Asian Journal of Information Technology 15 (22): 4578-4583, 2016
ISSN: 1682-3915
© Medwell Journals, 2016

Search Optimization Using Dynamic Query Forms for Heterogeneous Databases

[1]G. Balakrishnan and [2]M. Indra Devi
[1]Department of Computer Science and Engineering,
KLN College of Information Technology, Pottapalayam, Tamil Nadu, India
[2]Department of Computer Science and Engineering,
Kamaraj College of Engineering and Technology, Virudhunagar, Tamil Nadu, India

Abstract: The development of technical and internet information databases build the huge, complex and various types of heterogeneous data. These real-world databases contain variety of relations and attributes. Query forms which are predefined forms that able to satisfy various informal queries from users on relational databases. We proposed the work intends DQFHD, a novel database query form interface which is able to dynamically generate query forms for heterogeneous databases. The fortitude of DQFHD is to capture a user's preference supporting the user to make conclusion. The query form generation is an iterative process and can be refined by the user by modifying attributes of particular databases. In the each retrieval of information, the system routinely generates ranking lists of form components and the user then adds the preferred 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. In this way, a query form could be dynamically refined till the user satisfies with the query results. This system is designed query forms to support various database queries like relational and XML databases. A probabilistic model is developed for estimating the goodness of a query form in DQFHD.

Key words: Dynamic query form, heterogeneous databases, attributes, query, generation

INTRODUCTION

Query form is one of the most widely used user interfaces for querying databases. Traditional query forms are designed and predefined 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. In natural sciences such as genomics and diseases, the databases have over hundreds of entities for chemical and biological data resources 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.

Many existing database management and development tools such as easy query, cold fusion, SAP and microsoft access, provide several mechanisms to let users create customized queries on databases. However, the creation of customized queries totally depends on users' manual editing. If a user is not familiar with the database schema in advance, those hundreds or thousands of data attributes would confuse the user.

Literature review: The existing solutions are using loading of static forms from the stored forms list (Agrawal *et al*., 2009; Chatzopoulou *et al*., 2009; Jayapandian and Jagadish, 2009; Joachims and Radlinski, 2007; Khoussainova *et al*., 2010). Also, it enables the end users to pose desired query statements that generate reports for decision making (Cohen *et al*., 2007). If any updates are made to the database they are not reflected in the forms which provide the user old forms which are not modified as in the database. If any data is added and user wants to retrieve it, the static forms will not reflect the added data in the database (Aggarwal *et al*., 2003; Jayapandian and Jagadish, 2008a, b, 2009). DQF, a query interface which is capable of dynamically generating query forms for users (Teng *et al*., 2014; Chen *et al*., 2010). Different from traditional document retrieval, users in database retrieval are often willing to perform many rounds of actions (i.e., refining query environment) before identifying the final candidates. It is evaluated the quality of the approach with a user survey on a real database (Chaudhuri *et al*., 2006). At query time, a user with a question to be answered issues standard

Corresponding Author: G. Balakrishnan, Department of Computer Science and Engineering,
KLN College of Information Technology, Pottapalayam, Tamil Nadu, India

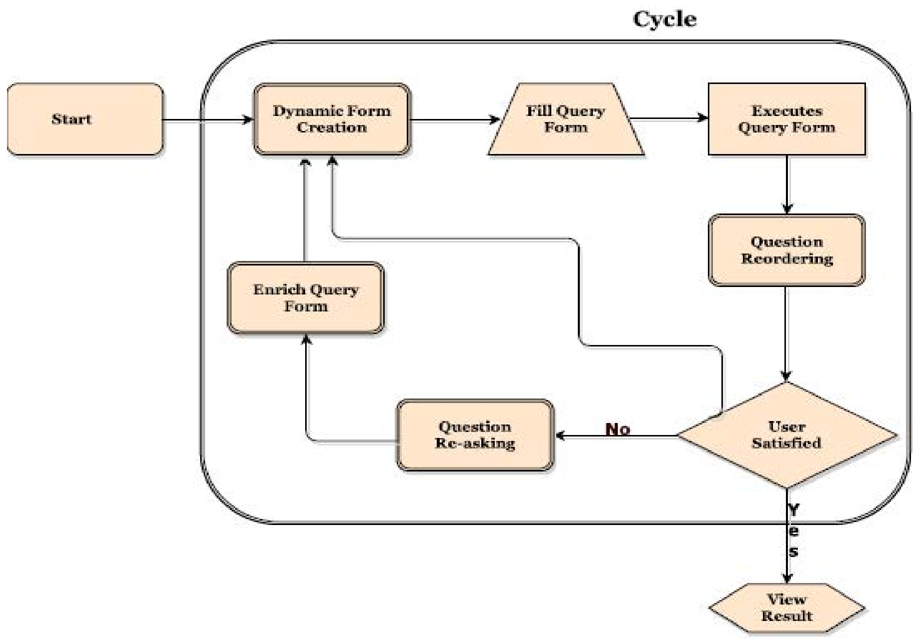


Fig. 1: DQFHD system architecture

keyword search queries (Chu *et al*., 2009; Li *et al*., 2010; Nandi and Jagadish, 2007). The heart of DQF is to capture user interests through user communications and to adapt the query form repeatedly. A crucial component for turning any temporal reasoning system into a real-world application that can be adopted by a wide base of users is given by its user interface (Aggarwal *et al*., 2009; Boriah *et al*., 2008; Rafiei *et al*., 2010; Roy *et al*., 2009). After analyzing and discussing the state of the art for the visualization of temporal intervals and relations, it was proposed three new solutions to the problem of visualizing temporal intervals and their relations for querying databases containing several histories. The metaphors exploited in the proposed visual vocabularies are based on real-world, concrete objects such as strips, springs, weights and wires (Chaudhuri and Das, 2003). A method for mapping queries composed by the visual vocabularies into SQL queries is then described and discussed. There was developed BioGuide to assist scientists search for relevant data within external sources while taking their preferences and strategies into account (Liu and Jagadish, 2009) Context-based similarity measures also used to similar set of customer data (Das and Mannila, 2000; Khoussainova *et al*., 2010).

The visual vocabulary which provided the best results has been adopted in a medical system for visual querying clinical temporal databases. It is also implemented query forms for biological data (Mork *et al*., 2002). Devising a scheme for efficient and scalable querying of Resource Description Framework (RDF) data has been an active area of current research (Khoussainova *et al*., 2010). Some generalized metrics compared to results produced by commercial search engines (Joachims and Radlinski, 2007).

MATERIALS AND METHODS

Proposed system: In this study, we propose a Dynamic Query Form system DQFHD, a query interface which is capable of dynamically generating query forms for users. Different from traditional document retrieval, users in database retrieval are often willing to perform many rounds of actions before identifying the final candidates (Fig. 1). The spirit of DQFHD is to capture user interests during user interactions and to adapt the query form iteratively. In every iteration, it consists of two types of user interactions form enhancement and form implementation. It starts with a basic query form which contains very few major attributes of the database. The basic query form is then enriched iteratively via the interactions between the user and our system until the user is satisfied with the query results. In this study, we mainly study the ranking of query form components and the dynamic generation of query forms. The features of the proposed system are: DQFHD is a query form interface which is able to dynamically generate query forms. The spirit of DQFHD is to capture a user's preference and order query form components. The ordering of query form components makes it easier for users to customize query forms. The generation of a query

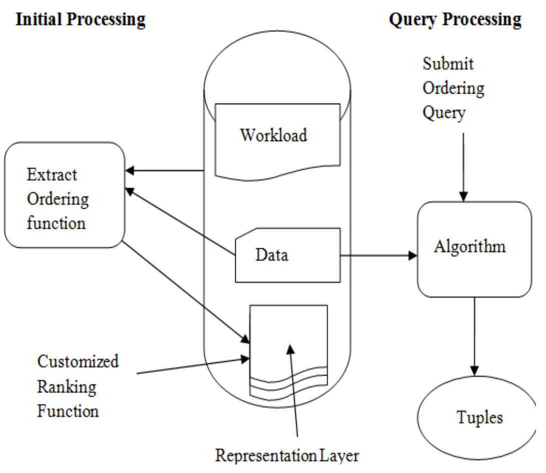


Fig. 2: Architecture of ranking system

form is an iterative process. A query form could be dynamically refined until the user is satisfied with the query results. DQFHD is extended to process Non Relational database queries.

DQFHD query form dynamically and adds a secure CAPTCHA to the generated query form to make it more secure. The heart of DQFHD is to capture user interests during user interactions and to adjust the query form repeatedly. Each iteration consists of two types of user interactions: Query Form Enrichment and Query Execution. Query Form Enrichment is used to enrich the generated query form based on the user feedback.

The feedback is used to rank the generated query forms (Fig. 2). We first specify a subset of SQL to be the target language implementing the queries supported by the query forms considered in this study. In principle, many different subsets of SQL can be considered. Since the goal of this study is to explore the challenges and the promise of the approach of using keyword search to identify relevant forms for posing structured queries, the SQL we use is intended to be simple enough to allow us to explore the challenges in depth and build an end-to-end prototype solution to evaluate the approach while being expressive enough to cover a useful fraction of potential user queries that cannot be expressed with keyword search alone.

Query mapping: Each form has a brief english description based on its skeleton template. Ideally, a form description would be a user intelligible summary of which queries are supported by the form. Much as snippets of documents are used to label the documents returned by an Internet search, these descriptions would be used to label the forms returned by a keyword query. Unfortunately, generating natural-language descriptions for forms is a difficult challenge. Once we have generated a set of query templates, we can map each of them to a form. To build a form for each query template we use the following standard form components:

Label: For displaying text such as description for the form, the name of an attribute, a database constant, etc .

Drop-down list: For displaying a list of parameter values from which users can choose one.

Input box: For specifying a parameter value on the form.

Button: For functions such as submit, cancel and reset.

Unfortunately, generating natural-language descriptions for forms is a difficult challenge. Even when it is done manually, it is often unclear what a good description is especially when a form supports a wide variety of queries. Also, a manual approach quickly becomes inefficient when there are more forms, but automatically generating good form descriptions is yet another challenge.

Reconstruct: If the database schema is large and complex, user queries could be quite diverse. In that case, the end-user can regenerate the query form and can execute that as a new query. A solution is that the user inputs several keywords to find relevant query forms from a large number of pre-generated query forms. It works well in the databases which have rich textual information in data tuples and schemas. Moving beyond these approaches, we transform a user's keyword query by checking to see whether the terms from the query appear in the database and if so, modifying the query with relevant schema terms. Therefore, the system should first find out the relevant attributes for creating the selection components. We first describe how to select relevant attributes and then describe a naive method and a more efficient one query method to rank selection components as mentioned in Fig. 2.

Query construction: F-measure is a typical metric to evaluate query results. This metric is also appropriate for query forms because query forms are designed to help users query the database. The goodness of a query form is determined by the query results generated from the query form. Based on this, we rank and recommend the potential query form components so that users can refine 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 important because DQF is an online system where users often expect quick response. 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. After the expansion of both the events and the geographic terms, the query can be constructed using boolean operators. In order to achieve both high precision and recall, we setup four levels of queries, giving different weights for the retrieved documents. The higher levels of queries aim to obtain accurate results while the lower levels for the recall.

Ranking metric: Query forms are designed to return the user's desired result. There are two traditional measures to evaluate the quality of the query results: precision and recall. Query forms are able to produce different queries by different inputs and different queries can output different query results and achieve different precisions and recalls, so we use expected precision and expected recall to evaluate the expected performance of the query form. Intuitively, expected precision is the expected proportion of the query results which are interested by the current user. Expected recall is the expected proportion of user interested data instances which are returned by the current query form.

Software implementation and description: This system is implemented by PHP software and XAMP is used as a server. The data sources are MySQL and XML. The following algorithm is proposed in our system.

Algorithm:

```
Input: The link with multiple keys k
Output: Set of network elements and identifying the data.
Step 1. Split k into k1,k2,k3……kn
Step 2. t- a table where every tuple captures a join sequence of key relationships and the score of each t and the combined score of the join sequence; it is initially empty.
Step 3. On each row of t do
<ti, tj> ? t.pop();
Compute the relevant keyword r for t.
Each keyword perform the matching process to t
If matched then
Identify the field name and add it to the graph
Else
Repeat the process
Compute the score in t in the graph
Step 4. Sort the score
```

Interactions between users and DQFHD

Query form enhancement:

- DQFHD suggests a ranked list of query form components to the user
- The user selects the desired form components into the current query form

Query form implementation:

- The user fills out the current query form and submit a query
- DQFHD executes the query and shows the results
- The user provides the feedback about the query results

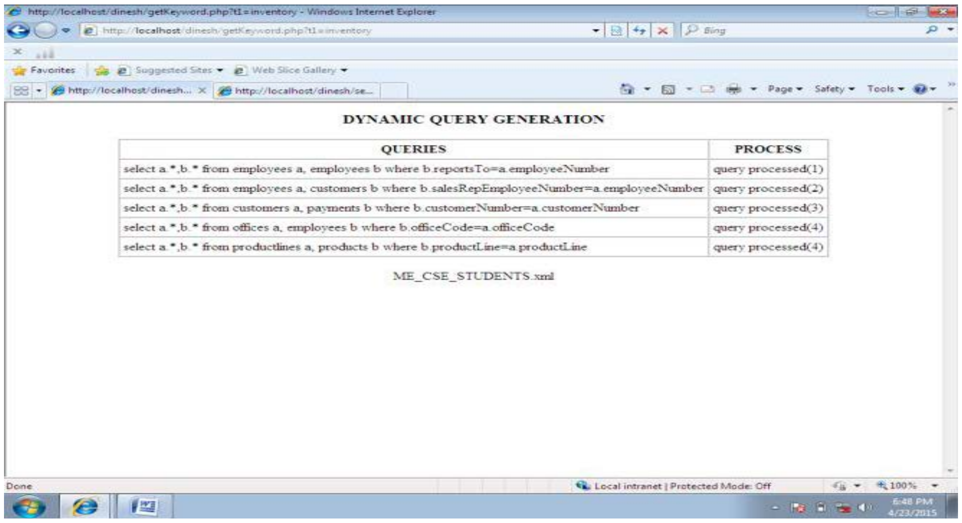
RESULTS AND DISCUSSION

The dynamic query form for heterogeneous database queries system which generates the results based on the user preferences. This system accepts various types of databases from various types of data sources as input. Then the system processed and provides the relationship among various databases which means how the attributes are interconnected with various databases. The result will be the relationship between the various attributes.

The next result will be the dynamic query generation of various types of databases which all are given as input. The result will be the basic query formation of various types of data source. Here we implemented two type of data source namely MySQL and XML databases (Fig. 3).

Here, we can also include a preliminary step as a recommended system which is contains some constraints about the databases. The constraints should be specified in the recommended form those constraints should be chosen by the user. If the generated result is satisfied by the user then it will be no problem. If the results are not fulfilled to the user then the constraints can be changed. So that new set of results will be generated by our system.

The generated results provides with better performance than the previous systems. The result fully based on various types data sources and also reveals their relationship among (Fig. 4). As a related reseaerch, we also performed the keyword search among various types of databases which are given as input. The DQFHD system provides better results than the earlier systems. The results will be the database name, the table name and the field in which the searching keyword is stored. This is a work which is related to our system which is used to show the performance of our system.

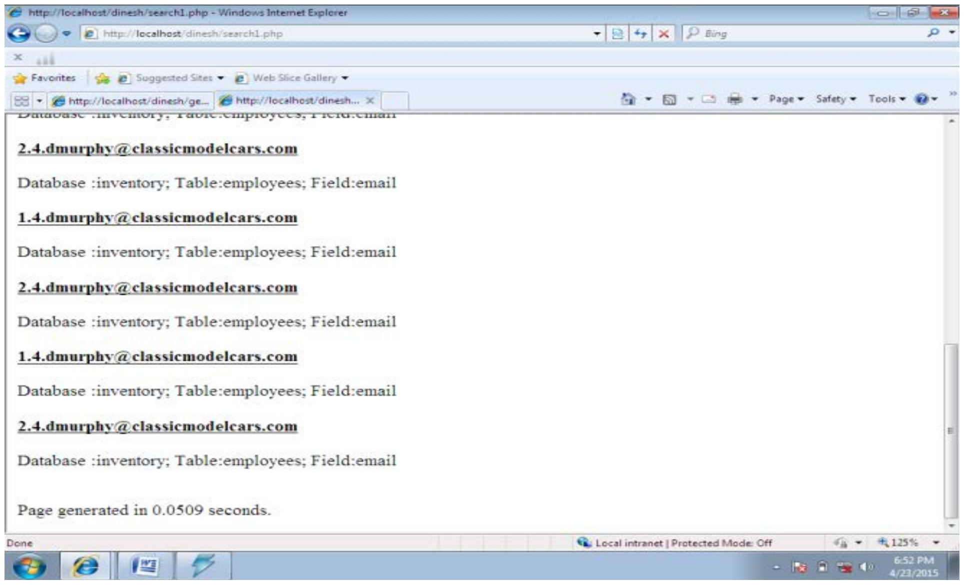


DYNAMIC QUERY GENERATION

QUERIES	PROCESS
select a.*,b.* from employees a, employees b where b.reportsTo=a.employeeNumber	query processed(1)
select a.*,b.* from employees a, customers b where b.salesRepEmployeeNumber=a.employeeNumber	query processed(2)
select a.*,b.* from customers a, payments b where b.customerNumber=a.customerNumber	query processed(3)
select a.*,b.* from offices a, employees b where b.officeCode=a.officeCode	query processed(4)
select a.*,b.* from productlines a, products b where b.productLine=a.productLine	query processed(4)

ME_CSE_STUDENTS.xml

Fig. 3: Dynamic query generation



```
2.4.dmurphy@classicmodelcars.com
Database :inventory; Table:employees; Field:email
1.4.dmurphy@classicmodelcars.com
Database :inventory; Table:employees; Field:email
2.4.dmurphy@classicmodelcars.com
Database :inventory; Table:employees; Field:email
1.4.dmurphy@classicmodelcars.com
Database :inventory; Table:employees; Field:email
2.4.dmurphy@classicmodelcars.com
Database :inventory; Table:employees; Field:email

Page generated in 0.0509 seconds.
```

Fig. 4: Keyword search

CONCLUSION

In this study, we propose a dynamic query form generation for heterogeneous database queries come within reach of which helps users animatedly engender query forms. The main suggestion is to use a dynamic model to rank form components based on user preferences. We capture user preference using both historical queries and runtime feedback such as click-through. The sample results prove that the dynamic approach will produce higher success rate and simpler query forms compared with the existing approaches. The ranking of form attributes also makes it easier for users to customize query forms. As future work, we will study how our approach can be extended to various ranking methods which are applied to various attributes of database. And also we plan to develop multiple methods to capture the user's interest for the queries besides the click feedback. For example, a text-box can be added for users to input some keywords queries. The relevance score between the keywords and the query form can be incorporated into the ranking of form components at each step.

REFERENCES

Aggarwal, C.C., J. Han, J. Wang and P.S. Yu, 2003. A framework for clustering evolving data streams. Proceedings of the 29th International Conference on Very large Data Bases, September 12-13, 2003, Berlin, pp: 81-92.

Agrawal, R., S. Gollapudi, A. Halverson and S. Ieong, 2009. Diversifying search results. Proceedings of the Second ACM International Conference on Web Search and Data Mining, February 9-11, 2009, ACM, Barcelona, Spain, ISBN:978-1-60558-390-7, pp: 5-14.

Boriah, S., V. Chandola and V. Kumar, 2008. Similarity measures for categorical data: A comparative evaluation. Red., 30: 243-254.

Chatzopoulou, G., M. Eirinaki and N. Polyzotis, 2009. Query Recommendations For Interactive Database Exploration. Scientific and Statistical Database Management. Maruanne, W. (Ed.). Springer, Berlin, Germany, ISBN: 978-3-642-02278-4, pp: 3-18.

Chaudhuri, S. and G. Das, 2003. Automated ranking of database query results. CIDR., 2003: 888-899.

Chaudhuri, S., G. Das, V. Hristidis and G. Weikum, 2006. Probabilistic information retrieval approach for ranking of database query results. ACM. Trans. Database Syst. TODS., 31: 1134-1168.

Chen, K., H. Chen, N. Conway, J.M. Hellerstein and T.S. Parikh, 2011. Usher: Improving data quality with dynamic forms. IEEE Trans. Knowl. Data Eng., 23: 1138-1138.

Chu, E., A. Baid, X. Chai, A. Doan and J. Naughton, 2009. Combining keyword search and forms for ad hoc querying of databases. Proceedings of the ACM SIGMOD International Conference on Management of Data, June 29-July 2, 2009, ACM, New York, USA., ISBN: 978-1-60558-551-2, pp: 349-360.

Cohen, B.S., O. Biton, S. Davidson and C. Froidevaux, 2007. BioGuideSRS: Querying multiple sources with a user-centric perspective. Bioinf., 23: 1301-1303.

Das, G. and H. Mannila, 2000. Context-Based Similarity Measures For Categorical Databases. In: Principles of Data Mining and Knowledge Discovery. Djamel, A.Z., J. Komoroeski and J. Zytkow (Eds.). Springer, Berlinm, Germany, ISBN:978-3-540-41066-9, pp: 201-210.

Jayapandian, M. and H.V. Jagadish, 2008a. Expressive query specification through form customization. Proceedings of the 11th International Conference on Extending Database Technology: Advances in Database Technology, March 25-30, 2008, ACM, New York, USA., ISBN: 978-1-59593-926-5, pp: 416-427.

Jayapandian, M. and H.V. Jagadish, 2008b. Automated creation of a forms-based database query interface. Proc. VLDB Endowment, 1: 695-709.

Jayapandian, M. and H.V. Jagadish, 2009. Automating the design and construction of query forms. IEEE. Trans. Knowl. Data Eng., 21: 1389-1402.

Joachims, T. and F. Radlinski, 2007. Search engines that learn from implicit feedback. IEEE. Comput., 40: 34-40.

Khoussainova, N., Y. Kwon, M. Balazinska and D. Suciu, 2010. SnipSuggest: Context-aware autocompletion for SQL. Proc. VLDB. Endowment, 4: 22-33.

Li, C., N. Yan, S.B. Roy, L. Lisham and G. Das, 2010. Facetedpedia: Dynamic generation of query-dependent faceted interfaces for wikipedia. Proceedings of the 19th International Conference on World Wide Web, April 26-30, 2010, ACM, Raleigh, North Carolina, USA., ISBN:978-1-60558-799-8, pp: 651-660.

Liu, B. and H.V. Jagadish, 2009. Using trees to depict a forest. Proc. VLDB. Endowment, 2: 133-144.

Mork, P., R. Shaker, A. Halevy and H.P. Tarczy, 2002. PQL: A declarative query language over dynamic biological schemata. Proc. AMIA. Symp., 2002: 533-537.

Nandi, A. and H.V. Jagadish, 2007. Assisted querying using instant-response interfaces. Proceedings of the International Conference on Management of Data, June 11-14, 2007, ACM, New York, USA., ISBN: 978-1-59593-686-8, pp: 1156-1158.

Rafiei, D., K. Bharat and A. Shukla, 2010. Diversifying web search results. Proceedings of the 19th International Conference on World Wide Web, April 26-30, 2010, ACM, Raleigh, North Carolina, USA., ISBN:978-1-60558-799-8, pp: 781-790.

Roy, S.B., H. Wang, U. Nambiar, G. Das and M. Mohania, 2009. Dynacet: Building dynamic faceted search systems over databases. Proceedings of the 2009 IEEE 25th International Conference on Data Engineering, March 29-April 2, 2009, IEEE, Arlington, Texas, ISBN:978-1-4244-3422-0, pp: 1463-1466.

Tang, L., T. Li, Y. Jiang and Z. Chen, 2014. Dynamic query forms for database queries. IEEE. Trans. Knowl. Ddata Eng., 26: 2166-2178.