

Sistemas de Gerência de Bancos de Dados

Módulo 1 - Armazenamento

PONTIFÍCIA UNIVERSIDADE CATÓLICA
DO RIO DE JANEIRO



Tópicos

- Introdução
- Organização Física de Discos
- Gerência do Buffer Pool
- Armazenamento em SGBDs Convencionais
- Armazenamento em SGBDs Orientados a Objeto

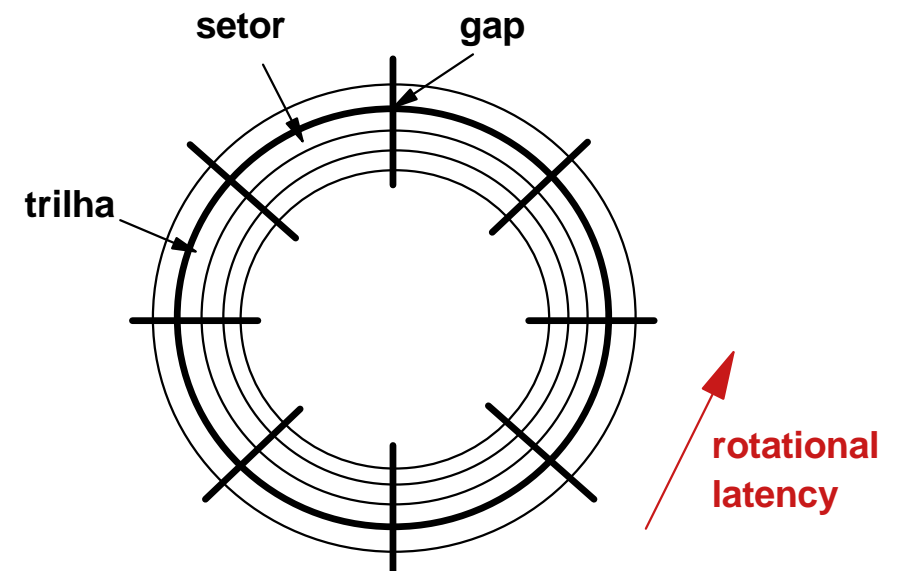
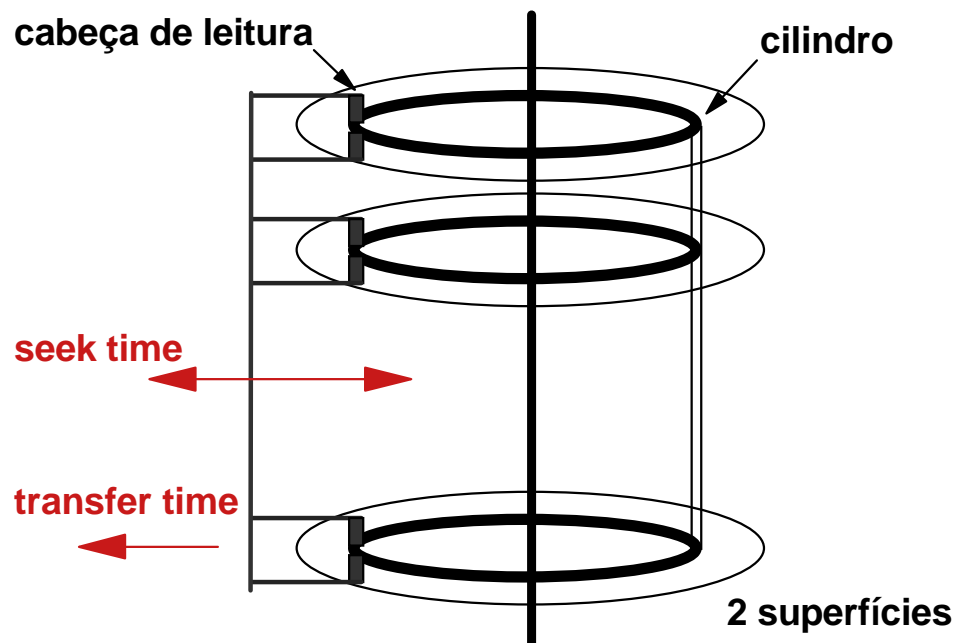
Organização Física de Discos

■ Disco rígido:

► organização:

- dividido em cilindros, trilhas, setores
- trilhas externas podem ter mais setores do que as trilhas internas para manter aproximadamente a mesma densidade de bits

► velocidade de rotação constante



Organização Física de Discos

■ CD-ROM

▶ organização física

- trilha única, em espiral, do centro para a borda
- trilha dividida em setores de 2 Kbytes úteis (+bits de correção de erro)
- total de 650 Mbytes úteis

▶ **velocidade e taxa de leitura constantes**

- velocidade de rotação muda com a posição da cabeça

■ CD-DA (áudio)

▶ organização física:

- semelhante ao do CD-ROM, porém
- setores com menos bits alocados para correção de erro

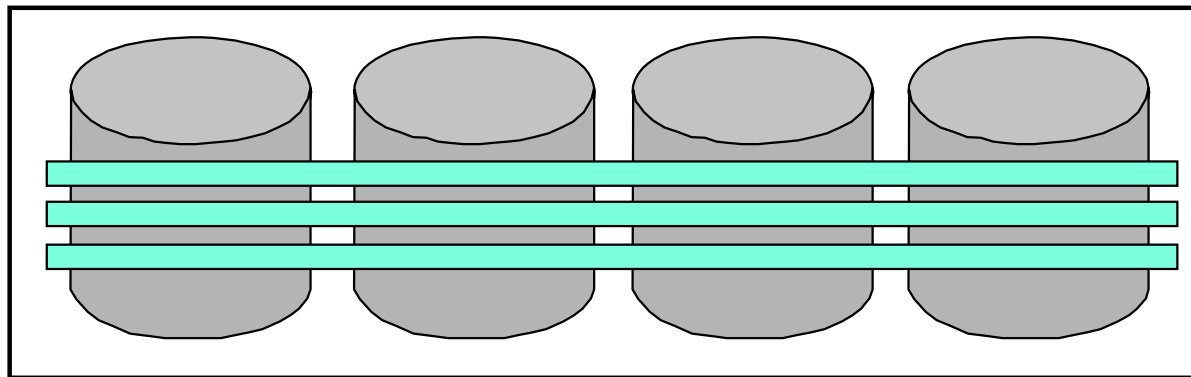
▶ velocidade de leitura padrão de 75 setores / s

Organização Física de Discos

- RAID - Redundant Array of Independent Disks
 - ▶ discos utilizados em conjunto para:
 - aumentar a taxa de transferência de dados (RAID 0)
 - aumentar a confiabilidade através de redundância (RAID 1 a 6)

Organização Física de Discos

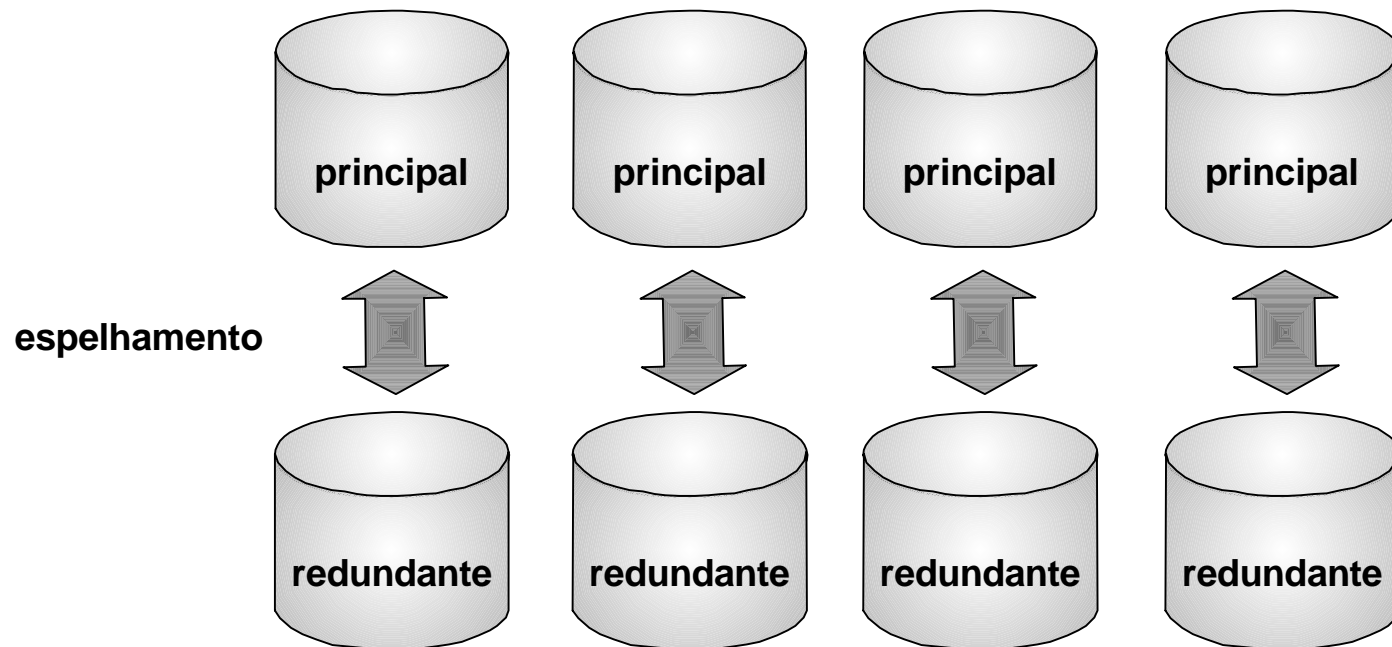
- RAID 0 (non-redundant striping)
 - ▶ sem redundância
 - ▶ acesso em paralelo aos dados (stripped retrieval)
 - supondo que um setor é a unidade de striping, o setor lógico N é quebrado em partes e armazenado nos setores físicos N dos discos
 - stripes completas são lidas em paralelo, multiplicando assim a taxa agregada de transferência



Organização Física de Discos

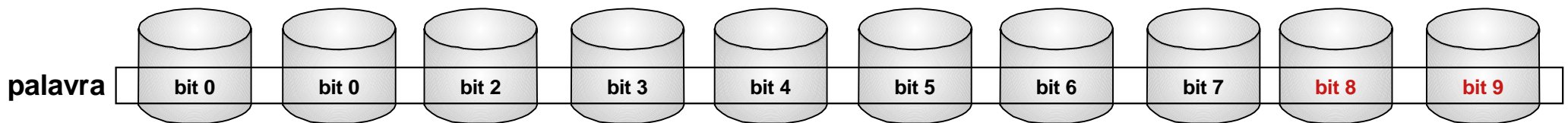
- RAID 1 (mirrored disks)

- ▶ usa espelhamento dos discos
- ▶ permite recuperação de falha em qualquer disco (principal ou redundante, mas não em ambos simultaneamente!)



Organização Física de Discos

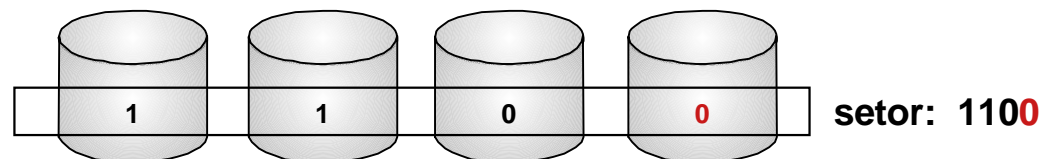
- RAID 2 (memory style error correcting code)
 - ▶ usa código de correção de erro
 - ▶ armazena os bit de cada palavra (do código) em discos separados
 - ▶ permite recuperação de falha em 1 ou mais discos, dependendo do código de correção de erro adotado
 - ▶ exemplo
 - palavra de 10 bits
 - bit 0 a 7 = bits de dados
 - bit 8 e 9 = bits de correção



Organização Física de Discos

■ RAID 3 (bit interleaved parity)

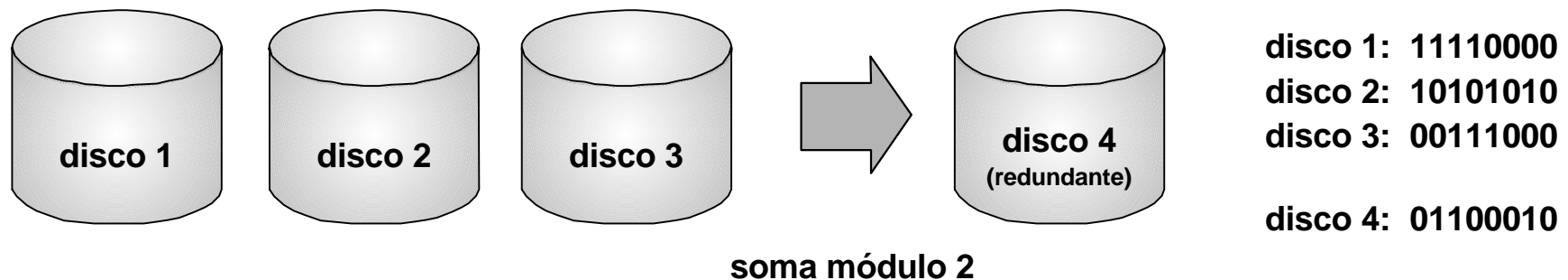
- ▶ usa apenas paridade
- ▶ armazena os bit (de cada setor e de paridade) em discos separados
 - bit de paridade = "soma módulo 2" dos bits dos outros discos
- ▶ permite recuperação de falha em 1 disco, dentre um conjunto de n discos (mas não falha simultânea em 2 discos)
 - controladora detecta qual disco (bit) precisa ser recuperado
 - utiliza paridade para corrigir o disco (bit) em erro
- ▶ exemplo (n=4)
 - bit 0 a 2 = bits de dados
 - bit 4 = bit de paridade



Organização Física de Discos

■ RAID 4 (block interleaved parity)

- ▶ usa apenas paridade
- ▶ armazena o setor com os bits de paridade em um disco separado
 - setor de paridade = "soma módulo 2" dos setores dos outros discos
- ▶ permite recuperação de falha em 1 disco, dentre um conjunto de n discos (mas não falha simultânea em 2 discos)
 - controladora detecta qual disco precisa ser recuperado
 - utiliza paridade para corrigir o disco (setor) em erro
- ▶ exemplo (n=4)



Organização Física de Discos

■ RAID 4 - Exemplo de gravação de bloco

► gravação de novo valor de bloco do disco 2:

- leia o valor antigo do bloco do disco 2
- leia o bloco correspondente do disco 4 (redundante)
- grave o novo valor do bloco do disco 2
- recalcule o novo valor do bloco do disco 4
 - bits modificados no bloco do disco 2

$$10101010 +_2 00101011 = 10000001$$

- novo valor do bloco do disco 4

$$01100010 +_2 10000001 = 11100011$$

- grave o novo valor do bloco do disco 4

antes:

disco 1: 11110000

disco 2: 10101010

disco 3: 00111000

disco 4: 01100010

depois:

disco 1: 11110000

disco 2: 00101011

disco 3: 00111000

disco 4: 11100011

Organização Física de Discos

■ RAID 4

- ▶ recuperação em caso de falha em 1 disco
 - substitua o disco defeituoso por um novo
 - recompute os dados perdidos a partir dos discos restantes:
 - o bit em uma dada posição é a soma módulo 2 dos bits correspondentes dos outros discos
 - exemplo:

disco 1: 11110000	→	disco 1: 11110000
disco 2: ???????		disco 2: 10101010
disco 3: 00111000		disco 3: 00111000
disco 4: 01100010		disco 4: 01100010

■ Problema com RAID 4:

- ▶ o esquema de gravação cria um gargalo no disco redundante

Organização Física de Discos

- RAID 5 (block interleaved distributed parity)
 - ▶ considera cada disco como disco redundante para certas unidades
 - ▶ exemplo:
 - suponha $n+1$ discos numerados de 0 a n
 - o cilindro i do disco j será tratado como redundante se j é o resto da divisão de i por $n+1$
 - esquema para $n=3$:

Disco	Redundante para os cilindros
0	4, 8, 12, ...
1	1, 5, 9, ...
2	2, 6, 10, ...
3	3, 7, 11, ...

Organização Física de Discos

■ RAID 6 (P + Q redundancy)

- ▶ permite recuperação de falhas simultâneas em mais de um disco
- ▶ utiliza vários discos redundantes

▶ exemplo:

- proteção contra 2 falhas simultâneas, utilizando Código de Humming (com distância mínima = 3)
- aplica a mesma estratégia de RAID 4 para 4 discos de dados e 3 discos redundantes, dentro do seguinte esquema:

- bits do disco 5 = soma módulo 2 dos bits correspondentes dos discos 1, 2, 3
- bits do disco 6 = soma módulo 2 dos bits correspondentes dos discos 1, 2, 4
- bits do disco 7 = soma módulo 2 dos bits correspondentes dos discos 1, 3, 4

Disco de Dados				Redundante		
1	2	3	4	5	6	7
1	1	1	0	1	0	0
1	1	0	1	0	1	0
1	0	1	1	0	0	1

Organização Física de Discos

■ RAID 6 - Exemplo de gravação de bloco

► gravação de novo valor de bloco do disco 2:

- discos redundantes afetados = 5 e 6
 - disco 5 = soma módulo 2 dos discos 1, 2, 3
 - disco 6 = soma módulo 2 dos discos 1, 2, 4
- leia o valor antigo do bloco do disco 2
- leia o bloco correspondente do disco 5
- grave o novo valor do bloco do disco 2
- recalcule o novo valor do bloco do disco 5
 - bits modificados no bloco do disco 2
$$10101010 +_2 00001111 = 10100101$$
 - novo valor do bloco do disco 5
$$01100010 +_2 10100101 = 11000111$$
- grave o novo valor do bloco do disco 5
- recalcule o novo valor do bloco do disco 6
 - bits modificados no bloco do disco 2
$$00011011 +_2 10100101 = 10111110$$
- grave o novo valor do bloco do disco 6

Disco de Dados				Redundante		
1	2	3	4	5	6	7
1	1	1	0	1	0	0
1	1	0	1	0	1	0
1	0	1	1	0	0	1

antes:

disco 1: 11110000

disco 2: 10101010

disco 3: 00111000

disco 4: 01000001

disco 5: 01100010

disco 6: 00011011

disco 7: 10001001

depois:

disco 1: 11110000

disco 2: 00001111

disco 3: 00111000

disco 4: 01000001

disco 5: 01100010

disco 6: 10111110

disco 7: 10001001

Organização Física de Discos

■ RAID 6 - Exemplo de recuperação de 2 falhas

► suponha falhas simultâneas nos discos 2 e 5:

- linha 2 difere nos valores das colunas 2 e 5:
 - 1 na coluna 2 e 0 na coluna 5
 - (para quaisquer 2 discos, há sempre uma linha que difere nas colunas correspondentes aos discos)
- linha 2 possui valor 1 na coluna 2 e 1, 4, 6:
 - nenhum destes discos falhou por suposição
 - reconstrua o disco 2 a partir dos discos 1, 4, 6:
 - ♦ disco 2 = soma módulo 2 dos discos 1, 4, 6
- linha 1 possui valor 1 na coluna 5 e 1, 2, 3:
 - reconstrua o disco 5 a partir dos discos 1, 2, 3:
 - ♦ disco 5 = soma módulo 2 dos discos 1, 2, 3

Disco de Dados				Redundante		
1	2	3	4	5	6	7
1	1	1	0	1	0	0
1	1	0	1	0	1	0
1	0	1	1	0	0	1

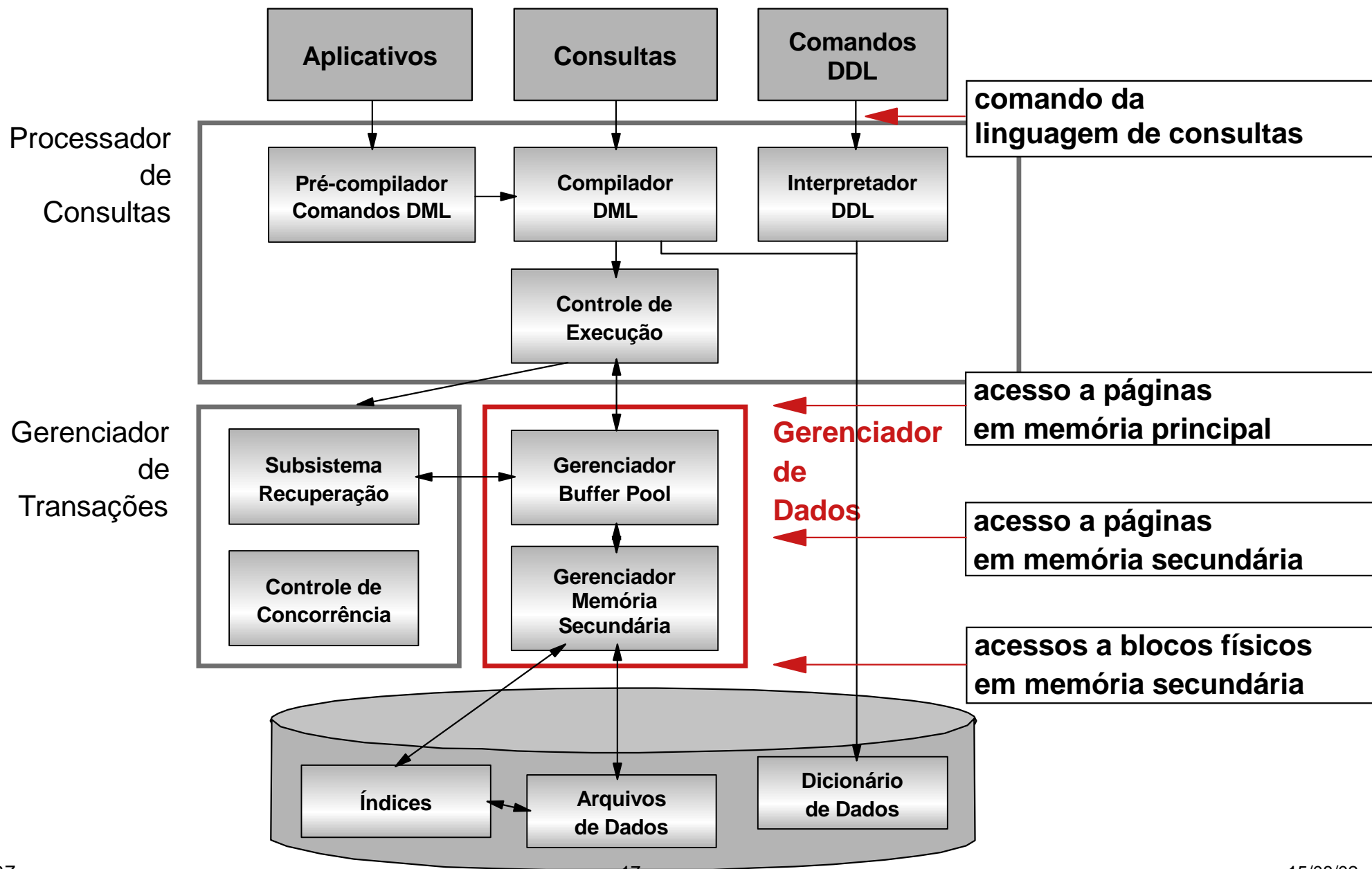
antes:

disco 1: 11110000
disco 2: ????????
disco 3: 00111000
disco 4: 01000001
disco 5: ????????
disco 6: 00011011
disco 7: 10001001

depois:

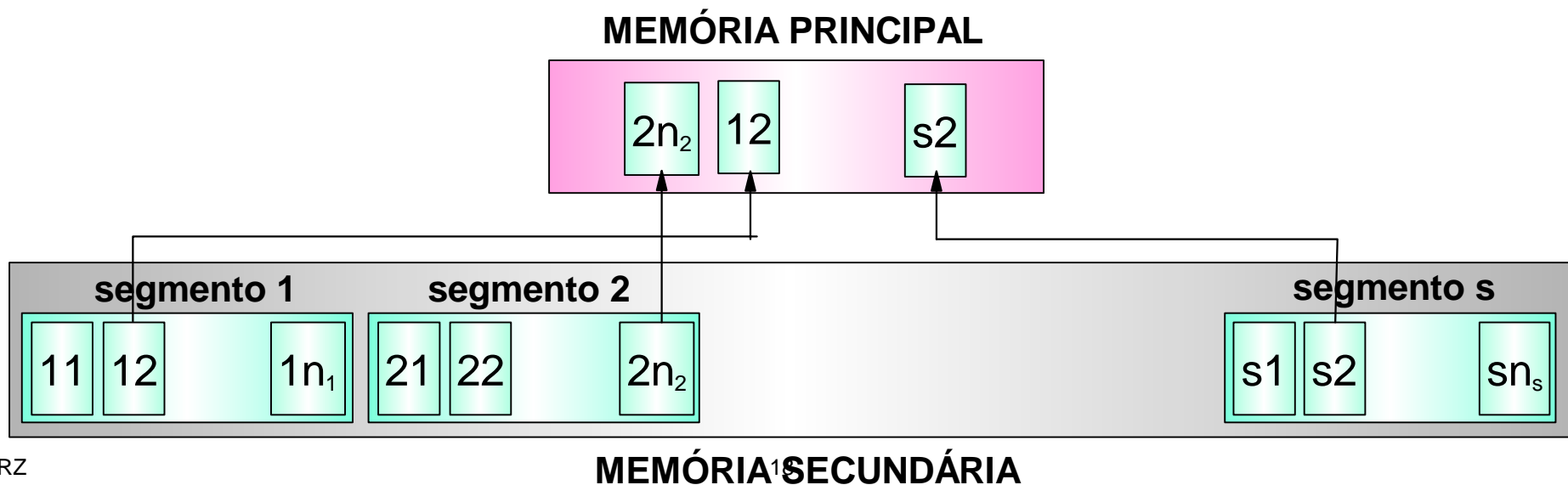
disco 1: 11110000
disco 2: 10101010
disco 3: 00111000
disco 4: 01000001
disco 5: 01100010
disco 6: 00011011
disco 7: 10001001

Gerenciador de Dados



Gerenciador de Dados

- Interface:
 - ▶ comandos internos utilizados pelo otimizador
- Organização da memória secundária:
 - ▶ memória secundária = seqüência de segmentos
 - ▶ segmento = seqüência de páginas
- Organização da memória principal:
 - ▶ buffer pool = conjunto de páginas



Gerência do Buffer Pool

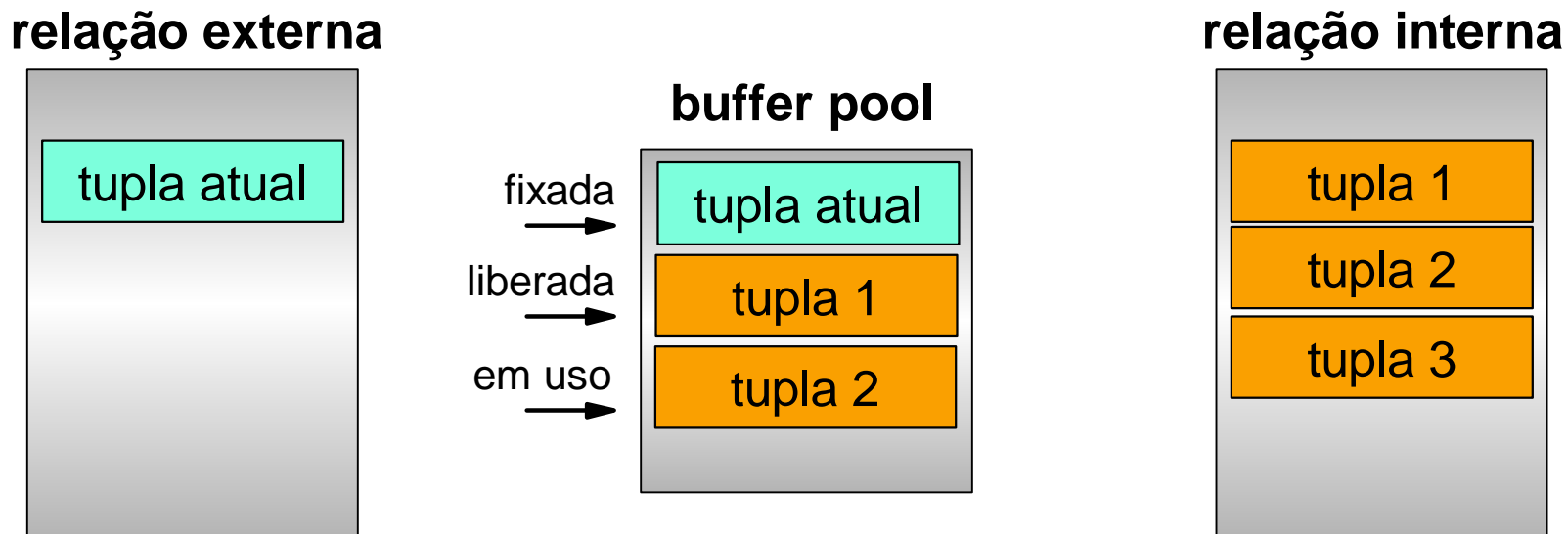
- Esquemas de Gerenciamento
 - ▶ tradicionais de sistemas operacionais
 - ▶ específicas de SGBDs
 - imobilização de buffers
 - necessária para os algoritmos de recuperação e em alguns casos de otimização de consultas
 - reescrita forçada de buffers
 - necessária para os algoritmos de recuperação

Gerência do Buffer Pool

■ Exemplos de Políticas Específicas

► junção aninhada:

- o buffer contendo a tupla corrente da relação externa deve ser fixado em memória até que a relação interna seja completamente percorrida



Gerência do Buffer Pool

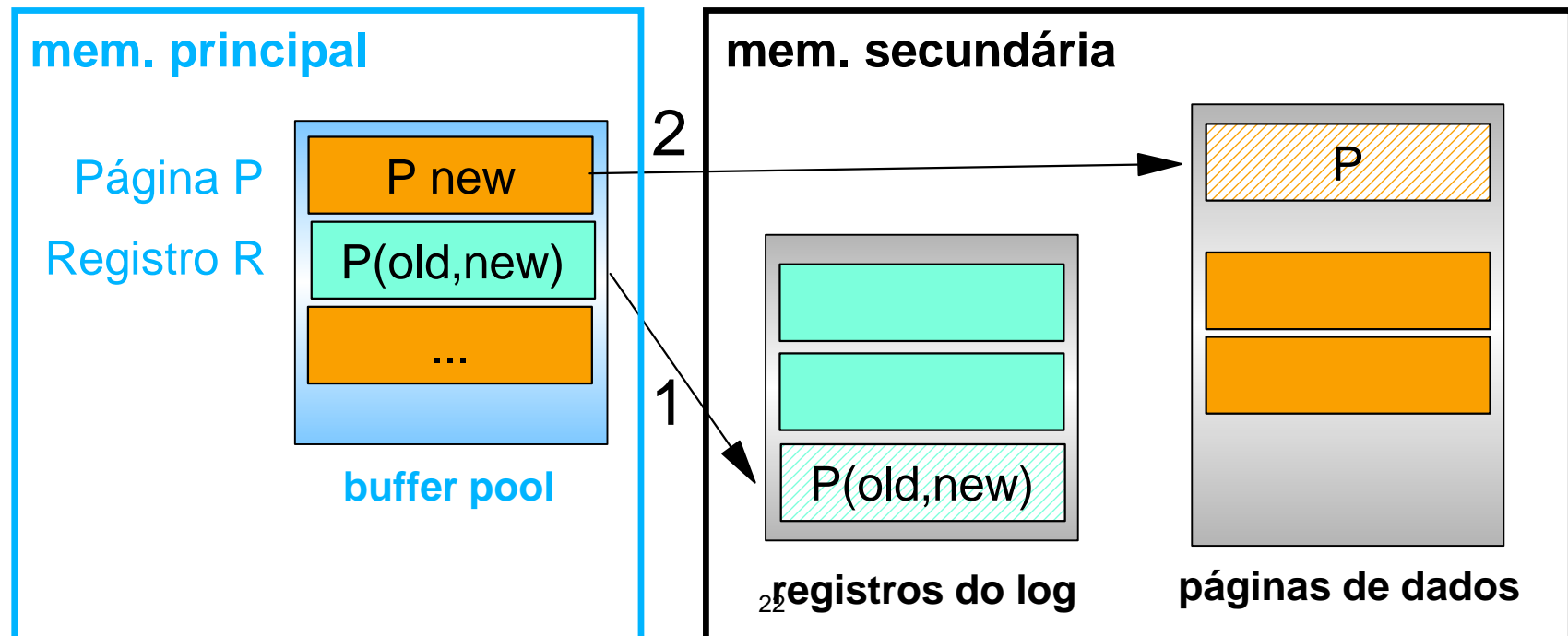
- Exemplos de Políticas Específicas
 - ▶ otimização de várias consultas em conjunto
 - Problema:
dado um conjunto de consultas que serão processadas em conjunto,
como minimizar o tráfego de páginas?

Gerência do Buffer Pool

■ Exemplos de Políticas Específicas

► algoritmos de recuperação:

- o registro R do log contendo o valor anterior e o novo valor de uma página P deve ser escrito no disco antes da nova versão de P



Armazenamento em SGBDs Convencionais

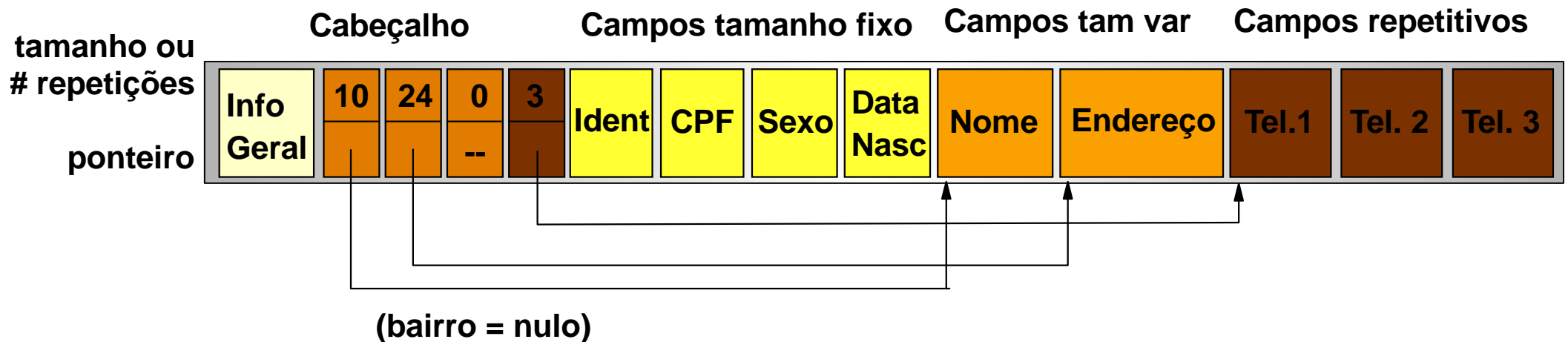
- Estrutura interna dos registros físicos:
 - ▶ como organizar os campos de uma tupla em um registro físico
- Estrutura interna das páginas:
 - ▶ como organizar os registros físicos em uma página
- Armazenamento de relações:
 - ▶ como organização as tuplas de uma ou mais relações nas páginas

Armazenamento em SGBDs Convencionais

- Estrutura interna dos registros físicos:
 - ▶ requisito obrigatório:
 - tratar campos:
 - tamanho fixo
 - tamanho variável
 - repetitivos
 - longos
 - ▶ requisitos desejáveis:
 - representar campos nulos eficientemente
 - armazenar campos longos sem interferir na recuperação dos campos convencionais

Armazenamento em SGBDs Convencionais

■ Exemplo:



■ Exercício:

- ▶ definir um esquema genérico para organização dos campos de um registro físico

Armazenamento em SGBDs Convencionais

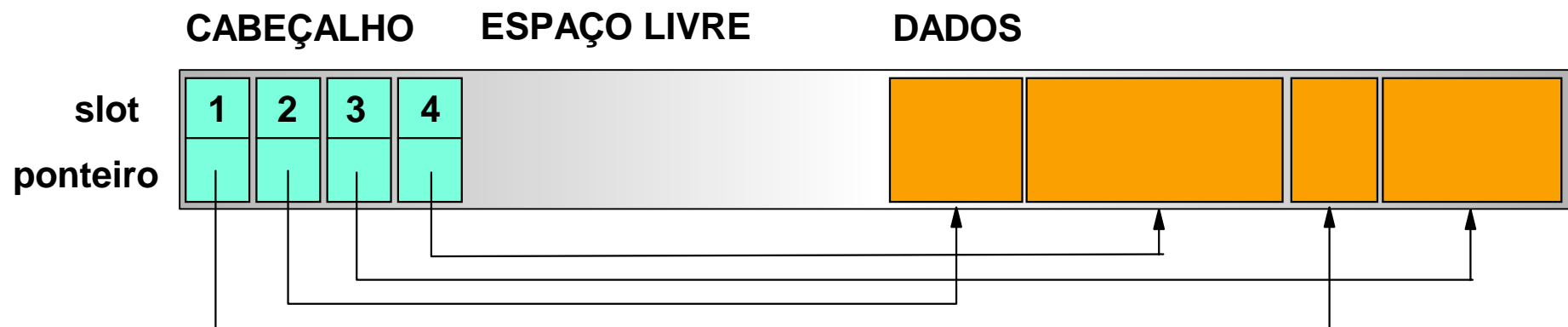
- Estrutura interna das páginas - Slotted-page:
 - ▶ introduz um nível de indireção para acessar os registros físicos armazenados em uma página
 - ▶ permite reorganizar os registros armazenados em uma página de forma simples
 - ▶ eficiência equivalente à de acesso direto aos registros

Armazenamento em SGBDs Convencionais

- Slotted-pages:

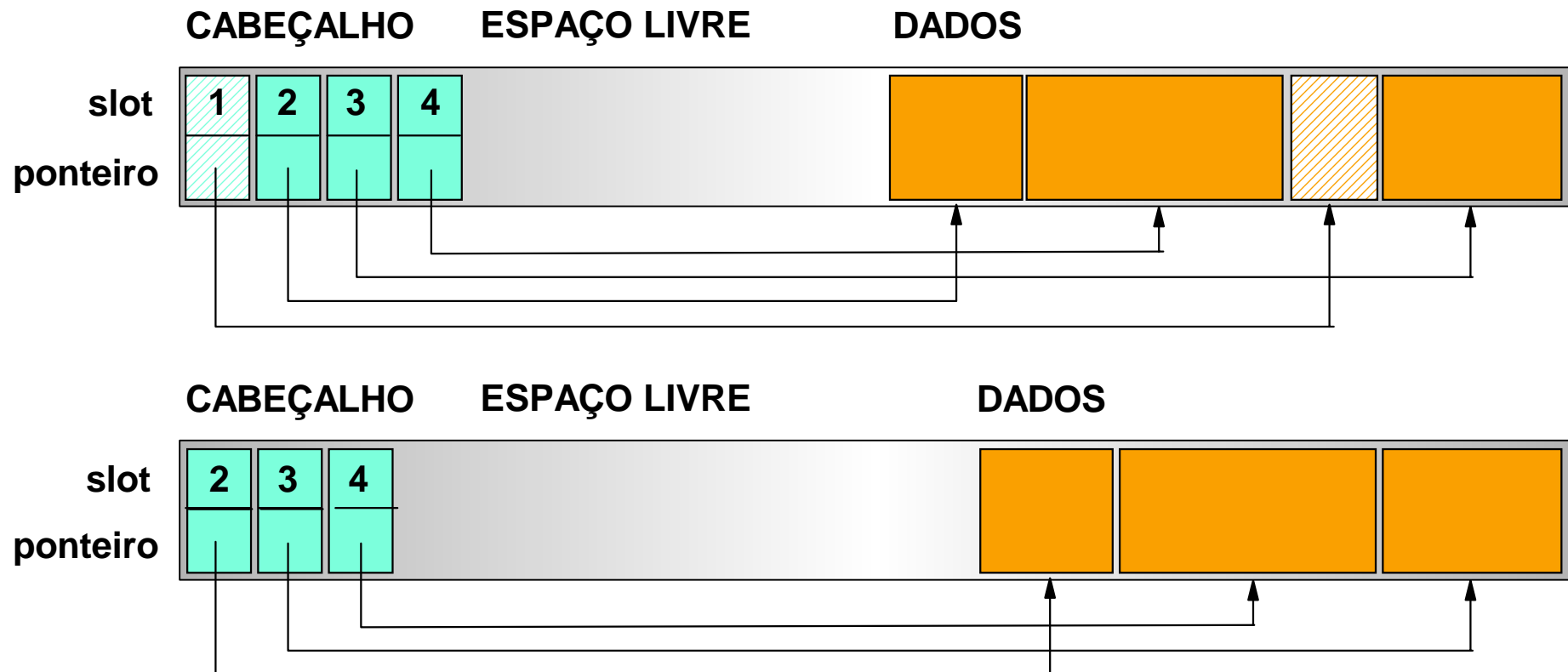
- ▶ record ID = (segment, page, slot)

- ▶ estrutura da página:



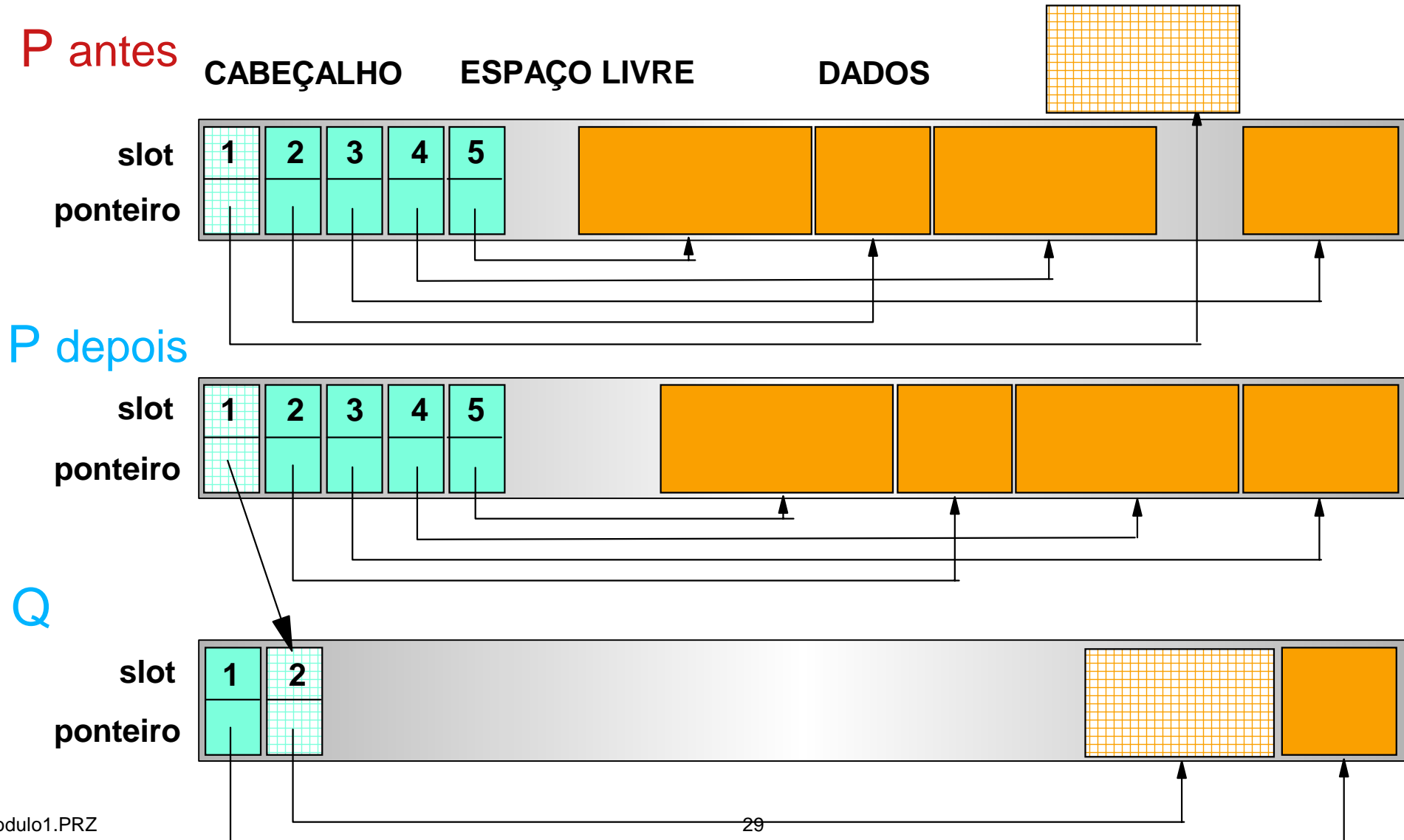
Armazenamento em SGBDs Convencionais

- Slotted-pages - remoção de registro:



Armazenamento em SGBDs Convencionais

- Slotted-pages - tratamento de overflow de página:



Armazenamento em SGBDs Convencionais

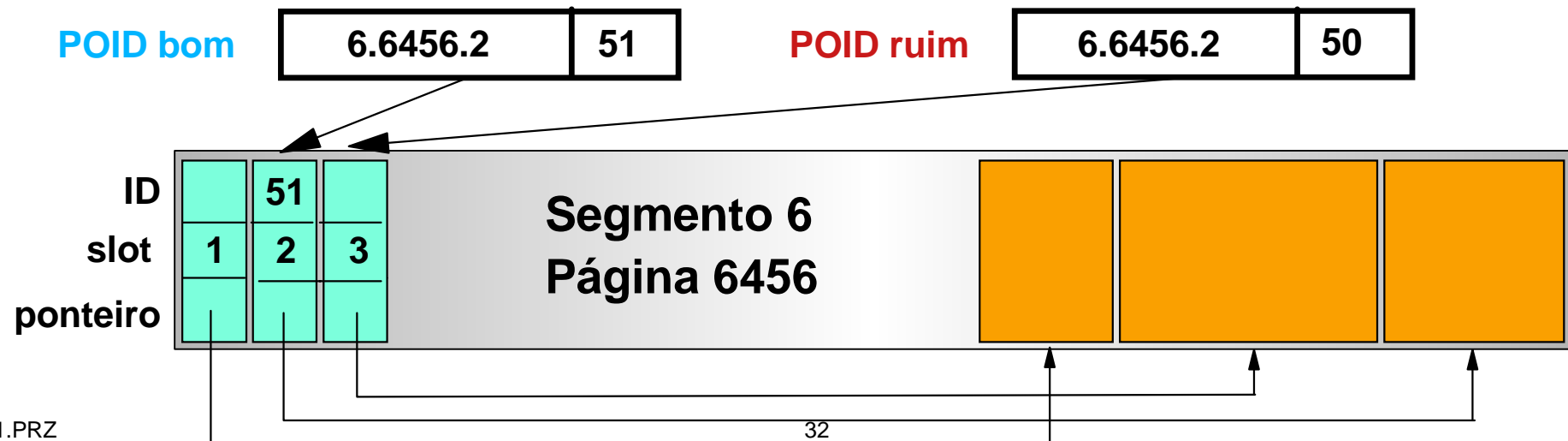
- Estratégias de Armazenamento de Relações:
 - ▶ heap
 - não há ordem no armazenamento dos registros
 - ▶ sequencial
 - armazenamento em ordem seqüencial de chave primária
 - ▶ hashing
 - armazenamento na ordem determinada por uma função de hashing aplicada a um conjunto de atributos
 - ▶ clustering
 - armazenamento de registros de várias relações, agregados por valor de chave

Armazenamento em SGBDs OO

- Identificadores para Objetos (OIDs):
 - ▶ OIDs lógicos (LOIDs):
 - não codificam diretamente a localização física do objeto
 - subsistema de armazenamento deve manter uma tabela mapeando cada LOID na localização física do objeto

Armazenamento em SGBDs OO

- Identificadores para Objetos (OIDs):
 - ▶ OIDs físicos (POIDs):
 - codificam a localização física do objeto
 - podem usar um esquema idêntico ao da slotted-page
POID = (segment, page, slot)
ou podem incluir ID do objeto para evitar problemas
POID = (segment, page, slot, ID)
 - exemplo de problema: POID acidentalmente reutilizado



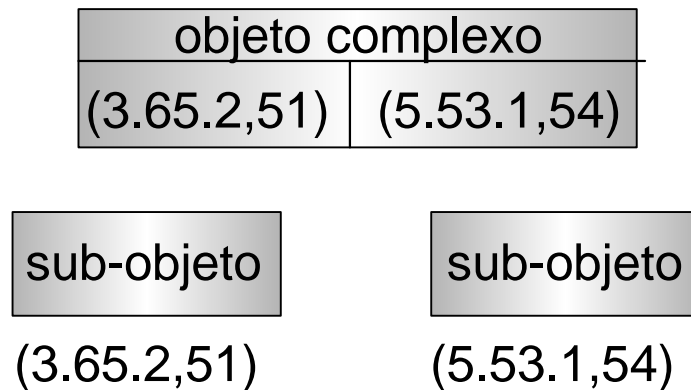
Armazenamento em SGBDs OO

■ Pointer Swizzling:

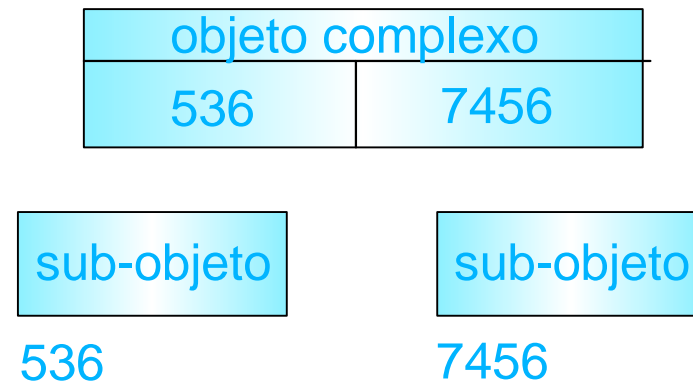
► Problema:

- quando um objeto complexo está armazenado no BD, OIDs são utilizados como ponteiros persistentes para os sub-objetos
- porém, quando o objeto complexo está em memória principal, ponteiros "normais" são mais apropriados

memória secundária

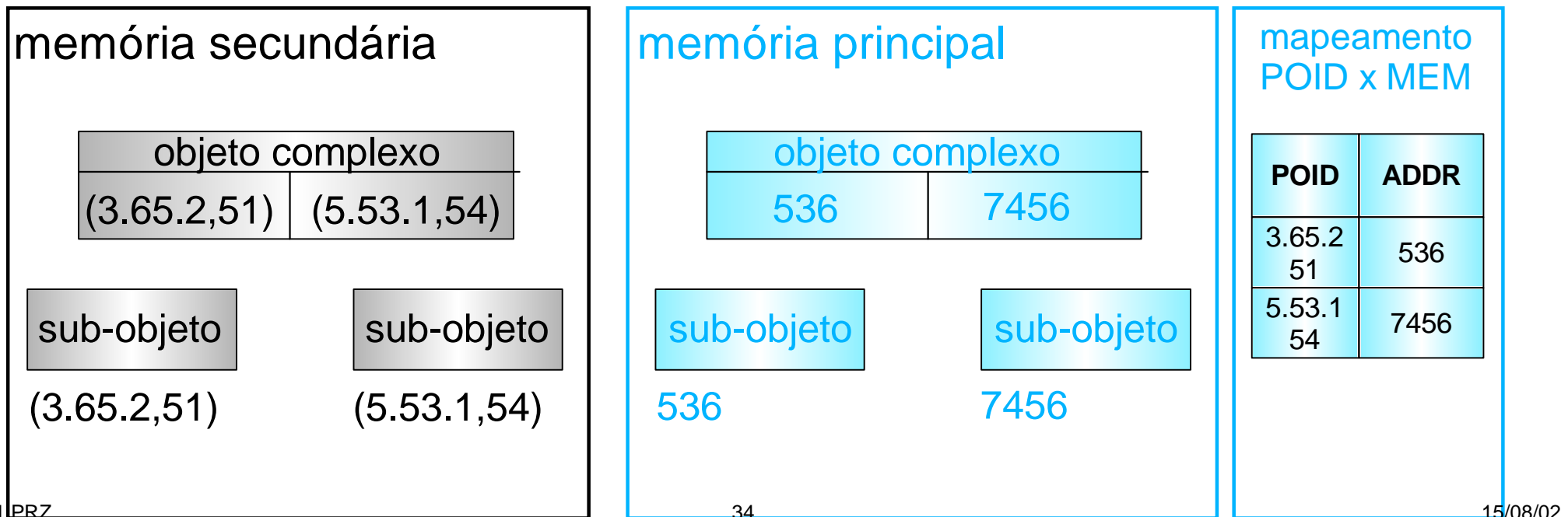


memória principal (c/ swizzling)



Armazenamento em SGBDs OO

- Software Pointer Swizzling:
 - ▶ quando uma aplicação navega por um ponteiro persistente em um objeto complexo pela primeira vez, o SGBD traz o sub-objeto referenciado para mem. principal e substitui o ponteiro persistente por um ponteiro "normal" apontando para a posição de memória utilizada pelo sub-objeto



Armazenamento em SGBDs OO

- Software Pointer Swizzling:
 - ▶ Alternativas:

Os OIDs dos sub-objetos podem ser trocados:

 - imediatamente quando o objeto complexo é acessado
 - posteriormente quando o componente é acessado
 - ▶ Vantagens:
 - trabalha a nível de objetos
 - ▶ Desvantagem:
 - realocação de sub-objetos em mem. principal requer atualização dos ponteiros
 - swizzling deve ser revertido quando o objeto é devolvido para mem. secundária

Armazenamento em SGBDs OO

- Hardware Pointer Swizzling:
 - ▶ utiliza o próprio mecanismo de paginação do sistema operacional
 - ▶ quando um objeto complexo é trazido para memória, os OIDs dos seus componentes são trocados por endereços de páginas virtuais fora do *working set* corrente
 - ▶ a navegação para um componente de um objeto complexo traduz-se no acesso à página virtual correspondente, gerando uma *page fault*, que é tratada pelo SGBD-OO

Armazenamento em SGBDs OO

■ Hardware Pointer Swizzling:

- ▶ utiliza uma tabela de tradução interna a cada página
 - Identificador de página IDP = apontador para a tabela de tradução, com o mesmo comprimento de um ponteiro "normal"
 - informação extra indica onde estão os pointers dentro dos objetos
- ▶ o SGBD-OO mantém um mapeamento auxiliar indicando quais páginas em mem. secundária já foram mapeadas para memória principal

Página P

Objeto 1		Objeto 2		Objeto 3	
IDP	Off.	IDP	Off.	IDP	Off.
2395	5	4867	9	2395	25

IDP	ID completo da pag
2395	679.24
4867	519.56

info extra sobre localização dos pointers

Mapeamento

Pag Mem	Pag Sec

Armazenamento em SGBDs OO

- Hardware Pointer Swizzling:
 - ▶ quanto o SGBD-OO traz uma página P do BD para mem. principal:
 - executa o swizzling para todos os ponteiros persistentes de objetos em P, alterando os IDPs
 - consulta e atualiza o mapeamento auxiliar

Página P (já em memória)

Objeto 1		Objeto 2		Objeto 3	
IDP	Off.	IDP	Off.	IDP	Off.
5001	5	4867	9	5001	25

IDP	ID completo da pag
5001	679.24
4867	519.56

5001

frame de página de
memória principal
alocado para a pág 679.24

4867

frame de página de
memória principal
alocado para a pág 519.56

Mapeamento

Pag Mem	Pag Sec
5001	679.24
4867	519.56

nota: o endereço do frame de memória pode ser o próprio IDP, no exemplo, o IDP 4867

Armazenamento em SGBDs OO

■ Hardware Pointer Swizzling:

- ▶ quanto a aplicação navegar pelo ponteiro 5001.5, um page fault será gerado, sendo tratado pelo SGBD-OO
- ▶ o SGBD-OO traz a página 679.24 para o frame 5001

Página P (já em memória)

Objeto 1

IDP	Off.
5001	5

Objeto 2

IDP	Off.
4867	9

Objeto 3

IDP	Off.
5001	25

IDP	ID completo da pag
5001	679.24
4867	519.56

5001

página de memória principal
contendo a página 679.24

4867

frame de página de memória principal
alocado para a página 519.56

Armazenamento em SGBDs OO

- Hardware Pointer Swizzling:
 - ▶ quanto o subsistema de armazenamento retornar P para memória secundária, não precisará desfazer o swizzling !

Página P

Objeto 1		Objeto 2		Objeto 3	
IDP	Off.	IDP	Off.	IDP	Off.
5001	5	4867	9	5001	25

IDP	ID completo da pag
5001	679.24
4867	519.56