

# Chương 3:

# Thiết kế cơ sở dữ liệu phân tán

Thời lượng: 12 tiết

GV: ThS. Thái Bảo Trần

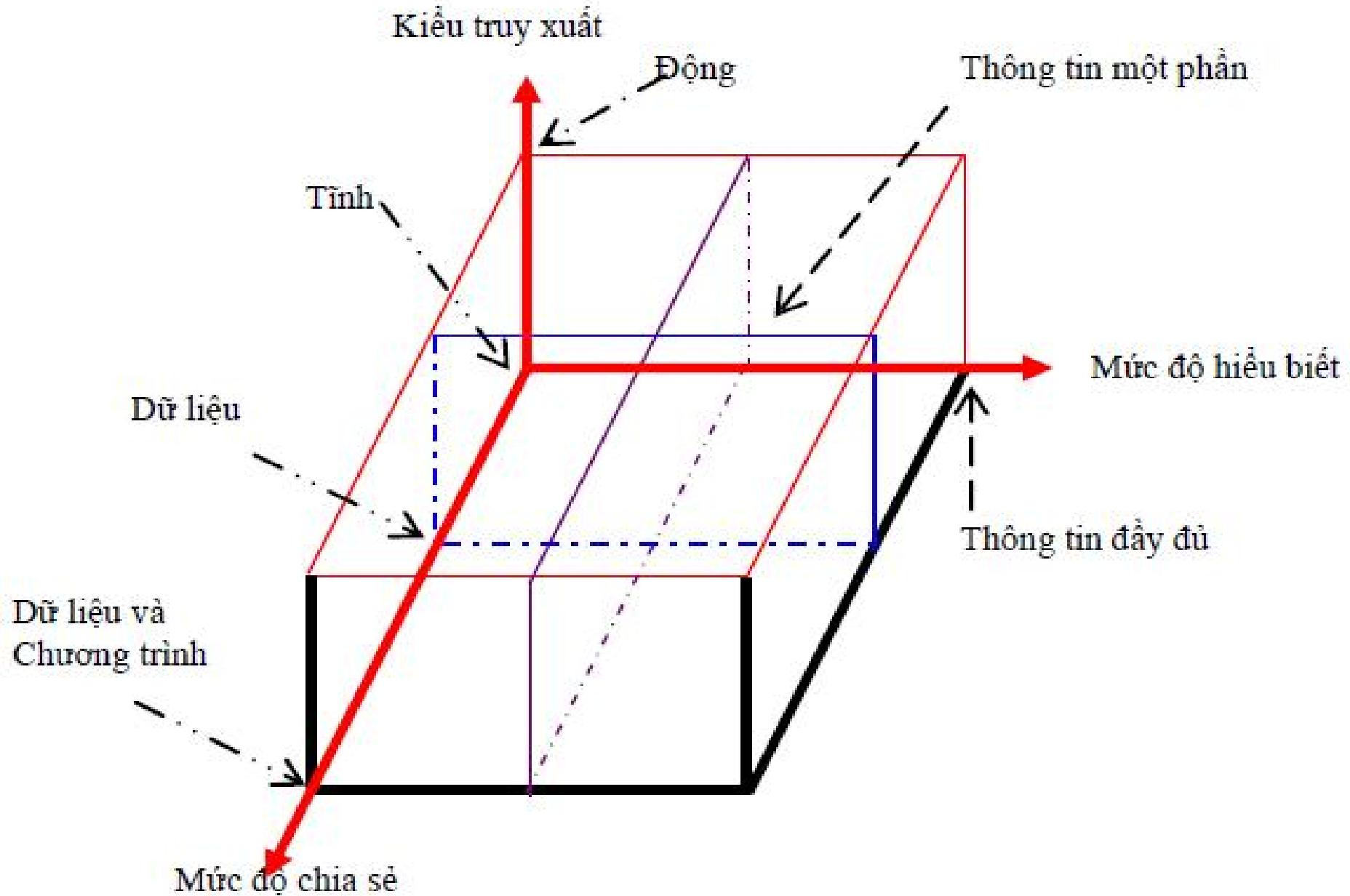
# Nội dung

- 1. Giới thiệu**
- 2. Mục tiêu của thiết kế CSDL phân tán**
- 3. Các bước thiết kế CSDL phân tán**
- 4. Các chiến lược phân tán dữ liệu**
- 5. Các phương pháp thiết kế CSDL phân tán**
- 6. Thiết kế phân mảnh**
- 7. Thiết kế định vị dữ liệu**
- 8. Cấp phát tài nguyên trong hệ phân tán**

# GIỚI THIỆU

- ▶ **Thiết kế hệ thống một máy tính phân tán** cần phải chọn những **vị trí đặt dữ liệu và các chương trình trên một mạng máy tính**.
- ▶ Phải qua bước phân tích trước khi thiết kế. Các bước này phải độc lập với mọi giải pháp cài đặt.
- ▶ Đối với hệ quản trị CSDL phân tán, việc phân tán ứng dụng đòi hỏi hai vấn đề:
  - Phân tán DBMS
  - Phân tán các chương trình ứng dụng chạy trên DBMS đó.

# Bộ khung phân tán

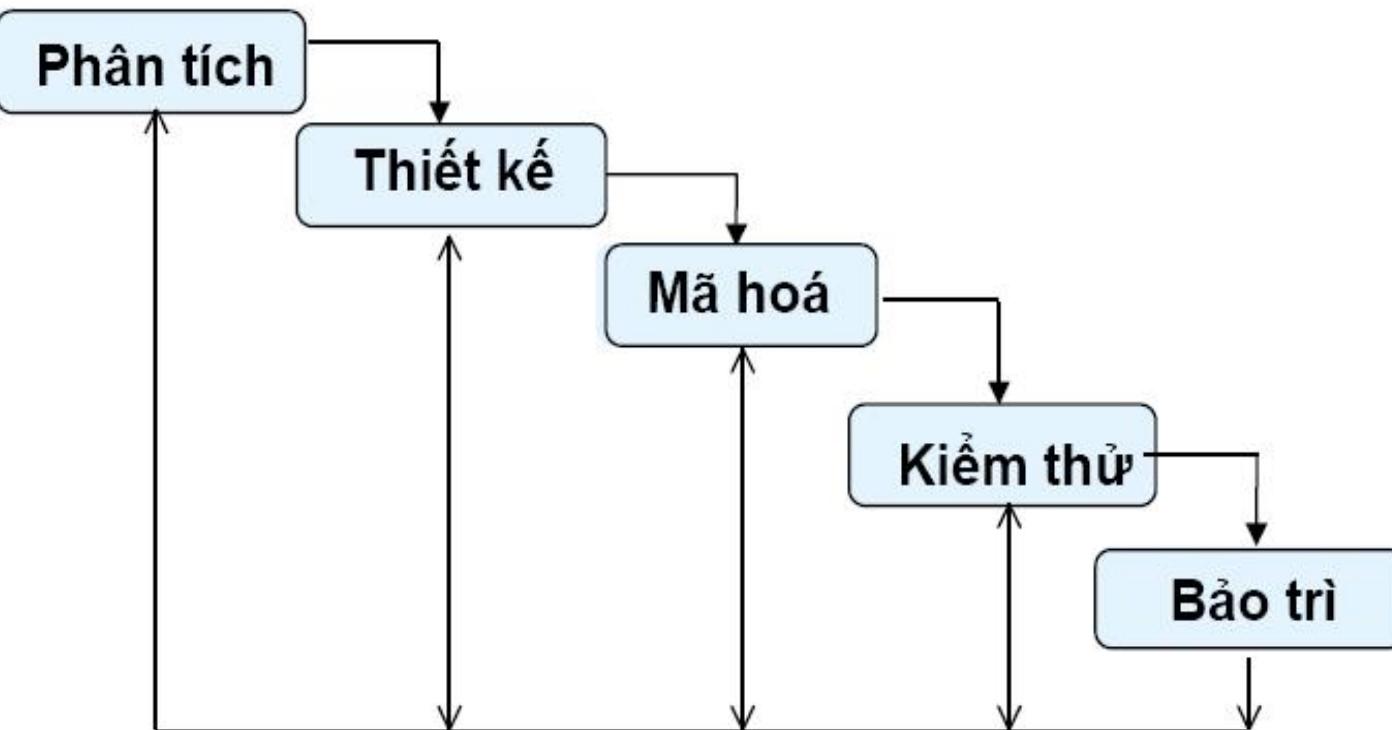


# **GIỚI THIỆU**

- Có nhiều điểm tương đồng với việc thiết kế hệ thống tập trung.
- Điều khác nhau cơ bản là hệ thống được phân bố trên một số địa điểm khác nhau.
- Tính khả thi, chu kỳ sống, tính mở, tính sẵn sàng,...
- Thiết kế phân cứng: máy trạm, máy chủ, mạng ,...
- Ví dụ:

# MÔ HÌNH THÁC NƯỚC

(Water Fall Model)



# GIỚI THIỆU

## 1. Các công việc cần phải làm để thiết kế HT phân tán:

- Xác định kiến trúc mô hình phân tán tổng thể
- Định vị các địa phương cần phân tán, loại hình phân tán sử dụng cho mỗi địa phương (tổn bộ, bản sao, lai,...).
- Tiến hành cân đối các yếu tố được phân tán bao gồm các phần tử dữ liệu và các hoạt động xử lý trên mỗi trạm.
- Thiết kế cơ sở dữ liệu phân tán.
- Thiết kế các chương trình ứng dụng.

# GIỚI THIỆU

## 2. Các sản phẩm yêu cầu sau khi phân tích thiết kế

### ❖ Mô tả các trạm

- . Thông tin địa lý
- . Thiết bị vật lý
- . Thông tin hạ tầng
- . Đặc trưng về con người (trình độ, kỹ năng,...)

### ❖ Mô tả về sử dụng dữ liệu cho mỗi trạm

- . Các phần tử dữ liệu sử dụng từ hệ thống
- . Các phần tử dữ liệu cần phải tạo ra
- . Các phần tử dữ liệu cập nhật

Các phần tử cần xóa

# GIỚI THIỆU

## 2. Các sản phẩm yêu cầu sau khi phân tích thiết kế (tiếp)

### ❖ Mô tả quá trình nghiệp vụ cho mỗi trạm

- Danh sách các xử lý (sơ đồ chức năng) ở các trạm
- Mô tả các xử lý

### ❖ Các thỏa thuận về phương án kiến trúc hệ thống cho mỗi trạm, cho nhu cầu về dữ liệu và xử lý của trạm đó

- Có cần hay không về các trợ giúp không phải kỹ thuật
- Có cần hay không về hệ thống địa phương, về nối mạng
- Có cần hay không về các cấu hình phân tán khác

# Nội dung

- 1. Giới thiệu**
- 2. Mục tiêu của thiết kế CSDL phân tán**
- 3. Các bước thiết kế CSDL phân tán**
- 4. Các chiến lược phân tán dữ liệu**
- 5. Các phương pháp thiết kế CSDL phân tán**
- 6. Thiết kế phân mảnh**
- 7. Thiết kế định vị dữ liệu**
- 8. Cấp phát tài nguyên trong hệ phân tán**

# Mục tiêu của thiết kế CSDL phân tán

- ❖ **Tính cục bộ xử lý**

- ▶ *processing locality*
- ▶ Phân tán dữ liệu để làm **cực đại hóa** tính cục bộ xử lý là đặt dữ liệu càng gần các ứng dụng sử dụng các dữ liệu này càng tốt.
- ▶ Một quan hệ không là một đơn vị phân tán.
- ▶ Tính cục bộ xử lý dựa vào các tham chiếu cục bộ và các tham chiếu từ xa.
- ▶ **Tính cục bộ hoàn toàn (*complete locality*)**.

- ❖ **Tính sẵn sàng và độ tin cậy của dữ liệu**

- ▶ **Tính sẵn sàng (*availability*)**.
- ▶ **Độ tin cậy (*reliability*)**.

# Mục tiêu của thiết kế CSDL phân tán

## ❖ Điều phối tải làm việc

- ▶ Cực đại hóa mức độ thực hiện song song các ứng dụng.
- ▶ Điều phối tải làm việc có thể ảnh hưởng ngược lại với tính cục bộ xử lý.
- ▶ Tính đồng thời nội truy vấn.

## ❖ Chi phí lưu trữ và khả năng lưu trữ có sẵn

- ▶ Khả năng lưu trữ có sẵn tại mỗi nơi.
- ▶ Chi phí lưu trữ dữ liệu là không đáng kể so với các chi phí CPU, nhập / xuất và truyền thông của các ứng dụng.

# Nội dung

- 1. Giới thiệu**
- 2. Mục tiêu của thiết kế CSDL phân tán**
- 3. Các bước thiết kế CSDL phân tán**
- 4. Các chiến lược phân tán dữ liệu**
- 5. Các phương pháp thiết kế CSDL phân tán**
- 6. Thiết kế phân mảnh**
- 7. Thiết kế định vị dữ liệu**
- 8. Cấp phát tài nguyên trong hệ phân tán**

# Các bước thiết kế CSDL phân tán

## ❖ Thiết kế CSDL tập trung

- ▶ Thiết kế lược đồ ý niệm.
- ▶ Thiết kế CSDL vật lý.

## ❖ Thiết kế CSDL phân tán

- ▶ Thiết kế lược đồ toàn cục.
- ▶ Thiết kế phân mảnh.
- ▶ Thiết kế định vị mảnh.
- ▶ Thiết kế CSDL vật lý cục bộ.

# Các bước thiết kế CSDL phân tán

- ❖ Thiết kế CSDL phân tán: cần phải hiểu biết thật chính xác về các yêu cầu của ứng dụng, nhất là đối với các ứng dụng *quan trọng hơn*.
- ❖ Cần quan tâm đến:
  - ▶ Nơi chạy ứng dụng.
  - ▶ Tần suất chạy ứng dụng.
  - ▶ Số lượng, loại và sự phân tán của các truy xuất trong mỗi ứng dụng đến mỗi đối tượng dữ liệu cần thiết.

# Nội dung

- 1. Giới thiệu**
- 2. Mục tiêu của thiết kế CSDL phân tán**
- 3. Các bước thiết kế CSDL phân tán**
- 4. Các chiến lược phân tán dữ liệu**
- 5. Các phương pháp thiết kế CSDL phân tán**
- 6. Thiết kế phân mảnh**
- 7. Thiết kế định vị dữ liệu**
- 8. Cấp phát tài nguyên trong hệ phân tán**

# Các chiến lược phân tán dữ liệu

- ❖ Việc định vị và phân tán dữ liệu ở các nút trong một mạng máy tính sẽ quyết định tính hiệu quả và đúng đắn của hệ thống phân tán.
- ❖ Có 4 chiến lược phân tán dữ liệu cơ bản:
  - Tập trung dữ liệu
  - Chia nhỏ dữ liệu
  - Nhân bản dữ liệu
  - Tổ chức lại

# Các chiến lược phân tán dữ liệu

## 1. Tập trung dữ liệu:

Tất cả các dữ liệu được tập trung một chỗ. Cách này đơn giản nhưng có 3 nhược điểm:

- Dữ liệu không sẵn sàng cho người sử dụng truy nhập từ xa
- Chi phí truyền thông lớn, thường làm cực đại việc truy nhập dữ liệu tới nơi tập trung.
- Toàn bộ hệ thống ngừng khi cơ sở dữ liệu bị sự cố

## 2. Chia nhỏ dữ liệu:

- Cơ sở dữ liệu được chia thành các phần nhỏ liên kết nhau (không trùng lặp).
- Mỗi phần dữ liệu được đưa đến các trạm một cách thích hợp

# Các chiến lược phân tán dữ liệu

## 3. Nhân bản dữ liệu:

- CSDL được nhân thành nhiều bản **tùng phần** hoặc **đầy đủ** và được đặt ở nhiều trạm trên mạng.
- Nếu bản sao của CSDL được lưu giữ tại mọi trạm của hệ thống ta có trường hợp **nhân bản đầy đủ**.
- Hiện nay có nhiều kỹ thuật mới cho phép tạo bản sao không đầy đủ phù hợp với yêu cầu dữ liệu ở mỗi trạm và một bản đầy đủ được quản lý ở server.
- Sau một khoảng thời gian nhất định các bản sao được làm đồng bộ với bản chính bằng một ứng dụng nào đó.

# Các chiến lược phân tán dữ liệu

## 4. Tổ chức lại:

- Cơ sở dữ liệu được phân thành nhiều phần: **quan trọng** và **không quan trọng**.
- Phần ít quan trọng được lưu giữ một nơi
- Phần quan trọng được lưu trữ ở nhiều nơi khác.

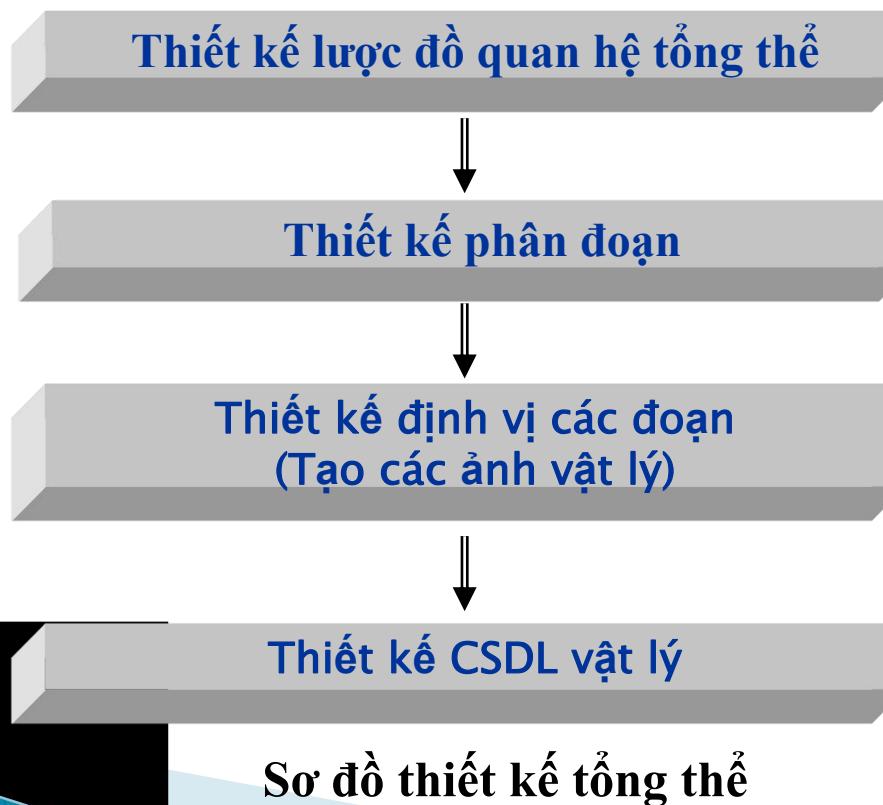
# Nội dung

- 1. Giới thiệu**
- 2. Mục tiêu của thiết kế CSDL phân tán**
- 3. Các bước thiết kế CSDL phân tán**
- 4. Các chiến lược phân tán dữ liệu**
- 5. Các phương pháp thiết kế CSDL phân tán**
- 6. Thiết kế phân mảnh**
- 7. Thiết kế định vị dữ liệu**
- 8. Cấp phát tài nguyên trong hệ phân tán**

# CÁC PHƯƠNG PHÁP THIẾT KẾ CSDL PHÂN TÁN

## ❑ Sơ đồ thiết kế tổng thể cơ sở dữ liệu phân tán

Hiện nay chưa có một kỹ thuật cụ thể nào nói một cách chi tiết việc thiết kế một CSDL phân tán. Tuy nhiên, một cách tổng quát chúng ta có thể thiết kế CSDL phân tán theo các bước sau:



# CÁC PHƯƠNG PHÁP THIẾT KẾ CSDL PHÂN TÁN

## ❖ *Thiết kế lược đồ quan hệ tổng thể:*

- Thiết kế các quan hệ tổng thể
- Mô tả toàn bộ dữ liệu sẽ được dùng trong hệ thống

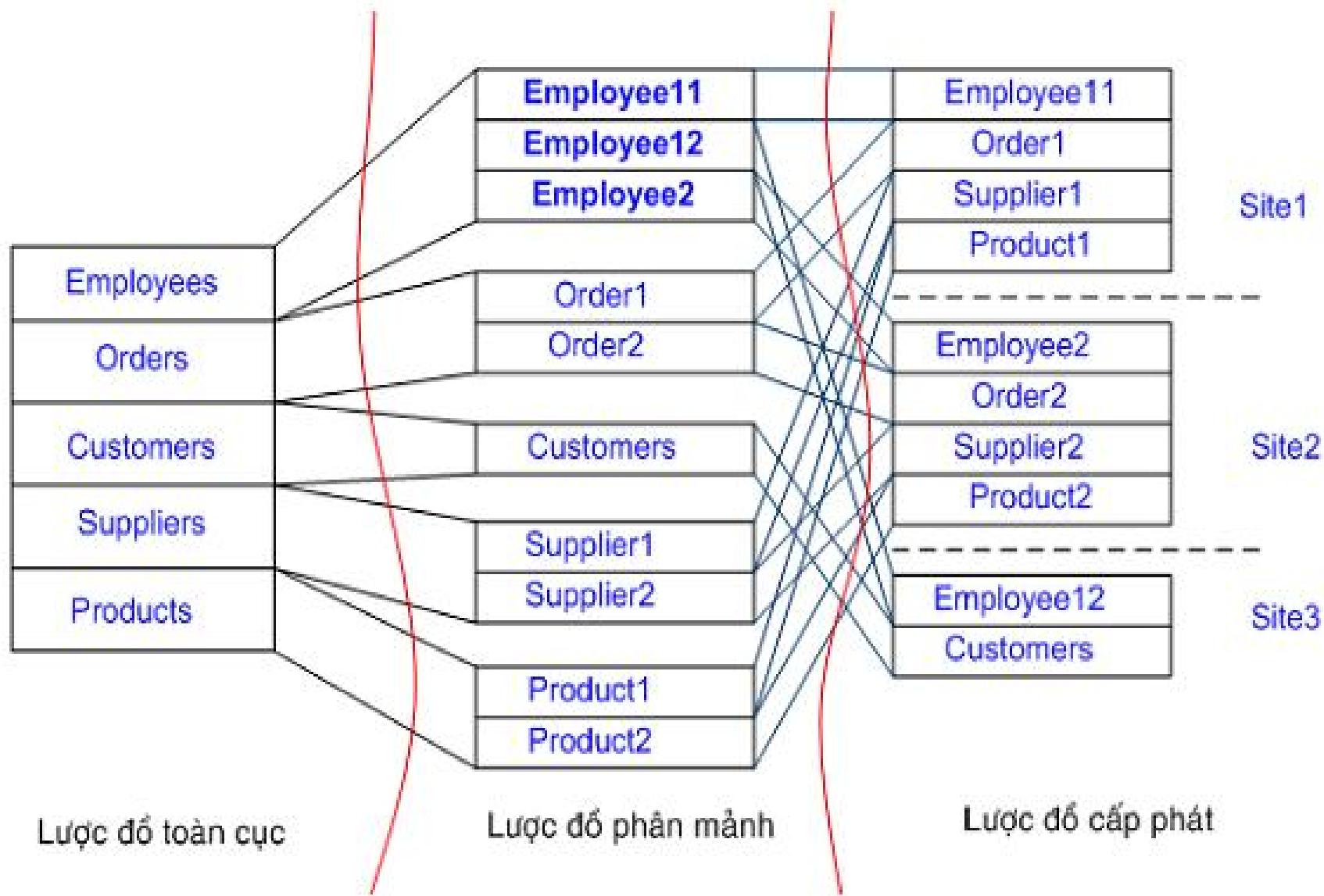
## ❖ *Thiết kế phân đoạn:* thực hiện chia nhỏ dữ liệu thành các phần.

## ❖ *Thiết kế định vị các đoạn:*

- Là quá trình thực hiện ánh xạ các đoạn vào các trạm khác nhau
- Tạo các ảnh vật lý tại các trạm.
- Các đoạn dữ liệu được đưa vào các vị trí lưu trữ thích hợp với yêu cầu hoạt động thực tế của hệ thống.

## ❖ *Thiết kế cơ sở dữ liệu vật lý:* thiết kế dữ liệu vật lý cho các quan hệ tại các trạm.

# Kiến trúc tham khảo



# CÁC PHƯƠNG PHÁP THIẾT KẾ CSDL PHÂN TÁN

❑ Có 2 phương pháp thiết kế CSDL phân tán

1. Phương pháp tiếp cận từ trên xuống
2. Phương pháp tiếp cận từ dưới lên.

# CÁC PHƯƠNG PHÁP THIẾT KẾ CSDL PHÂN TÁN

## 1. Phương pháp thiết kế từ trên xuống

- Thiết kế từ tổng thể đến riêng biệt
- Phân rã một hệ thống lớn thành các hệ thống con
- Phân tích các yêu cầu nhằm định nghĩa môi trường hệ thống
- Thu thập các yêu cầu về dữ liệu và nhu cầu xử lý của các trạm có sử dụng CSDL.

# CÁC PHƯƠNG PHÁP THIẾT KẾ CSDL PHÂN TÁN

## ❖ Thiết kế từ trên xuống

- ▶ *top-down design*
- ▶ Thiết kế lược đồ toàn cục.
- ▶ Thiết kế phân mảnh CSDL.
- ▶ Định vị các mảnh tại các nơi.
- ▶ Thiết kế dữ liệu vật lý đặt tại mỗi nơi.

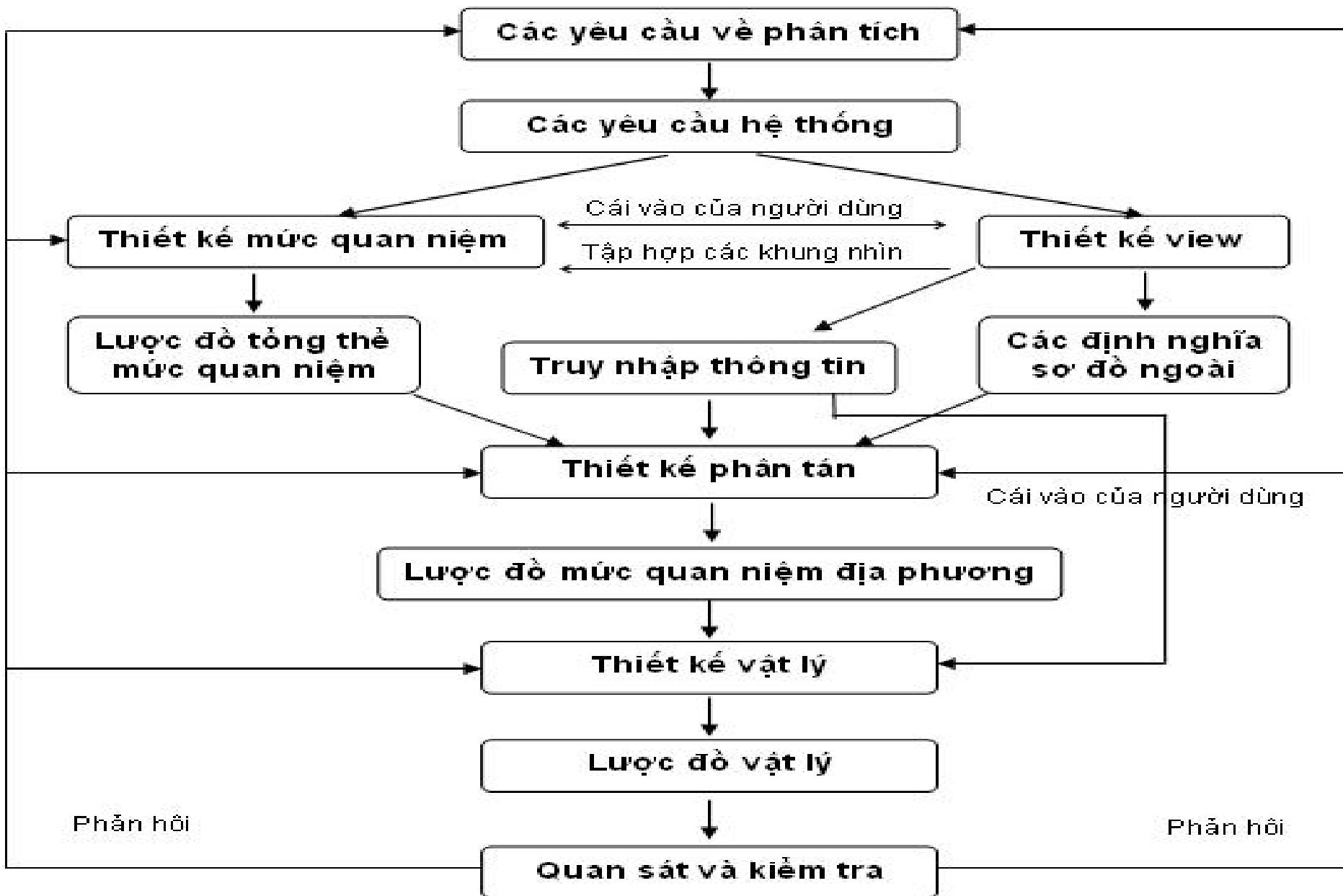
# CÁC PHƯƠNG PHÁP THIẾT KẾ CSDL PHÂN TÁN

- ❖ *Thiết kế view:* xây dựng khung nhìn dữ liệu cho người sử dụng ở các trạm.
- ❖ *Thiết kế mức quan niệm:* là một tiến trình kiểm tra và xác định rõ hai nhóm quan hệ: *phân tích thực thể* và *phân tích chức năng*.
  - + *Phân tích thực thể:* xác định các tập thực thể, các thuộc tính và các mối quan hệ giữa chúng.
  - + *Phân tích chức năng:* xác định các chức năng của hệ thống và đưa ra các chức năng cơ sở.

# CÁC PHƯƠNG PHÁP THIẾT KẾ CSDL PHÂN TÁN

- ❖ *Thiết kế phân tán*: bao gồm hai phần:
  - + *Thiết kế phân đoạn*
  - + *Thiết kế định vị*
- ❖ *Thiết kế lược đồ quan niệm địa phương*: tạo ra các lược đồ mức quan niệm tại các địa phương
- ❖ *Thiết kế vật lý*: thực hiện ánh xạ lược đồ mức quan niệm tại các địa phương ra các đơn vị lưu trữ vật lý
- ❖ *Quan sát và kiểm tra*: kiểm tra các giai đoạn của quá trình thiết kế cơ sở dữ liệu

# CÁC PHƯƠNG PHÁP THIẾT KẾ CSDL PHÂN TÁN



# CÁC PHƯƠNG PHÁP THIẾT KẾ CSDL PHÂN TÁN

## 2. Phương pháp thiết kế từ dưới lên

- Phương pháp thiết kế trên xuống thực sự có hiệu quả khi xây dựng một hệ thống mới.
- Trong thực tế, một số CSDL đã tồn tại trước, được tổ chức trong môi trường tập trung và CSDL phân tán được phát triển bằng cách liên kết chúng lại thành một CSDL mới thống nhất (Các DBMS địa phương khác nhau đã được sử dụng)

### *Cách thiết kế*

- A1. Chọn một mô hình dữ liệu chung để mô tả lược đồ tổng thể
- A2. Chuyển mỗi lược đồ địa phương theo mô hình dữ liệu chung đã chọn
- A3. Tích hợp các lược đồ địa phương vào lược đồ tổng thể

# Nội dung

- 1. Giới thiệu**
- 2. Mục tiêu của thiết kế CSDL phân tán**
- 3. Các bước thiết kế CSDL phân tán**
- 4. Các chiến lược phân tán dữ liệu**
- 5. Các phương pháp thiết kế CSDL phân tán**
- 6. Thiết kế phân mảnh**
- 7. Thiết kế định vị dữ liệu**
- 8. Cấp phát tài nguyên trong hệ phân tán**

# Thiết kế phân mảnh

- ▶ Việc phân tán dữ liệu được thực hiện trên cơ sở cấp phát các tập tin cho các nút trên một mạng máy tính. Các nút mạng thường nằm ở các vị trí địa lý khác nhau trải rộng trên một diện tích lớn. Do vậy để tối ưu việc khai thác thông tin thì dữ liệu không thể để tập trung mà phải **phân tán trên các nút của mạng**.
- ▶ Một quan hệ không phải là một đơn vị truy xuất dữ liệu tốt nhất.

Ví dụ: nếu ứng dụng được thực hiện trên một bộ phận nhỏ các dữ liệu của QH mà QH đó nằm tại các vị trí khác nhau thì có thể gây ra những truy xuất thừa và hơn thế việc nhân bản các quan hệ làm tốn không gian bộ nhớ.

# Thiết kế phân mảnh

- ▶ Do vậy phân rã một quan hệ thành nhiều mảnh, mỗi mảnh được xử lý như một đơn vị sẽ cho phép thực hiện nhiều giao dịch đồng thời. Một câu truy vấn ban đầu có thể được chia ra thành một tập các truy vấn con, các truy vấn này có thể được thực hiện song song trên các mảnh sẽ giúp cải thiện tốc độ hoạt động của hệ thống.
- ▶ **Thể hiện của các quan hệ chính là các bảng**, vì thế vấn đề là tìm những cách khác nhau để chia một bảng thành nhiều bảng nhỏ hơn.
- ▶ **Phân mảnh quan hệ là gì?**  
Việc chia một quan hệ thành nhiều quan hệ nhỏ hơn được gọi là phân mảnh quan hệ.

# Thiết kế phân mảnh

- ▶ **Có hai phương pháp khác nhau:** Chia bảng theo chiều dọc và chia bảng theo chiều ngang.
- **Chia dọc** ta được các quan hệ con mà mỗi quan hệ chứa một tập con các thuộc tính của quan hệ gốc – gọi là phân mảnh dọc.
- **Chia ngang** một quan hệ ta được các quan hệ con mà mỗi quan hệ chứa một số bộ của quan hệ gốc – gọi là phân mảnh ngang.
- Ngoài ra còn có một khả năng **hỗn hợp**, đó là phân mảnh kết hợp cách phân mảnh ngang và dọc.

# Thiết kế phân mảnh

- ▶ Khi thiết kế các hệ thống CSDL phân tán người ta thường tập trung xoay quanh các câu hỏi?
  - Tại sao lại cần phải phân mảnh?
  - Làm thế nào để thực hiện phân mảnh?
  - Phân mảnh nên thực hiện đến mức độ nào?
  - Có cách gì kiểm tra tính đúng đắn của việc phân mảnh?
  - Các mảnh sẽ được cấp phát trên mạng như thế nào?
  - Những thông tin nào sẽ cần thiết cho việc phân mảnh và cấp phát?

# Các lý do phân mảnh

- ❖ Khung nhìn hoặc đơn vị truy xuất của các ứng dụng không phải là toàn bộ quan hệ mà thường là một phần quan hệ.
- ❖ Việc phân rã một quan hệ thành nhiều mảnh, mỗi mảnh được xử lý như một đơn vị, sẽ cho phép thực hiện nhiều giao dịch đồng thời.
- ❖ Việc phân mảnh các quan hệ sẽ cho phép thực hiện song song một câu vấn tin bằng cách chia nó ra thành một tập các câu vấn tin con hoạt tác trên các mảnh.

# Các lý do phân mảnh

- ❖ Nếu các ứng dụng có các khung nhìn được định nghĩa trên một quan hệ cho trước nằm tại những vị trí khác thì có hai cách chọn lựa đơn vị phân tán:
  - + hoặc là toàn bộ quan hệ
  - + hoặc quan hệ được lưu ở một vị trí có chạy ứng dụng.
- ❖ Nhận xét: Chọn lựa thứ nhất gây ra một số lượng lớn các truy xuất không cần thiết đến dữ liệu ở xa. Chọn lựa sau sẽ gây ra nhiều vấn đề khi cập nhật và lãng phí không

# KHUYẾT ĐIỂM CỦA VIỆC PHÂN MẢNH

- ❖ Nếu ứng dụng có những yêu cầu ngăn cản việc phân rã thành các mảnh để được sử dụng độc quyền, thì những ứng dụng có các khung nhìn được định nghĩa trên nhiều mảnh sẽ bị giảm hiệu suất hoạt động.
- ❖ Nếu một khung nhìn đòi hỏi thông tin ở nhiều mảnh thì việc truy xuất dữ liệu để nối lại sẽ có chi phí cao.
- ❖ Kiểm soát ngữ nghĩa dữ liệu (semantic data control): Do kết quả của phân mảnh, các thuộc tính tham gia vào một phụ thuộc có thể bị phân rã vào các mảnh khác nhau và được cấp phát cho những vị trí khác nhau. Trong trường hợp này, một nhiệm vụ đơn giản như kiểm tra các phụ thuộc cũng phải thực hiện truy cập dữ liệu từ nhiều vị trí.

# Các loại phân mảnh dữ liệu

- ❖ Phân mảnh ngang (horizontal fragmentation)
- ❖ Phân mảnh dọc (vertical fragmentation).
- ❖ Phân mảnh hỗn hợp (hibrid fragmentation)

*Chú ý: Quá trình phân mảnh phải được gắn liền với vấn đề cấp phát dữ liệu và bài toán cụ thể như thế nào.*

# Các qui tắc phân mảnh đúng đắn

- ❖ Việc phân mảnh một quan hệ tổng thể cũng phải tuân theo một số quy tắc nhất định để khi tái thiết lại quan hệ cũ vẫn bảo đảm ngũ nghĩa của nó.
- ❖ Một phương pháp thiết kế các phân mảnh đúng đắn phải thỏa mãn ba tính chất sau:
  - a. *Tính đầy đủ (completeness)*
  - b. *Tính tái tạo được (reconstruction)*
  - c. *Tính tách biệt (disjointness)*

# Các qui tắc phân mảnh đúng đắn

a. **Tính đầy đủ:** Nếu một quan hệ R được phân rã thành các mảnh  $R_1, R_2, \dots, R_k$  thì mỗi mục dữ liệu có trong R phải có trong ít nhất một mảnh  $R_i$  nào đó.

b. **Tính tái tạo được:**

- Nếu một quan hệ R được phân rã thành các mảnh  $R_1, R_2, \dots, R_k$  thì phải tồn tại một toán tử  $\theta$  sao cho  $R = \theta(R_i), \forall i$ .
- Toán tử  $\theta$  thay đổi tùy theo từng loại phân mảnh.
- Trong thực tế khi các mảnh được phân mảnh ngang thì  $\theta$  là phép hợp, phân mảnh dọc thì  $\theta$  là phép nối  
nối hợp thì  $\theta$  là phép nửa nối .

# Các qui tắc phân mảnh đúng đắn

## c. *Tính tách biệt:*

- Nếu một quan hệ R được phân mảnh ngang thành các quan hệ  $R_1, R_2, \dots, R_k$  và mục dữ liệu  $t_i$  nằm trong mảnh  $R_i$  thì nó sẽ không nằm trong một mảnh  $R_k$ ,  $k \neq i$ .
- Tiêu chuẩn này bảo đảm các mảnh ngang phải được tách rời nhau.
- Nếu quan hệ được phân mảnh dọc thì thuộc tính chung phải được lặp lại trong mỗi mảnh. Do đó, trong trường hợp phân mảnh dọc tính tách biệt chỉ được định nghĩa trên các trường không phải là thuộc tính chung của quan hệ.

## Các điều kiện đúng đắn

- ❖ Quan hệ  $R$  được phân rã thành các mảnh  $R_1, R_2, \dots, R_n$
- ❖ Điều kiện đầy đủ
  - ▶ completeness condition
  - ▶ Mỗi mục dữ liệu trong  $R$  phải có trong một hoặc nhiều mảnh  $R_i$
  - ▶ Phân mảnh ngang:  
$$\forall u \in R, \exists i \in [1, n]: u \in R_i$$
  - ▶ Phân mảnh dọc:  
$$\forall A \in Attr(R), \exists i \in [1, n]: A \in Attr(R_i)$$

# Các điều kiện đúng đắn

## ❖ Điều kiện tái tạo

- ▶ *reconstruction condition*
- ▶ Luôn luôn có thể xác định một phép toán quan hệ  $\nabla$  sao cho:

$$R = \nabla R_i, \forall R_i \in F_R \text{ với } F_R = \{R_1, R_2, \dots, R_n\}$$

## ▶ Phân mảnh ngang:

$$R = R_1 \cup R_2 \cup \dots \cup R_n$$

## ▶ Phân mảnh dọc:

$$R = R_1 \triangleright \triangleleft R_2 \triangleright \triangleleft \dots \triangleright \triangleleft R_n$$

# Các điều kiện đúng đắn

## ❖ Điều kiện tách biệt

- ▶ *disjointness condition*
- ▶ Nếu mục dữ liệu  $d_i$  có trong  $R_i$  thì nó không có trong bất kỳ mảnh  $R_k$  khác ( $k \neq i$ ).
- ▶ **Phân mảnh ngang:**

$$\forall i \neq k \text{ và } i, k \in [1, n]: R_i \cap R_k = \emptyset$$

hoặc

$$\forall u \in R_i, \forall i \neq k \text{ và } i, k \in [1, n]: u \notin R_k$$

# Các loại phân mảnh

## ❖ Phân mảnh ngang

- ▶ Horizontal fragmentation
- ▶ Phân mảnh ngang chính (primary horizontal fragmentation)
- ▶ Phân mảnh ngang dẫn xuất (derived horizontal fragmentation)

# PHÂN MẢNH NGANG

❖ Phân mảnh ngang một quan hệ tổng thể n-bộ R là tách R thành các quan hệ con n-bộ  $R_1, R_2, \dots, R_k$  sao cho quan hệ R có thể được khôi phục lại từ các quan hệ con này bằng phép hợp:

$$R = R_1 \cup R_2 \cup \dots \cup R_k$$

❖ Có hai loại phân mảnh ngang:

➤ **Phân mảnh ngang chính** (primary horizontal fragmentation): phân mảnh ngang chính của một quan hệ được thực hiện dựa trên các **vị từ** được định nghĩa trên quan hệ đó.

➤ **Phân mảnh ngang dẫn xuất** (derived horizontal fragmentation): phân mảnh ngang dẫn xuất của một quan hệ được thực hiện dựa trên các **vị từ** được định nghĩa trên quan hệ khác.

Như vậy, trong phân mảnh ngang tập các **vị từ** đóng một vai trò quan trọng.

# VỊ TỪ

- ❖ Cho lược đồ quan hệ  $R(U)$ ,  $U = A_1, A_2, \dots, A_N$  trong đó mỗi  $A_i$  là một thuộc tính có miền giá trị  $\text{dom}(A_i)$ .
- ❖ Một vị từ đơn giản  $P$  được định nghĩa trên  $R$  có dạng:

**P:  $A_i \theta <\!\!\text{giá trị}>$**

Trong đó  $\theta \in \{=, <, \leq, \geq, >, \neq\}$ ,  $A_i$  là một thuộc tính,  $<\!\!\text{giá trị}> \in \text{dom}(A_i)$ .

Như vậy cho trước một lược đồ  $R$ , nếu các miền giá trị  $\text{dom}(A_i)$  là hữu hạn chúng ta có thể xác định được tập tất cả các vị từ đơn giản trên  $R$ .

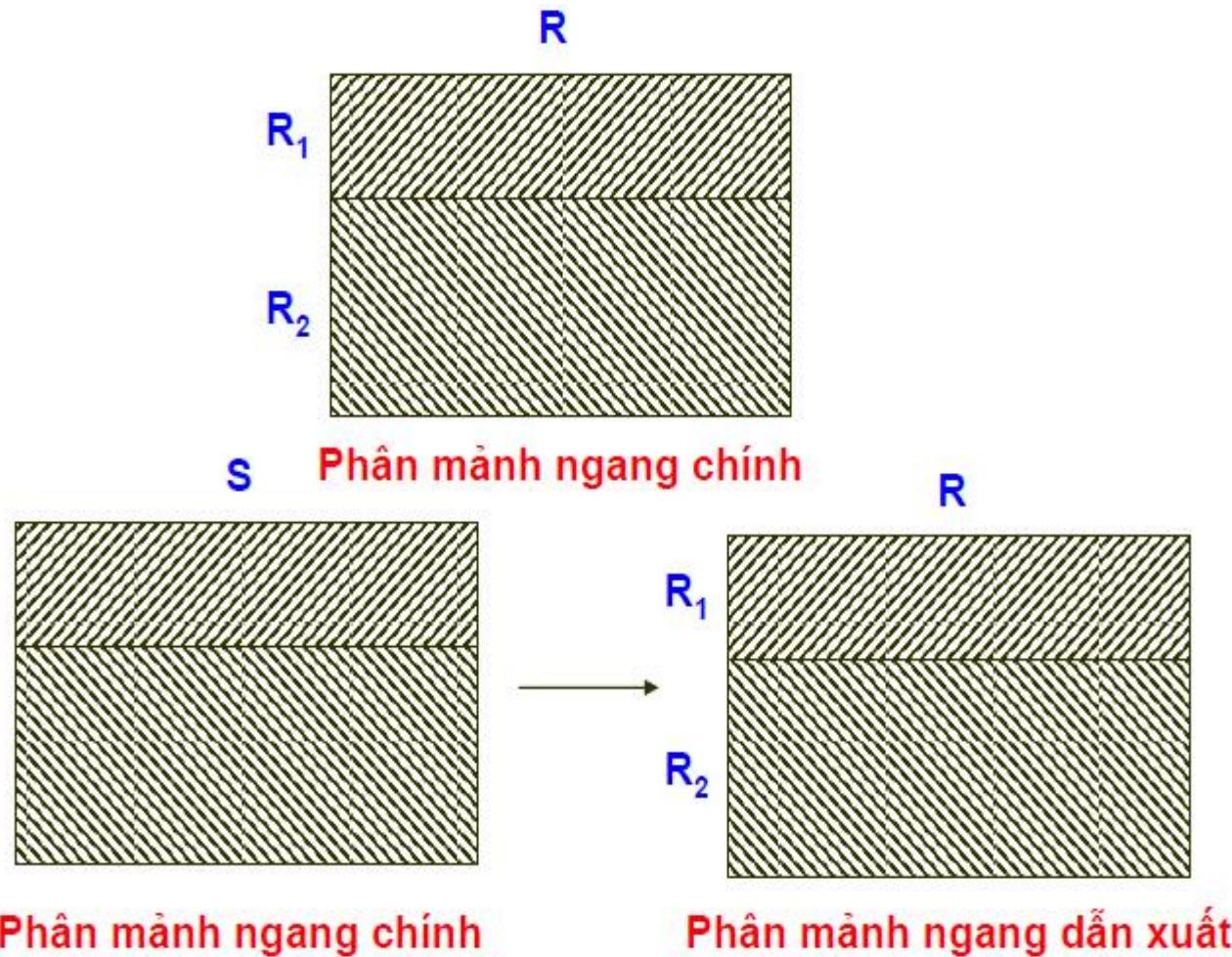
**Ví dụ:** Các vị từ đơn giản của quan hệ:

DuAn(MaDA, TenDA, Kinhphi,...):

P1: TenDA = “Bảo trì”

KinhPhi  $\leq 200000$

# Các loại phân mảnh ngang



## Phân mảnh ngang chính

- ❖ *Phân mảnh ngang chính (primary horizontal fragmentation)* là sự phân chia các bộ của một quan hệ toàn cục thành các tập hợp con dựa vào các thuộc tính của quan hệ này, mỗi tập hợp con được gọi là *mảnh ngang (horizontal fragment)*.
- ❖ Mỗi mảnh ngang được tạo bởi một phép chọn trên quan hệ toàn cục.
- ❖ *Vị từ định tính (qualification)* của mảnh ngang.

## Example.DDB

### ❖ Lược đồ toàn cục:

emp (empnum, name, sal, tax, mgrnum, deptnum)

dept (deptnum, name, area, mgrnum)

supplier (snum, name, city)

supply (snum, pnum, deptnum, quan)

### ❖ Lược đồ phân mảnh:

$\text{emp}_1 = \sigma_{\text{deptnum} \leq 10} \Pi_{\text{empnum, name, mgrnum, deptnum}} \text{emp}$

$\text{emp}_2 = \sigma_{10 < \text{deptnum} \leq 20} \Pi_{\text{empnum, name, mgrnum, deptnum}} \text{emp}$

$\text{emp}_3 = \sigma_{\text{deptnum} > 20} \Pi_{\text{empnum, name, mgrnum, deptnum}} \text{emp}$

$\text{emp}_4 = \Pi_{\text{empnum, name, sal, tax}} \text{emp}$

# Phân mảng ngang chính

## ❖ Ví dụ

### ▶ Quan hệ toàn cục:

supplier (snum, name, city)

### ▶ Các mảng ngang:

$\text{supplier}_1 = \sigma_{\text{city} = 'SF'} \text{ supplier}$

$\text{supplier}_2 = \sigma_{\text{city} = 'LA'} \text{ supplier}$

### ▶ Các vị từ định tính:

$q_1: \text{city} = 'SF'$

$q_2: \text{city} = 'LA'$

### ▶ Xét các điều kiện đúng đắn

## Phân mảnh ngang chính

---

- ❖ Điều kiện đầy đủ: tập hợp các vị từ định tính của tất cả các mảnh ngang phải đầy đủ.
- ❖ Điều kiện tái tạo: phép hợp.
- ❖ Điều kiện tách biệt: các vị từ định tính phải loại trừ nhau.

Ví dụ: KhachHang (MaKH, HoTen, NgaySinh, SoDT, DiaChi)

- $KH_1 = \delta_{DiaChi='Hà Nội'}(KhachHang)$
- $KH_2 = \delta_{DiaChi='Hải Phòng'}(KhachHang)$
- $KH_3 = \delta_{DiaChi='Tp. HCM'}(KhachHang)$

Đảm bảo điều kiện phân đoạn đúng đắn:

- Điều kiện đầy đủ: Mọi bộ đều thuộc một đoạn.
- Điều kiện xây dựng lại:  $KhachHang = KH_1 \cup KH_2 \cup KH_3$
- Điều kiện tách rời:  $KH_i \cap KH_j = \emptyset, \forall i \neq j$ .

## Phân mảnh ngang dẫn xuất

- ❖ *Phân mảnh ngang dẫn xuất* (*derived horizontal fragmentation*) là sự phân chia các bộ của một quan hệ toàn cục thành các tập hợp con, được gọi là các mảnh ngang, dựa vào phân mảnh ngang của một quan hệ khác (được gọi là quan hệ chủ).
- ❖ Vị từ định tính của mảnh ngang dẫn xuất bao gồm *điều kiện kết* và *vị từ định tính của mảnh ngang chủ* tương ứng.

## Example.DDB

### ❖ Lược đồ toàn cục:

emp (empnum, name, sal, tax, mgrnum, deptnum)

dept (deptnum, name, area, mgrnum)

supplier (snum, name, city)

supply (snum, pnum, deptnum, quan)

### ❖ Lược đồ phân mảnh:

$\text{emp}_1 = \sigma_{\text{deptnum} \leq 10} \Pi_{\text{empnum, name, mgrnum, deptnum}} \text{emp}$

$\text{emp}_2 = \sigma_{10 < \text{deptnum} \leq 20} \Pi_{\text{empnum, name, mgrnum, deptnum}} \text{emp}$

$\text{emp}_3 = \sigma_{\text{deptnum} > 20} \Pi_{\text{empnum, name, mgrnum, deptnum}} \text{emp}$

$\text{emp}_4 = \Pi_{\text{empnum, name, sal, tax}} \text{emp}$

# Phân mảng ngang dẫn xuất

## ❖ Ví dụ

### ► Quan hệ toàn cục:

supply (snum, pnum, deptnum, quan)

### ► Các mảng ngang dẫn xuất:

supply<sub>1</sub> = supply >< snum = snum supplier<sub>1</sub>

supply<sub>2</sub> = supply >< snum = snum supplier<sub>2</sub>

### ► Các vị từ định tính:

q<sub>1</sub>: supply.snum = supplier.snum AND  
supplier.city = 'SF'

q<sub>2</sub>: supply.snum = supplier.snum AND  
supplier.city = 'LA'

### ► Xét các điều kiện đúng đắn

# Phân mảng ngang dẫn xuất

- ▶ Với  $\times$  là phép toán nửa kết (Semi Join)

$$R \times_F S = \Pi_A (R \bowtie_F S)$$

- ▶ Trong đó:

- F là điều kiện
  - A là tập các thuộc tính của R

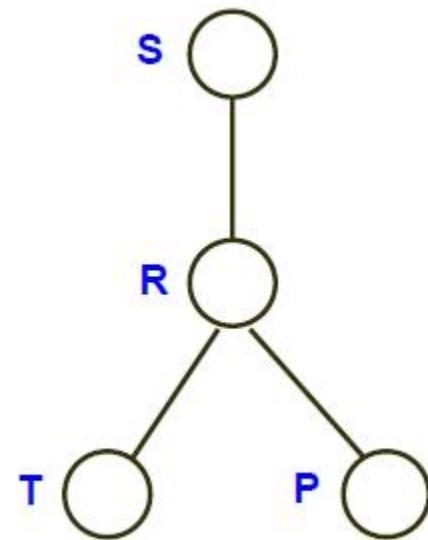
- ▶ Ví dụ:

CUNG\_UNG  $\times_F$  SAN\_PHAM

với F là điều kiện:

(CUNG\_UNG.MA\_SP = SAN\_PHAM.MA\_SP) AND (SOLUONG > 1500).

# Cây phân mảnh ngang dẫn xuất



# Các loại phân mảnh dữ liệu

## ❖ Phân mảnh dọc

- ▶ *vertical fragmentation*
- ▶ **Phân mảnh dọc gom tụ (*vertical clustering fragmentation*)**
  - Phân mảnh dư thừa (*redundant fragmentation*)
  - Phân mảnh không dư thừa (*non-redundant fragmentation*)
- ▶ **Phân mảnh dọc tách biệt (*vertical partitioning fragmentation*)**

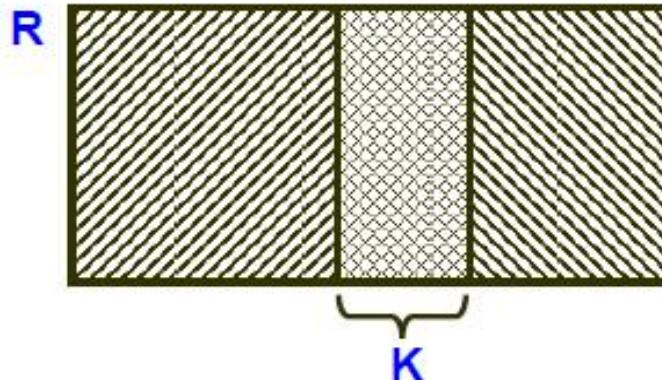
# PHÂN MẢNH DỌC

❖ **Phân mảnh dọc** một quan hệ tổng thể n-bộ R là tách R thành các quan hệ con  $R_1, R_2, \dots, R_k$  sao cho quan hệ R có thể được khôi phục lại từ các quan hệ con này bằng phép nối:

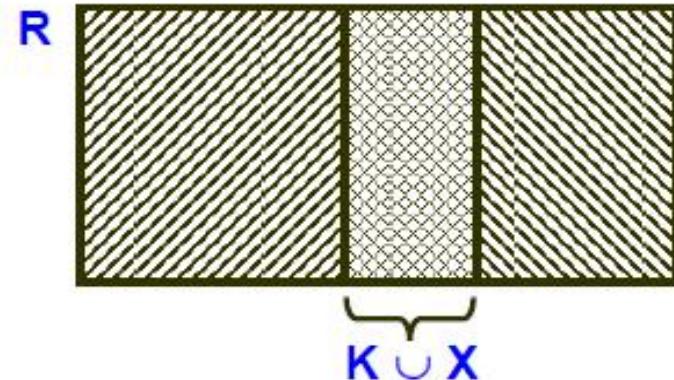
$$R = R_1 \bowtie R_2 \bowtie \dots \bowtie R_k$$

❖ **Phân mảnh dọc** là sự phân chia tập thuộc tính của một quan hệ toàn cục thành các tập thuộc tính con; các mảnh dọc (vertical fragment) có được bằng cách chiếu quan hệ toàn cục trên mỗi tập thuộc tính con.

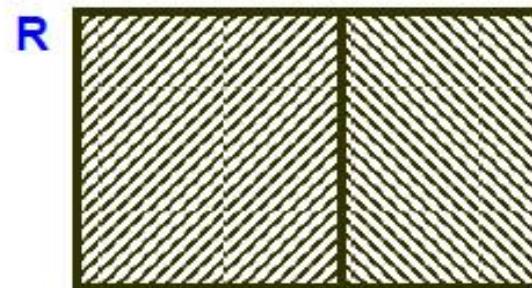
# Các loại phân mảng dữ liệu



Phân mảng gom tụ không dư thừa



Phân mảng gom tụ có dư thừa



Phân mảng dọc tách biệt

Các loại phân mảng dọc

# Phân mảng dọc

- ❖ Ví dụ

- ▶ Quan hệ toàn cục:

emp (empnum, name, sal, tax, mgrnum, deptnum)

- ▶ Phân mảng dọc không dư thừa:

$\text{emp}_1 = \Pi_{\text{empnum, name, mgrnum, deptnum}} \text{emp}$

$\text{emp}_2 = \Pi_{\text{empnum, sal, tax}} \text{emp}$

$\text{emp} = \text{emp}_1 \triangleright \triangleleft \text{empnum} = \text{empnum} \text{emp}_2$

- ▶ Phân mảng dọc dư thừa:

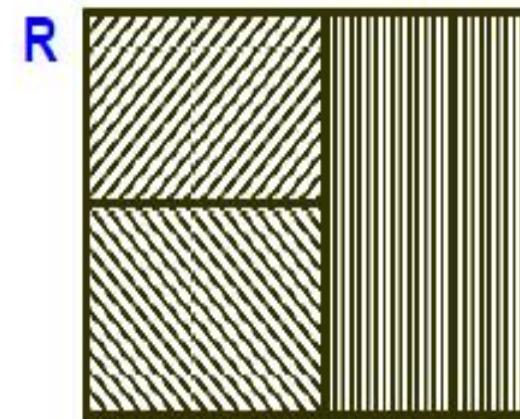
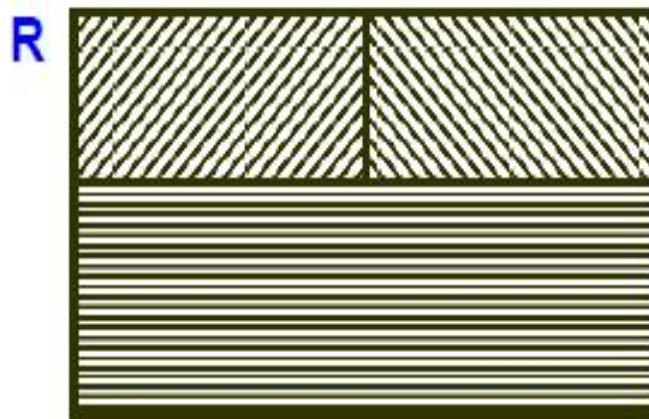
$\text{emp}_1 = \Pi_{\text{empnum, name, mgrnum, deptnum}} \text{emp}$

$\text{emp}_2 = \Pi_{\text{empnum, name, sal, tax}} \text{emp}$

$\text{emp} = \text{emp}_1 \triangleright \triangleleft \text{empnum} = \text{empnum} \Pi_{\text{empnum, sal, tax}} \text{emp}_2$

# PHÂN MẢNH HỖN HỢP

- ▶ *mixed fragmentation*
- ▶ Kết hợp giữa phân mảnh ngang và phân mảnh dọc.



# Phân mảng hỗn hợp

- ❖ Một mảng ngang được phân mảng dọc.
- ❖ Một mảng dọc được phân mảng ngang.
- ❖ **Ví dụ**

► **Quan hệ toàn cục:**

**emp** (empnum, name, sal, tax, mgrnum, deptnum)

► **Phân mảng hỗn hợp:**

**emp<sub>1</sub>** =  $\sigma_{deptnum \leq 10} \Pi_{empnum, name, mgrnum, deptnum}$  **emp**

**emp<sub>2</sub>** =  $\sigma_{10 < deptnum \leq 20} \Pi_{empnum, name, mgrnum, deptnum}$  **emp**

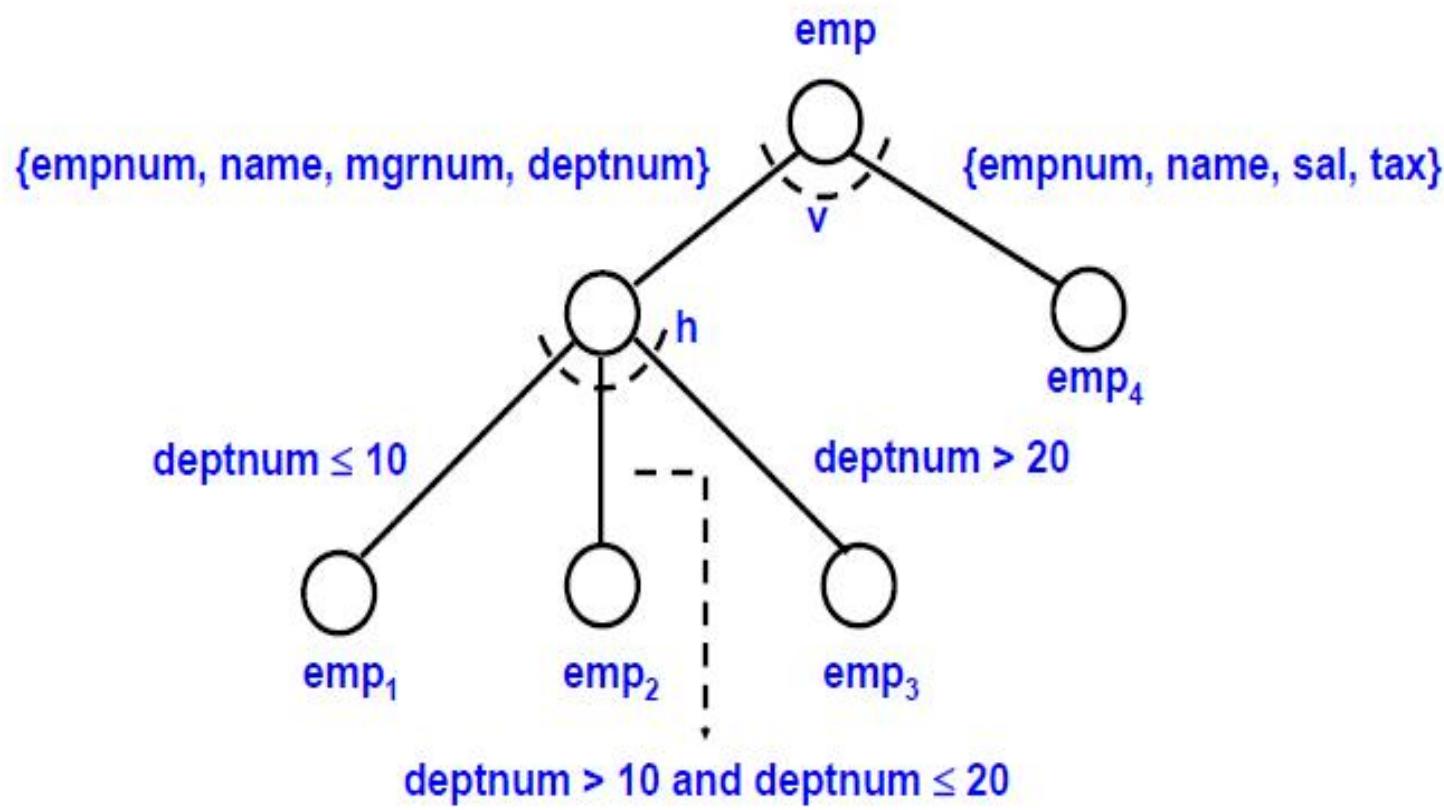
**emp<sub>3</sub>** =  $\sigma_{deptnum > 20} \Pi_{empnum, name, mgrnum, deptnum}$  **emp**

**emp<sub>4</sub>** =  $\Pi_{empnum, name, sal, tax}$  **emp**

**emp** = (**emp<sub>1</sub>**  $\cup$  **emp<sub>2</sub>**  $\cup$  **emp<sub>3</sub>**)

▷◁  $empnum = empnum \Pi_{empnum, sal, tax}$  **emp<sub>4</sub>**

# Phân mảnh hỗn hợp



y phân mảnh của quan hệ EMP

## Example.DDB

### ❖ Lược đồ toàn cục:

emp (empnum, name, sal, tax, mgrnum, deptnum)

dept (deptnum, name, area, mgrnum)

supplier (snum, name, city)

supply (snum, pnum, deptnum, quan)

### ❖ Lược đồ phân mảnh:

$\text{emp}_1 = \sigma_{\text{deptnum} \leq 10} \Pi_{\text{empnum, name, mgrnum, deptnum}} \text{emp}$

$\text{emp}_2 = \sigma_{10 < \text{deptnum} \leq 20} \Pi_{\text{empnum, name, mgrnum, deptnum}} \text{emp}$

$\text{emp}_3 = \sigma_{\text{deptnum} > 20} \Pi_{\text{empnum, name, mgrnum, deptnum}} \text{emp}$

$\text{emp}_4 = \Pi_{\text{empnum, name, sal, tax}} \text{emp}$

## Example.DDB

❖ Lược đồ phân mảnh:

$\text{dept}_1 = \sigma_{\text{deptnum} \leq 10} \text{dept}$

$\text{dept}_2 = \sigma_{10 < \text{deptnum} \leq 20} \text{dept}$

$\text{dept}_3 = \sigma_{\text{deptnum} > 20} \text{dept}$

$\text{supplier}_1 = \sigma_{\text{city} = 'SF'} \text{supplier}$

$\text{supplier}_2 = \sigma_{\text{city} = 'LA'} \text{supplier}$

$\text{supply}_1 = \text{supply} \bowtie_{\text{snum} = \text{snum}} \text{supplier}_1$

$\text{supply}_2 = \text{supply} \bowtie_{\text{snum} = \text{snum}} \text{supplier}_2$

# Các yêu cầu thông tin

❖ Các yếu tố trong thiết kế tối ưu ảnh hưởng đến các quyết định phân tán.

- ▶ Tổ chức luận lý của CSDL.
- ▶ Vị trí của các ứng dụng.
- ▶ Các đặc điểm truy xuất CSDL của các ứng dụng.
- ▶ Các đặc tính của các hệ thống máy tính tại mỗi nơi.

# Các yêu cầu thông tin

---

- ❖ **Các loại thông tin để thiết kế phân tán**
  - ▶ Thông tin về CSDL
  - ▶ Thông tin về ứng dụng
  - ▶ Thông tin về mạng truyền thông
  - ▶ Thông tin về hệ thống máy tính

# Thiết kế phân mảnh ngang

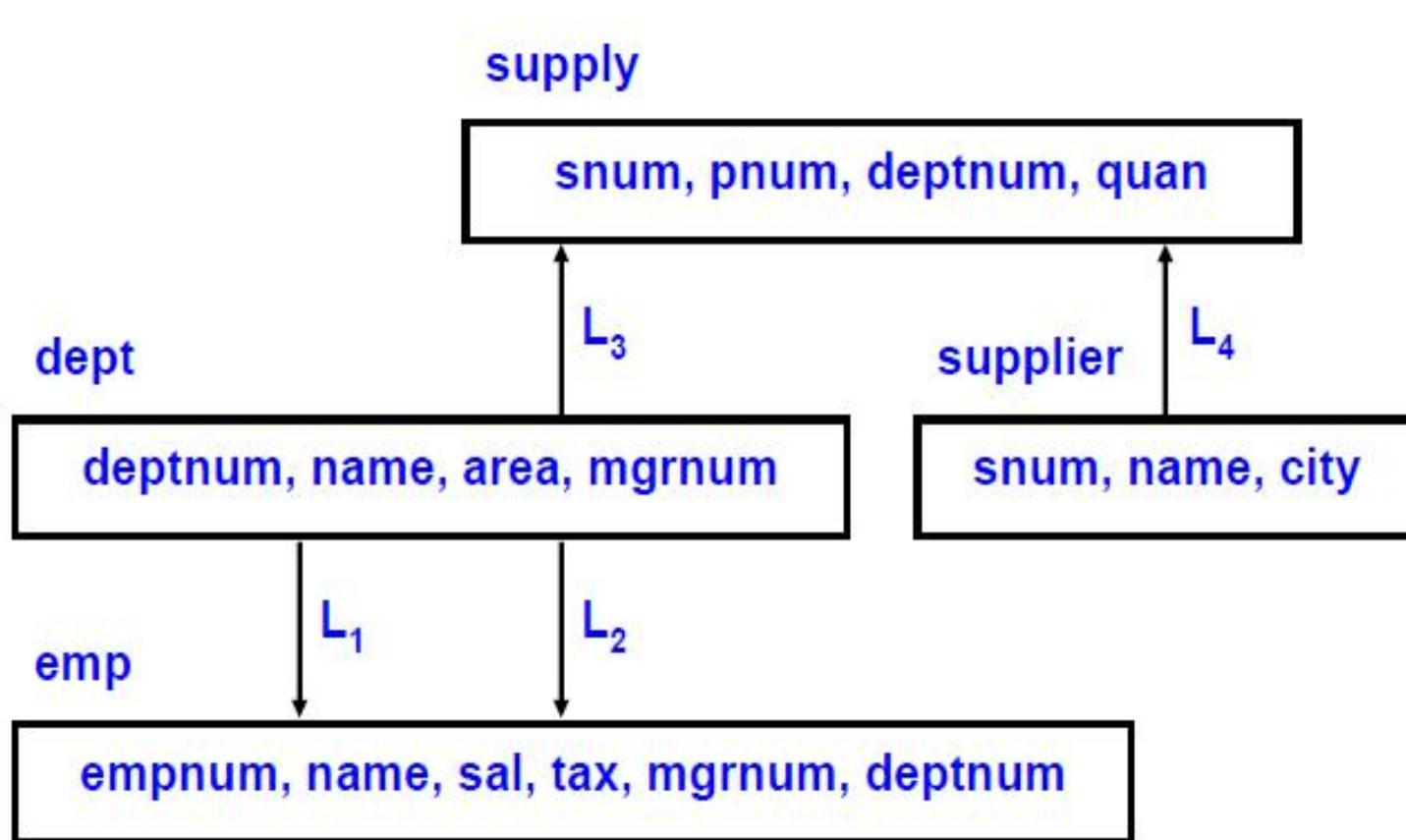
- ❖ Mỗi mảnh là một tập hợp con gồm các bộ của quan hệ.
- ❖ Phân mảnh ngang chính là phân chia một quan hệ dựa vào các vị từ định tính được định nghĩa trên quan hệ này.
- ❖ Phân mảnh ngang dẫn xuất là phân chia một quan hệ dựa vào các vị từ định tính được định nghĩa trên một quan hệ khác.

# Thiết kế phân mảnh ngang

## ❖ Thông tin về CSDL

- ▶ Trong lược đồ ý niệm toàn cục, các quan hệ được kết với nhau.
- ▶ Trong mô hình liên kết thực thể (*ER model*):
  - Quan hệ chủ hoặc quan hệ nguồn
  - Quan hệ bộ phận hoặc quan hệ đích
  - Các hàm *owner* và *member*

# Thiết kế phân mảnh ngang



$owner(L_1) = \text{dept}$

$member(L_1) = \text{emp}$

# Thiết kế phân mảnh ngang

## ❖ Thông tin về ứng dụng

- ▶ Các vị từ được sử dụng trong các truy vấn.
- ▶ Chỉ phân tích các ứng dụng quan trọng để xác định các vị từ này.
- ▶ Giả sử phân mảnh ngang quan hệ  $R(A_1, A_2, \dots, A_n)$ , với  $A_i$  là thuộc tính được định nghĩa trên miền  $D_i$ .

# Thiết kế phân mảnh ngang chính

▶ Cho R là quan hệ toàn cục mà chúng ta phân mảnh ngang chính. Chúng ta có một số định nghĩa sau:

1. Một **vị từ đơn giản (simple predicate)** là vị từ có kiểu:

**<Thuộc tính> θ <giá trị>**

Với **θ** là một trong các phép so sánh:  $=, \neq, >, \geq, <, \leq$

2. Một **vị từ giao tối thiểu / hội sơ cấp (minterm predicate)** y cho một tập các vị từ đơn giản P là chuẩn hội của tất cả các vị từ xuất hiện trong P :

$$y = \Lambda p_i^*$$

Với  $p_i^* = p_i$  hoặc  $p_i^* = \text{not } p_i$  và  $y \neq \text{false}$ .

# Thiết kế phân mảnh ngang chính

3. Một phân mảnh là một tập các bộ tương ứng với một vị từ giao tối thiểu.
4. Một vị từ **đơn giản**  $p_i$  là thích hợp đối với một tập các vị từ đơn giản  $P$  nếu tồn tại ít nhất hai vị từ **giao tối thiểu** mà biểu thức của nó chỉ khác nhau do vị từ  $p_i$  (xuất hiện ở dạng thông thường và dạng phủ định của nó) mà các phân mảnh tương ứng được tham khảo đến bởi ít nhất một ứng dụng.

## Thiết kế phân mảnh ngang chính

- ❖ Mảnh ngang chính được xác định bằng phép chọn trên quan hệ toàn cục.

$$R_i = \sigma_{F_i}(R); \quad 1 \leq i \leq n$$

- ▶  $F_i$  là điều kiện chọn của mảnh  $R_i$ ,
- ▶ Nếu  $F_i$  ở dạng chuẩn giao thì nó là một vị từ giao tối thiểu  $m_i$ ,
- ❖ Tính đúng đắn của phân mảnh ngang chính: mỗi bộ của quan hệ toàn cục được đưa vào trong một và chỉ một mảnh.

## Thiết kế phân mảnh ngang chính

- ❖ Xác định phân mảnh ngang chính của một quan hệ toàn cục là xác định một tập các **vị từ chọn** (*selection predicate*) đầy đủ và tách biệt.
- ❖ Các bộ thuộc cùng một mảnh phải được tham chiếu giống nhau trong tất cả các ứng dụng.
- ❖ **Mảnh ngang** (*horizontal fragment*) hoặc **mảnh giao tối thiểu** (*minterm fragment*)  $R_i$  bao gồm tất cả các bộ của  $R$  thỏa mãn vị từ giao tối thiểu  $m_i$ .

# Thiết kế phân mảnh ngang chính

- ▶ Ví dụ: Xét sự phân mảnh ngang cho quan hệ EMP.  
**EMP(EMPNUM, NAME, SAL, TAX, MNRNUM, DEPTNUM)**
- ▶ Giả sử rằng các ứng dụng quan trọng của CSDLPT này yêu cầu thông tin từ quan hệ EMP về **các nhân viên là thành viên của các dự án**. Mỗi phòng ban là một site của CSDLPT.
- ▶ Các ứng dụng có thể được gọi từ **bất kỳ phòng ban nào**; tuy nhiên khi chúng được gọi từ một PB thì nó sẽ ưu tiên tìm các bộ nhân viên trong PB đó trước với xác suất cao hơn ở những nhân viên của PB khác. Trong trường hợp này các nhân viên được phân mảnh ngang **“làm việc cùng một phòng ban”**.

# Thiết kế phân mảnh ngang chính

- ▶ Ví dụ: Xét sự phân mảnh ngang ở ví dụ trước (**slide 79**).
- ▶ Giả sử có một số ứng dụng quan trọng yêu cầu các thông tin về các nhân viên tham gia vào các dự án; lại có một số ứng dụng quan trọng khác không chỉ yêu cầu thông tin trên mà còn cần thông tin về nghề nghiệp.
- ▶ Hai **vị từ đơn giản** cho ví dụ này là:  
**DEPT =1 và JOB =”P”**
- ▶ Các **vị từ giao tối thiểu** cho hai vị từ này là:

# Thiết kế phân mảnh ngang chính

- ▶ Ví dụ: (tt)

**DEPT = 1 AND JOB = "P"**

**DEPT = 1 AND JOB ≠ "P"**

**DEPT ≠ 1 AND JOB = "P"**

**DEPT ≠ 1 AND JOB ≠ "P"**

- ▶ Tất cả các vị từ đơn giản trên là thích hợp

# Thiết kế phân mảnh ngang chính

- ▶ Cho  $P = \{p_1, p_2, \dots, p_n\}$  là tập các **vị từ đơn giản**. Để  $P$  thể hiện sự phân mảnh một cách đúng đắn và hiệu quả.  $P$  phải **đầy đủ và tối thiểu**.
  1. Tập các vị từ đơn giản  $P_r$  được gọi là **đầy đủ** nếu và chỉ nếu xác xuất mỗi ứng dụng truy xuất đến một bộ bất kỳ thuộc về một mảnh hội sơ cấp nào đó được định nghĩa theo  $P_r$  đều bằng nhau.
  2. Tập các vị từ đơn giản  $P_r$  được gọi là **tối thiểu** nếu tất cả các vị từ của nó thích hợp.

# Thiết kế phân mảnh ngang chính

Ví dụ: Hai ví dụ vừa nêu có thể được sử dụng để làm rõ các định nghĩa này.

P1 = { DEPT = 1 } không đầy đủ, vì các ứng dụng tham khảo các bộ của các lập trình viên với xác suất lớn hơn những phân mảnh khác dẫn từ P1.

P2 = { DEPT = 1, JOB = “P” } là tối thiểu và đầy đủ.

P3 = {DEPT = 1, JOB = “P”, SAL >5 } là đầy đủ nhưng không tối thiểu vì SAL > 5 không thích hợp ?.

# Thiết kế phân mảnh ngang chính

Sự phân mảnh có thể thực hiện như sau:

➤ **Nguyên tắc:** Xét một vị từ  $p_1$  phân chia các bộ của  $R$  thành hai phần mà chúng được tham khảo khác nhau bởi ít nhất một ứng dụng. Đặt  $P = p_1$ .

➤ **Phương pháp:** Xét một vị từ đơn giản mới  $p_i$  phân chia ít nhất một phân mảnh của  $P$  thành hai phần mà được tham khảo khác nhau bởi ít nhất bởi một ứng dụng.

Đặt  $P = P \cup p_i$ .

Xoá các vị từ không thích hợp khỏi  $P$ . Lặp lại bước này cho đến khi tập của các phân mảnh cơ sở là **đầy đủ**.

# Thiết kế phân mảnh ngang chính

Ví dụ: Lấy lại các ví dụ để làm ví dụ minh họa cho phương pháp ở trên.

➤ Xét vị từ đầu tiên SAL > 5; giả sử lương trung bình của các lập trình viên lớn hơn 5, vị từ này xác định hai tập nhân viên mà được tham khảo một cách khác nhau bởi các ứng dụng.

Ta có **P1 = { SAL > 5}**

➤ Chúng ta xét DEPT = 1; vị từ này là thích hợp và được thêm vào tập P1, ta được **P2 = {SAL > 5, DEPT = 1}**

➤ Cuối cùng, xét JOB = “P”. Vị từ này cũng thích hợp và thêm nó vào P2, ta được **P3 = { SAL>5, DEPT = 1, JOB =”P”}**. Chúng ta khám phá ra SAL>5 không thích hợp trong P3. Vì thế chúng ta nhận được tập cuối cùng là:

**P4 = { DEPT =1 , JOB = “P”} tối thiểu và đầy đủ.**

# Thiết kế phân mảnh ngang chính

- ❖ Ví dụ tổng quát: CSDL gồm có các quan hệ EMP, DEPT, SUPPLIER, SUPPLY.
  - Giả sử CSDLPT của công ty ở California có ba **sites** tại San Francisco (site 1), Fresno (site 2), và Los Angeles (site 3); Fresno nằm giữa San Francisco và Los Angeles.
  - Có tất cả 30 **phòng ban** được nhóm lại như sau: 10 PB ban đầu tiên ở gần San Francisco, các PB từ 11 đến 20 ở gần Fresno và các PB trên 20 thì ở gần Los Angeles. Tất cả các nhà cung cấp ở San Francisco hoặc ở Los Angeles.
  - Ngoài ra công ty cũng được chia theo khái niệm **miền**: San Francisco ở miền Bắc, Los Angeles ở miền nam còn Fresno nằm giữa hai miền đó nên một số phòng ban nằm gần Fresno sẽ rơi vào miền trung và một số phòng ban khác nằm gần Los Angeles.

# **Thiết kế phân mảnh ngang chính**

- ❖ Chúng ta thiết kế sự phân mảnh của SUPPLIER và DEPT với sự phân mảnh ngang chính.
  - Các nhà cung cấp trong quan hệ SUPPLIER(SNUM, NAME, CITY) có giá trị của thuộc tính CITY là “SF” hoặc là “LA”.
  - Giả sử có một ứng dụng quan trọng yêu cầu cho biết tên nhà cung cấp (NAME) khi nhập mã số nhà cung cấp (SNUM).
  - Câu lệnh SQL cho ứng dụng đó như sau:

```
Select NAME  
from SUPPLIER  
where SNUM = $X
```

# Thiết kế phân mảnh ngang chính

- Ứng dụng được gọi tại bất kỳ site nào:
  - Nếu nó được gọi tại site 1, nó sẽ tham khảo đến SUPPLIERS có CITY = “SF” với xác suất 80%;
  - Nếu được gọi từ site 2, nó sẽ tham khảo đến SUPPLIERS của “SF” và “LA” với xác suất bằng nhau;
  - Nếu nó được gọi từ site 3, nó sẽ tham khảo đến SUPPLIERS của “LA” với xác suất 80%. Điều này dẫn đến là các phòng ban sẽ liên hệ đến các nhà cung cấp ở gần đó.
- Chúng ta đưa các vị từ sau:

**p1 : CITY = “SF”**

**P2: CITY = “LA”**

**Tập {p1 , p 2} là tối thiểu và đầy đủ.**

# Thiết kế phân mảnh ngang chính

- Ví dụ này minh họa hai tính chất quan trọng sau:
  - Các vị từ thích hợp mô tả cho phân mảnh này không thể được suy ra bằng cách phân tích mã lệnh của ứng dụng.
  - Quan hệ mật thiết giữa các vị từ giảm đi số lượng phân mảnh. Trong trường hợp này chúng ta nên xem xét những vị từ tương ứng với các vị từ sơ cấp sau:

y1: (CITY = “SF”) AND (CITY = “LA”)

y2: (CITY = “SF”) AND NOT(CITY = “LA”)

y3: NOT(CITY = “SF”) AND (CITY = “LA”)

y4: NOT(CITY = “SF”) AND NOT(CITY = “LA”)

Nhưng: (CITY = “LA”)  $\Rightarrow$  NOT (CITY = “SF”) và  
(CITY = “SF”)  $\Rightarrow$  NOT (CITY = “LA”)

Vì thế chúng ta suy ra y1 và y4 mâu thuẫn lẫn nhau và y2 và y3 sẽ đơn giản thành hai vị từ p1 và p2.

# Thiết kế phân mảnh ngang chính

➤ Xét quan hệ toàn cục sau:

**DEPT(DEPTNUM, NAME, AREA, MGRNUM)**

➤ Giả sử có các ứng dụng quan trọng sau:

- Các ứng dụng quản trị chỉ được gọi từ site 1 và site 3;
- Các ứng dụng quản trị về các phòng ban ở miền bắc được gọi tại site 1 và các ứng dụng quản trị về các phòng ban ở miền nam được gọi tại site 3.
- Các ứng dụng về công việc được quản lý tại mỗi phòng ban; chúng có thể được gọi từ bất kỳ phòng ban nào nhưng chúng phải tham khảo các bộ của phòng ban gần site của nó nhất với xác xuất cao hơn các bộ ở những lưu ở những nơi khác.

# Thiết kế phân mảnh ngang chính

➤ Chúng ta có các vị từ sau:

p1: DEPTNUM  $\leq 10$

p2:  $10 < \text{DEPTNUM} \leq 20$

p3:  $\text{DEPTNUM} > 20$

p4: AREA = “North”

p5: AREA = “South”

➤ Có một số quan hệ giữa các vị từ trên như AREA= “North” kéo theo DEPTNUM  $> 20$  là sai; vì thế sự phân mảnh giảm còn 4 phân mảnh:

y1: DEPTNUM  $\leq 10$

y2:  $(10 < \text{DEPTNUM} \leq 20) \text{ AND } (\text{AREA} = \text{“North”})$

y3:  $(10 < \text{DEPTNUM} \leq 20) \text{ AND } (\text{AREA} = \text{“South”})$

y4:  $\text{DEPTNUM} > 20$

# Thiết kế phân mảnh ngang chính

	p <sub>4</sub> : AREA = “North”	p <sub>5</sub> : AREA = “South”
p <sub>1</sub> : DEPTNUM <= 10	y <sub>1</sub>	FALSE
p <sub>2</sub> : 10 < DEPTNUM <= 20	y <sub>2</sub>	y <sub>3</sub>
p <sub>3</sub> : DEPTNUM > 20	FALSE	y <sub>4</sub>

Sử phân mảnh của quan hệ DEPT

# Thiết kế phân mảnh ngang chính

## Nhận xét

- Sự cấp phát phân mảnh cũng dễ dàng thấy qua sự phân mảnh này.
- Các phân mảnh tương ứng với vị từ y1 và y4 được lưu trữ tại site 1 và site 3;
- Các phân mảnh ứng với các vị từ y2 hoặc y3 thể hiện các ứng dụng quản trị thì được phân bố tại site 1 hoặc 3 và các phân mảnh ứng về công việc của phòng ban có thể được lưu trữ tại site 2.

**Thuật toán COM\_MIN được giới thiệu để tìm tập các vị từ đầy đủ và tối thiểu:**

# Thiết kế phân mảnh ngang chính

---

## ❖ Các bước thiết kế phân mảnh ngang

- ▶ **Bước 1:** Tìm tập các vị từ chọn  $P_r$ , là đầy đủ và tối thiểu.
- ▶ **Bước 2:** Tìm tập các vị từ giao tối thiểu có thể được định nghĩa trên các vị từ của  $P_r$ .

## Thiết kế phân mảnh ngang chính

- ❖ Một vị từ đơn giản  $p_i$  được gọi là **thích hợp** (*relevant*) đối với một tập  $P_r$ , các vị từ đơn giản, nếu tồn tại ít nhất hai vị từ giao tối thiểu  $m_i$  và  $m_j$  của  $P_r$ , mà các biểu thức của chúng chỉ khác nhau ở  $p_i$  (tức là  $m_i$  chứa  $p_i$  và  $m_j$  chứa  $\neg p_i$ ) và tồn tại ít nhất một ứng dụng tham chiếu khác nhau đến hai mảnh  $f_i$  và  $f_j$  (tương ứng với  $m_i$  và  $m_j$ ).

## Thiết kế phân mảnh ngang chính

- ❖ Một tập các vị từ đơn giản  $P_r$  được gọi là **đầy đủ** (*complete*) nếu và chỉ nếu bất kỳ hai bộ nào thuộc bất kỳ mảnh giao tối thiểu nào được định nghĩa theo  $P_r$ , thì bất kỳ ứng dụng nào đều tham chiếu đến hai bộ này với cùng một xác suất.
- ❖ Một tập các vị từ đơn giản  $P_r$  được gọi là **tối thiểu** (*minimal*) nếu tất cả các vị từ của nó là các vị từ thích hợp.
- ❖ Cho  $P_r = \{p_1, p_2, \dots, p_m\}$  là một tập các vị từ đơn giản. Để cho  $P_r$  biểu diễn phân mảnh đúng đắn và hiệu quả thì  $P_r$  phải **đầy đủ** và **tối thiểu**.

# Thiết kế phân mảnh ngang chính

- Thuật toán COM\_MIN
- Input :  $P_r$  là tập các vị từ đơn giản
- Output :tập các vị từ đầy đủ và tối thiểu
- Xác định các vị từ liên đới và tối thiểu trong  $P_r$ .

# Thiết kế phân mảnh ngang chính

Thuật toán **COM\_MIN**

Input: R: quan hệ; Pr: tập các vị từ đơn giản;

Output: Pr': tập các vị từ tối thiểu và đầy đủ;

Declare

F: tập các mảnh giao tối thiểu;

Begin

Pr' =  $\Phi$ ; F =  $\Phi$ ;

For each vị từ  $p_i \in Pr$  if  $p_i$  phân hoạch R theo **Quy tắc 1** then

Begin

Pr' := Pr'  $\cup$   $p_i$

Pr := Pr -  $p_i$

F := F  $\cup$   $f_i$  { $f_i$  là mảnh giao tối thiểu theo  $p_i$ }

End {Chúng ta đã chuyên các vị từ có phân mảnh R vào Pr'}

**Qui tắc 1:**

- Ít nhất 2 mảnh

- Truy suất khác nhau bởi ít nhất 1 ứng dụng

# Thiết kế phân mảnh ngang chính

Repeat

For each  $p_j \in Pr$

if  $p_j$  phân hoạch một mảnh  $f_k$  của  $Pr'$  theo QT1 then

Begin

$Pr' := Pr' \cup p_j;$

$Pr := Pr - p_j;$

$F := F \cup p_j;$

- Thêm các  $P_j$  vào  $Pr$  cho đến khi đầy đủ
- Khi thêm  $P_j$  vào nhớ kiểm tra  $P_j$  có thỏa mãn tối thiểu ko. (Truy suất khác nhau bởi 1 UD)

End;

Until  $Pr'$  **đầy đủ** {Không còn  $p_j$  nào phân mảnh  $f_k$  của  $Pr'$ }

For each  $p_k \in Pr'$ , if  $\exists p' \text{ mà } p_k \Leftrightarrow p'$  then

Begin {Kiểm tra tối thiểu}

$Pr' := Pr' - p_k;$

$F = F - f_k$

End;

End. {COM\_MIN}

# Thiết kế phân mảnh ngang chính

- Thuật toán PHORIZONTAL
- Input :  $R, Pr$  là tập các vị từ đơn giản
- Output :tập các vị từ giao sơ cấp
- Xác định tập  $M$  các vị từ giao sơ cấp
- Xác định các phép kéo theo
- Rút gọn  $M$

# Thiết kế phân mảnh ngang chính

## Thuật toán PHORIZONTAL

**Input:** R: quan hệ; Pr: tập các vị từ đơn giản;

**Output:** M: tập các vị từ giao tối thiểu p;

**Begin**

Pr' := COM\_MIN(R, Pr);

Xác định tập M các vị từ giao tối thiểu;

Xác định tập I các phép kéo theo giữa các  $pi \in Pr'$ ;

**For each**  $mi \in M$  **do**

**Begin**

**IF**  $mi$  mâu thuẫn với I **then**

M := M - mi

**End;**

PHORIZONTAL}

# Thiết kế phân mảnh ngang chính

- ▶ Ví dụ: Sử dụng giải thuật COM-MIN, tính pr' thỏa mãn tối thiểu và đầy đủ ?.
  - PAY

TITLE	SAL
KS. Điện	4000
KS. Hệ thống	7000
KS. Cơ khí	3500
KS. Lập trình	2000

Giả sử có 1 ứng dụng truy suất để PAY:

Q1: SELECT \* FROM PAY WHERE SAL > 3500

# Thiết kế phân mảnh ngang chính

- Tập vị từ đơn giản sử dụng để phân hoạch PAY:  
p1:  $\text{SAL} \leq 3500$   
p2:  $\text{SAL} > 3500$
- Khởi tạo:  $\text{Pr} = \{p1, p2\}$
- Áp dụng thuật toán COM-MIN: với  $i=1$  làm giá trị khởi đầu sinh ra  $\text{Pr}' = \{p1\}$ .
- Đây là tập đầy đủ và tối thiểu vì  $p2$  không phân hoạch  $f1$  (là mảnh giao tối thiểu tạo ra ứng với  $p1$ ) theo Qui tắc 1.
- Các vị từ giao tối thiểu là:

$m1: \text{SAL} \leq 3500$

$(\text{SAL} \leq 3500) = (\text{SAL} > 3500)$

# Thiết kế phân mảnh ngang chính

Cuối cùng ta có PAY được phân mảnh:

PAY1

TITLE	SAL
KS. Cơ khí	3500
KS. Lập trình	2000

PAY2

TITLE	SAL
KS. Điện	4000
KS. Hệ thống	7000

# Thiết kế phân mảnh ngang chính

- ▶ **Bài tập:** Sử dụng giải thuật COM-MIN, tính pr' thỏa mãn tối thiểu và đầy đủ ?.
  - DUAN

MADA	TENDA	NGANSACH	VITRI
P1	Thiết bị	150	HUE
P2	CSDL	125	TPHCM
P3	Games	75	TPHCM
P4	CAD	100	HN

Giả sử có 2 ứng dụng truy suất đến DUAN:

Q1: SELECT \* FROM DUAN WHERE VITRI= Value

Q2: SELECT \* FROM DUAN WHERE NGANSACH>100

# Thiết kế phân mảnh ngang chính

- Tập vị từ đơn giản sử dụng để phân hoạch DUAN:
  - p1: VITRI = ‘HUE’
  - p2: VITRI = ‘TPHCM’
  - p3: VITRI = ‘HN’
  - p4: NGANSACH > 100
  - p5: NGANSACH <= 100
- Khởi tạo:  $Pr=\{p1, p2, p3, p4, p5\}$
- Áp dụng thuật toán COM-MIN:
  - i=1:** Vị từ p1 thỏa qui tắc 1.  $Pr'=\{p1\}$
  - i=2:** Vị từ p2 thỏa qui tắc 1 .  $Pr'=\{p1, p2\}$
  - i=3:** Ta có vị từ p3 không phân hoạch f2 (là mảnh giao tối thiểu tạo ra ứng với p2) theo Qui tắc 1. Vì vậy:  $Pr'=\{p1, p2\}$

i=4: Vị từ p4 thỏa qui tắc 1 .  $Pr' = \{p1, p2, p4\}$

i=5: Ta có vị từ p3 không phân hoạch f4 (là mảnh giao tối thiểu tạo ra ứng với p4) theo Qui tắc 1. Vì vậy:  $Pr' = \{p1, p2, p4\}$

Kết luận:  $Pr' = \{p1, p2, p4\}$  là đầy đủ và tối thiểu.

## Thuật toán PHORIZONTAL

➤ Các vị từ giao tối thiểu là:

$$m1: p1 \wedge p4;$$

$$m2: p1 \wedge \neg p4 = p1 \wedge p5;$$

$$m3: p2 \wedge p4;$$

$$m4: p2 \wedge \neg p4 = p2 \wedge p5;$$

$$m5: \neg(p1 \wedge p2) \wedge p4 = p3 \wedge p4;$$

$$m6: \neg p3 \wedge \neg p4 = p3 \wedge p5$$

Vì vậy, ta ta tập các vị từ giao tối thiểu là:

$$\{m1, m2, m3, m4, m5, m6\}$$

➤ Phân tích: Xem xét hình ảnh minh quan hệ DUAN theo M là:

- ▶ DUAN1 =  $\delta_{(m1)}$  (DUAN)
- ▶ DUAN2 =  $\delta_{(m2)}$  (DUAN)
- ▶ DUAN3 =  $\delta_{(m3)}$  (DUAN)
- ▶ DUAN4 =  $\delta_{(m4)}$  (DUAN)
- ▶ DUAN5 =  $\delta_{(m5)}$  (DUAN)
- ▶ DUAN6 =  $\delta_{(m6)}$  (DUAN)

▶ **Đối với dữ liệu Demo:**  
**DUAN2 và DUAN5 là Rỗng.**

**DUAN1**

	MADA	TENDA	NGANSACH	VITRI
	P1	Thiết bị	150	HUE

**DUAN3**

	MADA	TENDA	NGANSACH	VITRI
	P2	CSDL	125	TPHCM

**DUAN4**

	MADA	TENDA	NGANSACH	VITRI
	P3	Games	75	TPHCM

**DUAN6**

	MADA	TENDA	NGANSACH	VITRI
	P4	CAD	100	HN

## Thiết kế phân mảnh ngang dẫn xuất

- ❖ Phân mảnh ngang dẫn xuất được định nghĩa trên các quan hệ bộ phận của đường liên kết theo phép chọn trên quan hệ chủ của đường liên kết này.
- ❖ Đường liên kết giữa quan hệ chủ và quan hệ bộ phận được định nghĩa là một phép kết bằng.
- ❖ Một phép kết bằng có thể được thực hiện bằng các phép nửa kết.

# Thiết kế phân mảnh ngang dẫn xuất

- ❖ Xét đường liên kết  $L$  với  $\text{owner}(L) = S$  và  $\text{member}(L) = R$ , các mảnh ngang dẫn xuất của  $R$  được định nghĩa như sau:

$$R_i = R \triangleright\triangleleft_F S_i, \quad 1 \leq i \leq n$$

- ▶  $n$  là số lượng lớn nhất các mảnh được định nghĩa trên  $R$ .
- ▶  $S_i = \sigma_{F_i}(S)$  với  $F_i$  là công thức dùng để định nghĩa mảnh ngang chính  $S_i$ ,
- ▶  $F$  là điều kiện nửa kết.

## **Thiết kế phân mảnh ngang dẫn xuất**

- ❖ Để thực hiện phân mảnh ngang dẫn xuất, cần có:
  - ▶ Tập các mảnh của quan hệ chủ
  - ▶ Quan hệ bộ phận
  - ▶ Tập các vị từ nửa kết giữa quan hệ chủ và quan hệ bộ phận.

# Thiết kế phân mảnh ngang dẫn xuất

- ❖ *Phép kết phân tán (distributed join)* là một phép kết giữa các quan hệ được phân mảnh ngang.

$$R \bowtie_F S = (\cup_i R_i) \bowtie_F (\cup_j S_j)$$

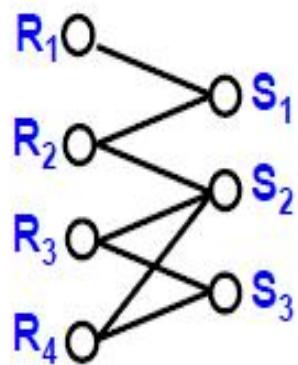
$$R \bowtie_F S = \cup_{ij} (R_i \bowtie_F S_j)$$

- ▶ Có thể suy diễn để xác định một số phép kết từng phần  $R_i \bowtie_F S_j = \emptyset$ .
- ❖ Phép kết phân tán được biểu diễn bằng *đồ thị kết (join graph)*.

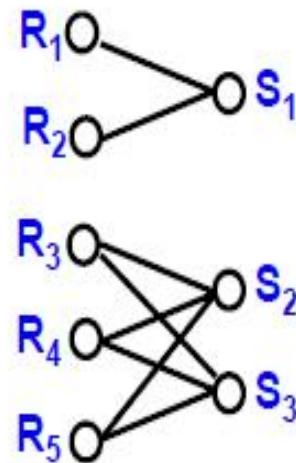
## Thiết kế phân mảnh ngang dân xuất

- ❖ Đồ thị kết được gọi là **hoàn toàn** (*total*) nếu nó chứa tất cả các cạnh có thể có giữa các mảnh của  $R$  và  $S$ .
- ❖ Đồ thị kết được gọi là **suy giảm** (*reduced*) nếu không có một số cạnh giữa các mảnh của  $R$  và  $S$ .
  - ▶ Đồ thị kết suy giảm được gọi là **phân hoạch** (*partitioned*) nếu nó bao gồm hai hoặc nhiều đồ thị con và không có các cạnh giữa chúng.
  - ▶ Đồ thị kết suy giảm được gọi là **đơn giản** (*simple*) nếu nó là phân hoạch và mỗi đồ thị con có đúng một cạnh.

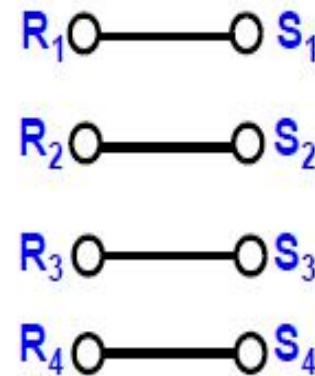
# Thiết kế phân mảnh ngang dẫn xuất



(a) Đồ thị kết



(b) Đồ thị kết phân hoạch



(c) Đồ thị kết đơn giản

Các loại đồ thị kết

# Thiết kế phân mảnh ngang dẫn xuất

- ❖ Có thể có nhiều đường liên kết đến một quan hệ  $R$  và có nhiều cách phân mảnh ngang dẫn xuất cho  $R$  dựa trên hai tiêu chuẩn:
  - ▶ Sự phân mảnh có các đặc điểm kết tốt hơn.
  - ▶ Sự phân mảnh được sử dụng trong nhiều ứng dụng hơn.

# Thiết kế phân mảnh dọc

- Thông tin về CSDL
  - Các quan hệ.
- Thông tin về ứng dụng
  - Tần số truy xuất thuộc tính của các ứng dụng.
  - Chỉ phân tích các ứng dụng liên quan tới quan hệ cần xét.
  - Ma trận sử dụng thuộc tính.
  - Ma trận ái lực thuộc tính.
  - Ma trận ái lực tựu giữa các thuộc tính.
  - Thuật toán phân hoạch.

# Công cụ: Ma trận ái lực (AA)

- Ví dụ:
- Projekt(Pnum, Pname, Budget, Loc)
- Đặt A1 A2 A3 A4
- Câu truy vấn (Ứng dụng)
  - Q1: Cho biết ngân sách của dự án theo số dự án
  - Q2: Cho biết tên và ngân sách của tất cả các dự án
  - Q3: Tên các dự án của 1 thành phố
  - Q4: Tổng ngân sách của các dự án của 1 thành phố

# Ma trận sử dụng

- $\text{Use}(q_i, A_j) = \begin{cases} 1 & \text{nếu thuộc tính } A_j \text{ được truy vấn } q_i \text{ tham chiếu} \\ 0 & \text{Trong trường hợp ngược lại} \end{cases}$
- Ví dụ

	A1	A2	A3	A4
q1	1	0	1	0
q2	0	1	1	0
q3	0	1	0	1
q4	0	0	1	1

## Ma trận AA: Định nghĩa

- $\text{Ref}_s(q_k)$ , cho hai thuộc tính  $(A_i, A_k) = \text{số truy xuất được thực hiện bởi truy vấn } q_k \text{ tại site } s$  tham chiếu đến  $A_i$  và  $A_k$
- $\text{Acc}_s(q_k) = \text{tần số của } q_k \text{ tại site } s$
- Ma trận ái lực:

$$\text{Aff}(A_i, A_j) = \sum_{k | \text{use}(q_k, A_i) = 1 \wedge \text{use}(q_k, A_j) = 1} \text{Ref}_s(q_k) \text{Acc}_s(q_k)$$

## Ma trận AA: Ví dụ

Giả sử:  $\text{Ref}_s(q_k) = 1 \quad \forall s, k$

Các tần số truy xuất (3 sites)

- $\text{acc}_1(q_1)=15 ; \text{acc}_2(q_1)=20 ; \text{acc}_3(q_1)=10$
- $\text{acc}_1(q_2)=5 ; \text{acc}_2(q_2)=0 ; \text{acc}_3(q_2)=0$
- $\text{acc}_1(q_3)=25 ; \text{acc}_2(q_3)=25 ; \text{acc}_3(q_3)=25$
- $\text{acc}_1(q_4)=3 ; \text{acc}_2(q_4)=0 ; \text{acc}_3(q_4)=0$

# Ma trận AA: Ví dụ

- $\text{Aff}(A_1, A_3) = \sum_{k: \text{use}(q_k, A_i) = 1 \wedge \text{use}(q_k, A_j) = 1} \sum_{\forall S | \text{ref}_l(q_k) \in S} \text{acc}_l(q_k)$   
=  $\text{acc}_1(q_1) + \text{acc}_2(q_1) + \text{acc}_3(q_1)$   
= 45
- Kết quả: ma trận ái lực thuộc tính(AA)

	A1	A2	A3	A4
A1	45	0	45	0
A2	0	80	5	75
A3	45	5	53	3
A4	0	75	3	78

# Gom nhóm các thuộc tính

- Gom nhóm các thuộc tính có ái lực cao
- Ví dụ

	A1	A2	A3	A4
A1	45	0	45	0
A2	0	80	5	75
A3	45	5	53	3
A4	0	75	3	78



	A1	A3	A2	A4
A1	45	45	0	0
A3	45	53	5	3
A2	0	5	80	75
A4	0	3	75	78

# Thuật toán gom nhóm

- Mục tiêu: Nhóm các thuộc tính của một quan hệ dựa trên các giá trị ái lực thuộc tính trong ma trận ái lực.
- Thuật toán năng lượng nối : Bond Energy Algorithm BEA

# Các số đo

Số đo ái lực chung (Affinity Measure)

$$AM = \sum_{i=1} \sum_{j=1} \text{Aff}(A_i, A_j) [\text{Aff}(A_i, A_{j-1}) + \text{Aff}(A_i, A_{j+1})]$$

Cầu nối

$$\text{Bond}(A_x, A_y) = \sum_{z=1} \text{Aff}(A_z, A_x) \text{Aff}(A_z, A_y)$$

Đóng góp thực (net contribution) cho độ đo ái lực khi đặt thuộc tính  $A_k$  ở giữa  $A_i$  và  $A_j$  là

$$\text{Cont}(A_i, A_k, A_j) = 2 \text{ bond}(A_i, A_k) + 2 \text{ bond}(A_k, A_j) - 2 \text{ bond}(A_i, A_j)$$

## Ví dụ về độ đo bond và cont

- Xét ma trận AA, tính toán phần đóng góp khi di chuyển thuộc tính A<sub>4</sub> giữa các thuộc tính A<sub>1</sub> và A<sub>2</sub>:
- $\text{Cont}(A_1, A_4, A_2) = 2 \text{ bond}(A_1, A_4) + 2 \text{ bond}(A_4, A_2) - 2 \text{ bond}(A_1, A_2)$
- $\text{Bond}(A_1, A_4) = 45 \times 0 + 0 \times 75 + 45 \times 3 + 0 \times 78 = 135$
- $\text{Bond}(A_4, A_2) = 11865$
- $\text{Bond}(A_1, A_2) = 225$   
 $\text{Cont}(A_1, A_4, A_2) = 2 \times 135 + 2 \times 11865 - 2 \times 225$

# Thiết kế phân mảnh đọc

- Thuật toán BEA
  - Input: ma trận ái lực thuộc tính(AA)
  - Output: ma trận ái lực tựu(CA Clustered Affinity)

Begin

1. Khởi gán: Đặt và cố định cột 1 trong AA vào trong CA
2. Lặp: Lấy lần lượt một trong n-i cột còn lại thử đặt vào vị trí i+1 còn lại trong ma trận CA. Chọn nơi đặt sao cho độ đo ái lực là lớn nhất. Tiếp tục cho đến khi không còn cột nào để đặt
3. Sắp thứ tự hàng

## Thuật toán BEA

Input: AA : ma trận ái lực thuộc tính;  
Output: CA : ma trận ái lực tự sau khi đã sắp xếp lại các hàng các cột;  
Begin  
    {Khởi gán: cần nhớ rằng AA là một ma trận n x n}  
    CA(•, 1)←AA(•, 1)  
    CA(•, 2)←AA(•, 2)  
     $A_0 = 0$   
    Index:=3  
    while index <= n do       {chọn vị trí “tốt nhất” cho thuộc tính Aindex}  
        begin  
             $A_{Index+1} = 0$   
            for i :=1 to index-1 by 1 do  
                tính cont(Ai-1, Aindex, Ai);  
                Tính cont(Aindex-1, Aindex, **Aindex+1**);       { điều kiện biên}  
                Loc ← nơi đặt, được cho bởi giá trị cont lớn nhất;  
                for i: = index downto loc do               {xáo trộn hai ma trận}  
                    CA(•, j)←CA(•, j:1);  
                    CA(•, loc)←AA(•, index);  
                Index←index+1;  
        end  
    end

    {Lưu ý: Khi đã sắp xếp xong, ta có thể lặp lại quá trình trên để sắp xếp các hàng theo thứ tự tương đối của cột.  
    }

## Ma trận CA: Ví dụ

- $\text{cont}(A_0, A_3, A_1) = 2\text{bond}(A_0, A_3) + 2\text{bond}(A_3, A_1) - 2\text{bond}(A_0, A_1)$
- $\text{bond}(A_0, A_3) = 0$
- $\text{bond}(A_0, A_1) = 0$
- $\text{bond}(A_3, A_1) = 45 * 45 + 5 * 0 + 53 * 45 + 3 * 0 = 4410$
- $\text{cont}(A_0, A_3, A_1) = 2\text{bond}(A_3, A_1) = 8820$

# Ma trận CA: Ví dụ

- $\text{cont}(A_1, A_3, A_2) = 2\text{bond}(A_1, A_3) + 2\text{bond}(A_3, A_2) - 2\text{bond}(A_1, A_2)$
- $\text{bond}(A_3, A_1) = 45 * 45 + 5 * 0 + 53 * 45 + 3 * 0 = 4410$
- $\text{bond}(A_3, A_2) = 45 * 0 + 5 * 80 + 53 * 5 + 3 * 75 = 890$
- $\text{bond}(A_1, A_2) = 45 * 0 + 0 * 80 + 45 * 5 + 0 * 75 = 225$
- $\text{cont}(A_1, A_3, A_2) = 2 * (4410 + 890 - 225) = 10150$

# Ma trận CA: Ví dụ

- $\text{cont}(A_2, A_3, A_4) = 2\text{bond}(A_2, A_3) + 2\text{bond}(A_3, A_4) - 2\text{bond}(A_2, A_4)$
- $\text{bond}(A_2, A_3) = 0 * 45 + 80 * 5 + 5 * 53 + 75 * 3 = 890$
- $\text{bond}(A_3, A_4) = 0$
- $\text{bond}(A_2, A_4) = 0$
- $\text{cont}(A_2, A_3, A_4) = 2 * 890 = 1780$
- Ta thấy  $\text{cont}(A_1, A_3, A_2) = 10150$  có giá trị lớn nhất nên ta quyết định chèn cột mới vào giữa vị trí 1 và 2.

# Ma trận CA: Ví dụ

- Ví dụ:

Ma trận AA

	A1	A2	A3	A4
A1	45	0	45	0
A2	0	80	5	75
A3	45	5	53	3
A4	0	75	3	78

Chèn cột 3 của AA vào CA

Ma trận CA

	A1	A3	A2	
A1	45	45	0	
A2	0	5	80	
A3	45	53	5	
A4	0	3	75	

Thêm cột A3 vào ma trận CA xét 3 trường hợp :

- $\text{cont}(A0, A3, A1) = 8820$
- $\text{cont}(A1, A3, A2) = 10150$
- $\text{cont}(A2, A3, A4) = 1780$

Chèn cột A3 vào giữa cột A1 và A2 của CA

# Ma trận CA: Ví dụ

- Xét cột 4: cột 4 có thể nằm ở 1 trong 4 vị trí sau: 413, 143, 342, 324
- Nếu 413 ta phải tính  $\text{cont}(A_0, A_4, A_1)$
- Nếu 143 ta phải tính  $\text{cont}(A_1, A_4, A_3)$
- Nếu 342 ta phải tính  $\text{cont}(A_3, A_4, A_2)$
- Nếu 324 ta phải tính  $\text{cont}(A_2, A_4, A_5)$
- Trong đó cột 0,5 là rỗng.

# Ma trận CA: Ví dụ

- $\text{cont}(A_0, A_4, A_1) = 2\text{bond}(A_0, A_4) + 2\text{bond}(A_4, A_1) - 2\text{bond}(A_0, A_1)$
- $\text{bond}(A_0, A_4) = \text{bond}(A_0, A_1) = 0$
- $\text{bond}(A_4, A_1) = 0 * 45 + 75 * 0 + 3 * 45 + 78 * 0 = 135$
- $\text{cont}(A_0, A_4, A_1) = 2 * 135 = 270$

# Ma trận CA: Ví dụ

- $\text{cont}(A_1, A_4, A_3) = 2\text{bond}(A_1, A_4) + 2\text{bond}(A_4, A_3) - 2\text{bond}(A_1, A_3)$
- $\text{bond}(A_1, A_4) = 45 * 0 + 0 * 75 + 45 * 3 + 0 * 78 = 135$
- $\text{bond}(A_4, A_3) = 0 * 45 + 75 * 5 + 3 * 53 + 78 * 3 = 768$
- $\text{bond}(A_1, A_3) = 45 * 45 + 0 * 5 + 45 * 53 + 0 * 3 = 4410$
- $\text{cont}(A_1, A_4, A_3) = 2 * (135 + 768 - 4410) = -7014$

# Ma trận CA: Ví dụ

- $\text{cont}(A_3, A_4, A_2) = 2\text{bond}(A_3, A_4) + 2\text{bond}(A_4, A_2) - 2\text{bond}(A_3, A_2)$
- $\text{bond}(A_3, A_4) = 45*0 + 5*75 + 53*3 + 3*78 = 768$
- $\text{bond}(A_4, A_2) = 0*0 + 75*80 + 3*5 + 78*75 = 11865$
- $\text{bond}(A_3, A_2) = 45*0 + 5*80 + 53*5 + 3*75 = 890$
- $\text{cont}(A_3, A_4, A_2) = 2*(768 + 11865 - 890) = 23486$

# Ma trận CA: Ví dụ

- $\text{cont}(A_2, A_4, A_5) = 2\text{bond}(A_2, A_4) + 2\text{bond}(A_4, A_5) - 2\text{bond}(A_2, A_5)$
- $\text{bond}(A_4, A_5) = \text{bond}(A_2, A_5) = 0$
- $\text{bond}(A_2, A_4) = 0 * 0 + 80 * 75 + 5 * 3 + 75 * 78 = 11865$
- $\text{cont}(A_2, A_4, A_5) = 2 * (11865) = 23730$
- Ta thấy  $\text{cont}(A_2, A_4, A_5) = 23730$  có giá trị lớn nhất nên ta quyết định chèn cột mới vào vị trí cuối cùng.

# Ma trận CA: Ví dụ

- Sắp thứ tự các hàng theo cột

	A1	A3	A2	A4
A1	45	45	0	0
A2	0	5	80	75
A3	45	53	5	3
A4	0	3	75	78

	A1	A3	A2	A4
A1	45	45	0	0
A3	45	53	5	3
A2	0	5	80	75
A4	0	3	75	78

# Phân hoạch (Partition)

- Mục tiêu:

Tách các thuộc tính được truy xuất cùng nhau thành một phân mảnh dọc.

Tối ưu các truy xuất vào 1 phân mảnh

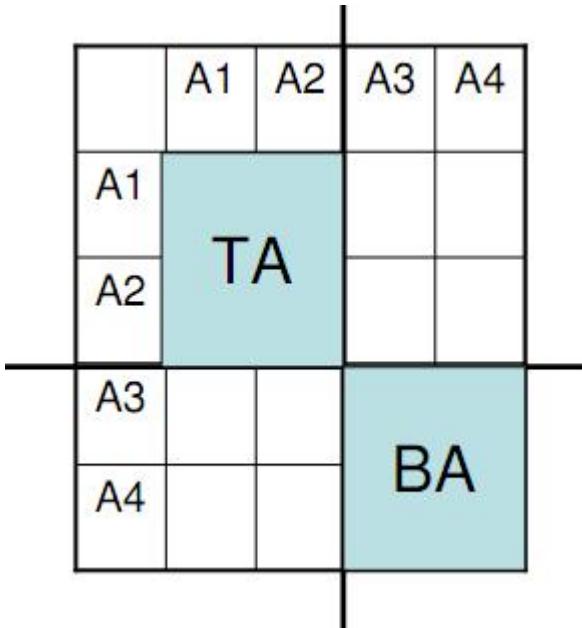
Giảm thiểu các truy xuất trên 2 phân mảnh

# Phân hoạch

Cách làm:

Tìm một điểm trong ma trận ái lực tụ để tạo  
ra hai tập các thuộc tính: TA và BA

# Phân hoạch



- $Q = \{q_1, q_2, \dots, q_n\}$
- $AQ(q_i) = \{A_j | use(q_i, A_j) = 1\}$
- $TQ = \{q_i | AQ(q_i) \subseteq TA\}$
- $BQ = \{q_i | AQ(q_i) \subseteq BA\}$
- $OQ = Q - \{TQ \cup BQ\}$

??? Tìm một vị trí trong ma trận ái lực tụ (CA) để tạo ra 2 tập thuộc tính: TA và BA

Vị trí tốt nhất để phân chia là vị trí sinh ra các tập TQ và BQ sao cho: tổng các truy suất chỉ 1 mảnh là lớn nhất còn tổng các truy suất cả 2 mảnh là nhỏ nhất.

# Phân hoạch

## Các phương trình chi phí

- $CQ = \sum_{q_i \in Q} \sum_{\forall S_j} ref_j(q_i) acc_j(q_i)$
- $CTQ = \sum_{q_i \in TQ} \sum_{\forall S_j} ref_j(q_i) acc_j(q_i)$
- $CBQ = \sum_{q_i \in BQ} \sum_{\forall S_j} ref_j(q_i) acc_j(q_i)$
- $COQ = \sum_{q_i \in OQ} \sum_{\forall S_j} ref_j(q_i) acc_j(q_i)$
- $Z = CTQ * CBQ - COQ^2$
- Tìm điểm x trên ma trận ái lực tụ sao cho Z là lớn nhất

# Thuật toán phân hoạch

- Thuật toán PARTITION
  - Input: ma trận ái lực tựu(CA), quan hệ R, ma trận sử dụng thuộc tính, ma trận tần số truy xuất.
  - Output: F: tập các mảnh

Begin

Tính CTQ<sub>n-1</sub>

Tính CBQ<sub>n-1</sub>

Tính COQ<sub>n-1</sub>

best=CTQ<sub>n-1</sub>\*CBQ<sub>n-1</sub>-(COQ<sub>n-1</sub>)<sup>2</sup>

# Thuật toán phân hoạch

do

Begin

for(*i*=n-2;*i*>0;*i*--)

tính CTQ<sub>*i*</sub>

tính CBQ<sub>*i*</sub>

tính COQ<sub>*i*</sub>

*z*=CTQ<sub>*i*</sub>.CBQ<sub>*i*</sub>-COQ<sub>*i*</sub><sup>2</sup>

if *z*>best)

    best=*z*

end for

SHIFT(CA)

End begin

Until không thể SHIFT được nữa

# Thuật toán phân hoạch

Xây dựng lại ma trận theo vị trí xê dịch

$R_1 = \Pi_{TA}(R) \cup K$  //K là tập thuộc tính  
khóa

$R_2 = \Pi_{BA}(R) \cup K$

$F = R_1 \vee R_2$

End PARTITION

## Ví dụ:

### Ma trận sử dụng

- $\text{Use}(q_i, A_j) = \begin{cases} 1 & \text{nếu thuộc tính } A_j \text{ được truy vấn } q_i \text{ tham chiếu} \\ 0 & \text{Trong trường hợp ngược lại} \end{cases}$

- Ví dụ

	A1	A2	A3	A4
q1	1	0	1	0
q2	0	1	1	0
q3	0	1	0	1
q4	0	0	1	1

## Ma trận AA:

Giả sử:  $\text{Ref}_s(q_k) = 1 \quad \forall s, k$

Các tần số truy xuất (3 sites)

- $\text{acc}_1(q_1)=15 ; \text{acc}_2(q_1)=20 ; \text{acc}_3(q_1)=10$
- $\text{acc}_1(q_2)=5 ; \text{acc}_2(q_2)=0 ; \text{acc}_3(q_2)=0$
- $\text{acc}_1(q_3)=25 ; \text{acc}_2(q_3)=25 ; \text{acc}_3(q_3)=25$
- $\text{acc}_1(q_4)=3 ; \text{acc}_2(q_4)=0 ; \text{acc}_3(q_4)=0$

# Ma trận CA:

- Sắp thứ tự các hàng theo cột

	A1	A3	A2	A4
A1	45	45	0	0
A2	0	5	80	75
A3	45	53	5	3
A4	0	3	75	78

	A1	A3	A2	A4
A1	45	45	0	0
A3	45	53	5	3
A2	0	5	80	75
A4	0	3	75	78

# Ví dụ phân hoạch

	A	A	A	A
	1	3	2	4
A1				
A3				
A2				
A4				

- $Q = \{q_1, q_2, q_3, q_4\}$
- $AQ(q_1) = \{A_1, A_3\}$
- $AQ(q_2) = \{A_2, A_3\}$
- $AQ(q_3) = \{A_2, A_4\}$
- $AQ(q_4) = \{A_3, A_4\}$
- ❖  $TA = \{A_1, A_3, A_2\}$
- ❖  $BA = \{A_4\}$
- ❖  $TQ = \{q_1, q_2\}$
- ❖  $BQ = \{\Phi\}$
- ❖  $OQ = \{q_3, q_4\}$
- ❖  $CTQ = 45 + 5 = 50$
- ❖  $CBQ = 0$
- ❖  $COQ = 78$
- ❖  $Z = 50 * 0 - 78^2 = -78^2$

# Ví dụ phân hoạch

	A1	A2	A3	A4
1				
3				
A1				
A3				
A2				
A4				

- $Q = \{q_1, q_2, q_3, q_4\}$
- $AQ(q_1) = \{A_1, A_3\}$
- $AQ(q_2) = \{A_2, A_3\}$
- $AQ(q_3) = \{A_2, A_4\}$
- $AQ(q_4) = \{A_3, A_4\}$

- ❖  $TA = \{A_1, A_3\}$
- ❖  $BA = \{A_2, A_4\}$
- ❖  $TQ = \{q_1\}$
- ❖  $BQ = \{q_3\}$
- ❖  $OQ = \{q_2, q_4\}$
- ❖  $CTQ = 45$
- ❖  $CBQ = 75$
- ❖  $COQ = 8$
- ❖  $Z = 45 * 75 - 8^2 = 3375 - 64 = 3311$

# Ví dụ phân hoạch

	A	A	A	A
	1	3	2	4
A1				
A3				
A2				
A4				

- $Q = \{q_1, q_2, q_3, q_4\}$
  - $AQ(q_1) = \{A_1, A_3\}$
  - $AQ(q_2) = \{A_2, A_3\}$
  - $AQ(q_3) = \{A_2, A_4\}$
  - $AQ(q_4) = \{A_3, A_4\}$
- 
- ❖  $TA = \{A_1\}$
  - ❖  $BA = \{A_3, A_2, A_4\}$
  - ❖  $TQ = \{\Phi\}$
  - ❖  $BQ = \{q_2, q_3, q_4\}$
  - ❖  $OQ = \{q_1\}$
  - ❖  $CTQ = 0$
  - ❖  $CBQ = 83$
  - ❖  $COQ = 45$
  - ❖  $Z = 0 * 83 - 45^2 = -45^2$

# Thiết kế phân mảnh dọc

## SHIFT

Áp dụng cho ma trận CA từ quan hệ Project, kết quả là định nghĩa các mảnh F: Proj={Project1, Project2}

Trong đó:

Project1={A1, A3}

Project2= {A1, A2, A4}

Vì thế:

Project1={JNO, Budget}

Project2={JNO, JNAME, LOC}

JNO là thuộc tính khoá của Project

	A	A	A	A
1	3	2	4	
A1				
A3				
A2				
A4				

# Nội dung

- 1. Giới thiệu**
- 2. Mục tiêu của thiết kế CSDL phân tán**
- 3. Các bước thiết kế CSDL phân tán**
- 4. Các chiến lược phân tán dữ liệu**
- 5. Các phương pháp thiết kế CSDL phân tán**
- 6. Thiết kế phân mảnh**
- 7. Thiết kế định vị dữ liệu**
- 8. Cấp phát tài nguyên trong hệ phân tán**

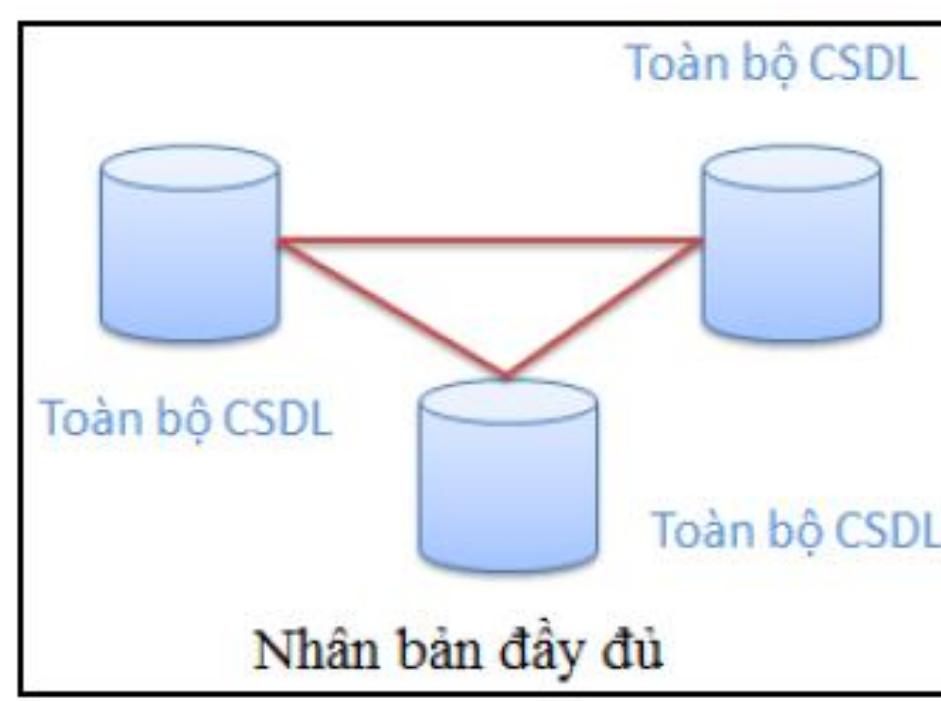
# **Thiết kế định vị dữ liệu**

- ❖ Thiết kế định vị dữ liệu dựa trên nguyên tắc:
  - ▶ Nếu tần số sử dụng của đoạn  $R_i$  tại trạm  $j$  vượt quá một ngưỡng  $\theta$  nào đó thì đoạn  $R_i$  sẽ được đặt tại trạm  $j$ .
  - ▶ Mỗi đoạn có thể được đặt trên nhiều trạm.

# Thiết kế định vị dữ liệu

❖ Có 3 cách định vị:

- ▶ Nhân bản đầy đủ: Mỗi trạm chứa toàn bộ CSDL.
  - **Ưu điểm:** Xử lý nhanh.
  - **Nhược điểm:** Mất thời gian cập nhật, sửa đổi.



# Thiết kế định vị dữ liệu

❖ Có 3 cách định vị:

▶ Không nhân bản:

- **Ưu điểm:** Cập nhật nhanh.

- **Nhược điểm:** Xử lý chậm trong việc thống kê, tổng hợp dữ liệu.

- **Ví dụ:**

$$\begin{cases} R \rightarrow R1, R2, R3 \\ S \rightarrow S1, S2, S3 \end{cases}$$

▶ Kết hợp.

# Thiết kế định vị dữ liệu

## › Ví dụ:

Một công ty có hệ thống mạng máy tính với 3 trạm tương ứng với 3 đơn vị có mã số: 1, 2 và 3.

- › **Trạm 1:** Ban quản lý dự án.
- › **Trạm 2 và 3** dành cho hai đơn vị 2, 3. Trong mỗi trạm:
  - Phòng nhân sự thường xuyên cần các thông tin: **MasoNV, HoDem, Ten, NgaySinh, GioiTinh, DiaChi.**
  - Phòng kế toán thường xuyên cần các thông tin về: **MasoNV, MasoNGS, MasoDV, Luong.**

# Thiết kế định vị dữ liệu

› Các bảng cơ sở dữ liệu như sau:

**DonVi** (**MasoDV**, **TenDV**, **MasoNQL**, **Ngay**)

**NhanVien** (**MasoNV**, **HoDem**, **Ten**, **NgaySinh**,  
**GioiTinh**, **DiaChi**, **Luong**, **MaSoNGS**,  
**MasoDV**)

**DuAn** (**MasoDA**, **TenDA**, **DiaDiemDA**, **MasoDV**)

**NV\_DA** (**MasoNV**, **MasoDA**, **SoGio**)

# Thiết kế định vị dữ liệu

► Thiết kế định vị như sau:

- Trạm 1: Toàn bộ CSDL

- Trạm 2:

o  $NV2 = \delta_{MasoDV=2}(NhanVien)$

o  $NV2$  tách thành:  $\begin{cases} NV21 = \Pi_{MasoNV, HoDem, Ten, NgaySinh, GioiTinh, DiaChi}(NV2) \\ NV22 = \Pi_{MasoNV, MasoNGS, MasoDV, Luong}(NV2) \end{cases}$

o  $DA2 = \delta_{MasoDV=2}(DuAn)$

o  $NV\_DA2 = NV\_DA \ltimes_{MasoDV} DA2$

Đến  
trạm 2.

# Nội dung

- 1. Giới thiệu**
- 2. Mục tiêu của thiết kế CSDL phân tán**
- 3. Các bước thiết kế CSDL phân tán**
- 4. Các chiến lược phân tán dữ liệu**
- 5. Các phương pháp thiết kế CSDL phân tán**
- 6. Thiết kế phân mảnh**
- 7. Thiết kế định vị dữ liệu**
- 8. Cấp phát tài nguyên trong hệ phân tán**
- 9. Kiểm soát dữ liệu ngũ nghĩa**

# Cấp phát tài nguyên trong hệ phân tán

## 1. Bài toán cấp phát (allocation problem):

Giả sử có một tập các mảnh  $F = \{F_1, F_2, \dots, F_k\}$  và một mạng máy tính bao gồm các vị trí  $S = \{S_1, S_2, \dots, S_m\}$  trên đó có một tập các ứng dụng  $Q = \{Q_1, Q_2, \dots, Q_q\}$  đang thực thi.

Hãy tìm một phân phối tối ưu các mảnh  $F$  cho các vị trí  $S$ .

Một phân phối được gọi là tối ưu nếu thỏa mãn hai yếu tố sau:

# Cấp phát tài nguyên trong hệ phân tán

a. Chi phí nhỏ nhất: hàm chi phí bao gồm:

- Chi phí lưu mỗi mảnh dữ liệu  $F_i$  tại vị trí  $S_j$
- Chi phí vấn tin  $F_i$  tại vị trí  $S_j$
- Chi phí cập nhật  $F_i$  tại tất cả các vị trí có chứa nó
- Chi phí truyền dữ liệu.

Bài toán cấp phát sẽ tìm một lược đồ cấp phát với hàm chi phí là cực tiểu.

b. Hiệu quả: chiến lược cấp phát được thiết kế nhằm cực tiểu hóa thời gian thực hiện và tăng tối đa lưu lượng hệ

thông tin.

# Cấp phát tài nguyên trong hệ phân tán

Bài toán cấp phát tổng quát, ký hiệu DAP (Database Allocation Problem), là một bài toán **NP-đầy đủ**. Vì thế hầu hết các nghiên cứu đã được dành cho việc tìm ra được các thuật giải heuristic để có được lời giải tối ưu cho loại bài toán này.

Hiện nay chưa có một mô hình heuristic tổng quát nào nhận một tập các mảnh và sinh ra một chiến lược cấp phát gần tối ưu ứng với các ràng buộc cho trước mà chỉ mới đưa ra một số giả thiết đơn giản hóa và dễ áp dụng cho một số cách đặt vấn đề đơn giản.

# Cấp phát tài nguyên trong hệ phân tán

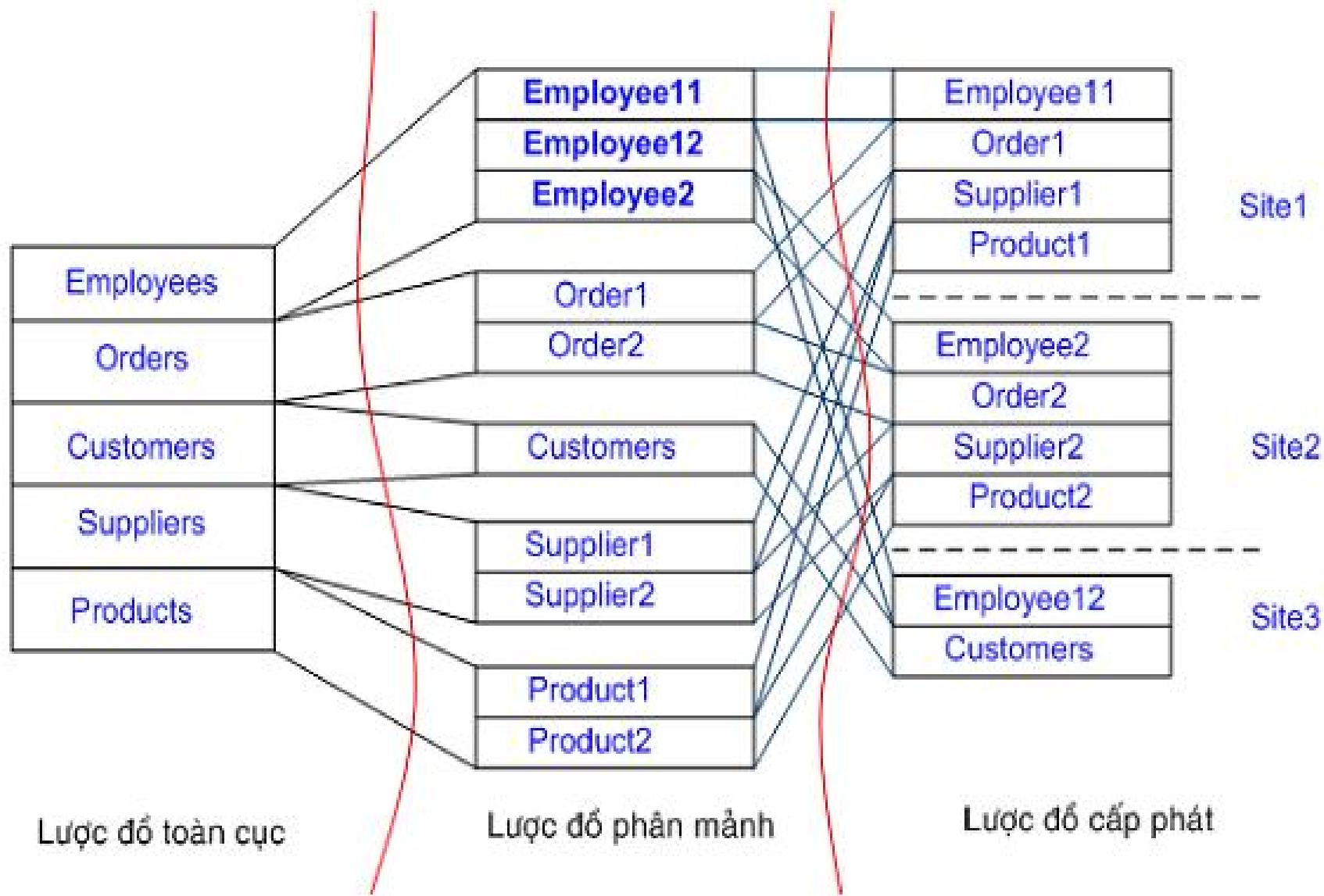
## 2. Thông tin cấp phát

Ở giai đoạn cấp phát, chúng ta cần các thông tin định lượng về cơ sở dữ liệu, về các ứng dụng chạy trên đó, về cấu trúc mạng, về khả năng xử lý và giới hạn lưu trữ của mỗi vị trí trên mạng.

- a. Thông tin về cơ sở dữ liệu
- b. Thông tin về ứng dụng
- c. Thông tin về vị trí

lượng

# Kiến trúc tham khảo



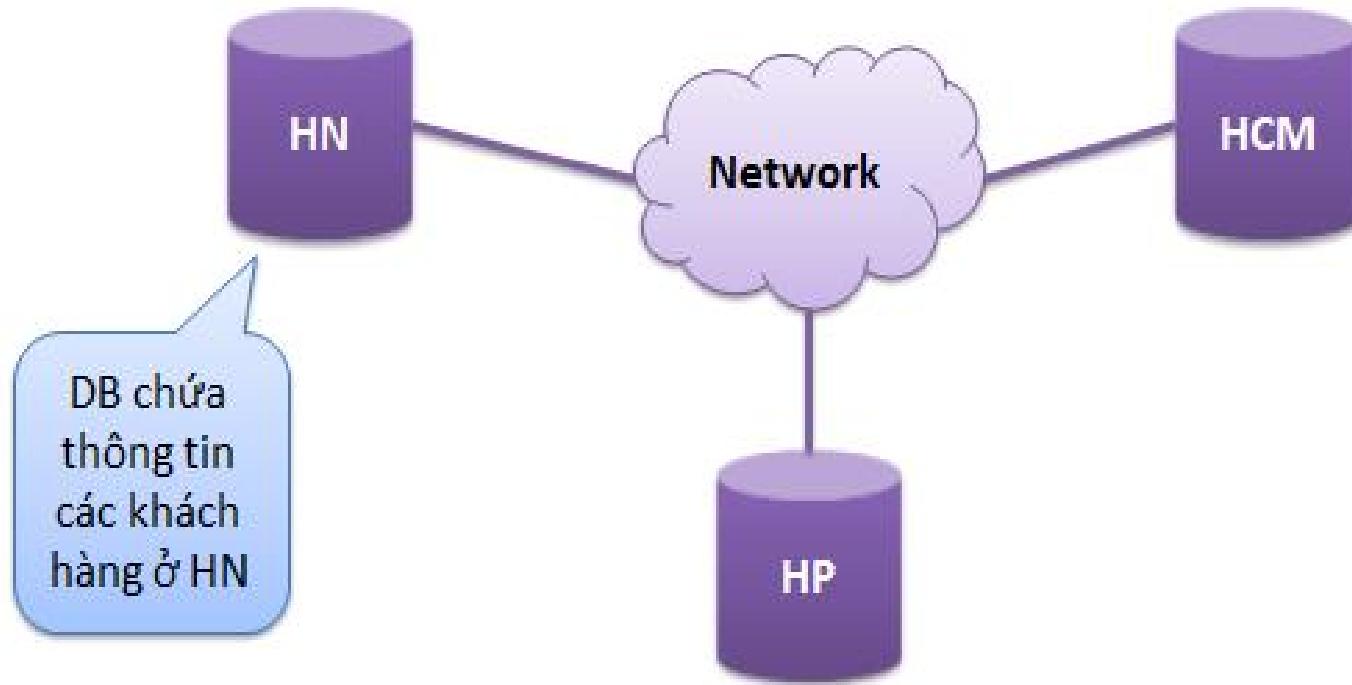
# Câu hỏi cuối chương

- 1. Các vấn đề cần phải làm để thiết kế một HT phân tán**
- 2. Các sản phẩm yêu cầu sau khi phân tích thiết kế một HT phân tán**
- 3. Các chiến lược phân tán dữ liệu**
- 4. Nội dung của phương pháp thiết kế từ trên xuống**
- 5. Các kiểu phân mảnh và các yêu cầu của việc phân mảnh.**
- 6. Bài toán cấp phát**

# Review

- ▶ **Thiết kế hệ thống thông tin có CSDL phân tán bao gồm:**
  - Phân tán và chọn những vị trí đặt dữ liệu;
  - Các chương trình ứng dụng tại các điểm;
  - Thiết kế tổ chức khai thác hệ thống đó trên nền mạng.

# Review



# Review

- ▶ **Khi thiết kế các hệ thống CSDL phân tán người ta thường tập trung xoay quanh các câu hỏi?**
  - Tại sao lại cần phải phân mảnh?
  - Làm thế nào để thực hiện phân mảnh?
  - Phân mảnh nên thực hiện đến mức độ nào?
  - Có cách gì kiểm tra tính đúng đắn của việc phân mảnh?
  - Các mảnh sẽ được cấp phát trên mạng như thế nào?
  - Những thông tin nào sẽ cần thiết cho việc phân mảnh và cấp phát?

# Review

- ▶ **Hệ quản trị cơ sở dữ liệu phân tán (DDBMS)**
  - Cho phép người dùng tạo, sử dụng csdl.
  - Đảm bảo an ninh (cấp phát quyền, 1 nhóm người được sử dụng, ...)
  - Đảm bảo tính trong suốt của csdl (Transperence):
  - Người dùng sử dụng như csdl tập trung.
  - Truy vấn tập trung → Truy vấn phân tán.
- ▶ **Các ứng dụng:**
  - ▶ Ứng dụng cục bộ: Chỉ quan tâm tới dữ liệu ở 1 trạm.
  - ▶ Ứng dụng toàn cục: Liên quan đến nhiều trạm.

# Review

- ▶ **Các bước thiết kế cơ sở dữ liệu phân tán**
  - ❖ Thiết kế lược đồ quan niệm: Đối tượng + Mối quan hệ → ER (Giống csdl tập trung).
  - ❖ Thiết kế logic – vật lý:
- ▶ Thiết kế phân đoạn: Chia các quan hệ thành các đoạn → LĐồ phân đoạn CSDL.
- ▶ Thiết kế định vị: Đặt các đoạn lên các trạm → LĐồ định vị.