Project Requirements and Solution (Deliverable #1)

For this project, our group was tasked with creating a working database that accurately reflected the logical structure of a car rental business.

The car rental business has several locations with different addresses and several classifications of car. For the business locations, we needed to keep track of the store ID, street, number, city, state, and postal code. For the car, we kept track of the VIN, year, color, make, license plate, classification, and model. There were 4 different classes of car to keep track of. A car could be compact, sub compact, sedan, or luxury.

For each rental transaction, we needed to keep track of the transaction ID, pickup location, drop-off location, beginning odometer, ending odometer, end tank volume, rental date, and rental policy.

For each person we needed to keep track of their license ID, first name, last name, street, city, state, postal code, phone #1, and phone #2. Additionally, we needed to keep track of different attributes based upon whether the person was an employee, a customer, or both. If the person was a customer, we would also need to keep track of their membership level, first purchase date, and customer type. If the person was an employee, we would keep track of their position type, position title, benefits, and start date. If the person was both an employee and a customer, we would keep track of all of these attributes.

Initially, we planned on creating a person class, customer class, and employee class, where the customer and employee class would inherit the properties from the person class. However, we skipped the chapter on inheritance during class, so instead, we combined all the attributes into one person class. The attributes that are unique to employee would be null if the person were not an employee. The attributes for customer would be null if the person were not a customer. This way, we would be able to create SQL queries to differentiate between person and customer by selecting tuples that did not contain null values in certain columns.

The person class connected in a one-to-many relationship with the rental transaction class. This way a person could have many rental transactions. Additionally, the rental transaction class connected to the car class in a many-to-one relationship. That way, a car could be rented out many times. The store location class connected to the car class in a one-to-many relationship so that a single location could have many different cars. The store location class also connected to the person class in a one-to-many relationship so that there could be many employees at a particular location. The car classification class connected to the car in a one-to-many relationship, so that one car classification could have many different cars. Finally, the promotion class connected to the car classification class in a one-to-many relationship, so that a single promotion could apply to several different car classifications.