IOs and data warehouse directors are under pressure and overwhelmed with user requests for more resources, applications and power—often without an accompanying increase in budget. Fortunately, it’s possible to relieve some of the pressure and “find money” by effectively tuning applications on the data warehouse.

Sometimes queries that perform unnecessary full-table scans or other operations that consume too many system resources are submitted to the data warehouse. Application tuning is a process to identify and tune target applications for performance improvements and proactively prevent application performance problems.

Application tuning focuses on returning capacity to a system by concentrating on query optimization. Through application tuning, database administrators (DBAs) look for queries wreaking havoc on the system and then target and optimize those queries to improve system performance and prevent application performance problems. The results can be dramatic, often providing a gain of several nodes’ worth of processing power.

**Savings in the works**
A holistic view of Teradata performance—gained through the timely collection of data—is a good precursor to application tuning. Many customers have engaged Teradata Professional Services to install the performance data collection and reporting (PDCR) database.

This historical performance database and report toolkit provides diagnostic reports and graphs to help tune applications, monitor performance, manage capacity and operate the Teradata system at peak efficiency. If the PDCR database is not installed for performance tuning, it is imperative to enable Database Query Log (DBQL) detail, SQL and objects data logging for a timeframe that best represents the system workload to identify the optimum queries for tuning.

To optimize performance and extract more value from your Teradata system, follow these application tuning steps:

**STEP 1: Identify performance-tuning opportunities**
Some data warehouses handle millions of queries in a day. This makes it difficult for DBAs to identify suspect queries. A suspect query is one that either consumes too many system resources or is not taking advantage
of Teradata’s parallelism. While most DBAs are aware of the problem queries that most affect system performance, there are several ways to help prioritize what to tackle first. One way is to analyze ResUsage Data looking for the days or times of the day when the system is running close to or at 100% busy. AmpUsage data can be used to identify a particularly consumptive application or group of users. When it gets down to the real tuning analysis, though, DBQL data is the place to go.

The DBQL logs historical data about queries including query duration, CPU consumption and other performance metrics. It also offers information to calculate suspect query indicators such as large-table scans, skewing (when the Teradata system is not using all the AMPs in parallel) and large-table-to-large-table product joins (a highly consumptive join).

STEP 2: Find and record “like queries” with similar problems
While the DBQL is used to find specific incidents of problem queries, it can also be used to examine the frequency of a problem query. In this scenario, a DBA might notice that a marketing manager runs a problem query every Monday morning, and the same problem query is run several times a day by various users. Identifying and documenting the frequency of problem queries offers a more comprehensive view of the queries affecting data warehouse performance and helps prioritize tuning efforts.

STEP 3: Determine a tuning solution
Performance analysis should be completed by an experienced Teradata DBA with a fundamental understanding of Teradata performance. Identifying problem queries and recording instances of like queries is easy; the difficulty is analyzing and tuning a specific query. This analysis and tuning takes time and attention to detail, and it requires experience with the Teradata system and knowledge of Teradata features, including:

- EXPLAIN. Analyzes a problem query and shows step-by-step text for how Teradata’s Optimizer executes a query
- Visual Explain Tool. Shows the same EXPLAIN information but in graphics

When a DBA collects statistics depends on how frequently data is loaded in the data warehouse. Collecting statistics ensures the Optimizer has the most accurate and timely information about the data. Teradata Statistics Wizard analyzes specified workloads or databases and creates a recommended list of statistics collection operations. The collection of missing statistics can help the Optimizer choose the best query plan.

DBQL Step data can help a DBA drill deeper into a problem query by showing which query step is causing the problem skewing, product joining or large scan. This data, when matched up with the EXPLAIN plan, can provide the insight needed to tune the query.

To improve query performance—particularly queries with large-scan indicators—additional indexes or index changes should be considered. Teradata’s various indexing options enable efficient resource use, saving I/O and CPU time and thereby making more resources available for other work. Options such as partitioned primary index (PPI), secondary indexes and join indexes can help reduce resource consumption and make queries more efficient.

Teradata’s Index Wizard can recommend where to add secondary indexes and PPIs in tables based on a particular workload. DBQL Object Data can be used to determine and remove unused secondary indexes, which will reduce maintenance overhead when loading data and help reclaim storage space.

STEP 4: Determine the best solution
To determine the best tuning options, it is important to baseline existing performance conditions (using DBQL data), pilot potential solutions through experimentation and analyze the results. If multiple optimization strategies are found, DBAs should test one strategy at a time by temporarily creating the new scenario, changing the queries to use the new objects, running the queries, and measuring, documenting and analyzing the results.

DBAs must run tests on the same production system and take the following steps to determine the solution with the best cost/benefit and viability of the final performance fixes:

- Test the system, using a user ID with a low workload priority.
- Use each of the optimization strategies and gather the new DBQL data.
> Compare the new DBQL measurements with the original baseline.

**STEP 5: Regression testing**

Regression testing is an important quality control process to ensure any optimization changes or fixes will not adversely affect data warehouse performance. First, the DBA must determine a representative list of queries that apply to the selected performance fix. From there, a regression test suite is created to gauge the effectiveness of the solution before production.

In regression testing, the new environment is re-created on the same production system, and the effects of the change are measured and documented. The goal is to ensure queries that are not part of the tuning process are not unduly affected by the optimization changes.

To illustrate, suppose the DBA wants to make a primary index (PI) change, a modification the DBA thinks will improve system performance. Regression testing shows that this particular PI change causes another application to perform full-table scans for its most common queries. In this case, changing the PI may not be the optimal choice, and other tuning options should be considered.

**STEP 6: Quantify and translate performance gains into business value**

CIOs are routinely pressed to show how their IT dollars affect operations and enable cost reduction and business growth. Quantifying the business value of query optimization, or any IT improvement, is an important step to showcasing the value of the data warehouse.

Determining business value can be broken into calculations and sub-calculations. To answer the question “How many CPU seconds equals a node?” use the following calculations:

> Determine **per node CPU seconds in a day** (number of CPUs per node X 86,400) - 20%, where 86,400 equals the number of seconds in a day, and 15% to 20% is subtracted from the equation to account for system-level work not recorded in DBQL.

> Multiply **per node CPU seconds in a day** by 30 to get **CPU seconds per node per month**. On a four-CPU node, the equation would look something like this:  

\[(4 \times 86,400) - (4 \times 86,400)/5) \times 30 = 8,294,400 \text{ CPU seconds.}\]

> Check the impact of making a tuning change: **Monthly CPU saved** = Total old CPU for a month X the average improvement percent.

To illustrate calculations to life, imagine the DBA makes an optimization change where regression testing shows an overall 50% improvement in query CPU consumption. Queries that previously averaged 8 million CPU seconds a month can now average 4 million CPU seconds a month. This equates to half a node of processing power savings.

Taking this a step further, if the targeted queries run only during normal business hours (a 12-hour day), the performance improvement would be even greater.

**STEP 7: Document and implement**

Presenting application tuning recommendations to IT management and business users typically requires more than a spreadsheet of data, although a spreadsheet can be used for backup material or a deeper dive into performance data and options. The presentation should be tailored to a specific audience and should capture the value of application tuning. The presentation might include:

- Query optimization process
- Options found and tested
- Best option
- Options discarded, and why
- Lists of what still needs testing
- Observations and recommendations
- Anticipated savings

**Money talks**

Application tuning provides DBAs the opportunity to “find money” by returning processing power to a Teradata system. With application tuning, the system can handle new users and applications or latent demand. Workload management, capacity planning and other techniques are important to help keep the data warehouse running at peak performance; however, only application tuning can return CPU processing power to the Teradata system.

Customers looking to add new applications, improve application performance or quantify the need for hardware expansion can benefit from application tuning. Following the application tuning methodology will help you optimize performance and gain more value from your Teradata system.

Donna Becker has been with Teradata since 1993 and is director of America’s Professional Services Performance and Workload Management Center of Expertise.

Paul A. Barsch has more than 12 years of IT experience and manages Teradata Professional Services marketing programs.