Estruturas de armazenamento de dados no SQL Server

Olá pessoal. Esta semana vamos falar um pouco sobre como o SQL Server organiza internamente os dados. O assunto desta coluna é de extrema importância tanto para o DBA como para o desenvolvedor, pois conhecendo a estrutura de armazenamento podemos entender melhor como está a 'saúde' do nosso banco de dados.

Este assunto é tão importante que serve de base para o aprendizado de vários tópicos como: Backup e Restore, índices, elaboração de um capacity plan, locks, block e deadlocks, replicação, tuning,etc. A partir do que vamos aprender nesta coluna, podemos começar a seguir para outros assuntos mais avançados em termos de bancos de dados, não só do SQL Server, mas de outros fabricantes também, pois a maioria dos bancos de dados comerciais trabalha com conceitos similares.

A primeira definição que veremos está ligada diretamente com o Sistema Operacional. Para o SQL Server manipular informações, ele deve acessar tanto para a memória (RAM) ou o disco (HD). Todos os acessos feitos a estes dois recursos sempre serão feitos através do Sistema Operacional, pois é ele que faz este meio de campo entre as aplicações e os recursos.

Pois bem, o sistema operacional precisa trabalhar com blocos de informações para serem lidos ou gravados, de modo a otimizar o acesso à memória e ao disco. Estes blocos de informações são chamados de página.

Especificamente para o SQL Server, a página possui um tamanho fixo de 8Kbytes (8192 bytes) e é a menor unidade de alocação do banco de dados. Outros produtos, como o Oracle, possuem tamanho de página variado, dependendo de vários fatores, como o Sistema Operacional. Repetindo: o SQL Server sempre vai ter uma página com 8Kbytes, independente do Sistema Operacional na qual ele está sendo executado.

Existem vários tipos de páginas, e as que nos interessam são as páginas de dados, representada pela figura 1:

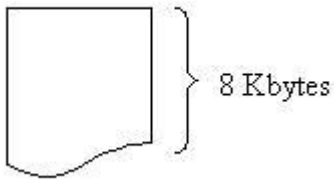

Figura 1. Representação de uma página de 8Kbytes de dados do SQL Server

Como o próprio nome já diz, estas páginas armazenam dados. No SQL Server, os dados são armazenados em uma estrutura lógica chamada de tabela, que possui suas linhas (ou registros) que por sua vez são compostos de colunas (ou campos). Quando criamos um banco de dados, devemos dizer qual é o tamanho inicial do arquivo de dados que geralmente possui a extensão .mdf. Este arquivo será dividido logicamente em páginas de dados de 8Kbytes tendo, a cada megabyte, 128 páginas de 8Kbytes.

Mais uma definição: a cada 8 páginas de dados temos uma extent, que para nós não será muito importante, pelo menos por enquanto. A figura 2 mostra as 8 páginas de uma extent.

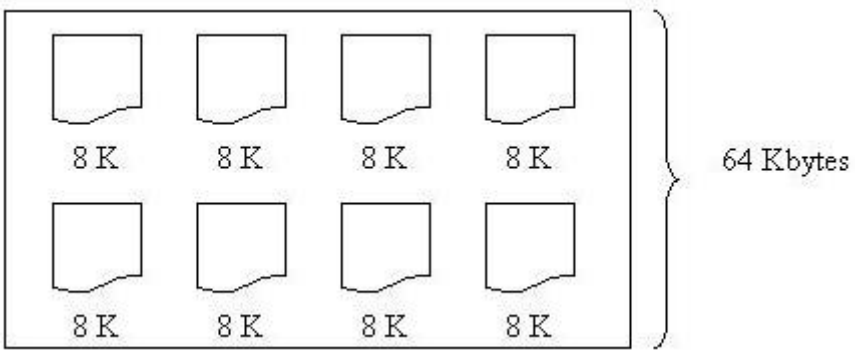

Figura 2. Representação de uma extent do SQL Server, que possui 64 Kbytes de dados

Veremos agora como os dados de nossas tabelas são armazenados dentro das páginas de dados. Lembrando que aqui estamos falando somente de dados, e não de transações, que ficam armazenadas de outra maneira dentro do arquivo especificado para o Transaction log, que geralmente possui a extensão .ldf..

Mas antes devemos saber quanto custa uma linha de dados de uma tabela, em termos de bytes, para podermos analisar como esta linha será encaixada na página. Uma linha de dados de uma tabela depende principalmente dos tipos de dados, cada qual ocupando uma quantidade de bytes. Antes de vermos alguns exemplos, devemos saber como é que fica o armazenamento de uma linha em uma tabela. A figura 3 abaixo mostra o esquema utilizado para o cálculo de custo de bytes por uma linha de dados:

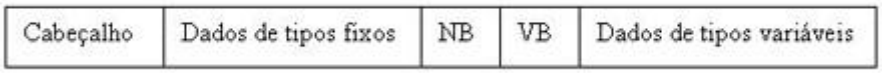

Figura 3. Esquema utilizando para o cálculo de custo de uma linha de dados no SQL Server

Onde:

Cabeçalho: Toda linha de dados possui um cabeçalho fixo de 4 bytes que conterá informações sobre as colunas.

Dados de tipos fixos: Aqui serão armazenados os valores dos tipos de dados fixos, como INT, CHAR,BIT, etc.

NB: Null Bock. Serve para armazenar a nulabilidade dos campos desta linha. Se a tabela possui até 8 campos, 1 byte é utilizado. Se possui de 8 a 16 campos, dois bytes são utilizados e assim por diante.

VB: Variable Block. Serve para armazenar a quantidade de colunas com tipos variáveis. Ocupa dois bytes. Porém para cada tipo de dados de tamanho variável temos que acrescentar mais dois bytes.

Dados de tipos variáveis: Neste local vamos armazenar os valores ocupados pelos tipos de dados de tamanho variável.

Vamos ver alguns exemplos de cálculos de tamanho de linha. Suponha a seguinte tabela abaixo:

```
CREATE TABLE TB_TAMANHO1
(
COD INT,
NOME CHAR(10),
DATA DATETIME
)
```

Vamos fazer um cálculo de quanto custa um registro desta tabela. Porém precisamos saber quanto ocupa cada tipo de dados. Esta informação é facilmente encontrada no Books OnLine do SQL Server e para o nosso exemplo temos:

INT: 4 bytes
CHAR(10): 10 Bytes (isso depende da Collation utilizada, mas vamos supor uma Collation onde cada caracter gasta um byte).
DATETIME: 8 Bytes
TOTAL: 22 bytes

Agora que já sabemos o valor dos tipos de dados, vamos calcular o valor total da linha:

```
TAMANHO = 4 + 22 + 1 = 27 Bytes
```

Percebam que na conta acima não somamos nem o VB nem o tamanho para tipos de dados de tamanho variável. Vamos ver outro exemplo:

```
CREATE TABLE TB_TAMANHO2
(
COD SMALLINT,
NOME VARCHAR(15),
FLAG BIT,
ABREV NVARCHAR(2),
HORA SMALLDATETIME
)
```

Dos tipos de dados de tamanho fixo temos:

SMALLINT: 2 bytes
BIT: 1 byte

SMALLDATETIME: 4 bytes
TOTAL: 7 bytes

Agora os tipos de dados de tamanho variável. Aqui vamos considerar o pior caso: 15 caracteres na coluna NOME e 2 caracteres na coluna ABREV:

VARCHAR(15): 15 Bytes (mais uma vez, considerando uma Collation onde cada caracter ocupa um byte).

NVARCHAR(2): 4 Bytes, pois cada caracter ocupará 2 bytes sempre, já que estamos utilizando um tipo de dados unicode.

TOTAL: 19 Bytes

Calculando o total:

TAMANHO = 4 + 7 + 1 + 2 + 19 + 2 + 2 = 37 Bytes

Percebam que desta vez acrescentamos os valores do VB, e mais 2 bytes para cada campo de tipo variável.

Agora que já sabemos como calcular o valor de uma linha, vamos ver como esta linha de encaixa em uma página de dados. Duas premissas são importantes:

1) O SQL Server nunca vai quebrar uma linha de modo a acomoda-la em uma página. Se a linha inteira não cabe na página, uma nova página de dados é alocada.

2) Em uma página de dados somente teremos linhas de uma mesma tabela.

Quando uma nova página de dados é alocada para armazenar dados, um cabeçalho de 96 bytes é alocado para esta página, deixando 8096 bytes (8192-96). Porém a própria Microsoft documenta que, no máximo, uma linha poderá conter 8060, não incluindo tipos de dados TEXT, NTEXT e IMAGE, que são armazenados em outro local.

Após o cabeçalho, cada linha é colocada seqüencialmente na página. Para cada linha colocada na página um indicador é colocado no final da página em ordem inversa. Por exemplo: suponha que em uma página estamos armazenando três linhas da tabela TB_TAMANHO1, que ocupa 27 Bytes por linha. A figura 4 abaixo mostra como ficará esta alocação:

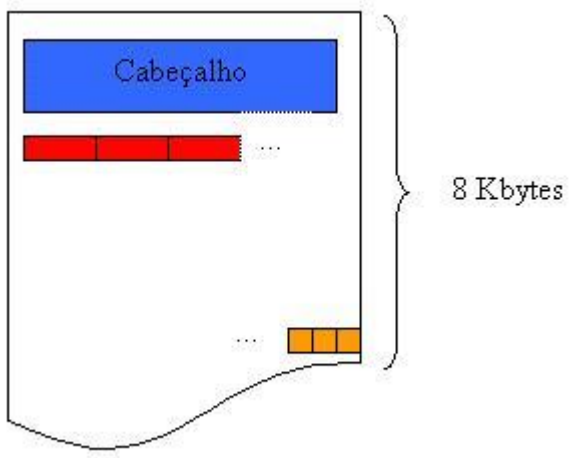

Figura 4. Representação da alocação de três linhas em uma página de dados

Percebam que na figura 4 o cabeçalho está representado em azul, as três linhas estão representadas em vermelho e os indicadores destas três linhas são representados em laranja no final da página.

Com isso concluímos como é feito o armazenamento dos dados no SQL Server. Na coluna da próxima semana mostrarei como comprovar e medir exatamente o preenchimento da página de dados.

Um detalhe final importante: dependendo do propósito, não precisamos ser exatos nas contas mostradas acima. Ás vezes somente uma boa estimativa já é o suficiente, como nos casos que se deseje saber ordem de grandeza ou uma aproximação ao invés de valores exatos.