International Journal of Emerging Trend in Engineering and Basic Sciences (IJEEBS)
*ISSN (Online)*2349-6967 *Volume 2 , Issue 1(Jan-Feb 2015), PP1-5*

SURVEY ON DYNAMIC QUERY FORMS

Rupali Suroshe[1], Abhijit Patil[2]

[12] *(Computer Department, Pillai HOC college of engineering and technology/ Mumbai university, India)*

Abstract- The modern scientific databases and web databases maintain large and complex data. These real-world databases contain over hundreds or even thousands of relations and attributes. Traditional query forms are designed and pre-defined by developers or DBA in various information management systems. With the rapid development of web database and scientific databases, modern databases become huge and complex. Therefore, it is difficult to design a set of static query forms to satisfy various ad-hoc database queries on those complex databases. At each iteration, our system automatically generates ranking lists of form components and the user then adds the desired form components into the query form based on the captured user preference ranking of form component. A probabilistic model is developed for estimating the goodness of a query form in Dynamic query form. Our system provide the security to the system and make it efficient for private database an encryption technique is applied.**Keywords:** Query form, F-measure, Query form generation

I. INTRODUCTION

Automated information retrieval (IR) systems were originally developed to help manage the huge scientific literature that has developed since the 1940s. Many university, corporate, and public libraries now use IR systems to provide access to books, journals, magazines and other documents. Commercial IR systems offer databases containing millions of documents in myriad subject areas. IR has been found useful in such disparate areas as office automation and software engineering. Indeed, any discipline that relies on documents to do its work could potentially use and benefit from IR. Information retrieval is the activity of obtaining information resources relevant to an information need from collection of information resources.

An IR system must support some basic operations. There must be a way to enter documents into a database, change the documents, and delete the data. There must also be some way to search for documents, and present them to users that are required.

Query form is one of the most extensively used user interfaces for querying databases to access information. Historic query forms are configured and predefined by developers or Database Administrators in different information management systems. With the fast development of web information and scientific databases, new databases become very huge and difficult. In natural sciences, like genomics and diseases, the databases have number of entities for chemical and/or biological data resources. Different types of web databases, like Freebase and DBPedia, have thousands of structured web entities. Therefore, it is difficult to design a set of static query forms to answer various ad-hoc database queries on those difficult and complex databases.

Many existing database management and development tools, like Easy Query, Cold Fusion, SAP and Microsoft Access, provide various mechanisms to let users generate customized queries on databases. But, thecustomized queries generation totally depends on users' manual editing's. If a user is not familiar with the database schema in advance, those thousands of data attributes will confuse him or her.

II. LITERATURE SURVEY

For novice users make use of the relational database is a challenging topic. A lot of research and development works focus on database interfaces which assist users to query the relational database without SQL. QBE (Query-By-Example) and Query Form are two most widely used database querying interfaces. Query forms have been utilized in most real-world business or scientific information systems.

2.1 Customized Query Form:-

Existing database clients and tools make great efforts to help developers design and generate the query forms, such as EasyQuery, SAP, Microsoft Access etc. They provide visual interfaces for developers to create or customize query forms. The drawback of those tools is that, they are provided for the professional developers who are familiar with their databases, not for end-users.

2.2 Automatic Static Query Form: -

Recently, proposed automatic approaches to generate the database query forms without user participation. Many have presented a data-driven method. It first finds a set of data attributes, which are most likely queried based on the database schema and data instances. Then, the query forms are generated based on the selected attributes. Many of them are a workload-driven method. The query forms are then generated based on those representative queries. One problem of these approaches is that, if the database schema is too big and complex, user queries could be quite diverse. In that case, even if we generate lots of query forms in advance, there are still user queries that cannot be satisfied by any one of query forms. Another problem is that, when we generate a large number of query forms, how to let users find an appropriate and desired query form would be challenging.

2.3 Auto completion for Database Queries:-

In many of novel user interfaces have been developed to assist the user to type the database queries based on the query workload, the data distribution and the database schema.

2.4 Query Refinement: -

Query refinement is a common practical technique used by most information retrieval systems. The relationship of IR systems to other information systems is discussed, as is the evaluation of IR systems. But for the database query form, a database query is a structured relational query, not just a set of terms.

2.5 Dynamic Faceted Search: -

Dynamic faceted search is a type of search engines where relevant facts are presented for the users according to their navigation paths. Dynamic faceted search engines are similar to our dynamic query forms if we only consider Selection components in a query. However, besides Selection components, a database query form has other important components, such as Projection components. Projection components control the output of the query form and cannot be ignored. Moreover, designs of Selection and Projection have inherent influences to each other.

2.6 Database Query Recommendation: -

Recent studies introduce collaborative approaches to recommend database query components for database exploration. They treat SQL queries as items in the collaborative filtering approach, and recommend similar queries to related users. However, they do not consider the goodness of the query results.

2.7 Dynamic Data Entry Form:-

During entry, it dynamically adapts the form, which can be dynamically changed according to the previous data input by the end user? Our work is different as we are dealing with database query forms instead of data-entry forms.

2.8 Active Feature Probing: -

S. Zhu, T. Li [12] developed the active featuring probing technique for automatically generating clarification questions to provide appropriate recommendations to users in database search. Different from their work which consider on finding the appropriate questions to ask the user, Dynamic Query Form aims to select appropriate query components.

III. DYNAMIC QUERY FORM SYSTEM

As per our research on Dynamic Query Form system (DQF), a query interface which is capable of dynamically generating query forms for end users is different from traditional document recollection, users in database recollection are often willing to perform many rounds of actions (i.e. refining query conditions) before identifying the final candidates. The core principle of DQF is to capture user interests during user interactions and to adapt the query form iteratively. Each iteration consists of two methods of user interactions which are Query Form Enrichment and Query Execution. Starting of DQF is a basic query form which contains very few primary attributes of the database. The basic idea of query form is then enriched iteratively via the interactions between the user and our systems until the user is satisfied with the results. Our main study is the ranking of query form components and the dynamic generation of query forms.

Interaction between User and DQF:

Query Form:

3.1) DQF supports a ranked list of query form components to the user.

3.2) The user selects the desired form Components into the current query form.

3.3) DQF executes the query and shows the results.

3.4) The user provides the feedback about the query results.

As per our research on dynamic query form system which generates the query forms according to the user wants at run time. The system provides a solution for the query interface in large and complex databases.
We apply F-measure to estimate the goodness of a query form. F-measure is a regular metric to evaluate query results. This metric is also suitable for query forms because query forms are designed to help users query the database. The plus point of a query form is to determined by the query results generated from the query form. Based on this we rank and suggest the potential query form components so that users can filter the query form easily.
Based on the proposed metric, we develop efficient algorithms to estimate the goodness of the projection and selection form components. Here efficiency is essential because DQF is an online system where users often expect quick response

IV. QUERY FORM INTERFACE

4.1 Query Form:

We define a query form F is defined as a tuple (AF , RF , σF , ⋈ (RF)), which represents a database query template as follows:

F = (SELECT A1,A2, ...,Ak

FROM ⋈ (RF) WHERE σF),

where AF ={ A1,A2, ...,Ak} are k attributes for

projection, k > 0. R_F = {R1,R2, ...,Rn} is the set of n relations (or entities) involved in this query, n > 0. Each attribute in AF belongs to one relation in RF. In the user interface of a query form F, AF is the set of columns of the answer table. σF is the set of input components for users to enter. Query forms allow users to enter parameters to generate different queries. R_F and ⋈ (R_F) are not visible in the user interfaces which are generally produce by the system according to the database schema.

For a query form F, ⋈ (R_F) is automatically constructed according to the foreign keys among relations in R_F . Meanwhile, R_F is determined by A_F and σF. R_F is the union set of relations which contains at least one attribute of A_F or σF . Hence, the components of query form F are actually determined by A_F and σF . As we mentioned, only A_F and σF are visible to the user in the user interface. We focus on the projection and selection components of the query form. For example, ”Aggregation” can only be MAX, MIN, AVG, and so on and ”Order by” can only be ”increasing order” and ”decreasing order”. Our dynamic query form can be easily flexible to add those options by implementing them as drop down boxes in the user interface of the query form.

4.2 Query Results:

To decide whether a query form is expected or not, a user does not have any time to go over every instance of data in the query results. In addition many database queries result is a huge amount of data instances. In order to avoid this “Many-Answer” problem we have only output of a compressed result table to show a high-level view of the query results first. Each instance in the compressed table represents a cluster of actual data instances. Then, the user just clicks on particular clusters to view the detailed data instances.

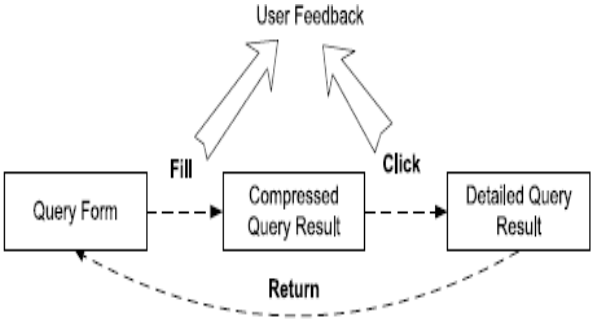

Fig 1. User Actions

Figure 1 shows the flow of user actions. There are many one-pass clustering algorithms for generating the compressed view efficiently. In our study, we choose the data clustering framework because of the efficiency issue.

Certainly, different data clustering methods would have different compressed views for the users. Our system provide a better view of the query results for the user. We can select a different clustering algorithm if needed.

In the query form results, we assume most of the queried data instances are not desired by the users because if they are already desired, then the query form generation is almost done. Therefore, the true or positive feedback is more important than the negative feedback in the query form generation. This proposed model can be easily extended for incorporating the negative feedback.

V. CONCLUSION AND FUTURE WORK

We have studied a dynamic query form generation approach which helps users dynamically create query forms. The idea is to use a probabilistic model to rank form components based on user preferences. We have studied that dynamic query form can be developed by user preference using both historical queries and run-time feedback such as click-through. Also the dynamic approach often leads to the higher success rate and simpler query forms compared with a static approach. Ranking of form components also makes it easier for users to customize query form. As a future work, we can add a text-box for users to input some keywords queries. The relevancy score between the keywords and the query form can be incorporated into the ranking of form components at each step. We plan to develop cache optimization techniques. We can extend our approach can be extended to non-relational data.

REFERENCES

1. Liang Tang, Tao Li, Yexi Jiang, and Zhiyuan Chen "Dynamic Query Forms For Database Queries" IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING VOL:PP NO:99 YEAR 2013DBPedia. http://DBPedia.org.
2. M. Jayapandian and H. V. Jagadish. Automated creation of a forms-based database query interface. In *Proceedings of theVLDB Endowment*, pages 695–709, August 2008.
3. M. Jayapandian and H. V. Jagadish. Expressive query specification rough form customization. In *Proceedings of International Conference on Extending Database Technology (EDBT)*, pages 416–427, Nantes, France, March 2008.
4. M. Jayapandian and H. V. Jagadish. Automating the design and construction of query forms. *IEEE TKDE*, 21(10):1389- 1402, 2009.
5. N. Khoussainova, Y. Kwon, M. Balazinska, and D. Suciu. Snipsuggest: Context-aware autocompletion for sql. *PVLDB*, 4(1):22–33, 2010
6. W. B. Frakes and R. A. Baeza-Yates. *Information Retrieval: Data Structures and Algorithms*. Prentice-Hall, 1992.
7. S. B. Roy, H. Wang, U. Nambiar, G. Das, and M. K. Mohania. Dynacet: Building dynamic faceted search systems over databases. In *Proceedings of ICDE*, pages 1463–1466, Shanghai, China, March 2009.
8. C. Li, N. Yan, S. B. Roy, L Lisham, and G. Das. Facetedpedia: Dynamic generation of query-dependent faceted interfaces for wikipedia. In *Proceedings of WWW*, pages 651–660, Raleigh, North Carolina, USA, April 2010.
9. 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.
10. K. Chen, H. Chen, N. Conway, J. M. Hellerstein, and T. S. Parikh. Usher: Improving data quality with dynamic forms In *Proceedings of ICDE conference*, pages 321–332, Long Beach, California, USA, March 2010.
11. S. Zhu, T. Li, Z. Chen, D. Wang, and Y. Gong. Dynamic active probing of helpdesk databases. *Proc. VLDB Endow.*, 1(1):748–760, Aug. 2008.
12. R. Agrawal, S. Gollapudi, A. Halverson, and S. Ieong. Diversifying search results.