Volume 6, Issue 4, April 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Enhanced Translating Relational Queries into Spreadsheet

[1]Priyanka Danawale, [2]Sonal Bhorkar, [3]Nilam Bhapkar, [4]Suraj Bhoite, [5]Parchure S. V.
[1, 2, 3, 4] Final Year Students of Department of Computer Engineering SBPCOE, Indapur, Pune (SPPU), Maharashtra, India
[5] Asst. Professor of Department of Computer Engineering SBPCOE, Indapur, Pune (SPPU), Maharashtra, India

*Abstract— **Spreadsheets are among the most commonly used applications for data management and analysis. They combine data processing with very several supplementary features: statistics, visualization, reporting, linear programming solvers, Web queries periodically downloading data from external sources, etc. However, the spreadsheet ensample of computation still lacks sufficient analysis. In this paper we determine that a spreadsheet can implement all data transformations definable in SQL, simply by utilizing spreadsheet formulas. We provide a query compiler, which translates any given SQL query into a worksheet of the same definition, including NULL values. Thereby database operations become available to the users who do not want to immigrate to a database. They can define their queries using a high-level language and then get their execution plans in a plain vanilla spreadsheet. The functions available in spreadsheets introduce limitations on the algorithms one can implement. In this paper we offer O(nlog^{2}n) sorting spreadsheet, using a non-constant number of rows, and, specially, Depth-First-Search and Breadth-First-Search on graphs.***

Keywords— Relational Databases, physical database design prototypes, spreadsheets, query languages.

I. INTRODUCTION

Spreadsheets are among the most commonly used applications for data management and analysis. Spreadsheet systems offer users high level of flexibility. This aspect makes it easier for people to get started working with spreadsheets. Now a days Spreadsheets are often described as the"killer app" for personal computers .Today spreadsheets are used to handle student database, home budgets, but also to create, manage and test extremely sophisticated models and cultured data arising in business and research. Now days many users search the true databases complex enough that they simply go into either the Microsoft word,with the table-type capabilities, or into the spreadsheet, which I'd say is a little more typical, and use that as their way of structuring data. And, obviously, you get a large discontinuity because, as you want to do database-type operations, the spreadsheet is not set up for that. And so then you have to gain a lot of new commands and shift your data into another location.Despite that encouragement, relatively little research has been devoted to spreadsheets and consequently they are still poorly understood. In particular, 16 years later Excel users show up at community forums asking for help in performing database operations on their spreadsheet data. 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.

II. PROPOSED SYSTEM

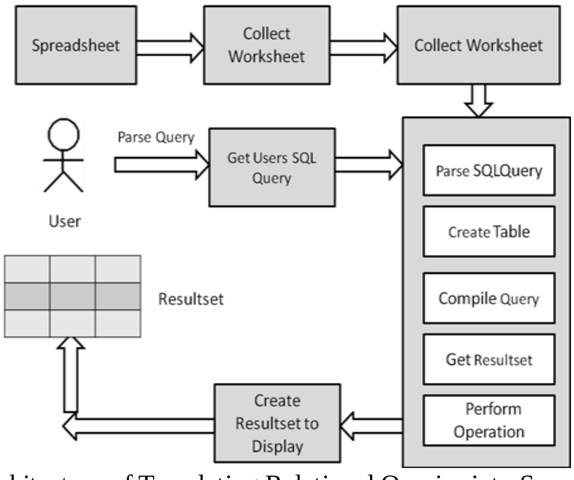


Fig: Architecture of Translating Relational Queries into Spreadsheet

© 2016, IJARCSSE All Rights Reserved

In Enhanced Translating Relational Queries into Spreadsheets, We are going to solve problem in existing system. We provide a query compiler, which translate any given sql query into worksheet of the same semantics.We assume the reader to be basically familiar with spreadsheets. The present article is written to make the solutions compatible with Microsoft Excel, version 2007 or later**.**

Above System Architecture consist of various blocks as follows:

1. Spreadsheets
2. Worksheet.
3. Query compiler.
4. Resultset.

First we have to create spreadsheet which consist of different worksheet and their data.This worksheet data is stored in query compiler.Users pass their sql queries according to their need.Compiler takes user query and perform various operations.

After compiling query,it creates final Resulset to display to the user.

III. FLOW OF PROJECT

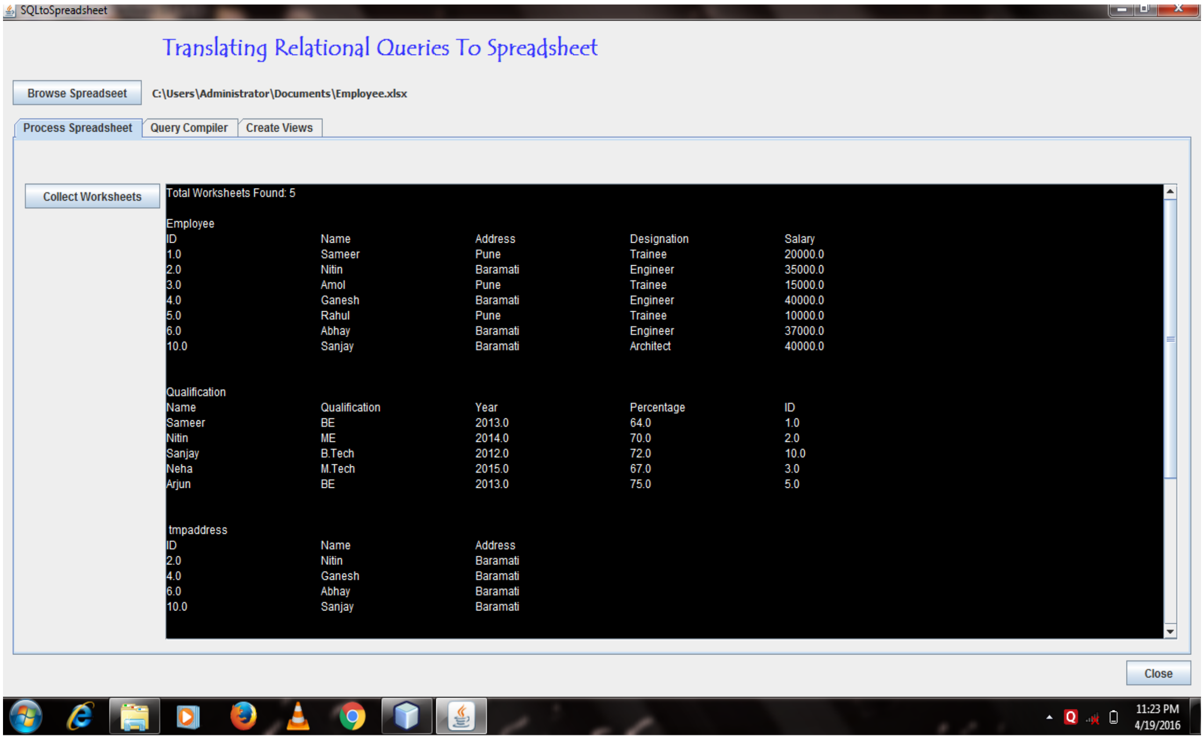
Step1: Create Spreadsheet which contains number of worksheet.
Step2: Collect Worksheet and their data.
Step3: Create Query Compiler which performs different operations on Query given by user.
Step4: User pass their SQL Query to the compiler.
Step5: Compiler take user query and perform following various operations:

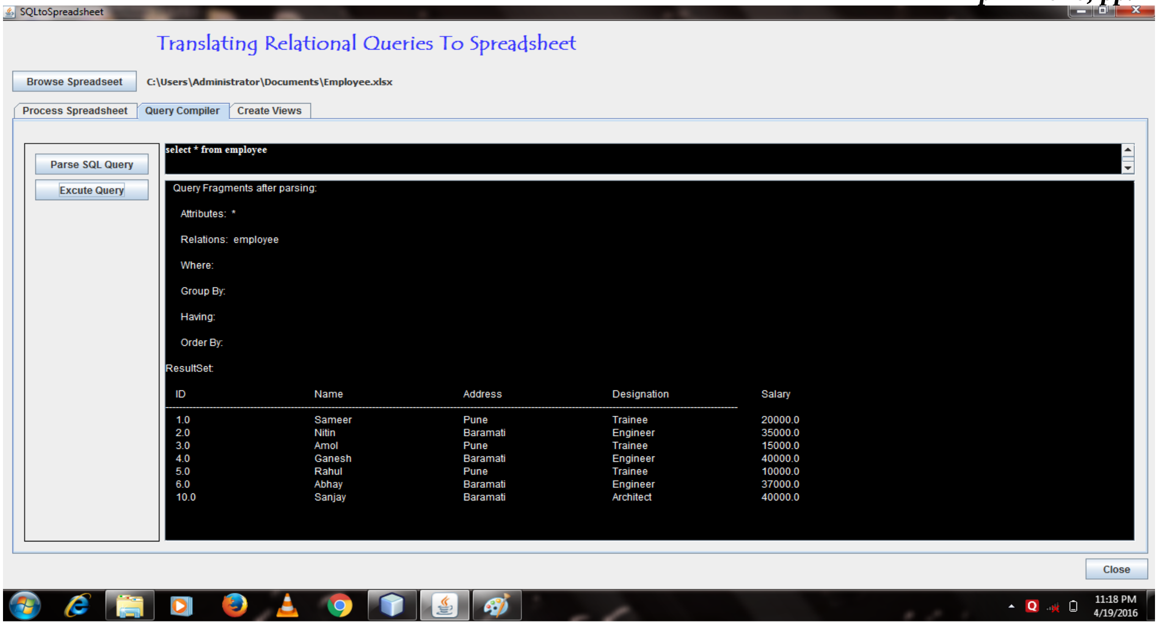
1. Parse SQL query.
2. Create tables.
3. Compile query.
4. Get Resultset.
5. Perform operations:Aggregation,Composition,join etc.

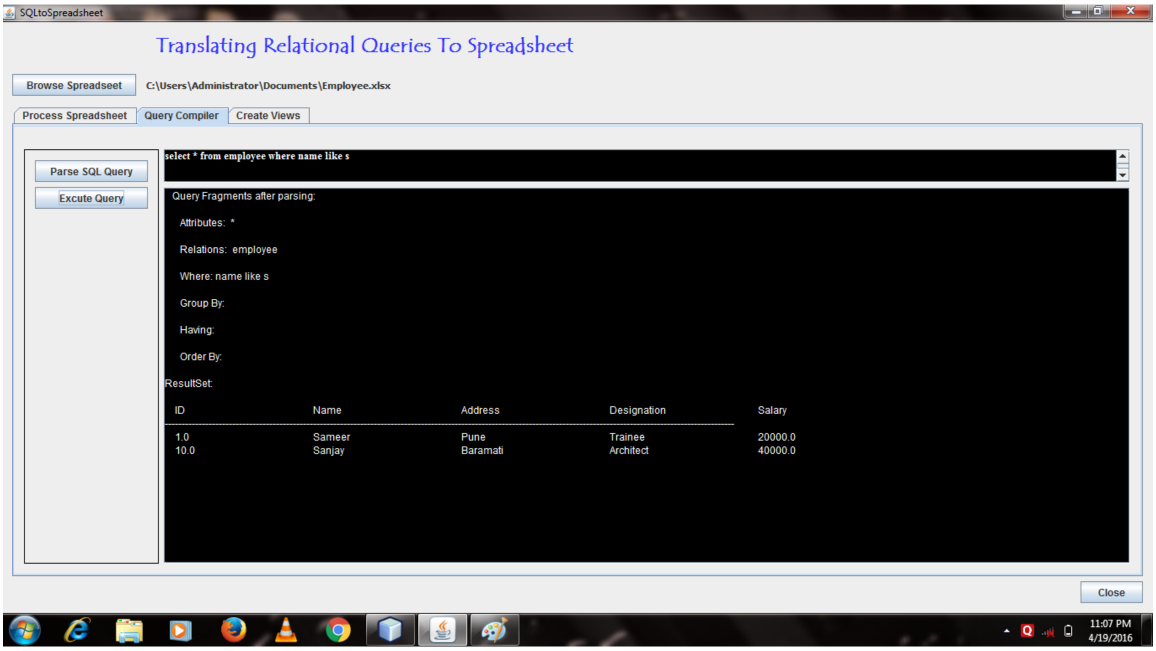
Step6: After compiling query create final Resultset to display to user.

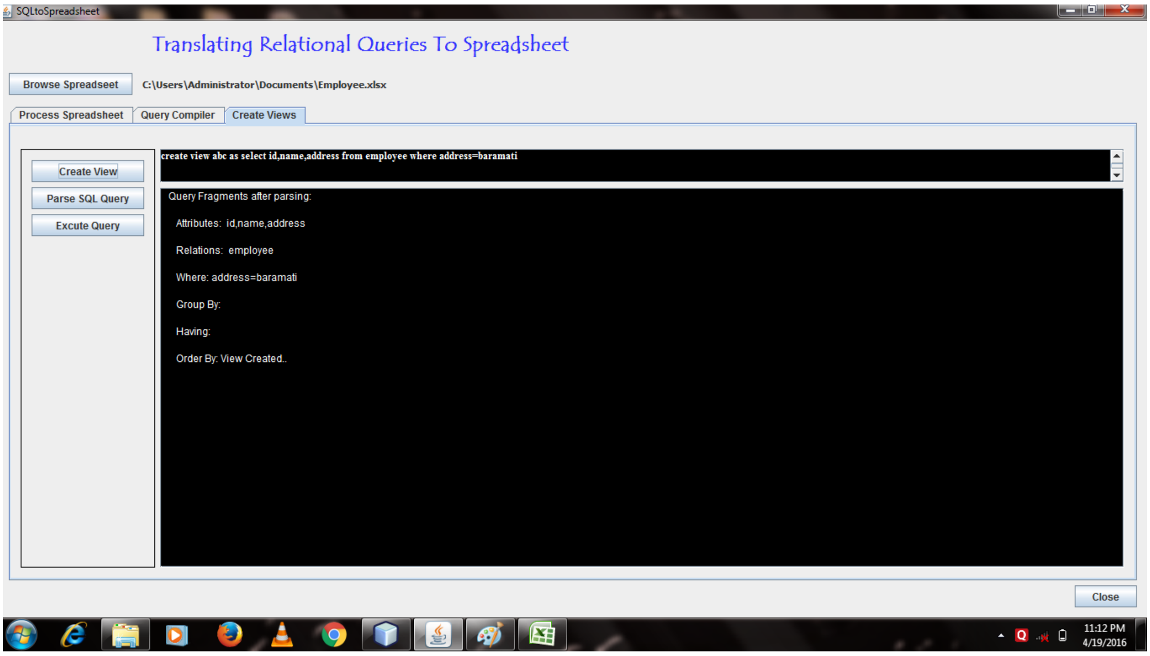
IV. APPLICATION SCENARIO

Almost every present day bank allows its customers to manage their accounts online. One of the standard features is the download of a list of all transactions on the account in a given period as an Excel file.Let us assume a bank account owner who wants to do a serious analysis of her financial activities in the past year. Probably he/she does not make more than 10 bank operations per day, which makes the data to consist of at most 4000 rows. Data of this size can be reasonably processed in a spreadsheet.Details of the data organization will differ from country to country and from bank to bank, but certainly the following fields will be provided in the file: Date of the transaction, Trans_type, Amount,Card_number (NULL in case of non-card operations),and Address where the transaction took place.

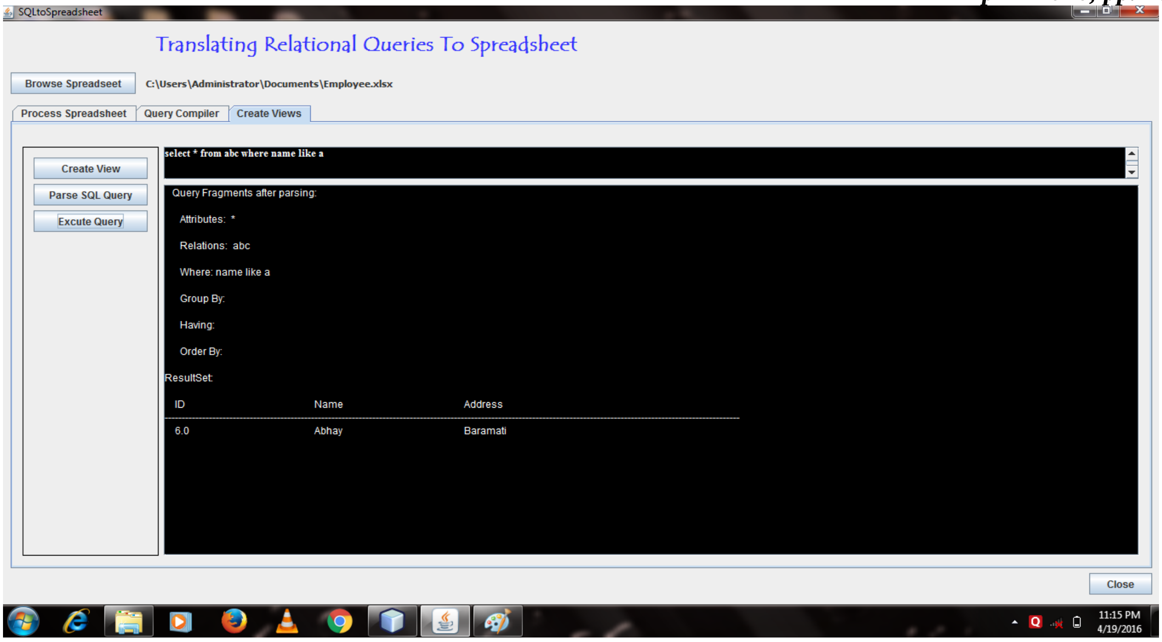


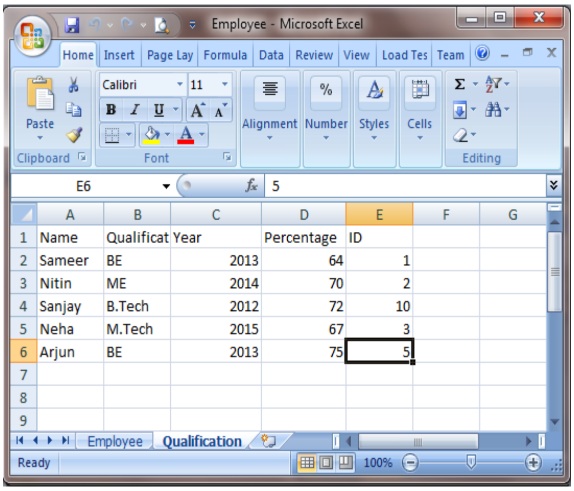


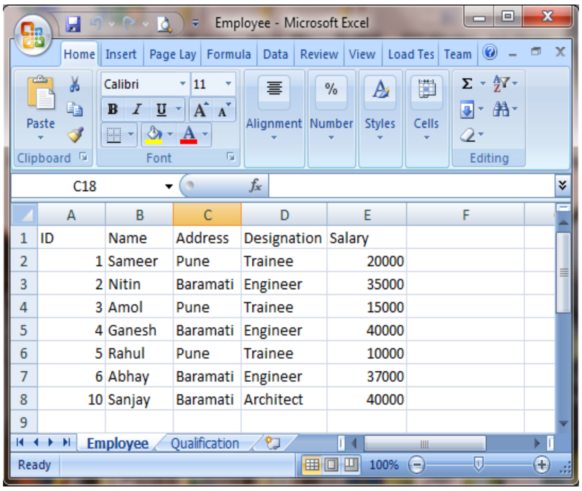




© 2016, IJARCSSE All Rights Reserved







Now the user has numerous options, how to process the data:

1) The query
SELECT Address, count(Amount)
FROM Transactions
WHERE Trans_type=’ATM withdrawal’GROUP BY Address
returns the list of addresses of ATM’s used together with the frequency of their use.

2) SELECT Card_number, sum(Amount)
FROM Transactions
WHERE Trans_type=’Card payment’
GROUP BY Card_number
returns the sums paid using each of the cards operating on the account.

V. MATHEMATICAL MODEL

Input sets:

$S=\{w_0,w_1,\dots\dots w_{n-1};0<i<n\}$
where,
n=no. of worksheets.

$A=\{a_0,a_1,\dots\dots a_{n-1};0<i<n\}$
Where,
n=no. of attributes in query string

$R=\{r_0,r_1,\dots\dots r_{n-1};0<i<n\}$
where,
n=no. of relations in query string

$P=\{p_0,p_1,\dots\dots p_{n-1};0<i<n\}$

© 2016, IJARCSSE All Rights Reserved

where,
n=no. of predicate in query string
$C=\{c_0,c_1,\ldots\ldots.c_{n-1};0<i<n\}$
Where ,
n=no.of conditions in query string
$Cs=\{cs_0,cs_1\ldots..cs_{n-1};0<i<n\}$
Where,
n=no.of constants in query string

Processing Sets:
$W=\{c_{ij} \mid \exists\, c_{ij},\, i\epsilon A, j\epsilon R';0<i<n;0<j<m\}$
Where wϵs
$R'=\{r'_0,r'_1\ldots\ldots\ldots r'_{n-1};0<i<n\}$
Where
n=no.of rows in a worksheet
$C'=\{c'_0,c'_1\ldots\ldots..c'_{n-1};0<i<n\}$
Where
n=no.of columns in worksheet
$\exists c'\epsilon w$
$w' \subseteq \{w_0 \cup w_1 \cup w_2\ldots\ldots. \cup w_{n-1}\}$
$F=\{f'_0,f_1,\ldots\ldots\ldots\ldots f_{n-1};0<i<n\}$
Where
n=no.of formulae in query string
$\ni f_i \mid f_i \subseteq \{W \cup C \cup Cs \cup 0\}$
$O=\{sum.avg,count,min,max,+,-,*,\%,<,>,!,=,\leq,\geq,\neq,==\}$
Where
O is an enumerable set of operator
$Cm=\{cm_0,cm_1,\ldots\ldots.cm_{n-1}:0<i<n\}$
Where
$\forall\, Cm_i \mid cm_i \subseteq \{A \cup R \cup W\}$
$A \cong C'$
$R \cong R'$
$Cm_i \subseteq F$

Output sets:
$\bar{R} =\{v_{ij} \mid 0<i<n;0<j<m\}$
Where,
n=no. of attribute
m=no. of rows
$\forall\, v_{ij}$ { True, $Cm_i \neq \phi$
false, otherwise
$\bar{R}\, \epsilon\{R' \propto C'\}$
Let Y be the system,we mathematically represent Y using set theory as,
$Y=\{S,A,R,P,C,Cs;W,R',C',O,F,Cm;\ \bar{R}\}$

VI. CONCLUSION

We have established that SQL can be automatically converted into spreadsheet code. Thus, we have shown the power of the spreadsheet paradigm, which subsumes the paradigm of relational databases.Apart from SQL, we have also applied a few specific algorithms: a linearithmic sorting procedure and two graph traversing algorithms: BFS and DFS. As the next steps we plan to develop optimizations for SQL queries translated into spreadsheets. We are also interested whether spreadsheets can naturally implement other models of databases, like semi-structured or object-relational ones.. This requires translating Google specific functions QUERY, SORT, FILTER and UNIQUE which are not recognized by other spreadsheet systems.

ACKNOWLEDGEMENT

We are thankful to our principal Dr. P. D. Nemade, project guide prof. Parchure S.V, and project coordinator prof. Gavali A. B., for their guidance and support in the successful completion of this study. We are also thankful to all our friends for their positive support.

REFERENCES

[1] J. Tyszkiewicz, Spreadsheet as a relational database engine, in Proceedings of the 2010 ACM SIGMOD International Conference on Management of data, ser.SIGMOD 10. New York, NY, USA: ACM, 2010,pp.195206.

[2] B. Liu and H. V. Jagadish, A spreadsheet algebra for a direct data manipulation query interface, in ICDE 09: Proceedings of the 2009 IEEE International Conference on Data Engineering. Washington, DC, USA: IEEE Computer Society, 2009,pp. 417 428.

[3] A. Witkowski, S. Bellamkonda, T. Bozkaya, G. Dorman, N. Folkert, A. Gupta,L. Shen, and S. Subramanian, Spreadsheets in RDBMS for OLAP, in SIGMOD 03: Proceedings of the 2003 ACM SIGMOD international conference on Management of data. New York, NY, USA: ACM, 2003, pp. 5263.

[4] A. Witkowski, S. Bellamkonda, T. Bozkaya, N. Folkert, A. Gupta, L. Sheng, and S. Subramanian, Business modeling using SQL spreadsheets, in VLDB 2003: Proceedings of the 29th international conference on Very large data bases. VLDB Endowment, 2003, pp. 11171120.

[5] H. Garcia-Molina, J. D. Ullman, and J. Widom, Database System Implementation. Prentice-Hall, 2000.

[6] J. V. den Bussche and S. Vansummeren, Translating SQL into the relational algebra, Lecture material, Universiteit Limburg, lecture INFO-H-417: Database Systems Architecture.

[7] P. W. P. J. Grefen and R. A. de By, A multi-set extended relational algebra – a formal approach to a practical issue, in ICDE. IEEE Computer Society, 1994.

[8] L. V. S. Lakshmanan, S. N. Subramanian, N. Goyal, and R. Krishnamurthy, “On query spreadsheets,” in *ICDE*. IEEE Computer Society, 1998, pp. 134–141.

[9] A. G. Yoder and D. L. Cohn, “Architectural issues in spreadsheet languages,” in *Proceedings of the international conference on Programming languages and system architectures*. New York,NY, USA: Springer-Verlag New York, Inc., 1994, pp. 245–258.

[10] R. Abraham and M. Erwig, “Type inference for spreadsheets,”in *PPDP ’06: Proceedings of the 8th ACM SIGPLAN Symposium on Principles and Practice of Declarative Programming*. New York, NY, USA: ACM, 2006, pp. 73–84.

© 2016, IJARCSSE All Rights Reserved