Vehicle Fuel Consumption Ratings

D532 Final Project (Part 2)

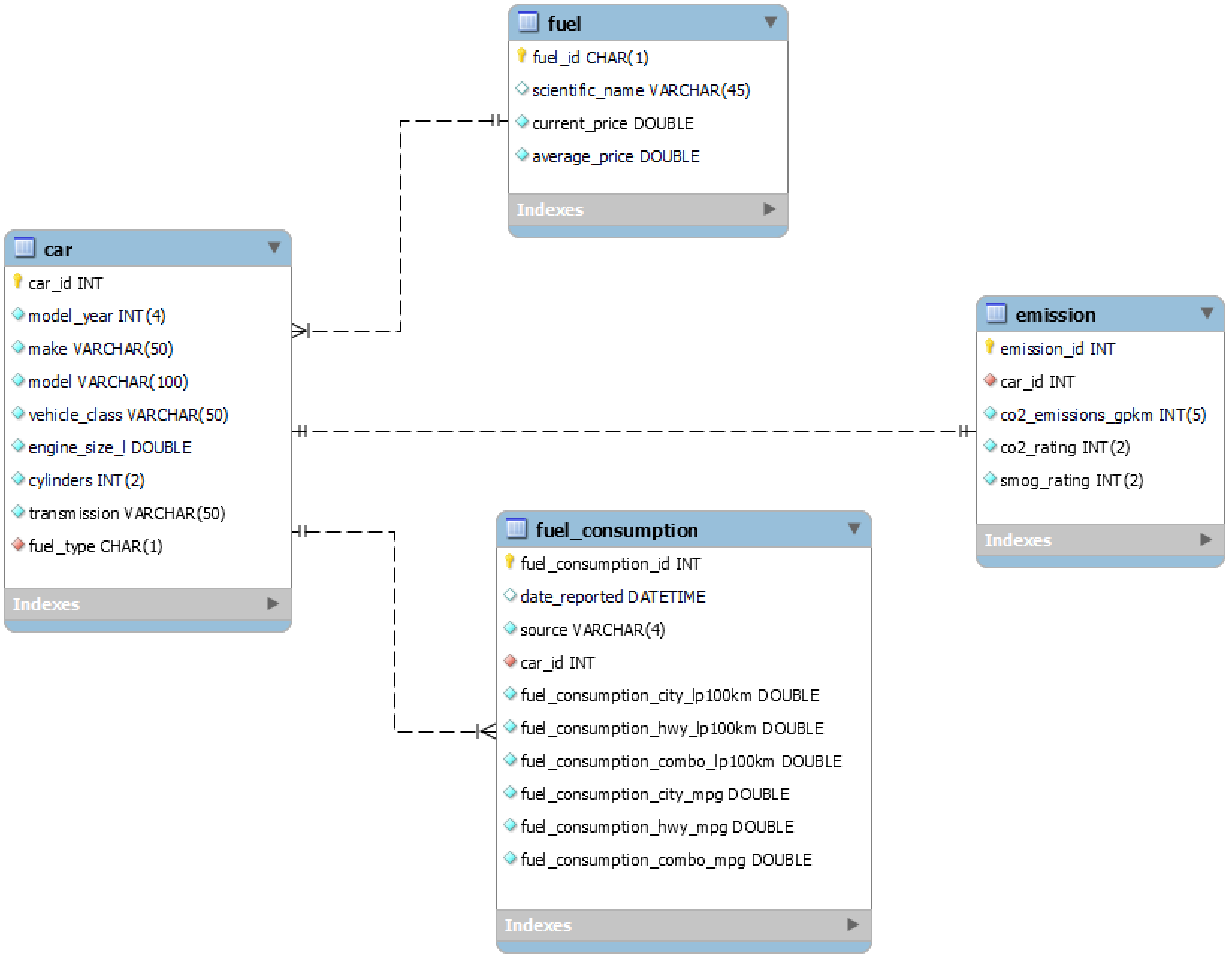
Team

Anne Wesley (aewesley@indiana.edu),

Shubham Saurabh (ssaurab@iu.edu),

Varun Joshi (vajoshi@iu.edu)

1. Database Schema



The above diagram represents the database modeling for our database application. The structure of the model has the following entities:

- Car
    - Attributes for each car model.
- Fuel
    - Attributes for each fuel type and its price.
- Fuel Consumption
    - Attributes for fuel consumption rating for each car model.
- Emission
    - Attributes for emission rating documented for each car model.

The entities are related as per the following rules:

- Car- Fuel

    - Many-to-one. Many car models have the same fuel type.
- Car-Fuel_Consumption
    - One-to-many. The fuel consumption table has attributes **source** and **date_reported** to capture user reported fuel consumption in addition to the fuel consumption rating provided by the car manufacturer. Hence, one-to-many relationship between the entities.
- Car-Emission
    - One-to-one. Each car model has one emission rating reported by the manufacturer.

2. Database

a. Constraints

The following Referential Integrities exist in the schema:

Serial No.	**Constraint Name**	**Foreign Key**		**Primary Key**	
		Referencing Table	**Referencing Column**	**Referenced Table**	**Referenced Column**
1	FK_Fuel	car	fuel_type	fuel	fuel_id
2	FK_Car_Fuel	fuel_consumption	car_id	car	car_id
3	FK_Car_Emission	emission	car_id	car	car_id

Following column level constraints are used:

Serial No.	**Table**	**Column**	**Constraint Description / Comment**
1	fuel	fuel_id	Primary key for fuel table. Using the Fuel Type as a unique single character Primary Key. We cannot have more than 26 types of fuel in the system. It is future-proof for cars.
2	fuel	current_price	Value defaults to 0 until updated by the user.
3	fuel	average_price	Value defaults to 0 until updated by the user.
4	car	car_id	Auto incrementing primary key to identify cars. Also used to join with other tables.

Serial No.	Table	Column	Constraint Description / Comment
5	car	fuel_type	Foreign Key to identify the fuel used.
6	fuel_consumption	fuel_consumption_id	Auto incrementing primary key to identify each entry. May be needed for debugging.
7	fuel_consumption	source	'LAB' and 'USER' are the only two valid values. Used to indicate if the values were reported by a Lab Test or End User.
8	fuel_consumption		Foreign Key to identify the car for the fuel consumption record.
9	emission	emission_id	Auto incrementing primary key to identify each entry. May be needed for debugging.
10	emission	car_id	Foreign Key to identify the car for the emission record. Also has a UNIQUE constraint to maintain one-to-one relation.
11	(All)	(Most)	Most of the columns (except those described below) have a NOT NULL constraint because their values are required in order for the application to provide full functionality
12	fuel	scientific_name	NULL is allowed because it is not trivial to any functionality
13	fuel_consumption	date_reported	NULL is allowed
14	fuel_consumption	fuel_consumption_city_mpg	Value automatically derived from column fuel_consumption_city_lp100km
15	fuel_consumption	fuel_consumption_hwy_mpg	Value automatically derived from column fuel_consumption_hwy_lp100km
16	fuel_consumption	fuel_consumption_combo_mpg	Value automatically derived from column fuel_consumption_combo_lp100km

b. Views

There are no views in the schema for now, but they might be introduced based on front-end integration requirements.

c. Functions

- DB Functions

The `fuel_consumption` table has three columns which have auto-generated values. The fuel consumption for each vehicle is being stored in two units: *liters per 100 kilometers (l/100km)* and *miles per gallon (mpg - US)*. The data is being entered only in the l/100km units and the following formula is used to convert into mpg:

- ➢ `fuel_consumption_city_mpg = 235.215 / fuel_consumption_city_lp100km`
- ➢ `fuel_consumption_hwy_mpg = 235.215 / fuel_consumption_hwy_lp100km`
- ➢ `fuel_consumption_combo_mpg = 235.215 / fuel_consumption_combo_lp100km`

- Python Functions

  We intend to build our application using Flask in Python. There will be Python functions to support CRUD operations on the database routed through Flask.

  - ➢ Create Function - allows users to report real world fuel consumption ratings. This will have an insert statement at the backend which will target the fuel consumption table.

  - ➢ Read Function - allows users to read the fuel consumption, car, and emissions information. The backend tables that will be called are fuel-consumption, car, and emission with appropriate select and where clause statements to filter the results as requested by the user.

  - ➢ Update Function - allows users to update the previously reported real world fuel consumption values. The backend table that will be called is fuel consumption with update statements. It also allows users to update average fuel price with fuel as the target table for update.

  - ➢ Delete Function - allows users to delete previously reported real world fuel consumption values.

d. Procedures

There are no PL/SQL procedures in the schema for now, but they might be introduced based on front-end integration requirements.

e. Queries

We built a variety of queries that explore aspects of our dataset that contain relevant information that a user of our application would want to explore. An overview of these queries is as follows:

- We built multiple queries that return fuel consumption, CO2 emissions, and smog ratings for a specific make, model, and model year of vehicle.
  - This query would be useful for a user who wants to better understand how their car, or a car that they are interested in, impacts the environment.
- We built a similar query that pulls fuel consumption, CO2 and smog rating for a more general model of cars such as all Fords.
  - The output of this query would be useful to a user who wanted to understand the environmental impacts of a certain brand of car, but wasn't sure which specific make or model they wanted to look at.
- Additionally, we built some more general queries that would enable a user to see across all makes and models which vehicles have the best fuel consumption, CO2 rating and smog rating.
  - These queries could be limited to a certain number of results as input by the user, such as the top 5, or they could be filtered based on criteria such as vehicles with mpg greater than 25.

3. Individual Assessment

Assessment	Comments
Task Completion Satisfaction [1-10 scale; 10 highest]	
Team Work	
Time Commitment	
Upcoming Work/Enhancements	