Michał Soszko

To answer the questions included in the task, I created a project in Bigquery and imported the data (instructions in the README file of the GitHub repository).

dbt project

I treated the imported dataset as a source dataset and configured a new dbt project. The dbt project is available in the GitHub repository. Most of the data transformations (cleaning, aggregation, column naming adjustment, etc.) were carried out in SQL as part of a classic data pipeline.

Answers

Ad. 1. To answer the question, I created the mart_fact_invoices table (schema 3_marts) which is a cleaned version of the source fact table. Based on it, I performed aggregation and ranking, summarized in the 4_reports.repor_top_10_retailers_aggy table.

Ad. 2a During the analysis of the integrity of the source data, I divided the data into fact and dimension tables. It quickly turned out that in the dimension table, which collected information about stores, the data were not consistent.

- Address data such as - street name, county, or geolocation data could differ from each other (slightly) or sometimes they were missing (nulls). There were situations where one store_number (which should be a primary key) corresponded to several different sets of dimensions. In this case, the store_number lost its uniqueness. Here I decided to extract the most frequently repeating sets of parameters and (assuming that they do not change in the studied period) and assign them per store_number (separately for GEO data and address information).
- Store names (store_name), against which individual outlets should be grouped, differed from each other. In some cases (like for Hy-Vee) it was easy to locate a pattern that allowed to extract the proper name (or part of it) for the retailer. Unfortunately, in many cases simple regexp expressions were not sufficient and it was difficult for me to develop a grouping script in a reasonable time. Here I decided to refer to the NLP model (GPT-3.5), using a primitive script written in Python (folder analyses/match_store_names). The script deduced retailer groups and generated a dictionary assigning store names to their retailer names. Then I imported the dictionary into dbt as a csv (the file itself required some processing, which was done manually).
- Raw Fact data has incomplete invoices - for a very small fraction of data, the number of records per invoice_id was lower than the maximum purchase index on the invoice. This suggests that in their case, sales amounts will be underestimated.
- The sales calculation in a fairly large number of invoices was incorrect ((state_bottle_retail * bottles_sold) != sales__dollars) and required correction.

Ad. 2b Reporting bugs to the data author, hoping he will listen :) But seriously - in my opinion there is no universal way to solve data quality issues. It is necessary to approach individually what type of data it is, where it comes from and consider whether e.g. problems are not the result of a disturbed data ingestion process. If not - missing, nullified values can be tried to be supplemented using external data (I attempted to do that here as well, but none of the supplementary tables from IOWA data was sufficient) or based on existing records and their frequency of occurrence.