ADS 507 – Final Project – Team 2

Design Document

GitHub: [Pii-USD/ADS507 (github.com)](https://github.com/Pii-USD/ADS507)

Marvin Moran

Muris Saab

Ravita Kartawinata

# Source Datasets

The datasets being used are originated from the following sources. They we selected because they allowed us to implement the concepts and techniques we learned during the course.

|  |  |
| --- | --- |
| Source | Datasets |
| Drinks API <https://www.thecocktaildb.com/>  An open, crowd-sourced database of drinks and cocktails from around the world. | Glass and  Drink tables |
| Comma Separated Values (CSV) files  These CSV files were generated to represent the concept of multiple sources. | Category and  Ingredients |

# Goals

The goal is to ingest the data for alcoholic and non-alcoholic beverages via an Extract, Transform, and Load (ETL) approach that will generate a final SQL dataset. The final dataset will then be used to support a searchable catalog for users.

# Pipeline diagram

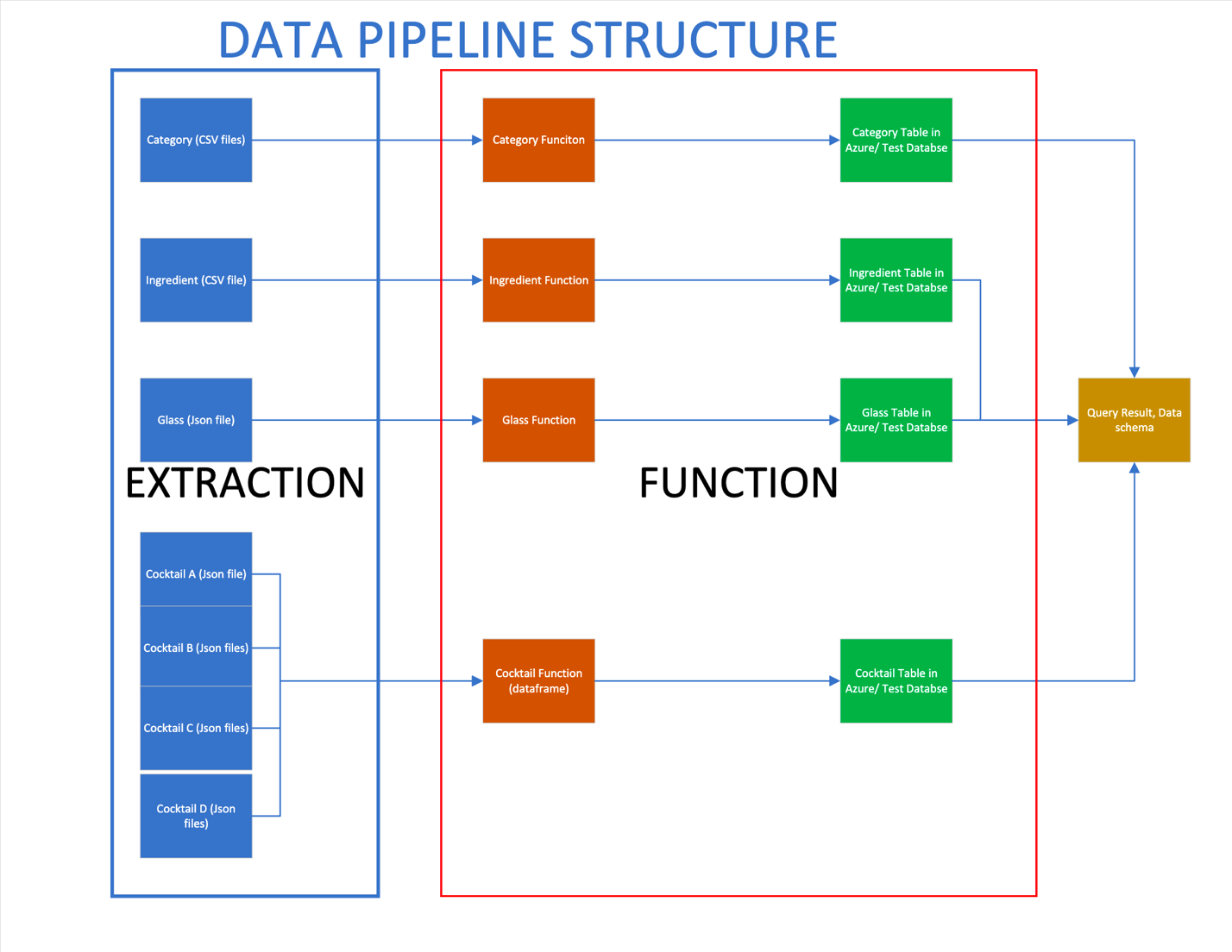


Figure 1. Data Architecture diagram

Figure 1 illustrates the process of extracting, transforming, and loading data. This diagram depicts four primary data sources. These data sources originate from cocktaildb.com, which offers five distinct datasets; we incorporated four of these datasets into our project. The initial dataset, Category, is presented in CSV format and comprises two attributes: strCategory and strDescription.

A picture containing calendar

Description automatically generated

Figure 2. Category data extraction

The second dataset, comprised of ingredients, is structured as a CSV file and includes two attributes: strIngredient1 and strCountryofOrigin

The third and fourth datasets are formatted as JSON files. The third dataset, referred to as the glass table, comprises one attribute .The fourth dataset, known as the drink table, contains 51 attributes and is characterized by semi-structured data as can be seen in Figure 3.

Graphical user interface, text, application, email

Description automatically generated

Figure 3. Drink JSON file

The extraction of the Drink JSON file was accomplished by grouping entries based on the first letter of the drink name. Using Python code, this extraction process was automated through a loop to systematically handle each complete first letter of the drink name.

After extracting these datasets, they were converted into data frames for review. The Category, Ingredients, and Glass datasets contain structured data and can be loaded in batches. However, the Drink dataset contains semi-structured data, with some attributes containing images or file texts.

No transformations were conducted on the structured data. However, since the Drink dataset has 51 attributes, many of which are unnecessary for the project's goals, dropping these attributes will expedite the loading process. Additionally, a simple transformation was applied to the Drink dataset by adding an additional column to identify whether it's an adult drink or not.

Text

Description automatically generated

Figure 4. Drink Data clean-up

Once the clean-up was completed, the data were loaded into our Azure Test Database. Before loading the data, a series of syntaxes to create tables were executed.

# Steps to deploy pipeline

Running Data pipeline in your local instance

1. Ensure you have a local instance of MySqlWorkbench running
2. Create a test Database in your localhost server

Running Data pipeline in Azure Database

1. Ensure parameter to Azure Database is enabled
2. Disable the local parameter

Graphical user interface, text, application, email

Description automatically generated

1. Download drinks.ipynb and the 2 CSV files to the same folder
2. Open the drinks.ipynb notebook file
3. Update the connection parameters in the first code block to your local instance of MySqlWorkbench
4. Run all code blocks in sequence
   1. First, the code will ensure that the necessary libraries and packages are downloaded to create a connection to the Azure data store. Also, the require parameters for creating a connection to the data store are set within the first code cell.
   2. The subsequent code cell then creates the connection engine utilizing the sqlalchemy package.
   3. Next, the following code cells connect to the Drinks API noted above and extract data in a JSON format.
   4. Similarly, the following code cells thereafter ingests the 2 target CSV files.

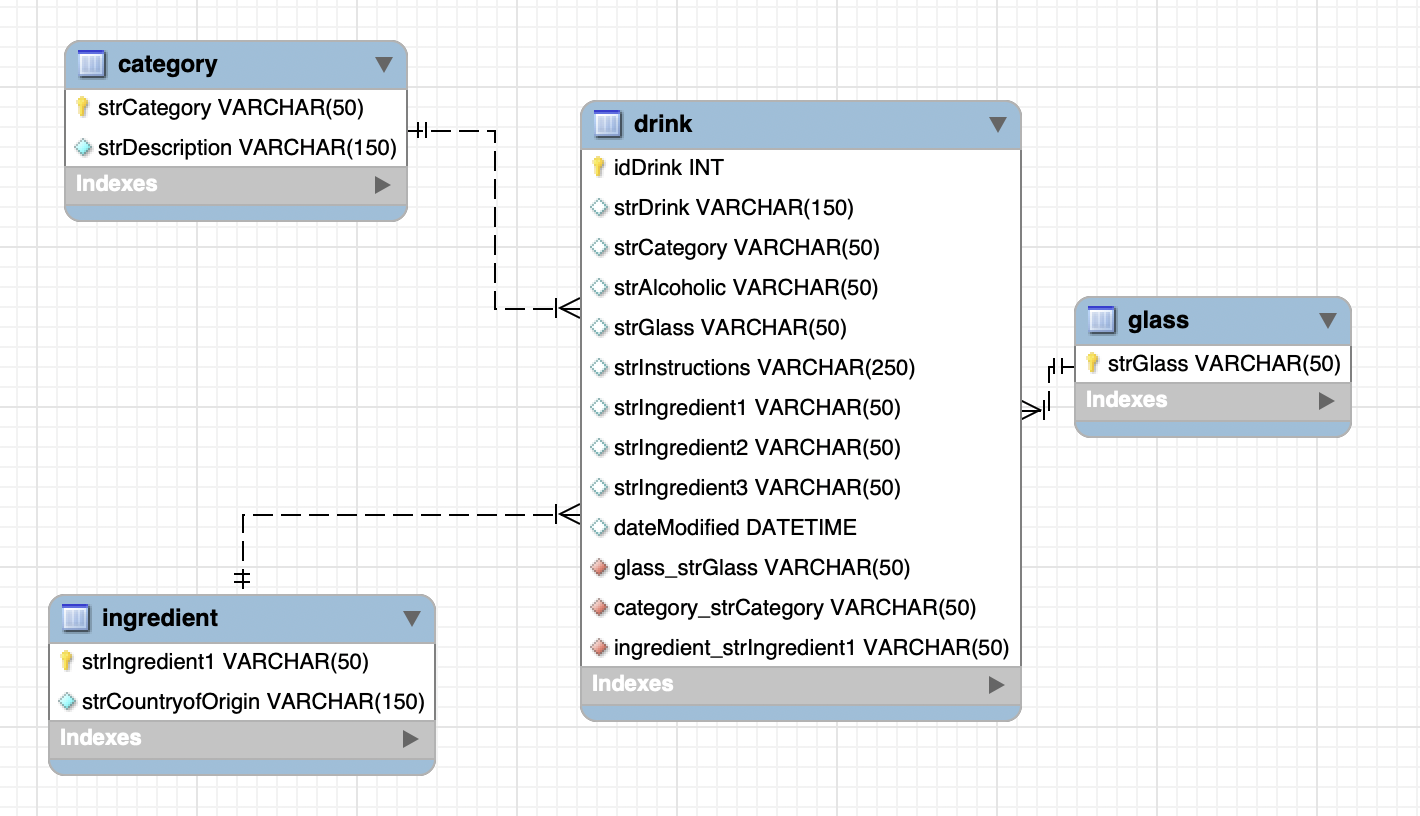
Following these code executions, the next code cells allow user to:

1. trigger the input process in which they submit a value (e.g., ingredient = 'vodka').
2. trigger the query execution and results process.

# Schema

Supporting data is present in this repository, both as CSV files and a JSON file. The 'category.csv' file and 'ingredients' file are the 2 target CSV files while the 'drinkDB\_schema.mwb' is the target JSON file.

Upon the tables’ creation, a reverse engineering process was conducted in SQL Workbench to generate the final data schema, as illustrated in below Figure 9. The schema depicts one-to-many relationships between category and drink, one-to-many relationships between ingredient and drink, and one-to-many relationships between glass and drink. Each of these relationships with the Drink table is connected to the primary keys of Category (strCategory), Glass (strGlass), and Ingredient (strIngredient1).



# Output validation

The pipeline output is validated by a sample code running Step 4 in the Jupiter notebook:

In this example, if we select ‘beer’ glass, the output table can be queried for a simple EDA analysis of the category:

A graph of blue rectangular bars

Description automatically generated with medium confidence

# Considerations and known gaps

This pipeline was designed for educational purposes. We aimed to enable any student running a local instance of MySqlWorkbench to reproduce the same steps and learn about the process.

The dataset is both small and primarily static, which means that the system resources required to process it are minimum. The frequency of updates to the upstream data sources is very low so no automated ingestion mechanisms were considered. No growth is expected for this dataset while it is used as a proof of concept.

From a security perspective, the use of a local instance of MySqlWorkbench also assume the use for educational purposes, otherwise the use of hard coded credentials in code is not be recommended.