Paper 133

How and Why a SAS® Data Warehouse Is Used to Manage the 2000 Decennial Census

Gail S. Davidson, U.S. Bureau of the Census, Washington, D.C.

ABSTRACT

The Decennial Census is the largest undertaking of the U.S. government, excluding war. The Management Information System (MIS) used to manage the 2000 Decennial Census operations is a complex, top to bottom SAS data warehouse. The warehouse data is stored on a UNIX server in a series of SAS data structures and is expected to grow to approximately 1TB in size by 2003.

This paper will answer the questions: why was SAS chosen as the preferred data repository, not a traditional database product? What were our goals and constraints that shaped the design of the warehouse? And last but not least, shortcuts and lessons we learned along the way to enhance performance of loading and mining the warehouse data.

INTRODUCTION

The Decennial Census is very complex and consists of many operations that must be completed on time and within budget. It requires a sophisticated Management Information System (MIS) to make sure it is well managed. The Cost and Progress System (C&P), part of the Decennial MIS, reports and stores daily information about the various census operations in a SAS data warehouse. The information is mined by census managers and executives through a SAS desktop application utilizing SAS/AF® and SAS/EIS®. Given that most up to date information about the progress of the census operations and the associated costs to date is at their fingertips, the managers and executives can take a pro active versus reactive management role. The SAS data warehouse and mining tools provide managers the ability to easily monitor the operations, conduct trend analysis, and conduct cross-operation analysis while the operations are ongoing. In previous Censuses they did not have this ability.

The Census Bureau chose SAS as the data warehouse repository for the Cost and Progress System for the very simple reason that a traditional relational database was not required. A data warehouse by definition is a read-only repository of data. Traditional relational database software packages provide a lot of bells and whistles that are not absolutely required in a data warehouse and these extra bells and whistles could potentially weigh down system performance. We did not need added security provided by a traditional DBMS, all our security requirements could be satisfied at the system level. Another layer of complexity that a traditional DBMS provides that the C&P did not require was enforcement of key relationship and constraints. Our constraints and key relationships are relatively simple and can be easily handled in the load programs. Another reason the Census Bureau chose SAS for the C&P is that we could get everything we needed to load, store, and mine the data from one vendor. In short, SAS was able to provide all we needed to build the warehouse and reporting application with a minimum of system overhead and with minimum complexity.

GOALS

When we started to design the logical structure of the data warehouse, we began by asking ourselves what the most important goals of the system should be. What we settled on was basically common sense and probably should be the first goal in any management information system. The number one goal was: it should be as easy as possible for the users to find the information they seek. A very close second goal was that predefined reports and graphs should be displayed on the user's desktop in less than five seconds and ad hoc data requests should be as easy to construct as possible for the users. Obviously, from day one, the focus has been on our users. Efficiency of storage was not one of the primary goals, storage is relatively cheap, the time of census managers and executives is not. The census managers and executives know the operational details well but some are very technically challenged when it comes to getting information from automated systems. These are our primary users so the warehouse and user application had to be designed such that the information could easily be found and ad hoc queries could be constructed with minimal SQL joins.

LOGICAL DESIGN

We had a general idea of the type of operational data that needed to be stored based on requirements that users provided to the MIS staff in a series of meetings. We also knew that the tool used to mine the data and provide canned reports would make use of SAS/AF and SAS/EIS. Those tools for information delivery drove the design of the data warehouse. The challenge for the design was to satisfy the goals of easily finding data, downloading it to the users with decent performance and utilizing all the wonderful GUI features on a Microsoft Windows 95 platform.

To determine the final design, we tested two alternative warehouse designs. We interviewed some potential users and conducted testing labs to evaluate the different warehouse designs. We loaded each of the warehouse test designs with the same information which represented the same volume of data we expect for the census operations. Then, we asked the testers to quantify the different designs based on how easy it was for them to find the information they were looking for and how satisfied they were with the performance. We then had the testers, who were familiar with census operations, ask a few general questions of each warehouse design and rate how easy they got the results they needed and how quickly the answers were returned. From these test results we were able to decide on a high level logical design for the data warehouse.

One of the designs we tested was traditional database design, mostly normalized, containing detail tables, lookup tables, and related time tables. The testers quickly rejected this design because the queries were too hard to build. Creating queries that joined tables was far beyond their technical capabilities. In addition, the time it took to download the requested data was much too slow. Each time information was requested for a specific operation many tables needed to be joined and filtered.

The design that was finally settled upon was an operational centric design. All the information about each distinct census operation is stored in a series of related tables and MDDBs, separate from information on the other census operations. Most queries only access data for one specific operation at a time. The first iteration of the final design started with a series of three data structures used to store data for each operation. These were a base table which has all the information we've ever collected on an operation (a complete historical record), a table which only shows that latest snapshot of the operation, and an MDDB of the snapshot data table for the user application to access. The last two data structures, the snapshot and MDDB, were added for data delivery and performance reasons.

PHYSICAL DESIGN AND DATA FLOW

The data is stored on a Silicon Graphics Challenge L server. It is refreshed nightly, and users access data on demand through their desktop applications. Figure 1 represents the flow of data from the source systems to the users. As the volume of data grows in the warehouse, files which are not accessed frequently will be moved to a near line storage device, a robotic tape library. We plan to implement this strategy with hierarchal storage management software. Currently we have enough online storage capacity to hold all the files for past and current operations. But, since data will never be purged, the volume is cumulative, we will need to move data off the online disk in the near future.

Each night while an operation is ongoing data may be loaded from up to five different source systems. The load programs take the data from the different source systems and summarize and collate it to represent one big picture of the operation's progress and cost. For example, there is one source system that tracks each employee's work progress, i.e., how many housing units they have enumerated that day. There is another system that tracks how many hours that employee had worked and how much mileage they need to be reimbursed for. The C&P system loads data from both of those systems and summarizes the data to the field office level to show the relationship between that office's progress and costs to date. We add one record per office per day to the base table for each operation that office is working on.

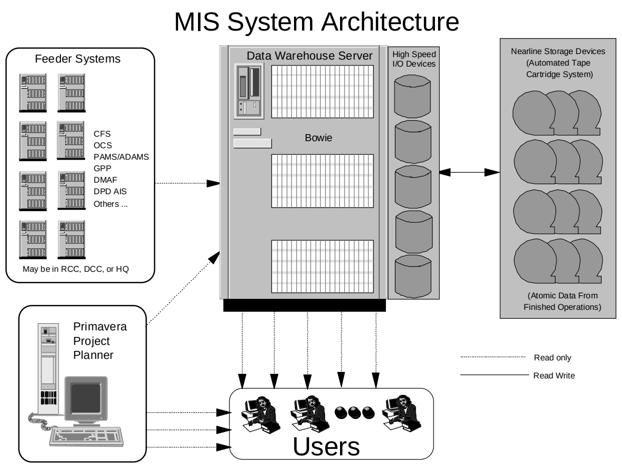

Figure 1

USER APPLICATION

The fact that we knew the primary method of data mining by our users would be a SAS/EIS application, in particular multidimensional reports, did have an impact on the structure of the warehouse. That is what drove the decision to create the snapshot data sets. The MDDBs use the snapshot data sets as their base tables and were added only to boost performance of the user application.

Most of the users access only the snapshot data through the canned MDDB reports. This gives them a picture of the progress and costs to date. Are we where we expect to be given how much money has been spent? Have we hired enough people to complete the operation on time? All the basic information needed to answer these questions is stored on an MDDB data structure.

The base tables and the snapshot data sets have the same structure. The only difference is that the snapshot data sets are replaced every night with the most current data and the base table acts as a historical record and grows with each load cycle. The historical record is very important because sometimes our managers have to justify decisions they've made months past. Our application is the only place at the bureau where there is a historical record of what the data was on each day of the operation.

LESSONS LEARNED

When we started the Census operations, the user application performance ground to a halt because of the massive amount of data, 70,000 records per day for some operations. This was because the report title's method was reading from the snapshot data set and not the MDDB to pull off certain dates. Each operation takes place at many different sites and each site may have different end dates. The application scanned the snapshot data set and pulled off the max date regardless of the site. We alleviated the problem by storing the max fields used in the report titles in a separate data set. This added a fourth table that is updated during the load cycle and is stored for each census operation. The title information data set has only one observation and only two variables that are used in the report titles. Changing the design of the titles method and the creation of the extra data set took what was an eight minute process down to less than one second.

Another lesson we learned that considerably improved the wall time of the nightly data refresh was that we need to use data step as much as possible. Most of the data we refresh with nightly is stored in Oracle® database tables. We create permanent pass through views to the Oracle tables using SAS/ACCESS®. My background is primarily in database, Oracle and RDB. Therefore, when we started writing the load programs we used a lot of PROC SQL. In some cases we still use PROC SQL to compute some variables in the warehouse, but wherever possible data step is used because it is much faster. To quantify this performance improvement, we had one load program that was taking five hours to load with most of the variables computed in PROC SQL. When we changed the code to use data step, the load process took only two hours.

In addition, the job control system for the load programs was improved greatly when the census operations began. When we first started this data warehouse, we were still in the dress rehearsal phase of the census. The dress rehearsal is a field test of all the operational procedures and a shake down of all the data processing systems. The dress rehearsal only took place at three sites and the census is staged from 450+ sites. The C&P processes loaded all information for each operation from one batch SAS process, i.e., one site, then the next site and so on sequentially. We knew that for census operations that algorithm would be insufficient because of the increase in the number of sites. We created a job control system that would query each of the sites independently. After all the processing at each of the separate sites completed, another process would collate all the sites data and bring it together to present a national view of the operation.

CONCLUSION

To summarize what we have learned from so far in the C&P data warehouse can basically be boiled down to common sense.

1. Design the data layout the way the users think of the data. Our users were nontechnical, program managers. They need to be able to find the data about their program areas and operations without having to call a programmer.

2. Keep the data mining tools in mind. You don't want to have to store the data twice, once for the warehouse and then massage it to fit the delivery tool.

3. Minimize the data volume passed to the clients and pre summarize wherever possible.

4. Break the data refresh processes up into many independent pieces that can be run concurrently. This will save many hours of wall time while loading the the data.

If you follow these common sense guidelines, you will have a good system that stores massive amounts of data but is delivered to the users easily and with decent performance.

ACKNOWLEDGMENTS

SAS®, SAS/ACCESS®, SAS/AF®, SAS/EIS® are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. Oracle® is a registered trademark of the Oracle Corporation. ® Indicates USA registration.

Other brand and product names are registered trademarks or trademarks of the respective companies.

CONTACT INFORMATION

Gail S. Davidson
DMD/MIS 1422-2
U.S. Bureau of the Census
Washington D.C. 20233
e-mail: gail.s.davidson@ccmail.census.gov