A Glossary of Temporal Database Concepts*

C. S. Jensen J. Clifford S. K. Gadia A. Segev R. T. Snodgrass

Abstract

This glossary contains concepts specific to temporal databases that are well-defined, well understood, and widely used. In addition to defining and naming the concepts, the glossary also explains the decisions made. It lists competing alternatives and discusses the pros and cons of these. It also includes evaluation criteria for the naming of concepts.

This paper is a structured presentation of the results of e-mail discussions initiated during the preparation of the first book on temporal databases, *Temporal Databases: Theory, Design, and Implementation,* published by Benjamin/Cummings, to appear January 1993. Independently of the book, an initiative aimed at designing a consensus Temporal SQL is under way. The paper is a contribution towards establishing common terminology, an initial subtask of this initiative.

1 Introduction

Maintaining a precise, well-defined, and intuitive technical language is important to the scientific community. In this glossary we propose definitions and names of a range of concepts specific to temporal databases that are well-defined, well understood, and widely used. The glossary meets a need for creating a higher degree of consensus on the definition and naming of central concepts from within the field. The use of inconsistent terminology adversely affects the accessibility of the literature and has also an adverse effect on progress.

Being a proposal, simply stating definitions and names would be counter-productive and against the intentions. Consequently, the paper presents alternatives and discusses why the specific decisions were made. When several alternative names for a concept were considered before a decision was made, the paper not only states that decision, but it also presents the alternatives and discusses why the decision was made. This glossary arose from e-mail discussions among the authors considering appropriate terminology for a forthcoming collection of articles [TCG+92]. A list of papers and other bibliographies that define and use the concepts discussed here may be found elsewhere [Soo91].

A consensus Temporal SQL is being designed. An initial white paper [Sno92] lists fifteen design subtasks. This paper addresses one of the first tasks, the establishment of common terminology.

The next section first presents a list of relevance criteria for concepts. All concepts included in the glossary satisfy these criteria. Second, it defines a number of evaluation criteria for the naming of concepts. These criteria have emerged from the discussions. Third, the structure of the proposal is described. Section 3 presents the proposed terms and concepts.

2 Criteria for the Glossary

We have found it useful to impose four relevance criteria for concepts when compiling the glossary—these are presented first. Then evaluation criteria for the naming of concepts are intro-

*Correspondence may be directed to the following addresses: C. S. Jensen, Aalborg University, Datalogi, Fr. Bajers Vej 7E, DK–9220 Aalborg Ø, Denmark, csj@iesd.auc.dk, J. Clifford, Information Systems Dept., Stern School of Business, 613 Tisch Hall, 40 W. 4th Street, New York, NY 10003, jcliffor@is-4.stern.nyu.edu, S. K. Gadia, Iowa State University, Computer Science Dept., 226 Computer Science Building, Ames, IA 50010, gadia@cs.iastate.edu, A. Segev, University of California, School of Business Adm. and Computer Science Research Dept., Lawrence Berkeley Laboratory, 350 Barrows Hall, Berkeley, CA 94720, segev@csr.lbl.gov, and R. T. Snodgrass, University of Arizona, Computer Science Dept., Tucson, AZ 85721, rts@cs.arizona.edu.

This paper appeared in SIGMOD Record, Vol. 21, No. 3, September 1992.

duced. The criteria will be employed in Section 3. Finally, the structure of the presentation of each proposed term is outlined.

2.1 Relevance Criteria for Concepts

We will attempt only to name concepts that fulfill the following four requirements.

R1 The concept must be specific to temporal databases. Thus, concepts used more generally are excluded.

R2 The concept must be well-defined. Before attempting to name a concept, it is necessary to agree on the definition of the concept itself.

R3 The concept must be well understood. We have attempted to not name a concept if a clear understanding of the appropriateness, consequences, and implications of the concept is missing. Thus, we avoid concepts from research areas that are currently being explored.

R4 The concept must be widely used. We have avoided concepts used only sporadically within the field.

2.2 Evaluation Criteria for Naming Concepts

We list a series of criteria for what is a good name. These criteria are sometimes conflicting, making the choice of names a difficult and challenging task.

E1 The naming of concepts should be orthogonal. Parallel concepts should have parallel names.

E2 Names should be easy to write, i.e., they should be short or possess a short acronym, should be easily pronounced (the name or its acronym), and should be appropriate for use in subscripts and superscripts.

E3 Already widely accepted names are preferred over new names.

E4 Names should be open-ended in the sense that the name of a concept should not prohibit the invention of a parallel name if a parallel concept is defined.

E5 We have avoided creating homographs and homonyms. Names with an already accepted meaning, e.g., an informal meaning, should not be given an additional meaning.

E6 We have striven to be conservative when naming concepts. No name is better than a bad name.

E7 New names should be consistent with related and already existing and accepted names.

E8 Names should be intuitive.

E9 Names should be precise.

While we do find the above list to be comprehensive, we do not claim that is is complete. In Section 3, we will refer to and exemplify the criteria.

2.3 Structure of the Proposal

In the following, we will name concepts selected by applying the four above principles. For most of the concepts being named, we will employ the same template:

Name—the chosen name of the concept is used as the heading.

Definition—the definition of the concept.

Alternative Names—names that were considered and subsequently rejected in favor of the chosen name.

Discussion—reasons for the particular choice of name (and concept) and reasons for not selecting considered names (and concepts).

When no alternative names have been proposed, the third entry is not included. With no alternatives at all, the last entry is also omitted.

3 Proposed Names and Concepts

3.1 Valid Time

Definition

The *valid time* of a fact is the time when the fact is true in the modeled reality. A fact may have associated any number of events and intervals, with single events and intervals being important special cases.

Alternative Names

Real-world time, intrinsic time, logical time, data time.

Discussion

Valid time is widely accepted already (+E3); it is short and easily spelled and pronounced (+E2). Most importantly, it is intuitive (+E8).

The name "real-world time" derives from the common identification of the modeled reality (opposed to the reality of the model) as the real world (+E8). This name has no apparent advantages to valid time, and it is less frequently used and longer (−E3, −E2).

"Intrinsic time" is the opposite of extrinsic time. Choosing intrinsic time for valid time would require us to choose extrinsic time for transaction time. The names are appropriate: The time when a fact is true is intrinsic to the fact; when it happened to be stored in a database is clearly an extrinsic property. Still, "intrinsic" is rarely used (−E3) and is longer and harder to spell than "valid" (−E2). As we shall see, transaction time is preferred over "extrinsic time" as well. Also, should a third concept of time be invented, there will be no obvious name for that concept (−E4).

"Logical time" has been used for valid time in conjunction with "physical time" for transaction time. As the discussion of intrinsic time had to include extrinsic time, discussing logical time requires us to also consider physical time. Both names are more rarely used than valid and transaction time (−E3), and they do not posses clear advantages over these.

The name "data time" is probably the most rarely used alternative (−E3). While it is clearly brief and easily spelled and pronounced, it is not intuitively clear that the data time of a fact refers to the valid time as defined above (+E2,−E8).

3.2 Transaction Time

Definition

A database fact is stored in a database at some point in time, and after it is stored, it may be retrieved. The *transaction time* of a database fact is the time when the fact is stored in the database. Transaction times are consistent with the serialization order of the transactions. Transaction time values cannot be after the current time. Also, as it is impossible to change the past, transaction times cannot be changed. Transaction times may be implemented using transaction commit times.

Alternative Names

Registration time, extrinsic time, physical time.

Discussion

Transaction time has the advantage of being almost universally accepted (+E3), and it has no conflicts with valid time (+E1, +E4, +E7).

Registration time seems to be more straight forward. However, often a time of a particular type is denoted by t_x where x is the first letter of the type. As r is commonly used for denoting a relation, adopting registration time creates a conflict (−E2).

Extrinsic time is rarely used (−E3) and has the same disadvantages as intrinsic time.

Finally, physical time is used infrequently (−E3) and seems vague (−E8).

3.3 User-defined Time

Definition

User-defined time is an uninterpreted attribute domain of date and time. User-defined time is parallel to domains such as "money" and integer—unlike transaction time and valid time, it has no special query language support. It may be used for attributes such as "birth day" and "hiring date."

Conventional database management systems generally support a time and/or date attribute domain. The SQL2 standard has explicit support for user-defined time in its `datetime` and `interval` types.

3.4 Valid-Time Relation

Definition

A *valid-time relation* is a relation with exactly one system supported valid time. In agreement with the definition of valid time, there are no restrictions on how valid times may be associated with the tuples (e.g., attribute value time stamping may be employed).

Alternative Names

Historical relation.

Discussion

While historical relation is used currently by most authors (+E3), two problems have been pointed out. First, the qualifyer "historical" is too generic (−E5). Second, "historical," being a reference to the past, is misleading because a valid-time relation may also contain facts valid in the future (−E8, −E9).

"Valid-time relation" is straight forward and avoids these problems. Also, it is consistent with "transaction time relation," to be discussed next (+E1).

3.5 Transaction-Time Relation

Definition

A *transaction-time relation* is a relation with exactly one system supported transaction time. As for valid-time relations, there are no restrictions as to how transaction times may be associated with the tuples.

Alternative Names

Rollback relation.

Discussion

"Transaction-time relation" is already used by several authors, but other authors use the name "rollback relation." The motive for adopting transaction-time relation is identical for the motive for adopting valid-time relation. The motive for adopting rollback relation is that this type of relation supports a special rollback operation (+E7). But then, for reasons of parallelity, should not a valid-time relation be named for the special operation on valid-time relations corresponding to the rollback operation, namely transaction timeslice (−E4)?

3.6 Snapshot Relation

Definition

Relations of a conventional relational database system incorporating neither valid-time nor transaction-time timestamps are *snapshot relations*.

Alternative Names

Relation, conventional relation, static relation.

Discussion

With several types of relations, simply using "relation" to denote one type is often inconvenient. The modifier "snapshot" is widely used (+E3). In addition, it is easy to use and seems precise and intuitive (+E2,9,8). The alternative "conventional" is longer and used more infrequently. Further, "conventional" is a moving target—as technologies evolve, it changes meaning. This makes it less precise. Finally, "static" is less frequently used than "snapshot," and it begs for the definition of the opposite concept of a dynamic relation, which will not be defined (−E3, −E1).

3.7 Bitemporal Relation

Definition

A *bitemporal relation* is a relation with exactly one system supported valid time and exactly one system-supported transaction time.

Alternative Names

Temporal relation, fully temporal relation, valid-time and transaction-time relation, valid-time transaction-time relation.

Discussion

We first discuss the concept; then we discuss the name.

In the adopted definition, "bi" refers to the existence of exactly two times. An alternative definition states that a bitemporal relation has one or more system supported valid times and one or more system supported transaction times. In this definition, "bi" refers to the existence of exactly two types of times.

Most relations involving both valid and transaction time are bitemporal according to both definitions. Being the most restrictive, the adopted definition is the most desirable: It is the tightest fit, giving the most precise characterization (+E9).

The definition of bitemporal is used as the basis for applying bitemporal as a modifier to other concepts such as "query language." This adds more important reasons for preferring the adopted definition.

Independently of the precise definition of bitemporal, a query language is bitemporal if and only if it supports any bitemporal relation (+E1), see Section 3.8. With the adopted definition, most query languages involving both valid and transaction time may be characterized as bitemporal. With the alternative definition, query languages that are bitemporal under the adopted definition are no longer bitemporal. This is a serious drawback of the alternative definition. It excludes the possibility of naming languages that may be precisely named using the adopted definition. With the alternative definition, those query languages have no (precise) name. What we get is a concept and name (bitemporal query language) for which there is currently little or no use.

Also, note that a query language that is bitemporal with the alternative definition is also bitemporal with regard to the adopted definition (but the adopted definition does not provide a precise characterization of this query language). Thus, the restrictive definition of a bitemporal relation results in a non-restrictive definition of bitemporal query language (and vice-versa).

The name "temporal relation" is commonly used. However, it is also used in a generic and less strict sense, simply meaning any relation with some time aspect. It will not be possible to change the generic use of the term (−E7), and since using it with two meanings causes ambiguity (−E9), it is rejected as a name for bitemporal relations. In this respect "temporal relation" is similar to "historical relation."

Next, the term "fully temporal relation" was proposed because a bitemporal relation is capable of modeling both the intrinsic and the extrinsic time aspects of facts, thus providing the "full story." However, caution dictates that we avoid names that are absolute (−E6). What are we going to name a relation more general than a temporal relation?

The name "valid-time and transaction-time relation" is precise and consistent with the other names, but it is too cumbersome to be practical (−E2). Also, it may cause ambiguity. For example, the sentence "the topic of this paper is valid-time and transaction-time relations" is ambiguous.

We choose to name relations as opposed to databases because a database may contain several types of relations. Thus, naming relations is a more general approach.

3.8 Snapshot, Valid- and Transaction-Time, and Bitemporal as Modifiers

The definitions of how "snapshot," "valid-time," "transaction-time," and "bitemporal" apply to relations provide the basis for applying these modifiers to a range of other concepts. Let x be one of snapshot, valid-time, transaction-time, and bitemporal. Twenty derived concepts are defined as follows (+E1).

relational database An x relational database contains one or more x relations.

relational algebra An x relational algebra has relations of type x as basic objects.

relational query language An x relational query language manipulates any possible x relation. Had we used "some" instead of "any" in this definition, the defined concept would be very imprecise (−E9).

data model An x data model has an x query language and supports the specification of constraints on any x relation.

DBMS An x DBMS supports an x data model.

The two model-independent terms, data model and DBMS, may be replaced by more specific terms. For example, "data model" may be replaced by "relational data model" in "bitemporal data model."

3.9 Temporal as Modifier

Definition

The modifier *temporal* is used to indicate that the modified concept concerns some aspect of time.

Alternative Names

Time-oriented.

Discussion

"Temporal" is already being used in the sense defined here. In addition, some researchers have used in a more specific sense (i.e., supports both transaction time and valid time). This practice was awkward: Using "temporal" with the general definition in the beginning of a paper and then adopting the more specific meaning later in the paper created confusion. It also lead to the use of "time-oriented" instead of temporal in the generic sense.

Realizing that the use of the generic meaning of "temporal" cannot be changed prompted the adoption of "bitemporal' for the specific meaning.

Being only the name of a generic concept, "temporal" may now be used instead of the more cumbersome "time-oriented." It may be applied generically as a modifier for "database," "algebra," "query language," "data model," and "DBMS."

3.10 Temporal Database

Definition

A *temporal* database supports some aspect of time, not counting user-defined time.

Alternative Names

Time-oriented database, historical database.

Discussion

The concept of a temporal database is defined separately due to its importance. The discussion in Section 3.9 applies here.

3.11 Transaction Timeslice Operator

Definition

The *transaction timeslice operator* may be applied to any relation with a transaction time. It also takes as argument a time value not exceeding the current time, *NOW*. It returns the state of the argument relation that was current at the time specified by the time argument.

Alternative Names

Rollback operator, timeslice operator, state query.

Discussion

The name "rollback operator" has procedural connotations, which in itself is inappropriate (−E8). Why not use "rollforward operator?" The choice between one of them is rather arbitrary. Further, the transaction timeslice operator may be computed using both rollback (decremental computation) and rollforward (incremental computation).

"State query" seems less precise than transaction timeslice operator (−E9). It is equally applicable as a name for the valid timeslice operator (−E8). Further, "state operator" is better than "state query."

The name "transaction timeslice" may be abbreviated to timeslice when the meaning is clear from the context.

3.12 Valid Timeslice Operator

Definition

The *valid timeslice operator* may be applied to any relation with a valid time. It takes as argument a time value. It returns the state of the argument relation that was valid at the time of the time argument.

Alternative Names

Timeslice operator.

Discussion

"Valid timeslice operator" is consistent with transaction timeslice operator (+E1). "Timeslice" is appropriate only in a disambiguating context (+E2).

3.13 Temporal Element

Definition

A *temporal element* is a finite union of n-dimensional time boxes. Temporal elements are closed under the set theoretic operations of union, intersection and complementation.

Temporal elements may be used as timestamps. Special cases of temporal elements occur as timestamps in valid-time relations, transaction-time relations, and bitemporal relations. These special cases are termed *valid-time elements*, *transaction time elements*, and *bitemporal elements*. They are defined as finite unions of valid-time intervals, transaction-time intervals, and bitemporal rectangles, respectively.

Alternative Names

Temporal element.

Discussion

A valid time element was previously termed a temporal element. However, for the naming to be consistent with the remainder of the glossary, "temporal" is reserved as a generic modifier, and more specific modifiers are adopted.

3.14 Chronon

Definition

A *chronon* is the shortest duration of time supported by a temporal DBMS—it is a nondecomposable unit of time. A particular chronon is a subinterval of fixed duration on time-line.

Various models of time have been proposed in the philosophical and logical literature of time (e.g., van Benthem). These view time, among other things, as discrete, dense, or continuous. Intuitively, discrete models of time are isomorphic to the natural numbers, i.e., there is the notion that every moment of time has a unique successor. Dense models of time are isomorphic to (either) the real or rational numbers: between any two moments of time there is always another. Continuous models of time are isomorphic to the real numbers, i.e., both dense and also, unlike the rational numbers, with no "gaps."

Alternative Names

Instant, moment, time quantum, time unit.

Discussion

"Instant" and "moment" invite confusion between a *point* in the continuous model and a nondecomposable *unit* in the discrete model (−E8). Clocking instruments invariably report the occurrence of events in terms of time intervals, not time "points." Hence, events, even so-called "instantaneous" events, can best be measured as having occurred during an interval (−E9). "Time quantum" is precise, but is longer and more technical than "chronon" (−E2). "Time unit" is perhaps less precise (−E9).

3.15 Timestamp

Definition

A *timestamp* is a time value associated with some time-stamped object, e.g., an attribute value or a tuple. The concept may be specialized to valid timestamp, transaction timestamp, interval timestamp, event timestamp, bitemporal element timestamp, etc.

3.16 Lifespan

Definition

The *lifespan* of a database object is the time over which it is defined. The valid-time lifespan of a database object refers to the time when the corresponding object exists in the modeled reality, whereas the transaction-time lifespan refers to the time when the database object is current in the database.

If the object (attribute, tuple, relation) has an associated timestamp then the lifespan of that object is the value of the timestamp. If components of an object are timestamped, then the lifespan of the object is determined by the particular data model being employed.

Alternative Names

Timestamp, temporal element, temporal domain.

Discussion

Lifespan is widely accepted already (+E3); it is short and easily spelled and pronounced (+E2). Most importantly, it is intuitive (+E8).

3.17 Temporally Homogeneous

Definition

A temporal tuple is *temporally homogeneous* if the lifespan of all attribute values within it are identical. A temporal relation is said to be temporally homogeneous if its tuples are temporally homogeneous. A temporal database is said to be temporally homogeneous if all its relations are temporally homogeneous. In addition to being specific to a type of object (tuple, relation, database), homogeneity is also specific to some time dimension, as in "temporally homogeneous in the valid-time dimension" or "temporally homogeneous in the transaction-time dimension."

The motivation for homogeneity arises from the fact that no timeslices of a homogeneous relation produce null values. Therefore a homogeneous relational model is the temporal counterpart of the snapshot relational model without nulls. Certain data models assume temporal homogeneity. Models that employ tuple timestamping rather than attribute value timestamping are necessarily temporally homogeneous—only temporally homogeneous relations are possible.

Alternative Names

Homogeneous.

Discussion

In general, using simply "homogeneous" without "temporal" as qualifier may cause ambiguity because the unrelated notion of homogeneity exists also in distributed databases (−E5).

3.18 Event

Definition

An *event* is an isolated instant in time. An event is said to occur at time t if it occurs at any time during the chronon represented by t.

Alternative Names

Instant, moment.

Discussion

Both "instant" and "moment" may be confused with the distinct term "chronon" (−E5, −E7).

3.19 Interval

Definition

An *interval* is the time between two events. It may be represented by a set of contiguous chronons.

Alternative Names

Time period.

Discussion

The name "interval" is widely accepted (+E3). The name "period" often implies a cyclic or recurrent phenomenon (−E8, −E9). In addition, "time period" is longer (−E2).

3.20 Span

Definition

A *span* is a directed duration of time. A duration is an amount of time with known length, but no specific starting or ending chronons. For example, the duration "one week" is known to have a length of seven days, but can refer to any block of seven consecutive days. A span is either positive, denoting forward motion of time, or negative, denoting backwards motion in time.

Alternative Names

Duration, interval, time distance.

Discussion

It is already accepted that "interval" denotes an anchored span (−E7). A "duration" is generally considered to be non-directional, i.e., always positive (−E7). The term "time distance" is precise, but is longer (−E2).

3.21 Temporal Expression

Definition

A *temporal expression* is a syntactic construct used in a query that evaluates to a temporal value, i.e., an event, an interval, a span, or a temporal element.

In snapshot databases, expressions evaluate to relations and therefore they may be called relational expressions to differentiate them from temporal expressions. All approaches to temporal databases allow relational expressions. Some only allow relational expressions, and thus they are unisorted. Some allow relational expressions, temporal expressions and also possibly boolean expressions. Such expressions may defined through mutual recursion.

3.22 Time-invariant Attribute

Definition

A *time-invariant attribute* is an attribute whose value is constrained to not change over time. In functional terms, it is a constant-valued function over time.

3.23 Time-varying Attribute

Definition

A *time-varying attribute* is an attribute whose value is not constrained to be constant over time. In other words, it may or may not change over time.

Acknowledgements

We thank Sushil Jajodia and Abdullah Tansel for promoting the glossary. Their comments greatly improved its quality. Michael Soo provided definitions for event, interval, and span.

References

[Sno92] Richard Snodgrass. TSQL: A Design Approach. White paper, University of Arizona, Department of Computer Science, Tucson, April 1992.

[Soo91] M. D. Soo. Bibliography on Temporal Databases. *ACM SIGMOD Record*, 20(1):14–23, Mar 1991.

[TCG+92] A. Tansel, J. Clifford, S. Gadia, S. Jajodia, A. Segev, and R. Snodgrass (eds.). *Temporal Databases: Theory, Design, and Implementation.* Database Systems and Applications Series. Benjamin/Cummings Pub. Co., Redwood City, CA. To appear January 1993.