CPSC 3660

Project Proposal

By Chad Klassen, Jordan Peoples and Adam Shepley

Preliminary:

The project we wish to create is a Customer Relations Management system.

We believe that such a system inherently offers the perfect level of complexity on both the presentation/front-end side of things, and the back-end database itself and the database design side.

A CRM by nature can be extremely complex or extremely simple or anywhere in between, as the feature set and design is only as thorough as the designers want it to be. That is to say, a CRM will allow us to add more features or remove them to change the complexity of the project, should there be a need.

In this regard, if our design looks like the scale of the project is too large, we could scale it down to a smaller size; similarily, this could work to covert a small scale project to a larger scale one. Our preliminary research on CRM design shows that no real standard or model exists when it comes to creating one, which allows us to be creative and flexible.

Project Specifics

Our project will entail a actual database, as well as the front end for user to interact with the database to track the relations and relational data between a fictional company and it's outside affiliates (clients) which, in theory, will be divided up into Suppliers, Customer, and Partners. The main focus will be our client base, which will contain details about our individual clients. Things like name, sex, age, what organization they are with, a general summary of who they are, and what relationship they have to our company(listed above), will be stored.

The clients organization field will point to another record in our Organization table. This table will store organization ID, name, and a general summary of their workings and interaction with us. This table is only for superficial information, as the actual information about out clients will be stored in their own personal record in our clients table.

The next important thing to note, is we will be tracking communication records between our employees and our clients through the use of a Communication Record table. This table will hold records with the following information: an employee number who this communication record “belongs” to, a client id/number whom our employee contacted or was contacted by, a occurrence date, what medium did the communication occur on (text, phone, email, etc...), and a simple breakdown summary of the communication (what it pertained to).

We will have our own employees using this CRM, so we will be implementing a user table, which will store information and Ids of our employees, and what privileged level they are. We will use this privilege level to enforce permissions, and we will use the ID number to track communication records. Our users will be able to look up a client, based on any information they have, or they could pull up a list of all clients and search through them. The employees will also be able to insert a new communication record, a new client record, and a new organization record. Extra functionality here could be looking through the organization table, and pulling up employees that are associate with said organization, as well as pulling up the organizations and just viewing them .

TODO – reword this shit.

At the moment, we've chosen to abstract our three client relationship types, using a Relationship super-type, which instead of pointing to another table for the relationship type field, will just simply be one of three strings: “Supplier”, “Customer”, or “Partner”. If need be, we could expand upon our relationship organization in a few ways. One way would be to add in a Transaction table, which would track past orders, both incoming and outgoing. To restate, this table would hold the purchase history of our company and also the selling history, and to what client and company these came from or went to. Another way would be to expand the relationship type field in our table, back to linking with one of three relationship type tables (supplier, customer, partner tables)

On the last page is a *very* preliminary relational (non extended) diagram to give a basic idea of what this all means.