SQL Server Expert

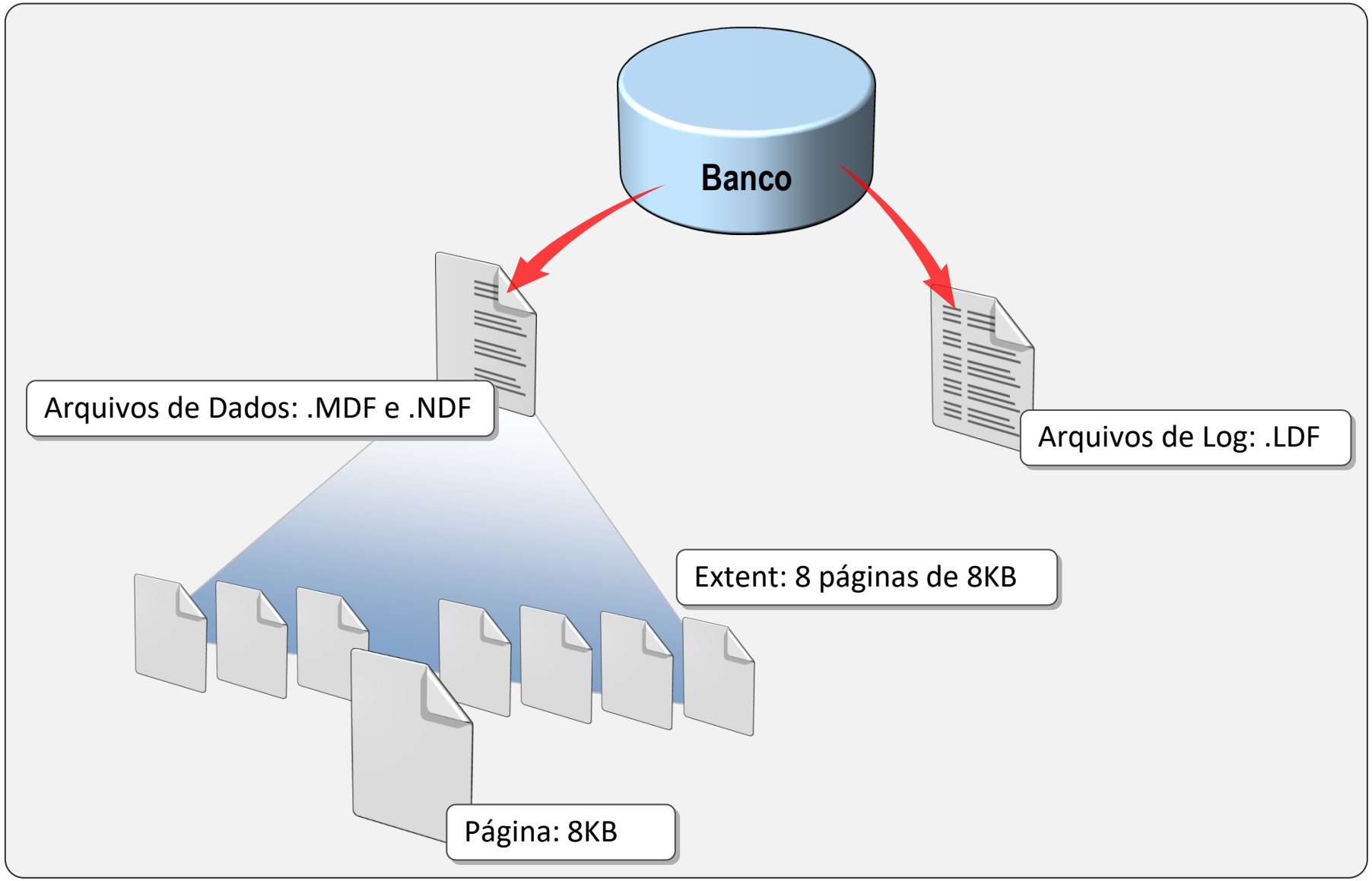
Prof.Landry

LIVE #004

Tuning de Consultas no Microsoft SQL Server

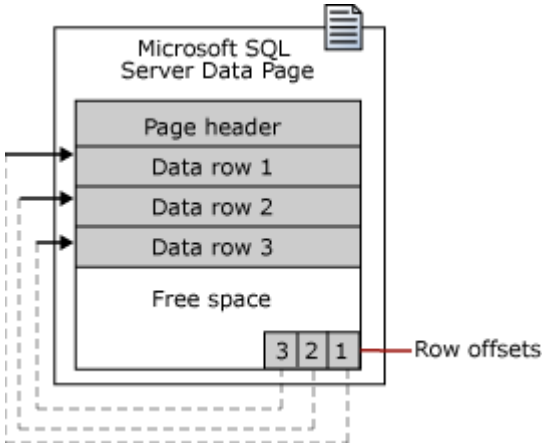
Live vai iniciar em instantes...

- **Agenda**
  1. Porque aprender Query Tuning?
  2. Arquitetura Tabelas e Índices
  3. Otimizador de Consultas
  4. Estatísticas de Banco de Dados
  5. Estratégias de Indexação
  6. Escrevendo Consultas Eficientes


Banco
Arquivos de Dados: .MDF e .NDF
Arquivos de Log: .LDF
Extent: 8 páginas de 8KB
Página: 8KB

- **Arquitetura dos Arquivos de Dados**
  - ✓ Arquivos de dados (.MDF e .NDF) possuem a mesma estrutura interna
  - ✓ Subdivididos em páginas de 8KB agrupadas em Extents
  - ✓ Cada Extent é compostas por 8 páginas de 8KB contínuas , seu endereço é igual o endereço da sua primeira Página
  - ✓ Cada Página de 8KB possui um endereço, composto por 2 números
  - ✓ Cada Linha possui um endereço composto por 3 números, por exemplo 1:3:9
    - **1:** Identifica o arquivo de dados na ordem de criação, 1 é MDF.
    - **3:** Identifica a página, numeração começa com zero, isto é, página 3 é na verdade a quarta página do arquivo.
    - **9:** Identifica a linha dentro da página, numeração começa com zero, isto é, linha 9 é a décima linha.

- **Arquitetura das páginas**
  - ✓ **Data Pages**
    - Contém linhas de uma única tabela
    - Também chamada de In Row Pages



  - ✓ **Image/Text Pages**
    - Armazena até 2GB tipos de dados TEXT, NTEXT, IMAGE
    - Contém também o excedente aos 8k dos tipos de dados VARCHAR, NVARCHAR, VARBINARY, XML, SQL_VARIANT
  - ✓ **Index Pages**
    - Contém estrutura de índice
  - ✓ **Outros tipos de páginas**
    - Index Allocation Map (IAM), Global Allocation Map (GAM), Secondary Global Allocation Map (SGAM), Page Free Space(PFS), Bulk Changed Map (BCM), Differential Changed Map (DCM)

- **Tamanho máximo da linha**
  - ✓ **SQL 6.5:** página 2Kb linha 1962bytes
  - ✓ **SQL 7.0:** página 8Kb linha 8060bytes - erro se tentar criar tabela maior que 8Kb, mesmo com coluna de tamanho variável.
  - ✓ **SQL 2000:** página 8Kb linha 8060bytes - com coluna de tamanho variável deixa criar retornando warning, INSERTE e UPDATE falha se ultrapassar o limite de 8Kb.
  - ✓ **SQL 2005 ou sup.:** página 8Kb linha xxbytes - pode criar tabela com linha maior que 8060 bytes, porém o total das colunas de tamanho fixo ainda não podem ultrapassar o limite dos 8060 bytes.
  - ✓ Estrutura das linhas sys.system_internals_allocation_units

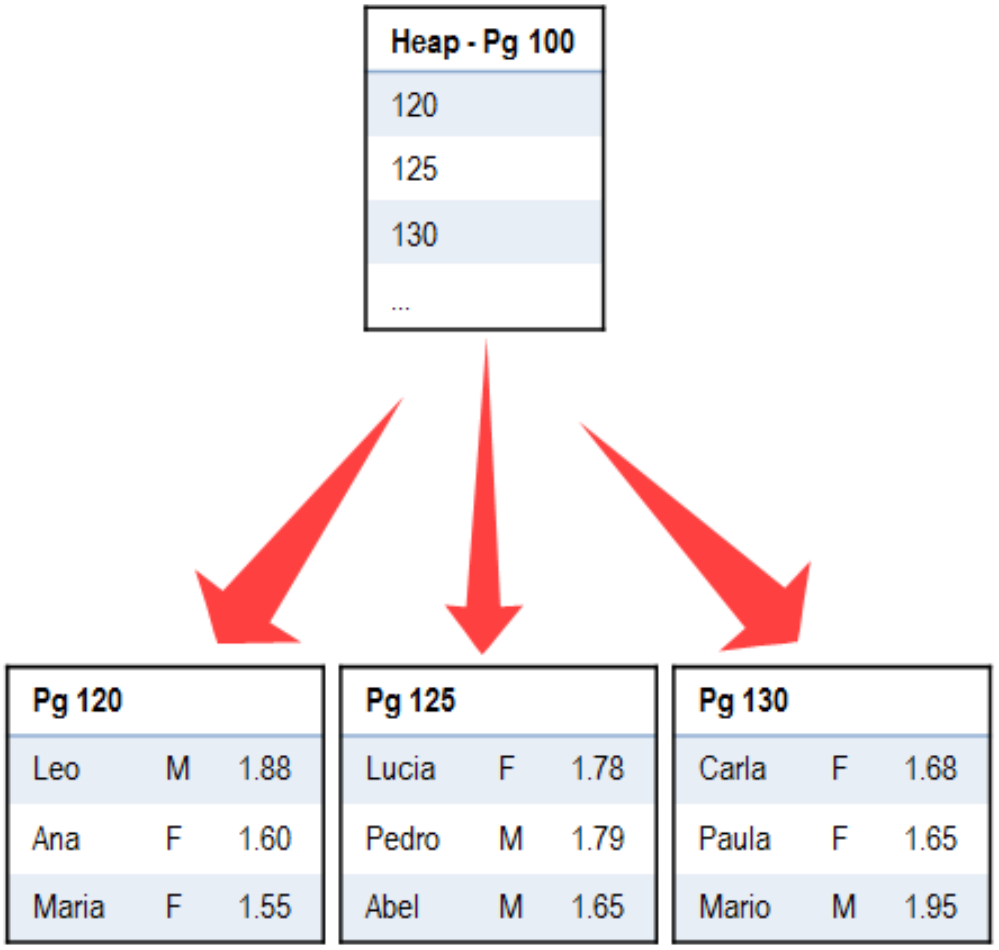
Results | Messages

	Nome	pnum	hobt_id	rows	au_id	type	TypeOfPages	pages
1	Menor8k	1	288230381520093184	0	1513209480866430976	1	InRow	0
2	Maior8k	1	360287975598981120	0	1585267074945318912	1	InRow	0
3	Maior8k	1	360287975598981120	0	1657324669024206848	3	OutRow	0

- **Hands On:** Tamanho máximo da linha

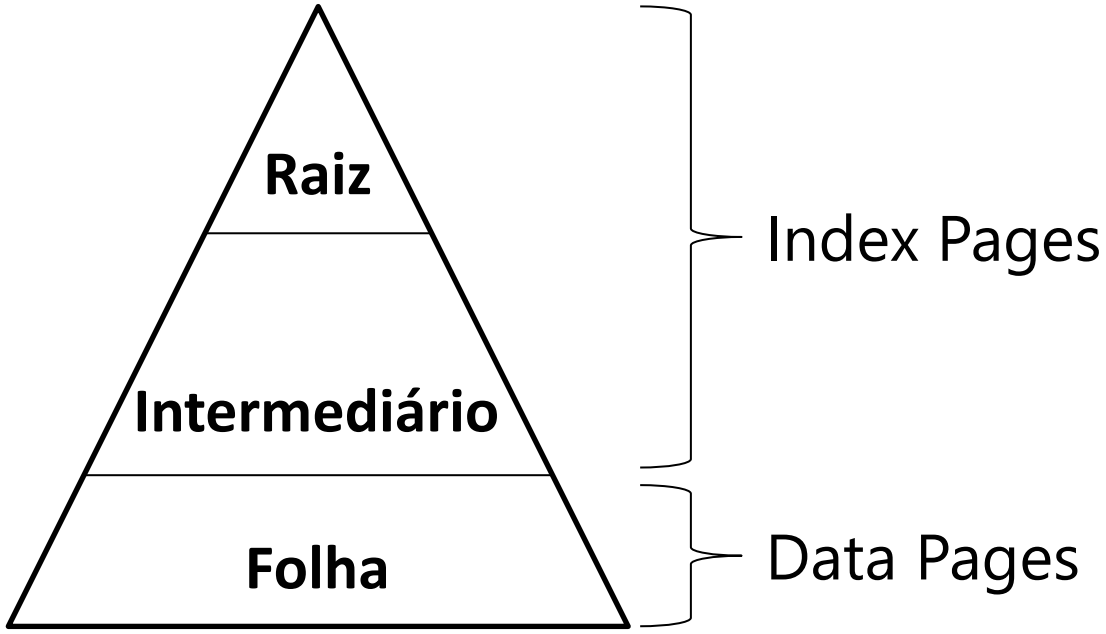
- **Tipos de Índices**
  - ✓ Btree Clustered
  - ✓ Btree Nonclustered
  - ✓ Columnstore Clustered
  - ✓ Columnstore Nonclustered
  - ✓ Columnstore Nonclustered Hash (In-Memory OLTP)
  - ✓ Fulltext Index

- **Heap Table**



- ✓ Estrutura de Tabelas sem Índice Clustered
- ✓ Linhas não ficam armazenadas em ordem
- ✓ Melhor desempenho no INSERT
- ✓ Não utilizar quando ocorre muito UPDATE

- **Índice Clustered**
  - ✓ Um por Tabela
  - ✓ Lock de Tabela do tipo Exclusivo (X) durante a criação
  - ✓ Necessita de 120% do espaço que a tabela ocupa para criar um Índice Clustered
  - ✓ Nível folha composto pela Tabela ordenada pela chave



- **Índice Clustered**

Chave	Pagina
Abel	Pg 60
Ian	Pg 70

Pg 50

Chave	Pagina
Abel	Pg 120
Carla	Pg 130
	Pg 70 »

Pg 60

Chave	Pagina
Ian	Pg 140
Mario	Pg 150
« Pg 60	

Pg 70

Chave	Colunas	
Abel	M	65
Ana	F	16
Beto	M	15
Bia	F	56
	Pg 130 »	

Pg 120

Chave	Colunas	
Carla	F	48
Carlos	M	67
Daniel	M	23
Fabio	M	76
« Pg120	Pg 140 »	

Pg 130

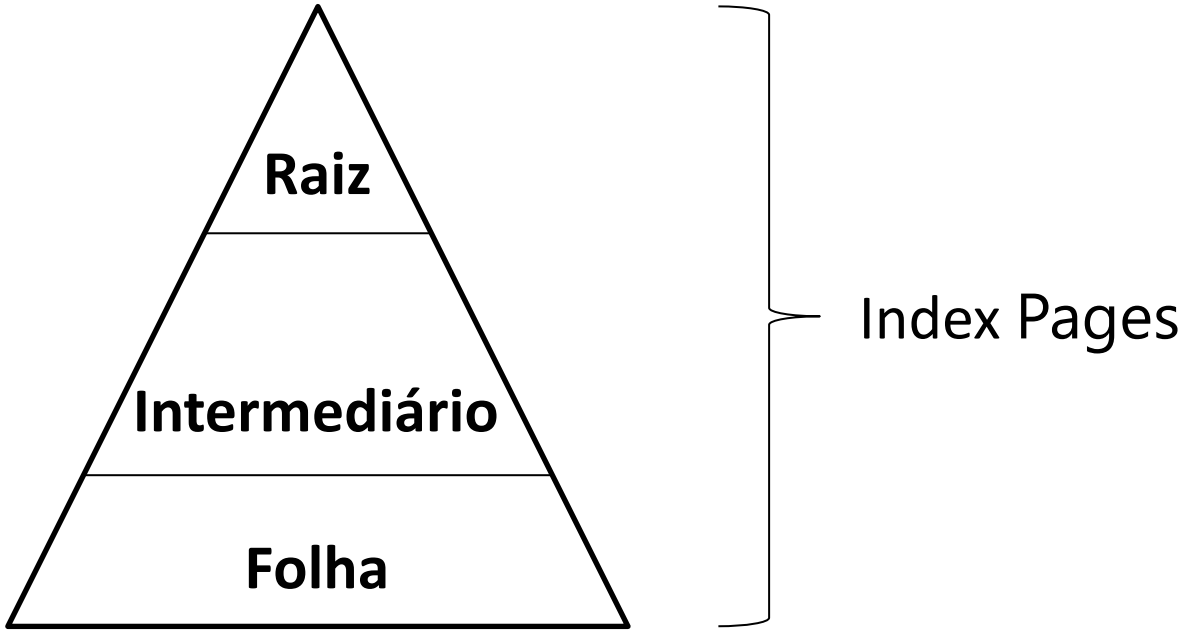
Chave	Colunas	
Ian	M	62
Leo	M	80
Lucia	F	38
Maria	F	55
« Pg130	Pg 150 »	

Pg 140

Chave	Colunas	
Mario	M	19
Paula	F	65
Paulo	M	28
Pedro	M	79
« Pg140		

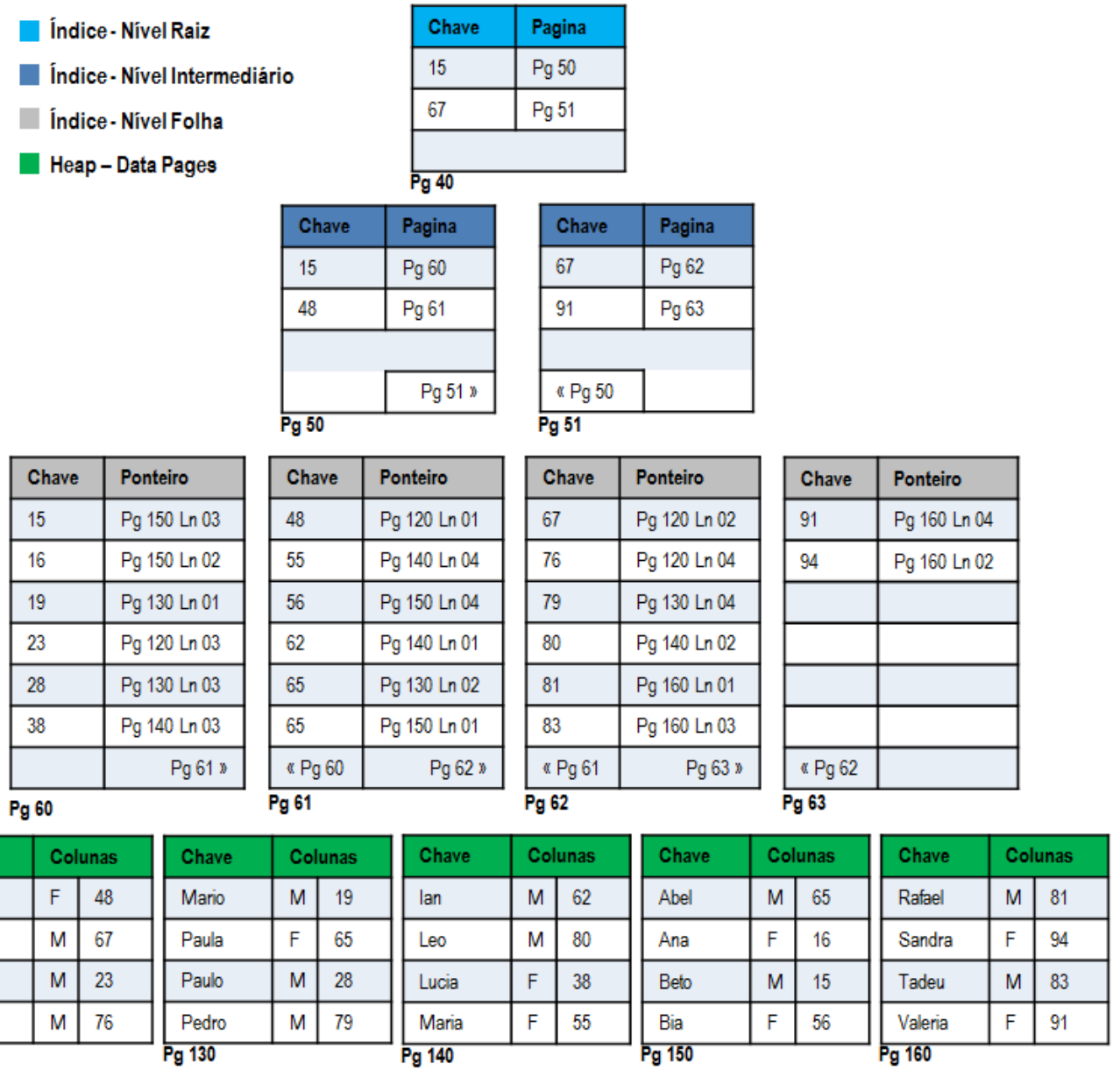
Pg 150

- **Índice Nonclustered**
  - ✓ Estrutura totalmente a parte da tabela (Index Pages)
  - ✓ Lock de Tabela do tipo Shared (S) durante a criação
  - ✓ Até SQL Server 2005 máximo de 249 por tabela, a partir do SQL Server 2008 máximo de 999
  - ✓ Nível folha contém a chave do índice ordenada e um ponteiro que localiza o Data Page



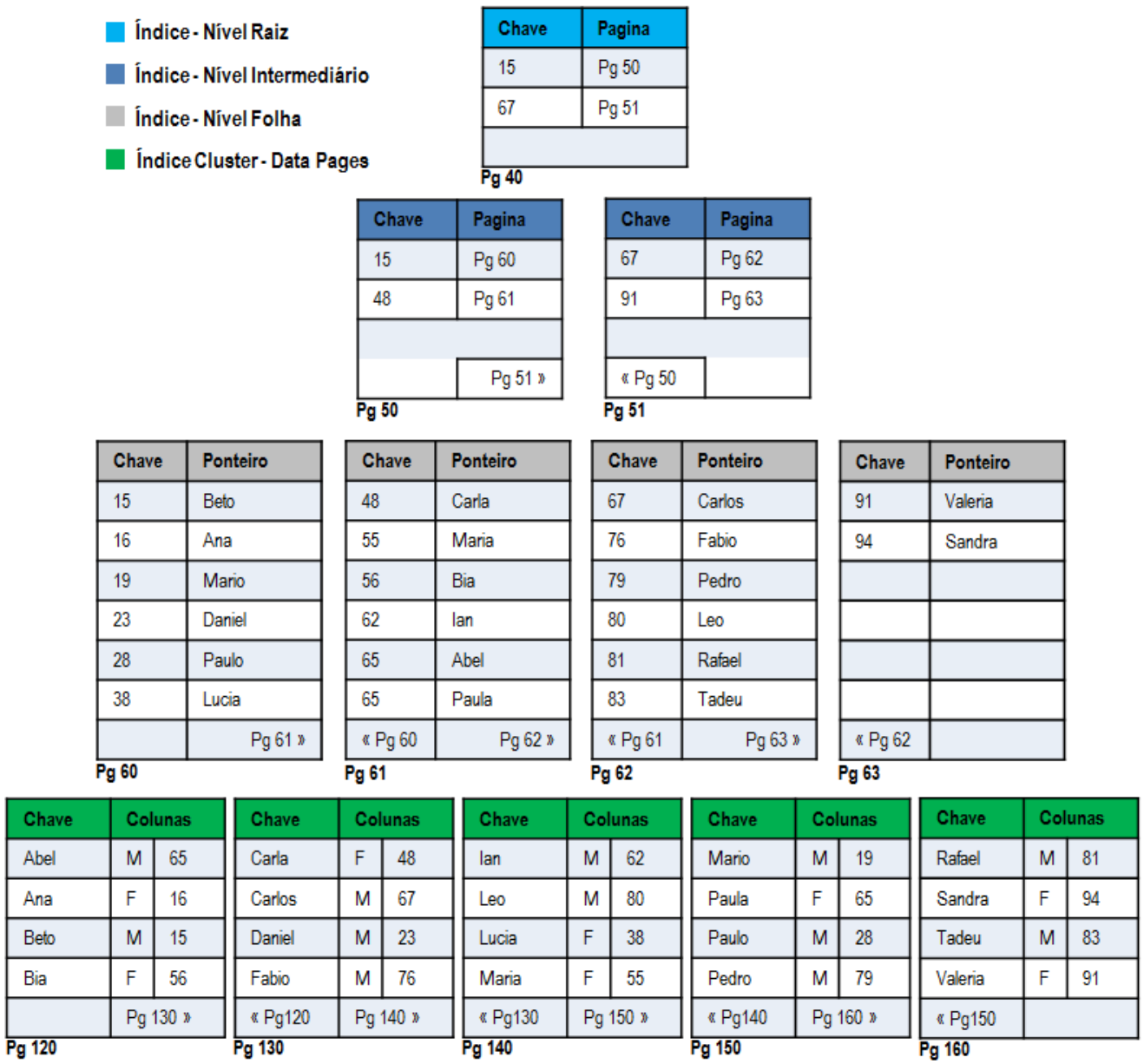
- **Índice Nonclustered**

  ✓ **Sem** Índice Clustered



- **Índice Nonclustered**

  ✓ **Com** Índice Clustered



- **O que é um Otimizador de Consultas?** 
  - ✓Otimizador com Base em Regras (Rule Based Optimizer - RBO)
  - ✓Otimizador com base em custo (Cost Based Optimizer - CBO)
    - Seleciona plano com menor consumo de recurso (I/O, CPU e memória), dentre aqueles analisados
    - Utiliza estatísticas de banco de dados para análise
    - Análise pode ser diferente entre Edições do SQL Server
    - Nem sempre é possível optar pelo “melhor plano de execução”

- **Fases de execução de consultas**

1. **Parse (análise)** – Verifica a sintaxe, construindo uma representação em árvore de operadores chama "Parse Tree".
2. **Resolve (resolver)** – Identifica se os objetos referenciados na consulta realmente existem. O produto final desta fase se chama "Algebrized Tree".
3. **Optimize (otimizar)** – Seleciona o plano de execução.
4. **Compile (compilar)** – Compila o plano de execução selecionado pela faze anterior, depois armazena na memória no Plan Cache.
5. **Execute (execução)** – Execução da consulta.

- **O que é Estatística de Banco de Dados?** 

  ✓Mantém informações referente a distribuição de valores de uma chave

  ✓SQL Server cria estatística automaticamente quando criamos índice

  ✓Controlada por duas propriedades de banco de dados:

    - Auto Create Statistics:
    - Auto Update Statistics

  ✓SQL Server monitora atualização na tabela e atualiza as estatísticas

  ✓Atualização das estatísticas pode ser feita por **Sampling** ou **Full Scan**

- **Conceitos Importantes**

✓**Seletividade**

- Quantidade de linhas retornada por uma consulta
- Quanto **maior** a quantidade de linhas, **menor** é a seletividade
- Consultas de alta seletividade tem maior probabilidade de utilizar índices

✓**Densidade**

- Indicador que reflete a repetição de valores
- Quanto mais valor repetido uma chave tem, maior é a densidade e menor a probabilidade de utilizar índices
- Costuma ser inversamente proporcional Seletividade e Densidade
  - ✓ **Maior** Densidade **Menor** Seletividade
  - ✓ **Menor** Densidade **Maior** Seletividade

- **Hands On:** Estatística de Banco de Dados

- **Covering Index**
  - ✓ Covering Index é a melhor estratégia de indexação!
  - ✓ Índice atende por completo a consulta
  - ✓ Todas as colunas referenciadas na consulta fazem parte da estrutura do índice
  - ✓ Não ocorre acesso a **Data Page** (Tabela)
  - ✓ Não ocorre **Bookmark Lookup**

- **Estratégia de Indexação para "AND"**
  - ✓ O **AND** é o "e" lógico
  - ✓ Todas as expressões devem retornar verdadeiro para fazer parte do resultado
  - ✓ **Exemplo:**

    ```
    WHERE Bairro = 'Ipanema' AND EstadoCivil = 'Casado'
    ```

  - ✓ **Resolução:**

    1. Filtro do Bairro descarta todas as linhas diferentes de 'Ipanema'
    2. Para as linhas restantes aplica-se o filtro EstadoCivil = 'Casado'.

  - ✓ **Indicado definir o índice na cláusula mais seletiva**

- **Estratégia de Indexação para "OR"**
  - ✓ O **OR** é o "ou" lógico
  - ✓ Sendo verdadeiro em uma condição a linha fará parte do resultado
  - ✓ **Exemplo:**

    ```
    WHERE Bairro = 'Ipanema' OR EstadoCivil = 'Casado'
    ```

  - ✓ **Resolução:**

    1. Verifica todas as linhas e separa para o resultado as linhas com Bairro igual a 'Ipanema'
    2. Verifica todas as linhas e separa para o resultado as linhas com EstadoCivil = 'Casado'

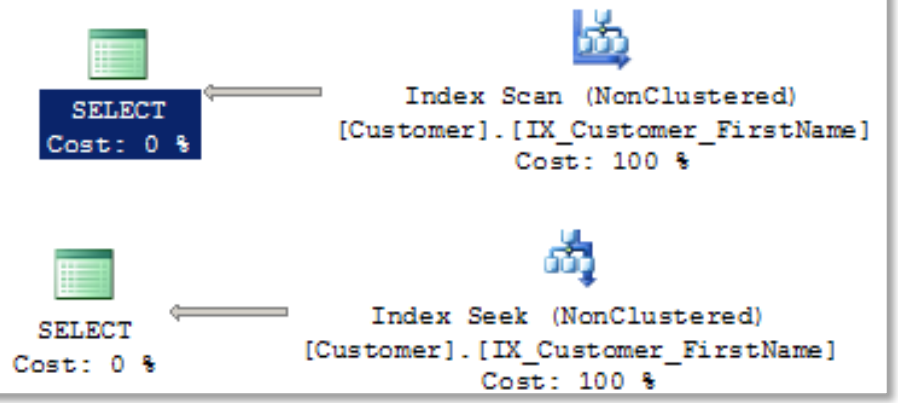
  - ✓ **Necessita de um índice para cada cláusula, se não faz Scan**
  - ✓ **Solução paliativa, criar um único Covering Index**

- **Sargable X Non-Sargable**
  - ✓Evitar condições Non-Sargable no WHERE
    - **Operadores Sargable:** =,>,<,>=,<=,BETWEEN,LIKE (fixo no início)
    - **Operadores Non-Sargable:** <>,NOT IN, NOT LIKE, LIKE (variável no inicio)
  - ✓Não utilizar
    - WHERE Função(Coluna) = Expressão

```
CREATE INDEX IX_Customer_FirstName ON tempdb.dbo.Customer (FirstName)
INCLUDE (CustomerID, LastName)

-- Consulta 1
SELECT CustomerID, FirstName, LastName
FROM tempdb.dbo.Customer WHERE left(FirstName,1) = 'G'
-- Index Scan: Table 'Customer'. Scan count 1, logical reads 114

-- Consulta 2
SELECT CustomerID, FirstName, LastName
FROM tempdb.dbo.Customer WHERE FirstName like 'G%'
-- Index Seek: Table 'Customer'. Scan count 1, logical reads 7
```



Live #005

Segurança do Microsoft SQL Server

 **E-mail:** proflandry.sqlexpert@gmail.com

 **Facebook:** @SQLServer.Expert.Landry

 **Youtube:** @prof-landrySQLServerExpert

 **Instagram:** @sqlserver.expert