John Haney

COSC 640 – Database Systems 1

Frostburg State University

# Car Rental Database System

## Project Description

The goal of this project is to design, develop, and implement a normalized database for a peer-to-peer car rental organization. This will function like an Airbnb model where people will be able to list their vehicles for rental to potential customers. Vehicle owners would be able to list details of their vehicle have it available for certain dates. Customers would then be able to find vehicles in their area that they could choose to rent for a given time. This would mean that when a customer orders a rental from a client that the client should receive an order and be able to either accept or reject rental. This system should be able to store the details of the vehicles that are listed as well. This way a potential customer can browse the through and find the vehicle they need.

Another customer facing feature of the system should be a way to file accident reports. This way the organization can keep a record of any disputes and ensure that they can properly document and adjudicate issues.

The system will also store data that would be business facing and not accessible to the customers. A record of employees should be kept. This would allow the employees of the organization to add, remove, update, and retrieve information on anyone that works for them. Additionally, information regarding the different office locations should be tracked by the system.

## Online Offices

The system should be able to store data about the organizations offices. This data should include attributes such as:

**Office: Attributes**

* Office ID (auto generated by the RDBMS)
* The name of the office
* Phone number
* Fax number
* Email
* Cost of rent for the location
* Number of the employees
* Country
* State
* Zip code
* City
* Street address

The original address attribute would be split into several different attributes. This will be done to prevent the column from containing multiple pieces of data. For the same reason we will turn the “dates open” attribute into a weak entity. This will end up being an extension of the office entity since there can be no date open without an office. We will also remove the attribute “office manager name” to instead reference their ID in a sperate junction table. That way we will make sure that we reference the correct manager incase multiple managers share the same name while properly organizing the data.

## Manages office

Manages office will be track the relationship between the manager (employee) and the office that they oversee. It will also keep track of the date that this particular manager started at this office.

**Manages office: Attributes**

* Employee ID
* Office ID
* Data Started

## Employee

The organization will also have many different employees that it will need to keep track of. These employees will have many basic personal details and work-related details that will be stored. These attributes would be:

**Employee: Attributes**

* Employee ID
* First Name
* Last Name
* Middle Name
* SSN
* Annual salary
* Tax deduction
* Birth date
* Marital status
* Sex
* Name of spouse
* Date started at this office
* Date started at this company
* Highest degree earned
* Date graduated
* Number of dependents
* Home phone
* Cell phone
* Street address
* Zip code
* City
* Country
* State

Like in the other tables that need to track a precise address, we will split it into separate attribute columns. We will also split the “marital status and age” and “Home phone number and cell phone number” into two separate categories since they are all separate pieces of information. Similarly, we will change and split “Last Degree & Date” to “highest degree earned” and “date graduated” to avoid any ambiguity. We will also break off “Name of Employee Manager” into its own relationship table. Similarly, we will break off and remove “Office ID” here and convert the data into its own relationship table.

We will change the last two attributes of the employee from “Number of Years an Employee Work at this office” and “Number of Years an Employee Work for this Company” to “date started at this office” and “Date started at this company” accordingly. This will be done since the system will be able to calculate the number of years from the start date. It will also help to prevent future errors where each employee would need to have their data updated whenever the next year occurs. Finally, we will remove the “List of Certificates & Dates” from this table. This will become its own table (weak entity) and will make organization of the employee data easier, more efficient, and less error prone.

## Manages Employee

This relationship table will help to keep track of which employees are managers and of which employees they oversee. This is where we will also keep track of the date when they became managers for the company. The Attributes we are interested in are as follows:

**Manages Employee: Attributes**

* Manager ID (Employee ID)
* Employee ID
* Date started as manager

## Works at Office

This is another relationship table between office and employee. The data we will record here will help to keep track of which employee works where. We will also store the date that when a particular employee started at a specific location. The attributes the table will record are:

**Works at Office: Attributes**

* Employee ID
* Office ID
* Date Started at Office

## Certificate

Since employees will be able to have multiple certificates that the system will need to keep track of, we have taken them from the employee table and made them into their own table. Here we will be able to store multiple certificates and link them to the employee by referring to their ID. The attributes we will keep track of are:

**Certificate: Attributes**

* Employee ID
* Certificate
* Date Earned

## Car Owner (Client)

The Clients will be the users who wish to put their vehicles up for rental. They will be the ones that provide the products to the customers. The system will therefore need to store personal data about the client along with business details as well. The attributes that we will store are as follows:

**Car Owner (Client): Attributes**

* Client ID
* SSN
* First name
* Last name
* Middle name
* Birth date
* Cell phone
* Address (Location ID)
* Email
* Credit score number
* Zip code
* City
* State
* Country

There are several changes that we will suggest here. The first is that we will change the “M.I.” (Middle Initial) to be Middle Name. This will be done to avoid and ambiguity and to keep the data consistent with other user entities. One restriction is that if the employee is the one attempting to register as a client, they will need another employee to sign off. This could be handled with back-end programming logic and a single query to the database and with company policies as well. This is to prevent employees from self-generating their own credit score. We will also remove the “Home phone” attribute as it is not necessary to be able to rent the cars and may cause confusion for the end user. We will also remove the street address completely but separate the city, state, country, and zip code attributes. This will allow the system to be able to let customers search for vehicles based off of the client’s general area. We will also remove the credit card data from the table entirely. Payment information will not need to be stored in the system so we will not worry about it.

## Checks Client

This relationship table will be used to track and organize data regarding which employee checked the client. We will also save the date that the client was checked.

**Checks Client: Attributes**

* Employee ID
* Client ID
* Date Checked

## Customer (Car Renter)

The customer or car owner will be the other primary users of the system. The system will need to be able to store personal details about the customers as it relates to the act of renting a vehicle. This means that along with personal details we will need to store some additional information about the eligibility of the customer. The attributes the system will store about the customer are as follows:

**Customer (Car Renter): Attributes**

* Customer ID
* First name
* Last name
* Middle name
* Birth date
* Cell phone
* Email
* State the driver’s license was issued
* Driver’s license number

As before in the client, the first change we will make will be to “M.I.” to Middle name.

This will be done to keep basic data consistent for users and to help with ambiguity. Another change we will make is to remove all the location data from the table. We will not need to store any of the customers location information since they will just be able to search for a vehicle by looking up a location. Finally, like we did in the client table, we will remove the credit card attributes.

## Days open

This table is the result of splitting off the “days open” attribute from the office table. The table here will only contain two pieces of information. The first will be a reference to the office ID and the other will be a say that it is open. This means each office will be listed as many times as there are days it is open (Ex. Office 1 is open Monday – Friday, office 1 will be listed five times).

**Days open: Attributes**

* Office ID
* Day open

## Vehicle (Car)

It goes without saying that the vehicles are the main product that the organization will be dealing with. The system will therefore need to keep track of as many descriptive details on the vehicle as possible. The attributes we will keep track of are as follows:

**Vehicles (Car): Attributes**

* Car ID (VIN)
* License plate number
* State registered
* Year registered
* Current mileage
* Class (Compact, SUV, Van, etc.)
* Number of doors
* Manufacturer
* Car model
* Color
* Model Year
* Daily price
* Miles Allotted
* Additional cost per mile
* Weekly price
* Monthly price
* Car description
* Listed (y/n)

The first change that we made is to separate the “Car Info…” into “License plate number”, “state registered”, and “year registered”. Another small modification that we made was to rename “features” to “number of doors”. The original attribute was ambiguous and seemed to be looking to store the number of doors. If there are any luxury features that a client wishes to list to the customer, they would be able to describe them in the “Car description” attribute. Another small clarity change was to rename “year” to “model year”. The attribute “Miles included” was renamed to “Miles allotted” for clarity. The “Weekly discount” and the “Monthly discount” were both changed to “Weekly price” and “Monthly price” accordingly. This is because calculating the price over time will be simpler this way where we will add the prices together based on time rented. The final change is to add a listed attribute. This will allow the client to store the care in the system and list or un-list it as they choose.

## Lists Vehicle

The lists vehicle table will store the relational data between a client and a vehicle. It will also keep track of the date the client listed it.  
  
**List Vehicle: Attributes**

* Client ID
* Vehicle ID
* Date listed

## Accidents

In the business of renting vehicles there is one inevitable reality that we will have to deal with. Accidents will occur and it will be out jobs to document them in as much detail necessary. This will allow us to mediate the process of dispute and settlement when a customer is involved in an accident. The attributes we will look to document in an accident report are as follows:

**Accidents: Attributes**

* Accident ID
* Customer ID
* Vehicle ID
* Date of accident
* Time of accident
* Description of damage
* Cost of damage
* Reporting officer first name
* Reporting officer last name
* Summary of police report
* Country
* City
* State
* Street address
* Zip code

There are several changes to the accident entity. The first is to change the “Customer Info” to “Customer ID”. This is because we can retrieve all the information about the customer from their ID rather than save duplicate data in the accident table. For the same reasons, we have changed “Car Info” to “Vehicle ID”. We have also split and renamed “Date & Time” to “Date of accident” and “Time of accident”. This is of course to prevent unrelated data from being saved together in a single column and to help with ambiguity. The address of accident location will be broken into its most atomic values possible. We have also changed the “Extent of Damage” to “Description of damage” to clarify what should be entered here. Finally, the “Police Report (y/n)” we have removed and replaced with “Reporting officer first name” and “Reporting officer last name”. This is because the presence or absence of data in any of the police report attributes will allow us to know of the existence of one. Additionally, if the accident is large enough to warrant a police report, it would be useful to know the name of the reporting officer.

## Transaction

One of the tables that was missing from the initial proposal is a transaction section. This would be where the system would store transactional data about the rentals. This is essential for our organization and users all may pay and receive what is needed. Transaction s will be made up by the customers. From there an order will be sent to the vehicle owner for their approval. The attributes the system will store related to order will be as follows:

**Transaction: Attributes**

* Transaction ID
* Customer ID
* Vehicle ID
* Payment ID
* Request start date
* Request end date
* Order price
* Client approval

Transaction s will be kept track of by an identifying order ID. The inquiring customer will be referenced here by their customer ID number. The vehicle in question will also be referenced here by the vehicle ID. The payment ID will also be referenced. The dates the customer wishes to rent the vehicle will be stored which will allow us to calculate the price of the vehicle. We will keep track of the approval status of the order. The client will be able to review the details of the order and submit an approval.

## Reviews (feedback)

Since one of the features that the project owner wishes to have implemented into their system is a way for clients and customers to leave feedback for each other there will need to be a way to create and keep track of these reviews. Reviews should only be able to be made by two users that have conducted business and once the vehicle has been returned. The attributes that we are interested in storing are:

**Reviews (feedback): Attributes**

* Review ID
* Customer ID
* Client ID
* Vehicle ID
* Order ID
* Review post date
* Review Body
* Rating (1-10)
* Accident ID

The review will need to reference several other tables. Both the customer and client ID’s should be referenced and recorded. This is so there will be any easy way to find all the reviews attributed to any user. For the same reason reviews should reference and record the vehicle ID. The order ID will also be referenced. This is so that the users can see to which order and instance of a rental that the review is about. A post date should be kept track of as well. The core of the review will be the “Review body”, where users will be able to write about their experiences regarding their dealings with each other. Additionally, there should be a rating system so that users can have a quantifiable metric of satisfaction. Lastly, there should be a reference to an Accident ID if there is one to be referenced.

## Payments

A crucial oversight from the original proposal is a way for the system to keep track of payments. There should be a separate section from orders to record all the information regarding payment amounts, status, etc. The attributes that we are interested in recording are:

**Payment: Attributes**

* Payment ID
* Credit card number
* Card expiration date
* Subtotal
* Fees
* Taxes
* Total
* Status (processed/rejected/hold)
* Payment date

A subtotal should be calculated, this would be the same as the price of the transaction price. The fee for our services would be calculated and added. Taxes will also be calculated an added. This will allow for an easy calculation of the total that will be charged. A status will also be recorded here in case a payment does not go through. Finally, a date stamp of when the payment was made will be recorded.

## Receives Payment

There will need to be a relationship between the payment and client so that the system may record that a payment was received. This table will store the following attributes:

**Receives Payment: Attributes**

* Receives payment ID
* Payment ID
* Client ID
* Date payment was received