CECS 323 Term Project

This is a team Project

Each team will be three to four students. You will need at least three strong participants to get this all done in the time allotted. Please let me know the team roster as soon as you form the team so that I can get group accounts set up promptly. Each of you will fill out a peer evaluation of the other members on the team. You can find that here. I also ask each team to agree among themselves on who will function as team lead. The team lead will be in charge of making assignments of specific tasks to specific team members, and getting status on a regular basis. I will be looking to the team leads for a weekly status report on how the team as a whole is doing. There will be extra credit points awarded to the team lead if their team feels that they are doing a good job.

Project Description

You are designing a database and proof of concept for Dave's Automotive, a small auto repair shop that specializes in providing preventative maintenance that saves the customer in the end by staying ahead of the normal wear and tear on a vehicle.

Dave's Automotive has three types of customers: steady, premier, and prospective. The steady customers are those who have signed up for generated E-mail notifications to alert them when their automobile(s) are likely to be due for some routine maintenance. When a customer registers a vehicle with Dave's we ask the customer for an estimate of how many miles a year they expect to put on that vehicle. Each time they bring that vehicle in for service; we note the mileage and update the estimated yearly mileage accordingly. Each model and make of vehicle has a set of pre-defined maintenance intervals. Each of those maintenance intervals has a set of services that are called for at that time. For instance, the Toyota Camry may have a 60,000-mile maintenance interval that calls for an oil change, transmission oil and filter change, air cleaner change, rotation of the tires, and so forth. Another model or make may have a different suite of maintenance actions required at a slightly different mileage. When Dave's calculates that one or another of a customer's vehicles is due for preventative maintenance, an E-mail is sent to them telling them the vehicle that is due for maintenance, the mileage that the maintenance is supposed to be done at, the time to expect for the maintenance to take, and the estimate of what that maintenance will cost. The steady customers will then reply to the E-mail indicating which days they would be willing to come in for that maintenance. They get another E-mail confirming the date and time of their appointment.

The premier customers pay an annual fee in monthly installments for their preventative maintenance. The amount of the fee will be a function of the estimate of the number of miles that the customer will put on the vehicle each year, the model of the vehicle, and the make. Dave's assumes that the customer will have that vehicle with us for 5 years or more, so we calculate which maintenance intervals will come up during that 5 year time, the likely cost for the maintenance required at each of those maintenance intervals, and then amortize it over the 5 years so that the customer is essentially purchasing maintenance insurance. If the customer sticks to the schedule and brings their vehicle within 2000 miles of the maintenance interval, and never gets into an accident during their time with Dave's, we will never charge them over the set monthly fee for their maintenance, regardless of any unforeseen maintenance that comes up. To protect ourselves from out of control maintenance costs, Dave's Automotive does not provide premier coverage for any vehicle with over 100,000 miles on it.

Regardless of how many vehicles a given customer has, all of those vehicles are all on the same plan. That means that if a premier customer has a vehicle with over 100,000 miles on it, they will not be able to get that vehicle serviced at Dave's since we do not handle vehicles on the premier plan with over 100,000 miles. Another simplifying assumption that we make is that each vehicle has one and only one owner.

The prospective customers are where the growth occurs. We provide a free oil change to our steady customers or $50 off their next monthly payment to our premier customers if they refer us to someone who is not currently a customer. When they make the referral, we get contact information on their friend/relative from the existing customer, and put it into our database. We also keep track of which of our existing customers made the referral. We will only accept one referral for a given prospective customer. On a periodic basis, we have specials that we have for first time customers, and either send out E-mail or automated phone calls to the prospective customers. We track the date on which these contacts have occurred. If a prospective customer does not become a steady or premier customer after three contacts, we flag that customer as a "dead prospect". In this way, we do not start the whole process of trying to entice them in all over again.

A given customer can be either a private individual or a corporation of some sort. If they are a private individual, we only track one address for them, their mailing address. On the other hand, if they are a corporate entity, we can optionally track several addresses for them. Examples of the types of address that we might track for a given customer is a) mailing, b) billing, c) vehicle pickup, d) vehicle delivery. Each customer will only have one address of a given kind. For instance, Enterprise Rent a Car has only one mailing address as far as Dave's Automotive is concerned. For our corporate customers, we provide free towing if needed. The tow is included in the final bill as a maintenance item, but the price for that particular service for that particular maintenance visit is zeroed out in the final bill.

Each time a customer comes in, one of our service technicians writes up the maintenance visit order. They capture the maintenance items for the maintenance visit, and make an assignment of a mechanic for each of the maintenance items in the maintenance visit. The mileage of the vehicle is logged at the time that it is brought in so that we can update our records of how much the rate at which mileage accumulates on the vehicle.

We track individual skills that the mechanics have so that we can better match them up to a particular maintenance item within a particular service visit. Each maintenance item requires one or more skills. For instance, a maintenance item might be "engine rebuild" which could have skills such as "hoist operation", "head machining", and "ring replacement". It is possible for a mechanic to get assigned to a particular maintenance item that they lack one or more of the necessary skills to perform that maintenance item.

Each mechanic with a given skill is encouraged to mentor another mechanic in that skill. The other mechanic may already have that skill, in which case they are either brushing up on that skill or attempting to achieve greater mastery of that skill. Alternatively, a mechanic may not have a given skill and establish a mentoring relationship with another mechanic to achieve that skill. Either way, only a mechanic with a given skill can mentor another mechanic in that particular skill. When a mentoring relationship starts, we make a formal record of the start of the mentoring relationship. If a given mentee decides to stop the mentoring relationship, we capture that as well. A given mentor/mentee relationship between two mechanics could start and stop multiple times.

Each service visit has a set of maintenance items in it. A maintenance item could be something as simple as changing out the windshield wiper blades to changing the motor mounts, or replacing the struts on the vehicle.

A maintenance **package** is a maintenance item composed of maintenance items. A maintenance package will **not** include other maintenance packages, just individual maintenance **items**. The identifier for a maintenance package is the mileage, the make, and the model. A given service visit could be a combination of a maintenance package as well as any number of additional maintenance items, or it could just be a grab bag of maintenance items. Each maintenance item will have a mechanic assigned to it. Since a maintenance package could be rather complex, we do not assign a single mechanic to the whole package, rather the assignment is done at the individual maintenance item level. Any given service visit will only address the needs of a given vehicle. If a customer brings in more than vehicle at the same time, we create two separate maintenance visits, one for each vehicle. **While it is possible that two maintenance packages could have overlapping maintenance items, just pretend that could never happen.** It turns out that trying to resolve such a case is rather more complex than we want to delve into for this particular assignment, so we will make a somewhat artificial simplifying assumption for now.

The steady customers receive a customer loyalty point for every 10 dollars that they spend with us. The customer can use loyalty points to pay for several of the more common maintenance items such as “ignition tune up”, “oil change”, and “tire rotation”. Every time that a customer spends loyalty points, their loyalty point balance goes down, but they do not earn loyalty points for spending loyalty points.

In order to better manage the cost of the premier package, we track not only how much money comes in from a given premier customer but also the actual cost that they **would** have paid, had they just been steady customers. That way, we can tell whether we are charging them enough.

Additional Business Rules

You will add five business rules to the above business rules. You will have to provide some means to enforce these business rules in your database, either by means of one or more triggers or a database constraint such as referential integrity, uniqueness constraint, not null constraint or the like. The business rule needs to be something that will show up in the model of your design. For instance, your business rule might be that a given mechanic can have no more than three maintenance items going at the same time. None of your additional business rules can contradict any of the business rules provide in the project definition.

Denormalization

Denormalization is a conscious, deliberate change of a design from 3rd normal form to some lower normal form in order to meet some particular objective. To give you some practice at this, I will require that you select some specific place in your model to denormalize. You will describe the denormalization part of phase 1 of the project. Then you will have to explain how you are going to maintain data integrity in spite of the redundancy that you have introduced into the design by the denormalization. Most often, one or more triggers will synchronize the redundant copies of the data.

Please be sure that your UML model depicts a 3rd Normal Form design. The changes to your design to reflect the denormalization that you have selected will appear in the relation scheme diagram.

For the purposes of this project, a denormalization must introduce redundancy of some sort into the physical structure of the dataset. For instance, merging a child table and its parent together and creating a subkey in the resulting table, or creating a multi-valued attribute while maintaining a junction table to represent those values as well would also introduce redundancy into the structure. You will be required to create the necessary triggers to ensure that the redundancies in your structure do not allow conflicting data to be stored.

Project Phases

I have found that students package their deliverables in an apparently endless variety of possible configurations. This makes the grading of these projects much more difficult, particularly if the team makes more than one submission for either of the phases. By giving you the outline of what I expect to see in your drop boxes, I hope that the teams will each structure their deliverables in the same fashion and it will be easy for me to find what I need when it comes time to do the grading.

Phase 1 - Design

1. The description of your five additional business rules. Please call this BusinessRules.docx, txt, …
2. The explanation of your denormalization and how you are going to enforce data integrity in spite of the redundancy of the denormalized structure. Please call this document Denormalization.docx, .txt, …
3. The normalized UML class diagram – either as a DIA model or draw.io. If you have a different tool that you would like to use, please check with me first.
4. English description of all classes and associations. Please call this ClassAndAssociationDefinitions.txt, or docx, or …

Phase 2 - Implementation

The Revised UML diagram

The DDL

- This is all of the create table statements. Please put the primary key constraint and the foreign key constraint into the create table statement. Please call this file create_table.sql.

The Relation Scheme diagram

Call this relation.dia or relation.xml, depending on whether you are using DIA or draw.io.

The Views

Put the DDL for creating the views into a file called create_view.sql.

1. Customer_v – for each customer, indicate his or her name as well as the customer type (prospect, steady or premier) as well as the number of years that customer has been with us.
2. Customer_addresses_v – for each customer, indicate whether they are an individual or a corporate account, and display all of the addresses that we are managing for that customer.
3. Mechanic_mentor_v – reports all of the mentor/mentee relationships at Dave’s, sorted by the name of the mentor, then the name of the mentee.

4. Premier_profits_v – On a year by year basis, show the premier customer's outlay versus what they would have been charged for the services which they received had they merely been steady customers.
5. Prospective_resurrection_v – List all of the prospective customers who have had three or more contacts, and for whom the most recent contact was more than a year ago. They might be ripe for another attempt.

Please perform a select * from each view and put the results of that select into a single file called view_output.sql.

The DML

- The insert statements used to populate the tables. Please call this file insert.sql.

The Queries

Write the SQL to perform the following queries. If it seems to you that it would make the queries easier to write and understand, please feel free to write additional views to facilitate your query writing. Each query is a single SQL statement. **Never** return just the ID of a given thing in your queries, always do any necessary joins so that you can display a proper name. I will dock points for using literals in your queries. For instance, use the now() function to get the current date when asked to find visits within the past year, do not use a literal and put in the due date of the assignment for the current date. Be sure that the sample data that you insert into your tables is adequate to return **some** data from each of these queries:

1. List the customers. For each customer, indicate which category he or she fall into, and his or her contact information. If you have more than one independent categorization of customers, please indicate which category the customer falls into for all of the categorizations.
2. For each service visit, list the total cost to the customer for that visit.
3. List the top three customers in terms of their net spending for the past two years, and the total that they have spent in that period.
4. Find all of the mechanics who have three or more skills.
5. Find all of the mechanics who have three or more skills **in common**.
    a. Please give the name of each of the two mechanics sharing 3 or more skills.
    b. Please make sure that any given pair of mechanics only shows up once.
6. For each maintenance package, list the total cost of the maintenance package, as well as a list of all of the maintenance items within that package.
7. Find all of those mechanics who have one or more maintenance items that they lacked one or more of the necessary skills.
8. List the customers, sorted by the number of loyalty points that they have, from largest to smallest.

9. List the premier customers and the difference between what they have paid in the past year, versus the services that they actually used during that same time. List from the customers with the largest difference to the smallest.
10. Report on the steady customers based on the net profit that we have made from them over the past year, and the dollar amount of that profit, in order from the greatest to the least.
11. List the three premier customers who have paid Dave's Automotive the greatest amount in the past year, and the sum of their payments over that period. Be sure to take into account any discounts that they have earned by referring prospective customers.
12. List the five model, make, and year that have caused the most visits on average to Dave's automotive **per vehicle** in the past three years, along with the average number of visits per vehicle.
13. Find the mechanic who is mentoring the most other mechanics. List the skills that the mechanic is passing along to the other mechanics.
14. Find the three skills that have the fewest mechanics who have those skills.
15. List the employees who are both service technicians as well as mechanics.
16. Three additional queries that demonstrate the five additional business rules. Feel free to create additional views to support these queries if you so desire.

Sample Output from the Queries

Be sure to include sample output in with your queries. The sample output could just be text, or screen captures. Call this file query_output.sql. Please include the SQL for the query immediately before the query output. This makes it immensely easier for me to review the query output.

Description of the attributes

Each attribute in the UML model must have a description for it in Phase 2. Please call this AttributeDescription.docx, .txt, …

Trigger Code

Please provide a listing of your triggers. Please make sure that the triggers are adequately documented so that I know what their purpose is, and how they are achieving that purpose. Finally, have some sample output showing an attempted insertion/update to the data in which the trigger prevents that wayward update to the database, and another demonstration showing the trigger allowing good data in.

Hints

Project Management

This is a big project make no mistake about that.

- Get as much done as you possibly can early on in the time that you have for this project, other classes will be crowding you for time near the end of the semester.
- Make good use of the preliminary phase one turn in. Be as complete in your preliminary design as you can so that I can give you as much feedback as possible.

- It is always possible that my feedback on your preliminary design will cause you to make significant changes to that design. So do not get ahead of the process and start building tables and queries before finalizing the design.
- **Design for the queries**. I have tried to be sure that the first part of the prompt alludes to all of the information that will be in the queries, but take a look at the queries and the views to be sure that your design is not making it unnecessarily difficult to perform the necessary queries or build the necessary views.
- Be **specific** about who on your team performs what and by when. It is even better if you can think of some measurable quality metrics to apply to the deliverables so that everyone on the team can feel secure that each contribution will be of high quality.
- Be sure to test the MySQL environment early:
    - Your individual account
        - You are able to create tables, indexes, and insert data into the tables
        - You are able to create and execute stored procedures, functions and triggers
    - Your team group account
        - You are able to do the same things as the individual accounts
        - That everyone on the team is able to access the team account
    - If you have any issues with the technology, let me know immediately and I'll help you work it out.

Configuration Management

In the business world, there is nothing more deadly to a project than sloppy or spotty configuration management. I will leave it up to the team how they manage the configuration of the various deliverables for this project, but please be certain that the team as a whole is 100% certain that what is going into the project drop box is what the team wants to turn in. I am not going to be sympathetic to the team that tells me the day after a given phase is due that the wrong version of something got turned in by mistake, or that one of the team members turned something in before the rest of the team was entirely ready.

Content

If you are unfamiliar with the automotive world, that need not be a problem. Please check out some service websites such as: http://eurosportautomotive.com/know-your-bmws-recommended-maintenance-schedule/, or http://www.toyotaofsantabarbara.com/yaris-recommended-maintenance-oil-change.htm for some ideas about what typically goes into routine, preventative maintenance.

Technology

Complete this project in MySQL. Nothing else will meet the requirements. Each of you will get an individual MySQL account. Each team gets a group MySQL account as well. All of the team members of a given team will be able to login using their individual account, and then jointly access the team database within MySQL. For that reason, it's important that everyone on the team make sure that you can access the campus MySQL server both through BeachNet+ as well as from home so that you don't all have to be here on campus to work together on this. You will use your individual account for development work. Once you are confident that your work is ready to share, you will run those scripts in the group database so that the rest of the team can benefit from your work. You will **not** be able to grant access to any objects in your personal database to the rest of your team.

Latest Update: 1/11/2017 10:51 AM

David.brown@csulb.edu