Benchmarking Performance of a Query

MS SQL

Posted Feb 4, 2004

Benchmarking Performance of a Query - Part 2 CPU and I/O

By Gregory A. Larsen

This is the second article in a series discussing benchmarking T-SQL query performance. The first article in this series discussed different methods of capturing the elapsed time of a T-SQL batch, pieces of a batch, or a single T-SQL statement. This article will discuss how to determine the amount of CPU and I/O used when a given T-SQL statement or a series of statements are executed.

When you are tuning your query, you need a way to measure whether you are making improvements. You can use CPU and I/O as a measurement tool to determine if a particular query is faster than another is, or a particular database design is better than another one. In this article, I will show you how to measure the CPU and I/O so you can determine whether you are making improvements when tuning your queries.

Measuring CPU

There are a number of different methods to determine how much CPU is consumed by your query. I am going to show you two different methods you can use from within Query Analyzer.

The first method, "SET STATISTICS TIME ON," to show CPU usage, was briefly discussed in my first article. Using this method is useful in getting the CPU for a single statement, but when you process millions of commands, you also get millions of lines of CPU time statistics. If you turn on the statistics gathering process, remember you will need to issue the "SET STATISTICS TIME OFF" to turn off the statistics gathering processes. Since this method produces lots of output when executing T-SQL batches, I use this method only when I have a single T-SQL statement I am interested in measuring CPU.

The other method is to use the @@CPU_BUSY system variable to calculate the CPU resources consumed. Since the @@CPU_BUSY is a counter that contains the number of milliseconds that SQL Sever has used since it was started, I only use this method on a stand-alone machine, like a laptop, or desktop machine. If you use @@CPU_BUSY on a multi-user machine, then the @@CPU_BUSY variable will reflect CPU used by all users, not just the T-SQL query you are trying to benchmark. Here is an example of some code that calculates the amount of CPU used to process two different methods of padding a number with leading zeroes. By using this example, you can determine which method uses the least amount of CPU.

```
DECLARE @I INT
DECLARE @C CHAR(8)
DECLARE @CPU_START int
DECLARE @X INT
SET @C = ''
SET @I = 123
SET @X = 0
SET @CPU_START = @@CPU_BUSY
WHILE @X < 1000000
```

Benchmarking Performance of a Query

MS SQL

Posted Feb 4, 2004

Benchmarking Performance of a Query - Part 2 CPU and I/O

By Gregory A. Larsen

This is the second article in a series discussing benchmarking T-SQL query performance. The first article in this series discussed different methods of capturing the elapsed time of a T-SQL batch, pieces of a batch, or a single T-SQL statement. This article will discuss how to determine the amount of CPU and I/O used when a given T-SQL statement or a series of statements are executed.

When you are tuning your query, you need a way to measure whether you are making improvements. You can use CPU and I/O as a measurement tool to determine if a particular query is faster than another is, or a particular database design is better than another one. In this article, I will show you how to measure the CPU and I/O so you can determine whether you are making improvements when tuning your queries.

Measuring CPU

There are a number of different methods to determine how much CPU is consumed by your query. I am going to show you two different methods you can use from within Query Analyzer.

The first method, "SET STATISTICS TIME ON," to show CPU usage, was briefly discussed in my first article. Using this method is useful in getting the CPU for a single statement, but when you process millions of commands, you also get millions of lines of CPU time statistics. If you turn on the statistics gathering process, remember you will need to issue the "SET STATISTICS TIME OFF" to turn off the statistics gathering processes. Since this method produces lots of output when executing T-SQL batches, I use this method only when I have a single T-SQL statement I am interested in measuring CPU.

The other method is to use the @@CPU_BUSY system variable to calculate the CPU resources consumed. Since the @@CPU_BUSY is a counter that contains the number of milliseconds that SQL Sever has used since it was started, I only use this method on a stand-alone machine, like a laptop, or desktop machine. If you use @@CPU_BUSY on a multi-user machine, then the @@CPU_BUSY variable will reflect CPU used by all users, not just the T-SQL query you are trying to benchmark. Here is an example of some code that calculates the amount of CPU used to process two different methods of padding a number with leading zeroes. By using this example, you can determine which method uses the least amount of CPU.

```
DECLARE @I INT
DECLARE @C CHAR(8)
DECLARE @CPU_START int
DECLARE @X INT
SET @C = ''
SET @I = 123
SET @X = 0
SET @CPU_START = @@CPU_BUSY
WHILE @X < 1000000
```

Benchmarking Performance of a Query

MS SQL

Posted Feb 4, 2004

Benchmarking Performance of a Query - Part 2 CPU and I/O

By Gregory A. Larsen

This is the second article in a series discussing benchmarking T-SQL query performance. The first article in this series discussed different methods of capturing the elapsed time of a T-SQL batch, pieces of a batch, or a single T-SQL statement. This article will discuss how to determine the amount of CPU and I/O used when a given T-SQL statement or a series of statements are executed.

When you are tuning your query, you need a way to measure whether you are making improvements. You can use CPU and I/O as a measurement tool to determine if a particular query is faster than another is, or a particular database design is better than another one. In this article, I will show you how to measure the CPU and I/O so you can determine whether you are making improvements when tuning your queries.

Measuring CPU

There are a number of different methods to determine how much CPU is consumed by your query. I am going to show you two different methods you can use from within Query Analyzer.

The first method, "SET STATISTICS TIME ON," to show CPU usage, was briefly discussed in my first article. Using this method is useful in getting the CPU for a single statement, but when you process millions of commands, you also get millions of lines of CPU time statistics. If you turn on the statistics gathering process, remember you will need to issue the "SET STATISTICS TIME OFF" to turn off the statistics gathering processes. Since this method produces lots of output when executing T-SQL batches, I use this method only when I have a single T-SQL statement I am interested in measuring CPU.

The other method is to use the @@CPU_BUSY system variable to calculate the CPU resources consumed. Since the @@CPU_BUSY is a counter that contains the number of milliseconds that SQL Sever has used since it was started, I only use this method on a stand-alone machine, like a laptop, or desktop machine. If you use @@CPU_BUSY on a multi-user machine, then the @@CPU_BUSY variable will reflect CPU used by all users, not just the T-SQL query you are trying to benchmark. Here is an example of some code that calculates the amount of CPU used to process two different methods of padding a number with leading zeroes. By using this example, you can determine which method uses the least amount of CPU.

```
DECLARE @I INT
DECLARE @C CHAR(8)
DECLARE @CPU_START int
DECLARE @X INT
SET @C = ''
SET @I = 123
SET @X = 0
SET @CPU_START = @@CPU_BUSY
WHILE @X < 1000000
```