CƠ SỞ DỮ LIỆU

Chương 1, 2

Nội dung môn học

- Chương 1: Đại cương về các hệ CSDL.
- Chương 2: Các mô hình dữ liệu.
- Chương 3: Ngôn ngữ định nghĩa và thao tác dữ liệu đối với mô hình quan hệ.
- Chương 4: Lý thuyết thiết kế CSDL quan hệ.
- Chương 5: Tối ưu hóa câu truy vấn
- Chương 6: An toàn và toàn vẹn dữ liệu.
- Chương 7: Tổ chức dữ liệu vật lý

Thông tin

- Thời lượng: 2 tín chỉ
- Yêu cầu: nghe giảng, bài tập về nhà, kiểm tra, thi cuối kỳ
- Đánh giá: QT(BT,KT) 50% - Thi 50%

  (làm tròn điểm !)
- Giáo trình:

  Nguyễn Kim Anh, Nguyên lý của các hệ cơ sở dữ liệu, NXB Đại học Quốc gia, Hà Nội, 2004

Chương 1 - Đại cương về các hệ cơ sở dữ liệu

1.1. Các hệ thống xử lý tệp truyền thống và những hạn chế của nó.

1.2. Các hệ CSDL: khái niệm, khả năng, kiến trúc, người dùng của một hệ quản trị CSDL.

1.3. Sự phân loại các hệ CSDL.

1.1. Các hệ thống xử lý tệp truyền thống

- Bước khởi đầu của quá trình tin học hóa doanh nghiệp.
- Tập trung vào nhu cầu xử lý dữ liệu của các phòng riêng lẻ trong tổ chức mà không xem xét tổng thể tổ chức này.
- Viết một chương trình mới đối với mỗi ứng dụng đơn lẻ, không có kế hoạch, không có mô hình hướng đến sự tăng trưởng.

Các hệ thống xử lý tệp truyền thống

- Mỗi chương trình ứng dụng định nghĩa và quản lý các tệp dữ liệu của riêng nó.
- Trước khi xuất hiện các phần mềm hệ quản trị CSDL, trong quá khứ các hệ thống trên cơ sở tệp đã được tạo lập để xử lý một số lượng lớn dữ liệu.

Hạn chế của các hệ thống xử lý tệp truyền thống

- Dư thừa và không nhất quán dữ liệu
- Khó khăn trong truy nhập dữ liệu
- Cô lập và hạn chế chia sẻ dữ liệu
- Các vấn đề về an toàn và toàn vẹn
- Các vấn đề về độ tin cậy
- Sự phụ thuộc dữ liệu của các chương trình ứng dụng

1.2. Các hệ cơ sở dữ liệu

- CSDL (database) là gì ?
- Tại sao phải sử dụng CSDL ?
- Tại sao phải tìm hiểu về các hệ CSDL (database systems) ?

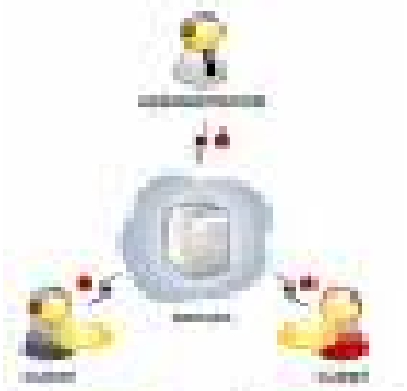
Ví dụ về quản lý đào tạo

- Thông tin cần quan tâm
  - Khóa học, lớp học, sinh viên, giáo viên, môn học,…
  - Thông tin về sinh viên: thông tin cá nhân, thông tin học tập,…
  - Thông tin về môn học: khối lượng, giáo viên, lịch học,…
- Cần *lưu trữ* những thông tin đa dạng

⟶ **Cơ sở dữ liệu**

Ví dụ: khai thác thông tin

- Sinh viên
  - Các môn học trong CTĐT?
  - Điểm thi môn "Cơ sở dữ liệu"?
- Giáo viên
  - Danh sách sinh viên lớp 111565 ?
  - Thời khóa biểu của lớp KHMT-K64 ?
- Giáo vụ
  - Danh sách sinh viên K62 tốt nghiệp loại giỏi ?...



→ **Phần mềm ứng dụng**

"Hình dung" về xây dựng một CSDL

- Yêu cầu
  - Lưu trữ thông tin cần thiết một cách chính xác
  - Truy xuất thông tin hiệu quả
- Thực hiện
  - Xác định yêu cầu nghiệp vụ
  - Xác định thông tin cần lưu trữ
  - Xác định cách thức lưu trữ
- Cần công cụ trợ giúp xây dựng một CSDL

→ **Phần mềm quản trị CSDL**

Cơ sở dữ liệu (database)

- Là một tập hợp các dữ liệu
  - Biểu diễn một vài khía cạnh nào đó của thế giới thực
  - Có liên hệ logic thống nhất
  - Được thiết kế và bao gồm những dữ liệu phục vụ một mục đích nào đó.
- Là một bộ sưu tập các dữ liệu tác nghiệp được lưu trữ lại và được các hệ ứng dụng của một xí nghiệp cụ thể nào đó sử dụng.

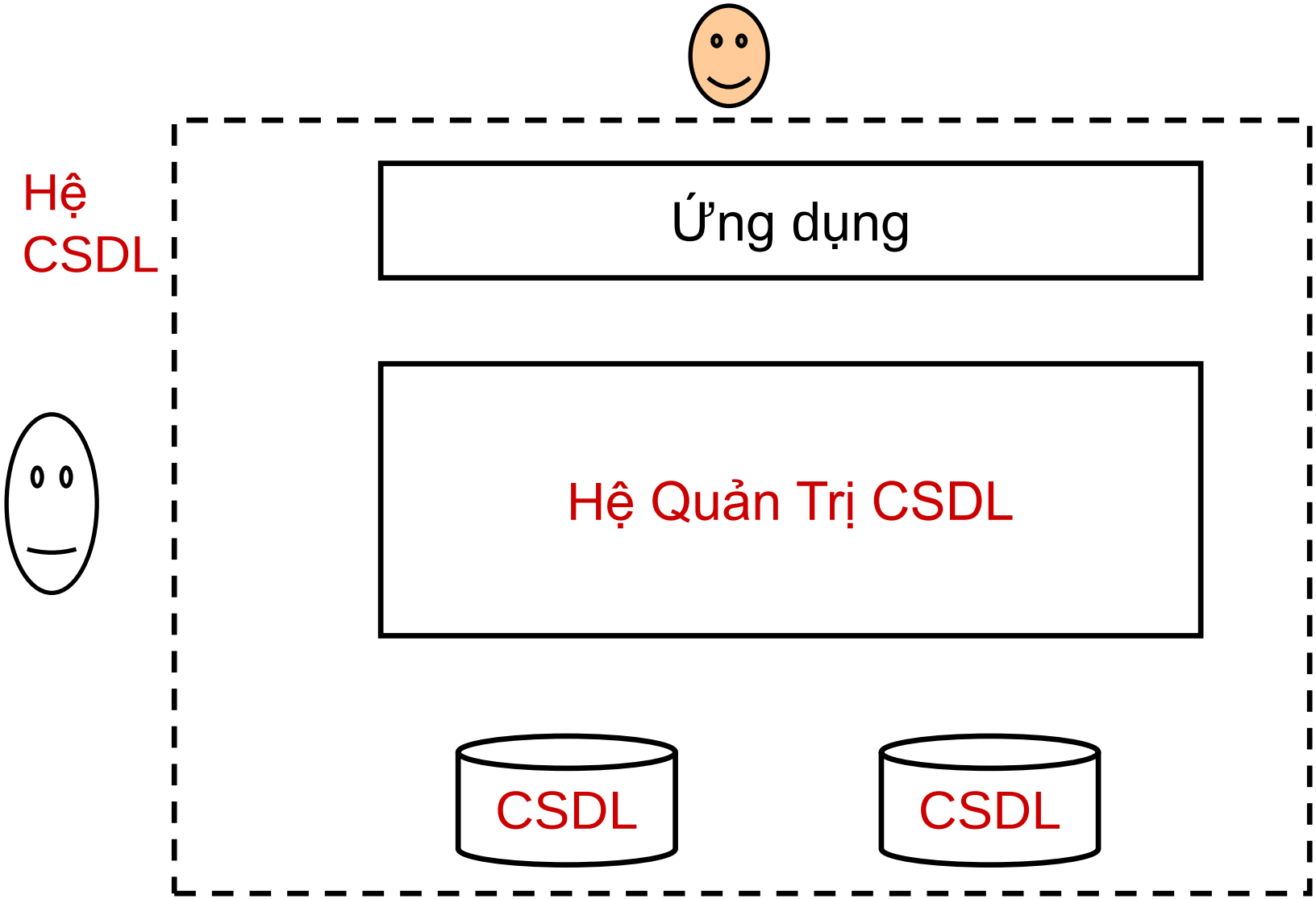
Hệ quản trị cơ sở dữ liệu
(Database Management System-DBMS)

- Là một hệ thống phần mềm cho phép
  - Định nghĩa, tạo lập: xác định kiểu, cấu trúc, ràng buộc dữ liệu, lưu trữ dữ liệu trên các thiết bị nhớ.
  - Thao tác: truy vấn, cập nhật, kết xuất,… các CSDL cho các ứng dụng khác nhau
- Ví dụ: MS SQL Server, mySQL, PostGreSQL, DB2, MS Access, Oracle, FoxPro,…

Hệ cơ sở dữ liệu

- Là một hệ thống gồm 4 thành phần
  - Hệ quản trị CSDL
  - Phần cứng
  - CSDL và phần mềm ứng dụng
  - Những người sử dụng

- Ví dụ: Hệ quản lý đào tạo, hệ quản lý nhân sự, hệ quản lý kinh doanh,…


Hệ
CSDL
Ứng dụng
Hệ Quản Trị CSDL
CSDL
CSDL

Các tính năng của hệ quản trị CSDL

- Quản lý dữ liệu tồn tại lâu dài
  - Định nghĩa dữ liệu
  - Quản lý lưu trữ
- Truy xuất dữ liệu một cách hiệu quả
  - Biểu diễn các thao tác dữ liệu
  - Xử lý câu hỏi
  - Quản trị giao dịch

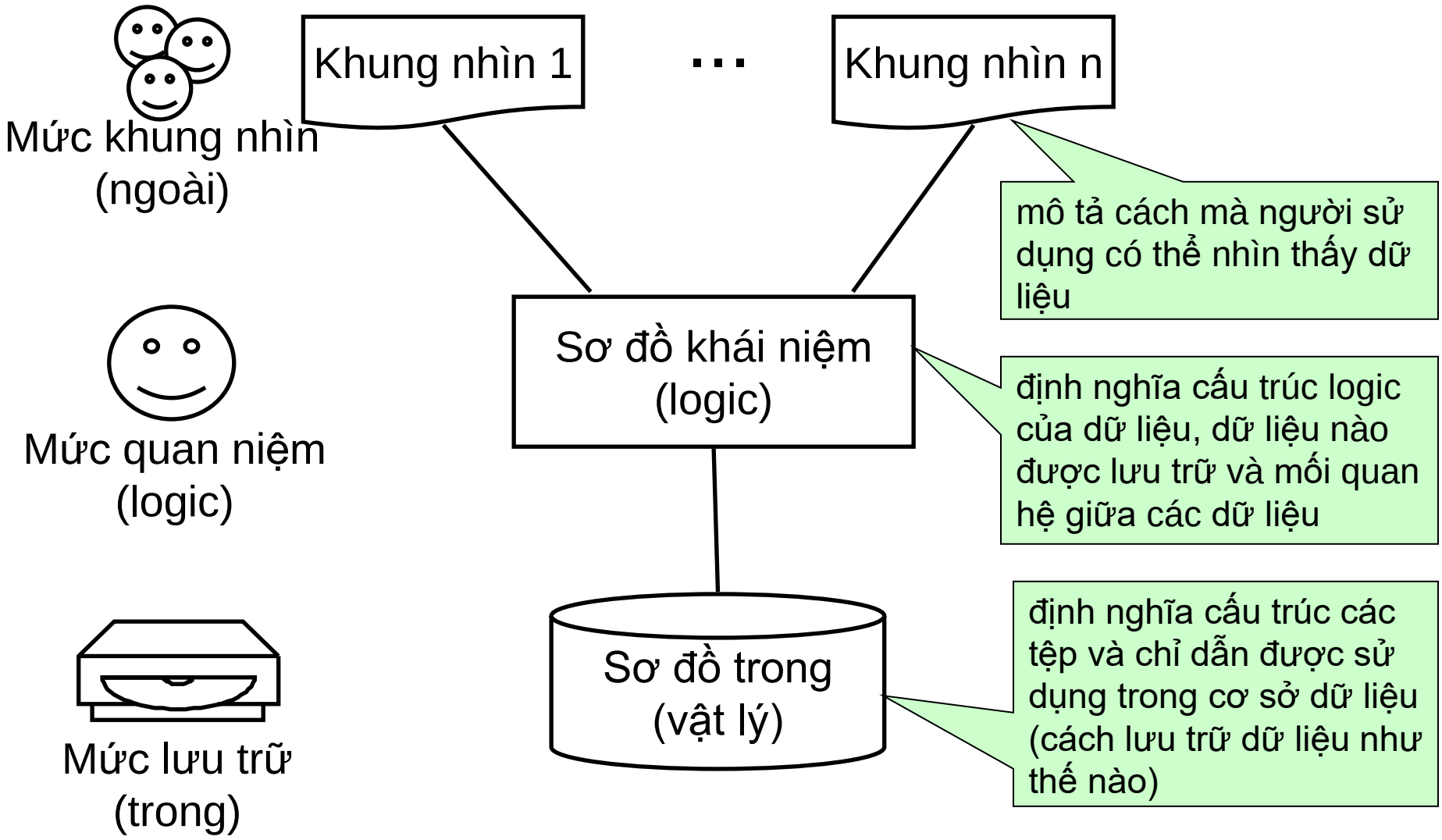
Các tính năng của hệ quản trị CSDL

- Hỗ trợ ít nhất một mô hình dữ liệu
- Đảm bảo tính độc lập dữ liệu
- Hỗ trợ các ngôn ngữ cấp cao nhất định cho phép người sử dụng định nghĩa cấu trúc của dữ liệu, truy nhập và thao tác dữ liệu
- Điều khiển truy nhập
- Phục hồi dữ liệu

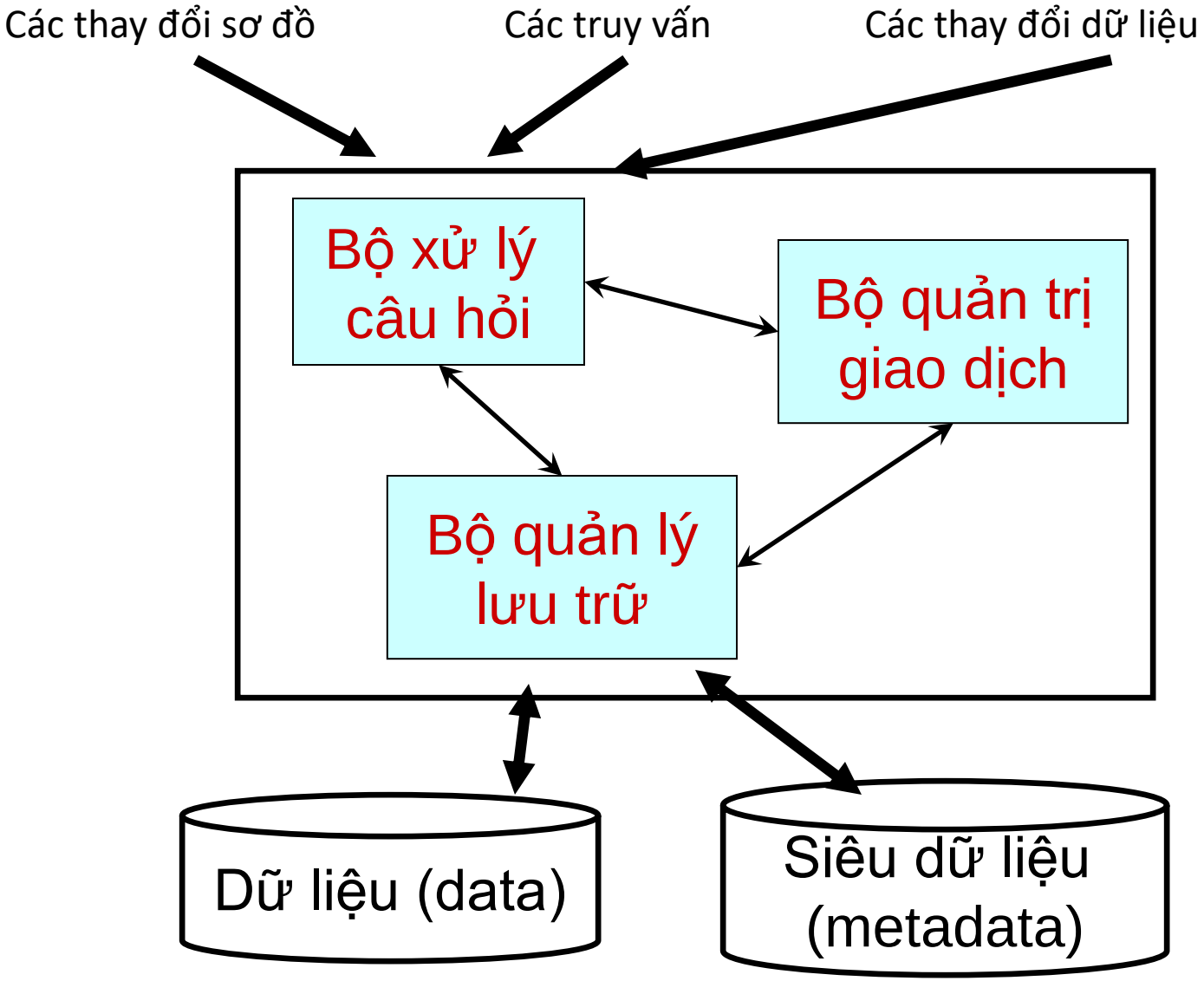
Các ngôn ngữ

- Ngôn ngữ định nghĩa dữ liệu (**D**ata **D**efinition **L**anguage - DDL)
  - Cấu trúc dữ liệu
  - Mối liên hệ giữa các dữ liệu và quy tắc, ràng buộc áp đặt lên dữ liệu
- Ngôn ngữ thao tác dữ liệu (**D**ata **M**anipulation **L**anguage - DML)
  - Tìm kiếm, thêm, xóa, sửa dữ liệu trong CSDL
- Ngôn ngữ điều khiển dữ liệu (**D**ata **C**ontrol **L**anguage - DCL)
  - Thay đổi cấu trúc của các bảng dữ liệu
  - Khai báo bảo mật thông tin
  - Quyền hạn của người dùng trong khai thác CSDL

Sự trừu tượng hóa dữ liệu

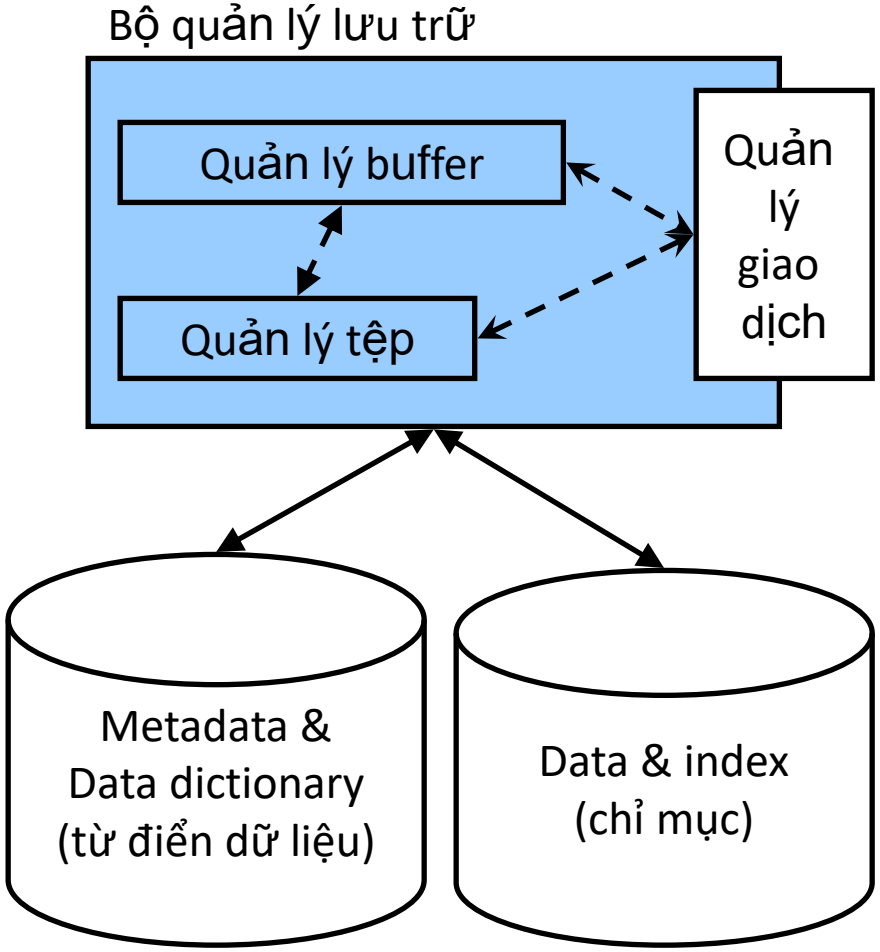


Kiến trúc của một hệ quản trị CSDL



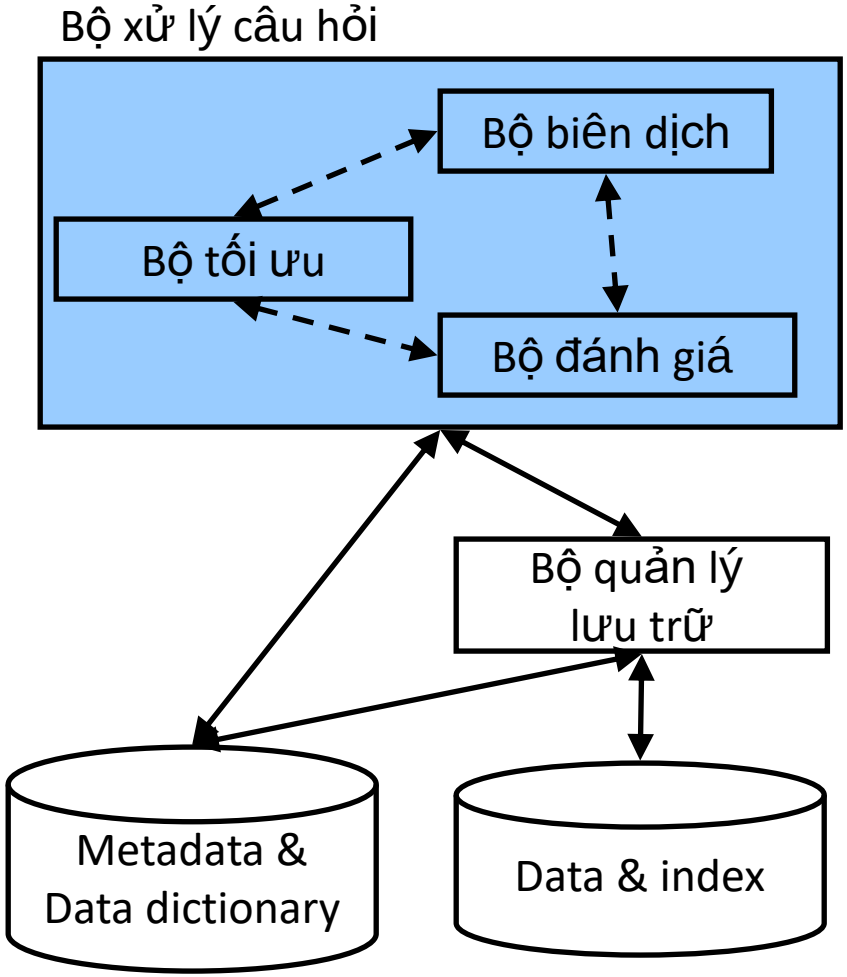
Quản lý lưu trữ

- Yêu cầu
  - lưu trữ và truy xuất dữ liệu trên các thiết bị nhớ
- Thực hiện
  - Tổ chức tối ưu dữ liệu trên thiết bị nhớ
  - Tương tác hiệu quả với bộ quản lý tệp



Xử lý câu hỏi

- Yêu cầu
  - Tìm kiếm dữ liệu trả lời cho một yêu cầu truy vấn.
- Thực hiện
  - Biến đổi truy vấn ở mức cao thành các yêu cầu có thể hiểu được bởi hệ CSDL.
  - Lựa chọn một kế hoạch tốt nhất để trả lời truy vấn này.



Quản trị giao dịch

- Yêu cầu
  - Định nghĩa giao dịch: một tập các thao tác được xử lý như một đơn vị không chia cắt được.
  - Đảm bảo tính đúng đắn và tính nhất quán của dữ liệu.
- Thực hiện
  - Quản lý điều khiển tương tranh.
  - Phát hiện lỗi và phục hồi CSDL

Người dùng

- **Người thiết kế và cài đặt hệ QTCSDL**: chịu trách nhiệm thiết kế và cài đặt các module của hệ QTCSDL và các giao diện dưới hình thức các gói phần mềm

- **Người phát triển công cụ**: chịu trách nhiệm thiết kế và cài đặt các gói phần mềm hỗ trợ cho việc thiết kê, sử dụng cũng như tăng cường hiệu năng của các hệ CSDL.

Người dùng

- **Người phân tích hệ thống và phát triển ứng dụng**: chịu trách nhiệm xác định yêu cầu của người dùng cuối, xác định các giao dịch cần thiết để đáp ứng các yêu cầu người dùng. Người lập trình ứng dụng cài đặt những yêu cầu này trong chương trình, kiểm thử, gỡ rối, lập tài liệu cho chương trình
- **Người thiết kế CSDL**: chịu trách nhiệm xác định dữ liệu lưu trữ trong CSDL và cấu trúc biểu diễn và lưu trữ những dữ liệu này

Người dùng

- **Người sử dụng cuối**: là người khai thác các hệ CSDL
- **Người quản trị CSDL**: chịu trách nhiệm cho phép truy nhập CSDL, điều phối và kiểm tra sử dụng CSDL, quản lý tài nguyên phần cứng và phân mềm khi cần thiết
- **Người bảo trì hệ thống**: là những người quản trị hệ thống chịu trách nhiệm việc hoạt động và bảo trì môi trường (phần cứng và phần mềm) cho hệ CSDL

1.3. Phân loại các hệ CSDL

- Mô hình dữ liệu
  - MH Mạng, MH phân cấp, MH quan hệ, MH hướng đối tượng, ...
- Số người sử dụng
  - Một người dùng, nhiều người dùng
- Tính phân tán của CSDL
  - Tập trung, Phân tán
- Tính thống nhất của dữ liệu
  - Đồng nhất, Không đồng nhất
- ...

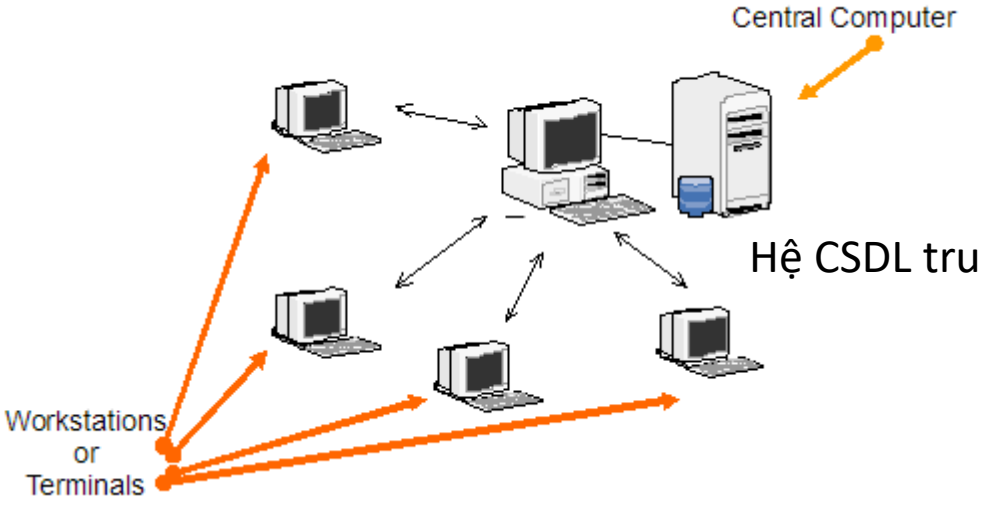
Các hệ CSDL tập trung

- Hệ CSDL cá nhân: một người sử dụng đơn lẻ vừa thiết kế, tạo lập CSDL, cập nhật, bảo trì dữ liệu, lập và hiển thị báo cáo.

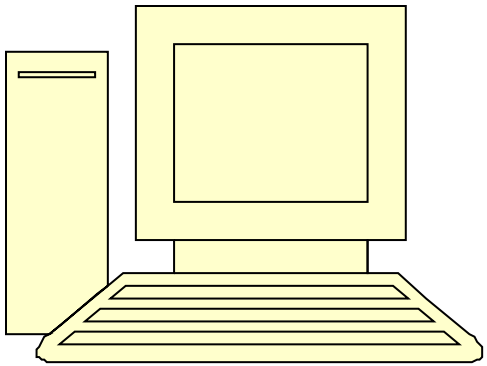
  → đảm nhiệm vai trò: người quản trị CSDL, người viết chương trình ứng dụng, end-user.

- Hệ CSDL trung tâm: dữ liệu được lưu trữ trên một máy tính trung tâm.
- Hệ CSDL khách-chủ:
  - Các máy tính trung tâm lớn → đắt so với các máy nhỏ và máy trạm.
  - Các ứng dụng máy khách truy nhập dữ liệu được quản lý bởi máy chủ.

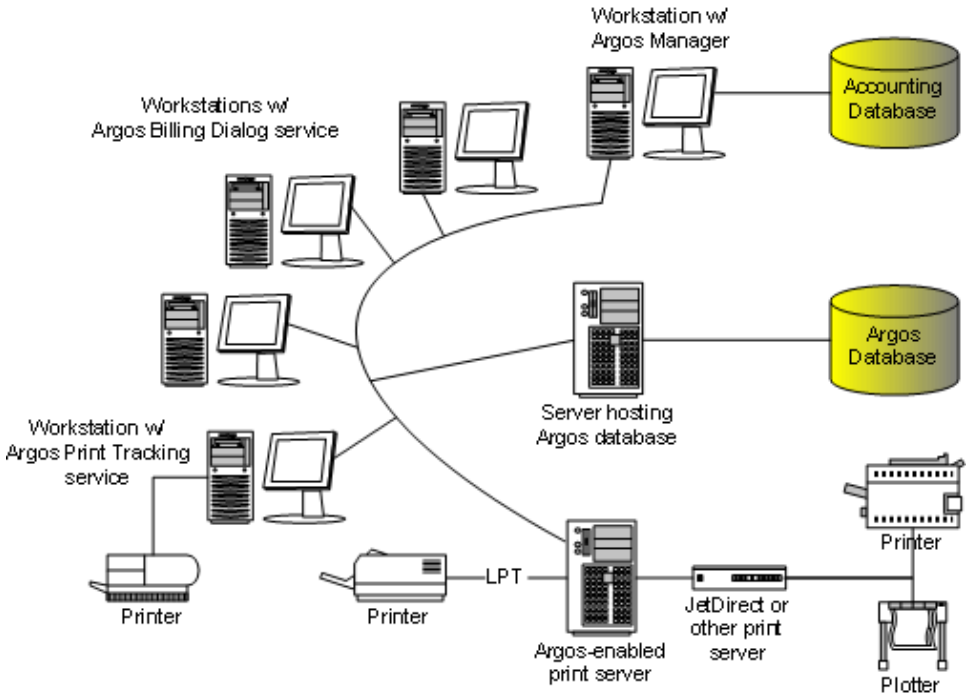
Các hệ CSDL tập trung (tiếp)



Hệ CSDL trung tâm



Hệ CSDL cá nhân

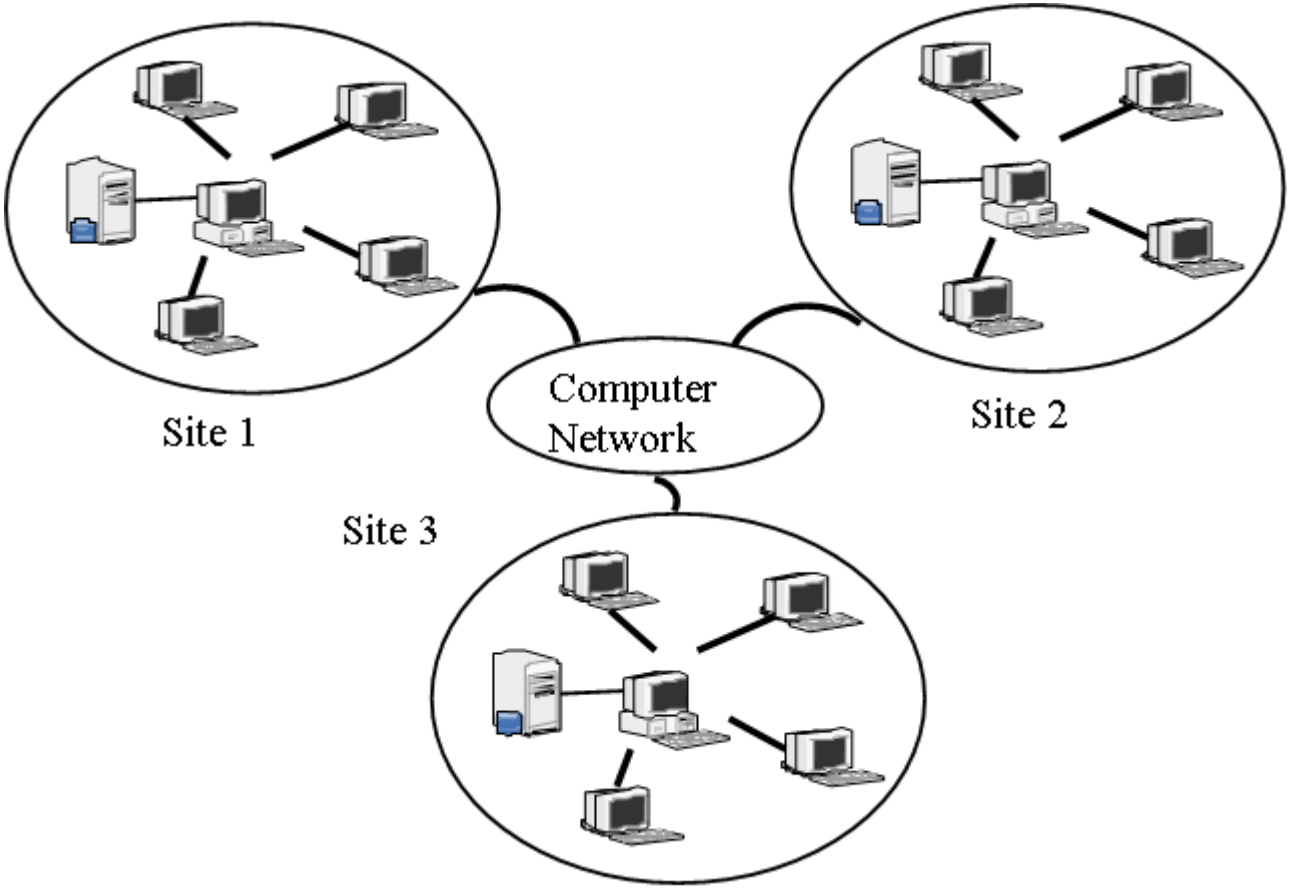


Hệ CSDL khách-chủ

Các hệ CSDL phân tán

- CSDL phân tán? Là một tập các CSDL có quan hệ logic với nhau nhưng được trải ra trên nhiều trạm làm việc của một mạng máy tính.
- Có 2 tính chất: quan hệ logic và phân tán
- Hệ QTCSDL phân tán: Là một hệ thống phần mềm cho phép tạo lập CSDLPT và điều khiển các truy nhập đối với CSDLPT này.
- Chia ra 2 loại: CSDLPT thuần nhất và không thuần nhất

Các hệ CSDL phân tán (tiếp)



Kết luận Chương 1

- CSDL cho phép lưu trữ và khai thác dữ liệu một cách thống nhất và hiệu quả (đặc biệt trong trường hợp khối lượng dữ liệu lớn).
- Sự trừu tượng về dữ liệu và tính độc lập dữ liệu cho phép phát triển ứng dụng "dễ dàng hơn".
- Hệ quản trị CSDL cung cấp các công cụ hữu hiệu trợ giúp việc tạo lập CSDL và phát triển ứng dụng

Chương 2 – Các mô hình dữ liệu

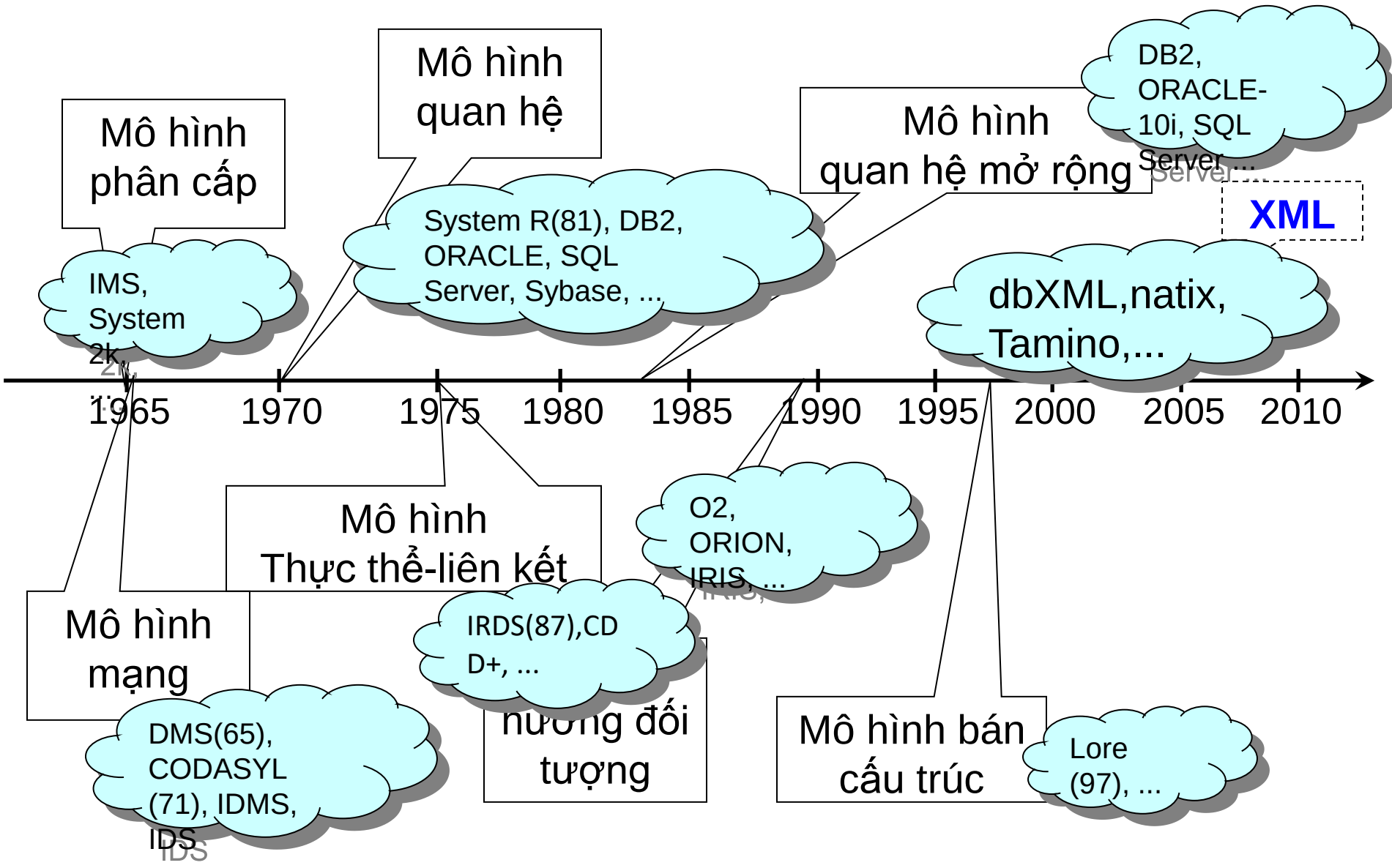
- Tổng quan về mô hình dữ liệu
- Mô hình phân cấp
- Mô hình mạng
- Mô hình quan hệ
- Mô hình thực thể liên kết
- Mô hình hướng đối tượng

Tổng quan về mô hình dữ liệu

- Mô hình dữ liệu [Codd, 1980] gồm:
  - Một tập hợp các cấu trúc của dữ liệu
  - Một tập hợp các phép toán để thao tác với các dữ liệu
  - Một tập hợp các ràng buộc về dữ liệu
- Mô hình dữ liệu là một tập hợp các khái niệm dùng để mô tả:
  - Dữ liệu
  - Ngữ nghĩa của dữ liệu
  - Các mối quan hệ trong dữ liệu
  - Các ràng buộc dữ liệu

Tổng quan về mô hình dữ liệu

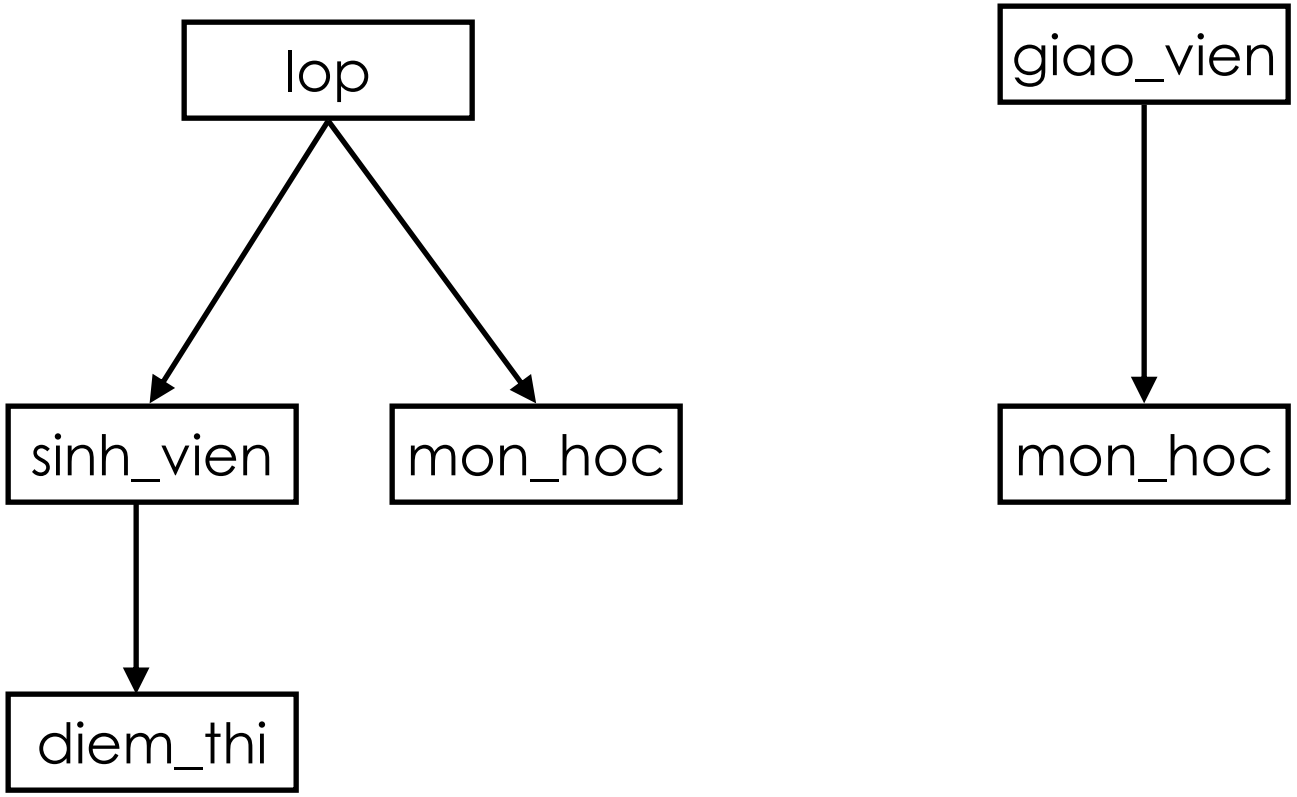
- Nhiều mô hình còn bao gồm cả một tập các phép toán để thao tác các dữ liệu
- Mô hình thuộc dạng ngữ nghĩa: tập trung về ngữ nghĩa của dữ liệu như mô hình thực thể liên kết, sử dụng để hỗ trợ người dùng có cái nhìn khái quát về dữ liệu
- Mô hình thuộc dạng khái niệm: tập trung vào cách thức tổ chức dữ liệu tại mức khái niệm như mô hình mạng, mô hình liên kết, mô hình quan hệ, độc lập với DBMS và hệ thống phần cứng để cài đặt cơ sở dữ liệu


Mô hình phân cấp
Mô hình quan hệ
Mô hình quan hệ mở rộng
DB2, ORACLE-10i, SQL Server,...
XML
IMS, System 2k, ...
System R(81), DB2, ORACLE, SQL Server, Sybase, ...
dbXML,natix, Tamino,...
1965
1970
1975
1980
1985
1990
1995
2000
2005
2010
Mô hình Thực thể-liên kết
O2, ORION, IRIS, ...
Mô hình mạng
IRDS(87),CDD+, ...
DMS(65), CODASYL (71), IDMS, IDS
hướng đối tượng
Mô hình bán cấu trúc
Lore (97), ...

Mô hình dữ liệu phân cấp (*Hierarchical data model*)

- Ra đời những năm 60-65
- Biểu diễn bằng cây
  - Quan hệ cha-con
  - Mỗi nút có 1 cha duy nhất
  - 1 CSDL = 1 tập các cây = 1 rừng
- Các khái niệm cơ bản
  - Bản ghi
  - Móc nối
  - Các phép toán: GET, GET UNIQUE, GET NEXT, GET NEXT WITHIN PARENT,…

Ví dụ

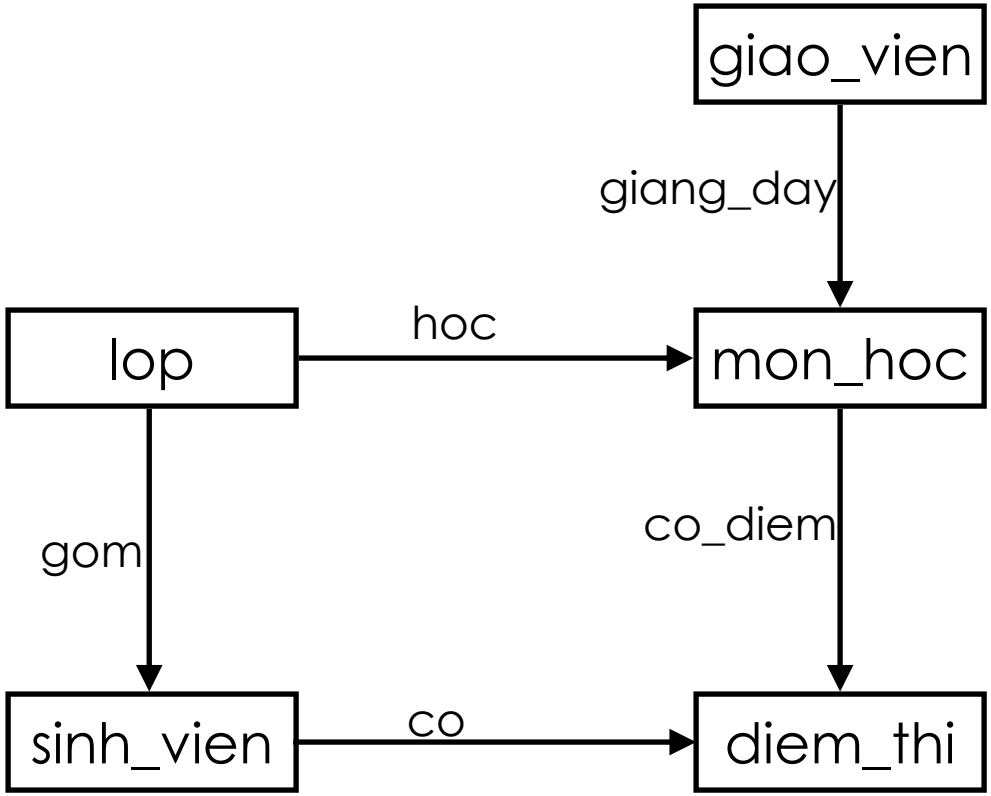


- Ưu điểm
  - Dễ xây dựng và thao tác
  - Tương thích với các lĩnh vực tổ chức phân cấp
  - Ngôn ngữ thao tác đơn giản: duyệt cây.
- Nhược điểm:
  - Sự lặp lại của các kiểu bản ghi → dữ liệu dư thừa và không nhất quán.
    - Giải pháp: bản ghi ảo
  - Hạn chế trong biểu diễn ngữ nghĩa của các móc nối giữa các bản ghi (chỉ cho phép quan hệ 1-n)

Mô hình dữ liệu mạng (*Network data model*)

- Sự ra đời: Sử dụng phổ biến từ những năm 60, được định nghĩa lại vào năm 71
- Biểu diễn bằng đồ thị có hướng
- Các khái niệm cơ bản
  - Tập bản ghi (record): Kiểu bản ghi (record type); Các trường (field)
  - Móc nối: Tên của móc nối; Chủ (owner) – thành viên (member): theo hướng của móc nối; Kiểu móc nối: 1-1, 1-n, đệ quy
  - Các phép toán
    - Duyệt: FIND, FIND member, FIND owner, FIND NEXT
    - Thủ tục: GET

Ví dụ


giao_vien
giang_day
lop
hoc
mon_hoc
gom
co_diem
sinh_vien
co
diem_thi

- Ưu điểm
  - Đơn giản
  - Có thể biểu diễn các ngữ nghĩa đa dạng với kiểu bản ghi và kiểu móc nối
  - Truy vấn thông qua phép duyệt đồ thị (navigation)
- Nhược điểm:
  - Số lượng các con trỏ lớn
  - Hạn chế trong biểu diễn ngữ nghĩa của các móc nối giữa các bản ghi

Mô hình dữ liệu quan hệ

- Sự ra đời: vào năm 1970[Codd, 1970]
- Dữ liệu được biểu diễn dưới dạng bảng
- Là mô hình dữ liệu khái niệm phổ biến cho đến tận thời điểm hiện tại
- Dựa trên lý thuyết toán học, đồng thời cũng gần với cấu trúc tệp và cấu trúc dữ liệu nên có hai loại thuật ngữ liên quan:
  - Thuật ngữ toán học: quan hệ, bộ, thuộc tính
  - Thuật ngữ hướng dữ liệu: bảng, bản ghi, trường

Ví dụ mô hình dữ liệu quan hệ

MON_HOC

maHP	tenHP	soTC
CNTT01	Nhập môn CSDL	4
CNTT02	Truyền DL và mạng	4
CNTT03	Phân tích và thiết kế hệ thống	4
HTTT01	Quản lý dự án	3

LOP

malop	lop	khoa	GVCN	loptruong
IT4	Tin 4	CNTT	Ng. V. Anh	Trần T. Bình
IT5	Tin 5	CNTT	Lê A. Văn	Ng. Đ. Trung
IT6	Tin 6	CNTT	Ng. T. Thảo	Trần M. Quế
IT7	Tin 7	CNTT	Ng. V. Quý	Ng. T. Phương

SINH_VIEN

maSV	tenSV	ngaysinh	gt	diachi	malop
SV0011	Trần T. Bình	1/4/1981	0	21 T. Q. B	IT4
SV0025	Ng. Đ. Trung	3/2/1980	1	56 Đ. C. V	IT5
SV0067	Trần M. Quế	26/3/1982	0	45 H. B. T	IT6
SV0034	Ng. T. Phương	29/2/1980	0	86 L. T. N	IT7

Mô hình dữ liệu quan hệ – Các khái niệm

- ***Thuộc tính*** (~trường): là các đặc tính của một đối tượng
- Mỗi thuộc tính được xác định trên một miền giá trị nhất định gọi là ***miền thuộc tính***
- Ví dụ:
  - Sinhviên (MãSV, TênSV, Năm sinh, GiớiTính, ĐịaChỉ)
  - dom(MãSV) = {char(5)}
  - dom(TênSV) = {char(30)}
  - dom(Năm sinh) = {date}
  - dom(GiớiTính) = {0, 1}
  - dom(ĐịaChỉ) = {char(50)}

- ***Quan hệ*** (~bảng): Cho n miền giá trị D_1, D_2, …, D_n không nhất thiết phân biệt, r là một quan hệ trên n miền giá trị đó nếu r là một tập các n-bộ (d_1 , d_2 , …, d_n) sao cho $d_i \in D_i$
- Một quan hệ có thể được biểu diễn dưới dạng một bảng trong đó một dòng trong bảng tương đương với một bộ , một cột trong bảng tương đương với một thuộc tính của quan hệ
- Bậc của một quan hệ là số các thuộc tính trong quan hệ
- Lực lượng của một quan hệ là số các bộ trong quan hệ

Mô hình dữ liệu quan hệ – Các định nghĩa

- **Định nghĩa:** Cho U = {A_1, A_2, …, A_n} là một tập hữu hạn các thuộc tính trong đó dom(A_i) = D_i, r là quan hệ trên tập thuộc tính U ký hiệu là r(U) nếu:

$$r \subseteq D_1 \times D_2 \times ... \times D_n$$

- U được gọi là sơ đồ quan hệ (lược đồ quan hệ)

- Định nghĩa ***Khoá*** của quan hệ r trên tập thuộc tính $U = \{A_1, A_2, \ldots, A_n\}$ là một tập $K \subseteq U$ sao cho với bất kỳ 2 bộ t_1, t_2 thuộc r đều tồn tại một thuộc tính A thuộc K mà $t_1[A] \neq t_2[A]$
- Một quan hệ có thể có nhiều khoá
- Nếu K là khoá của r thì mọi K' sao cho $K \subseteq K'$ đều là khoá của r. K' được gọi là ***siêu khoá*** của r
- **Định nghĩa**: K là ***khoá tối thiểu*** của r nếu K là một khoá của r và bất kỳ tập con thực sự nào của K đều không phải là khoá của r
- **Định nghĩa**: Một tập con $K \subseteq U$ được gọi là ***khoá ngoài*** của quan hệ r(U) tham chiếu đến một quan hệ r' nếu K là khoá chính của r'

Ví dụ

- Quan hệ: SinhViên, Lớp
- Siêu khoá: {MãSV, HọTên};
- Khoá tối thiểu: {MãSV}; {HọTên, NămSinh}
- Khoá ngoài: TênLớp nếu coi nó là khoá chính của quan hệ Lớp

MãSV	HọTên	NămSinh	GiớiTính	TênLớp
001	Nguyễn Văn An	1999	1	KHMT1
002	Nguyễn Văn An	1998	1	KHMT2
003	Lê Văn Cường	1997	1	KHMT2
004	Nguyễn Thùy Linh	1997	0	KTMT1

Tên Lớp	SĩSố
KHMT1	60
KHMT2	65
KTMT1	58

Nhận xét

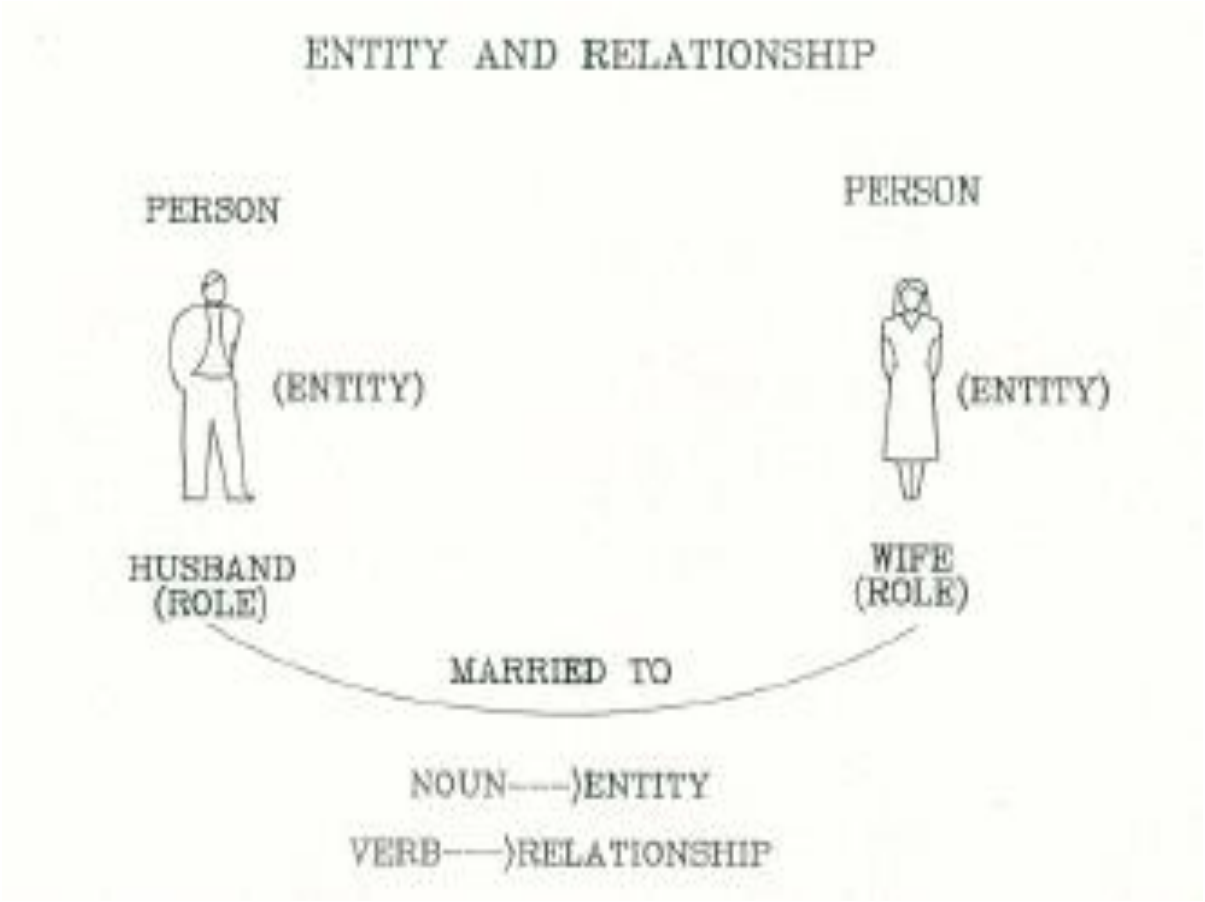
- Ưu điểm
  - Dựa trên lý thuyết tập hợp
  - Khả năng tối ưu hoá các xử lý phong phú
- Nhược điểm
  - Hạn chế trong biểu diễn ngữ nghĩa
  - Cấu trúc dữ liệu không linh hoạt

Mô hình thực thể liên kết (*Entity-Relationship data model*)

- Cho phép mô tả các dữ liệu có liên quan trong một xí nghiệp trong thế giới thực dưới dạng các đối tượng và các mối quan hệ của chúng.
- Được sử dụng cho bước đầu thiết kế CSDL, làm nền tảng để ánh xạ sang một mô hình khái niệm nào đó mà Hệ quản trị CSDL sẽ sử dụng
- Trong mô hình thực thể liên kết, CSDL được mô hình hóa như là:
  - Một tập hợp các thực thể
  - Liên hệ giữa các thực thể này

Mô hình thực thể liên kết – Các khái niệm

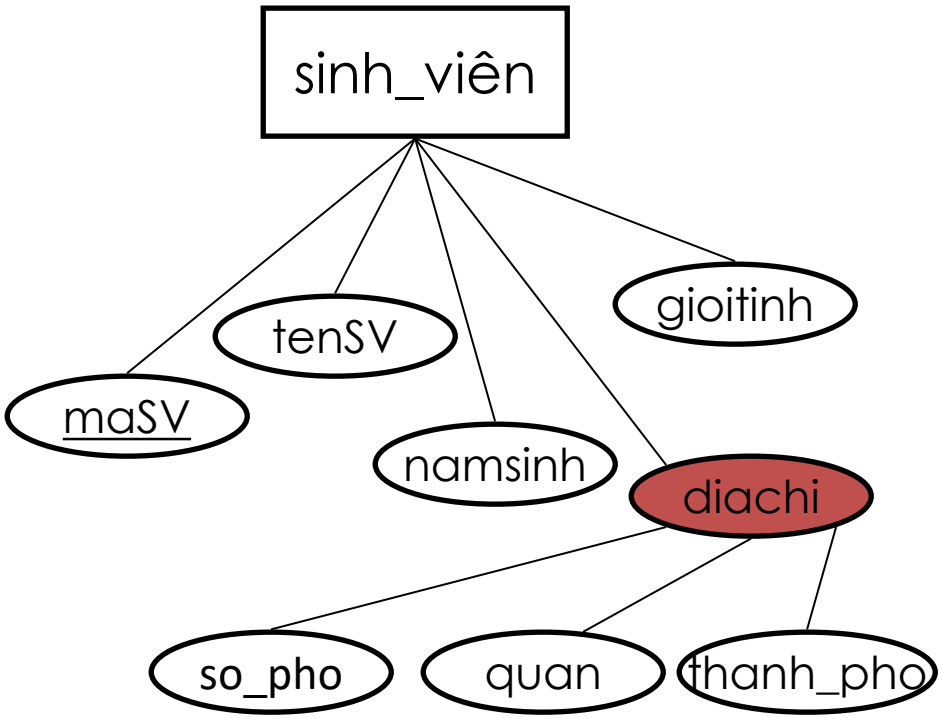
- Thực thể, tập thực thể
- Thuộc tính
- Khoá
- Liên kết, tập liên kết



- Thực thể: một đối tượng trong thế giới thực, tồn tại độc lập và phân biệt được với các đối tượng khác
- Tập thực thể: một tập hợp các thực thể có tính chất giống nhau
- Ví dụ: – Thực thể: một sinh viên, một lớp
  - Tập thực thể: toàn thể sinh viên của 1 lớp, toàn thể các lớp của 1 khoa
- Thuộc tính là đặc tính của một tập thực thể
  - Tập thực thể SinhViên có các thuộc tính như: TênSV, NămSinh,…
- Mỗi thực thể trong tập thực thể có một giá trị đặc tính nằm trong miền giá trị của thuộc tính
  - Sinh viên 1 có: Họtên là Nguyễn Hải Anh, Năm sinh 1980

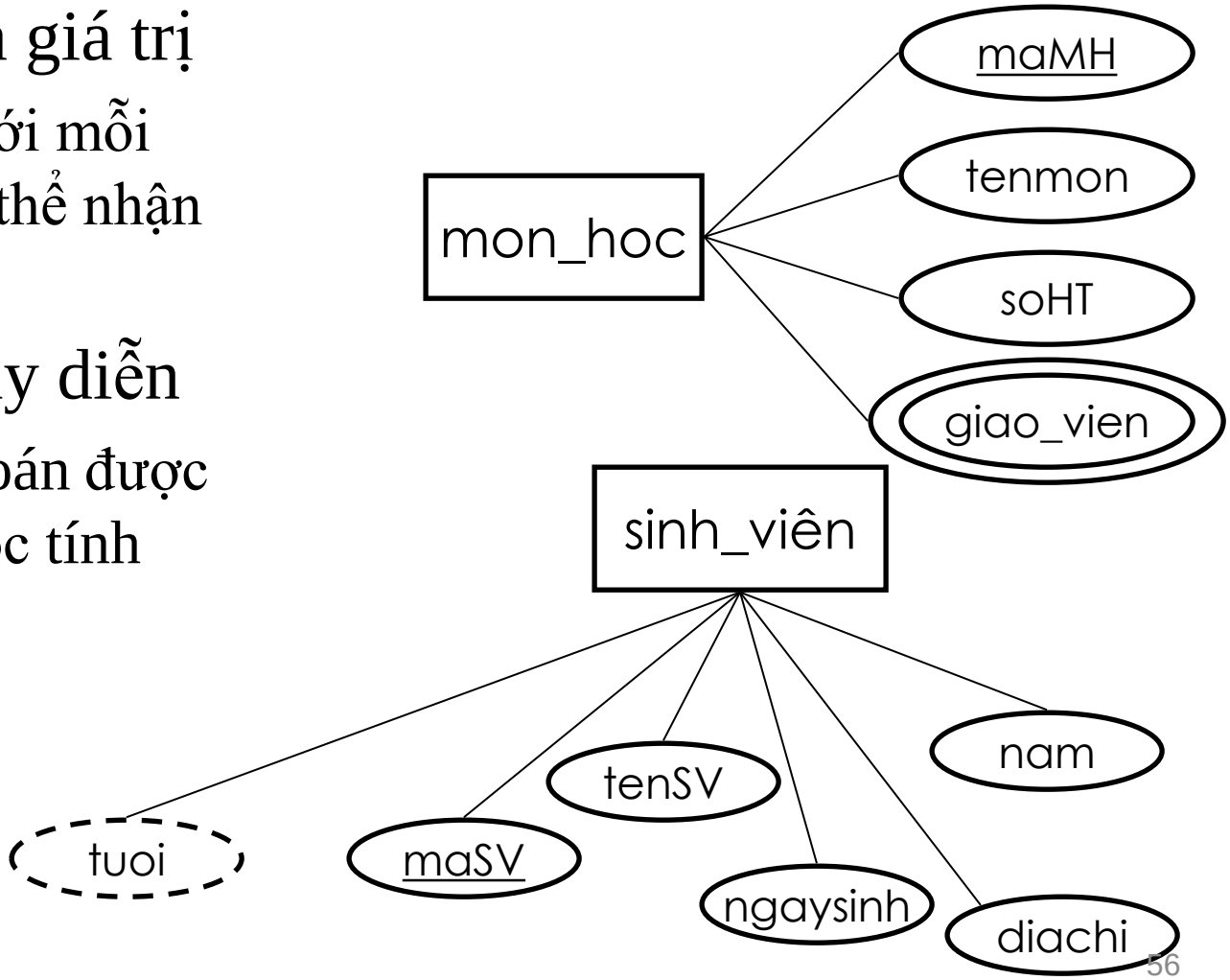
Kiểu thuộc tính

- Thuộc tính đơn giản (thuộc tính nguyên tố)
  - có kiểu dữ liệu nguyên tố
- Thuộc tính phức
  - có kiểu phức, định nghĩa bởi các thuộc tính khác



Kiểu thuộc tính

- Thuộc tính đa giá trị
  - tương ứng với mỗi thực thể, có thể nhận nhiều giá trị
- Thuộc tính suy diễn
  - có thể tính toán được từ (các) thuộc tính khác

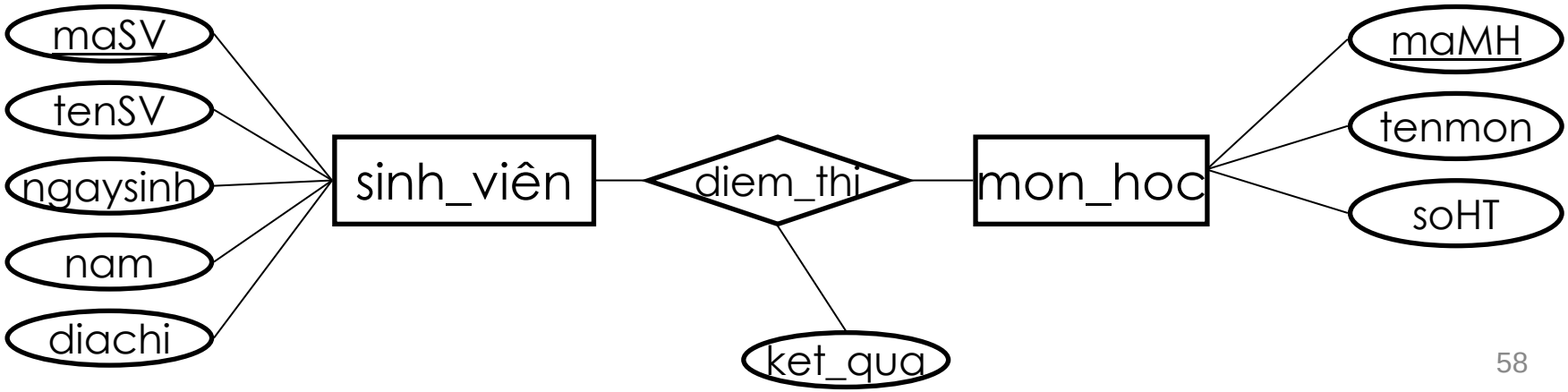


Khóa

- Một hay một tập thuộc tính mà giá trị của chúng có thể xác định duy nhất một thực thể trong tập thực thể
  - Tập thực thể SinhViên có thể dùng MãSV làm khoá
- Khoá gồm nhiều thuộc tính thì gọi là khoá phức
- Một tập thực thể có thể có nhiều khoá nhưng chỉ một trong số các khoá được chọn làm khoá chính
- Trong sơ đồ ER, thuộc tính nào được chọn làm khoá chính sẽ được gạch chân

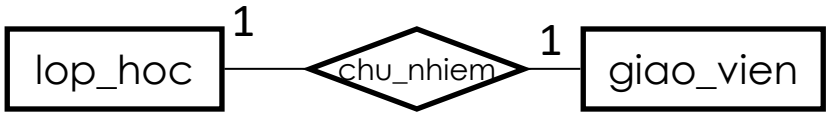
Liên kết - Tập liên kết

- Một liên kết là một mối liên hệ có nghĩa giữa nhiều thực thể
  - Cho một thực thể SinhViên1 và LớpA, liên kết ThànhViên chỉ ra rằng SinhViên1 là 1 thành viên của LớpA
- Tập liên kết là một tập hợp các liên kết cùng kiểu
  - Giữa tập thực thể SinhViên và Lớp có 1 tập liên kết ThànhViên, chỉ ra rằng mỗi sinh viên đều là thành viên của 1 lớp nào đó
- Một liên kết có thể có thuộc tính

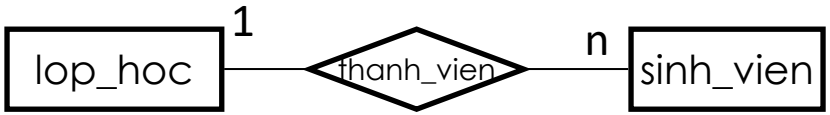


Ràng buộc của kết nối

- **1-1**: Liên kết 1 thực thể của một tập thực thể với nhiều nhất 1 thực thể của tập thực thể khác

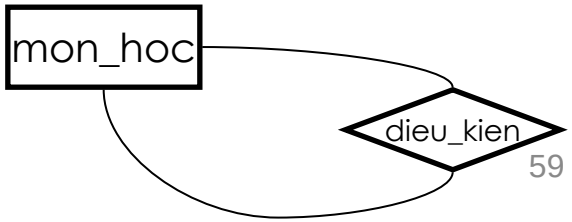


- **1-n**: Liên kết 1 thực thể của một tập thực thể với nhiều thực thể của tập thực thể khác



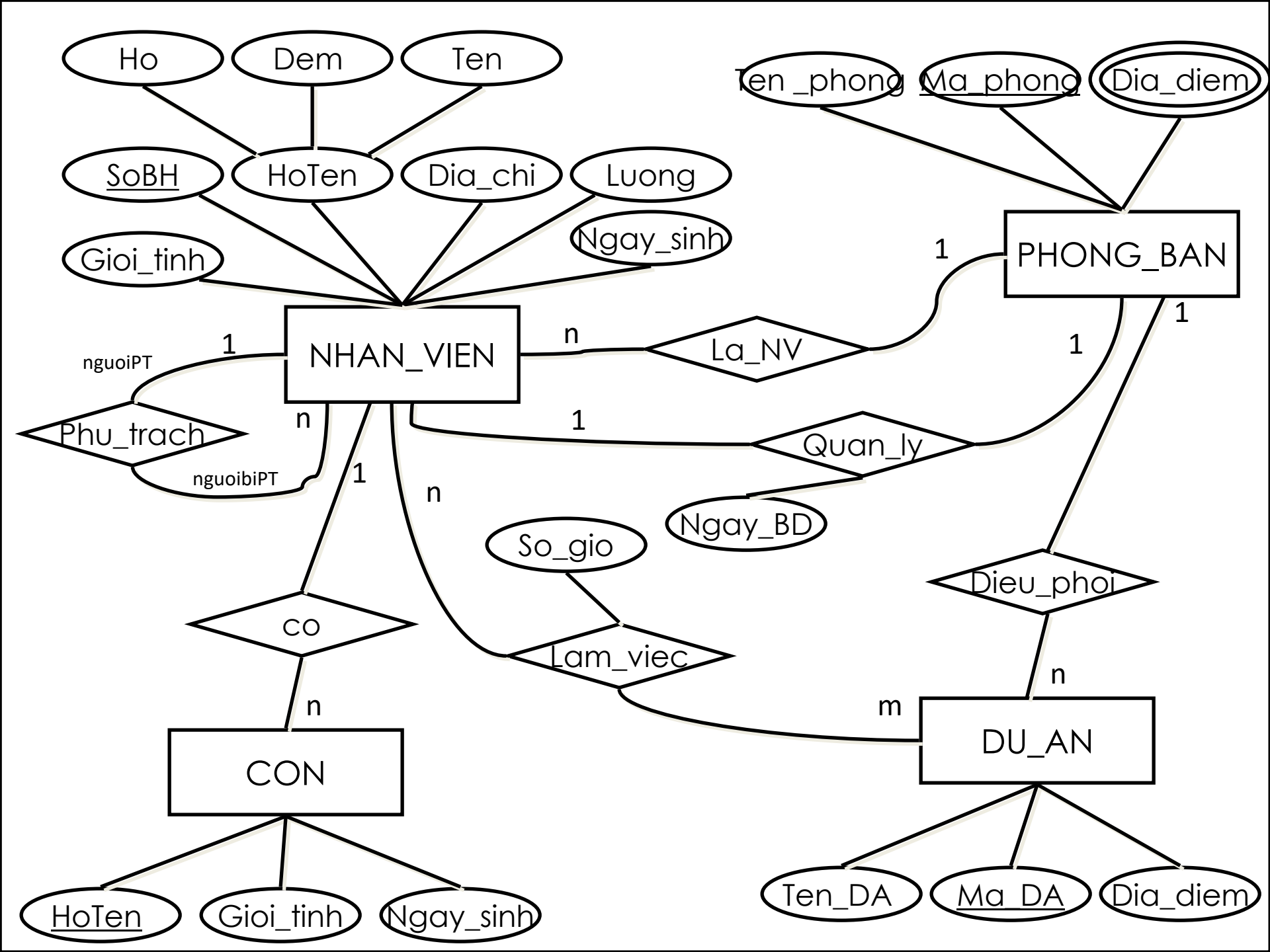
- **n-n**: Liên kết 1 thực thể của một tập thực thể với nhiều thực thể của tập thực thể khác và ngược lại

- **đệ quy**: Liên kết giữa các thực thể cùng kiểu



Cách thiết lập sơ đồ thực thể - liên kết

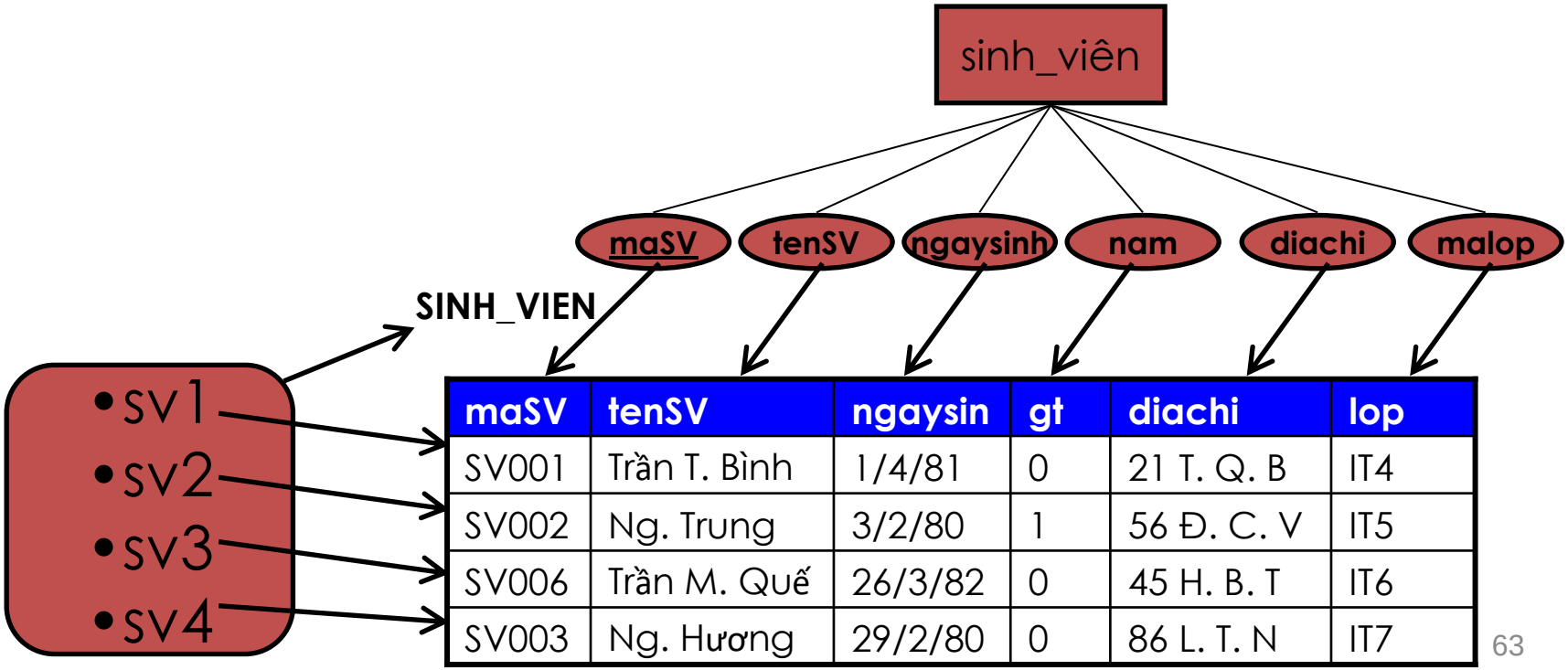
- Bước 1: Xác định các thực thể
- Bước 2: Xác định các liên kết giữa các thực thể
  - Bậc của liên kết
  - Ràng buộc (1-1, 1-n, n-n, đệ quy)


Ho
Dem
Ten
Ten _phong
Ma_phong
Dia_diem
SoBH
HoTen
Dia_chi
Luong
Ngay_sinh
Gioi_tinh
PHONG_BAN
1
1
1
nguoiPT
NHAN_VIEN
n
La_NV
1
n
Phu_trach
1
Quan_ly
nguoibiPT
1
n
Ngay_BD
So_gio
Dieu_phoi
co
Lam_viec
n
n
m
CON
DU_AN
HoTen
Gioi_tinh
Ngay_sinh
Ten_DA
Ma_DA
Dia_diem

Biến đổi sơ đồ thực thể liên kết sang sơ đồ quan hệ

- Biến đổi tập các thực thể
- Biến đổi các liên kết
- Các khoá của các sơ đồ quan hệ
- Các sơ đồ quan hệ với khoá chung

- **Bước 1**: (biến đổi các tập thực thể) 1 tập thực thể → 1 quan hệ
  - thuộc tính → thuộc tính (trường)
  - 1 thực thể → 1 bộ
  - khoá của tập thực thể → khoá của quanhệ



maSV	tenSV	ngaysin	gt	diachi	lop
SV001	Trần T. Bình	1/4/81	0	21 T. Q. B	IT4
SV002	Ng. Trung	3/2/80	1	56 Đ. C. V	IT5
SV006	Trần M. Quế	26/3/82	0	45 H. B. T	IT6
SV003	Ng. Hương	29/2/80	0	86 L. T. N	IT7

- **Bước 2:** (biến đổi các tập thực thể) 1 tập thực thể xác định từ tập thực thể khác (E) qua 1 liên kết → 1quan hệ chứa khoá cuả E:

  LOPTRUONG(maSV)

- **Bước 3:** (Biến đổi các liên kết) Liên kết 1-1

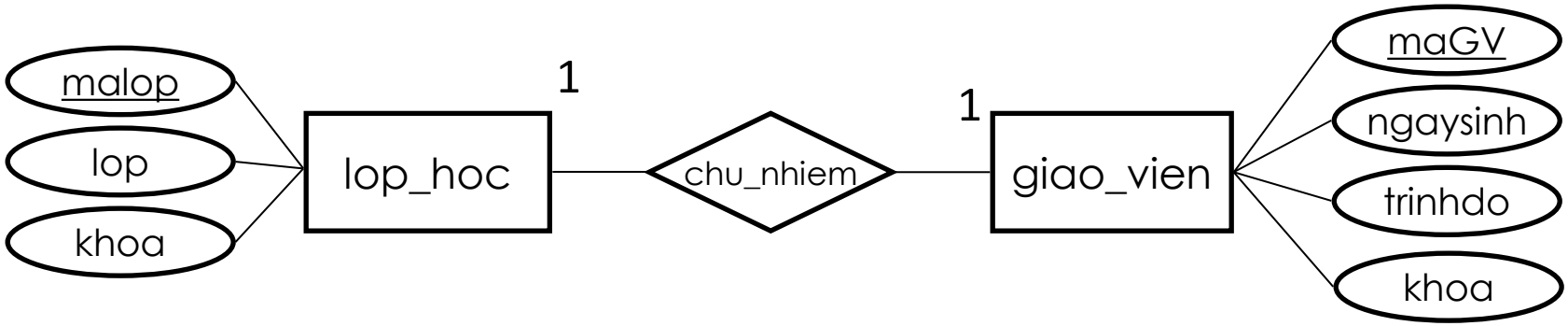
➢ Thêm 1 quan hệ mới xác định bởi các thuộc tính nằm trong khoá của các thực thể có liên quan

CHU_NHIEM_LOP(malop,maGV)

hoặc

➢ Dùng khoá ngoài

LOP_HOC(malop,lop,khoa,*maGV*)



- **Bước 4**: Liên kết 1-n

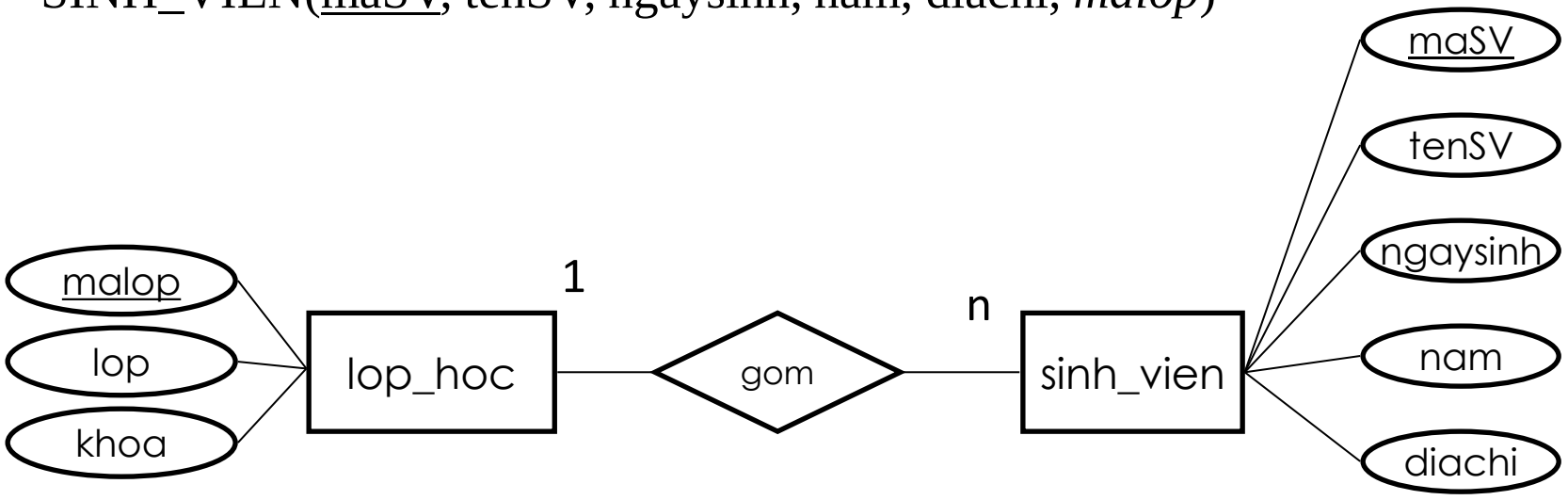
➢ Thêm 1 quan hệ mới xác định bởi các thuộc tính nằm trong khoá của các thực thể có liên quan

SINHVIEN_LOP(malop, maSV)

hoặc

➢ Dùng khoá ngoài: thêm khoá chính của quan hệ bên 1 vào quan hệ bên n làm khoá ngoài

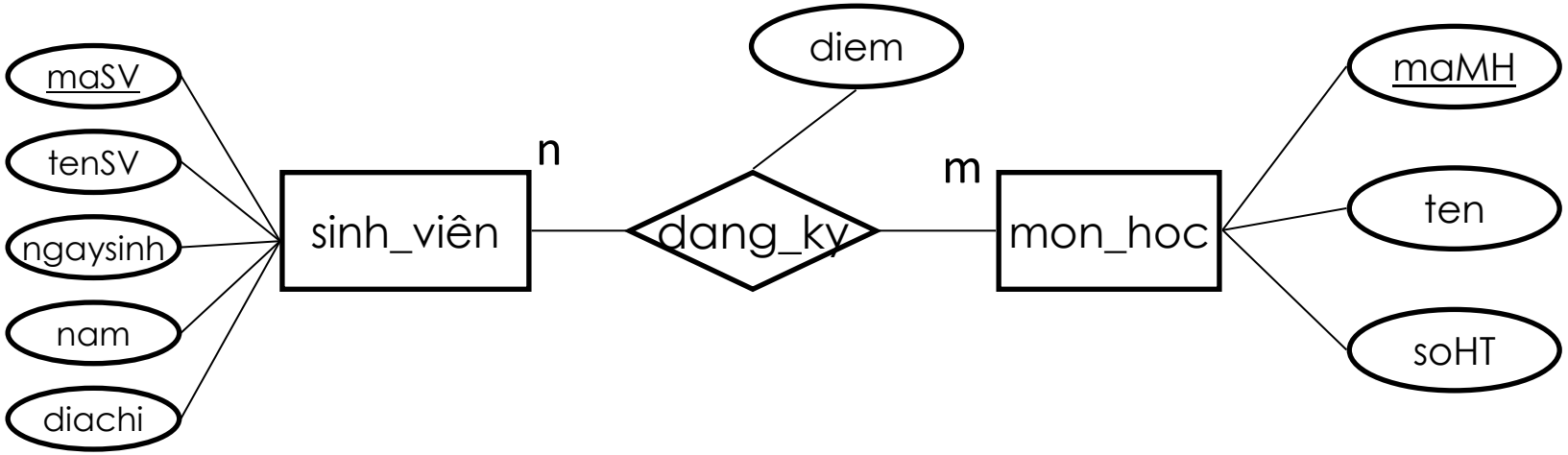
SINH_VIEN(maSV, tenSV, ngaysinh, nam, diachi, *malop*)



- **Bước 5:** Liên kết n-n

➢ Thêm 1 quan hệ mới xác định bởi các thuộc tính nằm trong khoá của các thực thể có liên quan và các thuộc tính của liên kết

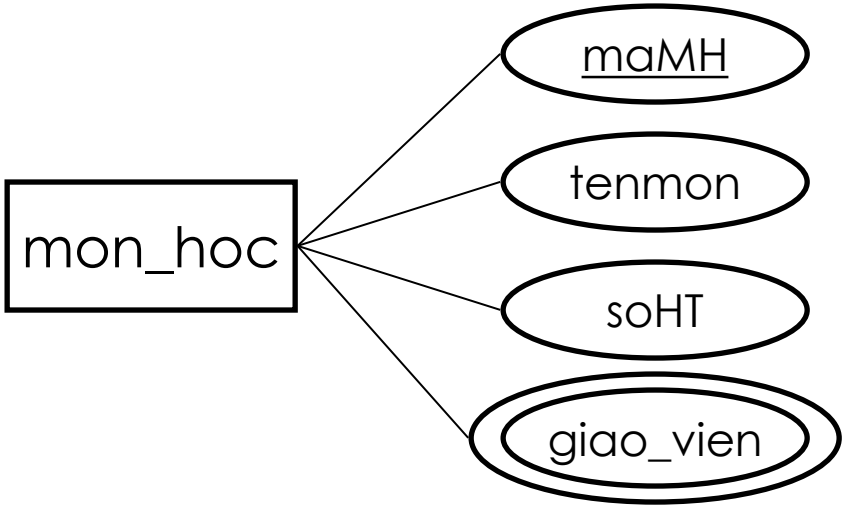
DANG_KY(<u>maSV,maMH,</u> diem)



- **Bước 6**: Với mỗi thuộc tính đa trị

➢ Thêm 1 quan hệ mới xác định bởi thuộc tính đa trị và khoá của tập thực thể tương ứng

MH_GV(<u>maMH,giao_vien</u>)



Mô hình dữ liệu hướng đối tượng (Object-oriented data model)

- Sự ra đời
  - Khoảng đầu những năm 90
- Biểu diễn: sơ đồ lớp
- Các khái niệm cơ bản
  - Đối tượng: một đối tượng trong thế giới thực, được xác định bởi một định danh duy nhất
  - Thuộc tính: biểu diễn một đặc tính của đối tượng,
  - Phương thức : thao tác được thực hiện trên đối tượng.
    - Tất cả các truy nhập vào thuộc tính của đối tượng đều phải được thực hiện thông qua các phương thức này.
  - Lớp: một cách thức để khai báo một tập các đối tượng có chung một tập thuộc tính và phương thức

Ví dụ

```
class sinh_vien {
        string maSV;
        string tenSV;
        date ngaysinh;
        boolean nam;
        string diachi;
        string lop;

        string ten();
        string ngay_sinh();
        string dia_chi();
        string lop();
        void gan_DC(string DC_moi);
        void gan_lop(string lop);
}
```

Nhận xét

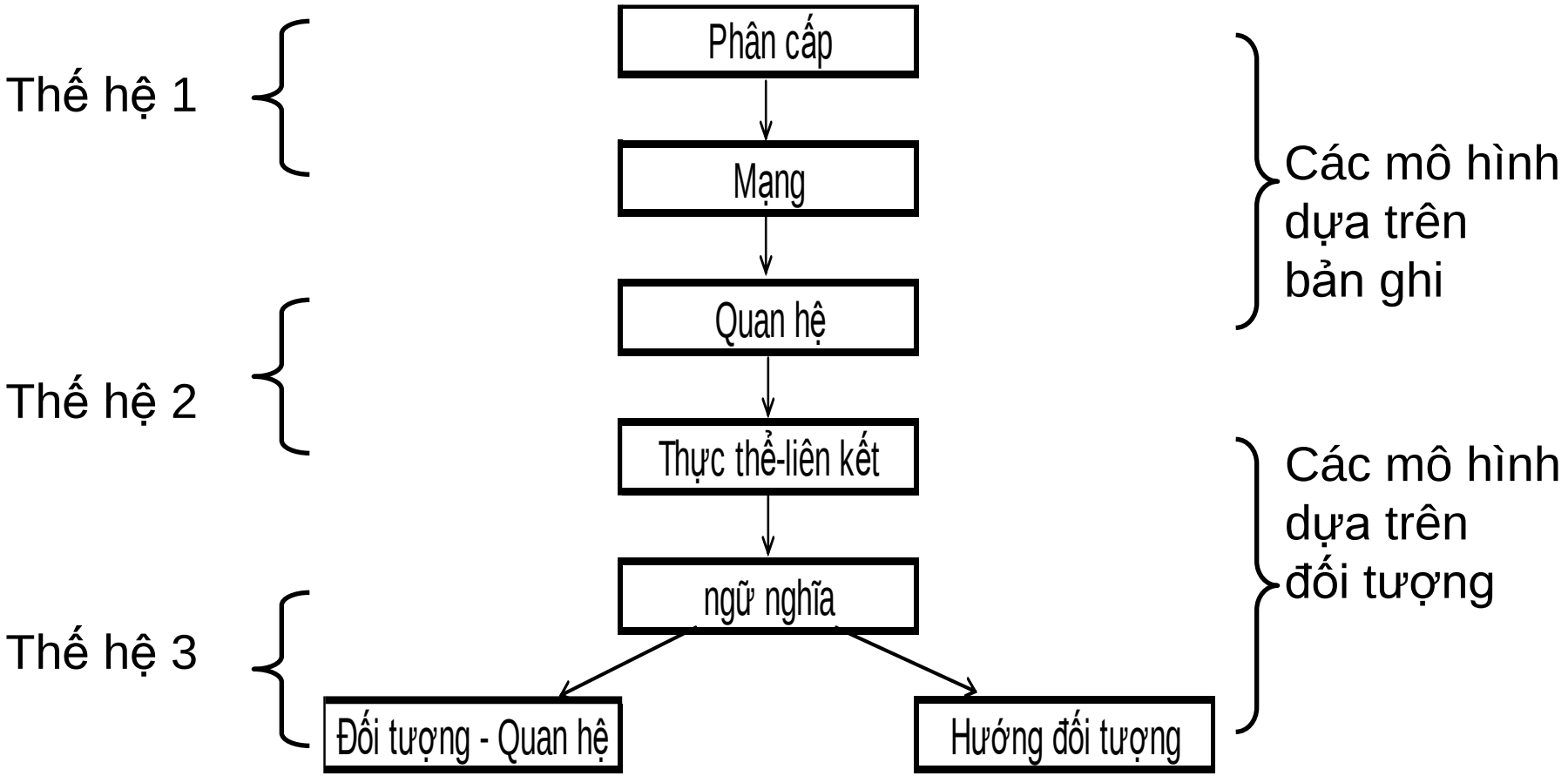
- Ưu điểm
  - Cho phép định nghĩa kiểu đối tượng phức tạp
  - Tính chất: bao đóng (*encapsulation*), kế thừa (*heritage*), đa hình (*polymorphism*)
- Nhược điểm
  - Cấu trúc lưu trữ phức tạp và có thể sử dụng nhiều con trỏ
  - Khả năng tối ưu hoá các xử lý bị hạn chế trong nhiều trường hợp

So sánh và đánh giá

Nhắc lại: Mô hình dữ liệu là một tập hợp các khái niệm dùng để mô tả cấu trúc của một CSDL

	Mô hình mạng	Mô hình phân cấp	Mô hình quan hệ	Mô hình TT-LK	Mô hình HĐT
biểu diễn ngữ nghĩa DL	hạn chế	hạn chế	***tương đối đa dạng***	đa dạng	đa dạng
lưu trữ DL	s/d nhiều con trỏ	dữ liệu lặp lại	**dễ dàng và hiệu quả**	khó lưu trữ	***cấu trúc phức tạp***
khả năng truy vấn	đơn giản	đơn giản	đa dạng		đa dạng
hiệu quả của truy vấn	ít khả năng tối ưu	ít khả năng tối ưu	tối ưu hoá tốt	không được xem xét (không hiệu quả)	***không hiệu quả khi sử dụng nhiều con trỏ***

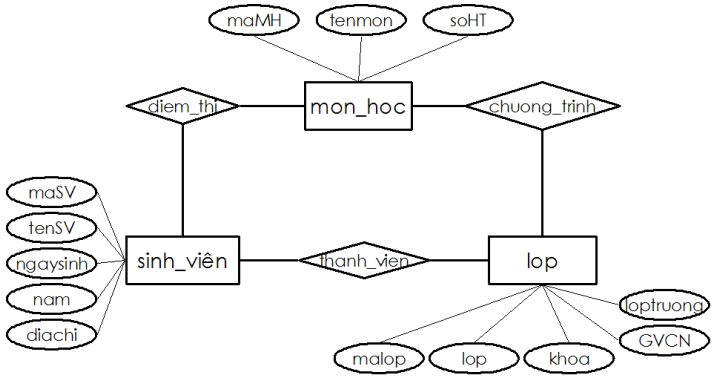
Phân loại các mô hình



Các bước xây dựng một hệ CSDL

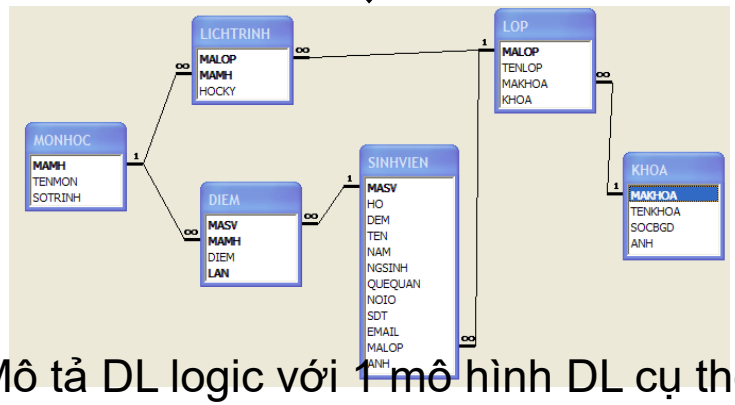
Mô tả ứng dụng

1: PHÂN TÍCH



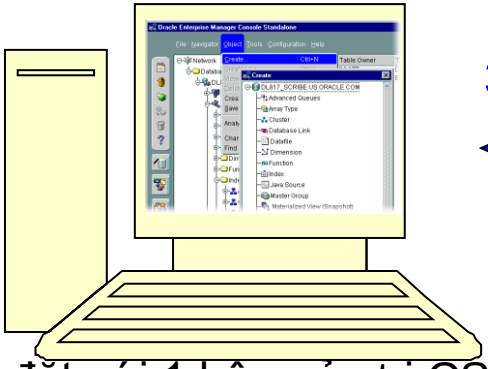
Mô hình hoá DL (vd: Sơ đồ thực thể-liên kết)

2: THIẾT KẾ



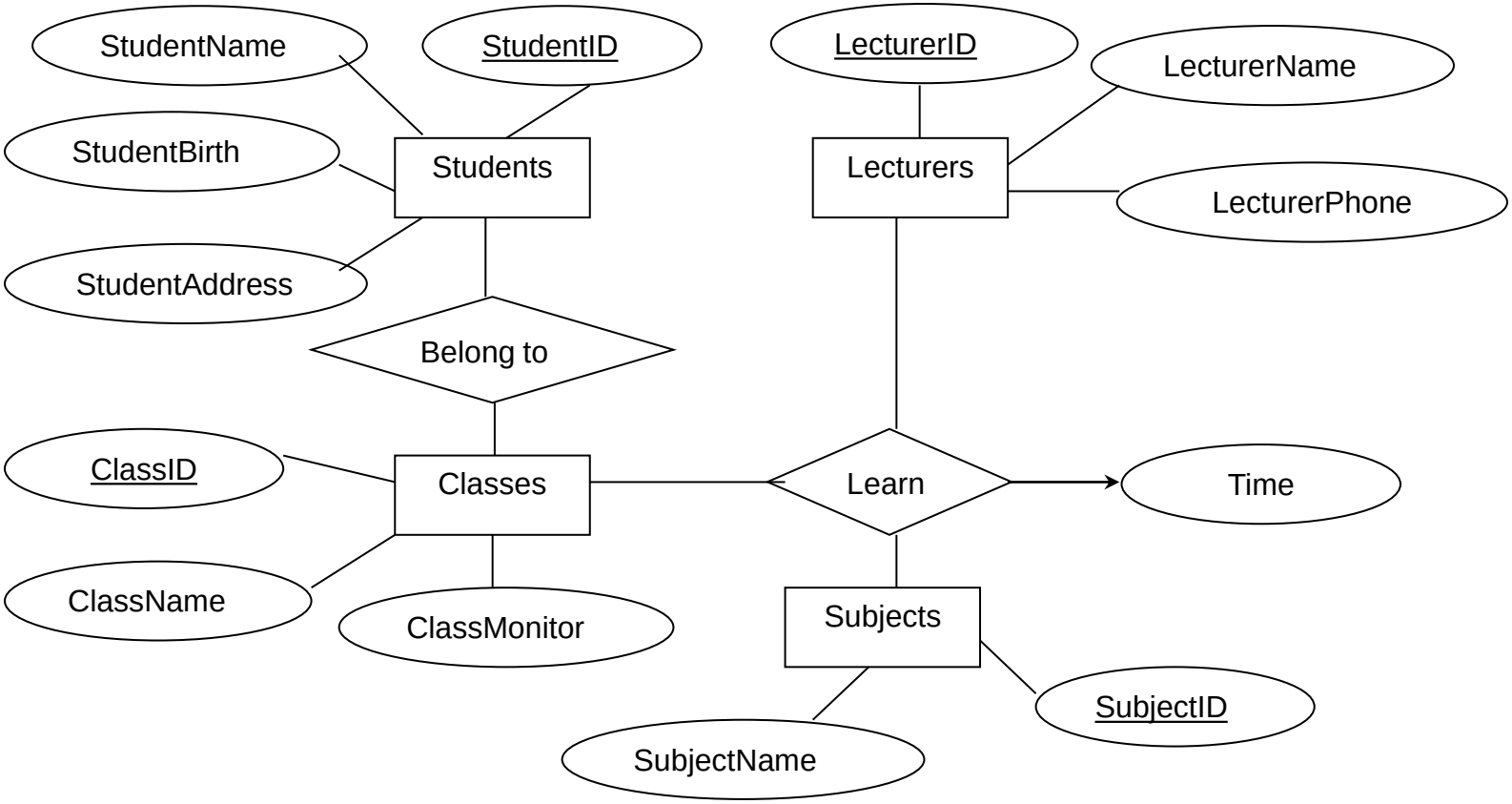
Mô tả DL logic với 1 mô hình DL cụ thể
(vd: Sơ đồ quan hệ)

3: CÀI ĐẶT



Cài đặt với 1 hệ quản trị CSDL
(vd: ORACLE)

- Cho sơ đồ thực thể liên kết bên dưới, hãy biến đổi sang mô hình quan hệ:



CƠ SỞ DỮ LIỆU

Chương 3

Chương 3 - Ngôn ngữ định nghĩa và thao tác dữ liệu đối với mô hình quan hệ

NỘI DUNG:

- Các cách tiếp cận đối với thiết kế ngôn ngữ của CSDL quan hệ
  - Giới thiệu một số ngôn ngữ và phân loại
  - ➢ So sánh và đánh giá
- Một số ngôn ngữ dữ liệu mức cao
  - QBE (***Q****uery* ***B****y* ***E****xample*)
  - SQL (***S****tructured* ***Q****uery* ***L****anguage*)
- Kết luận

Ví dụ

- Tìm các sinh viên đăng ký khoá học có mã số 113
  - Tìm các giá trị SID trong bảng Enrol có Course tương ứng là 113
  - Đưa các bộ của bảng Student có SID trong các giá trị tìm thấy ở trên

Student

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

Enrol

SID	Course
3936	101
1108	113
8507	101

Course

No	Name	Dept
113	BCS	CSCE
101	MCS	CSCE

Phân loại các ngôn ngữ truy vấn

- Ngôn ngữ đại số
  - 1 câu hỏi = 1 tập các phép toán trên các quan hệ
  - Được biểu diễn bởi một biểu thức đại số (quan hệ)
- Ngôn ngữ tính toán vị từ
  - 1 câu hỏi = 1 mô tả của các bộ mong muốn
  - Được đặc tả bởi một vị từ mà các bộ phải thoả mãn
  - Phân biệt 2 lớp:
    - ngôn ngữ tính toán vị từ biến bộ
    - ngôn ngữ tính toán vị từ biến miền

Ngôn ngữ đại số quan hệ

- Gồm các phép toán tương ứng với các thao tác trên các quan hệ
- Mỗi phép toán
  - Đầu vào: một hay nhiều quan hệ
  - Đầu ra: một quan hệ
- Biểu thức đại số quan hệ = chuỗi các phép toán
- Kết quả thực hiện một biểu thức đại số là một quan hệ
- Được cài đặt trong phần lớn các hệ CSDL hiện nay

Các phép toán đại số quan hệ

- Phép toán quan hệ
  - Phép chiếu (*projection*)
  - Phép chọn (*selection*)
  - Phép kết nối (*join*)
  - Phép chia (*division*)
- Phép toán tập hợp
  - Phép hợp (*union*)
  - Phép giao (*intersection*)
  - Phép trừ (*difference*)
  - Phép tích đề-các (*cartesian product*)

Phép toán tập hợp

- Định nghĩa: Quan hệ khả hợp

  Hai quan hệ r và s được gọi là khả hợp nếu chúng được xác định trên cùng một miền giá trị

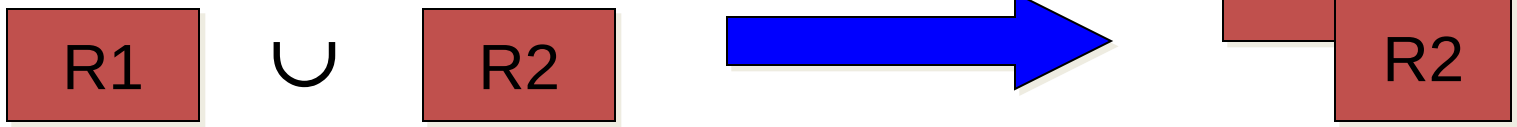
  r xác định trên $D_1 x\ D_2\ x \ldots x\ D_n$

  s xác định trên $D'_1 x\ D'_2\ x \ldots x\ D'_m$

  → $D_i = D'_i$ và $n=m$

Phép hợp

- Đ/n: gồm các bộ thuộc ít nhất một trong hai quan hệ đầu vào
- Hai quan hệ đầu vào phải là khả hợp
- Cú pháp: $R = R_1 \cup R_2$



Subject1

Name	Course
Systems	BCS
Database	BCS
Database	MCS
Algebra	MCS

∪

Subject2

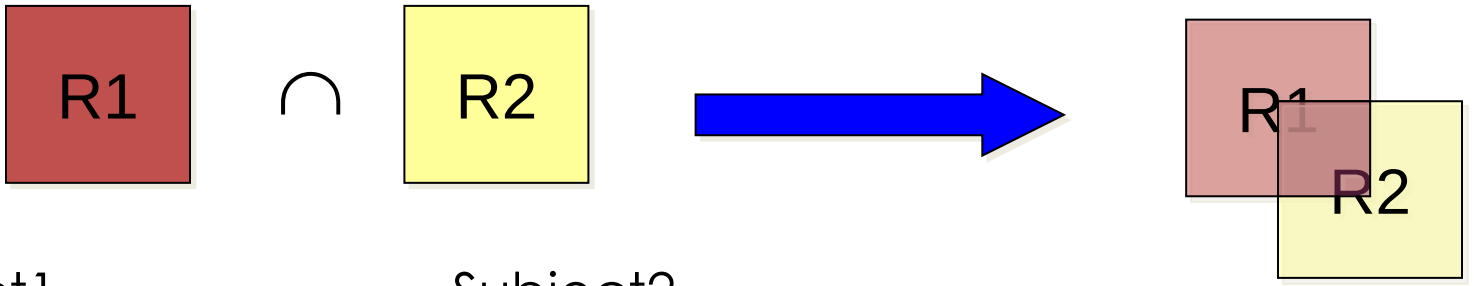
Name	Course
DataMining	MCS
Writing	BCS

Kết quả

Name	Course
Systems	BCS
Database	BCS
Database	MCS
Algebra	MCS
DataMining	MCS
Writing	BCS

Phép giao

- Đ/n: gồm các bộ thuộc cả hai quan hệ đầu vào
- Hai quan hệ đầu vào phải là khả hợp
- Cú pháp: $R_1 \cap R_2$



Subject1

Name	Course
Systems	BCS
Database	BCS
Database	MCS
Algebra	MCS

∩

Subject2

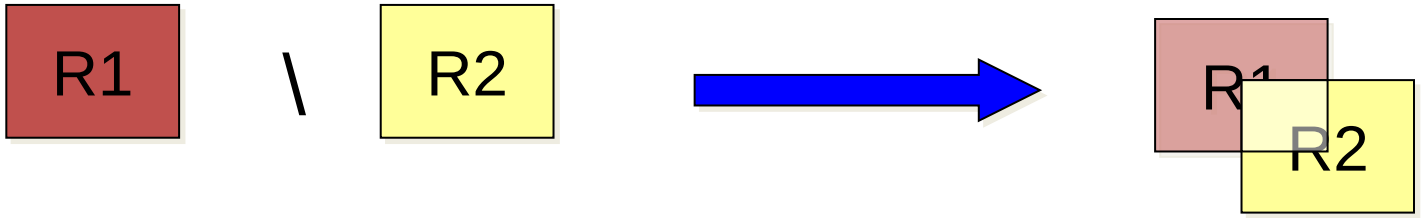
Name	Course
DataMining	MCS
Database	MCS
Systems	BCS
Writing	BCS

Kết quả

Name	Course
Systems	BCS
Database	MCS

Phép trừ

- Đ/n: gồm các bộ thuộc quan hệ thứ nhất nhưng không thuộc quan hệ thứ hai
  - Hai quan hệ phải là khả hợp
- Cú pháp: $R_1 \setminus R_2$ hoặc $R_1 - R_2$



Subject1

Name	Course
Systems	BCS
Database	BCS
Database	MCS
Algebra	MCS

\

Subject2

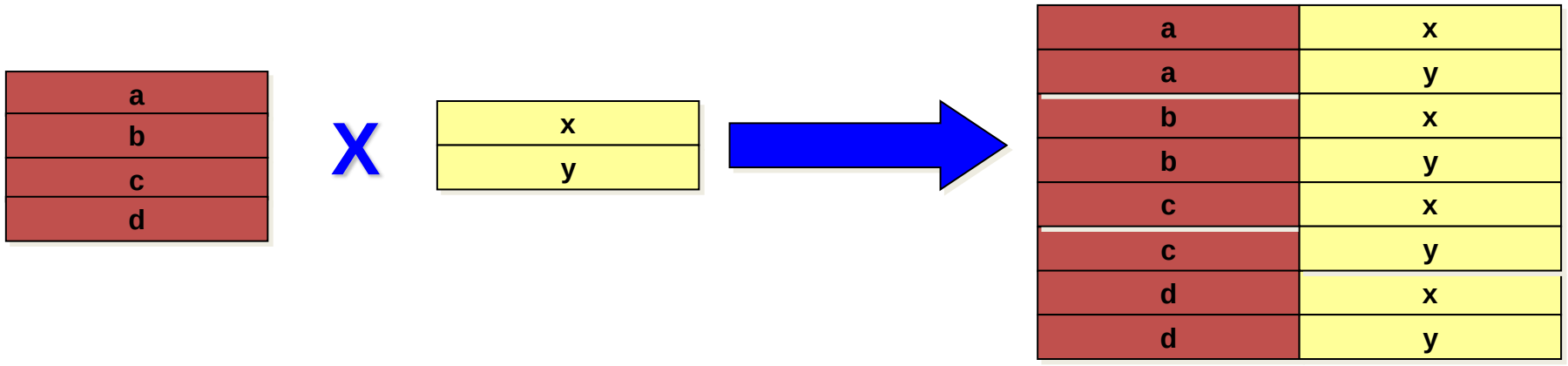
Name	Course
DataMining	MCS
Database	MCS
Systems	BCS
Writing	BCS

Kết quả

Name	Course
Database	BCS
Algebra	MCS

Phép tích Đề-các

- Đ/n: là kết nối giữa từng bộ của quan hệ thứ nhất với mỗi bộ của quan hệ thứ hai
- Cú pháp: $R = R_1 \; x \; R_2$



Student

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

X

Sport

SportID	Sport
05	Swimming
09	Dancing

Student_Sport

Id	Name	Suburb	SportID	Sport
1108	Robert	Kew	05	Swimming
1108	Robert	Kew	09	Dancing
3936	Glen	Bundoora	05	Swimming
3936	Glen	Bundoora	09	Dancing
8507	Norman	Bundoora	05	Swimming
8507	Norman	Bundoora	09	Dancing
8452	Mary	Balwyn	05	Swimming
8452	Mary	Balwyn	09	Dancing

Phép chiếu

- Đ/n: Lựa chọn một số thuộc tính từ một quan hệ.
- Cú pháp: $\prod_{A1,A2,\ldots}(R)$

❖Ví dụ: đưa ra danh sách tên của tất cả các sinh viên

Student

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

$\prod_{Name}(Student)$

Name
Robert
Glen
Norman
Mary

Phép chọn

- Đ/n: Lựa chọn các bộ trong một quan hệ thoả mãn điều kiện cho trước.
- Cú pháp: $\sigma_{<condition>}(R)$

- Ví dụ: đưa ra danh sách những sinh viên sống ở Bundoora

Student

$\sigma_{suburb='Bundoora'}(Student)$

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

Id	Name	Suburb
3936	Glen	Bundoora
8507	Norman	Bundoora

Phép chọn - Điều kiện ?

- Điều kiện chọn còn gọi là biểu thức chọn.
- Biểu thức chọn F: một tổ hợp logic của các toán hạng. Mỗi toán hạng là một phép so sánh đơn giản giữa hai biến là hai thuộc tính hoặc giữa một biến là một thuộc tính và một giá trị hằng.
  - Các phép so sánh trong F: $<, =, >, \leq, \geq, \neq$
  - Các phép toán logic trong F: $\wedge, \vee, \neg$

Ví dụ: chọn và chiếu

- Đưa ra tên của các sinh viên sống ở Bundoora

$$\prod_{Name}(\sigma_{suburb='Bundoora'}Student)$$

Student

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

Name
Glen
Norman

Phép kết nối (join) hai quan hệ r và s

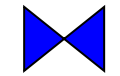
- Khái niệm ghép bộ: $u = (a_1,..,a_n)$; $v=(b_1,..,b_m)$
  $(u,v) = (a_1,..,a_n,b_1,..,b_m)$
- Phép kết nối hai quan hệ thực chất là phép ghép các cặp bộ của hai quan hệ thỏa mãn một điều kiện nào đó trên chúng.
- Biểu thức kết nối là phép hội của các toán hạng, mỗi toán hạng là một phép so sánh đơn giản giữa một thuộc tính của quan hệ r và một thuộc tính của quan hệ s.
- Cú pháp: $R1 \bowtie_{<<\text{điều kiện}>>} R2$

- Đưa ra danh sách các sinh viên và mã khoá học mà sinh viên đó tham gia:

$$\text{Student} \bowtie_{\text{Id=SID}} \text{Enrol}$$

Student

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn



Id=SID

Enrol

SID	Course
3936	101
1108	113
8507	101

Kết quả

SID	Id	Name	Suburb	Course
1108	1108	Robert	Kew	113
3936	3936	Glen	Bundoora	101
8507	8507	Norman	Bundoora	101

Phép kết nối bằng - kết nối tự nhiên

- Định nghĩa: Nếu phép so sánh trong điều kiện kết nối là phép so sánh bằng thì kết nối gọi là **kết nối bằng**
- Định nghĩa: Phép kết nối bằng trên các thuộc tính cùng tên của hai quan hệ và sau khi kết nối một thuộc tính trong một cặp thuộc tính trùng tên đó sẽ bị loại khỏi quan hệ kết quả thì phép kết nối gọi là **kết nối tự nhiên**
- Cú pháp phép kết nối tự nhiên: $R_1 * R_2$

Takes

SID	SNO
1108	21
1108	23
8507	23
8507	29

*

Enrol

SID	Course
3936	101
1108	113
8507	101

SID	SNO	Course
1108	21	113
1108	23	113
8507	23	101
8507	29	101

- Đưa ra tên của các sinh viên sống ở Bundoora và mã khoá học mà sinh viên đó đăng ký:

$$\prod_{Name,Course}(\sigma_{Suburb='Bundoora'}(Student \triangleright\!\triangleleft_{Id=SID} Enrol))$$

Student

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

Enrol

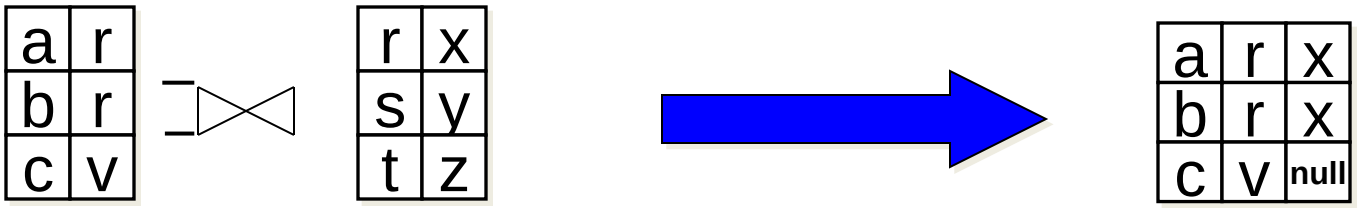
SID	Course
3936	101
1108	113
8507	101

Kết quả

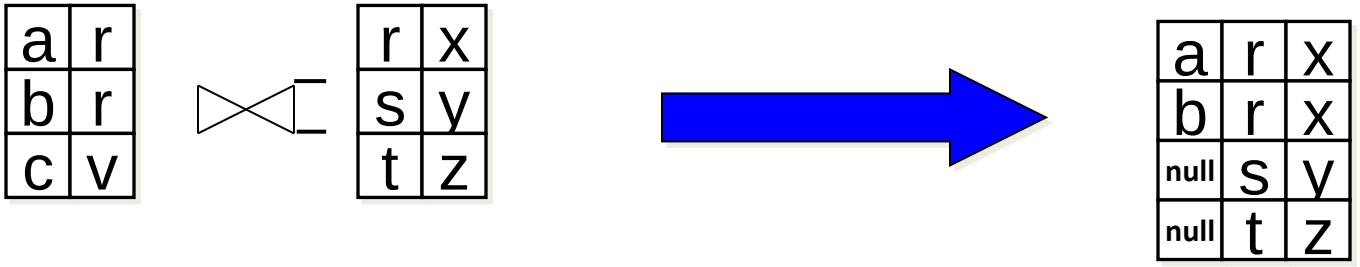
Name	Course
Glen	101
Norman	101

Phép kết nối ngoài

- Phép kết nối ngoài trái



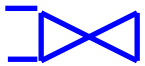
- Phép kết nối ngoài phải



- Đưa ra danh sách các sinh viên và mã khoá học mà sinh viên đó đăng ký nếu có

Student

ID	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn



Enrol

SID	Course
3936	101
1108	113
8507	101

Kết quả

ID	Name	Suburb	Course
1108	Robert	Kew	113
3936	Glen	Bundoora	101
8507	Norman	Bundoora	101
8452	Mary	Balwyn	**null**

Phép chia

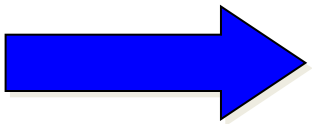
- Định nghĩa: Phép chia giữa một quan hệ r bậc n và quan hệ s bậc m (m<n) với sơ đồ quan hệ của s là tập con của sơ đồ quan hệ của r là một tập các (n-m) – bộ t sao cho khi ghép mọi bộ thuộc s với t thì ta đều có một bộ thuộc r
- Cú pháp: $R = R_1 : R_2$

$$\mathbf{r \div s = \{ t \mid \forall\ v \in s \Rightarrow (t,v) \in r\}}$$

a	x
a	y
a	z
b	x
c	y

:

x
z



a

- Ví dụ: Đưa ra môn học được dạy ở tất cả các khoá học

Subject

Name	Course
Systems	BCS
Database	BCS
Database	MCS
Algebra	MCS

:

Course

Course
BCS
MCS

Kết quả

Name
Database

Bài tập

- Cho CSDL gồm 3 quan hệ sau: S (Các hãng cung ứng), P (các mặt hàng), SP (các sự cung ứng).

S (S# SNAME STATUS CITY)

S#	SNAME	STATUS	CITY
S1	Smith	20	London
S2	Jones	10	Paris
S3	Black	30	Paris

SP (S# P# QTY)

S#	P#	QTY
S1	P1	300
S1	P2	200
S1	P3	400
S2	P1	300
S2	P2	400
S3	P2	200

P (P# PNAME COLOR WEIGHT CITY)

P#	PNAME	COLOR	WEIGHT	CITY
P1	Nut	red	12	London
P2	Bolt	green	17	Paris
P3	Screw	blue	17	Rom
P4	Screw	red	14	London

- Biểu diễn các truy vấn sau bằng đại số quan hệ:
  - Đưa ra danh sách các mặt hàng màu đỏ
  - Cho biết S# của các hãng cung ứng mặt hàng 'P1' hoặc 'P2'
  - Liệt kê S# của các hãng cung ứng cả hai mặt hàng 'P1' và 'P2'
  - Đưa ra S# của các hãng cung ứng ít nhất một mặt hàng màu đỏ
  - Đưa ra S# của các hãng cung ứng tất cả các mặt hàng.

Bài tập

- Cho các quan hệ sau:

Supplier

sid	sname	size	city
S1	Dustin	100	London
S2	Rusty	70	Paris
S3	Lubber	120	London

Product

pid	pname	colour
P1	Screw	red
P2	Screw	green
P3	Nut	red
P4	Bolt	blue

SupplyProduct

sid	pid	quantity
S1	P1	500
S1	P2	400
S1	P3	100
S2	P2	200
S3	P4	100
S2	P3	155

- Biểu diễn các truy vấn sau bằng biểu thức đại số quan hệ:
  1) Đưa ra {sid,sname,size,city} của các Supplier có trụ sở tại London
  2) Đưa ra {pname} của tất cả các mặt hàng
  3) Đưa ra {sid} của các Supplier cung cấp mặt hàng P1 hoặc P2
  4) Đưa ra {sname} của các Supplier cung cấp mặt hàng P3
  5) Đưa ra {sname} của các hãng cung ứng ít nhất một mặt hàng màu đỏ
  6) Đưa ra {sid} của các hãng cung ứng tất cả các mặt hàng màu đỏ
  7) Đưa ra {sname} của các hãng có cung ứng mặt hàng màu đỏ hoặc màu xanh
  8) Đưa ra {sname} của các hãng cung ứng ít nhất một mặt hàng màu đỏ và ít nhất một mặt hàng màu xanh
  9) Đưa ra {sid} của các hãng không cung ứng mặt hàng nào

Ngôn ngữ QBE

QBE (*Query-By-Example*)

- Là một ngôn ngữ truy vấn dữ liệu
- Các câu truy vấn được thiết lập bởi một giao diện đồ hoạ
- Phù hợp với các câu truy vấn đơn giản, tham chiếu đến ít bảng
- Một số sản phẩm: IBM™ (IBM Query Management Facility), Paradox, MS. Access, ...

Truy vấn trên một quan hệ

- P.~ Print

Student	ID	Name	Suburb
		P._x	Bundoora

- Biểu thức đại số quan hệ tương đương

$$\sigma_{suburb='Bundoora'}(Student)$$

- Lựa chọn tất cả các cột

Student	ID	Name	Suburb
P.			Bundoora

- Sắp xếp

Student	ID	Name	Suburb
		P.AO(1)	P.AO(2)

- AO: sắp xếp tăng dần
- DO: sắp xếp giảm dần

Truy vấn trên nhiều quan hệ

- Đưa ra tên của các sinh viên có đăng ký ít nhất một khoá học

Student	ID	Name	Suburb
	_id	P._name	

Enrol	SID	Course
	_id	

- Đưa ra tên các sinh viên không đăng ký một khoá học nào

Student	ID	Name	Suburb
	_id	P._name	

Enrol	SID	Course
¬	_id	

Các tính toán tập hợp

- Các phép toán: AVG, COUNT, MAX, MIN, SUM
- Ví dụ: đưa ra tên các thành phố và số lượng sinh viên đến từ thành phố đó

Student	ID	Name	Suburb	
	_id		G.P.	P.COUNT._id

- G. ~ Grouping

Hộp điều kiện

- Được sử dụng để biểu diễn
  - Điều kiện trên nhiều hơn một thuộc tính
  - Điều kiện trên các trường tính toán tập hợp
- Ví dụ: đưa ra danh sách các thành phố có nhiều hơn 5 sinh viên

Student	ID	Name	Suburb	Condition
	_id		G.P.	COUNT._id > 5

Các thao tác thay đổi dữ liệu

- Xóa

Student	ID	Name	Suburb
D.	1108		

- Thêm

Student	ID	Name	Suburb
I.	1179	David	Evry

- Sửa

Student	ID	Name	Suburb
	1179		U.Paris

Tính đầy đủ của QBE

- Có thể biểu diễn cả 5 phép toán đại số cơ sở (σ,Π,$\cup$,\,x)

Định nghĩa dữ liệu trong QBE

- Sử dụng cùng qui cách và giao diện đồ họa như đối với truy vấn.

I.Student	I.	ID	Name	Suburb
KEY	I.	Y	N	N
TYPE	I.	CHAR(5)	CHAR(30)	CHAR(30)
DOMAIN	I.	Sid	SName	Surb
INVERSION	I.	Y	N	N

- Các khung nhìn

I.View V	I.	ID	Name	Course
	I.	_id	_name	_course

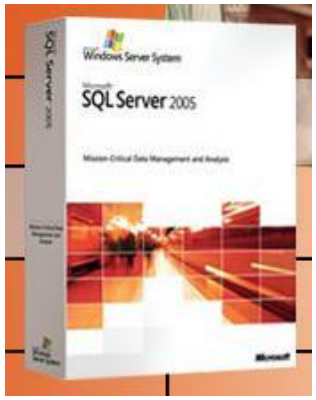
Student	ID	Name	Suburb
	_id	_name	

Enrol	SID	Course
	_id	_course

Ngôn ngữ SQL

SQL (Structured Query Language)

- 1975: SEQUEL
  - System-R
- 1976: SEQUEL2
- 1978/79: SQL
  - System-R
- 1986: chuẩn SQL-86
- 1989: chuẩn SQL-89
- 1992: chuẩn SQL-92
- 1996: chuẩn SQL-96



Các thành phần của SQL

- Ngôn ngữ định nghĩa dữ liệu (***D**ata **D**efinition **L**anguage*)
  - Cấu trúc các bảng CSDL
  - Các mối liên hệ của dữ liệu
  - Quy tắc, ràng buộc áp đặt lên dữ liệu
- Ngôn ngữ thao tác dữ liệu (***D**ata **M**anipulation **L**anguage*)
  - Thêm, xoá, sửa dữ liệu trong CSDL
  - Truy vấn dữ liệu
- Ngôn ngữ điều khiển dữ liệu (***D**ata **C**ontrol **L**anguage*)
  - Khai báo bảo mật thông tin
  - Quyền hạn của người dùng trong khai thác CSDL

Ngôn ngữ định nghĩa dữ liệu

- Các thông tin được định nghĩa bao gồm
  - Sơ đồ quan hệ
  - Kiểu dữ liệu hay miền giá trị của mỗi thuộc tính
  - Các ràng buộc toàn vẹn
  - Các chỉ số đối với mỗi bảng
  - Thông tin an toàn và ủy quyền đối với mỗi bảng
  - Cấu trúc lưu trữ vật lý của mỗi bảng trên đĩa

➢ Được biểu diễn bởi các lệnh định nghĩa dữ liệu

Quy ước đặt tên và kiểu dữ liệu

- Quy ước đặt tên
  - 32 ký tự: chữ cái, số, dấu _
- Kiểu dữ liệu (SQL-92)
  - char(n)
  - varchar(n)
  - int
  - smallint
  - numeric(p,d)
  - real, double
  - float(n)
  - date
  - time

Cú pháp

- Tạo bảng

CREATE TABLE tên-bảng(
    cột-1 kiểu-dữ-liệu-1 [**NOT NULL**], ...,
    cột-2 kiểu-dữ-liệu-2 [**NOT NULL**], ...,
        .....
    [**CONSTRAINT** tên-ràng-buộc kiểu-ràng-buộc]
    ...
);

- Xoá bảng

DROP TABLE tên-bảng

```
CREATE TABLE Supplier(
   sid char(4) NOT NULL,
   sname varchar(30) NOT NULL,
   size smallint,
   city varchar(20),
   CONSTRAINT KhoachinhS primary key(sid)
);

CREATE TABLE Product(
   pid char(4) NOT NULL,
   pname varchar(30) NOT NULL,
   colour char(8),
   weight int,
    city varchar(20),
   CONSTRAINT KhoachinhP primary key(pid)
);
```

```
CREATE TABLE SupplyProduct(
        sid char(4) NOT NULL,
        pid char(4) NOT NULL,
        quantity smallint,
        primary key(sid,pid),
        foreign key(sid) references Supplier(sid),
        foreign key(pid) references Product(pid),
        check(quantity >0)
  );
```

Kiểu ràng buộc

- Ràng buộc toàn vẹn về giá trị miền

  CONSTRAINT <tên ràng buộc>

  CHECK <điều kiện>

- Ràng buộc toàn vẹn về khoá ngoại hay phụ thuộc tồn tại

  CONSTRAINT <tên ràng buộc>

  FOREIGN KEY (fk_i) **REFERENCES** tên-bảng(k_i);

Thêm/xoá/sửa cột của các bảng

- Thêm

  ALTER TABLE <tên bảng>

  ADD COLUMN <tên cột> <kiểu dữ liệu> [NOT NULL];

- Xoá

  ALTER TABLE <tên bảng>

  DROP COLUMN <tên cột>;

- Sửa

  ALTER TABLE <tên bảng>

  CHANGE COLUMN <tên cột> TO <kiểu dữ liệu mới>;

- ALTER TABLE SupplyProduct ADD COLUMN price real NOT NULL;
- ALTER TABLE SupplyProduct DROP COLUMN price;
- ALTER TABLE Supplier CHANGE COLUMN sname TO varchar(20);

Thêm/xóa các ràng buộc

- Thêm

  ALTER TABLE <tên bảng>

  ADD CONSTRAINT <tên ràng buộc>
  <kiểu ràng buộc>

- Xóa

  ALTER TABLE <tên bảng>

  DROP CONSTRAINT <tên ràng buộc>

Ngôn ngữ thao tác dữ liệu

- Cú pháp câu lệnh SQL:

SELECT [**DISTINCT**] <DS cột>|*|<Biểu thức>|<Hàm TV>
FROM <DS bảng>
[**WHERE** <Điều kiện tìm kiếm>]
[**GROUP BY** <DS cột> [**HAVING** <Điều kiện>]]
[**ORDER BY** <Danh sách cột> [**ASC**|**DESC**]]
[**UNION** |**INTERSECT**| **MINUS** <Câu truy vấn khác>]

Truy vấn không điều kiện trên một bảng

- Tìm thông tin từ các cột của bảng
  - **SELECT** <DS cột>
    FROM <Tên bảng>;
  - **SELECT** *
    FROM <Tên bảng>;
- Ví dụ

```
SELECT    Name
FROM      Student;
```

Student

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Robert	Bundoora
8452	Mary	Balwyn

$$\prod_{Name}(Student)$$

Name
Robert
Glen
Mary

- Đưa ra tên của các mặt hàng

  SELECT pname FROM Product;
- Đưa ra tên khác nhau của các mặt hàng

  SELECT DISTINCT pname

  FROM Product;
- Đưa ra toàn bộ thông tin về các hãng cung ứng

  SELECT * FROM Supplier;
- Đưa ra mã số hãng cung ứng, mã mặt hàng được cung ứng và 10 lần số lượng mặt hàng đã được cung ứng

  SELECT sid, pid, quantity*10

  FROM SupplyProduct;

Truy vấn có điều kiện trên một bảng

- Chọn các bản ghi (dòng)

```
SELECT    <DS cột>
FROM      <Tên bảng>
WHERE     <Điều kiện tìm kiếm>
```

- Ví dụ

```
SELECT    *
FROM      Student
WHERE     Suburb='Bundoora' ;
```

Student

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Robert	Bundoora
8452	Mary	Balwyn

$\sigma_{Suburb='Bundoora'}(Student)$

Id	Name	Suburb
3936	Glen	Bundoora
8507	Robert	Bundoora

- Đưa ra tên của các hãng cung ứng có trụ sở tại London

```
SELECT sname FROM Supplier
WHERE city = 'London';
```

- Đưa ra mã số và tên của các hãng cung ứng nằm ở London và có số nhân viên lớn hơn 75

```
SELECT sid, sname FROM Supplier
WHERE city = 'London' AND size > 75;
```

Biểu diễn điều kiện lựa chọn

- Các phép toán quan hệ: =, !=, <, >, <=, >=
- Các phép toán logic: NOT, AND, OR
- Phép toán phạm vi: BETWEEN, IN, LIKE
  - Kiểu dữ liệu số
    - attr **BETWEEN** val1 **AND** val2 (⇔ (attr>=val1) and (attr<=val2))
    - attr **IN** (val1, val2, ...) (⇔ (attr=val1) or (attr=val2) or ...)
  - Kiểu dữ liệu xâu
    - **LIKE**: sử dụng đối sánh mẫu xâu với các ký tự thay thế cho 1 ký tự bất kỳ (_, ?), thay thế cho 1 xâu ký tự bất kỳ (*, %)

    (PostGreSQL sử dụng dấu % và dấu _)

- Đưa ra thông tin của các hãng cung ứng có số nhân viên trong khoảng từ 100 đến 150

```
SELECT * FROM Supplier
WHERE size BETWEEN 100 AND 150;
```

- Đưa ra mã số của hãng cung ứng mặt hàng P1 hoặc P2
  - Cách 1:

```
SELECT sid FROM SupplyProduct
WHERE pid = 'P1' OR pid = 'P2';
```

  - Cách 2:

```
SELECT sid FROM SupplyProduct
WHERE pid IN ('P1', 'P2');
```

- Đưa ra thông tin của hãng sản xuất có trụ sở đặt tại thành phố bắt đầu bằng chữ New (New York, New Jersey, New Mexico, New Hampshire)

```
SELECT * FROM SUPPLIER
WHERE city LIKE 'New%';
```

Loại trừ các bản ghi trùng nhau

- Từ khoá **DISTINCT**

  SELECT DISTINCT <DS cột>

  FROM <DS bảng>

- Ví dụ: đưa ra danh sách tên các khoa (Dept) tương ứng với các khoá học (Course). Mỗi giá trị chỉ hiện thị một lần

  SELECT DISTINCT Dept

  FROM Course

Truy vấn có sử dụng phép toán đổi tên

- SQL cho phép đổi tên các bảng và các cột trong một câu truy vấn (sau mệnh đề SELECT và FROM) sử dụng cấu trúc:
- <tên cũ> AS <tên mới>
  - Đưa ra tên và số nhân viên của các hãng cung ứng ở Paris

```
SELECT sname AS HangOParis, size AS SoNhanVien
FROM Supplier
WHERE city = 'Paris';
SELECT    SID , Stud.Name as SName,
          Sub.Name as Subject
FROM      Student as Stud,Takes,
          Subject as Sub
WHERE     (Id=SID) and (SNO = No)
```

Truy vấn phức tạp trên nhiều bảng

- Điều kiện kết nối

```
SELECT  <DS cột>
FROM    <DS bảng>
WHERE   <Điều kiện tìm kiếm>
```

- Ví dụ: đưa ra danh sách mã sinh viên (Id), tên sinh viên (Name), thành phố (Suburb), mã khoá học (Course) mà các sinh viên đã đăng ký

```
SELECT  Id, Name, Suburb, Course
FROM    Student, Enrol
WHERE   Id=SID;
```

Kết nối tự nhiên

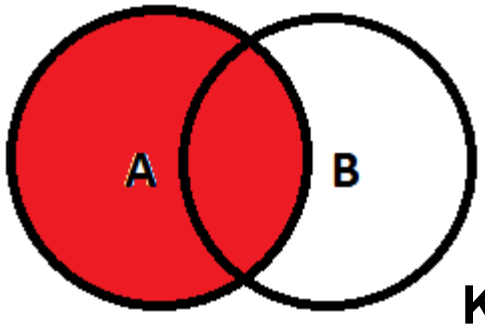
```
SELECT <DS cột>
FROM A, B, C
WHERE A.CộtX = B.CộtX AND B.CộtY = C.CộtY
```

```
SELECT <DS cột>
FROM A NATURAL JOIN B NATURAL JOIN C
```

Kết nối hai bảng

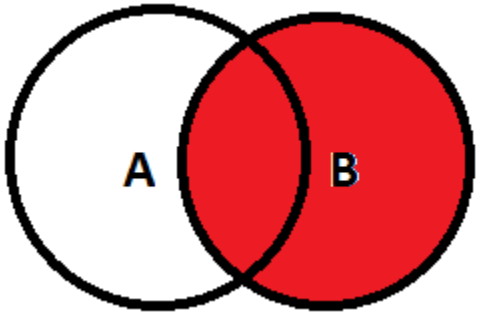
Kết nối ngoài trái

SELECT <DS cột>
FROM A LEFT JOIN B
ON A.Key = B.Key



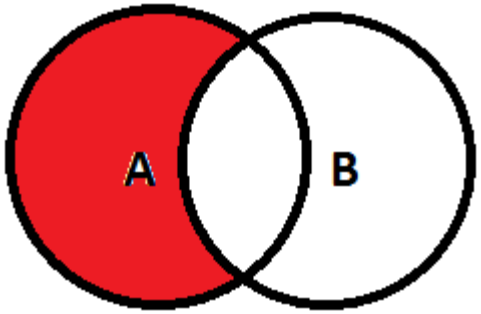
Kết nối ngoài phải

SELECT <DS cột>
FROM A RIGHT JOIN B
ON A.Key = B.Key



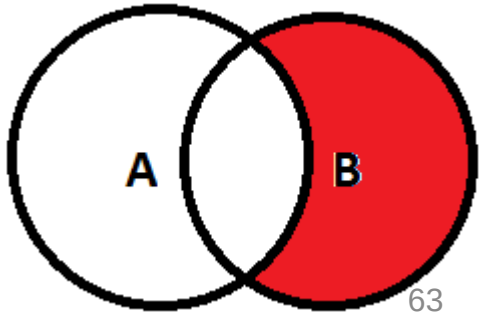
Kết nối ngoài trái

SELECT <DS cột>
FROM A LEFT JOIN B
ON A.Key = B.Key
WHERE B.Key IS NULL



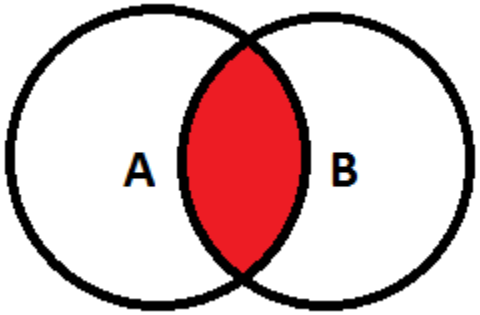
Kết nối ngoài phải

SELECT <DS cột>
FROM A RIGHT JOIN B
ON A.Key = B.Key
WHERE A.Key IS NULL



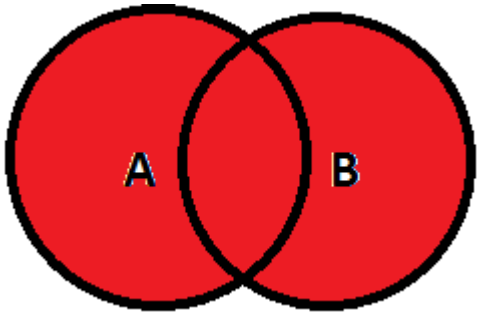
Kết nối trong

SELECT <DS cột>
FROM A INNER JOIN B
ON A.Key = B.Key



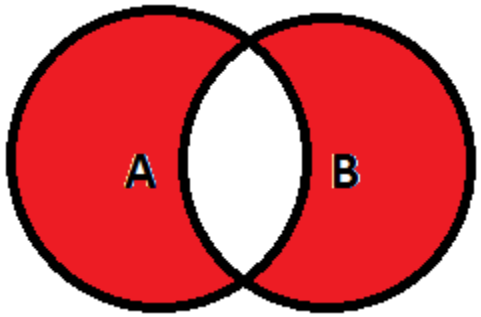
Kết nối OUTER JOIN

SELECT <DS cột>
FROM A FULL OUTER JOIN B
ON A.Key = B.Key



Kết nối OUTER JOIN

SELECT <DS cột>
FROM A FULL OUTER JOIN B
ON A.Key = B.Key
WHERE A.Key IS NULL OR
B.Key IS NULL



- Đưa ra tên của hãng có cung ứng mặt hàng P1

```
SELECT sname
FROM Supplier S, SupplyProduct SP
WHERE S.sid = SP.sid AND SP.pid = 'P1';
```

- Đưa ra tên và mã số của hãng cung ứng ít nhất một mặt hàng màu đỏ

```
SELECT sname, S.sid
FROM Supplier S, SupplyProduct SP, Product P
WHERE S.sid = SP.sid AND P.pid = SP.pid AND P.colour =
    'red';
```

Tìm kiếm có sắp xếp

- Sắp xếp các bản ghi kết quả theo một thứ tự cho trước

```
SELECT    <DS cột>
FROM      <DS bảng>
[WHERE    <Điều kiện tìm kiếm>]
ORDER BY  <DS cột> [ASC|DESC]
```

- Ví dụ: đưa ra danh sách tên các sinh viên theo thứ tự tăng dần

```
SELECT    Name
FROM      Student
ORDER BY Name ASC
```

Phân nhóm các bản ghi kết quả

- Phân nhóm các bản ghi kết quả theo giá trị của một hoặc nhiều thuộc tính

```
SELECT      <DS cột>
FROM        <DS bảng>
[WHERE      <Điều kiện tìm kiếm>]
[GROUP BY   <DS cột>]
```

- Cột được chỉ ra trong mệnh đề GroupBy được sử dụng làm cơ sở để chia nhóm. Cột này cũng bắt buộc phải được chỉ ra trong mệnh đề Select
- Ví dụ đưa ra tên các sinh viên nhóm theo thành phố của sinh viên đó

```
SELECT Suburb, Name
FROM Student
GROUP BY Suburb
```

```
SELECT Suburb, Count(Id)
FROM Student
GROUP BY Suburb
```

Điều kiện hiển thị các bản ghi kết quả

- Lựa chọn các bản ghi kết quả để hiển thị

  SELECT <DS cột>
  FROM <DS bảng>
  [**WHERE** <Điều kiện tìm kiếm>]
  GROUP BY <Ds cột> **HAVING** <Điều kiện>

- Ví dụ: đưa ra tên các thành phố có nhiều hơn 3 sinh viên

  SELECT Suburb, COUNT(ID)
  FROM Student
  GROUP BY Suburb
  HAVING COUNT(ID) > 3

Các phép toán tập hợp: UNION, MINUS, INTERSECT

- Ví dụ: đưa ra danh sách tên các môn học không có sinh viên nào tham dự

```
SELECT DISTINCT  Subject.Name
FROM  Subject
MINUS
SELECT DISTINCT  Subject.Name
FROM   Student, Takes, Subject
WHERE  Student.Id = Takes.SID and  Takes.SNO = Subject.No
```

- Tìm sid của hãng cung ứng đồng thời 2 mặt hàng P1 và P2

```
SELECT sid FROM SupplyProduct WHERE pid = 'P1'
INTERSECT
SELECT sid FROM SupplyProduct WHERE pid = 'P2'
```

- Tìm mã số của hãng không cung ứng mặt hàng nào

```
SELECT sid FROM Supplier
MINUS
SELECT sid FROM SupplyProduct
```

Các câu truy vấn lồng nhau

- Là trường hợp các câu truy vấn (con) được viết lồng nhau
- Thường được sử dụng để
  - Kiểm tra thành viên tập hợp (**IN, NOT IN**)
  - So sánh tập hợp (**>ALL, >=ALL, <ALL,<=ALL,=ALL, NOT IN, SOME,**)
    - vd: SELECT *
      FROM Supplier
      WHERE SIZE>=ALL(SELECT SIZE FROM Supplier);
  - Kiểm tra các bảng rỗng (**EXISTS** hoặc **NOT EXISTS**)
- Các truy vấn con lồng nhau thông qua mệnh đề **WHERE**

- Kiểm tra thành viên tập hợp với IN và NOT IN:
  - Đưa ra mã số của các hãng cung ứng đồng thời 2 mặt hàng P1 và P2:

```
SELECT DISTINCT sid FROM SupplyProduct
WHERE pid = 'P1' AND sid IN (SELECT sid FROM
  SupplyProduct SP2 WHERE SP2.pid = 'P2');
```

  - Đưa ra sid của các hãng không cung ứng mặt hàng P3:

```
SELECT sid FROM SupplyProduct
WHERE sid NOT IN (SELECT sid From SupplyProduct SP2
  WHERE SP2.pid = 'P3');
```

- So sánh tập hợp: Sử dụng các phép toán <,>, >=,<=,=,!= (<>) kèm với các mệnh đề ANY và ALL
  - Đưa ra tên của các hãng có số nhân viên đông nhất:

```
SELECT sname FROM Supplier
WHERE size >= ALL(SELECT size FROM Supplier)
```

  - Đưa ra sid của hãng cung ứng một mặt hàng với số lượng bằng ít nhất 1 trong số lượng các mặt hàng được cung ứng bởi S2

```
SELECT sid FROM SupplyProduct
WHERE sid != 'S2' AND quantity = ANY(SELECT quantity
    FROM SupplyProduct SP2 WHERE SP2.sid = 'S2');
```

- Kiểm tra tập hợp rỗng với EXISTS và NOT EXISTS
  - EXISTS(câu truy vấn con): nhận giá trị đúng khi câu truy vấn con cho ra kết quả là một quan hệ khác rỗng
  - NOT EXISTS(câu truy vấn con): nhận giá trị đúng khi câu truy vấn con cho ra kết quả là một quan hệ rỗng

- Đưa ra thông tin của các nhà cung cấp đã cung ứng ít nhất một mặt hàng

```
SELECT * FROM Supplier S
WHERE EXISTS (SELECT sid FROM SupplyProduct SP
    WHERE S.sid = SP.sid);
```

- Đưa ra thông tin của các nhà cung cấp không cung ứng mặt hàng nào

```
SELECT * FROM Supplier S
WHERE NOT EXISTS (SELECT * FROM SupplyProduct
    SP WHERE S.sid = SP.sid);
```

Các hàm thư viện

- Hàm tính toán trên nhóm các bản ghi
  - MAX/MIN
  - SUM
  - AVG
  - COUNT
- Hàm tính toán trên bản ghi
  - Hàm toán học: ABS, SQRT, LOG, EXP, SIGN, ROUND
  - Hàm xử lý xâu ký tự: LEN, LEFT, RIGHT, MID
  - Hàm xử lý thời gian: DATE, DAY, MONTH, YEAR, HOUR, MINUTE, SECOND
  - Hàm chuyển đổi kiểu giá trị: FORMAT

SQL Server

```
SELECT * FROM GiangVien
WHERE DATEPART(year, GETDATE()) - DATEPART(year, NgaySinh) > 40
```

PostGreSQL

```
SELECT * FROM "GiangVien"
WHERE date_part('year', current_date) - date_part('year', "NgaySinh") > 40
```

Một số ví dụ với các hàm thư viện

- Có bao nhiêu mặt hàng khác nhau được cung ứng

```
SELECT COUNT(DISTINCT pid)
FROM SupplyProduct;
```

- Có tổng cộng bao nhiêu nhân viên làm cho các hãng ở Paris

```
SELECT SUM(size) FROM Supplier
WHERE city = 'Paris';
```

- Đưa ra số lượng mặt hàng trung bình mà hãng S1 cung ứng

```
SELECT AVG(quantity)
FROM SupplyProduct
WHERE sid = 'S1';
```

Một số truy vấn phức tạp

- Đưa ra tên của hãng S1 và tổng số lượng các mặt hàng mà hãng đó cung ứng

```
SELECT sname, SUM(quantity)
FROM Supplier S, SupplyProduct SP
WHERE S.sid = SP.sid AND S.sid = 'S1'
GROUP BY sname;
```

- Đưa ra mã số các hãng cung ứng và số lượng trung bình các mặt hàng được cung ứng bởi từng hãng

```
SELECT sid, AVG(quantity) FROM SupplyProduct
GROUP BY sid;
```

- Đưa ra mã số các hãng cung ứng mà số lượng mặt hàng trung bình được cung cấp bởi hãng đó là trong khoảng từ 75 đến 100

```
SELECT sid, AVG(quantity) FROM SupplyProduct
GROUP BY sid HAVING AVG(quantity) BETWEEN 75 AND 100
```

Thêm bản ghi vào bảng

- **INSERT INTO** table[(col1,col2,…)]
  VALUES (exp1,exp2,…)
- **INSERT INTO** table[(col1,col2,…)]
  SELECT col1,col2, …
  FROM tab1, tab2, …
  WHERE <dieu_kien>

– Ví dụ

- **INSERT INTO** Student **VALUES** ('1179','Jane','California');
- **INSERT INTO** Student(Id, Name, Suburb)
  VALUES ('1180','David','NewYork');
- **INSERT INTO** Student(Name, Id, Suburb)
  VALUES ('Mary','1181','Texas');
- **INSERT INTO** Student(Id, Name, Suburb)
  VALUES ('1182','John','Ohio'), ('1183','Tom','Georgia'), ('1184','Declan','Arizona');

Xóa bản ghi trong bảng

DELETE FROM <Tên bảng>
WHERE <Điều kiện xóa>;

- Ví dụ:

DELETE FROM SupplyProduct
WHERE sid = 'S4';
DELETE FROM Student
WHERE Suburb = 'Indiana';

Sửa dữ liệu trong bảng

```
UPDATE <tên bảng>
SET (<Tên cột> = Giá trị mới , …)
[WHERE <Điều kiện sửa đổi>];
```

- Ví dụ:
  - Hãng S1 chuyển tới Milan

```
UPDATE Supplier
SET city = 'Milan'
WHERE sid = 'S1';
```

  - Tất cả các mặt hàng được cung cấp với số lượng nhỏ hơn 100 đều tăng số lượng lên 1.5 lần

```
UPDATE SupplyProduct
SET quantity = quantity * 1.5
WHERE quantity < 100;
```

CƠ SỞ DỮ LIỆU

Chương 4

Chương 4 - Lý thuyết thiết kế cơ sở dữ liệu quan hệ

NỘI DUNG:

- Tổng quan về thiết kế CSDL quan hệ
- Phụ thuộc hàm
- Phép tách các sơ đồ quan hệ (SĐQH)
- Các dạng chuẩn đối với các SĐQH

Tổng quan về thiết kế CSDLQH

- Vấn đề của một sơ đồ quan hệ được thiết kế chưa tốt:

  Giả sử ta cần một cơ sở dữ liệu lưu trữ thông tin về các hãng cung ứng. Sơ đồ quan hệ được thiết kế trong đó tất cả các thuộc tính cần thiết được lưu trong đúng 1 quan hệ:

  Suppliers(sid, sname, city, numofemps, product, quantity)

sid	sname	city	NOE	product	quantity
S1	Smith	London	100	Screw	50
S1	Smith	London	100	Nut	100
S2	J&J	Paris	124	Screw	78
S3	Blake	Tokyo	75	Bolt	100

- **Dư thừa dữ liệu**: Hãng nào cung ứng nhiều hơn một mặt hàng thì thông tin của hãng đó sẽ bị lặp lại trong bảng (VD S1), mặt hàng được cung ứng bởi nhiều hãng cũng bị lặp lại (VD Screw)
- **Dị thường dữ liệu khi thêm**: Nếu có một hãng chưa cung cấp mặt hàng nào, vậy giá trị cho thuộc tính product và quantity trong bộ dữ liệu mới được thêm vào sẽ không được xác định
- **Dị thường dữ liệu khi xóa**: Nếu một hãng chỉ cung cấp một mặt hàng, nếu ta muốn xóa thông tin về sự cung cấp này thì ta sẽ mất thông tin về hãng cung cấp
- **Dị thường dữ liệu khi sửa đổi**: Do thông tin bị lặp lại nên việc sửa đổi một bộ dữ liệu có thể dẫn đến việc không nhất quán trong dữ liệu về một hãng nếu sơ sót không sửa đổi trên toàn bộ các bộ giá trị liên quan đến hãng đó

Đề xuất giải pháp

- Nếu sơ đồ trên được thay thế bằng 2 sơ đồ quan hệ
  - *Supp(sid, sname, city, numofemps)*
  - *Supply(sid, product, quantity)*

  thì tất cả các vấn đề nêu ở trên sẽ được loại bỏ. Tuy nhiên, khi tìm kiếm dữ liệu thì phải kết nối 2 bảng chứ không chỉ là chọn và chiếu trên một bảng như ở cách thiết kế trước.

Mục đích của chuẩn hoá

- Xác định được một tập các lược đồ quan hệ, cho phép tìm kiếm thông tin một cách dễ dàng, **đồng thời** tránh được dư thừa dữ liệu.
- Hướng tiếp cận:
  - Một trong những kỹ thuật được sử dụng là Tách các lược đồ quan hệ có vấn đề thành những lược đồ quan hệ chuẩn hơn.
  - Phụ thuộc hàm (functional dependencies) được sử dụng để nhận biết các lược đồ chưa chuẩn và đề xuất hướng cải tiến.

Phụ thuộc hàm

- Định nghĩa:
  - Cho R(U) là một sơ đồ quan hệ với U là tập thuộc tính {A_1, A_2,…,A_n}. X, Y là tập con không rỗng của U.
  - Nói X xác định hàm Y, hay Y là phụ thuộc hàm vào X (viết: $X \rightarrow Y$) nếu với một quan hệ r xác định trên R(U) và với 2 bộ bất kỳ t_1, t_2 thuộc r mà $t_1[X] = t_2[X]$ thì $t_1[Y] = t_2[Y]$
- Bản chất, nếu 2 bộ giống nhau về giá trị của các thuộc tính X thì cũng giống nhau về giá trị của các thuộc tính Y.
- Phụ thuộc hàm là một trường hợp của ràng buộc toàn vẹn, tổng quát hóa khái niệm khóa.

- Ví dụ 1: AB → C

- Cho các quan hệ sau:

Supplier

sid	sname	size	city
S1	Dustin	100	London
S2	Rusty	70	Paris
S3	Lubber	120	London

Product

pid	pname	colour
P1	Screw	red
P2	Screw	green
P3	Nut	red
P4	Bolt	blue

SupplyProduct

sid	pid	quantity
S1	P1	500
S1	P2	400
S1	P3	100
S2	P2	200
S3	P4	100
S2	P3	155

A	B	C	D
a1	b1	c1	d1
a1	b1	c1	d2
a1	b2	c2	d1
a2	b1	c3	d1

- Ví dụ 2: trong cơ sở dữ liệu mẫu dùng trong Chương 3, ta có bảng S, với mỗi giá trị của sid đều tồn tại một giá trị tương ứng cho sname, city và status.

  Do đó, có sid → sname, sid → city, sid → status

Hệ tiên đề Amstrong đối với phụ thuộc hàm

Cho R(U) là 1 sơ đồ quan hệ, U là tập các thuộc tính.

$X,Y,Z,W \subseteq U$, Ký hiệu: $XY = X \cup Y$

- Phản xạ (*reflexivity*) : Nếu $Y \subseteq X$ thì $X \rightarrow Y$
- Tăng trưởng (*augmentation*) : Nếu $X \rightarrow Y$ thì $XZ \rightarrow YZ$
- Bắc cầu (*transitivity*) : Nếu $X \rightarrow Y$, $Y \rightarrow Z$ thì $X \rightarrow Z$

Hệ quả của hệ tiên đề Amstrong

- Luật hợp (*union*): Nếu $X \rightarrow Y$, $X \rightarrow Z$ thì $X \rightarrow YZ$.
- Luật tựa bắc cầu (*pseudo-transitivity*)

  Nếu $X \rightarrow Y$, $WY \rightarrow Z$ thì $XW \rightarrow Z$.
- Luật tách (*decomposition*): Nếu $X \rightarrow Y$, $Z \subseteq Y$ thì $X \rightarrow Z$

- Ví dụ 1:

  Cho tập phụ thuộc hàm {AB→C, C→A}

  Chứng minh: BC → ABC

  C → A BC → AB

  AB → C AB → ABC

  BC → AB, AB → ABC BC → ABC

- Ví dụ 2:

  Cho lược đồ quan hệ R(ABEIJGH) và tập phụ thuộc hàm F = {AB→E, AG→J, BE→I, E→G, GI→H}

  Chứng minh: AB → GH

Bao đóng của một tập phụ thuộc hàm

- **Định nghĩa:**
  - Cho F là một tập phụ thuộc hàm. Bao đóng của F ký hiệu là F^+ là tập lớn nhất chứa các phụ thuộc hàm có thể được suy ra từ các phụ thuộc hàm trong F.
- Đặc điểm của bao đóng của một tập phụ thuộc hàm:
  - Có thể rất lớn
  - Chi phí rất tốn kém cho việc tìm kiếm
- Vấn đề đặt ra: Kiểm tra xem một phụ thuộc hàm có được suy diễn từ một tập phụ thuộc hàm có sẵn không => sử dụng bao đóng của một tập thuộc tính đối với tập phụ thuộc hàm.

Bao đóng của một tập các thuộc tính đối với một tập các phụ thuộc hàm

- Định nghĩa:
  - Cho một sơ đồ quan hệ R(U), F là một tập phụ thuộc hàm trên U. X là tập con của U. Bao đóng của tập thuộc tính X đối với tập F, ký hiệu là X^+_F (X^+), là tập tất cả các thuộc tính được xác định hàm bởi X thông qua tập F

$$X^+ = \{A \in U | X \rightarrow A \in F^+\}$$

- Có thể thấy, định nghĩa về bao đóng của một tập thuộc tính dựa trên bao đóng của tập phụ thuộc hàm.

  =>Thuật toán xác định bao đóng của một tập thuộc tính

Thuật toán 1: Tìm bao đóng của một tập thuộc tính đối với tập phụ thuộc hàm

- **Vào**: Tập hữu hạn các thuộc tính U, tập các pth F trên U, $X \subseteq U$
- **Ra**: X^+_F
- **Thuật toán**

Bước 0: $X^0 = X$

Bước i: Tính X^i từ X^{i-1}

Nếu $\exists\, Y \rightarrow Z \in F$ và $Y \subseteq X^{i-1}$ và $A \in Z$ và $A \notin X^{i-1}$

thì $X^i = X^{i-1} \cup A$

ngược lại, $X^i = X^{i-1}$

Nếu $X^i \neq X^{i-1}$

thì lặp **Bước i**

ngược lai, chuyển **Bước i+1**

Bước cuối cùng: khi $X^i = X^{i-1}$ (không tăng thêm): $X^+_F = X^i$

Ví dụ

- Cho R(U) , U = {A, B, C, D, E, F}
  F = {AB→C, BC→AD, D→E, CF→B}
  Tính $(AB)^+$
- Thực hiện:
  - Bước 0: $X^0 = AB$
  - Bước 1: $X^1 = ABC$ (do AB→ C)
  - Bước 2: $X^2 = ABCD$ (do BC→AD)
  - Bước 3: $X^3 = ABCDE$ (do D→E)
  - Bước 4: $X^4 = ABCDE$

Bổ đề

- $X \rightarrow Y$ được suy diễn từ tập F dựa trên hệ tiên đề Amstrong khi và chỉ khi $Y \subseteq X^+_F$
- Chứng minh:
  - Giả sử $Y=A_1...A_n$, với A_1, ..., A_n là các thuộc tính và $Y \subseteq X^+$
  - Từ Định nghĩa X^+ ta có $X \rightarrow A_i$. Áp dụng tiên đề Amstrong cho mọi i, suy ra $X \rightarrow Y$ nhờ luật hợp.
  - Ngược lại, giả sử có $X \rightarrow Y$, áp dụng hệ tiên đề Amstrong cho mỗi i, ta có $X \rightarrow A_i$, $A_i \in Y$ nhờ luật tách. Từ đó suy ra $Y \subseteq X^+$

Khoá tối thiểu

- Định nghĩa: Cho sơ đồ quan hệ R=<U,F>, U là tập thuộc tính, F là một tập các phụ thuộc hàm xác định trên U. K được gọi là khoá tối thiểu của R nếu:
  - $K \subseteq U$
  - $K \rightarrow U \in F^+$
  - Với mọi $K' \subset K$, thì $K' \rightarrow U \notin F^+$
- Với những gì đã đề cập trong phần bao đóng ở trên, có thể nói, để thỏa mãn là một khoá tối thiểu thì $K^+ = U$ và K là tập thuộc tính nhỏ nhất có tính chất này.

Thuật toán 2: Tìm khoá tối thiểu

- **Vào**: $U = \{A_1, A_2, \ldots, A_n\}$, F
- **Ra**: khoá tối thiểu K xác định được trên U và F
- **Thuật toán**

  Bước 0: $K^0 = U$

  Bước i: Nếu $(K^{i-1} \backslash \{A_i\}) \rightarrow U$

  thì $K^i = K^{i-1} \backslash \{A_i\}$

  ngược lại, $K^i = K^{i-1}$

  Bước n+1: $K = K^n$

Ví dụ

Cho U = {A, B, C, D, E}, F = {AB→C, AC→B, BC→DE}. Tìm một khoá tối thiểu của một quan hệ r xác định trên U và F

Thực hiện

B0: K^0= U = ABCDE

B1: Kiểm tra xem có tồn tại phụ thuộc hàm (K^0\{A})→U (BCDE→U) hay không. Ta cần phải sử dụng Thuật toán 1 để kiểm tra điều kiện tương đương là $(BCDE)^+$ có bằng U không. $(BCDE)^+$= BCDE , khác U. Vậy $K^1 = K^0$ = ABCDE

B2: Tương tự, thử loại bỏ B ra khỏi K^1 ta có $(ACDE)^+$ = ABCDE = U. Vậy $K^2 = K^1$ \ {B} = ACDE

B3: K^3 = ACDE

B4: K^4 = ACE

B5: K^5 = AC

B6: Vậy AC là một khoá tối thiểu mà ta cần tìm

Thuật toán khác tìm tất cả các khóa trong lược đồ quan hệ

Gọi:

- U là tập tất cả các thuộc tính CSDL, F là tập phụ thuộc hàm
- L(left): là các thuộc tính xuất hiện bên trái, R(right): là các thuộc tính xuất hiện ở vế phải
- S(superkey): là tập các siêu khóa, K(key): là tập các khóa

Tập thuộc tính nguồn (TN): gồm các thuộc tính chỉ xuất hiện ở vế trái, không xuất hiện ở vế phải của F và các thuộc tính không xuất hiện ở cả vế trái và vế phải của F.

Vậy TN = U \ R

- **Ví dụ**: Cho sơ đồ U = {A, B, C, D, E},

  F = {AB→C, AC→B, BC→DE}

  L = {A, B, C}, R = {B, C, D, E}, TN = U \ R = {A}

Tập thuộc tính đích (TĐ): gồm các thuộc tính chỉ xuất hiện ở R, không xuất hiện ở L. **Vậy TĐ = R \ L**

- **Ví dụ**: Cho L = {A, B, C}, R = {B, C, D, E}

  TĐ = {D, E}

Tập thuộc tính trung gian (TG): chứa các thuộc tính xuất hiện ở cả L và R. **Vậy TG = L ∩ R**

- **Ví dụ**: Cho L = {A, B, C}, R = {B, C, D, E}

  TG = L ∩ R = {B, C}

Thuật toán khác tìm tất cả các khóa trong lược đồ quan hệ

Thuật toán:

Bước 1: Tìm tập thuộc tính nguồn TN và tập thuộc tính trung gian TG

Bước 2: Nếu TG = ∅ thì K(Key) = TN, và kết thúc thuật toán, xuất ra K của sơ đồ quan hệ <U,F>

Ngược lại, nếu TG ≠ ∅ thì qua Bước 3

Bước 3: Tìm tất cả các tập con X_i của TG

Bước 4: Tìm Siêu khóa (S_i)

Với $\forall X_i$, nếu $(TN \cup X_i)^+ = U$ thì khi đó $S_i = TN \cup X_i$

Bước 5: Tìm Khóa (K_i) bằng cách loại bỏ các siêu khóa không tối thiểu

- Với mọi S_i S_j thuộc S, nếu S_i chứa trong S_j thì loại bỏ S_j ra khỏi tập siêu khóa. Khi đó, tập S còn lại chính là tập khóa cần tìm

- Ví dụ : Cho sơ đồ quan hệ R = <U, F>, với U = {A, B. C}, F = {AB→C, C→A}. Tìm tất cả các khóa thuộc tập cơ sở dữ liệu trên.
- Lời giải:
  - L = {ABC} R = {CA} TN = {B}
  - TG = {AC}≠∅ nên ta làm tiếp Bước 3
  - Ta có tập con X_i của tập TG = {∅, A, C, AC}
  - Lấy từng thuộc tính thuộc tập con X_i của tập TG hợp với TN ta có các thuộc tính sau:

  + S_1 = TN ∪ ∅ = B, có B^+ = B ≠ U nên S_1 = B không là siêu khóa
  + S_2 = TN ∪ A =AB Ta có AB^+ =ABC= U nên S_2=AB là siêu khóa
  + S_3 = TN ∪ C =BC Ta có BC^+ =ABC= U nên S_3=BC là siêu khóa
  + S_4 = TN∪AC =ABC Ta có ABC^+ =ABC=U nên S_4=ABC là siêu khóa

  - Vậy ta có tập siêu khóa S = {AB, BC, ABC}. Tuy nhiên, vì AB ⊂ ABC và BC ⊂ ABC nên loại bỏ ABC ra khỏi tập siêu khóa
  - Vậy ta có, tập K = {AB, BC} là khóa của lược đồ quan hệ

Nhận xét về phụ thuộc hàm

- Từ một tập các phụ thuộc hàm có thể suy diễn ra các phụ thuộc hàm khác
- Trong một tập phụ thuộc hàm cho sẵn, có thể có các phụ thuộc hàm bị coi là dư thừa

→Làm thế nào để có được một tập phụ thuộc hàm tốt?

Hai tập phụ thuộc hàm tương đương

- **Định nghĩa**: Tập phụ thuộc hàm F là phủ của tập phụ thuộc hàm G, nếu $G \subset F^+$, G là phủ của F, nếu $F \subset G^+$, hay F và G tương đương nếu $F^+ = G^+$, ký hiệu là $F \cong G$
- Kiểm tra tính tương đương của 2 tập phụ thuộc hàm
  - Bước 1. Nếu với $\forall$ phụ thuộc hàm $f_i \in F$, f_i có dạng $X_{fi} \rightarrow Y_{fi}$, mà $f_i \in G^+$ thì $F^+ \subseteq G^+$. Kiểm tra $f_i \in G^+$ bằng cách kiểm tra $Y_{fi} \subseteq (X_{fi})^+_G$
  - Bước 2. Tương tự, nếu $\forall$ phụ thuộc hàm $g_j \in G$, mà $g_j \in F^+$ thì $G^+ \subseteq F^+$
  - Bước 3. Nếu $F^+ \subseteq G^+$ và $G^+ \subseteq F^+$ thì $F \cong G$

Ví dụ

- Cho sơ đồ quan hệ R(U) với U = {A, B, C, D, E, F}

  F = {AB→C, D→EF, C→BD}

  G = {AC→B, D→EF, B→CD}

  Hỏi F và G có phải là 2 tập phụ thuộc hàm tương đương hay không?
- Thực hiện:

  Đối với các phụ thuộc hàm trong F
  - f_1= AB→C. AB^+_G = ABCDEF = U. Vậy $f_1 \in G^+$
  - f_2= D→EF ∈ G nên chắc chắn ∈ G^+
  - f_3= C→BD. C^+_G = C không chứa BD. Vậy $f_3 \notin G^+$
  - Kết luận F ≇ G

Thuật toán 3: Tập phụ thuộc hàm không dư thừa

- **Định nghĩa**: Tập phụ thuộc hàm F là không dư thừa nếu không tồn tại $X \rightarrow Y \in F$ sao cho $F \setminus \{X \rightarrow Y\} \approx F$.
- **Thuật toán 3**: Tìm phủ không dư thừa của một tập phụ thuộc hàm
  - Vào: Tập thuộc tính U, $F = \{L_i \rightarrow R_i \mid i = 1..n\}$
  - Ra : Phủ không dư thừa F' của F
  - **Thuật toán**

    Bước 0: $F^0 = F$

    Bước i: Nếu $F^{i-1} \setminus \{L_i \rightarrow R_i\} \approx F^{i-1}$

    thì $F^i = F^{i-1} \setminus \{L_i \rightarrow R_i\}$

    ngược lại, $F^i = F^{i-1}$

    Bước n+1: $F' = F^n$

Phủ tối thiểu của một tập phụ thuộc hàm

- Định nghĩa: F_c được gọi là phủ tối thiểu của một tập phụ thuộc hàm F nếu thỏa mãn ba điều kiện:
  - (Đk1) Với $\forall$ f $\in$ F_c, f có dạng X $\rightarrow$ A, trong đó A là một thuộc tính.
  - (Đk2) Với $\forall$ f = X$\rightarrow$Y $\in$ F_c, không tồn tại A $\in$ X (A là một thuộc tính) mà $(F_c \setminus f) \cup \{(X \setminus A) \rightarrow Y\} \cong F_c$
  - (Đk3) không tồn tại X$\rightarrow$A $\in$ F_c mà $F_c \setminus \{X \rightarrow A\} \cong F_c$

Thuật toán 4: Tìm phủ tối thiểu của một tập phụ thuộc hàm

- **Vào**: Tập thuộc tính U, F = $\{L_i \rightarrow R_i : i = 1..n\}$
- **Ra**: phủ tối thiểu F_c của tập phụ thuộc hàm F
- **Thuật toán**

B.1. Biến đổi F về dạng $F_1 = \{L_i \rightarrow A_j\}$

trong đó A_j là một thuộc tính bất kỳ thuộc U (thoả mãn đk1)

B.2. Loại bỏ thuộc tính thừa trong vế trái của các phụ thuộc hàm

Lần lượt giản ước từng thuộc tính trong vế trái của từng phụ thuộc hàm trong F_1 thu được F_1'. Nếu $F_1' \cong F_1$ thì loại bỏ thuộc tính đang xét

Khi không có sự giản ước nào xảy ra nữa, thu được F_2 thỏa mãn đk2

B.3. Loại bỏ phụ thuộc hàm dư thừa

Lần lượt kiểm tra từng phụ thuộc hàm f. Nếu $F_2 \setminus f \cong F_2$ thì loại bỏ f

Khi không còn phụ thuộc hàm nào có thể loại bỏ thì thu đươc F_3 thoả mãn đk3

B.4. $F_c = F_3$

Ví dụ 1

- U = {A,B,C}

  F = {A→BC, B→C, A→B, AB→C}. Tìm phủ tối thiểu của F?

  - F_1 = {A→B, A→C, B→C, AB→C}
  - Xét các phụ thuộc hàm trong F_1 mà vế trái có nhiều hơn một thuộc tính, chỉ có AB→C. Giản ước A thì ta còn B→C có trong F_1, vậy A là thuộc tính thừa. Tương tự ta cũng tìm được B là thừa, vậy loại bỏ luôn AB→C khỏi F_1. F_2 = {A→B, A→C, B→C}
  - Bỏ phụ thuộc hàm thừa: A→C là thừa.

    Vậy F_c = {A→B, B→C}

Ví dụ 2

Tìm phủ tối thiểu của tập phụ thuộc hàm sau:
F = {A→B, ABCD→E, EF→G, ACDF→EG}

- Bước 1: có F_1={A→B, ABCD→E, EF→G, ACDF→E, ACDF→G}
- Bước 2: F_2^0=F_1={A→B, ABCD→E, EF→G, ACDF→E, ACDF→G}
  - Loại bỏ thuộc tính thừa ở vế trái của 4 pth ABCD→E, EF→G, ACDF→E và ACDF→G
  - Xét ABCD→E: có B dư thừa, vì $(ACD)^+_{F2}{}^0$=ACDBE, chứa E, vậy ACD→E được suy diễn ra từ F_2^0
  - Xét EF→G: không dư thừa
  - Xét ACDF→E: dư thừa F, còn lại ACD→E
  - Xét ACDF→G: không dư thừa
  - Vậy F_2^2={A→B, ACD→E, EF→G, ACDF→G}
- Bước 3: F_3^0=F_2^2={A→B, ACD→E, EF→G, ACDF→G}
  - Loại bỏ các phụ thuộc hàm dư thừa trong F_3^0
  - Xét ACDF→G: $(ACDF)^+$=ACDFBEG, chứa G. Vậy loại bỏ phụ thuộc hàm này trong F_3^0,

  Vậy F_C= F_3^1={A→B, ACD→E, EF→G}

Phép tách các sơ đồ quan hệ

- Mục đích
  - Thay thế một sơ đồ quan hệ $R(A_1, A_2, \ldots, A_n)$ bằng một tập các sơ đồ con $\{R_1, R_2, \ldots, R_k\}$ trong đó $R_i \subseteq R$ và $R = R_1 \cup R_2 \cup \ldots \cup R_k$

- Yêu cầu của phép tách
  - Bảo toàn thuộc tính, ràng buộc
  - Bảo toàn dữ liệu

Phép tách không mất mát thông tin

- Định nghĩa: Cho sơ đồ quan hệ R(U) phép tách R thành các sơ đồ con $\{R_1, R_2, \ldots, R_k\}$ được gọi là phép tách không mất mát thông tin đối với một tập phụ thuộc hàm F nếu với mọi quan hệ r xác định trên R thỏa mãn F thì:

  $r = \Pi_{R1}(r) \bowtie \Pi_{R2}(r) \bowtie \ldots \bowtie \Pi_{Rk}(r)$

- Ví dụ: Phép tách **mất mát** thông tin

  Supplier(sid, sname, city, NOE, pid, pname, colour, quantity)

  →S1(sid, sname, city, NOE) và SP1(pid, pname, colour, quantity)
- Ví dụ: Phép tách **không mất mát** thông tin

  →S1(sid, sname, city, NOE) và

  SP2(sid, pid, pname, colour, quantity)

Định lý tách đôi

- Cho sơ đồ quan hệ R(U), tập phụ thuộc hàm F , phép tách R thành $R_1(U_1)$, $R_2(U_2)$ là một phép tách không mất mát thông tin nếu một trong hai phụ thuộc hàm sau là thỏa mãn trên F^+:

$$U_1 \cap U_2 \rightarrow U_1 - U_2$$
$$U_1 \cap U_2 \rightarrow U_2 - U_1$$

- Hệ quả: Cho sơ đồ quan hệ R(U) và phụ thuộc hàm $X \rightarrow Y$ thỏa mãn trên R(U). Phép tách R thành hai lược đồ con $R_1(U_1)$, $R_2(U_2)$ là một phép tách không mất mát thông tin với:

$$U_1 = XY$$
$$U_2 = XZ$$
$$Z = U \setminus XY$$

Thuật toán 5: Kiểm tra tính không mất mát thông tin của một phép tách

- **Vào**: R(A_1, A_2, …, A_n), F, phép tách {R_1, R_2, …, R_k}
- **Ra**: phép tách là mất mát thông tin hay không
- **Thuật toán**

Bước 1. Thiết lập một bảng k hàng, n cột

Nếu A_j là thuộc tính của R_i thì điền a_j vào ô (i,j).

Nếu không thì điền b_{ij}.

Bước i. Xét f = X→Y ∈F

Nếu ∃ hai hàng t1, t2 thuộc bảng: t1[X] = t2[X] thì đồng nhất t1[Y] = t2[Y], ưu tiên về giá trị a

Lặp cho tới khi không thể thay đổi được giá trị nào trong bảng

Bước cuối. Nếu bảng có một hàng gồm các kí hiệu a_1, a_2, … , a_n thì phép tách là không mất mát thông tin

ngược lại, phép tách không bảo toàn thông tin

Ví dụ

- R=ABCD được tách thành R_1=AB, R_2 =BD, R_3=ABC, R_4=BCD, F = {A→C, B→C, CD→B, C→D}
- Bước 1: Tạo bảng gồm 4 hàng, 4 cột
- Bước 2: Từ A→C, thay hàng 1, cột 3 bằng a_3
- Bước 3: Từ B→C, thay hàng 2, cột 3 bằng a_3
- Bước 4: Từ C→D, thay ô (1,4) và (3,4) bằng a_4
- Vậy, có hai hàng có toàn các giá trị a_j, chứng tỏ phép tách đã cho không mất mát thông tin

	A	B	C	D
R_1	a_1	a_2	b_{13}	b_{14}
R_2	b_{21}	a_2	b_{23}	a_4
R_3	a_1	a_2	a_3	b_{34}
R_4	b_{41}	a_2	a_3	a_4

	A	B	C	D
R_1	a_1	a_2	a_3	b_{14}
R_2	b_{21}	a_2	a_3	a_4
R_3	a_1	a_2	a_3	b_{34}
R_4	b_{41}	a_2	a_3	a_4

	A	B	C	D
R_1	a_1	a_2	a_3	a_4
R_2	b_{21}	a_2	a_3	a_4
R_3	a_1	a_2	a_3	a_4
R_4	b_{41}	a_2	a_3	a_4

Phép tách bảo toàn tập phụ thuộc hàm

- Hình chiếu của tập phụ thuộc hàm

  Cho sơ đồ quan hệ R, tập phụ thuộc hàm F, phép tách $\{R_1, R_2, \dots, R_k\}$ của R trên F.

  Hình chiếu F_i của F trên R_i là tập tất cả $X \rightarrow Y \in F^+$:

  $$XY \subseteq R_i$$

- Phép tách sơ đồ quan hệ R thành $\{R_1, R_2, \dots, R_k\}$ là một phép tách bảo toàn tập phụ thuộc hàm F nếu

  $$(F_1 \cup F_2 \dots \cup F_k)^+ = F^+$$

  hay hợp của tất cả các phụ thuộc hàm trong các hình chiếu của F lên các sơ đồ con sẽ suy diễn ra các phụ thuộc hàm trong F.

Ví dụ

- **Ví dụ 1**: R = {A, B, C} F = { A→B, B→C, C→A} được tách thành R_1 = AB, R_2 = BC. Phép tách này có phải là bảo toàn tập phụ thuộc hàm không?
- **Ví dụ 2**: R = {A, B, C} , F = {AB→C, C→B} được tách thành R_1 = AB, R_2 = BC. Phép tách này có bảo toàn tập phụ thuộc hàm không, có mất mát thông tin không?
- **Ví dụ 3**: R = { A, B, C, D} , F = {A→B, C→D} được tách thành R_1 = AB, R_2 = CD. Phép tách này có bảo toàn tập phụ thuộc hàm không, có mất mát thông tin không?
- Vậy một phép tách có bảo toàn tập phụ thuộc hàm thì không đảm bảo là nó sẽ không mất mát thông tin và ngược lại

Các dạng chuẩn đối với SĐQH

- Vấn đề thiết kế CSDL quan hệ:
  - Có cần thiết phải tinh chỉnh thiết kế?
  - Thiết kế có được đã là tốt hay chưa?

  => các dạng chuẩn: có một vài dạng chuẩn, có thể coi như là một số các vấn đề về dư thừa dữ liệu hay dị thường dữ liệu đã được ngăn ngừa hay tối thiểu hóa
- Các dạng chuẩn mà chúng ta quan tâm
  - Dạng chuẩn 1 (1NF)
  - Dạng chuẩn 2 (2NF)
  - Dạng chuẩn 3 (3NF)
  - Dạng chuẩn Boyce-Code (BCNF)

Dạng chuẩn 1 (1NF)

- **Định nghĩa**: Một sơ đồ quan hệ R được gọi là ở dạng chuẩn 1 nếu tất cả các miền giá trị của các thuộc tính trong R đều chỉ chứa giá trị nguyên tố
  - Giá trị nguyên tố là giá trị mà không thể chia nhỏ ra được nữa
- Một quan hệ r xác định trên sơ đồ quan hệ ở dạng chuẩn 1 thì quan hệ đấy là ở dạng chuẩn 1
- Ví dụ: Quan hệ không ở dạng chuẩn 1 và quan hệ sau khi chuẩn hóa về dạng chuẩn 1

sname	city	product	
		name	price
Blake	London	Nut	100
		Bolt	120
Smith	Paris	Screw	75

sname	city	item	price
Blake	London	Nut	100
Blake	London	Bolt	120
Smith	Paris	Screw	75

Dạng chuẩn 2 (2NF)

- **Định nghĩa**: Một sơ đồ quan hệ R được coi là ở dạng chuẩn 2 nếu
  - Sơ đồ quan hệ này ở 1NF
  - Tất cả các thuộc tính không khoá đều phụ thuộc hàm đầy đủ vào khoá chính

    (Lưu ý, A là một thuộc tính khoá nếu A thuộc một khoá tối thiểu nào đó của R. Ngược lại A là thuộc tính không khoá)

Phụ thuộc hàm đầy đủ

- Định nghĩa: Cho sơ đồ quan hệ R(U), F là tập phụ thuộc hàm trên R. X, Y $\subseteq$ U. Y được gọi là phụ thuộc đầy đủ vào X nếu:
  - $X \rightarrow Y$ thuộc F^+
  - không tồn tại $X' \subset X$ $(X' \neq X)$: $X' \rightarrow Y \in F^+$

- Các phụ thuộc hàm không đầy đủ còn gọi là phụ thuộc bộ phận

Ví dụ

- Sales(sid, sname, city, item, price)
- F = {sid→(sname,city), (sid,item)→price}
- Khoá chính (sid,item), ta có sname, city không phụ thuộc hàm đầy đủ vào khoá chính => Quan hệ Sales không thuộc 2NF
- S(sid, sname, city) và Sales (sid, item, price) là quan hệ thuộc 2NF

Dạng chuẩn 3 (3NF)

- Định nghĩa: Một sơ đồ quan hệ R được coi là ở dạng chuẩn 3 nếu
  - Sơ đồ quan hệ này ở 2NF
  - Mọi thuộc tính không khoá đều không phụ thuộc bắc cầu vào khoá chính
- Định nghĩa: Cho sơ đồ quan hệ R(U). F là tập phụ thuộc hàm trên R(U). X,Y, Z $\subseteq$ U. Nói Z là phụ thuộc bắc cầu vào X nếu ta có X→Y, Y→Z thuộc F^+. Ngược lại, ta nói Z không phụ thuộc bắc cầu vào X

Ví dụ

- Ví dụ 1: Trong ví dụ tách về dạng chuẩn 2, có: S (sid, sname, city) và Sales(sid, item, price).

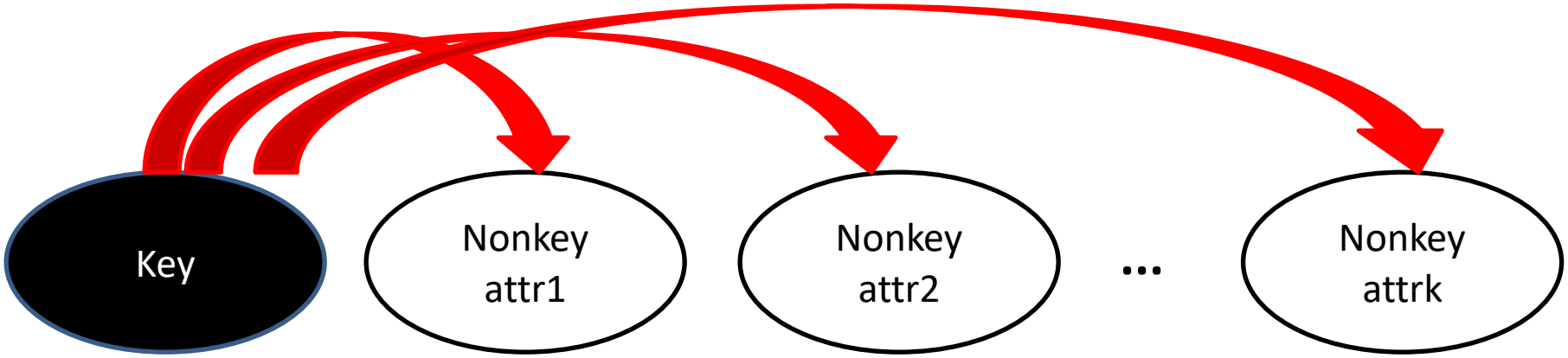
  Xét quan hệ S, pth sid → (sname, city) tồn tại trên S, sid là khoá chính, các thuộc tính không khoá sname, city đều phụ thuộc trực tiếp vào sid. S thuộc 3NF. Tương tự, có Sales cũng thuộc 3NF
- Ví dụ 2:
  - ItemInfo(item, price, discount). F = {item→price, price→discount}. Khoá chính là item, thuộc tính không khoá discount phụ thuộc bắc cầu vào khoá chính item. Vậy quan hệ này không ở 3NF.
  - ItemInfo(item, price) và Discount(price, discount) thuộc 3NF.

Dạng chuẩn Boyce-Codd

- Định nghĩa: Một sơ đồ quan hệ R(U) với một tập phụ thuộc hàm F được gọi là ở dạng chuẩn Boyce-Codd (BCNF) nếu với $\forall X \rightarrow A \in F^+$ thì
  - A là thuộc tính xuất hiện trong X *hoặc*
  - X chứa một khoá của quan hệ R.
- Trong một quan hệ ở BCNF, các phụ thuộc hàm không tầm thường duy nhất là một khóa xác định một số thuộc tính. (*không có thuộc tính khóa phụ thuộc vào thuộc tính không khóa*). Do đó, mỗi bộ được xem là một thực thể hoặc liên kết, được nhận diện bởi một khóa và được mô tả bởi các thuộc tính còn lại.

Dạng chuẩn Boyce-Codd (tiếp)

- Hình oval thể hiện thuộc tính/tập thuộc tính
- Các cung thể hiện các phụ thuộc hàm



- Nếu một quan hệ ở BCNF, thì mỗi trường của mỗi bộ chứa thông tin không thể được suy diễn ra từ các giá trị trong các trường khác (mà chỉ sử dụng các phụ thuộc hàm)

Dạng chuẩn Boyce-Codd (tiếp)

- Ví dụ
  - R = {A,B,C} ; F = {AB→C , C→B}.
  - R không phải ở BCNF vì ∃ C→B, C không phải là khoá
- Chú ý:
  - Một quan hệ thuộc 3NF thì chưa chắc đã thuộc BCNF. Nhưng một quan hệ thuộc BCNF thì thuộc 3NF

Tách bảo toàn tập phụ thuộc hàm về 3NF

- **Vào**: R(U), F (giả thiết F là phủ tối thiểu)
- **Ra**: Phép tách bảo toàn tập phụ thuộc hàm về 3NF
- **Thuật toán**

  Bước 1. Với các $A_i \in U$, $A_i \notin F$ thì loại A_i khỏi R và lập một quan hệ mới cho các A_i

  Bước 2. Nếu $\exists$ $f \in F$, f chứa tất cả các thuộc tính của R (đã bỏ các A_i ở bước trên) thì kết quả là R

  Bước 3. Ngược lại, với mỗi $X \rightarrow A \in F$, xác định một quan hệ $R_i(XA)$.

  Nếu $\exists$ $X \rightarrow A_i$, $X \rightarrow A_j$ thì tạo một quan hệ chung $R'(XA_iA_j)$

Ví dụ

Cho R = {A,B,C,D,E,F,G}

F = {A→B, ACD→E, EF→G} (đã tối thiểu)

- Xác định phép tách bảo toàn tập phụ thuộc hàm về 3NF

  Bước 1. Không lập được quan hệ nào mới.

  Bước 2. !∃ f ∈ F: f chứa tất cả các thuộc tính của R

  Bước 3.

  A→B ⇨ R1(AB)

  ACD→E ⇨ R2(ACDE)

  EF→G ⇨ R3(EFG)

Tách không mất mát thông tin và bảo toàn tập phụ thuộc hàm về 3NF

- Yêu cầu:
  - Bảo toàn tập phụ thuộc hàm (như thuật toán trên)
  - Đảm bảo là có một lược đồ con chứa khoá của lược đồ được tách
- Các bước tiến hành

  Bước 1. Tìm một khoá tối thiểu của lược đồ quan hệ R đã cho

  Bước 2. Tách lược đồ quan hệ R theo phép tách bảo toàn tập phụ thuộc hàm.

  Bước 3. Nếu một trong các sơ đồ con có chứa khoá tối thiểu thì kết quả của Bước 2 là kết quả cuối cùng

  Ngược lại, thêm vào kết quả đó một sơ đồ quan hệ được tạo bởi khoá tối thiểu tìm được ở Bước 1

Ví dụ

- Cho R(U) trong đó U = {A,B,C,D,E,F,G}.
  F = {A→B, ACD→E, EF→G}
- Tìm một khoá tối thiểu của R:

  K^0 = ABCDEFG

  $K^1 = K^0$ do nếu loại A thì BCDEFG → U không thuộc F^+

  $K^2 = K^1 \setminus \{B\}$ = ACDEFG do ACDEFG → U thuộc F^+

  $K^3 = K^2$ do nếu loại C thì ADEFG → U không thuộc F^+

  $K^4 = K^3$ do nếu loại D thì ACEFG → U không thuộc F^+

  $K^5 = K^4 \setminus \{E\}$ = ACDFG do ACDFG → U thuộc F^+

  $K^6 = K^5$ do nếu loại F thì ACDG → U không thuộc F^+

  $K^7 = K^6 \setminus \{G\}$ = ACDF do ACDF → U thuộc F^+

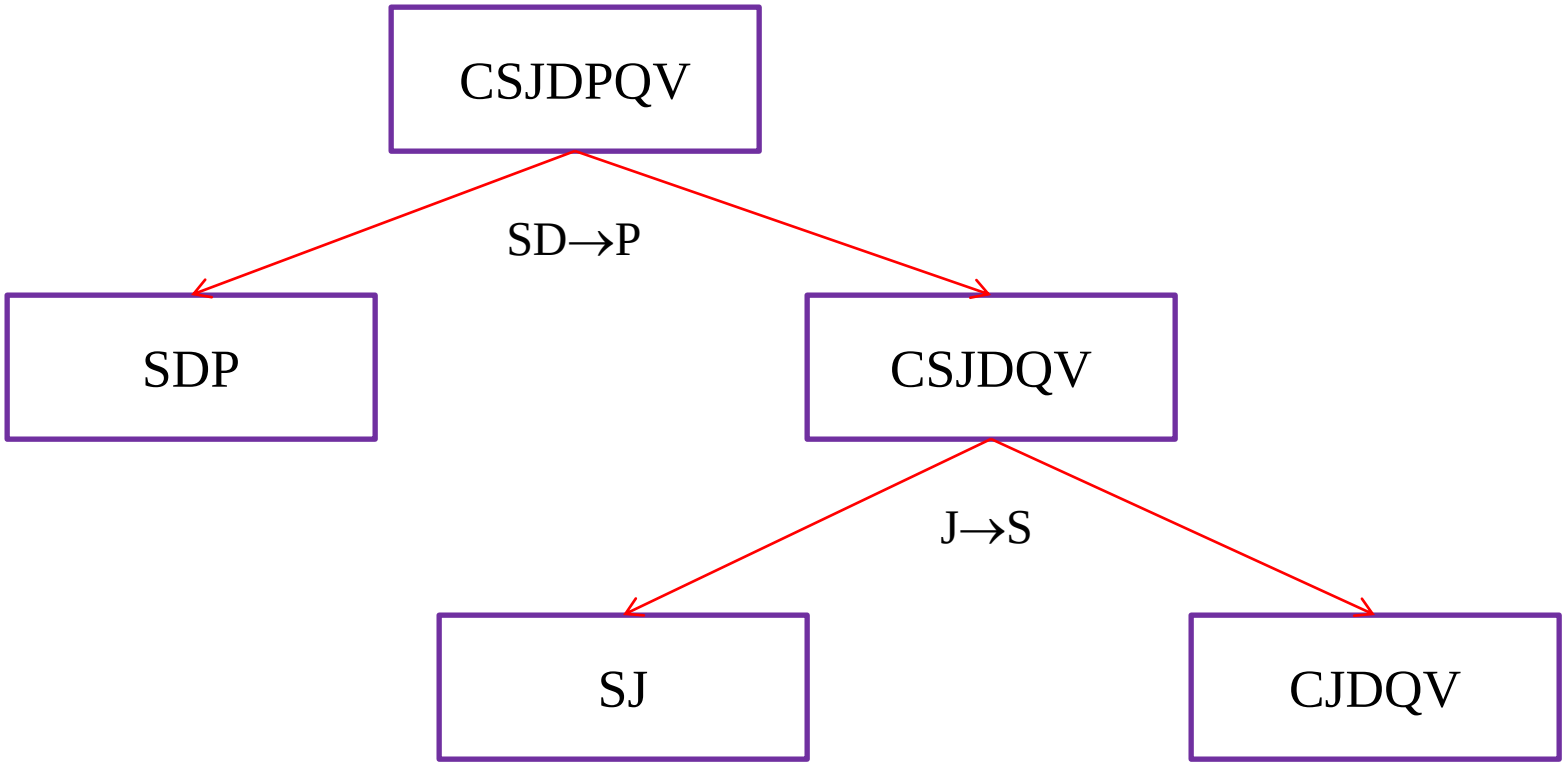
  Vậy khoá tối thiểu cần tìm là ACDF
- Tách bảo toàn tập phụ thuộc hàm thành 3 sơ đồ con R_1 = AB, R_2= ACDE, R_3 = EFG
- Do khoá ACDF không nằm trong bất kỳ một sơ đồ con nào trong 3 sơ đồ con trên, ta lập một sơ đồ con mới R_4 = ACDF. Kết quả cuối cùng ta có phép tách R thành 4 sơ đồ con {R_1, R_2, R_3, R_4} là một phép tách không mất mát thông tin và bảo toàn tập phụ thuộc hàm

Tách không mất mát thông tin về BCNF

- **Vào**: Sơ đồ quan hệ R, tập phụ thuộc hàm F.
- **Ra**: phép tách không mất mát thông tin bao gồm một tập các sơ đồ con ở BCNF với các phụ thuộc hàm là hình chiếu của F lên sơ đồ đó.
- Cách tiến hành
  - Giả sử R không ở BCNF, $X \subset R$, A là một thuộc tính trong R, $X \rightarrow A$ là một phụ thuộc hàm gây ra vi phạm BCNF. Tách R thành R–A và XA
  - Nếu cả R–A và XA chưa ở BCNF, tiếp tục thực hiện việc tách như trên.

- **Ví dụ**: Contracts(contractid, supplierid, projectid, deptid, partid, qty, value)
  - Viết gọn là CSJDPQV
- Các ràng buộc toàn vẹn:

  C→CSJDPQV, JP→C, SD→P, J→S



Bài tập

Cho sơ đồ S(U), U = {A, B, C, D, E, F, G, H}, tập phụ thuộc hàm F={AB →CDE, CD→E, ABC→FG}

Hãy chuẩn hóa S về dạng chuẩn 3 với phép tách bảo toàn thông tin và phụ thuộc hàm.

Kết luận

- Tầm quan trọng của thiết kế CSDL
  - ảnh hưởng đến chất lượng dữ liệu lưu trữ
  - Hiệu quả của việc khai thác dữ liệu
- Mục đích của thiết kế CSDL:
  - Tránh dư thừa dữ liệu
  - Tránh dị thường dữ liệu khi thêm/xoá/sửa đổi
  - Hiệu quả trong tìm kiếm

➢ Đưa về các dạng chuẩn

  - 2NF: giản ước sự dư thừa để tránh các dị thường khi cập nhật
  - 3NF: tránh các dị thường khi thêm/xoá

CƠ SỞ DỮ LIỆU

Chương 5,6,7

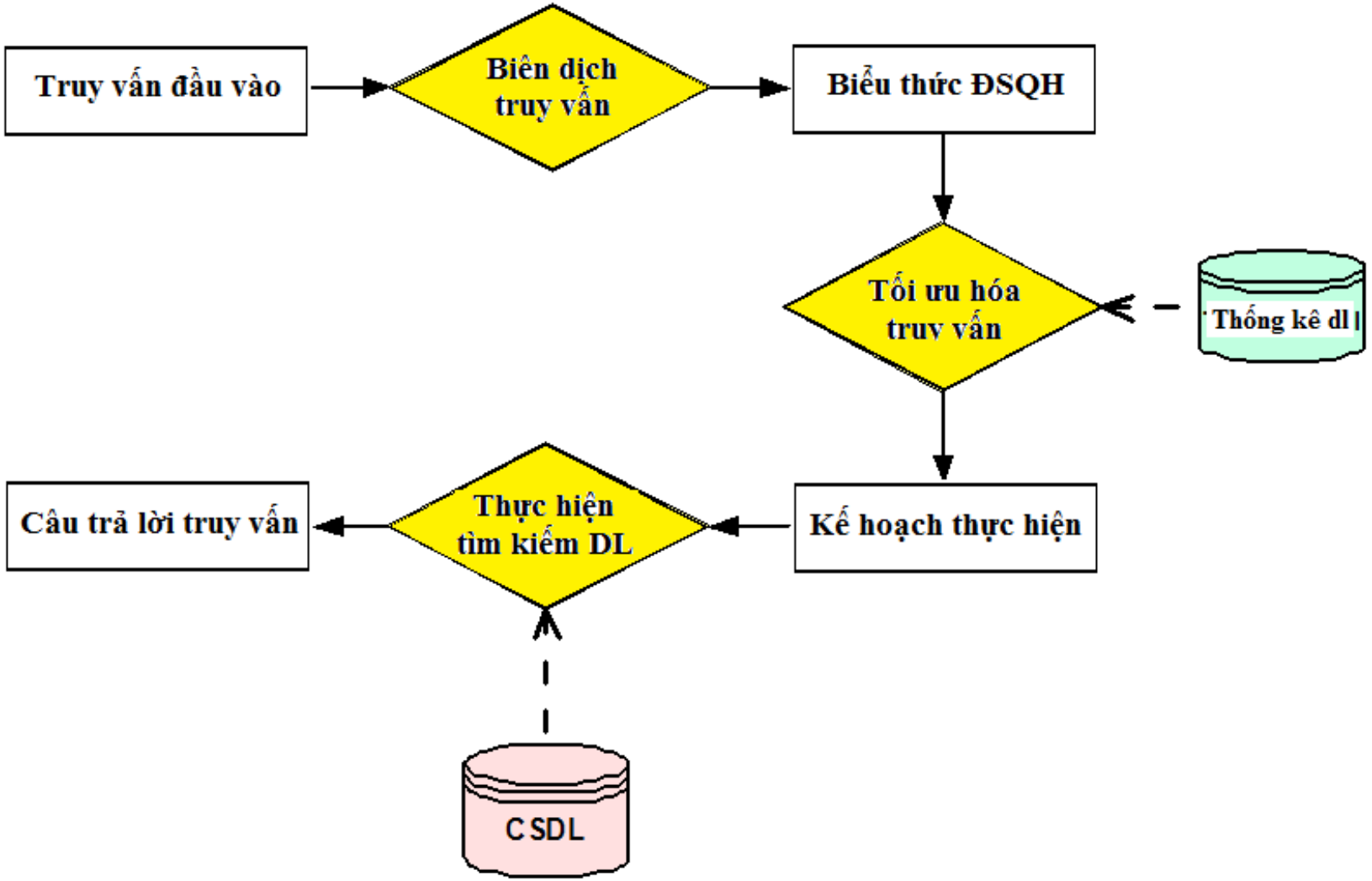
Chương 5 - Tối ưu hóa câu truy vấn

NỘI DUNG:

- Tổng quan về xử lý truy vấn
- Tối ưu hóa các biểu thức đại số quan hệ

5.1. Tổng quan về xử lý truy vấn

- Xử lý một truy vấn bao gồm 3 bước chính:
  - Phân tích và Biên dịch câu truy vấn: dịch câu truy vấn từ dạng ngôn ngữ bậc cao thành một ngôn ngữ biểu diễn dữ liệu bên trong để máy tính có thể thao tác trên đó. Một biểu diễn thích hợp là bằng ngôn ngữ đại số quan hệ
  - Tối ưu hóa câu truy vấn: Mục tiêu của bước tối ưu hóa là chọn ra một kế hoạch thực hiện câu truy vấn có chi phí thấp nhất.
  - Thực hiện đánh giá truy vấn: Từ một kế hoạch thực hiện có được do Trình tối ưu hóa cung cấp, hệ thống sẽ tiến hành thực hiện các thao tác trên dữ liệu trong CSDL và đưa ra câu trả lời cho truy vấn đó.


Truy vấn đầu vào
Biên dịch truy vấn
Biểu thức ĐSQH
Tối ưu hóa truy vấn
Thống kê dl
Kế hoạch thực hiện
Thực hiện tìm kiếm DL
Câu trả lời truy vấn
CSDL

Tối ưu hóa câu truy vấn

Mục tiêu của bước tối ưu hóa là chọn ra một kế hoạch thực hiện câu truy vấn có chi phí thấp nhất.

- Để thực hiện được điều này, trước tiên cần biến đổi một biểu thức ĐSQH đầu vào thành một biểu thức ĐSQH tương đương nhưng có thể xử lý được một cách hiệu quả và ít tốn kém hơn. Bước này gọi là tối ưu hóa đại số.
- Tiếp theo đó, cần phải đặc tả các thuật toán đặc biệt tiến hành thực thi các phép toán , chọn một chỉ dẫn cụ thể nào đó để sử dụng.
- Các dữ liệu thống kê về CSDL sẽ giúp ta trong quá trình xem xét và lựa chọn. Ví dụ như: Số bộ trong quan hệ; Kích thước của một bộ; Số khối (block) chứa các bộ của quan hệ; Số bộ của quan hệ mà một khối có thể chứa; Các thông tin về cơ chế truy nhập, chỉ dẫn trên quan hệ
- Chi phí cho việc thực hiện một truy vấn được đo bởi chi phí sử dụng tài nguyên như: việc truy cập đĩa, thời gian CPU dùng để thực hiện truy vấn.
- Chương này tập trung vào việc đánh giá các biểu thức đại số quan hệ chứ không đi vào chi tiết tính toán chi phí cho việc thực hiện đánh giá truy vấn.

Đánh giá biểu thức ĐSQH

- Sau bước phân tích và biên dịch, ta có một truy vấn được biểu diễn bằng một biểu thức đại số quan hệ bao gồm nhiều phép toán và tác động lên nhiều quan hệ khác nhau, cần tiến hành đánh giá biểu thức này. Có 2 hướng tiếp cận: (i) Vật chất hóa (Materialize), và (ii) Đường ống (Pipeline).
- Vật chất hóa: lần lượt đánh giá các phép toán theo một thứ tự thích hợp. Kết quả của việc đánh giá mỗi phép toán sẽ được lưu trong một quan hệ trung gian tạm thời để sử dụng làm đầu vào cho các phép toán tiếp theo. Điểm bất lợi là cần các quan hệ trung gian (*ghi ra đĩa có chi phí khá lớn*).
- Đường ống: kết hợp một vài phép toán quan hệ vào một đường ống của các phép toán. Trong đường ống thì kết quả của một phép toán được chuyển trực tiếp cho phép toán tiếp theo mà không cần phải lưu lại trong quan hệ trung gian. Cách tiếp cận thứ hai sẽ hạn chế được nhược điểm của cách tiếp cận đầu tiên, nhưng có những trường hợp, ta bắt buộc phải vật chất hóa chứ không dùng đường ống được.

Đánh giá biểu thức ĐSQH (tiếp)

- Ví dụ: Chúng ta có một biểu thức đại số quan hệ gồm 2 phép toán: kết nối và chiếu.

$$\prod_{\{sname,\ pname,\ quantity\}} (S * SP * P)$$

- Trong cách tiếp cận vật chất hóa, xuất phát từ phép toán ở mức thấp nhất là phép kết nối tự nhiên, kết quả của phép kết nối này sẽ được lưu trong một quan hệ trung gian. Sau đó , đọc từ quan hệ trung gian này để tiến hành chiếu lấy kết quả mong muốn.
- Trong cách tiếp cận đường ống, khi một bộ được sinh ra trong phép kết nối 2 quan hệ, bộ này sẽ được chuyển trực tiếp đến phép chiếu để xử lý và kết quả được ghi vào quan hệ đầu ra. Quan hệ kết quả sẽ được tạo lập một cách trực tiếp.

5.2. Tối ưu hóa các biểu thức ĐSQH

- Mục tiêu là tổ chức lại trình tự thực hiện các phép toán trong biểu thức để giảm chi phí thực hiện đánh giá biểu thức đó.
- Trong quá trình tối ưu hóa, ta biểu diễn một biểu thức ĐSQH dưới dạng một cây toán tử. Trong cây thì các nút lá là các quan hệ có mặt trong biểu thức, các nút trong là các phép toán trong biểu thức
- Ví dụ : Đưa ra tên hãng cung ứng mặt hàng có mã là 'P1':

  Select sname From S, SP Where S.sid = SP.sid And pid = 'P1'
- Biểu thức ĐSQH tương ứng là ?
- Cây toán tử tương ứng là ?

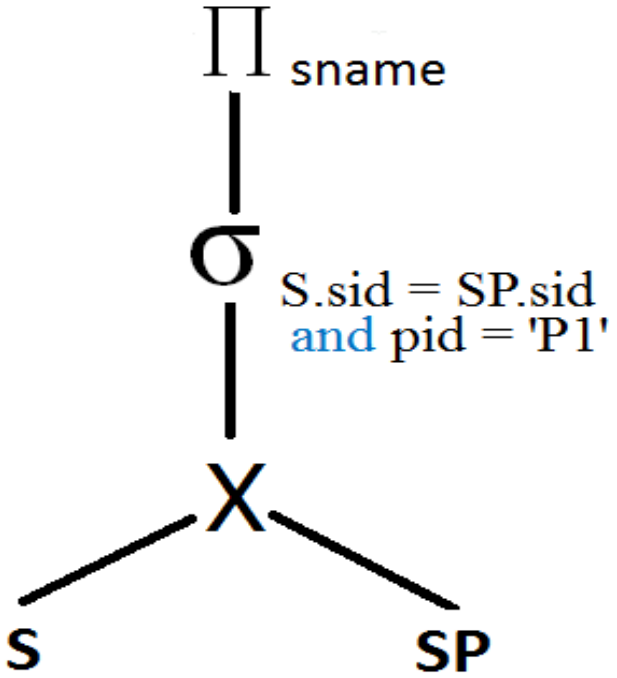
Ví dụ

Cho CSDL gồm các quan hệ:

S (sid, sname, size, city)

P (pid, pname, colour, weight, city)

SP (sid, pid, quantity)



- Đưa ra tên hãng cung ứng mặt hàng có mã là 'P1‘

Select sname From S, SP Where S.sid = SP.sid and pid = 'P1‘

➔ Biểu thức đại số quan hệ và cây toán tử:

$$\prod_{\text{sname}} \left(\sigma_{\substack{\text{S.sid = SP.sid} \\ \text{and pid = 'P1'}}} (\text{S x SP}) \right)$$

Các chiến lược tối ưu tổng quát

1. Đẩy phép chọn và phép chiếu xuống thực hiện sớm nhất có thể: vì hai phép toán này giúp làm giảm kích thước của quan hệ trước khi thực hiện các phép toán 2 ngôi
2. Nhóm dãy các phép chọn và chiếu: Sử dụng chiến lược này nếu như có một dãy các phép chọn hoặc dãy các phép chiếu trên cùng một quan hệ
3. Kết hợp phép chọn và tích Đề các thành phép kết nối: Nếu kết quả của một phép tích Đề các là đối số của 1 phép chọn có điều kiện chọn là phép so sánh giữa các thuộc tính trên 2 quan hệ tham gia tích Đề các thì ta nên kết hợp 2 phép toán thành phép kết nối.
4. Tìm các biểu thức con chung trong biểu thức đại số quan hệ để đánh giá chỉ một lần
5. Xác định các phép toán có thể được đưa vào đường ống và thực hiện đánh giá chúng theo đường ống
6. Xử lý các tệp dữ liệu trước khi tiến hành tính toán: Tạo lập chỉ dẫn hay sắp xếp tệp dữ liệu có thể góp phần làm giảm chi phí của các phép tính trung gian
7. Ước lượng chi phí và lựa chọn thứ tự thực hiện: Do với mỗi câu truy vấn có thể có nhiều cách khác nhau để thực hiện, với việc ước lượng chi phí (số phép tính, tài nguyên sử dụng, dung tích bộ nhớ, thời gian thực hiện ..) ta có thể chọn cách đánh giá biểu thức ĐSQH có chi phí nhỏ nhất.

Các phép biến đổi tương đương biểu thức ĐSQH

- Hai biểu thức ĐSQH E_1 và E_2 là tương đương nếu chúng cho cùng một kết quả khi áp dụng trên cùng một tập các quan hệ
- Trong phần này, ta có các ký hiệu dạng sau: E_1, E_2, E_3, … là các biểu thức đại số quan hệ; F_1, F_2, F_3, … là các điều kiện chọn hoặc là các điều kiện kết nối; X_1, X_2, … Y, Z, U_1, U_2, … là các tập thuộc tính

1. Quy tắc kết hợp của phép tích Đề các và kết nối

$$(E_1 \times E_2) \times E_3 \equiv E_1 \times (E_2 \times E_3)$$

$$(E_1 * E_2) * E_3 \equiv E_1 * (E_2 * E_3)$$

$$(E_1 \underset{F1}{\triangleright\!\triangleleft} E_2) \underset{F2}{\triangleright\!\triangleleft} E_3 \equiv E_1 \underset{F1}{\triangleright\!\triangleleft} (E_2 \underset{F2}{\triangleright\!\triangleleft} E_3)$$

- Qui tắc này sử dụng cho chiến lược số 7. Thứ tự thực hiện các phép kết nối hay tích Đề các là rất quan trọng vì kích thước của quan hệ trung gian có thể rất lớn. Lựa chọn thứ tự tùy thuộc vào kích thước của các quan hệ tham gia phép toán và cả ngữ nghĩa của quan hệ (mối liên hệ)

- Ví dụ: S* SP * P có thể được thực hiện theo 3 thứ tự như sau
  1) (S*SP)*P
  2) (S*P)*SP
  3) S*(SP*P)

  Xét theo ngữ nghĩa S, P không kết nối được nên (1) và (3) là tốt hơn (2). Xét về kích thước thì (3) tốt hơn (1) vì S có 4 thuộc tính còn P có 3 thuộc tính, tuy nhiên, cũng còn tùy thuộc vào lực lượng của 2 quan hệ S và P nữa

Các phép biến đổi tương đương biểu thức ĐSQH

2. Quy tắc giao hoán trong phép tích Đề các và kết nối

$$E_1 \times E_2 \equiv E_2 \times E_1$$

$$E_1 * E_2 \equiv E_2 * E_1$$

$$E_1 \rhd\lhd_F E_2 \equiv E_2 \rhd\lhd_F E_1$$

3. Quy tắc đối với dãy các phép chiếu

$$\Pi_{X_1}(\Pi_{X_2} \ldots \Pi_{X_n}(E)\ldots) \equiv \Pi_{X_1}(E)$$

$$X_1 \subseteq X_2 \subseteq \ldots \subseteq X_n$$

4. Quy tắc đối với dãy các phép chọn

$$\sigma_{F1}(\sigma_{F2} \ldots . \sigma_{Fn}(E)\ldots) \equiv \sigma_{F1 \wedge F2 \wedge \ldots \wedge Fn}(E)$$

5. Quy tắc giao hoán phép chọn và phép chiếu

$$\prod_X(\sigma_F(E)) \equiv \sigma_F(\prod_X(E))$$

Quy tắc này áp dụng khi F là điều kiện xác định được trên tập thuộc tính X. Tổng quát hơn ta có:

$$\prod_X(\sigma_F(E)) \equiv \prod_X(\sigma_F(\prod_{XY}(E)))$$

6. Quy tắc đối với phép chọn và phép tích Đề các: Ký hiệu: $E_1(U_1)$ có nghĩa là biểu thức E_1 xác định trên tập thuộc tính U_1; $F_1(U_1)$ có nghĩa là điều kiện chọn F_1 xác định trên tập thuộc tính U_1. Quy tắc biến đổi liên quan đến phép chọn và tích Đề các được phát biểu như sau:

$\sigma_F(E_1(U_1) \times E_2(U_2))$ tương đương với:

- $\sigma_{F1}(E_1) \times E_2$ trong trường hợp $F = F_1(U_1)$
- $\sigma_{F1}(E_1) \times \sigma_{F2}(E_2)$ trong trường hợp $F = F_1(U_1) \wedge F_2(U_2)$
- $\sigma_{F2}(\sigma_{F1}(E_1) \times E_2)$ trong trường hợp $F = F_1(U_1) \wedge F_2(U_1U_2)$

7. Quy tắc đối với phép chọn và phép hợp:

$$\sigma_F(E_1 \cup E_2) \equiv \sigma_F(E_1) \cup \sigma_F(E_2)$$

8. Quy tắc đối với phép chọn và phép trừ:

$$\sigma_F(E_1 - E_2) \equiv \sigma_F(E_1) - \sigma_F(E_2)$$

9. Quy tắc đối với phép chiếu và tích Đề các:

$$\prod_X(E_1(U_1) \times E_2(U_2)) \equiv \prod_Y(E_1) \times \prod_Z(E_2)$$

$$X = YZ, Y \subset U_1, Z \subset U_2$$

10. Quy tắc đối với phép chiếu và phép hợp:

$$\prod_X(E_1 \cup E_2) \equiv \prod_X(E_1) \cup \prod_X(E_2)$$

Ví dụ

Cho CSDL gồm các quan hệ:

S (sid, sname, size, city)

P (pid, pname, colour, weight, city)

SP (sid, pid, quantity)

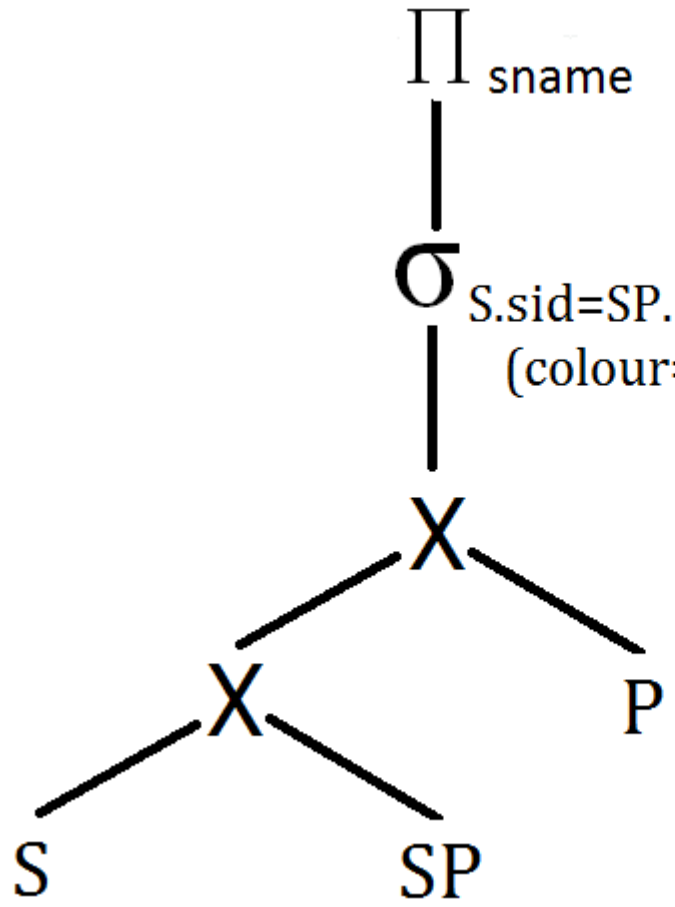
- Tìm tên hãng cung ứng ít nhất một mặt hàng màu đỏ hoặc màu xanh

  SELECT sname FROM S, P, SP

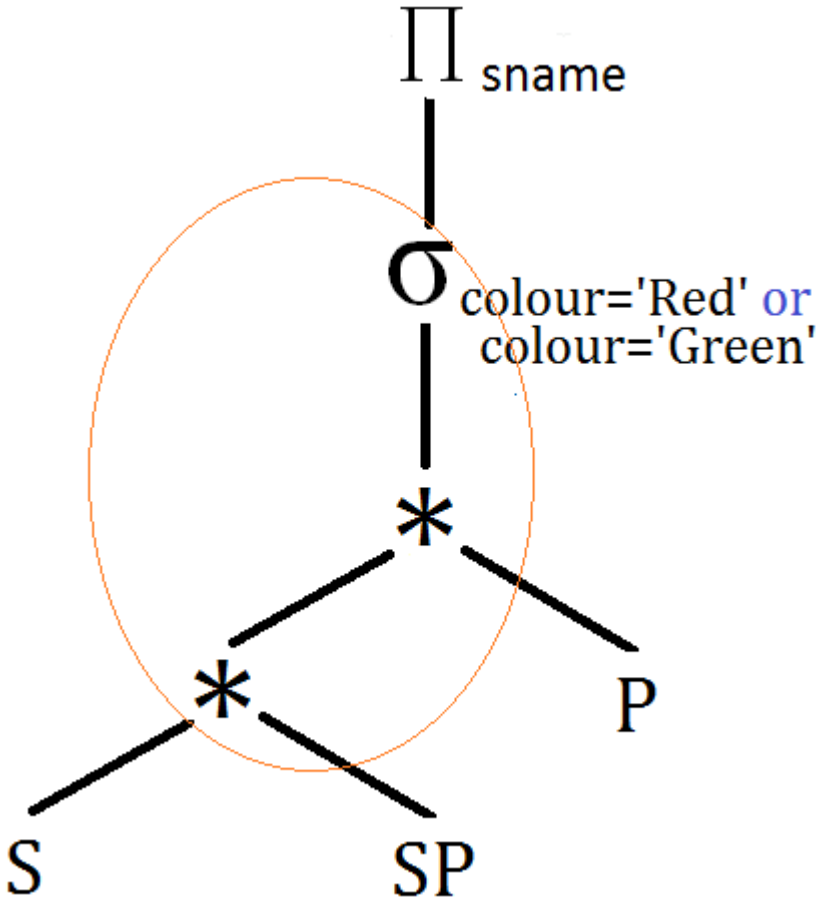
  WHERE S.sid = SP.sid AND P.pid = SP.pid AND (colour = 'Red' OR colour = 'Green');

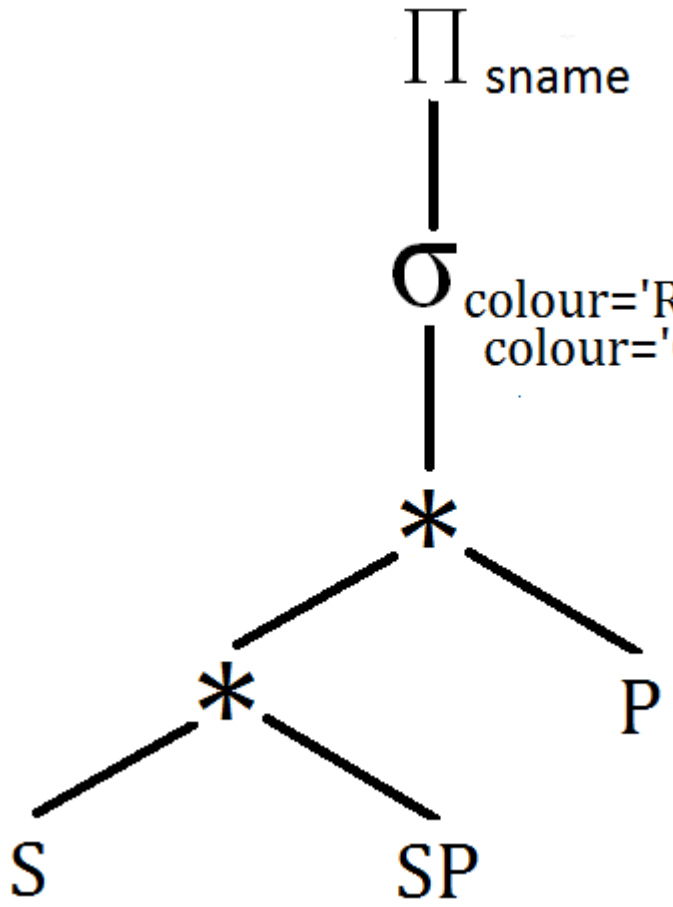
- Biểu thức đại số quan hệ tương đương với câu truy vấn trên là:

$$\Pi_{sname}(\sigma_{S.sid=SP.sid \wedge P.pid=SP.pid \wedge (colour='Red' \vee colour='Green')}(S \times SP \times P))$$

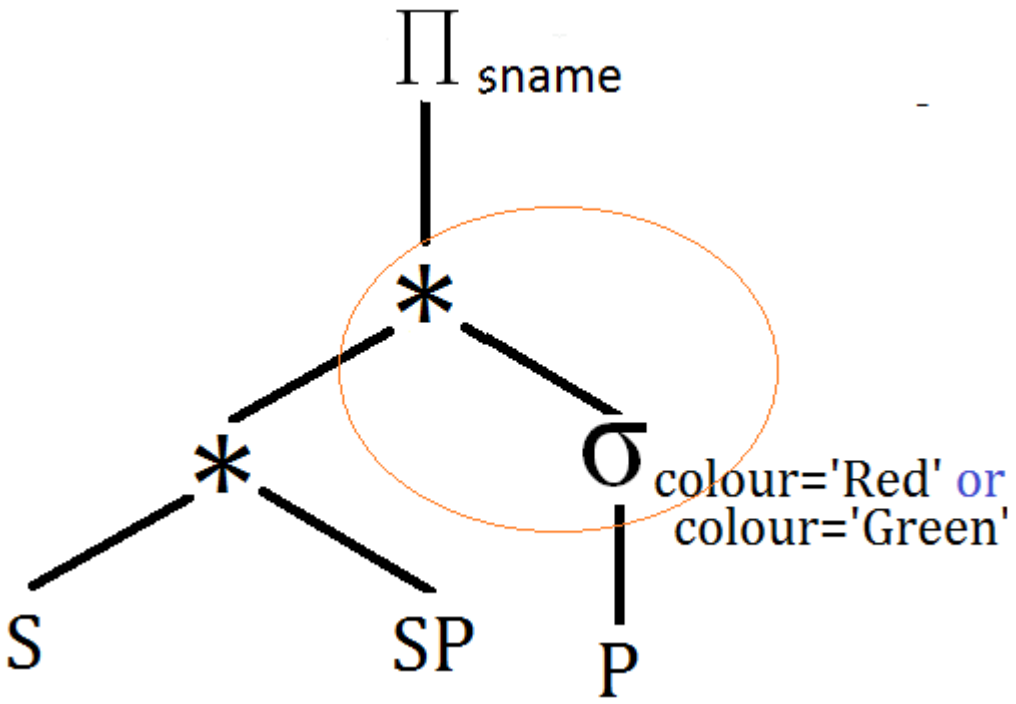


$\Pi_{\text{sname}} \sigma_{\text{S.sid=SP.sid and SP.pid=P.pid and (colour='Red' or colour='Green')}} (S \times SP \times P)$

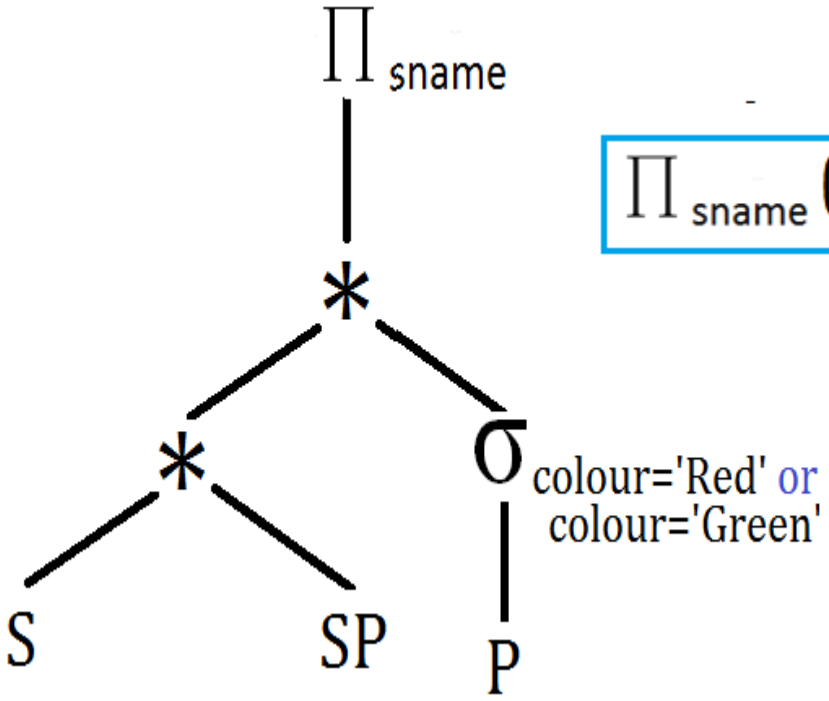




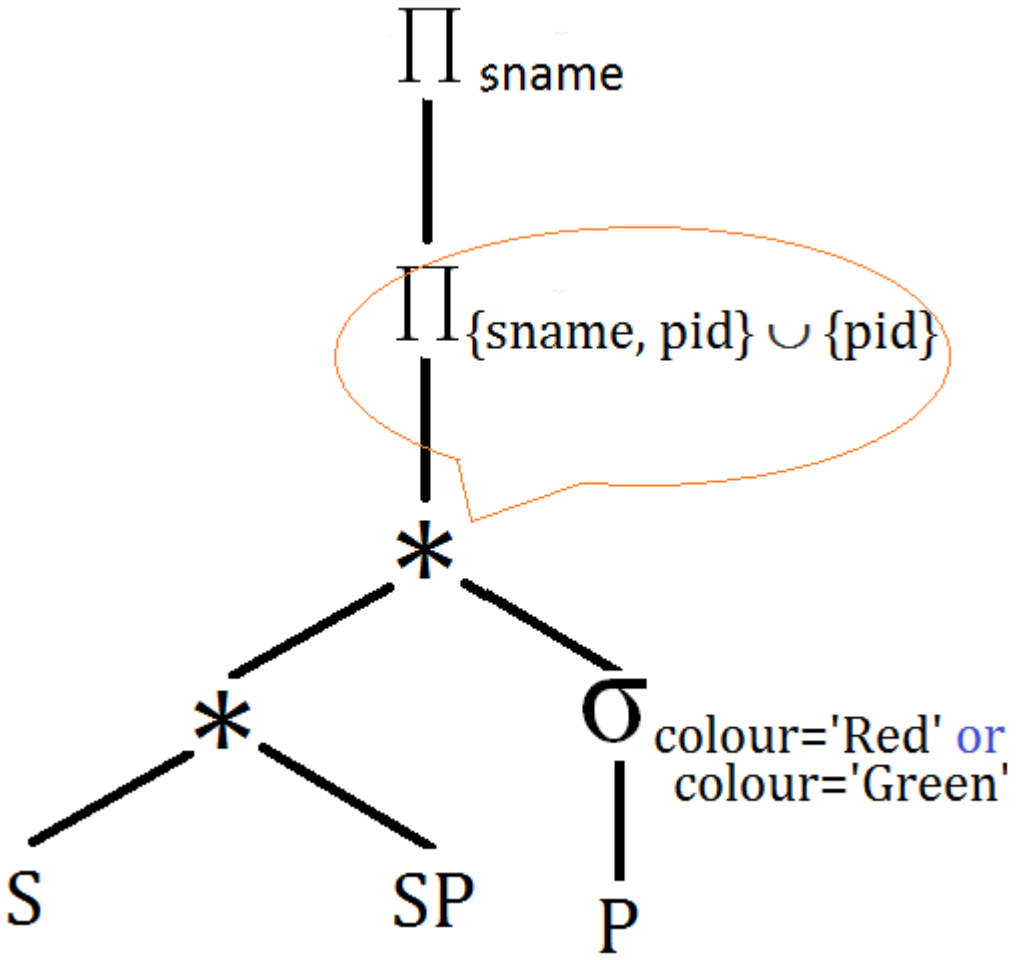
$$\prod_{sname} \sigma_{colour='Red' \text{ or } colour='Green'} (S * SP * P)$$



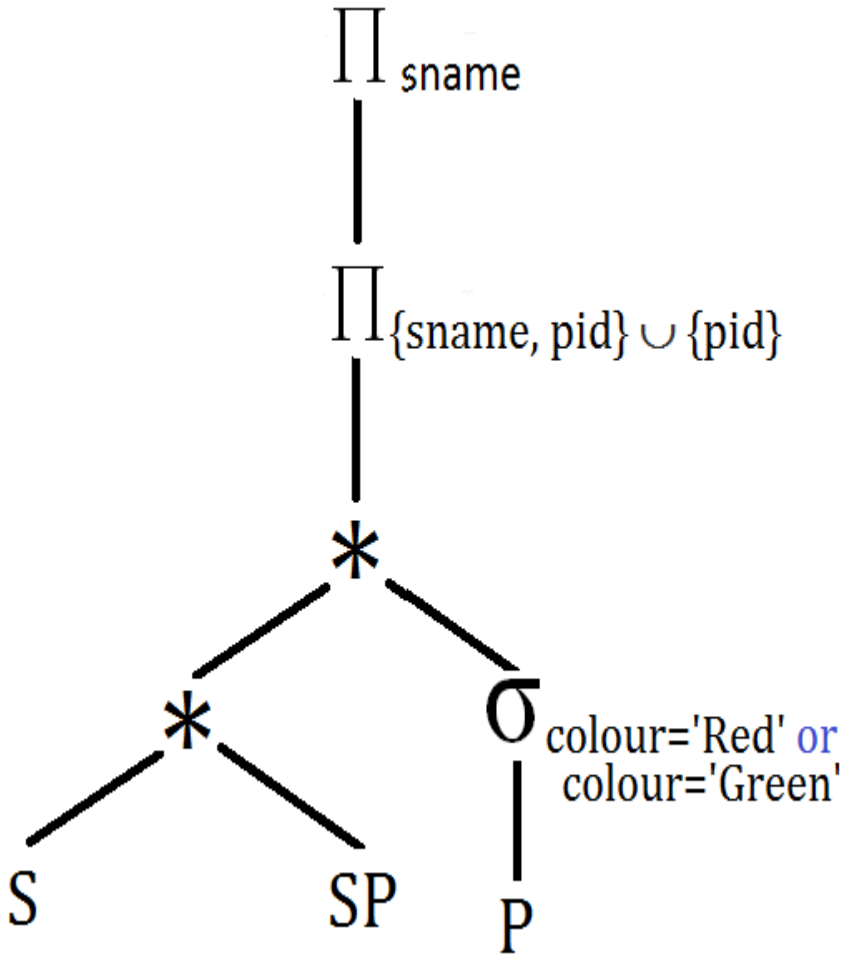
$$\prod_{sname} (S * SP * \sigma_{colour='Red' \text{ or } colour='Green'} (P))$$

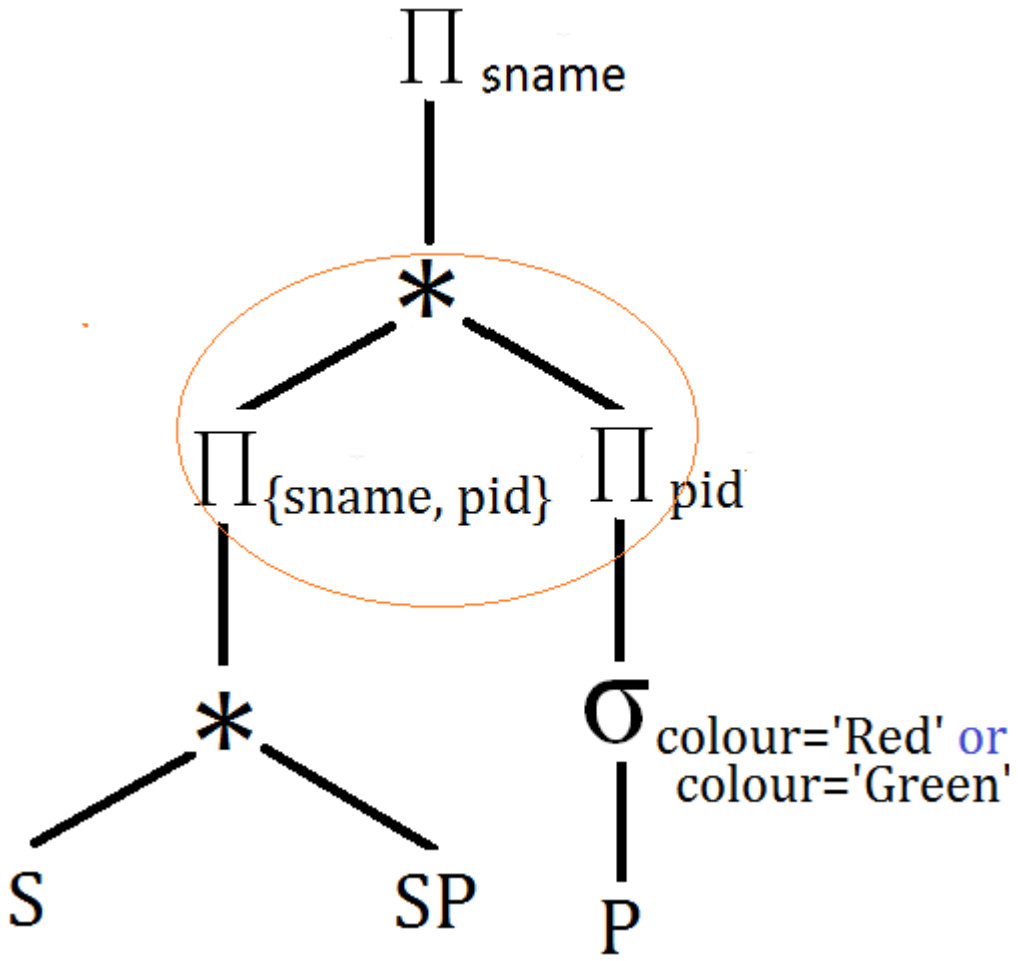


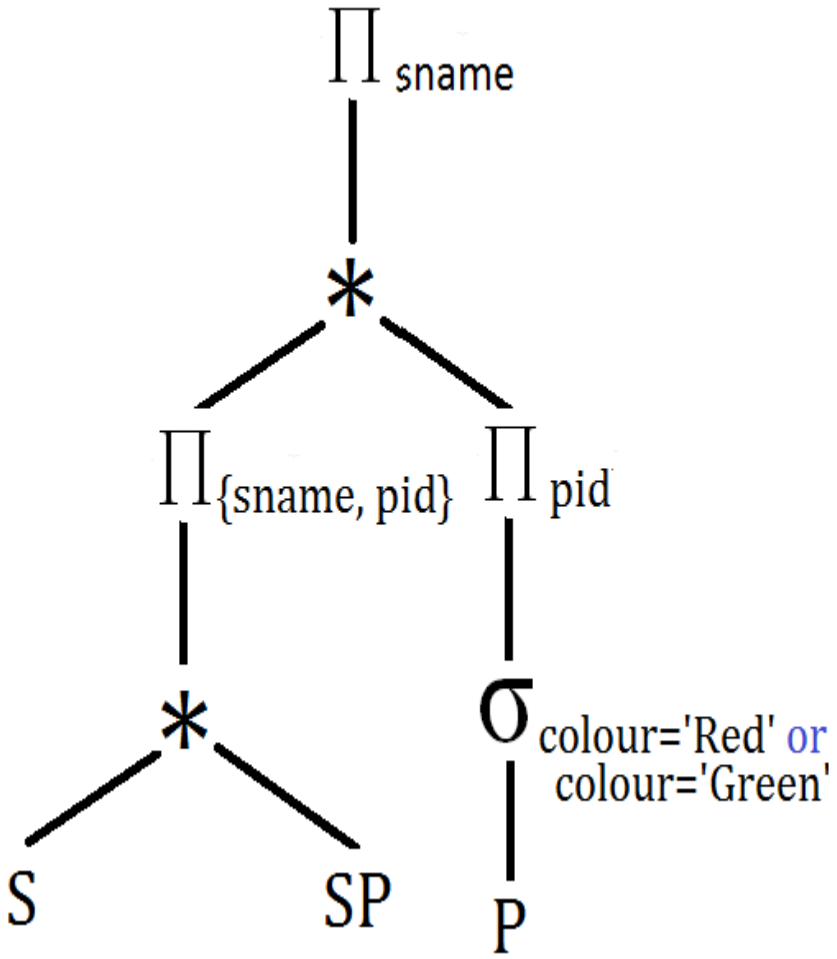
$$\prod_{sname}(S * SP * \sigma_{colour='Red' \text{ or } colour='Green'}(P))$$

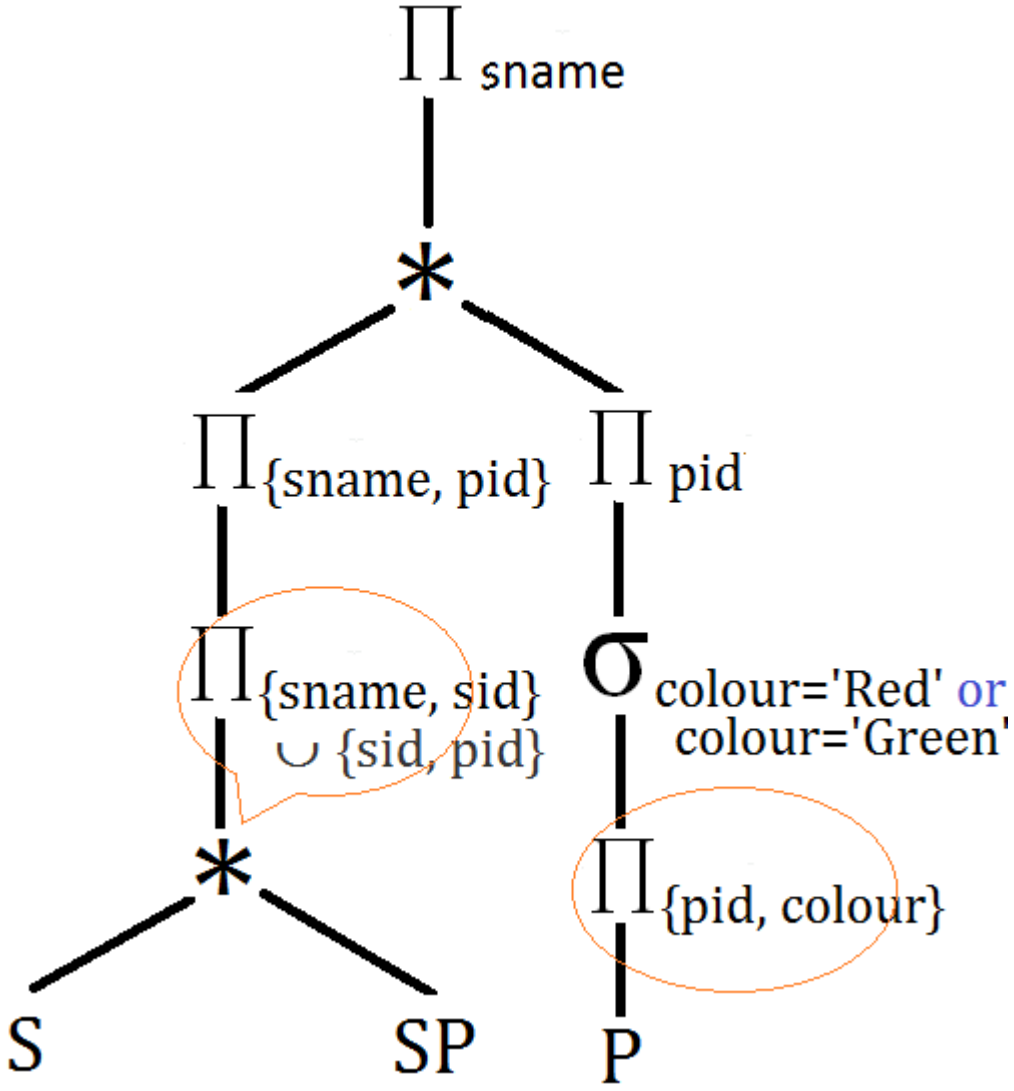


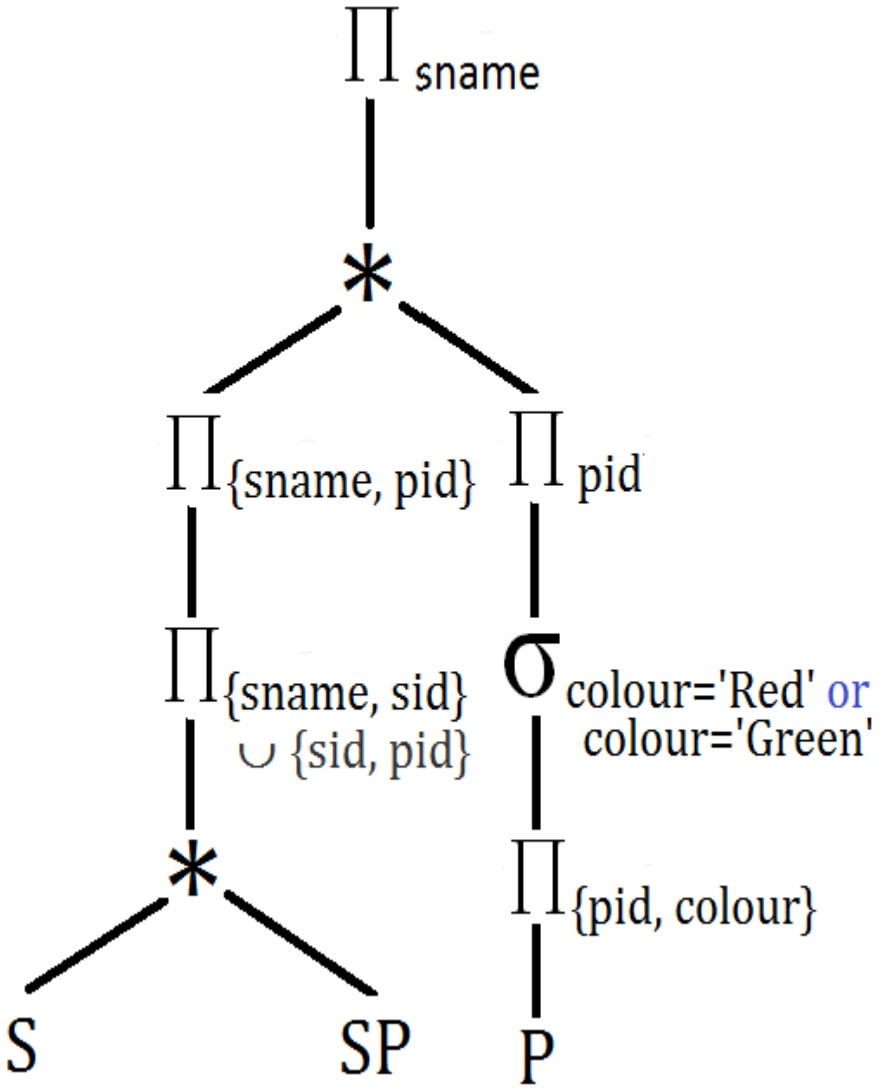
$$\prod_{\text{sname}} \prod_{\{\text{sname, pid}\}} (S * SP * \sigma_{\text{colour='Red' or colour='Green'}} (P))$$

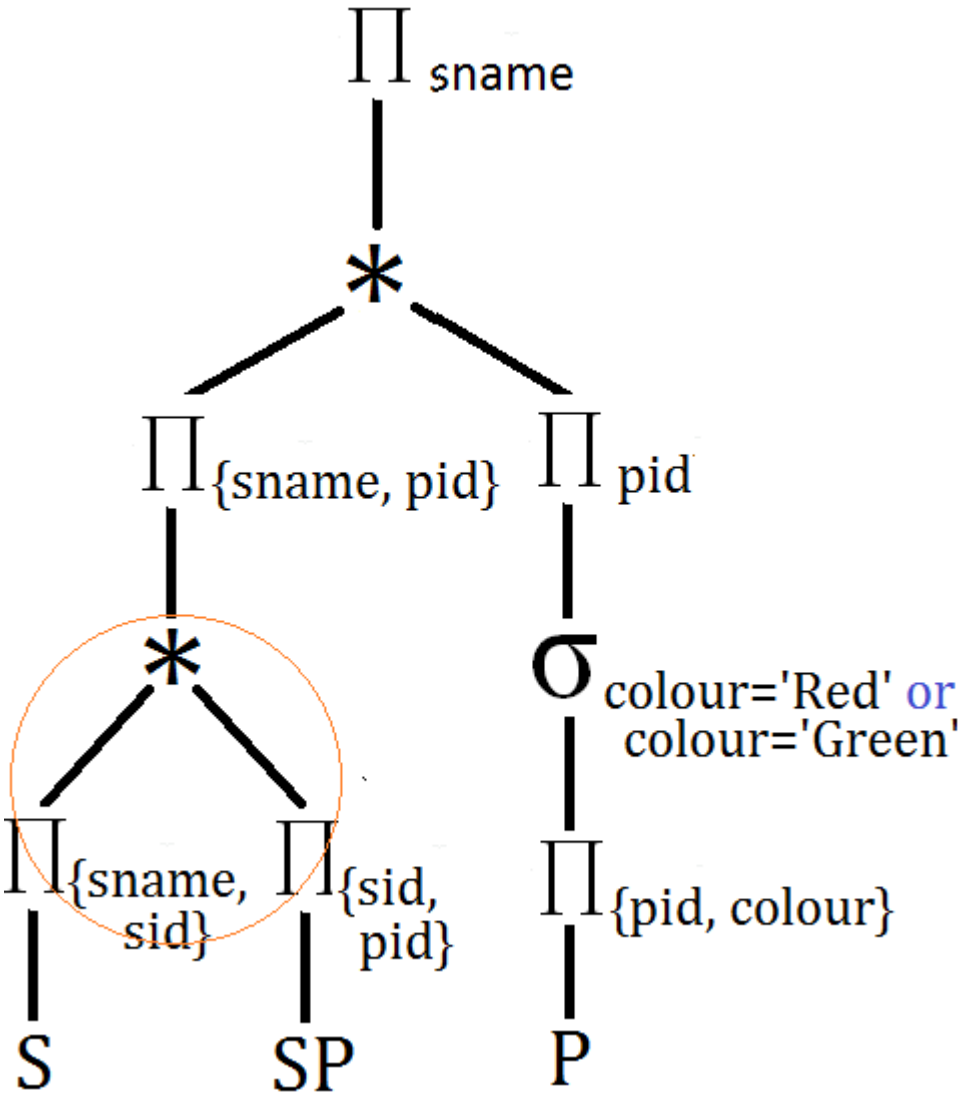





∏ sname
*
∏ {sname, pid}
∏ pid
*
σ colour='Red' or colour='Green'
S
SP
P


∏ sname
*
∏ {sname, pid}
∏ pid
∏ {sname, sid} ∪ {sid, pid}
σ colour='Red' or colour='Green'
*
∏ {pid, colour}
S
SP
P


∏ sname
*
∏ {sname, pid}
∏ {sname, sid} ∪ {sid, pid}
*
S
SP
∏ pid
σ colour='Red' or colour='Green'
∏ {pid, colour}
P

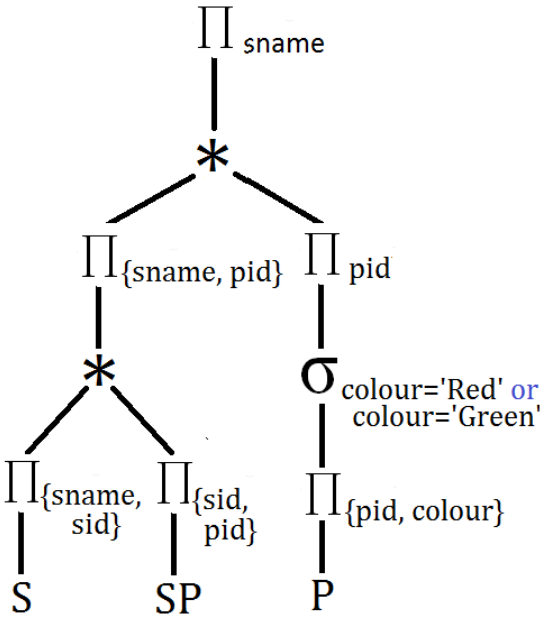

∏ sname
*
∏ {sname, pid}
*
∏ {sname, sid}
∏ {sid, pid}
S
SP
∏ pid
σ colour='Red' or colour='Green'
∏ {pid, colour}
P

Cho CSDL gồm các quan hệ:

S(sid, sname, size, city), P(pid, pname, colour, weight, city), SP (sid, pid, quantity)

Tìm tên hãng cung ứng ít nhất một mặt hàng màu đỏ hoặc màu xanh

$$\prod_{sname}\left(\prod_{\{sname,\ pid\}}\left(\prod_{\{sname,\ sid\}}(S)*\prod_{\{sid,\ pid\}}(SP)\right)*\right.$$
$$\left.*\prod_{pid}\left(\sigma_{colour='Red'\ or\ colour='Green'}\left(\prod_{\{pid,\ colour\}}(P)\right)\right)\right)$$




```
SELECT sname FROM
  (SELECT sname, pid FROM
     (SELECT sname, sid FROM S) AS S1,
     (SELECT sid, pid FROM SP) AS SP1
   WHERE S1.sid = SP1.sid    )   AS   SSP1,
  (SELECT pid FROM
     (SELECT pid, colour FROM P) AS  P1
   WHERE P1.colour = 'Red' or P1.colour = 'Green') AS P2
WHERE  SSP1.pid = P2.pid
```

```
SELECT sname FROM S WHERE sid IN
   (SELECT sid FROM SP WHERE pid IN
      (SELECT pid FROM P WHERE
         colour = ‘Red’ OR colour = ‘Green’ ) )
```

Kết luận

- Xử lý truy vấn:
  - Biên dịch câu truy vấn
  - Tối ưu hóa câu truy vấn
  - Thực hiện tìm kiếm dữ liệu
- Tối ưu hóa câu truy vấn:
  - Biến đổi tương đương
  - Đặc tả các phép toán của biểu thức
  - Đánh giá chi phí, chọn kế hoạch thực hiện

Chương 6 – An toàn và toàn vẹn dữ liệu

NỘI DUNG:

- Đặt vấn đề
- An toàn dữ liệu
- Toàn vẹn dữ liệu
- Giao dịch, điều khiển đồng thời

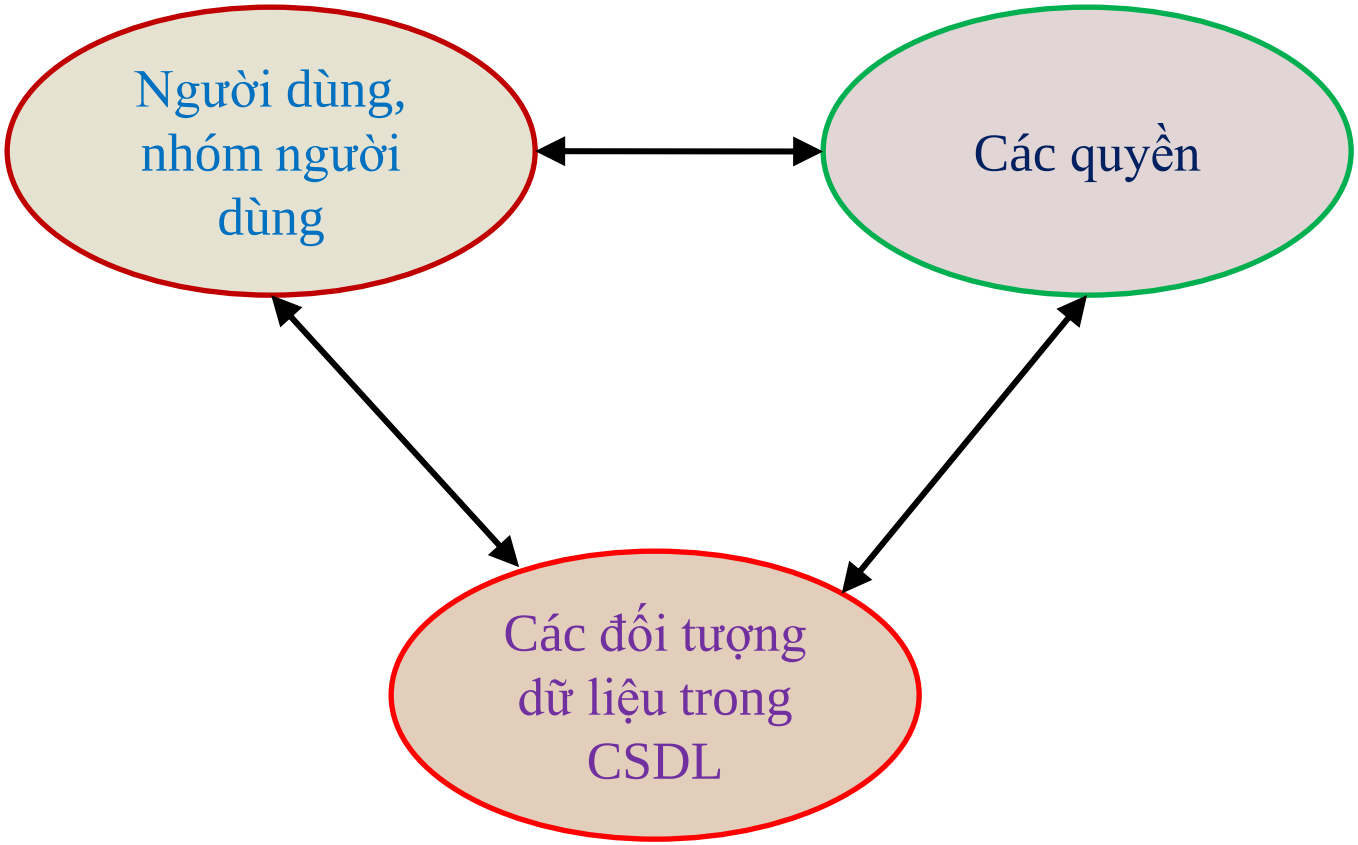
6.1. Đặt vấn đề

- Một số yêu cầu đối với thiết kế, cài đặt và quản trị CSDL:
  - Đảm bảo tính an toàn của dữ liệu
    - Tránh truy nhập không hợp lệ từ phía người dùng: phân quyền, xác minh và kiểm tra quyền hạn người sử dụng.
  - Đảm bảo tính đúng đắn của dữ liệu
    - Tránh sai sót khi cập nhật dữ liệu: định nghĩa và kiểm tra các ràng buộc dữ liệu.
    - Tránh sai sót trong quá trình thao tác với dữ liệu: kiểm tra tính toàn vẹn của các thao tác với dữ liệu.

6.2. An toàn dữ liệu

- **Định nghĩa:** Tính an toàn dữ liệu là sự bảo vệ dữ liệu trong CSDL chống lại những truy nhập, sửa đổi hay phá hủy bất hợp pháp.
- Người sử dụng hợp pháp là những người sử dụng được cấp phép, ủy quyền.
- Để đảm bảo tính an toàn cho CSDL, cần có một cơ chế để quản lý người dùng hợp lý.
- Những nhóm người dùng khác nhau trong hệ CSDL có quyền sử dụng khác nhau đối với các đối tượng dữ liệu trong CSDL.

Trục tam giác



6.2.1. Các quyền truy nhập của người sử dụng

- Đối với người khai thác CSDL:
  - Quyền đọc dữ liệu: được phép đọc một phần hay toàn bộ dữ liệu trong CSDL.
  - Quyền cập nhật dữ liệu: được phép sửa đổi một số giá trị nhưng không được xóa dữ liệu trong CSDL.
  - Quyền xóa dữ liệu: được phép xóa dữ liệu trong CSDL.
  - Quyền bổ sung dữ liệu: được phép thêm dữ liệu mới vào trong CSDL nhưng không được phép thay đổi dữ liệu
- Đối với người quản trị CSDL:
  - Quyền tạo chỉ dẫn trên các quan hệ trong CSDL
  - Quyền thay đổi sơ đồ cơ sở dữ liệu: thêm hay xóa các thuộc tính của các quan hệ trong CSDL
  - Quyền loại bỏ quan hệ trong CSDL
  - Quyền quản lý tài nguyên: được phép thêm các quan hệ mới vào CSDL

6.2.2. Người dùng

- Một người dùng cụ thể (user)
- Một nhóm người dùng (group)
- Một số hệ quản trị không phân biệt hai khái niệm trên, mà gọi chung là vai trò (role)

6.2.3. Các đối tượng dữ liệu

- Tables
- Views

Trách nhiệm của người quản trị hệ thống

- Để có thể phân biệt được người sử dụng trong hệ CSDL, người quản trị hệ thống phải có trách nhiệm:
  - Xác định các quyền cụ thể mà mỗi người sử dụng hay một nhóm người sử dụng được phép thực hiện, xác định vai trò và trách nhiệm của mỗi người sử dụng. Điều này được gọi chung là Phân quyền người sử dụng.
  - Cung cấp một phương tiện cho người sử dụng để hệ thống có thể nhận biết được người sử dụng đó hay còn gọi là Xác minh người sử dụng.

Xác minh người sử dụng

- Để xác minh được người sử dụng, người ta có thể dùng các kỹ thuật sau:
  - Kỹ thuật dùng tài khoản có tên và mật khẩu, mật khẩu cũng được bảo vệ bởi hệ thống.
  - Kỹ thuật sử dụng các hàm kiểm tra người sử dụng: Hệ thống đưa cho người sử dụng một số ngẫu nhiên x, người sử dụng dùng một hàm F tính nhẩm kết quả và đưa kết quả y = F(x) vào hệ thống. Trong lúc đó, hệ thống cũng tính toán và so sánh kết quả với y. Người sử dụng hợp pháp là người biết hàm biến đổi F và đưa vào giá trị y đúng.
  - Kỹ thuật dùng thẻ điện tử, thẻ thông minh.
  - Kỹ thuật sử dụng nhận dạng tiếng nói, vân tay v..v.

Kiểm tra quyền truy nhập của người sử dụng

- Mỗi người sử dụng sẽ có một bộ hồ sơ do người quản trị thiết lập và được hệ thống quản lý, trong hồ sơ đó sẽ có chi tiết về các thao tác người sử dụng được phép thực hiện:
  - Phân quyền người sử dụng: Người quản trị hệ thống phải có trách nhiệm xác định khung nhìn để kiểm soát xem mỗi người sử dụng chỉ được truy nhập phần dữ liệu nào trong CSDL và có được các quyền nào trong số các quyền đọc, thêm, xóa, sửa đổi.
  - Xác định và kiểm soát sự lưu chuyển dữ liệu: Hệ thống phải bảo trì danh sách các quyền một cách chặt chẽ vì người sử dụng có thể được quyền lan truyền các quyền cho người sử dụng khác.

Các câu lệnh an toàn dữ liệu trong SQL

- Câu lệnh tạo khung nhìn
- Câu lệnh phân quyền cho người sử dụng
- Câu lệnh thu hồi quyền của người sử dụng

Câu lệnh tạo khung nhìn

- CREATE VIEW <Tên khung nhìn> [(d/s cột)] AS <Câu truy vấn>
- Danh sách các cột trong khung nhìn là phần không bắt buộc. Trong trường hợp người sử dụng muốn đặt tên khác cho các cột xuất hiện trong khung nhìn thì người sử dụng có thể chỉ ra tên các cột, dữ liệu trên cột thì tương ứng với các cột trong mệnh đề SELECT của câu truy vấn.

VÍ DỤ

- Cho cơ sở dữ liệu gồm 2 quan hệ:

Nhânviên(Id, Họtên, Địachỉ, Lương, NămBD, Đánhgiá, PhòngCT)

Phòng(PId, Tên, ĐC, Điệnthoại, Trưởngphòng)

- Câu lệnh tạo khung nhìn cho một nhân viên của phòng 'Khoa học' có thể được định nghĩa như sau:

```
CREATE VIEW NVKH(HọtênNhânviên, Địachỉliênlạc) AS
SELECT Họtên, Địachỉ  FROM Nhânviên WHERE PhòngCT IN
  (SELECT PId FROM Phòng WHERE Tên ='Khoa học')
```

Câu lệnh phân quyền cho NSD

- GRANT <D/s thao tác> ON <Đối tượng>
  TO <D/s người dùng> [WITH GRANT OPTION]
- <D/s thao tác>: có thể bao gồm một hay nhiều thao tác được liệt kê dưới đây:
  - Insert: chèn dữ liệu vào trong CSDL có sẵn nhưng không được thay đổi bất kỳ mục dữ liệu nào trong CSDL
  - Update: sửa đổi dữ liệu nhưng không được xóa dữ liệu
  - Delete: xóa dữ liệu trong CSDL
  - Select : tìm kiếm
  - Create: tạo lập các quan hệ mới
  - Alter: Thay đổi cấu trúc của quan hệ
  - Drop: Loại bỏ quan hệ
  - Read/Write: Đọc và Ghi
- <Đối tượng>: bảng hoặc khung nhìn

- <D/s người dùng>: Một người hay một nhóm hay một danh sách người sử dụng. Từ khóa public được dùng thay thế cho mọi người sử dụng
- [With Grant Option] Nếu dùng từ khóa này trong câu lệnh phân quyền thì người dùng xuất hiện trong <D/s người dùng> có quyền được lan truyền các quyền vừa được tuyên bố cho những người dùng khác

VÍ DỤ

- Trao quyền đọc, ghi, tìm kiếm, sửa đổi dữ liệu cho nhân viên tên 'Hoa' của phòng 'Khoa học' trên khung nhìn vừa tạo lập trong phần trước

```
GRANT read, write, select, update ON NVKH TO Hoa;
```

- Trao quyền cho trưởng phòng Khoa học – ông HungNC

```
GRANT read, write, select, update, delete ON NVKH TO HungNC
WITH GRANT OPTION;
```

Câu lệnh thu hồi quyền của NSD

- REVOKE <D/s thao tác> ON <Đối tượng> FROM <D/s người dùng> [RESTRICT/CASCADE]
- <D/s thao tác>, <Đối tượng>, <D/s người dùng> giống như đối với câu lệnh GRANT.
- Phần [RESTRICT/CASCADE] là chỉ ra cơ chế thu hồi với các quyền đã được người dùng trong <D/s người dùng> lan truyền
- Nếu Restrict thì có nghĩa là chỉ hủy bỏ quyền của những người có trong danh sách, quyền đã được lan truyền cho người khác không bị thu hồi.
- Nếu dùng Cascade thì hủy bỏ quyền của người trong <D/s người dùng>, đồng thời kéo theo hủy bỏ quyền mà người dùng đó đã luân chuyển cho những người khác.
- Ví dụ:

REVOKE update, delete ON NVKH FROM HungNC CASCADE

6.3. Toàn vẹn dữ liệu

- Định nghĩa: Tính toàn vẹn dữ liệu là sự bảo vệ dữ liệu trong CSDL chống lại những sự sửa đổi, phá hủy vô căn cứ để đảm bảo tính đúng đắn và chính xác của dữ liệu.
- Các thao tác có thể ảnh hưởng đến tính đúng đắn của CSDL là thêm, xóa, sửa đổi.
- Để đảm bảo tính toàn vẹn dữ liệu, cần phải chỉ ra và duy trì những ràng buộc toàn vẹn liên kết với mỗi quan hệ. Các ràng buộc toàn vẹn cung cấp một phương tiện để đảm bảo rằng các thao tác được thực hiện bởi những người sử dụng hợp pháp không làm mất đi tính đúng đắn của CSDL.
- Trong hệ thống đa người dùng, để đảm bảo được toàn vẹn dữ liệu, hệ thống còn phải có được một trình điều khiển tương tranh để tránh đụng độ giữa các thao tác được đưa ra bởi những người sử dụng khác nhau tại cùng một thời điểm

Các ràng buộc toàn vẹn trong SQL

- Các ràng buộc về khóa chính, khóa ngoài, kiểm tra miền giá trị sử dụng Check đã được đề cập đến khi nói về câu lệnh tạo bảng trong CSDL.
- Các khẳng định (assertion): Là một vị từ biểu thị một điều kiện mà CSDL phải luôn luôn thỏa mãn.

  Các khẳng định được tạo ra bằng câu lệnh:

  CREATE ASSERTION <Tên khẳng định> CHECK <Vị từ>
- Các kích hoạt (trigger): Là một thủ tục lưu trữ hệ thống (stored procedure) đặc biệt, được thực thi một cách tự động khi có sự kiện gây biến đổi dữ liệu như Update, Insert hay Delete

  Được dùng để đảm bảo toàn vẹn dữ liệu hay thực hiện các quy tắc nghiệp vụ nào đó.

Ví dụ về khẳng định

- Số lượng mặt hàng được cung cấp bởi các hãng có số nhân viên < 50 phải nhỏ hơn 100:

```
CREATE ASSERTION KĐSốlượng CHECK NOT EXISTS
(SELECT * FROM S WHERE numofemps < 50 AND sid
IN (SELECT sid FROM SP WHERE quantity >= 100))
```

- Lương của nhân viên không được cao hơn lương người quản lý phòng ban của nhân viên đó.

```
CREATE ASSERTION Salary_Constraint
CHECK (NOT EXISTS
        (SELECT * FROM Employee E,
        Manager M, Department D
        WHERE E.Salary>M.Salary AND
E.Dno=D.Number AND D.MgrSSN=M.SSN))
```

Sử dụng trigger

- Khi nào sử dụng trigger?
  - khi các biện pháp đảm bảo toàn vẹn dữ liệu khác như Constraint không thể thỏa mãn yêu cầu của ứng dụng
    - Constraint thuộc loại toàn vẹn dữ liệu khai báo: kiểm tra dữ liệu trước khi cho phép nhận vào bảng
    - Trigger thuộc loại toàn vẹn dữ liệu thủ tục nên việc Insert, Update, Delete xảy ra rồi mới kích hoạt trigger.
  - Đôi khi, do nhu cầu thay đổi dây chuyền, có thể sử dụng trigger

Đặc điểm của trigger

- một trigger có thể làm nhiều công việc, có thể được kích hoạt bởi nhiều sự kiện
- trigger không thể được tạo ra trên bảng tạm hoặc bảng hệ thống
- trigger chỉ có thể được kích hoạt tự động bởi các sự kiện mà không thể chạy thủ công được.
- có thể áp dụng trigger cho view
- khi trigger được kích hoạt
  - dữ liệu mới được insert sẽ được chứa trong bảng "inserted"
  - dữ liệu mới được delete sẽ được chứa trong bảng "deleted"
  - đây là hai bảng tạm nằm trên bộ nhớ, và chỉ có giá trị bên trong trigger

Trigger

- ECA: Sự kiện – Điều kiện – Hành động
- Sự kiện (event): có sự cập nhật dữ liệu (insert, update, delete)
- Điều kiện (condition): biểu thức SQL có giá trị Boolean
- Hành động (Action): câu lệnh SQL

Ví dụ về trigger

- Nhânviên(ID, Họtên, Lương, Địachỉ, Ngườiquảnlý)
- Một nhân viên bao giờ cũng có lương ít hơn lương người trưởng phòng, điều kiện này phải được kiểm tra khi thêm bộ dữ liệu.

```
CREATE TRIGGER ThemNV INSERT ON Nhânviên
IF Nhânviên.Lương > (SELECT E.Lương FROM
   Nhânviên AS E WHERE E.ID =
   Nhânviên.Ngườiquảnlý)
THEN ABORT;
```

Ví dụ

- **Constraint:** Khi thêm một sinh viên, thì số sinh viên trong lớp sẽ tăng lên

student(**student_id**, first_name,last_name, dob, gender, address, note, email, *clazz_id*)

clazz(**clazz_id**, name, *lecturer_id*, *monitor_id*, number_students)

```
CREATE TRIGGER clazz_changes_tg
AFTER INSERT ON student
REFERENCING NEW ROW AS nnn
FOR EACH ROW
WHEN (nnn.clazz_id IS NOT NULL)
BEGIN
        update clazz
        set number_students = number_students + 1
        where clazz_id = nnn.clazz_id;
END;
```

Event

Condition

Action

Cú pháp

- Tạo một Trigger

```
CREATE [OR REPLACE] TRIGGER <trigger_name>
      {BEFORE | AFTER | INSTEAD OF }
      {INSERT | DELETE | UPDATE [OF <attribute_name>]}
      ON <table_name>
      REFERENCING {NEW | OLD} {ROW | TABLE} AS <name>
      [FOR EACH ROW ]
      [WHEN (<condition>) ]
      BEGIN
            <trigger body goes here >
      END;
```

- Xóa Trigger

```
DROP TRIGGER <trigger_name>;
```

- AFTER, BEFORE: used for tables / views
- INSTEAD OF: used only for views
- UPDATE OF <colums>: update on a particular column
- Row-level trigger: Indicated by option FOR EACH ROW

6.4. Giao dịch và điều khiển đồng thời

- Trong hệ CSDL đa người dùng, hệ thống cần đưa ra giải pháp chống đụng độ giữa các giao dịch (một dãy các thao tác) được đưa ra bởi những người dùng khác nhau để tránh việc một đối tượng dữ liệu nào đó bị làm mất tính đúng đắn trong quá trình cập nhật.

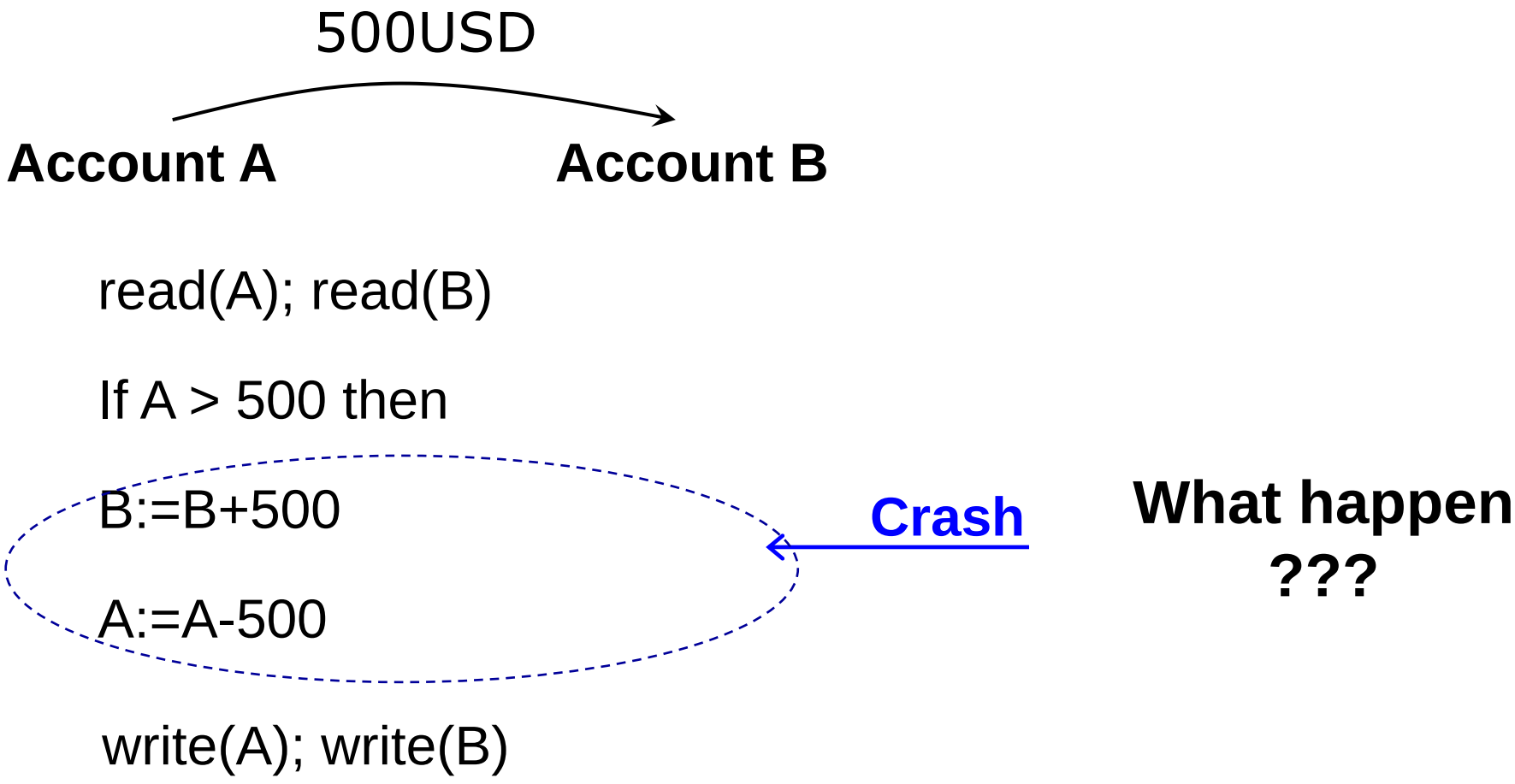
6.4.1. Giao dịch

- Một giao dịch hình thành nên một đơn vị công việc trong một DBMS đối với một CSDL, được coi là cố kết và tin cậy độc lập với các giao dịch khác.
- Giao dịch có nhiều bước, và phải được thực hiện một cách trọn vẹn.
- Trạng thái trung gian giữa các bước là ẩn đối với các giao dịch khác.
- Nếu có sự cố mà giao dịch không thể hoàn thành, thì tất cả các bước không ảnh hưởng lên CSDL.

Ví dụ về giao dịch

- Một CSDL ngân hàng chứa thông tin tài khoản của các khách hàng và tài khoản quỹ của các chi nhánh.
- Giả sử, thực hiện giao dịch chuyển 500 đô-la từ tài khoản của Alice sang tài khoản của Bob:

```
UPDATE accounts SET balance = balance - 500.00
    WHERE name = 'Alice';
UPDATE branches SET balance = balance - 500.00
    WHERE name = (SELECT branch_name FROM accounts
                  WHERE name = 'Alice');
UPDATE accounts SET balance = balance + 500.00
    WHERE name = 'Bob';
UPDATE branches SET balance = balance + 500.00
    WHERE name = (SELECT branch_name FROM accounts
                  WHERE name = 'Bob');
```

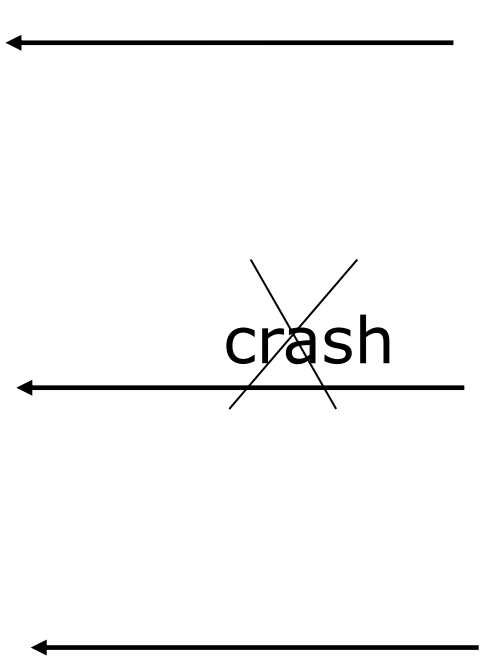

500USD
Account A
Account B
read(A); read(B)
If A > 500 then
B:=B+500
A:=A-500
write(A); write(B)
Crash
What happen ???

6.4.2. Các tính chất của một giao dịch

- Tính nguyên tố (Atomicy):
  - Hoặc là tất cả các thao tác được thực hiện hoặc là không thao tác nào được thực hiện
  - Đảm bảo tính không thể chia cắt, không thể rút gọn
- Tính nhất quán (Consistency): Dữ liệu nhất quán sau khi giao dịch thực hiện.
- Tính cách ly (Isolation): xác định cách mà những thay đổi bởi một thao tác là ẩn với các thao tác đồng thời khác.
- Tính bền vững (Durability):
  - Đảm bảo giao dịch đã được xác nhận sẽ tồn tại vĩnh cửu.
  - ví dụ, một vị trí chỗ ngồi trên máy bay đã được đặt chỗ thì vị trí đó sẽ vẫn trong trạng thái đã đặt chỗ cho dù hệ thống có vấn đề.
- Tất cả 4 tính chất trên, gọi tắt là ACID

Tính nguyên tố

```
T: Read(A,t1);
   If t1 > 500 {
        Read(B,t2);
        t2:=t2+500;
        Write(B,t2);
        t1:=t1-500;
        Write(A,t1);
   }
```



Tính nhất quán

```
T: Read(A,t1);
   If t1 > 500 {
        Read(B,t2);
        t2:=t2+500;
        Write(B,t2);
        t1:=t1-500;
        Write(A,t1);
   }
```

⟵ A+B = C

⟵ A+B = C

Tính cách ly

A= 5000, B= 3000

```
T: Read(A,t1);
   If t1 > 500 {
        Read(B,t2);
        t2:=t2+500;
        Write(B,t2);
        t1:=t1-500;
        Write(A,t1);
   }
```

← *T': A+B (= 5000+3500)*

← *(A+B = 4500+3500)*

Tính bền vững

A= 5000, B= 3000

```
T: Read(A,t1);
   If t1 > 500 {
        Read(B,t2);
        t2:=t2+500;
        Write(B,t2);
        t1:=t1-500;
        Write(A,t1);
   }
```

crash

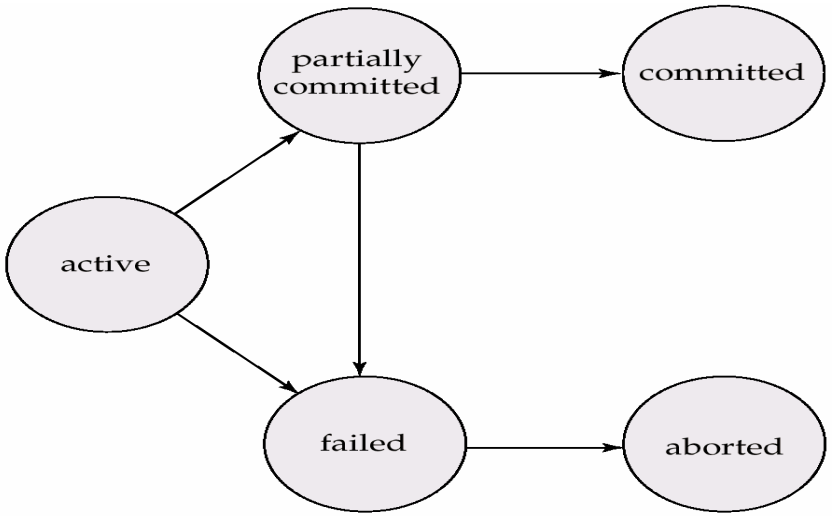
A= 4500, B=3500

- Các thao tác hình thành nên một giao dịch có thể được nhúng trong một chương trình ứng dụng hoặc được xác định bởi ngôn ngữ cấp cao như SQL.
- Để biết một giao dịch diễn ra, cần xác định tường minh câu lệnh bắt đầu (begin transaction) và kết thúc (end transaction) của giao dịch trong chương trình ứng dụng.
- Giao dịch chỉ đọc (read-only)
  - không cập nhật dữ liệu, mà chỉ lấy ra dữ liệu.
- Để đơn giản, giả sử các thao tác truy cập CSDL trong một giao dịch gồm có:
  - read_item(X): đọc 1 khoản mục CSDL tên là X vào một biến trong chương trình cũng tên là X.
  - write_item(X): ghi giá trị của biến X trong chương trình vào khoản mục CSDL cũng tên là X.

- Đơn vị cơ bản của việc chuyển dữ liệu từ đĩa vào bộ nhớ chính là khối (block).
- Thực hiện lệnh read_item(X) bao gồm các bước sau:
  - Tìm địa chỉ của khối trên đĩa mà chứa khoản mục X.
  - Sao chép khối đó vào bộ đệm trên bộ nhớ chính (nếu trên bộ đệm chưa có).
  - Sao chép khoản mục X từ bộ đệm vào biến chương trình X.
- Thực hiện lệnh write_item(X) bao gồm các bước sau:
  - Tìm địa chỉ khối trên đĩa mà nó chứa khoản mục X.
  - Sao chép khối đó vào bộ đệm trong bộ nhớ chính (nếu bộ đệm chưa có).
  - Sao chép khoản mục X từ biến chương trình tên là X vào vị trí chính xác trên bộ đệm.
  - Lưu trữ khối đã cập nhật này từ bộ đệm lên đĩa (có thể ngay tức thì hoặc lưu trữ sau).

Quản lý giao dịch

- Các trạng thái: Active, Failed, Aborted, Committed, Partially committed



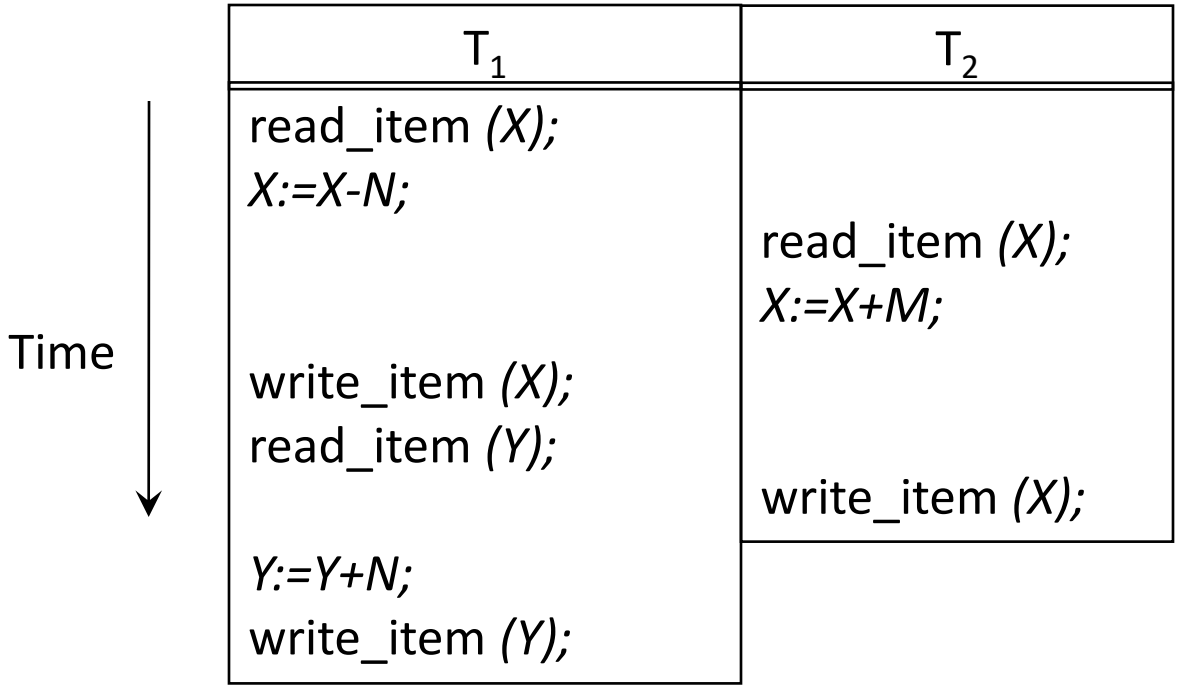
- Lệnh (SQL): Active, Begin Trans, Commit(), Abort(), Savepoint Save(), Rollback (savepoint) (savepoint = 0 ==> Abort)
  (commit để lưu, rollback quay lại trạng thái trước)

6.4.3. Tại sao lại phải điều khiển đồng thời?

- Các vấn đề
  - The Lost Update
  - The Temporary Update (Dirty Read)
  - The Incorrect Summary
  - The Unrepeatable Read

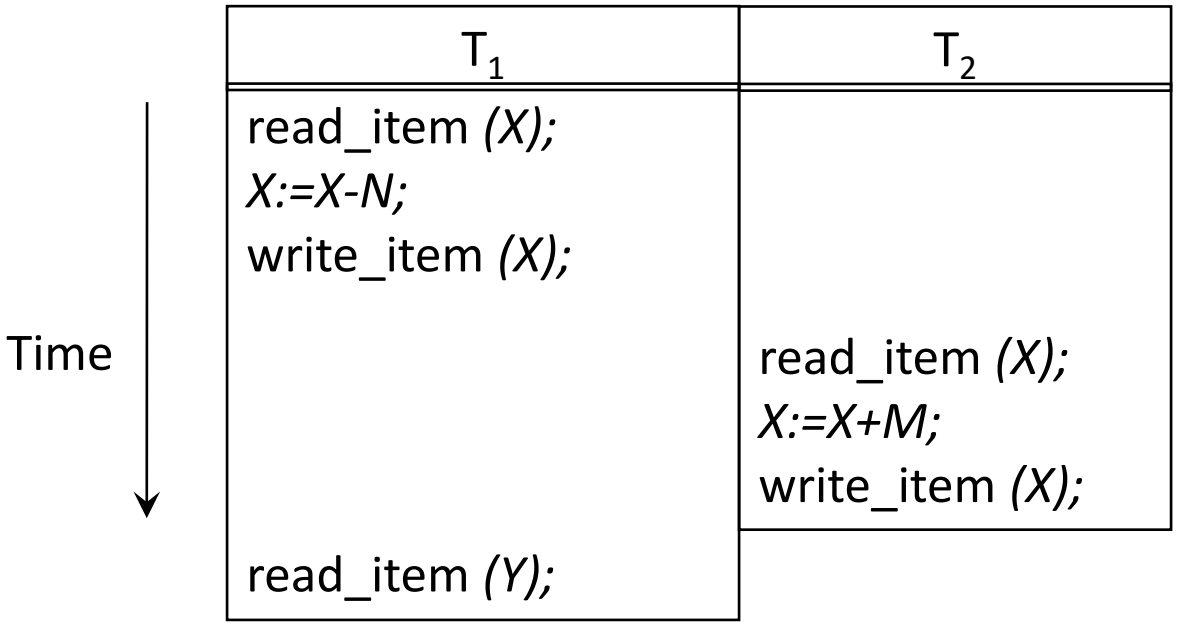
The lost update

- Hiện tượng này xuất hiện khi hai giao dịch truy cập vào cùng các khoản mục dữ liệu nhưng thao tác của hai giao dịch lại xen kẽ, làm cho giá trị của các khoản mục dữ liệu không còn đúng nữa.



The Temporary Update

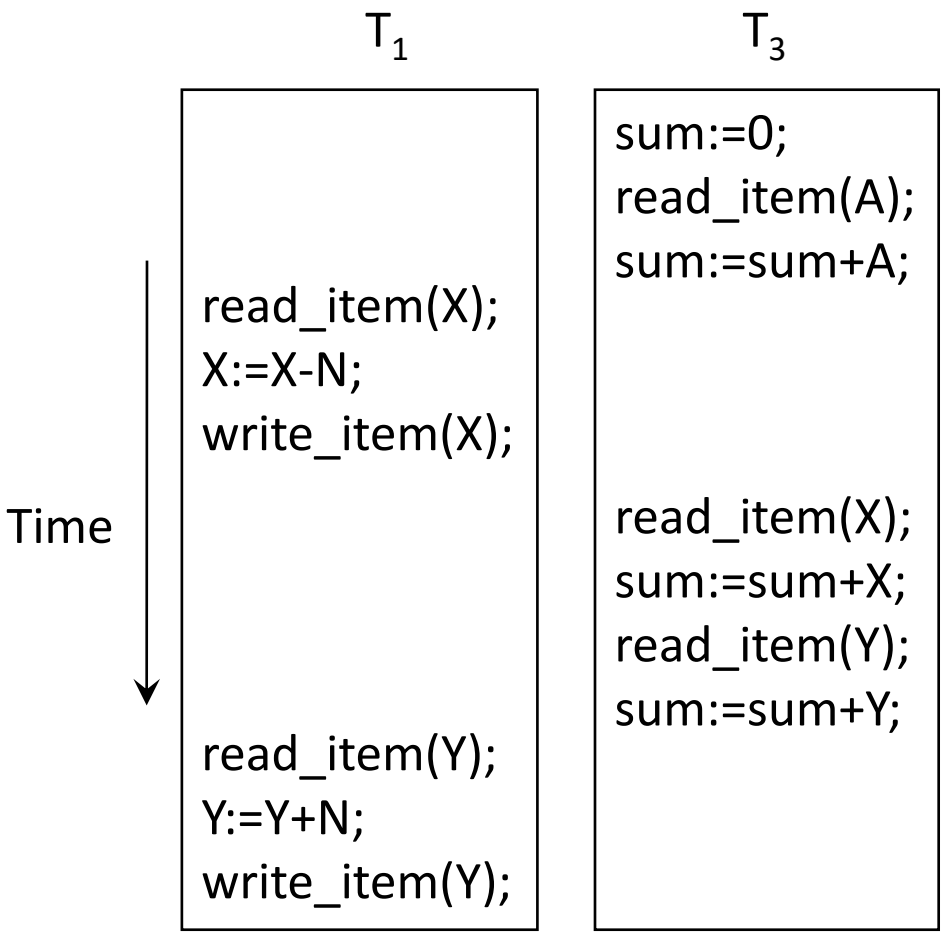
- Xuất hiện khi một giao dịch cập nhật một khoản mục dữ liệu, nhưng sau đó không hoàn thành các bước tiếp theo => giao dịch không trọn vẹn nên phải roll-back giá trị đã cập nhật về giá trị cũ.
- Khoản mục dữ liệu đã được cập nhật kia lại được sử dụng bởi một giao dịch khác trước khi nó được roll-back về giá trị cũ



Giao dịch *T1 fails và phải thay đổi giá trị của X về giá trị cũ của nó; trong khi đó T2* lại đọc giá trị không đúng tạm thời của X.

The Incorrect Summary

- Một giao dịch tính hàm tích lũy trên các bản ghi đang bị cập nhật bởi một giao dịch khác => hàm tích lũy này có thể tính toán dựa trên một số giá trị đã cập nhật và một số giá trị chưa cập nhật.



T3 đọc X sau khi X đã trừ N và đọc Y trước khi Y cộng N

The Unrepeatable Read

- Một giao dịch T đọc một khoản mục dữ liệu hai lần. Khoản mục dữ liệu này bị một giao dịch khác thay đổi giữa hai lần đọc đó.
- Do đó, T nhận các giá trị khác nhau cho hai lần đọc cùng một khoản mục.

Điều khiển đồng thời

- Mục tiêu: Các giao dịch được thực hiện đồng thời mà không vi phạm tính toàn vẹn dữ liệu; Các giao dịch thành công được lưu lại, các giao dịch bị hủy bỏ không còn trong CSDL
- Lịch trình (schedule) của một tập giao dịch là một thứ tự tuyến tính các hành động, ví dụ, R1(X) R2(X) W1(X) W2(X)
- Lịch trình tuần tự có các bước của mỗi giao dịch diễn ra nối tiếp

T_0	T_1
read(A)	
A := A - 50	
write(A)	
read(B)	
B := B + 50	
write(B)	
	read(A)
	temp := A * 0.1
	A := A -temp
	write(A)
	read(B)
	B := B + temp
	write(B)

(1)

T_0	T_1
	read(A)
	temp := A * 0.1
	A := A -temp
	write(A)
	read(B)
	B := B + temp
	write(B)
read(A)	
A := A - 50	
write(A)	
read(B)	
B := B + 50	
write(B)	

(2)

T_0	T_1
read(A)	
A := A - 50	
	read(A)
	temp := A * 0.1
	A := A -temp
	write(A)
	read(B)
write(A)	
read(B)	
B := B + 50	
write(B)	
	B := B + temp
	write(B)

(3)

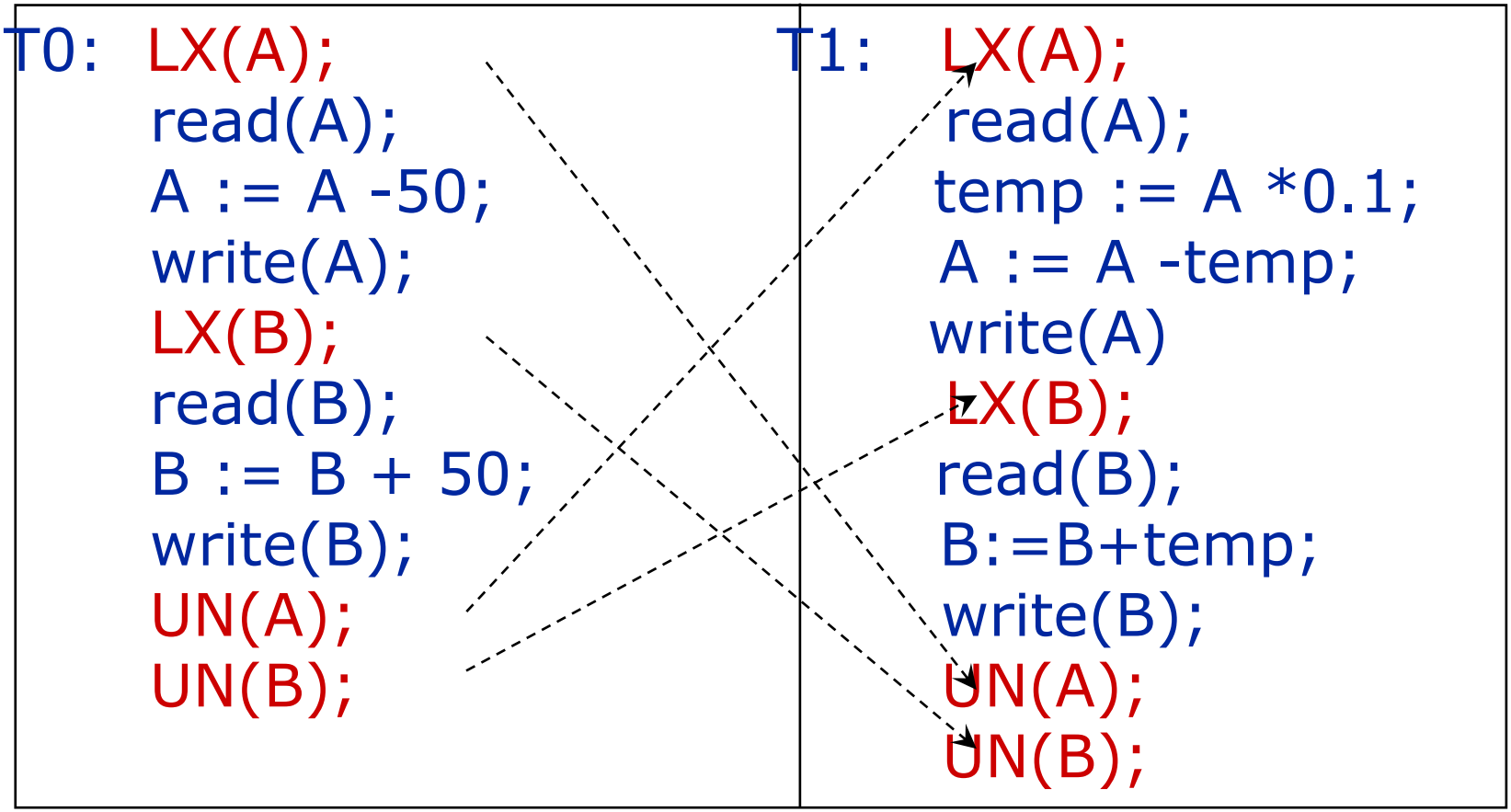
6.4.4. Các kỹ thuật điều khiển đồng thời

- **Kỹ thuật dùng khóa**: Khi một giao dịch cần dữ liệu nào thì xin hệ điều hành một khóa trên phần dữ liệu đó, các giao dịch khác phải đợi đến khi giải phóng khóa mới được sử dụng phần dữ liệu đó. Có thể người ta sử dụng các loại khóa khác nhau ví dụ như khóa đọc – cho phép nhiều giao dịch đọc cùng một lúc, khóa ghi – chỉ một giao dịch có được tại một thời điểm.
- **Kỹ thuật gán nhãn thời gian**: Mỗi giao dịch được gán một nhãn T theo thời gian, giao dịch nào cần được ưu tiên thì được gán nhãn thời gian nhỏ hơn và được thực hiện trước. Kỹ thuật này giúp đưa yêu cầu đồng thời về thực hiện tuần tự.

Khóa

- Shared lock (LS): khóa đọc, chỉ đọc, không ghi
- Exclusive lock (LX): khóa ghi, có thể đọc, ghi
- UN(D): unlock

	LS	LX
LS	true	false
LX	false	false


T0: LX(A);
read(A);
A := A -50;
write(A);
LX(B);
read(B);
B := B + 50;
write(B);
UN(A);
UN(B);
T1: LX(A);
read(A);
temp := A *0.1;
A := A -temp;
write(A)
LX(B);
read(B);
B:=B+temp;
write(B);
UN(A);
UN(B);

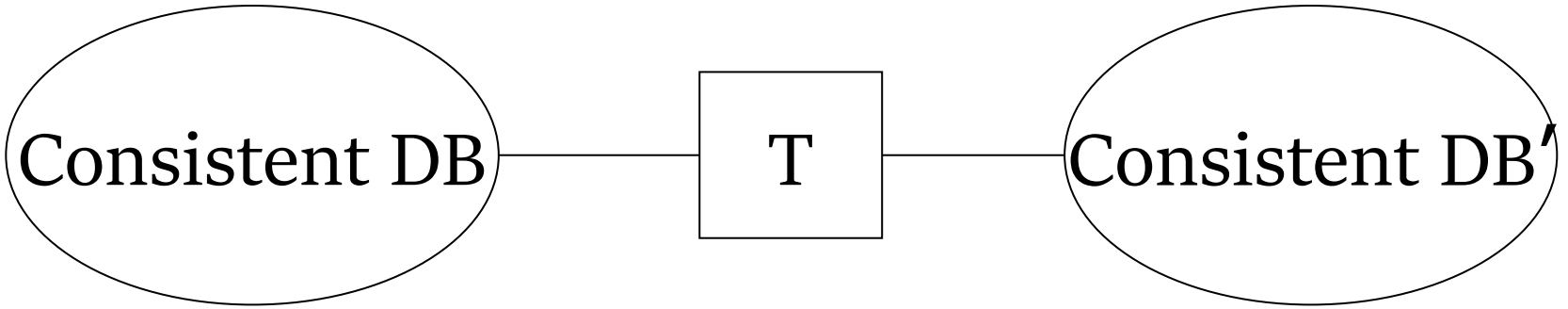
Mức độ cô lập (mức cách ly)

```
Set isolation level  <level>
```

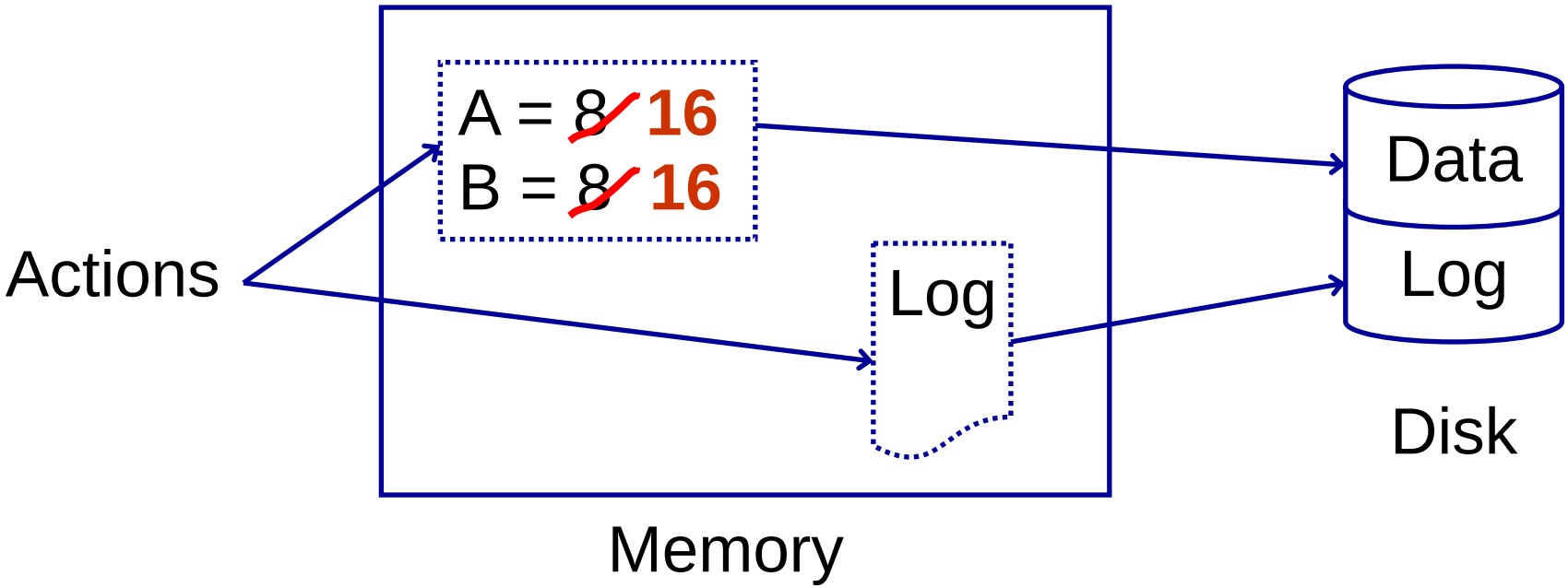
- Read Uncommited: Khóa ghi trong suốt giao dịch, không khóa đọc
- Read Committed: Khóa đọc được giải phóng ngay khi đọc xong, khóa ghi
- Repeatable Read: Khóa đọc và ghi trên khối SQL được chọn
- Serializable: khóa đọc và ghi trên bảng đầy đủ

6.4.5. Khôi phục dữ liệu

- Nếu giao dịch T bắt đầu với Consistent DB, T thỏa tính cách ly, thì T kết thúc với Consistent DB



- Vấn đề - sự cố: lỗi giao dịch, lỗi DBMS, lỗi phần cứng (sự cố đĩa), chia sẻ dữ liệu (T1, T2 song song) ==> Cần phục hồi dữ liệu (ROLLBACK về trạng thái nhất quán gần nhất): sử dụng LOG


A = 8 16
B = 8 16
Log
Actions
Data
Log
Disk
Memory

Undo logging

- Với mọi hành động, tạo bản ghi undo log (giá trị cũ); khi X thay đổi thì bản ghi log liên quan X phải có trên đĩa

Step	Action	t	Mem A	Mem B	Disk A	Disk B	Mem Log
1							<start T>
2	Read(A,t)	8	8		8	8	
3	t:=t*2	16	8		8	8	
4	Write(A,t)	16	16		8	8	<T, A, 8>
5	Read(B,t)	8	16	8	8	8	
6	t:=t*2	16	16	8	8	8	
7	Write(B,t)	16	16	16	8	8	<T, B, 8>
8	**Flush log**						
9	Output(A)	16	16	16	16	8	
10	Output(B)	16	16	16	16	16	
11							<commit T>
12	**Flush log**						

Redo logging

- Sử dụng memory lưu các bản ghi log, ghi vào đĩa khi giao dịch hoàn thành

Step	Action	t	Mem A	Mem B	Disk A	Disk B	Mem Log
1							<start T>
2	Read(A,t)	8	8		8	8	
3	t:=t*2	16	8		8	8	
4	Write(A,t)	16	16		8	8	<T, A, 16>
5	Read(B,t)	8	16	8	8	8	
6	t:=t*2	16	16	8	8	8	
7	Write(B,t)	16	16	16	8	8	<T, B, 16>
8							<commit T>
9	**Flush log**						
10	Output(A)	16	16	16	16	8	
11	Output(B)	16	16	16	16	16	<T, end>

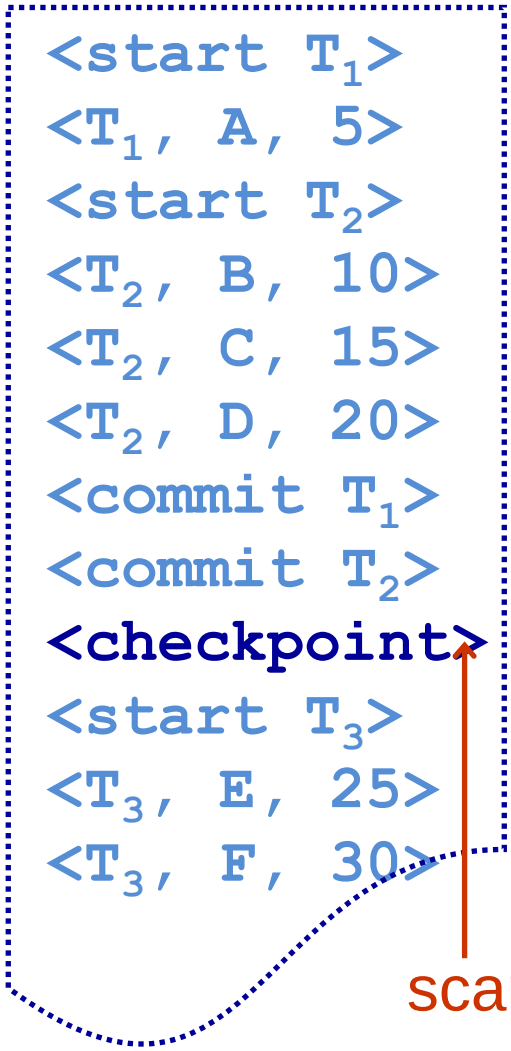
Undo/Redo logging

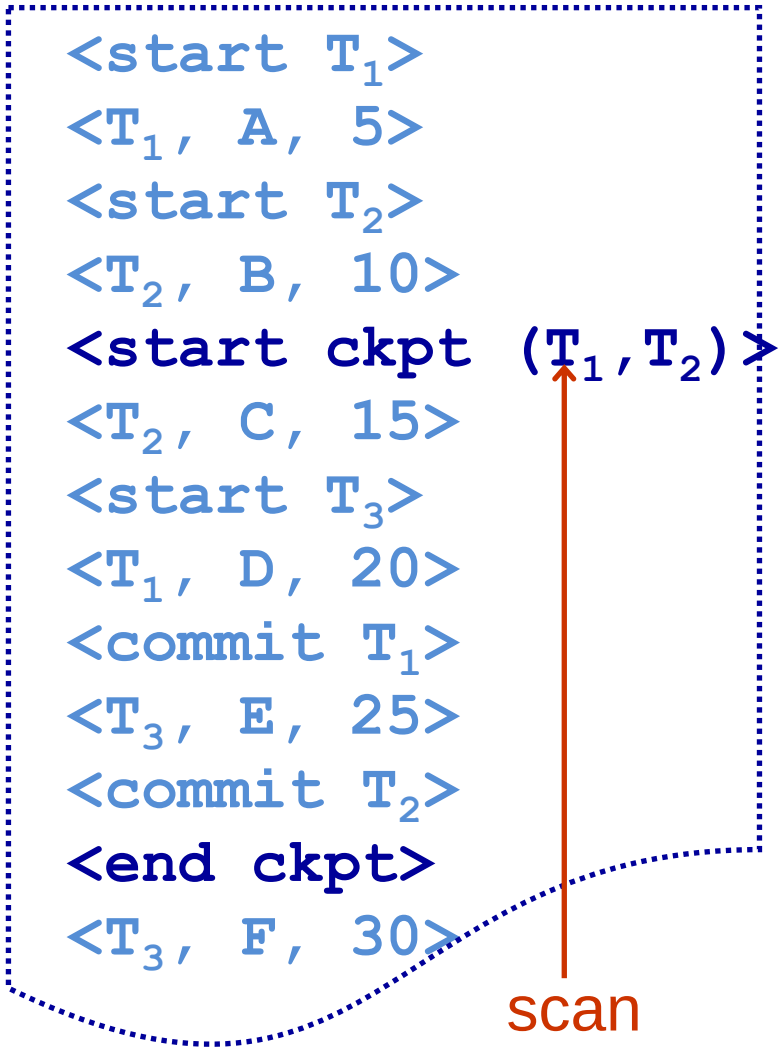
Step	Action	t	Mem A	Mem B	Disk A	Disk B	Mem Log
1							<start T>
2	Read(A,t)	8	8		8	8	
3	t:=t*2	16	8		8	8	
4	Write(A,t)	16	16		8	8	<T, A, 8, 16>
5	Read(B,t)	8	16	8	8	8	
6	t:=t*2	16	16	8	8	8	
7	Write(B,t)	16	16	16	8	8	<T, B, 8, 16>
8	**Flush log**						
9	Output(A)	16	16	16	16	8	
10							<commit T>
11	Output(B)	16	16	16	16	16	

Checkpoint

- Nhật ký giao dịch có thêm <checkpoint> là thời điểm ghi những thay đổi CSDL từ vùng đệm xuống đĩa; Khi đến điểm kiểm tra, DBMS tạm dừng tiếp nhận giao dịch mới, đợi các giao dịch đang thực hiện commit hoặc abort để ghi xuống đĩa ==> Giảm bớt các nhật ký giao dịch không còn cần thiết cho việc khôi phục dữ liệu trong tương lai ==> Giảm thời lượng cần cho phục hồi dữ liệu (chỉ cần duyệt đến điểm kiểm tra gần nhất)
- Điểm kiểm tra linh động: <start ckpt (T1,…)> - <end ckpt> cho phép tiếp nhận các giao dịch mới trong quá trình checkpoint

Điểm kiểm tra cho Undo logging





Chương 7 – Tổ chức dữ liệu vật lý

NỘI DUNG

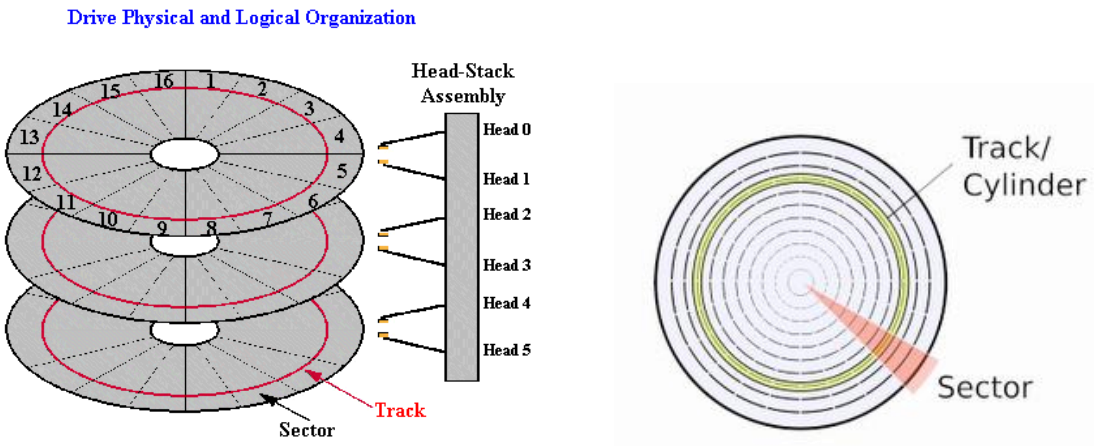
1. Mô hình tổ chức bộ nhớ
2. Tổ chức tệp đống
3. Tổ chức tệp băm
4. Tổ chức tệp chỉ dẫn
5. Cây cân bằng

7.1. Mô hình tổ chức bộ nhớ

- Phương tiện nhớ của máy tính hình thành một phân cấp bộ nhớ bao gồm 2 loại chính:
  - Bộ nhớ sơ cấp:
    - Bao gồm các thiết bị nhớ mà CPU của máy tính có thể thao tác trực tiếp trên đó, như bộ nhớ chính, bộ nhớ đệm cache.
    - Cung cấp cơ thế truy cập dữ liệu nhanh nhưng lại bị giới hạn về dung lượng.
  - Bộ nhớ thứ cấp:
    - Bao gồm các đĩa từ, đĩa quang, băng từ.
    - Dung lượng lớn hơn, chi phí rẻ hơn.
    - Cung cấp cơ chế truy cập dữ liệu chậm hơn.
    - CPU không xử lý dữ liệu trực tiếp trên bộ nhớ thứ cấp mà dữ liệu được sao chép sang bộ nhớ sơ cấp để CPU xử lý.

Bộ nhớ ngoài

- Bộ nhớ ngoài (bộ nhớ thứ cấp): đĩa từ, băng từ,...

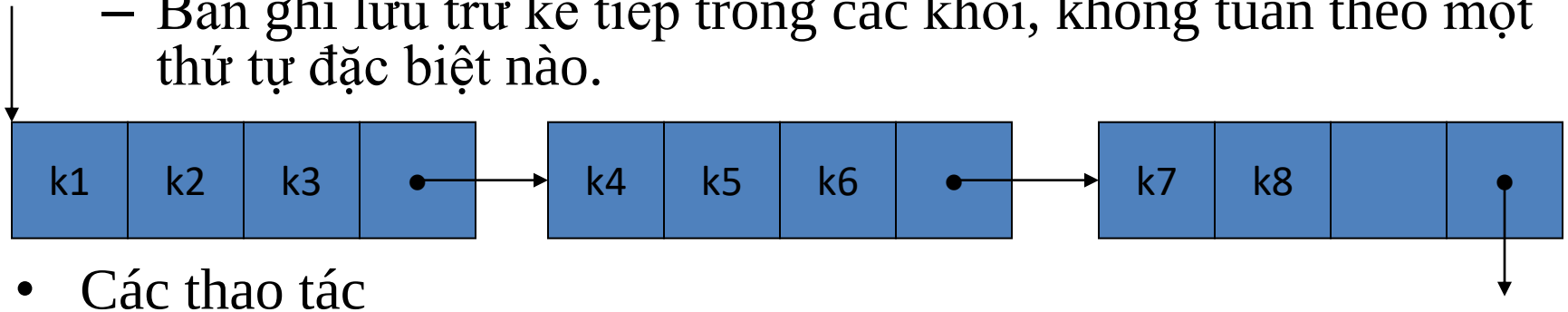


- Đĩa được chia thành các khối vật lý (sector) - 512 byte đến 4096 byte được đánh địa chỉ khối gọi là địa chỉ tuyệt đối
- Mỗi tệp dữ liệu chiếm một hoặc nhiều khối
- Mỗi khối chứa một hoặc nhiều bản ghi

- Thao tác với dữ liệu của tệp thông qua địa chỉ tuyệt đối của các khối.
- Các bản ghi đều có địa chỉ:
  - địa chỉ tuyệt đối của byte đầu tiên
  - địa chỉ khối và số byte tính từ đầu khối đến vị trí đầu bản ghi
- Địa chỉ của các bản ghi/khối được lưu ở một tệp => sử dụng con trỏ (pointer) để truy cập dữ liệu của tệp.

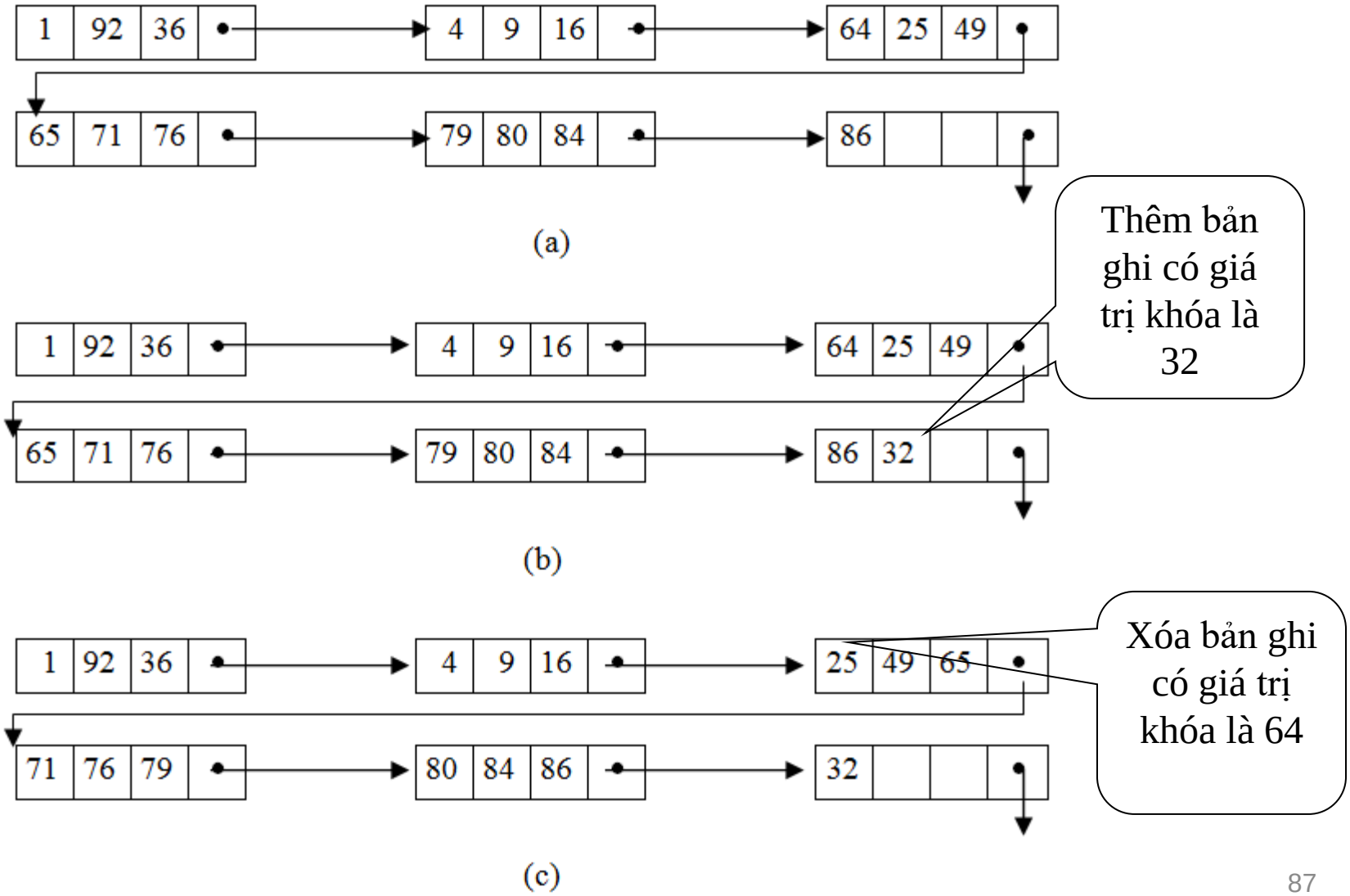
7.2. Tổ chức tệp đống (Heap file)

- Tổ chức dữ liệu
  - Bản ghi lưu trữ kế tiếp trong các khối, không tuân theo một thứ tự đặc biệt nào.



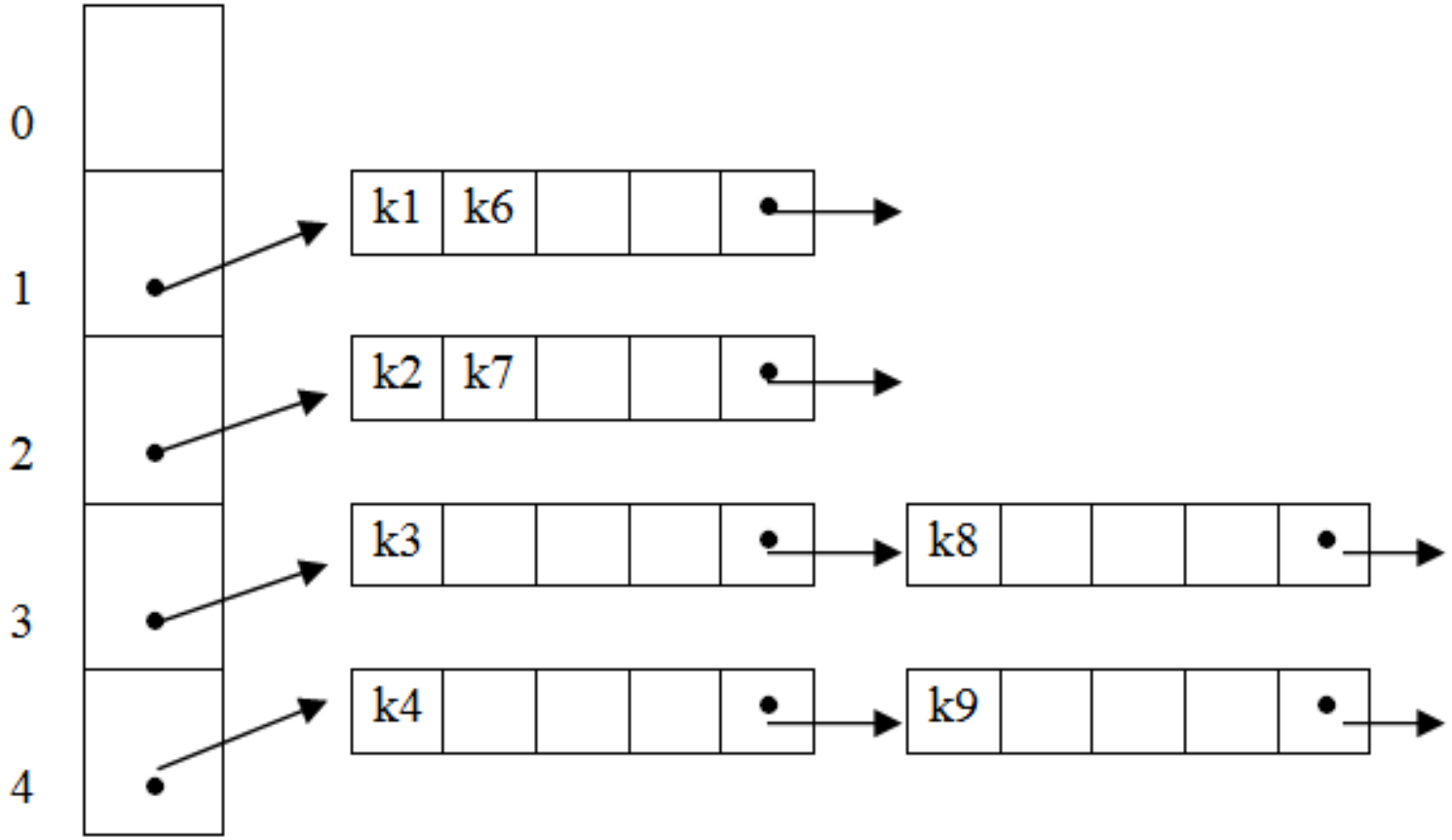
- Các thao tác
  - Tìm kiếm một bản ghi: tìm kiếm một bản ghi có giá trị khóa cho trước => quét toàn bộ tệp.
  - Thêm một bản ghi: thêm bản ghi mới vào sau bản ghi cuối cùng
  - Xóa một bản ghi: thao tác xóa bao hàm thao tác tìm kiếm. Nếu có bản ghi cần xóa thì nó sẽ được đánh dấu là xóa => hệ thống cần tổ chức lại đĩa định kỳ.
  - Sửa một bản ghi: tìm bản ghi rồi sửa một hay nhiều trường.

- **Ví dụ:**

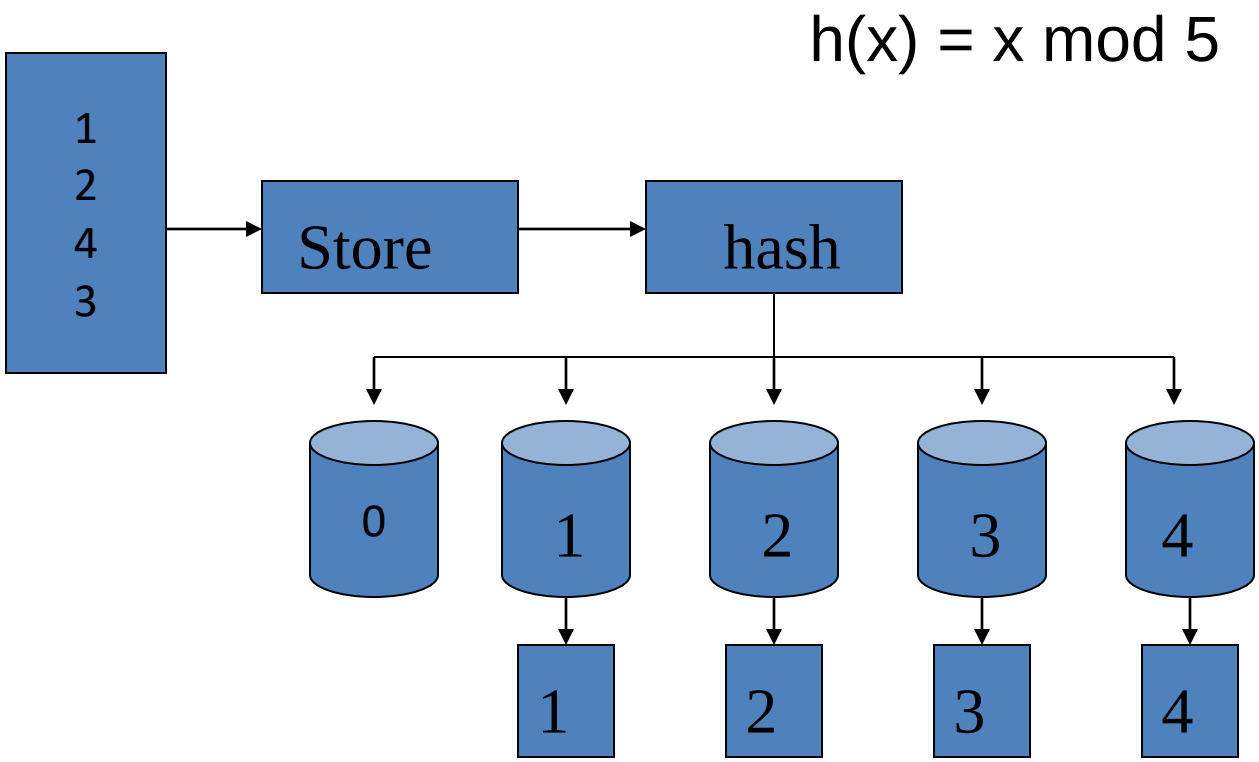

1 92 36
4 9 16
64 25 49
65 71 76
79 80 84
86
(a)
Thêm bản ghi có giá trị khóa là 32
1 92 36
4 9 16
64 25 49
65 71 76
79 80 84
86 32
(b)
Xóa bản ghi có giá trị khóa là 64
1 92 36
4 9 16
25 49 65
71 76 79
80 84 86
32
(c)

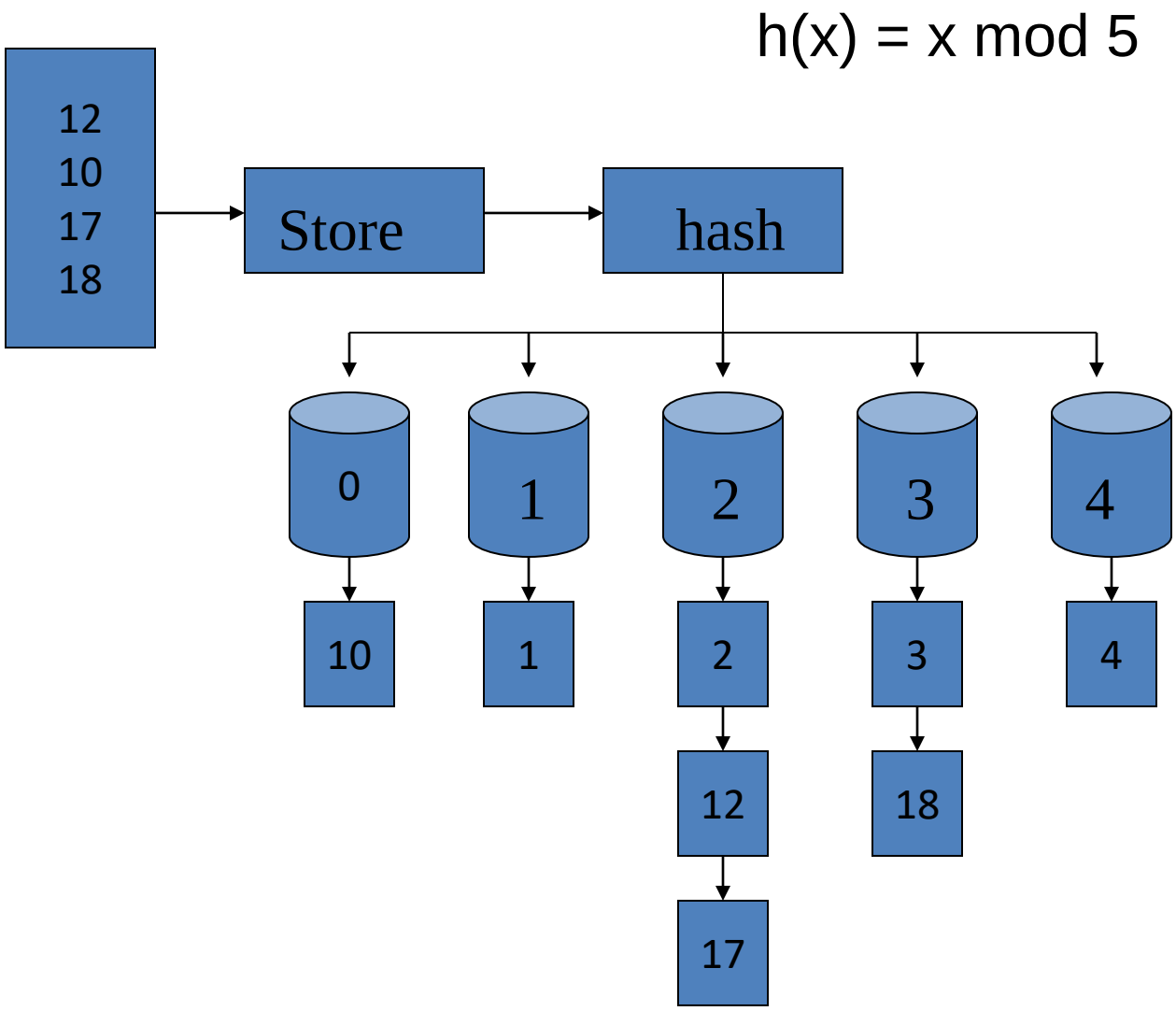
7.3. Tổ chức tệp băm (Hashed files)

- Hàm băm: h(x) nhận một giá trị trong đoạn [0,k], ví dụ: h(x) = x mod k
- Tổ chức tệp dữ liệu
  - Phân chia các bản ghi vào các cụm.
  - Mỗi cụm gồm một hoặc nhiều khối.
  - Mỗi khối chứa số lượng bản ghi cố định.
  - Tổ chức lữu trữ dữ liệu trong mỗi cụm áp dụng theo tổ chức đống
- Tiêu chí chọn hàm băm: phân bố các bản ghi tương đối đồng đều theo các cụm.



Bảng chỉ dẫn cụm

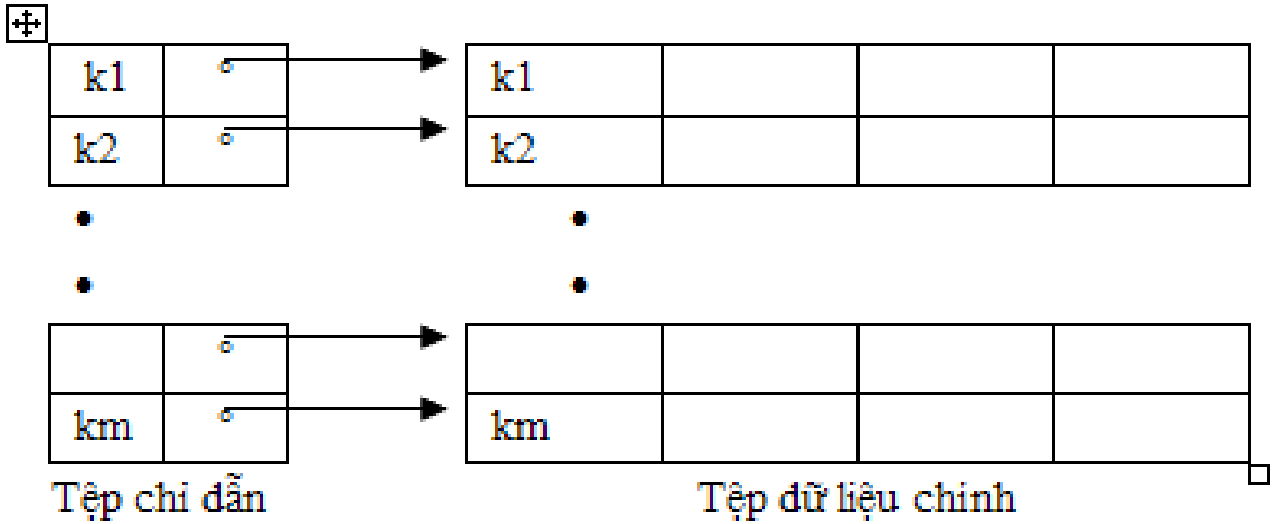

h(x) = x mod 5
1
2
4
3
Store
hash
0
1
2
3
4
1
2
3
4


h(x) = x mod 5
12
10
17
18
Store
hash
0
1
2
3
4
10
1
2
3
4
12
18
17

- Các thao tác:
  - Tìm kiếm một bản ghi: để tìm bản ghi có khóa x, tính h(x) sẽ được cụm chứa bản ghi, sau đó tìm kiếm theo tổ chức đống.
  - Thêm một bản ghi: thêm một bản ghi có giá trị khóa là x.
    - nếu trong tệp đã có một bản ghi có trùng khóa x =>bản ghi mới sai (vì khóa là duy nhất!)
    - nếu không có bản ghi trùng khóa, bản ghi được thêm vào khối còn chỗ trống đầu tiên trong cụm, nếu hết chỗ thì tạo khối mới.
  - Xóa một bản ghi: tìm kiếm bản ghi rồi xóa
  - Sửa đổi một bản ghi:
    - nếu trường cần sửa có tham gia vào trong khóa thì việc sửa sẽ là loại bỏ bản ghi này và thêm mới một bản ghi (bản ghi có thể thuộc vào một cụm khác)
    - nếu trường cần sửa không thuộc khóa: tìm kiếm rồi sửa. Nếu bản ghi không tồn tại thì xem như có lỗi.

7.4. Tổ chức tệp chỉ dẫn(Indexed Files)

- Giả sử giá trị các khóa của các bản ghi được sắp xếp tăng dần.
- Tệp chỉ dẫn được tạo bằng cách chọn các giá trị khóa trong các bản ghi
- Tệp chỉ dẫn bao gồm các cặp (k,d), trong đó k là giá trị khoá của bản ghi đầu tiên, d là địa chỉ của khối (hay con trỏ khối).

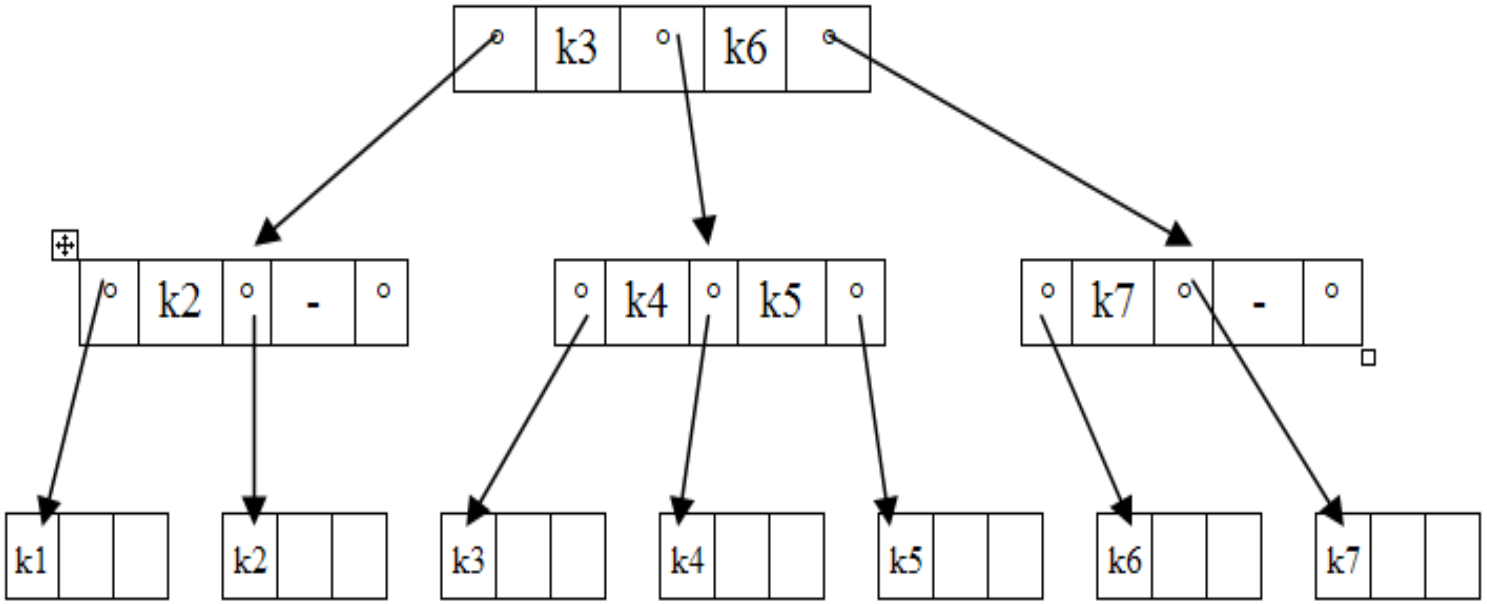


- Tìm kiếm trên tệp chỉ dẫn
  - Cho một giá trị khóa k_i, tìm một bản ghi (k_m,d) trong tệp chỉ dẫn sao cho $k_m <= k_i$ và:
    - hoặc (k_m,d) là bản ghi cuối cùng trong tệp chỉ dẫn
    - hoặc bản ghi tiếp theo (k_{m+1},d') thỏa mãn $k_i<k_{m+1}$
  - Khi đó, chúng ta nói k_m phủ k_i
  - Tìm kiếm này có thể là:
    - tuần tự
    - nhị phân

- Các thao tác:
  - Tìm kiếm một bản ghi
  - Thêm một bản ghi: xác định khối i sẽ chứa bản ghi đó
    - nếu trong khối i còn chỗ thì đặt bản ghi này vào đúng chỗ theo thứ tự sắp xếp của khóa, dồn toa các bản ghi đằng sau nó.
    - nếu khối i hết chỗ thì việc thêm này sẽ đẩy bản ghi cuối cùng trong khối sang làm bản ghi đầu tiên của khối tiếp theo i+1 => sửa bản ghi chỉ dẫn tương ứng
    - nếu bản ghi mới này có giá trị khóa lớn hơn tất cả mọi khóa trong tệp dữ liệu chính và không còn chỗ thì tạo thêm một khối mới.
  - Xóa một bản ghi: giống như thêm một bản ghi, nếu xóa mà tạo thành một khối rỗng, khi đó có thể loại bỏ cả khối đó.
  - Sửa một bản ghi:
    - Sử dụng thủ tục tìm kiếm để xác định bản ghi cần sửa
    - nếu các trường cần sửa không phải là khóa thì sửa bình thường
    - nếu các trường cần sửa tham gia vào khóa thì quá trình sửa sẽ là quá trình thêm và xóa một bản ghi.

7.5. Cây cân bằng(Balanced-trees)

- B-tree được tổ chức theo cấp m, có các tính chất sau đây:
  - Gốc của cây hoặc là một nút lá hoặc ít nhất có hai con.
  - Mỗi nút (trừ nút gốc và nút lá) có từ [m/2] đến m con.
  - Mỗi đường đi từ nút gốc đến bất kỳ nút lá nào đều có độ dài như nhau.



- Cấu trúc của mỗi nút trong B-cây có dạng (p_0, k_1, p_1, k_2,...,k_n, p_n) với p_i (i=1..n) là con trỏ trỏ tới khối i của nút có k_i là khoá đầu tiên của khối đó. Các khoá k trong một nút được sắp xếp theo thứ tự tăng dần.
- Mọi khoá trong cây con, trỏ bởi con trỏ p_0 đều $\leq k_1$;
- Mọi khoá trong cây con, trỏ bởi con trỏ p_i đều nhỏ hơn k_{i+1}.
- Mọi khoá trong cây con, trỏ bởi con trỏ p_n đều $\geq k_n$.

- Các thao tác:
  - Tìm kiếm một bản ghi: xác định đường dẫn từ nút gốc tới nút lá chứa bản ghi này
  - Thêm một bản ghi:
    - Xác định vị trí nút lá sẽ chứa bản ghi này (như tìm kiếm)
    - Nếu còn chỗ thì thêm bình thường
    - Nếu hết chỗ thì phải tạo thêm nút lá mới, chuyển nửa dữ liệu cuối của nút lá hiện tại sang nút mới, sau đó thêm bản ghi mới này vào vị trí phù hợp nút lá hiện tại hoặc nút mới tạo
    - Rất có khả năng "động chạm" đến nút cha,….nút gốc.
  - Loại bỏ một bản ghi:
    - Dùng thủ tục tìm kiếm một bản ghi để xác định nút L có thể chứa bản ghi đó.
    - Rất có khả năng "động chạm" đến nút cha,…,nút gốc.

Kết luận

- Tổ chức tệp chỉ dẫn:
  - được áp dụng phổ biến
  - Với các ứng dụng yêu cầu cả xử lý tuần tự và truy nhập trực tiếp đến các bản ghi
  - Hiệu năng sẽ giảm khi kích thước tệp tăng =>chỉ dẫn B-cây
- Tổ chức băm:
  - Dựa trên một hàm băm, cho phép tìm thấy địa chỉ khoản mục dữ liệu một cách trực tiếp
  - Hàm băm tốt? Phân bố các bản ghi đồng đều trong các cụm