Development of Data Warehouse

John Rome

John Rome is director, data administration and data warehouse at Arizona State University.

PRICEWATERHOUSECOOPERS

Managerial Analysis and Decision Support is made possible by generous support from PricewaterhouseCoopers.

Boy, was *Wired* magazine wrong! In April 1996, the magazine stated, "[D]ata warehousing… [is] the hottest buzzword in corporate MIS departments right now. The idea is to collect all the company's information and put it in one big database. Expected lifetime—4 months."[1] It is now known that data warehousing was more than just the technology flavor of the month. Data warehousing has grown into a huge industry and is being used by most Fortune 500 companies as well as higher education institutions. This chapter will focus on data warehousing in higher education. The "Amazon.com" barometer can be used to judge the popularity of data warehousing. Today, there are over 180 books on data warehousing available at Amazon, along with additional titles in similar categories such as decision support and business intelligence. In 1994, the number of books available on data warehousing, most authored by Bill Inmon, the father of data warehousing, could be counted on one hand.

Definition

Inmon's definition of a data warehouse as a "subject-oriented, integrated, time-variant, nonvolatile collection of data in support of management's decision-making process"[2] is still the industry's standard definition. Ralph Kimball, the developer of the dimensional data model, defines a data warehouse as "a copy of transaction data specifically structured for query and analysis."[3] Another early warehouse pioneer, S.G. Kelly, supports Inmon's integration concept but mentions information applications in his definition, "a single, integrated store of corporate data which provides the infrastructural basis for informational applications in the enterprise."[4] In this case a data warehouse is more than just a database or data store: it supports the applications built by the institution for its information needs.

This author suggests the following working definition of a data warehouse: an integrated repository of enterprise-generated, departmentally captured, and/or externally acquired data used to facilitate data access, reporting, and tactical/strategic decision making. Furthermore, the term *data warehouse* needs to have the adjective *good* placed in front of it. A good data warehouse should be the Holy Grail of an institution, especially one struggling with data access or using preformatted green-bar reports off the mainframe. A good data warehouse can identify data problems in time to avoid them, and can locate opportunities that an institution might otherwise miss. However, data warehousing is not always the panacea for an institution's data problems. Many data warehouse initiatives do not deliver what they promise. It is estimated that almost half of all data warehousing projects either are not completed or do not deliver the anticipated benefits.[5]

Planning a Successful Data Warehouse

Project planning is vital to the success of a data warehousing project. Through planning, agreement across functional groups involved in the project takes place. Planning leads to clarity concerning the goals and deliverables of the project. Major activities that need to be performed are revealed and formalized during planning. Those activities can include tool selection, data interrogation, and security implementation.

To Warehouse or Not to Warehouse?

An institution has to justify the establishment of a data warehouse beyond the rationale that "everyone is building one." A formal cost-benefit analysis is appropriate to determine whether the project is worth the effort and the institution's financial investment. The evaluation process should uncover good reasons to build a data warehouse and expose any good reasons not to build one. If an institution has or is planning to implement an enterprise resource planning (ERP) system, the vendor of the system may have a data warehouse for sale. This chapter will discuss vendor-neutral solutions to building a data warehouse, not data warehouse software that can be purchased from ERP vendors.

A good reason to build a data warehouse is to provide access to data. Accessing data in legacy systems can be difficult. Legacy systems are built and optimized for running the operations of the institution, not for data access. In fact, accessing legacy systems may require specialized skills or programming that the typical customer may not have. A good reason to build a data warehouse is that it delivers the right data to the right people in an environment with very few boundaries. Once an institution can access its information, it can serve customer needs more quickly and can be more flexible in delivering services. The "provision of data access" reason alone may trump the reasons for not building a data warehouse.

Consider the reasons for not building a warehouse. First, the cost may be prohibitive. The expense of building a data warehouse for a large college or university could easily reach seven figures, with costs stemming from hardware, software, consulting, and personnel. This is a huge investment, and the benefits must be seriously considered. Another reason for not building a warehouse is that all the data may be accessible through the current legacy systems. Despite its difficulty, this method of data access could make the cost of building a data warehouse unnecessary. Yet another reason to defer a data warehouse project is the complexity of moving the data to a new server and supporting that server. Some institutions do not have the resources or skill level to manage the new warehouse environment.

Another discussion during the evaluation process involves whether to build a data mart instead of a data warehouse. While some believe that differentiating between these is a matter of semantics, data warehouses and data marts have very different definitions and ultimate goals. Data marts are typically built to help with a specific problem or a specific application. They are "stand-alone" applications and are tactical in nature. Since data marts are not concerned with integration and do not follow the formal standards of an enterprise data warehouse, they can be developed more quickly and at a lower cost. Users cannot distinguish between a data mart and a data warehouse. There is a risk in building data marts without keeping the enterprise in mind. Data marts do not completely integrate with the enterprise data warehouse, and they might duplicate effort.

Goals and Measures of Success

While the reasons for building a data warehouse include providing easy access to data and solving particular business problems, the primary goal is obvious. The data warehouse must integrate the institution's data and make them consistent across the enterprise for reporting purposes. This goal will bring additional benefits, which include better data administration and improved data management capabilities. These improvements occur when information is accessible. If increased exposure of data points to quality problems or to standards not followed in the legacy systems occurs, the likelihood that these problems will be addressed also rises.

For the warehouse initiative to be successful, goals need to be established within boundaries. The scope of the project should be manageable even as it aims to satisfy the needs of the business area. While it is attractive to try to build the entire "financial" system, for example, in a data warehouse, tackling a particular business process or area (e.g., Financial: Accounts Receivable, or Student: Admissions) is a more achievable goal. Consider looser data delivery schedules and milestones for a data warehouse project. The development of a data warehouse is iterative in nature, and this fact should be taken into consideration during project planning. Leave room for plenty of data warehouse discovery during development. The end of the fiscal year and the start of the academic term are some target completion dates to be considered.

Success can be measured by customer use of the data warehouse. Audit mechanisms within the database can capture who is using the data warehouse and how often. Departments using the data warehouse and peak and nonpeak use times can be figured out from this information. Not only is tracking this kind of information important in justifying the data warehouse, but it can also help in planning the direction of future development. The audit information shows what data are being used most and where to concentrate developmental efforts. While use of the warehouse is important, its ultimate indicator of success is that it becomes integral to the institution's decision-making process.

Tool Overview

Important software and other tool decisions must be made during the development of a vendor-neutral data warehouse. It is essential to know the types of tools available. While this software requires a monetary investment, the time savings and other benefits to the development process can be enormous. Many of the mundane tasks of old can be automated with these tools. Those to consider are categorized as data modeling tools; data extract, transform, and load (ETL) tools; database software; and reporting/business intelligence tools. A list of some of the tools that fall into these categories can be found in Table 1.

Table 1.

Examples of Data Warehouse Tools

Data Modeling Tools
- ERwin (Computer Associates)
- Power Designer (Sybase)
- ER/Studio (Embarcadero Technologies)

Data Extract, Transform, and Load (ETL) Tools
- PowerMart (Informatica)
- DataStage (Ascential Software)
- DT/Studio (Embarcadero Technologies)

Database Software
- Oracle (Oracle)
- SQL Server (Microsoft)
- Adaptive Server (Sybase)

Reporting/Business Intelligence Tools*
- Business Objects
- Cognos
- Brio Software

* Vendors offer both desktop and Web versions.

Data modeling tools. A modeling tool helps organize the data that will go into the data warehouse and flushes out business rules in the process. In such a tool, entities and their relationships are built with a graphical user interface (GUI). Conceptual and physical data models are typically supported. The tool automatically generates the code needed to build warehouse tables and indexes. While there is some debate about the appropriate data modeling technique to use in warehouse development, data modeling tools support the various techniques. Also known as upper-case tools or entity/relationship modeling tools, they typically cost in the thousands of dollars.

Extract, transform, and load (ETL) tools. The days of writing COBOL programs for data warehouses may be numbered. An ETL tool helps automate the movement of source data to a target database, in this case, the data warehouse. ETL software combines three separate “data” functions in a single programming tool. First, the *extract* function selects data from a source database and extracts the desired data. Next, the *transform* function works with the extracted data using a variety of techniques to convert them to the desired state. Finally, the *load* function writes the resulting data to a target database. The target tables are typically created with data modeling tools and exist on the data warehouse database. Most have a GUI interface that helps build the ETL jobs, as well as an engine that schedules the extracts and loads. Tools in this category range from $10,000 to $200,000.

Database software. A relational database management system (RDBMS) is usually used to store warehouse data. An RDBMS stores data in the form of related tables. RDBMSs are reliable, well understood, and commonplace in information technology. These systems support structured query language (SQL), the lingua franca of data access in relational databases. Most reporting tools use SQL to access information. Typically, an institution and its information technology organization will have standardized a particular RDBMS. Multidimensional databases (MDDBs), though not as popular, are an alternative to RDBMSs. An MDDB is very efficient because it stores precalculated data, but not all reporting tools support access to an MDDB. MDDB software includes Hyperion Essbase OLAP Server and Microsoft’s Analysis Services.

Reporting/business intelligence tools. These tools are used for querying, data mining, or online analytical processing (OLAP) analysis. They "run queries" against the data warehouse, allowing manipulation of the data within the tool and exporting capabilities to a familiar tool like Microsoft Excel. While these tools have historically been available on the desktop, some of them now have a Web component. This makes desktop configuration easier, since all that is needed is an Internet browser. For the Web, a separate application server is necessary, which manages the queries sent to the data warehouse. The Web client software is a Java implementation or a plug-in-type application. Reporting tools from the major vendors cost about a thousand dollars per workstation. The equivalent Web products from these vendors are significantly lower, but the cost of ownership then shifts to the server application software. Site licenses or volume discounts are usually available.

Understanding the Data to Be Warehoused

Achieving good data quality in a data warehouse is difficult. Data can be "good," "bad," or "ugly," thereby posing a number of challenges. Many institutions overestimate the quality of the data in their legacy systems. Known problems may be lurking in the data: domain conflicts to solve, misspellings to correct, missing data to find or derive, and accuracy to be determined. The data that are found might not be the data that are expected. Achieving good data quality is a difficult, lengthy, and often contentious process. Achieving quality in a data warehouse, though, may be easier than doing so in an institution's legacy systems.

Similarly, institutions typically miscalculate the difficulty of transforming legacy data for use in the data warehouse. The transformation (or ETL) task is difficult if data are spread across complex, disparate systems. There may be many instances of the "system of record." The data may be found in a nonrelational legacy online transaction processing (OLTP) system, in an extract file, or as a replication in another database. Trying to determine the best source for the data warehouse is difficult.

While many of the data repositories in a higher education institution are candidates for inclusion in the data warehouse, the major subject areas are students, financial matters, human resources, and possibly research. Data within these areas are typically the first to be developed in a data warehouse initiative. Financial and student data are good candidates for starting the process, because there is enormous interest in customers (students) and money matters (financial). Many institutions that have developed warehouses have started with financial data simply because the system can be more easily justified by the nature of the data. Other data areas important in higher education include alumni, courses, financial aid, and facilities. Choosing the most effective and actionable data to include in the data warehouse will contribute to the project's success. As a data warehouse matures and includes an increasing number of subject areas, external or purchased data may be added. External data may include U.S. Bureau of the Census or peer institution data. Figure 1 (page 11) contains many of the higher education subject areas to be considered.

Security

Security is not free. It takes clear institutional policies and commitment of time and resources to manage. As with the security of other systems, data warehouse security must be based on institutional policies that address data access and appro-

Figure 1.

Higher education subject areas

Students	Financial	Human Resources
Alumni	Facility	Course
Directory	Financial Aid	Official
Property	Sponsored Research	User Data
Department Data	External Data	Dictionary Metadata

priate use. Higher education institutions are bound by various external laws and policies that dictate security requirements. For instance, the federal Family Educational Right to Privacy Act (FERPA) protects the privacy of a student's educational records and places specific limits on how student data can be used. If such policies are not already in place, the warehouse project will benefit greatly from their establishment in collaboration with the institution's legal counsel.

Additional policies can help determine the process by which users obtain access to the data warehouse. Access and usage policies run the spectrum, from restrictive— "A person can have only the one thing they told me they needed, and if they need something else, they must tell me first!"—to very broad— "Anyone can have anything they want for any purpose." A policy that tends toward the broad side is recommended because of the nature of a data warehouse. The institution must consider the level of trust it can place in its users, the potential for liability or risk associated with misuse of the data, and the staff and technical resources available for enforcing policies.

The next step is to establish an access request process, which involves data trustees. A data trustee is a person who is knowledgeable in a particular data area and generally has authority to make decisions related to the development and access of the data associated with the business function. Data trustees may already be in place for the operational or legacy systems from which the data originated. Trustees typically help create data policies, raise security awareness, and approve an individual's access to data and systems. Data trustees have an important role in monitoring data access and data misuse.

The technology used to enforce security policies varies with the institution. Many components may play a role, including relational database security (user IDs, passwords, and groups), file security, row-level security, network security, and personal computer security. The technical architecture of the institution can dictate the security implementation. Careful design of the data warehouse (e.g., integration of security profile information) can make security easier to maintain. Some institutions implement a "single sign-on" to the data warehouse or network, with access based on the role of the individual. A primary goal is to secure the network connection from the warehouse to the user, which can be accomplished by using a virtual private network (VPN). The VPN connection encrypts all the communication on the network. With the VPN, outside "listeners" cannot snoop or make sense of the communications. Firewalls and Secure Sockets Layer browser-embedded encryption technology may be part of the security plan.

Steps to Successful Implementation

Once project boundaries have been set, readiness has been assessed, and funding has been approved, implementation begins. The following steps are necessary for a successful data warehouse implementation.

Creation of a Steering Committee

The steering committee should have a vision for the potential impact of a data warehouse project. The members of the steering committee are typically senior staff at the institution who have the financial resources and ability to garner the manpower necessary to contribute to the project. Information technology should be represented on this committee, as well as the data administration/data management group. It is essential to include the business sponsor as part of the committee. The business sponsor is the ultimate client of the particular data and has great interest in the success of the project. The steering committee membership will change over time as the development subject areas (i.e., data) change. The committee's job is to determine funding and staffing allocation, resolve major issues that arise, and communicate with the project team.

Selection of a Project Team

The project team is generally created from individuals in cross-functional areas who combine a variety of skills and institutional knowledge. This team should be both dedicated to and passionate about its work to ensure success. Star technical staff and effective communicators must be included on the team. The project team members will often have to continue other job duties while working on the warehouse project. It should be clear that the warehouse priority is as high as other current job tasks. Outside consultants can be part of the project team, but the institution must retain ownership of the project once the contract is over. If consultants are hired, skill and knowledge transfer must occur during the project and before they leave.

Technical and business staff must be brought together for the success of the data warehouse. Other higher education departments that play an important role in the data warehouse project include institutional research, data administration/data management, and planning/budgeting. These units understand the importance of data and information. They have been involved in integrating data across systems and can serve as great resources in the development of the warehouse.

Some fortunate institutions have been able to set up funded data warehouse departments. These departments may go under monikers such as "decision support" or "business intelligence." The advantage of having such a group is that it is clearly dedicated to data warehousing activities. The titles and roles of the staff in this type of department are similar to the ones mentioned in Table 6.2. Such a department should not be treated as temporary. The demand for expanding or maintaining the data warehouse will continue long after the project is officially finished. The new department may be, but does not have to be, created within the information technology department.

The clarification of roles and responsibilities is crucial to project success. A person on the data warehouse project commonly has more than one role, either because the individual has particular skills and experience or because there is a shortage of personnel. There may be overlaps and gaps in responsibilities that the team members will need to negotiate over time. In smaller institutions, one person might fill all the major roles found in Table 2.

Table 2.

Data Warehouse Roles
Data Warehouse Manager
Business Sponsor
Technical Architect
Data Architect
Database Administrator
Business Analyst
Data Transformation Analyst/Programmer
Business Intelligence Analyst
Data Educator/Trainer

Data warehouse manager. The data warehouse manager is in charge of the day-to-day management of the project. This person works closely with the steering committee. The individual should be respected, experienced, and able to work well with both the business areas and the technical staff. He or she also manages the ongoing support of the warehouse once the components are built. The success of the project hinges greatly on the ability of this person.

Business sponsor. Understanding the business requirements of the project is a must for the business sponsor. This person realizes the value of the data warehouse and is a strong promoter. He or she may not be on the project full-time but must be truly committed. This individual is also part of the data warehouse steering committee and often serves as a data trustee, granting access to users.

Technical architect. The technical architect ties together the technical pieces of the project, which might include architectural design, product evaluation/support, and security. Coordinating the data staging of the ETL processes and the deployment of the business intelligence tools is one of the responsibilities of this position. This individual does not have to be an expert in every technological detail but should know how all the pieces fit together.

Data architect. The data architect is responsible for data modeling and understanding data issues for the data warehouse project. Fluency in the various design methods of data warehousing, including dimensional and tabular design, is a must. Knowledge of the legacy system's data models and data is an extremely valuable asset for this role. This person works closely with the database administrator.

Database administrator. The database administrator (DBA) is in charge of building and maintaining the warehouse database. He or she helps the data architect implement the warehouse design, creating actual database tables. The DBA has to be willing to set aside some traditional thinking (e.g., that all databases should be to the *n*th form of normalization). This individual is in charge of the day-to-day operation of the data warehouse database.

Business analyst. The business analyst knows his or her business well and translates business knowledge into terms the technical staff can understand. An individual with a strong information technology and technical background often fills this role. Unlike the business sponsor, this person works more than part-time on the project. He or she should have strong communication skills and should be respected by the business area represented.

Data transformation analyst/programmer. The data transformation analyst/programmer is responsible for getting data to the data warehouse. He or she is responsible for the beginning-to-end design of data movement from the legacy system to the warehouse. Responsibilities may entail finding and mapping the source data to the target data. This position may require extensive programming, especially if no ETL tool is available.

Business intelligence analyst. The business intelligence analyst works closely with the reporting tool used for the data warehouse. This person spends time trying to understand the data as well as the reporting tool. Knowledge of SQL and the ability to develop ad hoc reports are essential. This person may be a power-user selected from one of the business areas and may perform customer support.

Data educator/trainer. As the data warehouse starts to take shape, a person who worked closely on the project team must take the role of data educator/trainer. This person has a good understanding of the business, is familiar with the data reporting tool(s), and has experience as a trainer. The data educator/trainer role is often overlooked or underutilized. This individual develops the initial training material and supports ongoing training.

Data Delivery/Reporting Requirements

Customers will want their data and reports from the warehouse delivered by various means. The project team may offer all or several of these alternatives to warehouse customers. Some customers will want parameter-driven applications to deliver their data or reports, preferably via the Internet. Some will want their reports mailed to them on a routine basis as a Microsoft Excel spreadsheet or in an Adobe Acrobat file. Many will want to build their own ad hoc queries and use the capability of the reporting tool, or export it to Microsoft Excel or Microsoft Access. A few customers will use sophisticated modeling or forecasting tools to access their information. Some may even get information from the warehouse and load it back into a legacy system where a statistical tool can be used.

A data warehouse can build on an institution's earlier attempts to construct a reporting database. Reporting requirements gathered from that attempt can serve as a starting place for data warehouse development. One technique in gathering reporting requirements for a data warehouse is to ask customers to make lists of questions they want answered. The data required can be determined from these lists of questions. Another technique that provides insight into data needs is to ask customers to show the reports they use. It is important to verify which reports are actually used in order to limit requirements. Finally, it is useful to find out what decisions the end users want to make with the help of the data warehouse. Sometimes reporting requirements are determined by the shortcomings of application packages or ERP systems.

Design of the Data Warehouse

Forget all you've learned about data modeling. For the data warehouse, there is no need to try to achieve the "third normal form" of normalization by eliminating redundancy. Data modeling for the warehouse is much different, and "managed" redundancy is an acceptable modeling practice. In fact, some "denormalization" is desirable, which means the same data items appear in multiple tables. This makes user interface with the data warehouse simpler and improves warehouse performance by reducing the number of table joins required in frequent queries.

In higher education, many institutions are willing to share their data models. Other institutions can use these shared examples as a guide to jump-start the designing of their own models. Ronald Allan of Georgetown University, in conjunction with EDUCAUSE, manages a Web site of higher education data warehouse configurations (http://www.georgetown.edu/users/allanr/dwconfig/). On this Web site, the primary data warehouse contacts are available. The site also provides information on what databases and modeling tools are used at particular institutions. A few books with generic data models are available from a variety of industries, including education. The second edition of the *Data Warehouse Toolkit* (2002) by Ralph Kimball contains accounting, human resource, and educational models.[6] Data models can also be found in *Data Warehouse Design Solutions* (1998), written by Christopher Adamson and Michael Venerable.[7] Another great resource is Len Silverston's *Data Model Resource Book* (2001).[8]

Two common techniques for modeling data warehouses are dimensional modeling and tabular modeling. The determination of whether granular or summary data are needed might influence the type of modeling technique used for a data warehouse project. If more granular or detailed data are required, then tabular modeling is the suggested modeling technique. If summary data are needed, then dimensional modeling is the technique to use. Dimensional design often contains only summary data, while tabular design usually contains detailed data as well as summary data.[9] Many higher education institutions have taken a hybrid approach, including both tabular and dimensional modeling and choosing the best design for the particular situation. Tabular modeling predates dimensional modeling, so many institutions are just starting to discover the dimensional technique.

Tabular modeling. Tabular-style tables look very similar to tables found in normalized transactional databases.

However, they are constructed following some loose guidelines. In a tabular design, an element of time may be added to the table to allow tracking of historical data. This might be a fiscal year, pay period date, or academic term code. Since true normalization is discouraged, derived columns may be added to the table. For example, many columns must be interrogated to identify “research accounts.” In a tabular design, inclusion of a “yes” or “no” flag in the table makes the research accounts easier to find. Tabular design includes all the data that will answer the questions posed to the warehouse. Tabular tables can become rather large because of the number of columns they may contain.

Tables may be combined in a tabular data model. For example, a vendor has multiple contact persons for its company. In the operational system, this information may be found in two tables that allow an unlimited number of contacts. In a tabular design, placeholders for the maximum number of contacts are put in the table. While there may be some unused columns in the table, there is no need to use another table to get contact information. Another guideline for a tabular table is to include both codes and descriptions. For example, storing both the six-character expenditure code *711001* and the description *Salary* makes it easier for a user to find that information. The user will not have to use another table to look up the code description. While some critics point to the extra storage needed to follow these guidelines, it is all about making it easier for a user of the warehouse to get information. An example of a tabular model appears in Figure 2. (See page 21.)

Dimensional modeling. A dimensional model is made up of one *fact* table and many *dimension* tables. The fact table contains one or more numeric columns that support numeric analysis: dollar amounts, average figures, or counts. Numeric

Figure 2.

Tabular design sample

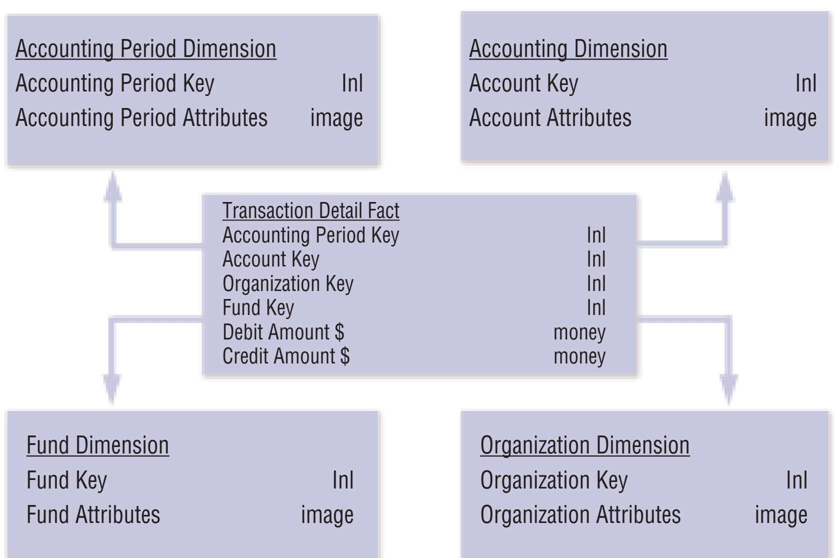
Account	
Fiscal Year	char(4)
Account	char(7)
Description	char(30)
Financial User ID	char(4)
Manager Affiliate ID	char(10)
Manager Full Name	char(30)
Accountant Agency Code	char(3)
Accountant Name and Phone	char(30)
Officer Agency Code	char(3)
Officer Name and Phone	char(30)
Agency Code	char(3)
Agency Code Description	char(30)
Organization	char(4)
Rollup Account	char(7)
Rollup Account Description	char(30)
Rollup Organization	char(4)
Fund Code	char(4)
Fund Code Description	char(30)
Fund Group Code	char(1)
Fund Group Code Description	char(30)
Fund Category Code	char(3)
Fund Category Code Description	char(30)
Fund Class Code	char(3)
Fund Class Code Description	char(30)
Campus Code	char(1)
Activity Code	char(4)
Activity Code Description	char(30)
State of AZ Program Code	char(4)
State of AZ Program Code Description	char(30)
Activity NACUBO Code	char(4)
Activity NACUBO Code Description	char(30)
Activity Group Code	char(2)
Activity Group Code Description	char(30)
A21 Basic Distribution Code	char(1)
A21 Basic Distribution Code Description	char(30)
A21 On Off Campus Code	char(1)
A21 On Off Campus Code Description	char(30)
A21 Activity Code	char(1)
A21 Activity Code Description	char(30)
Grant Account Code	char(1)
Financial Group Code	char(4)
Financial Group Code Description	char(30)
Account Status	char(1)
Multiple Year Flag	char(1)
Extract Datetime	datetime

Transaction Detail	
Fiscal Year	char(4)
Fiscal Quarter	int
Fiscal Month	int
Budget Year	char(4)
Batch Number	char(6)
Transaction Document Code	char(2)
Transaction Number	char(11)
Recorded Date	datetime
Acceptance Date	datetime
Document Action Code	char(1)
Line Action Code	char(1)
Order Type Code	char(1)
Vendor Code	char(11)
Document Description	char(12)
Line Number	char(2)
Line Description	char(30)
Account Type Code	char(2)
Fund Group Code	char(1)
Fund Code	char(4)
Account	char(7)
Sub Organization	char(2)
Reporting Category	char(8)
Appropriation Unit	char(9)
Appropriation Unit Code	char(2)
Area Code	char(3)
Expenditure Code	char(6)
Object Code	char(4)
Revenue Code	char(6)
Revenue Source Code	char(4)
Balance Sheet Account Code	char(4)
Reference Transaction Code	char(2)
Reference Transaction Number	char(11)
Reference Transaction Line	char(2)
Series Number	char(3)
Amount $	money
Debit Credit Code	char(1)
Random Sequence Number	int
Extract Datetime	datetime

columns (facts) can be summarized, averaged, or counted. The dimension tables are lookup tables. The dimensions describe the fact table. For example, an account dimension helps define the general ledger journal entry fact table. A time or date dimension is almost always associated with a fact. This date dimension is integral to all fact tables needed for a dimensional design. The dimension's foreign keys (often artificial rather than natural keys) make up the uniqueness of the fact table. Because the dimensions are organized around the fact table, making the model look starlike, the dimensional model is sometimes known as the "star schema." Ralph Kimball is the father of dimensional modeling and has written many books on this technique. An example of a dimensional model is shown in Figure 3.

Figure 3.

Dimensional model design



Creation of a Prototype

Prototyping promotes further understanding of the technology and concepts in a data warehousing initiative. This valuable activity is the institution's first exposure to the benefits of data warehousing. A prototype contains a subset of data that might be content material for the enterprise data warehouse. The prototype allows the project team to "get its feet wet" in technologies such as ETL tools, relational databases, and reporting tools. The timetable for the prototype is noncritical. However, it should be built very quickly to allow users to work with live data and help the project team identify and refine the requirements for a full system. The steps of a prototype follow.

- Develop it quickly.
- Acknowledge that first iterations will have problems.
- Seek top-down/bottom-up feedback and document it.
- Modify the next iteration.
- Keep building.

Find volunteers to participate in a pilot user group. These individuals must have some technical skills as well as an understanding of the institution's data. Choose pilot users in partnership with their supervisors, who must realize the benefits of their participation. Look for enthusiastic staff members who are willing to try something new and better. Give these individuals access to the data at no cost, provide one-on-one training, and communicate constantly with them. This will help meet the goals of the prototype and provide something to build on for the "full-blown" data warehouse.

Setup of the Production Data Warehouse Environment

A data warehouse environment needs an architecture. Whether formally planned or informally designed, it shows how all the components fit together and work over time. The architecture is an evolutionary process: it will take shape during the process of learning more about the environment, solving immediate problems, gaining a better understanding of the data issues, and scheduling ETL processes. Setting up both a development and a production environment should be part of the architectural plan to allow testing and development to continue without affecting the customer. Some major components of the establishment of the production data warehouse environment follow.

Server. No longer are mainframes needed to run applications such as data warehouses. Windows NT/2000 and Unix/Linux have brought mission-critical stability to the server. Servers have become easier to set up, and the cost of ownership is dropping. Server selection is critical to the data warehouse, since good query response time is expected. It is important to evaluate the server's processing power and scalability (ability to expand). As the demand for data warehousing grows, scalability allows the addition of processors or memory to improve performance. It is of paramount importance to choose servers and operating systems that meet more than just the original requirements (in terms of the number of users or amount of data) of the data warehouse project. Popular warehouse server vendors include Sun, Hewlett-Packard, Dell, and IBM. The database software runs on the server.

Automation of loads. The most labor intensive and tedious activities in a data warehouse project involve building the ETL processes. Setting up a process to get data from the legacy system, transform them, and then load them into the data warehouse takes planning, time, and patience. Once in place, this process should involve as little human intervention as possible. The process will be different if an ETL tool is performing the function or if most of the transformation is done through a programming language such as COBOL. The ETL tool will take care of the extraction, movement, data staging, and loading into the data warehouse. Otherwise, an extraction program needs to be written, a routine to transfer the data to the warehouse server must be created, and scripts need to be written to load the data. Procedures must be in place to handle problems in the ETL process. This might require individuals to be on call to troubleshoot and fix problems. Regardless of whether the ETL process is software based or manually built, bulk loading of data to the data warehouse should be used. Record-at-a-time processing is too slow for data loading and could affect availability.

Maintenance and support. Ongoing "care and feeding" of the data warehouse must continue beyond the initial construction phase. Maintenance and support requires a whole gamut of activities. Technical and backroom operations need attention, which includes making sure the servers are running, the network is available, and the database is optimized. Other tasks include ensuring that the transformation processes are running and the loading of the databases is optimized. Monitoring the warehouse for poorly performing queries or runaway processes is part of this support.

Support and maintenance should center on warehouse customers. It is important to keep them happy. Institutions should consider giving customers a Web site with data warehouse information, tips and techniques, and development status reports. Another suggestion is to create a data warehouse users group. This vehicle can be used to share ideas and provide a supplement to training. Customers can even participate and influence the direction of the group. E-mail is a good medium for discussing project updates, broadcasting warehouse maintenance, and announcing problems. The bottom line is that nothing should get in the way of helping a customer use the data warehouse.

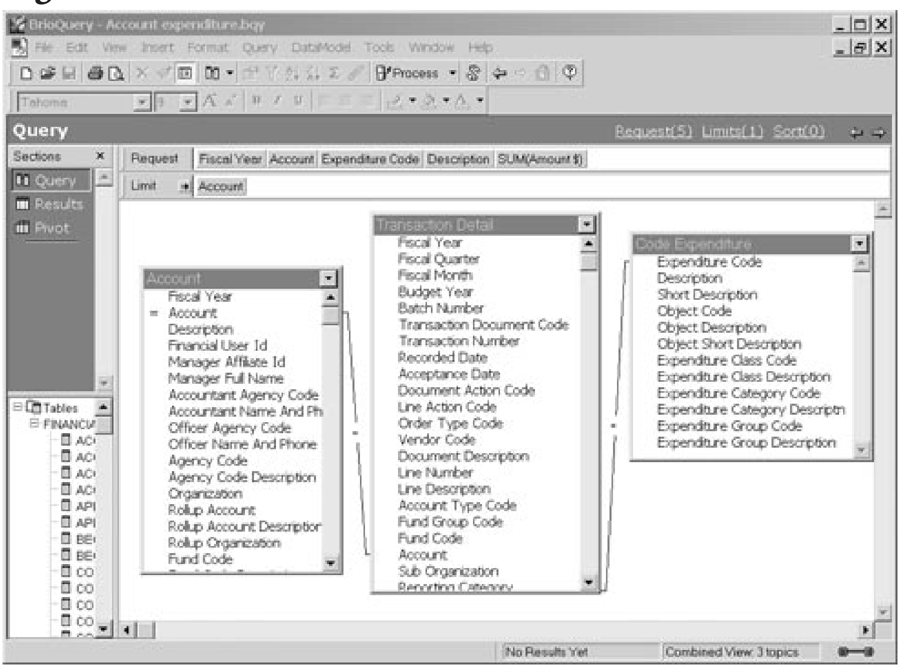
Benchmarking Performance

As applied to a data warehouse, benchmarking performance is the process of identifying and using best practices to improve query response time. Two types of queries are run against a data warehouse: static and ad hoc queries. Static, or "canned," queries contain known SQL. These types of queries are predictable, are run at expected times, and can easily be optimized for performance. Ad hoc queries, on the other hand, are not predictable, especially in terms of when they are run and the resources they need. Ad hoc queries allow the user to ask "what if" and develop investigative scenarios, which make them powerful. However, the SQL created in ad hoc queries might not run well against the warehouse. Inefficient ad hoc queries, or "queries from hell," affect not only the customer running the query but all customers. There needs to be a balance of database optimization and tuning that supports both types of queries. Optimizing for static queries only could adversely affect ad hoc queries, and vice versa.

Adding indexes is one technique for improving the performance of a data warehouse. Indexes are recognized search paths within a database. The most frequently limited columns in a query (SQL "where" clauses) are good candidates for indexes. For example, a vendor table may contain an index (primary key) of vendor codes, but the search path is typically by vendor name. The vendor name column is a good candidate for an index. While an operational table in a legacy system may have only one or two indexes, a data warehouse table could have a handful of them. There are several index technologies that an RDBMS may support, including B-tree indexes and bit-mapped indexes.

Many institutions deliver their warehouse data via the World Wide Web. The Web has a different expectation level for performance than a desktop reporting tool. For the Web, near-instantaneous performance is needed: "A page has to deliver the first screenful of its useful content in less than ten seconds."[10] If it takes longer, there should be a useful message or progress bar telling how much longer the wait will be. The requirement for instantaneous performance from the Web is not negotiable.

Figure 4.



Selection of a Reporting Tool

A reporting tool is one of the major tool purchases for a data warehouse project. From the customer's perspective on the data warehouse, the selection of the reporting tool is the most important decision an institution makes. Customers do not care what database is used, how the data was loaded to the warehouse, or how the data warehouse was modeled. The reporting tool determines how customers interact with the data warehouse and the interface they will see when they access the warehouse. A sample of a reporting tool can be found in Figure 4. A variety of customers with a wide range of computer skills should be included in the reporting tool selection process. Customers using these tools during an evaluation or pilot period can point out any problems or issues that they encounter. Furthermore, finding out which tool the customers prefer can help ensure project success. The selection process should ensure that the reporting tool

- is easy to use;
- supports RDBMS middleware and open database connectivity (SQL);
- works with multiple databases and platforms;
- exports results to various formats (Microsoft Excel, delimited, etc.);
- supports ad hoc and static query environments;
- is Web-enabled; and
- is affordable.

It is important to make sure that the reporting tool vendor is financially solid. Information can be found on the Internet, in an annual report, or by visiting and talking with companies or universities using these reporting tool products. The reporting tool vendor will be happy to provide the names of satisfied customers; it is more useful, though, to find the "not-so-happy" customers. These customers are usually willing to share their experiences and point out the problems. Once a particular company and product are selected, this "bad" information collection might prove to be handy ammunition during negotiations. Nigel Pendse and Richard Creeth have a Web site (www.olapreport.com) that contains a variety of information including market shares of the various reporting tools and news about these vendors. Industry analysts such as the Gartner Group, Giga, and the Meta Group offer information on reporting tools, with a subscription or fee to belong to these groups and to obtain the reports. Those selecting reporting tools should check to see whether their institutions already have a membership.

End-User Training

A data warehouse makes data accessible to staff, many of whom are not knowledgeable about the data or about query tools and languages. To learn how to use a data warehouse well and get the maximum benefit from it, staff need training and assistance. In fact, training is essential if the project is to be successful. It is important to include business analysts in the preparation of the training material. These businesspersons know their data well and can be very beneficial to the data trainer/educator. A touch of marketing helps in this training endeavor, too, especially if the new users are already very busy in their jobs. Several topic areas for training that will benefit users are listed below.

Data and the Reporting Tool

Users often need more training about the data than about the tools used to access the data. Training that incorporates the actual data as users learn about the reporting tool is suggested. It is ineffective to have business operations managers learn to query on a typical sales database that has nothing to do with their job function. Instead, have customers work with the actual data they will use and show them techniques that will enhance their job performance. Hands-on training is the preferred technique for data and tool user training.

SQL and Relational Theory

While these topics should not be mandatory, knowing SQL and relational theory might be useful to some data warehouse customers, especially power users performing ad hoc analyses. With this theoretical background, power users can take advantage of built-in SQL functions in the database or create more complex queries, such as subselects, than the reporting tool will allow with the GUI interface. Sometimes a user will get unanticipated results using the query tool. With the extra knowledge, a user can troubleshoot the SQL generated by the reporting tool without any help. Techniques such as "left joins" are easier to understand for customers who have some relational theory and SQL background.

Boolean Logic

Not understanding Boolean logic is the cause of many wrong query results. Combining the Boolean operators or, and, and not to get information from the warehouse can become complicated. Users need to learn how these operators work and how the use of parentheses affects them. With an understanding of Boolean logic and concepts, the desired results can be achieved more easily in complicated queries (such as "I want a list of all the vendors that reside within the state, or that we have purchased computers from, but not

Acme Computers").

Customers historically have seen data in canned reports generated by information technology; data in "raw" form as presented by newfangled reporting tools can be confusing. That is why end-user training is so important. Data are rarely self-explanatory, and metadata may not provide all the answers. Healthy skepticism regarding data results and eyeballing of the data are some good training techniques to be shared with users. Many users assume that their query results are correct when they are not. Teaching these topics and techniques can help build confidence and minimize bad results. In fact, it is a good idea to consider a "no training, no access" policy for the data warehouse.

Analysis of Results

Rather than simply hope that customers have confidence in the warehouse and that it will be available, effective, and reliable, it is prudent to solicit feedback on the data warehouse. The accuracy of the data and the content of data warehouse reports must be continuously monitored. Unless the data reconcile precisely with official numbers and totals from the legacy system, warehouse customers will quickly lose confidence in the data warehouse and look for other ways to meet their information needs. The data warehouse project team must be prepared to deal with data issues quickly and efficiently.

Establishing and maintaining these reconciliation processes can be time consuming and tedious, but it must be done. It's hard to analyze and collect quantifiable information about the benefits realized from a data warehouse. Since a data warehouse project requires so much work, relying on unsolicited feedback from warehouse customers may be the only option. Well-designed user surveys can provide valuable feedback and insight into the data warehouse and customers' needs. Management feedback is imperative as well. Regardless, satisfied customers are willing to sing the warehouse's praises, but they may need to be asked.

Lessons from a Successful Data Warehouse Implementation

This author's institution has learned many lessons along the data warehouse development path. Hopefully other institutions can learn from the experience. Table 3 presents the current list of lessons, which is constantly evolving.

Table 3.

Data Warehouse Lessons Learned
Secure a budget for the warehouse project.
Make sure you have a historical data plan.
Build in integrity and integration.
Remember that data quality is not as good as you think.
Find ways to capture useful metadata.
Be aware that customers will want more granular data with more frequent loads.
Allow customers the opportunity to use their local data.
Support all tools but standardize on one.
Make customers happy.
Invest in end-user support structure.

Secure a budget for the warehouse project. Tangible benefits are difficult to estimate for a data warehouse initiative, and intangible benefits may even be dismissed. Regardless, building a data warehouse requires more than a shoestring budget. A substantial investment beyond the assignment of staff to the project is required to ensure the success of the project. The one-time costs include capital purchases of hardware and software tools, and the ongoing costs include hardware upgrades and maintenance and software maintenance (usually 15 percent annually). A development environment that mirrors the production environment is required. Some of these costs, such as that of the reporting tool, can be transferred to the customer, and a charge-back system can generate funding based on use.

Make sure you have a historical data plan. In legacy systems, new data tend to be more important than old data. Warehouse customers, though, will want to see more than just new or current data. Having the ability to compare data over time or see trends is important. To accomplish this, the warehouse must have historical records, which can be provided by building a time or date dimension into the design. Some types of data (e.g., IDs and ethnicity) may change over time, and a historical plan will capture this. Popular time dimensions in higher education include fiscal year, academic term, and pay period date. With historical data in place, charting five years' worth of a department's expenditures to anticipate future spending is easy.

Build in integrity and integration. A data warehouse may acquire data from numerous operational sources. The department code of the ERP human resources system may be different from the department code of the financial system. The data warehouse offers the opportunity to build in integrity and integration. The warehouse requires standardized code sets, data formats, domain values, and so on. In some cases, standardization can become a complicated political issue. Trying to get institutional agreement on a common department, academic structure, or facility structure takes time, energy, and commitment. Another difficult obstacle is the integration of the various time dimensions, such as the fiscal year and the academic term, especially if an academic term spans more than one fiscal year. Addressing these integration challenges yields huge rewards. Consider using national code structures (such as the U.S. census or IPEDS) throughout the data warehouse if they are available.

Remember that data quality is not as good as you think. The fact that the legacy system is running smoothly does not mean an institution has quality data. Data problems often go undetected until the data are visible in a data warehouse. The warehouse can uncover problems such as bad dates (e.g., a student who appears to be only three years old), and overvalued property (e.g., a desktop computer with a book value of $6 million). The data warehouse is not the best place to fix the data, since problems will continue to be introduced. The fix must occur at the source, which is usually the legacy system. Data quality should be taken personally. An institution does not want to release numbers that are wrong due to poor data quality. Preventing internal or external embarrassment should not be the sole incentive for improving data quality. It is important to question the quality of data continuously: "The business cost of nonquality data, including irrecoverable costs, rework of projects and services, workarounds, and lost and missed revenue may be as high as 10 to 25 percent of revenue or total budget of an organization, according to Larry English."[11]

Find ways to capture useful metadata. Just as DNA carries the genetic makeup of a person, metadata contains information on the creation and maintenance of a data warehouse. Warehouse development tools automatically capture volumes of metadata, including data element descriptions, data type information, and domain values. While these metadata are helpful, especially to the information technology staff, the most useful metadata are those that are manually created and maintained. These might include special insights or rules that are inherent to a data item, such as having all vendor codes that represent minority-owned businesses start with a specific number.

Metadata are not exclusively available to developers: several reporting tools have the ability to show warehouse users metadata.

Be aware that customers will want more granular data with more frequent loads. Customers will not be satisfied with summary data. They want to be able to see the detailed transactions that make up the summary data.

Information-hungry customers will demand more frequent updates of the data warehouse (monthly becomes weekly becomes daily) with the ability to drill to detail. The design of the warehouse tables can help with this need.

Tabular-designed tables lend themselves better to detailed data. Nonetheless, data feeds must coincide with decision-making timetables (e.g., daily information would be appropriate for student admissions application totals). The trick is to anticipate the timing of data needs and not perpetuate a reactive mode of business.

Allow customers the opportunity to use their local data. Customers may have data that they capture locally that needs to be combined with the warehouse data. This may be data that the legacy system does not capture or data that they want personally organized. Some examples are lists of prospective student IDs, charts of accounts, and specialized budget formulas. While some reporting tools have the ability to combine local data with the data warehouse, performance is often a problem. An area within the data warehouse that is dedicated to customer data may be desirable. Customers would have the ability to add, update, or delete data from this area. Query performance would improve, since the local data would now be part of the warehouse. Limits should be put in place to ensure that the presence of the local data does not affect overall warehouse performance. This "user" area should be monitored so it does not become a place for departmental data dumping or archiving.

Support all tools but standardize on one. The data warehouse should not discriminate against any SQL reporting tool. If a reporting or application development tool supports SQL, it should be allowed to access the data warehouse. Developers might want to use Macromedia's Cold Fusion or Microsoft's Active Server Page (ASP) to build an end-user Web interface to the data warehouse. Customers may want to use a familiar reporting tool. Cost might also determine which tool a customer decides to use. Regardless, the institution needs to standardize on one recommended reporting tool. Standardization is important if customers expect data warehouse staff to help troubleshoot query problems. Customers cannot expect a high level of service on a reporting tool that is not the institution's standard. Standardization also makes training easier, since it focuses on one reporting tool.

Make customers happy. Customers of the warehouse must find the data warehouse easy to use and accurate. If not, they will lose their confidence in the warehouse. All it takes is one bad experience. It is important to take time to understand customers' job responsibilities and computer experience. A little user "hand-holding" at the beginning pays big dividends in the long run. Training should be adjusted to meet users' needs. The data warehouse must provide customers the ability to assemble knowledge, not just access data. By making users happy and keeping their trust, an institution can continue to justify staffing, software, and hardware expenditures for the data warehouse.

Invest in end-user support structure. Once a data warehouse has been deployed, it's time to start answering questions. Users of the warehouse will have questions on data, problems installing reporting tool software, trouble remembering their passwords, and so on. Mechanisms to support them can include a Web site with data warehouse information or an E-mail address to which users can send questions or problem queries to be checked out.

A policy should be in place to get back to those individuals within 24 hours, or sooner if possible. A data warehouse users' group can be set up to share data findings, document frequently asked questions (FAQs), share queries, and educate members. Like training, end-user support is recognized but often not funded. It needs to happen.

Conclusion

A data warehouse can become an agent of change. Through it, an institution can now have easy access to quality data. This information can help it make better, more timely decisions. A warehouse can identify potential business opportunities as well. While a data warehouse is not a silver bullet or the next "killer" application, it can have an enormous effect on an organization. Building one takes a lot of hard work and dedication, but it is well worth the effort. Data warehousing is here to stay. In a private conversation, Bill Inmon told this author, "I never thought that data warehousing would become so fashionable!" Not only is it fashionable, it can be invaluable to an institution.

Glossary of Terms

Ad hoc query. Any query that cannot be determined prior to the moment the query is issued. An ad hoc query consists of dynamically constructed SQL, which is usually constructed by desktop query tools.

Business intelligence (BI). A term that is becoming popular to describe a broad category of technologies and software for helping organizations make better business decisions. The Gartner Group coined this term in 1996.

Customer. A person who interacts with the data warehouse. This term is preferred over user.

Data mart. A subset of information relevant to a group of users that may be transferred to a separate departmental work-group server to cut down on network traffic. The database involved can be relational, although a multidimensional OLAP server can be appropriate.

Data mining. A process that identifies significant relationships among data in a data warehouse. An often-used example of a data-mining result is the relationship between beer and diaper sales (proved to be urban legend).

Data modeling. A method used to define and analyze the data requirements needed to support the business functions of an enterprise. Data modeling defines the relationships between data elements and structures.

Data warehouse. An integrated repository of enterprise-generated, departmentally captured, and/or externally acquired data used to facilitate data access, reporting, and tactical/strategic decision making.

DBMS (database management system). A formal, usually computerized collection of data. Multiple users are able to manipulate, structure, access, and secure the data.

Denormalized. Not normalized. See Normalized.

Derived data. Data that are the result of a computational step applied to other data. Derived data can result from relating two or more elements to create a new element or from using an algorithm or rule to create a new element. For example, a length of time can be derived from the difference between two dates, and age can be derived from a birthday and the current date.

Dimensional data modeling. A data modeling technique that starts with a fact table surrounded by dimension tables. Because of the appearance of the structure, it is often called the "star schema." See also Tabular data modeling.

Domain. The set of values that are possible for a particular data element or column in a table.

ERP (enterprise resource planning). A set of enterprise software applications that automate finance, human resources, and other areas of an institution. Some examples of ERP vendors are PeopleSoft and SAP.

ETL (extract, transform, and load). A term that refers to three separate functions combined into a single programming tool for moving a population of data into a data warehouse.

FAQ. A frequently asked question (pronounced "F-A-Q"). Lists of FAQs are created to answer the majority of questions a newcomer to warehousing might have.

Granular data. Data with a high level of detail contained in the data warehouse. The more detailed the information, the finer the granularity.

Integration. The process of bringing together related data from different sources. This may involve transactions with the same data from different transaction processing systems.

Join. The process of combining data from two or more tables.

Left join. A technique for joining tables in which all of the records in the first (left-hand) table are included even if there is no match in the joined table.

Legacy system. A software system found on the mainframe system that is vital to the institution but hard to cope with. See also Operational system.

Lingua franca. A common language that is employed when people speak in different and mutually unintelligible tongues. SQL is such a computer language.

MDDB (Multidimensional database). A type of database that is optimized for the data warehouse. Multidimensional databases are frequently created from data in existing relational or warehouse databases. They support measures, dimensions, and cubes.

Metadata. Data about data. One example is the definition of a data element. Metadata is to the data warehouse what the card catalog is to the traditional library. It is an information directory, containing the yellow pages, a road map, and a list of places of interest for navigating a data warehouse.

Middleware. Software that mediates between an application program and a data warehouse. It manages the interaction between reporting tools and applications with the data warehouse.

Normalized. Of data, organized into tables in order to eliminate certain types of redundancy and incompleteness. See also Denormalized).

ODBC (open database connectivity). Middleware developed by Microsoft and based on the call-level interface specification of the SQL Access Group (a standards organization). ODBC allows users to access data in heterogeneous environments of relational and nonrelational database management systems.

ODS (operational data store). A type of database often used as an interim area for a data warehouse. The line between a data warehouse and an ODS is hard to draw.

OLAP (online analytical processing). A term that originated with Codd and Date's white paper that defined 12 OLAP product evaluation rules as the basis for selecting multidimensional products. OLAP is also defined as multidimensional tools for accessing, storing, and manipulating decision support information.

OLTP (online transaction processing). A term used to describe the transaction-processing environment of an operational system.

Operational system. A system of record; a transaction-update system. The operational system contains detailed data used to run the day-to-day operations of the business. The data continually change as updates are made, and reflect the current value of the last transaction. The operational system is the source of data for the data warehouse. See also Legacy system.

Power user. A sophisticated user of a data warehouse. A power user has considerable experience with computers and utilizes the most advanced features of the reporting tool.

Query. A SQL statement that requests data for decision support from a data warehouse.

Scalability. The ability to scale to support larger or smaller volumes of data and more or fewer users. The ability to increase or decrease size or capability is cost-effective with minimal impact on the services.

Server. A physical computer from which services are provided.

SQL. Structured query language (pronounced "sequel"). SQL is the lingua franca of relational database systems. It is used to build queries to be performed on a data warehouse. Some business intelligence tools can translate user queries into SQL code invisibly.

Subselect. A powerful feature of SQL that specifies a result table derived from the tables identified in the SQL from clause.

Summary data. Data that are the result of applying a process to combine data elements. These data may be taken collectively or in summary form.

System of record. A source of data for the data warehouse. The system of record, which may consist of legacy databases, extract files, and so on, is treated as the official record.

Tabular data modeling. A design method in which the tables look similar to those organized for an operational database except that they are denormalized. See also Dimensional data modeling.

Notes

1. Steve Steinberg, "Hype List," *Wired,* April 1996, 152.
2. William H. Inmon, "What Is a Data Warehouse?" *Tech Topic 1,* no. 1 (Sunnyvale, Calif.: PRISM Solutions, 1992), 1.
3. Ralph Kimball, *The Data Warehouse Toolkit* (New York: John Wiley & Sons, 1996), 310.
4. Sean Kelly, *Data Warehousing: The Route to Mass Customization* (England: John Wiley & Sons, 1996), 55.
5. Srikumar S. Rao, "Diaper-Beer Syndrome," *Forbes*, April 6, 1998, 128–130.
6. Ralph Kimball and Margy Ross, *The Data Warehouse Toolkit* (New York: John Wiley & Sons, 2002).
7. Christopher Adamson and Michael Venerable, *Data Warehouse Design Solutions* (New York: John Wiley & Sons, 1998).

8. Len Silverston, *The Data Model Resource Book* (New York: John Wiley & Sons, 2001).
9. Paul Winsberg, "Modeling the Data Warehouse and Data Mart," *InfoDB* 10, no. 3 (June 1996): 1–10 (http://www.dbaint.com/InfoDBDW.html).
10. Ralph Kimball and Richard Merz, *The Data Webhouse Toolkit* (New York: John Wiley & Sons, 2000), 217.
11. Larry English, *Improving Data Warehouse and Business Information Quality* (New York: John Wiley & Sons, 1999), 12.