**Desafio SQL – 2**

1. Tempo para rodar o desafio antes da solução:

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

1. Tempo de para rodar após a solução:

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

1. Profiler Antes da solução:

Interface gráfica do usuário, Tabela

Descrição gerada automaticamente

1. Profiler após a solução:

Interface gráfica do usuário, Aplicativo, Tabela

Descrição gerada automaticamente

1. Plano de execução Antes da solução:

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

1. Plano de execução após a solução:

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

1. Código da procedure após a solução:

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

1. Inclusão dos campos [CompanyName],[ContactName],[InsertedDate] no índice [ixcKey]

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

**Solução**

É possível verificar que o processo executado antes da solução consumia muitos recursos de CPU e também realizava a leitura de muitas páginas lógicas. Isso se dá o fato de na procedure **st\_Proc0** existir um **OR** para colocar o conteúdo da varável em maiúscula e minúscula, sendo assim, mesmo que já exista um índice na coluna **cKey**, ele não seja usado da melhor forma para fazer a busca, pois o SQL deixa de fazer um seek. Verifiquei também. Como o OR na procedure foi incluído apenas para deixar o conteúdo em maiúscula e minúsculo, logo imaginei que a collation do campo fosse do tipo Case Sensitive, e conclui quando realizei a verificação.

Realizei 4 procedimentos para melhorar a performance da consulta:

1. Dropei o índice ixcKey para poder alterar a collation do campo;
2. Realizei a alteração da collation do campo cKey para **SQL\_Latin1\_General\_CP1\_CI\_AI**;
3. Recriei o índice agora incluindo os campos **[CompanyName],[ContactName],[InsertedDate]**;
4. Alterei a procedure st\_Proc0, removendo o OR e alterei o campo cKey para ser listado primeiro na query.

Dessa forma o SQL realizou um Seek direto no campo, evitando o paralelismo, consumiu menos CPU e realizou menos leitura de páginas conforme pode ser verificado nos prints acima.