ISSN:0975-9646

A Review paper on Construction of Query Forms

Jayashri M.Jambukar, Prof. **M.B.Vaidya**

Computer Department
Amrutvahini College of Engineering, Sangamner, Pune University,Maharashtra, India

Abstract— **With the very rapid development of various scientific databases and internet information, databases are becoming very gigantic in size and quite complex in nature. These databases feature large and heterogeneous data, with very large number of attributes and relations. So it is very complex to design a set of static question forms to answer various ad-hoc database queries on these day-todays' modern databases. Thus this emphasizes need of such system which generates Query Forms dynamically according to the user's run time needs. The discussed system Dynamic Query Form i.e. DQF system provides a solution by the query interface in large and complex databases. Herein with the proposed system, the core concept and idea is to capture users area of interests throughout various user interactions and to adapt the question type iteratively. Every query form consists of 2 sorts of user interactions: Query Form Enrichment and Query Execution.**

In the Query Form Enrichment DQF recommends a prioritized or ranked list of query form component to user so he/she can select desired form components into current query form. In Query Execution user has to fill current query form and submit query, DQF going to show result and take the feedback from user as well on the provided query results. The user would have choice to fill the query form and submit queries to view the query result during every iteration. So that the query form could be dynamically refined until the user satisfies with the query results.

Keywords—**Query Form, Query Execution, Key Database User Interaction**.

INTRODUCTION

A database is only as functional as query interface allows it to be. When the user is incapable to communicate to the database what he or she desires the results or info from it, even the richest data store provides little or zero significant value to users. Henceforth writing well structured queries, in languages such as SQL and XQuery, could be always challenge due to the number of various reasons, including the user's unfamiliarity with the query language and his or her ignorance of the underlying schema. The form based query interface, requires filling blanks to identify query parameters, is valuable since it helps make data for users with no knowledge of official query languages or the database schema. In actual, form-based interfaces are used recurrently, but generally each form is designed in an Ad-hoc way and its usability is restricted to generally a small set of fixed queries. Query form is one of the commonly used user interfaces for querying databases. Customary query forms were designed and predefined by developers or DBA in various information management systems. Along with the rapid development of web information and various scientific databases, modern databases have already become very gigantic, intricate and complex in nature.

Dynamic question type system: DQF is the question interface capable of dynamically generating query forms for respective users. Different from earlier document retrieval, the user in information retrieval area unit typically wants to perform va rious rounds of actions (i.e., refinement of question, conditions) before distinctive the final candidates. The core of DQF is to capture user interests throughout various user interactions and to adapt the question type iteratively as well. Every iteration consists of two sorts of user interactions: generally it contains only a few primary attributes of the information. The main question type is then enriched iteratively via the interactions between the user and our system untill the user becomes satisfied with the results or outcome of question.

Main goal of this paper is to display the advantages of using dynamic query forms for database over the existing traditional static query forms.

QUERY FORM TECHNIQUES

The normal (non-expert) user use of the relational database as a challenging topic.Various works focus on database interfaces which helps users to query the relational database without SQL. QBE (Query-By-Example) and Query Form are two most extensively used database querying interfaces .The query forms have been used in most real-world business or in scientific information systems..

1. **Query by Example** (**QBE**)

It's a database query language for relational databases. It's the first main query language used to make visual tables where the user can enter commands. Many graphical front-ends for databases use the ideas from query by example today. QBE was earlier limited only for the main purpose of retrieving data; Query by example was later protracted to allow other many operations, like select, insert,delete, updates, forms, create tables etc.

The motivation behind QBE is that a parser can convert the action by users into statements expressed in a database manipulation language, such as SQL. A front-end can help to reduce the burden on the user to remember the importance of SQL, and it is much simple and more productive for end-users (and even programmers) to select tables and columns by selecting them instead of typing in their names. The problem of query by example mainly is relational completeness and the ordering problem.

EXAMPLE FORM

.....Name: Kiran
..Address:
.....City:
....State: MH
..Zip code:
Resulting SQL: SELECT * FROM Contacts WHERE Name='Kiran' AND State='MH'

2. **Automated Ranking of Database Query**

Automated ranking of the results of a query is a widespread aspect of the query model in Information Retrieval (IR) on which we have to become depend on. The database systems backs, a selection query on a SQL database returns all tuples which fulfills the many conditions in the query. The following two scenarios are not tackled by a SQL system:

1. *Empty answers*: If the query is selective, the answers may be empty. In such kind of case, it is required to have the option of requesting ranked list of almost matching tuples without having to mention the ranking function that captures "proximity" to the query. A Government agent or an analyst involved in data exploration can find such functionality appealing.

2. *Many answers*: If the query is not selective, many tuples may be in the answer. In such case, it is desired to have the choice of ordering the matches automatically which ranks more "globally important" answer tuples higher and then returning only the best matches. Customer browsing a product catalogues can find such functionality striking. The main problem of automated ranking of database query result is the ranking functions which might fail to perform as many tuples can tie for same similarity score. It can also arise for empty answer problem.

Query forms are generated based on selected specific attributes. Then apply clustering algorithm on historical queries to find the representative queries. The forms are generated based on those representative queries. The problem of the approaches is, if the database schema is complex and large, user queries could be quite diverse. Even when we generate lots of query forms, there are still user queries that will not be satisfied by any one of query forms. The main problem is, when we generate a large number of query forms, to let the users search an appropriate & anticipated query form can be quite challenging. The solution which combines keyword search along with query form generation proposed in this paper. It automatically creates lot of query forms well in advance. User can give some keywords to find relevant query forms from a large number of regenerated query forms. It functions well in the databases which have rich textual information in data schemas. However, it isn't appropriate if the user does not have concrete keywords to term the queries especially for the numeric attributes.

3. Querying using Instant-Response Interfaces

The problem of searching for information in large and complex databases is always a daunting task. In current database systems, the user has to overcome many challenges. The major challenge is of schema complexity: large organizations can have employee records in different schema, particular to each department and area. The user may not be aware of the exact values of the selection and may put only the partial, similar or misspelled parameter values. To issue queries which are meaningful in terms of result size, like a query listing all employees in the organization may not be helpful to the query user, and will be expensive, complex and time consuming as well for the system to compute. Also we should not expect the user to be proficient in such complex database query languages so as to access the database. The problem about assisted querying using instant response interface is that the user's information need which is mostly explicit .

3. Forms based Database Query Interface

Forms based query interfaces are broadly used to access databases now a days. It is often a key step in the deployment of a database. The form is an interface capable of expressing only a very limited range of queries. The set of forms as a whole should be able to express all possible queries which any user can have. For creating an interface that approaches this ideal is very hard. To maximize the ability of a forms-based interface to support queries, the user can ask when bounding both the number of forms and the complexity of any form. A database schema and content we show an automated technique to generate a good set of forms that fulfils the above criteria. A detailed analysis of actual or expected query workloads are quite useful in designing the interface, the query sets are often not available or difficult to obtain prior to the database even are being deployed. Generating a good set of forms using just the database itself is a challenging but vital problem. The problem of automated creation of forms based on database querying interfaces is use of history log and use association mining for Related Entities.

4. **Form Customization**

A form-based query interface is in general the preferred means to provide access to an unsophisticated user to a database. The interface is easy to use, requiring almost no training, but it also requires little or no knowledge of how the data is structured in the database. A typical form is generally static and can express only a very limited set of queries. Query specification is limited by the expertise and vision of the interface developer at the time the form is created. The modifications are themselves specified through filling forms to create an expression in an underlying form manipulation expression language we define. To modify forms is not much larger than form filling. A form editor is used to implements form manipulation language. A query generator that modifies the form's original query based on a user's changes. The tool provides a powerful means for specifying complex queries. The problem of form customization is limiting the usefulness of form and their somewhat restrictive nature. Existing database clients and tools make great efforts to help developers design and generate the query forms, such as SAP, Microsoft Access, Easy Query, Cold Fusion and so on. The existing databases also provide visual interfaces for developers to create or customize desired query forms. This tool is generally used by the professional developers who are familiar with their databases, but not for end-users a system which allows end-users to customize the existing query form at run time. The end-user perhaps may not be familiar with the database always. If the database is huge and complex, it is difficult for them to look appropriate database entities and attributes and to create the desired query forms.

FEATURES AND PROBLEMS OF QUERY FORM TECHNIQUE

Title	Features	Problems
Query-by example	Provides a simple interface for a user to enter queries.	1) Relational completeness 2) Ordering problem
Automated Ranking of Database Query	To build a generic automated ranking infrastructure for SQL databases.	1) Ranking function might fail to perform
Instant Response Interfaces	Interface developed to assist the user to type the database queries	1) The users information need is explicit 2) Attempt to apply efficient index structures for basic keyword querying
Forms-based Database Query Interface	Automatic approaches to generate the database query forms without user participation	1) Not appropriate when the user does not have concrete keywords to describe the queries.
Form Customization	A system which allows end-users to customize the existing query format run time	1) Database schema is very large so it is difficult to create desired query forms
Forms for Ad Hoc Querying of Databases	Form Based Interfaces and Keyword Search	1) Hard for users, uncomfortable with a formal query language

Difference Between Static query form and Dynamic query form Technique

1. Static query form Technique

Traditional query forms are designed and pre-defined by developers or DBA in various information management systems. With the rapid development of web information and scientific databases, modern databases become very large 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.

system has following limitations :

A.Query forms are designed and pre-defined by developers in information management systems.

B. Difficult to design a set of static query forms to satisfy various ad-hoc database queries on complex databases. The above disadvantages are motivated to design such DQF system which can solve mentioned limitations.

2. Dynamic query form Technique

The proposed a dynamic query form system which generates the query forms according to the user's desire at run time. The system provides a solution for the query interface in large and complex databases. This paper proposes DQF, a novel database query form interface, which is able to dynamically generate query forms. The essence of DQF is to capture a user's preference and rank query form components, assisting him/her to make decisions. The generation of a query form is an iterative process and is guided by the user. At each iteration, the system automatically generates ranking lists of form components and the user then adds the desired form components into the query form. The ranking of form components is based on the captured user preference. A User can also fill the query form and submit queries to view the query result at each iteration. In this way, a query form could be dynamically refined till the user satisfies with the query results.

The proposed system has following advantages:

A.The proposed a dynamic query form generation approach which helps users dynamically generate query forms.

B. The dynamic approach often leads to higher success rate and simpler query forms compared with a static approach.

C.The ranking of form components also makes it easier for users to customize query forms.

CONCLUSION

Query interfaces play an important and complex role in determining the usefulness of the database. A form-based interface is broadly regarded as the most user-friendly querying method available.

In this paper, we have developed the mechanisms to overcome the challenges that limit the usefulness of forms, specially their restrictive nature. Here in this paper we propose the interactive query form generation approach which helps users to dynamically generate query forms.The key idea is to use the probabilistic model to rank form components, based on the user preferences. Ranking of form components also makes it easier for users to customize query form.

ACKNOWLEDGMENT

We dedicate a special thanks to our Computer Department, professors and our college Amrutvahini College of Engineering for the vital support and guidance regarding paper.

REFERENCES

[1] S. Agrawal, S. Chaudhuri, G. Das, and A. Gionis. Automated ranking of database query results. In CIDR, 2003.

[2] 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.

[3] M. Jayapandian and H. V. Jagadish. Expressive query specification through 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] ColdFusion. http://www.adobe.com/products/coldfusion/.

[6] EasyQuery. http://devtools.korzh.com/eq/dotnet/.

Liang Tang, Tao Li, Yexi Jiang, and Zhiyuan Chen, "Dynamic Query Forms for Database Queries," IEEE Trans. On Knowledge and Data Engg. Vol: PP No: 99 Year 2013.