International Journal of Engineering Science Invention
ISSN (Online): 2319 – 6734, ISSN (Print): 2319 – 6726
www.ijesi.org Volume 2 Issue 7‖ July 2013 ‖ PP.29-31

Data Warehousing Issues and Problems in the Real World Entities

[1]M.Sathish, [2]S.Vydehi
[1]*Research Scholar, Dept of Computer Science, Dr. Sns Rajalakshmi College of Arts and Science, Coimbatore-49, India*
[2]*Head of the Department, Dept of Computer Science, Dr. Sns Rajalakshmi College of Arts and Science, Coimbatore-49, India*

ABSTRACT: *Data Warehousing has emerged as an alternative to conventional warehousing practices in order to meet the high demand of applications for up-to-date information. In a nutshell, an active warehouse is refreshed on-line and thus achieves a higher consistency between the stored information and the latest data updates. The need for on-line warehouse refreshment introduces several challenges in the implementation of data warehouse transformations with respect to their execution time and their overhead to the warehouse processes.*

I. INTRODUCTION

A data warehouse (DW) provides information for analytical processing, decision making and data mining tools. A DW collects data from multiple heterogeneous operational source systems (OLTP–On-Line Transaction Processing) and stores summarized integrated business data in a central repository used by analytical applications with different user requirements and extracting the relevant data from the OLTP source systems, customizing and integrating this data into a common format, cleaning the data and conforming it into an adequate integrated format for updating the data area of the DW and, finally, loading the final formatted data into its database. Traditionally data warehouse stores the historical data. So, for e-business, stock brokering, online telecommunications, for instance relevant information needs to be delivered as fast as possible to knowledge workers or decision systems that rely on it to react in a near real-time manner, according to the new and most recent data captured by an organization's information system.. The main characteristics of Active Data Warehousing environment.

1) Active Data Warehousing means large volume of data and, therefore, scalability becomes critical and absolutely required to provide the large amounts of detailed data needed to understand business events.
2) Therefore, the technology should provide the means to referee the contending actions and balance the conflicting needs of these various types of users.
3) The availability (and thus the reliability) is perhaps the most distinguishing characteristic of the technology to support both tactical and strategic queries. As a result, the Active Data Warehouse must never go down, must never be inaccessible, or the business simply cannot operate.
4) Data Acquisition (ETL) must be performed much closer to the time a business event took place. Ideally, the acquisition mechanism will provide a continuous feed of new or changed data into the environment without blocking access to the very tables being updated. Now a day the demand for fresh data in the Data Warehouse is increased. Data Warehouse refreshments are traditionally performed in an off-line fashion. This means that when this process is going on then the data in the Data Warehouse is not available to the OLAP users and applications to analyze the data. So, to overcome this RTDW (Active DW) are coming.

Neoklis Polyzotis Spriros Skiadopoulos,Panos Vassilidais,Member,IEEE,Alkis Simitsis and Nils-Erik Frantzell **.,[**1] We propose a specialized join algorithm, termed mesh join (MESHJOIN), that compensates for the difference in the access cost of the two join inputs by (a) relying entirely on fast sequential scans of R, and (b) sharing the I/O cost of accessing R across multiple tuples of S. We detail the MESHJOIN algorithm and develop a systematic cost model that enables the tuning of MESHJOIN for two objectives: maximizing throughput under a specific memory budget, or minimizing memory consumption for a specific throughput. We present an experimental study that validates the performance of MESHJOIN on synthetic and real-life data. Our results verify the scalability of MESHJOIN to fast streams and large relations, and demonstrate its numerous advantages over existing join algorithms. In general, the shedding mechanism has to take into account the details of the complete workflow in order to minimize the effect of shedding on the quality of the approximate results. A variant of this problem has been explored in the context of streaming data, where the ETL workflow essentially consists of relational operators. The problem becomes substantially more complex in our context due to the generality of ETL operations. Here, we examine different shedding mechanisms under the assumption

that meshjoin is the most expensive operation in the workflow and the shedder is thus driven by the cost model of the meshjoin operator.

The design of shedding strategies for complex (and active) ETL workflows is an interesting topic for future work.mesh join is most expansive operation in the work flow.the design of shedding stratiges for complex and ETL work is an intresting topic for future work. Hadj Mahboubi,Jerome Darmont**.,[**2] In this paper, author propose the use of a k-means-based fragmentation approach that allows to master the number of fragments through its k parameter. We experimentally compare its efficiency to classical derived horizontal fragmentation algorithms adapted to XML data warehouses and show its superiority data warehouse models have been proposed to handle data heterogeneity and complexity in a way relational data warehouses fail to achieve. However, XML-native database systems currently suffer from limited performances, both in terms of manageable data volume and response time

Neoklis Polyzotis, Spiros Skiadopoulos, Panos Vassiliadis, Alkis Simitsis, Nils-Erik Frantzell**.,[**3] proposed Most importantly, multi-way joins between a stream and many relations is a research topic that requires the fine-tuning of the iteration of the multiple relations in main memory as the stream tuples flow through the Join operator(s). The investigation of other common operators for active warehousing (e.g., multiple on-line aggregation) is another topic for future work. We have proposed the mesh join (MESHJOIN), a novel join operator that operates under minimum assumptions for the stream and the relation. We have developed a systematic cost model and tuning methodology that accurately associates memory consumption with the incoming stream rate. Finally, we have validated our proposal through an experimental study that has demonstrated its scalability to fast streams and large relations under limited main memory. Francisco Araque**.,[**4] developed a systematic cost model and tuning methodology that accurately associates memory consumption with the incoming stream rate. Finally, we have validated our proposal through an experimental study that has demonstrated its scalability to fast streams and large relations under limited main memory To load the RTDW for future analysis or data mining. To notify to the user when a specific value (or threshold) has been reached.

Sirish Chandrasekaran, Michael J. Franklin.,[5] propose PSoup, a query processor based on the Telegraph framework. The core insight in PSoup that allows us to support such applications is that both data and queries are streaming and, more importantly, are duals of each other: multiquery processing is viewed as a join of query and data streams. In addition, PSoup also partially precomputes and materializes results to support disconnected operation and to improve data throughput and query response times. in conclusion, we have described the design and implementation of a novel query engine that treats data and query streams analogously and performs multiquery evaluation by joining them. This allows PSoup to support queries that require access to both data that arrived prior to the query specification and also data that appear later. PSoup also separates the computation of the results from their delivery by materializing the results: this allows PSoup to support disconnected operation. These two features enable data recharging and monitoring applications that intermittently connect to a server to retrieve the results of a query.We also describe techniques for sharing both computation and storage across different queries. In terms of future work, there is much to be done. Psoup is currently implemented as a main-memory system.We would like to be able to archive data streams to disk and support queries over them. Disk-based stores raise the possibility of swapping not only data but also queries between disk and main memory. Swapping queries out of main memory would effectively deschedule them and could be used as a scheduling mechanism if some queries were invoked much more frequently than others. In this paper,we have only briefly touched upon the relation of registered queries in PSoup to materialized views. We intend to further explore the space of materialized views over infinite streams, especially under resource constraints. The current implementation of PSoup allows the client only to retrieve answers corresponding to the current window. We intend to relax this restriction and allow clients to treat PSoup more generally as a query browser for temporal.

Goetz Graefe.,[6] propose Some researchers have considered resource contention among multiple query processing operators with the focus on buffer management. The goal in these efforts was to assign disk pages to buffer slots such that the benefit of each buffer slot would be maximized, i.e., the number of I/O operations avoided in the future. Sacco and Schkolnick analyzed several database algorithms and found that their cost functions exhibit steps when plotted over available buffer space, and suggested that buffer space should be allocated at the low end of a step for the least buffer use at a given cost [Sacco and Schkolnik 1982; Sacco and Schkolnik 1986]. Chou and DeWitt took this idea further by combining it with separate page replacement algorithms for each relation or scan, following observations by Stonebraker on operating system support for database systems [Stonebraker 1981], and with load control, calling the resulting algorithm DBMIN [Chou 1985; Chou and DeWitt 1985]. Faloutsos et al. generalized this goal and used the classic economic concepts of decreasing marginal gain and balanced marginal gains for maximal overall gain [Faloutsos, Ng, and Sellis 1991; Ng, Faloutsos, and Sellis 1991]. Their measure of gain was the reduction in the number of page

faults. Zeller and Gray designed a hash join algorithm that adapts to the current memory and buffer contention each time a new hash table is built [Zeller and Gray 1990; Zeller 1991]. Most recently, Brown et al. Have considered resource allocation tradeoffs among short transactions and complex queries [Brown et al. 1992].

V.Mallikarjuna Reddy, S.K.Jena, M.Nageswara Rao.,[7] proposed a GUI based ETL procedure to the continuous loading of the data in the Active DataWarehouse by enabling continuous data integration while minimizing impact in query execution on the user end of the DW. This is achieved by data structure replication and adapting query instructions in order to take advantage of the new real time data warehousing schemas. And for designing the mappings for this Active Data Warehouse in code based ETL takes long time, execution of the this method will consume more time but in this GUI based ETL design takes very less time, for the execution of these mappings also takes less amount of time compared to code based ETL. In code based ETL we need to develop the procedures and triggers, this consumes the maximum time but we don't require it in GUI based ETL. So, automatically the performance of the OLAP queries and Applications is improved. And with the help of schedulers we can run the session variables continuously. So, the fresh data is loaded and is available for OLAP users and applications. As future work we intend to develop an ETL tool which will integrate this methodology with extraction and transformation routines for the OLTP systems. There is also room for optimizing the query instructions used for our methods.

Panos Vassiliadis Alkis Simitsis .,[8] proposed an article that has been near real time ETL. Our intention has been to sketch the big picture of providing end users with data that are clean, reconciled (and therefore, useful) while being as fresh as possible, at the same time, without compromising the availability or throughput of the source systems or the data warehouse. We have discussed how such a configuration is different from the setting of an ETL process in a traditional warehouse. We have sketched the high level architecture of such an environment and emphasized the QoS based characteristics we believe it should have. Moreover, we have discussed research topics that pertain to each of the components of the near real time warehouse, specifically, the sources, the data processing area, and the warehouse, along with issues related to the flow regulators that regulate the flow of data under data quality and system overhead "contracts". Clearly, the importance, complexity and criticality of such an environment make near real time warehousing a significant topic of research and practice; therefore, we conclude with the hope that the abovementioned issues will be addressed in a principled manner in the future by both the industry and the academia.

Lukasz Golab And M. Tamer Ozsutraditional .,[9] proposed in bases of databases store sets of relatively static records with no pre-de_ned notion of time, unless timestamp attributes are explicitly added. While this model adequately represents commercial catalogues or repositories of personal information, many current and emerging applications require support for online analysis of rapidly changing data streams. Limitations of traditional DBMSs in supporting streaming applications have been recognized, prompting research to augment existing technologies and build new systems to manage streaming data. The purpose of this paper is to review recent work in data stream management systems, with an emphasis on application requirements, data models, continuous query languages, and query evaluation.

II. CONCLUSION

This paper presents a continuous loading of the data in the Active Data Warehouse by enabling continuous data integration while minimizing impact in query execution on the user end of the DW. we have discussed various proposals and problems in data warehousing procedurals and also discussed about future work which are obtained in various author views in the article. the importance, complexity and criticality of such an environment make real time warehousing a significant topic of research and practice therefore, we conclude with the hope that the above mentioned issues will be addressed in the future by both the industry and the academia.

REFERENCES

[1]. A. Karakasidis, P. Vassiliadis, and E. Pitoura, "ETL queues for active data warehousing," in Proc. of IQIS, 2005.
[2]. W. Labio and H. Garcia-Molina, "Efficient snapshot differential algorithms for data warehousing," in Proc. of VLDB, 1996.
[3]. W. Labio, J. Yang, Y. Cui, H. Garcia-Molina, and J. Widom, "Performance issues in incremental warehouse maintenance." in Proc. of VLDB, 2000.
[4]. D. Burleson. New developments in oracle data warehousing. Burleson Consulting, April 2004.
[5]. C. White. Intelligent business strategies: Real-time data warehousing heats up. DM Review, 2002
[6]. Ricardo Jorge Santos and Jorge Bernardino," Real-Time DataWarehouse Loading Methodology", IDEAs'o8, ACM 978-1-60158-188-0/08/09,2008.
[7]. A. Simitsis, P. Vassiliadis and T. Sellis, "Optimizing ETL Processes in Data Warehouses", International Conference on Data Engineering (ICDE), 2005
[8]. Jovanka Adzic and Valter Fiore. Data Warehouse Population Platform. In DMDW, 2003.
[9]. W. H. Inmon, R. H. Terdeman, J. Norris-Montanari, and D.Meers, "Data Warehousing for E-Business", J. Wiley &Sons, 2001.
[10]. X. Zhang and E. A. Rundensteiner. Integrating the maintenance and synchronization of data warehouses using a cooperative framework. Inf. Syst., 27(4), 2002