IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 16, Issue 5, Ver. IV (Sep – Oct. 2014), PP 154-159
www.iosrjournals.org

Dynamic Query Form with query Refinement and Database encryption

Meenu Joy   Bhruguram T M
Adi Shankara Institute of Engineering and Technology, Kalady

Abstract: *In the present circumstances real world databases contain thousands of attributes and relations. Access the information from this corpulent database is a nontrivial task and is an exploring area. Queries are using for database access, but it is a thought-provoking task to the end user because, dearth of familiarity with query language and illiteracy of underlying schema. The new system uses the allowance of query by form methodology. In which the user is granted with a query form that will help the user to iteratively search the database and query enhancement is provided to the user by means of feedback or response. Ad hoc queries can also gratified by using this proposed system. Query enhancement by response is provided by ranking the attribute used in the form. Ranking is mean by precision and recall. To provide the security to the system and make it efficient for private database an encryption is applied.*

Keywords: *Query Interface, Precision, Recall.*

I. Introduction

Query by form is a simple and intuitive methodology that is frequently used as an entry to database. Google and yahoo using this query by form approach that requires fill in the blanks to specify the parameters for information retrieval. Query by form have an advantage that user does not need to worry about how data is organized in storage and no expertise in query language like SQL, so query by form is an most frequently used mechanism to access the database. The existing systems like SQF (static Query Form), CQF (Customized query form) have lot of drawbacks that means these have no query refinement and there was no dynamic nature to these methods to satisfy ad-hoc queries.

The problem identified here is how an efficient query form can be designed to boost the user satisfaction in information retrieval. The problem can be solved by designing a good query form with dynamic nature it means the user can iteratively search to the database until he or she get satisfied. Also the query form is provided with a query refinement by means of ranking the attributes of encrypted database. Ranking is done by precision and recall which are measures used for performance evaluation of information retrieval.

1.1 Proposed Approach

Proposed approach is a dynamic query form system with query refinement and database encryption. This system provides a query by form interface to the user. The query form contains many fields to the user for filling the attributes those he or she want to view. In addition with the attribute the user is provided a constraint specification field by using that capability the user can easily make conditions. Every entry to the form is taken as input to the database.

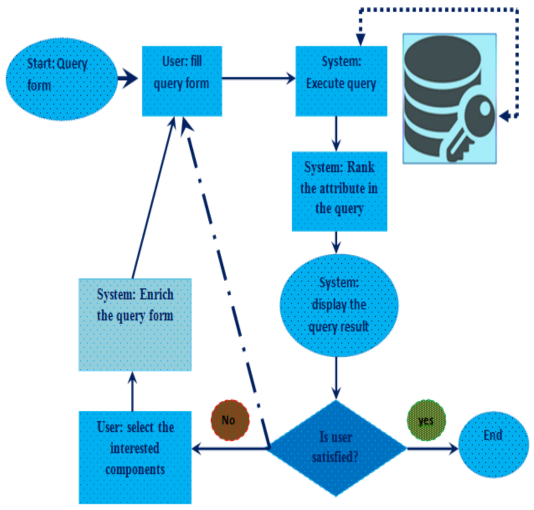
Start: Query form
User: fill query form
System: Execute query
System: Rank the attribute in the query
System: display the query result
Is user satisfied?
yes
End
No
User: select the interested components
System: Enrich the query form

Fig.1. Dynamic Query Form System With Query Refinement and Database Encryption.

All attributes from the form along with condition is taken and query is generated, the system will execute the generated query by accessing the encrypted database. The database encrypted by means of symmetric cipher algorithm Advanced Encryption Standard.AES is old encryption standard even it is an old standard it is more secure and the system will become more secure. By providing the security the usage of the system can be extended to secure database applications.

The system will execute the query by means of structured query language. Along with execution the ranking of attribute is also done. The attribute that is specified in the form is taken and a probability measure is calculated by precision and recall. Precision means the ability to retrieve the top ranked documents that are mostly relevant. Recall means the ability of the search to find all relevant items in the corpus.

For a given query, produce the ranked list of retrievals. Adjusting a threshold on this ranked list produces different sets of retrieved documents, and therefore different recall/precision measures. Mark each document in the ranked list that is relevant according to the standard. Compute a recall/precision pair for each position in the ranked list that contains a relevant document. Using this recall and precision F-Measure is calculated, one measure of performance that takes into accounts both recall and precision. Harmonic mean of recall and precision. That F-Measure is sorted and highest score attribute is listed to the user as query refinement and feedback. The system will display the result to the user. If the user is satisfied with the result user can stop the search else the system will display the ranked attribute and user can add these ranked attribute to the form that is query enrichment. These query enrichment can help the user for boosting the user satisfaction. This search can iteratively progress until user gets satisfied. Keyword search is also given to the user for accessing the database. In this case user has to know about the keywords.

1.2 Own Contribution

- In order to extent the system performance , security and make the system suitable for private database applications an encryption is provided to the database.AES advanced encryption Standard algorithm is using here for encryption.
- The user is provided a keyword search option as an entry to database access.

II. Related Work

How to let non-proficient 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) [36] 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. [1]

2.1 Customized Query Form: Existing database clients and tools make great efforts to help developers design and generate the query forms, such as Easy Query [3], Cold Fusion [1], SAP, Microsoft Access and so on. They 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 [16]. [17] Proposed a system which allows end-users to customize the existing query form at run time. However, an 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.

2.2 Autocompletion for Database Queries: In [26], [21], novel user interfaces have been developed to assist the user to type the database queries based on the query workload, the data distribution and the database schema. Different from our work which focuses on query forms, the queries in their work are in the forms of SQL and keywords.

2.3 Query Refinement: Query refinement is a common practical technique used by most information retrieval systems [15]. It recommends new terms related to the query or modifies the terms according to the navigation path of the user in the search engine. But for the database query form, a database query is a structured relational query, not just a set of terms.

III. Query Form

3.1 Query by Form Interface

In this section we formally define the query form. Each query form corresponds to an SQL query template. A query form QF is defined as a tuple (A_F, R_F, σ_F, $\bowtie$, (R_F)), which represents a database query template as follows:

Where $A_F = \{A_1, A_2... An\}$ are n attributes for projection, $n > 0$. $R_F = \{R_1, R_2... R_m\}$ is the set of m relations (or entities) involved in this query, $m > 0$. Each attribute in A_F belongs to one relation in R_F. σ_F is a conjunction of expressions for selections (or conditions) on relations in R_F. ⋈ (R_F) is a join function to generate a conjunction of expressions for joining relations of R_F .

In the user interface of a query form F, A_F is the set of columns of the result table; σ_F is the set of input components for users to fill. Query forms allow users to fill parameters to generate different queries. R_F and ⋈ (R_F) are not visible in the user. Interface, which are usually generated by the system according to the database schema. For a query form F, (R_F) is automatically constructed according to the foreign keys among relations in R_F. Meanwhile, R_F is determined by A_F and σ_F. R_F is the union set of relations which contains at least one attribute of A_F or σ_F. Hence, the components of query form F are actually determined by A_F and σ_F. As we mentioned, only A_F and σ_F are visible to the user in the user interface. In this paper, we focus on the projection and selection components of a query form. Ad-hoc join is not handled by our dynamic query form because join is not a part of the query form and is invisible for users.

3.2 Query Result

To check the query form is useful or not, user does not have time to go through every result .To avoid the many answer problem [10] in database here uses a clustering technique. Then, the user can click through interested clusters to view the detailed data instances. There are many one-pass clustering algorithms for generating the compressed view efficiently [34], [5]. In our implementation, we choose the incremental data clustering framework [5] because of the efficiency issue Certainly, different data clustering methods [1].

Another important usage of the compressed view is to collect the user feedback. Using the collected feed-back, we can estimate the goodness of a query form so that we could recommend appropriate query form components. In real world, end-users are reluctant to provide explicit feedback [19]. The click-through on the compressed view table is an implicit feedback to tell our system which cluster (or subset) of data instances is desired by the user. The clicked subset is denoted by S_{ud}. Note that S_{ud} is only a subset of all user desired data instances in the database. But it can help our system generate recommended form components that help users discover more desired data instances. In some recommendation systems and search engines, the end-users are also allowed to provide the negative feedback. The negative feedback is a collection of the data instances that are not desired by the users. In the query form results, we assume most of the queried data instances are not desired by the users because if they are already desired, then the query form generation is almost done. Therefore, the positive feedback is more informative than the negative feedback in the query form generation. Our proposed model can be easily extended for incorporating the negative feedback.

IV. Ranking of Attributes

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 [30]. 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. The user interest is estimated based on the user's click through on query results displayed by the query form. For example, if some data instances are clicked by the user, these data instances must have high user interests. Then, the query form components which can capture these data instances should be ranked higher than other components. Next we introduce some notations and then define expected precision and recall.

Let F be a query form with selection condition σ_F and projection attribute set A_F. Let D be the collection of instances in ⋈ (R_F). N is the number of data instances in D. Let d be an instance in D with a set of attributes A = {A_1, A_2, ...,A_n}, where n = |A|. We use d_{AF} to denote the projection of instance d on attribute set A_F and we call it a projected instance, P(d) is the occurrence probability of d in D. P(σ_F |d) is the probability of d satisfies σ_F. P(σ_F |d) ∈ {0, 1}.P(σ_F |d) = 1 if d is returned by F and P(σ_F |d) = 0 otherwise.

Metrics: We now describe the two measures expected Precision and expected recall for query forms [1].

Definition 2: Given a set of projection attributes A and a universe of selection expressions σ, the expected precision and expected recall of a query form F=(AF ,
R_F, σF, ⋈ (R_F)) are Precision_E (F) and Recall_E (F) respectively.

$$Precision_{E^F} \quad \frac{\Sigma d\epsilon DA_F P_{u(d_{AF})P(d_{AF})P\left(\frac{\sigma_F}{d}\right)N}}{\Sigma_{d\epsilon DA_F} P_{(d_{AF})}\text{P}\left(\frac{\sigma_F}{d}\right)\text{N}} \quad (1)$$

$$Recall_{E^F} \quad \frac{\Sigma d\epsilon DA_F P_{u(d_{AF})P(d_{AF})P\left(\frac{\sigma_F}{d}\right)N}}{\alpha N} \quad (2)$$

α is the fraction of instances desired by the user, i.e.,

$$\alpha = \Sigma_{d \epsilon D_{AF}} P_{(d)} P_{u(d)} \quad (3)$$

The numerators of both equations represent the expected number of data instances in the query result that are desired by the user. In the query result, each data instance is projected to attributes in A_F. So P_u (d_{AF}) represents the user interest on instance d in the query result. P (d_{AF})N is the expected number of rows in D that the projected instance d_{AF} represents. Considering both expected precision and expected recall, we derive the overall performance measure, expected F-Measure as shown in Equation 4. Note that β is a constant parameter to control the preference on expected precision or expected recall.

$$FScore_E{}^F = \left(1+\beta^2\right) * Precision_{E^F} * Recall_{E^F} \Big/ \beta^{2 * Precision_{E^F} + Recall_{E^F}} \quad (4)$$

The FScore calculated is sorted and the attribute with highest FScore is listed to the user as query refinement.

V. Evaluvation

The goal of our evaluation is to verify the following hypotheses:

H1: Is DQF more usable than existing approaches such as static query form and customized query form?

H2: Is DQF more effective to rank projection and selection components than the baseline method and the random method ?

H3: Is DQF efficient to rank the recommended query form components in an online user interface?

VI. System Implementation and Experimental Setup

We implemented the dynamic query forms as a web based system using JDK 1.6 with Java Server Page. The dynamic web interface for the query forms used open-source javascript library jQuery 1.4. We used MySQL 5.1.39 as the database engine. All experiments were run using a machine with Intel Core 2 CPU @2.83GHz, 3.5G main memory, and running on Windows XP SP2.

Data sets: 3 databases: NBA 1, Green Car 2 and
Geobase 3 was used in our experiments.

6.1 User study Result

Usability Metrics: In this paper, we employ some widely used metrics in Human-Computer interaction and Software Quality for measuring the usability of a system [31], [27]. These metrics are listed in Table 1.

Metric	Definition
MN_{min}	The minimal no of actions for users
MN	The actual no of actions performed by the users
MN_{ratio}	MN_{min}/ MNX100%
FN_{max}	The total no of user interface functions provided to user
FN	The actual no of user interface used by the user
FN_{ratio}	FN_{max}/ FNX100%

Table 1: Table of metric.

In database query forms, one action means a mouse click or a keyboard input for a textbox. AC_{min} is the minimal number of actions for a querying task. One function means a provided option for the user to use, such as a query form or a form component. In a web page based system, FN_{max} is the total number of UI components in web pages explored by the user. In this user study, each page at most contains 5 UI components. The smaller AC_{min}, AC, FN_{max}, and FN, the better the usability. Similarly, the higher the AC_{ratio}, FN_{ratio}, and Success, the better the usability. There is a trade-off between AC_{min} and FN_{max}. An extreme case is that, we generate all possible query forms in one web page, the user only needs to choose one query form to finish his(or her) query task, so AC_{min} is 1.

Task	Query
T1	SELECT ilkid, firstname, lastname FROM players
T2	SELECT p.ilkid, p.firstname, p.lastname FROM players p, player_playoffs_career c WHERE p. ilkid = c. ilkid AND c.minutes > 5000
T3	SELECT t.team, t.location, c.firstname, c.lastname, c.year FROM teams t, coaches c WHERE t.team=c.team AND t.location = 'Los Angeles'
T4	SELECT Models, Hwy_MPG FROM cars WHERE City_MPG > 20
T5	SELECT Models, Displ, Fuel FROM cars WHERE Sales_Area = 'CA'
T6	SELECT Models, Displ FROM cars WHERE Veh_Class ='SUV'
T7	SELECT Models FROM cars WHERE Drive = '4WD'

Table 2: Table of Query Task.

6.2 Usability Result

Task	MN_{min}	MN	MN_{ratio}	FN_{max}	FN	FN_{ratio}	Success
T1	6	6.7	90.0%	40.0	3	7.5%	97.33%
T2	7	7.7	91.0%	65.5	4	6.2%	98.00%
T3	10	10.7	93.5%	133.4	6	3.8%	98.00%
T4	11	11.7	94.3%	71.1	6	8.4%	98.4%
T5	5	5.8	87.7%	28.9	3	10.6%	99.32%
T6	7	7.7	91.0%	61.2	4	6.5%	98.20%

Table 3: Usability result of dynamic query form with query refinement and database encryption.

Task	Query Form	MN_{min}	MN	MN_{ratio}	FN_{max}	FN	FN_{ratio}	Success
T1	DQF	6	6.7	90.0%	40.0	3	7.5%	97.33%
	CQF	6	7.1	85.7%	60.0	3	5%	96.54%
	SQF	1	1.0	100%	35.0	3	7.5%	54.44%
T2	DQF	7	7.7	91.0%	65.5	4	6.2%	98.00%
	CQF	8	10.0	80.1%	86.3	4	4.6%	98.00%
	SQF	1	1.0	100.0%	38.4	4	10.4	16.7%
T3	DQF	10	10.7	93.5%	133.4	6	3.8%	98.00%
	CQF	12	13.3	94.85%	183	6	4.9%	98.5%
T4	DQF	11	11.7	94.3%	71.1	6	8.4%	98.4%
	CQF	12	13.3	90.2%	103	6	5.8%	92.4%
	SQF	1	1.0	100%	70.0	6	8.6%	44.4%
T5	DQF	5	5.8	87.7%	28.9	3	10.6%	99.32%
	CQF	6	6.7	90.0%	56.7	3	5.3%	98.45%
	SQF	1	1.0	100.0%	10.0	3	30.0%	59.87%
T6	DQF	7	7.7	91.0%	61.2	4	6.5%	98.2%
	CQF	8	10.1	80.0%	61.9	4	6.5%	98.2%
	SQF	1	1.0	100.0%	23.3	4	7.2%	54.00%

Table 4: Comparison of dynamic query form with static query form and customized query form.

VII. Conclusion and Future Work

In this paper we propose a dynamic query form generation approach which helps users dynamically generate query forms. The key idea is to use a probabilistic model to rank form components based on user preferences. We capture user preference using both

Historical queries and run-time response such as click through. Experimental results show that 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 can extend our approach can be extended to non relational data.

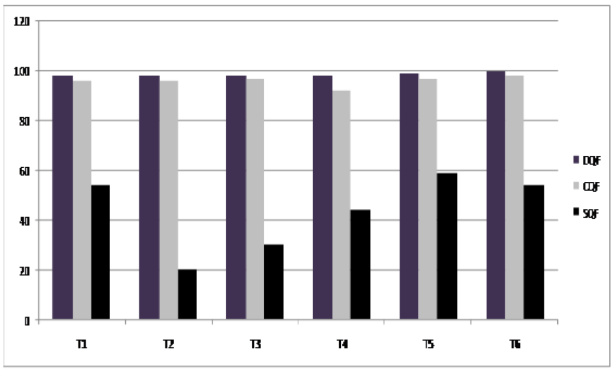

Fig 2. Comparison of dynamic query form system with static query form and customized query form.

Acknowledgement

First of all I would like to thank almighty god. I would like to thank my guide Bhruguram TM Assistant professor in information technology in Adi Shankara institute of Engineering and Technology, Kalady .Also to my friends and family.

References

[1]. Liang Tang, Tao Li, Yexi Jiang, and Zhiyuan Chen Dynamic Query Forms for Database Queries, in proceedings of TKDE.2013.62,pages 1041-4347,U.S.A,June 2013
[2]. DBPedia. http://DBPedia.org.
[3]. EasyQuery. http://devtools.korzh.com/eq/dotnet/.
[4]. Freebase. http://www.freebase.com.
[5]. 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.

[6]. R. Agrawal, S. Gollapudi, A. Halverson, and S. Ieong. Diversifying search results. In Proceedings of WSDM, pages 5–14,Barcelona, Spain, February 2009.

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

[8]. S. Boriah, V. Chandola, and V. Kumar. Similarity measures for categorical data: A comparative evaluation. In Proceedings of SIAM International Conference on Data Mining (SDM 2008), pages 243–254, Atlanta, Georgia, USA, April 2008.

[9]. 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.

[10]. S. Chaudhuri, G. Das, V. Hristidis, and G. Weikum. Probabilistic information retrieval approach for ranking of database query results. ACM Trans. Database Syst. (TODS), 31(3):1134–1168, 2006.

[11]. 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.

[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.

[13]. 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.

[14]. G. Das and H. Mannila. Context-based similarity measures for categorical databases. In Proceedings of PKDD 2000, pages 201–210, Lyon, France, September 2000.

[15]. W. B. Frakes and R. A. Baeza-Yates. Information Retrieval: Data Structures and Algorithms. Prentice-Hall, 1992.

[16]. 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.

[17]. 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.

[18]. M. Jayapandian and H. V. Jagadish. Automating the design and construction of query forms. IEEE TKDE, 21(10):1389–1402, 2009.

[19]. T. Joachims and F. Radlinski. Search engines that learn from implicit feedback. IEEE Computer (COMPUTER), 40(8):34–40,2007.

[20]. 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.

[21]. N. Khoussainova, Y. Kwon, M. Balazinska, and D. Suciu. Snipsuggest: Context-aware autocompletion for sql. PVLDB, 4(1):22–33, 2010.

[22]. 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. Diskin, 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.

[23]. 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.

[24]. B. Liu and H. V. Jagadish. Using trees to depict a forest. PVLDB, 2(1):133–144, 2009.

[25]. 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.

[26]. A. Nandi and H. V. Jagadish. Assisted querying using instantresponse interfaces. In Proceedings of ACM SIGMOD, pages 1156–1158, 2007.

[27]. J. Nielsen. Usability Engineering. Morgan Kaufmann, San Francisco, 1993.

[28]. D. Rafiei, K. Bharat, and A. Shukla. Diversifying web search results. In Proceedings of WWW, pages 781–790, Raleigh, North Carolina, USA, April 2010.

[29]. 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.

[30]. G. Salton and M. McGill. Introduction to Modern Information Retrieval. McGraw-Hill, 1984.J. Crank, The Mathematics of Diffusion, Clarendon Press, Oxford, 1975.