ISSN:0975-9646

P.N.V.S. Pavan Kumar et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (3) , 2014, 3227 - 3231

A Survey on Bitemporal Data Warehousing System and Data Warehouse Design Techniques.

P.N.V.S.Pavan Kumar, K.Bala Chowdappa, S.Subba Lakshmi

CSE Department, G.Pulla Reddy Engineering College(Autonomous)
Nandyala Road,Kurnool,Andhra Pradesh, INDIA

***Abstract*—In this paper we consider the former accepted definition of W.H.Innmon and is rewritten in terms of well established later temporal concepts. With reference to a three-level architecture, we address some topics namely: Temporal databases and granularity system, handling changes in the data warehouse, handling changes in the data mart, and designing temporal data warehouses and a survey is made on these concepts.**

***Keywords*—Data warehouse, data mart, design evolution, temporal databases.**

I. INTRODUCTION

A Data warehouse is defined as an environment of information and multidimensional database whose content is subject-oriented, integrated, time-variant, and nonvolatile collection of data, from various operational databases, for making strategic decisions in the business. Since the decision process typically requires an analysis of historical trends, time and its management acquire a huge importance. Data warehousing is the design and implementation of processes, tools, and facilities to manage and deliver complete, timely, accurate, and understandable information for decision making. It includes all the activities that make it possible for an organization to create, manage, and maintain a data warehouse or data mart.

"Time-variance" simply specifies that every record in the data warehouse is accurate relative to some moment in time. On the other hand, the definition of "Valid Time" states that it is the time when the fact is true in the modeled reality. Moreover,"Non-volatility" refers to the fact that changes in the Data warehouse are captured in the form of a "time variant snap shot". Instead of true updates, a new snapshot is added to the Data warehouse in order to reflect changes. This concept can be clearly identified with that of "Transaction Time", defined as the time when the fact is current in the data base.

A Data warehouse is a Bitemporal database containing integrated, subject-oriented data in support of the decision making process, as it is sketched in Fig.1 that relies on three levels[1].The first implication of this definition is that Transaction Time is entirely maintained by the system and no user is allowed to change it. Moreover the system should also provide specific management mechanisms for valid time. The bitemporal data warehouse definition shows how the existence of the temporal dimension is inferred from its definition.

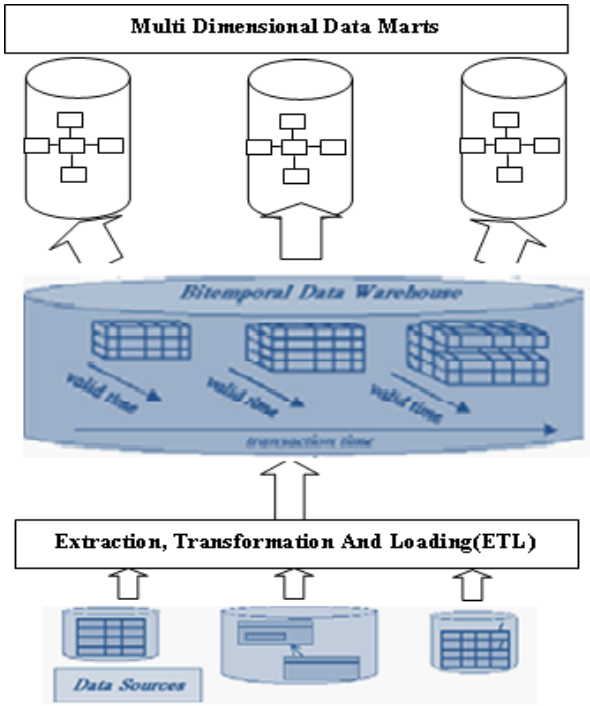

Fig.1 A Bitemporal three level architecture of a data warehouse.

There has been a lot of research so far regarding temporal issues in data warehousing systems. Basically, the approaches devised in the literature can be accommodated in the following (sometimes overlapping) categories:

- Handling changes in the data warehouse.
- Handling changes in the data mart.
- Designing temporal data warehouses

The paper outline is completed by the second section that introduces the main concepts and terminology of temporal databases and granularity system. The last section summarizes some open issues and draws the conclusions.

II. TEMPORAL DATABASES AND GRANULARITY SYSTEM.

Databases where time is not represented are often called transient databases. Within a transient database, only the current representation of real-world objects is stored and no track of changes is kept, so it is impossible to reconstruct how the object was in the past. Conversely, temporal databases focus on representing the inherent temporal nature of objects through the time-dependent recording of their structure and state. Two different time

dimensions are normally considered in temporal databases, namely valid time and transaction time [2]. Valid time is the "real-world time", i.e., it expresses the time when a fact is true in the business domain. Transaction time is the "database system time", i.e., it expresses the time when facts are registered in the database. Temporal database systems are called valid time databases, transaction-time databases or bi-temporal databases depending on their capacity to handle either or both of these two time dimensions [3].

Multidimensional modeling perfectly suits for small departmental data warehouses. Hence data marts are built. Data marts contain a partial history of data or data current at a given time. Therefore it is not only necessary to extract temporal data, but also to convert bitemporal data in the data warehouse to data structures with only one temporal dimension in the data marts. In this sense it is very helpful to apply the valid time and transaction time snapshot operations of temporal databases [4]. The valid time snapshot operation is applied to extract, from a bitemporal relation, the tuples valid at a given time and the transaction time snapshot operation is applied to extract the tuples current at a given time.

A data base which allows facts to be expressed in terms of different granularities is called a temporal database with multiple granularities. The notion of granularity system is an excellent temporal database tool to solve the data warehouse problem of different granularities integration.Formally,in temporal data bases, a granule is a set of time instants perceived as a non decomposable temporal entity when used to describe a phenomenon or when used to time stamp a set of data. A granule can be composed of a single instant, a set of contiguous instants (time intervals), or even a set of non-contiguous instants. The use of this algebra [5] for symbolic manipulation of granularities could be a well data warehouse integration solution.

III. Handling Changes In The Data Warehouse.

This mainly has to do with maintaining the data warehouse in sync with the data sources when changes on either of these two levels occur.

A. Temporal Study of Data Sources

The input data of the Data warehouse is provided by the data sources that are integrated. Depending on whether the data sources manage Transaction Time and Valid Time or not, we could obtain the Valid Time for the Data warehouse or not. Transaction Time in the data warehouse can always be obtained, because it is internal to a given storage system. When an event is loaded into the Data warehouse, its Valid Time, supplied by the "Extraction, Transformation and Load" (ETL) module, is transformed into a bitemporal element, adding Transaction Time, generated by the data warehouse DBMS.Different kinds of data sources can be classified based on the temporal information we could obtain from them (see Fig.2):

1. From "snapshot" and "queriable" sources that do not keep any kind of time, we can only store the TT (Transaction Time) in the data warehouse.
2. From "logged" and "specific" (those able to write "delta files") sources, if they timestamp the entries, we can consider that the TT in the sources corresponds to VT(Valid Time) for the data warehouse. If no other information exists, the data is considered valid while it is current in the operational database.
3. From "cooperative" (for instance, those that implement triggers) sources, the TT in the sources corresponds again to the VT in the data warehouse. Moreover, since both repositories are updated at the same time, TT in the data warehouse also corresponds to TT in the sources.
4. From bitemporal data sources, we could obtain VT and TT.

Out of this four situations, the most common is the second one.

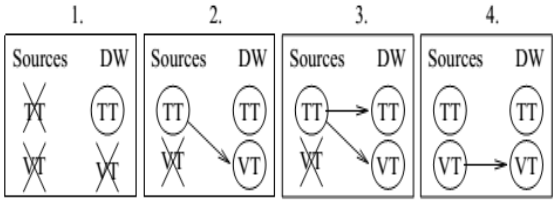

Fig 2 Transformations of time attributes

B. Description of Delta Operations

The delta files contain time stamped inserts, updates and deletes of values in the data sources. Let us analyze the effect that each of them has in the data warehouse:

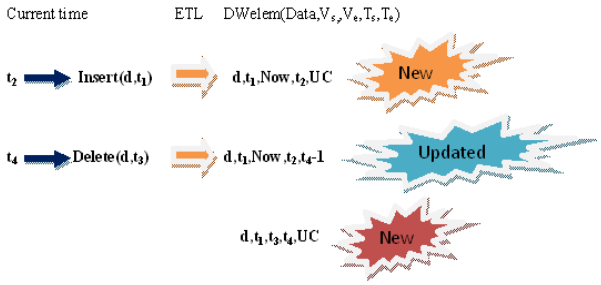

Fig 3 Effect of delta operations

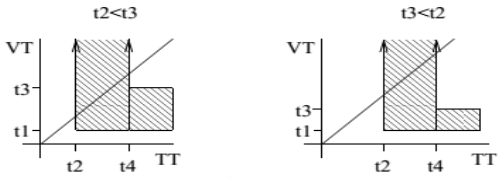

Fig 4 Valid area for VT and TT values

Insert: When an insertion (time stamped with the operational TT) is found in a delta file a new bitemporal element is always generated in the data warehouse (as depicted in Fig.3). A bitemporal event occurs at a "starting VT" (V_s) and is true until an "ending VT" (V_e). The V_s corresponds to the timestamp in the delta file (i.e. t1). However, V_e is not known at this moment, since data is currently valid in the sources. This is expressed with the special VT value "Now", whose semantics are explained in [6]. For example, if we hire an employee, the V_s will be the

time when her/his data are introduced in the personnel database, and the V_e will be the value "Now", until s/he is fired. Insertions initialize the "starting TT" (T_s) to the current time (i.e. t2) and the "ending TT" (T_e) to the value "Until changed" (UC). As the current time inexorably advances, the value of UC always reflects the current time.

Delete: A deletion (also time stamped with the operational TT) generates the logical removal of the existing bitemporal element in the DW. The value of T_e is changed to the current time, when the DW is loaded, minus one (i.e. t4-1). However, as it is shown in Fig.3, this is not enough. A new bitemporal element is required, which expresses that from now on we know the V_e , i.e. the timestamp in the delta file (i.e. t3). Therefore, a deletion in the data sources implies an update and an insert in the DW.

Update: Without loss of generality, the modification is defined by the deletion of old data immediately followed by the insertion of new values.

Fig. 4 graphically shows the four temporal points of Fig 3. Since we cannot delete data that was not previously inserted, it is true that t1<t3. Moreover, the delete will always be found strictly after the insert in the delta file. Therefore, t2<t4 and for every data there exist two bitemporal rectangles. One of them is open at top, because V_e is "Now". Notice that all time values in the delta files are previous to the load of the DW, which implies that t1<t2 and t3<t4. Since t1<t2, both rectangles have the bottom line below the diagonal of the graphic (which represents events that occur at the same time that they are recorded in the DW). The fact that t3< t4 implies that the closed rectangle also has the top line below the diagonal. However, nothing can be said about the relationship between t2 and t3, because the deletion could happen between the insertion and its load (t3<t2) or after the load of the insertion (t2<t3). Nevertheless, since t2 and t3 are never used in the same temporal element, it only has effect on the position of the rectangles along the TT axis.

When considering temporal data, it is first of all necessary to understand how time is reflected in the database, and how a new piece of information affects existing data. From this point of view, Devlin (1997) proposes the following classification [7]:

- **Transient data:** alterations and deletions of existing records physically destroy the previous data content.
- **Periodic data:** once a record is added to a database, it is never physically deleted, nor is its content ever modified. Rather, new records are added to reflect updates or deletions. Periodic data thus represent a complete record of the changes that have occurred in the data.
- **Semi-periodic data:** in some situations, due to performance and/or storage constraints, only the more recent history of data changes is kept.
- **Snapshot data:** a data snapshot is a stable view of data as it exists at some point in time, not containing any record of the changes that determined it. A series of snapshots can provide an overall view of the history of an organization.

Data sources normally adopt either a transient or a (semi-)periodic approach, depending on whether the application domains require keeping history of past data or not. The historical depth of a data warehouse is typically not less than the one of its data sources, thus data warehouses more often contain periodic data. Conversely, data marts normally conform to the snapshot model

In order to model historical data in the data warehouse, Abello and Martin (2003) propose a bi-temporal storage structure [8] where each attribute is associated to two couples of timestamps, so as to track the history of its values according to both valid and transaction time. Since the data warehouse can be thought of

as a set of derived, materialized views defined over a set of source schemata, the problem of evolving the content and the schema of derived views in connection to the source changes is highly relevant in the context of temporal data warehouses. Bellahsene (2002) distinguishes two sub problems, view maintenance and view adaptation [9]. Considering the width of the problem, we refer the reader to Gupta & Mumick (1995) for taxonomy of view maintenance problems [10] and a description of the main techniques proposed in the literature.

Yang & Widom (1998)[11] describe an architecture that uses incremental techniques to automatically maintain temporal views over non-temporal source relations, allowing users to ask temporal queries on these views. De Amo & Halfeld Ferrari Alves (2000) present a self-maintainable temporal data warehouse that, besides a set of temporal views, includes a set of auxiliary relations containing only temporal information. Bellahsene (1998) [12] proposes an extended relational view model to support view adaptation, aimed at maintaining data coherence and preserving the validity of the existing application programs.

In the EVE framework (Lee, Nica, & Rundensteiner, 2002), in order to automate the redefinition of a view in response to schema changes in the data sources, the database administrator is database administrator is allowed to embed her preferences about view evolution into the view definition itself. The preference-based view rewriting process, called view synchronization, identifies and extracts appropriate information from other data sources as replacements of the affected components of the original view definition, in order to produce an alternative view that somehow preserves the original one. Finally, the DyDa framework (Chen, Zhang, & Rundensteiner, 2006) supports compensating queries that cope with erroneous results in view maintenance due to concurrent updates in data source, in presence of data and schema changes.

A distinctive feature of the AutoMed system (Fan & Poulovassilis, 2004) is the capability of handling not only schema evolutions in materialized data integration scenarios, but also changes in the data model in which the schema is expressed (e.g., XML vs. relational).

With reference to the problem of keeping the data warehouse in sync with the sources, Wrembel and Bebel (2007) propose a metamodel for handling changes in the operational data sources, which supports the automatic detection of structural and content changes in the sources and their automatic propagation to the data warehouse. Finally, Combi & Oliboni (2007) focus on the management

of time-variant semi-structured XML data within the data warehouse. In particular, they propose a representation based on graphs whose nodes denote objects or values and are labeled with their validity interval.

IV. HANDLING CHANGES IN THE DATA MART

Content changes result from user activities that perform their day-to-day work on data sources by means of different applications (Wrembel & Bebel, 2007). These changes are reflected in the data warehouse and then in the data marts fed from it. The multidimensional model provides direct support for representing the sequence of events that constitute the history of a fact: by including a temporal dimension (say, with date granularity) in the fact, each event is associated to its date. For instance, if we consider an ORDER fact representing the quantities in the lines of orders received by a company selling PC consumables, the dimensions would probably be product, order Number, and order Date. Thus each line of order would be associated to the ordered product, to the number of the order it belongs to, and to the order date. On the other hand, the multidimensional model implicitly assumes that the dimensions and the related levels are entirely static. This assumption is clearly unrealistic in most cases; for instance, considering again the order domain, a company may add new categories of products to its catalog while others can be dropped, or the category of a product may change in response to the marketing policy. Another common assumption is that, once each line of order has been registered in a data mart, it is never modified so that the only possible writing operation consists in appending new line of orders as they occur. While this is acceptable for a wide variety of domains, some applications call for a different behavior; for example the quantity of a product ordered in a given day could be wrongly registered or could be communicated after the ETL process has run. These few examples emphasize the need for a correct handling of changes in the data mart content. Differently from the problem of handling schema changes, the issues related to data changes have been widely addressed by researchers and practitioners, even because in several cases they can be directly managed in commercial DBMSs.

V. DESIGNING TEMPORAL DATA WAREHOUSES

It is widely recognized that designing a data warehousing system requires techniques that are radically different from those normally adopted for designing operational databases (Golfarelli & Rizzi, 1999). On the other hand, though the literature reports several attempts to devise design methodologies for data warehouses, very few attention has been posed on the specific design issues related to time. Indeed, as stated by Rizzi et al. (2006), devising design techniques capable of taking time and changes into account is one of the open issues in data warehouse research. Pedersen and Jensen (1999) [13] recognize that properly handling time and changes is a must-have for multidimensional models. Sarda (1999) summarizes the distinguishing characteristics of time dimensions: they are continuously valued and constantly increasing, they can be associated with multiple user-defined calendars, they express the validity of both facts and other dimensions (either in the form of time instants or validity intervals). Sarda also proposes a design methodology for temporal data warehouses featuring two phases: logical design, that produces relations characterized by a temporal validity, and physical design, that addresses efficient storage and access.Considering the leading role played by temporal hierarchies within data marts and OLAP queries, it is worth adopting adhoc approaches for their modeling not only from the logical, but also from the conceptual point of view. While all conceptual models for data marts allow for temporal hierarchies to be represented like any other hierarchies, to the best of our knowledge the only approach that provides ad hoc concepts for modeling time is the one by Malinowski & Zimányi (2008), based on a temporal extensions of the MultiDim conceptual model. Different temporality types are allowed (namely, valid time, transaction time, lifespan, and loading time), and temporal support for levels, properties, hierarchies, and measures is granted. Finally, Golfarelli & Rizzi (2007) [14] discuss the different design solutions that can be adopted in presence of late measurements, depending on the flow or stock nature of the events and on the types of queries to be executed.

VI. OPEN ISSUES AND CONCLUSIONS

In this survey we classified and discussed the issues related to temporal data warehousing.We believe that, considering the maturity of the field and the wide diffusion of data warehousing systems, in the near future decision makers will be more and more demanding for advanced temporal support. Thus, it is essential that both vendors and researchers be ready to deliver effective solutions. In this direction we envision two main open issues. On the one hand, some research aspects indeed require further investigation. For instance, support for cross version queries is not satisfactory yet, and its impact on performance has not been completely investigated; similarly, the effectiveness of view adaptation approaches is still limited.

ACKNOWLEDGEMENT:

We would like to give special thanks to K.Bala Chowdappa, and S.Subbalakshmi, Assistant Professors in CSE Department of G.Pulla Reddy Engineering College,who participated in our survey studies and paper preparation and provided valuable suggestions in this survey.

Thanks for all my faculty members, students and other authors who directly or indirectly supported me in writing this journal.

REFERENCES

[1]. Matteo Golfarelli, Stefano Rizzi,DEIS-University of Bologna, Italy."Survey Article: A survey on temporal data warehousing". International Journal of Data Warehousing & Mining, 5(1), 1-17, January-March 2009.

[2]. Jensen, C., Clifford, J., Elmasri, R., Gadia, S. K., Hayes, P. J., & Jajodia, S. (1994)." A Consensus Glossary of Temporal Database Concepts".ACM SIGMOD Record, 23(1),52-64.

[3]. Tansel, A. U., Clifford, J., Gadia, S. K., Jajodia, S., Segev, A., & Snodgrass, R. T. (1993). Temporal databases: theory, design and implementation.Benjamin Cummings.
[4]. Christian S.Jensen, Michael D.Soo, and Richard T.Snodgrass. "Extending Normal forms to Temporal Relations". Technical Report TR-92-17, Computer Science Department. University of Arizona, 1992.
[5]. Claudio Bettini, Sushil Jajodia,and X.Sean Wang. Time Granularities in databases,Data Mining and temporal reasoning.Springer-Verlag, 2000.
[6]. Clifford, J., Dyreson, C., Isakowitz, T., Jensen, C. S., and Snodgrass, R. T. (1997). On the Semantics of "Now" in Databases. ACM Transactions on Database Systems, 22(2):171–214.
[7]. Devlin, B. (1997). "Managing Time In The Data Warehouse". InfoDB, 11(1),7-12.
[8]. Abelló, A., & Martin, C. (2003). A Bi-temporal Storage Structure for a Corporate Data warehouse. Proceedings International conference on Enterprise Information Systems, Angers, France, 177-183.
[9]. Bellahsene, Z. (2002). Schema Evolution in Data Warehouses. Knowledge and Information Systems, 4(3), 283-304.
[10]. Gupta, A., & Mumick, I. S. (1995). Maintenance of materialized views: problems, techniques, and applications. Data Engineering Bulletin, 18(2), 3-18.
[11]. Yang, J. & Widom, J. (1998). Maintaining Temporal Views over Non-Temporal Information Sources for Data Warehousing. Proceedings International Conference on Extending Database Technology, Valencia, Spain, 389-403.
[12]. Bellahsene, Z. (1998). View Adaptation in Data Warehousing Systems. *Proceedings International Conference on Database and Expert Systems Applications*, Vienna, Austria, 300-309.
[13]. Pedersen, T. B. & Jensen, C. (1999). Multidimensional Data Modeling for Complex Data. *Proceedings International Conference on Data Engineering*, Sydney, Austrialia, 336-345.
[14]. Golfarelli, M. & Rizzi, S. (2007b). Managing late measurements in data warehouses. *International Journal of Data Warehousing and Mining*, 3(4), 51-67.