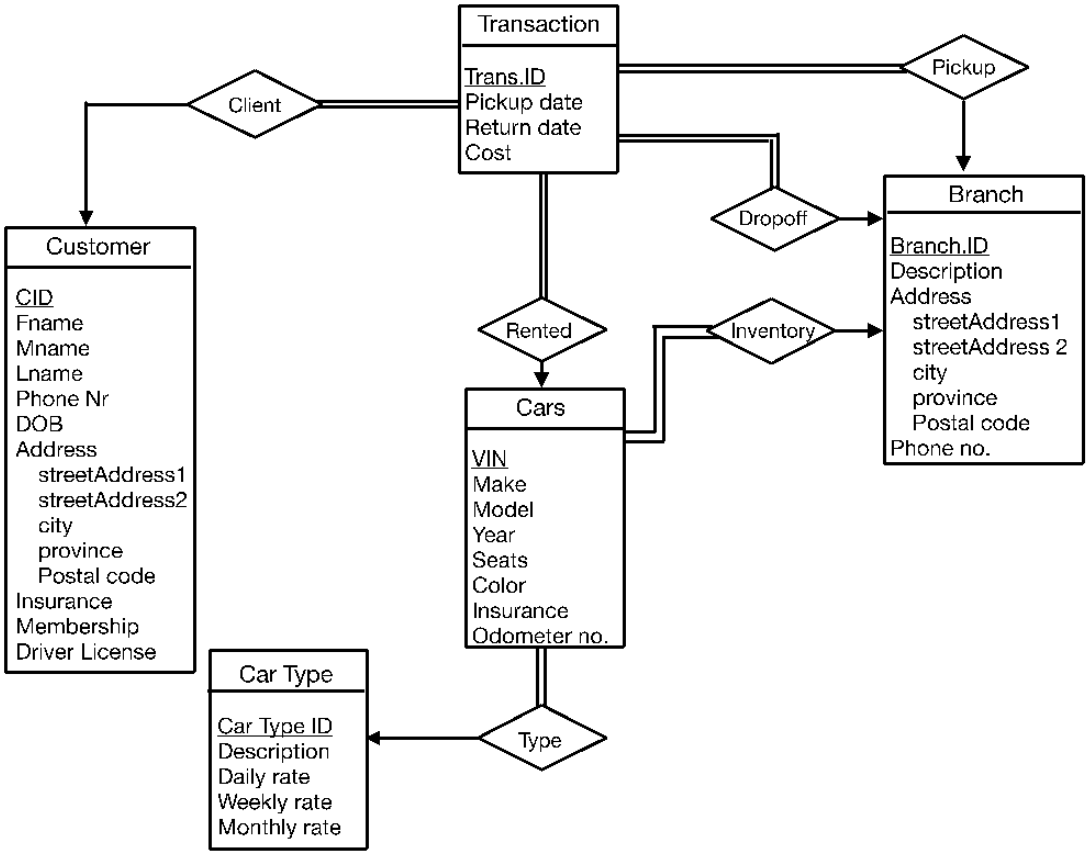
Design

Group Members: Vlad, Balkirat, Adam, Romel

Entity-Relationship Diagram



Database Description

Our database consists of 5 tables. These tables are:

1. Customer(Customer_ID, FirstName, MiddleName, LastName, Street_address1, Street_address2, City, province, postalCode, DateofBirth, Phonenumber, Insurance, Diving_License, Membership_Status)

    - This table contains all needed customer info.
    - From this table, any information about a customer can be obtained.

2. Branch(BID, Description, Street_address1, Street_address2, City, province, postalCode, Phonenumber)

    - This table contains all needed branch info
    - From this table any information about a branch can be obtained

3. CarType(CarTypeID, Description, dailyRate, weeklyRate, monthlyRate)

    - This table contains all needed info for the rates of a car with the associated CarTypeID

4. Car(VIN, Make, Model, year, noOfSeats, colour, InsuranceNo, odometernumber, Branch_ID, CarTypeID)

    Branch_ID FK references Branch
    CarType FK references Car Type

    - This table contains all needed Car info
    - From this table any information about a car can be obtained
    - A car **must** belong to a branch and **must** be associated with a CarTypeID
        - A branch can then list out all the cars that belong to it
        - A cars daily/weekly/monthly rates can then be calculated by referring to the CarTypeID

5. Rentals(TID, PickupdateDate, returnDate, Customer_ID, VIN, PickupBID, ReturnBID, total_rentValue)

    Customer_ID FK references Customer
    VIN FK references Car
    ReturnBID FK references Branch
    PickupBID FK references Branch

    - This table contains all needed rental/transaction info
    - From this table any information about a transaction can be obtained
    - A transaction **must** include a car and a branch (drop-off and pickup)

Layout design Note:

- The app tabs which contain the tick box for admin will be used at the discretion of the client. Checking the box will make the tab visible only to employees and not customers. This can be implemented upon client request.

Criteria and decisions

Deletion of database entries:

-Issue: Due to the functional dependencies we have where Car references CarType, or Rentals references Car, Customers, Branch, we cannot delete any database entries where those references appear.

Decision: If the user tries to delete a reference foreign key that appears in another table, a warning appears that deleting that entry will delete any other database entry where that reference appears.
We then updated the tables with a deletion cascading feature which deletes that entry anywhere else it exists.

Integrity Constraints

- A customer must have a (non-null) ID
- A branch must have a (non-null) ID
- A car must have a (non-null) VIN, Branch ID and Car Type ID
- A Car Type ID must have a (non-null) ID

Referential Integrity:

- If a Car Type ID is an ID that appears in one of the cars in the Car relation, that same ID must also exist in the Car Type relation
- If a Branch ID appears in one of the cars in the Car relation, that same ID must also exist in the Branch relation
- When a transaction occurs, the following must exist:
    - The branch to pickup and drop-off the rental
    - The car being selected
        - The car must belong to the same branch it is being picked up from