

SQL QUERY TRANSLATOR: AN AUTOMATED TOOL FOR SQL TO SPREADSHEET FORMULAE CONVERSION

1.Priyanka Wani, 2. Dr. K. V.Metre

¹ BE Computer, MET's Institute of Engineering, Bhujbal Knowledge City, Nashik, Maharashtra, India

²Professor K. V. Metre ,Dept of Computer Engineering, MET's Institute of Engineering Nashik, Maharashtra, India

ABSTRACT

Spreadsheet or excel sheet is an efficient way of data analysis and management. It has diverse supplementary features such as, linear structure, visualization, statistics, reporting, periodic web query etc. There is several computation parameters required sufficient analysis. Mostly, user's not much familiar with the complicated functions of spreadsheet but many times they have basic SQL query knowledge. Spreadsheet can implement all transformation of data in SQL by utilizing SQL formulae. A query compiler is used to translate SQL query into spreadsheet with similar semantics having NULL values. User can define their queries using high-level language which is further executed on plain spreadsheet. Our main is migration of SQL data into spreadsheet format. Result graph visualization is the contribution part of system.

Keyword : - Databases, Relational Algebra, SQL Queries, Spreadsheet, Parser.

1. INTRODUCTION

Spreadsheets are the most commonly used application or tool. Spreadsheet is a computational tool for data management and analysis in business modeling applications. It is very easy to understand and to work on by any individual user or any organization. The most commonly used spreadsheet is the Microsoft Excel, used by any user. Spreadsheets offer many functions in the field of engineering, mathematics, financial, statistics so on. It also offers services like pivot, aggregation, lookups, graphs, customizable menus and so on. They are highly portable and act as a virtual machine where spreadsheet applications offered by vendors can run.

The notable thing is that spreadsheet language of formulas of Excel has become a de facto standard. It is design in a large number of spreadsheet systems, available for all major operating systems and hardware platforms, starting from handhelds and ending in the cloud, from proprietary to close source. Computer applications in the form of formula-only spreadsheets are therefore highly manageable, possibly to the extent comparable with Java bytecode. Spreadsheet systems can be observe as virtual machines, offered by various vendors , on which spreadsheet applications can be run. It is therefore extremely surprising that those machines are substantially programmed manually, with no compilers producing spreadsheet code from higher-level languages.

Still spreadsheets may have certain disadvantages. It lacks the analysis required enough to be work on. It may have scalability and share ability problems. Spreadsheet is an analytical tool having data and formula bounded together at one place giving rapid prototype.

2. RELATED WORK

In 2015, Jery Tyszkiewicz [2] stated that spreadsheets can act as a relational database engine, just by using spreadsheet formulas and no any programming language to be built in. The implementation is based on Microsoft Excel thus contributing on other spreadsheet models too. When the user enters, alters or deletes data in the spreadsheet worksheet, the formulas in query worksheet automatically computes the actual results of the queries.

In 2010, Bin Liu et al. [3] presented direct data manipulation query interface. Main objective is to create a spreadsheet like interface that directly query and access relational databases through direct manipulation. Nontechnical users find it challenging defining queries against structured databases. So a direct manipulation interface provides some datasets to the user on hand to be analysed or manipulated.

Andrew Witkowski et al. [4], [5] proposed spreadsheet like computation in RDBMS through extensions to SQL, allowing OLAP tools to handle the user interface. Spreadsheet is a successful analytical tool for business data but it has scalability problem. SQL doesn't support for n-dimensional array based computations which are frequent in OLAP scale computation while spreadsheets and specialized MOLAP engines are good for constructing formulas for mathematical modelling. In 2003, Andrew Witkowski et al. [6] Proposed automatic translation of Excel computation into SQL. Main goal was to translate Excel computation to SQL and user extensions to Excel formulas and menus to perform relational operations on RDBMS.

A fundamental challenge in spreadsheet is lack of ill-defined schema. So L. V. S. Lakshmanan [7] Proposed a framework where user specify the layout of data in a spreadsheet. Introduced the concept abstract database machine (ADM) which user layout specification viewed as a physical schema of spreadsheet which provides relational view of data in spreadsheet applications. Error rate is much high in the spreadsheets. Robin Abraham [8] introduced a type system and type inference algorithm for spreadsheets and demonstrated how these algorithms will identify the programming errors in spreadsheets.

Simon Peyton Jones et al. [9] contributed for a possible extension to Excel that supports user defined functions.

3. PROBLEM FORMULATION

Generally, in many cases users are capable understanding complex functions or operations of spreadsheet. Even also technical developer is unaware of this. But they are smart in handling of SQL queries hence to design and develop such system to provide solution to combine the database concept with another application such as, excel or spreadsheet to make it simple and easy.

4. SYSTEM ARCHITECTURE

Phase-I:

Module 1: Fire schema generation :Query At the primary level of system execution, user fires the query to generation scheme (i.e. empty spreadsheet). Mostly, user fires create table query by which system can generates the no. of attributes and records after analysis of query. For schema generation users have to fire primary DDL query i.e. create table query.

Module 2: RA formula creation : RA is the relational algebra. User query is analyzed by system, the query analysis is based on structure of query, conditions etc. Query form is then converted into RA.

Module 3: Apply RA to spreadsheet : System applies the RA formula on empty spreadsheet. It will generate automatic results when user enters some input values into spreadsheet. Rest part is cover in phase-II

Phase-II:

The empty schema structure with formula is already generated in phase-I. Now, in the second phase of system, user fires the select query on already available schema structure to get automatic results after giving some input values. Phase-II is given as below:

Module 4: Query analysis : User fires the 'select' query with specific condition such as, 'select * from table_name where condition1 && condition2'

Module 5: Apply formula : By making analysis of query, query specific formula or formulas get apply on spreadsheet to generate result.

Module 6: Generate result : At the final state, when user adds the values into spreadsheet, results are automatically generated in spreadsheet. These results are generated as per, query structure well as generated formula.

For spreadsheet generation POI system library is used to deal with Microsoft excel sheet generation with cell value customization. As per defined criteria cell data is customized with given value, font size, font color and cell

size. Now, the contribution part of the system is query result graph visualization is covered in phase-III given as below:

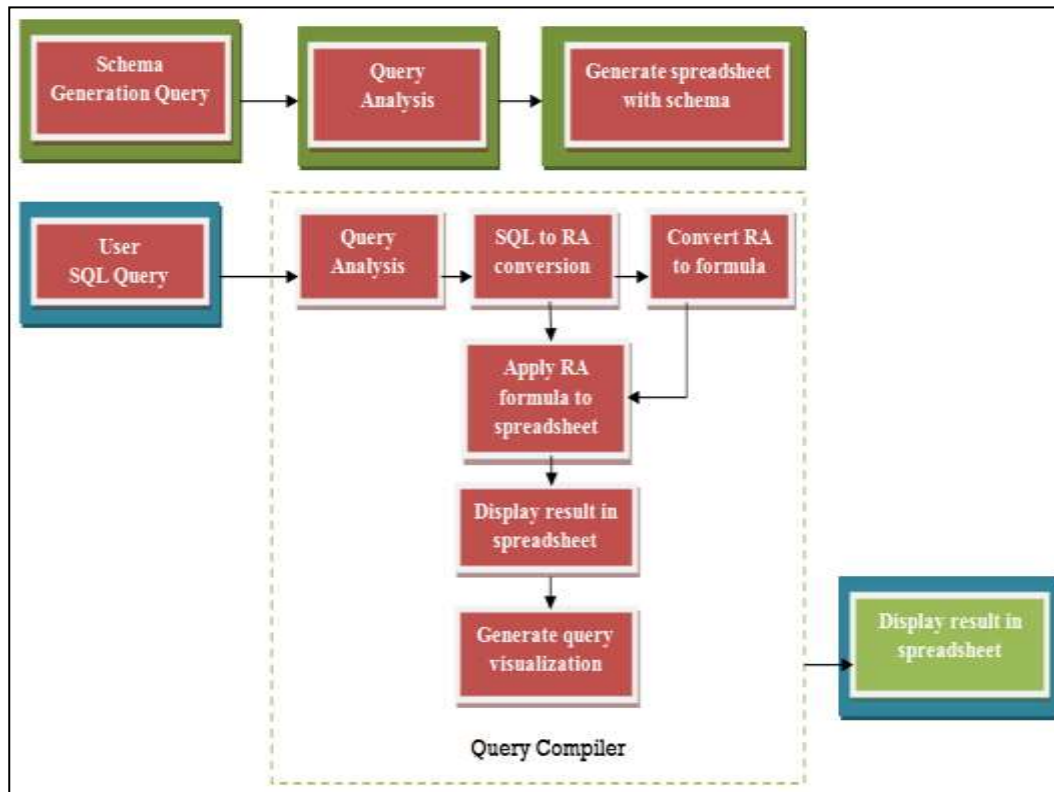


Fig-1:System Architecture

Phase-III:

Module 7: Upload spreadsheet : User uploads spreadsheet which is already available on given system as an input. This spreadsheet contains some attributes and records in it and they are mandatory because query graph is generated as per the data available in input spreadsheet.

Module 8: Analysis of spreadsheet attributes: In this module, attributes of input spreadsheet get analyzed. Attributes are nothing but different columns of spreadsheet as well column wise data is also get extracted.

Module 9: Chart column selection : In this module, user selects the label and data. Labels are the names specified for data such as, LABEL_A, LABEL_B etc and data columns are nothing but the values relevant to the given labels such as, 20%, 40% etc.

Module 10: Graph visualization: As a part of contribution, result graph is generated into spreadsheet using Jfreechart & POI libraries. It contains X-axis and Y-axis.

As everyone is aware that, MS-Excel can also generate graph as per given attribute values. But there is no facility to generate aggregate/ group by results from given table. Hence, proposed system overcomes this limitation by providing result aggregate solution to generate graph.

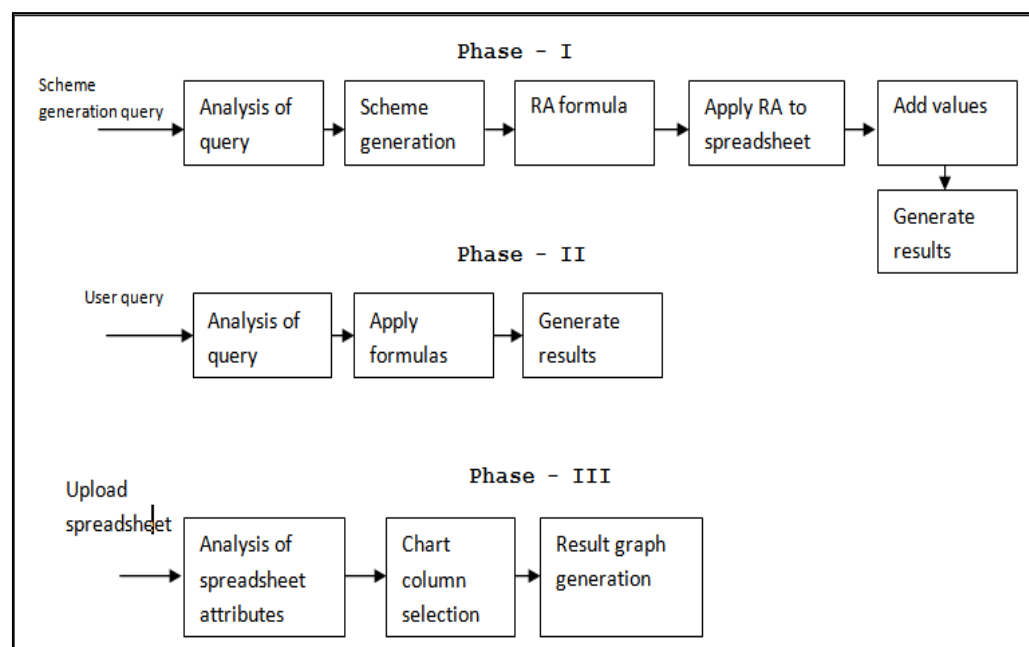


Fig-2: Phases of System

5. ALGORITHM:

Input:

- SQL Schema Query
- SQL query

Output:

- Spreadsheet
- Analysis report
- graph visualisation

Processing Steps:

A. Schema or Spreadsheet generation

1. Accept Schema query as an input to the system.
2. Make analysis of query structure
3. Generate and display spreadsheet

B. Generate visual report

1. Fire SQL query
2. Make query structure analysis
3. Analyse query type and operator
4. Translating SQL Queries into Relational Algebra i.e. RA
5. Translating RA into formula
6. Translating SQL Queries into Relational Algebra
7. Apply formula on spreadsheet for further query evaluation or calculations
8. Generate analysis report.

6. EXPERIMENTAL SETUP:

System is implemented in java-jdk 1.7. Swing components are used to design system interface. For excel connection POI libraries are used. To understand sql grammar and to convert sql to RA Antlr library is used.

7. RESULTS:

Following screenshot is the application interface using which user can write DDL table schema using table create query and DML statement as data selection query for spreadsheet.

Fig-3: Generating Spreadsheet

System generates Relational algebra expression for the given DML statement. This relational algebra is then converted in the excel formula. Following fig represents the excel implementation for the given query. A, B, C column represents year, type and total columns present in DDL statement. According to the DML statement Aggregate summation value is present in column H.

```

*** SQL query:***
select sum(tot) from Animals group by year
*** Translated to relational algebra:***
      Project(
        GroupBy(
          Animals=Relation(A,B,C),
          [1]
        ).Sum[3],
        [2]
      )

```

Fig- 4a: Relational Algebra of Query

A	B	C	H
2000	Deer	100	180
2000	Tiger	80	#N/A
2001	Deer	30	117
2001	Tiger	44	#N/A
2001	Lion	43	#N/A
2002	Deer	71	124
2002	Tiger	32	#N/A
2002	Lion	21	#N/A
2003	Deer	91	91

Fig- 4b: Entries in spreadsheet

Along with the query evaluation system generates the graph for selected column using the generated application. To draw graph in excel jFreechart library is used.

Following figure (5a) represents the graphical user interface provided to the user using which user can generate pie chart. The above excel is given as an input to the system. User can select label column and data column. User can add chart title and can provide destination file location. Figure (5b) represents its generated pie chart graph.



Fig-5: Generating Graph

8. CONCLUSIONS:

The proposed approach translates relational queries into spreadsheet. The proposed system aims to provide appropriate data management and data analysis. Spreadsheet implements all data transformations definable in SQL simply by utilizing formulas of spreadsheet. Query compiler is provided for the translation of n SQL query into a worksheet of the same semantics, including NULL values (n log2 n) sorting on spreadsheet is also offered in proposed work. As a part of contribution, query visualization graph is also generated by system.

9. REFERENCES

- [1] JacekSroka, Adrian Panasiuk, KrzysztofStencel, and Jerzy Tyszkiewicz, "Translating Relational Queries into Spreadsheets", IEEE Trans. On Knowledge and Data Engineering, Vol. 27, No. 8, August 2015
- [2] J. Tyszkiewicz, "Spreadsheet as a relational database engine," in Proc. ACM SIGMOD Int. Conf. Manag. Data, 2010, pp. 195–206.
- [3] B. Liu and H. V. Jagadish, "A spreadsheet algebra for a direct data manipulation query interface," in Proc. IEEE Int. Conf. Data Eng., 2009, pp. 417–428.
- [4] A. Witkowski, S. Bellamkonda, T. Bozkaya, G. Dorman, N. Folkert, A. Gupta, L. Shen, and S. Subramanian, "Spreadsheets in RDBMS for OLAP," in Proc. ACM SIGMOD Int. Conf. Manag. Data, 2003, pp. 52–63.
- [5] A. Witkowski, S. Bellamkonda, T. Bozkaya, N. Folkert, A. Gupta, L. Sheng, and S. Subramanian, "Business modeling using SQL spreadsheets," in Proc. 29th Int. Conf. Very Large Data Bases, 2003, pp. 1117–1120.
- [6] A. Witkowski, S. Bellamkonda, T. Bozkaya, A. Naimat, L. Sheng, S. Subramanian, and A. Waingold, "Query by excel," in Proc. 31st Int. Conf. Very Large Data Bases, 2005, pp. 1204–1215.
- [7] L. V. S. Lakshmanan, S. N. Subramanian, N. Goyal, and R. Krishnamurthy, "On query spreadsheets," in Proc. Int. Conf. Data Eng., 1998, pp. 134–141.
- [8] R. Abraham and M. Erwig, "Type inference for spreadsheets," in Proc. 8th ACM SIGPLAN Symp. Principles Prac. Declarative Program. 2006, pp. 73–84.
- [9] S. P. Jones, A. Blackwell, and M. Burnett, "A user-centred approach to functions in Excel," in Proc. 8th ACM SIGPLAN Int. Conf. Functional Program. 2003, pp. 165–176.