Hayley Hansson

Nicole Rifkin

Software Systems Homework 3

For our database design and implementation, we initially decided to work with Mongo DB since we were provided with documentation to help us get started. It was free, somewhat complicated to download but well documented. However, implementation proved to be very difficult, and we immediately ran into problems. We discovered that the make file and associated scripts needed to be moved to the source folder inside of the mongo-c-driver. However, this fixed one error and created another. The file found an error in the file Mongo.h, which was part of the package that we needed to download in order to implement the database. We had not modified this file at all, and decided that it would be a better use of our time to switch to MySQL because there was not enough documentation for us to time-effectively fix the errors that appeared to be internal to the database.

MySQL is open source software and was a free download the could be installed using apt-get in one line. We found an excellent source of documentation for an implementation using a C API, and the supporting documentation for the database was excellent. We ran into some issues using the tutorial with setting up a user and password. However, there was clear documentation on how to create a user and change the password. The password input had to be a string with the password written out, which was not immediately obvious from any of the documentation.

Due to the amount of raw data indexing is not particularly efficient in MySQL. Our idea for a database implementation is a one to one mapping of actors to movies and movies to actors. Therefore, it could be extremely efficient one query at a time, but will be slower for large numbers of queries, for example. In addition, it will take a very long time to load a complete set of data into the database. However, this is not inherent to MySQL, and can be solved given more time to develop a system. Since MySQL is also a relational database, it can handle large queries efficiently if all of the features are taken advantage of. However, there is a very high learning curve for a successful implementation.

Although the C API was extremely helpful, there were a number of errors with the documentation in the MySQL interface for relatively simple instructions. For example, after building a database, the commands USE and SHOW TABLES returned errors, and we were not able to create another user account. All of these features were very clearly explained and documented, but simply did not work according to the documentation.

In order to interface with the IMDB text files, we found some python code in github to parse the lists. It takes the actors, actress’s, or movies list right from the text file and outputs a clean list into an SQL file. Unfortunately, we have, so far, been unable to interface this system with MySQL.

Our current code is still very fragmented. We have a file to create a database, and two functions in c that take in a movie and actor and augment the database. We also have code to parse the imdb database, but have not successfully integrated them together in a make file.

MySQL is the most used open source database. Sites like wikipedia use it to organize their information, and it is well maintained. Therefore, we have confidence that it is reliable.