ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization   Volume 3, Special Issue 5, July 2014

International Conference On Innovations & Advances In Science, Engineering And Technology [IC - IASET 2014]

Organized by

Toc H Institute of Science & Technology, Arakunnam, Kerala, India during 16th -18th July -2014

Dynamic Querying In NoSQL System

Reshma Suku[1], Kasim K.[2]

Student, Dept. of Computer Science and Engineering, M. E. A Engineering College, Perinthalmanna, Kerala, India[1]

Assistant Professor, Dept. of Computer Science and Engineering, M. E. A Engineering College, Perinthalmanna, Kerala, India [2]

ABSTRACT: Three mege trends Big data, Big user and cloud computing that leads to the adoption of NoSQL. NoSQL simply means Not Only SQL. Today most of the applications are hosted in cloud and that available through internet. They must support large number of users 24 hours a day, 365 days a year. This create explosion in number of concurrent users. So here needs a technique to handle large number of data. This paper proposes a dynamic query form interface for database exploration of an organization. Here use a document oriented NoSQL database ie, MONGODB. MONGODB support dynamic queries that do not require predefined map reduce function. The generation of a query form is an iterative process guided by user. At each iteration, system automatically generate ranking list of form components and user adds the desired form component into query form and submit queries to view query result. There are two traditional measures to evaluate the quality of query result ie, precision and recall. From the quality measures we can derive overall performance measure as F-measure.

KEYWORDS: query form , user interaction , F-measure.

I. INTRODUCTION

Modern scientific database and web database maintain large and heterogeneous data. Web databases include Freebase[5] and DBPedia[3] have thousands of structured web entities. These real world database contain over hundreds or even thousands of relations and attributes. Traditional predefined query forms are not able to satisfy various ad-hoc queries on those complex database. Query form is one of the most widely used user interface to return users desired result.

In this paper, here proposes Dynamic Query Form System[1], a query form interface which is able to dynamically generate query form according to user desire at run time. This is entirely different from document retrieval, users in database retrieval perform many refining query condition before identifying the final candidates.

Interactive applications have changed dramatically over last 15 years. In the late 90's, large web companies deals with increase in scale on many dimensions inclue:

i. Big Data
ii. Big User
iii. Cloud Computing

Dealing with these issues was more difficult using relational database technology. The key reason behind is that relational database are architected to run on a single machine and use a rigid, schema based approach to modelling data. Google, Amazon, Facebook and LinkedIn were among the first companies to discover the limitation of relational database technology for supporting new application requirement.

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 3, Special Issue 5, July 2014

International Conference On Innovations & Advances In Science, Engineering And Technology [IC - IASET 2014]

Organized by

Toc H Institute of Science & Technology, Arakunnam, Kerala, India during 16th -18th July -2014

In this paper, use MONGODB[14] instead of relational base. It is a cross platform, document oriented database that provide high performance, high availability and scalability. MONGODB works on the concept of collection and document. Document oriented database are different from traditional relational database. Rather than store data in rigid structures like tables, they store data in loosely defined document. With relational database management system table, if we need to add new column, we need to change the definition of table itself, which will add that column to every existing record. This is due to RDBMS strict schema based design. However, with documents we can add new attributes to individual document without changing any other documents. This is because document oriented database are generally schema less by design.

II. RELATED WORKS

In Assisted querying using instant response interfaces, A.Nandi and H.V. Jagdish[6], This proposed auto completion approaches for database queries. Here a novel user interface have been developed to assist the user to type the database queries based on query workload, the data distribution and the database schema.

In automated creation of a form-based database query interface and In expressive query specification through form customization, M. Jayapandian and H.V. Jagadish[7][8], Here proposed a customized approach to provide visual interface for developers to create or customize query form. EasyQuery, ColdFusion, SAP are the main tools to help developers design and generate the query form. The problem of these tool is that, they are provided for the professional developers who are familiar with their database not for end users. Another problem is that, if the database schema is very large, it is difficult for them to find appropriate database entities, attributes and to create desired query form.

In Combining keyword search and forms for adhoc querying of database, E.Chu, A.Baid X. Chai, Doan and J.F. Naughton[10], It automatically generate a lot of query form in advance. Here user input several keywords to find relevant query form. This leads to the conclusion that a query rewrite by mapping data values to schema values during keyword search. Another one is that, simply displaying the returned form as a flat list.

In automating the design and construction of query forms, M.Jayapandian and H.V. Jagadish[9], Here propose automatic approaches to generate the database query without user participation. It is a work-load driven model. It applies clustering algorithm to find representative queries. One of the disadvantage is that if we generate lots of query forms in advance, there are still user queries that cannot be satisfied by one of the query form.

Query recommendation for interactive database exploration, M. Eirinaki and N. Polyzotis[11], Here introduce a collaborative approach to recommend database query components for database exploration. They treat SQL queries as item in the collaborative filtering approach and recommend similar queries to related users. One of the problem is that they do not consider the goodness of query result. In proposed system recommendation is a query component for each iteration.

In Building dynamic faceted search systems over database, S.B. Roy, H. Nambiar, G. Das and M.K. Mohania[12], a domain independent system, that provides effective minimum- effort based dynamic faceted search solution over enterprise database. It present relevant facets for the users according to navigation path. Dynamic faceted search engine are similar to our dynamic query form if we only consider selection components in query.

In Usher: Improving Data Quality with dynamic forms, K.Chen, H.Chen, N.Conway, J.M. Hellerstein and T.S. Parikh[13], Data quality is a critical problem in modern database. USHER, an end to end system for form design, entry and data quality assurance. In DQF, deals with database query forms instead of data entry forms.

III. PROBLEM FORMULATION

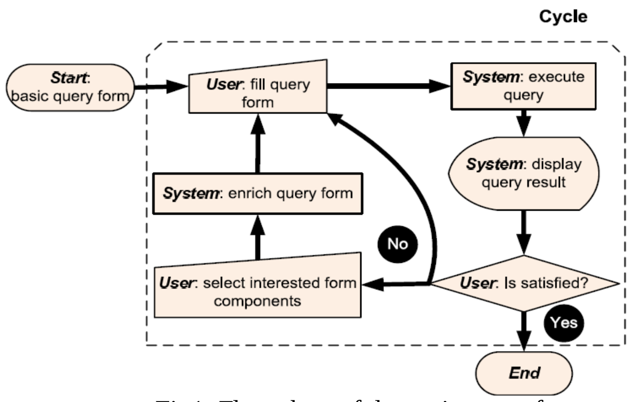


Fig1: Flow chart of dynamic query form

Figure shows flow chart of dynamic query form. A dynamic query form system which generates query form according to the users desire at run time. The system provides a solution for query interface in large and complex database. Apply F-measure to estimate the goodness of a query form. F-measure is a typical metric to evaluate query result. The metric is also appropriate for query form 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 can rank and recommend the potential query form components. Here efficiency is important because dynamic query form is an online system where users often expect quick response.

Each query form corresponds to SQL query template. Query forms allow users to fill parameters to generate different queries. 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. To decide whether a query form is desired or not, a user does not have time to go over every data instance in the query result. In addition, many database queries output a huge amount of data instances. In order to avoid this many-answer problem, only outputcompressed result table to show a high level view of the query result first. Each instance in the compressed table represents a cluster of actual data instances. 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. In our implementation, we choose the incremental data clustering framework, because of the efficiency issue. Certainly, different data clustering methods would have different compressed views for users. Also, different clustering methods are preferable to different data types. Here, clustering is just to provide a better view of the query result for user. The system developers can select a different clustering algorithm if needed. Another important usage of the compressed view is to collect the user feedback. Using the collected feedback, 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. Figure below shows the user action.

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 3, Special Issue 5, July 2014

International Conference On Innovations & Advances In Science, Engineering And Technology [IC - IASET 2014]

Organized by

Toc H Institute of Science & Technology, Arakunnam, Kerala, India during 16th -18th July -2014

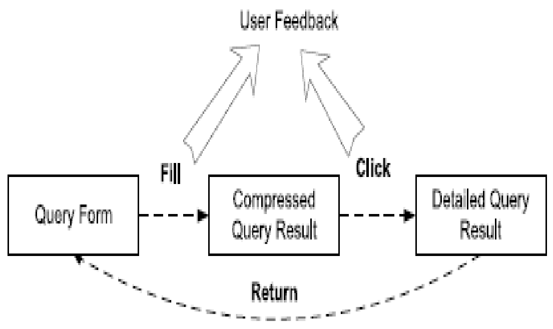


Fig.2 User action

IV. METHODOLOGY

i). Component ranking

DQF has two-level ranked list for projection components. The first level is the ranked list of entities. Second level is the ranked list of attributes in the same entity. For ranking attribute we have to calculate F-Score. This can be calculated from corresponding precision and recall as follows. Let current query form be F_i, the next query form be F_{i+1}, we obtain FScore(F_{i+1}) as follows:

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

Here apply kernel density estimation method to estimate user interest on instances in the query result. We utilize the schema graph to compute the relevance of two attribute. The ranking score of an entity is just average of $FScore(F_{i+1})$ of that entity's attributes. If one entity has many high score attribute, then it should have a higher rank.

The selection attribute are relevant to the current projected entities, otherwise selection would be meaningless. Relevant attributes can be selected based on the database schema. For simplicity of user interface, most of the query form's selection component are simple binary relation in the form "A_j *op* C_j". Where "A_j" is an attribute, "C_j" is a constant and "*op*" is a relational operator. The following algorithm describes the algorithm of one-query's query construction. One –Query adds all the selection attributes into projection of query. The function "Generate Query" is to generate the database query based on projection attribute "A_{one}" and selection expression" σ_{one}".

When the component attributes were ranked, we can determine the probability of user interested form components. Here we can use schema graph to show relevance of attribute. When comes to entity ranking, if one entity has many high score attribute, then it should have higher rank.

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 3, Special Issue 5, July 2014

International Conference On Innovations & Advances In Science, Engineering And Technology [IC - IASET 2014]

Organized by

Toc H Institute of Science & Technology, Arakunnam, Kerala, India during 16th -18th July -2014

Algorithm 1: Query Construction

```
Data: Q={$Q_1$, $Q_2$, ……………..,} is a set of previous queries executed on $F_{i+1}$
Result: $Q_{one}$ is the query of one-Query
begin
    $\sigma_{one} \leftarrow 0$
    for $Q \in Q$ do
        $\sigma_{one} \leftarrow \sigma_{one} \vee \sigma Q$

    $A_{one} \leftarrow A_{Fi} \cup A_r(F_r)$
    $Q_{one} \leftarrow$ GenerateQuery($A_{one}$,$\sigma_{one}$)
```

ii). Quality Metric Module

The quality of query result can be described by paying more importance to precision and recall. Precision is also called positive predicate value. It is the fraction of retrieved instance that are relevant. Recall is also called sensitivity. Recall is the fraction of relevant instance that are retrieved. Query forms are able to produce different queries by different input and different queries can output different query result and achieve different precision and recall. Hence we use expected precision and expected recall to evaluate expected performance of query form. Probabilistic model can be used to find precision and recall.

iii). Metaprocessor Module

Metadata can be defined as the data providing information about one or more aspects of the data. This provides user friendly interface to novel users. Map-reduce function is used to extract keys from collection. In map-reduce operation, MONGODB applies map phase to each input documents. The 'map' function emit key-value pairs. For those keys have multiple values, MONGODB applies the 'reduce' phase, which collect and condense the aggregated data. All map reduce functions in MONGODB are javascript and run within the mongod process.

iv). Query Processor Module

The essence of DQF is to capture users interest during user interactions and adapt the query form iteratively. In each iteration the user interaction be two types as follows:

Query Form Enrichment

1. DQF recommends a ranked list of query form components to the user.
2. The user select the desired form components into the current query form.

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization   Volume 3, Special Issue 5, July 2014

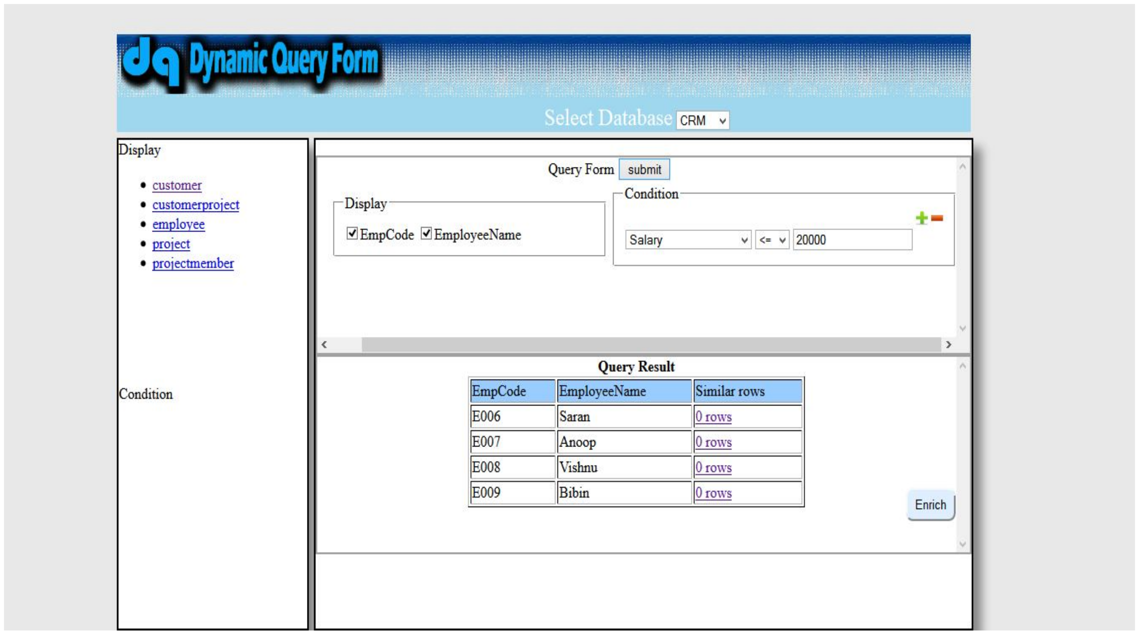
International Conference On Innovations & Advances In Science, Engineering And Technology [IC - IASET 2014]

Organized by

Toc H Institute of Science & Technology, Arakunnam, Kerala, India during 16th -18th July -2014

Query Execution

1. The user fills out the current query form and submit a query.
2. DQF executes the query and shows the result.
3. The user provides the feedback about the query result.



EmpCode	EmployeeName	Similar rows
E006	Saran	0 rows
E007	Anoop	0 rows
E008	Vishnu	0 rows
E009	Bibin	0 rows

Fig 3: Data retrieval

SQL to MONGODB mapping chart as follows:

SQL Terms/Concept	MONGODB Tems/Concept
Database	Database
Table	Collection
Row	Document/ BSON Document
Column	Field
Index	Index

Table Join	Embedded document or linking
Primary Key	Primary Key
Specify any unique coloumn or coloumn combination as primary key	In MONGODB, primary key is set to the _id field
Aggregation(Group By)	Aggregation pipeline

V. FACILITIES FOR THE REQUIRED WORK

The system will be implemented as a J2EE application and is a web based system. The application is based on MVC application architecture and is confirmed to the modern n- tier application pattern. The system front end is normal web browser(Internet explorer). This constitutes the “thin client” tier. The browser connects to a web server and web server make use of a Java application server to invoke the JSP components(the “web tier”). These JSP components realize presentation logic for the system. The JSP components in turn call into Java Bean components. The Java Bean components are responsible for implementing the business logic of the application. This is the “business logic tier”. Finally the data persistence is done using an NoSQL system comprising the “data tier”. Here use a document oriented NoSQL such as MONGODB.

A standard server machine will be required which will act as the server machine and a web container like Apache Tomcat will be installed in the server on which the application will be running.

VI. CONCLUSION AND FUTURE WORK

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 feedback such as clickthrough. Experimental result shows that the dynamic approach often leads to higher success rate and simpler query form compared with a static approach. The ranking of form components also makes it easier for users to customize query forms. Here use a NoSQL database such as MONGODB[14].

As for 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. The relevance score between the keywords and the query form can be incorporated into the ranking form components at each step.

REFERENCES

[1]. Dynamic Query Forms for Database Queries Liang Tang, Tao Li, Yexi Jiang and Zhiyuan Chen
[2]. Cold Fusion. http://www.adobe.com/products/coldfusion/.
[3]. DBPedia. http://DBPedia.org
[4]. EasyQuery. http://devtools.korzh.com/eq/dotnet/.
[5]. Freebase. http://www.freebase.com

[6]. A.Nandi and H.V. Jagdish. Assisted querying using instant responseinterfaces. In proceedings of ACMSIGMOD, pages 1156-1158,2007.
[7]. 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.
[8]. 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.
[9]. M. Jayapandian and H. V. Jagadish. Automating the design and construction of query forms. IEEE TKDE, 21(10): 1389-1402,2009.
[10]. E.Chu, A. Baid, X. Chai, A. Doan and J. F. Naughton. Combaining keyword search and form for ad hoc querying of databases. In proceedings of ACM SIGMOD Conference, pages 349-360, providence, Rhode IsLand, USA, June 2009.
[11] 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.
[12]. S. B. Roy, H. Wang, U. Nambiar, G. Das and M. K. Mohsnia. Dynacet: Building faceted search systems over databases. In proceedings of ICDE Conference, pages 1463-1466, Shanghai, China, March 2009.
[13]. K. Chen, H. Chen, N. Conway, J. M. Hellerstein and T. S. Parikh. Usher: Improving data quality with dynamic forms. Im proceedings of ICDE conference, pages 321-332, Long Beach, California, USA, March 2010.
[14]. MONGODBmanual.http://docs.mongodb.org/v2.4/mongodb_manual.pdf