# Representação de Elementos de Dados


Valéria Times

## TÓPICOS

- Elementos de Dados e Campos
- Registros
- Dados e Registros de Comprimento Variável
- Endereços de Blocos e Registros
- Modificações de Registros
- Formatos de Blocos



## Representação de Elementos de Dados

- Introdução:
  - Atributos são representados por seqüências de bytes de comprimento fixo ou variável (campos).
  - Campos são reunidos em coleções de comprimento fixo ou variável (registros).
  - Registros são armazenados em blocos físicos.
  - Uma coleção de registros é armazenada como uma coleção de blocos (arquivo).
    - Para dar suporte eficiente à realização de consultas e de operações de modificação, uma estrutura de índice é inserida no arquivo.



## Representação de Elementos de Dados

- Elementos de Dados e Campos:
  - Campos são os elementos de dados mais primitivos.
  - Muitos deles simplesmente recebem um número apropriado de bytes no armazenamento secundário.
    - Inteiros
    - *Strings* de Comprimento Fixo



## Representação de Elementos de Dados

- Elementos de Dados e Campos: (Cont.)
  - *Strings* de Comprimento Fixo
    - Campo para um atributo definido como sendo do tipo de SQL CHAR(n) corresponde a um *array* de n bytes.
    - Caso o valor deste atributo tenha um comprimento < n, o *array* será completado com um caracter especial de preenchimento.
    - Exemplo: considere um atributo A declarado como do tipo CHAR (5).
      - Se A = 'cat', então seu valor = c a t ⊥ ⊥



## Representação de Elementos de Dados

- *Strings* de Comprimento Variável
  - Implementação de atributos declarados como do tipo VARCHAR(n) usa (n + 1) bytes para manter o valor do *string*, independentemente do seu comprimento.
  - Tipo VARCHAR(n) de SQL representa na realidade, campos de comprimento fixo, embora seu valor possa ter um comprimento variável.
  - Há duas representações comuns para *strings* VARCHAR:
    - Comprimento mais Conteúdo
    - *String* terminado em Nulo

## Representação de Elementos de Dados

- *Strings* de Comprimento Variável (Cont.)
  - Comprimento mais Conteúdo
    - Aloca-se um *array* de (n + 1) bytes.
    - *String* não pode exceder n caracteres.
    - Primeiro byte contém o número de bytes no *string* (como um inteiro de 8 bits).
    - Segundo byte em diante contém os caracteres do *string*.
    - Quaisquer bytes do *array* que não sejam usados, são ignorados.
    - Exemplo: Seja A do tipo VARCHAR(10).

      Se A = 'cat', então seu valor = 3 c a t



## Representação de Elementos de Dados

- *Strings* de Comprimento Variável (Cont.)
  - *String* terminado em Nulo
    - Aloca-se um *array* de (n + 1) bytes.
    - *String* não pode exceder n caracteres.
    - *Array* é preenchido com os caracteres do *string*, seguidos por um caracter nulo.
    - Terminador nulo indica o momento de encerrar a busca.
    - Exemplo: Seja A do tipo VARCHAR(10).

      Se A = 'cat', então seu valor = c a t ⊥



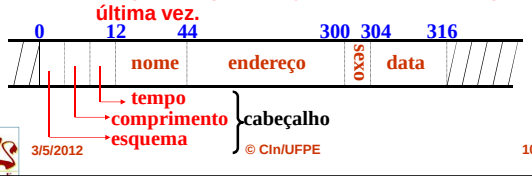
## Representação de Elementos de Dados

- Registros:
  - Cada tipo de registro mantido por um SGBD possui um esquema.
  - Cada esquema de registro possui as seguintes informações sobre os campos:
    - nomes e tipos de dados
    - número total de tipos de dados
    - ordem em que eles aparecem na tupla
    - restrições sobre os atributos e a própria relação
    - deslocamento dentro do registro
      - número de bytes desde o início do registro em que o próprio campo começa.



## Representação de Elementos de Dados

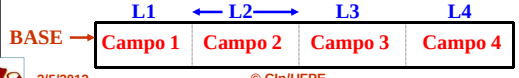
- Registros: (Cont.)
  - São compostos de vários campos, além de um cabeçalho.
  - Informações do cabeçalho de registro:
    - ponteiro para o esquema do registro
    - comprimento do registro
    - tempo em que o registro foi acessado pela última vez.






## Representação de Elementos de Dados

- Registros de Comprimento Fixo
  - Cada campo tem um tamanho fixo.
  - Número de campos é também fixo.
  - Campos podem ser armazenados consecutivamente.
  - Dado o endereço de um registro (base), o endereço de um campo específico pode ser obtido a partir:
    - deslocamento do campo dentro do registro
    - tamanho dos campos precedentes






## Representação de Elementos de Dados

- Registros de Comprimento Fixo em Blocos


  - Cabeçalho de bloco guarda informações como:
    - Ponteiros para um ou mais blocos diferentes que fazem parte de uma rede blocos
    - Função desempenhada pelo bloco na rede
    - Nome da relação das tuplas do bloco
    - Deslocamento de cada registro no bloco
    - ID do bloco
    - Tempo do último acesso ao bloco.

## Representação de Elementos de Dados

- Dados e Registros de Comprimento Variável:
  - Formatos de registros de tamanho variável podem também ser usados para armazenar registros de tamanho fixo:
    - custo adicional é justificado pela flexibilidade:
      - suportar valores nulos
      - adicionar novos campos
  - Pode-se desejar representar:
    - Itens de dados cujo tamanho varia
    - Campos multivalorados
    - Registros de formato variável
    - Registros que não se encaixam em um bloco



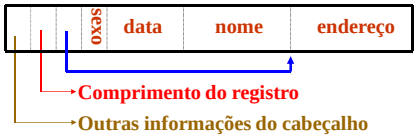
## Representação de Elementos de Dados

- Itens de dados cujo tamanho varia:
  - Se um ou mais campos de um registro têm comprimento variável, então o registro deve conter informações suficientes para localizar qualquer campo.
  - Abordagem simples, mas efetiva:
    - Colocar todos os campos de comprimento fixo antes dos campos de comprimento variável.
    - Inserir no cabeçalho do registro:
      - comprimento do registro
      - ponteiros para o início de todos os campos de comprimento variável.



## Representação de Elementos de Dados

- Itens de dados cujo tamanho varia (Cont.):
  - Se a ordem dos campos de comprimento variável é sempre a mesma, então o primeiro deles não precisa de nenhum ponteiro.
  - Exemplo: Considere um tipo de registro:
    - sexo e data de nascimento: fixo
    - nome e endereço: variável




Comprimento do registro

Outras informações do cabeçalho



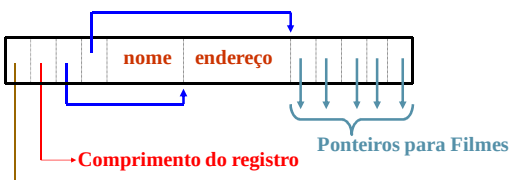
## Representação de Elementos de Dados

- Campos multivalorados:
  - Situação onde um campo C de um registro possui um número variável de ocorrências.
  - Abordagem:
    - Agrupar todas as ocorrências do campo C.
    - Inserir no cabeçalho do registro, um ponteiro para a primeira ocorrência.
  - Como localizar as ocorrências:
    - Seja L o número de bytes necessários para representar uma instância do campo C.
    - Adiciona-se ao deslocamento de C, todos os múltiplos inteiros de L (0, L, 2L, 3L,...) até alcançar o deslocamento do campo que segue C.



## Representação de Elementos de Dados

- Campos multivalorados (Cont.):
  - Exemplo: Considere um tipo de registro:
    - fimes: multivalorado
    - nome e endereço: variável




Comprimento do registro

Outras informações do cabeçalho (incluindo L)

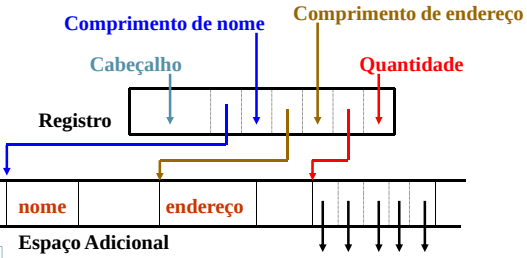


## Representação de Elementos de Dados

- Campos multivalorados (Cont.):
  - Representação Alternativa:
    - Manter o registro de comprimento fixo.
    - Inserir a parte de comprimento variável (campos de comprimento variável ou multivalorados) em um bloco separado.
    - No próprio registro, guardar para cada campo multivalorado:
      - ponteiro para o início do campo.
      - quantidade de valores mantidos ou ponteiro para o final do campo.

## Representação de Elementos de Dados

- Exemplo: Considere um tipo de registro:
  - fimes: multivalorado
  - nome e endereço: variável






## Representação de Elementos de Dados

- Representação Alternativa: (Cont.)
  - Consiste no uso de Indireção para os campos de comprimento variável de um registro.
  - Vantagem:
    - manter o próprio registro de comprimento fixo permite:
      - registros sejam lidos com maior eficiência.
      - minimizar o *overhead* em cabeçalhos de blocos.
      - registros sejam movidos dentro ou entre blocos com mínimo esforço.



## Representação de Elementos de Dados

- Representação Alternativa: (Cont.)
  - Desvantagem:
    - armazenamento de campos de comprimento variável em outro bloco aumenta o número de operações de E/S de disco necessárias para ler todos os campos de um registro.
  - Solução ?
    - Uso de uma estratégia "meio-termo"



## Representação de Elementos de Dados

- Campos multivalorados (Cont.):
  - Estratégia de Meio-Termo:
    - Manter na parte de comprimento fixo do registro, espaço suficiente para:
      - Algum número razoável de ocorrências dos campos multivalorados.
      - Um ponteiro para onde seriam encontradas ocorrências adicionais.
      - Contagem do número de ocorrências adicionais existentes.



## Representação de Elementos de Dados

- Estratégia de Meio-Termo: (Cont.)
  - Se houver menos informação do que é possível inserir no registro de comprimento fixo:
    - uma parte do espaço reservado neste registro não será usada.
    - ponteiro para o espaço adicional será nulo.
  - Senão:
    - pode-se localizar as ocorrências adicionais seguindo-se o ponteiro apropriado (não nulo).



## Representação de Elementos de Dados

- Registros de Formato Variável
  - Situação ainda mais complexa ocorre quando registros não têm um esquema fixo.
  - Exemplo:
    - Aplicações de integração de informações
    - Qualquer outra área de aplicação com necessidade de esquemas flexíveis.
  - Representação mais simples de registros de formato variável consiste em uma seqüência de campos marcados.

## Representação de Elementos de Dados

- Registros de Formato Variável (Cont.)
  - Campos marcados consistem em:
    - Informações sobre a função deste campo:
      - nome do atributo
      - tipo do campo
      - comprimento do campo
    - Valor do campo




## Representação de Elementos de Dados

- Registros que não se encaixam em um bloco
  - Em geral, registros são armazenados dentro de blocos.
    - Um cabeçalho de bloco com informações sobre este bloco consome uma parte do espaço no bloco, com o restante ocupado por um ou mais registros.
  - Porém, se os registros forem maiores que os blocos, ou se desejarmos fazer uso do espaço restante dentro dos blocos, podemos dividir registros em dois ou mais fragmentos, um em cada bloco.



## Representação de Elementos de Dados

- Registros que não se encaixam em um bloco(Cont.)
  - A parte de um registro que aparece em um bloco é chamada fragmento de registro.
  - Um registro com dois ou mais fragmentos é chamado registro com faixas.
  - Registros que não cruzam o limite de um bloco são registros sem faixas.
  - Um cabeçalho de fragmento é então necessário para vincular os fragmentos de um registro.



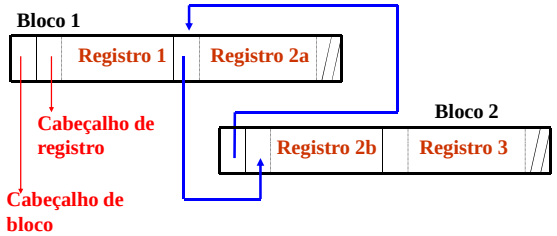
## Representação de Elementos de Dados

- Registros que não se encaixam em um bloco(Cont.)
  - Todo registro e todo fragmento de registro exige algumas informações extras de cabeçalho:
    - Cada cabeçalho de registro ou fragmento deve conter um *bit* informando se ele é ou não um fragmento.
    - Se for um fragmento, ele deve indicar se é o primeiro ou o último fragmento de seu registro.
    - Se existir um fragmento seguinte e/ou anterior para o mesmo registro, então o fragmento deve ter ponteiros para estes outros fragmentos.



## Representação de Elementos de Dados

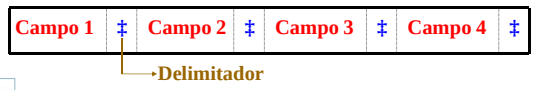
- Formato de um registro com faixas em vários blocos:






## Representação de Elementos de Dados

- Formatos Adicionais para Registros com Campos de Comprimento Variável:
  - Uso de Delimitadores:
    - Armazenamento consecutivo de campos separados por delimitadores
      - delimitador: caracter especial que não aparece no dado propriamente dito
    - Requer uma varredura no registro para localizar o campo desejado.



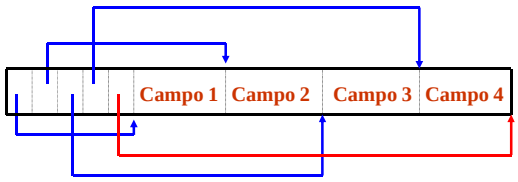
## Representação de Elementos de Dados

- Formatos Adicionais para Registros com Campos de Comprimento Variável (Cont.):
  - Uso de *Array* de Deslocamento de Campos:
    - Reservar um espaço no cabeçalho do registro para manter uma seqüência de ponteiros ou *array* de deslocamento de campos, onde:
      - i-ésimo elemento deste *array* consiste no endereço relativo ao início do registro do i-ésimo campo (deslocamento do i-ésimo campo)
      - deslocamento referente ao final do registro é também mantido para indicar onde o último campo termina.



## Representação de Elementos de Dados

- Formatos Adicionais para Registros com Campos de Comprimento Variável (Cont.):
  - Uso de *Array* de Deslocamento de Campos:




    - Comprimento do registro não é armazenado.
    - Em vez disso, tem-se o deslocamento do final do registro.



## Representação de Elementos de Dados

- Uso de Delimitador x *Array* de Deslocamentos
  - Apesar do custo de armazenamento e de manutenção do *array*, esta abordagem é geralmente superior.
    - Acesso direto a qualquer campo é garantido
    - Flexibilidade para representar valores nulos
      - Valor Nulo: valor especial usado para denotar que a informação é indisponível ou não aplicável.
      - Nesta abordagem, se o valor do campo é nulo, então o ponteiro para o início do campo é igual ao ponteiro para o final dele.
      - Nenhum espaço é desperdiçado.



## Representação de Elementos de Dados

- Representação de valores grandes:
  - Exemplos incluem:
    - imagens em diversos formatos
    - clipes de vídeo
    - sinais de todos os tipos (áudio, radar).
  - São freqüentemente chamados de Objetos Binários Extensos ou BLOBS (Binary Large OBjectS).
  - Devem ser armazenados em muitos blocos.
  - Dependendo dos requisitos de acesso, pode ser aconselhável:
    - manter o BLOB em um único cilindro
    - desmembrá-lo em diversos discos



## Representação de Elementos de Dados

- Campos de tamanho variável podem implicar em processamento adicional, especialmente quando o registro é modificado:
  - Alterações no campo podem fazê-lo crescer, necessitando do deslocamento de todos os campos subsequentes.
  - Um registro alterado pode não mais caber em seu bloco:
    - Deve ser movido para outro bloco
    - Bloco antigo deve ter um endereço de encaminhamento (*forwarding address*) para identificar a nova localização do registro.



## Representação de Elementos de Dados

- Um registro alterado pode não caber em nenhum bloco:
  - Registro deve ser decomposto em partes menores (fragmentos).
  - Fragmentos de registros devem ser conectados via apontadores para permitir a recuperação do registro original.

## Representação de Elementos de Dados

- Endereços de Blocos e Registros:
  - Estes endereços podem ser representados de várias formas
    - Endereços Físicos: Seqüência de bytes que permitem localizar o bloco ou o registro dentro do sistema de armazenamento secundário. Podem incluir:
      - *Host* ao qual o espaço de armazenamento está associado
      - identificador para o disco
      - número do cilindro do disco
      - número da trilha dentro do cilindro



## Representação de Elementos de Dados

- Endereços Físicos:(Cont.)
  - número do bloco dentro da trilha
  - deslocamento do início do registro dentro do bloco (se for o caso).
- Endereços Lógicos: Cada bloco ou registro tem uma seqüência de bytes arbitrária com algum comprimento fixo.
- Tabela de Mapas: É armazenada no disco e relaciona endereços lógicos e físicos.




## Representação de Elementos de Dados

- Tabela de Mapas (Cont.):
  - Nível de indireção envolvido na Tabela de Mapas oferece uma maior flexibilidade.
    - Muitas operações sobre os dados exigem a movimentação de registros
      - dentro de um mesmo bloco
      - de um bloco para outro
    - Todos os ponteiros externos para o registro fazem referência a esta tabela.
    - Tudo que precisa ser feito ao se excluir ou alterar um registro é alterar a entrada para este registro na tabela.



## Representação de Elementos de Dados

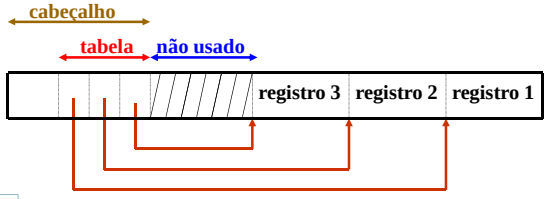
- Endereços de Blocos e Registros (Cont.):
  - São possíveis muitas combinações de endereços lógicos e físicos, formando esquemas de endereços estruturados.
    - Exemplo:

Endereço Estruturado = { endereço físico para o bloco (sem o deslocamento dentro do bloco) + valor da chave para o registro referenciado }



## Representação de Elementos de Dados

- Outro Exemplo: Tabela de Deslocamentos
  - Manter em cada bloco uma tabela de deslocamentos que contém os deslocamentos dos registros dentro do bloco.






## Representação de Elementos de Dados

- Tabela de Deslocamentos: (Cont.)
  - Tabela cresce a partir do início do bloco, enquanto os registros são posicionados a partir do final do bloco.
  - Útil quando os registros não precisam ter o mesmo tamanho.
    - Não se sabe com antecedência quantos registros o bloco conterá
    - Não precisa alocar inicialmente uma parte fixa do cabeçalho de bloco para a tabela.

## Representação de Elementos de Dados

- Tabela de Deslocamentos: (Cont.)

Endereço do Registro (Estruturado) = endereço físico do bloco + deslocamento da entrada na tabela de deslocamentos do bloco para este registro

- Este nível de indireção dentro do bloco oferece muitas das vantagens de endereços lógicos, sem a necessidade de uma tabela de mapas global.



## Representação de Elementos de Dados

- Tabela de Deslocamentos: (Cont.)
  - Vantagens:
    - Pode-se mover o registro dentro do bloco e tudo que precisa ser feito é alterar a sua entrada na tabela de deslocamentos.
      - ponteiros para o registro ainda seriam capazes de localizá-lo.
    - Registro pode ser deslocado para outro bloco, se um endereço de encaminhamento puder ser inserido em sua entrada na tabela de deslocamentos.



## Representação de Elementos de Dados

- Tabela de Deslocamentos: (Cont.)
  - Vantagens:
    - Caso o registro seja excluído, pode-se deixar em sua entrada na tabela de deslocamentos, uma lápide.
      - Lápide - Definição:
      - Valor especial indicando que o registro foi excluído.
      - Adverte o sistema de que o registro não está mais disponível.



## Representação de Elementos de Dados

- Endereços de Blocos e Registros (Cont.):
  - Endereços Estruturados - Definição:
    - Pode-se localizar registros usando parte do endereço físico (localização do bloco em que o registro se encontra), além de informações adicionais, como:
      - chave para o registro
      - posição na tabela de deslocamentos de um bloco que contém o registro.



## Representação de Elementos de Dados

- Modificações de Registros:
  - Muitas vezes, Inserções, Exclusões e Atualizações de registros geram problemas.
- Inserção:
  - Se registros não são mantidos em uma ordem particular:
    - Encontrar um bloco com espaço disponível ou obter um novo bloco.
    - Inserir o registro neste bloco.
  - Porém, registros são geralmente classificados em uma dada ordem (chave primária).



## Representação de Elementos de Dados

- Inserção: (Cont.)
  - Se registros são mantidos classificados:
    - Localizar o bloco apropriado.
    - Verificar se existe espaço no bloco para inserir o novo registro.
    - Analisar a necessidade de deslizar registros pelo bloco para abrir espaço em um local apropriado do bloco.
    - Se for preciso deslizar registros, então o uso de uma tabela de deslocamentos é bastante útil.

## Representação de Elementos de Dados

- Se registros são mantidos classificados:(Cont.)
  - Após o deslizamento de registros (se necessário) e conseqüente ajuste de ponteiros:
    - Novo registro é inserido no bloco.
    - Novo ponteiro para o registro é adicionado à tabela de deslocamentos do bloco.
  - Porém, se não houver espaço no bloco para o novo registro, uma das abordagens é usada (ou ambas):
    - Bloco Vizinho
    - Bloco de Estouro



## Representação de Elementos de Dados

- Encontrar Espaço em um Bloco Vizinho:
  - Examinar o(s) bloco(s) vizinho(s) na ordem classificada de blocos.
  - Se houver espaço, mover registros apropriados de um bloco para outro (de B1 para B2).
  - Se houver ponteiros externos para os registros movidos, deve-se deixar endereços de encaminhamento (em B1) para indicar suas novas localizações (em B2).
  - Em geral, o uso de tais endereços aumenta o espaço necessário para as entradas da tabela de deslocamento.



## Representação de Elementos de Dados

- Criar um Bloco de Estouro:
  - Cada bloco pode ter em seu cabeçalho, um ponteiro para um bloco de estouro ou uma cadeia de blocos, onde podem ser incluídos registros adicionais que pertencem logicamente ao primeiro bloco.
  - Cada bloco de estouro pode apontar para um segundo bloco de estouro e assim por diante (cadeia de blocos).
  - Esquema usado para dar suporte a inserções e a registros crescentes.



## Representação de Elementos de Dados

- Modificações de Registros: (Cont.)
- Exclusão: Nesta operação:
  - Se uma tabela de deslocamentos é usada, então é possível compactar a área de dados do bloco de tal forma que sempre exista uma região não usada no centro.
  - Senão, deve-se manter uma lista de espaços disponíveis no cabeçalho do bloco.
  - Se um registro for excluído de um bloco ou de sua cadeia de estouro, o espaço disponível em todos os blocos deve ser revisto para analisar a possibilidade de remoção de um bloco de estouro.



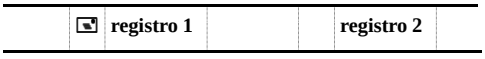
## Representação de Elementos de Dados

- Exclusão: (Cont.)
  - Como pode haver ponteiros para o registro excluído, é necessário a inclusão de uma lápide.
    - Se uma tabela de deslocamentos for usada, então a lápide pode ser um ponteiro nulo nesta tabela, enquanto os ponteiros do registro estariam na realidade, apontando para uma entrada desta tabela.
    - Se uma tabela de mapas for usada, então a lápide pode ser um ponteiro nulo em vez do endereço físico.



## Representação de Elementos de Dados

- Exclusão: (Cont.)
  - Se for preciso substituir registros por lápides:
    - Deve-se ter no início do cabeçalho do registro um bit que sirva como uma lápide.




    - Apenas este bit deve permanecer onde o registro costumava começar, e os bytes subsequentes poderão ser reusados.
    - Se o registro tiver sido excluído, os bytes seguintes não seriam mais examinados.

## Representação de Elementos de Dados

- Modificações de Registros: (Cont.)
- Atualização:
  - Se um registro de comprimento fixo é atualizado:
    - Não há nenhum efeito sobre o sistema de armazenamento.
  - Quando um registro de comprimento variável é atualizado:
    - Tem-se todos os problemas associados com a inclusão/exclusão (exceto pela criação da lápide).



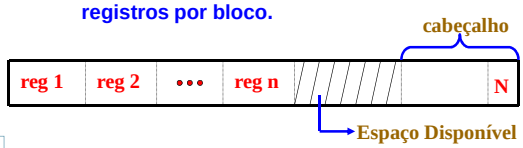
## Representação de Elementos de Dados

- Atualização: (Cont.)
  - Se o registro atualizado for:
    - Maior que a sua versão antiga:
      - Criar mais espaço em seu bloco, movendo registros ou criando novos blocos (blocos de estouro).
    - Menor que a sua versão antiga:
      - Consolidar espaço, possivelmente eliminando blocos de estouro.



## Representação de Elementos de Dados

- Formatos de Blocos para Registros de Comprimento Fixo:
  - Registros são organizados uniformemente e consecutivamente dentro do bloco.
  - Primeira Abordagem:
    - Armazena registros nas primeiras N partições do bloco, onde N é o número de registros por bloco.





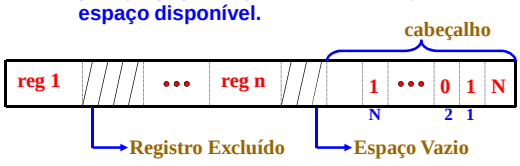

## Representação de Elementos de Dados

- Primeira Abordagem: (Cont.)
  - Sempre que um registro é excluído, o último registro do bloco é movido para a partição vaga.
  - Falha se existem referências externas para o registro movido.
  - Todas as partições vazias estão localizadas juntas no final do bloco.
  - Este esquema permite localizar o i-ésimo registro do bloco através do seu deslocamento.



## Representação de Elementos de Dados

- Formatos de Blocos para Registros de Comprimento Fixo(Cont.):
  - Segunda Abordagem:
    - Exclusões são controladas através do uso de um *array* (um elemento do *array* por partição) para guardar informação sobre espaço disponível.






## Representação de Elementos de Dados

- Segunda Abordagem:
  - Busca de registros requer o acesso ao *array* para identificar partições cujo valor = 1.
  - Quando um registro é removido, sua posição no *array* assume o valor 0.

## Representação de Elementos de Dados

- Formatos de Blocos para Registros de Comprimento Variável:
  - Blocos não podem ser divididos em um número fixo de partições.
  - Manter um diretório de partições para cada bloco, contendo a tupla:

<No.Bloco, No.Partição, Deslocamento, Tamanho>

onde:

Deslocamento:

- Ponteiro para o registro
- Número de bytes desde o início do bloco até o local onde começa o registro.



## Representação de Elementos de Dados

- Formatos de Blocos para Registros de Comprimento Variável:(Cont.)
  - Exclusões são feitas atribuindo o valor -1 ao deslocamento do registro.
  - RID (ID do registro) = No. Bloco + No. Partição
  - Registros podem ser movidos entre as partições do bloco sem afetar referências externas porque apenas seus deslocamentos são alterados.
    - RIDs permanecem inalterados
  - Ponteiro para o início da área de armazenamento disponível é mantido.
  - Entradas do diretório de partições (tuplas) não podem ser removidas.



## Representação de Elementos de Dados

- Formatos de Blocos para Registros de Comprimento Variável:(Cont.)
  - Quando um registro é inserido, uma tupla com deslocamento = -1 pode ser reaproveitada com o novo registro.
  - Esquema é também útil para registros de tamanho fixo, especialmente se são ordenados.
    - Tamanho do registro é mantido em um único local (cabeçalho do bloco ou esquema do BD).
  - Uma variante desta abordagem consiste em manter apenas deslocamentos no diretório.
    - Tamanho é mantido nos primeiros bytes de cada registro