DEVELOPING THE TEMPORAL LIBRARY DATABASE SYSTEM

Aye Aye Khaing
University of Computer Studies (Meiktila), Myanmar
Khainga79@gmail.com

ABSTRACT

A temporal database is formed by compiling, storing temporal data. The difference between temporal data and non-temporal data is that a time period is appended to data expressing when it was valid or stored in the database. The data stored by conventional databases consider data to be valid at present time as in the time instance now. When data in such a database is modified, removed or inserted, the state of the database is overwritten to form a new state. The Temporal Database Management System (TDBMS) is to provide efficient storage of temporal data and support temporal querying over the information. In many Libraries system, there are many books and many students studied of these books. So Librarian was so busy for borrowing of books and recording in manual. Manual system can be so many errors and lost of books. The result is a temporal data model that is generalized, conceptually tractable, and inherently contains a full description of the primary data it organizes. A practical working relational database design based on this general temporal data model is presented and demonstrated.

1. INTRODUCTION

Conventional database management systems (DBMS) are responsible for the storage and processing of huge amounts of information. The data stored by these database systems refers to information valid at present time, valid now. It concerns data that is believed to be true in reality at the present moment. Past data refers to information that was stored in the database at an earlier time, data that is believed to have existed in the past, valid at some time before now.

Most database application manage temporal data, such as time and date of withdrawal of money, closing values of stocks on the stock exchange, or the periods over which employees are associated with projects. Temporal data management applications could thus benefit substantially from built-in support. Temporal databases extend conventional databases by associating timestamps with facts. Implementing a Temporal Database Management System (TDBMS) on top of a conventional DBMS has generally not been pursued because it cannot take advantage of well-known temporal implementations techniques. Further, it seems that there has been an implicit assumption that the performance of temporal DBMSs should be similar to that of conventional DBMSs, even when a temporal DBMS manages multiple versions of data and a conventional DBMS manages only one version.

It will be beneficial to use the library be via the Developing the Temporal Library Database System. It encourages accessing of more books in a short duration. And this system can also provide the time and energy-saving to the librarian and all the library users. It reduces the unnecessary steps for borrowing and returning the book, and upgrading the usual manual procedures of a library system catalogue. Besides, it is both systematic and easy to use.

This paper aims to describe the underlying principles that are related to creating and manipulating temporal databases. The temporal database managed by the TDBMS will be in historical form. The data stored in the temporal database will be with respect to valid time.

2. DATABASE SYSTEMS AND THE DIFFERENT TYPES

A Database Management System (DBMS) is a collection of interrelated data and a set of programs to access data stored. A DBMS that stores data must have a well-defined format to model and store information [4]. The term database has fallen into

loose use lately, losing much of its original meaning. There are four well known modeling techniques to represent data for storage, a relational database, an object-oriented database, a spatial database and a temporal database. Database management systems providing support for the storage and retrieval of time-dependent data are called temporal database management systems and the corresponding databases, temporal databases.

2.1. Relational Database

A relational database stores data in tables, known as relations. Each table consists of rows, known as tuples and columns, known as attributes or fields. A row contains data about a specific entity. Thus, a relational database stores a set of tables. Relational databases have replaced databases built according to earlier models because the relational type has valuable attributes that distinguish relational databases from those other database types.

In a relational data model, tuples are time-stamped and in an object-oriented data model, objects/attributes are timestamped. Each ordinary data has two time values attached to it, a start time and an end time to establish the time interval of the data [8].

2.2. Object-Oriented Database

Object-oriented database stores data about entities in objects. The type of object specifies the properties (attributes) the object contains. Sets of objects of the same type are called collections. Thus, an object-oriented database contains a set of collections [2]. Object oriented features such as inheritance mean that it will be easier and possible to reuse code.

2.3. Spatial Database

Spatial database concerns the storing of data in relation to space. It offers spatial data types and stores information relating to geometric or geographical space, for example, the 2-D abstraction of the Earth's surface or the 3-D space representing the arrangement of chains of protein molecules. Thus, a spatial database stores a collection of space related data. Spatial database is a database that embodies spatial, temporal, and spatiotemporal database concepts, and captures spatial and temporal aspects of data. It is dealing with geometry changing over time.

2.4. Temporal Database

Temporal database stores data relating to time instances. It offers temporal data types and stores information relating to past, present and future time, for example, the history of the stock market or the movement of employees within an organization. Thus, a temporal database stores a collection of time related data [1].

Time in temporal databases can be expressed in many different ways; it may be of interest to record the exact time when an event happened or the period in which an event took place or the duration of an event or even the periodicity of an incident – the frequency with which an incident occurs in time. Time is used to distinguish between past, present or future states. The recording of time allows the identification when facts are true in the modeled reality (valid time) or when facts are current in the database (transaction time).

2.5. Transaction Time

Time is used to distinguish between past, present or future states. The recording of time allows the identification when facts are true in the modeled reality (valid time) or when facts are current in the database (transaction time).

A database object is stored in a database at some point in time. The transaction time of an object is the time when the object is stored in the database, the time that it is present in the database. Transaction time values cannot be after the current time. Transaction time is the time a transaction was made. This enables queries that show the state of the database at a given time [6].

2.6. Valid Time

Valid time is the time for which a fact is true in the real world. The valid time of a database object is the time when the object is effective or holds (is true) in reality. The time when the event occurred, took place in reality. Objects in the temporal

database system will have a time component associated to it; this will hold either the valid time or the transaction time [6].

3. OVERVIEW OF TEMPORAL DATABASE SYSTEM

The roles of the individual components in temporal database system:

- User – interacts with the TDBMS.
- Interface – resides on the top of the Middleware. A friendly graphical user interface that allows the user to make requests to the TDBMS and displays the results. The interface should be simple to use and easy to learn. The user interface is the only way the user can communicate with the temporal database system.
- Middleware – the core engine that contains the functionality of the system, which processes the request from the user via the interface and retrieves information from the underlying conventional relational database.
- Relation Database System – stores temporal data and is configured to behave as a temporal database [6].

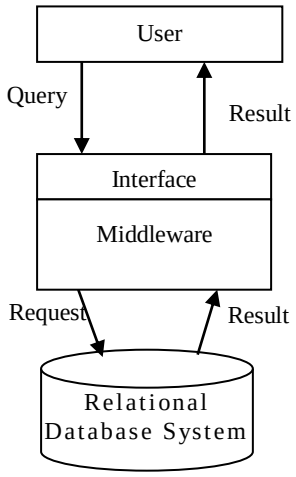

Figure 1. Overview of Temporal Database

4. COALESCE FUNCTION

The aim of the function is to arrange temporal data to abide by the Temporal Normal Form (TNF), remove any violations of the TNF [8]. The following pseudo algorithm defines the coalesce function:

```
function coalesce(R)
R := R order by a1,….,an, start
Rc = empty
Size = count(R)
i = 1
repeat
        Ri = (a1,….,an) [start, end]
        while Ri+1 = (a′1,….,a′n) [start′, end′ ]
and a1= a′1,…., an= a′n
                and start'≤end+1
        end := max(end, end′ )
        i = i+1
end_while
insert (a1,….,an) [start, end] into Rc
i := i+1
while i ≤ size
return Rc
```

5. TEMPORAL LIBRARY DATABASE SYSTEM

In our proposed system, there are two types of data that are used as examples in the thesis. The first one does not contain any temporal data while the second one does. However, the two examples contain similar information with the addition of temporal attributes in the second case.

In our system, the registering process must be done for two conditions: member registration and book registration. There are two types of system users: Administrator and member.

Accessing System by Administrator: The administrator have to login with admin account to manage staff from the system. Admin can add new books or edit book information and member information or delete member or books. Admin can also view books and members detail history at any time as shown in figure 2. This system can add a new member ID and book ID in library.

Accessing System by Member: To access their working history, member needs to login with

his/her account ID and password. After he or she has logged in, member can view their current information of book list and detail information for his or her history borrow list as shown in figure 3.

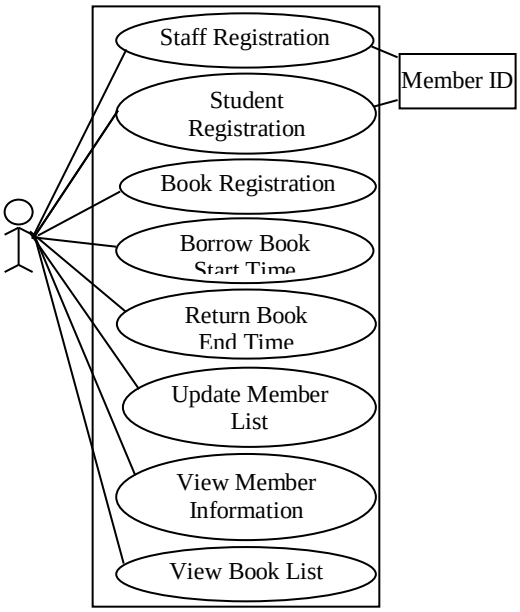

Figure 2. Use Case Diagram for Administrator

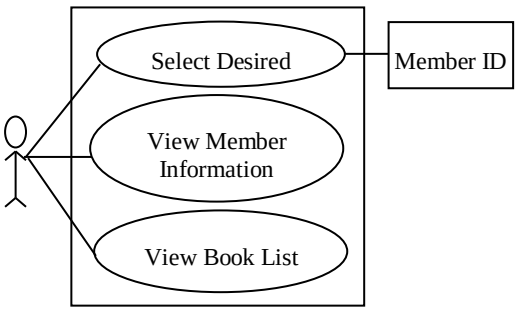

Figure 3. Use Case Diagram for Member

5.1. Processing design of Temporal Library System

In Borrow form, if the user wants to borrow the book which is Present, the user has to choose the Borrow date. In the Remark column there are two signals: "Present" and "Borrow". One can only get the book marked with "Present". When completed, click the OK button. Then the user has to supply in the text boxes of Accession No., Member Name and Member ID. Finally, when all these steps are passed, the user has to click the "Borrow" button.

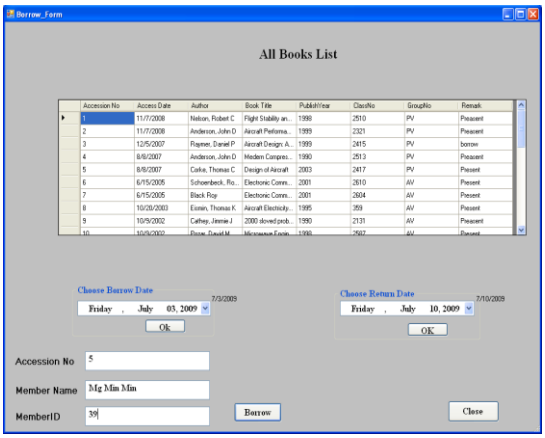

Figure 4. Book Borrowing Process Form

If one would like to send back the book to library, the user has to open the Return Book Form. The user has to select the Return Date and click Return button. Then put in "Enter Accession No." when the user clicks the Return button, the information of the book the user returns will emerge.

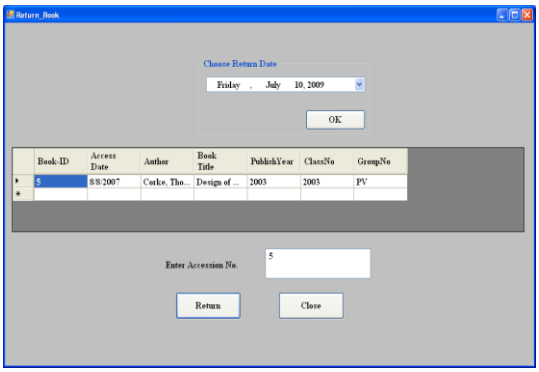

Figure 5. Return Book Form

The figure 6 has two processes for administrator. The first one is to process for member borrowing book. If the user enter the system and the user want to borrow book, user choose desired book. Administrator fills system date (up to date), and inserts the information in the particular form. Then administrator save this borrow list in the Temporal Database.

The second is Return book process. In this process, administrator inserts Book ID and view information. So the information of this book will arrive at the original database automatically.
Administrator also provides the transaction process of book list from the book table of original database to the temporal database. While the member arrive at the system to view the information for borrowing the book, or to enter the system for registering member, the administrator must be done this process with currently.

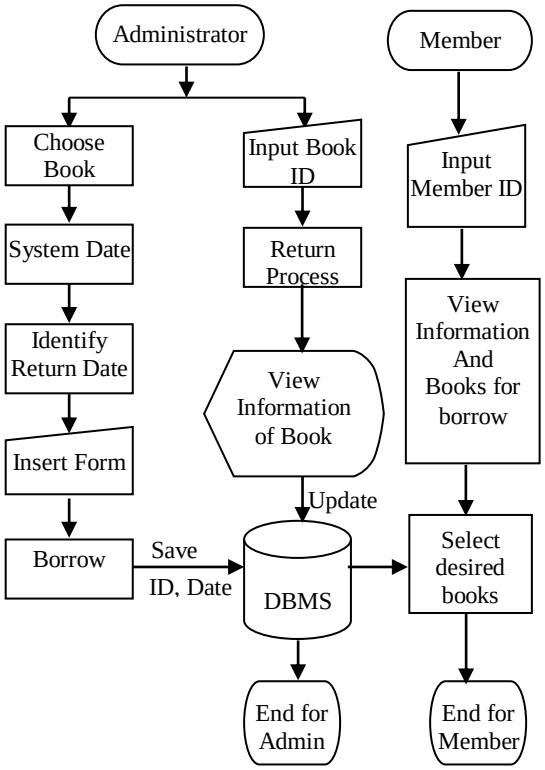

Figure 6. System Flow Diagram

6. EXPERIMENTAL RESULT

This system use the coalesce function because of coalesce function is to sort the results of a query on a relation based on its attributes and merge all tuples that have the same non-temporal attributes and overlapping intervals.

This system can provide accurately and erroneously for library system. The aim of this system present an overview of the techniques proposed to date that deal specifically with temporal data model. The overall aim of the thesis is to create a Temporal Database Management System (TDBMS).

This system helps to provide for librarian and also provide for finding books in library. Often we are not only interested in the actual information but also a trace of previous information.

With the general goal of providing built-in support for time-varying data without having to construct a temporal DBMS from scratch, we explore in this system how a temporal DBMS can be implemented in a stratum on top of an existing, conventional DBMS.

The idea is to reuse the functionality of existing DBMS technology. This system shows one approach to managing temporal data, along with the corresponding prototype implementations. This system presents a tested data of library more than 2000 and transaction event in daily.

7. CONCLUSION

Although research on temporal database systems has been active for about twenty years, implementations have not appeared until recently. This is one reason why current commercial database systems provide only limited temporal functionality. The temporal data and relations stored by the database will be according to a linear discrete bounded data model and will reside on a conventional relational database. Building a temporal DBMS from scratch is a daunting task, which may only be successfully taken on by the major DBMS vendors.

The fundamental components of a temporal system are identified and both they and their relationships to each other are given simple, standardized definitions.

The system mainly focuses on and concerns the extraction and manipulation of temporal data. In this system, we provide a survey of temporal database techniques. Indeed, library is vitally important to all learners, not only the students but also to everyone.

This system can be used to retrieve the data by querying of library data. There are several observations which can be updated in such system. Indeed, this system has been based on the relation of the tables with one another.

Using the Temporal Database, library data are managed according to "Date", so that other parameters can be used. Temporal Database can also be built via the Object-oriented database. Temporal Database system can be set up for the applications which have transaction.

7.1. Advantages of the System

The system is flexible and portable. Flexibility in terms of the tool being independent to the underlying relational database system used to hold temporal relations. The reason being is that all relations are retrieved from the underlying physical database and then processed.

The system gave a more thorough understanding of temporal databases. Due to the ubiquity of time and its importance to most database management applications, in general, the users can easily understand the structure of a temporal database which consists only of tuple timestamping relations and express queries using this approach. The system will support for librarian and their organization more effectively.

REFERENCES

[1] A.Tansel, J.Clifford, S.Jajodia, A.Segev and R.T.Snodgrass, Temporal Databases: Theory Design and Implementation.

[2] Ariav G. A Temporal Oriented Data Model. ACM Transaction on Database Systems, 11(4), 499-527 (1986).

[3] Bettini C., Jajodia S. and Wang S. Time Granularities in Databases, DataMining, and Temporal Reasoning. Berlin: Springer-Verlag (2000).

[4] C.J.Date. An Introduction to Database system

[5] G.Slivinskas, C.S.Jensen and R.T.Snodgrass, Adapting Query Optimisation and Evaluation in Temporal Middleware.

[6] G.Slivinskas, C.S.Jensen and R.T.Snodgrass, Query Plans for Conventional and Temporal Queries Involving Duplicates and Ordering.

[7] P.J.McBrien and D.Gabbay, Temporal Logic & Historical Databases

[8] P.J.McBrien, Principles of Implementing Historical Databases in RDBMS.