CISC 332 / CMPE 332 - Database Management Systems

Winter 2017

Course Project – K-Town Car Share

The course project is intended to give students an opportunity to participate in all phases of the development of a database application. The development of the application requires students to use ER modeling techniques, relational modeling techniques, SQL, MySQL and PHP.

You are required to develop a Web-based application and backend database for a new car sharing venture in Kingston called K-Town Car Share (KTCS for short). KTCS will provide people with the convenience of a vehicle when they need it without having to purchase one and promote a healthier environment.

KTCS plans to maintain a fleet of cars placed in several designated parking locations across Kingston. Individuals become members of KTCS using a registration form on the KTCS Web site. Members can then use the web site to perform all of the tasks related to the use of a KTCS vehicle including reserving a vehicle, picking up a reserved vehicle and returning a vehicle.

KTCS makes the following assumptions regarding the operation of their service:

- Only registered members of KTCS are able to reserve a car. When a member makes a reservation they are given a unique access code that can be used to unlock the car.
- Rentals are on a daily basis.
- Members are charged a monthly membership fee and then a daily rental fee that can vary depending on the type of car. A monthly invoice is generated for each member and sent to them by email.
- A vehicle is always picked up from, and returned to, the same parking location.
- A member fills in a pick-up form when picking up a vehicle and a drop-off form when returning it.
- Members pay for any gas they use and return a car with the same amount of gas as when they picked it up.
- A rental history is maintained for each member and for each car.
- Members are able to rate vehicles and post comments on their experience with a particular vehicle.
- A maintenance history is maintained for each vehicle.

Data Requirements

The database must, at a minimum, contain information about the following:

- Parking locations: address; number of spaces.
- Cars: vehicle identification number (VIN); make, model and year; pick-up/drop-off location; daily rental fee.
- Car rental history: for each rental – member; pick-up odometer reading; drop-off odometer location; status on return (normal, damaged, not running).
- Car maintenance history: for each maintenance visit – car; date; odometer reading; maintenance type (scheduled, repair, body work); description.
- KTCS members: member number; name, address, phone number and email; driver's license number; annual membership fee.
- Member rental history: for each rental – car, pick-up odometer reading; drop-off odometer location; status on return (normal, damaged, not running).
- Rental comments: for each comment – member; car; rating (1 – 4 *'s); comment text.
- Reservations: reservation number; member; car; date; access code; length of reservation (in days).

Functional Requirements

The KTCS application needs to support 2 categories of functions – one for members and one for system administrators.

The KTCS system will need to support the following tasks by a member:

1) Register as a new member.
2) Find KTSC locations.
3) Find all cars available to rent on a specific day.
4) Reserve a car.
5) Pick-up a car (record the time, odometer reading, car status).
6) Drop-off a car (record the time, odometer reading, car status).
7) Show the member's rental history.
8) Provide a feedback on a rental.

The KTCS system also needs to support the following tasks for an administrator:

1) Generate a monthly invoice for a member.
2) Add a car to the fleet.
3) Show the rental history for a car.
4) For a given location, show all cars currently available in that location and reservations for those cars, if any.
5) Show all cars that have travelled 5000 kms or more since their last maintenance.
6) Find the car with the highest/lowest number of rentals.

7) Find all cars that are damaged or need a repair.
8) Show all the reservations for a given day.
9) Respond to a member's comment.

The above data and functional requirements are a minimum and you are free to add other information or functionality you think are necessary.

Project Deliverables

Projects should be done in groups of 2 or 3 individuals. The groups will remain the same for all project deliverables. Group members will be assigned the same mark for each project deliverable. Each group should send an email to the professor prior to handing in the first deliverable of the project identifying the members of the group. Students may choose to work alone but no compensation will be made for this when marking. Groups of larger than 3 students must be approved by the professor and will be expected to propose and implement an extension to the given application requirements.

Note: Students with enough previous experience developing similar applications can request a "personalized" project. Arrangements must be made with the professor before the first stage of the project is due!

Hand in each of the database design and final report deliverables as single pdf files using OnQ. One submission per team is sufficient.

Deliverable 1: Database Design and Implementation. Due March 2, 2017. (10% of final mark)

Design the ER schema for your database. Show all constraints and state any assumptions that you make which are not specified in the requirements. As you design your schema, keep in mind the Functional Requirements. A good design both represents the necessary data and facilitates effective use of the data. Capture as many of the constraints as possible in your design. Your ER diagram may be hand-drawn, but it must be legible.

Convert the ER schema for your application to a relational schema. Make sure you indicate the primary key and any foreign keys, as well as NOT NULL constraints, for each table. Create a database instance for your schema in MySQL and populate your database with some reasonable data. A few tuples for each table is sufficient.

Hand in the following:

- A list of all your assumptions
- Your ER schema.
- The DDL for your relational schema.
- A listing of the sample contents of each of the relations in your database (produced using phpmyadmin or a PHP script.

Deliverable 2: Application Demonstration. Due the last week of classes. (15% of final mark)

Write PHP programs that implement the functional requirements outlined above. You will need to add more data to your database to adequately demonstrate all the functionality and to make the results more interesting. You may assume that user input is correct so input syntax checking can be minimal. You must, however, handle cases where queries return no results. As the minimum, your program can simply use prompted input and formatted output from the browser. Higher marks will be given to more sophisticated Web applications.

You will be required to demonstrate your application in the CASLAB (on a CASLAB, or your own, machine). All team members should be present and participate in the demo. A demo schedule in the last week of classes will be established later in the term.

For the demo you should

- Create a sample database with enough data to adequately demonstrate all of the required functionality.
- Prepare a script to follow that allows you to show all of the features of your application in a logical sequence in a 10 minute demo.
- Have a hard copy of the ER schema available for reference during the demo.
- Be prepared to answer questions about the capabilities of your application and the development process you followed.

Deliverable 3: Final Report. Due April 6, 2017. (20% of final mark)

Hand in the following:

- List of your assumptions.
- The final versions of your ER and relational schemas.
- A state machine diagram that describes the flow through the application's Web pages.
- The SQL interactions associated with the states in the state machine diagram and sample output for each SQL statement.
- A discussion including the following: problems encountered during the development and how you solved them; important design and implementation decisions; the technologies and tools you used in developing your application, why you chose them and what your experience was using them; things you would like to go back and change or do differently if you had the chance.
- A user's guide to your application. This should be a document you could give to a new KTSC member or administrator to help them use your application.