



Administração de banco de dados

Aula 09 - Indexação

Gustavo Maia

Gustavo.Maia@FaculdadeImpacta.com.br

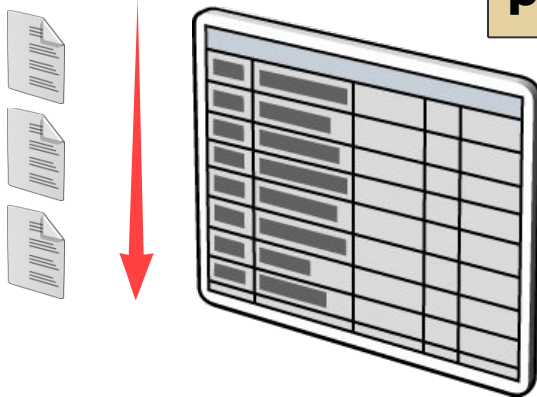
Agenda

▪ Indexação

- O que são índices / definições
- Estrutura dos índices (árvores)
- Prós e Contras (bons índices vs maus índices)
- Tipos de índices
 - Heap
 - Clustered
 - NonClustered
- Manutenção:
 - Fragmentação

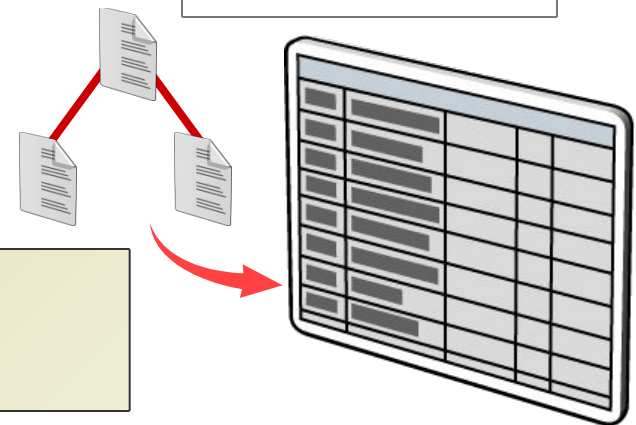
Como o banco acessa dados

Table Scan



O banco faz a leitura de todas as páginas para encontrar registros

Index



O banco usa as páginas de índices para encontrar registros

Necessidade dos Índices

- **ANSI SQL não menciona índices**
 - Geralmente são considerados como sendo externo a lógica de modelo de dados
- **Qualquer consulta pode ser executada sem índices**
 - O desempenho pode ser significativamente melhorado pela presença de índices
- **Algumas restrições são implementadas através de índices**
 - Os índices podem ser utilizados para aplicar restrições (constraints) eficientes, mas teoricamente, podem ser implementadas de outras maneiras

Necessidade dos Índices

- **Analogia: Uma biblioteca física para livros**
 - Índice por autor poderia ser útil
 - Índices adicionais também poderiam ser úteis como:
Editora, título, ISBN, ...

Estrutura dos Índices

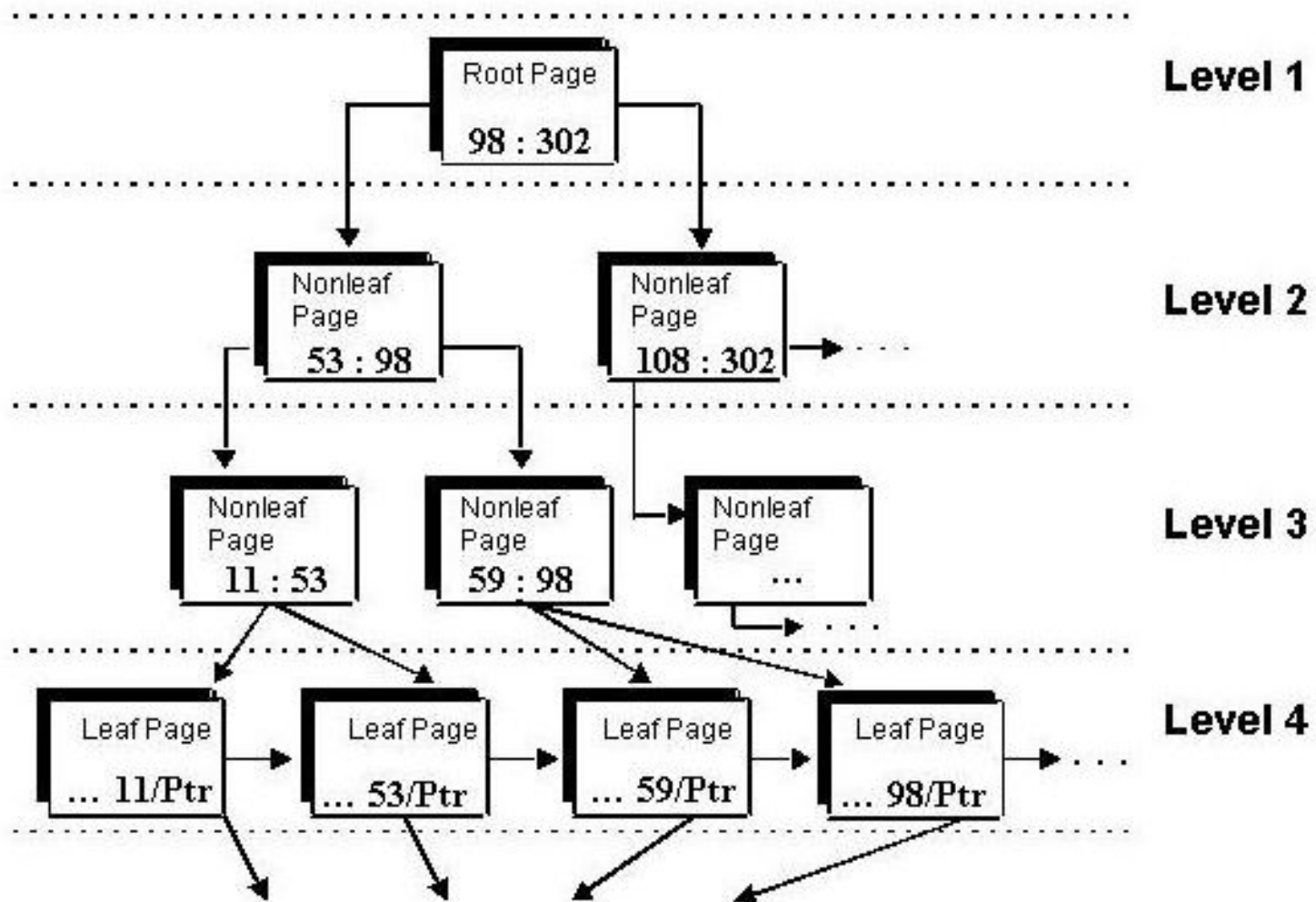
- Os índices são geralmente baseados em estruturas de árvore, mais especificamente em árvores balanceadas (b-tree)
 - Não somente como árvore binária, os nós podem ter mais de um filho
 - São chamadas “balanceadas” pois sua estrutura faz com que qualquer que seja o elemento procurado, o banco de dados faz com que sejam lidas as mesmas quantidades de páginas para encontrar o resultado, ou seja, independente do elemento procurado o custo para encontrá-lo será igual a qualquer outro elemento

Estrutura dos Índices

- O nó superior é chamado de página RAIZ (root page)
- Nós de nível inferiores (intermediários) são chamados de páginas não FOLHA (nonleaf page)
- Os nós de último nível são chamados de páginas FOLHA (leaf page)
- O banco de dados é responsável por distribuir os níveis e número de páginas adequadamente

Estrutura dos Índices

Figure 1. The structure of a b-tree index.



Seletividade, Densidade e Profundidade

- **Selectivity**

- É uma medida de quantas linhas são retornadas em comparação com o número total de linhas
- Elevada seletividade significa um pequeno número de linhas, quando relacionado com o número total de linhas

- **Density**

- Uma medida para a falta de especificidade dos dados na tabela
- Alta densidade indica um grande número de duplicados

- **Index Depth**

- Número de níveis dentro do índice
- Equívoco comum é pensar que os índices possuem muita profundidade

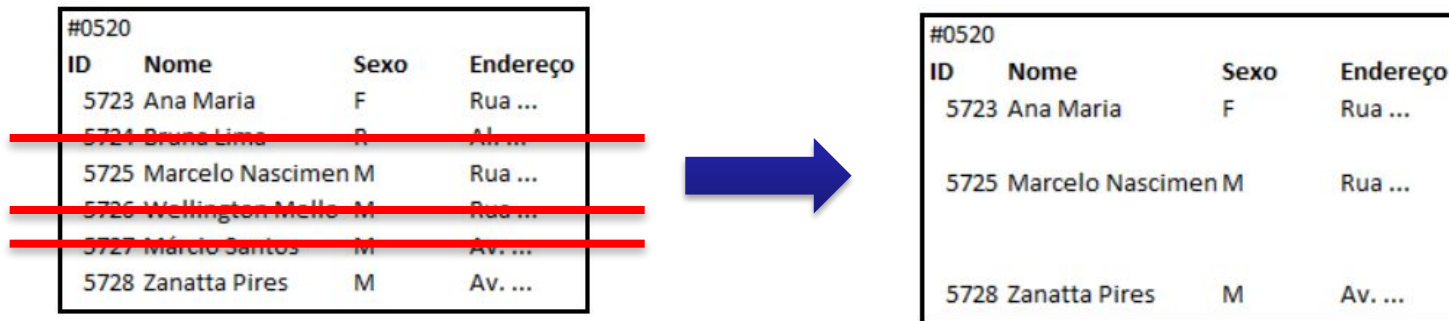
Fragmentação de Índices

- **Como é que a fragmentação ocorre ?**
 - O banco reorganiza páginas de índice quando os dados são modificados (inserção, deleção, update), causando divisões de páginas
- **Tipos de fragmentação:**
 - Internal – páginas não estão totalmente preenchidas
 - External – páginas estão fora de uma sequência lógica

Fragmentação de Índices

- **Fragmentação Interna**

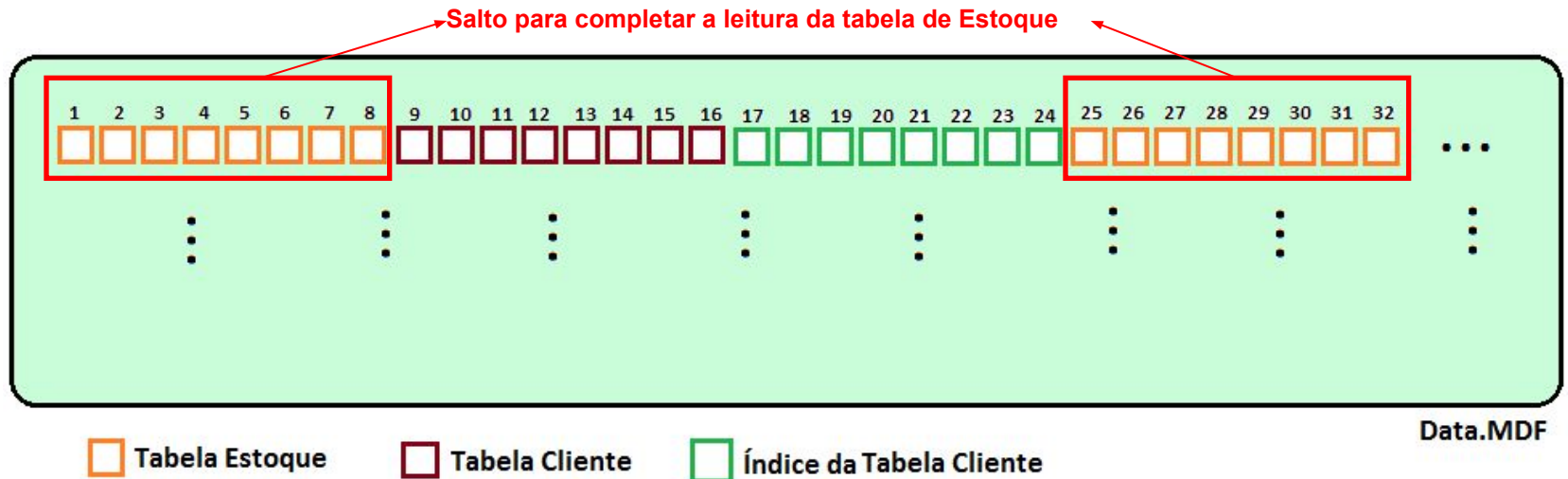
- Ocorrem dentro das páginas de dados. **Deleções de registros** podem deixar as páginas pouco preenchidas, o que leva a ter uma maior quantidade de páginas para armazenar os mesmos dados



Fragmentação de Índices

- **Fragmentação Externa**

- Ocorrem dentro dos arquivos de dados. Durante o uso do sistema, novas páginas serão necessárias para armazenar registros e estas poderão ficar separadas fisicamente, ou seja, para fazer uma leitura da tabela, teremos saltos entre as páginas de dados, o que pode afetar negativamente o desempenho.



Fragmentação de Índices

- **Detectando fragmentação**
 - SQL Server Management Studio – propriedades do índice
 - Funções de Sistema–
 - sys.dm_db_index_physical_stats
 - sys.dm_db_index_usage_stats

Índices Simples X Índices Compostos

- **Índices nem sempre são construídos numa única coluna**
 - Índices com apenas uma coluna são chamados ÍNDICES SIMPLES
 - Índices com várias colunas são chamados ÍNDICES COMPOSTOS
- **Índices COMPOSTOS são frequentemente úteis**
 - Tendem a ser mais útil do que os índices simples, na maioria dos aplicativos típicos de negócios
 - Tendo um índice classificado pelo cliente e em seguida, por ordem de data do pedido, faz com que seja muito fácil de encontrar as encomendas de um cliente específico em uma determinada data
 - Uma consulta pode envolver vários predicados de pesquisa
 - Duas colunas em conjunto podem ser bem mais seletivas que dois índices cada um apontado para uma destas colunas
- **Índice de A, B não é o mesmo que um índice de B, A**
 - Recomendação é indexar a(s) coluna(s) mais restritiva(s) primeiro

Índices Crescentes X Decrescentes

- Índices podem ser construídos em ordem crescente ou decrescente
- Em geral, para os índices de coluna única, ambos são igualmente úteis
 - Cada nível de um índice é ligado duplamente (ou seja ligado em ambos os sentidos)
 - O banco pode começar em cada uma das extremidades e trabalhar em direção a outra extremidade
- Cada componente de um índice composto pode ser crescente ou decrescente
 - Pode ser muito útil para evitar operações de classificação (sort)

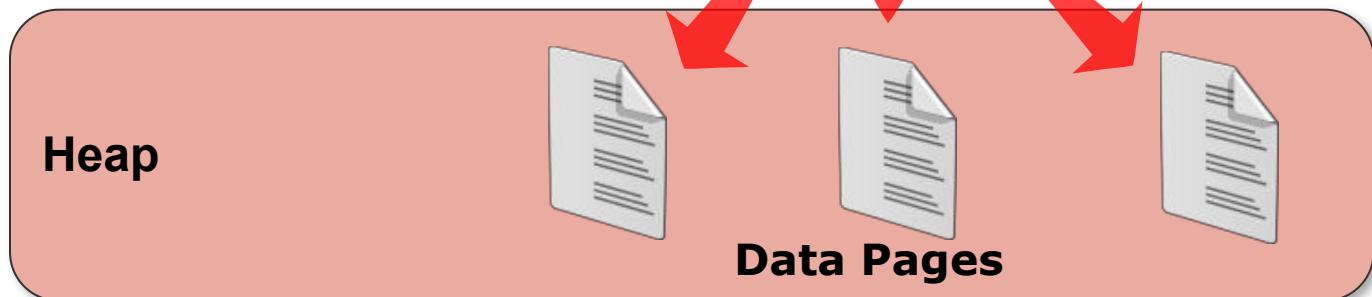
Estatísticas de Índices

- **O banco de dados precisa ter conhecimento do layout dos dados em uma tabela ou índice antes de otimizar e executar consultas**
 - Precisa criar um plano razoável para executar a consulta
 - Importante saber a utilidade de cada índice
 - A selectividade é a métrica mais importante
- **Por padrão, o SQL Server cria automaticamente estatísticas sobre índices**
 - Pode ser desativado
 - Recomendação é deixar auto-criação e de auto-atualização habilitada

Índices – O que é uma Heap

- Uma tabela sem um índice Clustered
- Páginas são armazenadas sem qualquer tipo de ordem
- Os dados que são inseridos ou modificados podem ser colocados em qualquer lugar dentro da tabela

id	index_id=0	first_iam_page
----	------------	----------------



Operações em Heaps

- **INSERT**

- Cada nova linha pode ser colocada na primeira página disponível com espaço suficiente

- **UPDATE**

- A linha pode permanecer na mesma página se ainda se encaixar (espaço para armazenamento igual ou inferior a atualização), de outro modo, pode ser removida da página atual e colocada na primeira página disponível com espaço suficiente

- **DELETE**

- Libera espaço na página atual
- Os dados não são substituídos, o espaço é apenas marcado como disponível para reutilização

- **SELECT**

- Tabela inteira precisa ser lida para a maioria das consultas, caso não haja índices disponíveis

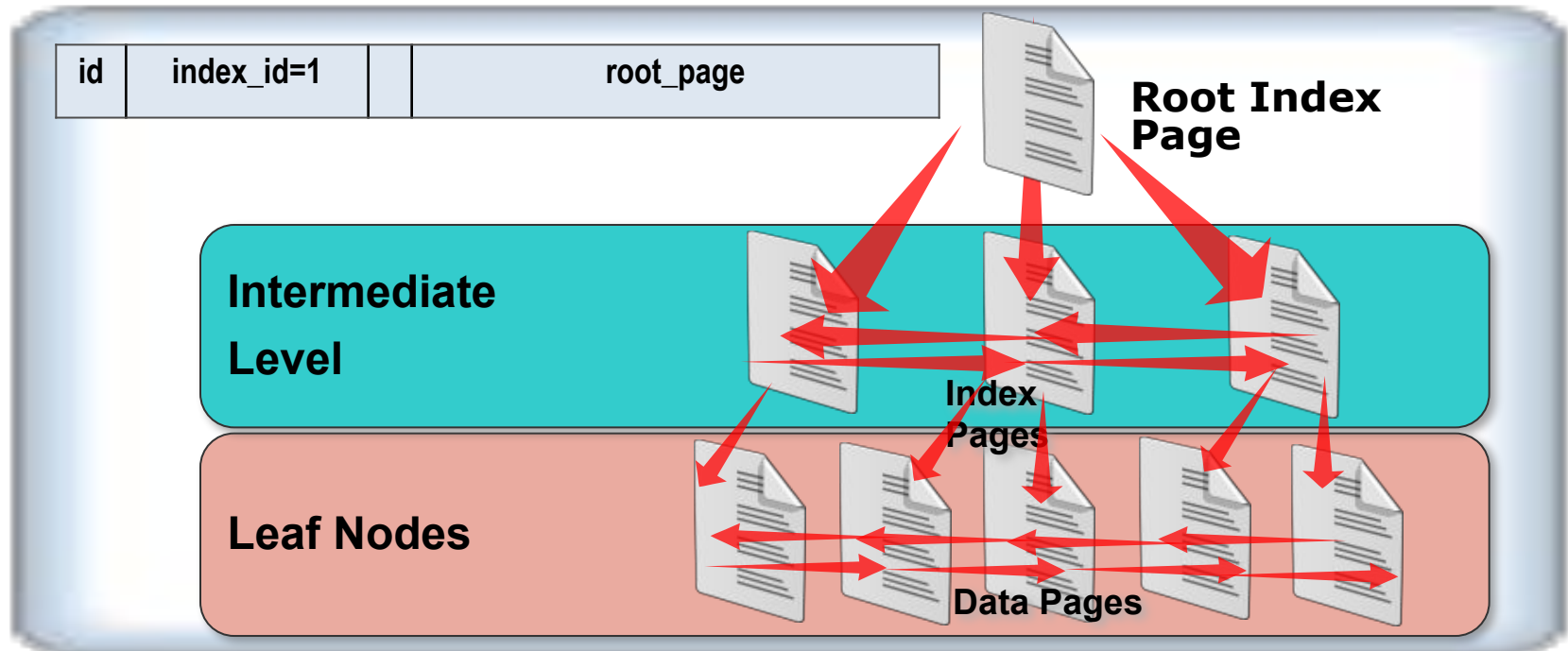
Ponteiros de encaminhamento

Ponteiros de encaminhamento são referências deixadas no local original de uma linha, quando esta linha foi movida.

- **Modificações de dados em HEAPs podem deixar ponteiros de encaminhamento**
 - Row IDs em outros índices não precisam ser atualizados
 - Pode levar a problemas de desempenho ao longo do tempo
 - Apenas um único ponteiro de encaminhamento é usado
- **Desempenho seria melhorado através da remoção de ponteiros de encaminhamento e atualização de outros índices**
 - Sem opção fácil para isto antes de SQL Server 2008
 - SQL Server 2008 e versões posteriores introduziram a capacidade para reconstruir uma tabela (incluindo uma HEAP)
 - `ALTER TABLE <tablename> REBUILD`

Índices Clustered

- Páginas de dados da tabela são armazenadas em ordem lógica
- Linhas são armazenadas em ordem lógica dentro das tabelas
- Só pode existir um índice clustered por tabela



Operações em Índices Clustered

- **INSERT**

- Cada nova linha deve ser colocada na posição lógica correta
- Pode envolver divisões (splits) de páginas da tabela

- **UPDATE**

- A linha pode permanecer no mesmo lugar, se ainda se encaixar e se o valor da chave da ordem lógica ainda for o mesmo
- Se a linha não se encaixar mais na página, a página precisa ser dividida
- Se a chave de cluster mudar, a linha tem de ser removida e colocada na posição correta dentro da lógica da tabela

- **DELETE**

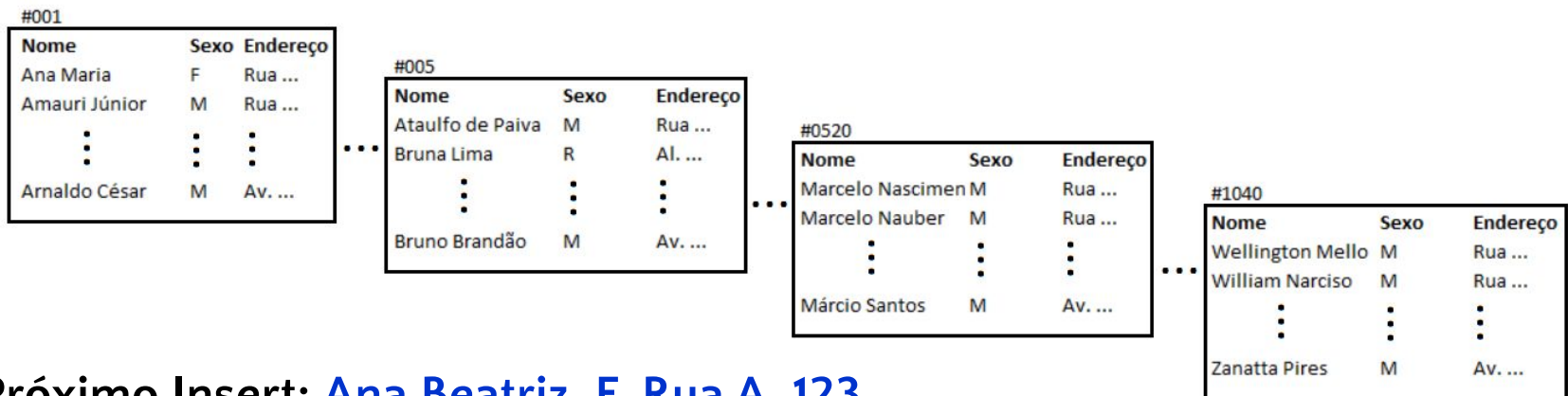
- Libera espaço marcando os dados como não utilizados

- **SELECT**

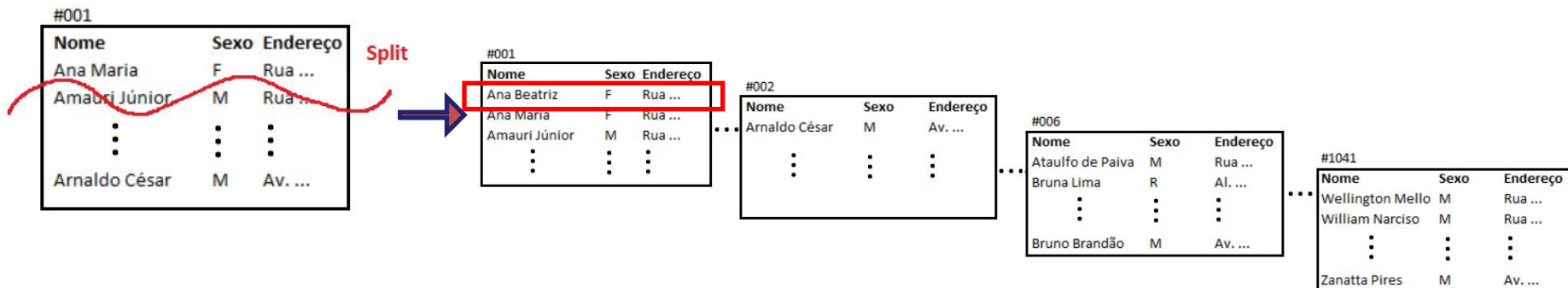
- Pesquisas relacionadas com a chave de cluster realizam procura (seek)
- Pesquisas relacionadas com a chave de cluster podem varrer (scan) e evitar classificação (sort)

Índices Clustered

- A escolha do(s) campo(s) para o índice cluster é a chave do sucesso ou de péssima performance da tabela.
- Exemplo de Cluster pelo campo **Nome**.



□ Próximo Insert: **Ana Beatriz, F, Rua A, 123**



Índices Clustered

- A escolha do(s) campo(s) para o índice cluster é a chave do sucesso ou de péssima performance da tabela.
 - Exemplo de Cluster pelo campo ID.

The diagram shows four nodes in a sequence, each containing a table with the following structure:

ID	Nome	Sexo	Endereço
1	Bruno Brandão	M	Av. ...
2	Marcelo Nauber	M	Rua ...
3	Amauri Júnior	M	Rua ...
⋮	⋮	⋮	⋮

Node #001 is connected to Node #005, which is connected to Node #0520, which is connected to Node #1040. Each node's table contains three data rows and a continuation symbol (⋮) for the last row.

📌 Próximo Insert: Ana Beatriz, F, Rua A, 123

The diagram shows a sequence of four nodes, each containing a table with the following structure:

ID	Nome	Sexo	Endereço
1	Bruno Brandão	M	Av. ...
2	Marcelo Nauber	M	Rua ...
3	Amauri Júnior	M	Rua ...
⋮	⋮	⋮	⋮

Node #001 is connected to Node #005, which is connected to Node #0520, which is connected to Node #1040. Node #1040 contains the following data:

ID	Nome	Sexo	Endereço
13311	Arnaldo César	M	Av. ...
13312	Zanatta Pires	M	Rua ...
⋮	⋮	⋮	⋮
13413	Ataulfo de Paiva	M	Rua ...
13414	Ana Beatriz	F	Rua ...

The last two rows of Node #1040 (13413 and 13414) are highlighted with a red border.

Índices Clustered Exclusivos (Unique) X Não Exclusivos

- **Clustered indexes podem ser**
 - Unique (exclusivos)
 - Non-Unique (não Exclusivos)
- **Em qualquer um dos casos, o banco de dados deve ser capaz de identificar linhas como únicas**
 - Nos índices não exclusivos o SQL Server adiciona um identificador (uniquifier - 4 bytes de tamanho) quando necessário, garantindo assim que a linha será única
- **A recomendação é sempre especificar o índice como UNIQUE se os dados tiverem unicidade**

Overview sobre criação de índices

Use SQL Server Management Studio


-OR-

CREATE INDEX Transact-SQL statement

```
CREATE [UNIQUE] [CLUSTERED | NONCLUSTERED ]  
    INDEX index_name ON { table | view } ( column [ ASC  
| DESC]  
    [ , . . . n ] )  
    INCLUDE ( column [ , . . . n ] )  
    [ WITH option [ , . . . n ] ]  
    [ ON { partition_scheme (column) | filegroup |  
"default" } ]
```


Índices exclusivos (unique)

```
CREATE UNIQUE NONCLUSTERED INDEX  
[AK_Employee_LoginID]  
ON [HumanResources].[Employee] ( [LoginID] ASC )
```



EmployeeID	LoginID	Gender	MaritalStatus	...
216	mike0	M	S	...
231	fukiko0	M	M	...
242	pat0	M	S	...

...



291	pat0	F	S	...
-----	------	---	---	-----



**Duplicate key value
not allowed**

Criação de índices compostos

Índices Compostos

- **Incluem até 16 colunas e 900 bytes na chave**
- **Definir colunas de maior unicidade primeiro**

```
CREATE INDEX K_Contact_LastName_FirstName ON  
Person.Contact ( LastName ASC, FirstName DESC)
```

Criando Índices Clustered

- Pode ser criado especificando uma chave primária na tabela ou diretamente

```
CREATE TABLE dbo.Article  
( ArticleID int IDENTITY(1,1) PRIMARY KEY,  
  ArticleName nvarchar(50) NOT NULL,  
  PublicationDate date NOT NULL  
);
```

```
CREATE TABLE dbo.LogData  
( LogID int IDENTITY(1,1),  
  LogData xml NOT NULL  
);
```

```
ALTER TABLE dbo.LogData  
  ADD CONSTRAINT PK_LogData  
  PRIMARY KEY (LogId);
```

```
CREATE CLUSTERED INDEX CL_LogTime  
  ON dbo.LogTime(LogTimeID);
```

Removendo Índices Clustered

- Remover um índice externo via **DROP INDEX**
- Remover uma restrição (constraint) **PRIMARY KEY** via **ALTER TABLE**
 - Pode não ser possível quando existir referências de chave estrangeira

```
DROP INDEX CL_LogTime ON dbo.LogTime;
```

```
ALTER TABLE dbo.LogData  
DROP CONSTRAINT PK_LogData;
```

Alterando Índices Clustered

- Algumas modificações são permitidas via **ALTER INDEX**
 - Com **REBUILD** ou **REORGANIZE**
 - Com **DISABLE**
- Não podemos modificar as colunas chaves dos índices
 - **CREATE INDEX WITH DROP_EXISTING** pode fazer isto

Incorporando espaços em Índices

- Espaço livre pode ser deixado em índices, incluindo índices clustered
 - FILLFACTOR
 - PAD_INDEX
- Espaço livre pode melhorar o desempenho de certas operações
- O valor padrão pode ser alterado usando sp_configure

```
ALTER TABLE Person.Contact  
ADD CONSTRAINT PK_Contact_ContactID  
PRIMARY KEY CLUSTERED  
(  
    ContactID ASC  
) WITH (PAD_INDEX = OFF, FILLFACTOR = 70);  
GO
```

Incorporando espaços em Índices

Espaços livres afetam a performance de update dos índices

- **FILLFACTOR** determina o montante de páginas livres nos nós folhas (leaf)
 - ✱ Use baixo FILLFACTOR para aplicações OLTP
 - ✱ Use alto FILLFACTOR para aplicações OLAP
- **PAD_INDEX** determina o montante de páginas livres em nós não folhas (non-leaf)

```
CREATE UNIQUE NONCLUSTERED INDEX  
[AK_Employee_LoginID]  
ON [HumanResources].[Employee]  
([LoginID] ASC)  
WITH (FILLFACTOR = 65, PAD_INDEX = ON)
```


Características de Bons Índices Clustered

- **Boas chaves de clustered tem propriedades específicas**
 - Curtas (tamanho pequeno)
 - Estáticas (não sofrem modificações)
 - Crescentes
 - Exclusivas (unique)
- **Limites das chaves de clustered**
 - 16 colunas
 - 900 bytes

Tipos de dados apropriados para Índices Clustered

Tipo de Dados	Comentário
int	Bom candidato, particularmente se for IDENTITY
bigint	Bom candidato, particularmente se for IDENTITY
uniqueidentifier	Algumas preocupações com o tamanho e grande preocupação com a falta de valores crescentes
varchar	Preocupações com o tamanho e com desempenho de classificação. Preocupações com a natureza estática
date	A preocupação com a unicidade, mas excelente para consultas por abrangência (between)
smalldatetime	Preocupações com unicidade

Persistindo Dados pelo uso de índices

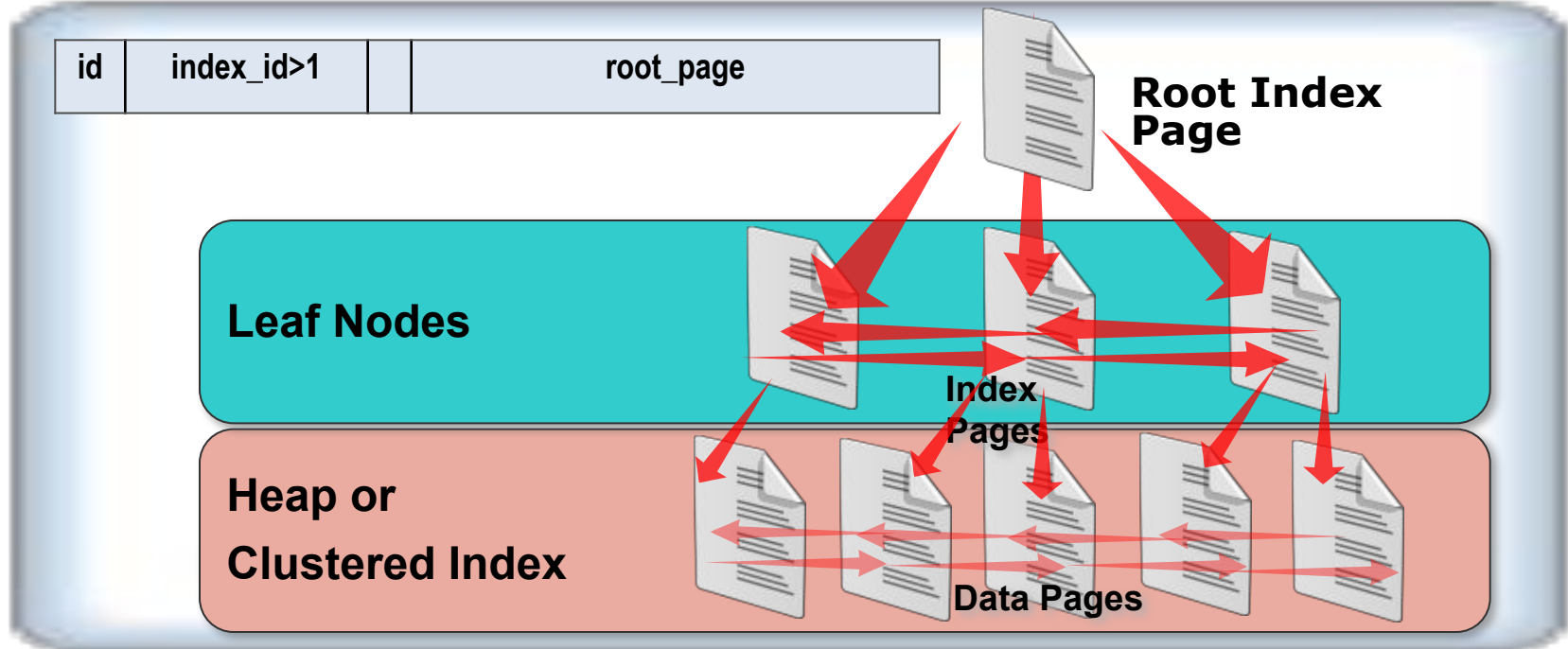
- Criando um clustered index em uma VIEW persiste o conjunto de dados que a consulta retorna
- Consultas serão mais rápidas porque o conjunto de dados, incluindo agregações e junções, já foram criadas
 - O otimizador de consulta pode utilizar um índice que se baseia em uma visão, mesmo se a view não está incluída na cláusula FROM de uma consulta (Enterprise Edition)
 - Use visões indexadas (indexed views) para views frequentemente usadas que possuem lógica complexa
 - Evite o uso de visões indexadas para views que raramente são executadas

O que são Índices Nonclustered ?

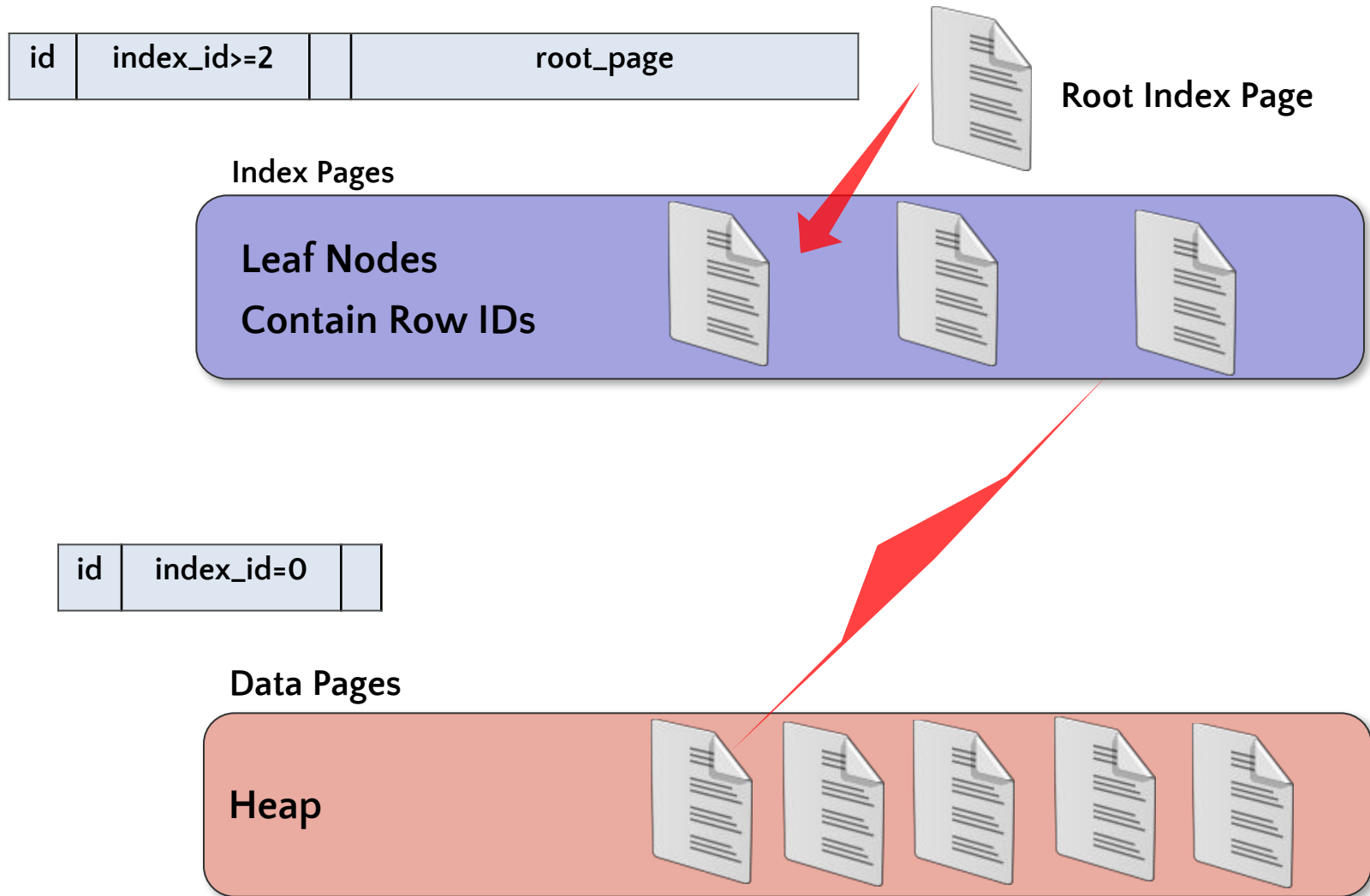
- **Tabelas são estruturadas como uma heap ou índice clusterizado**
 - Heap = Index ID 0
 - Clustered index = Index ID 1
- **Índices adicionais podem ser criados**
 - Eles são chamados de índices não clusterizados
 - Eles também são baseados em B-Trees
 - Níveis Folha apontam para estrutura de tabela base em vez de dados
 - Nonclustered indexes = Index ID 2 ou posteriores
 - Eles podem melhorar o desempenho das consultas usadas com frequência
 - O impacto no desempenho de modificação de dados precisa ser considerado

Índice Nonclustered

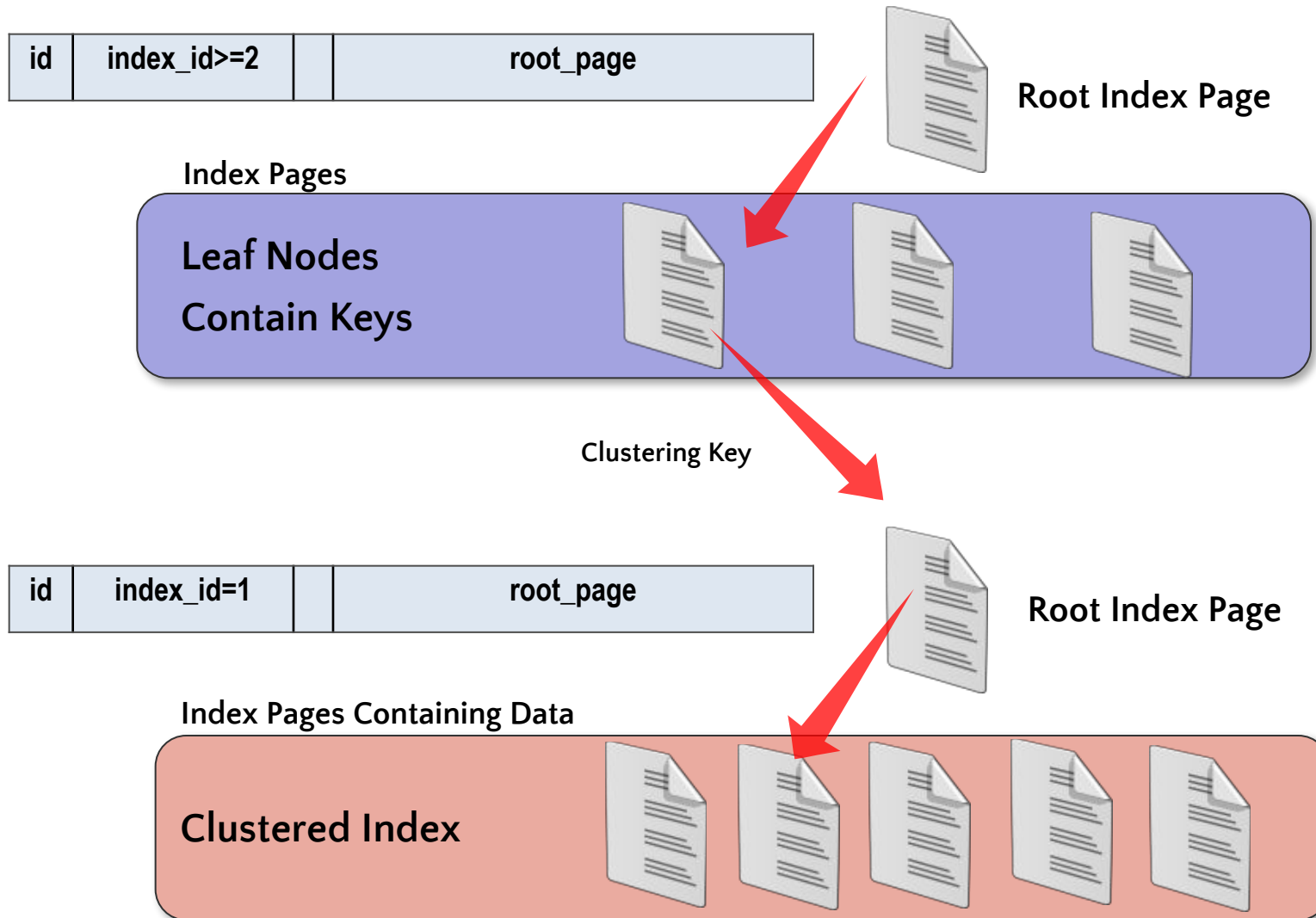
- Mesmas referências B-tree que heap ou clustered index
- Até 249 nonclustered indexes por tabela



Índices Nonclustered sobre Heaps



Índices Nonclustered sobre Clustered Indexes



Criando Nonclustered Indexes

Use a instrução **CREATE INDEX** para especificar:

- Um nome para o índice
- A tabela a ser indexada
- As colunas que compõem a chave de índice

```
CREATE TABLE dbo.Book  
( ISBN nvarchar(20) NOT NULL PRIMARY KEY,  
  Title nvarchar(50) NOT NULL,  
  ReleaseDate date NOT NULL,  
  PublisherID int NOT NULL  
);  
GO  
CREATE NONCLUSTERED INDEX IX_Book_Publisher  
  ON dbo.Book (PublisherID, ReleaseDate DESC);  
GO
```


Cláusula INCLUDE

- Índices de cobertura (covering indexes) pode aumentar significativamente o desempenho de consultas
- Os índices não clusterizados podem usar a cláusula INCLUDE

```
CREATE NONCLUSTERED  
INDEX  
AK_Employee_LoginID ON  
HumanResources.Employee  
( LoginID ASC) INCLUDE  
( ContactID,  
NationalIDNumber )
```

Colunas Included

- **Colunas não chave podem ser incluídas no índice**
- **Melhora a performance de consultas de cobertura**

Cláusula INCLUDE

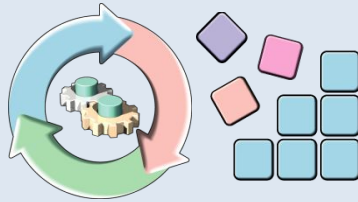
- A cláusula INCLUDE permite o armazenamento de colunas de dados selecionados a nível folha de um índice nonclustered

```
CREATE NONCLUSTERED INDEX IX_Book_Publisher  
ON dbo.Book (PublisherID, ReleaseDate DESC)  
INCLUDE (Title);  
GO
```

```
SELECT PublisherID, Title, ReleaseDate  
FROM dbo.Book  
WHERE ReleaseDate > DATEADD(year,-1,SYSDATETIME())  
ORDER BY PublisherID, ReleaseDate DESC;  
GO
```

Opções para Defragmentação

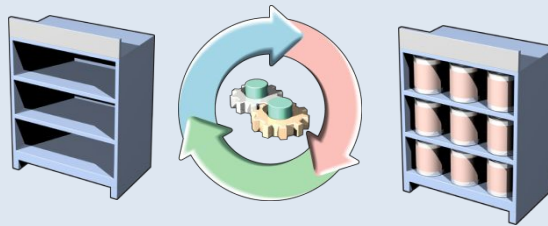
Reorganize



- **Fragmentação entre 10% e 30%**

```
ALTER INDEX AK_Product_Name ON  
Production.Product REORGANIZE
```

Rebuild



- **Fragmentação superior a 30%**

```
ALTER INDEX AK_Product_Name ON  
Production.Product REBUILD
```

Métodos para obter informações de índices

SQL Server Management Studio:

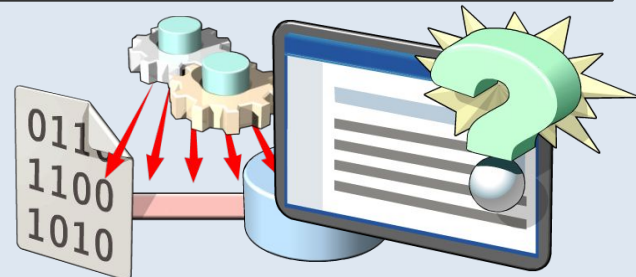
- Object Explorer
- Index Properties window
- Reports

System stored procedures:

- sp_help
- sp_helpindex

Catalog Views

System functions





Obrigado !

Gustavo Maia
Gustavo.Maia@FaculdadeImpacta.com.br