Constructing Object Oriented Class for extracting and using data from data cube

Antoaneta Ivanova

Abstract: *The goal of this article is to depict Object Oriented Conceptual Model Data Cube using it as an example. In short, the paper has described: 1) Structural level in object oriented class definition of data cube; 2) The basic methods for computing data cube from relational database by select operator; 3) To set tasks for future work to develop appropriate methods of class to construct data cube in different way from relational operator.*

Key words: *data warehouse, data cube, MOLAP, object oriented classes*

INTRODUCTION

Aggregation is predominant operation in decision support database systems. On-Line Analytical Processing (OLAP) databases often need to summarize data at various levels of detail and on various combinations of attributes.

Most developers agree that data warehouse, multidimensional database (MDB), and OLAP applications emphasize multidimensional modeling, which offers two benefits:

- ✓ closely parallels how data analyzers think and, therefore, helps users understand data;
- ✓ helps predict what final users want to do, thereby facilitating performance improvements.

Developers have proposed various approaches for the conceptual design of multidimensional systems. These proposals try to represent the main multidimensional properties at the conceptual level with special emphasis on data structures – for example as Object Oriented (OO) structure.

The goal of this paper is:

- ✓ to depict similar data cube class definition supplied cancer registry in our country;
- ✓ to test with real data.

Related works

Trujillo, Palomar and Gomez [8] propose an OO approach to accomplish the conceptual modeling of data warehouses, MDB, and OLAP applications. This approach introduces a set of minimal constraints and extensions to UML for representing multidimensional modeling properties for these applications. They base these extensions on the standard mechanisms that UML provides for adapting itself to a specific method or model, such as constraints and tagged values.

They believe that their innovative approach provides a theoretical foundation for the use of OO databases and object-relational databases in data warehouses, MDB, and OLAP applications. They use UML to design data warehouses because it considers an information system's structural and dynamic properties at the conceptual level more naturally than do classic approaches such as the Entity- Relationship model. Further, UML provides powerful mechanisms—such as the Object Constraint Language and the Object Query Language—for embedding data warehouse constraints and initial user requirements in the conceptual model. This approach to modeling a data warehouse system yields simple yet powerful extended UML class diagrams that represent main data warehouse properties at the conceptual level.

Multidimensional Modelling Properties

A *data cube* [2],[6],[7] is constructed from a subset of attributes in the database. Certain attributes are chosen to be *measure attributes*, i.e., the attributes whose values are of interest. Other attributes are selected as *dimensions* or *functional attributes.* The measure attributes are aggregated according to the dimensions.

The Figure 1 below depicts a practical data cube example, consider a hypothetical database of number of patients and number of diagnoses. This particular data cube has three feature attributes - *place, national cancer register*, and *time* - and a single measure attribute – *number of localization*. On the base of measure attribute may ask other questions. Such as: the most frequently occurrence of localization (using max function), the greatest age group of patients at risk for year or 5 years an example.

By selecting cells, planes, or subcubes from the base cuboid, we can analyze count of patients. In total, a d-dimensional base cube is associated with 2^d cuboids. Each cuboid represents a unique view of the data at a given level of granularity. Not all these cuboids need actually be present, however, since any cuboid can be computed by aggregating across one or more dimensions in the base cuboid.

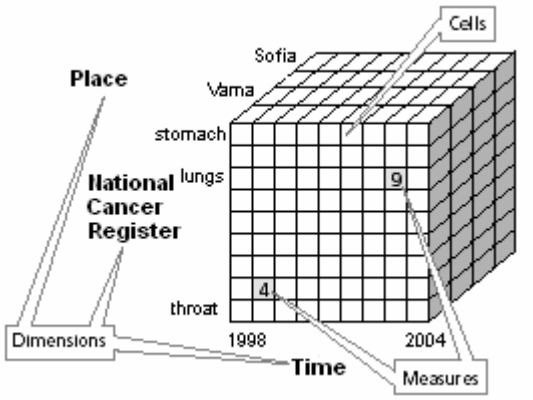


Figure 1. A multidimensional model data cube: - The cube is composed of cells that define fact attributes

Figure 2 shows the different classification hierarchies defined for the *National Cancer Register /localizations/*, *Place* and *time* dimensions. There is added the fourth hierarchy about *Patient.*

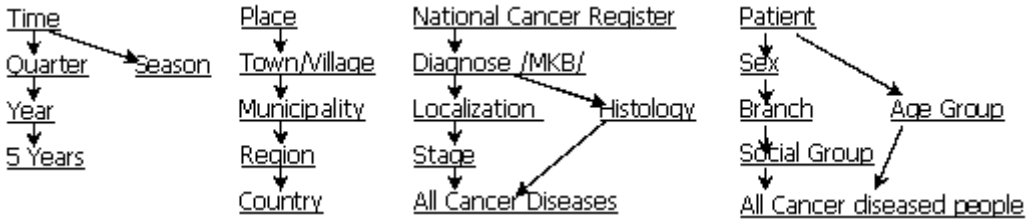


Figure 2. The classification hierarchies display the dimensions that define the cube – *Time, Place, National Cancer Register* and *Patient*

A measure is additive along a dimension if they can use the SUM operator to aggregate attribute values along all hierarchies defined on that dimension. The aggregation of some fact attributes—called roll-up in OLAP terminology— might not, however, be semantically meaningful for all measures along all dimensions.

Defining the classification hierarchies of certain dimension attributes is crucial because these classification hierarchies provide the basis for the subsequent data analysis. Because a dimension attribute can also be aggregated to more than one other attribute, multiple classification hierarchies and alternative path hierarchies are also relevant. For this reason, directed acyclic graphs provide a common way of representing and analyzing dimensions with their classification hierarchies.

Figure 2 shows the different classification hierarchies defined for the *time*, *place, National Cancer Register (diagnoses)* and *Patient* dimensions. On the *National Cancer Register (diagnoses)* dimension, a multiple classification hierarchy has been defined so that data values can be aggregated along two different hierarchy paths:

- ✓ *National Cancer Register–Diagnoses–Localization–Stage*
- ✓ *National Cancer Register–Diagnoses–Histology*.

For the *Patient* dimension, an alternative path classification hierarchy has been defined with two different paths that converge into the same hierarchy level:

- ✓ *Patient–Sex–Branch–Social group*
- ✓ *Patient–Age group.*

Finally, another alternative path classification hierarchy have been also defined with the following paths for the *time* dimension:

- ✓ *Time–quarter–year–5 years*
- ✓ *Time–season*.

On the forth fact dimension – *Place,* classification hierarchy without alternative path have been defined:

- ✓ P*lace–Town/Village–Municipality–Region–Country.*

In most cases, however, classification hierarchies are not so simple. The concepts of *strictness* and *completeness* are important for both conceptual purposes and for further multidimensional modeling design.

Once developers define the multidimensional model structure, users can define a set of initial requirements as a starting point for the subsequent data-analysis phase. From these initial requirements, users can apply a set of OLAP operations to the multidimensional view of data for further data analysis. These OLAP operations usually include the following:

- *roll-up,* which increases the level of aggregation along one or more classification hierarchies;
- *drill-down,* which decreases the level of aggregation along one or more classification hierarchies;
- *slice-dice,* which selects and projects the data;
- *pivoting,* which reorients the multidimensional data view to allow exchanging dimensions for facts symmetrically.

Method for data retrieval - CUBE operator

However, in the case of a MOLAP server, it is also the physical model, as MOLAP stores the cube structure directly as a multi-dimensional array. Conversely, ROLAP servers must map this representation to a relational design.

Figure 3 shows Relational database model - tables, witch consist of sets of fact attributes.

Fact attribute *Place* had been presented with multiple tables – *LiveInPlaces, Municipalities and Regions.*

Information about Town/Village is represented in table *LiveInPlaces.* There is data about 5400 towns and villages.

Table *Municipalities* consists of 260 rows and table *Regions* – 30 rows.

Fact attribute *Patient* had been presented with tables *Pac_bas* consists of data about over 80 000 patients from Varna region, mainly divided into 50 branches and 8 social groups.

Data of all cancer diseases consist in 2 basic tables – *MKB* and *Histol*, each one consists of 480 rows. Data of diagnoses are grouped in 15 localizations.

OLAP *is* multi-dimensional data. That being said, one might legitimately ask "How does one prepare data for multi-dimensional analysis, particularly if some or all of the 2^d cuboids are required?" Strictly speaking, no special operators or SQL extensions are required to take a raw data set, composed of detailed transaction-level records, and turn it into a data structure, or group of structures, capable of supporting subject-oriented analysis. Rather, the SQL *group-by* and *union* operators can be used in conjunction with *d* sorts of the raw data set to produce all cuboids. However, such an approach would be

both tedious to program and immensely inefficient, given the obvious inter-relationships between the various views. Consequently Jim Gray [4] proposed the data cube operator as a means of simplifying the process of data cube construction.

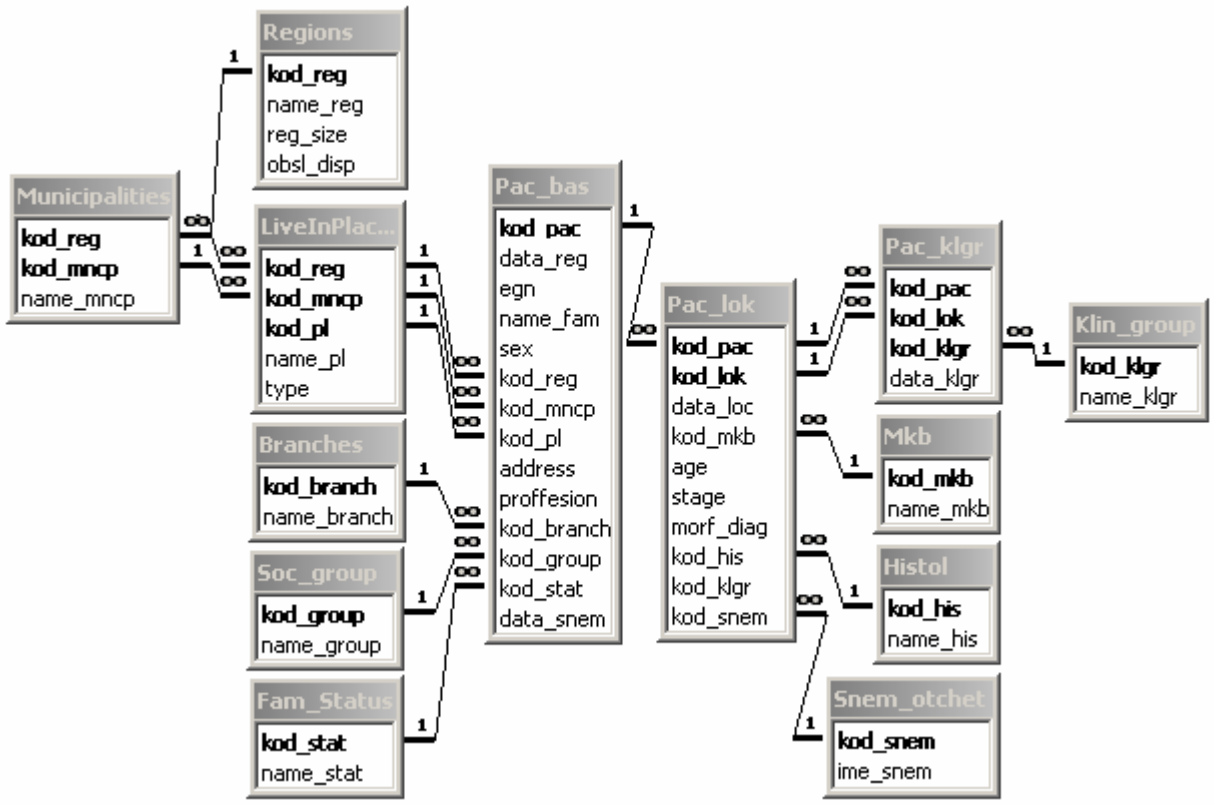


Figure 3. A relational data model

OO CONCEPTUAL MODEL APPROACH

OO approach can elegantly represent multidimensional properties at both levels:

Structural level

This OO approach is not restricted to using flat UML class diagrams to model large, complex data warehouse systems. UML's package grouping mechanism groups classes into higher-level units, creating different levels of abstraction and simplifying the final model. In this way, a UML class diagram improves and simplifies the system specifications created with classic semantic data models such as the Entity-Relationship model. This approach clearly separates the structure of a multidimensional model specified with a UML class diagram into facts and dimensions.

Facts and dimensions - Fact classes represent facts and the measures they are interested in, defined as attributes within these classes. Dimension classes represent dimensions.

Figure 4 shows the *Pac_lok* fact class—which consists of localizations—and the dimension classes *Place*, *MKB, Patient*, and *Time*. The fact class is thus specified as a shared-aggregation relationship between all dimension classes.

Derived measures are considered by placing the constraint /next to a measure in the fact class/. For example - *num_of_lok, num_of_pac.* Derivation rules are presented between braces in Figure 4.

Classification hierarchies – For dimensions, a base class represents every classification hierarchy level. An association of classes specifies the relationships between two levels of a classification hierarchy. The structure can represent both *alternative path* and *multiple classification hierarchies*.

For example *Place* classification hierarchy is shared with *Patient* attribute.

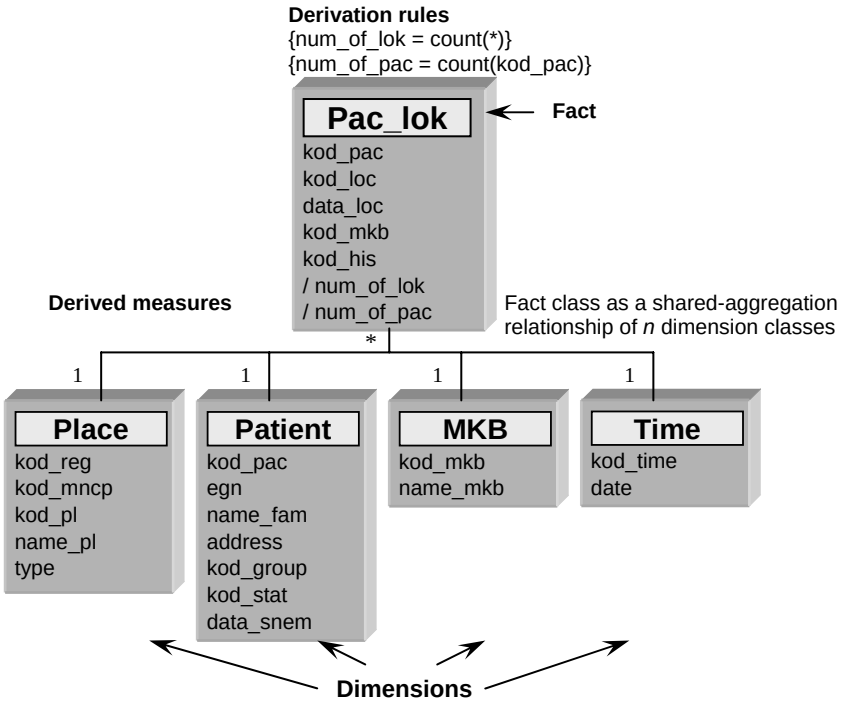


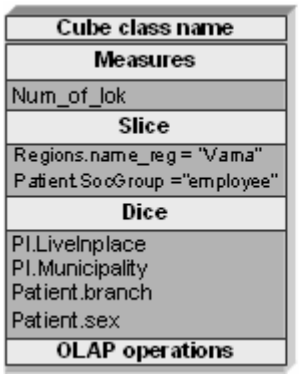
Figure 4. A fact class as a shared-aggregation relationship of *n* dimension classes. The *Pac_lok* class consists *derivation rules*, *derived measures* and has shared – aggregation relationship with the *Place*, *Patient*, *Mkb* and *Time* dimension

Relational commercial OLAP tools, however, use a default attribute within every classification hierarchy level that will be used in the subsequent data analysis phase. A *default* is a dimensional attribute that users want to analyze.

This default attribute displays every time the user applies an OLAP operation, rather than having the tool prompt the user to specify which attribute to display before executing each operation.

Therefore, to plan for subsequent automatic generation into a target relational OLAP tool, we must qualify the default attribute for every hierarchy level in the UML class diagram. Default attribute called a *descriptor* because they consider the term "default" too general. Thus, a descriptor in every class had been defined that represents a classification hierarchy level.

Dynamic level



Select count(p.kod_pac) as num_of_lok, *LiveInPlace*, *Municipality*
.....
From Pac_lok p, *LiveInPlaces* pl, *Municipalities* m, *Regions* r
....
Where r.name_reg= "Varna"
....
Group by *LiveInPlaces*: pl.name_pl, *Municipality*: m.name_mncp
....
Order by LiveInPlace, Municipality

Figure 5. Cube class example with parameters specified in the measures, slice, dice, and operations areas, and the class's corresponding Object-Query Language specification.

These *cube classes* are used to represent initial user requirements as the starting point for the subsequent data-analysis phase.

The basic components of the cube classes include the:

- ✓ head area, which contains the cube class's name;
- ✓ measures area, which contains the measures to be analyzed;

- ✓ slice area, which contains the constraints to be satisfied;
- ✓ dice area, which contains the dimensions and their grouping conditions to address the analysis; and
- ✓ cube operations, which cover the OLAP operations for a further data-analysis phase.

For nonexpert UML or database users, the cube class's graphical notation facilitates the definition of initial user requirements. Every cube class has a more formal underlying OQL specification. Experts can use OQL to define cube classes by specifying the appropriate OQL sentences.

CONCLUSIONS AND FUTURE WORK

OO Conceptual models of data cube are a very interesting direction for constructing and using efficiently information extracted from data cube.

OLAP tools implement a multidimensional model from two different levels:

• *Structural*—the structures that form the database schema and the underlying multidimensional model—also known as the metadata—that provides the model's key semantics (facts, measures, dimensions).

• *Dynamic*—refers to the definition of final user requirements—also known as method and OLAP operations for further analyzing data.

For further work:

- to develop appropriate method in object oriented data cube class definition for constructing data cube by means of array-based algorithm;

- to examine if this class is appropriate for other type of data – in example data describing properties of geographical map;

- add additional properties or methods in class definition for further analyzing data.

REFERENCES

[1] Ivanova A., B.Rachev, Multidimensional models – Constructing DATA CUBE, *In Proceedings of CompSysTech'04***,** Rousse (Bulgaria), 2004.

[2] Agarwal S., R. Agrawal, P. Deshpande, On the Computation of Multidimensional Aggregates, *In Proceedings of the .22nd international Conference on Very Large Databases***,** 506-521**,** Mumbai (Bombay), 1996.

[3] Geffner S., D. Agrawal, A. El Abbadi, T. Smith. Relative Prefix Sums: An Efficient Approach for Querying Dynamic OLAP Data Cubes, *In Proc. of the 15th International Conference on Data Engineering*, Sydney, Australia, March 1999.

[4] Gray J.,A. Bosworth, A. Layman, H. Pirahesh. Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-total, *In Proceedings of the 12$^{Th.}$ IEEE International Conference on Data Engineering,* 152–159, New Orleans,LA, February - March 1996

[5] Kotidis Y., N. Roussopoulos, An alternative storage organization for ROLAP aggregate views based on cubetrees, *ACM SIGMOD Record* 1998*,* 27(2): 248-258

[6] Lewis P., Database and Transaction Processing: An Application – Oriented Approach, Addison Wesley, New York, 2003

[7] Molina.H, J. Ullman, J. Windom, DataBase Systems: The Complete Book, Prentice Hall, 2002, 1119p.

[8] Trujillo J., M. Palomar, J. Gomez, Y. Song. Designing Data Warehouses with OO Conceptual Models, *Computer*, 66-75, December 2001, IEEE

ABOUT THE AUTHOR

Antoaneta Ivanova, PhD Student, Department of Computer Science and Technologies, TU-Varna, E-mail:antoaneta_ii@yahoo.com