

Processamento de Consultas

Banco de Dados: Teoria e Prática

André Santanchè

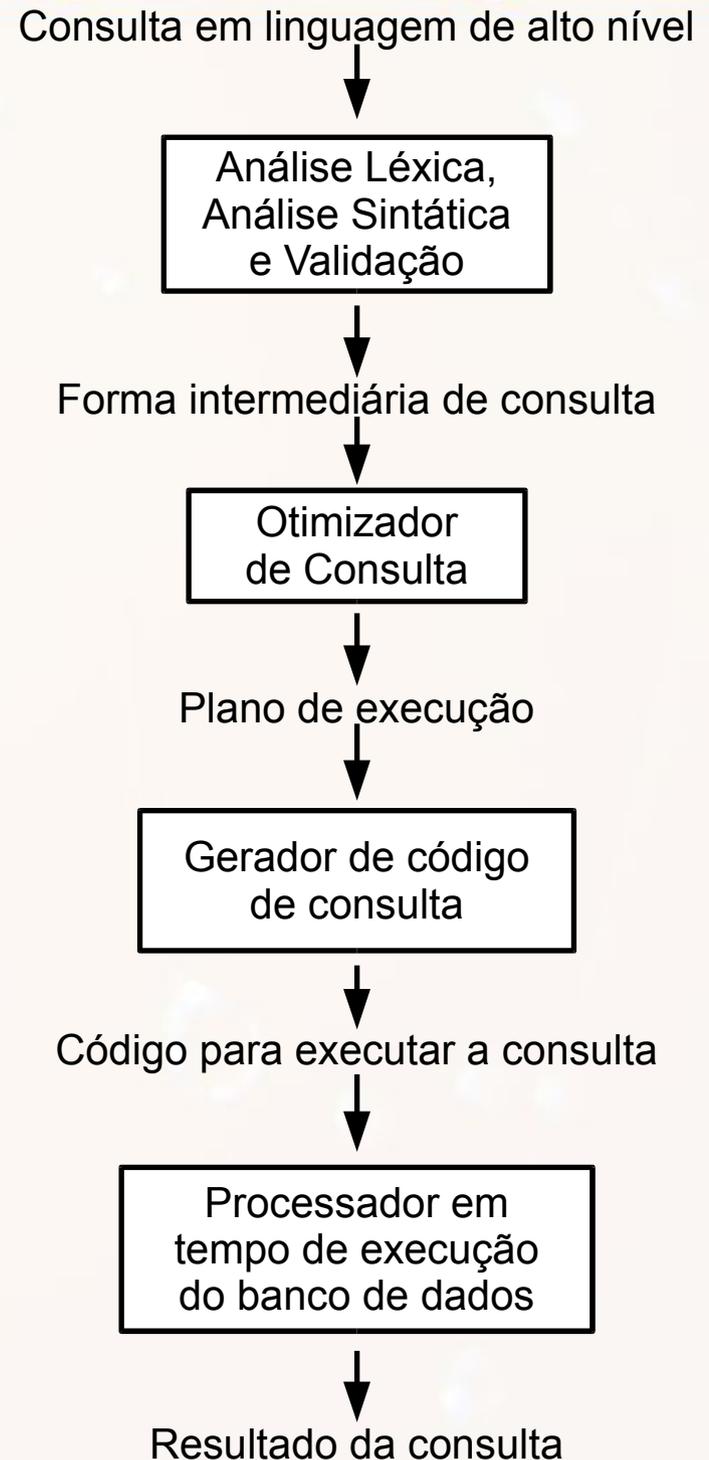
Instituto de Computação - UNICAMP

Outubro 2012

Execução de Consulta

-

Passos Típicos (Elmasri, 2010)



Execução de Consulta

-

Passos Típicos

Consulta em linguagem de alto nível

Análise Léxica,
Análise Sintática
e Validação

Forma intermediária de consulta

Otimizador
de Consulta

Plano de execução

Gerador de código
de consulta

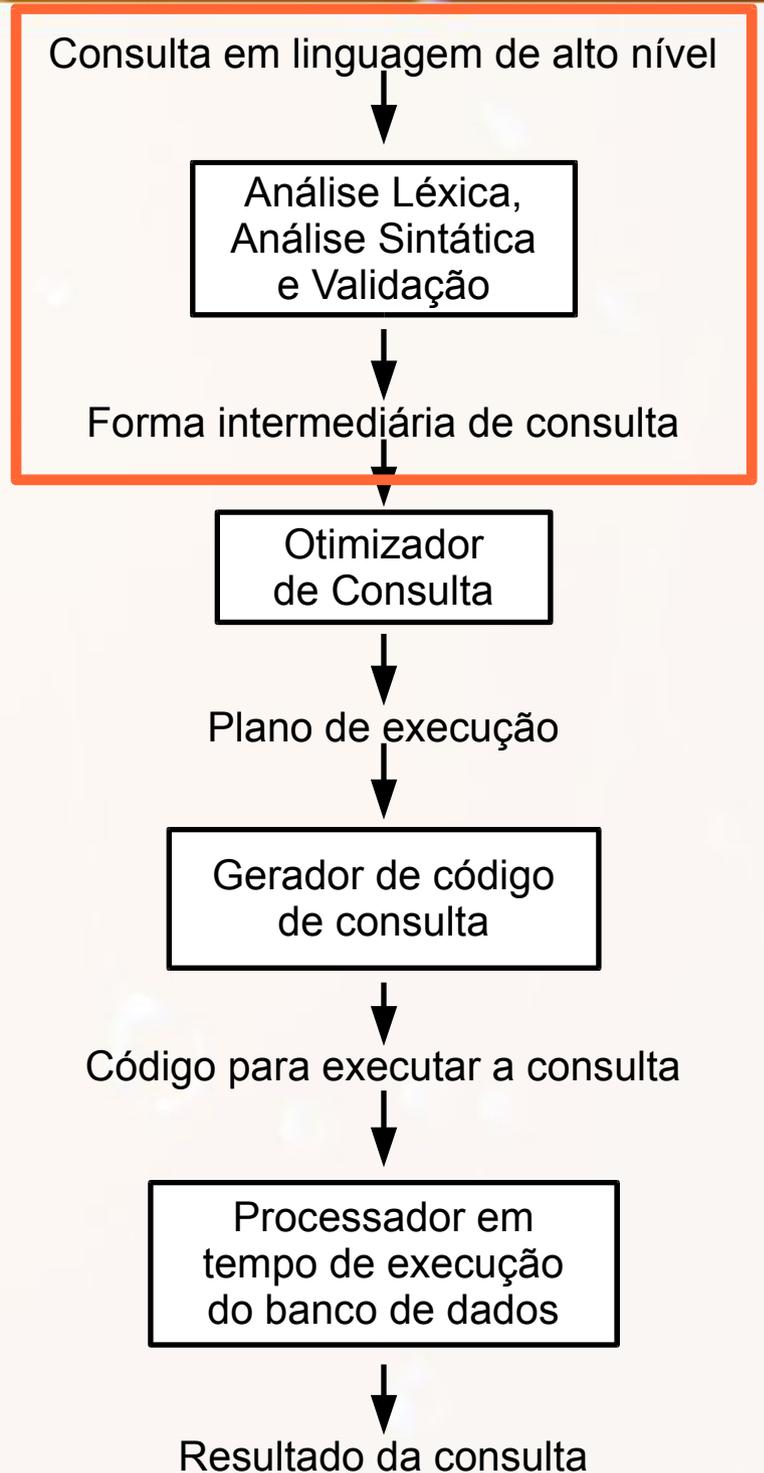
Código para executar a consulta

Processador em
tempo de execução
do banco de dados

Resultado da consulta

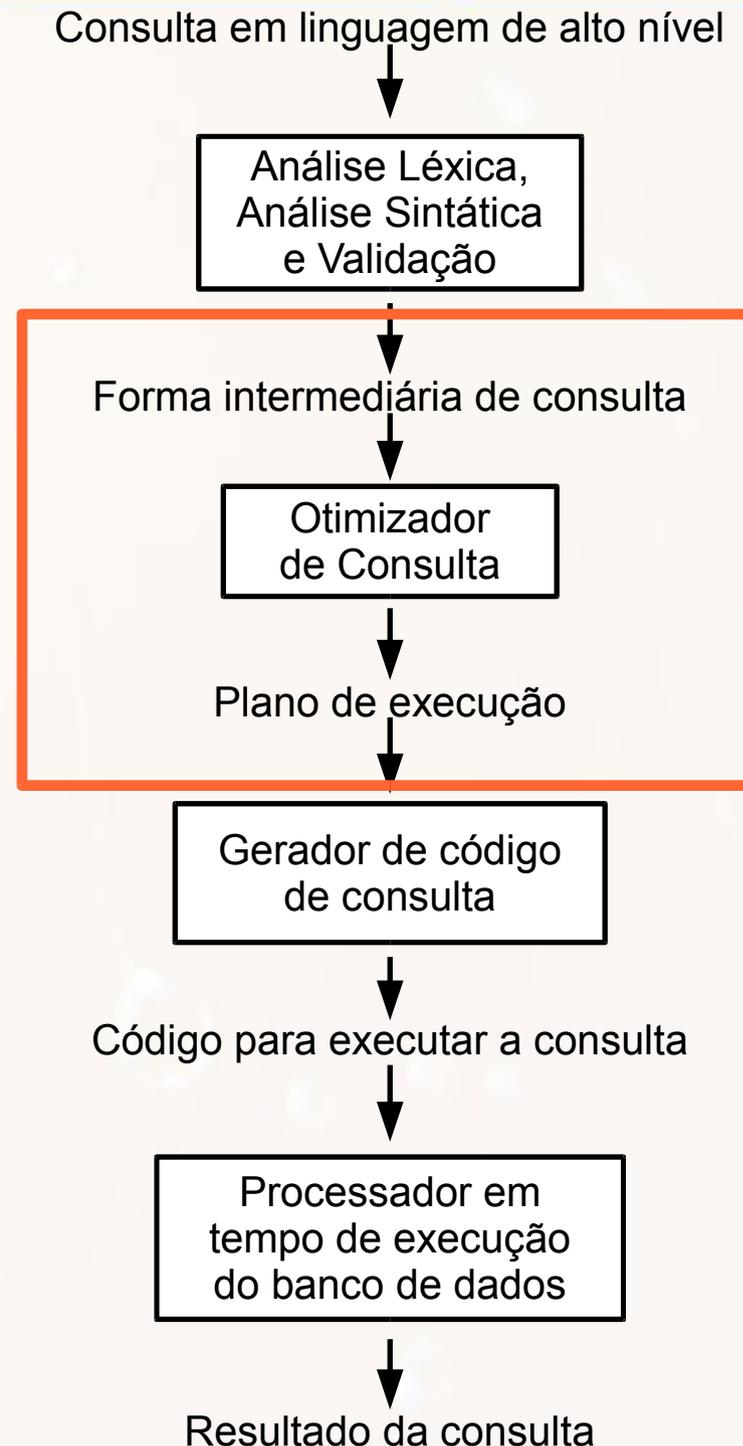
Análise e Validação

- Análise e Validação
 - Análise léxica
 - Análise sintática
 - Validação
- Representações internas:
 - árvore de consulta
 - grafo de consulta



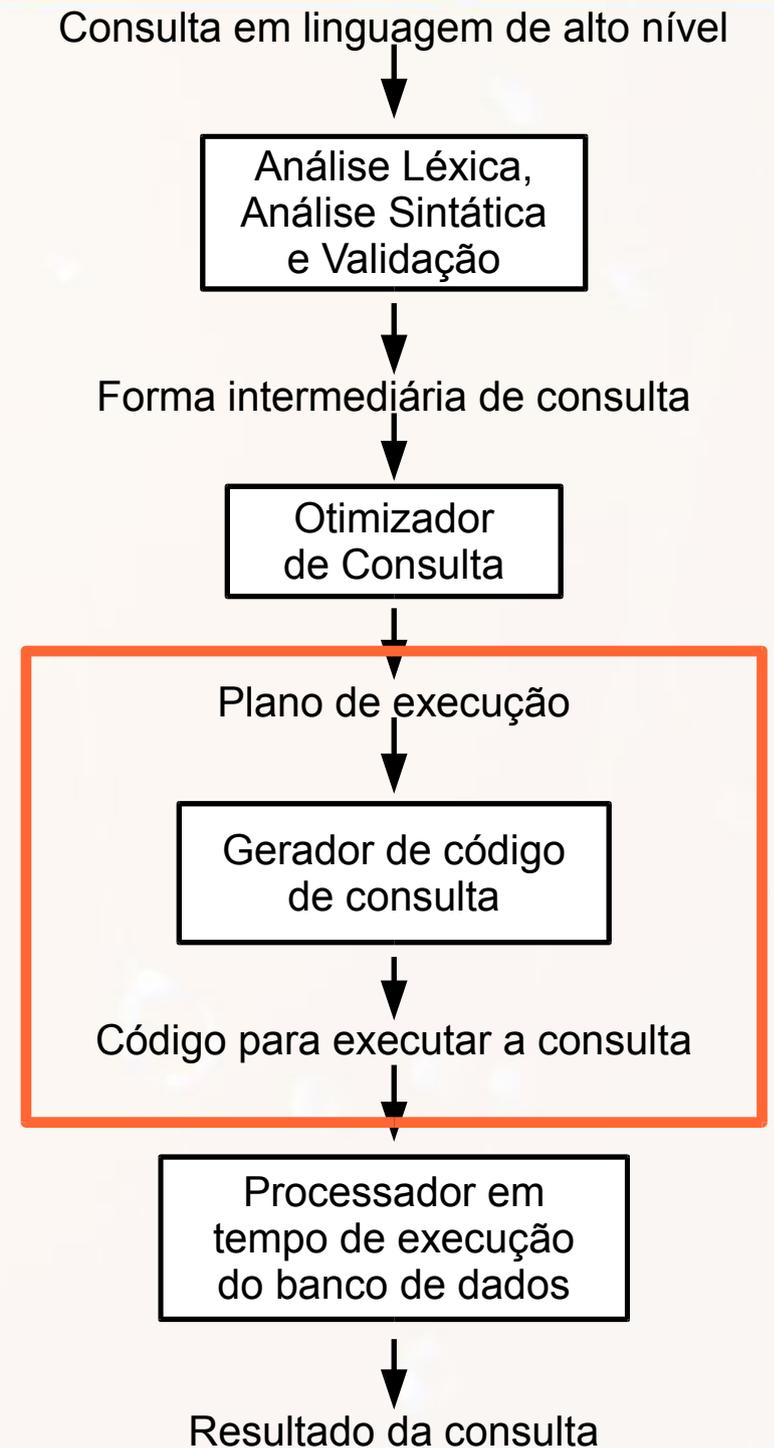
Estratégia de Execução

- Consulta possui muitas estratégias de execução possíveis
- Planejamento da Estratégia de Execução
 - Otimização → processo de escolha da estratégia adequada (razoavelmente eficiente)



Código da Consulta

- Pode ser:
 - Executado diretamente
 - modo interpretado
 - Armazenado e executado quando necessário
 - modo compilado



Execução do Código

- Processador executa código da consulta
- Produz resultado da execução

Consulta em linguagem de alto nível

Análise Léxica,
Análise Sintática
e Validação

Forma intermediária de consulta

Otimizador
de Consulta

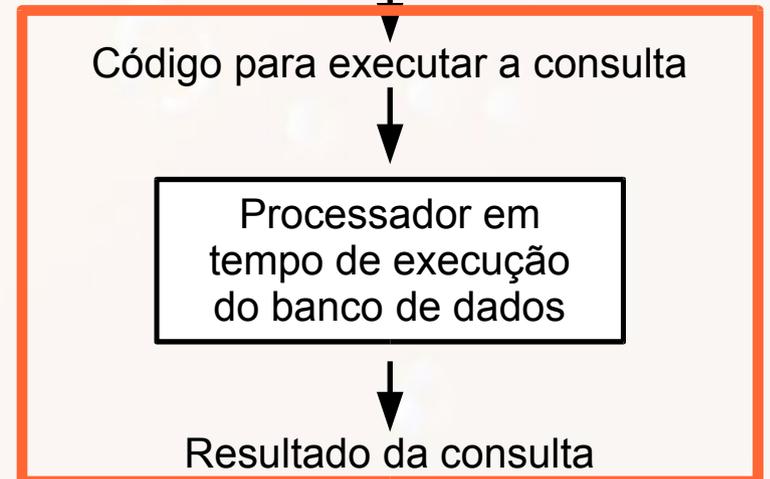
Plano de execução

Gerador de código
de consulta

Código para executar a consulta

Processador em
tempo de execução
do banco de dados

Resultado da consulta



**Ênfase desta aula:
Otimização de Consultas**

Consultas Declarativas

- “O quê” ao invés de “Como”
- Otimização de consulta
 - Solução razoavelmente eficiente (Elmasri, 2011)
 - Solução ótima pode ser muito custosa

Consulta SQL em Álgebra Relacional

- Consulta SQL → Álgebra Relacional Estendida
 - Inclui operadores como COUNT, SUM e MAX
- Consulta SQL decomposta em blocos
 - Bloco de Consulta ou Bloco Simples:
 - Contém uma única expressão SELECT-FROM-WHERE (GROUP BY e HAVING se houver)
 - Sem aninhamento
 - Consultas aninhadas são identificadas como consultas independentes

Decomposição em Blocos

Exemplo

■ Tabela

Pessoa (Codigo, Nome, Telefone, AnoFiliacao)

■ Nome dos filiados mais antigos:

```
SELECT Codigo, Nome
FROM PESSOA
WHERE AnoFiliacao = (SELECT MIN(AnoFiliacao))
                     FROM PESSOA)
```

■ Blocos

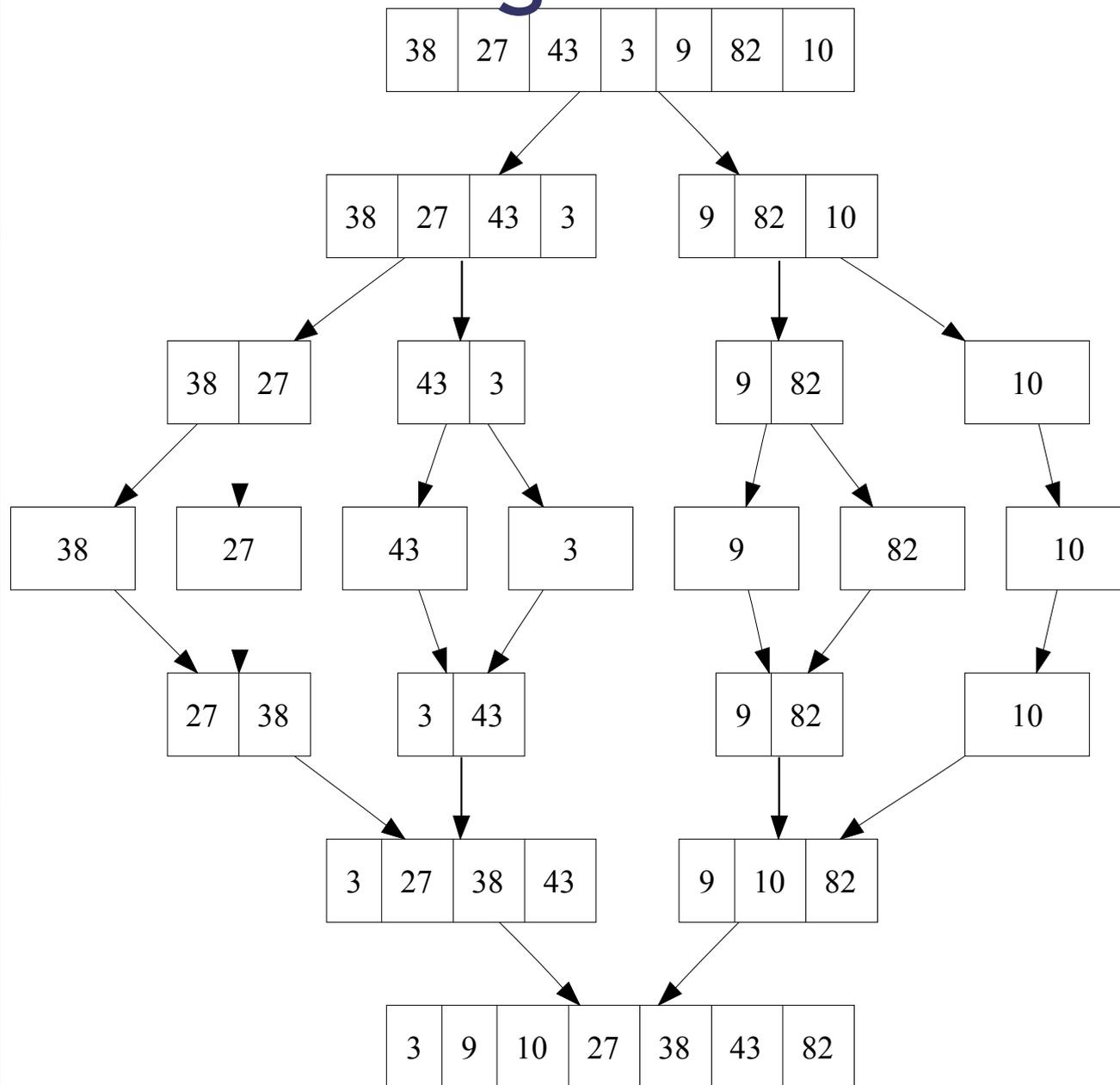
```
① SELECT Codigo, Nome
   FROM PESSOA
   WHERE AnoFiliacao = (referência ②)
```

```
② SELECT MIN(AnoFiliacao)
   FROM PESSOA
```

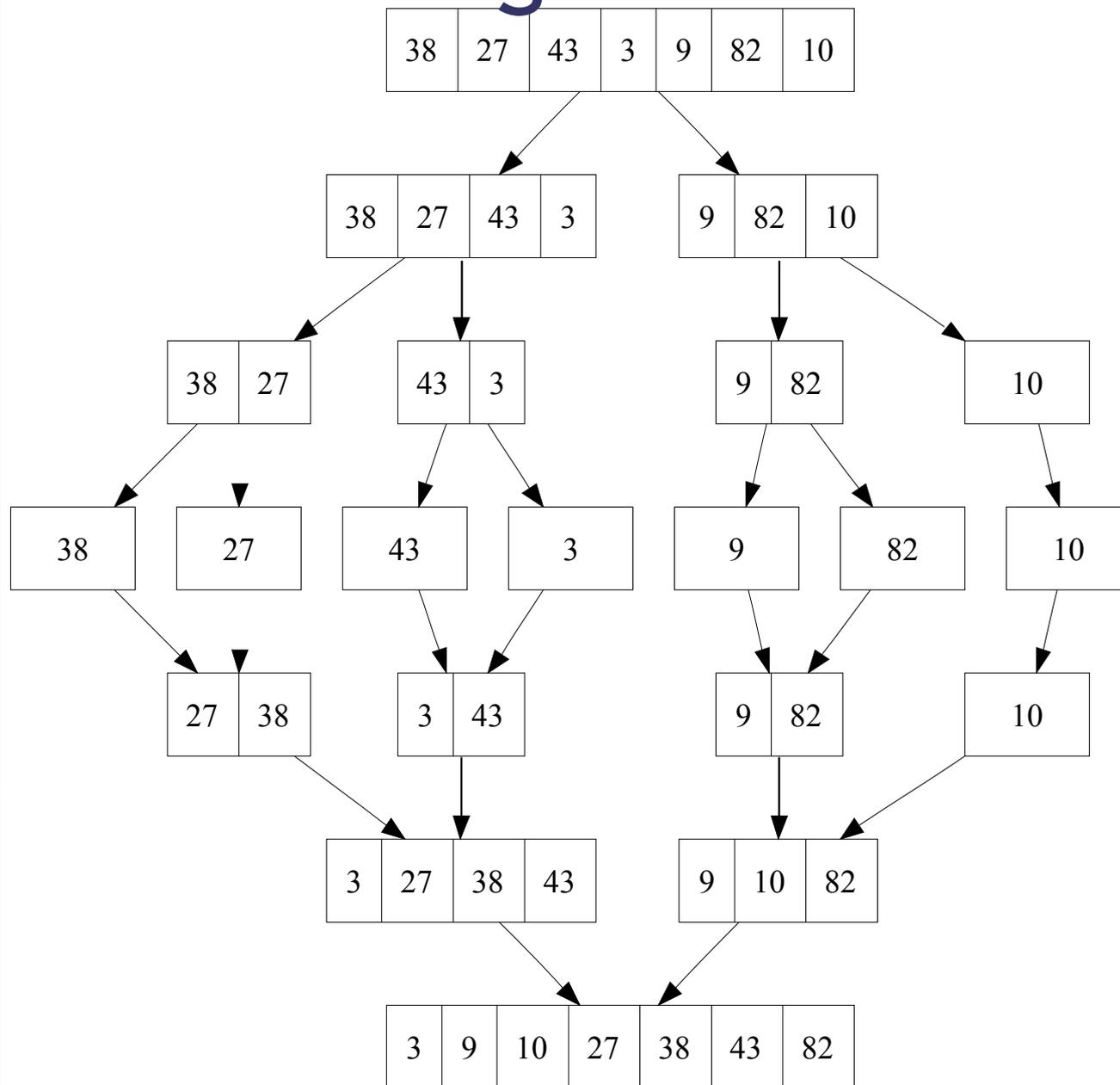
Algoritmos para Operações

Ordenação Externa

Merge Sort

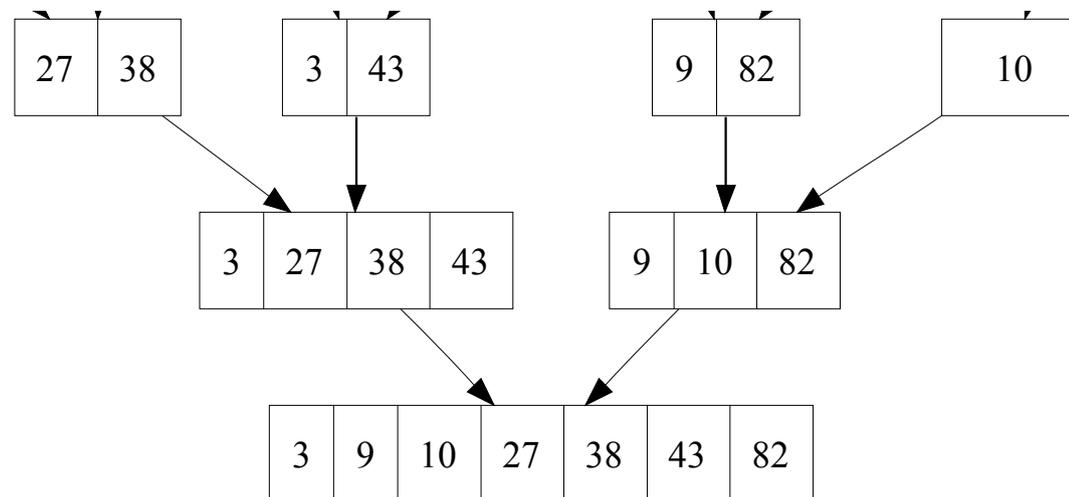


Merge Sort



Ordenação Externa

38	27	43	3	9	82	10
----	----	----	---	---	----	----



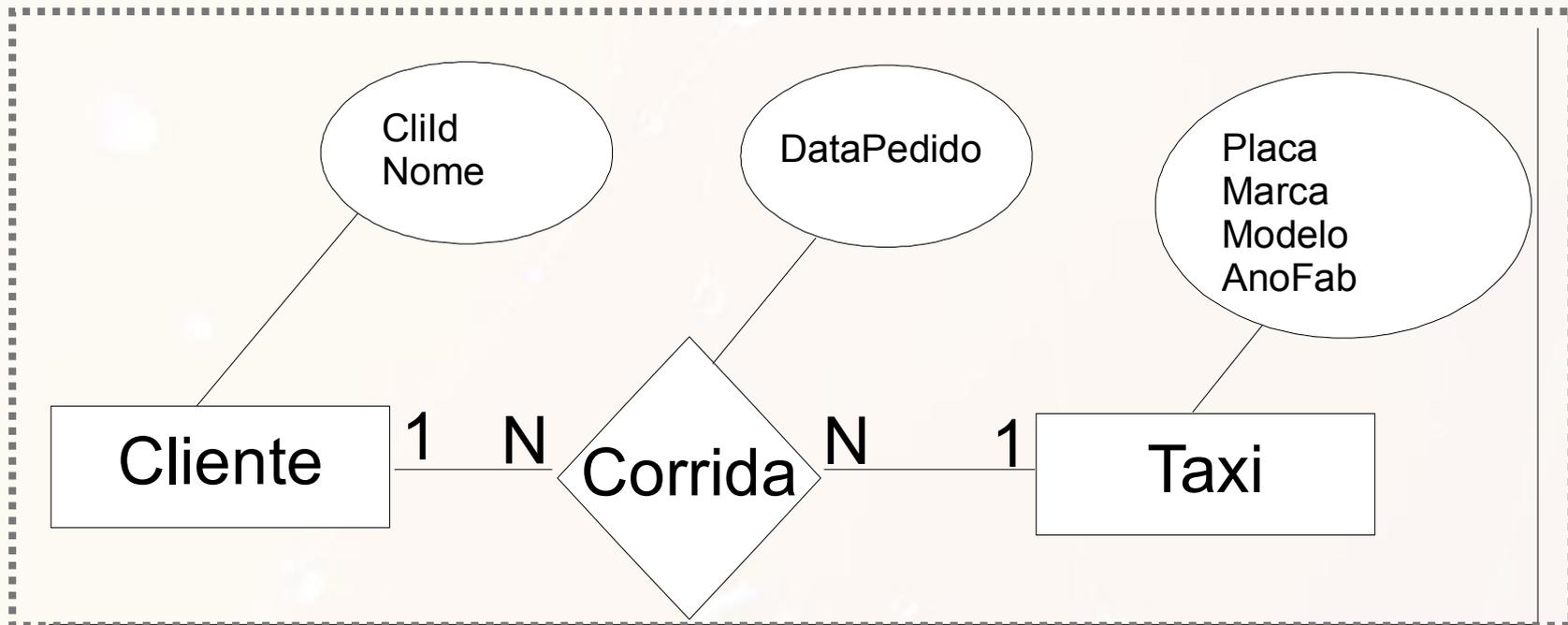
Ordenação Externa

Números

- bd - blocos em disco
- bm - blocos de memória
- Ordenação
 - $2 \cdot bd$ transferências (leitura e gravação)
- Merge
 - $2 \cdot bd$ transferências a cada estágio
 - $\lceil bd/bm \rceil$ rodadas de ordenação
 - $\lceil \log_{bm-1}(bd/bm) \rceil$ níveis
- Custo: $(2 \cdot bd) + (2 \cdot bd * \lceil \log_{bm-1}(bd/bm) \rceil)$

Seleção

Esquema Conceitual - Exemplo Táxis



Este é um subconjunto do Estudo de Caso proposto “Despacho e controle de Táxis via terminais móveis ligados on-line com um sistema multi-usuário” por prof. Geovane Cayres Magalhães

Tabelas para exemplo - Táxis

Táxi (TX)

<u>Placa</u>	Marca	Modelo	AnoFab
DAE6534	Ford	Fiesta	1999
DKL4598	Wolkswagen	Gol	2001
DKL7878	Ford	Fiesta	2001
JDM8776	Wolkswagen	Santana	2002
JJM3692	Chevrolet	Corsa	1999



Corrida (R1)

<u>ClId</u>	<u>Placa</u>	<u>DataPedido</u>
1755	DAE6534	15/02/2003
1982	JDM8776	18/02/2003



Seleção?

⑩ Placa='JDM8776'(TX)

<u>Placa</u>	Marca	Modelo	AnoFab
DAE6534	Ford	Fiesta	1999
DKL4598	Wolkswagen	Gol	2001
DKL7878	Ford	Fiesta	2001
JDM8776	Wolkswagen	Santana	2002
JJM3692	Chevrolet	Corsa	1999

Exatamente Igual Chave Primária

⑩ Placa='JDM8776'(TX)

<u>Placa</u>	Marca	Modelo	AnoFab
DAE6534	Ford	Fiesta	1999
DKL4598	Wolkswagen	Gol	2001
DKL7878	Ford	Fiesta	2001
JDM8776	Wolkswagen	Santana	2002
JJM3692	Chevrolet	Corsa	1999

Exatamente Igual Outra Chave

⑩ AnoFab=2002(TX)

<u>Placa</u>	Marca	Modelo	AnoFab
DAE6534	Ford	Fiesta	1999
DKL4598	Wolkswagen	Gol	2001
DKL7878	Ford	Fiesta	2001
JDM8776	Wolkswagen	Santana	2002
JJM3692	Chevrolet	Corsa	1999

Seleção?

⑩ AnoFab=2002(TX)

<u>Placa</u>	Marca	Modelo	AnoFab
DAE6534	Ford	Fiesta	1999
DKL4598	Wolkswagen	Gol	2001
DKL7878	Ford	Fiesta	2001
JDM8776	Wolkswagen	Santana	2002
JJM3692	Chevrolet	Corsa	1999

Seleção?

⑩ AnoFab > 2000 (TX)

<u>Placa</u>	Marca	Modelo	AnoFab
DAE6534	Ford	Fiesta	1999
DKL4598	Wolkswagen	Gol	2001
DKL7878	Ford	Fiesta	2001
JDM8776	Wolkswagen	Santana	2002
JJM3692	Chevrolet	Corsa	1999

Faixa (>, <, >=, <=)

⑩ AnoFab>2000(TX)

<u>Placa</u>	Marca	Modelo	AnoFab
DAE6534	Ford	Fiesta	1999
DKL4598	Wolkswagen	Gol	2001
DKL7878	Ford	Fiesta	2001
JDM8776	Wolkswagen	Santana	2002
JJM3692	Chevrolet	Corsa	1999

Algoritmos de Seleção

- Exatamente igual
 - chave primária
 - outra chave
- $>$, $<$, $>=$, $<=$
- compostos

Algoritmos de Seleção

- Pesquisa linear
- Pesquisa binária
- Usando índice primário
- Usando chave hash
- Combinado com o índice primário
- Usando índice de agrupamento
- Usando índice secundário

Seleção Conjuntiva x Dijuntiva

- seleção conjuntiva - e.g., and
- seleção dijuntiva - e.g., or

Algoritmos de Seleção Conjuntiva

- Índice para uma das condições
- Índice composto envolvendo ambas as condições
- Índice individual para cada condição

Seletividade

- seletividade: valor entre 0 e 1
- n registros
- igualdade atributo único
 - seletividade: $1/n$

Seletividade

Atributo Não Único

- i valores
- i igualmente distribuído
- registros por valor?
- seletividade?

Seletividade

Atributo Não Único

- i valores
- i igualmente distribuído
- n/i registros por valor
- seletividade: $1/i$

Seletividade

Atributo Não Único

- primeiro as condições com valor menor de seletividade

Junção (Join)

Junção (Join) de Loop Aninhado

```
for each ti
  for each tj
    if match(ti, tj)
      add-result(ti, tj)
```

Junção de Loop Aninhado Números

- n_i - número de tuplas t_i
- n_j - número de tuplas t_j
- pares de tuplas? (comparações?)

Junção de Loop Aninhado

Números

- n_i - número de tuplas t_i
- n_j - número de tuplas t_j
- $n_i * n_j$ - pares de tuplas

Junção de Loop Aninhado

Números

- n_i - número de tuplas t_i
- n_j - número de tuplas t_j
- $n_i * n_j$ - pares de tuplas

- b_i - bloco de tuplas t_i
- b_j - bloco de tuplas t_j
- leituras de blocos?

Junção de Loop Aninhado

Números

- n_i - número de tuplas t_i
- n_j - número de tuplas t_j
- $n_i * n_j$ - pares de tuplas

- b_i - bloco de tuplas t_i
- b_j - bloco de tuplas t_j
- $b_i + b_j * n_i$ leituras de blocos

Junção de Loop Aninhado Números

- Situações:
 - Quantas transferências de bloco se todos os blocos estiverem na memória?
 - Quantas transferências se os blocos de um dos loops estiver todo na memória e qual deles escolher (bi ou bj)

Junção de Loop Aninhado Números

■ Situações:

- Quantas transferências de bloco se todos os blocos estiverem na memória?
 - $b_i + b_j$ transferências
- Quantas transferências se os blocos de um dos loops estiver todo na memória e qual deles escolher (b_i ou b_j)?
 - escolher b_j
 - $b_i + b_j$ transferências

Junção de Loop Aninhado em Bloco

```
for each bi
  for each bj
    for each ti
      for each tj
        if match(ti, tj)
          add-result(ti, tj)
```

Junção de Loop Aninhado em Bloco Números

- b_i - bloco de tuplas t_i
- b_j - bloco de tuplas t_j
- leituras de blocos?

Junção de Loop Aninhado em Bloco Números

- b_i - bloco de tuplas t_i
- b_j - bloco de tuplas t_j
- $b_i + b_j * b_i$ leituras de blocos

Outras Junções

- Junção Indexada
- Junção Merge
- Junção Hash

Projeção

- Recorte dos campos
- (?)

Projeção

- Recorte dos campos
- Registros sem duplicatas
 - SQL → padrão não eliminar duplicatas
 - DISTINCT → elimina duplicatas
 - Registros com garantia de ser únicos
 - e.g., contendo chave primária
 - Registros sem garantia de ser únicos
 - ordenação
 - hashing

Otimização de Consulta

SQL p/ Álgebra

■ Versão SQL

```
SELECT Codigo, Nome  
FROM PESSOA  
WHERE AnoFiliacao = 1990
```

■ Versão em álgebra

$\textcircled{7}$ Codigo, Nome ($\textcircled{10}$ AnoFiliacao=1990 (PESSOA))

■ Versão Árvore



Combinação de Operações usando Pipelining

- Uma consulta é mapeada em uma sequência de operações
- A execução de cada operação produz um resultado temporário
- Alternativa
 - Evitar ao máximo resultados temporários
 - Pipelining
 - concatena operações
 - conforme uma saída é produzida gera entrada para a operação subsequente

Pipelining Pattern Pipe & Filter



exemplo: Java Writer



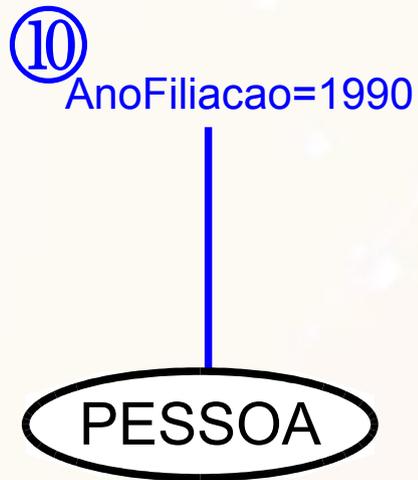
Exemplo de Pipeline

PESSOA

<u>Codigo</u>	Nome	Telefone	AnoFiliacao
1525	Asdrúbal	5432-1098	1990
1637	Doriana	9876-5432	1983
1701	Quincas	8765-4321	1985
2042	Melissa	7654-3210	1990
2111	Horácio	6543-2109	1983

PESSOA

Exemplo de Pipeline



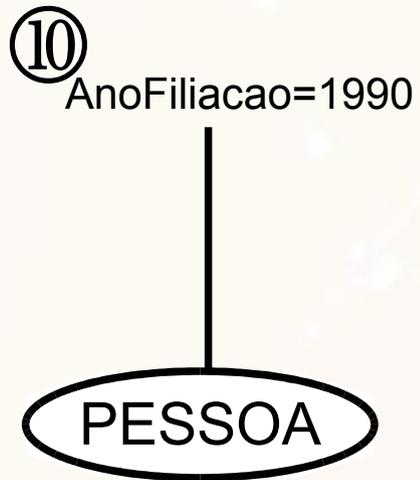
PESSOA

<u>Codigo</u>	Nome	Telefone	AnoFiliacao
1525	Asdrúbal	5432-1098	1990
1637	Doriana	9876-5432	1983
1701	Quincas	8765-4321	1985
2042	Melissa	7654-3210	1990
2111	Horácio	6543-2109	1983

Exemplo de Pipeline

PESSOA

<u>Codigo</u>	Nome	Telefone	AnoFiliacao
1525	Asdrúbal	5432-1098	1990
2042	Melissa	7654-3210	1990



Exemplo de Pipeline

⑦
Codigo, Nome



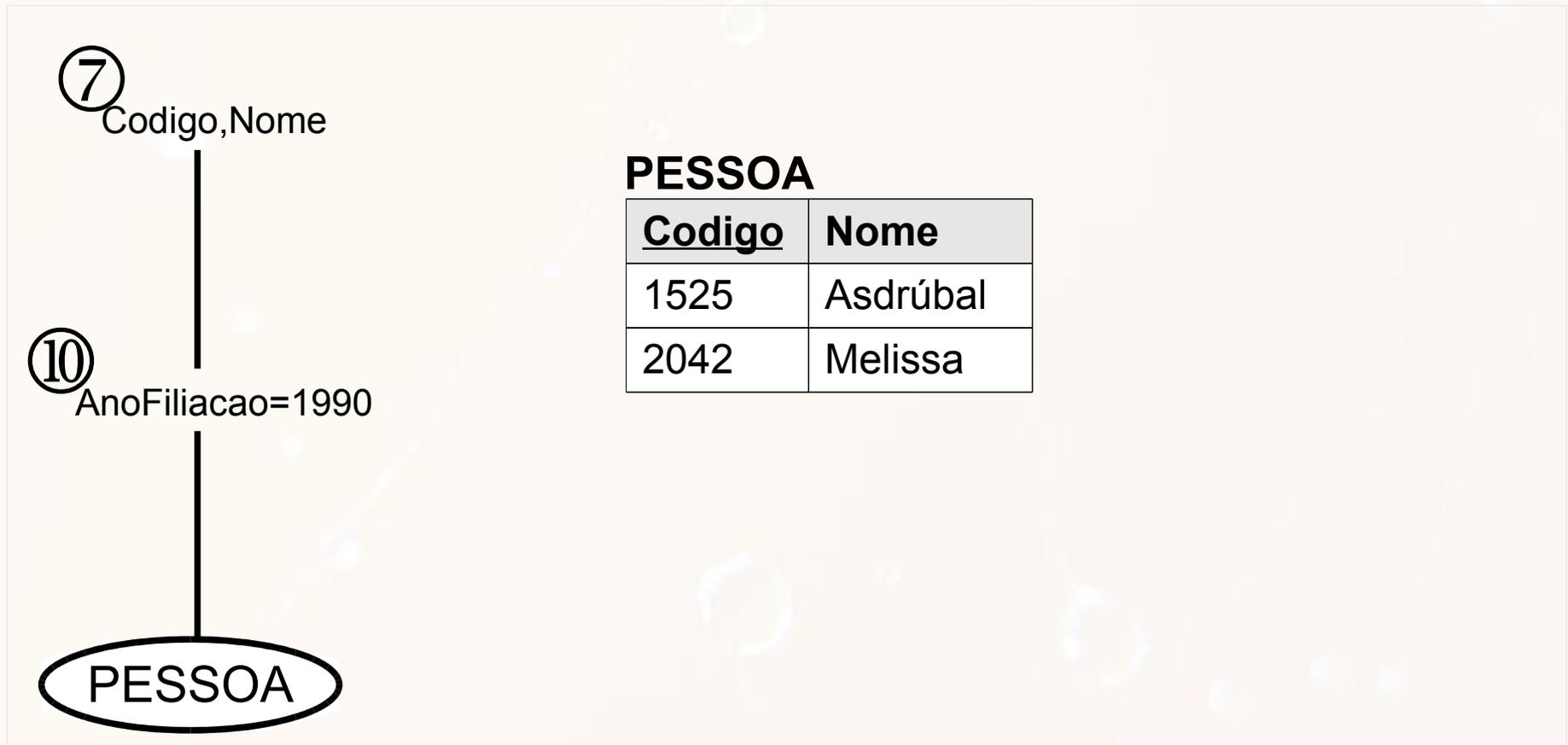
⑩
AnoFiliacao=1990



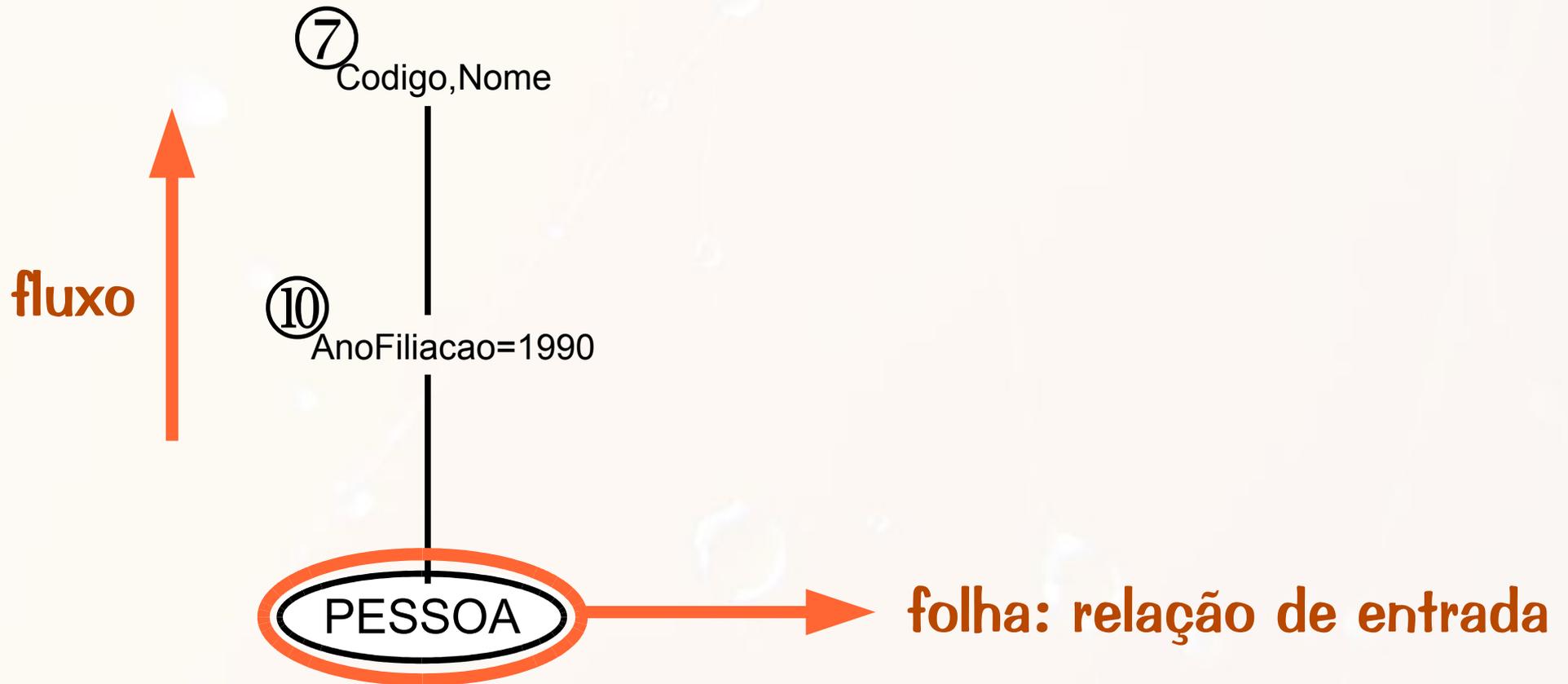
PESSOA

<u>Codigo</u>	Nome	Telefone	AnoFiliacao
1525	Asdrúbal	5432-1098	1990
2042	Melissa	7654-3210	1990

Exemplo de Pipeline



Árvore de Consulta



Heurísticas para Otimização de Consulta (Elmasri, 2011)

Heurísticas para Otimização de Consulta

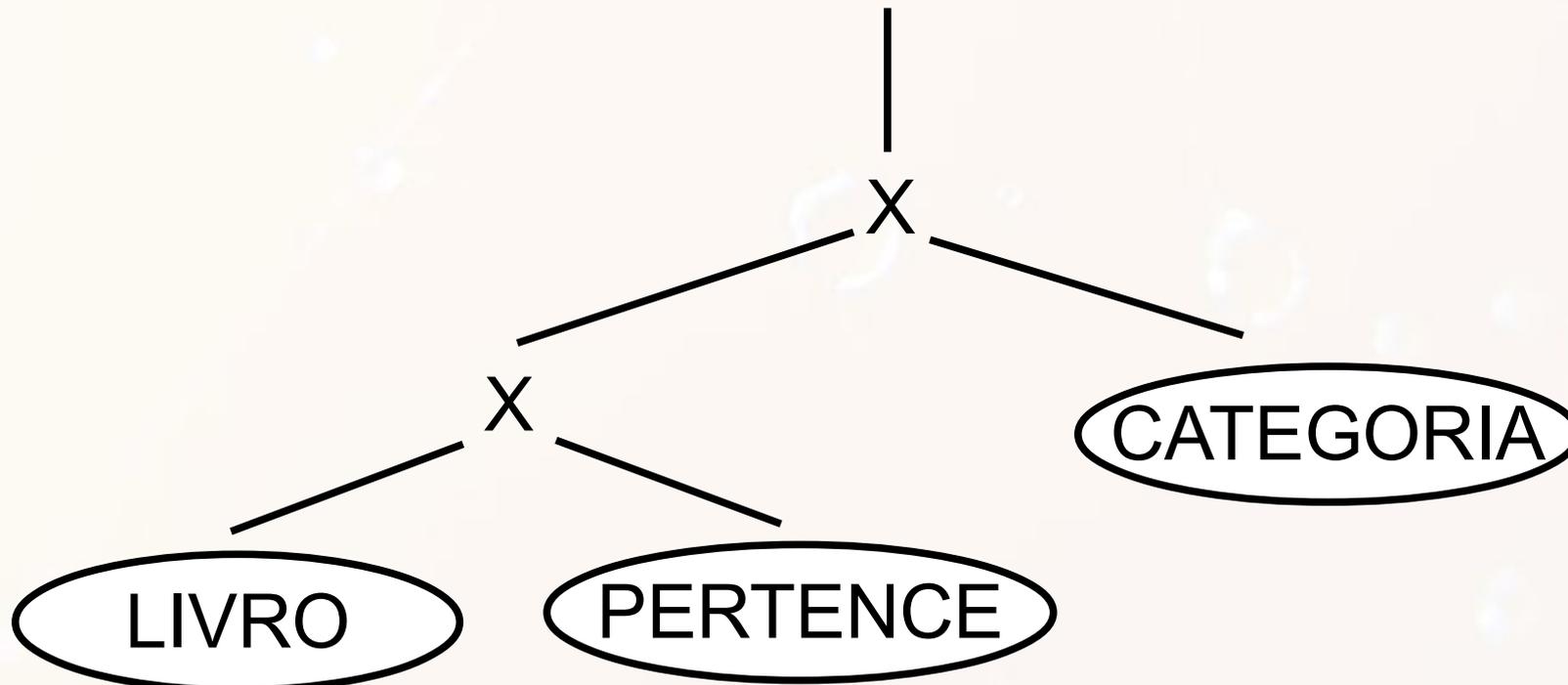
- Título dos livros sobre poesia escritos depois de 1996

```
SELECT LIVRO.Titulo
FROM LIVRO, PERTENCE, CATEGORIA
WHERE CATEGORIA.Nome = "poesia" AND
LIVRO.ISBN = PERTENCE.ISBN AND
CATEGORIA.Codigo = PERTENCE.CodCategoria AND
LIVRO.Ano > 1996
```

Heurística para Otimização de Consulta

⑦ LIVRO.Titulo

⑩ CATEGORIA.Nome="poesia" **AND** LIVRO.ISBN=PERTENCE.ISBN **AND** CATEGORIA.Codigo=PERTENCE.CodCategoria **AND** LIVRO.Ano>1996



Regras de Transformação

1. Operações seleção conjuntivas podem se converter em cascatas de seleção
2. Operação de seleção é comutativa
3. Comutação de seleção com projeção
 - caso o resultado da projeção tenha atributos requeridos pela seleção

Regras de Transformação

4. Seleção e junção (ou produto cartesiano) são comutativas

- se atributos da seleção são de apenas uma das relações

5. Operações de união e interseção são comutativas

- diferença não é

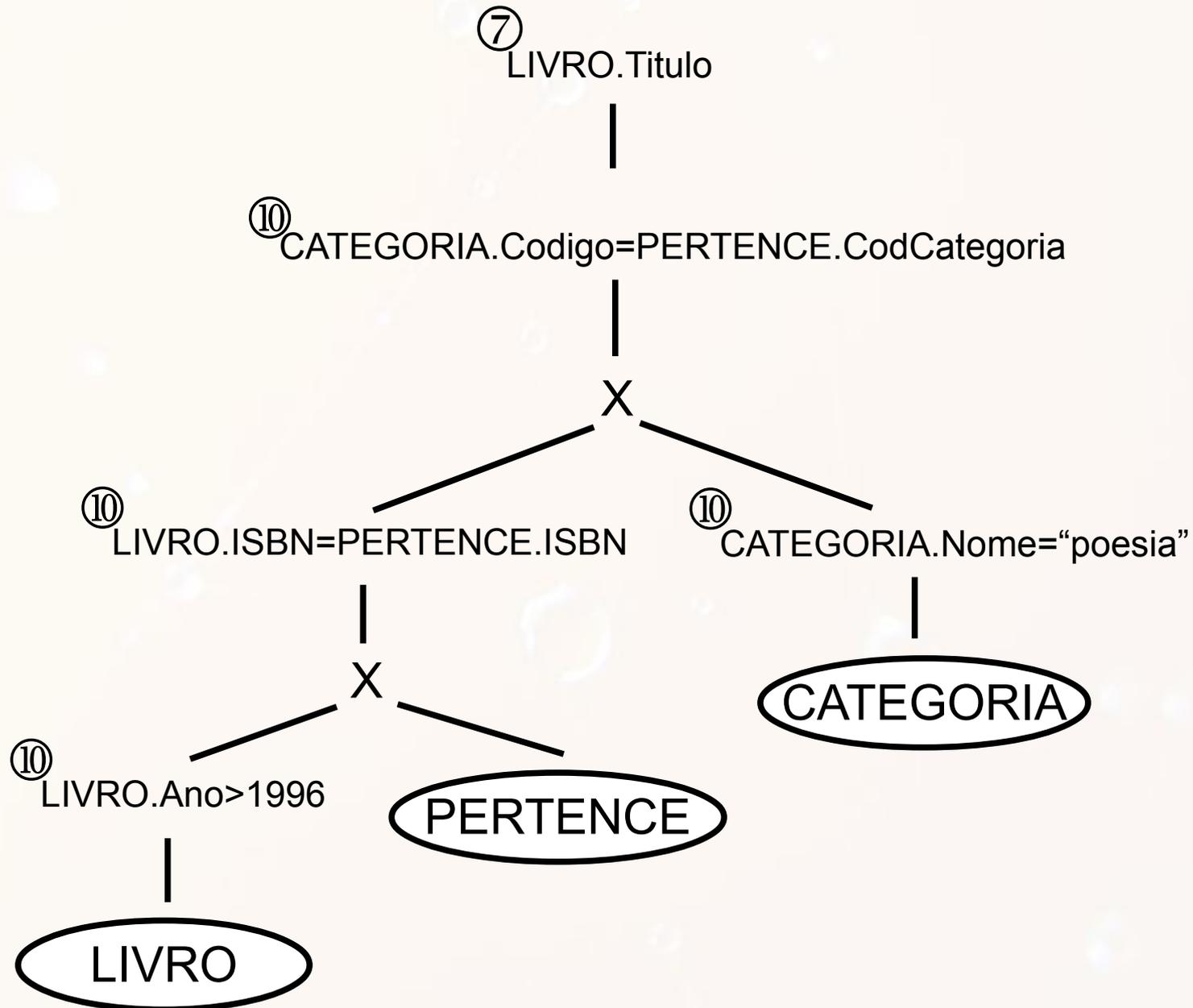
Regras de Transformação

6. Seleção é comutativa com operações de conjunto (união, interseção e diferença)
- $\text{sel}(A @ B)$ equivale $\text{sel}(A) @ \text{sel}(B)$

Heurísticas

- Quebrar operações de seleção conjuntivas (1)
 - maior liberdade
- Mover seleção em direção às folhas (2), (3), (4), (5) e (6)
 - apenas 1 tabela → acima da tabela
 - duas tabelas → acima da junção

Quebrando e Descendo Seleções



Regras de Transformação

7. As operações de junção e produto cartesiano são comutativas
8. As operações de junção, produto cartesiano, união e interseção são associativas

Heurística

- Operações de seleção mais restritivas devem ser executadas primeiro (5) e (6)

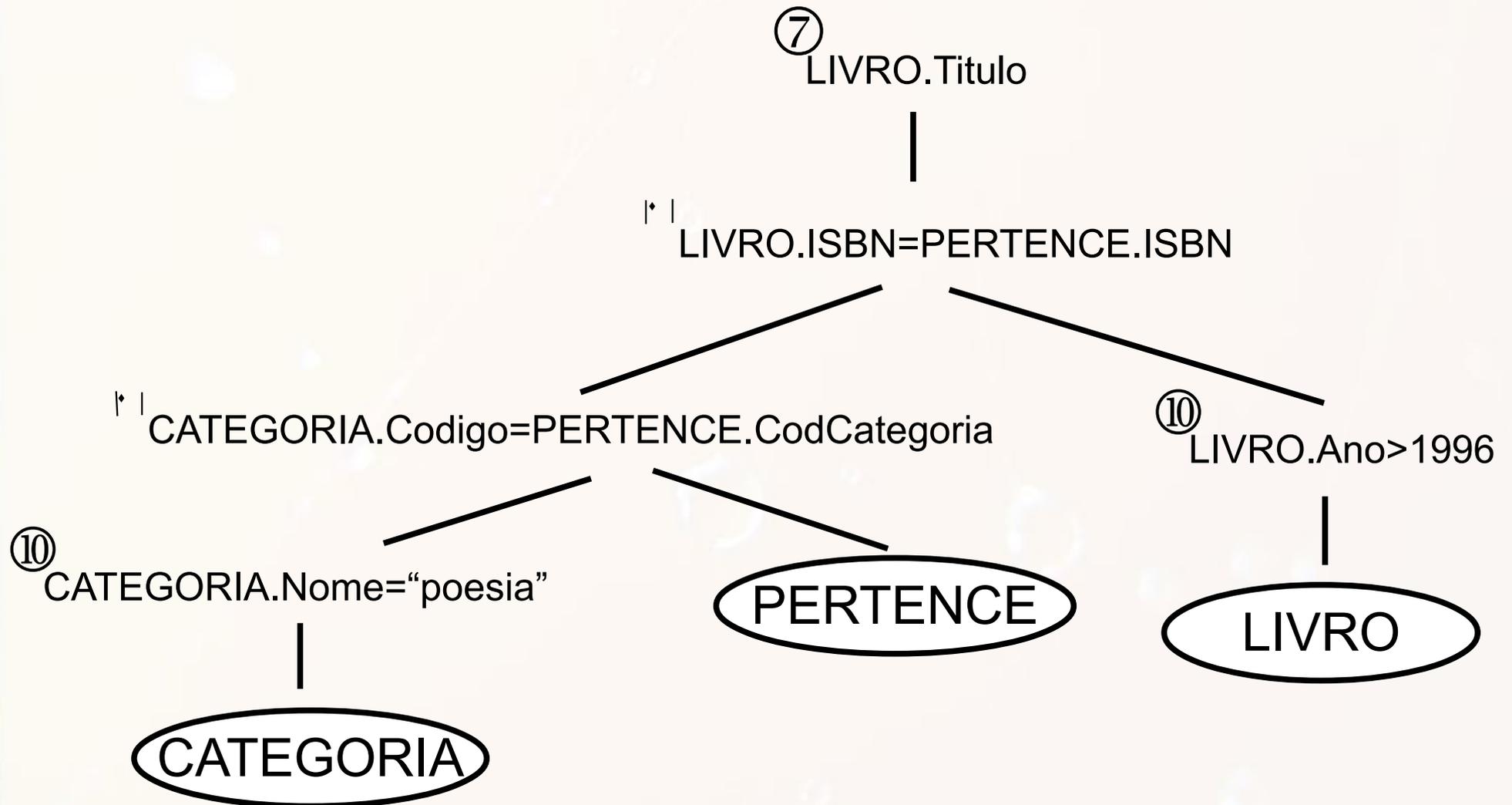
Regra de Transformação

9. Operações de produto cartesiano + seleção podem se converter em junção

Heurística

- Converta produtos cartesianos + seleções em junções

Produto Cartesiano + Seleção = Junção



Regras de Transformação

10. Cascata de projeções podem ser ignoradas e convertidas na última

- $\text{Pr1}(\text{Pr2}(\text{Pr3}(A)))$ equivale $\text{Pr1}(A)$

11. Operações de projeção e união são comutativas

- $\text{proj}(A \cup B)$ equivale $\text{proj}(A) \cup \text{proj}(B)$

Regras de Transformação

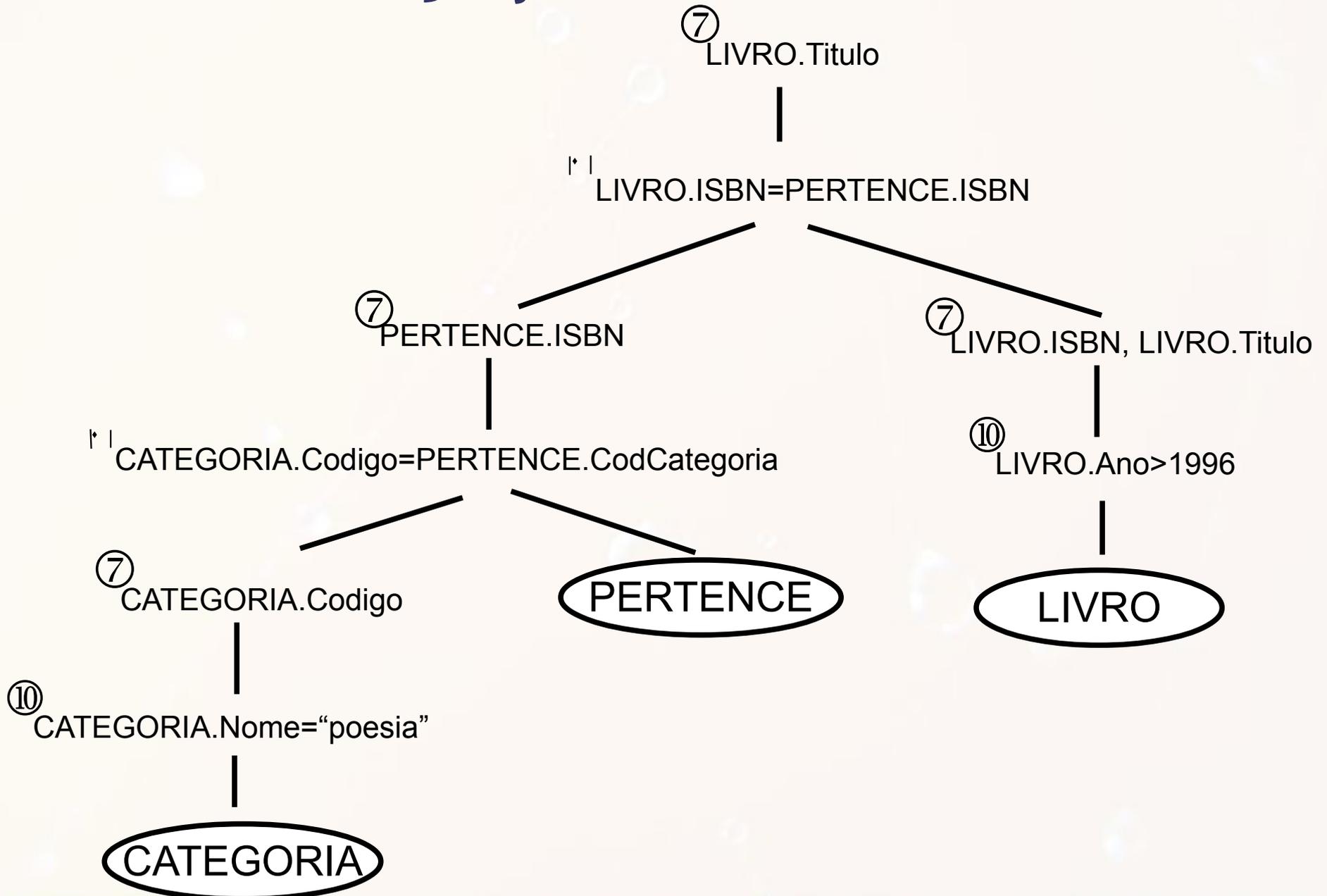
12. Operação de projeção pode ser comutada com junção (ou produto cartesiano)

- Relação A \rightarrow atributos a_1, \dots, a_n
- Relação B \rightarrow atributos b_1, \dots, b_m
- $L = (a_1, \dots, a_n, b_1, \dots, b_m)$
- Condição só contém atributos L
- $\text{proj}_L(A \text{ junção } B)$ equivale $(\text{proj}_{a_1, \dots, a_n}(A)) \text{ junção } (\text{proj}_{b_1, \dots, b_m}(B))$

Heurística

- Baseados em (10), (11) e (12)
 - Desmembrar operações de projeção
 - Mover projeções em direção às folhas
 - Criar operações de projeção para manter apenas atributos necessários

Projeções Mais Cedo



Heurística

- Identificar subárvores com operações a ser combinadas em um algoritmo

Referências

- Elmasri, Ramez; Navathe, Shamkant B. (2005) **Sistemas de Bancos de Dados**. Addison-Wesley, 4^a edição em português.
- Elmasri, Ramez; Navathe, Shamkant B. (2011) **Sistemas de Bancos de Dados**. Addison-Wesley, 6^a edição em português.
- Ramakrishnan, Raghu; Gehrke, Johannes (2003) **Database Management Systems**. McGraw-Hill, 3rd edition.

André Santanchè

<http://www.ic.unicamp.br/~santanche>

Licença

- Estes slides são concedidos sob uma Licença Creative Commons. Sob as seguintes condições: Atribuição, Uso Não-Comercial e Compartilhamento pela mesma Licença.
- Mais detalhes sobre a referida licença Creative Commons veja no link:
<http://creativecommons.org/licenses/by-nc-sa/3.0/>
- Fotografia da capa e fundo por
<http://www.flickr.com/photos/fdecomite/>
Ver licença específica em
<http://www.flickr.com/photos/fdecomite/1457493536/>

Processamento de Consultas

Banco de Dados: Teoria e Prática

André Santanchè

Instituto de Computação - UNICAMP

Outubro 2012

Picture by <http://www.flickr.com/photos/fdecomite/1457493536/>

Execução de Consulta

–

Passos Típicos (Elmasri, 2010)

Consulta em linguagem de alto nível

Análise Léxica,
Análise Sintática
e Validação

Forma intermediária de consulta

Otimizador
de Consulta

Plano de execução

Gerador de código
de consulta

Código para executar a consulta

Processador em
tempo de execução
do banco de dados

Resultado da consulta

Figure 18.1 Typical steps when processing a high-level query.

Query in a high-level language

SCANNING,
PARSING, AND
VALIDATING

Intermediate form of query

QUERY OPTIMIZER

Execution plan

QUERY CODE
GENERATOR

Code to execute the query

RUNTIME DATABASE
PROCESSOR

Result of query

Code can be:

- Executed directly (interpreted mode)
- Stored and executed later whenever needed (compiled mode)

Execução de Consulta

Passos Típicos

Consulta em linguagem de alto nível

Análise Léxica,
Análise Sintática
e Validação

Forma intermediária de consulta

Otimizador
de Consulta

Plano de execução

Gerador de código
de consulta

Código para executar a consulta

Processador em
tempo de execução
do banco de dados

Resultado da consulta

Figure 18.1 Typical steps when processing a high-level query.

Query in a high-level language

SCANNING,
PARSING, AND
VALIDATING

Intermediate form of query

QUERY OPTIMIZER

Execution plan

QUERY CODE
GENERATOR

Code to execute the query

RUNTIME DATABASE
PROCESSOR

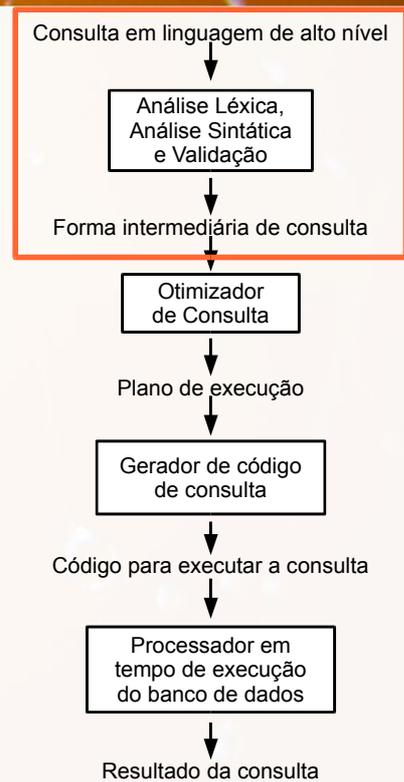
Result of query

Code can be:

- Executed directly (interpreted mode)
- Stored and executed later whenever needed (compiled mode)

Análise e Validação

- Análise e Validação
 - Análise léxica
 - Análise sintática
 - Validação
- Representações internas:
 - árvore de consulta
 - grafo de consulta



“The scanner identifies the language tokens – such as SQL keywords, attribute names, and relation names – in the text of the query, whereas the parser checks the query syntax to determine whether it is formulated according to the syntax rules (rules of grammar) of the query language. The query must also be validated, by checking that all attribute and relation names are valid and semantically meaningful names in the schema of the particular database being queried. An internal representation of the query is then created, usually as a tree data structure called a query tree. It is also possible to represent the query using a graph data structure called a query graph. The DBMS must then devise an execution strategy for retrieving the result of the query from the database files. A query typically has many possible execution strategies, and the process of choosing a suitable one for processing a query is known as query optimization.” (Elmasri, 2005, p. 493)

Estratégia de Execução

- Consulta possui muitas estratégias de execução possíveis
- Planejamento da Estratégia de Execução
 - Otimização → processo de escolha da estratégia adequada (razoavelmente eficiente)

Consulta em linguagem de alto nível

Análise Léxica,
Análise Sintática
e Validação

Forma intermediária de consulta

Otimizador
de Consulta

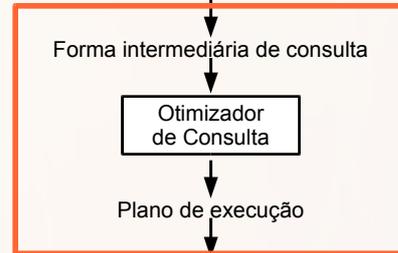
Plano de execução

Gerador de código
de consulta

Código para executar a consulta

Processador em
tempo de execução
do banco de dados

Resultado da consulta



Código da Consulta

- Pode ser:
 - Executado diretamente
 - modo interpretado
 - Armazenado e executado quando necessário
 - modo compilado

Consulta em linguagem de alto nível

Análise Léxica,
Análise Sintática
e Validação

Forma intermediária de consulta

Otimizador
de Consulta

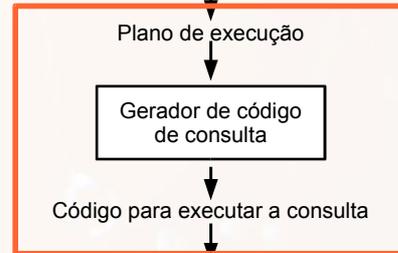
Plano de execução

Gerador de código
de consulta

Código para executar a consulta

Processador em
tempo de execução
do banco de dados

Resultado da consulta



Execução do Código

- Processador executa código da consulta
- Produz resultado da execução

Consulta em linguagem de alto nível

Análise Léxica,
Análise Sintática
e Validação

Forma intermediária de consulta

Otimizador
de Consulta

Plano de execução

Gerador de código
de consulta

Código para executar a consulta

Processador em
tempo de execução
do banco de dados

Resultado da consulta



Ênfase desta aula:
Otimização de Consultas

Consultas Declarativas

- “O quê” ao invés de “Como”
- Otimização de consulta
 - Solução razoavelmente eficiente (Elmasri, 2011)
 - Solução ótima pode ser muito custosa

Consulta SQL em Álgebra Relacional

- Consulta SQL → Álgebra Relacional Estendida
 - Inclui operadores como COUNT, SUM e MAX
- Consulta SQL decomposta em blocos
 - Bloco de Consulta ou Bloco Simples:
 - Contém uma única expressão SELECT-FROM-WHERE (GROUP BY e HAVING se houver)
 - Sem aninhamento
 - Consultas aninhadas são identificadas como consultas independentes

“15.1.1 Decomposition of a Query into Blocks

When a user submits an SQL query, the query is parsed into a collection of query blocks and then passed on to the query optimizer. A query block (or simply block) is an SQL query with no nesting and exactly one SELECT clause and one FROM clause and at most one WHERE clause, GROUP BY clause, and HAVING clause.”
(Ramakrishnan, 2003, p. 480)

“Typically, SQL queries are decomposed into query blocks, which form the basic units that can be translated into the algebraic operators and optimized. A query block contains a single SELECT-FROM-WHERE expression, as well as GROUP BY and HAVING clauses if these are part of the block. Hence, nested queries within a query are identified as separate query blocks.” (Elmasri, 2005, p. 495)

Decomposição em Blocos Exemplo

- Tabela

Pessoa(Codigo, Nome, Telefone, AnoFiliacao)

- Nome dos filiados mais antigos:

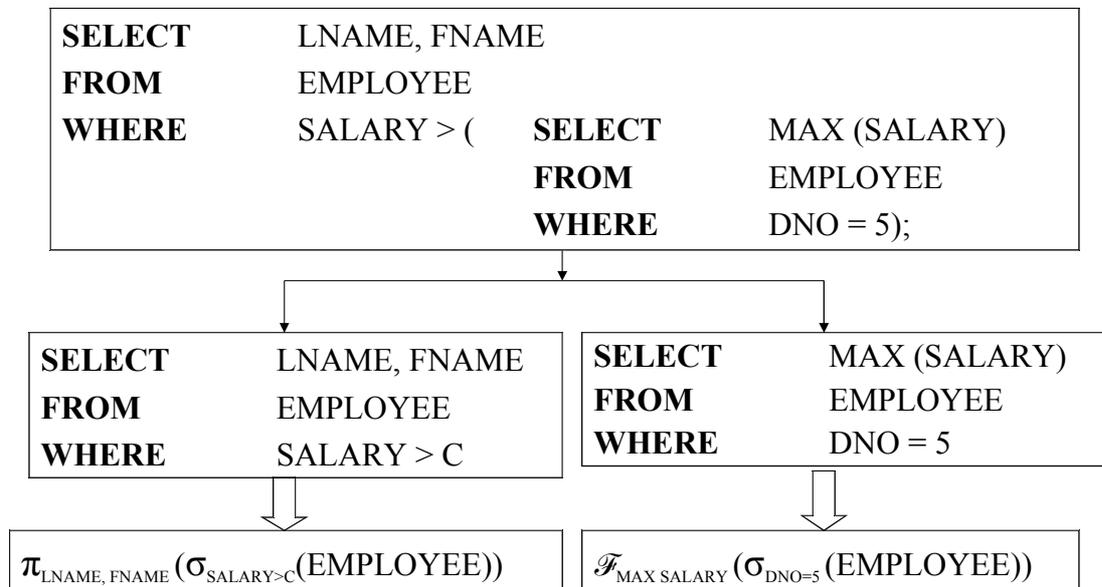
```
SELECT Codigo, Nome
FROM PESSOA
WHERE AnoFiliacao = (SELECT MIN(AnoFiliacao)
                     FROM PESSOA)
```

- Blocos

```
❶ SELECT Codigo, Nome
   FROM PESSOA
   WHERE AnoFiliacao = (referência ❷)

❷ SELECT MIN(AnoFiliacao)
   FROM PESSOA
```

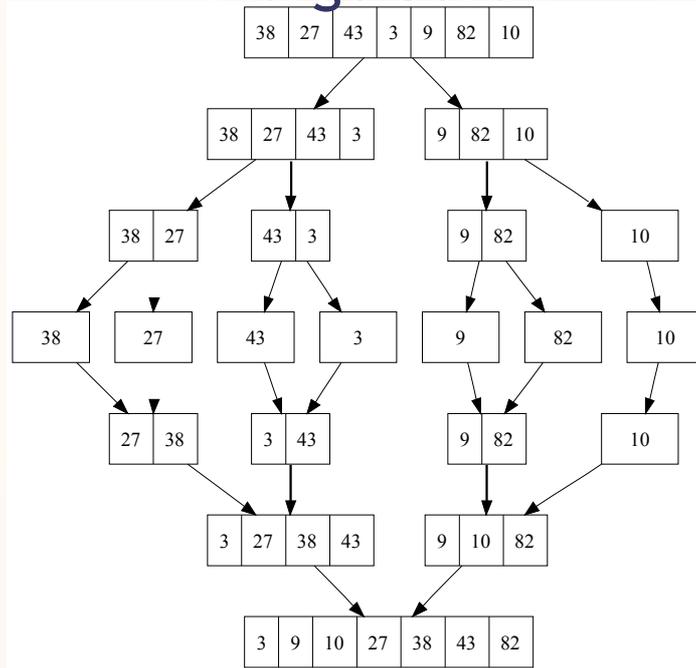
(Elmasri, 2007, s. 15-8)



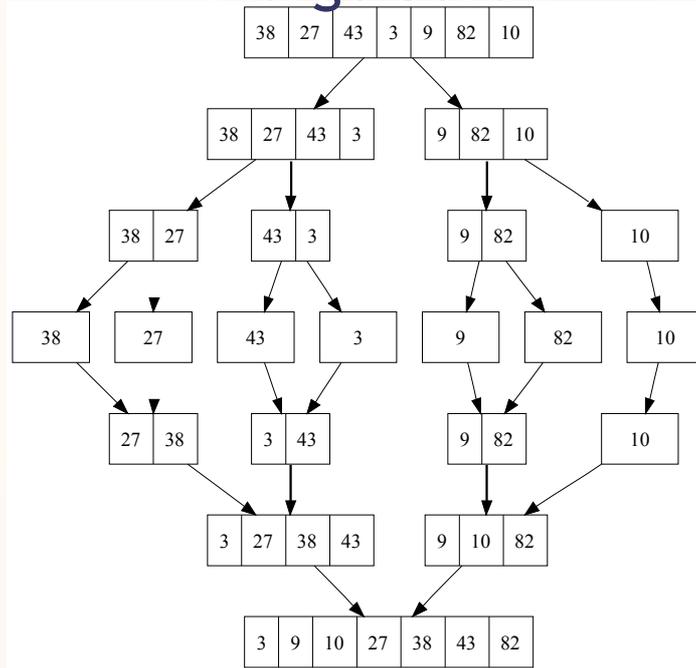
Algoritmos para Operações

Ordenação Externa

Merge Sort

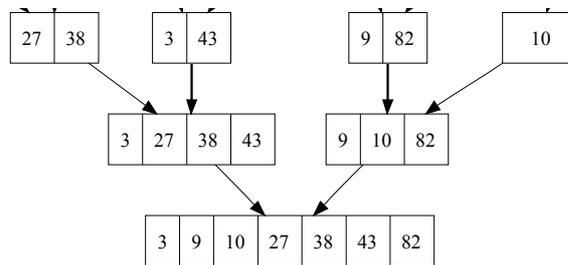


Merge Sort



Ordenação Externa

38	27	43	3	9	82	10
----	----	----	---	---	----	----

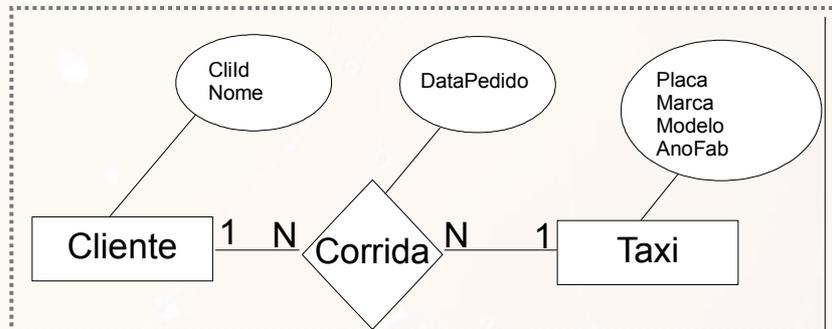


Ordenação Externa Números

- bd - blocos em disco
- bm - blocos de memória
- Ordenação
 - $2 \cdot bd$ transferências (leitura e gravação)
- Merge
 - $2 \cdot bd$ transferências a cada estágio
 - $\lceil bd/bm \rceil$ rodadas de ordenação
 - $\lceil \log_{bm-1}(bd/bm) \rceil$ níveis
- Custo: $(2 \cdot bd) + (2 \cdot bd \cdot \lceil \log_{bm-1}(bd/bm) \rceil)$

Seleção

Esquema Conceitual - Exemplo Táxis



Este é um subconjunto do Estudo de Caso proposto “Despacho e controle de Táxis via terminais móveis ligados on-line com um sistema multi-usuário” por prof. Geovane Cayres Magalhães

Tabelas para exemplo - Táxis

Táxi (TX)

<u>Placa</u>	<u>Marca</u>	<u>Modelo</u>	<u>AnoFab</u>
DAE6534	Ford	Fiesta	1999
DKL4598	Wolksvagen	Gol	2001
DKL7878	Ford	Fiesta	2001
JDM8776	Wolksvagen	Santana	2002
JJM3692	Chevrolet	Corsa	1999



Corrida (R1)

<u>ClId</u>	<u>Placa</u>	<u>DataPedido</u>
1755	DAE6534	15/02/2003
1982	JDM8776	18/02/2003



Seleção?

⑩ Placa='JDM8776'(TX)

Placa	Marca	Modelo	AnoFab
DAE6534	Ford	Fiesta	1999
DKL4598	Wolkswagen	Gol	2001
DKL7878	Ford	Fiesta	2001
JDM8776	Wolkswagen	Santana	2002
JJM3692	Chevrolet	Corsa	1999

Exatamente Igual Chave Primária

⑩ Placa='JDM8776'(TX)

Placa	Marca	Modelo	AnoFab
DAE6534	Ford	Fiesta	1999
DKL4598	Wolkswagen	Gol	2001
DKL7878	Ford	Fiesta	2001
JDM8776	Wolkswagen	Santana	2002
JJM3692	Chevrolet	Corsa	1999

Exatamente Igual Outra Chave

⑩ AnoFab=2002(TX)

Placa	Marca	Modelo	AnoFab
DAE6534	Ford	Fiesta	1999
DKL4598	Wolkswagen	Gol	2001
DKL7878	Ford	Fiesta	2001
JDM8776	Wolkswagen	Santana	2002
JJM3692	Chevrolet	Corsa	1999

Seleção?

⑩ AnoFab=2002(TX)

Placa	Marca	Modelo	AnoFab
DAE6534	Ford	Fiesta	1999
DKL4598	Wolkswagen	Gol	2001
DKL7878	Ford	Fiesta	2001
JDM8776	Wolkswagen	Santana	2002
JJM3692	Chevrolet	Corsa	1999

Seleção?

⑩ AnoFab > 2000 (TX)

Placa	Marca	Modelo	AnoFab
DAE6534	Ford	Fiesta	1999
DKL4598	Wolksvagen	Gol	2001
DKL7878	Ford	Fiesta	2001
JDM8776	Wolksvagen	Santana	2002
JJM3692	Chevrolet	Corsa	1999

Faixa (>, <, >=, <=)

⑩ AnoFab>2000(TX)

Placa	Marca	Modelo	AnoFab
DAE6534	Ford	Fiesta	1999
DKL4598	Wolksvagen	Gol	2001
DKL7878	Ford	Fiesta	2001
JDM8776	Wolksvagen	Santana	2002
JJM3692	Chevrolet	Corsa	1999

Algoritmos de Seleção

- Exatamente igual
 - chave primária
 - outra chave
- $>$, $<$, \geq , \leq
- compostos

Algoritmos de Seleção

- Pesquisa linear
- Pesquisa binária
- Usando índice primário
- Usando chave hash
- Combinado com o índice primário
- Usando índice de agrupamento
- Usando índice secundário

Seleção Conjuntiva x Dijuntiva

- seleção conjuntiva - e.g., and
- seleção dijuntiva - e.g., or

Algoritmos de Seleção Conjuntiva

- Índice para uma das condições
- Índice composto envolvendo ambas as condições
- Índice individual para cada condição

Seletividade

- seletividade: valor entre 0 e 1
- n registros
- igualdade atributo único
 - seletividade: $1/n$

Seletividade Atributo Não Único

- i valores
- i igualmente distribuído
- registros por valor?
- seletividade?

Seletividade Atributo Não Único

- i valores
- i igualmente distribuído
- n/i registros por valor
- seletividade: $1/i$

Seletividade Atributo Não Único

- primeiro as condições com valor menor de seletividade

Junção (Join)

Junção (Join) de Loop Aninhado

```
for each ti
  for each tj
    if match(ti, tj)
      add-result(ti, tj)
```

Junção de Loop Aninhado Números

- n_i - número de tuplas t_i
- n_j - número de tuplas t_j
- pares de tuplas? (comparações?)

Junção de Loop Aninhado Números

- n_i - número de tuplas t_i
- n_j - número de tuplas t_j
- $n_i * n_j$ - pares de tuplas

Junção de Loop Aninhado Números

- n_i - número de tuplas t_i
- n_j - número de tuplas t_j
- $n_i * n_j$ - pares de tuplas

- b_i - bloco de tuplas t_i
- b_j - bloco de tuplas t_j
- leituras de blocos?

Junção de Loop Aninhado Números

- n_i - número de tuplas t_i
- n_j - número de tuplas t_j
- $n_i * n_j$ - pares de tuplas

- b_i - bloco de tuplas t_i
- b_j - bloco de tuplas t_j
- $b_i + b_j * n_i$ leituras de blocos

Junção de Loop Aninhado Números

- Situações:
 - Quantas transferências de bloco se todos os blocos estiverem na memória?
 - Quantas transferências se os blocos de um dos loops estiver todo na memória e qual deles escolher (b_i ou b_j)

Junção de Loop Aninhado Números

■ Situações:

- Quantas transferências de bloco se todos os blocos estiverem na memória?
 - $b_i + b_j$ transferências
- Quantas transferências se os blocos de um dos loops estiver todo na memória e qual deles escolher (b_i ou b_j)?
 - escolher b_j
 - $b_i + b_j$ transferências

Junção de Loop Aninhado em Bloco

```
for each bi
  for each bj
    for each ti
      for each tj
        if match(ti, tj)
          add-result(ti, tj)
```

Junção de Loop Aninhado em Bloco Números

- b_i - bloco de tuplas t_i
- b_j - bloco de tuplas t_j
- leituras de blocos?

Junção de Loop Aninhado em Bloco Números

- b_i - bloco de tuplas t_i
- b_j - bloco de tuplas t_j
- $b_i + b_j * b_i$ leituras de blocos

Outras Junções

- Junção Indexada
- Junção Merge
- Junção Hash

Projeção

- Recorte dos campos
- (?)

Projeção

- Recorte dos campos
- Registros sem duplicatas
 - SQL → padrão não eliminar duplicatas
 - DISTINCT → elimina duplicatas
 - Registros com garantia de ser únicos
 - e.g., contendo chave primária
 - Registros sem garantia de ser únicos
 - ordenação
 - hashing

Otimização de Consulta

SQL p/ Álgebra

▪ Versão SQL

```
SELECT Codigo, Nome  
FROM PESSOA  
WHERE AnoFiliacao = 1990
```

▪ Versão em álgebra

⑦ Codigo, Nome (⑩ AnoFiliacao=1990 (PESSOA))

▪ Versão Árvore



Combinação de Operações usando Pipelining

- Uma consulta é mapeada em uma sequência de operações
- A execução de cada operação produz um resultado temporário
- Alternativa
 - Evitar ao máximo resultados temporários
 - Pipelining
 - concatena operações
 - conforme uma saída é produzida gera entrada para a operação subsequente

Combining Operations using Pipelining (1)

● **Motivation**

- A query is mapped into a sequence of operations.
- Each execution of an operation produces a temporary result.
- Generating and saving temporary files on disk is time consuming and expensive.

● **Alternative:**

- Avoid constructing temporary results as much as possible.
- Pipeline the data through multiple operations - pass the result of a previous operator to the next without waiting to complete the previous operation.

Exemplo de Pipeline

PESSOA

Codigo	Nome	Telefone	AnoFiliacao
1525	Asdrúbal	5432-1098	1990
1637	Doriana	9876-5432	1983
1701	Quincas	8765-4321	1985
2042	Melissa	7654-3210	1990
2111	Horácio	6543-2109	1983

PESSOA

Exemplo de Pipeline

⑩ AnoFiliacao=1990

PESSOA

Codigo	Nome	Telefone	AnoFiliacao
1525	Asdrúbal	5432-1098	1990
1637	Doriana	9876-5432	1983
1701	Quincas	8765-4321	1985
2042	Melissa	7654-3210	1990
2111	Horácio	6543-2109	1983

The diagram illustrates a data pipeline filter. A blue circle containing the number '10' is connected by a vertical line to the text 'AnoFiliacao=1990'. This text is further connected by another vertical line to an oval labeled 'PESSOA', which represents the table of data being filtered.

Exemplo de Pipeline

PESSOA

<u>Codigo</u>	Nome	Telefone	AnoFiliacao
1525	Asdrúbal	5432-1098	1990
2042	Melissa	7654-3210	1990

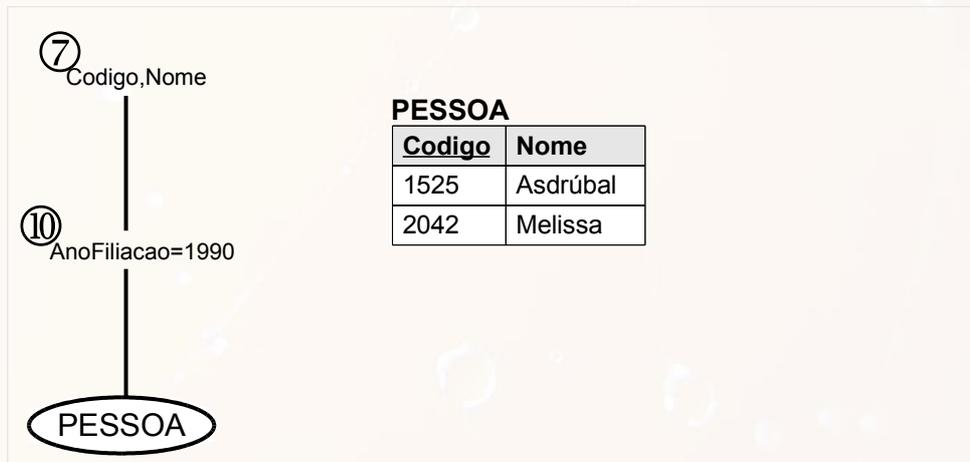
⑩
AnoFiliacao=1990



Exemplo de Pipeline



Exemplo de Pipeline



Árvore de Consulta



Heurísticas para Otimização de
Consulta
(Elmasri, 2011)

Heurísticas para Otimização de Consulta

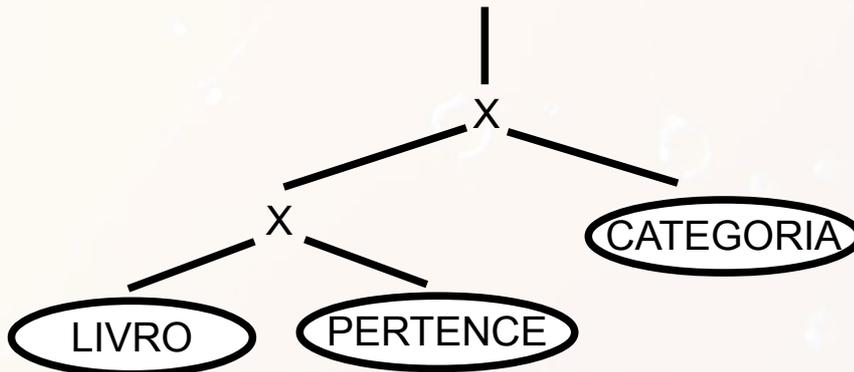
- Título dos livros sobre poesia escritos depois de 1996

```
SELECT LIVRO.Titulo
FROM LIVRO, PERTENCE, CATEGORIA
WHERE CATEGORIA.Nome = "poesia" AND
LIVRO.ISBN = PERTENCE.ISBN AND
CATEGORIA.Codigo = PERTENCE.CodCategoria AND
LIVRO.Ano > 1996
```

Heurística para Otimização de Consulta

⑦ LIVRO.Titulo

⑩ CATEGORIA.Nome="poesia" **AND** LIVRO.ISBN=PERTENCE.ISBN **AND**
CATEGORIA.Codigo=PERTENCE.CodCategoria **AND** LIVRO.Ano>1996



Regras de Transformação

1. Operações seleção conjuntivas podem se converter em cascatas de seleção
2. Operação de seleção é comutativa
3. Comutação de seleção com projeção
 - caso o resultado da projeção tenha atributos requeridos pela seleção

Regras de Transformação

4. Seleção e junção (ou produto cartesiano) são comutativas

- se atributos da seleção são de apenas uma das relações

5. Operações de união e interseção são comutativas

- diferença não é

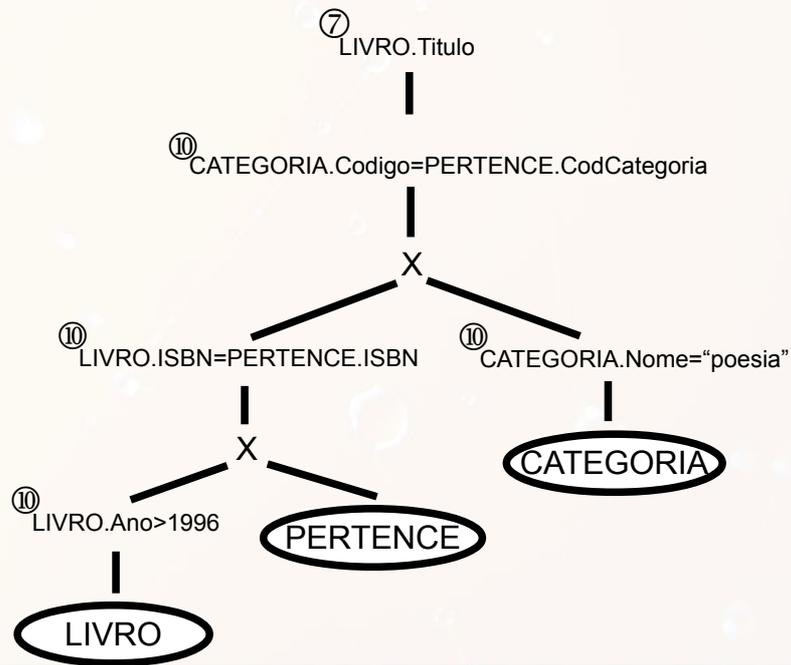
Regras de Transformação

6. Seleção é comutativa com operações de conjunto (união, interseção e diferença)
- $\text{sel}(A @ B)$ equivale $\text{sel}(A) @ \text{sel}(B)$

Heurísticas

- Quebrar operações de seleção conjuntivas (1)
 - maior liberdade
- Mover seleção em direção às folhas (2), (3), (4), (5) e (6)
 - apenas 1 tabela → acima da tabela
 - duas tabelas → acima da junção

Quebrando e Descendo Seleções



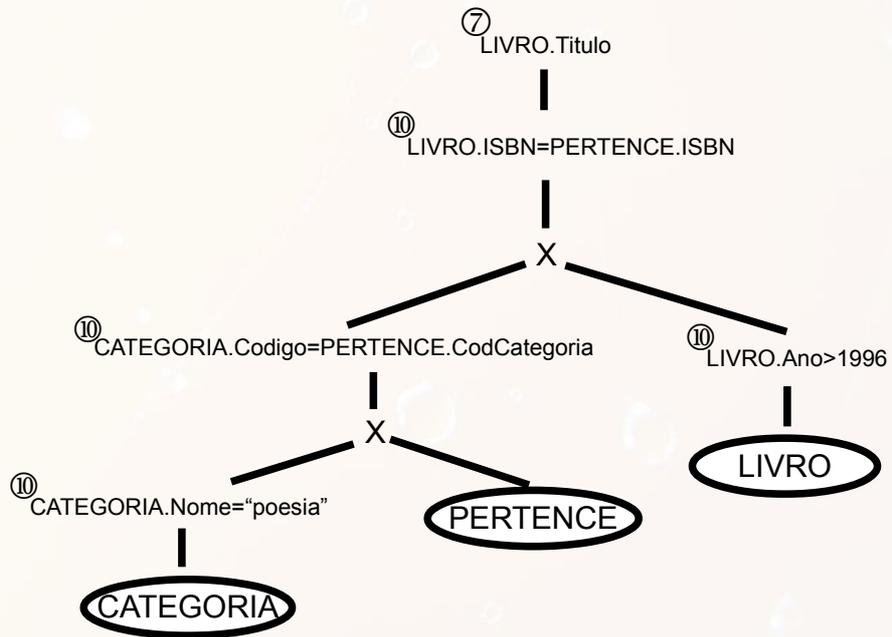
Regras de Transformação

7. As operações de junção e produto cartesiano são comutativas
8. As operações de junção, produto cartesiano, união e interseção são associativas

Heurística

- Operações de seleção mais restritivas devem ser executadas primeiro (5) e (6)

Troca de Categoria com Livro



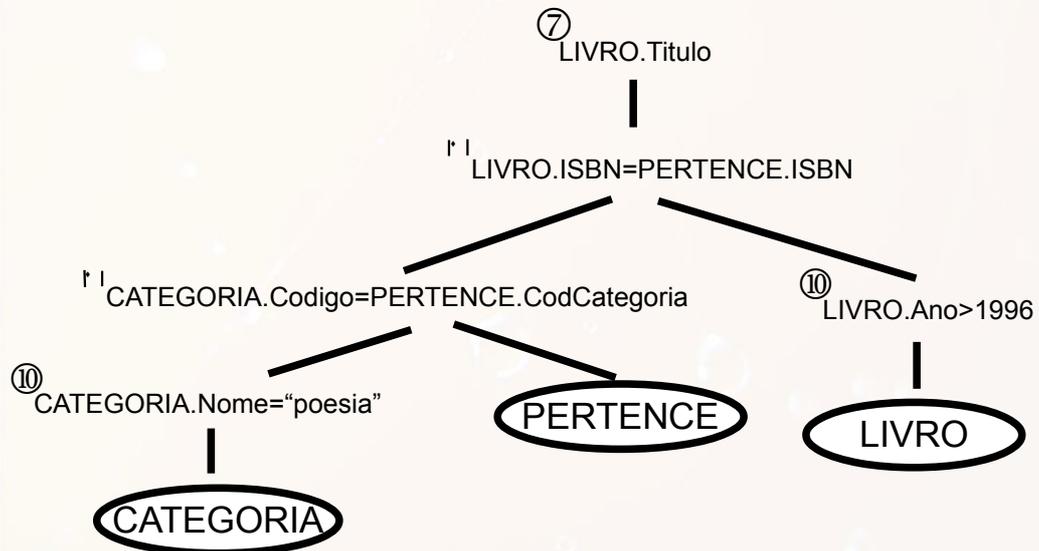
Regra de Transformação

9. Operações de produto cartesiano + seleção podem se converter em junção

Heurística

- Converta produtos cartesianos + seleções em junções

Produto Cartesiano + Seleção = Junção



Regras de Transformação

10. Cascata de projeções podem ser ignoradas e convertidas na última

- $\text{Pr1}(\text{Pr2}(\text{Pr3}(A)))$ equivale $\text{Pr1}(A)$

11. Operações de projeção e união são comutativas

- $\text{proj}(A \cup B)$ equivale $\text{proj}(A) \cup \text{proj}(B)$

Regras de Transformação

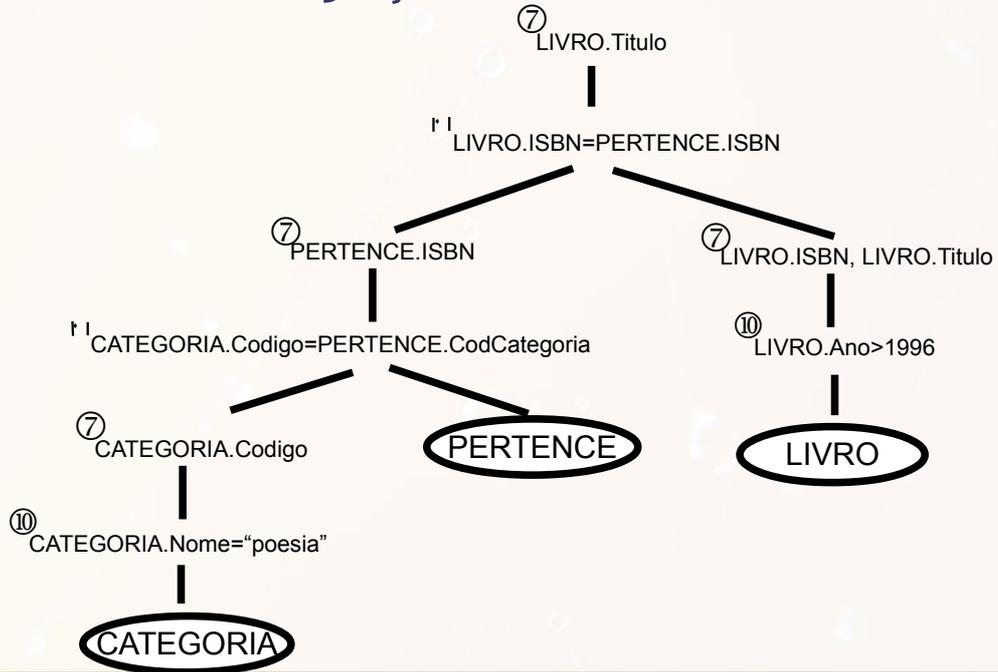
12. Operação de projeção pode ser comutada com junção (ou produto cartesiano)

- Relação A → atributos a_1, \dots, a_n
- Relação B → atributos b_1, \dots, b_m
- $L = (a_1, \dots, a_n, b_1, \dots, b_m)$
- Condição só contém atributos L
- $\text{proj}_L(A \text{ junção } B)$ equivale $(\text{proj}_{a_1, \dots, a_n}(A)) \text{ junção } (\text{proj}_{b_1, \dots, b_m}(B))$

Heurística

- Baseados em (10), (11) e (12)
 - Desmembrar operações de projeção
 - Mover projeções em direção às folhas
 - Criar operações de projeção para manter apenas atributos necessários

Projeções Mais Cedo



Heurística

- Identificar subárvores com operações a ser combinadas em um algoritmo

Referências

- Elmasri, Ramez; Navathe, Shamkant B. (2005) **Sistemas de Bancos de Dados**. Addison-Wesley, 4ª edição em português.
- Elmasri, Ramez; Navathe, Shamkant B. (2011) **Sistemas de Bancos de Dados**. Addison-Wesley, 6ª edição em português.
- Ramakrishnan, Raghu; Gehrke, Johannes (2003) **Database Management Systems**. McGraw-Hill, 3rd edition.

André Santanchè

<http://www.ic.unicamp.br/~santanche>

Licença

- Estes slides são concedidos sob uma Licença Creative Commons. Sob as seguintes condições: Atribuição, Uso Não-Comercial e Compartilhamento pela mesma Licença.
- Mais detalhes sobre a referida licença Creative Commons veja no link:
<http://creativecommons.org/licenses/by-nc-sa/3.0/>
- Fotografia da capa e fundo por
<http://www.flickr.com/photos/fdecomite/>
Ver licença específica em
<http://www.flickr.com/photos/fdecomite/1457493536/>