Available online at www.sciencedirect.com
ScienceDirect
Information Sciences 177 (2007) 3679–3695
INFORMATION SCIENCES
AN INTERNATIONAL JOURNAL
www.elsevier.com/locate/ins

A grain preservation translation algorithm: From ER diagram to multidimensional model

Yen-Ting Chen [a,b,*], Ping-Yu Hsu [a]

[a] *Department of Business Administration, National Central University, No. 300, Jhongda Road, Jhongli City, Taoyuan County 32001, Taiwan, ROC*

[b] *Department of Information Management, Lunghwa University of Science and Technology, Gueishan Shiang, Taoyuan County 333, Taiwan, ROC*

Received 20 December 2005; received in revised form 21 February 2007; accepted 16 March 2007

Abstract

Many IT practitioners and researchers advocate that data models of data warehouses should incorporate the sources of their data in order to achieve maximum efficiency. As the source data are probably derived from system designed with ER diagrams, a great deal of research has been devoted to the design of methodologies for building multidimensional models based on source ER diagrams. However, to the best of our knowledge, no algorithm has been proposed that can systematically translate an entire ER diagram into a multidimensional model with hierarchical snowflake structures. In this paper, we propose an algorithm that achieves the above goal because it incorporates two features, namely, grain preservation and the minimal distance from each dimension table to the fact table. The grain preservation feature guarantees that the translated multidimensional model will maintain cohesive granularity among the entities. Meanwhile, the minimal distance feature guarantees that if an entity can be connected to the fact table in the multidimensional model by more than one path, the path with the smallest number of hops will always be chosen. The first feature is derived by translating ambiguous relationships between entities into weighting factors stored in bridge tables and enhancing fact tables with unique primary keys. The second feature results from including a revised shortest path algorithm in the translating algorithm, with the distance being calculated as the number of relationships required between entities. A prototype system based on the methodology is also developed, and snapshots of the screens used for the system's execution are presented.

Keywords: Data warehouse; Entity relationship diagram; Grain preservation; Multidimensional models; Star schema

1. Introduction

As an increasing number of enterprises experience a rapid growth in the amount of data stored in their operational systems, many consider implementing enterprise-wide data automation software to organize the

* Corresponding author. Address: Department of Business Administration, National Central University, No. 300, Jhongda Road, Jhongli City, Taoyuan County 32001, Taiwan, ROC. Tel.: +886 2 82093211x6326.
E-mail addresses: ytchen@mail.lhu.edu.tw (Y.-T. Chen), pyhsu@mgt.ncu.edu.tw (P.-Y. Hsu).

0020-0255/$ - see front matter © 2007 Elsevier Inc. All rights reserved.
doi:10.1016/j.ins.2007.03.017

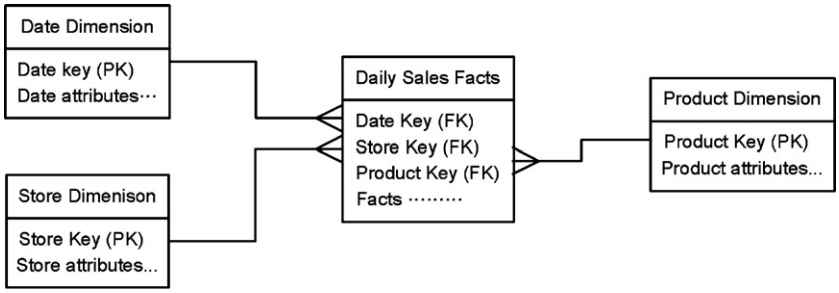

Fig. 1. A sample multidimensional model; source [19].

data and facilitate decision-making. The implementation of data warehousing software by organizations is expected to grow rapidly in the next decade [29,34,25]. When correctly implemented, a data warehouse (DW) system enables companies to reap many benefits and obtain timely information for decision-making [15].

In DW projects, data are collected, cleaned, integrated, and organized into special data models designed for quick access and easy comprehension by decision-makers [17,18]. Because of the importance of such applications, a great deal of research has been devoted to studying appropriate data presentation models [1,2,4,5,7,12–14,16,18,19,23,24,26–28,30,32]. At present, the widely used DW model is the multidimensional model[1], which comprises a fact table and a set of dimension tables connected to the fact table around the periphery. The fact table stores the measures of the performance indicators that are of interest to managers, while the dimension tables provide viewpoints or entry points for accessing the data. Fig. 1, taken from [19], shows a sample multidimensional model with a fact table and three dimension tables.

One of the most important tasks when designing a multidimensional model involves deciding the granularity of the model. The grain is the level of detail at which measurements or events are stored [19,30]. To return cohesive querying results, a multidimensional model design must have a consistent grain, as noted by Inmon and Kimball [17,19]. After a model's grain has been decided, all measures in the fact table and all dimension tables must adhere to this grain; otherwise, serious mistakes can easily occur when query programs are being written.

The following scenario is an example of a grain mismatch. In a typical supermarket visit, a customer buys several products in a single transaction and the dollar amount of each product is aggregated into a total. If a multidimensional model is used to record the total dollar amount of each transaction, the grain is on the level of the transaction (shown in Fig. 2). If products are also stored as a dimension in the model, then querying the transaction amounts in the product dimension will return figures that not only include the cost of a particular product, but also the cost of other products purchased in the same transaction. This difference is due to a grain mismatch between the fact table and the dimension table. The grain of the fact table is on the transaction level, which may include more than one product. It can be seen from this example that designing a model with coherent facts and dimensions is vital in designing multidimensional models.

Moreover, data in a data warehouse are derived from a variety of systems whose data are probable modeled by ER diagrams [8]. In practice, model designers must study the ER diagrams of the source systems, decide the granularity of the target models, and design a multidimensional model manually. Since ER models can be quite complicated, the definition of the grain for each multidimensional model can be subtle; and to make matters worse, the models may be designed by more than one person. Although grain mismatches between facts and related dimension tables are common, errors may not be detected until the query programs are completed and the results are shown to business users. Debugging the errors may result in program redesigns, project delays, and provoke user frustration and distrust of the generated reports. A systematic approach that can automate the design processes is thus long overdue. Automatic processes would greatly reduce designers' efforts and, more importantly, reduce errors due to grain mismatches.

[1] The multidimensional model is also referred to as a dimensional model in some studies.

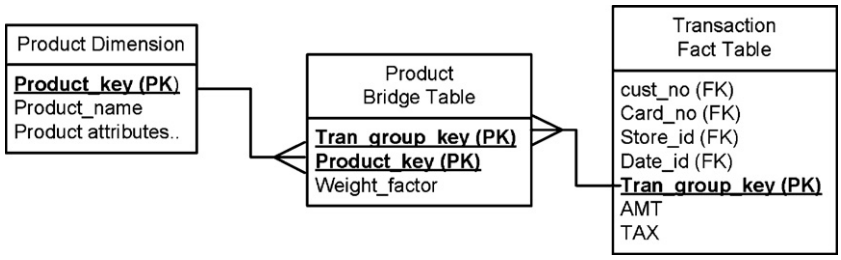

Fig. 2. The many-to-many relationship between the fact and dimension tables.

Given such potentially substantial benefits, many IT practitioners and researchers have striven to develop methodologies for designing multidimensional models from operational systems. Golfarelli and Rizzi [14] considered that the techniques required for designing a data warehouse are completely different from those used when designing operational systems. They also noted that there is no complete and consistent design methodology available for designing a data warehouse. Therefore, they outlined several key steps and crucial issues that should be considered when designing a data warehouse. Dori et al. [11] provided a survey of techniques for transforming an operational system model into a data warehouse model, and classified the construction techniques into two categories. In the first category, structural (data) models, such as ERD and XML, are used as source systems to build corresponding data warehouse models [4,7,13,14,21,23,24,28,30,32]. In the second, the data warehouse model is derived from business processes [10,3,20]. The former method has attracted much more attention than the latter, so we only review research in the first category. Moody and Kortink [24] proposed a three-step method that derives multidimensional models from entity relational models, namely, entity classification, identification of hierarchies, and the production of dimensional models. They also described five optional schemas as outcomes, ranging from a simple flat schema to a complex snowflake schema. Song et al. [30] presented five methods for handling specific issues of many-to-many relationships that might arise while transforming an ER diagram into a dimensional model. Tryfona et al. [32] proposed the starER model, which can handle richer semantics than the traditional multidimensional model when recording many-to-many relationships between a fact table and dimension tables. Bonifati et al. [5] proposed a three-step method for designing a data mart. They used top–down requirement analysis to elicit and consolidate user-requirements, bottom–up data model extraction to form candidate models, and consolidation to derive ideal data models. Cabibbo and Torlone [7] proposed a method that constructs a multidimensional schema from an underlying operational database. The schema consists of a finite set of dimensions, a finite set of fact tables, and a finite set of level descriptions of the dimensions. Golfarelli et al. [13] suggested using a graphical conceptual model (Dimensional Fact model) for data warehouses and a semi-automated methodology to construct a tree-structured fact schema from an Entity–Relation schema. Marotta et al. [23] provided a set of reasoning rules to trace the mapping between the source's logical schema and the data warehouse's logical schema. Boehnlein et al. [4] proposed the Structured Entity Relationship Model (SERM), which visualizes the dependencies between different types of data objects, while Lechtenboerger and Vossen [22] discussed the normal forms of multidimensional models.

Although the above approaches provide guidelines for building multidimensional models from source ER diagrams, they do not formally propose an automatic algorithm that can preserve the granularity of data during the construction process. In an attempt to fill this research gap, we propose a novel algorithm that can achieve the following:

(a) given a fact table and an ER diagram, the algorithm automatically builds a multidimensional model from the ER diagram,
(b) it generates a multidimensional model that has consistent granularity,
(c) in the case where a dimension table is connected to the fact table by more than one path in the ER diagram, the algorithm chooses the path with the smallest number of relationships to minimize the number of join operations needed to query measures via a dimension table.

The remainder of the paper is organized as follows. Section 2 formally defines the concept of grain preservation. Section 3 describes the algorithm used to translate an ER into a multidimensional model. Section 4 explains how the algorithm preserves the grain of a fact table and calculates the minimum distance between tables. Section 5 describes how we implement the proposed algorithm. Finally, in Section 6, we present a summary of the paper and suggestions for future work.

2. Grain preservation

According to Kimball [19], a multidimensional model usually contains a fact table and a set of dimension tables. Each dimension table has a primary key, which is a foreign key for the fact table. The primary key of the fact table is a combination of the foreign keys of all the dimension tables. If there is a grain mismatch between the fact table and any of the dimension tables, the query results may be incorrect [19,30], since erroneous or ambiguous queries return values that cause individual measures to be aggregated more than once. In contrast, a multidimensional model with a consistent grain always aggregates at most one copy of an individual measure, regardless of the dimensions a user adopts to submit a query. Before formally defining grain preservation, we define two operators, namely, $\sum$ and ★.

Given a table, T, in which attributes $a_1, \ldots, a_m$ are measures and attributes $a_{m+1}, \ldots, a_n$ are weight factors, where $m \leqslant n$,

- $S_{a_k}(T) = \sum_{t \in T} t \cdot a_k$ if $1 \leqslant k \leqslant m$, where a is attribute value of t,
- $S(T) = \langle S_{a_1}(T), \ldots, S_{a_m}(T) \rangle$,
- $\bigstar(T) = \{t' | \forall a \in$ attributes of $T, t \in T, (a \in \{a_1, \ldots, a_m\} \rightarrow t' \cdot a = t.a * t \cdot a_{m+1} * \ldots * t \cdot a_n) \wedge (a \notin \{a_1, \ldots, a_m\} \rightarrow t' \cdot a = t \cdot a)\}$.

The ★ operator is used to refine a measure to a finer grain. Table 1 shows an example where the customer# is a non-measured attribute, the amount and the cost are measures, and weight_factor$_1$ and weight_factor$_2$ are weight factor attributes.

Table 2 shows the results of applying the ☆ operator to Table 1.

In a multidimensional snowflake model, a new dimensional table can be connected to the fact table directly, or connected as a table in a snowflake hierarchy. The latter is composed of a set of tables connected like a tree, with the fact table as the root. A table added to a multidimensional model without breaking the existing grain provides an entry point to correctly summarize the measures.

Table 1
A sample table with measures and weight factors

Customer#	Amount	Cost	Weight_factor$_1$	Weight_factor$_2$
c125	50	30	0.4	0.2
c125	50	30	0.6	0.8
c125	50	30	0.4	0.8
c125	50	30	0.6	0.2
c127	40	30	1.0	1.0

Table 2
Applying ★ to Table 1

Customer#	Amount	Cost	Weight_factor$_1$	Weight_factor$_2$
c125	4	2.4	0.4	0.2
c125	24	14.4	0.6	0.8
c125	16	9.6	0.4	0.8
c125	6	3.6	0.6	0.2
c127	40	30	1.0	1.0

Given a multidimensional model with a fact table, F, and a new table T. When T is added to the model, it generates a path of $F, D_1, \ldots, D_k, T$ that connects T to F, and $S(\bigstar(F \bowtie D_1 \ldots \bowtie D_k \bowtie T)) = S(F)$, where $\bowtie$ is a natural join operator. This type of addition is called *Grain Preservation*.

Example 1. Let F be a fact table in which the summary of the purchase amount and the cost of the items are measures. In addition, let M be a store dimension and let D be the number of departments in each store. This is a snowflake table of the store dimension. Let B_1 and B_2 be the bridge tables connecting F and M, and M and D, respectively. The details of this example are listed in Tables 3–7.

The joined result of $F \bowtie B_1 \bowtie M \bowtie B_2 \bowtie D$ is shown in Table 1. The results of applying operator ★ to Table 1 are shown in Table 2. Note that the sum of the measure columns is equal to those in F. Since $S(F) = S(\bigstar F \bowtie B_1 \bowtie M \bowtie B_2 \bowtie D)$, D is added to the multidimensional model and the model's grain is preserved.

Table 3
F: Summary of customer purchase

Customer#	Amount	Cost
c125	50	30
c127	40	30

Table 4
M: Mall

Store#	Address
M1	123 Main Street
M2	456 High Street
M3	890 Low Street

Table 5
D: Departments

Department#	Description
D3	Clothing
D4	Hardware

Table 6
B_1: Bridge Table 1

Customer#	Store#	Weight_factor$_1$
c125	M1	0.4
c125	M2	0.6
c127	M3	1

Table 7
B_2: Bridge Table 2

Store#	Department#	Weight_factor$_2$
M1	D3	0.2
M1	D4	0.8
M2	D3	0.2
M2	D4	0.8
M3	D3	1

3. The mapping algorithm

In this section, we discuss the relationship translation algorithm, which translates the entities and relationships in an ER diagram into multidimensional model tables, while using the smallest number of join operators to maintain the grain preservation.

In the ER diagram of a source system, some entities are connected to each other via many-to-many relationships. The cumulative numeric attributes are candidates for inclusion in the fact tables [18]. We assume that a table in the source ER diagram is designated as the fact table, which contains several fact attributes (also known as measures).

Given a source ER diagram with $\langle E, R\rangle$, where E is the set of entities and R is the partial functions of relationships in the ER diagram, then $R : E \times E \rightarrow \{$'1–1', '1–*M*', '*M*–1', '*M*–*N*'$\}$, where '1–1', '1–*M*', '*M*–1' and '*M*–*N*' denote the cardinality of the relationships.

A multidimensional model is represented by $\langle DE, DR\rangle$, where ***DE*** denotes the set of tables in the model, and *DR* denotes the partial function of the relationships between the tables. Every entity in ***E*** or ***DE*** is assumed to have a primary key.

3.1. Naive mapping rules

Naive mapping rules are used to analyze the relationships between the entities in the source (original) ER diagram, and to translate the corresponding entities and relationships into a multidimensional model. The translation of '1–*M*' and '*M*-to-*N*' relationships may cause grain mismatching if it is not handled carefully. As noted in [18,19,30], a mismatch can be corrected by either relaxing the grain of the fact tables or the grain of the dimension tables. In the case of a snowflake multidimensional model, we argue that lowering (relaxing) the grain of the dimension tables would be preferable, since the same methodology could be used to relax the grain of tables in the snowflake hierarchy.

Given a numeric attribute w in a table, B, which has two and only two foreign keys, f_1 and f_2, let f_1 be the primary key to the table closer to the fact table. B is termed a Sound Bridge Table if the entries in f_1 and f_2 cover all entries in the original tables, the composite value of f_1 and f_2 does not have duplicates in B, and $\forall v \in f_1, t \in B$, $\sum_{t.f_1=v} t.w = 1$.

If $R(E_i, E_j)$ exists and $E_i \in DE$, then E_j is added to ***DE*** in one of the following ways:

- *Rule#1*: $R(E_i, E_j) =$ '*M*-to-1'
  In this case, the translation is straightforward.
  $$DE = DE \bigcup \{E_j\}$$
  $$DR(E_i, E_j) = R(E_i, E_j)$$
  Fig. 3 shows an example of such a case.

- *Rule#2*: $R(E_i, E_j) =$ '1-to-*M*'
  Since the grain of E_j is finer than the grain of E_i, a weight factor is added to E_j to tune the grain of E_j. The corresponding Sound Bridge Table is reduced to a single attribute in E_j:
  $$E'_j = E_j + weight\_factor$$
  $$DE = DE \bigcup \{E'_j\}$$
  $$DR(E_i, E'_j) = R(E_i, E_j)$$
  Fig. 4 shows an example of '1–*M*' translation.

- *Rule#3*: $R(E_i, E_j) =$ '*M*-to-*N*'
  Since the grains of the two tables are incompatible, a Sound Bridge Table, ***B***, is added to tune the grains. The table has two foreign keys, derived from E_j and E_i, respectively, and a weight factor [18,30]. The foreign key from E_i groups entries in E_j so that the combinations have the same grains as the corresponding entries

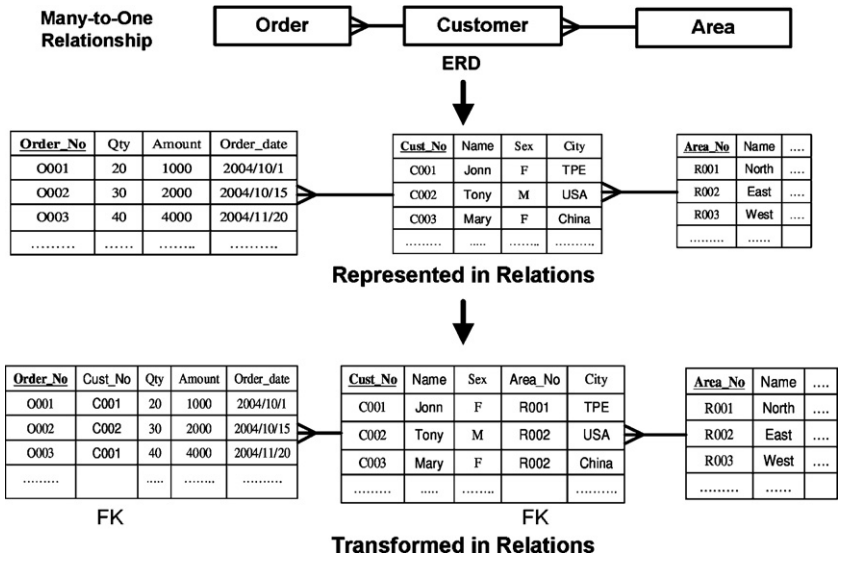

Fig. 3. Translation of a 'Many-to-One' relationship.

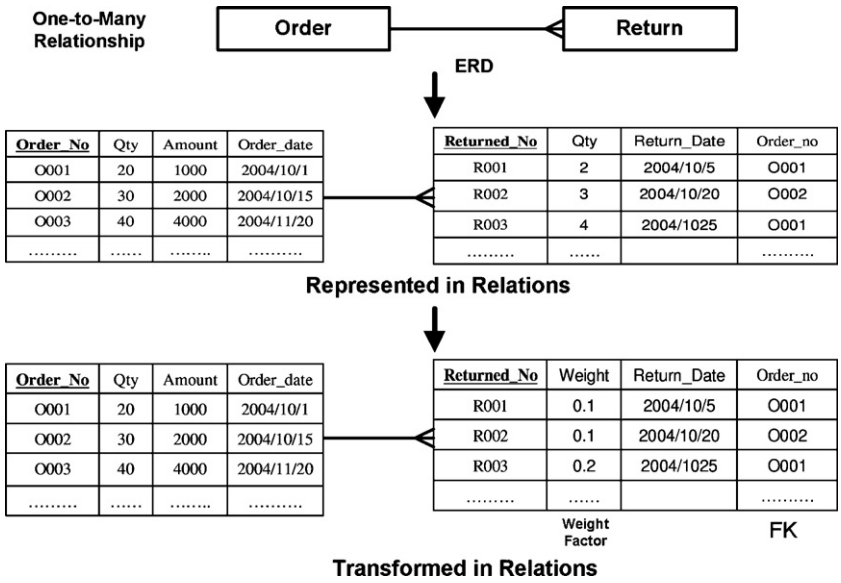

Fig. 4. Translation of a 'One-to-Many' relationship.

in the E_i. The weight factor records the contribution of each entry in the group. The sum of the weight factors in each group should be equal to one:

$$B = \langle weight\_factor \rangle$$
$$DE = DE \bigcup \{E_j, B\}$$
$$DR(E_i, B) = \text{'1-to-}M\text{'}$$
$$DR(B, E_j) = \text{'}M\text{-to-1'}$$

Fig. 5 shows such an example. The algorithm is shown in Fig. 6.

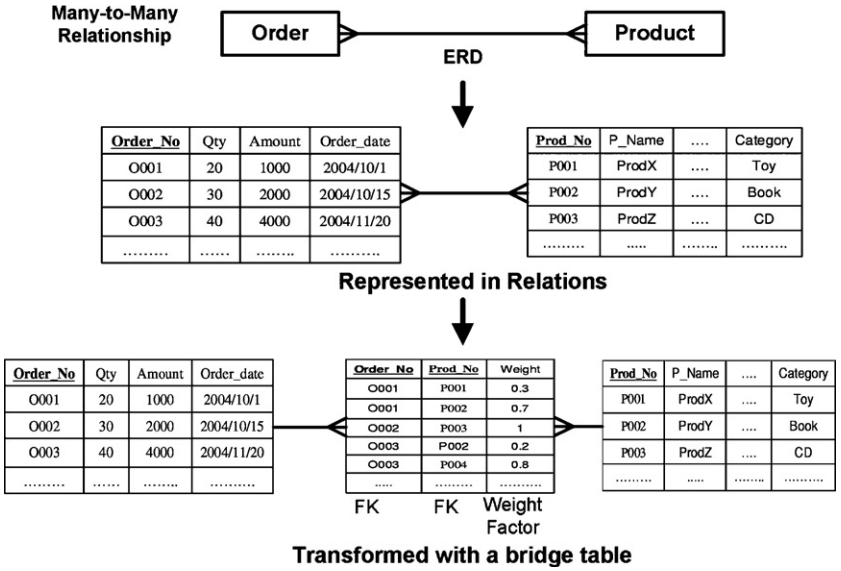

Fig. 5. Translation of a 'Many-to-Many' relationship.

Algorithm: MR
Input: F //the fact table
$\langle \boldsymbol{E}, \boldsymbol{R} \rangle$ //the original ERD
Output: $\langle \boldsymbol{DE}, \boldsymbol{DR} \rangle$ //the desired multidimensional model

Begin
$P = \text{Shortest\_distance}(\langle F, \boldsymbol{E}, \boldsymbol{R} \rangle)$
$\boldsymbol{DE} = \{F\}$
For each $E_j \in \boldsymbol{E} - \boldsymbol{DE} \wedge \exists E_p \in \boldsymbol{DE}, P(E_i) = E)$ do {
case $\boldsymbol{R}(E_i, E_j)$ {
when (1,1) then apply rule#1
when (M,1) then apply rule#1
when (1,M) then apply rule#2
when (M,N) then apply rule#3
}
$\boldsymbol{DE} = \boldsymbol{DE} \bigcup \{\boldsymbol{E}_j\}$
}
Return $\langle \boldsymbol{DE}, \boldsymbol{DR} \rangle$
End;

Fig. 6. The main mapping rules algorithm.

3.2. Applying the mapping rules to an ER diagram

In this section, the process of building an entire multidimensional model from a source ER diagram is explained in detail. The proposed algorithm identifies the fact table in the original ER diagram, and tries to construct a multidimensional model using the smallest number of relationships.

Since an ER diagram can be translated into more than one multidimensional model, the most important decision is the selection of the relationships that will form the translated multidimensional model. The point is illustrated by the following example.

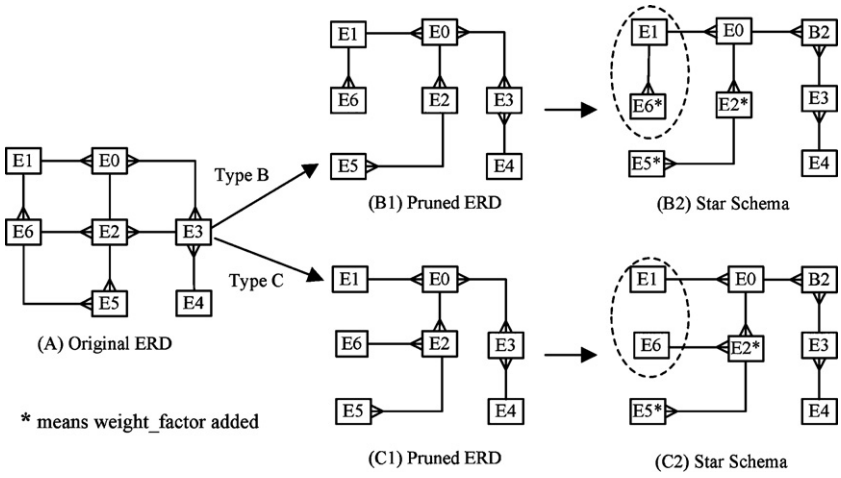

Fig. 7. An example of the same ERD translated into different multidimensional models.

Example 2. The original ER diagram, shown in Fig. 7A, includes several loop relationships.

A loop between snowflake tables causes problems when the data is queried because of the aggregating measures. A loop means that more than one path joins a particular table in the loop to the fact table. Different paths may cause measures to be aggregated differently, which may cause confusion. Thus, the loop in the ER diagram has to be broken when the ER diagram is translated into a multidimensional model.

A loop can be broken in several ways. For example, the loop relationship in the original ER diagram shown in Fig. 7A can be pruned to two ERDs (in Fig. 7B1,C1). Next, different multidimensional models (see Fig. 7B2,C2) are generated by applying the naive mapping rules outlined in Section 3.1. The differences between the two figures are highlighted by dotted lines.

Hence, there is more than one way to build a multidimensional model from an ER diagram. In this paper, we use a single source shortest path algorithm to construct a multidimensional model, in which each entity follows the shortest path to the fact table. The distances between entities and the fact table are calculated by the number of relationships between the entities, since the greater the number of relationships, the greater the number of join operations needed to submit queries. Note that the distance can be determined in many different ways, such as the number of tuples in each join, without jeopardizing the integrity of the algorithm.

To compute the shortest distances between the fact table and the other tables, an initial distance matrix must be built. The matrix is formed by scanning the entire ER diagram. For each entity pair connected by relationships other than '*M*-to-*N*', an initial value of 1 is assigned. For '*M*-to-*N*' relationships, an initial value of 2 is assigned to the corresponding entities, since a bridging table will be needed in the translation.

After deriving the initial matrix, the algorithm calculates the shortest path vector and the predecessor matrix. The former records the shortest path between the fact table and the corresponding entity, while the latter records other entities connected to the corresponding entity along the path. The algorithm is a revision of the famous Dijkstra algorithm [9].

Zero is inserted into the initial distance matrix to indicate the distance of an entity to itself. The assumption may not be applicable to ER diagrams in which entities are allowed to have self-reference relationships. However, we assume that the translated multidimensional model is free of such relationships. This assumption is based on the widely adopted practice whereby most self-reference relationships are flattened into their corresponding entities by the hierarchy recorded in the attributes. The algorithm for computing the shortest distances is shown in Fig. 8.

The performance cost is determined by the *Shortest_Distance* algorithm, which calculates the most cost-effective means of connecting dimension tables to the fact table in terms of the number of relationship traversals required. The complexity of the Shortest_Distance procedure is $O(|E|^3)$, where $|E|$ is the number of entities in the source ER diagram. The cube comes from the *While Loop* in the procedure, which examines an added

```
Algorithm: Shortest_Distance
Input:   F          //the fact table
         ⟨E, R⟩     //the original ERD
Output:  P          //The proposed predecessor of each entity
Begin
    Fill L_0 with ∞
    //Scan the original ERD and assign an initial value
    //to each relationship
    for each defined R(E_i, E_j) do {
       if R(E_i, E_j) = 'M-to-1' then L_0(E_i, E_j) = 1
       elseif R(E_i, E_j) = '1-to-1' then L_0(E_i, E_j) = 1
       elseif R(E_i, E_j) = '1-to-M' then L_0(E_i, E_j) = 1
       elseif R(E_i, E_j) = 'M-to-N' then L_0(E_i, E_j) = 2
    }
    //Compute the shortest path vector and the shortest paths
    // to the fact table
    for each E in E do {
        L(E) = ∞
    }
    L(F) = 0
    Q = E
    While Q ≠ ∅ do {
          E_i = ARG_Q(min_{E_k∈Q}(L(E_k)))
          Q = Q − {E_i}
          For each defined R(E_i, E_j) do {
              if L(E_j) > L(E_i) + L_0(E_i, E_j) then
                 L(E_j) = L(E_i) + L_0(E_i, E_j)
                 P[E_j] = E_i
          }
    }
    Return P
End;
```

Fig. 8. Finding the shortest distances between entities and the fact table F.

entity and its associated relationships to derive the shortest distances between all entities. The loop terminates when all entities have been examined.

4. Correctness

We now prove that, from an ER diagram of $\langle E, R\rangle$ with a designated fact table, the MR algorithm generates a $\langle DE, DR\rangle$ that satisfies the *Grain Preservation* requirement. In addition, we show that all dimension tables are connected to the fact table by the smallest number of relationships.

Theorem 3. *Given an ER diagram and a fact table in it, the multidimensional model constructed by the MR algorithm connects each entity to the fact table with the smallest number of relationships.*

Rationale. *The predecessor of each entity in the multidimensional model is decided by Dijkstra's algorithm* [9], *which discovers the shortest path to the fact table. The weights on the edges represent the number of relationships needed by the multidimensional model to translate the corresponding relationships in the ER diagram. Therefore, MR returns a multidimensional model with the smallest number of relationships between each dimension table and the fact table.*

Let F be the designated fact entity in the ER diagram of $\langle E, R\rangle$, and let E denote the entity added to the multidimensional model $\langle DE, DR\rangle$ to form $\langle DE', DR'\rangle$.

Lemma 4. *If $\langle DE, DR\rangle$ is a grain preserved multidimensional model in which F is the fact table, and E is connected to F in the original $\langle ER\rangle$ (but it is not yet included in the multidimensional model), then connecting E to F by the proposed rules will preserve the grain of the model.*

Proof. In $\langle DE, DR\rangle$, F and E can be connected by any of the three types of relationships, namely, '1-to-M', 'M-to-1' and 'M-to-N' while keeping grain preserved with MR's corresponding translations.

'M-to-1':
According to the rule, the entity is connected to F directly in $\langle DE', DR'\rangle$. Since no bridge table is added between F and E, and every entry in F is matched to exactly one record in E,

$$\begin{aligned} S(\bigstar(\mathrm{F} \bowtie \mathrm{E})) &= S(\mathrm{F} \bowtie \mathrm{E}) \\ &= S(F) \end{aligned}$$

'1-to-M':
In this case, a weight factor from a Sound Bridge Table is added to E, and the new E′ is connected to F by the original relationship.

$$\begin{aligned} &S(\bigstar(\mathrm{F} \bowtie \mathrm{E}')) = S(\bigstar\{u+v | \forall u \in Fv \in \mathrm{E}'\ u \cdot primary\_key = v \cdot foreign\_key\}) \\ &= S(\{\langle u \cdot a'_1 \ldots u \cdot a'_m\rangle | \forall\ measures\ a_i\ inF,\ \forall u \in Fv \in E' \\ &\quad u \cdot primary\_key = v \cdot foreign\_key \wedge u \cdot a'_i = u \cdot a_i * v \cdot weight\_factor\}) \\ &= S(\mathrm{F}) \end{aligned}$$

'M-to-N':
In MR, a Sound Bridge Table, B, is inserted between F and E with two '1-to-M' relationships originating from F and E.

$$\begin{aligned} S(\bigstar(\mathrm{F} \bowtie B \bowtie \mathrm{E})) &= S(\bigstar\{u+w+v | \forall u \in \mathrm{F} w \in Bv \in \mathrm{E} \\ &\quad u \cdot primary\_key = w \cdot foreign\_key_1 \wedge \\ &\quad w \cdot foreign\_key_2 = v \cdot primary\_key\}) \\ &= S(\{\langle u \cdot a'_1 \ldots u \cdot a'_m\rangle | \forall\ measures\ a_i\ in\ \mathrm{F},\ \forall u \in \mathrm{F} \\ &\quad w \in Bv \in \mathrm{E} u \cdot primary\_key = w \cdot foreign\_key_1 \wedge \\ &\quad w \cdot foreign\_key_2 = v \cdot primary\_key \wedge \\ &\quad u \cdot a'_i = u \cdot a_i * w \cdot weight\_factor\}) \\ &= S(\mathrm{F}) \end{aligned}$$

Therefore, the translation done by MR is *Grain Preservation* when dimensions are connected to the fact table directly. □

Theorem 5. *Given an ER diagram, $\langle E, R\rangle$, a multidimensional model, $\langle DE, DR\rangle$, is derived from part of the diagram using MR. An entity E in the ER diagram (E has not been included in the multidimensional model yet) is added to $\langle DE, DR\rangle$ using MR to preserve the grain.*

Proof. When E is connected to more than one entity in the ER diagram, the Shortest_Distance algorithm is applied to decide which relationship should be retained. Therefore, the proof proceeds with the assumption that E is connected to only one entity in the multidimensional model. The proof is derived by mathematical induction with the parameter ranges over the number of dimension tables positioned between the fact table F and E in $\langle DE, DR\rangle$.

Induction base ($N = 0$)

In the case where entity is linked to the fact table F directly, Lemma 4 proves that the MR approach can preserve the grain.

Induction hypothesis ($N = k$)

Assuming that MR's translation preserves grain when there are k entities between the fact table and E.

Induction Step. We assume that E is connected to D_k in $\langle DE, DR\rangle$, D_k is connected to the fact table though $(k-1)$ dimension tables $D_1 \ldots, D_{k-1}$, F has measures $a_1, \ldots, a_m$, and $T = \mathrm{F} \bowtie D_1 \bowtie \ldots \bowtie D_k$. The proof shows that, even when an entry in T matches more than one record in E, the value of the S operation of the joined result is the same as the original value. The relationship between D_k and E is again one of the three types: '1-to-M', 'M-to-1', and 'M-to-N'. The proof shows that MR can preserve the grain in all of the following cases.

'M-to-1':

Since each record in T is mapped to exactly one entry in E, $\bigstar(T \bowtie \mathrm{E}) = \bigstar(T)$.

'1-to-M':

For each $a_i \in \{a_1, \ldots, a_m\}$,

$$\begin{aligned} S_{a_i}(\bigstar(T \bowtie \mathrm{E})) &= \sum\{a_i' | \forall t \in \bigstar(T), e \in E \\ &\quad e \cdot foreign\_key \in t \rightarrow a_i' = t \cdot a_i * e \cdot weight\_factor\} \\ &= \sum\{a_i' | \forall t \in \bigstar(T), a_i' = t \cdot a_i\} \\ &= S_{a_i}(\bigstar(T)) \\ &= S_{a_i}(\mathrm{F}) \end{aligned}$$

'M-to-N':

Let B be the Sound Bridge Table connecting D_k and E. For each $a_i \in \{a_1, \ldots, a_m\}$,

$$\begin{aligned} S_{a_i}(\bigstar(T \bowtie B \bowtie \mathrm{E})) &= \sum\{a_i' | \forall t \in \bigstar(T), b \in B, e \in \mathrm{E}\ b \cdot foreign\_key_1 \in t \wedge \\ &\quad b \cdot foreign\_key_2 \in e \rightarrow a_i' = t \cdot a_i * b \cdot weight\_factor\} \\ &= \sum\{a_i' | \forall t \in \bigstar(T), a_i' = t \cdot a_i\} = S_{a_i}(\mathrm{F}) \end{aligned}$$

Thus, E can be safely added to the multidimensional model $\langle DE, DR\rangle$ by MR. □

5. Implementation

We have developed a prototype of the proposed system. Since the program is coded in Java, it can be run on most personal computers. The program provides an interface to input an ER diagram, and a function bar

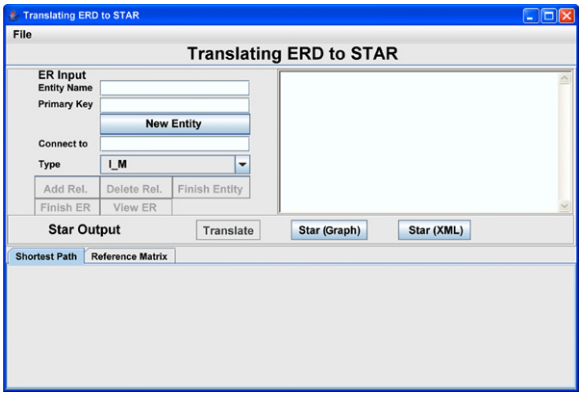

Fig. 9. The main interface of the translation program.

to calculate the distances from the entities to the fact table and trigger the translation. The main screen, shown in Fig. 9, has four major functional areas.

(a) In the top-left part of the screen, the entity and the relationships of an ER diagram can be entered; the first entity is the fact table.
(b) There is a Message window in the top-right.

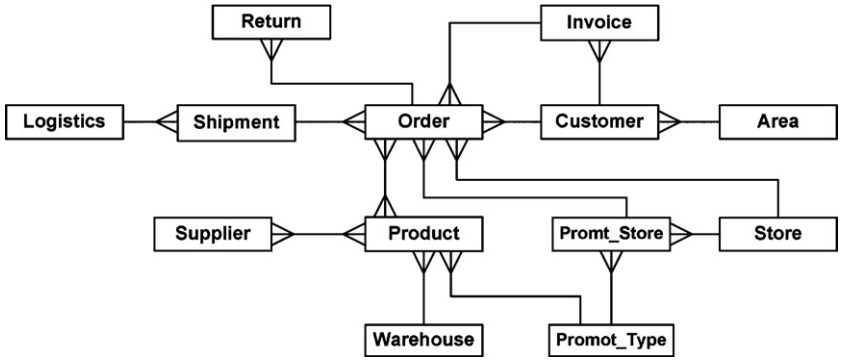

Fig. 10. An original ER diagram.

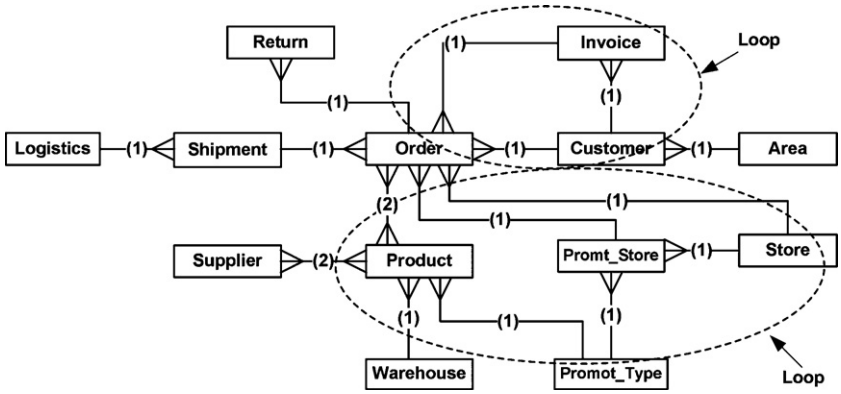

Fig. 11. Original ERD marked with the initial number of levels.

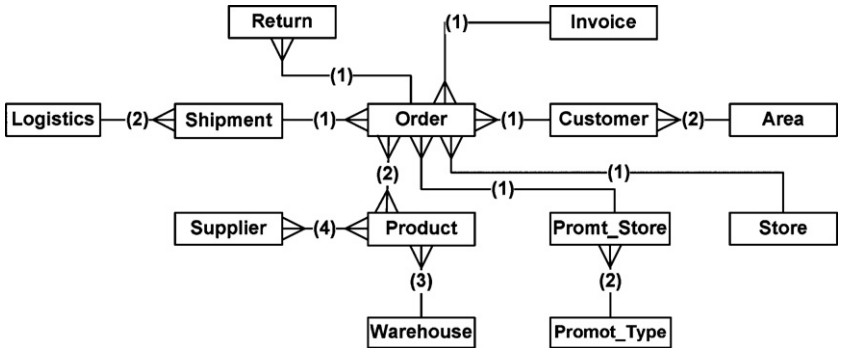

Fig. 12. Pruned ER diagram showing the shortest paths.

(c) A function bar in the middle contains the TRANSLATE option, which calculates the shortest distances. There are also two STAR options, which review the multidimensional model output in Graph and XML Format, respectively.
(d) At the bottom of the screen, two matrix windows show the Shortest Path and the Reference Matrix between the fact table and all dimensions.

Next, we describe the translation process using an ER diagram. The original ER diagram, shown in Fig. 10, is taken from a commercial sales order tracking system. The entity 'Order' is the designated fact table, so the

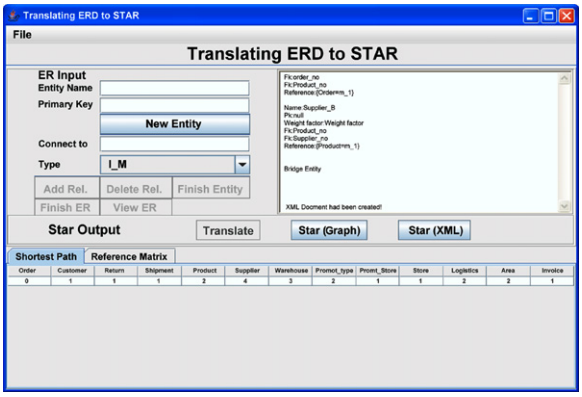

Fig. 13. A snapshot of the translation program's execution.

```
<?xml version="1.0" encoding="UTF-8"?>
<ER>
<Entity name="Order">
      <number>0</number>
      <pk>order_no</pk>
      <relationship>
            <ref-to type="m_1">Invoice</ref-to>
            <ref-to type="1_m">Return</ref-to>
            <ref-to type="m_1">Customer</ref-to>
            <ref-to type="m_n">Product</ref-to>
            <ref-to type="m_1">Promt_Store</ref-to>
            <ref-to type="m_1">Store</ref-to>
            <ref-to type="m_1">Shipment</ref-to>
      </relationship>
</Entity>
<Entity name="Customer">
      <number>1</number>
      <pk>customer_no</pk>
      <relationship>
            <ref-to type="1_m">Invoice</ref-to>
            <ref-to type="m_1">Area</ref-to>
            <ref-to type="1_m">Order</ref-to>
      </relationship>
</Entity>
      ......
</ER>
```

Fig. 14. Part of the input ER diagram in XML format.

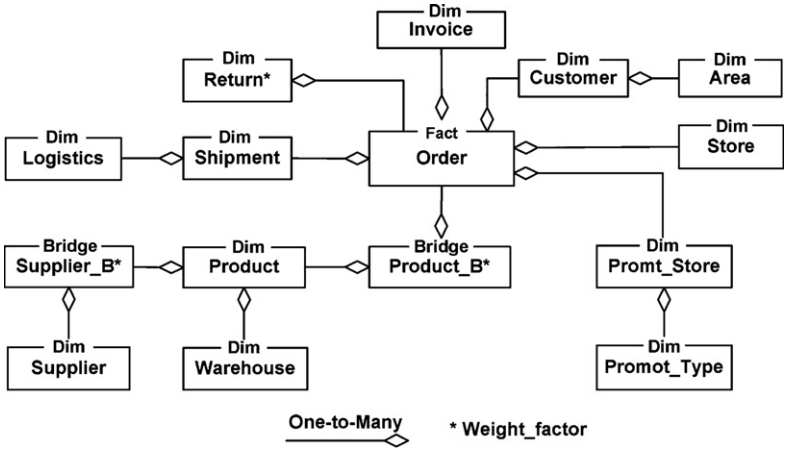

Fig. 15. The output graph generated by the translation program.

```
<?xml version="1.0" encoding="UTF-8"?>
<Star>
<FactTable name="Order">
        <pk>order_no</pk>
        <fk>customer_no</fk>
        <fk>shipment_no</fk>
        <fk>promt_store_no</fk>
        <fk>store_no</fk>
        <fk>invoice_no</fk>
        <att>measures...</att>
        <relationship>
              <ref-to type="m_1">Store</ref-to>
              <ref-to type="m_1">Promt_Store</ref-to>
              <ref-to type="m_1">Shipment</ref-to>
              <ref-to type="m_1">Invoice</ref-to>
              <ref-to type="m_1">Customer</ref-to>
              <ref-to type="1_m">Return</ref-to>
        </relationship>
</FactTable>
<Dimension name="Customer">
        <pk>customer_no</pk>
        <fk>area_no</fk>
        <att>attributes...</att>
        <relationship>
              <ref-to type="1_m">Order</ref-to>
        </relationship>
</Dimension>
        ......
        ......
</Star>
```

Fig. 16. Part of the output in XML.

other entities are treated as dimension tables for the translation process. The annotated edges are the relationships needed to complete the MR translation (see Fig. 11). The shortest path from the fact table to each entity is shown by the ER diagram in Fig. 12.

The sample ER is input via the interface. Fig. 13 shows a snapshot of the program's execution. The ER can be viewed in XML format, as shown in Fig. 14. The initial distance matrixes of the ER diagram and the shortest path vector are shown in the bottom window of Fig. 13. The loops in the keyed-in ER diagram can be broken by removing any edge whose distance from the fact table is longer than the preserved paths. The acyclic ERD is shown in Fig. 12.

The output of the sample case generated by the program is shown in Fig. 15; it can also be generated in XML format, as shown in Fig. 16.

6. Conclusions

As enterprises place a high priority on real time management, data warehouse systems have become critical tools of information analysis. However, most companies lack experienced data warehouse professionals who can design effective multidimensional models. In contrast, professionals with experience in ER concepts are more widely available. Hence, deriving a data warehouse schema from an ER diagram may be a compromise solution. Even so, it is not easy for an inexperienced team to build a corporate data warehouse in a short time. It is our hope that the techniques and mapping rules presented in this paper will help such individuals.

We have proposed a method for deriving data warehouse schema from ER diagrams. The translation rules provide a very effective tool, while the proposed grain preservation algorithm automatically translates an ER diagram with a fact table into a multidimensional model. The algorithm is very efficient because every dimension table in the proposed hierarchy is connected to the fact table via the smallest number of relationships.

The approach presented in this paper provides a foundation for future research on automatic translation of ER diagrams into multidimensional models. A number of issues still need to be addressed. For example, integrating multiple ER diagrams with heterogenous data types in the original source systems is a practical and difficult issue. A great deal of work has been done on integrating heterogeneous data in ETL (Extraction, Transformation, and Loading) operations [6,31,33]. Further research based on that work and the results published in this paper may lead to fully automatic multidimensional model design processes.

Acknowledgements

The authors acknowledge the financial support provided by the National Science Council of Taiwan, through Project No. NSC93-2416-H-008-010.

References

[1] L. Baekgaard, F. Alle, Event–entity–relationship modelling in data warehouse environment, in: Proceedings of the ACM Second International Workshop on Data Warehousing and OLAP (DOLAP) November 6, ACM, Kansas City, MO, USA, 1999, pp. 9–14.
[2] P.A. Bernstein, E. Rahm, Data warehouse scenarios for model management, in: Proceedings of 19th International Conference on Entity–relationship Modeling, LNCS Lecture Notes in Computer Science, vol. 1920, Springer, Salt Lake City, UT, USA, 2000, pp. 1–15, October.
[3] M. Boehnlein, A.U. Ende, Business process oriented development of data warehouse structures, in: Data Warehousing (DW 2000), Friedrichshafen, Germany, November 2000, pp. 3–21.
[4] M. Boehnlein, A. Lbrich, Deriving initial data warehouse structures from the conceptual data models of the underlying operational information systems, in: Proceedings of the ACM Second International Workshop on Data Warehousing and OLAP (DOLAP), Kansas City, MO, USA, November 1999, pp. 15–21.
[5] A. Bonifati, F. Cattaneo, S. Ceri, A. Fuggetta, S. Paraboschi, Designing data marts for data warehouse, in: ACM Transactions on Software Engineering and Methodology (TOSEM), vol. 10 of 4, October 2001, pp. 452–483.
[6] F. Boufares, S. Hamdoun, Integration techniques to build a data warehouse using heterogeneous data sources. Journal of Computer Science (Special Issue), 48 (2005) 48–55.
[7] L. Cabibbo, R. Torlone, A logical approach to multidimensional databases, in: Proceedings of the Sixth International Conference on Extending Database Technology: Advances in Database Technology, Valencia, Spain, March 1998, pp. 183–197.
[8] P.P.S. Chen, The entity–relationship model – toward a unified view of data, ACM Transactions on Database Systems 1 (1) (1976) 9–36.

[9] E.W. Dijkstra, A note on two problems in connexion with graphs, Numerische Mathematik 1 (1) (1959) 269–271.
[10] Dov Dori, Roman Feldman, Arnon Sturm, An opm-based method for transformation of operational system model to data warehouse model, in: Proceedings of the International Conference on Software – Science, Technology & Engineering (SwSTE'05), IEEE, 2005, pp. 57–66.
[11] Dov Dori, Roman Feldman, Arnon Sturm, Transforming an operational system model to a data warehouse model: A survey of techniques, in: Proceedings of the International Conference on Software – Science, Technology & Engineering (SwSTE'05), IEEE, 2005, pp. 47–56.
[12] E. Franconi, U. Sattler, A data warehouse conceptual data model for multidimensional aggregation, in: Proceedings of the International Workshop on Design and Management of Data Warehouses (DMDW'99), Heidelberg, Germany, 1999, p. 13.
[13] M. Golfarelli, D. Maio, S. Rizzi, Conceptual design of data warehouses from ER schemes, in: Proceedings of the Hawaii International Conference on system Sciences, Kona, Hawaii, January 1998, pp. 334–343.
[14] M. Golfarelli, S. Rizzi, Designing the data warehouse: Key steps and crucial issues, Journal of Computer Science and Information Management 2 (3) (1999) 1–14.
[15] N. Gorla, Features to consider in a data warehousing system, Communications of the ACM 46 (11) (2003) 111–115.
[16] B. Husemann, J. Lechtenborger, G. Vossen, Conceptual data warehouse design, in: Proceedings of the International Workshop on Design and Management of Data Warehouse (DMDW2000), Stockholm, Sweden, Technical University of Aachen (RWTH), 2000, p. 6.
[17] W.H. Inmon, Building the Data Warehouse, third ed., John Wiley & Sons, Inc., New York, 2002.
[18] R. Kimball, L. Reeves, M. Ross, W. Thornthwaite, The Data Warehouse Life Cycle Toolkit, John Wiley & Sons, Inc., New York, 1998.
[19] R. Kimball, M. Ross, The Data Warehouse Toolkit, second ed., John Wiley & Sons, Inc., New York, 2002.
[20] H. Kim, T.H. Lee, S. Lee, J. Chun, Automated data warehousing for rule-based CRM systems, in: Proceedings of 14th Australasian Database Conference (ADC), vol. 17, Adelaide, Australia, 2003, pp. 67–73.
[21] T.M. Krippendorf, I.Y. Song, The translation of star schema into entity–relationship diagrams, in: Proceedings of the Eighth International Conference and Workshop on Database and Expert-systems Applications (DEXA'97), Toulouse, France, September 1997, pp. 390–395.
[22] J. Lechtenboerger, G. Vossen, Multidimensional normal forms for data warehouse design, Information Sciences 28 (5) (2003) 415–434.
[23] A. Marotta, R. Ruggia, Data warehouse design: a schema-transformation approach, in: Proceedings of the XXII International Conference of the Chilean Computer Science Society (SCCC 2002), IEEE-CS, 2002, pp. 153–162, November.
[24] D.L. Moody, M.A.R. Kortink, From enterprise models to dimensional models: A methodology for data warehouse and data mart design, in: Proceedings of the International Workshop on Design and Management of Data Warehouse (DMDW2000), Stockholm, Sweden, Technical University of Aachen (RWTH), June 2000, p. 5.
[25] D. Mukherjee, D. Dsouza, Think phased implementation for successful data warehousing, Information Systems Management 20 (2) (2003) 82–90.
[26] B. Pedersen, C.S. Jensen, Multidimensional data modeling for complex data, in: Proceedings of the 15th International Conference on Data Engineering (ICDE), Sydney, Australia, March 1999, pp. 336–345.
[27] F. Ravat, O. Teste, G. Zurfluh, Towards data warehouse design, in: Proceedings of the ACM 1999 International Conference on Information and Knowledge Management (CIKM), Kansas City, MO, USA, November 1999, pp. 359–366.
[28] C. Sapia, M. Blaschka, G. Hofling, B. Dinter, Extending the e/r model for the multidimensional paradigm, in: Proceedings of the Workshops on Data Warehousing and Data Mining: Advances in Database Technologies, LNCS, vol. 1552, Springer-Verlag, 1998, pp. 105–116.
[29] B. Shin, A case of data warehousing project management, Information and Management 39 (7) (2002) 581–592.
[30] I.Y. Song, C. Medsker, W. Rowen, E. Ewen, An analysis of many-to-many relationships between fact and dimension tables in dimensional modeling, in: Proceedings of the International Workshop on Design and Management of Data Warehouses (DMDW'2001), Interlaken, Switzerland, June, 4 2001, p. 13.
[31] R. Torlone, I. Panella, Design and development of a tool for integrating heterogeneous data warehouses, Lecture Notes in Computer Science 3589 (2005) 105.
[32] N. Tryfona, F. Busborg, J.G.B. Christiansen, Starer: A conceptual model for data warehouse design, in: Proceedings of the ACM Second International Workshop on DataWarehousing and OLAP (DOLAP), Kansas City, MO, USA, November 1999, pp. 3–8.
[33] F.S.C. Tseng, C.-W. Chen, Integrating heterogeneous data warehouses using xml technologies, Journal of Information Science 31 (3) (2005) 209–229.
[34] H.J. Watson, D.L. Goodhue, B.H. Wixom, The benefits of data warehousing: why some organizations realize exceptional payoffs, Information and Management 39 (6) (2002) 491–502.