A Comprehensive Study of View Maintenance Approaches in Data Warehousing Evolution

Hemant Jain
Guru Gobind Singh Indraprastha University
Delhi, India
919873517612
hemantjain.ipu@gmail.com

AnjanaGosain
Guru Gobind Singh IndraprasthaUniversity
Delhi, India
919811055716
anajna_gosain@yahoo.com

ABSTRACT
A data warehouse mainly stores integrated information over data from many different remote data sources for query and analysis. The integrated information at the data warehouse is stored in the form of materialized views. Using these materialized views, user queries may be answered quickly and efficiently as the information may be directly available. These materialized views must be maintained in answer to actual relation updates in the different remote sources. One of the issues related to materialized views is that whether they should be recomputed or they should be adapted incrementally after every change in the base relations. View maintenance is the process of updating a materialized view in response to changes to the underlying data is called view maintenance. There are several algorithms developed by different authors to ease the problem of view maintenance for data warehouse systems. In this paper, we have provided a comprehensive study on research works of different authors related to DW view maintenance considering various parameters and presented the same in tabular way.

Categories and Subject Descriptors
H.2.7 [Database Management]: Database Administration- data warehouse and maintenance, updating, Source, query processing.

General Terms
Management, Theory, Documentation.

Keywords
Data warehouse, view maintenance, materialized views, data warehouse evolution.

1. INTRODUCTION
The A Data warehouse is a data repository used to provide reporting and analysis of large amounts of historical data. In a data warehouse, the query expressions that define views and real relations may be stored at different database sources residing at different sites. Once data sources modify, the data warehouse, mainly the materialized views in the data warehouse, should be updated also. This is the problem of view maintenance in data warehouses [15]. The sources may inform the data warehouse when an update occurs but they might not be able to determine what data is needed for updating the views at the data warehouse [27]. To avoid accessing the original data sources and increase the efficiency of the queries posed to a DW, some intermediate results in the query processing are stored in the DW. These intermediate results stored in a DW are called materialized views [26].

Materialized views are the derived relations, which are stored as relations in the database. Materialization of views is one of the classical problems in data warehousing. Materialized views can be used for reducing query response time. Because of the query intensive nature of data warehousing, materialized views approach is quite promising in efficiently processing the queries [21]. To keep, a materialized view up-to date there is a need to propagate the changes from remote data source to the destined materialized view in the warehouse. The aim of this paper is to present a survey of efforts done by different researchers in context of data warehouse materialized view maintenance and associated issues along with the devised techniques to deal with them.

The layout of the paper is as follows. In section 2, we discuss the existing approaches for data warehouse view maintenance. Section 3, presents a comparative study of the related works in a tabular manner based on certain parameters. Lastly, we conclude in section 4.

2. STATE OF THE ART
In literature, different approaches have been proposed to deal with Data Warehouse view maintenance namely basic view maintenance [8, 11, 12, 14, 20, 21, 22], incremental view maintenance [3, 6, 7, 9, 13, 15, 16, 17, 18, 23, 24, 25] and self maintainable maintenance [1, 2. 4, 5, 10, 19].

2.1 Basic View Maintenance
In Basic view maintenance approach, source and the data warehouse communicate with each other, when update occurs at source; it sends the notification to warehouse after that warehouse sends the query to source for the corresponding update as source receives the query it sends the reply to warehouse to that corresponding query [24]. This basic algorithm is neither convergent nor weakly consistent in a warehousing environment [28].

In [8] authors have motivated materialized views, their applications, and the problems and techniques for their maintenance. They have also considered new and promising application domains that are likely to drive work in materialized views and view maintenance.

In [11] authors have presented a comparison of three materialized join view maintenance methods in a parallel RDBMS, which they refer to performance at the cost of using more space. The results of this study show that the method of choice depends on the environment, in particular, the update activity on base relations and the amount of available storage space as the naive, auxiliary relation, and global index methods.

In [12] authors extend the PVM-MED WRAP algorithm to achieve the Complete level of consistency, and also presented a scalable architecture for the proposed algorithm. In this Simulation shows that extending the PVM-MED WRAP algorithm to achieve the complete level of consistency limits the maximal parallelism feature.

In [14] authors have proposed a method to minimize the unnecessary updates for materialized views without increasing of response time. This method, which is named VMOST (View Maintenance based On State Transferring), introduced four states for materialized web views. In this when receiving query/update requests, web views transfer between the four states in accordance with their accessing and changing history. According to the experimental data, VMOST is adaptive to the fast changing web environment and has a good overall performance.

A model of P2P-based collaborative view maintenance is proposed in this paper [20], which can not only exploit the potential merits of P2P and CSCW, but also overcome the problems that are overload and even

DOI: 10.1145/2347696.2347705 http://doi.acm.org/10.1145/2347696.2347705

crash at the end of data warehouse. In this authors also proposed a collaborative maintenance.

In [21] authors have discussed capabilities of PIVOT and UNPIVOT operators; materialized view maintenance, view maintenance work with PIVOT and UNPIVOT operators and finally they have focused on the research trends in view maintenance.

In [22] authors have presented an efficient technique aimed at updating data and performing view maintenance for real-time data warehouses while still enforcing these two timing requirements for the OLAP transactions. Authors proposed approach aims at addressing the issues of applying updates and performing view maintenance while still effectively serving user OLAP queries in real-time data warehouse environment.

2.2 Incremental View Maintenance

In Incremental view maintenance approach, only changes in the materialized views of the data warehouse are computed rather than recomputing every view from scratch. ECA is an incremental view maintenance algorithm based on the centralized view maintenance algorithm. It is also the fastest algorithm that will let the data warehouse remain in a consistent state [28].

In [3] authors have described the architecture of the Whips prototype system, which collects, transforms, and integrates data for the warehouse. They have showed how the required functionality can be divided among cooperating distributed CORBA objects, providing both scalability and the flexibility needed for supporting different application needs and heterogeneous sources.

In [6] authors have proposed a new incremental approach to maintaining materialized views both in the data warehouse and in the data marts. This approach uses auxiliary relations derived from the materialized view and the base relations by normalizing the view according to the functional dependencies valid in the view. The motivation for using normalization in this approach is to increase the likelihood of sharing the auxiliary relations across several data marts, as has been shown by the use of normalization in relational database design.

In [7] authors have proposed a new compensation algorithm that is used in removing the anomalies, caused by interfering updates at the base relations, of incremental computation for updating the view. This algorithm does not assume that messages from a data source will reach the view maintenance machinery in the same order as they are generated, and they are also able to detect update notification messages that are lost in their transit to the view, which would otherwise cause the view to be updated incorrectly. Proposed algorithm also does not require that the system be quiescent before the view can be refreshed.

In [9] authors have proposed a maintenance algorithm that does not need the compensation step and applies to general view expressions of the bag algebra, without limit on the number of base relations per data source.

In [13] authors have proposed an incremental maintenance method for temporal views that allows improvements over the re-computation from scratch. They introduce formalism for temporal data warehouse specification that summarizes information needed for its incremental maintenance.

In [15] authors have presented an incremental view maintenance approach based on schema transformation pathways. This approach is not limited to one specific data model or query language, and would be useful in any data transformation or integration framework based on sequences of primitive schema transformations.

In [17] authors have tackled the problem of finding the most efficient batch incremental maintenance strategy under a refresh response time constraint; that is, at any point in time, the system, upon request, must be able to bring the view up to date within a specified amount of time. In this authors have also presented a series of analytical results leading to the development of practical algorithms.

In [18] authors have developed the change-table technique for incrementally maintaining general view expressions involving relational and aggregate operators. They show that the change-table technique outperforms the previously proposed techniques by orders of magnitude. The developed framework easily extends to efficiently maintaining view expressions containing outer join operators. They have proved that the developed change-table technique is an optimal incremental maintenance scheme for a given view expression tree under some reasonable assumptions.

Incremental maintenance technique is accepted in this paper [23]. In this idea and strategy of minimum incremental maintenance is presented. The materialized view definitions and maintenance expressions, as well as algorithms are given. The experiment shows that the maintenance cost of materialized views is decreased and data warehouse processing efficiency is improved.

In [24] authors have proposed an algorithm for incremental view maintenance with the inclusion of some existing approaches. They utilized the concept of version store for older versions of tables that have been updated at the source and they are also able to detect the update notification messages that are lost during updating the view.

In [25] authors have proposed an incremental view maintenance approach based on data source compensation called DSCM, which effectively overcomes the shortcomings of previous compensation solutions through minimizing the extent of compensation to the base relations of data sources and using the precise source states to evaluate the maintenance queries.

2.3 Self maintainable Maintenance

View self-maintenance is the process of incrementally refreshing a materialized view using the view instance and the update to some base relation, but without examining any of the base relations, or using only a specified subset of the base relations. When a view together with a set of auxiliary views can be maintained at the warehouse without accessing base data, we say the views are self-maintainable [2] [5].

In [1] author has reported on some interesting new results for conjunctive-query views under insertion updates: 1) the CTSM's are extremely simple queries that look for certain tuples in the view to be maintained; 2) these CTSM's can be generated at view definition time using a very simple algorithm based on the concept of Minimal Z-Partition; 3) view self-maintenance can also be expressed as simple update queries over the view itself.

In [2] authors have showed that by using key and referential integrity constraints, they often can maintain a select-project-join view without going to the data sources or replicating the base relations in their entirety in the warehouse. They derive a set of auxiliary views such that the warehouse view and the auxiliary views together are self-maintainable-they can be maintained without going to the data sources or replicating all base data.

In [4] authors have proposed an incremental technique for efficiently maintaining materialized views in these high performance applications by materializing additional relations which are derived from the intermediate results of the view computation. They presented an algorithm which uses the operator tree for the view to determine which additional relations need to be materialized in order to maintain the view. They also give an incremental algorithm for maintaining both the view and the additional relations which has several desirable features.

In [5] author has focused on the problem of determining view in the presence of functional dependencies. Here author, shows (i) SM of a conjunctive-query (CQ) view can be reduced to a problem of query containment, whose solution can be expressed as a (Boolean) query on

DOI: 10.1145/2347696.2347705 http://doi.acm.org/10.1145/2347696.2347705

the view in safe, non recursive Data log. (ii) Special class of CQ views with no repeated predicates; two useful concepts can be defined: the well-founded derivation DAG and sub goal partitioning. (iii) Derive three simple conditions that each guarantee view SM under general functional dependencies.

In [10] authors gave a preliminary result on self-maintainability of deletions of views over XML data. We give a necessary and sufficient condition of self-maintainability of deletions, and an algorithm to implement self-maintenance of deletions of views for XML data.

This paper [19] provided an online view self-maintenance method based on source view's increment to keep the materialized view consistent with the data source. In this, authors have used the cooperation between the integrator at warehouse and the monitor at data source to maintenance the view, and the method does not require to query back to data source, it can accelerate view maintenance and lessen the burden of communication between data warehouse and data sources.

3. COMPARITIVE STUDY

We have analyzed the various research works on several parameters and presented their comparison below in the table 1. In the table the symbol '×' indicates that the corresponding parameter does not exist and symbol '√' indicates its existence in the related paper.

Table 1. Comparison of work by different authors in tabular manner

Authors	Technique/ Category	Issues addressed	Changes handled	Proposed Work	Type of Queries/ Operations	Advantages	Disadvantages	Tool support / Implem-entation	Meta data supported
Nam Huyn (1996) [1]	Self Maintainable Maintenance	Maintaining materialized view without accessing base relations	Separating SM test generation from SM test evaluation	Determine SM of views that are expressed as conjunctive queries	Not Mentioned	Speed up the update time testing & Maintenance works + Improvement in time & space	Increases complexity + For large view this approach is not practical	Not Addressed	×
Quass, Gupta, Mumick (1996) [2]	Self Maintainable Maintenance	Auxiliary Views + Materialized View Maintenance	Insertion of multiple relations at once + Protected updates	Present an algorithm for determining auxiliary views	Select-Project- Join view + Semi Join operators + SQL Expressions	Auxiliary views can be maintained without going to the data sources + Reduce the sizes of auxiliary view	N.A.	Not Addressed	×
Wiener, Gupta, Zhuge (1996) [3]	Incremental View Maintenance	Maintain one or more materialized view over the source data	Not Mentioned	Whips architecture for warehouse creation and maintenance	Select-Project- Join view + Aggregate operators	Allow concurrent queries + Consistently updating	Needed multiple query processor	C, C++ language	√
M.W. Vincent & M. Mohania (1997) [4]	Self Maintainable Maintenance	Materialized View + Multiple updates	Update the view using expression tree + Update are propagated in a bottom up fashion	Present an algorithm which use the operator tree for the view in order to maintain the view	Selection (σ) + Projection (Π) + Union (U) +	Allows multiple updates + View and relation are self maintainable	N.A.	Not Addressed	×

DOI: 10.1145/2347696.2347705 http://doi.acm.org/10.1145/2347696.2347705

					Join (><) + Difference (-)	+ Computes the effects of multiple updates simultaneously to the view			
Nam Huyn (1997) [5]	Self Maintainable Maintenance	Self Maintainability in presence of functional dependencies	Not Mentioned	Method based on testing query containment + Direct method based on the concept of well-founded deviation DAG	SQL queries	Direct method is more intuitive & more amenable to optimizations + Simple SM test queries	Further optimization is still a problem	Not Addressed	✖
Mohina & Vincent (1999) [6]	Incremental View Maintenance	Maintain data in warehouse + Data marts	Sharing the auxiliary relations	New incremental approach for maintaining materialized view	Relational algebraic operators + Aggregate Operators	Maintaining materialized view efficiently	Storage overhead + Multiple query optimization + Provide multiple caching	Not Addressed	✖
T.W. Ling & E.K. Sze (1999) [7]	Incremental View Maintenance	Materialized View + Base relation	Use of version numbers	Developed a new compensation algorithm	Not Mentioned	Message from the data source is out of order as they are generated	Detect update notification messages that are lost during transmission	Not Mentioned	✖
Gupta & Mumick (1999) [8]	Basic View Maintenance	Materialized views + Application + Problems	Classification of View Maintenance problem along dimensions	VM Techniques + New applications	Select Project Join View	Query Optimization + Integrity constraint checking	Information Identification is difficult	SQL Based Server	✖
G.Moro &C.Sartoi [2001] [9]	Incremental View Maintenance	Compensation approaches	Changes propagation rules for DWH and Centralized database	Proposed an maintenance algorithm that does not need the compensation step	Select-Project-Join relational Views	Avoid the problems of compensation based algorithms + Strong consistency	N.A.	Not Addressed	✖

DOI: 10.1145/2347696.2347705 http://doi.acm.org/10.1145/2347696.2347705

Hua, Gao, Chen (2003) [10]	Self Maintainable Maintenance	Maintain Materialized Views + XML Data	Used XML query language in the place of relational query Language	Algorithm to implement self maintaining deletions of views over XML Data	XML Views	Improve performance of view maintenance	Consider only simple updates of database + XML query language are much more complex	Not Addressed	×
Luo & Naughton (2003) [11]	Basic View Maintenance	Parallel DBMS + Single node updates	Eliminate expensive all node operation	Comparison of three materialized join VM	Select Statement + Update + View	Improve efficiency	More workload	DBMS Client application + Server	×
S.A. Fouad, O.H. Karam, & A.E. Sharkawy (2004) [12]	Basic View Maintenance	Concurrent updates + Parallel View Maintenance + Distributed Environment	Introduce Commit manager module + Commit Queue module	Extends the PVM-MEDWRAP algorithm to achieve the complete level of consistency	Not Mentioned	Provide complete level of consistency	Decrease the performance improvement of the system + Increase time cost	SQL Server + JAVA using ODBC	√
S.Amo & M.Alves (2004) [13]	Incremental View Maintenance	Temporal View + Temporal Relational Algebra	Build a uniform algorithm by combining two previously unrelated techniques	Adapted to treating databases with complex valued attributes	Not Mentioned	No need for storing the entire history of source database	Large overhead + Communication cost is more	Not Addressed	×
Y.Zhang S.Tang &D.Yang (2004) [14]	Basic View Maintenance	Materialized Views updating	Introduce mew states like Semi-Active, Semi-Sleeping	Introduce VMOST method to minimize the unnecessary updates view	Not Mentioned	During Minimization process response time is not increase	Query response time is little longer compare to eager update	Not Addressed	×
Hao Fan (2005) [15]	Incremental View Maintenance	Approach based on schema transformation Pathway	Maintaining Data Warehouse Data using AutoMed	Perform IVM along such evolvable pathways	Select Project Join Expression	Useful for any Data Transformation + Integration framework	Increases the storage overhead of the data warehouse	Not Addressed	√

DOI: 10.1145/2347696.2347705 http://doi.acm.org/10.1145/2347696.2347705

Y. Zuo, Y.Tang & Z.shu (2005) [16]	Incremental + Self Maintainable Maintenance	Concurrent Updates + Multiple dimensional view	Introduce Commit agent + Temporal Detector + TimeStamp Assign	Parallel Strategy for multiple dimensional view evaluation approach PMDVRES	Temporal Subtraction Operator(-) + Intersection Operator (U)	Detection of relevant updates + Detection of concurrent updates + Better performance in response time	Due to additional Function complexity of the architecture is increase	Not Addressed	√
Hao He, L.Xie, Yang [2005] [17]	Incremental View Maintenance	Maintaining a materialized view under a response time	Not Mentioned	Asymmetric approach for more effective VM	SQL Statements	Minimize the cost of view maintenance	N.A.	Commercial DBMS + Linux Server	✖
Gupta & Mumick (2006) [18]	Incremental View Maintenance	Aggregate & outer join operators	Update into view by refresh expression	Change-table technique + Involving aggregate & outer join operators	Projection + Group by operators	Self Maintainable + Minimize the No. of queries	Cost Model for maintenance is not suitable	Not Addressed	✖
Hia Liu, Y.Tang & Q.chen (2007) [19]	Self Maintainable Maintenance	Materialized Views + Source View	Add source view in the system architecture	Online view self-maintainable method based on source view's increment	Not Mentioned	Decrease the burden of view maintenance + Reduce complexity + Improve performance and efficiency of the integrator	Need to copy of source view in warehouse + Required more space in data ware house	Not Addressed	✖
Z.Shu, Y.Zuo & Y.Tang (2008) [20]	Basic View Maintenance	Overload unbalance between DWH + Parallel View Maintenance	Introducing P2P and CSCW idea in WHIPS prototype	P2P- based collaborative view maintenance (PCVM) + Collaborative maintenance algorithm is proposed	Not Mentioned	No. of messages in the PCVM is less compare to WHIPS + PCVM can parallelize the concurrent updates independently	Complex architecture compared to WHIPS architecture	Not Addressed	√

DOI: 10.1145/2347696.2347705 http://doi.acm.org/10.1145/2347696.2347705

Rashid & Islam (2009) [21]	Basic View Maintenance	Maintenance + Updated Materialized views	Easy to Create aggregated Cross-tabular output + Reshapes data	View Maintenance work with PIVOT and UNPIVOT operation	Select Statement + PIVOT Operation + UNPIVOT Operation	Efficient Maintenance of materialized View + Cost based optimizer + Query Transformation	Cost of generating maintenance plan is more	Oracle Database + SQL Server	✖
Hoang Vu & Gopalkrishnan (2009) [22]	Basic View Maintenance	Updates + VM + OLAP Transaction	OLAP Transaction associated with deadlines	Technique (ORAD) aimed at updating data + Performing VM	Not Mentioned	Reducing memory overhead + Reducing no. of on-demand requests	N.A.	Used Simulator	✖
Zhou, Shi & Geng [2010] [23]	Incremental View Maintenance	Materialized Views	Changes in basic relations	Minimum Incremental Technique	Selection-Projection-Join View	Reduce maintenance cost + Improve DWH performance	Not emphasize on computation complexity	Used Simulator	✖
Almazyad & Siddiqui (2010) [24]	Incremental View Maintenance	Incremental computation + Existing Approaches	Version store + TXN No. + Statement No.	Algorithm with concept of Version store	Not Mentioned	No need to update from scratch + Free from locks	Message transfer in worst case is not desirable	Not Addressed	√
Zhang, Yang & Wang (2010) [25]	Incremental View Maintenance	Concurrent Updates + Round trip + Overhead	Update of source base relations	VM Approach based on data source compensation	Not Mentioned	Reduce the roundtrip of maintenance query + Reduce the overhead of VM	Handle only single VM + Compensation cost is more	Java + Oracle, SQL server	✖

4. CONCLUSION AND FUTURE WORK

Data warehouses are prepared by collecting information from a number of sources and integrating it into one repository modified according to user requirement. Hence, solutions for data warehouse maintenance are important that can handle such sources updates. Recently, view maintenance approaches proposed by different authors have began to deal with problems like concurrent updates of different autonomous sources, materialized views consistency, functional dependencies, multiple materialized views etc. In this paper we have presented an analysis of different approaches being proposed by various authors to deal with the view maintenance in data warehouse. We have examined these approaches on various parameters and provided a comparative study in a tabular manner. In the future work, we propose a framework of data warehouse materialized view maintenance to overcome the above problems observed by various authors.

DOI: 10.1145/2347696.2347705 http://doi.acm.org/10.1145/2347696.2347705

5. REFERENCES

[1] N. Huyn, Efficient View Self-Maintenance. Proceedings of the ACM Workshop on Materialized Views: Techniques and Applications, Montreal, Canada, June 7, 1996.

[2] D. Quass, A. Gupta, I. S. Mumick, and J. Widom, Making Views Self-Maintainable for Data Warehousing. Proceedings of the Conference on Parallel and Distributed Information Systems, Miami Beach, FL, December 1996.

[3] J. L. Wiener, H. Gupta, W. J. Labio, Y. Zhuge, H. Garcia-Molina, and J. Widom, A system prototype for view maintenance. Proceedings of the ACM Workshop on Materialized Views , June 7, 1996, pp. 26-33.

[4] Vincent, M. Mohania, Y. Kambayashi. A Self Maintainable View Maintenance Technique for Data Warehouses. ACM SIGMOD (1997) 7-22.

[5] N. Huyn, Exploiting Dependencies to Enhance View Self Maintainability. http://wwwdb.stanford.edu/pub/papers/fdvsm.ps. Technical Note, 1997.

[6] Mukesh Mohania, Kamalakar Karlapalem and Yahiko Kambayashi, Maintenance of Data Warehouse Views Using Normalisation. Dexa'99, LNCS 1677, pp. 747-750, 1999.

[7] Tok Wang Ling, Eng Koon Sze. Materialized View Maintenance Using Version Numbers. Proceedings of the Sixth International Conference on Database Systems for Advanced Applications, 1999.

[8] Ashish Gupta, Inderpal Singh Mumick, Maintenance of Materialized Views: Problems, Techniques, and Applications. IEEE Data Eng. Bull. 18(2): 3-18(1999).

[9] Gianulca Moro, Claudio Sartori, Incremental Maintenance of Multi-Source Views. Proceedings of the 12th Australasian database conference 2001.

[10] Cheng Hua, Ji Gao, Yi Chen, Jian Su, Self-maintainability of deletions of materialized views over XML data. International conference on machine learning and cybernetics, 2003.

[11] Gang Luo, Jeffrey F. Naughton, Curt J. Ellmann, Michael Watzke, A Comparison of Three Methods for Join View Maintenance in Parallel RDBMS. ICDE 2003: 177-188.

[12] Fouad, S.A. , Karam, O.H. , El-Sharkawy, M.A. . A parallel view maintenance algorithm for complete consistency. International Conference on Electrical, Electronic and Computer Engineering, 2004.

[13] Sandra de Amo, Mirian Halfeld Ferrari Alves, Incremental Maintenance of Data Warehouses Based on Past Temporal Logic Operators. J. UCS 10(9): 1035-1064 (2004).

[14] Yan Zhang, Shiwei Tang, Dongqing Yang, Efficient View Maintenance in a Large-scale Web Warehouse. Fourth International conference on Computer and Information technology, CIT 2004.

[15] Hao Fan, Using Schema Transformation Pathways for Incremental View Maintenance. Proceedings of the 7th international conference on Data Warehousing and Knowledge Discovery (2005).

[16] Yayao Zuo , Yong Tang , Zhongmei Shu . A parallel multiple dimensional view re-evaluation strategy. Proceedings of the Ninth International Conference on Computer Supported Cooperative Work in Design, 2005.

[17] Hao He, Junyi Xie, Jun Yang, Hai Yu, Asymmetric Batch Incremental View Maintenance. 21st International Conference on Data Engineering, 2005.

[18] Himanshu Gupta, Inderpal Singh Mumick, Incremental maintenance of aggregate and outerjoin expressions. InformationSystems, Vol. 31, Nr. 6 (2006), p. 435—464.

[19] Hai Liu, Yong Tang, Qimai Chen, The Online Cooperating View Maintenance Based on Source View Increment. CSCWD 11th International conference, pp. 753-756, April 2007.

[20] Zhongmei Shu , Yayao Zuo , Yong Tang . A collaborative framework for parallel view maintenance. 12th International Conference on Computer Supported Cooperative Work in Design, 2008.

[21] A.N.M.B. Rashid and M.S. Islam, Role of Materialized View Maintenance with PIVOT and UNPIVOT Operators. IEEE International Advance Computing Conference (IACC'09), Patiala, India, pp. 951-955, March 6-7, 2009.

[22] Nguyen Hoang Vu, Vivekanand Gopalkrishnan, On Scheduling Data Loading and View Maintenance in Soft Real-time Data Warehouses. Comad, 2009.

[23] Lijuan Zhou, Qian Shi, Haijun Geng, The Minimum Incremental Maintenance of Materialized Views in Data Warehouse. 2nd International Asia Conference (CAR), March 2010.

[24] Abdulaziz S. Almazyad, Mohammad Khubeb Siddiqui, Incremental View Maintenance: An Algorithmic Approach. International Journal of Electrical & Computer Sciences IJECS-IJENS Vol: 10 No: 03, June2010.

[25] Xiaogang Zhang, Luming Yang, De Wang, Incremental View Maintenance Based on Data Source Compensation in Data Warehouses. Changsha, china, October 2010.

[26] C.Zhang, Xin Yao, An Evolutionary Approach to Materialized Views Selection in a Data Warehouse Environment. IEEE Transactions on systems, Man, and Cybernetics, Vol.31, No.3, August 2001.

[27] X.Wang, L.Gruenwald, G.Zhu, A Performance Analysis of View Maintenance Techniques for Data Warehouses. Submitted to Journal of Knowledge and Data Engineering, July 2004

[28] Y. Zhuge, H. Garcia-Molina, J. Hammer, and J. Widom.View maintenance in a warehousing environment. In *SIGMOD*,pages 316–327, San Jose, California, May 1995.

DOI: 10.1145/2347696.2347705 http://doi.acm.org/10.1145/2347696.2347705