ISBNs: 0-471-41254-6 (Hardback); 0-471-22162-5 (Electronic)

CHAPTER 2

DATA WAREHOUSE: THE BUILDING BLOCKS

CHAPTER OBJECTIVES

- Review formal definitions of a data warehouse
- Discuss the defining features
- Distinguish between data warehouses and data marts
- Study each component or building block that makes up a data warehouse
- Introduce metadata and highlight its significance

As we have seen in the last chapter, the data warehouse is an information delivery system. In this system, you integrate and transform enterprise data into information suitable for strategic decision making. You take all the historic data from the various operational systems, combine this internal data with any relevant data from outside sources, and pull them together. You resolve any conflicts in the way data resides in different systems and transform the integrated data content into a format suitable for providing information to the various classes of users. Finally, you implement the information delivery methods.

In order to set up this information delivery system, you need different components or building blocks. These building blocks are arranged together in the most optimal way to serve the intended purpose. They are arranged in a suitable architecture. Before we get into the individual components and their arrangement in the overall architecture, let us first look at some fundamental features of the data warehouse.

Bill Inmon, considered to be the father of Data Warehousing provides the following definition: "A Data Warehouse is a subject oriented, integrated, nonvolatile, and time variant collection of data in support of management's decisions."

Sean Kelly, another leading data warehousing practitioner defines the data warehouse in the following way. The data in the data warehouse is:

Separate

Available

Integrated
Time stamped
Subject oriented
Nonvolatile
Accessible

DEFINING FEATURES

Let us examine some of the key defining features of the data warehouse based on these definitions. What about the nature of the data in the data warehouse? How is this data different from the data in any operational system? Why does it have to be different? How is the data content in the data warehouse used?

Subject-Oriented Data

In operational systems, we store data by individual applications. In the data sets for an order processing application, we keep the data for that particular application. These data sets provide the data for all the functions for entering orders, checking stock, verifying customer's credit, and assigning the order for shipment. But these data sets contain only the data that is needed for those functions relating to this particular application. We will have some data sets containing data about individual orders, customers, stock status, and detailed transactions, but all of these are structured around the processing of orders.

Similarly, for a banking institution, data sets for a consumer loans application contain data for that particular application. Data sets for other distinct applications of checking accounts and savings accounts relate to those specific applications. Again, in an insurance company, different data sets support individual applications such as automobile insurance, life insurance, and workers' compensation insurance.

In every industry, data sets are organized around individual applications to support those particular operational systems. These individual data sets have to provide data for the specific applications to perform the specific functions efficiently. Therefore, the data sets for each application need to be organized around that specific application.

In striking contrast, in the data warehouse, data is stored by subjects, not by applications. If data is stored by business subjects, what are business subjects? Business subjects differ from enterprise to enterprise. These are the subjects critical for the enterprise. For a manufacturing company, sales, shipments, and inventory are critical business subjects. For a retail store, sales at the check-out counter is a critical subject.

Figure 2-1 distinguishes between how data is stored in operational systems and in the data warehouse. In the operational systems shown, data for each application is organized separately by application: order processing, consumer loans, customer billing, accounts receivable, claims processing, and savings accounts. For example, *Claims* is a critical business subject for an insurance company. Claims under automobile insurance policies are processed in the Auto Insurance application. Claims data for automobile insurance is organized in that application. Similarly, claims data for workers' compensation insurance is organized in the Workers' Comp Insurance application. But in the data warehouse for an insurance company, claims data are organized around the subject of claims and not by individual applications of Auto Insurance and Workers' Comp.

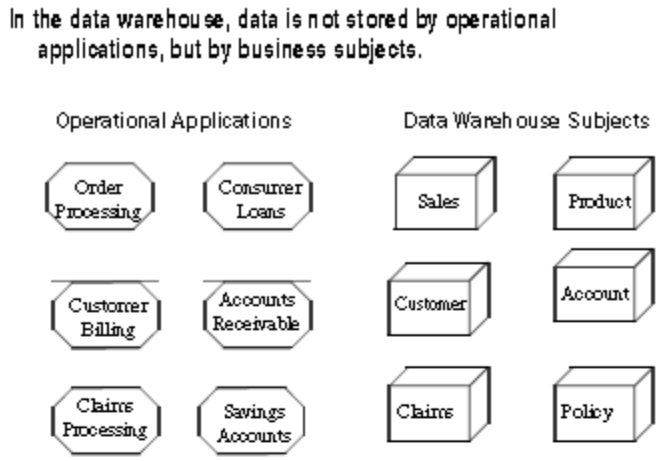


Figure 2-1 The data warehouse is subject oriented.

In a data warehouse, there is no application flavor. The data in a data warehouse cut across applications.

Integrated Data

For proper decision making, you need to pull together all the relevant data from the various applications. The data in the data warehouse comes from several operational systems. Source data are in different databases, files, and data segments. These are disparate applications, so the operational platforms and operating systems could be different. The file layouts, character code representations, and field naming conventions all could be different.

In addition to data from internal operational systems, for many enterprises, data from outside sources is likely to be very important. Companies such as Metro Mail, A. C. Nielsen, and IRI specialize in providing vital data on a regular basis. Your data warehouse may need data from such sources. This is one more variation in the mix of source data for a data warehouse.

Figure 2-2 illustrates a simple process of data integration for a banking institution. Here the data fed into the subject area of *account* in the data warehouse comes from three different operational applications. Even within just three applications, there could be several variations. Naming conventions could be different; attributes for data items could be different. The account number in the Savings Account application could be eight bytes long, but only six bytes in the Checking Account application.

Before the data from various disparate sources can be usefully stored in a data warehouse, you have to remove the inconsistencies. You have to standardize the various data elements and make sure of the meanings of data names in each source application. Before moving the data into the data warehouse, you have to go through a process of transformation, consolidation, and integration of the source data.

Data inconsistencies are removed; data from diverse operational applications is integrated.

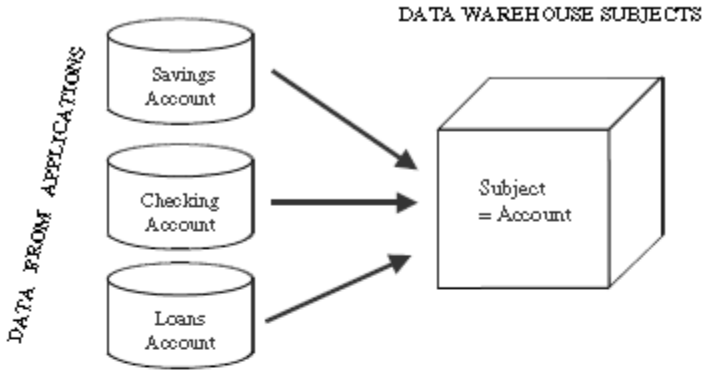


Figure 2-2 The data warehouse is integrated.

Here are some of the items that would need standardization:

- Naming conventions
- Codes
- Data attributes
- Measurements

Time-Variant Data

For an operational system, the stored data contains the *current* values. In an accounts receivable system, the balance is the current outstanding balance in the customer's account. In an order entry system, the status of an order is the current status of the order. In a consumer loans application, the balance amount owed by the customer is the current amount. Of course, we store some past transactions in operational systems, but, essentially, operational systems reflect current information because these systems support day-to-day current operations.

On the other hand, the data in the data warehouse is meant for analysis and decision making. If a user is looking at the buying pattern of a specific customer, the user needs data not only about the current purchase, but on the past purchases as well. When a user wants to find out the reason for the drop in sales in the North East division, the user needs all the sales data for that division over a period extending back in time. When an analyst in a grocery chain wants to promote two or more products together, that analyst wants sales of the selected products over a number of past quarters.

A data warehouse, because of the very nature of its purpose, has to contain historical data, not just current values. Data is stored as snapshots over past and current periods. Every data structure in the data warehouse contains the time element. You will find histor-

ical snapshots of the operational data in the data warehouse. This aspect of the data warehouse is quite significant for both the design and the implementation phases.

For example, in a data warehouse containing units of sale, the quantity stored in each file record or table row relates to a specific time element. Depending on the level of the details in the data warehouse, the sales quantity in a record may relate to a specific date, week, month, or quarter.

The time-variant nature of the data in a data warehouse

- Allows for analysis of the past
- Relates information to the present
- Enables forecasts for the future

Nonvolatile Data

Data extracted from the various operational systems and pertinent data obtained from outside sources are transformed, integrated, and stored in the data warehouse. The data in the data warehouse is not intended to run the day-to-day business. When you want to process the next order received from a customer, you do not look into the data warehouse to find the current stock status. The operational order entry application is meant for that purpose. In the data warehouse, you keep the extracted stock status data as snapshots over time. You do not update the data warehouse every time you process a single order.

Data from the operational systems are moved into the data warehouse at specific intervals. Depending on the requirements of the business, these data movements take place twice a day, once a day, once a week, or once in two weeks. In fact, in a typical data warehouse, data movements to different data sets may take place at different frequencies. The changes to the attributes of the products may be moved once a week. Any revisions to geographical setup may be moved once a month. The units of sales may be moved once a day. You plan and schedule the data movements or data loads based on the requirements of your users.

As illustrated in Figure 2-3, every business transaction does not update the data in the data warehouse. The business transactions update the operational system databases in real time. We add, change, or delete data from an operational system as each transaction happens but do not usually update the data in the data warehouse. You do not delete the data in the data warehouse in real time. Once the data is captured in the data warehouse, you do not run individual transactions to change the data there. Data updates are commonplace in an operational database; not so in a data warehouse. The data in a data warehouse is not as volatile as the data in an operational database is. The data in a data warehouse is primarily for query and analysis.

Data Granularity

In an operational system, data is usually kept at the lowest level of detail. In a point-of-sale system for a grocery store, the units of sale are captured and stored at the level of units of a product per transaction at the check-out counter. In an order entry system, the quantity ordered is captured and stored at the level of units of a product per order received from the customer. Whenever you need summary data, you add up the individual transac-

Usually the data in the data warehouse is not updated or deleted.

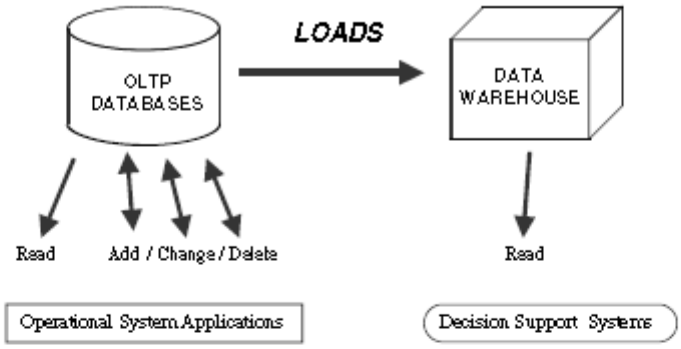


Figure 2-3 The data warehouse is nonvolatile.

tions. If you are looking for units of a product ordered this month, you read all the orders entered for the entire month for that product and add up. You do not usually keep summary data in an operational system.

When a user queries the data warehouse for analysis, he or she usually starts by looking at summary data. The user may start with total sale units of a product in an entire region. Then the user may want to look at the breakdown by states in the region. The next step may be the examination of sale units by the next level of individual stores. Frequently, the analysis begins at a high level and moves down to lower levels of detail.

In a data warehouse, therefore, you find it efficient to keep data summarized at different levels. Depending on the query, you can then go to the particular level of detail and satisfy the query. Data granularity in a data warehouse refers to the level of detail. The lower the level of detail, the finer the data granularity. Of course, if you want to keep data in the lowest level of detail, you have to store a lot of data in the data warehouse. You will have to decide on the granularity levels based on the data types and the expected system performance for queries. Figure 2-4 shows examples of data granularity in a typical data warehouse.

DATA WAREHOUSES AND DATA MARTS

If you have been following the literature on data warehouses for the past few years, you would, no doubt, have come across the terms "data warehouse" and "data mart." Many who are new to this paradigm are confused about these terms. Some authors and vendors use the two terms synonymously. Some make distinctions that are not clear enough. At this point, it would be worthwhile for us to examine these two terms and take our position.

Writing in a leading trade magazine in 1998, Bill Inmon stated, "The single most important issue facing the IT manager this year is whether to build the data warehouse first

THREE DATA LEVELS IN A BANKING DATA WAREHOUSE

Daily Detail	Monthly Summary	Quarterly Summary
Account	Account	Account
Activity Date	Month	Month
Amount	Number of transactions	Number of transactions
Deposit/Withdrawal	Withdrawals	Withdrawals
	Deposits	Deposits
	Beginning Balance	Beginning Balance
	Ending Balance	Ending Balance

Data granularity refers to the level of detail. Depending on the requirements, multiple levels of detail may be present. Many data warehouses have at least dual levels of granularity.

Figure 2-4 Data granularity.

or the data mart first." This statement is true even today. Let us examine this statement and take a stand.

Before deciding to build a data warehouse for your organization, you need to ask the following basic and fundamental questions and address the relevant issues:

- Top-down or bottom-up approach?
- Enterprise-wide or departmental?
- Which first—data warehouse or data mart?
- Build pilot or go with a full-fledged implementation?
- Dependent or independent data marts?

These are critical issues requiring careful examination and planning.

Should you look at the big picture of your organization, take a top-down approach, and build a mammoth data warehouse? Or, should you adopt a bottom-up approach, look at the individual local and departmental requirements, and build bite-size departmental data marts?

Should you build a large data warehouse and then let that repository feed data into local, departmental data marts? On the other hand, should you build individual local data marts, and combine them to form your overall data warehouse? Should these local data marts be independent of one another? Or, should they be dependent on the overall data warehouse for data feed? Should you build a pilot data mart? These are crucial questions.

How are They Different?

Let us take a close look at Figure 2-5. Here are the two different basic approaches: (1) overall data warehouse feeding dependent data marts, and (2) several departmental or lo-

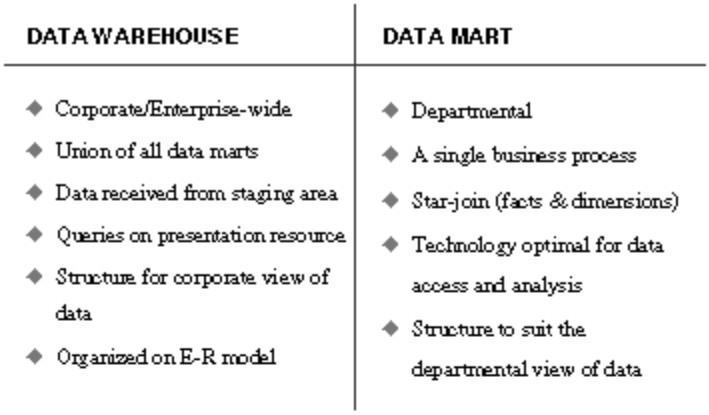


Figure 2-5 Data warehouse versus data mart.

cal data marts combining into a data warehouse. In the first approach, you extract data from the operational *systems*; you then transform, clean, integrate, and keep the data in the data warehouse. So, which approach is best in your case, the top-down or the bottom-up approach? Let us examine these two approaches carefully.

Top-Down Versus Bottom-Up Approach

Top-Down Approach

The advantages of this approach are:

- A truly corporate effort, an enterprise view of data
- Inherently architected—not a union of disparate data marts
- Single, central storage of data about the content
- Centralized rules and control
- May see quick results if implemented with iterations

The disadvantages are:

- Takes longer to build even with an iterative method
- High exposure/risk to failure
- Needs high level of cross-functional skills
- High outlay without proof of concept

This is the big-picture approach in which you build the overall, big, enterprise-wide data warehouse. Here you do not have a collection of fragmented islands of information. The data warehouse is large and integrated. This approach, however, would take longer to build and has a high risk of failure. If you do not have experienced professionals on your team, this approach could be dangerous. Also, it will be difficult to sell this approach to senior management and sponsors. They are not likely to see results soon enough.

Bottom-Up Approach

The advantages of this approach are:

- Faster and easier implementation of manageable pieces
- Favorable return on investment and proof of concept
- Less risk of failure
- Inherently incremental; can schedule important data marts first
- Allows project team to learn and grow

The disadvantages are:

- Each data mart has its own narrow view of data
- Permeates redundant data in every data mart
- Perpetuates inconsistent and irreconcilable data
- Proliferates unmanageable interfaces

In this bottom-up approach, you build your departmental data marts one by one. You would set a priority scheme to determine which data marts you must build first. The most severe drawback of this approach is data fragmentation. Each independent data mart will be blind to the overall requirements of the entire organization.

A Practical Approach

In order to formulate an approach for your organization, you need to examine what exactly your organization wants. Is your organization looking for long-term results or fast data marts for only a few subjects for now? Does your organization want quick, proof-of-concept, throw-away implementations? Or, do you want to look into some other practical approach?

Although both the top-down and the bottom-up approaches each have their own advantages and drawbacks, a compromise approach accommodating both views appears to be practical. The chief proponent of this practical approach is Ralph Kimball, an eminent author and data warehouse expert. The steps in this practical approach are as follows:

1. Plan and define requirements at the overall corporate level
2. Create a surrounding architecture for a complete warehouse
3. Conform and standardize the data content
4. Implement the data warehouse as a series of supermarts, one at a time

In this practical approach, you go to the basics and determine what exactly your organization wants in the long term. The key to this approach is that you first plan at the enterprise level. You gather requirements at the overall level. You establish the architecture for the complete warehouse. Then you determine the data content for each supermart. Supermarts are carefully architected data marts. You implement these supermarts, one at a time. Before implementation, you make sure that the data content among the various supermarts are conformed in terms of data types, field lengths, precision, and semantics. A certain data element must mean the same thing in every supermart. This will avoid spread of disparate data across several data marts.

A data mart, in this practical approach, is a logical subset of the complete data warehouse, a sort of pie-wedge of the whole data warehouse. A data warehouse, therefore, is a conformed union of all data marts. Individual data marts are targeted to particular business groups in the enterprise, but the collection of all the data marts form an integrated whole, called the enterprise data warehouse.

When we refer to data warehouses and data marts in our discussions here, we use the meanings as understood in this practical approach. For us, a data warehouse means a collection of the constituent data marts.

OVERVIEW OF THE COMPONENTS

We have now reviewed the basic definitions and features of data warehouses and data marts and completed a significant discussion of them. We have established our position on what the term data warehouse means to us. Now we are ready to examine its components.

When we build an operational system such as order entry, claims processing, or savings account, we put together several components to make up the system. The front-end component consists of the GUI (graphical user interface) to interface with the users for data input. The data storage component includes the database management system, such as Oracle, Informix, or Microsoft SQL Server. The display component is the set of screens and reports for the users. The data interfaces and the network software form the connectivity component. Depending on the information requirements and the framework of our organization, we arrange these components in the most optimum way.

Architecture is the proper arrangement of the components. You build a data warehouse with software and hardware components. To suit the requirements of your organization you arrange these building blocks in a certain way for maximum benefit. You may want to lay special emphasis on one component; you may want to bolster up another component with extra tools and services. All of this depends on your circumstances.

Figure 2-6 shows the basic components of a typical warehouse. You see the *Source Data* component shown on the left. The *Data Staging* component serves as the next building block. In the middle, you see the *Data Storage* component that manages the data warehouse data. This component not only stores and manages the data, it also keeps track of the data by means of the metadata repository. The *Information Delivery* component shown on the right consists of all the different ways of making the information from the data warehouse available to the users.

Whether you build a data warehouse for a large manufacturing company on the Fortune 500 list, a leading grocery chain with stores all over the country, or a global banking institution, the basic components are the same. Each data warehouse is put together with the same building blocks. The essential difference for each organization is in the way these building blocks are arranged. The variation is in the manner in which some of the blocks are made stronger than others in the architecture.

We will now take a closer look at each of the components. At this stage, we want to know what the components are and how each fits into the architecture. We also want to review specific issues relating to each particular component.

Source Data Component

Source data coming into the data warehouse may be grouped into four broad categories, as discussed here.

Architecture is the proper arrangement of the components.

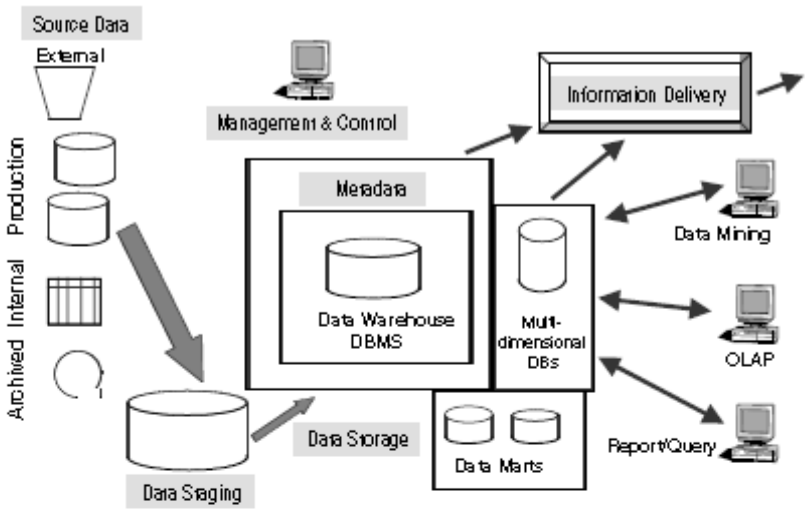


Figure 2-6 Data warehouse: building blocks or components.

Production Data. This category of data comes from the various operational systems of the enterprise. Based on the information requirements in the data warehouse, you choose segments of data from the different operational systems. While dealing with this data, you come across many variations in the data formats. You also notice that the data resides on different hardware platforms. Further, the data is supported by different database systems and operating systems. This is data from many vertical applications.

In operational systems, information queries are narrow. You query an operational system for information about specific instances of business objects. You may want just the name and address of a single customer. Or, you may need the orders placed by a single customer in a single week. Or, you may just need to look at a single invoice and the items billed on that single invoice. In operational systems, you do not have broad queries. You do not query the operational system in unexpected ways. The queries are all predictable. Again, you do not expect a particular query to run across different operational systems. What does all of this mean? Simply this: there is no conformance of data among the various operational systems of an enterprise. A term like *an account* may have different meanings in different systems.

The significant and disturbing characteristic of production data is disparity. Your great challenge is to standardize and transform the disparate data from the various production systems, convert the data, and integrate the pieces into useful data for storage in the data warehouse.

Internal Data. In every organization, users keep their "private" spreadsheets, documents, customer profiles, and sometimes even departmental databases. This is the internal data, parts of which could be useful in a data warehouse.

If your organization does business with the customers on a one-to-one basis and the contribution of each customer to the bottom line is significant, then detailed customer profiles with ample demographics are important in a data warehouse. Profiles of individual customers become very important for consideration. When your account representatives talk to their assigned customers or when your marketing department wants to make specific offerings to individual customers, you need the details. Although much of this data may be extracted from production systems, a lot of it is held by individuals and departments in their private files.

You cannot ignore the internal data held in private files in your organization. It is a collective judgment call on how much of the internal data should be included in the data warehouse. The IT department must work with the user departments to gather the internal data.

Internal data adds additional complexity to the process of transforming and integrating the data before it can be stored in the data warehouse. You have to determine strategies for collecting data from spreadsheets, find ways of taking data from textual documents, and tie into departmental databases to gather pertinent data from those sources. Again, you may want to schedule the acquisition of internal data. Initially, you may want to limit yourself to only some significant portions before going live with your first data mart.

Archived Data. Operational systems are primarily intended to run the current business. In every operational system, you periodically take the old data and store it in archived files. The circumstances in your organization dictate how often and which portions of the operational databases are archived for storage. Some data is archived after a year. Sometimes data is left in the operational system databases for as long as five years.

Many different methods of archiving exist. There are staged archival methods. At the first stage, recent data is archived to a separate archival database that may still be online. At the second stage, the older data is archived to flat files on disk storage. At the next stage, the oldest data is archived to tape cartridges or microfilm and even kept off-site.

As mentioned earlier, a data warehouse keeps historical snapshots of data. You essentially need historical data for analysis over time. For getting historical information, you look into your archived data sets. Depending on your data warehouse requirements, you have to include sufficient historical data. This type of data is useful for discerning patterns and analyzing trends.

External Data. Most executives depend on data from external sources for a high percentage of the information they use. They use statistics relating to their industry produced by external agencies. They use market share data of competitors. They use standard values of financial indicators for their business to check on their performance.

For example, the data warehouse of a car rental company contains data on the current production schedules of the leading automobile manufacturers. This external data in the data warehouse helps the car rental company plan for their fleet management.

The purposes served by such external data sources cannot be fulfilled by the data available within your organization itself. The insights gleaned from your production data and your archived data are somewhat limited. They give you a picture based on what you are doing or have done in the past. In order to spot industry trends and compare performance against other organizations, you need data from external sources.

Usually, data from outside sources do not conform to your formats. You have to devise

conversions of data into your internal formats and data types. You have to organize the data transmissions from the external sources. Some sources may provide information at regular, stipulated intervals. Others may give you the data on request. You need to accommodate the variations.

Data Staging Component

After you have extracted data from various operational systems and from external sources, you have to prepare the data for storing in the data warehouse. The extracted data coming from several disparate sources needs to be changed, converted, and made ready in a format that is suitable to be stored for querying and analysis.

Three major functions need to be performed for getting the data ready. You have to extract the data, transform the data, and then load the data into the data warehouse storage. These three major functions of extraction, transformation, and preparation for loading take place in a staging area. The data staging component consists of a workbench for these functions. Data staging provides a place and an area with a set of functions to clean, change, combine, convert, deduplicate, and prepare source data for storage and use in the data warehouse.

Why do you need a separate place or component to perform the data preparation? Can you not move the data from the various sources into the data warehouse storage itself and then prepare the data? When we implement an operational system, we are likely to pick up data from different sources, move the data into the new operational system database, and run data conversions. Why can't this method work for a data warehouse? The essential difference here is this: in a data warehouse you pull in data from many source operational systems. Remember that data in a data warehouse is subject-oriented and cuts across operational applications. A separate staging area, therefore, is a necessity for preparing data for the data warehouse.

Now that we have clarified the need for a separate data staging component, let us understand what happens in data staging. We will now briefly discuss the three major functions that take place in the staging area.

Data Extraction. This function has to deal with numerous data sources. You have to employ the appropriate technique for each data source. Source data may be from different source machines in diverse data formats. Part of the source data may be in relational database systems. Some data may be on other legacy network and hierarchical data models. Many data sources may still be in flat files. You may want to include data from spreadsheets and local departmental data sets. Data extraction may become quite complex.

Tools are available on the market for data extraction. You may want to consider using outside tools suitable for certain data sources. For the other data sources, you may want to develop in-house programs to do the data extraction. Purchasing outside tools may entail high initial costs. In-house programs, on the other hand, may mean ongoing costs for development and maintenance.

After you extract the data, where do you keep the data for further preparation? You may perform the extraction function in the legacy platform itself if that approach suits your framework. More frequently, data warehouse implementation teams extract the source into a separate physical environment from which moving the data into the data warehouse

would be easier. In the separate environment, you may extract the source data into a group of flat files, or a data-staging relational database, or a combination of both.

Data Transformation. In every system implementation, data conversion is an important function. For example, when you implement an operational system such as a magazine subscription application, you have to initially populate your database with data from the prior system records. You may be converting over from a manual system. Or, you may be moving from a file-oriented system to a modern system supported with relational database tables. In either case, you will convert the data from the prior systems. So, what is so different for a data warehouse? How is data transformation for a data warehouse more involved than for an operational system?

Again, as you know, data for a data warehouse comes from many disparate sources. If data extraction for a data warehouse poses great challenges, data transformation presents even greater challenges. Another factor in the data warehouse is that the data feed is not just an initial load. You will have to continue to pick up the ongoing changes from the source systems. Any transformation tasks you set up for the initial load will be adapted for the ongoing revisions as well.

You perform a number of individual tasks as part of data transformation. First, you clean the data extracted from each source. Cleaning may just be correction of misspellings, or may include resolution of conflicts between state codes and zip codes in the source data, or may deal with providing default values for missing data elements, or elimination of duplicates when you bring in the same data from multiple source systems.

Standardization of data elements forms a large part of data transformation. You standardize the data types and field lengths for same data elements retrieved from the various sources. Semantic standardization is another major task. You resolve synonyms and homonyms. When two or more terms from different source systems mean the same thing, you resolve the synonyms. When a single term means many different things in different source systems, you resolve the homonym.

Data transformation involves many forms of combining pieces of data from the different sources. You combine data from a single source record or related data elements from many source records. On the other hand, data transformation also involves purging source data that is not useful and separating out source records into new combinations. Sorting and merging of data takes place on a large scale in the data staging area.

In many cases, the keys chosen for the operational systems are field values with built-in meanings. For example, the product key value may be a combination of characters indicating the product category, the code of the warehouse where the product is stored, and some code to show the production batch. Primary keys in the data warehouse cannot have built-in meanings. We will discuss this further in Chapter 10. Data transformation also includes the assignment of surrogate keys derived from the source system primary keys.

A grocery chain point-of-sale operational system keeps the unit sales and revenue amounts by individual transactions at the check-out counter at each store. But in the data warehouse, it may not be necessary to keep the data at this detailed level. You may want to summarize the totals by product at each store for a given day and keep the summary totals of the sale units and revenue in the data warehouse storage. In such cases, the data transformation function would include appropriate summarization.

When the data transformation function ends, you have a collection of integrated data that is cleaned, standardized, and summarized. You now have data ready to load into each data set in your data warehouse.

Data Loading. Two distinct groups of tasks form the data loading function. When you complete the design and construction of the data warehouse and go live for the first time, you do the initial loading of the data into the data warehouse storage. The initial load moves large volumes of data using up substantial amounts of time. As the data warehouse starts functioning, you continue to extract the changes to the source data, transform the data revisions, and feed the incremental data revisions on an ongoing basis. Figure 2-7 illustrates the common types of data movements from the staging area to the data warehouse storage.

Data Storage Component

The data storage for the data warehouse is a separate repository. The operational systems of your enterprise support the day-to-day operations. These are online transaction processing applications. The data repositories for the operational systems typically contain only the current data. Also, these data repositories contain the data structured in highly normalized formats for fast and efficient processing. In contrast, in the data repository for a data warehouse, you need to keep large volumes of historical data for analysis. Further, you have to keep the data in the data warehouse in structures suitable for analysis, and not for quick retrieval of individual pieces of information. Therefore, the data storage for the data warehouse is kept separate from the data storage for operational systems.

In your databases supporting operational systems, the updates to data happen as transactions occur. These transactions hit the databases in a random fashion. How and when the transactions change the data in the databases is not completely within your control. The data in the operational databases could change from moment to moment. When your analysts use the data in the data warehouse for analysis, they need to know that the data is stable and that it represents snapshots at specified periods. As they are working with the

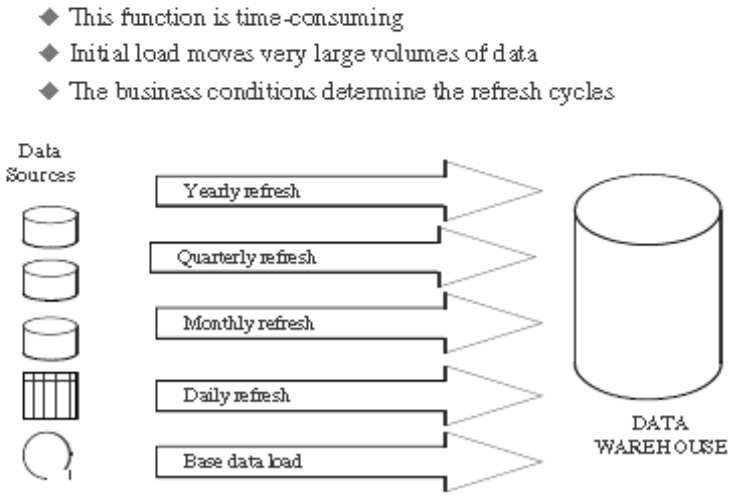


Figure 2-7 Data movements to the data warehouse.

data, the data storage must not be in a state of continual updating. For this reason, the data warehouses are "read-only" data repositories.

Generally, the database in your data warehouse must be open. Depending on your requirements, you are likely to use tools from multiple vendors. The data warehouse must be open to different tools. Most of the data warehouses employ relational database management systems.

Many of the data warehouses also employ multidimensional database management systems. Data extracted from the data warehouse storage is aggregated in many ways and the summary data is kept in the multidimensional databases (MDDBs). Such multidimensional database systems are usually proprietary products.

Information Delivery Component

Who are the users that need information from the data warehouse? The range is fairly comprehensive. The novice user comes to the data warehouse with no training and, therefore, needs prefabricated reports and preset queries. The casual user needs information once in a while, not regularly. This type of user also needs prepackaged information. The business analyst looks for ability to do complex analysis using the information in the data warehouse. The power user wants to be able to navigate throughout the data warehouse, pick up interesting data, format his or her own queries, drill through the data layers, and create custom reports and ad hoc queries.

In order to provide information to the wide community of data warehouse users, the information delivery component includes different methods of information delivery. Figure 2-8 shows the different information delivery methods. Ad hoc reports are predefined reports primarily meant for novice and casual users. Provision for complex queries, multidimensional (MD) analysis, and statistical analysis cater to the needs of the business analysts and power users. Information fed into Executive Information Systems (EIS) is meant for senior executives and high-level managers. Some data warehouses also provide data to data-mining applications. Data-mining applications are knowledge discovery systems

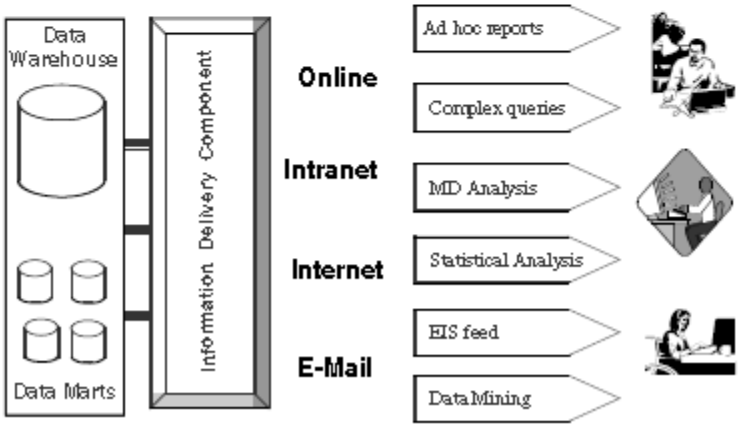


Figure 2-8 Information delivery component.

where the mining algorithms help you discover trends and patterns from the usage of your data.

In your data warehouse, you may include several information delivery mechanisms. Most commonly, you provide for online queries and reports. The users will enter their requests online and will receive the results online. You may set up delivery of scheduled reports through e-mail or you may make adequate use of your organization's intranet for information delivery. Recently, information delivery over the Internet has been gaining ground.

Metadata Component

Metadata in a data warehouse is similar to the data dictionary or the data catalog in a database management system. In the data dictionary, you keep the information about the logical data structures, the information about the files and addresses, the information about the indexes, and so on. The data dictionary contains data about the data in the database.

Similarly, the metadata component is the data about the data in the data warehouse. This definition is a commonly used definition. We need to elaborate on this definition. Metadata in a data warehouse is similar to a data dictionary, but much more than a data dictionary. Later, in a separate section in this chapter, we will devote more time for the discussion of metadata. Here, for the sake of completeness, we just want to list metadata as one of the components of the data warehouse architecture.

Management and Control Component

This component of the data warehouse architecture sits on top of all the other components. The management and control component coordinates the services and activities within the data warehouse. This component controls the data transformation and the data transfer into the data warehouse storage. On the other hand, it moderates the information delivery to the users. It works with the database management systems and enables data to be properly stored in the repositories. It monitors the movement of data into the staging area and from there into the data warehouse storage itself.

The management and control component interacts with the metadata component to perform the management and control functions. As the metadata component contains information about the data warehouse itself, the metadata is the source of information for the management module.

METADATA IN THE DATA WAREHOUSE

Think of metadata as the Yellow Pages© of your town. Do you need information about the stores in your town, where they are, what their names are, and what products they specialize in? Go to the Yellow Pages. The Yellow Pages is a directory with data about the institutions in your town. Almost in the same manner, the metadata component serves as a directory of the contents of your data warehouse.

Because of the importance of metadata in a data warehouse, we have set apart all of Chapter 9 for this topic. At this stage, we just want to get an introduction to the topic and highlight that metadata is a key architectural component of the data warehouse.

DIMENSIONAL ANALYSIS

In several ways, building a data warehouse is very different from building an operational system. This becomes notable especially in the requirements gathering phase. Because of this difference, the traditional methods of collecting requirements that work well for operational systems cannot be applied to data warehouses.

Usage of Information Unpredictable

Let us imagine you are building an operational system for order processing in your company. For gathering requirements, you interview the users in the Order Processing department. The users will list all the functions that need to be performed. They will inform you how they receive the orders, check stock, verify customers' credit arrangements, price the order, determine the shipping arrangements, and route the order to the appropriate warehouse. They will show you how they would like the various data elements to be presented on the GUI (graphical user interface) screen for the application. The users will also give you a list of reports they would need from the order processing application. They will be able to let you know how and when they would use the application daily.

In providing information about the requirements for an operational system, the users are able to give you precise details of the required functions, information content, and usage patterns. In striking contrast, for a data warehousing system, the users are generally unable to define their requirements clearly. They cannot define precisely what information they really want from the data warehouse, nor can they express how they would like to use the information or process it.

For most of the users, this could be the very first data warehouse they are being exposed to. The users are familiar with operational systems because they use these in their daily work, so they are able to visualize the requirements for other new operational systems. They cannot relate a data warehouse system to anything they have used before.

If, therefore, the whole process of defining requirements for a data warehouse is so nebulous, how can you proceed as one of the analysts in the data warehouse project? You are in a quandary. To be on the safe side, do you then include every piece of data you think the users will be able to use? How can you build something the users are unable to define clearly and precisely?

Initially, you may collect data on the overall business of the organization. You may check on the industry's best practices. You may gather some business rules guiding the day-to-day decision making. You may find out how products are developed and marketed. But these are generalities and are not sufficient to determine detailed requirements.

Dimensional Nature of Business Data

Fortunately, the situation is not as hopeless as it seems. Even though the users cannot fully describe what they want in a data warehouse, they can provide you with very important insights into how they think about the business. They can tell you what measurement units are important for them. Each user department can let you know how they measure success in that particular department. The users can give you insights into how they combine the various pieces of information for strategic decision making.

Managers think of the business in terms of business dimensions. Figure 5-1 shows the

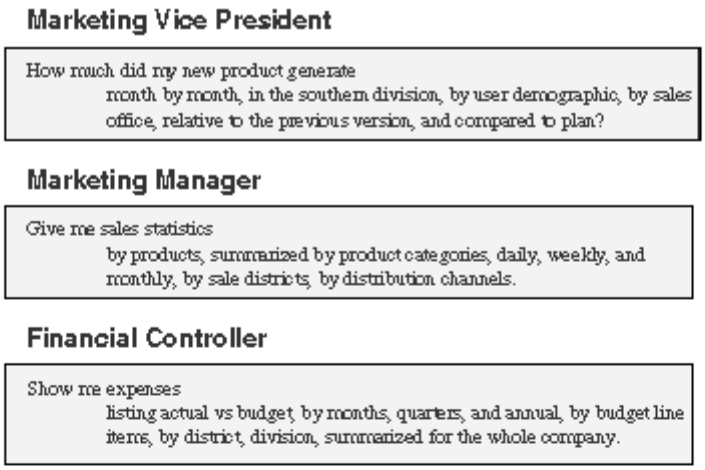


Figure 5-1 Managers think in business dimensions.

kinds of questions managers are likely to ask for decision making. The figure shows what questions a typical Marketing Vice President, a Marketing Manager, and a Financial Controller may ask.

Let us briefly examine these questions. The Marketing Vice President is interested in the revenue generated by her new product, but she is not interested in a single number. She is interested in the revenue numbers by month, in a certain division, by demographic, by sales office, relative to the previous product version, and compared to plan. So the Marketing Vice President wants the revenue numbers broken down by month, division, customer demographic, sales office, product version, and plan. These are her business dimensions along which she wants to analyze her numbers.

Similarly, for the Marketing Manager, his business dimensions are product, product category, time (day, week, month), sale district, and distribution channel. For the Financial Controller, the business dimensions are budget line, time (month, quarter, year), district, and division.

If your users of the data warehouse think in terms of business dimensions for decision making, you should also think of business dimensions while collecting requirements. Although the actual proposed usage of a data warehouse could be unclear, the business dimensions used by the managers for decision making are not nebulous at all. The users will be able to describe these business dimensions to you. You are not totally lost in the process of requirements definition. You can find out about the business dimensions.

Let us try to get a good grasp of the dimensional nature of business data. Figure 5-2 shows the analysis of sales units along the three business dimensions of product, time, and geography. These three dimensions are plotted against three axes of coordinates. You will see that the three dimensions form a collection of cubes. In each of the small dimensional cubes, you will find the sales units for that particular slice of time, product, and geographical division. In this case, the business data of sales units is three dimensional because

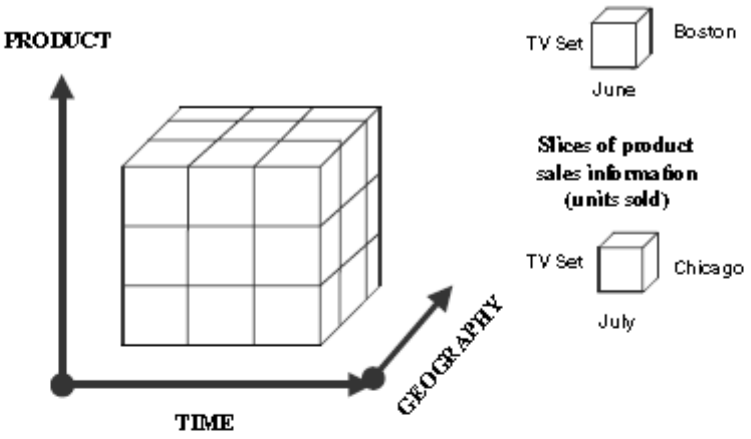


Figure 5-2 Dimensional nature of business data.

there are just three dimensions used in this analysis. If there are more than three dimensions, we extend the concept to multiple dimensions and visualize multidimensional cubes, also called hypercubes.

Examples of Business Dimensions

The concept of business dimensions is fundamental to the requirements definition for a data warehouse. Therefore, we want to look at some more examples of business dimensions in a few other cases. Figure 5-3 displays the business dimensions in four different cases.

Let us quickly look at each of these examples. For the supermarket chain, the measurements that are analyzed are the sales units. These are analyzed along four business dimensions. When you are looking for the hypercubes, the sides of such cubes are time, promotion, product, and store. If you are the Marketing Manager for the supermarket chain, you would want your sales broken down by product, at each store, in time sequence, and in relation to the promotions that take place.

For the insurance company, the business dimensions are different and appropriate for that business. Here you would want to analyze the claims data by agent, individual claim, time, insured party, individual policy, and status of the claim. The example of the airlines company shows the dimensions for analysis of frequent flyer data. Here the business dimensions are time, customer, specific flight, fare class, airport, and frequent flyer status.

The example analyzing shipments for a manufacturing company show some other business dimensions. In this case, the business dimensions used for the analysis of shipments are the ones relevant to that business and the subject of the analysis. Here you see the dimensions of time, ship-to and ship-from locations, shipping mode, product, and any special deals.

What we find from these examples is that the business dimensions are different and relevant to the industry and to the subject for analysis. We also find the time dimension to

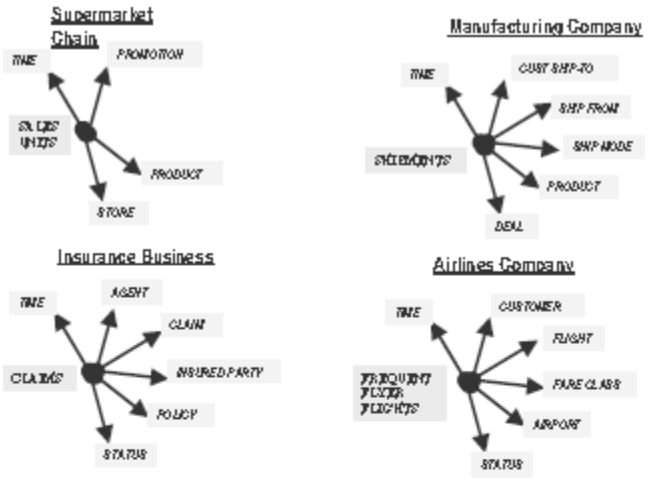


Figure 5-3 Examples of business dimensions.

be a common dimension in all examples. Almost all business analyses are performed over time.

INFORMATION PACKAGES—A NEW CONCEPT

We will now introduce a novel idea for determining and recording information requirements for a data warehouse. This concept helps us to give a concrete form to the various insights, nebulous thoughts, and opinions expressed during the process of collecting requirements. The information packages, put together while collecting requirements, are very useful for taking the development of the data warehouse to the next phases.

Requirements Not Fully Determinate

As we have discussed, the users are unable to describe fully what they expect to see in the data warehouse. You are unable to get a handle on what pieces of information you want to keep in the data warehouse. You are unsure of the usage patterns. You cannot determine how each class of users will use the new system. So, when requirements cannot be fully determined, we need a new and innovative concept to gather and record the requirements. The traditional methods applicable to operational systems are not adequate in this context. We cannot start with the functions, screens, and reports. We cannot begin with the data structures. We have noted that the users tend to think in terms of business dimensions and analyze measurements along such business dimensions. This is a significant observation and can form the very basis for gathering information.

The new methodology for determining requirements for a data warehouse system is based on business dimensions. It flows out of the need of the users to base their analysis on business dimensions. The new concept incorporates the basic measurements and the

business dimensions along which the users analyze these basic measurements. Using the new methodology, you come up with the measurements and the relevant dimensions that must be captured and kept in the data warehouse. You come up with what is known as an information package for the specific subject.

Let us look at an information package for analyzing sales for a certain business. Figure 5-4 contains such an information package. The subject here is sales. The measured facts or the measurements that are of interest for analysis are shown in the bottom section of the package diagram. In this case, the measurements are actual sales, forecast sales, and budget sales. The business dimensions along which these measurements are to be analyzed are shown at the top of diagram as column headings. In our example, these dimensions are time, location, product, and demographic age group. Each of these business dimensions contains a hierarchy or levels. For example, the time dimension has the hierarchy going from year down to the level of individual day. The other intermediary levels in the time dimension could be quarter, month, and week. These levels or hierarchical components are shown in the information package diagram.

Your primary goal in the requirements definition phase is to compile information packages for all the subjects for the data warehouse. Once you have firmed up the information packages, you'll be able to proceed to the other phases.

Essentially, information packages enable you to:

- Define the common subject areas
- Design key business metrics
- Decide how data must be presented
- Determine how users will aggregate or roll up
- Decide the data quantity for user analysis or query
- Decide how data will be accessed

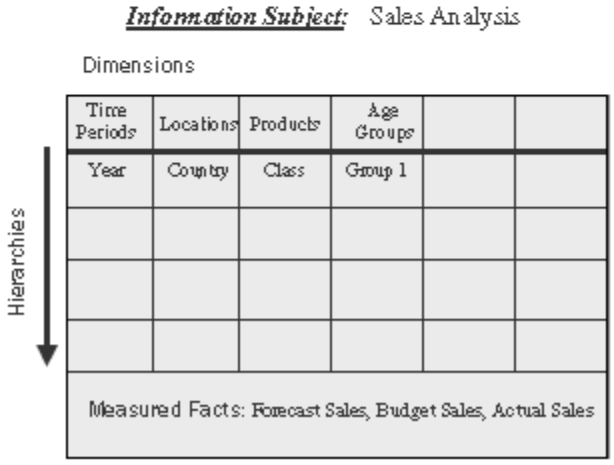


Figure 5-4 An information package.

- Establish data granularity
- Estimate data warehouse size
- Determine the frequency for data refreshing
- Ascertain how information must be packaged

Business Dimensions

As we have seen, business dimensions form the underlying basis of the new methodology for requirements definition. Data must be stored to provide for the business dimensions. The business dimensions and their hierarchical levels form the basis for all further phases. So we want to take a closer look at business dimensions. We should be able to identify business dimensions and their hierarchical levels. We must be able to choose the proper and optimal set of dimensions related to the measurements.

We begin by examining the business dimensions for an automobile manufacturer. Let us say that the goal is to analyze sales. We want to build a data warehouse that will allow the user to analyze automobile sales in a number of ways. The first obvious dimension is the product dimension. Again for the automaker, analysis of sales must include analysis by breaking the sales down by dealers. Dealer, therefore, is another important dimension for analysis. As an automaker, you would want to know how your sales break down along customer demographics. You would want to know who is buying your automobiles and in what quantities. Customer demographics would be another useful business dimension for analysis. How do the customers pay for the automobiles? What effect does financing for the purchases have on the sales? These questions can be answered by including the method of payment as another dimension for analysis. What about time as a business dimension? Almost every query or analysis involves the time element. In summary, we have come up with the following dimensions for the subject of sales for an automaker: product, dealer, customer demographic, method of payment, and time.

Let us take one more example. In this case, we want to come up with an information package for a hotel chain. The subject in this case is hotel occupancy. We want to analyze occupancy of the rooms in the various branches of the hotel chain. We want to analyze the occupancy by individual hotels and by room types. So hotel and room type are critical business dimensions for the analysis. As in the other case, we also need to include the time dimension. In the hotel occupancy information package, the dimensions included are hotel, room type, and time.

Dimension Hierarchies/Categories

When a user analyzes the measurements along a business dimension, the user usually would like to see the numbers first in summary and then at various levels of detail. What the user does here is to traverse the hierarchical levels of a business dimension for getting the details at various levels. For example, the user first sees the total sales for the entire year. Then the user moves down to the level of quarters and looks at the sales by individual quarters. After this, the user moves down further to the level of individual months to look at monthly numbers. What we notice here is that the hierarchy of the time dimension consists of the levels of year, quarter, and month. The dimension hierarchies are the paths for drilling down or rolling up in our analysis.

Within each major business dimension there are categories of data elements that can

also be useful for analysis. In the time dimension, you may have a data element to indicate whether a particular day is a holiday. This data element would enable you to analyze by holidays and see how sales on holidays compare with sales on other days. Similarly, in the product dimension, you may want to analyze by type of package. The package type is one such data element within the product dimension. The holiday flag in the time dimension and the package type in the product dimension do not necessarily indicate hierarchical levels in these dimensions. Such data elements within the business dimension may be called categories.

Hierarchies and categories are included in the information packages for each dimension. Let us go back to the two examples in the previous section and find out which hierarchical levels and categories must be included for the dimensions. Let us examine the product dimension. Here, the product is the basic automobile. Therefore, we include the data elements relevant to product as hierarchies and categories. These would be model name, model year, package styling, product line, product category, exterior color, interior color, and first model year. Looking at the other business dimensions for the auto sales analysis, we summarize the hierarchies and categories for each dimension as follows:

Product: Model name, model year, package styling, product line, product category, exterior color, interior color, first model year

Dealer: Dealer name, city, state, single brand flag, date first operation

Customer demographics: Age, gender, income range, marital status, household size, vehicles owned, home value, own or rent

Payment method: Finance type, term in months, interest rate, agent

Time: Date, month, quarter, year, day of week, day of month, season, holiday flag

Let us go back to the hotel occupancy analysis. We have included three business dimensions. Let us list the possible hierarchies and categories for the three dimensions.

Hotel: Hotel line, branch name, branch code, region, address, city, state, Zip Code, manager, construction year, renovation year

Room type: Room type, room size, number of beds, type of bed, maximum occupants, suite, refrigerator, kitchenette

Time: Date, day of month, day of week, month, quarter, year, holiday flag

Key Business Metrics or Facts

So far we have discussed the business dimensions in the above two examples. These are the business dimensions relevant to the users of these two data warehouses for performing analysis. The respective users think of their business subjects in terms of these business dimensions for obtaining information and for doing analysis.

But using these business dimensions, what exactly are the users analyzing? What numbers are they analyzing? The numbers the users analyze are the measurements or metrics that measure the success of their departments. These are the facts that indicate to the users how their departments are doing in fulfilling their departmental objectives.

In the case of the automaker, these metrics relate to the sales. These are the numbers that tell the users about their performance in sales. These are numbers about the sale of

each individual automobile. The set of meaningful and useful metrics for analyzing automobile sales is as follows:

Actual sale price
MSRP sale price
Options price
Full price
Dealer add-ons
Dealer credits
Dealer invoice
Amount of downpayment
Manufacturer proceeds
Amount financed

In the second example of hotel occupancy, the numbers or metrics are different. The nature of the metrics depends on what is being analyzed. For hotel occupancy, the metrics would therefore relate to the occupancy of rooms in each branch of the hotel chain. Here is a list of metrics for analyzing hotel occupancy:

Occupied rooms
Vacant rooms
Unavailable rooms
Number of occupants
Revenue

Now putting it all together, let us discuss what goes into the information package diagrams for these two examples. In each case, the metrics or facts go into the bottom section of the information package. The business dimensions will be the column headings. In each column, you will include the hierarchies and categories for the business dimensions.

Figures 5-5 and 5-6 show the information packages for the two examples we just discussed.

Information Subject: Automaker Sales

Dimensions

Hierarchies / Categories

Time	Product	Payment Method	Customer Demo-graphics	Dealer	
Year	Model Name	Finance Type	Age	Dealer Name	
Quarter	Model Year	Term (Months)	Gender	City	
Month	Package Styling	Interest Rate	Income Range	State	
Date	Product Line	Agent	Marital Status	Single Brand Flag	
Day of Week	Product Category		House-hold Size	Date First Operation	
Day of Month	Exterior Color		Vehicles Owned		
Season	Interior Color		Home Value		
Holiday Flag	First Year		Own or Rent		
Facts: Actual Sale Price, MSRP Sale Price, Options Price, Full Price, Dealer Add-ons, Dealer Credits, Dealer Invoice, Down Payment, Proceeds, Finance					

Figure 5-5 Information package: automaker sales.

Information Subject: Hotel Occupancy

Dimensions

Hierarchies / Categories

Time	Hotel	Room Type			
Year	Hotel Line	Room Type			
Quarter	Branch Name	Room Size			
Month	Branch Code	Number of Beds			
Date	Region	Type of Bed			
Day of Week	Address	Max. Occupants			
Day of Month	City/State/ Zip	Suite			
Holiday Flag	Construc-tion Year	Refriger-ator			
	Renova-tion Year	Kitchen-ette			
Facts: Occupied Rooms, Vacant Rooms, Unavailable Rooms, Number of Occupants, Revenue					

Figure 5-6 Information package: hotel occupancy.

CHARACTERISTICS	OLTP SYSTEMS	DATA WAREHOUSE
Analytical capabilities	Very low	Moderate
Data for a single session	Very limited	Small to medium size
Size of result set	Small	Large
Response time	Very fast	Fast to moderate
Data granularity	Detail	Detail and summary
Data currency	Current	Current and historical
Access method	Predefined	Predefined and ad hoc
Basic motivation	Collect and input data	Provide information
Data model	Design for data updates	Design for queries
Optimization of database	For transactions	For analysis
Update frequency	Very frequent	Generally read-only
Scope of user interaction	Single transactions	Throughout data content

Figure 15-2 OLTP and data warehouse environments.

OLAP is the Answer

Users certainly need the ability to perform multidimensional analysis with complex calculations, but we find that the traditional tools of report writers, query products, spreadsheets, and language interfaces are distressfully inadequate. What is the answer? Clearly, the tools being used in the OLTP and basic data warehouse environments do not match up to the task. We need different set of tools and products that are specifically meant for serious analysis. We need OLAP in the data warehouse.

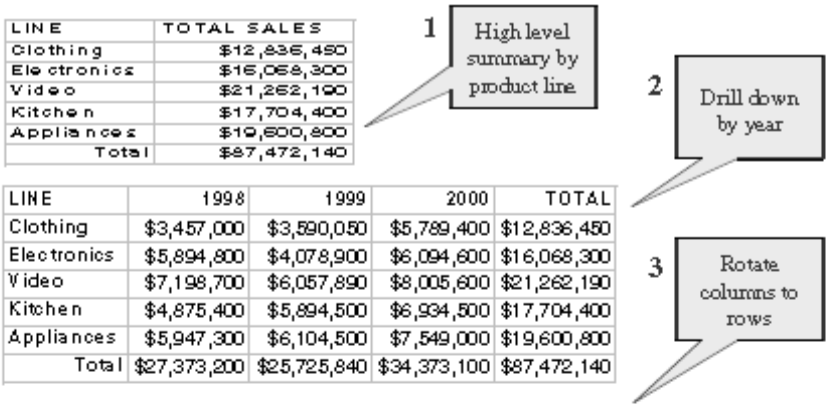
In this chapter, we will thoroughly examine the various aspects of OLAP. We will come up with formal definitions and detailed characteristics. We will highlight all the features and functions. We will explore the different OLAP models. But now that you have an initial appreciation for OLAP, let us list the basic virtues of OLAP to justify our proposition.

- Enables analysts, executives, and managers to gain useful insights from the presentation of data.
- Can reorganize metrics along several dimensions and allow data to be viewed from different perspectives.
- Supports multidimensional analysis.
- Is able to drill down or roll up within each dimension.
- Is capable of applying mathematical formulas and calculations to measures.
- Provides fast response, facilitating speed-of-thought analysis.
- Complements the use of other information delivery techniques such as data mining.
- Improves the comprehension of result sets through visual presentations using graphs and charts.
- Can be implemented on the Web.
- Designed for highly interactive analysis.

Even at this stage, you will further appreciate the nature and strength of OLAP by studying a typical OLAP session (see Figure 15-3). The analyst starts with a query requesting a high-level summary by product line. Next, the user moves to drilling down for details by year. In the following step, the analyst pivots the data to view totals by year rather than totals by product line. Even in such a simple example, you observe the power and features of OLAP.

OLAP Definitions and Rules

Where did the term OLAP originate? We know that multidimensionality is at the core of OLAP systems. We have also mentioned some other basic features of OLAP. Is it a vague



LINE	TOTAL SALES
Clothing	$12,836,450
Electronics	$16,068,300
Video	$21,262,190
Kitchen	$17,704,400
Appliances	$19,600,800
Total	$87,472,140

LINE	1998	1999	2000	TOTAL
Clothing	$3,457,000	$3,590,050	$5,789,400	$12,836,450
Electronics	$5,894,800	$4,078,900	$6,094,600	$16,068,300
Video	$7,198,700	$6,057,890	$8,005,600	$21,262,190
Kitchen	$4,875,400	$5,894,500	$6,934,500	$17,704,400
Appliances	$5,947,300	$6,104,500	$7,549,000	$19,600,800
Total	$27,373,200	$25,725,840	$34,373,100	$87,472,140

YEAR	Clothing	Electronics	Video	Kitchen	Appliances	TOTAL
1998	$3,457,000	$5,894,800	$7,198,700	$4,875,400	$5,947,300	$27,373,200
1999	$3,590,050	$4,078,900	$6,057,890	$5,894,500	$6,104,500	$25,725,840
2000	$5,789,400	$6,094,600	$8,005,600	$6,934,500	$7,549,000	$34,373,100
Total	$12,836,450	$16,068,300	$21,262,190	$17,704,400	$19,600,800	$87,472,140

Figure 15-3 *Simple OLAP session.*

collection of complex factors for serious analysis? Is there a formal definition and a set of fundamental guidelines identifying OLAP systems?

The term OLAP or online analytical processing was introduced in a paper entitled "Providing On-Line Analytical Processing to User Analysts," by Dr. E. F. Codd, the acknowledged "father" of the relational database model. The paper, published in 1993, defined 12 rules or guidelines for an OLAP system. Later, in 1995, six additional rules were included. We will discuss these rules. Before that, let us look for a short and precise definition for OLAP. Such a succinct definition comes from the OLAP council, which provides membership, sponsors research, and promotes the use of OLAP. Here is the definition:

> On-Line Analytical Processing (OLAP) is a category of software technology that enables analysts, managers and executives to gain insight into data through fast, consistent, interactive access in a wide variety of possible views of information that has been transformed from raw data to reflect the real dimensionality of the enterprise as understood by the user.

The definition from the OLAP council contains all the key ingredients. Speed, consistency, interactive access, and multiple dimensional views—all of these are principal elements. As one trade magazine described it in 1995, OLAP is a fancy term for multidimensional analysis.

The guidelines proposed by Dr. Codd form the yardstick for measuring any sets of OLAP tools and products. A true OLAP system must conform to these guidelines. When

your project team is looking for OLAP tools, it can prioritize these guidelines and select tools that meet the set of criteria at the top of your priority list. First, let us consider the initial twelve guidelines for an OLAP system:

Multidimensional Conceptual View. Provide a multidimensional data model that is intuitively analytical and easy to use. Business users' view of an enterprise is multidimensional in nature. Therefore, a multidimensional data model conforms to how the users perceive business problems.

Transparency. Make the technology, underlying data repository, computing architecture, and the diverse nature of source data totally transparent to users. Such transparency, supporting a true open system approach, helps to enhance the efficiency and productivity of the users through front-end tools that are familiar to them.

Accessibility. Provide access only to the data that is actually needed to perform the specific analysis, presenting a single, coherent, and consistent view to the users. The OLAP system must map its own logical schema to the heterogeneous physical data stores and perform any necessary transformations.

Consistent Reporting Performance. Ensure that the users do not experience any significant degradation in reporting performance as the number of dimensions or the size of the database increases. Users must perceive consistent run time, response time, or machine utilization every time a given query is run.

Client/Server Architecture. Conform the system to the principles of client/server architecture for optimum performance, flexibility, adaptability, and interoperability. Make the server component sufficiently intelligent to enable various clients to be attached with a minimum of effort and integration programming.

Generic Dimensionality. Ensure that every data dimension is equivalent in both structure and operational capabilities. Have one logical structure for all dimensions. The basic data structure or the access techniques must not be biased toward any single data dimension.

Dynamic Sparse Matrix Handling. Adapt the physical schema to the specific analytical model being created and loaded that optimizes sparse matrix handling. When encountering a sparse matrix, the system must be able to dynamically deduce the distribution of the data and adjust the storage and access to achieve and maintain consistent level of performance.

Multiuser Support. Provide support for end users to work concurrently with either the same analytical model or to create different models from the same data. In short, provide concurrent data access, data integrity, and access security.

Unrestricted Cross-dimensional Operations. Provide ability for the system to recognize dimensional hierarchies and automatically perform roll-up and drill-down operations within a dimension or across dimensions. Have the interface language allow calculations and data manipulations across any number of data dimensions, without restricting any relations between data cells, regardless of the number of common data attributes each cell contains.

Intuitive Data Manipulation. Enable consolidation path reorientation (pivoting), drill-down and roll-up, and other manipulations to be accomplished intuitively and directly via point-and-click and drag-and-drop actions on the cells of the analytical model. Avoid the use of a menu or multiple trips to a user interface.

Flexible Reporting. Provide capabilities to the business user to arrange columns, rows, and cells in a manner that facilitates easy manipulation, analysis, and synthesis of information. Every dimension, including any subsets, must be able to be displayed with equal ease.

Unlimited Dimensions and Aggregation Levels. Accommodate at least fifteen, preferably twenty, data dimensions within a common analytical model. Each of these generic dimensions must allow a practically unlimited number of user-defined aggregation levels within any given consolidation path.

In addition to these twelve basic guidelines, also take into account the following requirements, not all distinctly specified by Dr. Codd.

Drill-through to Detail Level. Allow a smooth transition from the multidimensional, preaggregated database to the detail record level of the source data warehouse repository.

OLAP Analysis Models. Support Dr. Codd's four analysis models: exegetical (or descriptive), categorical (or explanatory), contemplative, and formulaic.

Treatment of Nonnormalized Data. Prohibit calculations made within an OLAP system from affecting the external data serving as the source.

Storing OLAP Results. Do not deploy write-capable OLAP tools on top of transactional systems.

Missing Values. Ignore missing values, irrespective of their source.

Incremental Database Refresh. Provide for incremental refreshes of the extracted and aggregated OLAP data.

SQL Interface. Seamlessly integrate the OLAP system into the existing enterprise environment.

OLAP Characteristics

Let us summarize in simple terms what we have covered so far. We explored why the business users absolutely need online analytical processing. We examined why the other methods of information delivery do not satisfy the requirements for multidimensional analysis with powerful calculations and fast access. We discussed how OLAP is the answer to satisfy these requirements. We reviewed the definitions and authoritative guidelines for the OLAP system.

Before we get into a more detailed discussion of the major features of OLAP systems, let us list the most fundamental characteristics in plain language. OLAP systems

- let business users have a multidimensional and logical view of the data in the data warehouse,
- facilitate interactive query and complex analysis for the users,
- allow users to drill down for greater details or roll up for aggregations of metrics along a single business dimension or across multiple dimensions,
- provide ability to perform intricate calculations and comparisons, and
- present results in a number of meaningful ways, including charts and graphs.

MAJOR FEATURES AND FUNCTIONS

Very often, you are faced with the question of whether OLAP is not just data warehousing in a nice wrapper? Can you not consider online analytical processing as just an information delivery technique and nothing more? Is it not another layer in the data warehouse, providing interface between the data and the users? In some sense, OLAP is an information delivery system for the data warehouse. But OLAP is much more than that. A data warehouse stores data and provides simpler access to the data. An OLAP system complements the data warehouse by lifting the information delivery capabilities to new heights.

General Features

In this section, we will pay special attention to a few major features and functions of OLAP systems. You will gain greater insight into dimensional analysis, find deeper meanings about the necessity for drill-downs and roll-ups during analysis sessions and gain greater appreciation for the role of slicing and dicing operations in analysis. Before getting into greater details about these, let us recapitulate the general features of OLAP. Please go to Figure 15-4 and note the summary. Also note the distinction between basic features and advanced features. The list shown in the figure includes the general features you observe in practice in most OLAP environments. Please use the list as a quick checklist of features your project team must consider for your OLAP system.

Dimensional Analysis

By this time, you are perhaps tired of the term "dimensional analysis." We had to use the term a few times so far. You have been told that dimensional analysis is a strong suit in the

BASIC FEATURES	Multidimensional analysis	Consistent performance	Fast response times for interactive queries
	Drill-down and roll-up	Navigation in and out of details	Slice-and-dice or rotation
	Multiple view modes	Easy scalability	Time intelligence (year-to-date, fiscal period)
ADVANCED FEATURES	Powerful calculations	Cross-dimensional calculations	Pre-calculation or pre-consolidation
	Drill-through across dimensions or details	Sophisticated presentation & displays	Collaborative decision making
	Derived data values through formulas	Application of alert technology	Report generation with agent technology

Figure 15-4 General features of OLAP.

arsenal of OLAP. Any OLAP system devoid of multidimensional analysis is utterly useless. So try to get a clear picture of the facility provided in OLAP systems for dimensional analysis.

Let us begin with a simple STAR schema. This STAR schema has three business dimensions, namely, product, time, and store. The fact table contains sales. Please see Figure 15-5 showing the schema and a three-dimensional representation of the model as a cube, with products on the X-axis, time on the Y-axis, and stores on the Z-axis. What are the values represented along each axis? For example, in the STAR schema, time is one of the dimensions and month is one of the attributes of the time dimension. Values of this attribute month are represented on the Y-axis. Similarly, values of the attributes product name and store name are represented on the other two axes.

This schema with just three business dimensions does not even look like a star. Nevertheless, it is a dimensional model. From the attributes of the dimension tables, pick the attribute product name from the product dimension, month from the time dimension, and store name from the store dimension. Now look at the cube representing the values of these attributes along the primary edges of the physical cube. Go further and visualize the sales for coats in the month of January at the New York store to be at the intersection of the three lines representing the product: coats, month: January, and store: New York.

If you are displaying the data for sales along these three dimensions on a spreadsheet, the columns may display the product names, the rows the months, and pages the data along the third dimension of store names. See Figure 15-6 showing a screen display of a page of this three-dimensional data.

The page displayed on the screen shows a slice of the cube. Now look at the cube and move along a slice or plane passing through the point on the Z-axis representing store: New York. The intersection points on this slice or plane relate to sales along product and

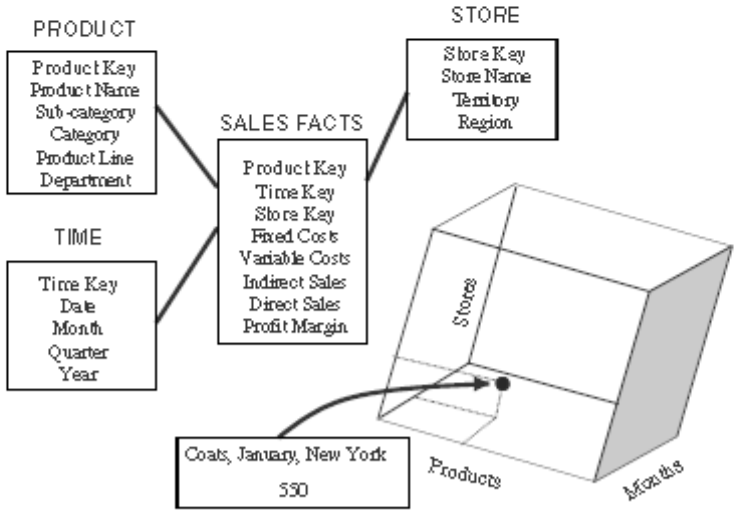


Figure 15-5 Simple STAR schema.

Store: New York

Products

PAGES: STORE dimension

COLUMNS: PRODUCT dimension

ROWS: TIME dimension

Months

	Hats	Coats	Jackets	Dresses	Shirts	Slacks
Jan	200	550	350	500	520	490
Feb	210	480	390	510	530	500
Mar	190	480	380	480	500	470
Apr	190	430	350	490	510	480
May	160	530	320	530	550	520
Jun	150	450	310	540	560	330
Jul	130	480	270	550	570	250
Aug	140	570	250	650	670	230
Sep	160	470	240	630	650	210
Oct	170	480	260	610	630	250
Nov	190	520	280	680	700	260
Dec	200	560	320	750	770	310

Figure 15-6 A Three-dimensional display.

time business dimensions for store: New York. Try to relate these sale numbers to the slice on the cube representing store: New York.

Now we have a way of depicting three business dimensions and a single fact on a two-dimensional page and also on a three-dimensional cube. The numbers in each cell on the page are the sale numbers. What could be the types of multidimensional analysis on this particular set of data? What types of queries could be run during the course of analysis sessions? You could get sale numbers along the hierarchies of a combination of the three business dimensions of product, store, and time. You could perform various types of three-dimensional analysis of sales. The results of queries during analysis sessions will be displayed on the screen with the three dimensions represented in columns, rows, and pages. The following is a sample of simple queries and the result sets during a multidimensional analysis session.

Query

Display the total sales of all products for past five years in all stores.

Display of Results

Rows: Year numbers 2000, 1999, 1998, 1997, 1996

Columns: Total Sales for all products

Page: One store per page

Query

Compare total sales for all stores, product by product, between years 2000 and 1999.

Display of Results

Rows: Year numbers 2000, 1999; difference; percentage increase or decrease

Columns: One column per product, showing all products

Page: All stores

Drill-Down and Roll-Up

Return to Figure 15-5. Look at the attributes of the product dimension table of the STAR schema. In particular, note these specific attributes of the product dimension: product name, subcategory, category, product line, and department. These attributes signify an ascending hierarchical sequence from product name to department. A department includes product lines, a product line includes categories, a category includes subcategories, and each subcategory consists of products with individual product names. In an OLAP system, these attributes are called hierarchies of the product dimension.

OLAP systems provide drill-down and roll-up capabilities. Try to understand what we mean by these capabilities with reference to above example. Please see Figure 15-12 illustrating these capabilities with reference to the product dimension hierarchies. Note the different types of information given in the figure. It shows the rolling up to higher hierarchical levels of aggregation and the drilling down to lower levels of detail. Also note the sales numbers shown alongside. These are sales for one particular store in one particular month at these levels of aggregation. The sale numbers you notice as you go down the hierarchy are for a single department, a single product line, a single category, and so on. You drill down to get the lower level breakdown of sales. The figure also shows the drill-across

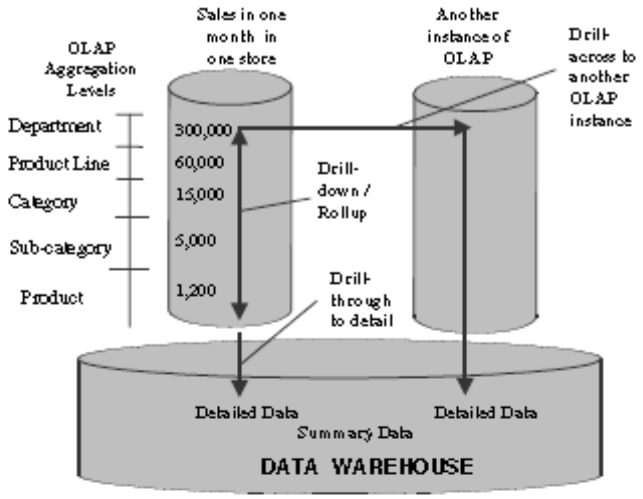


Figure 15-12 Roll-up and drill-down features of OLAP.

to another OLAP summarization using a different set of hierarchies of other dimensions. Notice also the drill-through to the lower levels of granularity, as stored in the source data warehouse repository. Roll-up, drill-down, drill-across, and drill-through are extremely useful features of OLAP systems supporting multidimensional analysis.

One more question remains. While you are rolling up or drilling down, how do the page displays change on the spreadsheets? For example, return to Figure 15-6 and look at the

Store: New York

Sub-categories

PAGES: STORE dimension

COLUMNS: PRODUCT dimension

ROWS: TIME dimension

Months

	Outer	Dress	Casual
Jan	1,100	1,020	490
Feb	1,080	1,040	500
Mar	1,050	980	470
Apr	970	1,000	480
May	1,010	1,080	520
Jun	910	1,100	330
Jul	880	1,120	250
Aug	960	1,320	230
Sep	870	1,280	210
Oct	910	1,240	250
Nov	980	1,380	260
Dec	1,080	1,520	310

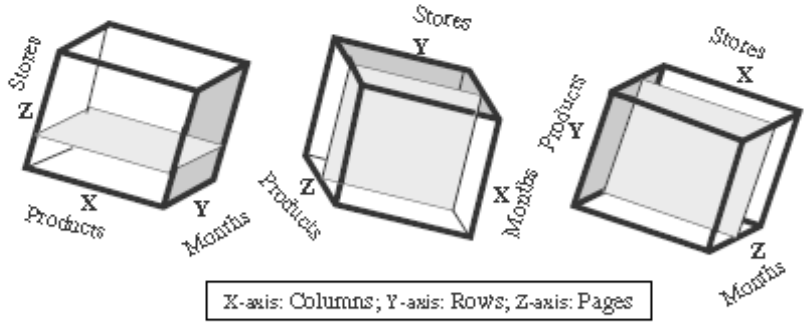
Figure 15-13 Three-dimensional display with roll-up.

page display on the spreadsheet. The columns represent the various products, the rows represent the months, and the pages represent the stores. At this point, if you want to roll up to the next higher level of subcategory, how will the display in Figure 15-6 change? The columns on the display will have to change to represent subcategories instead of products. Please see Figure 15-13 indicating this change.

Let us ask just one more question before we leave this subsection. When you have rolled up to the subcategory level in the product dimension, what happens to the display if you also roll up to the next higher level of the store dimension, territory? How will the display on the spreadsheet change? Now the spreadsheet will display the sales with columns representing subcategories, rows representing months, and the pages representing territories.

Slice-and-Dice or Rotation

Let us revisit Figure 15-6 showing the display of months as rows, products as columns, and stores as pages. Each page represents the sales for one store. The data model corresponds to a physical cube with these data elements represented by its primary edges. The page displayed is a slice or two-dimensional plane of the cube. In particular, this display page for the New York store is the slice parallel to the product and time axes. Now begin to look at Figure 15-14 carefully. On the left side, the first part of the diagram shows this alignment of the cube. For the sake simplicity, only three products, three months, and three stores are chosen for illustration.



Store: New York

	Hats	Coats	Jackets
Jan	200	550	350
Feb	210	480	390
Mar	190	480	380

Product: Hats

	Jan	Feb	Mar
New York	200	210	190
Boston	210	250	240
San Jose	130	90	70

Month: January

	New York	Boston	San Jose
Hats	200	210	130
Coats	550	500	200
Jackets	350	400	100

Figure 15-14 Slicing and dicing.

Now rotate the cube so that products are along the Z-axis, months are along the X-axis, and stores are along the Y-axis. The slice we are considering also rotates. What happens to the display page that represents the slice? Months are now shown as columns and stores as rows. The display page represents the sales of one product, namely product: hats.

You can go to the next rotation so that months are along the Z-axis, stores are along the X-axis, and products are along the Y-axis. The slice we are considering also rotates. What happens to the display page that represents the slice? Stores are now shown as columns and products as rows. The display page represents the sales of one month, namely month: January.

What is the great advantage of all of this for the users? Did you notice that with each rotation, the users can look at page displays representing different versions of the slices in the cube. The users can view the data from many angles, understand the numbers better, and arrive at meaningful conclusions.

Uses and Benefits

After exploring the features of OLAP in sufficient detail, you must have already deduced the enormous benefits of OLAP. We have discussed multidimensional analysis as provided in OLAP systems. The ability to perform multidimensional analysis with complex queries sometimes also entails complex calculations.

Let us summarize the benefits of OLAP systems:

- Increased productivity of business managers, executives, and analysts
- Inherent flexibility of OLAP systems means that users may be self-sufficient in running their own analysis without IT assistance
- Benefit for IT developers because using software specifically designed for the system development results in faster delivery of applications
- Self-sufficiency of users, resulting in reduction in backlog
- Faster delivery of applications following from the previous benefits
- More efficient operations through reducing time on query executions and in network traffic
- Ability to model real-world challenges with business metrics and dimensions

OLAP MODELS

Have you heard of the terms ROLAP or MOLAP? There is another variation, DOLAP. A very simple explanation of the variations relates to the way data is stored for OLAP. The processing is still online analytical processing, only the storage methodology is different.

ROLAP stands for relational online analytical processing and MOLAP stands for multidimensional online analytical processing. In either case, the information interface is still OLAP. DOLAP stands for desktop online analytical processing. DOLAP is meant to provide portability to users of online analytical processing. In the DOLAP methodology, multidimensional datasets are created and transferred to the desktop machine, requiring only the DOLAP software to exist on that machine. DOLAP is a variation of ROLAP.

Overview of Variations

In the MOLAP model, online analytical processing is best implemented by storing the data multidimensionally, that is, easily viewed in a multidimensional way. Here the data structure is fixed so that the logic to process multidimensional analysis can be based on well-defined methods of establishing data storage coordinates. Usually, multidimensional databases (MDDBs) are vendors' proprietary systems. On the other hand, the ROLAP model relies on the existing relational DBMS of the data warehouse. OLAP features are provided against the relational database.

See Figure 15-15 contrasting the two models. Notice the MOLAP model shown on the left side of the figure. The OLAP engine resides on a special server. Proprietary multidimensional databases (MDDBs) store data in the form of multidimensional hypercubes. You have to run special extraction and aggregation jobs to create these multidimensional data cubes in the MDDBs from the relational database of the data warehouse. The special server presents the data as OLAP cubes for processing by the users.

On the right side of the figure you see the ROLAP model. The OLAP engine resides on the desktop. Prefabricated multidimensional cubes are not created beforehand and stored in special databases. The relational data is presented as virtual multidimensional data cubes.

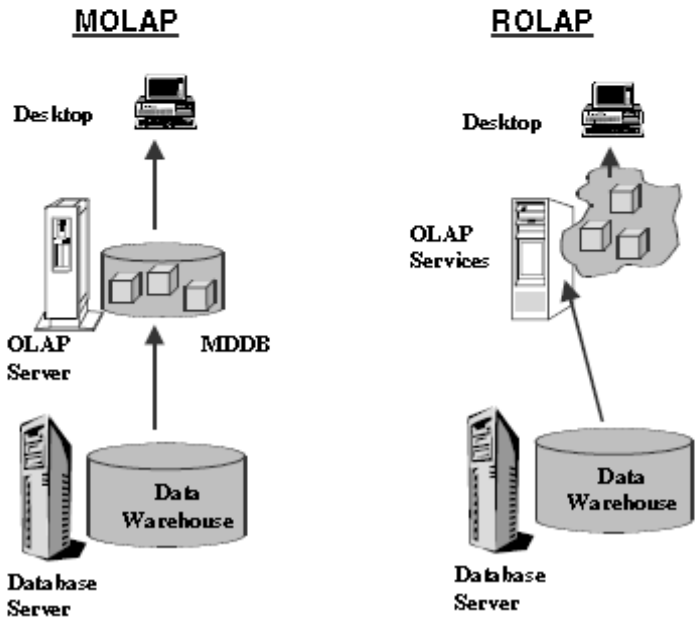


Figure 15-15 OLAP models.

The MOLAP Model

As discussed, in the MOLAP model, data for analysis is stored in specialized multidimensional databases. Large multidimensional arrays form the storage structures. For example, to store sales number of 500 units for product ProductA, in month number 2001/01, in store StoreS1, under distributing channel Channel05, the sales number of 500 is stored in an array represented by the values (ProductA, 2001/01, StoreS1, Channel05).

The array values indicate the location of the cells. These cells are intersections of the values of dimension attributes. If you note how the cells are formed, you will realize that not all cells have values of metrics. If a store is closed on Sundays, then the cells representing Sundays will all be nulls.

Let us now consider the architecture for the MOLAP model. Please go over each part of Figure 15-16 carefully. Note the three layers in the multitier architecture. Precalculated and prefabricated multidimensional data cubes are stored in multidimensional databases. The MOLAP engine in the application layer pushes a multidimensional view of the data from the MDDBs to the users.

As mentioned earlier, multidimensional database management systems are proprietary software systems. These systems provide the capability to consolidate and fabricate summarized cubes during the process that loads data into the MDDBs from the main data warehouse. The users who need summarized data enjoy fast response times from the preconsolidated data.

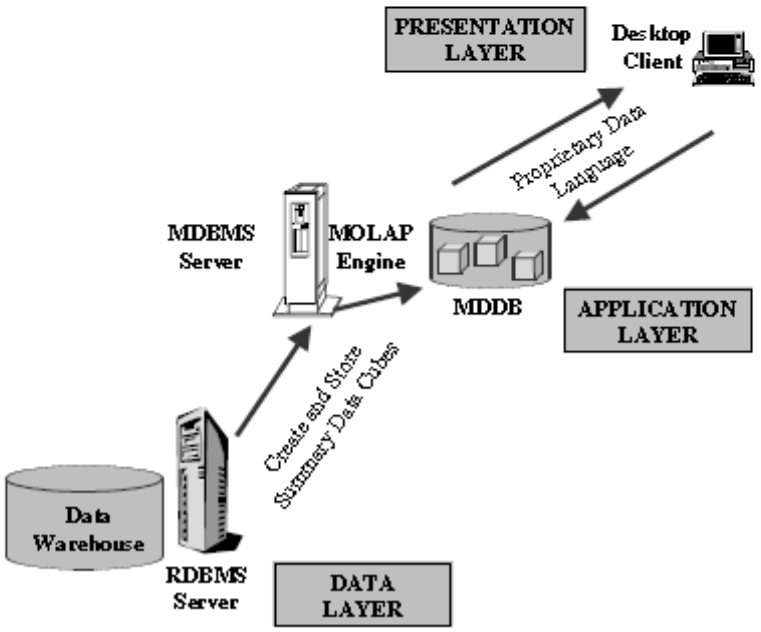


Figure 15-16 The MOLAP model.

The ROLAP Model

In the ROLAP model, data is stored as rows and columns in relational form. This model presents data to the users in the form of business dimensions. In order to hide the storage structure to the user and present data multidimensionally, a semantic layer of metadata is created. The metadata layer supports the mapping of dimensions to the relational tables. Additional metadata supports summarizations and aggregations. You may store the metadata in relational databases.

Now see Figure 15-17. This figure shows the architecture of the ROLAP model. What you see is a three-tier architecture. The analytical server in the middle tier application layer creates multidimensional views on the fly. The multidimensional system at the presentation layer provides a multidimensional view of the data to the users. When the users issue complex queries based on this multidimensional view, the queries are transformed into complex SQL directed to the relational database. Unlike the MOLAP model, static multidimensional structures are not created and stored.

True ROLAP has three distinct characteristics:

- Supports all the basic OLAP features and functions discussed earlier
- Stores data in a relational form
- Supports some form of aggregation

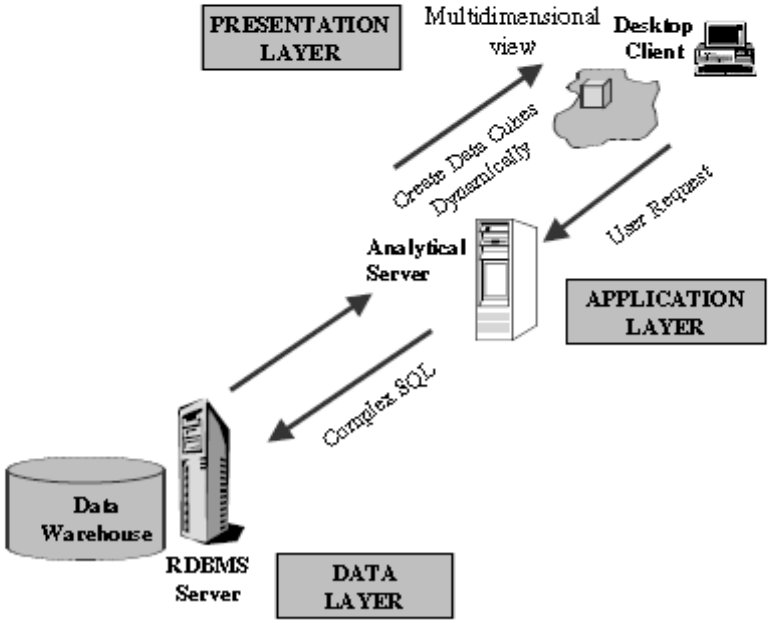


Figure 15-17 The ROLAP model.

ROLAP VERSUS MOLAP

Should you use the relational approach or the multidimensional approach to provide online analytical processing for your users? That depends on how important query performance is for your users. Again, the choice between ROLAP and MOLAP also depends on the complexity of the queries from your users. Figure 15-18 charts the solution options based on the considerations of query performance and complexity of queries. MOLAP is the choice for faster response and more intensive queries. These are just two broad considerations.

As part of the technical component of the project team, your perspective on the choice is entirely different from that of the users. Users will get the functionality and benefits of multidimensionality from either model but are more concerned with questions relating to the extent of business data made available for analysis, the acceptability of performance, and the justification of the cost.

Let us conclude the discussion on the choice between ROLAP and MOLAP with Figure 15-19. This figure compares the two models based on the specific aspects of data storage, technologies, and features. This figure is important, for it pulls everything together and presents a balanced case.

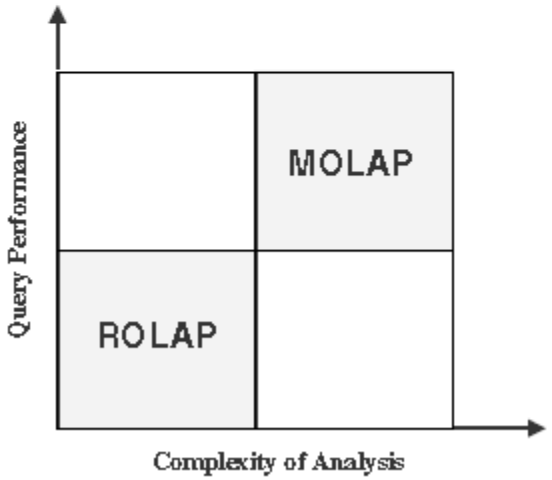


Figure 15-18 ROLAP or MOLAP?

	Data Storage	Underlying Technologies	Functions and Features
ROLAP	Data stored as relational tables in the warehouse. Detailed and light summary data available. Very large data volumes. All data access from the warehouse storage.	Use of complex SQL to fetch data from warehouse. ROLAP engine in analytical server creates data cubes on the fly. Multidimensional views by presentation layer.	Known environment and availability of many tools. Limitations on complex analysis functions. Drill-through to lowest level easier. Drill-across not always easy.
MOLAP	Data stored as relational tables in the warehouse. Various summary data kept in proprietary databases (MDDBs) Moderate data volumes. Summary data access from MDDB, detailed data access from warehouse.	Creation of pre-fabricated data cubes by MOLAP engine. Proprietary technology to store multidimensional views in arrays, not tables. High speed matrix data retrieval. Sparse matrix technology to manage data sparsity in summaries.	Faster access. Large library of functions for complex calculations. Easy analysis irrespective of the number of dimensions. Extensive drill-down and slice-and-dice capabilities.

Figure 15-19 ROLAP versus MOLAP.