ABSTRACT

This project is to develop a software that allows users to provide simple statements (of language expression) rather than SQL querying constructs to query a database. Thus on the user interface front a general search box allows users to enter their query statement. The system does a search on the query terms as soon as they are entered by the user thus providing an overview of the most relevant results that match the query received thus far. The ranking of the query results is based on a number of factors which have been discussed at length in this report.

PROJECT REQUIREMENTS

The project requirements were thought out meticulously as they are critical for development of all the other downstream software components of the projects.

Requirements elicitation or gathering the requirements for a project such as ours was in large part done through the problem definition provided to us. However a significant number of requirements and many important ones were understood by extensive use of similar tools which are already in the market by the team members and fellow colleagues. Thus the problem definition was modified and the requirements of the project were dynamic as new features and specifications were added through the course of development. Still the most basic requirements were documented and prioritized. These and other newly added requirements were realized by broadly segregating them into two domains – Functional Requirements and Non Functional Requirements.

Functional requirements primarily document the functionalities and services that the software is expected to provided to the end user. They describe the general behavior of the system in particular situations with given inputs.

The functional requirements satisfied by the said software system are -

* **Provision of a generic search facility to the user.** The generic property of the search enables a user to enter search terms without specifying any columns, tables or any other meta data about the database or the search query.
* **Ability to associate any database or CSV format file to the software.** The software provides a clean interface to the client to associate any file (which is the source of the data) to the software on which the searches are to be made.
* **Relevancy based search results on the given query terms.** This is the primary requirement of the system. The results of a query (which are the tuples in the database) are ranked based on their relevance to the query and on history of previous related searches by other users of the system.
* UI

The Non Functional requirements satisfied are -

* **Quick response times.** The preliminary aim during development has been to reduce the Run Time of the system as much as possible, enhancing the user experience significantly. This is the period after entering the query and before the results start to appear, also referred to as the Response Time of the system. The choice of the programming language, the algorithms and the data structures used in the implementation all have been selected meticulously with an overall effect of optimal time complexity of the system leading to reduced response times.
* **Ability to handle large volumes of data.** The software has been developed such that files of large sizes can be attached to it for search operations. The algorithms and their implementation is irrespective of the volume of data that is being sifted through. Although the size of the data files would have a proportionate effect on the response time of the system.
* **Interoperability of the software system.** The software can be executed on any computer system and source code of the program can be compiled on any system with a C++ compiler. Using the user interface provided with the software package, the user can query a database while the back end could be executed on any other system independent of the client's system.

RESULTS DISCUSSION

The results or the output of the execution of the program is a list of tuples from the database. This list is a ranked list of tuples, where the ranking depends on the query terms entered by the user.

The results were tested for correctness and consistencies after several executions of the program on different inputs. We manually went through several iterations of the program, verifying the viability of the resulting tuples.

The forms of the input queries can be broadly classified as belonging to two types – many answers (query terms which are expected to produce a lot of tuples in the result) and empty answer (query terms which are expected to produce very little or no results from the data).

Many answer queries contain terms which occur frequently in the database. In this case the algorithms rank the many tuples in the order of most relevant to least relevant to the user.

Empty answer queries contain terms which either do not occur in the data or occur very seldom. The software gracefully handles this case giving a blank output and not failing on run time error of any kind. (runtime errors are supposed to be the most common types of errors that may occur in cases as such). We plan to accommodate empty answer queries as a further enhancement of the project.

We also timed the results for different types of queries and concluded that the average time taken by the system was always under half a second, with the exact figure lying around 0.33 seconds. On contemplating these results against those produced by MySQL software