
CHƯƠNG 5:

TỐI ƯU HÓA TRUY VẤN TRONG CSDL PHÂN TÁN

Thời lượng: 6 tiết

Giảng viên: ThS. Thái Bảo Trân

NỘI DUNG

1. Giới thiệu về xử lý truy vấn
2. Xử lý truy vấn trong môi trường tập trung
3. Xử lý truy vấn trong môi trường phân tán
4. Tối ưu hoá truy vấn trong CSDL phân tán

MỤC ĐÍCH

- Giới thiệu một bức tranh tổng quát của bộ tối ưu hóa truy vấn trong môi trường tập trung và phân tán
- Đề cập đến vấn đề tối ưu hoá trong cơ sở dữ liệu phân tán nghĩa là: **giảm chi phí bộ nhớ trung gian, giảm thời gian truy vấn cũng như giảm thời gian truyền dữ liệu trong các truy vấn phân tán.**
- Trình bày các quy trình xử lý truy vấn trong hệ thống phân tán

1. GIỚI THIỆU VỀ XỬ LÝ TRUY VẤN

Mục đích của xử lý truy vấn:

- Giảm thiểu thời gian xử lý
- Giảm vùng nhớ trung gian
- Giảm chi phí truyền thông giữa các trạm.
- Sử dụng ít tài nguyên

Chức năng của xử lý truy vấn:

- Biến đổi một truy vấn phức tạp thành một truy vấn tương đương đơn giản hơn.
- Phép biến đổi này phải đạt được cả về tính **đúng đắn** và **hiệu quả**
- Mỗi cách biến đổi dẫn đến việc sử dụng tài nguyên máy tính khác nhau, nên vấn đề đặt ra là lựa chọn phương án nào **dùng tài nguyên ít nhất**.

1. GIỚI THIỆU VỀ XỬ LÝ TRUY VẤN

Các phương pháp xử lý truy vấn cơ bản

- **Phương pháp biến đổi đại số:**

Đơn giản hóa câu truy vấn nhờ các phép biến đổi đại số tương đương nhằm giảm thiểu thời gian thực hiện các phép toán.

Phương pháp này không quan tâm đến kích thước và cấu trúc dữ liệu.

- **Phương pháp ước lượng chi phí:**

Xác định kích thước dữ liệu, thời gian thực hiện mỗi phép toán trong câu truy vấn.

Phương pháp này quan tâm đến kích thước dữ liệu và phải tính toán chi phí thời gian thực hiện mỗi phép toán.

2. XỬ LÝ TRUY VẤN TRONG MÔI TRƯỜNG TẬP TRUNG

2.1 So sánh xử lý truy vấn tập trung và phân tán

- Tập trung:

- Chọn một truy vấn đại số quan hệ tốt nhất trong số tất cả các truy vấn đại số tương đương.
- Các chiến lược xử lý truy vấn có thể biểu diễn trong sự mở rộng của đại số quan hệ.

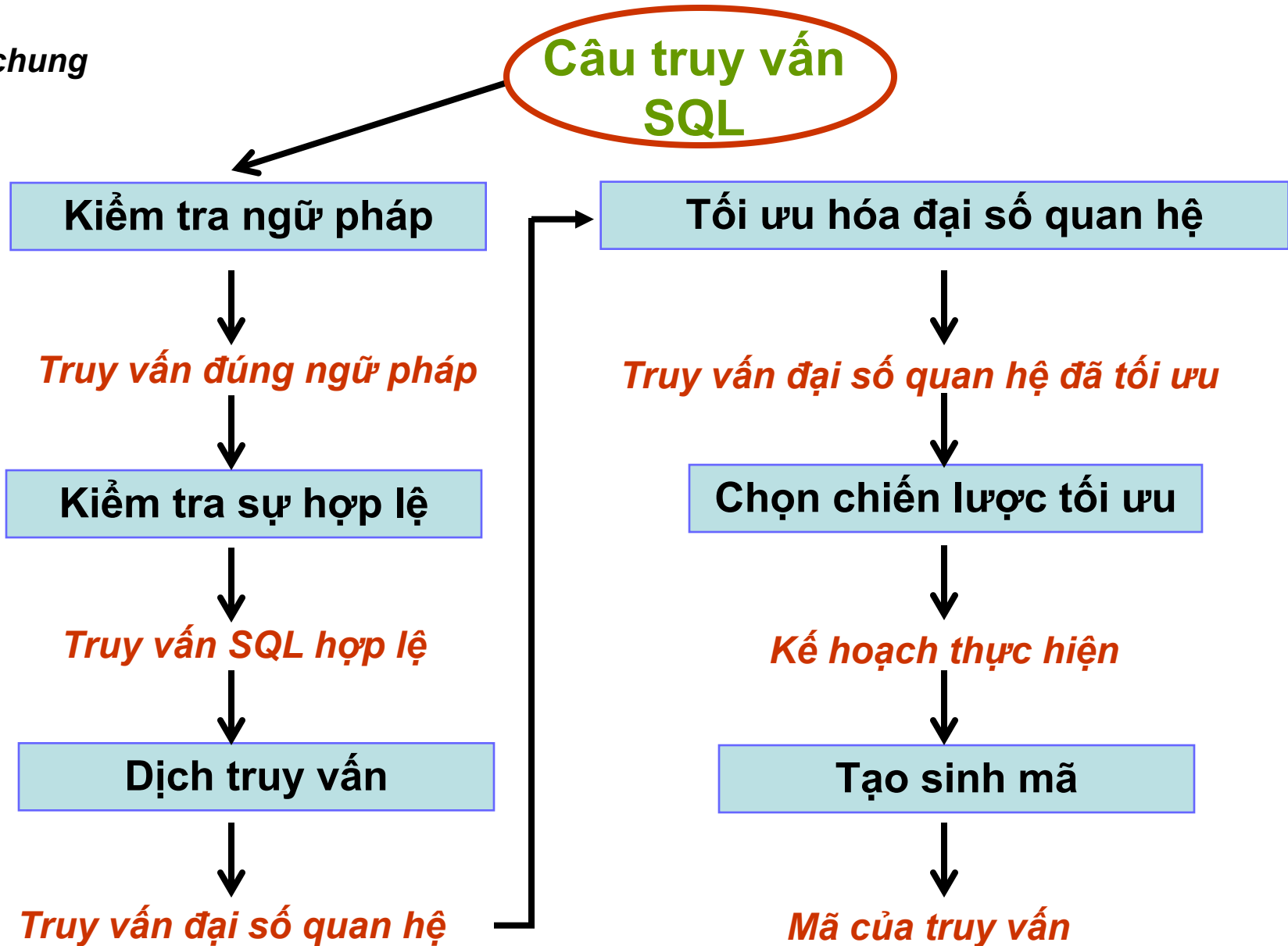
- Phân tán

- Kế thừa chiến lược xử lý truy vấn như môi trường tập trung
- Còn phải quan tâm thêm
 - Các phép toán truyền dữ liệu giữa các trạm
 - Chọn các trạm tốt nhất để xử lý dữ liệu
 - Cách truyền dữ liệu

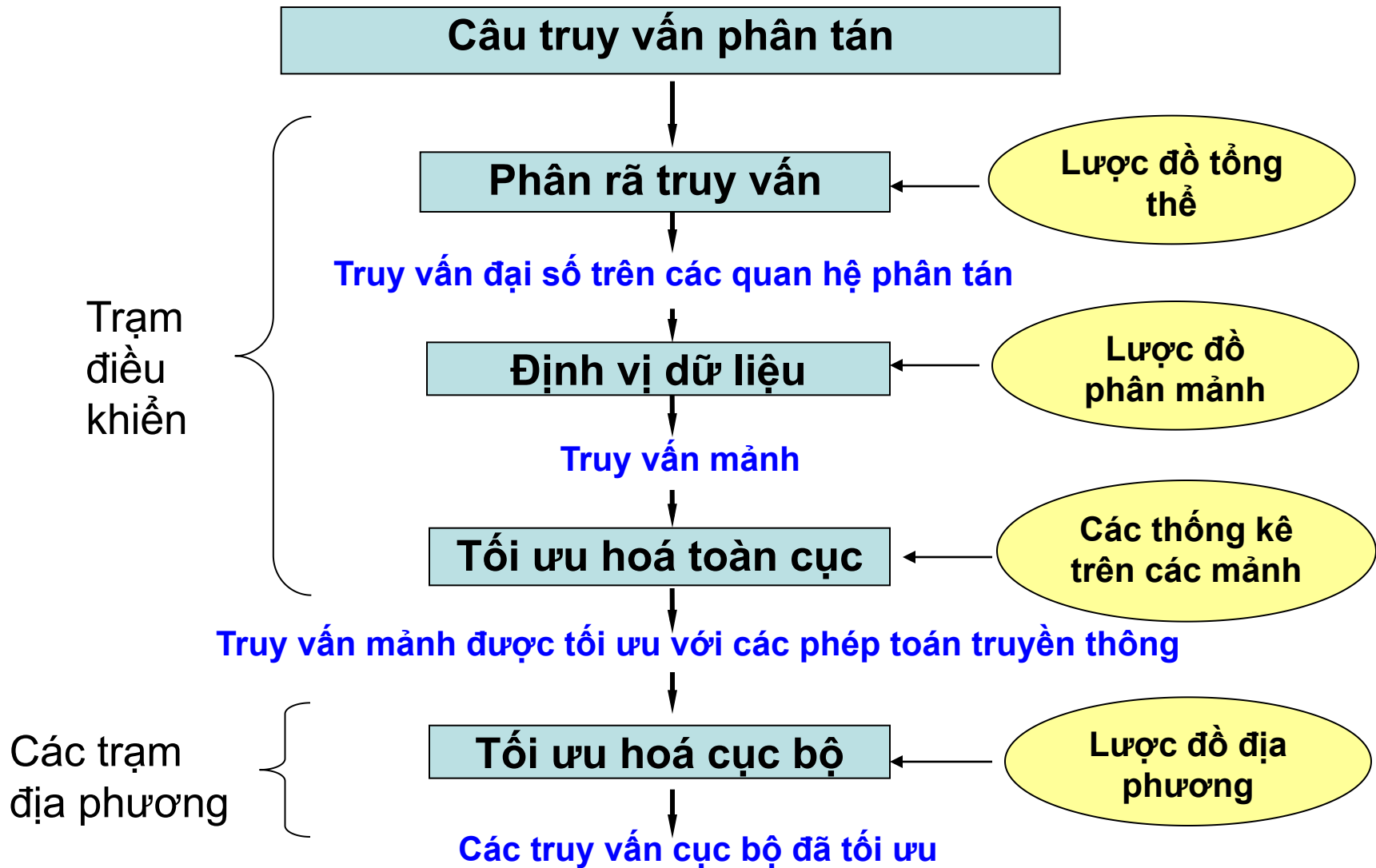
TỐI ƯU HÓA TRUY VẤN

Trong môi trường tập trung

Sơ đồ chung



Tối ưu hóa trong môi trường phân tán



Sơ đồ phân lớp chung cho xử lý truy vấn phân tán

2. XỬ LÝ TRUY VẤN TRONG MÔI TRƯỜNG TẬP TRUNG

2.2. Tối ưu hóa trong CSDL tập trung

Tại sao phải nghiên cứu xử lý truy vấn tập trung?

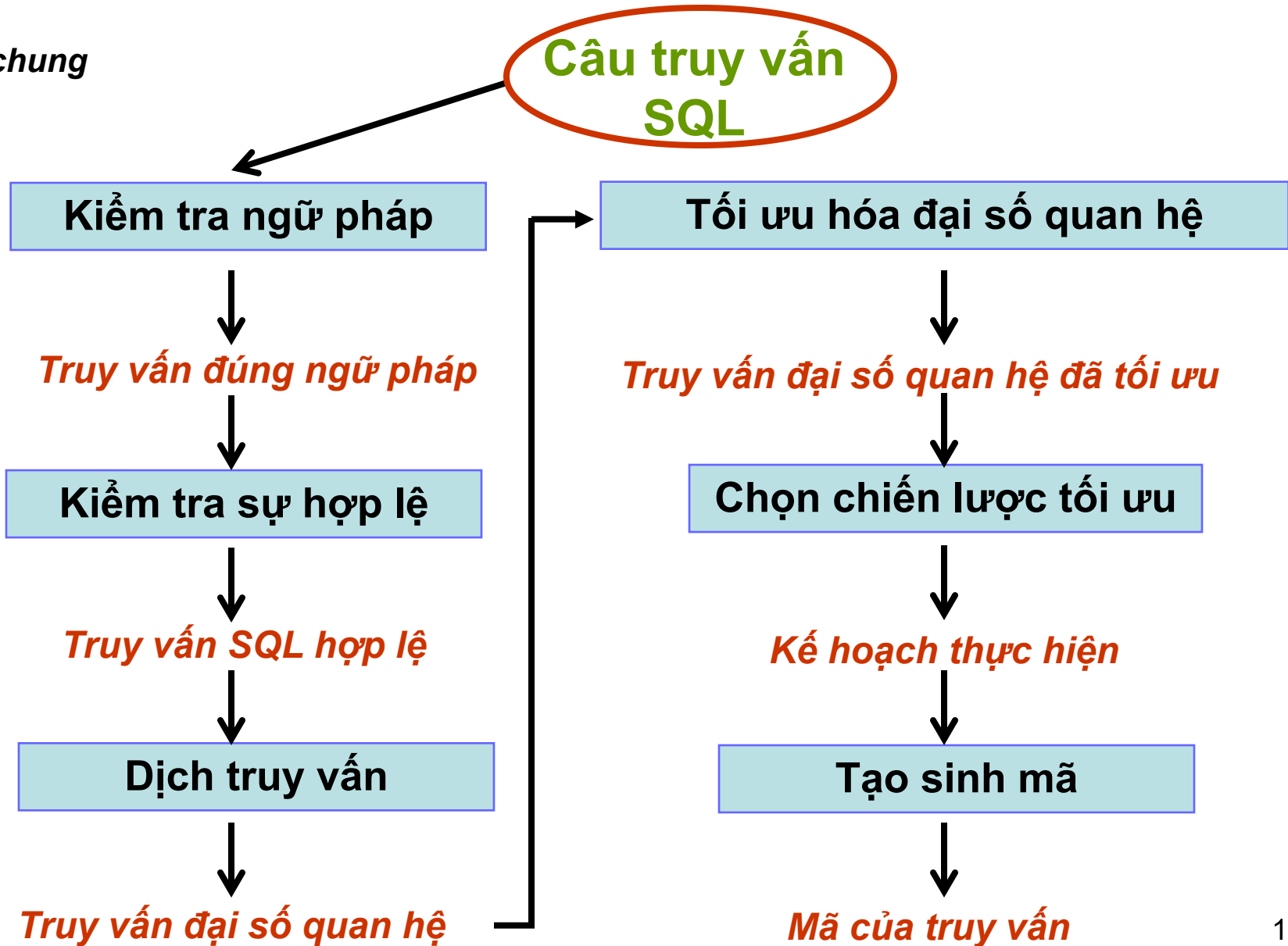
Để hiểu được các kỹ thuật tối ưu phân tán vì ba lí do:

- ❖ Câu truy vấn phân tán phải được dịch thành các câu truy vấn cục bộ, và được xử lí theo phương pháp tập trung.
- ❖ Các kỹ thuật tối ưu hoá phân tán thường là các mở rộng của kỹ thuật tập trung.
- ❖ Tối ưu hoá tập trung thường đơn giản.

TỐI ƯU HÓA TRUY VẤN

Trong môi trường tập trung

Sơ đồ chung



Ví dụ minh họa: xét CSDL của một công ty phần mềm

NHANVIEN (E)

MANV	TENNV	CHUCVU
A1	Nam	Phân tích HT
A2	Trung	Lập trình viên
A3	Đông	Phân tích HT
A4	Bắc	Phân tích HT
A5	Tây	Lập trình viên
A6	Hùng	Kỹ sư điện
A7	Dũng	Phân tích HT
A8	Chiến	Thiết kế DL

HOSO (G)

MANV	MADA	NHIEMVU	THOIGIAN
A1	D1	Quản lý	12
A2	D1	Phân tích	34
A2	D2	Phân tích	6
A3	D3	Kỹ thuật	12
A3	D4	Lập trình	10
A4	D2	Quản lý	6
A5	D2	Quản lý	20
A6	D4	Kỹ thuật	36
A7	D3	Quản lý	48
A8	D3	Lập trình	15

DUAN (J)

MADA	TENDA	NGANSACH
D1	CSDL	20000
D2	CÀI ĐẶT	12000
D3	BẢO TRÌ	28000
D4	P H Á T TRIỂN	25000

TLUONG (S)

CHUCVU	LUONG
Kỹ sư điện	1000
Phân tích HT	2500
Lập trình viên	3000
Thiết kế DL	4000

Bước 1: Kiểm tra ngữ pháp

DBMS sẽ kiểm tra ngữ pháp của truy vấn ban đầu:

- ❖ Nếu truy vấn bị sai ngữ pháp thì DBMS sẽ thông báo truy vấn bị sai ngữ pháp và truy vấn này sẽ không được thực hiện.
- ❖ Nếu truy vấn đúng ngữ pháp (syntactically correct SQL query) thì DBMS sẽ tiếp tục thực hiện bước 2.

**Q1: SELECT MANV, TENNV
FORM NHANVIEN**

Truy vấn này bị sai ngữ pháp (viết sai từ khóa FROM)

Bước 2: Kiểm tra sự hợp lệ

DBMS sẽ thực hiện các công việc:

- ❖ Kiểm tra **sự tồn tại** của các đối tượng dữ liệu (các cột, các biến, các bảng, ...) của truy vấn trong cơ sở dữ liệu.
- ❖ Kiểm tra **sự hợp lệ** về kiểu dữ liệu của các đối tượng dữ liệu (các cột, các biến, ...) trong truy vấn.

**Q2: SELECT MANV, TENNV
FROM NHAN_VIEN**

**Q3: SELECT MANV, TENNV
FROM NHANVIEN
WHERE MANV = 123**

Bước 3: Dịch truy vấn

❖ DBMS sẽ biến đổi truy vấn hợp lệ này thành một dạng biểu diễn bên trong hệ thống ở mức thấp hơn mà DBMS có thể sử dụng được.

❖ Một trong các dạng biểu diễn bên trong này là việc sử dụng đại số quan hệ bởi vì các phép toán đại số quan hệ được biến đổi dễ dàng thành các tác vụ của hệ thống:

“truy vấn ban đầu được biến đổi thành một biểu thức đại số quan hệ hay còn gọi là truy vấn đại số quan hệ (relational algebra query)”

Bước 3: Dịch truy vấn

Q4 : **SELECT** E.TENNV
 FROM E, G, J
 WHERE E.MANV = G.MANV
 AND G.MADA = J.MADA
 AND TENDA = "CSDL"

? Viết bằng ĐSQH

Bước 4: Tối ưu hóa biểu thức ĐSQH

❖ DBMS sử dụng các **phép biến đổi tương đương** của đại số quan hệ để biến đổi biểu thức đại số quan hệ có được ở bước 3 thành một biểu thức đại số quan hệ tương đương (theo nghĩa chúng có cùng một kết quả) nhưng biểu thức sau sẽ hiệu quả hơn: loại bỏ các phép toán không cần thiết và giảm vùng nhớ trung gian.

❖ Cuối cùng, **DBMS tạo ra một truy vấn đại số quan hệ đã được tối ưu hóa (optimized relational algebra query).**

Bước 4: Tối ưu hóa biểu thức ĐSQH

Q4 : **SELECT** E.TENNV
 FROM E, G, J
 WHERE E.MANV = G.MANV
 AND G.MADA = J.MADA
 AND TENDA = "CSDL"

? Tối ưu hóa Q4 + viết biểu thức ĐSQH

Bước 4: Tối ưu hóa biểu thức ĐSQH

Thuật toán INGRES

Ý tưởng thuật toán: Thuật toán tổ hợp hai giai đoạn **phân rã** và **tối ưu hoá**.

- Đầu tiên phân rã câu truy vấn dạng phép toán quan hệ thành các phần nhỏ hơn.
- Câu truy vấn được phân rã thành một chuỗi các truy vấn có một quan hệ chung duy nhất.
- Sau đó mỗi câu truy vấn đơn quan hệ được xử lý bởi một “**thể xử lý truy vấn một biến**” (one variable query processor-OVQP)

Bước 4: Tối ưu hóa biểu thức ĐSQH

Thuật toán INGRES (cont.)

- Trước tiên OVQP sẽ thực hiện các phép toán đơn ngôi và cố gắng giảm thiểu kích thước của các kết quả trung gian bằng các **phép tách** (detachment) và **Phép thế** (substitution)
- Kí hiệu $q_{i-1} \rightarrow q_i$ để chỉ câu truy vấn q được phân rã thành hai câu truy vấn con q_{i-1} và q_i , trong đó q_{i-1} được thực hiện trước và kết quả sẽ được q_i sử dụng.
- **Phép tách**: OVQP sử dụng để tách câu truy vấn q thành các truy vấn $q' \rightarrow q''$ dựa trên một quan hệ chung là kết quả của q' .

Bước 4: Tối ưu hóa biểu thức ĐSQH

Nếu câu truy vấn q được biểu diễn bằng SQL có dạng:

q : **SELECT** $R_2.A_2, R_3.A_3, \dots, R_n.A_n$
FROM R_1, R_2, \dots, R_n
WHERE $P_1(R_1.A'_1)$ **AND** $P_2(R_1.A_1, R_2.A_2, \dots, R_n.A_n)$

Trong đó: A_1 và A'_1 là các thuộc tính của quan hệ R_1 ,

P_1 là vị từ có chứa các thuộc tính của các quan hệ R_1, R_2, \dots, R_n .

Câu truy vấn trên có thể phân rã thành hai câu truy vấn con, q' theo sau là q'' qua **phép tách** dựa trên quan hệ chung R_1 như sau:

q' : **SELECT** $R_1.A_1$ **INTO** R'_1
FROM R_1
WHERE $P_1(R_1.A'_1)$

Trong đó R'_1 là một quan hệ tạm thời chứa các thông tin cần thiết để thực hiện tiếp tục câu truy vấn:

q'' : **SELECT** $R_2.A_2, \dots, R_n.A_n$
FROM R'_1, R_2, \dots, R_n
WHERE $P_2(R'_1.A_1, R_2.A_2, \dots, R_n.A_n)$

Ví dụ minh họa: xét CSDL của một công ty phần mềm

NHANVIEN (E)

MANV	TENNV	CHUCVU
A1	Nam	Phân tích HT
A2	Trung	Lập trình viên
A3	Đông	Phân tích HT
A4	Bắc	Phân tích HT
A5	Tây	Lập trình viên
A6	Hùng	Kỹ sư điện
A7	Dũng	Phân tích HT
A8	Chiến	Thiết kế DL

HOSO (G)

MANV	MADA	NHIEMVU	THOIGIAN
A1	D1	Quản lý	12
A2	D1	Phân tích	34
A2	D2	Phân tích	6
A3	D3	Kỹ thuật	12
A3	D4	Lập trình	10
A4	D2	Quản lý	6
A5	D2	Quản lý	20
A6	D4	Kỹ thuật	36
A7	D3	Quản lý	48
A8	D3	Lập trình	15

DUAN (J)

MADA	TENDA	NGANSACH
D1	CSDL	20000
D2	CÀI ĐẶT	12000
D3	BẢO TRÌ	28000
D4	P H Á T TRIỂN	25000

TLUONG (S)

CHUCVU	LUONG
Kỹ sư điện	1000
Phân tích HT	2500
Lập trình viên	3000
Thiết kế DL	4000

Bước 4: Tối ưu hóa biểu thức ĐSQH

Xét câu truy vấn $q_1 =$ “***Cho biết tên của các nhân viên đang làm việc trong dự án có tên CSDL***”

Diễn tả q_1 bằng SQL:

q_1 : **SELECT** E.TENNV
 FROM E, G, J
 WHERE E.MANV = G.MANV
 AND G.MADA = J.MADA
 AND TENDA = “CSDL”

q_1 được tách thành $q_{11} \rightarrow q'$, trong đó TGIAN1 là quan hệ trung gian.

q_{11} : **SELECT** J.MADA **INTO** TGIAN1
 FROM J
 WHERE TENDA = “CSDL”

q' : **SELECT** E.TENNV
 FROM E, G, TGIAN1
 WHERE E.MANV = G.MANV
 AND G.MADA =TGIAN1.MADA

Bước 4: Tối ưu hóa biểu thức ĐSQH

Các bước tách tiếp theo cho q' có thể tạo ra:

q_{12} : **SELECT** G.MANV INTO TGIAN2
 FROM G, TGIAN1
 WHERE G.MADA =TGIAN1.MADA

q_{13} : **SELECT** E.TENNV
 FROM E, TGIAN2
 WHERE E.MANV = TGIAN2.MANV

Truy vấn q_1 đã được rút gọn thành chuỗi truy vấn $q_{11} \rightarrow q_{12} \rightarrow q_{13}$. Truy vấn q_{11} là loại đơn quan hệ và có thể cho thực hiện bởi OVQP. Tuy nhiên các truy vấn q_{12} và q_{13} không phải loại đơn quan hệ và cũng không thể rút gọn hơn nữa bằng phép tách.

Các câu truy vấn đa quan hệ không thể tách tiếp được nữa (chẳng hạn q_{12} và q_{13}) được gọi là **bất khả giản** (irreducible).

Bước 4: Tối ưu hóa biểu thức ĐSQH

Các truy vấn bất khả giản được biến đổi thành câu truy vấn đơn quan hệ nhờ *phép thế bộ* (tuple substitution).

Phép thế bộ: Cho câu truy vấn n -quan hệ q , các bộ của một biểu được thay bằng các giá trị của chúng, tạo ra được một tập các truy vấn $(n-1)$ biểu.

Phép thế bộ được thực hiện như sau:

- Chọn một quan hệ trong truy vấn q để thay thế, gọi R_1 là quan hệ đó.
- Với mỗi bộ t_{1i} trong R_1 , các thuộc tính được tham chiếu trong q được thay bằng các giá trị thật sự trong t_{1i} , tạo ra một câu truy vấn q' có $(n-1)$ quan hệ. Như vậy số câu truy vấn q' được sinh ra bởi phép thế bộ là $\text{card}(R_1)$.

Tổng quát, phép thế bộ có thể mô tả như sau:

$q(R_1, R_2, \dots, R_n)$ được thay bởi $\{q'(t_{1i}, R_2, R_3, \dots, R_n), t_{1i} \in R_1\}$
Vì thế đối với mỗi bộ thu được, câu truy vấn con được xử lý đệ quy bằng phép thế nếu nó chưa bất khả giản.

Bước 4: Tối ưu hóa biểu thức ĐSQH

Ví dụ minh họa:

Xét tiếp câu truy vấn q_{13}

q_{13} : **SELECT** E.TENNV
 FROM E, TGIAN2
 WHERE E.MANV = TGIAN2.MANV

Quan hệ được định nghĩa bởi biến TGIAN2 chạy trên thuộc tính duy nhất MANV. Giả sử rằng nó chỉ chứa hai bộ: <E1> và <E2>. Phép thể cho TGIAN2 tạo ra hai câu truy vấn con đơn quan hệ:

q_{131} : **SELECT** E.TENNV
 FROM E
 WHERE E.MANV = "E1"

q_{132} : **SELECT** E.TENNV
 FROM E
 WHERE E.MANV = "E2"

Sau đó chúng có thể được OVQP quản lý và sử dụng.

Bước 4: Tối ưu hóa biểu thức ĐSQH

Nhận xét:

- Thuật toán tối ưu hoá INGRES (được gọi là INGRES - QOA) sẽ xử lý **đệ qui** cho đến khi không còn câu truy vấn đa quan hệ nào nữa.
- Thuật toán có thể được áp dụng cho các phép chọn và các phép chiếu ngay khi có thể sử dụng kỹ thuật tách.
- Kết quả của câu truy vấn đơn quan hệ được lưu trong những cấu trúc dữ liệu có khả năng tối ưu hoá những câu truy vấn sau đó (như các nối) và sẽ được OVQP sử dụng.
- Các câu truy vấn bất khả giản còn lại sau phép tách sẽ được xử lý bằng phép thế bộ.
- Câu truy vấn bất khả giản, được kí hiệu là MRQ'. Quan hệ nhỏ nhất với lực lượng của nó đã được biết từ kết quả của câu truy vấn trước đó sẽ được chọn để thay thế.

Bước 4: Tối ưu hóa biểu thức ĐSQH

Thuật toán *INGRES-QOA*

Input: MRQ: câu truy vấn đa quan hệ (có n quan hệ)

Output: Câu truy vấn tối ưu

Begin

Output $\leftarrow \phi$

If $n=1$ **then**

Output $\leftarrow \text{run}(\text{MRQ})$ {thực hiện câu truy vấn một quan hệ}

Else {Tách MRQ thành m tr.vấn một quan hệ và một tr.vấn đa quan hệ}

$\text{ORQ}_1, \dots, \text{ORQ}_m, \text{MRQ}' \leftarrow \text{MRQ}$

For $i \leftarrow 1$ **to** m

Output' $\leftarrow \text{run}(\text{ORQ}_i)$ {thực hiện ORQ_i }

Output $\leftarrow \text{output} \cup \text{output}'$ {trộn tất cả các kết quả lại}

Endfor

$R \leftarrow \text{CHOOSE_VARIABLE}(\text{MRQ}')$ {R được chọn cho phép thế bộ}

For mỗi bộ $t \in R$

$\text{MRQ}'' \leftarrow \text{thay giá trị cho } t \text{ trong } \text{MRQ}'$

Output' $\leftarrow \text{INGRES-QOA}(\text{MRQ}'')$ {gọi đệ qui}

Output $\leftarrow \text{output} \cup \text{output}'$ {trộn tất cả các kết quả lại}

Endfor

Endif

End. {INGRES-----QOA}

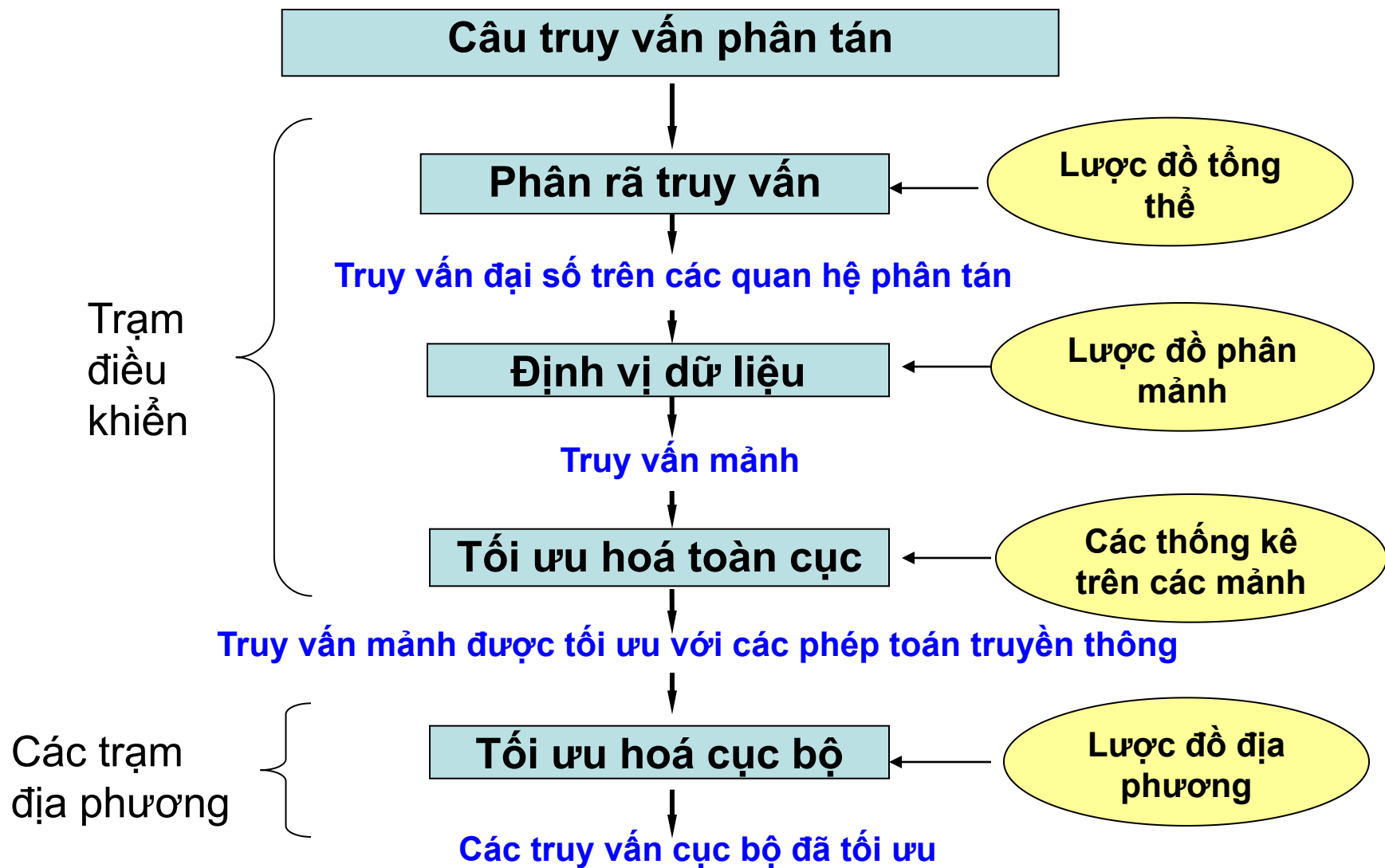
Bước 5:

- DBMS sử dụng các thông số về **kích thước** của các bảng, các chỉ mục ... để xác định cách xử lý truy vấn.
- DBMS sẽ **đánh giá chi phí** của các kế hoạch thực hiện khác nhau có thể có để từ đó chọn ra một kế hoạch thực hiện (execution plan) cụ thể sao cho tốn ít chi phí nhất (thời gian xử lý và vùng nhớ trung gian).
- Các thông số dùng để đánh giá chi phí của kế hoạch thực hiện gồm: số lần và loại truy xuất đĩa, kích thước của vùng nhớ chính và vùng nhớ ngoài và thời gian thực hiện của các tác vụ để tạo ra kết quả của truy vấn.
- Cuối cùng, DBMS tạo ra một **kế hoạch** thực hiện cho truy vấn.

Bước 6: Tạo sinh mã

- Kế hoạch thực hiện của truy vấn có được ở cuối bước 5 sẽ được **mã hoá** và thực hiện.

3. Xử lý truy vấn trong môi trường phân tán



Sơ đồ phân lớp chung cho xử lý truy vấn phân tán

3. Xử lý truy vấn trong môi trường phân tán

3.1 Phân rã truy vấn - Tối ưu hóa toàn cục

Bước này giống với các bước 1, 2, 3 và 4 của tối ưu hóa truy vấn trong cơ sở dữ liệu tập trung.

Giai đoạn **Phân rã truy vấn** chia làm bốn bước:

- **Chuẩn hoá**
- **Phân tích**
- **Loại bỏ dư thừa**
- **Viết lại**

3.1.1 Chuẩn hoá

Mục đích:

Chuyển đổi truy vấn thành một dạng chuẩn để thuận lợi cho các xử lý tiếp theo.

3. Xử lý truy vấn trong môi trường phân tán

Với SQL, có hai dạng chuẩn cho các vị từ trong mệnh đề WHERE là:

Dạng chuẩn hội là hội (\wedge) của những phép toán tuyển (\vee):

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

Dạng chuẩn tuyển là tuyển (\vee) của những phép toán hội (\wedge):

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

trong đó p_{ij} là các biểu thức nguyên tố.

ĐẠI SỐ MỆNH ĐỀ

Bảng các tương đương logic thường dùng

Đặt T= hằng đúng, F = hằng sai

1. $p \wedge F \Leftrightarrow F$

Domination laws-Luật nuốt

2. $p \vee T \Leftrightarrow T$

3. $p \vee F \Leftrightarrow p$

Identity laws-Luật đồng nhất

4. $p \wedge T \Leftrightarrow p$

5. $p \wedge p \Leftrightarrow p$

Idempotent laws-Luật lũy đẳng

6. $p \vee p \Leftrightarrow p$

7. $\neg(\neg p) \Leftrightarrow p$

Double negation law-Luật phủ định kép

8. $p \wedge \neg p \Leftrightarrow F$

Cancellation laws-Luật xóa bỏ

9. $p \vee \neg p \Leftrightarrow T$

10. $p \wedge q \Leftrightarrow q \wedge p$

Commutative laws-Luật giao hoán

ĐẠI SỐ MỆNH ĐỀ

Bảng các tương đương logic thường dùng (tt)

11. $p \vee q \Leftrightarrow q \vee p$

12. $(p \wedge q) \wedge r \Leftrightarrow p \wedge (q \wedge r)$ Associative laws-Luật kết hợp

13. $(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$

14. $p \wedge (q \vee r) \Leftrightarrow (p \wedge q) \vee (p \wedge r)$ Distributive laws-Luật phân phối

15. $p \vee (q \wedge r) \Leftrightarrow (p \vee q) \wedge (p \vee r)$

16. $\neg(p \vee q) \Leftrightarrow \neg p \wedge \neg q$ De Morgan's laws-Luật De Morgan

17. $\neg(p \wedge q) \Leftrightarrow \neg p \vee \neg q$

18. $(p \Rightarrow q) \Leftrightarrow (\neg p \vee q)$ Implication law-Luật kéo theo

19. $p \vee (p \wedge q) = p$

20. $p \wedge (p \vee q) = p$

3. Xử lý truy vấn trong môi trường phân tán

Ví dụ minh họa: xét CSDL công ty phần mềm đã cho

Từ các quan hệ: E= NHANVIEN (MANV, TENNV, CHUCVU) và
 G= HOSO (MANV, MADA, NHIEMVU, THOIGIAN).

Xét truy vấn: ***“Tìm tên các nhân viên làm dự án có mã số J1 với thời gian 12 hoặc 24 tháng”*** .

Truy vấn trên được biểu diễn trong SQL:

```
SELECT      E.TENNV
FROM        E, G
WHERE       E.MANV= G.MANV
           AND  G.MADA="J1"
           AND  THOIGIAN=12 OR THOIGIAN=24
```

Điều kiện trong dạng chuẩn hội là:

$E.MANV=G.MANV \wedge G.MADA="J1" \wedge (THOIGIAN=12 \vee THOIGIAN=24)$

Điều kiện trong dạng chuẩn tuyển là:

$(E.MANV=G.MANV \wedge G.MADA="J1" \wedge THOIGIAN=12) \vee$
 $(E.MANV=G.MANV \wedge G.MADA="J1" \wedge THOIGIAN=24)$

3. Xử lý truy vấn trong môi trường phân tán

3.1.2 Phân tích

Mục đích:

Phát hiện ra những thành phần không đúng (sai kiểu hoặc sai ngữ nghĩa) và loại bỏ chúng sớm nhất nếu có thể.

Truy vấn sai kiểu: nếu một thuộc tính bất kỳ hoặc tên quan hệ của nó không được định nghĩa trong lược đồ tổng thể, hoặc phép toán áp dụng cho các thuộc tính sai kiểu.

Ví dụ: Truy vấn dưới đây là sai kiểu

```
SELECT      E#  
FROM        E  
WHERE       E.TENNV > 200
```

vì hai lý do:

- Thuộc tính E# không khai báo trong lược đồ
- Phép toán “>200” không thích hợp với kiểu chuỗi của thuộc tính E.TENNV

3. Xử lý truy vấn trong môi trường phân tán

Truy vấn sai ngữ nghĩa: nếu các thành phần của nó không tham gia vào việc tạo ra kết quả.

Để xác định truy vấn có sai về ngữ nghĩa hay không, ta dựa trên việc biểu diễn truy vấn như một đồ thị gọi là ***đồ thị truy vấn***.

Đồ thị này được xác định bởi các truy vấn liên quan đến phép chọn, chiếu và nối. Nếu đồ thị truy vấn mà **không liên thông thì truy vấn là sai ngữ nghĩa**.

3. Xử lý truy vấn trong môi trường phân tán

❖ *Đồ thị truy vấn:*

- Có **một nút** dùng để biểu diễn cho quan hệ **kết quả**
- Các **nút khác** biểu diễn cho các toán hạng trong câu truy vấn (các quan hệ)
- **Cạnh nối** giữa hai nút mà **không phải là nút kết quả** thì biểu diễn một **phép nối**.
- Cạnh có **nút đích là nút kết quả** thì biểu diễn một **phép chiếu**.
- Một nút **không phải là nút kết quả** có thể được gán nhãn bởi **phép chọn** hoặc **phép tự nối** (self-join: nối của quan hệ với chính nó).

❖ *Đồ thị kết nối:*

- Là một đồ thị con của đồ thị truy vấn (join graph), trong đó chỉ có **phép nối**.

3. Xử lý truy vấn trong môi trường phân tán

Ví dụ: Từ các quan hệ:

E = E (MANV, TENNV, CHUCVU),

G = HOSO (MANV, MADA, NHIEMVU, THOIGIAN) và

J = DUAN (MADA, TENDA, NGANSACH).

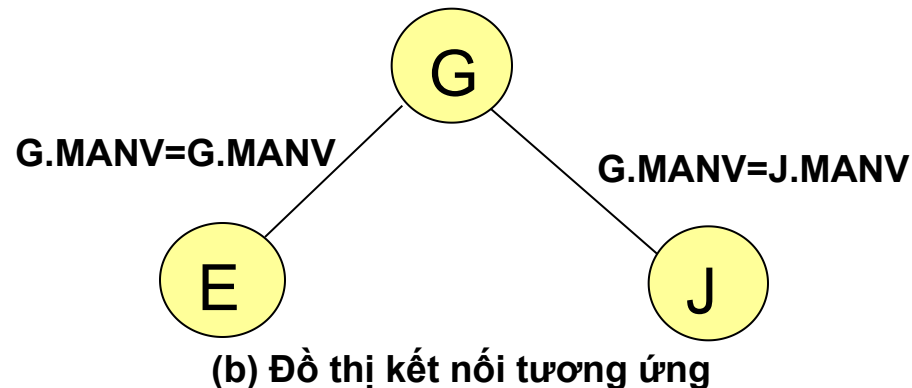
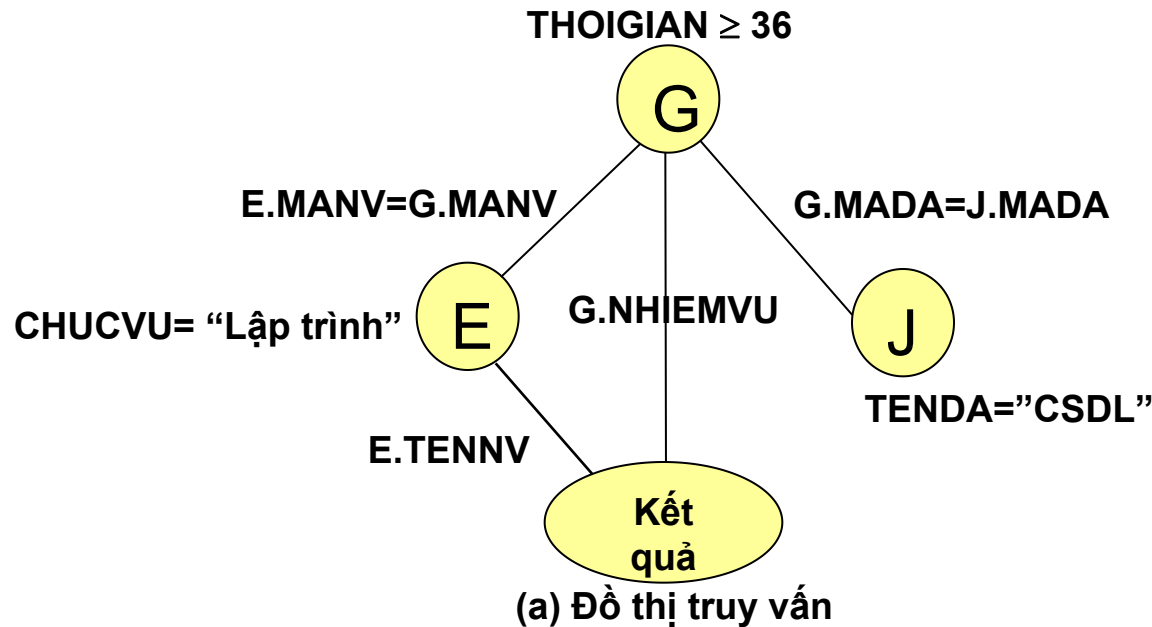
Hãy xác định “*Tên và nhiệm vụ các lập trình viên làm dự án CSDL có thời gian lớn hơn 3 năm.*”

Truy vấn SQL tương ứng là:

```
SELECT    E.TENNV, G.NHIEMVU
FROM      E, G, J
WHERE     E.MANV=G.MANV
          AND  G.MADA.= J.MADA
          AND  TENDA="CSDL"
          AND  THOIGIAN ≥ 36
          AND  NHIEMVU="LTRINH"
```

3. Xử lý truy vấn trong môi trường phân tán

Đồ thị truy vấn và đồ thị kết nối tương ứng

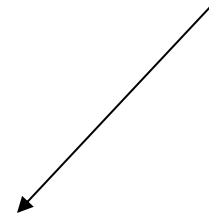


3. Xử lý truy vấn trong môi trường phân tán

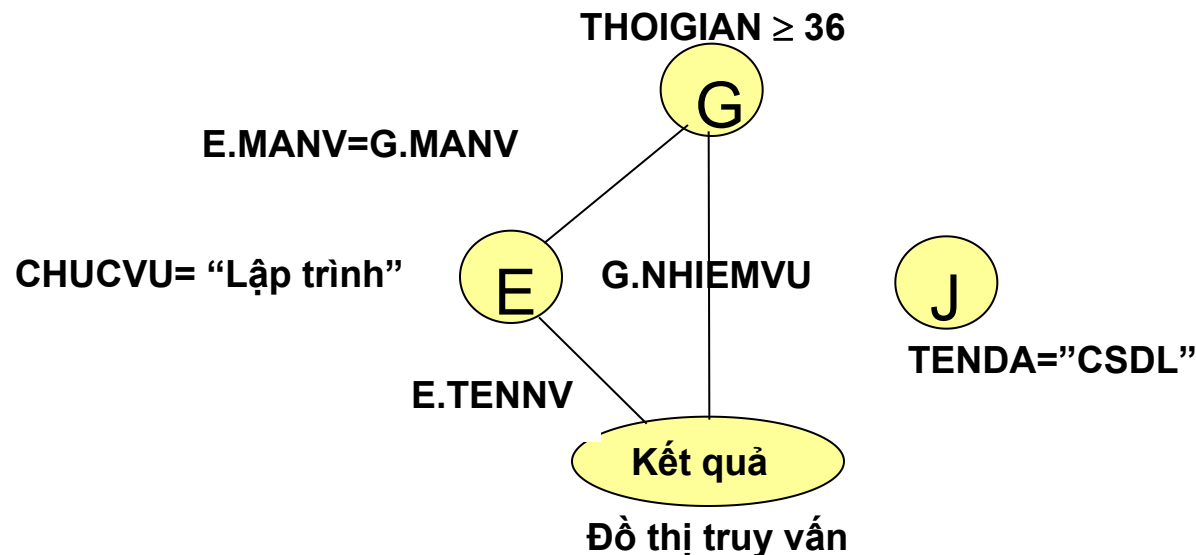
Xét câu truy vấn SQL tương ứng:

```
SELECT   E.TENNV, NHIEMVU
FROM     E, G, J
WHERE     E.MANV=G.MANV
           AND   TENDA="CSDL"
           AND   THOIGIAN ≥ 36
           AND   CHUCVU="Lập trình"
```

thiếu **AND** G.MADA=J.MADA



Truy vấn này là sai ngữ nghĩa vì đồ thị truy vấn của nó không liên thông.



3. Xử lý truy vấn trong môi trường phân tán

3.1.3 Loại bỏ dư thừa

- Điều kiện trong các truy vấn có thể có chứa các vị từ dư thừa.
- Một đánh giá sơ sai về một điều kiện dư thừa có thể dẫn đến lặp lại một số công việc.
- Sự dư thừa vị từ và dư thừa công việc có thể được loại bỏ bằng cách làm đơn giản hoá các điều kiện thông qua các **luật lũy đẳng** sau:

$$1. p \wedge p \Leftrightarrow p$$

$$3. p \vee p \Leftrightarrow p$$

$$5. p \wedge \text{true} \Leftrightarrow p$$

$$7. p \vee \text{false} \Leftrightarrow p$$

$$9. p \wedge \text{false} \Leftrightarrow \text{false}$$

$$2. p \vee \text{true} \Leftrightarrow \text{true}$$

$$4. p \wedge \neg p \Leftrightarrow \text{false}$$

$$6. p \vee \neg p \Leftrightarrow \text{true}$$

$$8. p_1 \wedge (p_1 \vee p_2) \Leftrightarrow p_1$$

$$10. p_1 \vee (p_1 \wedge p_2) \Leftrightarrow p_1$$

Ví dụ: Đơn giản hoá câu truy vấn sau:

3. Xử lý truy vấn trong môi trường phân tán

```
SELECT  E.CHUCVU
FROM    E
WHERE   (NOT(E.CHUCVU="Lập trình")
        AND (E.CHUCVU="Lập trình" OR E.CHUCVU="Kỹ sư điện")
        AND NOT(E.CHUCVU="Kỹ sư điện")
        OR E.TENNV="Dũng")
```

Đặt $p1$: $\langle \text{CHUCVU} = \text{"Lập trình"} \rangle$, $p2$: $\langle \text{CHUCVU} = \text{"Kỹ sư điện"} \rangle$,
 $p3$: $\langle \text{E.TENNV} = \text{"Dũng"} \rangle$.

Các vị từ sau mệnh đề WHERE được mô tả lại:

$$\begin{aligned} p &: (\neg p1 \wedge (p1 \vee p2) \wedge \neg p2) \vee p3 \\ &\Leftrightarrow ((\neg p1 \wedge p1 \wedge \neg p2) \vee (\neg p1 \wedge p2 \wedge \neg p2)) \vee p3 && \text{(áp dụng luật 7)} \\ &\Leftrightarrow ((\text{false} \wedge \neg p2) \vee (\neg p1 \wedge \text{false})) \vee p3 && \text{(áp dụng luật 5)} \\ &\Leftrightarrow (\text{false} \vee \text{false}) \vee p3 && \text{(áp dụng luật 4)} \\ &\Leftrightarrow p3 \end{aligned}$$

Vậy câu truy vấn được biến đổi thành:

```
SELECT  E.CHUCVU
FROM    E
WHERE   E.TENNV="Dũng"
```

3. Xử lý truy vấn trong môi trường phân tán

3.1.4 Viết lại

Bước này được chia làm hai bước con như sau:

- **Biến đổi** trực tiếp truy vấn phép tính sang **đại số quan hệ**.
- **Cấu trúc lại** truy vấn đại số quan hệ để **cải thiện hiệu quả** thực hiện.

Đại số quan hệ là một cây mà nút **lá** biểu diễn một quan hệ trong CSDL, các nút **không lá** là các quan hệ trung gian được sinh ra bởi các phép toán đại số quan hệ.

3. Xử lý truy vấn trong môi trường phân tán

Cách chuyển một truy vấn phép tính quan hệ thành một cây đại số quan hệ:

- Các **nút lá** khác nhau được tạo cho mỗi biến bộ khác nhau (tương ứng một quan hệ). Trong SQL các nút lá chính là các quan hệ trong mệnh đề **FROM**.
- **Nút gốc** được tạo ra bởi một phép chiếu lên các thuộc tính **kết quả**. Trong SQL nút gốc được xác định qua mệnh đề **SELECT**.
- **Điều kiện** (mệnh đề **WHERE** trong SQL) được biến đổi thành dãy các phép toán đại số thích hợp (phép chọn, nối, phép hợp, v.v...) đi từ lá đến gốc, có thể thực hiện theo thứ tự xuất hiện của các vị từ và các phép toán.

3. Xử lý truy vấn trong môi trường phân tán

Ví dụ:

Truy vấn “*Tìm tên các nhân viên không phải là “Dũng”, làm việc cho dự án CSDL với thời gian một hoặc hai năm*”.

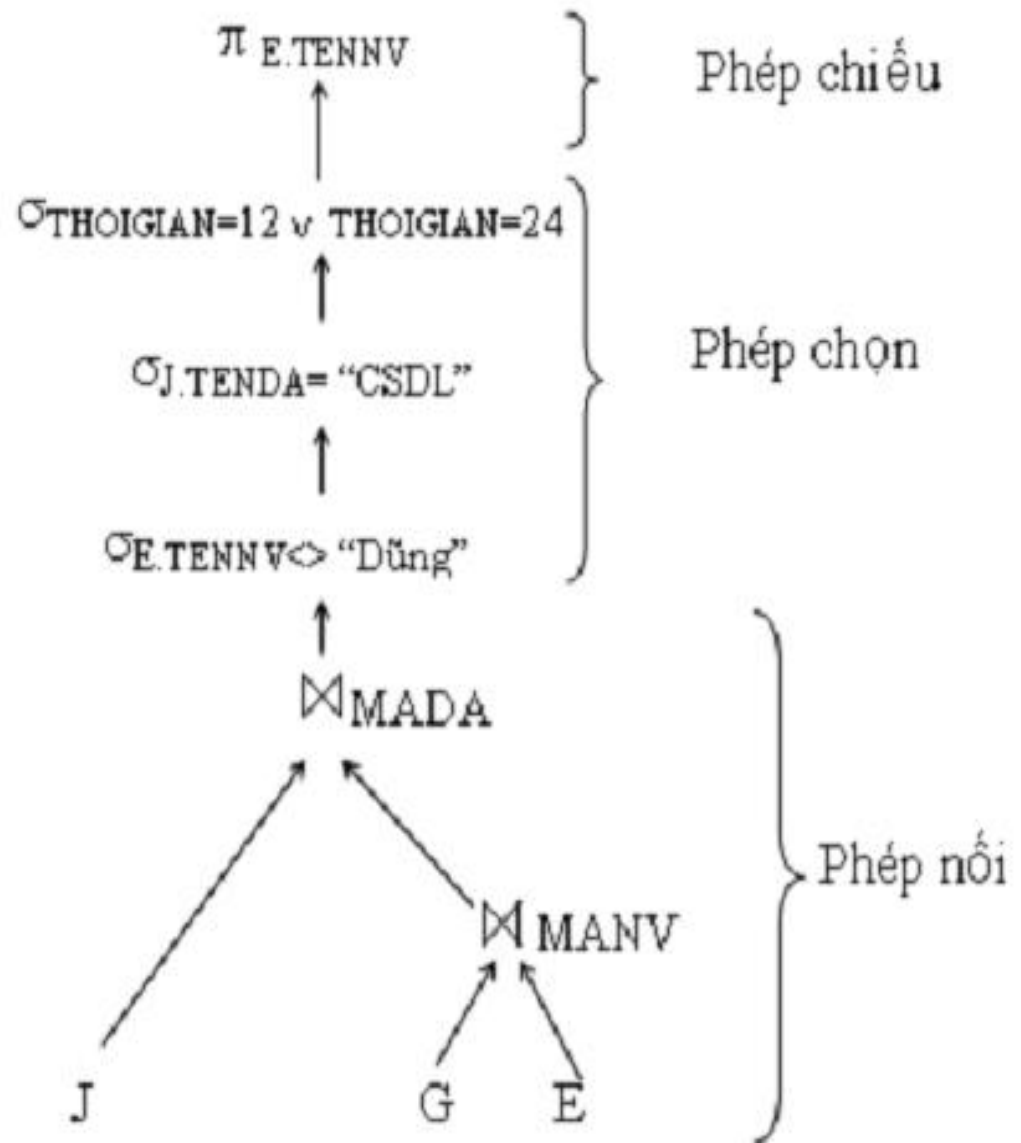
Biểu diễn truy vấn này trong SQL là:

```
SELECT      E.TENNV
FROM        J, G, E
WHERE        G.MANV=E.MANV
               AND  G.MADA= J.MADA
               AND  E.TENNV <> “Dũng”
               AND  J.TENDA= “CSDL”
               AND  (THOIGIAN=12 OR THOIGIAN=24)
```

3. Xử lý truy vấn trong môi trường phân tán

**E = E (MANV, TENNV, CHUCVU),
G = HOSO (MANV, MADA, NHIEMVU,
THOIGIAN) và
J = DUAN (MADA, TENDA,
NGANSACH).**

SELECT E.TENNV
FROM J, G, E
WHERE G.MANV=E.MANV
AND G.MADA= J.MADA
AND E.TENNV <> "Dũng"
AND J.TENDA= "CSDL"
AND (THOIGIAN=12 **OR**
THOIGIAN=24)



3. Xử lý truy vấn trong môi trường phân tán

06 luật biến đổi phép toán đại số quan hệ:

Mục đích: dùng để biến đổi cây đại số quan hệ thành các cây tương đương (trong đó có thể có cây tối ưu).

Giả sử R, S, T là các quan hệ, R được định nghĩa trên toàn bộ thuộc tính $A=\{A_1, \dots, A_n\}$, S được định nghĩa trên toàn bộ thuộc tính $B=\{B_1, \dots, B_n\}$.

(1). Tính giao hoán của các phép toán hai ngôi:

Phép tích Decartes và phép nối hai quan hệ có tính giao hoán.

i. $R \times S \Leftrightarrow S \times R$

ii. $R \bowtie S \Leftrightarrow S \bowtie R$

(2). Tính kết hợp của các phép toán hai ngôi:

Phép tích Decartes và phép nối hai quan hệ có tính kết hợp.

i. $(R \times S) \times T \Leftrightarrow R \times (S \times T)$

ii. $(R \bowtie S) \bowtie T \Leftrightarrow R \bowtie (S \bowtie T)$

3. Xử lý truy vấn trong môi trường phân tán

(3). Tính luỹ đẳng của những phép toán một ngôi

- Dãy các phép chiếu khác nhau trên cùng quan hệ được tổ hợp thành một phép chiếu và ngược lại:

$$\Pi_{A'}(\Pi_{A''}(R)) \Leftrightarrow \Pi_{A'}(R) \quad A', A'' \subseteq R \text{ và } A' \subseteq A''$$

- Dãy các phép chọn khác nhau $\sigma_{p_i(A_i)}$ trên cùng một quan hệ, với p_i là một vị từ được gán vào thuộc tính A_i , có thể được tổ hợp thành một phép chọn.

$$\sigma_{p_1(A_1)}(\sigma_{p_2(A_2)}(R)) = \sigma_{p_1(A_1) \wedge p_2(A_2)}(R)$$

3. Xử lý truy vấn trong môi trường phân tán

(4). Phép chọn giao hoán với phép chiếu

$$\Pi_{A_1, \dots, A_n} (\sigma_{p(A_p)} (R)) = \Pi_{A_1, \dots, A_n} (\sigma_{p(A_p)} (\Pi_{A_1, \dots, A_n, A_p} (R)))$$

Nếu A_p là thành viên của $\{A_1, \dots, A_n\}$, biểu thức trên trở thành

$$\Pi_{A_1, \dots, A_n} (\sigma_{p(A_p)} (R)) = \sigma_{p(A_p)} (\Pi_{A_1, \dots, A_n, A_p} (R))$$

(5). Phép chọn giao hoán với những phép toán hai ngôi

- Phép chọn với phép nhân: $\sigma_{p(A_p)} (R \times S) \Leftrightarrow \sigma_{p(A_p)} (R) \times S$
- Phép chọn với phép nối:

$$\sigma_{p(A_i)} (R \bowtie_{(A_i, B_k)} S) \Leftrightarrow \sigma_{p(A_i)} (R) \bowtie_{(A_i, B_k)} S$$

- Phép chọn với phép hợp: Nếu R và T cùng bộ thuộc tính.

$$\sigma_{p(A_i)} (R \cup T) \Leftrightarrow \sigma_{p(A_i)} (R) \cup \sigma_{p(A_i)} (T)$$

3. Xử lý truy vấn trong môi trường phân tán

(6). *Phép chiếu giao hoán với những phép toán hai ngôi*

- ***Phép chiếu và tích Decartes:***

Nếu $C=A' \cup B'$ với $A' \subseteq A$, $B' \subseteq B$, và A, B là tập các thuộc tính trên quan hệ R, S ta có:

$$\Pi_C(R \times S) = \Pi_{A'}(R) \times \Pi_{B'}(S)$$

- ***Phép chiếu và phép nối:***

$$\Pi_C(R \bowtie_{p(A_i, B_j)} S) = \Pi_{A'}(R) \bowtie_{p(A_i, B_j)} \Pi_{B'}(S)$$

- ***Phép chiếu và phép hợp:***

$$\Pi_C(R \cup S) = \Pi_{A'}(R) \cup \Pi_{B'}(S)$$

Chú ý: Việc sử dụng sáu luật trên có khả năng sinh ra nhiều cây đại số quan hệ tương đương nhau. Vấn đề là xác định cho được cây tối ưu.

3. Xử lý truy vấn trong môi trường phân tán

Chú ý:

Trong giai đoạn tối ưu, sự so sánh các cây có thể thực hiện dựa trên chi phí dự đoán của chúng. Tuy nhiên, nếu số lượng các cây quá lớn thì cách tiếp cận này sẽ không hiệu quả. Chúng ta có thể dùng 6 luật trên để cấu trúc lại cây, nhằm loại bỏ những cây đại số quan hệ “tồi”.

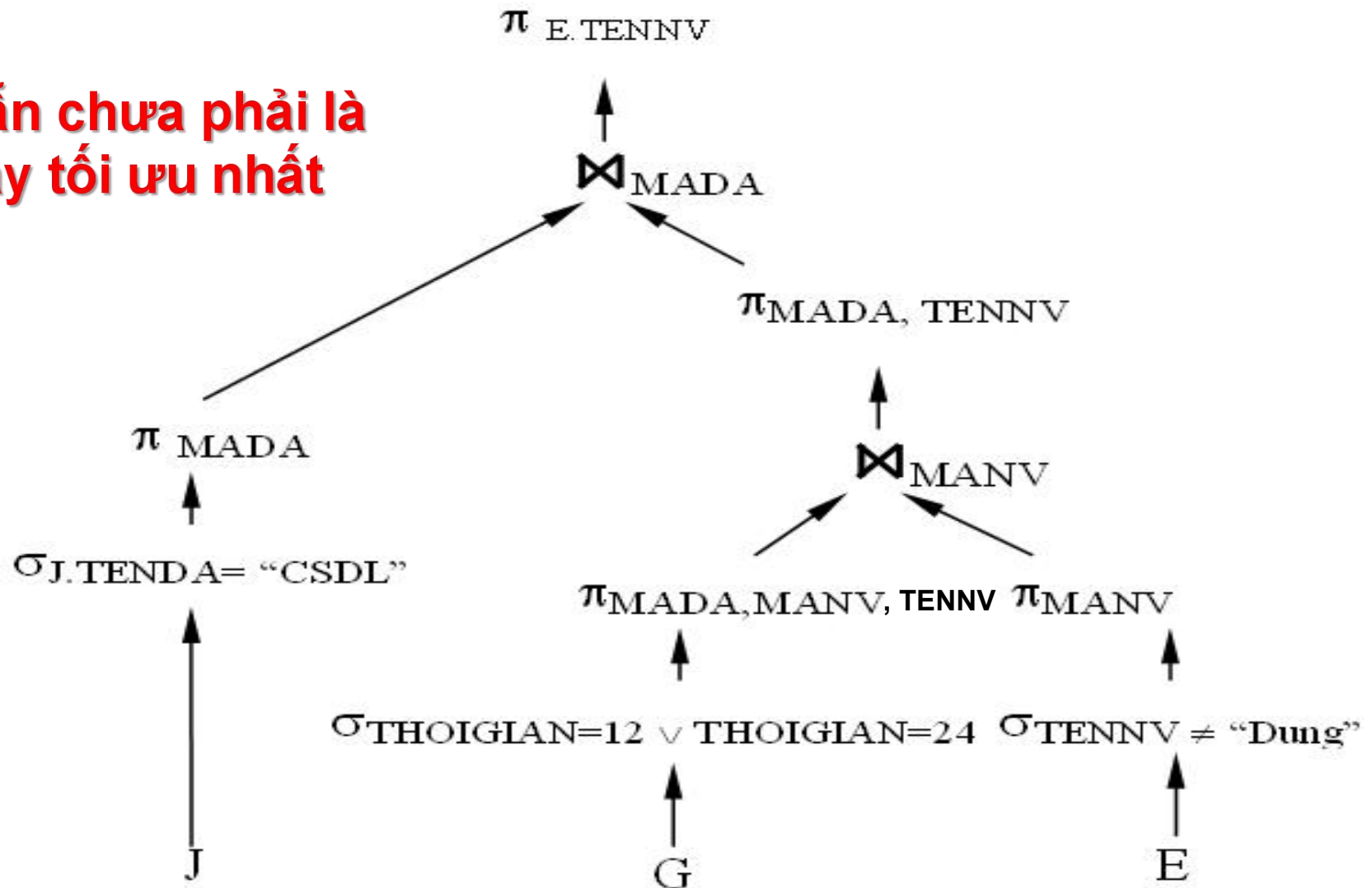
Các luật trên có thể sử dụng theo bốn cách như sau:

- **Phân rã** các phép toán một ngôi, **đơn giản hóa** biểu thức truy vấn .
- **Nhóm** các phép toán một ngôi trên cùng một quan hệ để giảm số lần thực hiện.
- **Giao hoán** các phép toán một ngôi với các phép toán hai ngôi để ưu tiên cho một số phép toán (chẳng hạn phép chọn).
- **Sắp thứ tự** các phép toán hai ngôi trong thực hiện truy vấn.

3. Xử lý truy vấn trong môi trường phân tán

Ví dụ: Cấu trúc lại cây truy vấn ở ví dụ trên, cho ra cây kết quả tốt hơn cây ban đầu.

**Vẫn chưa phải là
cây tối ưu nhất**



3. Xử lý truy vấn trong môi trường phân tán

3.2. Định vị dữ liệu phân tán - Tối ưu hóa cục bộ

- ❖ Lớp định vị biến đổi một truy vấn đại số quan hệ tổng thể thành một truy vấn đại số được biểu thị trên các mảnh vật lý.
- ❖ Sử dụng thông tin được lưu trữ trên các lược đồ phân mảnh để định vị.
- ❖ Chương trình đại số quan hệ xây dựng lại quan hệ tổng thể từ các phân mảnh của nó gọi là **chương trình định vị**.
- ❖ Truy vấn có được từ chương trình định vị gọi là truy vấn ban đầu.
- ❖ **Chú ý**: Trong phần dưới đây, với **mỗi kiểu phân mảnh** chúng ta sẽ biểu diễn một **kỹ thuật rút gọn** để sinh ra truy vấn được tối ưu và đơn giản hoá.

3. Xử lý truy vấn trong môi trường phân tán

- ❖ **Tối ưu hóa truy vấn trên lược đồ phân mảnh bao gồm 2 bước sau:**
 - **Bước 1:** Biến đổi biểu thức đại số quan hệ trên lược đồ toàn cục (chứa các quan hệ toàn cục) thành biểu thức đại số quan hệ trên lược đồ phân mảnh (chứa các mảnh của quan hệ toàn cục) bằng cách thay thế các quan hệ toàn cục bởi biểu thức tái lập của chúng.
 - **Bước 2:** Đơn giản hoá (Rút gọn) biểu thức đại số quan hệ trên lược đồ phân mảnh để có được một biểu thức hiệu quả (loại bỏ các phép toán không cần thiết giảm vùng nhớ trung gian) bằng cách sử dụng các phép biến đổi tương đương của đại số quan hệ và các đại số quan hệ được tuyển chọn.

3. Xử lý truy vấn trong môi trường phân tán

❖ Đơn giản hoá một biểu thức đại số quan hệ trên lược đồ phân mảnh được thực hiện dựa trên các luật sau:

1. Di chuyển các phép chọn xuống các nút lá của cây, và sau đó áp dụng chúng bằng cách dùng đại số quan hệ được tuyển chọn; thay thế các kết quả chọn lựa bởi quan hệ rỗng nếu điều kiện chọn của kết quả bị mâu thuẫn.
2. Để phân phối các phép kết xuất hiện trong một truy vấn toàn cục, các phép hợp (biểu diễn tập hợp của các phân mảnh) phải được di chuyển lên phía trên các phép kết mà chúng ta muốn phân phối để loại bỏ các phép kết không cần thiết.
3. Dùng đại số quan hệ được tuyển chọn để định vị điều kiện chọn của các toán hạng của các phép kết; thay thế cây con, bao gồm phép kết và các toán hạng của nó bằng quan hệ rỗng nếu điều kiện chọn của kết quả của phép kết bị mâu thuẫn.

3. Xử lý truy vấn trong môi trường phân tán

3.2.1 Rút gọn theo phân mảnh ngang nguyên thủy

Xét quan hệ $E(\text{MANV}, \text{TENNV}, \text{CHUCVU})$.

Tách quan hệ này thành ba mảnh ngang E_1 , E_2 và E_3 như sau:

$$E_1 = \sigma_{\text{MANV} \leq "e3"}(E) \quad E_2 = \sigma_{"e3" < \text{MANV} \leq "e6"}(E) \quad E_3 = \sigma_{\text{MANV} > "e6"}(E)$$

- ❑ Chương trình định vị cho quan hệ E : $E = E_1 \cup E_2 \cup E_3$.
- ❑ Dạng ban đầu của bất kỳ truy vấn nào được xác định trên E là có được bằng cách thay thế nó bởi $E_1 \cup E_2 \cup E_3$.
- ❑ Việc rút gọn các truy vấn trên các quan hệ đã được phân mảnh ngang bao gồm việc *xác định câu truy vấn*, sau khi đã cấu trúc lại cây con. Điều này sẽ sinh ra một số quan hệ rỗng, và sẽ *loại bỏ* chúng.
- ❑ Phân mảnh ngang có thể được khai thác để làm **đơn giản cả phép chọn và phép nối**.

3. Xử lý truy vấn trong môi trường phân tán

a. Rút gọn với phép chọn: cho một quan hệ R được phân mảnh ngang thành R_1, R_2, \dots, R_n với $R_j = \sigma_{p_j}(R)$

Qui tắc 1: $\sigma_{p_i}(R_j) = \phi$ nếu $\forall x \in R : \neg(p_i(x) \wedge p_j(x))$.

Trong đó:

p_i, p_j là vị từ chọn, x là bộ dữ liệu, $p(x)$ là vị từ p chiếm giữ x .

- Di chuyển các phép chọn xuống các nút lá của cây và sau đó áp dụng chúng bằng cách dùng đại số quan hệ được tuyển chọn;
- Thay thế các kết quả chọn lựa bởi quan hệ rỗng nếu điều kiện chọn của kết quả bị mâu thuẫn.

3. Xử lý truy vấn trong môi trường phân tán

Ví dụ: Hãy rút gọn truy vấn

SELECT *

FROM E

WHERE MANV="e5"

Với E được tách thành ba mảnh ngang E_1 , E_2 và E_3 :

$$E_1 = \sigma_{MANV \leq "e3"}(E)$$

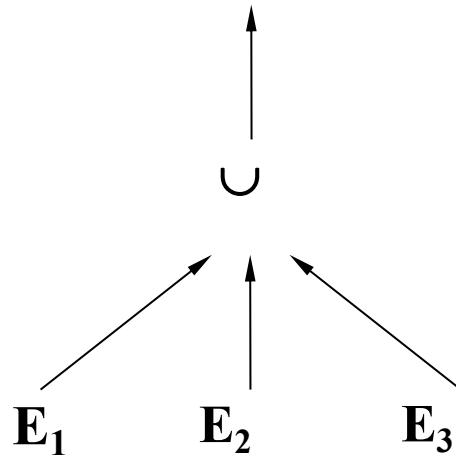
$$E_2 = \sigma_{"e3" < MANV \leq "e6"}(E)$$

$$E_3 = \sigma_{MANV > "e6"}(E)$$

3. Xử lý truy vấn trong môi trường phân tán

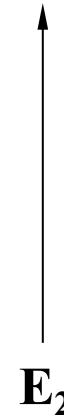
$$\begin{aligned} E_1 &= \sigma_{\text{MANV} \leq \text{"e3"}}(E) \\ E_2 &= \sigma_{\text{"e3"} < \text{MANV} \leq \text{"e6"}}(E) \\ E_3 &= \sigma_{\text{MANV} > \text{"e6"}}(E) \end{aligned}$$

$\sigma_{\text{MANV}=\text{"e5"}}$



(a) Truy vấn ban đầu

$\sigma_{\text{MANV}=\text{"e5"}}$



(b) Truy vấn rút gọn

Rút gọn bằng cách sử dụng tính chất giao hoán phép chọn với phép hợp, chúng ta thấy **vị từ chọn đối lập với vị từ E1 và E3** nên sinh ra các quan hệ rỗng.

3. Xử lý truy vấn trong môi trường phân tán

b. Rút gọn với phép nối

- Các phép nối trên quan hệ đã được phân mảnh ngang có thể đơn giản khi chúng được phân mảnh theo thuộc tính nối.
- Việc rút gọn được thực hiện dựa trên tính **phân phối giữa phép nối và phép hợp** và **loại bỏ các phép nối vô ích**.
- Với tính chất, $(R_1 \cup R_2) \bowtie R_3 = (R_1 \bowtie R_3) \cup (R_2 \bowtie R_3)$, R_i là các phân mảnh. Chúng ta có thể xác định được các phép nối vô ích của các mảnh khi các điều kiện nối mâu thuẫn nhau. Sau đó, dùng luật 2 dưới đây để loại bỏ các phép nối vô ích.

3. Xử lý truy vấn trong môi trường phân tán

➤ Qui tắc 2:

$R_i \bowtie R_j = \emptyset$ nếu $\forall x \in R_i, \forall y \in R_j : \neg (p_i(x) \wedge p_j(y))$.

Trong đó R_i, R_j được xác định theo các vị từ p_i, p_j trên cùng thuộc tính.

- Để phân phối các phép kết xuất hiện trong một truy vấn toàn cục, các phép hợp (biểu diễn tập hợp của các phân mảnh) phải được di chuyển lên phía trên các phép kết mà chúng ta muốn phân phối để loại bỏ các phép kết không cần thiết.

3. Xử lý truy vấn trong môi trường phân tán

Nhận xét:

- Việc xác định các phép nối vô ích được thực hiện bằng cách chỉ xem xét các vị từ mảnh.
- Truy vấn rút gọn không phải luôn tốt hơn hoặc đơn giản hơn truy vấn ban đầu.
- Một thuận lợi của truy vấn rút gọn là những phép nối có thể thực hiện song song.

3. Xử lý truy vấn trong môi trường phân tán

Ví dụ: Giả sử quan hệ E được phân mảnh thành các mảnh

$$E_1 = \sigma_{MANV \leq "e3"}(E) \quad E_2 = \sigma_{"e3" < MANV \leq "e6"}(E) \quad E_3 = \sigma_{MANV > "e6"}(E)$$

Quan hệ G được phân làm hai mảnh:

$$G_1 = \sigma_{MANV \leq "e3"}(G) \text{ và } G_2 = \sigma_{MANV > "e3"}(G).$$

Nhận xét:

- E_1 và G_1 được định nghĩa bởi cùng vị từ.
- Vị từ định nghĩa G_2 là hợp của các định nghĩa của những vị từ E_2 và E_3 .

Xét truy vấn

```
SELECT      *  
FROM        E, G  
WHERE       E.MANV=G.MANV
```


3. Xử lý truy vấn trong môi trường phân tán

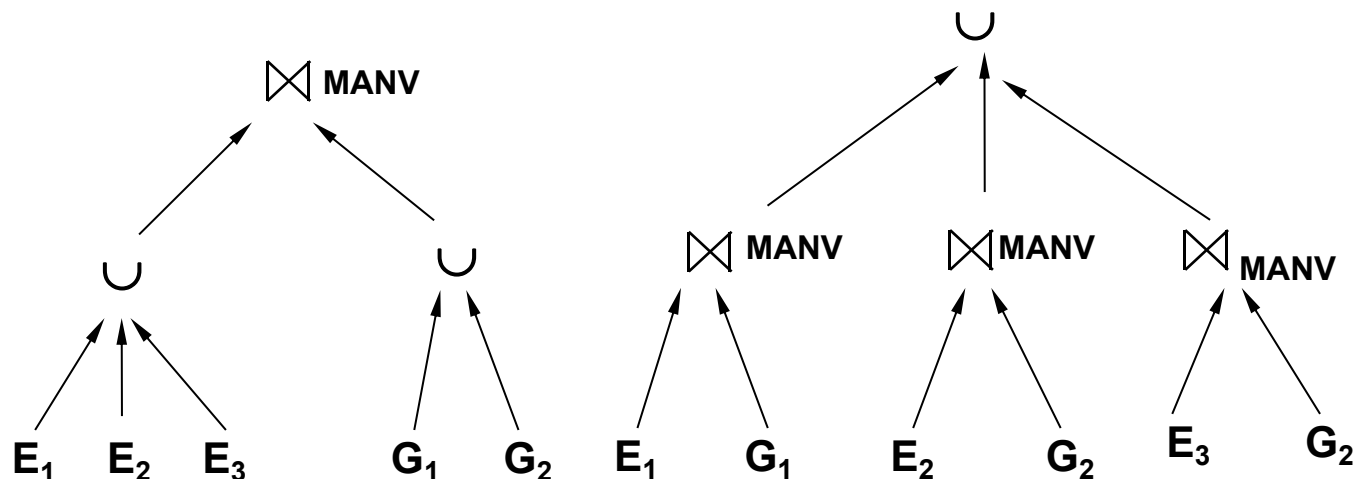
$$E_1 = \sigma_{MANV \leq "e3"}(E) \quad E_2 = \sigma_{"e3" < MANV \leq "e6"}(E) \quad E_3 = \sigma_{MANV > "e6"}(E)$$

$$G_1 = \sigma_{MANV \leq "e3"}(G) \quad G_2 = \sigma_{MANV > "e3"}(G).$$

$$E \bowtie G = (E_1 \cup E_2 \cup E_3) \bowtie (G_1 \cup G_2)$$

$$= (E_1 \bowtie G_1) \cup (E_1 \bowtie G_2) \cup (E_2 \bowtie G_1) \cup (E_2 \bowtie G_2) \cup (E_3 \bowtie G_1) \cup (E_3 \bowtie G_2)$$

$$= (E_1 \bowtie G_1) \cup (E_2 \bowtie G_2) \cup (E_3 \bowtie G_2)$$



(a) Truy vấn ban đầu

(b) Truy vấn rút gọn

Sự rút gọn phân mảnh ngang với phép nối

3. Xử lý truy vấn trong môi trường phân tán

3.2.2 Rút gọn phân mảnh dọc

- Chức năng của việc phân mảnh dọc là tách quan hệ dựa vào thuộc tính của các phép chiếu.
- Vì phép toán xây dựng lại đối với phân mảnh dọc là nối, nên chương trình định vị một quan hệ đã được phân mảnh dọc là nối của các mảnh trong vùng thuộc tính chung.

Ví dụ: Quan hệ E được phân mảnh dọc thành E_1 , E_2 , với thuộc tính khoá MANV được lặp lại như sau:

$$E_1 = \Pi_{\text{MANV}, \text{TENNV}}(E) \text{ và} \quad E_2 = \Pi_{\text{MANV}, \text{CHUCVU}}(E)$$

Chương trình định vị là: $E = E_1 \bowtie_{\text{MANV}} E_2$

- Các truy vấn trên phân mảnh dọc có thể rút gọn bằng cách xác định các **quan hệ trung gian vô ích** và loại bỏ các cây con chứa chúng.
- Các phép chiếu trên một phân mảnh dọc không có thuộc tính chung với các thuộc tính chiếu (ngoại trừ khóa của quan hệ) là vô ích, mặc dù các quan hệ là khác rỗng.

3. Xử lý truy vấn trong môi trường phân tán

Qui tắc 3:

$\Pi_{D,K}(R_i)$ là vô ích nếu $D \cap A' = \emptyset$.

Trong đó, quan hệ R xác định trên $A = \{A_1, \dots, A_n\}$; $R = \Pi_A(R)$, $A' \subseteq A$, K là khoá của quan hệ, $K \subseteq A$, D là tập các thuộc tính chiều, $D \subseteq A$.

- Dùng đại số quan hệ được tuyển chọn để định vị điều kiện chọn của các toán hạng của các phép kết;
- Thay thế cây con, bao gồm phép kết và các toán hạng của nó bằng quan hệ rỗng nếu điều kiện chọn của kết quả của phép kết bị mâu thuẫn.

3. Xử lý truy vấn trong môi trường phân tán

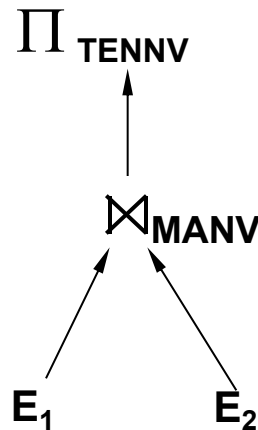
Ví dụ: Với quan hệ E được phân mảnh dọc như sau:

$$E_1 = \Pi_{\text{MANV}, \text{TENNV}}(E) \text{ và}$$

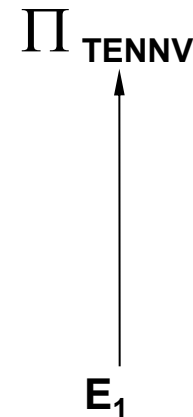
$$E_2 = \Pi_{\text{MANV}, \text{CHUCVU}}(E)$$

Xét truy vấn SQL:

SELECT	TENNV
FROM	E



(a) Truy vấn ban đầu



(b) Truy vấn rút gọn

Rút gọn đối với việc phân mảnh dọc

Nhận xét: phép chiếu trên E_2 là vô ích vì TENNV không có trong E_2 , nên phép chiếu chỉ cần gán vào E_1

3. Xử lý truy vấn trong môi trường phân tán

3.2.3 Rút gọn theo phân mảnh ngang dẫn xuất

- Sự phân mảnh ngang dẫn xuất là một cách tách hai quan hệ để việc xử lý nối của các phép chọn và phép nối
- Nếu quan hệ R phụ thuộc vào sự phân mảnh ngang dẫn xuất nhờ quan hệ S, thì các mảnh của R và S, mà có cùng giá trị thuộc tính nối sẽ được định vị tại cùng trạm. Ngoài ra, S có thể được phân mảnh tùy thuộc vào vị từ chọn.
- Khi các bộ của R được đặt tùy theo những bộ của S, thì sự phân mảnh dẫn xuất chỉ nên sử dụng mỗi quan hệ một nhiều từ $S \rightarrow R$ (i.e. với một bộ của S có thể phù hợp với n bộ của R, Nhưng với một bộ của R chỉ phù hợp với một bộ của S).
- Truy vấn trên các phân mảnh dẫn xuất cũng có thể rút gọn được, nếu các vị từ phân mảnh mâu thuẫn nhau thì phép nối sẽ đưa ra quan hệ rỗng.
- Chương trình định vị một quan hệ đã được phân mảnh ngang dẫn xuất là **hợp** của các mảnh.

3. Xử lý truy vấn trong môi trường phân tán

Ví dụ: Cho mỗi quan hệ một nhiều từ E đến G, quan hệ G (MANV, MADA, NHIEMVU, THOIGIAN) được phân mảnh ngang dẫn xuất theo E(MANV,TENNV,CHUCVU) như sau:

$$G_1 = G \bowtie_{\text{MANV}} E_1 \quad \text{và} \quad G_2 = G \bowtie_{\text{MANV}} E_2.$$

Trong đó E được phân mảnh ngang như sau:

$$E_1 = \sigma_{\text{CHUCVU}=\text{"Lập trình"}}(E) \quad \text{và} \quad E_2 = \sigma_{\text{CHUCVU} \neq \text{"Lập trình"}}(E)$$

Chương trình định vị cho một quan hệ đã được phân mảnh ngang dẫn xuất là hợp của các mảnh $G = G_1 \cup G_2$.

Để rút gọn các truy vấn trên phân mảnh gián tiếp này, phép nối sẽ đưa ra quan hệ rỗng nếu các vị từ phân mảnh mâu thuẫn nhau.

Ví dụ vị từ G_1 và E_2 mâu thuẫn nhau, nên $G_1 \bowtie E_2 = \phi$.

$$G_1 = G \bowtie_{\text{MANV}} E_1 \quad \text{và} \quad G_2 = G \bowtie_{\text{MANV}} E_2$$

$$E_1 = \sigma_{\text{CHUCVU}=\text{"Lập trình"}}(E) \quad \text{và} \quad E_2 = \sigma_{\text{CHUCVU} \neq \text{"Lập trình"}}(E)$$

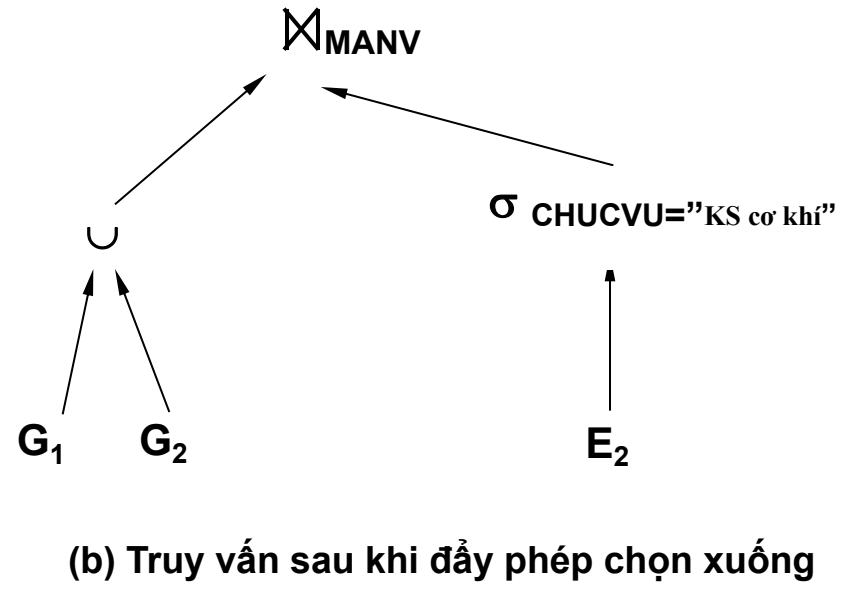
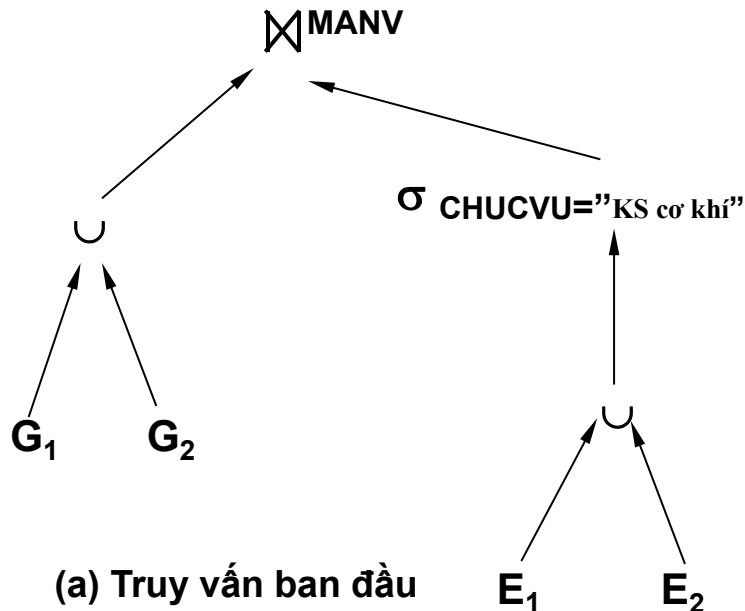
Ví dụ: Xét truy vấn

SELECT *

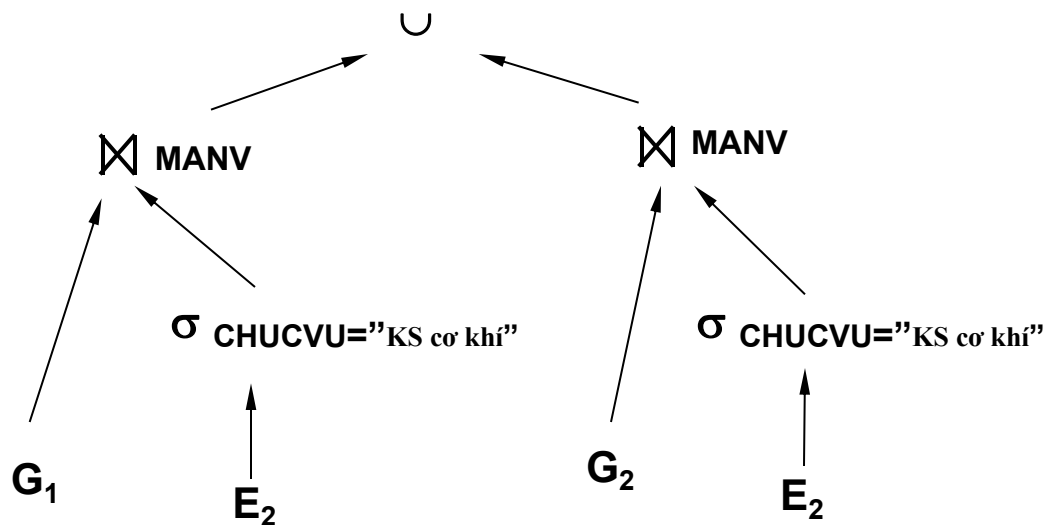
FROM E, G

WHERE G.MANV=E.MANV

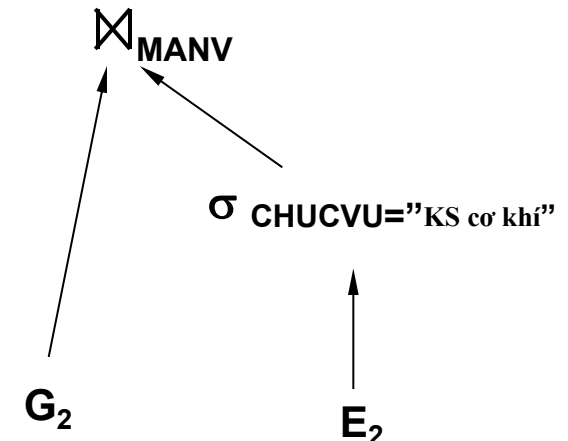
AND CHUCVU="KS cơ khí"



Chú ý: $(G_1 \cup G_2) \bowtie_{\sigma_{CHUCVU="ks cơ khí"}}(E_2)$
 $= (G_1 \bowtie_{\sigma_{CHUCVU="ks cơ khí"}}(E_2)) \cup (G_2 \bowtie_{\sigma_{CHUCVU="ks cơ khí"}}(E_2))$



(c) Truy vấn sau khi đẩy phép hợp lên



(d) Truy vấn đã rút gọn

Rút gọn của phân mảnh gián tiếp

Nhận xét:

- Truy vấn ban đầu trên các mảnh E_1 , E_2 , G_1 và G_2 tương ứng hình (a).
- Bằng cách đẩy phép chọn xuống các mảnh E_1 và E_2 , được truy vấn rút gọn ở hình (b).
- Phân phối các phép nối với phép hợp, chúng ta thu được cây hình (c).
- Cây con bên trái đưa ra một quan hệ rỗng, nên cây rút gọn có được trong hình (d).

3. Xử lý truy vấn trong môi trường phân tán

3.2.4 Rút gọn theo phân mảnh hỗn hợp

- Sự phân mảnh hỗn hợp là sự kết hợp giữa phân mảnh dọc và phân mảnh ngang.
- Mục đích của phân mảnh hỗn hợp là hỗ trợ các truy vấn liên quan đến phép chiếu, phép chọn và phép nối
- Chương trình định vị cho một quan hệ đã phân mảnh hỗn hợp là phép hợp và phép nối của các mảnh.

Ví dụ: Xét quan hệ E được phân mảnh hỗn hợp như sau:

$$E_1 = \sigma_{MANV \leq "e4"}(\Pi_{MANV, TENNV}(E)), E_2 = \sigma_{MANV > "e4"}(\Pi_{MANV, TENNV}(E))$$
$$E_3 = \Pi_{MANV, CHUCVU}(E)$$

Chương trình định vị là: $E = (E_1 \cup E_2) \bowtie_{MANV} E_3$

3. Xử lý truy vấn trong môi trường phân tán

Các truy vấn trên các mảnh hỗn hợp có thể được rút gọn bằng cách kết hợp các qui tắc sử dụng trong phân mảnh ngang nguyên thủy, phân mảnh dọc, phân mảnh ngang gián tiếp, tương ứng như sau:

1. **Loại bỏ các quan hệ rỗng** sinh bởi sự mâu thuẫn giữa các phép chọn trên các phân mảnh ngang.
2. **Loại bỏ các quan hệ vô ích** sinh bởi các phép chiếu trên các phân mảnh dọc.
3. **Phân phối các phép nối với các phép hợp** để tách và loại bỏ các phép nối vô ích.

Ví dụ: $E_1 = \sigma_{MANV \leq "e4"}(\Pi_{MANV, TENNV}(E))$,

$E_2 = \sigma_{MANV > "e4"}(\Pi_{MANV, TENNV}(E))$

$E_3 = \Pi_{MANV, CHUCVU}(E)$

SELECT
FROM
WHERE

TENNV

E

MANV="e5"

π TENNV

$\sigma_{MANV="e5"}$



E_1

E_2

E_3

(a) Truy vấn ban đầu

π TENNV

$\sigma_{MANV="e5"}$

E_2

(b) Truy vấn đã rút gọn

Rút gọn của phân mảnh hỗn hợp

Biến đổi truy vấn toàn cục thành truy vấn đoạn

- **Kỹ thuật:**

- Biểu diễn truy vấn toàn cục dưới dạng cây
- Thay thế mỗi quan hệ bằng biểu thức xây dựng lại:
 - Phân đoạn ngang: $R = R_1 \cup R_2 \cup \dots \cup R_n.$
 - Phân đoạn dọc: $R = R_1 * R_2 * \dots * R_n.$
- Biến đổi truy vấn: Sử dụng các phép biến đổi tương đương.
- Bỏ các nhánh của cây trong đó điều kiện truy vấn và điều kiện phân đoạn mâu thuẫn với nhau.

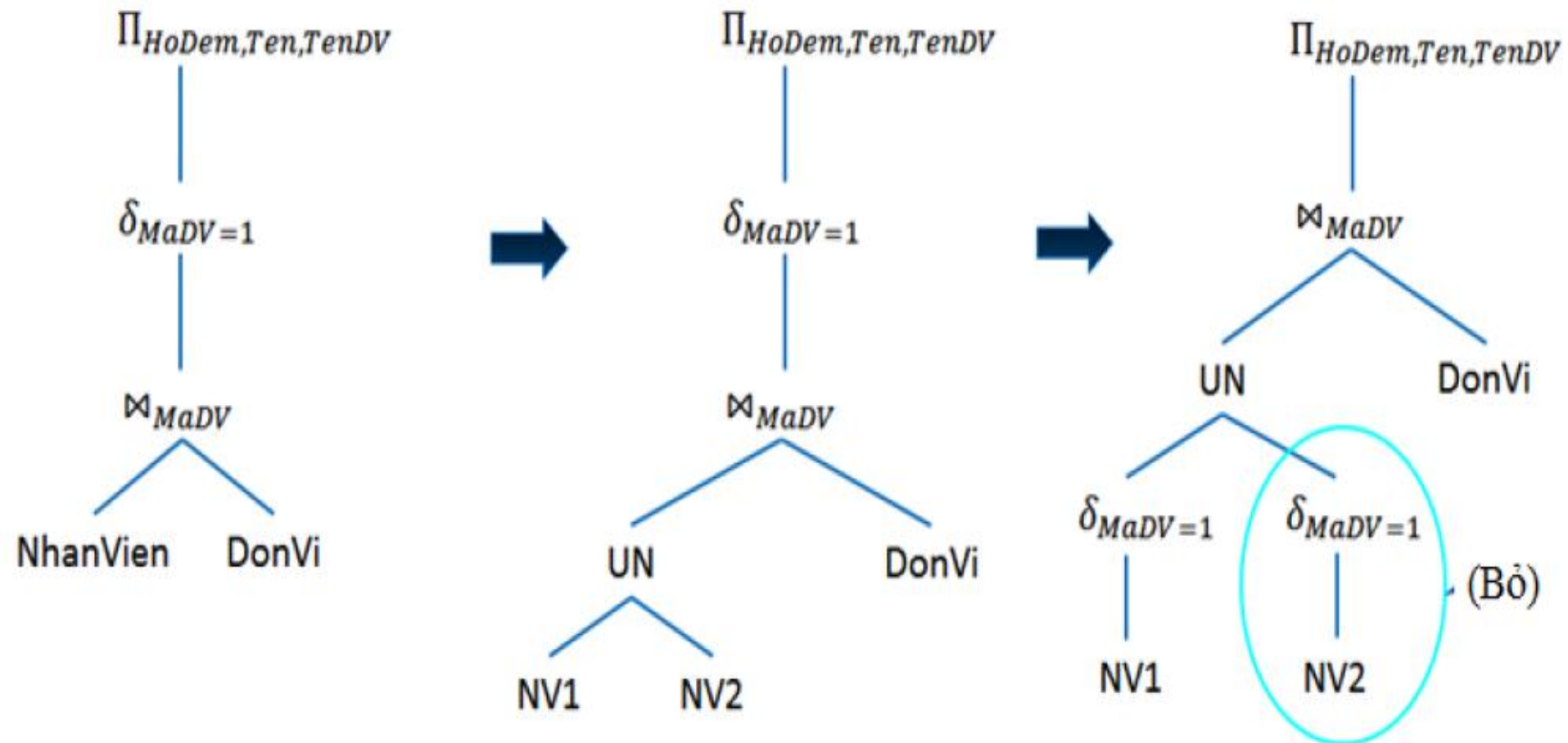
Biến đổi truy vấn toàn cục thành truy vấn đoạn

Ví dụ: Xét hai bảng NhanVien, DonVi:

- NhanVien(MaNV, HoDem, Ten, Luong, MaDV)
- DonVi(MaDV, TenDV, MaNQL)
- NhanVien được phân đoạn ngang làm 2 theo MaDV = 1, 2:
 - $NV1 = \delta_{MaDV=1}(NhanVien)$
 - $NV2 = \delta_{MaDV=2}(NhanVien)$
- Cho truy vấn toàn cục: Đưa ra họ đệm, tên, tên đơn vị của các nhân viên có MaDV = 1.

Biến đổi truy vấn toàn cục thành truy vấn đoạn

Truy vấn toàn cục Q: $\Pi_{HoDem, Ten, TenDV}(\delta_{MaDV=1}(NhanVien \bowtie_{MaDV} DonVi))$. Chuyển Q thành truy vấn đoạn:



Bài tập

- Cho các quan hệ toàn cục:

KhoHang (MaKH, TenKH, ViTri)

LoaiHang (MaLH, TenLH, MoTa, MaKH)

- Giả sử KhoHang, LoaiHang được phân đoạn ngang theo $\text{MaKH} = 1$ và $\text{MaKH} = 2$ thành KhoHang1, KhoHang2, LoaiHang1, LoaiHang2.
 1. Viết biểu thức đại số quan hệ của các đoạn trên.
 2. Viết biểu thức xây dựng lại của 2 quan hệ toàn cục KhoHang, LoaiHang.
 3. Hãy viết truy vấn toàn cục QTC: Lập danh sách gồm TenLH, ViTri của loại hàng có $\text{MaLH} = \text{"LH01"}$ và ở kho hàng có $\text{MaKH} = 2$.
 4. Chuyển truy vấn QTC ở trên thành truy vấn đoạn.

4. Tối ưu hóa truy vấn trong CSDL phân tán

- Trong hệ phân tán, truy vấn thu được từ **giai đoạn phân rã và định vị dữ liệu** có thể được thực hiện một cách đơn giản bằng việc thêm vào các thao tác truyền thông.
- Việc **hoán vị** thứ tự các phép toán trong một câu truy vấn có thể cung cấp nhiều chiến lược tương đương khác nhau.
- Bài toán xác định cây truy vấn tối ưu là NP-khó. Thông thường bộ tối ưu tìm một chiến lược **gần** tối ưu và tránh các chiến lược “tồi”.
- Đầu ra của bộ tối ưu là một lịch trình được tối ưu bao gồm truy vấn đại số được xác định trên các mảnh và các phép toán truyền thông hỗ trợ việc thực hiện truy vấn trên các trạm.
- Để chọn lựa được một chiến lược tối ưu nói chung, bộ tối ưu còn phải xác định **chi phí thực hiện** câu truy vấn.
- Chi phí thực hiện là tổ hợp có trọng số của **chi phí truyền thông**, **chi phí I/O** và **chi phí CPU**.

4. Tối ưu hóa truy vấn trong CSDL phân tán

4.1 Mô hình chi phí của bộ tối ưu hóa truy vấn

Chi phí của một chiến lược thực hiện phân tán có thể được biểu diễn hoặc theo **tổng chi phí** hoặc theo **thời gian trả lời**.

- **Tổng chi phí** là tổng của tất cả các thành phần chi phí. bao gồm **chi phí truyền thông**, **chi phí I/O** và **chi phí CPU**. Trong đó, chi phí truyền thông là quan trọng nhất.
- **Thời gian trả lời truy vấn** là thời gian được tính từ khi bắt đầu xử lý đến khi hoàn thành truy vấn.

4. Tối ưu hoá truy vấn trong CSDL phân tán

2. Công thức chung cho sự xác định tổng chi phí:

Tổng chi phí: tổng của tất cả các chi phí C_{CPU} , $C_{\text{I/O}}$, C_{MSG}

$$\text{Total_cost} = C_{\text{CPU}} * \#instr + C_{\text{I/O}} * \#I/O_s + C_{\text{MSG}} * \#msg_s + C_{\text{TR}} * \#bytes$$

Trong đó: **Total_cost**: tổng chi phí

C_{CPU} : chi phí của một lệnh CPU

$C_{\text{I/O}}$: chi phí của một xuất/nhập đĩa

C_{MSG} : chi phí của việc khởi đầu và nhận một thông báo.

C_{TR} : chi phí truyền một đơn vị dữ liệu từ trạm này đến trạm khác, ta xem **C_{TR}** như là một hằng số.

$\#instr$: tổng tất cả các lệnh CPU ở các trạm

$\#I/O_s$: số lần xuất/nhập đĩa

$\#msg_s$: số thông báo

$\#bytes$: tổng kích thước của các thông báo.

4. Tối ưu hoá truy vấn trong CSDL phân tán

Chú ý: Trong công thức:

$$\text{Total_cost} = C_{\text{CPU}} * \# \text{instr} + C_{\text{I/O}} * \# \text{I/O} + C_{\text{MSG}} * \# \text{msgs} + C_{\text{TR}} * \# \text{bytes}$$

- Hai thành phần chi phí đầu ($C_{\text{CPU}}, C_{\text{I/O}}$) là *chi phí địa phương*.
- Hai thành phần chi phí sau ($C_{\text{MSG}}, C_{\text{TR}}$) là *chi phí truyền thông*.
- Chi phí truyền thông để chuyển #byte dữ liệu từ trạm này đến trạm khác được giả thiết là một hàm tuyến tính theo số #bytes được truyền đi, được xác định bởi công thức

$$\text{CC}(\# \text{byte}) = C_{\text{MSG}} + C_{\text{TR}} * \text{bytes}$$

4. Tối ưu hóa truy vấn trong CSDL phân tán

1. Công thức chung cho sự xác định thời gian trả lời

$$\text{Response_time} = C_{\text{CPU}} * \text{seq_\#insts} + C_{\text{I/O}} * \text{seq_\#I/O}_s + C_{\text{MSG}} * \text{seq_\#msg}_s + C_{\text{TR}} * \text{seq_\#bytes}$$

Trong đó:

seq_#x (x có thể là số lệnh của CPU, I/O, số thông báo, số byte) là số lớn nhất của x khi thực hiện truy vấn một cách tuần tự.

Trong đó: **Response_time**: thời gian trả lời truy vấn

C_{CPU}: chi phí của một lệnh CPU

C_{I/O}: chi phí của một xuất/nhập đĩa

C_{MSG}: chi phí của việc khởi đầu và nhận một thông báo.

C_{TR}: chi phí truyền một đơn vị dữ liệu từ trạm này đến trạm khác

#insts: tổng tất cả các lệnh CPU ở các trạm

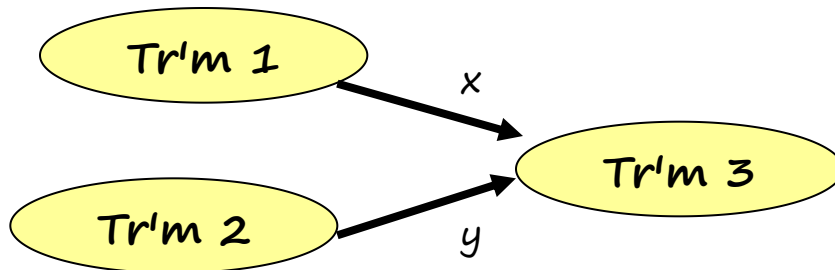
#I/O_s: số lần xuất/nhập đĩa

#msg_s: số thông báo

#bytes: tổng kích thước của tất cả các thông báo.

4. Tối ưu hoá truy vấn trong CSDL phân tán

Ví dụ: Minh hoạ sự khác nhau giữa *tổng chi phí* và *thời gian trả lời*, trong đó máy tính trả lời truy vấn tại trạm 3 với dữ liệu từ trạm 1 và 2, ở đây chỉ có chi phí truyền thông được xét



Hình 4.12: Ví dụ của sự biến đổi truy vấn

Giả sử, C_{MSG} và C_{TR} được biểu thị theo đơn vị thời gian.

Tổng chi phí truyền x đơn vị từ trạm 1 đến trạm 3 và y đơn vị từ trạm 2 đến trạm 3 là:

$$\text{Total_cost} = C_{MSG} + C_{TR} * x + C_{MSG} + C_{TR} * y = 2C_{MSG} + C_{TR} * (x+y)$$

Với việc truyền dữ liệu có thể được thực hiện song song nên thời gian trả lời của truy vấn là:

$$\text{Response_time} = \max\{C_{MSG} + C_{TR} * x, C_{MSG} + C_{TR} * y\}$$

4. Tối ưu hoá truy vấn trong CSDL phân tán

4.2 Các thông kê dữ liệu

- Yếu tố chính ảnh hưởng đến hiệu suất của một chiến lược thực thi là kích thước của các quan hệ trung gian sinh ra trong quá trình thực hiện.
- Khi phép toán tiếp theo đặt tại một trạm khác, quan hệ trung gian phải được truyền trên mạng.
- Do đó để tối thiểu hoá khối lượng dữ liệu truyền đi, điều quan tâm đầu tiên là đánh giá kích thước kết quả trung gian của các phép toán đại số quan hệ.
- Đánh giá này dựa trên các thông tin thống kê về các quan hệ cơ sở và các công thức ước tính lực lượng của kết quả các phép toán quan hệ.

4. Tối ưu hoá truy vấn trong CSDL phân tán

Mục đích của thống kê dữ liệu:

- Xác định kích thước của các quan hệ trung gian sinh ra trong quá trình thực hiện câu truy vấn
- Xác định chi phí truyền thông cho các đại lượng trung gian

Một số ký hiệu

Cho quan hệ R xác định trên tập thuộc tính $A=\{A_1, \dots, A_n\}$.
 R được phân mảnh thành R_1, R_2, \dots, R_r .

- length(A_i)**: độ dài (byte) của thuộc tính A_i , $A_i \in A$,
- card($\pi_{A_i}(R_j)$)**: lực lượng của phép chiếu của mảnh R_j lên thuộc tính A_i (số giá trị phân biệt trên thuộc tính A_i).
- max(A_i)**: giá trị cực đại của thuộc tính A_i trong $\text{Dom}(A_i)$
- min(A_i)**: giá trị cực tiểu của thuộc tính A_i trong $\text{Dom}(A_i)$
- card(dom(A_i))**: lực lượng của thuộc tính A_i
- card(R_i)**: số các bộ trong mảnh R_i

4. Tối ưu hoá truy vấn trong CSDL phân tán

Ngoài ra, dữ liệu thống kê cũng bao gồm *hệ số chọn của phép nối* ($\mathbf{SF_J}$) đối với một số cặp đại số quan hệ, hệ số $\mathbf{SF_J}$ của quan hệ R và S là một số thực giữa 0 và 1, được xác định bởi:

$$\mathbf{SF_J} = \frac{\mathbf{card(R \bowtie S)}}{\mathbf{card(R) * card(S)}}$$

- Hệ số $\mathbf{SF_J}$ nhỏ thì phép nối có tính chọn tốt, ngược lại có tính chọn tồi.
- Các thống kê này có lợi để đánh giá kích thước của quan hệ trung gian.
- Kích thước một quan hệ trung gian R được xác định bởi $\mathbf{size(R) = card(R) * length(R)}$. Trong đó,
 - + $\mathbf{length(R)}$ là độ dài (số byte) của mỗi bộ trong R, được tính theo độ dài các thuộc tính của nó,
 - + $\mathbf{card(R)}$ là số các bộ của R được tính theo công thức ở phần tiếp theo.

4. Tối ưu hoá truy vấn trong CSDL phân tán

4.3 Lực lượng của các kết quả trung gian

☞ Các công thức để ước tính lực lượng kết quả các phép toán cơ sở của đại số quan hệ (chọn, chiếu, tích Decartes, nối, nửa nối, hợp và hiệu).

☞ Các toán hạng quan hệ được ký hiệu bởi R và S.

☞ Hệ số chọn của một phép toán SF_{OP} , (OP biểu thị phép toán) là tỷ lệ giữa các bộ của một toán hạng quan hệ tham gia vào kết quả của phép toán với số các bộ của quan hệ .

Ví dụ:

- SF_J : hệ số chọn của phép nối
- SF_S : hệ số chọn của phép chọn

4. Tối ưu hoá truy vấn trong CSDL phân tán

(1). Phép chọn: $\text{card}(\sigma(R))$?

$$\text{card}(\sigma(R)) = \text{SF}_S(F) * \text{card}(R)$$

Trong đó $\text{SF}_S(F)$ phụ thuộc vào công thức chọn và có thể tính như sau, với $p(A_i)$, $p(A_j)$ là các vị từ tương ứng với các thuộc tính A_i , A_j .

$$\text{SF}_S(A=\text{value}) = \frac{1}{\text{Card}(\pi_A(R))}$$

$$\text{SF}_S(A > \text{value}) = \frac{\max(A) - \text{value}}{\max(A) - \min(A)}$$

$$\text{SF}_S(A < \text{value}) = \frac{\text{Value} - \min(A)}{\max(A) - \min(A)}$$

$$\text{SF}_S(p(A_i) \wedge p(A_j)) = \text{SF}_S(p(A_i)) * \text{SF}_S(p(A_j))$$

$$\text{SF}_S(p(A_i) \vee p(A_j)) = \text{SF}_S(p(A_i)) + \text{SF}_S(p(A_j)) - \text{SF}_S(p(A_i)) * \text{SF}_S(p(A_j))$$

$$\text{SF}_S(A \in \{\text{value}\}) = \text{SF}_S(A=\text{value}) * \text{card}(\{\text{value}\})$$

4. Tối ưu hoá truy vấn trong CSDL phân tán

(2). Phép chiếu: $\text{card}(\pi_A(R))$?

- ☞ Phép chiếu có thể có hoặc không loại bỏ các bản sao. Ở đây chỉ xét phép chiếu loại bỏ các bản sao.
- ☞ Lực lượng quan hệ kết quả của một phép chiếu tùy ý là khó đánh giá chính xác, vì tương quan giữa thuộc tính chiếu thường không biết.
- ☞ Tuy nhiên, có hai trường hợp tầm thường nhưng đặc biệt có lợi:
 - ☞ Nếu phép chiếu của R trên một thuộc tính đơn A thì $\text{card}(\pi_A(R)) = \text{card}(R)$.
 - ☞ Nếu một trong các thuộc tính chiếu là khoá của R, thì $\text{card}(\pi_A(R)) = \text{card}(R)$ và $\text{card}(R \times S) = \text{card}(R) * \text{card}(S)$

4. Tối ưu hoá truy vấn trong CSDL phân tán

(3). Phép nối: $\text{card}(R \bowtie S)$?

☞ Không có một cách tổng quát để xác định lực lượng của một phép nối nếu không có các thông tin thêm.

☞ Cận trên của lực lượng của phép nối chính là lực lượng của tích Decartes.

☞ Tuy nhiên, có một số trường hợp xuất hiện thường xuyên và việc đánh giá là đơn giản:

☞ - Nếu $R \bowtie_{AB} S$ với $A \in R$, $B \in S$, trong đó A là khoá của R , B là khoá ngoài của S , thì lực lượng của kết quả xấp xỉ là:
 $\text{card}(R \bowtie_{AB} S) = \text{card}(R)$

☞ Với các phép nối khác, lực lượng của kết quả là:

☞ $\text{card}(R \bowtie S) = SF_J * \text{card}(R) * \text{card}(S)$

4. Tối ưu hoá truy vấn trong CSDL phân tán

(4). Phép nửa nối

Hệ số chọn của phép nửa nối (SF_{SJ}) xấp xỉ là:

$$SF_{SJ}(R \bowtie S) = \frac{\text{Card}(\pi_A(S))}{\text{Card}(\text{dom}[A])}$$

Công thức này chỉ phụ thuộc vào thuộc tính A của S, nên thường được gọi là hệ số chọn thuộc tính A của S, ký hiệu $SF_{SJ}(S.A)$ và là hệ số chọn của S.A trên bất cứ thuộc tính nối khác. Vì thế, lực lượng của phép nối được tính như sau:

$$\text{card}(R \bowtie AS) = SF_{SJ}(S.A) * \text{card}(R)$$

4. Tối ưu hóa trong CSDL phân tán

(5). Phép hợp

- Rất khó đánh giá số lượng của $R \cup S$, vì các bộ giống nhau giữa R và S bị loại bỏ bởi phép hợp.
- Ở đây chúng ta chỉ đưa ra công thức tính:
 - cận trên của $\text{card}(R \cup S)$ bằng $\text{card}(R) + \text{card}(S)$,
 - cận dưới của $\text{card}(R \cup S)$ bằng $\max\{\text{card}(R), \text{card}(S)\}$ (giả sử R và S không chứa các bộ lặp).

(6). Phép trừ

Cũng như phép hợp ở đây chỉ đưa ra cận trên và cận dưới, cận trên của $\text{card}(R - S)$ là $\text{card}(R)$, cận dưới là 0.

Về việc ít sử dụng tài nguyên?

4. Tối ưu hóa trong CSDL phân tán

Ví dụ Xét hai quan hệ trong cơ sở dữ liệu của công ty máy tính: E=NHANVIEN (MANV, TENNV, CHUCVU) và G=HOSO (MANV, MADA, NHIEMVU, THOIGIAN).

Với câu truy vấn “Cho biết tên các nhân viên hiện đang quản lý một dự án”. Ta có câu truy vấn SQL tương ứng là:

```
SELECT      TENNV
FROM        E, G
WHERE       E.MANV=G.MANV
           AND NHIEMVU="Quản lý"
```

Hai truy vấn đại số tương đương với truy vấn trên là:

$$\Pi_{TENNV}(\sigma_{NHIEMVU="Quản lý" \wedge E.MANV=G.MANV} (E \times G)) \quad (1)$$

$$\text{và } \Pi_{TENNV}(E \bowtie_{MANV}(\sigma_{NHIEMVU="Quản lý"} G)) \quad (2)$$

Rõ ràng truy vấn (2) tránh được khỏi phải tích số của E và G, nên dùng ít phép tính tài nguyên hơn truy vấn (1)

Câu hỏi cuối chương

1. Mục đích của tối ưu hoá truy vấn trong CSDL phân tán
2. Chức năng của tối ưu hoá truy vấn phân tán
3. Các phương pháp xử lý truy vấn cơ bản
4. Ý tưởng của thuật toán Ingres. Ví dụ
5. Sơ đồ phân lớp chung cho xử lý truy vấn phân tán.
6. Cách chuyển một truy vấn phép tính quan hệ thành một cây đại số quan hệ
7. Sử dụng các luật biến đổi phép toán đại số quan hệ để biến đổi cây đại số quan hệ thành các cây tương đương.
8. Định vị dữ liệu phân tán-Tối ưu hoá cục bộ
9. Xác định tổng chi phí và thời gian trả lời của một câu truy vấn.