Administração de Banco de dados

José Antônio da Cunha
CEFET-RN

Índices na Otimização de Consultas

Criar *índice* eficiente não é uma tarefa simples; requer conhecimento das queries em execução e dos diferentes tipos de índices disponíveis.

Índices na Otimização de Consultas

Estrutura interna de um índice

Índices são estruturas que possuem algoritmos otimizados para acessar dados. Assim como nas tabelas, páginas de índices também ocupam espaço físico. O corpo de um índice é formado pelas colunas da tabela cujos dados se deseja classificar seguido de uma referência conhecida como “ponteiro”, que serve para localizar a chave na página de dados da tabela.

Nota: o índice cluster não utiliza ponteiro.

Índices na Otimização de Consultas

Estrutura interna de um índice

Índices no SQL Server 2000 são construídos sobre estruturas denominadas árvores balanceadas (=“B-Tree”), cujo desenho lembra o esqueleto de uma pirâmide. A idéia desse algoritmo é fornecer um modelo de pesquisa que agilize o processo de busca, efetuando um número reduzido de leituras nas páginas do índice para que se obtenha a localização da chave pesquisada.

Índices na Otimização de Consultas

Estrutura interna de um índice

Quando procuramos por determinada palavra num livro, localizamos a(s) página(s) desejada(s) através de uma busca em seu índice. Se fossemos ensinar alguém como procurar a palavra "ADMIN" num livro de SQL Server, provavelmente ensinaríamos alguma coisa assim:

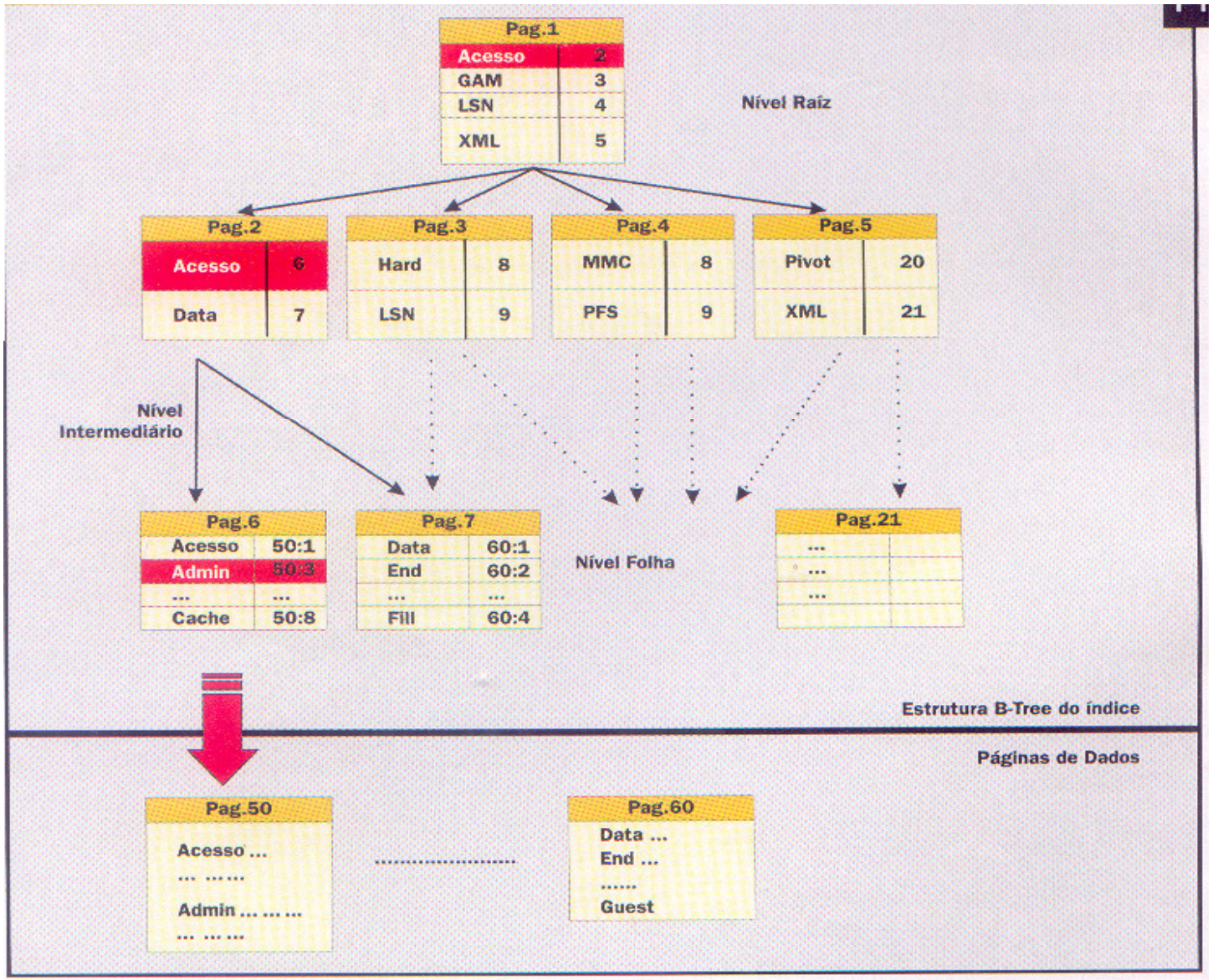
1. Localize o índice remissivo no final do livro;
2. Procure o bloco de palavras que iniciam pela letra "A";
3. Efetue uma leitura seqüencial nesse bloco até localizar a palavra desejada.

Índices na Otimização de Consultas

Estrutura interna de um índice

A Figura na página seguinte, ilustra um processo de busca envolvendo a mesma pesquisa anterior numa árvore B-Tree de um índice não-cluster. O processo tem início numa página-mestre conhecida como “root page”, procurando pela maior chave da página cujo valor é menor ou igual à palavra pesquisada. Em nosso exemplo, a primeira palavra cujo código alfabético é menor ou igual à “ADMIN” é “ACESSO”, portanto seguiremos nessa direção até a página de número 2, localizada num nível intermediário conhecido por “non leaf level”. A busca é finalizada no nível folha ou “leaf level page”, onde encontramos a referência para a página de dados onde se localiza a palavra “ADMIN”.

Índices na Otimização de Consultas



Índices na Otimização de Consultas

Tipos de índices existentes no SQL Server 2000

Existem dois tipos básicos de índices:

1. Cluster
2. Não-cluster

Índices na Otimização de Consultas

Tipos de índices existentes no SQL Server 2000

Índices cluster impõem uma organização na própria página de dados da tabela, fazendo com que permaneçam classificadas de acordo com a composição de sua chave. Portanto, se você executar o comando a seguir:

Select * from Northwind.dbo.Orders

Irá notar que os pedidos são ordenados pela coluna OrderID que faz parte do índice cluster PK_Orders. Podemos então afirmar que o leaf level de um índice cluster representa a própria página de dados da tabela, descartando a utilização de ponteiros para páginas de dados.

Índices na Otimização de Consultas

Tipos de índices existentes no SQL Server 2000

Já os índices não-cluster possuem estrutura própria, mantendo-se vinculados às páginas de dados pela utilização de ponteiros.

Índices na Otimização de Consultas

Tipos de índices existentes no SQL Server 2000

A tabela SysIndexes é responsável pelo armazenamento dos metadados do índice. Nessa tabela localizamos o nome do índice, uma indicação de seu tipo (cluster ou não-cluster), o número de páginas utilizadas, o número de alterações desde que o último cálculo de estatísticas foi executado etc.

Índices na Otimização de Consultas

Tipos de índices existentes no SQL Server 2000

Tabelas sem índice cluster – conhecidas por heaps – possuem uma linha em SysIndexes para IndId=0. Se uma tabela possui índice cluster, este será indicado por IndId=1. Porntanto, se você quiser listar as tabelas que não possuem índice cluster em seu database, basta selecionar as entradas em SysIndexes para IndId=0.

Índices na Otimização de Consultas

Tipos de índices existentes no SQL Server 2000

•O termo **cluster index scan** – é utilizado para especificar varreduras seqüenciais nas páginas de dados de uma tabela que possui índice cluster. Nesse caso, a página inicial da tabela encontra-se em SysIndexes para IndId=1.

•O termo **table scan** – é utilizado para especificar varreduras seqüenciais nas páginas de dados heaps. Nesse caso, a página inicial da tabela encontra-se em SystemIndexes.FirstIam para IndId=0.

Índices na Otimização de Consultas

Tipos de índices existentes no SQL Server 2000

Páginas de dados de tabelas com índice cluster são “ligadas” uma às outras, isto é, no cabeçalho de cada página são encontradas referências à página anterior e posterior (= Next/Previous Page). Para um processo efetuar uma leitura seqüencial numa tabela com índice cluster – conhecida como cluster index scan – precisará apenas localizar a página inicial em SysIndexes.Root; as páginas seguintes estarão encadeadas.

Índices na Otimização de Consultas

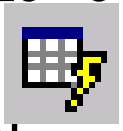
Tipos de índices existentes no SQL Server 2000

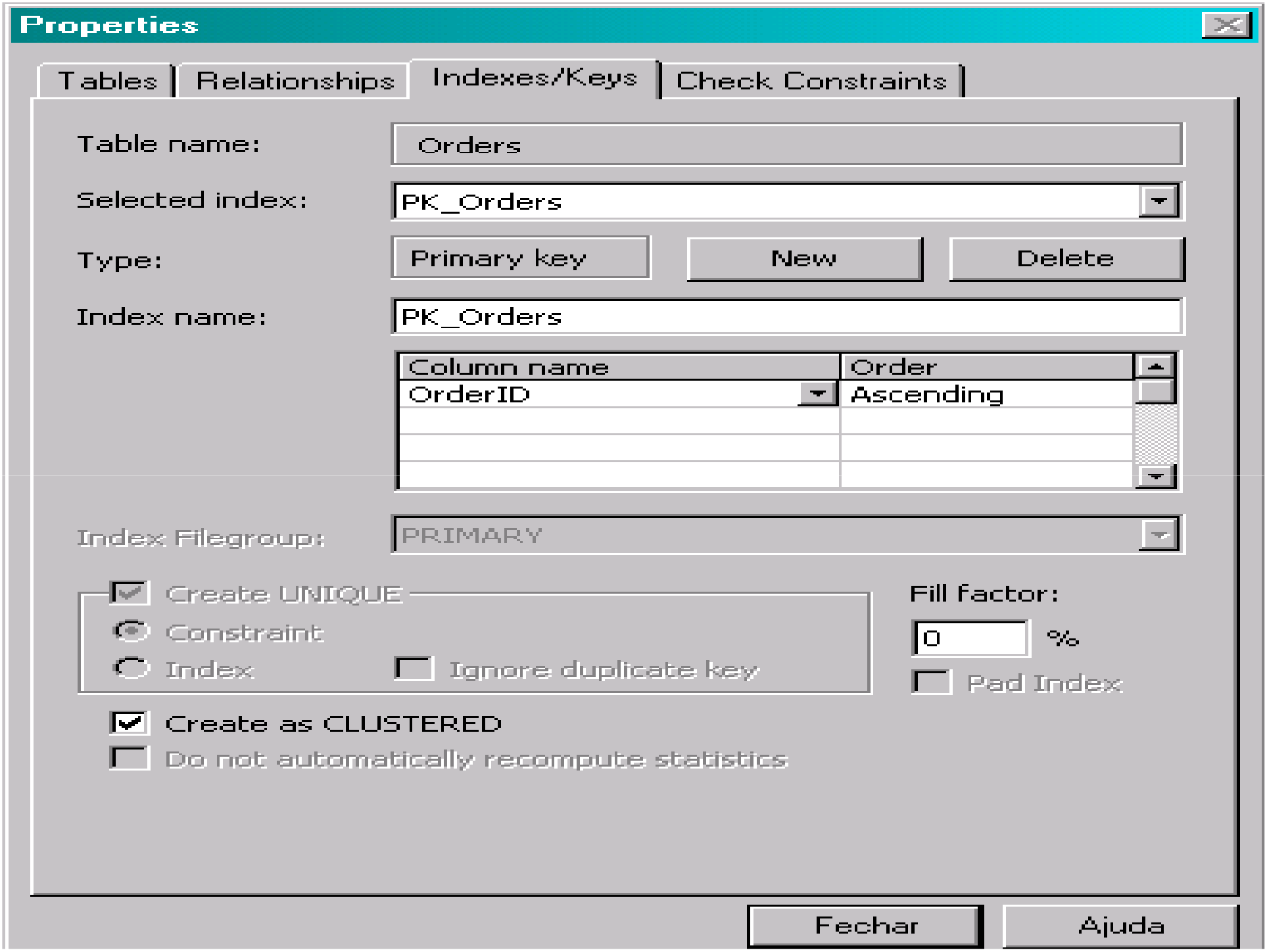
Já em **heaps** o processo é diferente pelo fato das páginas de dados não possuírem ordenação. Pode-se iniciar um lote de inserção numa página localizada “no meio da tabela”, utilizando espaço gerado por uma série de deleções e terminar o processo “no fim da tabela”, alocando-se uma nova extend.

Como já vimos, em heaps as páginas não são ligadas uma as outras. Então, para varrer as páginas pertencentes a uma heap, o SQL Server utiliza páginas especiais – denominadas páginas IAM – que controlam as páginas utilizadas por uma tabela. Portanto, num processo de leitura de um heap o SQL Server 2000 se norteia pelas páginas **IAM**.

Índices na Otimização de Consultas

Criação de um índice passo a passo

Para criar um índice na tabela Orders do banco de dados Northwind no Enterprise Manager, expanda o banco de dados selecionando a opção Tables. Clique com o botão direito sobre a tabela Orders, selecione Design Table e na barra de ferramentas clique em manager Indexes/keys  para que a tela apresentada na Figura 2 obtenha o foco principal.


Properties
Tables
Relationships
Indexes/Keys
Check Constraints
Table name:
Orders
Selected index:
PK_Orders
Type:
Primary key
New
Delete
Index name:
PK_Orders
Column name
Order
OrderID
Ascending
Index Filegroup:
PRIMARY
Create UNIQUE
Constraint
Index
Ignore duplicate key
Fill factor:
0
%
Pad Index
Create as CLUSTERED
Do not automatically recompute statistics
Fechar
Ajuda

Índices na Otimização de Consultas

As opções disponíveis na tela de manutenção de índices são:

- **Table Name**: nome da tabela onde se deseja criar o índice.
- **Type**: selecione new para criar um novo índice ou Delete para excluir um índice existente. Os tipos possíveis são Index ou Primary Key.
- **Index Name**: nome do índice.
- **Column Name... Order**: colunas que compõem a chave do índice.
- **Index Filegroup**: indicação do filegroup para criação do índice. Se você não possui discos RAID, uma boa opção para ganho de performance é criar tabelas e índices em filegroups diferentes, localizados em dispositivos distintos.

Índices na Otimização de Consultas

As opções disponíveis na tela de manutenção de índices são:

•**Create Unique**: Unique quer dizer único, que não permite duplicidades.

•**Fill Factor**: indica o percentual de preenchimento das páginas do índice no momento de sua criação. Um fator de preenchimento de 80% informa que será utilizado somente 80% da capacidade da página para ocupação das linhas do índice. O fill factor atua somente no momento da criação ou reestruturação do índice, não sendo mantido durante os processos posteriores de atualização do índice.

Índices na Otimização de Consultas

As opções disponíveis na tela de manutenção de índices são:

Vale a pena destacar também que:

1. O valor default para fill factor é zero (visível no Query Analyzer sob o comando sp_configure 'fill factor').
2. Fill factor é uma opção avançada de otimização, portanto deve ser utilizada somente naqueles índices onde se observou excessiva fragmentação. Utilizar essa opção de uma maneira genérica para todos os índices do database não é boa prática.

Índices na Otimização de Consultas

As opções disponíveis na tela de manutenção de índices são:

•**Pad Index:**fill factor atua somente no nível leaf level do índice. Assinalando essa opção, o percentual definido em fill factor será propagado para os níveis intermediários da árvore B-Tree.

•**Create as Clustered**: indica que o índice criado será do tipo cluster. Lembre-se que só é possível criar um índice cluster por tabela.

•**Do not automatically recompute statistics**: as estatística de distribuíção de dados pela chave do índice são essenciais para o otimizador avaliar uma query e, por default, são atualizadas automaticamente após um determinado número de modificações no índice.

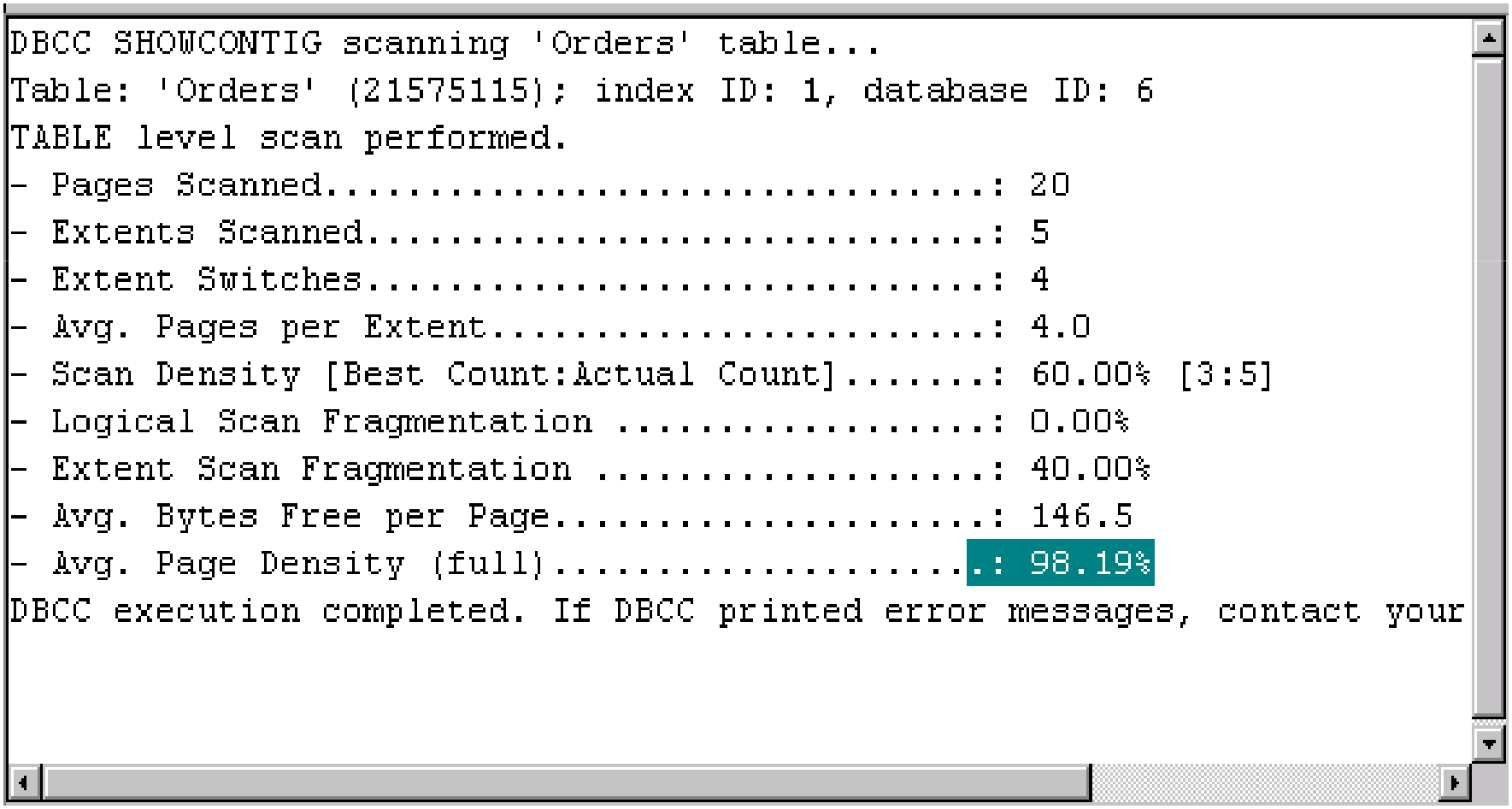
Índices na Otimização de Consultas

observação:

Considerando-se um processo semanal de reestruturação de índices, pode-se dizer que fill factor de determinado índice está adequado à medida que os indicadores do comando DBCC SHOWCONTIG Scan Density e Avg. Page Density (full) se mantêm próximos de 100%. Quanto mais distante de 100%, maior a necessidade de utilização do fillfactor para controle dos custosos page-splits. Portanto, se você encontrar índices de scan density muito inferiores a 80%, experimente estabelecer um pequeno fill factor e reavalie a fragmentação após o mesmo período. Comece, por exemplo, com um índice de 95% para fill factor e vá diminuindo até encontrar seu ponto ótimo.

Índices na Otimização de Consultas

DBCC SHOWCONTIG (Orders)



Índices na Otimização de Consultas

A sintaxe T-SQL para a criação de índices:

```
CREATE [ UNIQUE ][ CLUSTER | NONCLUSTER ] INDEX index_name
   ON { table | view } ( COLUMN [ asc | desc ] [, ...N ])
[ with < index_option > [ ,...N] ]
< index_option > ::=
     { pad_index |
       FILL FACTOR = fillfactor |
       IGNORE_DUP_KEY |
       DROP_EXISTING | STATISTICS_NORECOMPUTE |
       SORT_IN_TEMPDB
]
```

Índices na Otimização de Consultas

As opções disponíveis na tela de manutenção de índices são:

•**DROP_EXISTING:** Se droparmos o índice cluster numa tabela que possui também índice não-cluster, todos os índices não-cluster serão reconstruído, pois o ponteiro desses índices para a página de dados passará a ser RowID. Utilizando a cláusula DROP-EXISTING para que o rebuild nos índices seja efetuado SOMENTE UMA VEZ. (é aplicável somente sobre índices).

•**STATISTIC_NORECOMPUTE**: desabilita a atualização automática das estatísticas do índice, informando ao SQL Server 2000 que as estatísticas do índice serão atualizadas por processo manual. Estatística desatualizadas acarretam na escolha de planos de execução ineficientes, portanto sugire-se não utilizar essa opção.

Índices na Otimização de Consultas

As opções disponíveis na tela de manutenção de índices são:

•**SORTE_IN_TEMPDB:** se você possui o TempDB localizado num conjunto de discos separados do filegroup do banco de dados, utilize essa opção para ganho de performance na reconstrução do índice.

Índices na Otimização de Consultas

Dicas para construir e manter índices eficientes:

- Quanto mais compacto o tamanho da chave do índice, melhor;
- Criar um índice composto ou vários índices?
- Processo de Scan (Clustered Index Scan ou Table Scan) em tabelas com grande número de linhas representam gargalho de execução. Fique atento para isso.
- Procure criar sempre um índice cluster em suas tabelas.
- Bases OLTP são responsáveis por um grande volume de acesso pontuais. Nesses casos, procure criar PK´s clusterizadas e curtas.
- Em bases destinadas a consultas, reserve o índice cluster para colunas que são acessadas por range. (Notas → data)

Índices na Otimização de Consultas

Dicas para construir e manter índices eficientes:

•Se sua base de dados é utilizada tanto para operações on-line como para consultas diversas, use o bom senso: se for interessante privilegiar os processos on-line, opte por clusterizar as PK´s. se for interessante privilegiar os relatórios, reserve o índice cluster para aquelas colunas que são pesquisadas com cláusulas between, order by etc.

•Não crie índices em colunas com baixa seletividade. Colunas com alto grau de duplicidade não são uma boa escolha para índices não-cluster em função do alto custo.

•Não crie índices em tabelas com pequeno número de linhas.

•Mantenha as estatística atualizada. Mantenha as opções Auto-Create/Update Statistics ligadas.

Índices na Otimização de Consultas

Dicas para construir e manter índices eficientes:

•Crie rotinas de indexação periódicas. Rotinas de indexação são fundamentais para garantia de performance. Não se esqueça delas.

•Utilize o Profiler como ferramenta de apoio no rastreamento de queries com longo tempo de execução. Aproveite a oportunidade para criar índices mais eficientes ou mesmo dropar índices inúteis.

•Utilize o Index Tunning Wizard como ferramenta de apoio para tuning de índices.

•Ao criar índices compostos, mantenha a coluna mais seletiva no primeiro nível da chave.

•Dê preferência por índices baseados em colunas numéricas em oposição a colunas char ou varchar. Índices baseados em colunas numéricas são mais eficientes.

Índices na Otimização de Consultas

Dicas para construir e manter índices eficientes:

•Não crie índices em duplicidade. Um erro bastante comum é criar índice com a mesma estrutura de outros já existentes. Habitue-se a executar um **sp_HelpIndex** para confirmação dos índices existentes.

Índices na Otimização de Consultas

Conclusão:

Índice devem ser criados para agilizar a performance do sistema como um todo, mas freqüentemente nos esquecemos disso. Sub-avaliamos o impacto da criação de índices na performance geral do sistema, e aquilo que foi concebido como objetivo inicial de ganho de performance resulta mais em um ponto de contenção.

Otimizar um processo pode significar eliminar um índice ineficiente, implementar novos filtros ou alterar os parâmetros da cláusula join das queries em execução. Devemos sim considerar a criação de índices como recurso de otimização, mas numa análise conjunta com todos esses fatores.