Empowering the OLAP Technology to Support Complex Dimension Hierarchies

Svetlana Mansmann, University of Konstanz, Germany

Marc H. Scholl, University of Konstanz, Germany

ABSTRACT

Comprehensive data analysis has become indispensable in a variety of domains. OLAP (On-Line Analytical Processing) systems tend to perform poorly or even fail when applied to complex data scenarios. The restriction of the underlying multidimensional data model to admit only homogeneous and balanced dimension hierarchies is too rigid for many real-world applications and, therefore, has to be overcome in order to provide adequate OLAP support. We present a framework for classifying and modeling complex multidimensional data, with the major effort at the conceptual level as to transform irregular hierarchies to make them navigable in a uniform manner. The properties of various hierarchy types are formalized and a two-phase normalization approach is proposed: heterogeneous dimensions are reshaped into a set of well-behaved homogeneous subdimensions, followed by the enforcement of summarizability in each dimension's data hierarchy. Mapping the data to a visual data browser relies solely on metadata, which captures the properties of facts, dimensions, and relationships within the dimensions. The navigation is schema-based, that is, users interact with dimensional levels with on-demand data display. The power of our approach is exemplified using a real-world study from the domain of academic administration.

Keywords: *data warehousing; OLAP (online analytical processing; meltidimensional database design*

INTRODUCTION

Data warehouse technology introduced in the early 90s to support data analysis in business environments has recently reached out to non-business domains such as medicine, education, research, government, etc. End-users interact with data using advanced visual interfaces that enable intuitive navigation to the desired data subset and granularity and provide a visually enhanced presentation using a variety of visualization techniques.

Data warehouse systems adopt a *multidimensional data model* tackling the challenges of the online analytical processing (OLAP) (Codd, Codd, & Salley, 1993) via efficient execution of queries that aggregate over large amount of detailed data. The analysis is preceded by a highly complex ETL (extract, transform, load) process

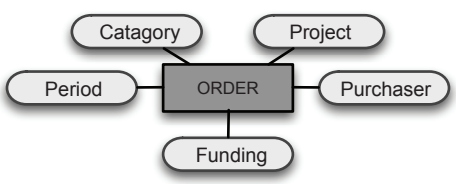
of integrating the data from multiple systems and bringing it into a consistent state.

In relational OLAP systems, multidimensional views of data, or *data cubes*, are structured using a *star* or a *snowflake* schema consisting of fact tables and dimension hierarchies. *Fact tables* contain data records (*facts*) such as transactions or events, which represent the focus of the analysis. Facts are composed of two types of attributes: (1) *measures* (i.e., the actual elements of the analysis), and 2) *dimensions,* which uniquely determine the measures and serve as exploration axes for aggregation. Members of a dimension are typically organized in a containment type hierarchy to support multiple granularities. In the dimension table, the attributes that form the hierarchy are called *dimension levels*, or *categories*. Other descriptive attributes belonging to a particular category are known as *property* attributes. Dimension levels along with parent/child relationships between them are referred to as the dimension's *intension*, or *schema*, whereas the hierarchy of its members forms its *extension*.

Figure 1 shows the star schema view of a data cube storing the administrative expenditures of a university: the facts in the fact table ORDER are determined by five dimensions. In the star schema, the whole dimension hierarchy is placed into a single table, whereas the snowflake schema enforces the hierarchy to be decomposed into separate tables, one table per dimension level.

The two logical design options are illustrated in Figure 2 at the example of the dimension Period. The star schema produces a single table period with all dimension levels and property attributes. Obviously, in such denormalized view it is impossible to explicitly recognize the hierarchical relationships. In the snowflake schema, however, each dimension category with its property attributes is placed into a separate table referencing its parent. The arrows correspond to the foreign keys (i.e., the roll-up relationships between the levels). The resulting schema is rather complex, but it offers the advantage of automatic extraction of the hierarchy schema with all valid aggregation paths from the foreign key constraints. Notice that reoccurring intervals such as weeks, months, quarters, etc. are presented by a two-category lattice (e.g., months → month) in order to be able to roll-up single instances to the instance's type. For example, months instances *"January 1997"* and *"January 1998"* rollup to month instance *"January."*

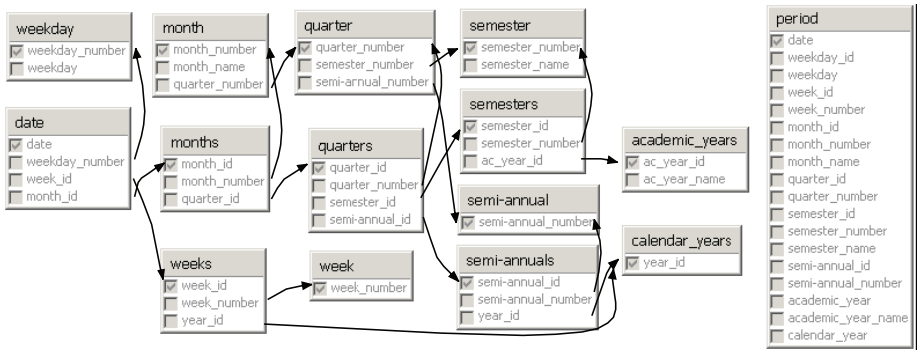
Figure 1. Star schema view of data



Summarizability and Homogeneity

The rigidness of the standard OLAP technology is caused primarily by the enforcement of summarizability for all dimensional hierarchies. The concept of summarizability, coined by Rafanelli and Shoshani (1990), and further explored by other authors (Hurtado & Mendelzon, 2001; Lenz & Shoshani, 1997), requires distributive aggregate functions and dimension hierarchy values, or informally, that (1) facts map directly to the lowest-level dimension values and to only one value per dimension, and (2) dimensional hierarchies are balanced trees (Lenz et al., 1997). In practice, summarizability guarantees correct aggregation and optimized performance, as any aggregate view is obtainable from a set of pre-computed views defined at lower aggregation levels. However, the hierarchies in many real-world applications are not summarizable and, therefore, cannot be used as dimensions in their original form. In case of small irregularities, the tree can be balanced by filling the "gaps" with artificial nodes. In highly unbalanced hierarchies, such transformations may be very confusing and undesirable. Yet in other scenarios, it is crucial to preserve the original state of the hierarchy.

Figure 2. Snowflake schema (left) vs. star schema (right) of a time hierarchy



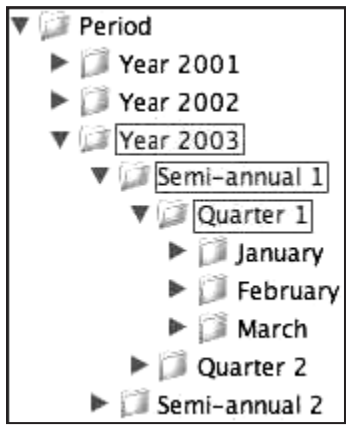
At the level of visual analysis, summarizability is also imperative for generating a proper navigation hierarchy. Data browsers present hierarchical dimensions as recursively nested folders of their levels allowing users to browse either directly in the dimension's data or to access its schema. In the former approach, henceforth denoted "*extension-based,"* the navigation tree of a dimension is a straightforward mapping of the dimension's data tree: each hierarchical value is a node that can be expanded to access the child values nested therein. Popular commercial and open-source OLAP tools, such as Cognos PowerPlay (http://www.cognos.com/powerplay) and Mondrian OLAP Server (http://mondrian.sourceforge.net) provide only this simple navigation.

Alternatively, the navigation hierarchy can explicitly display the schema of the dimension, with each category as a child node of its parent category. This so called *"intension-based"* approach is especially suitable for power analysis and employment of advanced visualization techniques. Sophisticated OLAP solutions, such as Tableau Software (http://www.tableausoftware.com) and SAP NetWeaver BI (http://www.sap.com/solutions/netweaver/components/bi), combine schema navigation with data display. Figure 3 shows the difference between instance-based and schema-based browsing for a hierarchical dimension Period.

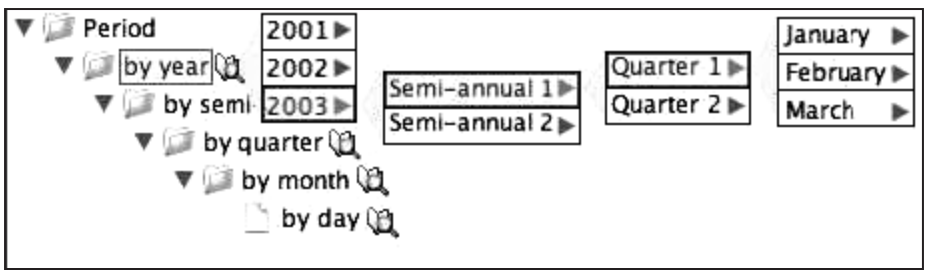
Another restriction of the traditional approach to dimension modeling is that of *homogeneity*. Even though it is admissible to define multiple hierarchies within the same dimension (e.g., date in Figure 2 can be rolled-up to weekday, weeks, or months), each of those hierarchies must be homogeneous (i.e., each level of the tree corresponds to a single dimension category and all members of a given category have ancestors in the same set of categories (Hurtado & Mendelzon, 2002)). The necessity of dropping this restriction has been recognized by the researchers who proposed respective extensions in the form of multidimensional normal forms (Lechtenbörger & Vossen, 2003; Lehner, Albrecht, & Wedekind, 1998;), dimension constraints (Hurtado et al., 2002), transformation techniques (Pedersen, Jensen, & Dyreson, 1999), and mapping algorithms (Malinowski & Zimányi, 2006).

Analysts are frequently confronted with non-summarizable data that cannot be adequately supported by standard models and systems. In a survey on open issues in multidimensional modeling, Hümmer, Lehner, Bauer, & Schlesinger (2002) identified unbalanced and irregular hierarchies as one of the major modeling challenges for both researchers and practitioners. To overcome the restrictions of summarizability and homogeneity and thus increase the capacity of the OLAP technology to handle a broader spectrum of practical situations, analysis tools have to be extended at virtually all levels of the system architecture:

Figure 3. Browsing in dimensional hierarchies: extension vs. intension navigation



(a) dimension instances



(b) dimension levels with on-demand data display

- Recognition and classification of complex hierarchies,
- Conceptual and logical model extensions,
- Data and schema normalization techniques,
- Enhanced metadata model to ensure correct querying and aggregation,
- Lossless mapping of dimension schema to a visual navigation,
- Adequate visualization techniques for presenting complex query results.

Related Work

Limitations and deficiencies of the classical multidimensional data model have become a fundamental issue in the data warehousing research in the last decade. The necessity to develop novel concepts has been recognized (Zurek & Sinnwell, 1999) and a series of extensions have been proposed in the recent past. As state-of-the-art solutions are far from being ultimate and overall satisfactory, the problem will continue to attract interest and encourage new contributions in the years to come.

A powerful approach to modeling dimension hierarchies along with SQL query language extensions called SQL($\mathcal{H}$) was presented by Jagadish, Lakshmanan, and Srivastava (1999). SQL($\mathcal{H}$) does not require data hierarchies to be balanced or homogeneous. Niemi, Nummenmaa, and Thanisch (2001) analyzed unbalanced and ragged data trees and demonstrated how dependency information can assist in designing summarizable hierarchies. Lehner et al. (1998) relaxed the condition of summarizability to enable modeling of generalization hierarchies by

defining a generalized multidimensional normal form (GMNF) as a yardstick for the quality of multidimensional schemata. Lechtenbörger et al. (2003) pointed out the methodological deficiency in deriving multidimensional schema from the relational one and extended the framework of normal forms proposed by Lehner et al. (1998) to provide more guidance in data warehouse design.

Other works focus on formalizing the dimension hierarchies and summarizability related constraints. Hurtado et al. (2001) proposed integrity constraints for inferring summarizability in heterogeneous dimensions and defined a comprehensive formal framework for constraint-conform hierarchy modeling in a follow-up work in 2005. Another remarkable contribution to the conceptual design was made by Malinowski et al. (1996) who presented a classification of dimensional hierarchies, including those not addressed by current OLAP systems in 2004 and formalized their conceptual model and its mapping to the relational schema in 2006.

Pedersen et al. (1999) have made manifold contributions in the area of multidimensional modeling. In 2001, they formulated the major requirements an extended multidimensional data model should satisfy and examined 14 state-of-the-art models from both the research community and commercial systems. Since none of the models was even close to meeting most of the defined requirements, the authors proposed an extended model for capturing and querying complex multidimensional data. This model, supporting non-summarizable hierarchies, many-to-many relationships between facts and dimensions, handling temporal changes and imprecision, is by far one of the most powerful multidimensional data models of the state of the art.

Support for complex hierarchies in the existing OLAP systems falls far behind the respective abilities of the formal models: to our best knowledge, most of the extensions proposed in the above contributions have not been incorporated into any analysis software. In a previous work (Vinnik & Mansmann, 2006), we presented some insights into visual querying of a subset of irregular dimensions. A more recent work (Mansmann & Scholl, 2006) analyzes the limitations of the standard OLAP systems and the underlying data model in handling complex dimensional hierarchies and proposes the extensions at the conceptual level, which are then propagated to an advanced visual interface. The current work builds upon the proposals of the latter work, however, with the focus on providing a more formal and comprehensive categorization of dimensional hierarchies and proposing the approaches to their modeling and transformation at the conceptual and logical level. As a proof of concept, an approach to visual querying and analysis of complex hierarchies via a schema-based navigation is presented.

Contributions and Outline

This work is an attempt to further reduce the gap between powerful concepts and deficient practices in providing data warehouse support for complex data. In the context of our research, complex data is used as a generic term referring to all types of dimension hierarchies not supported by traditional systems. The techniques we present evolved from the practical experiences with data warehouse design and challenges encountered in the real-world applications. The related proposals to handling complex hierarchies found in the literature tend to focus on the formal aspects of the multidimensional data modeling and provide no solution for implementing the proposed extensions in a visual interface. However, since the analysis is performed predominantly via visual tools, we consider the practicability a crucial aspect for judging about the value of the modeling proposals. Therefore, we also identify the potential problems of supporting complex data at the level of user interfaces and present an approach to adequately mapping the extended data model to a navigation structure of a prototypical OLAP tool.

Our approach to extending the capabilities of OLAP systems is meant as a comprehensive framework that includes the conceptual and the

logical design, transformation techniques, metadata management, and mapping algorithms for presenting data cubes as navigation hierarchies and translating user interactions into valid queries. The advantage of the presented solution is its ability to handle a wide spectrum of hierarchy patterns in a uniform and intuitive manner. At the conceptual level, we propose a systematic categorization of dimension hierarchies, with formal definitions, examples from a real-world scenario, and a description of relationships between various dimension types.

We also describe a two-phase data transformation approach aimed at bringing complex hierarchies into a navigable state. The awareness of the supported hierarchy types is propagated to the analysis tools by enriching the metadata that describes the schema of the data warehouse. Consequently, the analysis tools need to implement the necessary "database-to-navigation," "navigation-to-query," and "query-to-visualization" mappings to support querying of the newly added types of dimensions.

The article is structured as follows: the second section sets the stage by formalizing the basis elements of the model. In the third section, a case study from the area of academic administration is presented and used for deriving a comprehensive classification of dimension hierarchy types. The fourth section introduces the mapping algorithms for the relational implementation of the proposed conceptual extensions, consisting of schema and data normalization techniques. The process of generating a powerful navigation framework from the data schema and examples of using the latter for visual exploration of complex data are presented in the fifth section. We summarize our contribution and identify future research directions in the sixth section.

TERMINOLOGY AND BASIC CONCEPTS

This section describes the formal framework of the multidimensional modeling. We rely on the terminology and formalization introduced by Pedersen et al. (2001) since their model is the most powerful of the state of the art in part of handling complex hierarchies. We also adopt some elements of SQL($\mathcal{H}$) model (Jagadish et al., 1999) to account for heterogeneous hierarchies.

Hierarchy Schema and Instance

Intuitively, data hierarchy is a tree with each node being a tuple over a set of attributes. A dimension hierarchy is based on a hierarchical attribute, also referred to as the *analysis criterion*, propagated to all levels of the tree. It is possible to impose different hierarchies within the same dimension by defining multiple criteria, for instance, the projects can be analyzed along the hierarchy of geographic locations or along that of supervising institutions.

Definition 2.1.1. A hierarchical domain is a non-empty set $V_{\mathcal{H}}$ with the only defined predicates = (identity), < (child/parent relationship), and << (transitive closure, or descendant/ancestor relationship) such that the graph $G_<$ over the nodes e_i of $V_{\mathcal{H}}$ is a tree. Attribute A of $V_{\mathcal{H}}$ is called a hierarchical attribute.

A hierarchy H is non-strict whenever $\exists(e_1, e_2, e_3) \in V_H : e_1 < e_2 \wedge e_1 < e_3 \wedge e_2 \neq e_3$, or, informally, if any node is allowed to have more than one parent.

Definition 2.1.2. A hierarchy schema $\mathcal{H}$ is a four-tuple $(C, \sqsubseteq_{\mathcal{H}}, \top_{\mathcal{H}}, \bot_{\mathcal{H}})$, where $C = \{C_j, j = 1, \ldots, k\}$ is a set of category types, $\sqsubseteq_{\mathcal{H}}$ is a partial order on C, $\top_{\mathcal{H}}$ is a distinguished root category and $\bot_{\mathcal{H}}$ is the bottom level of the ordering.

Predicates =, $\sqsubseteq$, and $\sqsubseteq^*$ specify identity, child/parent, and descendant/ancestor relationship, respectively, between the category types in C. The only possible relation of a category with its own self is identity (i.e., the category itself does not belong to the subset of its descendant categories). C_j is said to be a category type in $\mathcal{H}$, denoted $C_j \in \mathcal{H}$, if $C_j \sqsubseteq^* C$. Thereby, the hierarchy schema defines a skeleton of the associated data tree, for which the following conditions hold:

a. $\forall C_j \in C : C_j \sqsubseteq^* \top_{\mathcal{H}}$ (each category rolls up to the root);
b. $\nexists C_j' \in C : C_j' \sqsubseteq \bot_{\mathcal{H}}$ (childless bottom category).

Definition 2.1.3. An aggregation path in C is any pair of category types C_i, C_j such that C_i, $C_j \in C \wedge C_i \sqsubseteq^* C_j$.

The instance, or the intension, of a hierarchy results from specifying the elements for each category type and the child/parent relationships between them.

Definition 2.1.4. A hierarchy (instance) H associated with hierarchy schema $\mathcal{H}$ is a two-tuple $(C, <_{\mathcal{H}})$, where $C = \{C_j\}$ is a set of categories such that $Type(C_j) = C_j$ and $<_{\mathcal{H}}$ is a partial order on $\cup_j C_j$, the union of all dimensional values in the individual categories.

A category C_j is a set of dimensional values e, also known as the member set. The number of values in set C_j is denoted $|C_j|$. The following conditions hold for a hierarchy instance:

a. $\forall e_i \in C', \forall e_j \in C'' : e_i < e_j \Rightarrow C' \sqsubset C''$ (connectivity);
b. $\nexists e_i \in C', \nexists e_j \in C'' : e_i = e_j \wedge C' \neq C''$ (disjointness of categories);
c. $\forall C' : (\nexists e_i \in C', \nexists e_j \in C' : e_i \ll e_j)$ (stratification, i.e. no parent/child relations between the members of the same category);
d. $(Type(C_j) = \top_{\mathcal{H}}) \Rightarrow |C_j| = 1 \wedge C_j = \{\mathcal{ALL}\}$ (root category consists of a single value $\mathcal{ALL}$).

Definition 2.1.6. Dimension schema $\mathcal{D}$ is a directed graph constructed by merging the graphs of all hierarchy schemata associated with the given dimension. The set of vertices in $\mathcal{D}$ is a duplicate-free union of category types from $\{\mathcal{H}\}$ and the edges are the union of the partial orders from $\{\mathcal{H}\}$. $\mathcal{D}$ contains each schema $\mathcal{H}' \in \{\mathcal{H}\}$ as its subgraph.

From the previous definitions, a generic notion of a dimension can be derived:

Definition 2.1.5. A dimension D is defined by a set of hierarchy schemata $\{\mathcal{H}\}$ with a common root category type $\top_{\mathcal{D}}$, a common bottom category type $\bot_{\mathcal{D}}$, and the associated set of hierarchy instances {H}.

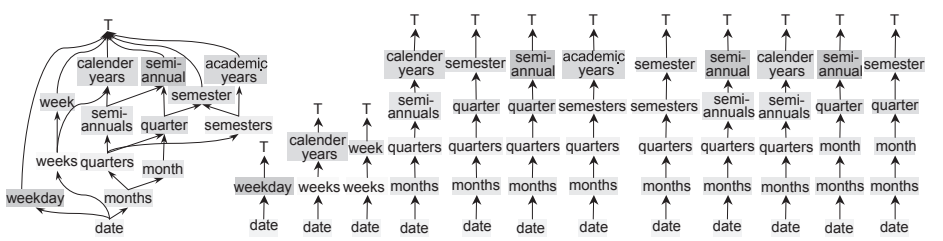
The previous definition without further constraints admits a broad variety of hierarchical behaviors within a dimension. For instance, it is not prohibited to have multiple hierarchies, shared category types, or partial roll-up relationships.

Figure 4 illustrates the schema of the entire dimension Period as a graph resulting from merging the schemata of all hierarchies found in that dimension.

Dimension Categorization via Constraint Definition

We proceed by deriving the definitions of dimension types by introducing various constraints on its schema and instances. Adherence or violation of a particular constraint is used for drawing a classification of dimension types. The

Figure 4. Dimension schema (left) as a super-graph of its subdimensions (right)



first constraint is concerned with the number of hierarchies in a dimension.

Definition 2.2.1. A dimension is simple if it contains exactly one hierarchy, so that the dimension schema is identical to this hierarchy schema:

Simple(D): $|\mathcal{H}| = 1 \wedge \mathcal{D} = \mathcal{H}$.

A dimension with more than one hierarchy represents multiple hierarchies:

Multiple(D): $|\mathcal{H}| > 1$.

Another distinction criterion is the number of levels in the dimension schema.

Definition 2.2.2. A simple dimension is flat if its hierarchy schema consists solely of the top and the bottom category type:

Flat(D): $\nexists C_i \in \mathcal{H}: C_i \neq \top_D \wedge C_i \neq \infty_D$.

A simple dimension is hierarchical if its schema contains at least one category type, which is neither the top nor the bottom one:

Hierarchical(D): $\exists C_i \in \mathcal{H}: C_i \neq \top_D \wedge C_i \neq \perp_D$.

Non-flat dimensions can be recursively decomposed into their subdimensions.

Definition 2.2.3. Dimension D' is a subdimension of D if the schema of D' is a subgraph of the schema of D and each $<_{H'}$, $H' \in D'$, *is a restriction of* $<_H$, $H \in D$, *to the corresponding categories.*

To assess the homogeneity of a dimension, it is necessary to examine the roll-up behavior in all its categories:

Definition 2.2.4. A dimension is homogeneous if each of its categories totally rolls-up to the parent category: $\forall C_i \sqsubset C_j$, $\forall e_i \in C_i : \exists e_j \in C_j \wedge e_i < e_j$. *A heterogeneous dimension admits multiple exclusive paths (partial roll-up) in its hierarchy, with each member belonging to one path:* $\exists e_i \in C_i$, $\exists C_j : C_i \sqsubset C_j \wedge (\nexists e_j \in C_j : e_i < e_j)$.

In the next section, we present a case study that provides examples helpful for explaining further types of dimension hierarchies.

OVERVIEW OF DIMENSIONAL HIERARCHIES: A CASE STUDY

The presented case study is concerned with the expenditures of a university, registered in form of purchase orders. Academic management wishes the data to be organized into an OLAP cube where the fact table ORDERS contains single orders with the measure attributes amount and items and dimensional characteristics period, category, project, purchaser, and funding. The values of each dimension are further arranged into hierarchies by defining the desired granularity levels, as illustrated by a diagram in ME/R notation (multidimensional entity/relationship, proposed by Sapia, Blaschka, Höfling, and Dinter (1999)) shown in Figure 5.

The hierarchy types are introduced starting with more general categories and proceeding to their subclasses and special cases, with the entire categorization tree depicted in Figure 6. First of all, a dimension is a *hierarchy* or a *non-hierarchy*.

- **Non-hierarchy:** dimension consists of a single category (i.e., not involved in any incoming or outgoing roll-up relationship, as is the case with funding.

The next distinction is made between *simple* and *multiple* hierarchies.

- **Simple hierarchy:** is a dimension with a single hierarchy defined upon it. Each member value rolls-up to a single parent value in a single category. Example of a simple hierarchy is category where cost class members roll-up either to cost group or to cost category, but not to both.
- **Multiple hierarchies:** arise whenever more than one hierarchy is defined within

a dimension (i.e., when a category has multiple outgoing roll-up relationships). Thereby, multiple non-exclusive aggregation paths exist within the same dimension.

Simple Hierarchy Types

Simple hierarchies are subdivided into two major classes, namely *homogeneous* and *heterogeneous* ones, as defined in the previous section. The dimension project contains a homogeneous subdimension office → building → city. However, the entire path from project to city is heterogeneous as project values are allowed to roll up either to office or city.

Another property of a simple hierarchy is whether it is *strict* or *non-strict*.

- **Strict hierarchy:** disallows many-to-many cardinalities in the child/parent relationships of its members. At the schema level, each category has at most one outgoing roll-up relationship, as observed in the subdimension chair → department → faculty → section.
- **Non-strict hierarchy:** has at least one many-to-many relationship between its categories. In our example, such relationship exists in project → project group where a project may be associated with multiple project groups.
- **Weighted non-strict:** hierarchy restores the summarizability by enforcing to specify each element's degree of belonging to each of its parent elements. The process of obtaining this hierarchy type is described in a following section.

A homogeneous hierarchy should be tested for symmetry:

- **Symmetric:** hierarchy is a simple hierarchy in which all levels in the schema are mandatory and whose instance is thus a balanced tree. For instance, the hierarchy of the schema office → building → city is symmetric.
- **Asymmetric:** (non-onto) hierarchy is a simple hierarchy that allows childless members in non-bottom categories. For example, in the roll-up relationship administrative staff → administrative division, a division may appear to have no staff in purchaser role.

Heterogeneous hierarchies occur whenever the members of the same category roll-up along different paths. Typically, such hierarchies result

Figure 5. University expenditures case study as ME/R Diagram

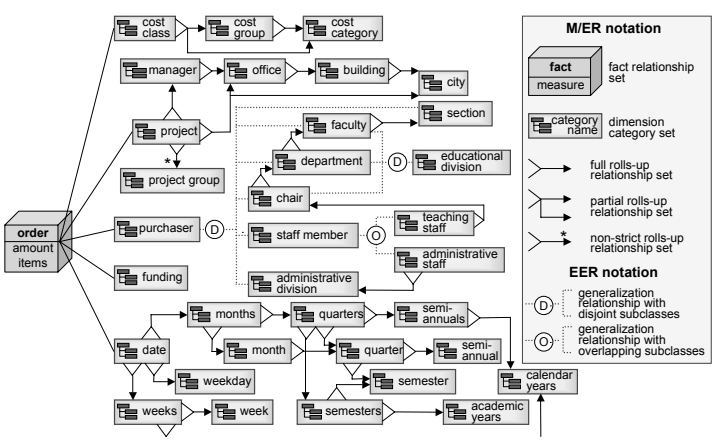
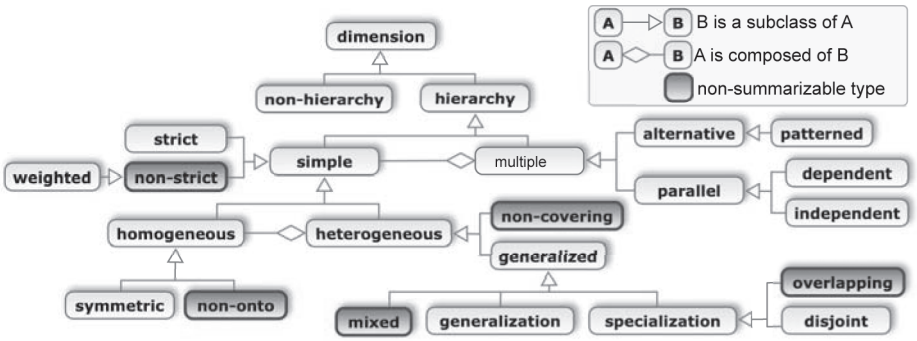


Figure 6. Categorization of dimension hierarchies



from generalization/specialization relationship between some categories.

- **Generalized hierarchy:** contains categories that can be represented by a generalization relationship (i.e., a category is superclass for other categories).

The categories at which the alternative paths split and join are called *splitting* and *joining* levels, respectively. Dimension purchaser is an example of a generalized hierarchy with the bottom category purchaser as the splitting level and educational division as the joining level. We distinguish special cases of specialization and generalization hierarchies that admit only splitting and only joining levels, respectively.

- **Generalization:** hierarchy uses superclasses for uniting multiple categories to treat their members as one category. For example, educational division is introduced to treat the members of chair, faculty, and department as one class. The actual members belong to the subclass categories and the superclass is introduced upon it.
- **Specialization:** hierarchy uses subclasses as roll-up categories of the superclass category, as can be observed in staff member, which is subdivided into teaching staff and administrative staff. The actual members belong to the superclass and the subclasses are introduced upon it. A specialization hierarchy may emerge in case a dimension originally treated as homogeneous is divided into subclasses in order to refine its characteristics for the analysis. As in the previous example, subdivision enables the introduction of subclass-specific aggregation hierarchies.
- **Disjoint specialization:** guarantees non-overlapping subclass hierarchies of a superclass category, thus preserving the strictness of the hierarchy.
- **Overlapping specialization:** allows the subclasses of the same category to overlap, resulting in a non-strict hierarchy. For example, the overlapping subcategories of staff member allow the same element to belong to both teaching staff and administrative staff.
- **Mixed:** hierarchy is a special case of a generalized hierarchy, in which a roll-up relationship exists between the subclass categories of the same superclass. In our example, the granularity of the category purchaser is mixed, as its subclasses chair, department, and faculty build their own hierarchy.
- **Non-covering:** hierarchy is a subclass of heterogeneous hierarchies, in which exclusive paths are obtained by allowing the roll-up relationships to be partial. For instance, members of cost class roll up either to cost group or to cost category.

Figure 7. Time hierarchy modeled as an ordinary dimension

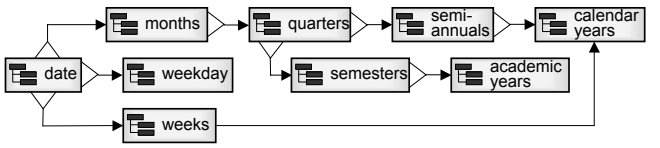
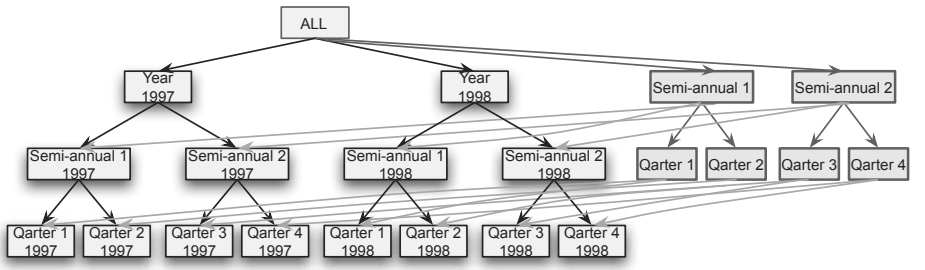


Figure 8. Adding categories semi-annual and quarter *(right) to the time hierarchy (left)*



Types of Multiple Hierarchies

Multiple hierarchies are subdivided into *alternative* and *parallel* according to whether their constituent hierarchies refer to the same or to different analysis criteria, respectively.

- **Multiple alternative:** hierarchies are non-exclusive aggregation paths with at least one shared level in the dimension schema. It is essential that all hierarchies be based on the same analysis criterion. The entire dimension period is an example of such hierarchies—any pair of aggregation paths shares at least the bottom level date and accounts for the same criteria, namely the date value.
- **Patterned:** hierarchies are a special case of multiple alternatives in which a certain regularity, or pattern, re-occurs in the child/parent relationships of its members. The classical example is the time hierarchy.

Treating time as an ordinary dimension would yield a schema shown in Figure 7. However, recognition of the reoccurring patterns, such as that each year consist of the 1st and the 2nd semi-annual, of the same four quarters, the same 12 months, etc., makes it possible to generate further valid hierarchies. This is done by explicitly representing the type of the patterned category as its parent category. For instance, members of quarters roll up to the category quarter (e.g., "*1st Quarter 1997*" is a child of "*1st Quarter*"). The new enriched schema of period is capable of supporting additional aggregation paths, as shown in Figure 8.

- **Parallel:** hierarchies in a dimension account for different analysis criteria. An example of parallel hierarchies can be found in the dimension project: there exists a hierarchy office → building→ city for project locations and a hierarchy manager→ office → building→ city for project supervisions.
- **Parallel independent:** hierarchies do not share levels (i.e., have non-overlapping sets of members). In project, the hierarchy of project locations with categories office, building, and city is independent of the project group hierarchy.
- **Parallel dependent:** hierarchies have shared categories, as is the case with the hierarchies on project locations and on project supervisions as they have a shared set of categories {office, building, city}.

FROM CONCEPTUAL TO LOGICAL DESIGN

To proceed with the logical design, the choice is to be made between the star and the snowflake schema options. Even though the star schema prevails in current systems and is favored for its simplicity and improved performance, it is not adequate for handling complex dimensions. For instance, an attempt to store a heterogeneous hierarchy in a single relation would produce NULL-valued attributes in each tuple, resulting in unpredictable aggregation behavior. Therefore, we opt for the snowflake schema, capable of handling heterogeneity, shared dimension levels and explicit mapping of the relationships and their cardinalities in form of integrity constraints. According to the snowflake schema, each dimension category is stored in a separate relation, so that the following conditions hold:

a. Each category is represented by the same attribute set;
b. A relation may reference only its immediate parent categories;
c. The value of the reference to the parent category may not be NULL (Jagadisch et al., 1999).

The penalty of obeying the above rules of the normalized storage is the inability to map non-summarizable hierarchy types, such as *non-strict*, *non-covering*, and *non-onto*. Moreover, snowflake schema provides no guidelines for mapping the schemata of heterogeneous dimensions and their subclasses.

We propose a two-phase hierarchy transformation approach for obtaining a logical representation from the conceptual model:

1. **Schema normalization:** inspects the schemata of heterogeneous hierarchies and transforms them into a state that satisfies the summarizability constraints for heterogeneous dimensions, as defined by Hurtado et al. (2001).
2. **Instance normalization:** is about transforming homogeneous hierarchies of types *non-strict*, *non-covering*, and *non-onto* into a summarizable state.

Schema transformation is done prior to instance normalization. Since the latter has the task to normalize all homogeneous subdimensions in the dimension schema, it is imperative to have the final state of the schema at this point. As a preparation step, the fact schema of the case study is presented as a directed graph, using a slightly adjusted the notation of the DF model (dimensional fact model by Golfarelli, Maio, & Rizzi, 1998)), as shown in Figure 9.

Schema Normalization Techniques

Standard data warehouse systems do not provide methodological guidelines for managing heterogeneous dimensions. Obviously, the concerns of ensuring the summarizability and adequately mapping all aggregation paths remain valid in

Figure 9. University expenditures modeled as a 5-dimensional fact schema

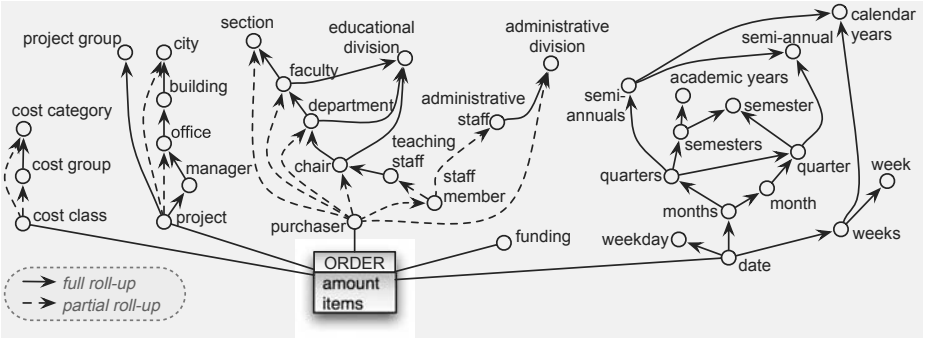
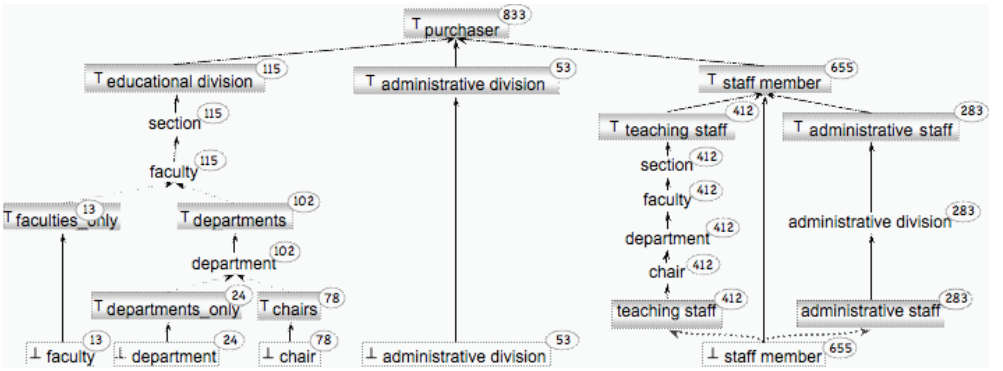


Figure 10. Reshaping a generalized hierarchy using subclasses as abstract root nodes



this context. To understand the requirements of supporting heterogeneity, we take a closer look at the categories of heterogeneous dimensions in Figure 6. We propose to distinguish between non-covering and generalized hierarchies. In the former, the members are simply allowed to skip levels, while generalization implies the existence of superclass/subclass relationships between schema nodes (i.e., there exist categories explicitly composed of or decomposed into multiple other categories (see the categories purchaser and staff member in Figure 9)). We expect different analysis requirements for these dimension types and propose to handle their modeling accordingly:

- **Con-covering:** hierarchy preserves its schema and is passed over to the instance normalization phase.
- **Generalized:** hierarchy is normalized by eliminating mixed granularity and overlapping specialization.

The only unavoidable case of having heterogeneous members in the same table is when the very bottom category is defined as a superclass. Since the entire dimension is referenced from within the fact table via the bottom category, it is imperative to have all members of the finest granularity in the referenced table. Figure 10 shows the dimension schema of purchaser with the bottom level resolved into its actual five categories. Attached to each node is the number of bottom-level members that roll up to that category.

The transformed schema of a generalized hierarchy is a result of applying the following rules:

- The most general superclass serves as the root category of type $\top_D$;
- Further superclasses are presented as normal categories;
- Subclasses are child categories of their superclass category;
- A non-bottom-level subclass category is of abstract type $\top_{D'}$ since it plays the role of an abstract root node for subdimension *D'*;
- A subclass is used repeatedly as the bottom category $\top_{D'}$ if it corresponds to the finest granularity in *D'*.

The heterogeneous schema is normalized by inspecting each subclass' hierarchy in a bottom-up fashion (i.e., from the most specialized subclasses toward the root.) The first irregularity to resolve is the non-strictness provoked by the overlapping subclasses, as is the case with staff member subtree. A disjoint specialization is obtained in the following steps:

1. A new subclass category is defined for each subset of members that cause the overlap, re-assigning those members this new category;
2. An additional "placeholder" subclass is defined, if the members in the superclass

are allowed not to roll up to any of the defined subclasses. For instance, to account for staff members who are neither teaching nor administrative, a subclass other staff is introduced.

Figure 11 shows the resulting schema of subdimension staff member in which the bottom level rolls-up to four disjoint subclasses.

A complex case of heterogeneity is a mixed generalized hierarchy, in which subclasses categories are also used as hierarchy levels, as observed in the schema of educational division. Categories faculty and department are, on the one hand, purchasers in their own right and, on the other hand, serve as aggregation levels for chair. Our approach to handling mixed granularity is a straightforward mapping of the twofold nature of the respective categories by introducing additional subclasses: each mixed-granularity category is viewed as a heterogeneous dimension sub-divided into a non-hierarchical and a hierarchical sub-dimension, corresponding to its respective two roles. Further, the general approach to modeling a generalized hierarchy is applied. The resulting schema for educational division is shown in Figure 10.

Instance Normalization Techniques

The categorization of dimension types depicted in Figure 6 is itself a hierarchy, and can thus be used as a decision tree for determining the properties of a particular dimension. Hierarchies of types *non-onto* and *non-covering*, as well as of type *non-strict* must undergo the instance normalization. Since any combination and even all of the above irregularities may occur in the same hierarchy, it is necessary to define the precedence in which the summarizability conditions are enforced.

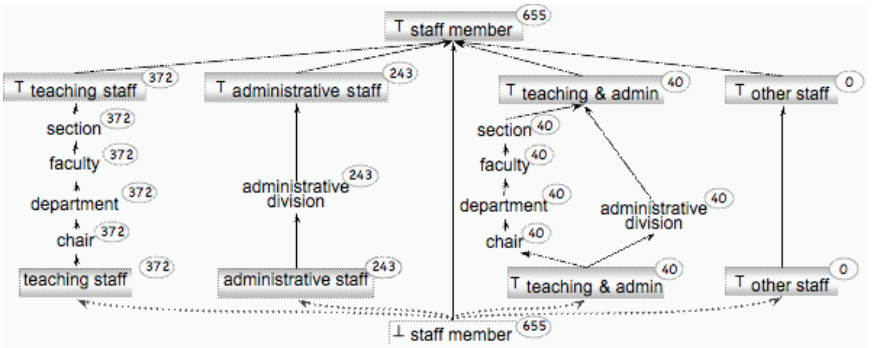
To identify all candidates for this transformation step, it is necessary to recursively decompose multiple hierarchies into their constituent simple hierarchies as well as to decompose heterogeneous dimensions into homogeneous subdimensions.

The result of the instance transformation is a symmetric hierarchy whose instance strictly adheres to its schema, or, formally, if for any two categories C_i of type C_i and C_j of type C_j such that $C_i \sqsubset C_j$ the following conditions hold:

1. **Covering:** $\forall\, e_1 \in C_i : \exists\, e_2 \in C_j \wedge e_1 < e_2$;
2. **Onto:** $\forall\, e_2 \in C_j : \exists\, e_1 \in C_i \wedge e_1 < e_2$;
3. **Strict:** $\forall\, e_1 \in C_i : (e_2, e_3 \in C_j \wedge e_1 < e_2 \wedge e_1 < e_3) \Rightarrow e_2 = e_3$.

Various transformation techniques for enforcing summarizability have been proposed in the literature (Malinowski et al., 2006). In general, the choice of a particular technique depends largely on the semantics behind the data at hand. If irregularity is caused by missing or imprecisely captured values and it is crucial to produce imprecision-aware queries and results (e.g., in clinical diagnosing or risk assessment), the approach of Pedersen et al. (2001), in which the original data remains denormalized and imprecision is made explicit to the user by providing a set of alternative

Figure 11. Eliminating overlap via additional subclasses in a specialization hierarchy



queries, may provide an adequate solution. However, if the data hierarchy is intrinsically irregular (e.g., the hierarchical structure of an organization, it needs to be normalized to ensure correct aggregate navigation.)

We adopt and modify the dimension transformation technique proposed by Pedersen et al. (1999). The original algorithm normalizes irregular hierarchies by enforcing the summarizability conditions in the order as listed above. In our approach, strictness is enforced prior to onto, as the hierarchy may become non-onto as a result of applying certain strictness enforcement approaches.

Our choice of the approaches for each of the transformation steps was driven by the considerations of minimality, transparence, and intuitiveness for the user.

Mapping to *covering* is traditionally achieved by inserting placeholder elements to fill the gaps of the missing parent nodes, as demonstrated at the example of the dimension project in Figure 12. If a placeholder element e_i is inserted at the highest aggregation level C_j (i.e., $e_i \in C_j, C_j \sqsubset \top_D$), the former may be named *"Others"*; otherwise, it makes sense to inherit the identifier of the parent node and derive the node's name from that of its parent.

Mapping to *strict* is realized by resolving multiple-parent relations, whereas applicability of a particular technique depends on the nature of non-strictness and users' requirements (Malinowski et al., 2006). We limit ourselves to naming five options, with an illustrating example in Figure 13:

1. If the accuracy of many-to-many relationships is not crucial for the analysis, a single "priority"-parent for each child member can be specified (grey edges in Figure 13 (b) are the ones to be ignored while aggregating).
2. Pedersen et al. (1999) propose a solution based on turning a non-strict hierarchy into multiple strict ones. A new category is inserted in-between the child and the parent category whenever a multi-parent relationship is encountered. This new category contains a "fused" value for each multi-parent, which is made a parent of the respective child values and a child of the parent values fused in it, as shown in Figure

Figure 12. Normalizing a non-covering hierarchy (left) by inserting placeholder elements

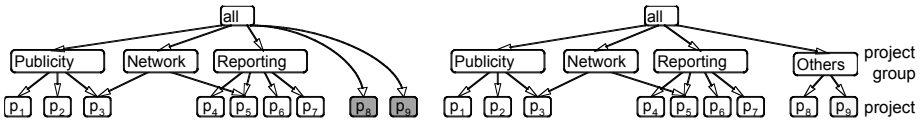
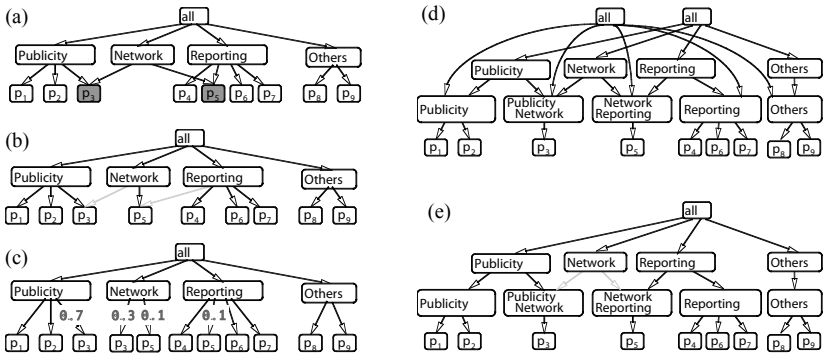


Figure 13. A non-strict hierarchy (a) and various normalization options



13 (d). Each of the alternative aggregation hierarchies produced thereby is strict.

3. The previous approach can be modified to avoid generation of multiple alternative hierarchies by inserting the new categories directly into the original hierarchy and specifying the "priority"-parent for aggregation, as demonstrated in Figure13(e).
4. A lossless transformation option with respect to preserving all child/parent relationships is to specify for each multi-parent element its degree of belonging to each of its parent elements, as shown in Figure 13 (c). A possible relational implementation for this option is to define a *bridge table* (Kimball & Ross, 2002), which stores each parent-child pair along with the specified degree of belonging. The resulting hierarchy is called *weighted non-strict* and it can ensure summarizability if the implementation of the query generation is adapted so as take these degrees into account when computing the measure aggregate whenever such weighted dimension is involved.

Mapping to *onto* is done in the last step since a symmetric hierarchy may become non-onto as a result of eliminating parent edges in the previous step, as in the case depicted in Figure 13(e). Interestingly, this last step may be omitted altogether in those cases where childless members in the hierarchy are guaranteed to have no associated fact data. Consider the subdimension administrative staff that rolls-up to administrative division. Even if there exists a division without staff in purchaser role, there cannot exist any facts about the orders of that division's staff. In those scenarios, in which mapping to onto is required, this can be done by inserting placeholder child values.

To reduce the overhead of managing "fake" members, inserted in the first and the last steps, we suggest that the inserted node inherits the identifier and the name (possibly with adjustments) of the parent value. For instance, if department *A* (id=14) has no chairs, a chair member with same id and named *A *all** is inserted into the category chair.

NAVIGATION FRAMEWORK

Users interact with multidimensional data in a predominantly "drill-down" fashion, starting with coarsely grained aggregates and descending gradually to the desired level of detail. The analysis task can thus be reduced to (a) selecting the measure and the aggregation function, (b) browsing to the desired granularity in dimensional hierarchies, and (c) filtering the data subset to display. A visual OLAP interface consists of two major components, the navigation tree for browsing through dimensional data, and the main window for displaying visually formatted query results. Selection of measures, functions, dimensional levels and values is done interactively.

The fact table is represented by cube icon at the top, containing the lists of DIMENSIONS, MEASURES, and FUNCTIONS. Each hierarchical dimension *D* is a folder containing its schema categories as nested subfolders, from the root category $\top_D$ at the top-level to the bottom category $\bot_D$. Non-abstract categories are supplied with a button for displaying their actual data. Figure 14 shows the navigational structure of our case study's data cube with a click on the data of calendar year.

The navigation structure of a dimension is a recursive nesting of sub-dimension nodes, where each node is used for drilling down to the respective granularity. The results of a drill-down are the sub-aggregates computed for each dimensional value. With respect to its underlying data hierarchy, the behavior of a sub-dimensional schema node can be reduced to the following types:

- **Non-hierarchical:** (i.e., the bottom level), displayed as a non-expandable page icon;
- **Simple hierarchy:** node is a folder containing a single child category;
- **Multiple hierarchies:** node contains a subfolder for each of the alternative paths. These paths are mutually exclusive

Figure 14. Fact table navigation

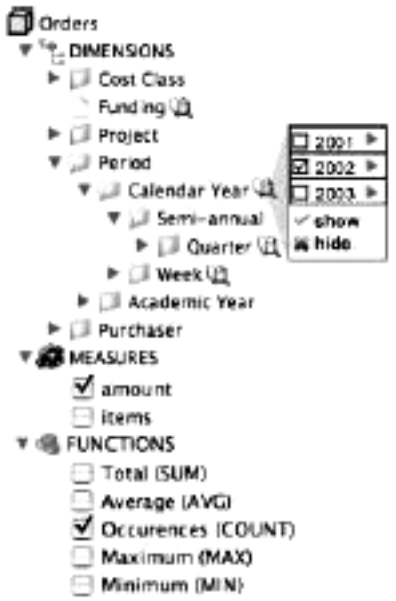
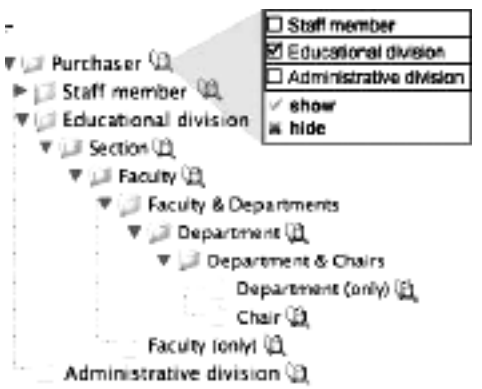


Figure 15. Navigation fragment of a heterogeneous dimension



in terms of navigation (i.e., once the user has selected one path), all others should be disabled for further interaction.

- **Superclass:** is a folder containing all subclass categories as subfolders. Since the superclass has no data of its own, the subclass categories are shown as its data items, so that a drill-down would produce the aggregates of the subclass categories. For instance, decomposing along the top-level node purchaser produces the aggregates of its three immediate subclasses educational division, administrative division, and staff member, as shown in Figure 15. Subclass node icons are colored differently and visually connected to each other to be distinguished from the case of multiple hierarchies.
- **Abstract Root:** node corresponds to the dimension's top category $\top_D$ and is used purely as a "wrapper" for the entire hierarchy underneath. Abstract root is superfluous in case of a non-hierarchy (nothing to "wrap") or a generalization hierarchy (abstract root already available).

The resulting navigation tree of mixed granularity nodes deserves special attention due to its non-triviality. Figure 15 displays the structure of the category section derived from the schema depicted in Figure 10. Mixed granularity of the category faculty is presented by a node Faculty at the top and its two subclasses Faculty & Departments and Faculty (only). Querying the top node produces the aggregate for each faculty, that includes the faculty's own orders and those of its departments. Abstract node Faculty & Departments can be used to decompose the aggregate into the parts of the subclasses (i.e., the departments and the faculty itself), and so on.

A prototype of the presented schema-based exploration approach for complex OLAP data has been implemented as a Java application that connects to a specified database and allows user to navigate in OLAP cubes presenting the results as a pivot table, business chart, or a more sophisticated visualization. The entire navigation hierarchy is generated from the metadata of the data warehouse. The prototype expects the metadata to be input in the XML format, modeled according to the OMG common warehouse metamodel (CWM) specification (OMG, 2003). At this stage, performance and scalability issues were left out of consideration.

An example of how the data from the case study is explored along the heterogeneous dimension purchaser is depicted in Figure 16. Enhanced decomposition tree visualization technique (Mansmann & Scholl, 2007) is used to present the results of a series of disaggregation steps in form of a decomposition hierarchy. The first level is obtained by selecting the top-level category purchaser, the second level is a drill-down into each of the three subclass categories, and the third level is obtained by decomposing section along faculty and decomposing each of the subclasses of staff member into their next levels. Notice the reuse of section and administrative division in different parts of the tree. Compact layout is achieved by mapping the aggregate values to the areas of the rectangular regions, so that the sub-aggregates across the whole tree can be compared visually by comparing the sizes of the rectangles.

CONCLUSION AND FUTURE WORK

Inspired by the growing demand for OLAP applications in novel domains and the challenges of dealing with complex dimensional hierarchies, not supported by the classical multidimensional model, we developed a framework for classifying and modeling complex data and its seamless mapping to a schema-based navigation tree of a visual analysis tool. Using a case study from the area of academic administration, we provided a categorization of hierarchy types and identified those that lead to non-summarizable behavior.

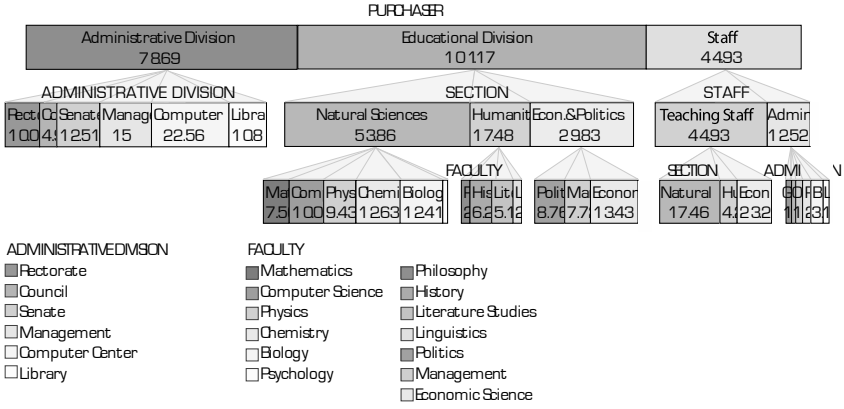
Our modeling approach in based on a two-phase transformation of irregular dimensions: (1) reshaping heterogeneous schemata into a set of well-behaved subdimensions, and (2) enforcing summarizability within each homogeneous subdimension. Our model does not introduce any query language extensions; it rather relies on the metadata (i.e., a standardized description of the fact-dimensional schema for mapping OLAP data to a visual browser and translating user interaction back to the database operations.)

Among our future research directions are to provide explicit handling of temporal and spatial aspects in modeling and querying OLAP data, to investigate the applicability of schema-based browsing for semi-structured and high-dimensional data, and to search for novel visualization and interaction techniques capable of presenting large volumes of complex data for explorative analysis.

REFERENCES

Codd, E. F., Codd, S. B., & Salley, C. T. (1993). Providing OLAP (online analytical processing)

Figure 16. Displaying the total amount disaggregated along the subclasses of purchaser in form of an enhanced decomposition tree



to user-analysts: An IT mandate. Technical report, *E.F. Codd & Associates*.

Golfarelli, M., Maio, D., & Rizzi, S. (1998). The dimensional fact model: A conceptual model for data warehouses. *International Journal of Cooperative Information Systems, 7*(2-3), 215-247.

Hümmer, W., Lehner, W., Bauer A., & Schlesinger, L. (2002). A decathlon in multidimensional modeling: Open issues and some solutions. *DaWaK 2000: Proceedings of 4th International Conference on Data Warehousing and Knowledge Discovery* (pp. 275-285). London, UK.

Hurtado, C. A., & Mendelzon, A. O. (2001). Reasoning about summarizability in heterogeneous multidimensional schemas. *ICDT 2001: Proceedings of 8th International Conference on Database Theory* (pp. 375-389). London, UK.

Hurtado, C. A., & Mendelzon, A. O. (2002). OLAP dimension constraints. In *PODS '02: Proceedings of 21st ACM Symposium on Principles of Database Systems* (pp. 169-179). Madison, USA.

Hurtado, C. A., Gutierrez, C., & Mendelzon, A. O. (2005). Capturing summarizability with integrity constraints in OLAP. *ACM Transactions on Database Systems*, *30*(3), 854-886.

Jagadish, H. V., Lakshmanan, L. V. S., & Srivastava, D. (1999). What can hierarchies do for data warehouses? *VLDB '99: Proceedings of 25th International Conference on Very Large Databases* (pp. 530-541). Edinburgh, Scotland, UK.

Kimball, R., & Ross, M. (2002). *The data warehouse toolkit: The complete guide to dimensional modeling*. New York, NY: John Wiley & Sons, Inc.

Lechtenbörger, J., & Vossen, G. (2003). Multidimensional normal forms for data warehouse design. *Information Systems, 28*(5), 415-434.

Lehner, W., Albrecht, J., & Wedekind, H. (1998). Normal forms for multidimensional databases. *SSDBM: Proceedings of 10th International Conference on Scientific and Statistical Database Management* (pp. 63-72). Capri, Italy,.

Lenz, H. J., & Shoshani, A. (1997). Summarizability in OLAP and statistical databases. *SSDBM: Proceedings of 9th International Conference on Scientific and Statistical Database Management* (pp. 132-143). Olympia, WA, USA.

Malinowski, E., & Zimányi, E. (2006). Hierarchies in a multidimensional model: From conceptual modeling to logical representation. *Data & Knowledge Engineering*, *59*(2), 348–377.

Malinowski, E., & Zimányi, E. (2004). OLAP hierarchies: A conceptual perspective. *CAiSE'04: Proceedings of 16th International Conference on Advanced Information Systems Engineering* (pp. 477-491). Riga, Latvia,.

Mansmann, S., & Scholl, M. H. (2007). Exploring OLAP aggregates with hierarchical visualization techniques. *SAC 2007: Proceedings of 22nd Annual ACM Symposium on Applied Computing, Multimedia, & Visualization Track* (pp. 1067-1073). Seoul, Korea.

Mansmann, S., & Scholl, M. H. (2006). Extending visual OLAP for handling irregular dimensional hierarchies. *DaWaK'06: Proceedings of 8th International Conference on Data Warehousing and Knowledge Discovery* (pp. 95-105). Krakow, Poland.

Niemi, T., Nummenmaa, J., & Thanisch, P. (2001). Logical multidimensional database design for ragged and unbalanced aggregation. *DMDW'2001: Proceedings of 3rd International Workshop on Design and Management of Data Warehouses* (pp. 7.1-7.8). Interlaken, Switzerland.

Object Model Group. (OMG). (2003). *Common warehouse metamodel (CWM) specification 1.1*. Retrieved January, 31 2006, from http://www.omg.org/cgi-bin/doc?formal/03-03-02

Pedersen, T. B., Jensen, C. S., & Dyreson, C. E. (2001). A foundation for capturing and querying complex multidimensional data. *Information Systems*, *26*(5), 383-423.

Pedersen, T. B., Jensen, C. S., & Dyreson, C. E. (1999). Extending practical pre-aggregation in online analytical processing. *VLDB'99: Proceedings of 25th International Conference on Very Large Databases* (pp. 663-674). Edinburgh, Scotland, UK.

Rafanelli, M., & Shoshani, A. (1990). STORM: A statistical object representation model. *SSDBM:*

Proceedings of 5th International Conference on Statistical and Scientific Database Management (pp. 14-29). Charlotte, NC, USA.

Sapia, C., Blaschka, M., Höfling, G., & Dinter, B. (1999). Extending the E/R model for the multidimensional paradigm. *Proceedings of the ER '98 Workshops on Data Warehousing and Data Mining* (pp. 105-116). Singapore.

Vinnik, S., & Mansmann, F. (2006). From analysis to interactive exploration: Building visual hierarchies from OLAP cubes. *EDBT 2006: Proceedings of 10th International Conference on Extending Database Technology* (pp. 496-514). Munich, Germany.

Zurek, T., & Sinnwell, M. (1999). Datawarehousing has more colours than just black & white. *VLDB'99: Proceedings of 25th International Conference on Very Large Databases* (pp. 726-729). Edinburgh, Scotland, UK.

Svetlana Mansmann received a diploma in international economic relations from the Belarusian State University, Belarus, in 1999, and an MSc degree in information engineering from the University of Konstanz (Germany) in 2003. She is currently pursuing a PhD degree in computer science, working as a research assistant in the Databases and Information Systems Group at the University of Konstanz, Germany. Previous research fields are decision support systems and e-Government. Recent publications focus on data warehousing and OLAP, multidimensional data modeling, business process intelligence, and visual exploration of multidimensional cubes. She received Best Paper Awards of ICECE 2005 (International Conference on Engineering and Computer Education) and DaWaK 2006 (International Conference on Data Warehousing and Knowledge Discovery). Mansmann has been an associate member of the Graduate College "Explorative Analysis and Exploration of Large Information Spaces" at the University of Konstanz (Germany) since 2004.

Marc H. Scholl received his MSc degree (Dipl.-Inform.) and PhD degree (Dr.-Ing.) both in computer science from the Technical University of Darmstadt, Germany, in 1982 and 1988, respectively. He is currently a full professor of computer science at the University of Konstanz, Germany. During the years 1998-2004 he served as vice-president of the University, responsible for the information infrastructure. Previous positions held include an associate professorship at the University of Ulm, Germany (1992-94) and an assistant professorship at ETH Zurich, Switzerland (1989-92). Current research topics include XML and databases, query processing & optimization, DBMS architecture, digital libraries, data warehouse applications and interfaces for decision support and e-Government applications. Scholl is a member of IEEE Computer Society, ACM and ACM SIGMOD, EDBT (currently EDBT President), the ICDT Council, and the German Computer Society (GI).