CS 4347 Deliverable Two

**Topic**

* Car Listing

**Group members:**

* Aashutosh Krishnan
* Kevin Puga
* Hoang Tran
* John Thomas
* Saurabh Mittal
* Edith Solorzano
* Blanca Berrios

**What you will be doing**

Our goal is to develop a database of used cars and create a front-end interface that allows users to view a listing and browse through the available entries in the database. The system will provide a user-friendly way to search and filter car options based on the various attributes such as make, model, year, color, features, location and price.

**A detailed description of your motivation (why you chose to do this particular project),**

**where you expect your design to be used in real life.**

Our motivation to develop this project was due to the increasing demand for easy to use platforms where consumers can browse and compare used cars. We were inspired by the similar systems that are already in the market and wanted to create one that is scalable and easy to maintain as our system will not limit to used cars that are only at one dealer.

It could be used by car dealerships, independent sellers and buyers who are looking for an easier way to buy and sell vehicles. It also can be apple in local markets, online advertise, event, dealership management software. We will scale in and compete with Carmax or AutoTrader.

**• The list of tasks delegated to each member: the task delegation should be distributing what to do from start to completion of the project to each member of your group. It does not have to be too detailed. One sentence for each member is sufficient.**

Front end: Blanca, Aashutosh, and Kevin will create the UI, and listing, filtering and managing user authentications.

Back end: Hoang, Saurabh, John, and Edith will set up the server, database integration and functional requirements.

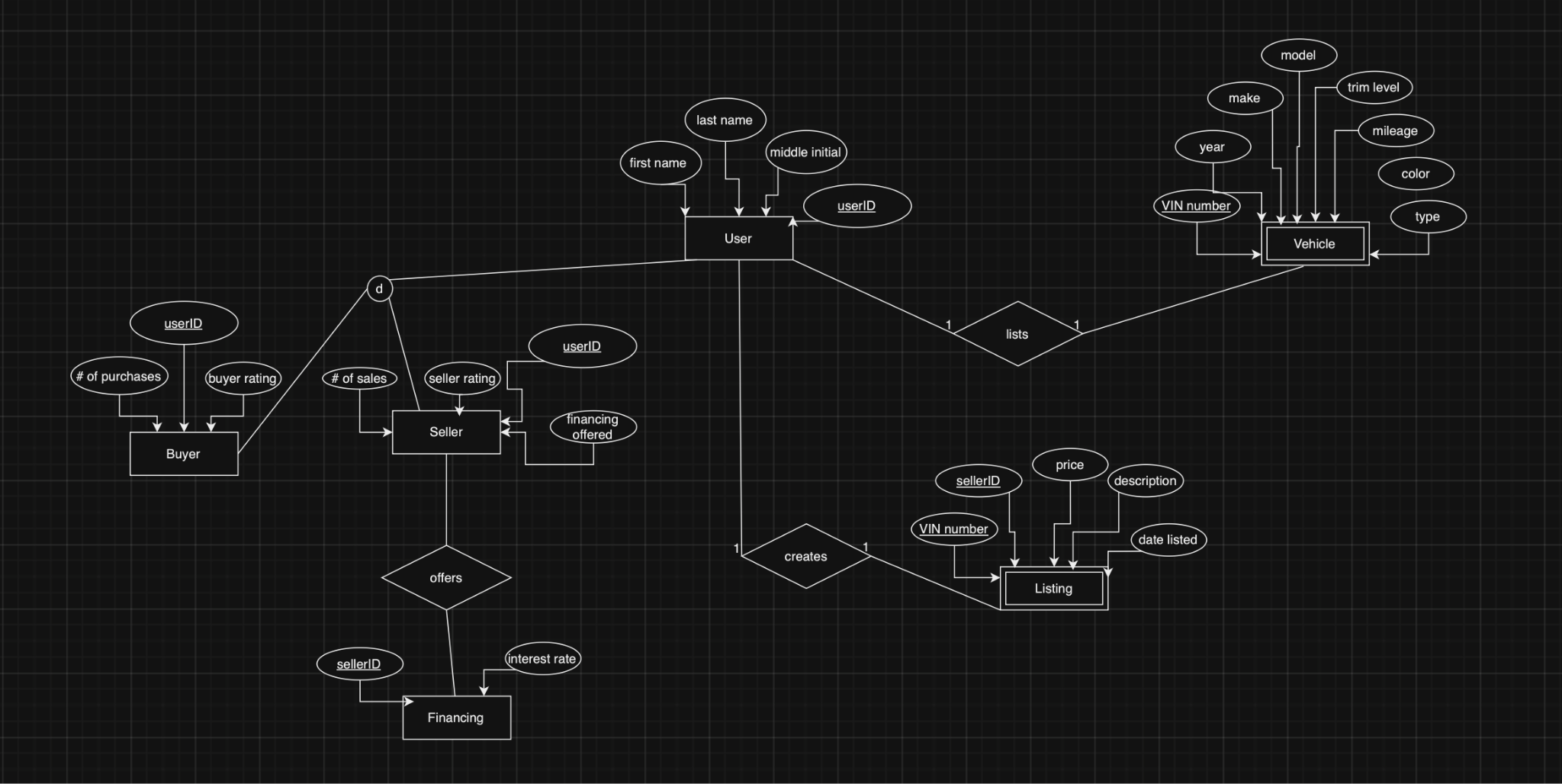
Database: All of us will handle database design and implementation, for the purpose of this class.

• **Are you interested in writing a scholar paper in the end (no extra grade advantage)**

* No

**[59 POINTS] 3. Design & Implementation (Phase II):**

[15 POINTS] 3.1. Normalization of EER Conceptual Data Model: Use your original EER design from Final project deliverable 1, section 3.1. to normalize all your tables in your database to 3NF. Clearly specify what changes are made and explain why those changes were needed for normalization. Please note that if your tables in deliverable 1 were already in 3NF, then there will be no change on these tables for deliverable 2. If this is the case, you should still specify clearly and justify why you did not change your tables.



[10 POINTS] 3.2. Relational Data Model Design Using Normalized EER Diagram: Use your normalized EER diagram to perform EER to relational model mapping and create the following: Draw the schema diagram of your design USING A TOOL, rather than drawing manually. Specify Primary Keys, Foreign Keys and referential integrity constraints in this schema. Please note that if your tables in deliverable 1 were already in 3NF, then there will be no change on these tables for deliverable 2. If this is the case, you should still specify clearly and justify why you did not change your tables. Provide your final EER diagram reflecting the decompositions (if any) w.r.t. the 3NF normalization process you have implemented. Again, your EER diagram should present all relations, primary key constraints, and referential integrity constraints.



In our relational schema of 5 tables, 4 of the tables (Vehicle, Listing, Buyer, Seller, and Financing) were not changed as they are already in 3NF. In the Vehicle table, VIN number is the primary key, and it forms a functional dependency with all the remaining vehicle attributes. There are no other functional dependencies. In the Listing table, Seller ID and VIN number form the primary key, and they form a functional dependency with all the remaining listing attributes. The Buyer, Seller, and Financing tables all use a buyer/seller ID as the primary key, and that value forms a functional dependency with all remaining attributes in the table. As such, all four of these tables are in 3NF.

The one change was to the User table. Originally, user ID and username were two separate fields, with user ID being the primary key, but this created a dependency on a non-prime attribute, because the username must be unique to each user and thus also forms a dependency with the remaining user attributes. To resolve this and get the table into 3NF, we merged the user ID and username fields together, now using the username string as the user identifier and the primary key of the table.

However, we were unable to reflect this change in our database implementation, as it was in a development stage where making the change would require many other components to be rebuilt.

[15 POINTS] 3.3. Create your Normalized Database and Populate: Use an SQL platform to create your database and its tables as you described in sections 3.1 and 3.2. Then, populate each table with corresponding data. Remember that each table should have a minimum of 10 tuples. Provide a screenshot of each database creation statement, and each database table population, as well as the resulting table statement in your report exactly in this section. Please note that if your tables in deliverable 1 were already in 3NF, then there will be no change on these tables for deliverable 2. If this is the case, you should still specify clearly and justify why you did not change your tables. -

**Database:**

**Creation code:**

CREATE DATABASE Project;

CREATE TABLE Users (

userID INT PRIMARY KEY,

userName VARCHAR(50) NOT NULL,

firstName VARCHAR(50) NOT NULL,

middleInit CHAR(1),

lastName VARCHAR(50) NOT NULL );

CREATE TABLE Vehicle (

vinNum VARCHAR(17) PRIMARY KEY,

year INT NOT NULL, make VARCHAR(50) NOT NULL,

model VARCHAR(50) NOT NULL,

trim\_lvl VARCHAR(50),

mileage INT NOT NULL,

color VARCHAR(30),

type VARCHAR(20) );

CREATE TABLE Listing (

sellerID INT,

vinNum VARCHAR(17),

price DECIMAL(10, 2) NOT NULL,

description TEXT,

dateListed DATE NOT NULL,

PRIMARY KEY (sellerID, vinNum),

FOREIGN KEY (sellerID) REFERENCES Users(userID) ON UPDATE CASCADE ON DELETE CASCADE,

FOREIGN KEY (vinNum) REFERENCES Vehicle(vinNum) ON UPDATE CASCADE ON DELETE CASCADE );

CREATE TABLE Buyer (

userID INT PRIMARY KEY,

numOfPurchase INT DEFAULT 0,

buyerRating FLOAT CHECK (buyerRating BETWEEN 1 AND 5),

FOREIGN KEY (userID) REFERENCES Users(userID) ON UPDATE CASCADE ON DELETE CASCADE );

CREATE TABLE Seller (

userID INT PRIMARY KEY,

numOfSale INT DEFAULT 0,

sellerRating FLOAT CHECK (sellerRating BETWEEN 1 AND 5),

financingOffered BOOLEAN,

FOREIGN KEY (userID) REFERENCES Users(userID) ON UPDATE CASCADE ON DELETE CASCADE );

CREATE TABLE FinancingInfo (

sellerID INT PRIMARY KEY,

interestRate DECIMAL(3, 2) CHECK(interestRate BETWEEN 0 AND 100),

FOREIGN KEY (sellerID) REFERENCES Seller(userID) ON UPDATE CASCADE ON DELETE CASCADE );

**Populating Code:**

INSERT INTO Users (userID, userName, firstName, middleInit, lastName) VALUES

(1, 'john\_doe', 'John', 'A', 'Doe'),

(2, 'jane\_smith', 'Jane', 'B', 'Smith'),

(3, 'mark\_johnson', 'Mark', 'C', 'Johnson'),

(4, 'lisa\_brown', 'Lisa', 'D', 'Brown'),

(5, 'peter\_parker', 'Peter', 'E', 'Parker'),

(6, 'clark\_kent', 'Clark', 'F', 'Kent'),

(7, 'bruce\_wayne', 'Bruce', 'G', 'Wayne'),

(8, 'tony\_stark', 'Tony', 'H', 'Stark'),

(9, 'natasha\_romanoff', 'Natasha', 'I', 'Romanoff'),

(10, 'diana\_prince', 'Diana', 'J', 'Prince');

INSERT INTO Vehicle (vinNum, year, make, model, trim\_lvl, mileage, color, type) VALUES

('1HGCM82633A123456', 2005, 'Honda', 'Accord', 'EX', 120000, 'Blue', '4-door'),

('1FTFW1ET0EKE12345', 2014, 'Ford', 'F-150', 'Lariat', 80000, 'Red', '4-door'),

('2FMDK3KC7DBA98765', 2013, 'Ford', 'Edge', 'SEL', 60000, 'Black', '4-door'),

('1C4RJFBGXEC123456', 2014, 'Jeep', 'Grand Cherokee', 'Limited', 70000, 'White', '4-door'),

('1N4AL3AP7EC123456', 2014, 'Nissan', 'Altima', '2.5 SV', 50000, 'Silver', '4-door'),

('1HGCR2F3XFA123456', 2015, 'Honda', 'Civic', 'LX', 30000, 'Gray', '4-door'),

('1GNEK13Z03J123456', 2003, 'Chevrolet', 'Tahoe', 'LT', 150000, 'Green', '4-door'),

('1ZVBP8AM1A1234567', 2010, 'Ford', 'Mustang', 'GT', 45000, 'Yellow', '2-door'),

('1FAHP2E84BG123456', 2011, 'Ford', 'Taurus', 'SEL', 70000, 'Blue', '4-door'),

('2C4RDGEG7ER123456', 2014, 'Chrysler', 'Town & Country', 'Limited', 80000, 'Brown', '4-door');

INSERT INTO Listing (sellerID, vinNum, price, description, dateListed) VALUES

(1, '1HGCM82633A123456', 8000.00, 'Well maintained Honda Accord.', '2023-10-01'),

(2, '1FTFW1ET0EKE12345', 25000.00, 'Great condition Ford F-150.', '2023-09-15'),

(3, '2FMDK3KC7DBA98765', 22000.00, 'Family-friendly Ford Edge.', '2023-08-20'),

(4, '1C4RJFBGXEC123456', 30000.00, 'Luxury Jeep Grand Cherokee.', '2023-07-30'),

(5, '1N4AL3AP7EC123456', 15000.00, 'Economical Nissan Altima.', '2023-06-25'),

(6, '1HGCR2F3XFA123456', 18000.00, 'Reliable Honda Civic.', '2023-10-10'),

(7, '1GNEK13Z03J123456', 15000.00, 'Spacious Chevrolet Tahoe.', '2023-10-05'),

(8, '1ZVBP8AM1A1234567', 30000.00, 'Classic Ford Mustang GT.', '2023-09-01'),

(9, '1FAHP2E84BG123456', 16000.00, 'Comfortable Ford Taurus.', '2023-09-12'),

(10, '2C4RDGEG7ER123456', 21000.00, 'Versatile Chrysler Town & Country.', '2023-09-20');

INSERT INTO Buyer (userID, numOfPurchase, buyerRating) VALUES

(1, 5, 4.5),

(2, 10, 4.8),

(3, 2, 4.0),

(4, 3, 4.2),

(5, 1, 5.0),

(6, 4, 3.8),

(7, 6, 4.1),

(8, 5, 4.7),

(9, 2, 4.6),

(10, 7, 4.9);

INSERT INTO Seller (userID, numOfSale, sellerRating, financingOffered) VALUES

(1, 10, 4.6, TRUE),

(2, 8, 4.9, FALSE),

(3, 5, 4.5, TRUE),

(4, 3, 4.3, FALSE),

(5, 7, 4.8, TRUE),

(6, 9, 4.1, FALSE),

(7, 4, 4.4, TRUE),

(8, 12, 4.7, FALSE),

(9, 5, 4.2, TRUE),

(10, 6, 4.5, FALSE);

INSERT INTO FinancingInfo (sellerID, interestRate) VALUES

(1, 3.5),

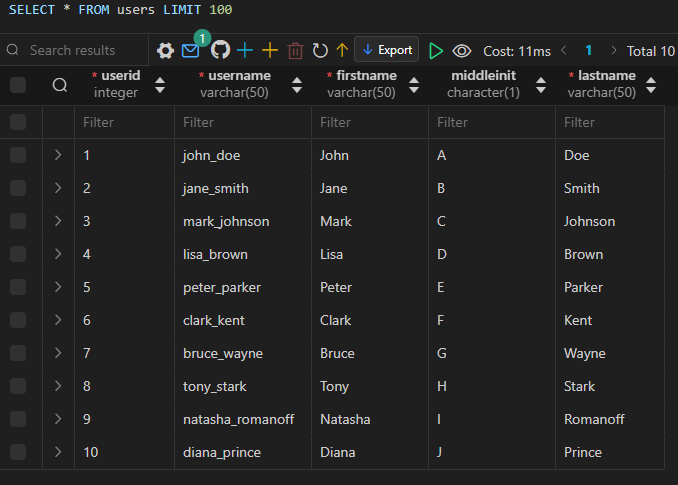
(3, 4.0),

(5, 3.0),

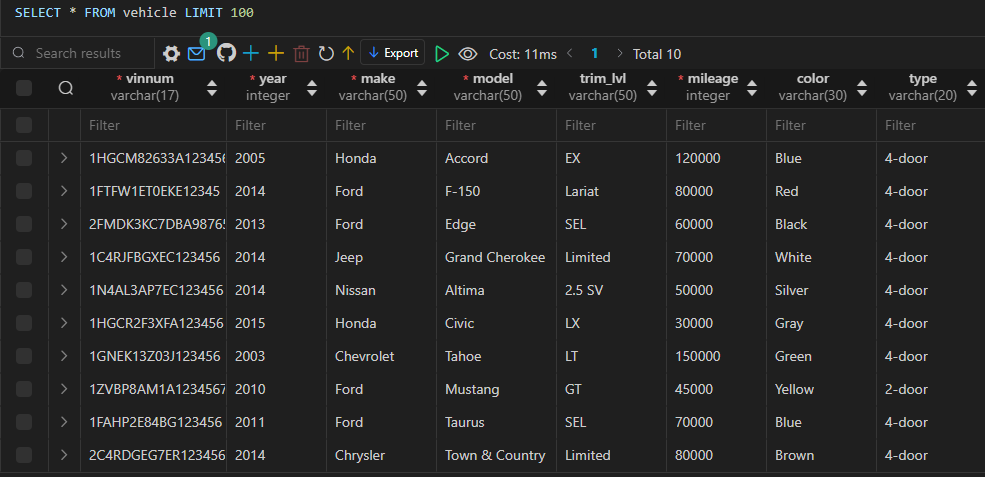
(7, 3.8),

(9, 4.5);

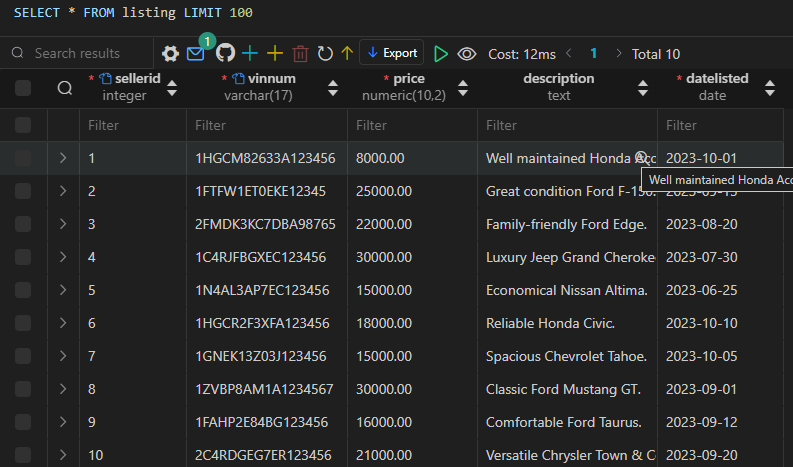
Users:



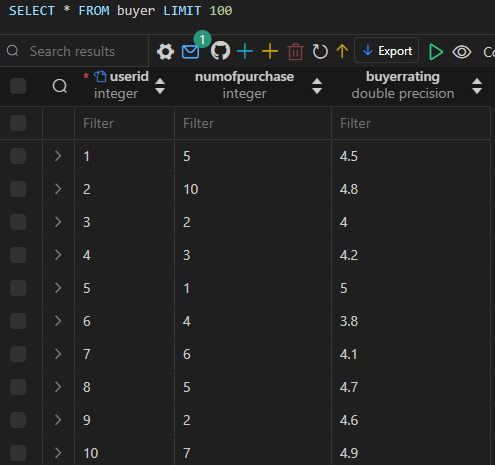
Vehicle:



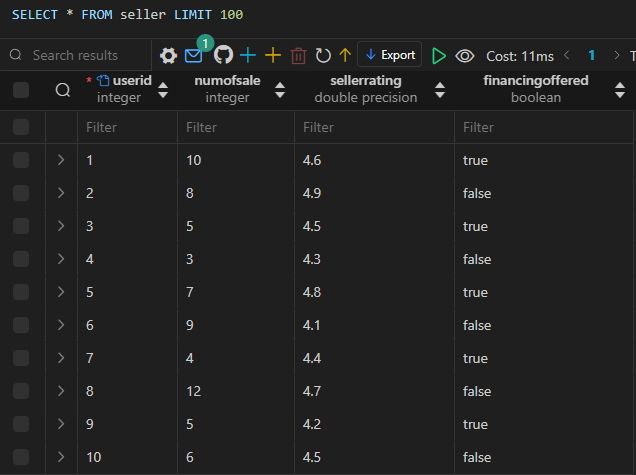
Listing:



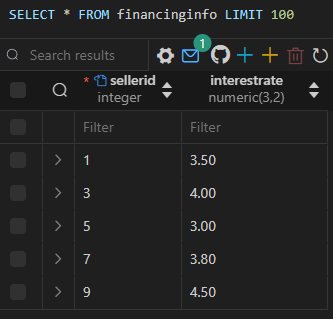
Buyer:



Seller:



Financing:

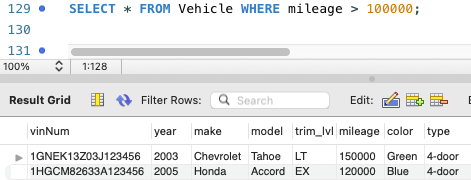


[15 POINTS] 3.4. Database Query Execution on your Normalized Database (from inside your SQL client): Use an SQL platform to provide sample executions for each of the following operations on each table of your normalized database:

* Query – to perform operations such as list employees earning more than 150K per year
* Insert – to add new tuple(s), and/or field(s) to your table(s)
* Delete - to remove tuple(s), and/or field(s) from your table(s)
* Update - to modify tuple(s), and/or field(s) from your table(s)

Provide a screenshot of each operation and each resulting table in your report exactly in this section.

Query:

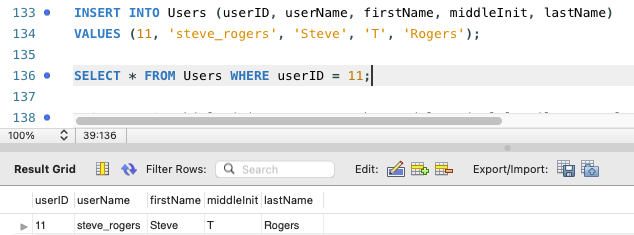


Insert:

Before:



After:

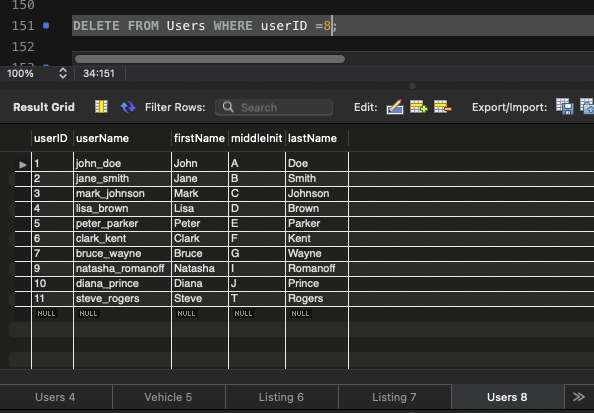


Delete:

Before:

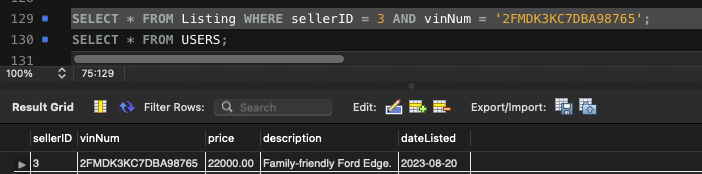


After:

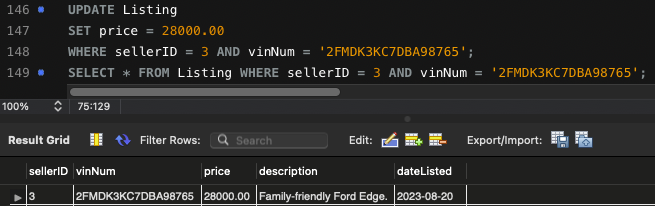


Update:

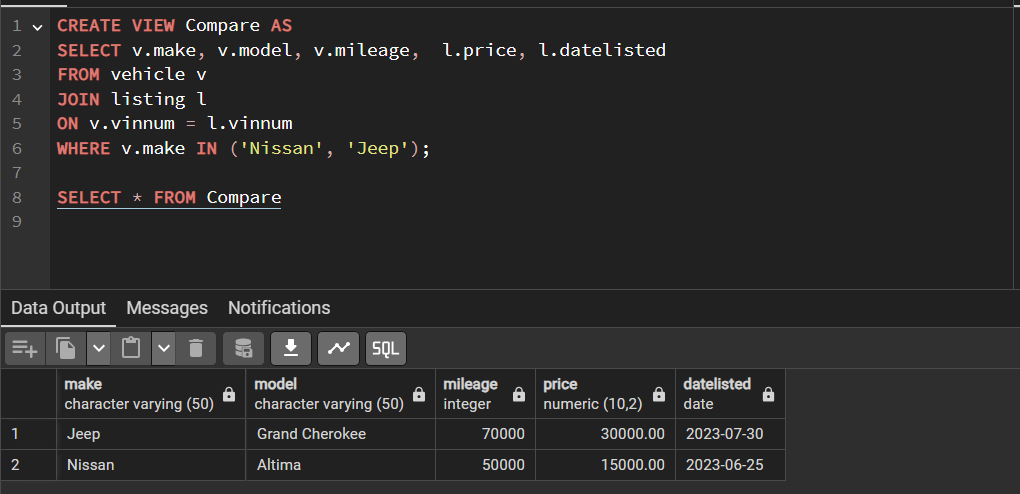
Before:



After:



[4 POINTS] 3.5. Create View: Use minimum one CREATE VIEW statement in your normalized database to implement a view based on your specific database design. Please indicate what this view is for. Provide a screenshot of your view(s), as well as each resulting table in your report exactly in this section.



This view is the frontend query, and what the user will see when selecting a car.

[25 POINTS] 4. Front End User Interface: In this part of your project, you are required to use a programming platform to present a front end that will communicate with your normalized backend database you already created earlier. Please note that you are free to choose a web-based or a non-web based design. One has no (dis)advantage over the other. If you choose to use a web-based design, please note that you will need to have a web server to store and maintain your database. It will be your responsibility to design, install the necessary tools (such as an ER diagram drawing tool, DBMS, web server, compiler, etc.), and implement the database of your design. Using a programming platform of your choice, implement a code to provide the following:

Display a menu with the listed options:

1. Query – to perform operations such as list employees earning more than 150K per year

2. Insert – to add new tuple(s), and/or field(s) to your table(s)

3. Delete - to remove tuple(s), and/or field(s) from your table(s)

4. Update - to modify tuple(s), and/or field(s) from your table(s)

5. Quit

Once the user selects an option, perform the selected option. Provide a screenshot of each of the above 5 menu options, as well as the resulting output displayed by your program in your report exactly in this section. Offer some variety by including more complex queries such as subqueries, aggregates, set operators, etc

QUERY

A screenshot of a car listing

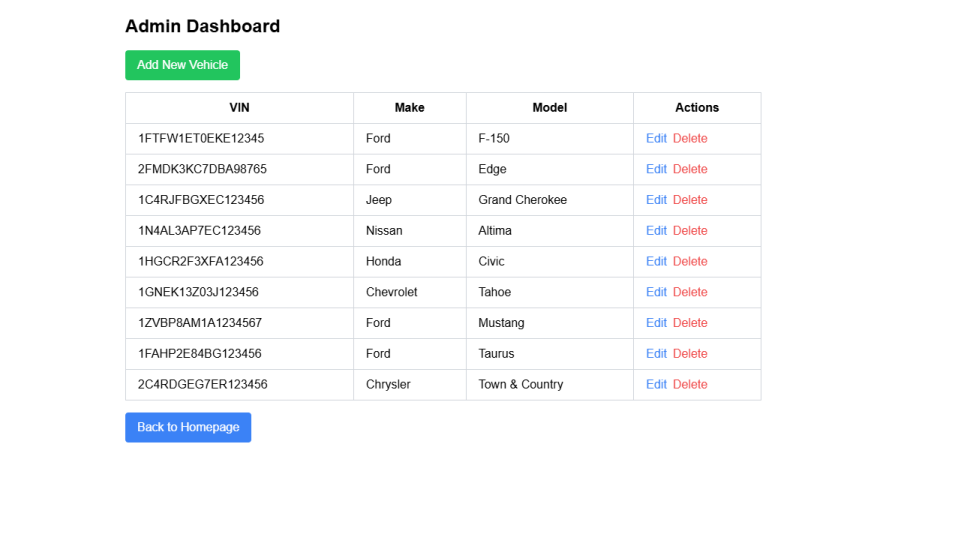
Description automatically generated

A screenshot of a computer

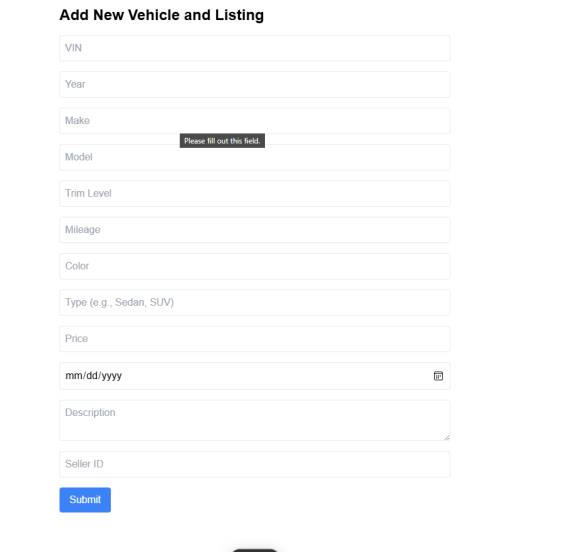
Description automatically generated

This is what the user will see when querying a car.

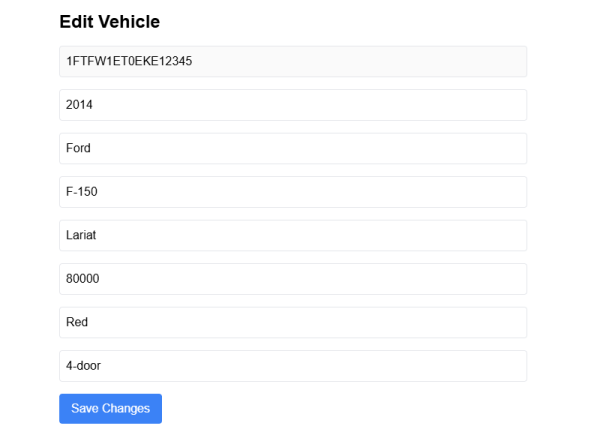
INSERT, DELETE, UPDATE, and QUIT



In this screenshot, you can see insert, update, and delete options present in the screenshot. The “Add New Vehicle” option serves as the insert functionality. The update functionality is done through the edit option. The delete option is also present next to the edit option. The “Back to Homepage” button basically quits out of the database that the user will see and serves as the quit functionality in our UI. These operations outside of QUIT are only permissible for admins.



Insert a new vehicle into the database.



Edit or update a new vehicle in the database.

A screenshot of a dashboard

Description automatically generated

A screenshot of a car dashboard

Description automatically generated

In these two screenshots, you can see that the Honda Accord car has been deleted. Just by clicking the delete button.

[3 POINTS] 5. Conclusion and Future Work: Provide a summary of your work here. Also provide any changes that you need to make (if any), if things have deviated from what you had originally planned for and try to give justification for such changes. Suggest future expansion ideas for your project to promote its validity.

Overall, throughout the project, we didn’t deviate from much prior planning significantly as everyone was diligent in doing their role. The only major change was made was creating a superuser, called an admin which has special privileges such as CRUD (create, read, update, and delete). The reason we did this was so we could have a user that makes sure that there is a user that can make sure the database is running correctly and handle any data constraints. In the future, we could possibly introduce more advanced searching such as car make, model, or color. We could also implement a confirmation pop-up for when we want to delete or confirm changes on the database.

## References:

[1] J. Jiang and Q. Ni, "Relational Models for Vehicle Listings," *Journal of Database Management*, vol. 24, no. 2, pp. 45-59, Feb. 2010.

[2] eBay Motors, "eBay Motors: Buy and Sell Vehicles", eBay, Inc., 2000. [Online]. Available: https://www.ebay.com/motors. [Accessed: Oct. 18, 2024].

[3] J. Smith, "A Review of Online Vehicle Marketplaces: eBay Motors and Beyond," *Journal of E-Commerce*, vol. 20, no. 3, pp. 55-67, Mar. 2008.

[4] Smith, J. "Automotive E-Commerce Trends," 2021.