CS 166 Project Description

Tuesday, July 2, 2019

1 Introduction

In this project, we will model and build a database for a mechanics shop. The system will be used track information about customers, cars, mechanics, car ownership, service request and billing information.

The project is divided into three phases: (i) requirement analysis using the ER-model, (ii) Relational schema design, and (iii) implementation. Finally, the students will to demo the complete working system at the end of the quarter.

Phase 1: ER Design

In the first phase, you will do the requirement analysis using ER-model. All the requirements can be obtained from Section 2 of this document. After this phase you should generate and ER-diagram with any other supporting documentation, describing the assumptions you made. For the ER-diagram you can use any graphical editor you want, and you should finally create a PDF file using ER notations from the lectures/lab/book. You have to submit all your files compressed into a single file through iLearn by the deadline. In this phase, we will evaluate the correctness of your ER-diagram. You can make reasonable assumptions on your design, as long as:

- That you state them clearly in the documentation for this phase.
- They do not contradict the system requirements analysis we provide.

*The due date for this phase is: **July 17th, 2019**.*

Phase 2: Relational Schema Design

In this phase, we will provide you with a common (final) ER-diagram (so that the whole class will proceed with the same design). This final ER-diagram will be the starting point for the second phase, which involves the creation of the relational schema.

Your task in this phase will be to translate the provided ER design to a PostgreSQL relational database schema. The database schema will be in a form of a single executable SQL script (*.sql file with SQL statements), and you have to turn in your script through iLearn. For this phase, you will be evaluated for the correctness and completeness of your relational schema.

You may find some constraints in the model and/or system requirement analysis that are not possible to represent or enforce in the relational schema. You may specify all these issues in the documentation, and it will be considered in your final grade.

The due date for this phase is ***July 31st, 2019****.*

Phase 3: Implementation

Your tasks in this phase will be:

🎬 Develop a client application using the Java Database Connector (jdbc) for psql.
🎬 Use the client application to support specific functionality and queries for your online booking system.

In this phase, we will provide you with a create.sql that recreates the relational schema of phase 2. You will use this schema to test and demo your application to us. Additionally, we will give you a collection of .csv files containing dummy data that are compatible with the provided relational schema. You will have to create your own .sql scripts to insert the data from the given .csv files into the database.

Finally, we will give you a skeleton code for the client application. The code will be in Java and will contain some basic functionality that will help you to communicate with the database and issue various .sql statements. You will have to implement your own code for a certain number of functions, described in more detail below:

🎬 **Add Customer Function**
Add a new customer into the database. You should provide an interface that takes as input the information of a new customer (i.e. first, last name, phone, address) and checks if the provided information are valid based on the constraints of the database schema.

🎬 **Add Mechanic**
Add a new mechanic into the database. You should provide an interface that takes as input the information of a new mechanic (i.e. first, last name, specialty, experience) and checks if the provided information is valid based on the constraints of the database schema.

🎬 **Add Car**
This function should allow for adding a new car into the database. You should provide an interface that takes as input the information of a new car (i.e. vin, make, model, year) checking if the provided information is valid based on the constrains of the database schema.

🎬 **Initiate a Service Request**
This function will allow you to add a service request for a customer into the database. Given a last name, the function should search the database of existing customers. If many customers match, a menu option should appear listing all customers with the given last name asking the user to choose which customer has initiated the service request. Otherwise, the client application should provide the option of adding a new customer. If an existing customer is chosen, the client application should list all cars associated with that client providing the option to initiate the service request for one of the listed cars, otherwise a new car should be added along with the service request information for it.

🎬 **Close A Service Request**
This function will allow you to complete an existing service request. Given a service request number and an employee id, the client application should verify the information provided and attempt to create a closing request record. You should make sure to check for the validity of the provided inputs (i.e. does the mechanic exist, does the request exist, valid closing date after request date, etc.)

🎬 **List date, comment, and bill for all closed requests with bill lower than 100**
List the customers that have paid less than 100 dollars for repairs based on their previous service requests.

🎬 **List first and last name of customers having more than 20 different cars**
Find how many cars each customer has counting from the ownership relation and discover who has more than 20 cars.

🎬 **List Make, Model, and Year of all cars build before 1995 having less than 50000 miles**
Get the odometer from the service_requests and find all cars before 1995 having less than 50000 miles in the odometer.

🎬 **List the make, model and number of service requests for the first k cars with the highest number of service orders.**
Find for all cars in the database the number of service requests. Return the make, model and number of service requests for the cars having the k highest number of service requests. The k value should be positive and larger than 0. The user should provide this value. Focus on the open service requests.

🎬 **List the first name, last name and total bill of customers in descending order of their total bill for all cars brought to the mechanic.**
For all service requests find the aggregate cost per customer and order customers according to that cost. List their first, last name and total bill. We strongly recommended that you start early and allocate at least 25 hours per person to get it finished. Don't forget that each group has to schedule a presentation to show the system running with all its functionalities to the TA. Slots for the presentation will be available online on a first come-first served basis.

For this phase you will be evaluated based on the system requirements. Your GUI and source code will also be taken into consideration in your final evaluation. Groups that implement systems with user-friendly interfaces, extra functionalities and error handling (i.e. invalid values, wrong operations, meaningful messages) will receive an extra credit. A final report about your system along with its source code has to be submitted through iLearn. Please keep in mind that we have already prepared a set of data, which you can load in the database once you create it.

*The due date for this phase is **August 26th, 2019**.*

1.1 Grading

Your contribution to this project will be graded based on the following characteristics:

🎬 Phase 1 (30%)
- Conceptual Design (ER Diagram)
- This phase will be completed individually. You must submit your solutions on iLearn.

🎬 Phase 2 (10%)
- Logical DB Design (Relational Database Schema)
- This phase will also be completed individually. You must submit your solutions on iLearn.

🎬 Phase 3 (60%)
- Documentation of the project including details about your assumptions (10%)
- Implementation of SQL queries in the Client Application (30%)
- Physical DB Design (DB performance tuning indexes) (10%)
- Documentation of the project including any assumptions that you have made (10%)
- This phase will be performed in groups of TWO students. No individual submissions are allowed. Choose your partner wisely because the final evaluation is based on the group performance! In your report explicitly

enumerate the tasks that each member of your group was responsible for. If one of the group members does most of the work the grade will be proportional to the effort. If you are not able to find a partner, one will be assigned to you at random. Please e-mail the TAs immediately if you need help finding a partner. Extra credit will be given for good GUI design and interface, any dataset or schema changes/extensions, etc. (10%)

2 Requirement Analysis

1. **Customer**

🎬 The customer has a first name, last name, phone number and address.
🎬 Customers can own many different cars
🎬 Customers may bring any of their car for service
🎬 Customers bring their car for service at a certain date indicating to the mechanic if there is any problem with the car

2. **Cars**

🎬 Cars have a unique VIN, a year, a make and a model
🎬 A car has exactly one owner.
🎬 A car may have many outstanding service requests.
🎬 For each outstanding service request, the car has an odometer reading and the date the car was brought in by the customer.
🎬 The service request will be closed when the car is fixed a mechanic

3. **Mechanic**

🎬 Mechanics have a first name, a last name, an employee id, and some years of experience.
🎬 Mechanics work on a single car at a time.
🎬 When a mechanic fixes a car, he/she closes the service request indicating when it was closed, any comments they have and the final bill.
🎬 Exactly one mechanic can create a service request.
🎬 Both open and closed service requests need to be available at any time.