ISSN: 2277-9655
(I2OR), Publication Impact Factor: 3.785

IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

ATTRIBUTE RANKING IN QUERYING THE DATABASE

Mr.Prashant S. Chavan*, Prof. Dr.B.D.Phulpagar
* Postgraduate Research Scholar, PESMCOE, Pune, India
Department of Computer engineering, PESMCOE, Pune, India

ABSTRACT

Now a days, databases are with huge number of attributes. To get the info from such type of databases main approaches are query forms. Furthermore static and dynamic query forms have their own advantages and disadvantages. Present paper consider various parameters and find the remedy by designing dynamic form called as Adaptive Query Interface (AQI).

KEYWORDS: Attribute Ranking, Query form, Relations

INTRODUCTION

Modern day databases are huge considering size as well as parameters or attributes therein. Various approaches have been studied till now to retrieve the information from these type of databases. To get the info which approach is to be followed is depend upon the database from where we want the info. Two main approaches we can consider are static and dynamic query interfaces.

Static Query Forms Vs Dynamic Query Forms

Static Query Form:

Static Query Forms is one of the approaches for searching. In this, forms used for searching are fixed. It means that all the attributes have specific sequence which will not change according to the user preference. Thus, in this type of searching where static query forms are used ranking of the attributes is not possible.

Advantages:

- All the attributes are included with fixed sequence.
- No algorithm applied for attribute ranking so in some cases performance may be greater than dynamic query forms.
- Many forms can be generated according to the users.

Disadvantages:

- Since sequence is fixed, no user specific behaviour.
- If the attributes are large in number, difficult to handle.
- For large number of attributes, too many forms are generated.

Dynamic Query Forms:

As the name suggests, dynamic query forms are forms which are subject to change. Now adaptability of the forms completely depends upon the developer approach. But the best suited approach is according to the user behaviour or feedback. It means we can rank the attributes according to the user need or interest.

Advantages:

- User specific ranking of the attributes.
- Can handle large number of attributes.
- Handles ad-hoc queries.

Data is simply as accommodating as its question interface grants it to be. In the event that a client can't pass on to the data what he or she fancies from it, even the wealthiest information store gives next to no or no worth. Static query forms or predefined question shapes range unit used by the DBA to recover the learning from the data. however present utilized databases contain assortment of qualities and relations. Along these lines, recovering data with static query forms is troublesome. Conjointly it's illogical to style static query kind with excessively a few characteristics, making it impossible to handle. A few course instruments offer systems to style predefined question forms. The technique is progressed as an after-effect of client ought to physically alter to style predefined question shapes. On the off chance that a client is unconscious of the data pattern then taking care of qualities inside of the strategy for thinking of predefined question shapes turns out to be excessively best in class, making it impossible to handle.

Adaptive query Interface is a approach that generates program dynamically. In static query forms obtaining desired result's one step method however if the information is big, user find obtaining too several instances of results and therefore desired info is inadequate. Projected approach uses several rounds of actions as inputted by the user to get adaptive query forms dynamically. Since filtration of results is predicated on user actions. Methods are often extended until satisfactory result or satisfactory varieties of filtered results are often found. Projected approach is additionally helpful to the non skilled user. It starts with a basic query type that contains only a few primary attributes of the information. The essential query type is then enriched iteratively via the interactions between the user and our system till the user is satisfied with the query results. In this paper, we have a tendency to chiefly study the ranking of query type elements and also the dynamic generation of query forms. Query forms generated once more are often refined in line with user feedback and dynamically be modified therefore name given as Adaptive Query Interface. Figure one shows the flow diagram of the method

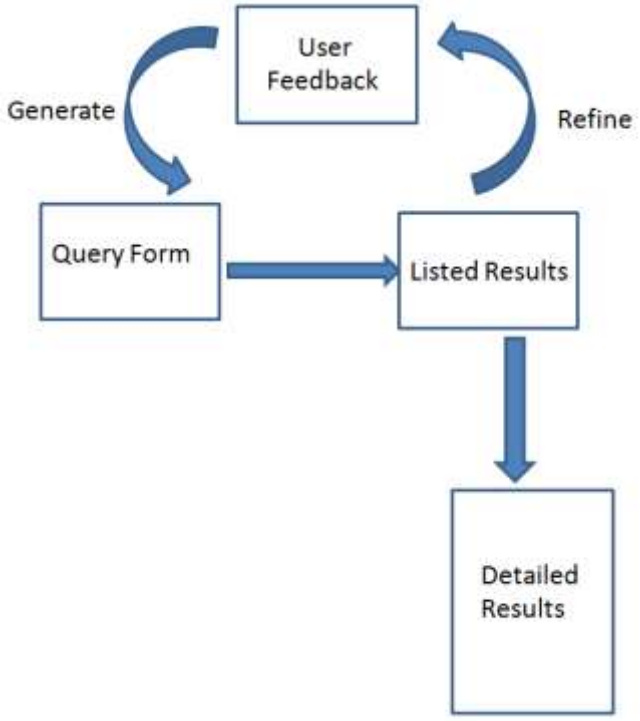

Fig.1. User Participation

LITERATURE SURVEY

Now a days for non-skilled user getting info is difficult. Therefore during this space, chiefly work is concentrated on a way to generate query forms so while not knowing the fields of information schema non skilled user conjointly can

able to fetch info. Presently, query forms area unit accustomed meet this want up to some extent. To spotlight a 1 query by Example is one sort of information querying interface. Existing information purchasers like Microsoft access can also be accustomed give interface to developers to form customise query forms. However to use this tool one have to be compelled to understand the information schema therefore it's helpful to developer and to not user. Paper [3] proposes a system that is automatic approach to get query forms. Here user participation isn't necessary, system initial finds knowledge attributes in schema and consequently generates query forms. though this method having advantage of automatic generation. it's not appropriate for the information schemas having thousands of attributes. If variety of attributes area unit quite kinds that area unit generated area unit too several in numbers and therefore it's confusing for user that form is to be used. Therefore, during this approach end product isn't satisfactory. Paper [5] can also be taken on similar lines as explained.

A structure based inquiry interface is generally the favoured intends to provide an unsophisticated client access to a database. Not just is such an interface simple to utilize, obliging no specialized training, however it moreover it requires next to zero learning of how the information is organized in the database. Be that as it may, form structure is static and can express just a exceptionally restricted arrangement of queries i.e. form is static or fixed. Without space for change, question specification is restricted by the ability and vision of the interface developer at the time the structure was made. On the off chance that an accessible structure can't express a wanted question, then user is stuck.

To overcome downside of said approach paper [1] proposes a system which might be aforementioned as extension of labor [3] and [5]. during this paper they enclosed feature of keyword looking within the generated kinds therefore user currently will realize that form are often used for looking. Therefore system generates ton of query forms beforehand and user then searches the forms with keywords. This method although takes downside from the higher than system its best fitted for information schemas that involves concrete keywords for attributes. However during this system it should be noted that this comes with the disadvantage of knowing the schema beforehand. It means that user should understand the information schema to look desired forms.

IMPLEMENTATION DETAILS

Mathematical Model

Definition 1: A query form F is defined as a tuple (A_F , R_F , σ_F, $\bowtie$ (R_F)), which represents a database query template as follows:

F = (SELECT A_1,A_2, ..., A_k From $\bowtie$ (R_F) WHERE σ_F),
where A_F = {A_1,A_2, ..., A_k} are k attributes for projection and k > 0.

R_F = {R_1,R_2, ...,R_n} is the set of n relations (or entities) involved in this query, n > 0.

Every ascribe in AF has a place with one connection in RF .σF is a conjunction of expressions for determinations (or conditions) on relations in RF. $\bowtie$ (RF) is a join capacity to create a conjunction of expressions for joining relations of RF. In the client interface of an inquiry structure F, AF is the arrangement of sections of the outcome table. σF is the arrangement of info parts for clients to fill. Inquiry shapes permit clients to fill parameters to create distinctive inquiries. RF and $\bowtie$ (RF) are not noticeable in the client interface, which are generally produced by the framework as indicated by the database outline.

For a question structure F, $\bowtie$(RF) is consequently developed by remote keys among relations in RF . In the interim, RF is controlled by AF and σF . RF is the union arrangement of relations which contains no less than one trait of AF or σF . Consequently, the segments of inquiry structure F are really dictated by AF and σF. As we said, just AF and σF are noticeable to the client in the client interface. In this paper, we concentrate on the projection and determination segments of a question structure. Specially appointed join is not took care of by our dynamic question structure since join is not a part of the inquiry shape and is imperceptible for clients.

To decide whether generated query interface is desired or not, it is difficult to decide by checking every instance of the result. This give rise to many answer problem. To address this, only compressed results can be shown with higher level view. Furthermore to get accuracy, user can participate and get results from required category. In first pass of the results since we are targeting to view results in sets, where set means results having same type of results. These results can be clustered by using clustering algorithm [4]. These clustered results can be explored according to the user click through. Figure 2 shows the flowchart of the process.

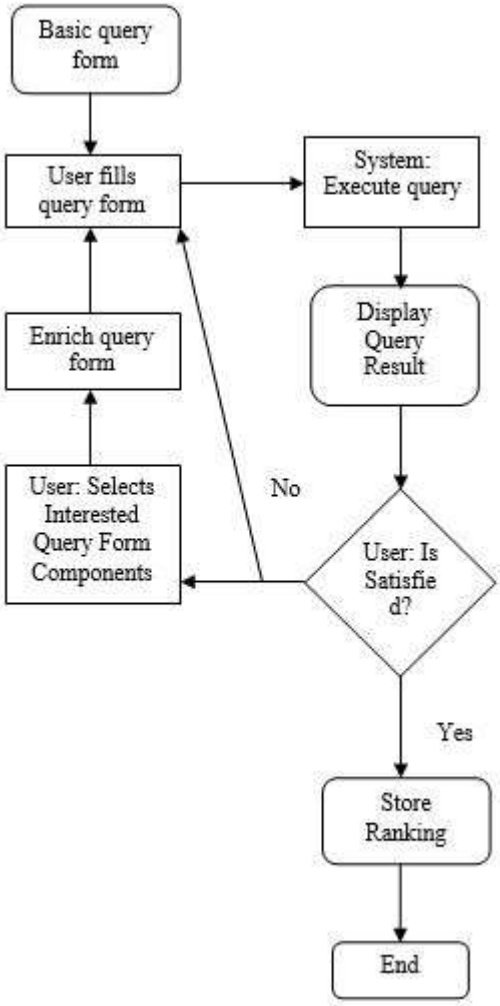

Fig. 2. Flowchart of Adaptive Query Interface

Another imperative use of the packed perspective is to gather the client input. Utilizing the gathered criticism, we can evaluate the decency of a question shape so we could suggest suitable inquiry structure segments. In true, end-clients are hesitant to give express input [15]. The navigate on the compacted view table is an understood criticism to tell our framework which group (or subset) of information cases is fancied by the client. It can offer our framework some assistance with generating suggested structure segments that help clients find more coveted information examples. In some suggestion frameworks and web crawlers, the end-clients are additionally permitted to give the negative input. The negative criticism is an accumulation of the information occurrences that are not coveted by the clients. In the question structure results, we accept The majority of the questioned information occurrences are not fancied by the clients in light of the fact that on the off chance that they are as of now wanted, then the inquiry structure era is verging on done. In this manner, the positive criticism is more instructive than the negative input in the inquiry structure era. Our proposed model can be effectively stretched out for fusing the negative criticism.

Considering both expected accuracy and expected review, we determine the general execution measure, expected F-Measure as appeared in above mathematical statement. Note that β is a steady parameter to control the inclination on expected exactness or expected review.

The framework usage we have chosen to utilize Oracle SQL as back end and ASP.net or JSP as front end. Since it can be conveyed as web interface, the framework is stage free.

DATASET

Implemented system uses student database as input to the system. We have taken all the information related to academics for the students. It contains personal as well as academic information. Whole information is organised in 10 tables comprising 116 attributes. According to the search, instances are projected by combining all the selected attributes.

ALGORITHMIC APPROACH

For projection of query, attributes are selected from various tables. This attributes are refined according to the selection of the attributes by the user. The said approach is taken care of by the query construction

Query Construction:

Data: $Q_f \leftarrow A_1 \cup A_2 \cup A_3 \cup .... \cup A_i$

Result: Q_f is the final query

Begin:

$\sigma_{(one)} \leftarrow \emptyset$

for $Q \in Q$ do

$\sigma_{(one)} \leftarrow \sigma_{(one)} \cup \sigma Q_f$

$A_{(one)} \leftarrow A_F \cup Ar_F$

$Q_{(one)} \leftarrow GenerateQuery(A_{(one)}, \sigma_{(one)})$

$F \leftarrow Project\ (A_{(one)}, \sigma_{(one)})$

Where, Q_f is query form

A_1-A_i: Attributes

$Q_{(one)}$: Representative query

Thus, as above algorithm suggests query selection can be repetitively refined till we get satisfactory results.

DESIGN AND IMPLEMENTATION

In system, two login types are maintained. First one is super admin login where we can add and delete users who are expected users. Also super admin has authority to reset the database.

Fig 3: Admin Login

Second one is for normal users for database searching. As shown in the following form left side pane maintains the list of attributes and the result instances are projected according to the selection. As user selects attributes, in each iteration attributes are ranked according to the user preference.

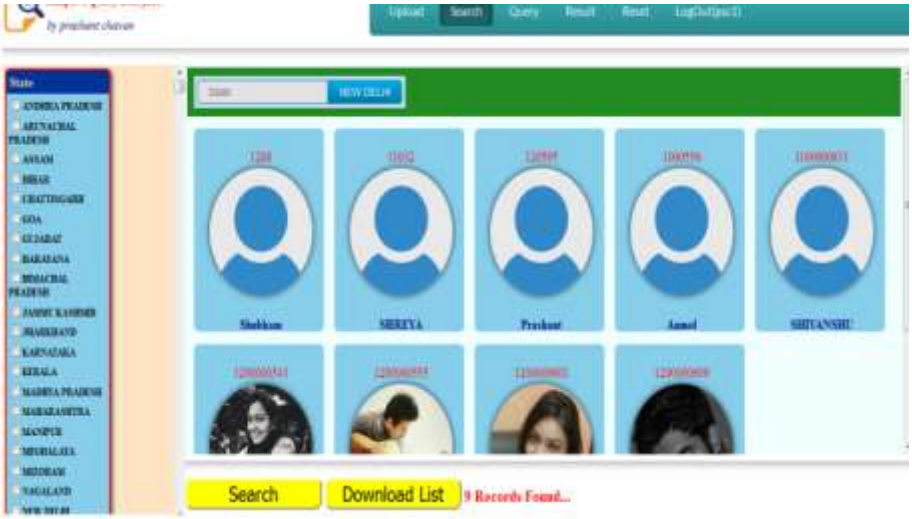

Fig. 4 Query Interface of Searching

After getting result we can get collective information of single instance by clicking on it. As shown in figure below it includes all the information of single student.

Fig. 5 Detailed information of single instance

RESULTS

Result shows adaptive query interface is more effective than that of static query interface. As we can see, as we iterate the searching, accuracy and efficiency of the searching increases. If static interface is consider improvement is not linear as it does not adapt according to the user preference.

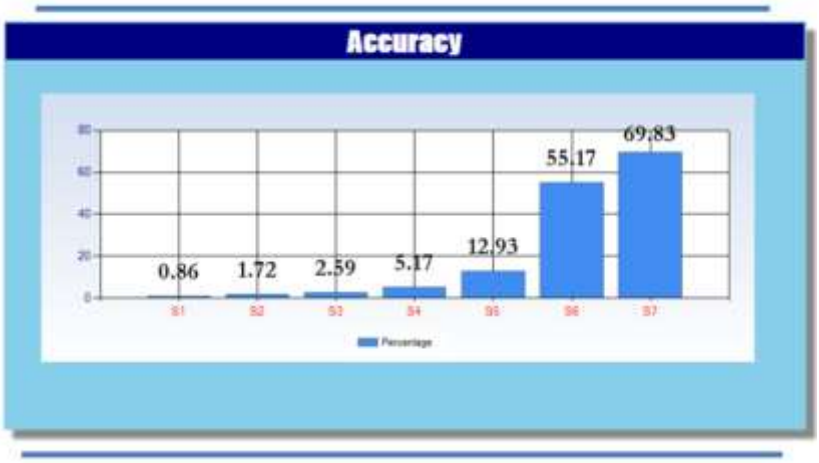

Fig. 6 Incremental improvement in accuracy

This indicates that even though initially accuracy of the searching not up to the mark. Later it increases linearly.

CONCLUSION AND FUTURE SCOPE

In this paper, idea of adaptive query interface is proposed. This system generates query forms dynamically. To capture user feedback run time click through process is used which helps in filtering of the results. From the related concepts studied we can conclude that success rate in this approach will be higher as compared to static approach.

For future scope, as we have used student information as input various algorithm can be applied to find patterns in student performance. Likewise, we may also predict future dropout and failures in student academics.

REFERENCES

[1] Eirinaki M., Abraham S., Polyzotis N., Shaikh N. QueRIE: Collaborative Database Exploration in IEEE Transactions on Knowledge and Data Engineering, (Volume:PP , Issue: 99), May 2013.
[2] E. Chu, A. Baid, X. Chai, A. Doan, and J. F. Naughton. Combining keyword search and forms for ad hoc querying of databases. In Proceedings of ACM SIGMOD Conference, pages 349–360, Providence, Rhode Island, USA, June 2009.
[3] M. Jayapandian and H. V. Jagadish. Automated creation of a forms-based database query interface. In Proceedings of the VLDB Endowment, pages 695–709, August 2008.
[4] C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu. A framework for clustering evolving data streams. In Proceedings of VLDB, pages 81–92, Berlin, Germany, September 2003.
[5] M. Jayapandian and H. V. Jagadish. Automating the design and construction of query forms. IEEE TKDE, 21(10):1389–1402, 2009.
[6] Liang Tang, Tao Li, Yexi Jiang, and Zhiyuan Chen. Dynamic Query Forms for Database Queries,IEEE transaction on knowledge and data engineering, March'2013.
[7] E.Chu, A. Baid, X. Chai, A. Doan, and J. F. Naughton.Combining keyword search and forms for ad hoc querying of databases. In Proceedings of ACM SIGMOD Conference, pages 349–360, Providence, Rhode Island, USA, June 2009.
[8] S. Chaudhuri, G. Das, V. Hristidis, and G. Weikum. Probabilistic information retrieval approach for ranking of database query results. ACM Trans. Database Syst. (TODS), 31(3):1134–1168, 2006.
[9] XMark: An XML Benchmark Project: http://www.xml-benchmark.org/.
[10] Magesh Jayapandian, Adriane Chapman, et al. Michigan Molecular Interactions (MiMI): Putting the Jigsaw Puzzle Together. Nucleic Acids Research (Database Issue), 35, 2007.

[11] Daniele Braga, Alessandro Campi, and Stefano Ceri. XQBE (XQuery By Example): A Visual Interface to the Standard XML Query Language. ACM TODS, 30(2), 2005.
[12] Shishir Gundavaram. CGI Programming on the World Wide Web. O'Reilly, 1996.
[13] D. J. Helm and B. W. Thompson. An Approach for Totally Dynamic Forms Processing in Web-Based Applications. In ICEIS (2), 2001.
[14] Sergey Melnik. Generic Model Management: Concepts and Algorithms. Chapter 7. PhD thesis, University of Leipzig, 2004.
[15] Anders Tornqvist, Chris Nelson, and Mats Johnsson. XML and Objects-The Future for E-Forms on the Web. In WETICE. IEEE Computer Society, 1999.
[16] T. Joachims and F. Radlinski. Search engines that learn from implicit feedback. IEEE Computer (COMPUTER), 40(8):34–40, 2007.
[17] S. Cohen-Boulakia, O. Biton, S. Davidson, and C. Froidevaux.Bioguidesrs: querying multiple sources with a user-centric perspective. Bioinformatics, 23(10):1301–1303, 2007.