**CSCI 2141 – Intro to Database Systems**

Group Project – Business Rules, Data Models, & SQL Scripts

Wesam Jawich - B00710297

Jonathan Ignacio - B00632348

Harvey Wang - B00726196

November 28, 2016

**Business Rules**

The topic for our project is Project 4, the online vehicle rental agency. The service must have a list of attributes to describe a vehicle. The description includes qualities of the vehicle such as if it has AC, if it is automatic transmission, its color, its size, its passenger space, its brand, its model and its year of production. This description also applies to a single car, but that car may have duplicates that simply add to the quantity of cars in stock. Users must be able to search the database using these attributes to display any cars matching the description.

There must be a list describing the different pricings available that apply to a vehicle. The price describes the daily rental fee and late fee of a vehicle. A list of customers must be maintained to store information about who rents which vehicles.

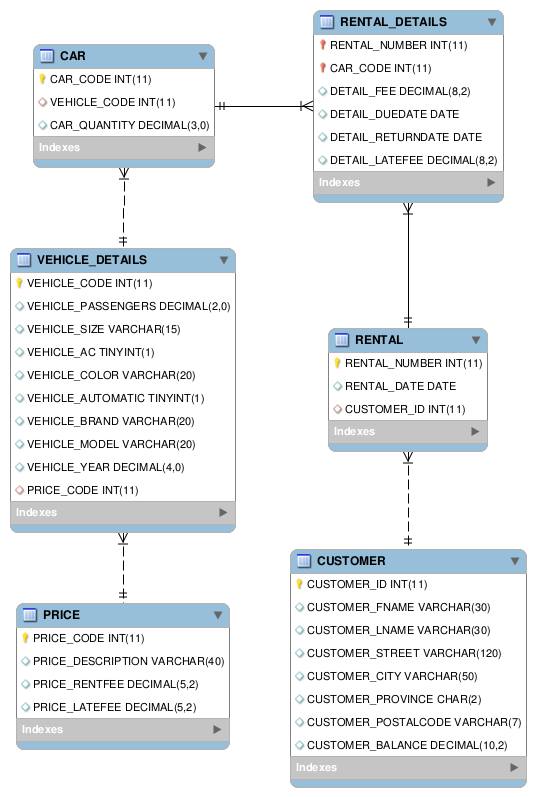
A list of customers must also be included to identify who possesses rentals and who still owes the company money. Standard attributes must be included such as first name, last name, address, etc.

Finally, rentals will be able to be created by users by specifying which car they want to rent. They will specify how long the rental is for and then the charge will be applied to their account. The rental will keep track of the fee incurred and, if the car is not returned by the due date, the late fee.

**Conceptual Model**

****

**Physical Model**

****

**SQL Scripts (Creating the database)**

CREATE DATABASE IF NOT EXISTS CAR\_RENTAL\_SERVICE;

USE CAR\_RENTAL\_SERVICE;

CREATE TABLE IF NOT EXISTS PRICE (

PRICE\_CODE INT PRIMARY KEY AUTO\_INCREMENT,

PRICE\_DESCRIPTION VARCHAR(40),

PRICE\_RENTFEE NUMERIC(5 , 2 ) CHECK (PRICE\_RENTFEE >= 0),

PRICE\_LATEFEE NUMERIC(5 , 2 ) CHECK (PRICE\_LATEFEE >= 0)

);

CREATE TABLE IF NOT EXISTS VEHICLE\_DETAILS (

VEHICLE\_CODE INT PRIMARY KEY AUTO\_INCREMENT,

VEHICLE\_PASSENGERS NUMERIC(2 , 0 ) CHECK (VEHICLE\_PASSENGERS >= 0),

VEHICLE\_SIZE VARCHAR(15),

VEHICLE\_AC BOOLEAN,

VEHICLE\_COLOR VARCHAR(20),

VEHICLE\_AUTOMATIC BOOLEAN,

VEHICLE\_BRAND VARCHAR(20),

VEHICLE\_MODEL VARCHAR(20),

VEHICLE\_YEAR NUMERIC(4 , 0 ),

PRICE\_CODE INT,

FOREIGN KEY (PRICE\_CODE)

REFERENCES PRICE (PRICE\_CODE)

);

CREATE TABLE IF NOT EXISTS CAR (

CAR\_CODE INT PRIMARY KEY AUTO\_INCREMENT,

VEHICLE\_CODE INT,

CAR\_QUANTITY NUMERIC(3 , 0 ) CHECK (CAR\_QUANTITY >= 0),

FOREIGN KEY (VEHICLE\_CODE)

REFERENCES VEHICLE\_DETAILS (VEHICLE\_CODE)

);

CREATE TABLE IF NOT EXISTS CUSTOMER (

CUSTOMER\_ID INT PRIMARY KEY AUTO\_INCREMENT,

CUSTOMER\_FNAME VARCHAR(30),

CUSTOMER\_LNAME VARCHAR(30),

CUSTOMER\_STREET VARCHAR(120),

CUSTOMER\_CITY VARCHAR(50),

CUSTOMER\_PROVINCE CHAR(2),

CUSTOMER\_POSTALCODE VARCHAR(7),

CUSTOMER\_BALANCE NUMERIC(10 , 2 )

);

CREATE TABLE IF NOT EXISTS RENTAL (

RENTAL\_NUMBER INT PRIMARY KEY AUTO\_INCREMENT,

RENTAL\_DATE DATE,

CUSTOMER\_ID INT,

FOREIGN KEY (CUSTOMER\_ID)

REFERENCES CUSTOMER (CUSTOMER\_ID)

);

CREATE TABLE IF NOT EXISTS RENTAL\_DETAILS (

RENTAL\_NUMBER INT,

CAR\_CODE INT,

DETAIL\_FEE NUMERIC(8 , 2 ),

DETAIL\_DUEDATE DATE,

DETAIL\_RETURNDATE DATE,

DETAIL\_LATEFEE NUMERIC(8 , 2 ),

PRIMARY KEY (RENTAL\_NUMBER , CAR\_CODE),

FOREIGN KEY (RENTAL\_NUMBER)

REFERENCES RENTAL (RENTAL\_NUMBER),

FOREIGN KEY (CAR\_CODE)

REFERENCES CAR (CAR\_CODE)

);

**SQL Scripts (Populating the tables)**

**Prices**

INSERT INTO `PRICE` (`PRICE\_DESCRIPTION`, `PRICE\_RENTFEE`, `PRICE\_LATEFEE`) VALUES

('Basic', 25, 25);

INSERT INTO `PRICE` (`PRICE\_DESCRIPTION`, `PRICE\_RENTFEE`, `PRICE\_LATEFEE`) VALUES

('Economy', 50, 30);

INSERT INTO `PRICE` (`PRICE\_DESCRIPTION`, `PRICE\_RENTFEE`, `PRICE\_LATEFEE`) VALUES

('Standard', 75, 35);

INSERT INTO `PRICE` (`PRICE\_DESCRIPTION`, `PRICE\_RENTFEE`, `PRICE\_LATEFEE`) VALUES

('Premium', 100, 40);

INSERT INTO `PRICE` (`PRICE\_DESCRIPTION`, `PRICE\_RENTFEE`, `PRICE\_LATEFEE`) VALUES

('Ultimate', 200, 50);

**Vehicle Details**

INSERT INTO `VEHICLE\_DETAILS` (`VEHICLE\_PASSENGERS`, `VEHICLE\_SIZE`, `VEHICLE\_AC`, `VEHICLE\_COLOR`, `VEHICLE\_AUTOMATIC`, `VEHICLE\_BRAND`, `VEHICLE\_MODEL`, `VEHICLE\_YEAR`, `PRICE\_CODE`)

VALUES (4, 'Coupe', 1, 'Orange', 1, 'BMW', 'M4', '2016', '04');

INSERT INTO `VEHICLE\_DETAILS` (`VEHICLE\_PASSENGERS`, `VEHICLE\_SIZE`, `VEHICLE\_AC`, `VEHICLE\_COLOR`, `VEHICLE\_AUTOMATIC`, `VEHICLE\_BRAND`, `VEHICLE\_MODEL`, `VEHICLE\_YEAR`, `PRICE\_CODE`)

VALUES (4, 'Sedan', 1, 'Red', 1, 'BMW', 'M3', '2014', '04');

INSERT INTO `VEHICLE\_DETAILS` (`VEHICLE\_PASSENGERS`, `VEHICLE\_SIZE`, `VEHICLE\_AC`, `VEHICLE\_COLOR`, `VEHICLE\_AUTOMATIC`, `VEHICLE\_BRAND`, `VEHICLE\_MODEL`, `VEHICLE\_YEAR`, `PRICE\_CODE`)

VALUES (2, 'Convertible', 1, 'White', 0, 'Mercedes-Benz', 'SL400 Roadster', '2016', '04');

INSERT INTO `VEHICLE\_DETAILS` (`VEHICLE\_PASSENGERS`, `VEHICLE\_SIZE`, `VEHICLE\_AC`, `VEHICLE\_COLOR`, `VEHICLE\_AUTOMATIC`, `VEHICLE\_BRAND`, `VEHICLE\_MODEL`, `VEHICLE\_YEAR`, `PRICE\_CODE`)

VALUES (4, 'Sedan', 1, 'Black', 1, 'Rolls-Royce', 'Phantom', '2012', '05');

INSERT INTO `VEHICLE\_DETAILS` (`VEHICLE\_PASSENGERS`, `VEHICLE\_SIZE`, `VEHICLE\_AC`, `VEHICLE\_COLOR`, `VEHICLE\_AUTOMATIC`, `VEHICLE\_BRAND`, `VEHICLE\_MODEL`, `VEHICLE\_YEAR`, `PRICE\_CODE`)

VALUES (4, 'Coupe', 1, 'Black', 1, 'Rolls-Royce', 'Wraith', '2015', '05');

INSERT INTO `VEHICLE\_DETAILS` (`VEHICLE\_PASSENGERS`, `VEHICLE\_SIZE`, `VEHICLE\_AC`, `VEHICLE\_COLOR`, `VEHICLE\_AUTOMATIC`, `VEHICLE\_BRAND`, `VEHICLE\_MODEL`, `VEHICLE\_YEAR`, `PRICE\_CODE`)

VALUES (2, 'Coupe', 1, 'Red', 1, 'Ferrari', 'LaFerrari', '2016', '05');

INSERT INTO `VEHICLE\_DETAILS` (`VEHICLE\_PASSENGERS`, `VEHICLE\_SIZE`, `VEHICLE\_AC`, `VEHICLE\_COLOR`, `VEHICLE\_AUTOMATIC`, `VEHICLE\_BRAND`, `VEHICLE\_MODEL`, `VEHICLE\_YEAR`, `PRICE\_CODE`)

VALUES (2, 'Coupe', 1, 'Yellow', 0, 'Aston Martin', 'Vantage', '2013', '04');

INSERT INTO `VEHICLE\_DETAILS` (`VEHICLE\_PASSENGERS`, `VEHICLE\_SIZE`, `VEHICLE\_AC`, `VEHICLE\_COLOR`, `VEHICLE\_AUTOMATIC`, `VEHICLE\_BRAND`, `VEHICLE\_MODEL`, `VEHICLE\_YEAR`, `PRICE\_CODE`)

VALUES (2, 'Coupe', 1, 'Blue', 1, 'Lamborghini', 'Aventador', '2015', '05');

INSERT INTO `VEHICLE\_DETAILS` (`VEHICLE\_PASSENGERS`, `VEHICLE\_SIZE`, `VEHICLE\_AC`, `VEHICLE\_COLOR`, `VEHICLE\_AUTOMATIC`, `VEHICLE\_BRAND`, `VEHICLE\_MODEL`, `VEHICLE\_YEAR`, `PRICE\_CODE`)

VALUES (4, 'Sedan', 1, 'Black', 1, 'Bentley', 'Mulsanne', '2016', '05');

INSERT INTO `VEHICLE\_DETAILS` (`VEHICLE\_PASSENGERS`, `VEHICLE\_SIZE`, `VEHICLE\_AC`, `VEHICLE\_COLOR`, `VEHICLE\_AUTOMATIC`, `VEHICLE\_BRAND`, `VEHICLE\_MODEL`, `VEHICLE\_YEAR`, `PRICE\_CODE`)

VALUES (4, 'Convertible', 1, 'Red', 1, 'Maserati', 'GranTurismo', '2012', '05');

INSERT INTO `VEHICLE\_DETAILS` (`VEHICLE\_PASSENGERS`, `VEHICLE\_SIZE`, `VEHICLE\_AC`, `VEHICLE\_COLOR`, `VEHICLE\_AUTOMATIC`, `VEHICLE\_BRAND`, `VEHICLE\_MODEL`, `VEHICLE\_YEAR`, `PRICE\_CODE`)

VALUES (4, 'Sedan', 1, 'Red', 0, 'BMW', 'M6', '2014', '05');

**Customers**

INSERT INTO `CUSTOMER` (`CUSTOMER\_FNAME`, `CUSTOMER\_LNAME`, `CUSTOMER\_STREET`, `CUSTOMER\_CITY`, `CUSTOMER\_PROVINCE`, `CUSTOMER\_POSTALCODE`, `CUSTOMER\_BALANCE`)

VALUES ('Timothy', 'Thompson', '25 South Street', 'Halifax', 'NS', 'B0B 0B0', '0');

INSERT INTO `CUSTOMER` (`CUSTOMER\_FNAME`, `CUSTOMER\_LNAME`, `CUSTOMER\_STREET`, `CUSTOMER\_CITY`, `CUSTOMER\_PROVINCE`, `CUSTOMER\_POSTALCODE`, `CUSTOMER\_BALANCE`)

VALUES ('Drake', 'Golfer', '409 Forward Crescent', 'Bedford', 'NS', 'B1B 1B1', '500');

INSERT INTO `CUSTOMER` (`CUSTOMER\_FNAME`, `CUSTOMER\_LNAME`, `CUSTOMER\_STREET`, `CUSTOMER\_CITY`, `CUSTOMER\_PROVINCE`, `CUSTOMER\_POSTALCODE`, `CUSTOMER\_BALANCE`)

VALUES ('Fredrick', 'MacDougall', '823 Crabdale Avenue', 'Truro', 'NS', 'B9L 4T9', '0');

INSERT INTO `CUSTOMER` (`CUSTOMER\_FNAME`, `CUSTOMER\_LNAME`, `CUSTOMER\_STREET`, `CUSTOMER\_CITY`, `CUSTOMER\_PROVINCE`, `CUSTOMER\_POSTALCODE`, `CUSTOMER\_BALANCE`)

VALUES ('Tracy', 'Foxton', '535 Forkspoon Lane', 'Moncton', 'NB', 'C3T 2Y3', '1000');

INSERT INTO `CUSTOMER` (`CUSTOMER\_FNAME`, `CUSTOMER\_LNAME`, `CUSTOMER\_STREET`, `CUSTOMER\_CITY`, `CUSTOMER\_PROVINCE`, `CUSTOMER\_POSTALCODE`, `CUSTOMER\_BALANCE`)

VALUES ('Taylor', 'Erikson', '190 Keltic Drive', 'Sydney River', 'NS', 'B4L 9C2', '0');

INSERT INTO `CUSTOMER` (`CUSTOMER\_FNAME`, `CUSTOMER\_LNAME`, `CUSTOMER\_STREET`, `CUSTOMER\_CITY`, `CUSTOMER\_PROVINCE`, `CUSTOMER\_POSTALCODE`, `CUSTOMER\_BALANCE`)

VALUES ('Robert', 'Chang', '122 Main Street', 'Glace Bay', 'NS', 'C2B 5B2', '0');

INSERT INTO `CUSTOMER` (`CUSTOMER\_FNAME`, `CUSTOMER\_LNAME`, `CUSTOMER\_STREET`, `CUSTOMER\_CITY`, `CUSTOMER\_PROVINCE`, `CUSTOMER\_POSTALCODE`, `CUSTOMER\_BALANCE`)

VALUES ('Sarah', 'Goode', '708 Fairview Road', 'Yarmouth', 'NS', 'Y6T 3B6', '250');

**Cars**

INSERT INTO `CAR` (`VEHICLE\_CODE`, `CAR\_QUANTITY`) VALUES (01, 2);

INSERT INTO `CAR` (`VEHICLE\_CODE`, `CAR\_QUANTITY`) VALUES (02, 1);

INSERT INTO `CAR` (`VEHICLE\_CODE`, `CAR\_QUANTITY`) VALUES (03, 3);

INSERT INTO `CAR` (`VEHICLE\_CODE`, `CAR\_QUANTITY`) VALUES (04, 1);

INSERT INTO `CAR` (`VEHICLE\_CODE`, `CAR\_QUANTITY`) VALUES (05, 1);

INSERT INTO `CAR` (`VEHICLE\_CODE`, `CAR\_QUANTITY`) VALUES (06, 1);

INSERT INTO `CAR` (`VEHICLE\_CODE`, `CAR\_QUANTITY`) VALUES (07, 3);

INSERT INTO `CAR` (`VEHICLE\_CODE`, `CAR\_QUANTITY`) VALUES (08, 2);

INSERT INTO `CAR` (`VEHICLE\_CODE`, `CAR\_QUANTITY`) VALUES (09, 1);

INSERT INTO `CAR` (`VEHICLE\_CODE`, `CAR\_QUANTITY`) VALUES (10, 1);

**SQL Scripts (Creating a rental)**

The following script is more complicated. The purpose is to create Most of the logic is handled through the front-end application and the procedures are included as pseudocode in the following SQL Scripts.

# script for creating a new rental.

# the script will require these inputs variables, handled by the front-end:

# inputID - the id of the customer making the rental

# inputcarcode - the car code of the desired rental car

# inputdate - the date of the rental

# inputduration - how many days the car is to be rented

# front-end must check for the following conditions:

# 1. specified customer\_id must be an existing customer:

# can check if null with

# (SELECT CUSTOMER\_ID FROM CUSTOMER WHERE CUSTOMER\_ID = inputID)

# 2. specified car\_code must exist AND car\_quantity must be above 0

# can check if null with

# (SELECT CAR\_CODE FROM CAR WHERE CAR\_CODE = inputcarcode AND CAR\_QUANTITY > 0)

# 3. specifed date for rental must have not already passed

# this may be handled on the front-end using system date

# 4. inputduration must be > 0

#

#if the conditions are met, execute the rental:

#create the basic rental

INSERT INTO RENTAL (`RENTAL\_DATE`, `CUSTOMER\_ID`) VALUES (inputdate, inputID) #the input variables

#get the rental code of the newly created rental for the details

#int rentnum = recent primary key

SELECT LAST\_INSERT\_ID() #should return the most recently inserted primary key (our new rental number)

#get the daily rental fee based on the car\_code as float dailyprice

SELECT PRICE\_RENTFEE

FROM CAR AS C

INNER JOIN VEHICLE\_DETAILS AS V

on C.VEHICLE\_CODE = V.VEHICLE\_CODE

INNER JOIN PRICE AS P

on P.PRICE\_CODE = V.PRICE\_CODE

#float totalprice = dailyprice\*inputduration //total price of the rental given daily rental fee and number of days

INSERT INTO RENTAL\_DETAILS (`RENTAL\_NUMBER`, `CAR\_CODE`, `DETAIL\_FEE`, `DETAIL\_DUEDATE`, `DETAIL\_LATEFEE`) #return date is null until a return is made

VALUES (rentnum, inputcarcode, totalprice, inputdate + inputduration, 0); #latefee is always 0 at the being

#update the customer’s balance according to the rental price

UPDATE CUSTOMER SET CUSTOMER\_BALANCE = CUSTOMER\_BALANCE + totalprice WHERE CUSTOMER\_ID = inputID

#and lastly, reduce the quantity of the cars by 1

UPDATE CAR

SET CAR\_QUANTITY = CAR\_QUANTITY - 1

WHERE CAR\_CODE = inputcarcode