Modeling and Querying Temporal Data by using XML Database

Lily Ning Ngaih Zam, Kyaw Zar Zar Phyu
University of Computer Studies, Yangon
nnzam88@gmail.com,kyawzarzarphyu@gmail.com

Abstract

Database management Systems support to model reality, temporal database has the capability to record and process time-varying aspect of the real world. Conventional database represents the state of an enterprise at a single moment of time and changes are viewed as modification to the state. Many applications need information about the past: such as Financial records, Medical information, Library information system, and etc. Temporal Database System is a system that manages time varying data and temporal database contains historical data as well as current data. XML database offers more benefits than traditional relational database in temporal information processing because XML is well suited to model temporal data and XQuery have many feature that facilitate the complex querying for temporal data. XML with XQuery is used to model temporal data to eliminate the inflexibility of the relational data model and the extensible of SQL. In this paper, Library System is implemented to manage temporal data and querying temporal query for library system.

Keywords: Temporal database, XML, XML Database, XQuery

1. Introduction

Time is an important aspect of all real world phenomena. Events occur at specific point in time; objects and relationship among objects exists over time. The ability to model this temporal dimension of the real world and to respond within time constraints to changes in the real world as well as to application-dependent operation is essential to many computer applications such as accounting, banking, economics, geographical information systems, inventory control, law, medical records, multimedia, process control, reservation system, and scientific data analysis.

Temporal database system maintains historical data records as well as changes are inserted as additions to the information stored in the database. Therefore, it can be efficient access to past states. Temporal data models is the extension of relational model by adding temporal attributes to each relation and reality through the use of two orthogonal time dimensions: the valid and transaction time. Depending on which time dimension is supported, a temporal database is characterized as transaction time, valid time or bi-temporal.

The classical database is generally two dimensional, and contains only current data. Whereas Temporal databases are three-dimensional with time interval as the third dimension. Temporal databases can also be referred to as time-oriented databases, time-varying databases, or historical databases [1]. A true temporal database is a bi-temporal database that supports both valid time and transaction time. More specially, temporal aspects usually include valid time and transaction time. Valid time is the actual or real world time at which point the data is valid. Transaction time is the actual time recorded in the database at which the data is entered.

XML is also a new database model serving as a powerful tool for approaching semi structured data. Much research work has recently focused on adding temporal features to XML, i.e. to take into account change, versioning, evolution and also explicit temporal aspects of XML data, like, e.g., the problem of representing historical information in XML. XML documents are related to time in two aspects: they contain temporal information and/or their contents evolve with time. Technically, to develop an XML temporal data model, it is necessary to extend an XML data model by a time dimension. This paper represents a temporal data model on XML and querying temporal information for a library system.

2. Related Work

Time in XML Some interesting research work has recently focused on the problem of representing historical information in XML. An annotation-based object model is proposed to manage historical semistructured data [1], and a special Chorel language is used to query changes. A new <valid> markup tag for XML/HTML documents is proposed to support valid time on the Web, thus temporal visualization can be implemented on web browsers with XSL [2]. The author uses valid tag embodied in XML with the help of XSL infrastructure to enable navigation and browsing of museum web document in a temporal manner. A dimension-based method is proposed to manage changes in XML document is presented in [3], however how to support queries is not discussed. They used additional attributed in XML to track changes in XML document such as update, delete and insert operation on XML document. A data model is proposed for temporal XML documents in [4]. However, since a valid interval is represented as a mixed string, queries have to be supported by extending DOM APIs or XPath. Various temporal representations for XML are discussed in [5]. The author compares various representations for temporal XML document which use schema and tools.

3. Database, XML Database and Temporal Database

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. Future data refers to information considered to be valid at a future time instance, data that will be true in the near future, valid at some time after now.

3.1 Different Types of Databases

- *Relational database* stores data in tables, known as relations. Each table consists of rows, known as tuples and columns, known as attributes or fields.
- *Object-oriented database* stores data about entities in objects.
- *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.
- *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.
- *Native XML database* generally refers to a database that is designed for XML content from the ground up, as opposed to a traditionally relational database.

3.2 Temporal Database

A wide range of database applications manage time varying data. In contrast, existing database technology provides little support for managing such data. The research area of temporal databases aims to change this state of affairs by characterizing the semantics of temporal data and providing expressive and efficient ways to model, store, and query temporal data. 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.

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. For example, in banking system, the transaction time of a withdrawal would be form the time the clerk entered the payment of withdrawal into the database to the time that it was made invalid in the database. Transaction time values cannot be after the current time.

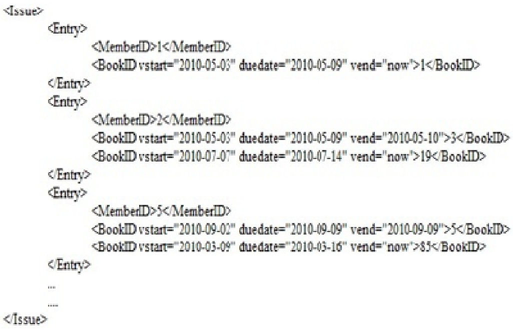
The valid time of a database object is the time when the object is effective or holds (is true) in reality. The time when the event occurred, took place in reality. For example, in a banking system, the payments and withdrawals made by a customer have a valid time associated with the time the customer performs the transaction at the bank. 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]

4. System Overview

This section describes the temporal data model used by the system and example temporal queries for some operations in the system.

4.1 Temporal XML

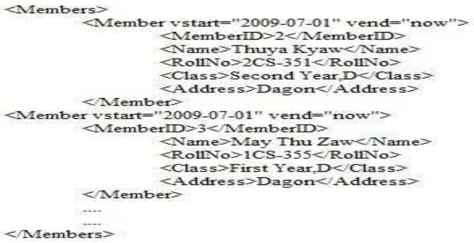
The system uses vstart and vend for temporal storage in XML document, time information is record as attributed on the XML tags. There are four XML for our library system, Users.xml store user control, Catalog.xml stores book related information, Issue.xml store book borrowed information and Member.xml stored member information of our system. This system used Qizx [7] open source xml native database.



```
<Issue>
        <Entry>
                <MemberID>1</MemberID>
                <BookID vstart="2010-05-03" duedate="2010-05-09" vend="now">1</BookID>
        </Entry>
        <Entry>
                <MemberID>2</MemberID>
                <BookID vstart="2010-05-03" duedate="2010-05-09" vend="2010-05-10">3</BookID>
                <BookID vstart="2010-07-07" duedate="2010-07-14" vend="now">19</BookID>
        </Entry>
        <Entry>
                <MemberID>5</MemberID>
                <BookID vstart="2010-09-02" duedate="2010-09-09" vend="2010-09-09">5</BookID>
                <BookID vstart="2010-03-09" duedate="2010-03-16" vend="now">85</BookID>
        </Entry>
        ...
        ....
</Issue>
```

Figure 4.1 Issue.xml

The literal "now" represent present time. For each borrow operation system record <Entry> tags in Issue.xml. The system record BookID and Member ID and time of book borrowed in vstart. When the book is returned the corresponding vend tag of the <Entry> tag is modified to reflect change. All of the above operations are performed by using XQuery statement. When a member already borrowed book in the system, the system need not to add <Entry> tag, instead it append a <BookID> tag with corresponding vstart and vend for borrowing a book.



```
<Members>
            <Member vstart="2009-07-01" vend="now">
                        <MemberID>2</MemberID>
                        <Name>Thuya Kyaw</Name>
                        <RollNo>2CS-351</RollNo>
                        <Class>Second Year,D</Class>
                        <Address>Dagon</Address>
            </Member>
<Member vstart="2009-07-01" vend="now">
            <MemberID>3</MemberID>
                        <Name>May Thu Zaw</Name>
                        <RollNo>1CS-355</RollNo>
                        <Class>First Year,D</Class>
                        <Address>Dagon</Address>
            </Member>
            ----
            ----
</Members>
```

Figure 4.2 Members.xml

Members.xml records information about member and times at which members are created and other general information.

4.2 System Design

There are two users in the system, user and administrator, users are student and administrator is the librarian. User can only perform query operation and system administrator performs setup such as book, member information, book borrows and returns process and perform temporal query operation on the library system. Librarian can manage or add book into the system, the system perform this by issuing XQuey insert node statement. Figure 4.3 shows the flows diagram of the system administrator. Administrator can manage and query operation on book such as adding, borrowing book, and can perform various temporal query operations. Figure 4.4 is the flow diagram of the user. It states that user perform operation such as returning book and perform temporal query operation on the library. In order to use the system, the user must first logged into the system as user, specify the required operation via the GUI of the system, the system construct the temporal query according to user operation, executes the query and will display the result to the user.

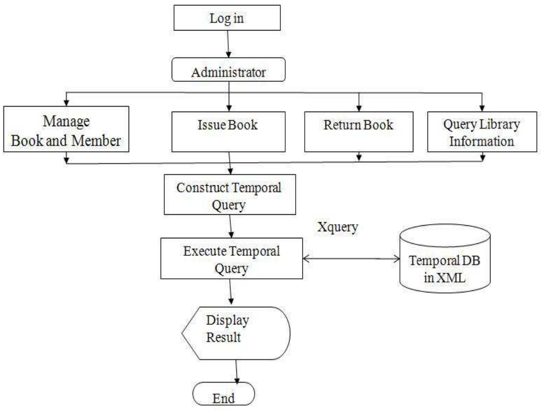


Figure 4.3 System flow for Administrator

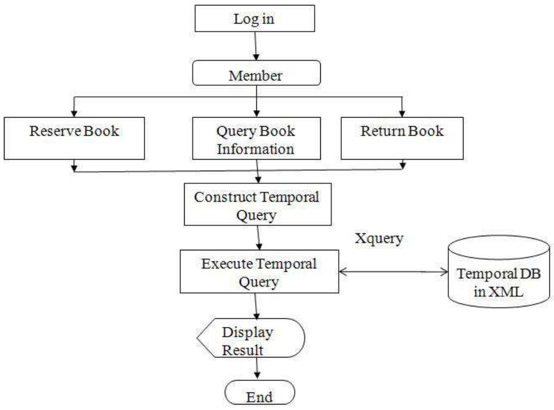


Figure 4.4 System flow for user

4.3 Example General Queries

This section describes the example for general queries used in the system. Temporal query operations are expressed in the XQuery statement that can be executed by the XML native database, performing the temporal query operation over XML database.

4.3.1 Add a user

The following XQuery is send to the XML database to add a user.

```
for $a in doc('Users.xml')/Users
return insert node
<User><Name>usertwo</Name><Password>usert
wo</Password><Right>student</Right></User>
as last into $a;
```

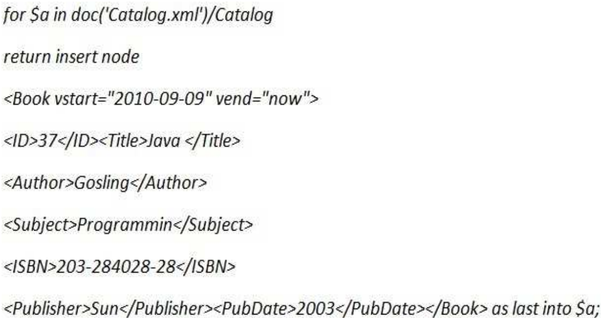
4.3.2 Deleting a user

The following XQuery is send to the XML database to delete a user whose name is ‘UserOne’.

```
for $a in doc ('Users.xml' ) /Users/User[Name='
UserOne']return delete node $a;
```

4.3.3 Adding a book

The following XQuery is send to the XML database to add a book to the catalog.



```
for $a in doc('Catalog.xml')/Catalog
return insert node
<Book vstart="2010-09-09" vend="now">
<ID>37</ID><Title>Java </Title>
<Author>Gosling</Author>
<Subject>Programmin</Subject>
<ISBN>203-284028-28</ISBN>
<Publisher>Sun</Publisher><PubDate>2003</PubDate></Book> as last into $a;
```

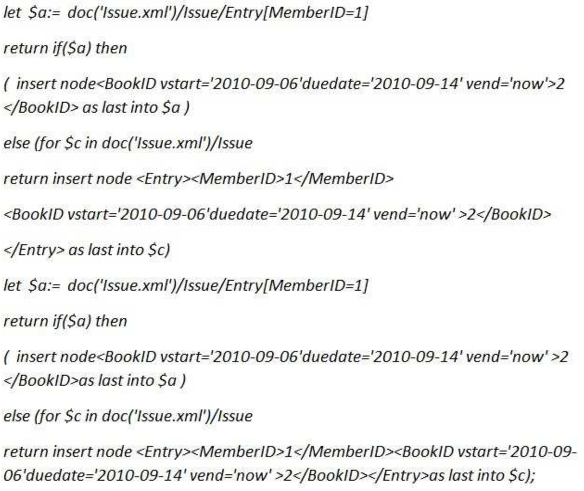
4.3.3 Deleting a book

The following XQuery is send to the XML database to delete a book in the catalog with ID=37.

```
for $a in doc ('Users.xml' ) /Users/User[Name='
UserOne']return delete node $a;
```

4.3.4 Borrowing a book

The following XQuery is used to borrow book id 2 by member 1.



```
let  $a:=  doc('Issue.xml')/Issue/Entry[MemberID=1]
return if($a) then
(  insert node<BookID vstart='2010-09-06'duedate='2010-09-14' vend='now'>2
</BookID> as last into $a )
else (for $c in doc('Issue.xml')/Issue
return insert node <Entry><MemberID>1</MemberID>
<BookID vstart='2010-09-06'duedate='2010-09-14' vend='now' >2</BookID>
</Entry> as last into $c)
let  $a:=  doc('Issue.xml')/Issue/Entry[MemberID=1]
return if($a) then
(  insert node<BookID vstart='2010-09-06'duedate='2010-09-14' vend='now' >2
</BookID>as last into $a )
else (for $c in doc('Issue.xml')/Issue
return insert node <Entry><MemberID>1</MemberID><BookID vstart='2010-09-
06'duedate='2010-09-14' vend='now' >2</BookID></Entry>as last into $c);
```

4.3.4 Returning a book

The following XQuery is used to return book id 2 by member 1. It modifies Issue.xml so that the corresponding vend of the returned book is set to return date in this case it is *'2010-09-23'*.

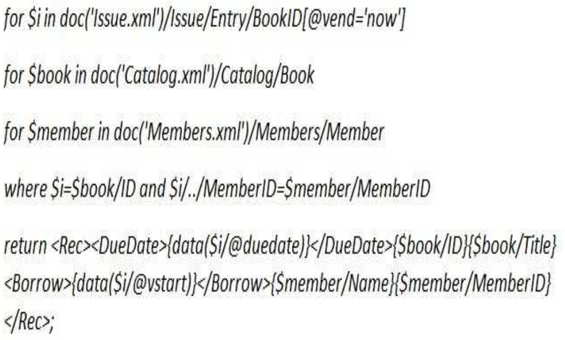
```
for $a in doc('Issue.xml')/Issue/Entry[MemberID=1]
for $b in $a/BookID
where data($b)=2  and $b/@vend='now'
return replace value of node $b/@vend with '2010-
09-23';
```

4.4 Example Temporal Queries

Temporal operator can be easily implemented in XQuery because it support many built in function for date and time manipulation, user can write user define XQuery function.

4.4.1 Temporal Join

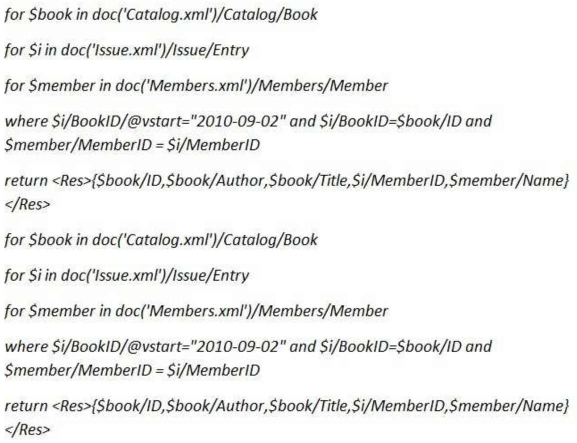
The following query is used to retrieve all currently un-return books in the system. This query used join operation on Issue.xml, Catalog.xml and Members.xml by using for statement and specifying the required condition for join in where clause.



```
for $i in doc('Issue.xml')/Issue/Entry/BookID[@vend='now']
for $book in doc('Catalog.xml')/Catalog/Book
for $member in doc('Members.xml')/Members/Member
where $i=$book/ID and $i/../MemberID=$member/MemberID
return <Rec><DueDate>{data($i/@duedate)}</DueDate>{$book/ID}{$book/Title}
<Borrow>{data($i/@vstart)}</Borrow>{$member/Name}{$member/MemberID}
</Rec>;
```

4.4.2 Temporal Snapshot

Temporal snapshot is used to retrieve the information at a given time period. The following query shows the book borrowed at date *2010-09-02* by applying temporal snapshot. Temporal snapshot can be specified by condition that states vstart must be equal to given date in where clause of the query.

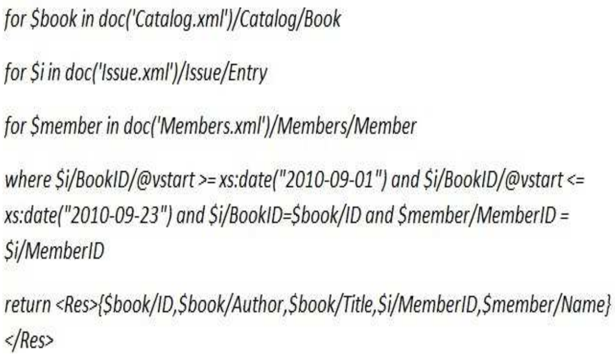


```
for $book in doc('Catalog.xml')/Catalog/Book
for $i in doc('Issue.xml')/Issue/Entry
for $member in doc('Members.xml')/Members/Member
where $i/BookID/@vstart="2010-09-02" and $i/BookID=$book/ID and
$member/MemberID = $i/MemberID
return <Res>{$book/ID,$book/Author,$book/Title,$i/MemberID,$member/Name}
</Res>
```

```
for $book in doc('Catalog.xml')/Catalog/Book
for $i in doc('Issue.xml')/Issue/Entry
for $member in doc('Members.xml')/Members/Member
where $i/BookID/@vstart="2010-09-02" and $i/BookID=$book/ID and
$member/MemberID = $i/MemberID
return <Res>{$book/ID,$book/Author,$book/Title,$i/MemberID,$member/Name}
</Res>
```

4.4.3 Temporal From To

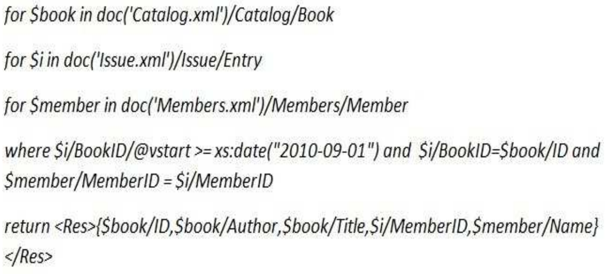
The following query used temporal from to query to show the book borrow from *2010-09-01* to *2010-09-23*. Temporal from to query can be accomplished by specifying the time period condition in the where clause of the query.



```
for $book in doc('Catalog.xml')/Catalog/Book
for $i in doc('Issue.xml')/Issue/Entry
for $member in doc('Members.xml')/Members/Member
where $i/BookID/@vstart >= xs:date("2010-09-01") and $i/BookID/@vstart <=
xs:date("2010-09-23") and $i/BookID=$book/ID and $member/MemberID =
$i/MemberID
return <Res>{$book/ID,$book/Author,$book/Title,$i/MemberID,$member/Name}
</Res>
```

4.4.5 Temporal since

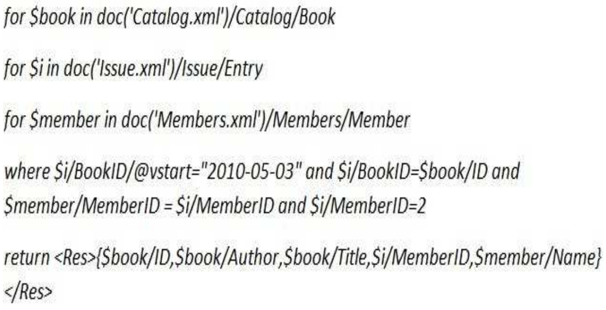
Book borrowed from *2010-09-01*.



```
for $book in doc('Catalog.xml')/Catalog/Book
for $i in doc('Issue.xml')/Issue/Entry
for $member in doc('Members.xml')/Members/Member
where $i/BookID/@vstart >= xs:date("2010-09-01") and  $i/BookID=$book/ID and
$member/MemberID = $i/MemberID
return <Res>{$book/ID,$book/Author,$book/Title,$i/MemberID,$member/Name}
</Res>
```

4.4.6 Temporal snapshot by Member

Book borrowed at *2010-05-03* by Member2.



```
for $book in doc('Catalog.xml')/Catalog/Book
for $i in doc('Issue.xml')/Issue/Entry
for $member in doc('Members.xml')/Members/Member
where $i/BookID/@vstart="2010-05-03" and $i/BookID=$book/ID and
$member/MemberID = $i/MemberID and $i/MemberID=2
return <Res>{$book/ID,$book/Author,$book/Title,$i/MemberID,$member/Name}
</Res>
```

5. Conclusion

Using temporal database, the system can stores all information, about past, current or even future behavior of the enterprise. In temporal database, the system updates that happen in increasing transaction time order. Therefore, the consequence of temporal database is that it can be used to store historical and current information about library information system. The key advantages of temporal database using XML is that powerful temporal queries can be expressed in Xquery without requiring the introduction of new constructs in language. The transaction time histories of relational database can be stored and queried efficiently by using XML (i) to provide temporally grouped representations of such histories, and (ii) to implement temporal queries expressed in Xquery against these representations. This system provides the user of library system to store and process temporal information for library system.

5. References

1. C.E. Dyreson. “Observing Transaction-Time Semantics with TTXPath”, WISE, 2001.
2. F. Grandi and F. Mandreoli. “The Valid Web: An XML/XSL Infrastructure for Temporal Management of Web Documents”, ADVIS, 2000.
3. M. Gergatsoulis and Y. Stavrakas. “Representing Changes in XML Documents using Dimensions”, Xsym, 2003.
4. K Ali, J Pokorny, "XML-based Temporal Models",Technical Report, Czech Technical University, June 2006..
5. Stephen W. Thomas, "Implementation and Evaluation of Temporal Representation in XML", Master Thesis, University of Arizona, March 2009.
6. Patel, J., “Temporal Database System”. Department of Computing, Imperial College, University of London, 18 June 2003.
7. http://www.xmlmind.com/qizx/