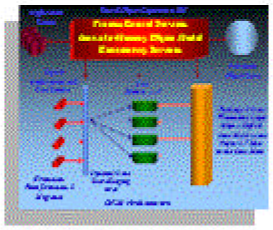

DKMS Briefs

Joseph M. Firestone

DKMS Brief No. Five: Is Data Staging Relational? A Comment

Introduction

In the data warehousing process, the data staging area is composed of the data staging server application and the data store archive (repository) of the results of extraction, transformation and loading activity. The data staging application server temporarily stores and transforms data extracted from OLTP data sources and the archival repository stores cleaned, transformed records and attributes for later loading into data marts and data warehouses.

A recent question raised by Ralph Kimball [1] is whether the data staging area is relational or has more to do with sequential processing of flat files. He concludes that "most data staging activities are not relational, but rather they are sequential processing. If your incoming data is in flat-file format you should finish your data staging processes as flat files before loading it into a relational database." [1, P. 71] He also states that if both the source and target databases are relational it may be appropriate to retain this format and not convert to flat files.

This answer to the question of the character of the data staging area assumes that the issue boils down to the nature of the file processing associated with the database used in the data staging database servers. But I think the issue is broader than this, and encompasses both the database format and process logic characteristic of: the data staging application, the archival repository, and the metadata and associated metamodel driving the data staging process. This brief examines the above issue. It briefly describes the data staging process, and then discusses the nature of the data staging application, repository, and the metadata and metamodel that drive the data staging process.

The Data Staging Process

The data staging process imports data either as streams or files, transforms it, produces integrated, cleaned data and stages it for loading into data warehouses, data marts, or Operational Data Stores. Kimball, [2] and Kimball, Reeves, Ross and Thornthwaite [3] provide clear detailed accounts of the specific work that is performed in data staging. There is no need to repeat the details of their descriptions. But I would like to highlight a few points relevant to the later discussion.

First, Kimball et.al., distinguish two data staging scenarios. In (1) a data staging tool is available, and the data is already in a database. The data flow is set up so that it comes out of the source system, moves through the transformation engine, and into a staging database. The flow is illustrated in Figure One.

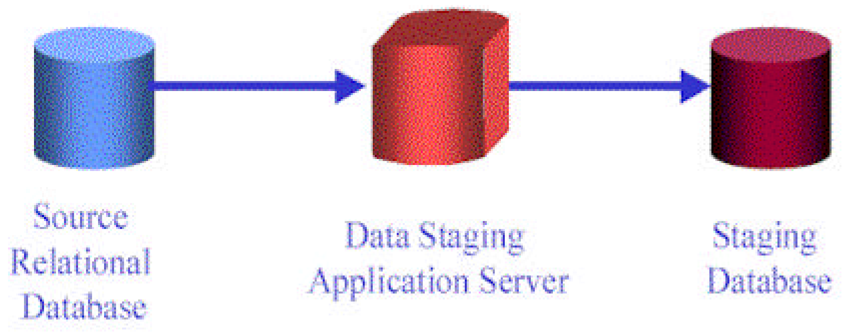

Figure One -- First Data Staging Scenario

In the second scenario, begin with a mainframe legacy system. Then extract the sought after data into a flat file, move the file to a staging server, transform its contents, and load transformed data into the staging database. Figure Two illustrates this scenario.

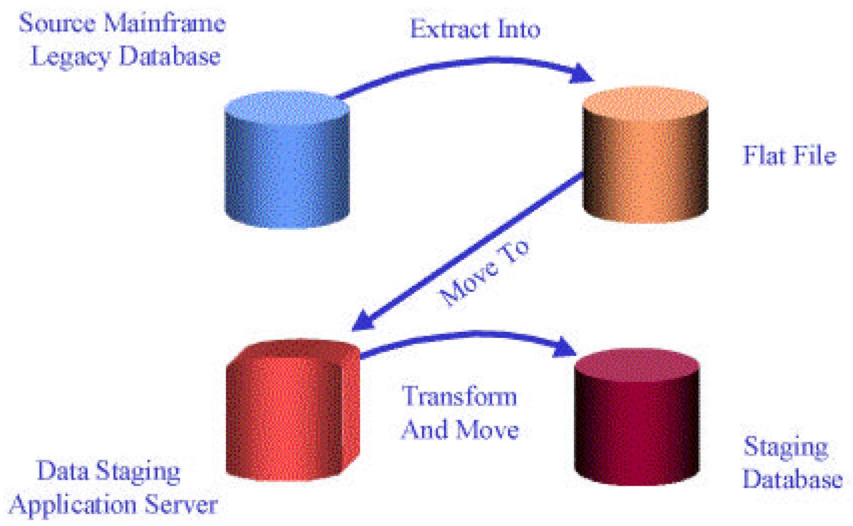
Source Mainframe Legacy Database
Extract Into
Flat File
Move To
Transform And Move
Data Staging Application Server
Staging Database

Figure Two -- Second Data Staging Scenario

Second, "almost all processing in the data staging process is sorting, followed by a single sequential pass through one or two tables." [3, P. 616]. This suggests that a relational data staging server is not desirable, in general, though it may be advantageous if both the input data source and the eventual target for loading processed data is an E-R structured database. Otherwise, flat files may be the preferable physical format for the batch sequential processing data staging server application, and perhaps for the data staging database as well.

Third, "the data staging area will archive and store data for a number of purposes. Conformed dimensions are created in the data staging area and replicated out to all the requesting data marts. These conformed dimensions must be permanently housed in the data staging areas as flat files ready for export. The data staging area may be the best place to hold data for emergency recovery operations, especially if the data mart machines are remote affairs under the control of user departments. The data staging area must also be the source of the most atomic transactional data, especially if the client data marts do not use all of that data at any given point in time. This atomic data then becomes available for further extraction. Again, this archival data may well be stored as flat files available for export or processing by a variety of tools." [3, P. 345]

Fourth, the data staging process is driven in an essential way by metadata, including business rules. Metadata is used along with administrative tools to guide data extractions, transformations, archiving, and loading to target data mart and data warehouse schemas. What is the nature of this metadata? Should it be relational in character? Is it handled best through flat files? Or is there yet another alternative?

Database Type and the Data Staging Application Server

What format should the data be in prior to transformation? If the required data is already in a relational database, use that database for the data staging application server. Don't take resources to transform into flat files, especially if the target presentation database is relational, and is based on the same product. But relational structure is not as efficient for transforming data as a flat file structure, so there will be a performance penalty, which hopefully will be compensated for by the ease of loading into the target relational database.

If the required data is in a flat file, keep it in the flat file. Data staging and cleansing tools have sophisticated batch sequential processing capabilities for sorting and merging flat files; and since almost all processing in data staging is sorting and merging, it wastes time and resources to convert to any other format prior to transformation.

Database Type and the Data Staging Repository

What format should the data be in following transformation and prior to loading? If the data was in relational format prior to transformation processing, then keep it in that format for immediate loading into a presentation relational or multidimensional database server. If dimension and fact tables are to be archived for future data marts, then output these in flat file format, as this is most convenient for export to a variety of tools and applications.

If the data was in flat file format before processing, keep it in flat file format for immediate loading into a relational database. Or if you want to archive it, keep it in flat file format for later export to various applications or to data marts.

Metadata, Metamodel, and Data Staging

The data staging process is metadata and metamodel driven. The metadata is currently expressed in relational format, and the associated metamodel that uses metadata to drive data staging, is expressed as procedural code, frequently scripted in a language provided by a tool vendor.

Both Metadata and its associated metamodel could be expressed in an object model, where metadata would be encapsulated as object attributes, and the rules defining the metamodel could be encapsulated as object methods. This is the trend in data staging process development.

It is reflected in the appearance of such tools as Template's EIT, [4] and Daman Consulting's InfoManager. [5] It is reflected in Informatica' s [6] commitment to develop it's MDX2 as a DCOM-compatible object model. It is also reflected in the development of Distributed Knowledge Management Architecture, [7] the architecture of Distributed Knowledge Management Systems (DKMS). [8]

DKM architecture may be viewed as adding an object layer to architectures based on relational constructs and logics. [9] The object layer is added to provide integration through automated change capture and management. Figure Three depicts DKM architecture in a data warehousing context from the viewpoint of the role of the object layer.

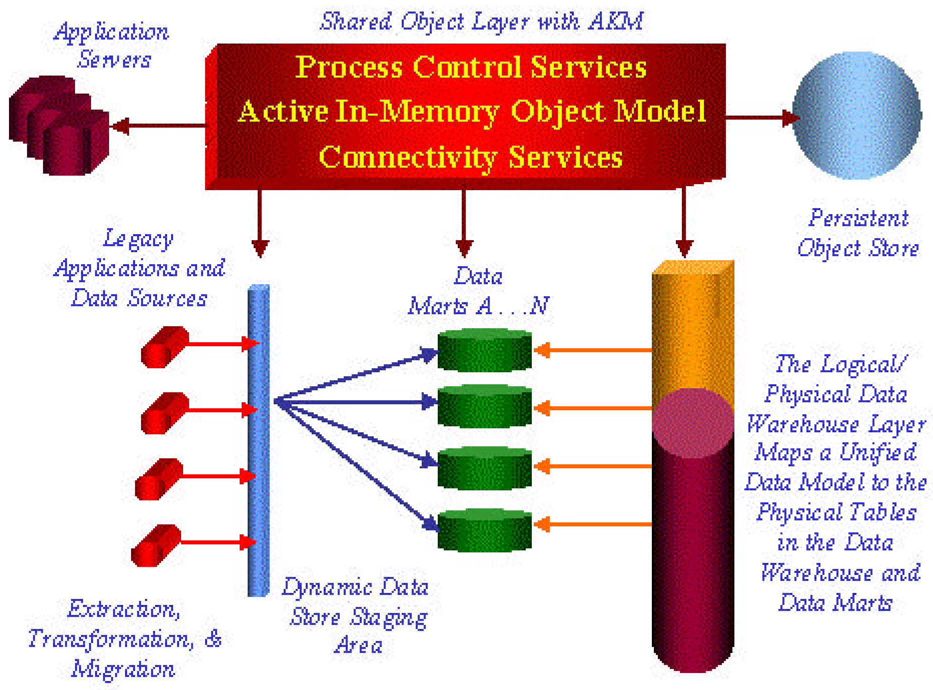
Shared Object Layer with AKM
Process Control Services
Active In-Memory Object Model
Connectivity Services
Application Servers
Persistent Object Store
Legacy Applications and Data Sources
Data Marts A . . . N
The Logical/ Physical Data Warehouse Layer Maps a Unified Data Model to the Physical Tables in the Data Warehouse and Data Marts
Dynamic Data Store Staging Area
Extraction, Transformation, & Migration

Figure Three -- DKM Architecture

In Figure Three the object layer unites and integrates all DKMS components. The object layer requires an architectural component called an Active Knowledge Manager (AKM) [10]. An AKM provides process control services, an object model of the DKMS, and connectivity to all enterprise information, data stores, and applications. The AKM's object model includes entity objects encapsulating metadata and control objects encapsulating related business process rules (metamodels). The preferred format for persistent metadata and metamodel storage is the format of an Object-Oriented Database Management System (OODBMS). The OODBMS form is not necessary for data staging metadata and metamodels, since the AKM can access persistent metadata in a variety of formats. But it is the only form that avoids the performance penalty caused by the "impedance mismatch" between the AKM and relational, flat file, hierarchical or other non-O-O forms of data storage.

Conclusion

The data staging area is not simply relational, but it is also not simply sequential/flat file in character. Table One provides a summary of the variation in preferred storage format across different aspects of the data staging area.

Table One -- The Data Staging Area and preferred Database Type

Data Staging Component	**Data Base Type**		
	Flat File	**Relational**	**OODBMS**
Data Staging Application Server	If already in Flat File	If already Relational	Waste of Resources
Data Staging Repository	If already in flat file or if dimension and fact tables are to be archived	If already Relational	Waste of Resources
Metadata and Metamodel Repository	----	-----	Preferred Format Is OODBMS

The picture offered in Table One is, more detailed than previous descriptions of the problem, and makes the point that data staging has multiple aspects and not just file processing. It shows appreciable roles for all three types of storage somewhere in the data staging process. Flat files and relational databases are dominant, but there is a definite role for OODBMSs, and it is one that will probably increase along with the commitment to metadata in data warehousing.

References

[1] Ralph Kimball, "Is Data Staging Relational?" DBMS, April 1998, Pp. 14, 16, 71

[2] Ralph Kimball, The Data Warehouse Toolkit (New York: John Wiley, 1996).

[3] Ralph Kimball, Laura Reeves, Margy Ross, and Warren Thornthwaite, The Data Warehouse Life Cycle Toolkit (New York: John Wiley & Sons, 1998)

[4] Template Software, "Integration Solutions for the Real-Time Enterprise: EIT - Enterprise Integration Template," Dulles, VA, White Paper May 8

[5] Inquire at: http://www.damanconsulting.com

[6] Informatica's "second generation" Metadata exchange Architecture includes a commitment to COM and Object Technology. According to Informatica: "MX2 includes a powerful, object-oriented framework based on Microsoft's COM technology, making it interoperable with other COM-based programs and repositories, including the Microsoft Repository. MX2 will comply with the Unified Modeling Language (UML) an Object Management Group standard currently supported by Microsoft, Informatica and other industry leading IT companies."

[7] "Architectural Evolution in Data Warehousing." available at http://www.dkms.com/White_Papers.htm.

[8] I introduced the DKMS concept in two previous White Papers "Object-Oriented Data Warehouse," and "Distributed Knowledge Management Systems: The Next Wave in DSS." Both are available at

http://www.dkms.com/White_Papers.htm.

[9] "Architectural Evolution . . ." P. 12-14

[10] Ibid.

Biography

Joseph M. Firestone is an independent Information Technology consultant working in the areas of Decision Support (especially Data Marts and Data Mining), Business Process Reengineering and Database Marketing. He formulated and is developing the idea of Market Systems Reengineering (MSR). In addition, he is developing an integrated data mining approach incorporating a fair comparison methodology for evaluating data mining results. Finally, he is formulating the concept of Distributed Knowledge Management Systems (DKMS) as an organizing framework for the next business "killer app." You can e-mail Joe at eisai@home.com.

[Up] [Data Warehouses and Data Marts: New Definitions and New Conceptions]
[Is Data Staging Relational: A Comment]
[DKMA and The Data Warehouse Bus Architecture]
[The Corporate Information Factory or the Corporate Knowledge Factory]
[Architectural Evolution in Data Warehousing]
[Dimensional Modeling and E-R Modeling in the Data Warehouse]
[Dimensional Object Modeling] [Evaluating OLAP Alternatives]
[Data Mining and KDD: A Shifting Mosaic]
[Data Warehouses and Data Marts: A Dynamic View]
[A Systems Approach to Dimensional Modeling in Data Marts]
[Object-Oriented Data Warehousing]