Project 2: DBMS

Project Design Document

DataBased

Maximiliano Robledo

Michael Mengistu

Joshua Gonzalez

Blaine Britton

Abraham Sanchez

Department of Computer Science and Engineering

Texas A&M University

Date: September 14, 2020

Table of Contents

Executive summary (1 page)

The overall goal of this database we are creating is to address the needs of students in a marketing statistics class at Texas A&M University. In order to achieve this, the database should be capable of receiving queries of features that pertain to various businesses and extrapolating specified data from their entities. In doing this, the database should be both intuitive and highly capable of receiving high volumes of data in a query. Querying should be highly customizable to allow for searches for data in a particular scope. When this is done, querying should prove to be highly effective in getting relevant data that a student can use as input for various statistical calculations.

We designed this in a waterfall fashion, first by figuring out what the deliverable was asking for, and then working on how to implement that. The next thing we did was create the entities required for this database, and the way we developed them was by creating a "host" entity labeled Business along with 9 other "inherited" entities (e.g. BusinessIncome, BusinessBalance etc.).

The implementation of this database will make searching for data that pertains to a particular business attribute orders of magnitude easier. This can allow for statistical computations to be performed much easier, and allow for the discovery of surprising trends in the market. Through the use of this database, many new discoveries that can be made that are easily applicable and relevant to most of the working world. Finding trends for success and failure in businesses is an invaluable piece of information that this database system makes much easier.

1 Introduction

1.1 Purpose of the Project

The Texas A&M University's Mays Business School requires a system that stores and manages relevant data for all businesses, services, and any other entities that may be of interest to the school. The relevancy of entity data is determined by the usefulness deemed by the school's students; Whether the data can be used to answer customer reviews or aid the students with their assignments, for example. The existence of this system will decrease the search time for the data they require and provide localization for their queried data, so future queries from the same entity will be easily accessible after an initial query of that entity. The system will be used by students and course coordinators of the Mays Business School. The distinction between the two different users should be that the course coordinators can add or remove data in order to ensure that the data stored is relevant and accurate. The course coordinators' ability to add and remove data allows the school to maintain the database without the need of an outside party.

1.2 Needs Statement

The client requires a system that students and course coordinators can use to search for business, service, or any other similar type of entity data. The system should be able to distinguish between different types of companies as the data the users need may not be the same for each.

2 High Level Entity Design

These entities all hold relevant attributes for each respective business ID. All business entries in the Business table have a business ID and share their name in the entities below.

Entities:

1. Entity Business()
    - Purpose: Stores general information about a business that should be present for all companies despite the company type.
    - Justification: Need a way to contextualize companies and what they potentially offer
    - System Role: Is the entity that is used as an intermediary to go between the relevant data sets.
2. Entity BusinessIncome()
    - Purpose: Tracks the most important data in a businesses financial income statement
    - Justification: The data contained here can be used in computing important statistics that students may need for class.
    - System Role: This is a subset of entity *business*
3. Entity BusinessBalance()
    - Purpose: Tracks the most important data in a businesses financial balance statement
    - Justification: This data contained here can be used in computing important statistics that students may need for class
    - SystemRole: Is a particular set of data that can be queried along with a identifier
4. Entity BusinessCashFlow()
    - Purpose: Tracks the most important data in a businesses financial cash flow statement
    - Justification: This data contained here can be used in computing important statistics that students may need for class
    - System Role: Is a particular set of data that can be queried along with an identifier (name, type, etc.)

5. Entity BusinessShare()
    - Purpose: Tracks the Businesses share cost at a particular moment in time
    - Justification: Can be used to find correlation between other data points in a business financial statement
    - System Role: Is a particular set of data that can be queried along with an identifier (name, type, etc.)
6. Entity BusinessContactInfo()
    - Purpose: Stores a businesses's contact information
    - Justification: Having every business's contact information is essential to communicate with them properly.
    - System Role: Is a particular set of data that can be queried along with an identifier(name, type, etc.)
7. Entity BusinessAddress()
    - Purpose: to locate business.
    - Justification: will be used to guide customer to business
    - System Role: address
8. Entity BusinessServiceType()
    - Purpose: Has a complete compendium of the business products/services
    - Justification: This information can be used to find correlation with certain financial data by students.
    - System Role: Is a particular set of data that can be queried along with an identifier (name, type, etc.)
9. Entity BusinessEmployee()
    - Purpose: instantiates an employee for a specific business
    - Justification: get a list and a number of employees per business to see the size of it
    - System Role: many employees to one business
10. Entity BusinessChildCompany()
    - Purpose: Lists child companies associated with each company
    - Justification: Try and see how big the company is, and what companies are tied
    - System Role: A set of data

3 Low Level Entity Design

A. Usage:

The entity *Business* is used to store general information common to all businesses. The justification for the business entity is to mimic inheritance-like properties for businesses (e.g. all businesses have a name, but only some businesses keep track of how many books they sell in a week). Each entry in *Business* corresponds to a unique business with its own business ID, which is a primary key that will be shared among all other following entities in order to associate the business with its specific entry or attribute.

The entity *BusinessIncome* is used to store all data that is present on a typical financial income statement for a business. The justification for the business income entity is to store all financial income statement data for a business. The income of any business is associated with a business ID native to the business table. A businesses income information is relevant when calculating attributes in the *BusinessShare* entity.

The entity *BusinessBalance* is used to store all data that is present on a typical financial balance statement for a business. The justification for the business balance entity is to store all financial balance statement data for a business. A business balance corresponds to a yearly financial statement, and each business would theoretically have a single entry for a year.

The entity *BusinessCashFlow* is used to store all data that is present on a typical financial cash flow statement for a business. The justification for the business cash flow entity is to store all financial cash flow statement data for a business. Business cash flow is used to show flow of assets (generally liquid assets)

The entity *BusinessShare* is used to store business share cost at an instance in time. The justification for the business share entity is that it stores share costs of the business. Business share cost at an instant of time for a business.

The entity *BusinessContactInfo* is used to store the business contact information. The justification for the business contact info entity is that all the business contact information is stored. Business contact info is unique among businesses. Queryable by user.

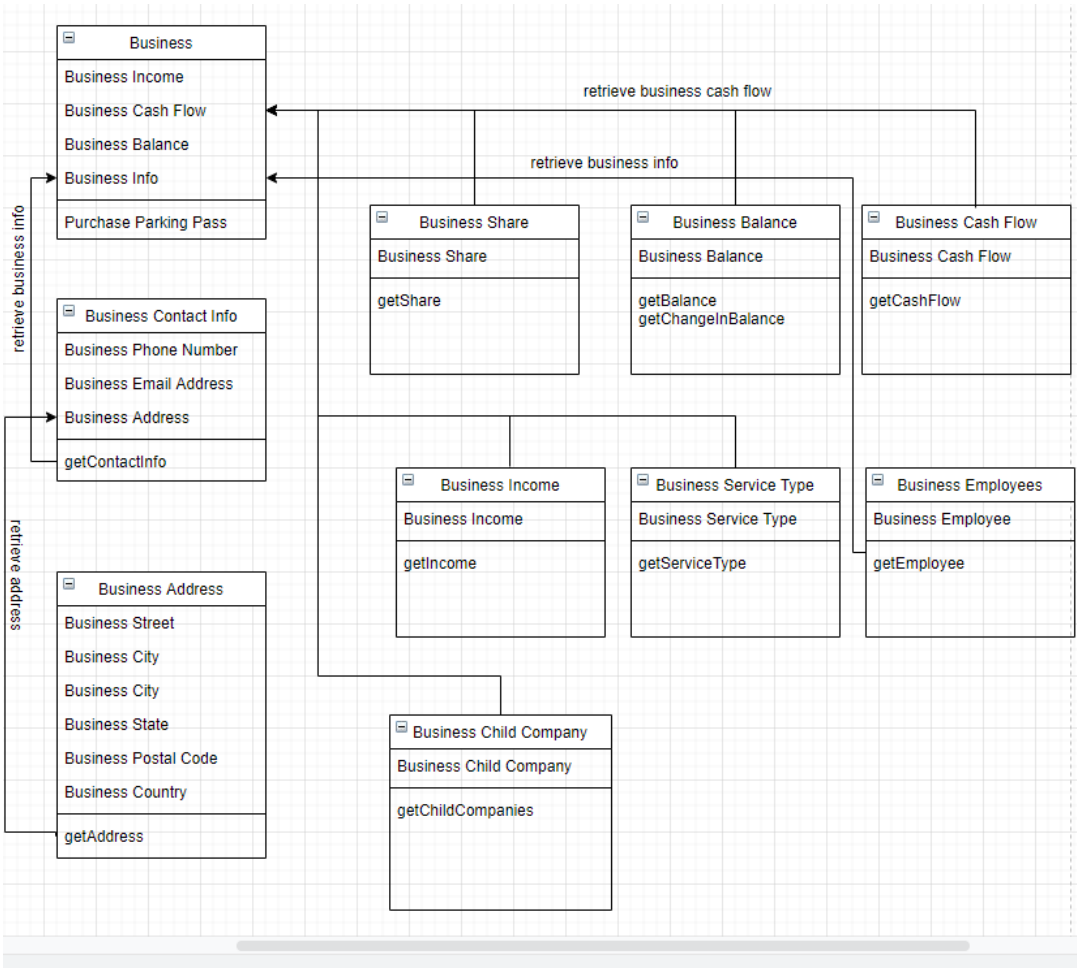
The entity *BusinessAddress* is used to store the business address. The justification for the business address entity is that business addresses are stored for client use. Business address is a subset of business contact info.

The entity *BusinessServiceType* is used to store all service types for a business. The justification for the business product entity is that a list of services for a business is stored for client use. This function can be expanded upon to organize businesses by their service type, which would make querying faster.

The entity *BusinessEmployee* is used to store all employees that belong to a business. The justification for the business employees entity is that a list of all employees for a business is stored for client use. Each *BusinessEmployee* entry has a business ID attribute, and there can be multiple entries with the same business ID.

The entity *BusinessChildCompany* is used to store all child companies to a business. The justification for the business child companies entity is that a list of all child companies for a business is stored for client use. It should be noted that a business child company entry should not have an identical entry in the *business* table to avoid conflicts. Instead, there should be a business ID attribute associated with an entry in the *business* table.

B. Model:



C. Interaction:

We used the UML model as an interaction model by labeling the arrows that are tying each entity together.

4 Expectations

- 4.1. Benefits.

The benefits of implementing this system is that it gives many ways to query and pull data from the database. The identifier need not be as specific as a business name. You can use identifiers such as service\product type in order to pull a multitude of businesses that match this identifier. Additionally, you can specify the type of financial/business specific data you want in a query to pull all of the relevant information you need for statistical calculation or just to inform general information. This would be a great boon to users because it should allow querying for large amounts of information to be intuitive and easy.

- 4.2. Assumptions.

Some of the assumptions that we have made in this database is that each business is going to have their own corresponding entity for relevant financial information. It is also assumed that the relevant information for each business financial information is going to have the same fields. This is not necessarily true in the real world because different businesses have attributes that may or may not apply to others. (Ex: taxes, shares, investments…). Due to this, only the most common attributes that are likely to be applied to all businesses are included, even if exceptions may exist.

- 4.3. Risks.

Possible risks with the current database design is that potentially there can be a lot of empty fields in the case of businesses that do not conform to our underlying assumptions. If you are using some sort of program to query and process data, then this can potentially cause problems for users. Another possible risk is that a user can if they are too unspecific query too much data. The current volume of data that a database is expected to hold is unknown, so trying to query all businesses of a particular type might prove to be too much.

- 4.4. Issues.

There are some non-trivial issues that are caused by our design. When storing large quantities of data, storing every business into a single database table makes searches costly. This is also true when searching for information related to the business such as its income or address, since all businesses share the same table for this information. The time required for each query might become a problem in the future when the number of database entries reaches tens of thousands businesses.

To fix this problem, in the future it might be worthwhile to section off database business entities by some dozen or so sections, such as dividing all food services into a restaurant table, or to associate each business with a business type, which could be expressed as a primary/secondary key or other significant attribute that would make querying faster.