# SISTEMAS DE BANCO DE DADOS 2

## AULA 11

## Processamento e Otimização de Consultas

Vandor Roberto Vilardi Rissoli

# APRESENTAÇÃO

- Processamento de Consultas
- Otimização de Consultas
- Representações em Escala
- Referências

# PROCESSAMENTO DE CONSULTAS

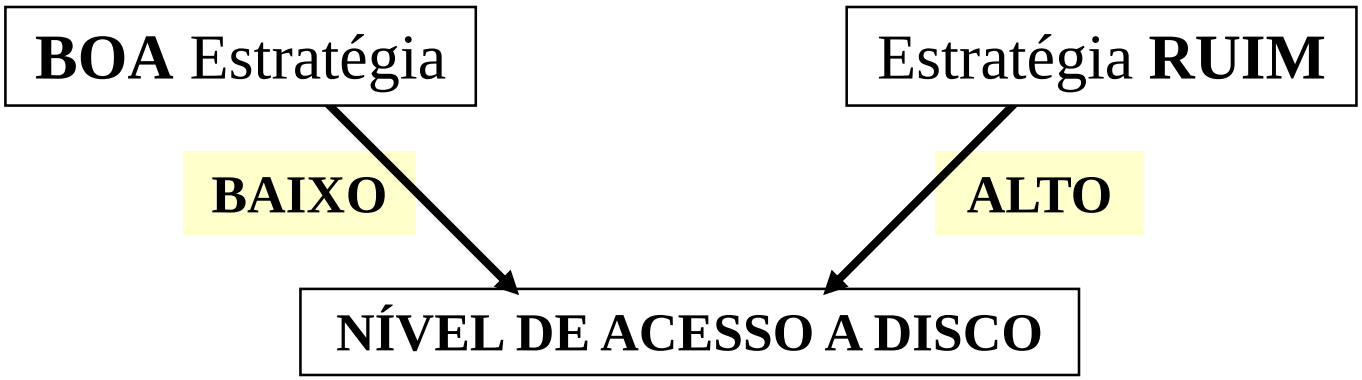
O processamento de consultas são as atividades envolvidas em extrair dados de um BD. Estas atividades incluem:

- Tradução da linguagem de alto nível do BD para expressões que podem ser implementadas no nível físico do sistema de arquivo;
- Otimização;
- Avaliação das consultas.

# PROCESSAMENTO DE CONSULTAS

→ O "CUSTO" de uma consulta é determinado, principalmente, pelo acesso a disco (muito lento em relação ao acesso a memória principal).

→ Normalmente, há muitas estratégias possíveis para processar uma consulta, especialmente se ela for complexa.



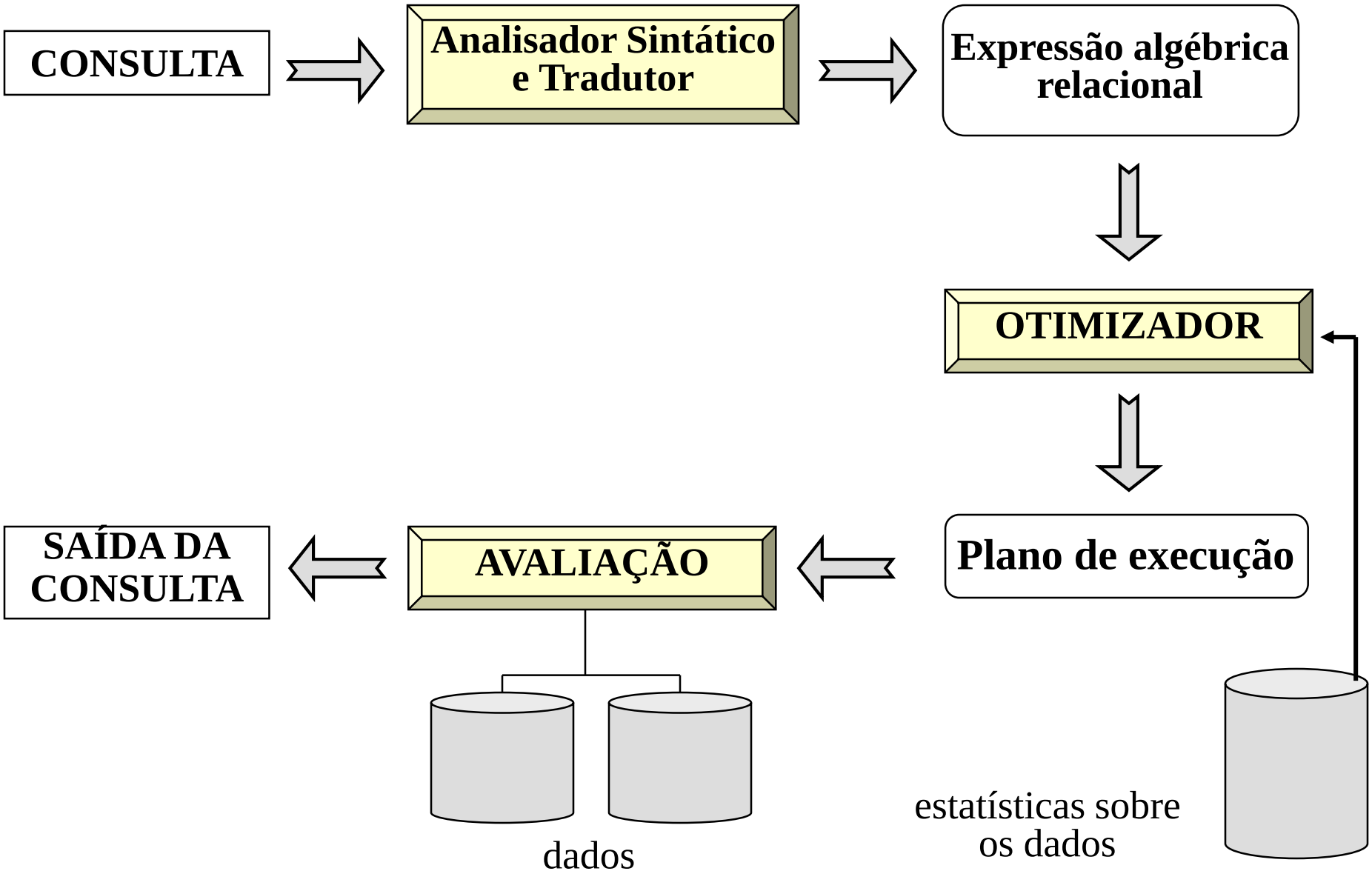
# PROCESSAMENTO DE CONSULTAS

→ Vale a pena o sistema "GASTAR" uma pequena quantia de tempo e esforço na seleção de uma **BOA** estratégia para processar uma consulta.

→ Os passos básicos no processamento de uma consulta são:

- Análise sintática e tradução;
- Otimização;
- Avaliação.

# PROCESSAMENTO DE CONSULTAS



# PROCESSAMENTO DE CONSULTAS

→ As linguagens de consulta relacionais são declarativas ou algébricas, o que permite aos usuários especificarem o que uma consulta deve gerar, sem informar como o sistema deve operar para fornecer o resultado desta consulta;

→ É relativamente fácil, baseado na especificação da consulta, um otimizador gerar uma variedade de planos equivalentes de execução para cada consulta;

→ Dentre os planos equivalentes gerados é escolhido o menos oneroso ao sistema.

# PROCESSAMENTO DE CONSULTAS

Exemplo:

*SELECT saldo*

*FROM conta*

*WHERE saldo < 2000*

↓

pode ser traduzida nas seguintes
expressões algébricas relacionais

↓

$\sigma$ [saldo < 2000] ($\pi$ saldo (Conta))

$\pi$ saldo ($\sigma$ [saldo < 2000] (Conta))

↓

usando vários algoritmos diferentes
para cada operação algébrica

# PROCESSAMENTO DE CONSULTAS

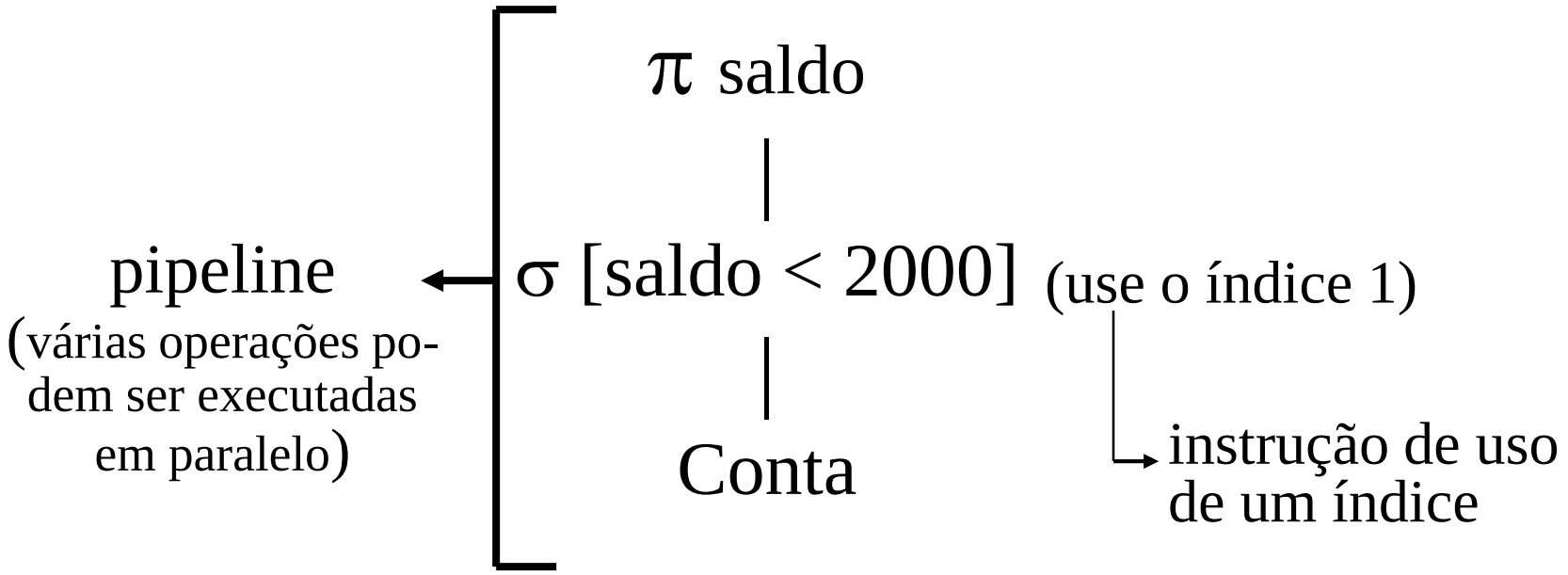
Exemplo para realização da consulta:

- Procurar em todas as tuplas de contas por saldos menores que dois mil (saldo < 2000);
- Na existência de um índice de árvore $B^+$ sobre o atributo *saldo*, ele pode ser usado ao invés da análise de todas as tuplas;

**EXPRESSÕES ALGÉBRICAS** + **ANOTAÇÕES (como avaliar cada expressão)** → AVALIAÇÃO PRIMITIVA

# OTIMIZAÇÃO DE CONSULTAS

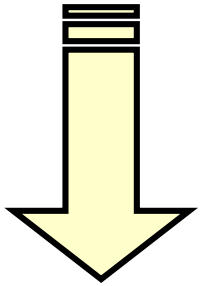
→ Uma sequência de operações primitivas, que podem ser usadas para avaliar uma consulta, consiste em um *plano de execução de consulta* ou *plano de avaliação de consulta*.



# OTIMIZAÇÃO DE CONSULTAS

**PLANOS DE AVALIAÇÃO** $\neq$ **CUSTOS (desempenho)**

→ Encontrar o plano mais eficiente é responsabilidade do sistema.



**OTIMIZAÇÃO** ⟶ **CONSULTA MAIS EFICIENTE**

# OTIMIZAÇÃO DE CONSULTAS

- Encontrar as expressões equivalentes a álgebra relacional desejada, onde principalmente diminua o acesso a disco;
- Selecionar uma estratégia detalhada para o processamento da consulta (escolha do algoritmo, índice, etc.);
- Estimar "custos" dos planos de avaliação (otimizadores fazem uso de informações estatísticas sobre as relações – tamanho das relações, profundidade dos índices – para realizar uma melhor estimativa de custo de um plano);
- Escolhido o plano de avaliação, a consulta é executada de acordo com o mesmo (plano) e o resultado da consulta é produzido.

# OTIMIZAÇÃO DE CONSULTAS

**ÁLGEBRA RELACIONAL** $\neq$ **ÁRVORE SINTÁTICA** → **REPRESENTAÇÃO GENÉRICA**

↓

**BASE PARA O PROCESSAMENTO DE CONSULTAS**

# OTIMIZAÇÃO DE CONSULTAS

## ESTIMATIVA DE CUSTOS

Os otimizadores de consulta usam dados estatísticos, armazenados no catálogo do BD, para verificar os custos de um plano. Estes dados sobre as relações incluem:

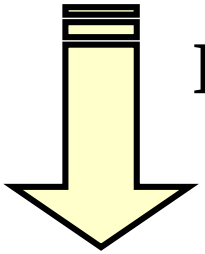
- Número de tuplas;
- Número de blocos que contém as tuplas;
- Tamanho em bytes das tuplas;
- Número de valores distintos em uma relação para um atributo;
- Cardinalidade da seleção do atributo da relação.

→ Os dados aqui mencionados são simplificados, pois otimizadores reais mantém mais dados estatísticos, permitindo uma melhor estimativa de custos dos planos de avaliação.

# OTIMIZAÇÃO DE CONSULTAS

- O catálogo (ou metadados) armazena dados sobre as relações, além das informações sobre os índices existentes;
- Os dados estatísticos são usados a fim de estimar o tamanho do resultado e do custo para várias operações e algoritmos;

**ESTATÍSTICAS PRECISAS**



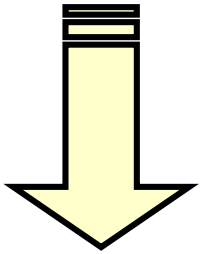
**MODIFICAÇÕES**
**(**atualizações no BD**)**

**ATUALIZAR ESTATÍSTICAS**

→ GERA UM *OVERHEAD* SIGNIFICATIVO

# OTIMIZAÇÃO DE CONSULTAS

As atualizações das estatísticas ocorrem normalmente na carga do sistema



NÃO SÃO COMPLETAMENTE PRECISAS

(precisam ser avaliadas - se estas estimativas oferecem estatísticas suficientemente precisas para fornecerem os custos coerentes relativos aos diferentes planos)

# OTIMIZAÇÃO DE CONSULTAS

## MEDIDAS DE CUSTO DE UMA CONSULTA

O custo de avaliação para uma consulta pode ser medido por meio de vários recursos diferentes, incluindo acesso a disco, tempo de CPU, custo de comunicação, entre outros.

**PARA GRANDES BD's** → **Número de transferência de blocos do disco são os mais relevantes devido a lentidão na sua velocidade em relação a memória.**

# OTIMIZAÇÃO DE CONSULTAS

## OPERAÇÃO DE SELEÇÃO

- "Varrer" (ou vasculhar) arquivos
  - Operação de mais baixo nível para ter acesso aos dados;
- Observe os dois tipos de "*varredura*":
  - Busca Linear: cada bloco do arquivo é varrido e todos os registros são testados para verificar se satisfazem a condição de seleção.
  - Busca binária: o arquivo deve estar ordenado em um atributo e a condição de seleção é uma comparação de igualdade no atributo.

# OTIMIZAÇÃO DE CONSULTAS

→ Apesar da ineficiência do algoritmo da busca linear em muitos casos, ele sempre pode ser aplicado a qualquer arquivo, indiferentemente da ordem do arquivo ou da disponibilidade de índice.

Exemplo:

Suponha os seguintes dados estatísticos sobre a relação CONTA, sendo ilustrada a seguir as suas estimativas:

- $f_{conta} = 20$ (20 tuplas de conta cabem em um único bloco)
- $V_{(agência,conta)} = 50$ (50 agências diferentes na relação conta)
- $V_{(saldo,conta)} = 500$ (500 saldos diferentes na relação conta)
- $n_{conta} = 10.000$ (10.000 tuplas na relação conta)

# OTIMIZAÇÃO DE CONSULTAS

Suponha agora a seleção ($\sigma$):

$\sigma$ [agencia = “São Paulo”] (conta)

$\rightarrow$ relação = 10.000 tuplas, sendo 20 tuplas por bloco

tem-se então:

$\rightarrow$ 500 blocos para acessar toda a relação (varredura simples (ou linear) consumirá 500 acessos a bloco)

Caso conta esteja ordenada por agência e são 50 agências diferentes, espera-se que 200 (10000/50) tuplas da relação conta sejam da agência São Paulo (distribuição uniforme).

Sendo assim, estas tuplas caberiam em 10 blocos (200/20).

# OTIMIZAÇÃO DE CONSULTAS

Baseado na estimativa de uso da busca binária tem-se:

$\rightarrow \log_2(500) = 9$ acessos a bloco (tendo assim o custo total de $9 + 10 - 1 = 18$)

- $\log_2(500)$: equação válida se todas as tuplas estiverem em um mesmo arquivo físico
- 500: número de blocos
- 9: custo para encontrar a 1ª tupla por meio da busca binária nos blocos
- 10: Estimativa do número total de registros que satisfarão a seleção

estimativa para os blocos do arquivo a ser varrido na busca binária

# OTIMIZAÇÃO DE CONSULTAS

A escolha de uma estratégia para avaliar uma operação depende do:

1- tamanho de cada relação;

2- distribuição de valores dentro das colunas.

Procurando conseguir uma estratégia baseada em informação confiável, os sistemas de BD podem armazenar estatísticas para cada relação.

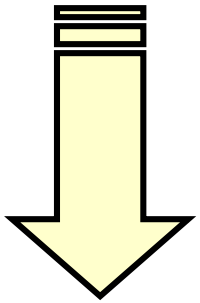
# OTIMIZAÇÃO DE CONSULTAS

Essas ESTATÍSTICAS:

- Permitem estimar os tamanhos dos resultados de várias operações;
- Possibilitam estimar os custos para executar estas operações;
- Especialmente útil quando vários índices estão disponíveis para ajudar no processamento de uma consulta;
- A presença destas estruturas (índices) tem influência significativa na escolha de uma estratégia de processamento de consulta.

# OTIMIZAÇÃO DE CONSULTAS

A estrutura de índice também é chamada de *path*



caminho de acesso aos dados

→ possibilidade de *overhead* aos blocos que contém o índice

# OTIMIZAÇÃO DE CONSULTAS

**PODE-SE PROCESSAR CONSULTAS QUE ENVOLVAM**

**A.** Seleções simples por meio da execução de:

- Busca (ou varredura) linear;
- Busca binária;
- Uso de índice;

**B.Seleções complexas (**computando uniões e interseções de seleções simples**);**

**C. Ordenar relações maiores** que a memória usando o algoritmo de *merge-sort* externo**;**

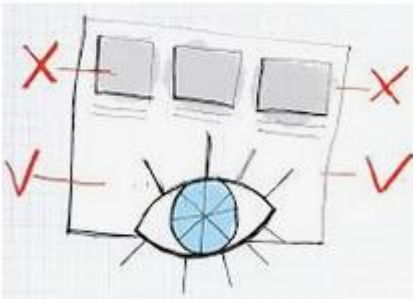
**D.Consultas com junção natural podem ser processadas de vários modos**, de acordo com a disponibilidade de índices e da forma de armazenamento físico nas relações.

# OTIMIZAÇÃO DE CONSULTAS

Aplicação de algumas técnicas:

- Se o resultado da junção é quase tão grande quanto o produto cartesiano das duas relações, uma estimativa de junção de laço aninhado de bloco pode ser vantajosa;

- Se índices estão disponíveis, a junção de laço aninhado indexada pode ser usada;



# OTIMIZAÇÃO DE CONSULTAS

- Se as relações estão classificadas, uma merge-junção pode ser desejável
  - Pode ser vantajoso ordenar uma relação antes de calcular a junção para uso da merge-junção;
  - Também pode ser vantajoso calcular um índice temporário com o propósito exclusivo de permitir a utilização de uma estratégia de junção mais eficiente.

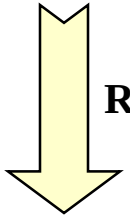
- O algoritmo de hash-junção particiona as relações em vários pedaços, de forma que cada pedaço caiba na memória (particionamento feito por uma função de hash sobre os atributos da junção, para que se possa fazer independentemente a junção de pares correspondentes de partições).

# OTIMIZAÇÃO DE CONSULTAS

## MERGE-SORT

1. Lê o primeiro bloco para memória principal;
2. Escolhe a primeira tupla do primeiro bloco e escreve em um arquivo temporário;
3. Apaga a tupla lida do primeiro bloco e vai para a próxima tupla (ou novo bloco);

→ Repetir este processo até todos os blocos estarem vazios.



**RESULTADO**

**RELAÇÃO CLASSIFICADA**

# OTIMIZAÇÃO DE CONSULTAS

## CLASSIFICAÇÃO

A classificação dos dados tem papel importante em Banco de Dados:

1. As consultas podem solicitar que seus resultados sejam apresentados ordenadamente;
2. Diversas das operações relacionais podem ser implementadas eficazmente, se as relações de entrada forem primeiramente classificadas;
3. Apaga a tupla lida do primeiro bloco e vai para a próxima tupla (ou novo bloco);

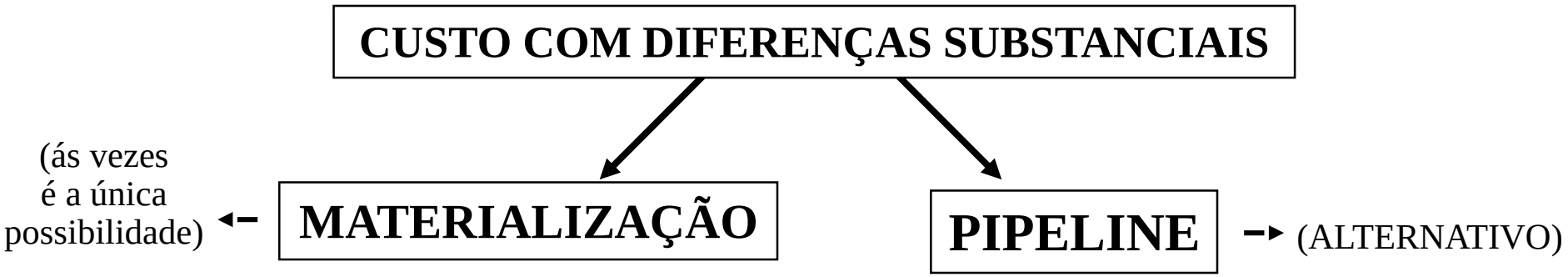
Classificação externa: classificação de relações que não cabem completamente na memória principal.

**ORDENAÇÃO POR ÍNDICE** $\neq$ **ORDENAÇÃO FÍSICA**

# OTIMIZAÇÃO DE CONSULTAS

## Como Avaliar Expressões com OPERAÇÕES MULTIPLAS

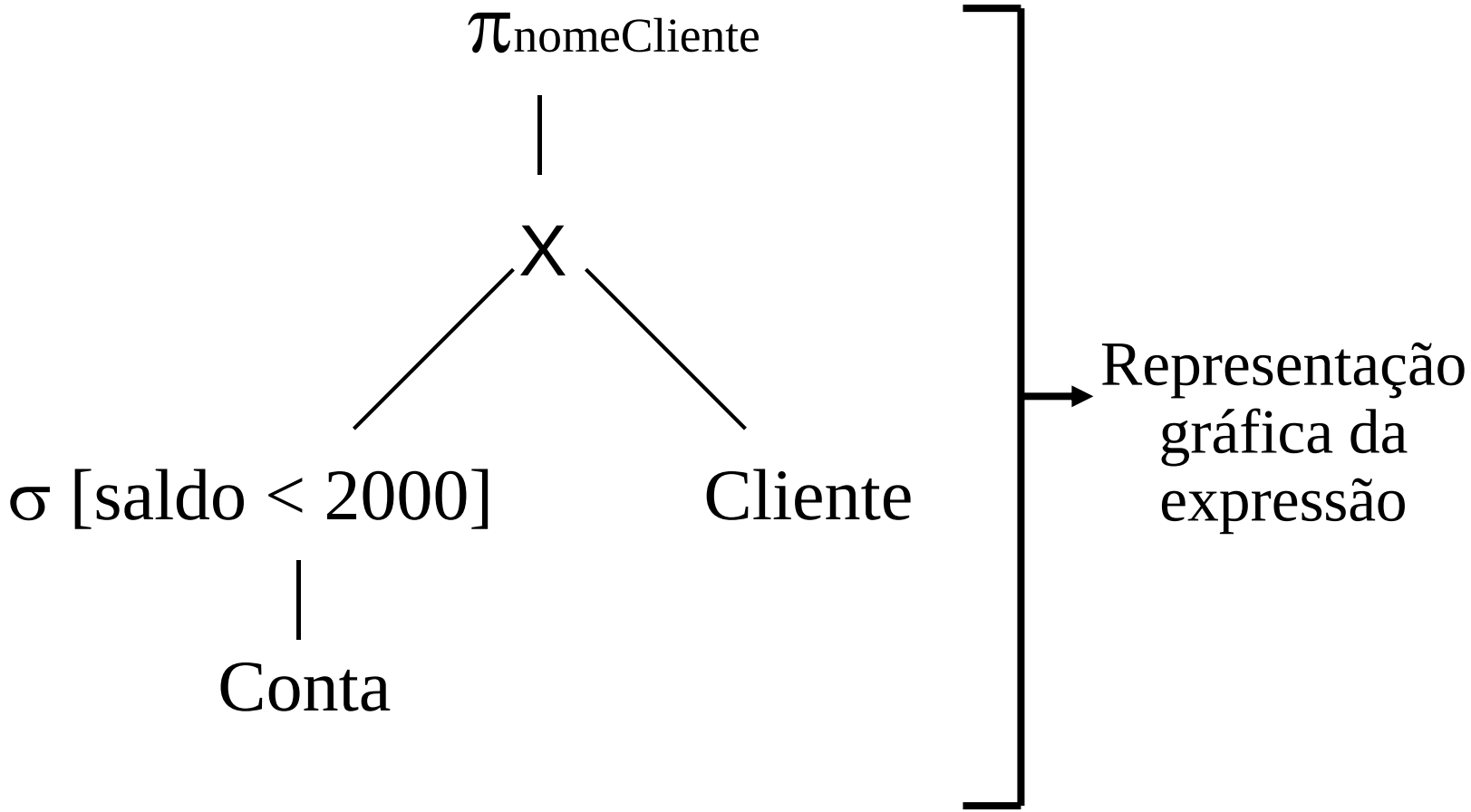
- Uma operação por vez, em uma ordem apropriada, gerando um resultado (materializado) para cada avaliação em uma relação temporária para ser usada na sequência;
- Avaliar várias operações simultaneamente em um *pipeline*, com os resultados de uma operação sendo passados para a próxima, sem a necessidade de relações temporárias;



# OTIMIZAÇÃO DE CONSULTAS

Exemplo:

$\pi_{nomeCliente}$ ($\sigma$ [saldo < 2000] (Conta) X Cliente)



# OTIMIZAÇÃO DE CONSULTAS

Observe no exemplo anterior a sequência de execução:

- Inicia no nível mais baixo da árvore;
- Resultados das operações mais baixas são armazenadas em relações temporárias que serão usadas na execução dos níveis mais altos;
- Cria-se uma nova relação temporária na execução de uma nova operação (nível acima);
- Executando a operação da raiz da árvore têm-se o resultado final da expressão desejada.

→ O custo da materialização deve considerar também o tempo da escrita no disco dos resultados intermediários (relações temporárias).

# OTIMIZAÇÃO DE CONSULTAS

Dobrando o *buffer* (usando dois *buffers*)

- um *buffer* prossegue à execução do algoritmo;
- o outro está escrevendo no disco;

Execução mais rápida devido a atividade em paralelo da CPU sobre as E/S

Os custos destas operações podem ser reduzidos pela combinação de várias operações relacionais em um PIPELINE de operações, em que os resultados de uma operação são passados diretamente para próxima operação, sem a necessidade de criação das relações temporárias.

# OTIMIZAÇÃO DE CONSULTAS

Exemplo:

junção

$\pi_{agencia,\ conta}$ (Agencia X Conta)

- ✓ A junção gera uma tupla
- ✓ Projeção da tupla gerada

→ Combinação direta das operações

→ Pipeline dirigido por demanda: as solicitações de tuplas são operações repetidas no topo do pipeline;

- maneira mais utilizada devido a facilidade de implementação;

**DEMANDA** = puxão (para cima a partir do topo) para uma árvore de operações

→ Pipeline dirigido pelo produtor: as operações não esperam solicitações para produzir tuplas, elas geram tuplas "ansiosamente" até encher seu buffer;

**PRODUTOR** = empurrão (de baixo para cima) para uma árvore de operações

- de maneira contínua

as tuplas vão sendo removidas, após serem usadas, enchendo o buffer novamente até a geração de todas as tuplas.

# OTIMIZAÇÃO DE CONSULTAS

## EXPRESSÕES EQUIVALENTES

**EXPRESSÃO** ⇒ **Diversas Formas** (mesmo resultado) → **Diferentes Custos de Avaliação**

**GERAM**

**EXPRESSÕES EQUIVALENTES OU ALTERNATIVAS**

# OTIMIZAÇÃO DE CONSULTAS

Exemplo:

Considere a consulta

- Encontre os nomes de todos os clientes que possuem conta na agência de São Paulo
- A expressão na álgebra relacional seria:

$$\pi_{\text{nomeCliente}} (\sigma_{[\text{agencia}=\text{"São Paulo"}]}(\text{Agencia } X(\text{Conta } X \text{ Cliente})))$$

→ Expressão com relação intermediária grande;

→ Interesse somente nas tuplas que pertencem a agência de São Paulo;

→ Necessidade de apenas um atributo da relação;

# OTIMIZAÇÃO DE CONSULTAS

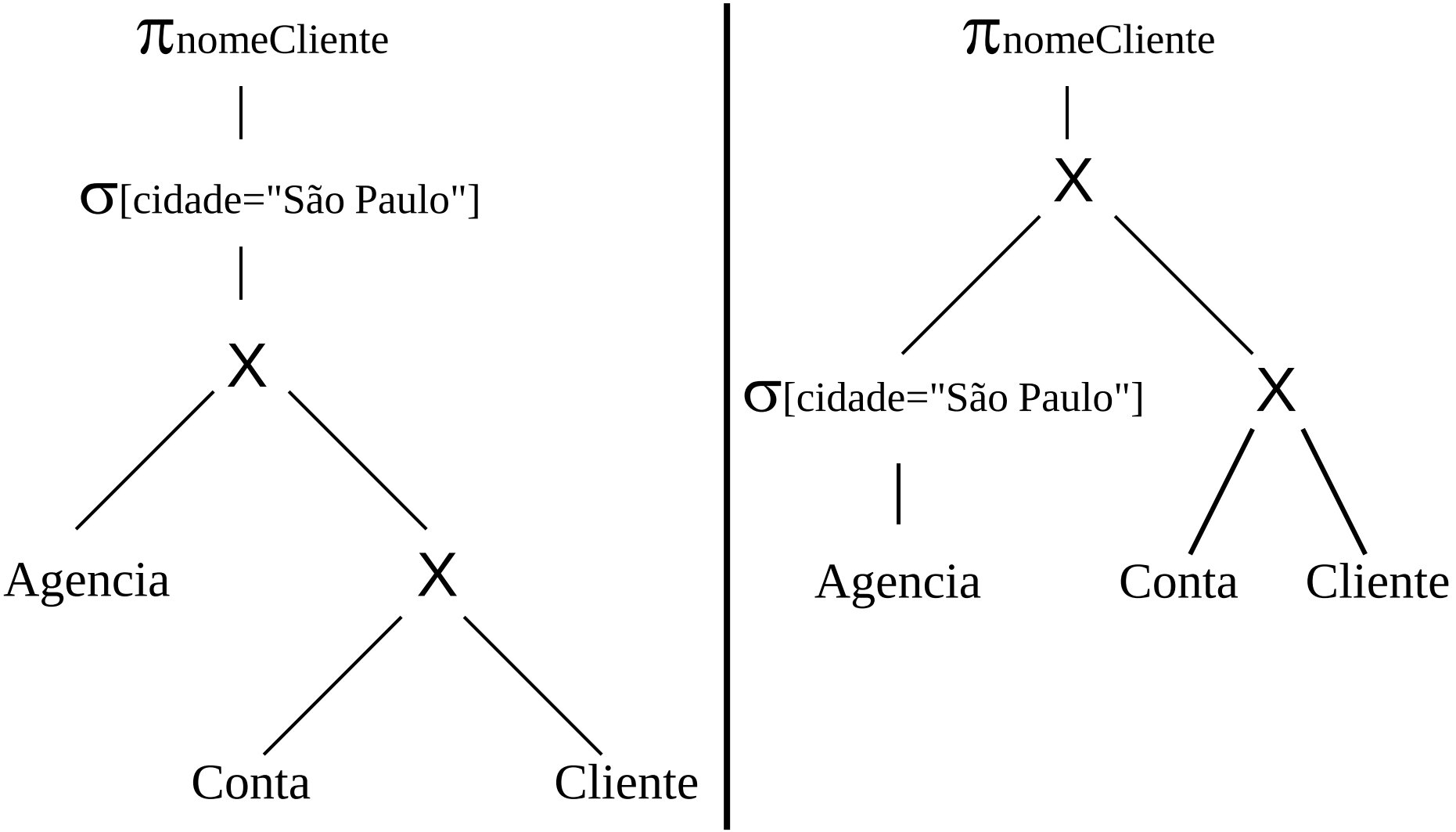
continuação para o mesmo exemplo...

O custo da expressão proposta poderia ser reduzido com uma geração de tuplas mais coerente para a relação intermediária.

Compare o resultado da expressão algébrica a seguir:

$$\pi_{\text{nomeCliente}}((\sigma_{[\text{agencia}=\text{"São Paulo"}]}(\text{Agencia}))X(\text{Conta } X \text{ Cliente}))$$

# OTIMIZAÇÃO DE CONSULTAS



# OTIMIZAÇÃO DE CONSULTAS

## FUNÇÃO DO OTIMIZADOR

- Propor um plano de avaliação da consulta que gere o mesmo resultado da expressão fornecida (planos alternativos);
- Encontrar uma maneira menos onerosa de gerar o resultado desejado (plano menos caro);

# OTIMIZAÇÃO DE CONSULTAS

## GERAÇÃO EM DOIS PASSOS

1. Geração das expressões que são logicamente equivalentes a expressão solicitada;

2. Escrever as expressões resultantes de maneiras alternativas para gerar planos de avaliação alternativos;

→ Passos intercalados no otimizador de consultas:

algumas expressões são geradas e escritas, então expressões adicionais são geradas e escritas e assim por diante.

# OTIMIZAÇÃO DE CONSULTAS

## REGRA DE EQUIVALÊNCIA

Ela pode transformar uma expressão em outra, preservando a equivalência.

Preservar a equivalência significa que as relações geradas pelas duas expressões tem o mesmo conjunto de atributos e contém o mesmo conjunto de tuplas, embora seus atributos possam estar ordenados de forma diferente.

OTIMIZADOR – usa as regras de equivalência para transformar expressões em outras logicamente equivalentes.

# OTIMIZAÇÃO DE CONSULTAS

Exemplo (bancário):

Agencia

| nomeAgencia | cidade | fundos |
|---|---|---|

Conta

| nomeAgencia | conta | saldo |
|---|---|---|

Cliente

| nomeCliente | conta |
|---|---|

→ Expressão original:

$\pi_{\text{nomeCliente}}(\sigma_{[\text{cidade}=\text{"São Paulo"}]}(\text{Agencia} \times (\text{Conta} \times \text{Cliente})))$

→ que foi transformada em:

$\pi_{\text{nomeCliente}}((\sigma_{[\text{cidade}=\text{"São Paulo"}]}(\text{Agencia})) \times (\text{Conta} \times \text{Cliente}))$

→ que é equivalente a expressão original, mas gera relações intermediárias (temporárias) menores.

# OTIMIZAÇÃO DE CONSULTAS

- As regras identificam somente as equivalências das expressões, sem dizer qual é a melhor em termos de custo (onerosidade);

- Regras de equivalência múltipla podem ser usadas em uma consulta ou em partes dela;

Suponha que na consulta original fosse adicionada também a restrição de **clientes** com **saldo** maior que 1000 (mil).

não é possível

$\pi_{nomeCliente}(\sigma_{[cidade="São Paulo" \wedge saldo>1000]}(Agencia) \times (Conta \times Cliente))$

# OTIMIZAÇÃO DE CONSULTAS

$\pi_{nomeCliente}(\sigma_{[cidade="São Paulo" \wedge saldo>1000]}((Agencia \times Conta) \times Cliente))$

$\rightarrow$ tem-se então que:

$\pi_{nomeCliente}((\sigma_{[cidade="São Paulo" \wedge saldo>1000]}(Agencia \times Conta)) \times Cliente)$

$\rightarrow$ Examinando a sub-expressão da expressão, usando as regras, tem-se:

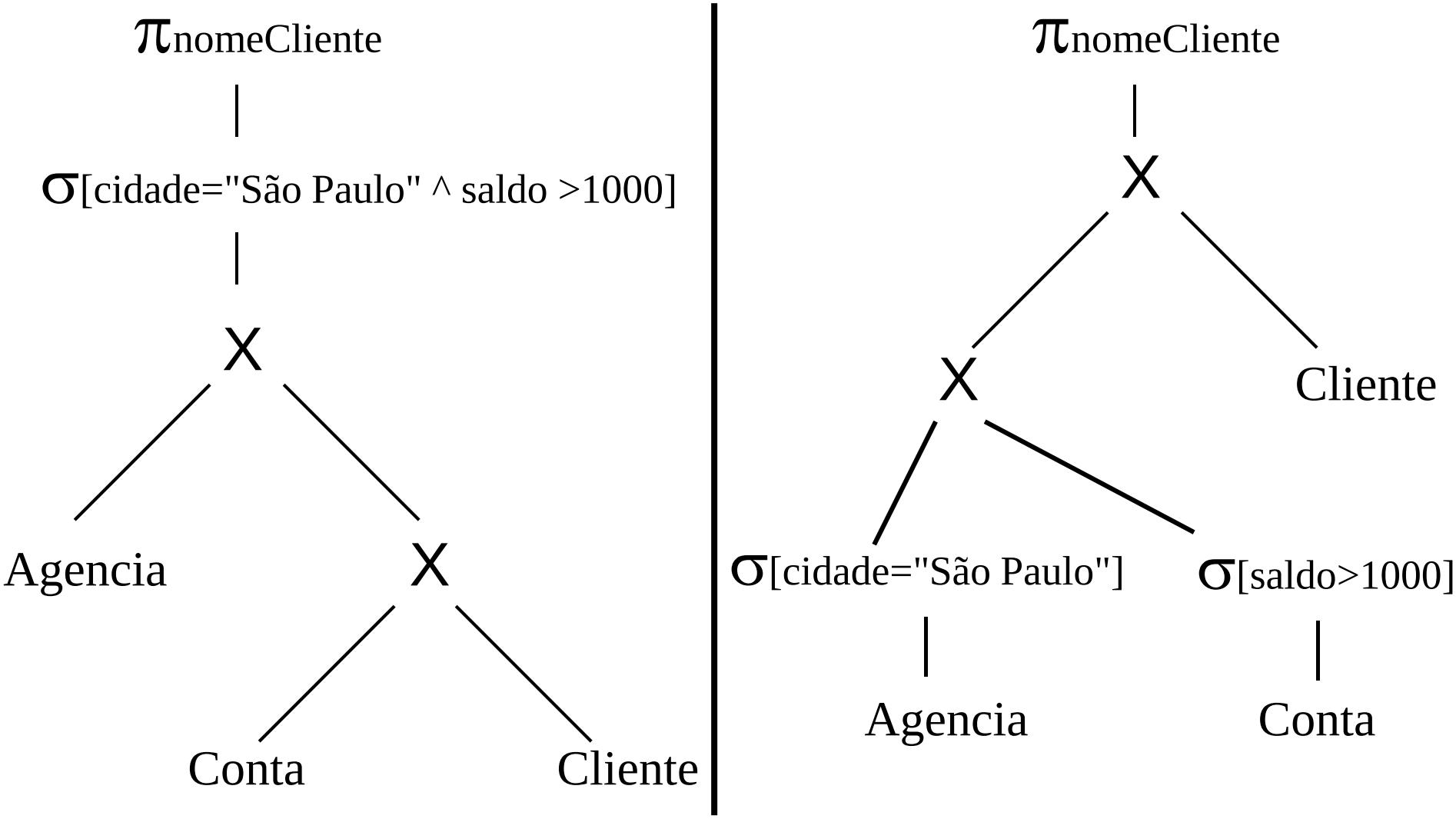
$\sigma_{[cidade="São Paulo"]}(\sigma_{[saldo>1000]}(Agencia \times Conta))$

$\rightarrow$ podendo ainda:

$\sigma_{[cidade="São Paulo"]}(Agencia) \times (\sigma_{[saldo>1000]}(Conta)$

Observe as transformações representadas nos gráficos a seguir:

# OTIMIZAÇÃO DE CONSULTAS



# OTIMIZAÇÃO DE CONSULTAS

- O conjunto de regras de equivalência é dito mínimo se nenhuma regra pode ser derivada de qualquer combinação das outras;
- O exemplo anterior não é mínimo;

Usando um novo exemplo tem-se:

$\pi_{\text{nomeCliente}}(\sigma_{[\text{cidade}=\text{"São Paulo"}]}(\text{Agencia}) \times \text{Conta}) \times \text{Cliente})$

$\rightarrow$ sendo a sub-expressão:

$(\sigma_{[\text{cidade}=\text{"São Paulo"}]}(\text{Agencia}) \times \text{Conta})$

A relação resultante da junção terá o seguinte esquema:

| nomeAgencia | cidade | fundos | nomeAgencia | conta | saldo |
|---|---|---|---|---|---|

# OTIMIZAÇÃO DE CONSULTAS

- Pode-se **eliminar vários atributos** de um esquema por meio da projeção;
- Os únicos atributos a serem mantidos são os que aparecem no:
  - resultado final da consulta;
  - necessários para processar as operações subsequentes;
- Reduzindo as colunas necessárias para a manipulação da consulta, se está diminuindo o custo da operação;

→ Assim, pode-se modificar o exemplo anterior para:

$$\pi_{\text{nomeCliente}}((\pi_{\text{conta}}((\sigma_{[\text{cidade}=\text{“São Paulo”}]}(\text{Agencia})) \times \text{Conta})) \times \text{Cliente})$$

→ A projeção $\pi_{\text{conta}}$ reduz o tamanho dos resultados da junção intermediária.

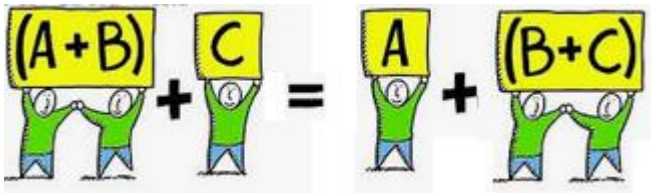
# OTIMIZAÇÃO DE CONSULTAS

## ORDENAÇÃO DE JUNÇÃO

A ordenação de junções é importante para reduzir o tamanho dos resultados intermediários.

$$(\text{relação}_1 \text{X relação}_2) \text{X relação}_3 = \text{relação}_1 \text{X } (\text{relação}_2 \text{ X relação}_3)$$

↳ **junção é ASSOCIATIVA**
(produz mesmo resultado)




Apesar dessas expressões serem equivalentes, seus custos podem diferir.

# OTIMIZAÇÃO DE CONSULTAS

Análise o exemplo a seguir:

$\pi_{\text{nomeCliente}}((\sigma_{[\text{cidade}=\text{"São Paulo"}]}(\text{Agencia}))\text{X Conta X Cliente})$

$\rightarrow$ Poderia ser realizada primeiro a junção de conta e cliente, sendo em seguida a junção com a sub-expressão:

$$\sigma_{[\text{cidade}=\text{"São Paulo"}]}(\text{Agencia})$$

Esta junção (Conta X Cliente) produzirá uma relação grande, pois conterá uma tupla para cada conta. Análise então a sub-expressão proposta a seguir:

$$\sigma_{[\text{cidade}=\text{"São Paulo"}]}(\text{Agencia}) \text{ X Conta}$$

# OTIMIZAÇÃO DE CONSULTAS

Provavelmente, será uma relação **menor** (pequena), pois trata-se de uma junção dos dados, só para as contas da cidade de São Paulo ( [cidade = "São Paulo"] ).

A junção natural também é **COMUTATIVA**, facilitando assim a modificação na ordem de apresentação dos atributos no resultado final.


Produzem o mesmo resultado,
apesar da ordem ser diferente.

# OTIMIZAÇÃO DE CONSULTAS

Usando destas propriedades da junção natural (ASSOCIATIVA E COMUTATIVA) pode-se reescrever a expressão da seguinte forma:

$$\pi_{\text{nomeCliente}}(((\sigma_{[\text{cidade}=\text{"São Paulo"}]}(\text{Agencia})) \times \text{Cliente}) \times \text{Conta})$$

→ Podendo assim fazer primeiro a seleção (σ) depois a junção com conta;

→ Porém, observe que não existem atributos em comum entre Agencia e Cliente, gerando assim o **produto cartesiano** entre estas relações;

→ Como a relação temporária será fruto de um produto cartesiano, provavelmente grande, esta estratégia será **rejeitada** pelo SGBD.

# OTIMIZAÇÃO DE CONSULTAS

A geração de equivalências é um processo "CARO", pois consome ESPAÇO e TEMPO, por isso tem-se:

- Sub-expressões **compartilhadas**;
  - Reduz a necessidade de espaço significativamente;
  - Muito usada em otimizadores de consultas;

Considerando as estimativas de custo, um otimizador pode ser capaz de evitar o exame de algumas expressões, reduzindo também o tempo necessário para a otimização.

# OTIMIZAÇÃO DE CONSULTAS

## OTIMIZAÇÃO BASEADA EM CUSTO

Este tipo de otimização gera uma faixa de planos de avaliação a partir de uma determinada consulta.

- Ele usa das regras de equivalência para gerar os novos planos e escolhe entre eles o de menor CUSTO;
- Para uma consulta complexa, a quantidade de planos de avaliação diferentes podem ser grandes;
- Não será necessário gerar todas as expressões equivalentes a uma determinada consulta;

# OTIMIZAÇÃO DE CONSULTAS

Observe o exemplo a seguir:

$(\text{relação}_1 \text{X relação}_2 \text{X relação}_3) \text{ X relação}_4 \text{ X relação}_5$

**12** possíveis combinações

01 => (r1 X r2 X r3) X r4 X r5
02 => (r1 X r3 X r2) X r4 X r5
03 => (r2 X r1 X r3) X r4 X r5
04 => (r2 X r3 X r1) X r4 X r5
05 => (r3 X r1 X r2) X r4 X r5
06 => (r3 X r2 X r1) X r4 X r5
07 => (r1 X r2 X r3) X r5 X r4
08 => (r1 X r3 X r2) X r5 X r4
09 => (r2 X r1 X r3) X r5 X r4
10 => (r2 X r3 X r1) X r5 X r4
11 => (r3 X r1 X r2) X r5 X r4
12 => (r3 X r2 X r1) X r5 X r4

# OTIMIZAÇÃO DE CONSULTAS

$(\text{relação}_1 \text{ X } \text{relação}_2 \text{ X } \text{relação}_3) \text{ X } \text{relação}_4 \text{ X } \text{relação}_5$

→ Encontrar a melhor ordem de junção para o subconjunto das relações $\{r_1, r_2, r_3\}$;

→ Resultado da junção anterior será usado nas junções posteriores (com $r_4$ e $r_5$);

→ Ignora as outras ordens de junção mais caras para $r_1, r_2, r_3$;

→Tem-se assim um custo de (12 + 12) alternativas ao invés de 144 ($r_1, r_2, r_3 , r_4 , r_5$);

# OTIMIZAÇÃO DE CONSULTAS

→ O uso de técnicas de propagação dinâmica reduz significativamente o número de expressões examinadas (ou avaliadas);

→ A ordem em que as tuplas são geradas pela junção também são importantes para encontrar a melhor ordem global de junção, pois isso pode afetar o custo das junções posteriores.

$$(\text{relação}_1 \text{X relação}_2 \text{X relação}_3) \rightarrow \text{X relação}_4 \text{ X relação}_5$$

junções posteriores

# OTIMIZAÇÃO DE CONSULTAS

## OTIMIZAÇÃO HEURÍSTICA

Este tipo de otimização faz uso de heurísticas para reduzir a quantidade de planos de avaliação que serão completamente examinados.

Experiência Conhecimento ...

(geralmente sem comprovação matemática)

# OTIMIZAÇÃO DE CONSULTAS

- Faz-se primeiro uma suposição heurística para um "BOM" plano, com estimativas sobre seus custos;
- Uma desvantagem da otimização baseada no custo é o próprio custo da otimização;
- Otimizações heurísticas reduzem significativamente o *overhead* da otimização de consultas;
- Alguns sistemas usam somente a heurística, não usando a otimização baseada em custos.

# OTIMIZAÇÃO DE CONSULTAS

## ATIVIDADE EM GRUPO

TRABALHOS (livro do KORTH capítulo 12 da 3ª edição ):

1- Tipos de Seleção (4 - pág. 386)
2- Tipos de Junção (5 - pág. 399)
3- Outras Operações (5 - pág. 411)
4- Regras de Equivalência (12 - pág. 421)

Atividade (painel integrado):

- desenvolver o estudo dos temas pelo grupo
- discutir e anotar os conteúdos mais relevantes
- trocar os membros de cada grupo que explicaram aos novos companheiros de grupo o material estudado

# Referência de Criação e Apoio ao Estudo

## Material para Consulta e Apoio ao Conteúdo

- ELMASRI, R. e Navathe, S. B., Fundamentals of Database Systems, Addison-Wesley, 3rd edition, 2000
  - Capítulo 18
- SILBERSCHATZ, A. & Korth, H. F., Sistemas de Banco de Dados
  - Capítulo 12
- DATE, C. J., Introdução a Sistemas de Banco de Dados, Editora Campus
  - Páginas 466 - 504
- Universidade de Brasília (UnB Gama)
  - https://cae.ucb.br/conteudo/unbfga
    (escolha no menu superior a opção **Labor. Banco Dados (SBD2)** )