Veterans Affairs Canada

Analyzes Data With ACL

By Steve Force

Government agencies, like private organizations, are searching for ways to cut costs, streamline the delivery of IS services and empower end users. The Department of Veterans Affairs for Canada (VAC), with 3700 employees and a budget of $1.8 billion (Canadian), is in the final stages of an initiative that has put new power in the hands of end users, allowing them to retrieve and analyze data without requiring time-consuming IS intervention. The results have been gratifying: reduced costs, improved service levels and more timely and effective decision making.

"When VAC began this initiative in 1988, the main challenge was to access and report on legacy data," reports Bob Gardham, chief of financial systems development at VAC's headquarters in Charlottetown, Prince Edward Island. VAC maintains extensive databases on Canadian veterans on an Amdahl 5870 mainframe system operating under MVS. The primary DBMS is CA-DATACOM (Computer Associates, Islandia, NY).

"There is a continuing need to be able to efficiently access the information in these databases and create reports required by changing and ongoing projects," Gardham notes. The previous system for generating reports, in which end users would submit requests to IS, was inadequate to meet the emerging needs of the organization. VAC experienced all the limitations of the traditional reporting process. "It often took up to three months to receive the results of a request," Gardham says. "If there were errors or questions, either by IS or the person requesting the report, the process would begin again."

Other problems plagued the agency. For example, once a report was obtained, there was further need for additional analysis to verify the accuracy of data or to reconcile it with other information. The manual requirements to accommodate some of these needs were so extensive that they overwhelmed the IS staff. Other difficulties involved reconciliation of payments made by contractors to pharmacies and other treatment providers on VAC's behalf.

A Tool For Finance-Oriented Professionals

In 1988, VAC began seeking a more modern solution for its financial internal control application needs. Available mainframe tools were deemed inappropriate because they required too much technical training to be used by finance-oriented professionals. VAC was committed to giving end users a tool they could use without extensive training to meet a large percentage of their information collecting and reporting requirements. The agency sought a tool that would allow nontechnical end users to extract mainframe data and download it to their desktops for data analysis and reporting.

VAC embarked on an extensive review of the available alternatives and tested a number of host- and PC-based data management products. Eventually, thanks to a recommendation by the office of the Comptroller General for Canada, VAC evaluated ACL, a comprehensive data access and analysis program from ACL Services, Inc. (Vancouver, BC, Canada and Encino, CA). Initial tests indicated the system was faster in analyzing data than many of the alternatives the finance department evaluated.

Introduced in 1985, ACL has evolved into a general data inquiry and reporting system for end users and is available for DOS, Microsoft Windows 3.1 (Microsoft Corp., Redmond, WA) and MVS users. Users may interrogate any data structure on mainframe or PC platforms and perform complex file analyses. The system has both client and server functions. It works with most networks, hardware platforms and other software systems. It can extract files or portions of files from the mainframe (or any server). It can deliver that extracted data directly to the PC (or any client) for further analysis and reporting.

ACL requires no computer programming or coding experience to operate. Users do not need to know the structure of the file being analyzed. ACL provides intelligence to tell the user what of interest is in the file and what files are available for interrogation on any particular platform.

Sort, Sample, Analyze And Report

For the VAC's first application, ACL extracted information from tape drive files created on the mainframe. In conjunction with a nine-track tape drive connected to a desktop workstation, ACL extracted data from the 6250 BPI mainframe tape to PC files. In this process, ACL automatically converted EBCDIC format to standard ASCII format. This feature allowed nonprogrammers to sort, sample, analyze and report on the detailed financial data directly from billing tapes provided by outside insurance contractors.

After the initial use of the product, VAC installed six additional copies of ACL for use by finance personnel. Currently, a total of 14 copies of ACL are in production: five in VAC regional offices, eight in the head office in Charlottetown and one in Ste. Anne de Bellevue Hospital, Montreal, Quebec. Application usage continues to expand into areas not originally envisioned, such as using ACL to discover and utilize data structures that were undefined or unfamiliar to users.

As other VAC divisions saw the uses to which the finance division put ACL, they also began to use the system. For example, the programming department uses ACL to help debug IS reports. ACL is also used as a tool for analysis and quality control of large files and databases. Today, ACL is used throughout the organization to benefit all divisions, including finance, management, quality control and operations.

The system resides on the finance division's NetWare 3.11 (Novell, Inc., Provo,

UT) LAN so it can be used network-wide. It has become a standard part of ad hoc reporting, analysis and even production systems. "VAC averages 35 ad hoc inquiries daily. Routine jobs are scheduled for batch processing. Batch jobs can be set up and changed as required. These batch jobs can be scheduled to run nightly for multiple applications by all personnel," Gardham points out.

After VAC personnel extract mainframe data, they either load it into word processing documents or transfer it to a dBASE (Borland International, Scotts Valley, CA) file format for general use in other database programs. "ACL can be used to verify the integrity of the data within dBASE files," Gardham notes. "Other uses include the transfer of ACL output into graphics programs for colorful presentations to managers, other departments or other government agencies. And all of this can be done within the PC environment or with a mix between PCs and mainframes or other hardware platforms."

End-User Empowerment

ACL empowers end users by reducing mainframe hardware, software and personnel resource dependency. No computer or programming experience is required. ACL actions can be performed through pop-up menus or through its simple but powerful command language. Help screens are available throughout. Prompts and available data are automatically presented to users on request. A single keystroke displays file or data choices in a pop-up window. Reports have default formats that users can change conversationally as required.

The finance division uses the PC version of the product. ACL on the mainframe is virtually the same as the PC version. Users are able to access mainframe data such as SQL, IMS, DB2, VSAM or other complex data structures and present that data to the user in a clear and uniform format. Once that is done, users can easily identify those files or file portions they wish to extract and download; since ACL can do the extraction analysis directly on the mainframe, users download only required data.

The major goal of giving finance division end users a sophisticated yet easy-to-use data management tool has been met, Gardham believes. The system not only is simple to operate, but provides a tutorial function.

The LEARN command, for example, is a type of macro facility within ACL that allows the user to create a batch file while typing in commands. This batch file can then be run against the data file as needed. If users receive similar files each month, they could run the batch file created under LEARN to sort, stratify or sample the file data automatically.

The ANALYZE command allows users to view a file with virtually no knowledge about the file or its environment. The STRATIFY command produces a file report automatically according to natural groupings with subtotals for each.

By all accounts the VAC finance division is more productive, thanks to its investment in technology. The users are well-positioned to respond promptly to requests for information. The result is higher quality service for the Canadian veterans the agency serves.

ABOUT THE AUTHOR

Steve Force is a freelance writer specializing in applied connectivity and DP solutions. He is based in Clarkston, MI.

DSN8HUFF

Continued from page 38

noting can offer a better understanding of how compressed strings are handled by the sample.

The most obvious point concerns the extra byte by which DSN8HUFF can cause a row to grow. In the process of encoding a string, if the encoded string becomes larger than the original string, DSN8HUFF adds a byte of 'FF'X to the beginning of the original string to arrive at the stored format. This minimizes the impact of expansion by limiting any possible expansion to no more than one byte per row. During decode, the flag byte notifies DSN8HUFF that decoding requires stripping away only the first byte rather than performing a Huffman decode. Since it is possible that the first byte of a string actually encoded with the Huffman technique would be 'FF'X, a flag bit of '1'0 is used to begin the Huffman encoded string. In this way, if DSN8HUFF detects 'FF'X as the first byte of the string to be decoded, it is certain that no Huffman decode is required; anything other than 'FF'X denotes a string that requires Huffman decoding after stripping off the single flag bit.

The second interesting point is that Huffman generates a bit string with no regard to byte boundaries. The record length within DB2 must be in multiples of bytes, so padding is added to the Huffman string to round up the length to a byte boundary. Obviously, the padding bits must not result in the decoding of an extraneous character since DSN8HUFF will continue to decode characters until the entire row length has been processed. Any bit string of 7 bits that does not result in a decoded character can be tacked on to the Huffman encoded string that is not an even multiple of bytes and the row then given to DB2 with a length equal to the highest full byte boundary. For DSN8HUFF, determining what padding string of 7 bits to use is not necessary since '1111111'B will always work. This is a secondary result of the tree transformation that DSN8HUFF requires since you will never be able to reach a leaf node with a string of '1111111'B.

Conclusion

Obviously, the process of building a customized, DSN8HUFF-compatible compression tree is not a job to be done with pencil and paper. You will attempt to build a 256-leaf node tree only once before you write a program to accomplish the job. It is an arduous task and hard to fit on a single piece of paper, as well. The customization is well worth the trouble, however, as it will yield the maximum amount of compression. And, as implemented in DSN8HUFF, maximum compression yields the lowest CPU overhead. Customization of this DB2 2.3 sample application is worthwhile if you wish to put a table on a diet.

ABOUT THE AUTHORS

Dave Hauser has more than eight years experience in DB2 development and performance at IBM's Santa Teresa Lab. IBM Corp., M60/B220, P.O. Box 49023, San Jose, CA 95161.

Greg Davoll has four years DB2 utility development experience at IBM's Santa Teresa Lab. IBM Corp., M09/B423, P.O. Box 49023, San Jose, CA 95161.