ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 1, January 2015

A User's Feedback Relevant Dynamic Query Forms for Database Queries

Neenu Anna Philip[1], S.S Jaya[2], Pradeep G[3]

P.G. Scholar, Department of CSE, R.V.S. College of Engineering and Technology, Coimbatore, India[1].

Assistant Professor, Department of CSE, R.V.S. College of Engineering and Technology, Coimbatore, India[2].

P.G. Scholar, Department of CSE, R.V.S. College of Engineering and Technology, Coimbatore, India[3].

ABSTRACT: Scientific databases and web databases maintain large and diverse data. Real-world databases contain over hundreds or even thousands of relations and attributes. Old predefined query forms don't seem to beable to satiate various ad-hoc queries from users on those databases. A novel user's feedback relevant dynamic database query form interface is implemented, which is able to dynamically generate query forms result. The system is to capture a user's preference and rank query form components, helping him/her to take decisions. The generation of a query form is an iterative process and is guided by the user. The ranking of form components is based on the captured user preference. A user can fill the query form and submit queries to view the query result at each iteration. In this way, a query form results could be dynamically refined till the user satisfies with the query results.

KEYWORDS: Query form, user interaction, query form generation

I. INTRODUCTION

One of the most extensively used user interfaces for querying databases to access information is query form. Historic query forms are configured and predefined by developers or Database Administrator in different information management systems. With the fast development of web information and scientific databases, new databases become very huge and difficult. In natural sciences, like genetics and diseases, the databases have variety of entities for chemical and/or biological data resources. Diversekinds of web databases have thousands of structured web entities. Therefore, it is difficult to plan a set of static query forms to answer various ad-hoc database queries on those difficult and complex databases.

Existing tools for database management and development, such as SAP and Microsoft Access,users create customized queries on databases. However, the design of customized queries entirely depends on user's manual editing .If a user is not familiar with the database schema prior to, those hundreds or thousands of data attributes would confuse him/her.

1.1 Approach

A user's feedback relevant dynamic query form system is implemented, a query interface which is capable of dynamically generating query forms for users. Users in database retrieval are often willing to perform many rounds of actions (i.e., refining query conditions) before identifying the final candidates.

The essence of this is to capture user interests during user interactions and to adapt the query results iteratively. It starts with a basic query form which contains very few primary attributes of the database. Figure 1 shows user's feedback relevant dynamic query form system'sflow. The basic query form is then iteratively enriched via the interactions between the user and our system till the user is content with the query results.

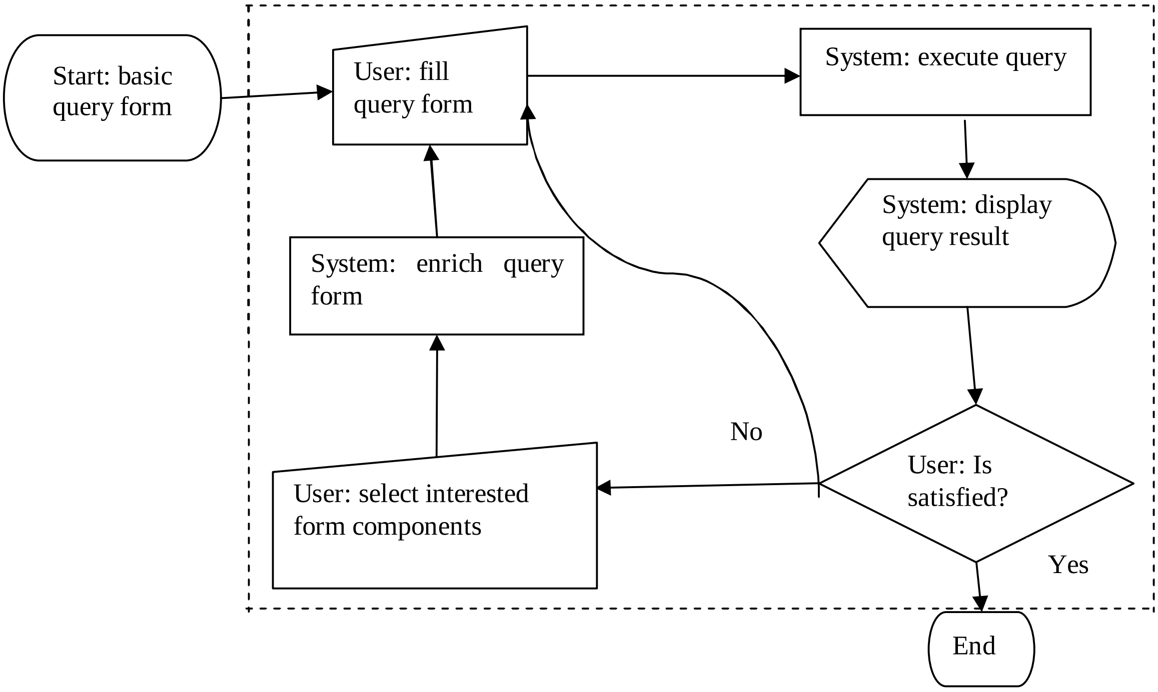


Figure. 1 System flow chart

II. RELATED WORK

Customized Query Form:Existing database clientsand tools make great efforts to help developers design and generate the query forms, such as EasyQuerySAP, Microsoft Access and so on. They deliver visual interfaces for developers to create or customize query forms. The difficulty of those tools is that, they are provided for the professional developers who are familiar with their databases, not for end-users .Proposed a system which allows end-users to customize the existing query form at run time. However, an end-user may not be aware with the database. For huge database schema is, it is difficult for them to catch appropriate database entities and attributes and to create desired query forms.

Auto-completion for Database Queries:In user interfaces have been developed to assist the user to type the database queries based on the query load, the data distribution and the database schema. Different from this which focuses on query forms, the queries in their work are in the forms of SQL and keywords.

III.QUERY FORM

3.1 Interface for Query form

Each query form corresponds to an SQL query template. A query form F is defined as a tuple (A_F, R_F, σ_F, $\bowtie(R_F)$), which represents a database query template as follows:
F= (SELECT $A_1, A_2, ..., A_k$FROM $\bowtie(R_F)$ WHERE σ_F),where $A_F = \{A_1, A_2, ..., A_k\}$ are k attributes for projection, k >0. $R_F = \{R_1, R_2, ..., R_n\}$ is the set of n relations (or entities) involved in this query, n >0. Each attribute in $A_F$belongs to one relation in R_F. $\sigma_F$is a conjunction of expressions for selections (or conditions) on relations in R_F. $\bowtie(R_F)$ 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. $\sigma_F$is the set of input components to be filled by users. Query forms allow users to fill parameters to produce different queries. $R_F$and $\bowtie(R_F)$ are not visible in the user interface, which are usually created by the system according to the database schema. For a query form F, $\bowtie(R_F)$ 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

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 1, January 2015

query form F are actually determined by A_F and σ_F. Only $A_F$and $\sigma_F$are visible to the user in the user interface. In this, the focus is on projection and selection components of a query form.

3.2 Ranking Estimation

The user's desired result is returned by query form. Two measures to evaluate the quality of the query results: precision and recall . Query forms yield different queries by different inputs, and different queries can output different query results and achieve different precisions and recalls, so expected precision and expected recall is used to evaluate the expected performance of the query form. Expected proportion of the query results which are interested by the current user is the expected precision. Expected proportion of user interested data instances which are returned by the current query form is the expected recall. The user interest is estimated based on the user's click-through on query results displayed by the query form.

3.2.1 Ranking of Attributes

Suggesting projection components is actually suggesting attributes for projection. The current query form be F_i, the next query form be F_{i+1}. Let $A_{Fi} = \{A_1, A_2, ..., A_j\}$, and $A_{Fi+1} = A_{Fi} \cup \{A_{j+1}\}$, $j+1 \leq |A|$. A_{j+1} is the projection attribute suggest for the F_{i+1}, which maximizes $FScore_E(F_{i+1})$. $FScore_E(F_{i+1})$ is obtained as follows:

$$FScore_E(F_{i+1}) = (1+\beta^2) . \frac{Precision_E(F_{i+1}) . Recall_E(F_{i+1})}{\beta^2 .Precision_E(F_{i+1}) .Recall_E(F_{i+1})}$$

$$= \frac{(1+\beta^2).\Sigma_{d\epsilon DAFi+1} P_u(d_{AFi+1}) P(d_{AFi+1}) P(\sigma_{Fi+1} d)}{\Sigma_{d\epsilon D} P(d_{AFi+1}) P(\sigma_{Fi+1}|d) + \beta^2 \alpha}$$

Adding a projection component A_{j+1} does not affect the selection part of F_i. Hence, $\sigma_{Fi+1} = \sigma_{Fi}$ and $P(\sigma_{Fi+1}|d) = P(\sigma_{Fi}|d)$. Since F_i is already used by the user, estimate $P(d_{AFi+1})P(\sigma_{Fi+1}|d)$ as follows. For each query submitted for form F_i, keep the query results including all columns in R_F. Clearly, for those instances not in query results their $P(\sigma_{Fi+1}|d) = 0$ and do not need to consider them. For each instance d in the query results, simply count the number of times they appear in the results and $P(d_{AFi+1})P(\sigma_{Fi+1}|d)$ equals the occurrence count divided by N.

$P_u(d_{Aj+1}|d_{AFi})$ is not visible in the runtime data, since d_{Aj+1} has not been used before F_{i+1}.The conditional probability $P_u(d_{Aj+1}|d_{AFi})$ is estimated from following approach.

- Workload-Driven based approach: The conditional probability of $P_u(d_{Aj+1}|d_{AFi})$ could be estimated from query results of historic queries. If a lot of users queried attributes A_{Fi} and A_{j+1} together on instance d, then $P_u(d_{Aj+1}|d_{AFi})$ must be high.
- Schema-Driven based approach: The database schema implies the relations of the attributes. If two attributes are contained by the same entity, then they are more relevant.

The schema graph is utilized to compute the relevance of two attributes. A database schema graph is denoted by G = (R,FK,ξ,A), in which R is the set of nodes representing the relations, A is the set of attributes, FK is the set of edges representing the foreign keys, and $\xi : A \rightarrow R$ is an attribute labelling function to indicate which relation contains the attribute.

3.2.2 Ranking Entities

The ranking score of an entity is just the averaged $FScore_E(F_{i+1})$ of that entity's attributes. Intuitively, if one entity has many high score attributes, then it should have a higher rank.

3.3 Relevant selection based on user feedback

The selection attributes must be relevant to the current projected entities, otherwise that selection would be meaningless. Therefore, first the relevant attributes is find out for creating the selection components.

3.3.1 Relevant Attribute Selection

The relevance of attributes in system is measured based on the database schema as follows.

Relevant Attributes: For a database query form F with a schema graph G=(R,FK,ξ,A), the relevant attributes is: $A_r(F) = \{A | A \in A, \exists A_j \in A_F, d(A,A_j) \leq t\}$, where t is a user-defined threshold and $d(A,A_j)$ is the schema distance.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 1, January 2015

The choice of t depends on how compact of the schema is designed. For instance, some databases put all attributes of one entity into a relation, then t could be 1. Some databases separate all attributes of one entity into several relations, then t could be greater than 1.

3.3.2 Ranking Selection Components

For enriching selection form components of a query form, the set of projection components $A_F$is fixed, i.e., $A_{Fi+1} = A_{Fi}$. Therefore, $FScore_E(F_{i+1})$ only depends on σ_{Fi+1}. To find the best selection component for the next query form, the first step is to query the database to retrieve the data instances. $P(\sigma_{Fi+1}|d)$ depends on the previous query conditions σ_{Fi}. If $P(\sigma_{Fi}|d) = 0$, $P(\sigma_{Fi}+1|d)$ must be 0. In order to compute the $P(\sigma_{Fi+1}|d)$ for each $d \in D$, no need to retrieve all data instances in the database. The set of data instances $D' \subseteq D$ is only needed such that each $d \in D'$ satisfies $P(\sigma_{Fi}|d) > 0$. So the selection of One-Query's query is the union of query conditions executed in F_i. One-Query adds all the selection attributes into the projections of the query.

Query Creation:

Data: $Q = \{Q_1, Q_2, \ldots.\}$ is the set of previous queries executed on F_i

Result: Q_{one} is the query of One-Query

begin

$\sigma_{one} \leftarrow 0$

for Q ε Q do

$\sigma_{one} \leftarrow \sigma_{one} \vee \sigma_Q$

$A_{one} \leftarrow A_{Fi} \cup A_r(F_i)$

$Q_{one} \leftarrow$ GenerateQuery (A_{one}, σ_{one})

When the system receives the result of the query Q_{one} from the database engine, it calls the second algorithm of One-Query to find the best query condition the system. The pseudocode for finding best" $\leq$ " condition is as follows:

Find Finest Less Equation Condition

Data**:** α is the fraction of instances desired by user, D_{Qone} 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 As into an ordered set D_{sorted}

$D_{sorted} \leftarrow$ Sort (D_{Qone}, A_s)

$s^* \leftarrow \emptyset$, $fscore^* \leftarrow 0$

$n \leftarrow 0$, $d \leftarrow \alpha\beta^2$

for $i \leftarrow 1$ to $|D_{sorted}|$ do

$d \leftarrow D_{sorted}[i]$

$s \leftarrow$ "$A_s <= d_{As}$"

// compute fscore of "$A_s <= d_{As}$"

$n \leftarrow n + P_u(d_{AFi})\, P(d_{AFi})\, P(\sigma_{Fi} \mid d)\, P(s|d)$

$d \leftarrow d + P(d_{AFi})\, P(\sigma_{Fi} \mid d)\, P(s|d)$

$fscore \leftarrow (1+\beta^2).\, n/d$

if fscore >= fscore* then

$s^* \leftarrow s$

$fscore^* \leftarrow fscore$

3.4 Evaluation

To evaluate the quality of the query results mainly : precision and recall is used. Different inputs through query forms produce different queries, and different queries produce different query results and achieve different precisions and recalls, so to evaluate the expected performance of the query form expected precision and expected recall is used. The expected proportion of the query results which are interested by the current user is Expected precision.

$$Precision_E(F) = \frac{\Sigma_{d\epsilon DAF}\, P_u(d_{AF})\, P(d_{AF})\, P(\sigma_F|d)\, N}{\Sigma_{d\epsilon DAF}\, P(d_{AF})\, P(\sigma_F|d)\, N}$$

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

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 1, January 2015

$$\text{Recall}_E(F) = \frac{\Sigma_{d\epsilon DAF} P_u(d_{AF})\, P(d_{AF})\, P(\sigma_F|d)\, N}{\alpha N}$$

Considering both precision and recall, Fscore is derived as follows:

$$\text{FScore}_E\,(F) = (1+\beta^2)\,.\ \frac{\text{Precision}_E(F)\ .\ \text{Recall}_E(F)}{\beta^2.\text{Precision}_E(F)\ .\ \text{Recall}_E(F)}$$

The user interest is based on the user's click-through on query results displayed by the query form.

III. RESULT

The query form captures user's feedback. User select components and on that selection basis query result is generated. User can refine component selection for a refined query results. The basic query form is enriched iteratively through the interaction between the user and the system until the user is satisfied with the query results. Ranking of form components is done and query results are generated.

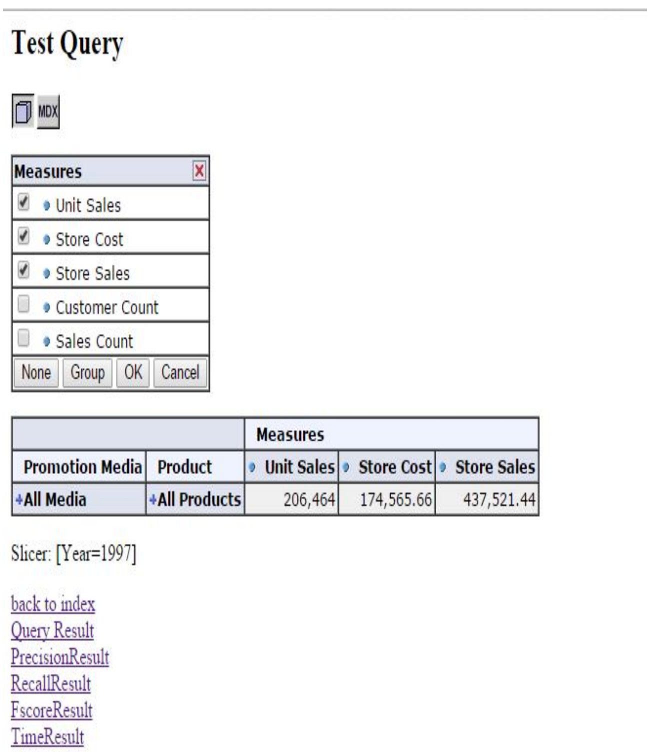


Figure.1 Column component selection

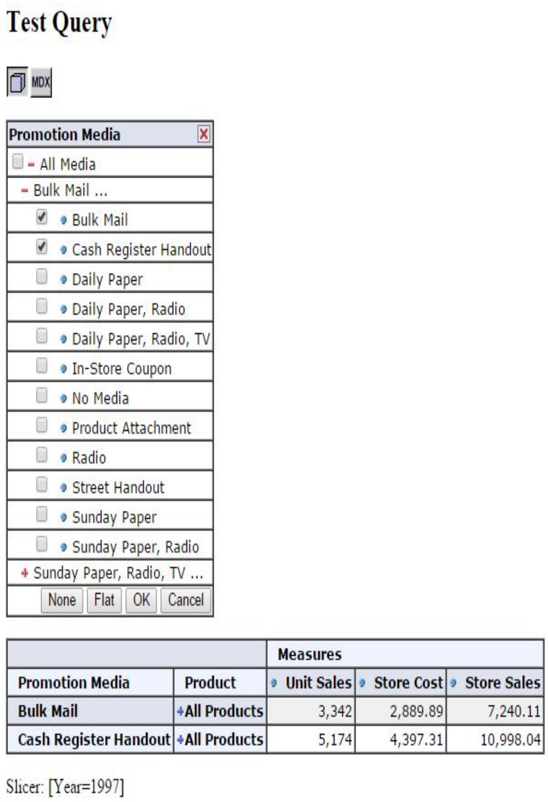


Figure.2 Row component selection

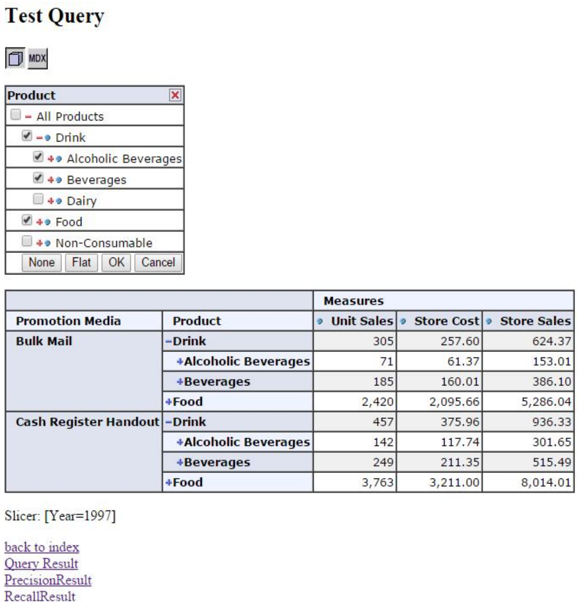
In the above figures, user selects desired column component and row component in query form for the desired results.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

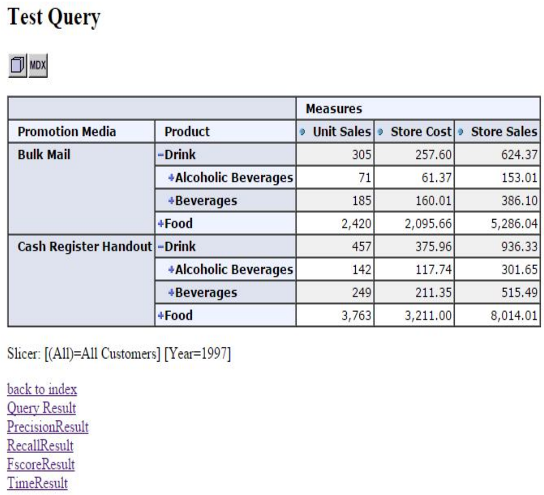
(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 1, January 2015



		Measures		
Promotion Media	Product	Unit Sales	Store Cost	Store Sales
Bulk Mail	-Drink	305	257.60	624.37
	+Alcoholic Beverages	71	61.37	153.01
	+Beverages	185	160.01	386.10
	+Food	2,420	2,095.66	5,286.04
Cash Register Handout	-Drink	457	375.96	936.33
	+Alcoholic Beverages	142	117.74	301.65
	+Beverages	249	211.35	515.49
	+Food	3,763	3,211.00	8,014.01

Figure.3 Refined row component selection



		Measures		
Promotion Media	Product	Unit Sales	Store Cost	Store Sales
Bulk Mail	-Drink	305	257.60	624.37
	+Alcoholic Beverages	71	61.37	153.01
	+Beverages	185	160.01	386.10
	+Food	2,420	2,095.66	5,286.04
Cash Register Handout	-Drink	457	375.96	936.33
	+Alcoholic Beverages	142	117.74	301.65
	+Beverages	249	211.35	515.49
	+Food	3,763	3,211.00	8,014.01

Figure.4 Query result

More refined row component selection is done for the refined query results. Finally after making the relavant component seleciton refined query results is obtained.

IV. CONCLUSION AND FUTURE WORK

A user's feedback relevant dynamic query form generation is done which helps users dynamically generate query forms. The vitalnotion is to use a probabilistic model to rank form components based on user preferences. User preference is captured using both historical queries and run-time feedback such as click. The ranking of form components makes it easier for users to customize query forms. As future work, this approach can be extended to non-relational data. Multiple methods to capture the user's interest for the queries besides the click feedback is planned to develop. For instance, a text-box can be added for users to input some keywords queries.

REFERENCES

1. Liang Tang, Tao Li ; Yexi Jiang; Zhiyuan Chen, "Dynamic Query Forms for Database Queries", Knowledge and Data Engineering, IEEE Transactions on (Volume:26 , Issue: 9)
2.EasyQuery.http://devtools.korzh.com/eq/dotnet.
3. A. Nandi and H. V. Jagadish "Assisted querying using instantresponseinterfaces". In Proceedings of ACM SIGMOD, pages1156–1158, 2007
4.Yun Zhou and W. Bruce Croft "Ranking Robustness: A Novel Framework to Predict Query Performance" CIKM'06, November 5–11, 2006, Arlington, Virginia, USA
5. Ricardo Baeza-Yates, Berthier Ribeiro-Neto"Modern information retrieval" Publish Addison –Wesley

BIOGRAPHY

Neenu Anna Philipis pursuing Master of Engineering in Computer Science Engineering in R.V.S College of Engineering and Technology, Coimbatore,India.She received her M.Sc. degree in Computer Science from Assumption College,Mahatma Gandhi University,Kerala in 2012.

S.S.Jaya is an Assistant Professor in Department of Computer Science,R.V.S College of Engineering andTechnology,Coimbatore,India. Her area of interest include Data Structures and Data Mining.

Pradeep G is pursuing Master of Engineering in Computer Science Engineering in R.V.S College of Engineering and Technology, Coimbatore,India.