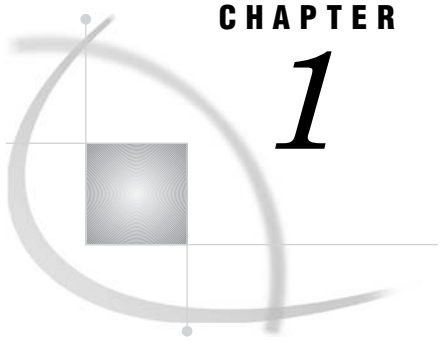

CHAPTER

1

OLAP Introduction and Overview

What Is OLAP?

Online Analytical Processing (OLAP) is a technology that is used to create decision support software. OLAP enables application users to quickly analyze information that has been summarized into multidimensional views and hierarchies. By summarizing predicted queries into multidimensional views prior to run time, OLAP tools provide the benefit of increased performance over traditional database access tools. Most of the resource-intensive calculation that is required to summarize the data is done before a query is submitted.

Data Storage and Access

Decision makers are asked to make timely and accurate decisions that are based on the past performance and behavior of an organization as well as on future trends and directives. To make effective business decisions, business analysts must have access to the data that their company generates and responds to. This access must include timely queries, summaries, and reviews of numerous levels and combinations of large, recurrent amounts of data. The information that business analysts review determines the quality of their decisions.

Organizations usually have databases and data stores that maintain repeated and frequent business transaction data. This provides simple yet detailed storage and retrieval of specific data events. However, these data storage systems are not well suited for analytical summaries and queries that are typically generated by decision makers. For decision makers to reveal hidden trends, inconsistencies, and risks in a business, they must be able to maintain a certain degree of momentum when querying the data. An answer to one question usually leads to additional questions and review of the data. Simple data stores do not successfully support this type of querying.

A second type of storage, the data warehouse, is better suited for this. Data is maintained and organized so that complicated queries and summaries can be run. OLAP further organizes and summarizes specific categories and subsets of data from the data warehouse. This results in a robust and detailed level of data storage with efficient and fast query returns. SAS OLAP cubes can be built from either partially or completely denormalized data warehouse tables. Stored, precalculated summarizations called *aggregations*, can be added to the cube to improve cube access performance. Aggregations can either be pre-built relational tables, or you can let the cube create its own optimized aggregates.

Benefits of OLAP

The ability to have coherent and relevant information is the reason OLAP has gained in popularity. OLAP systems help reveal evasive inconsistencies and trends in data that might not have been seen before. OLAP users can intuitively search data that has been consolidated and summarized within the OLAP structure. In addition, OLAP tools allow for tasks such as sales forecasting, asset analysis, resource planning, budgeting, and risk assessment. OLAP systems also provide the following benefits:

- fast access, calculations, and summaries of an organization's data
- support for multiple user access and multiple queries
- the ability to handle multiple hierarchies and levels of data
- the ability to presummarize and consolidate data for faster query and reporting functions
- the ability to expand the number of dimensions and levels of data as a business grows.

To fully understand the benefits of OLAP and the details of its effective implementation, it helps to examine the technology from two perspectives—first, from that of the users and second, from that of the information technology (IT) administrators who are responsible for OLAP implementation. The users, typically business analysts and executives, expect the data to be organized according to categories that reflect the way in which they think about the enterprise. For IT administrators, OLAP can present a long list of technical issues, including these concerns:

- storage requirements and associated costs
- client and server capabilities

- maintenance activities such as update and backup
- performance considerations such as the amount of time that is required to build a multidimensional model
- the ability of the OLAP solution to integrate with current or planned data warehouse strategies and architectures.

OLAP Variations

OLAP technology can be further defined by the methods for storing and accessing data and by the performance of queries against that data. SAS OLAP supports three different variations of OLAP technology:

- MOLAP
- ROLAP
- HOLAP

MOLAP — Multidimensional OLAP

MOLAP (multidimensional online analytical processing) is a type of OLAP that stores summaries of detail data (aggregates) in multidimensional database structures. MOLAP cubes are most suited for slicing and dicing of data and are used when performance and query speed is critical. A unique feature of MOLAP is that calculations are predetermined and created with the cube. Whereas MOLAP is well suited for complex queries and calculations, it is limited in the amount of data it can handle.

ROLAP — Relational OLAP

ROLAP (relational online analytical processing) is a type of OLAP in which multidimensional data is stored in a relational database such as a SAS table or an ORACLE table. ROLAP is more scalable than other OLAP types and handles extensive amounts of data well. Although performance can be somewhat slow, ROLAP is limited only by the size of the relational database it is identified with.

ROLAP data is stored in either a flat file or with a star schema. With ROLAP, each instance of slicing and dicing of data is part of an SQL query (or multiple SQL queries) and is comparable to a WHERE clause in the SQL statement.

HOLAP — Hybrid OLAP

HOLAP (hybrid online analytical processing) is a type of OLAP in which relational OLAP (ROLAP) and multidimensional OLAP (MOLAP) are combined. In HOLAP, the source data is usually stored using a ROLAP strategy, and aggregations are stored using a MOLAP strategy. It combines the best features of both ROLAP and MOLAP. This combination usually results in the smallest amount of storage space. In HOLAP, aggregates can be precalculated and can be linked into a hybrid storage model.

What Is a Cube?

One of the advantages of OLAP is how data and its relationships are stored and accessed. OLAP systems house data in structures that are readily available for detailed queries and analytics. Cubes are central to the OLAP storage process.

A *cube* is a set of data that is organized and structured in a hierarchical, multidimensional arrangement. The cube is usually derived from a subset of a data

warehouse. Unlike relational databases that use two-dimensional data structures (often in the form of columns and rows in a spreadsheet), OLAP cubes are logical, multidimensional models that can have numerous dimensions and levels of data. Also, an organization typically has different cubes for different types of data.

One of the challenges of OLAP cube data storage and retrieval is the growth of data and how that growth affects the number of dimensions and levels in a cube hierarchy. As the number of dimensions increases over time, so does the number of data cells on an exponential scale. To maintain the efficiency and speed of the OLAP queries, the cube data is often presummarized into various consolidations and subtotals (aggregations).

Note: The SAS OLAP Server term *cube* is synonymous with the terms *hyper-cube* and *multi-cube*. △

Understanding the Cube Structure

OLAP cubes organize data in a hierarchical arrangement. Data is structured according to dimensions and measures.

Dimensions group the data along natural categories. (Examples of dimensions are Time, Products, Organization). Typically, dimensions offer different levels of grouping (for example, the Time dimension can be grouped by Years, Months, Days, etc.). *Levels* are organized into one or more hierarchies, typically from a coarse-grained level (for example, Year) down to the most detailed one (for example, Day). The individual category values (for example, 2002 or 21Jan2002) are called *members*.

Measures are the data values that are summarized and analyzed. Examples of measures are sales figures or operational costs. The data for measures is located in *cells*. Cells are the intersection of one member for every dimension.

Presummarized data in a cube is stored in aggregations. Aggregations are the basis for fast response to data queries in OLAP applications. An aggregation is possible at each intersection of a level of one or more dimensions. The selection of aggregations to presummarize is one of the major factors that determine query response time and cube size.

SAS Servers

SAS Servers and SAS OLAP

SAS OLAP uses a combination of SAS servers to store cube metadata, to store the physical cube structure, and to query cubes after they are created. Several types of SAS servers are available to handle different workload types and processing intensities. The term *server* refers to a program or programs that wait for and fulfill requests from client programs for data or services.

The SAS servers use the SAS Integrated Object Model (IOM), which is a set of distributed object interfaces that make SAS software features available to client applications when SAS is executed on a server. Each server uses a different set of IOM interfaces and has a different purpose. The following servers are used by SAS OLAP:

- the SAS Metadata Server
- the SAS Workspace Server
- the SAS OLAP Server

In addition to the other servers that SAS OLAP uses to store and process cube data, the SAS Stored Process Server is used in several different client applications that access SAS OLAP cubes and report on OLAP cube data.

SAS Metadata Server

The SAS Metadata Server controls access to a central repository of metadata that is shared by all of the applications in the system. It enables all users to access consistent and accurate data. The SAS Metadata Server stores the metadata that defines the cubes. It is a multi-user server that enables users to manage metadata in one or more metadata repositories by using the SAS Open Metadata Interface.

Note: The SAS Open Metadata Interface is an object-oriented application programming interface (API) that interacts with the SAS Metadata Server. SAS OLAP Cube Studio is an example of an application that is compliant with the SAS Open Metadata Interface. △

The SAS Metadata Server uses the Integrated Object Model (IOM) that is provided by SAS Integration Technologies. IOM provides distributed object interfaces to Base SAS software features. It enables you to use industry-standard languages, programming tools, and communication protocols to develop client programs that access these services on IOM servers.

In the SAS OLAP Server, all relevant structural information is contained within the cube and most of it is also replicated within the SAS Open Metadata Architecture. This is done so you can do the following:

- disassociate the cube definition process from cube creation, thus enabling you to create a cube by using its stored definition
- define and enforce security at the SAS Open Metadata Architecture level
- manage and control the data source in the centralized SAS Metadata Repository

Documentation about the SAS Open Metadata Architecture is available at **`http://support.sas.com`**.

SAS Workspace Server

The SAS Workspace Server enables client applications to submit SAS code to a SAS session using an application programming interface (API). The SAS Workspace Server provides access to SAS software features such as the SAS language, SAS libraries, the server file system, results content, and formatting services.

A program called the object spawner runs on a workspace server's host machine. The spawner listens for incoming client requests and launches server instances as needed. You can run as many instances of workspace servers as are needed to support your workload.

SAS OLAP Server

SAS OLAP Server is a scalable server that provides multi-user access to the data that is stored in SAS OLAP cubes. This server is designed to reduce the load on traditional back-end storage systems by quickly delivering summarized views, irrespective of the amount of data that underlies the summaries.

Processing data by using a multi-threaded kernel enables you to take advantage of your server's parallel processing abilities. SAS OLAP Server accepts data queries in the

industry-standard MDX query language, which opens it up to a variety of clients. The following features are also included:

- the SAS OLAP Cube Studio user interface, which is an alternative Java interface for building and maintaining cubes
- PROC OLAP for programmatically building and maintaining cubes
- server management by using SAS Management Console
- support for processing external aggregates
- support for OLE DB for OLAP

Note: OLAP queries are performed by using the Multidimensional Expressions (MDX) query language in client applications that are connected to the OLAP server by using OLE DB for OLAP (an extension of OLE DB that is used by COM-based clients), or through a similarly designed Java interface. △

SAS Stored Process Server

In addition to the other servers that SAS OLAP uses to store and process cube data, the SAS Stored Process Server is used to execute SAS Stored Processes. A stored process is a SAS program that is stored on a server and can be executed as required by requesting applications. SAS Stored Processes can be used in several different client applications that access SAS OLAP cubes and report on OLAP cube data, including the following:

- SAS Enterprise Guide
- SAS Data Integration Studio
- SAS Information Map Studio
- SAS Web Report Studio

The ability to store your SAS programs on the server provides an effective method for change control management. For example, instead of embedding the SAS code into client applications, you can centrally maintain and manage this code from the server. This gives you the ability to change your SAS programs and at the same time ensure that every client that invokes a stored process will always get the latest version available.

Stored process servers have MultiBridge connections, which enable multiple processes on different ports of the same server. A program called the object spawner runs on a stored process server's host machine. The spawner listens for incoming client requests and launches server instances as needed.

Why You Should Use Cubes

SAS cubes are designed to offer efficient data storage, fast data access, easy data maintenance, and flexibility in data management. The following sections explore cubes and multidimensional storage.

Cube Usage and Storage Space Reduction

While cubes are the format of choice to guarantee fast query response times against your data warehouse, SAS OLAP cubes are also often a very space efficient choice for data storage. In many cases, a basic cube without additional aggregations can be smaller than the input data because the process of creating the cube consolidates

records. SAS OLAP cubes use the hierarchy information for efficient aggregations storage. SAS OLAP cubes also deal efficiently with data sparsity by using virtual placeholders for empty cells. This removes the need for any physical representation of empty cells. A good rule of thumb is, the larger your input data, the greater the storage gain by loading data into a cube.

Multi-Threading Capabilities

Loading data into cubes and executing queries against the cube take advantage of the multi-threading capabilities of your server machine. Aggregations are created in parallel at cube build time. The creation of individual aggregations takes advantage of the Parallel Group-By capabilities of SAS' data engine. At query execution, the multi-threading capabilities of your server machine are fully used to concurrently serve queries by multiple users. Both query evaluation and data access are executed in parallel. To further increase query performance and reduce disk access, you can allocate additional memory on your server to be used for an in-memory aggregation cache.

Easy Setup and Maintenance

A cube is the physical representation of your logical dimensional model. The tools that are provided to update and maintain the cube reflect the multidimensional model, which makes both setup and maintenance of your cube as intuitive as possible. SAS' thin-client, Web-based administrator interface, SAS Management Console, enables you to set up and manage OLAP servers. SAS OLAP Cube Studio provides the workspace and cube designer tools that you need to create and maintain cubes. You can also use the SAS OLAP procedure to create and maintain cubes in a batch environment.

Data Management: Choosing Your Own Tool

If you create your own aggregations by using data management tools such as SQL, PROC SUMMARY, or the tools of your preferred relational database management system (RDBMS), then you can link those aggregations to your cubes without replicating the data within the cube. Any queries against those aggregations are executed by the appropriate SQL engine, and take advantage of any capabilities that engine might have. This allows you the flexibility to use the data management tools of your choice. It also allows you to distribute your data for your cube aggregations across multiple database systems, servers, and platforms. If you choose to let the cube builder create the aggregations, then you can control where to store the data and index files for each aggregation.

Analyzing Your Data

Data Preparation and Dimension Design

The goal of an OLAP system is to have data that is organized, available, and presented as relevant information to decision makers. OLAP cubes are based on data from data warehouses. A data warehouse consists of data that is extracted from

transactional systems at regular intervals. The extraction process works very closely with data quality control, making sure that the data is complete and accurate. Extensive data cleansing (which includes eliminating variant spellings of names) can be part of this task.

Building a data warehouse also implies transforming data that is optimized for transactional processing into data that is optimized for user-driven analysis. Part of that process is grouping facts and attributes into entities that correspond to the users' view of the organization. These groupings are known as dimensions. For related information, see the SAS online documentation for SAS Data Integration Studio.

An established technique for implementing a dimensional model is to create star join schemas that are based on the data. SAS OLAP cubes can be loaded from star schemas, or from further denormalized tables or views that include some or all dimensions in the fact table.

Data Tables Used to Define SAS OLAP Cubes

Detail Tables

A detail, or base, table is any table defined in a SAS Metadata Repository that contains the measures and levels for a cube. The detail table consists of unsummarized data that must include one column for each level and one numeric analysis column for each set of measures that will be generated.

Fact Tables and Dimension Tables

A star schema uses a set of input tables that are defined in the SAS Metadata Repository. The set of tables includes a single fact table and one or more dimension tables. The fact table must contain one numeric analysis column for each set of measures that will be generated. For levels, the fact table will either contain the columns for the levels of a dimension or it will contain a key column that links the fact table with a dimension table that contains the columns for the levels of a dimension. These statements are also true for star schemas:

- A dimension can be in the fact table. In this case, all the level columns are in the fact table and no fact or dimension key is required.
- If the dimension levels are defined in a dimension table, all the level columns for that dimension must be contained in the same dimension table.
- Both the dimension keys and fact keys are single columns, not combinations of columns.
- The dimension key can also be a level in the dimension.

Aggregation Tables

Aggregation tables are fully summarized external relational tables that are used to build cubes. All aggregation tables must contain a column for each measure in the cube where the statistic for the measure is one of the following: N, NMISS, SUM, MAX, MIN, or USS. Columns for derived measures cannot be stored on the aggregation table and are ignored if they exist. Derived measures are always computed at query time. An aggregation table can be used in two ways:

- as an NWAY data source for the cube. In this case, the table must contain a column for every level in the cube and a column for every stored measure.

- as a subaggregation for the cube. In this case, the table must include a column for each level of the aggregation and a column for every stored measure.

Drill-Through Tables

Drill-through tables are views maintained by the user that represent all of the input data used to define a cube. The drill-through table name can be set either when the cube is first created or during an update. Drill-through tables can be used by client applications to provide a view from processed data into the underlying data source.

Aggregation Design

Efficient drilling or traversing of the cube data is a key factor in flexible and quick decision making and analysis. In order to maintain speed and consistency in reporting, data is usually precalculated or aggregated. An important factor in query performance is good aggregation design, which includes decisions about total storage space, available build time, storage location, and storage format.

When planning your data storage and design, it is helpful to approximate the size of aggregations. A basis for estimating aggregation size is the number of distinct values in a dimension level, otherwise known as cardinality. The other factor that determines aggregations size is density. Density is a measure of how many members of each dimension in an aggregation occur in combination with the members of the other dimensions (for example, there might not be sales of a specific product on a specific date). The total cube size as well as the resources that are available for the cube build process determine the build time that is needed. It is also important to note that build time should not exceed the cube update interval.

Aggregation size and available hardware influence your choices for aggregation partitioning. You can separate aggregations into multiple files. A reduced file size might accelerate OLAP server access time, particularly if multiple processors are available for multi-threaded processing. You can use preaggregated summary tables, the cube's own efficient aggregation storage, or a combination of both. Using indexes on either storage type might increase query performance, while also increasing storage space and build time.

After an initial aggregation design is chosen, subsequent cube builds enable you to optimize the cube's performance and size by adding or removing aggregations. You can analyze the OLAP users' behavior by using Application Response Measurement (ARM) logs, showing which aggregations are needed most and would be the most efficient.