Otimização de consultas

Carlos A. Heuser

2006

Necessidade de otimização

- Uma consulta a uma base de dados pode ser expressa através de muitas expressões de álgebra relacional diferentes
- O ideal é que o tempo de execução de diferentes expressões seja idêntico:
  - **O programador não deve necessitar de conhecimentos acerca do funcionamento interno do SGBD**

Necessidade de otimização - exemplo

- Exemplo de consulta sobre o BD acadêmico
  - Obter os nomes dos alunos que cursaram com conceito "A" uma disciplina do curso de nome "Computação" que ofereça mais de 5 créditos
  - Uma solução possível:

```
π NomeAl
  (σ (NomeCur="Computação" and
      Conceito="A" and
      CredDisc>5)
    (Aluno ⋈
      (Histórico ⋈
        (Disciplina ⋈
         (Curric ⋈ Curso)))) )
```


Necessidade de otimização - expressão alternativa

```
π NomeAl
  ((Aluno
   ⋈
   ((σ (Conceito="A") Histórico)
   ⋈
    ((σ (NomeCur="Computação") Curso)
    ⋈
     (Curric
     ⋈
      (σ (CredDisc>5) Disciplina)))
```

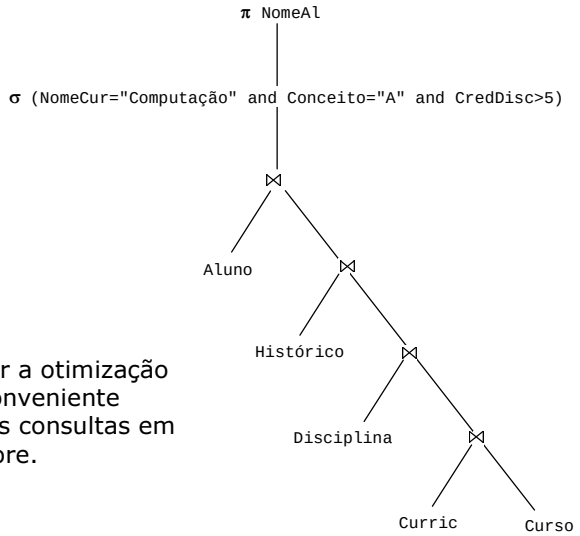
Otimização algébrica

- Baseia-se em **regras heurísticas** de transformação de expressões relacionais
- Objetivo:
  - Obter expressões de álgebra relacional
    equivalentes à consulta original e
    cuja execução direta demande menos tempo e menos recursos de máquina
- **Regra geral** da otimização:
  - Diminuir os resultados intermediários

Otimização algébrica - como fazer

- **Executar seleções e projeções o mais cedo possível**
  Essas operações geram resultados menores (menos linhas, menos colunas) que seus operandos
- **Executar operações binárias o mais tarde possível**
  Operações binárias aumentam o volume de dados (efeito multiplicativo) e são onerosas em termos de tempos de execução
- **Transformar as operações de produto cartesiano sempre que possível em operações de junção**

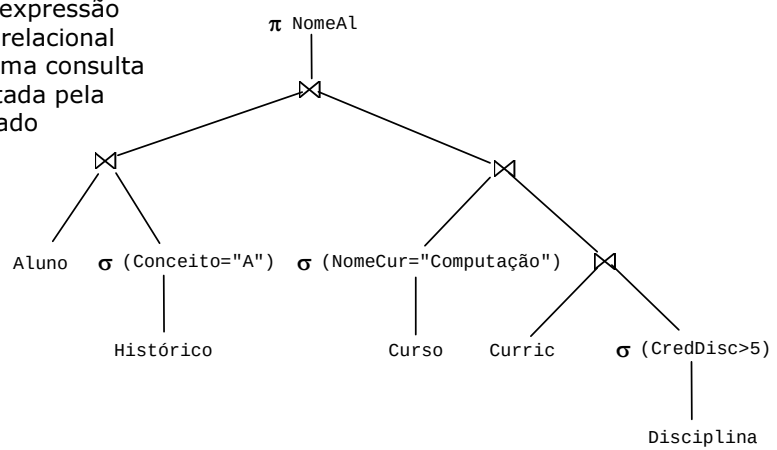
Árvore de consulta (plano de execução)



Para visualizar a otimização algébrica é conveniente representar as consultas em forma de árvore.

Árvore de consulta - segunda consulta

A segunda expressão de álgebra relacional para a mesma consulta é representada pela árvore ao lado



Exemplo de otimização

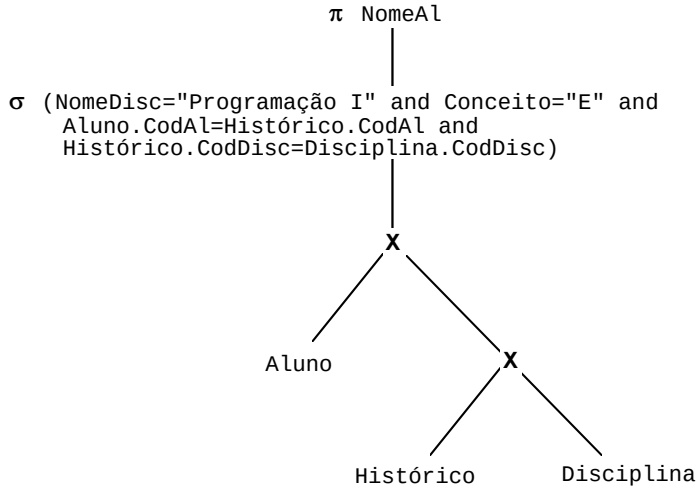
- Consulta sobre o BD acadêmico

  Obter os nmes dos alunos que obtiveram conceito "E" em disciplina denominada "Programação I"

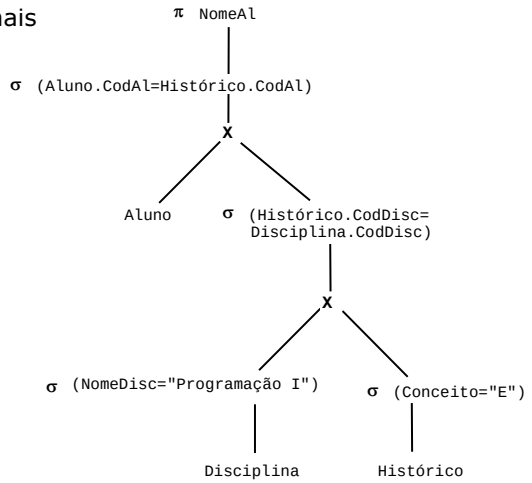
- Expressão em álgebra

```
π NomeAl
(σ (NomeDisc="Programação I" and Conceito="E"
        and
   Aluno.CodAl=Histórico.CodAl
        and
    Histórico.CodDisc=Disciplina.CodDisc)
    (Aluno X
      (Histórico X
        (Disciplina))) )
```

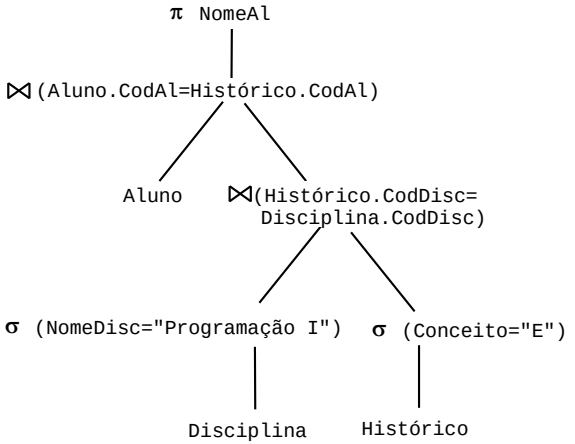

Árvore de consulta (expressão original)

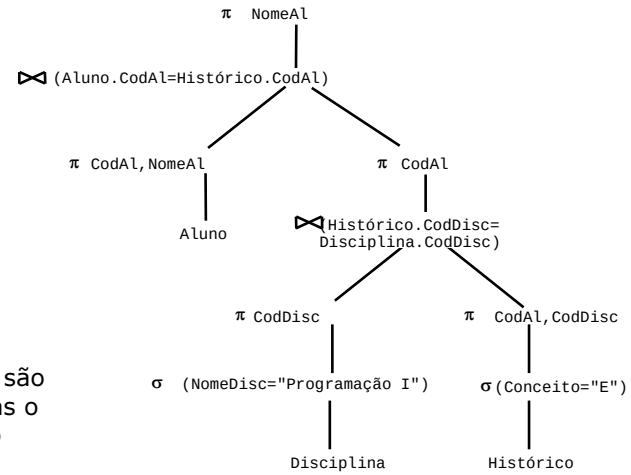


Seleções são executadas mais cedo



Seleções são combinadas com produtos cartesianos, formando junções





Projeções são executadas o mais cedo possível

Otimização baseada em estimativas de custo

- A otimização heurística não leva em conta **volumes de dados** nem **tempos de acesso**
- Com isso, pode gerar árvores de consulta menos eficientes
- Exemplo:

  Caso a tabela de alunos fosse consideravelmente menor que as demais, poderia ser mais eficiente incluí-la na primeira operação de junção
- Otimização baseada em estimativas de custo:
  - Para diferentes árvores de consulta, equivalentes em performance do ponto de vista das regras heurísticas, é escolhida uma que se estima ser de menor custo de implementação
  - **Custo** = **volume de acesso** a disco
  - Como é oneroso verificar volumes a cada transação, SGBD normalmente coleta dados estatísticos sobre as tabelas (número de linhas, número de valores diferentes por coluna, …)

Otimização em produtos comerciais

- Informix 5.0, Ingres 6.4, Oracle 7.0 e Sybase 4.9 usam **otimização heurística**.
- Todos combinam otimização heurística, com estimativa de custo baseada em estatísticas colecionadas pelo próprio SGBD
- Todos mantém **estatísticas** sobre as tabelas (número de linhas, número de entradas sobre índices, …)
- Ingres procura manter estatísticas sobre **distribuição de valores** em colunas (mínimo, média, máximo)
- Oracle e Ingres podem coletar estatísticas através de **amostragem** em tabelas, evitando varredura da tabela completa
- Nenhum otimizador usa custos de transmissão em rede nas estimativas de custo
- Oracle, Ingres e Informix são capazes de mover pequenas tabelas locais para estações remotas, onde ocorrerá a junção com grandes tabelas