published in: Proceedings of the 15th IEEE Symposium on Computer Based Medical Systems (CBMS 2002), IEEE Nr. PR01614, ISSN 1063-7125, pp 91-96

Incorporating ICD-9 and ICD-10 Data in a Warehouse

Johann Eder and Christian Koncilia
University of Klagenfurt
Dep. of Informatics-Systems
{eder,koncilia}@isys.uni-klu.ac.at

March 2002

Abstract

The shift from version 9 to version 10 of the ICD-code ("International Statistical Classification of Diseases and Related Health Problems") causes enormous problems for the exploitation of medical data warehouses, since conventional data warehouses do not support the change of the structure of dimensions, i.e. the content and relationships of master data like the diagnostic codes, or other key values. This shortcoming results in a reduction of possible analysis, and unfortunately is the cause of many wrong statistics and analysis results. In this paper we analyze the problem and show how to superimpose conventional multidimensional data warehouses with temporal master data to allow queries spanning multiple periods to return correct answers.

1 Introduction

A data warehouse (DWH) is an integrated repository for data stemming from several heterogenous database systems [7, 5]. This source data may be structured, semi-structured or unstructured. The most important usage of data warehouses is On-Line Analytical Processing (OLAP), typically using a multi-dimensional view of the data. OLAP tools then allow to aggregate and compare data along dimensions relevant to the application domain.

During the last years there has been an increasing pressure exerted on non-profit organizations (NPOs) and especially on health maintenance organizations (HMOs) to reduce costs for long term care, home health care, infusion services, acute inpatient care, etc. By giving managers of HMOs a tool to analyse the cost structure along multiple dimensions, data warehouses are an enabling technology. They will help to increase the confidence in data quality, to analyse data with an easy-to-use interface by means of

On-Line Analytical Processing (OLAP) tools, to develop benchmarks, etc. [4]

Data warehouses, i.e. OLAP tools, are well prepared to deal with modifications in transaction data, e.g. the changing values of the fact *Turnover* over time can be covered by introducing a dimension *Time*. Surprisingly, data warehouses are not well prepared for changes of the structure of dimensions in spite of their requirement for serving as long term memory.

Consider for example a data warehouse that stores information about diseases for all European countries along the dimensions *Time*, *Geography*, *Diagnosis* and *Facts*. In this real-world example at least two dimensions changed over time: *Geography* (for instance, the re-unification of Germany in 1990) and *Diagnoses* (for instance, the code for "malignant neoplasm of stomach" has changed from 151 in ICD-9 to $C16$ in ICD-10).

The question is now: how can we get correct results for queries like "did liver cancer increase over the last 5 years" or "show number of patients with diagnoses X in Germany over the last 25 years". Without knowing the above changes we will end up with incorrect results.

In this paper, we will discuss two different approaches how to deal with modifications in the dimensional structure: Representing temporal data in a temporal data warehouse and representing temporal data in a non-temporal OLAP system.

2 Temporal Data in a Temporal Warehouse

Our concept called *COMET* proposed in [1] and [2] extends the well known data warehouse approach with aspects of temporal databases and schema versioning. The changes we have to cope with are not only schema changes, but also changes in the dimension data (also called master data). The dimension *Time* ensures to keep track of the history of transaction data, i.e., measures. Nevertheless, for correct query results after modifications of dimension data we have to track modifications of these data [1].

Therefore, we extended the well known data warehouse approach with the following aspects [1]:

- **Temporal extension**: dimension data has to be time stamped in order to represent their *valid time*. The *valid time* represents the time when a "fact is true in the modeled reality" [6].
- **Structure versions**: by providing time stamps for dimension data the need arises that our system is able to cope with different versions of structure.
- **Transformation functions**: Our system has to support functions to transform data from one structure or schema version into another.

All dimension members and all hierarchical links between these dimension members have to be time stamped with a time interval $[T_s, T_e]$ representing the valid time where T_s is the beginning of the valid time, T_e is the end of the valid time and $T_e \geq T_s$. Furthermore, we timestamp all schema definitions, i.e. dimensions, categories and their hierarchical relations, in order to keep track of all modifications of the data warehouse schema [2].

If we represent all time stamps of all modifications within our data warehouse on a linear time axis the interval between two succeeding time stamps on this axis represents a structure version. This means that a structure version is a view on a temporal data warehouse valid for a given time period $[T_s, T_e]$. Therefore, within a structure version the structure of dimension data on both the schema level and on the instance level is stable.

The data returned by a query may originate in several (different) structure versions. If the data was affected by structural changes, it is necessary to provide transformation functions mapping data from one structure version to a different structure version.

Using transformation functions enables us to assure that a successful analysis can be made even though there might be changes in the dimension data and dimension structure. The combination of structure versions and transformation functions enables the user to analyze data with dimension data and dimension structures "backward" or "forward" in the time axis.

3 Temporal Data in a Non-Temporal OLAP System

The approach discussed in section 2 stores temporal data in a temporal data warehouse model. However, frequently the need arises that temporal data should be represented in a non-temporal data warehouse, i.e. a non-temporal OLAP system.

In this section we will discuss how to deal with temporal data in a non-temporal OLAP system, like Oracle Express or Cognos PowerPlay. In particular, we will discuss how to store temporal data in Hyperion Essbase Version 6.0, a popular and widely-used multidimensional database management system. However, we will not focus on the specific nuts and bolts of Hyperion Essbase.

3.1 Modifications of Master Data

We identified (beside the basic operations `INSERT` and `DELETE`) the following types of modifications:

- **Move**: The hierarchical position, i.e. the parent of a dimension member changes. For example "transient cerebral ischaemic attacks" ($G45$)

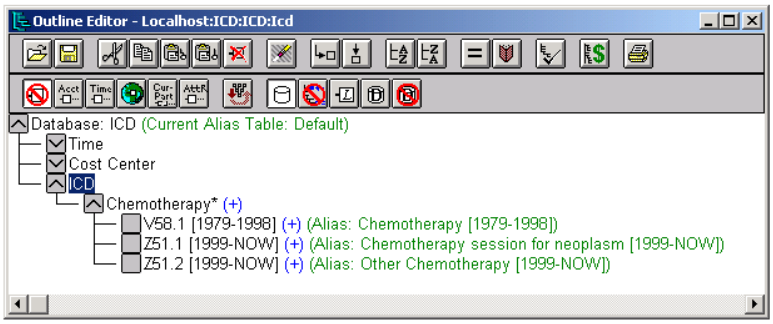

Figure 1: Incorporating a Subordinate Level

has moved from "Diseases of the circulatory system" $(390 - 459)$ to "Diseases of the nervous system" $(G00 - G99)$.

- **Change**: The key of a dimension member changes. For instance the code for "malignant neoplasm of stomach" has changed from 151 in ICD-9 to $C16$ in ICD-10.

- **Split**: A dimension member splits up into several other dimension members. For instance, the ICD-9 code for "Chemotherapy" $V58.1$ split up into $Z51.1$ "Chemotherapy session for neoplasm" and $Z51.2$ "Other Chemotherapy". Another example is the code 175 that split up into ten different ICD-10 codes, namely $C50.0$ to $C50.9$.

- **Merge**: Several dimension members merge into one dimension member. The ICD-9 codes 2350 and 2351 for instance, merged into the ICD-10 code $D37.0$.

3.2 Prefix/Suffix Notation

The main idea of this approach is to extend the keys of dimension members with a prefix or suffix representing the valid time of the dimension member. Usually, the keys used in Hyperion Essbase for dimension members are their names. For sake of readability, only dimension members that were subject of modifications get time stamped.

According to [6] data models for temporal database management systems "may represent a time line by a sequence of non-decomposable, consecutive time intervals of identical duration." These intervals are named chronons. A dimension *Time* is usually a part of multidimensional database applications. As this dimension defines the finest level on which modifications of transaction data are being stored, the chronon for the time stamp should be equal to the chronon of the dimension Time.

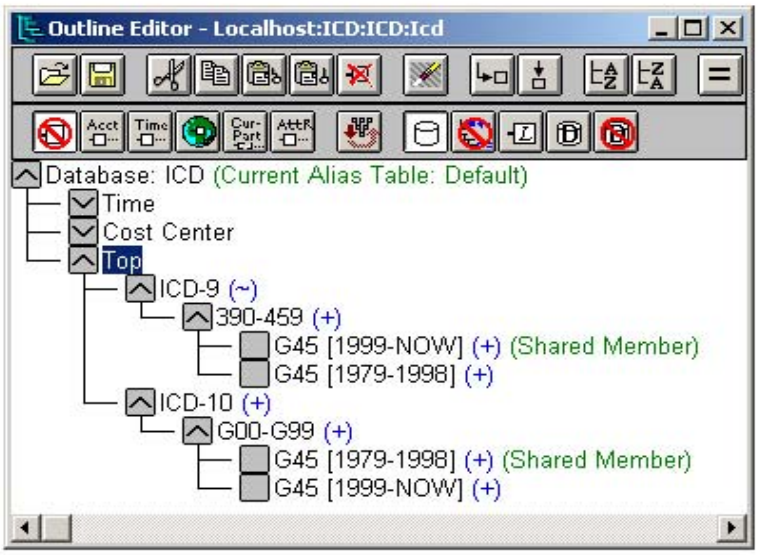

Figure 2: Modifications of Type *Move* in Hyperion Essbase

The chronon and format of the time stamps should be uniform within the whole cube, i.e., within the database. The format of the time stamp should comprise both, the start and the end of the valid time.

To enable easy multi-period comparisons even after modifications of dimension members, we propose an approach where modifications of dimension members are stored in a subordinate hierarchical level. In other words, we insert a new, artificial hierarchical level above the level of the dimension member which was subject of changes.

Figure 1 shows an example of such a subordinate level. In this example, we introduced a new hierarchy level by inserting a new dimension member named "Chemotherapy *". We added this asterisk to inform the user that this hierarchy is an artificial hierarchy. The children of this member are both, the old and the new version of the corresponding dimension member.

The main advantage of this approach is that it allows multi-period comparisons on the level of the artificial hierarchy for all mentioned types of modifications.

For a modification of the type *MOVE*, no artificial hierarchy level is necessary. Figure 2 shows how to represent the mentioned example ($G45$ has moved from $390-459$ in ICD-9 to $G00-G99$ in ICD-10). In order to avoid to double the values on the top level, correct consolidation functions have to be given. Hence, we have to exclude the ICD-9 path from consolidation. In Hyperion Essbase, this can be done by using the consolidation function ~ (tilde), were ~ means "exclude member from consolidation". In order to allow comparisons on the upper levels, the corresponding member has to inserted as a *shared member*. A shared member is a member with multiple

parents. In this example, the member $G45$ is a child of both $390 - 459$ and $G00 - G99$.

4 Conclusions

In this paper we discussed how to represent the changes that occurred due to the shift from version 9 to version 10 of the ICD-code in data warehouses. We focused on how represent this temporal behavior of master data in non-temporal data warehouses, i.e., in non-temporal OLAP systems.

We presented an approach that allows to represent the different types of modifications to master data, e.g. *Move*, *Split* and *Merge*. The mentioned approach allows multi-period comparisons, even after such types of modifications.

References

[1] J. Eder and C. Koncilia. Changes of Dimension Data in Temporal Data Warehouses. In *Proc. of the DaWak 2001 Conference*, Munich, Germany, 2001.

[2] J. Eder, C. Koncilia, and T. Morzy. The COMET Metamodel for Temporal Data Warehouses. In *Proc. of the 14th Int. Conference on Advanced Information Systems Engineering (CAISE'02)*, Toronto, Canada, 2002.

[3] O. Etzion, S. Jajodia, and S. Sripada, editors. *Temporal Databases: Research and Practise*. Number LNCS 1399. Springer-Verlag, 1998.

[4] E. Ewen, C. Medsker, L. Dusterhoft, K. Levan-Shultz, J. Smith, and M. Gottschall. Data Warehousing in an Integrated Health System: Building the Business Case. In *Proc. of the 3rd Int. Workshop on Data Warehousing and OLAP*, 1998.

[5] B. Hüsemann, J. Lechtenbörger, and G. Vossen. Conceptual Data Warehouse Design. In *Proc. of the International Workshop on Design and Management of Data Warehouses (DMDW 2000)*, Stockholm, 2000.

[6] C. S. Jensen and C. E. Dyreson, editors. *A consensus Glossary of Temporal Database Concepts - Feb. 1998 Version*, pages 367–405. Springer-Verlag, 1998. in [EJS98].

[7] M. Wu and A. Buchmann. Research Issues in Data Warehousing. *BTW'97*, 1997.