Yale Maps

CPSC 437 Databases Project

Creators:

Carlo Abelli - cfa25 (front end)
Michael Hong - mth43 (front end)
Lan Luo - ll773 (data scraping, coordinates)
Abhi Sivaprasad - ass52 (API spec, back end)

Overview:

This project endeavors to show a map visualization of students at Yale by their home address, while allowing for filtering of students by combinations of the following categories: First Name, Last Name, Class Year, College, Major, Birth Month, and Birth Day. Users of the program will also be allowed to save multiple custom filters for later use. We used advanced clustering algorithms to group together many students located in the same areas to avoid overpopulation.

The data is gathered from the Yale Facebook and only for students who have agreed to release their full student information publicly to Yale students. The data used for the visualization in the project is completely anonymized, with last name removed and replaced by first name and also permuted within each column to make it impossible to identify students by row. Despite the anonymization, any visualizations will still properly display a subset of Yale student addresses.

Our project is organized as follows. Front end uses mapbox API, displaying student home address locations (from the latitude and longitude fields) as dots from the back end. Back end takes filter parameters and queries the SQL database.

Why we chose this topic:

Students from Yale come from many countries; from https://www.yale.edu/about-yale/yale-facts, the report mentions that 118 countries are represented by international students! Although this fact alone is already impressive, there's no better way to visualize and truly appreciate how diverse the student body is at Yale than to map students as dots on a map and physically see how Yalies really do come from every corner of the world.

Why this project is interesting and useful:

This project is interesting, because it offers the only way that we are aware of to visualize from where students at Yale come. The project can be used to express how diverse the Yale student body is and also to visualize potentially interesting facts. For example, by using the filter functionality, you could see if people at Yale who are Mathematics majors and have birthdays in March are more concentrated in some particular area.

Technical Challenges:

The first challenge involved scraping the data from the Yale Facebook website, which organized the data not in a single table with cleanly organized rows and columns, but rather as groups of strings for each student. We downloaded html pages from Yale Facebook by each college and then used the BeautifulSoup4 package in python to organize students into a list by using the package's html parse. Afterwards, we needed to clean the data and organize the string information into columns by using regular expressions.

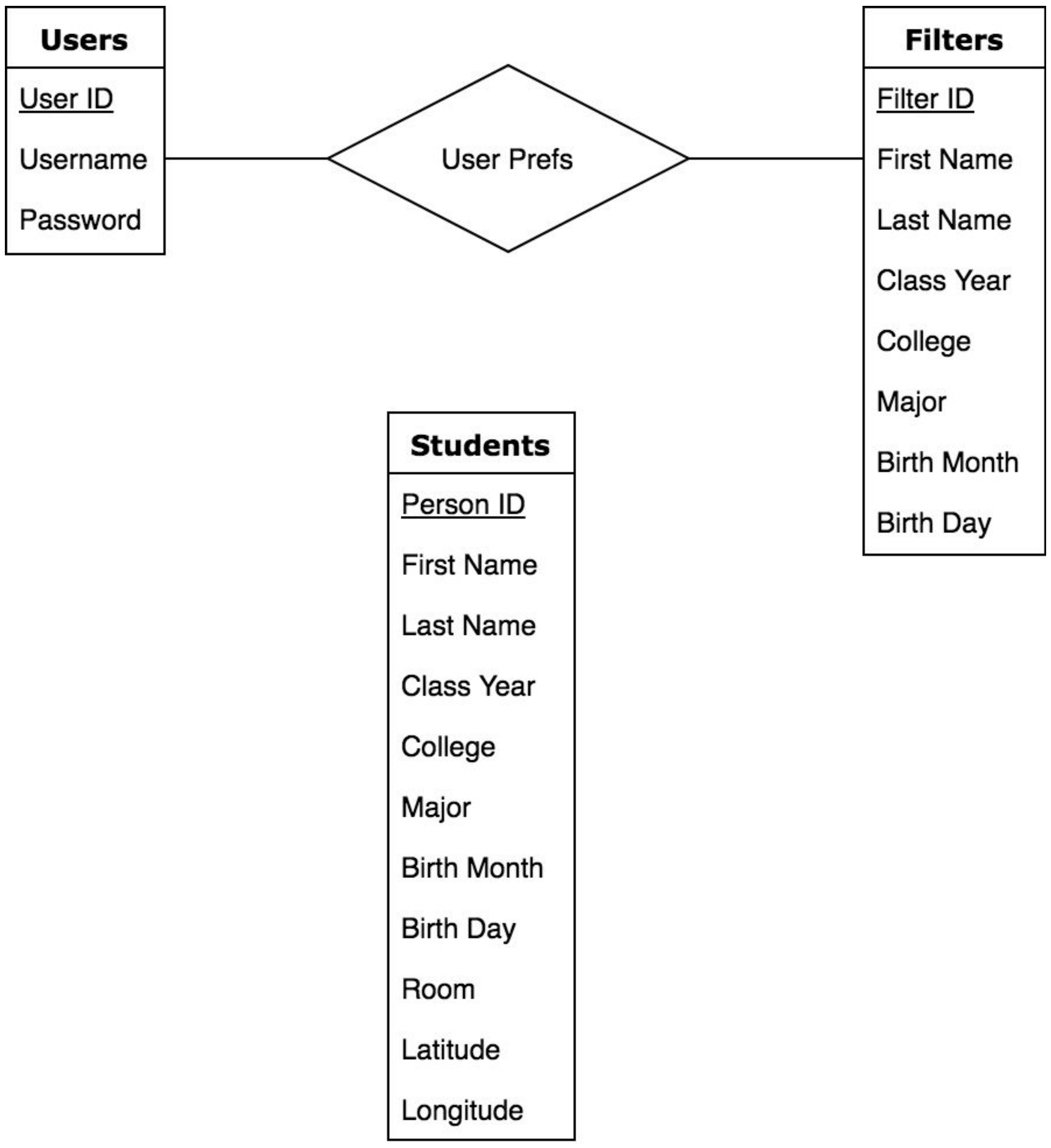
The second challenge involved getting coordinate information from the home addresses. After the students relation was represented as a pandas datatable in python, coordinate information was gathered from the addresses using a combination of the Google Maps geocoder package as well as functionality from the geocod.io website.

The third challenge involved getting around Google Maps limitations, since its geocoder and other requests would time-out and limit the number of requests per day. So, we obtained coordinates separately and put them in the students dataset to avoid continuous and redundant retrieval of information. The visualization uses mapbox rather than Google Maps, which gets around the request limitations.

Schema:

Students(<u>Person ID</u>, First Name, Last Name, Class Year, College, Major, Birth Month, Birth Day, Room, Latitude, Longitude)
Users(<u>User ID</u>, Username, Password)
User Prefs(<u>User ID</u>, <u>Filter ID</u>)
Filters(<u>Filter ID</u>, First Name, Last Name, Class Year, College, Major, Birth Month, Birth Day)

E-R Diagram:


Users
User ID
Username
Password
User Prefs
Filters
Filter ID
First Name
Last Name
Class Year
College
Major
Birth Month
Birth Day
Students
Person ID
First Name
Last Name
Class Year
College
Major
Birth Month
Birth Day
Room
Latitude
Longitude

Normal Form the table meets:

We claim that our database relational schema is in BCNF, and the proof is as follows:

- Users
    - Superkeys: User ID, Username
    - Functional dependencies
        - User ID > Username, Password
            - Each user is identified by a User ID, so Username and Password are both determined uniquely by User ID
        - Username > User ID, Password
            - Likewise for Username
        - Password is not necessarily unique for users (we can only hope that it is), so no functional dependencies
    - Left side of functional dependencies are superkeys, so BCNF
- Filters
    - Superkeys: Filter ID
    - Functional dependencies
        - Filter ID > Filter
            - Each filter uniquely identified by Filter ID
        - Filter attributes will not uniquely identify any other attributes, as each filter can differ by an individual attribute, so no functional dependencies
    - Left side of functional dependencies are superkeys, so BCNF
- Students
    - Superkeys: Person ID
    - Functional dependencies
        - Person ID > First Name, Last Name, Class Year, College, Major, Birth Day, Birth Month, Room, Latitude, Longitude
            - Person ID uniquely identifies all attributes
            - Similar to filter, no group of the attributes for a person will uniquely identify another attribute. Hypothetically, there could be two people with the same name, class year, college, major, birthday, room, lat/long, with only one attribute different. Thus, no functional dependencies
    - Left side of functional dependencies are superkeys, so BCNF

Thus, since each relation is in BCNF, our relational schema is in BCNF.

Most Interesting Factor:

The most interesting factor of our project is the vast combinations of filters that can allow someone to create many different visualizations. Besides the examples already mentioned, another interesting use of this project's code could be to see approximately how many students from your residential college live near you.

Source Code: https://github.com/ericluo04/Databases-Project