Anjali Jain
anjali9@illinois.edu
Team - Team232
Team Name - Mocking Bird
Team Members - Single member Project

Final Project Phase-I (Summer 2024)

CS513: Theory & Practice of Data Cleaning

Overview
For my Summer Final Project, the NYPL-menus dataset was assigned to me. This dataset consists of four tables: Dish, Menu, MenuItem, and MenuPages. It provides a detailed historical overview of dining, featuring 17,545 menus from 233 venues across 3,714 locations, with 423,397 unique dishes. Notable trends include frequent appearances of coffee, tea, and celery. Predominantly featuring daily menus and special occasions like anniversaries, the dataset offers insights into culinary trends, pricing patterns, and dining habits over time. Analysis revealed the top occasions, status distribution, popular dishes, and the impressive longevity of celery, which appeared from year 1 to 2928. This comprehensive collection is a valuable resource for food historians and researchers.
About Dataset
NYPL-menus a. Content: "What's on the menu?": A mix of simple bibliographic description of the menus (created by The New York Public Library) and the culinary and economic content of the menus themselves (transcribed by you). b. Source: http://menus.nypl.org/data
Dataset in details

The NYPL-menu dataset offers a fascinating glimpse into the history of dining, capturing detailed information about menus and dishes served at various events and locations over multiple years. As mentioned in overview this dataset includes 17,545 menus from 233 venues across 3,714 locations, featuring an impressive 423,397 unique dishes. On average, each menu showcases around 76 dishes, contributing to a total of 1,332,343 dish appearances. As we delve into the data, notable trends emerge. Coffee, tea, and celery stand out as the most frequently listed items. This rich trove of information includes metadata about the physical characteristics of the menus, the events they were created for, and the establishments that hosted these events. The collection predominantly features daily menus, but also includes special occasions like anniversaries. This dataset have potential to provide provides invaluable insights into culinary trends, pricing patterns, and dining habits over time, making it a robust resource for food historians and researchers in various fields. Using Python in a Jupyter Notebook, a detailed analysis of the Menu and Dish tables was conducted by me. The results revealed a total of 17,545 menus, representing 233 unique venues and 3,714 unique locations, with an average of 75.62 dishes per menu. Among the top occasions, daily menus were the most common, followed by complimentary/testimonial events and anniversaries. Interestingly, the dataset showed a sparse language distribution but a clear status distribution, with 99.01% of the data marked as complete and 0.99% under review. The analysis identified 423,397 unique dishes, which appeared a total of 1,332,343 times on menus. The date range of the menus extends from year 0 to year 2928, highlighting the longevity of certain dishes. Among the top five most popular dishes, coffee topped the list with 8,484 appearances, followed by tea, celery, olives, and radishes. Notably, celery holds the distinction of being the longest-running dish, appearing consistently from year 1 to year 2928. In summary, the NYPL-menu dataset is a treasure trove of historical culinary data, offering deep insights into the evolution of food trends, pricing, and dining practices over an expansive timeframe.
Snapshot of from my Jupyter notebook

Run analysis & plot graph for table DISH -

jupyter Data_Cleaning Last Checkpoint: 5 hours ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help — Trusted — Python 3 (ipykernel)

```
    plt.xlabel('Year')
    plt.ylabel('Number of Dishes')
    plt.show()

# Run the analysis
analyze_dish_data(dish_df)
```

```
Total number of unique dishes: 423397
Total appearances on menus: 1295802
Total times dishes appeared: 1332343

Date range: 0 to 2928

Top 5 most popular dishes:
         name  times_appeared
88     Coffee            8484
89        Tea            4769
13     Celery            4690
978    Olives            4553
5    Radishes            3346

Average dish price: $1.28
Most expensive dish: Cream cheese with bar-le-duc jelly ($3050.00)

Longest running dish: Celery
  First appeared: 1
  Last appeared: 2928
```

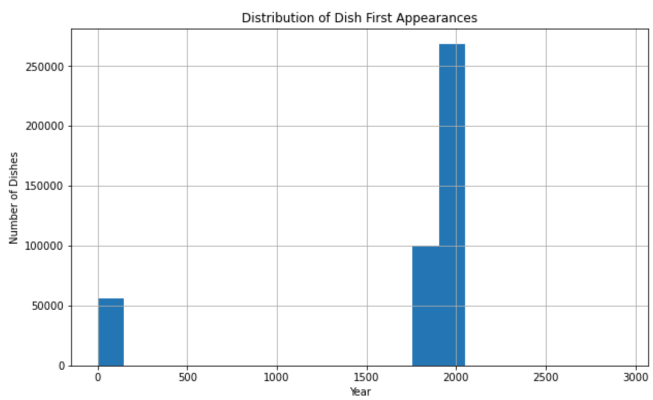


Table Fields & description

After performing initial analysis on the four CSV files—Menu, MenuPage, MenuItem, and Dish—I created four tables using Microsoft Excel. Here are the details of each table and its attributes.

	Table Attributes	
Table Name	Field Name	Field Description
Dish	ID	Unique identifier for each dish
	Name	Name of the dish
	Description	Description of the dish.
	Menus_appeared	Number of menus on which dish has appeared
	times_appeared	Total number of times the dish has appeared on all menus.
	first_appeared	Year when the dish first appeared on a menu.
	last_appeared	Year when the dish last appeared on a menu.
	lowest_price	Lowest recorded price of the dish
	highest_price	Highest recorded price of the dish

	Table Attributes		
Table Name	Field Name	Field Description	
Menu	ID	Unique identifier for each menu	
	name	Name of the event or establishment	
	Sponsor	Sponsor of the menu	
	Event	Type of event for example is it breakfast or Dinner ?	
	Venue	Venue of the event	
	Place	Place (city ,state)where the event took place .	
	physical_description	Physical description of the menu (e.g., card, booklet).	
	Occasion	Special occasion associated with the menu.	
	Notes	Additional notes about the menu.	
	call_number	Call number for the menu.	
	Keywords	Keywords associated with the menu.	
	Language	Language of the menu.	
	Date	Date of the event.	
	Location	Specific location of the venue.	

	Table Attributes	
		the venue.
	location_type	Type of location (e.g., hotel, restaurant).
	Currency	Currency used in the menu.
	Currency_symbol	Symbol of the currency.(A currency symbol or currency sign is a graphic symbol used to denote a currency unit.)
	Status	Status of the menu (e.g., complete, under review).
	Page_count	Number of pages in the menu.
	Dish_count	Number of dishes in the menu.

	Table Attributes		
Table	Field Name	Field Description	
MenuItem	ID	Unique identifier for each menu item	
	Menu_page_id	Foreign key referencing the menu page.	
	Price	Price of the dish.	
	high_price	High price of the dish.	
	dish_id	Foreign key referencing the dish.	
	created_at	Timestamp when the price was recorded.	
	updated_at	Timestamp when the price was last updated.	
	xpos	X-coordinate position of the price on the menu page.	
	ypos	Y-coordinate position of the price on the menu page.	

	Table Attributes		
Table	Field Name	Field Description	
MenuPage	ID	Unique identifier for each menu page	
	menu_id	Foreign key referencing the menu.	
	page_number	Page number in the menu	
	image_id	Image ID of the scanned menu page.	
	full_height	Full height of the scanned image.	
	full_width	Full width of the scanned image.	
	uuid	Unique identifier for each image	

Database Schema SQL

Below schema defines the structure of the dataset and outlines the relationships between the tables. The Dish table captures information about individual dishes, the Menu table captures information about specific menus and events, the MenuPage table captures metadata about the scanned pages of each menu, and the MenuItem table captures the pricing information for each dish on each menu page.

Each table has primary key ID,Table MenuPage has foreign key menu_id and MenuItem has 2 foreign key dish_id and menu_page_id.

```
Dish -

CREATE TABLE Dish (
id INT PRIMARY KEY,
name VARCHAR(255),
description TEXT,
menus_appeared INT,
times_appeared INT,
first_appeared YEAR,
last_appeared YEAR,
lowest_price DECIMAL(5,2),
highest_price DECIMAL(5,2)
);

Menu -

CREATE TABLE Menu (
id INT PRIMARY KEY,
name VARCHAR(255),
sponsor VARCHAR(255),
event VARCHAR(255),
venue VARCHAR(255),
place VARCHAR(255),
physical_description TEXT,
occasion VARCHAR(255),
notes TEXT,
call_number VARCHAR(255),
keywords TEXT,
language VARCHAR(255),
date DATE,
location VARCHAR(255),
location_type VARCHAR(255),
currency VARCHAR(50),
currency_symbol VARCHAR(5),
 status VARCHAR(50),
page_count INT,
dish_count INT
 );
```

Menu Page -

```
CREATE TABLE MenuPage (
id INT PRIMARY KEY,
menu_id INT,
page_number INT,
image_id INT,
full_height INT,
full_width INT,
uuid VARCHAR(255),
FOREIGN KEY (menu_id) REFERENCES Menu(id)
);
```

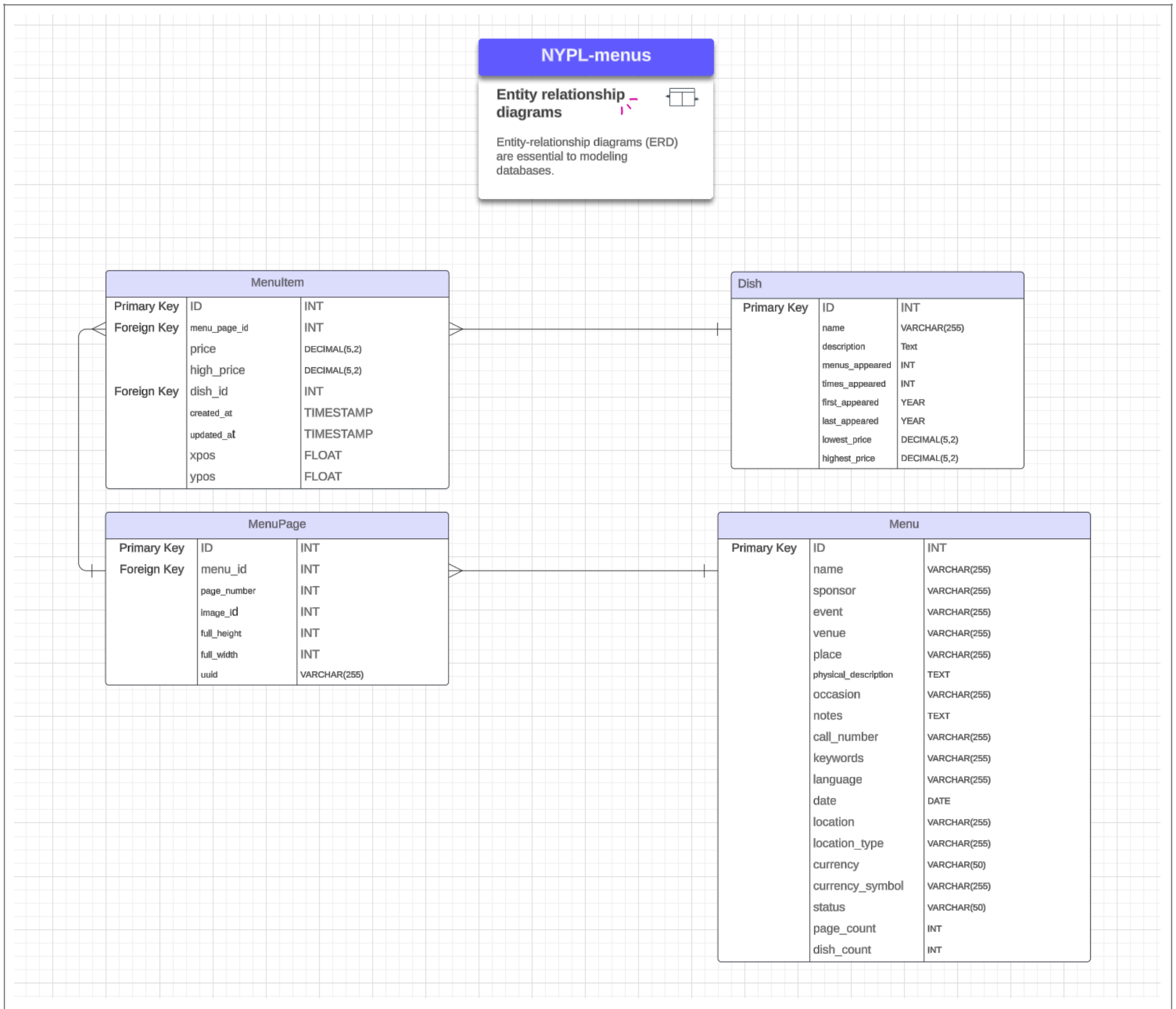
MenuItem -

```
CREATE TABLE MenuItem (
id INT PRIMARY KEY,
menu_page_id INT,
price DECIMAL(5,2),
high_price DECIMAL(5,2),
dish_id INT,
created_at TIMESTAMP,
updated_at TIMESTAMP,
xpos FLOAT,
ypos FLOAT,
FOREIGN KEY (menu_page_id) REFERENCES MenuPage(id),
FOREIGN KEY (dish_id) REFERENCES Dish(id)
 );
```

ER diagram
Below ER diagram is created from the above schemas illustrates the entities involved in a NYPL-menus, their attributes, and the relationships that exist between them.
ER Overview :

Menu->MenuPage(1,Many)
MenuPage-MenuItem(1,Many)
Dish -> MenuItem (1,Many)

Below is the ERD diagram created using LucidChart online tool.



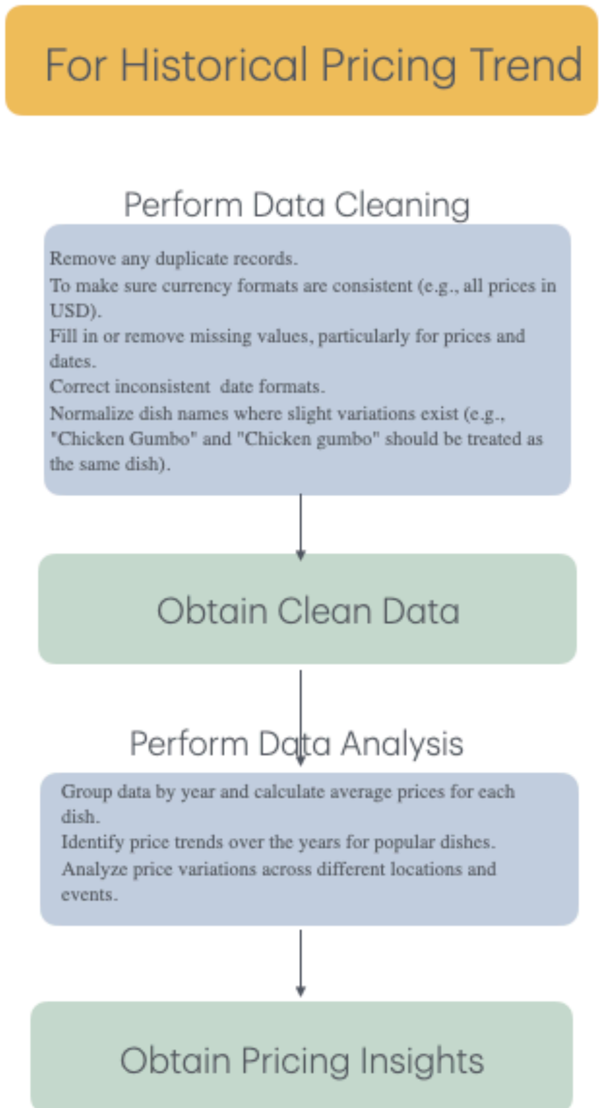
Develop three use cases.

All three cases are visually represented in a flowchart created using Keynote.

Target (Main) use case U1: data cleaning is necessary and sufficient

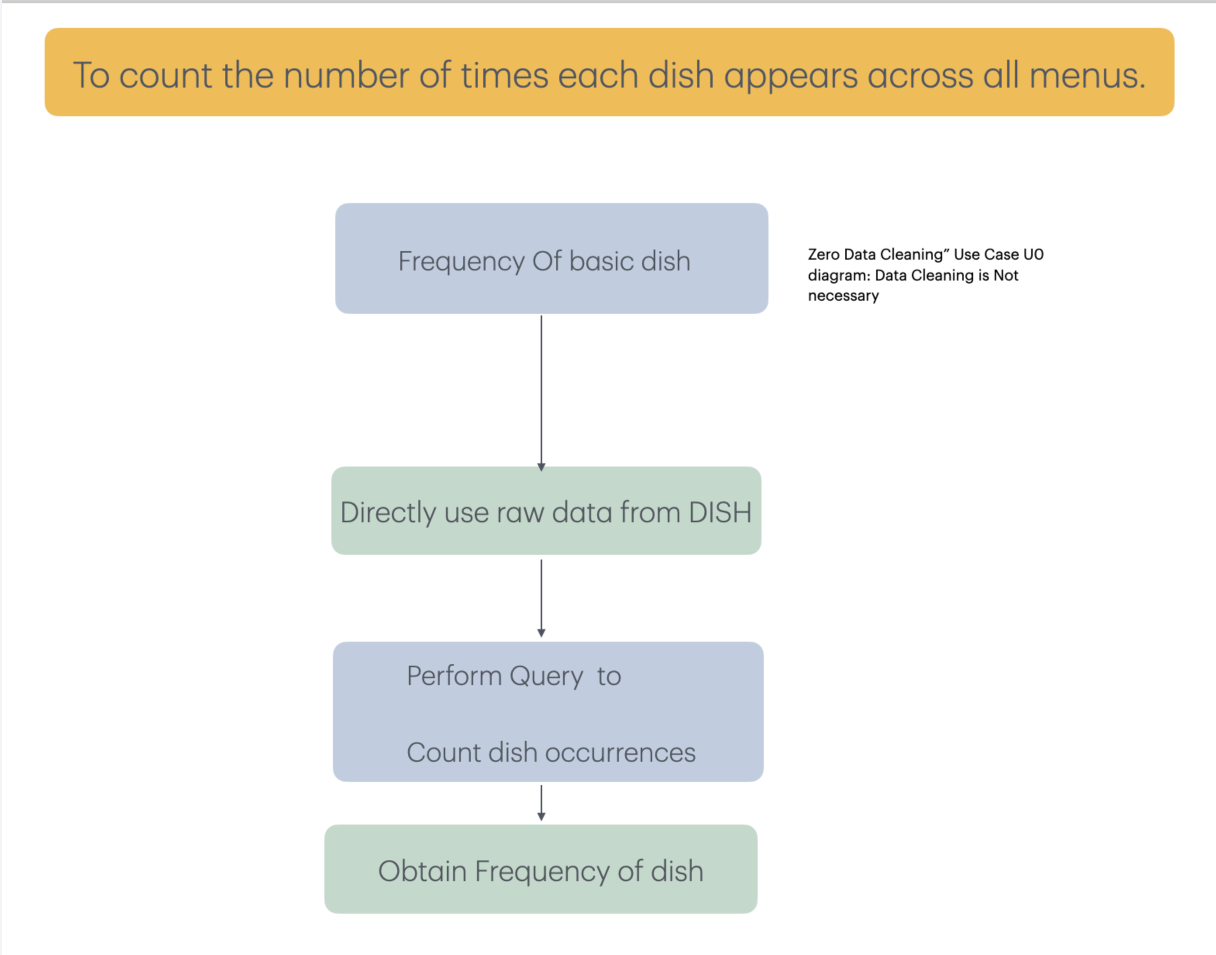
Use Case U1 - Analyze historical pricing trends of dishes across various menus over time

In order to perform analysis on historical pricing trends of dishes across various menus over time, a detailed and cleaned dataset is must which includes accurate dates, consistent currency formats, and complete price information.

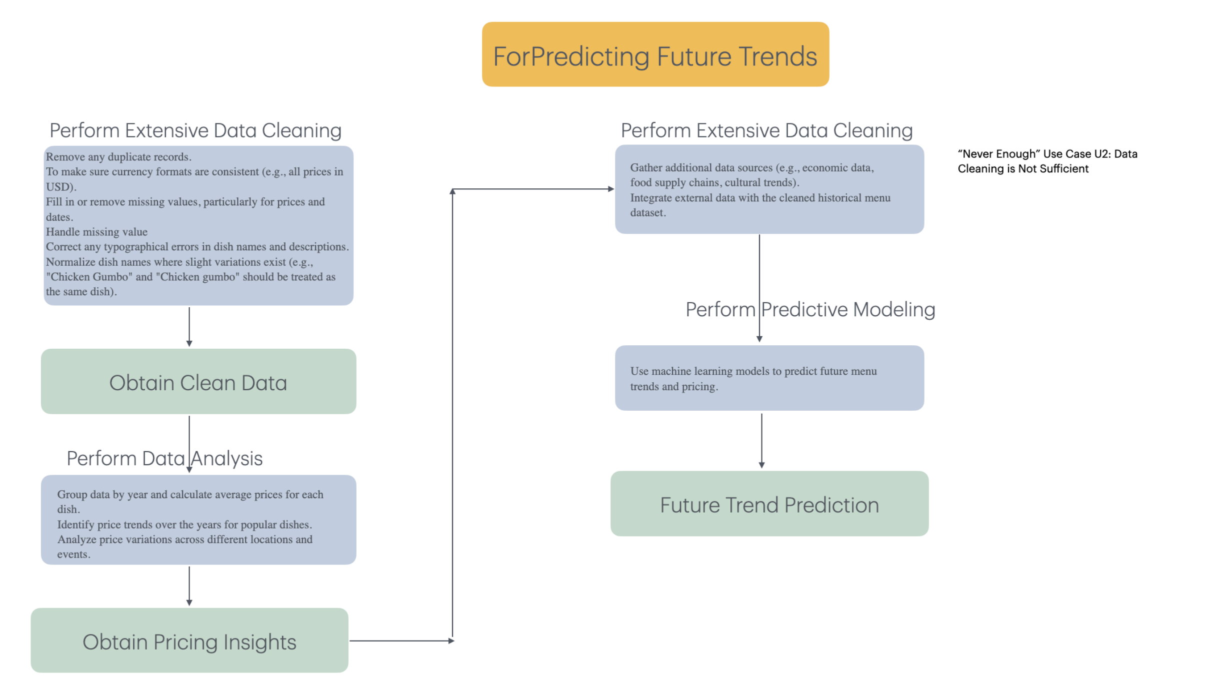


Target (Main) use case U1 Diagram: data cleaning is necessary and sufficient

Result - Cleaned data will allow intended user for accurate and meaningful analysis of historical pricing trends, providing insights into how food prices have evolved over time.
“Zero data cleaning” use case U0: data cleaning is not necessary
Use Case U0 - To count the number of times each dish appears across all menus using the raw data
In order to count the number of times each dish appears across all menus. This use case does not require any data cleaning since the raw data is sufficient for counting occurrences.



Result - The raw data of DISH is good enough to provide a count of dish occurrences without any cleaning.
"Never enough" use case U2 : data cleaning is not sufficient
Use Case U2 - Predicting Future Menu Trends and Pricing
In order to achieve prediction of future trends in menu offerings and pricing based on historical data, more comprehensive data and additional external factors are required. This use case involves complex modeling and external data sources such as economic indicators, food supply data, and cultural trends.



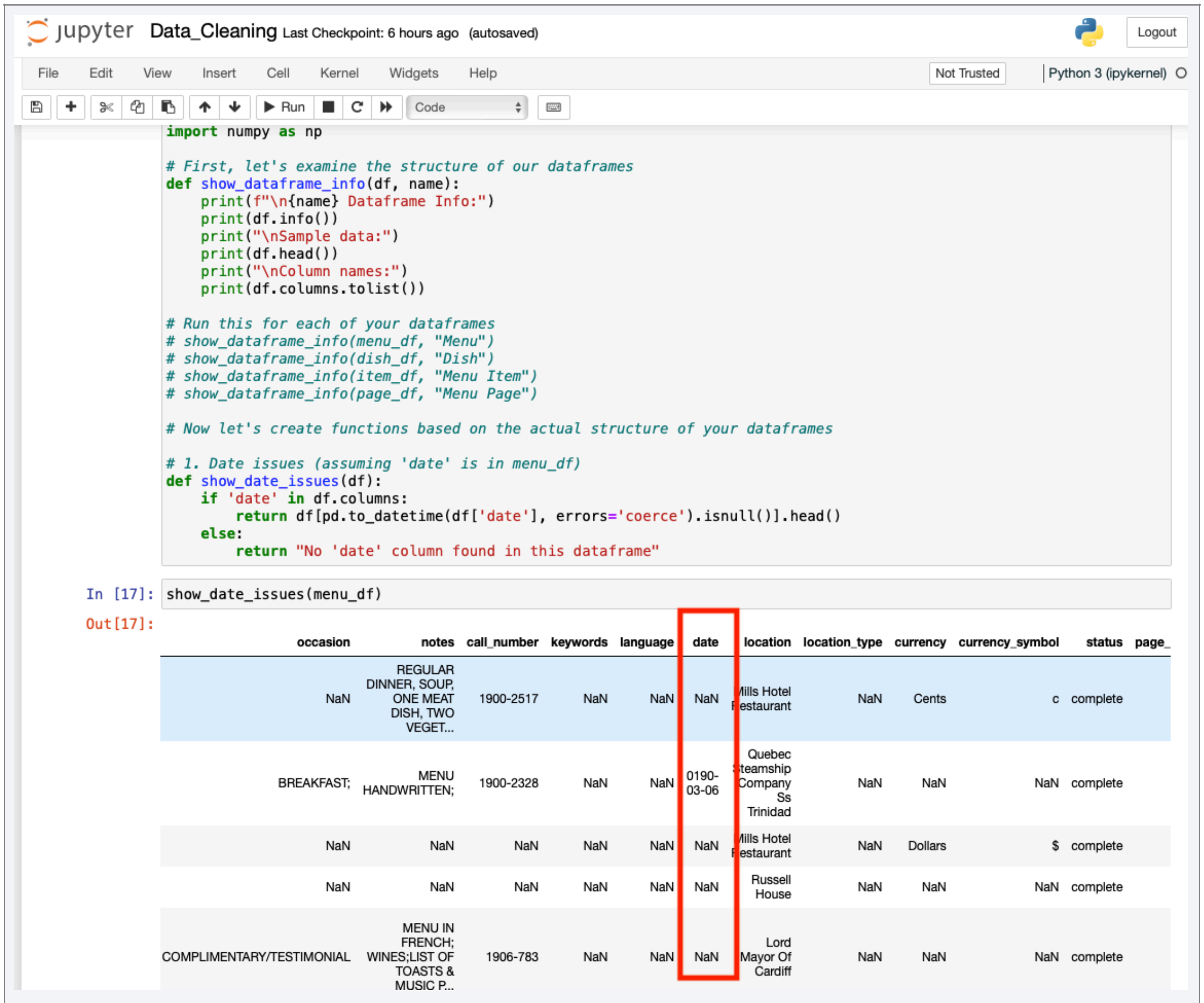
Result - Even after extensive data cleaning, the original dataset is not sufficient alone. The integration of external data sources and advanced modeling techniques are required to accurately predict future trends.

List obvious data quality problems

In order to perform analysis on historical pricing trends of dishes across various menus over time, a detailed and cleaned dataset is must which includes accurate dates, consistent currency formats, and complete price information however during inspection of existing dataset several data quality issues have been identified that must be addressed for successful and correct results.

Below is a list of data quality issues identified during my analysis of the data files using Python in Jupyter Notebook. Please find below the issues and screenshots from my code explains reason for Data cleaning :

1. For consistency - Inconsistent data formats (dates, currency, place address) make it difficult to perform aggregations and comparisons.Therefore it is must to standardizing formats ensures uniformity, allowing for accurate temporal and spatial analysis.




```python
import numpy as np

# First, let's examine the structure of our dataframes
def show_dataframe_info(df, name):
    print(f"\n{name} Dataframe Info:")
    print(df.info())
    print("\nSample data:")
    print(df.head())
    print("\nColumn names:")
    print(df.columns.tolist())

# Run this for each of your dataframes
# show_dataframe_info(menu_df, "Menu")
# show_dataframe_info(dish_df, "Dish")
# show_dataframe_info(item_df, "Menu Item")
# show_dataframe_info(page_df, "Menu Page")

# Now let's create functions based on the actual structure of your dataframes

# 1. Date issues (assuming 'date' is in menu_df)
def show_date_issues(df):
    if 'date' in df.columns:
        return df[pd.to_datetime(df['date'], errors='coerce').isnull()].head()
    else:
        return "No 'date' column found in this dataframe"
```

In [17]:
```python
show_date_issues(menu_df)
```

Out[17]:

occasion	notes	call_number	keywords	language	date	location	location_type	currency	currency_symbol	status	page_
NaN	REGULAR DINNER, SOUP, ONE MEAT DISH, TWO VEGET...	1900-2517	NaN	NaN	NaN	Mills Hotel Restaurant	NaN	Cents	c	complete	
BREAKFAST;	MENU HANDWRITTEN;	1900-2328	NaN	NaN	0190-03-06	Quebec Steamship Company Ss Trinidad	NaN	NaN	NaN	complete	
NaN	NaN	NaN	NaN	NaN	NaN	Mills Hotel Restaurant	NaN	Dollars	$	complete	
NaN	NaN	NaN	NaN	NaN	NaN	Russell House	NaN	NaN	NaN	complete	
COMPLIMENTARY/TESTIMONIAL	MENU IN FRENCH; WINES;LIST OF TOASTS & MUSIC P...	1906-783	NaN	NaN	NaN	Lord Mayor Of Cardiff	NaN	NaN	NaN	complete	

2. As seen above field values are missing "NAN", inconsistent data and typographical errors can lead to incorrect calculations and trends. Therefore filling missing values, removing duplicates, and correcting errors ensures that the analysis is based on accurate data." and write for this point as reason of data cleaning Different currencies for a same dish.

3. To standardize currency values across datasets, particularly when different currencies are used to price the same dish. Without normalization, analysis may be skewed, leading to inaccurate comparisons and trends. By converting all currency values to a single standard, such as USD or EUR, data integrity is maintained, ensuring consistent and reliable analysis."

Different currencies for a same dish

In [19]:

```
def show_currency_issues_for_same_dish(menu_df, dish_df, menu_item_df, menu_page_df):
    # First, let's merge the necessary dataframes
    merged_df = pd.merge(menu_item_df, menu_page_df[['id', 'menu_id']], left_on='menu_page_id', right_on='id', suffi
    merged_df = pd.merge(merged_df, menu_df[['id', 'currency', 'currency_symbol']], left_on='menu_id', right_on='id'
    merged_df = pd.merge(merged_df, dish_df[['id', 'name']], left_on='dish_id', right_on='id', suffixes=('', '_dish'

    # Group by dish name and count unique currencies
    dish_currency_counts = merged_df.groupby('name')['currency'].nunique().reset_index()

    # Filter dishes that appear with multiple currencies
    multi_currency_dishes = dish_currency_counts[dish_currency_counts['currency'] > 1]

    # Get examples for these dishes
    examples = []
    for dish in multi_currency_dishes['name'].head():
        dish_examples = merged_df[merged_df['name'] == dish][['name', 'currency', 'currency_symbol', 'price', 'high_
        examples.append(dish_examples)

    # Combine all examples
    result = pd.concat(examples).reset_index(drop=True)

    return result

# Usage
currency_issues = show_currency_issues_for_same_dish(menu_df, dish_df, item_df, page_df)
```

In [20]:

```
currency_issues
```

Out[20]:

	name	currency	currency_symbol	price	high_price
0	" American Rye	Belgian Francs	BEF	0.50	7.5
1	" American Rye	Francs	FF	0.50	7.5
2	"Red Brand" sirloin steak	Canadian Dollars	C$	1.25	NaN
3	"Red Brand" sirloin steak	Dollars	$	1.25	NaN
4	"Red Brand" small sirloin steak	Canadian Dollars	C$	1.00	NaN
5	"Red Brand" small sirloin steak	Dollars	$	1.00	NaN
6	"Salad" Lettuce & Tomatoes	Belgian Francs	BEF	NaN	NaN
7	"Salad" Lettuce & Tomatoes	Francs	FF	NaN	NaN
8	(1) Lambchops a la Murillo	Cents	c	0.00	NaN
9	(1) Lambchops a la Murillo	Dollars	$	0.40	NaN

4. Inconsistent dish names across datasets can create ambiguity and hinder accurate analysis. Data cleaning involves standardizing dish names by resolving spelling variations, abbreviations, and synonyms. This ensures that all references to the same dish are uniform, enabling precise aggregation and analysis of culinary trends and preferences.

I took below example of chicken salad.

```
9                                  8, 62, 18, 2, 122, 3, 1
```

Out[22]:

	name	id	menus_appeared
201	Chicken salad	217	1809
11018	Chicken Salad.	13723	8
26298	Chicken [salad]	33341	9
26699	Chicken Salad,	33844	1
31238	Chicken (SALAD)	40006	6
155858	Chicken Salad 3 25	195543	1
286578	Chicken Salad 1.75	361195	1
298440	Chicken Salad	375982	408
298656	Chicken [Salad]	376218	2
298915	chicken salad	376510	26
302318	Chicken Salad	380239	13
305282	Chicken (salad)	383517	8
317433	Chicken salad	397020	1
320011	Chicken [Salad]	399860	1
354394	CHICKEN SALAD	439070	8
369827	chicken salad	455981	1
378079	chicken Salad	464458	1
397279	Chicken, Salad	486569	1
409519	Chicken [Salad] (1)	500251	1

5.Identifying and correcting multiple outlier details is crucial in data cleaning to ensure statistical robustness and accuracy in analysis. Outliers, when left unaddressed, can skew results and misrepresent trends. Cleaning involves methods such as removing or adjusting outliers based on statistical measures like z-scores or interquartile range, thereby improving the reliability of insights derived from the data.

```
In [24]: multiple_outlier_details
Out[24]:
```

	Item ID	Dish Name	Price	Currency	Outlier Type	Menu ID	Menu Date	Menu Venue	Page Number
0	26464	Dom Perignon	180000.0	Italian Lire	High	26464	NaN	HOTEL	5.0
1	26464	Cristal	180000.0	Italian Lire	High	26464	NaN	HOTEL	5.0
2	26464	Krug	160000.0	Italian Lire	High	26464	NaN	HOTEL	5.0
3	26464	Veuve Clicquot Ponsardin	110000.0	Italian Lire	High	26464	NaN	HOTEL	5.0
4	26464	Moet et Chandon	100000.0	Italian Lire	High	26464	NaN	HOTEL	5.0

Some additional reason are -

1. For accurate outcomes - As seen above in all tables(Menu,Dish,MenuPage and MenuItem) of dataset values are missing, inconsistent data and typographical errors can lead to incorrect calculations and trends. Therefore filling missing values, removing duplicates, and correcting errors ensures that the analysis is based on accurate data.

2. Reliability - Inconsistent or incorrect data can lead to unreliable insights, affecting decision-making. Therefore Data cleaning improves the reliability of the dataset, making it fit for generating trustworthy insights.

3. Completeness - Missing descriptions or partial records can result in incomplete analysis Therefore one must Ensure all necessary data is present and complete allows for comprehensive analysis.

In conclusion, data cleaning transforms the raw dataset into a cleaned version that is accurate, consistent, reliable, and complete. This cleaned dataset is crucial for performing meaningful and accurate historical pricing trends analysis, as described in the main use case U1.

Devise an initial plan

My plan for Phase 2 is to ensure early submission to avoid any last minute hassle.

Step Number	Activity	Responsibility	Timeline
S1	Description of dataset D and matching use case U1	Anjali	07/10/2024
S2	Profiling of D to identify the quality problems P that need to be addressed to support U1;	Anjali	07/13/2024

S3	Performing the data cleaning process using one or more tools to address the problems P (here you should describe which tools you are planning to use, e.g., OpenRefine; Python; etc.) S	Anjali	07/20/2024
S4	Checking that your new dataset D′ is an improved version of D, e.g., by documenting that certain problems P are now absent and that U1 is now supported	Anjali	07/21/2024
S5	Documenting the types and amount of changes that have been executed on D to obtain D′ .	Anjali	07/24/2024