Multi-Valued Relationship Attributes in Extended Entity Relationship Model and Their Mapping to Relational Schema

Tauqeer Hussain*, Shafay Shamail, Mian M. Awais
Department of Computer Science
Lahore University of Management Sciences (LUMS), Lahore, Pakistan

Abstract

Conceptual modeling is one of the most important phases in designing database applications. The success of this design relies heavily on how clearly the real world requirements are represented in the conceptual model. To date, the Extended Entity Relationship (EER) model extended from the traditional Entity Relationship (ER) model is a widely used modeling technique during the phase of conceptual modeling. This paper identifies semantic ambiguities that are still present in the EER model leading to incorrect knowledge representation and eventually to incorrect design of relational database schema. These ambiguities are identified in case of many-to-many relationships which have their own attributes. This paper shows that mapping such relationships to a relational database schema generates relations having primary keys which cannot guarantee unique tuples for real world data thus violating the definition of a primary key. In addition, it shows that these relations may not satisfy second normal form. A number of such cases are elaborated and a new concept of multi-valued relationship attribute is introduced that can successfully represent these real world constraints. For this concept, a diagrammatic notation to use in ER diagram is introduced. A mapping algorithm to transform the corresponding EER model to a relational database schema is also defined. This concept of multi-valued relationship attribute and its mapping to relational schema generate relations which satisfy higher normal forms.

Keywords: Extended ER model, conceptual modeling, EER-to-Relational mapping, relationship attribute, normalization

1. Introduction

Designing a good database is one of the most important steps of systems design phase and it provides a strong foundation for the success of database applications. A database design methodology for relational databases is defined in three steps: 1) Conceptual modeling: the data requirements are conceptualized using a conceptual model representing the semantics of real world, 2) Mapping: the conceptual model is transformed into a set of candidate relations, and 3) Normalization: the candidate relations are further refined to remove data redundancy and to achieve higher degree of data integrity [Teorey86]. The most demanding and challenging step in this design methodology is conceptual modeling whereas the later steps are merely transformations [Engels92]. For conceptual modeling, the entity-relationship model [Chen76] has been successfully used for traditional database applications because of its ease of understanding and its convenience in representation [Engels92]. However, the task of capturing the semantics of data is a never-ending one [Codd79]. The ER model lacked modeling constructs like specialization (or generalization) to represent more complex requirements specially needed in applications of newer database technology [Teorey86]. In EER model, important enhancements to the traditional ER model were suggested in literature [Elmasri85, Teorey86, and Gogolla91]. These enhancements introduced concepts of subclass and superclass, class/subclass relationships, category which is a representation of union of different entity types and the related concepts of generalization and specialization [Elmasri04]. Thalheim (2000) notes the shortcomings of the ER literature as the use of ER concepts often

- lack a clear statement of the intended semantics
- applies different semantics to the same concept, and

* Corresponding author
In this paper, various situations are presented to appreciate that existing EER model does not have clear representation for relationship attributes. It results in relations which may not satisfy certain normal forms. A solution proposed in the literature is to introduce artificial constructs (weak entity types and more relationship types) [Thalheim00] which do not exist in the real world. This makes the schema less understandable [Thalheim00]. So, it is a problem not only of accurate knowledge representation but also of generating a normalized database schema. Analysis of this problem reveals that the definition of relationship attribute should be clarified in terms of the two concepts introduced in this paper - Single Valued Relationship Attribute (SVRA) and Multi-Valued Relationship Attribute (MVRA).

In the next section, we formally define concepts of SVRA and MVRA. We also define situations where a single MVRA is required on a relationship type, where multiple MVRA are required and where SVRA and MVRA together represent a real world situation. We also introduce the diagrammatic notation that can be used in ER diagram corresponding to these concepts. Finally, we define an algorithm which provides an EER-to-relational mapping and establish that the resulting relations satisfy those normal forms which are otherwise violated. The underlying assumption is that we are using EER diagram for conceptual modeling and the database schema required is for a relational database.

2. Multi-Valued Relationship Attributes

This paper introduces the concept of a multi-valued relationship attribute which is primarily different from nested attributes [Thalheim00], complex attributes [Elmasri04], and multi-valued attributes of entity types [Elmasri04]. This new concept solves conceptual differences as posed in the problems presented in the following sections.

We now present various scenarios for a common real world example of a sales system. In this system, CUSTOMER and PRODUCT are identified as entity types having appropriate attributes and PURCHASES is identified as a relationship type between these two entity types.

2.1. Many-To-Many (M:N) Relationship Type Attributes

Scenario 1: Consider a situation where a customer may purchase a number of products and a product can be purchased by a number of customers. We are interested in keeping track of the date when a customer purchases a product.

This situation is represented in Fig.1 using EER model notation. In order to keep track of the date when a customer purchases a product, ‘Date’ is marked as an attribute on the relationship type ‘Purchases’. This model can be viewed in terms of a semantic net model as shown in Fig. 2, where for example the relationship instance r1 relating customer (id) 29 to the product (code) P-101 has a value 14-Jan.-2003 for its ‘Date’ attribute.

When this part of the EER model is mapped to a relational database schema following the mapping algorithm given by Elmasri and Navathe [Elmasri04], the following relations are created:

CUSTOMER (ID, Name)
PRODUCT (Code, Desc)
PURCHASES (CustID$_{f.k.}$, ProdCode$_{f.k.}$, Date)

The underlined attribute represents a primary key of the relation whereas a foreign key is represented by an attribute with subscript $f.k$. For relations CUSTOMER and PRODUCT, ID and Code are primary keys respectively, whereas, for the relation PURCHASES created corresponding to M:N relationship type ‘Purchases’, primary keys of relations corresponding to participating entity types namely CUSTOMER and PRODUCT become foreign keys and together form the primary key of relation PURCHASES. This model and its mapping (as per the algorithm given by Elmasri and Navathe [Elmasri04]) works fine as long as a customer purchases a...
product only on a single date, that is, only a single date is defined for each relationship instance of Fig. 2. In terms of functional dependency, this constraint can be written as:

\[ \text{CustID, ProdCode} \rightarrow \text{Date} \]

But this is an unrealistic constraint for most of the real world situations where a customer is not bound to purchase a product only once. This leads us to scenario 2 where we discuss the situation of a relationship attribute having more than one value.

**Definition 1:** A Multi-Valued Relationship Attribute (MVRA) is a relationship attribute which may have more than one value for a relationship instance of the relationship set.

**Definition 2:** A Single-Valued Relationship Attribute (SVRA) is a relationship attribute which cannot have more than one value for a relationship instance of the relationship set.

A relationship attribute (single-valued or multi-valued) can now be defined mathematically as:

\[ A : R \rightarrow P(V) \]

This definition covers single-valued and multi-valued relationship attributes, as well as nulls. A null value is represented by the empty set. For single-valued relationship attributes, \( A(r) \) is always a singleton for each relationship instance \( r \) of the set \( R \); whereas there is no such restriction for a MVRA. Here \( A(r) \) refers to the value of attribute \( A \) for relationship instance \( r \).

Based upon the concept and definitions given above, a new notation is proposed to represent the concept of MVRA in the EER diagram. This notation is writing the name of the attribute in the set notation i.e. braces within the oval (symbol for attribute) corresponding to the idea that this attribute may have a set of values. Now ‘Date’ attribute in Fig. 1 changes to ‘\{Date\}’ in Fig. 4 for the revised situation.

**Definition 3:** An attribute \( A \) of relationship type \( R \) whose value set is \( V \) is a function from \( R \) to the power set \( P(V) \) of \( V \):

\[ A : R \rightarrow P(V) \]

Consequently the EER-to-relational mapping algorithm given by Elmasri and Navathe [Elmasri04] should also be modified to take care of this situation because otherwise there will be multiple tuples in the relation PURCHASES with identical values of ‘CustID’ and ‘ProdCode’ violating the primary key constraint for this relation. This modified algorithm for mapping of M:N relationship type is presented in Algorithm 1 below:

**Algorithm 1: Mapping M:N relationship type to a relational schema**

For every binary M:N relationship type \( R \) between entity types \( E_1 \) and \( E_2 \) having a set of multi-valued relationship attributes \{MVRA\},

---

**Fig. 2: Relationship Instances with SVRA**

**2.2. MVRA**

**Scenario 2:** A customer purchases the same product on different dates and we want to keep track of all such purchases

This scenario is represented in Fig. 3 where the relationship instance \( r_1 \) has two values for the attribute ‘Date’: 14-Jan-2003 and 12-May-2003 against two purchases of the same product by the same customer. This requires a relationship attribute which can have more than one value (defined as multi-valued relationship attribute in definition 1) to be differentiated from the one which may have at most one value for a relationship instance (defined as single-valued relationship attribute in definition 2).

**Fig. 3: Relationship Instances with MVRA**

**Fig. 4: A M:N relationship type with a MVRA**
1. create a new relation $S$ to represent $R$ such that:
   \[
   \text{Attr}(S) = \{\text{PK}(E_1)\} \cup \{\text{PK}(E_2)\} \\
   \cup \text{Attr}(R) \cup \{\text{MVRA}\}
   \]
   where $\text{PK}(E_i)$ is primary key of the relation created for entity type $E_i$, and
   $\text{Attr}(R)$ is the set of simple attributes (or simple components of composite attributes) of $R$

2. \[
   \{\text{PK}(S)\} = \{\text{PK}(E_1)\} \cup \{\text{PK}(E_2)\} \cup \\
   \{\text{MVRA}\}
   \]

It should be noted that $\text{PK}(E_1)$ and $\text{PK}(E_2)$ are foreign keys in relation $S$. Step 2 of this algorithm suggests that MVRA should also be marked as part of the primary key of relation $S$ along with $\text{PK}(E_1)$ and $\text{PK}(E_2)$. Applying this algorithm, we get the following relation in our schema:

   PURCHASES (CustID f.k., ProdCode f.k., Date)

It is interesting to note that the given scenario could be modeled by introducing an artificial entity type PURCHASE and by adding its relationship types with CUSTOMER and PRODUCT; however, introduction of these artificial constructs produces a complex conceptual model which is less explicable [Thalheim00].

Now, let us extend this scenario a bit further.

### 2.3. Multiple MVRA

**Scenario 3:** Apart from the date, we want to keep track of the quantity of a product, as well, purchased in every instance.

In this scenario, the relationship type PURCHASES will have two attributes ‘Date’ and ‘Quantity’. The question then arises: Is each of these two attributes a MVRA? According to the definition of MVRA given above, the answer is, of course, yes; because each of these attributes may have multiple values for a single relationship instance present in the relationship set. This implies that, as per Algorithm 1, the relation PURCHASES will have attributes ‘Date’ and ‘Quantity’ both as a part of the primary key.

   PURCHASES (CustID, ProdCode, Date, Quantity)

### 2.4. MVRA and SVRA

**Scenario 4:** We have an additional constraint that a customer purchases a particular product always in the same quantity.

The attribute ‘Quantity’, in this case, is no more a MVRA but it is a SVRA; because it always has only a single value for each relationship instance. The ER model for this situation is given in Fig. 5

![Fig. 5: A M:N relationship type with a MVRA and a relationship attribute](image)

Applying Algorithm 1 on Fig. 5 for its transformation to a relational schema, a relation PURCHASES is created having attributes ‘CustID’ and ‘ProdCode’ as foreign keys, and the attributes ‘Date’ and ‘Quantity’. Since ‘Date’ is a MVRA, the primary key for this relation comprises of CustID, ProdCode, and Date. This solution, however, violates second normal form (2NF), in this case, due to the existence of the following functional dependency:

   \[
   \text{CustID, ProdCode} \rightarrow \text{Quantity}
   \]

This requires further refinement of the mapping algorithm (Algorithm 1) which is then given below as Algorithm 2:

**ALGORITHM 2: Mapping M:N relationship type generating a normalized relational database schema**

For every binary M:N relationship type $R$ between entity types $E_1$ and $E_2$,

1. create a new relation $S$ to represent $R$ such that:
   \[
   \text{Attr}(S) = \{\text{PK}(E_1)\} \cup \{\text{PK}(E_2)\} \\
   \cup \{\text{SVRA}\}
   \]
   where $\text{PK}(E_i)$ is primary key of the relation created for entity type $E_i$, and
   \{SVRA\} is the set of all single-valued relationship attributes of $R$

2. \[
   \{\text{PK}(S)\} = \{\text{PK}(E_1)\} \cup \{\text{PK}(E_2)\}
   \]

3. if there exists a MVRA of $R$, then:
   a. create a new relation $T$ such that:
      \[
      \text{Attr}(T) = \{\text{PK}(E_1)\} \cup \{\text{PK}(E_2)\} \cup \\
      \{\text{MVRA}\}
      \]
Applying this algorithm for relationship type PURCHASES, we get the following relations:

PURCHASES1 (CustID, ProdCode, Quantity)

and

PURCHASES2 (CustID, ProdCode, Date)

Each of these relations now satisfy 2NF and, if no other functional dependency violation occurs, 3NF and BCNF are also satisfied.

It should be noted that the concept of multi valued relationship attribute is not exclusive to M:N relationship types. It is as good as for other relationship types, as well.

3. Conclusion

In this paper we highlighted the deficiency of EER model in semantic representation for many-to-many relationship types. This deficiency results in unclear conceptual model and in a poor database design having violations of key constraints and of second normal form. In order to eliminate this deficiency, a new concept of multi-valued relationship type was formally defined. It was shown with various examples that the new concept of MVRA nicely resolved the semantic and normalization problems. For this new concept, an ER diagram notation and a mapping algorithm for its transformation to relational schema were also devised. It was demonstrated that the relations created using this algorithm satisfied the normal forms which were otherwise violated. In our future research, we intend to prove formally that if no other violations occur, the solution presented to the above stated problems would satisfy relations up to 4NF.

4. References


