

Introduction

Organizations today are facing increasingly complex challenges in terms of management and problem solving in order to achieve their operational goals. This situation compels people in those organizations to utilize analysis tools that will better support their decisions. **Decision support systems** provide assistance to managers at various organizational levels for analyzing strategic information. These systems collect vast amount of data and reduce it to a form that can be used to analyze organizational behavior [54].

Since the early 1990s, **data warehouses** have been developed and deployed as an integral part of modern decision support systems [207, 272]. A data warehouse provides an infrastructure that enables users to obtain efficient and accurate responses to complex queries. Various systems and tools can be used for accessing and analyzing the data contained in data warehouses. For example, **online analytical processing (OLAP)** systems allow users to interactively query and automatically aggregate the data contained in a data warehouse. In this way, decision-making users can easily access the required information and analyze it at various levels of detail.

However, current data warehouse systems do not consider three aspects that are important for decision-making users.

The first aspect concerns the support for complex kinds of hierarchies. Current systems only allow simple hierarchies, typically those where the relationship between instances can be represented as a balanced tree. An example is a store–city–state hierarchy, where every store is related to a city, which in turn is related to a state. However, in many real-world situations the hierarchies correspond to unbalanced trees or to graphs. For example, depending on whether a client is a person or a company, different hierarchies may need to be considered, and these hierarchies may share some levels.

The second aspect relates to the inclusion of spatial data, which is not supported by current systems. In many situations, representing data spatially (e.g., on a map) could help to reveal patterns that are difficult to discover otherwise. For example, representing customers' locations on a map may help

the user to discover that most customers do not necessarily buy products in nearby stores.

The third aspect relates to the inclusion of temporal (or time-varying) data. Current data warehouse systems allow one to keep track of the evolution in time of organizational aspects under analysis (i.e., measures), such as the amount of sales or the quantities of items in an inventory. Nevertheless, there is a need to store historical data for the other elements contained in a data warehouse (i.e., dimensions and hierarchies) to better establish cause-effect relationships. For example, storing changes to the ingredients of products may help one to analyze whether those changes have any influence on sales.

Typical data warehouse systems are very complex, and their development is a difficult and costly task. Data warehouse developer teams must establish what data and what kind of analysis the users require for supporting the decision-making process. For this it is necessary to provide a model that facilitates communication between users and data warehouse designers. In addition, a suitable method should be followed to ensure successful data warehouse development.

In this chapter, we refer to various aspects that motivated us to develop a conceptual multidimensional model with spatial and temporal extensions and to propose a method for data warehouse design. First, in Sect. 1.1 we briefly describe some general concepts related to data warehouses and to their spatial and temporal extensions. We also refer to conceptual modeling and the method used for designing conventional, spatial, and temporal data warehouses. Then, Sect. 1.2 presents the motivations behind this book. Finally, in Sect. 1.3 we describe the scope of this book and the contributions to research contained in it, and in Sect. 1.4 we refer to its organization.

1.1 Overview

1.1.1 Conventional Data Warehouses

As a consequence of today's increasingly competitive and rapidly changing world, organizations in all sectors need to perform sophisticated data analysis to support their decision-making processes. However, traditional databases, usually called **operational** or **transactional databases**, do not satisfy the requirements for data analysis. These databases support daily business operations, and their primary concern is to ensure concurrent access by multiple users and recovery techniques that guarantee data consistency. Typical operational databases contain detailed data, do not include historical data, and, since they are usually highly normalized, they perform poorly when executing complex queries that involve a join of many tables or aggregate large volumes of data. Furthermore, when users need to analyze the behavior of an organization as a whole, data from several different operational systems must be

integrated. This can be a difficult task to accomplish because of the differences in data definition and content.

Data warehouses were proposed in order to better respond to the growing demands of decision-making users. A data warehouse is usually defined as a collection of subject-oriented, integrated, nonvolatile, and time-varying data to support management decisions [119]. This definition emphasizes some salient features of a data warehouse. **Subject-oriented** means that a data warehouse targets one or several subjects of analysis according to the analytical requirements of managers at various levels of the decision-making process. For example, a data warehouse in a retail company may contain data for the analysis of the purchase, inventory, and sales of products. **Integrated** expresses the fact that the contents of a data warehouse result from the integration of data from various operational and external systems. **Nonvolatile** indicates that a data warehouse accumulates data from operational systems for a long period of time. Thus, data modification and removal are not allowed in data warehouses: the only operation allowed is the purging of obsolete data that is no longer needed. Finally, **time-varying** underlines that a data warehouse keeps track of how its data has evolved over time; for instance, it may allow one to know the evolution of sales or inventory over the last several months or years.

Operational databases are typically designed using a **conceptual model**, such as the entity-relationship (ER) model, and **normalization** for optimizing the corresponding relational schema. Several authors (e.g., [147, 299]) have pointed out that these paradigms are not well suited for designing data warehouse applications. Data warehouses must be modeled in a way that ensures a better understanding of the data for analysis purposes and gives better performance for the complex queries needed for typical analysis tasks. In order to meet these expectations, a **multidimensional model** has been proposed.

The multidimensional model views data as consisting of facts linked to several dimensions. A **fact** represents a focus of analysis (for example analysis of sales in stores) and typically includes attributes called measures. **Measures** are usually numeric values that allow quantitative evaluation of various aspects of an organization to be performed. For example, measures such as the amount or quantity of sales might help to analyze sales activities in various stores. **Dimensions** are used to see the measures from different perspectives. For example, a time dimension can be used for analyzing changes in sales over various periods of time, whereas a location dimension can be used to analyze sales according to the geographic distribution of stores. Users may combine several different analysis perspectives (i.e., dimensions) according to their needs. For example, a user may require information about sales of computer accessories (the product dimension) in July 2006 (the time dimension) in all store locations (the store dimension). Dimensions typically include attributes that form **hierarchies**, which allow decision-making users to explore measures at various levels of detail. Examples of hierarchies are month–quarter–year in the time dimension and city–state–country in the location dimension. Aggre-

gation of measures takes place when a hierarchy is traversed. For example, moving in a hierarchy from a month level to a year level will yield aggregated values of sales for the various years.

The multidimensional model is usually represented by relational tables organized in specialized structures called **star schemas** and **snowflake schemas**. These relational schemas relate a fact table to several dimension tables. Star schemas use a unique table for each dimension, even in the presence of hierarchies, which yields denormalized dimension tables. On the other hand, snowflake schemas use normalized tables for dimensions and their hierarchies.

OLAP systems allow end users to perform dynamic manipulation and automatic aggregation of the data contained in data warehouses. They facilitate the formulation of complex queries that may involve very large amounts of data. This data is examined and aggregated in order to find patterns or trends of importance to the organization. OLAP systems have typically been implemented using two technologies: **relational OLAP (ROLAP)** and **multidimensional OLAP (MOLAP)**. The former stores data in a relational database management system, while the latter uses vendor-specific array data structures. **Hybrid OLAP (HOLAP)** systems combine both technologies, for instance using ROLAP for detailed fact data and MOLAP for aggregated data. Although current OLAP systems are based on a multidimensional model, i.e., they allow one to represent facts, measures, dimensions, and hierarchies, they are quite restrictive in the kinds of hierarchies that they can manage. This is an important drawback, since the specification of hierarchies in OLAP systems is important if one is to be able to perform automatic aggregation of measures while traversing hierarchies.

1.1.2 Spatial Databases and Spatial Data Warehouses

Over the years, **spatial data** has increasingly become part of operational and analytical systems in various areas, such as public administration, transportation networks, environmental systems, and public health, among others. Spatial data can represent either **objects** located on the Earth's surface, such as mountains, cities, and rivers, or geographic **phenomena**, such as temperature, precipitation, and altitude. Spatial data can also represent non-geographic data, i.e., data located in other spatial frames such as a human body, a house, or an engine. The amount of spatial data available is growing considerably owing to technological advances in areas such as remote sensing and global navigation satellite systems (GNSS), namely the Global Positioning System (GPS) and the forthcoming Galileo system.

The management of spatial data is carried out by **spatial databases** or **geographic information systems (GISs)**. Since the latter are used for storing and manipulating **geographic** objects and phenomena, we shall use the more general term “spatial databases” in the following. Spatial databases allow one to store spatial data whose location and shape are described in a two- or three-dimensional space. These systems provide a set of functions

and operators that allow users to query and manipulate spatial data. Queries may refer to spatial characteristics of individual objects, such as their area or perimeter, or may require complex operations on two or more spatial objects. **Topological relationships** between spatial objects, such as “intersection”, “inside”, and “meet”, are essential in spatial applications. For example, two roads may intersect, a lake may be inside a state, two countries may meet because they have a common border, etc. An important characteristic of topological relationships is that they do not change when the underlying space is distorted through rotation, scaling, and similar operations.

Although spatial databases offer sophisticated capabilities for the management of spatial data, they are typically targeted toward daily operations, similarly to conventional operational databases. Spatial databases are not well suited to supporting the decision-making process. As a consequence, a new field, called **spatial data warehouses**, has emerged as a combination of the spatial-database and data-warehouse technologies. As we have seen, data warehouses provide OLAP capabilities for analyzing data using several different perspectives, as well as efficient access methods for the management of high volumes of data. On the other hand, spatial databases provide sophisticated management of spatial data, including spatial index structures, storage management, and dynamic query formulation. Spatial data warehouses allow users to exploit the capabilities of both types of systems for improving data analysis, visualization, and manipulation.

1.1.3 Temporal Databases and Temporal Data Warehouses

Many applications require the storage, manipulation, and retrieval of **temporal data**, i.e., data that varies over time. These applications need to represent not only the changes to the data but also the time when these changes occurred. For example, land management systems need to represent the evolution of land distribution over time by keeping track of how and when land parcels have been split or merged, and when their owner changed. Similarly, healthcare systems may include historical information about patients’ clinical records, including medical tests and the time when they were performed.

Temporal databases allow one to represent and manage time-varying information. These databases typically provide two orthogonal types of temporal support. The first one, called **valid time**, indicates the time when data was (or is or will be) valid in the modeled reality, for example the time when a student took some specific course. The second one, called **transaction time**, specifies the time when a data item was current in a database, for example when the information that the student took the course was stored in the database. An important characteristic of temporal data is its **granularity**, or level of detail. Considering valid time, employees’ salaries may be captured at a granularity of a month, whereas stock exchange data may be captured at a granularity of a day or an hour. On the other hand, the granularity of transaction time is defined by the system, typically at a level of a millisecond.

Two approaches have been proposed for providing temporal support in relational databases, depending on whether modifications to the relational model are required or not. In one approach [276], temporality is represented in tables by means of hidden attributes, i.e., attributes that cannot be referenced by simple names in the usual way. Using this approach, temporal data is handled differently from traditional nontemporal data. Therefore, the various components of a database management system (DBMS) must be extended to support temporal semantics; this includes adequate storage mechanisms, specialized indexing methods, temporal query languages, etc. Another approach [55] does not involve any change to the classical relational model and considers temporal data just like data of any other kind, i.e., temporal support is implemented by means of explicit attributes. This approach introduces new interval data types and provides a set of new operators and extensions of existing ones for managing time-varying data.

Nevertheless, neither of the above approaches has been widely accepted, and current DBMSs do not provide yet facilities for manipulating time-varying data. Therefore, complex structures and significant programming effort are required for the correct management of data that varies over time.

Although temporal databases allow historical data to be managed, they do not offer facilities for supporting the decision-making process when aggregations of high volumes of historical data are required. Therefore, bringing together the research achievements in the fields of temporal databases and data warehouses has led to a new field, called **temporal data warehouses**. Temporal semantics forms an integral part of temporal data warehouses in a similar way to what is the case for temporal databases.

1.1.4 Conceptual Modeling for Databases and Data Warehouses

The conventional database design process includes the creation of database schemas at three different levels: conceptual, logical, and physical [67].

A **conceptual schema** is a concise description of the users' data requirements without taking into account implementation details. Conceptual schemas are typically expressed using the ER model [46] or the Unified Modeling Language (UML) [30].

A **logical schema** targets a chosen implementation paradigm, for example the relational, the object-oriented, or the object-relational paradigm. A logical schema is typically produced by applying a set of mapping rules that transform a conceptual schema. For example, a conceptual schema based on the ER model can be transformed into the relational model by applying well-known mapping rules (see, e.g., [67]).

Finally, a **physical schema** includes the definition of internal data structures, file organization, and indexes, among other things. The physical schema is highly technical and considers specific features of a particular DBMS in order to improve storage and performance.

In the database community, it has been acknowledged for several decades that conceptual models allow better communication between designers and users for the purpose of understanding application requirements. A conceptual schema is more stable than an implementation-oriented (logical) schema, which must be changed whenever the target platform changes [227]. Conceptual models also provide better support for visual user interfaces; for example, ER models have been very successful with users.

However, currently there is no well-established conceptual model for multidimensional data. There have been several proposals based on UML (e.g., [5, 170]), on the ER model (e.g., [266, 302]), and using specific notations (e.g., [83, 110, 254, 305]). Although these models include some features required for data warehouse applications, such as dimensions, hierarchies, and measures, they have several drawbacks, to which we shall refer in Sect. 1.2 and, in more detail, in Sect. 3.8.

1.1.5 A Method for Data Warehouse Design

The method used for designing operational databases includes well-defined phases. The first one, referred to as **requirements specification**, consists in gathering users' demands into a coherent and concisely written specification. The next phases, **conceptual design**, **logical design**, and **physical design**, consists in translating the requirements gathered in the first phase into database schemas that increasingly target the final implementation platform.

Since data warehouses are specialized databases aimed at supporting the decision-making process, the phases used in conventional database design have also been adopted for developing data warehouses. During the requirements-gathering process, users at various levels of management are interviewed to find out their analysis needs. The specification obtained serves as basis for creating a database schema capable of responding to users' queries. In many situations, owing to the lack of a well-accepted conceptual model for data warehouse applications, designers skip the conceptual-design phase and use instead a logical representation based on star and/or snowflake schemas. Afterwards, the physical design considers the facilities provided by current DBMSs for storing, indexing, and manipulating the data contained in the warehouse.

However, experience has shown that the development of data warehouse systems differs significantly from the development of conventional database systems. Therefore, modifications to the above-described method are necessary. For example, unlike conventional databases, the data in a warehouse is extracted from several source systems. As a consequence, some approaches for data warehouse design consider not only users' requirements but also the availability of data. Other approaches mainly take the underlying operational databases into account, instead of relying on users' requirements. Additionally, since in many cases the data extracted from the source systems must be transformed before being loaded into the data warehouse, it is necessary

to consider during data warehouse design the **extraction-transformation-loading (ETL)** process.

On the other hand, there is not yet a well-established method for designing spatial or temporal databases. The usual practice is to design a conventional database ignoring the spatial and temporal aspects and, later on, extend the conventional schema produced so far with spatial or temporal features. A similar situation arises for spatial and temporal data warehouses, although this is compounded by the fact that, being very recent research areas, much less experience is available to assist in developing them.

1.2 Motivation for the Book

In this section, we explain the reasons that motivated us to explore the field of conceptual modeling for data warehouse and OLAP applications. We also refer to the importance of including spatial and temporal data in a conceptual multidimensional model. Finally, we show the significance of defining a method for designing conventional, spatial, and temporal data warehouses.

The domain of conceptual design for data warehouse applications is still at a research stage. The analysis presented by Rizzi [262] shows the small interest of the research community in conceptual multidimensional modeling. Even though conceptual models are closer than logical models to the way users perceive an application domain, in the current state of affairs data warehouses are designed using mostly logical models. Therefore, when building data warehouses, users have difficulties in expressing their requirements, since specialized knowledge related to technical issues is required. Further, logical models limit users to defining only those elements that the underlying implementation systems can manage. As a typical example, users are constrained to use only the simple hierarchies that are implemented in current data warehouse tools.

Even though there are some conceptual models for data warehouse design, they either provide a graphical representation based on the ER model or UML with little or no formal definition, or provide only a formal definition without any graphical support. In addition, they do not allow users to express analysis requirements in an unambiguous way; in particular, they do not distinguish the various kinds of hierarchies that exist in real-world applications. This situation is considered as a shortcoming of existing models for data warehouses [104].

Furthermore, although location information, such as address, city, or state, is included in many data warehouses, this information is usually represented in an alphanumeric manner. Representing location information as spatial data leads to the emerging field of spatial data warehouses. However, this field raises several research issues.

First, there is no consensus in the literature about the meaning of a spatial data warehouse. The term is used in several different situations, for example

when there are high volumes of spatial data, when it is required to integrate or aggregate spatial data, or when the decision-making process uses spatial data. Even though it is important to consider all the above aspects in spatial data warehouses, what is still missing is a proposal for spatial data warehouses with the clearly distinguished spatial elements required for multidimensional modeling. Such a proposal should take account of the fact that although both spatial databases and spatial data warehouses manage spatial data, their purposes are different. Spatial databases are used for answering queries that involve spatial locations. Examples of such queries are “where is the closest store to my house?”, “which highways connect Brussels and Warsaw?”, and “how do I get to a specific place from my current position given by a GPS device?” In contrast, spatial data warehouses use spatial data to support the decision-making process. Examples of analytical queries are “what is the best location for a new store?”, “what roads should be constructed to relieve traffic congestion?”, and “how can we divide a specific area into sales regions to increase sales?”.

Second, applications that include spatial data are usually complex and need to be modeled taking user requirements into account. However, the lack of a conceptual approach for data warehouse and OLAP systems, joined with the absence of a commonly accepted conceptual model for spatial applications, makes the modeling task difficult. Further, although some existing conceptual models for spatial databases allow relationships among spatial objects to be represented, they are not adequate for multidimensional modeling, since they do not include the concepts of dimensions, hierarchies, and measures. To the best of our knowledge, there is as yet no proposal for a conceptual model for spatial-data-warehouse applications.

Finally, any proposal for a conceptual multidimensional model including spatial data should take account of various aspects not present in conventional multidimensional models, such as topological relationships and aggregation of spatial measures, among others. Some of these aspects are briefly mentioned in the existing literature, for example spatial aggregation, while others are neglected, for example the influence on aggregation procedures of topological relationships between spatial objects forming hierarchies.

Another important characteristic of data warehouses is that data is stored for long periods of time. In accordance with the “time-varying” and “non-volatility” features of data warehouses (see Sect. 1.1.1), changes to data cannot overwrite existing values. However, although the usual multidimensional models allow changes in measures to be represented, they are not able to represent changes in dimension data. Consequently, the features of data warehouse mentioned above apply only to measures, leaving the representation of changes in dimensions to the applications.

Several solutions have been proposed for representing changes to dimension data in the context of relational databases, called **slowly changing dimensions** [146]. The first solution, called type 1 or the overwrite model, corresponds to the usual situation where, upon an update, old dimension data

is replaced by new data, thus losing track of data changes. The second solution, called type 2 or the conserving-history model, inserts a new record each time the value of a dimension attribute changes. However, this approach is complex, since it requires the generation of different keys for records representing the same instance. It also introduces unnecessary data redundancy, since the values of attributes that have not changed are repeated. Finally, this solution also demands significant programming effort for querying time-varying dimension data. The last solution, called type 3 or the limited-history model, introduces an additional column for every attribute for which changes in value must be kept. These columns store the old and new values of the attribute. In this case the history of changes is limited to the number of additional columns. As can be seen, these solutions are not satisfactory, since they either do not preserve the entire history of the data or are complex to implement. Further, they do not take account of research related to managing time-varying information in temporal databases. Therefore, temporal data warehouses were proposed as a solution to the above problems.

Currently, the research related to temporal data warehouses raises many issues, mostly dealing with implementation aspects, such as special aggregation procedures, special storage, and indexing methods, among others. However, very little attention has been given by the research community to conceptual modeling for temporal data warehouses and to the analysis of what temporal support should be included in these systems. Such analysis should take account of the fact that temporal databases and conventional data warehouses have some similarities, in particular both of them manage historical data, but should also take into account their differences. For example, data in data warehouses is integrated from existing source systems, whereas data in temporal databases is introduced by users in the context of operational applications; data warehouses support the decision-making process, whereas temporal databases reflect data changes in reality and in the database content; and data in data warehouses is neither modified nor deleted¹ but in contrast, users of temporal databases change data directly. Therefore, it is necessary to propose some form of temporal support adequate for data warehouse applications, and a conceptual model with time-varying elements that takes into account the specific semantics of data warehouses.

The development of conventional, spatial, and temporal data warehouses is a very complex task that requires several phases for its realization. It is therefore necessary to specify a method in order to assist developers in the execution of these phases. Existing methods offer a variety of solutions, especially for the requirements specification phase. However, such a diversity of approaches can be confusing for designers. For example, one approach first considers users' requirements and then checks them against the availability of data in the source systems, while another approach first develops the data

¹ We ignore modifications due to errors during data loading, and deletions for purging data in data warehouses.

warehouse schema on the basis of the underlying source systems and then adapts it to users' requirements. It is not clear in which situations it is better to use either one of these approaches. As a consequence, many data warehouse projects do not give much importance to the requirements specification phase, and focus instead on more technical aspects [222]. This may lead to a situation where the data warehouse may not meet users' expectations for supporting the decision-making process. In addition, some of the existing proposals include additional phases corresponding to particular characteristics of the conceptual multidimensional model in use. In this way, these methods impose on designers a conceptual model that may be unknown to them and may also be inadequate for a particular data warehouse project.

Moreover, even though many organizations may require a data warehouse for the purpose of enhancing their decision-making processes, they might not be able to afford external teams or consultants to build it. Nevertheless, these organization will typically have in them professionals highly skilled in the development of traditional operational databases, although inexperienced in the development of data warehouses. Therefore, providing a methodological framework for data warehouse development that is based on the one used for traditional databases may facilitate this transition process.

Finally, since spatial and temporal data warehouses are new research fields, there is not yet a methodological approach for developing them. However, it is well known that the inclusion of spatial data increases the power of the decision-making process by expanding the scope of the analysis and by providing enhanced visualization facilities. Similarly, temporal support expands the scope of the analysis by allowing all elements of a multidimensional model to be time-varying. Therefore, any proposed method should specify how and when spatial and temporal support may be included, considering all the phases required for the development of spatial and temporal data warehouses.

1.3 Objective of the Book and its Contributions to Research

The general objective of this book was to define a multidimensional conceptual model and an associated design method for data warehouse applications manipulating conventional, spatial, and temporal data. Our conceptual model, called MultiDim, includes both graphical and textual notation, as well as a formal definition.

This book summarizes original research conducted by the authors, which has been partly published previously in international journals, conference proceedings, and book chapters [174, 175, 176, 177, 178, 179, 180, 181, 182, 183, 184, 186]. Since the book covers several research areas, we describe next the contributions of this research to each of them.

1.3.1 Conventional Data Warehouses

The main contributions of our proposal in the field of conventional data warehouses include the following aspects.

The MultiDim model is based on well-known ER constructs, i.e., entity types, relationship types, and attributes, with their usual semantics. Therefore, it allows designers to apply the same modeling constructs as those used for operational database design, and it provides a conceptual representation independent of technical details. The model considers a multidimensional view of data in which the various elements of the model, such as facts, measures, dimensions, and hierarchies, can be clearly distinguished in a visual manner, thus facilitating the representation of the data required for data warehouse and OLAP applications.

Our proposal extends the ER model by including various kinds of hierarchies that exist in real-world applications. We classify these hierarchy kinds, taking into account their differences at the schema and instance levels. Since current data warehouse and OLAP systems support only a limited number of hierarchy kinds, our proposal has important implications for users, designers, and implementers. It will give users a better understanding of the data to be analyzed and help designers to build conceptual schemas for data warehouse applications. Additionally, the proposed classification will provide OLAP tool implementers with the requirements needed by business users for extending the functionality of current OLAP tools. The mappings to the relational and object-relational models show that the various hierarchy kinds can be implemented in current DBMSs, regardless of the fact that some of them are considered as advanced features of a multidimensional model [299].

Our proposal contains additional contributions related to the methodological framework proposed for conventional-data-warehouse design.

Despite the apparent similarity of the various phases proposed for designing databases and data warehouses, to account for the specificities of data warehouse design we both modify the content of these phases and include additional ones. In particular, on the basis of existing proposals, we consider three approaches to the requirements specification phase. We discuss the general advantages and disadvantages of these approaches, and include guidelines for choosing one of them depending on the characteristics of the users, the developer team, and the source systems. Further, we include an additional phase for the data extraction, transformation, and loading processes.

The proposed method for data warehouse design will provide the developer team with precise guidance on the required phases and their content. In addition, a comprehensive method combining various approaches will allow the most appropriate approach to be chosen, taking into consideration the experience of the developer team and the particularities of the data warehouse project. Since the proposed method is independent of any conceptual model, logical database model, or target implementation platform, it may be useful to

both database developers who want to acquire basic knowledge for developing a data warehouse and to experienced data warehouse developers.

1.3.2 Spatial Data Warehouses

We extend the MultiDim model by the inclusion of spatial support. The novelty of our approach consists in the following aspects.

The multidimensional model is seldom used for representing spatial data even though this model provides a concise and organized representation for spatial data warehouses [19] and facilitates the delivery of data for spatial-OLAP systems. Our proposal provides spatial support for the various elements of the multidimensional model, i.e., facts, measures, dimensions, and hierarchies. In particular, the model includes a new feature, called the spatial fact relationship, which allows the topological relationships between spatial dimensions to be represented.

We also extend the previous classification of hierarchies and show its applicability to spatial hierarchies. We study how the topological relationships between spatial objects in a hierarchy influence the aggregation of measures. This gives insights to spatial-OLAP implementers for determining when measure aggregation can be done safely and when special aggregation procedures must be developed.

We provide a mapping for translating our spatially extended conceptual multidimensional model into the object-relational model. In this way, we show that schemas created using our model can be implemented in general-purpose DBMSs that include spatial extensions.

Finally, our proposal extends the method for conventional-data-warehouse design by taking into account the inclusion of spatial support in the various elements composing the multidimensional model. This method differs from the method proposed for spatial-database design, since it considers whether the users have knowledge of the analysis and manipulation of spatial data and whether the source systems include spatial data.

1.3.3 Temporal Data Warehouses

We also consider the inclusion of temporal support in the MultiDim model. The proposed temporal extension is based on previous research on temporal databases but takes account of the semantic differences between temporal data warehouses and temporal databases. Our proposal contributes in several ways to the field of temporal data warehouses.

We include in the MultiDim model several different temporality types that may be required for data warehouse applications but are currently ignored in research work. These temporality types allow us to extend the multidimensional model by including temporal support for dimensions, hierarchies, and measures. Therefore, we incorporate temporal semantics as an integral part of a conceptual multidimensional model. This should help users and designers

to express precisely which elements of a data warehouse they want to be time-invariant and for which elements they want to keep track of changes over time. The proposed temporal extension allows changes to dimension data to be expressed in the same way as changes to measures, which cannot be achieved by current multidimensional models. An important consequence of this is that the temporal dimension is no longer required to indicate the validity of measures. We also show the usefulness of having several different temporality types for measures. This aspect is currently ignored in research work, which considers only valid time.

As for conventional and spatial data warehouses, we propose a mapping of temporal data warehouses to the object-relational model. Since current DBMSs are not temporally enhanced (even though some proposals exist, e.g., [55, 276]), this mapping will help implementers who use the MultiDim model for designing temporal data warehouses.

Finally, we propose a method for temporal-data-warehouse design that extends the method described for conventional-data-warehouse design. We refer to various cases, considering users' requirements for temporal support and the availability of historical data in the source systems.

1.4 Organization of the Book

Figure 1.1 illustrates the overall structure of the book and the interdependencies between the chapters. Readers may refer to this figure to tailor their use of this book to their own particular interests. Various paths may be followed, depending on whether the reader is interested in conventional, spatial, or temporal data warehouses. Readers interested only in conventional data warehouses may read the chapters following the path without labels, i.e., Chaps. 1, 2, 3, 6, and 8. Chapter 2 could be skipped by readers proficient in the database and data warehouse domains. On the other hand, readers interested in spatial or temporal data warehouses should include in addition Chaps. 4 and 7 or Chaps. 5 and 7, respectively.

We briefly describe next the remaining chapters of the book.

Chapter 2 provides an introduction to the fields of databases and data warehouses, covering the conceptual, logical, and physical design. This chapter thus provides the necessary background for the rest of the book. We provide also an introduction to two representative data warehouse platforms, Microsoft Analysis Services 2005 and Oracle 10g with the OLAP option.

Chapter 3 defines the MultiDim model, including the various kinds of hierarchies and their classification. For each of them, graphical notations and mappings to relational and object-relational databases are given.

Chapter 4 refers to the spatial extension of the MultiDim model, considering both the conceptual and the logical level. By means of examples, we discuss the inclusion of spatial characteristics in the various elements forming a multidimensional model, such as dimensions, hierarchies, and measures. Further,

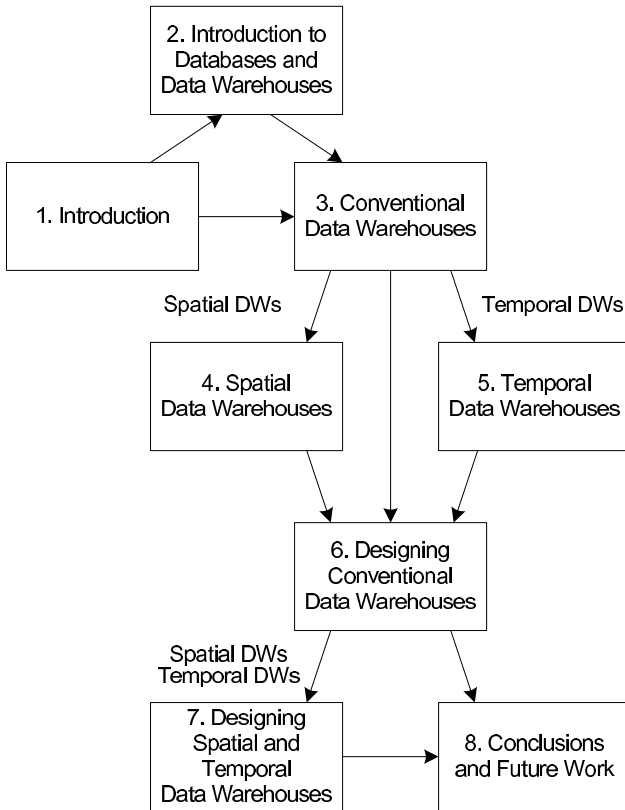


Fig. 1.1. Relationships between the chapters of this book

we study and classify the topological relationships that may exist between spatial levels forming a hierarchy, indicating when aggregation of measures can be done safely and when it requires special handling. We also present how the semantics can be preserved when a conceptual model is transformed to a logical model.

Chapter 5 presents the temporal extension of the MultiDim model. We propose the inclusion of various types of temporal support that are required for analysis purposes. Then, we discuss the inclusion of temporal support for dimensions and measures, taking into account the various roles that they play in a multidimensional model. We also consider the case where the time granularities in the source systems and the data warehouse are different. A logical representation of our conceptual model is also presented using an object-relational approach.

Chapter 6 specifies a method for conventional-data-warehouse design. Our proposal is in line with traditional database design methods that include the phases of requirements specification, conceptual design, logical design, and

physical design. We describe three different approaches to the requirements specification phase, giving recommendations on when to use each of them. Then, we refer to conceptual, logical, and physical design, considering both data warehouse structures and extraction-transformation-loading processes.

Chapter 7 extends the proposed method for conventional-data-warehouse design by including spatial and temporal data. We refer to all phases used for conventional-data-warehouse design and modify the three approaches used for requirements specification. For each of these approaches, we propose two solutions, depending on whether spatial or temporal support is considered at the beginning of the requirements-gathering process or at a later stage, once a conceptual schema has been developed.

Finally, Chap. 8 summarizes the results contained in the book and indicates some directions for future research.

Appendix A presents the formalization of the MultiDim model, dealing with conventional, spatial, and temporal aspects. This appendix defines the syntax, semantics, and semantic constraints of the model.

Appendix B summarizes the notation used in this book for representing the constructs of the ER, relational, and object-relational models, as well as for conventional, spatial, and temporal data warehouses.

Review Questions

- 1.1 Why are traditional databases inappropriate for data analysis?
- 1.2 Discuss four main characteristics of data warehouses.
- 1.3 Describe the different components of a multidimensional model, i.e., facts, measures, dimensions, and hierarchies.
- 1.4 What is the purpose of online analytical processing (OLAP) systems and how are they related to data warehouses?
- 1.5 What is spatial data? Give an example of an application that manipulates spatial data.
- 1.6 Explain the differences between spatial databases and spatial data warehouses.
- 1.7 What is temporal data? Describe different types of temporal support needed by applications.
- 1.8 Do current data warehouses manipulate temporal data?
- 1.9 Describe the different levels of schemas that are used for designing databases.
- 1.10 Explain the advantages of using a conceptual model when designing a data warehouse.
- 1.11 Specify the different steps used for designing a database.
- 1.12 Why do we need a method for data warehouse design?



<http://www.springer.com/978-3-540-74404-7>

Advanced Data Warehouse Design
From Conventional to Spatial and Temporal Applications
Malinowski, E.; Zimanyi, E.
2008, XXII, 444 p. 152 illus., Hardcover
ISBN: 978-3-540-74404-7