International Journal of Soft Computing 1(2) : 97-102, 2006

Recent Issues Identified and Conceptualized for Simulating Business Scenarios in Multi-Version Data Warehouse

Muhammad Khurram Shahzad
10A : 11, Kista Allevåge, Kista 16455, Department of Computer and System Sciences, Kista, Stockholm University/Royal Institute of Technology (KTH), Stockholm, Sweden

Abstract: Data Warehouse provides an opportunity to analyze businesses in order to precise decision- making, various tools has been build and used for such purpose. But due to static structure of DW it is unable to manage changes in external sources, also simulating various scenarios can increase the decision efficiency, for such purpose Multi-Version Data Warehouse is proposed which contains two types of versions called real and alternative. But due to diversity in simulations two types of alternative versions has been proposed. Also for querying these versions we are extending the functionality of Synthetic Warehouse Builder. This study presents the challenges we are facing and visualizing during the extensions.

Key words: Data warehouse, evolution, versioning, business alternatives, simulations, transparency, query processing architectures, integrate catalog

INTRODUCTION

Data Warehouse (DW) is defined as a subject-oriented, integrated, time variant collection of data in favor of decision-making[1,2]. In other words, DW provides integrated access to External Data sources (ED) which can be centralized and/or distributed over a computer network. These sources are autonomous in their architecture and do not depend upon each other, but DW due to its transformation processes (called ETL) depend upon ED's, hence effecting their autonomy. It has been found that changes to external data sources may result in derivation of inconsistent analytical from DW[1-3]. These changes are categorized into two types[4,5]. a) Content changes, due to insert update and delete commands. b) Schema changes, are due to create, delete alter commands. Various evolutionary and temporal-versioning approaches have been proposed[1,2,6,7] to solve the problems of schema and content changes. In addition to it, the conception of Multi-Version Data Warehouse (MV-DW) is proposed for simulating various business scenarios[3]. For querying MV-DW, multi-version query language is proposed by making extensions to standard query language[8].

A quality approach to handle problematic changes has been modeled in[2]. This approach with its versioning function evaluates change-intensity and based on it either evolution or versioning of schema is proposed. Versioning function is defined as[2,9]:

Definition 1: A versioning function takes multi-dimensional schema as input argument and produces new one due to operation $Oper_a$ on $\aleph_0$. This operation results in $\aleph_n$ MD-schema, mathematically:

$$\aleph_n : Oper_a(\aleph_0)$$

$\aleph_n$ is the evaluated schema or new version.

We have already eliminated the limitations of[8] by building Synthetic Warehouse Builder (SWB) in[9]. But it has been found that in-order to obtain the required flexibility and vital advantages of simulating business scenarios, extensions are required to the conception of evolution of versions and hence in SWB. Contributions: This study reports challenges we are facing and conceptualizing while making extensions to SWB, in order to achieve vital advantages of flexible what-if-analysis in simulating business scenarios. For such purpose we have divided simulating business scenarios into two types a) Static simulations also called static business alternative, which are handled by maintaining Materialized Alternative Versions (MAV). b) Flexible simulations or dynamic business alternatives, which are vibrant as they posses no instance. These are maintained by Virtual Alternative Versions (VAV). Version manager is proposed as a component of SWB for querying various types of versions.

MOTIVATION

Due to static structure of multi-dimensional schema of DW, conventional DW is unable to support business dynamics. Even small changes to contents or schema of ED's sources may result in derivation of

inconsistent analytical results which lead to deficiencies in decision-making[1,3,10]. DW is dependent on its transactional data sources which consequents autonomy problems for ED's. So evolution of DW and related research issues has been visualized and proposed to be handled in this study.

Ongoing information age has increased the worth of decision-making by appending number of analytical variables with it, like simulating business scenarios. But conventional DW along with its evolutionary approaches does not support these simulations and hence ignores alternative business scenarios, which are found to be very useful for malicious decision-making[3].

To handle business dynamics and simulating alternative business scenarios, a model of MV-DW has been proposed with two types of versions[3]. a) Real versions, to handle transactional data source changes. b) Alternative versions, for simulating alternative business scenarios.

But various businesses are found to have multiple alternatives scenarios, some of these alternatives are static and others are vibrant. If all the alternative versions retained for such scenarios are materialized then it is not possible to handle dynamic simulations, as changes to simulations causes changes in schema which is not possible in various cases due to attached instances. In other words, required flexibility cannot be achieved and space issues may also come into play due to terabytes of data. So a balanced approached is required for simulating various types of simulations.

RELATED WORKS

Data Warehouse acts as a source for various analytical applications so for better analysis it is required to keep consistent and correct dimensional data in data warehouse. But due to dynamics in sources of data warehouse inconsistent results can be derived[1,3,9,10], hence decreases analytical quality. Two categories of approaches has been proposed for such problems a) Data and Schema evolution approach[6,7,11], in which schema is upgraded and data is transferred to the latest schema and only this schema is maintained. b) Versioning approach[1,9,12-14], various such types of approaches has been proposed like temporal versioning, in which changes are applied in time stamp order and multiple versions are maintained, but also set of disadvantages has been defined[3,12,15].

Another approach for maintaining consistency and for better what-if analysis is done by using the concept of simulating business scenarios[3,7,9,16]. For the implementation of simulating business scenarios two types of versions has been used[1-3,10] a) Real Versions, are used to handle changes in sources. Whenever change comes it is applied to new schema and hence new version is born, newly born version is called child version which is derived from parent version. Two or more versions can share data to solve the storages issues. b) Alternative Versions are generated and used in simulating business scenarios, for better what if analysis i.e. it may be required that user want to create a scenarios in its business and analyze the happenings.

For retrieval from various versions standard query language has been extend to form multi-version query language[8] also a prototype has been implemented with certain limitations. But query writing becomes complex due to increased number of versions, resulting in need of transparent querying method. So we have developed Synthetic Warehouse Builder (SWB)[9] with two major components a) Multi-version integrated logical schema builder, which gives union of all the versions. b) Synthetic Query Analyzer, used to write and process queries on multiple versions.

Although, SWB provides transparent retrieval method but a) do not support changing multi-dimensional schemas i.e. alternative version are populated so facility of addition, modification or deletion of dimension or fact may not be supported. b) Do not provide flexible environment for achieving vital advantages of simulating alternative business scenarios c) Querying scope is restricted to set of real and alternative versions d) In case of evolution of version MV-ILS is not automatically updated.

Given disadvantage gives rise to the need for flexibility in alternative versions, because a) if all the versions are materialized storage issues my come into play and simulation scope is restricted b) If all the versions are not materialized then performance issues may come into play for commonly used simulations. Hence, there is a need of such an approach will can handle these problems.

PROPOSED VERSIONING OF DW

Our approach[2,17] to solve schema and content changes problem is based on explicitly versioning of DW, stating that as soon as business change appears it is required to make adjustments to DW so that such changes can be captured. Also, evolutionary operations and versioning algebra based on multidimensional paradigms are defined for such changes[2].

However, here we are purposefully extending the concept of alternative versions in order to acquire vital advantages of simulating business alternatives. These extensions are made by defining two types of alternative versions:

- Static alternatives, which are considered and used all the time to evaluate business alternatives, so in our approach they are explicitly handled by Materialized Alternative Versions, which has schema and an instance attached with it. These simulations are permanently checked as an alternative business scenario for quality decision- making.
- Dynamic alternatives, the alternatives which are very dynamic and are changed all the time, it is not at all suitable to materialize them and so these types of version are handled by Virtual Alternative Versions. These are stored in Multi-version Catalog (MV-Catalog) in the form of virtual relations and are very dynamic.

To reveal versioning status, graphical method called Schema-Versioning Graph (SVG)[1,9] can also be amended. SVG is not only used to portray evolutions versions and relationship between versions with the help of version derivation relationship but also offers a graphical interface to play with the versions. It is obvious that these versions are heterogeneous (due to evolutionary operations) and each version will have validity time attached with it, which indicates the time for which this version is valid.

ARCHITECTURE OF MV-DW

This section gives the complete architecture and flow of data between different components of MV-DW. For ease of understanding Fig.1 also indicates the existence of all types of versions. Like conventional DW architecture, it includes tools for extracting data from multiple ED's for cleaning and loading purposes but in respective versions.

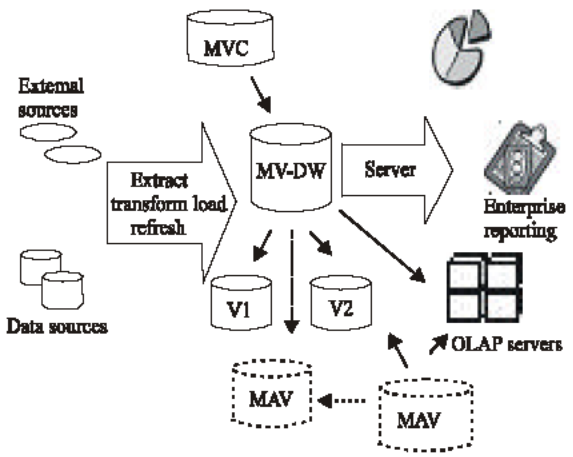

Fig. 1: MV-DW architecture with real, materialized and virtual alternative versions

Multi-version Catalog in the architecture is used to maintain evolutionary operations and versioning metadata which can further be used for ease of querying various versions. It will also contain the logical schema called Multi-Version Integrated Logical Schema (MV-ILS) for transparent querying real set of versions[9]. Further MV-DW presents views of data to a variety of front end tools e.g. OLAP, dashboards and report writers.

For flexibility in simulating business alternatives two types of alternative versions called materialized and virtual alternative versions are also shown in Fig. 1. RV/s acts as a source of data for MAV's while VAV's have no instance attached with them. Instead, they are only mapped to respective RV and MAV along with transformations rules which are stored in multi-version catalog.

QUERYING MV-DW

With the proposed evolution of versions, MV-DW has three types called RV set, MAV and VAV. Although[8] proposes multi-version querying language by making extensions to standard query language but query writing becomes impossible with increased number of versions. So for transparent querying purposes and in-order to remove the limitations of[8] we had implemented Synthetic Warehouse Builder (SWB)[9].

But limitations of SWB include: A) Auto up-gradation of integrated logical schema b) Vital advantages of simulating business scenarios can not be achieved. c) SWB do not provide flexible environment for simulating business alternatives. In short, until now no complete solution exist for our here proposed versioning approach due to various appended issues. This section presents a number of technical issues of exploratory research in order to retrieve data from various, different-typed and heterogeneous versions of MV-DW.

According to transparent retrieval approach SWB, users are provided with a logically integrated schema called Multi-Version Integrated Logical Schema (MV-ILS), which provides union of all the real versions set, hence a user facilitator for querying multiple versions. Similarly MAV and VAV are provided for simulations purposes.

For querying purpose user need not to know about the existence of data location in versions, instead he is requirMarch 7, 2006ed to keep track of MV-ILS. It is the responsibility of the SWB to store metadata about MV-ILS and its mapping to original versions, nevertheless the heterogeneity issues comes into play e.g. one version stores monthly salary in dollars while another version stores annual salary in pounds. Also, active schema integration architecture is required to be researched for

auto up-gradation of MV-ILS and inheritance of attached semantic properties, as soon new versions are evolved.

For better analysis user can retrieve data from real and alternative versions and make comparison[3], but VAV's have no instance attached with them and as an alternative it is required to define and store mapping along with transformation rules from VAVs to MAVs and RVs.

Hence, for retrieval from various versions, it is required to break multi-version query on integrated schema into mono-version queries executable on, a) Real versions b) Materialized alternative versions c) Virtual alternative versions. This process of breaking a multi-version query into mono-version queries is called confinement process. Since VAV's share data from real and set of materialized alternative version so it is required to further break VAV query into smaller parts and send it to respective mapped versions. Finally, the issues of merging results of these mono-version queries require more elaboration.

As a result of confinement process concurrent execution of multiple queries on single version is possible. These queries can be a) Mono-version query of first confinement to be executed on RVx b) Mono-version query to be executed on RVx, in order to retrieve data for VAV. So, it is also required to maintain the consistency of results by defining concurrency control algorithms. More research issues include the efficient use of temporary storage, aggregate merging techniques, query buffering and the concept of read transaction for maintaining query atomicity.

Maintaining large set of versioning metadata can only solve these querying problems, for this reason we have proposed multi-version catalog with the abilities of storing relevant metadata.

METADATA AND MV-CATALOG

Conventional catalogs[15] and integrated data dictionaries[18] cannot store metadata for querying multiple DW versions. Hence, this section outlines metadata requirements, multi-version catalog for storing metadata and its use for querying multiple DW versions.

For querying multi-version DW it is required to store complete metadata about dimension tables, fact tables, relationship between dimension and fact tables and necessary semantic properties of MV-ILS along with mapping and transformation rules from individual versions to integrated schema. Each version has its time stamp which indicates the validity time of the versions denoted by VTm. If the version is disabled it is required to make changes to MV-ILS and hence in its metadata.

Since, three types of versions are used so metadata should be defined in such a way that type of each version is reflected. Also, it should be indicated that version x is for static simulations and y is for dynamic simulations so changes can only be made in VAVs. Since various versions are also sharing data between each other so it is obvious that metadata about schema and attached instances will be maintained along with derivation relationship of versions. Especially with VAVs with zero instances, metadata about mapping from VAVs and its mapping to set of real and materialized alternative versions is required to be maintained. All this metadata data will be used during the confinement of multi-version query to mono-version queries and during the mapping of VAV query onto set of real and materialized alternative versions.

A data dictionary or catalog is a repository of information describing the data in database; it is called data about data or metadata[15,18]. Typically conventional catalogs are not found to be suitable to store mentioned metadata, so MV-Catalog is proposed. This catalog due to its architecture and various functions attached with it should grow and contract with the evolution of DW and it should facility the query processing algorithms by providing them required metadata.

VERSION-MANAGER FOR SWB

To achieve the objective of getting vital advantages of simulating business scenarios in the presence of schema and content change problems, we are working to build complete solution with the name 'Synthetic Warehouse Builder'. SWB has two major components[9] a) MV-ILS builder used to represent the union of versions b) Synthetic Warehouse Analyzer, for transparent querying and simulating various business scenarios.

For implementing proposed theories a new component with the name Version Manager (VM) is added to SWB, whose responsibility is to provide flexible simulating environment for decision-makers. While, using VM user can create, modify and delete new virtual dimensions, facts and virtual alternative versions, the only querying requirement is one-time definition of mapping rules from VAV to relevant versions.

VM take multi-version query from user interface and interact with metadata manager of MV-Catalog for confinement metadata. By using this metadata multi-version query is divided into mono-version queries executable on individual versions. Since, VAV's have no instance attached so their query is further divided into smaller queries for real and materialized alternative

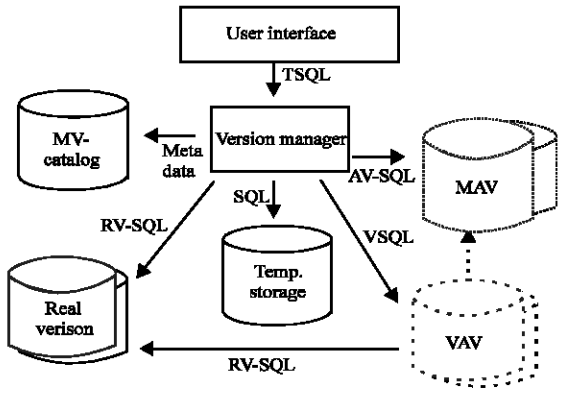

Fig. 2. Architecture of version-manager

versions. After execution of mono-version queries result set saved in temporary stored for the purpose of derivation of final results for end users. For optimized utilization of temporary storage *Inlining technique*[19] can be used with few modifications (Fig. 2).

OTHER RESEARCH CHALLANGES

It is expected that SWB after its completion will provide a flexible environment for what-if-analysis and simulating alternative business scenarios in the presence of transparency. But still some more issues require attention of the researchers, which are:

- Multi-version indexing technique to retrieve data efficiently, this can be done by making extensions to conventional techniques.
- Development and Implementation of consistency control algorithms for consistent results.
- Populating OLAP with MV-DW data and its simulations.
- Maintaining query atomicity during its execution on multiple versions.
- Incorporating various types of external sources with dimensional data of MV-DW.

CONCLUSION

This study presents research issues and challenges we are facing and visualizing during the development of a flexible environment of what-if-analysis and simulating business scenarios in order to improve the quality of decision-making.

Finally, we have proposed extensions to Synthetic Warehouse Builder by adding a new component Versions Manager to handle such simulations. Maintaining consistency of results, query atomicity, architecture of version-manager, multi-version catalog architecture and metadata management are few directions of future research.

ACKNOWLEDGEMENTS

Although this article will only be added to my list of publications and will be memorized as my individual work but the fact stands otherwise and I don't have suitable words to write thanks to my parents, my wife and my brothers and sisters who supported and assisted me in visualizing this work.

REFERENCES

1. Shahzad, M.K., J.A. Nasir and M.A. Pasha, 2005. SVF: Schema-Versioning Framework for Data Warehouse, Abstract in Proceedings of National Conference on Information Technol. Applications, Quetta, Pakistan.
2. Shahzad, M.K., J.A. Nasir and M.A. Pasha, 2005. CEV-DW: Creation and Evolution of Versions in Data Warehouse Asian J. Infor. Technol. Medwell online, 4: 910-917.
3. Bebel, B., J. Eder, C. Koncilia, T. Morzy and R. Wrembel, 2004. Creation and Management of Versions in Multiversion Data Warehouse Proceedings of 2004 ACM Symposium on Applied Computing.
4. Rundensteiner, E., A. Koeller and X. Xhang, 2000. Maintaining Data Warehouse over changing information sources, Communications of the ACM, pp: 43.
5. Soberg, D., 2004. Quantifying Schema, Evolution Information Software Technol., 35: 35-54.
6. Hurtado, C.A., A.O. Mendelzon and A.A. Vaisman, 1999. Maintaining Data Cubes under Dimension Updates. Proceedings of ICDE Conference, Australia.
7. Blaschka, M., C. Sapia and G. Hofling, 1999. On Schema Evolution in Multidimensional Databases. Proceedings of DaWak99 Conference, Italy.
8. Morzy, T. and R. Wrembel, 2004. On Querying Versions of Multi-version Data Warehouse. In Proceedings of 7th ACM DOLAP'04, Washington, USA.
9. Nasir, J.A., M.K. Shahzad and M.A. Pasha, 2005. Transparent querying multi-version data warehouse, to appear in Information Technol. J. Ansinet.
10. Pasha, M.A., J.A. Nasir and M.K. Shahzad, 2004. Semi-star schema for managing data warehouse consistency, In Proceedings of National Conference on Emerging Technologies, IEEE-ACM, NCET'04, Karachi, Pakistan.
11. Hurtado, C.A., A.O. Mendelzon and A.A. Vaisman, 1999. Updating OLAP Dimensions, Proceeding of DOLAP Workshop.

12. Chamoni, P. and S. Stock, 1999. Temporal structures in Data Warehousing. Proceedings of the DaWaK, Italy.
13. Eder, J. and C. Koncilia, 2001. Changes of Dimension Data in Temporal Data Warehouses. Proceedings of DaWak.
14. Eder, J., C. Koncilia and T. Morzy, 2002. The COMET Metamodel for temporal Data Warehouses, Proceedings of CAISE'02, Canada.
15. Ozsu, M.T. and P. Valduriez, 2002. Principles of Distributed Database SystePedersen, D. Riis, K. Pedersen, T.B. Query optimization for OLAP XML federation, In Proceedings of 5th ACM Workshop on DOLAP, USA.
16. Wrembel, R. and B. Bebel, 2005. Metadata Management in a Multiversion Data Warehouse, In Proceedings of Ontologies, Databases and Applications of Semantics (ODBASE), Cyprus.
17. Wrembel, R. and T. Morzy, 2005. Multiversion Data Warehouses: Challenges and Solutions. Proc. of the 3rd IEEE Conference on Computational Cybernetics (ICCC 2005), Mauritius.
18. Frank, W.A., E.S.L. Mary and V.M. Mannino. The Integrated Dictionary/Directory System.
19. Pedersen, D., K. Riis and T.B. Pedersen, 2002. Query optimization for OLAP XML federation, In Proceedings of 5th ACM Workshop on DOLAP, USA.