FOREX Publication
Open Access | Rapid and quality publishing

International Journal of Computer Science Research (IJCSR)
Research Article | Volume 1, Issue 2 | Pages 47-49 | ISSN XXXX-XXXX

Query Based System for Data Retrieval

Ankita Sharma[1] and Parminder Kaur[2]
[1]*Department of Computer Science, Chandigarh University, Mohali, India, ankita.sharma931@gmail.com*
[2]*Department of Computer Science, Chandigarh University, Mohali, India, Parminder.cu@gmail.com*

***Correspondence:** *ankita.sharma931@gmail.com*

ABSTRACT- Query Based System (QBS) divide into three parts pre-processing, query processing and post processing. In the pre -processing end user will give the query input then query will break down in small queries then auto correction will be there and obtain the result in query processing. At last in post processing analyze the result combine the result and return the result to the user. A dynamic database query handler for databases of different size as well as different table structures. The database handling becomes usually a big issue for the small enterprises which are not able to afford the cost of the database designer and administrators.

General Terms: Database, queries, pre-processing, query-processing, post-processing.
Keywords: SQL, join, union.

ARTICLE INFORMATION
Author(s): Ankita Sharma and Parminder Kaur;
Received: 23/09/2022; **Accepted**: 10/12/2022; **Published**: 30/12/2022;
e-ISSN: XXXX-XXXX;
Paper Id: IJCSR010204;
Citation: 10.37391/IJCSR.010204

Publisher's Note: FOREX Publication stays neutral with regard to Jurisdictional claims in Published maps and institutional affiliations.

1. INTRODUCTION

A database is a collection of information that is organized so that it can easily be accessed, managed, and updated. Formally, a "database" refers to a set of related data and the way it is structured or organized.

Database Management System or DBMS in short, refers to the technology of storing and retrieving user's data with efficiency along with safety and security features. DBMS allows its users to create their own databases which are relevant with the nature of work they want. These databases are highly configurable and offers bunch of options.

Database is collection of data which is related by some aspect. Data is collection of facts and figures which can be processed to produce information. Name of a student, age, class and her subjects can be counted as data for recording purposes. Mostly data represents recordable facts. Data aids in producing information which is based on facts. For example, if we have data about marks obtained by all students, we can then conclude about toppers and average marks etc.A database management system stores data, in such a way which is easier to retrieve, manipulate and helps to produce information e. g my SQL, SQL, oracle etc.

SQL (Structured Query Language) is a special-purpose programming language designed for managing data held in a relational database management system (RDBMS), or for stream processing in a relational data stream management system (RDSMS). Originally based upon relational algebra and tuple relational calculus, SQL consists of a data definition language and a data manipulation language. The scope of SQL includes data insert, query, update and delete, schema creation and modification, and data access control. Although SQL is often described as, and to a great extent is, a declarative language (4GL), it also includes procedural elements. Proposed Model is based on Modal driven methodology.

2. RELATED WORK

Cheung, Alvin et al. [1] presented an optimizing database-backed applications with query synthesis. In this paper, the authors have presented QBS, a system that automatically transforms fragments of application logic into SQL queries. QBS differs from traditional compiler optimizations as it relies on synthesis technology to generate invariants and post conditions for a code fragment. The post conditions and invariants are expressed using a new theory of ordered relations that allows us to reason precisely about both the contents and order of the records produced complex code fragments that compute joins and aggregates. The theory is close in expressiveness to SQL, so the synthesized post conditions can be readily translated to SQL queries.

Le, Wangchao et al. [2] has proposed a scalable multi-query optimization for SPARQL. In light of the NP-hardness of the multi-query optimization for SPARQL, the authors have proposed heuristic algorithms that partition the input batch of queries into groups such that each group of queries can be optimized together. An essential component of the optimization incorporates an efficient algorithm to discover the common sub-structures of multiple SPARQL queries and an effective cost model to compare candidate execution plans. Since our optimization techniques do not make any assumption about the underlying SPARQL query engine, they have the advantage of being portable across different RDF stores.

Llopis, Miguel et al. [3] have worked on the idea to make a natural language interface to query databases accessible to everyone. This paper describes {AskMe*}, a new system that leverages some of these approaches and adds an innovative

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Computer Science Research (IJCSR)
Research Article | Volume 1, Issue 2 | Pages 47-49 | ISSN XXXX-XXXX

feature: query-authoring services, which lower the entry barrier for end users. Advantages of these approaches are proven with experimentation. Results confirm that, even when {AskMe*} is automatically reconfigurable against multiple domains, its accuracy is comparable to domain-specific NLIDBs.

Gottlob, Georg et al. [4] has proposed the ontological queries: Rewriting and optimization. The authors have reviewed previous results and present a new rewriting algorithm for rather general types of ontological constraints (description logics). In particular, they have shown how a conjunctive query (CQ) against an enterprise ontology can be compiled into a union of conjunctive queries (UCQ) against the underlying database. Ontological query optimization, in this context, attempts to improve this process so to produce possibly small and cost-effective output UCQ.

Abouzied et al. [5] have worked on the dataplay for SQL. The dataplay is an interactive tweaking and example-driven correction of graphical database queries. The authors have given remedy on these issues with DataPlay, a query tool with an underlying graphical query language, a unique data model and a graphical interface. DataPlay provides two interaction features that support trial-and-error query specification. First, DataPlay allows users to directly manipulate a graphical query by changing quantifiers and modifying dependencies between constraints. Users receive real-time feedback in the form of updated answers and non-answers. Second, DataPlay can auto-correct a user's query, based on user feedback about which tuples to keep or drop from the answers and non-answers.

Franklin et al. [6] have worked on answer the queries with the crowdsourcing in the CrowdDB. A major change in the proposed model is that the traditional closed-world assumption for query processing does not hold for human input. From an implementation perspective, human-oriented query operators are needed to solicit, integrate and cleanse crowd sourced data. Furthermore, performance and cost depend on a number of new factors including worker affinity, training, fatigue, motivation and location.

Zhang et al. [7] have developed a system for automatically synthesizing SQL queries from input-output examples. The authors have worked upon the development of SQL Synthesizer. The author have presented a programming by example technique (and its tool implementation, called SQL Synthesizer) to help end-users automate such query tasks. SQL Synthesizer takes from users an example input and output of how the database should be queried, and then synthesizes a SQL query that reproduces the example output from the example input.

Basin, David et al. [8] has proposed a model-driven methodology for developing secure data-management applications. The authors have presented a novel model-driven methodology for developing secure data-management applications. System developers proceed by modeling three different views of the desired application: its data model, security model, and GUI model. These models formalize respectively the application's data domain, authorization policy, and its graphical interface together with the application's behavior.

I. F. Ilyas et al. [9] presented the address supporting top-k join queries in practical relational query processors. A new rank-join algorithm that is independent of the join strategy, along with its correctness proof. In this paper, physical query operator to implement rank-join based on ripple join, the hash rank join (HRJN). This paper address supporting top-k join queries in relational query processors.

3. PROPOSED WORK

Database query-based applications are the applications build to extract the information from the back-end databases. These databases are the collection of heavy amounts of data stored in them in the tree structures of database tables. The database driven based applications are expected to generate accurate and quick results. For the accuracy, the model has to be perfectly designed by purely analyzing the applications requirements for the development of the database driven or database query applications. The poorly designed application does not generate good and accurate results. Also, poorly designed applications take more time than usual. The major problems behind above listed issues are the flaws, loopholes or gaps occurred during the analysis and development of these applications, are difficult to find and fix. To avoid such development nightmares, the database query application development process has to undergo near perfect designing, development and testing environments.

4. METHODOLOGY

Query Based System (QBS), the query based system divide into three parts pre-processing, query processing and post processing. In the pre -processing end user will give the query input then query will break down in small queries then auto correction will be there and obtain the result in query processing .At last in post processing analyze the result combine the result and return the result to the user. In this paper we worked on a dynamic database query handler for databases of different sizes as well as different table structures. The database handling becomes usually a big issue for the Small enterprises which are not able to afford the cost of the database designer and administrators.

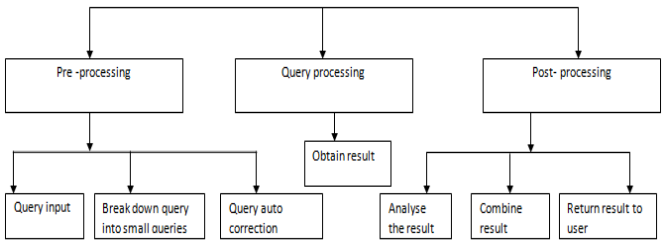

Figure 1: Query Based System

This model is a large project in size, and then the proposed model has to break down in the small models of the software, and that is the reason we use model driven approach. After breaking the database query in the smaller parts in order to obtain the results correctly and quickly. The existing work is

International Journal of Computer Science Research (IJCSR)
Research Article | Volume 1, Issue 2 | Pages 47-49 | ISSN XXXX-XXXX

based on the fixed database size and database structure, whereas are we are going to work on a dynamic database sizes and structures.

The proposed model has been designed to break the query in the multiple sub queries, which are broken into the keywords using the ontology extraction algorithm. The ontology extraction is based on the hybrid system, which is based upon the knowledge based and pattern based key extraction from the given SQL query

5. CONCLUSION AND FUTURE SCOPE

The proposed model has been designed to fetch ontology using the ontology extraction process. The proposed model has been designed to work on the basis of knowledge-based extraction mixed with position-based extraction. The proposed model has been designed to extract the operator, table names and field names from the query string. The query string extraction process is a multilevel process. At first step, the knowledge-based extraction process is initiated to extract the type of operator from the SQL query string. Afterwards, the table names and field names are extracted from the query string using the position-based attributes in accordance with the position of the operator in the SQL query string. In the future we implemented this methodology and improve previous results.

REFERENCES

[1] Cheung, Alvin, Armando Solar-Lezama, and Samuel Madden. "Optimizing database-backed applications with query synthesis." ACM SIGPLAN Notices48, no. 6 (2013): 3-14.

[2] Le, Wangchao, Anastasios Kementsietsidis, Songyun Duan, and Feifei Li. "Scalable multi-query optimization for SPARQL." In Data Engineering (ICDE), 2012 IEEE 28th International Conference on, pp. 666-677. IEEE, 2012.

[3] Llopis, Miguel, and Antonio Ferrández. "How to make a natural language interface to query databases accessible to everyone: An example." Computer Standards & Interfaces 35, no. 5 (2013): 470-481.

[4] Gottlob, Georg, Giorgio Orsi, and Andreas Pieris. "Ontological queries: Rewriting and optimization." In Data Engineering (ICDE), 2011 IEEE 27th International Conference on, pp. 2-13. IEEE, 2011.

[5] Abouzied, Azza, Joseph Hellerstein, and Avi Silberschatz. "Dataplay: interactive tweaking and example-driven correction of graphical database queries." In Proceedings of the 25th annual ACM symposium on User interface software and technology, pp. 207-218. ACM, 2012.

[6] Franklin, Michael J., Donald Kossmann, Tim Kraska, Sukriti Ramesh, and Reynold Xin. "CrowdDB: answering queries with crowdsourcing." InProceedings of the 2011 ACM SIGMOD International Conference on Management of data, pp. 61-72. ACM, 2011.

[7] Zhang, Sai, and Yuyin Sun. "Automatically synthesizing SQL queries from input-output examples." In Automated Software Engineering (ASE), 2013 IEEE/ACM 28th International Conference on, pp. 224-234. IEEE, 2013.

[8] Basin, David, Manuel Clavel, Marina Egea, M. A. Garcia de Dios, and Carolina Dania. "A model-driven methodology for developing secure data-management applications." (2014): 1-1.

[9] M. J. Carey and D. Kossmann. Reducing the Braking Distance of an SQL Query Engine. In VLDB, pages 158–169, 1998.

[10] I. F. Ilyas, W. G. Aref, and A. K. Elmagarmid. Supporting Top-k Join Queries in Relational Databases. In VLDB, pages 754–765, 2003.

[11] Z. G. Ives, D. Florescu, M. Friedman, A. Y. Levy, and D. S. Weld. An Adaptive Query Execution System for Data Integration. In SIGMOD 1999, pages 299–310, 1999.

[12] C. Jermaine, A. Pol, and S. Arumugam. Online maintenance of very large random samples. In SIGMOD, pages 299–310, 2004.

[13] D. T. Liu and M. J. Franklin. GridDB: A Data-Centric Overlay for Scientific Grids. In VLDB, pages 600–611, 2004.

[14] G. Luo, C. J. Ellmann, P. J. Haas, and J. F. Naughton. A scalable hash ripple join algorithm. In SIGMOD, pages 252– 262, 2002.

[15] G. Luo, J. F. Naughton, C. Ellmann, and M. Watzke. Toward a progress indicator for database queries. In SIGMOD, pages 791–802, 2004