Hyunji Seo  
Parth Patel  
CS 327E: Elements of Databases  
April 6, 2015

**Final Project Report**

In order to utilize the yelp academic dataset, we created a general tool that would analyze relational information. When my partner and I were first exposed to the yelp dataset, we were immediately overwhelmed at the enormous size of the data. In fact, most of our efforts of working on the project were spent working on properly loading the data into SQL Developer. We needed to spend a lot of time cleaning the data and frequently asked Yuming for help because there were many incorrect data entries. Common errors include data being entered in the wrong column and messy data that was out of range/inappropriate for the column, such as “0” under number of stars. For our initial proposal, my partner and I had decided to use three yelp tables – USER, REVIEW, and BUSINESS. During the data loading stage, we ran into a lot of trouble with the DATE column in the REVIEW table. We wanted to retrieve the DATE attribute because one of our initial questions that we wanted to answer was to observe the trend of change in given star count over time for a unique user. However, we realized that the format of the DATE attribute changed as it was transferred/imported to SQL Developer. Eventually, we changed almost all of the questions we chose to answer from the initial proposal process to the final SQL queries. Additionally, we used reCSV editor which is an application that allowed us to load large files since Excel would freeze up and became unusable.

When the application runs, the user will first encounter an options menu, listing out 6 different options for the user to choose, as well as a ‘QUIT’ option. This options menu circularly shows up after the user has selected an option and the application has returned the answer. If the user chooses the ***first*** option, the application asks the user to input a city name. Afterwards, the application will return a table of three columns (“Rating | Total Reviews in Range | Percentile of Total”), which displays the distribution of the number of businesses in the user-inputted city according to star rating, as well as a percentile distribution by star rating. If the user chooses the ***second*** option, then the user is requested to input a city name. The application then returns a table of three columns (“Range | Number of Businesses | Percent of Total”), which displays the distribution of number of businesses in the given city according to the range of reviews written for that business, as well as a percentile distribution by each range. If the user chooses the ***third***option, then the user is requested to enter a franchise name. The application then returns two results – the average star rating for that franchise across all cities and the total number of locations for that franchise. If the user chooses the ***fourth***option, then the application automatically returns the grand average star rating across all users. If the user chooses the ***fifth*** option, then the application automatically returns a table of two columns (“Rating Range | Average Review Count”), which displays the average of the number of reviews for each range of star ratings. If the user chooses the ***sixth*** option, then the user is asked to enter a user id, to which the application returns the distribution of average star ratings by year.

In order to verify the solution, we ran a test case for each option. For option 1, we inputted two cities we knew was associated with a lot of businesses, Las Vegas and Phoenix. For option 2, we inputted the same populated cities. For option 3, we entered several popular franchises, such as Starbucks, Walmart, and Taco Bell. For option 4 and 5, we didn’t need to input any data as the results would remain consistent. For option 6, we first chose a user we knew would have reviewed consistently over many years. So, we first looked at the REVIEW table and searched for the user with the most reviews. We used this user to return their average rating distribution by year.