ISSN XXXX XXXX © 2018 IJESC

Research Article Volume 8 Issue No.6

Automated Dynamic Query formation for Database Queries using Ranking

Yudhishthir D. Chavan[1], Prof. S.M. Shinde[2]
ME Student[1], Assistant Professor[2]
Department of Computer Science and Engineering
SVERI's College of Engineering, Pandharpur, India

Abstract:
A database is only as functional as query interface allows it to be. If a user is not capable to communicate to the database what user wishes from it, even the richest data store provides petite or no value. Writing well-structured queries, in languages such as SQL and XQuery, can be challenging due to a number of reasons, including the user's lack of familiarity with the query language and the user's ignorance of the underlying schema. A form based query interface, which only requires filling blanks to identify query parameters, is precious since it helps make data users with no knowledge of official query languages or the database schema. In practice, form-based interfaces are used frequently, but usually each form is designed in an adhoc way and its applicability is restricted to a small set of fixed queries. Query form is one of the majority 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. Dynamic question type system: DQF, a question interface that is capable of dynamically generating question forms for users. Different from ancient document retrieval, users in information retrieval area unit usually willing to perform several rounds of actions (i.e., refinement question conditions) before distinctive the final candidates. The essence of DQF is to capture user interests throughout user interactions and to adapt the question type iteratively. Every iteration consists of two sorts of user interactions: it contains only a few primary attributes of the information. The essential question type is then enriched iteratively via the interactions between the user and our system till the user is satisfying with the question results. Goal of this Project is to show that the advantages of using dynamic query forms for database over the existing static forms for database.

Keywords: DQF, Query Forms, Query Generation, Query Execution , Ranking.

I. INTRODUCTION

A database is only as functional as query interface allows it to be. If a user is not capable to communicate to the database what he or she wishes from it, even the richest data store provides petite or no value. Writing well-structured queries, in languages such as SQL and XQuery, can be challenging due to a number of reasons, including the user's lack of familiarity with the query language and the user's ignorance of the underlying schema. A form based query interface, which only requires filling blanks to identify query parameters, is precious since it helps make data users with no knowledge of official query languages or the database schema. In practice, form-based interfaces are used frequently, but usually each form is designed in an adhoc way and its applicability is restricted to a small set of fixed queries. Query form is one of the majority 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.

The main purpose of DQF is to understand the user interests about query form and to adapt the question type repeatedly. The question type iteration consists of two sorts of user interactions: it contains attributes of the information. The interaction continues between the user and the system till the user satisfies with query results. Goal of this Project is to show that the advantages of using dynamic query forms for database over the existing static forms for database. We reach to user satisfaction using ranking attribute. Ranking attribute stores users query results. Whenever user revisits the site it will display all related query results in ranking list. If user is satisfied with these results then he will use the results otherwise he will go with new form. menu to the left of the text, or on your toolbar or formatting palette. Click on the down arrow to access the various styles (for example, the style at this point in the document is "First Paragraph"). Scroll through the style list and you will find "First Paragraph" highlighted. To use these built-in style guides, highlight a section that you want to designate with a certain style, and then select the appropriate name on the style pull-down menu.

II. PROPOSED SYSTEM

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, thesystem 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.

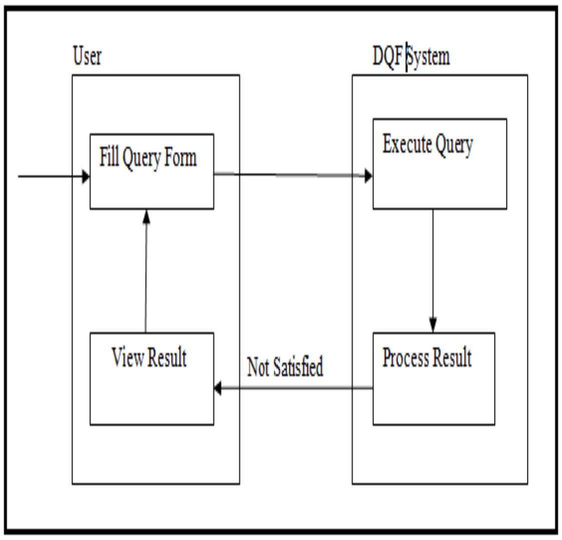


Figure.1. Proposed System

The above proposed system has following advantages:

- The proposed a dynamic query form generation approach which helps users dynamically generate query forms
- The dynamic approach often leads to higher success rate and simpler query forms compared with a static approach.
- The ranking of form components also makes it easier for users to customize query forms.

DQF will actually work in two phases one is Query Execution and other is Query Enhancement.

A. Query Execution :

In Query execution user will fill the query form with desired form components, query form will be executed by the system and result will be displayed. If user gets desired output of that query form then user will rank the query form for future use. Depending on the ranking, query form will be restored for further use. And user can exit from the systems else user can get another query form by selecting other form components of his desire to get expected query result. After that user will execute the query form and the process will repeat until user will get desired output.

B. Query Form Enhancement :

As said in above query execution method, if user is not satisfied with query results then again he will select the query component, fill the form and he will get the results. If user is satisfied then use the query from ranking list, otherwise he will select another form component. By this query component enhanced based on user choice and query form improvement and efficiency is increased.

C. Ranking Attribute:

Ranking attribute stores query component of user and restore it whenever he revisits the system. If his query is related to ranking list then he will select and use the rank or he go with new query form.

D. Ranking Matrices :

There are two measures for evaluating the quality of the query results: precision and recall. Expected precision and expected recall to evaluate the expected performance of the query form. Expected precision is the expected proportion of the query results which are interested by the current user.

$$\text{Precision}\, E(F) = \frac{\sum_{d \in XAf} P_C(dAf)\, P(dAf)\, P(\alpha f \mid d) N}{\sum_{d \in XAf} P(dAf)\, P(\alpha f \mid d) N} \qquad (1)$$

Expected recall is the expected proportion of user interested data instances which are returned by the current query form.

$$\text{Recall}\, E(F) = \frac{\sum_{d \in XAF} P_C(dAf)\, P(dAf)\, P(\alpha f \mid d) N}{\alpha N} \qquad (2)$$

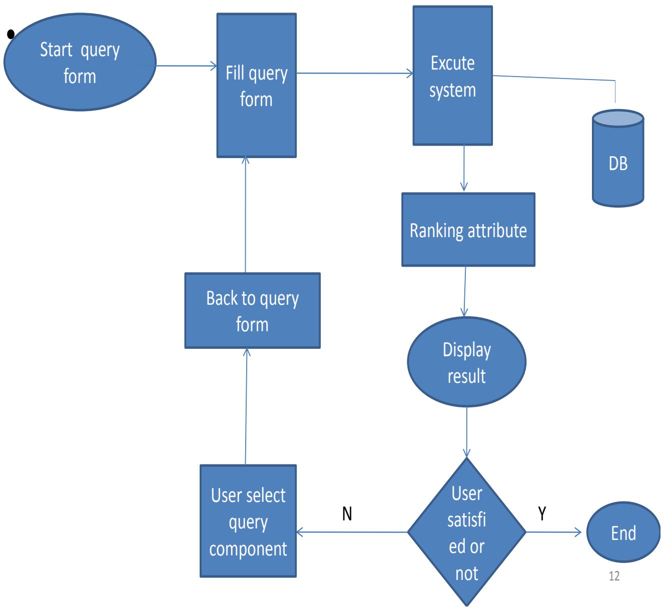


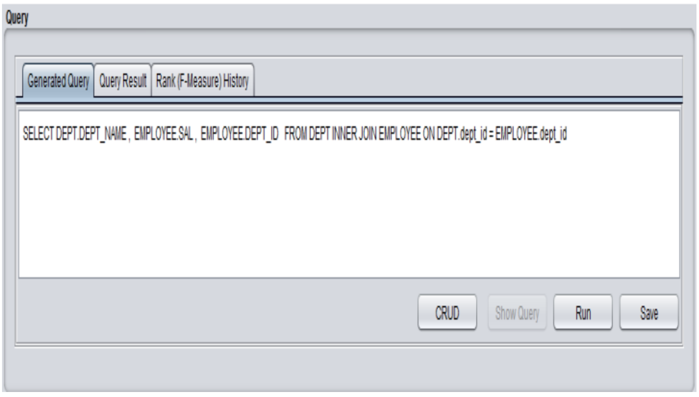
Figure.2. System Architecture

The calculation of ranking score is an entity is by the averaged F-Score. Rank score is used to calculate the accuracy. Initially the user select the components from more no of components and the ranking are calculated for selected components has the highest rank.

III. EXPERIMENT AND RESULT

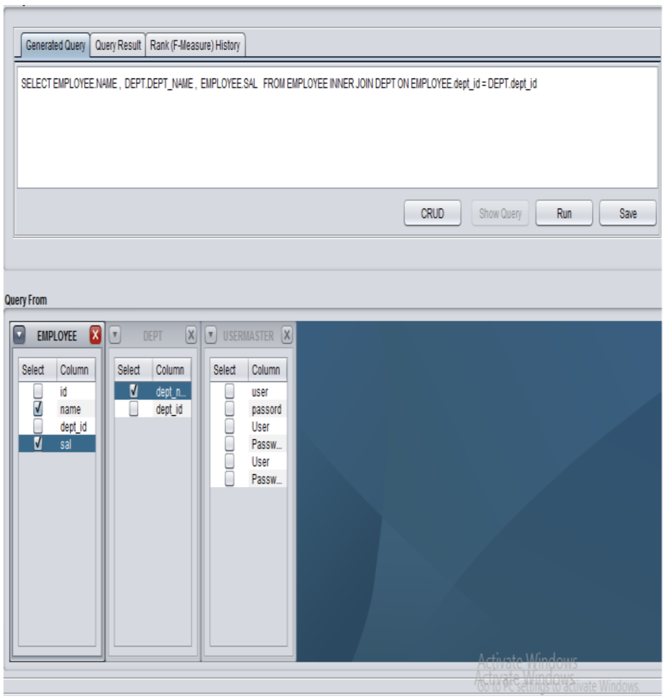
Step 1 Query Generation :

The system provides a form window. User needs to select the generate query tab and he need to select the tables, attributes of the table and operation to be performed (i.e. Insert, delete and update method).



```
SELECT DEPT.DEPT_NAME , EMPLOYEE.SAL , EMPLOYEE.DEPT_ID  FROM DEPT INNER JOIN EMPLOYEE ON DEPT.dept_id = EMPLOYEE.dept_id
```

Figure.3. Query Generation



```
SELECT EMPLOYEE.NAME , DEPT.DEPT_NAME , EMPLOYEE.SAL  FROM EMPLOYEE INNER JOIN DEPT ON EMPLOYEE.dept_id = DEPT.dept_id
```

EMPLOYEE

Select	Column
☐	id
✓	name
☐	dept_id
✓	sal

DEPT

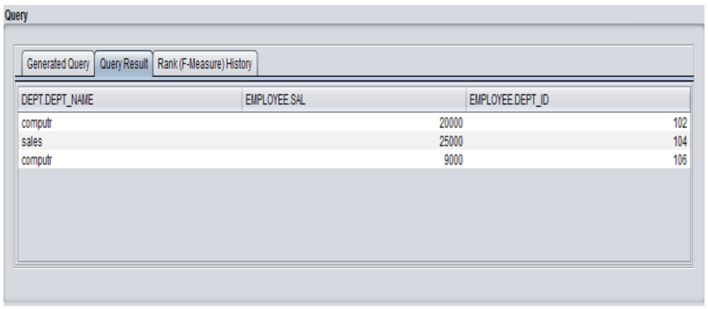
Select	Column
✓	dept_n...
☐	dept_id

USERMASTER

Select	Column
☐	user
☐	passord
☐	User
☐	Passw...
☐	User
☐	Passw...

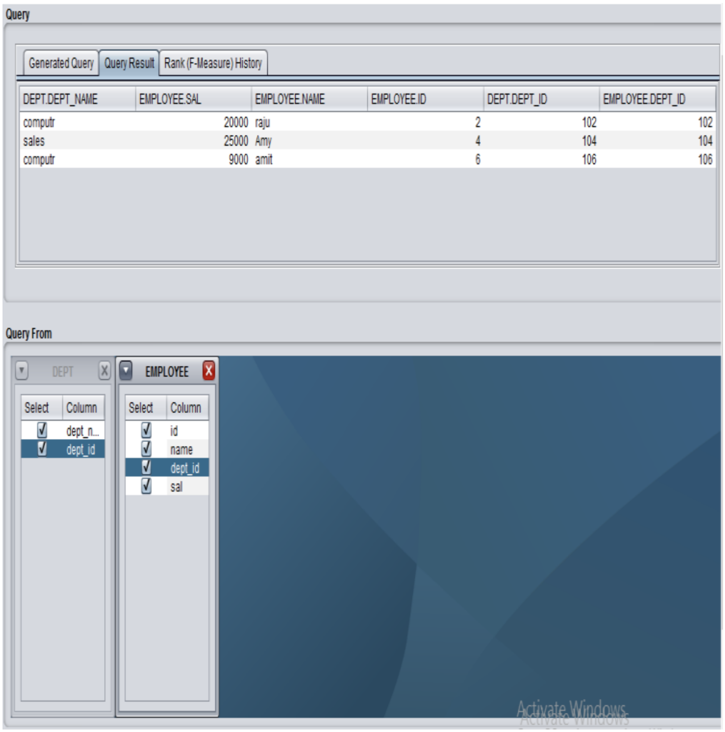
Figure.4. Query Generation

Step 2 Query Result:
Query results Once user selects all the details in generate query tab, he will click on show query. Based on the given values the system will display query results.



DEPT.DEPT_NAME	EMPLOYEE.SAL	EMPLOYEE.DEPT_ID
computr	20000	102
sales	25000	104
computr	9000	106

Figure.5. Query Result



DEPT.DEPT_NAME	EMPLOYEE.SAL	EMPLOYEE.NAME	EMPLOYEE.ID	DEPT.DEPT_ID	EMPLOYEE.DEPT_ID
computr	20000	raju	2	102	102
sales	25000	Amy	4	104	104
computr	9000	amit	6	106	106

DEPT

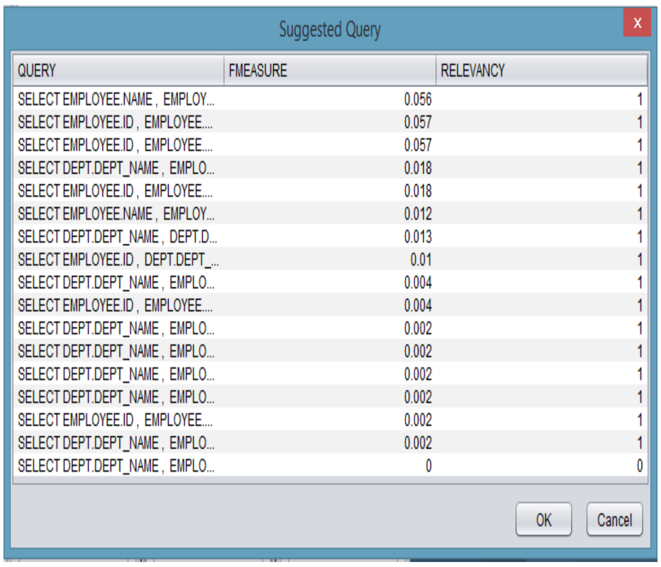
Select	Column
✓	dept_n...
✓	dept_id

EMPLOYEE

Select	Column
✓	id
✓	name
✓	dept_id
✓	sal

Figure.6. Query Result

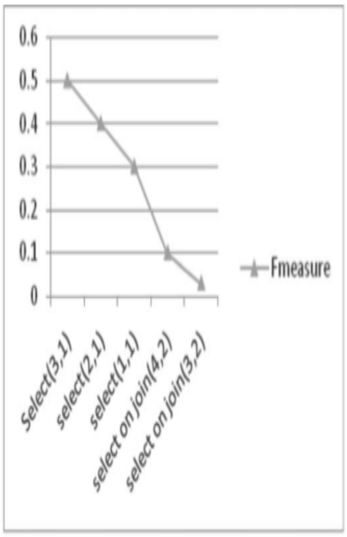
Step 3 Ranking Attribute:
Rank Attribute Rank stores the query results for future use and restores when user executes another query in future. Using rank it is possible to know the user interest and we make query enhancement easily.



QUERY	FMEASURE	RELEVANCY
SELECT EMPLOYEE.NAME , EMPLOY...	0.056	1
SELECT EMPLOYEE.ID , EMPLOYEE....	0.057	1
SELECT EMPLOYEE.ID , EMPLOYEE....	0.057	1
SELECT DEPT.DEPT_NAME , EMPLO...	0.018	1
SELECT EMPLOYEE.ID , EMPLOYEE....	0.018	1
SELECT EMPLOYEE.NAME , EMPLOY...	0.012	1
SELECT DEPT.DEPT_NAME , DEPT.D...	0.013	1
SELECT EMPLOYEE.ID , DEPT.DEPT_...	0.01	1
SELECT DEPT.DEPT_NAME , EMPLO...	0.004	1
SELECT EMPLOYEE.ID , EMPLOYEE....	0.004	1
SELECT DEPT.DEPT_NAME , EMPLO...	0.002	1
SELECT DEPT.DEPT_NAME , EMPLO...	0.002	1
SELECT DEPT.DEPT_NAME , EMPLO...	0.002	1
SELECT DEPT.DEPT_NAME , EMPLO...	0.002	1
SELECT EMPLOYEE.ID , EMPLOYEE....	0.002	1
SELECT DEPT.DEPT_NAME , EMPLO...	0.002	1
SELECT DEPT.DEPT_NAME , EMPLO...	0	0

Figure.7. Ranking

The Concept of this paper has implemented and results are shows as follows.



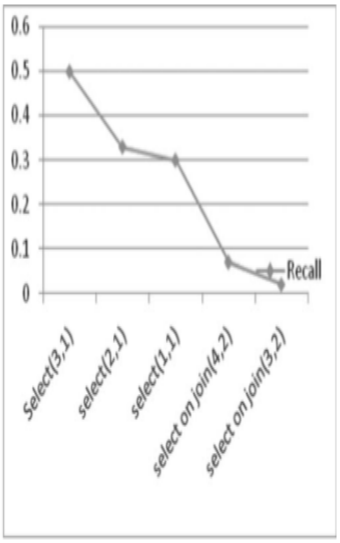
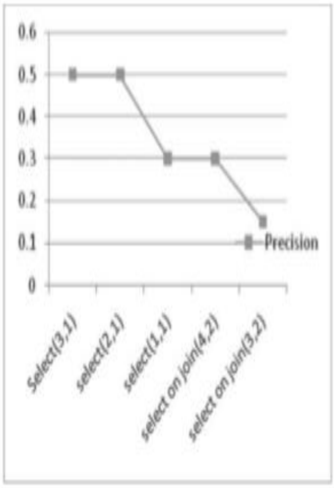


Figure.8. FMeasure Figure.9. Recall



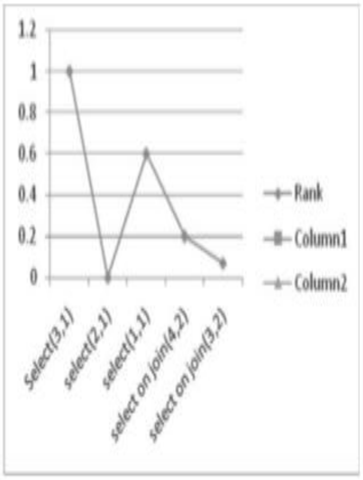


Figure.10. Precision Figure.11. Rank

IV. CONCLUSION

Query interfaces play a vital role in determining the usefulness of a database. A form-based interface is widely regarded as the most user-friendly querying method. In this paper, we have developed mechanisms to overcome the challenges that limit the usefulness of forms, namely their restrictive nature. In this paper we propose an interactive query form generation approach which helps users to dynamically generate query forms. As future work we will study how our approach can be extended to non relational data. As for the future work, we plan to develop multiple methods to capture the user's interest for the queries besides the click feedback. For instance, we can add a text-box for users to input some keywords queries.

V. FUTURE WORK

We extend our method to non relational data. And try to add keyword matching concept, so that user can just search any previous query results in search window. And the system matches the keywords with existing data and displays the results.

VI. REFERANCES

[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 2013

[2] 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.

[3] M. Jayapandian and H. V. Jagadish. Automating the design and construction of query forms. IEEE TKDE, 21(10): 13891402, 2009.

[4] 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.

[5] W. B. Frakes and R. A. Baeza-Yates. "Information Retrieval: Data Structures and Algorithms". Prentice-Hall, 1992. [15] T. Joachims and F. Radlinski. "Search engines that learn from implicit feedback". IEEE Computer (COMPUTER), 40(8):34–40, 2007.

[6] N. Khoussainova, M. Balazinska, W. Gatterbauer, Y. Kwon, and D. Suciu." A case for a collaborative query management system". In Proceedings of CIDR, Asilomar, CA, USA, January 2009.

[7] Q. T. Tran, C.-Y. Chan, and S. Parthasarathy. "Query by output". In Proceedings of SIGMOD, pages 535–548, Providence, Rhode Island, USA, September 2009.

[8] Jayashri M. Jambukar." Survey Paper on Creation Of dynamic query Form for mining highly Optimized transactional databases". International Journal of Computational Engineering Research||Vol, 04||Issue, 3||, March 2014.

[9] Random Query Formulation for Database Queries, Gopi Krishna Lakksani, BalakrishnaNayudari, Department of Computer Science, Mandapalle University, AP. 8 August 2014.