TIME DIMENSION IN THE RELATIONAL MODEL

by

YERRAPRAGADA CHAYA

M.Tech., (Comp. Sc.), Osmania University, 1985, India

A MASTER'S THESIS

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Computing and Information Sciences

KANSAS STATE UNIVERSITY

Manhattan, Kansas

1987

Approved by :

Major Professor

TABLE OF CONTENTS

LISTS OF FIGURES AND TABLES

ACKNOWLEDGEMENTS

I wish to express my sincere thanks to my advisor, Dr. Elizabeth Unger, for her valuable help, guidance, time, and encouragement during this work. It was a pleasure to work with her.

My thanks also go to the other members of my committee, Dr. Austin Melton and Dr. Richard McBride.

Last, I wish to thank my husband, Pratap, for his continuous support during my Graduate study.

Chapter 1.

INTRODUCTION

1.1 Introduction

To increase the semantic content of a database means to store more information in the database about the meaning of the data. There are different dimensions or attributes of information that can be incorporated. One attribute, time, is of interest to many, especially when it records the initiation or creation time of something. Representing knowledge about time, i.e., temporal knowledge, is the subject of this thesis. None of the three commonly recognized data models provides any special mechanism to store and process temporally-oriented data.

Humans live within a time continuum and objects are created and events occur within that continuum. We often can discern from the context of a situation the proper incarnation of an object to use. The question, "What does the budget indicate ?" within the context of a current monetary decision for a business, indicates to all concerned that the current budget is required. The question, "What did we have in last year's budget ?" indicates a reference to a budget instance at least one time unit old or timed as having a year value one less than current year. To deal with such temporal issues, time has to be recorded and manipulated and provision has to be made to represent the latest version of an output when no temporal reference is given.

In databases, for example, one must deal with the problem of outdated data. One approach to this problem is simply to delete the outdated data. But, this

approach eliminates the possibility of accessing any information except that which is presently "true". Queries such as "Which employees worked for us last year and made over $ 15,000 ? " become unanswerable. In the relational model, a database is a collection of relations represented in a table. For example, at a certain moment of time the extension of the relation, Employees, with three attributes, Name, Salary, and Department, could contain the data shown in Fig. 1.1.

Name	Salary	Department
John	15K	Toys
Lou	23K	Toys
Mary	25K	Credit

Fig. 1.1 : Employees relation extension.

In such relations, changes in the value of Salary and Department could occur. If at a certain point of time John's salary increased to 20K, then the database would be updated to contain this information and the previous information (history) about John's salary is lost. If any employee changes department, this changes the database but no history of the old department is kept nor is the time of the change recorded automatically. Traditionally the old information is simply deleted but some situations demand an archival facility or an audit trail. The database traditionally reflects only the present facts. Nothing about the past or the future is

recorded. Under the traditional approach, using the above relation, queries like, "Was John's salary at any time equal to 15K?", "Was John's salary increased ?", "Did Leu change departments ? If so when ?", "List the names of the persons if and when they got a raise of 1k in their salary.", become unanswerable. In order to answer these queries, when an information item becomes outdated, it need not be, perhaps must not be, forgotten. To model reality, it is not sufficient for databases to reflect only the present facts. They should not only be capable of representing the snapshot descriptions, but also must be capable of representing the evolution of descriptions over time.

This thesis proposes a model, TDRM, Time Dimension in the Relational Model, to incorporate the time dimension into the relational database. The model uses the technique of viewing relations as objects (see definition in Chapter 3) and instances of relations as instances of objects. Objects can have various components just as relations in a conventional database have attributes. These components in turn can have various attributes just as relation attributes can have attributes, e.g., type. Objects can be static or dynamic depending on the type of information contained. Terms like temporal object and temporal domain are defined with respect to the model.

In section 1.2, relevant research discussing the conventional relational database model, attributes of data, relational query languages and the concepts of modeling time is presented. Section 1.3 deals with the problem definition.

1.2 Relevant Research

1.2.1 Introduction

This section discusses the fundamental concepts that are necessary in order to incorporate temporal knowledge in the conventional relational model. These include the definition of the conventional relational database model, attributes of data, relational query languages and the extant concepts of modeling time.

In the subsection discussing the conventional relational database model, the characteristics of the model are provided in the context of its inadequacy to deal with temporal issues. The inadequacy lies with the attribute representation. If the semantic information represented by the attributes is increased then the model will be better able to represent and process temporal information.

In the subsection discussing the relational query languages, relational algebra and relational calculus are defined so that an appropriate set of them can be incorporated into TDRM.

In the subsection discussing the concepts of modeling time, the extant concepts for representing time are presented.

1.2.2 Conventional Relational Database Model

Conventional relational databases as all other logical models, model only the most recent snapshot of the real world. They generally lack the capability to record and process the time varying aspects of the real world. As events take place, the real world changes state and, at some point in time after the change, new values are incorporated into the database to revise the most recent snapshot. The

lack of storage of data and of models to organize its use indicates a lack of support for applications incorporating temporal requirements. This can cause serious problems, e.g., trend analysis (essential for decision support systems). In this case there is no convenient way to represent past, present, and future information; support for error correction or audit trails necessitates costly maintenance of backups, checkpoints, or transaction logs to preserve past states.

In the conventional normalized relational model, a relation is defined as a subset of the cartesian product D1 x D2 x D3 x .....x Dn, where Di are domains [ULL82]. These domains are not necessarily distinct. The characteristics of a relation are shown in table 1.1.

Table 1.1 : Characteristics of a relation

1. A relation extension is a set which can be thought of as a two dimensional table or a flat file.

2. The table has rows and columns and the cell entries are single valued.

3. Each column has a name. Columns are referred to as "attributes" (a1,a2,...,an). The values of attributes come from a set D of domains, D = {d1,d2,d3,...,dn}, each di being any nonempty set. The entries in a column are all of the same domain. To distinguish columns defined over the same domain, the attribute names are distinct within a relation. To relate the attributes with their domains, assume U is the set of all the attributes in the database and a function DOM : U —> D which maps each attribute onto its corresponding domain, that is, DOM(ai) is the domain of the attribute ai. Let UD denote the union of these domains, UD = d1 U d2 U ...U dn [CLI85].

4. A state or an instance of a relation is its current content. A relational database is a collection of relation instances on relation schemes [CLI85].

A relational scheme intension say R = <A,K> is an ordered pair consisting of a

finite set of attributes A = { a1,a2,a3,a4,...,an} and a finite set of key attributes K = { k1,k2,...,km}, where K has the properties of uniqueness and minimality [CLI85]. A relation extension r on relation scheme R = < A,K > is a finite set of mappings {t1,t2,...,tn}, where each ti is a function from A to UD such that ti(Aj) belongs to DOM(Aj) for all ti belonging to r and all Aj belonging to A [CLI85].

Fig 1.2 shows a relation extension of the relation, Faculty, with two attributes Name and Rank.

Name	Rank
Merrie	Associate Professor
Tom	Associate Professor

Fig. 1.2 : Faculty relation extension

This extension is inadequate to answer the queries such as, What was Merrie's rank two years ago ?, How did the number of faculty change over the last five years? It is also inadequate to record facts like, Merrie was promoted to a full professor last month, Lee is joining the faculty next year.

The reason for this inadequacy lies in the limited knowledge represented by the attributes of the data.

1.2.3 Attributes of data

Attributes of relations can be thought of as having many attributes including, owner, person who made the last change, security constraints, privacy constraints, creation/change time, storage format, etc. Conventional relational database systems traditionally allow but one attribute of semantic content, type. For our purposes the addition of time representation in some form is necessary. Note, this concept of having attributes of an attribute of a relation is different than the process of allowing the user to add temporal information as another attribute of the relation. In this latter case the entire responsibility for the integrity of the value of that appended temporal information lies with the ultimate user. In the former, our proposal, the information is managed by the system.

A database can be viewed as a collection of objects. The concept of an object to comprehensively represent all forms of information including database objects was proposed by E. A. Unger in 1978 [UNG78]. Such an object is defined as a five tuple O = (d, a, r, c, v), where d is a designator, a sequence of names that identifies the object and includes a user-defined name, context of creation, and particular instance of information. This is comparable to a relation instance of a relation; a is the attribute, containing information about the underlying representation, the internal structure, and the external relationships of the objects; r is the representation, containing information about coding, compression, and overlay characteristics of the object; c is the corporality, which provides information about the longevity of existence, the environment in which the object exists, the number of replications and their availability, the authorization for use, and a record of use and/or attempted uses; v is either an atomic value, an object, a number of atomic values, or a number of objects compatible with the attribute. A simplified version of this

type of object is used in this thesis.

1.2.4 Relational Query Languages

Relational Query Languages which are based on relational theory can be divided into three main groups [Ull82], [KRO83]. They are,

i) languages based on Relational Algebra

ii) languages based on Relational Calculus

iii) languages which are a combination of i) and ii)

To manipulate the relations, the query languages of the first category apply operators, e.g., union, defined in the relational algebra. These languages are more procedure oriented than the languages of the second category and require the user to understand not just what they want, but also how to compute it. An example for a query language in this category is ISBL [ULL82].

The query languages of the second category describe a desired set of tuples by specifying a predicate the tuples must satisfy. These languages are further divided into two classes depending on whether the primitive objects are tuples (tuple calculus) or are elements of the domain of some attribute (domain calculus). It tells only what is wanted and not how to get it, i.e., they are more non-procedural.

Five basic operations that define a complete relational algebra [ULL82] are given to illustrate the type of traditional query operators available (see Table 1.2).

Table 1.2 : Definition of the relational algebra operators.

Union:
The union of relations R or S, denoted R U S, is the set of tuples that are in R or S or both. Union operator is applied to relations which are " union compatible " (same number of attributes and the attributes in corresponding columns have the same domain).

Set Difference:
The difference of relations R and S, denoted R - S, is the set of tuples in R but not in S. R and S must be "union compatible".

Cartesian Product:
The Cartesian product of two relations is the concatenation of every tuple of one relation with every tuple of a second relation.

Projection:
The projection of relation R is an operation that selects specified attributes from R. The result of the projection is a new relation with the selected attributes.

Selection:
The selection operator takes a horizontal subset (rows). It identifies the tuples to be included in the new relation using logical or comparison operators.

Operations in the conventional query languages must be analyzed and extended to deal with the temporal dimension when it is introduced into the relational model. In our proposal we have analyzed the relational operators and extended the projection and selection operators to deal with time.

1.2.5 Modeling Time

Researchers in disciplines as varied as artificial intelligence, logic, natural language processing, distributed processing and database systems have studied the role(s) that time plays in information processing. There are various perspectives on

time that is important to save/use in systems dealing with temporal information. One perspective is the level of granularity of the information which is "timed". For instance, one may consider the entirety of the information as a unit, this results in things like snapshot databases, in which case any change to the information or the completion of a duration of time triggers a new timed instance. At the other end of the spectrum time may apply to the smallest atomic unit of data and each atomic unit only is incarnated when a change occurs. A second perspective is the "type"of time one would record, for instance the time of creation. Another aspect of the type of time is the expression of such as a date of incarnation or date and hour of incarnation. We surveyed several areas in the literature. These include the question answering systems [FIN71] (providing insight into time measurement), relational theory in which Codd provides insight into algebraic extensions to handle the time measurement, the area of logic in which the Legol system designed by Jones and Mason provides a relational specification language for writing rules, temporal logic in which the representation of information in a database has been explored by Ortowska, historical databases in which the changing states of information has been studied by Clifford and Warren, incorporation of the temporal dimension into the relational model has been studied by Clifford and Tansel and finally incorporation of the temporal dimension at the tuple level was suggested by Gadia. The results of this survey are given below.

There has been some work done in the area of intelligent question-answering systems [FIN71]. These systems do more than retrieve "raw" data ; they make deductive inferences in order to return all valid responses and report logical inconsistencies. More information is requested from the user if a question proves to be ambiguous to the system. From these systems, the specific temporal notions of

importance to this work are : event start time, finish time, and duration. Any of these times may be unknown. Two types of events are distinguished, instantaneous, and duration. The instantaneous events take place with no duration time, i.e., start time is equal to finish time. The relations "before", "after", and "simultaneous" with respect to events are defined. Time points can be identified in one of two modes, dates, and simple integers, which are dependent on the application. The start and finish times of events can be related by timing relations to the times on one or more of these time scales. Also, the interval between two events can be specified in terms of units of time in each scale.

The relational database model has been extended to capture more meaning by Edgar Codd in 1979 [COD79]. Some of the extensions are the definition of four relations between event types: unconditional successor, alternate successor, unconditional precedence and alternate precedence. An event E1 is an unconditional successor (unconditional precedence) of an event E2, if E1 always succeeds (precedes) E2 and there is no intermediate event satisfying this property. Alternate successor (alternate precedence) represents an exclusive-or of successors (precedences). An event of a given type must be followed (preceded) by an event that belongs to one of its alternate successor (precedence) types, if any.

These relationships can be used during insertion and update to check the validity of a requested transaction. For example, for a new addition to the quantity of stock in the database, the system can check for an order event that should unconditionally precede the arrival of new stock.

Using a time dimension in data modeling, updating, and developing a user interface is discussed by S. Jones and P. Mason in 1980 [JON80] within the context of a system named Legol. Legol is a relational specification language for writing

rules. Because Legol is based on the relational model, all data is recorded in flat files or tables, but with the extension of three types of attributes: identifier attributes that name entities, characteristic attribute that designates a property of an entity, and time attributes (start time and finish time) that designate a period over which an entity had a property. So, the historical and the current information are stored together.

The use of temporal logic formalism for representing information in a database is discussed by E. Ortowska in 1982 [ORT82]. Two different ways of viewing information are given; the first syntactic view (objects) and the second semantic view (set of objects) are presented. Two languages are introduced corresponding to two different ways of viewing information. Further a dynamic information system is defined as S = (Ob, T, R, At, { Va} where 'a' belongs to At, F), where Ob is a nonempty set whose elements are called objects, T is a nonempty set whose elements are moments of time, R is a linear order on the set T, At is a nonempty set whose elements are called attributes, Va, a belongs to At, is a nonempty set whose elements are called values of attributes a, and F is a function from time to the domain of the attribute. The model uses tense operators : always in the future, always in the past, sometime in the future, sometime in the past. These tense operators are used to recall any past state or any future state of a system with respect to a given state. The language based on a syntactic view enables one to ask queries like, "Is there any moment earlier (later) than 't' such that a certain assertion about properties of objects is true ?". The language based on the semantic view would give the ability to ask a query like, "What are the objects which in the time period between moment t1 and moment t2 have a certain property ?".

The concept of historical databases (defined in [CLI85]), as a tool for modeling the changing states of information about some part of real world was proposed by J. Clifford and D.S. Warren in 1981 [CLI81]. The real world is modeled in terms of states. A new state description is associated with changes in the real world. Their work extends the relational model to include the time dimension. Database attributes are viewed as functions from moments in time to values (in the appropriate domain). The technique used to build an historical database is to time stamp the tuples in the database. Time stamping is done by adding a new attribute, STATE to the relational scheme. A relation instance is associated with a state. The historical database is viewed as a three dimensional relation scheme. Time adds the third dimension to the normal flat table view of relations. Some of the entities that exist in one state may not be there in some other state. A boolean-valued attribute, EXISTS, is introduced into the historical database to indicate the entities that are of interest in any state. If an entity, for example, an Employee, does not exist in a state, say Si, then the relation instance in this state contains a value 0 for the attribute EXISTS for that employee.

Two view points to the problem of augmenting the relational model are presented by J. Clifford and A. Tansel in 1985 [CLI85]. The intersection of the two views is to incorporate the temporal dimension into the relational model at the attribute level, rather than at the tuple level as has been generally proposed. Thus attributes have domains which are either simple or are complex with some form of time information. The two views differ in the exact nature of this complex domain. Clifford treats these domains as functions from time points to simple values. Tansel treats them as functions from time interval to simple values. Due to the complex structure of domains for some attributes the relations no longer are in first

normal form, for example consider an attribute Salary for a relation Employee. In conventional databases this attribute reflects the salary of the employees in the relation at any given time. In order to make this attribute reflect the salary of employees over the evolution of time, it must have time information along with the value of the salary (complex domain). A person can be an employee for certain disjoint periods of time as shown in relation extension of relation Professor in Fig. 1.3. The figure shows that the relations are no longer in first normal form.

Name	Rank	Salary
Merrie	Associate Professor	1977 to 1980 30K
		1982 to 1985 40K
Tom	Associate Professor	1982 to 1985 40K

Fig. 1.3 : Professor relation extension

In the historical database model, as suggested by Clifford [CLI85], an historical database consists of a collection of historical relations, each relation consisting of a set of attributes and each attribute defined over some interval of time. The set Time is a set of times, {t1, t2, ...} that is infinite, and with a linear order, BEFORE, where BEFORE = { ti,tj | ti is before tj}. Three different kinds of attributes are distinguished : constant attributes, time varying attributes, and temporal attributes. Constant attributes are defined as attributes that are time invariant. Time varying

attributes potentially vary over time and temporal attributes are attributes that do not change with time but have a time domain, for example date-of-birth.

Tansel discusses the method of time_stamping the attributes, for incorporating time dimension into the relational model. Time is considered as a set of consecutive, equally_distanced points. Between two consecutive points there is a time duration which is equivalent to one unit. No definitions or semantics of the temporal domain or the temporal relation are given.

A model has been proposed for a temporal database by S.K. Gadia in 1985 [GAD85]. The change in the value of the attributes is recognized by this model through an association of a period of time for which a value is valid. It is a homogeneous model in the sense that the periods of validity of all the attributes in a given tuple of a temporal relation is identical (similar to time stamping a tuple). This work defines the temporal domain and period of validity with respect to a tuple. A data type, called a temporal element, is introduced to represent time. All references about time are confined to the interval [0 , now] of instants in time. The author presents a relational algebra for the homogeneous temporal database.

Addition of attributes "STATE" and "EXISTS" [CLI81], START and END [JON80], to a relation to represent time leads to a model that is highly redundant and involves indirect representation of time. Our model incorporates time directly. The models ([CLI81], [GAD85], and [JON80]) that incorporate the temporal information at the tuple level are not realistic because in the real world, attributes in a tuple may change values at different times. Therefore, time should be modeled at the attribute level. Models ([CLI85]) that use point-based representation of time involve unnecessary overhead to answer queries dealing with the duration concept. Discussion in chapter 2, section 2.2, compares the two types of

representation of time : point-based and interval-based. We arrive at a conclusion that interval-based representation should be used. Models should define or give semantics of concepts like the temporal domain and the temporal relation [GAD85].

All the models do make an attempt to handle "time" in databases. However, the models lack some desirable properties like for example, management of temporal information by the system.

Our research is focused on giving a more abstract view of a temporal database. The fundamental difference between our proposed model and the models discussed is that our model takes the view point of an "object", which has the inherent advantage of being abstract and flexible. Our model does not treat "time" as a special dimension. The concept that we use is to treat "time" as an attribute of an attribute (in a sense, property of an attribute) and not as another attribute of a relation. Due to this the temporal information, in our model, can be managed by the system, where as in the other models, user is responsible for the integrity of that appended attribute.

1.3 Problem definition

This thesis proposes a model, TDRM; the aim of which is to incorporate time in relational model. One of the ways to incorporate time is to view a database as a collection of objects. The type of object selected is a simplification of the object introduced by E. A. Unger in 1978 [UNG78]. The time information is added in the model as a property of a data item with a default value assumed. The time information can be of two types : static and dynamic. Static means the value once given does not change and dynamic means the value changes with time and a new object

instantiation is created each time the value changes. New instances of objects are associated with the changes in the real world which is similar to associating new state description with the changes in the real world as done by Clifford [CLI81]. The model is able to differentiate between active and inactive objects through the use of the time information. If the time information for an object involves the present then that object is active otherwise it is inactive. Clifford's model differentiates between active and inactive tuples by using a boolean-valued attribute "Exists". Definitions of terms like temporal object and temporal domain are given with respect to the model. An attempt is made to define the relational algebra operators for this model. Examples are given illustrating the use of the operators.

TDRM uses the concept of time intervals to represent the temporal information. This representation is similar to the concept presented by N. Findler and D. Chen in 1971 [FIN71], but TDRM does not deal with unknown values. The concepts of Legol system can be used with TDRM to develop a relational specification language. The event relationships developed by Codd [COD79] can be used in relation to TDRM when the aspects of temporal logic are to be explored.

Chapter 2 discusses the techniques for representation and modeling of time. Chapter 3 defines, TDRM. Chapter 4 presents the results and the future direction of research that can be pursued.

Chapter 2.

REPRESENTATION AND MODELING OF TIME

2.1 Introduction

There are two representations of time that were considered for use in TDRM, point representation and interval representation. The following discussion justifies our choice : the interval-based representation of time.

In an interval-based representation, time takes the notion of a "temporal interval" or time interval as primitive. In a point-based representation, time takes the notion of "time at a point" as primitive.

In natural language, we refer to time as points and as intervals, for example,

"He arrived at 1.00 P.M."
"He arrived yesterday."

In the first sentence "at 1.00 P.M" refers to a precise time point at which the event "arrived" occurred. In the second sentence "yesterday" refers to an interval in which the event "arrived" occurred.

Events are often related to other events, for example,

"We found the letter while John was away."

This indicates that the event "found" occurred during the time when John was away. Some events appear to be instantaneous on the surface, but they can be further decomposed, for example, "finding the letter" can be composed of "looking at spot X" and "realizing that it was a letter". Similarly "realizing that it was a

letter" could be further composed of series of inferences. Thus, time points may be considered to be very small time intervals. There are two types of systems as a result of the two time representations. The relative merits of interval-based systems versus point-based systems are discussed next.

2.2 Time points vs Time Intervals.

Objects/descriptors may have corporality. The corporality value is recorded as <time>. This time can be taken as the time point at which the object/descriptor value is acquired, i.e., <t> or as the time interval in which the value is valid, i.e., <t1,t2>. Representing time as a point (Fig. 2.1) is simple and requires less storage space. However, to determine the time duration over which a value is valid or to determine a condition's truthfulness over an interval, the successor time point has to be examined. This creates complexity in operators. To have an interval representation (Fig. 2.2) requires an understanding of open and closed intervals.

An open interval is a interval which does not include both the ends, e.g., (left, right) of the interval. A closed interval is a interval which includes both the ends, e.g., [left, right] of the interval. If the intervals are open, then there exists times, the end points, at which an event is neither true nor false and if the intervals are closed there exists times, the end points, at which an event is both true and false. Consider two intervals (p,q) and (q,r), since the intervals are open then the times p,q,r are not included in the duration of the intervals. If the intervals are closed, i.e., [p,q] and [q,r], then the times p,q are included in the duration of the first interval and the times q,r are included in the duration of the second interval. If an event, J, is true in the interval (p,q) and is false in the interval (q,r) then since these are open intervals, at time q the event is neither true nor false. In the case of

closed intervals, i.e., [p,q], [q,r], at time q the event is both true and false. One solution to this problem of indeterminism is to define the concepts of left close or right close, where left close means that the time at the left end of the interval is included but not the right end, e.g., [p,q). Right close means that the time at the right end of the interval is included but not the left end, e.g., (p,q].

Name	Salary
John	11, 15K
Tom	21, 20K

Fig. 2.1 : Point representation

Name	Salary
John	[11,50) 15K
	[50,55) 20K
Tom	[0,21) 20K
	[41,52) 30K

Fig. 2.2 : Interval representation

To test a condition P during an interval "t", which is included in an interval "T", the concept of inheritance could be is used. If P is true during T, then P is true during t. For example, in databases, a query of the form : Was John an employee of XYZ during 1977-1985 ? is very common. Now, since this interval 1977-1985 is in turn during an interval 1965-Now, where "Now" represents the present and if it is known that John is an employee of XYZ during 1965-Now, then John being

an employee of XYZ during 1977-1985 is true. So, we can have hierarchy of intervals in which propositions (operands that have true or false values) can be inherited. This is a definite advantage of an interval-based system.

Since time points could be taken as very small intervals and since there are advantages associated with time intervals, it is appropriate to incorporate the interval-based representation of time in the temporal relational model.

2.3 Current Techniques for Modeling Time

The previous work, in modeling time, is roughly divided into four categories : State Space approaches, Date-Line systems, Before/After chaining, and Formal Methods.

The state space approaches [FIK71] provide a crude sense of time. They are useful in simple problem-solving tasks. A state in this approach is a description of the world (i.e., a database of true facts) at an instantaneous point of time. Actions are modeled as functions which map between states. For example, if an action occurs that causes P to become true and causes fact Q to be no longer true, its effect is the addition of P to the current state and deletion of fact Q from the current state.

In the date-line systems [BUB80], each fact is indexed by a "date" where the date is a representation of time. In this scheme the temporal ordering between two dates can be computed by simple operations.

The before/after chains [KAH75], approach captures the relative temporal information. When information about a new event is added to the database, the system fits it into a before/after chain. Due to this, as the amount of temporal

information grows, searching the chain becomes a problem. A fact that two events A and B did not occur at the same time can be represented by the position of the two events in the chain. Facts like, event A and event B occurred at the same time cannot be captured in these systems.

Formal methods [DER81], in artificial intelligence include the situation calculus. In situation calculus, knowledge is represented as a series of situations, each being a description of the world at an instantaneous point of time. Actions are means to transform one situation into another situation. Here a fact that is true at one situation, needs to be rechecked for truth in another situation.

The technique that the proposed model uses is the interval-based approach, where time is represented as an interval. This representation was chosen assuming most of the events do not occur at a time point and even if they occur at a time point, that point can still be represented as a time interval, i.e., [a,a]. The interval approach also allows easier processing to answer queries like, "What is the duration that John worked for 20K in shoe department? ".

The proposed model, TDRM, Time Dimension in Relational Model is presented in chapter 3.

Chapter 3.

THE MODEL

3.1 Introduction

In this model, a relation scheme is viewed as a meta object and an instance of a relation scheme is called an object (discussed in section 3.2). Each object has components and a description of these components is given in section 3.3. An overview of the proposed model is presented in subsection 3.4.1. We have made some conditions about "time" in the model and these are given in subsection 3.4.2. Subsection 3.4.3 is devoted to a discussion of events, which are actions causing the assignment of new values to the components of an object. The incorporation of the time dimension in the relational model is discussed in section 3.4.4. Relational algebra is used as a query language, to manipulate the relations. A brief discussion on the influence of the time dimension on the relational algebra is presented in the subsection 3.4.5.

3.2 Objects and Meta objects

In TDRM, the relational scheme is viewed differently than it is in the conventional relational model. The relational intensional scheme R (A, B, C,...,Z) is considered in this model to be a meta object. The attributes of the relational scheme A, B, C,...,Z are called descriptors in TDRM. Each tuple in the conventional relational model is considered to be an object in TDRM.

Definition 1 : Meta object
A relational intensional scheme is a meta object. The universe of all possible meta objects, MO, in the model is the cartesian product of five components [UNG78] :

$$MO = D * De * R * C * V$$

where,
D is the designator space.
De is the descriptor space.
R is the representation space.
C is the corporality space and
V is the value space.

Definition 2 : Object
An instance of a meta object is an object, o, which is a five tuple of components :

$$o = (d, de, r, c, v)$$

where,
d, belongs to D, is the designator.
de, belongs to De, is the descriptor.
r, belongs to R, is the representation.
c, belongs to C, is the corporality and
v, belongs to V, is the value.

The first component of the object is the designator. It is a simple name that uniquely identifies the object. The second component, descriptor, contains information describing the attributes of an object itself, e.g., format. Representation, the third component, contains the representation details (coding, compression, etc.) or the internal structural details of the object. In TDRM this component is considered to be null. Corporality, the fourth component, provides information about the longevity of the object's existence. The longevity of existence can be either static or dynamic. These terms are defined in section 3.3. Value, the last component, is an object. Values in TDRM are restricted (simple values). In a meta object instance, i.e., a tuple, the value consists of the set of simple values which represent that

instantiation of the meta object. The next section discusses the various components of an object.

3.3 Components of an object.

3.3.1 Designator

Each object in this model has an identifier or a name that identifies it uniquely. A name could consists of a sequence of names (context information, instance information, etc.).

3.3.2 Descriptor

This component corresponds to the attributes in conventional relational model. In this model they are called the descriptors as they describe the object. Descriptors contain what are called the attributes. The descriptors can be further qualified with information specifying, format, form, date, and longevity. The longevity specifies the duration in which a value is valid.

In the proposed model, the value of the descriptors are divided into two classes : static and dynamic. Descriptors such as Social security number (SSN), Name in an employee object would be descriptors of static values. Values of descriptors such as Salary, Department in an employee object would be dynamic as their value may vary over time.

3.3.3 Representation

The third component of the object describes the representation within the system, for example the coding scheme and packing information which is utilized.

This information is not needed by the application-level users and hence can be transparent to the user.

3.3.4 Corporality

The fourth component describes the state of the object in terms of longevity and context of existence. The longevity of objects can be considered as a continuous spectrum. In this model the continuous spectrum is broken into two categories : static and dynamic. Objects with static longevity are those objects whose components do not change once a component of this type of object exists. Such objects are called static objects. Objects with dynamic longevity are those objects whose value may change over time and each value change causes a new distinguished incarnation or instance to exist. Such objects are called dynamic objects. All the instances of the same object can be maintained. The most recent incarnation of the created object is active and all the rest of the objects are considered inactive. Thus there is a direct correspondence of these objects to active and inactive tuples in Clifford's model.

Consider a variable "Now" which represents the present moment. An active object has "Now" as the right end value of the period of its longevity component of its corporality. Similarly an object is an inactive object if its longevity component of its corporality contains a time point which is less than "Now" as the right end value of the period.

3.3.5 Value

The fifth component of an object is its value. The value represents the information which in the general case of this type of object may be an object. However,

for the purpose of this thesis we are limiting these values to be simple primitive or unstructured values.

3.4 TDRM

3.4.1 Introduction

In TDRM, the database consists of a collection of objects and meta objects. The schema describing the entire database (intension) is called a meta object as is each individual relational schema. Over time, an essential property to consider is the corporality of a tuple/object. In TDRM the relations are viewed as three dimensional cubes, the first dimension being the row, representing an object instance, the second being the column, representing the components of an object, and the third dimension being the attributes of the components, one of which is the longevity of the objects which is the time period during which the value of an object is valid (refer to Fig. 3.1).

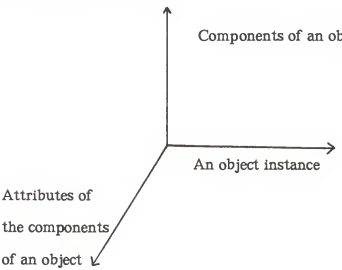

Fig. 3.1 : Three dimensions of a relation

Deletion of objects from the database is not done except for storage management purposes. The "deleted" objects are stored in the database with the associated corporality information. Consider for example, events like "firing an employee", "rehiring an employee", these events change the time of existence of a dynamic longevity object "employee". A change of value in an object with dynamic longevity is reflected by the creation of a new instance of that object reflecting the changes.

The period of existence of an object is represented as an interval [t1, t2), where t1 is the time at which the modeling of an object started and t2 is the moment the object ceased to exist. We have made some conditions about representing "time" in our model, which are discussed next.

3.4.2 Conditions

The data type for the domain of time is considered as a set of time points, which is linearly ordered. Fig. 3.2 shows the time axis with the discrete points, t0, t1, t2, t3,..., Now, where "Now" represents the current time point. The discrete points, t0, t1, t2, t3,..., Now, may or may not be equally distanced.

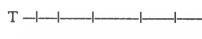

Fig. 3.2 : Time axis

Time points are represented or visualized as closed intervals [ti,ti] (chapter 2, section 2.2). There is no assumption made concerning the length of the time unit.

It could be days, hours, minutes, years, a combination of these or some user defined time which can be represented as a monotonically ordered sequence of points. A universe for Time 'T' is T = { t1, t2, t3,...,ti,..., Now}, where the points are strictly monotonically increasing, i.e., t1 < t2 < t3 < ...< ti < Now. If we represent ti - tj = d(i,j), where i >= j and j >= 1, then, ti = tj + d(i,j). Between two adjacent points say : t1 and t2 there is a time duration of (t2 - t1) .

The interval between two time points is the union of intervals between the time points which are in between the original two time points, for example the time interval between t1 and ti is (interval between t1 and t2) U (interval between t2 and t3) U... U (interval between tj and ti), where i >= j , j >= 1 and U is the union operator. The lower bound of the interval is t1 and the upper bound is ti, where t1 <= ti. The interval from t1 to ti is closed at the beginning (left close) and is open at the end (right open). This is represented as [t1 , ti).

Two intervals X = [u , v) and Y = [x , y) are disjoint if they do not have any points in common, i.e., their intersection is null, i.e., X Y = O.

Two intervals X = [u , v) and Y = [x , y) are said to overlap, if they have time points in common, i.e., their intersection is not null, i.e., X Y = O.

Two intervals are said to be adjacent, if one follows the other. In other words, the right end of the first interval is the left end of the second interval, for example, [u , v) and [v , x).

The interval [u , v) is a subset of the interval [w , x), if the latter interval includes all the time points that are in the former interval. The time intervals are like sets because a time interval consists of elements which are time points and do not contain duplicate elements.

Any interval which includes "Now" as its upper bound is an expanding interval. As time advances the interval [ti, Now] also expands. Unlike the other intervals, this is a closed interval and so it includes both the lower and upper bounds.

These conditions about "time" are useful in computation of responses to queries refering to time. For example, to answer the query "Was Joe not an employee of the organization for a period of time ?" involves the adjacency condition if Joe had changed departments within the same organization. Also, to answer the query "Is Joe an employee of the organization ?" involves the use of the concept of "Now". In effect, these conditions help in analyzing the time intervals to answer the queries dealing with time.

3.4.3 Events

A relation/object R has components and one of the component is the descriptors. The descriptors of an object may be independent which means that they may take different values at different times. For example, in the object R with descriptors (Name, Salary, Department) the descriptor, salary and the descriptor, department, of an employee may change with time. Further, they may not change at the same time, for example, an employee's salary may change from 15K to 20K at time 't1' but he may not change the department at that time. Also, an employee may change department at time 't2' while still earning the same salary. Hence the two descriptors are independent. Changes in the descriptor values are triggered by events.

Events are transactions, operations or actions causing the assignment of new values to the components of an object. Removing an employee, changing department assignment and increasing a salary, are examples of events. Events could

occur in an interval or at a time point. In Fig. 3.3 e1, e2, and e3 are events. They assign the values a1, a2, and a3 to descriptor A of a particular instance of an object respectively. a1 is assigned at time t4 and is valid for the time interval [t4 , ti). a2 is assigned at ti and is valid for the interval [ti , tj). a3 is assigned at tj and is valid for the interval [tj , Now].

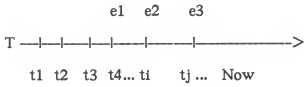

Fig. 3.3 : Time axis with events

3.4.4 Time

Some researchers treat "time" as a component of tuples. However, in the proposed model, the corporality component applies to the object and the longevity of its components provides the information about time. This is because of the following reasons :

a) Value of the descriptors of an object even if the object is dynamic may be, static or dynamic.

b) Value of the descriptors of an object, if the object is static, will be, static.

c) When a change occurs, it is generally to the value of a descriptor of an object and not to the value of all descriptors of an object.

Corporality is attributed to the objects and longevity is attributed to the descriptors of the object as shown in Fig. 3.4. This attribution is done while inserting and updating. Each corporality/longevity component is recorded as a <time interval> : [t1, t2), where t1 and t2 are the lower and upper bounds of an interval respectively in which the value of an object or a descriptor is valid. Fig. 3.4 represents the meta object Employee and its components. The descriptors of this object are Name, Birth-Year, Salary and Dept. Each of these descriptors is in turn an object with attributes Format, Type,..., Longevity. Depending on the type of value of the descriptor the longevity attribute is attributed as static or dynamic. If an object has one or more descriptors whose value is attributed as dynamic then the corporality component of the object will be attributed as dynamic otherwise it is considered to be static. If the value of a descriptor is attributed as static in the meta object specification then the longevity attribute of that descriptor in an instance of the object is not specified. The corporality of an instance of an object is defined as the time interval resulting from the union of the time intervals in the longevity attribute of the descriptors of dynamic value.

Definition 3: Temporal object
Temporal object r over a meta object R is a finite set of objects representing information about its components consisting of designator, descriptor of value types: dynamic and static, representation, corporality and value.

An example of an instance of a temporal object is shown in Fig. 3.5.

Meta object	:	EMPLOYEE				
Descriptors	:	Name				
		Birth-Year				
		Salary				
		Dept				
Corporality	:	dynamic	:	[t1,t2)		
Object	:	Name				
Attributes	:	Format				
		Type				
		:				
		:				
		Longevity	:	Static		
Object	:	Birth-Year				
Attributes	:	Format				
		Type				
		:				
		:				
		Longevity	:	Static		
Object	:	Salary				
Attributes	:	Format				
		Type				
		:				
		:				
		Longevity	:	Dynamic	:	[0,Now]
Object	:	Dept				
Attributes	:	Format				
		Type				
		:				
		:				
		Longevity	:	Dynamic	:	[0,Now]

Fig. 3.4 : Object Employee and its components

Object A	:	employee				
Descriptors	:	Name	:	John		
		Birth-year	:	1960		
		Salary	:	15K longevity	:	[11,50)
		Dept	:	Toys longevity	:	[11,45)
Corporality	:	[11,50)				

Object A	:	employee				
Descriptors	:	Name	:	John		
		Birth-year	:	1960		
		Salary	:	20K longevity	:	[50,55)
		Dept	:	Shoe longevity	:	[45,55)
Corporality	:	[45,55)				

Object B	:	employee				
Descriptors	:	Name	:	Tom		
		Birth-year	:	1961		
		Salary	:	20K longevity	:	[0,21)
		Dept	:	Toys longevity	:	[0,21)
Corporality	:	[0,21)				

Object B	:	employee				
Descriptors	:	Name	:	Tom		
		Birth-year	:	1961		
		Salary	:	30K longevity	:	[41,52)
		Dept	:	Shoe longevity	:	[41,52)
Corporality	:	[41,52)				

Object C	:	employee				
Descriptors	:	Name	:	Mary		
		Birth-year	:	1958		
		Salary	:	25K longevity	:	[0,Now]
		Dept	:	Credit longevity	:	[0,Now]
Corporality	:	[0,Now]				

Object D	:	employee				
Descriptors	:	Name	:	Leu		
		Birth-year	:	1962		
		Salary	:	23K longevity	:	[0,Now]
		Dept	:	Toys longevity	:	[0,Now]
Corporality	:	[0,Now]				

Fig. 3.5 : Example : Employee Relation/objects

The longevity information is initialized to some initial value [0,Now]. When an event occurs in the interval [t , Now], this interval is split into two intervals [t , u) and [v , Now] such that u = v and v = value of Now when event, e, occurred. This is done to record and later to be able to process the time varying aspect of information in the database. For example, assume s is an instantiation of an object R, one of whose descriptors is A. Let Event e assign a new value a' to A. Before e occurs s[A] contains "a" with longevity information [1,Now]. After e occurs, an instance of the object s is created say s1 whose descriptor A will contain " a' " with longevity information [v,Now]. In the object s the descriptor A would contain "a" with longevity information [1,u), where u = v as above. The various instances of an object are maintained. The present moment 'Now' can be viewed as a variable. To update Now, we simply have to reassign the variable to a new time value which is after the previous time value representing the present moment.

Since the descriptors of dynamic value change value with time, we have provided a model to represent the corporality aspects. The domain for the longevity of a value of a descriptor is from the temporal domain, which is defined as follows:

Definition 4: Temporal domain
Temporal domain is a finite union of intervals in T (set of time points). It is represented as 'tdom'. An element of the temporal domain is an interval of time.

The life span or the corporality of a relation/object is also represented as an interval. The temporal domain of a relation is defined as follows :

Definition 5 : Temporal domain of an object r
Temporal domain of an object r, denoted as tdom(r), is either static if all descriptors are of static corporality or is the union of temporal domains of corporality of all instances of the object if any one descriptor is not of static

corporality. r(v) represents an object r restricted to a temporal domain v. At any given point of time if a relation/object is viewed then it is called the snapshot view of the relation/object. It can be defined as follows :

Definition 6: Snapshot of a relation/object r at an instant t
A snapshot of a relation/object r at an instant t is the relation/object r([t,t]).

Fig. 3.6 shows the snapshot of Employee relation/object at [32,32] (refer to Fig. 3.4 for a definition of the meta object employee).

Object A	:	employee				
Descriptors	:	Name	:	John		
		Birth-Year	:	1960		
		Salary	:	15K longevity	:	[11,32)
		Dept	:	Toys longevity	:	[11,32)
Corporality	:	[11,32)				
Object C	:	employee				
Descriptors	:	Name	:	Mary		
		Birth-Year	:	1958		
		Salary	:	25K longevity	:	[0,32]
		Dept	:	Credit longevity	:	[0,32]
Corporality	:	[0,32]				
Object D	:	employee				
Descriptors	:	Name	:	Leu		
		Birth-Year	:	1962		
		Salary	:	23K longevity	:	[0,32]
		Dept	:	Toys longevity	:	[0,32]
Corporality	:	[0,32]				

Fig. 3.6 : Snapshot of object Employee at [32,32]

Definition 7: Equivalent temporal objects
Two objects r and s, instances of the same meta object, are equivalent if all their snapshots are identical.

This section has presented the basic model with corporality information. The next section makes an attempt to deal with queries.

3.4.5 Algebra Operations

Standard relational algebra operations can be applied to the model with no modification by defining these operations to act only on the active incarnation of the object. Modifications to accommodate the value types, static and dynamic, can be made to enhance the retrieval capability of the system. Thus, an extended query language for the proposed model can be defined. This section presents two extended operators of such a query language. These operators are called T-Projection and T-Selection.

1. T-Projection : T-Projection over any descriptor in this model carries essentially the same definition as projection over any attribute in the conventional model but with all incarnations available. The projection on a temporal object R is an operation that selects values of the specified descriptors from R. The result of projection is a new temporal object with the selected descriptors. An example of T-Projection is given in Fig. 3.7 based upon the object defined in Fig. 3.4 and the extension given in Fig. 3.5.

T-Projection EMPLOYEE (Name, Birth-Year)

Object A	:	employee		
Descriptors	:	Name	:	John
		Birth-Year	:	1960
Corporality	:	Static		
Object B	:	employee		
Descriptors	:	Name	:	Tom
		Birth-Year	:	1961
Corporality	:	Static		
Object C	:	employee		
Descriptors	:	Name	:	Mary
		Birth-Year	:	1958
Corporality	:	Static		
Object D	:	employee		
Descriptors	:	Name	:	Leu
		Birth-Year	:	1962
Corporality	:	Static		

Fig 3.7 : T-Projection of EMPLOYEE (Name, Birth-Year)

2. T-Selection : The user may ask queries with the selection criteria involving descriptors of static or dynamic values. The selection operator takes a horizontal subset of a temporal object based on the specified criteria. If the criteria is based on a descriptor of a dynamic value then one may specify the longevity information. Some examples are shown below (Fig. 3.8, Fig. 3.9, Fig 3.10): (refer to Fig 3.4 and Fig 3.5 for a definition of the meta object employee and for an instance of the object employee respectively).

a. T-Selection on a descriptor of a static value

T-Selection EMPLOYEE (Name = Leu)

Object D	:	employee				
Descriptors	:	Name	:	Leu		
		Birth-Year	:	1962		
		Salary	:	23K longevity	:	[0,Now]
		Dept	:	Toys longevity	:	[0,Now]
Corporality	:	[0,Now]				

Fig. 3.8 : T-Selection of EMPLOYEE (Name = Leu)

For descriptors of dynamic values, selection may be based on the value of a descriptor at a time point or in some time interval. Two examples follow.

b. Time point example: List the names of the employees working in Toys department in 1933.

The intermediate computations and results for the above query are shown below and in Fig. 3.9.

Step(1) R1 = T-Selection EMPLOYEE (Dept = Toys)
Step(2) R2 = T-Selection R1(Dept.Longevity = [33,33])
Step(3) R3 = T-Projection R2(Name)

Result Step 1	Object A	:	employee				
	Descriptors	:	Name	:	John		
			Birth-Year	:	1960		
			Salary	:	15K longevity	:	[11,50)
			Dept	:	Toys longevity	:	[11,45)
	Corporality	:	[11,50]				
	Object B	:	employee				
	Descriptors	:	Name	:	Tom		
			Birth-Year	:	1961		
			Salary	:	20K longevity	:	[0,21)
			Dept	:	Toys longevity	:	[0,21)
	Corporality	:	[0,21]				
	Object D	:	employee				
	Descriptors	:	Name	:	Leu		
			Birth-Year	:	1962		
			Salary	:	23K longevity	:	[0,Now]
			Dept	:	Toys longevity	:	[0,Now]
	Corporality	:	[0,Now]				

Result Step 2	Object A	:	employee				
	Descriptors	:	Name	:	John		
			Birth-Year	:	1960		
			Salary	:	15K longevity	:	[11,33)
			Dept	:	Toys longevity	:	[11,33)
	Corporality	:	[11,33)				
	Object D	:	employee				
	Descriptors	:	Name	:	Leu		
			Birth-Year	:	1962		
			Salary	:	23K longevity	:	[0,33)
			Dept	:	Toys longevity	:	[0,33)
	Corporality	:	[0,33)				

Result Step 3	Object A	:	employee		
	Descriptors	:	Name	:	John
	Corporality	:	Static		
	Object D	:	employee		
	Descriptors	:	Name	:	Leu
	Corporality	:	Static		

Fig. 3.9 : T-Selection (time point) and T-Projection on EMPLOYEE

c. Time interval example: List the name of the employees working in Toys department in the interval [22,33).

The intermediate queries are shown below.

Step(1)

R1 = T-Selection EMPLOYEE (Dept = Toys)

R2 = T-Selection R1(Dept.Longevity = [22,33))

Step(2) T-Projection R2 (Name)

Object A	:	employee		
Descriptors	:	Name	:	John
Corporality	:	Static		
Object D	:	employee		
Descriptors	:	Name	:	Leu
Corporality	:	Static		

Fig. 3.10 : T-Selection (time interval) and T-Projection on EMPLOYEE

A complete relational algebra for active objects in TDRM has been defined. An enhanced relational algebra for TDRM needs to be completely defined with T-Projection and T-Selection as two of the operators.

Chapter 4.

CONCLUSIONS & FUTURE RESEARCH

A model which uses the interval based approach to represent time of existence or that is the corporality of an object and its components is proposed. The relations are viewed as objects, each relation described by a meta object, and the database is viewed as a collection of objects. Each object can have descriptors which have a longevity or time of existence of either static or dynamic. The model, TDRM, incorporates the time dimension into the relational model through longevity, the length of time from an object creation to its demise (or that is until it is reincarnated into another object because of a change of a value). A query language has been proposed based upon the relational algebra.

The future research in this area can be in many directions. A prototype implementation of the model could be developed and analyzed. The complete relational algebra and relational calculus for TDRM can be developed. The definitions of key, functional dependencies and other aspects of conventional databases could be extended meaningfully to the temporal databases. The temporal dimension for incompletely specified databases requires study.

BIBLIOGRAPHY

[ALL81] J.F. Allen, "Maintaining knowledge about temporal intervals," Technical report 81, Dept. of Computer Science, The University Of Rochester, NY 14627 (January 1981).

[ALL83] J.F. Allen, "Maintaining Knowledge about Temporal Intervals," Communications of the ACM, vol. 26, no. 11, (November 1983).

[AND81] T.L. Anderson, "The Data Base Semantics of Time," Ph.D. Dissertation, University of Washington (1981).

[AND82] T.L. Anderson, "Modeling time at conceptual level," in Proceedings of the Second International Conference on Databases, Jerusalem, Israel (June 1982).

[ANT81] V. De Antonellis, and B. Zonta, "Modeling events in data base applications design," Proceedings of the Seventh International Conference on Very Large Data Bases, Cannes, France, pp. 23-31, (September 1981).

[ARI81] G. Ariav, and H.L. Morgan, "MDM : Handling the time dimension in generalized DBMS," Working Paper, Dept. of Decision Sciences, The Wharton School, University of Pennsylvania (May 1981).

[ARI83] G. Ariav, J. Clifford and M. Jarke, "Time and Databases," ACM-SIGMOD Int. Conf. on Mgt. of Data, pp. 243-245, (May 1983).

[ARI84] G. Ariav, J. Clifford and J.A. Shiftan, "A Framework for Implementing Temporally Oriented Databases," Technical Report, Dept. of Computer Applications and Information Systems, New York University, 1984.

[BEN82] J. Ben-Zvi, "The time relational model," Ph.D. Thesis, Dept. Comp. Sci., Univ. Cal. - Los Angeles, (1982).

[BOU81] A. Boulour and L.J. Dekeyser, "Time relations and event relations," Submitted for publication (July 1981).

[BOU82] A. Boulour, T.L. Anderson, L.J. Dekeyser, and H.K.T. Wong, "The role of time in information processing," A Survey. SIGART Newsl., 82, pp. 28-48, (April 1982).

[BRU72] B. Bruce, "A model for temporal reference and its application in a question answering program," Artificial Intelligence, vol. 3(1) (1972).

[BUB77] J.A. Bebenko, "The Temporal Dimension in Information Modeling," in Architecture and Models in Data Base Management Systems, G.M. Nijssen (ed.) North Holland Publishing Company, Amsterdam, The Netherlands, pp. 93-118, (1977).

[BUL60] W.E. Bull, "Time, tense, and the verb," University of California, Publications in Linguistics, vol. 19 (1960).

[CLI81] J. Clifford and D.S. Warren, "Formal semantics for time in databases," Technical Report 81/025, Dept. of Computer Science, SUNY, Stony Brook, NY 11794, (November 1981).

[CLI82] J. Clifford, "A model for historical databases," Proc. Logical Bases for Databases, Toulouse, France, (December 1982).

[CLI85] J. Clifford, "On an algebra for historical relational databases : two views," Proc. ACM-SIGMOD Int. Conf. on Mgt. of Data, pp. 247-267, (1985).

[COD79] E.F. Codd, "Extending the data base relational model to capture more meaning," Proceedings ACM-SIGMOD International Conf. on the Management of Data, (May 1979). Revised in ACM Transactions on Database Systems, vol.4(4), pp. 397-434, (December 1979).

[DER81] D. McDermott, "A temporal logic for reasoning about processes and plans," Research Report #196, Dept. of Computer Science, Yale University (March 1981).

[FAL74] E.F. Falkenberg, "Time-handling in data base management systems," Report 07/1974, University of Stuttgart (July 1974).

[FIN71] N. Findler and D. Chen, "On the problems of time retrieval, temporal relations, causality, and coexistence," Proceedings of the Second International Joint Conference on Artificial Intelligence, Imperial College, (1-3 September 1971).

[FRA72] J.T. Fraser, F.C. Haber, and G.H. Muller, "The study of time," Proceedings of the First Conference of the International Society for the Study of time, Springer-Verlag, (1972).

[FRI72] J.F. Fries, "Time-oriented patient records and a computer data bank," Journal of the American Medical Association, vol. 22 (12) (December 1972).

[GAD85] S.K. Gadia, "A Query Language for a Homogeneous Temporal Database," in Proceedings of the Fourth ACM SIGACT-SIGMOD Symposium on Principles of Database Systems, (1985).

[GAD86] S.K. Gadia, "Toward a Multihomogeneous Model for a Temporal Database," in Proceedings of the International Conference on data Engineering, (1986).

[JON80] S. Jones and P. Mason, "Handling the Time Dimension in Databases," Proc. Int. Conf. on Databases, British Computer Society. Univ. Aberdeen, pp. 66-83, (July 1980).

[KAH75] K.M. Kahn, "Mechanization of temporal knowledge," Technical report MAC-TR-155, Artificial Intelligence Laboratory, MIT (April 1975).

[KLO81] M.R. Klopprogge, "TERM:An approach to include the time dimension in the entity relationship model," Second International Conference on the Entity Relationship Approach, (October 1981).

[KRO83] D. Kroenke, "Database processing," 2nd. Ed., Science Research Associates, Inc., (1983).

[MAR78] J.M. Martin, J.P. Pointel, J. Martin, G. Debry, "The notion of time in medical records - structure of chronology, validation and calculation of a time interval," Methods of Information in Medicine, vol. 17(2), pp. 90-95, (April 1978).

[MAT67] R. Mattison, "A formal system for the logical analysis of temporal relationships between intervals of time," Memo. RM-5279-PR, Rand Corporation, Santa Monica, CA (April 1967). .IP 3. R. Mattison, "An introduction to the model theory of first-order predicate logic and a related temporal logic," Memo. RM-5580-1-PR, Rand Corporation, Santa Monica, CA (June 1969).

[NEE81] P. Needham, "Temporal intervals and temporal order," Logic and Philosophy, (1981).

[ORT82] E. Ortowska, "Representation of temporal information," International Journal of Computer and Information Sciences, vol.11, no. 6, (August 1982).

[OSB86] S.L. Osborn and T.E. Heaven, "The Design of a Relational Database System with Abstract Data types for Domains," ACM Transactions on Databases Systems, vol. 11, no. 3, pp. 357-373, (September 1986).

[RES71] N. Rescher and A. Urquhart, Temporal Logic, Springer Verlag, New York (1971).

[SNO84] R. Snodgrass, "The Temporal Language TQUEL," Proc. 3rd ACM SIGACT-SIGMOD Symp. on Principles of Database Systems, Waterloo, pp. 204-212, (April 1984).

[TSO81] J.K. Tsotsos, "Temporal event recognization: an application to left ventricular performance, "Proceedings of the Seventh International Joint Conference on Artificial Intelligence, Vancouver, British Columbia, pp. 900-907, (August 1981).

[ULL82] J.D. Ullmann, "Principles of Database Systems," 2nd Ed., Computer Science press, (1982).

[UNG78] E.A. Unger, "A natural model for concurrent computation," Ph.D Thesis, (1978).

[VAI84] J.H. Vaishnav, "A Homogeneous Temporal extension of QUEL," M.S. Thesis, Texas Tech. Univ., (1984).

[WIE75] G. Wiederhold, J.F. Fries and S. Weyl, "Structured organization of clinical data bases," Proceedings of the AFIPS National Computer Conference, (1975).

[YAM79] G.E. Yamami, "The Semantics of Time Series Data Modeling," Master's Thesis, Dept. of EECS, University of California, Berkeley (December 1979).

TIME DIMENSION IN THE RELATIONAL MODEL

by

YERRAPRAGADA CHAYA

M.Tech., (Comp. Sc.), Osmania University, 1985, India

AN ABSTRACT OF A THESIS

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Computing and Information Sciences

KANSAS STATE UNIVERSITY

Manhattan, Kansas

1987

ABSTRACT

To increase the semantic content of a database means to store more information in the database about the meaning of the data. There are different dimensions or attributes of information that can be incorporated. One attribute, time, is of interest to many, especially the initiation or creation time of something. Representing knowledge about time, i.e., temporal knowledge, is the subject of this thesis.

Databases supposedly model reality, but conventional databases are lacking in capability to record and process time varying aspects of the real world. A proper understanding and a consistent view is required, to represent temporal knowledge, to model reality completely and to allow the answering of queries involving a temporal aspect.

This thesis proposes a model to incorporate temporal information of the real world into the relational database model. Time is introduced by using the corporality concept of objects. A distinction between the active and the inactive objects is made. Some aspects of manipulation of data with temporal objects are explored.