International Journal of
Emerging Research in Management &Technology
ISSN: 2278-9359 (Volume-4, Issue-7)

Research Article

July 2015

Generation of Query Forms Dynamically for Database Queries

Priyanka P. Nikam
Department of Computer Engineering,
Matoshri College of Engineering and Research Center,
Eklahere, Nashik (MS), Savitribai Phule University Pune, India

Abstract—

With speedy development of scientific databases and internet information databases have become terribly large in size and complex in nature. These databases maintain large and diverse data, with large number of relations and attributes. So it is not easy to build a set of static question forms to answer varied ad-hoc database queries on these modern databases. Thus there is requirement of such system which generate Query Forms dynamically as user would like at run time. The Dynamic Query Form i.e.DQF system is proposed in this paper going to provide a solution by the query interface in large and complex databases for both relational and non-relational. In proposed system, the main concept is to take and consider user interests throughout user interactions and to adapt the question type iteratively. Every iteration consists of two types of user interactions: Query Execution and Query Form Enrichment. In Query Form Enrichment proposed system recommends a ranked list of query form component to user so he/she can select desired form components into current query form. User fills current query form and submit query, proposed system shows result and take feedback from user for provided query results, in Query Execution. A user is provided with the facility to fill the query form and submit queries to get and look the query result at each iteration. That enables to develop query form dynamically developed till the user satisfies with the query results.

Keywords— Database, Query Form, Query Execution, User Interaction

I. INTRODUCTION

The query form is a general document that is used for receiving queries from the user. Query form is one of the most widely used user interfaces for querying databases. Filling forms is easy, the user needs only little bit training to learn how to fill a form correctly. In contrast, traditional database querying requires users to be able to write code in SQL (or XQuery) and to also know the database schema[1]. A form-based query interface is usually the preferred means to provide an unsophisticated user access to a database. Not only is such an interface easy to use, requiring no technical training, but it also requires little or no knowledge of how the data is structured in the database. One of the simplest ways to query a database is through a form, where a user can fill in relevant information and obtain desired results by submitting the form. However, a typical form is static and can express only a very limited set of queries.

Traditional query forms are designed and pre-defined by developers or DBA in various information management systems. Modern scientific databases and web databases maintain large and heterogeneous data. These real-world databases contain over hundreds or even thousands of relations and attributes. Many web databases, such as Freebase and DBPedia, typically have thousands of structured web entities. Therefore, it is difficult to design a set of static query forms to satisfy various ad-hoc database queries on those complex databases.

Hence to avoid drawbacks of previous system, we proposed the Dynamic Query Form(DQF) System. DQF system is being used to generate the query forms according to the users need at runtime. The DQF going to capture a users 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 could 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 revised till the user satisfies with the query results. So, the proposed DQF system is useful in every field which contain hug datasets i.e. big data. For Example Government system, science and research sector, health care, employment, economic productivity. Also can be use in private sector, any organisations data can be handled with DQF system.

II. LITERATURE SURVEY

To let non-expert users make use of the relational database is a challenging topic. A lot of research 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. At present, query forms have been utilized in most real-world business or scientific information systems. Current studies and works mainly focus on how to generate the query forms.

SAP, MSAcess provide visual interfaces for developers to create or customize query forms. The problem of those tools is that, they are provided for the professional developers who are familiar with their databases, not for end-users [2]. The proposed a system which allows end-users to customize the existing query form at run time. However, an

© 2015, IJERMT All Rights Reserved

end-user may not be familiar with the database. If the database schema is very large, it is difficult for them to find appropriate database entities and attributes and to create desired query forms.

In automated creation of a form-based database query interface and In expressive query specification through form customization, M. Jayapandian and H.V. Jagadish[3], Here proposed a customized approach to provide visual interface for developers to create or customize query form. EasyQuery, ColdFusion, SAP are the main tools to help developers design and generate the query form. The problem of these tool is that, they are provided for the professional developers who are familiar with their database not for end users. Another problem is that, if the database schema is very large, it is difficult for them to find appropriate database entities, attributes and to create desired query form.

In Assisted querying using instant response interfaces, A.Nandi and H.V. Jagdish[4], This proposed auto completion approaches for database queries. Here a novel user interface have been proposed to assist the user to type the database queries based on query workload, the data distribution and the database schema. In Building dynamic faceted search systems over database, S.B. Roy, H. Nambiar, G. Das and M.K. Mohania[5], a domain independent system, that provides effective minimum effort based dynamic faceted search solution over enterprise database. It present relevant facets for the users according to navigation path.[6] Dynamic faceted search engine are similar to our dynamic query form if we only consider selection components in query.

In automating the design and construction of query forms, M.Jayapandian and H.V. Jagadish[7], Here propose automatic approaches to generate the database query without user participation. It is a work-load driven model. It applies clustering algorithm to find representative queries. One of the disadvantage is that if we generate lots of query forms in advance, there are still user queries that cannot be satisfied by one of the query form.

Recent studies introduce collaborative approaches to recommend database query components for database exploration [6] [7]. 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. [8] proposes a method to recommend an alternative database query based on results of a query. In Usher: Improving Data Quality with dynamic forms, K.Chen, H.Chen, N.Conway, J.M. Hellerstein and T.S. Parikh[9], Data quality is a critical problem in modern database. USHER, an end to end system for form design, entry and data quality assurance. In DQF, deals with database query forms instead of data entry forms.

S. Zhu, T. Li, Z. Chen, D. Wang, and Y. Gong. [10] develop the active featuring probing technique for automatically generating clarification questions to provide appropriate recommendations to users in database search. But it does not focuses on finding the appropriate questions to ask the user, while DQF aims to select appropriate query components.

Till now query forms are designed and pre-defined by developers or DBA in various information management systems but it is difficult to design a set of static query forms to satisfy various ad-hoc database queries on complex databases. No such system able to deal with non-relational database to build query form.

III. PROPOSED SYSTEM

User has to select the relational database on which he/she want to access. After selection of specific table from the desired database, the proposed DQF system rank the attributes based on the existing personalised query log of particular user and generate basic query form. User can select other attributes which are not selected previously. User is also has facility to enter various queries with simplified form. This proposed system preserve the queries which are mostly entered by user as frequent conditions user can also directly access them from the more condition panel provided on query form. After user feedback DQF system enrich the query form based on the selected attributes and the entered conditions. This is an iterative process until user is not satisfied with the results. For accessing non-relational database, user can upload the XML file in the system. After uploading XML file this file is converted into the JSON format. This JSON collection is saved in MongoDb database. Query form is created for this XML file and remaining query execution and ranking attributes is same as for relational database.

Overall algorithm of the system is as follows:

1. Start
2. If user want to upload file containing non-relational data
    a. Upload file
    b. If file type is XML or JSON
    c. Parse data using JSON-XML parser
    d. Save into MongoDb
3. Else user selects database
4. Extract previous queries on selected database
5. Identify frequent attributes and rank queries
6. Generate query using one-query algorithm and execute query
7. Generate Basic query form
8. User selects desire attributes and conditions
9. If user select enrich option get feedback and goto step 5
10. Else Stop.

© 2015, IJERMT All Rights Reserved

Following figure shows the architecture of proposed DQF system.

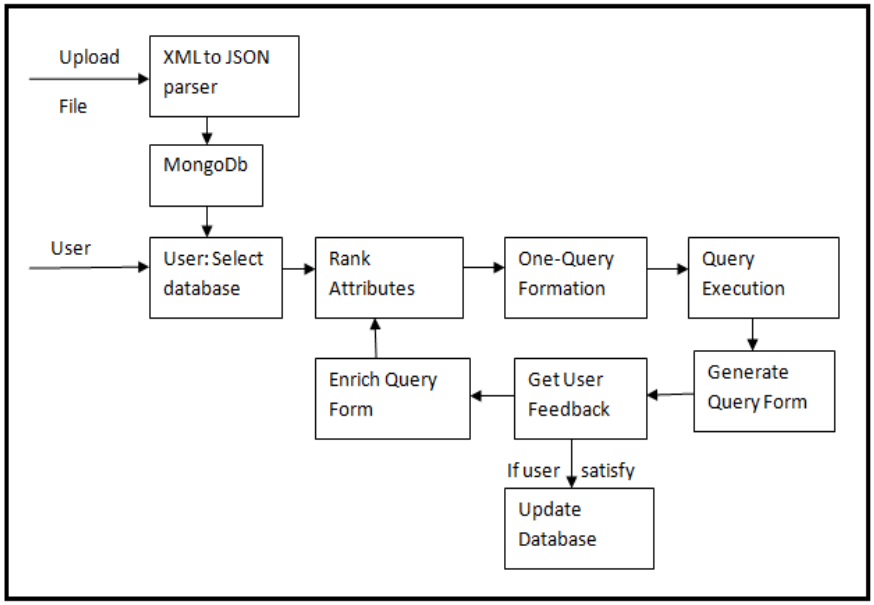

Fig. 1 DQF System Architecture

IV. RESULTS

This project can be implement on any database. It is very efficient to implement proposed DQF system on the database which contain large number of attributes and tables. As basic aim of system is to provide easy handling on the huge database. This system is implemented and tested on following databases:

1. NBA: The National basketball Association (NBA) is the pre-eminent men's professional basketball league in North America, and is widely considered to be the premier men's professional basketball league in the world.
   It contains :
   - Number of Relations : 13
   - Number of Attributes : 205
   - Number of Instances : 45579

2. GeenCar: A green vehicle or environmentally friendly vehicle is a road motor vehicle that produces less harmful impacts to the environment than comparable conventional internal combustion engine vehicles running on gasoline or diesel, or one that uses certain alternative fuels. Environmental Protection Agency(EPA) developed the Green Vehicle Guide to help people find information on vehicles that are more efficient and less polluting. Reducing vehicle emissions and increasing people's fuel economy helps the environment and save money.
   It contains :
   - Number of Relations : 2
   - Number of Attributes : 19
   - Number of Instances : 2204

3. GeoBase: GeoBase is a federal, provincial and territorial government initiative that is over-seen by the Canadian Council on Geomatics (CCOG). It is undertaken to ensure the provision of, and access to, a common, up-to-date and maintained base of quality geospatial data for Canada. Through the GeoBase, users with an interest in geomatics have access to quality geospatial information at no cost and with unrestricted use.
   It contains :
   - Number of Relations : 10
   - Number of Attributes : 36
   - Number of Instances : 1557

A. Personalized Frequent Attributes:

In proposed DQF system frequent attributes are calculated from the queries saved in database with respect to user login. Due to this instead of total attributes present in table only the frequent attributes are provided with checked for ease for query access. Following graphs indicate the number of total and frequent attributes for various tables of various databases. X axis shows all the tables of NBA, GeoBase and Car database and Y-axis shows number of attributes in fig 2,3 and 4 resp.

© 2015, IJERMT All Rights Reserved

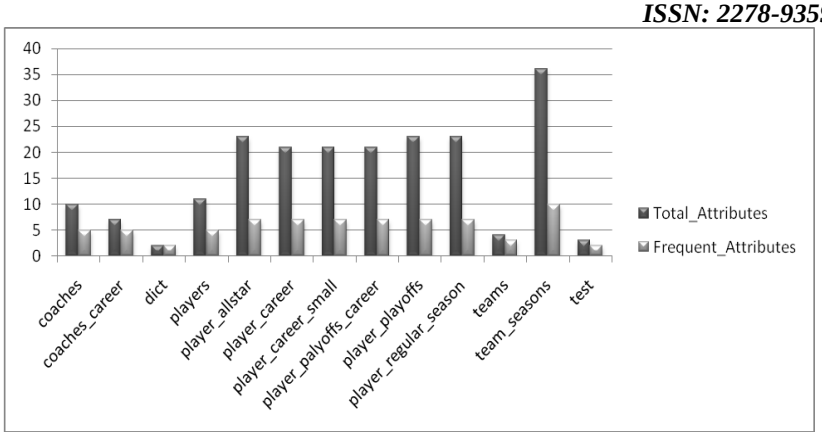

Fig. 2 Graph of number of Total attributes versus Frequent attributes of NBA database

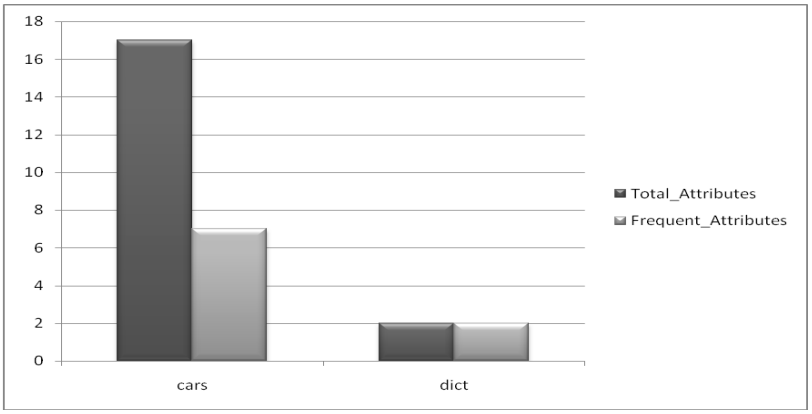

Fig. 3 Graph of number of Total attributes versus Frequent attributes of GreenCar database

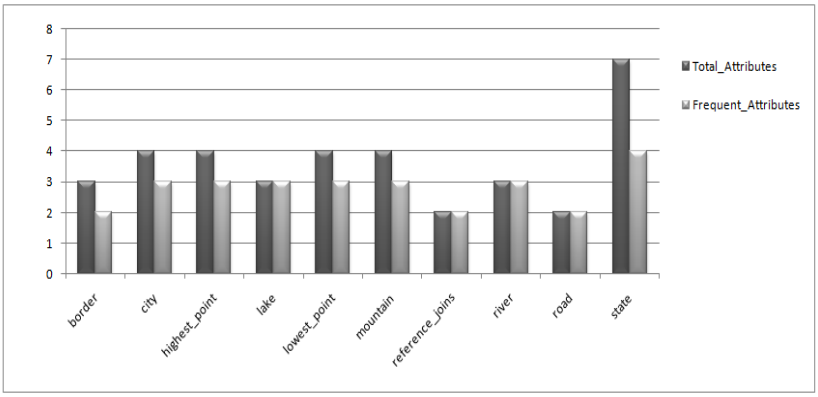

Fig. 4 Graph of number of Total attributes versus Frequent attributes of GeoBase database

B. Frequent Conditions:

In the existing system frequent conditions are shown to every user is same as there is nothing considered like user login or personalisation. But in proposed system the frequent conditions are searched on the basis of queries saved in database with respect to every user not as generalised case. Hence proposed system provide more specific frequent queries than any existing system. Following graph shows comparison based on two tables players and coaches.

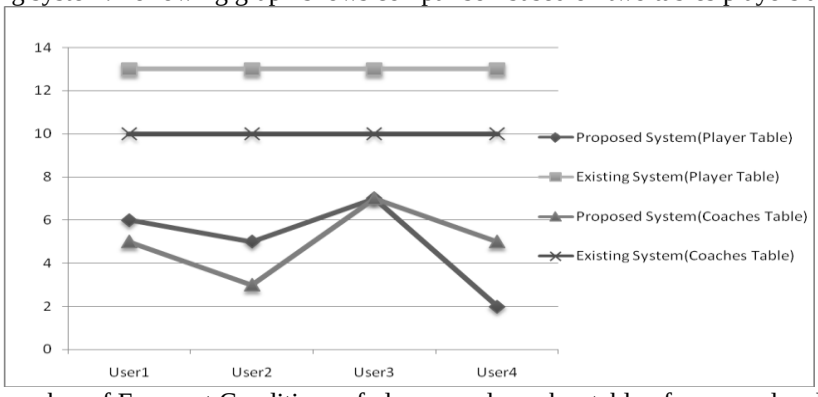

Fig. 5 Graph of number of Frequent Conditions of players and coaches table of proposed and existing system

© 2015, IJERMT All Rights Reserved

C. Effectiveness:

We tested proposed system against the existing system in terms of minimum number of user actions required to get desired query output. One action means a mouse click or a keyboard input for a textbox.

I. Query 1 : SELECT * FROM payers
II. Query 2 : SELECT ilkid, firstname, lastname FROM players
III. Query 3 : SELECT ilkid, firstname, lastname FROM players where position=”G”
IV. Query 4 : SELECT p.ilkid, p.firstname, p.lastname FROM players p, player-playoffs-career c WHERE p.ilkid = c.ilkid AND c.minutes > 5000

TABLE I RESULT TABLE

Query	Generalised DQF (Existing System)	Personalised DQF (Proposed System
Q1	13	4
Q2	6	3
Q3	NA	11
Q4	7	4

Above table shows that the proposed system is more effective than existing one at it requires less number of actions by user to get desired query result. The third query Q3 is not handled by the existing system.

V. CONCLUSIONS

We propose dynamic query form for relational and non-relational database which will help users who are unknown to the database. In this system query form can generate results dynamically, based on user preference, historical queries and runtime feedback. Here F-measure is used to estimate the goodness of query form. F-measure is a typical metric to evaluate query result. The metric is appropriate for query form because query forms are designed to help users query the database. The goodness of query form is determined by the query result generated from query form. Based on this, rank and recommend the query form components so that users can refine the query form easily. We capture user preference using both historical queries and run-time feedback such as click through.

REFERENCES

[1] 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.
[2] Priyanka Nikam,“A review on dynamic query forms for database queries”, In (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5(5) , 2015, 8079-8081
[3] M. Jayapandian and H. V. Jagadish. “Automated creation of a formsbased database query interface”. In Proceedings of the VLDB Endowment, pages 695-709, August 2008
[4] A. Nandi and H. V. Jagadish. “Assisted querying using instant response interfaces”, In Proceedings of ACM SIGMOD, pages 1156-1158, 2007.
[5] M. 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.
[6] 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. (2002)
[7] M. Jayapadian and H.V. Jagdish, “Automating the design and construction of query forms”.IEEE TKDE,21(10):1389-1402,2009.
[8] G. Chatzopoulou, M. Eirinaki, and N. Polyzotis. “Query recommendations for interactive database exploration”. In Proceedings of SSDBM, pages 3-18, New Orleans, LA, USA, June 2009.
[9] 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.
[10] 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.
[11] 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.
[12] 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.

© 2015, IJERMT All Rights Reserved