Livro Eletrônico

Estratégia
CONCURSOS

Aula 00

Banco de Dados AvanÃ§ados para Concursos - Curso Regular 2017

Professor: Thiago Rodrigues Cavalcanti

AULA 00: Armazenamento de Dados e Indexação

Sumário

Apresentação do professor

Olá senhoras e senhores! Hoje começamos mais um curso relacionado a Banco de dados, apresentado alguns **conceitos avançados sobre o assunto**.

É um prazer imenso fazer parte desta equipe de professores do Estratégia Concursos e ter a oportunidade de apresentar um pouco do meu conhecimento e experiência em concursos públicos! Gostaria, antes de começar de fato o conteúdo teórico desta aula, de me apresentar de forma rápida.

Meu nome é Thiago, sou casado, tenho um filho de cinco anos. Sou cristão. Frequento a IPN – Igreja Presbiteriana Nacional. Sou formado em Ciência da Computação pela UFPE. Tenho mestrado em engenharia de software na mesma instituição. Frequento academia para manter a forma, mas meu hobby mesmo é pedalar! Decidi vender o carro e viver num desafio intermodal de transporte. Vou para o trabalho de *bike* sempre que possível! Ultimamente tenho usado mais Uber do que a magrela, mais isso é um detalhe!

Onde eu trabalho? No Banco Central do Brasil! Fruto de uma trajetória de dois anos de estudos diários. Aposentei as canetas em 2010. Trabalho com análise e modelagem de dados. Já passei por equipes de desenvolvimento de

software, mas, desde 2014, estou em uma nova área dentro do Departamento de Informática (Deinf) que trata de dados e inteligência de negócio (BI). Falamos um pouco sobre como está estruturado o Deinf de uma das principais autarquias da administração pública federal no **webnário** do Estratégia.

Minha mais recente experiência com dados, seja na administração ou modelagem, é parte de uma estratégia profissional de alinhar meu trabalho diário como servidor público com minha carreira paralela de professor e consultor de Banco de Dados (BD) e *Bussiness Inteligence* (BI). A ideia é conseguir me especializar cada vez mais no tema, nesta nova carreira dentro da TI, que o mercado está denominando de cientista dos dados.

Entrei no universo de concurso há alguns anos. Desde 2012 tenho me dedicado especificamente ao conteúdo de BD e BI. Tenho experiência em cursos presencias aqui em Brasília e em diversas partes do Brasil, bem como tenho gravado sistematicamente aulas on-line. É com essa bagagem que eu me apresento aos senhores como professor. A ideia é desenvolver um material completo, recheado de questões e com dicas exclusivas para ajudar você no seu objetivo: **ser aprovado e nomeado**!

Para finalizar, não deixe de seguir minha página no Facebook® (profthiagocavalcanti), onde eu publico, sistematicamente, questões comentadas e dicas semanais. Outra atividade que mantenho regularmente são as *lives* via Periscope® (@rcthiago), toda quinta-feira, 21h, apresentamos um conteúdo rápido, focado em concurso que pode ajudar na sua preparação.

Motivação para o curso

Esse curso vai ser desenvolvido para lapidar e aprimorar os seus conhecimentos em banco de dados. A ideia é construir a base teórica dos conteúdos avançados de banco de dados que geralmente são cobrados pelas mais diversas bancas. Neste contexto, optamos por selecionar os assuntos de maior relevância para concursos públicos.

Teremos muito trabalho pela frente. Por isso, montamos um **curso teórico em PDF,** baseado nas mais diversas bancas, e apresentando o conteúdo observando as variadas formas de cobrança do mesmo pelas bancas examinadoras.

Teremos ainda videoaulas que apresentam o conteúdo teórico de forma detalhada para algumas partes da matéria. Acredito que todos os vídeos referentes às aulas deste curso deverão estar disponíveis **até o final de 2017**. Nosso objetivo é garantir que você tenha capacidade e conhecimento para ser aprovado. Logo, todo conteúdo necessário para a prova estará presente nos PDFs e será complementado pelas videoaulas.

Vamos juntos?

Observação importante II: todo o conteúdo deste curso encontra-se completo em nossos textos escritos. As videoaulas, caso existam, visam reforçar o aprendizado, especialmente para aqueles que possuem maior facilidade de aprendizado com vídeos e/ou querem ter mais uma opção para o aprendizado.

Cronograma

Para proporcionar uma visão geral do assunto e fornecer uma linha de ação para o estudo da matéria dividimos o curso em cinco aulas, sendo esta a aula 00. A aula engloba a parte de estruturas de arquivos e métodos de acesso. Falaremos ainda sobre indexação de banco de dados. As demais aulas, seguindo a ementa do curso, são apresentadas abaixo e estão distribuídas como se segue:

EMENTA DO CURSO: Armazenamento de dados e indexação, Processamento e otimização de consulta, Banco de dados distribuídos e paralelos, Segurança em banco de dados e Noções e georreferenciamento e banco de dados orientado a objetos

Pois bem, e como serão distribuídas as nossas aulas?

Aula 00 – Armazenamento de dados e indexação

Aula 01 - Processamento e otimização de consulta

Aula 02 – Banco de dados distribuídos e paralelos

Aula 03 – Segurança em banco de dados

Aula 04 - Noções de georreferenciamento e banco de dados orientado a objetos

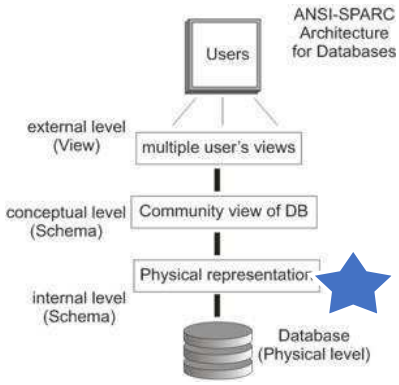
Definido o cronograma, vamos partir para o conteúdo da nossa aula demonstrativa.

Armazenamento de Dados.

O assunto da nossa aula da tratará dos aspectos físicos do banco de dados. Nosso roteiro terá como foco a **organização de arquivos e métodos de acesso**. Outra forma de descrever esse mesmo assunto usada por alguns autores é relacionar a estruturas de arquivo, indexação e *hashing*. Veremos como a estrutura física efetivamente contribui para o desempenho de um sistema de banco de dados.

1. Estrutura de arquivos

Quando estudamos estrutura de arquivos, tratamos da organização dos bancos de dados em locais de armazenamento e as técnicas para acessá-los de modo eficiente usando diversos algoritmos, alguns dos quais exigindo estruturas auxiliares chamadas índices. Se relembramos da arquitetura ANSI/SPARC, estamos falando do nível interno ou representação física dos dados.



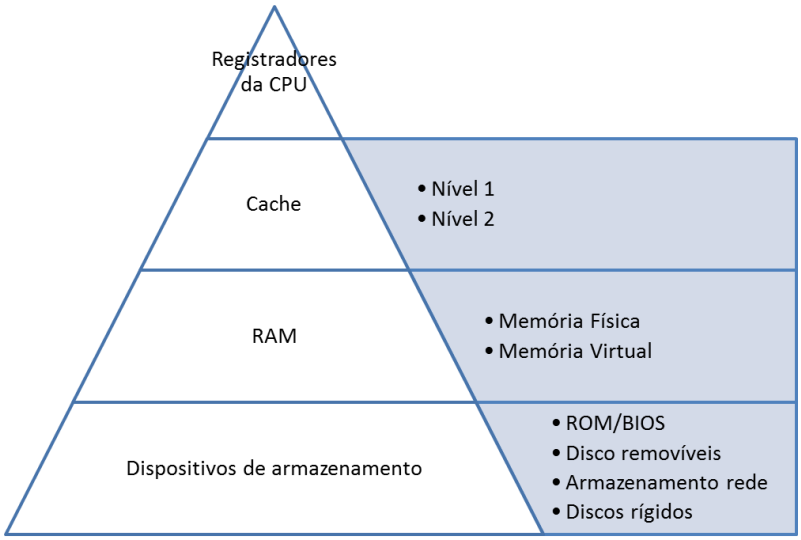
O armazenamento em diversas mídias nos leva a primeira classificação relevante dentro do assunto. As estruturas de armazenamento podem ser divididas em primárias, secundárias e terciárias. O **nível primário** pode ser operado diretamente pela CPU, essa característica faz com que ele ofereça acesso rápido aos dados, contudo sua capacidade de armazenamento é limitada e são mais caras em relação as demais.

Os níveis secundário e terciário de armazenamento estão associados a memórias não voláteis que **não** são operadas diretamente pela CPU. Por não poderem ser manipuladas diretamente pela CPU, existe uma necessidade de transferência das informações para a memória primária. Esta operação é um procedimento demorado quando comparado com as demais etapas do processamento de consultas. Podemos listar como exemplo de armazenamento secundário os discos rígidos e de terciário as fitas removíveis.

É possível ainda, separar os dispositivos físicos de armazenamento em volátil ou não volátil. Esse conceito se refere ao fato da informação ser ou não

perdida caso falte energia no sistema. A memória RAM, sigla para Random Access Memory, é um tipo de **memória volátil** que serve para rodar aplicações depois que o computador já está ligado, e cujas informações são perdidas depois do desligamento da máquina.

Na figura a seguir você por observar uma hierarquia de armazenamento, os três primeiros níveis da pirâmide tratam de armazenamento temporário. O quarto nível mostra os dispositivos de armazenamento permanente. Falaremos um pouco sobre alguns dispositivos de acesso com armazenamento temporário logo adiante.



Existem dois tipos de memória RAM que devem ser consideradas no nosso estudo: DRAM e SRAM. DRAM é a sigla em inglês para *Dynamic Random Access Memory*, ou Memória de Acesso Randômico Dinâmica. Isso significa que ela **precisa que a informação seja atualizada** o tempo todo para que permaneça armazenada. Com isso, esse tipo de RAM **gasta mais energia** se comparado com a SRAM.

A Memória de Acesso Randômico Estática (SRAM) consegue **manter os bytes mesmo sem atualização** contínua, os dados são perdidos somente após a interrupção da fonte de energia. A memória RAM estática é mais econômica, além de entregar uma performance melhor. Vejamos como esse assunto já foi cobrado em provas anteriores.

CAIU na prova!

1. ANO: 2015 BANCA: CONSULPLAN ÓRGÃO: HOB PROVA: TÉCNICO ADMINISTRATIVO - INFORMÁTICA

A memória RAM possui como características, EXCETO:

A Armazena os dados que o processador utiliza para trabalhar.

B É uma memória não volátil, pois armazena seus dados temporariamente.

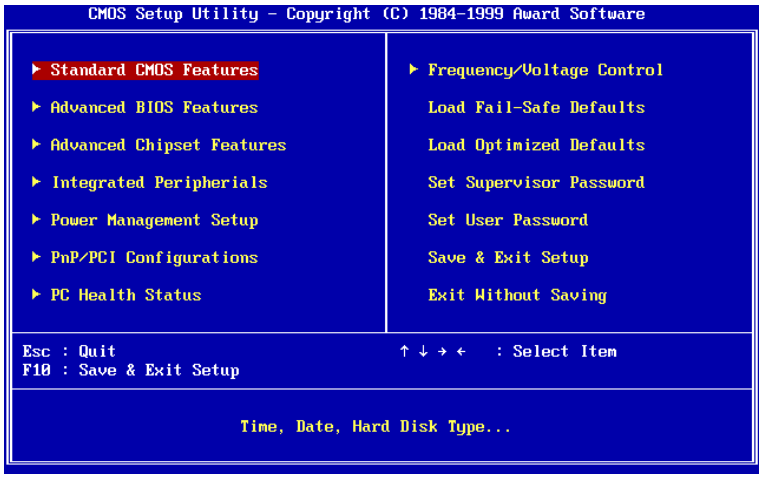
C SRAM e DRAM são tipos de tecnologia de memória RAM muito utilizados.

D A quantidade de memória influencia diretamente na capacidade de trabalho do computador.

Comentários: Vejam que acabamos de expor a características de volatilidade das memórias RAM, seja SRAM ou DRAM. Desta forma podemos concluir que existe um erro no texto na alternativa B. Como a questão pede a assertiva que não representa uma característica da memória, essa é a nossa resposta.

Gabarito: B

Quando passamos a analisar as memórias não voláteis, o primeiro tipo que devemos analisar seria as **ROM/BIOS**. ROM significa *Read-Only Memory* é uma memória somente de leitura, e está, principalmente, localizada no chip responsável pela iniciação do sistema. É lá que as informações básicas do computador ficam armazenadas, portanto não são afetadas quando o dispositivo é desligado. Lembra da tela azul que aparece quando apertamos F8 na inicialização do computador?



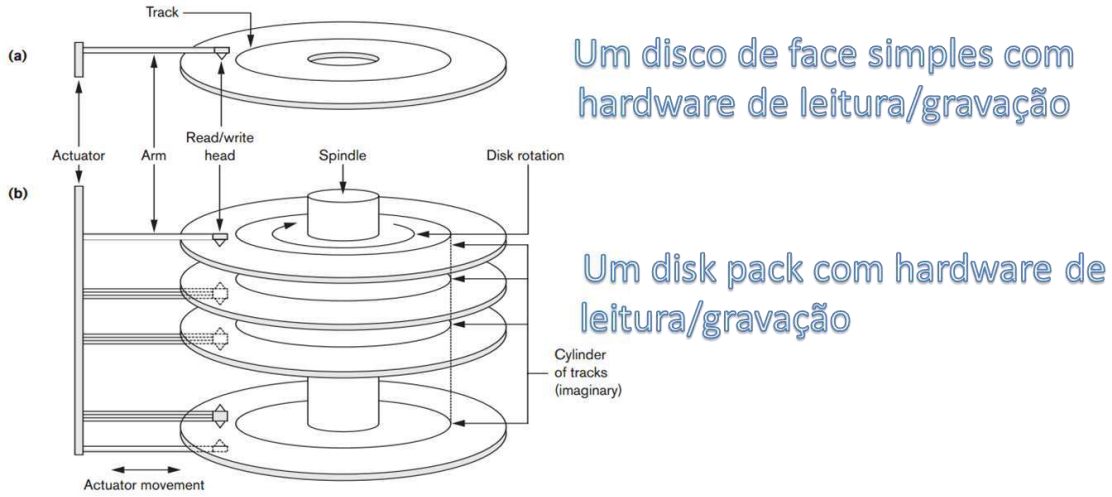
Outros dispositivos de armazenamento permanente são os discos removíveis, os dispositivos de armazenamento em rede e os discos rígidos. A organização dos dados dentro dos discos são de extrema importância dentro do assunto. Dedicaremos as próximas linhas para tratar sobre como os arquivos são organizados no disco.

1.1. Dispositivos de armazenamento secundário

Ao definir como os arquivos são organizados nos disco, definimos por tabela como podemos acessá-los. Outro ponto que temos que ter em mente: Os dados são transferidos da memória secundária para a primária, onde a CPU pode manipular os dados e, em seguida, gravados de volta na memória secundária.

Uma organização secundária ou estrutura de acesso auxiliar permite acesso mais eficiente aos registros do arquivo com base em campos alternativos, além dos que foram usados para a organização de arquivo primário. A ideia da maioria desses dispositivos auxiliares se concentra na criação de índices, falaremos sobre eles mais adiante.

Segundo Navathe: "É importante estudar e entender as propriedades e as características dos discos magnéticos e o modo como os arquivos podem ser organizados neles a fim de projetar banco de dados eficazes, com desempenho aceitável". Independente da sua capacidade, todos os discos são feitos de um material magnético modelado como um disco circular fino, ver figura abaixo:



Um disco pode ter apenas uma das suas faces uteis conforme podemos observar na parte (a) da figura acima. Outra possibilidade é ter as duas faces com capacidade de leitura e gravação. Podemos ainda ter um disk pack com vários discos em apenas um dispositivo como exemplificado na parte (b) da figura.

A figura apresenta ainda os termos técnicos que descrevem uma estrutura de disco. Por estarem em inglês vamos aproveitar as próximas linhas para descrever os termos aproveitando para expor a devida palavra em português.

Track (trilha) – é um círculo de pequena largura. Existem vários em cada uma das faces de um disco. Em disk packs, as trilhas de mesmo diâmetro são denominados cilindros. Dados armazenados no mesmo cilindro são recuperados mais rapidamente. Cada trilha é dividida em blocos ou setores.

Actuator (acionador) - é o responsável por mover o braço sob a superfície dos pratos, e assim permitir que as cabeças façam o seu trabalho. Para que a movimentação ocorra, o atuador contém em seu interior uma bobina que é induzida por imãs.

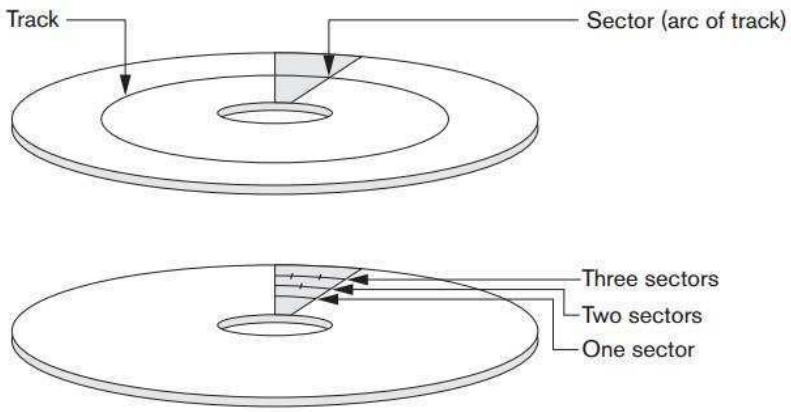
Arm (braço) - Se há um componente dos discos rígidos considerado o Tendão de Aquiles, este é o braço de acesso. O braço deve mover muito rápida e corretamente através de distâncias relativamente longas. Além disso, o movimento do braço de acesso não é contínuo — deve acelerar rapidamente ao se aproximar do cilindro desejado e então desacelerar rapidamente para evitar passar do ponto. Consequentemente, o braço de acesso deve ser forte (para suportar as forças violentas provocadas pela necessidade de movimentos rápidos) e leve ao mesmo tempo (para que haja menor massa para acelerar/desacelerar).

Read/write head (cabeça de leitura/gravação) - As cabeças de leitura/gravação do disco rígido funcionam somente quando os pratos do disco sobre os quais elas atuam estão rodando. Como é o movimento da mídia sob as cabeças que permite acessar ou ler os dados, o tempo que leva para a mídia contendo o setor desejado passar completamente por baixo da cabeça é o único determinante da contribuição da cabeça para o tempo total de acesso. A média é de 0,0086 milissegundos para um drive de 10.000 RPM com 700 setores por faixa.

Spindle (eixo) - discos ficam posicionados sob o eixo, que é responsável por fazê-los girar (**Disk Rotation**)

OK! Já sabemos como o disco funcionam e quais os elementos físicos principais compõe a estrutura de um HD. Agora vamos entender como os pratos são organizados. Cada um dos pratos presentes em um disk pack devem ser divididos em partes denominadas **blocos** e **setores.** Esses geralmente são definidos durante a formatação do disco.

Um setor é um **espaço físico** em um disco formatado que contém informações. Quando um disco é formatado, as faixas são definidas (anéis concêntricos a partir do interior para o exterior do prato de disco). Cada faixa é dividida em fatias, conhecida como setores. Os setores podem se estender por um ângulo fixo ou manter uma densidade de gravação uniforme – veja a figura a seguir. Em discos rígidos e disquetes, cada setor pode armazenar 512 bytes de dados.



Um **bloco**, por outro lado, é um **grupo de setores** que o sistema operacional pode endereçar. Um bloco pode ter um ou vários setores (2,4,8, ou até 16). Quanto maior a unidade, mais setores estarão presentes em um único bloco. Você define o tamanho do bloco durante a formatação. Um fator que influencia é o tamanho do buffer do disco.

E qual a utilidade desse buffer? Imagine que o processador está trabalhando na execução de uma determinada tarefe, ao mesmo tempo o dispositivo de entrada e saída pode ler para o buffer os próximos blocos de disco. Enfim, buffer é aquela parte de memória principal disponível para armazenamento de cópias de blocos de disco. É necessário desenvolver uma estratégia para o gerenciamento desse espaço. Observe a figura a seguir, ela tenta clarear a sua percepção sobre o uso do buffer.

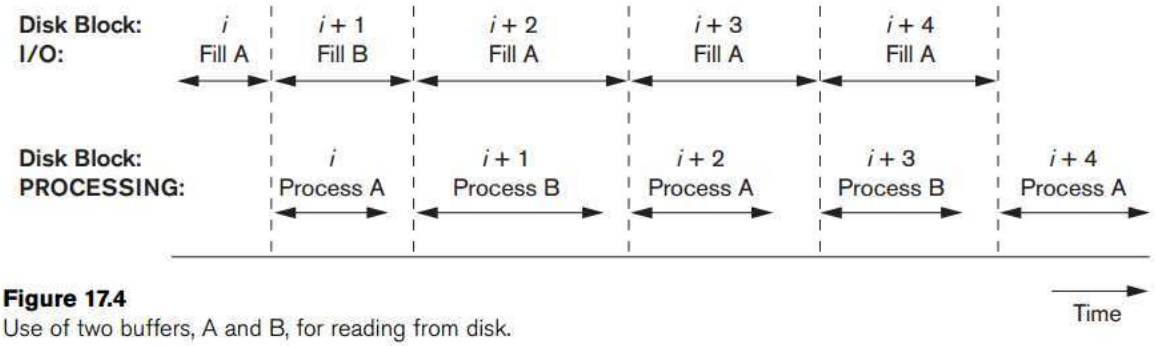


Figure 17.4
Use of two buffers, A and B, for reading from disk.

Uma consideração importante é que um dos objetivos dos sistemas de banco de dados é minimizar o número de transferências de blocos entre o disco e a memória. O tempo para transferência dos dados do disco para a memória é formado por algumas etapas. O tempo de busca, o atraso rotacional ou latência e o tempo de transferência. O tempo de busca é considerado o principal culpado pelo atraso envolvido na transferência de blocos entre o disco e a memória.

Analisaremos a seguir como os arquivos de dados ou informações são gravadas dentro dos blocos de disco. Veremos como a distribuição dos registros de um arquivo podem fazer diferença no processamento de uma consulta.

1.2. Arquivos no disco

Primeiramente precisamos da definição de **registro**, que trata basicamente de uma coleção de valores ou itens de dados. Um **arquivo,** por sua vez, é definido como uma sequência de registros. A primeira opção para compor um arquivo seria utilizarmos um conjunto de registros de tamanho fixo. Imagine que tenhamos x registros, desta forma fica fácil de definir e alocar espaço em disco, basta multiplicar a quantidade pelo tamanho de um registro.

A segunda opção seria definir arquivos cujos registros tem tamanho variável. São várias as possibilidades para que isso aconteça. Um ou mais campos de um registro são de tamanho variado. Um ou mais campos podem ter múltiplos valores para registros individuais. Um ou mais campos são opcionais. O arquivo pode ser misto, composto por registros de tamanho fixo e variável.

O Silberchatz gasta algumas páginas para evoluir com o conceito de tamanho fixo para tamanho variável de registro no arquivo. Aqui, vamos nos limitar a exemplificar os conceitos com uma figura, na nossa opinião isso é maios do que suficiente para provas de concurso. Vejam a imagem a seguir.

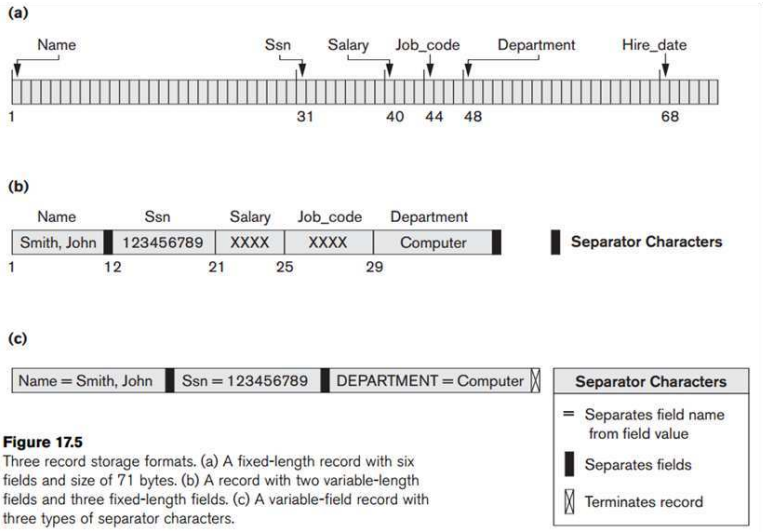


Figure 17.5
Three record storage formats. (a) A fixed-length record with six fields and size of 71 bytes. (b) A record with two variable-length fields and three fixed-length fields. (c) A variable-field record with three types of separator characters.

fixo

Observem que na primeira figura acima temos um registro de tamanho fixo. Observem que sabemos a posição das informações dentro do registro quando observamos a definição do mesmo. Para lermos o salário do funcionário basta lermos o campo que começa na posição 40 do vetor de bytes. Vejam que temos seis campos de tamanho total de 71 bytes.

No registro apresentado na letra (b) temos dois campos de tamanho variáveis que são delimitados por um caractere de separação. E temos ainda três campos de tamanho fixo, quais sejam: San, Salary e Job_Code. Por fim, na opção (c) temos um registro que apresenta algumas propriedades interessantes. Primeiro a descrição e o valor de cada campo aparecem dentro do registro.

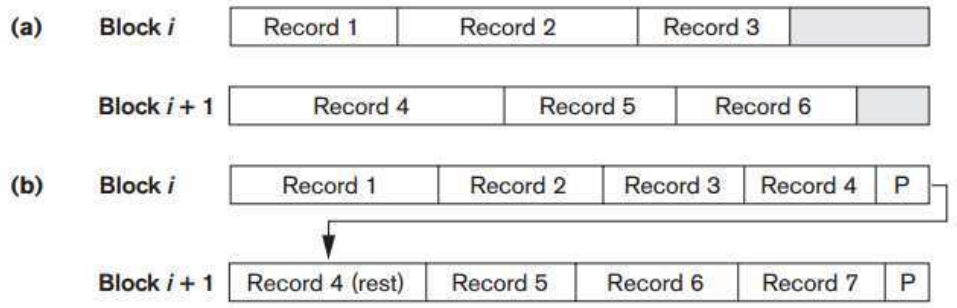
Segundo, da mesma forma da alternativa anterior (b), temos uma marca para delimitar os campos. Por fim, temos um delimitador de fim registro.

Vejam que do ponto de vista prático o que acontece nos arquivos de dados que conhecemos hoje é uma variação do que foi descrito acima com várias outras informações que compõe o cabeçalho do arquivo. Esta aula não está interessada neste cabeçalho, que deve ser tratada dentro do conteúdo de sistemas operacionais na parte de gerenciamento de arquivos.

Reforçamos que o objetivo de uma boa organização de arquivos é localizar o bloco que contém um registro desejado com um número mínimo de transferência de bloco. Esse cabeçalho ou descritor contém informações que são exigidas pelos programas do sistema que acessam os registros do arquivo. Servem para determinar os endereços de disco dos blocos, registrar descrições de formato, tamanho e ordem dos campos em um registro, para registros não espalhados de tamanho fixo. Ou ainda registra o código de tipo de campo, caracteres separadores e códigos de tipo de registro, para registros de tamanho variado.

Agora já sabemos como cada arquivo é composto. Que os registros fazem parte dos arquivos, vamos fazer a associação entre os registros e os blocos do disco. Imagine que o espaço restante em um bloco não seja suficiente para o armazenamento do próximo registro a ser inserido no arquivo. Qual atitude devemos tomar?

São duas as opções possíveis. A primeira é conhecida como *unspanned* ou não espalhada, onde os registros não podem ultrapassar o tamanho do bloco. A outra opção é denominada *spanned* ou espalhada, neste caso um ponteiro no final do primeiro bloco aponta para o bloco que contém o restante do registro, caso não seja o próximo bloco consecutivo. Vejam a diferença na figura apresentada a seguir, a letra (a) representa a opção não espalhada e a letra (b) representa a opção espalhada:

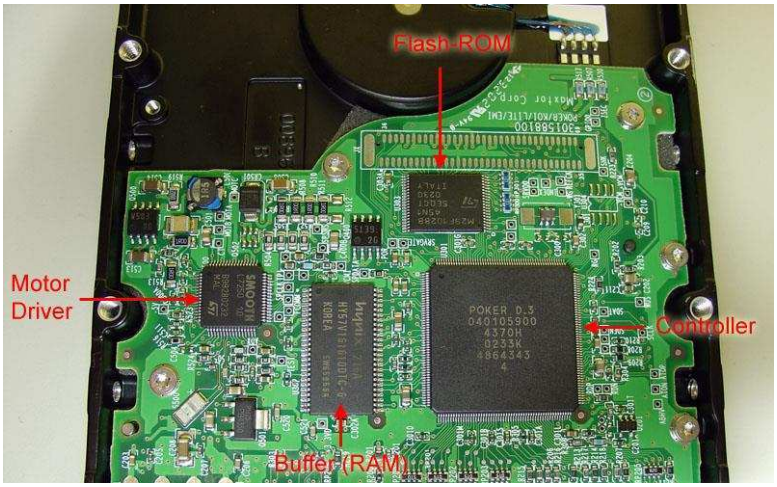


A figura acima nos ajuda a calcular outra definição importante dentro do assunto: o fator de blocagem (Bfr). Seja B o tamanho do bloco e R o tamanho do registro, se B ≥ R, bfr = ⌊B/R⌋ (função piso da divisão). Esse valor define a quantidade registros por bloco de disco.

Para alocar os blocos de arquivos no disco, o ideal é que tenhamos uma **alocação contínua**. Esta posição dos blocos no disco torna a litura mais rápida, principalmente, se tiver utilizando uma buffer de leitura. Outra opção para **alocação** do arquivo seria a forma **ligada**. Nela cada bloco contém um ponteiro para o próximo. Isso facilita a expansão do arquivo, contudo torna a leitura mais lenta.

Podemos juntar as duas opções acima criando **clusters de bloco**. Nele temos conjuntos de blocos contínuos para cada segmento de arquivo. A última opção seria a alocação indexada, onde um ou mais blocos de índice contêm ponteiros para os blocos de arquivos reais.

Quando pensamos em discos rígidos, o endereço de hardware de um bloco é uma combinação de **número de cilindro**, **número de trilha** e **número de bloco,** ele é fornecido ao hardware de E/S do disco. Algumas características dos discos são o endereçamento por acesso aleatório, a transferência de dados para memória principal que ocorre em unidades de blocos. Esses blocos tem um endereçamento físico, conhecido como endereço de hardware, um endereço lógico, denominado LBA – Logical Block Address, e o endereço de buffer que é usado na leitura. Veja a figura abaixo uma foto de um disco com alguma das suas unidades ou estruturas apontadas.



Os HDs são conectados ao computador por meio de interfaces capazes de transmitir os dados entre um e outro de maneira segura e eficiente. Há várias tecnologias para isso, sendo as mais comuns os padrões IDE, SCSI e, atualmente, SATA.

O controlador de disco (HDC) é comumente embutido na unidade de disco. Ele controla a unidade de disco e interliga ao sistema de computação, geralmente por meio de uma interface. O controlador aceita comandos de alto nível de E/S e controla a execução. A localização dos dados no disco é um gargalo principal nas aplicações de banco de dados. O tempo de E/S é

constituído pelo somatório do tempo de busca, do atraso rotacional ou latência e do tempo de transferência.

Outro dispositivo comum à placa lógica é um pequeno chip de memória conhecido como buffer (ou cache). Cabe a ele a tarefa de armazenar pequenas quantidades de dados durante a comunicação com o computador. Como este chip consegue lidar com os dados de maneira mais rápida que os discos rígidos, seu uso agiliza o processo de transferência de informações. No mercado, atualmente, é comum encontrar discos rígidos que possuem buffer com capacidade entre 2 MB e 64 MB.

Já temos os arquivos e seus registros! Que tal agora aprendermos como fazer operações sobre eles? Apresentamos abaixo um conjunto de operações que são executadas para recuperação de informações em um arquivo.

Operação	Descrição
Open	Prepara o arquivo para leitura ou gravação. Aloca buffers apropriados. Define o ponteiro para o início do arquivo
Reset	Define o ponteiro aberto para o início do arquivo
Find (Locate)	Procura o primeiro registro que satisfaça um condição de pesquisa. Transfere o bloco para o buffer de memória principal. O ponteiro aponta para o registro no buffer e este se torna o registro atual.
Read (Get)	Copia o registro atual do buffer para uma variável de programa no programa do usuário. Esse comando também pode avançar o ponteiro do registro atual para o próximo registro no arquivo, que pode precisar ler o próximo bloco de arquivo no disco.
FindNext	Procura o próximo registro no arquivo que satisfaz a condição. Transfere o bloco para o buffer. O registro via o atual.
Delete	Exclui o registro atual e atualiza o disco (no fim)
Modify	Modifica valores do registro e atualiza (no fim) o disco
Insert	Insere um novo registro no arquivo ao localizar o bloco onde o registro deve ser inserido, transfere o bloco para o buffer da memória principal, gravando o registro no buffer e (no fim) no disco
Close	Completa o acesso ao arquivo liberando os buffers e realizando quaisquer outras operações de limpeza necessárias.

Estas operações, exceto open e close, são chamadas operações um registro por vez, pois cada uma se aplica a um único registro a cada instante. Outra operação importante é o scan, vista no quadro abaixo:

Operação	Descrição
Scan	Se o arquivo já estiver aberto ou reiniciado, Scan retorna o primeiro registro; caso contrário, ele retorna o próximo registro. Se uma condição específica for definida, o registro retornado é o primeiro ou o próximo que satisfaz a condição.

Existem algumas outras operações que podem ser fornecidas pelo SGBD ou pelo sistema de entrada e saída do SO. Essas operações são responsáveis por retornar ou manipular mais de um registro que satisfaça a uma determinada condição. Observem a lista no quadro a seguir:

Operação	Descrição
FindAll	Localiza todos os registros no arquivo que satisfazem uma condição de pesquisa
Find (Locate) n	Procura o primeiro registro que satisfaz a condição de pesquisa e depois continua a localizar os próximos n – 1. Transfere os blocos que contém os n registros para o buffer da memória principal.
FindOrdered	Recupera todos os registros no arquivo em alguma ordem especificada.
Reorganize	Inicia o processo de reorganização. Algumas organizações de arquivo exigem reorganização periódica. Ex: reordenar os registros classificando-os em um campo especificado.

Agora que já conhecemos os princípios básicos dos arquivos vamos seguir para entender sua organização.

1.3. Organização de arquivos

A organização de arquivo e métodos de acesso são estudas em conjunto mais possuem definições distintas. Antes de começarmos a tratar dos termos práticos, vamos fazer um exercício mental. Pegue o livro que está mais próximo de você! Procure a expressão banco de dados. Você tem várias formas de executar essa pesquisa. Supondo que seja um livro técnico de Tecnologia da Informação, você pode folhear as páginas até encontrar o termo solicitado. Outra opção seria pesquisar no índice, ou no índice remissivo, que geralmente é colocado ao final do livro.

Na história que acabamos de contar o livro é o arquivo, e a forma como eles está estruturado é conhecido como organização. O método de acesso é o passo-a-passo que você segue para chegar ao assunto que ti interessa. Após a nossa rápida metáfora, vamos agora para os termos e conceitos mais formais sobre o assunto.

A **organização de arquivos r**efere-se à organização dos dados de um arquivo em registros, blocos e estruturas de acesso. Inclui o modo como registros e blocos são colocados no meio de armazenamento e interligados. Os **métodos de acesso** oferecem operações que podem ser aplicadas a um arquivo. É possível aplicar vários métodos de acesso a uma organização de arquivos.

Alguns métodos de acesso, porém, só podem ser aplicados a arquivos organizados de certas maneiras. Por exemplo, não podemos aplicar o método de acesso indexado a um arquivo sem um índice. Várias técnicas gerais, como ordenação, hashing e indexação, são usadas para criar métodos de acesso.

As formas de organização de arquivos podem ser classificadas em: registros desordenados, conhecido também como arquivo de *heap*; registros ordenados ou arquivo classificado; registros com *hashing*; registros mistos; e *B-tree* (árvore). Falaremos de cada um deles a partir de agora.

Os **arquivos de registros desordenados** ou arquivos de *heap* (*Heap files*) são organizados na ordem em os registros que são inseridos no arquivo. A principal característica é o fato de novos registros serem sempre inseridos no final do arquivo. Isso garante uma inserção extremamente eficiente! Contudo a pesquisa é feita de forma linear, lendo todos os registros em ordem, o que torna o tempo de leitura alto quando comparado com outras organizações. Outro problema é a exclusão que deixa espaço livre. Existem algumas formas de resolver esse problema.

Uma delas é usar o espaço de registros excluídos ao inserir novos registros, embora isso exija uma manutenção extra para se manter informado sobre os locais vazios. Outra opção é usar a reorganização de arquivo para reduzir os espaços livre resultantes da exclusão. Todas as opções acabam por requerer processamentos adicionais.

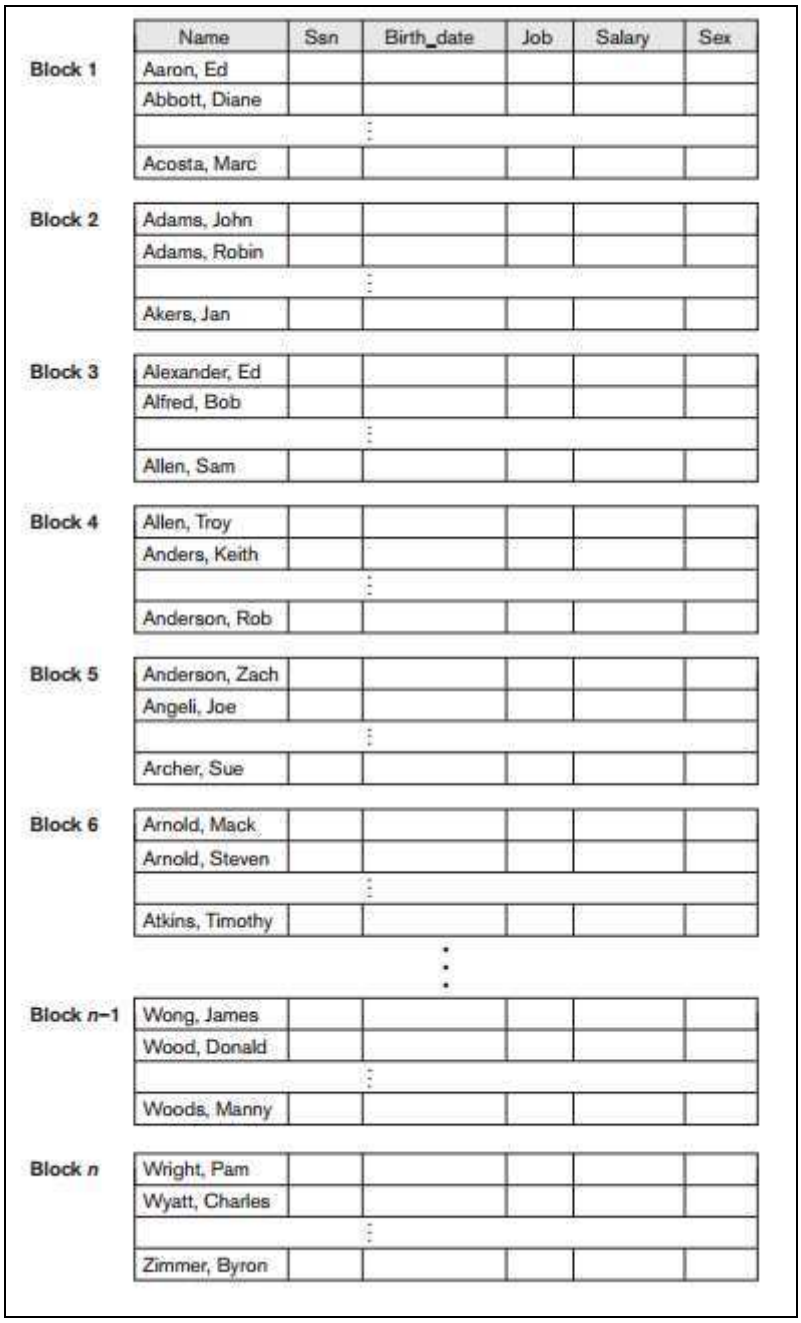
Para ler todos os registros na ordem dos valores de algum campo, criamos uma cópia classificada do arquivo. A classificação é uma operação cara para um arquivo de disco grande, e técnicas especiais de classificação externa são utilizadas. Uma solução é ter os arquivos já ordenados.

Os **arquivos de registros ordenados**, classificados ou sequencial são armazenados de forma ordenada de acordo com algum campo do registro, conhecido como campo de ordenação. Se esse campo for a chave do registro em questão, também podemos chama-la de chave de ordenação. As vantagens desta forma de organização são uma leitura mais eficiente dos registros, carregar o próximo registro, em geral, não precisar recarregar o bloco do disco e a possibilidade de usar busca binária.

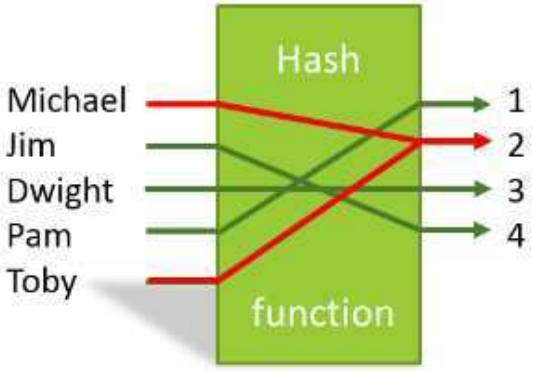
Vejamos um exemplo para verificarmos essas características. Observem a relação EMPLOYEE descrita na figura abaixo. A figura representa blocos com informações dos funcionários ordenadas alfabeticamente. Percebam que podemos fazer uso de pesquisa binária sobre os blocos. Os arquivos ordenados estão em blocos e armazenados em cilindros contíguos para minimizar o tempo de busca.

Os arquivos de registros ordenados, porém, **não** oferece vantagem para busca por campos diferentes do ordenado. A inserção e a exclusão de registros são operações dispendiosas. Na inclusão podemos ter que deslocar, em média, metade dos registros. Podemos então deixar espaços em branco em cada bloco, para evitar que o deslocamento supere o âmbito do bloco; ou criar um arquivo overflow ou transação. Com essa técnica, o arquivo ordenado real é chamado de arquivo principal ou mestre e outro arquivo armazena os novos registros que excedem o tamanho do bloco. Isso aumenta o custo e complexidade do algoritmo de escrita, pois um arquivo separado do principal tem que ser consultado e analisado me todas as operações.

Existem também problemas com a exclusão, que são considerados menos graves, a solução é usar marcadores e fazer a reorganização do arquivo. A preocupação com modificações só deve ser relevante se alterarmos valores de campos que fazem parte do campo de ordenação.



A próxima forma de organização de arquivos são os **arquivos de acesso direto** ou arquivos de *hash* que utilizam as técnicas de *hashing*. A ideia por trás do *hashing* é oferecer uma função h, chamada **função de *hash*** ou função de **randomização**, que é aplicada ao valor do campo de *hash* de um registro e gera o endereço **do bloco** de disco em que o registro está armazenado. Vejam o funcionamento do *hash* na figura abaixo:



A Condição de pesquisa precisa ser uma condição de igualdade em um único campo conhecido como campo *hash* ou chave *hash*. Vamos aproveitar a oportunidade para falar um pouco das técnicas de *hashing* existentes. Também é utilizado como uma estrutura de pesquisa interna em um programa. Sempre que um grupo de registros é acessado exclusivamente pelo uso do valor de um campo, por exemplo, CPF. Basicamente temos os seguintes tipos de *hashing*: *hashing* interno, *hashing* externo para arquivos de disco e técnicas de *hashing* que permitem a expansão dinâmica do arquivo.

Na minha concepção o Navathe e outros autores apresentam *hashing* interno na teoria para podermos entender como ele funciona. Nosso objetivo é fazer com que você entenda o que acontece quando aplicamos a função *hash* ao campo de *hash*. E o mais importante: após entender que o número de valores possível da função é muito menor do que todas as entradas, as colisões tornam-se, portanto, uma realidade. Mas como tratar as colisões? É o que veremos mais adiante.

O *hash* interno pode ser entendido por meio da figura anterior. A aplicação da função de *hash* nos leva a uma posição em uma tabela que armazena um registro. Vejam que existe a possibilidade da função nos levar para o mesmo endereço de memória, é o que chamamos de colisão. A pergunta mais uma vez é: como tratar as colisões?

A primeira forma é usar o **endereçamento aberto** que usa a próxima posição do *array* de registro que está disponível. Outra opção é usar o **encadeamento**, neste caso utilizamos uma lista ligada onde o registro armazenado na posição do *array* aponta para o próximo registro que tem como resultado da função *hash* o mesmo valor. No encadeamento, o novo registro é colocado em um local de overflow e um ponteiro aponta para ele. Por fim, temos a possibilidade de utilizar o ***hash* múltiplo** ou ***double-hash***, neste caso existe uma nova função que aplicaremos o nosso valor de entrada para obtermos um endereço de memória disponível.

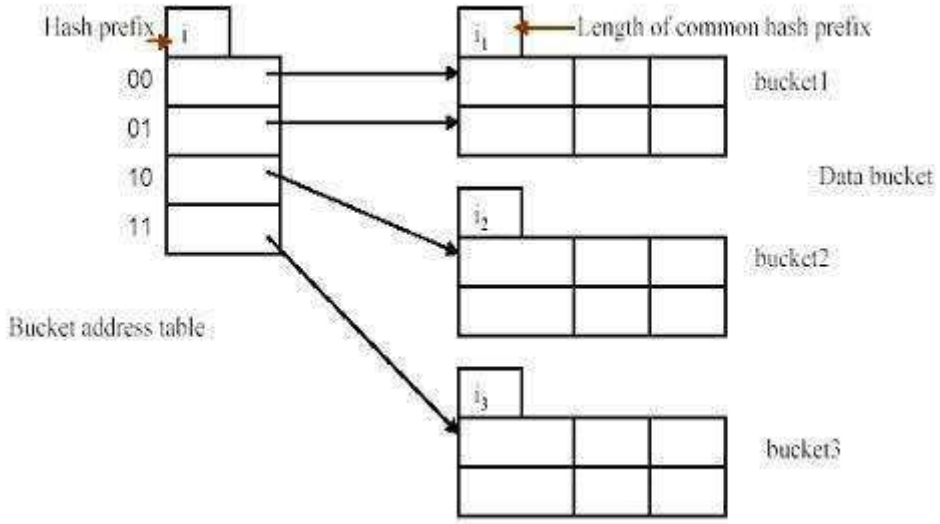
No ***hash* para arquivos em disco**, uma tabela mantida no cabeçalho do arquivo converte o número do *bucket* para o endereço do bloco de disco

correspondente. O problema da colisão é menos sério, pois, independentemente de quantos registros possam caber no *bucket* eles podem ser definidos por *hashing* ao mesmo *bucket* sem causar problema.

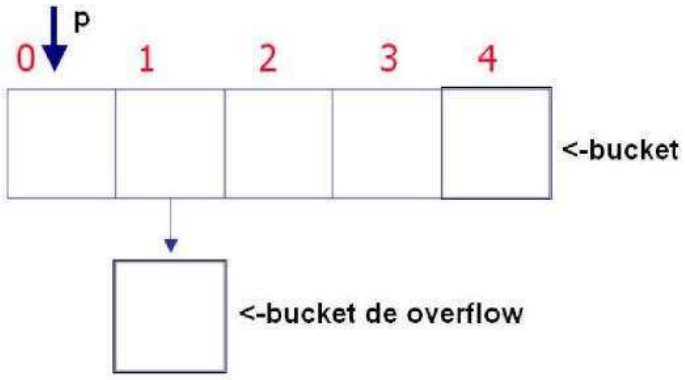
Esse modelo é conhecido como *hashing* externo. O *bucket* é o espaço de endereços de destino, é um bloco de disco ou um cluster de bloco de disco contíguos. Uma função mapeia uma chave no bucket relativo e não no absoluto. Até aqui o endereçamento é estático, não é possível crescer ou diminuir o tamanho do espaço de endereçamento do hash. Os próximos modelos estão baseados na criação de buckets de forma dinâmica.

A próxima opção é usar o ***hashing* extensível** que armazena uma estrutura de acesso fora do arquivo, semelhante a indexação. O *hash* extensível usa um diretório dinâmico de registros que armazena uma tabela, onde cada registro contém um ponteiro para um balde (tabela que armazena os registros) e cada balde tem um número fixo de itens.

A ideia é usar um diretório de ponteiros para baldes e duplicar o número de baldes através da duplicação do diretório, particionando justamente o balde que transbordou. Veja a figura abaixo:



Outra opção seria utilizar o *hashing* linear que não requer estruturas adicionais. Essa abordagem usa *buckets* que são estruturas com espaços para mais de um registro, equivale a baldes alocados em uma estrutura sequencial. Cada *bucket* possui um tamanho fixo M, ou seja, armazena M chaves. Inicia-se a estrutura com um número fixo de *buckets*. Caso o espaço do *bucket* estoure, é utilizado um *bucket* de overflow que pode ser implementado como uma lista encadeada. Um ponteiro p determina qual o *bucket* a ser duplicado. Veja a figura abaixo para entender melhor:



A criação de novo *bucket* é determinado pelo fator de carga da estrutura. Ou seja, vários *buckets* podem conter mais registros do que o espaço determinado e fazer uso de *buckets* de overflow. A cada inserção, é verificado o fator de carga. Caso seja maior do que um limite estipulado, um novo espaço de alocação é determinado.

Existem ainda outras organizações de arquivos primárias. Temos os **arquivos de registro misto** que possui diferentes tipos de registros e as **B-tree** que são árvores balanceadas projetadas para trabalhar com dispositivos de armazenamento secundário como discos magnéticos.

Considerando a forma como o arquivo está organizado, o sistema de arquivos pode recuperar registros de diferentes maneiras. Antigamente, o acesso só poderia ser feito de modo **sequencial**, neste caso, os sistemas operacionais só armazenavam arquivos em fitas magnéticas, com isso, o acesso era restrito a leitura dos registros na ordem em que eram gravados e a gravação de novos registros só era possível de ser feita no final do arquivo. Este tipo de acesso, chamado de **acesso sequencial**, é próprio da fita magnética que, como meio de armazenamento, possuía esta limitação.

O próximo método seria o **acesso direto**. Com o advento dos discos magnéticos, foi possível a introdução de métodos de acesso mais eficientes. O primeiro foi o acesso direto, que permite a leitura/gravação de um registro diretamente na sua posição. Este método é realizado através do número do registro que é a sua posição relativa em relação ao início do arquivo. É importante ressaltar que o acesso direto somente é possível quando o arquivo é definido com **registros de tamanho fixo**.

Temos também o **acesso indexado**. Esse método de acesso mais sofisticado, que tem como base o acesso direto, é o chamado acesso indexado ou acesso por chave. Para este acesso, o arquivo deve possuir uma área de índice onde existam ponteiros para os diversos registros. Sempre que a aplicação desejar acessar um registro, deverá ser especificada uma chave através do qual o sistema pesquisará na área de índice o ponteiro correspondente.

Após apresentar essa rápida introdução as organizações de arquivos e métodos de acesso, nós vamos seguir em frente falando sobre RAID e novos sistemas de armazenamento.

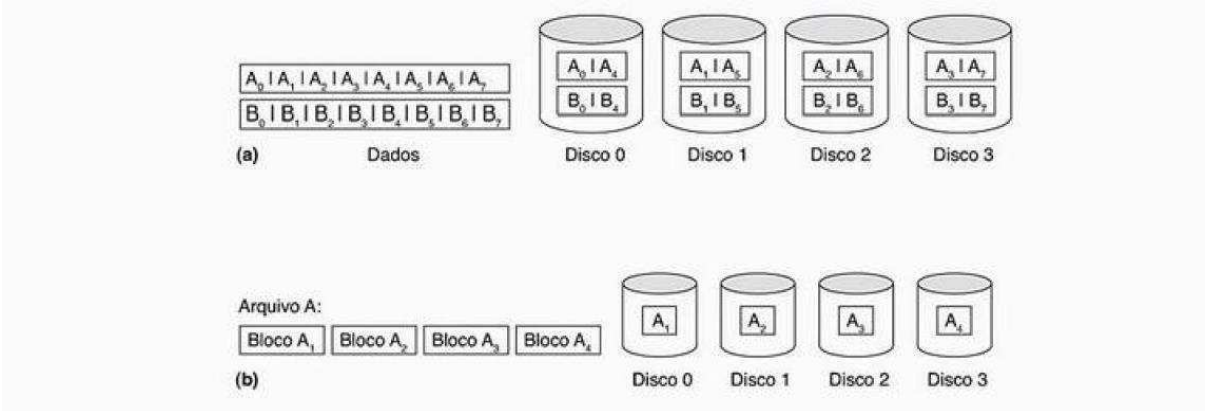
1.4. RAID – Redundant Array of Independent Disks

Focado em paralelizar o acesso aos discos o RAID - *Redundant Arrays of Inexpensive (Independent) Disks* – traz consigo duas principais vantagens para a sua implementação: **maior confiabilidade** e **melhor desempenho**. Já vimos que o gargalo no processamento de dados acontece na maioria das vezes por conta do tempo de E/S. RAID veio nivelar as diferentes taxas de melhoria de desempenho do disco contra as da memória e dos microprocessadores.

Ele possui implementações em hardware e software. Nas implementações em hardware as vantagens são o fato de ser transparente para o sistema operacional e não utilizar recursos do processador. Em implementações por software, o sistema operacional gerencia o RAID através da controladora de discos, sem a necessidade de um controlador de RAIDs, o que torna a implementação mais barata.

O conceito se baseia na ideia de um grande *array* de pequenos discos independentes, que atuam como um único disco lógico maior. Para isso eles fazem o que chamamos de *striping* de dados que nada mais é do que o emprego do paralelismo para melhorar o desempenho, alocando cada parte das informações em diferentes discos de forma que elas possam ser lidas simultaneamente.

O *striping* de dados pode acontecer em diferentes níveis. *striping* em nível de bit, como o exemplo da figura abaixo, que acontece em quatro discos. Temos também o *striping* em nível de bloco, também apresentado em quatro discos. O paralelismo tem dois objetivos: balancear a carga e realizar grandes acessos em paralelo. Esse paralelismo, portando, influência diretamente no desempenho.



Outro ponto importante é a confiabilidade que é uma consequência da redundância. O espelhamento ou sombreamento dos dados vai duplicar a informação. Neste caso, você teria os dados em diferentes discos e poderia também paralelizar a leitura. O ponto importante aqui é a resiliência do sistema quando um disco falha. Como existe o espelhamento, as requisições são repassadas para o disco que continua ativo. O que acaba acontecendo nestes casos é uma queda de performance por falta do paralelismo, que pode ser retomando assim que o disco for restaurado.

Outra possibilidade que garante a confiabilidade é a utilização do código de *Hamming* ou bits de paridade. A ideia é conseguir recuperar um ou mais discos em uma composição de RAID por meio das informações contidas nos demais discos que continuam ativos. A forma pela qual os bits de paridade estão divididos entre os discos é uma das propriedades que definem a classificação do RAID em diferentes níveis.

1.4.1. Níveis de RAID

Vamos tratar primeiramente dos níveis sete níveis de RAID mais comuns na literatura. Vejamos cada um deles na lista abaixo:

RAID 0 - também conhecido como *striping* de disco, é uma técnica que divide um arquivo e distribui os dados em todas as unidades de disco em um grupo RAID. **Não** existe nenhum espelhamento ou controle de paridade. Neste caso todos os discos funcionam como apenas um, multiplicando a performance geral pelo número de discos utilizados no conjunto (desde que o sistema operacional ofereça suporte), unicamente com o objetivo de aumentar o desempenho.

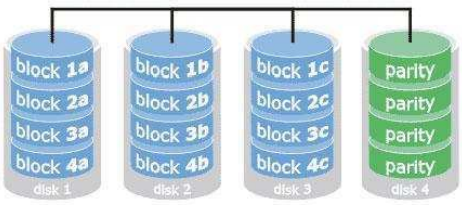
RAID 1 – Conhecido como discos espelhados. É utilizado quando a confiabilidade dos dados gravados é a maior preocupação, utiliza no **mínimo dois discos** e basicamente copia os dados de um em outro. Possui uma performance geral menor em comparação a um disco comum e não utiliza paridade.

RAID 2 - é um tipo de solução de armazenamento que surgiu no final dos anos 1980. Naquela época e nos anos seguintes, os HDs não tinham o mesmo padrão de confiabilidade que têm hoje. Por este motivo, foi criado o **RAID 2**. Ele é, até certo ponto, parecido com o RAID 0, mas conta com um mecanismo de detecção de falhas do tipo **ECC (*Error Correcting Code*)**. Hoje, este nível quase não é mais utilizado, uma vez que praticamente todos os HDs já possuem o referido recurso implementado internamente.

RAID 3 - reserva uma unidade de armazenamento apenas para guardar as informações de paridade, razão pela qual são necessários **pelo menos três discos** para montar o sistema. A paridade é intercalada por bit. Este nível

também pode apresentar maior complexidade de implementação pelo fato de as operações de escrita e leitura de dados considerarem todos os discos em vez de tratá-los individualmente. Vejam um exemplo de RAID 3 na figura a seguir.

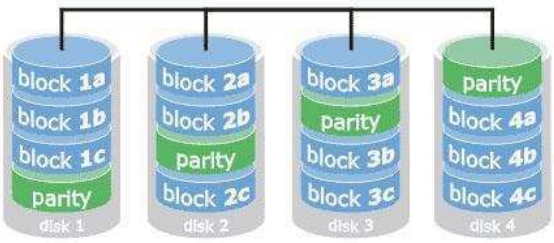
RAID 3



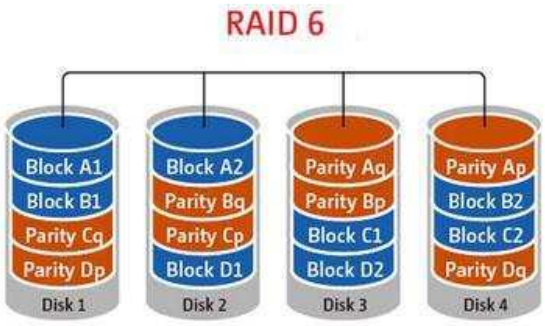
RAID 4 - também utiliza o esquema de paridade, tendo funcionamento similar ao RAID 3, com o diferencial de dividir os dados em blocos maiores e de oferecer acesso individual a cada disco do sistema. Neste caso a paridade é intercalada por bloco de disco. Este nível pode apresentar algum comprometimento de desempenho, pois toda e qualquer operação de gravação exige atualização na unidade de paridade. Por este motivo, seu uso é mais indicado em sistemas que priorizam a leitura de dados, ou seja, que realizam muito mais consultas do que gravação.

RAID 5 - utiliza paridade para a verificação de dados em todos os discos utilizados, sendo muito parecido com o RAID 0, mas com tolerância a falhas devido à utilização de um ECC (Error Correcting Code). É umas das opções mais vantajosas, mas tem uma implementação difícil. Veja a distribuição da paridade entre os diferentes discos na figura abaixo.

RAID 5



RAID 6 - se aplica ao chamado esquema de redundância P + Q usando código de Reed-Soloman para proteger contra até duas falhas de discos simultâneas. Trata-se de uma especificação mais recente e parecida com o RAID 5, mas com uma importante diferença: trabalha com dois bits de paridade. Observem a figura a seguir com uma abstração da implementação de RAID 6.

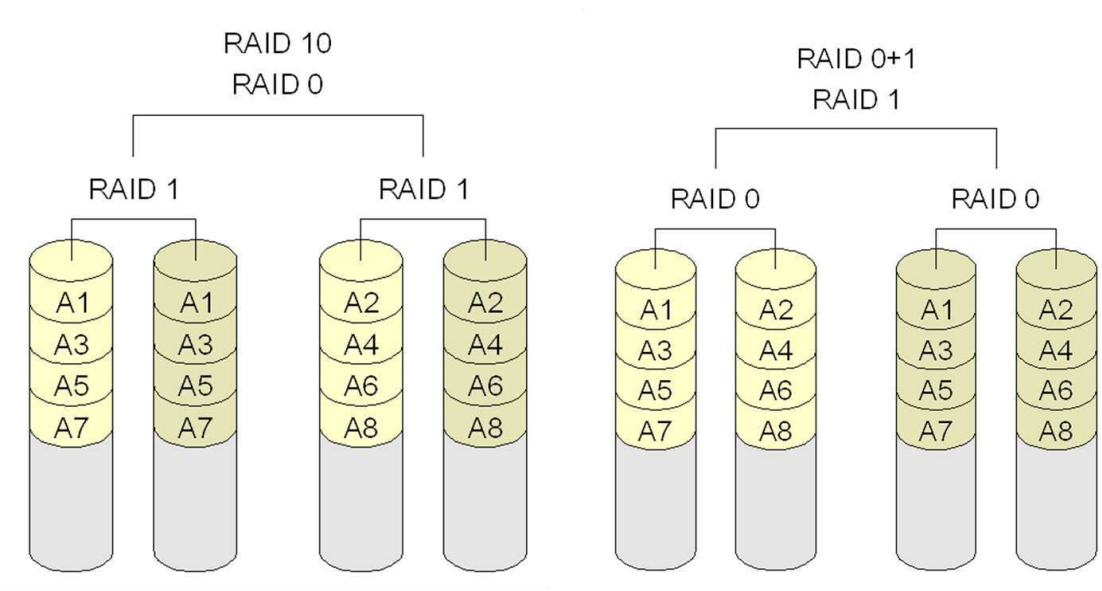


1.4.2. Nested RAID

Existe ainda a possibilidade de unir os conceitos das diferentes implementações de RAID listadas anteriormente. As possibilidades mais conhecidas são RAID 0+1 e RAID 10.

Tal como você já deve ter imaginado, o nível RAID 0+1 é um sistema "híbrido" (hybrid RAID), ou seja, que combina RAID 0 com RAID 1. Para isso, o sistema precisa ter pelo menos quatro unidades de armazenamento, duas para cada nível. Assim, tem-se uma solução RAID que considera tanto o aspecto do desempenho quanto o da redundância.

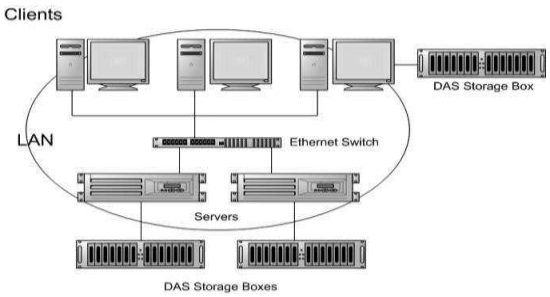
Há uma variação chamada RAID 10 (ou RAID 1+0) de funcionamento semelhante. A diferença essencial é que, no RAID 0+1, o sistema se transforma em RAID 0 em caso de falha, no RAID 1+0, o sistema assume o nível RAID 1. Vejam na figura abaixo um exemplo da implementação das duas possibilidades tratadas aqui.



1.5. Novos sistemas de armazenamento

Passaremos agora a análise dos dispositivos de armazenamentos. Nossa intenção é apresentar uma descrição sucinta para os seguintes termos: DAS, SAN, iSCSI, NAS.

Nosso estudo começa no DAS (*Direct Attached Storage).* São os primeiros dispositivos de armazenamento e suas características servirão de base para a comparação com os demais sistemas de armazenamento. Suas unidades de armazenamento são os blocos de disco. Um dos seus problemas é a conectividade, considerada a principal limitação do DAS. Entre o host e o dispositivo não há elementos de rede (como hub, switches). Não atua como servidor, é um conjunto de HDs acessado apenas por uma ou várias máquinas (desde que o dispositivo possua várias portas). Os principais protocolos/barramentos usados pelo DAS são ATA, SATA, USB, Firewire, eSATA, SCSI, SAS, Fibre Channel. Observe a figura abaixo para entender um exemplo da utilização do DAS.



Outro conhecimento importante para o entendimento do assunto é a definição de NFS. Em sistema de arquivos distribuídos podemos compartilhar arquivos e diretórios entre computadores conectados em rede. É importante tornar o acesso remoto transparente para o usuário, por exemplo, disponibilizando as áreas de trabalho dos usuários em toda a rede.

O NFS é um serviço de rede que permite o compartilhamento transparente de sistemas de arquivos ou diretórios entre os nós de uma rede. Permite que os administradores criem sistemas de arquivo centralizados que facilitam tarefas de gerência tais como manutenção e suporte. Em outras palavras, o NFS é um mecanismo que permite você montar um disco de uma máquina remota na sua, usando TCP/IP ou outro meio de transporte. Vejam que o cliente não precisa ter disco local. É possível ainda rodar NFS em redes de longa distância.

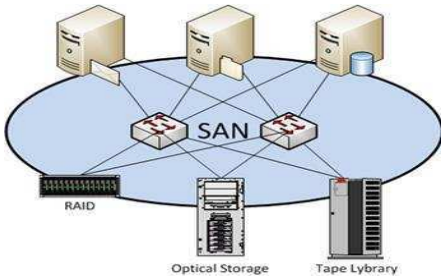
Passaremos agora aos conceitos de NAS, conhecido armazenamento conectado à rede, são servidores que não oferecem quaisquer dos serviços comuns a um servidor, mas simplesmente permitem o acréscimo de armazenamento para compartilhamento de arquivos. Conhecidos também como appliances.

Um NAS pode armazenar quaisquer dados que apareçam na forma de arquivos, como caixas de e-mail, conteúdo Web, backups de sistema remoto e

assim por diante. Alto grau de escalabilidade, confiabilidade, flexibilidade e desempenho. NAS possui maior independência do sistema operacional do cliente em comparação com os servidores de arquivo dos sistemas operacionais.

O próximo tipo de armazenamento que seria a rede de área de armazenamento (*SAN - Storage Attachment Network*). Em uma SAN, os periféricos de armazenamento on-line são configurados como nós em uma rede de alta velocidade e podem ser conectados e desconectados dos servidores de uma maneira bastante flexível.

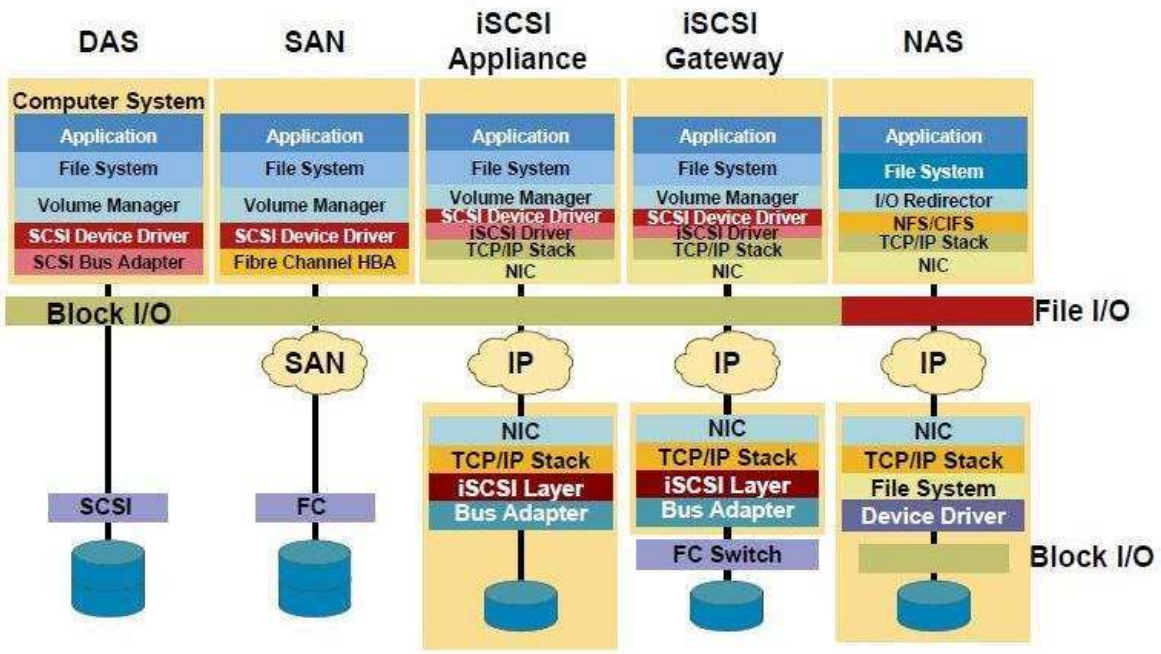
Várias empresas têm surgido como provedores de SAN e fornecem as próprias topologias proprietárias. Elas permitem que os sistemas de armazenamento sejam colocados a distâncias maiores dos servidores e oferecem diferentes opções de desempenho e conectividades. Para implementar uma SAN são necessários: um switch SAN, um dispositivo de armazenamento e um servidor. Vejam um exemplo de uma rede SAN na figura abaixo:



Vamos falar agora dos sistemas de armazenamento que usam *iSCSI*. O *iSCSI* é um protocolo que permite que os clientes enviem comandos *SCSI* para dispositivos de armazenamento SCSI em canais remotos. Ele não exige cabeamento especial por meio de *Fibber Channel* pois usa o protocolo IP.

Nota: Fibre channel protocol – FCP é padrão desenvolvido especialmente para uso de unidades remotas de armazenamento e de alta velocidade.

Falamos sobre os padrões de dispositivos de armazenamento que interessam para concursos públicos. A figura a seguir mostra as camadas presentes na arquitetura de cada um desses padrões. Em seguida, partimos para tratar do assunto de indexação de arquivos.



Estrutura de indexação para arquivo

Aqui faremos uma análise da estrutura de indexação usada dentro do contexto de banco de dados.

2. Índices

O primeiro conceito que devemos ter em mente do assunto é a definição de índices. Um índice é um mecanismo utilizado para melhorar a velocidade de acesso aos dados. Ele é composto por uma chave que é um atributo ou conjunto de atributos usado para procurar registros em um arquivo. Ele também possui um ponteiro que consiste em um identificador para um bloco de disco, além do deslocamento dentro do bloco para encontrar o registro.

Um arquivo de índice consiste em um conjunto de registros com o formato apresentado na figura abaixo. Esse registro se dá o nome de registro de índice ou entrada de índice.

Como uma técnica para criar estruturas de dados auxiliares, os índices agilizam a busca e a recuperação de registros, para isso eles envolvem armazenamento de dados auxiliares. Esses dados são armazenados nos arquivos de índices. Alguns tipos de acesso podem se beneficiar dos índices, por exemplo,

a localização de registro com um valor especificado e a localização de registros em um intervalo especificado de valores.

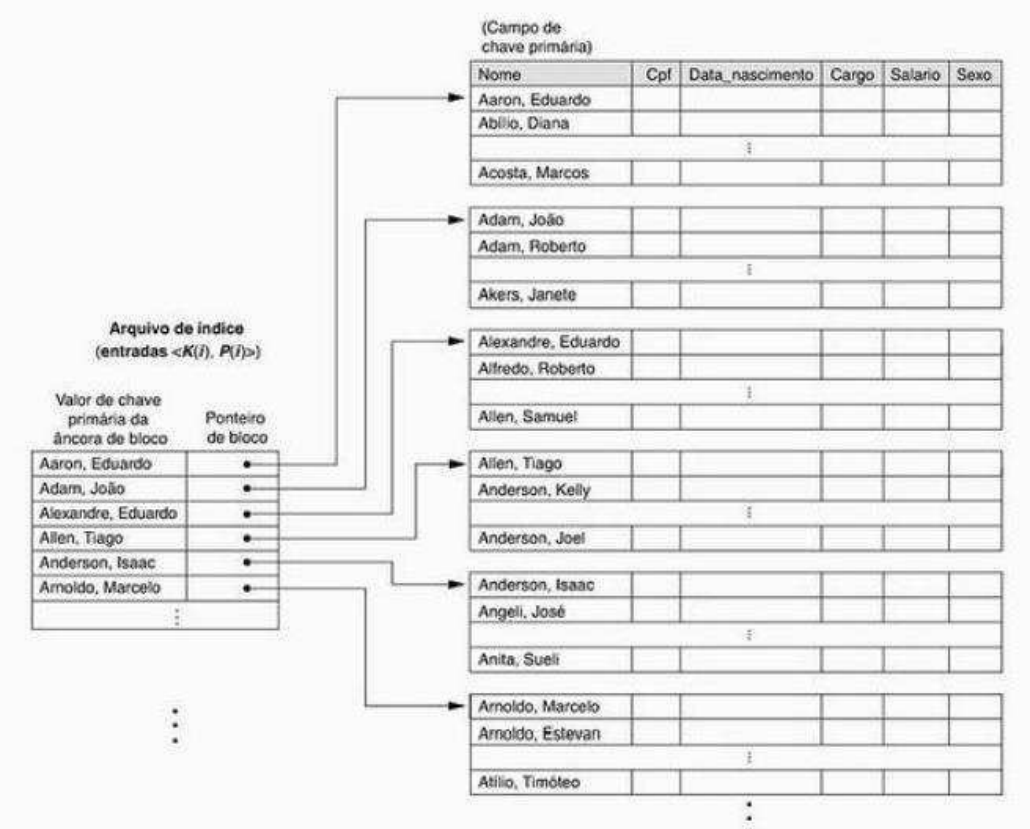
É necessário saber se esse benefício de fato traz um ganho de desempenho ao sistema de banco de dados. Avaliar os índices para medir seus efeitos na performance é um passo importante. Vários SGBDs possuem utilitários que ajudam a quantificar os efeitos pretendidos com a criação de índices sobre tabelas. Esse valor é baseado em alguns fatores, entre eles, o tempo de acesso, inserção e deleção; o overhead de espaço em disco e os métodos de acesso suportados.

Os **arquivos de índices** são geralmente muito menores que os arquivos originais. Dois tipos de índices são mais conhecidos. Os índices ordenados onde as chaves de busca são armazenadas de forma ordenada. E os índices hash, neles as chaves são distribuídas uniformemente através de "bucktes" usando uma função hash.

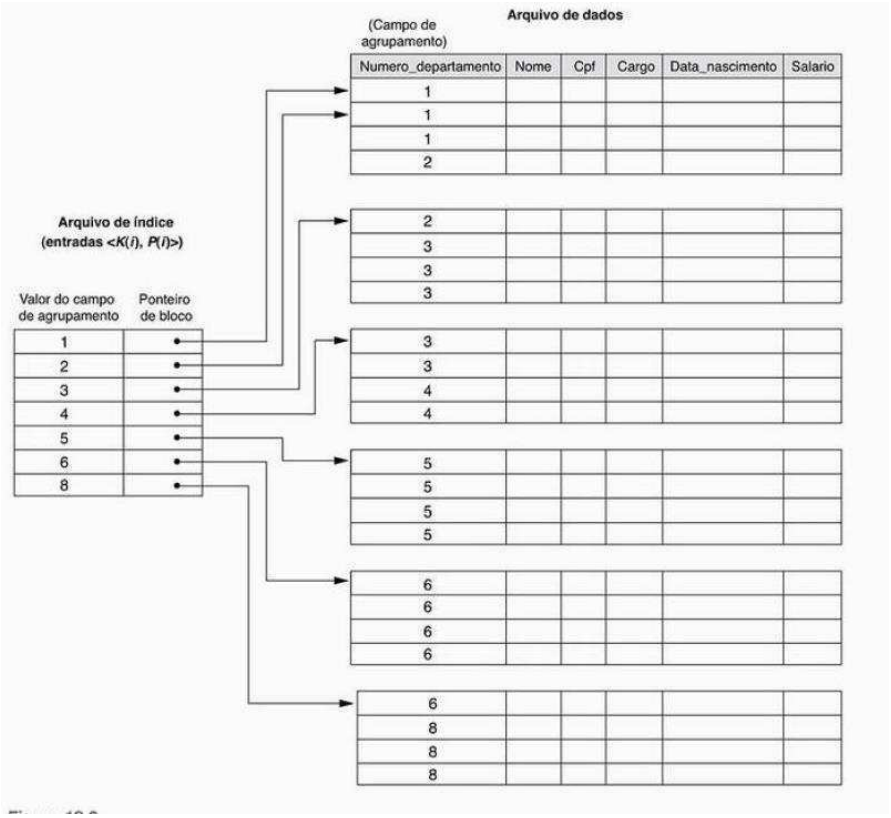
Vamos falar um pouco mais sobre a classificação dos índices. Num primeiro momento focaremos nos **índices ordenados de único nível.** Eles são semelhantes aos índices remissivos que aparecem ao final de livros técnicos. O arquivo de índice deve conter o campo de índice ou atributo de indexação. A primeira classificação para índices divide os índices em três grupos: os **primários** que usam a chave de ordenação do arquivo, os de ***clustering*** que são definidos para atributos não chave (arquivo agrupado), e os secundários sobre campos não chave primária do arquivo.

Os **índices primários** possuem uma entrada de índice ou registro de índice no arquivo de índice para **cada bloco** no arquivo de dados. Cada entrada de índice tem um valor do campo da chave primária para o primeiro registro em um bloco e um ponteiro para esse bloco. Veja que o arquivo está ordenado pelo menos atributos do índice. Imagine que os nomes Flavia e Vinicius aparecem no arquivo de índice, o primeiro apontando para o bloco 01 e o outro apontando para o bloco dois. Eu não preciso da entrada Thiago no arquivo de índice, mas se eu fizer uma busca por Thiago eu sei que ele está no mesmo bloco de Flavia. Esse processo de ordenação sequencial do arquivo, leva a uma ocupação de menos espaço na memória.

Os problemas surgem na inserção e remoção, principalmente quando temos de fazer reorganização dos registros nos blocos de disco. Quando isso ocorre temos que reajustar também o arquivo de índices. Vejam abaixo um exemplo de índices primários. O primeiro registro de cada bloco é chamado de **registro âncora** ou âncora do bloco.

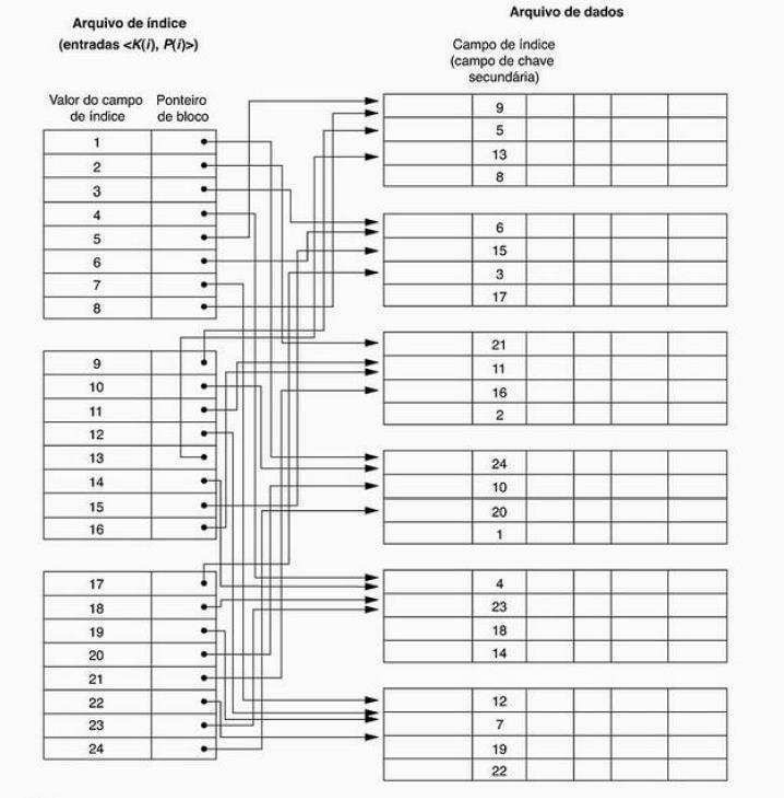


Os índices de ***clustering*** agilizam a recuperação de todos os registros que tem o mesmo valor para o campo de agrupamento. Existe, neste casso, uma entrada para cada valor distinto do campo de agrupamento, juntamente com um ponteiro para o primeiro bloco no arquivo com o valor. Semelhantemente aos índices primários, eles têm problemas com a inserção e remoção de registros. Neste caso é sempre necessário avaliar possíveis mudanças de blocos. Uma solução é reservar espaço para expansão dentro de cada bloco. A figura a seguir apresenta a distribuição de um índice de clustering.

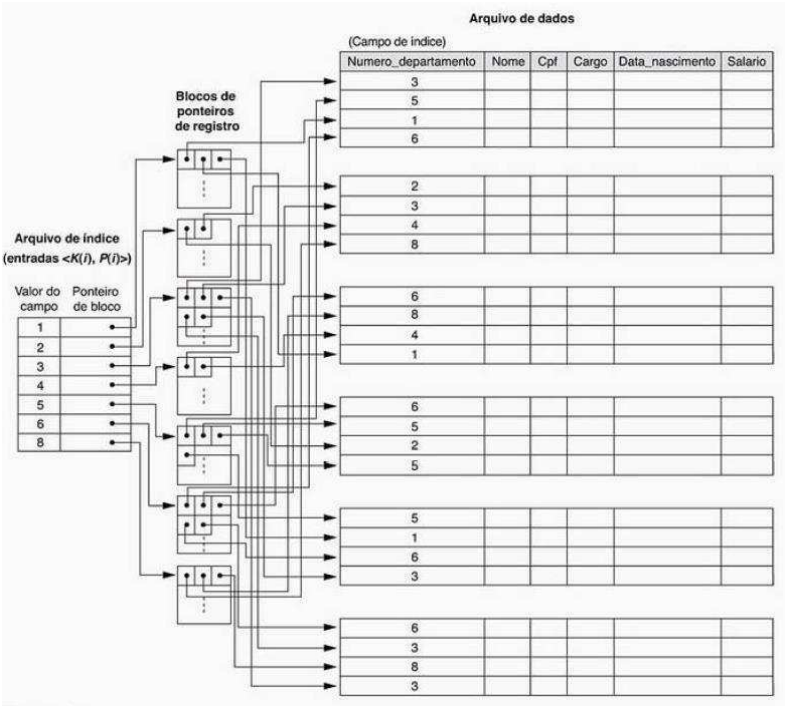


Após observar os índices primários e de *clustering*, fica mais fácil entendermos outra classificação de índices que os divide em **densos** ou **esparsos**. Um índice é dito **denso** quando existe uma entrada de índice no arquivo para cada valor de chave de pesquisa e, portanto, para cada registro no arquivo. Definimos um índice como **esparso**, quando temos entradas de índices apenas para alguns valores de pesquisa. Os índices primários e de *clustering* são categorizados como esparsos.

Mas professor, e os índices densos? Você não vai mostrar nenhum exemplo!? Claro que vou ... vamos falar sobre eles agora! Mais especificamente sobre os **índices secundários** que podem ser criados em campos que são chaves candidatas ou não. Novamente é um arquivo ordenado com dois campos <campo índice, ponteiro para o bloco ou registro>. Neste caso não podemos usar âncoras de bloco, pois o arquivo de registros não está ordenado pelo valor do índice. Todos os valores possíveis de busca devem estar no índice! Ele, portanto, usa um espaço maior de armazenamento que o primário e a sua ordenação é lógica e não física. Veja um exemplo na figura a seguir:



Quando o índice não é uma chave candidata, podemos ter mais de um registro para cada valor de entrada no arquivo de índices. Desta forma, precisamos de algum artifício para termos ponteiros ou referências para todos os registros associados a um determinado índice. Uma solução é a criação de um bucket ou bloco de ponteiros intermediário que tem a lista de todas as ocorrências. Veja a figura para entender melhor o que estou explicando.

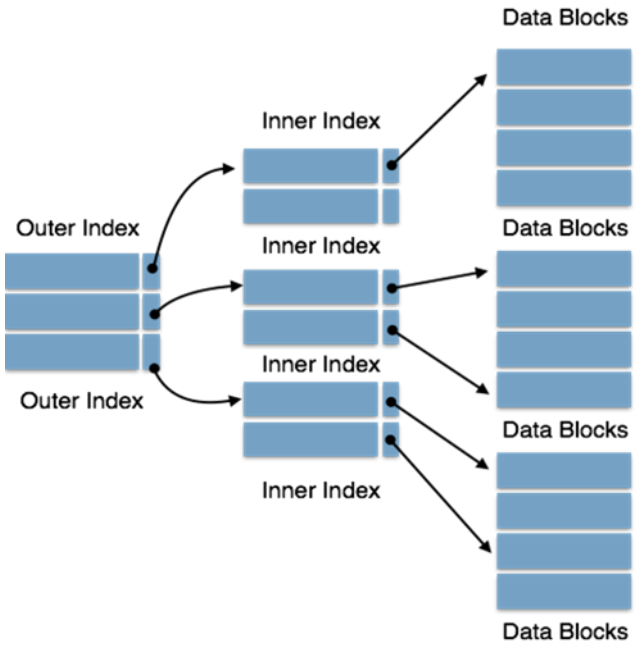


Para finalizar, apresentamos abaixo uma lista com as características de cada um dos índices discutidos até o momento.

Tipo de Índice	Número de entradas de índice	Denso ou não denso	Ancoragem de bloco no arquivo de dados
Primário	Número de blocos no arquivo de dados	Não denso	Sim
Clustering	Número de valores de campo de índice distintos	Não denso	Sim/não
Secundário (chave)	Número de registros no arquivo de dados	Denso	Não
Secundário (não chave)	Número de registros ou número de valores do campo de índices distintos	Denso ou não denso	Não

3. Índices multiníveis

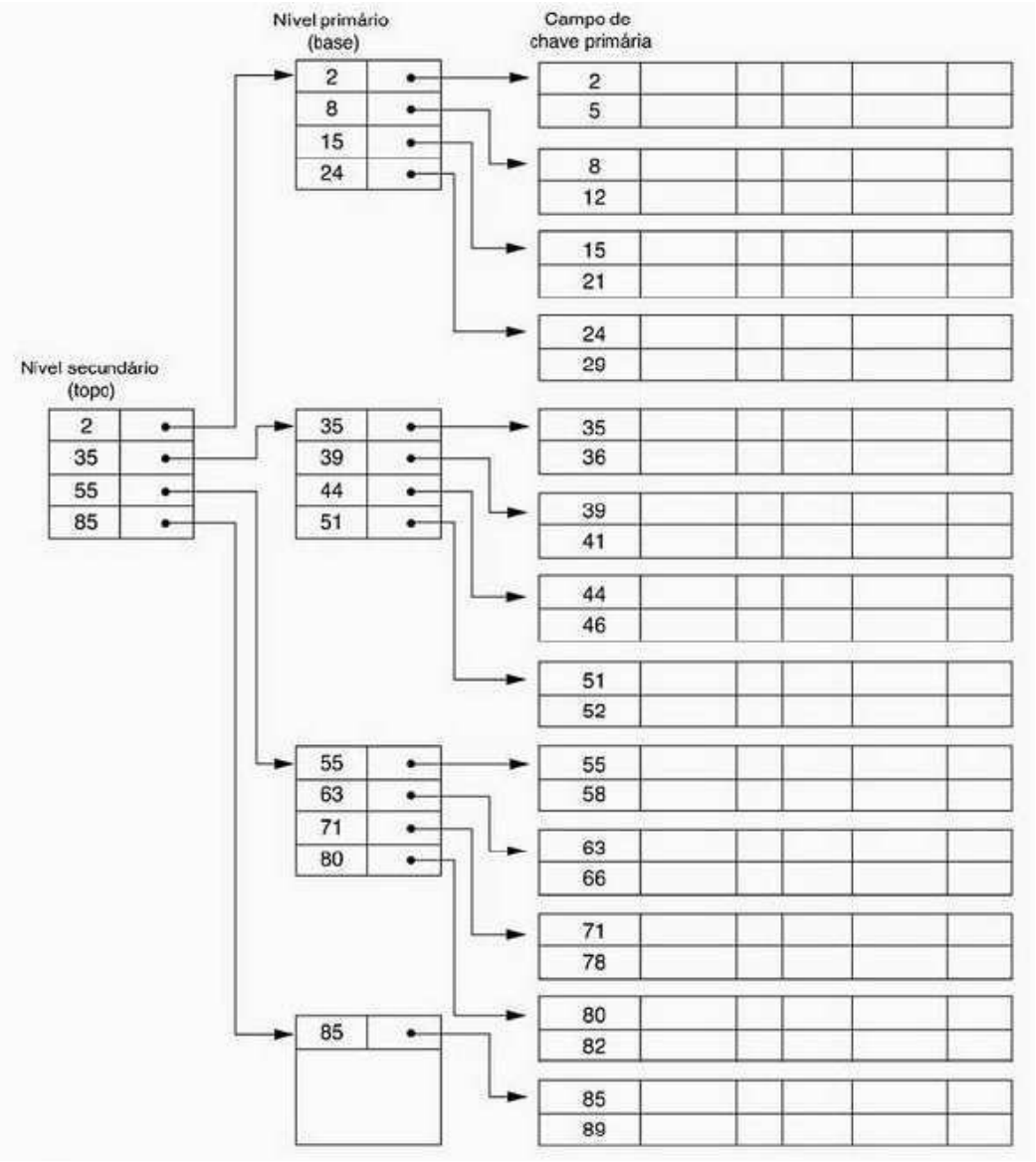
Para falarmos dos índices mulitníveis vamos começar recorrendo a uma imagem que está apresentada abaixo.



Vejam que temos dois tipos de arquivos de índices, os internos e os externos. A ideia é diminuir a quantidade de transferências de blocos de índices entre a memória e o disco. Neste contexto criamos uma estrutura de árvore que começa como os índices externos, segue por meio dos índices internos e o

último nível do índice interno aponta para os blocos de dados propriamente ditos. Como esse contexto conseguimos entender o objetivo dessa hierarquização dos índices. Reduzir a parte do índice que continuamos a pesquisar por bfr_i, o fator de bloco para o índice.

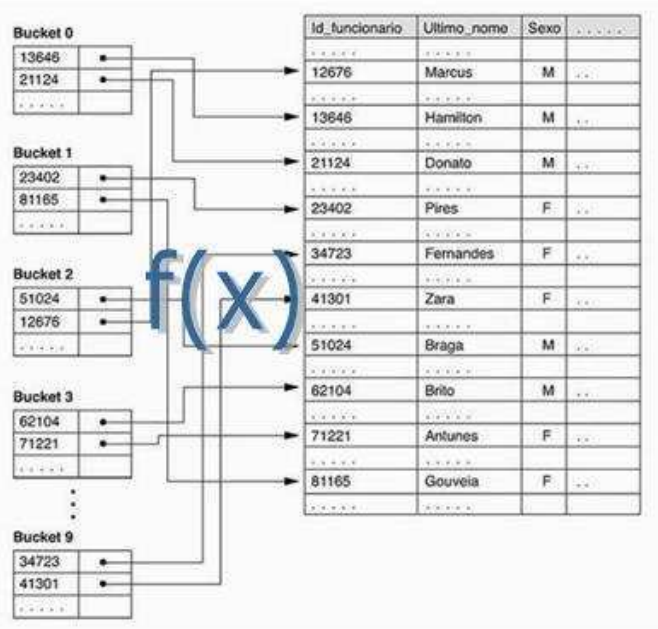
O primeiro nível do arquivo de índice é conhecido como **base**. Ele é ordenado com um valor distinto para cada ki. Só exigimos um nível subsequente se o nível anterior não couber em n blocos de disco. O último nível é conhecido como topo. Vejam um exemplo de índice multinível na figura abaixo.



4. Outros tipos de índices

Vamos tratar agora de outros tipos de índices. Primeiramente faladaremos dos índices de hash. Um **índice de hash** consiste em uma coleção de buckets organizados em uma matriz. Uma função de hash mapeia chaves de índice para buckets correspondentes no índice de hash. A figura a seguir mostra três

chaves de índice mapeadas para três *buckets* diferentes no índice de *hash*. Para fins ilustrativos, o nome da função de *hash* é f(x).



Outro índice interessante usa o que chamamos de mapa de bits. O **índice de bitmap** é um tipo de índice que pode ser utilizado para otimizar consultas que utilizam como filtro de dados, colunas que possuem baixa cardinalidade, ou seja, colunas que possuem pouca variação de valores nas linhas de uma tabela. Ao criar um índice bitmap em uma coluna, o SGBD monta um mapa de bits para todas as linhas da tabela, contendo todos os valores possíveis para a coluna. Para cada linha há um mapa de todos os valores possíveis da coluna indexada. O SGBD então grava um bit 1 onde o valor existe em uma determinada linha e 0 para os valores que não existem nesta linha. Vejam o exemplo na figura abaixo.

Funcionario

Linha_id	Func_id	Unome	Sexo	Cep	Faixa_salarial
0	51024	Braga	M	09404011	..
1	23402	Pires	F	03002211	..
2	62104	Brito	M	01904611	..
3	34723	Fernandes	F	03002211	..
4	81165	Gouveia	F	01904611	..
5	13646	Hamilton	M	01904611	..
6	12676	Marcus	M	03002211	..
7	41301	Zara	F	09404011	..

Índice bitmap para Sexo

M	F
10100110	01011001

Índice bitmap para Cep

Cep 01904655	Cep 03002211	Cep 09409433
00101100	01010010	100000001

Temos ainda a indexação baseada em função. O algoritmo do otimizador dos SGBD comerciais desconsidera, na montagem do plano de acesso, o índice associado a uma coluna chave referenciada em uma função como, por exemplo, TO_CHAR ou TO_NUMBER. Para suplantar essa limitação, os índices B-tree ou de bitmap podem ser criados com a inclusão de uma função – padrão SQL ou construída pelo desenvolvedor – ou mesmo uma expressão aritmética, conforme a sentença de criação dos índices abaixo.

```
CREATE INDEX idx_maiusc ON Funcionario
(UPPER(Unome));

CREATE INDEX idx_renda
ON Funcionario(Salario + (Salario*pct_Comissao));
```

O nosso assunto teórico está concluído, vamos logo a seguir passar para o comentário das questões sobre os temas discutidos durante a aula. Antes, porém, gostaria de fazer um rápido comentário sobre indexação de textos.

INDEXAÇÃO DE TEXTOS

Existem dois métodos principais de busca por frases em bancos de dados textuais utilizando indexação de textos, um é **arquivo invertidos com contadores de posição** e o outro é **índice para a próxima palavra**.

Um arquivo invertido possui duas partes principais: uma estrutura de busca, chamada de **vocabulário**, contendo todos os termos distintos existentes nos textos indexados e, para cada termo, uma lista invertida que armazena os identificadores dos registros contendo o termo. Consultas são feitas tomando-se a lista invertida correspondente ao termo procurado. As consultas booleanas são feitas obtendo-se a conjunção ou disjunção entre as listas relativas aos termos presentes na consulta.

Os índices para a próxima palavra apresentam uma abordagem mais eficiente do que o uso de arquivos invertidos com contadores de posição. Nessa abordagem, para cada palavra existente no **vocabulário** é criado uma **lista com as palavras** que ocorrem em uma **posição subsequente** no texto, justamente com apontadores de posição para essas ocorrências.

Questões Comentadas

Vamos continuar nosso estudo de fazendo algumas questões sobre o assunto. Sempre que possível vamos inserir algum detalhamento teórico na explicação da questão. Esperamos que vocês gostem. Nesta aula optamos por colocar as questões apenas ao final devido à quantidade escassa de questões disponíveis.

2. ANO: 2015 BANCA: COSEAC ÓRGÃO: UFF PROVA: ANALISTA - TECNOLOGIA DA INFORMAÇÃO

Em relação aos métodos de acesso a arquivos, é correto afirmar que:

A as fitas magnéticas implementam o método de acesso indexado.

B o método de acesso direto possui uma restrição à ordem em que os registros são lidos ou gravados.

C o método de acesso direto trabalha com registros de tamanho variável.

D o método de acesso indexado tem por base o método de acesso sequencial.

E o método de acesso direto pode ser combinado com o método de acesso sequencial.

Comentários: Os Métodos de acesso são os procedimentos empregados pelo gerenciador de banco de dados com o objetivo de acelerar a localização e a recuperação de algum dado. Analisando cada uma das alternativas acima podemos encontrar os seguintes erros em cada uma das letras.

A. As fitas, como falamos durante a aula, possuem métodos de acesso sequencial, e não indexado como sugere a questão.

B. O método de acesso direto consiste na gravação dos registros em endereços determinados com base no valor de uma chave primária (por exemplo), de modo que se tenha acesso rápido aos registros especificados por argumentos de pesquisa, sem que haja necessidade de percorrer uma estrutura auxiliar (índice).

C. Vimos durante a aula que o acesso direto trabalha com registros de **tamanho fixo**.

D. O acesso indexado tem por base a utilização de índice.

A alternativa E está correta e apresenta a nossa resposta. A ideia é que você use uma função *hash* para apontar para um determinando endereço e em seguida siga por meio do bloco de disco até achar o valor correspondente. Trata de uma função *hash* com endereçamento aberto.

Gabarito: E

CAIU na prova!

3. ANO: 2015 BANCA: NUCEPE ÓRGÃO: SEFAZ - PI PROVA: ANALISTA - SISTEMAS JÚNIOR

Um Sistema Gerenciador de Banco de Dados (SGBD) permite criar e manipular o banco de dados. Entre as alternativas abaixo, qual está INCORRETA sobre o SGBD?

A Garantem restrições de integridade.

B Tem suporte a controle de concorrência.

C A abstração é suportada para os níveis de visão do usuário e conceitual, sendo o nível físico negligenciado e de responsabilidade somente do sistema operacional.

D Permite controle de acesso.

E Otimiza as consultas por métodos de acessos eficientes.

Comentários: Vejam que estamos procurando a alternativa incorreta, desta forma, vamos analisar apenas a alternativa que temos como resposta. Sabemos que temos abstração de nível conceitual, lógico e físico. O SGBD trabalha na implantação dos detalhes lógicos e físicos. Aspectos de controle da operação de entrada e saída, atividade típica do sistema operacional, podem inclusive serem repassadas ao controle do SGBD.

Gabarito: C

CAIU na prova!

4. ANO: 2012 BANCA: FCC ÓRGÃO: TRE-SP PROVA: TÉCNICO DO JUDICIÁRIO - PROGRAMADOR DE SISTEMAS

Em SGBDs,

A os metadados resultam da conversão de comandos DDL pelo compilador da DDL.

B mapeamentos, restrições de integridade, mensagens de comunicação e restrições de segurança são recursos contidos no dicionário de dados.

C o arquivo de dados é o componente que cuida da alocação do espaço na armazenagem no disco e das estruturas de dados usadas para representar a informação armazenada.

D a estrutura de armazenagem e os métodos de acesso são especificados por um conjunto de definições em um tipo especial de DML.

E a abstração dos níveis físico, conceitual e de visão aplica-se, exclusivamente, à definição e estrutura de dados.

Comentários: Um esquema de banco de dados é especificado por um conjunto de definições expressas por uma linguagem especial chamada linguagem de

definição de dados (*Data Definition Language*, DDL). O resultado da compilação de comandos de uma DDL é um conjunto tabelas que são armazenadas em um arquivo especial chamado dicionário (ou diretório) de dados.

Gabarito: A

5. ANO: 2012 BANCA: CESPE ÓRGÃO: BANCO DA AMAZÔNIA PROVA: TÉCNICO CIENTÍFICO - ADMINISTRADOR DE DADOS

Acerca de mapeamento físico de dados, julgue os itens seguintes.

88 Para cada atributo usado em operações de join, deve-se criar um índice.

89 Na seleção dos métodos de acesso a registros físicos, deve-se considerar a forma como os dados serão utilizados pelas diversas aplicações.

90 Em operação de junção (join), atinge-se maior eficiência quando os registros físicos estão ordenados pelo atributo usado na junção.

Comentários: Vamos comentar cada uma das alternativas acima.

88 Não existe nenhuma obrigatoriedade de atributos usados em operações de join terem um índice associado. Há necessidade apenas dos atributos que estão nas diferentes tabelas que participam do Join operem sobre o mesmo domínio, para que a junção possa ser feita efetivamente.

89 Lembre-se do início da nossa aula, um método de acesso está diretamente ligado à organização do arquivo, não é possível usar o acesso indexado se não tivermos o arquivo de índice relacionado com o arquivo de dados.

90 Esse operação é estudada em otimização de consulta, o merge-join entre dois arquivos ordenados pelo atributo de junção é muito mais rápido. Desta forma a alternativa está correta.

Gabarito: E C C

6. ANO: 2015 BANCA: UERJ ÓRGÃO: UERJ PROVA: ANALISTA DE SISTEMAS - GRID

Técnicas eficientes para o uso de memória, como memória virtual e caching, podem ser aproveitadas pelo seguinte motivo:

A o princípio da localidade pode ser aplicado

B memórias dinâmicas são mais rápidas que memórias estáticas

C a velocidade de acesso à memória RAM aumentou muito recentemente

D o espaço de armazenamento na memória RAM aumentou muito recentemente

Comentários: O princípio da localidade pode ser dividido em localidade temporal e espacial. Na primeira, ao acessar uma palavra na memória principal é muito provável que o processador volte a acessar essa mesma palavra novamente durante a execução dos programas (loops). Na localidade espacial ao

acessar uma palavra na memória principal é provável que em seguida o processador tente acessar uma palavra de memória subjacente à acessada previamente. Desta forma, temos os exemplos de memória virtual e caching como exemplos da implementação do princípio da localidade.

Gabarito: A

7. ANO: 2015 BANCA: FCC ÓRGÃO: TRT - 15ª REGIÃO (CAMPINAS-SP) PROVA: TÉCNICO JUDICIÁRIO - TECNOLOGIA DA INFORMAÇÃO

Existem diferentes tecnologias para a construção de dispositivos de armazenamento de dados, e o uso de cada um deles depende da finalidade. Caso a finalidade seja utilizar o dispositivo de armazenamento para realizar uma cópia de segurança, a única tecnologia que NÃO pode ser utilizada é

A Memória FLASH.

B DVD.

C Fita magnética.

D Memória RAM.

E CDROM.

Comentários: Para ser considerado um dispositivo de armazenamento de dados que guarda cópias de segurança é necessário que o dispositivo possua a não volatilidade como característica. A memória RAM, por ser uma memória volátil, que apaga os dados quando o fluxo de energia para, não pode ser utilizada para esse propósito.

Gabarito: D

8. ANO: 2015 BANCA: FCC ÓRGÃO: TRE-RR PROVA: TÉCNICO DO JUDICIÁRIO - OPERAÇÃO DE COMPUTADORES

"A memória do tipo ___I___ se diferencia das memórias convencionais ___II___ por serem muito rápidas. Por outro lado, são muito mais caras.

A memória ___III___ consiste em uma quantidade menor de memória embutida no processador. Quando este precisa ler dados na memória RAM, um circuito especial transfere blocos de dados muito utilizados da RAM para esta memória. Assim, no próximo acesso do processador, este consultará esta memória, que é bem mais rápida, permitindo o processamento de dados de maneira mais eficiente".

Completam, correta e respectivamente, as lacunas de I a III do texto o que consta em:

A DRAM (Dynamic RAM) – SRAM (Static RAM) – cache.

B cache – RAM – DRAM (Dynamic RAM).

C DDR – DDR2 – DDR3.

D cache – RAM – SDRAM.

E SRAM (Static RAM) – DRAM (Dynamic RAM) – cache.

Comentários: Veja que é a segunda questão que trata do fato da velocidade das memórias SRAM ser maior que as DRAM. Esse fato nos ajuda a preencher os campos I e II.

O campo III trata da memória cache. Cache é um dispositivo de acesso rápido, interno a um sistema, que serve de intermediário entre um operador de um processo e o dispositivo de armazenamento ao qual esse operador acede. A vantagem principal na utilização de uma cache consiste em evitar o acesso ao dispositivo de armazenamento – que pode ser demorado, armazenando os dados em meios de acesso mais rápidos

O cache de disco é uma pequena quantidade de memória incluída na placa lógica do HD. Tem como principal função armazenar as últimas trilhas lidas pelo HD. Esse tipo de cache evita que a cabeça de leitura e gravação passe várias vezes pela mesma trilha, pois como os dados estão no cache, a placa lógica pode processar a verificação de integridade a partir dali, acelerando o desempenho do HD, já que o mesmo só requisita a leitura do próximo setor assim que o último setor lido seja verificado.

Gabarito: E

9. ANO: 2015 BANCA: CESPE ÓRGÃO: MEC PROVA: TÉCNICO DE NÍVEL SUPERIOR - ANALISTA DE SISTEMA OPERACIONAL

Acerca das soluções de alta disponibilidade, julgue os seguintes itens.

[1] Se um disco falhar em um sistema com configuração RAID 5, é possível recuperar os dados. Contudo, caso um segundo disco falhe antes do término da recuperação dos dados do primeiro disco defeituoso, todos os dados armazenados serão perdidos.

[2] Considerando-se que sejam utilizados quatro discos de mesma capacidade em um arranjo RAID 0 e quatro discos idênticos aos anteriores na configuração RAID 1, é correto afirmar que a capacidade de armazenamento é a mesma nos dois sistemas, havendo apenas uma diferença de velocidade entre os arranjos.

Comentários: Vamos analisar as alternativas acima:

1. A alternativa trata de RAID 5 conhecida como striping with parity across drives. O RAID 5 é o nível de RAID mais comum e mais utilizado. Para sua configuração são necessários no mínimo 3 discos de dados no Array e no máximo 16. Os dados são distribuídos entre os discos existentes no Array e os dados de paridade são espalhados por todos os discos.

Utilizando os dados de paridade, o sistema pode calcular os dados de um bloco, caso este dado não esteja mais disponível. Isso significa que o RAID 5 tem tolerância à falha de um disco no Array, sem que haja perda de dados. Embora o RAID 5 seja suportado por softwares de RAID, é recomendado uma controladora RAID para sua implementação pois muitas vezes é necessário a utilização de memória cache para melhorar o desempenho de gravação dos dados.

2. RAID 0 não temos replicação dos dados, apenas uma divisão que favorece o desempenho, mas não ajuda na tolerância a falhas. Diferentemente do RAID 1 que gasta metade dos discos com uma cópia dos dados armazenados nos demais dispositivos de armazenamento. A questão está, portanto, incorreta, pois afirma que RAID 0 e RAID 1 tem a mesma capacidade de armazenamento.

Gabarito: C E

10. ANO: 2015 BANCA: IESES ÓRGÃO: TRE-MA PROVA: ANALISTA JUDICIÁRIO - ANÁLISE DE SISTEMAS

Considere as seguintes afirmativas relativas a RAID:

I. Todos os níveis de RAID são formados por um conjunto de unidades de discos físicos, vistas pelo sistema operacional como uma única unidade lógica.

II. Os níveis de RAID 1, 3, 4, 5 e 6 contam com redundância, obtida através do armazenamento de informações de paridade e visando conseguir uma alta disponibilidade.

III. Os dados são distribuídos pelos discos físicos de um array em um esquema conhecido como intercalação de dados (striping).

Assinale a alternativa correta com relação as afirmativas:

A Somente uma está correta.

B Somente duas estão corretas.

C Todas estão corretas.

D Todas estão incorretas.

Comentários: Ok! Vamos comentar as alternativas I, II e III.

I. RAID é a sigla para Redundant Array of Independent Disks ou, em tradução livre, algo como "Matriz Redundante de Discos Independentes". Trata-se, basicamente, de uma solução computacional que combina vários discos rígidos (HDs) para formar uma única unidade lógica de armazenamento de dados.

E o que é unidade lógica? Em poucas palavras, no que se refere a RAID, **trata-se de fazer com que o sistema operacional enxergue o conjunto de HDs como uma única unidade de armazenamento**, independentemente da quantidade de dispositivos que estiver em uso. Hoje, além de HDs, é possível montar sistemas RAID baseados em SSD.

II. Apenas um contraexemplo que torna a questão errada, o RAID nível 1. Nele, uma unidade "**duplica**" a outra, isto é, faz uma "cópia" da primeira, razão pela qual o nível também é conhecido como *mirroring* (espelhamento). Com isso, se o disco principal falhar, os dados podem ser recuperados imediatamente porque existe cópias no outro. Desta forma a alternativa II está incorreta.

III. *Striping* é o processo de dividir um conjunto de dados em blocos e distribuir os blocos de dados em vários dispositivos de armazenamento, como discos rígidos ou SSDs. Esse conceito é de fato utilizado por RAID para distribuir a informação entre diferentes discos. Desta forma, a alternativa III está correta.

Gabarito: B

11. ANO: 2013 BANCA: FGV ÓRGÃO: AL-MA PROVA: TÉCNICO DE GESTÃO ADMINISTRATIVA - PROGRAMADOR DE SISTEMAS

Com relação ao sistema de armazenamento SAN (Storage Area Network), analise as afirmativas a seguir.

I. O uso de iSCSI em SANS é adequado para aplicações de alto desempenho, como banco de dados.

II. Alta segurança, redundância e tolerância a falhas são características do SAN.

III. O uso do protocolo iSCSI em SANs facilita a transferência de dados em Intranets de forma robusta e confiável.

Assinale:

A se somente a afirmativa I estiver correta.

B se somente a afirmativa II estiver correta.

C se somente a afirmativa II e III estiverem corretas.

D se somente as afirmativas I e II estiverem corretas.

E se todas as afirmativas estiverem corretas.

Comentários: Vamos analisar cada uma das alternativas abaixo:

I. O iSCSI é basicamente o protocolo SCSI encapsulado via IP, é um meio mais barato para se criar uma rede SAN, porém com menor desempenho e confiabilidade, já que na maior parte das vezes, usam-se switchs compartilhados com outros ativos de rede. Sendo assim, aplicações de alto desempenho não devem utilizar iSCSI.

II. Um *storage area network* (SAN) é uma rede de dispositivos que conectam vários clientes e dispositivos de armazenamentos. Portanto, na SAN existem vários caminhos disponíveis para o transporte de dados entre dois pontos. Esta característica de múltiplos caminhos da SAN permite flexibilidade, disponibilidade e escalabilidade.

III. Vimos no item I que iSCSI está relacionado ao protocolo IP, protocolo padrão na internet.

Gabarito: C

12. ANO: 2014 BANCA: CESPE ÓRGÃO: TJ-CE PROVA: ANALISTA JUDICIÁRIO - CIÊNCIAS DA COMPUTAÇÃO

A implantação de uma rede SAN (Storage Area Network) apresenta vantagens como o alto desempenho, a escalabilidade e a redução de custos em relação à manutenção das informações armazenadas. Com relação a esse assunto, assinale a opção correta.

A Em virtude de algumas restrições contidas nas especificações técnicas dos dispositivos de interconexão, algumas tecnologias de comunicação como iSCSI e Ethernet não são suportadas.

B A distância compreendida entre os dispositivos de armazenamentos e os hosts deve ser curta, pois distâncias superiores a 100 metros prejudicam consideravelmente o desempenho dessa tecnologia.

C O consumo de largura de banda na rede é mínimo considerando-se as operações de entrada e saída relacionadas ao armazenamento, característica que faz com que a SAN se destaque em relação às demais estruturas de armazenamento de dados.

D A recomendação da maioria dos especialistas é que a SAN seja implementada em cluster com mais de 20 nodos, o que aumenta consideravelmente o poder de escalabilidade em razão do balanceamento de carga.

E Uma das vantagens desse tipo de rede consiste na centralização do gerenciamento do armazenamento, mesmo em SAN mais complexa.

Comentários: Vejamos as características de redes SAN.

Os desempenhos do SAN estão diretamente ligados ao tipo de rede utilizado. No caso de uma rede Fibra Chanel, a banda concorrida é de cerca de 100 Mo/s (1000 Mbit/s) e pode ser estendida multiplicando as relações de acesso.

A capacidade de um SAN pode ser aumentada de maneira quase ilimitada e atingir centenas, ou mesmo milhares de terá octetos.

Graças ao SAN, é possível partilhar dados entre vários computadores da rede sem sacrificar os desempenhos, na medida em que o tráfego SAN está completamente separado do tráfego dos utilizadores. São os servidores aplicativos que desempenham o papel de interface entre a rede de dados (geralmente Fibra Chanel) e a rede dos utilizadores (geralmente Ethernet).

Por outro lado, o custo de aquisição de um SAN é muito mais elevado do que um dispositivo NAS, na medida em que se trata uma arquitetura completa, utilizando tecnologias ainda caras. Quando analisamos o custo por byte do TCO, a despesa pode justificar-se para muitas empresas.

Analisando o texto acima, podemos concluir que a alternativa **E** está correta!

Gabarito: E

13. ANO: 2015 BANCA: CESPE ÓRGÃO: MEC PROVA: TÉCNICO DE NÍVEL SUPERIOR - ANALISTA DE SISTEMA OPERACIONAL

Julgue os itens a seguir, acerca de arquiteturas e protocolos para redes de armazenamento de dados.

[1] O iSCSI é um protocolo embasado em IP que estabelece e gerencia conexões entre armazenamentos, hosts e dispositivos de ponte sobre IP, encapsulando dados e comandos SCSI para permitir que estes blocos de dados sejam transportados por meio de pacotes TCP/IP.

[2] O iSCSI é um protocolo de camada de sessão que inicia uma sessão confiável com um dispositivo que reconhece comandos SCSI e TCP/IP, sendo sua interface responsável pela manipulação de login, autenticação e gerenciamento da sessão.

[3] Se um storage array que suporte iSCSI é implantado, o próprio host pode agir como iniciador iSCSI e se comunicar diretamente com o armazenamento através de uma rede IP. Porém, em todos os casos, ao se usar o iSCSI, é necessário um componente FC (Fiber Channel).

Comentários:

Gabarito: C C E

14. ANO: 2015 BANCA: IESES ÓRGÃO: IFC-SC PROVA: INFORMÁTICA - SEGURANÇA E PROJETOS DE REDES

Em relação a implementação de redes e dispositivos do tipo SAN (Storage Area Network) o que é correto afirmar?

A Em função de requisitos de performance, alguns protocolos como iSCSI e Ethernet não podem ser utilizados.

B É uma excelente alternativa para se utilizar em unidades de armazenamento de dados distribuídas. Em função do seu baixo consumo de banda, podem ser largamente utilizados em conexões WAN.

C Trata-se de uma tecnologia que vem perdendo aplicabilidade hoje em dia. Outras tecnologias de armazenamento como DAS e NAS naturalmente a substituem.

D Apresenta vantagens como: alto desempenho, alta disponibilidade, facilidade de gerenciamento e redução de custos operacionais em relação a outras tecnologias de storage do gênero.

Comentários: Vamos analisar cada uma das alternativas abaixo**.**

A. Alternativa incorreta. Utiliza principalmente os protocolos FC (e suas variações) e iSCSI.

B. Alternativa incorreta. Seu armazenamento é centralizado, consiste em uma rede dedicada (segregada) e de alta velocidade de servidores e dispositivos de armazenamento compartilhados.

C. Alternativa incorreta. Não está perdendo aplicabilidade.

D. Correta.

Gabarito: D

15. ANO: 2015 BANCA: MP-RS ÓRGÃO: MP-RS PROVA: TÉCNICO EM INFORMÁTICA - SISTEMAS

Qual é o tipo de organização de arquivo, no qual a ideia principal é fornecer uma função de randomização que, aplicada ao valor do campo-chave de um registro, gere o endereço do bloco do disco onde o registro está armazenado?

A Overflow.

B Árvore-B *.

C Árvore-B.

D Hashing.

E Indexação linear.

Comentários: Observem que as palavras ou expressão "função de randomização" nos remete a ideia de hashing, presente na alternativa D.

Gabarito: D

16. ANO: 2014 BANCA: FCC ÓRGÃO: METRÔ-SP PROVA: ANALISTA DESENVOLVIMENTO GESTÃO JÚNIOR - CIÊNCIAS DA COMPUTAÇÃO

O estudo das estruturas de dados envolve um objetivo teórico, que procura identificar e desenvolver modelos matemáticos, determinando que classes de problemas podem ser resolvidos com o seu uso, e um objetivo prático, que busca criar representações concretas dos objetos e desenvolver rotinas capazes de atuar sobre estas representações, de acordo com o modelo considerado. Considere as definições das estruturas de dados:

I. São conhecidas como listas LIFO – Last In First Out. Uma máquina puxando vagões de trens é um exemplo de funcionamento de uma estrutura deste tipo.

II. O armazenamento de dados em Memória Secundária (MS) ou externa, se dá através da sua utilização. Os algoritmos e as estruturas de dados para processamento de dados em MS têm que considerar que o custo para se acessar um registro é algumas ordens de grandeza maior do que o custo de processamento na Memória Primária (MP) ou interna.

III. É uma forma especial de se agrupar dados, em que cada item possui uma referência para o próximo item, como se fosse uma corrente, com cada item sendo um dos elos. Costuma-se chamar esses itens de nós ou nodos.

IV. São conhecidas como listas FIFO – First In First Out. Pessoas organizadas para entrar em um trem metropolitano é um exemplo de funcionamento desta estrutura.

V. É utilizada para pesquisa em MS, quando os arquivos contêm mais registros do que a MP pode armazenar. É uma estrutura de dados utilizada para manutenção e organização de arquivos, podendo ser utilizada para armazenar e recuperar informações que estão em grandes repositórios de dados.

As estruturas definidas nos itens de I a V são, respectivamente:

A Queues – Heaps – Árvore Binária de Busca – Pilhas – Árvore AVL.

B Pilhas – Diretórios – Heap – Filas – Tabelas Hashing.

C Queues – Arquivos – Lista Duplamente Encadeada – Stacks – Heap.

D Stacks – Heaps – Lista Encadeada Circular – Queues – Tabelas Hashing.

E Pilhas – Arquivos – Lista Encadeada – Filas – Árvore B.

Comentários: Vejam que essa é uma questão que trata das possíveis estruturas de dados. Ao analisar cada uma das alternativas podemos, por meio de uma associação direta, entre os termos e as definições marcar nosso gabarito na alternativa E.

Gabarito: E

17. ANO: 2012 BANCA: FUNRIO ÓRGÃO: MPOG PROVA: ANALISTA - TECNOLOGIA DA INFORMAÇÃO

Sobre a organização de arquivos de registros desordenados (Heap Files), usada frequentemente em sistemas de gerenciamento de banco de dados, é correto afirmar que

A é possível usar pesquisa binária sobre campo chave.

B seus registros são armazenados em árvores-B.

C permite o uso de índices primários.

D requer uma função de hashing externo.

E incluir um novo registro é muito eficiente.

Comentários: Quando tratamos da organização de registro em arquivos tempos basicamente 4 tipos segundo o silberchatz:

Organização de arquivo heap: Nessa organização, qualquer registro pode ser colocado em qualquer lugar no arquivo onde haja mais espaço para registro. Não há uma ordem de registro. Normalmente, há um único arquivo para cada relação.

Organização de arquivo sequencial: Nessa organização, os registros são armazenados em ordem sequencial, baseada no valor da chave primária de cada registro.

Organização de arquivo hashing: Nessa organização, uma função hash é calculada sobre algum atributo de cada registro. O resultado da função hash especifica em qual bloco do arquivo o registro deve ser colocado.

Organização de arquivo clustering: Nessa organização, os registros de diferentes relações podem ser armazenados no mesmo arquivo. Registros são relacionados de diferentes relações são armazenados no mesmo bloco, de forma que uma operação de I/O busque os registros relacionados de todas a relações.

Vejam que da lista de opções acima a organização de arquivo heap garante uma inclusão de registro mais eficiente.

Gabarito: E

18. ANO: 2012 BANCA: ESAF ÓRGÃO: MI PROVA: ANALISTA DE SISTEMAS - INFORMÁTICA E REDES

Em relação a sistemas de gerenciamento de arquivos, é correto afirmar que:

A a organização de arquivos consiste em como seus dados estão quantitativamente dispostos.

B cada arquivo possui informações de acesso denominadas flowindexes.

C a forma como o sistema organiza logicamente os diversos arquivos contidos em um disco é a estrutura em diretórios.

D um mapa de bytes (middle byte) permite implementar uma estrutura de espaços de controle.

E a alocação contígua consiste em armazenar um arquivo em blocos aleatoriamente dispostos no disco.

Comentários: Ao analisar cada uma das alternativas acima, podemos verificar que a única que faz sentido do ponto de vista teórico é a alternativa C. Sua descrição trata da definição de uma estrutura de diretórios.

Gabarito: C

19. ANO: 2013 BANCA: FGV ÓRGÃO: AL-MT PROVA: ANALISTA DE SISTEMAS - PROGRAMADOR

As alternativas a seguir apresentam características dos arquivos Heap, à exceção de uma. Assinale-a.

A Inclusão de novos registros é muito eficiente.

B Organização de arquivos é a mais simples e básica

C Pesquisa de registros envolve busca sequencial bloco a bloco.

D Para um arquivo com b blocos, uma busca envolve em média b/2 blocos.

E Os registros são posicionados de modo ordenado através da chave de classificação.

Comentários: Vejam que a que pede a alternativa que não representa uma característica dos arquivos de Heap.

Os dados nas páginas de um arquivo heap não estão ordenados de nenhuma maneira, e a única garantia é que se podem recuperar todos os registros no arquivo por meio de repetidas solicitações do próximo registro. Cada registro no arquivo possui um id único, e todas as páginas de um arquivo são do mesmo tamanho. Também são conhecidos como **Arquivos de Registros Desordenados**.

É considerado o método mais simples, onde os registros são armazenados nos arquivos na ordem em que são inseridos (isto é, na última posição). A inserção

de um novo registro é muito eficiente neste tipo de estrutura. O último bloco do banco de dados no disco é copiado para um buffer, o novo registro é acrescentado e depois ele é regravado de volta no disco.

No entanto, a busca e exclusão neste tipo de arquivo é mais trabalhoso, pois demanda uma pesquisa linear. Para excluir um registro, o programa deve primeiro encontrar o seu bloco, copiá-lo para o buffer, excluir o registro no buffer, reorganizar o bloco e regravar o bloco no disco. Isto pode deixar espaços livres nos blocos, que após um grande número de exclusões pode resultar em muito espaço desperdiçado.

Após toda essa explicação podemos marcar a resposta na alternativa E que apresenta uma característica que **não** pertence aos arquivos Heap.

Gabarito: E

20. ANO: 2015 BANCA: IDECAN ÓRGÃO: INMETRO PROVA: ANALISTA EXECUTIVO EM METROLOGIA E QUALIDADE - INFRAESTRUTURA

"Nos bancos de dados relacionais, os dados são armazenados em tabelas. Um banco de dados Oracle tem uma estrutura física e lógica. Uma vez que tais estruturas são separadas no servidor, o armazenamento físico dos dados pode ser gerenciado, de modo a não afetar o acesso às estruturas lógicas de armazenamento. A estrutura lógica do Oracle é determinada por um ou mais tablespaces – que são espaços lógicos do armazenamento – e pelos objetos de esquema do banco de dados. Para agilizar o acesso às linhas de uma tabela, o Oracle usa o conceito de índices, semelhante aos encontrados em livros, que é uma estrutura opcional associada com tabelas e clusters que permite a execução mais rápida de comandos SQL. O Oracle possui diversos tipos de índices que oferecem vantagens para determinados tipos de aplicações. Um desses índices é compacto e trabalha melhor com colunas com pouca variação de conteúdo." Trata-se do

A B-tree indexes.

B Bitmap indexes.

C Reverse Key indexes.

D B-tree cluster indexes.

E Global and local indexes.

Comentários: Aproveitaremos essa questão para mostrar como um SGBD, no caso o Oracle, implementa os índices.

Um índice é um objeto de schema que contém uma entrada para cada valor que aparece numa coluna indexada de uma tabela ou cluster e oferece acesso rápido e direto às linhas correspondentes. O banco de dados Oracle suporta uma variedade de tipos de índices:

Normal indexes: por padrão (default) o Oracle cria índices do tipo B-tree;

Bitmap indexes: armazena os rowids associados a um valor de chave em forma de bitmap;

Partitioned indexes: consiste em um particionamento contendo uma entrada para cada valor que aparece na coluna indexada da tabela;

Function-based indexes: representam expressões. Possibilitam a construção de consultas que avaliam o valor retornado por uma expressão, que por sua vez podem incluir buit-in functions ou user-defined functions;

Domain indexes: são instâncias de uma aplicação específica de um tipo de index.

Sobre os Bitmap Indexes precisamos detalha-los um pouco mais. Bitmap indexes são amplamente utilizados em ambientes de data warehouse. Este tipo de índice oferece:

Tempo de resposta reduzido para grandes volumes de dados; Espaço de armazenamento reduzido comparado com outras técnicas de indexação; Ganhos dramáticos de performance e relativamente pequeno consumo de CPU; Manutenção eficiente durante operações de DML paralelas ou cargas de dados.

Bitmap indexes são muito efetivos para consultas que contêm muitas condições em suas respectivas cláusulas WHERE. Linhas que satisfaçam algumas, porém não todas as condições, são filtradas antes da tabela ser acessada, reduzindo muito o tempo de acesso das consultas. São geralmente usados para colunas com pouca variação de conteúdo, por exemplo, sexo (masculino, feminino), formação (superior completo, superior incompleto, mestrado, doutorado, PhD).

Gabarito: B

21. ANO: 2015 BANCA: FCC ÓRGÃO: CNMP PROVA: ANALISTA DO CNMP - SUPORTE E INFRAESTRUTURA

Em um sistema de banco de dados relacional, os índices representam um tipo de estrutura de grande importância. Considerando os tipos de índices existentes, é correto afirmar que

A um índice ordenado do tipo esparso não contém registros para todos os valores da chave de busca.

B um índice ordenado do tipo esparso ou denso não pode utilizar a chave primária da tabela como chave de busca.

C uma vez montado um índice, esparso ou denso, ele não precisa mais ser atualizado no decorrer do uso do banco de dados.

D quando um registro é inserido em um índice, ele deve permanecer até que a tabela que originou o índice seja excluída.

E um banco de dados com poucas tabelas não comporta os índices chamados multinível.

Comentários: Índices são estruturas de dados que recebem como entrada uma propriedade de registro (um valor de um ou mais campos) e encontra os registros com essa propriedade rapidamente. Portanto, os índices são estruturas auxiliares cujo único propósito é tornar mais rápido o acesso a registros baseados em certos campos, chamados campos de indexação.

Existem diversos tipos de índices conforme vimos na nossa aula:

Primário: baseado na chave de ordenação;

Agrupamento/Clustering: baseado no campo de ordenação não-chave de um arquivo;

Secundário: baseado em qualquer campo não ordenado de um arquivo;

Índices multiníveis;

Árvores B e B+;

Tabelas Hash.

Índices sobre arquivos sequenciais geralmente são classificados como **densos** ou **esparsos**:

Índices **Densos**: uma entrada no arquivo de índice para cada entrada no arquivo de dados. Desta forma, os índices densos apresentam uma sequência de blocos contendo apenas as chaves dos registros e os ponteiros para os próprios registros.

Índices **Esparsos**: apenas alguns registros de dados são representados no arquivo de índices. Utilizam menos espaço de armazenamento do que o índice denso ao custo de um menor tempo de acesso.

Em outras palavras, um índice ordenado (sequencial) do tipo esparso, ao contrário do denso, não contém registros para todos os valores da chave de busca. Cada entrada de um índice esparso aponta para um bloco de registros. Cada entrada de um índice denso aponta exatamente para o registro em si.

Desta forma, a resposta correta para esta questão é a letra A.

Gabarito: A

22. ANO: 2014 BANCA: VUNESP ÓRGÃO: TCE-SP PROVA: AGENTE DA FISCALIZAÇÃO FINANCEIRA - INFRAESTRUTURA DE TI E SEGURANÇA DA INFORMAÇÃO

Em sistemas gerenciadores de bancos de dados relacionais, uma das técnicas largamente utilizadas para a otimização de desempenho consiste no (na)

A limitação no número de campos de cada tabela do banco de dados.

B proibição da inserção de valores nulos nos registros das tabelas.

C uso extensivo de triggers ou gatilhos para a realização de procedimentos.

D criação de grupos de usuários, com características semelhantes, do banco de dados.

E criação de índices sobre alguns campos das tabelas do banco de dados.

Comentários: Já falamos que os SGBDs utilizam ferramentas próprias para analisar a base de dados e sugerir a criação de índices. Essa criação é geralmente baseada em critérios operacionais que vão melhorar o desempenho de determinadas consultas sobre o banco de dados.

Gabarito: E

23. ANO: 2010 BANCA: CESPE ÓRGÃO: TRT - 21ª REGIÃO (RN) PROVA: ANALISTA JUDICIÁRIO - TECNOLOGIA DA INFORMAÇÃO

Acerca de desempenho e otimização de consultas SQL no SQL Server 2008, julgue os itens de 63 a 69.

[1] Os índices do tipo clustered determinam a ordem física dos dados em uma tabela e mostram-se particularmente eficientes em colunas pesquisadas frequentemente por uma faixa de valores ou quando o valor do registro é único na tabela.

Comentários: Dentre as estratégias de otimização para bancos de dados os índices são os mais usados e atuam diretamente no sequenciamento dos dados mapeados, criando links para as linhas das tabelas e aumentando a performance das consultas.

Um índice clusterizado determina a ordem em que as linhas de uma tabela são armazenadas no disco. Se uma tabela tem um índice clusterizado, no momento de um INSERT as linhas dessa tabela serão armazenadas em disco na ordem exata do mesmo índice. Por exemplo, suponha que temos uma tabela chamada "Livro" que tem uma coluna de chave primária "livroID" e que criamos um índice clusterizado para essa mesma coluna. Ao fazer isso, todas as linhas dentro da tabela Livro serão fisicamente ordenadas (no disco atual em que estão inseridas) através dos valores que estão na coluna livroID.

Isso implicará em um ganho enorme na performance das pesquisas, pois as colunas da tabela estarão ordenadas na mesma ordem dos índices clusterizados por intermédio do modelo de armazenamento usado por esse tipo de índice.

Baseado no que acabamos de explicar podemos marcar a alternativa como correta.

Gabarito: C

24. ANO: 2014 BANCA: FGV ÓRGÃO: CM-RECIFE PROVA: ASSISTENTE LEGISLATIVO - PROGRAMADOR

Árvores B são largamente utilizadas na construção de índices em implementações de bancos de dados. Considere as seguintes afirmativas sobre esse tipo de organização:

I. Há apenas um nó raiz.

II. O algoritmo de remoção de uma chave não preserva o balanceamento da árvore, o que é feito periodicamente nos bancos de dados por meio de um processo de limpeza dos índices.

III. O algoritmo de inserção preserva o balanceamento da árvore, criando novos nós e alterando a estrutura da árvore quando necessário.

IV. Numa tabela de banco de dados onde a chave de indexação é composta por mais de uma coluna, a ordem dessas colunas no comando de criação do índice é irrelevante.

Assinale se:

A todas as afirmativas estão corretas;

B somente as afirmativas I, II e IV estão corretas;

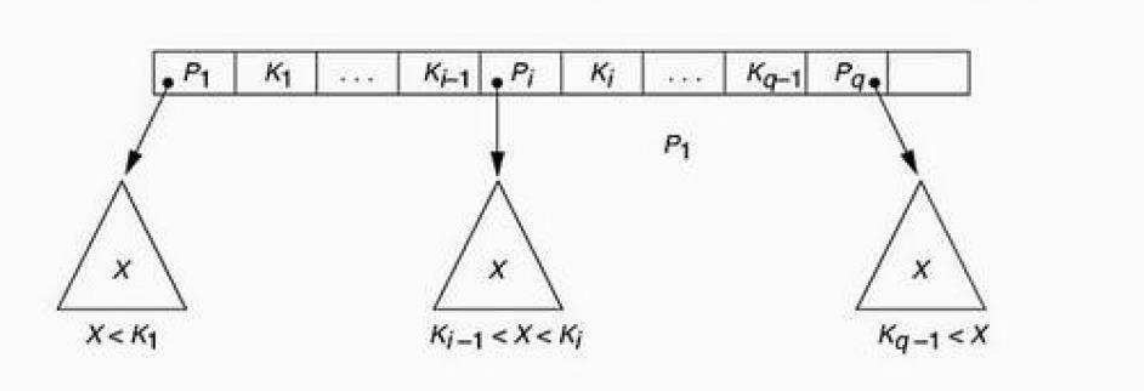
C somente as afirmativas II e III estão corretas;

D somente as afirmativas I e III estão corretas;

E nenhuma das alternativas está correta.

Comentários: Árvore B ou B-Tree são estruturas de dados muito utilizadas em banco de dados e sistema de arquivos.

As árvores B são um tipo de árvores de busca que foram projetadas para minimizar o número de acessos à memória secundária. Como o número de acessos à memória secundária depende diretamente da altura da árvore, as árvores B foram projetadas para ter uma altura inferior às árvores rubro-negras ou AVL, para um dado número de chaves. Para manter o número de registros armazenados e, ao mesmo tempo, diminuir a altura, uma solução é aumentar o grau de ramificação da árvore (o número máximo de filhos que um nó pode ter). Assim árvores B são árvores de busca que possuem um grau de ramificação geralmente muito maior que 2. Além disso, a cada nó de uma árvore B são associados mais de um registro de dados: se o grau de ramificação de um nó for g, este pode armazenar até g-1 registros. A figura abaixo mostra um exemplo de árvore B.



Na árvore B temos apenas um Nó Raiz, e as operações de inserção e exclusão preservam o balanceamento da árvore.

Para inserir um novo elemento em uma árvore B, basta localizar o nó folha X onde o novo elemento deva ser inserido. Se o nó X estiver cheio, será necessário realizar uma subdivisão de nós que consiste em passar o elemento mediano de X para seu pai e subdividir X em dois novos nós com t - 1 elementos e depois inserir a nova chave.

Até aqui podemos observar que as alternativas I e III estão corretas e alternativa II está errada.

A última alternativa trata da ordem dos atributos na criação dos índices, isso certamente fara diferença na hora de se construir ou criar o índice, pois pode mudar a formas como o SGBD estrutura os índices fisicamente.

Gabarito: D

25. ANO: 2015 BANCA: IESES ÓRGÃO: IFC-SC PROVA: INFORMÁTICA - BANCO DE DADOS

Qual das afirmações a seguir é uma característica positiva atribuída ao uso de índices em banco de dados?

A Menor consumo de espaço em disco para armazenamento dos dados.

B Garantia de integridade dos dados.

C Atribuição de um identificador único para cada registro gravado.

D Melhor desempenho nas consultas às tabelas do banco de dados.

Comentários: Algumas características dos índices que nós conhecemos, primeiramente eles melhoram o desempenho das consultas que tem como referencias o (s) campo (s) utilizado (s) na construção do índice. Outro ponto é que o arquivo de índice precisa ser armazenado, esses vai ocupar um espaço extra diferente do arquivo de dados.

Os índices não têm preocupação direta com a integridade dos dados, esses são controlados pelo SGBD por meio da garantia das restrições de integridade e do controle de transações. A identificação única só tem guarida quando temos um índice denso, o que não é verdadeiro para 100% dos casos.

Pelo exposto, temos nossa resposta na alternativa D.

Gabarito: D

26. ANO: 2015 BANCA: CESPE ÓRGÃO: TCU PROVA: AUDITOR FEDERAL DE CONTROLE EXTERNO - TECNOLOGIA DA INFORMAÇÃO

Considere as seguintes configurações de servidores de banco de dados (SBD):

I O controle de concorrência do servidor X foi configurado para o tipo bloqueio de modo múltiplo.

II Especificamente na tabela T do servidor X, foram criados, em dois campos distintos, dois índices (IdxA e IdxB) contendo apenas um campo para cada um: o primeiro, IdxA, do tipo primário, e o segundo, IdxB, do tipo secundário, em um campo não chave.

III Dois servidores foram configurados para trabalhar de forma distribuída do tipo SBDF (sistemas de banco de dados federado), com intuito primordial de garantir mais disponibilidade, no caso de falha de um dos servidores.

IV O servidor Z foi configurado com um sistema do tipo orientado a objeto.

[1] Na configuração II, o índice IdxA é não denso e possui ancoragem de bloco no arquivo de dados. O índice IdxB pode ser denso ou não, mas não possui ancoragem de bloco no arquivo de dados.

Comentários: Esta questão está de acordo com Navathe. Quando temos um índice primário, ele necessariamente será esparso e deve utilizar âncora de bloco no arquivo de dados. Quando o índice é secundário, ou seja, é baseado em um campo não chave, ele pode ser denso ou esparso, sem âncora de bloco no arquivo de dados.

Gabarito: C

27. ANO: 2015 BANCA: CESPE ÓRGÃO: STJ PROVA: ANALISTA JUDICIÁRIO - INFRAESTRUTURA

[1] No que se refere a fundamentos de banco de dados, julgue os itens a seguir.

O uso de índices pode aumentar o desempenho na operação de bancos de dados. No entanto, caso um arquivo com índice sequencial seja utilizado, o desempenho do banco de dados é reduzido à medida que o tamanho desse arquivo aumenta.

Comentários: O índice sequencial trata basicamente de indexar um arquivo sequencial. Parece óbvio que mesmo com a melhora introduzida pela indexação o crescimento do arquivo reduz o desempenho do banco.

Gabarito: C

28. ANO: 2015 BANCA: CESPE ÓRGÃO: STJ PROVA: ANALISTA JUDICIÁRIO - INFRAESTRUTURA

Com referência a tunning de banco de dados, julgue os itens subsecutivos.

[1] O ajuste de índices é um procedimento utilizado para aumentar o desempenho de um banco de dados. Esse ajuste é realizado removendo-se ou inserindo-se índices apropriados para as relações, visto que o gargalo de desempenho pode ser causado pelo excesso de atualizações ou de consultas no banco de dados.

Comentários: A questão está correta. Um dos utilitários apresentado pelos SGBDs permite que o DBA, baseado nas estatísticas presentes no dicionário de dados sobre a tabela e em informações sobre o uso do sistema de banco de dados, defina índices que otimizem as consultas e demais operações sobre o banco.

Gabarito: C

29. ANO: 2015 BANCA: IESES ÓRGÃO: TRE-MA PROVA: ANALISTA JUDICIÁRIO - ANÁLISE DE SISTEMAS

Os índices são elementos fundamentais em um projeto de Data Warehouse. Os tipos de índices usados em sistemas de Data Warehouse são:

A Índice invertido.

B Índice bitmap e índices join.

C Round índice.

D Índices analíticos semânticos.

Comentários: Já comentamos na aula que os índices de bitmap são importantes para projetos de DW. Os índices de join são criados a partir dos relacionamentos entre duas tabelas para agilizar as consultas. Vejam a figura abaixo que mostra um exemplo doe join index (JI) retirado do paper que define o termo, disponível aqui.

CUSTOMER

csur	cname	city	age	job
1	Smith	Boston	21	clerk
2	Collins	Austin	26	secretary
3	Ross	Austin	36	manager
4	Jones	Paris	29	engineer

JI

csur	cpsur
1	2
1	3
3	1

CP

cpsur	cname	pname	qty	date
2	Smith	jeans	2	052585
3	Smith	shirt	4	052585
1	Ross	jacket	3	072386

Fig. 1. Join index for relations CUSTOMER and CP on attribute cname.

Gabarito: B

Considerações finais

Chegamos, pois, ao final da nossa aula sobre organização de arquivos, métodos de acesso, indexação e *hashing*! Este é o primeiro assunto que consideramos quando pensamos em um curso avançado de banco de dados. Não é à toa que ele está presente na primeira aula do curso de banco de dados avançados.

Espero que você esteja curtindo o assunto de banco de dados. Cada passo dentro do assunto deve ser dado com segurança, portanto, não deixe de mandar suas dúvidas por um dos canais de atendimento do Estratégia.

Peço que depois de lerem esse material, tentem esgotar todas as questões sobre o assunto de algum site de questões. Qualquer dúvida pode postar no fórum aqui do Estratégia.

Thiago Cavalcanti

Referências

Fiz uma lista com alguns links de referências caso você queria se aprofundar um pouco.

i. Sistemas de banco de dados – Navathe, Elmari – 6ª edição.
ii. Introdução a sistemas de banco de dados – C.J. Date – 8 ª edição.
iii. Sistemas de banco de dados – Silberchatz, Korth Sudarshan - 5 ª edição.
iv. Texto sobre HD - http://www.infowester.com/hd.php

ESSA LEI TODO MUNDO CONHECE:
PIRATARIA É CRIME.

Mas é sempre bom revisar o porquê e como você pode ser prejudicado com essa prática.

1. 

Professor investe seu tempo para elaborar os cursos e o site os coloca à venda.

2. 

Pirata divulga ilicitamente (grupos de rateio), utilizando-se do anonimato, nomes falsos ou laranjas (geralmente o pirata se anuncia como formador de "grupos solidários" de rateio que não visam lucro).

3. 

Pirata cria alunos fake praticando falsidade ideológica, comprando cursos do site em nome de pessoas aleatórias (usando nome, CPF, endereço e telefone de terceiros sem autorização).

4. 

Pirata compra, muitas vezes, clonando cartões de crédito (por vezes o sistema anti-fraude não consegue identificar o golpe a tempo).

5. 

Pirata fere os Termos de Uso, adultera as aulas e retira a identificação dos arquivos PDF (justamente porque a atividade é ilegal e ele não quer que seus fakes sejam identificados).

6. 

Pirata revende as aulas protegidas por direitos autorais, praticando concorrência desleal e em flagrante desrespeito à Lei de Direitos Autorais (Lei 9.610/98).

7. 

Concurseiro(a) desinformado participa de rateio, achando que nada disso está acontecendo e esperando se tornar servidor público para exigir o cumprimento das leis.

8. 

O professor que elaborou o curso não ganha nada, o site não recebe nada, e a pessoa que praticou todos os ilícitos anteriores (pirata) fica com o lucro.

Deixando de lado esse mar de sujeira, aproveitamos para agradecer a todos que adquirem os cursos honestamente e permitem que o site continue existindo.