ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Forms of Dynamic Query for Evaluation of Database Queries Using Interfaces

Mrs. Jayashri M. Jambukar
M.E Computer Student,
AVCOE Sangamner, University Pune.

Mr.Vaidya Milindkumar B
Co-ordinator,
AVCOE Sangamner, University Pune.

ABSTRACT:

Recent available databases and web databases maintain big and variable data in the database. The original databases contain over 100 or 1000 of relations and attributes which is. Traditional predefined query forms are not able to comfortable various ad-hoc queries from users on those databases. In this paper DQF, the acual database query form interface, this is able to dynamically create the query forms for databases. Here, the need of DQF is to gain a user's preference and proper rank query form components for query formation. Here, the creation of a query form is repeatative process and is suggested by the user in the implementation.

Among this, at each and every iterations, one of system automatically creates ranking lists of form components and the user then includes the form components into the query form in the database. Mainly The ranking of form components is based on the captured user preference data. In this, a user can also fill and complete the query form and submit queries to view the query result for every iteration of the query formation. So, in this way, a query form could be dynamically cleared till the user satisfies with the query results in to the database.

INDEX TERMS:

Query Formation, User Interaction in the system, Query Form Generation in the database.

I.INTRODUCTION:

Generally, Query form is most rarely used user interfaces for querying databases for query formation. Here, some Traditional query forms are designed and predefined by developers or DBA in various information management systems in query formation for the database implementation.

The development of web information and recent databases, modern databases become very big and complicated. The databases have over 100s of new entities for different types of data resources [6] [5] [7] in the databases for query formation. In this paper, various web databases, such as Freebase and DBPedia, like typically have thousands of structured web entities [4] [2]. So, it is hard to set of static query forms to verify various ad-hoc database queries on those complex databases in DQF. Many existing database management and development tools provide various mechanisms to let users create customized and unique queries on databases for DQF.

III. HARDWARE DESIGN OF PROPOSED SYS¬TEM:

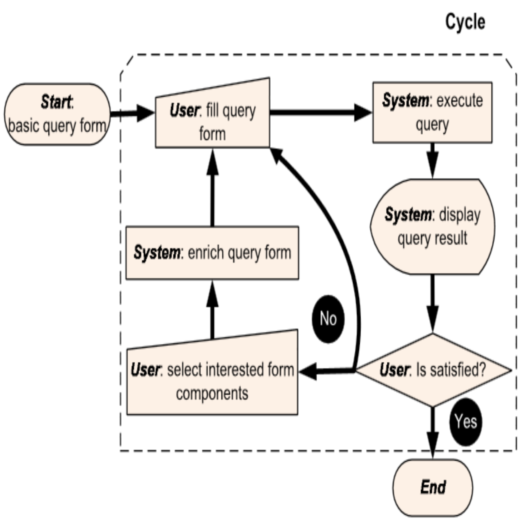


Fig.1 System architecture

A.EXISTING SYSTEM:

Recently proposed inbuilt approaches to create the database query forms without any user participation available a data-driven method in the database for DQF.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

So, initially it finds a set of data attributes, in which most likely queried based on the database schema and data instances. When we create a big number of query forms, then how to let users find an accurate and convenient query form would be challenging for nation. That concatenate the searching of keyword with query form creation is used. Here, It directly generates a full of query forms in advance. Here, the user inputs various keywords to search the related query forms from a vast number of old query forms in the database.

B.ALGORITHM:

B1.PAGE RANK ALGORITHM:

We proposed a Page Rank-like algorithm to compute the importance of an attribute in the schema for query formation. Here, we use the schema graph to compute the relevance of two attributes of database. So, In this query formation A database schema graph is denoted by G= (R1, Fkey, ξ, A), where R1 is the set of nodes representing the relations, A is the set of attributes, then Fkey is the set of edges representing the foreign keys in the database, and ξ: A-R is an attribute labeling function to indicate which relation contains the attribute in the database.

```
Algorithm 1:  Query Construction
Data: q={q1,q2,…..,} is a set of queries which is executed in previous on Si.
Output:  qone is query of one-query form.
 Begin
б one ‹— 0
for q  € Ø do
б one ‹— б one  v   бq
Aone ‹— ASi U Ar(Si)
Qone ‹— Genquery(Aone, бone)
```

C.PROPOSED SYSTEM:

We propose a Dynamic Query Formation system: DQF, in this a query interface which is capable of dynamically generating query forms for users. Here, it is Different from traditional document store and retrieval; users in database retrieval are generally willing to perform many rounds of actions before identifying the last candidates in the implementation.

The requirement of DQF is to obtain the user interests at the time of user interactions and to adapt the query form iteratively one by one. In this, each iterations includes two types of user interactions as below:

1. Query Form Enrichment

2. Query Execution.

D. QUERY FORM INTERFACE:

D1. QUERY FORM:

In this, we will define the query form for generation of query. For this, each query form corresponds to an SQL query template. Definition 1: A query form F1 is describe as a tuple (AF1 , RF1 , σF1 , (RF1)), which represents a database query template as follows:
F1 = (SELECT A1,A2, ...,Ak
FROM (RF) WHERE σF1),
where AF1 = fA1,A2, ...,Akg are k attributes for the projection, k > 0. RF1 = fR1,R2, ...,Rn is the set ofn relations (or entities) involved in this query, n > 0.Each attribute in AF1 belongs to one relation in RF1 . σF1 is a conjunction of expressions for selections (or conditions) on relations in RF1 . (RF1) is a join function to generate a conjunction of expressions for joining relations of RF1 .In the user interface of a query form F1, AF1 is the set of columns of the result table. σF1 is the set of input components for users to fill.

Query forms allow users to fill parameters to generate different queries. RF1 and (RF1) are not visible in the user interface at the time of query formation, which are usually generated by the system according to the database schema in the databases.

E.QUERY RESULTS:

As per the query result, To decide whether a query form is usable or not, in this a user does not have so much time to go with each data instance in the query results in the database. Actually, many database queries give a vast amount of data instances in the database query formation. In order to remove this many types of output problem, here we got the result in a compressed result table to show a high level view of the query results initially. Figure 2 shows the main flow of user actions diagram.

There are many clustering algorithms for creating the compressed view mainly. So, In our implementation, we choose the incremental data clustering frame work because of the efficiency issue. Certainly, different data clustering methods would have different compressed views for the users. There are different types of clustering therapies that are preferable to different data types. In this paper the system developers can select a different clustering algorithm if needed in the query formation.

Algorithm 2: FindBestLessEqCondition

```
Data: $\alpha$ is the fraction of instances desired by user,
      $D_{Q_{one}}$ is the query result of $Q_{one}$, $A_s$ is the
      selection attribute.
Result: $s^*$ is the best query condition of $A_s$.
begin
    // sort by $A_s$ into an ordered set $D_{sorted}$
    $D_{sorted} \leftarrow \mathtt{Sort}(D_{Q_{one}}, A_s)$
    $s^* \leftarrow \emptyset$, $fscore^* \leftarrow 0$
    $n \leftarrow 0$, $d \leftarrow \alpha\beta^2$
    for $i \leftarrow 1$ to $|D_{sorted}|$ do
        $d \leftarrow D_{sorted}[i]$
        $s \leftarrow$ "$A_s \leq d_{A_s}$"
        // compute fscore of "$A_s \leq d_{A_s}$"
        $n \leftarrow n + P_u(d_{A_{F_i}})P(d_{A_{F_i}})P(\sigma_{F_i}|d)P(s|d)$
        $d \leftarrow d + P(d_{A_{F_i}})P(\sigma_{F_i}|d)P(s|d)$
        $fscore \leftarrow (1+\beta^2) \cdot n/d$
        if $fscore \geq fscore^*$ then
            $s^* \leftarrow s$
            $fscore^* \leftarrow fscore$
```

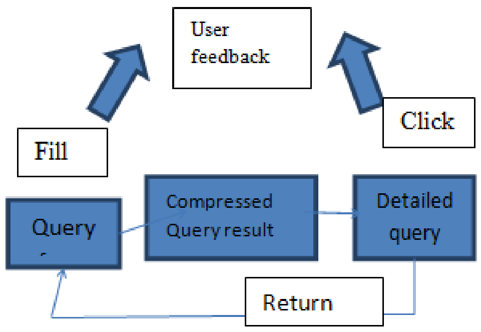


Fig.2 User Action

F. EVALUATION:

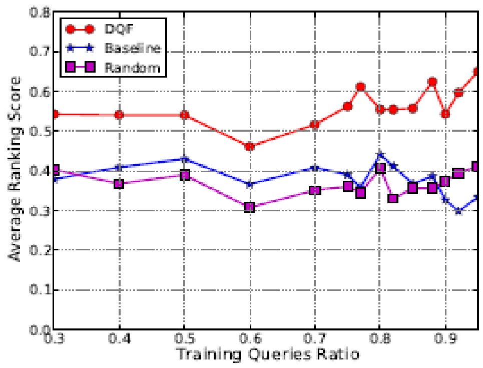
The aim of our evaluation is to confirm the following hypotheses as given :

H1: Is DQF which is used in the database more important than existing approaches like static query form and customized query form? Then H2: Is DQF in the database more effective to rank projection and selection components than the baseline and the random method? And H3: Is DQF needs to rank the required query form components in an online user interface?

G.RESULT ANALYSYS:

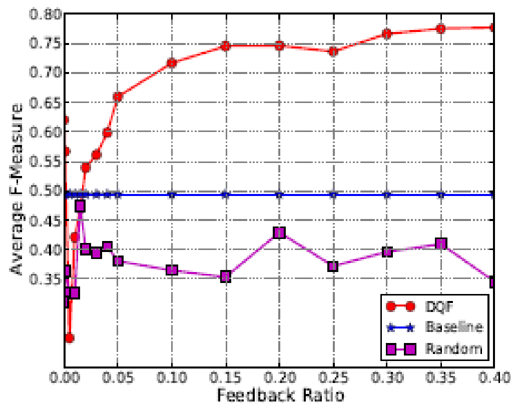
A1.RANKING SELECTION COMPONENTS:

Generally, F-Measure is use to calculate and measure ranking of user preference using historical queries and run-time feedback such as click-through. Experimental results show that selection components. In ranking selection if the output of query is obtained by suggested selection component is closer to the original query result in selection component then the F-Measure should be greater. In selection component and a test query P1 we define the set of collected data instances returned by the query P1. In the ranking process we also generated a query ^P1, where ^P1 is similar to P1 except selection component in this process. At the end selection component of ^P1 is generated by the highest ranked component returned by following ranking methods.



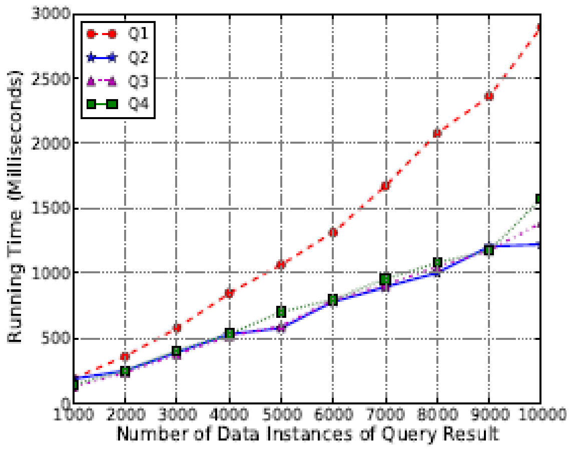
(a1) Ranking Scores in patient Data Increase Average ranking score value of DQF

A2. AVERAGE F-MEASURE OF SUGGESTED SELECTION COMPONENTS:



(a2) Average F-Measure for patient Data (Top 5 Ranked Components) Increase Average F-Measure value of DQF

A3 SCALABILITY OF RANKING SELECTION COMPONENTS:



(a3) Running time on patient Data Reduce the running time (milliseconds)

IV CONCLUSION:

In this DQF paper, we propose a very essential dynamic query form creation approach which helpful for query formation. In this paper, the key concept is to use a probabilistic model to rank form components according to the user preferences. Here, We absorbs 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. As future work, we will study how our approach can be extended to relational data.

ACKNOWLEDGEMENT:

We dedicate a special thanks to our department and our institute for the great support regarding paper and their all views.

REFERENCES:

[1]ColdFusion.http://www.adobe.com/products/coldfusion/.

[2] DBPedia. http://DBPedia.org.

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

[4] Freebase. http://www.freebase.com.

[5] S. Cohen-Boulakia, O. Biton, S. Davidson, and C. Froidevaux.Bioguidesrs: querying multiple sources with a user-centric perspective. Bioinformatics, 23(10):1301–1303, 2007.

[6] J. C. Kissinger, B. P. Brunk, J. Crabtree, M. J. Fraunholz,B. Gajria, A. J. Milgram, D. S. Pearson, J. Schug, A. Bahl, S. J.iskin, H. Ginsburg, G. R. Grant, D. Gupta, P. Labo, L. Li,M. D. Mailman, S. K. McWeeney, P. Whetzel, C. J. Stoeckert,and J. . D. S. Roos. The plasmodium genome database:Designing and mining a eukaryotic genomics resource. Nature,419: 490–492, 2002.

[7] P. Mork, R. Shaker,A. Halevy, and P. Tarczy-Hornoch. Pql: a declarative query language over dynamic biological schemata.In In Proceedings of American Medical Informatics Association Fall Symposium, pages 533–537, San Antonio, Texas, 2007.