International Journal of Advanced Computational Engineering and Networking, ISSN (p): 2320-2106, Volume- 1, Issue- 8, Oct-2013

A COMPARATIVE ANALYSIS OF TEMPORAL DATA MODELS

[1]PRAVEEN KUMAR GUPTA, [2]RAHUL RISHI, [3]RANJIT BISWAS

Associate Professor, Department of Computer Science and Engineering, SKITM, India
Professor, Department of Computer Science and Engineering, UIET, MD University, Rohtak, India
Professor, Department of Computer Science, Jamia Hamdard, New Delhi, India
Email: onlinepg@gmail.com, rahulrishi@rediffmail.com, ranjitbiswas@yahoo.com

Abstract: The objective of this paper is to make a concerted effort for minimizing the computing cost by comparing and analyzing the different temporal data models of databases. This is a review work. The paper begins with critical investigation of the constraints of relational model and then comparing the key concepts of various temporal data models. Finally, in the context of temporal perspective, the paper concludes with the analysis of these different models with a view to design an effective conceptual temporal database framework.

Keywords- Databases, Data Models, Normal Form, Relational Model, Temporal.

I. INTRODUCTION

The current interest in the relational approach to databases is largely due to Codd's paper [5] on relational model, containing an explanation of relational structure and associated advantages with it. Thus the theoretical foundation for the relational model through a series of papers [6][7] were laid down.

Mathematically, A relation schema R is represented as a 4-tuple: R (T_R, AN_R, n, m) where T_R represents the set of tuples, $m=|T_R|$ is the cardinality of the relation, AN_R represents the set of attribute names and n is the degree or arity of the relation. Relational model is based on classical set theory and conventional functional dependency (fd). Given a universal set X, a classical set A may be represented as $A= \{x|P(x)\}$. A set can also be defined by a function, usually called a characteristic function that declares which elements of X are members of the set and which are not. Classical sets are also called crisp sets since they contain precise data only.

One of the single most important concept, w.r.t. relational schema design, is that of functional dependency. It is a constraint between two sets of attributes X and Y where $X \subseteq R$, $Y \subseteq R$ and $R = \{A_1,A_2,...,A_n\}$. A functional dependency $X \rightarrow Y$ specifies a constraint such that for any two tuples t_1 and t_2 in r(R), whenever $t_1[X] = t_2[X]$ then $t_1[Y] = t_2[Y]$ must also exist. Thus X functionally determines Y in R iff, whenever two tuples of r(R) agree on their X-values, they must necessarily agree on their Y - values. It is also important to discuss the difference between first normal form (FNF) relations and non first normal form (NFNF) relations. A relation is in FNF when the domains of the attributes in its schema may only be of scalar data types. If not, then the relation is in NFNF.

The Codd's relational model and commercially available relational databases consider only the traditional bivalent boolean logic and do not consider the temporal dimension as they do not keep track of past or future database states. Since the implementation of relational model is in terms of precise data only, The comparison of the data of the same data types is done with classical boolean logic. But in real life problems and decision making, the data associated is often imprecise. Consequently, for comparing such data, boolean logic is inadequate. Many applications in the real world require management of time varying ing data such as financial applications, healthcare management systems, reservation systems, insurance applications and enterprise resource planning (ERP) systems. Temporal databases, in particular, may be useful in extrapolating trends and future forecasting e.g. national governments may apply it for disaster management and minimizing losses, other sectors like financial sector may use it to extrapolate trends of stock markets, and transport sector like airlines and railways may use it for better logistics and efficient functioning.

This paper is organized in six sections. Section one introduces constraints of relational model. Section two describes key concepts of temporal databases. Section three mentions about methodology of work. Section four presents the summary of data models in temporal perspective. Section five discusses about key concepts and features of these data models. Finally, section six concludes with the work reported here.

II. TEMPORAL DATABASES

Codd's relational model does not address the temporal dimension of data [7][22][28]. A database which maintains past, present and future data is called a temporal database. Temporal data stored in a temporal database is different from the data stored in non-temporal database in that a time period attached to the data expresses when it was valid or stored in the database.

In temporal databases, the first step is to consider temporal dimension, i.e. to timestamp the data. This allows the distinction of different database states. One approach is that a temporal database may timestamp entities with time periods. Another approach is the time-stamping of the property values of the entities. In the relational data model, tuples are time-stamped.

What time period is stored in these timestamps? There are mainly two different notions of time which are relevant for temporal databases. One is called the valid time (VT), the other one is the transaction time (TT). Valid time denotes the time period during which a fact is true with respect to the real world. Transaction time is the time period during which a fact is stored in the database. Note that these two time periods do not have to be the same for a single fact. Imagine that a temporal database is storing data about the 18th century. The valid time of these facts is somewhere between 1700 and 1799, whereas the transaction time starts when we insert the facts into the database, for example, January 21, 1998. The real world data about the employees can be shown as below in table 1:

TABLE 1
Temporal database example

EmpID	Name	Department	Salary	Time Start	Time End
101	Ram	Research	11K	1985	1990
101	Ram	Sales	11K	1990	1993
101	Ram	Sales	12K	1993	NOW
111	Krishna	Research	10K	1988	1995
112	Rashi	Research	10K	1991	NOW
113	Madhu	Sales	15K	1988	NOW

The above valid-time table stores the history of the employees with respect to the real world. The attributes TimeStart and TimeEnd actually represent a time interval which is closed at its lower end and open at its upper bound. Thus, we see that during the time period [1985 - 1990), employee Ram was working in the research department, having a salary of 11000. Then he changed to the sales department, still earning 11000. In 1993, he got a salary raise to 12000. The upper bound NOW denotes that the tuple is valid till current time instant (NOW). Note that it is now possible to store information about past states e.g. Krishna was employed from 1988 until 1995, which was not possible in Codd's model.

The two different notions of time - valid time and transaction time - allow the distinction of different forms of temporal databases [26]. A historical database stores data with respect to valid time, a rollback database stores data with respect to transaction time. A bi-temporal database stores data with respect to both valid time and transaction time. Commercial DBMS are said to store only a single state of the real world, usually the most recent state. Such databases usually are called snapshot databases. A snapshot database in the context of valid time and transaction time is depicted in figure 1:

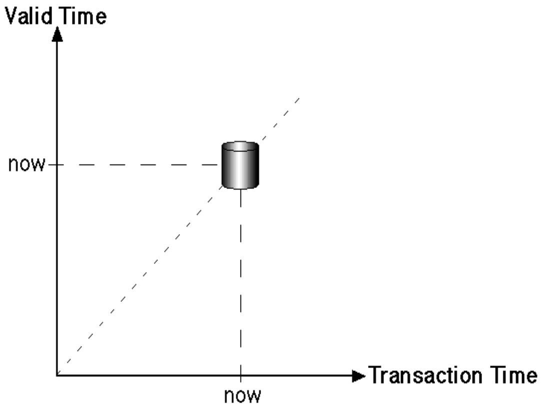

Figure 1, Bi-temporal time representation

On the other hand, a bi-temporal DBMS, stores the history of data with respect to both valid time and transaction time. Note that the history of data was stored in the database is limited to past and present database states, since it is managed by the system directly which does not know anything about future states.

III. METHODOLOGY

The comparison methodology is adopted by considering key concepts and features of the temporal data models based on relevant parameters. Some of the parameters are author's name, normal form, time stamping, time dimensions and key concepts and features of the model under consideration.

IV. TEMPORAL DATA MODELS

Most of the work in the research area of temporal databases has been done in respect of the relational data model. In this section, as shown below in table 2 and Table 3, some of the most important temporal data models are compared.

TABLE 2
Summary of data models in temporal perspective

model / parameter	Ariav model [1]	Ben-Zvi model [2]	Clifford and Croker model [3][4]	Gadia model [12][13]	Jensen & Snodgrass Model [19][20]
Time	VT & TT	VT & TT	VT	VT	VT & TT
Timestamp	Tuple	Tuple	Attrib	Attrib	Tuple
Functional	VTC	VTC	Valid	VTC	Valid

dependency					
Nornal form – conventional	S	NS	NS	NS	S
Nornal form – time	NS	S	S	S	NS
Query language support	NS	S	S	S	S
Optimization	NS	S	NS	S	NS
Temporal operator support	NS	S	S	S	S
Distribution	NS	NS	NS	NS	NS
Temporal keys	S	S	S	S	S

Attrib:Attribute; NS:Not supported; S:Supported; VTC:Validity to be checked.

TABLE 3

Summary of data models in temporal perspective

model / parameter	Lorentzos model [21]	Snodgrass Model [27]	Tansel Model [29]	Vianu model [30]	Wijsen Model [31]
Time	VT	VT & TT	VT	VT	VT
Timestamp	Attrib	Tuple	Attrib	Tuple	Attrib
Functional dependency	VTC	Valid	VTC	VTC	VTC
Nornal form – conventional	S	NS	NS	S	S
Nornal form – time	NS	S	S	NS	NS
Query language support	NS	S	S	NS	NS
Optimization	NS	S	S	NS	NS
Temporal operator support	S	S	S	S	S
Distribution	NS	NS	NS	NS	NS
Temporal keys	S	S	S	S	S

Attrib:Attribute; NS:Not supported; S:Supported; VTC:Validity to be checked.

V. KEY FEATURES OF TEMPORAL DATA MODELS

Ariav's model [1] used tuple time stamping with time being represented by discrete time points in bi-temporal mode. The model is conceptually simplistic but difficult to implement in efficiency and reliability terms.

Ben-Zvi's time relational model (TRM) [2] was a pioneering work in many aspects. The most important idea of TRM is perhaps the non first normal form (NFNF). Ben-Zvi's concept of effective time and registration time, which are now known as valid time (VT) and transaction time (TT), respectively, added new dimensions to time varying information computing. Ben-Zvi was the first to coin the term and notion of time-invariant key for his non first normal tuples, called tuple version sets in his terminology. Differentiation between an error and a change was recognized and both of them were made queriable. Also the need for fast access to current data was recognised.

Clifford and Croker's model [3][4] followed historical relational data model (HRDM) and tuples are heterogeneous in their temporal dimension. Unfortunately, the historical relational algebra is not a complete algebra w.r.t. HRDM. So, the cartesian product of three-dimensional relation (e.g. join operation) is not clear and hence results are not reliable.

Gadia's model [12][13] has temporal element as an appropriate datatype for time. This model assumes that key values donot change with time. Another requirement is all attributes in a tuple have the same time domain. This requirement is called homogeneity. The positive aspect of Gadia's model is that it minimizes redundancy. But when concatenation of partial temporal elements along with tuple homogeneity is implemented, the query results into incomplete or missing information.

Jensen & Snodgrass model [19][20] proposed bi-temporal conceptual data model (BCDM), allowing to associate both valid and transaction times with data. [28] The domains of valid and transaction times are the finite sets D_{VT} and D_{TT}, respectively. A valid time chronon c_v is a time point belonging to D_{VT} and a transaction time chronon c_t is a time point belonging to D_{TT}. A bitemporal chronon cb = (c_t , c_v) is an ordered pair consisting of a transaction time chronon and a valid time chronon. The schema of a bitemporal relation R, defined on the set $U = \{A_1,A_2,...,A_n\}$ of non-timestamp attributes, is of the form $R = (A_1,A_2,...,A_n \mid T)$, that is, it consists of n non-timestamp attributes $A_1,A_2,...,A_n$, with domain dom(A_i) for each $i \in [1,n]$, and an implicit timestamp attribute T. The domain of T is $(D_{TT} \cup \{UC\}) \times D_{VT}$, where UC is a special value that can be assumed by a transaction time chronon to express the condition "until changed". For instance, to state that a tuple valid at time c_v is current in the database, the bitemporal chronon (UC, c_v) must be assigned to the tuple timestamp. As a general rule, they associate a set of bitemporal chronons in the two-dimensional space with every tuple.

Lorentzos's model [21] followed interval-extended relational model (IXRM) and an interval relational algebra for the management of interval relations. The fundamental properties of a model are that it must be satisfactory and simple. Lorentzos model satisfied both aspects. However, when a model is defined, efficiency issues are of minor importance. IXRM operations require a great deal of space and time, is a point of concern.

Snodgrass's Model [27] uses temporal query language (TQuel) which is based on the predicate calculus. One of the key features of this model is when the algebra is used to implement the TQuel, the a conversion will be necessary between tuple time-stamping (where each tuple is associated with a single interval) and attribute-value time-stamping (where each attribute is associated with potentially multiple intervals). This conversion is formalized in a transformation function (T). Though this model seems to be more efficient but relatively less user friendly.

Tansel's model [29] used Attribute-value time-stamping and used the concepts of time by example (TBE) and query by example (QBE). This model is quite user friendly. However, nested temporal relations are an area of concern since structuring nested temporal relations hinges upon the type of associations between the involved entities.

Vianu proposed a simple extension of the relational data model in order to describe evolution of a database over time.[30] A database sequence is defined as a sequence of consecutive instances of the database, plus "update mappings" from one instance (the "old" one) to the next one (the "new" instance). Constraints on the evolution of attribute values of tuples (objects) over time are expressed by means of dynamic functional dependencies (DFDs), that make it possible to define dependencies between old and new values of attributes on updates.

Wijsen and his colleagues temporal data model [31] proposed three types of keys i.e. snapshot keys (SK), dynamic keys (DK) and temporal keys (TK) corresponding to snapshot functional dependency, dynamic dependency and temporal dependency, respectively. Let dom be a set of atomic values, that is, the union of disjoint domains corresponding to atomic types, att be a set of attribute names, and λ be a special attribute used to denote object identity. Moreover, let obj be an infinite set of object identifiers (OIDs) and class be a set of class names. Given a finite set of class names C, a type over C is a set $\{A_1: \tau_1, A_2: \tau_2,..., A_n: \tau_n\}$, where $A_1, A_2,... A_n$ are distinct attribute names and each τ_i with $1 \leq i \leq n$, is either an atomic type or a class name in C. A schema is a pair (C,ρ), where C is a finite set of class names and ρ is a total function that maps each class name in C into a type over C.

Some data models use FNF and others prefer NFNF. The choice may depend on the consideration of time as discrete or interval based or continuous. Also, the traditional Entity relationship model (ERM) can be extended for temporal data models (TDM) by considering suitable operators and constructs for their effective and efficient implementation. The traditional ERM is capable of capturing the whole temporal aspects. Many extensions [15][16] have been proposed to extend the ERM in order to capture time varying information. For graphical representation, Unified modeling language (UML) is normally used. However, UML constructs in reference of temporal data models can be possibly used and drawn using application softwares like Rational software architecture.

The temporal query language (TQuel) [27] supports both valid time and transaction time. It also supports user defined time. Tuples are optimally time-stamped with either a single valid time stamp (if a relation models events) or a pair of valid timestamps (if a relation models intervals), along with transaction timestamps, denoting when the tuple was logically inserted into the relation. A transaction timestamp of "until changed" indicates that the tuple has not been deleted yet.

A functional example of temporal database is TimeDB [32]. It uses the extension approach with respect to the data structures. TimeDB uses a layered approach which means it was built as a front end to a commercial DBMS that translates temporal statements into standard SQL statements. This way, it is possible to support features such as persistence, consistency, concurrency, recovery etc. without having to implement from the scratch. It is a bi-temporal DBMS. TimeDB implements the query language ATSQL2. ATSQL2 includes not only a bi-temporal query language but also a bi-temporal modification, data definition and constraint specification language. TimeDB implements the temporal algebra operations using standard SQL statements. TimeDB supports a command oriented user interface.

CONCLUSION

These models are analyzed carefully with a view to employing techniques to minimize complexity and computing costs within the framework of temporal database design. These temporal data models are reviewed for a range of parameters. Whether it is Snodgrass's transformation function (T) which takes relatively less time and space, or Tansel's model for effectively structuring the temporal relations to avoid data redundancy, the relations should exhibit a natural and intuitive conceptualization of reality and the inherent complexity of temporal data models. Temporal data models need careful attention for balanced approach by taking care of the efficiency aspects during implementation and abstraction aspects for user-friendliness.

REFERENCES

[1] Ariav, G., “A Temporally Oriented Data Model”, ACM Transactions on Database Systems, 1986, 11(4), pages 499-527.

[2] Ben-Zvi, “The Time Relational Model”, 1982, PhD thesis, Computer Science Department, UCLA.

[3] Clifford, J. and Croker, A., “The Historical Relational Data Model (HRDM) and Algebra Based on Lifespans”, In Proceedings of the International Conference on Data Engineering. IEEE Computer Society Press, 1987, pages 528-537.

[4] Clifford, J. and Croker, A., “The Historical Relational Data Model (HRDM) Revisited”, In A. Tansel, J. Clifford, S. Gadia, S. Jajodia, A. Segev, and R. Snodgrass, editors, Temporal Databases: Theory, Design, and Implementation, Benjamin/Cummings Publishing Company, 1993, pages 6-27.

[5] Codd, E. F., “A relational model of data for large shared data banks”, Comunications of the ACM, 1970, Vol. 13, No. 6.

[6] Codd, E. F. and Rustin R., “Further Normalisation of the Database relational model”, Database Systems, Englewood Cliffs, New Jersey: Prentice-Hall, 1972, pages 33-64.

[7] Codd, E. F., “Recent investigation in relational database systems”, Proceedings of IFIP 74, 1974, pages 1017-1021.

[8] Date, C. J., “An introduction to database systems”, Narosa Publishing House, 1999, 3rd edition.

[9] Desai, B.C., “An introduction to Database Systems”, Galgotia Publications (West Publishing Co., USA), 2000, 4th edition.

[10] Elmasri, R. and Navathe, S. B., “Fundamentals of Database management systems”, Pearson Education, 2003, 3rd edition

[11] Fox J., "Towards a reconciliation of fuzzy logic and standard logic," International Journal of Man-Machine Studies, 1981, Vol. 15, pages 213-220

[12] Gadia, S. K., “A Homogeneous Relational Model and Query Languages for Temporal Databases”, ACM Transactions on Database Systems, 1988, 13(4), pages 418-448.

[13] Gadia, S. K., “A Seamless Generic Extension of SQL for Querying Temporal Data”, Computer Science Department, Iowa State University, 1992, Technical report TR 92-02.

[14] Gougen, J.A., “The logic of inexact concepts”, Synthese, 1969, Vol. 19, pages 325-373

[15] Gregersen, H. and Jensen, C. S., “ Conceptual Modeling of Time-varying Information” , In Proceedings of International Conference on Computing, Communications and Control Technologies, 2004, pages 248–255.

[16] Gregersen, H., “TimeERplus: A Temporal EER Model Supporting Schema Changes”, In BNCOD, 2005, volume 3567 of Lecture Notes in Computer Science, Springer, pages 41–59.

[17] Hamouz, S. and Biswas, R., “Fuzzy functional dependencies in Relational Databases”, International Journal of Computer Cognition, 2006, Vol. 4, No. 1, pages 39-43

[18] Jensen, C. S. and Dyreson, C. E. [editors], “The Consensus Glossary of Temporal Database Concepts” In Temporal Databases: Research and Practice, 1998, volume 1399 of Lecture Notes in Computer Science, Springer-Verlag, pages 367–405.

[19] Jensen, C. S. and Snodgrass, R. T., “Proposal for a Data Model for the Temporal Structured Query Language” Department of Computer Science, University of Arizona, 1992, TEMPIS technical report number 37.

[20] Jensen, C. S. and Snodgrass, R. T., “Temporal Data Management.”, IEEE Transactions on Knowledge and Data Engineering, 1999, volume 11, no. 1, pages 36-44.

[21] Lorentzos, N. A., “The Interval-extended Relational Model and its Application to Validtime Databases.”, Temporal Databases: Theory, Design, and Implementation, Benjamin/Cummings Publishing Company, 1993, chapter 3, pages 67-91.

[22] Mahmood N., Burney A., Ahsan K., “A logical temporal relational data model”, International Journal of computer science issues, 2010, Volume 7, Issue 1, No. 1.

[23] Osherson, D.N., & Smith, E.E., “On the adequacy of prototype theory as a theory of concepts”, Cognition, 1981, Vol. 9, pages 35-38.

[24] Osherson, D.N., & Smith, E.E., “Gradedness and conceptual combination”, Cognition, 1982, Vol. 12, pages 299-318.

[25] Black, Max, “Vagueness: An exercise in logical analysis", Philosophy of Science, 1937, volume 4, pages 427–455.

[26] Radecki T., “An evaluation of the fuzzy set theory approach to information retrieval”, Progress in Cybernetics and System Research, 1982, Vol. 11: Proceedings of a Symposium Organized by the Austrian Society for Cybernetic Studies, Hemisphere Publ. Co.

[27] Snodgrass, R. T., “The Temporal Query Language TQuel.” ACM Transactions on Database Systems, 1987, volume 12(2), pages 247- 298.

[28] Snodgrass R., Jensen C. and Soo M., “Extending existing dependency theory to temporal databases”, IEEE Transactions on Knowledge and Data Engineering, 1996, Vol. 8(4), pages 563–581.

[29] Tansel, A.U., Clifford, J., Gadia, S., Jajodia, S., Segev, A. and Snodgrass, R., “Temporal Databases: theory, design and implementation”, Benjamin cummins publishing company, 1999, 1st Edition.

[30] Vianu, V., “Dynamic functional dependency and database aging”, Journal of the ACM, 1987, vol. 34(1), pages 28–59.

[31] Wijsen J., “Design of temporal relational databases based on dynamic and temporal functional dependencies” International Workshop on Temporal Databases, 1995, Springer pub., pages 61–76.

[32] Steiner A. “A time relational DBMS: TimeDB”, Ph. D. thesis, 1998, Chapter 4, pages 53- 79

★★★