Processamento de Consultas em BD Distribuídos

IN1128/IF694 – Bancos de Dados Distribuídos e Móveis
Ana Carolina Salgado – acs@cin.ufpe.br
Bernadette Farias Lóscio – bfl@cin.ufpe.br

Centro de Informática — Universidade Federal de Pernambuco — CIn.ufpe.br

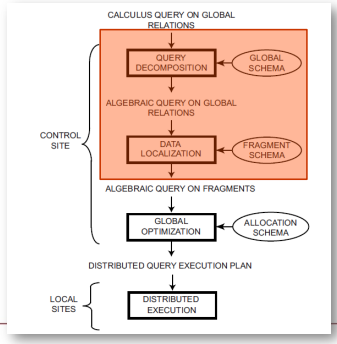
Decomposição de consultas e Localização de dados

Centro de Informática — Universidade Federal de Pernambuco — CIn.ufpe.br

Agenda

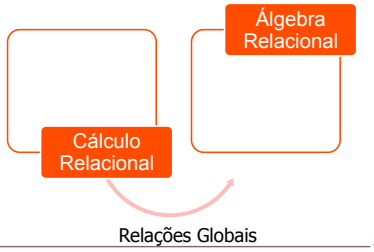
- **Introdução**
- **Decomposição de consultas**
  - Normalização
  - Análise
  - Eliminação da redundância
  - Reescrita
- **Localização de dados**
  - Fragmentação horizontal
  - Fragmentação vertical
  - Fragmentação derivada
  - Fragmentação híbrida
- **Conclusão**

Introdução



Introdução

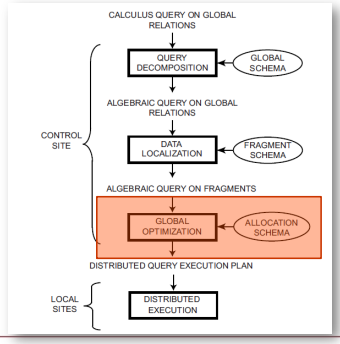
- **Decomposição de consultas**



Introdução

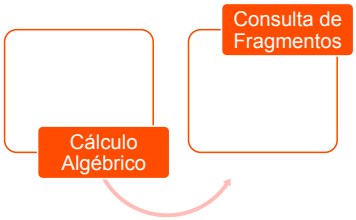
- **Técnicas semelhantes ao SGBD centralizado**
- **Resultado:**
  - Consulta algébrica BOA

E a consulta ÓTIMA?



Introdução

- **Localização de dados**



Esquema de Fragmentação

Decomposição de consultas

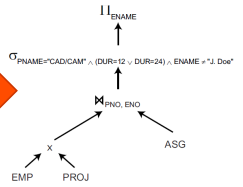
Universidade Federal de Pernambuco

CIn.ufpe.br

Decomposição de consultas

- **Relações globais**
- **SGBD == SGBDD**
- **Lógica clássica**

```
SELECT ENAME
FROM   PROJ, ASG, EMP
WHERE  ASG.ENO = EMP.ENO
AND    ASG.PNO = PROJ.PNO
AND    ENAME != "J. Doe"
AND    PROJ.PNAME = "CAD/CAM"
AND    (DUR = 12 OR DUR = 24)
```



Decomposição de consultas

- **Fases**
  - Normalização
  - Análise
  - Eliminação de redundâncias
  - Reescrita

Transformações Equivalentes

Normalização

- **Transformar a consulta em uma forma normalizada**
- **A parte mais importante é a transformação dos qualificadores (Predicados WHERE)**
- **Lógica clássica**
  - Forma Normal Conjuntiva

    $(p_{11} \vee p_{12} \vee \cdots \vee p_{1n}) \wedge \cdots \wedge (p_{m1} \vee p_{m2} \vee \cdots \vee p_{mn})$
  - Forma Normal Disjuntiva

    $(p_{11} \wedge p_{12} \wedge \cdots \wedge p_{1n}) \vee \cdots \vee (p_{m1} \wedge p_{m2} \wedge \cdots \wedge p_{mn})$

Normalização

- **"Encontre os nomes dos funcionários que estão trabalhando no projeto P1 por 12 ou 24 meses":**

```
SELECT ENAME
FROM   EMP, ASG
WHERE  EMP.ENO = ASG.ENO
AND    ASG.PNO = "P1"
AND    DUR = 12 OR DUR = 24
```

- **FNC**

  EMP.ENO = ASG.ENO ∧ ASG.PNO = "P1" ∧ (DUR = 12 ∨ DUR = 24)

- **FND**

  (EMP.ENO = ASG.ENO ∧ ASG.PNO = "P1" ∧ DUR = 12) ∨
  (EMP.ENO = ASG.ENO ∧ ASG.PNO = "P1" ∧ DUR = 24)

Análise

- **Detectar consultas que devem ser rejeitadas!**
- **Consulta de tipo incorreto**
  - Semelhante à Verificação de Tipos em LP

```
SELECT E#
FROM   EMP
WHERE  ENAME > 200
```

- **Consulta semanticamente incorreta**

Análise

- **Para detectar se uma consulta está correta semanticamente a consulta pode ser representada por meio de um grafo**
- **Grafo de consulta**
  - Operadores (Seleção, Projeção, Junção)
  - Nós: operandos
  - Nó especial: RESULT
  - Aresta: junção ou projeção
- **Grafo de junção**
  - Apenas as junções são consideradas
  - Útil na fase de otimização de consultas

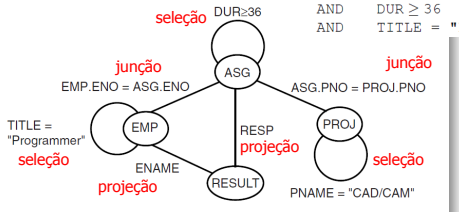
Análise

- **Exemplo:**
  - "Encontre os nomes e as responsabilidades dos programadores que trabalharam no projeto CAD/CAM por mais de três anos"

```
SELECT ENAME, RESP
FROM   EMP, ASG, PROJ
WHERE  EMP.ENO = ASG.ENO
AND    ASG.PNO = PROJ.PNO
AND    PNAME = "CAD/CAM"
AND    DUR ≥ 36
AND    TITLE = "Programmer"
```

Análise

```
SELECT ENAME, RESP
FROM   EMP, ASG, PROJ
WHERE  EMP.ENO = ASG.ENO
AND    ASG.PNO = PROJ.PNO
AND    PNAME = "CAD/CAM"
AND    DUR ≥ 36
AND    TITLE = "Programmer"
```



(a) Query graph

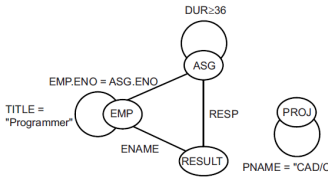
Análise

- **Pra que serve isso?**
  - Identificar a corretude semântica de uma consulta
- **Uma consulta é semanticamente incorreta se seu grafo de consulta não for conectado.**
- **Exemplo:**

```
SELECT ENAME, RESP
FROM   EMP, ASG, PROJ
WHERE  EMP.ENO = ASG.ENO
AND    PNAME = "CAD/CAM"
AND    DUR ≥ 36
AND    TITLE = "Programmer"
```

Análise

- **Soluções:**
  - Rejeitar a consulta
  - Supor um produto cartesiano implícito (ASG X PROJ)
  - Deduzir o predicado ASG.PNO = PROJ.PNO



Eliminação de redundância

- **Qualificação da consulta enriquecida pode conter predicados redundantes**
- **Redundância e duplicidade de trabalho podem ser eliminadas**

Eliminação de redundância

```
SELECT TITLE
FROM   EMP
WHERE  (NOT (TITLE = "Programmer")
AND    (TITLE = "Programmer"
OR     TITLE = "Elect. Eng.")
AND    NOT (TITLE = "Elect. Eng."))
OR     ENAME = "J. Doe"
```

```
SELECT TITLE
FROM EMP
WHERE ENAME = "J DOE"
```

1. $p \wedge p \Leftrightarrow p$
2. $p \vee p \Leftrightarrow p$
3. $p \wedge true \Leftrightarrow p$
4. $p \vee false \Leftrightarrow p$
5. $p \wedge false \Leftrightarrow false$
6. $p \vee true \Leftrightarrow true$
7. $p \wedge \neg p \Leftrightarrow false$
8. $p \vee \neg p \Leftrightarrow true$
9. $p_1 \wedge (p_1 \vee p_2) \Leftrightarrow p_1$
10. $p_1 \vee (p_1 \wedge p_2) \Leftrightarrow p_1$

Reescrita

- **Etapas**
  - Transformação direta da consulta de Cálculo Relacional para Álgebra Relacional
  - Reestruturação da consulta algébrica para melhorar o desempenho
- **A consulta em álgebra relacional é representada por uma árvore de operadores**
- **Árvore de Operadores**
  - Folhas = Relações
  - Não-folhas = relações intermediárias
  - Sentido: folhas até a raiz

Reescrita

- **Árvore de Operadores**
  - 1- Cada relação é uma folha
    - FROM
  - 2- Raiz é a operação de projeção
    - SELECT
  - 3- Qualificações são sequências das folhas até a raiz (operações relacionais)
    - WHERE

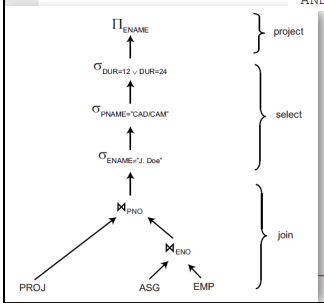
Reescrita

- **"Selecionar todos empregados que não J.Doe que trabalharam no projeto de CAD/CAM por 1 ou 2 anos"**

```
SELECT ENAME
FROM   PROJ, ASG, EMP
WHERE  ASG.ENO = EMP.ENO
AND    ASG.PNO = PROJ.PNO
AND    ENAME != "J. Doe"
AND    PROJ.PNAME = "CAD/CAM"
AND    (DUR = 12 OR DUR = 24)
```

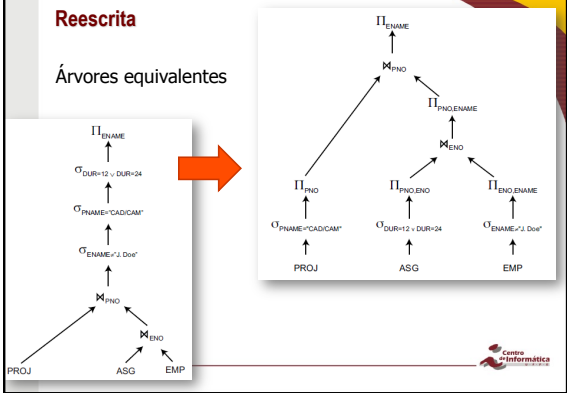
Reescrita

```
SELECT ENAME
FROM   PROJ, ASG, EMP
WHERE  ASG.ENO = EMP.ENO
AND    ASG.PNO = PROJ.PNO
AND    ENAME != "J. Doe"
AND    PROJ.PNAME = "CAD/CAM"
AND    (DUR = 12 OR DUR = 24)
```



Reescrita

Árvores equivalentes



Reescrita

- **Árvores equivalentes**
  - Fase de otimização: comparação de todas as árvores possíveis
  - # Árvores possíveis → ∞
  - A ideia consiste em aplicar as regras de transformação para que as "árvores ruins" sejam eliminadas
  - Vantagens:
    - 1. Separação de operações unárias (simplifica a consulta)
    - 2. Agrupamento de operações unárias (acesso à relação de uma vez)
    - 3. Comutação (permutação) de operações unárias e binárias (operações de seleção podem ser executadas primeiro)
    - 4. Ordenação das operações binárias (usada na otimização de consultas)

Localização de dados distribuídos

Universidade Federal de Pernambuco

cin.ufpe.br

Localização dos Dados Distribuídos

- **Localização de dados fragmentados entre as unidades de armazenamento do SGBDD.**
- **Converter uma consulta algébrica sobre relações globais em uma consulta algébrica expressa sobre fragmentos físicos.**

Localização dos Dados Distribuídos

- **Programa de localização**
  - É um programa em álgebra relacional cujos operandos são os fragmentos
  - obtido a partir das regras de reconstrução das consultas globais
- **Cada relação global é substituída pelo seu programa de localização**
- **A consulta resultante é chamada de "consulta localizada"**
- **A "consulta localizada" ainda pode ser simplificada e reestruturada por meio da aplicação de regras de redução!**

Redução da fragmentação horizontal primária

- **A função de fragmentação horizontal faz a distribuição de uma relação com base nos predicados de seleção (o operador de reconstrução é a união)**
- **As técnicas de redução para a fragmentação horizontal primária buscam determinar quais subárvores produzirão relações vazias para que sejam removidas**

Redução da fragmentação horizontal primária

- **Redução com seleção**
  - Dada uma relação R que foi fragmentada horizontalmente como R_1, R_2, ..., R_w, onde $R_j = sel_{pj}(R)$, a regra pode ser enunciada formalmente como:

$$\textbf{Rule 1: } \sigma_{p_i}(R_j) = \phi \text{ if } \forall x \text{ in } R : \neg(p_i(x) \wedge p_j(x))$$

  - onde pi e pj são predicados de seleção, x denota uma tupla e p(x) denota "predicado p(x) válido para x".

Um predicado da consulta é aplicado a um fragmento e retorna um conjunto de vazio!

Redução da fragmentação horizontal primária

Exemplo:

EMP(ENO, ENAME, TITLE)

$$EMP_1 = \sigma_{ENO \leq "E3"}(EMP)$$
$$EMP_2 = \sigma_{"E3" < ENO \leq "E6"}(EMP)$$
$$EMP_3 = \sigma_{ENO > "E6"}(EMP)$$

Fragmentação horizontal

$$EMP = EMP_1 \cup EMP_2 \cup EMP_3$$

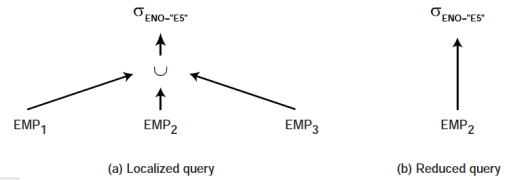
Programa de localização

Redução da fragmentação horizontal primária

Exemplo:

```
SELECT *
FROM   EMP
WHERE  ENO = "E5"
```

$$EMP_1 = \sigma_{ENO \leq "E3"}(EMP)$$
$$EMP_2 = \sigma_{"E3" < ENO \leq "E6"}(EMP)$$
$$EMP_3 = \sigma_{ENO > "E6"}(EMP)$$



(a) Localized query (b) Reduced query

Os predicados de EMP1 e EMP3 são contraditórios com o predicado da seleção da consulta, logo os fragmentados podem ser eliminados

Redução da fragmentação horizontal primária

- **Redução com junção**
  - Distribuir junções sobre uniões e eliminar junções inúteis.
  - Primeiro as uniões devem ser movidas para cima da árvore para que as junções apareçam
  - A distribuição da junção sobre a união pode ser expressa como:

$$(R_1 \cup R_2) \bowtie S = (R_1 \bowtie S) \cup (R_2 \bowtie S)$$

  - onde Ri são fragmentos de R, e S é uma relação.

Redução da fragmentação horizontal primária

- **Segundo, junções inúteis devem ser removidas**
- **Supondo que os fragmentos R_i e R_j sejam definidos respectivamente de acordo com os predicados p_i e p_j sobre o mesmo atributo, a regra de simplificação pode ser expressa da seguinte maneira:**

Rule 2: $R_i \bowtie R_j = \phi$ if $\forall x$ in $R_i, \forall y$ in $R_j : \neg(p_i(x) \wedge p_j(y))$

Redução da fragmentação horizontal primária

Exemplo:

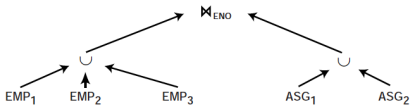
```
SELECT *
FROM   EMP, ASG
WHERE  EMP.ENO = ASG.ENO
```

$ASG_1 = \sigma_{ENO \leq "E3"}(ASG)$
$ASG_2 = \sigma_{ENO > "E3"}(ASG)$

ASG: funcionários e os projetos onde eles trabalham

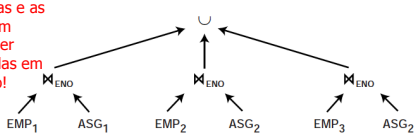
$EMP_1 = \sigma_{ENO \leq "E3"}(EMP)$
$EMP_2 = \sigma_{"E3" < ENO \leq "E6"}(EMP)$
$EMP_3 = \sigma_{ENO > "E6"}(EMP)$

Redução da fragmentação horizontal primária



(a) Localized query

As junções inúteis são eliminadas e as que ficam podem ser implementadas em paralelo!



(b) Reduced query

Redução para fragmentação vertical

- **O programa de localização para uma relação fragmentada verticalmente consiste na junção dos fragmentos sobre o atributo comum**
- **Consultas podem ser reduzidas pela determinação das relações intermediárias inúteis e pela remoção das subárvores que as produzem**

Redução para fragmentação vertical

- **Dada uma relação R definida sobre atributos A = {A_1, ..., A_n}, fragmentada verticalmente como Ri = $\Pi_{A'}(R)$, onde A' está contido em A, a regra pode ser expressa como:**

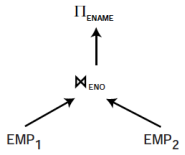
Rule 3: $\Pi_{D,K}(R_i)$ is useless if the set of projection attributes D is not in A'.

Redução para fragmentação vertical

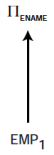
Exemplo:

```
SELECT ENAME
FROM   EMP
```

$EMP_1 = \Pi_{ENO,ENAME}(EMP)$
$EMP_2 = \Pi_{ENO,TITLE}(EMP)$



(a) Localized query



(b) Reduced query

O fragmento EMP2 não tem ENAME!

Redução para fragmentação derivada

- **A operação de junção pode ser otimizada quando as relações foram fragmentadas usando o mesmo predicado de seleção**
  - A junção de duas relações pode ser implementada como a união das junções parciais (as junções parciais podem ser executadas em paralelo)
- **Só vale quando os predicados usados na fragmentação são os mesmos!**

Redução para fragmentação derivada

- **Fragmentação derivada**
  - Uma maneira de distribuir duas relações para que o processamento conjunto da seleção e da junção seja otimizado
  - Os predicados não são os mesmos!
  - Consultas definidas sobre fragmentos derivados podem ser reduzidos
  - Este tipo de fragmentação é feito para otimizar junções
  - Uma regra útil consiste em :
    - 1. distribuir as junções sobre as uniões (i.e. Substituir junções por uniões de junções parciais)
    - 2. aplicar a regra 2 para redução de fragmentação horizontal

Redução para fragmentação derivada

Exemplo:

$ASG(ENO, PNO, RESP, DUR)$ Informações sobre os projetos onde os empregados estão alocados

$ASG_1 = ASG \ltimes_{ENO} EMP_1$
$ASG_2 = ASG \ltimes_{ENO} EMP_2$ } Fragmentação derivada

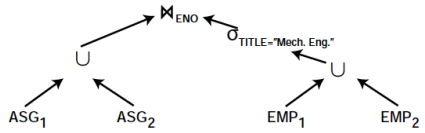
$EMP_1 = \sigma_{TITLE="Programmer"}(EMP)$
$EMP_2 = \sigma_{TITLE\neq"Programmer"}(EMP)$

$ASG = ASG_1 \cup ASG_2$ } Programa de localização

Redução para fragmentação derivada

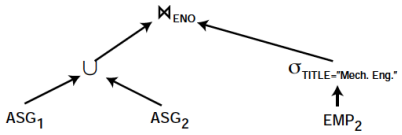
Exemplo:

```
SELECT *
FROM   EMP, ASG
WHERE  ASG.ENO = EMP.ENO
AND    TITLE = "Mech. Eng."
```



(A) LOCALIZED QUERY

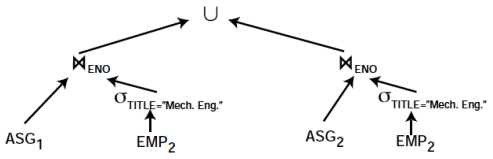
Redução para fragmentação derivada



(B) QUERY AFTER PUSHING SELECTION DOWN

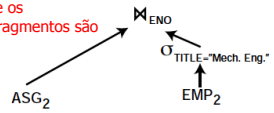
EMP1 é removido porque só tem programadores

Redução para fragmentação derivada



(C) QUERY AFTER MOVING UNIONS UP

A subárvore da esquerda é eliminada porque os predicados dos fragmentos são contraditórios



(D) REDUCED QUERY AFTER ELIMINATING THE LEFT SUBTREE

Redução para fragmentação híbrida

- **Obtida pela combinação da fragmentação horizontal e da fragmentação vertical**
- **O programa de localização emprega uniões e junções de fragmentos**
- **Consultas podem ser reduzidas combinando-se as regras utilizadas**
  - fragmentação horizontal primária
  - fragmentação vertical
  - fragmentação horizontal derivada

Redução para fragmentação híbrida

- **Consultas sobre fragmentos híbridos podem ser reduzidas combinando as regras anteriores:**
  1. Remover relações vazias geradas por seleções contraditórias sobre fragmentos horizontais
  2. Remover relações inúteis geradas por projeções sobre fragmentos verticais
  3. Distribuir junções sobre uniões para isolar e remover junções inúteis

Redução para fragmentação híbrida

Exemplo:

$$EMP_1 = \sigma_{ENO \leq "E4"}(\Pi_{ENO,ENAME}(EMP))$$
$$EMP_2 = \sigma_{ENO > "E4"}(\Pi_{ENO,ENAME}(EMP))$$
$$EMP_3 = \Pi_{ENO,TITLE}(EMP)$$

Fragmentação híbrida

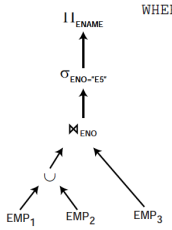
$$EMP = (EMP_1 \cup EMP_2) \bowtie_{ENO} EMP_3$$

Programa de localização

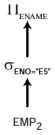
Redução para fragmentação híbrida

Exemplo:

```
SELECT ENAME
FROM   EMP
WHERE  ENO="E5"
```



(A) Localized query



(B) Reduced query

1. O fragmento EMP1 é eliminado por causa da seleção
2. O fragmento EMP3 é eliminado por causa da projeção

Conclusão

- **Decomposição de consultas e localização de dados são duas funções sucessivas**
- **Muitas consultas algébricas podem ser equivalentes a mesma consulta de entrada**
  - abordagens ingênuas são ineficientes
- **Reestruturação com a utilização de regras e regras de redução.**
- **Mais otimizações são geradas nas camadas subsequentes, considerando informações sobre o ambiente de processamento**