**SQL Queries**

**Create Account statement:**

INSERT INTO gtcr\_employee (Username, Password)

VALUES ($Username, $Password)

**Add Car**

INSTERT INTO car (Vehicle\_Sno, model\_name, car\_type, LocationName, color, hourly\_rate, daily\_rate, seating\_capacity, transmission\_type,

bluetooth\_connectivity, auxilary\_cable)

VALUES ($Vehicle\_Sno, $model\_name, $car\_type, $LocationName, $color, $hourly\_rate, $daily\_rate, $seating\_capacity, $transmission\_type,

$bluetooth\_connectivity, $auxilary\_cable)

**Search for a car to rent:**

//assume $Username of current user is managed by application

//read $pickuptime, $returntime, $location

SELECT model\_name, car\_type, LocationName, color, hourly\_rate, discounted\_rate,

daily\_rate, seating\_capacity, transmission\_type, bluetooth\_connectivity, auxillary\_cable,

available\_till, ((SELECT hourlyrate from car where Car.VehicleSno NOT in Reservation)

\*($returntime - $pickuptime) - (Select discount from driving plan INNER JOIN gt\_student\_faculty\_member ON gt\_student\_faculty\_member.driving\_plan\_type = DrivingPlan.driving\_plan\_type) as estimatedcost,

from car

where $pickuptime, $returntime NOT in Reservation or Reservation.returntime < $pickuptime or

Reservation.pickupdatetime < $returntime and Car.LocationName = $location

**Rent a car:**

INSERT INTO Reservation(pick\_up\_datetime, ReturnDateTime, Username, LateBy, ReturnStatus, EstimatedCost, LateFees, LocationName, VehicleSno)

VALUES ($pickuptime, $returntime, $Username, 0, “ON TIME”, [estimated cost],

0, $location, $vehiclesno)

//we are assuming that when we make a reservation, the car is not late by any time yet.

//we are assuming that when a reservation is made, the reservation status is on time

**Administrative report:**

Select Vehicle\_Sno, car\_type, model\_name, ((SELECT EstimatedCost from car JOIN Reservation

on Car.VehicleSno = Reservation.VehicleSno) + Select LateFees from car join

Reservation on car.VehicleSno = Reservation.sno)

as ReservationRevenue,

(SELECT LateFees from car join Reservation on Car.VehicleSno = Reservation.VehicleSno) as LateFeeRevenue

FROM car JOIN Reservation

ON Car.Vehicle\_Sno= Reservation.VehicleSno And Reservation.PickupTime BETWEEN UNIX\_TIMESTAMP(DATE\_SUB(now(), INTERVAL -3 MONTH) AND UNIX\_TIMESTAMP(now())

GroupBy car\_type;

**Frequent Users Report;**

SELECT Username, Driving\_Plan, Count(Pick\_Up\_Date) AS No. Reservations per month

FROM GT\_student\_faculty\_member INNER JOIN Reservation ON

GT\_student\_faculty\_member.Username=Reservation.Username

WHERE DATEDIFF(MONTH, Pick\_Up\_Date, CURRDATE()) < 3

GROUP BY Username

ORDER BY No. Reservations per month DESC

LIMIT 5

**Location Preference Report**

SELECT MONTH(Pick\_up\_date) AS Month, Location\_Name AS Location, Count(Pick\_Up\_Date) AS No of Reservations, Count(\*) AS Total no of hours

FROM Location INNER JOIN Reservation

ON Location.Location\_Name=Reservation.Location\_Name

WHERE DATEDIFF(MONTH, Pick\_Up\_Date, CURRDATE()) < 3

GROUP BY Month

ORDER BY Month DESC

**Enter Payment Information:**

INSERT INTO CreditCard (CardNo, NameonCard, ExpiryDate, BillingAddress)

Values($CardNo, $NameonCard, $ExpiryDate, $BillingAddress)

WHERE CurrDate() < ExpiryDate

Semantic Constraint- current date cant be less than expiry date

**Update Payment Information:**

UPDATE CreditCard SET (CardNo, NameonCard, ExpiryDate, BillingAddress)

Values($CardNo, $NameonCard, $ExpiryDate, $BillingAddress)

WHERE CurrDate() < ExpiryDate

Semantic Constraint- current date cant be less than expiry date

**Enter Personal Information:**

INSERT INTO GT\_student\_faculty\_member

(First\_Name, Middle\_Initial, Last\_Name, Email\_Address, Phone\_Number, Address)

VALUES

($First\_Name, $Middle\_Initial, $Last\_Name, $Email\_Address, $Phone\_Number, $Address)

**Update Personal Information:**

UPDATE GT\_student\_faculty\_member SET

(First\_Name, Middle\_Initial, Last\_Name, Email\_Address, Phone\_Number, Address)

VALUES

($First\_Name, $Middle\_Initial, $Last\_Name, $Email\_Address, $Phone\_Number, $Address)

**Login of users:**

SELECT username,password FROM gt\_student\_faculy\_member WHERE username=$username AND password=$password

**login of employees**

SELECT username,password FROM gtcr\_employee WHERE username=$username AND password=$password

**Login of Administrator:**

SELECT username,password FROM administrator WHERE username=$username AND password=$password

**Driving Plan for Everyone**

SELECT \* FROM driving\_plan

**driving plan for users**

SELECT driving\_plan FROM gt\_student\_faculty\_member WHERE username=$username

UPDATE gt\_student\_faculty\_member SET driving\_plan=$chosenplan WHERE usename=$username

**driving plan for employees and administrators**

SELECT username,driving\_plan FROM gt\_student\_faculty\_member

**viewing rental info for users**

SELECT \* FROM Reservation WHERE username=$username

**viewing rental info for employees**

SELECT \* FROM Reservation

**extend reservation time(done by employee when user calls)**

INSERT INTO Reservation\_Extended\_Time (username,pickup\_datetime,extended\_time,return\_datetime)

VALUES ($customerusername,(SELECT pickup\_date\_time FROM Reservation WHERE Username=$customerusername),$extendedtime,

(SELECT return\_date\_time FROM Reservation WHERE Username=$customerusername)+$extendedtime)

WHERE username=$customerusername

**rental request change**

UPDATE Reservation SET (pickup\_datetime,ReturnDateTime,EstimatedCost,LocationName,VehicleSno)

VALUES ($pickuptime,$returntime,($returntime-$pickuptime)\*(SELECT hourly\_rate FROM car WHERE VehicleSno=$vehiclesno),

(SELECT location FROM car WHERE VehicleSno=$vehiclesno),$vehiclesno)

WHERE VehicleSno=$vehiclesno AND pick\_up\_time<current\_timestamp

Semantic Constraint: check that the capacity of the location is not exceeded

Count(SELECT VehicleSno FROM car WHERE Locationname = $locationName

**Maintainence History** (selected either by which car request was for or which user put it in)

SELECT \* FROM maintainence\_request\_problems WHERE VehicleSno=$vehiclesno

or

SELECT \* FROM maintainence\_request\_problems WHERE Username=$username

**Manage cars**

Semantic constraint to make sure that we cant change a car location if the location capacity will be exceeded

WHERE VehicleSno=$vehiclesno AND COUNT(SELECT VehicleSno FROM car WHERE location\_name = $location) < SELECT Location.capacity

FROM Location WHERE  
 Location.Name = $location

(used in queries below)

UPDATE car SET (location\_name) VALUES ($location)

WHERE VehicleSno=$vehiclesno AND COUNT(SELECT VehicleSno FROM car WHERE location\_name = $location) < SELECT Location.capacity

FROM Location WHERE  
 Location.Name = $location

INSERT INTO car (**Vehicle\_Sno,**auxilary\_cable,under\_maintainence\_flag,model\_name,car\_type,color,hourly\_rate,daily\_rate,blutoothe\_connectivity,seating\_capacity,Location\_name,transmission\_type)

VALUES ($vehiclesno,$auxcable,$undermaintainence,$modelname,$cartype,$color,$hourly,$daily,$bluetooth,$seating,$location,$transmission) WHERE VehicleSno=$vehiclesno AND COUNT(SELECT VehicleSno FROM car WHERE location\_name = $location) < SELECT Location.capacity

FROM Location WHERE  
 Location.Name = $location

**Maintenance Requests**

SELECT Location\_Name FROM Location

SELECT Make, Model FROM Car INNER JOIN Location ON Location.Location\_Name=Car.Location\_Name

WHERE Car.Location\_Name=$Location\_Name

UPDATE Car SET (maintenance\_flag) VALUES (YES)

WHERE Location\_Name=$Location\_Name AND Make=$Make AND Model=$Model