**UsedCarCentral**

**The Online Hub for Buying and Selling Used Cars**

**Project Description:**

Using the Kaggle dataset, we aim to build a web application that allows users to post or edit their existing used car listings (sellers), as well as browse, add to favorites or cart, and buy used cars (buyers). Users can register as either a seller or a buyer on the platform.

Our objectives for this project are to:

* Create a user-friendly web application for buying and selling used cars.
* Allow users to easily search for and find the car they are looking for based on their preferences (SELECT statements).
* Provide sellers with a platform to easily list and manage their used car listings (Insert, delete and Update statements).
* Create a safe and secure platform for transactions between buyers and sellers.

**Team:**

* Ram Kiran Devireddy (radevir)
* Syam Prajwal Kammula (skammul)
* Revanth Posina (rposina)

**Schemas:**

Our project utilizes two schemas, namely the "Test Schema" and "Real Schema". The Test Schema was created to facilitate data loading, table creation, and testing of various table creation methods. This helped to bring greater clarity to the data and enabled us to create meaningful data divisions. Once we had finalized the data structure, we moved the code and data to the Real Schema, which serves as the main schema for our project. From this schema, we will be sending data to the frontend based on requested calls and manage the data in an organized and efficient manner. By utilizing these schemas, we can effectively manage and analyze our data and experiment with different methods of organizing it while ensuring its safety and its support for our project objectives.

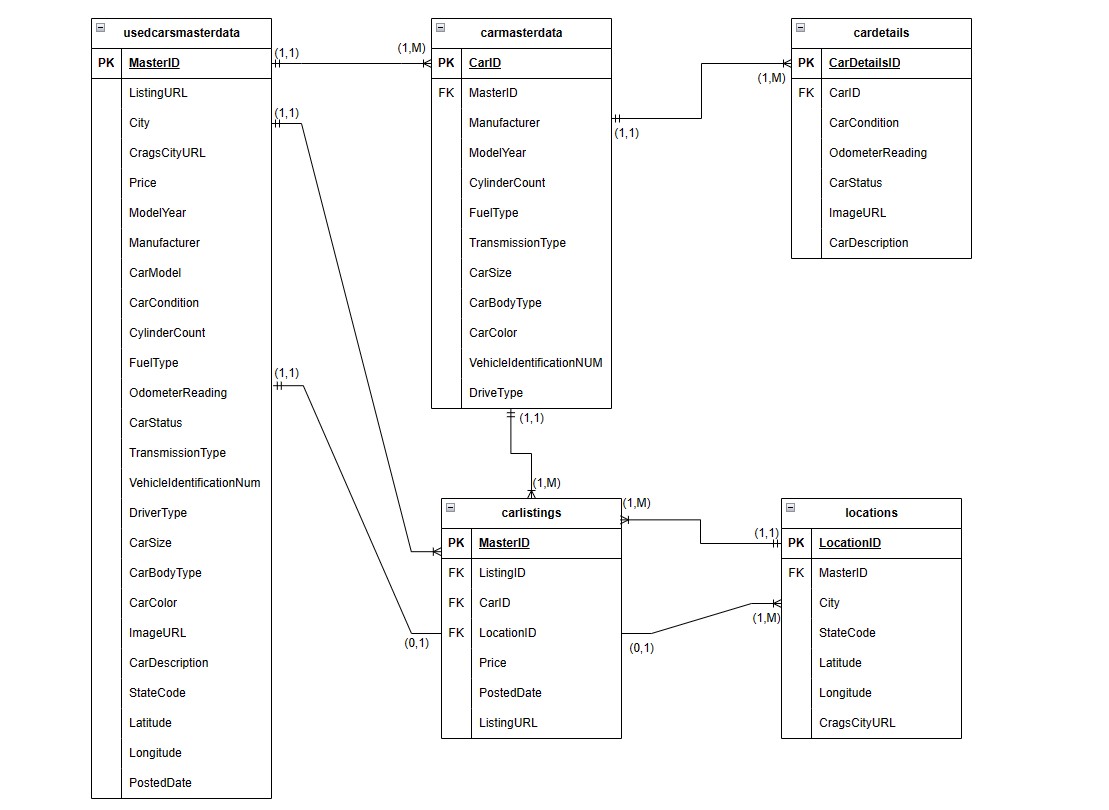
**Data Base Constraints:**

A **primary key** (**PK**) is a field or set of fields in a table that uniquely identifies each record. It is used to enforce integrity constraints and ensure that each record is distinct.

A **foreign key (FK)** is a field or set of fields in one table that references the primary key of another table. It is used to establish relationships between tables and ensure that data is consistent across them. FKs help maintain data integrity by preventing actions that would leave orphaned records or violate referential integrity.

The relationship between the tables can be described as follows:

* The UsedCarsMasterData table has a one-to-many relationship with the CarsMasterData table, as one used car can have multiple car models.
* The CarsMasterData table has a one-to-many relationship with the CarDetails table, as one car can have multiple details such as odometer reading and car condition.
* The UsedCarsMasterData table has a one-to-many relationship with the Locations table, as one used car can be available in multiple locations.
* The UsedCarsMasterData table has a one-to-many relationship with the CarListings table, as one used car can have multiple listings or posts.



**Nulls:**

We discovered that the "County" column in our raw data contains only null values. Upon analyzing the data, we determined that this column does not contribute any value to our project, and we identified other columns that provide similar information and meet our requirements. Therefore, we made the decision to remove the "county" column entirely and proceeded with further data processing.

**Uniqueness:**

To ensure the uniqueness of our data, we have implemented several constraints across multiple tables in our database. These constraints include unique IDs such as “MasterID”, "CarID", "CarDetailsID", "LocationID", and "ListingID" in their respective tables. However, the primary source of uniqueness is the "UsedCarsMasterData" table, which serves as the main master table in our database. This table utilizes a "BIGINT" datatype to ensure that each record is unique, providing us with a reliable and consistent way to identify and manage our data. With these constraints in place, we can maintain the integrity of our data and provide our users with a reliable and trustworthy platform for buying and selling used cars.

**Table 1**: UsedCarsMasterData

* Primary key: MasterID
* The MasterID column ensures that each record in the table is unique.

**Table 2**: CarsMasterData

* Primary key: CarID
* Foreign key: MasterID (from UsedCarsMasterData)
* The CarID column ensures that each record in the table is unique. The MasterID foreign key links each record in this table to a specific record in the UsedCarsMasterData table.

**Table 3**: CarDetails

* Primary key: CarDetailsID
* Foreign key: CarID (from CarsMasterData)
* The CarDetailsID column ensures that each record in the table is unique. The CarID foreign key links each record in this table to a specific record in the CarsMasterData table.

**Table 4**: Locations

* Primary key: LocationID
* Foreign key: MasterID (from UsedCarsMasterData)
* The LocationID column ensures that each record in the table is unique. The MasterID foreign key links each record in this table to a specific record in the UsedCarsMasterData table.

**Table 5**: CarListings

* Primary key: ListingID
* Foreign keys: MasterID (from UsedCarsMasterData), CarID (from CarsMasterData), LocationID (from Locations)
* The ListingID column ensures that each record in the table is unique. The foreign keys link each record in this table to specific records in the other tables. The MasterID foreign key links each record in this table to a specific record in the UsedCarsMasterData table. The CarID foreign key links each record to a specific record in the CarsMasterData table. The LocationID foreign key links each record to a specific record in the Locations table.

**Code Base (Github):**

The whole code base for our project includes code and documentation.

[**https://github.com/ramkiran55/UsedCarCentral\_Team\_SRR**](https://github.com/ramkiran55/UsedCarCentral_Team_SRR)

**Views, Procedures and Functions:**

We plan on creating a few views, procedures, and functions from our tables for better methodology. Please note that these views, functions and procedures are future speculation and might subject to change based on the code requirements **Views**:

Instead of using multiple complex select queries with multiple joins, we’ll be using the following views for data retrieval from the database.

* AveragePriceByCity: calculates and displays the average price of used cars in each city.
* ManufacturerCount: shows the number of cars in the database for each manufacturer.
* HighMileageCars: displays all cars with an odometer reading over 100,000 miles.

**Functions**:

* CalculateDepreciation: takes in a car's manufacturer, model year, and odometer reading and returns an estimate of the car's depreciation value.
* GetCarAge: takes in a car's model year and returns its age in years.
* GetCarColorHex: takes in a car's color and returns its hex code value for web display.

**Procedures**:

* UpdatePrice: takes in a CarID and a new price and updates the price for that car in the CarsMasterData and CarListings tables.
* AddNewLocation: takes in a MasterID, city, state code, and latitude/longitude coordinates, and adds a new location entry to the Locations table.
* DeleteListing: takes in a ListingID and removes the corresponding listing from the CarListings table.
* AddListing: takes in a ListingID, Car details and Location to add the corresponding listing to the database.

**Formatting:**

The database mainly consists of five tables: UsedCarsMasterData, CarsMasterData, CarDetails, Locations, and CarListings.

The UsedCarsMasterData table serves as the main table and contains information about each used car being tracked, including its make, model, year, condition, mileage, and location. This table is linked to several other tables via foreign keys.

The CarsMasterData table contains similar information to the UsedCarsMasterData table but is linked to it via a foreign key and serves to provide more detailed information about each car, including its cylinder count, fuel type, and transmission type.

The CarDetails table contains even more specific details about each car, such as its current status and description, as well as a link to an image of the car. The Locations table contains information about the location of each car, including its city, state, and geographic coordinates.

The CarListings table serves as a way to track each individual listing for a car, including the date posted, price, and URL for the listing page. This table is linked to the other tables via foreign keys to ensure that all relevant information is connected to each individual listing.

Finally, the Locations table in the database stores information about the cities where the used cars are available. It includes the City name, State code, and the corresponding latitude and longitude coordinates. The table also contains the URL for the city's attractions and details. The primary key in the Locations table is LocationID. It is used to uniquely identify each record in the table. The MasterID column serves as a foreign key from the UsedCarsMasterData table. This allows for a relationship to be established between the two tables, where one Location record corresponds to one or many records in the UsedCarsMasterData table.

**Assessment Table:**

|  |  |  |
| --- | --- | --- |
| **Name: Ram Kiran Devireddy (radevir)** |  | |
| Criteria | Score (1-10) | |
| **Task completion** | 9 | |
| **Teamwork** | 10 | |
| **Time Commitment** | 8 | |
| **What could be done better** | We could’ve spent more time normalizing our database to reduce data redundancy. We could’ve split the database into a few more meaningful tables and taken this normalization form to 3NF. We will try to improve our normalization for this database in the future if possible. | |
| **Name: Syam Prajwal Kammula (skammul)** | |  | |
| Criteria | | Score (1-10) | |
| **Task completion** | | 10 | |
| **Teamwork** | | 10 | |
| **Time Commitment** | | 9 | |
| **What could be done better** | | I think we should have discussed more about the future scope of the project and fixed views and functions. We will make sure to discuss this further and fix our methodology thereby deciding on what views and procedures we need. | |

**Name: Revanth Posina (rposina)**

|  |  |
| --- | --- |
| Criteria | Score (1-10) |
| **Task completion** | 9 |
| **Teamwork** | 9 |
| **Time Commitment** | 8 |
| **What could be done better** | Improving indexing can enhance the database's performance. Frequent searches, like primary keys, foreign keys, and fields used in JOIN statements, should be indexed appropriately. We will have to give indexing for our primary keys in the future. |