International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 04, April-2018

A Robust Scheme for Keyword Based Search over Relational Databases

Alok Kumar Gupta
(Assistant Professor)
Department of Computer Science & Engineering
BBD Engineering College, Lucknow

Sumit Kumar Mishra
(Assistant Professor)
Department of Computer Science & Engineering
BBD Engineering College, Lucknow

***Abstract*— Keyword - based search is the most popular paradigm opted by traditional search engines where a user can specify a string of keywords and expect to retrieve relevant documents, possibly ranked by their relevance to the query. While searching within relational database systems, user needs to learn SQL and to know the schema of the underlying data to pose simple searches. This is a substantial barrier for casual users, such as users of Web-based information systems. Therefore, there is a need of a system that can eliminate this requirement is needed. To accomplish this task, this paper proposes a novel technique that is based on conventional keyword-based search and this technique is applied over relational databases. User has to give his/her query in the form of keywords like Google to get the desired results from database. The implementation of this query processing scheme gives efficient result.**

Keywords- Keyword Based Search, Relational Database, and Column Information.

1. INTRODUCTION

Keyword search provides a simple yet effective way for the users to query and explore the underlying documents.

The last decade has seen ever expanding adoption of the keyword search technology, and it has become a de facto standard for user interaction on the World Wide Web (WWW). In the recent years, there has been a great deal of research and development activities on extending keyword search capabilities to handle relational data, the dominant form in which business data are stored. One important advantage of keyword search is that it enables users to search for information without having to know complex query languages such as SQL and XQuery or prior knowledge about the structure of the underlying data. Keyword search provides an alternative means of querying relational databases, which is simple to most internet users. It lowers the access barrier for average users.

A vast amount of current data resides in relational databases at enterprises, government agencies, research organizations, and on the PCs of home users. As such, the data is often "locked away," reachable only via SQL query interfaces. To facilitate access to this data, recent work has studied the problem of keyword search over relational databases ([1, 2, 6, 7, 8, 9, and 11]). Such keyword search facilities allow users to query the databases quickly, with no need to know SQL or the database schemas. In addition, keyword search can help discover unexpected answers that are often difficult to obtain via rigid-format SQL queries.

Relational databases store large amounts of data which are queried using structured query languages. A user needs to know the underlying schema and the query language in order to make meaningful ad hoc queries on the data. This is a substantial barrier for casual users, such as users of Web- based information systems. HTML forms can be provided for

predefined queries. For example, a university Web site may provide a form interface to search for faculty and students. Searching for departments would require yet another form, as would search for courses offered. However, creating an interface for each such task is laborious, and is also confusing to users since they must first expend effort finding which form to use. Furthermore, they are not suitable for ad hoc querying or exploratory browsing.

The paper has been organized as follows: Section 2 describes the related work in the area of keyword based search. Section 3 describes the proposed work i.e a new algorithm for search over relational database using no column information and also perform ranking of results. Section 4 presents the implementation of proposed work. Section 5 draws the conclusion and describes the future research.

2. RELATED WORK

Keyword-based search is a well studied problem in the world of text documents and Internet search engines. *Inverted lists* are common data structures used for solving keyword queries [3, 5, 13, 14, and 15]. An interesting post-search activity is the ranking of results [3, 14].

Sanjay et.al [1] proposed a technique which deals with search within a single database as well as multiple databases where for a set of query keywords, System returns all rows (either from single tables, or by joining tables connected by foreign-key joins) such that the each row contains *all* keywords. Enabling such keyword search requires (a) a preprocessing step called *Publish* that enables databases for keyword search by building the symbol table and associated structures, and (b) a *Search* step that gets matching rows from the published databases.

Phyo Thu et.al [4] proposed a technique that deals with online search and offline indexing method to do efficient keyword based search where for a given a set of query keywords, the system returns the rows (either from single tables, or by joining tables connected by foreign-key joins) such that the each row contains all keywords. Enabling such keyword search requires (a) a preprocessing step called Indexing that enables databases for keyword search by building the Index Table and (b) a Search step that retrieves the ranked tuples to the user.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 04, April-2018

Our approach is basically different from these and uses no column information in advance to retrieve the appropriate result. Most major commercial database vendors allow a full text search engine (e.g., [10, 12]) to be invoked while processing SQL (that is extended by specialized predicates). However, such engines cannot by themselves identify matching rows that result from joining multiple stored tables on-the-fly. The approach in [5] addresses the problem of keyword search over XML documents. It parses XML documents to generate and load inverted file information (i.e., a map of values to individual rows) into a relational database.

DataSpot [3] is a commercial system that supports keyword- based searches by extracting the content of the database into a *hyperbase*. Thus, this approach duplicates the content of the database, which makes data integrity and maintenance difficult.

[11] Focused on effective keyword search. They proposed to use information retrieval (IR) ranking technologies for keyword search in relational databases to get more effective results. [11] Also proposed some efficient query-processing algorithms to obtain Top-K results.

3. PROPOSED WORK

In this work, it is assumed that there is no column information we have. We have to get result without using column name to check all possibilities of column within each table in a database. The result set contains the entire tables name along with field- name and its value that contains the query. To show the result there is a need to click on the any result set and user will get corresponding complete row as required result. This work also presents result set of table names in ranked form.

Query Preprocessing:

User is provided with the search box. When he/she fills the query in the form of keywords, this string of keywords is split into tokens that are searched in all existing table of selected database. All the keywords are now get stored into string type array so that search operation can be performed with an individual token passing one by one. There may be possibilities of many more instances of database like check whether table is existing within database or not. If table exist within the database then checking for all existing columns in that table because we don't have any column information in advance. After completing this step apply search operation in this table. Here we are checking for all possibilities of each and every token whether it matches in table or not, if any match found then store that complete row result in table (created) of system PUB-database. This process is repeated for all tokens that are stored in string type array and after that the method is proceeded to next (final) step.

Proposed Algorithm

Algorithm-
SEARCH

Inputs: A query consisting of keywords K1, K2… Kn.

Outputs: All database rows matching all keywords, includin rows derived by joining tables on the fly.

Step 1: Perform query preprocessing (Optional).

Step 2: Store tokens within the token array.

Step 3: Pass the each token one by one to next step.

Step 4: Find all the instances of Database.

Step 5: Find all the name of database. In each instance

While (table !=NULL)

{

(Start with one table and find all existing column name in that) While (@tbl=! Null)

Begin

while
(@col=!Null
) begin
insert into @temptable

select Rowid from @tbl where @col like @searchstring end
end

If any result found then store it in table of PUB-Database.

}

Step 6: Perform optimization of above results on the basis o

no. of updates arises
in above step table.

@temptable - temporary table create run time

Fig 1.Proposed Algorithm

Optimization & Ranking:

This step is an optimization of result set by making a column entry named no. of updates in table1 of PUB-database and set its value by 1 at starting stage. When a result added to this table, an entry is created in the table and corresponding value of added column set by 1.There may be chance of getting more and more result for a single token, so system matches the row- ids of result and delete latest one result of same row-id and increment old result of same row-id by new result row-id by its no. of updates value. So in this way we are getting efficient result with maximum matching keywords in result set. If all the keywords are not matching then minimum matching result is returned.

Advantages

- This implementation gives more efficient results for a given set of keywords that are given by a user, without having any column information in advance.
- This also performs ranking of results on the basis of maximum set of keywords matching with result.
- In SQL if we don't write complete description for a or set of proper answers then it returns no results, but in this implementation if any part of string matches anywhere in any table then it returns all the rows in proper manner.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 04, April-2018

- This algorithm will not be expensive and complex because in case of relational databases numbers of columns belonging to such databases are much less.

4. IMPLEMENTATION AND RESULTS

To implement this method, Dot net(.Net) as a platform, Windows XP as an operating system, VB.Net as a language, SQL SERVER 2008 as database is used. These are followings result snapshots describing overall functionalities of the system:

Since this project will work for any database existing within the system, so we have to select a particular database which may have thousands of tables within it. Therefore following entries should be filled up to connect with a particular Database as shown in fig 2.

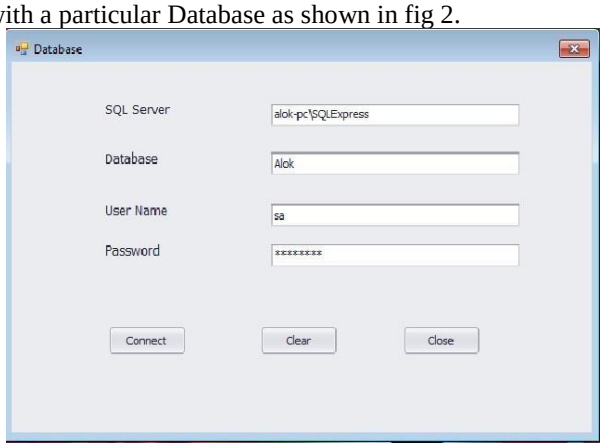

Fig 2. Login details of a database.

- In case of user gives random sequence of keywords as given following snapshot represented, then first result among results view contains all matching keywords of user (if possible).

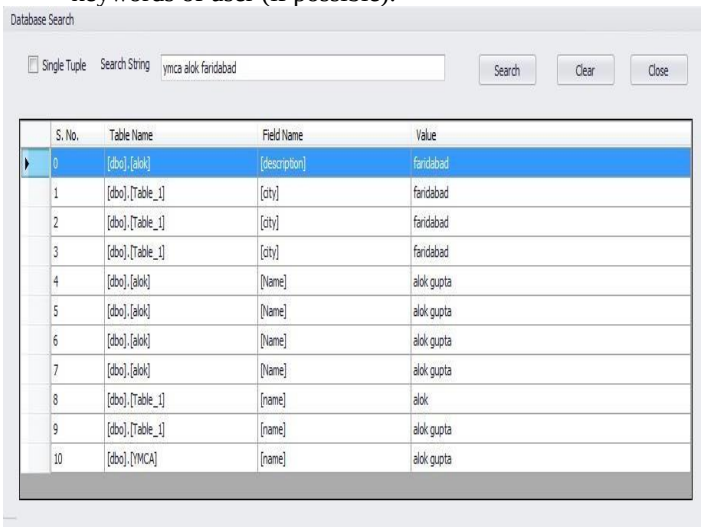

Fig 3.Result view of user query (random sequence of keywords).

- Following fig 4 shows the result of random sequence of keywords entry by a user. Result can be shown in similar fashion by clicking on particular as above mentioned.

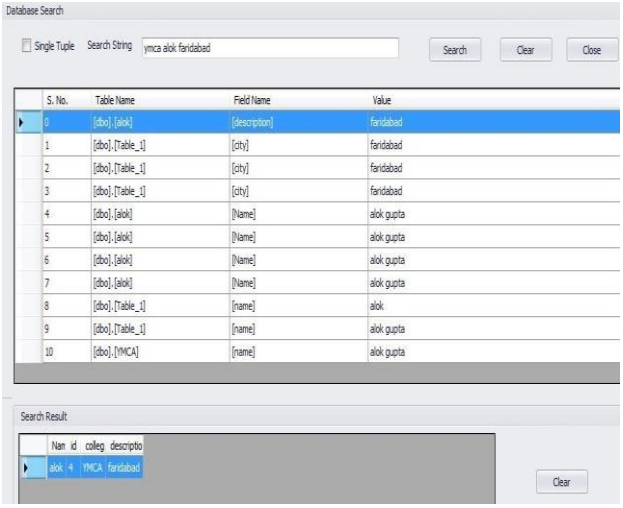

Fig 4.Row returned after selecting a result no.

- In case of proper result not exist in any table of the selected database, proposed system returned the result corresponding to any matching or less matching keywords (from keywords given by user).

5. CONCLUSION AND FUTURE WORK

This paper proposed a technique where keyword based searching is done over relational databases. User has to give a query in a list of keywords, and does not need to specify any relation or attribute names. The main idea is to perform the traditional keyword searching over relational databases and generate all the answers without using any column information. The ranking of results is also done. This proposed technique to search over relational database is not expensive. Results from experiments with this approach indicate that it is successful in always returning a covering combination of answers.

The above proposed scheme can be enhanced in future with:-

- The keyword based searching can be made on even metadata such as table name, column name, store procedure, function name, their constraints & references etc, and instance object.
- Ranking of results can be made more efficient by various techniques such as “no. of hitting” of that keyword.
- Keyword based searching can be done on multiple databases in distributed and network environment.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 04, April-2018

REFERENCES

[1] S Agrawal, S Chaudhuri, G Das: DBXplorer: A system for keyword-based search over relational databases. ICDE 2002.

[2] V. Hristidis and Y. Papakonstantinou, "DISCOVER: Keyword search in relational databases." in *VLDB*, 2002, pp. 670–681.

[3] R. Baeza-Yates, B. Ribeiro-Neto, Modern Information Retrieval, ACM Press, 1999.

[4] Phyo Thu, Htwe Pa, Khin New,SYSTEM: Indexing Relational Databases for Efficient Keyword Search, IJSER, Oct-2011.

[5] D. Florescu, I. Manolescu, Integrating Keyword Search into XML Query Processing, 9th WWW Conf., 2000.

[6] Balmin, V. Hristidis, and Y. Papakonstantinou. Authority-based keyword queries in databases using ObjectRank. In *VLDB-04*.

[7] G. Bhalotia, A. Hulgeri, C. Nakhey, S. Chakrabarti, and S. Sudarshan. Keyword searching and browsing in databases using BANKS. In *ICDE-02*.

[8] V. Kacholia et al, Bidirectional expansion for keyword search on graph databases, In *VLDB-05*.

[9] B. Kimelfeld and Y. Sagiv, Efficient engines for keyword proximity search, In *WebDB-05*. Http://www.microsoft.com/ntserver/web/exec/feature/Index ServerSummary.asp.

[10] V. Hristidis, L. Gravano, Y. Papakonstantinou, Efficient IR-Style Keyword Search over Relational Databases, In VLDB, 2003.

[11] Http://www.microsoft.com/sql/productinfo/fulltext.htm

[12] N. Ziviani, E. Silva de Moura, G. Navarro, R. Baeza- Yates, Compression: A Key for Next Generation Text Retrieval Systems, Computer 33(11): 37-44, 2000.

[13] J. Zobel, A. Moffat, K. Ramamohanarao, Inverted Files versus Signature Files for Text Indexing, ACM TODS, 1998.

[14] S. Melnik, S. Raghavan, B. Yang, H. Garcia-Molina, Building a Full Text Index for the Web, http://dbpubs.stanford.edu/pub/2000-29, 2000.