Semantics of Temporal Models

With Multiple Temporal Dimensions

Peter Kraft and Jens Otto Sørensen

Aarhus School of Business, Denmark

Email: p-g-kraft@mail.dk; jos@asb.dk

Abstract: Semantics of temporal models with multi temporal dimensions are examined progressing from non-temporal models unto uni-temporal, and further unto bi- and tri-temporal models. An example of a uni-temporal model is the valid time model, an example of a bi-temporal model is the valid time/transaction time model, and finally an example of a tri-temporal model is a model with valid time, decision time and transaction time dimensions. The semantics are at first specified at the highest conceptual level possible by a graphical notation. Next, the conceptual models are translated into lower level models ending up with lexical data models. In particular we look upon the representations by sets of normalised tables, by sets of 1NF tables and by sets of N1NF/nested tables. At each translation step we focus on how the temporal semantic is consistently maintained. In this way we recognise the requirements for representation of temporal properties in different models and the correspondence between the models.

The results rely on the assumptions that the temporal dimensions are interdependent and ordered. Thus for example the valid periods of existences of a property in a mini world are dependent on the transaction periods in which the corresponding recordings are valid. This is not the normal way of looking at temporal dimensions and we give arguments supporting our assumption.

Keywords: Temporal dimensions. Multi temporal dimensions. Temporal conceptual models. Temporal data models. Lexical temporal models. Normalised temporal models. 1NF temporal models, N1NF temporal models. High-level conceptual models. Translation of models.

Content

1. Introduction

The paper explores the semantic properties that should be maintained consistently in temporal models and how different types of models should be constructed and constrained for this purpose.

Temporal models may work with several time dimensions. Well known are the uni-temporal model with only a valid time dimension and the bi-temporal model with a valid time and a transaction time dimension. However we search for general principles of how several time dimensions should be handled and how their semantics should be understood. We take a closer look initially on non-temporal models and then successively on uni-, bi-, and tri-temporal models. An example of the latter is a model with one dimension for assumed valid time properties, one decision time dimension measuring decision periods, the periods in which assumptions are unchanged, and one transaction time dimension for measuring periods in which recorded information is unchanged. Such a tri-temporal model may be used for budgeting and planning systems where one wants to keep track of the changes of budges and plans over time.

The semantics of the temporal properties are best understood and maintained by conceptual models. However, implementation maintaining temporal information will be done using some kind of lexical data models e.g. relational models like SQL2 or SQL3 allowing also for nested tables and object-oriented facilities. Using lexical models to examine temporal properties is both hard and complex. Sets of special constructs should be used and complicated constraints should be accounted for to guard a consistent handling of the temporal semantics. We look closer into models consisting of sets of normalised tables, of sets of 1NF tables, and of sets of N1FN tables.

We pursue an idea of high-level conceptual models by which selected types of properties can be described directly and handled consistently. After having used a high level model to examine the property types in question we suggest rules for translating the high level descriptions into descriptions by lower level models. The translation rules will advice on how to do the constructions on the lower level and the constraints to be guarded. Eventually we reach descriptions for lexical models.

The following section concentrates on the assumptions on which our examinations rely. In section three we introduce a special version of the ER model that we use as a high-level non-temporal conceptual model and we show how it translates into normalised sets of tables. Section four introduces the uni-temporal model and shows how it translates first into the ER model and next into sets of normalised tales, sets of 1NF tables and sets of N1NF tables. Section five and six show how bi- and tri-temporal models respectively translate into sets of normalised tables and sets of 1NF tables. Finally section seven concludes the paper and suggests further examinations.

There is a rich literature on ER models and temporal models. In connection with the sections we comment on some of that.

2. Assumptions

Our examinations rely on two independent major assumptions: One is concerned with how several time dimensions are interrelated and how extensions in time are conceptualised. The other concerns the degree to which conceptual models capture given properties and how they translate into other models.

Assumptions on the time dimensions

The kind of time dimensions that we are interested in are e.g. valid time and transaction time. As an example of a third time dimension we could have a decision time that measures periods in which decisions about phenomena in the mini world is not changed.

We regard this kind of time dimensions as not entirely independent and they can be ordered. Let us look closer at the tri-temporal example with the dimensions stated above. First there is the valid time dimension that measures periods in which phenomena in an actual mini world exist. Next decision time dimension measures the time at which decisions on the existence of phenomena in the mini world are made and periods in which the decisions are not renewed. Finally transaction time dimension measures the time information on decisions and the existence of phenomena is recorded and the periods in which the registrations are not regretted. Thus decision periods determine valid time periods and transaction time periods determine decision periods and thereby valid time periods.

This of course does not rule out that one can draw a volume in a vector space with orthogonal axis depicturing the information. The dimensions form an orthogonal set of axis but they are never the less dependent in the sense just described meaning that not any volume forms a legal set of events.

Another concern is how periods should adjunct to each other. A first recording of a property is made at a given transaction time. Later the recording can be regretted and changed and so on until the last change that will be valid now and until further changes. The transaction time periods for the property in question should adjunct continuously from the first recording until 'now'. Similar is the case for recording of decisions. Decisions are current until they are changed at later times. For existences in valid time there is no structural requirement for time continuity. When such a request is made it will be due to semantically circumstances in the mini-world.

Initially on a conceptual level we regard time as measured in a continuum. In implemented temporal models time measures are represented by discrete attribute values. We will not discuss the various possibilities of implementing these attribute values but only assume, that it is possible to define an abstract data type (ADT) that implements periods and that the period-ADT is equipped with functions etc. that corresponds to measuring time in a continuum.

A *period* is an anchored duration of time as opposed to an *interval* which is a duration of time. A *point* in time is regarded as a period with a duration of time equal to zero.

Assumption on conceptual models and translations into other models

The concept of a conceptual model is normally rather vague. One assumes a model with a schema in which the symbols may associate to concepts of a mini world. We will use a more restrictive definition of what we call a high level model.

First with regard to the extensions: A high level model for given selected property types maps these properties uniquely and precisely. That is each element, recognised in the mini world in question is mapped only once and no other element are included in the mapping. Further the properties of the elements are maintained automatically guarded by the syntax defined for the model. Thus the mapping will always be consistent and complete with respect to some mini world.

Next we look at the intensions: When the mapped elements are classified as belonging to given types a schema may be specified with symbols for each element type. If further we can shape the symbols and their semantics such that the specific property for each element type are reflected in the schema, then the symbols of the schema correspond to concepts applicable in the mini world. The latter has a memento: In classifying elements we may regard other properties than those mapped into the high level model. Appropriated naming of the symbols in the schema may hint at those other anonymous properties.

We use a high level ER model that only guards existence of entities and their mutual relationship at the high level as a non-temporal model, and a high level conceptual uni-temporal model that guards the existences in valid time. We are not able to design high-level temporal models having several time dimensions due to limitations on what we can draw.

A high level model may be translated into lower level models. This implies that further elements are added to the translated model and constraints should guard the consistent handling of the high level properties. We shall show how the high level temporal model translates into an ER model that will be on a lower level with respect to the mapping of temporality, and how it translates further into different lexical models, e.g. an SQL schema, implying further elements and constraints.

Finally we would like to mention some guidelines used in our design of models. Often it is possible to conceptualise a given problem area in different ways. When this is the case we try to use as primitive and general concepts as possible in the model, which again imply that the given problem area will be mapped in one way only in the model.

We use these ideas twice. In our design of relationship symbol of the ER model we decompose complex relationships into entities and sets of simpler relationship ending up with the simplest possible relationship symbol.

The other place is concerned with identities of entities and their relationships. The question is whether we can regard the same entity or relationship to exist in different separated periods or whether we should regard entities and relationships only to exist in one continuous period. We chose the latter as the most primitive and general concept. Assume an entity with given properties, that is an entity with given attribute values related with given other entities, exists in one period, and an entity with the same properties also exists in another period, then we regard these existences as two entities each with their own identity. The case is accentuated in models with several time dimensions. Assume for example a bi-temporal model with a valid time and a transaction time. In a first transaction period is recorded that a new entity exists in a given valid period, in the next transaction period the existence of the entity is regretted, and in the following transaction period it is regretted that the existence of the entity was regretted and it is thus reintroduced. Also in such a case we assume two entities each with their own identity.

3. *Non-temporal models*

We introduce a high-level ER model as a non-temporal conceptual model and state the rules for translating the ER model into the lower level relational model. In the following sections the ER model is used as an intermediate model into which we translate conceptual temporal models. We end these sections with a comparison of our ER model with some Chen-like ER models because these are often used as a base for adding temporality on a conceptual level.

The high-level ER model

The model maps the existence of entities and their mutual relationships at a high level. The extensions are kept in a binary graph, where each existing entity is mapped once as a vertex and each relationship once as an edge. The syntax of a graph maintains the property that relationships are in between entities.

Basically we use only two symbols in a schema: a "box" for entity types and a "craw foot" for a simple relationship types with a mandatory one-side and an optional many-side. The semantics of the relationship symbol is the simplest possible. You may always conceptualise a complex relationship as an entity related by a set of simpler relationships to the entities related by the original relationship. In doing so you eventually reach a state that cannot be decomposed further. The result will be a relationship type with a semantic corresponding to the semantic of our simple relationship symbol.

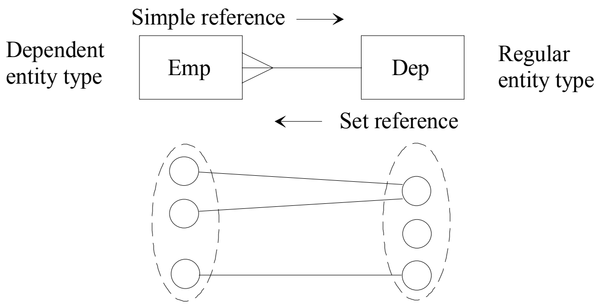


Figure 3.1. Semantics of the basic symbols

Figure 3.1 shows the basic symbols together with a sample of extensions. The Emp-entities on the many-side of the relationship symbol must each refer to one Dep-entity. The Emp-entities are *dependent* on the existence of Dep-entities to refer to and have *simple references* to Dep-entities. The Dep-entities on the one-side of the relationship symbol can refer to a set of Emp-entities, where the set may possibly be empty. Thus Dep-entities are *regular* and have *set references* to sets of Emp-entities. Applying the symbols of the schema as concepts of some mini world we have described that each employee is always attached one and only one department and that departments can have from zero to many employees attached.

We shall later talk about the *identity* of an entity and a relationship. In our ER model an entity or a relationship has an identity simply by being mapped as respectively a vertex and an edge in the graph. You can always distinguish between the entities and the relationships by pointing at the respectively elements in the graph.

Attributes are regarded as special entities. Besides their existence attributes have "visible values". We assume that for every attribute type all possible attribute values exist a priori and occur only once in the extension. Further, attribute values are only referred to by simple references and will thus be absolute regular.

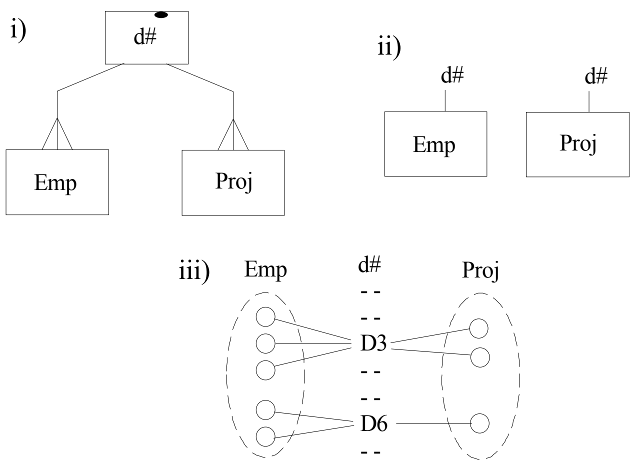


Figure 3.2. Semantics of attributes

In Figure 3.2 at i) the attribute type, d#, is diagrammed as an entity type with a black mark. (The mark indicates that attributes are identified by their values). As an alternative that puts lees load on the diagram, attributes are also diagrammed directly as shown at ii). At iii) in the figure the sketched extensions indicate the existence of all possible d# values and show references to the some of these.

When we regard the ER model as a high level model we only account for the existence of attribute values and references to these values. Later we shall apply meanings to the values, in cases temporal meanings, introducing functions and similar for use on the values. When doing so the ER model will not be on a high level with respect to the properties maintained by the attributes.

Only entity types including attribute types are named in the schema. Relationship types are referred to implicitly by path expressions leading from one entity type through a diagrammed relationship type to an entity type. If necessary because more than one relationship type is diagrammed between two entity types, or for explanatory purposes, an entity type may be renamed as it is seen through a given relationship type.

Corresponding to the property of the existents of entities and their relationships the derivation rules for the graph is defined as an equivalent class where relations are reflective, symmetric and transitive. The rules imply that we can regard entity types as more or les complex relationship types. With the naming conventions we can use the names of entity types as variables, state path expressions that interrelate entities across a model, and construct languages by which given condition can be stated and by which entities representing given specified relationships across a model can be generated.

Figure 3.3 shows two different complex relationships types that both imply many-to-many relationships. In the conjunctive relationship a) of third degree there is a Job-instance for every relationship between an Emp-entity, an Proj-entity and a time-values. (Remember that attributes are regarded as entities). In the disjunctive relationship b) of second degree each Dep-instance relates a set of Emp-entities with a set of Proj-entities.

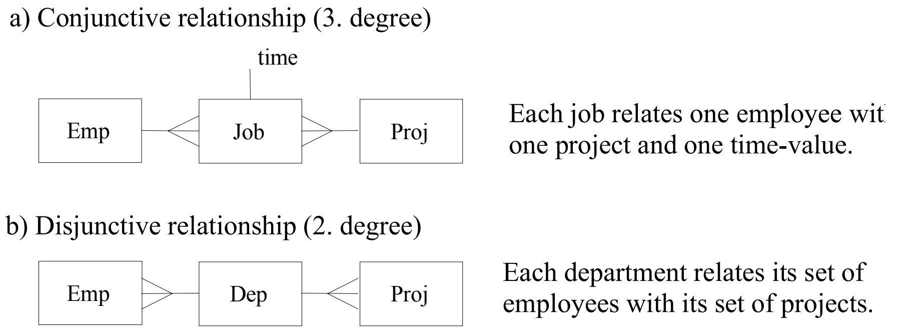

a) Conjunctive relationship (3. degree)
time
Emp
Job
Proj
Each job relates one employee wit
one project and one time-value.
b) Disjunctive relationship (2. degree)
Emp
Dep
Proj
Each department relates its set of
employees with its set of projects.

Figure 3.3. Complex relationship

By means of the entity type symbol and the relationship type symbol, we can specify a model corresponding to those entire concepts for which we want to keep information. The semantics of the relationship symbol imply a uniform way of conceptualising into entity types and simple relationship types, and regarding attributes as entities where all possible values exists a priori implies a simple uniform model. The model is on a high level with respect to the existence of entities and their relationships. The symbols of the schema will always correspond to concepts applicable on a mini world and the information kept in the graph will always be complete and consistent corresponding to existences in a possible mini world.

We will sometimes talk about the *properties of an entity type* by which we refer to the entities themselves and their simple references to other entities including their attributes.

The freedom allowed by using the diagrammed symbols will often be limited by given business rules or the like. We may constraint a model such that no information entered into a model is in conflict with business rules. Business rules as such are not guarded by the graph, and we cannot guarantee that any specified constraint will correspond to some applicable business rule. We may even specify constraints that are mutual inconsistent.

Key and cardinality constraints can be diagrammed. Other constraints are specified explicitly by predicates. Keys that identify entities are diagrammed by marks in front of simple references. At a) in Figure 3.4 is diagrammed a composed key by black marks. If alternative keys are specified other types of marks can be used. At a) a Job-entity is identified by the Emp-entity and the Proj-entity referred to. We emphasise that constraints add to the precisions of the modelled concepts. At a) a job will thus comprise the total time an employee uses on a given project – there can be only one job per employee per project. At c) a job can just comprice any time an employee uses on a project. At c) we have added some part of the identification system used by a company to the model. Employees are identified globally in the company by an e#, and projects are identified locally within each department by a p#.

Cardinality constraints are used to restrict the number of entities of given types and to restrict the many-sides of simple relationships. In Figure 3.4 b) the ball on the relationship symbol restricts the many-side to be zero or one. The semantic implication is that there will now be an optional reference from an Emp-entity to Dep-entity, so that an employee is possibly attached one department. Another implication is that the simple reference from the Attached-entity type to the Emp-entity type will be a non-composed key. The employee in question identifies an attachment. We will show more examples of cardinality constraints later.

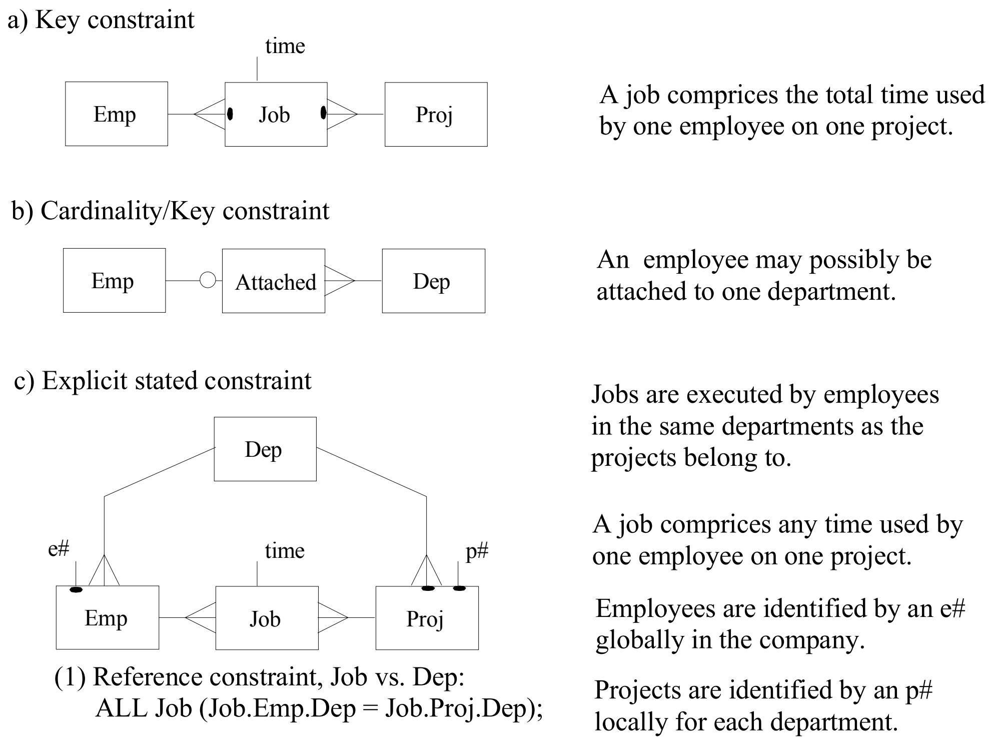

a) Key constraint
time
Emp
Job
Proj
A job comprices the total time used by one employee on one project.
b) Cardinality/Key constraint
Emp
Attached
Dep
An employee may possibly be attached to one department.
c) Explicit stated constraint
Dep
e#
time
p#
Emp
Job
Proj
Jobs are executed by employees in the same departments as the projects belong to.
A job comprices any time used by one employee on one project.
Employees are identified by an e# globally in the company.
Projects are identified by an p# locally for each department.
(1) Reference constraint, Job vs. Dep:
ALL Job (Job.Emp.Dep = Job.Proj.Dep);

Figure 3.4. Constrained models

Figure 3.4 c) also shows the predicate for the explicit specified constraint (1). In the notation we spell the key-symbols in full and take advantage of diagrammed paths.

The predicate says that for all Job-entities the path leading from Job via Emp to Dep should meet the same Dep-entity as the path leading from Job via Proj to Dep. The constraint is called a reference constraint. It implies that for all jobs the employee and the project should belong to the same department.

At this point we may notice that also the constraints introduced so fare adhere to the existence-property of the high-level ER model. They don't account for specific meanings of attribute values other than their existences and will thus only restrict the information about existences that is adopted by a model.

In the next section on temporal models we need to apply specific meaning of attribute-values in order to use the ER model for a safe handling of temporal properties. That is, the ER model is used for - or programmed for - handing other types of properties that the properties of pure existence. With respect to these other properties the ER model will, as mentioned above be, on a lower level.

Translation of ER models into relational models

Initially we should be aware of the medium in which the information is kept by respectively an ER model and a relational model. In ER models we use a two-dimensional graph and in relational models a one-dimensional lexical description. In a lexical description we may apply meaning to the symbols, to the order in which symbols are stated, to groups of symbols and to the order in which groups of symbols are stated. That is, the most complex structure that maps on a high level in lexical described models are nested structures. When information concerning more complex structure is described lexically one needs recognisable marks in the lexical sequences to tie different parts of the descriptions together. In the relational model repeated use of attribute values serve this purpose.

The translation of a model into another model requires rules or methods by which any legal structures of the former maps into legal structures of the later. The following steps specify the translation of our ER model into a relational model:

Step 0: In order to provide for foreign key references in a relational model every entity type referred to by simple references should have a key. Entities not specified with a key is added a key-attribute possibly some kind of artificial surrogate.

Step 1: A table corresponding to each entity type is specified. The name of the table can be the same as the name of the entity type.

Step 2: Attributes specified for the entities are copied to the corresponding tables. Attribute name need not be changed. If some of these attributes constitute keys corresponding primary keys are specified for the tables.

For each entity having simple references foreign key attributes are added to the corresponding tables. The foreign key attributes are inherited from the primary key-attributes of the tables corresponding to the entity type referred to. If the simple references in the ER model are key components the foreign key attributes are included in the primary key of the table in question. Thus, cascading through several entity types can possibly be needed for inheriting foreign key attributes. Renaming of foreign key attributes can be necessary.

Step 3: Constraints in the ER model is accounted for. By step 2 some key constraints are maintained as the primary keys. Alternative keys in the ER model are transferred as alternative keys for the tables.

Reference constraints can in special cases, depending on the adopted identification system in the mini world, be directly maintained in the relational model. When attributes are inherited by cascading the same key attributes may in some cases be inherited more than once. Omitting the double inherits will guard the reference constraints.

Other constraints require other means like check-clauses or awareness of the constraints in the usage of the tables.

The set of tables reached at this stage is complete. When the ER model is well structured the tables will be normalised. The tables do not allow undefined attribute values (NULL-values).

Step 4: Tables that correspond to entity types in between which there are optional one-to-one relationships can collapse into a common table. (The tables have common primary keys). The common table should be based on the table corresponding to the regular entity type and should absorb attributes from the table corresponding to the dependent entity type in the relationship. NULL-values should be allowed for the absorbed attributes. Foreign key specifications and other constraints on the absorbed attributes should also be maintained.

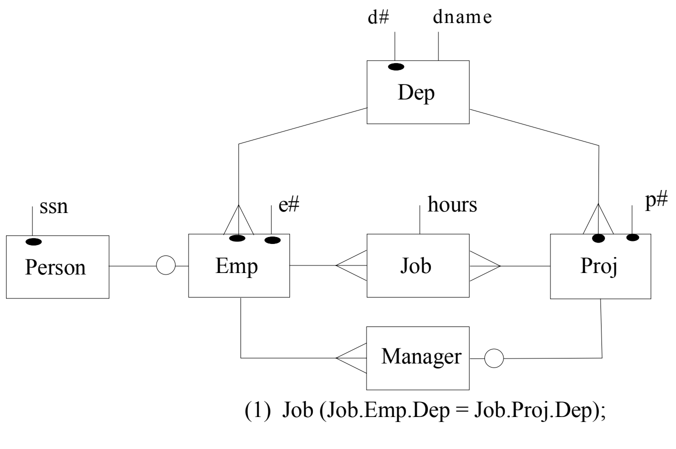


Figure 3.5. Company case

Figure 3.5 shows an ER model for a company that we translate into a relational model. We have already commented on most of the semantics of the model and can add that the person may occur as an employee, that a project can possibly have a manager, and that the same employee may occur as a manager several times.

Step 0: The ER model accounts for the identification system of the company. Numbers local to departments identify employees and projects. Being a given person alternatively identifies an employee. Managing a given project identifies a manager. The Job-entity type is not referred to by simple references and needs no key. Thus the requirements of keys are satisfied.

Step 1 and 2: The tables after step 2 is shown below. We use a short notation where primary keys, alternative keys and foreign keys are specified by respectively PK-, ALTK- and FK-clauses. An AS-clause is used to rename attributes.

```
1)  Dep (d#, dname) PK (d#);

2)  Person (ssn) PK (ssn);

3)  Emp (e#, d#, ssn, d#) PK (e#, d#),
        ALTK (ssn),
        FK (d#) REF Dep,
        FK (ssn) REF Person;

4)  Proj  (p#, d#) PK (p#, d#),
        FK (d#) REF (Dep);

5)  Job (hours, e#, d# AS emp-d#, p#, d# AS proj-d#)
        FK (e#, emp-d#) REF Emp,
        FK (p#, proj-d#) REF Proj;

6)  Manager (e#, d# AS emp-d#, p#, d# AS proj-d#) PK (p#, proj-d#),
        FK (e#, emp-d#) REF Emp,
        FK (p#. proj-d#) REF Proj;
```

Step 3: The reference constrain (1) for the ER model can in this case due to the identification system be acknowledged by removing one of the d# attributes from the Job-table. Table 5) is replaced by table 5a) below:

```
5a) Job (hours, e#, d#, p#)
        FK (e#, d#) REF Emp,
        FK (p#, d#) REF Proj;
```

Step 4: The Proj-entity type has an optional relationship with the Manager-entity type. The Manager-table can therefore be absorbed by the Proj-table. Table 4) and table 6) is replaced by table 4a) below. (We use the OPT-clause to specify that NULL values be allowed):

```
4a) Proj  (p#, d#, e#, d# AS emp-d#) PK (p#, d#),
        OPT (e#, emp-d#),
        FK (d#) REF (Dep),
        FK (e#, emp-d#) REF Emp;
```

Comparison with Chen-like ER models

The high level ER model introduced above is described in detail in a Danish book titled “En uniform entity-relationship model” (Kraft and Sørensen 2001)and sketched in an English paper (Kraft and Soerensen 1998). In this subsection we call it the UER model. The UER model is also equipped with constructs capturing concepts for specialisations/generalisations and aggregation, and a language closed around the UER model by which derived entities and relationships are related to the base model operated upon. The language is used to construct loss less transformation of models and for specifying information structures to be used in interfaces. The language is also used as a query language.

We shall only make some remarks on how the UER model deviates from P. Chen's ER model (Chen 1976) and later variants of this model e.g. (Teory 1990; Batini, Ceri et al. 1992; Embley 1997; Elmasri and Navathe 2000; Garcia-Molina, Ullman et al. 2002), where we call these ER models for Chen-like models:

Both the UER model and the Chen-like models have a graphical diagrammed schema. The extensions of the UER model described by a graph conform to its graphical schema. The description of extensions in the Chen-like models is seldom explicitly discussed. P. Chen (Chen 1976) assumes that the extension is constituted by lexical relations/tables where tuples are identified by anonymous surrogate attributes implying a mismatch against the graphical schema. Embley (Embley 1997) though, illustrates an extensional sample by a graph.

The UER model regards attribute values as entities referred to from entities. In Chen-like models entities as well as relationships comprise assigned attribute values. The same value occurs several times like in the relational model. The UER model has two basic element types, entities and relationships, Chen-like models have three, entities, relationships and attributes.

In the UER model the relationship symbol specifies basically a binary one-to-one-many relationship with a mandatory one-side and an optional many-side. The relationship type cannot be decomposed. Complex relationships are modelled as entities related with other entities. In Chen-like models the relationship symbol - normally diagrammed as a diamond - specifies relations of any degree from two and up with many-to-many relationships in between the involved entities, the entities being mutual regular. There is no rules determining what should be modelled as relationships or not.

In the UER model constraints are used to limit the information adopted by a model and thus add to the semantics of the diagrammed symbols. Cardinality and key constraints are diagrammed; other constraints are explicitly specified by predicates. In Chen-like models it is not that easy to determine the purpose of constraints, they often refer implicitly to the underlying extensions in form of relational tables. Relationship types can have cardinality restrictions. In the EER model (Elmasri and Navathe 2000) cardinality constraints are used to restrict the number of times an entity of a given type can/shall join a relationship of a given type. Decomposing such a relationship type will result in an entity and a set of relationships with a semantic like the relationship symbol of the UER model. In other models cardinalities specifies how many entities of a given type can be seen from an entity of the other types through the relationship. The interpretation of the latter may be uncertain in cases where the degree of a relationship is greater than two. In (Teory 1990) the cardinality specification is used to state functional dependencies in between the related entities.

In Chen-like models marked attributes specify keys. Both entities and relationships can have key-attributes. This is in accordance with extensional tables where the table should have a primary key. There are however some exceptions. Often relationships are determined by the related entities, thus the related entities constitute a composed key for the relationship. Chen-like models may include a concept for a weak entity type. A weak entity type is dependent on another entity type and has a key component in a form of a relationship with this other entity type, e.g. (Chen 1976; Garcia-Molina, Ullman et al. 2002). Double frames mark both a week entity type and the involved relationship types. In (Batini, Ceri et al. 1992) a cleaner diagramming of key component substitute the need for week entity types. In the UER model week entity types have no special position; the semantics of the relationship symbol specify which entities are dependent on which entities and the semantics of keys is something else.

In our use of the UER model emphasise first of all its conformity between the graph-extensions and the diagrammed schema. It allows us to define the model as a high-level model. Next, regarding attribute values as entities and using only the simple relationship types in the diagramming, free us for considering sets of variants that would be the case if we had used Chen-like models. Last but not least the UER model eases the specification of complex constraints that will be of major importance when examining temporal models. It is a model with simpler symbols that other ER models. Yet it is not ambiguous and can capture more meaning.

4. Uni-temporal models (Valid time models)

Uni-temporal models map existences in the mini world as they occur in time – valid time. Every entity and every relationship exists in given possible limited lifetimes between genesis and eternity. To study the properties of uni-temporal models we first suggest a high-level uni-temporal model on which we may add some temporal constraints. Next we examine rules that translate high-level temporal models into ER models emphasising the constraints needed for the consistence handling of the temporal properties. We translate further into some lexical models: a normalised relational model, a 1NF relational model, and a N1NF or nested model. Finally we compare the results obtained with some other suggested models capturing valid time properties.

A high-level uni-temporal model (The temporal base model)

The high-level temporal model builds on the idea introduced for the high-level ER model. The extensional graph of the ER model is expanded with a continuous time dimension thus mapping existences in time consistently. We assume that the corresponding concepts are stable over time and preliminary that the concepts are applicable on the mini world at any one instance in time. For such an *instance time model* we may use a schema with the same set of symbols and constraints that apply to the ER model. We define that entities exist in uninterrupted periods and that simple reference may vary referring to different entities during their lifetime. Attributes that exist a priory exist at any time. We diagram that entities and relationships can have explicitly limited lifetimes by a *valid period mark*, and use a double stoke for this purpose.

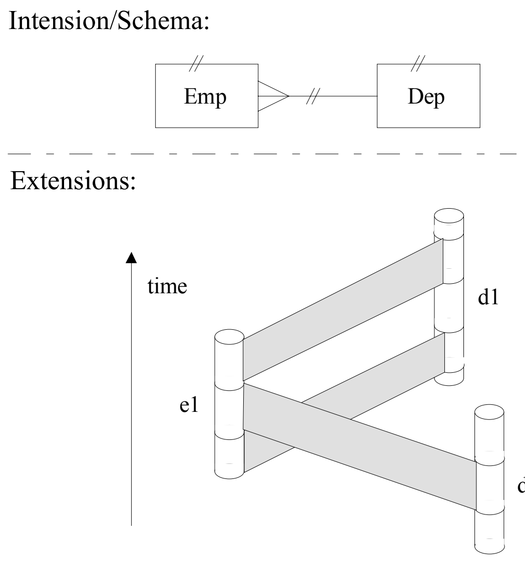


Figure 4.1. High-level uni-temporal model, semantics of basic symbols

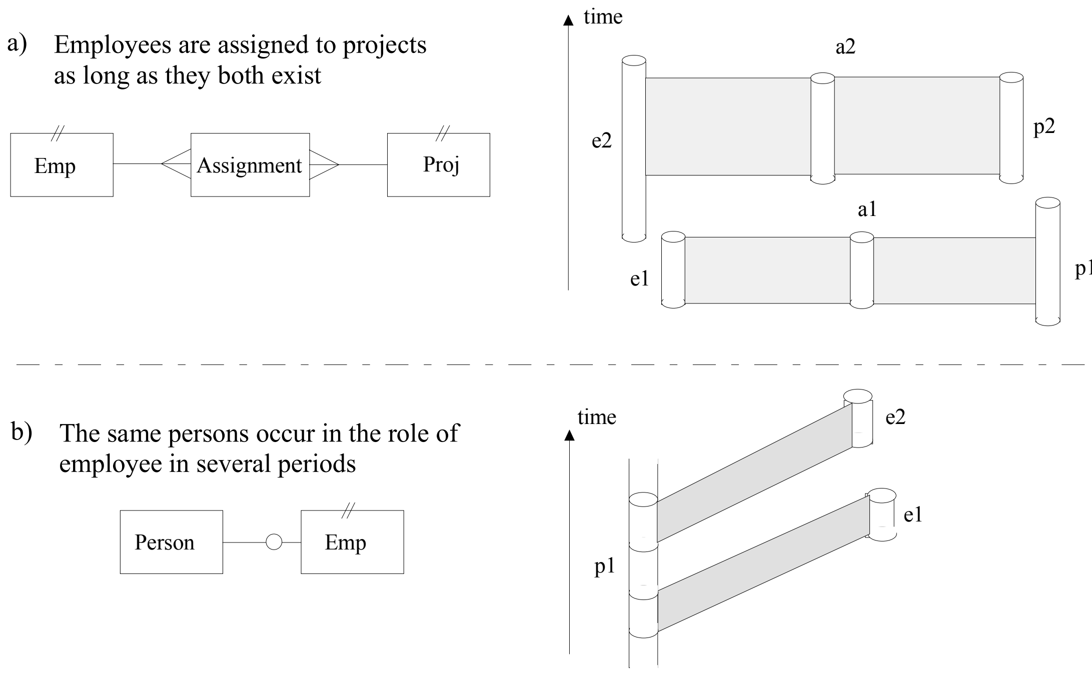
Figure 4.1 illustrates a schema of a high-level uni-temporal model and shows a sample of the three-dimensional extensions. The sample shows the lifetimes of one employee and two departments, where the employee at first is attached to one department, then to another department and finally attached again to the first department.

The syntax for the three-dimensional mapping is such that the consistency of temporality cannot be compromised. Pillars parallel with the time-axes map existing entities and curtains orthogonal to the time-axe map existing relationships. The semantic of the relationship symbol requires, that there at any instance in time during the lifetime of a dependent entity is a simple reference referring to an existing regular entity. In Figure 4.1 the dependent employee is in his entire lifetime attached one but not necessary the same department, while the regular departments need not to have attached employees all the times.

In an instant time model there are some restrictions on the expansions in time for both dependent entities and for simple references. A dependent entity can at most exist during the lifetime of the corresponding regular entities referred to. A simple reference can at most exist as long as the corresponding dependent entity exists. When the lifetimes for entities and relationships are expanded as far as possible they have *derived lifetimes*. We constraint lifetimes as derived by omitting the double strokes in the diagram. Please note that a simple reference with derived lifetime is *fixed* as opposed to a not constraint simple reference that is *varying*. When an entity type is absolute regular its derived lifetime is from genesis to eternity. Attribute values that are absolute regular exist at any time.

At a) in Figure 4.2 the assignment entity type and the two relationship types have derived life times; they are diagrammed with no double strokes. The lifetime of an assignment is limited by the intersections of lifetimes for the actual employee and project. In the figure the lifetime of assignment a1 is determined by the lifetime of employee e1, and the lifetime of assignment a2 is determined by the lifetime of the project p2.

At b) in the figure the person-entity type is absolute regular with the derived lifetime from genesis to eternity. A Person-entity can at any time be related with at most one Emp-entity and the relationship will not change during the lifetime of the related Emp-entity. That is a person can in different periods have the role of an employee and will occur as different employees in each period. In the figure person p1 is in one period employed as employee e1 then there is a vacant period and next p1 is employed as employee e2.



a) Employees are assigned to projects as long as they both exist
Emp
Assignment
Proj
time
a2
e2
p2
a1
e1
p1
b) The same persons occur in the role of employee in several periods
Person
Emp
time
e2
e1
p1

Figure 4.2 Derived lifetimes in instance time models

We terminate our discussions on instant time high-level uni-temporal models by showing a temporal schema for the company case.

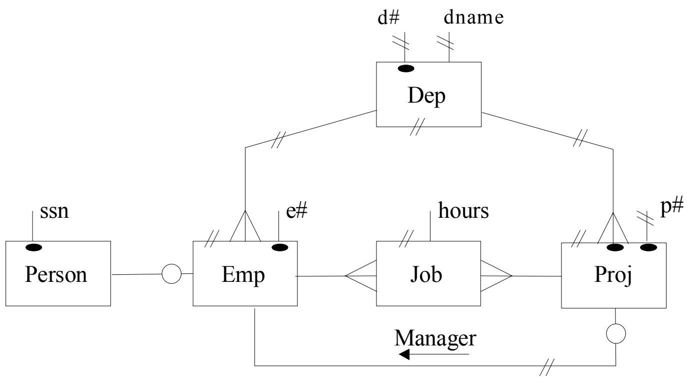
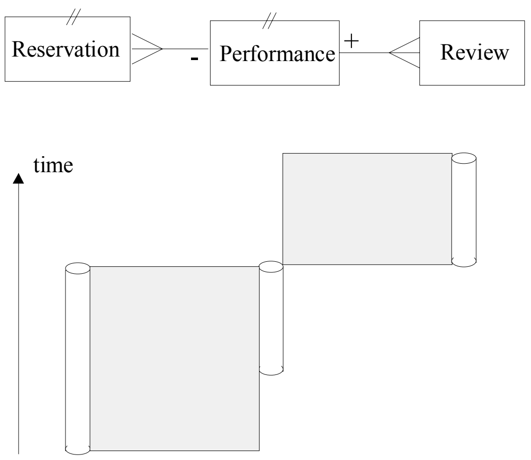


Figure 4.3. Company case, a time instance model

Figure 4.3 shows the schema. From the concepts described by the schema we read: A company may have several departments. The departments have at any time a unique d# and a not necessarily unique dname, where both d# and dname may vary during the lifetimes of the departments. An employee is at any time attached to one department and can be moved from one department to another. Employees have in their lifetime an unchangeable unique e# as key. An employee is a person. We are not concerned with the lifetime of persons assuming they exist forever, but we have modelled that a person can appear as different employees in different periods. Projects can also move from one department to another. Projects have a changeable p# which at any time should be unique within the department that the project currently belongs to. Projects have at any times an employee as manager, but not necessarily the same employee in the lifetimes of the projects. Jobs are arbitrary work executed by one employee on one project. Jobs run over limited lifetimes and accounts for a number of used hours.

In a time instance model, the concepts displayed by a schema should be applicable on a mini world at any moment in time. This implies that relationships can only exist in periods where also the related entities exist. In some cases however, it will be necessary to interrelate entities beyond the periods of their existence. For example it will be natural to relate a reservation for a performance with the performance in question. The period in which a reservation should be valid would properly be from the time it is made to the time when the performance ends, while the period for the performance is the period in which it takes place.

To cope with this kind of problems we constraint simple reference such that dependent entities can refer to entities in periods that exceed the lifetimes of the entities referred to. We allow *non-instance time references*. In a schema we use marks on the one-side of the relationship symbol, a plus '+' and a minus '-' sign to indicate that the periods of references can exceed respectively the upper and the lower limits of the lifetime of the entities referred to.



A reservation is made before the performance, and
a review of the performance is made after the performance.

Figure 4.4. Model with non-instance time references

Figure 4.4 illustrates the case where reservations are made before the performance takes place and reviews are made after the performance.

The high-level uni-temporal model describes the mini world by concepts immediately applicable on the mini world. In the following where we translate the high-level model into lower level models and also when we introduce further time dimensions, we need to refer to the concepts of the high-level model. For ease we call the high-level uni-temporal model for a *temporal base model* and its entity types for *temporal base entity types.*

Translation of high-level uni-temporal models into ER models

In order to map the continuous expansion in time into an ER model, we must introduce some extra constructs. We introduce an abstract data type, say P, to capture expansions of periods in time. For P we assume some functions, relations, and constants. Among these:

Start of period P:	BeginP (P)
End of period P:	EndP (P)
P1 and P2 coincide:	P1 = P2
P1 included in P2:	P1 IN P2
P1 and P2 overlap:	P1 OVERLAP P2
Constant now and forward:	Now

We do not consider possible granularities for measuring periods, but we want that certain conditions hold when using the functions:

An event with only a point in time:	BeginP (P) = EndP (P)
P2 adjunct to P1:	EndP (P1) = BeginP (P2)

Below we show how different schema patterns from temporal models translate into ER models. Our first concern is the instance time model, where we distinguish between patterns with fixed simple references and patterns with varying simple references.

Figure 4.5 shows patterns with fixed simple references. At 1) all lifetimes are derived. There will be no explicit need for information about lifetimes after translation into an ER model. In other words entity types and relationship types that have no period marks (double strokes) in the temporal model translate into a corresponding ER model with no explicitly information about periods.

At 2) the lifetimes of the entities are limited. In the translated model, we use the P-attribute to capture lifetime. Further, we have to specify a *period inclusion constraint* (1) which guards that the lifetime of the dependent A-entities are included in the lifetime of the entities referred to. In the figure we have shown both an informal and a formal specification as the constraint (1).

At 3) the simple reference from A to B is the key of A. At any time, only one A-entity should refer to a given B-entity. However, over time several A-entities could refer to the same B entity. In the translated model we get a non-restricted set reference from

B- to A-entities, and a key for A-entities composed of a simple reference to a B-entity and a P-attribute value. When a P-attribute is part of a key, comparison of two P-values should not overlap. The specified key implies that at no time two A-entities refer to the same B-entity. Also here we need the period inclusion constraint (1).

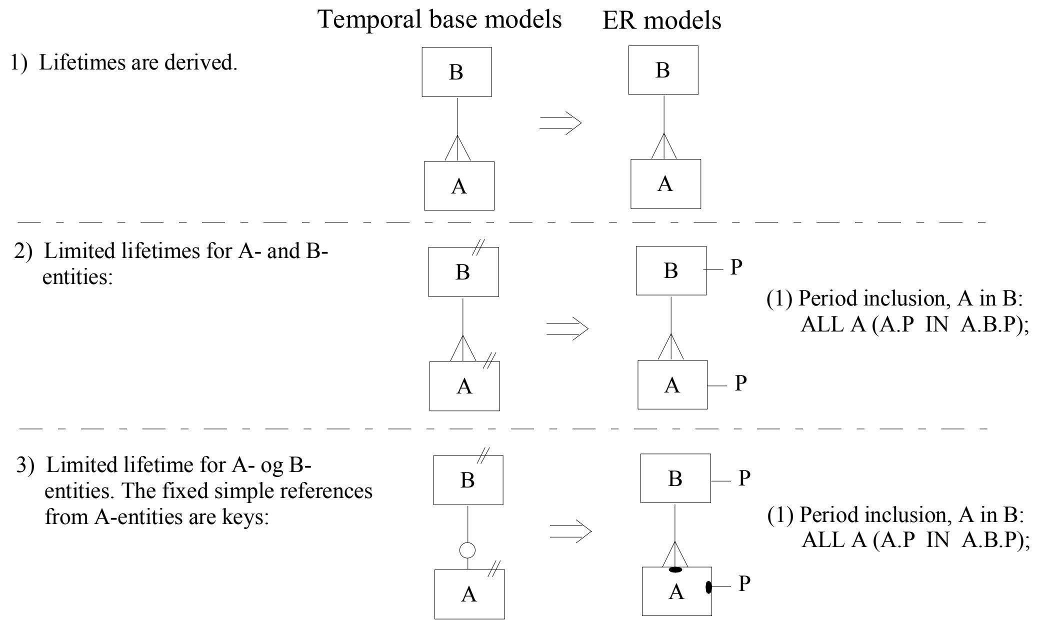

Temporal base models
ER models
1) Lifetimes are derived.
B
A
B
A
2) Limited lifetimes for A- and B-entities:
B
A
B
P
A
P
(1) Period inclusion, A in B: ALL A (A.P IN A.B.P);
3) Limited lifetime for A- og B-entities. The fixed simple references from A-entities are keys:
B
A
B
P
A
P
(1) Period inclusion, A in B: ALL A (A.P IN A.B.P);

Figure 4.5. Translation of pattern with fixed references

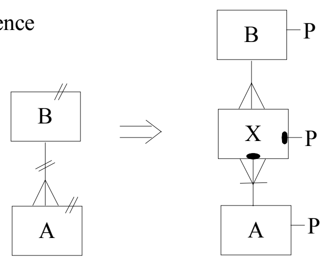
Figure 4.6 shows translation of patterns with varying simple references. A varying reference implies that an extra entity type is added to the translated schema mapping the variations.

At 4) in the figure there is a varying simple reference from A to B. In different periods an A-entity may refer to different B-entities such that, disregarding lifetimes, there is a many-to-many relationship between A- and B-entities. In the translated model we insert the X-entity type to capture these relationships. Further, we have to add a set of constraints: the bar on the relationship symbol is a cardinality constraint indicating that every A-entity should at least relate to one X-entity, and thereby to a B-entity. The period inclusion constraint (1) says that the lifetimes of X-entities should be included in the lifetimes of the B-entities referred to. The *Period continuity constraint* (2) guaranties that during the total lifetimes of A-entities X-entities exist that in adjunct periods refer to A-entities, and the key constraint of the X-entities guaranties that no two different X-entities refer to the same A-entity at the same time.

At 5) in the figure the temporal model has varying simple references from A- to B-entities being the key of the A-entities. The translation is the same as for the pattern at 4) except that there should not be two X-entities, which at the same time relate a B-entity with more than one A-entity and vice versa. This is guarded by the two alternative key constraints for the X-entities. With the common P-attribute as component of both keys, no two X-entities can in the same period refer to the same A- and B-entity.

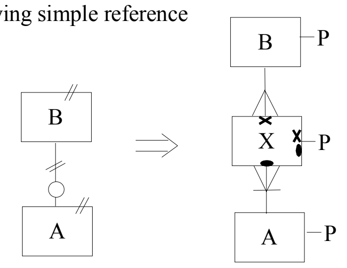
Temporal base models ER models

4) Varying simple reference from A:



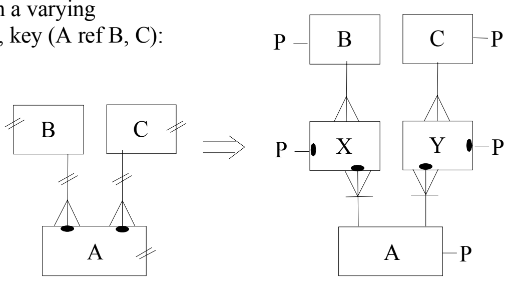
```
(1) Period inclusion, X in B:
     ALL X (X.P  IN  X.B.P);

(2) Period continuity, X vs. A:
     ALL X AS X1,
     (EXISTS X AS X2
       (X1.A = X2.A AND
       (BeginP (X1.P) = BeginP (X1.A.P)
       OR
       BeginP (X1.P) = EndP (X2.P))))
     AND
     (EXISTS X AS X3
       (X1.A = X3.A AND
       EndP (X3.P) = EndP (X3.A.P))));
```

5) A-entities with a varying simple reference as key, key(A ref B):



```
(1) Period inclusion, X in B:
     ALL X (X.P  IN  X.B.P);

(2) Period continuity, X vs. A:
     - - same predicat as above - -
```

6) A-entities with a varying composed key, key (A ref B, C):



(1) Period inclusion, X in B;

(2) Period inclusion, Y in C;

(3) Period continuity, X vs. A;

(4) Period continuity, Y vs. A;

```
(5) Period key constraint, key A (A.X.B, A.Y.C):
     ALL X AS X1, ALL X AS X2, ALL Y AS Y1, ALL Y AS Y2
       ((X1.B = X2.B AND Y1.C = Y2.C AND
        X1.A = Y1.A AND X2.A = Y2.A AND X1.A <> X2.A)
      IMPLY
        NOT (X1.P OVERLAP X2.P AND Y1.P OVERLAP Y2.P));
```

Figure 4.6. Translation of pattern with time instance and varying references

At 6) we have a temporal base model with a composed key for the A-entities where the corresponding simple references may vary in time. In the translated model, we should add the entity types, X and Y, for the respectively varying simple references with corresponding constraints like in the pattern shown at 4). To guarantee the key property we specify the *Period key constraint* (5). It says that for two different X-entities and for two different Y-entities when they all refer to the same A-entity, when the X-entities refer to the same B-entity, and when the Y-entities refer to the same C-entity, then the X-entities and the Y-entities should not both overlap in time.

The last set of translated patterns are concerned with non-instant time simple references. We have chosen to define and diagram three variant. In the translation, the otherwise involved inclusion constraints are relaxed corresponding to the variant in question. Figure 4.7 shows patterns and translations for the three variants. At 7) the lifetime of the dependent entities A can expire later than the lifetime of the B-entities referred to. At 8) the lifetime of B-entities can start before the lifetime of A. Finally at 9) the lifetimes of A-entities are independent of the lifetime of the B-entities. In the latter case no constraint is needed. In the following we shall say no more about non-instant time properties.

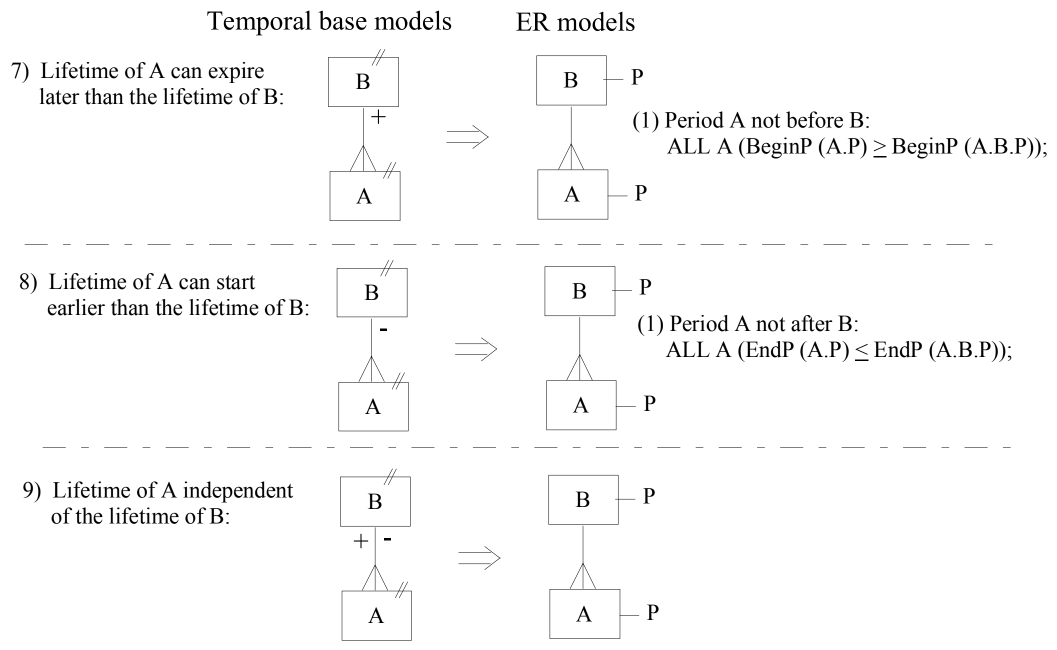
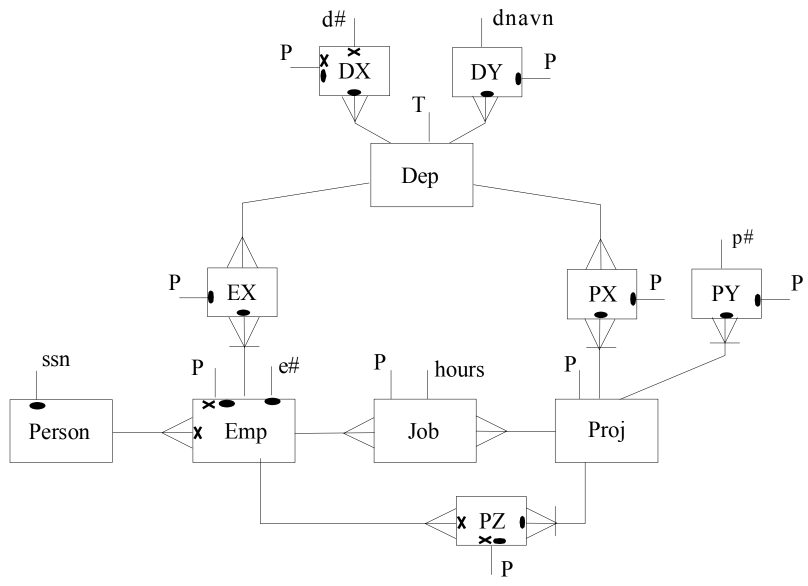

Temporal base models
ER models
7) Lifetime of A can expire later than the lifetime of B:
B
+
A
⇒
B
P
A
P
(1) Period A not before B:
ALL A (BeginP (A.P) ≥ BeginP (A.B.P));
8) Lifetime of A can start earlier than the lifetime of B:
-
(1) Period A not after B:
ALL A (EndP (A.P) ≤ EndP (A.B.P));
9) Lifetime of A independent of the lifetime of B:
+ -

Figure 4.7. Translation of non-instant time simple references

We terminate this subsection by showing the translated version of the time instant model for the company case. Figure 4.8 shows the resulting ER model. Those constraints that must be specified as predicates are stated informally. We leave it to the reader to verify that the translation is in accordance with the rules stated above.

As a result of this subsection, where we translate high-level temporal models into ER models, which with respect to temporality will be on a lower level, we may notice: First, we need an explicit attribute type to measure the continuous expansions in time. Second, we need to supplement with entity types to capture the variations of the varying simple references. And third, there will be a set of constraints that guard the temporal consistency. Constraints originally specified for the temporal model can be more complicated when specified for the ER model, e.g. key constraints in a temporal base model are much harder to specify when maintained in the translated ER model.

We will see these points reinforced when we in the next subsection translate the ER model guarding temporalities into various lexical models.



(1) Period continuity, (DX, DY) vs. Dep;
(2) Period inclusion, (EX, PX) in Dep;
(3) Period continuity, EX vs. Emp;
(4) Period continuity, (PX, PY, PZ) vs. Proj;
(5) Period key constraint, key Proj (Proj.PX.Dep,Proj.PY.p#);
(6) Period inclusion, (PZ, Job) in Emp;

Figure 4.8. Translated ER model for the temporal company case

Translations into relational models and other lexical models

We translate an ER model maintaining temporality into a relational model using the rules from subsection 3.2. As pointed out above, the resulting set of tables will normally be fully normalised. But the set of tables will be rather large with extra tables due to varying properties. Two other kinds of lexical models have been proposed that need not these extra tables. One is called a 1NF temporal model having relational tables on 1. Normal Forms. The other is called a N1NF temporal model consisting of so-called nested tables.

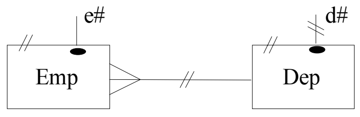
In the following we examine the translations into these three kinds of tables. We use a subset of the company case as illustrations.

The translation into normalised relational tables is illustrated in Figure 4.9 using a subpart of the company example. Initially at i) is shown the temporal base model. Employees and departments have limited lifetimes. Employees are attached to a department but not necessarily the same department all the time – the simple reference is varying. Employees are identified by a fixed e# and departments by a varying d#. At ii) the translated ER model is shown resulting in four entity types and a set of constraints, where the EX-entities map the varying simple references and the dX-entities map the varying d#-attributes. Please also note that the Dep-entity type that had a varying key-attribute after the translation has no key.

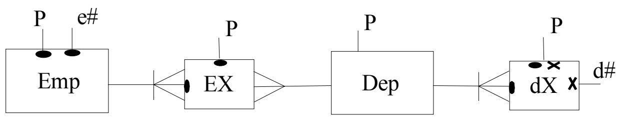
As step 0 in the translation into relational tables we add a surrogate key-attribute Did to identify Dep-entities. We also rename some of the P-attributes, using an AS-clause, because it will be needed in the following steps. The result is shown at iii).

The resulting four tables at iv) correspond to the four entity types. We have shown the CHECK-specifications corresponding to the three explicit constraints (1) to (3), where we regard the period-attributes as abstract data types having functions etc. as explained earlier.

i) Temporal base model

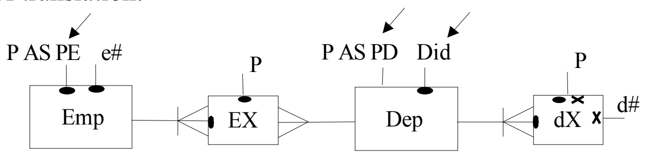


ii) ER model



(1) Period inclusion, EX in Dep;
(2) Period continuity, EX vs. Emp;
(3) Period continuity, dX vs. Dep;

iii) Step 0 of translation:



(1) Period inclusion, EX in Dep;
(2) Period continuity, EX vs. Emp;
(3) Period continuity, dX vs. Dep;

iv) Set of normalised tables:

```
Dep (PD, Did), PK (Did);

Emp (PE, e#), PK (PE, e#);

EX (P, PE, e#, Did), PK (P, PE, e#),
     FK (PE, e#) REF Emp,
     FK (Did) REF Dep;

dX (P, d#, Did), PK (P, Did),
     ALTK (P, d#),
     FK (Did) REF Dep;

(1) Period inclusion, EX in Dep
    CHECK: NOT EXISTS
    (SELECT * FROM EX, Dep
    WHERE EX.Did = Dep.Did AND
    NOT (EX.P IN Dep.PD);
```

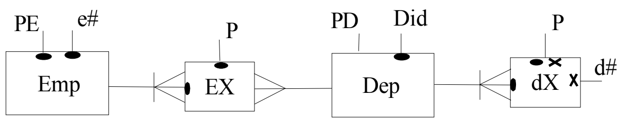
```
(2) Period continuity, EX vs. Emp
    CHECK: NOT EXISTS
    (SELECT * FROM EX X1
    WHERE NOT EXISTS
       (SELECT * FROM EX X2
       WHERE X1.e# = X2.e#  AND X1.PE = X2.PE AND
       (BeginP (X1.P) = BeginP (X1.PE)
       OR
       BeginP (X1.P) = EndP (X2.P))
    AND NOT EXISTS
       (SELECT * FROM EX X3
       WHERE X1.e# = X3.e# AND X1.PE = X3.PE AND
       EndT (X3.P) = EndT (X3.PE)));

(3) Period continuity, dX vs. Dep
    CHECK:  - - ;
```

Figure 4.9. Translations into normalised relational tables

It is interesting to notice that although the tables are normalised, the interrelationships between given periods may in some cases imply redundancy. Thus in the EX-table the periods of the PE-attribute type is redundant as the periods of the EX.P-attributes are included in the EX.PE attributes and the EX.PE attributes are inherited from Emp.PE as a foreign key component.

This can be illustrated by the sample of the normalised tables shown in Figure 4.10. (In the samples we measure a period by a begin-integers and an end-integer, and use the convention that two periods adjunct when the end of one period is one less than the beginning of the next period). In the Emp-table we have information about three employees, where the e#, e1, is reused when a former employee has left. In the EX-table we can remove the PE-attribute both from the table and the primary key. The foreign key referring to the Emp-table cannot be maintained, but we can always refer to a tuple in the Emp-table by the e# and by requiring that the period-P in the EX-table is included in the period PE of the tuples in the Emp-table referred to. The CHECK-clause (r2) taking care of the period continuity constraint is revised accordingly.

iii) ER model plus step 0:



(1) Period inclusion, EX in Dep;
(2) Period continuity, EX vs. Emp;
(3) Period continuity, dX vs. Dep;

iv) Sample of normalised tables

Table Emp

e#	PE
el	1 - 10
el	11- 12
el	14- 16

Table EX

e#	~~PE~~	P	Did
el	~~1 - 10~~	1 - 4	D1
el	~~1 - 10~~	5 - 10	D2
el	~~11- 12~~	11- 12	D2
el	~~14- 16~~	14- 16	D2

Table Dep

Did	PD
D1	1 - 8
D2	5 - 16

Table dX

Did	d#	P
D1	d1	1 - 2
D1	d2	3 - 6
D1	d1	7 - 8
D2	d3	5 - 6
D2	d2	7 - 16

Revised set of normalised tables:

```
Dep (PD, Did), PK (Did);
Emp (PE, e#), PK (PE, e#);
EX (P, PE, e#, Did), PK (P, PE, e#),
    ~~FK (PE, e#) REF Emp,~~
    FK (Did) REF Dep;
dX (P, d#, Did), PK (P, Did),
    AK (P, d#),
    FK (Did) REF Dep;
```

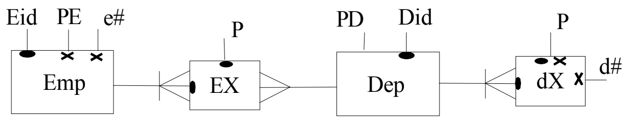
```
(2r) Period continuity, EX vs. Emp
   CHECK: NOT EXISTS
   (SELECT * FROM EX X1, Emp E
   WHERE NOT EXISTS
      (SELECT * FROM EX X2
      WHERE X1.e# = E.e#  AND  X1.P IN E.PE
      AND  X2.e# = E.e#  AND  X2.P IN E.PE AND
      (BeginP (X1.P) = BeginP (X1.PE)
      OR
      BeginP (X1.P) = EndP (X2.P)))
   AND NOT EXISTS
      (SELECT * FROM EX AS X3
      WHERE X3.e# = E.e#  AND X3.P IN E.PE AND
      EndT (X3.P) = EndT (E.PE)));
```

Figure 4.10. Revised set of normalised tables

Finally we show a version of normalised table where all the entities corresponding to temporal base entities are identified by surrogate attributes. Although surrogates are not needed here for the Emp-entity type we shall see in the following that when we have several time dimension, surrogates should identify tuples in tables corresponding to temporal base entity types.

The version of the ER model prepared for translation into relational tables by the step 0 with surrogate identifiers for both the Emp-entities and the Dep-entities is illustrated in Figure 4.11. The Emp entity type gets the surrogate attribute Eid as key and retains the former key as an alternative key.

iii) ER model plus step 0:



(1) Period inclusion, EX in Dep;
(2) Period continuity, EX vs. Emp;
(3) Period continuity, dX vs. Dep;

iv) Normalised tables with surrogate keys:

Emp (Eid, PE, e#), PK (Eid),
    ALTK (e#, PE);
EX (Eid, P, PE, Did), PK (Eid, e#),
    FK (Eid) REF Emp,
    FK (Did) REF Dep;

Dep (PD, Did), PK (Did);

dX (P, d#, Did), PK (P, Did),
    ALTK (P, d#),
    FK (Did) REF Dep;

Table Emp

Eid	e#	PE
E1	e1	1 - 10
E2	e1	11- 12
E3	e1	14- 16

Table EX

Eid	P	Did
E1	1 - 4	D1
E1	5 - 10	D2
E2	11- 12	D2
E3	14- 16	D2

Table Dep

Did	PD
D1	1 - 8
D2	5 - 16

Table dX

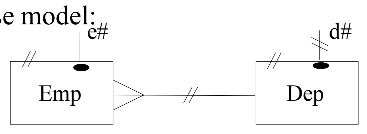
Did	d#	P
D1	d1	1 - 2
D1	d2	3 - 6
D1	d1	7 - 8
D2	d3	5 - 6
D2	d2	7 - 16

Figure 4.11. Normalised tables using surrogate identifications

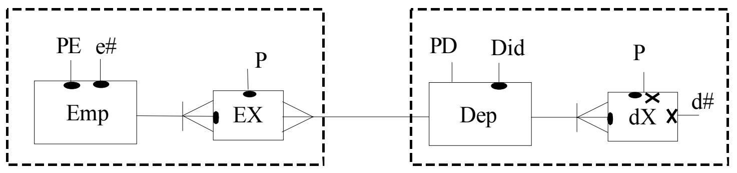
By non-normalised tables that maintain temporal qualities, there are only one table per temporal base entity type accounting for all properties of the entity type in question. The attractiveness lies in that the set of tables are smaller compared with the normalised set of tables, and that the concepts attached to a table will be easier to apply to a mini world. We shall also see that 1NF tables can be rather voluminous and have to adjust to specific rules to maintain the overall temporal consistence. By and large a N1NF or nested table for a base entity type correspond to the normalised set of tables, only the tables mapping varying properties are moved into tuples in the N1NF table as sub-tables. We shall illustrate the N1NF model only for uni-temporal models.

1NF models will have tables corresponding to each temporal base entity. The tables can be structured in several ways with various numbers of key and period attributes. We aim at a kind of canonical structure of 1NF tables that have one surrogate attribute identifying the temporal base entities and one period attribute specifying parts of the lifetime of the temporal base entities. Figure 4.12 illustrates how we reach the canonical 1NF structure using our example with the Emp and Dep temporal base entity types each having one time varying property, the varying simple reference from Emp to Dep and the varying d#-attribute of Dep respectively. At i) is shown the temporal base model and at iia) the translation into an ER model as it is prepared by step 0 for further translations. Some period attributes are renamed and the needed surrogate identifying Dep-entities is included. The translation into 1NF tables should result in an Emp-table that accommodates information from the Emp- and the EX-entity type and a Dep-table that accommodates information from the Dep- and the dX-entity type. The Dep-table is derived at having tuples corresponding to the EX-entity type that includes the attributes from the Emp-entity type. Similarly, the Dep-table is based on the dX-entity type that includes attributes from the Dep-entity type. At iiia) in the figure we have shown a sample for each of the 1NF tables not on a canonical form.

i) Temporal base model:



iia) ER model, step 0:



iiia) 1NF Tables

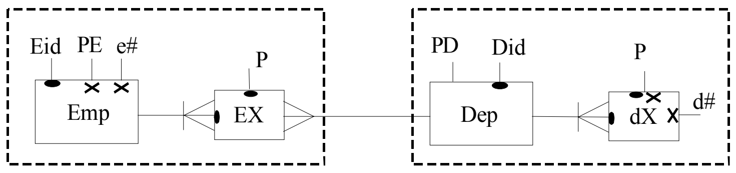
Table Emp

e#	PE	P	Did
e1	1 - 10	1 - 4	D1
e1	1 - 10	5 - 10	D2
e1	11- 12	11- 12	D2
e1	14- 16	14- 16	D2

Table Dep

Did	PD	d#	P
D1	1 - 8	d1	1 - 2
D1	1 - 8	d2	3 - 6
D1	1 - 8	d1	7 - 8
D2	5 - 16	d3	5 - 6
D2	5 - 16	d2	7 - 16

iib) ER model, step 0:



iiib) 1NF Tables with surrogates

Table Emp

Eid	e#	Did	P
E1	e1	D1	1 - 4
E1	e1	D2	5 - 10
E2	e1	D2	11- 12
E3	e1	D2	14- 16

Table Dep

Did	d#	P
D1	d1	1 - 2
D1	d2	3 - 6
D1	d1	7 - 8
D2	d3	5 - 6
D2	d2	7 - 16

Figure 4.12. 1NF temporal tables

In the 1NF Dep-table at iiia) with PD and P period attributes, the P-values should be included in PE values. We can omit the PD-attributes because at an earlier stage we have introduced the surrogate-attribute Did. The lifetimes for the departments are afterward found by coalescing the P-values for tuples with the same Did-values. It is not possible to maintain the PD-attributes and omit the Did-attributes instead, as two departments can have the same lifetime. The 1NF Emp-table at iiia) has two separate period attributes, PE and P. Although the P-values should be included in the PE-values we cannot omit the PE-attributes, as PE- together with the e#-attributes determine the set of tuples that identify the same Emp-base entities.

The canonical form is illustrated by iib) and iiib) in Figure 4.12. At iib we have added the surrogate attribute, Eid, to identify Emp temporal base entities. When we do this we can, like the Dep-entities identified by Did, omit the PE-attribute. The resulting 1NF tables, Emp and Dep are shown at iiib). In the tables each temporal base entity is identified by a surrogate attribute and represented by a set of tuples. Each table has one period attribute, and the lifetimes for the base entities are found by coalescing periods for each surrogate value.

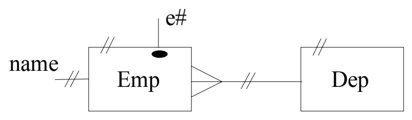
The explicit constraints that apply to the ER model, see Figure 4.11 should also apply to the 1NF model. Maintaining the period continuity constraints (2) and (3) are relatively easy. They reduce to a request that periods for the set of tuples corresponding to a temporal base entity should adjunct. The time inclusion constraint (1) is a bit harder to maintain. The surrogate attribute, Did, is used in the Emp-table to refer to departments specified in the Dep-table. The inclusion constraint requires that the period of the Emp-tuple should be included in the period obtained by coalescing the periods for Dep-tuples referred to by the Did surrogate. We have not shown the corresponding CHECK-clauses.

When temporal base entity types have several varying properties a combination of the variations for each property should be included in the 1NF-table, resulting in further tuples per temporal base entity. We shall illustrate how by an example, where the temporal base entity type Emp has a varying name-attribute and a varying simple reference to Dep-entities.

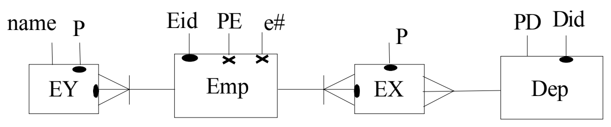
Figure 4.13 i), ii) and iii) show the temporal base model, the translated ER model and the translated normalised set of tables, respectively, where the temporal base entities are identified by surrogates. The shown samples are only concerned with one employee. The employee changes name three times and work in two different departments during his lifetime. During the time he uses the name Peter he stays in the department identified by the surrogate D1. This implies one tuple in the 1NF Emp-table. During the time he uses the name Paul he moves from department D1 to D2. This implies two tuples, one for the time in which he is in department D1, and one for the time he is in department D2. Finally while staying in department D2 he changes his name to Paulo requiring yet another tuple. The result is shown by the Emp-table at iv).

In general, when a temporal base entity type has several varying properties, the periods between the various changes may overlap. In 1NF tables these periods are *sliced* such that no overlapping occurs, and the table will have one tuple for each of the resulting slices. (The influence of slicing corresponds in some respects to tables mapping multi-valued dependencies not being on 4NF).

i) Temporal base model



ii) ER model



(1) Time inclusion, EX in Dep;
(2) Time continuity, EY vs. Emp;
(3) Time continuity, EX vs. Emp;

iii) Normalised tables

Table EY

Eid	P	name
E1	1 - 2	Peter
E1	3 - 6	Paul
E1	7 - 10	Paulo

Table Emp

Eid	e#	PE
E1	e1	1 - 10

Table EX

Eid	P	Did
E1	1 - 4	D1
E1	5 - 10	D2

Table Dep

Did	PD
D1	1 - 8
D2	5 - 16

iv) 1NF tables

Table Emp

Eid	e#	name	Did	P
E1	e1	Peter	D1	1 - 2
E1	e1	Paul	D1	3 - 4
E1	e1	Paul	D2	5 - 6
E1	e1	Paulo	D2	7 - 10

Table Dep

Did	PD
D1	1 - 8
D2	5 - 16

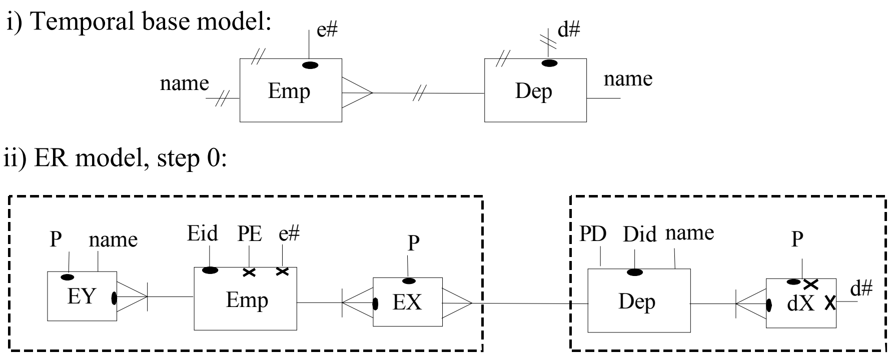
Figure 4.13. 1NF table with several varying properties

We may explain the slicing mechanism leading to a canonical 1NF structure as follows: Outset is taken in the set of normalised tables. Corresponding to a temporal base entity type there will be a *primary table* with one tuple per base entity identified by a surrogate, and a set of *property tables* with a table for each time varying property. The property tables will encounter the surrogates as foreign keys. The tuples in both the primary table and property tables are for each primary tuple – identified by the surrogate – split with respect to their period attributes such that no split period overlap. Thereby each of the tables will get the same cardinality with corresponding sets of tuples for each surrogate value. A 1NF table is then compiled by composing tuples from the split tables with the same surrogate and the same period. The attributes of the 1NF table will be composed of the surrogate, the split period, and the other attributes from each table.

N1NF temporal tables are nested tables where attributes are possibly constituted by sub-tables. Corresponding to each temporal base entity type there is a table where each tuple hold the total information for one base entity. Each varying property has a sub-table as attribute where the sub-tables have tuples for each variation of the properties in question.

Figure 4.14 shows at i) the temporal base model, where we have added a time varying name-attribute to the Emp-entity type and a non-varying name-attribute to the Dep-entity type. At ii) is shown the ER model after step 0 of the translation. The N1NF-tables are show at iii).

The tables will, as we have decided to model them here, have a period attribute type for the lifetimes of the temporal base entities and for each varying property. In principle the lifetime-attributes for the entities, PE and PD, for the Emp and the Dep table respectively may be omitted as the lifetime may be found by coalescing the periods for any of their sub-tables. We have provided each temporal base entity with a surrogate. Also here the surrogate is only necessary for entity types with time varying key components.



iii) N1NF temporal model:

Table Emp

Eid	e#	PE	EY (name, P)	EX (Did, P)
E1	e1	1 - 10	Peter 1 - 2 Paul 3 - 6 Paulo 7 - 10	D1 1 - 4 D2 5 - 10
E2	e1	11- 12	Willy 11- 12	D2 11- 12
E3	e1	14- 16	Bill 14- 16	D2 14- 16

Table Dep

Did	PD	eX (d#, P)	name
D1	1 - 8	d1 1 - 2 d2 3 - 6 d1 7 - 8	Sale
D2	5 - 16	d3 5 - 6 d2 7 - 16	Advert

Figure 4.14 N1NF temporal tables

Comparison with other valid-time models

In this section we have identified major valid time semantics by means of the introduced high-level temporal model, the temporal base model, and shown how it translate into first an ER model - the UER model - and next into three types of lexical models: a normalised relational model, a 1NF model and a N1NF model. These ideas are also sketched in(Kraft and Soerensen 2001).

Similar approaches are to our knowledge only made in parts. Several temporal conceptual models have been suggested. They are normally designed as extensions to non-temporal Chen-like ER models. The TEER model (Elmasri, El-Assal et al. 1990) distinguish static concepts and time varying concept. Conventional ER models are build with static concept elements. A temporal model is obtained by adding time varying concept elements to the conventional model being almost a copy of the conventional model. The ERT model (McBrien, Seltveit et al. 1992) and the TimerER model (Gregersen and Jensen 1998) have a non-temporal model as base and marked entity and relationship types to indicate their time varying properties. The ERT model uses symbol for binary relationships corresponding to UER conjunctions of 2nd degree. The TimerER model allows Chen-like relationship symbols modelling conjunctions of any degree. In both cases the difference between a non-temporal model and a temporal model will be a question of cardinalities of the relationship types. In an UER model we model time varying simple references by adding intermediate entity types capturing the extended cardinalities of the relationships. The concepts modelled in the schema of our high-level temporal model are based on an assumption of instant time stability. In both the ERT model and the TimerER model cardinalities correspond to an instant time view. Besides, the TimerER model also allow for cardinality specifications that correspond to a view over time. In our high-level temporal model we distinguish between instant time and non-instant time simple references. A similar distinction is made in the ERT model where relationship types can be marked by what is called a history mark to indicate non-instant time relationships. In our high-level temporal model the lifetimes of entities are unbroken, while simple references can either be varying or fixed during the lifetime of the corresponding dependant entity type. The TimerER model also adopts this distinction. The temporal mark can either be a lifetime mark, meaning an unbroken time, or a valid time mark allowing for varying relationships. Finally we notice the need for the time continuity constraint used in the translated UER model. Such a constraint is almost never mentioned for any other temporal model. A reason can be that the Chen-like relationship symbols basically relate mutually regular entities. But when the specified cardinalities imply a many-to-one instant time relationship with a mandatory one-side, the constraints are indeed needed and are probably implicitly assumed.

Our distinctions between the different kinds of lexical temporal model are mentioned in several places. In (Navathe and Ahmed 1993) a temporal normal form corresponding to sets of normalised relational tables are specified. The distinction between 1NF and N1NF temporal tables are discussed in (Clifford, Croker et al. 1993). SQL can be used both for a normalised set of relational tables and for 1NF temporal tables (Snodgrass 2000). SQL3 allows for N1NF tables.

Most proposals for temporal conceptual models - temporal ER models - advice how they translate into sets of normalised tables or 1NF temporal tables. For the translation of Chen-like based temporal models we can point to the translations of TimerER models into normalised set of relational tables (Gregersen, Mark et al. 1998). The translation is made in two steps. In step 1 a model is translated into a surrogate model build over the RM/T concept (Codd 1979). The surrogate-attributes identify existing entities, relationships, and attributes independent of how they exist in time. This is like our use of surrogate attributes identifying temporal base entities. The so-called E- and A-tables both include attributes to measure existence in periods. The constraints specified for a surrogate model account for the temporal consistence corresponding to the constraints we have discussed. Step 2 translates the surrogate model into a relational model with no surrogates relying on specified key-attributes. Thus the TimerER model doesn’t allow varying key-attributes. The importance of having surrogates to cope with varying key attributes is noted various places e.g. in (Elmasri and Navathe

2000). The importance is stressed further when we in the next section examine the bi-temporal models.

5. *Bi-temporal models (Valid and Transaction time models)*

A bi-temporal model is a valid time model with an added time dimension - transaction time - accounting for the periods in which the various recordings about the mini world are valid.

We cannot imagine a high-level bi-temporal model. Instead we will advice what we call a semi-high-level model. First we examine the semantics of transaction time, introduce the semi-high-level modelling, and translate the semi-high-level model into an ER model. Next we look on how ER models maintaining bi-temporalities translate into lexical normalised set of tables and 1NF tables. Also here we terminate the section by mentioning other approaches to bi-temporal models.

Semantics of bi-temporal model

We argue that there are dependencies between transaction periods and valid periods such that transaction periods determine valid period and nothing else.

First we look at models with no derived valid periods where entities have a limited lifetime and simple references are time varying. An existence of a new entity is recorded at a given transaction time together with its valid period. The valid period could be from the current time until ‘now’, but not necessarily so. The registration will be valid for a transaction period from current time until ‘now’. The valid period for the entity in question may be reconsidered and changed by further recordings. Each recording will terminate the previous transaction period and be valid for a next transaction period from current time until ‘now’. In this way the registration for the entity will be a set of valid periods, valid in a continued sequence of transaction periods. If the registration of an existing entity is regretted and recorded at a given transaction time the sequence of transaction periods for the entity is terminated and not prolonged further. This is in accordance with our assumptions on identities; the identity of the entity cannot be referred to in later transactions. Therefore we can in analogy with the time varying simple references regard the valid period property for an entity type as a time varying property now varying with respect to transaction periods. Similarly each valid period for a time varying simple reference can be regarded as a time varying property, varying with respect to transaction periods.

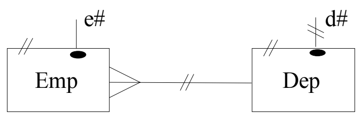
Next we consider models with derived valid periods. A simple reference with derived valid period should exist in the same valid period as the corresponding dependent entity. However, it can be recorded that a simple reference in different transaction periods refers to different entities. Thus a simple reference with derived valid period will directly be a time varying property but now with respect to transaction periods. An entity with a derived valid period will exist in a period corresponding to the intersections of the valid periods of the entities referred to by simple references. At one transaction the existence of the entity can be recorded and possibly at a later transaction the existence of the entity can be regretted. Since no other recordings are made concerning the existence of the entity, only one transaction period value will be given for entities with derived valid periods. Also this is in accordance with our assumptions on identities.

Our strategy for reaching a semi-high-level bi-temporal model and from there an ER model will be as follows: First we model the temporal base model and translate it into an ER model. In this model valid periods occur as attributes that we now name VP. With our assumption that the valid time properties are time varying properties depending on transaction periods, we can mark the simple references to valid period attributes as time varying attributes. We use a *wave-mark* instead of the double stroke to indicate that the variation is with respect to transaction periods. Simple references with derived valid periods also vary with transaction periods, and entities with derived valid period are recorded in one transaction period; both are marked with a wave-mark. We characterise the model with the wave-marks as a *semi-high-level bi-temporal model*. This model can then with the same rules used for translating uni-temporal models into ER models, be translated into an *bi-temporal ER model*.

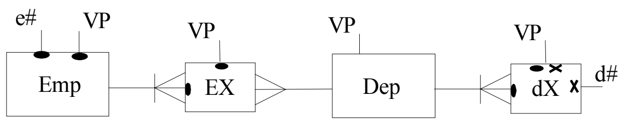
The steps are illustrated in Figure 5.1 using our former sub-model for the company case. At i) is shown the temporal base model. At ii) the temporal base model is translated into an ER model maintaining the uni-temporal properties. In the figure is shown the VP-attributes and the necessary constraints. At iii) is the semi-high-level bi-temporal model. The VP-attributes are marked with a wave-mark indicating that they vary with transaction periods. In the model the e#-attributes are the only properties that have derived valid periods. The e#-attribute is also marked with a wave mark.

We translate the semi-high-level bi-temporal model into the bi-temporal ER model shown at iv) in Figure 5.1, where the TP-attributes model transaction periods. The entity types eT, ET, EXT, DT and dXT models the varying transaction periods each entity mapping one period. We have also stated the needed constraints. The constraints of the uni-temporal ER model ii) should be guarded at any moment in transaction time. The constraints are repeated in the specification of the semi-high-level bi-temporal model iii) having an implicit meaning and stated with an explicit meaning for the bi-temporal ER model iv). Thus the constraints specified for model ii) should be modified such that they also account for transaction time. In the list of the explicit constraints for model iv), specified informally, we have prefixed the name of some of the constraints by the letter V to indicate that they guard valid time properties at any transaction time. The explicit constraints (1) to (3) from model ii) are restated as constraints (1) to (3) for model iv). The constraints corresponding to the key constraints for model ii) cannot be diagrammed. They are specified as the explicit constraints (4) to (7) in model iv). Constraint (8) guards that at any transaction time it is possible to derive a valid period for the e# attribute. Finally varying transaction periods should adjunct each other. The corresponding time continuity constraints (9) to (13) we have prefixed by the letter T.

i) Temporal base model

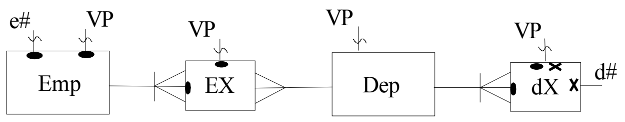


ii) Uni-temporal ER model



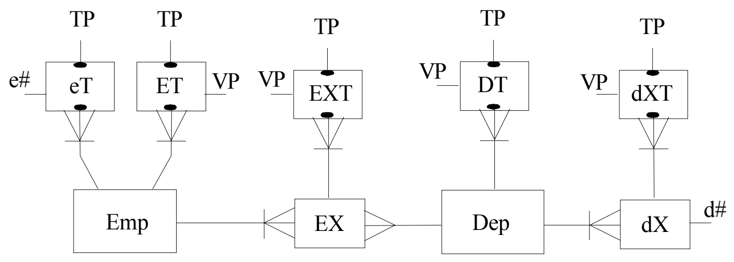
(1) Period inclusion, EX in Dep;
(2) Period continuity, EX vs. Emp;
(3) Period continuity, dX vs. Dep;

iii) Semi-high-level bi-temporal model



(1) Period inclusion, EX in Dep;
(2) Period continuity, EX vs. Emp;
(3) Period continuity, dX vs. Dep;

iv) Bi-temporal ER model

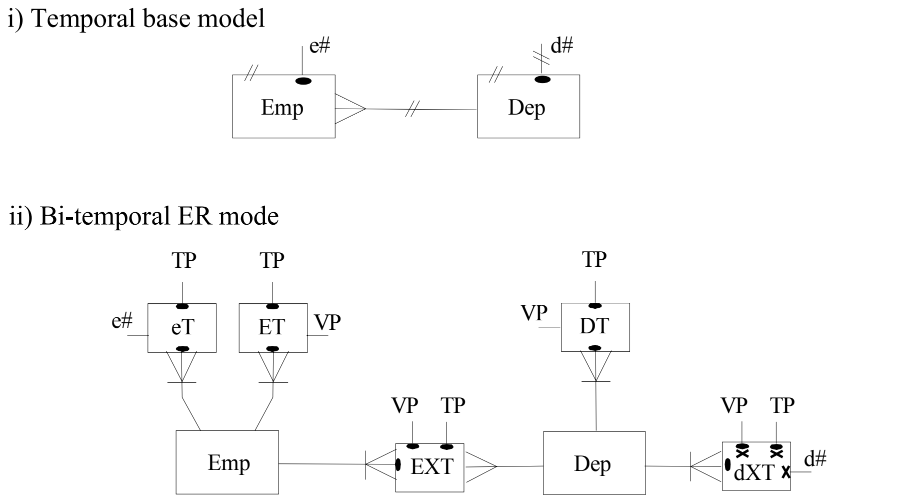


(1) VPeriod inclusion, EX in Dep;
(2) VPeriod continuity, EX vs. Emp;
(3) VPeriod continuity, dX vs. Dep;
(4) VPeriod key, Emp (Emp.eT.e#, Emp.ET.VP);
(5) VPeriod Key, EX (EX.Emp, EX.EXT.VP);
(6) VPeriod Key, dX (dX.Dep, dX.dXT.VP);
(7) VPeriod Key, dX (dX.d#, dX.dXT.VP);

(8) VPeriod derivation, eT from ET;
(9) TPeriod continuity, eT vs. Emp;
(10) TPeriod continuity, ET vs. Emp;
(11) TPeriod continuity, EXT vs. EX;
(12) TPeriod continuity, DT vs. Dmp;
(13) TPeriod continuity, dXT vs. dX;

Figure 5.1 Bi-temporal models

In the bi-temporal ER model, model iv) in Figure 5.1, the EX and the EXT entity types each represent a level of time continuity, EX a valid time level and EXT a transaction time level. We can transform the model collapsing the two entity types with only inferior loss of information. By letting the EXT-entities via the EX-entities inherit the references to the Emp-entities and the Dep-entities respectively we may omit the EX-entities. The result is shown in Figure 5.2 at ii). The benefit obtained besides the omission of the EX-entity types is that the key of the former EX entity types is now accommodated into the key of the EXT-entity and diagrammed directly. Thus we omit the former constraints (5). The former Vperiod continuity constraint (2) and the Tperiod continuity constraints (11) should be accounted for in common. Informally we call it a VTPeriod continuity constraint as shown as constraint (2a). The prefix, VT, should indicate that the constraint is a two-level constraint with an upper transaction period level where the set of transaction periods should adhere continuously and a lower level where for each transaction period the set of valid periods should adhere continuously. The semantic of the constraint will be illustrated below when we have translated the ER model into a set of normalised tables. The lost information concern the way the former set of EX entities was divided. We cannot from the information in the transformed ER model regenerate the set of EX-entities. Similar the former dX and the dXT entity types that also represent two time varying levels with the constraints (3) and (13) can collapse and replaced by constraint (3a).



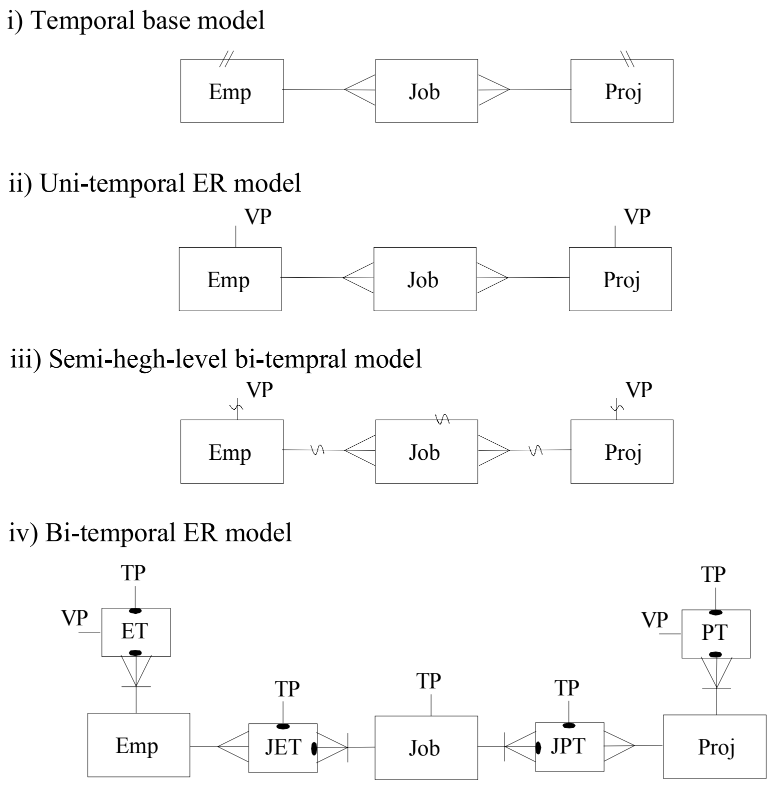
(1) VPeriod inclusion, EXT in Dep;
(2) Replaced by (2a);
(3) Replaced by (3a);
(4) VPeriod key, Emp (Emp.eT.e#, Emp.ET.VP);
(5) Omitted;
(6) Omitted;
(7) Omitted;
(8) VPeriod derivation, eT from ET;
(9) TPeriod continuity, eT vs. Emp;
(10) TPeriod continuity, ET vs. Emp;
(11) Replaced by (2a);
(12) TPeriod continuity, DT vs. Dmp;
(13) Replaced by (3a);
(2a) VTPeriod contionuity, EXT vs. EMP;
(3a) VTPeriod contionuity, dXT vs. Dep;

Figure 5.2. Transformed bi-temporal ER model

We shall use the transformed version of the bi-temporal ER model as the model for further translation into lexical models.

In Figure 5.1 and Figure 5.2 only the e#-attributes have derived valid periods or rather the relationships referring from Emp-entities to e#-values have derived valid periods. The case where both entities and their relationships have derived valid periods is illustrated in Figure 5.3. At i) is the temporal base model, at ii) the uni-temporal ER model and at iii) the semi-high-level bi-temporal model. We can explain the influence of transaction time as follows: The lifetimes for jobs are derived from the intersection of lifetimes of the employee and project in question. By transactions are only recorded whether a job exists or the existence is regretted. In accordance with our assumption on identities, a job with a given identity does only exist in one transaction period. At any given transaction time the simple references from a job to an employee or a project respectively may change. The simple references are thus time varying references dependent on transaction periods.

Model iv) show the result of the translation into the bi-temporal ER model. The constraint (1) insure that during the transaction period in which a Job-entity exists also Emp and Proj-entities exist from who's valid period the valid period for the Job can be derived. The constraints (2) to (5) are all period continuity constraints that insure that the respective varying transaction periods adjunct.



(1) VPeriod derivation, Job from ET and PT;
(2) TPeriod continuity, ET vs. Emp;
(3) TPeriod continuity, PT vs. Proj;
(4) TPeriod continuity, JET vs. Emp;
(5) TPeriod continuity, JPT vs. Proj;

Figure 5.3. Bi-temporality with derived valid periods for entities and relationships

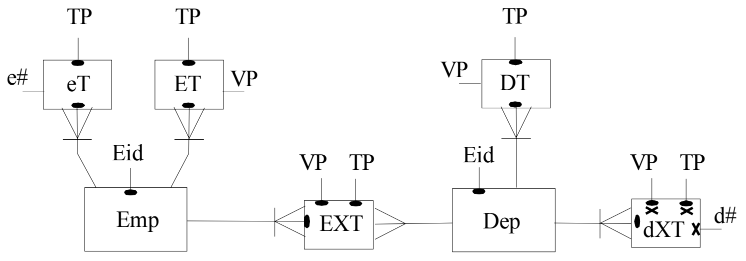
In principle all properties are stated within a valid time and a transaction time dimension in a bi-temporal model. If we omit to include transaction marks in the semi-high-level bi-temporal model for some of the properties and corresponding transaction period attributes in the bi-temporal ER model, we cannot like for the uni-temporal model derive the transaction period for the properties in question. If we however omit to account for transaction periods for some properties we cannot keep track of recorded changes of the properties; every new recording of a property will implicitly have a transaction period from genesis to eternity. The constraints involving properties with no transaction periods should be satisfied in every transaction period where other properties are recorded, which can be an elaborated task to control.

Bi-temporal lexical models

We look at the bi-temporal normalised relational model and the bi-temporal 1NF relational model.

To illustrate how the normalised relational model maintains bi-temporality we take an outset in the transformed bi-temporal ER model, Figure 5.2 ii). Step 0 in the translation adds the surrogate attributes Eid and Did to the model as shown in Figure 5.4 i). The surrogates are needed as the properties of the temporal base entities will at least be varying with transaction periods.

i) ER model after step 0:



(1) VPeriod inclusion, EXT in Dep;
(2) VTPeriod continuity, EXT vs. Emp;
(3) VTPeriod continuity, dXT vs. Dep;
(4) VPeriod key, Emp (Emp.eT.e#, Emp.ET.VP);
(5) VPeriod derivation, eT from ET;
(6) TPeriod continuity, eT vs. Emp;
(7) TPeriod continuity, ET vs. Emp;
(8) TPeriod continuity, DT vs. Dmp;

ii) Normalised tables::

Emp (Eid) PK (Eid);

ET (VP, Eid, TP) PK (Eid, TP),
FK (Eid) REF Emp;

eT (e#, Eid, TP) PK (Eid, TP),
FK (Eid) REF Emp;

EXT (Eid, VP, TP, Did) PK (VP, Eid, TP),
FK (Eid) REF Emp
FK (Did) REF Dep;

Dep (Did) PK (Did);

DT (VP, Did, TP) PK (Did, TP),
FK (Did) REF Dep;

dXT (d#, VP, Did, TP) PK (VP, Did, TP),
ALTK (d#, VP, TP),
FK (Did) REF Dep;

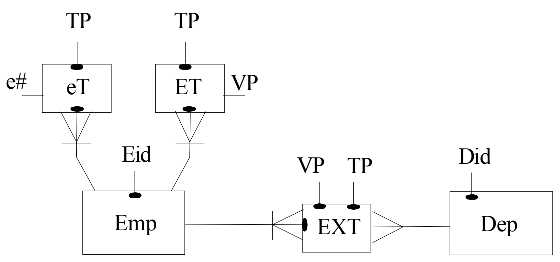
Figure 5.4. Bi-temporal normalised tables

Figure 5.4 ii) shows the translated normalised set of tables. There are three kinds of tables: The *primary-tables*, Emp and Dep in the figure, corresponding to the base entity types in the temporal base model having a surrogate as their only attribute. The *lifetime-tables*, ET and DT, that maintain the lifetimes for the base entities, and the *property-tables*, eT, EXT and dXT that maps the properties of the base entity types.

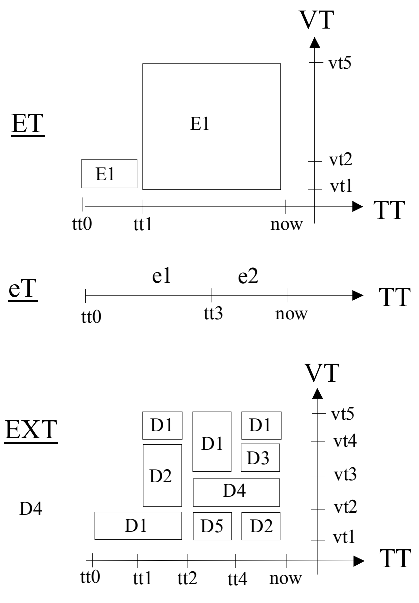
We haven't shown the explicit constraints that should guard the tables. They will by and large be as complicated as those for the ER model. However we show a sample of tables and explain how the constraints are satisfied but only for the Emp-entity type and its properties.

The sample in Figure 5.5 is concerned with one employee identified by the surrogate E1 and five departments identified by the surrogates D1 to D5. The primary tables, Emp and Dep respectively, map this. The drawings to the left in the figure show how the different time varying properties occur within valid and transaction times. The ET drawing corresponding to the lifetime table, ET, shows that in the transaction period from tt0 to tt1 is recorded that the employee E1 is employed in valid time from vt1 to vt2 and in transaction period from tt1 to 'now' E1 is employed in the period from vt1 to vt5. Drawing eT for the property table eT shows that in the transaction period from tt0 to tt3 the employee is identified by e# equal to e1 and in transaction period tt3 to 'now' by e# equal to e2. Finally drawing EXT for the property table EXT shows how E1 is related with a department in different valid periods as recorded in different transaction periods.

ER model:



(1) VPeriod inclusion, EXT in Dep;
(2) VTPeriod continuity, EXT vs. Emp;
(3) VPeriod key, Emp (Emp.eT.e#, Emp.ET.VP);
(4) VPeriod derivation, eT from ET;
(5) TPeriod continuity, eT vs. Emp;
(6) TPeriod continuity, ET vs. Emp;



Normalised tables:

Table Emp

Eid
E1

Table ET

Eid	VP	TP
E1	vt1-vt2	tt0-tt1
E1	vt1-vt5	tt1-now

Table eT

Eid	e#	TP
E1	e1	tt0-tt3
E1	e2	tt3-now

Table EXT

Eid	Did	VP	TP
E1	D1	vt1-vt2	tt0-tt2
E1	D2	vt2-vt4	tt1-tt2
E1	D1	vt4-vt5	tt1-tt2
E1	D5	vt1-vt2	tt2-tt4
E1	D4	vt2-vt3	tt2-now
E1	D1	vt3-vt5	tt2-tt3
E1	D2	vt1-vt2	tt4-now
E1	D3	vt3-vt4	tt4-now
E1	D1	vt4-vt5	tt4-now

Table Dep

Did
D1
D2
D3
D4
D5

Figure 5.5. Samples for bi-temporal normalised tables

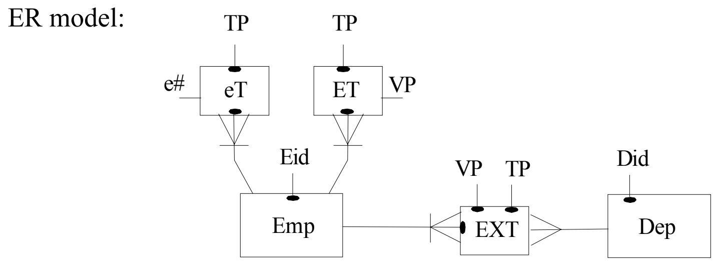
The drawings also illustrate how the constraints are satisfied. The area covered in the ET drawing shows the lifetimes of the employee E1. In all this times there should be a simple reference to a department such that the area covered by drawing EXT should be the same as the area covered at ET. This satisfies one part of the valid period inclusion constraint (1) in Figure 5.5. Each box in the ET drawing represents an occurrence of an ET-entity and each box in the EXT drawing represents an occurrence of an EXT-entity. As the boxes don't overlap both the valid time and the transaction time continuity constraints are satisfied corresponding to the constraints (2) and (6). The eT drawing stretch the same transaction times as the ET drawing satisfying the valid time derivation constraint (4) and the period continuity constraint (5). The rest of the valid period inclusion constraint (1) is satisfied because of the rudimentary modelling of departments with no valid and no transaction periods. The valid period key constraint is not illustrated as the sample only concerns one employee.

At this point we can return to the influence of the transformation of the ER model and look at the sample in drawing EXT in Figure 5.5. The EXT entity type models time varying simple references from employee to departments. In the transaction period tt1 to tt2 is recorded that both in the valid period vt1 to vt2 and in the valid period vt4 to vt5 the employee E1 is related to department D1. Looking back into model iv) in Figure 5.1 we need two instances of the EX-entity type in the mapping. If we look at the relationship between E1 and D1 in the valid period vt3 to vt5 and the transaction period from tt1 until now in Figure 5.5 the recorded valid period changes with different transaction periods. In Figure 5.1 this is mapped by one EX-entity and three EXT entities. After the transformation of the ER model the three EXT entities remain mapping the time extensions in valid and transaction time as shown in the EXT drawing in Figure 5.5, marked as D1-boxes. We may recognise the influence of our assumption that transaction time determine valid time in the way the periods are divided over the EXT entities: The division into three EXT entities are produces corresponding to stable valid periods in each transaction period. If we did not keep to this dependency we could just as well make a division corresponding to stable transaction period in each valid period resulting in two EXT entities, one with transaction period tt1 until now and valid period from vt4 to vt5 and one with transaction period from tt2 to tt4 and valid period from vt3 to vt4.

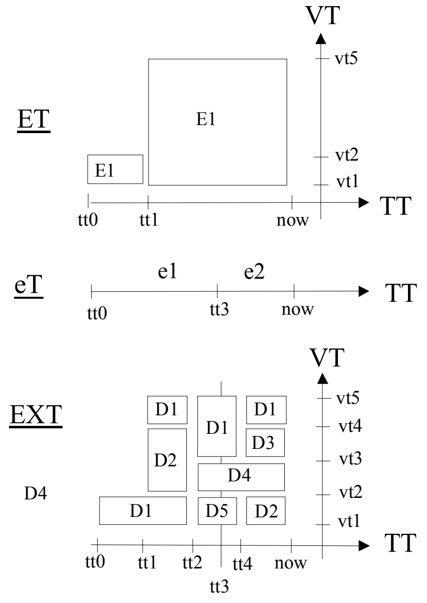
In the bi-temporal 1NF model there is one table corresponding to each temporal base entity type capturing all properties for the entity type as they vary with valid and transaction periods. Like for the uni-temporal model we derive at the 1NF table by slicing now accounting for both time dimensions. But since transaction periods determine valid periods we need only slice with respect to varying transaction periods as it automatically covers slicing with respect to valid periods.

In Figure 5.6 we have the same situation as in the previous figure. Looking at the drawings to the left in the figure, corresponding to samples for the normalised tables, we have different overlapping transaction periods. The transaction periods for the ET-table overlap the transaction periods for eT that again overlap the transaction period for EXT. Now we can *slice* the periods for the normalised tables such that no overlapping transaction period occurs. The 1NF table will accordingly have tuples combining tuples from the corresponding sliced normalised tables.

The way we have specified the slicing mechanism for uni-temporal models (se the previous section) is also used here. It implies a continuous set of transaction periods derived from the set of recordings of the properties of the temporal base entity in question and tuples that measure the properties accounting for how they occur in each of these transaction periods. The sample in Figure 5.6 shows five sliced transaction periods. A recording about one property may cause a duplication of several tuples. For instance imply the change of the e# from e1 to e2 at transaction time tt3 duplication of three tuples.



(1) VPeriod inclusion, EXT in Dep;
(2) VTPeriod continuity, EXT vs. Emp;
(3) VPeriod key, Emp (Emp.eT.e#, Emp.ET.VP);
(4) VPeriod derivation, eT from ET;
(5) TPeriod continuity, eT vs. Emp;
(6) TPeriod continuity, ET vs. Emp;



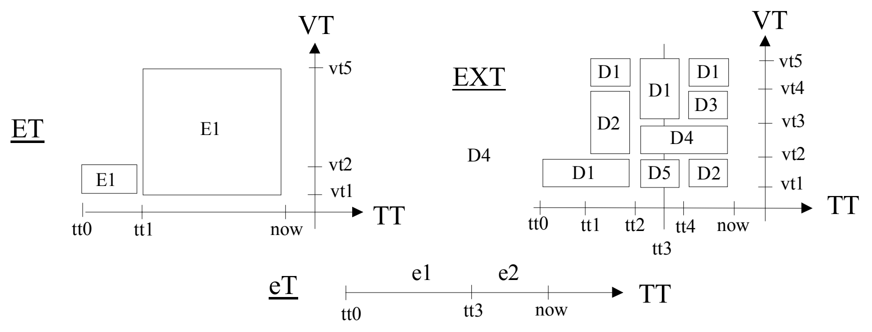
1NF: Table Emp, sliced

Eid	e#	Did	VP	TP
E1	e1	D1	vt1-vt2	tt0-tt1
E1	e1	D1	vt1-vt2	tt1-tt2
E1	e1	D2	vt2-vt4	tt1-tt2
E1	e1	D1	vt4-vt5	tt1-tt2
E1	e1	D5	vt1-vt2	tt2-tt3
E1	e1	D4	vt2-vt3	tt2-tt3
E1	e1	D1	vt3-vt5	tt2-tt3
E1	e2	D5	vt1-vt2	tt3-tt4
E1	e2	D4	vt2-vt3	tt3-tt4
E1	e2	D1	vt4-vt5	tt3-tt4
E1	e2	D2	vt1-vt2	tt4-now
E1	e2	D4	vt2-vt3	tt4-now
E1	e2	D3	vt3-vt4	tt4-now
E1	e2	D1	vt4-vt5	tt4-now

Figure 5.6. Bi-temporal 1NF table obtained by slicing

If we look at adjunct transaction slices there are tuples where the attribute values save the transaction period value are the same. Such tuples can be *coalesced* into one tuple with a transaction period being the union of the adjunct periods. Figure 5.7 shows how the sliced 1NF table is coalesced into the resulting 1NF table.

We can characterise the resulting bi-temporal 1NF model. The model has one table for each temporal base entity type. Each temporal base entity is identified by a surrogate and represented by a set of tuples in the table. In the table in Figure 5.7 the Eid-attribute type is the primary surrogate. A table will only have one valid period attribute type and one transaction period attribute type. The other attributes describe the property of the entities as they occur in given valid and transaction periods - e# and Did are property attributes. When temporal base entity types have no varying properties there will only be one tuple per transaction period slice. The valid time period attribute maps its lifetime. When an entity type has varying properties the lifetime of the entities are found by concealing the corresponding valid time slices. For a given primary surrogate value there will be no two tuples with overlapping or adjunct transaction periods that determines the same set of the other attribute values. This is due both to the slicing and coalescing of transaction periods. Surrogates are also used to refer to other tables - Did is a surrogate. Temporal constraints specified on the ER model level should also apply to the 1NF model. We can divide the constraints in those concerned locally with one table and those concerned with relationships among tables. The local constraints are such as instant period key constraints and period continuity constraints and the non-local constraints involve foreign surrogates and are such as period inclusion constraints.



1NF: Table Emp, sliced

Eid	e#	Did	VP	TP
E1	e1	D1	vt1-vt2	tt0-tt1
E1	e1	D1	vt1-vt2	tt1-tt2
E1	e1	D2	vt2-vt4	tt1-tt2
E1	e1	D1	vt4-vt5	tt1-tt2
E1	e1	D5	vt1-vt2	tt2-tt3
E1	e1	D4	vt2-vt3	tt2-tt3
E1	e1	D1	vt3-vt5	tt2-tt3
E1	e2	D5	vt1-vt2	tt3-tt4
E1	e2	D4	vt2-vt3	tt3-tt4
E1	e2	D1	vt4-vt5	tt3-tt4
E1	e2	D2	vt1-vt2	tt4-now
E1	e2	D4	vt2-vt3	tt4-now
E1	e2	D3	vt3-vt4	tt4-now
E1	e2	D1	vt4-vt5	tt4-now

Resulting 1NF table, sliced and coalecsed

Eid	e#	Did	VP	TP
E1	e1	D1	vt1-vt2	tt0-tt2
E1	e1	D2	vt2-vt4	tt1-tt2
E1	e1	D1	vt4-vt5	tt1-tt2
E1	e1	D5	vt1-vt2	tt2-tt3
E1	e1	D4	vt2-vt3	tt2-tt3
E1	e1	D1	vt3-vt5	tt2-tt3
E1	e2	D5	vt1-vt2	tt3-tt4
E1	e2	D4	vt2-vt3	tt3-now
E1	e2	D1	vt4-vt5	tt3-tt4
E1	e2	D2	vt1-vt2	tt4-now
E1	e2	D3	vt3-vt4	tt4-now
E1	e2	D1	vt4-vt5	tt4-now

Figure 5.7 1NF table, sliced and coalesced

Comparison with other bi-temporal models

Bi-temporal models are considered by several authors and normally the valid time and the transaction time dimensions are regarded as being on equal term and in most cases also as independent of each other. This is the case for the TimerER model (Gregersen and Jensen 1998; Gregersen, Mark et al. 1998). On the conceptual level, the diagrammed TimerER model, the entity types and the relationship types can be marked to account for either valid time, for transaction time or for both time dimensions. After translation into corresponding tables the tables include respectively attributes to map valid periods and/or transaction periods. TSQL (Zaniolo, Ceri et al. 1997) is basically a bi-temporal relational 1NF model. Tables are declared to account either for valid periods, for transaction periods or for both and will include corresponding but anonymous sets of attributes. The defined language for TSQL comprises a set of constructs that utilises the mapped temporalities.

6. Tri-temporal models (Valid, Decision, and Transaction times)

The recognition that transaction times determine valid times, the consequence of which is used to extend a uni-temporal model into a bi-temporal model, can be repeated introducing further temporal dimensions. When we look upon decisions concerning properties in valid time they are suggested at a given point in time and remain in effect until they are changed at a later point in time. Regarded in this way, decision time play the same role against valid time properties, as did the transaction time in the previous section also against valid time properties. Thus it is possible to replace a transaction time dimension in a bi-temporal model with a decision time dimension and afterwards add the transaction time dimension such that transaction times now determine decision times that again determine valid time properties.

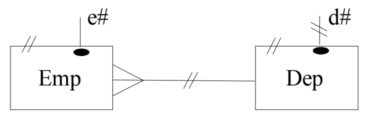
Based on this argument it is straightforward to examine the consequences of having tri-temporal models and in general multi-dimensional temporal models with more than two time-dimensions. We discuss the semantics of tri-temporal models by showing how they are derived at, based on a uni-temporal model mapping of valid time properties, the temporal base model, and how they translate into ER models. Further we show how they translate into normalised sets of tables and into sets of 1NF tables. There are to our knowledge no systematic approaches to temporal models with more than two time dimensions, so we are not able to refer to other works.

Semantics of tri-temporal model

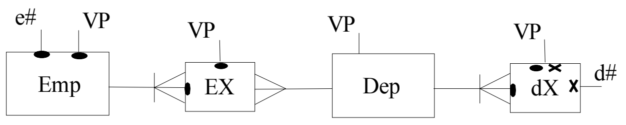
A tri-temporal model can be thought off as a model that maps valid time properties as decided at points in decision time and recorded at points in transaction time. In Figure 6.1 at i) we show the temporal base model for the part of the company case used above. The model maps what is regarded as valid periods for the properties in the mini world as they are determined at a given point in decision time. When we translate this model into an ER model some valid periods will be modelled explicitly by valid-period attributes and some valid periods will be derived from other valid periods as discussed above. The result with the required constraints is shown at ii).

When we have a model that accounts for decisions every property in valid time will be decided upon. A decision concerning a given property may remain unchanged for a given period - a decision period - after which it may be changed. This is similar to the situation discussed above for transaction times where each property is recorded and stays unchanged in a period until it is possibly regretted and changed.

i) High level uni-temporal model (Valid time)

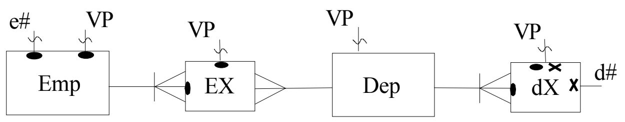


ii) ER model mapping uni-temporality (Valid time)



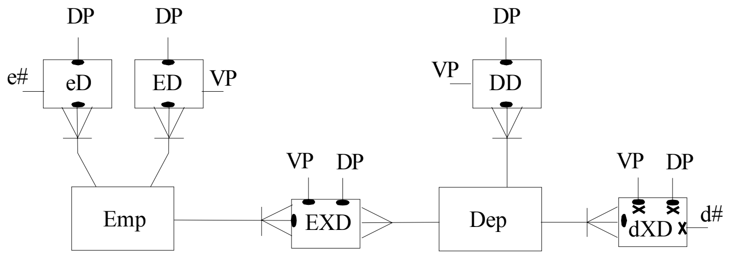
(1) Period inclusion, EX in Dep;
(2) Period continuity, EX vs. Emp;
(3) Period continuity, dX vs. Dep;

iii) Semi-high-level bi-temporal model (Valid and Decision times):



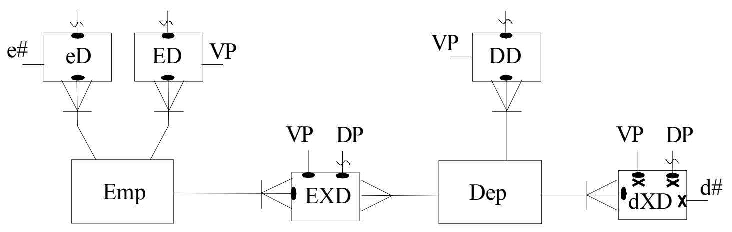
(1) Period inclusion, EX in Dep;
(2) Period continuity, EX vs. Emp;
(3) Period continuity, dX vs. Dep;

iv) ER model mapping bi-temporalities (Valid and Decision times):



(1) VPeriod inclusion, EXD in Dep;
(2) VDPeriod continuity, EXD vs. Emp;
(3) VDPeriod continuity, dXD vs. Dep;
(4) VPeriod key, Emp (Emp.eD.e#, Emp.ED.VP);
(5) VPeriod derivation, eD from ED;
(6) DPeriod continuity, eD vs. Emp;
(7) DPeriod continuity, ED vs. Emp;
(8) DPeriod continuity, DD vs. Dep;

v) Semi-high-level tri-temporal model (Valid, Decision and Transaction times):



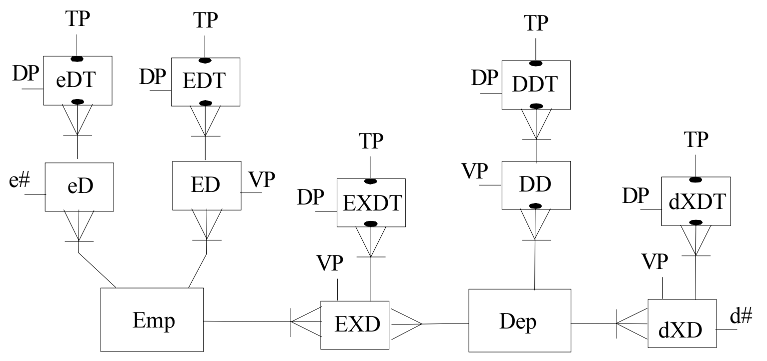
(1) VPeriod inclusion, EXD in Dep;
(2) VDPeriod continuity, EXD vs. Emp;
(3) VDPeriod continuity, dXD vs. Dep;
(4) VPeriod key, Emp (Emp.eD.e#, Emp.ED.VP);
(5) VPeriod derivation, eD from ED;
(6) DPeriod continuity, eD vs. Emp;
(7) DPeriod continuity, ED vs. Emp;
(8) DPeriod continuity, DD vs. Dep;

Figure 6.1. Semi-high-level tri-temporal model

At iii) in Figure 6.1 we show the semi-high-level bi-temporal model where the wave-makes now indicate varying temporalities with respect to decision time. We translate the semi-high-level bi-temporal model into an ER model as shown at iv). The resulting model with its constraints will be similar to the ER model discussed earlier with respect to transaction time. In particular we should notice that the time continuity constraints required for the transaction periods are also required for decision periods. A decision is changed by a following decision and the last decision stays in effect until now. In model iv) the attributes measuring decision periods are named DP.

Finally we add the transaction time to the model. By transactions as well the periods for given decisions concerning given properties as the properties themselves are victims for recording. Now valid time properties are determined by the decision periods in which they are decided, and when having a transaction time dimension the decision periods and thereby the valid time properties are determined by the transaction periods in which the recording holds. Or in other words we may regard decision periods as time varying properties depending on transaction periods. Thus we can model a semi-high-level tri-temporal model based on the ER model for the bi-temporal case where we use the wave mark to indicate that the decision period attributes are varying depending on transaction periods. This is shown by model v) in Figure 6.1.

vi) ER model (Valid, Decision and Transaction times):



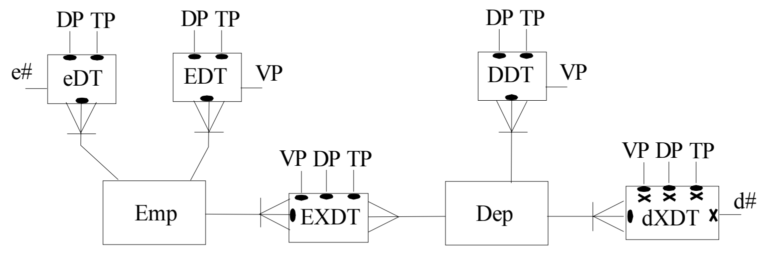
(1) VPeriod inclusion, EXD in Dep;
(2) VDPeriod continuity, EXD vs. Emp;
(3) VDPeriod continuity, dXD vs. Dep;
(4) VPeriod key, Emp (Emp.eD.e#, Emp.ED.VP);
(5) VPeriod derivation, eD from ED;
(6) DPeriod continuity, eD vs. Emp;
(7) DPeriod continuity, ED vs. Emp;
(8) DPeriod continuity, DD vs. Dep;
(9) TPeriod continuity, eDT vs. eD;
(10) TPeriod continuity, EDT vs. ED;
(11) TPeriod continuity, EXDT vs. EXD;
(12) TPeriod continuity, DDT vs. DD;
(13) TPeriod continuity, dXDT vs. dXD;
(14) DPeriod key, eD (eD.Emp, eD.eDT.DP);
(15) DPeriod key, ED (ED.Emp, ED.EDT.DP);
(16) DPeriod Key,
EXD (EXD.Emp, EXD.VP, EXD.EXDT.DP);
(17) DPeriod key,
dXD (dXD.Dep, dXD.VP, dXD.dXDT.DP);
(18) DPeriod key,
dXD (dXD.d#, dXD.VP, dXD.dXDT.DP);

Figure 6.2. ER model maintaining tri-temporality

Figure 6.2 shows the translation of the semi-high-level tri-temporal model into an ER model. In the model the valid period constraints (1) to (5) is adjusted to the translated entity types. Similar is the case for the decision period constraints (6) to (8). Further we have to use explicit constraints to specify the keys for the entity types modelling the decision period variations, the constraints (14) to (18). Finally we should include the transaction period continuity constraints (9) to (13).

As we did with the bi-temporal model we can transform the ER model so that the different levels of varying temporal dimensions are captured by one entity type. After the transformation we don't need the explicit decision period key constraints as they are diagrammed, and have to modify some of the period continuity constraints.

vi) ER model transformed (Valid, Decision and Transaction times):



(1) VPeriod inclusion, EXDT in Dep;
(2) VDTPeriod continuity, EXDT vs. Emp;
(3) VDTPeriod continuity, dXDT vs. Dep;
(4) VPeriod key, Emp (Emp.eD.e#, Emp.EDT.VP);
(5) VPeriod derivation, eDT from EDT;
(6) DTPeriod continuity, eDT vs. Emp;
(7) DTPeriod continuity, EDT vs. Emp;
(8) DTPeriod continuity, DDT vs. Dep;

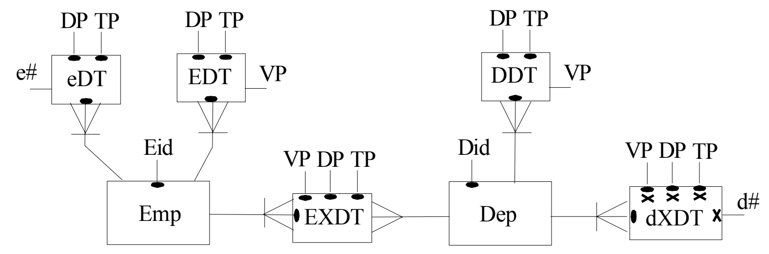
Figure 6.3. Transformed ER model maintaining tri-temporality

Figure 6.3 vi) shows the result. The diagram looks systematic and so does the constraints. When compared with a corresponding ER model maintaining bi-temporality, e.g. model iv) in Figure 6.1, the diagrammed entity type and the constraints are almost the same only the property entity types and the constraints should account for more continuous varying time dimensions.

Lexical tri-temporal models

In a tri-temporal ER model entity types corresponding the temporal base entity types will be absolute regular. As step 0 in the translations into lexical models surrogate attributes identifying the entities should be added to the ER model. The ER model with the surrogate attributes Eid and Did is shown at vi) in Figure 6.4. From here it is trivial to translate the tri-temporal ER model into a set of normalised tables using the rules from section 2. The result is shown at vii) in Figure 6.4.

vi) ER model transformed, step 0 (Valid, Decision and Transaction times):



(1) VPeriod inclusion, EXDT in Dep;
(2) VDTPeriod continuity, EXDT vs. Emp;
(3) VDTPeriod continuity, dXDT vs. Dep;
(4) VPeriod key, Emp (Emp.eD.e#, Emp.EDT.VP);
(5) VPeriod derivation, eDT from EDT;
(6) DTPeriod continuity, eDT vs. Emp;
(7) DTPeriod continuity, EDT vs. Emp;
(8) DTPeriod continuity, DDT vs. Dep;

vii) Normalised tables (Valid, Decision and Transaction times):

Emp (Eid) PK (Eid);

Dep (Did) PK (Eid);

EDT (Eid, VP, DP, TP) PK (Eid, DP, TP),
FK (Eid) REF Emp;

eDT (Eid, e#, DP, TP) PK (Eid, DP, TP),
FK (Eid) REF Emp;

EXDT (Eid, Did, VP, DP, TP) PK (Eid, VP, DP, TP),
FK (Eid) REF Emp,
FK (Did) REF Dep;

DDT (Did, VP, DP, TP) PK (Did, DP, TP),
FK (Did) REF Dep;

dXDT (Did, d#, VP, DP, TP) PK (Did, VP, DP, TP),
ALTK (d#, VP, DP, TP),
FK (Did) REF Dep;

Figure 6.4. Normalised tables maintaining tri-temporality

The 1NF set of tables for the tri-temporal case is also derived at in analogy with what we did for the bi-temporal case here by accounting for all the three time dimensions. Slicing should be done with respect to all of the three time dimensions. Coalescing should be done first with respect to the decision time dimension and next with respect to the transaction time dimension.

The resulting 1NF tables for the case above are shown in Figure 6.5.

vii) 1NF tables (Valid, Decision and Transaction times):

Emp (Eid, e#, Did, VP, DP, TP) PK (Eid, VP, DP, TP);

Dep (Did, d#, VP, DP, TP) PK (Did, VP, DP, TP);

Figure 6.5. 1NF tables maintaining tri-temporality

7. Conclusion

The results obtained here do not differ much from various results obtained by other authors. What we especially hope to have obtained however is an enlightens of temporal semantics and how the semantics correspond when modelled and maintained in different models, conceptual models as well as different forms for lexical models. Other objections that we would like to stress concern the methods and the requirements we have used to obtain these results.

We have examined the semantics of temporal models using the idea of modelling temporal properties on a high level using modelling tools that maintain the properties uniquely and consistently. The advantage is that a mapping into such a model always will be consistent and complete with respect to the properties in question. We have examined how lower level models can maintain the temporal properties by giving rules for translating into the lower level models. The high-level temporal model is translated first into an ER model and next into various lexical models. In this way we have examined how sets of normalised tables, sets of 1NF tables and sets of N1NF tables should be constructed and constrained such that the temporal properties are maintained consistently and precisely.

It is only a limited set of properties for which we can suggest a high-level modelling tool. The ER model used in this paper is able to handle the existence of entities and their mutual relationships on a high level. The modelling tool for the ER model is a graph that captures the existences uniquely and consistent. For the uni-temporal model mapping the existences in valid time a high-level model is also advised. The modelling tool maps the existence in time in a three-dimensional graph with one continuous time dimension. When we introduce further time dimensions we could possibly use modelling tools with a corresponding set of continuous dimensions, but such tools will probably be unmanageable. Instead we have to use other techniques and here we have taken advantage of some dependency in between the time dimensions.

Another benefit of high-level models is that the elements mapped in a corresponding schema will correspond to concepts directly applicable in a mini world. However the way a mini world is conceptualised is not given a priori, which again will have consequences on the semantics that should be applied on the symbols used in a schema. Out of various possibilities we choose the most primitive and general semantics for the symbols that again normally lead to models where the concepts describing a mini world are specified in one way only. In the ER model the choice of relationship symbol is a symbol for a binary one-to-may relationship having a mandatory one-side. In considering the identity of entities over time the choice is that entities exist in unbroken spans of time. Finally we have chosen to conceptualise the mini world regarding valid time such that the concepts mapped in a uni-temporal schema should be applicable at any moment in time.

A high level model can be constrained with respect to the properties mapped on the high level. The high-level ER model is constrained with respect to existence of entities and existence of paths though relationships: cardinality, key and other constraints involving path expression. The high-level uni-temporal model may have the same constraints but now with the implicit assumption that the constraints account for the existences in time. Beside, the high-level uni-temporal model can be constrained with respect to the lifetime of the existences. Thus we have discussed constraints that allow derived lifetime where the lifetime of entities or simple references are derived from the lifetime of related entities. Simple references are constrained such that the entities referred to may exist earlier or later than the period in which the entities referred from exist.

A lower level model with respect to given properties will normally require special semantically elements and sets of constraints involving the semantics of these elements to maintain a consistent and complete handling of the properties. When the ER model is used to maintain temporal properties it will be on a lower level and the ER model is provided with special attributes that measure expansions in time. We have suggested a period attribute introduced as an abstract data type equipped with functions and constants that manipulate and compare period values. It is straightforward from a high level model with its direct and consistent description of the given properties to advise rules for a translation into a lower level model, this although the rules may turn out to be rather complicated. We have showed rules that translate a high-level uni-temporal model into an ER model and further into three types of lexical models. We should mention that the exhibited rules are not exhaustive and that especially involved constraints may cause rather complicated translation rules.

For the semantics of bi-temporal models and models with further time dimensions we cannot suggest a high level modelling tool and thus have to use lower level models. However in this particular case we are able to utilize some interdependencies between the time dimensions that will guide us to a semantically safe model on the lower level. In the bi-temporal case with a valid time and a transaction time dimension we regard any entity and relationship to exist in valid time and that these existences are the only properties that are recorded. A recording is valid in a given transaction period until it is possibly redone. Looking at the recordings for a given existence in the mini world it can be brought into existence for a given valid period, the valid period can be changed several times, and the existence can be regretted such that there is no valid period. In any case there is a sequence of recordings where each recording is valid for a given transaction period. By this observation we recognise that the valid period is a time varying property that vary by a sequence of transaction periods. We utilise this dependency as follows: First we model the corresponding high-level uni-temporal model, called the temporal base model. Next we translate this model into an ER model with corresponding valid period attributes. These attributes we regard as time varying attributes varying within sequences of transaction periods. Hereby we get what is called a semi-high-level bi-temporal model. Finally we translate the semi-high-level model into an ER model introducing transaction period attributes, and from there further into a lexical model.

Also at this point we have to decide on the identity of existing entities and suggest that an entity that at a given transaction time is regretted to exist cease to exist. If it is regretted that we earlier regretted something a new entity should be brought into existence.

By examining the bi-temporal model we now have a mechanism with which we can introduce models with more dimensions. An example is a tri-temporal model that has the ordered time dimensions: valid time, decision time and transaction time. The model captures information on valid period properties in the mini world that are decided and re-decided in sequences of decision periods and recorded and re-recorded in sequences of transaction periods.

Our examination of temporal models has stopped at this point leaving a set of open questions. We have assumed that properties mapped into a temporal model with multiple ordered time dimensions are present in all dimensions. In the uni-temporal model we examine properties with derived valid periods, and showed that in the translated ER model and the lexical models there is no need for explicit valid period attributes to cover their valid time. In the bi- and tri-temporal models we have not pursued in detail what semantics should be applied when transaction period and decision period attributes are omitted from properties in the corresponding ER model and lexical models.

We have not discussed query languages neither for the high level models nor when translated into ER models and the various lexical models. We are aware that quite a number of problems exist and that several language constructs have been suggested especially for the lower level lexical models (e.g. for TSQL in Zaniolo, Ceri et al. 1997). The problems concern both the semantic of the answers to queries and the representation of requested properties. To name some semantic problems they concern the extent to which the language should be closed around the temporal properties: In for instance a bi-temporal model, should the answers account for bi-temporal properties or should answers project into models that encompass a constant transaction time or a constant valid time, or some kind of models in which distances in time could be compared. Among the problems with representation it is not at all obvious how sets of bi-temporal properties should be represented on a two-dimensional surface. For lower level models where measures on the time dimensions are performed by means of attributes there is a choice to represent periods by a begin and an end time or by a begin-time and an interval. Also problems depending on the semantics of granularities are taken under serious discussion by several authors (e.g. by Snodgrass 2000). When does two periods coalesce and when are they separated by an intermediate period hided due to the grained gradualism of the time values?

We would like to end the paper by touching on some few general considerations derived from the approach we have used in the examination of the semantics of temporal models.

Normally a model has to capture information of many different types of properties each type of property with its own semantic. You may think of temporal properties like those we have been occupied with above, of space-properties, of amount of for example goods, of movements, of the more abstract money-values, etc. Each property type should as far as possible be examined by high-level models to insure a consistent and complete handling of the properties in question. When the property types are brought together in a common model there should be translation rules maintaining the consistence and completeness in the common model. One should also be aware that often there are dependencies in between different property types that will put further constraints on the common model. Thus for example movements will involve as well temporal as space properties and possibly also amount properties. We have no idea of general methods or procedures of how different property types at higher levels should be integrated in a common model. In this paper we have integrated the properties from the different time dimensions in a common ER model by regarding dependencies among the dimensions and adding the influence from each dimension successively in an ordered manner.

The ontology of different property types will play a role in the integration and in the choice of a common model. In our temporal case the pure existences of phenomena are prior to their existences in time. Similar can be said about most other phenomena. Their properties will be a specialisation of the general property of existence. Further, when we look at the pure existences the ER model used here is a perfect high-level model to capture information of these. It maps existing entities and their mutual relationships and is equipped with a language build over the predicate calculus comparing and measuring existences accounting for the binary property of existence. When information of phenomena of different specialised property types are integrated in an ER model their general property of existence is maintained at the high level, and their special properties are maintained by added facilities to the ER model. In our case we have added attribute types mapping time-periods and supported their special semantics by functions that integrate into the ordinary language of the ER model.

The conceptual level of the common model, the schema of the ER model, is not always able directly to map the concepts that comprise the specialised properties. In particular special semantics for relationships should be provided for and guarded in the common model. In the case of temporal properties we have seen that time varying simple references are mapped into the ER model by means of extra intermediate entity types and various constraints utilising the semantics of the period attributes.

Traditionally a lexical model, often a relational model of some variant, is used as a common model integrating information of phenomena with different property types. One reason can be that lexical models have a close accordance with natural languages. Another reason is probably that ultimate the computer requires that the information is stored lexically as bit-sequences, that lexical models ease the translation into these sequences, and that models on today's computers are lexical models. But, and that is what we think is correct, reasoning and modelling directly on the lexical level is both circumstantial and difficult with risk of loosing the semantically overview. The way we have illustrated it here, where temporal modelling is performed at a high level, and where we look for and used rules for translating the high-level models into the lower level ER model and next into different lexical models, may be a better choice.

References:

Batini, C., S. Ceri, et al. (1992). Conceptual Database Design. An Entity-relationship Approach, The Benjamin/Cummings Publishing Company, Inc.

Chen, P. P. (1976). "The Entity-Relationship Model - Toward a Unified View of Data." ACM, Transaction on Database Systems **1**(1).

Clifford, J., A. Croker, et al. (1993). On the Completeness of Query Languages for Grouped and Ungrouped Historical Data Models. Temporal Databases. Theory, Design, and Implementation. R. Snodgrass**:** 497-533.

Codd, E. F. (1979). "Extending the Database Relational Model to Capture More Meaning." ACM Transactions on Database Systems **4**(4): 397.

Elmasri, R., I. El-Assal, et al. (1990). Semantics of Tempoarl Data in an Extended ER model. 9th International Conference on The Entity-Relationship Approach, Lausanne Switzerland.

Elmasri, R. and S. B. Navathe (2000). Fundamentals of Database Systems, 3. edition, Addison-Wesley.

Elmasri, R. and S. B. Navathe (2000). Fundamentals of Database Systems, Third Edition, Addison-Wesley.

Embley, D. W. (1997). Object Database Development, Concepts and Principles, Addison-Wesley.

Garcia-Molina, H., J. D. Ullman, et al. (2002). Database Systems: The Complete Book. New Jersey, Printice-Hall, Inc.

Gregersen, H. and C. S. Jensen (1998). Conceptual Modeling of Time-Varying Information. Aalborg, Denmark, TimeCenter.

Gregersen, H., L. Mark, et al. (1998). Mapping Temporal ER Diagrams to Relational Schemas. Aalborg, Denmark, TimeCenter.

Gregersen, H., L. Mark, et al. (1998). Mapping Temporal ER Diagrams to Relational Schemas½. Aalborg, denmark, TimeCenter.

Kraft, P. and J. O. Soerensen (1998). Accessing Data Bases through Interface Views using a Unified Graph-Oriented Entity-Relationship Model. Integrated Design & Process Technology (IDPT), Berlin, Society for Design and Prtocess Science.

Kraft, P. and J. O. Soerensen (2001). Translation of a High-level Temporal Model into Lower Level Models. Conceptual Modeling - ER 2001, Yokohama, Japan, Springer.

Kraft, P. and J. O. Sørensen (2001). En Uniform Entity_Relationship Model, Dafolo Forlag.

McBrien, P., A. H. Seltveit, et al. (1992). An Entity-Relationship Model Extended to Describe Historical Information. CISMOD 92, Bangalore, India.

Navathe, S. B. and R. Ahmed (1993). Temporal Extensions to the Relational Model and SQL. Temporal Databases. Theory, Design, and Implementation. R. Snodgrass, Benjamin/Cummings Publishing Company, Inc.**:** 92-109.

Snodgrass, R. T. (2000). Developing Time-Oriented Database Applications in SQL, Morgan Kaufmann Publichers.

Teory, T. J. (1990). Database Modelling and Design. The Entity-Relationship Approach, Morgan Kaufmann.

Zaniolo, C., S. Ceri, et al. (1997). Advanced Database Systems, Morgan Kaufmann Publishers, Inc.