Universidade Federal de Uberlândia (UFU)

GBC053 Gerenciamento de Banco de Dados

18/11/2019 – Prof. Humberto Razente

**Lista de exercícios**

1. *Defina árvore B. Explique o funcionamento dos algoritmos de inserção e remoção.*

Uma árvore B é uma árvore de índices multinível construída de baixo para cima que resolve o problema do custo linear da inserção e remoção, possui mutiplas chaves por nó, não requer que páginas índice sejam completas, não extende overflow para próxima página fazendo com que a página seja dividida em duas (semelhante para remoção) e funciona no geral em dispositivos de armazenamento secundário de forma mais eficiente.

* **Inserção:** Se a folha não estiver cheia a chave é inserida dentro dela de forma ordenada e se a folha estourar com a inserção a mesma é dividida em 2 novas folhas com metade dos elementos indo para uma e a outra metade para a outra, logo após isso é feita a promoção de chaves, ou seja, a maior chave das duas novas folhas sobem para o nó pai (sem inserir elementos repetidos).
* **Remoção:** Caso a remoção não cause *underflow* e não mude a maior chave do nó a chave só é removida sem outras alterações. Caso a remoção não cause *underflow* mas cause a mudança da maior chave do nó deve ser feita a troca da chave que foi promovida para o nível anterior antes da remoção pela nova maior chave. Caso a remoção cause *underflow* (número de chaves do nó < ordem / 2) ou aquele nó folha deve ser desfeito e o restante dos elementos que estavam nele fazem *merge* com algum nó irmão (nó irmão possui numero minimo de chaves) ou caso o nó irmão tenha chaves extras o nó não é desfeito mas sim é feito um balançeamento entre os nós irmãos para atingir a quantidade minima de chaves nos dois.

1. Considere uma árvore B com ordem 3 em que todos os elementos estão presentes em páginas folha. Considere o algoritmo de divisão de páginas padrão. Apresente graficamente as seguintes operações de inserção na árvore: 13, 28, 36, 45, 52, 3, 4, 8, 17, 20, 10, 13, 25, 40, 28, 36, 50, 43, 30, 7, 14, 55. Apresente graficamente a remoção dos elementos 28, 10, 40, 45, 3, 14.
2. *Qual é o número máximo de acessos a disco para encontrar uma chave em uma árvore B de ordem m no pior caso?*

Pior caso: quando todas as páginas da árvore têm somente o número mínimo de descendentes. (Considerando *m* = numero máximo de descendentes de cada página)

Então: Nível d = 2 \* [m/2] ^ (d – 1), sendo *d* o nivel da árvore (Raíz = nível 1) e sendo o nível proporcional ao numero de acessos a disco.

1. Descreva as árvores B virtuais.

Possuem a mesma estrutura que as árvores B normais porém com diferença na quantidade de seeks feitos no disco que não precisam depender necessariamente do nível da árvore, pode ser menor que isso! Isso devido ao processo de páginação em um buffer de memória em que sempre que ocorrer um *page fault* nele a página requisitada será lida do disco e logo após armazenada no mesmo para caso precise daquela página novamente não seja necessário fazer outro seek no disco. A estratégia mais comum é a LRU (*Least Recently Used*), na qual páginas vão sendo substituídas no buffer substituindo a página que está a mais tempo sem ser utilizada por uma nova.

1. Descreva as estratégias de alocação de buffers para árvores B.

Uma estrátegia é a já explicada na questão anterior que é a a LRU (*Least Recently Used*), na qual páginas vão sendo substituídas no buffer substituindo a página que está a mais tempo sem ser utilizada por uma nova. E outra estratégia seria se basear na altura da árvore, que é um modo mais direto que a LRU pois se retem no buffer as páginas que ocorrem nos primeiros níveis da árvore. Exemplo: considerando buffer de 256 KB e indice de 1 MB, utilização de 1,2 MB no disco e uso de páginas de 4 KB e a árvore contendo uma raíz e 10 páginas no segundo nível seria possível armazenar todas essas páginas dentro do buffer e ainda sobraria espaço para a utilização de outra estratégia ao mesmo tempo (como a LRU por exemplo).

1. Descreva as árvores B+ de prefixo simples.

São árvores muito parecida com as B normais porém com uma estrutura na qual somente os nós folhas (blocos) possuem ponteiros para os dados dentro do arquivo de registros, diferente das B em que todas as chaves de todos os nós possuem ponteiros para os dados do arquivo de registros. Além disso, os blocos das árvores B+ mantem um ponteiro para seu successor e seu antecessor garantindo acesso sequencial a todos eles através de qualquer um. Outro detalhe é que blocos logicamente adjacentes na árvore (devido a ordenação) não necessariamente estão adjacentes físicamente dentro do arquivo.

1. Explique o algoritmo de ordenação externa sort-merge.

Pega-se o arquivo original e divide em pequenas partes que caibam na memória principal, após isso, na RAM, é utilizado algum algoritmo de ordenação para ordenar cada umas dessas partes separadamente fazendo com que, no término de cada uma, a saída seja temporariamente salva em um arquivo em disco para que, quando esse processo finalize para todo o arquivo original, seja feita a intercalação de todos esses arquivos temporários e assim gerar um único arquivo final ordenado.

1. Cite e explique os algoritmos para implementação da operação SELECT.

* **Busca linear (força bruta):** recupera cada registro e testa se os valores dos atributos satisfazem o critério de seleção.
* **Usar a chave primária ou chave hash** para recuperar um único registro: se a condição de seleção envolver uma comparação de igualdade, usar o índice para recuperar o registro.
* **Usar um índice primário para recuperar múltiplos registros:** se a condição de comparação for >, >=, < ou <= sobre uma chave com um índice primário, usar o índice para encontrar o registro que satisfaz a igualdade e então percorrer o índice em ordem para recuperar os registros seguintes que também satisfazem a condição.
* **Uso de um índice secundário (B+-tree)**: em uma comparação por igualdade, a B+tree pode ser usada para recuperar um único registro (se o campo for chave) ou recuperar múltiplos registros (se o campo não for chave). Adicionalmente, pode ser usado para recuperar registros em condições envolvendo >,>=, <= (abrangência)
* **Uso de um índice de bitmap:** atributos com baixa cardinalidade de valores distintos podem se beneficiar de um índice de bitmap

**–** exemplos atributos como: estado civil, região, estado

**--** desde que apresentem baixa cardinalidade

**--** tabelas majoritariamente estáticas: tabelas que não sofram atualizações constantes, pois atualizações nesses índices são caras!

1. Cite e explique os algoritmos para implementação da operação JOIN.
2. Explique porque as operações básicas para execução de uma consulta são implementadas como pipelines.

Uma consulta é mapeada em uma sequência de operações

Cada execução produz um resultado temporário

Salvar arquivos temporários em disco consome tempo

* Criar pipeline de dados através de múltiplas operações: passando os resultados de um operador prévio ao próximo sem esperar completar a operação previa (Evita ao máximo valores temporários)