Interactive data exploration with the help of Querie

[*]Deepali D. Shinde, [#]Prof. Dr. V. B. Kambale

[*]P.G. Student, [#]Professor, Department of Computer Sci. & Engg., P.E.S College of Engineering, Aurangabad, India.

ABSTRACT - Database Management Systems interact with the user to capture and to analyze data. The non-expert user of SQL or the user who is not familiar with database schema face great difficulties in analyzing and mining interesting information from this system. Query Recommendation for Interactive Database Exploration (QueRIE) is a recommendation system that supports interactive database exploration. The users who are not familiar with the database schema may face great difficulty in performing this job. This system aims at assisting non expert users of relational databases by generating personalized query recommendations. QueRIE tracks the querying behavior of previous users and identifies similar patterns. These similar query patterns are used to generate recommendations. There are three approaches used in this work for generating query recommendations viz., suggested query by dictionary mapping, tuple based and fragment history. And also the performance is analyzed among the three approaches.

General Terms - Query recommendation, fragmentation and Agglomerative algorithm.

Keywords - Data mining, interactive data exploration and discovery, personalization.

I. INTRODUCTION

Relational database systems are playing vital role in variety of application because of its infrastructure to access and analyze large volume of data. Applying collaborative filtering technique to the database context may face several challenges. First, SQL is a declarative language, there can be two SQL statement which has different syntax but retrieve the same content, so we cannot consider the textual property of queries to find similarity between the users. Similarity among users, since it is opposite to the web paradigm where the similarity between two users can be expressed as the similarity between the items they visit/rate/purchase, we cannot rely directly on the SQL queries. A second important issue is how to create implicit user profiles that represent interested data of that user. This helps to measure the level of importance of the same data for different users. Finally, contrary to the user-based collaborative filtering approach, the recommendations to the users have to be in the form of SQL queries, since those actually describe what the retrieved data represent.

The results of tremendously increased data increases need for data discovery tools. The users often have difficulties in understanding the underlying complicated schema and formulating queries despite the availability of querying tools over large databases. Even though such database systems offer the means to run complex queries over large data sets, the discovery of useful information remains a big challenge. First time user may not have necessary knowledge to know where to start their exploration. Second time, user may simply overlook queries that retrieve important information. To assist the non-expert user for retrieving interesting information, query recommender system is used. The Query Recommender systems elicit the interest of the user and make recommendations accordingly. That recommended query can be used as templates and submitted as it is instead of composing new ones or it can be further refined. This inspiration draws from Web Recommender System. The premise on which the system is built is simple: If user A and user B placed the same queries then the other queries of each user may be of interest of each other.

PROBLEM DEFINITION

To design the query recommendation system to assist the user for exploring database .we are inspired by the web recommender system which uses collaborative filtering technique. The theory on which the system is built is simple: If a new user has similar querying behavior to previous user, then they are likely interested in retrieving the same data. Hence, the queries of previous user can be recommended to help new user to get interested data.

 DOI : 10.35291/2454-9150.2019.0357

II. LITERATURE SURVEY

Any system that produces individualized recommendations as output or has the effect of guiding the user in a personalized way to interesting or useful objects in a large space of possible options is called Recommender system.. In other words "The goal of a recommender system is to provide lists of top N recommended object that is as per user requirement which is evaluated based on predictions.

There are main three types of recommendation systems

1. Collaborative Recommendation System
2. Content Based Recommendation System
3. Hybrid Recommendation System

III. RELATED WORK

Personalized queries under a generalized preference model (G. Koutrika and Y. Ioannidis) find the solution for problem that we face regularly that we present a preference model that combines expressivity and concision. In addition, we provide efficient algorithms for the selection of preferences related to a QueRIE, and an algorithm for the progressive generation of personalized results, which are ranked based on user interest. Several classes of ranking functions are provided for this purpose. We present results of experiments both synthetic and with real users (a) demonstrating the efficiency of our algorithms, (b) showing the benefits of QueRIE personalization, and (c) providing insight as to the appropriateness of the proposed ranking functions.

Amazon.com recommendations: Item-to-item collaborative filtering (G. Linden, B. Smith, and J. York) wrote that we can use recommendation algorithms to personalize the online store for each customer. The store radically changes based on customer interests, showing programming titles to a software engineer and baby toys to a new mother. There are three common approaches to solving the recommendation problem: traditional collaborative filtering, cluster models, and search-based methods. Here, we compare these methods with our algorithm, which we call item-to-item collaborative filtering. Unlike traditional collaborative filtering, our algorithm's online computation scales independently of the number of customers and number of items in the product catalog. Our algorithm produces recommendations in real-time, scales to massive data sets, and generates high quality recommendations.[8]

In Joydeep Sen Sarma, Namit Jain, Zheng Shao, Prasad Chakka, Ning Zhang and Raghotham Murthy, Hive- a Petabyte Scale Data Warehouse Using Hadoop, Hadoop is a popular open-source map-reduce implementation which is being used in companies like Yahoo, Facebook etc. to save and execute every big data sets on commodity hardware. Map-reduce is programming model for processing large data set with a parallel distributed algorithm on a cluster. Hive also includes a system catalog megastore that contains schemas and statistics, which are useful in data exploration, query optimization and query compilation. In Facebook, the Hive warehouse contains tens of thousands of tables and stores over 700TB of data and is being used extensively for both reporting and ad-hoc analyses by more than 200 users per month. The entire data processing infrastructure in facebook prior to 2008 was built around a data warehouse built using a commercial RDBMS. The data that we were generating was growing very fast - as an example we grew from a 15TB data set in 2007 to a 700TB data set today. In current world Hadoop is a technology that addresses our scaling needs. Hadoop was already an open source project that was being used at megabyte scale and provided scalability using commodity hardware was a very compelling proposition for us. The same jobs that had taken more than a day to complete could now be completed within a few hours using Hadoop. Currently author considers only a subset of SQL as valid queries. Authors are working towards making HiveQL subsume SQL syntax. Hive currently has a naive rule-based optimizer with a small number of simple rules [5].

In Badrul Sarwar , George Karypis , Joseph Konstan and John Riedl, Item Based Collaborative Filtering Recommendation Algorithms, Science and Engineering University of Minnesota, Minneapolis, MN 55455 Recommender systems apply knowledge discovery techniques to the problem of making personalized recommendations for information, products or services during a live interaction. These systems, especially the k-nearest neighbor collaborative filtering based ones, are achieving widespread success on the Web. The tremendous growth in the amount of available information and the number of visitors to Web sites in recent years poses some key challenges for recommender systems. These are: producing high quality recommendations, performing many recommendations per second for millions of users and items and achieving high coverage in the face of data sparsity. In traditional collaborative filtering systems the amount of work increases with the number of participants in the system. New recommender system technologies are needed that can quickly produce high quality recommendations, even for very large-scale problems [11].

In Gloria Chatzopoulou, Magdalini Eirinaki and Neoklis Polyzotis, Query Recommendations for Interactive Database Exploration. Users employ a query interface to issue a series of SQL queries that aim to analyze the data and mine it for interesting information. In this paper, authors present a query recommendation framework supporting the interactive exploration of relational databases and an instantiation of this framework based on user-based collaborative filtering. Such queries need to be considered in the recommendation process. First-time users, however, may not have the necessary knowledge to

 DOI : 10.35291/2454-9150.2019.0357

know where to start their exploration. Other times, users may simply overlook queries that retrieve important information. The experimental evaluation demonstrates the potential of the proposed approach. The authors should stress that this is a first-cut solution to the very interesting problem of personalized query recommendations. There are many open issues that need to be addressed. For instance, an interesting problem is that of identifying similar queries in terms of their structure and not the tuples they retrieve. Two queries might be semantically similar but retrieve different results due to some filtering conditions [9].

The Javad Akbarnejad , Gloria Chatzopoulou , Magdalini Eirinaki, Suju Koshy, Sarika Mittal, Duc On, Neoklis Polyzotis and Jothi S. Vindhiya Varman, SQL QueRIE Recommendations, This system aims at assisting non-expert users of scientific databases by tracking their querying behavior and generating personalized query recommendations. The system is supported by two recommendation engines and the underlying recommendation algorithms. The first identifies potentially interesting parts of the database related to the corresponding data analysis task by locating those database parts that were accessed by similar users in the past. The second identifies structurally similar queries to the ones posted by the current user. Both approaches result in a recommendation set of SQL queries that is provided to the user to modify, or directly post to the database. The demonstrated system will enable users to query and get real-time recommendations from the SkyServer database, using user traces collected from the SkyServer query log. QueRIE does not require an explicit user profile or keyword-based queries. On the contrary, it closes the loop by accepting SQL queries as input, decomposing them in order to identify interesting database areas for each user, and re-transforms them in SQL queries that are presented as recommendations.[4]

Purpose and Goal of Proposed System

The goal of this project is to design Personalized Query Recommendation System using Fragment based and tuple based approaches and Hierarchical Clustering algorithm. Literature survey is the first step towards the goal. The observation will focus on the requirements. The next step is to plan the project. Agglomerative algorithm is used for clustering. The tuple-based approach captures the users querying behavior at a very fine level of detail the individual witnesses to the user's queries. The idea behind fragment based approach is to recommend queries whose syntactical features match the queries of the current user, the coordinates of the session summaries correspond to fragments of queries instead of witnesses.

IV. PROPOSED SYSTEM ARCHITECTURE

The fragment based instantiation of the QueRIE framework works in a similar manner to the tuple-based one. The two main differences lie in the representation of the session summaries and the formulation of similarities. More specifically, the coordinates of the session summaries correspond to fragments of queries instead of witnesses. We identify as fragments the following syntactical features of the queries in the session: attribute references, table's references, join and selection predicates. At a high level, the idea behind this approach is to recommend queries whose syntactical features match the queries of the current user.

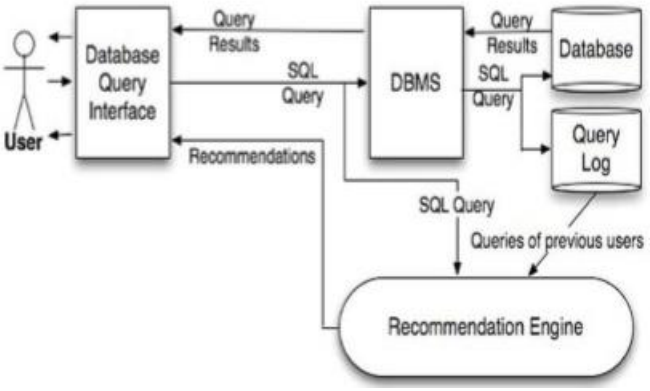

Fig 1: Architecture of QueRIE framework

Figure 1 shows the structure of the QueRIE framework. It includes different modules like database query interface, DBMS, Query log, Recommendation engine.

When a user logs in the system the user have to submit an SQL query to the query interface. The SQL query is sent to the database as well as the recommendation engine. Also the query is fragmented and stored in the system log. The query is processed by the query analyzer and returns the result. At the same time recommendation engine will generate the recommendation that match well with the query.

Modules

Database Monitoring

Each time when a user logs in the system he/she have to submit an SQL query through the database query interface. SQL query which is provided by the user is sent to syntax checker to check SQL syntax then the SQL query is fragmented and then transmitted to both the DBMS and recommendation engine. The input query is fragmented and stored in the system query log. This will make the comparison easier when finding the highly similar patterns from the system query log. Query fragmentation also makes storing the query in the database in an easier way.

Query Analyzer

The DBMS requested to the database and return the results to the user and then query is fragmented, creating an implicit query profile which consists of fragments. The given query is decomposed into fragments with respect to the keywords such as select, from, where, group by, having, order by. Names are given for the fragmented

 DOI : 10.35291/2454-9150.2019.0357

queries. The fragmented query attributes are stored in the fragment table with respect to the fragment name.

Query Recommendation

Recommendation engine combine the active user query and query log and generate the set of queries, which are the recommendations. These recommendations are generated by the methods such as suggested query by dictionary mapping, suggested query by tuple based query recommendations, suggested query by fragment history. The data dictionary will store the column name and the synonyms. This will map the column name given by the user to the synonyms given in the dictionary when the column name given by the user is invalid. Using the dictionary mapping the query analyzer will generate the result. In the tuple based instantiation user's querying behavior is used to generate result. This will captures the individual witness of the tuple. The similarity between the current user session and that of the previous users should be calculated every time when the user submits a new query. The fragment based method will compare the query fragments of the active user with the fragments stored in the query log. The idea behind this approach is to recommend queries whose syntactical features match the queries of the current user.

Benefits of the System

- The application will generate a list of SQL Queries as recommendations, which are similar to user's query.
- The user can use Individual Query Log or Collaborative Query Log to generate expected recommendations.
- Use Fragment Based and Tuple Based Recommendation techniques to find relevant SQL queries.
- The recommendations for Nested SQL Queries are generated by applying Hierarchical Clustering.
- The user can just select a recommendation to execute or can edit the recommendation, as per findings.
- The user also can browse tables' Definition Schema and can see the History Log too.

V. CONCLUSION

Finally as per the survey studies, it indicate the importance of the emerging problem of generating query recommendations. Any authorized user of the system can apply to the query recommendation system on available database. The QueRIE continuously monitor task performed by active user and finds a matching pattern with previous user from query log and identifies similar information needs.

REFERENCES

[1] Magdalini Eirinaki, Suju Abraham, Neoklis Polyzotis, and Naushin Shaikh, "QueRIE: Collaborative Database Exploration", IEEE Transactions on Knowledge and Data Engineering, VOL. 26, NO. 7, JULY 2014.

[2] J. Akbarnejad et al., "SQL QueRIE recommendations," PVLDB,vol. 3, no. 2, pp. 1597–1600, 2010.

[3] A. Thusoo et al., Hive - A petabyte scale data warehouse using hadoop, Proc. IEEE 26th ICDE, Long Beach, CA, USA, Mar. 2010, pp. 9961005.

[4] S. Mittal, J. S. V. Varman, G. Chatzopoulou, M. Eirinaki, and N. Polyzotis,QueRIE: A recommender system supporting interactive database exploration, Proc. IEEE ICDM, Sydney, NSW, Australia, 2010.

[5] J. Akbarnejad et al., SQL QueRIE recommendations, PVLDB, vol. 3, no. 2, pp.15971600, 2010.

[6] N. Alon, Y. Matias, and M. Szegedy, The space complexity of approximating the frequency moments, Proc. 28th STOC, New York, NY, USA, 1996.

[7] E. Cohen, Size-estimation framework with applications to transitive closure and reach ability, J. Comput. Syst. Sci., vol. 55, no.3, pp. 441453, 1997.

[8] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: Item-to-item collaborative filtering," IEEE Internet Comput., vol. 7, no. 1, pp. 76–80, Jan./Feb. 2003.

[9] N. Koudas, C. Li, A. K. H. Tung, and R. Vernica, "Relaxing join and selection queries," in Proc. 33nd Int. Conf. VLDB, Seoul, Korea,2006, pp. 199–210.

[10] V. Singh et al., "Sky server traffic report—The first five years, "Microsoft Research, Tech. Rep. MSR TR-2006190, 2006.

[11] B. Liu, Web Data Mining: Exploring Hyperlinks, Contents and UsageData, 2nd ed. Berlin, Germany: Springer, 2007.

[12] B. M. X. Jin and Y. Zhou, "Task-oriented web user modeling for recommendation," in Proc. User Modeling, Edinburgh, U.K., 2005.

[13] B. Mobasher, "Data mining for personalization," in The Adaptive Web: Methods and Strategies of Web Personalization. Berlin, Germany: Springer, 2007, pp. 90–135, LNCS 4321.

[14] K. Stefanidis, G. Koutrika, and E. Pitoura, "A survey on representation, composition and application of preferences in database systems," ACM Trans. Database Syst., vol. 36, no. 4, Article 19, 2011.

[15] G. Koutrika, "Personalized DBMS: An elephant in disguise or achameleon?" IEEE Data Eng. Bull., vol. 34, no. 2, pp. 27–34, Jun.2011.

[16] S. Borzonyi, D. Kossmann, and K. Stocker, "The skyline operator," in Proc. IEEE ICDE, Heidelberg, Germany, 2001.

[17] J. Chomicki, Preference formulas in relational queries, ACM Trans. Database Syst., vol. 28, no. 4, pp. 427466, 2003.

[18] W. Kiessling, Foundations of preferences in database systems, Proc. Int. Conf. VLDB, Hong Kong, China, 2002.

[19] W. Kiessling, M. Endres, and F. Wenzel, The preference SQL system - An overview, IEEE Data Eng. Bull., vol. 34, no. 2, pp. 1118,Jun. 2011.

[20] G. Koutrika and Y. Ioannidis, Personalized queries under a generalized preference model,Proc. 21st ICDE, Washington, DC,USA, 2005.

[21] J. Levandoski, M. Mokbel, and M. E. Khalefa, FlexPref: A framework for extensible preference evaluation in database systems,Proc. IEEE ICDE, Long Beach, CA, USA, 2010.

 DOI : 10.35291/2454-9150.2019.0357