

Business Intelligence

THE LEADING PUBLICATION FOR BUSINESS INTELLIGENCE AND DATA WAREHOUSING PROFESSIONALS

JOURNAL

A 101communications Publication

The Critical Business Need to Reduce Elapsed Time

Vic Werner and Craig Abramson



The Critical Business Need to Reduce Elapsed Time

Vic Werner and Craig Abramson



Vic Werner is Director of Marketing at Syncsort Incorporated, focusing on promotions, project management, and tradeshow. vwerner@syncsort.com

Craig Abramson is a technical analyst at Syncsort Incorporated, focusing on the latest data sorting, data aggregation, and backup and restore solutions. cabramson@syncsort.com

Remember the old business saying “Time is money”? It certainly rings true today. Packages are sent across the world to arrive the next business day. Contracts are faxed in order to get signatures immediately. Critical decisions are made within minutes. But what happens if the data needed to make those decisions takes too long to access? Perhaps the data warehouse is too packed, or the database takes too long to load, or even simple queries are running too slow.

A frequent problem is an overabundance of data that needs to be streamlined, reformatted, or re-grouped. We explain steps you can take when processing data to reduce elapsed time and improve the results of many applications.

Time Is Money

Whether you’re analyzing information from a data warehouse, reviewing clickstream data to find the latest trends, or running a query on current sales figures, the elapsed time to perform these processes means lost revenue. Fortunately, there are numerous steps you can follow to reduce the elapsed time needed to manage the data and run these various applications. The time savings you can achieve by following any of these steps can translate into significant improvements in your business processes and decision-making abilities. That added time translates into money.

Improving Data Warehouse Performance

If the performance of your data warehouse is an issue, then it’s time to tune your system to reduce the

elapsed time needed to process the data without making major changes to the system itself.

Tuning begins by monitoring the activity and data that moves through the data warehouse. When you're ready to implement a tuning procedure, the easiest adjustment is to add indexes to the data in order to reduce the system resources needed to find information. This technique should be used when the same data is accessed frequently. (Inmon, 2001)

You can also add data marts tailored to meet specific departmental needs. For instance, separate data marts can be created for the financial, marketing, and sales departments, which would reduce the amount of data each group needs to analyze. If your company doesn't want to pay for additional equipment to host the separate data marts, creating summary tables is a less costly option. Users are able to access summary data much faster than information kept in the data warehouse, although additional space in the system is still required for the summary tables. (Inmon, 2001)

Other tuning techniques include removing data stored in the data warehouse but not accessed, partitioning the data, and creating joins between a fact table and several dimension tables. (Inmon, 2001) Delta processing will also help reduce the amount of data in the data warehouse and improve performance. In this technique, an older file and a new file are compared to identify changes that have occurred. Only the changed data is updated in the master file.

Improving Query Response Time with Data Aggregates

When it comes to data warehouse query performance, speed is everything. Faster performance means needed data and analysis is available to act upon—quickly. In order to achieve this goal, data

warehouse experts agree that data aggregates can be a very powerful tool. According to Ralph Kimball, author of *The Data Warehouse Lifecycle Toolkit*, data warehouse administrators can “expect anywhere from a tenfold to a thousandfold improvement in runtime performance by having the right aggregates available.” (Kimball, 1998)

Aggregates are pre-stored summary records held in a data warehouse and are usually built from the base layer upward, culminating in high-level totals. For example, a data warehouse administrator may decide

Nabisco has been able to reduce the amount of time it takes to define and maintain aggregates by up to 40 percent.

to build aggregates for sales based on product, store, and time-related totals (such as for a month or year). Tables of aggregates eliminate the time and resources required to summarize data each time it's needed by a query, allowing the query to be answered very efficiently.

One company that has benefited from the use of data aggregates is the Nabisco Biscuit & Snacks Group, based in Northern New Jersey. Comprised of three divisions—the Nabisco Biscuit Division, the Confections Division, and the Planters Division—the company generates an enormous amount of customer, manufacturing, and sales data. After all, Nabisco sells such iconic brands as Oreo cookies, Ritz crackers, Life Savers candies, Altoids mints, and Planters nuts.

Nabisco's sales, finance, and marketing staff use the data aggregates to analyze both customer SKU and

manufacturing data, consisting of current year-to-date and previous two years' sales information. To create these aggregates, Nabisco used a combination of UNIX joins and SyncSort UNIX running on an IBM RISC 6000 Silver Node with four gigabytes of memory and four processors. The computer system stored Nabisco's 750 gigabyte fact table.

Although this process provided the results the company needed, the Information Services team at Nabisco realized they had a potential problem. While limited in the amount of time they had to create the aggregates, they were receiving a growing number of requests. "With a batch window of only five to six hours a night, we needed a faster methodology to build the multiple new aggregates that are continually required by our users," according to Neil Gurwitz, Project Manager of Information Services. After an extensive review of the products available, the team selected Sigma (from Syncsort Inc., Woodcliff Lake, NJ). The team runs the utility on an IBM RISC 6000. The computer system utilizes AIX 4.4.3, and the warehouse database features Informix Red Brick Release 6.0.2.

Nabisco uses the tool to build 13 aggregates that consist of current and two prior years of sales history. These base tables are rebuilt every four weeks due to store realignments and are updated with daily sales information six times a week. "Specifically, the aggregates are rolling up from the store or market level and consist of: team territory, team class of trade, region class of trade, area class of trade, branch, and account/sub-account on both the SKU and manufacturing. In addition, we have just completed one aggregate from the account/sub-account SKU aggregate which rolls up to a product grouping." Approximately 1,100 users generate 45,000 reports per month using this information. The application is written in PowerBuilder and has dynamic online reporting and dynamic batch reporting.

Since putting this new process in place, Nabisco has been able to reduce the amount of time it takes to define and maintain aggregates by up to 40 percent. It has dramatically cut response time for database queries, too. With the acquisition of Nabisco by the Kraft Company, Mr. Gurwitz saw even more potential for using aggregates. "Once all of the design details were worked out, we saw an immediate need for new aggregates to support the Snacking Company which grew out of the acquisition by the Kraft Company. Long range, we see building additional aggregates to expand our users' reporting requests." With this approach to creating new aggregates, Nabisco is confident that they have the ability to provide the queries their employees need in the fastest and most efficient way.

Faster Database and Data Warehouse Loading

With Nabisco, the data was already stored in the data warehouse. But for many companies, the problem is the elapsed time it takes to load a data warehouse. When faster and more efficient database and data warehouse loads are important, pre-processing the data is critical.

Keep this general rule in mind whenever you have a sizable amount of data to load: *The format and sequence of load data should be as close as possible to its format and sequence in the database.*

The reason for this rule is simple. Although some database engines load quite efficiently, databases are optimized for query processing and related tasks, not for loading. By pre-processing your data before loading, you leave your database free to do the work it's designed for, and you gain these performance advantages:

- Databases will load faster
- Indexing will be more efficient
- Data will be ready to use sooner
- Databases will be free to do other work

We have summarized the five major pre-processing techniques below. Depending on your situation, you may be able to use one or all of them. Remember that the more techniques you use, the faster and easier large amounts of data will load.

Pre-Processing Techniques

Selection

Many sites begin their database and data warehouse loads with a mountain of data gathered from heterogeneous systems and/or different processing locations. The important thing is not just that you select only the data you need, but also that you do the selection as quickly and efficiently as possible. You could create a custom application to do this, but the time spent on writing the code would be too time consuming. Instead, it's faster to add software that is designed to specifically extract the data you specify. Selection should usually be done first, so that you are working with the least amount of data possible as you perform your pre-processing work.

Reformatting

Chances are that records coming from several different systems will arrive with radically different formats. Reformatting allows you to rearrange the fields in these records so that all records have the same load record image. As an added bonus, reformatting lets you eliminate any fields that don't need to be stored in your database or data warehouse tables.

Summarization

Summarization is usually a prerequisite for optimizing a database or data warehouse load utility. You eliminate duplicate records, further reducing the amount of data you are loading. At the same time, you can "sum" a set of records by adding important numeric fields together, then only load one record in a set that contains the total. This kind of "summing" can be a critical element in optimizing query processing with aggregate tables.

Grouping

Grouping lets you split your data into separate partitioned table ranges after it has been selected, reformatted, and summarized. Splitting data into several files during one operation is far more efficient than running multiple select applications to create the same number of files. Each file, or table range, will then be loaded separately.

Sorting

Sorting is normally the second prerequisite (behind summarization) when you are planning to use a fast bulk-load utility. In this case, data is sorted into the

Alexander cut the processing time down to two hours, a 60-percent improvement over the SQL tools he was previously using.

order in which it will be stored in the tables or by which it will be indexed, depending on the load utility you use. If you are not using a bulk load utility, you can still accelerate a load by presorting with the clustered index as the sort key.

Speeding Up Business Intelligence Applications

Making the right decisions in a shorter amount of time—that's what business intelligence (BI) is all about. BI applications are designed to convert raw data into useful information that can be quickly analyzed by key decision makers. The data can reveal market trends to exploit, processes within the company to improve, or the effectiveness of a business plan. The analysis depends on the timeliness of the data, so it's imperative that the information is processed and can be accessed in the shortest amount of time. When a company has a

relatively small amount of data, speed isn't an issue. But when the gigabytes start adding up, some adjustments may need to be made to the BI application.

One organization that encountered this problem was Presbyterian Health Plan (PHP) with their disease management program. PHP is operated as a division of Presbyterian Healthcare Services, New Mexico's largest locally owned healthcare system. It offers a statewide healthcare delivery system and 17 years of experience in managed care.

The company needs to process, analyze, and classify more than 35 million patient claim entries as part of their disease management program. For this project, PHP uses an Intel running Windows NT 4.0 with 4 CPUs and 1.5GB of memory. The database engine is Oracle 8i. Bruce Alexander, Data Warehouse Manager at PHP, was using SQL and PL SQL to prepare the data.

Alexander first unloaded seven gigabytes of raw patient claims data from an Oracle database in delimited format. Next, he sorted the data by patient ID and service date. Additional processing was performed in order to group the data by clinically relevant classifications. He then loaded the data into Oracle for subsequent analysis.

The process was taking more than five hours, and Alexander ran into problems with the SQL tools. He explains, "We have to extract approximately 36-37 million rows of medical claims data. It has to be sorted prior to input into another product called Symmetry Episodic Treatment Groups, which looks for medical episodes across all the data. And in order for that product to work, it requires the data to be sorted in patient sequence. We started hitting some of the limitations of what we could do with the SQL tools, particularly where sorting was concerned."

Alexander used SyncSort (from Syncsort Inc.) to achieve the performance results he needed. The application now begins with a simple text file extract from Oracle. The output of this script is read by SyncSort, which sorts the records using the random number as a key. Alexander cut the processing time down to two hours, a 60-percent improvement over the SQL tools he was previously using.

Reducing the Time to Analyze Clickstream Records

E-commerce creates an enormous amount of data. Web sites generate a large number of clickstream records with every visit from a user. The most popular sites produce as much as a billion records of raw Web data every day. All this data can cripple your system's performance. To improve the elapsed time in the clickstream process, you'll have to cut through this data first.

The following techniques will help improve the management of your clickstream data and optimize the performance of your system.

- *Data Extraction/Data Transformation*—Use Include/Omit processing to quickly identify the exact records you need from the Web logs and reduce the total data to be processed. To achieve more efficient load performance, transform the record layouts by deleting or reformatting specific fields.
- *Merge*—Similar data, such as a list of names, can be merged together into a single file for analysis. During this process, you can specify how the data is ordered.
- *Join*—By joining data, you're matching keys in multiple files and creating one record from two records that have a common key.
- *Pattern Matching Field Extraction*—Search for patterns anywhere in specified fields and then

extract only those portions of the field you need for the analysis.

- *Output Record Numbering*—Add a unique record identifier number to the output. It can start at any value and can be useful for databases with a uniqueness constraint.
- *Web Log Format*—To simplify and reduce development time, utilize the two standard Web log formats: The Microsoft Standard Web log format (Microsoft Professional Internet Services format) and the National Center for Supercomputing (NCSA) Common Log File format.
- *Pre-sort, Bulk Load*—For high-performance relational databases such as Oracle, Sybase, and SQL Server, a bulk load followed by a separate index creation step provides you with the fastest way to load data. These database manufacturers recommend pre-sorting the data before the bulk load so the created index can bypass the sort operation. This can cut the total load time in half.
- *Pre-sort, Regular Insertion Load*—When a bulk load is not applicable, the load step can still be accelerated by pre-sorting the data, using the clustered index as the sort key. Significant savings in load time can be achieved.
- *Extract/Unload*—To accomplish the fastest large data extracts, remove all GROUP BY, ORDER BY, and DISTINCT clauses from the SQL SELECT statement that unloads the data, and then perform those operations with a special-purpose sort tool on the unloaded file. Keep in mind that the sort tool's SUMMARIZE processing is a much faster equivalent to the GROUP BY and DISTINCT clauses, while the sort tool's KEY option is a faster equivalent to the ORDER BY.

- *Reorg*—Reorg processing that consists of an unload, sort, and reload sequence will run significantly faster when these methods are used for the unload and reload and a special-purpose sort tool is used for the sort.

Summary

With all the key decisions that need to be made on a daily basis, the last thing a company needs is to wait for data to be processed. There are numerous ways that you can cut the elapsed time to run a variety of different applications. As Bill Inmon suggests, one way is to tune the data warehouse. This is accomplished by performing such tasks as adding indexes, creating data marts, building summary tables, and using delta processing.

Another way to improve query response time in a data warehouse is to build aggregates. Tables of aggregates significantly speed up queries by eliminating the time and resources spent summarizing data for each and every query. Improving database and data warehouse by pre-processing prior to load will also provide a boost to data intensive applications.

Business intelligence and Web applications are usually data intensive. Reducing the elapsed time to run a BI application or process clickstream records from a Web site will enable you to analyze and uncover key trends that will help your company make the right decisions. After all, time is money.

REFERENCES

Inmon, William H. **Enhancing Data Warehouse Performance**, BILLINMON.COM, 2001.

Kimball, Ralph, Laura Reeves, Margy Ross, and Warren Thornthwaite. **The Data Warehouse Lifecycle Toolkit: Expert Methods for Designing, Developing, and Deploying Data Warehouses**, John Wiley & Sons, 1998. ■



**50 Tice Boulevard
Woodcliff Lake, New Jersey 07677
(201) 930-8200 Phone
www.syncsort.com**