DATA WAREHOUSING – CONCEPTUAL SCHEME DESIGN AND MAPPING INTO RELATIONAL LOGICAL SCHEME

A. Rozeva

Assoc.Prof., Ph.D., Department of Computer Systems and Informatics, University of Forestry Sofia, Phone: +359 2 91 907 340, E-mail: arozeva@ltu.bg

Abstract: The paper deals with data warehouse design issues concerning conceptual modeling and implementation by generating a logical scheme. Data warehouse system is intended to synthesize information from multiple operational sources making it suitable for direct querying and analysis for the sake of decision making. The synthesis of information and its representation from data warehouse centric perspective which is the multidimensional one is accomplished by implementing the multidimensional conceptual model into a conceptual scheme. Further logical modeling and respective logical scheme generation provides for data warehouse complex querying and analytical processing. Conceptual scheme for an operational retail database has been designed and it's mapping into star scheme at logical level performed. Application issues concerning conceptual scheme elements' and semantics' formalization as well as mapping procedures have been presented.

Keywords: Data warehouse design, Conceptual scheme, Multidimensional model, Star scheme, Scheme generation

INTRODUCTION

Data warehouse systems have been designed and implemented in business organizations for facilitating the maintenance of massive amounts of information. The main topic of data warehouse is to synthesize information from multiple heterogeneous operational data bases and storing it into a single repository with the purpose to ensure complex querying and analysis. Thus data in the warehouse is only accessed for analytical query processing. A warehouse is physically separated from the operational databases, sourced from them and refreshed periodically. The basic issues of data warehouse technical architecture and design implementing multidimensional modeling concepts have been presented in [3]. The multidimensional model has been designed and targeted to the purposes of data warehousing. Its core topic represents the fact with all the data surrounding it. A lot of research work concerning the different phases of data warehouse design has been done some results of which are shown in [2], [5] and [10]. A design methodology for data warehouses is proposed in [1]. The main steps refer to: analysis of source operational databases, requirement specification, conceptual design, conceptual scheme validation, logical design and physical design.

Conceptual modeling provides for representing complex information at the semantic level. It's accomplished considering the source databases and data warehouse's user requirements. Data warehouses have imposed the specific modeling structure fact. The conceptual model therefore should reflect facts and objects that are semantically connected to them. Objects serve as focuses providing for facts' analysis. Some of the conceptual models for data warehouses are the dimensional fact model [1] and the star ER model [9]. Both models accommodate the set of concepts: facts, objects, associations, dimensions, hierarchies, aggregates. Conceptual model's graphical representation is the conceptual scheme. The conceptual scheme of the dimensional fact model is a quasi-tree. The one of the star ER is the classical ER scheme enriched with special types of associations and facts' properties providing for defining hierarchies and aggregations.

The design of a conceptual scheme provides for further logical design and logical scheme generation. Logical scheme design has been treated in [2], [5] and [8]. Having as input the conceptual scheme logical design considers query patterns as well describing the expected queries used for generation of periodical reports in the enterprise. Issues concerning aggregate query patterns and data structures for efficient computation have been treated by the author in [6] and [7]. Logical design is performed on the basis of a chosen target logical model, relational or multidimensional. Relational schemes that are most often implemented at logical design phase are the star and snowflake.

Our work concerns the phases of conceptual design and its mapping to a logical scheme as shown in Fig.1

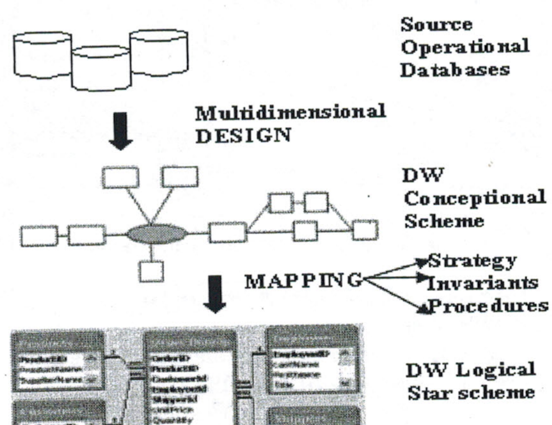

Fig.1. Key steps in data warehouse design process

Further on in the paper a data warehouse conceptual scheme design from a retail source database is presented with formal definition of the basic elements and hierarchies. A design strategy for mapping the conceptual scheme to a logical star scheme providing for basic data warehouse invariants referential integrity and hierarchies is presented. Mapping procedures concerning logical design are highlighted.

CONCEPTUAL SCHEME DESIGN

Conceptual scheme design takes into account source operational database and desired report content. Facts and objects associated to them can be derived from the source database scheme. Requirements concerning report content will be included by hierarchies' definition and query patterns describing aggregations. The source database that has been examined concerns orders and is similar to the Northwind database where some fields in the tables irrelevant to data warehouse design have been omitted. Its logical scheme is shown in Fig.2.

CUSTOMER (<u>CustId</u>, CustName, CustCity, CustRegion, CustCountry)
PRODUCT (<u>ProdId</u>, ProdName, SuppId, CatId, Price)
SUPPLIER (<u>SupId</u>, SupName, SupCity, SupRegion, SupCountry)
CATEGORY (<u>CatId</u>, CatName)
SHIPPER (<u>ShipId</u>, ShipName)
EMPLOYEE (Emp<u>Id</u>, EmpName)
ORDERS (<u>OrdId</u>, CustId, ShipId, Ord_Date, ShipToName, ShipToAdr)
ORDER_DETAILS (<u>OrdId, ProdId</u>, Quantity, Price, Discount)

Fig.2. Orders source database logical scheme

The conceptual model that'll be implemented for designing the warehouse conceptual scheme is the starER [9]. The main issue is the identification of facts, objects, properties and associations among them. Analyzing the logical scheme it turns out that the event generating data over time concerns orders. As orders include several lines of products we define a fact from the individual order line. Besides facts should have at least one quantitative property which is the case with an order line fact. As order lines are to be analyzed from viewpoints of customers, products, shippers, employees and recipients they represent the objects associated with the defined fact. The conceptual scheme designed is shown in Fig.3.

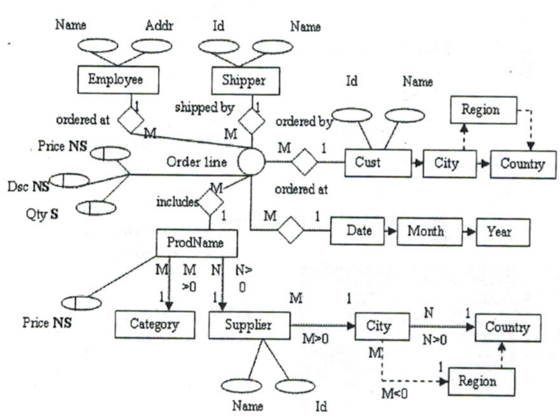

Fig.3. Data warehouse conceptual scheme

The conceptual scheme reflects the data warehouse point of view to the source operational database. It contains a fact 'order line' represented by a circle. Objects are shown with rectangles, objects' properties with ellipses and relationships representing associations with diamonds. Relationships among the fact and the objects are of type many-to-one and are denoted as M:1. Relationships denoted by solid arrows with cardinality and constraint represent an association between objects such that an object is a member of another object with the same characteristics and behaviour. The same cardinalities and constraints hold for all relationships denoted with a solid arrow. This is the association between customer, city and country as well as between date, month and year. Objects month and year have been added to the conceptual scheme in order to provide for useful reports generation. For such a relationship it holds that all members of the one object belong to only one "higher" object, higher in the meaning of including. Besides that all members of an object belong to the higher object which consists of those members only. Relationship shown by a dashed arrow with cardinality and constraint represents an association such that not all members belong to the higher object. This is the relationship between city and region. In the conceptual scheme all such relationships have the same cardinality and constraint. Fact's properties are shown with ellipses crossed by a line. The same sign is used to denote numeric properties of objects such as 'price' for object product. Fact 'order line' has the properties 'qty', 'dsc' and 'price'. Fact properties have a symbol/symbols attached to their name denoting whether they can be summarized in various ways in order to extract further information. This is a very important issue for data warehouses. Symbol 'S' means that the property can be summarized and "NS' that it can't. The property quantity 'qty' can be summarized, while prices and discounts 'dsc' can't. As seen from Fig.3. the conceptual scheme reflects the basics of data warehouse multidimensional model having the fact in the centre and the rest of the data is unfolded around it. Data warehouse conceptual design requires further definition of dimensions, hierarchies and measures. Dimensions are defined by the objects in the scheme that are connected via associations to the fact. They'll be the focuses for data warehouse analysis. The following dimensions can be defined from the conceptual scheme: Customer, Shipper, Employee, Product, TimeOrder. Relationships represented by solid arrows provide for defining hierarchies to dimensions. Dimensions Customer, TimeOrder and Product have hierarchies. Product dimension has two hierarchies. Hierarchies specify different granularities at which the connected fact can be summarized.

FORMALIZATION OF THE CONCEPTUAL SCHEME

Formal definitions of basic scheme's elements fact, some of the dimensions and hierarchies as inspired from [4] are presented further on.

- *Dimension definition*

Dimension type Customer **includes**
    hierarchy Customer
    composed of Cust, City, Country;
  Level type Customer **has**
    Attribute Id **of type integer**,
    Attribute Name **of type string**,
  ChildParent type CustCit **relates** Customer **and** City
  ChildParent type CitCount **relates** City **and** Country
  ChildParent type CustCit **relates** Customer **and** City
  ChildParent type CitReg **relates** City **and** Region **as**
    Hide member if region.**visible**=no
ChildParent type RegCount **relates** Region **and** Country **as**
    Hide member if region.**visible**=no
Dimension type TimeOrder **includes**
    hierarchy Date
    composed of Date, Month, Year
  ChildParent type DatMon **relates** Date **and** Month
  ChildParent MonYear **relates** Month **and** Year

Dimension type Shipper **includes**
level Shipper
Level type Shipper **has**
Attribute Id **of type integer,**
Attribute Name **of type string,**
Dimension type Product **includes**
hierarchy ProdCat
composed of Product, Category
ChildParent ProdCat **relates** Product **and** Category
hierarchy ProdSup
composed of Product, Supplier, City, Country
Level type Supplier **has**
Attribute Id **of type integer,**
Attribute Name **of type string,**
ChildParent type ProdSup **relates** Product **and** Supplier
ChildParent type SupCit **relates** Supplier **and** City
ChildParent type CitCoun **relates** City **and** Country
ChildParent type CitReg **relates** City **and** Region **as**
Hide member if region.**visible**=no
ChildParent type RegCount **relates** Region **and** Country **as**
Hide member if region.**visible**=no

- *Fact relationship definition*

Fact relationship type Order_line **has**
Attribute Qty **of type real,**
Attribute Dsc **of type real,**
Attribute Price **of type real,**
involves
Customer, Shipper, Employee, Product, TimeOrder;

Conceptual scheme's design involves besides the definitions of conceptual scheme elements semantic constraints they are involved in as well. The semantics is defined by semantic functions associating sets of objects with sets of value domains. Semantic function definition concerns value domain definition and function's interpretation. Further on functions concerning semantics of level types, child-parent types, fact relationship types and primary key constraints.

- *Semantic domains definition*

Semantic domains are those for the basic data types, i.e. integer, real and string, the sets of names assigned to level types and the sets of names assigned to fact relationship types. The functions defining the data domains are shown further on.

D(int) = $\boldsymbol{Z}$, D(real) = $\boldsymbol{R}$, D(string) = $\boldsymbol{A}$

$\boldsymbol{FL}$: L $\in$ Lev_Type $\rightarrow D_L$

$\boldsymbol{FFR}$: FR $\in$ FactRel_Type $\rightarrow D_F$

- *Interpretation functions definition*

Attribute A **of type** d = D(d)

attOfLevel: LevType_Decl $\rightarrow$ P(Attributes) – returns the attribute names of the level

attOfFR: FactRelType_Decl $\rightarrow$ P(Attributes) – returns the attribute names of the fact relationship

***L*[Level type** L **has** A_D] = {f| f={l, attOfLevel(L_D)} $\wedge$ f(l)$\in$$\boldsymbol{FL}$ $\wedge$ $\forall$ A_i $\in$ attOfLevel(L_D) f(A_i) $\in$ **[Attribute** A **of type** d]}

***CP*[ChildParent type** CP **relates** L_1 **and** L_2] = {(s_{L1}, s_{L2}) $\wedge$ f(s_{L1}) $\in$ $\boldsymbol{FL}$ $\wedge$ f(s_{L2}) $\in$ $\boldsymbol{FL}$}

FR*[Fact relationship type** FR **involves** ***FL(L)] = {f| f= {l, $\cup_{L_i \in FL(L)}$ l_{Li}} $\wedge$ f(l) $\in$ $\boldsymbol{FFR}$ $\wedge$ $\forall$ L_i $\in$ ***FL***(L) f(l_{Li}) $\in$ ***FL***(L_i)}

FR*[Fact relationship type** FR **has** A_D **involves** ***FL(L)] = {f| f= {l, attrOfFR(FR_D) $\cup$ $_{L_i \in FL(L)}$ l_{Li}} $\wedge$ f(l) $\in$ $\boldsymbol{FFR}$ $\wedge$ $\forall$ A_i $\in$ attrOfFR(FR_D) f(A_i) $\in$ D(d) $\wedge$ $\forall$ L_i $\in$ ***FL***(L) f(l_{Li}) $\in$ ***FL***(L_i)}

Key[**K is primary key of** L] = K $\in$ attOfLevel($\boldsymbol{L}$) $\wedge$ $\forall$ a_i, a_j $\in\boldsymbol{L}$ (a_i(K) = a_j(K) $\rightarrow$ a_i(s) = a_j(s))

MAPPING CONCEPTUAL TO LOGICAL SCHEME

Logical design is performed on the conceptual scheme as input in order to produce a logical scheme. The logical scheme depends on the target logical model that has been chosen, relational or multidimensional. For mapping the conceptual model we've adopted the relational model and the star scheme [3]. The mapping process involves the following steps:

- *Define fact table's grain;*
- *Define dimension tables;*
- *Define fact table's primary and foreign keys;*
- *Define star scheme's loading and table population;*
- *Define aggregates (consolidated data).*

The fact relationship Order_line in the conceptual scheme is mapped to the star scheme fact table with a granularity being the individual order line. The numeric attributes of the fact relationship become fields in the table. The objects involved in the relationship enumerated in the 'involve' statement are mapped to fields that will form the composite primary key of the table. Dimension tables are defined from the 'dimension type' statements of the formalized conceptual model. Time dimension shouldn't be explicitly created. Tables' primary keys are defined from the attributes of type integer of 'level type' objects. The star scheme that maps the conceptual scheme is shown in Fig.4.

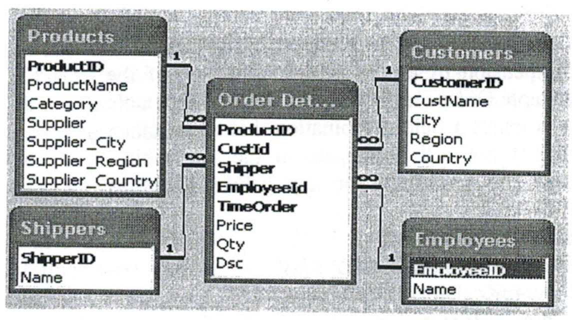

Fig.4. Star scheme mapping conceptual scheme from Fig.3

Having designed the star scheme it has to be loaded and corresponding consolidations are to be defined. Procedures for data load and consolidations are presented further on.

MAPPING PROCEDURES

- *Direct load procedure*

BULK INSERT Table FROM
...\Database \TableName

Tables Customers, Employees and Shippers are loaded without any modification from the source database. Field content from the table in the source database is copied into the destination field. The direct load procedure applied to Customers table is as follows:

BULK INSERT Customers FROM
D:\Orders\Customer

- *Load procedure with denormalization*

The procedure consists it several steps. Step1 bulk loads the table that'll be further denormalized. Step2 addresses definition of new fields in the table that'll be sourced from tables that are related to it via foreign keys. Step3 fills in values in the fields thus created by joining the corresponding tables. Step4 deletes foreign keys from the denormalized table. The procedure involving the stated steps is as follows:

Step1: Direct load procedure for Table1
Step2; INSERT FIELD
    Field1 datatype;
    …
INTO Table1;
Step3: UPDATE Table1
    INNER JOIN (Table2 INNER JOIN Table1 ON
    Table2.Key=Table1.ForeignKeyTable2) ON
    Table3.Key= Table1.ForeignKeyTable3
SET
    Table1.Field1 = [Table2]![Field$_i$],
    …
    Table1.Field1 = [Table3]![Field$_j$],
    …
Step4: DELETE FIELD
    ForeignKeyTable2, ForeignKeyTable3, …
FROM Table1

- *Fact table key generation procedure*

Fact table can be loaded either by direct load procedure or by loading with denormalization. The load procedure involves a step that generates the table's key. In a star scheme fact table's key is compound and consists of the dimensions' foreign keys. In case when the fact table is obtained by denormalization of two or more tables the choice of the star scheme's fact table is determined by the chosen granularity, i.e. orders as a whole or detailed order lines. In case of smaller granularity chosen the bigger table (with more records) is denormalized with fields from the one with fewer records. Order details fact table from Fig.4. is denormalized with fields representing foreign keys in the orders table as well as the date field, since time is an immanent dimension of a star scheme. The key of the degenerated table Order may be deleted from the fact table in case that it exists a field combination that represents a primary key. If not it'll participate in the fact table's primary key. The procedure for key generation is as follows:

PK = Initial Primary Key
GENERATE Fact table PRIMARY KEY
    ADD ForeignKeyDimTable[I];
    IF NotUnique (Primary Key)
        ADD DateField
        IF NotUnique (Primary Key)
          RESTORE PK
          ELSE
           DELETE Degenerate_Table_Key

- *Consolidation procedures*

They are defined for attributes of the fact relationship concerning levels of hierarchical dimensions. The output of a procedure is a table with aggregated values for the stated levels. A consolidation procedure in a general form is presented further on:

CONSOLIDATE FactAttribute FA, Expression (FA)
FOR $\cup$ Dimension [i].Level [j], Dimension [l], Expression (Dimension [x].Level [y])
BY ConsolidationFunction F
(CONSTRAINT ON $\cap$ Dimension [r].Level [s], Dimension [t])
OUTPUT TO Table$_K$

Brackets denote optional statement. The aggregation function for the attribute is derived from the conceptual scheme. For summarizable attributes the function by default is SUM, AVG for the nonsummarizable. Let's consider some applications of the consolidation procedure producing aggregations with finer or coarser granularity.

Qty_Category_Country (Qty, Product.Category, Customer.Country, SUM)
Qty_Month (Qty, Month(TimeOrder), Customer.City = "NY", SUM)
Sales_Year_Emp (Qty*(1-Dsc)*Price, Year(Date), Employee.Name, SUM)

CONCLUSION

The aim of our work has been to provide a framework for data warehouse design process that associates the conceptual and the logical design phase. Formal description of the conceptual scheme is implemented including hierarchies, semantic domains and interpretation functions. A strategy for logical design is presented. It takes as input the formal conceptual definitions and produces a logical star scheme. Mapping procedures are created thereabout. They provide for the basic scheme invariants concerning referential integrity and hierarchies. Future work is aimed at elaboration of load and consolidation procedures as rules for logical scheme design and further generalization of consolidation procedures into general select/project/join queries defining aggregations along dimensions' hierarchical levels.

REFERENCES

1. Golfarelli, M., S. Rizzi. Designing the Data Warehouse: Key Steps and Crucial Issues. Journal of Computer Science and Information Management, Vol. 2, No 3. 1999
2. Hahn, K., C. Sapia, M. Blaschka. Automatically Generating OLAP Schemata from Conceptual Graphical Models. Proceedings of the ACM DOLAP 2000 Workshop (DOLAP'2000), pp. 9-16. 2000
3. Kimball, R. The Data warehouse Toolkit. John Wiley & Sons, Inc. 1996
4. Malinovski, E., E. Zimanyi. Hierarchies in a Multidimensional Model: From Conceptual Modeling to Logical Representation. Data and Knowledge Engineering. 2005
5. Peralta, V., R. Ruggia. Using Design Guidelines to Improve Data Warehouse Logical Design. International Workshop on Design and Management of Data Warehouses, Berlin, Germany. 2003
6. Rozeva, A. Efficient Computation of Star-Net Query Model. Proceedings of the International Conference Automatics and Informatics'05, Sofia, Bulgaria, pp. 35-38. 2005
7. Rozeva, A. Performing Aggregate Queries on Partially Computed Data Cubes, Second International Scientific Conference Computer Science'2005, Chalkidiki, Greece, part 2, pp. 49-54. 2005
8. Sorensen, J., K. Alnor. Creating a Data Warehouse Using SQL Server. Proceedings of the International Workshop on Design and Management of Data Warehouses (DMDW'99), Germany. 1999
9. Tryfona, N., F. Busborg, J. Christiansen. starER: A Conceptual Model for Data Warehouse Design. Proceedings of ACM Second International Workshop on Data Warehousing and OLAP (DOLAP'99), USA, pp. 3-8. 1999
10. Wu, L., L. Miller, S. Nilakanta. Design of Data Warehouses Using Metadata. Information and Software Technology 43, pp 109-119. 2001