Relatório de consultoria

*Consultoria em SQL Server – análise de ambiente e recomendações de desempenho relacionados a* ***desenvolvimento***

Servidor: <NomeServidor>

Bancos de dados: <bancos de dados>

Marília, Abril de 2014

**Fabiano Neves Amorim**

*Microsoft MVP - SQL Server*

E-mail: [fabiano.amorim@srnimbus.com.br](mailto:fabiano.amorim@srnimbus.com.br)

<http://blogfabiano.com/> - (14) 98111-0294

Sumário

[Resumo do serviço 2](#_Toc386468818)

[Considerações gerais 2](#_Toc386468819)

[TOP <N> consultas por uso de <CPU|I/O|Duration|Número de execuções> 3](#_Toc386468820)

[TOP 1 - <nome da proc ou alias (ex:, select 1)> 3](#_Toc386468821)

[TOP 2 - <nome da proc ou alias (ex:, select 1)> 5](#_Toc386468822)

# Resumo do serviço

<descrição com resumo do serviço de tuning>

Neste projeto a consultoria foi executada pelo profissional Fabiano Neves Amorim.

# Considerações gerais

<considerações gerais e particularidades que merecem destaque>

# TOP <N> consultas por uso de <CPU|I/O|Duration|Número de execuções>

## TOP 1 - <nome da proc ou alias (ex:, select 1)>

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| CPU antes | CPU depois | Page reads antes | Page reads depois | Duration antes | Duration depois |
| 672 | 0 | 8758 | 88 | 276ms | 8ms |

Exemplo de execução do código:

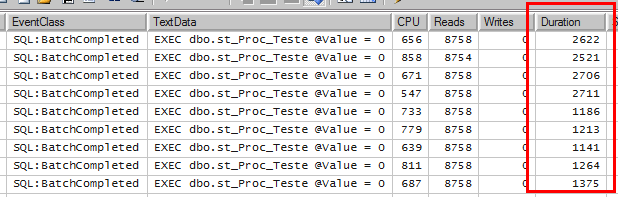
EXEC dbo.st\_Proc\_Teste @Value = 0

GO

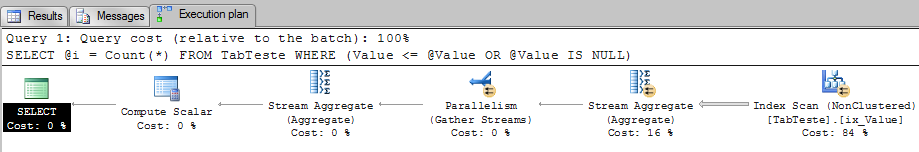
Performance atual:



Quando executamos o código simulando ***apenas 50*** usuários rodando o código o tempo de resposta sobe para média de 2 segundos.



Plano de execução:



Recomendações

Sempre que houver a possibilidade de um código processar uma quantidade de linhas muito diferentes dependendo dos parâmetros de entrada (ou filtros) informados, não podemos correr o risco de deixar o query processor (vide nota) executar a o código reutilizando um plano que possivelmente foi criado para processar poucas linhas. Neste caso existem algumas opções que devem ser consideradas, são elas:

* OPTION (RECOMPILE) – Esta é a opção mais simples de ser utilizada, porém irá gerar a recompilação (geração de um plano) do código todas as vezes que ela for utilizada.
* OPTION (OPTIMIZE FOR UNKNOWN) – Irá gerar um plano de execução baseado na densidade (média de valores distintos) da coluna que está sendo filtrada
* OPTION (OPTIMIZE FOR (@parametro = <algum valor>)) – Com esta opção é possível especificar um valor para garantir que um plano seja inicialmente criado com base neste valor, ignorando o valor passado como parâmetro de entrada.

Dependendo da quantidade de vezes que a o código é executado, recomendados utilizar a opção OPTION (RECOMPILE), porém testes devem ser realizados para analisar se o impacto de recompilar a consulta realmente não pode ser evitado.

Como a procedure em questão utiliza a atribuição de variável, o OPTION (RECOMPILE) tem uma limitação que não aplica a funcionalidade de parameter embedding optimization para obter o ganho do plano mais eficiente. Para resolver este problema, podemos jogar os dados em uma tabela temporária e depois atribuir o valor para a variável com base na temporária.

Abaixo um exemplo do código da procedure utilizando o OPTION (RECOMPILE) mais a tabela temporária:

CREATE PROCEDURE dbo.st\_Proc\_Teste @Value Int

AS

BEGIN

DECLARE @i Int

SELECT i = Count(\*)

INTO #TMP

FROM TabTeste

WHERE (Value <= @Value OR @Value IS NULL)

OPTION (RECOMPILE)

SELECT @i = i

FROM #TMP

SELECT @i "Resultado do Count"

END

Rodando a nova procedure temos o seguinte resultado:



## TOP 2 - <nome da proc ou alias (ex:, select 1)>

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| CPU antes | CPU depois | Page reads antes | Page reads depois | Duration antes | Duration depois |
| 5726 | 256 | 3791 | 3821 | 25 secs | 174ms |

Exemplo de execução do código:

SELECT \*

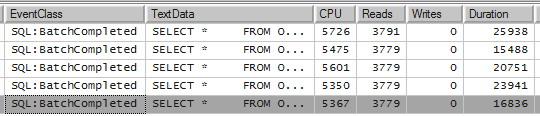
FROM OrdersBig

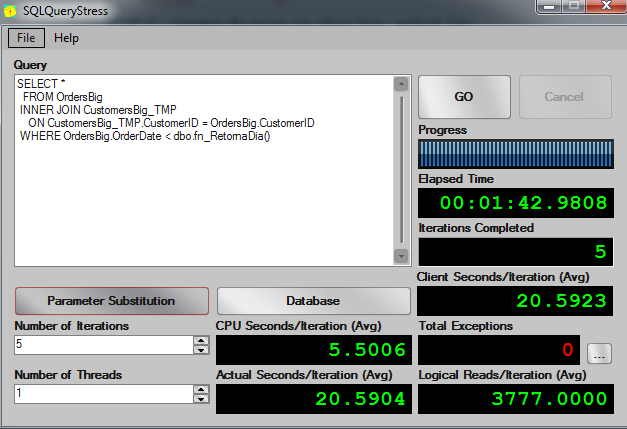
INNER JOIN CustomersBig\_TMP

ON CustomersBig\_TMP.CustomerID = OrdersBig.CustomerID

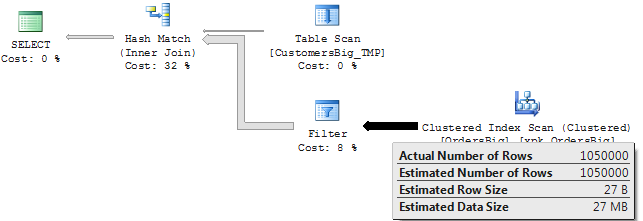
WHERE OrdersBig.OrderDate < dbo.fn\_RetornaDia()

Performance atual executando o código com 5 iterações pelo SQLQueryStress:





Plano de execução:



A function está sendo executada para todas as linhas da tabela OrdersBig.

Recomendação:

Alterar o código para jogar o valor de resultado da function para uma variável e utilizar a variável no filtro do select, por exemplo:

DECLARE @dt Date

SET @dt = dbo.fn\_RetornaDia()

SELECT \*

FROM OrdersBig

INNER JOIN CustomersBig\_TMP

ON CustomersBig\_TMP.CustomerID = OrdersBig.CustomerID

WHERE OrdersBig.OrderDate < @dt

OPTION (RECOMPILE)

O resultado final obtido é o seguinte:

