CMPSC 431W Project Proposal

Team: Big Leg Carry

Sui, Haojun

hzs5220@psu.edu

Zhang, Chenyu

dianachenyuzhang@gmail.com

Deng, Yuanpei

dengyuanpei@gmail.com

Wang, Hao

haowang5128@gmail.com

Chen, Shiqing

u0vv0u@gmail.com

March 3, 2016

Contents

List of Figures

List of Tables

1   Introduction

HelloWorld is a startup company that wants to pursue the opportunities of online business. We, team Big Leg Carry, have designed a database application for online Car shopping for HelloWorld. Users may log into our system by using registered user account or seller account. Registered users may easily search the kind of cars they want from search page, or they can fill out our quick survey to see what kind of cars suit their need. Users may compare different cars at the same time, if they are interested in multiple cars and have a hard time to choose the best one. Registered users may purchase a vehicle or place a bid on the vehicle. Once the transaction is confirmed, users can either choose to have vehicle delivered to them, or pick the vehicle themselves. Upon receiving the cars, users must finish the rest of the payment. Sellers can easily list the vehicles they want to sell from sellers' webpage. Both registered users and sellers can make rating on each other. We want such feature to maintain our friendly ecosystem. In order to fulfill these features, we define the following requirements.

2 Requirement Analysis

2.1 Sale Items

The items that we are offering for sale are automobiles. Items listed on our websites are stored in a database management system as an entity. Each item will have a primary key named the item_id which is unique to every item. As to offer better details to item listings as well as the sake of searching features. Several attributes will be present for the item entity. Some attribute will be visible to the customers as item specifics, and some will be only visible to those who have access to our database as listing specifics. The item specifics include Vehicle Make, Vehicle Condition, Vehicle Model, VIN (Vehicle Identification Number), Year, Mileage, Vehicle Title, Vehicle Type, Body Type, Options, Safety Features, Power Options, Sub Model, Fuel Type, Exterior Color, Interior Color, Drivetrain, For Sale By, Number of Cylinders, Engine Description, Transmission, Trim, and Warranty. On the other hand, the listing specifics indicators include Listing Status, Listing Type, Listing Duration, and Listing Start Time.

As most of these item specifics are very straightforward, some still need further clarification. Vehicle Title refers to the current status of a vehicle, and has two types: clean title and clear title. A clean title is usually used to refer to any car that passes inspection without having any serious physical issues. A clear title is usually used to describe a financial lien that has been placed on a car. For Sale By refers to the type of the seller, which can be either dealer or individual seller. For listing specifics, listing status refers to the status of the listings, and can be active, ended, unsold, sold, and removed. Listing Type can be either auction or buy-it-now. Listing Duration is the time period when the listing is active.

2.2   Categories

"Shop by Categories" is a necessary way to browse items as customers may not know what to search. The items available at HelloWorld are categorized using a predefined classification tree. Customers will first choose the body type of the vehicle. So the root of the tree will be labeled as "All Body Type", which we have the nodes of the root as SUV, Pickup Truck, Convertible, Sedan, Crossover, Coupe, Luxury Car, Wagons/Hatchback, Green Cars/Hybrids, Sports Cars, and Minivans/Vans. On the lower levels of the first tree, from top to the bottom, we have: Make, Year, Model, Sub Model, and Trim.

An item can be specified by a path through this classification tree. For example, we may categorize an item as:

1. SUV > Audi > 2016 > Q7 > Q7 eTron > Base

2. All Body Type > Fiat > 2011 > 500 > 500 Abarth

The depth of the tree varies but is no more than ten levels deep.

2.3   Suppliers

In our case, supplier is the car seller. The attributes of supplier (dealer) are dealer name, address, phone number, type, account number, and rating. The type attribute classifies supplier as car dealer and individual seller. The dname here is used as the primary ID and required to be unique. Phone numbers, rating, type, etc are attributes connected to each dealer. The rating is a weighted score calculated by buyer's (register user) rating on aspects like accuracy, price, choice, service, and feedback. Dealer will have an inventory. The relation is called owns and the entity is cars. The cars table is discussed in detail in the above sale items section. The dealer also has a relation named

locates and the entity is address. The address we used here is zip code. Address is an important attribute in car shopping. People would prefer a dealer near their location so that they can go to store for a test drive and get car service there after the purchase. The address would be used in the search to decrease the scope. The reason we choose make "address" as an entity instead of an attribute is that we can use the similar schema for registered users. Figure 1 shows the ER diagram for Suppliers.

Figure 1: ER diagram for Suppliers.
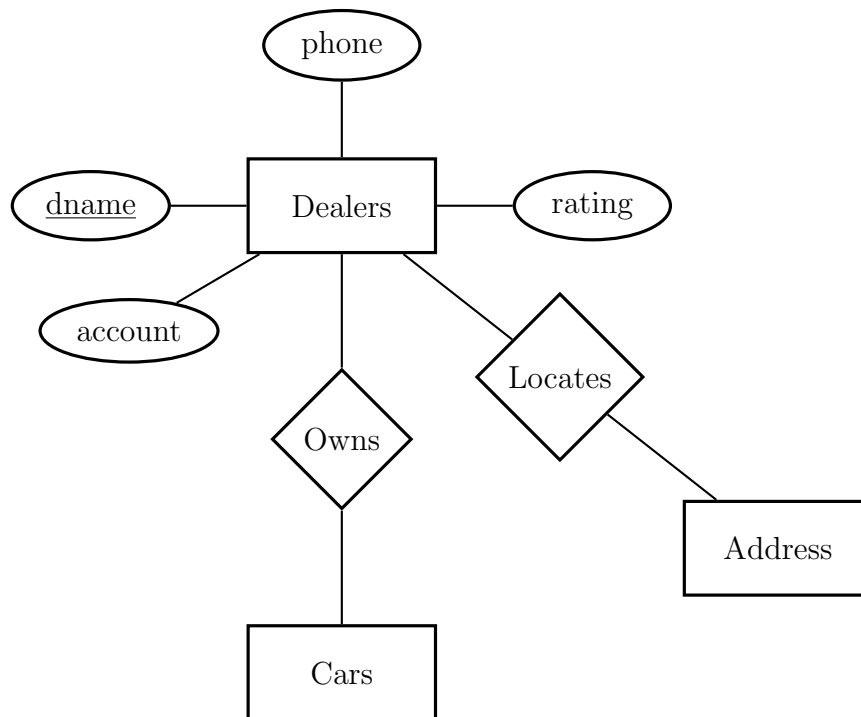
2.4   Registered Users

The registered users are the buyers who can buy or bid on an item (car). Buyer must be registered, and identified by a user name and authenticated with a password. When registering, the user would be asked the following information in order to register successfully. These information includes: email address, name, address, phone number and credit card info like type of card, card number, cvv and expiration date. Figure 2

shows how the interface of registering page looks like for Madison, Inc. (In real design the credit card information will be added.)

Figure 2: Interface of Registering Page for Madison, Inc.

In addition, after registering a user can complete his or her whole profile by adding other information like age, gender and annual income. These attributes can be NULL if the user chooses not to fill.

Register user also has a rating attribute. A new user has a rating of 0 to start with and recalculates his or her rating score after each successful business. Figure 3 shows the ER diagram for registered users.

Figure 3: ER diagram for Registered Users.
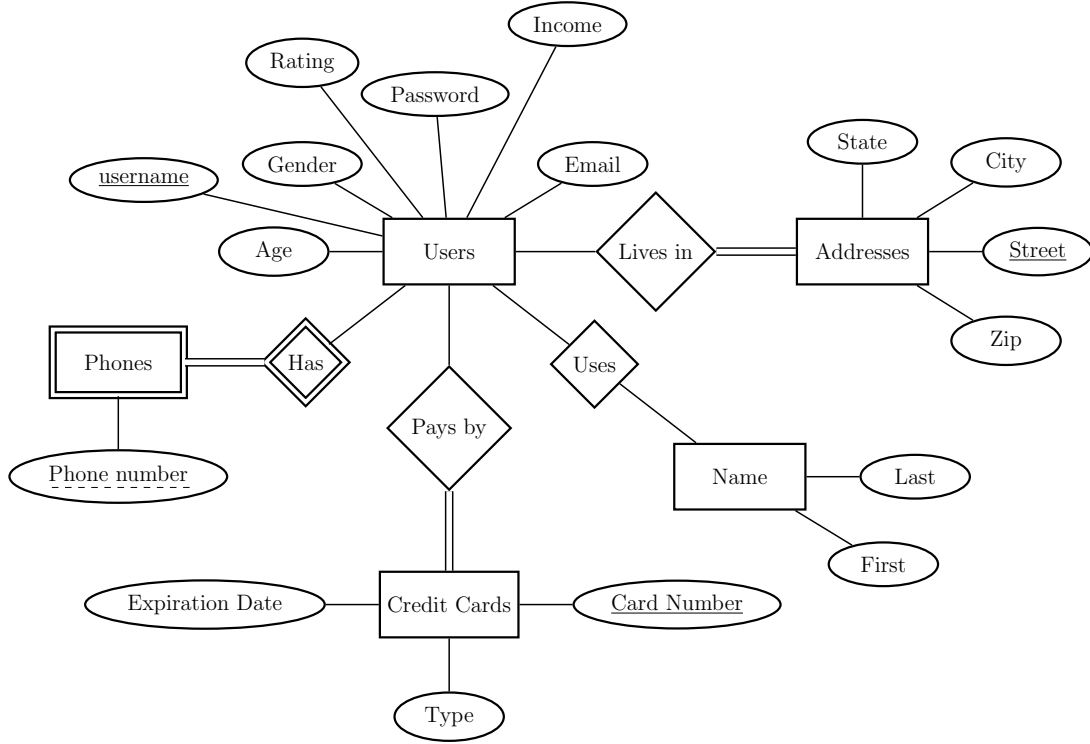
2.5   Rating

Each supplier and register user has a rating score to reflect their past business behaviors with other users. For suppliers who sell cars on the website, the rating is an importing factor to help customers, the register users, to compare service quality and make purchase decision. For register users who are customers or potential customers, the rating reflects a general feedback from the car sellers and their punctuality of payment. Mostly, the rating of supplier is more important compared to that of user but we would have ratings for both supplier and register user.

Each successful business would allow one rating. Supplier and register user would rate each other on the scale of 1-10. Suppliers would be rated on the aspects of accuracy, price, choice, service, and feedback.

1. Accuracy: If the information online the same as the real condition of the car.

2. Price: Price is probably one of the most important factors for used car shopping. Is the price provided by the dealer competitive compared to others'?

3. Choice: Does the dealer provide a lot of car choices?

4. Service: Is the dealer professional, nice, and informative? Does the dealer respond your call and emails in a timely manner? Does the dealer call you way too frequently and become bothering?

5. Feedback: The customer can add more feedback months or even years after the purchase. Is the car still working well after a year? For register user/buyer, the rating from supplier would be simpler. If the register users are new and did not purchase on the website before, they would start with a score of 0. It would be the case for most of the register users, especially at the beginning of website operation. The register user would be rated on feedback and payment punctuality.

6. Feedback: Is the register user a good and reasonable customer?

7. Payment: Does the customer pay the rest of money on time? Does customer finish the loan at the end as planned?

2.6   Browsing

Browsing is one of the most important feature of a website, especially for a car selling website. It has great impact on user experience for the website, a good browsing feature can give user easy access to the target items, however, a bad browsing feature will prevent user from accessing the target items and sometimes even browse the wrong items.

How it works

7

According to the input, using SQL to select proper table or a specific record. Users are able to browse the items by traversing the category tree. At each point, they are given a summary of all the items that appear in that category.

Design

A good browning feature should have the following attributes:

1. Easy to use (good design)

2. Accurate (well built database and proper SQL instructions)

3. Quick response (well built database and proper SQL instructions)

Browsing is usually done by a browsing bar. There are two types of browsing bar in our website: horizontal bar and vertical bar. Horizontal bar is on the top of the page and vertical bar (box) is on the right of the page.

Horizontal bar consists of 2 buttons and 3 drop-down menus (See in Figure 4):

Figure 4: Horizontal Bar Design.

On the main page, browsing bar is placed horizontally on top of the page which is a striking position. By doing this, it provides easy access for users to browse our inventory right after they enter the website.

New Car button and Used Car button: User can select New car, used car, or both to browse our inventory. User can combine buttons with drop-down menus to filter out undesired results. Results will be displayed after button selection.

Drop-down menus: For users who have a target car in mind. First, select make of the car and then select model will take them to the result selected from our database according to the given input. Or users can browse the car by type (sedan, coupe, SUV, other), and jump to the result page with cars of that type.

All inventory pages have a vertical browse bar with all the attributes according to previous input (new/used/type...) Figure 5 shows the design of the vertical bar.

Figure 5: Vertical Bar Design.

The reason that it is on vertical and on the right is that user are not disturbed by drop-down menus and focus more one the content.

Every change of the drop-down menu will immediately execute a SQL instruction and bring the result to the page.

2.7   Searching

Users are able to search the items by entering some keywords or conditions. As a search result, a list of items that satisfy the search criteria is returned to the users. Searching is another function to get quick access to the target item besides browsing. If a user has a specific car in mind, by using our search bar, he can filter out the non-matching records and gets the desired results directly from our database.

How it works

User types a keyword in the search bar and hit enter to search. Website breaks the keyword into several words. The website will search the first word in all the related tables in our database, then search the next word in the result from last word searching.

Simplified Example

Assume user search for Audi A5, we breaks the search words into keywords 'Audi', 'A5'. We first search for Audi. Figure 6 shows the results to users.

We then search for A5. Figure 7 shows the results to users.

Design

Figure 8 shows the design of the search bar.

It will be on every page, together with browsing bar so that user can perform a search no matter what page they are on.

Figure 6: Search for Audi.

Car Table					
Vid	Make	Model	Year	Price	New/used
100	BMW	M3	2015	60000	New
200	Audi	A5	2015	40000	New
300	Audi	A4	2013	25000	Used
400	Audi	A5	2014	35000	Used

Figure 7: Search for A5.

Vid	Make	Model	Year	Price	New/used
200	Audi	A5	2015	40000	New
300	Audi	A4	2013	25000	Used
400	Audi	A5	2014	35000	Used

Figure 8: Search Bar Design.

Keywords

2.8 Sale

When a buyer decide to purchase an item (car), the website will charge the buyer?s credit card the amount of the price of the item and the delivery fee. The website will hold it until the buyer confirmed that the car is delivered. Then the website will transfer the money to the seller. Then this transaction is closed.

2.9   Bidding

At HelloWorld, bidding is pretty easy and fair. When seller lists a vehicle on our website, he can choose auction as the listing type. Then he will set a starting price for the auction, a duration, and a reserve price as an optional feature. The starting price has to be greater than 0.99, and lower than the item's buy-it-now price if it has one. The duration can be 3 days, 5 days, and 7 days max. The reserve price which is invisible to buyers is the price threshold where buyers have to bid higher price than the reserve price in order to win the item. If no bid achieves the reserve price at the time when the listing ends, then the item isn't sold. Seller can't change an auction type listing to fixed price listing, but he may end an auction earlier than the projected end time. Seller can change a fixed price listing to an auction type at any time before the end of the listing period, and then the auction duration will be added to the listing upon a successful change from fixed price to auction. To maintain the fairness of the auction, bidders are anonymous to sellers, only the highest bid is visible to the seller.

On the buyer side, one can bid on any item at any time, and the number of bids is unlimited. There are two types of bids, one is a one-click-bid, and the other is a max-bid. A one-click-bid is the bid which is one bid increment higher than the current bidding price. The bid increment is determined by the range of current bidding price. Table 1 shows the current bid increment.

The max bid is the maximum price a buyer is willing to pay for the item. The max bid is required to be at least one bid increment higher than the current price. When a buyer placed a max bid on an item, the current bid will raise only one bid increment. Buyer can raise the max bid any time during the auction. All the one-click-bids are visible to all the buyers, but max bids are only visible to the buyer who placed it. Buyer can retract

Table 1: Current Price Bid Increment

Price Range($)	Bid Increment($)
Up to 99.99	5
100 to 999.99	25
1000 to 4999.99	50
5000 to 9999.99	100
10000 to 49999.99	200
50000 or more	500

his bid any time during the auction, and only if the bid is the highest bid at the moment, but leaving a record of retraction. After retraction, the highest bid placed by other buyer will be the current bid. When buyer A places a bid higher than the current bidding price, buyer B with that bid will be outbid, which means the buyer B will have to bid higher in order to win the item. If buyer B has a max bid, then the bid from buyer A may not be high enough to compete, then buyer A will be automatically outbid with a bid increment at the current bidding range. Buyer A will then place higher bids until the highest bid is lower than his max bid. The highest bid at the end of auction is called a winning bid, the buyer who placed the bid will be the winner of the auction. However, if the highest bid isn't higher than the reserve price, then the buyer isn't winning.

2.10   Order and Sale Reports

Every week, a report would be generated to summarize all the sale and website traffic information happened in the past week. Table 2 contains the order and report elements.

2.11   Delivery

After the buyer bought the vehicle from our website, two options of delivery will be offered. Buyer can drive to the dealer to pick up the vehicle, or he can use our insured and
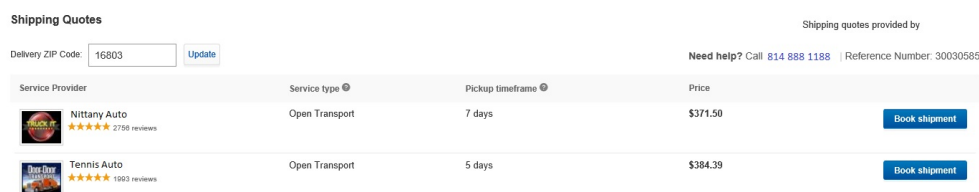
Table 2: Order and Report Elements

Section	Elements
Sale	Total number of business transaction Total number of car sold Total amount of sale figure(money)
Traffic	Total number of visits Pages viewed per visit Average time spent per visit
Figure and Comparison	Percentage of increment/decrement compared to last week The category with the largest change Visual representation of graphs and figures

convenient delivery services. If the buyer chooses to pick up the vehicle, he will complete the paper work and pay the vehicle price in full excluding the deposit when he arrives in the dealer?s. Buyer may have a test drive with the car, but he can't retract the purchasing agreement consented on our website. In the worst case, if the buyer persists that he can't purchase the car, the buyer will be charged a penalty no more than 5% of the final price.

The vehicle delivery services we provide on our website are offered by two partner: the Nittany Auto Delivery, and Tennis Auto Delivery. Both our partners have strongly skilled drivers with sales skills, and their offices are located all over the country. If a buyer choose to get the vehicle via one of the delivery partners, he can expect the vehicle at his door after 3-7 business days upon clear payments. The charges of the delivery will be determined on the distance between the two destination and the type of the vehicle. The optional insurance on the auto delivery will be offered at buyer's expense. At the time when the car is delivered to the buyer, buyer will complete the payments as well as paperwork with the trusted driver from our partners. Figure 9 shows shipping quotes from ebay.com.

Figure 9: Shipping quotes from ebay.com.

2.12 Smart Car Finder

For user who do not have a target car in mind or do not have knowledge about cars, car find will give them recommendations according to their needs.

How it works

User will take a short multiple choice survey. Car finder transfer the user-answers and transfer to proper keywords to browse the database and filter out the undesired results. After finishing the survey, what final results will show on the page.

Design
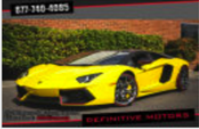
The questions in the survey are written in descriptive words instead of professional terms of a car. For example, survey will have expression like "big car", "large cargo space" instead of "SUV" since user may not know what is SUV. The survey will ask user in an easy way and get the following information from user: Price range, car type, mpg range, year of manufacture or other feature like backup cam, FWD/AWD, etc. Survey can be accessed by clicking "help me to find a car" button.

2.13 Item Comparison

Since this is car purchasing, user probably need compare several cars to determine which one to buy. So compare function is very important for this project. After the user searching for what kinds of car he wants to buy, there will be a list of cars that satisfied his

searching key word. Figure 10 shows how Cars.com implements the same kind of feature.

Figure 10: Car Comparison from Cars.com.

Then user can mark several cars as compare. On the top of website there will be a button "compare". By clicking this button the website will generated a table shows the data of the marked cars? entities.

3   Conceptual Design

Figure 11 shows the complete ER diagram design of our database application.

Figure 11: Complete ER diagram Design.
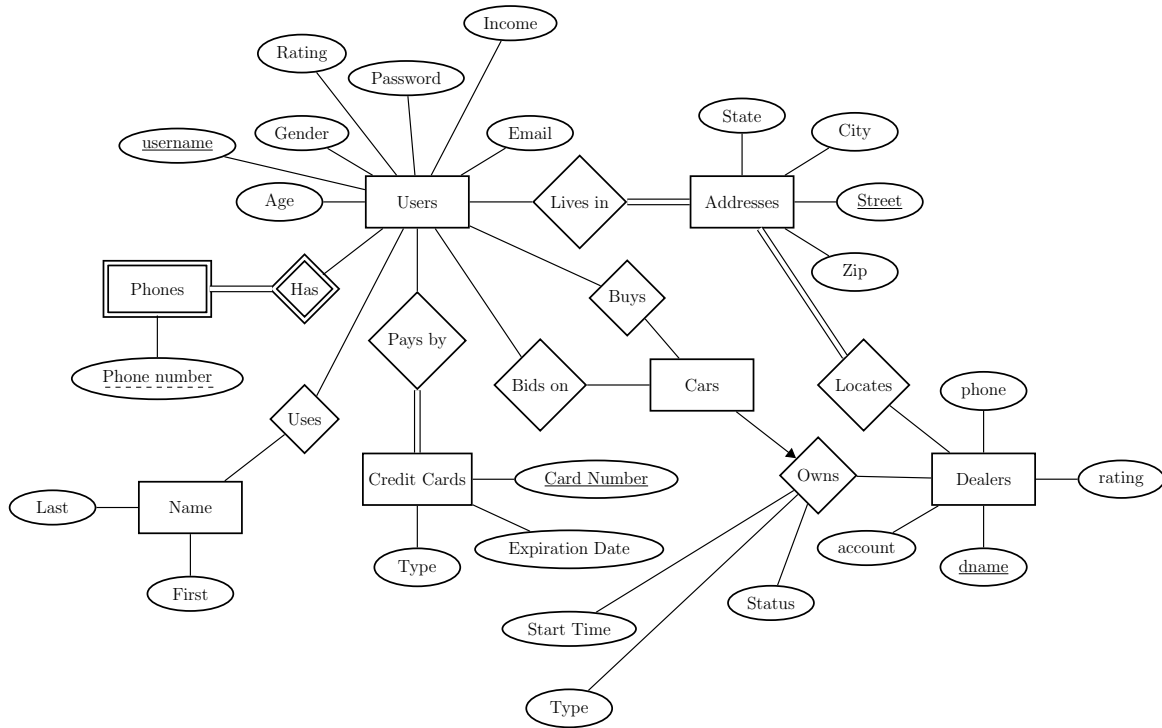
As the graphs shows, each user will be identified by their username, and each sellers will be identified by his *dname*. We define one weak entity set for users since we don't need to store users' phone number if they decide to deactivate their account. Cars can only be owned by a unique seller, which means no multiple sellers sell the same car. As credit card maybe used by other people, we will not make credit card as a weak entity set.

4 Data Schema

4.1 Original Schema

Users					
<u>username</u>	email	age	rating	password	income

Credit Cards			
<u>username</u>	card number	expiration date	type

User Addresses				
<u>username</u>	state	city	street	zip

Dealers			
<u>dname</u>	account	phone	rating

Owns				
<u>dname</u>	vin	start time	type	status

Phones	
<u>username</u>	phone number

Name		
<u>username</u>	last name	first name

Cars	
<u>vin</u>	attributes...[1]

Buys	
<u>username</u>	vin

Bids	
<u>username</u>	vin

Dealer Addresses				
<u>dname</u>	state	city	street	zip

4.2 Schema Refinement

Everything in green are the changes and updates, and everything in red are the objects to be removed.

User										
<u>user- -name</u>	email	dob	rating	pwd	income	phone1	phone2	last name	first name	address id

Instead of AGE, we have dob here so that we don't need to update user age every year. Addressid refers to the primary key in the Table Addresses to determine user's address. Table Phones is removed since we find it better as user attribute. Table Name is removed,

---

[1]year, mileage, make, model, body type, fuel, exterior color, interior color, drivetrain, transmission, engine, condition

and the last name and first name are added as user attributes.

Credit Cards			
username	card number	expiration date	type

Addresses				
address id	zip	street	city	state

We have combined the two tables of the User Addresses, and the Dealer Addresses into one table named Addresses. Addressed is the primary key of the Table. Zip can be determined by city and state. But city or state cannot be determined only by zip, example, 42223 can be Kentucky or Tennessee. And for city Gainesville, it can be in FL or VA (2 cities in different state named Gainesville).

Dealers				
dname	account	phone	rating	address id

Addressid refers to the primary key in the Table Addresses to determine dealer?s address.

Owns				
dname	vin	start time	type	status

Phones	
username	phone number

This table is removed; such entity has been added to table User as attribute.

Name		
username	last name	first name

This table is removed; such entity has been added to table User as attribute.

Cars	
vin	attributes...[2]

Buys			
username	vin	price	time

Price and time are added as attributes to the buying process.

Bids	
username	vin

Table Bidding replaces the table Bids.

---

[2]year, mileage, make, model, body type, fuel, exterior color, interior color, drivetrain, transmission, engine, condition

Dealer Addresses				
<u>dname</u>	state	city	street	zip

Table Dealer is removed, and it is merged into the Addresses table.

Bidding			
<u>vin</u>	price	time	<u>username</u>

Bidding table is created to reflect the bidding activities. The primary key is 'vin, username', since each user can only have one bidding value stored for each car.

Comments			
<u>username</u>	<u>dname</u>	comment	date

Comment table is added to meet the requirement of having 5 or more comment for a dealer. User can comment on a dealer only once but can comment on multiple dealers. User cannot comment on user. Dealers cannot comment on dealer or user.

4.3 Schema Normalization

Reducing to First Normal Form

To be in first normal form, A relation is in first normal form if and only if the domain

of each attribute contains only atomic(indivisible) values, and the value of each attribute contains only a single value from that domain.

User										
user--name	email	dob	rating	pwd	income	phone1	phone2	last name	first name	address id

The domain of each attribute contains only atomic(indivisible) values →check!

Value of each attribute contains only a single value from that domain →check!

Credit Cards			
username	card number	expiration date	type

The domain of each attribute contains only atomic(indivisible) values →check!

Value of each attribute contains only a single value from that domain →check!

Addresses			
address id	street	city	state

The domain of each attribute contains only atomic(indivisible) values →check!

Value of each attribute contains only a single value from that domain →check!

Dealers				
dname	account	phone	rating	address id

The domain of each attribute contains only atomic(indivisible) values →check!

Value of each attribute contains only a single value from that domain →check!

Owns				
<u>dname</u>	vin	start time	type	status

The domain of each attribute contains only atomic(indivisible) values →check!

Value of each attribute contains only a single value from that domain →check!

Cars	
<u>vin</u>	attributes...[3]

The domain of each attribute contains only atomic(indivisible) values →check!

Value of each attribute contains only a single value from that domain →check!

Buys			
<u>username</u>	<u>vin</u>	price	time

The domain of each attribute contains only atomic(indivisible) values →check!

Value of each attribute contains only a single value from that domain →check!

---

[3]year, mileage, make, model, body type, fuel, exterior color, interior color, drivetrain, transmission, engine, condition

Bidding			
<u>vin</u>	price	time	<u>username</u>

The domain of each attribute contains only atomic(indivisible) values →check!

Value of each attribute contains only a single value from that domain →check!

Comments			
<u>username</u>	<u>dname</u>	comment	date

The domain of each attribute contains only atomic(indivisible) values →check!

Value of each attribute contains only a single value from that domain →check!

Reducing to Second Normal Form

To be in second normal form, A table is in second normal form if it is in first normal form and every non-prime attribute of the table is dependent on the whole of every candidate key.

User										
<u>user--name</u>	email	dob	rating	pwd	income	phone1	phone2	last name	first name	address id

In first normal form →check!

Non-prime attribute of the table is dependent on the whole of candidate key→check!

(Username → email, dob, rating, pwd, income, phone, lastname, firstname, address id)

Credit Cards			
<u>username</u>	card number	expiration date	type

In first normal form →check!

Non-prime attribute of the table is dependent on the whole of candidate key→check!

(Username → card number, expiration date, type)

Addresses			
<u>address id</u>	street	city	state

In first normal form →check!

Non-prime attribute of the table is dependent on the whole of candidate key→check!

(address id → street, city, state)

Dealers				
<u>dname</u>	account	phone	rating	address id

In first normal form →check!

Non-prime attribute of the table is dependent on the whole of candidate key→check!

(dname → account, phone, rating, address id)

Owns				
<u>dname</u>	vin	start time	type	status

In first normal form $\rightarrow$check!

Non-prime attribute of the table is dependent on the <u>whole of candidate key</u>$\rightarrow$Uncheck!

We put startTime, type, status into cars table, as attributes. Now, Owns table changes into:

Owns	
<u>vin</u>	dname

Non-prime attribute of the table is dependent on the whole of candidate key$\rightarrow$check!

(vin → dname)

Cars	
<u>vin</u>	attributes...[4]

---

[4]year, mileage, make, model, body type, fuel, exterior color, interior color, drivetrain, transmission, engine, condition

In first normal form $\rightarrow$check!

Non-prime attribute of the table is dependent on the whole of candidate key$\rightarrow$check!

(vin $\rightarrow$ attributes)

Buys			
<u>username</u>	<u>vin</u>	price	time

In first normal form $\rightarrow$check!

Non-prime attribute of the table is dependent on the whole of candidate key$\rightarrow$check!

(username $\rightarrow$ vin, price, time)

Bidding			
<u>vin</u>	price	time	<u>username</u>

In first normal form $\rightarrow$check!

Non-prime attribute of the table is dependent on the whole of candidate key$\rightarrow$check!

(vin, username $\rightarrow$ price, time)

Comments			
<u>username</u>	<u>dname</u>	comment	date

In first normal form $\rightarrow$ check!

Non-prime attribute of the table is dependent on the whole of candidate key $\rightarrow$ check!

(username, dname $\rightarrow$ comment, date)

Reducing to Third Normal Form

To be in third normal form, the entity is in second normal form, and all the attributes in a table are determined only by the candidate keys of that table and not by any non-prime attributes.

User										
user--name	email	dob	rating	pwd	income	phone1	phone2	last name	first name	address id

The entity is in second normal form $\rightarrow$ check!

All the attributes in a table are determined only by the candidate keys of that table and not by any non-prime attributes. $\rightarrow$ Uncheck!

Since email is unique, it can be used to determine the following attributes just like username. Hence we decompose the User table into 2 table.

User									
user--name	dob	rating	pwd	income	phone1	phone2	last name	first name	address id

29

The entity is in second normal form   →check!

All the attributes in a table are determined only by the candidate keys of that table and not by any non-prime attributes.   →check!

Register	
<u>username</u>	email

The entity is in second normal form   →check!

All the attributes in a table are determined only by the candidate keys of that table and not by any non-prime attributes.   →check!

(username → email)

Credit Cards			
<u>username</u>	card number	expiration date	type

The entity is in second normal form   →check!

All the attributes in a table are determined only by the candidate keys of that table and not by any non-prime attributes.   →Uncheck!

Since card Number can determine expiration date and type, and card number is a non-prime attributes. It is not in third normal form. We decompose it into two tables:

Credit Cards		
<u>card number</u>	expiration date	type

The entity is in second normal form   →check!

All the attributes in a table are determined only by the candidate keys of that table and not by any non-prime attributes.   →check!

Payment	
<u>username</u>	card number

The entity is in second normal form   →check!

All the attributes in a table are determined only by the candidate keys of that table and not by any non-prime attributes.   →check!

Addresses			
<u>address id</u>	street	city	state

The entity is in second normal form   →check!

All the attributes in a table are determined only by the candidate keys of that table and not by any non-prime attributes.   →check!

Dealers				
<u>dname</u>	account	phone	rating	address id

The entity is in second normal form   →check!

All the attributes in a table are determined only by the candidate keys of that table and not by any non-prime attributes.   →check!

Owns				
<u>dname</u>	vin	start time	type	status

The entity is in second normal form   →check!

All the attributes in a table are determined only by the candidate keys of that table and not by any non-prime attributes.   →check!

Cars	
<u>vin</u>	attributes...[5]

The entity is in second normal form   →check!

All the attributes in a table are determined only by the candidate keys of that table

---

[5]year, mileage, make, model, body type, fuel, exterior color, interior color, drivetrain, transmission, engine, condition

and not by any non-prime attributes. →check!

Buys			
<u>username</u>	<u>vin</u>	price	time

The entity is in second normal form →check!

All the attributes in a table are determined only by the candidate keys of that table and not by any non-prime attributes. →check!

Bidding			
<u>vin</u>	price	time	<u>username</u>

The entity is in second normal form →check!

All the attributes in a table are determined only by the candidate keys of that table and not by any non-prime attributes. →check!

Comments			
<u>username</u>	<u>dname</u>	comment	date

The entity is in second normal form →check!

All the attributes in a table are determined only by the candidate keys of that table

and not by any non-prime attributes. →check!

5 SQL Statements

```
CREATE TABLE Users (username CHAR (20),

               AddressID CHAR (20),

               Password CHAR (20),

               Dob CHAR (8),

               Income INTEGER,

               Age INTEGER,

               Phone1 CHAR (10),

               Phone2 CHAR (10),

               Gender CHAR (6),

               Rating INTEGER,

               Last_name CHAR (20),

               First_name CHAR (20),

               PRIMARY KEY (username)

               FOREIGN KEY (AddressID) REFERENCE Address (AddressID));


CREATE TABLE Register (username CHAR (20),

               Email CHAR (50),

               UNIQUE (Email),

               FOREIGN KEY (username) REFERENCE User (username));


CREATE TABLE Credit_Card (Card_No CHAR (12),
```

```
                    Exp_date TIMESTAMP,

                    Type CHAR (20),

                    PRIMARY KEY (Card_No));



CREATE TABLE Payment (username CHAR (20),

                    Card_No CHAR (12),

                    FOREIGN KEY (username) REFERENCE Users (username),

                    FOREIGN KEY (Card_No) REFERENCE Credit_Card (Card_No),

                    ON DELETE CASCADE);



CREATE TABLE Address (AddressID CHAR (20),

                    State CHAR (20),

                    City CHAR (20),

                    Street CHAR (100),

                    PRIMARY KEY (AddressID));



CREATE TABLE Dealers (dname CHAR (50),

                    Account CHAR (20),

                    Rating INTEGER,

                    Phone_No CHAR(10),

                    AddressID CHAR (20),

                    PRIMARY KEY (dname)

                    FOREIGN KEY (AddressID) REFERENCE Address (AddressID));
```

```
CREATE TABLE Owns (VIN CHAR (17),

              dname CHAR (50),

              FOREIGN KEY (VIN) REFERENCE Cars (VIN),

              FOREIGN KEY (dname) REFERENCE Dealers (dname),

              ON DELETE CASCADE);


CREATE TABLE Buys (username CHAR (20),

              VIN CHAR (17),

              Price REAL,

              Time TIMESTAMP,

              FOREIGN KEY (username) REFERENCE Users (username),

              FOREIGN KEY (VIN) REFERENCE Cars (VIN),

              ON DELETE CASCADE);


CREATE TABLE Bidding (VIN CHAR (17),

                 Price REAL,

                 Time TIMESTAMP,

                 username CHAR (20),

                 PRIMARY KEY (username,VIN),

                 FOREIGN KEY (username) REFERENCE Users (username),

                 FOREIGN KEY (VIN) REFERENCE Cars (VIN),

                 ON DELETE CASCADE);


CREATE TABLE Cars (VIN CHAR (17),
```

```
                Year CHAR (4),

                Mileage REAL,

                Make CHAR (20),

                Model CHAR (20),

                Body_type CHAR (20),

                Fuel CHAR (15),

                Exterior_color CHAR (10),

                Interior_color CHAR (10),

                Drivetrain CHAR (3),

                Transmission CHAR (20),

                Engine CHAR (20),

                Condition CHAR (4),

                PRIMARY KEY (VIN),

                FOREIGN KEY (Current_bid) REFERENCE Bidding (BidID));


CREATE TABLE Comment (username CHAR (20),

                dname CHAR (50),

                Comment CHAR (1000),

                Date TIMESTAMP,

                PRIMARY (username, dname),

                FOREIGN KEY (username) REFERENCE Users (username),

                FOREIGN KEY (dname) REFERENCE Dealers (dname),

                ON DELETE CASCADE);
```

6   Technology Survey

For front end, we want to choose between Relational SQL database (MySQL) and NoSQL database (MongoDB). When compared to relational databases, NoSQL databases are more scalable. They support large volumes of rapidly changing data. Also, NoSQL databases are built to allow the insertion of data without a predefined schema. This saves the development time from migrating databases due to changes in schema. Since during the design phase, we changed a lot in our schema, we decided to use NoSQL for our database design.

For back end, we want to use something that's simple to program and also natively support NoSQL databases, such as MongoDB. PHP is now a widespread, well known "mainstream" technology, and also quite cheap to host and easily deployable. Also, PHP drops right into HTML code. Due to the program simplicity, we choose PHP as our back end language.

7   Populate Data for Database

We get 100 cars' data, including 50 items to be auctioned and 50 direct sale items to be sold directly from the Edmunds.com API.

We get data by calling, for example:

https://api.edmunds.com/api/vehicle/v2/makes?fmt=json&api_key=

{ cfj4zt9xwecz4wzhv3zr8yev }

Figure 12 shows the response of an example API call.

We get the user data from the Fake Name API by calling:

https://v5.fakenameapi.com/generate

Figure 12: Response of an Example API Call

```
    }]
}, {
    "id": "Audi_A7",
    "name": "A7",
    "niceName": "a7",
    "years": [{
        "id": 200423029,
        "year": 2013
    }]
}, {
    "id": "Audi_A8",
    "name": "A8",
    "niceName": "a8",
    "years": [{
        "id": 200421092,
        "year": 2013
    }]
}, {
    "id": "Audi_Q5",
    "name": "Q5",
    "niceName": "q5",
    "years": [{
        "id": 200437177,
        "year": 2013
    }]
}, {
    "id": "Audi_Q7",
    "name": "Q7",
    "niceName": "q7",
    "years": [{
        "id": 200423236,
        "year": 2013
    }]
```

Figure 13 shows the sample response.

Figure 13: Response of Sample Response

```
< Content-Type: application/json
< Connection: keep-alive
{
    "Number": 1,
    "Gender": "male",
    "NameSet": "American",
    "Title": "Mr.",
    "GivenName": "David",
    "MiddleInitial": "B",
    "Surname": "Banks",
    "StreetAddress": "2021 Hershell Hollow Road",
    "City": "Anaheim",
    "State": "CA",
    "StateFull": "California",
    "ZipCode": "92801",
    "Country": "US",
    "CountryFull": "United States",
```

Then we further select the attributes that we need and store them in local files.

Figure 14 shows the stored attributes. Figure 15 shows the stored user information.

Our category tree starts with car as root. The root has three children: New, Used, Certified Pre-Owned. Then each child is divided by Make, like Mazda, Audi, etc.; and then further divided by Mode, like Mazda3, Mazda5, Mazda6, etc. Figure 16 shows the category tree.

Figure 14: Stored Attributes

Auction						
Car Table						
vin	attributes.....					
vin	model_year	make_bough	model_boug	trim_bought	bodytype_b	transmission
WVWBA7AJ2CW002081	2013	Toyota	Highlander	SE	4 Door SUV	5-Speed Aut
9BWDH61J454099898	2014	Acura	ILX	2	4 Door Seda	5-Speed Aut
1FUPCSEB2XL951233	2014	Volkswagen	Tiguan	SE	4 Door SUV	6-Speed Aut
KL4CJCSBXDB107362	2014	Mazda	CX-5	Sport	4 Door SUV	6-Speed Aut
1FBNE31P86HA17986	2013	Fiat	500	Pop	2 Door Hatchback	
1GTEK14T8YE451833	2014	RAM	1500	SLT	4 Door Crew	8-Speed Aut
YV1LW5245W2554862	2014	Acura	RDX	Technology	4 Door SUV	6-Speed Aut
JKAVFED167B593117	2014	Honda	CR-V	EX	4 Door SUV	5-Speed Aut
SAJWA71B78SH74184	2014	Mazda	CX-5	Grand Tourir	4 Door SUV	6-Speed Aut

Figure 15: Stored User Information

User Database								
usernam	email	dob	rating	pwd	income	Phone1	Phone2	last nam
leom_stol	leom_stol@webmine.com	2/6/1974	3	yjyja5uLY{a-	650073	(605) 359-6381		Leoma	S
ryuicamen	ryuicamen@info.com	11/3/1973	5	UZu#Y]uDY	50878	(401) 709-8332		Ryuichi	A
ir_sto	ir_sto@bing.com	2/3/1954	4	uXe#uSe	77923	(947) 274-9163		Iratze	S
je_sokol	je_sokol@mail.com	1/17/1966	4	e^aRu5yZ	64440	(605) 719-2271		Jesal	
ryuimader	ryuimader@yahoo.ca	3/28/1988	1	ABeRY$U+	84492	(702) 783-7221		Ryuichi	
terryas	terryas@myspace.com	10/8/1950	3	eu9u+e2A9a	531729	(228) 366-6180		Terry	A
halidisan	halidisan@msn.com	5/24/1954	5	uvA`u-EdYrA	78840	(402) 758-4856		Hali	[
puvanderb	puvanderb@freespace.com	10/17/1958	4	A8u3A_Y/At	97157	(531) 032-5294		Pundarik	\
mace-rey	mace-rey@freewebmail.con	10/19/1980	4	UVudA?U5u	70933	(402) 040-1356		Mace	F
alde_ashcraf	alde_ashcraft@webmail.con	9/26/1956	5	UmU/u?uzA	351816	(601) 569-9464		Aldercy	A
nanttan	nant-tan@booksmart.com	7/3/1953	1	?YVSamUa	95335	(641) 848-3212		Nantale	T
adrianaad	adriana_ad@webmine.com	3/28/1954	1	YGU}EDYIe	99097	(712) 156-8358		Adriana	A
artana	ar-tana@infoseller.com	3/28/1980	3	ZAne3A	77633	(904) 456-1691		Arlen	T
nanjea	nan_jea@yoohoo.com	3/8/1973	3	u?ausUE@	80979	(443) 349-8551		Nantale	
sezem	se-zem@freespace.com	9/1/1986	3	eRuzYmu	126397	(712) 276-7024		Seldon	

Figure 16: Category Tree