March Project Update

Berkeley Willis

Database Project: March Update, Professor: Dr. Neel

COMP 3115 Section 002, Date: 3/31/2016

**Abstract.** The Fortunes database has been created and populates with several thousand entries using a short Python script. More tables will be inserted into the database to create easily search-able views, along with another Python script that will act as an intermediate application between the database and a user.

1 Introduction

The Fortunes database takes many of the files in the “fortune” program and makes it easy for a user to search for fortunes that pertain to a certain subject, are a quote from a specific person, or possibly to find a specific humorous one. To create this database the fortunes would have to be parsed and inserted into a newly created database, adding tables would make a user's searching experience much easier as well such as quotes from companies, offensive fortunes, and many others. A website with an intermediate application would make the database much more secure and be able to display the information that a user is looking for in a much more readable format.

2 Related work

The database being used is MySQL Server version 5.5.47 on my own machine, and all the scripts used so far have been written in Python 2.7.6. The fortunes files are being used are from the installing files on Ubuntu's packages with more additions to the fortunes, that are also found with Ubuntu's packages. With so many fortunes, and a few with inconsistent formatting, a Python script to interpret and parse them into a form that is uniform to make it easier for inserting into a database had to be created. This lead to the creation of text files of the fortunes, divided based on the three basic tables that will appear in the Fortunes database. These tables are Questions\_And\_Answers, Quotes, and Fortunes. Thus, what was needed next was a short script to take these parsed files and insert them into the my Fortunes database.

3 Design

Instead of having a single Python script to read all of the files parse them into a form that is simple to insert them and to insert them all at once, they have been split into two different scripts. This has been done because there is absence of uniform formatting between all the fortune files, all of the fortunes “types” are mixed together, and verification of correct parsing can easily be done by viewing the results in a text file. The design of the MySQL logical schema is fairly simple to understand.

3.1 Design of the Parsing Python Script. The design of this script creates the resulting three different files, searches for the fortune files using exact paths on my computer, and then reads the file line by line. While reading the fortune files line by line, it uses specific regular expressions to determine what “type” of fortune it is, reads all the lines in the fortune that are left before the start of the next fortune, and then writes it in another format that will be uniform throughout that specific file. The output files that this script creates are the same ones that the insertion script used.

3.2 Design of the MySQL Fortunes Database. The MySQL database was not too difficult to determine after a little help from Dr. Neel. The basic idea is to have three tables that are the three different types of fortunes there are. The Questions\_And\_Answers table holds all of the fortunes that are in a question answer format, and have five different attributes. These attributes are a QA\_ID for identification of each unique fortune, Questions containing all of the questions that appear, Answers containing all of the answer to the questions, Question\_And\_Answer which contains the entire fortune, and subject which one may or may not have but can be used for identification of many of a certain type such as “Fortune's Real-Life Courtroom Quote”. There is also a Quotes table containing all quotes that have three attributes. Quotes have a Q\_ID for identification of unique quotes, the Quotee for showing where the quote came from, and Quote containing the actual quote. The last table, Fortune, is used as an other table really, only containing an F\_ID for identification of unique fortunes, and Fortune that contains the whole fortune.

3.3 Design of the Insertion Python Script. This insertion Python script took all the previously parsed files and insert them into the database. To accomplish this I had to down load an extra python package called “MySQLdb” that gave me the ability to create a connection to the database, execute MySQL commands, and commit all work to the database or rollback if needed. Insertion was completed with only a possible truncation error, but when viewing the database it could not be seen.

4 Analysis

Using Python to parse the files was a step in the right direction but it is also possible to both parse the files and to insert them into the database. Though it is understandable to be making sure of the data that is going to be inserted into the database has a uniform format. Using MySQL Server does keep the same tool that is being used in the class the same, so all of the commands that will be used would be similar to the ones that would be seen during class.

5 Conclusion

The current state of the project is that the database itself is mostly finished. More tables will be added for user convenience, but the output of the database is difficult to read. The problem with the outputs is how MySQL wants to print things and all the newlines that exist in the fortunes. Besides its few problems the database reacts well enough and quickly enough to queries on specific items, meaning that the major goal of the project being reached that is to say that fortunes can be more easily searched.

6 Future works

The next steps on future work on the project are to add tables to the database for easier searching, an application that will communicate to the database for the user and display it in a more readable format, and possibly make a website that will use the application.

6.1 Python Database Application. The application will be designed to take care of all user interaction with the database. It will use the user's login information, which would be created by a user before hand and given certain privileges by the program as a standard user. When a user has created a database user then a connection can be established to the database via the application. The user then can put the queries directly into the application and the application would give the user the results of the query.

6.2 Website. If the stage for the website is reached it will be built off of simple HTML and Bootstrap. There would be two areas where a user can input a query. The first way would be a simple area that would have drop down boxes to input the tables, small text area where a user can specify what they are looking for and the ability to add more for sub-queries. The other type of input field will be straight text input where a more advance MySQL user can input a straight query. This website would user the previously created Python database application, after it has throughly sanitized input. Every time a user makes a bad query, a bofh execuse would be placed in text on the website.

Acknowledgments. Dr. Neel for help in giving the idea of the basic schema for the Fortunes Database.