Temporal Databases
Status and Research Directions

Richard Snodgrass

Department of Computer Science
University of Arizona
Tucson, AZ 85721
rts@cs.arizona.edu

It seems somehow fitting to begin this paper on databases that store historical information with a chronology, touching briefly on all work that I am aware of in this area. I discuss in some detail what I consider to be the ten most important papers and events in terms of their impact on the discipline of temporal databases. These are emphatically not meant to detract from the other excellent papers in temporal databases. My goal is to characterize the evolution of this field, as an introduction to the approximately 350 papers specifically relating time to databases that have appeared thus far. I then identify and discuss areas where more work is needed.

1 Chronology

What follows is a personal, subjective, and necessarily incomplete history of the field over the last 4,000 years. I attempt to identify the major players and research topics. Authors who have worked together are grouped together, and are listed alphabetically, both within and between groups. Specific references to the papers written by these authors may be found in the bibliographies [Bolour et al. 1982, McKenzie 1986, Stam & Snodgrass 1988, Soo 1991]. In fact, this summary may be viewed as an index into the more than 50 pages of bibliographic references. I've surely missed some researchers, for which I sincerely apologize.

1.1 2,000 B.C. to 1969

Hieroglyphics were arguably the first rollback databases[1]. Even today it is possible to roll back these databases several thousand years and look up the (then current) grain harvest. Oral histories, e.g., Homer's Illiad, were early historical databases. The first computerized database (the census of 1890, stored on punched cards) was an historical database, as it admitted update by replacing cards.

As far as I have been able to ascertain, the first academic treatment of time in databases was the 1956 Harvard dissertation by Frederick Brooks, Jr., where the three-dimensional view of an historical database was proposed (the two other dimensions being entities and attributes).

[1] A note on terminology: a *rollback* database supports *transaction time*, recording a history of updates made to the database. Such a database supports rollback to a previous stored state of the database. An *historical* database supports *valid time*, recording a history of the real world, as is currently best known. With an historical database we can query past events and states in the history of the enterprise modeled by the database. A *temporal* database supports both kinds of time, and can thus record, for example, retroactive changes, where the new value became valid in the real world at a time before the new value was recorded in the database. This terminology also applies to individual relations when a DBMS can support multiple kinds of time.

1.2 1970 to 1980

The early 1970's saw several Ph.D.dissertations (Bruetmann, Falkenberg, Sundgren) and other papers (Bubenko) on incorporating time into conceptual data models.

There was activity in areas outside of databases that would later have impact. A. Prior's book on temporal logic (1969) was followed by those by Rescher&Urquhart (1971) and McArthur (1976). Around this time, the artificial intelligence community started considering time storage, retrieval, inferencing, and causality (Bruce, Findler, Goldstein, Gorry&Kahn). The International Society for the Study of Time was formed; unfortunately, its periodic conferences have had little representation from the database community.

◇1◇ [Wiederhold et al. 1975] This paper is notable for several reasons. First, it was the first article on temporal databases to appear in a computer science forum. Second, it described both the data model (a set of entity-attribute-time-value quadruples) and implementation of the first historical DBMS, the Time Oriented Databank. Several subsequent medical information systems have included substantial support for time (Martin, Palley, Pauker).

The first papers on general DBMS facilities for *processes*, which change attributes of entities over time, appeared in the mid-1970's (Bradley, Flory).

◇2◇ [Schueler 1977] This intriguing paper was the first to deal with transaction time (though this term would not be introduced for another 8 years). It advanced the hypothesis that "a very large proportion of system expenses, limitations, and insecurities is directly or indirectly attributable to ... update." Instead, it advocated that *logical* update be implemented not with *physical update* (i.e., changing the bit pattern on a storage medium) but rather with physical *append*, which leaves the original information in place. All of the new data would be put in a memory hierarchy, including the write-once-read-many-times (WORM) storage devices that were just then starting to appear in research labs. This line of thinking reemerged five years later in Copeland's article entitled "What If Mass Storage Were Free?".

◇3◇ [Jones et al. 1979] This paper described the first general historical query language, LEGOL 2.0. Its name derived from the context of the interpretation of rules with temporal concerns that are often found in legislation. LEGOL 2.0 was an implemented algebraic query language that supported temporal joins, temporal selection, and temporal aggregates. It provided an early informal notion of temporal completeness, as most time-oriented query languages subsequently proposed were compared to it.

The late 1970's also saw the introduction of Codd's RM/T data model, which explicitly included events, and Yamami's time series data model. Interestingly, Codd's current definition of the relational model, RM/V2, does not include any time support beyond uninterpreted time constants, ostensibly because DBMS vendors are not ready for such concepts. I'll revisit this point later in this paper.

The DATA system (Ariav&Kimball&Morgan) was the first implementation of a rollback DBMS and the first to use differential files. Its influence, however, was limited, because the only description appeared in an unpublished M.S. thesis.

Finally, the specification of information systems in terms of information flow over time was first studied in the late 1970's (Bruetmann&Mauer, Bubenko, Rolland, Sernadas).

1.3 1980 to 1989

There was continued work in the early 1980's on AI approaches to managing uncertainty and temporal reasoning for planning and problem solving (Allen&Hayes&Kautz&Koomen, Dean&McDermott) and analysis of information in clinical databases, generally to detect and eventually to confirm causality (Blum&Downs&Walker&Wiederhold).

During the early part of this decade, support for time in the E-R model (DeAntonellis&Degli-Antoni&Mauri&Zonta, Klopprogge&Lockemann) and in semantic data models (Hammer&Mc-Leod) surfaced. Work continued in conceptual modeling (Anderson, Ariav, Bubenko&Gustafsson&Karlsson, DeAntonellis&Zonta).

Various aspects of transaction time were explored: hypothetical relations, which are rollback relations with branching transaction time (Ekland&Price, Stonebraker&Woodfill), snapshot relations (Adiba&Lindsay), and optical disk structures (Maier, Rathmann), which was followed by later work (Bulgren&Canas, Easton, Vitter). Overmyer&Stonebraker explored implementing valid time as an abstract data type.

◇4◇ [Clifford & Warren 1983] This paper was a watershed article. The paper provided a formal semantics for both an historical data model and a calculus-based query language, the first of many to be proposed. (Some felt the model was *too* formal because it involved a variant of Montague's complex intensional logic originally formulated in the context of computational linguistics.) It was the first article on time-oriented databases to appear in a major database journal, thereby introducing this community to the topic. Shortly after this article appeared, activity increased significantly.

◇5◇ [Dadam et al. 1984, Lum et al. 1984, Lum et al. 1985] These papers, following the Clifford and Warren paper by one to two years, offered a nice counterbalance. They described various aspects of the Advanced Information Management project in the IBM Heidelberg Scientific Center. The prototype temporal DBMS described in these papers was the first to support both valid and transaction time, and the first to support temporal indexing.

◇6◇ [Stonebraker 1987] Postgres embodies the first concrete implementation proposal for optical disks for rollback relations (Stonebraker imaginatively calls the rollback operation "time travel"). The proposed transaction management and concurrency control algorithms were designed with permanent archiving in mind.

In the mid 1980's, Ahn&Snodgrass showed that transaction time and valid time are truly orthogonal, allowing each to be pursued independently. Versioning, which is concerned with transaction time (again, generally branching time), garnered attention (Blanker&Ijbema, Chang&Katz, Chou&Kim, Dittrich, Lu&Verma, Weikum). Ginsburg defined *object histories* and investigated them in depth with his associates, Dong, Gyssens, Kurtman, Tanaka, Tang, and Tian. Work in the areas of temporal inferencing (Coelha&Cotta&Lee, Karlsson, Sheng) and integrity constraints across transactions (Abiteboul&Vianu, Casanova, Ceri, Ehrich&Gogolla&Lipeck&Saake, Kung, Mark&Roussopoulos, Ngu, Tanabe) continued. Algebraic and calculus-based query languages (and their associated data models) incorporating time also started to appear (Arkun&Tansel, Ariav&Beller&Morgan, BenZvi, Clifford&Croker, Gadia&Yeung, Snodgrass).

◇7◇ [TAIS 1987] This conference was the first devoted to temporal databases. The papers in this conference emphasized conceptual modeling, with individual papers considering most other topics active at the time.

By this point in time (mid-1987), many of the important aspects of time-oriented databases had been addressed, if only in an initial fashion. Both algebraic and calculus-based query languages had been defined, prototype implementations existed, and there were solid results in data modeling and conceptual design.

In the late 1980's, new query languages and their associated data models continued to proliferate (Abbod, Ahmed&Navathe, Bassiouni, Date, Dutta, Johnson&Lorentzos, Kim&Lee&Yoo, McKenzie&Snodgrass, Narasimhalu, Sadaghi&Samson, Sarda). Temporal aspects of deductive databases, related to but distinct from temporal inferencing, were investigated for the first time (Chomicki&Imielinski, Manchandra&Sen&Warren).

The late 1980's also saw a flourish of work in implementation aspects, including rollback databases (Kolovson&Stonebraker, Lomet&Salzberg, Sarda); historical databases (Adiba&Quang, Kalua&Robertson, Kim&Oh, Rotem, Sarda); temporal databases (Ahn&McKenzie&Snodgrass, Kim&Lee&Oh, Thirumalai); time sequences, which are especially appropriate for handling scientific data and music (Kawagoe&Segev&Shoshani, Rubenstein); graphical display (Ahn&Kim&Lee, Shannon&Snodgrass); query optimization (Chaudhuri, Gunadhi&Segev, McKenzie); schema evolution (Ahmed&Martin&Navathe, Banerjee&Chou&Kim&Kim&Korth, King&McLeod, McKenzie&Snodgrass, Roussopoulos&Mark, Skarra&Zdonik); and physical design (Gunadhi&Rotem&Segev).

Finally, conceptual design of the dynamic aspects of information systems received substantial attention in the mid to late 1980's (Barbic&Maiocchi&Pernici, Bubenko&Olive, Clifford&Rao, Deitz, Delcambre&Urban, Horndasch&Schiel&Studer, Kung&Zhenhe, Oberweis&Lausen, Sernadas). This research has concerned designing calendric systems, modeling office procedures (generally by modeling events, transitions between events, and states induced by events), and specifying temporal constraints and event triggering.

1.4 1990

Work continues in the design of temporal databases (Brunet&Cauvet&Lasoudris, Casanova&Furtado, El-Assal&Elmasri&Kouramajian), temporal query languages (Clifford&Tuzhilin, Elmasri, Sarda, Thompson), implementation aspects (Elmasri&Kim&Wuu, Kaefer&Ritter&Schoening), and deductive databases (Chomicki, Kabanza&Stevenne&Wolper).

◇8◇ [Leung & Muntz 1990] This paper very clearly makes the point that conventional query optimization approaches are insufficient for temporal queries, and argues that semantic optimization will be necessary for historical databases. This paper is important as much for its motivation of new optimization techniques as for the specific approaches it introduces.

◇9◇ [Jensen et al. 1990A, Jensen et al. 1990B] These ambitious papers, as well as several others the authors have recently written, propose a general architecture based on stored backlog relations and incremental (and decremental) computation, and discusses a data model, query optimization techniques, and query processing strategies in this context. This paper should spur much research to flesh out the general approach into a fully realized architecture and implementation.

◇10◇ Temporal databases appeared in two undergraduate database textbooks (Date, An Introduction to Database Systems, Volume I, Fifth Edition, and Elmasri&Navathe, Fundamentals of Database Systems). By this imperfect metric, temporal databases have joined the mainstream.

2 Status and Future Directions

For each of the nine general topics discussed below, I give a brief status and cast future directions as unresolved questions. In almost all cases, the answers to these questions will have a theoretical (semantic, formal) component and a practical (implementation) component. As an added challenge, the answers should be parsimonious, efficient, maximally consistent with current approaches, straight forward to implement, and provably correct.

Temporal, calculus-based query languages: About a dozen such languages have been proposed, nearly all being extensions of Quel or SQL. These proposals vary widely in their underlying data model, their comprehensiveness (ranging from support only of basic queries to full support of querying, update, aggregation, indeterminacy, schema evolution, and evaluation via an underlying algebra), and their definitional completeness (ranging from a brief prose description to a full formal tuple calculus semantics.) Despite this disparity, there does seem to be some consensus arising. What constructs are most appropriate in a temporal query language? How can time be successfully incorporated into form-based and high resolution graphics database user interfaces? How can relative time, e.g., "15 minutes", "one business day", be supported in a query language? Several standards for temporal completeness have been advocated, but none has attained the acceptance granted Codd's metric for conventional completeness (which also has its detractors). What is a suitable standard for temporal completeness for calculus-based query languages?

Temporal algebras: One obvious way to implement a temporal calculus-based query language is to use a temporal algebra. At last count, over a dozen algebras supporting some notion of time had been defined. Which algebra should a temporal DBMS be based on? Ultimately, the answer will depend on how the evaluator of algebraic expressions interacts with the other components of the DBMS: its storage structures (e.g., the data models of some algebras assume first normal form, while that of others assumes sets of attribute values or of timestamps), its optimization strategies (e.g., traditional relational algebraic tautologies are often but not always retained), and even its basic architecture (e.g., Jensen's and McKenzie's algebras allow incremental evaluation, which most architectures are unable to exploit).

Transaction time: Some DBMSs support versioning, and all support some form of schema evolution, yet these aspects are generally approached in isolation. There have been few attempts to support both in a comprehensive and efficient manner. How can schema evolution and versioning be integrated to fully support transaction time? Do incremental update algorithms proposed for schema evolution help or hinder versioning? How can versioning of complex objects be implemented? How can support for optical disks (both WORM and erasable) be smoothly and efficiently incorporated into the DBMS? Are other points in the spectrum between conventional relations, that store only the current state, and rollback and temporal relations, that follow a pure append-only protocol, relevant? If so, how should partial deletion be specified and implemented?

Conceptual and physical database design: Much has been written about the first topic, and little about the second. All that has been done has considered valid time. The related work concerning transaction time, specifying integrity constraints by considering sequences of transactions, has a stronger formal basis. How may conceptual database design, physical database design, and transaction semantics be integrated into a coherent strategy for temporal database design? A few temporal normal forms have been introduced, without wide acceptance. How may conventional normal forms be extended to accommodate evolution over time? Can appropriate normal forms aid in schema design? How can different assumptions concerning time (e.g., a bank day that ends at 3pm, fiscal years beginning at different dates) and varying time granularities be accommodated during schema integration and within federated databases?

Concurrency control and recovery: Schueler's thesis that replacing modification in place with append will simplify the algorithms in a DBMS remains speculative, in part because concurrent users and the possibility of transaction abort still imply that modifications must be supported. Postgres' transaction management seems to be a reasonable initial approach, though detailed performance studies have yet to be done. Does the ability to perform rollback eliminate a major advantage of multi-version concurrency control: the ability to execute non-interfering read-only queries? Is locking still the method of choice when transaction time is supported? Most changes to the database occur soon after a change occurs in reality. As we evolve towards a paperless office, where a change such as giving a salary raise is administratively put into effect by the act of recording the change in the database, this correspondence between transaction and valid time will strengthen. It is therefore likely that supporting valid time will involve changes to concurrency control and recovery strategies. How does adding valid time impact transaction management?

Object-oriented databases (OODBs): Some OODBMSs include significant support for versioning. Also, schema evolution in OODBs is fairly well understood. However, I'm aware of no work on valid time in OODBs. Is there any application of concepts in historical E-R data models and languages (several of which have been proposed) to OODBs? Is support for valid time even required by applications targeted by OODBs (e.g., CAD, CAM, CASE)?

Temporal Inferencing and deductive databases: How may the significant results concerning temporal inferencing obtained by the AI community be applied to relational, object-oriented, and deductive databases? Only a few papers have thus far appeared on this important topic. Significant semantic and operational problems arise if the database is versioned but the rule base is not. How may the evolution of the rule base (the intensional database), which properly concerns only transaction time, be coordinated with updates to the extensional database?

Real-time databases: In such databases, also called active databases, there is the requirement that queries and updates (both conventional and triggered) be performed within soft or hard deadlines. Timestamping stored data is often necessary to meet these constraints. What language constructs may be used to specify transaction deadlines? How do the system facilities needed to support real-time queries compare and interact with those used to support transaction time? What simplifications result when data is stored immediately as it occurs, resulting in identical valid and transaction timestamps for that data?

Commercial viability: There are many justifications voiced as to why commercial DBMSs support neither transaction nor valid time: (a) lack of a standard query language incorporating time (SQL's only time-related constructs are date and time constants, which are widely regarded as poorly designed), (b) inappropriate data structures (the common data structures, hashing, indexed sequential, and B-trees, have been shown to exhibit poor performance; most proposed indices of rollback or historical relations are based on some form of multi-key extensible hashing, which generally are not available in current DBMSs), (c) little interest by users (even though most applications deal with time-varying data), (d) novel storage devices are required (optical storage devices are a natural for rollback and temporal data, yet current DBMSs do not yet support such devices), and (e) timestamps are necessarily application specific, and thus should not be specially handled by the database (this argument applies equally to other features being proposed for inclusion into databases, such as rules, spatial information, methods, and abstract data types). While some of the impediments are non-technical, researchers can ease the transition to commercial temporal databases. Can temporal extensions be defined for SQL that represent an incremental change and thus do not invalidate other, existing parts of the language? How may the different notions of time prevalent in disparate industries such as banking, automated manufacturing, and insurance be accommodated in a general framework supported by the DBMS? What optimization strategies are appropriate for temporal queries assuming conventional access methods?

Clearly there is much work to be done in this area. The excitement continues.

3 Bibliography

[Bolour et al. 1982] Bolour, A., T.L. Anderson, L.J. Dekeyser and H.K.T. Wong. *The Role of Time in Information Processing: A Survey. SigArt Newsletter*, 80, Apr. 1982, pp. 28-48.

[Clifford & Warren 1983] Clifford, J. and D.S. Warren. *Formal Semantics for Time in Databases. ACM Transactions on Database Systems*, 8, No. 2, June 1983, pp. 214-254.

[Dadam et al. 1984] Dadam, P., V. Lum and H.-D. Werner. *Integration of Time Versions into a Relational Database System*, in *Proceedings of the Conference on Very Large Databases.* Ed. U. Dayal, G. Schlageter and L.H. Seng. Singapore: Aug. 1984, pp. 509-522.

[Jensen et al. 1990A] Jensen, C.S., L. Mark and N. Roussopoulos. *Incremental Implementation Model for Relational Databases with Transaction Time. Trans. Knowledge and Database Systems*, 1990, to appear.

[Jensen et al. 1990B] Jensen, C.S., L. Mark, N. Roussopoulos and T. Sellis. *Using Caching, Chache Indexing, and Differential Techniques to Efficiently Support Transaction Time.* Technical Report CS-TR-2413. Department of Computer Science. University of Maryland. Feb. 1990.

[Jones et al. 1979] Jones, S., P. Mason and R. Stamper. *LEGOL 2.0: A Relational Specification Language for Complex Rules. Information Systems*, 4, No. 4, Nov. 1979, pp. 293-305.

[Leung & Muntz 1990] Leung, T.Y.C. and R.R. Muntz. *Query Processing for Temporal Databases*, in *Proceedings of International Conference on Data Engineering.* IEEE. Los Angeles, CA: Computer Society Press, Feb. 1990, pp. 200-208.

[Lum et al. 1984] Lum, V., P. Dadam, R. Erbe, J. Guenauer, P. Pistor, G. Walch, H. Werner and J. Woodfill. *Designing DBMS Support for the Temporal Dimension*, in *Proceedings of ACM SIGMOD International Conference on Management of Data.* Ed. B Yormark. Association for Computing Machinery. Boston, MA: June 1984, pp. 115-130.

[Lum et al. 1985] Lum, V., P. Dadam, R. Erbe, J. Guenauer, P. Pistor, G. Walch, H. Werner and J. Woodfill. *Design of an Integrated DBMS to Support Advanced Applications*, in *Proceedings of the Conference on Foundation of Data Organization.* Kyoto, Japan: May 1985.

[McKenzie 1986] McKenzie, E. *Bibliography: Temporal Databases. ACM SIGMOD Record*, 15, No. 4, Dec. 1986, pp. 40-52.

[Schueler 1977] Schueler, B. *Update Reconsidered*, in *Architecture and Models in Data Base Management Systems.* Ed. G. M. Nijssen. North Holland Publishing Co., 1977.

[Soo 1991] Soo, M. *Bibliography on Temporal Databases. ACM SIGMOD Record*, Mar. 1991, to appear.

[Stam & Snodgrass 1988] Stam, R. and R. Snodgrass. *A Bibliography on Temporal Databases. Database Engineering*, 7, No. 4, Dec. 1988, pp. 231-239.

[Stonebraker 1987] Stonebraker, M. *The Design of the POSTGRES Storage System*, in *Proceedings of the Conference on Very Large Databases.* Ed. P. Hammersley. Brighton, England: Sep. 1987, pp. 289-300.

[TAIS 1987] *Proceedings of the Conference on Temporal Aspects in Information Systems.* AFCET. May 1987.

[Wiederhold et al. 1975] Wiederhold, G., J.F. Fries and S. Weyl. *Structured Organization of Clinical Data Bases*, in *Proceedings of the AFIPS National Computer Conference.* AFIPS. 1975, pp. 479-485.