

Car Mart Documentation

Problem

The problem arose when one of us was actually looking for a used car to purchase after graduating in December. We went through multiple sites to browse their inventory and visited dealerships that specialized in selling at most 2-3 car makes which meant a lot of traveling on our part. For online, there wasn't too many options of comparing across websites which provided extremely frustrating. All in all, the process was extremely tedious and the information was not consolidated in one place. That being said, the goal of our project is to create a unified portal where buyers can easily view all used car listings in their area and contact the buyers, negotiate price, and solidify the purchase all in one spot if they so desire. Our project also benefits sellers by providing a portal where they can make their used cars inventory seen. It benefits individual sellers by giving them the ability to put the car(s) they're trying to sell on the same visible platform as dealerships. Overall, this will provide an avenue for people to log into one system and view all their possible options while allowing sellers to reach out to a wider population.

Requirements Document

Car Mart should have the following functionality:

1. Allow for users to sign up and have their own unique account.
2. Allow for buyers to view all cars that have been put up for sale.
3. Allow for sellers to post their car and allow for buyers to purchase them.
4. Buyers should be able to see cars only within their price limit.
5. Sellers and buyers can both modify and change their profiles and respectively, inventory and carts.
6. The database must be secure and accessible through all functionality.

Data Integrity:

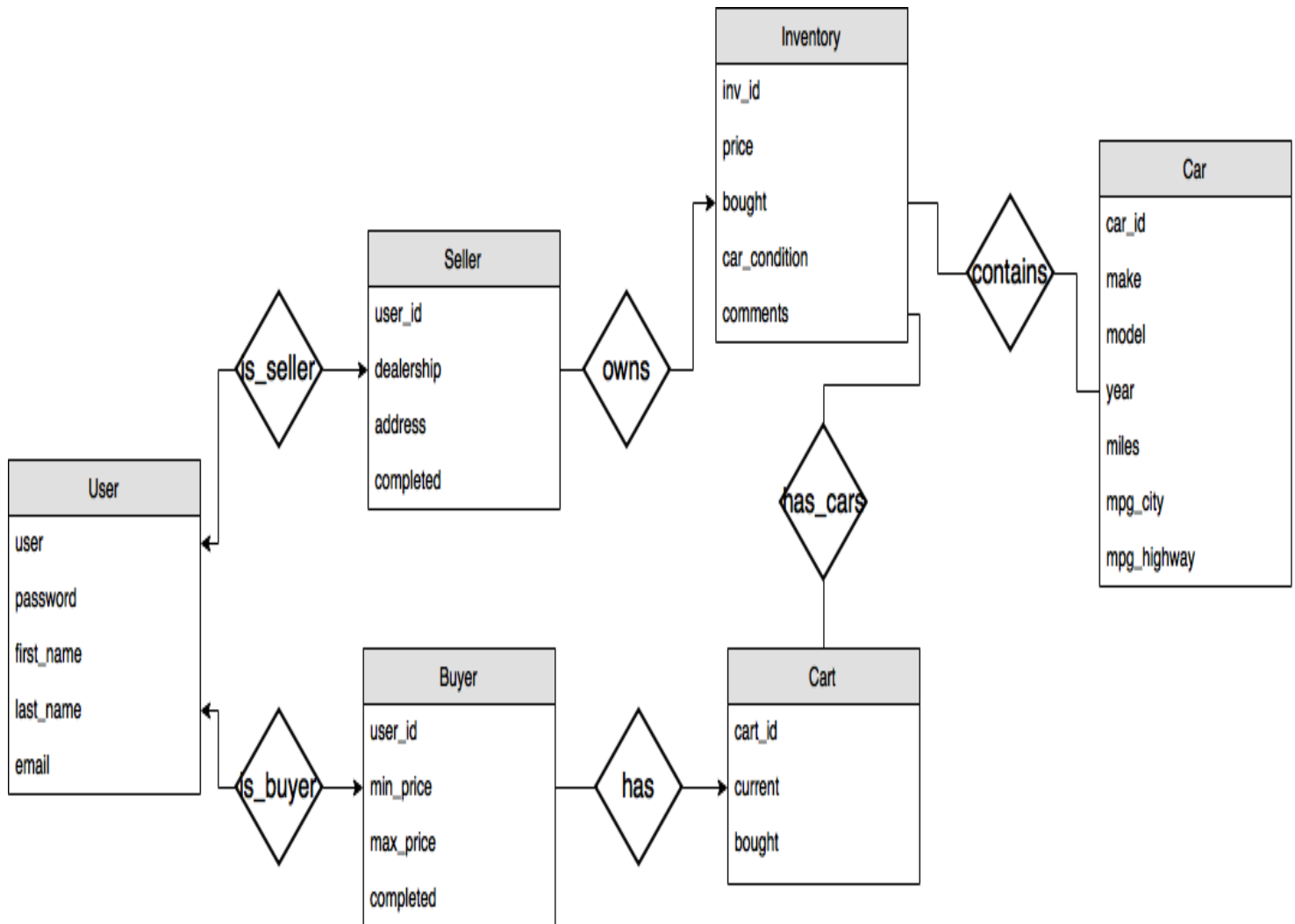
1. Validations are kept on the tables such as min_price for buyers can not be below than 0 nor can it be higher than max_price and similarly, max_price will have a cap of \$150,000 and can not be below than min_price.
2. There cannot be multiples of the same user with the same user name.
3. Primary keys ensure that the data stored is unique and there is no redundancy within the tables.
4. Relationship sets are also correctly stored and only link two tables through two fields, the primary key from each of the tables.
5. The database is in third normal form (as proven below).

Design Process

The project was implemented as a web portal with limited but sufficient functionality. The reasoning behind this was that storing all the information on the database and retrieving it would be easiest on the computer but also, it would make more sense that comparison and

serious searching would be done while sitting down in front of a computer. There was also more expertise in using PHP, HTML, MySQL to implement the entire project over mobile. Due to the lack of time, CSS was not implemented but with a little more time, the website could use Bootstrap or other free custom background/font frameworks could be used.

Place ER Diagram below:



(the primary keys are not underlines but are defined in the following section)

Database Schema

Due to inability of underlining in markdown, all primary keys have been listed instead.

- users(user varchar(30) primary key not null, password text not null, first_name text not null, last_name text not null, email text not null)
- buyers(user_id int primary key not null, min_price int not null, max_price int not null, completed boolean not null)
- sellers(user_id int primary key not null, dealership text not null, address text not null, completed boolean not null)
- cars(car_id int primary key not null, make text not null, model text not null, year int not null, miles int not null, mpg_city int not null, mpg_highway int not null)
- inventory(inv_id int primary key not null, price int not null, bought boolean not null, car_condition text not null, comments text not null)
- cart(cart_id int primary key not null, current boolean not null, bought boolean not null)
- is_seller(user varchar(30) not null, user_id int not null, primary key(user, user_id), foreign key(user) references users(user) on delete cascade, foreign key(user_id) references sellers(user_id) on delete cascade))
- is_buyer(user varchar(30) not null, user_id int not null, primary key(user, user_id), foreign key(user) references users(user) on delete cascade, foreign key(user_id) references buyers(user_id) on delete cascade)
- has(user_id int not null, cart_id int not null, primary key(user_id, cart_id), foreign key(user_id) references buyers(user_id) on delete cascade, foreign key(cart_id) references cart(cart_id) on delete cascade)
- owns(create table owns(user_id int not null, inv_id int not null, primary key (user_id, inv_id), foreign key(user_id) references sellers(user_id) on delete cascade, foreign key(inv_id) references inventory(inv_id) on delete cascade)
- contains(inv_id int not null, car_id int not null, primary key(inv_id, car_id), foreign key(inv_id) references inventory(inv_id) on delete cascade, foreign key(car_id) references cars(car_id) on delete cascade)
- has_cars(inv_id int not null, cart_id int not null, primary key(inv_id, cart_id), foreign key(inv_id) references inventory(inv_id) on delete cascade, foreign key(cart_id) references cart(cart_id) on delete cascade)

Users: user -> password, first_name, last_name, email

Buyers: user_id -> min_price, max_price, completed

Sellers: user_id -> dealership, address, completed

Cars: car_id -> make, model, year, miles, mpg_city, mpg_highway

Inventory: inv_id -> price, bought, car_condition, comments

Cart: cart_id -> current, bought

All RHS entities are dependent on themselves and all LHS entities are superkeys and the above is already the canonical form. Thus, the database is in 3NF based on the above relations. Going even further, the relationship sets that are created only contain two foreign keys, so they are also 3NF as well.

Sample Data and Queries

(A video was submitted alongside with this documentation and I would recommend watching the video to get a better understanding of how our system works. The queries and tables will make much more sense once the video is watched and understood.)

A majority of information was found from Google and the Environmental Protection Agency because they stores csv that contains information regarding past models, year, makes, miles per gallon on highway, miles per gallon in city, and etc. The following section will go into more detail regarding sample queries and some data that was pulled.

Initial data that was used:

```
1#1#Aston Martin#DB9#2015#10000#13#19#100000#Excellent
2#2#Aston Martin#Riptide#2015#20000#14#22#125000#Good
3#3#Audi#A8#2013#50000#18#28#35000#Okay
3#4#BMW#328i#2012#100000#23#34#15000#Excellent
2#5#Chevrolet#Camaro#2013#85000#19#30#10000#Poor
1#6#Maserati#Quattroporte#2010#250000#11#18#42000#Good
2#7#Mercedes Benz#E 550#2009#110000#14#22#55000#Good
2#8#Nissan#Infiniti G#2012#189000#20#29#10000#Okay
3#9#Nissan#Altima#2015#10000#27#38#10000#Excellent
3#10#Lexus#GS 350#2015#32500#19#29#28000#Poor
```
















Some of the data is unclear so going column by column, from left to right: user_id, car_id, make, model, year, miles, mpg (in city), mpg (on highway), price, car_condition. The data was parsed and added into the database using the following commands:

- `$db -> query("insert into cars values('$row_info[1]', '$row_info[2]', '$row_info[3]', '$row_info[4]', '$row_info[5]', '$row_info[6]', '$row_info[7]')");`
- `$db -> query("insert into inventory values('$row_info[1]', '$row_info[8]', '0', '$row_info[9]', '0')");`
- `$db -> query("insert into owns values('$row_info[0]', '$row_info[1]')");`
- `$db -> query("insert into contains values('$row_info[1]', '$row_info[1]')");`

+ Options									
		car_id	make	model	year	miles	mpg_city	mpg_highway	
<input type="checkbox"/>	 Edit  Copy  Delete	1	Aston Martin	DB9	2015	10000	13	19	
<input type="checkbox"/>	 Edit  Copy  Delete	2	Aston Martin	Riptide	2015	20000	14	22	
<input type="checkbox"/>	 Edit  Copy  Delete	3	Audi	A8	2013	50000	18	28	
<input type="checkbox"/>	 Edit  Copy  Delete	4	BMW	328i	2012	100000	23	34	
<input type="checkbox"/>	 Edit  Copy  Delete	5	Chevrolet	Camaro	2013	85000	19	30	
<input type="checkbox"/>	 Edit  Copy  Delete	6	Maserati	Quattroporte	2010	250000	11	18	
<input type="checkbox"/>	 Edit  Copy  Delete	7	Mercedes Benz	E 550	2009	110000	14	22	
<input type="checkbox"/>	 Edit  Copy  Delete	8	Nissan	Infiniti G	2012	189000	20	29	
<input type="checkbox"/>	 Edit  Copy  Delete	9	Nissan	Altima	2015	10000	27	38	
<input type="checkbox"/>	 Edit  Copy  Delete	10	Lexus	GS 350	2015	32500	19	29	
<input type="checkbox"/>	 Edit  Copy  Delete	11	Toyota	Sienna	1999	150000	18	23	

The tables were created using the following commands (main tables and then relationship tables, not going to include all of them):




























- \$db -> query("create table buyers(user_id int primary key not null, min_price int not null, max_price int not null, completed boolean not null)");
- \$db -> query("create table is_buyer(user varchar(30) not null, user_id int not null, primary key(user, user_id), foreign key(user) references users(user) on delete cascade, foreign key(user_id) references buyers(user_id) on delete cascade) engine=innodb");

+ Options			
		user	user_id
<input type="checkbox"/>	 Edit  Copy  Delete	adeel	2
<input type="checkbox"/>	 Edit  Copy  Delete	brad	4
<input type="checkbox"/>	 Edit  Copy  Delete	josh	1
<input type="checkbox"/>	 Edit  Copy  Delete	jw6dz	5
<input type="checkbox"/>	 Edit  Copy  Delete	kobe	3

When a new buyer is created (seller is extremely similar with just a change to table name):

- \$db -> query("insert into users values('\$user', '\$secure_pass', '\$first_name', '\$last_name', '\$email')");
- \$db -> query("insert into buyers values('\$rows', -10, 0, 0)") or ("Invalid: " . \$db -> error);
- \$db -> query("insert into is_buyer values('\$user', '\$rows')") or ("Invalid: " . \$db -> error);

+ Options

			user	password	first_name	last_name	email	
<input type="checkbox"/>				adeel	2cf24dba5fb0a30e26e83b2ac5b9e29e1b161e5c1fa7425e73...	Adeel	Ahmad	adeel@gmail.com
<input type="checkbox"/>				bob	2cf24dba5fb0a30e26e83b2ac5b9e29e1b161e5c1fa7425e73...	Bob	Joe	bob@gmail.com
<input type="checkbox"/>				brad	2cf24dba5fb0a30e26e83b2ac5b9e29e1b161e5c1fa7425e73...	Brad	Pitt	bradpitt@gmail.com
<input type="checkbox"/>				josh	2cf24dba5fb0a30e26e83b2ac5b9e29e1b161e5c1fa7425e73...	Josh	Wang	josh@gmail.com
<input type="checkbox"/>				jw6dz	2cf24dba5fb0a30e26e83b2ac5b9e29e1b161e5c1fa7425e73...	Josh	Wang	jw@gmail.com
<input type="checkbox"/>				kevin	2cf24dba5fb0a30e26e83b2ac5b9e29e1b161e5c1fa7425e73...	Kevin	Qian	kevin@gmail.com
<input type="checkbox"/>				kobe	2cf24dba5fb0a30e26e83b2ac5b9e29e1b161e5c1fa7425e73...	Kobe	Bryant	kobebryant@gmail.com
<input type="checkbox"/>				kq4hy	2cf24dba5fb0a30e26e83b2ac5b9e29e1b161e5c1fa7425e73...	Kevin	Qian	kev@gmail.com
<input type="checkbox"/>				roger	2cf24dba5fb0a30e26e83b2ac5b9e29e1b161e5c1fa7425e73...	Roger	Federer	rogerfederer@gmail.com


































When a seller updates his/her information (buyer is similar):

- \$db -> query("update users set first_name = '\$first_name', last_name = '\$last_name', email = '\$email' where user = '\$curr_user'");
- \$db -> query("update sellers set dealership = '\$dealership', address = '\$address' where user_id = '\$curr_id'");

+ Options				user_id	dealership	address	completed			
<div><div><div></div><div></div></div><div><div></div><div></div></div></div>	<div><div></div><div></div></div>	Edit	<div><div></div><div></div></div>	Copy	<div><div></div><div></div></div>	Delete	1	Car's Central	Virginia	0
<div><div><div></div><div></div></div><div><div></div><div></div></div></div>	<div><div></div><div></div></div>	Edit	<div><div></div><div></div></div>	Copy	<div><div></div><div></div></div>	Delete	2	UVa	Maryland	0
<div><div><div></div><div></div></div><div><div></div><div></div></div></div>	<div><div></div><div></div></div>	Edit	<div><div></div><div></div></div>	Copy	<div><div></div><div></div></div>	Delete	3			1
<div><div><div></div><div></div></div><div><div></div><div></div></div></div>	<div><div></div><div></div></div>	Edit	<div><div></div><div></div></div>	Copy	<div><div></div><div></div></div>	Delete	4	USA Cars	109 Clarke Court	1

When displaying information about the car in a seller's inventory:

- `$db->query("select inv_id, price, car_condition, bought from inventory natural join owns where user_id = '$curr_id'");`

+ Options			inv_id	price	bought	car_condition	comments
<input type="checkbox"/>				1	100000	0	Excellent
<input type="checkbox"/>				2	125000	0	Good
<input type="checkbox"/>				3	35000	1	Okay
<input type="checkbox"/>				4	15000	0	Excellent
<input type="checkbox"/>				5	10000	0	Poor
<input type="checkbox"/>				6	42000	0	Good
<input type="checkbox"/>				7	55000	0	Good
<input type="checkbox"/>				8	10000	0	Okay
<input type="checkbox"/>				9	10000	0	Excellent
<input type="checkbox"/>				10	28000	0	Poor
<input type="checkbox"/>				11	5000	0	Poor

When displaying information about a car that a buyer clicks:

- `$db->query("select distinct make, model, year, miles, mpg_city, mpg_highway, car_condition, price, comments, first_name, last_name, email, dealership, address, bought from inventory natural join owns natural join users natural join sellers natural join cars natural join contains natural join is_seller where inv_id = '$inv_id'");`

There were many more queries that were performed when using the portal but many of them are similar or no more complicated than the ones listed above. The data was used to ensure that updating and selecting was working while we were writing the code.

Testing

Many tests were performed throughout the construction and writing of the code. A lot of queries would be performed one by one to ensure that the data set that was being returned was the correct one. phpMyAdmin provided to be useful in visualizing what the data looked like as well as providing space to write queries to test. As more and more features and tables were being added, it was vital to ensure that relationship sets were actually linking the correct tables and that the foreign keys were correctly being referenced. This was done by creating tables one by one, performing queries, and ensuring that the data was correctly stored as we had intended it to. The next big test was ensuring that while inserting values into tables, all tables that were related would be updated as well. An example of this would be the insertion of all the mock car's data that we inputted. Inserting one car and matching that to a user inventory had multiple query statements and each one had to be exact or else the data would be inconsistent. Once again, query statements were performed one by one and looked up in phpMyAdmin to ensure that it looked exactly the way we wanted it to. Once everything was completed, a thorough testing of the system (going from signing up to creating a listing to purchasing),

proved that the system was robust enough to handle all the queries but still exposed many bugs that were eventually fixed.

Future Steps

The functionality that was implemented took awhile but overall, the implementation was extremely basic. There are a ton more things that can be done in order to create a much better, efficient system. For example, as of right now, users (both buyers and sellers) cannot modify their listing or cart. Ideally, they would be able to delete from their inventory (if they're a seller) or delete something from their cart (if they're a buyer). Furthermore, buyers and sellers have some differentiations but with the implementation of sales for sellers and buyers actually able to select preferences with regards to location, a distinction will become much clearer. Adding in a search bar, filtering system, or sorting capability would allow for both the seller and buyer to view the data in a more comprehensible way. One last capability that we envisioned would be necessary would be the ability to take and add pictures to listings.

Conclusion

Our project used HTML, PHP, and MySQL to implement a website for people to create an account, differentiate between buyer and seller, create listings, purchase cars, and update their profile. With more work, this website could become an actual page where people come and buy and sell used cars. Furthermore, if though the vision was only used cars, this website could easily be translated into a new car lot as well. There is still much work that can be done to improve the system but this project has served as a very solid foundation in terms of learning how to implement and incorporate a database management system into an application such as the one we created.