Object-Relational Representation of a Conceptual Model for Temporal Data Warehouses*

Elzbieta Malinowski** and Esteban Zimányi

Department of Informatics & Networks,
Université Libre de Bruxelles
emalinow@ulb.ac.be, ezimanyi@ulb.ac.be

Abstract. Temporal Data Warehouses (TDWs) allow to manage time-varying multidimensional data by joining the research of Temporal Databases and Data Warehouses. TDWs raise different issues such as temporal aggregations, multidimensional schema versioning, etc. However, very little attention from the research community has been drawn to conceptual modeling for TDWs and its subsequent logical representation. In this paper, we present a mapping transforming our conceptual model for TDW design into the conventional ER and an object-relational models. For the latter, we show some examples using the SQL:2003 standard. We include the mapping for time-varying levels, hierarchies, and measures. We also discuss the inconveniences of a pure relational representation.

1 Introduction

Data Warehouses (DWs) store and provide access to large volumes of historical data supporting the decision-making process. The structure of DWs is usually represented as a *star schema*, consisting of fact and dimension tables. A *fact table* contains numeric data called *measures*, e.g., sales. *Dimensions* are used for exploring the measures from different analysis perspectives. They usually contain hierarchies that allow to analyze detailed and generalized data using the roll-up and drill-down operations of On-Line Analytical Processing (OLAP) systems.

Current DW models include a time dimension that is used for grouping purposes (the roll-up operation) and also serves as a time-varying indicator for measures, e.g., sales in March 2005. However, the time dimension cannot be used for representing changes in other dimensions.

On the other hand, Temporal Databases (TDBs) allow to represent time-varying information. Two different temporal types[1] are considered: *valid time* (VT) and *transaction time* (TT) that indicate, respectively, when the data is

* The work of E. Malinowski was funded by a scholarship of the Cooperation Department of the Université Libre de Bruxelles.

** Currently on leave from the Universidad de Costa Rica.

[1] Usually called time dimensions; however, we use the term "dimension" in the multidimensional context.

E. Dubois and K. Pohl (Eds.): CAiSE 2006, LNCS 4001, pp. 96–110, 2006.

true in the modeled reality and when it is current in the database. If both temporal types are used, they define *bitemporal time* (BT). Further, the *lifespan* (LS) allows to record changes in time for an object as a whole.

Temporal Data Warehouses (TDWs) join the research achievements of TDBs and DWs in order to manage time-varying multidimensional data. In [6, 8] we proposed MultiDimER, a conceptual model for modeling TDW applications. In this work we give a logical representation for this conceptual model.

Two approaches can be used for logical-level design: using normalization (e.g., [5]) or mapping a conceptual model into a logical model (e.g., [4]). We choose the latter since there are no well-accepted normal forms for TDBs even though some formal approaches exist (e.g., [5, 13]). Further, the purpose of normalization is to avoid the problems of data redundancy, potential inconsistency, and update anomalies. However, the usual practice in DWs is to de-normalize relations to improve performance and to avoid the costly process of joining tables.

In this paper, we present a mapping of a conceptual model for time-varying multidimensional data into a classical ER and into an object-relational (OR) models. The ER representation allows a better understanding of the constructs used in our model. Further, to assist implementers who use our model for conceptual design, we propose general rules for mapping directly our model to the OR model. We consider the particularities of the different elements of a multidimensional model and exploit the features of OR databases for modeling complex objects. In this paper, we do not consider operations in TDWs. There are not easy to cope with since (1) different time granularities between dimension data and measures should be considered, and (2) as demostrated by, e.g., [2, 11], solutions for managing different schema versions should also be included.

Section 2 surveys works related to TDWs. Section 3 briefly presents the main features of the MultiDimER model. Section 4 presents our rationale for using the ER and OR models. Section 5 describes general rules for transforming temporal types to both models. Sections 6 to 8 present, respectively, mappings of temporal levels, temporal links between levels, and temporal measures. Finally, the conclusions are given in Section 9.

2 Related Work

TDWs raise many challenging issues, e.g., the inclusion of temporal types (e.g., [9]) or correct aggregation in the presence of data and structure changes (e.g., [2, 11]). However, very few conceptual models for TDWs have been proposed (e.g., [2, 11, 12]). These models formally describe the temporal support for multidimensional models, nevertheless, they do not consider several aspects proposed in our model, e.g., (non-) temporal relationships between (non-) temporal levels. Further, these models do not provide an associated logical representation.

On the other hand, [1] introduces a temporal star schema that differs from the classical one by the fact that the time dimension does not exist; instead the rows in all tables of the schema are timestamped. They compare this model with the classical star schema taking into account database size and performance.

They conclude that the temporal star schema facilitates expressing and executing queries, it is smaller in size, and it does not keep redundant information.

Since to the best of our knowledge there are not proposed solutions for a logical representation of TDWs, we briefly review logical models for TDBs with the goal to adapt some of these ideas for the logical representation of TDWs.

One approach for logical-level design of TDBs is to use normalization. Temporal functional dependencies have been defined (e.g., [5, 13]). New temporal normal forms (e.g., [13]) or extensions of conventional ones (e.g., [5]) have been also proposed. Most of these approaches rely on the first normal form (1NF), however, the non-first normal form (NF2) was proposed for solving the limitations of the 1NF for modeling complex data. The NF2 allows structured domains, collection domains, and relation-valued domains, which are also included in the SQL:2003 standard under the name of object-relational model [10].

Another approach for logical-level design of TDBs is based on mapping conceptual models. While this is the usual practice for conventional (i.e., non-temporal) database design, to the best of our knowledge only [4] propose such an approach. In general, the mapping of timestamped elements produces a table for each entity type that includes lifespan, a separate table for each timestamped monovalued attribute, and one additional table for each multivalued attribute, whether timestamped or not. This approach gives a significant number of tables since entities and their time-varying attributes are represented separately. Further, it is not intuitive for expressing the semantics of the modeled reality.

3 Overview of the MultiDimER Model

The MultiDimER model is a conceptual model allowing to represent time-varying multidimensional data [6][2]. Figure 1 shows notations used in our model and Figure 2 presents the metamodel. A *schema* is defined as a finite set of dimensions and fact relationships. A *dimension* is an abstract concept representing either a level, or one or more hierarchies. Levels are represented as entity types (Figure 1 a). An instance of a level is called a *member*.

A *hierarchy* contains several related levels (Figure 1 b) that are used for roll-up and drill-down operations. Given two consecutive levels of a hierarchy, the higher level is called *parent* and the lower level is called *child*. A level that does not have a child level is called *leaf*. Hierarchies express different structures according to an analysis *criterion* (Figure 1 c), e.g., geographical location.

Levels have one or several *key attributes* (represented in bold and italic in Figure 1) and may also have other *descriptive attributes*. Key attributes of a parent level define how child members can be grouped. Key attributes in a leaf level or in a level forming a dimension without hierarchy indicate the granularity of measures in the associated fact relationship.

A *temporal level* is a level for which the application needs to keep its time-varying characteristics. We allow support for both temporal attributes and level

[2] The non-temporal version of the model is presented in [7].

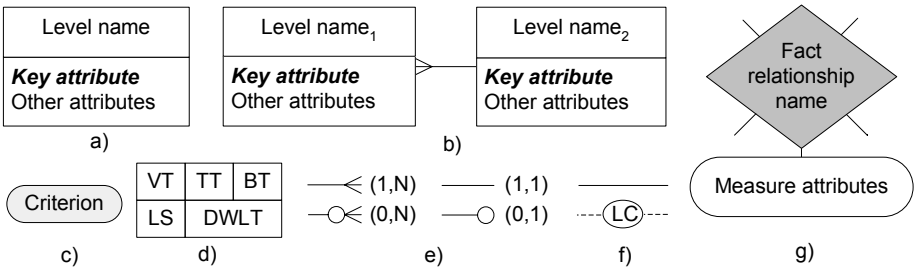


Fig. 1. Notations: a) one-level dimension, b) hierarchy, c) analysis criterion, d) temporal types, e) cardinality ratios, f) cardinality types, and g) fact relationship

lifespan. Figure 1 d) shows the different temporal types. We allow valid time (VT), transaction time (TT), bitemporal time (BT), and lifespan (LS) coming from source systems (if available), and data warehouse loading time (DWLT)[3].

The relationship between two levels is characterized by *cardinalities* (Figure 1 e), which restrict the minimum and the maximum number of members in one level that can be related to a member in another level. This cardinality may be interpreted in two possible ways. The *snapshot cardinality* is valid every time instant whereas the *lifespan cardinality* is valid over the entire members lifespan. The former is represented as a continuous line and the latter as a dotted line with LC symbol (Figure 1 f). The presence of only one cardinality symbol indicates that both cardinalities are the same. Further, the relationship between levels may include different temporal types: VT, TT, BT, and/or DWLT. There is no LS support for relationships since they do not exist by themselves without their participating levels.

A *fact relationship* (Figure 1 g) represents an n-ary relationship between leaf levels. It may contain attributes commonly called *measures*. In the MultiDimER model measures are *temporal*, i.e., they always require to include a temporal element (VT, TT, BT, and/or DWLT) [8].

An example of using our notation is given in Figure 3. It represents a schema for analysis of product sales where, for example, changes to products, categories, and relationships between them are kept. Further, changes in measure values are represented using a temporal type (VT in the figure) instead of relying on the conventional Time dimension.

4 Motivation for Mapping to the ER and OR Models

For implementing the MultiDimER model we propose two mappings: to the ER and OR models. The former is a well-known and widely-used model for conceptual modeling. Therefore, the ER representation of the constructs of the MultiDimER model allows a better understanding of their semantics. Further, the

[3] In [6, 8] we present our rationale for the inclusion of different temporal types in TDWs.

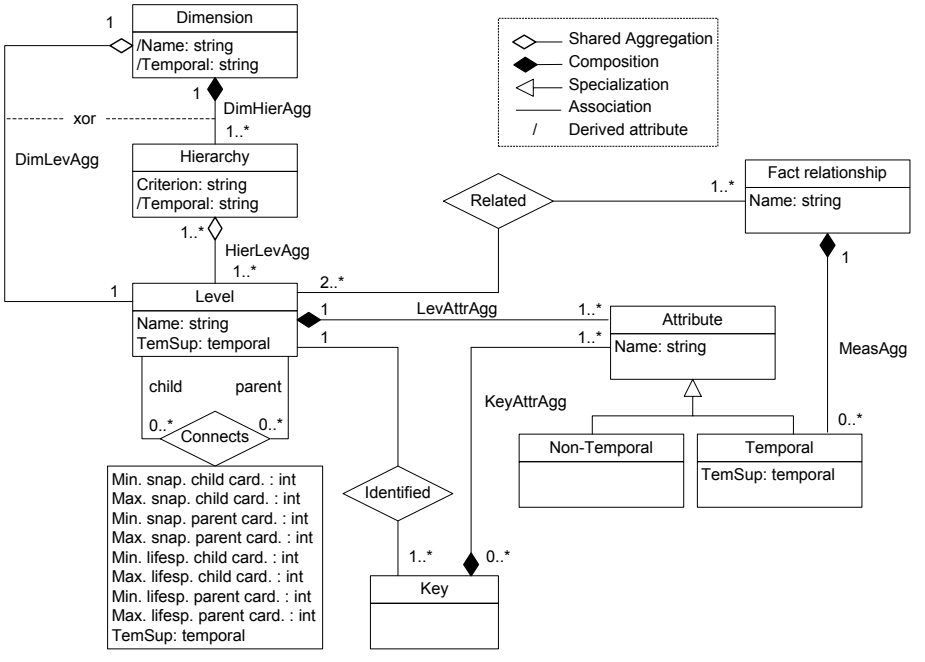


Fig. 2. Metamodel of the temporally-extended MultiDimER model

transformation of the ER model into operational data models is well understood (e.g., [3]) and this translation can be done using usual CASE tools.

On the other hand, in order to better assist the implementers who use the MultiDimER model for conceptual design of TDWs, we propose mapping rules that allow a direct translation of schemas from our model to the OR model. We choose the OR model as logical model since it preserves the foundations of the relational model while extending its modeling power. It also offers upward compatibility with existing relational languages allowing to "flatten" non-atomic data to a conventional 1NF. Further, the OR model allows to better represent the real world by inherently grouping related facts into a single row. In addition, OR features are also included in the SQL:2003 standard [10] and in leading DBMSs, e.g., Oracle or Informix.

5 Mapping of Temporal Types

The temporal support in the MultiDimER model is added in an implicit manner, i.e., the timestamp attributes used for capturing a temporal aspect are hidden using instead pictograms. Therefore, the transformation of the time-related data into classical non-temporal structures of the ER model requires additional attributes for timestamps, which are manipulated as usual attributes.

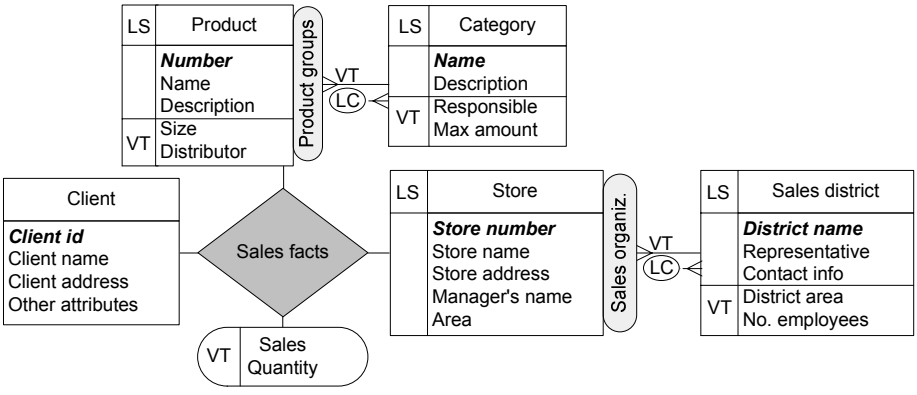


Fig. 3. Example of conceptual schema for a TDW

Further, mapping of temporal types to the ER model depends on whether these types are used for events or states. *Events* represent something that happens at a particular time point whereas *states* something that has extent over time. For the former an *instant* is used, i.e., a time point on an underlying time axis. A state is represented by an *interval* or *period* indicating the time between two instants. Sets of instants and sets of intervals can also be used.

Figure 4 presents different options for mapping VT: a monovalued attribute for an instant (Figure 4 a), a multivalued attribute for a set of instants (Figure 4 b), a simple composite attribute for a period (Figure 4 c), and a multivalued composite attribute for a set of periods (Figure 4 d). Notice that a set of periods or instants are used when the attribute has the same value in different periods or instants of time.

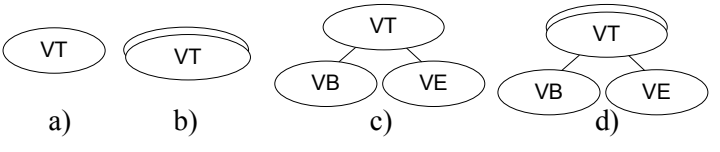


Fig. 4. Different representations of VT in the ER model

As LS can be represented by a period or a set of periods, it is transformed into a simple or multivalued composite attribute, respectively. The latter allows to include discontinuous lifespans, e.g., a professor leaving for sabbatical during some period of time. For representing TT, the usual practice in TDBs is to use a period (or a set of periods) similar to LS. Since DWLT represents the time when data was loaded into a TDW, an instant is used for representing this temporal type, which is transformed into a simple attribute in the ER model.

To specify the mapping rules to the OR model, we use the different elements included in the SQL:2003 standard. For example, we use a multiset composite

type, which allows to store unordered collections of values. Alternatively, the array composite type could be used if there is a limited number of elements. Further, since *structured user-defined types* are analogous to class declarations in object languages, they allow to group semantically related attributes[4].

The mapping rules to the OR model consider a multivalued attribute in the ER model as a multiset attribute while a composite attribute in the ER model as an attribute of a structured type comprising specified component attributes.

The mapping of different temporal types from our model to the OR model is based on the following rules:

Rule 1: A temporal type representing an instant is mapped to an attribute of date or timestamp type.

Rule 2: A temporal type representing a set of instants is mapped to a multiset attribute of date or timestamp type.

Rule 3: A temporal type representing a period is mapped to an attribute of a structured type composed of two attributes of date or timestamp type.

Rule 4: A temporal type representing a set of periods is mapped to a multiset attribute of a structured type consisting of two attributes of date or timestamp type.

For example, the different options for VT in Figure 4 can be represented in SQL:2003 as follows:

```
create type InstantT as date;
create type InstantSetT as (InstantT multiset);
create type PeriodT as (Pbegin date, Pend date);
create type PeriodSetT as (PeriodT multiset);
```

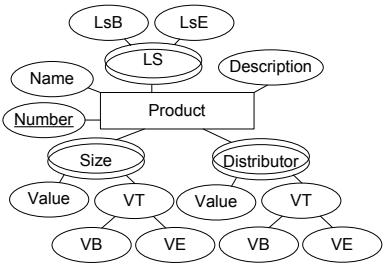
6 Mapping of Temporal Levels

Changes in a level can occur either for a member as a whole (e.g., deleting a product) or for attribute values (e.g., changing a product's size). Representing these changes in TDWs is important for analysis purposes, e.g., to discover how the exclusion of some products or the change in their sizes influences sales.

In the MultiDimER model changes to a level member as a whole are represented using the LS symbol next to the level name, e.g., the Product level in Figure 3. For representing changes in attribute values (Size and Distributor in the figure), we use attribute timestamping. We group temporal attributes in our model to ensure that they can be distinguished from non-temporal attributes and to minimize the number of symbols.

A level in our model corresponds to a regular entity type in the ER model. Each temporal attribute is represented in the ER model as a multivalued composite attribute that includes an attribute for the value and another attribute for a temporal type. Notice, that using a multivalued attribute allows to have

[4] Due to space limitations, in this paper we do not consider methods.



a)

Product

Sid	LS		Number	Other attrib.	Size		
	LsB	LsE			Value	VT	
						VB	VE
1	05/2002	NOW	QB876	...	10	05/2002	08/2002
					20	09/2002	07/2003
					15	08/2003	NOW
2	05/2002	NOW	QD555	...	18	05/2002	NOW
3	05/2002	08/2003	QE666	...	25	05/2002	08/2003
	09/2004	NOW			18	09/2004	NOW

b)

Fig. 5. A temporal level: a) the ER model and b) the OR representation

different values (e.g., sizes or distributors) of the attribute in different periods of time. For example, the transformation of the Product level (Figure 3) to the ER model is shown in Figure 5 a).

Mapping the corresponding ER model to the relational model gives four tables: one with all monovalued attributes and one for each multivalued attribute. This representation is not very intuitive since attributes of a level are stored as separate tables. It also has well-known performance problems due to the required join operations.

An OR representation allows to overcome these drawbacks. It preserves more semantics keeping together in a single table a level and its temporal attributes. The mapping of temporal attributes is straightforward:

Rule 5: An attribute with temporal support is mapped to the OR model as a multiset attribute of a structured type composed of two attributes: one for representing the value and another one for the associated temporal type.

For example, given the declarations for the temporal types in Section 4, the type for the Size attribute[5] is defined as follows: create type SizeT as (Value real, VT PeriodT). Since Size is a multivalued attribute, we represent it as a collection type using either array or multiset, e.g., create type SizeCT as (SizeT multiset).

Mapping a level to the OR model is straightforward once the types for its attributes are defined:

Rule 6: A level is mapped to a relation containing all its attributes and an additional attribute for a key. If a level has LS support, an additional attribute as specified by Rules 3 or 4 should be included.

Figure 5 b) shows the OR schema using a tabular representation containing the member key, the lifespan, and all its attributes represented together in the same table. It corresponds to a so-called temporally-grouped data model, which is considered as more expressive for modeling complex data.

The Product level can be represented in several ways in SQL:2003. For example, two types of tables can be used. *Relational tables* are usual tables while *typed tables* are tables that use structured user-defined types for their defini-

[5] For simplicity we do not represent in the figure the Distributor attribute, which can be mapped similarly to the Size attribute.

tion. Typed tables contain in addition an automatically-created *self-referencing column* keeping the value that uniquely identifies each row, i.e., a surrogate.

Surrogates are important in DWs for ensuring better performance during join operations and independence from transactional systems. Further, surrogates do not vary over time allowing to include historical data in an unambiguous way.

Therefore, to define a table for the Product level (Product) we use a typed table[6]. The declaration of a typed table requires first the definition of a type (ProductT) for the elements of the table:

```
create type ProductT as (LS PeriodSetT, Number integer, Name character varying(25),
    ..., Size SizeCT) ref is system generated;
create table Product of ProductT (constraint prodPK primary key (Number),
    ref is Sid system generated);
```

The clause ref is Sid system generated indicates that Sid is a surrogate attribute automatically generated by the system. In SQL:2003 these surrogates can also be generated by the user or derived from one or more attributes.

Until now we have discussed the representations of VT and LS. However, VT and LS can be combined with TT and/or DWLT. They can be mapped according to the explanations given in this section and in Section 5.

7 Mapping of Child-Parent Relationships

7.1 Non-temporal Relationships

Non-temporal relationships indicate that either these relationships never change or if they do, only the last modification is kept. To avoid an incorrect management of hierarchies and dangling references between levels, non-temporal relationships may only link levels that do not keep their LS and do not include VTs for their key attributes, which are used for aggregation purposes [6].

Figure 6 a) represents a hierarchy with a non-temporal relationship. The relationship between child and parent levels corresponds to a usual binary relationship in the ER model as shown in Figure 6 b)[7].

For obtaining the corresponding OR schema, first we represent each level as explained in Section 6. Then, we use the traditional mapping for binary many-to-one relationships.

Rule 7: A non-temporal many-to-one relationship between child and parent levels is mapped to the OR representation by including a parent key in the child level table.

For example, the mapping of the Product level and the Product–Category relationship gives the same relation as the one in Figure 5 b) including an additional attribute in the Product table with the foreign key of the Category table.

[6] For simplicity, in the examples we omit full specification of constraints and additional clauses required by the SQL:2003 standard.

[7] For simplicity we do not present level attributes.

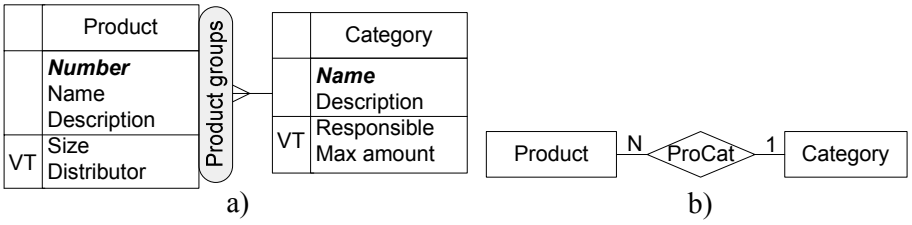


Fig. 6. A hierarchy with a non-temporal relationship: a) the MultiDimER model and b) the ER model

To define the Product table in SQL:2003 first we need to create a typed table Category with the surrogate in the Sid attribute. Then, we specify:

```
create type ProductT as (Number integer, ..., Sizes SizeCT, CatRef REF(CategoryT)
    scope Category references are checked) ref is system generated;
create table Product of ProductT (constraint prodPK primary key (Number),
    ref is Sid system generated);
```

The Product type includes a reference (REF) type that points to the corresponding row in the Category table. In this way, the OR approach replaces value-based joins with direct access to related rows using the identifiers.

7.2 Temporal Relationships

Temporal relationships allow to keep track of the evolution of links between parent and child members. These relationships can link non-temporal levels (Figure 7 a) and 8 a)) or temporal levels (Figure 3). For the former, in order to ensure correct measure aggregations and avoid dangling references, the

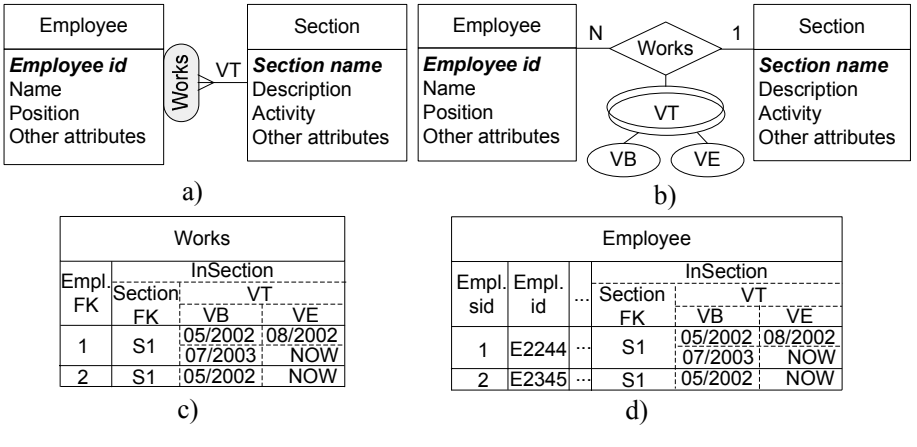


Fig. 7. Temporal relationships linking non-temporal levels: a) the MultiDimER model, b) the ER model, c) and d) alternative OR representations

modifications of levels are not allowed [6]. To represent temporal relationships we place the corresponding temporal symbol (VT, TT, BT, and/or DWLT) on the link between levels (Figure 7 a).

Temporal relationships have associated snapshot and lifespan cardinalities. In the example in Figure 7 a) these cardinalities are the same, many-to-one, indicating that an employee may work only in one section and if he returns after a leave, he must be assigned to the same section. In the example of Figure 8 a), the relationship has a many-to-one snapshot cardinality (continuous line) and a many-to-many lifespan cardinality (dotted line with the LC symbol), i.e., at each instant an employee works in exactly one section, but he may work in several sections over his entire lifespan.

Figure 7 b) shows the ER model for Figure 7 a). We use a multivalued composite attribute for representing VT since an employee can be hired several times in the same section.

For the OR representation we can either create a separate table for the Works relationship (Figure 7 c) or include a multivalued attribute in the child-level table, e.g., include in the Employee table an attribute for the Section surrogates with its temporal characteristics (Figure 7 d).

The definition of the Works relation in SQL:2003 requires that Employee and Section tables have already been declared as typed tables. We do not use a typed table for representing the Works relationship since the relationship does not exist without their levels.

```
create type WorksT as (SecFK REF(SectionT) scope Section references are checked,
    VT PeriodSetT);
create table Works (EmplFK REF(EmplT) scope Employee references are checked,
    InSection WorksT);
```

The SQL:2003 declaration for Employee in Figure 7 d) requires to include the InSection attribute of the WorksT type in the Employee type. This representation expresses in a better way the semantics of the relationship since all changes of working place of an employee are included in the same row.

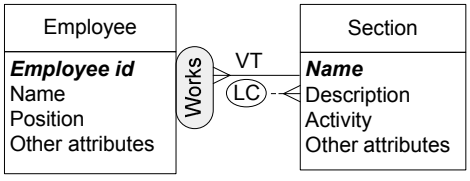
On the other hand, if the snapshot and the lifespan cardinalities are different (Figure 8 a), the lifespan cardinality is considered when mapping to the ER model. The mapping to the ER model is similar to the one in Figure 7 b) except that the cardinalities are many–to–many.

As for the previous case, two different OR representations may be used: either a separate table for the Works relationship or a table for a child level (Employee) with an additional attribute representing this relationship; the latter is shown in Figure 8 b). Notice, that the foreign key is represented as set of values since an employee can work in many sections over his lifespan. This leads to the inclusion of a multiset type for the InSection attribute of the EmployeeT type:

```
create type EmployeeT as (EmplID integer, ..., InSection WorksT multiset);
```

Another case arises when a child member is related to many parent members at every time instant[8], i.e., the snapshot and lifespan cardinalities between child

[8] Called in [7] *non-strict* hierarchies.



Employee

Empl. sid	Empl. id	...	InSection		
			Section FK	VT	
				VB	VE
1	E2244	...	S1	05/2002	08/2002
				07/2003	NOW
			S2	09/2002	06/2003
2	E2345	...	S1	05/2002	NOW

b)

Fig. 8. Temporal relationships linking non-temporal levels: a) the MultiDimER model with snapshot and lifespan cardinalities and b) the OR representation

and parent levels are many-to-many. This situation can be mapped in the same way as the previous one.

Summarizing, for mapping temporal relationships between the child and parent levels to the OR model the following rule is used:

Rule 8: First, a structured type composed of two attributes is defined: one attribute for surrogates of the parent level and another one for the corresponding temporal type. Then, this structured type is used for defining a simple or a multiset attribute depending on whether the cardinality between child and parent levels is many-to-one or many-to-many, respectively. Let us call this attribute TemRel. Finally, one of two possible OR representations can be used:

1. Creating a new relation that contains an attribute for the surrogate keys of the child level and the TemRel attribute.
2. Extending the relation corresponding to the child level with the TemRel attribute.

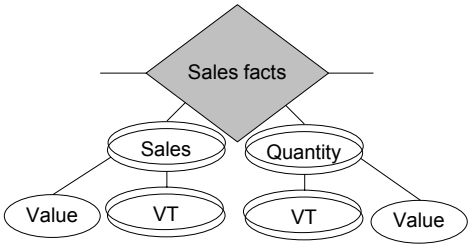
Even though the second option preserves more semantics, the choice among the alternative OR representations may depend on physical-level considerations for the particular DBMS, such as join algorithms, indexing capabilities, etc. For example, defining the InSection attribute as a nested table in Oracle 10g, will require a join of two tables, thus not offering any advantage with respect to the solution of a separate table for the Works relationship. Notice that for the previous cases, the relational model only offers the option of creating a separate table for the Works relationship.

Additionally, a relationship between levels can include TT and/or DWLT for which the mapping specified in Section 5 can be used.

8 Fact Relationships with Temporal Measures

The MultiDimER model includes temporal support for measures as shown in Figure 3. Notice that the usual Time dimension does not need to be attached to the fact relationship. In fact, temporal support applies to the whole schema of the TDW, i.e., to measures and dimensions.

Depending on analysis needs, time-varying measures may represent either events or states. In the following example, due to space limitations, we only refer to measures whose VT is represented as an instant with granularity month. Nevertheless, the results may be straightforwardly generalized if VT is represented by a period or a set of periods.



a)

Sales facts				
Client FK	Product FK	Store FK	Sales	VT
1	P1	W1	10000	05/2002
1	P1	W2	20000	05/2002
...	...	...	...	...
1	P1	W1	15000	06/2002
1	P1	W2	5000	06/2002
...	...	...	...	...
1	P1	W1	18000	07/2002
1	P1	W2	20000	07/2002

b)

Sales facts						
Client FK	Product FK	Store FK	Sales		Quantity	
			Value	VT	Value	VT
1	P1	W1	10000	05/2002	100	05/2002
			15000	06/2002		07/2002
			18000	07/2002	150	06/2002
1	P1	W2	20000	05/2002	200	05/2002
				07/2002	50	06/2002
			5000	06/2002	80	07/2002
...	...	...	...	...	...	...

c)

Fig. 9. Temporal measures: a) the ER representation, b) the relational table for the Sales measure, and c) the OR representation

A fact relationship in the MultiDimER model corresponds to an n-ary relationship in the ER model. Measures as attributes of a relationship are mapped to the ER model in the same way as temporal attributes of a level. Therefore, each measure is represented as a multivalued composite attribute (Figure 9 a).

Mapping this fact relationship to the relational model in 1FN gives two tables. In Figure 9 b) we only show the table with the Sales measure since the other table with the Quantity measure has similar structure. However, if additional information is available, this model can be simplified. For example, if all measures are calculated with respect to the same VT, they can be represented in one table and tuple timestamping can be applied.

The OR model creates also a separate table based on the following rule:

Rule 9: A fact relationship with temporal measures is mapped to the OR model by creating a new relation that includes as attributes the surrogate keys of the participating levels. In addition, every measure is mapped into a new temporal attribute according to Rule 5.

An example of the tabular OR representation is given in Figure 9 c).

However, even though the OR model allows to represent the changes in measure values for the same combination of foreign keys, in practice it may be not well suited for aggregations related to time. The objects created for every measure contain two-level nesting: one for representing different measure values for the same combination of foreign keys and another for representing a temporal element. Therefore, it is more difficult to express aggregation statements related to time accessing the second-level nesting. As a consequence, the relational representation is more adequate in order to represent in a more "balanced" manner all attributes that may be used for aggregation purposes.

9 Conclusions

The temporally-extended MultiDimER model [6, 8] is used for modeling time-varying data for DW applications. It is symmetric in the sense that it allows to represent changes for both measures and dimensions. In this paper we presented a mapping of this conceptual model into the ER model and the OR model.

We used an object-relational approach based on the SQL:2003 standard that allows better to represent complex data and preserve as much TDW semantics as possible. We discussed mappings for a temporal level and for hierarchies. Finally, we referred to temporal measures in fact relationships.

The object-relational model allows to better represent time-varying levels and hierarchies than the classical relational model. In the former model a level and its corresponding time-varying attributes are kept together while the relational model produces a significant number of tables with well-known disadvantages for modeling and implementation. Further, for representing a relationship between levels forming a hierarchy the object-relational model gives a designer several alternatives. Thus, he can choose the one considering semantics and physical level features of the particular OR DBMS. On the other hand, the relational model is more adequate for representing time-varying measures. It considers in the same manner all attributes including the ones that represent time, thus it facilitates aggregation procedures.

The proposed mapping shows that TDWs can be implemented using current OR DBMSs. Further, the features of OR databases for representating complex objects facilitate the implementation of attribute timestamping. As opposed to tuple timestamping, which is mostly used in the relational representation of TDBs, attribute timestamping not only allows a better representation of reality (temporal changes are represented only for the specified attributes) but also saves storage space during implementation (the values of attributes that do not vary on time are not repeated).

We have already undertaken a real-scale study in the domain of TDWs to evaluate the usability of our conceptual model and the feasibility of its implementation using Oracle 10g. The results will be reported in a forthcoming paper, but a first analysis has already given encouraging indications that allow us to move to the next research step, i.e., developing a methodology for TDW design.

The proposed mapping may vary according to the expected usage patterns, e.g., data mining algorithms, and specific features of the target implementation system. For example, a user may choose a multidimensional tool-specific storage (e.g., Analytic Workspace in Oracle 10g) instead of relying on more general solutions as the ones proposed in this paper.

References

1. R. Bliujute, S. Slatenis, G. Slivinskas, and C. Jensen. Systematic change mangement in dimensional data warehousing. Technical report, Time Center, TR-23, 1998.
2. J. Eder, C. Koncilia, and T. Morzy. The COMET metamodel for temporal data warehouses. In *Proc. of the 14th Int. Conf. on Advanced Information Systems Engineering*, pages 83–99, 2002.
3. R. Elmasri and S. Navathe. *Fundamentals of Database Systems*. Adison-Wesley, fourth edition, 2003.
4. H. Gregersen, L. Mark, and C. Jensen. Mapping temporal ER diagrams to relational schemas. Technical report, Time Center, TR-39, 1998.
5. C. Jensen and R. Snodgrass. Temporally enhanced database design. In M. Papazoglou, S. Spaccapietra, and Z. Tari, editors, *Advances in Object-Oriented Data Modeling*, pages 163–193. MIT Press, 2000.
6. E. Malinowski and E. Zimányi. A conceptual solution for representing time in data warehouse dimensions. In *Proc. of the 3rd Asia-Pacific Conf. on Conceptual Modelling*, pages 45–54, 2006.
7. E. Malinowski and E. Zimányi. Hierarchies in a multidimensional model: from conceptual modeling to logical representation. Data & Knowledge Engineering. To appear, 2006.
8. E. Malinowski and E. Zimányi. Inclusion of time-varying measures in temporal data warehouses. In *Proc. of the 8th Int. Conf. on Enterprise Information Systems*, 2006. To appear.
9. C. Martín and A. Abelló. A temporal study of data sources to load a corporate data warehouse. In *Proc. of the 5th Int. Conf. on Data Warehousing and Knowledge Discovery*, pages 109–118, 2003.
10. J. Melton. *Advanced SQL: 1999. Understanding Object-Relational and Other Advanced Features*. Morgan Kaufman Publisher, 2003.
11. A. Mendelzon and A. Vaisman. Temporal queries in OLAP. In *Proc. of the 26th Very Large Database Conference*, pages 243–253, 2000.
12. T. Pedersen, C. Jensen, and C. Dyreson. A foundation for capturing and querying complex multidimensional data. *Information Systems*, 26(5):383–423, 2001.
13. X. Wang, C. Bettini, A. Brodsky, and S. Jajodia. Logical design for temporal databases with multiple granularities. *ACM Transactions on Database Systems*, 22(2):115–170, 1997.