ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

Sixth Normal Form

Neha[1], Sanjeev Kumar[2]
[1]M.Tech, [2]Assistant Professor, Department of CSE, Shri Balwant College of Engineering &Technology, DCRUST University

***Abstract* – Sixth Normal Form (6NF) is a term used in relational database theory by Christopher Date to describe databases which decompose relational variables to irreducible elements. While this form may be unimportant for non-temporal data, it is certainly important when maintaining data containing temporal variables of a point-in-time or interval nature. With the advent of Data Warehousing 2.0 (DW 2.0), there is now an increased emphasis on using fully-temporalized databases in the context of data warehousing, in particular with next generation approaches such as Anchor Modeling . In this paper, we will explore the concepts of temporal data, 6NF conceptual database models, and their relationship with DW 2.0. Further, we will also evaluate Anchor Modeling as a conceptual design method in which to capture temporal data. Using these concepts, we will indicate a path forward for evaluating a possible translation of 6NF-compliant data into an eXtensible Markup Language (XML) Schema for the purpose of describing and presenting such data to disparate systems in a structured format suitable for data exchange.**
***Keywords* :, 6NF,SQL,DKNF,XML,Semantic data change, Valid Time, Transaction Time, DFM**

I. INTRODUCTION

Normalization is the process of restructuring the logical data model of a database to eliminate redundancy, organize data efficiently and reduce repeating data and to reduce the potential for anomalies during data operations. Sixth normal form as a normal form for databases based on an extension of the relational algebra. A relational variable R [table] is in sixth normal form (abbreviated 6NF) if and only if it satisfies no nontrivial join dependencies at all — where, as before, a join dependency is trivial if and only if at least one of the projections (possibly U_ projections) involved is taken over the set of all attributes of the relational variable [table] concerned. This normal form was, as of 2005, only recently proposed: the sixth normal form (6NF) was only defined when extending the relational model to take into account the temporal dimension (time). Unfortunately, most current SQL technologies as of 2005 do not take into account this work, and most temporal extensions to SQL are not relational. A relation in 6NF is also in 5NF.Sixth normal form is intended to decompose relation variables to irreducible components. Though this may be relatively unimportant for non-temporal relation variables, it can be important when dealing with temporal variables or other interval data. For instance, if a relation comprises a supplier's name, status, and city, we may also want to add temporal data, such as the time during which these values are, or were, valid (e.g., for historical data) but the three values may vary independently of each other and at different rates. We may, for instance, wish to trace the history of changes to Status.

II. DOMAIN KEY NORMAL FORM

Some authors use the term sixth normal form differently, namely, as a synonym for Domain/key normal form (DKNF) Domain/key normal form (DKNF) is a normal form used in database normalization which requires that the database contains no constraints other than domain constraints and key constraints. A domain constraint specifies the permissible values for a given attribute, while a key constraint specifies the attributes that uniquely identify a row in a given table.
The domain/key normal form is achieved when every constraint on the relation is a logical consequence of the definition of keys and domains, and enforcing key and domain restraints and conditions causes all constraints to be met. Thus, it avoids all non-temporal anomalies. The reason to use domain/key normal form is to avoid having general constraints in the database that are not clear domain or key constraints. Most databases can easily test domain and key constraints on attributes. General constraints however would normally require special database programming in the form of stored procedures that are expensive to maintain and expensive for the database to execute. Therefore general constraints are split into domain and key constraints. It's much easier to build a database in domain/key normal form than it is to convert lesser databases which may contain numerous anomalies. However, successfully building a domain/key normal form database remains a difficult task, even for experienced database programmers. Thus, while the domain/key normal form eliminates the problems found in most databases, it tends to be the most costly normal form to achieve. However, failing to achieve the domain/key normal form may carry long-term, hidden costs due to anomalies which appear in databases adhering only to lower normal forms over time.

III. TEMPORAL DATABASE DESIGN AND 6NF DATA MODELING

In This Paper Temporal Data and the Relational Model, Chris Date points out several shortcomings when attempting to model fully temporalized data using standard Fifth Normal Form (5NF) principles. To illustrate these shortcomings, first consider a

non-temporalaized , 5NF-compliant table schema (see Figure 1) in which is defined a predicate for the Suppliers relation variable (relvar) S as such: Supplier S# is under contract, is named SNAME, has status STATUS , and is located in city CITY:

S

S#	SNAME	STATUS	CITY
S1	SMITH	20	LONDON

Fig 1. Non-temporalized Table Schema

To introduce a temporal element into table S, we simply add a temporal attribute SINCE to the table which accounts for valid time (see Figure 2). The resulting table S_SINCE now represents that supplier S# has been under contract since day SINCE:

S_SINCE

S#	SNAME	STATUS	CITY	SINCE
S1	SMITH	20	LONDON	D04

Fig 2. Semi-temporalized Table Schema using SINCE

Alternatively, we may want to track an interval period during which a relvar is true (see Figure 3). The resulting table S_DURING would represent that supplier S# was under contract during time period DURING:

S_DURING

S#	SNAME	STATUS	CITY	DURING
S1	SMITH	20	LONDON	[D04-D06]

Fig 3. Semi-temporalized Table Schema using **DURING**

In both of these cases, even though a temporal element has been introduced into the database, the relvars are badly designed for two reasons: 1. One temporal attribute alone is insufficient to model conditions where each of the other individual attributes may vary independently of one another over time, and 2. Data can be lost when single temporal attributes are updated. Historical data thus cannot be tracked in this scenario. Due to these shortcomings, Date refers to this type of modeling as "semi-temporalizing" (Date, Darwen and Lorentzos, 2002).

IV. USAGES

The sixth normal form is currently being used in some data warehouses where the benefits outweigh the drawbacks, for example using Anchor Modeling. Although using 6NF leads to an explosion of tables, modern databases can prune the tables from select queries (using a process called 'table elimination') where they are not required and thus speed up queries that only access several attributes.

V. ANALYSIS OF ANCHOR MODELING

Anchor Modeling is a database modeling technique built on the premise that the environment surrounding a data warehouse is in a constant state of change, and furthermore that a large change on the outside of the model should result in only a small change on the inside of the model .The goal of using the anchor modeling technique is to "achieve a highly decomposed implementation that can efficiently handle growth of the data warehouse without having to redo any previous work", mimicking the ideas of isolated semantic temporal data put forth in DW 2.0 architecture. When anchor models are translated to physical database designs, content changes are emulated using standard 6NF principles and the tables in the resulting relational database will be 6NF-compliant. Pieces of data are tied to points in time or intervals of time. Time points are modeled as attributes (i.e., the time a part is shipped), and intervals of time are modeled as a historization of attributes or ties (i.e., times during which a part was in stock). Transaction time elements, such as the time information entered or was updated in the database, are handled by metadata and are not included in the standard anchor modeling constructs.

Anchor Modeling provides a graphical notation for conceptual database modeling similar to Entity-Relationship (ER) modeling, with extensions for temporal data. The model is based on 4 constructs: the Anchor, Attribute, Tie, and Knot. Anchors model entities and events, Attributes model properties of Anchors, Ties model relationships between Anchors, and Knots model shared properties.

A. Analysis of normal forms for anchor-tables

Let R be a relation and A, B, …, Z subsets of the attributes of R. A super-key for a relation R is a superset (not a proper one necessarily) of some candidate key for R. I.e. all candidate keys are super-keys but some super-keys are not candidate keys. R is said to satisfy the join dependency * {A, B, … Z} iff every legal value of R (i.e. every tuple of R) is equal to the join (natural join operator is assumed) of its (R that is) projections on A, B, … Z.

In other words: R satisfies * {A, B, … Z} iff R can be non loss-decomposed into the projections R(A), R(B), … R(Z). A join

dependency * {A, B, … Z}on R is trivial iff at least one of A, B, …, Z contains all the attributes of R. A join dependency * {A, B, … Z}on R is implied by the candidate key(s) of R iff each of A, B, …, Z is a super-key for R. A relation R is in 5NF iff every nontrivial join dependency that is satisfied by R is implied by the candidate key(s) of R. A table is in 6NF iff it satisfies no nontrivial join dependencies at all. Note the difference between 5NF and 6NF, for 5NF non-trivial join dependencies may be satisfied as long as these dependencies are implied by the candidate key(s), for 6NF no non-trivial join dependencies at all are allowed to be satisfied.

B. Anchor , knot- and attribute-tables

An anchor A(C) is a table with one column. The domain for C is ID. The primary key for A is C. A(C) is clearly in 6NF, no projection other than on the one attribute is possible. A knot K(S; V) is a table with two columns. The domain of S is ID, and the domain of V is a data type and never null. The primary key for K is S.

K(S,V) is in 3NF (assuming it is indeed in 1NF, it is in 2NF since the key is not composite, it is in 3NF since no transitive dependencies exist, it is in BCNF since we only have one determinant and it is our one candidate key, it is in 4NF since the existence of multi-valued dependencies implies at least three attributes, now for 5NF: the only two (V).

C. Tie-tables

Tie-tables that are all-key give rise to tables in 6NF (obviously, decomposing an all-key table clearly gives rise to spurious tuples when joining the projections, so no non-trivial join-dependencies will ever be satisfied by an all-key tie relation). Whether or not non all-key tie-tables are in 6NF depends on what functional dependencies hold in the Universe of Discourse that is to be represented by an anchor model. Non all-key tie-tables are neither more nor less normalized compared to relational tables based on any other modeling approach than anchor modeling. It shall be noted, however, that tie-tables (i.e. tables that correspond to relationships in ER-modelling) with n columns and less than n-1 columns in the key is very unusual according to our modeling experiences.

VI. CONCLUSION

With the emergence of data warehousing and the emphasis placed on the advantages of using temporal modeling in DW 2.0, the issue of how to store and model changes to data over time has become one of practical importance to business and industry. In this paper, we have explored a generalization of temporal aspects in database and data warehouse design using Sixth Normal Form (6NF) and Anchor Modeling techniques. By using such methods, snapshots of data may be captured which generate a historical record of data by which an analyst or end-user can easily locate and retrieve information while also providing a business with a technological infrastructure that can withstand change over time. There is now a need to describe and present temporal data to disparate data warehouse systems in a structured format suitable for data exchange across the internet and other networked resources. Since XML has become the preferred means of data exchange, an adequate definition of a standard XML Schema document definition template that describes 6NF data is desired. It is our intention to further this research with the result of producing an original standardized XML Schema mechanism for exchanging such information. We also hope to indicate advantages in application areas benefiting from the use of the proposed formats, as well as limitations. Researchers have previously described an XML anchor schema definition in the paper “From Anchor to XML” .We hope to expand upon this work by using meta models to address a logical level of design in defining a XML schema profile similar to one previously employed to describe Unified Modeling Language (UML) class diagrams. Ultimately, the outcome should define a method for using XML schema to communicate with DW 2.0-compliant temporal data warehouses.

REFERENCES

[1] Date, C.J., Darwen, H. and Lorentzos, N. (2002) Temporal data and the relational model: A detailed investigation into the application of interval and relation theory to the problem of temporal database management, Morgan Kaufmann Publishers, Amsterdam.
[2] Golfarelli, M. and Rizzi, S. (2009) Data warehouse design: Modern principles and methodologies, McGraw Hill, New York.
[3] Inmon, W.H., Strauss, D. and Neushloss, G. (2008) DW 2.0: The architecture for the next generation of data warehousing, Morgan Kaufmann Publishers, Amsterdam.
[4] Rönnbäck, L., Regardt, O., Bergholtz, M., Johannesson, P. and Wohed, P. (2010) Anchor modeling - agile information modeling in evolving data environments, Data & Knowledge Engineering, 1229–1253.
[5] Rönnbäck, L., Regardt, O., Bergholtz, M., Johannesson, P. and Wohed, P. (2010) From anchor model to XML, http://www.anchormodeling.com/wp-content/uploads/2010/09/AM-XML.pdf
[6] Routlige, N., Bird, L. and Goodchild, A. (2002) UML and XML Schema, 13th Australian Database Conference, Melbourne, Australia, 1-10.
[7] Snodgrass, R. and Ahn, I. (1986) Temporal databases, IEEE Computer, 19(9), 35-42.