CSC343:

Term Project

Phase 2

November 19th, 2021

Prerak Chaudhari | 1005114760

Jaakulan Subeethakumar | 1005225757

Design Decisions

In phase 1, one of our proposed changes to the schema from what was provided in our dataset’s data directory was to further split the cars-data csv file into 3 separate files: one for miles per gallon, one for horsepower and one for the remaining metrics. This is because some rows are missing a value for either miles per gallon or horsepower and we wished to enforce attribute requirements for all columns in our tables.

We received feedback from the TA that this change was unnecessary because there is a 1-to-1 relationship between car ID and these parameters. Therefore, we will be keeping the csv file as is, removing the MilesPerGallon and Horsepower relations, and adding nullable columns for mpg and horsepower to our Parameters table. Given that both mpg and horsepower are never both null for any row in this csv file, and our queries never involve both mpg and horsepower at the same time, we can simply add a where clause to our queries that filters out tuples with a null in the interested column.

Aside from this, we made some further changes to the schema on our own accord. Firstly, for the Parameters table, we updated the column type of Displacement to be FLOAT because we found a row with a float value for this column. Secondly, to better represent the dataset and domain, we also updated the CountryName and ContinentName columns of the respective Countries and Continents relations to be keys.

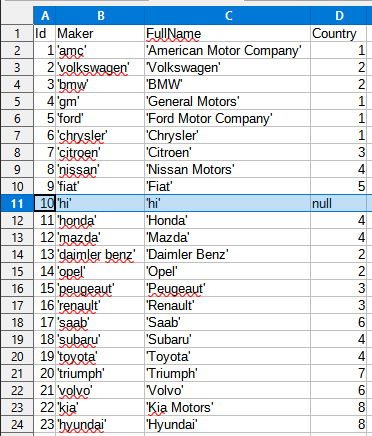
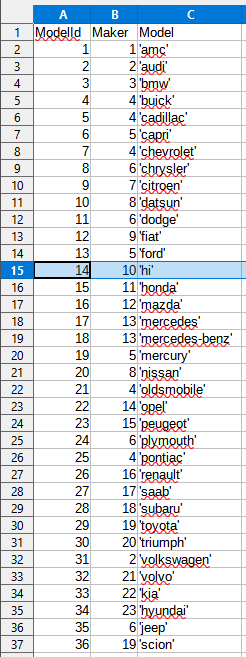
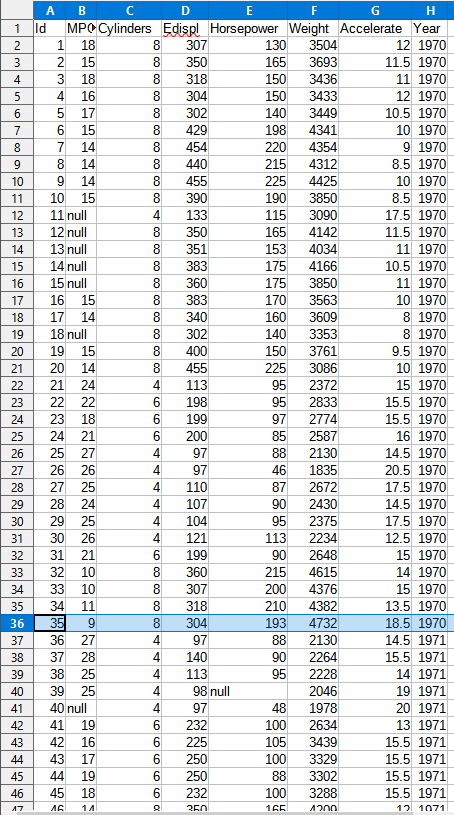
Our schema in phase 1 was described using relational algebra. Some of our relations have multiple keys, which isn’t allowed in SQL. Therefore, for each relation, we opted to use the integer ID key as the primary key of its respective SQL table, with any other keys being unique columns.

We are still replacing the Model column in the cars-names csv file with its numerical identifier from the model-list csv file. This is done to reduce redundancy as the model name should not be present in multiple relations. While this may come at a minor performance cost due to additional natural joins, we believe it is good practice to design the schema this way. For example, our first query deals with car manufacturers as we need to determine their country (thus, continent) of origin. This schema design means we don’t need to re-lookup the model name; furthermore, our comparison and lookup times are faster because we are using an integer key versus a string key (integers are smaller than strings).

Now, we believe the shown schema in our demo.txt file is good because not only is it easy to understand, but it eliminates redundancies in each table. Furthermore, each table has an integer primary key and its foreign keys are also integers. This speeds up comparisons and lookups, compared to using a string key which is present in some relations, because integers are smaller than strings. Finally, the check constraints in our Parameter table prevent one from inserting nonsensical car metrics (i.e. negative values for miles per gallon, acceleration, weight, etc).

Cleaning Process

Detailed below are the steps taken to clean our dataset. Pictures are included with every step for added clarity. While this causes the report to go past 2 pages, we believe that these visual cues provide greatly beneficial added context that would help one reproduce these steps with ease.

1. The nonsensical manufacturer ‘hi’ whose country of origin is null was removed from the car-makers csv file to enforce attribute requirements:  
   
2. The corresponding model was removed from the model-list csv file to avoid foreign key violations:  
   
3. There is 1 car of model ‘hi’ in the car-names csv file that had to be deleted to avoid foreign key violations:  
   
4. This car’s metrics also had to be deleted from the cars-data csv file to avoid foreign key violations:  
   
5. In the car-names csv file, there are 2 instances of erroneous whitespace padding that were corrected by removing the whitespace before the opening quotation mark:  
     
   
6. The Model column in the cars-names csv file is replaced with its numerical identifier from the model-list csv file.  
     
   This change was accomplished using the pandas library:

| import pandas as pd  carNames = pd.read\_csv('car-names.csv') modelList = pd.read\_csv('model-list.csv')  combined = pd.merge(carNames, modelList, on='Model') df = combined[['Id', 'ModelId', 'Make']].sort\_values(by=['Id'])  df.to\_csv('car-names.csv', index=False) |
| --- |

1. All instances of the word “null” in the cars-data csv file are replaced by an unquoted empty string as per standard CSV format for handling null values:  
   