1 report
   - [] (B) Improved (or extended) Phase-1 report

Phase I Report

Dataset

I will be using NYPL Menu data set for the final project which is available at http://menus.nypl.org/. This is dataset of Restaurant Menu collection dating from 1840s collected by New York Public Liberary. This collection is one of the largest in the world, used by historians, chefs, noveists and every food enthusiasts. This collection mainly contains details about different Menus from the past along with details of dishes, price range, menu size, location, sponsor details of the Menus.

Target Use Cases

U_1 - Main Use Case

One of the interesting use case with this data-set could be about dishes. Like when a dish appeared for the first time and on which Menu, who was the sponsor and which year it appeared. For this we will need to clean this dataset to make sure that we have cleaned information about name of the dishes, Restaurants, sponsor and year information. Based on that we should be able to search this database to filter out this details for any dish.

U_0 - No Data Cleaning Required

To find out average number of dishes in Menu for each by each year or decade.

U_2 - Data Clearning not Sufficient

Origin Language of the Menu. This information is available for some of the menus in Notes field, but it is not available for all of them. Also, there is a language column. But this information is not available there as well. So, even if we try to clean the data under notes column we will not be able to find this information for all the Menus.

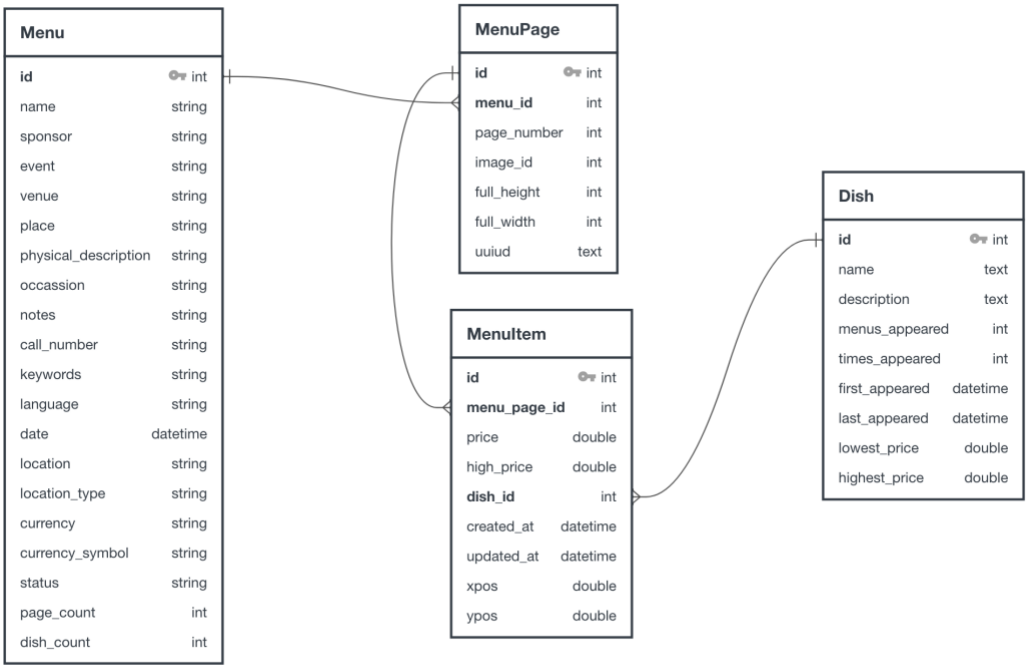
Dataset Details

This is data set for Menu details collected by NYPL with the help of Crowdsourcing. This menu collection go back to the year 1840 and contains details of Menus like dishes, occassion, sponsor, venue type, place, location, number of dishes along with physical properties of Menu like number of pages, bind type etc. This dataset mainly contains data in following 4 files

1. **Menu** - It contains overall details of each menu like Date, Sponsor, Event, Venue, Restaurant, Number of pages, Number of dishes in the Menu

2. **Menu Page** - Unique key in this set is Page id and for each Menu along with Page Number it contains Image details and dimensions of the Menu
3. **Menu Item** - Contains details of items/dishes, price, and dish id in a Menu
4. **Dish** - It contains details of the dish including name, description, number of occurrence in Menus, Occurrence timeline and price details.

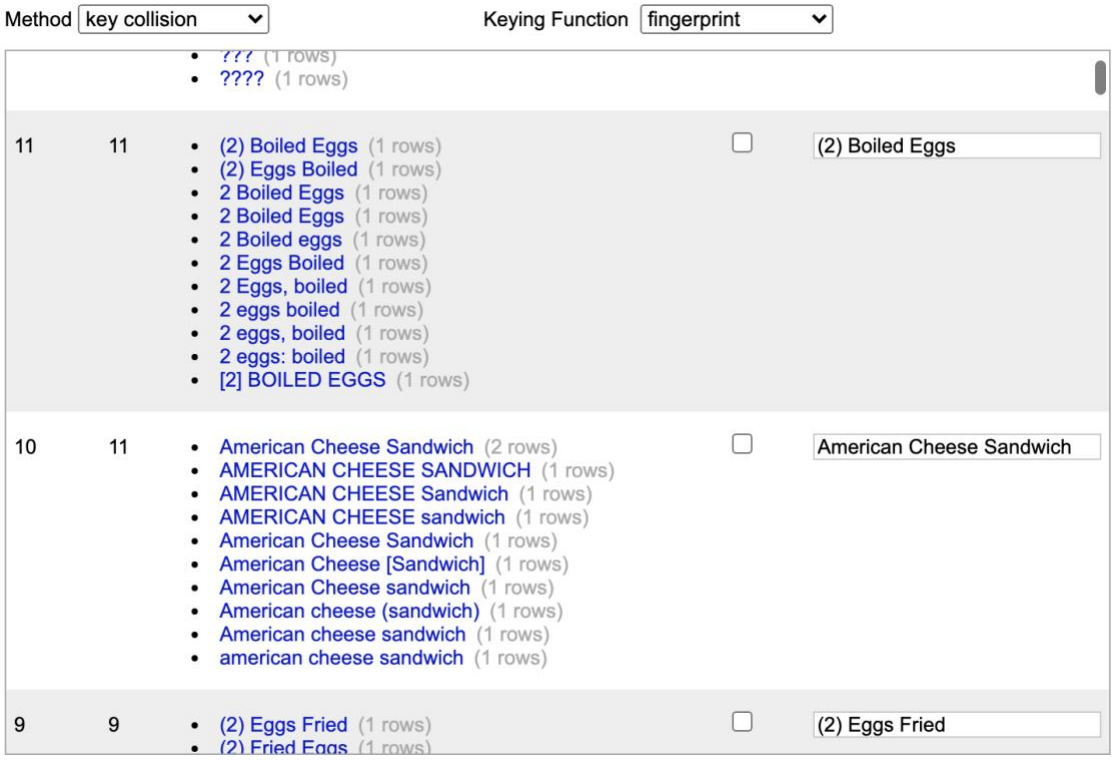
Basic database schema of this dataset is as follows



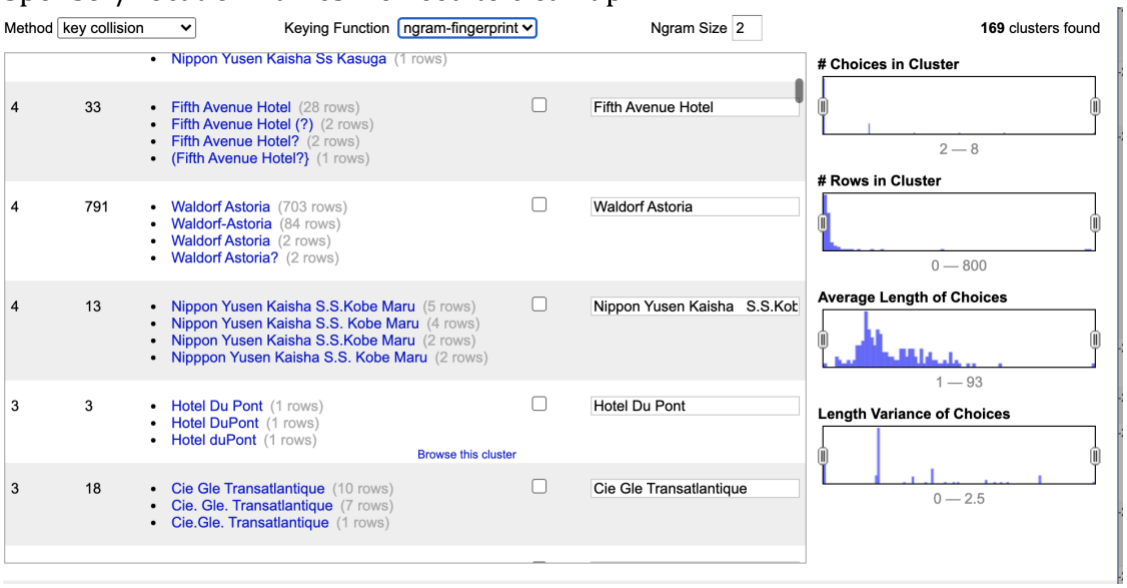
Data Quality Problems

At minimum we will need to address following data quality problems with this dataset to make it usable for use case U_1

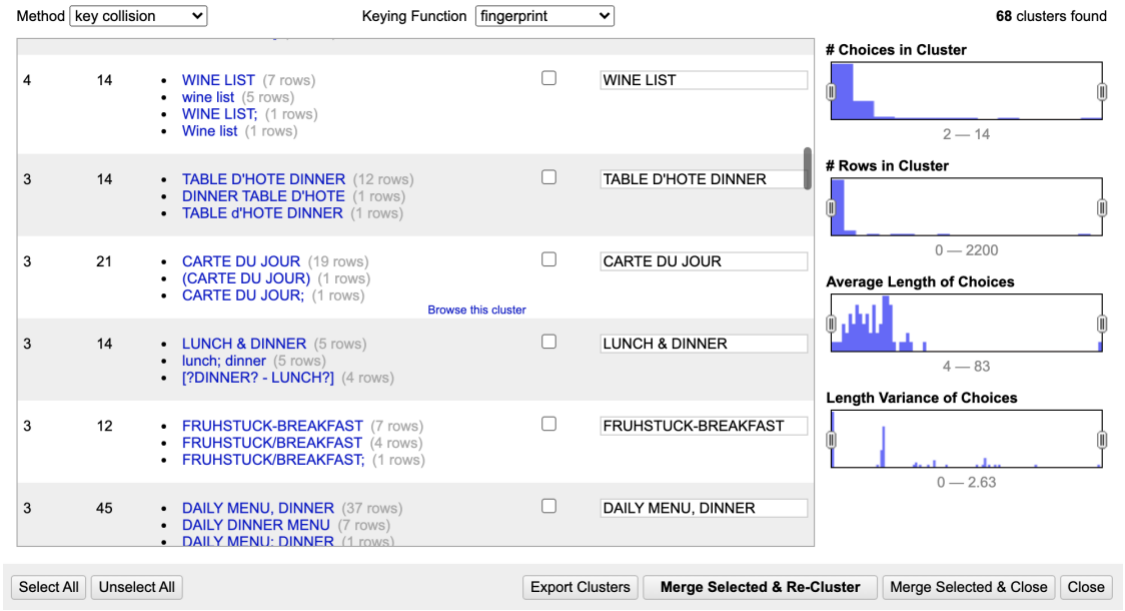
1. ***Cleaning up Dishes name***: Below is the screenshot of few examples name of the dishes we will need to clean.



2. ***Cleaning up Sponsor/Location Details***: Below is the screenshot of few examples Sponsor/Location names we need to clean up

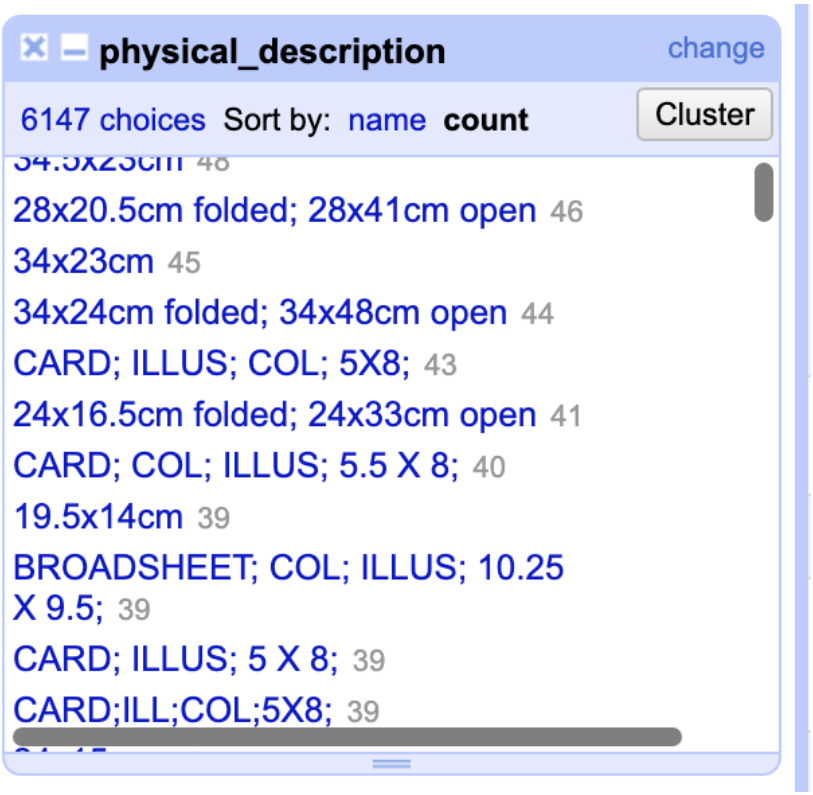


3. ***Clean Event Types***: Below is the screenshots for sample event types names required cleaning

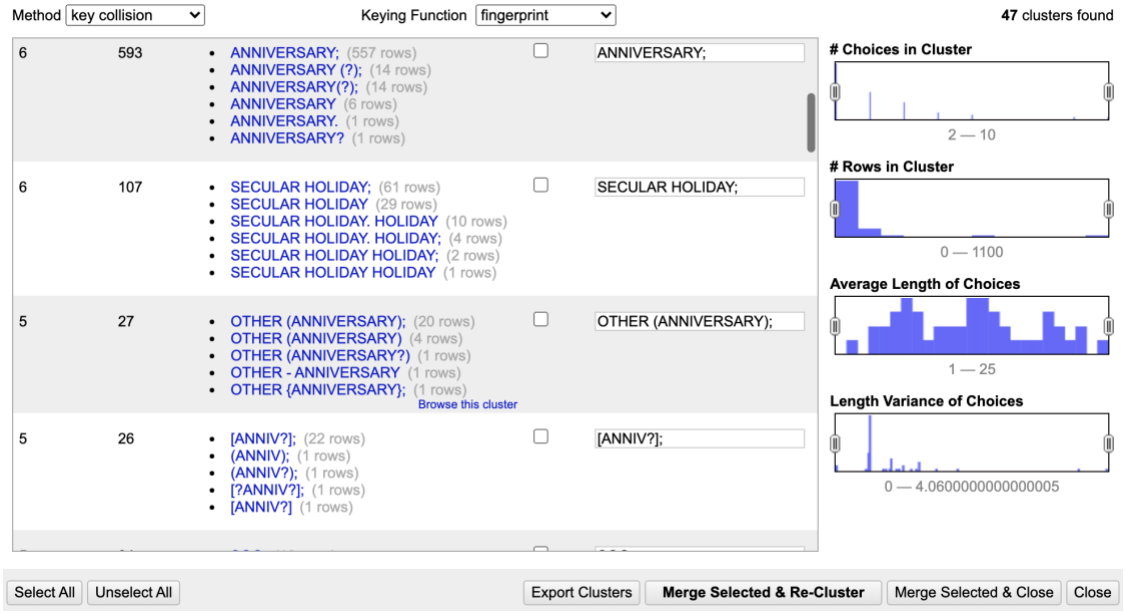


4. ***Date Values***: There seems to some date values outside range will require some cleaning, like following

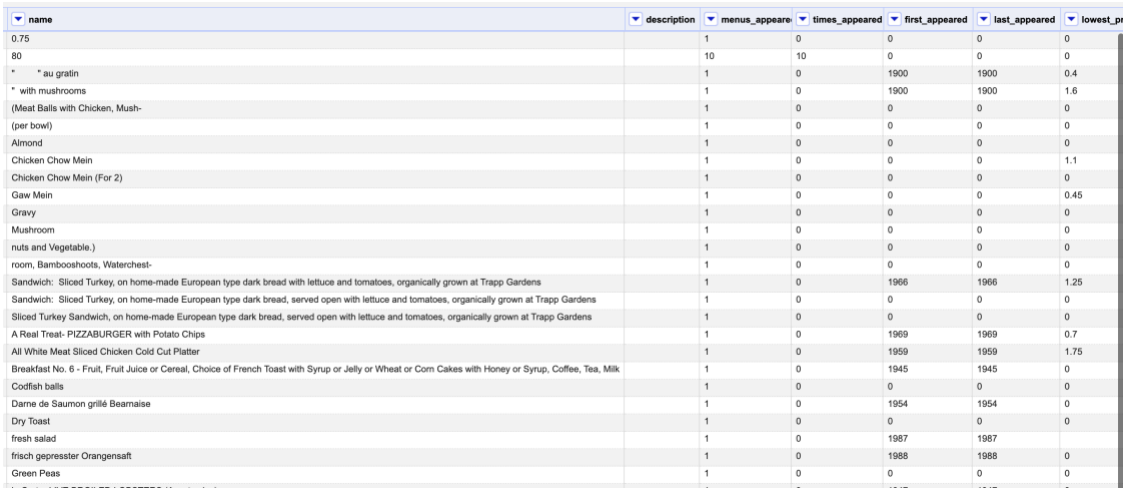
5. ***Additional Cleaning***: Apart from fields needed for usecase U1. There are some more fields can be cleaned and provide more details about menus. Like following
    1. *Physical Details of the Menus*: Seems like following can be cleaned and we can at-least get some information about Menu build like Card, Folder, etc.



2. *Occasion*: Occasion details can also be cleaned to get the details if Menu was used for certain Occasion



3. *Additional Cleaning for Dishes*: There are certain dishes having 0 count for Menu occurrence and not year information. We should be able to drop those from our analysis



name	description	menus_appeare	times_appeared	first_appeared	last_appeared	lowest_pr
0.75		1	0	0	0	0
80		10	10	0	0	0
" " au gratin		1	0	1900	1900	0.4
" with mushrooms		1	0	1900	1900	1.6
(Meat Balls with Chicken, Mush-		1	0	0	0	0
(per bowl)		1	0	0	0	0
Almond		1	0	0	0	0
Chicken Chow Mein		1	0	0	0	1.1
Chicken Chow Mein (For 2)		1	0	0	0	0
Gaw Mein		1	0	0	0	0.45
Gravy		1	0	0	0	0
Mushroom		1	0	0	0	0
nuts and Vegetable.)		1	0	0	0	0
room, Bambooshoots, Waterchest-		1	0	0	0	0
Sandwich: Sliced Turkey, on home-made European type dark bread with lettuce and tomatoes, organically grown at Trapp Gardens		1	0	1966	1966	1.25
Sandwich: Sliced Turkey, on home-made European type dark bread, served open with lettuce and tomatoes, organically grown at Trapp Gardens		1	0	0	0	0
Sliced Turkey Sandwich, on home-made European type dark bread, served open with lettuce and tomatoes, organically grown at Trapp Gardens		1	0	0	0	0
A Real Treat- PIZZABURGER with Potato Chips		1	0	1969	1969	0.7
All White Meat Sliced Chicken Cold Cut Platter		1	0	1959	1959	1.75
Breakfast No. 6 - Fruit, Fruit Juice or Cereal, Choice of French Toast with Syrup or Jelly or Wheat or Corn Cakes with Honey or Syrup, Coffee, Tea, Milk		1	0	1945	1945	0
Codfish balls		1	0	0	0	0
Darne de Saumon grillé Bearnaise		1	0	1954	1954	0
Dry Toast		1	0	0	0	0
fresh salad		1	0	1987	1987	
frisch gepresster Orangensaft		1	0	1988	1988	0
Green Peas		1	0	0	0	0

Initial Plan

Cleaning Steps

1. Convert data to appropriate formats like Date, number or text.
2. Using OpenRefine clean the text fields for leading and trailing spaces and also clean for consecutive spaces
3. Remove any special characters using OpenRefine and Regular Expression
4. Using OpenRefine use clustering methods to clean the fields like Dishes name, Sponsor, Location, Event, Occasion
5. Trying cleaning Physical Description of the Menu using Regular Expression to find Build Type of the Menu
6. Using SQLite build the database schema for the dataset to check for ICs like:
    1. All ids should be unique for Dishes, Menus, Menu Pages
    2. ids should not be null for Dishes, Menus, Menu Pages
    3. Check for the dishes where last appeared or first appeared is 0
    4. Lowest Price should not be greater than Highest Price
    5. First Appeared should not be greater than last appeared
    6. Page count in a Menu should not be null or 0

Phase II Report

Data cleaning performed

To meet the use case U_1 I decided the clean the following columns in the respective files using OpenRefine.

Cleaning Menu.csv file

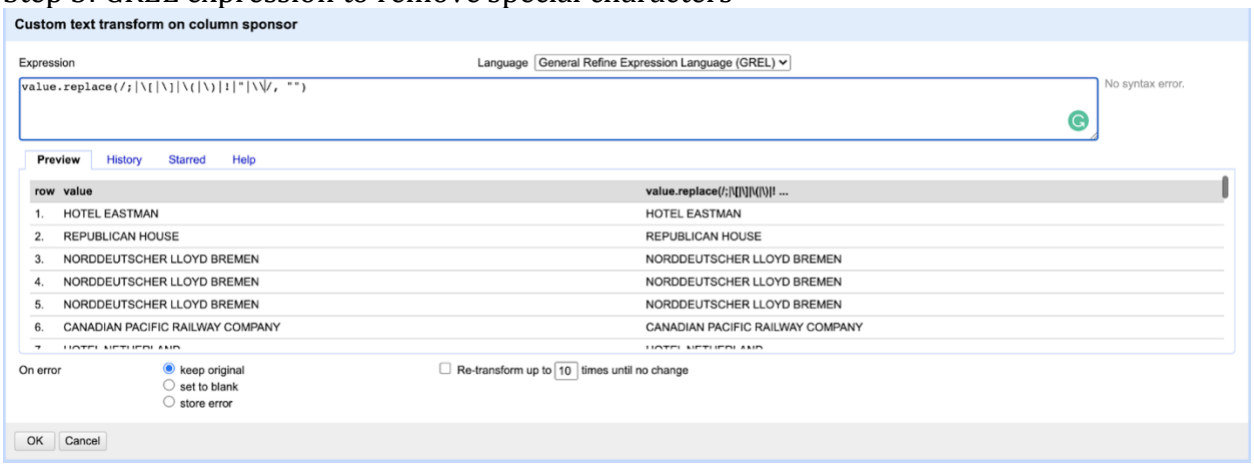
Column: sponsor

As in the U_1we are planning to identify who was the sponsor of a dish at a particular time. Hence cleaning of this column was required to get correct sponsor details.

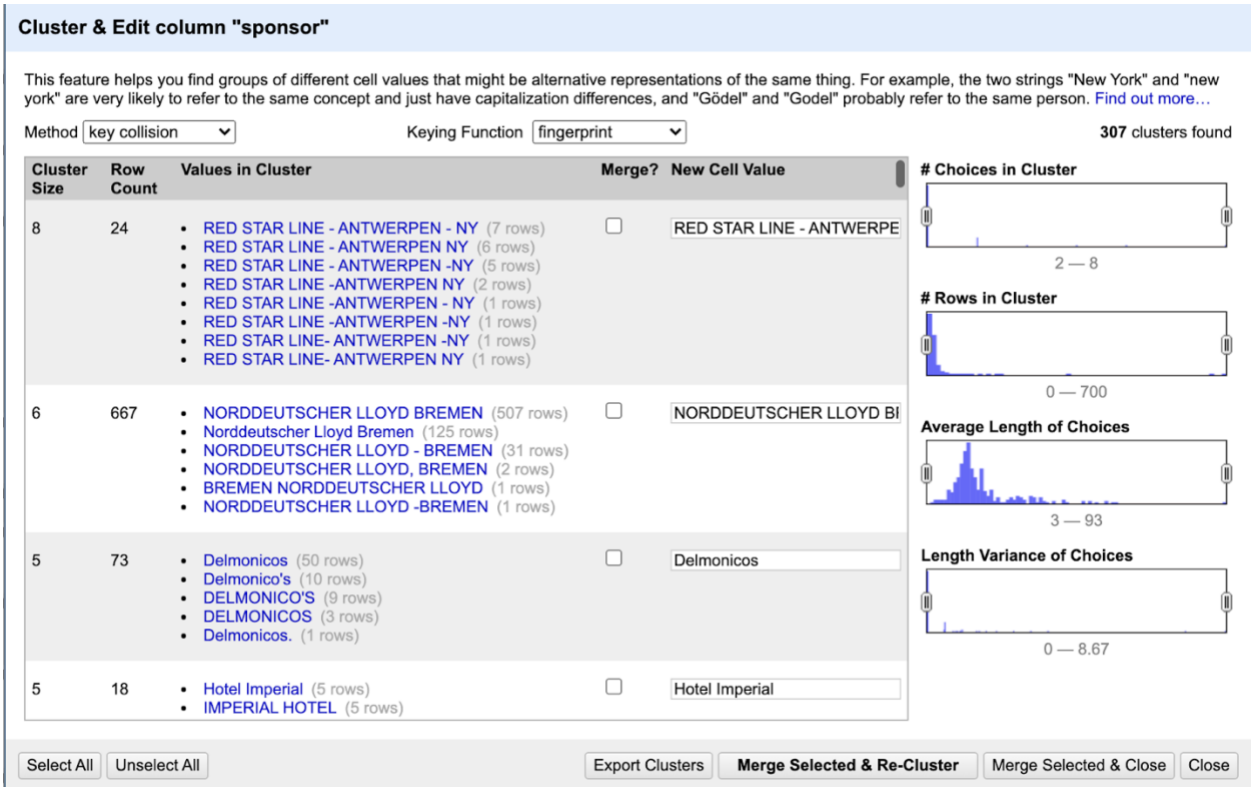
Step 1: Trim Leading and Trailing White Spaces

Step 2: Collapse consecutive white spaces

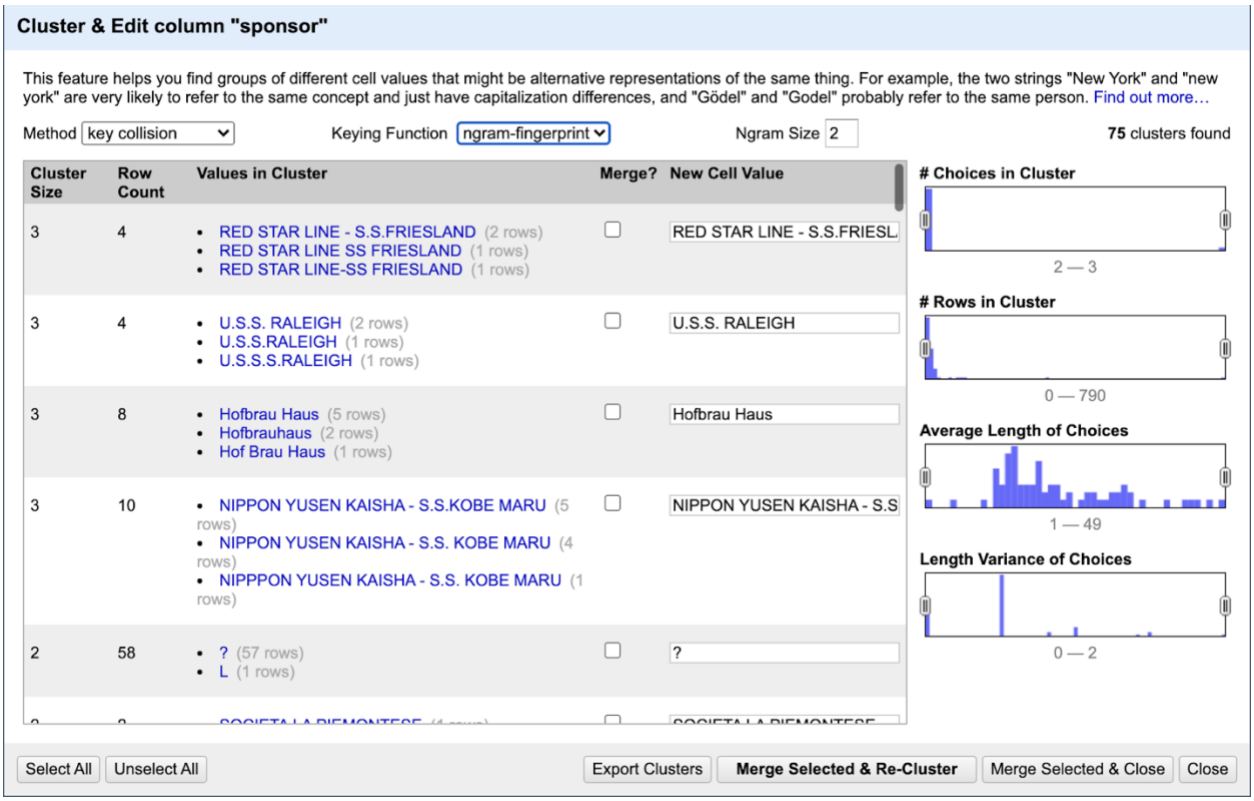
Step 3: GREL expression to remove special characters



Step 4: Cluster and Edit Column - fingerprint



Step 5: Cluster and Edit Column - ngram-fingerprint



Step 6: Redo of Trim Leading and Trailing White Spaces & Collapse consecutive white spaces

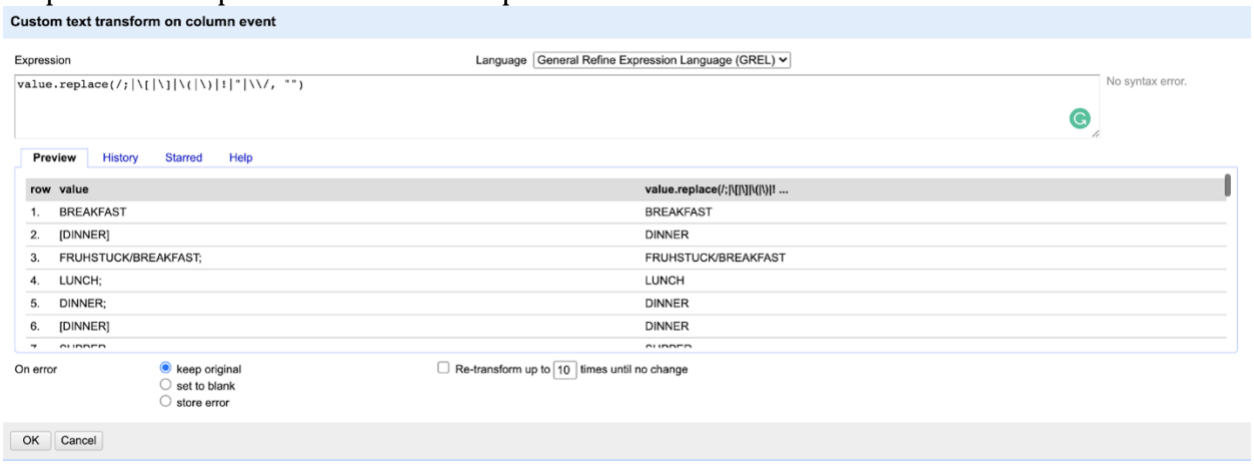
Column: event

As in the U_1 we are planning to identify which was the event of a dish at a particular time. Hence cleaning of this column was required to get correct event details.

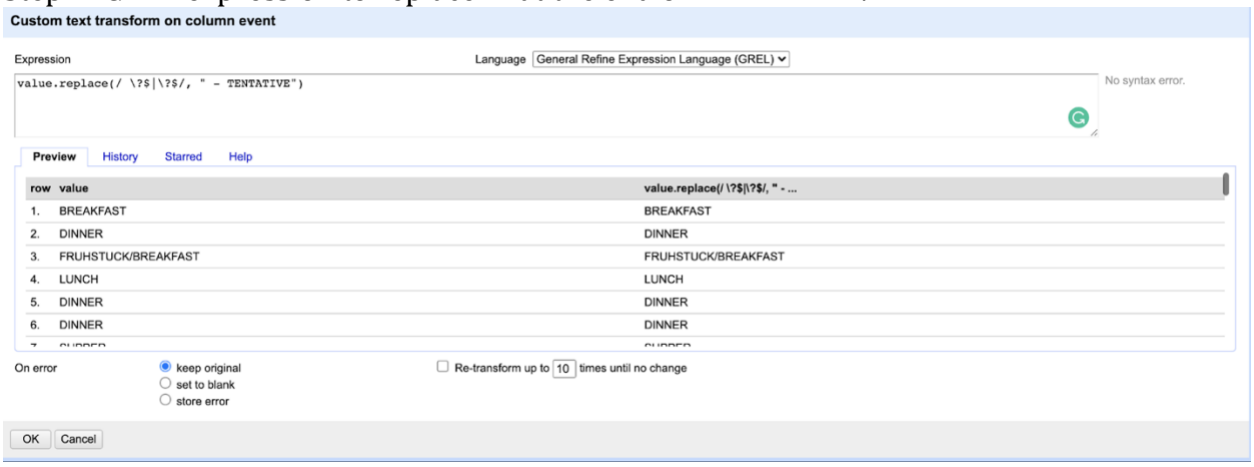
Step 1: Trim Leading and Trailing White Spaces

Step 2: Collapse consecutive white spaces

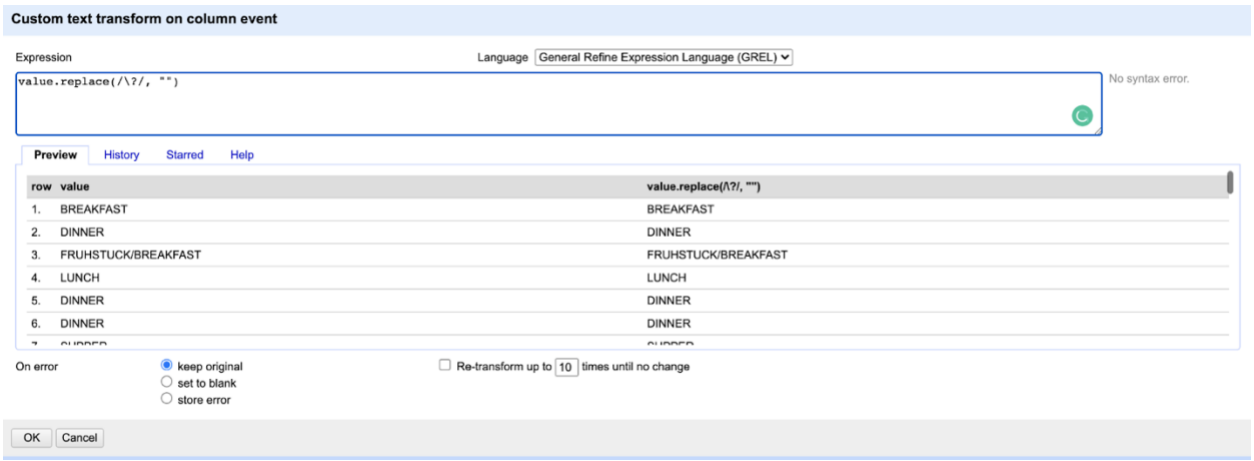
Step 3: GREL expression to remove special characters



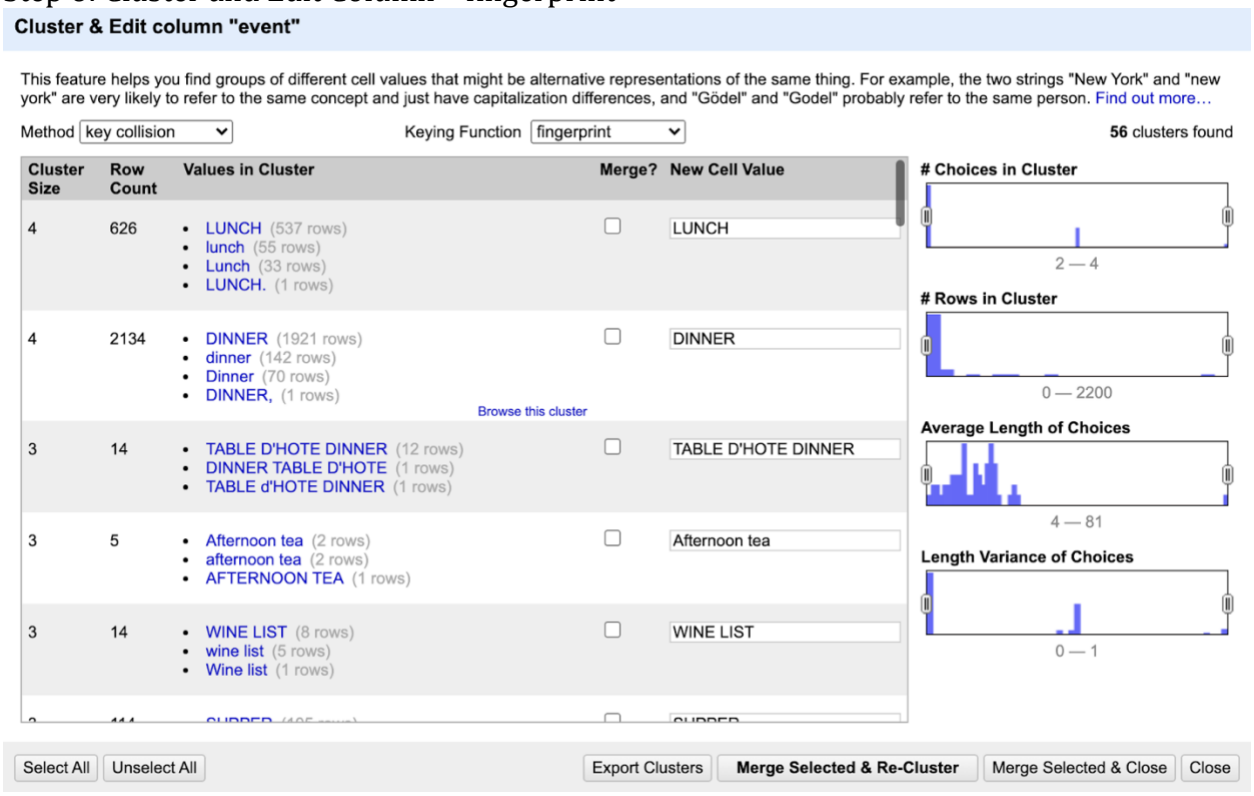
Step 4: GREL expression to replace '?' at the end of ' - TENTATIVE'



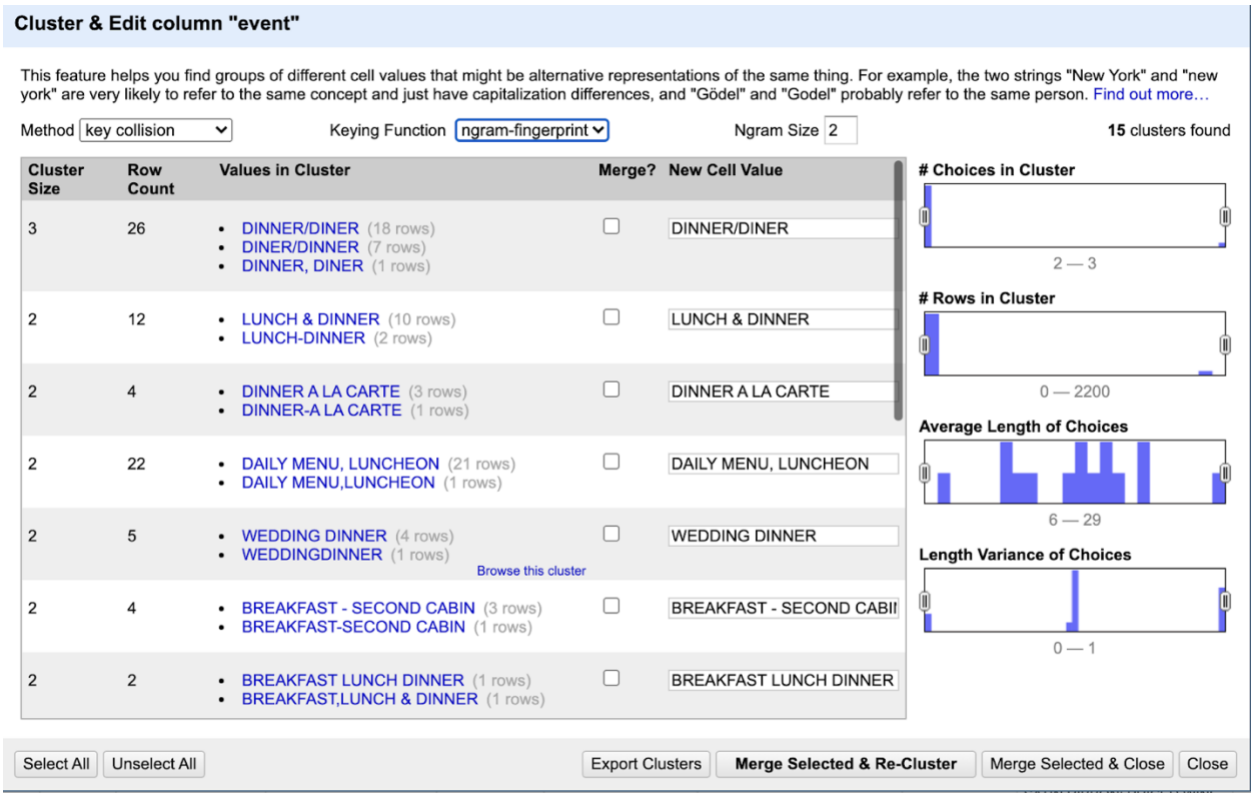
Step 5: Removing additional '?'



Step 6: Cluster and Edit Column – fingerprint



Step 7: Cluster and Edit Column – ngram-fingerprint



Step 8: Redo of Trim Leading and Trailing White Spaces & Collapse consecutive white spaces

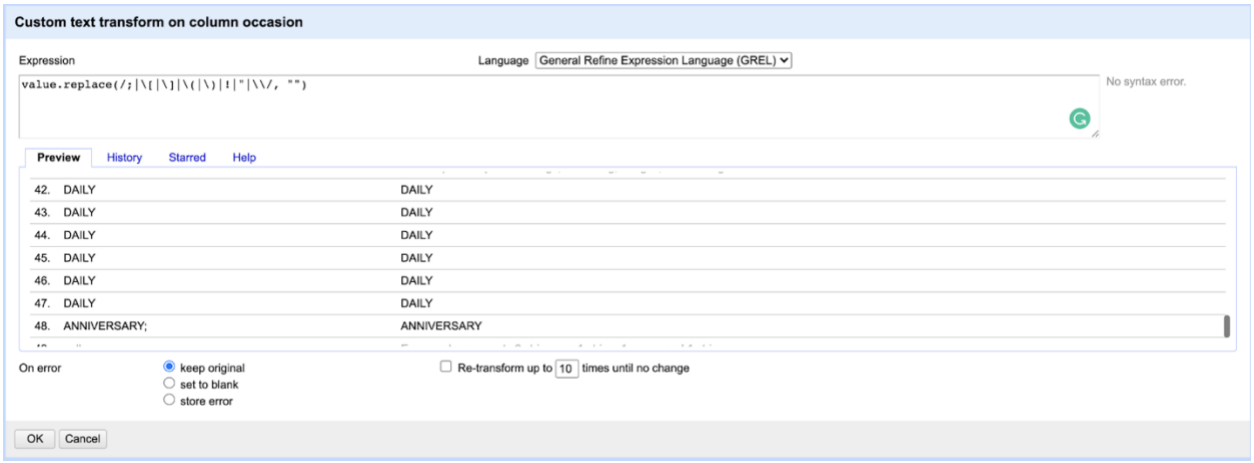
Column: occasion

As in the U_1 we are planning to identify which was the occasion of a dish at a particular time. Hence cleaning of this column was required to get correct occasion details.

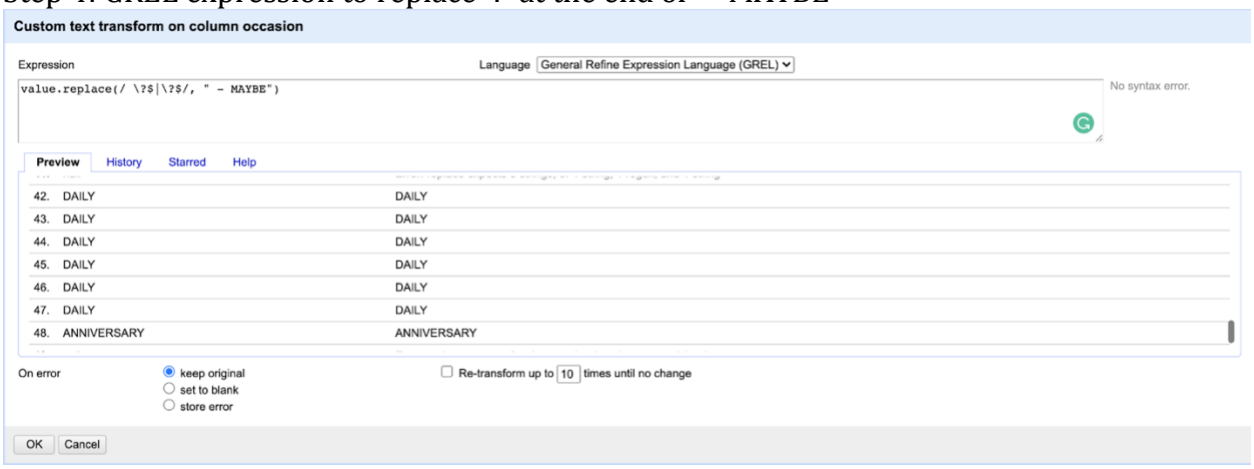
Step 1: Trim Leading and Trailing White Spaces

Step 2: Collapse consecutive white spaces

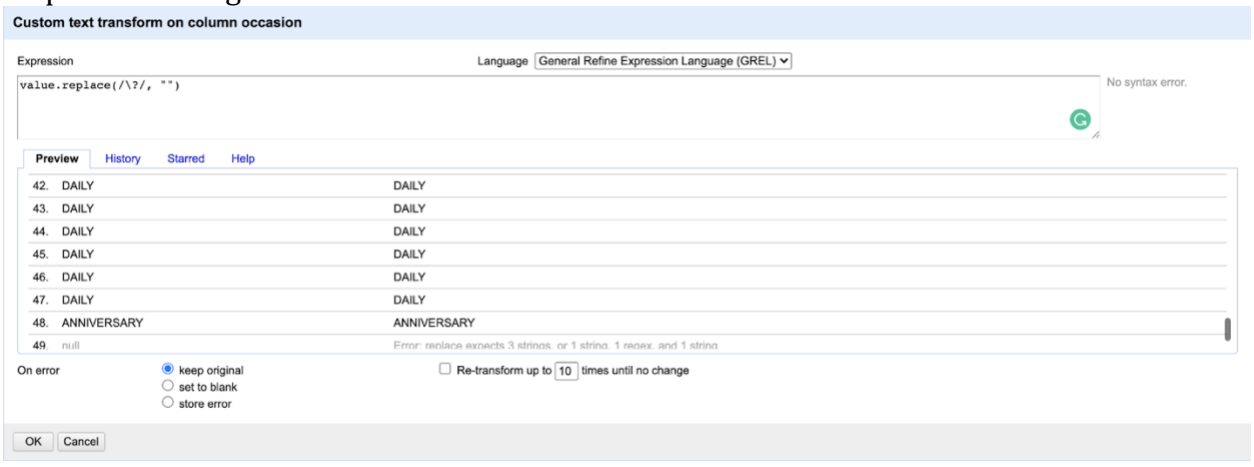
Step 3: GREL expression to remove special characters



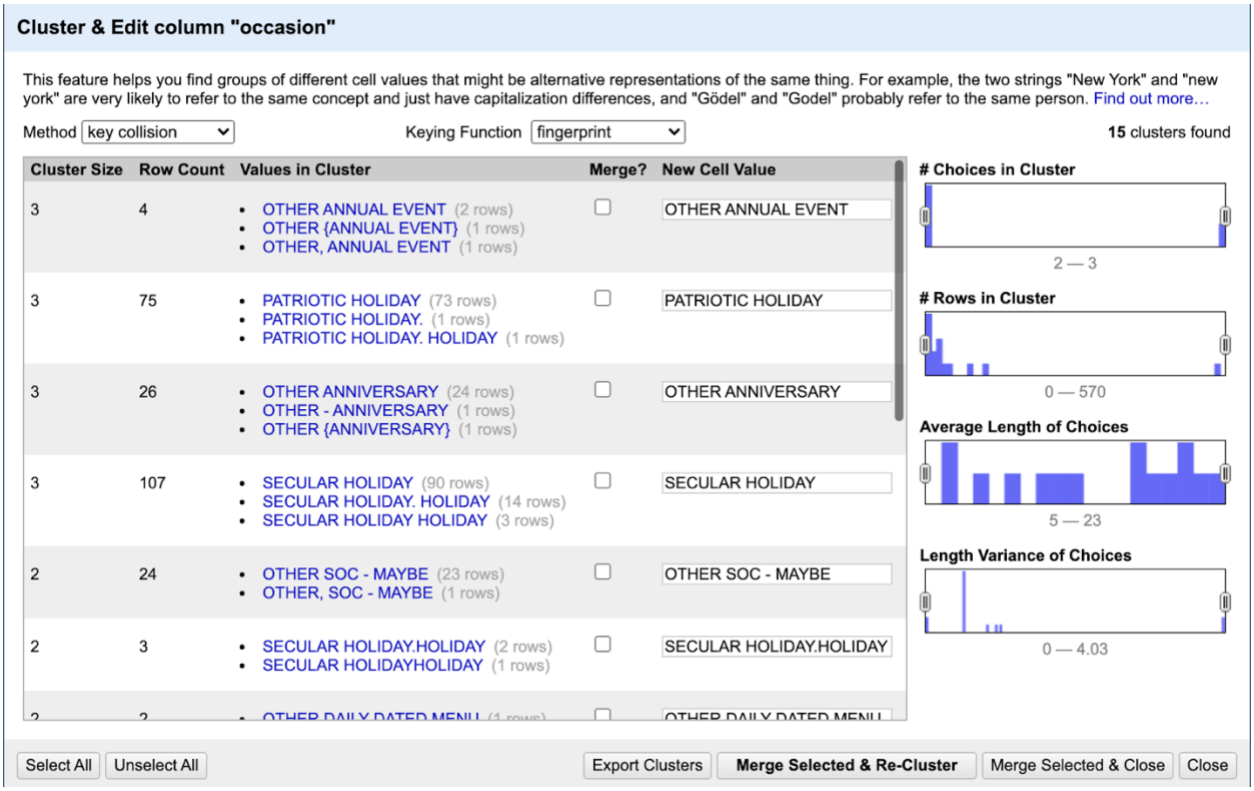
Step 4: GREL expression to replace '?' at the end of ' - MAYBE



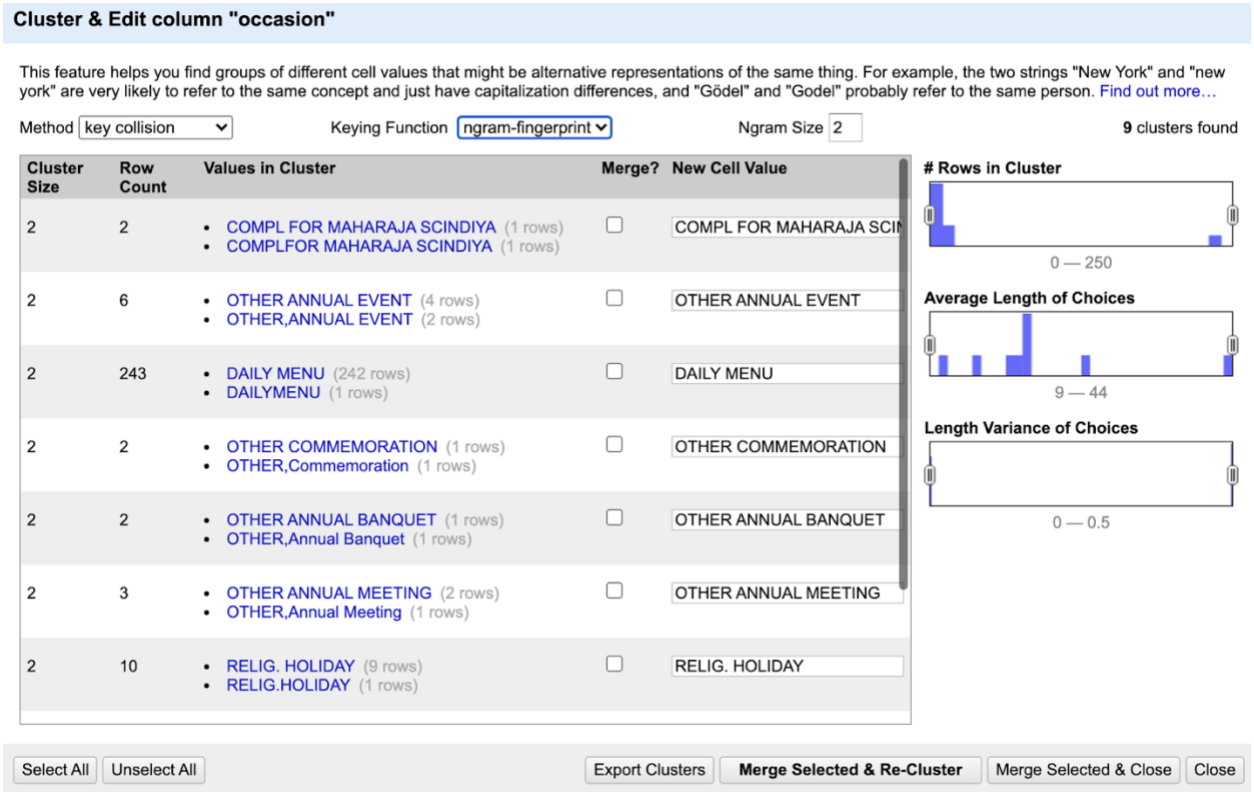
Step 5: Removing additional '?'



Step 6: Cluster and Edit Column – fingerprint



Step 7: Cluster and Edit Column – ngram-fingerprint



Step 8: Redo of Trim Leading and Trailing White Spaces & Collapse consecutive white spaces

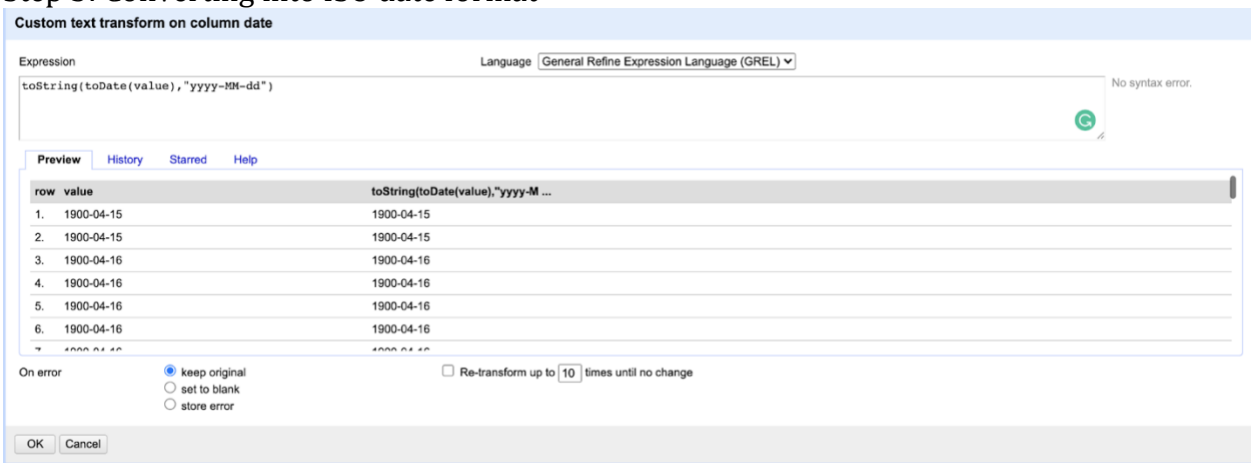
Column: date

As in the $\boldsymbol{U_1}$ we are planning to identify date of first or last occurrence of a dish. Hence cleaning of this column was required to get correct details.

Step 1: Trim Leading and Trailing White Spaces

Step 2: Collapse consecutive white spaces

Step 3: Converting into ISO date format



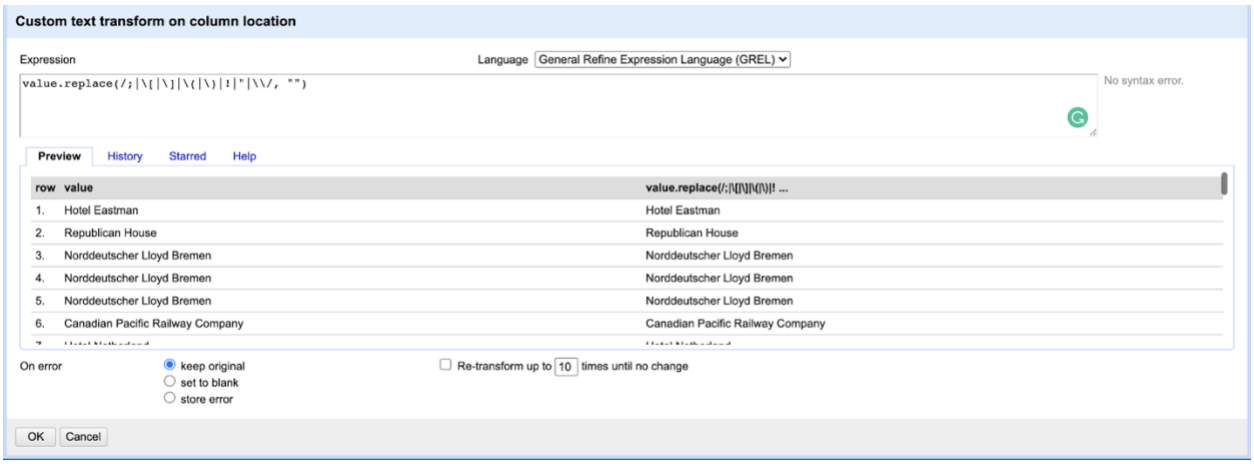
Column: location

As in the $\boldsymbol{U_1}$ we are planning to identify which was the location of a dish at a particular time. Hence cleaning of this column was required to get correct location details.

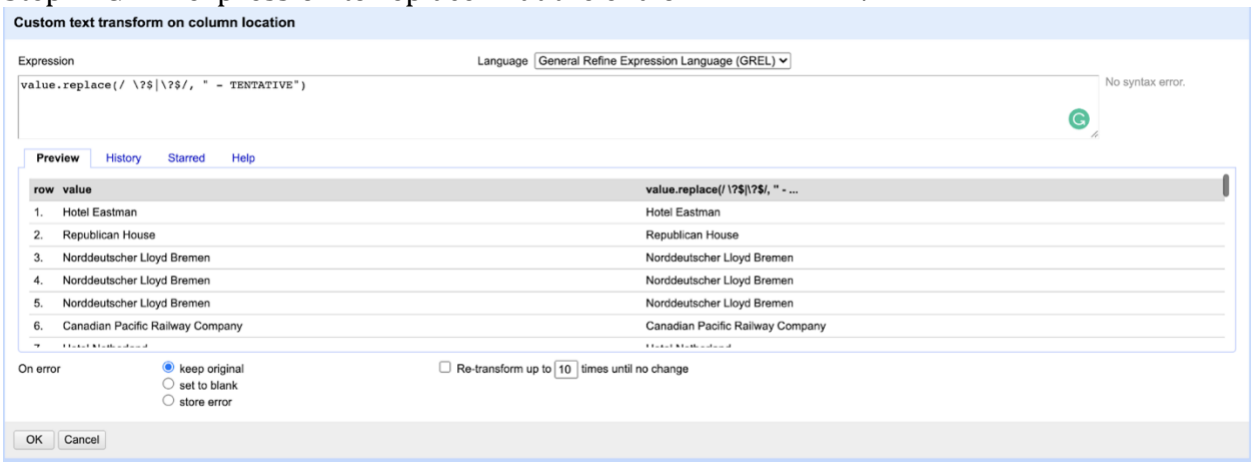
Step 1: Trim Leading and Trailing White Spaces

Step 2: Collapse consecutive white spaces

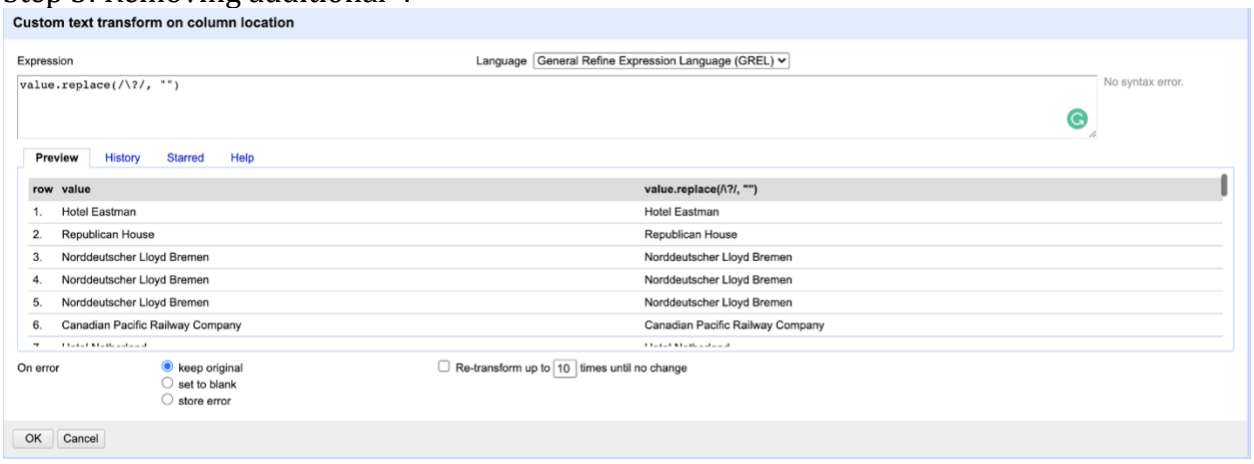
Step 3: GREL expression to remove special characters



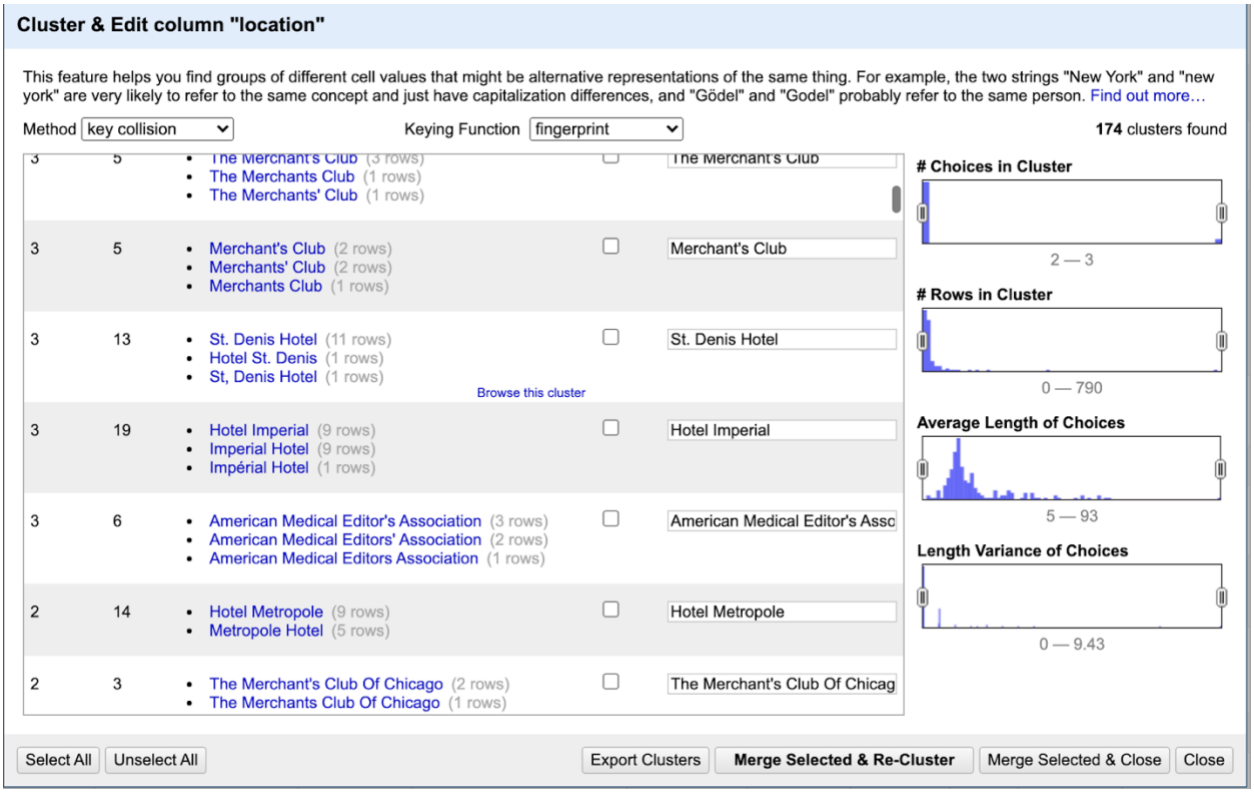
Step 4: GREL expression to replace '?' at the end of ' - TENTATIVE'



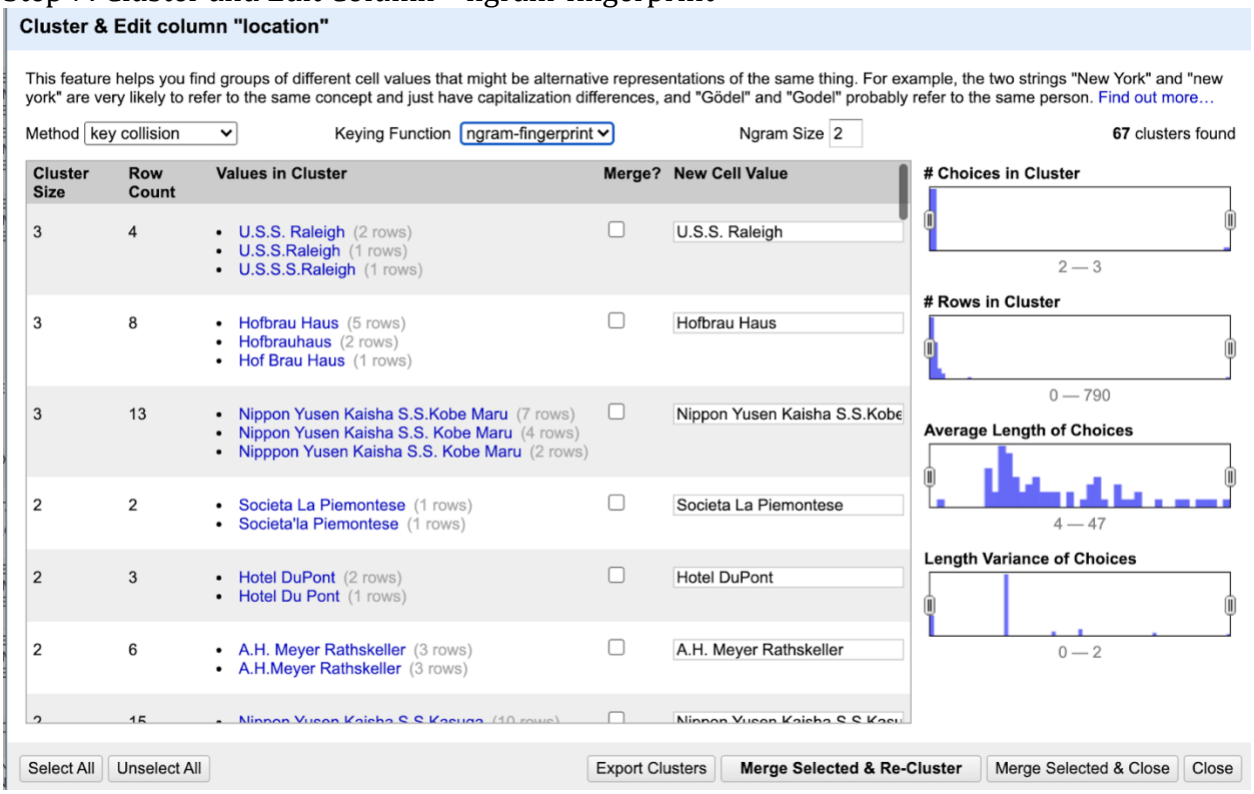
Step 5: Removing additional '?'



Step 6: Cluster and Edit Column – fingerprint



Step 7: Cluster and Edit Column – ngram-fingerprint



Step 8: Redo of Trim Leading and Trailing White Spaces & Collapse consecutive white spaces

Cleaning Dish.csv file

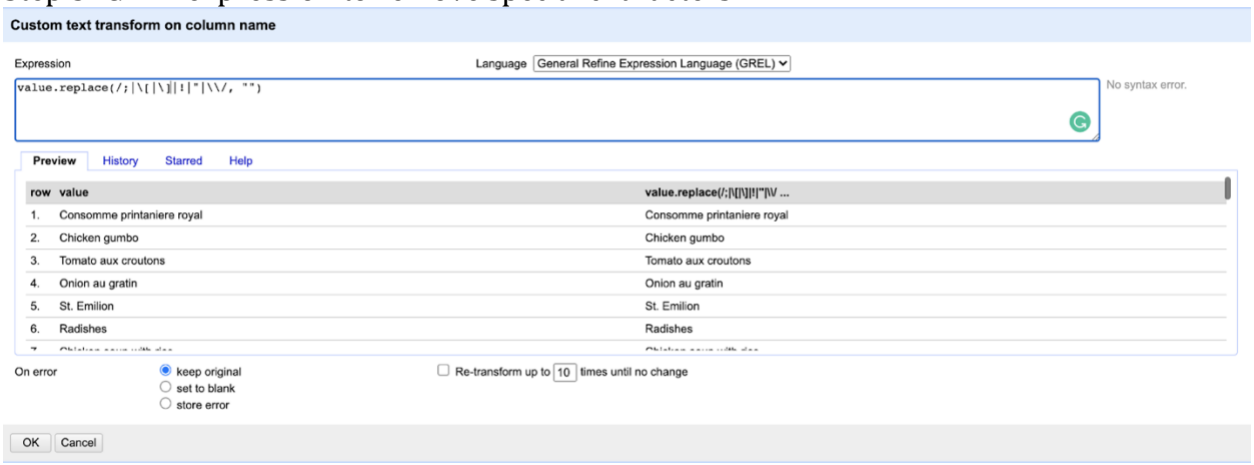
Column: name

As main target of the U_1 is to identify dish details. Hence cleaning of dish name column is the most import step to make sure we get the correct details about the dish based on the name.

Step 1: Trim Leading and Trailing White Spaces

Step 2: Collapse consecutive white spaces

Step 3: GREL expression to remove special characters



Step 4: Cluster and Edit Column – fingerprint
Note: this was one of the longest steps due to very high cluster count of 40K+. Hence this cleaning was performed in multiple steps by picking 4-5K clusters at a time.

Cluster & Edit column "name"

This feature helps you find groups of different cell values that might be alternative representations of the same thing. For example, the two strings "New York" and "new york" are very likely to refer to the same concept and just have capitalization differences, and "Gödel" and "Godel" probably refer to the same person. Find out more…

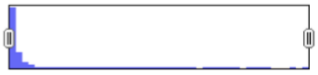
Method: key collision | Keying Function: fingerprint | **40874** clusters found

Cluster Size	Row Count	Values in Cluster	Merge?	New Cell Value
44	47	• EGGS poached on toast (2) (3 rows) • EGGS, poached on toast (2) (2 rows) • (2) Poached Eggs on Toast (1 rows) • 2 (Eggs) Poached on Toast (1 rows) • 2 Eggs Poached on Toast (1 rows) • 2 Eggs, Poached on Toast (1 rows) • 2 Eggs, Poached, on Toast (1 rows) • 2 Eggs, poached on toast (1 rows) • 2 Poached (Eggs) on Toast (1 rows) • 2 Poached (eggs) on Toast (1 rows) • 2 Poached Eggs on Toast (1 rows) • 2 Poached Eggs on toast (1 rows) • 2 Poached on Toast, Eggs (1 rows) • 2 eggs poached on toast (1 rows) • 2 eggs, poached on toast (1 rows) • 2 eggs, poached, on toast (1 rows) • EGGS Poached (2) on Toast (1 rows) • EGGS poached on toast 2 (1 rows) • EGGS, poached on toast 2 (1 rows) • Eggs (2) poached on toast (1 rows) • Eggs (2), Poached on Toast (1 rows) • Eggs - Poached on Toast (2) (1 rows) • Eggs Poached (2), on Toast (1 rows) • Eggs poached on toast (2) (1 rows) • Eggs poached on toast 2 (1 rows) • Eggs poached, on toast (2) (1 rows) • Eggs, 2 Poached on Toast (1 rows)	☐	EGGS poached on toast (2)

 # Choices in Cluster

2 — 44

 # Rows in Cluster



2 — 49

Average Length of Choices

0 — 660

Length Variance of Choices

0 — 34

Select All | Unselect All | Export Clusters | **Merge Selected & Re-Cluster** | Merge Selected & Close | Close

Cluster & Edit column "name"

This feature helps you find groups of different cell values that might be alternative representations of the same thing. For example, the two strings "New York" and "new york" are very likely to refer to the same concept and just have capitalization differences, and "Gödel" and "Godel" probably refer to the same person. Find out more…

Method: key collision | Keying Function: fingerprint | **40874** clusters found

		• Potatoes- French Fried (1 rows) • Potatoes-- French fried (1 rows) • french fried potatoes (1 rows) • potatoes, french fried (1 rows)		
33	36	• Hashed Browned Potatoes (2 rows) • Potatoes, Hashed browned (2 rows) • Potatoes: Hashed browned (2 rows) • Browned Hashed Potatoes (1 rows) • HAshed Browned Potatoes (1 rows) • Hashed Browned potatoes (1 rows) • Hashed Potatoes Browned (1 rows) • Hashed Potatoes, Browned (1 rows) • Hashed browned (potatoes) (1 rows) • Hashed browned POTATOES (1 rows) • Hashed browned Potatoes (1 rows) • Hashed browned potatoes (1 rows) • Hashed potatoes browned (1 rows) • Hashed, Browned Potatoes (1 rows) • Hashed, browned potatoes (1 rows) • POTATOES - hashed browned (1 rows) • POTATOES Hashed Browned (1 rows) • POTATOES Hashed browned (1 rows) • POTATOES- hashed browned (1 rows) • POTATOES: Hashed Browned (1 rows) • Potatoes - Hashed browned (1 rows) • Potatoes - hashed browned (1 rows) • Potatoes Hashed Browned (1 rows) • Potatoes Hashed browned (1 rows)	☐	Hashed Browned Potatoes

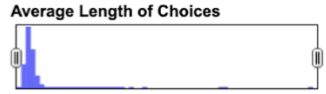
 # Choices in Cluster

2 — 44

 # Rows in Cluster

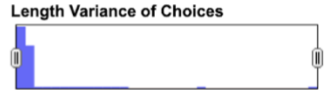
2 — 49

Average Length of Choices



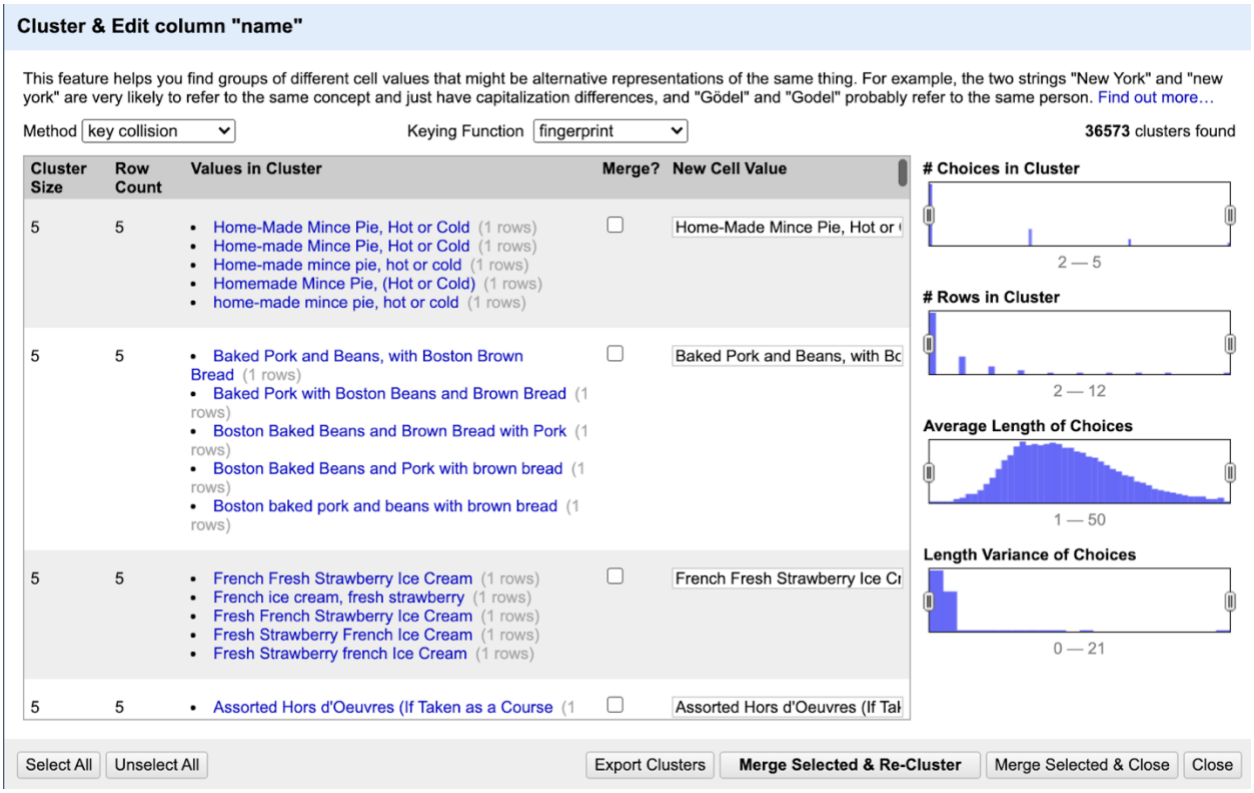
0 — 660

Length Variance of Choices

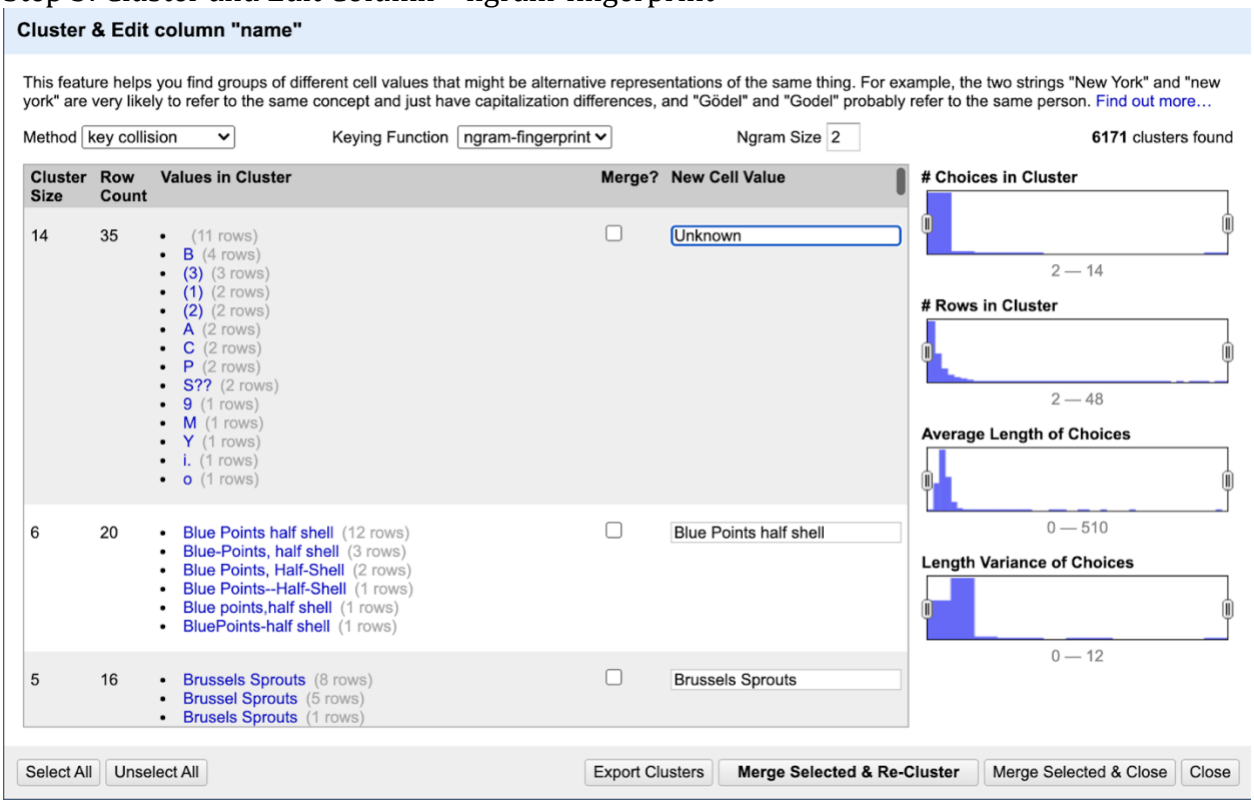


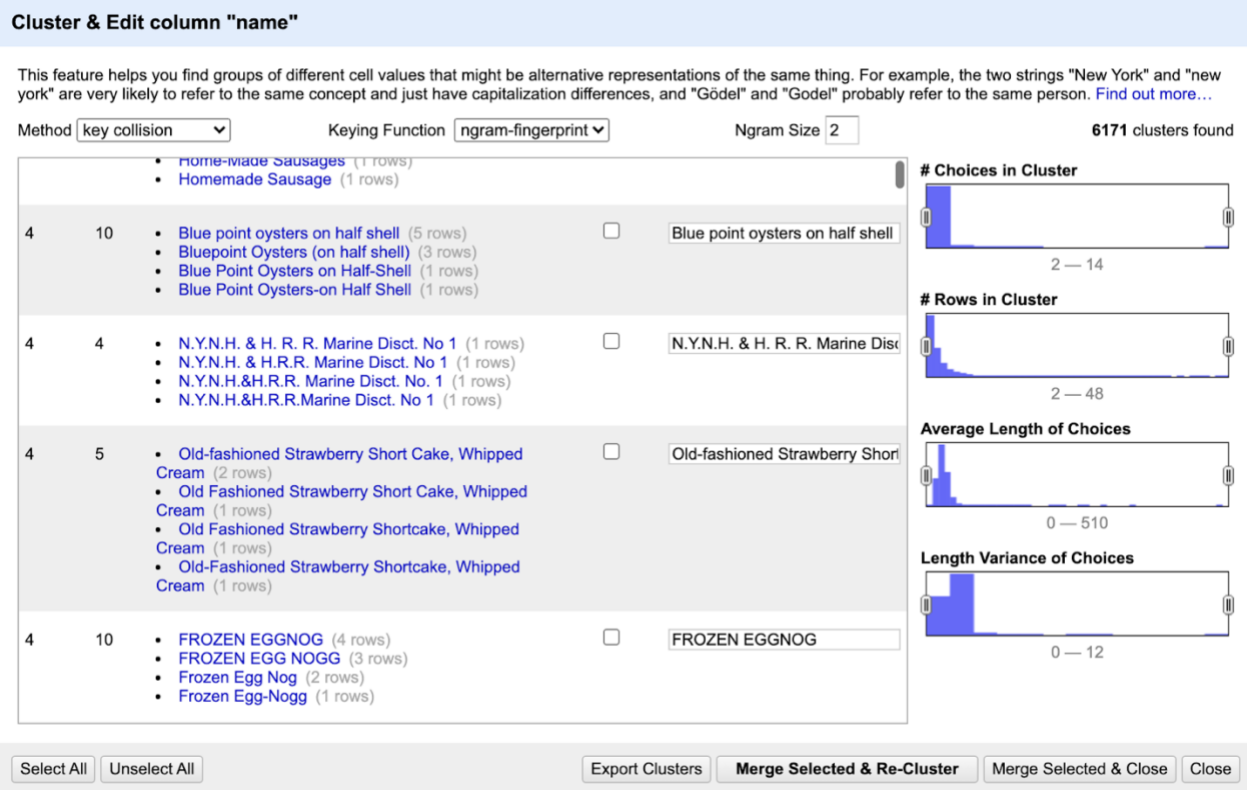
0 — 34

Select All | Unselect All | Export Clusters | **Merge Selected & Re-Cluster** | Merge Selected & Close | Close



Step 5: Cluster and Edit Column – ngram-fingerprint



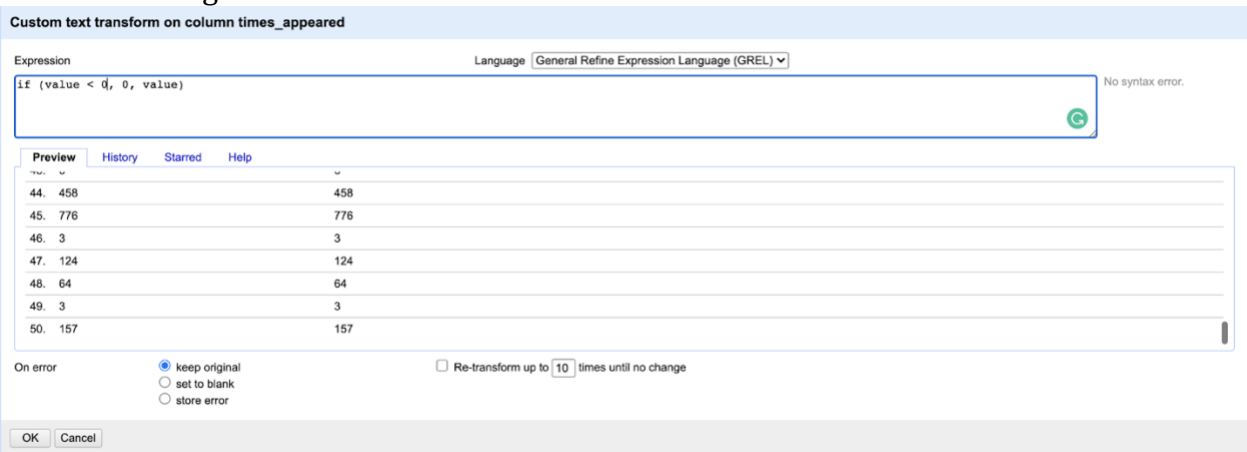


Step 6: Redo of Trim Leading and Trailing White Spaces & Collapse consecutive white spaces

Column: times_appared

This was one of the additional cleaning steps.

values less than 0 converted to 0
27 values changed



Data quality changes

Summary of changes

Following is a summary of the changes done and the impact of each change in terms of number of cells got changed based on each step.

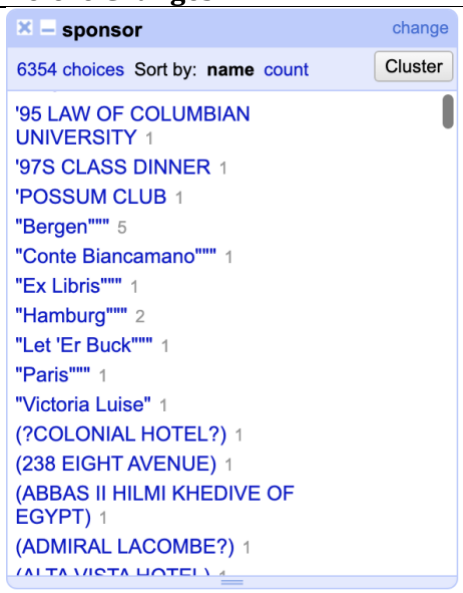
Column Name	Step Performed	Cells Cleaned
sponsor	Trim Leading and Trailing White Spaces	0
sponsor	Collapse consecutive white spaces	127
sponsor	GREL expression to remove special characters	880
sponsor	Cluster and Edit Column - fingerprint	5243
sponsor	ngram-fingerprint	1844
sponsor	Trim Leading and Trailing White Spaces	1
sponsor	Collapse consecutive white spaces	1
event	Trim Leading and Trailing White Spaces	0
event	Collapse consecutive white spaces	6
event	GREL expression to remove special characters	342
event	GREL expression to replace '?'	85
event	Trim Leading and Trailing White Spaces	19
event	Removing additional '?'	24
event	Collapse consecutive white spaces	0
event	Cluster and Edit Column - fingerprint	5460
event	ngram-fingerprint	2311
occasion	Trim Leading and Trailing White Spaces	0
occasion	Collapse consecutive white spaces	3
occasion	GREL expression to remove special characters	2465
occasion	GREL expression to replace '?'	171
occasion	Trim Leading and Trailing White Spaces	50
occasion	Removing additional '?'	2

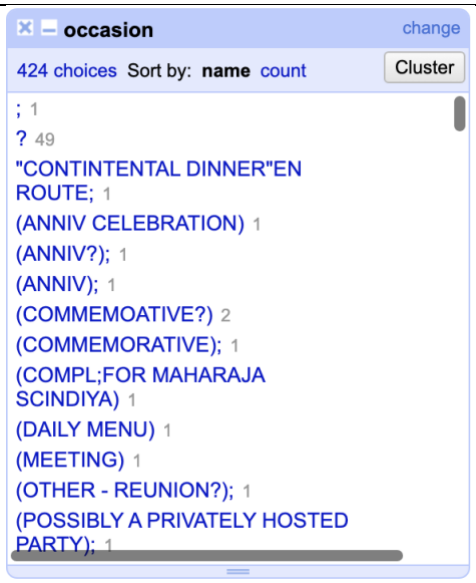
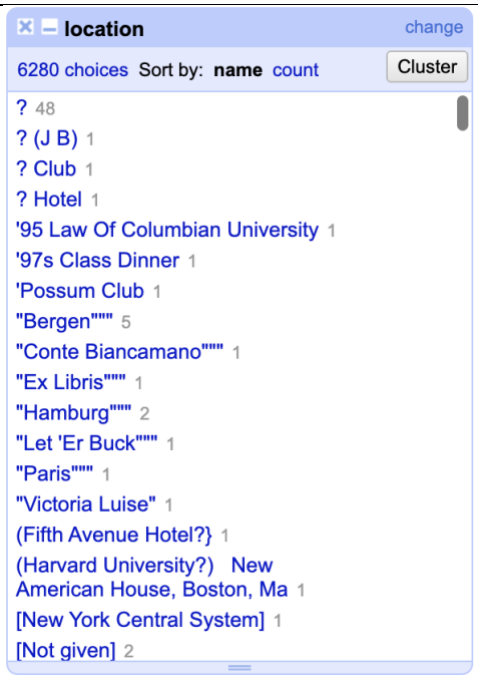
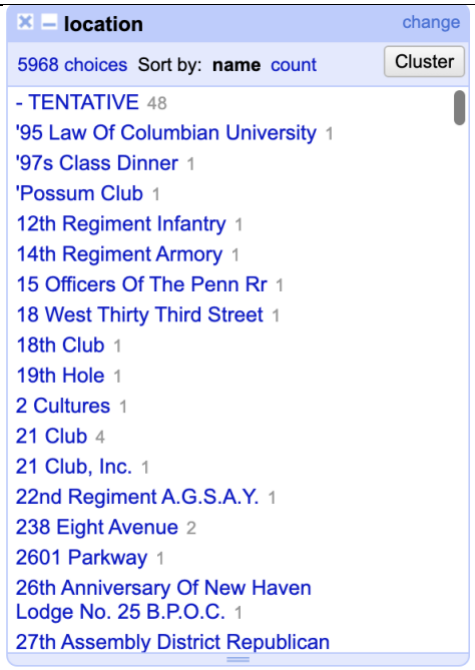
occasion	Collapse consecutive white spaces	0
occasion	Cluster and Edit Column - fingerprint	915
occasion	ngram-fingerprint	272
date	Trim Leading and Trailing White Spaces	0
date	Collapse consecutive white spaces	0
date	GREL expression to remove special characters	4
location	Trim Leading and Trailing White Spaces	0
location	Collapse consecutive white spaces	555
location	GREL expression to remove special characters	755
location	GREL expression to replace '?'	110
location	Removing additional '?'	14
location	Trim Leading and Trailing White Spaces	51
location	Collapse consecutive white spaces	1
location	Cluster and Edit Column - fingerprint	3561
location	ngram-fingerprint	1333
Name (Dish)	Trim Leading and Trailing White Spaces	0
Name (Dish)	Collapse consecutive white spaces	6582
Name (Dish)	GREL expression to remove special characters	10677
Name (Dish)	Trim Leading and Trailing White Spaces	364
Name (Dish)	Collapse consecutive white spaces	71
Name (Dish)	Cluster and Edit Column - fingerprint This was done in 12 steps, as cluster count was very high 40,874	120,392
Name (Dish)	ngram-fingerprint	25341

Before and After

This tables shows how a sample set of data looked before and after changes for different columns cleaned.

Notes: This was not possible to create facet for dish name column due to high cluster count of more than 40K+

Column	Before Changes	After Changes
sponsor		
event		

occasion		
location		

ICV Queries

Various ICV quires were performed using SQLite in Jupyter notebook to validate the dataset and verify the results of cleaning and use case $\boldsymbol{U_1}$

These are described below with screenshots

1. When a dish was appeared on which menu, sponsor, event, place (limiting to 10 counts). Example dish 'Fried Eggs'. **Use case U1**

In [12]:
```
%%sql

select dish.name, menu.sponsor, menu.event, menu.place, dish.first_appeared, menu.date
from menu inner join menupage on menu.id = menupage.menu_id
inner join menuitem on menupage.id = menuitem.menu_page_id
inner join dish on menuitem.dish_id = dish.id
where dish.name = 'Fried Eggs'
limit 10
```

```
 * sqlite:///menus.db
Done.
```

Out[12]:

name	sponsor	event	place	first_appeared	date
Fried Eggs	CUNARD LINE	BREAKFAST	SS ETRURIA	1856	1900-02-21
Fried Eggs				1856	1900-07-08
Fried Eggs				1856	1900-07-08
Fried Eggs				1856	1900-07-01
Fried Eggs	BATTERY PARK HOTEL	BREAKFAST	ASHEVILLE,NC	1856	1900-02-21
Fried Eggs				1856	1900-06-01
Fried Eggs				1856	1900-07-08
Fried Eggs	CANADIAN PACIFIC RAILWAY COMPANY	BREAKFAST	RMS EMPRESS OF CHINA	1856	1900-04-23
Fried Eggs				1856	1901-11-07
Fried Eggs				1856	1901-10-05

2. When a dish was **first** appeared on which menu, sponsor, event, place (limiting to 10 counts). Example dish 'French Fries. **Use case U1**

-

In [13]:
```
%%sql

select dish.name, menu.sponsor, menu.event, menu.place, menu.date, dish.first_appeared
from menu inner join menupage on menu.id = menupage.menu_id
inner join menuitem on menupage.id = menuitem.menu_page_id
inner join dish on menuitem.dish_id = dish.id
where dish.name = 'French Fries'
and menu.date is not ''
order by menu.date
limit 1
```

```
 * sqlite:///menus.db
Done.
```

Out[13]:

name	sponsor	event	place	date	first_appeared
French Fries	Hotel Worthy	DINNER	SPRINGFIELD, MASS.	1901-07-05	1901

3. When a dish was **last** appeared on which menu, sponsor, event, place (limiting to 10 counts). Example dish 'Fried Eggs'. **Use case U1**

-

In [14]:
```
%%sql

select dish.name, menu.sponsor, menu.event, menu.place, menu.date, dish.last_appeared
from menu inner join menupage on menu.id = menupage.menu_id
inner join menuitem on menupage.id = menuitem.menu_page_id
inner join dish on menuitem.dish_id = dish.id
where dish.name = 'Fried Eggs'
and menu.date is not ''
order by menu.date desc
limit 1
```

```
 * sqlite:///menus.db
Done.
```

Out[14]:

name	sponsor	event	place	date	last_appeared
Fried Eggs	Holland America Cruises			1987-01-01	1987

Additional verifications

4. Verify uniqueness of id field in files. No data output with these queries means, all of these IDs were unique in each file.

Menu Table

In [6]:
```
%%sql

select *
from menu
where id in (select id as c
from menu
group by id
having count(id) > 1 )
```

```
 * sqlite:///menus.db
Done.
```

Out[6]:

id	name	sponsor	event	venue	place	physical_description	occasion	notes	call_number	keywords	language	date	location	location_type	currency	curre

Dish Table

In [7]:
```
%%sql

select *
from dish
where id in (select id as c
from dish
group by id
having count(id) > 1 )
```

```
 * sqlite:///menus.db
Done.
```

Out[7]:

id	name	description	menus_appeared	times_appeared	first_appeared	last_appeared	lowest_price	highest_price

Menupage Table

In [8]:
```
%%sql

select *
from menupage
where id in (select id as c
from menupage
group by id
having count(id) > 1 )
```

```
 * sqlite:///menus.db
Done.
```

Menu Item Table

In [17]:
```
%%sql

select *
from menuitem
where id in (select id as c
from menuitem
group by id
having count(id) > 1 )
```

```
 * sqlite:///menus.db
Done.
```

5. Each Menu item should have a single menu page id - Verified

In [20]:
```
%%sql


select id, menu_page_id
from menuitem
where menu_page_id in (select id as c
from menuitem
group by id
having count(menu_page_id) > 1 )
```

```
 * sqlite:///menus.db
Done.
```

Out[20]:

id	menu_page_id

6. Dish Count less than menu page count – 150 instances found against this constraint, but this could be due to other details and pictures in the menu. Hence no data was changed for this. But it was an interesting discovery

Ideally, dish count should be more than menu page count. But there are some instancs where this is not correct. This could be due to other details and pictures in the menu

In [38]:
```
%%sql

select count(*)
from menu
where cast(page_count as integer) > cast(dish_count as integer)
```

```
 * sqlite:///menus.db
Done.
```

Out[38]:

count(*)
150

7. Top 5 dishes with most appearances

In [9]:
```
%%sql

select name, count(name)
from dish
group by name
order by count(name) desc
limit 5
```

```
 * sqlite:///menus.db
Done.
```

Out[9]:

name	count(name)
Fried sweet potatoes	49
EGGS poached on toast (2)	48
Cold Roast Beef	47
Grape Fruit, Half	45
G H Mumm's Extra Dry	45

8. Top 5 dishes appearing on the most number of menus

In [10]:
```
%%sql

select id, name, menus_appeared
from dish
order by menus_appeared desc
limit 5
```

```
 * sqlite:///menus.db
Done.
```

Out[10]:

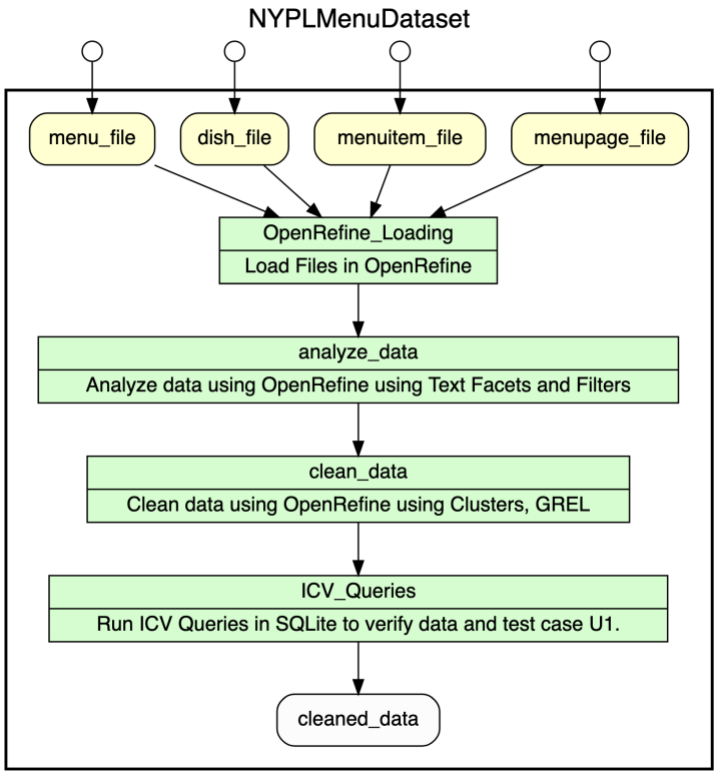
id	name	menus_appeared
150	Little Neck clams	995
1034	Anchovies on toast	99
1047	*Roast Capon	99
1372	Creme Yvette	99
2798	Pork Tenderloin	99

Workflow Model

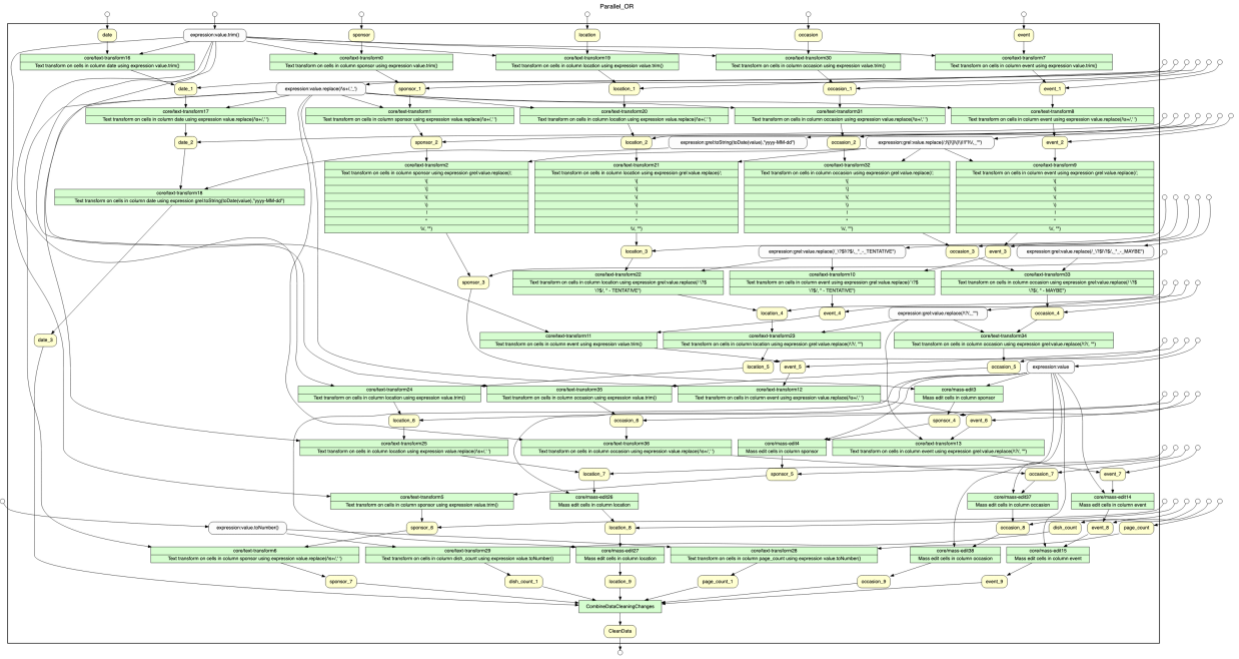
Workflow W0:

Overall Workflow of the process used in this exercise.

- OpenRefine was used as a tool to clean the data like remove special characters using GERL, Clustering, format conversion and other data cleaning steps
- SQLite was used to run ICV queries using Jupyter notebook

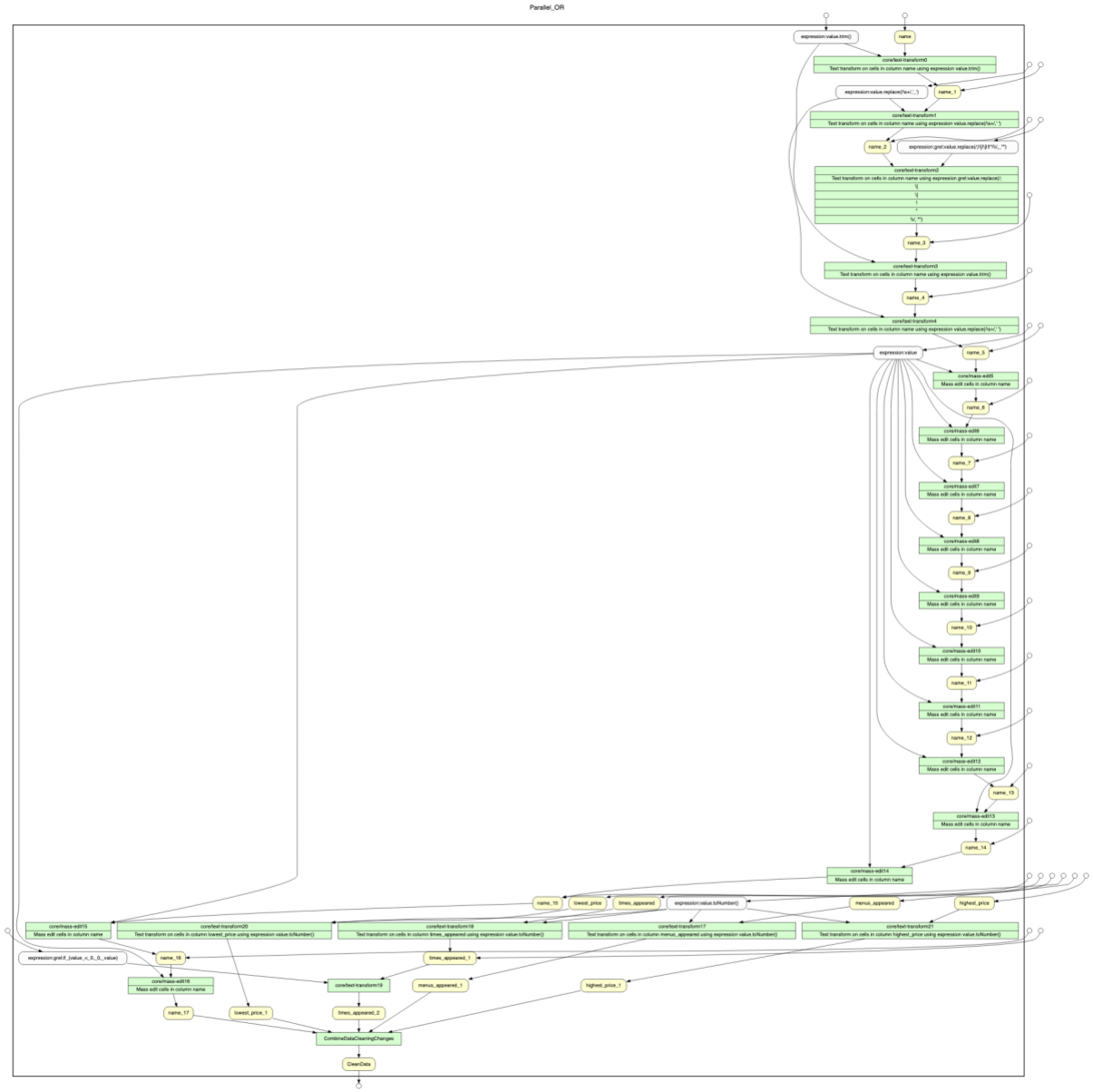


Workflow W1: Cleaning Menu.csv file



Note: Due to complex workflow model all the elements may not be visible in this report. So, a png file is separately added as well along with the submission

Workflow W2: Cleaning Dish.csv file



Note: Due to complex workflow model all the elements may not be visible in this report. So, a png file is separately added as well along with the submission

Conclusions & Summary

This data cleaning activity was quite fun and great learning activity. Working with OpenRefine, SQLite, using SQLite on Jupyter notebook, using YesWorkflow was a great experience. One of the big challenges of the activity was cleaning a huge dataset, specially Dish.csv file. Number of clusters were more than 40K and OpenRefine just kept running for hours. Then I decided to filter the clusters and perform the Steps by dividing it into 12 steps. As a team of one individual, I did perform all the activities in this project.

Although two files MenuItem and MenuPage are very less mentioned in this report. The reason is almost not data cleaning was needed for these two. But these two were key file to get the linking between a Dish and its respective Menu.

Overall, as a summary I was able to clean more than 190K cells including multiple steps and able to achieve target to find dish details like on which Menu dish first or last appeared.