“What’s on the Menu?” - Phase 1

Project Authored By: Scott Swanson

Course: CS513 at University of Illinois Urbana-Champaign

Date: Summer 2024

Dataset Courtesy of The New York Public Library

Narrative and Description

The dataset provided by the New York Public Library's "What's on the menu?" project consists of digitized historical restaurant menus. The data includes information about dishes, menus, menu items, and menu pages from various restaurants. The temporal extent of the data spans from the late 19th century to the present day, covering a wide geographic area primarily within The United States. Each menu captures the culinary offerings and economic conditions of its time, providing a valuable historical record.

The dataset is structured into four main entities: Dish, Menu, MenuItem, and MenuPage, with relationships linking menu items specific dishes and pages, and pages to menus. The following is a breakdown of the data model itself:

Entity Descriptions and Attributes

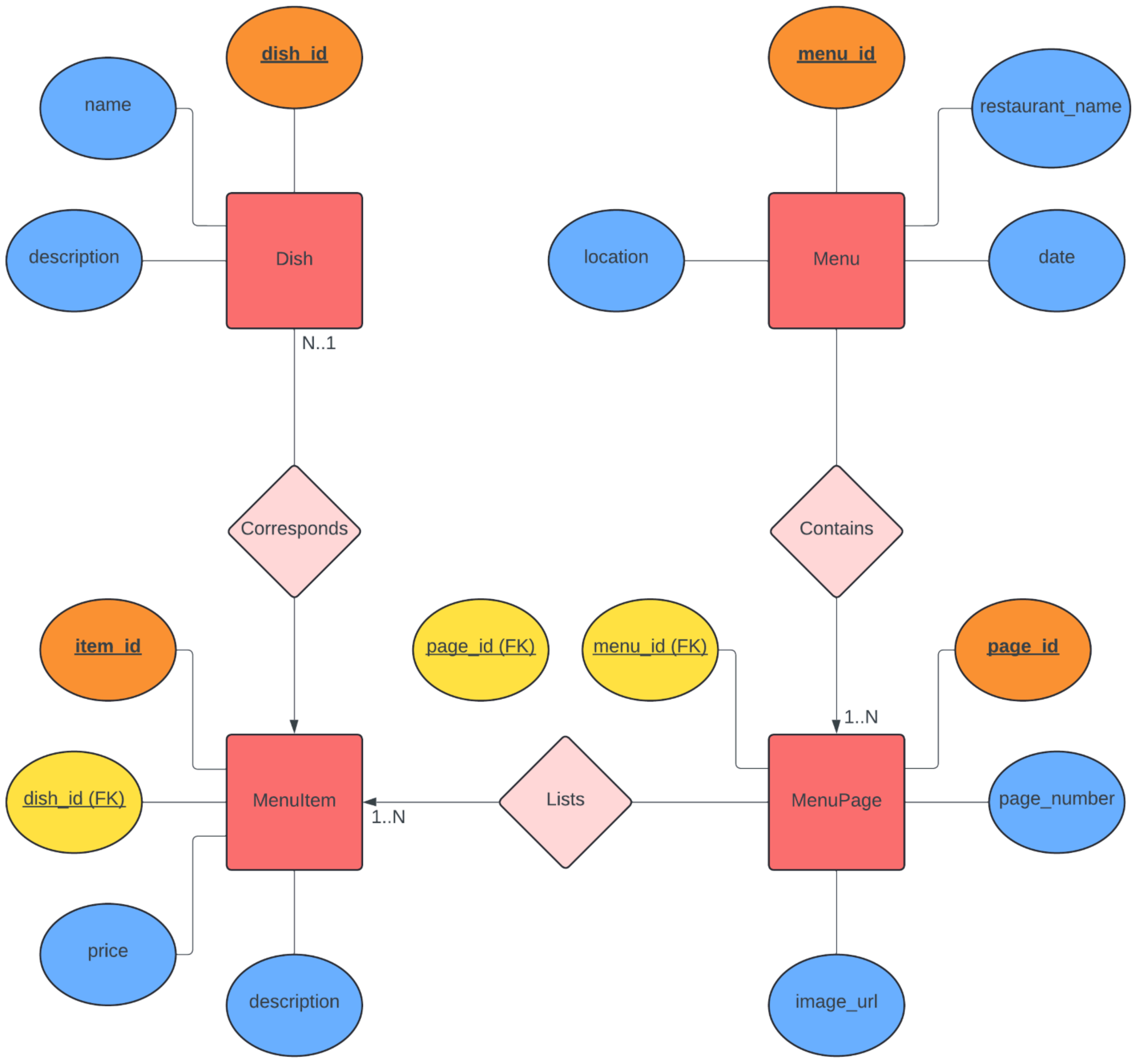
1. Dish
    a. `dish_id` (Primary Key): A unique identifier for each dish.
    b. `name`: The name of the dish.
    c. `description`: A textural description of the dish, which may include ingredients, preparation style, or other relevant details.
2. Menu
    a. `menu_id` (Primary Key): A unique identifier for each menu.
    b. `restaurant_name`: The name of the restaurant offering the menu.
    c. `date`: The date when the menu was available, providing a temporal context.
    d. `location`: The location of the restaurant, which can include city and state or more details address information.
3. MenuPage
    a. `page_id` (Primary Key): A unique identifier for each menu page.
    b. `menu_id` (Foreign Key): The identifier linking the page to a specific menu.
    c. `page_number`: The page number within the given menu, indicating its order in the menu's context.
4. MenuItem
    a. `item_id` (Primary Key): A unique identifier for each menu item.
    b. `page_id` (Foreign Key): The identifier linking the item to a specific menu page.
    c. `dish_id` (Foreign Key): The identifier linking the item to a specific dish.

d. `price`: The price of the dish as listed on the menu, which may vary over time and location.
e. `description`: A textural description of the menu item, which may include variations or special notes distinct from the general dish description.

Data Relationships

- `Menu` contains `MenuPage`: Each `Menu` can contain multiple `MenuPages`, representing the physical pages of a given menu.
- `MenuPage` lists `MenuItem`: Each `MenuPage` can list multiple `MenuItems`, representing the individual items listed on the given menu page.
- `MenuItem` corresponds to `Dish`: Each `MenuItem` corresponds to a specific `Dish`, allowing for the identification of the dish across different menus and time periods.

Entity-Relationship Diagram



Use Cases

Main Use Case: Analyzing Historical Pricing Trends of Dishes

This use case involves analyzing the pricing trends of specific dishes over time. The goal is to track how the prices of popular dishes have changed across different periods and locations. By examining the historical pricing data, we aim to identify trends, such as seasonal variations, inflation impacts, and changes in culinary preferences over time. This analysis will involve:

- **Identifying Popular Dishes**: Determine which dishes appear most frequently across different menus.
- **Tracking Price Changes**: Analyze how the prices of these popular dishes have evolved over the years, taking into account different locations and restaurant types.
- **Comparative Analysis**: Compare the pricing trends of similar dishes across different regions or time periods to identify any notable differences or patterns.

Example Query:

```sql
SELECT d.name, m.date, mi.price
FROM menu_item mi
INNER JOIN dish d ON mi.dish_id = d.id
INNER JOIN menu_page mp ON mi.menu_page_id = mp.id
INNER JOIN menu m ON mp.menu_id = m.id
WHERE d.name IN ('Cheeseburger', 'Lobster Bisque')
ORDER BY m.date;
```

Zero Data Cleaning Use Case: Counting the Number of Menus in the Dataset

This use case simply counts the total number of menus in the dataset. Since this operation only involves counting records, no data cleaning is required.

Example Query:

```sql
SELECT COUNT(*) AS num_menus
FROM menu;
```

Never Enough Use Case: Analyzing Nutritional Content of Dishes

This use case attempts to analyze the nutritional content of dishes. However, the dataset does not contain any nutritional information, and no amount of data cleaning can add this missing data.

NOTE: Just to be clear, the query below will never work because the nutritional data and columns referenced do not exist.

Example Query:

```sql
SELECT d.`name`, d.calories, d.protein, d.fat
FROM dish d
WHERE d.calories IS NOT NULL;
```

Data Quality Problems

Missing Data

- Missing `description` in `Dish.csv`: While not directly impacting price analysis, having descriptions can help in accurately identifying and categorizing dishes.
- Missing `lowest_price` and `highest_price` in `Dish.csv`: These need to be imputed to ensure accurate trend analysis.
- Missing `price` and `high_price` in `MenuItem.csv`: Critical for accurate price tracking and trend analysis.
- Others
    - Menu.csv
        - *name*: Most values are missing.
        - *event*, *venue*, *place*, *occasion*: High Percentage of missing values.
        - *currency* and *current_symbol*: Considerable number of missing values.
    - MenuItem.csv
        - *dish_id*: Some values are missing.

Inconsistent Formatting

- Inconsistent `date` formats in `Menu.csv`: Standardizing date formats is essential for accurate temporal analysis.
    - Needs standardization to a single format, for example, ISO 8601.

Duplicate Entries

- Duplicate entries in `Dish.csv`, `Menu.csv`, and `MenuItem.csv`: Removing duplicates ensures the integrity of the analysis.
    - Duplicates can occur if the same dish, menu, or menu item is entered multiple times.

Incomplete Data Fields

- Incomplete price information in `Dish.csv` and `MenuItem.csv`: This can skew the results of the price trend analysis.

Inconsistent Data Entries

- Variations in dish names and location entries: Standardizing these entries ensures accurate grouping and analysis.
    - For example, "Spaghetti Bolognese" vs. "Spaghetti Bolognaise" and "New York, NY" vs. "NYC, New York"

Examples of Dirty Data

`Dish.csv`

- 132844,Spaghetti Bolognaise,,46,55,1932,1968,0.0,28.0
- 132508,"Gefulltes Ei ""Paulusstuben""",,1,1,0,0,0.0,0.0
- 132570,"Reis, Brat-, Petersilienkartoffel, Sauerkraut, Rotkraut, Semmel-, od. Kartoffelknodel",,1,1,0,0,0.0,0.0

`Menu.csv`

- 12583,"",,,,,,,,,,1900-04-15,Hotel Eastman,,,,complete,2,67
- 16584,,ST. CHARLES HOTEL,DINNER,COMMERCIAL,"MILWAUKEE, WI",FOLDER; 5X6.5;,"",TABLE D'HOTE MENU;,1900-5039,,,1900-12-16,St. Charles Hotel,,,,complete,4,39

`MenuItem.csv`

- 1,1389,0.4,,1,2011-03-28 15:00:44 UTC,2011-04-19 04:33:15 UTC,0.111429,0.254735
- 2267,168,0.2,,1617,2011-04-19 23:17:41 UTC,2011-04-19 23:17:41 UTC,0.752857,0.805247
- 2292,134,,,1632,2011-04-19 23:33:51 UTC,2011-04-19 23:33:51 UTC,0.784286,0.848199

Planning

Data Cleaning

1. Handle Missing Data
    a. `Dish.csv`
        i. Impute missing `description` with “No description available,” `NULL`, or remove the column if it is not useful.
        ii. Impute `lowest_price` and `highest_price` using mean, median, or another suitable method.
    b. `Menu.csv`
        i. Impute missing `name`, `event`, `venue`, `place`, `occasion`, with “Unknown,” `NULL`, or another context-specific default value(s).
        ii. Impute missing `currency` and `currency_symbol` with the most common currency in the dataset (e.g., USD, $).
            1. We could use a more intelligent way of doing this that uses the location as a reference to infer the value. This is how I would do it in production, but I am not sure if it is overkill for this project or not.
    c. `MenuItem.csv`
        i. Impute missing `price` and `high_price` with the mean or median price of related items.
            1. Defining what is “related” is another task.
        ii. For missing `dish_id`, attempt to match items to dishes based on `name` or remove it no match can be found.
2. Standardize Data Formats
    a. `Menu.csv`
        i. Standardize date format to ISO 8601.
3. Remove Duplicates
    a. Identify and remove duplicate entries in all datasets using unique identifiers and a duplicate detection algorithm.
4. Validate and Clean Data Entries
    a. `Dish.csv`
        i. Standardize name entries’ naming conventions.
    b. `Menu.csv`
        i. Standardize location entries’ naming conventions.
5. Ensure Referential Integrity
    a. Verify that all foreign key relationships are valid.

i. Ensure that `menu_id` in `MenuPage.csv` exists in `Menu.csv`.
ii. Ensure that `page_id` in `MenuItem.csv` exists in `MenuPage.csv`.
iii. Ensure that `dish_id` in `MenuItem.csv` exists in `Dish.csv`.

Improved Dataset Validation

1. Run Validation Queries
    a. Check that the cleaned dataset supports the main use case by running the necessary queries and verifying the results.
    b. Compare the results from before and after cleaning to ensure improvement(s).

Document Changes

1. Track Changes Made
    a. Document all changes made to the dataset, including imputation methods, standardization rules, and any removed or corrected records.
        i. We will use tools like OpenRefine's history and Python scripts to keep track of changes.

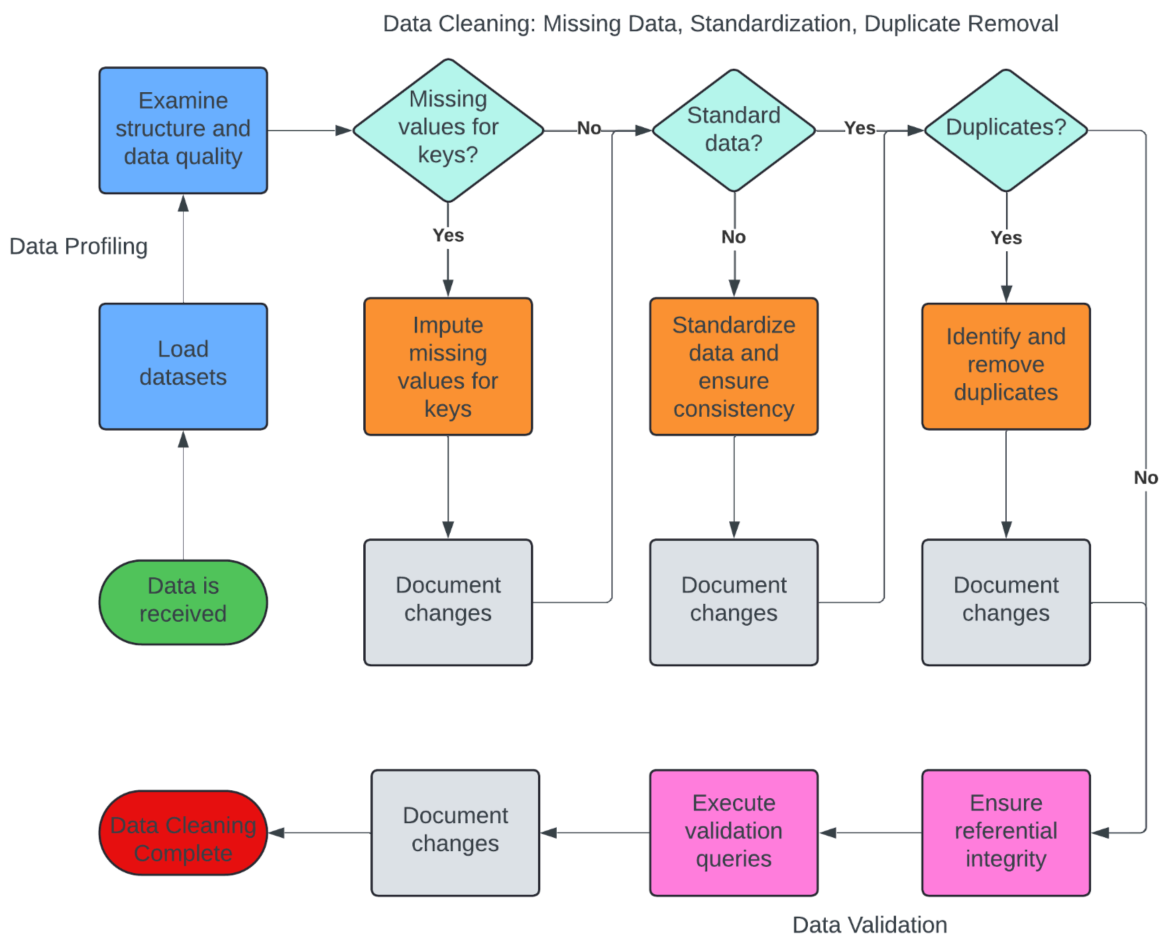
Tools

- OpenRefine: For interactive data cleaning, especially for handling missing data, duplicates, and standardization.
- Python: For scripting more complex cleaning operations and ensuring referential integrity.
- SQL: For validation of relationships and running queries to check data consistency and validating referential integrity constraints.
- Datalog: This may be used to reinforce confidence in referential integrity constraints.

Timeline

- **Week 1**: Data profiling and initial handling of missing data.
- **Week 2**: Standardize formats and remove duplicates.
- **Week 3**: Ensure referential integrity and validate data entries.
- **Week 4**: Document changes and prepare the Phase 2 report.

Workflow Diagram


Data Cleaning: Missing Data, Standardization, Duplicate Removal
Data Profiling
Data is received
Load datasets
Examine structure and data quality
Missing values for keys?
Yes
Impute missing values for keys
Document changes
No
Standard data?
No
Standardize data and ensure consistency
Document changes
Yes
Duplicates?
Yes
Identify and remove duplicates
Document changes
No
Ensure referential integrity
Execute validation queries
Document changes
Data Cleaning Complete
Data Validation