Final Project

End-To-End Data Cleaning Workflow – NYPL Data

by

Sudhir Behani (sbehani2@illinois.edu)

Bollam Raja Shekar (bollamr2@illinois.edu)

Table of Contents

1. Introduction

This Report presents one of the many alternatives in Data Cleaning and Provenance establishment of "What's on the menu?" dataset provided by New York Public Library and extracted from http://menus.nypl.org. We begin with describing the dataset and the distribution of information among the four datasets in .csv format. Then we go on to assess the data set by identifying the corrupt, incomplete, irrelevant and inaccurate parts in the dataset.

Upon successful assessment of the dataset we suggest and perform various data cleaning techniques using Open Refine tool. In the process we explain what techniques are employed, how they are processed and a yesworkflow is presented. The detailed openrefine recipes and the yesworkflow code are provided as supplementary documents.

On completion of the Data Cleaning activity, the cleaned datasets (csv) are converted into Relational Database Schema with the help of sqlite3. A RDS and a table is designed from the datasets with Primary and Foreign Keys. The relationship between the Tables is well explained with the help of Entity-Relationship (ER) Diagram generated from Workbench Tool. Furthermore, Integrity Constraints Checks are designed and executed on database for extraction of more reliable and accurate data.

Finally the Data Cleaning results are presented as ER diagrams and Yesworkflow diagrams explaining graphically about the Data Cleaning and Provenance establish of NYPL dataset.

2. Initial Assessment of the dataset

1. Dataset

The data in these files comes from several different sources. Some of the data is supplied by "volunteers," by which we mean people who have participated in the project through the What's On the Menu? site. Some of the information is generated by "web application." This means that some of this information was automatically created as the database supporting the application was constructed and populated (e.g., various ids); some is created as the web application runs (e.g. timestamps as data values are updated). Finally, a lot of information is generated from "NYPL metadata." This metadata comes from many places and reflects the long history of the project and the many parts of New York Public Library involved in it. Much of the data "supplied by NYPL metadata" in the menu spreadsheet is from the catalog cards made by Frank E. Buttolph in the early twentieth century.

1.1 Dish.csv

Information about all the dishes from all the menus transcribed by the project are stored in dish.csv. In this data file, a dish is represented by a row of values. Columns identify attributes of a dish. One of these attributes is an identifier, which identifies the dish. However, the identity of a dish appears to be based on the exact form of the string labeled "name." Thus, dishes with variant orthographic forms of their names, e.g. "half chicken", "Half Chicken" and "chicken [half]") are treated as separate entries with different identifiers.

1.2 Menu.csv

Information about the menus as physical objects, including historical information about their origins, uses, and formats are stored in menu.csv

1.3 MenuItem.csv

This is the largest data file. A "MenuItem" represents a single instance of a dish appearing somewhere on a menu page image. "MenuItem.csv" is useful as a mapping between multiple other data files.

1.4 MenuPage.csv

Information about individual pages of the menus is stored in "MenuPage.csv". Pages are modeled here as digital images produced as a result of digitization by the NYPL. Menus often have multiple pages.

1.5 Missing Data

We note the presence of missing data points in the column “missing values.” However, there are also strings in the present data that point to missing information (e.g. “unknown” or “?”), which do not formally appear as null values.

2. Quality Assessment

What’s on the Menu? Dataset is fairly well managed dataset. It is in a machine-readable, structured and non-proprietary format. We can extract useful information like the name of the restaurant, its geographical location, date etc. But the actual menu contents like all the dishes offered in the restaurant or the list of beverages in the menu cannot be easily sort out.

Upon careful observation, we get a fair idea on popular dishes based on its presence on how many times the dish has appeared in a menu, when the first and last time the dish was appeared in a menu. In order for us to know if a dish is appeared on a menu, we can use the MenuPage and MenuItem dataset to make a conclusion

But the accuracy and completeness of the data in this dataset is far from perfect. There are multiple occasions where the data in particular entries are inappropriate. For example in dish dataset, there are instances where “number of times a dish appeared” has negative values. Similarly “first appeared” and “last appeared” have zero values. In some cases the formatting in “physical address” is quite bad.

On the other hand can this dataset alone be used by researchers, historians, chefs, and nutritionist? Can derive a conclusion on where a specific food item is served in a particular year? Or what was the price of breakfast in 1971? How the popularity of a dish has change over a period of time? Can we predict food preferences over the period of time or during any special events? Can we predict anything about a dish’s price based on its name or description?

The answer is NO. For that, we need to do great deal of data cleaning and provenance. In the rest of this report we try to head towards addressing at least some of the questions using Open Refine and Relational Database Schema.

3. Data Cleaning methods and Process

Open Refine

I. Menu

Following operations are performed on columns in Menu dataset:

- Trim leading and trailing white spaces
- Collapse consecutive white spaces
- Convert columns to number type
- Convert columns date to format YYYY-MM-DD
- Remove special characters “()[]{}\?#!%” from the text columns. Also, remove characters at the end of the string like semicolon, comma and hyphen “;,-”

  ***Note**: Can’t remove (semicolon) character in the middle of string as it signifies multiple options- example multiple sponsors, event, venue, place.*

- Remove continuous hyphen characters from the text columns
- After removing special characters again perform white spaces removal for each the string data type columns
- Clusters columns based on method key collision and keying function as fingerprint. Accepted the default values for new cell value. Merge cluster and then perform ngram-fingerprint using ngram size as 2. Metaphore3 and cologne-phonetic were mostly creating clusters with somewhat different names too so were not used.
- Physical_description seems to contain information about the menu hard copy format and its size. Menu could be in different forms – accordian fold, book, booklet, broadside, card, folder, tri-folder, two cards joined by ribbons etc

  Split the physical_description column with separator as semicolon. It is split into 7 columns. Rename the first column as “physical_description_menu_type” and join other columns as rename it as “physical_description_menu_others”

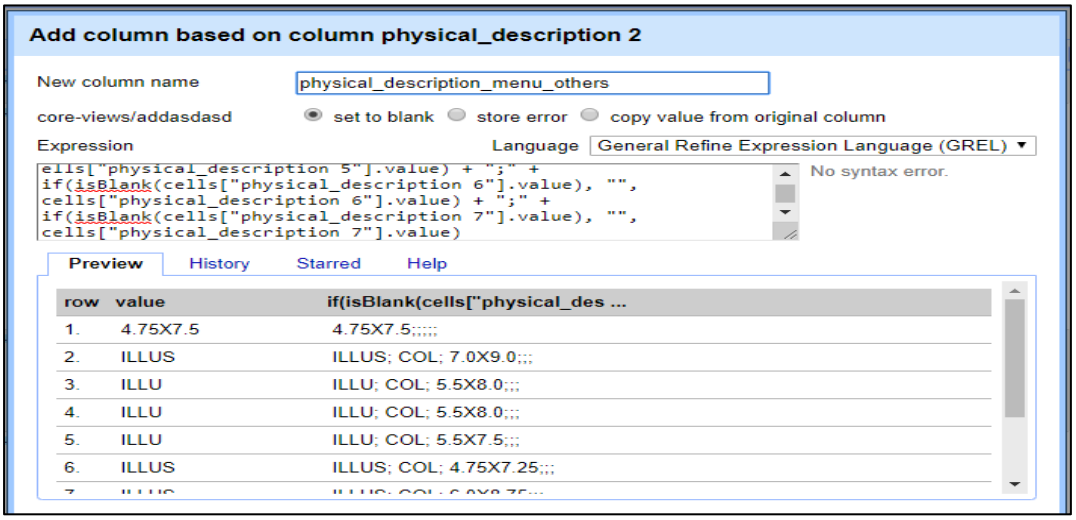


Figure 1 Open Refine physical_desciprtion

- Remove semicolon at the end of the string using expression

- Columns id, name, keywords, language, location_type, currency, currency_symbol are not updated.
- Convert all values in columns with string data type to upper case so that problem in clustering similar values is reduced considerably

II. Dish

Following operations are performed on columns in Dish dataset:

- Make a facet and perform the cluster operation using the “key-collision” method and “fingerprint” as shown below.

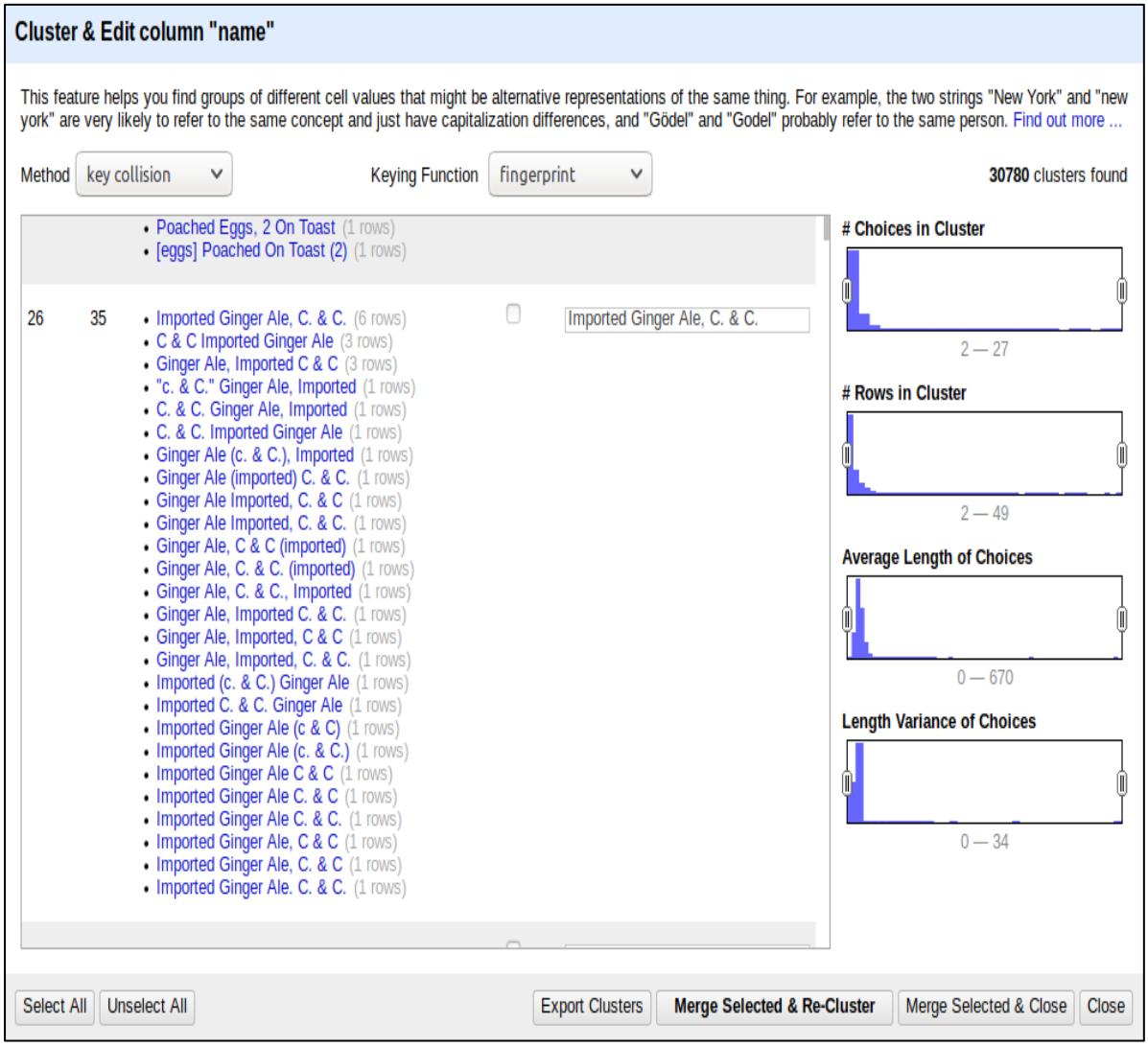


Figure 2 Open Refine facet for "name"

- Trim and Collapse white spaces in “name” column.
- Convert column “name” to title case.
- Remove special characters ““()[]{}\?#!%” in “name” column

- Make a facet and perform the cluster operation using the “key-collision” method and “ngram-fingerprint” as shown below.

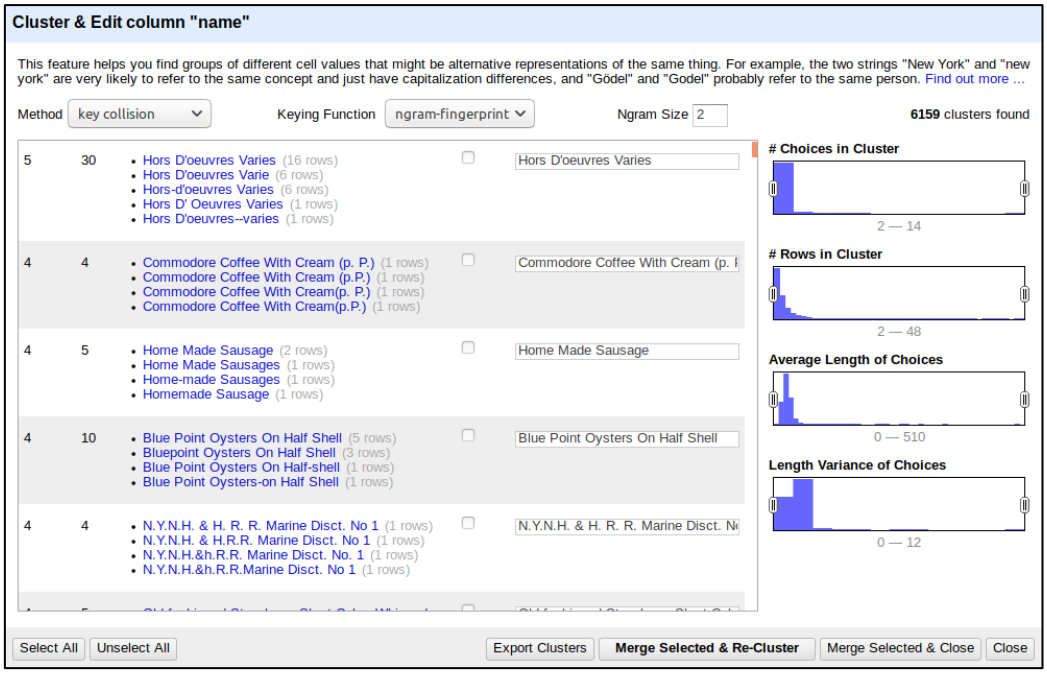


Figure 3 Open Refine facet for "name"

- Remove “description” column, since most of the entries are empty.

4. MenuPage

Following data cleaning steps were performced on the MenuPage dataset

- Convert columns to numeric type

4. MenuItem

Following data cleaning steps were performed on the MenuItem dataset.

- “high_price” column is removed
- “created_at” is transposed into “toDate()”
- “updated_at” is transposed into “toDate()”

5. Data Cleaning Results

A. Relation Database Schema

Using SQLITE we imported the above Openrefine cleaned dataset into a database “nypd.db “. Following commands shown in figure 3 are used to convert .csv files into a database and sql script.

```
bollam@bollam-desktop:~/Desktop/Illinois/DataCleaning_FinalProject/DataSetTemp$ sqlite3 nypd.db
SQLite version 3.22.0 2018-01-22 18:45:57
Enter ".help" for usage hints.
sqlite> .mode csv
sqlite> .import Dish.csv Dish
sqlite> .import Menu.csv Menu
sqlite> .import MenuItem.csv MenuItem
sqlite> .import MenuPage.csv MenuPage
sqlite> .output nypd.sql
sqlite> .dump
sqlite>
```

Figure 4 SQLite RDS Creation

After we generated the sql and nypd.db file. Using Workbench tool we have created a Entity-Relationship diagram. Below Figures explains the relationships between the Tables along with the Primary and Foreign keys.

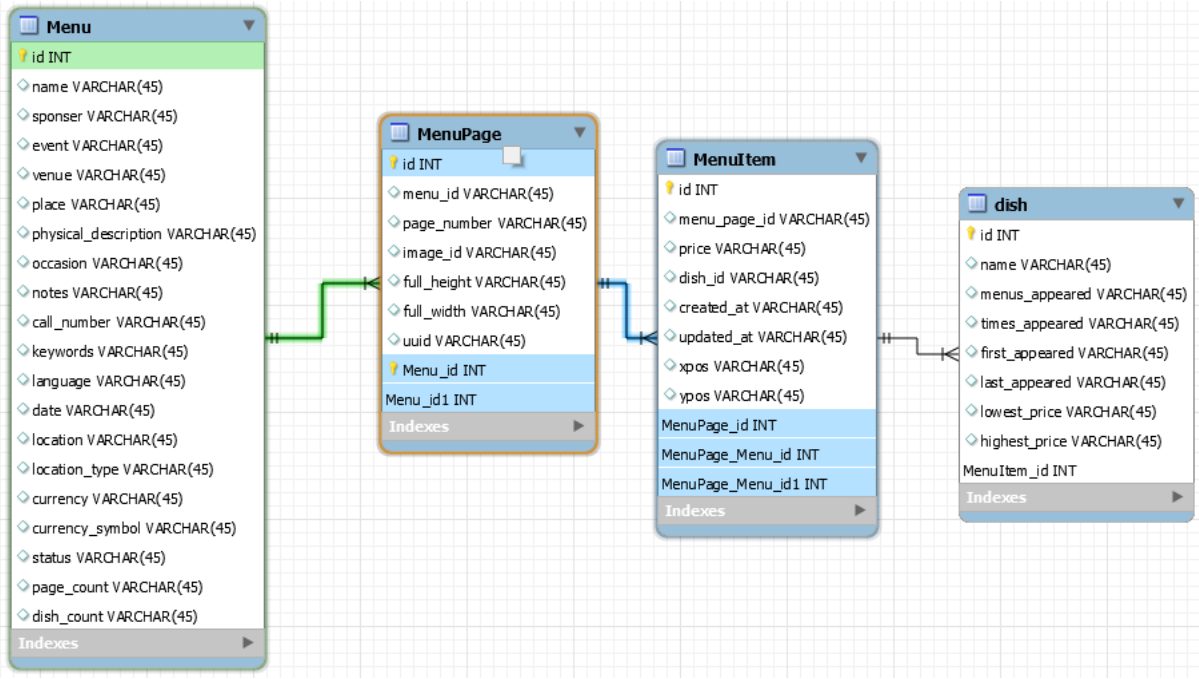


Figure 5 ER diagram Menu, MenuPage, MenuItem

Primary Keys and Foreign Keys are highlighted in green and blue in figure 4 and figure 5 are used for joining the tables for sql queries

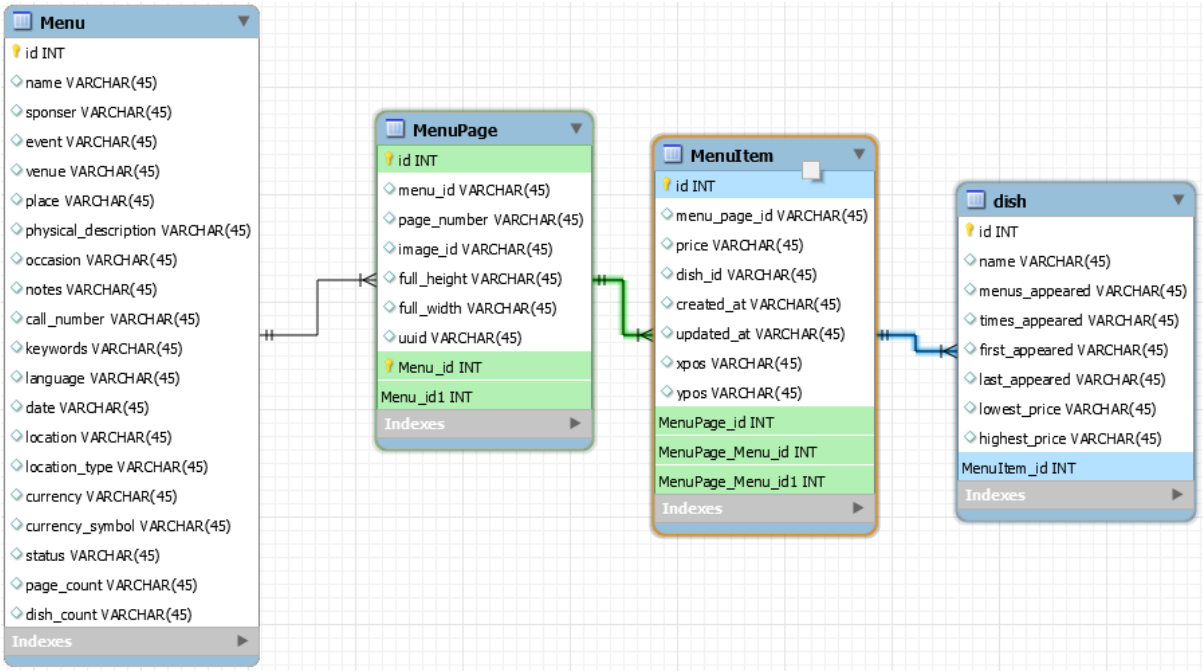


Figure 6 ER diagram MenuPage, MenuItem, Dish

B. Integrity Constraint Checks

For every table in the above generated database, few integrity constraint checks are performed and dirty data is discarded.

2.1 Dish

Following integrity constraints checks are done on dish table

- id has to be unique and not null

```
sqlite> select id, count(id) as cnt from dish group by id having cnt > 1;
sqlite>
```

- menus_appeared cannot be NULL or negative

```
sqlite> select * from dish where menus_appeared is NULL or menus_appeared < 0;
sqlite>
```

- times_appeared cannot be NULL or negative.

```
sqlite> select * from dish where times_appeared < 0 or times_appeared is NULL;
sqlite>
```

- "first_appeared" and the "last_appeared" should be in the range of 1851 and 2012.

```
sqlite> select * from dish where (first_appeared not between '1851' and '2012') and (last_appeared not between '1851' and '2012');
sqlite>
```

- "first_appeared" cannot be greater than "last_appeared". Integrity violation was observed.

```
sqlite> select * from dish where first_appeared > last_appeared;
164029,"Clear Beef Broth",0,1,1900,1851,0.25,0.25
204888,"Hot Roast Beef With Gravy",0,1,1900,1851,0.25,0.25
301736,"Cafe Glacee",0,2,1940,1851,0.4,0.4
309629,"Garlic Butter",0,1,1947,1851,0.4,0.4
sqlite>
```

- "lowest_price" and "highest_price" cannot be NULL

```
sqlite> select * from dish where (lowest_price is NULL) or (highest_price is NULL);
sqlite>
```

2.2 Menu

Following integrity constraint checks are done on menu table

- id has to be unique and not null

```
sqlite> select id, count(id) as cnt from menu group by id having cnt > 1;
sqlite>
```

- "sponsor" cannot be null

```
sqlite> select * from menu where sponsor is NULL;
sqlite>
```

- "page_count" cannot be zero

```
sqlite> select * from menu where page_count is 0;
sqlite>
```

2.3 MenuItem

- id has to be unique and not null

```
sqlite> select id, count(id) as cnt from menuitem group by id having cnt > 1;
sqlite>
```

- xpos and ypos should be in the range of 0 and 1

```
sqlite> select * from menuitem where (xpos < 0 or xpos > 1) or (ypos < 0 or ypos > 1);
sqlite>
```

2.4 MenuPage

- id has to be unique and not null

```
sqlite> select id, count(id) as cnt from page group by id having cnt > 1;
sqlite>
```

- "page_number" cannot be 0

```
sqlite> select * from page where page_number = 0;
sqlite>
```

C. Workflow Model

We used the Yesworkflow online editor to create the workflow graph for whole process

Source code for this workflow can be found in Reference Section

C.1 Or2yw Overall Workflow

The complete graph generated by the Yesworkflow in figure 7 with data and operations both graphs.

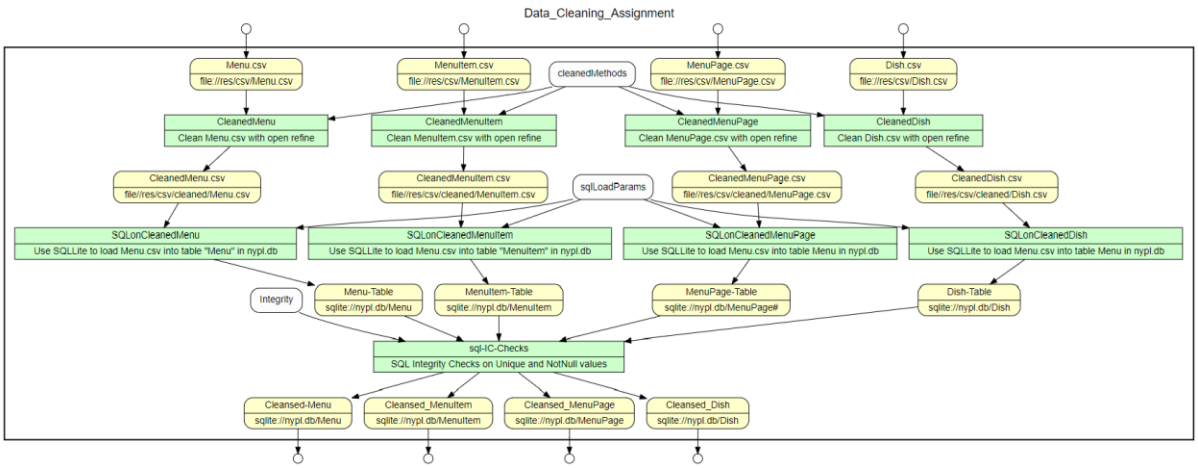


Figure 7 Overall Work flow

The complete data flow graph without the operations – Data lineage in figure 8.

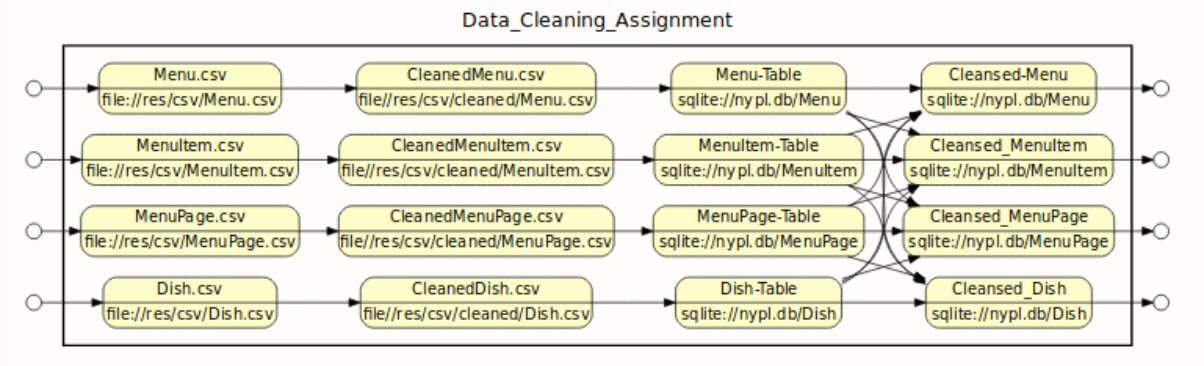


Figure 8 Overall Data Flow

Figure 9 showcases the operations graph without the data. Only operations' graph.

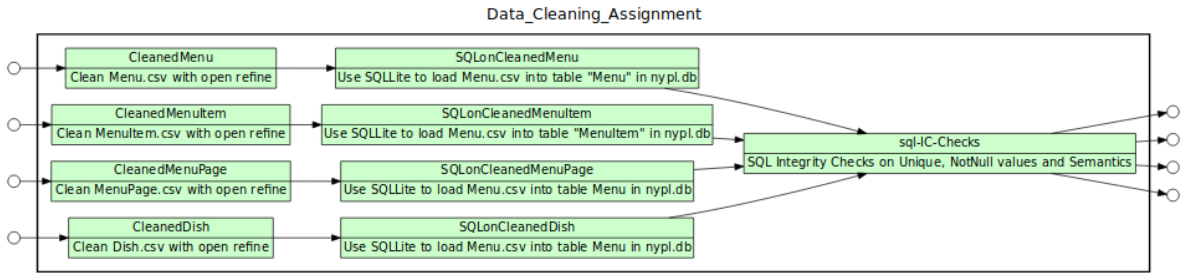
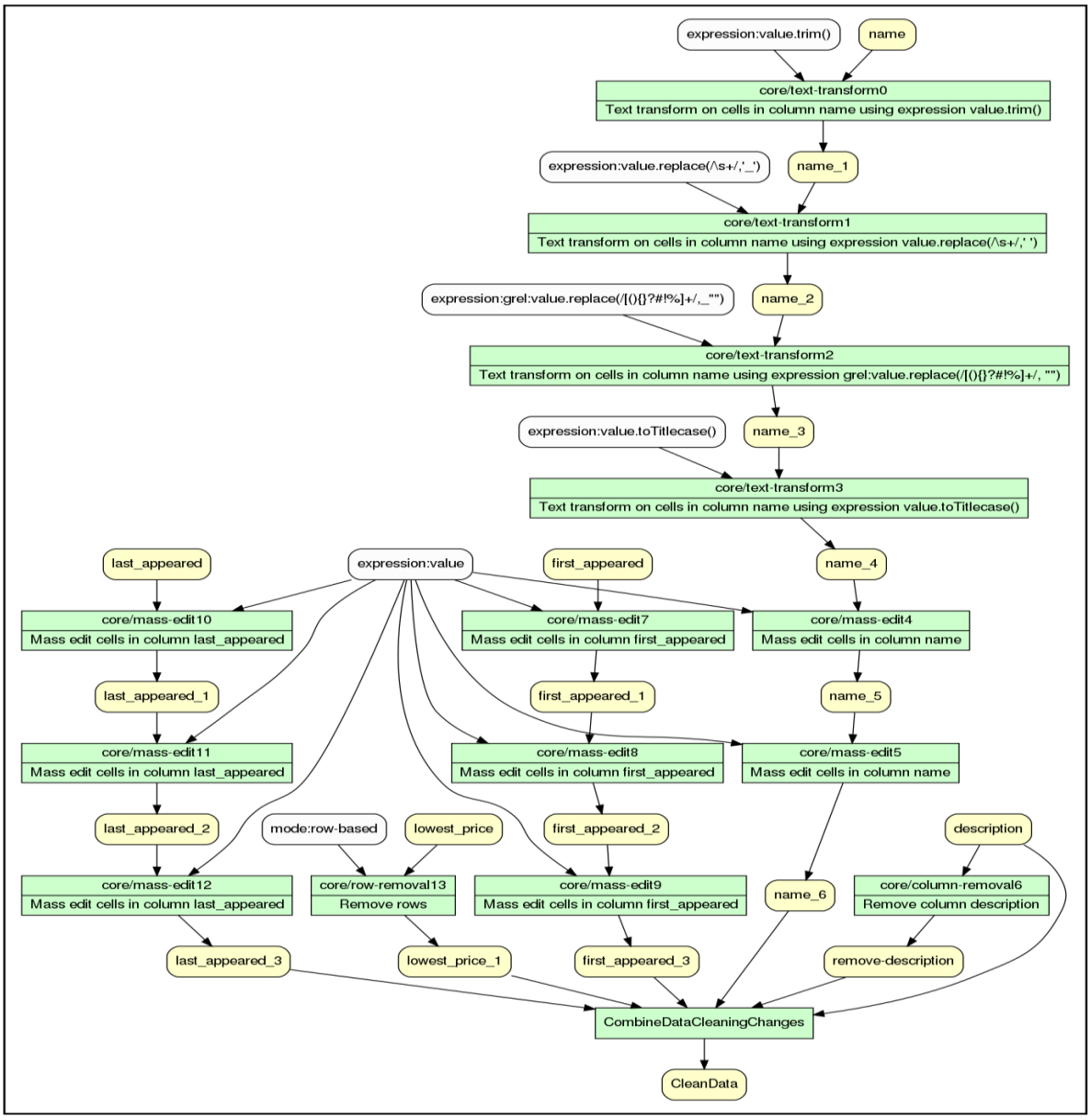


Figure 9 Overall Operations Flow

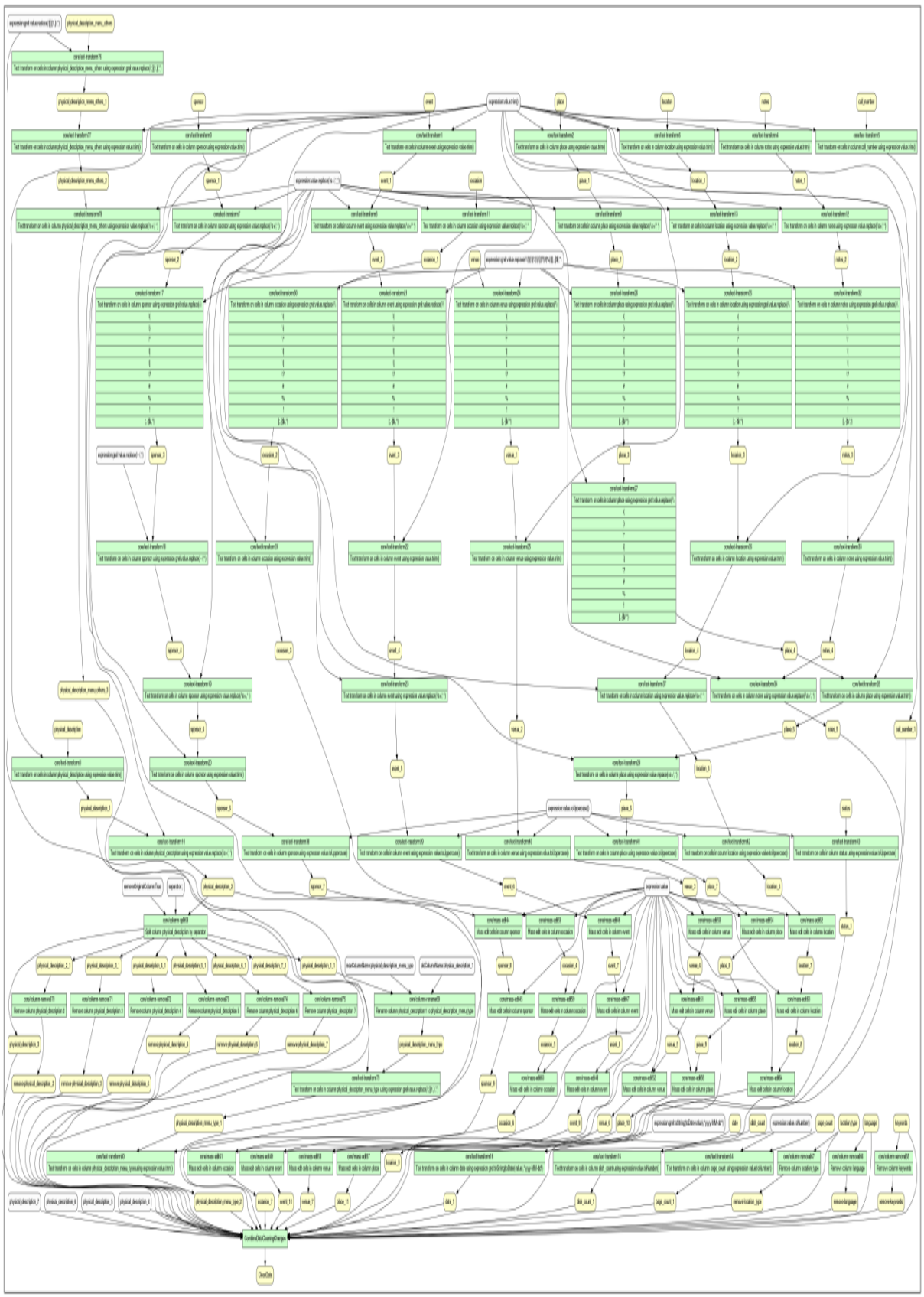
C.2 Or2yw Open Refine csv

Following graphs are generated using or2yw tool based on the recipes generated from the openrefine too.

1. Below is the workflow generated by yesworkflow tool on open refine transformation on dish dataset.



2. Below is the workflow generated by yesworkflow tool on open refine transformation on menu dataset.



6. Conclusion and Future Work

Data cleaning course helped us to understand the importance of having a cleaned dataset. NYPL being a crown sourced dataset is full of issues and needs good amount of cleaning activities. Majority of the cleaning works were required for string-based columns in the dataset files.

Main outcome of our cleaning project is to create a cleaned dataset. We also tried our best to create a yesworkflow visualization so that its easier for clients to get visual details about the cleaning activities performed.

An important takeaway from this cleaning activity is to first get an understanding about the importance of cleaned dataset. This was understood once we investigated the NYPL dataset and cleaned data was user readable and one could develop better understanding of the information. A simple removal of leading and trailing whitespaces and removal of consecutive white spaces bring so much consistency to the dataset. It is utmost importance to create a cleaned dataset. From NYPL dataset it was also understood that crowd source dataset needs a lot of cleaning activities.

There were number of challenges:

Datasets are huge are requires faster CPU and more RAM. Performing neighborhood clustering with PPM distance function consumed 100% CPU and openrefine hanged several occasions. Even after increasing the heap memory for openrefine it did not help. Openrefine tools have limitations to handle large dataset so there is a need to use python which is more efficient in handling large datasets.

As NYPL is a crowd sourced dataset there were many issues with characters might be from different foreign language. There are many which needs to be studied to completely comprehend the true meaning. Documentation related to understanding the NYPL dataset was limited.

There were instances when we had to assume things. For example, physical_description column is Menu.csv where no information was included but we performed steps based on self-understanding. Columns like notes in menu contain lot of information which needs to be proceeds but were not due to lack of understanding.

Future additional tasks:

In spite of no of cleaning activities perform on NYPL it does seems that it is still not near to perfection. There is further cleaning required to bring NYPL to a consumable state.
Further machine learning models could be run to infer more details about the dataset.
For Menu we could use status column having two values “Complete” and “Under Review” and perform classification task. We could try to predict which features in menu table led to the value Complete and Under Review.

7. Acknowledgment

We would like to thank Prof. Bertram Ludaescher for his deep dive in data cleaning course and his guidance in understanding the usefulness of data cleaning. We also thank all the TAs for their support.

Appendix A

Dataset Fields Descriptions

Dish.csv

Table Reference Notes:

i. id: a unique identifier for this dish

ii. name: name of the dish

iii. description: description of the dish; always blank. So remove in cleaned dataset

iv. menus_appeared: number of menus this dish appears on

v. times_appeard: number of times this dish appears

vi. first_appread: first year this dish appeared.

vii. last_appreared: last year this dish appeard

viii. lowerst price: lowest price that this item was sold for

ix. highest price: highest price this item was sold for

Size:

Number of entries in the File:

A.2 Menu.csv

Table Reference Notes:

i. id: unique identifier for this menu

ii. name: the name of this menu

iii. sponser: the sponser of this menu (name of the restaurant)

iv. event: event for which this menu was created

v. venue: the venue where food from this menu was served

vi. place: geographical location of this menu, in other words address

vii. physical_description: physical characteristics description of this menu

viii. occasion: any special occasion for the occurrence of this menu

ix. notes: any special notes about this menu

x. call_number: the menu's call number with the NYPL connection

xi. keywords: keywords for the menu;

xii. language: language the menu is printed

xiii. date: date the menu was collected

xiv. location: location where the menu was used

xv. location_type: type of the location value;

xvi. currency: price

xvii. currency_symbol: symbol of the currency

xviii. status: status

xix. page_count: number of the pages in the menu

xx. dish_count: number of dishes on the menu

A.3 MenuItem.csv

Table Reference Notes:

i. Id:: a unique identifier for the menu item

ii. menu_page_id: the id of the menu page

iii. price: price of the item in the menu

iv. high_price: highest price of the item in the menu

v. dish_id: the id of the dish

vi. created_at: time of creation of the menu item

vii. updated_at: last updated time of the menu item

viii. xpos: the x-axis position of the item on the scanned image

ix. ypos: the y-axis position of the item on the scanned image

A.4 MenuPage.csv

Table Reference Notes:

i. Id: a unique identifier for the menu item

ii. menu_id: id of the menu this page belongs to

iii. page_number:: page number in the menu

iv. image_id: image id of the scanned menu page

v. full_height: height of the scanned menu page

vi. full_width: width of the scanned menu page

vii. uuid: unique id

References

R.1 Original DataSet from http://menus.nypl.org/

R.2 GitHub

Note: Github repo contains original dataset as a zipped file due to 100MB file limitation.

***