On An Algebra For Historical Relational Databases. Two Views

James Clifford
Computer Applications and Information Systems
Graduate School of Business Administration
New York University

Abdullah Uz Tansel
Baruch College
Statistics and Computer Information Systems
City University of New York

Abstract. In search of the appropriate semantics for the inclusion of structures and operations that will meet the needs of a wide class of users interested in a database system supporting temporal views of their data, the paper includes both a discussion of many problems that must be addressed, and a proposal for an extended relational algebra The informal discussion of issues motivates the proposal to time-stamp the attributes of relations, rather than the tuples The proposed algebra incorporates this view with a general treatment of non-first-normal-form relations

Note Parts I and II were written and submitted independently by their respective authors to the 1985 SIGMOD Conference, they have been combined at the invitation of the 1985 SIGMOD Program Committee The names of the authors are listed in alphabetical order

1 Introduction

A relational database is defined as a set of time- varying relations of assorted degrees [10] However, both relational theory and current relational databases model only the most recent snapshot of the real world As events take place, the real world changes state and new values are incorporated into the database to form the most recent snapshot Thus, for an object, the database keeps the current values of its attributes As soon as a new value is assigned to an attribute, its previous value is erased Obviously, in order to retain complete information about the object, the current values of its attributes, as well as their histories, should be stored and managed by the DBMS

Recently there have been numerous proposals aimed at correcting this deficiency in existing data models. This paper presents the viewpoints of two researchers who are addressing the problem of augmenting the relational model in such a way as to allow a consistent extension of the relational algebra for historical relations A primary intersection in their approaches is the proposal that the most perspicuous place to incorporate the temporal dimension into the relational model is at the attribute level, rather than at the tuple level as has generally been proposed. Thus attributes in the proposed extension have domains which are either simple, as in traditional relations, or are complex values incorporating some form of time-stamping The exact nature of these complex domains differs in the two proposals, Clifford treats these domains as functions from points in time to simple values, while Tansel treats them as functions from time intervals to simple values In either case both recognize that this moves the resulting relations out of the class of first-normal form relations Tansel exploits this fact by developing a relational algebra that allows non-first-normal-form domains (whose elements can be sets), and treats his temporal domains as just a special case

2 Previous work

Although there are a number of researchers active in the area of temporal database models ([6] and [2] respectively survey the literature and delineate the major research areas) to date there has been no really comprehensive treatment of an operational view of these databases There have, however, been some attempts in this direction In the Time Relational Model (TRM) of [5] a temporal dimension is added to ordinary relations, in a manner similar to the HDBM Then an algebra for TRM is defined which, although internally consistent, is extremely limited Essentially, all the operators on time-relations (3-dimensional) specify a "Time-View" which "extracts a regular (two-dimensional) relation out of the Time-Relation " After this "one-shot" reference to the temporal dimension, the user must then work on this "view"

within the ordinary relational model A true time-relational algebra, which would stay within the the space of time relations, is not defined

In [3] an extension to the query language SQL was defined, in an effort to incorporate temporal elements in a way that would not penalize users not interested in access to the historical data Recent work in [22] reports on a similar extension to the query language QUEL. One problem with this work is that its model contains two different types of objects, event relations and interval relations This unduly complicates the model, particularly the definitions of the query language Neither of these papers addresses the formulation of an historical algebra, so that many operational issues that the algebra forces you to consider are not treated An unpublished manuscript [4] looks at the issue of an algebra of historical relations, but because of certain assumptions built into their structure, have run into a number of problems For instance, they have been unable to define a join that is symmetric (where A|X| B = B |X| A). [19] addresses the issue of the underlying data structures to support relational operations, proposing time-stamping of tuples and keeping current tuples in a relation History tuples "belonging" to a current tuple are chained together in reverse time order Dadan, Lum, and Weiner continue to explore implementation strategies for this model [12]

In all of these studies, relations are viewed as 3-dimensional cubes, the third dimension being the time. The 3-dimensional structure is converted to relational tables (2-dimensional) by time-stamping the tuples. [7], however, was the first to suggest a different view, one closer to the view of the intensional logic which provided the original impetus for the model in [8] and [9] In this view time is incorporated into the relational model, not at the level of the tuples, but at the level of the attributes In independently attempting to define an operational view of relational HDBs, both authors of this paper have become convinced of the correctness of this treatment of time in historical relational databases

The paper is organized as follows Part I represents the views of James Clifford, and presents a discussion of the semantic issues that must be considered in attempting to define a consistent, complete, and correct extension of the relational model that incorporates a temporal dimension This discussion is illustrated with numerous examples and should convince the reader of the subtle difficulties encountered when considering a formalization of our intuitive notions of time Part II represents the views of Abdullah Tansel, and presents a formal definition of an extended relational algebra that not only incorporates a temporal dimension, addressing many of the issues raised in Part I, but at the same time incorporates a general treatment of non-first-normal form domains.

3 PART I Towards An Algebra of Historical Relational Databases

James Clifford

In a previous paper [9] we presented the Historical Database Model (HDBM), a theory of the semantics of an extended relational database model having time as a fundamental organizing principle The present paper continues to explore such an HDBM, but from an operational perspective, in contrast to the denotational semantic view of the previous work. Whereas we previously defined a logical model theory for an HDB, we are here interested in defining a relational algebra for HDBs This has proven more difficult than we imagined, not because of technical difficulties, but because of subtle semantic nuances caused by the introduction of time into the model We thought that it might prove more informative, not to present a formal historical relational algebra in full detail, but rather to explore some of these issues, as they arise in our work and in the work of others in the area, and discuss notions relating to the correctness and completeness of such an algebra By proceeding in this fashion we may open the door to more discussion of the desired properties and functionality of an HDBM before, perhaps prematurely, becoming bound to a fixed system definition

There are a number of goals that we believe can provide some perspective in our search for the "right" model

1. It should, if at all possible, be a consistent extension of the traditional relational model Thus, flat relations and the familiar relational algebra should be treated as a special case of the historical relational model.

2 A corollary to this point, but one that bears emphasizing, is that the proposed historical relational algebra be, in fact, an algebra This is clearly one of the basic properties of the relational model, and one which we must be maintained

3 Less formalizable, but no less important, is the goal of semantic completeness, the model should be so natural as to accord with people's intuitive views of time and information over time, and so powerful as to allow the extraction of the temporal

information that it contains In this second sense it should be able to serve as a standard for defining the notion of "historical relational completeness," comparable to Codd's notion of "relational completeness "

4. It must adequately address the issue of succinctness, or minimality, with respect to the temporal dimension, i e , the structures seen by the user should contain only information at points in time when users record a change, yet the operations should allow the automatic inference of values at other time points "covered" by the database

5 Given the pervasiveness of the three-dimensional spatial metaphor for historical databases (the database as "cubes"), the model should if possible maintain this view at the external user's level

This part of the paper is organized as follows In Section 2 we discuss briefly some related work in the area Section 3 presents salient features of the model that we have developed so far, along with a discussion of the reasons for some of the decisions we have made Section 4 then discusses many of the subtleties inherent in a temporally-oriented database, in the context of developing extensions to the basic relational operations (project, select, and join) and two new operations, "time-slice" and "When " Finally in Section 5 we discuss directions for future research A running example throughout the text is a film-lover's database (Appendix, with apologies for errors, omissions and fabrications) significantly more rich in temporal information than many examples that stand behind some model proposals

In discussing the issue of time and databases, we not infrequently encounter a "so what?" attitude isn't it obvious that if you need to, you time-stamp your data Two points can be made in response to this attitude First, a point we have made before, none of the three great data models provides any mechanism at all for organizing temporally-oriented data It is the purpose of this paper to present the second point, namely, that time is something so taken for granted that its exact nature is highly elusive Thus, while it is not technically difficult to come up with a consistent model having various algebraic operations defined, intuitively it is far from obvious which operations are appropriate, meaningful, and correct As Augustine of Hippo long ago observed "What, then, is time? If no one asks me, I know, but, if I want to explain it to a questioner, I do not know " (Confessiones XI, XIV)

4 The HDB model

4 1 "What, then, is time?"

Perhaps the best place to start in an enterprise aimed at adding time into a database model is in an examination of this new object, time What kinds of objects are we going to allow users to treat as members of the set of times, and what properties will this set have? These are basic questions to any historical database model. About one property there has been little dissension· the set Time is a linear order, i e., for any two times t_1 and t_2, either t_1 equals t_2, t_1 is-less-than t_2, or t_2 is-less-than t_1 Indeed this ordering is perhaps time's essential property As to the members of the set, there has been less agreement In [9] it was treated as dense, essentially isomorphic to the set of reals For two reasons, we now prefer to treat time as discrete, and isomorphic to the natural numbers First, it is clear that any recording instrument must have at best a finite sampling quantum, and second, any practical domain (or language) that we might define for time attributes in an HDB would have at most a countably infinite set of names for time moments or time intervals Thus while it may be philosophically or theoretically interesting to consider a continuum of moments of time, from a practical standpoint the natural numbers seem a more useful candidate for modelling the properties of database time

A few other general properties of the set Time will emerge as we discuss some of the operations, but in general we adopt the position that the model should say as little as possible about Time except for properties that are essential to its proper use The specific elements of Time are best left for the user to define Others, for example [1], have adopted a somewhat different view, and have explored various additional properties of calendar systems, etc

4 2 Where does time fit into the model?

Given some structure for the set Time, which for the moment we will take to be <T,BEFORE>, where T is some countable set and BEFORE a linear order on T, the question of how best to fit this structure into the relational model naturally arises In most of the work to date, (e g , [16], [9], [22], [3], [5]) some form of tuple time-stamping has been adopted In attempting to define an operational view of HDBs we have become convinced of the correctness of the view we first proposed in in [7], which treats time as a component of the attributes rather than the tuples

Consider the following relation EREL, where each tuple is time-stamped, and consider how we might

define the "projection" operator for such an organization.

EREL (EMP	STATE	SAL	DEPT)
Peter	1 1.80	30K	Shoe
Peter	4 6.80	32K	Shoe
Margi	6 3.78	30K	Shoe
Margi	1 1 79	31K	Shoe
Margi	4 6 80	32K	Shoe
Jack	4 7 79	30K	Linen
Jack	4 12 80	30K	Shoe

Two possibilities readily suggest themselves For example, consider the projection of this relation onto the SAL attribute If we simply project out this column, we get the two-dimensional relation on the left, if we want to remain within the space of historical relations, and project both the STATE attribute (by fiat) and the specified attribute (SAL), we get the historical relation on the right

(SAL)	(STATE SAL)
30K	1 1 80 30K
32K	4 6 80 32K
31K	6 3 78 30K
	1 1 79 31K
	4 7 79 30K
	4 12.80 30K

The first method is clearly worse, we have no idea when these amounts were earned, if they are all current salaries (no), if they are related in some way (yes, but how?), etc The second technique is considerably better More information is present (although one of the two duplicate tuples <4 6 80,32K> was eliminated in making the result a set), but we still are not given any relationships among the tuples We have lost what, in intensional logical terms, are called the "individual concepts" (ICs) -- the salary "of" Margi, the salary "of" Peter, etc. These ICs are functions from times to dollar amounts, and projecting in this fashion loses the functions, or the "of-ness" of salaries Non-key attributes are always related in this "of" way to the "object" given by the key value The "pure" relational model did not have to deal with this, essentially semantic, point, although the RM/T extension [11], and other semantic extensions to the RM have had to consider it (e g , [21], or [14].) It appears that a proper treatment of HDBs must do likewise (Note that a projection onto DEPT might seem to avoid this issue, after all, {<Shoe>,<Linen>} appears to be a reasonable result But again, we have lost the "of-ness" of the use of the DEPT attribute in EREL, if we want information about departments, we should probably be examining a different relation, one "about" departments

There are other compelling reasons for considering STATE as a component of the attributes, including the following

1 different attributes may be measurable/recordable at different rates (days, months, seconds,),

2 some attributes are inherently not time-varying (e g , BIOLOGICAL-GENDER), and should not be encumbered with a tuple's time stamp,

3 attributes vary over time in different ways (how many is open question), e g continuous functions (e g , TEMPERATURE), aggregates over an interval (e g , SALES-VOLUME), and step functions (e g , MANAGER),

4 when a change occurs, it is generally to the value of an individual attribute, not to the values of all of the attributes in a tuple

There are at least two important ramifications of this view. The first is that relations in our model are no longer in First Normal Form, since the domain for time-varying attributes is non-simple. (However, the structure of domains is precise, highly constrained, and exploited by the algebraic operations) The second is that with attributes differing along various time-related dimensions, it becomes necessary to define some underlying "basic" view of time for the database This is necessary in order for the enterprise to be modelled with a consistent view of time, it becomes crucial when we consider, as in the join operation, the interaction between relations

5 Overview of the model

In the model that we have developed, an historical database consists of a collection of historical relations, each one defined over some interval of time (typically, but not necessarily, beginning at some point t and continuing to the present, or NOW) (The examples in the Appendix will be referred to as we present the model and issues of time)

The set Time is a set of times, $\{t_1, t_2, \quad\}$ that is at most countably infinite, with a linear order BEFORE, i e BEFORE = $\{ <t_i, t_j> \mid t_i$ is before $t_j >\}$ This set serves as the underlying domain of times for the entire database

In order to provide a full treatment of time, we have found it necessary to distinguish between three different kinds of attributes

1 Constant attributes (CA), such as

BLOOD_TYPE in DIRECTORS, which are time invariant and hence have simple domains

2 Time-varying attributes (TVA), such as STUDIO in DIRECTORS, which can potentially vary over time, the domains of these attributes are functions from Time to some simple domain

3 Temporal attributes (TA), such as YEAR in FILMS, whose domain is Time.

These are called the temporal type of an attribute The notion of CAs is of course related to the notion of the relation key all attributes in a relation key must be CAs, the other non-key attributes can be of any temporal type Also, it is important to note that the temporal type of an attribute is dependent upon the particular relation in which it appears (for instance, STUDIO in DIRECTORS is a TVA, but in STUDIOS it is a CA) We also point out that the inclusion of TAs (as in the relation FILMS) provides a solution to the problems addressed, e g in the use of two different types of relations, event relations and interval relations, in [22] We believe that this greatly simplifies the model and its operations

In modelling objects and their properties over time, an essential property to consider is the lifespan of an object, i e , the period of time during which the object and its properties are being modelled in the system Every historical database system must address this fundamental issue ([16], [9]) In our model we assume that each relation has an associated lifespan which is a complete segment [START,END] of the set TIME, END is typically the present moment, NOW As in [9] (the Comprehension Principle), we assume that a base relation R has complete information about the "objects" that R models over its lifespan Since objects can freely move into and out of our scope of interest (employees are hired, fired, rehired, etc), we must also provide for modelling the lifespan's of individual objects. While there are several possible solutions to this problem, for the moment we are inclined to utilize a "does not exist" null value Moreover, there is strong reason to believe that a "value unknown" null will be needed, since it is highly likely that in historical databases incomplete information, especially about the past, will abound Accordingly our model uses three different null values

1 $NULL_1$ value of TVA becomes unknown at this point in time

2 $NULL_2$ value of TVA becomes non-existent at this point in time

3 $NULL_3$ value of TVA is unknown at any points in time

The two-dimensional relational model provides two basic operations for reducing relations along each dimension select along the "object" (or tuple) dimension, and project along the attribute dimension. The third basic operation is the join, for combining two relations It should come as no surprise, therefore, that in extending this model to three-dimensional relations, we will need, in addition to these operations, a reducing operation, which we call time slice, for the new third dimension Most of the fundamental problems in building an historical model can be illustrated by examining potential definitions for these operations

5 1 Projection

On first glance, projection(π) appears easy to extend, after all, we have not really altered the attribute dimension of our model So, for instance,

$\pi_{NAME,STUDIO}$ (DIRECTORS) yields the relation

(NAME	STUDIO)
Sternberg	1924 --> MGM
	1926 --> Paramount
	1935 --> Columbia
	1938 --> MGM
	1952 --> $NULL_1$
Cukor	1930 --> Paramount
	1932 --> RKO
	1939 --> MGM
	1950 --> $NULL_1$
Hitchcock	1927 --> Brit Intl
	1934 --> Gaumont
	1938 --> Gainsborough
	1939 --> RKO
	1951 --> Warner Br
	1954 --> Paramount
	1959 --> MGM
	1960 --> Paramount
	1962 --> Universal
	1964 --> $NULL_1$

However, consider the following projection, which discards all of the TVAs in its operand

$\pi_{NAME,BLOOD\_TYPE}$ (DIRECTORS) yields the relation

(NAME	BLOOD_TYPE)
Sternberg	A
Cukor	O
Hitchcock	AB

Although this looks satisfactory, we must consider the question whether it is any longer an historical

relation If it is not, than we have abandoned our goal of defining an algebra, and if it is, then we will have to be especially careful in defining our other operations For example, what would be the result of taking a TIME-SLICE of this relation at the time 1934? or at the time NOW? This issue, of nailing down our definition of the structures in an historical relational model, will be addressed again when we consider some other operations

5 2 Selection

Since we have expanded our notion of domains to include both simple domains, and structured domains (functions from TIME to a simple domain), we can expect that the definition of select (σ), which reduces along the value dimension, will be significantly affected In fact, although most people have felt that JOIN was the difficult operation, most of the problems that arise in defining an historical relational model can be illustrated in considering σ, which will be easier since we can look at only one relation (Since σ can simulate a JOIN, it is no surprise that both operations address similar problems) We will consider seven examples

Select Example 1 Select on CA

σ(NAME = Hepburn) (STARS)

(NAME	DIRECTOR	BLOOD_TYPE)
Hepburn	1932 --> Cukor	A
	1938 --> Hawks	
	1940 --> Cukor	
	1953 --> NULL	

Select Example 2 Select on TA

σ(YEAR = 1935) (FILMS)

(FILM	STUDIO	YEAR)
The Devil is a Woman	Paramount	1935
Sylvia Scarlett	RKO	1935

Since both CAs and TAs are simple domains, they require no change to the standard definition of σ TVAs, however, present special problems There possible values can be given by the user as the selection criteria the value of a TVA at a specified time (Exs 3, 4 & 5), the value of a TVA for some time (Ex 6), or (rarely), the value of the TVA for the object's entire lifespan (Ex 7)

Select Example 3 Select on TVA,
Member of Function Given (1)

σ(STUDIO(1937) = MGM) (LAWYERS)

(LAWYER	STUDIO	SALARY)
Howell	1924 --> MGM	1924 --> 30K
	1930 --> Paramount	1925 --> 35K
	1937 --> MGM	1937 --> 40K
	1940 --> $NULL_1$	1940 --> $NULL_1$

In this example there are no difficulties, Howell is the only lawyer working for MGM in 1937, and both of his TVAs are explicitly defined for this time

Select Example 4 Select on TVA,
Member of Function Given (2)

σ(STUDIO(1925) = MGM) (LAWYERS)

(LAWYER	STUDIO	SALARY)
Howell	1924 --> MGM	1924 --> 30K
	1930 --> Paramount	1925 --> 35K
	1937 --> MGM	1937 --> 40K
	1940 --> $NULL_1$	1940 --> $NULL_1$

This example highlights the need for an interpolation function for TVAs, called a Continuity Assumption in [9] Users must be able to query the database at will with respect to time points or periods, and yet the database cannot possibly store values for every attribute at every point in time Thus, each attribute must have an associated interpolation function, so that the database system can reconstruct an entire time series over the lifespan of each object from the partial specification stored As discussed in [9], a very common interpolation function interprets a partial specification as a step function

Select Example 5 Select on TVA,
Members of Function Given (3)

σ(STUDIO([1925,1940]) = MGM) (LAWYERS)

(LAWYER	STUDIO	SALARY)
	Ø	

In this example a range of times is specified, only those lawyers who worked for MGM throughout the interval [1925,1940] are requested, and in this instance none fit the bill

Select Example 6 Select on TVA,
Value from Range Given

σ(STUDIO = Warner Br) (LAWYERS)

(LAWYER	STUDIO	SALARY)
Rosen	1912 --> Universal	1945 --> 70K
	1923 --> Warner Br	1953 --> $NULL_1$
	1930 --> $NULL_1$	
	1945 --> RKO	
	1953 --> $NULL_1$	

McManus	1923 --> Warner Br	1923 --> 35K
	1930 --> $NULL_1$	1926 --> 40K
		1930 --> $NULL_1$

In this example the user can ask for all tuples that have the value of Warner Br for any point in time in their STUDIO function, it mirrors an existential quantifier over times

Select Example 7 Select on TVA, Entire Function Given

σ (STUDIO = 1923 --> Warner Br
1930 --> $NULL_1$) (LAWYERS)

(LAWYER	STUDIO	SALARY)
McManus	1923 --> Warner Br	1923 --> 35K
	1930 --> $NULL_1$	1926 --> 40K
		1930 --> $NULL_1$

For completeness, a σ completely analogous to the traditional σ can be defined, where the user gives the entire value for a TVA and asks for all tuples having this value This flavor would be extremely rare, and syntactically quite messy Note that it is the only flavor involving TVAs which could be defined without reference to the function structure of the TVA domain

5 3 Time Slice

In a sense Time Slice (τ) is a kind of σ, in which a value from the domain of a TVA is given However, it is more general in that it allows the selection across all of the attributes in the relation. It is for this reason that it deserves treatment as an independent operator across a third dimension Again, some examples will illustrate the problems in defining an appropriate semantics

TIME SLICE Example 1 A Point in Time (1)

$\tau_{(1938)}$ (DIRECTORS)

(NAME	STUDIO	BLOOD_TYPE)
Sternberg	1938 --> MGM	A
Cukor	1938 --> MGM	O
Hitchcock	1938 --> Gainsborough	AB

As pointed out earlier in the discussion of Projection, there is a problem with reducing historical relations to flat ones. The Continuity Assumption and Comprehension Principle were defined to allow us to simulate three-dimensional relations, information complete over some time interval, out of much simpler stored relations, we must not fall into the trap of defining operators that subvert this end Specifically, this relation is less information-bearing than the base relation from which it is derived, we must not allow operations to apply the interpolation function of the base relation and incorrectly infer, for example, that Hitchcock worked for Gainsborough in 1939 This issue has been overlooked in any of the system proposals to date Our current solution is to attach a lifespan to each relation, and to have the operands of the algebraic operations be relations and their lifespans The lifespan of a relation is simply an interval $[t_1,t_2]$ over which, as in the Comprehension Principle, the relation is assumed to be completely defined Thus the result of an operation is always a relation with a (possibly new) lifespan Interpolation functions can be applied only over the lifespan of a relation, in this case the lifespan of the result is [1938,1938], and no problem need arise

TIME SLICE Example 2 A Point in Time (2)

$\tau_{(1927)}$ (DIRECTORS)

(NAME	STUDIO	BLOOD_TYPE)
Sternberg	1927 --> Paramount	A
Cukor	1927 --> $NULL_1$	O
Hitchcock	1927 --> Brit Intl	AB

TIME SLICE Example 3 An Interval of Time

$\tau_{([1923,1935])}$ (LAWYERS)

(LAWYER	STUDIO	SALARY)
Howell	1924 --> MGM	1924 --> 30K
	1930 --> Paramount	1925 --> 35K
	1935 --> MGM	1935 --> 35K
Rosen	1923 --> Warner Br.	$NULL_3$
	1930 --> $NULL_1$	
McManus	1923 --> Warner Br	1923 --> 35K
	1930 --> $NULL_1$	1926 --> 40K
		1930 --> $NULL_1$

This example highlights the need for understanding the difference between the lifespan of a tuple and of a relation the lifespan of this result relation is, of course, the interval of the time slice, [1923,1935], however, the lifespan of, for instance, the tuple with key Howell, is [1924,1935] Notice also that not all attributes need have values in a tuple (e g , SALARY in Rosen tuple) for the result to be meaningful -- hence the system must be able to generate null values in a result

TIME SLICE Example 4
Interpolation vs Non-interpolation

$\tau_{(1925)}$ (STUDIOS)

(STUDIO	HEAD	NUM_FILMS)
MGM	1925 --> Mayer	1925 --> 10
Paramount	1925 --> Schulberg	1925 --> 12
Warner Br	1925 --> J Warner	$NULL_3$
Universal	1925 --> Laemmle	$NULL_2$

A point that is also overlooked in existing models is that there are different ways of modelling data over time, and these need to be reflected in different types, or properties, of TVAs In this example, the attribute HEAD is interpolatable, and uses a simple step-function interpolation However, the attribute NUM_FILMS is inherently non-interpolatable From the value of NUM_FILMS at a given time point, we cannot infer anything about its value at any other time point Note also that there has to be a built-in semantics of the different nulls (e g , τ of $NULL_3$ should probably give $NULL_3$, but $NULL_2$ is generated for the case of non-interpolatability)

TIME SLICE Example 5
Disaggregatable vs Non-disaggregatable Attrs

$\tau_{(7/12/25)}$ (STUDIOS)

(STUDIO	HEAD	NUM_FILMS)
MGM	7/12/25 --> Mayer	$NULL_2$
Paramount	7/12/25 --> Schulberg	$NULL_2$
Warner Br	7/12/25 --> J Warner	$NULL_2$
Universal	7/12/25 --> Laemmle	$NULL_2$

This final example illustrates the need for flexibility in the user interface, and the problems this engenders Users should be allowed freedom to refer to time in ways that are natural to do so (in this instance, referring to a specific day), and the system should be able to (a) translate between different time representations and (b) know when values associated with larger intervals of time (years, e g) can correctly be associated with smaller intervals of time (days, e g) This requires an additional piece of information associated with an attribute, namely whether or not it is disaggregatable. In this example, the attribute HEAD can be disaggregated, but the attribute NUMFILMS clearly cannot, hence the result as shown.

5 4 JOIN

Most of the problems in understanding the meaning of historical JOINs are not essentially different from the problems we encountered with SELECT and TIME-SLICE, so we will not dwell on them here However, we point out that in joining R1 with R2 the following cases need to be considered

1 R1 and R2 have no attributes in common (Cartesian Product should result)

2 shared attributes are both CAs

3 shared attributes are both TVAs

4 shared attributes are TV and C.

5 shared attributes are TV and T.

One additional issue that arises with the JOIN is the interval over which the join is to be performed Given that the lifespan of R_1 is $I_1 = [t_1,t_2]$ and of R_2, $I_2 = [t_3,t_4]$, there are two possible cases to consider for the lifespan of the result either the intersection or the union of I_1 and I_2 Since both of these can have a reasonable interpretation, it seems appropriate to define both a UNION-JOIN and an INTERSECTION-JOIN

5 5 WHEN

Finally, an additional time-related operator called WHEN (Ω), is introduced to provide a mechanism for naming time values not simply with constants (like 1983) but with expressions (like WHEN A = v in r) This unary operator on relations, unlike the other relational operators, yields as a result a set of times rather than a relation. It is used to form temporal expressions which can serve as components of a τ or σ operation

In the simplest case the result of Ω is a connected interval, as in the first example which yields the times when Cukor headed Paramount

WHEN Example 1: Interval Result

$\Omega_{(STUDIO=Paramount,\ Head=Cukor)}$ (STUDIOS)

$= [1919,1925]$

Note that with this operator a query such as "Who were the lawyers at Warner Br when Cukor headed Paramount" is easily expressed

π_{LAWYER} ($\sigma_{(Studio=Warner\ Br\ )}$
($\tau_{(\Omega_{(STUDIO=Paramount,\ Head=Cukor)}\ (STUDIOS))}$
(LAWYERS)))

In general, however, the result of Ω may be a set of disconnected intervals, as in the query for the times when Hitchcock directed for Paramount

WHEN Example 2 Disconnected Intervals Result

Ω(STUDIO=Paramount, Name=Hitchcock) (DIRECTORS)

= {[1954,1959],[1960,1962]}

This implies that τ and σ must be able to deal with temporal arguments which are sets of intervals, for example, τ must be able to time slice a relation at disjoint intervals, producing null values for the unselected time points

6 Summary

We have presented a discussion of issues and problems related to the subtle interactions of time with the other components of the relational database model These issues must be addressed by any system that hopes to provide a complete range of temporal structures and operations, or, in short, that merits being called an historical database model We have presented the basic properties of a model that we are building to support a temporal view of data, and hope to spur further research into some of the problems we have addressed

Acknowledgements

I would like to acknowledge the insights into many of these problems that have been generated in numerous discussions with my colleague Gadi Ariav and our student Joseph Shiftan

7 Part II An Extension of Relational Algebra to Handle Time in Relational Databases

Abdullah Uz Tansel

In this paper we present a proposal, based on time-stamping attributes, for incorporating time dimension into the relational model Section 8 gives the definitions for incorporating the time dimension into the relational model. The new relational algebra operations are described in section 9 Section 10 lists the identities and the algebraic rules involving the new operations. We conclude the paper by briefly describing the future work we plan to do.

8 TIME DIMENSION IN THE RELATIONAL MODEL

8 1 Time

We will consider time as a set of consecutive, equally-distanced points Figure 1 shows the time axis illustrating t_0, t_1, t_2 now as the discrete time points. Now is the marking symbol for the current time

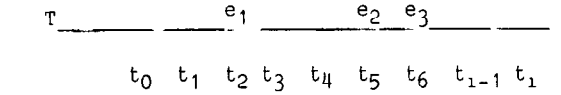

Figure 1 Time Axis

As time advances, the value of now also changes accordingly The time unit is not specified It may be days, hours, minutes, etc Between two consecutive points there is a time duration which is equivalent to one time unit The points are totally ordered and are identified relative to an origin, t_0

$$T = \{t_0, t_1, t_2 \ , t_i, \ , now\}$$

where $t_0 < t_1 < t_2 < \quad t_{i-1} < t_i < t_{i+1} \quad <$ now

$t_i = t_{i-1} + 1$

$t_i = t_0 + i$

The interval [l, u) is a set of consecutive time points between l and u l is the beginning (lower bound) of the interval and u is its end (upper bound), and l<u. $[l,u)=\{t|t\varepsilon T/\backslash l<t<u\}$

The interval [l,u) is closed at the beginning and it is open at the end So, it includes l and the time points up to u but not u Consider two intervals [l,u) and [l',u') They are disjoint if they do not have any points in common Two intervals overlap if they have time points in common, i e their intersection is not null In the case of adjacent intervals, one follows the other In other words l' = u or l = u' The interval [l,u) is a subset of the interval [l',u') if the latter includes all the time points which belong to the former Intervals can be considered as sets So, the set operations, like union, intersection, difference, etc , can be defined on intervals These definitions are not included here, since they are straightforward

8 2 Events

The attributes of an object (i e , an entity or a relationship) assume different values over time The set of values forms the history of that object A data base which maintains object histories is called an historical database (HDB) [9], [17], [19] Changes in the attribute values are triggered by events [13] Events are the transactions, operations, or actions causing the assignment of new values to the attributes of the involved objects Withdrawing money from an account, paying an invoice, and hiring an employee are examples of events Many of these events affect only a few attributes while the remaining attributes retain their existing values These are the update transactions and make up a considerable portion of the activities in many applications This observation forms the rationale for time-stamping the attributes However, insertion events introduce a new object to the database In this case, all the attributes acquire their initial values and have the insertion time as initial time

reference On the other hand, deletion transactions are the events which causes removal of the tuples from the database These tuples usually are archived However, in a HDB, deleted tuples should be kept, otherwise the database would contain incomplete data In our proposal the deleted tuples are kept in the database They are differentiated from the active tuples either by an existance flag and/or null attribute values from the time of deletion to the present A deleted tuple participates in database operations only when the queries refer to the period during which it exists

In Figure 1, e_1, e_2, and e_3 are events They assign the values, say, a_1, a_2, and a_3 to attribute A of an object x, respectively a_1 is assigned at time t_2, and is valid for the time interval $[t_2,t_5)$ In other words, object x has a_1 as the value of attribute A over the points t_2, t_3, t_4 Similarly, x acquires a_2 at time t_5 and retains it until t_6 At t_6, it acquires a_3 which is valid over the interval $[t_6$, now]

The interval $[t_1, t_{1+1})$ contains the point t_1 and the time duration equivalent to one time unit between t_1 and t_{1+1} The value, which is valid at t_1 is also valid over this duration between t_1 and t_{1+1} Any time point between t_1 and t_{1+1} is not visible unless a smaller time unit is used

Each attribute value is recorded as a <time, value> pair The time part of this pair can be taken as the time point at which the attribute value is acquired $<t_1,v>$ or the interval in which the value is valid $<[1,u),v>$ Figure 2 shows these two alternatives for a relation, EMP (NAME, SALARY) Representing time as a point (Figure 2,a) is simple and requires less storage space However, to determine the time duration over which a value is valid, the successor pair has to be examined This creates complications in expressing and interpreting the relational algebra operations Because of this reason, we choose the second alternative which represents the attribute values as triplets A triplet has the form $<[1,u),v>$ where 1 and u are the lower and upper bounds of an interval, respectively in which the value v is valid. [1,u) is the interval component of the triplet and is used as the time-stamp v is its data component. Figure 2 b shows this representation method. Note that 'n' stands for now in Figure 2.b

NAME	SALARY
	10, 10K
	15, 12K
10, BOB	25, 18K
	12, 15K
12, BILL	18, 20K

(a) Time point representation

NAME	SALARY
	<[10,15),10K>
	<[15,25),12K>
<[10,n],BOB>	<[25,n] ,18K>
	<[12,18),15K>
<[12,n],BILL>	<[18,n] ,20K>

(b) Interval representation

Figure 2 Example EMP relation

Any interval which includes now as its upper bound is an expanding interval. As time advances, the interval [1,now] also expands Unlike the other intervals, this is a closed interval and it includes both the lower and upper bounds When an event occurs this interval is split into two intervals, [1,u) and $[1_e$,now] such that $u = 1_e$. Assume s is a tuple of a relation one of whose attributes is A and event e assigns a new value a_e to A Before e occurs s[A] contains <[1,now],a> Afterwards, s[A] contains two triplets, <[1,u),a> and $<[1_e$,now],$a_e>$ Of course, 1_e is equal to the value of now when event e occurred

t_b is the time when an object, o, is introduced to the database All of its attributes has t_b as the initial time-stamp Each of o's attributes have either a non-null value or the value null or a combination of these in the interval $[t_b$,now]. A special integer with appropriate semantics attached to it, is used for representing the value null, denoted by '-' Value null indicates that the attribute value is unknown.

3. The model

Let U be the set of all values regarded as atomic (such as integers, reals, character strings, etc.) and the value null Let D_{a_1} , D_{a_m} be subsets of U and D_{s_1} .,D_{s_m} be subsets of P(U) where P(U) is the power set of U T_I is the set of intervals defined over the time points in T It is a subset of T x T, where x denotes the Cartesian product

$$T_I=\{[1,u)|\ 1 < u\ /\backslash\ t_0 < 1 < now\ /\backslash\ t_0 < u < now\ /\backslash\ 1 < u\ /\backslash\ <1,u> \varepsilon\ T\ X\ T\}$$

D_{t_1}, . ,D_{t_m} are the subsets of T_I X U and $P(D_{t_1})$, ,$P(D_{t_m})$ are their corresponding power sets $P_o(D_{t_1})$, ,$P_o(D_{t_m})$ are the subsets of the former power sets whose elements are ordered with respect to time and have the form

$d=\{<[l_1,u_1),a_1>,<[l_2,u_2),a_2>,$
$,<[l_k,u_k),a_k> \mid$
$l_1<l_2< \quad l_{k-1}<l_k \ /\backslash$
$u_1<u_2< \quad u_{k-1}<u_k \ /\backslash$
$[l_i,u_i) \cap [l_j,u_j) = \emptyset$
for $i=1,2,\ ,k$, $j=1,2,\ ,k$ and $i \neq j$
$/\backslash\ a_i \in U$ for $i=1,2,\ k\}$

Consider a collection of sets E_1, E_2, $,E_n$ where E_i is one of the above defined sets, D_{a_1}, $,D_{a_m},D_{s_1},.\ ,D_{s_m},D_{t_1}$, $D_{t_m},P_o(D_{t_1}),.\ ,P_o(D_{t_m})$ for $i=1,2,\ ,n$ An historical relation (HR), defined on the sets $E_1,E_2,\ E_n$, is a subset of the Cartesian product $E_1 \times E_2 \times . \times E_n$ The value n is called the degree of R and is denoted as deg(R) As is clear from the definition, an HR is a non-first normal form relation whose nesting depth is at most one $R(A_1,A_2,.\ ,A_n)$ is an historical relation scheme Atr(R) denotes the set of attributes of R and is equal to $\{A_1,A_2,\ ,A_n\}$ where $n = deg(R)$.

An historical relation may have four types of attributes Atomic attributes contain atomic values, that is , they receive values from the domains which are subsets of U Triplet-valued attributes contain triplets as atomic values where an atom is a 3-tuple in our context. These attributes quite often correspond to the primary, or foreign key attributes whose values are expected not to change over time. Values of a set-valued attribute are sets of atomic values. These values are considered to be independent of time Set-triplet-valued attributes contain sets of triplets as values Each set is a collection of one or more triplets, defined over the interval $[t_b$, now] and represents the attributes's history The interval component of a triplet is a subset of $[t_b$, now] and its value component corresponds to a value which the attribute has acquired Figure 3 a shows an example HR

r is the historical relation instance (occurance) for the historical relation scheme R Let R,S, be HR schemes, then the historical relational database schema DB is (R,S,) and (r,s,) is its one instance From now on, we will use the term relation to include HR's too

We use the following notation in naming the attributes of a relation. Let A,B, . be attributes of a relation Atomic attributes are referred to by their names, like A,B, etc Set-valued attributes are prefixed with an asterisk, e g , *A,*B, etc We denote triplet-valued attributes by a bar over the attribute name, e g , $\bar{A}$,B, etc Similarly, an asterisk and a bar are used for the set-triplet-valued attributes, e g $*\bar{A}$, *B, etc It is possible to refer to the components of a triplet-valued attribute $\bar{A}_l$, $\bar{A}_u$, and $\bar{A}_v$ represent the lower bound, the upper bound and the data (value) components, respectively of the

EMP relation

E#	ENAME	SALARY	MANAGER	FOR-LANGUAGE
			{<[15,19),RON>,	{FRENCH,
		{<[15,25),14K>,	<[19,22),GARY>,	SPANISH}
133	TOM	<[25,N],18K>]	<[22,N]AL>}	
140	ANN	{<[10,N],12K>}	{<[10,N],LIZ>}	{'-'}

EMP(E#,ENAME,*$\overline{\text{SALARY}}$,*$\overline{\text{MANAGER}}$,*FOR-LANGUAGE)

(a) An example historical relation

E#	ENAME	SALARY	MANAGER	FOR-LANGUAGE
135	TOM	{<[20,25),14K>	{<[20,22),GARY>	{FRENCH,
		<[25,40),18K>}	<[22,40),AL>}	SPANISH}
140	ANN	{<[20,40),12K>}	{<[20,40),LIZ)}	{'-'}

(b) instance of EMP, $emp_{[20,40)}$

Figure 3 Example HR and its instance over the interval [20,40)

triplet-valued attribute $\bar{A}$ Similarly, for a triplet x, x_l, x_u, and x_v represent the lower bound, the upper bound and the data (value) components, respectively

8 4 Defining relations over a time interval

In this section we will define instances of a relation, say R, over the interval [l,u) and at the time point t Let $r_{[l,u]}$ and r_t denote these instances, respectively We will also define tuple components and tuples over [l,u) and at time t Let s be a tuple of r $s[A]_{[l,u)}$ is the tuple component which corresponds to attribute A If A is an atomic attribute $s[A]_{[l,u)}$ is the same as s[A] Similarly, $s[*A]_{[l,u)}$ is the same as s[*A], because these attribute types are independent of time On the other hand, for the triplet-valued attribute $\bar{A}$, $s[\bar{A}]_{[l,u)}$ is equal to a triplet $<[l',u'),a>$ where $[l',u')=[l,u) \cap [t_b,now]$ Of course, when the intersection of intervals is null, $s[\bar{A}]_{[l,u)}$ is also null For the set-triplet-valued attribute $*\bar{A}$

$$s[*\bar{A}]_{[l,u)}=\{<[l',u'),a> \mid x \in s[\bar{A}] \ /\backslash \ [l',u') = [l,u) \cap [x_l,x_u) \neq \emptyset \ /\backslash \ a = x_v\}>$$

The tuple $s_{[l,u)}$ is then, defined as

$$s_{[l,u)} = (s[A_1]_{[l,u)} \circ s[A_2]_{[l,u)} \circ \quad s[A_n]_{[l,u)})$$

where $A_i \in$ Atr(R), $1<i<$ deg(R) and $\circ$ denotes concatenation.

$r_{[1,u)} = \{s_{[1,u)} | s \in r\}$

Time point t can be visualized as the interval [t,t+1) So, r_t is equivalent to $r_{[t,t+1)}$ Tuple components and tuples at time t can analogously be defined Figure 3 b shows EMP relation over the interval [20,40)

9 ALGEBRA OF HISTORICAL RELATIONS

In this section we define standard relational algebra operations as well as the new operations introduced for extending relational algebra Before proceeding with the definitions, some notation will be introduced

In the remainder of the paper we will use the following notation Let R and S be two relations and X and Y be attribute lists, $|X|=|Y|=n\geq 1$, $X \subseteq Atr(R)$ and $Y \subseteq Atr(S)$ XOY denotes $X_1 O Y_1 \wedge X_2 O Y_2 \wedge \ldots \wedge X_n O Y_n$ $Type(X_1)$ denotes the type of attribute X_1, i e , atomic, set-valued, etc , and the underlying domain $Type(X_1)=Type(Y_1)$ indicates that types of attributes X_1 and Y_1, and their underlying domains are the same In other words, X_1 and Y_1 are union compatible Similarly, Type(X)=Type(Y) is the shortened version of $Type(X_1)=Type(Y_1) \wedge \ldots \wedge Type(X_n)=Type(Y_n)$

A_a denotes the set of atomic attributes in R A_s denotes the set of set-valued attributes in R Similarly A_t and A_{st} denote the sets of triplet-valued and set-triplet-valued attributes, respectively

O_n contains the relational comparison operators, $\{=, \neq, >, \geq, <, \leq\}$ O_s contains the set comparison operators, $\{=, \subset, \subseteq, \supset, \supseteq\}$ and Θ_m is the set membership operator, $\{\in\}$

If A is an attribute of relation R, C_A denotes the remaining attributes of R, i e , Atr(R) - {A}

9 1 Standard relational algebra operations

Standard relational algebra operations can directly be applied to HR's with minor modifications Modifications are needed to accomodate new attribute types when attributes are used as operands in formulas Project (π), Cartesian product (x), union (U), set difference (-) operations do not require any modifications On the other hand, selection (σ) has to be modified to handle different attributes types It can be defined as

$\sigma_F(R)=\{t | t \in R \wedge F\}$ where F is a formula It is a conjunction of the terms X O Y and X O v v is a constant It should be consistent with the type of X The attributes X and Y and the comparison operator, O, may take the following forms

a $O \in O_r$ when $X,Y \in A_a$ or $X,Y \in A_t$ or $X \in A_a$, $Y \in A_t$ However, when triplet-valued attributes are referenced, their components should be used, i e X_l, X_u, or X_v For example, $X_l < Y_l \wedge X_u > Y_u \wedge X_v = Y_v$ specifies the condition that the interval part of X includes the interval part of Y and their values are equal

b $O \in O_s$ when $X,Y \in A_s$ or $X,Y \in A_{st}$

c $\Theta \in O_m$ when ($X \in A_a$ or $X \in A_t$) and $Y \in A_s$, or $X \in A_t$ and $Y \in A_{st}$ However, in the former case, if X is a triplet-valued attribute, its components, X_l, X_u, or X_v should be used

Join ($|X|_F$) can be similarly defined The possible forms the formula takes are the same as selection A new type of join operation, natural join by intersection has been defined for the set-valued attributes by [15]

9 2 Aggregate formation operation

Let R be a relation and $X \subseteq Atr(R)$ with $|X| = k$ For the atomic or triplet-valued attribute A of relation R and aggregate function f, aggregate formation operation is defined as

$$R(X,f_A)=\{t[X] \circ y \mid t \in R \wedge y=f_A(\{t' \mid t' \in R \wedge t[X]=t'[X]\})\}$$

and produces a relation of degree k+1 Aggregate formation operation partitions the tuples of R according to the values of X Each partition contains the same X-value Then, it applies function f to A-value of the tuples in each partition X-value and the result of aggregate function forms a tuple of the result relation Naturally, aggregation can be applied to the components of a triplet-valued attribute A too Aggregate formation operation is defined for 1NF relations by Klug [18] and is extended to non-first normal form relations by Ozsoyoglu and Ozsoyoglu [20]

9 3 New operations

New operations are introduced to manipulate historical relations These operations convert one type of attribute to another or form slices of relations Figure 4 shows four of the new operations and their functions The pack operation converts atomic and triplet-valued attributes to set-valued and set-triplet-valued attributes, respectively The unpack operation does the reverse, i e , converts set-valued and set-triplet-valued attributes to atomic and triplet-valued attributes Triplet-decomposition operation breaks

a triplet-valued attribute into its components, lower bound of interval, upper bound of interval, and the value Triplet-formation operation converts three attributes, corresponding to lower and upper bounds of an interval, and a value, into a triplet-valued attribute The slice operation restricts the time of an attribute according to the time of another attribute The drop-time operation discards the time component of triplet-valued and set-triplet-valued attributes

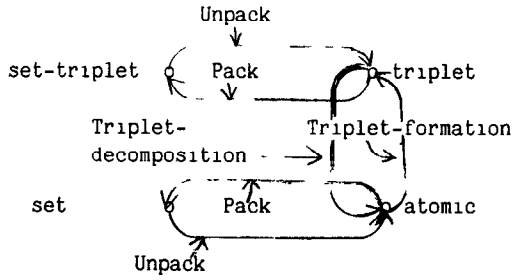

Figure 4 New algebraic operations

9 4 Pack operation (P)

The pack operation, when applied to attribute A of the relation R, collects the values in attribute A into a single tuple component for tuples whose remaining attributes agree [15], [20] Let R be a relation of degree n and A be one of its attributes For each (n-1) tuple in $\pi_{C_A}(R)$, an n-tuple W_g is defined as follows

$$W_g[C_A] = g$$

$$W_g[A] = \begin{cases} \{t[A] \mid t \in R \wedge t[C_A] = g\} & \text{if A is an atomic or triplet-valued attribute,} \\ \{x \mid (\exists t)\ (t \in R \wedge t[C_A] = g \wedge x \in t[A])\} & \text{if A is a set-valued or a set-triplet-valued attribute} \end{cases}$$

then

$$P_A(R) = \{W_g \mid g \in R[C_A]\}$$

Example 1 Consider the following relation, EMP_1

ENAME	SALARY	MANAGER
TOM	<[15,25),14K>	{<[15,19), RON>, <[19,22), GARY>, <[22,n],AL>}
TOM	<[25,n],18K>	{<[15,19), RON>, <[19,22), GARY>, <[22,n],AL>}
ANN	<[10,n],12K>	{<10,n],LIZ>}

$P_{SALARY}(EMP_1)$ produces the result relation EMP_2

ENAME	SALARY	MANAGER
TOM	{<[15,25),14K>, <[25,n],18K>}	{<[15,19), RON>, <[19,22), GARY>, <[22,n],AL>}
ANN	{<[10,n],12K>}	{<[10,n],LIZ>}

9 5 Unpack operation (U)

When applied to attribute A of the relation R, the unpack operation creates a family of tuples, one for each element of the set in the A-component of each tuple [15], [20]. Let t be of a tuple of R

$$U_A(t) = \begin{cases} \{t\} & \text{if A is an atomic or a triplet-valued attribute,} \\ \{t' \mid t'[A] \in t[A] \wedge t'[C_A] = t[C_A]\} & \text{if A is a set-valued or a set-triplet-valued attribute.} \end{cases}$$

Then

$$U_A(R) = \bigcup_{t \in R} U_A(\{t\})$$

Example 2 Consider the relation EMP_3 which is equal to

$$\pi_{ENAME,*SALARY,*MANAGER}(EMP)$$

Note that EMP is given in Figure 3 a. Applying the unpack operation on the SALARY attribute of EMP_3 produces the ralations EMP 4 given below

ENAME	SALARY	MANAGER
TOM	<[15,25),14K>	{<[15,19), RON>, <[19,22),GARY>, <[22,n,],AL>}
TOM	<[25,n],18K>	{<[15,19), RON>, <[19,22),GARY>, <[22,n],AL>}
ANN	<[10,n],12K>	<[10,n],LIZ>

Succesive application of the unpack operation to the attributes in Atr(R) of the relation R converts R into first normal form The order of operations is not significant

$$U_{\{A_1,A_2,\ldots,A_n\}}(R) = U_{A_1}(U_{A_2}(\ldots(U_{A_n}(R))\ldots))$$

Unpacking a tuple according to the attribute *A, produces a family of tuples which differ only in their A-values The rest of the resulting tuples are the same A-value of each tuple is a triplet whereas the other set-triplet-valued attributes still contain the entire history of the original unpacked tuple Thus, duplicate data is created. Successive application of the unpack operation on different set-triplet-valued attributes increases

this redundancy Considering the large size of the historical relations, successive application of the unpack operation may create prohibitively large relations The other attributes may be sliced according to the time of unpacked attribute to avoid this data duplication For this purpose, the slice operation (defined later) can be used

9 6 Triplet-decomposition operation (T-DEC)

This operation breaks a triplet-valued attribute $\bar{A}$ into its components It adds two new attributes to the relation for representing the time interval, one for the lower bound, and one for the upper bound The value component replaces $\bar{A}$ Let R be a relation of degree n, and A be one of its attributes Triplet-decomposition operation creates a new relation whose degree is n+2

$$\text{T-DEC}_{\bar{A}}(R = \{t\ o\ l\ o\ u \mid (\exists t')(t' \in R \wedge t[C_A] = t'[C_{\bar{A}}] \wedge t[A] = t'[\bar{A}_v] \wedge l = t'[\bar{A}_l] \wedge u = t'[\bar{A}_u])\})$$

The new attributes are named $L_{\bar{A}}$ and $U_{\bar{A}}$ $L_{\bar{A}}$ is the $(n+1)^{st}$ attribute and $U_{\bar{A}}$ is the $(n+2)^{nd}$ attribute The following example demonstrates triplet-decomposition

Example 3 Consider the relation EMP_5 which is given by

$U_{*SALARY}\ (\pi_{ENAME,*SALARY}(EMP))$

ENAME	SALARY
TOM	<[15,25),14K>
TOM	<[25,n],18K>
ANN	<[10,n],12K>

$\text{T-DEC}_{SALARY}\ (EMP_5)$ gives EMP_6 as the result

ENAME	SALARY	L_{SALARY}	U_{SALARY}
TOM	14 K	15	25
TOM	18 K	25	n
ANN	12 K	10	n

9 7 Triplet-formation operation (T-FORM)

Triplet-formation operation creates a triplet-valued attribute from the three attributes A,L, and U which correspond to the data-value, lower bound and upper bound components of the triplet, respectively The latter two attributes are needed to form the interval over which the value is valid The resulting triplet-valued attribute, A replaces attribute A and the other two attributes are projected out Let R be a relation of degree n+2 and A,L,U ε Atr(R)

$$\text{T-FORM}_{A,L,U}(R) = \{t \mid (\exists t')(t' \in R \wedge t[C_A] = t'[C_A] \wedge t[A_v] = t'[A] \wedge t[A_l] = t'[L] \wedge t[A_u] = t'[U])\}$$

The resulting relation has degree n

Example 4 Consider the relation EMP_6 created in example 3 $\text{T-FORM}_{SALARY, L_{SALARY}, U_{SALARY}}(EMP_6)$ produces the original relation EMP_5 Note that applying triplet-formation operation to the attributes created by a triplet-decomposition operation produces back the original relation, that is,

$$\text{T-FORM}_{A,L_A,U_A}(\text{T-DEC}_{\bar{A}}(R)) = R$$

9.8. Slice operation (SLICE)

This operation slices the triplets of an attribute according to the time of another attribute Let R be a relation and $\bar{A},\bar{B}$ be two tripled-valued attributes.

$$\begin{aligned}\text{SLICE}_{\bar{A},\bar{B}}(R) = \{t \mid (\exists t')(t' \in R \wedge t[C_{\bar{A}}] = t'[C_{\bar{A}}] \wedge t[\bar{A}_v] = t'[\bar{A}_v] \wedge t[\bar{A}_l] \geq t'[\bar{B}_l] \\ \wedge t[\bar{A}_u] \leq t'[\bar{B}_u] \wedge t[\bar{A}_l] \geq t'[\bar{A}_l] \wedge t[\bar{A}_u] \\ < t'[\bar{A}_u] \wedge (t[\bar{A}_l] = t'[\bar{A}_l] \vee t[\bar{A}_l] = t'[\bar{A}_l]) \\ \wedge (t[\bar{A}_u] = t'[\bar{A}_u] \vee t[\bar{A}_u] = t'[\bar{B}_u])\}\end{aligned}$$

The interval of the first attribute, $\bar{A}$, is sliced according to the interval of the second attribute, B Slice operation checks the interval portions of the triplets of attributes $\bar{A}$ and B to determine whether they overlap or not Let t be a tuple of R Intersection of the time intervals of $t[\bar{A}]$ and t[B] is assigned to the interval component of the new triplet for $\bar{A}$ The new triplet receives its data-value from attribute $\bar{A}$ If the two intervals do not overlap, the tuple is not considered The slice operation can be extended for any combination of triplet-valued and set-triplet-valued attributes, e g $\text{SLICE}_{\bar{A},*\bar{B}}(R)$, $\text{SLICE}_{*\bar{A},B}(R)$, or $\text{SLICE}_{*\bar{A},*\bar{B}}(R)$ These versions can be defined analogously They are not given here to save space The last three versions of the slice operation can also be done by first unpacking the set-triplet-valued attributes and then applying its first version The following example demonstrates the slice operation.

Example 5 Consider the relation EMP_4 which is obtained in example 2 $\text{SLICE}_{*MANAGER,SALARY}(EMP_4)$ slices the MANAGER attribute of EMP_4 according to the SALARY attribute and produces the output relation EMP_7

ENAME	SALARY	MANAGER
TOM	<[15,25), 14K>	{<[15,19), RON>, <[19,22), GARY>, <[22,25), AL>}
TOM	<[25,n], 18K>	{<[25,n], AL>}
ANN	<[10,n], 12K	{<[10,n], LIZ>}

The slice operation can also be used to

determine the values of time dependent attributes of a relation, R, over an interval [1,u) First, R is augmented by a new attribute which contains the desired time interval Then, the slice operation is applied

The following expression slices attribute $\bar{A}$ of R over the interval [1,u)

$$\pi_{Atr(R)}(SLICE_{\bar{A},\bar{I}}(RX\{<[1,u),->\}))$$

where $\bar{I}$ is the new column obtained after forming the Cartesian product That is, $\bar{I}$ is the (deg(R) +1)th attribute The instance of a relation over an interval [1,u) can also be obtained Successive application of the slice operation on the time dependent attributes, $\{A_1,A_2, \ ,A_n\}$ of the relation R returns $r_{[1,u]}$

$r_{[1,u)}=\pi_{Atr(R)}(SLICE_{A_1,I}(SLICE_{A_2,I}(\ (SLICE_{A}$
$SLICE_{\bar{A}_n,\bar{I}}(RX\{<[1,u),->\})....)))$

9.9 Drop-time operation (DROP-TIME)

This operation discards the time components of a triplet-valued or a set-triplet-valued attribute and converts it into an atomic or a set-valued attribute, respectively Let R be a relation and A ε Atr(R)

$$DROP\text{-}TIME_A(R) = \begin{cases} \{t \mid (\exists t')(t' \varepsilon R \wedge t[C_{\bar{A}}] = t'[C_{\bar{A}}] \wedge t[A]=t'[\bar{A}_v])\} \\ \text{if A is a triplet-valued attribute,} \\ \{t \mid (\exists t')(t' \varepsilon R \wedge t[C_A] = t'[C_A] \wedge x \varepsilon t'[\bar{A}] \wedge x_v \varepsilon t[A])\} \text{ if A is a set-triplet-valued attribute,} \\ R \text{ otherwise.} \end{cases}$$

The degree of the resulting relation is the same as the degree of R Drop-time operation allows us to disregard time It is a short-cut to manipulate a relation without considering time Example 6 Consider the relation EMP_4 given in example 2

DROP-TIME$_{*MANAGER}$](EMP_4) produces the following result, EMP_8

ENAME	SALARY	MANAGER
TOM	<[15,25), 14K>	{RON,GARY,AL}
TOM	<[25,n], 18K>	{RON,GARY,AL}
ANN	<[10,n], 12K>	{LIZ}

9.10 Arithmetic expressions

Simple arithmetic expressions which use tuple components as operands are useful tools [20] For example, in calculating the length of an interval, such an expression is needed. Let R be a relation, $X \subseteq Atr(R)$, and g be a simple arithmetic expression involving +,-,*,/ as operators and members of X as operands

$$R(g(X))=\{t[C_X] \circ y \mid t \varepsilon R \wedge y=g(t[X])\}$$

Note that g(t[X]) denotes $g(t[X_1],t[X_2], \ ,t[X_m])$ The degree of the result is k + 1 if $|C_A| = k<deg(R)$. Example 7 Consider the query, "for each employee, find the longest period of time, over which his salary has not changed" The algebra expression for this query is

$$EMP_9 = \Pi_{ENAME,I}(((T\text{-}DEC_{\overline{SALARY}}(U_{\overline{*SALARY}}(EMP)))(U_{\overline{SALARY}} - L_{\overline{SALARY}}))(ENAME,MAX_I))$$

Note that I denotes the column created by the arithmetic expression EMP_9 contains two tuples (TOM,10) and (ANN,20) assuming that n is 30 when the query is processed

9.11 Elementary operations of extended relational algebra

Projection, selection, cartesian product, union and set difference are the basic operations of relational algebra The other operations, like join, aggregate formation, etc can be expressed by these five basic operations In addition to the basic operations of relational algebra, pack, unpack, triplet-formation and triplet-decomposition operations form the elementary operations of extended relational algebra Slice and drop-time operations can be expressed in terms of the elementary operations Let R be a relation and $\bar{A}$, $\bar{B} \varepsilon A_t$ Let L_A,U_A,L_B, and U_B be the columns created by triplet-decomposition operation on attributes $\bar{A}$ and $\bar{B}$

$$SLICE_{\bar{A},\bar{B}}(R) = T\text{-}FORM_{A,L_A,U_A}(T\text{-}FORM_{B,L_B,U_B}($$
$$(\Pi_{C_{AB},A,B,L_B,U_A,L_B,U_B}(\sigma_{L_A \le L_B \wedge U_A \le U_B \wedge L_B < U_A}$$
$$(T\text{-}DEC_{\bar{A}}(T\text{-}DEC_{\bar{B}}(R))))) \cup$$
$$(\Pi_{C_{AB},A,B,L_A,U_B,L_B,U_B}(\sigma_{L_B \le L_A \wedge U_B \le U_A \wedge L_A < U_B}$$
$$(T\text{-}DEC_{\bar{A}}(T\text{-}DEC_{\bar{B}}(R))))) \cup$$
$$(\Pi_{C_{AB},A,B,L_B,U_B,L_B,U_B}(\sigma_{L_A \le U_B \wedge U_B \le U_A}($$
$$T\text{-}DEC_{\bar{A}}(T\text{-}DEC_{\bar{B}}(R))))) \cup$$
$$(\Pi_{C_{AB},A,B,L_A,U_A,L_B,U_B}(\sigma_{L_B \le L_A \wedge U_A \le U_B}($$
$$T\text{-}DEC_{\bar{A}}(T\text{-}DEC_{\bar{B}}(R)))))$$

First, attributes $\overline{A}$, and $\overline{B}$ are decomposed into their components Then, the intersection of their intervals are determined There are four possibilities as indicated in Figure 5 For each case, the attributes of R, the boundaries of the intersection, and the interval columns for the attribute B are projected The union of the four possibilities forms the result relation Triplet-formation is applied to attributes A and B to obtain back the original relation As it is seen, slicing is a restriction operation on the time dimension.

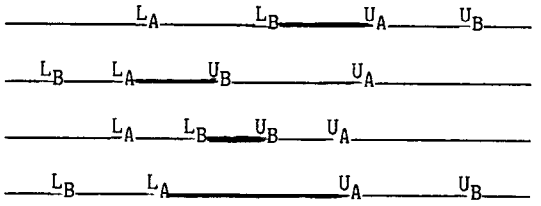

Figure 5 Possible intersection patterns

Similarly, drop-time can be expressed in terms of elementary operations The two possible expressions for drop-time are

$$\text{DROP-TIME}_{\overline{A}}(R) = \Pi_{Atr(R)}(\text{T-DEC}_{\overline{A}}(R))$$

$$\text{DROP-TIME}_{*\overline{A}}(R) = P_A(\Pi_{Atr(R)}(\text{T-DEC}_{\overline{A}}(U_{*\overline{A}}(R))))$$

10. ALGEBRAIC LAWS FOR THE NEW OPERATIONS

Following are the identities for the new operations defined. The proofs are straightforward and are left out due to space limitations. Let R and S be two relations, F be a formula as defined in Section 3 1, and $\overline{A}$ and $\overline{B}$ be triplet-valued attributes.

a Triplet decomposition

o $\text{T-DEC}_{\overline{A}}(\text{T-DEC}_{\overline{B}}(R)) = \text{T-DEC}_{\overline{B}}(\text{T-DEC}_{\overline{A}}(R))$

o $\text{T-DEC}_{\overline{A}}(R \times S) = \text{T-DEC}_{\overline{A}}(R) \times S$ provided that $\overline{A} \in Atr(R)$

o $\text{T-DEC}_{\overline{A}}(R \cup S) = \text{T-DEC}_{\overline{A}}(R) \cup \text{T-DEC}_{\overline{A}}(S)$ provided that Type (A)= Type (A')

o $\text{T-DEC}_{\overline{A}}(R-S) = \text{T-DEC}_{\overline{A}}(R)-\text{T-DEC}_{\overline{A}}(S)$ provided that Type (A) = Type (A')

o $\text{T-DEC}_{\overline{A}}(\text{T-FORM}_{B,L,U}(R)) = \text{T-FORM}_{B,L,U,}(\text{T-DEC}_{\overline{A}}(R))$ provided that interval attributes L, and U are properly referenced and A≠B When attributes are the same, i e A=B, right hand side gives the original relation R The left hand side gives the original relation only if triplet-formation can meaningfully be defined In this case, the equality does not hold

o $\text{T-DEC}_{\overline{A}}(\sigma_F(R)) = \sigma_{F'}(\text{T-DEC}_{\overline{A}}(R))$ F' is the same as F where references to attribute $\overline{A}$ in F are properly modified and converted to references to A, L_A, and U_A to obtain F' If A is not used in F, F and F' are the same

o $\text{T-DEC}_{\overline{A}}(\pi_X(R)) = \pi_{X,L_A,U_A}(\text{T-DEC}_{\overline{A}}(R))$ where $X \subseteq Atr(R)$ and $A \in X$ When $A \notin X$, the triplet-decomposition operation at the left hand side becomes null, because its operand is not among the attributes of the projected relation However the identity still holds

b Triplet-formation

o $\text{T-FORM}_{B,L_B,U_B}(\text{T-FORM}_{A,L_A,U_A}(R)) = \text{T-FORM}_{A,L_A,U_A}(\text{T-FORM}_{B,L_B,U_B}(R))$ provided that $L_A \neq L_B$ and $U_A \neq U_B$

o $\text{T-FORM}_{A,L_A,U_A}(R \times S) = \text{T-FORM}_{A,L_A,U_A}(R) \times S$ provided that $A, L_A, U_A \in Atr(R)$.

o $\text{T-FORM}_{A,L_A,U_A}(R \cup S) = \text{T-FORM}_{A,L_A,U_A}(R) \cup \text{T-FORM}_{A',L_A',U_A'}(S)$ provided that Type(A) = Type(A'), Type(L_A) = Type(L_A'), and Type(U_A) = Type(U_A')

o $\text{T-FORM}_{A,L_A,U_A}(R-S) = \text{T-FORM}_{A,L_A,U_A}(R)-\text{T-FORM}_{A',L_A',U_A'}(S)$ provided that Type(A) = Type(A'), Type(L_A)= Type(L_A'), and Type(U_A) = Type(U_A')

o $\text{T-FORM}_{A,L_A,U_A}(\sigma_F(R)) = \sigma_{F'}(\text{T-FORM}_{A,L_A,U_A}(R))$ F' is the same as F where references to A, L_A, and U_A are changed as A_V, A_1, and A_U, respectively If $\overline{A}$ is not used in F, F' is the same as F

o $\text{T-FORM}_{A,L_A,U_A}(\pi_X(R)) = \pi_{X'}(\text{T-FORM}_{A,L_A,U_A}(R))$ where $X \subseteq Atr(R)$ and $A,L_A,U_A \in X$, $X' = S-\{\overline{A},L_A,U_A\} \cup \{\overline{A}\}$ When $A,L_A,U_A \notin X$, the triplet-formation operation at the left hand side becomes null However the identity still holds

c Slice

o $\text{SLICE}_{\overline{A},B}(\text{SLICE}_{B,\overline{A}}(R)) = \text{SLICE}_{B,\overline{A}}(\text{SLICE}_{\overline{A},B}(R))$

o $\text{SLICE}_{\overline{A},B}(R \times S) = \text{SLICE}_{\overline{A},B}(R) \times S$ provided that $\overline{A},B \in Atr(R)$

o $\text{SLICE}_{\overline{A},B}(R \cup S) = \text{SLICE}_{\overline{A},B}(R) \cup \text{SLICE}_{\overline{A}',\overline{B}}(S)$, provided that Type ($\overline{A}$) = Type($\overline{A}'$), and Type(B) = Type($\overline{B}'$)

o $\text{SLICE}_{\overline{A},B}(R-S) = \text{SLICE}_{\overline{A},B}(R) - \text{SLICE}_{\overline{A}',\overline{B}}(S)$, provided that Type ($\overline{A}$) = Type($\overline{A}'$), and Type(B) = Type($\overline{B}'$)

o $SLICE_{A,B}(\sigma_F(R)) = \sigma_F(SLICE_{A,B}(R))$

o $SLICE_{A,B}(\pi_X(R)) = \pi_X(SLICE_{A,B}(R))$ where $X \subseteq Atr(R)$ and $A,B \in X$

d Drop-time

The algebraic identities for the drop-time operation are similar to the identities for the slice operation

11 CONCLUSION

In this paper, we introduced a methodology for incorporating time dimension into the relational model We propose time-stamping tuples Each attribute value is stored along with a time interval over which it is valid Non-first normal form relations are used for this purpose The attributes can be atomic, set-valued, triplet-valued or set-triplet-valued The latter two types of attributes preserve time (history) Furthermore, new operations are defined to extract information from historical relations

These operations are pack, unpack, triplet-decomposition, triplet-formation, slice, and drop-time The proposed approach has several advantages First, it is a closer abstraction of the real world Second, it eliminates data redundancy resulting from duplication of unchanged attributes which is inherent in the case of tuple time-stamping. Third, it is possible to obtain past relation instances which allow a user to extract information for any time point or time interval since the beginning of the database Furthermore, the proposed methodology is flexible and provides convenience to the user since it allows different types of attributes to coexist in the same relation

Further research is needed in several directions Currently, we are working on proving that the extended relational algebra has the same expressive power as the relational calculus for the historical relations and designing a graphical query language for HRDB's. We are also planning to explore the strategies for a prototype implementation of the proposed model and language

ACKNOWLEDGEMENTS

The author would like to express his gratitude to Dr E. Arkun of Baruch College, and Dr G Ozsoyoglu and Dr M Z. Ozsoyoglu of Case Western Reserve University for their valuable contributions

Appendix

STUDIOS

(STUDIO	HEAD	NUM_FILMS)
MGM	1924 --> Mayer	1924 --> 6
	1948 --> Schary	1925 --> 10
	1956 --> $NULL_2$	1970 --> 15
	1970 --> Aubrey	
	1974 --> $NULL_1$	
Paramount	1919 --> Cukor	1919 --> 2
	1925 --> Schulberg	1925 --> 12
	1935 --> $NULL_2$	1936 --> 10
RKO	1945 --> Schary	1945 --> 10
	1948 --> Hughes	1946 --> 11
	1957 --> $NULL_2$	1947 --> 12
Warner Br	1923 --> J Warner	
	1969 --> Ashley	$NULL_3$
	1972 --> $NULL_1$	
Universal	1912 --> Laemmle	1930 --> 6
	1936 --> Blumberg	1937 --> 9
	1946 --> Spitz	1965 --> 11
	1952 --> Rackmil	
	1955 --> Hunter	
	1965 --> Wasserman	

STARS

(NAME	DIRECTOR	BLOOD_TYPE)
Dietrich	1930 --> Sternberg	O
	1937 --> Lubitsch	
	1948 --> Wilder	
	1950 --> Hitchcock	
	1957 --> Wilder	
	1958 --> Welles	
	1959 --> $NULL_1$	
Grant	1932 --> Sternberg	AB
	1935 --> Cukor	
	1938 --> Hawks	
	1940 --> Cukor	
	1941 --> Hitchcock	
	1943 --> NULL	
	1946 --> Hitchcock	
	1960 --> NULL	
Hepburn	1932 --> Cukor	A
	1938 --> Hawks	
	1940 --> Cukor	
	1953 --> NULL	

DIRECTORS

(NAME	STUDIO	BLOOD_TYPE)
Sternberg	1924 --> MGM	A
	1926 --> Paramount	
	1935 --> Columbia	
	1938 --> MGM	
	1952 --> $NULL_1$	
Cukor	1930 --> Paramount	O
	1932 --> RKO	
	1938 --> MGM	
	1950 --> $NULL_1$	
Hitchcock	1927 --> Brit Intl	AB
	1934 --> Gaumont	
	1938 --> Gainsborough	
	1939 --> RKO	
	1951 --> Warner Br	
	1954 --> Paramount	
	1959 --> MGM	
	1960 --> Paramount	
	1962 --> Universal	
	1964 --> $NULL_1$	

FILMS

(FILM	STUDIO	YEAR)
The Blue Angel	Paramount	1930
Touch of Evil	Universal	1958
Angel	Paramount	1937
Stage Fright	Warner Br	1950
Witness for the Prosecution	Un Artists	1957
A Foreign Affair	Paramount	1948
Dishonored	Paramount	1931
Shanghai Express	Paramount	1932
Blonde Venus	Paramount	1932
The Scarlet Empress	Paramount	1934
The Devil is a Woman	Paramount	1935
Suspicion	RKO	1941
North By Northwest	MGM	1959
Notorious	RKO	1946
Sylvia Scarlett	RKO	1935
Bringing Up Baby	RKO	1938
The Philadelphia Story	MGM	1940
A Bill of Divorcement	RKO	1932
Little Women	RKO	1933
Adam's Rib	MGM	1949

LAWYERS

(LAWYER	STUDIO	SALARY)
Howell	1924 --> MGM	1924 --> 30K
	1930 --> Paramount	1925 --> 35K
	1937 --> MGM	1937 --> 40K
	1940 --> $NULL_1$	1940 --> $NULL_1$
Rosen	1912 --> Universal	1945 --> 70K
	1923 --> Warner Br	1953 --> $NULL_1$
	1930 --> $NULL_1$	
	1945 --> RKO	
	1953 --> $NULL_1$	
McManus	1923 --> Warner Br	1923 --> 35K
	1930 --> $NULL_1$	1926 --> 40K
		1930 --> $NULL_1$

References

1 Anderson, T.L Database Semantic of Time Ph.D. Th , Computer Science Dept., U of Washington, 1981

2 Ariav, G , Clifford, J , and Jarke, M Time and Databases ACM-SIGMOD International Conference on Management of Data, May, 1983, pp 243-245

3 Ariav, G Preserving the Time Dimension in Information Systems DS-WP 83-12-06, Decision Sciences Dept , Univ of Penn , December, 1983 (Ph D Dissertation)

4 Ariav, G , Beller, A , and Morgan, H L A Temporal Model DS-WP 82-12-05, Decision Sciences Dept , Univ of Penn , December, 1984

5 Ben-Zvi, J The Time Relational Model Ph D Th , Dept of Computer Science, University of California, Los Angeles, 1982 (Unpublished)

6 Bolour, A , Anderson, T L , Deketser, L J , Wong, H K T "The Role of Time in Information Processing A Survey" ACM SIGMOD Record 12, 3 (April 1982), 28-48

7 Clifford, J A Model for Historical Databases Proceedings of Logical Bases for Data Bases, Toulouse, France, December, 1982.

8 Clifford, J A Logical Framework for the Temporal Semantics and Natural-Language Querying of Historical Databases Ph D Th , Dept of Computer Science, SUNY at Stony Brook, December 1982

9. Clifford, J , and Warren D S "Formal Semantics for Time in Databases" ACM Trans on Database Systems 6, 2 (June 1983)

10 Codd, E F "A Relational Model of Data For Large Shared Data Banks" Comm of the ACM 13, 6 (June 1970), 377-387

11 Codd, E F "Extending the Database Relational Model to Capture More Meaning" ACM Trans. on Database Syst 4, 4 (December 1979), 397-434

12 Dadan, P , Lum, V.., and Werner, H D Integration of Time Versions in Relational Database Systems Proc. of The Tenth International Conference on Very Large Data Bases, 1984, pp 509-521

13 Falkenberg, E Time Handling in Database

Management Systems CIS-07174, University of Stuttgart, 1974.

14 Hammer, Michael,and Mcleod, Dennis The Semantic Data Model A Modelling Mechanism for Data Base Applications Proceedings of the ACM SIGMOD Conference, Austin, 1978

15 Jaeschke, G and Schek, H J Remarks on the Algebra of Non-First Normal Form Relations Proceedings of the 1st ACM SIGMOD Symp on Principles of Database Systems, 1982, pp 124-138

16 Klopprogge, M R Term An Approach to Include the Time Dimension in the Entity-Relationship Model In Entity-Relationship Approach to Information Modeling and Analysis, P P S Chen, Ed , ER Institute, 1981, pp 477-512

17. Klopprogge, M R , and Lockemann, P C Modelling Information Preserving Databases Consequences of the Concept of Time Proc of The Ninth International Conference on Very Large Data Bases, 1983, pp 399-416

18 Klug, A "Equivalence of Relational Algebra and Relational Query Languages Having Aggregate Functions" Journal of the ACM (1982), 699-717

19 Lum, V , et al Designing DBMS Support for the Temporal Dimension. Proceedings of the ACM SIGMOD Conference, Boston, June, 1984, pp 115-126.

20 Ozsoyoglu, M Z. and Ozsoyoglu, G An Extension of relational Algebra for Summary Tables Proceedings of the 2nd Intl Workshop on SDB Management, 1983.

21. Sciore, Edward Improving Semantic Specification in a Relational Database Proceedings of the ACM SIGMOD Conference, Boston, 1979

22 Snodgrass, R The Temporal Query Language TQuel Proceedings of the 3rd ACM SIGMOD Symp on Principles of Database Systems, Waterloo, Ontario, Canada, April, 1984, pp. 204-213