CS513: THEORY & PRACTICE OF DATA CLEANING

Final Project – Phase 1

Team64

Prashanth Giriyappa Shankarappa - pg14@illinois.edu
Navdeep Parmar - navdeep2@illinois.edu
Shubhendu Bhaskar - sb59@illinois.edu

Table of Contents

Introduction

This document summarizes preliminary analysis done by our team for CS513- Data cleaning project.

1. Chosen dataset

For the data cleaning project, we are choosing “The New York Public Library” menu’s dataset.

2. Develop Use cases

Target/Main Use case(U_1):

Query: Find the highest price of dishes (dish.csv) along with their names that were available at the dinner (menu.csv -event) at “Government” (menu.csv - venue) on the occasion of “Anniversary” (menu.csv - Occasion) irrespective of the currency.

AND/OR

Query: What are the different varieties of wines were in the menu in 1900s vs 2000s in each of the Venue setting (Commercial, Government, SOC etc.) on an Occasion of “Anniversary”?

AND/OR

Query: What are the different styles of eggs as a breakfast option were in the menu in 1900s vs. 2000s in each of the Venue setting (Commercial, Government, SOC etc.)?

AND/OR

Query:

In 2000s, what were some of the places that you could go for Lunch and Dinner that served traditional and new dishes?

No Cleaning required Use case (U_0):

Query: How many dishes are similar or repeated in the Dishes Dataset?

Not Sufficient cleaning Use case (U_2):

Query: How many menus were non-English and were Handwritten?

3. Dataset description

The New York Public Library has collected and created menus dataset which includes description of culinary details and price associated with items that constitute those menu items. “What’s on the menu” dataset can be accessed using the URL http://menus.nypl.org/data.

The latest version of this dataset is 07/01/22, we will however use an earlier version of this dataset possibly from the year 2021. Menus from 1850’s to 2015 are listed in the dataset. The dataset consists of four “.csv” files that are listed below.

Menu

Menu dataset consists of features that can be associated with a menu like the name, physical description and dimension, number of pages, number of dishes, currency, and status. Other columns listed in this dataset are related to the type of Venue where these menus were served. While the US Dollar seems to be the dominant currency listed on the menu, there are other foreign currencies like Deutsche Marks, Francs etc. associated with the menu. There are 14,281 rows in this dataset.

Column names: id, name, sponsor, event, venue, place, physical_description, occasion, notes, call_number, keywords, language, date, location, location_type, currency, currency_symbol, status, page_count and note

Dish

Dish dataset has names of dishes that are mentioned on a menu. It has other attributes like low price, high price and year when the dish appeared first on a menu and the last time it was updated. Many dishes have price listed on them, but there are dishes that don't have price listed or the price is 0. There is also a count of the number of times a dish has appeared on menus and the number of times they have appeared.

Column names: id, name, description, menus_appeared, times_appeared, first_appeared, last_appeared, lowest_price, highest_price

Menu Page

Menu can have multiple pages, specification about each menu page is represented in the menu page dataset.

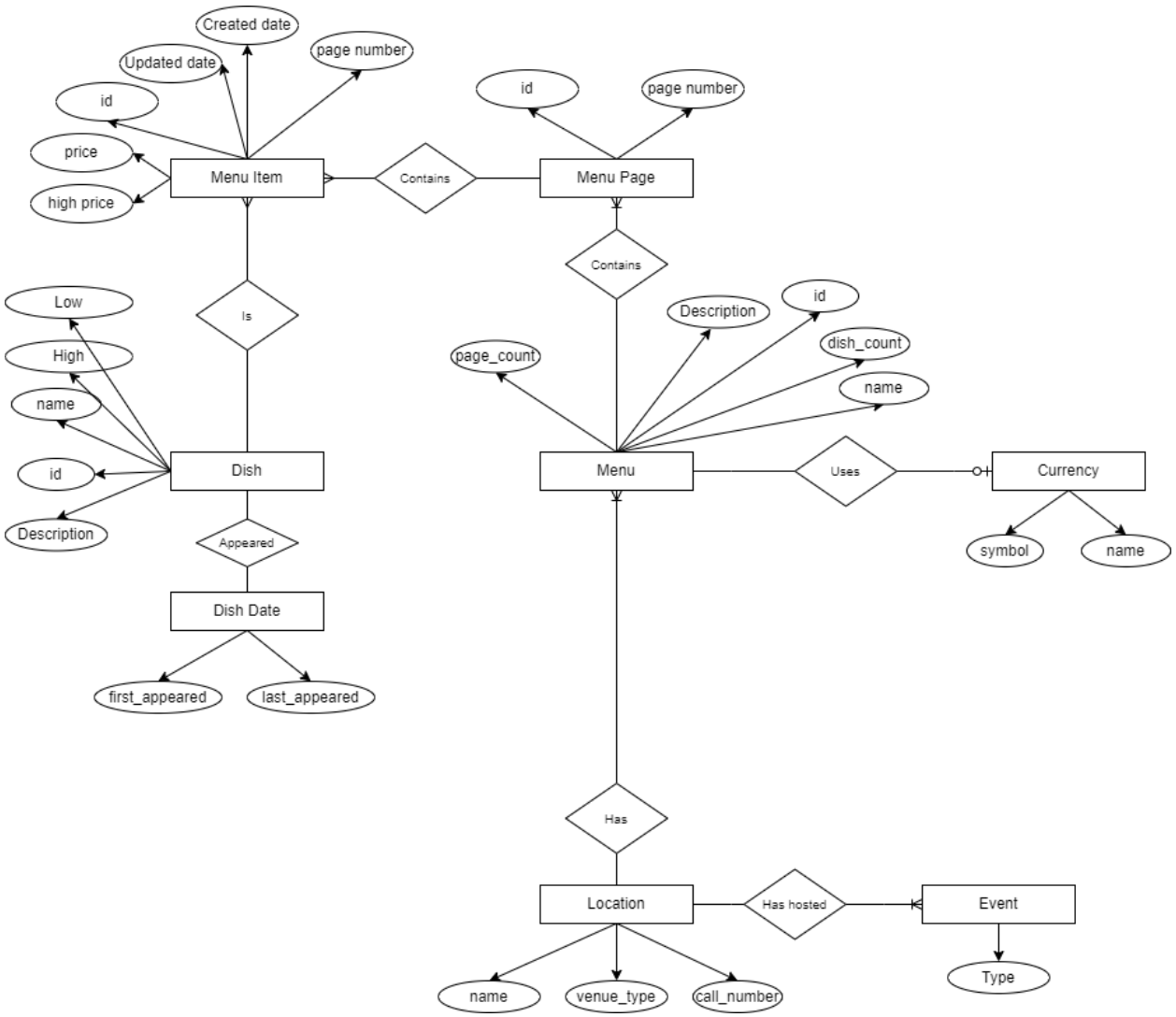
Column names: id, menu_id, page_number, image_id, full_height, full_width, uuid

Menu Item

Each dish is listed on a menu page, this information is presented in the Menu item dataset. Each page of the menu has a specification, the menu item references the position on the page where the dish is listed on the menu page. It combines dish id, menu page id with width and height positions of a menu page.

Column names: id, menu_page_id, price, high_price, dish_id, created_at, updated_at, xpos, ypos

Data Model: Entity relationship visualized



Create Table Statement:

Please refer to the below SQL create table statements:

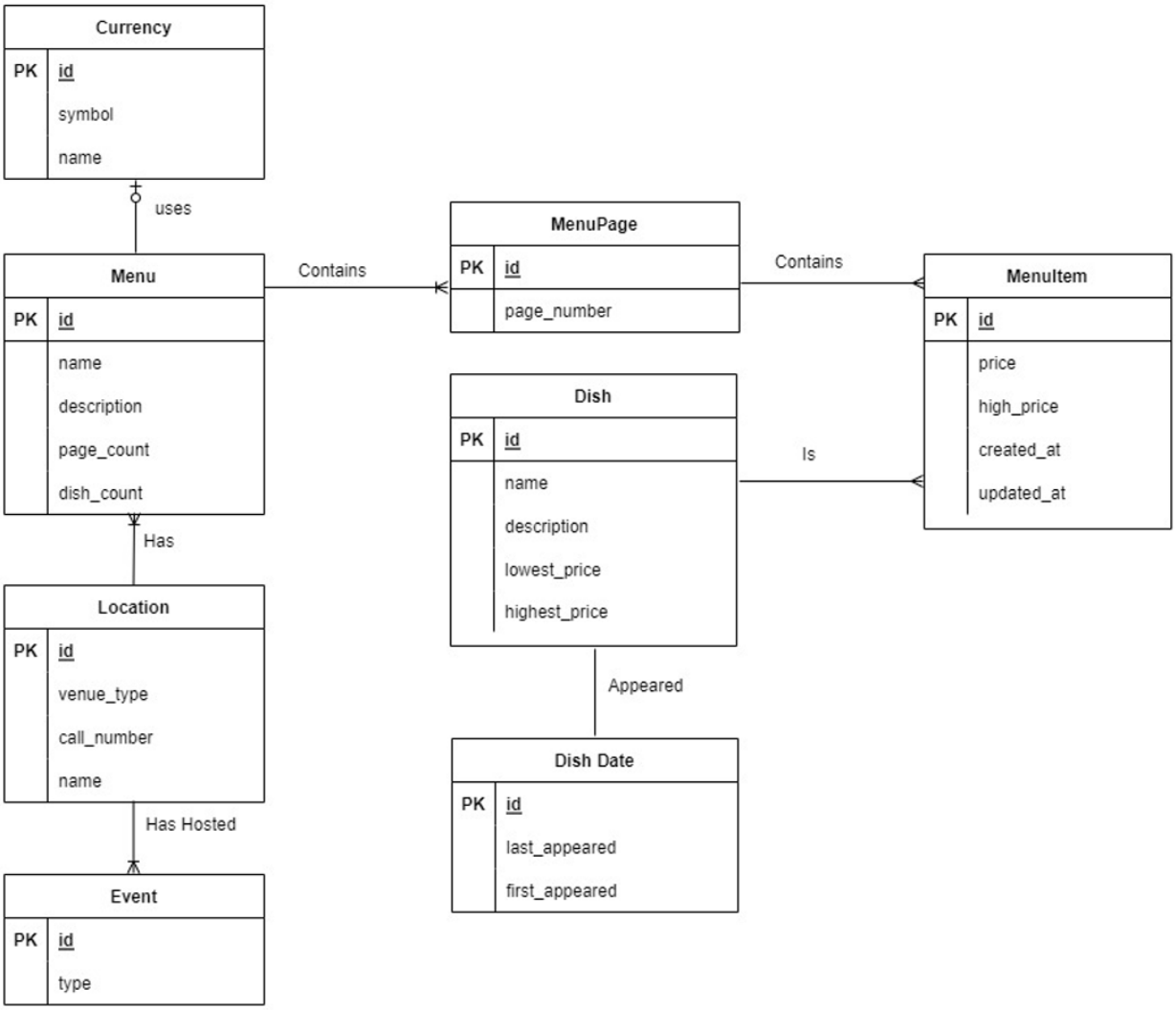
Create_tables_statements.sql

```
CREATE TABLE `Currency` (
  `id` int NOT NULL,
  `name` varchar(512) DEFAULT NULL,
  `symbol` varchar(512) DEFAULT NULL,
  PRIMARY KEY (`id`)
)

CREATE TABLE `Event` (
  `id` int NOT NULL,
  `type` varchar(512) DEFAULT NULL,
  PRIMARY KEY (`id`)
)

CREATE TABLE `Location` (
  `id` int NOT NULL,
  `venue_type` varchar(512) DEFAULT NULL,
  `call_number` varchar(512) DEFAULT NULL,
  `event_id` int NULL,
  PRIMARY KEY (`id`),
  KEY `event_id` (`event_id`),
  CONSTRAINT `loc_event_fk` FOREIGN KEY (`event_id`) REFERENCES `Event` (`id`)
)


CREATE TABLE `DishDate` (
  `id` int NOT NULL,
  `first_appeared` int DEFAULT NULL,
```

```
  `last_appeared` int DEFAULT NULL,
  PRIMARY KEY (`id`)
)

CREATE TABLE `Dish` (
  `id` int NOT NULL,
  `name` varchar(512) DEFAULT NULL,
  `dish` varchar(512) DEFAULT NULL,
  `description` varchar(512) DEFAULT NULL,
  `lowest_price` double DEFAULT NULL,
  `highest_price` double DEFAULT NULL,
  `dish_date_id` int NULL,
  PRIMARY KEY (`id`),
  KEY `dish_date_id` (`dish_date_id`),
  CONSTRAINT `dish_date_fk` FOREIGN KEY (`dish_date_id`) REFERENCES `DishDate` (`id`)
)

CREATE TABLE `Menu` (
  `id` int NOT NULL,
  `name` varchar(512) DEFAULT NULL,
  `description` varchar(512) DEFAULT NULL,
  `page_count` int DEFAULT NULL,
  `dish_count` int DEFAULT NULL,
  `currency_id` int NULL,
  `dish_id` int NULL,
  `location_id` int NULL,
  PRIMARY KEY (`id`),
  KEY `currency_id` (`currency_id`),
  CONSTRAINT `menu_curr_fk` FOREIGN KEY (`currency_id`) REFERENCES `Currency` (`id`),
  KEY `location_id` (`location_id`),
  CONSTRAINT `menu_location_fk` FOREIGN KEY (`location_id`) REFERENCES `Location` (`id`)
)
```

```
CREATE TABLE `MenuPage` (
  `id` int NOT NULL,
  `page_number` int DEFAULT NULL,
  `menu_id` int NULL,
  PRIMARY KEY (`id`),
  KEY `menu_id` (`menu_id`),
  CONSTRAINT `menuPage_menu_fk` FOREIGN KEY (`menu_id`) REFERENCES `Menu` (`id`)
)

CREATE TABLE `MenuItem` (
  `id` int NOT NULL,
  `price` double DEFAULT NULL,
  `high_price` double DEFAULT NULL,
  `created_at` timestamp DEFAULT NULL,
  `updated_at` timestamp DEFAULT NULL,
  `page_number` int DEFAULT NULL,
  `menu_page_id` int NULL,
  PRIMARY KEY (`id`),
  KEY `menu_page_id` (`menu_page_id`),
  CONSTRAINT `menuitem_page_fk` FOREIGN KEY (`menu_page_id`) REFERENCES `MenuPage` (`id`)
)
```

Database Schema Diagram:



Entities and Relation

Menu and Currency

A menu has an associated currency. A menu from a sponsor for an event can have different currency.

Menu and Location

A location can serve multiple menus for different events

Dish and Dish date

Each dish has a dish date with first appeared and last appeared date.

Dish and Menu Item

A dish can appear on multiple menu items.

Menu item, Menu Page and Menu

A menu page has multiple menu items and multiple menu pages constitute a menu.

4. Identified data quality problems

The NYPL dataset we have chosen consists of 4 csv files that has relationship. We would discuss the data quality issues one by one

1. Menu.csv
   - The “event” column has the same values with different syntaxes and extra spaces, for e.g. we see the presence of “(DINNER)” and “(?DINNER?)”.

event
(DINNER)
(?DINNER?)

   - The above-mentioned discrepancy also exists for “venue”, “place”, “occasion”, “sponsor” columns.

place
"KONIGEN LUISE" AT SEA
"KONIGIN LUSE" AT SEA
"KONIGIN LUISE"

venue
(SOC?);
(SOC);

occasion
(ANNIV CELEBRATION)
(ANNIV);

sponsor
(FIFTH AVENUE HOTEL)
(FIFTH AVENUE HOTEL)
(FIFTH AVE. HOTEL)

   - Columns “keywords”, “language”, “location_type” are blank and do not have a value for any row.
2. MenuItem.csv
   - The “price” column does not have values for all the rows in this dataset.
3. MenuPage.csv
   - The column “uuid” appears to be an id of some item but is not establishing any relationship with the other parts of the dataset.
4. Dish.csv
   - The “name” column may have extra spaces appended to the values.

name
" " Brut

   - The “description” column does not have any values for any of the rows.
   - We see some negative values in the column “times_appeared” which conflict with the column relevance.

times_appeared
-2
-6
-3

   - “first_appeared” has some single digit in years which makes it highly unlikely in relation to the dataset.

first_appeared
1
1
1

- "last_appeared" has some years beyond the current year, again which is not relevant to the column name.

last_appeared
2928
2928
2928

5. Implementation plan

Define need

- Establish a need for using this dataset and formulate use cases that support using the dataset. This is stated in section 1-Develop user cases.

Perform initial analysis

- Understand the dataset through FAQ's and other help documentation provided. How can we interpret fields and their values?
- Establish conceptual data model using Entity relationship and database schema diagrams. Understand the entities and interactions between entities through their relationships with each other.

Analyze dataset Quality

- Profile the dataset
    - Data completeness: Our chosen dataset contains transcribed data from menus in 1850's to 2015. Three would be differences in how menus and dishes were presented in the past versus more recent versions. We plan to assess and pick relevant data while ignoring or removing incomplete data.
    - Perform cursory analysis using OpenRefine to profile the data.

Identify different dimensions of data quality issues

- Quantitative errors
    - Detect outliers and take corrective action or eliminate them
- Qualitative errors
    - Syntax violations: Use Open refine to analyze cluster values, check spellings, format of fields, incorrect values
    - Semantic violations: Check integrity violations between identified entities
    - Duplicate values detection

Correct data errors

- Syntax violations
    - Use OpenRefine to cluster values, merge based similar clusters,

    - Check for format of fields using regular expressions
    - Trim the leading and trailing blank spaces off the values
    - Remove the columns which do not have values
    - Convert numerical relevant columns values to number type using common transformations
    - Similarly, transform date relevant columns to date format
- Semantic violations:
    - Create SQLite tables and run SQL to check for integrity violations based on established relationships
- Use SQL to detect and remove duplicate records

Document and Communicate actions taken (Provenance)

- Use OpenRefine to capture action history

Assess impact to established need

- How does the cleaned data help us find a response to our use case?
- What are some of the challenges encountered?

6. References

- Handbook of Data Quality by Shazia Sadiq