Complementos de Base de Dados

Resumos
2016/2017

João Alegria | 68661

Capítulo 5

Decomposição de consultas e Localização de dados

Decomposição de uma Query

A decomposição de uma query é realizada em 4 fases:

Nomalização	A query de cálculo é reescrita de forma normalizada
Análise	A query normalizada é analisada semanticamente, de forma a que as queries incorretas sejam rejeitadas
Simplificação	As queries corretas são simplificadas, eliminando predicados redundantes
Reestruturação	A query de cálculo é reestruturada para uma query algébrica

Normalização

Transformar a query numa forma normalizada
A parte mais importante é a transformação dos qualificadores (clausula 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

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

EXEMPLO

Encontre os nomes dos funcionários que trabalham no projeto P1 por 12 ou 24 meses

EMP(ENO, ENAME, TITLE)
PAY(TITLE, SAL)
PROJ(PNO, PNAME, BUDGET)
ASG(ENO, PNO, RESP, DUR)

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

Forma Normal Conjuntiva

$$\text{EMP.ENO} = \text{ASG.ENO} \wedge \text{ASG.PNO} = \text{“P1”} \wedge (\text{DUR} = 12 \vee \text{DUR} = 24)$$

Forma Normal Disjuntiva (Pode ser redundante)

$$(\text{EMP.ENO} = \text{ASG.ENO} \wedge \text{ASG.PNO} = \text{“P1”} \wedge \text{DUR} = 12) \vee$$
$$(\text{EMP.ENO} = \text{ASG.ENO} \wedge \text{ASG.PNO} = \text{“P1”} \wedge \text{DUR} = 24)$$

Análise

Espécie de compilador que serve para:
- Detetar erros de sintaxe
- Detetar se uma consulta está correta semanticamente
- Detetar tipos incorretos
  - nomes de atributos e relações
  - tipos de valores/atributos em operações

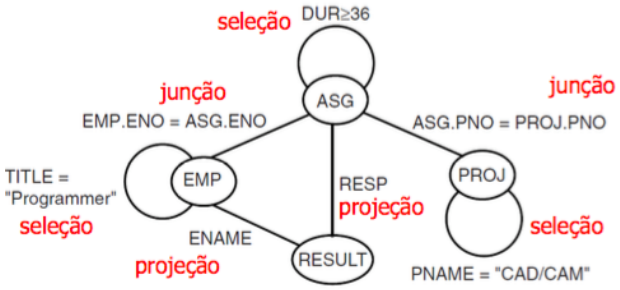
onde a consulta pode ser representada por meio de um grafo

EXEMPLO - SEMÂNTICA CORRETA

Encontre os nomes e as responsabilidades dos programadores que trabalham no projeto CAD/CAM por mais de três anos

EMP(ENO, ENAME, TITLE)
PAY(TITLE, SAL)
PROJ(PNO, PNAME, BUDGET)
ASG(ENO, PNO, RESP, DUR)

```
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"
```

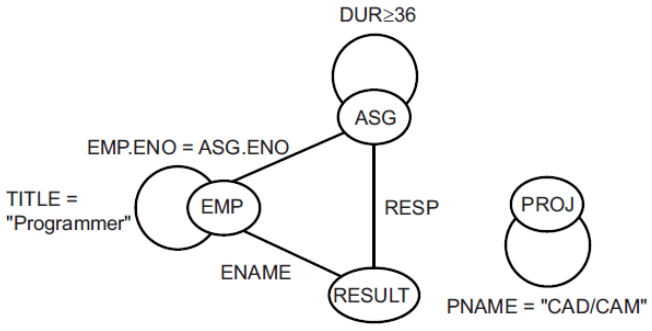


EXEMPLO - SEMÂNTICA INCORRETA

Uma query é semanticamente incorreta se o seu grafo da query não for conetado

EMP(ENO, ENAME, TITLE)
PAY(TITLE, SAL)
PROJ(PNO, PNAME, BUDGET)
ASG(ENO, PNO, RESP, DUR)

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



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

Simplificação / Eliminação de Redundância

Simplificação das consultas eliminando as redundâncias.

As redundâncias são muitas vezes devido a restrições de integridade semântica (forma normal conjuntiva ou disjuntiva)

Regras de Idempotência	Operadores Lógicos
$p \wedge p \Leftrightarrow p$	$p_1 \wedge p_2 \Leftrightarrow p_2 \wedge p_1$
$p \vee p \Leftrightarrow p$	$p_1 \vee p_2 \Leftrightarrow p_2 \vee p_1$
$p \wedge true \Leftrightarrow p$	$p_1 \wedge (p_2 \wedge p_3) \Leftrightarrow (p_1 \wedge p_2) \wedge p_3$
$p \vee false \Leftrightarrow p$	$p_1 \vee (p_2 \vee p_3) \Leftrightarrow (p_1 \vee p_2) \vee p_3$
$p \wedge false \Leftrightarrow false$	$p_1 \wedge (p_2 \vee p_3) \Leftrightarrow (p_1 \wedge p_2) \vee (p_1 \wedge p_3)$
$p \vee true \Leftrightarrow true$	$p_1 \vee (p_2 \wedge p_3) \Leftrightarrow (p_1 \vee p_2) \wedge (p_1 \vee p_3)$
$p \wedge \neg p \Leftrightarrow false$	$\neg(p_1 \wedge p_2) \Leftrightarrow \neg p_1 \vee \neg p_2$
$p \vee \neg p \Leftrightarrow true$	$\neg(p_1 \vee p_2) \Leftrightarrow \neg p_1 \wedge \neg p_2$
$p_1 \wedge (p_1 \vee p_2) \Leftrightarrow p_1$	$\neg(\neg p) \Leftrightarrow p$
$p_1 \vee (p_1 \wedge p_2) \Leftrightarrow p_1$	

EXEMPLO

p1: title="Programmer",
p2: title="Elect. Eng."
p3: ename="J. Doe"

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

$$(\neg p_1 \wedge (p_1 \vee p_2) \wedge \neg p_2) \vee p_3$$

$$(\neg p_1 \wedge \underline{(p_1 \vee p_2) \wedge \neg p_2}) \vee p_3$$

$$(\underline{\neg p_1 \wedge p_1} \wedge \neg p_2) \vee (\neg p_1 \wedge \underline{p_2 \wedge \neg p_2}) \vee p_3$$

$$(\underline{\neg p_1 \wedge ((p_1 \wedge \neg p_2) \vee (p_2 \wedge \neg p_2))}) \vee p_3$$

$$(false \wedge \neg p_2) \vee (\neg p_1 \wedge false) \vee p_3$$

$$false \vee false \vee p_3$$

p3: ename="J. Doe"

```
SELECT TITLE
FROM   EMP
WHERE  ENAME = "J. Doe"
```

Reestrutução / 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:**

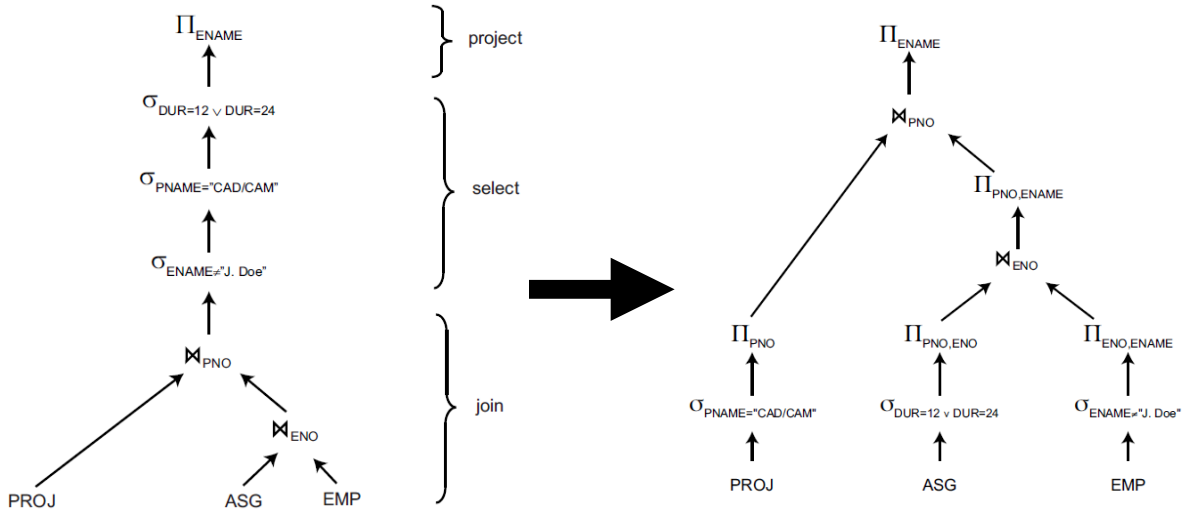
- Folhas = Relações
- Não folhas = Relações intermédias
- Sentido: folhas até à raíz

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

EXEMPLO

Selecionar todos empregados que não são "J.Doe" que trabalham no projeto 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)
```



Árvore Equivalentes:

- Fase de otimização: comparação de todas as árvores possíveis
- # árvores possíveis —> ∞
- A ideia consiste em aplicar as regras de transformação para que as “más árvores” 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)

Estrutura de Base de Dados Distribuídas

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

- 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.**

- **Regras de Redução:**
    - **Redução da Fragmentação Horizontal**
        - A função de fragmentação horizontal faz a distribuição duma relação com base nos predicados de seleção (o operador de reconstrução é a união)
        - As técnicas de redução para a fragmentação horizontal procuram determinar quais sub-árvores produzirão relações vazias para que sejam removidas

    - **Redução com seleção (select)**
        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 p_i e p_j são predicados de seleção, x denota um tuplo e p(x) denota “predicado p(x) válido para x”

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

EXEMPLO

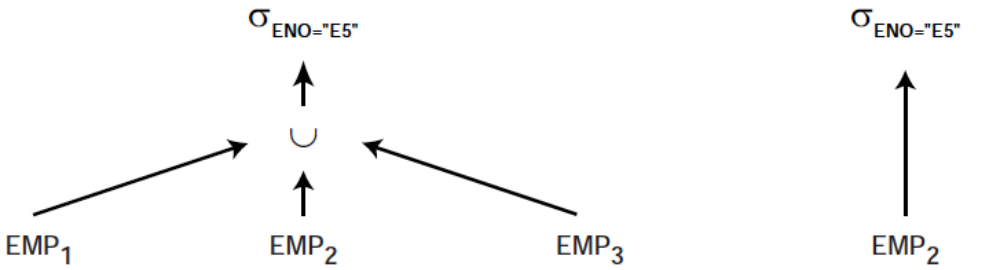
EMP(ENO, ENAME, TITLE)

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

→ Fragmentação Horizontal

$EMP = EMP_1 \cup EMP_2 \cup EMP_3$ → Programa de Localização (a evitar)

Considere a consulta:



(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 com junção (join)**
  - 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
  - Segundo, as junções inúteis devem ser removidas
  - Supondo que os R_i e R_j sejam definidos respetivamente 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 $R_j : \neg(p_i(x) \wedge p_j(y))$

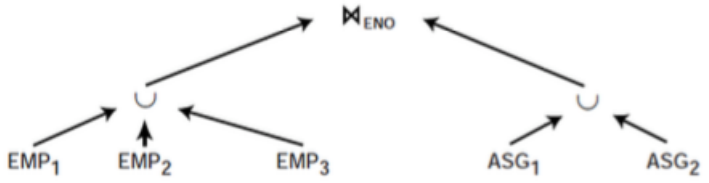
EXEMPLO

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

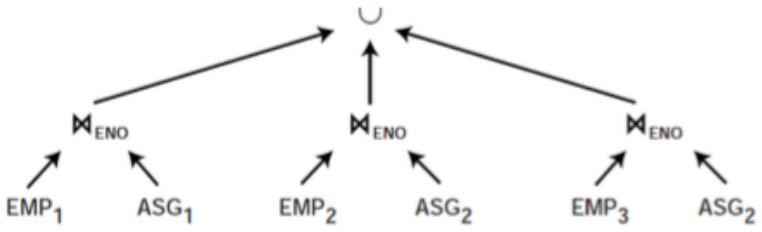
$ASG_1 = \sigma_{ENO \leq "E3"}(ASG)$
$ASG_2 = \sigma_{ENO > "E3"}(ASG)$

→ ASG: Funcionários e os projetos onde eles trabalham

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



(A) LOCALIZED QUERY



(B) REDUCED QUERY

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

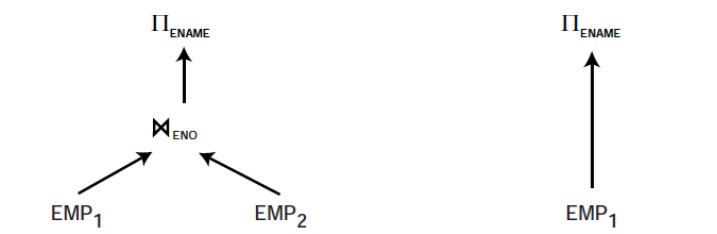
- **Redução da Fragmentação Vertical**
  - O programa de localização para uma relação fragmentada verticalmente consiste na junção dos fragmentos sobre o atributo comum
  - Consultas podem ser reduzidas pela determinação das relações intermédias inúteis e pela remoção das sub-árvores que as produzem
  - Dada uma relação R definida sobre atributos A = {A1, ..., An}, fragmentada verticalmente como Ri = πA'(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'.

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!

Fragmentação derivada

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

EXEMPLO

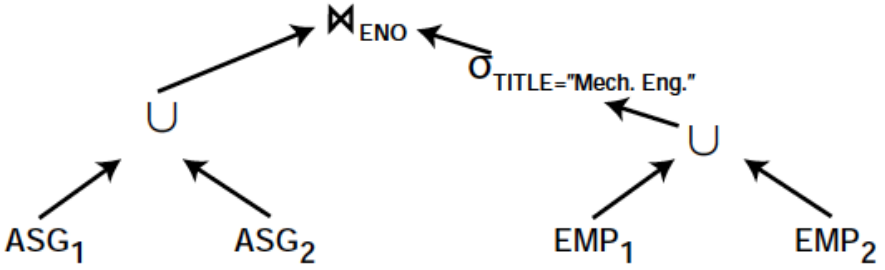
$ASG(ENO, PNO, RESP, DUR)$ → Informações sobre os projetos onde os empregados estão alocados

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

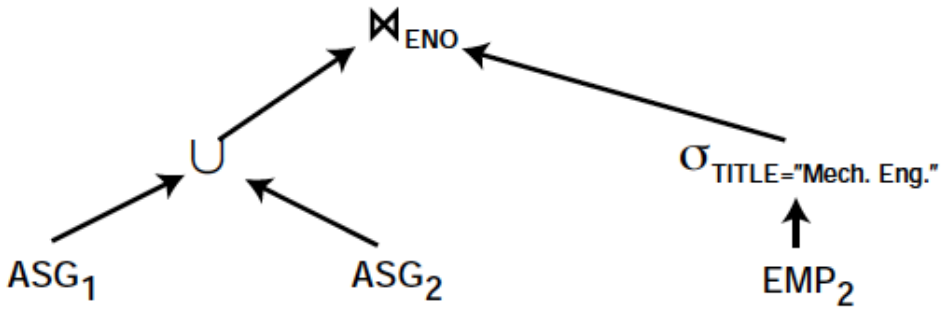
$$ASG_1 = ASG \ltimes_{ENO} EMP_1$$
$$ASG_2 = ASG \ltimes_{ENO} EMP_2$$

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

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

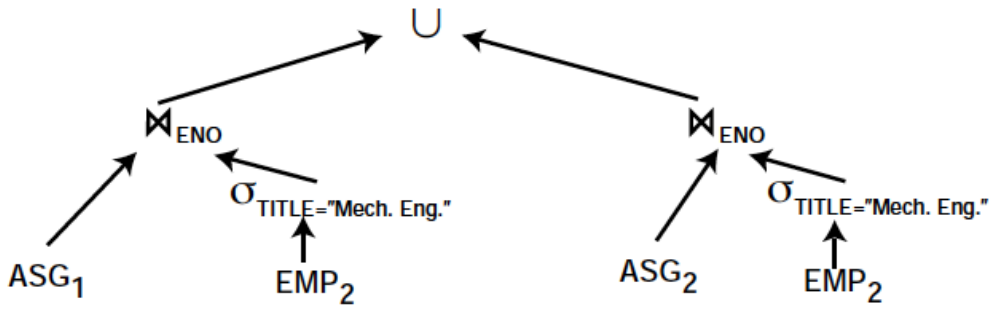


(A) LOCALIZED QUERY



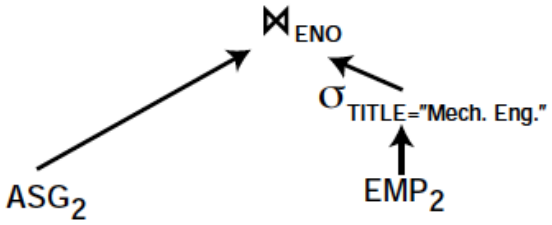
(B) QUERY AFTER PUSHING SELECTION DOWN

➡ EMP1 é removido porque só tem programadores



(c) QUERY AFTER MOVING UNIONS UP

➡A sub-árvore da esquerda é eliminada porque os predicados dos fragmentos são contraditórios



(D) REDUCED QUERY AFTER ELIMINATING THE LEFT SUBTREE

-**Redução para Fragmentação Híbrida**

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

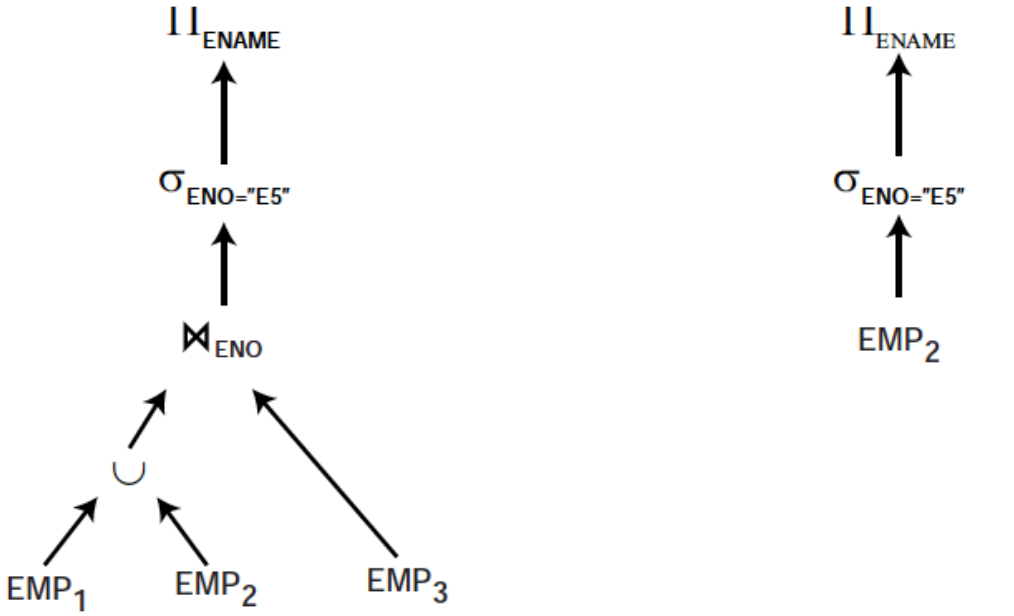
➡ Fragmentação híbrida

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

➡ Programa de localização

EXEMPLO

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



(A) LOCALIZED QUERY (B) REDUCED QUERY

➡ O fragmento EMP1 é eliminado por causa da seleção
➡ 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 à mesma consulta de entrada
- Abordagens ingénuas 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