International Electronic Journal of Pure and Applied Mathematics

Volume 9 No. 3 2015, 205-214
ISSN: 1314-0744
url: http://www.e.ijpam.eu
doi: http://dx.doi.org/10.12732/iejpam.v9i3.10

HOTEL RESERVATION DATABASE TEMPORAL MODEL

Dimitar Pilev

University of Chemical Technology and Metallurgy
8 Kl. Ohridski, 1756 Sofia, BULGARIA

Abstract: Series of areas exist, for which it is necessary to observe change in data over time. In this case it is necessary to store not only store current values of processed data in databases, but also their past or even future values. This article proposes database temporal model for the purposes of hotel reservation systems. Algorithms have been presented for each of the three forms of updating the data – adding, deleting and modifying of data in the database. The model allows the overall functionality, related to the storage, updating and manipulating data in the system, to be completed from the DB side.

Key Words: temporal database, database model, valid time, transaction time, hotel reservation system

1. Introduction

A series of areas exist, in which it is necessary to follow the change of processed data over time. In this case, it is necessary to store not only current values of processed data, but also their past and even future values in DB. Examples for such areas are financial applications, insurance applications, reservation systems, medical systems, decision-making systems, for prognosticating and planning, etc.

The constantly increasing information access requirements of users have necessitated the use of temporal databases (DB). In recent decades, series of temporal DB models have been developed, which utilized different approaches in modeling temporal data characteristics [1], [2]. They differ in the type of time stored in DB, timestamp methods and the timestamp elements used.

The time in which data has been evaluated as true in the modeled reality is named valid time [1]. Databases which store data and data valid time are named

Received: April 15, 2015

International Electronic Journal of Pure and Applied Mathematics – IEJPAM, Volume 9, No. 3 (2015)

historical DB[1]. The time during which the data has been stored in DB is named transaction time [1]. Databases, in which data and transaction time are stored, are named rollback DB [1],[2],[3]. DB, in which both data and their valid and transaction time, are named bitemporal DB [1],[2],[4].

Despite the great diversity of proposed time models, unresolved issues still exist, related to the algorithms for processing data stored in the database [5]. The idea is that the whole data processing is to be completed by the DB management system (DBMS), and not by the completed application. This on its own requires using a temporal DB model, compliant with the specifics of the modeled application environment.

The purpose of this development is to create a temporal DB model, which shall be used for the purposes of hotel reservation systems. According to the specifics of the application environment, the temporal DB model shall consider the following features – the request submitted to DB is fully dependant on the time of its submission; DB can only store facts, which are valid or will be valid as of the time of their registration; only facts which are valid or will be valid at the time of their registration could be stored; only facts valid in the future could be modified; access to current and past states of the database.

2. Motivation Example

When using historical temporal model of DB in applications of hotel reservation systems, only information regarding the time, during which a room has been reserved, shall be stored. For this type of systems, however, it is necessary to follow the time, during which the reservation has been made or cancelled. On the other hand, the transaction temporal DB model allows only the reporting of the time, during which the data has been registered in DB, but not the time, during which a room has been reserved. The nature of both models does not allow their use in hotel reservation systems. For the regular functioning of the two systems, it shall be necessary to track only time, during which data have been recorded in DB, but not the time, for which a given room has been reserved. The nature of the two models does not allow their use in the hotel reservation systems. For the regular functioning of these systems, follow-up of both times shall be necessary – the time when a reservation has been completed, and the time for which information for the respective reservation is stored in DB. In order to follow both times, it shall be necessary to use bitemporal time model of DB, reporting the valid and transaction time (Table 1).

Table 1 Bitemporal relation, containing hotel reservations sample data

Table 1 presents bitemporal relation, which stores data for completed hotel reservations. The valid time shall be used for reporting the reservation period, and the transaction time – the time when data is stored in the DB. What is typical for the bitemporal model of DB is that the existence of two identical tuples is not assumed

International Electronic Journal of Pure and Applied Mathematics – IEJPAM, Volume 9, No. 3 (2015)

clientID	roomID	Vs	Ve	Ts	Te
3	1	5	9	3	∞
2	3	4	12	4	∞
4	1	4	7	4	∞
5	1	12	18	5	∞

Typical for the DB bitemporal model is that the relation does not allow the existence of two identical tuples, marked with fully or partially overlapping temporal periods of valid time. As seen, this condition has been fulfilled in table 1. On the other hand it is clearly visible that a room with identification number 1 (roomID), lines 1 and 3 of Table 1 is occupied for overlapping time periods. This option, which is allowed by the bitemporal model, could not be allowed, and because of this, the typical functionality for the hotel reservation systems shall be realized by their servicing applications.

This work proposes temporal DB model for the purposes of hotel reservation systems. Under the proposed model, the overall functionality related to the storage, updating and manipulation of data in the system shall be completed by the DB.

3. Development of Temporal Database Model for Hotel Reservations

The hotel reservations temporal model (HRTM) stores both the time (V), for which the reservation has been completed and the time in which the information has been stored in the DB (T).

In HRTM, all data in the tuple is marked with two times – valid time and transaction time. Valid time is used to mark the time during which a room is reserved, while transaction time is used to designate the time, in which the tuple has been stored in DB. Using both times, the time period for which a room has been reserved, as well as the time in which the reservation has been completed or cancelled (in case of reservation cancellation) could easily be determined. The relation is always in the first normal form (1NF). The Relation schema is formed by adding four mandatory attributes to the standard non-temporal attributes (passive and active), which describe the subject properties: Vs – beginning of valid period, V – end of valid period, s – beginning of transaction period, and V – end of transaction period. Both periods – the valid period v_e = and the transaction period t_e = have closed bottom and open top limit.

Relation schema for HRTM is as follows:

$$R = (A_1, A_2, \ldots, A_n, B_1, B_2, \ldots, B_m, V_s, V_e, T_s, T_e)$$

Definition 1. The valid time for HRTM could only have current and/or future values.

International Electronic Journal of Pure and Applied Mathematics – IEJPAM, Volume 9, No. 3 (2015)

$V_s \geq CT$, where CT marks the current moment in time

Definition 2. Passive attributes $(a_1 a_2, \ldots, a_n$. These are attributes, which are not dependent on the valid time V that the tuple has been marked with.

Definition 3. Active attributes $(b_1 b_2, \ldots, b_m)$. These are attributes, which are dependent on the valid time V that the tuple has been marked with.

We can view the relation as consisting of two types of tuples – active and valid.

Definition 4. Active tuples are the tuples, for which$CT < T_e \wedge CT < V_e$. All active tuples determine the active state of the relation. These are tuples with a non-expired period for the reservation completed.

Definition 5. Valid tuples are the tuples, for which $CT < T_e \wedge V_e < CT$. The valid tuples store archive information for the reservations completed.

Definition 6. Deleted tuples are the tuples, for which $T_e < CT$. These tuples store history of cancelled reservations.

In table 2, a relation Reservations has been presented, which contains sample data for completed hotel reservations. The attribute clientID is a passive attribute and contains information regarding the user, which has completed the reservation. The reserved hotel room has been assigned using the active attribute roomID. As of day 25-th of the month, one deleted tuple is stored in the temporal relation (line 8), four active tuples (lines 11, 13, 15 and 16), as well as eleven valid (lines 1, 2, 3, 4, 5, 6, 7, 9, 10, 12 and 14).

Table 2 Relation Reservations, containing sample data for completed hotel reservations

clientID	**roomID**	**Vs**	**Ve**	**Ts**	**Te**
3	1	5	9	3	∞
2	3	4	12	4	∞
4	1	4	5	4	∞
5	1	12	18	5	∞
2	3	15	19	7	∞
25	5	11	15	9	∞
3	1	19	25	9	∞
15	3	20	27	10	12
15	3	20	22	12	∞
21	3	13	15	13	∞
21	3	24	31	13	∞
45	5	17	23	15	∞
33	1	25	27	15	∞
3	3	22	24	20	∞
5	1	28	31	21	∞
14	5	25	30	24	∞

International Electronic Journal of Pure and Applied Mathematics – IEJPAM, Volume 9, No. 3 (2015)

Figure 1 shows intervals of reservation completed for the databases stored in table 2. The X-axis shows the time for which the hotel rooms have been booked, and the Y-axis – the time of realization of the reservation. In this case, the granularity used is one day, but another one could be used, such as for example a minute or a second.

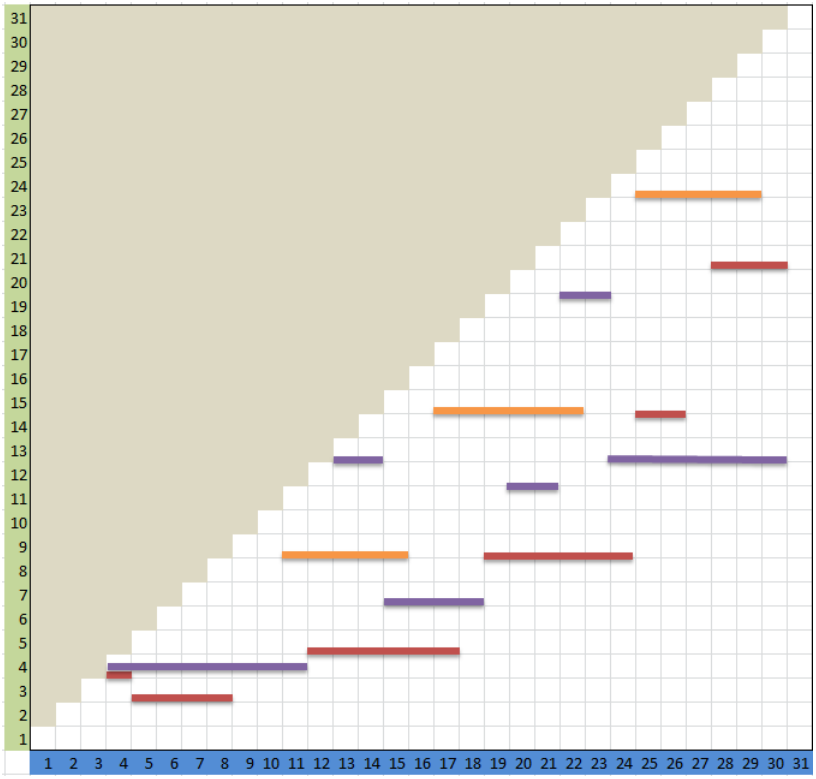

Figure 1: Hotel reservations intervals with HRTM

As noticeable from Figure 1, only reservations, during which the current time moment (CT) is lower than or equal to the beginning of the reservation periods (Vs) could be completed in HRTM. This is one of the main requirements for the hotel reservation systems, for which completing reservation for a past period of time is not allowed.

By using the valid and transaction time, with HRTM it is easy to follow both the history of the reservations completed, as well as the reservations cancelled. This in turn provides an opportunity for easy and quick generation of reports for past and expected loads, which significantly facilitates the hotel management personnel.

HRTM does not allow the existence of two or more tuples with identical active attributes, marked with fully or partially overlapping time periods of valid time. Adding data to the relation is completed when we would like to enter into DB facts referring to a new reservation, valid for a specific period of time $(a_1, a_2, \ldots, a_n)(b_1,$

International Electronic Journal of Pure and Applied Mathematics – IEJPAM, Volume 9, No. 3 (2015)

$b_2, \ldots, b_m$). The entry of data shall return a new updated version of the relation.

3.1. Algorithms for Adding Data to HRTM

Assume that v and t represent the valid and the transaction period of time, with which the values of the passive and active attributes entered in the DB are associated $(a_1, a_2, \ldots, a_n)\,(b_1, b_2, \ldots, b_m)$, $v = \quad t =$. $v^{'}$ designates the new temporal period, marking the same values of the active attributes $(b_1, b_2, \ldots, b_m)$, which shall be entered in DB, $\overset{V}{v^{'}} = .$

Figure 2 indicates the possible situations when adding new data to the relation.

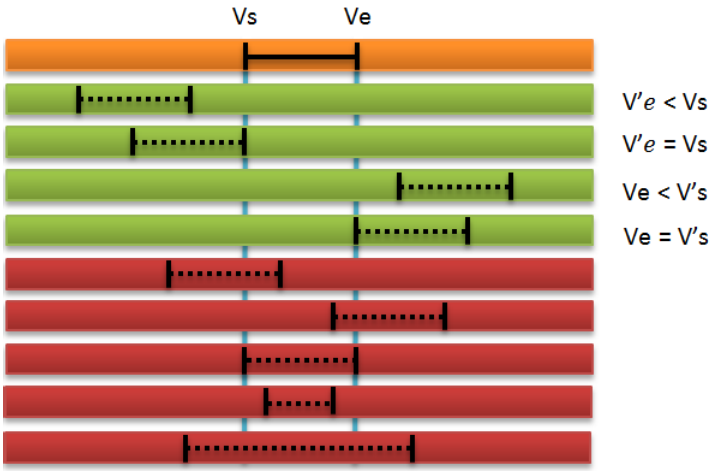

Figure 2: Possible situations adding new data to the relation with HRTM

Two cases of adding new data to the relation exist. Both cases are valid in case the tuple is active.

First case: If the temporal period v, marking the values of the active attributes, stored in the relation $(b_1, b_2, \ldots, b_m)$, fully or partially overlap with the new $v^{'}$ - adding new data to the relation is not allowed (Figure 2, line 1 in case of full overlap, lines 6-10 for partial overlap). In this case the hotel room is already reserved, and it could not be let in any possible manner.

Second case: If the temporal period v, marking the values of the active attributes, recorded in the relation $(b_1, b_2, \ldots, b_m)$, is not overlapped with the new $v^{'}$ - we add a new tuple to the relation $(a_1, a_2, \ldots, a_n)\,(b_1, b_2, \ldots, b_m)$, $v_s^{'}$, $v_s^{'}CT, \infty$ (Figure 2, line 2-5). In this case, the hotel room could be let.

In order to realize the data adding algorithm, overlap function has been defined:

International Electronic Journal of Pure and Applied Mathematics – IEJPAM, Volume 9, No. 3 (2015)

```
overlap(v_s, v_e, v'_s, v'_e):
    return true, if (v_s ≤ v'_s and v'_s < v_e) or (v_s < v'_e and v'_e ≤ v_e) or
                                                   (v'_s < v_s and v_e < v'_e)
    rerurn false, else
```

The relation r, which is updated, the values of the passive and the active attributes $(a_1, a_2, \ldots, a_n)\,(b_1, b_2, \ldots, b_m)$, and their marking time period $v^{'}$ shall be submitted as input parameters of the algorithm for adding data to the relation with HRTM.

```
insert(r, (a_1, a_2, ..., a_n), (b_1, b_2, ..., b_m), V'_s, V'_e):
CT ← current_time;
existTouple ← 0;
t_e = '9999 − 12 − 31';
if CT ≤ V'_s and V'_s < V'_e
    for each x ∈ r
        if CT < x[T_e] and CT < x[V_e]
            if x[B] = (b_1, b_2, ..., b_m) and overlap(x[V_s], x[V_e], V'_s, V'_e)
                existTouple ← 1 ;
                break;
    if existTouple = 0
        y[A] ← (a_1, a_2, ..., a_n);
        y[B] ← (b_1, b_2, ..., b_m);
        y[V_s] ← V'_s;
        y[V_e] ← V'_e;
        y[T_s] ← CT;
        y[T_e] ← t_e;
        r ← r ∪ {y};
return r;
```

This algorithm could be described in the following manner:

1. If the value assigned to $v^{'}$ satisfies the requirements of the model ($CT \leq V_s^{'} \wedge V_s^{'} < V_e^{'}$), proceed to item 2, otherwise completion of the algorithm is terminated. Proceed to item 5.

2. Determine the current time CT of the request for adding the new tuple to the relation.

3. If an active tuple with attributes$(b_1, b_2, \ldots, b_m)$, exists in the relation, whereas both intervals v and $v^{'}$ overlap fully or partially, proceed to item 5. Otherwise proceed to item 4.

International Electronic Journal of Pure and Applied Mathematics – IEJPAM, Volume 9, No. 3 (2015)

4. Tuple with attributes $(a_1, a_2, \ldots, a_n)\,(b_1, b_2, \ldots, b_m)$ shall be added to relation r, marked with time periods of valid time $\overset{V}{v'} =$ and transaction time $t =$.

   Proceed to item 5.

5. End of algorithm.

As a result of completion of the algorithm, new updated version of the relation r is returned.

3.2. Algorithm for Deleting Data with HRTM

Deleting tuples in the relation with HRTM is logical. Most often it is completed in case of cancellation of reservations. A given tuple $(a_1, a_2, \ldots, a_n)\,(b_1, b_2, \ldots, b_m)$ of relation r shall be deleted, and as the end of the period of the transaction time marking it, the current time for completion of the request for deletion shall be assigned. Only tuples which are part of the active state of the relation r could be deleted.

The relation r, which is updated, the values of the passive and active attributes $(a_1, a_2, \ldots, a_n)\,(b_1, b_2, \ldots, b_m)$, which have to be deleted, as well as their marking valid time period, shall be submitted as input parameters of the algorithm.

```
delete(r, (a1, a2, ..., an), (b1, b2, ..., bm), Vs, Ve):
CT ← current_time;
for each x ∈ r
    if CT < x[Te] and CT < x[Ve]
        if x[A] = (a1, a2, ..., an) and x[B] = (b1, b2, ..., bm) and equal(Vs, Ve, x[Vs], x[Ve])
            x[Te] ← CT;
            break;
return r;
```

The function equal shall return true, in case both intervals (the old and the new) shall match.

$$qual\left(v_s, v_e, v_s^{'}, v_e^{'}\right);$$

$$trueif v_s = v_s^{'} \wedge v_e = v_e^{'}$$

$$falseelse$$

International Electronic Journal of Pure and Applied Mathematics – IEJPAM, Volume 9, No. 3 (2015)

3.3. Algorithm for Modification of Data with HRTM

The existing tuple modification is executed by the operation update. It is defined as consecutive execution of the operations delete and inserts.

$$update\left(r,(a_1,a_2,\ldots,a_n),(b_1,b_2,\ldots,b_m),\left(a_1^{'},a_2^{'},\ldots,a_n^{'}\right),\left(b_1^{'},b_2^{'},\ldots,b_n^{'}\right),V_s,V_e,V_s^{'},V_e^{'}\right):$$

r$\leftarrow delete\left(r,(a_1,a_2,\ldots,a_n),(b_1,b_2,\ldots,b_m),V_s,V_e\right)$;

r$\leftarrow insert\left(r\left(a_1^{'},a_2^{'},\ldots,a_n^{'}\right),\left(b_1^{'},b_2^{'},\ldots,b_n^{'}\right),V_s^{'},V_e^{'}\right)$;

return r;

Using the proposed HRTM, the overall functionality regarding the storage, extracting and management of data stored shall be completed by DB, used by the hotel reservation system. References for expired, current and future reservations, as well as cancelled reservations could be completed quickly and easily.

4. Conclusion

In this work, a database temporal model has been proposed, used for the needs of hotel reservation systems. Algorithms have been presented for each of the three forms of updating – adding, deleting and modification of data in the database. In the proposed model, the overall functionality regarding the storage, extracting and management of stored data, shall be completed by the DB. References for expired, cancelled, current and future reservations could be completed quickly and easily.

References

[1] Christian Jensen, Richard Snodgrass, Temporal database entries for the Springer Encyclopedia of database systems, *A Timecenter Technical Report*, May 22, 2008.

[2] Jaymin Patel, *Temporal Database System*, Department of Computing, Imperial College, University of London, 2003.

[3] Christian Jensen, Michael Soo, Richard Snodgrass, Unifying temporal data models, *Temporal Database Management*, Dr.Techn. Thesis by Christian S. Jensen, 2000.

[4] Snodgrass, R., *A Case Study of Temporal Data*, Teradata Corporation, 2010.

[5] Dimitar Pilev, *Contemporary Information Systems with Temporal Databases*, Department of Programming and Computer Systems Application, Ph.D. Thesis, University of Chemical Technology and Metallurgy, 2011.

International Electronic Journal of Pure and Applied Mathematics – IEJPAM, Volume 9, No. 3 (2015)