



Managing SQL Server Performance with Embarcadero Performance Analyst

By Robin Schumacher
VP Product Management, Embarcadero Technologies, Inc.

April 15, 2004

Corporate Headquarters
Embarcadero Technologies
425 Market St., Suite 425
San Francisco, CA 94105
www.embarcadero.com

EMEA Headquarters
Embarcadero Technologies, Ltd.
Thames House
17 Marlow Road
Maidenhead
Berkshire
SL6 7AA
United Kingdom
www.embarcadero.co.uk

Table of Contents

Managing SQL Server Performance with Embarcadero Performance Analyst	1
Introduction	3
Performance Analysis Techniques	3
Ratio and Statistical-Based Analysis	4
Deficiencies of Only Using a Bottleneck or Workload Approach	5
Accurate Interpretations	5
Bottleneck Analysis	6
How to Find SQL Server Bottlenecks	6
Reviewing Wait Activity	7
Examining I/O Efficiency	7
Checking I/O Hotspots	8
Viewing Resource Utilization	9
Locating SQL Server Showstoppers	9
Workload Analysis	10
Session Resource Consumption	10
SQL Performance Metrics	10
Conclusion	12
About Embarcadero Technologies	12
About the Embarcadero Analyst Product Line	12

Introduction

Nearly every database vendor is talking about how “self-managing” their database is these days. Not surprisingly, such claims are directed squarely at corporate management who have long since regarded databases as complicated black boxes that require management by trained and expensive personnel. The new message being sent to IT managers is that all complexity has been removed from the database and that it, like a new network card being introduced into a PC, has become plug-and-play.

If you examine the self-management claims from the database giants, you will find that their main thrust revolves around the concept of a database recognizing inefficiencies in its performance and automatically taking corrective action to alleviate any perceived bottleneck. To a lesser degree you will find attempted simplification in the areas of storage and configuration management where less work is supposedly required from the database administrator to set up and revisit things like memory and space allocation.

Microsoft SQL Server has for many years attempted to auto-manage a lot of the administrative and performance characteristics of its database engine. Although some may disagree, SQL Server’s ease-of-use accolades in the areas of setup and basic administration are well deserved, but what about automatic performance management? Does SQL Server competently handle performance in a way that requires no DBA tuning or intervention? Perhaps a coy way to answer this question is by the following observation: If you visit one of the more popular SQL Server forums on the Internet (SQL Server Magazine’s), you will find over 2,000 posts in the performance question category of the site – more posts than in any category other than general administration. So the answer to the previous question is another question: if SQL Server performance analysis has been made obsolete by self-managed performance, then why so many questions on the topic?

Clearly, performance analysis is still required for SQL Server (and every other database engine). That being the case, how does one best approach the practical application of performance analysis? What techniques are needed to quickly find and remove performance inefficiencies that exist in a SQL Server database? The rest of this paper is devoted to answering these important questions. In addition, the paper delivers a performance optimization roadmap that you can immediately put into play to effectively manage the performance of all your critical SQL Server systems.

Performance Analysis Techniques

Database performance analysis can be pursued in many different ways, and it seems that every database professional has their own preferred method. As an analogy, the concepts of analyzing a database can be likened somewhat to investment/stock analysis. There are many techniques that investors use to choose stocks for their portfolios, but most can be boiled down to two basic methods: fundamental and technical analysis.

Those who follow fundamental analysis look for things like continuous increases in a company’s earnings per share, sales and revenue growth rates, profit margins, and other key factors that typically indicate a company’s stock may be ready to rise. Proponents of technical analysis turn

their noses up at fundamentalists and insist that the way to pick winning stocks is by examining chart patterns of a company's stock, along with other market-leading indicators that can signal when to buy or sell.

Even though both techniques have their cheerleaders, there are some investment professionals (most of them pretty good), who, instead of limiting themselves to one method, embrace both. The bursting of the tech bubble in the early 2000's taught technical enthusiasts one thing: a company's fundamentals and bottom line do matter. And fundamentalists learned that even a stock with outstanding corporate sales and revenue acceleration could be dragged down when its peers in the same industry group head south.

Like stock analysts, database performance analysts typically use one of three methods for examining the performance levels of a database server:

1. **Ratio and Statistical-based Analysis** – This involves examining a number of key database ratios and important resource metrics that hopefully can be used to indicate how well a database is running.
2. **Bottleneck Analysis** – This seems to be more in vogue today, with many experts on database performance deriding those who still dare to practice any ratio-based analysis. Instead of using ratios, this methodology focuses on finding the things that cause the database to wait, and removing them where possible.
3. **Workload Analysis** – This involves the investigation of two critical areas of a database's performance: session resource consumption and SQL execution metrics.

Rather than just focus on one of the above methods, could it be that all three analytical methods are needed to accurately diagnose a database's performance level? Just like smart investors who use both fundamental and technical analysis to make money in the stock market, should the smart database administrator use a combination of ratio/statistical-based, bottleneck, and workload analysis to determine if a database is performing well?

The answer of course is Yes. Let's see how you can accurately combine all three techniques to produce a single, winning game plan that can be used to help you troubleshoot SQL Server performance problems. You will also see how Embarcadero Performance Analyst is the only database monitor currently available that accurately uses all three methodologies to present a complete picture of a database server's performance.

Ratio and Statistical-Based Analysis

Ratio and statistical-based analysis has been around for many years, and used to be the only technique database administrators utilized when they were called upon to diagnose the cause of a database slowdown. Using one or more prized SQL scripts, the DBA would conduct an examination of key ratios and performance metrics such as database memory use, CPU usage, contention, I/O, network utilization, and others, to hopefully pinpoint the areas responsible for unacceptable performance. With SQL Server 7.0 and above, some administrators would monitor SQL Server via the Windows Performance Monitor and track the aforementioned metrics.

Regardless of the collection method, once a particular area was identified as a potential problem, the standard remedy was to increase the resources for that area (like memory), which sometimes did the trick, but oftentimes failed to produce any real performance gain.

Why did such analysis fail to produce a winning solution? Many times, it had to do with the fact that the areas perceived as the problem weren't the real culprit. Quick and easy ratios and rules of thumb can be misleading and must be evaluated in light of other analytic findings. Does this mean that this form of analysis doesn't provide value? Not at all. When done right, ratio and statistical-based analysis definitely has a place in the DBA's performance-tuning arsenal, but to be effective, it must be practiced right. Performance ratios and standalone statistical measures are very good roll-up mechanisms for busy DBAs, making possible the analysis-at-a-glance approach. Succinct performance metrics can assist in such situations by giving DBAs a few solid indicators that can be scanned quickly to see if any database needs immediate attention.

Performance Analyst presents all key server ratios and statistical metrics for DBA's who use this form of performance analysis for part or their entire database monitoring routine. All ratios and statistics display a currently computed value as well as a line chart that contains a history of each metric over time.

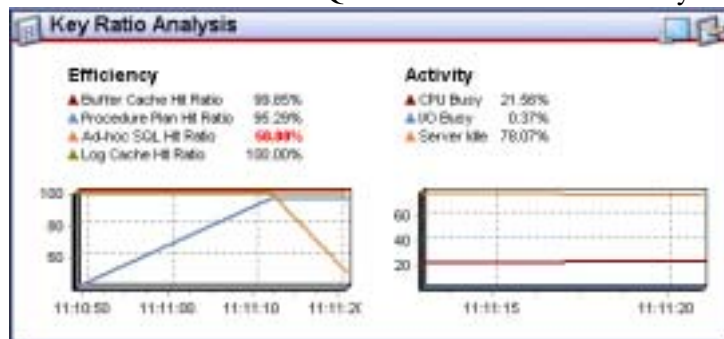
Deficiencies of Only Using a Bottleneck or Workload Approach

Ratio and statistical-based analysis is also still viable because a pure wait-based approach will miss a number of critical items that indicate poor performance. For example, the wait events in SQL Server do not do a very good job of identifying SQL and procedure reusage, but the procedure and ad-hoc SQL ratios do.

Accurate Interpretations

So, how does one accurately perform ratio and statistical-based analysis? While there are certainly many opinions as to what rules to follow, some standards should always be adhered to. To begin with, many of the formulas that make up performance ratios and statistics must be derived from delta measurements instead of cumulative numbers. Many of the SQL Server performance counters are cumulative in nature – in other words, they continue to increment from the moment SQL Server is started. For servers that are kept up for long periods of time, these values can grow quite large and will impact how a particular ratio or metric is interpreted.

Embarcadero Performance Analyst uses delta statistics where SQL Server doesn't for each key ratio and statistic where deltas are appropriate. Spotting delta-based statistics in Performance Analyst is easy because a triangle in the legend identifies a delta statistic, whereas cumulative statistics are marked with the square legend identifier.



In addition to using delta statistics to compute many of the key metrics in ratio-based performance analysis, DBAs must also examine key metrics across databases to see what databases are receiving the most attention from the user community.

Performance Analyst presents many of the key performance metrics in just this way on its Database Home Page.

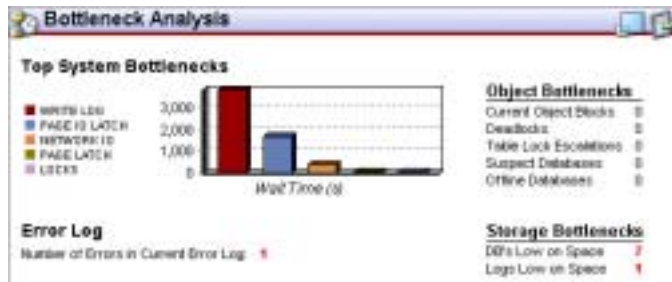
A final thing to remember about using ratio and statistical-based analysis is that, while there are several rules of thumb that can be used as starting points in the evaluation of database performance, each database has an individual personality. Some hard and fast rules simply will not apply to every database. Not all servers require high ratios to satisfy an end user community.

Bottleneck Analysis

When a SQL Server is up and running, every connected process is either busy doing work or waiting to perform work. A process that is waiting may mean nothing in the overall scheme of things, or it can be an indicator that a database bottleneck exists. And, this is where wait-based or bottleneck analysis comes into play. DBAs use this form of performance analysis to determine if perceived bottlenecks in a database are contributing to a performance problem.

Bottleneck analysis is a valid method of measuring performance because it helps a DBA track where a database server has been spending its time. If latch or lock contention has been dragging a database's performance down, a DBA can use bottleneck analysis to confirm the actual root cause. Once one or more bottlenecks have been pinpointed as possible performance vampires, the DBA can drill down and oftentimes discover a fair amount of detail about the sessions and objects that are causing the problem.

Bottleneck analysis is also the best technique to use for catching serious headaches that can stop a database entirely in its tracks (like a full transaction log).



Performance Analyst presents the bottleneck analysis methodology throughout every home page and also provides drill down details into every wait event-based and non-wait event-based bottleneck so the DBA won't miss any potential threat to their

database's performance.

How to Find SQL Server Bottlenecks

When SQL Server begins to exhibit less than ideal performance, where do you begin the search for the possible causes of response time degradation? The areas to begin your investigation include:

SQL Server Waits – examining wait events will tell you where the SQL Server engine has been held up, and therefore can lead you to areas that may be contributing to reduced response times. I/O Activity – the amount, type, and location of I/O activity can greatly contribute to a reduced performance level in SQL Server.

Resource Utilization – how much CPU has SQL Server been using and has the server’s CPU been constantly running at 100%? When you find answers to these questions, you may uncover the bottlenecks that are slowing down the overall performance of your SQL Server.

Reviewing Wait Activity

Many DBAs don’t know that SQL Server records the events that cause the engine to wait. Such waits can include delays for physical transaction log writes, waits for object locks, latching issues, and more. Waits can be examined at the system level as well as the session level, so you can get both a global perspective on what has been causing SQL Server to wait and also uncover what currently connected sessions have been or are waiting on.

Performance Analyst makes it very easy to review SQL Server wait events. Many of the home pages contain a summary of the top wait events for either the entire server or for a particular area (like top I/O wait events). Performance Analyst also contains powerful drill downs into system wait events so you can see the percentage of time that SQL Server has been waiting on for each recorded wait event. From a session perspective, you can also get an instant picture of what sessions are in a wait state (and what they are waiting on), what sessions have experienced the most delays, and what those delays were. If you are unfamiliar with the various SQL Server wait events, you can review the following Microsoft Knowledge Base Article: <http://support.microsoft.com/default.aspx?scid=kb:en-us:Q244455>.

Examining I/O Efficiency

When it comes to investigating I/O activity, you want to concern yourself with four things:

1. What amount of I/O is SQL Server experiencing?
2. What type of I/O occurs the most (physical or logical)?
3. Where are the I/O hotspots?
4. What sessions and SQL are responsible for the most I/O activity?

The first three fall into the bottleneck analysis area while the last falls into workload analysis.

To uncover the amounts of I/O SQL Server is encountering, you can review various I/O metrics such as page reads and writes, read ahead pages, DBCC activity, and space-related I/O activity such as extents allocated, page splits, bulk copy rows, bulk copy throughput, and more. As with statistical and ratio analysis, the best way to view these statistics is in delta fashion, rather than just looking at cumulative numbers. Cumulative metrics are OK to help you get your bearing, but delta measurements help you see what is currently happening on the server in a more clear fashion.

Physical I/O takes much longer to complete than logical I/O, so it’s helpful to determine whether SQL Server is experiencing heavy volumes of physical I/O. This can be understood by reviewing the buffer cache hit ratio along with the page reads statistic. Page reads are to be expected, especially after initial server start up. This is because SQL Server must first satisfy requests for data and meta-data by reading information in from physical disk. But numerous page reads can also be expected if the physical server does not contain an adequate amount of memory to hold often-requested blocks of information.

No hard and fast rules exist for how many page reads is too much. You can cross reference this statistic with the physical server disk statistics to see if physical page reads and accompanying physical disk I/O are approaching the server's premium capacity levels. And, as has already been mentioned, logical I/O is always many times faster than physical I/O so you should also evaluate the buffer cache hit ratio to determine overall memory vs. physical read efficiency.

If you find that the server is becoming overworked from a physical I/O standpoint, there are several courses of action you can take:

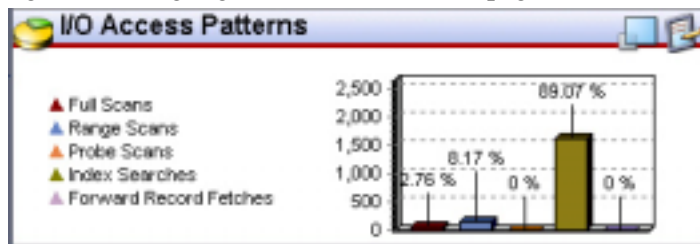
- Examine index usage to ensure that unnecessary table scans are not occurring
- Check the physical database design to see if table objects have been over-normalized.
- Ensure that SQL Server is configured to use sufficient amounts of memory. Examine the min server memory (MB) and max server memory (MB) parameters to see if SQL Server is constrained on either end of the memory spectrum.
- Check for large pinned table objects that could be using excessive amounts of space in the buffer cache. Conversely, ensure that small, frequently referenced lookup tables are indeed pinned in the cache.
- Last, but not least, investigate the possibility of adding more RAM to the physical server.

Keep in mind that if physical I/O is kept to a minimum, your goal with I/O is to eliminate *all* excessive I/O including logical I/O. It is not uncommon to see SQL Servers with excellent cache hit ratios, but still be I/O bound due to large amounts of logical I/O.

Checking I/O Hotspots

Another thing to examine with respect to I/O is *where* the I/O is occurring. DBAs sometimes do not smartly spread and stripe SQL Server files across their servers with the end result being I/O hotspots where one drive is terribly overworked while others remain almost completely idle. Another mistake is relegating write-intensive databases to RAID5 storage devices, which impose heavy write penalties because of their parity/mirroring mechanisms. In SQL Server 2000 and above, you can examine I/O patterns across databases and files to see what drives and files are receiving more than their fair share of attention.

In terms of monitoring SQL Server I/O activity, Performance Analyst provides everything you need to quickly determine if your databases are experiencing I/O-related bottlenecks. All physical and logical I/O counters are presented in delta fashion and are tracked historically and graphically. The hottest database and log files are highlighted on the I/O home page and powerful drill downs are available so you can see every detail with respect to I/O measurements at the system, user, database, and file level. Finally, I/O access patterns are tracked so you instantly know if full table scans or other prohibited I/O activities are occurring in any of your databases (a key piece of information that helps you understand if your SQL Server is experiencing excessive logical I/O).



You can also examine I/O activity at the server level through Performance Analyst to determine if disk queue problems exist, excessive read and write times are being experienced, or other bottlenecks are present on the actual server.

Viewing Resource Utilization

All the physical design and administration work of a DBA can be undone if SQL Server is placed on a server with less than adequate horsepower. SQL Server can be quite demanding on a server if its user community is large and active, so you need to keep an eye on memory and CPU utilization at the SQL Server and hardware level.

Performance Analyst makes it easy to watch both areas of CPU and memory utilization. SQL Server CPU and memory utilizations are displayed through Performance Analyst so you can immediately know how much CPU and memory SQL Server is using and if any limits are being reached. CPU and memory can be monitored at the hardware level also so you can quickly determine if the machine's CPU is constantly pegged at 100% or if the machine is experiencing memory exhaustion or paging problems. You can also view individual server processes to see which processes are using the most CPU, memory, etc., at both the SQL Server and machine level.

Locating SQL Server Showstoppers

When using bottleneck analysis, a DBA cannot rely only on wait events, I/O information, and resource measures. This is because such metrics won't catch even deadlier bottlenecks that may be present inside one or more of your databases. These types of blockages have the potential to cause everything to immediately stop. For example, a database's transaction log can run out of free space due to a large runaway transaction or repeated heavy DML activity. Such a failure will not be reflected in any wait event, but still represents a very real bottleneck to the database it supports. In the same way that a DBA cannot depend on only a few ratios to properly carry out ratio and statistical-based performance analysis; an administrator must include several statistical metrics in their overall bottleneck analysis framework to obtain an accurate performance-risk assessment.

For example, in the aforementioned transaction log failure, the DBA would want to include a query in their bottleneck analysis framework that returns a count of all transaction logs that have reached (or better yet, are approaching) their free space limits. Even with SQL Server's auto-growth mechanisms implemented for a transaction log, this information is good to have since further growth may be denied due to file size limits imposed at the SQL Server level or free space deficits encountered at the server level. Other showstopper items to consider are serious cases of lock contention and space limits being reached at the database or server level.

Performance Analyst presents all these showstopper-styled bottlenecks throughout every home page. The DBA can easily see these bottlenecks and take corrective action to avoid downtime or performance delays. For example, the main home page of Performance Analyst conveys a count of all database and transaction logs that are about to run out of free space. It also indicates if lock contention is occurring, and informs you if any database has gone offline or has been flagged as suspect in nature.

Workload Analysis

Key ratios and statistics help a DBA get a global perspective on database activity, and bottleneck analysis assists the DBA by offering insight into things that are holding up user activity and throughput, as well as things that threaten to throw the curtain down on SQL Server's operations. But another technique is necessary if a database professional is to really get a handle on what's occurring inside a badly performing SQL Server system.

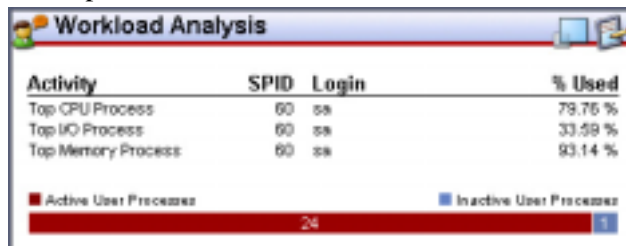
Workload analysis involves the investigation of two critical areas of a database's performance:

Session resource consumption and activity
SQL execution

Without looking at these two key performance categories, a DBA will miss most of what could be responsible for perceived performance problems.

Session Resource Consumption

When performance on a database takes a sudden nosedive, it is not uncommon to find one or two



sessions that are causing the bulk of the workload. Locating such sessions can easily be accomplished by viewing session metadata coupled with resource consumption and statistical execution statistics for each session.

Performance Analyst identifies top SQL

Server resource consumers in several different ways. On the main home page, Performance Analyst highlights the top resource sessions across I/O, memory usage, and CPU consumption. The percentage used across all statistical categories are displayed so a DBA can immediately pinpoint a session that is using all or most of a particular resource. The top resource sessions are also displayed on the memory, I/O, and other home pages as well with a listing of the top sessions for that category (for example, the top memory users appear on the memory home page, etc.)

The Top Sessions drill down view in Performance Analyst highlights the top resource-hungry sessions with many more details regarding a particular session's activity. With Performance Analyst, it takes only a couple of mouse clicks to quickly pinpoint sessions that are contributing to a particular database or server-wide slowdown.

SQL Performance Metrics

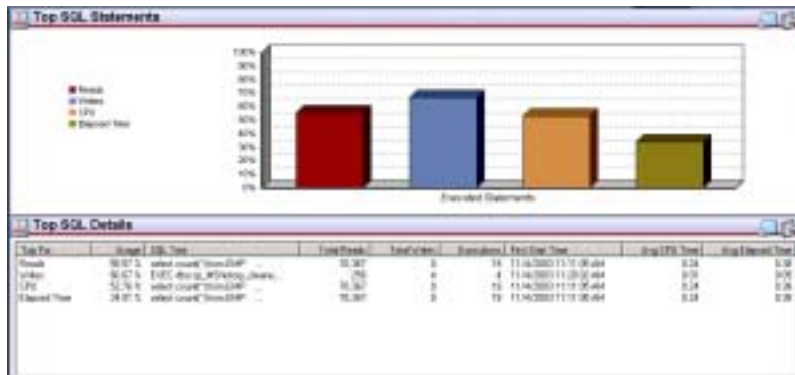
The floodgates of throughput and performance are oftentimes opened wide when a DBA pinpoints and corrects resource-intensive SQL that is running on a SQL Server. You will likely find that optimizing SQL code, or creating new indexes that existing SQL code can use, will produce some of the best performance boosts available for a server. Understanding current and historical SQL execution patterns will enable you to have the second (and key) set of data

necessary to properly perform workload analysis. The end goal here is, of course, to reduce end use response times.

To obtain SQL execution metrics, many SQL Server administrators use Microsoft's SQL Profiler tool to uncover untuned SQL statements or resource-expensive procedures. SQL Profiler is a terrific tool to use in locating all SQL statements that are fired at a SQL Server. You can also include filters that can help you hopefully obtain only the problematic SQL statements that need your attention.

One of the few drawbacks of SQL Profiler is that it doesn't roll up execution metrics for a particular SQL statement. For example, if SQL Statement A is issued 1,000 times (which occurs often in a packaged system or application where SQL code is embedded), your SQL Profiler trace will likely contain 1,000 instances of Statement A's execution along with each instance's performance metrics (I/O, etc.) A better way to understand if Statement A is causing a problem is to roll up all execution metrics so you can see that Statement A has been executed 1,000 times, has caused over 300,000 physical reads, with an average physical read per execution count of 300, and has an average elapsed execution time of 2 seconds. Oracle has these kinds of SQL performance metrics in its database, and such metadata has been invaluable for DBAs, on that platform, who have to troubleshoot SQL performance.

Performance Analyst offers plenty of insight into SQL performance and execution patterns and conveys SQL performance metrics in the exact ways you need to see them. The collection of SQL statements and stored procedure executions is optional in Performance Analyst, but is easy to turn on. Once Performance Analyst begins collecting SQL statements and procedure runs, it aggregates SQL statement and procedure performance metrics so you can instantly see the statements and



procedures that will benefit from tuning. On the main home page, Performance Analyst shows the SQL Statements that are responsible for the most I/O, CPU usage, and more. You can, for example, see that a single SQL statement is responsible for almost 50% of all the physical I/O on the server. Drill-down details are available so a DBA can see the full SQL syntax of resource hungry SQL statements as well as detailed performance statistics.

Keep in mind that while locating the SQL calls that are racking up expensive statistics is easy to do in Performance Analyst, the actual art of optimizing SQL can be quite difficult. While there are third party software products that assist in rewriting SQL queries, you won't find one that will tell you that a particular SQL statement should not be executed. Only by understanding a particular application's needs can this be accomplished. Removing SQL 'waste' in a database can produce dramatic results as, believe it or not, sometimes the best SQL query is the one you don't execute.

Conclusion

Even in the age of supposed “self-managing” databases, the DBA still needs to carry out performance analysis and tuning. Using a singular diagnostic approach to database performance analysis may, under the right conditions, result in pinpointing a database slowdown, but a more complete approach to performance analysis is to utilize metrics and collection methods from ratio/statistical-based analysis, bottleneck analysis, and workload analysis. As with investing in the stock market, a person might not be right every time, but they can limit most losses by staying true to a combined and proven methodology.

The holistic approach used by Performance Analyst allows a DBA to use any or all of the analysis styles that they prefer. With Performance Analyst, a DBA can easily combine and utilize all three performance methodologies to ensure that their critical SQL Server systems stay up and running at better than expected performance levels.

About Embarcadero Technologies

Embarcadero Technologies’ software products enable companies to build, optimize and manage their critical business applications and underlying databases. Embarcadero leads the market in providing high quality, easy-to-use database and application development tools that deliver cost-effective solutions.

About the Embarcadero Analyst Product Line

Embarcadero Technologies’ **Analyst** products provide advanced management and performance diagnostic capabilities for today’s database professional to use in managing complex database environments. Powerful enough for advanced experts, yet easy to use for database novices, all **Analyst** products from Embarcadero contain built-in intelligence that ease the process of troubleshooting and resolving problems that threaten the availability and performance of mission-critical databases.

Embarcadero, the Embarcadero Technologies logos and all other Embarcadero Technologies product or service names are trademarks of Embarcadero Technologies, Inc. All other trademarks are property of their respective owners.