

The SAS System in a Data Warehouse Environment

Randy Betancourt, SAS Institute Inc.

Abstract

The purpose of this paper is to provide the reader a general overview of the strategies employed in implementing a data warehouse and the role the SAS System® plays in these various steps. While this is not an in-depth methodology, it is an attempt to outline the various steps one would normally go through to implement a data warehouse. In order to make clear all of the terms and acronyms used in this paper, they will be underscored and defined in the glossary at the end of this paper.

Introduction

A data warehouse is a physical separation of an organization's on-line transaction processing (OLTP) systems from its decision support systems (DSS). It includes a repository of information that is built using data from the distributed, and often departmentally isolated, systems of enterprise-wide computing so that it can be modeled and analyzed by business managers in order to make them more competitive. Data warehousing is about turning data into information so that business users have more knowledge with which to make competitive decisions. Data in the warehouse are organized by subject rather than application, so the warehouse contains only the information necessary for decision support processing.

The data in the warehouse are collected over time and used for comparisons, trends and forecasting. These data are not updated in real-time, but are migrated from operational systems on a regular basis when data extraction and transfer will not adversely affect the performance of the operational systems.

Transformations are used in converting and summarizing operational data into a consistent, business oriented format. When the data is moved into the data warehouse, they should all be represented in the same fashion, for example, 'male' and 'female', regardless of their format in the operational system. This is also an opportunity to generate any derived information which is not contained in operational systems but can be useful in the decision support domain. The data warehouse may contain different summarization and transformation levels. In addition, the warehouse store is created to be read from, not written to or altered.

A critical component that crosses over most of these steps is the generation of both technical and business metadata which describes the data in the data warehouse (what, when,

how...). Technical metadata is used by a Data Warehouse Administrator to know when data was last refreshed, how it was transformed, and other details important for managing the data warehouse. Business metadata is data that is of more interest to end users of the data warehouse (data definitions, attribute and domain values, data recency, data coverage, business rules, data relationships, etc.). Metadata resides at all levels within the data warehouse. Metadata is the 'glue' which holds all the pieces together in warehouse environment.

A data warehousing strategy is designed to eliminate the traditional problems associated with allowing end-user access to operational data. Some of these problems are listed in Table 1 below.

Table 1
Possible Problems Encountered when allowing End-User Access to OLTP Data

- A given query may impact performance of the OLTP system
- The constantly changing state of an OLTP makes replication of an answer set difficult
- End-users must understand physical file attributes of the OLTP source
- End-users must write database-specific access logic to read many OLTP data sources
- To form a answer set, large numbers of tables may need to be joined together, adversely impacting performance of the OLTP system
- Data in the OLTP environment is rarely quality assured for DSS analysis
- OLTP systems may not store data over 90 days, making temporal comparisons difficult

While this is by no means an exhaustive list, any one of these issues should be sufficient for an organization consider a data warehousing strategy. The rest of this paper will explore the various steps for implementing a data warehouse, and the role the SAS System in this endeavor. The steps, outlined in Table 2, form the outline for this presentation.

Table 2
Steps for Implementing a Data Warehouse

- Subject Definition
- Data Acquisition
- Data Transformation
- Metadata Management
- Production Loading the Warehouse
- Exploitation

Subject Definition

Subject definition is the activity of determining which subjects will be created and populated in the data warehouse. This is always the starting point for implementing a data warehouse, and in fact, many data warehouse projects not succeeding can trace their failure to not clearly defining the subjects. A subject is a logical concept, for example, customers. Subject in a data warehouse for sales and marketing might consist of entities such as prospects, customers, competitors, etc. Subjects do not necessarily have a one-to-one correspondence to operational data sources. The steps in defining a subject are 1) conduct user and management interviews 2) build the logical data model and 3) from the logical data model, build the physical data model.

In an OLTP environment, data is organized around a particular business process, such as claims processing. The design principal behind OLTP environments, is to drive all data redundancy out of the database to ensure data integrity and ensuring that changes to data at an atomic level. In an OLTP environment, for example, information related to customers may be kept in a number of different tables. An even more challenging problem is many of the data elements for customers may even be stored in different OLTP systems. By starting with a logical concept of business subject, the data warehouse designer can begin to build logical model. Once built, the logical data model determines the physical model, and transformation models that define the warehouse environment. The purpose of these models is to determine the structure and content of the data warehouse and to define how operational data must be transformed to populate it.

As part of defining the business subjects the data warehouse designer will need to conduct interviews with a number of individuals in the organization with the goal of understanding the business unit objectives, understanding the data currently in use for decision support, and what data is lacking to support current and future decision making activities. These individuals will include business unit analysts, business unit managers, end-users and analysts from related business units.

Once the interview process is complete, the next step is to develop a data model. Building a data model is the process of translating

business processes and concepts into physical data structures. A good analogy is that of a blueprint to build a home.

Next, the logical model must be translated into a physical data model which defines the actual data storage architecture of the data warehouse. The physical design should take into account how the data is expected to be used, so as to organize data for the most frequent kinds of use; some degree of foresight is required here, given the increased value to be gained out of the data warehouse from ad-hoc, investigative query and reporting of the data. The physical data model should also give consideration to how any data-marts will be defined.

Physical models can draw on several design constructs, such as entity relationship model, star schemas or snowflake schemas, persistent multidimensional stores, or summary tables. It is possible that a single data warehouse implementation may combine one or more of these schemas.

- ***Entity Relationship Model.*** Based on set theory and SQL, the entity relationship model is the choice for modern OLTP DBMS systems. This model seeks to drive all of the redundancy from the database by dividing the data into many discrete entities across a large number of small tables. When a transaction needs to change data (through either adds, deletes, or updates), then the database need only be 'touched' in one place. Being optimized for online update and fast transaction turnaround, this model is not well suited for querying in a data warehouse environment. See Figure 3 in Appendix 1.
- ***Star Schema.*** Uses an asymmetrical relationship model employing a single, large fact table of highly additive numeric values along with smaller tables holding descriptive data, or dimensions. The fact table contains hundreds of millions of rows of continuous data values that can be added and thus quickly compressed into a small result set. Each dimension table holds a primary key, and a composite, foreign key is held in the fact table. Users typically spend 80% of their time browsing the dimension tables building query constraints, and then spend the other 20% of their time taking the selected constraints and constructing a query that joins a fact and dimension table together (through the primary/foreign key relationship). End-users should not construct the actual SQL query, but have an application interface that constructs the query logic on their behalf. See Figure 4 in Appendix 1.
- ***Snowflake schema.*** Uses a model similar to the star schema, with the addition of normalized dimension tables that create a tree structure. The normalization of the dimension table reduces storage overhead, by eliminating redundant values in the dimension table by keying on an outtrigger table. See Figure 5 in Appendix 1.

- Persistent Multi-Dimensional Stores.** New for an upcoming release of the SAS System, MDDBS uses the approach of creating and storing permanent N-Way crossings. This represents a "fact table" of the full list of crossings specified in the creation phase of the MDDB. Levels with valid values are stored, thus addressing the "sparsity" problem in the first phase. This step has shown significant reduction in size of data as compared to the target base table. Some of this reduction is due to subsetting the number of columns retained. Once this "fact table" is created, application programmers have two options. In one case, MDDB tables consolidated into defined hierarchies are created and stored. These hierarchical consolidations can be stored in the same location as the central "fact table", and are accessible to requesting applications. The performance implications for creating these specified consolidations ahead of time is improvement in access time when requested by the client application. On the down side, sparsity is reintroduced, because consolidating within the definitions of a hierarchy raises the possibility requesting summaries or crossings with no data. See Figure 6 in Appendix 1.
- Summaries Tables** Summarization consists of taking detail level data and "rolling-up" the data into a more compact form. Typically, summarizations tend to follow natural hierarchies. For example, we may summarize product sold on a daily, weekly, monthly, and annual basis. By permanently storing these summaries, end-user can use tools that allow the drilling-down or drilling-up on this summary information. Starting at the lowest level of summarization (in our example, daily), and going up the hierarchy, the table storage requirements get smaller, but at the same time some of the detail data values are 'lost'. Data can usually be pre-summarization prior to being loaded into the warehouse, in which case data volumes will be reduced, or may be summarized on an ad-hoc basis from within the warehouse. In this case, careful monitoring of data usage by the warehouse administrator should help identify where pre-summarization can be used to prevent excessive overhead by end-users constantly summarizing the same lower level detail data.

Finally, the transformation model must define how to translate the operational data into the target store for the data warehouse. This model is developed after the interview process and after investigation into the operational data sources. Investigation of the operational data sources determines whether a data source exists, its location and format, its level of granularity, its access method, and any other physical properties that help describe how to map the operational data sources to data warehouse target store. These transformations will consolidate and enrich the warehouse data. This is also the opportunity to create any derived information that is not stored explicitly in the operational data stores.

Data Acquisition

Data acquisition refers to the program logic that attaches to the operational data stores. From the SAS System's point of view, this refers to the family of SAS/Access® Software. SAS/Access software is an expression of SAS Institute's Multiple Engine Architecture (MEA) which uses a layered I/O model to abstract from SAS application logic the physical properties and I/O specific logic to a data source for read, write or update functions. This abstracted I/O model obviates the need to master a variety of data access languages. One need only understand SAS Application programming logic. In the current release of the SAS System, all data, regardless of its type or format, are accessed through a set of engines or access methods. These access methods provide the framework for translating SAS syntax for read, write and update services into the appropriate relational database management system (RDBMS) or file structure calls. Presently, the SAS System provides more than 50 different access methods for a variety of file types found in different hardware environments. The different types of access methods supported by the SAS System are listed in Table 3 below.

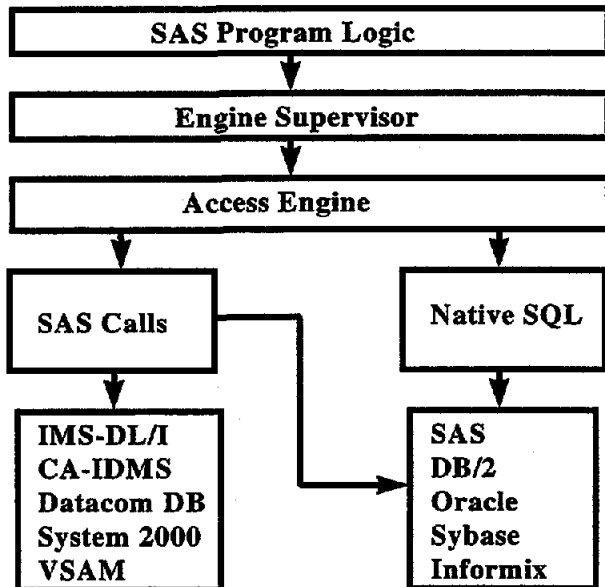
Table 3
Types of SAS/Access Methods

- SAS Tables
- Relational Database Management Systems
- Hierarchical Database Management System
- Network Database Management Systems
- Data Gateways and Standard API's such as ODBC
- External File Formats such as VSAM
- Sequential for Tape and Other Sequential Access Devices and Media

With the Multiple Engine Architecture, a single access environment is provided. In addition, the SAS System supports Structured Query Language (SQL). With SAS SQL support and the support for a variety of access methods, SQL in the SAS environment can be used as the data access language for relational as well as non-relational file structures. A pictorial representation of the SAS System's Multiple Engine Architecture is presented below.

In addition to translating SAS data management syntax to the data access language for the target data store, the SAS System provides a method for passing RDBMS-specific logic to the target RDBMS. This is particularly useful in those instances where the SAS internal SQL processor can not optimize queries for the target RDBMS or one wishes to support SQL extensions provided by the RDBMS such as stored procedures or triggers.

Figure 1
Multiple Engine Architecture



Data Transformation

Since data coming from the OLTP environment is typically in an inconsistent form for decision support, a process of data transformation is required. Transformation of data consists for two distinct steps. The first of these steps is integration and conversion. The second step is summarization.

Integration and conversion is aimed at resolving data inconsistencies in value definitions, formats among data, as well as this being an opportunity to create new columns for analytic purposes. An example of integration is combining different attributes from different sources to create a consistent entity. For example, customer name may be obtained from the customer's OLTP database, but in order to be able to conduct analysis about customers along a geographic dimension, we need to also include state and zip code from the shipping OLTP database. An example of conversion is to convert the values used to represent gender among different transaction databases. One OLTP database may use 'M' to represent males and 'F' to represent females. A second OLTP database may code males as '1' and females as '2'. Before passing data from the operational environment into the warehouse, these data values must be made consistent.

While the previous example is a rather simple example of a recoding technique, other conversions may be more complex, such as converting time units into consistent time units which all begin

at the start of a given epoch. This is needed for time dimension analysis. A significant feature for the SAS System is its ability to easily handle date-time arithmetic. The date-time values in the SAS system are stored internal as double-precision floating point, using an off-set from the date of January 1, 1960. The SAS System also provides a large number of additional tools to aid in data transformation. Some of the tools are listed in Table 4, Transformation Capabilities.

Table 4
Data Transformation Features in the SAS System

• Open 255 Input Files in a Single Step
• Open 255 Output Files in a Single Step
• Look-ahead Read Logic
• Table Look-up Logic
• Sorts Techniques with Different Character Sets and Collating Sequences for both ASCII and EBCDIC
• SQL for Grouping, Sorting, and Aggregation Functions
• SQL programming with Arithmetic, Random Number, Probability, and String Manipulation Functions
• Summary Procedures for Grouping by Classification Values
• Means Procedure for Collapsing Numerical Data Using different Univariate Statistical Methods
• Frequency analysis for one-way, two-way, and n-way classifications
• Multivariate Statistical Methods for Numerical Analysis

Summarization is another aspect of transformation. Summarization in the data warehouse environment is critical from the perspective of providing the analysts a historical view, rather than a record by record view provided by the OLTP database. Summarization can also help reduce the volume of data the analysts most process, when compared to the volume of data found in the OLTP environment. Summaries consists of both numerical summarizations as well as groupings, or counts. Take for example, the detail records from the sales subject in a data warehouse presented in Figure 2.

Figure 2
Detail Record for Sales Subject

Agent	Date	Customer	Product	Amount
Rush	13Mar96	Sears & Robuck	Data Warehouse	\$34,000
Smith	12Mar96	Macy's	Consulting	\$12,000

In our example, the column labeled "Amount", because of its additive quality, is a candidate column for summary statistics such as mean, sum, count, mode, etc. An appropriate analysis might include total sales, total sales within product, or total sales within customer, etc. The columns labeled "Agent", "Product", and "Date" are candidate columns for counts. The analysis possible with these counts might include a count of products sold by an agent or count of products sold by agent within product etc. A desirable strategy to pre-compute as many summaries as possible to obviate the need for the end-user access tool to compute summaries and counts on-the-fly. However, attempting to summarize and group every combination will quickly reach the point of diminishing returns, as disk space consumption increases. This is where a carefully modeled warehouse done with a thorough end-user requirements gathering phase pays dividends.

Metadata Management

In order to provide access to the data warehouse, it is absolutely necessary to maintain some form of data which describes the data warehouse. This data about the data is called meta data. Meta data has been around as long as there have been programs and data the programs act on. In most cases, meta data is scattered throughout the enterprise, and as a result, one of the major challenges facing the data warehouse implementers is the collection and consolidation of this information. Record descriptions in a COBOL program are meta data. So are DIMENSION statements in a FORTRAN program, or SQL Create statements. The information in an E-R diagram are also meta data, or even the knowledge a user has in his or her head about a given business process. Another way to view meta data, is as the warehouse repository that defines the rules and content of the warehouse and maps this data to the query user on one end and the operational sources of data on the other.

In the past, most Information Technology (I.T.) professionals have tended to pay scant attention to meta

data. This is because the I.T. community is intimately familiar with operational systems and can therefore navigate their way through these various systems. In the data warehouse environment, the business users and other end-users are introduced to technology which they are

Table 5
Business Meta Data

- Defined Subjects
- Hierarchies
- Drill Columns
- Analysis Columns
- Actual Values Column in Forecast or Budget
- Budget Values Columns in Forecast or Budget
- Time Dimensions
- Critical Success Values Columns
- Categorical Columns
- Classification Columns
- Dependent Variable Columns
- Independent Variable Columns
- Analysis Type
- Data Type for Target Column
- Display Attribute
- Value Constraints
- Date Time Value of last Refresh
- Summarization Values

generally not familiar with.

The advantages of having meta data accessible to the end-user are almost self-evident. Having meta data as

an abstraction layer which masks these technologies to make information resources access-friendly is essential. Ideally, end-users should be able to access data from the data warehouse without having to know where that data resides, its form or any other physical attributes. The term business meta data describes the abstraction of the warehouse data properties and attributes for end-users or business users.

From a process management point of view, another type of meta data required for the data warehouse is technical meta data. Because of the complexity of these data flows from operational systems into the data warehouse, technical meta data is needed to manage and track the various processes. It is often the case that meta data may need to be exploited by other programs. In such cases, it is appropriate to allow a query language, like Structured Query Language (SQL) to query the meta data, as well as offer appropriate Application Programming Interfaces (API's) which allow communication through object methods. In order to keep these distinctions clear, the term technical meta data will be used to describe the meta data for managing the process flow of data to and from the data warehouse. Typical types of technical meta data are listed in Table 6 below.

Table 6
Technical Metadata

- **Source Data**
- **Target Warehouse Data**
- **Aggregation Methods and Rules**
- **Roll-up Categories and Rules**
- **Availability of Summarizations**
- **Security Controls**
- **Mappings of Legacy Data to the Warehouse**
- **Purge and Retention Periods**
- **Frequency of Loadings**
- **Exception Rules**
- **Reference and Look-Up Tables**
- **Entity Ownership**
- **Access Patterns and Attributes**

Technical data defines the attributes that describe the physical characteristics of an item (where it came from, how it was transformed, who is responsible for it, when it was last loaded, etc.) While it may be the case that some of the technical meta data may be of interest to the business user, it is used mainly by the I.T. organization for the purpose of managing all of the processes that are required to flow data from the operational environment into the data warehouse environment.

Production Loading the Warehouse

In contrast to the OLTP environment, a data warehouse does not change its state from moment to moment, but is loaded or refreshed by bringing static snapshots from the OLTP environment on a regularly scheduled basis. This periodic loading of static snapshots from the OLTP environment give the warehouse its time-variant quality. In essence, the data warehouse is a time series.

In most cases, the data warehouse designer must consider 3 different types of loading strategies. They are 1) the loading of data already archived, 2) the loading of data contained in existing applications, and 3) incremental changes from the OLTP environment from the last time the data was loaded into the data warehouse.

The simplest loading technique is the loading of data already archived. Archival data is typically is usually stored on some form of sequential bulk storage, such as magnetic tape. As indicated previously, the SAS System offers a variety of sequential access methods for tape and other sequential media.

The loading of data contained in existing applications is similar to the loading of archived data. Existing files and tables are scanned and data is transformed according the established transformation model. In most cases, this process traverses a number of different technologies and file systems. For example, we may scan a segment with IMS running under MVS, transform the data, and finally, transport and load the data into a relational format on a UNIX file system. The resources consumed by this type of load are considerable. However, this should be a one-time load.

A strategy for minimizing the impact on the OLTP environment is to load the data elements into the SAS System and perform the transformation inside the SAS environment. In addition to minimizing the impact on the OLTP environment, one has to

understand a single framework as opposed to having to deal with various data access and data manipulation languages used by the various OLTP data stores.

The third type of load into the data warehouse is that of loading changes into the warehouse that have been made since the last time the data warehouse was refreshed. This is sometimes referred as change data capture. A number of strategies for change data capture exist. They are listed in Table 7 below.

Table 7
Change Data Capture Strategies

- **Replacement of the entire table from the OLTP Source**
- **Scanning for date-time stamps in the OLTP Source**
- **Reading operational audit files**
- **Trapping changes at the RDBMS level**
- **Reading RDBMS log tapes**
- **Comparison of OLTP 'before' and 'after' images to one another**

Exploitation

Getting information structured and organized to meet business needs is vitally important but it is a means to an end, not an end in itself. Your data warehouse is incomplete until it provides the exploitation tools that enable end users to view, analyze and report on data in ways that support better decision making.

Depending on the end users' requirements, data warehouse exploitation tools may be anything from ready-to-use and simple query and reporting tools, through multidimensional analysis tools, to advanced EIS applications designed to meet company-specific objectives.

The SAS System provides tools for ad hoc query and reporting and batch reporting of information in the SAS Data Warehouse, and where necessary, through to the underlying data in operational systems. The menu-driven

interface can be tailored to fit the user's individual wishes and requirements. Tools include a native SQL query dialogue as well as a reporting tools that allow ad hoc data selection (filtering) and execution. Reporting tools, including tabulation and printing functionality, may be fully implemented within the batch environment.

OLAP enables the full realization of enterprise-wide data's business potential by delivering the freedom to access, transform and explore data from any source, in any operating environment. For an OLAP tool to succeed, it must first provide power and flexibility in data access and transformation. This is what the SAS Data Warehouse delivers; once users have data in the right form, unlimited multidimensional analysis techniques and sophisticated reporting allow data exploration from infinite perspectives.

OLAP++ is SAS Institute's extension of the OLAP concept, and is specifically designed to address the needs of SAS software users building applications that require multidimensional views of large quantities of data from multiple sources. OLAP++ consists of a library of object classes that fall into two categories: a display class, which extends the flexibility of screen design, and a multidimensional engine class, for the registration of information about the multidimensional data.

EIS solutions ensure that decision makers have instant access to relevant and up-to-date information. The SAS Data Warehouse combines interactive, user-friendly interfaces with comprehensive functionality to place users in the driving seat. Multidimensional viewing enables data to be viewed from an unlimited number of perspectives: drill-down, hot-spotting and traffic lighting support the identification of business trends and long-term developments; critical success factors and key performance indicators help decision makers to focus on key issues.

About the Author

Randy Betancourt is Program Manager for Data Warehousing at SAS Institute Inc. He can be reached through e-mail at sastrb@unx.sas.com.

Sources Consulted

- Anderson, Scott and Morton, Steve. *Data Modeling for SAS Data Warehouse*. A SAS Institute White Paper.
- Eckerson, Wayne. *Data Warehouses: Product Requirements, Architectures, and Implementation Strategies*. **Open Information Systems** Volume 9, Number 8. Pages 3-27.
- Emmerich, Thomas. *The Rapid Warehousing Methodology*. A SAS Institute White Paper
- Inmon, William. *Loading Data into the Warehouse*. **Tech Topic** Volume 1, Number 11.
- Kimball, Ralph. *The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouses*. **John Wiley and Sons** 1996. ISBN 0-471-15337-0
- Poe, Vidette. *Building a Data Warehouse for Decision Support*. **Prentice Hall** 1996. ISBN 0-13-371121-8
- Raden, Neil. *Modeling A Data Warehouse*. **Information Week**. Pages 60-66. January 29, 1996 from <http://techweb.cmp.com/iw>
- Sacdeva, Satya. *Metadata: Guiding Users Through Disparate Data Layers*. A White Paper
- Strange, Kevin and Dessner, Howard. *The Four Styles of OLAP*. **Gartner Research Note from Strategic Data Management**. January 30, 1996.
- Tanler, Richard. *Data Warehouses & Data Marts: Choose Your Weapon*. **Data Management Review** Volume 6, Number 2, February 1996.
- Tasker, Dan *The Problem Space: Practical Techniques for Gathering & Specifying Requirements Using OBJECTS, EVENTS, RULES, Participants and Locations*. A self-published book. ISBN: 0646-12524-9. Dant@research.otc.com.au
- Tidename, Sue and Chu, Robert. *Building Efficient Data Warehouses: Understanding the Issues of Data Summarization and Partitioning*. A SUGI 21 Paper.
- Von Halle, Barbara. *Objects and Business Rules: Are They on a Collision Course?* **Database Programming and Design**, Volume 9, Number 3, March 1996.

Appendix 1 Database Schemas

Figure 3
Entity-Relationship Model

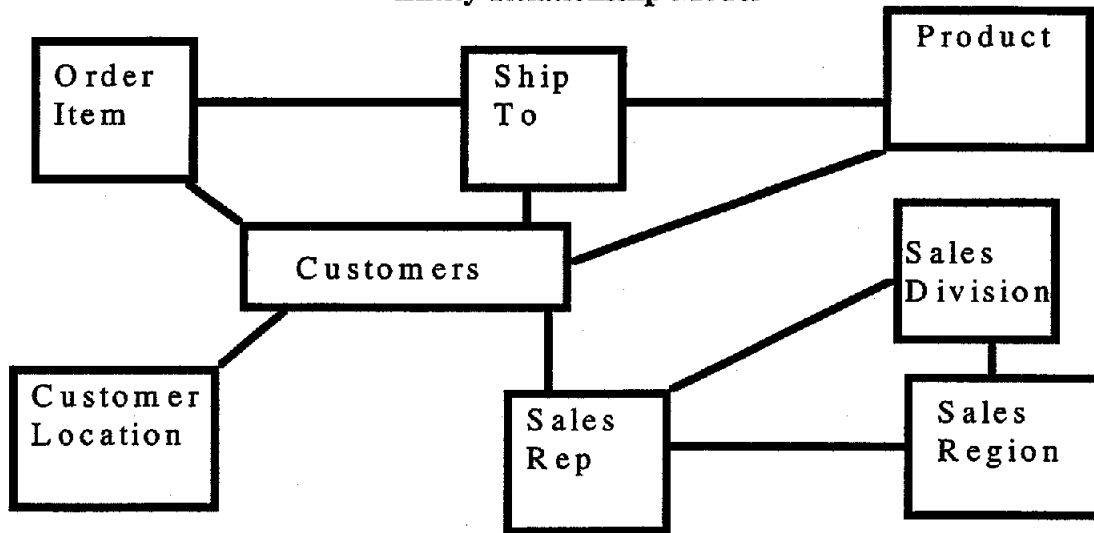


Figure 4
Star Schema Model

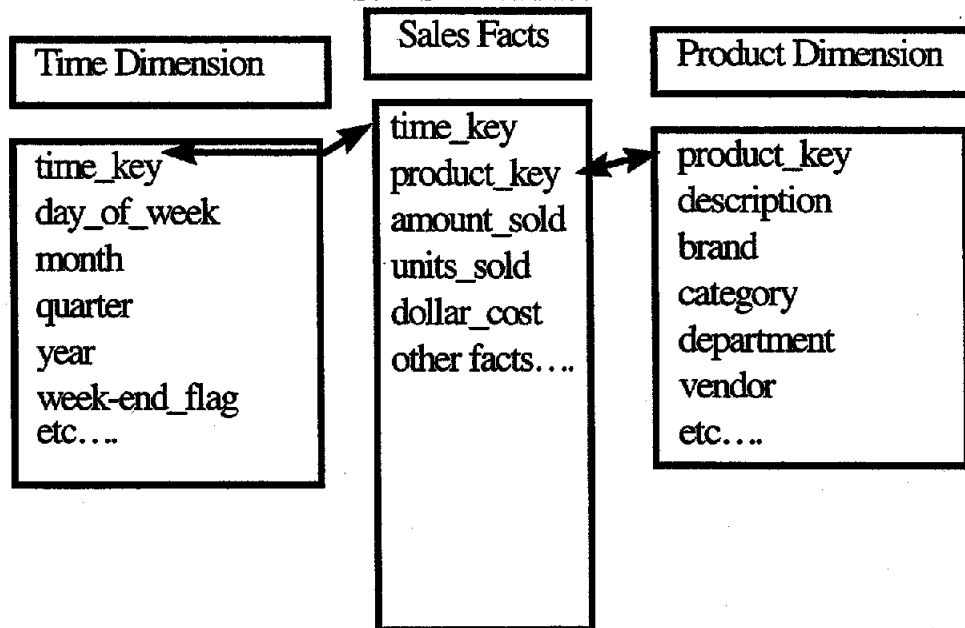


Figure 5
Snow-Flake Schema

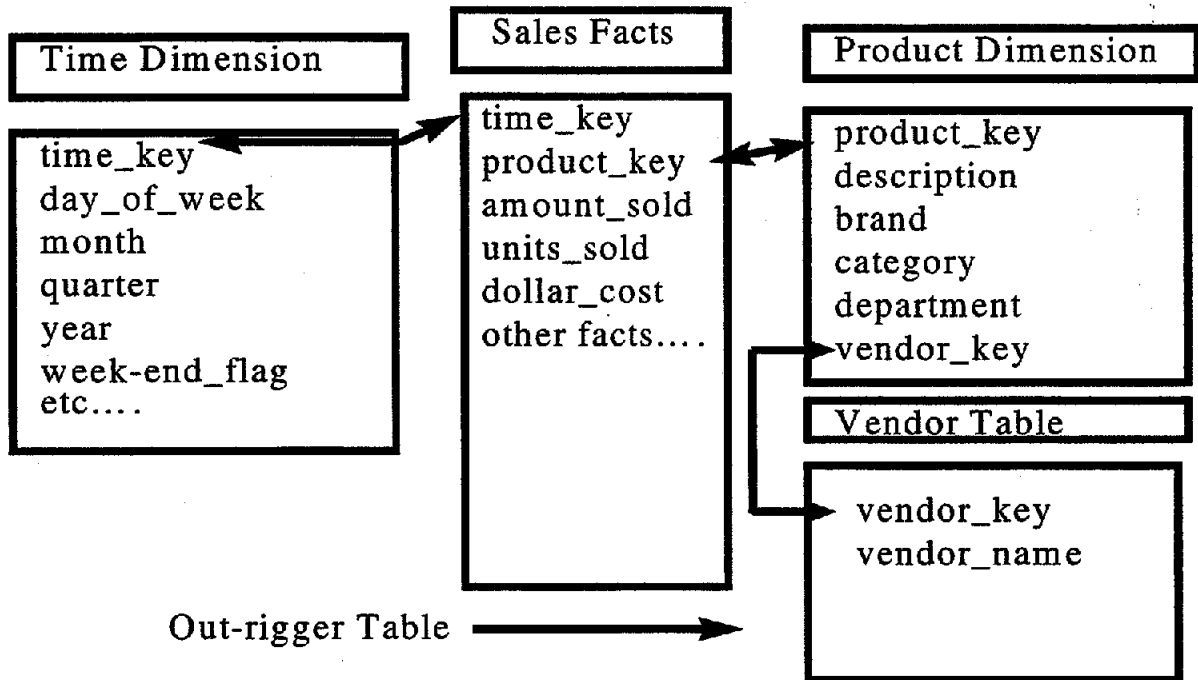
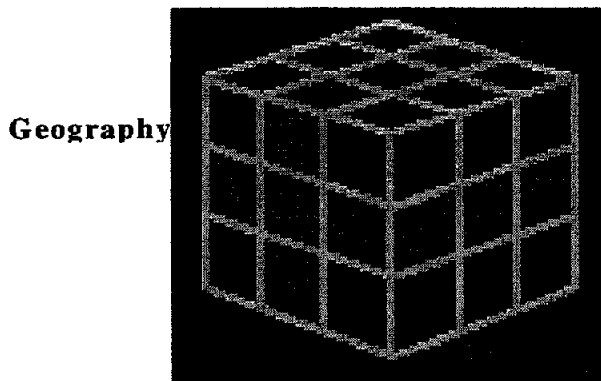


Figure 6

Persistent Multi-Dimensional Store



Glossary

Access Methods a set of routines that are particular to the open, read, write, and close protocol for a given data format.

Application Programming Interface (API) A well-defined and published set of calling routines which allow an application program to access a set of services. Thus the application program thus does not have to write the particular service, but can obtain them from the program that offers the interface.

Atomic Level the lowest level of value for a given datum such that there is no redundancy.

Business Meta Data data or descriptions of data in the warehouse. It describes the abstraction of the warehouse data properties and attributes for use by end-users or business users.

Business Unit Objectives the business goals and success factors, or metrics for measuring those goals for a department inside an enterprise.

Conversion the process of taking creating a single, consistent unit of measure for a given data element.

Data Extraction the process of copying from an OLTP environment to the data warehouse environment.

Data Integrity a result of applying constraints and rules to data inside a database to insure the accuracy of values.

Data Mart a sub-set or 'slice' of data from the data warehouse that is either highly summarized in a relational form or in a multi-dimensional cube form. Its organization is highly dependent on the query and reporting access patterns of the end-users.

Data Model a logical representation of a business process and concepts which is translated into physical data structures.

Data Transfer the process of moving data from one environment to another or from one file system to another file system, usually over a network.

Data Warehouse Administrator the individual(s) responsible for the day-to-day functioning and on-going maintenance of the data warehouse.

Data Warehousing a copy of from the OLTP environment that is refined and enhanced for query and reporting.

Decision Support Systems a process of using data to make both tactical and strategic decisions within an organization.

E-R Diagram A pictorial description of the entity relationship model for associating tables together in an RDBMS. For a simple example, see Figure 3 in Appendix 1.

Glossary a brief explanation of terms used in this paper.

Integration the process of bringing together data elements from different OLTP databases into a single representation in the data warehouse.

Logical Model an abstraction, usually in some symbolic form, of a given business process which identify the relationship of data elements. This activity precedes the physical modeling activity.

Metadata is data or information about the entities in the data warehouse used to support operations and use of the data warehouse.

Multiple Engine Architecture a model for layering I/O components within the SAS System which abstracts from the SAS application logic, specific instructions for the data format being read, written, or updated.

On-Line Transaction Processing a process of entering data reliable into a database that is modeled after a particular business function or process.

Operation Data Source the OLTP database environment where data for the warehouse originates.

Outrigger Table a secondary dimension table attached to the primary dimension table in a star schema. This normalization of the primary dimension table reduces redundancy.

Physical Model the design of a database, based on a logical model, that identifies actual tables and index structures.

Relational Database Management System RDBMS a software system that use set theory and relational algebra to dynamically determine how data in tables can be associated with one another, without having to describe these associations ahead of time. Structured Query Language (SQL) is the data access language used.

Snow-flake Schema a variation of the start schema design, where the dimension table is normalized, using an outrigger table. this creates additional dimension tables in a treed fashion.

Star Schema an arrangement of database tables in which a large fact table with a composite, foreign key is joined to a number of dimension tables. Each dimension table holds a single primary key.

Stored Procedure a piece of program logic inside an RDBMS environment that can be invoked to perform an action or piece of work.

Structured Query Language a data access language for accessing relational database systems (RDBMS).

Subject a logical entity in the warehouse that models a particular business subject area. Examples are customers or competitors.

Technical Meta Data is the data which describes data flows from operational systems into the data warehouse. Technical meta data is used by the warehouse administrator to manage and track the various processes that define the warehouse.

Triggers the invocation of piece of work, or action that is event-driven inside a relational database management system

Summarizations the process of collapsing data into a more compact form by either computing summary statistics, such as mean, sum, mode, etc. on numeric data, or by creating counts on non-continuous columns.

Target Store the physical database format for the data warehouse. Different vendors offer different database formats. The SAS System offers one such format.

Transformation the process of changing, filtering, or altering the value of a data element. These changes can apply to any number of different data types.

Transformation Model the description of data elements from the OLTP databases and how these values will be altered for use in the data warehouse.