Proficient Search in Relational Database Based on Keyword

R. Suresh, K. Saranya, S. Dhivya, K. Thilagapriya

Abstract— Information retrieval (IR) is the process of attaining documents relevant to an information need from a great document set. The integration of DB and IR provides flexible ways for users to query information in the same platform. Keyword search is the most popular information retrieval method as users need to know neither a query language nor the underlying structure of the data. In this paper we focus on the keyword query ranking and based on the user preference the top-k results will be retrieved from the relational database and returned to the user. This system will provide an efficient query processing and the ranking functions which is based on the scores calculated by different methods. These scores are the basis for the ranking of the documents.

Index Terms— Candidate network, Information retrieval, Ranking method.

I. INTRODUCTION

The integration of DB and IR provides flexible ways for users to query information in the same platform. On one hand, the sophisticated DB facilities provided by RDBMSs assist users to query well-structured information using SQL. On the other hand, IR techniques allow users to search unstructured information using keywords based on scoring and ranking, and do not need users to understand any database schemas. Information retrieval (IR) is the process of finding documents relevant to an information need from a large document set.IR typically deals with crawling, parsing and indexing document, retrieving documents. Many of the techniques used in data mining come from Information retrieval but data mining goes beyond information retrieval. Metrics often used for the evaluation of the IR techniques are: precision and recall. Recall measures the quantity of results returned by a search and precision is the measure of the quality of the results returned. Recall is the ratio of relevant results returned by all relevant results. Precision is the number of relevant results returned by the total number of results returned. An increase in the precision can lower the overall recall value while an increase in the recall value lowers precision value. searching can be done as phrase based or keyword based. The searching is done for the data based on the keywords that are given by the user in the query.

The Then based on the score calculated for the relevancy the ranking of the data will be done. Keyword searching is an important part of the information retrieval. As the amount of available text data in relational databases is growing rapidly, the need for ordinary users to be able to search such information effectively is increasing dramatically. Keyword search is the most popular information retrieval method as users need to Know neither a query language nor the underlying structure of the data. Keyword search in relational databases has recently emerged as an active research topic. We use the running example as in [1], here in traditional system we need to know the schema structure and give the query in the SQL language which is

```
SELECT * FROM COMPLAINT C
WHERE CONTAINS (C.comments,'maxtor',1)>0
ORDER BY score(1) DESC
```

This prevented many users from using the database and also from retrieving the information from it. Also the schema of the database must be known in advance in order to retrieve the data from the database. It became a tedious process for the novel users to derive the data present in the database and also searching from the database became a time consuming process. But by the integration of the database with the information retrieval we need not require the SQL language to retrieve the information. We can obtain the results by searching from the various relations that is by joining the relations of a database. The relations will be joined by the foreign key. A feature of keyword search over RDBMSs is that search results are obtained by combining relevant tuples from several relations so that they will collectively be relevant to the query. This provides several advantages as (1) Due to normalization, the data might get split and stored in different relations. Those data can be easily retrieved as searching is not limited to a single relation alone.(2) It can help in finding out new or unexpected relationship among the attributes.(3) The user need not have any knowledge about the schema structure and the SQL query language. The rest of the paper is organized as follows: In section 2, we describe about the techniques that were used in the searching and ranking in the existing systems. In section 3, we describe about our proposed system. We discuss about the various scoring functions we used to rank in section 4. The ranking function and the architecture of our system is discussed in section 5 followed by performance evaluation factors discussed in section 6. Section 7 describes the related works and section 8 concludes our paper.

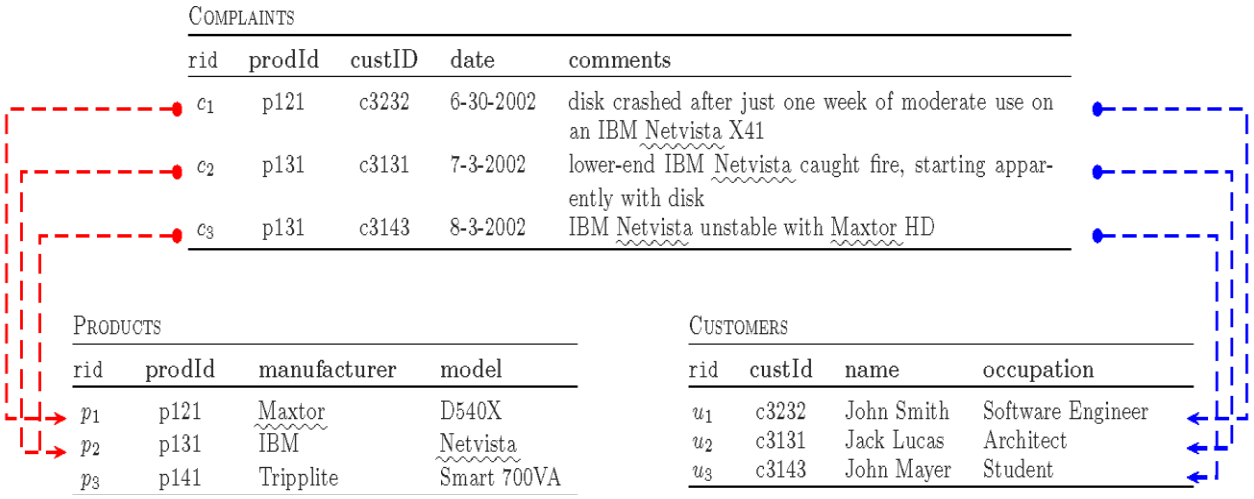
Manuscript received March 31, 2012

R.Suresh, Engineering College affiliated to the Pondicherry university, Puducherry, India, (E-mail: sureshramanujam@gmail.com).

K.Saranya, Engineering College affiliated to the Pondicherry university, Puducherry, India, (E-mail:saranya.kn90@gmail.com**).**

S.Dhivya, Engineering College affiliated to the Pondicherry university, Puducherry, India (E-mail:dhivyasugumar2712@gmail.com**).**

K.Thilagapriya, Engineering College affiliated to the Pondicherry university, Puducherry, India, (E-mail: thilagapriya268@gmail.com).



COMPLAINTS

rid	prodId	custID	date	comments
c_1	p121	c3232	6-30-2002	disk crashed after just one week of moderate use on an IBM Netvista X41
c_2	p131	c3131	7-3-2002	lower-end IBM Netvista caught fire, starting apparently with disk
c_3	p131	c3143	8-3-2002	IBM Netvista unstable with Maxtor HD

PRODUCTS

rid	prodId	manufacturer	model
p_1	p121	Maxtor	D540X
p_2	p131	IBM	Netvista
p_3	p141	Tripplite	Smart 700VA

CUSTOMERS

rid	custId	name	occupation
u_1	c3232	John Smith	Software Engineer
u_2	c3131	Jack Lucas	Architect
u_3	c3143	John Mayer	Student

Fig1:The matches to the query "Maxtor Netvista" are underlined

II. EXISTING SYSTEM

The information retrieval (IR) style keyword search on the web has been a success, keyword search in relational databases have recently emerged. Existing IR strategies are inadequate in ranking relational outputs, so a novel IR ranking strategy for effective keyword search by using a real world database and a set of keyword queries collected by a major search company which is better than existing strategies. The various score calculation techniques mainly used are: Ranked retrieval, Scoring documents, Term frequency, Collection statistics, Weighting schemes and Vector space scoring. The processing of keyword query is done by generating all candidate answers, each of which is a tuple tree by joining tuples from multiple tables. Then compute a single score for each answer. And finally return answers with semantics. In [4] a new concept called Compact Steiner Tree (CSTree), which can be used to approximate the Steiner tree problem for answering top-k keyword queries efficiently and also propose a structure-aware index, together with an effective ranking mechanism for fast, progressive and accurate retrieval of top-k highest ranked CSTrees. BANKS models tuples as nodes in a graph, connected by links induced by foreign key and other relationships. Answers to a query are modeled as rooted trees connecting tuples that match individual keywords in the query. Answers are ranked using a notion of proximity coupled with a notion of prestige of nodes based on inlinks, similar to techniques developed for Web search. A survey of the developments on finding structural information among tuples in an RDB using an l-keyword is given by representing RDB as a data graph GD(V,E). It is called a keyword based approach and is discussed in [4]. The existing algorithms use a parameter to control the maximum size of a structure allowed.

The problem of multiple-query optimization has been discussed. Here a systematic look at the problem, along with the presentation and analysis of algorithms that can be used for multiple-query optimization and presentation of experimental results is provided . Our results show that using multiple-query .These existing approaches are discussed in [1],[2][3],[4].

The candidate networks were used in many of the keyword based search systems and they use the breadth first search to search in the candidate network. There are three rules followed in the candidate network generation such as

Rule 1 Prune duplicate CNs.

Rule 2 Prune non-minimal CNs, i.e., CNs containing atleast one leave node which does not contain a query keyword.

Rule 3 Prune CNs of type: $R^Q \leftarrow S^* \rightarrow R^Q$. The rationale is that any tuple $s \varepsilon^{S^*}$ (S^* may be a free or non-free tuple set) which has a foreign key pointing to a tuple in R^Q must point to the same tuple in R^Q.

III. PROPOSED METHOD

In this paper we focus on the keyword query ranking based on the relevancy score and based on the user preference the top-k results will be retrieved from the relational database and returned to the user. This system will provide an efficient query processing and the ranking functions which is based on the scores calculated by different methods. The score calculation will be dealt in the Section 4. The existing systems have used only one scoring function and when many scoring functions are used, it will lead to more accurate results and the result of the query will be more accurate. The query processing in this paper is done by using the Tree pipeline algorithm which is an enhanced version of the block pipeline algorithm and is better in pruning the empty set. The candidate network will be generated for the generation of the result based on the keyword given by the user. We can specify whether a keyword must be present in the search query or whether it is optional. If the keyword must be searched, then it will use the AND semantics and otherwise it will use the OR semantics. This provides the flexibility to the user. The parameter p can be used to denote it which ranges from values 0 to 1.AND semantics has a value of 1 and the OR semantics takes a value of 0. A keyword can also use any of the values from 0 to1.

IV. SCORING FUNCTION USED

The score is calculated for the purpose of ranking and there are various different techniques that can be used for calculating the scores. Usually the score was calculated based on three factors namely (i) query term weight (ii) term frequency (iii) inverse document frequency. The scoring function are done by the tf-idf, cosine similarity , vector space model and Okapi BM25.

The user will prefer the document which will match with completely all the keywords in the query when compared to the documents which match only partially.

The IR style of ranking will use the IR ranking score as

$$score_a(T,Q) = \sum_{w \in T \cap Q} \frac{1+ln\left(1+ln\left(tf_w(T)\right)\right)}{(1-s)+s.\frac{dl_T}{avdl_{(CN*(T))}}}.ln(idf_w)$$

In SPARK[2] the completeness factor is used for this purpose which is defined as

$$score_b(T,Q) = 1 - \left(\frac{\sum_{i \in sim}(1-T.i)^p}{m}\right)^{\frac{1}{p}}$$

The size normalization factor is also an important factor and defined as

$$score_c = \left(1+s_1-s_1.size(CN)\right).\left(1+s_2-s_2.size(CN^{NF})\right)$$

The final score is computed by the product of all the three scores and is given by

$$score(T,Q) = score_a(T,Q).score_b(T,Q).score_c(T,Q)$$

Based on this final score only the ranking will be done in order to obtain the top-k relevant results that match the requirements of the user.

V.RANKING FUNCTION

Ranking is a major problem in many applications where information retrieval is performed. When documents are retrieved from RDBMS based on a user query then ranking plays an important part as to deliver the documents in a sorted order based on different parameters. Initially the ranking was based on the size of the CN[5].

The use of summary database reduces the number of database probing. The computational sharing is done by which the intermediate results are cached and thereby preventing the search to occur from the scratch.

Later DISCOVER2[6] was developed in which the ranking was based on the scores generated by the state-of-art IR scoring function. The document that are retrieved are ranked purely based on the relevance of the document to the given query.

In BANKS[7], the ranking is based on the hits generated by spanning multiple databases where a graph system is used. These relevance of the documents are obtained by the a score that is calculated by the relevance of one document to the query and also of the pairwise preferences in which two documents are compared and the document more relevant among the two is placed on top of the returned result. The selection of the suitable scoring function is a very important process as the ranking result varies with respect to the different scoring function used. In our paper the scores obtained from different scoring functions are aggregated together to obtain a single value that is used for ranking the final results.

The fig.2 gives you the proposed architecture of our system The user will give the keyword search query in the user interface. The query will be sent to the query evaluation engine where the query will be processed by the generation of the candidate network and the then it is sent to the searcher which will search for the keyword by performing the match. The searched results will be sent to the ranker which will rank the results based on the calculated score and the weight of the terms given by the user. Finally the ranked result will be sent to the user interface for the user to view the result and also the results that are obtained are stored in a summary database. When we query the database with the search query that was already used the system initially checks if similar searches has been performed. And if such search has been performed then the summary database is checked to retrieve the result and integrate it with the updated values in the database

Many different ranking schemes that have the characters of data nodes as node weight, term frequency or inter tuple characters as weight on edges have been used in the existing system. Several advanced features as the phrase based ranking also have been proposed [8], [9], [10], [11], [12], [13], [14], [15], [16], [17].

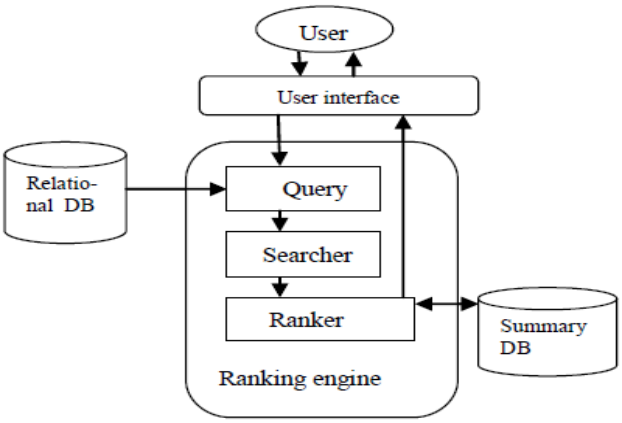


Fig.2 Proposed Architecture

VI. EVALUATING EFFICIENCY

The effectiveness of the results that are returned can be measured by various factors. Usually the precision which is the number of relevant results returned by the total number of results returned is given by

$$precision = \frac{n}{d}$$

Where, n is the number of relevant retrieved documents and d is the total number of retrieved documents. Precision measures among the retrieved documents, how many are relevant. It does not care if we do not retrieve all the relevant documents but penalizes if we retrieve non-relevant documents. The recall is also another factor to measure the effectiveness which tells how good the system is at finding the relevant documents.[3] It is defined as the ratio of relevant results returned by all relevant results and is given by

$$recall = \frac{n}{v}$$

Where n is the number of retrieved documents and v is the total number of relevant retrieved documents.

Recall measures how much of the relevant set of documents we can retrieve.

It does not care if we retrieve non-relevant documents also in the process.The precision and the recall are both dependent on each other. We cannot increase both at the same time because on trying to increase the precision, the recall will get reduced and vice versa.

By these various metrics we can find the effectiveness and how well our system is able to retrieve the relevant results and eliminate the irrelevant results.

VII. RELATED WORK

The main aim of keyword search is to find a top-k interconnected tuples that have high relevance to the query given by the user.

There are various approaches such as modeling of data set as a graph and the results are given in the form of sub graphs is one of the approaches. Most works use the heuristic approach to achieve the required efficiency whereas the steiner tree setback was solved by using an exhaustive search. Various other works include the Q-subtree[11], BANKS[7] and the indexing approach [12],[13] were developed for the purpose of efficiency.

The other approach includes the relational databases in which the structured data are stored. The earlier developed approach is the DBxplorer[18] where query processing is mainly focused. The various techniques of query processing are applied instantly in [5],[6],[7]. Further to the common explanation of keyword search studies have been conducted on identifying the entities which are relevant to the query in an implicit manner. The Rank aggregation method of query processing has also been studied lately which considers challenge of retrieving the k results that acquires the highest score.

VIII. CONCLUSION AND FUTURE WORK

In this paper, we have performed the keyword based search and ranked the retrieved results based on the degree of relevance to the user. The ranking was done using the score. The performance of the system using different metrics was also evaluated. We have performed the searching and ranking of the certain and structured data of the relational databases. In future we would like to extend our searching and ranking to uncertain and probabilistic databases.

REFERENCE

1. http://www.stanford.edu/~maureenh/quals/html/ml/node130.html
2. Yi Luo ,Xuemin Lin, Wei Wang and Xiaofang Zhou "SPARK: Top-k Keyword Query in Relational Databases", SIGMOD'07
3. S. Agrawal, S Databases," Proc. 18th Int'l Conf. Data Eng. (ICDE '02), pp. 5-16, 2002
4. Jeffrey Xu Yu, Lu Qin and Lijun Chang "Keyword Search in Relational Databases: A Survey",IEEE 2010.
5. V. Hristidis and Y. Papakonstantinou, "DISCOVER: Keyword Search in Relational Databases," Proc. Int'l Conf. Very Large DataBases (VLDB), pp. 670-681, 2002.
6. V. Hristidis, L. Gravano, and Y. Papakonstantinou, "Efficient IRStyle Keyword Search over Relational Databases," Proc. 29th Int'l Conf. Very Large Data Bases (VLDB), 2003.
7. G. Bhalotia, A. Hulgeri, C. Nakhe, S. Chakrabarti, and S.Sudarshan, "Keyword Searching and Browsing in Databases Using BANKS,"Proc. 18th Int'l Conf. Data Eng. (ICDE '02), pp. 431-440, 2002.
8. F. Liu, C.T. Yu, W. Meng, and A. Chowdhury, "Effective Keyword Search in Relational Databases," Proc. ACM SIGMOD Int'l Conf. Management of Data, pp. 563-574, 2006.
9. S. Wang, Z. Peng, J. Zhang, L. Qin, S. Wang, J.X. Yu, and B.Ding, "Nuits: A Novel User Interface for Efficient Keyword Search over Databases," Proc. 32nd Int'l Conf. Very Large Data Bases (VLDB), pp. 1143-1146, 2006.
10. B. Ding, J.X. Yu, S. Wang, L. Qin, X. Zhang, and X. Lin, "Finding Top-k Min-Cost Connected Trees in Databases," Proc. IEEE 23rd Int'l Conf. Data Eng. (ICDE), 2007.
11. K. Golenberg, B. Kimelfeld, and Y. Sagiv, "Keyword Proximity Search in Complex Data Graphs," Proc. 28th ACM SIGMOD Int'l Conf. Management of Data, 2008.
12. B.B. Dalvi, M. Kshirsagar, and S. Sudarshan, "Keyword Search on External Memory Data Graphs," Proc. VLDB Endowment, vol. 1,no. 1, pp. 1189-1204, 2008.
13. H. He, H. Wang, J. Yang, and P.S. Yu, "Blinks: Ranked Keyword Searches on Graphs," Proc. ACM SIGMOD Int'l Conf. Management of Data, pp. 305-316, 2007.
14. M. Sayyadan, H. LeKhac, A. Doan, and L. Gravano, "Efficient Keyword Search Across Heterogeneous Relational Databases," Proc. 23rd IEEE Int'l Conf. Data Eng. (ICDE), 2007.
15. S. Tata and G.M. Lohman, "SQAK: Doing More with Keywords," Proc. ACM SIGMOD Int'l Conf. Management of Data, pp. 889-902, 2008.
16. Q.H. Vu, B.C. Ooi, D. Papadias, and A.K.H. Tung, "A Graph Method for Keyword-Based Selection of the Top-k Databases,"Proc. ACM SIGMOD Int'l Conf. Management of Data, 2008.
17. B. Yu, G. Li, K.R. Sollins, and A.K.H. Tung, "Effective Keyword-Based Selection of Relational Databases," Proc. ACM SIGMOD Int'l Conf. Management of Data, pp. 139-150, 2007.
18. S. Agrawal, S. Chaudhuri, and G. Das, "DBXplorer: A System for Keyword-Based Search over Relational Databases," Proc. 18th Int'l Conf. Data Eng. (ICDE '02), pp. 5-16, 2002. Chaudhuri, and G. Das, "DBXplorer: A System for Keyword-Based Search over Relational

AUTHORS PROFILE

He is an Assistant professor of the Department of Information Technology at Sri Manakula Vinayagar Engineering College,Puducherry affliated to the Pondicherry university , Puducherry, India . He has finished his M.tech and working as the Assistant professor and currently pursuing his Ph.D. sureshramanujam@gmail.com.

She is a student pursuing her B.Tech in the Department of Information Technology at Sri Manakula Vinayagar Engineering College affiliated to the Pondicherry university , Puducherry, India . She is doing her research in the field of database management system and about the information retrieval in which ranking plays the major role. saranya.kn90@gmail.com.

She is a student pursuing her B.Tech in the Department of Information Technology at Sri Manakula Vinayagar Engineering College affiliated to the Pondicherry university, Puducherry, India. She is doing her research in the field of database management system and about the information retrieval in which ranking plays the major role. dhivyasugumar2712@gmail.com.

She is a student pursuing her B.Tech Department of Information Technology, Sri Manakula Vinayagar Engineering College affiliated to the Pondicherry university, Puducherry, India . She is doing her research in the field of database management system and about the information retrieval in which ranking plays the major role. thilagapriya268@gmail.com.