Entities and Relations for Historical Relational Databases

Robert A. Morris and Lina Khatib
Florida Institute of Technology
Melbourne, FL 32901
email:{morris,lina}@cs.fit.edu

Abstract

Research to extend models of data to handle the temporal dimension has been conducted mainly in the context of the relational data model. In the relational model, the primary object of database design, manipulation and retrieval is the relation, viewed extensionally as a finite set of tuples. This paper proposes three useful types of temporal entities: events, histories, and valid-time relations, and defines relations and operations which allow for patterns of ordering and duration information to be defined and extracted. These patterns can often be expressed as select conditions on temporal relations in an algebraic query language This paper is both a survey of existing trends in temporal database research, and a refinement of the conceptual framework used to characterize temporal information.

1 Introduction & background

Many models for systems for reasoning about elements of a time domain (points, intervals) have been proposed ([1], [4], [19]). Most of these systems were developed for AI applications, such as scheduling. An emphasis in this research is placed on efficient ways of propagating relationships between events. This emphasis occurs for two reasons. First, input to a scheduler is typically in the form of ordering or metric constraints, which need to be effectively represented and managed. Second, the goal of a scheduling problem is the assignment of time values to a set of tasks, which are typically drawn from a large, even infinite domain. The search space of possible schedules is typically too large to be traversed without a considerable amount of constraint tightening. Indeed some scheduling problems can be considerably simplified by redefining the problem as one of finding solutions to a constraint labeling problem [18].

In database applications, the focus tends to be slightly different. Of primary importance here are the ability to effectively *query* a large, but finite database, and a means of automating the process of maintaining database *integrity*, viz. by declaring constraints on the set of admissible databases. On the one hand, querying an existing database implies that assignments of times have already been made (although occasionally part of this information may be unknown or missing). Consequently, the problem becomes one of effectively organizing data, rather than generating consistent assignments. On the other hand, declaring temporal constraints on databases seems more similar to AI applications, since questions of consistency and satisfiability arise. Thus, in general, there are overlapping, but not identical, concerns which must be addressed by builders of either database or AI applications in which time is represented.

Extending data models and associated query languages to handle the temporal dimension has been examined mainly in the context of the relational data model, although some research has focused on the object data model [20] or has attempted a more general temporal model of data (e.g., [5]). In the relational model, the primary object of database design, manipulation and retrieval is the *relation*, viewed extensionally as a finite set of tuples, and visualized as a table. The addition of time to a relational database management system (DBMS) allows for a representation of either *valid time*, the time during which some data accurately depicts something in the world, or *transaction time*, the time at which data is entered, changed, or deleted.

Historical data is the result of the association of information about the world to a value of a time do-

0-8186-7937-9/97 $10.00

main for the purpose of storing the history of some aspect of an object. A history is made up of *phases*, which profiles a period during which the object does not change a value of the aspect whose history is being recorded.

A *time domain* consists of either points or intervals, and assumes an ordering, either linear or non-linear. In addition, time can be bounded or unbounded, and, if bounded, either bounded in the past or the future. Third, time has a set of *metrics* for measuring distance and duration. These correspond to values in a calendar domain (months, years, etc.). Finally, the cardinality of the time domain can be discrete, dense or continuous. In this paper, the time domain will consist of intervals, represented by ordered pairs of natural numbers. Time will be viewed as discrete, linear, unbounded in the future and bounded in the past. These assumptions are chosen because they are common models of time in the representation of temporal data. For simplicity, we consider a *generic* time metric whose values are chosen from the set of natural numbers. This generic metric will be called a *reference interval.*

Once design issues related to temporal domains are addressed, there needs to be considered the nature of the association between domain values and other, non-temporal information about the world. Two kinds of association have been proposed in the relational model: *attribute time-stamping* and *tuple time-stamping.* The former involves an extension to the classical relational data model whereby attributes (i.e., value types of arguments to relations) have domains which are partial functions from times to data values. For example, the relation **Salary** in Table 1 describes the history of employee salaries. Each employee is uniquely identified by the value of the *emp* attribute. The attribute *salary* is a time-varying attribute. A value for this attribute is a function, represented visually as a set of pairs of times and salaries ($\times 1000$).This approach to associating times with data is referred to as the *grouped* approach.

emp	*salary*
a	[0, 2], 40 [3, 5], 45 [7, 8], 50
b	[5, 9], 42
c	[0, 6], 35 [8, 10], 40
d	[4, 9], 35

Table 1. **Salary** (attribute time-stamping)

Alternatively, time can be associated with *tuples.* Table 2 contains the same information as Table 1, with time intervals being associated with tuples, rather than attributes. The table has been *normalized for time* [12], so that each time interval is associated with a unique combination of object (again represented by the unique identifier of the object) and data value. In a normalized table, there are no overlapping time periods involving the same combination of entity and data value. This approach to associating times with intervals is sometimes called the *ungrouped* approach.

emp	*salary*	T_S	T_E
a	40	0	2
a	45	3	5
a	50	7	8
b	42	5	9
c	35	0	6
c	40	8	10
d	35	4	9

Table 2. **Salary** (tuple time-stamping)

There are many issues that collectively determine whether to associate time with attributes or tuples. These issues are beyond the scope of this paper (cf. [3]). Primarily, the issues are those of *completeness* (with respect to a set of queries), and efficiency in performing queries or updates. Primarily for reasons of simplicity, we will select the tuple time-stamped model as our paradigm for the association of data with time, although the framework here has an obvious counterpart in the grouped time model.

The last stage in the specification of temporal models for database management, and the focus of this paper, is the identification and classification of temporal entities. Members of this set are the objects on which relations or operations for specifying and extracting data from relations are defined. This paper proposes a robust ontology for temporal database management, restricted to the representation of valid-time. Three distinct, but related, temporal entities are defined: durations, histories, and valid-time relations. The temporal component of these entities will be isolated to reveal structures whose properties can be identified using relations or operations defined over them.

This paper contributes to the development of formal systems for extracting and defining historical relational databases by classifying relations and operations over the temporal components of temporal database entities. Hence, the perspective of this paper is on models for formal systems, rather than on the systems themselves. Each of these relations or operations can be viewed as a finite series of simpler relations or operations on temporal domain el-

ements, and provide the interpretation of languages for defining and querying temporal databases. The focus of this paper is potentially useful for classifying and evaluating data definition and query languages for temporal database management systems in terms of their expressive power. This paper is presented as an informal survey, both synthesizing work that has been conducted previously, and introducing useful distinctions that have not, to our knowledge, been made before.

The remaining sections describe classes of temporal entities, the associated time domain elements from which they are constructed, and relationships that can occur among them, and from which meaningful patterns of temporal information can be defined and extracted.

2 Temporal Entities for Relational Data

Intuitively, an entity is anything "talked about" in discourse, anything with definable properties and relations. Formally, it has been proposed to ascribe entity status to anything in the domain of interpretations of formal languages, or, equivalently, anything that is the possible value of a variable of quantification in a formal language. For our purposes, an entity will be viewed as anything manipulated, compared, or retrieved in the process of defining or extracting data from a relational database.

Classes of temporal entities are distinguished by characteristics of their underlying temporal elements. We distinguish among three types of temporal data entity: durations, histories, and valid-time relations. A *duration* is the simplest type of temporal entity: it is the result of a single association of a piece of information with an element of a time domain. Formally, a duration corresponds to a tuple in a (normalized) tuple time-stamped relation. Thus, each row of Tables 2 and 3 correspond to a single duration. (In Table 1, where attributes rather than tuples are time-stamped, a duration corresponds to a tuple in which the time attribute is restricted to a single interval of time). Secondly, a *history* formally corresponds to a set of tuples from a time-valid relation, viz., the set with a common value for the unique identifier for the object, sometimes called the *time-invariant key.* For example, in Table 3, the first two rows is a history of employee a's position. The set of intervals in a history must be linearly ordered (each one but the last is either before or meets one and only one next element). Thus, it is natural in the case of histories to refer to the *ith* period within the history.

Finally, a *valid-time relation* generalizes the notion of a history into a set of durations which describe a lifespan of an association between objects and other data. For example, Table 3 provides a set of durations of positions held by employees. Another example of a valid-time relation is the result of the query *the set of tuples in the relation* **Position** *in which the value for* position *is X.* In a valid time relation, there is no constraint on the ordering of the times. However, we can identify a *lifespan* of a relation in terms of an interval bounded by the earliest start time and the latest end time of any tuple in the relation. For example, the lifespan of **Position** is $\langle 0, 10 \rangle$.

emp	*position*	T_S	T_E
a	X	0	2
a	W	3	6
b	X	4	6
b	Y	8	10
c	W	0	5
c	X	7	9
e	W	4	9

Table 3. **Position**

Associated with each of the temporal entities is a distinct variety of *temporal sub-strata* (to use a term of VanBenthem [16]). We refer to the temporal sub-strata of durations, histories, and valid-time relations as (closed) intervals, sequences of intervals, and unions of intervals, respectively. Formally, we represent an interval in the standard way as a pair $I = \langle I_S, I_E \rangle, I_S \leq I_E$ of natural numbers. An interval sequence will similarly be depicted as

$$I = \langle I_{S_1}, I_{E_1}, \ldots I_{S_n}, I_{E_n} \rangle, I_{S_i} \leq I_{E_i} \leq I_{S_{i+1}}$$

or, equivalently, as a sequence of intervals $I = \langle I_1, \ldots, I_k \rangle$, where each I_j is before or meets $I_{j+1}, j = 1 \ldots k-1$. Finally, a union of intervals is any finite set of intervals.

3 Operations and relations on temporal sub-strata

In this section, we provide a classification of binary relations that can occur among the temporal sub-strata of temporal entities. Restricting attention to binary relations does not limit the analysis, since, as will be illustrated, all relations and operations can be viewed as a series of operations on pairs of temporal entities.

We first broadly classify operations and relations into two kinds: unit and collective. *Unit* operations or relations are so-called because they treat temporal entities as a single unit. *Collective* operations and relations treat temporal entities as collections of simpler entities. Among the collective relations we further distinguish between *disjunctive*, *conjunctive* and *cumulative* relations. Conjunctive (disjunctive) relations are so-called because they can be represented naturally as conjunctions (disjunctions) of simpler relations; cumulative relations, by contrast, involve a function which accumulates some value related to order or duration. As example of a cumulative temporal relation is the difference between the average duration of pairs of distinct histories. More examples will be provided below.

Formally, a conjunctive relation on a pair S_1 and S_2 of histories or valid times is a conjunction of interval ordering relations r on a subset of $S_1 \times S_2$, i.e.

$$R(S_1, S_2) = \bigwedge r_{i,j}(i,j) : (i,j) \in S_1 \times S_2.$$

Disjunctive relations can be expressed analogously as disjunctions of simpler relations. Construction of a collective relation R requires determining which subset of $S_1 \times S_2$ is being compared with respect to ordering; this can be viewed as establishing a *correspondence* between elements of each set (the notion of correspondence is described more fully in [10]). There are two common ways of establishing correspondence: one is temporal proximity; the other, which assumes that the collections are linearly ordered, is position. For example, consider the statement *Each period in which a occupied some position is either started by or equal to one of a's salary period.* To evaluate this sentence, one compares each of a's position durations with a's salary durations. Contrast this statement with *a's nth position durations is started by or equal to a's nth salary duration, for all n.* This is made true by the same pairings of tuples, but the correspondence is established by position in the history, rather than location in time.

Time has two primary characteristics: *order* and *duration*. The following sections define and provide examples of unit and collective operations and relations involving order and duration.

3.1 Basic operations on temporal entities

This section describes a set of basic operations on the sub-strata of temporal entities. These operations form the basis for constructing the relations described below.

The simplest of operations are unary or binary operations on intervals. First, endpoints of intervals need to be compared and ordered with respect to time. Thus, relations on endpoints "less than", "equal to", and related operations such as *min* and *max* are required. Second, given a pair of intervals $\langle I_S, I_E \rangle$ and $\langle J_S, J_E \rangle$, where $I_E < J_S$, the operation of *difference* computes $J_S - I_E$. Another operation which involves difference is *duration*: this is the value $I_E - I_S$. End point comparison and difference together establish the basis for all discourse about time, viz., by defining the basic concepts of order and duration.

From these, useful operations for constructing temporal sub-strata from others can be defined. Three useful distinct operations are coalesce, intersection and gap. First, viewed as binary operations, *coalesce* takes a pair of intervals $\langle I_S, I_E \rangle, \langle J_S, J_E \rangle$, such that $I_S < J_E$ and returns the interval $\langle I_S, J_E \rangle$. *Intersection* is defined for a pair of intervals $\langle I_S, I_E \rangle, \langle J_S, J_E \rangle$, such that $J_S < I_E$ and returns the interval $\langle J_S, I_E \rangle$. Finally, the *gap* operation is defined for a pair of (closed) intervals $\langle I_S, I_E \rangle, \langle J_S, J_E \rangle$, such that $I_E < J_S$ and returns the interval $\langle I_E + 1, J_S - 1 \rangle$. Each of these simpler operations can be generalized into operations on sequences or unions of intervals. For example, history coalescence returns the interval bounded by the earliest start time and latest end time of the sequence. Similarly, (pairwise) intersection of two histories results in a history composed of the pairwise intersection of the *ith* stage of each history for each i less than or equal to the length of the smaller of the two histories.

Finally, there are operations that are applied only to collective entities. For example, the *number* operation returns the number of durations in either a history or a valid time relation.

3.2 Temporal ordering relations

Order can be established either absolutely, by associating a temporal entity with a reference interval, or relatively, in terms of other temporal entities. Furthermore, order can be exact or approximate. For example, *a's first position occurred at* $\langle 0, 2 \rangle$ is an exact absolute order assignment; by contrast, to associate *a*'s first salary with the interval $\langle 0, 5 \rangle$ is approximate. Approximate relations introduce notions associated with containment and overlap.

There are exact and approximate absolute ordering relations involving histories and valid time re-

lations: e.g., the association of a's position history with $\langle 0, 2, 3, 6\rangle$ is exact. The latter is a collective exact relationship, because it is a conjunction of exact relationships among the durations of a's position history.

Another important kind of unit relationship is between the *extent* of a history or valid time relation and a time. By *extent* is meant the interval bounded by the earliest start time and latest end time of the collection. For example, the association of a's position history with $\langle 0, 6\rangle$ treats the history as a single interval, and assigns it an absolute, exact ordering.

Relative temporal order associates an entity with another with respect to time. To say that a's first position is before b's expresses an exact relative ordering; by contrast, to say that a's first position is either before or during b's is approximate. To say that a held some position before e did is to establish a collective, disjunctive relation between histories. As with absolute order, relative unit ordering relations can be expressed, e.g., by sentences like *The time during which a held all her positions overlapped the time during which b held all his positions.*

An important relative ordering is containment, as between a history in a valid time relation and the relation as a whole. With histories, relative ordering is established by position in the sequence of phases in the history.

The final set of examples concern cumulative relations among histories or valid-time relations. Consider first this larger history of employee e's salary:

emp	*salary*	T_S	T_E
e	30	0	2
e	34	3	4
e	35	4	6
e	40	8	10
e	44	12	15
e	45	17	19
e	50	20	25

Table 4. e's Salary History

Cumulative relations allow for the statistical analysis of temporal data. An example of absolute cumulative ordering relations is *distribution of durations* as, e.g., expressed by *most of e's distinct salary periods happened during* the sequence $\langle 0, 2, 3, 4, 4, 6, 8, 10\rangle$ This relation assumes the ability to determine the *number* of stages in a history. Similarly, we can say *Most of e's salary periods happened while e earned a single salary*, which establishes a relative cumulative ordering. Quantitative as well as qualitative cumulative relations can be expressed by assigning percentages to orders. Cumulative ordering relations, whether absolute or relative, can be approximate or exact. For example, the sentence *Less than half of e's history occurred before time 11* is approximate with respect both to number and time.

There is a metric counterpart to relative and absolute order relations. *Metric relations* ([4]) compare the differences between times of temporal entity. The simplest absolute difference relation can be expressed in the form

$$I_k \; R \; n,$$

where $k \in \{S, E\}$, n is a positive integer, and $R \in \{<, \leq, =\}$ [8]. For example, the sentence *a's first salary started at time 0* expresses a metric ordering relation. A complete ordering can be expressed as a conjunction of difference relations among the endpoints. Similarly, a relative difference ordering can be expressed as a conjunction of expressions of the form

$$(I_F - J_G) \; R \; n$$

Collective and cumulative relations can be constructed out of metric relations by a series of operations on differences.

An important kind of difference relation is *duration*. The simplest duration relation is of the form $(I_E - I_S) = n$. This formalizes the answer to the query *how long is I?*. Similarly, durations can be compared, as is required to answer queries of the form: which is longer? These relations have the form $I_E - I_S \; R \; J_E - J_S$, $R \in \{<, \leq, =\}$. As with ordering relations, histories and valid time relations can be viewed either collectively or as a single temporal unit. Metric duration relations have qualitative counterparts in relations like *longer (in duration) than* [10].

This section has presented an informal survey of temporal operations and relations defined on durations, histories and valid-time relations. All temporal operations or relations concern some aspect of order or duration. Intervals, the temporal substrata of durations, have either absolute or relative relationships with other intervals, and have a part-whole structure defined with sequences or unions of intervals. Interval sequences, the temporal substrata of histories, as well as interval unions, the temporal sub-strata of valid-time relations, can be viewed as single units of time (via applying the coalesce operator) or as collections. Viewed as collections, collective and cumulative relations can be defined.

4 Other related research; extensions

In addition to the references already cited in this paper, here is a brief summary of other related research. The historical relational data model [2] describes lifespans of relations. The work of Gadia [6] employs the notion of a temporal element as a finite union of intervals. The concept of time normalization found in the work of Navathe [11] is important in formalizing the notion of temporal entity. The HSQL model (Sarda, [13]) emphasizes the fact that processing temporal queries can be viewed as a specialization of relational algebra query operations such as select and join. Finally, the work of Goodwin et. al. [7], is relevant in the characterization of cumulative relations, and the work of Ladkin [9], and Morris, et. al. [10] provides a discussion of what was here described as collective relations.

There are two important areas of temporal relational data processing not explored in this paper: handling uncertain information, and infinite databases. The first problem is addressed by extending the data model to include a method of *temporal projection*, which introduces non-monotonic reasoning into the model.

Handling infinite temporal data in the relational model requires a finite description of infinite relations. The work on periodic time [15] is relevant here. A generalization of the temporal relational data model to handle infinite temporal information results from the notion of a *generalized tuple* consisting of *linear repeating points* and *constraints*. A linear repeating point is an expression of the form $an + b$, which establishes the period between occurrences of events, whereas constraints allow for limits of duration to be expressed for each of the set of intervals in a repeated event. The distinctions drawn in this paper can be applied, with minor modifications, to infinite data.

5 Conclusion

This paper has explored the requirements for a language for querying and constraining relational databases containing temporal information. A temporal relational database can be viewed as supporting three kinds of temporal entities: durations, histories, and valid-time relations. The temporal substrata of each entity has its own structure and organization. Consequently, distinct classes of systems of temporal relations can be defined for members of each class. This paper has presented an informal survey of the classes of relations and operations on temporal entities. Additional theoretical work would consist of associating these relation classes with systems of temporal logic. Work of a more practical nature remains of finding historical data the definition and extraction of which would be effectively achieved by applying the framework presented here.

References

[1] Allen, J. 1983. Maintaining Knowledge About Temporal Intervals. In *Readings in Knowledge Representation*, Brachman, R., and Levesque, H., (editors). (SanMateo:Morgan Kaufman), 510-521.

[2] Clifford, J. and Croker, A. The Historical Relational Data Model (HRDM) and Algebra Based on Lifespans. In *Proceedings of the Third International Conference on Data Engineering*, 528-537, (1987).

[3] Clifford, J., Croker, A., and Tuzhilin, A. On the Completeness of Query Languages for Grouped

[4] Dechter, R., Meiri, I., and Pearl, J. Temporal Constraint Networks. In Brachman, R., Levesque, H., and Reiter, R., eds. *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning.* Morgan Kaufmann, 1989, 83-93.

[5] Elmasri, R., Wuu, G., and Kouramajian, V., A Temporal Model and Query Language for EER Databases. In [14], Chapter 9.

[6] Gadia, S.K. The Role of Temporal Elements in Temporal Databases. *IEEE Data Engineering*, 1988.

[7] Goodwin, Scott D., Neufeld, E., Trudel, A. Interpolating Definite Integral Information. *Proceedings of the Fifth Florida Artificial Intelligence Research Symposium, FLAIRS-92*, 119-123.

[8] Kautz, H., and Ladkin, P., Integrating Qualitative and Metric Temporal Reasoning. *AAAI Spring Symposium on Constraint-based Reasoning, Working Notes*, 1991, 66-78.

[9] Ladkin, P.B. 1986. Primitives and Units for Time Specification. In *Proceedings of AAAI-86*, (San Mateo:Morgan Kaufman), 354-359.

[10] Morris, R., Shoaff, W., and Khatib, L., (1996) Domain Independent Reasoning About Recurring Events. *Computational Intelligence* 12(3), 450-477.

[11] Navathe, S., and Ahmed, R. A Temporal Relational Model and A Query Language. *Information Sciences*, 49(2), 147-175, 1989.

[12] Navathe, S., and Ahmed, R. Temporal Extensions to the Relational Data Model and SQL. in [14], Chapter 4.

[13] Sarda, N. L. Modeling of Time and History Data in Database Systems. In *Proceedings CIPS Congress 87.* 15-30, CIPS, (1987).

[14] Tansel, A., Clifford, J., Gadia, S., Jajodia, S., Segev, A., and Snodgrass, R., *Temporal Databases: Theory, Design, and Implementation.* Benjamin/Cummings, 1993.

[15] Tuzhilin, A., and Clifford, J., 1995. On Periodicity in Temporal Databases. *Information Systems* **30**(5), 619-639.

[16] Van Benthem, J. 1980. Points and Periods. in (Rohrer 1980), 39-57.

[17] Van Beek, P. 1990. *Exact and Approximate Reasoning about Qualitative Temporal Relations.* Technical Report TR 90-29, University of Alberta, Edmonton, Alberta, Canada.

[18] Van Beek, P. 1992. Reasoning about Qualitative Temporal Information. *Artificial Intelligence* **58**:297-326.

[19] Villain, M., Kautz, H., and Van Beek, P. 1989. Constraint Propagation for Temporal Reasoning: A Revised Report. In *Readings in Qualitative Reasoning about Physical Systems*, Morgan-Kaufmann, 351-357.

[20] Wuu, G., and Dayal, U., A Uniform Model for Temporal and Versioned Object-oriented Databases. In [14], Chapter 10.