Temporal Databases

Mahesh S. Raisinghani  
*Texas Woman’s University, USA*

Chris Klassen  
*University of Dallas, USA*

INTRODUCTION

Information has emerged as an agent of integration and the enabler of new competitiveness for today's enterprise in the global marketplace. The degree of change in the paradigm for storage of data in databases is examined to determine whether it can support the accelerated response time required for information systems and technology. This paper discusses the key concepts for understanding temporal databases, including major data types and its principal purpose. Moreover, once the temporal extension is added to existing ANSI and ISO SQL standards, it will enable users to take advantage of new temporal features in the major database products.

Time has always been of great interest to mankind. In recent years, the computer industry has made a great influence in human lifestyle and it like other industries is time prone. For instance, the year 2000 problem (Y2K) was time-related. Not surprisingly, researchers working in the field of databases are expressing increasing interest in the dimension of time. Classical database design is two-dimensional and contains only current data, which can be termed snapshot data type. Today's businesses must adapt constantly to an ever-changing business environment, and databases must support an evolving business framework. More and more time-management seminars and devices are introduced everyday. As Snodgrass (1998) notes, “Time varying data is becoming pervasive. It has been estimated that one of every 50 lines of database application code involves a date or time value.”

BACKGROUND:
THE TIME DIMENSION IN DATABASES

The classical database is two-dimensional with columns and rows that intersect each other at cells, which contain particular values. Now extend this flat two-dimensional (2D) database into a three-dimensional (3D) figure, such as a cube. Apply this 3D concept to a database, instead of a flat 2D construct, and now you have an extended 3D figure with the third dimension being represented as various time intervals (Tansel et al., 1993). Date, Darwen, and Lorentzos (2003) provide a detailed investigation into the application of interval and relational theory to the problem of temporal database management.

Jensen, Mark, Roussopoulos, and Sellis (1993) presented an architecture for query processing in the relational model extended with transaction time that integrates standard query optimization and computation techniques with new differential computation techniques. The use of differential computation techniques is essential in order to provide efficient processing of queries that access very large temporal relations.

Temporal database systems are systems that provide special support for storing, querying, and updating historical and future data (Date et al., 2003). A temporal database records time-variant information. Date (2004) states that the relational model needs no extension or corruption in order to support the time dimension. Snodgrass (1998) defines the temporal database as “a database that supports some aspect of time”. Another definition that may be better structured to fit a Temporal Relational Model states that a temporal database is defined as “an union of two sets of relations Rs and R1, where Rs is the set of all static relations and R1 is the set of all time-varying relations” (Navathe & Ahmed, 1993). This article is limited to the consideration of the relational model of the temporal database to the exclusion of the other well-known types of databases such as object-oriented, network, and hierarchical. While there is little temporal database research currently on the latter two types of databases, an increasing amount of research is being done in the area of temporal object-oriented databases. We might also note that temporal databases have also been referred to as time-oriented databases, time-varying databases, or historical databases. While time-oriented database and time-varying database are equivalent in meaning to temporal database, historical database is not. As discussed later in this article, a historical database is actually a subset of the temporal database.
Temporal Databases

The first thing to note about the Snodgrass (1998) definition of a temporal database is that the time dimension requirement for a database to be temporal does not include user-defined time. User-defined time is some aspect of time that is not recognized by the database management system as a special data type. A classical database essentially treats data, such as birthdates, as text strings. This treatment of date data does not allow for much manipulation. One way of attempting to avoid this problem is to treat the date as a type of text data to store the date as a number. This approach, however, has its own problem set. One such problem found in Oracle version 7.2 is that in some instances in year 2000, dates are treated as earlier than dates in the 1900s. Converting dates to numbers internally may be best way of making possible a wide range of ad hoc queries on temporal data. The fact that support of user-defined time does not merit a database being considered temporal and does not mean that the temporal database cannot or will not include user-defined time.

TEMPORAL DATABASE CHARACTERISTICS

There are several types of time in a temporal database. Before describing these different types in detail, it is important to clarify the concept of precision relating to the time stored in a temporal database. This concept denoting precision in a temporal database is called the granularity of time (Howe, 1997). An example of progressive time granularity includes a day, an hour, a second, or a nanosecond.

It is important to understand three major concepts to grasp the precise nature of a temporal database. The first is the concept of valid time. The second concept is transaction time, the actual time recorded in the database at the time the data is entered. Time stamps can include either the date or the date and clock time. An object is an entity that has a well-defined role in the application domain, and its features include state, behavior, and identity. An employee is a good example of an object. In a classical database, once a change is made to an employee’s record, original data is changed, discarded, and replaced by new data. However, in a temporal database, which supports transaction time, transaction time can be attached in the form of a time stamp to both the old data and to the new data for that employee. In so doing, the database can store both the old data and new data for the same object. In this case, the salary of the employee was increased on a certain date. It is important to note here that the transaction time values or time stamps cannot be later than the current point in time nor can they be changed, just as the past cannot be changed.

Another major type of time in a temporal database is termed as valid time. Valid time is the actual or real-world clock time at which the data is valid. Continuing with the employee example, while transaction time is the point in time at which the data is entered into the database, valid time is the unique point when the entered data become true or take effect. For instance, on January 3, an employee is notified that an increase in salary will be effective February 1. The Human Resources department, after being notified of the employee’s raise, must enter the new salary into the database. Presumably, they will enter the data before the raise goes into effect. The actual time they enter the raise into the database is prior to February 1st, and that time will be the time stamp for the transaction time. The data, however, are not yet valid, and for the rest of January, the employee will continue to receive the current salary. However, on February 1, the raise data will become valid. Thus, February 1 is the valid time. Also, if an employee receives a raise that is retroactive, the transaction time may be later than the valid time.

The two major types of time unique to the temporal database, valid time and transaction time, allow for the possibility of three forms of temporal databases: historical, rollback, and bi-temporal (Steiner, 2003). A historical database supports valid time, but not transaction time. To reiterate, a historical database is a poor choice as an alternative term for the temporal database. The reason is that a historical database is but one type of temporal database. A historical database, however, as explained later, would be a poor choice for anyone wishing to deploy a temporal database. The second form of a temporal database is the rollback database. This database is the opposite of the historical database. The rollback database only supports transaction time and not valid time. As opposed to the historical database, rollback database is quite useful for data recovery after database failure. The reason then that a temporal database would rarely be desirable is that it could not support rollback after DBMS failure. It is also necessary if the database does not use the locking technique to ensure data security. Most databases on the market today do support at least some rollback features.

In reality, a temporal database is a bi-temporal database. This database supports both types of time that are necessary for storing and querying time-varying data. The bi-temporal database could aid significantly in knowledge discovery, since it is able to fully support the time dimension on three levels: the DBMS level with transaction time, the data level with valid time, and the user-level with user-defined time.
Related Content

Moving Objects Databases
[www.igi-global.com/chapter/moving-objects-databases/11176?camid=4v1a](www.igi-global.com/chapter/moving-objects-databases/11176?camid=4v1a)

A Case Study of the Military Utility of Telemedicine
[www.igi-global.com/chapter/case-study-military-utility-telemedicine/8284?camid=4v1a](www.igi-global.com/chapter/case-study-military-utility-telemedicine/8284?camid=4v1a)

Research Review: A Systematic Literature Review on the Quality of UML Models
[www.igi-global.com/article/research-review-systematic-literature-review/55133?camid=4v1a](www.igi-global.com/article/research-review-systematic-literature-review/55133?camid=4v1a)

Main Memory Databases
[www.igi-global.com/chapter/main-memory-databases/11170?camid=4v1a](www.igi-global.com/chapter/main-memory-databases/11170?camid=4v1a)