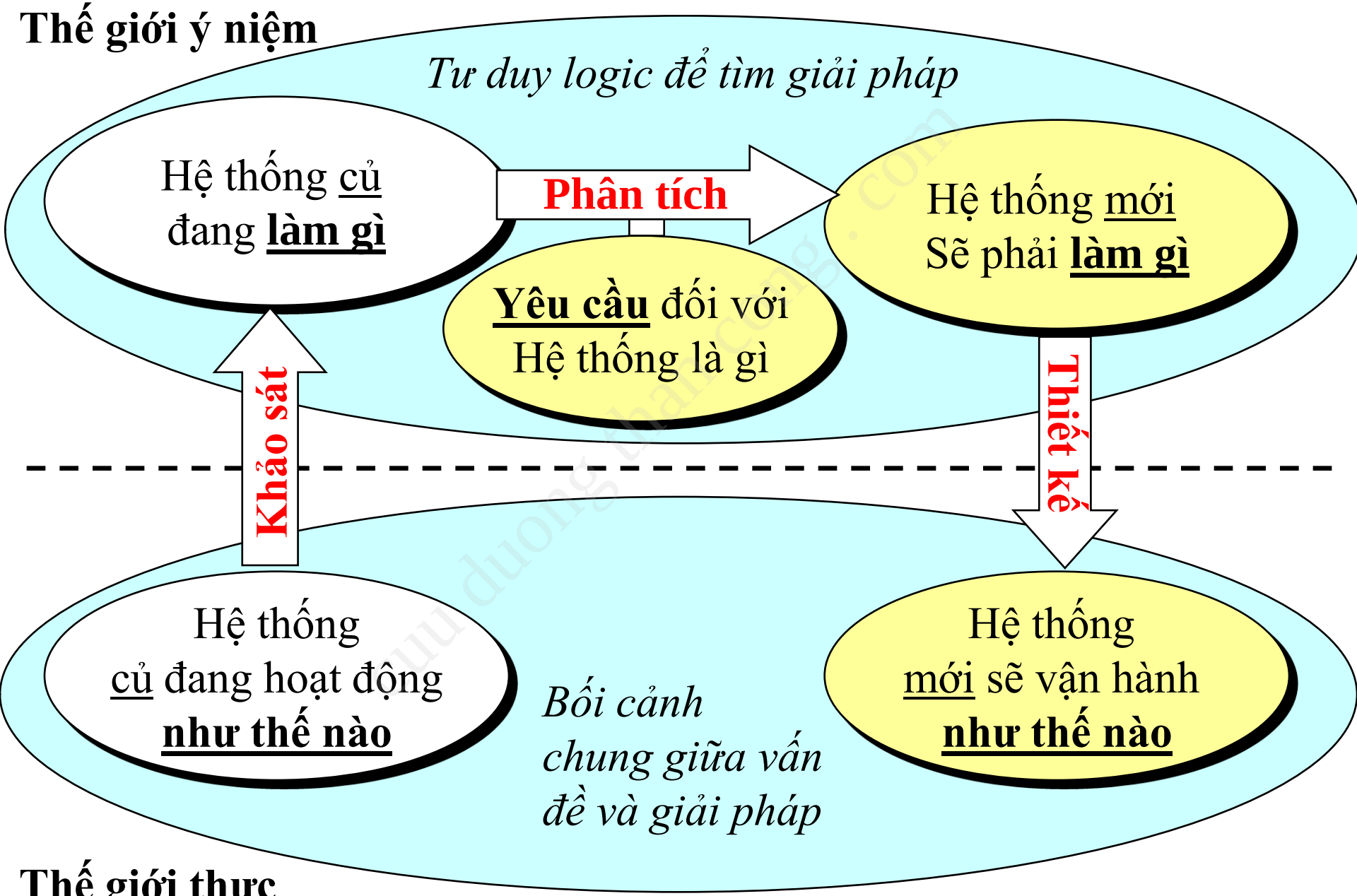
PHÂN TÍCH THIẾT KẾ HỆ THỐNG
Tiếp cận hướng cấu trúc

Tháng 9-2007

ThS. Nguyễn Anh Hào

Phương pháp luận PT-TK-HT



1. Xác định yêu cầu (khảo sát)

- Khảo sát hiện trạng là quá trình khám phá cách mà hệ thống đã được thiết kế và vận hành trong thực tế, làm bộc lộ các quan hệ nội tại giữa các thành phần trong hệ thống và mối liên hệ giữa hệ thống với yêu cầu.
- Xác định yêu cầu cho hệ thống là một quá trình tổng hợp thông tin mang tính hệ thống và khách quan, không thể chỉ dựa vào mô tả của một vài cá nhân, vì
  - Mỗi cá nhân chỉ nhìn hệ thống theo một lĩnh vực chuyên môn đang phụ trách; do đó các phát biểu thường không bộc lộ được các ràng buộc tổng thể của hệ thống.
  - Các phát biểu của nhiều người thường có mâu thuẫn nhau do mỗi người có quan điểm khác nhau về hệ thống hiện tại.

Nội dung khảo sát

1. Tìm hiểu mục đích, yêu cầu đối với hệ thống
   - Xác định vai trò (lợi ích) của hệ thống đối với tổ chức.
2. Tìm hiểu các quy trình giữa các thành phần trong hệ thống
   - Để làm bộc lộ mối quan hệ nội tại giữa các thành phần

   a. Công việc: quy tắc quản lý, cách làm, kết quả, chuyển giao

   b. Nguồn lực: mức độ, phương tiện, nhân lực, cách tổ chức bố trí
3. Tìm hiểu thông tin – dữ liệu của quy trình
   - Các quy định, hướng dẫn, tiêu chuẩn
   - Forms/Reports : có thông tin gì, khi nào cần, dùnd để làm gì,..
4. Hệ thống thông tin trên máy tính (CBIS) hiện có
   - Phạm vi, mức độ và cách nó trợ giúp users thực hiện công việc
   - Vai trò (roles) của các users trong hệ thống.
   - Phần mềm, mạng máy tính, thiết bị,…

Các phương pháp khảo sát truyền thống

1. Phỏng vấn cá nhân (interviews)
2. Phỏng vấn nhóm (group interviews)
3. Phiếu thăm dò (questionaires)
4. Quan sát người sử dụng (viewing)
5. Phân tích tài liệu (document analysis)

Các phương pháp này thuần túy là chỉ dùng để biết về hệ thống hiện tại.

1.Phỏng vấn cá nhân (interviews)

- Phỏng vấn: hẹn gặp, tiếp xúc, hỏi và ghi nhận cân trả lời.
- Ưu điểm
  - Có cơ hội hỏi thêm về những gì vừa mới biết
- Khuyết điểm
  - Có thể có mâu thuẩn ý kiến riêng giữa các cá nhân
  - Tốn nhiều thời gian nếu cần phỏng vấn nhiều người

2.Phỏng vấn nhóm (group interviews)

- Phỏng vấn nhóm: Đặt câu hỏi chung cho nhiều người chủ chốt cùng một lúc trong cuộc họp, hội thảo.
- Ưu điểm
  - Ít tốn thời gian hơn phỏng vấn cá nhân
  - Gia tăng sự trao đổi về các "phát hiện mới" giữa những người tham gia => có cơ hội hiểu biết sâu hơn.
  - Hạn chế bớt sự mâu thuẫn ý kiến cá nhân
- Khuyết điểm:
  1. khó thu xếp cho cuộc phỏng vấn
     - Do có khoảng cách kiến thức chuyên môn
     - Khó sắp xếp thời gian và địa điểm họp
  2. Có sự vị nể giữa các cá nhân → thiếu trung thực
  3. Có người nói quá nhiều → tốn thời gian vô ích

3.Phiếu thăm dò (questionaires)

- Gửi phiếu có ghi câu hỏi phỏng vấn đến nhiều người, sau đó phân tích/thống kê kết quả trả lời từ các phiếu đã quay về. Câu hỏi phải hết sức rõ ràng, dễ hiểu và dễ trả lời để người được phỏng vấn không bị nhầm lẫn.
- Ưu điểm
  - Rẻ hơn các loại phỏng vấn
  - Thống kê trên số lượng lớn phiếu quay về có thể nhận được thông tin tương đối khách quan.
- Khuyết điểm
  - Không có cơ hội để hỏi thêm !
  - Không chắc chắn ai là tác giả !!
  - Số phiếu quay về có thể quá ít.

So sánh Interviews và Questionaires

Tính chất	Interviews	Questionaires
Giàu thông tin	Cao	T.bình - Thấp
Thời gian	Có thể rất lâu	Thấp – T.bình
Chi phí	Có thể cao	vừa phải
Tìm hiểu sâu thêm	Tốt	Giới hạn
Độ tin cậy	Cao. Đã biết rõ người được phỏng vấn.	Không cao. Không xác định được tác giả.
Mức độ cộng tác	Người được phỏng vấn cùng tham gia giải quyết vấn đề và cam kết thực hiện	Không rõ các cam kết
Người tham dự	Số lượng giới hạn, đáp ứng tốt	Số lượng lớn, đáp ứng không tốt.

4. Quan sát người nhân viên

- Để biết họ thường làm gì, và xử lý công việc ra sao; đồng thời để đánh giá mức độ hiệu quả của các quy trình và các phương tiện hỗ trợ cho người nhân viên làm việc.
- Ưu điểm
  - Kiểm chứng được những gì đã biết.
  - Biết được cường độ của từng công việc trong thực tế
- Khuyết điểm
  - Sự quan sát có thể không khách quan, do người nhân viên sẽ thay đổi thói quen nếu biết mình đang bị quan sát.
  - Tốn nhiều thời gian ngồi quan sát.

5. Thu thập tài liệu

- Thu thập, đọc và tìm hiểu các tài liệu văn bản mô tả về hệ thống như hồ sơ thiết kế, các biểu mẫu nhập liệu, các báo cáo, các quy trình vận hành khai thác,…
- Ưu điểm:
  - Có nhiều thông tin chi tiết, chính xác.
  - Dể dàng khái quát/hệ thống hóa được toàn bộ hệ thống
- Khuyết điểm:
  - Tài liệu có thể bị lạc hậu so với thực tế

Các phương pháp khảo sát hiện đại

1. Làm mẫu thử (Prototyping)
2. Joint Application Design (JAD)
3. Tái cấu trúc tiến trình (Business Process Reengineering)

➲ Khác với phương pháp truyền thống, phương pháp hiện đại sử dụng nhiều loại nguồn lực trợ giúp (ngoài người được phỏng vấn) để định nghĩa hệ thống mới ngay trong khi đang khảo sát.

1. Làm mẫu thử (prototyping)

- Tìm hiểu sơ lược yêu cầu ban đầu, chuyển yêu cầu này thành ‘demo’ cho người sử dụng kiểm tra để hiệu chỉnh lại. Qua nhiều chu kỳ kiểm tra – hiệu chỉnh, bản demo (thể hiện yêu cầu+giải pháp của hệ thống mới) được hoàn chỉnh dần từ tổng quát đến chi tiết.
- Ưu điểm
  - Phân tích viên hiểu được cặn kẽ yêu cầu chi tiết từ User
  - User biết được hệ thống mới sẽ hoạt động như thế nào
- Khuyết điểm
  - Không thể diễn tả xử lý hệ thống (là những gì người sử dụng không nhìn thấy được, do tính tổng quát của nó)
  - Khó thống nhất quan điểm sử dụng từ nhiều users

2. Joint Application Design (JAD)

- Là một chuỗi các cuộc họp chuyên sâu có sử dụng các phương tiện "demo" như CASE tools, và có ❶ người sử dụng, ❷ người quản lý và ❸ người phát triển hệ thống cùng tham gia để đặc tả hoặc xem xét tường tận các yêu cầu cho hệ thống.
  1. Người sử dụng là người đưa ra yêu cầu về các chuyển giao sau khi thiết kế.
  2. Người quản lý là người nêu ra bài toán và quyết định có chấp nhận phương án không.
  3. Người phát triển hệ thống là người đưa ra phương án giải quyết bài toán.
- Ưu điểm: prototyping + phỏng vấn nhóm
- Khuyết điểm: khá tốn kém vì nhân lực+thời gian dài

3. Tái cấu trúc các tiến trình kinh doanh

- BPR: Thay vì "cải tiến" hệ thống cho phù hợp với mô hình tổ chức và các quy tắc quản lý hiện tại, phương pháp này hướng việc khảo sát vào việc tận dụng ***ưu thế của các loại nguồn lực bên trong và bên ngoài*** để tái thiết lại hệ thống.
  - Thay đổi mô hình và nghiệp vụ để ứng dụng CNTT
  - Phá bỏ các nguyên tắc lạc hậu trì trệ đang tồn tại
  - Quan điểm: "Nếu một tổ chức được xây dựng lại từ đầu, thì nó cần phải hoạt động như thế nào?"
- Ưu điểm: tạo ra thành tựu lớn nhưng có nhiều rủi ro.
- Ví dụ: Nhà sách "Amazon.com" bán sách điện tử thay cho các quyển sách giấy ⇒ không có chi phí lưu kho, không có quầy giao dịch và trưng bày, mở rộng kinh doanh, … nhưng phải đối mặt với vấn đề "copy rights" của sách điện tử.

Đánh giá sơ lược sau khảo sát

1. Kết luận sơ lược sau khi khảo sát
   - Tần suất, cường độ công việc cao ở đâu, khi nào
   - Có bị nghẽn cổ chai ?
   - Có xung khắc thông tin ?
   - Hiệu quả xử lý các báo cáo
2. Nhận định sơ lược về cơ hội và thách thức để khắc phục, cải tiến hoặc cải cách để định hướng tập trung phân tích
   1. Từ nội bộ của tổ chức: nguồn lực, kế hoạch, chiến lược
   2. Môi trường bên ngoài: quy định của chính phủ và phương pháp,công nghệ mới trên thế giới

2. Mô hình hóa hệ thống

- Sau khi khảo sát và hiểu rõ hệ thống hiện tại, người phân tích viên cần phải mô tả lại hệ thống:
  - Để hệ thống hóa những gì đã biết
  - Để chia sẽ hiểu biết của cá nhân với nhóm công tác (có người sử dụng, người quản lý, người thiết kế,…)
- Việc mô tả lại hệ thống chỉ thực sự hiệu quả khi các đặc trưng quan trọng nhất của hệ thống được làm sáng tỏ, **VÀ** các chi tiết không quan trọng đã được lược bỏ.
- Ngôn ngữ tự nhiên thường gây hiểu lầm, và không trợ giúp cho việc khái quát hóa (abstraction) và hệ thống hóa => người ta thay thế chúng bằng các mô hình (models).

Mô hình

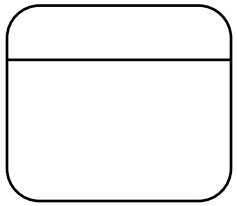
- Mô hình là sự thể hiện các đặt trưng quan trọng nhất của hệ thống theo một quan điểm phân tích nào đó, và lược bỏ các chi tiết không quan trọng. Trong hệ thống thông tin, mô hình là một hệ thống các lược đồ sử dụng các ký hiệu, hình ảnh gợi nhớ để diễn tả cho nhận thức thay cho các phát biểu mô tả dài dòng, như lược đồ DFD, ERD, UMLs.
- Mô hình có 3 đặc tính cơ bản:
  1. Ngữ pháp (notations): là các quy tắc sử dụng các ký hiệu hình thức cho mô hình, để loại bỏ những mô tả vô lý hoặc tối nghĩa.
  2. Ngữ nghĩa (semantics): là nội dung (ý) cần diễn tả lại.
  3. Ngữ cảnh (context): là kiến thức chung giữa người xem và người tạo ra mô hình để nội dung ngữ nghĩa của mô hình được truyền đạt trọn vẹn cho người đọc. Vì lý do này, một lược đồ cho hệ thống chỉ được tạo ra chỉ từ một quan điểm phân tích nào đó.

Các phương pháp mô hình hóa hệ thống

1. Hướng cấu trúc: Phân rã hệ thống bằng kỹ thuật phân tích có cấu trúc (Structured Analysis & Design Technique, SADT) và mô tả lại bằng lược đồ Data Flow Diagram (DFD), Processing Logic và Từ điển dữ liệu, + xác định các thực thể, quan hệ giữa các thực thể và các thuộc tính có dữ liệu của chúng để liên kết một cách có cấu trúc các thành tố dữ liệu trong hệ thống, và mô tả lại bằng Entity Relationship Diagram (ERD).
2. Hướng đối tượng: Xem xét toàn bộ hệ thống như một kiến trúc liên kết hoạt động của các đối tượng (có thuộc tính và phương thức) tham gia thực hiện công việc , mô tả lại theo nhiều hướng nhìn khác nhau (UML).

Lược đồ dòng dữ liệu - Data Flow Diagram

➲ DFD là lược đồ sử dụng 4 ký hiệu cùng với các quy tắc vẽ để diễn tả các dòng dữ liệu di chuyển trong hệ thống.



Process : Là một hành động hoặc một hệ thống con xử lý trên dữ liệu (biến đổi, lưu trữ hoặc phân phối dữ liệu).

Data store : Là bộ phận dùng để lưu trữ dữ liệu, như tập tin, hồ sơ, CSDL,…

Source/Sink : Là thành phần phát sinh dữ liệu (source) cho hệ thống, hoặc tiêu thụ dữ liệu (sink) từ hệ thống.

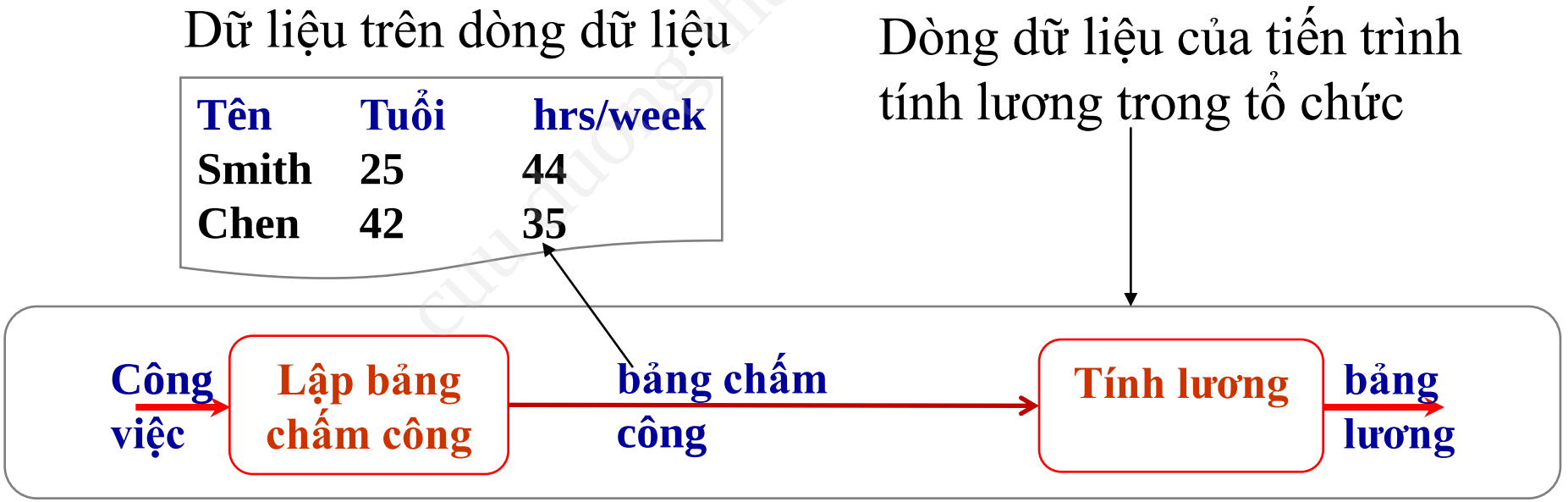
Data flow : dòng dữ liệu cơ bản của lược đồ, chỉ ra 1 nội dung dữ liệu (không đổi) được gởi từ đâu và đi đến đâu.

➲ Quy ước

- Dùng **Động từ** để đặt tên cho Process
- Dùng **Danh từ** để đặt tên Data store, Source, Sink và Data flow

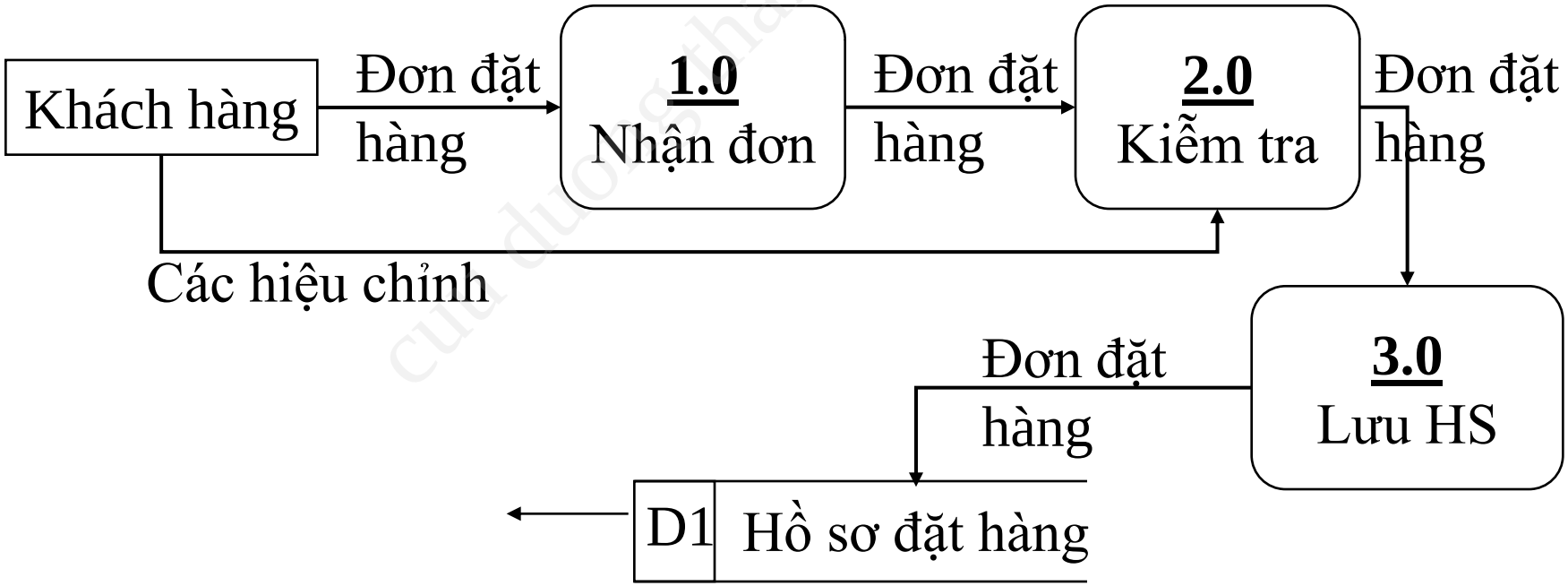
Dòng dữ liệu

- Dòng dữ liệu được tạo thành từ các vật mang dữ liệu (ví dụ: chứng từ, hóa đơn, phiếu nhập xuất kho,…) qua các xử lý trung gian biến đổi dữ liệu theo các quy tắc quản lý của tổ chức (business rules).
- Các dòng vật chất, tiền tệ, dịch vụ, thông tin quan trọng trong hệ thống là nguồn gốc làm phát sinh dòng dữ liệu trên lược đồ.



Dòng dữ liệu

- Các dòng dữ liệu trong hệ thống có kiểm soát như công ty, doanh nghiệp đều phải tuân thủ **các quy tắc quản lý (business rules);** vì vậy các quy tắc quản lý (và quy trình) của hệ thống là cơ sở để thiết lập các dòng dữ liệu trong lược đồ DFD.
- Ví dụ: Bộ phận giao dịch nhận đơn đặt hàng từ khách hàng, kiểm tra đơn đặt hàng để hiệu chỉnh nếu cần, sau đó lưu vào hồ sơ đặt hàng.

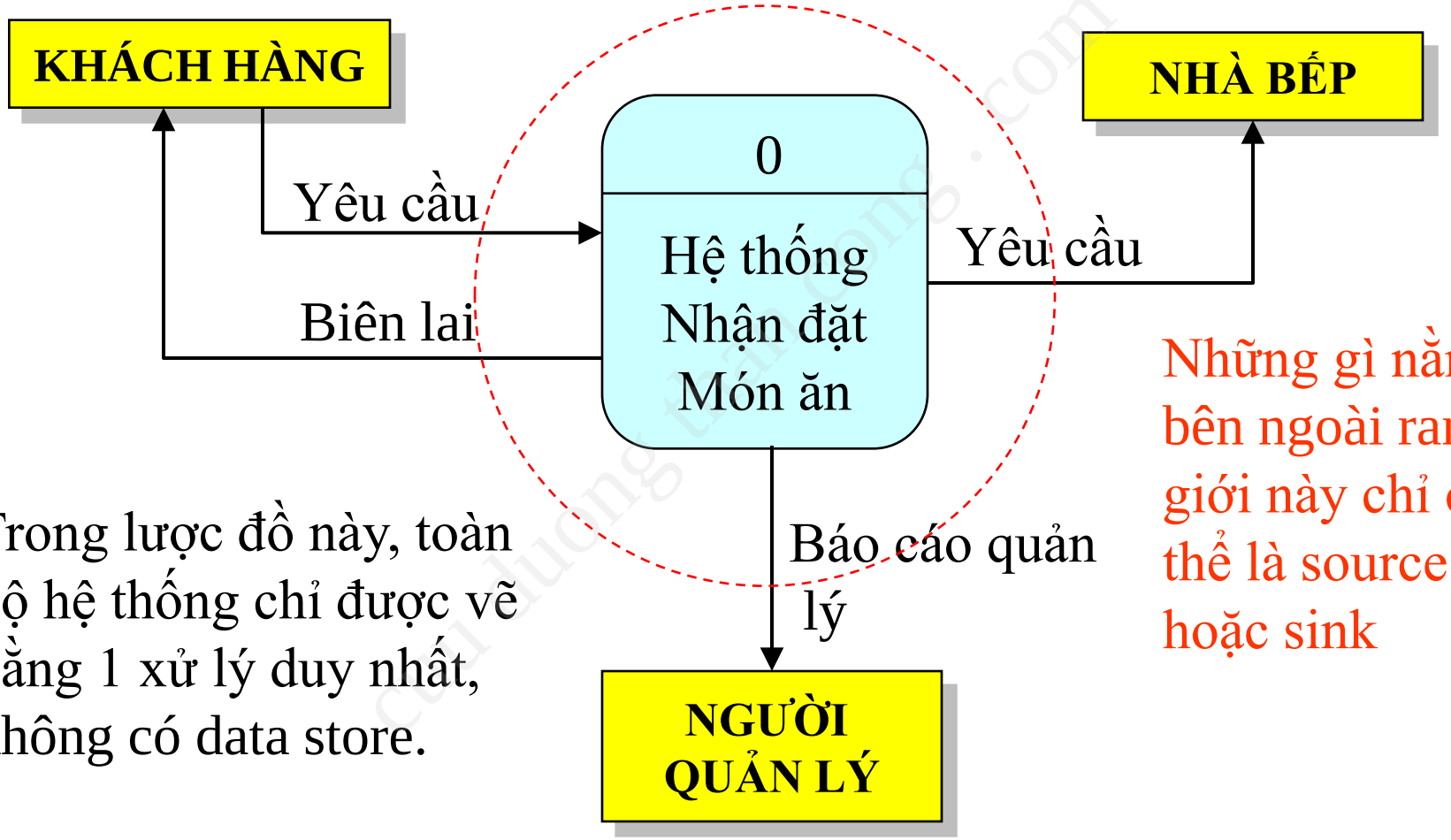


Lược đồ ngữ cảnh

- Lược đồ ngữ cảnh (context diagram) là lược đồ DFD được dùng để mô tả khái quát toàn bộ các dòng dữ liệu đi từ nguồn (SOURCE) vào hệ thống và đi từ hệ thống ra đến đích (SINK) mà không quan tâm đến các xử lý biến đổi và lưu trữ dữ liệu bên trong hệ thống.
- Lược đồ ngữ cảnh cho biết hệ thống cần chuyển giao (dữ liệu) gì và nó nhận được (dữ liệu) gì từ môi trường bên ngoài.
- Ví dụ: nhà hàng Hoosie Burger có một hệ thống "đặt hàng các món ăn" được mô tả tổng quát theo các quy tắc quản lý như sau: hệ thống sẽ nhận yêu cầu từ khách hàng, in biên lai thanh toán tiền cho khách hàng, chuyển yêu cầu cho nhà bếp thực hiện và in các báo cáo quản lý cho người quản lý nhà hàng vào cuối ngày.

Lược đồ ngữ cảnh

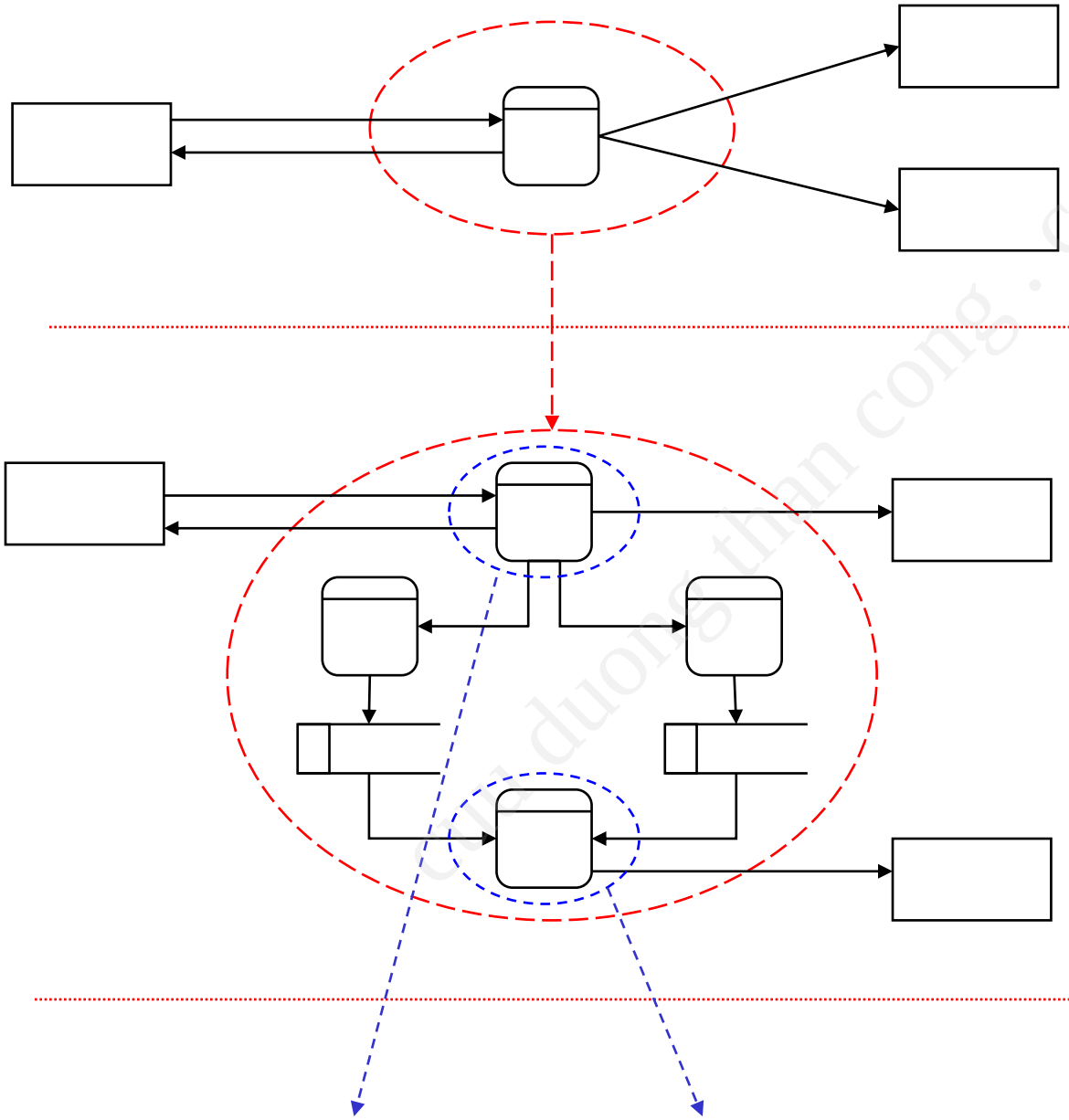
Ngữ cảnh của hệ thống nhận đặt món ăn của Hoosie Burger



Trong lược đồ này, toàn bộ hệ thống chỉ được vẽ bằng 1 xử lý duy nhất, không có data store.

Những gì nằm bên ngoài ranh giới này chỉ có thể là source hoặc sink

Quy tắc phân rã



Ngữ cảnh /
DFD mức n

DFD mức n+1

Một lược đồ DFD mức n+1 là lược đồ DFD mô tả chi tiết hơn cho một xử lý trong lược đồ DFD mức n

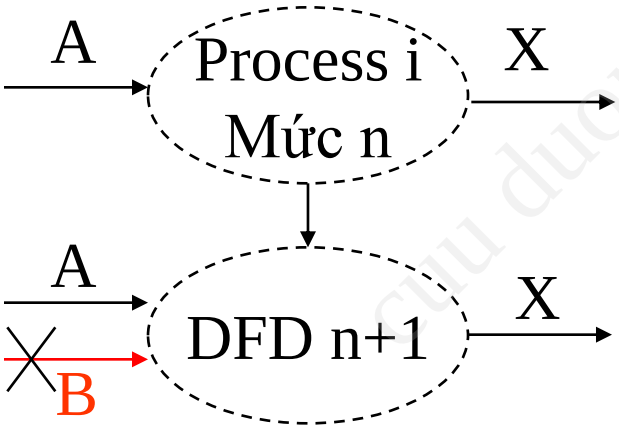
DFD mức n+2

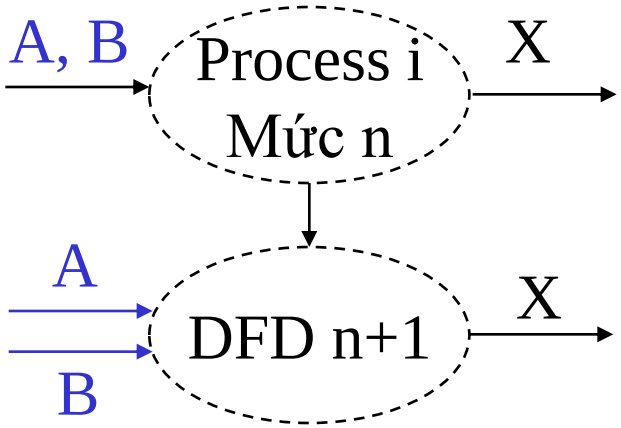
Quy tắc vẽ lược đồ

1. Nếu một đối tượng chỉ có outputs, chắc chắn đối tượng đó phải là source. Tương tự, nếu một đối tượng chỉ có inputs, nó phải là sink.
2. Một xử lý (hoặc data store) phải có cả inputs lẫn outputs. Không có xử lý nào chỉ có inputs mà không có outputs, hoặc ngược lại.
3. Một dataflow là một mũi tên phải có nhãn và có duy nhất một hướng để chỉ rõ nơi đi và nơi đến của dữ liệu. Do đó, nếu một nội dung dữ liệu được chuyển đi và nhận về giữa hai đối tượng thì nó phải được vẽ bằng 2 mũi tên (theo 2 hướng ngược nhau).
4. Không có dòng dữ liệu trực tiếp giữa các data store, source, sink; vì đây là những đối tượng "thụ động"; để di chuyển dữ liệu giữa các đối tượng này cần phải có ít nhất một xử lý của hệ thống.
5. Không có dòng dữ liệu rẽ nhánh (hoặc gộp) có nội dung (nhãn) khác nhau. Nội dung dữ liệu ở các nhánh phải giống y như nhau.
6. Không có dòng dữ liệu trực tiếp đi từ một xử lý đến chính nó (vì một xử lý không cần gửi dữ liệu cho chính nó).

Quy tắc cân bằng

1. Nếu một xử lý i (mức n) được phân rã thành một lược đồ DFD mức n+1 cho xử lý này thì nội dung dữ liệu vào ra của xử lý i phải được bảo toàn (không thêm/bớt).
2. Nếu dòng dữ liệu ở mức n là dạng tổng quát (như "thông tin khách hàng",..) trong khi các dòng dữ liệu ở lược đồ DFD mức n+1 là dạng chi tiết ("tên khách", "địa chỉ",…) thì phải sử dụng từ điển dữ liệu để liên kết chúng với nhau.



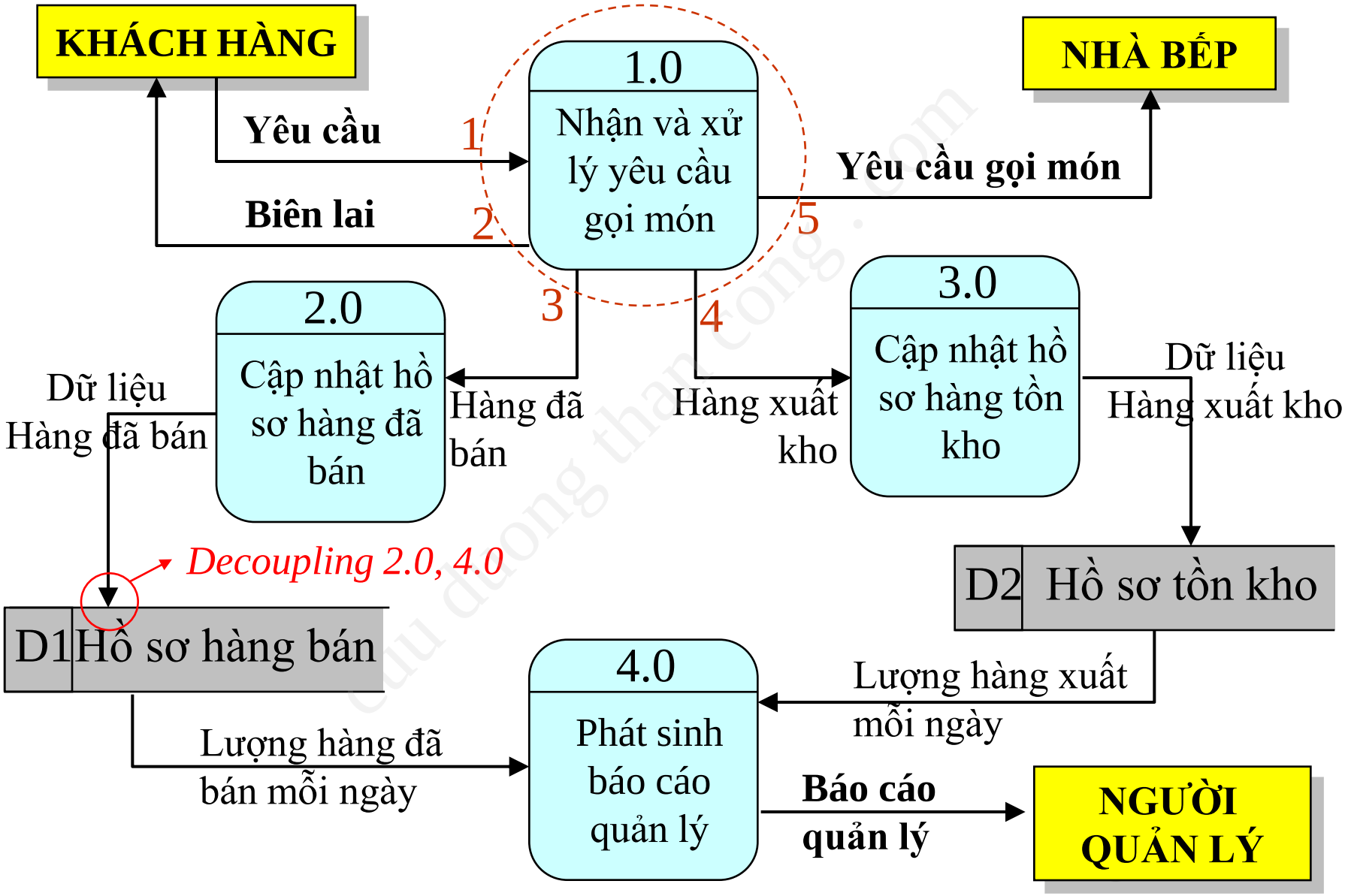


Lược đồ dòng dữ liệu mức 0 (DFD-0)

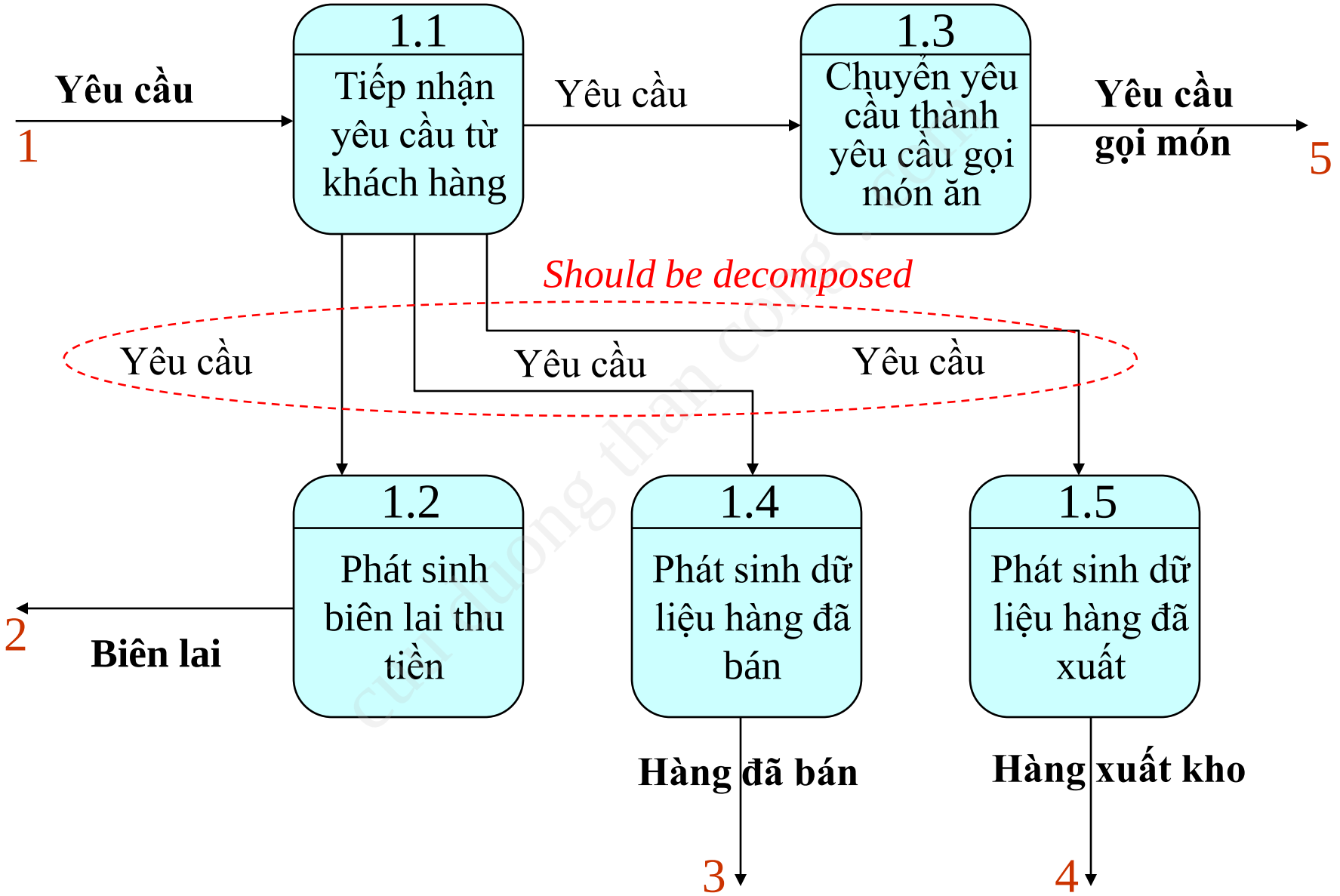
Hệ thống đặt món ăn của nhà hàng Hosier Burger được mô tả như sau:

1. Chức năng "tiếp nhận và xử lý yêu cầu gọi món": nhận yêu cầu từ khách hàng, chuyển yêu cầu gọi món đến nhà bếp, phát sinh biên lai thu tiền cho khách, tạo ra dữ liệu 'hàng đã bán' và 'hàng xuất kho' cho 2 chức năng 'cập nhật hồ sơ hàng bán' và 'cập nhật hồ sơ hàng tồn kho'.
2. Cập nhật hồ sơ hàng đã bán: cập nhật hồ sơ hàng đã bán theo đúng khuông mẫu dữ liệu của hồ sơ này.
3. Cập nhật hồ sơ hàng tồn kho: Cập nhật hồ sơ hàng tồn kho theo đúng khuông mẫu của hồ sơ này.
4. Phát sinh báo cáo quản lý: lấy dữ liệu hàng đã bán mỗi ngày từ hồ sơ hàng đã bán và dữ liệu hàng xuất kho mỗi ngày từ hồ sơ hàng tồn kho để in báo cáo cho người quản lý nhà hàng.

DFD-0 của hệ thống đặt món ăn



DFD-1 của xử lý 1.0

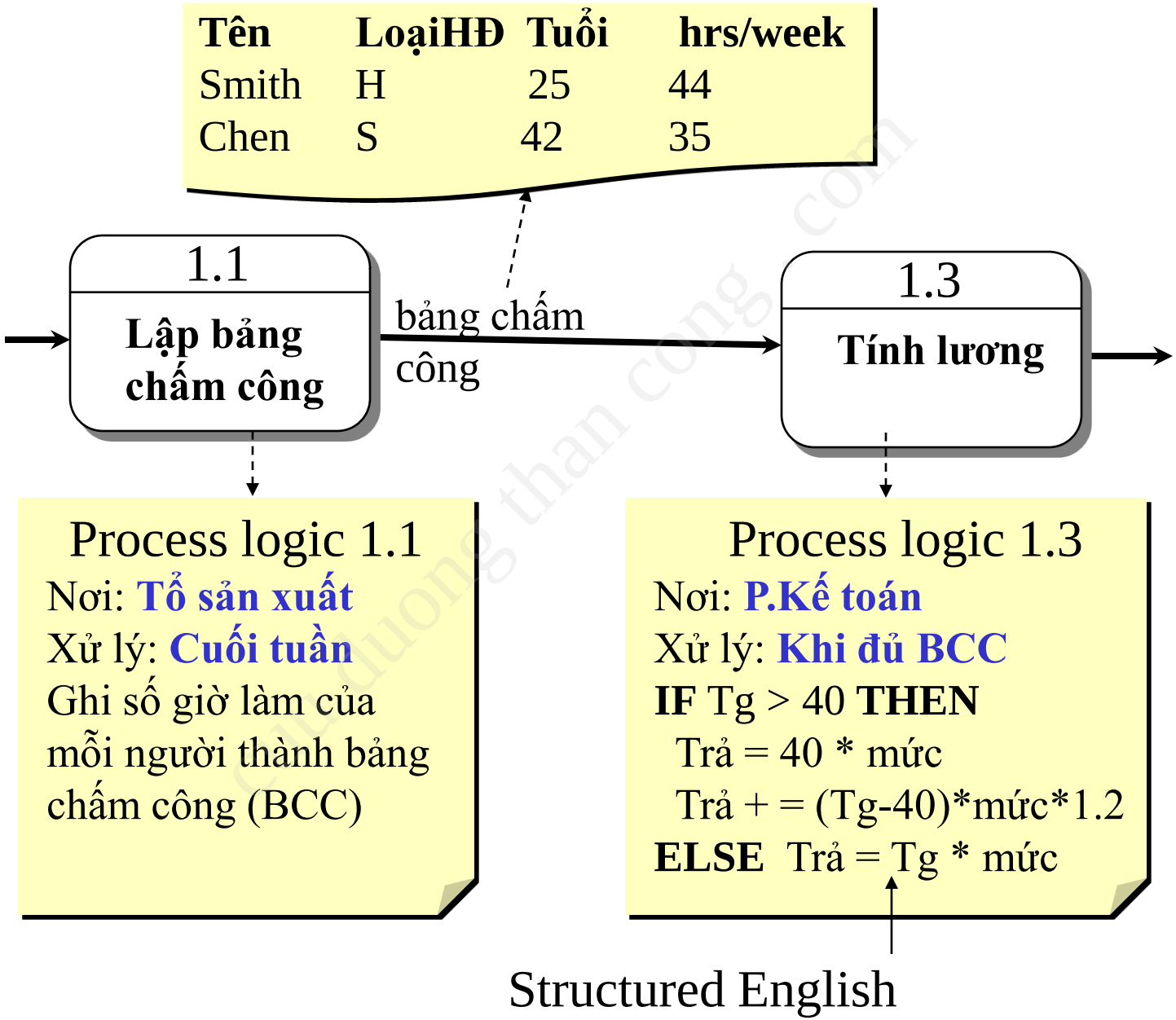


Processing Logic

- DFD trợ giúp hiểu cấu trúc xử lý của hệ thống, nhưng tên của các xử lý không đủ mô tả nội dung của nó, do đó ta cần dùng cấu trúc Processing Logic để mô tả chi tiết hơn cho các xử lý dựa theo các quy tắc quản lý của hệ thống.

1. *Ngôn ngữ tự nhiên có cấu trúc*: mô tả ngắn gọn về các quy tắc xử lý bằng ngôn ngữ tự nhiên giản lược (Việt / Anh)
2. *Bảng quyết định* và *Cây quyết định*: thể hiện các quy tắc xử lý "IF <điều kiện> THEN <xử lý>" bằng bảng hoặc lưu đồ rẽ nhánh nếu hệ thống có nhiều quy tắc xử lý phức tạp
3. *Từ điển dữ liệu*: là từ điển giải thích ý nghĩa của từng thành tố dữ liệu được các xử lý sử dụng hoặc tạo ra. Từ điển dữ liệu còn được dùng để liên kết các thành tố dữ liệu trong ERD với các dòng dữ liệu trong DFD.

Ngôn ngữ tự nhiên có cấu trúc

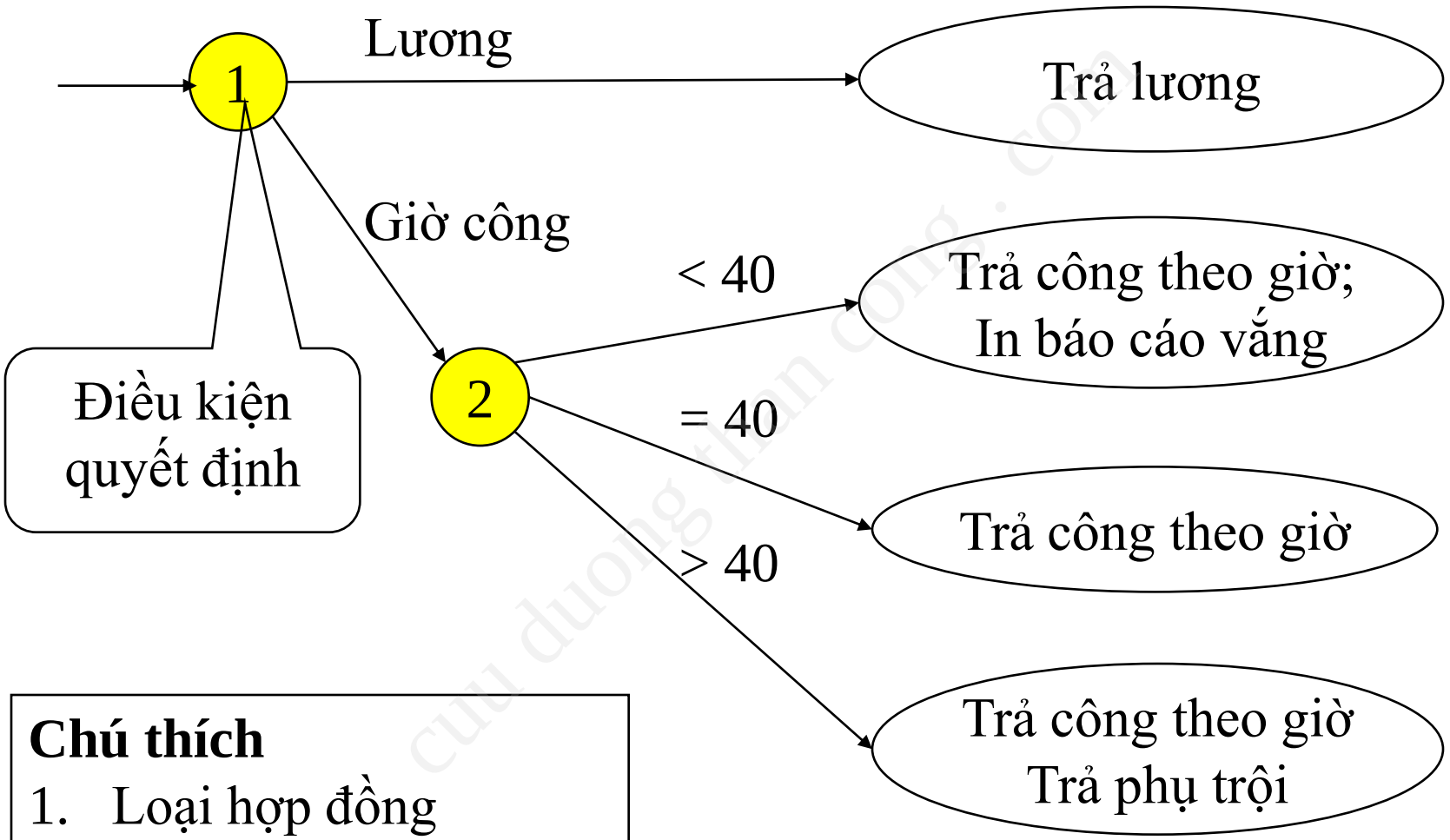


Bảng quyết định

ĐIỀU KIỆN	QUY TẮC ÁP DỤNG			
Loại hợp đồng	S	H	H	H
Số giờ làm việc	-	<40	40	>40
CÁC XỬ LÝ				
Trả lương (salary)	√			
Tính giờ công		√	√	√
Tính giờ phụ trội				√
In báo cáo vắng mặt		√		

S: Salary H: Hours

Cây quyết định



Chú thích

1. Loại hợp đồng
2. Số giờ công

Từ điển dữ liệu

➲ Là bảng mô tả chi tiết cho từng phần tử dữ liệu được dùng trong hệ thống. Thông thường, từ điển dữ liệu được lập thành một bảng gồm các cột như ví dụ sau:

Tên gọi	Ý nghĩa	Cấu trúc dữ liệu	Nguồn gốc
Bảng chấm công	Ghi số giờ công của từng nhân viên	Tên, tuổi, số giờ công trong tuần	Process 1.1
Tiền lương	Tiền trả theo giờ công trong tháng	Mức chuẩn + phụ trội	Process 1.3
Mức chuẩn	Mức tiền công trả trong định mức 40 giờ / tuần	Giờ công trong định mức * giá định mức	Process 1.3
Phụ trội	Mức trả vượt định mức 40 giờ / tuần	Giờ công vượt trội * giá định mức * 1.5	Process 1.3

Bài tập vẽ DFD

- Công ty Wonder Widget (WWC) có một hệ thống giám sát các *đơn đặt hàng* từ khách hàng. WWC lấy *mã số của các món hàng* được yêu cầu từ danh mục hàng (*Master Item Number List*), lấy *đơn giá* của món hàng từ hồ sơ giá (*Master Price List*), và kiểm tra *mức tín dụng* của khách hàng trong hồ sơ tín dụng (*Credit Rating File*). Các *phiếu đặt hàng được chấp nhận* sẽ lưu trong hồ sơ đặt hàng được chấp nhận (*Accepted Orders File*), các *phiếu đặt hàng bị từ chối* được lưu trong hồ sơ đặt hàng bị từ chối (*Rejected Orders File*). Hệ thống kho được kiểm tra để xem *số lượng hàng* có đủ cho các phiếu đặt hàng được chấp nhận không. Nếu đủ, *phiếu đặt hàng được phê duyệt* và gửi đến trung tâm giao hàng. Nếu không đủ, phiếu sẽ bị từ chối và đưa vào hồ sơ đặt hàng bị từ chối. Hồ sơ đặt hàng bị từ chối được dùng để phát sinh *báo cáo phiếu đặt hàng bị từ chối* cho giám đốc của WWC. Hãy vẽ lược đồ ngữ cảnh và DFD level 0 cho hệ thống.
- Các từ in nghiêng là nội dung dữ liệu, các từ gạch dưới là source hoặc sink. Hệ thống không giám sát việc giao hàng và thanh toán tiền.

Bài tập vẽ DFD

- Một cửa hàng bán quần áo muốn thiết lập một hệ thống giám sát bán hàng. Cửa hàng sử dụng bản copy của các biên lai thu tiền (có tên hàng, số lượng bán, đơn giá, thành tiền) được giao khách hàng để cập nhật hồ sơ hàng đã bán (Goods Sold File), hồ sơ hàng tồn kho (Inventory File), và hồ sơ doanh thu (Sales Total File). Cửa hàng sử dụng các hồ sơ này để phát sinh báo cáo đặt mua thêm hàng cho nhà kho và báo cáo cho người quản lý. Hãy vẽ lược đồ ngữ cảnh và DFD level 0 cho hệ thống.
- Chú ý: khi nội dung mô tả không đầy đủ thì lược đồ sẽ không được rõ ràng => quay lại khảo sát kỹ hơn.

Entity Relationship Diagram

- DFD chỉ ra *làm thế nào, ở đâu và khi nào* dữ liệu được xử lý, nhưng không chỉ ra *định nghĩa, cấu trúc và các quan hệ* của dữ liệu.
- Entity Relationship Diagram (ERD) là lược đồ thể hiện cấu trúc trừu tượng hóa của dữ liệu trong tổ chức, dựa trên khái niệm thực thể (entity) và quan hệ (relationship) giữa các thực thể, để nhằm thể hiện nội dung, ý nghĩa của dữ liệu trong hệ thống.

Thực thể (Entity)

➲ Nhân viên, Sinh viên, Môn học,… là các thực thể, là một khái niệm tổng quát hóa cho một nhóm các đối tượng (thể hiện, entity instance) trong thế giới thực có chung một số đặc điểm (thuộc tính). Vd: môn "PTTK", môn "CSDL" là các thể hiện của thực thể MônHọc.

1. *Thực thể xác thực* mô tả các đối tượng tồn tại thực sự trong thế giới thực: Xe đạp, xe hơi, nhà, quyển sách,…
2. *Thực thể chức năng* mô tả mục đích, chức năng, hoặc nhiệm vụ của con người, thiết bị hoặc tổ chức: Sinh viên, nhân viên, khách hàng, nhà kho,…
3. *Thực thể sự kiện* mô tả các sự kiện hoặc biến cố: biên nhận, biên bản họp, kỳ thi,…
4. *Thực thể quan hệ* mô tả các quan hệ giữa các đối tượng: Quản lý, đăng ký, hợp đồng,…

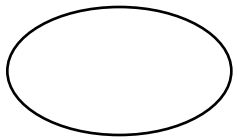
Thực thể (Entity)

- *Attribute* (thuộc tính) là đặc điểm chung của các đối tượng trong thực thể. Vd: MãNV, Tên, Địachỉ, KỹNăng là các thuộc tính được quan tâm khi nghĩ về thực thể NhânViên.
- Khóa là 1 hoặc kết hợp nhiều thuộc tính để phân biệt các đối tượng trong thực thể với nhau. Vd: MãNV là 1 thuộc tính dùng để phân biệt các nhân viên trong tập thực thể NhânViên. Nếu biết MãNV của 1 nhân viên, ta sẽ tìm được tên của nhân viên, địa chỉ và kỹ năng của nhân viên đó, dựa trên dữ liệu của thực thể NhânViên.
- Yêu cầu đối với khóa của thực thể:

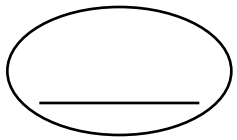
1. Không thay đổi giá trị khi thể hiện tương ứng còn tồn tại
2. Là thuộc tính không rỗng (Not Null)
3. Không chứa thuộc tính có cấu trúc. Mã vật tư dạng "kho,loại" là thuộc tính có cấu trúc (không nguyên tố).

Các ký hiệu dùng trong ERD

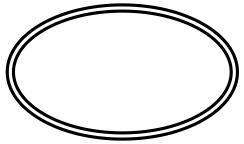
Entity



Attribute

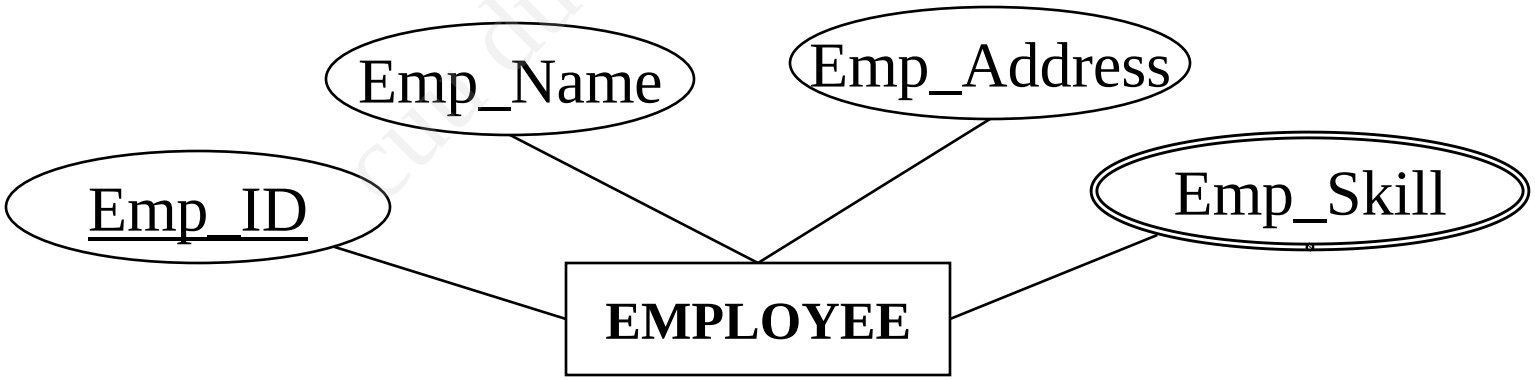


Key



Multivalue attribute

Ví dụ: một nhân viên có 1 mã nhân viên dùng để phân biệt. Cơ quan chỉ quan tâm quản lý tên nhân viên, địa chỉ nhà riêng, và các kỹ năng của từng nhân viên. Thực thể nhân viên được diễn tả như sau:

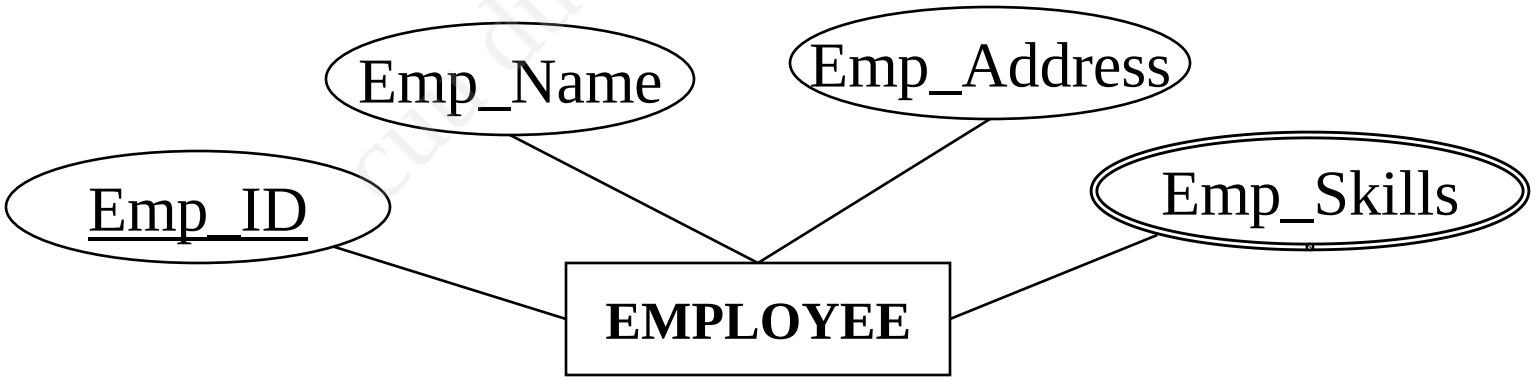


Thuộc tính đa trị (Multivalue Attribute)

➲ là một thuộc tính có nhiều giá trị được sử dụng đồng thời để mô tả cho một thể hiện của thực thể

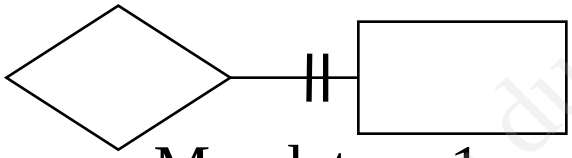
- Vd: thuộc tính "Skill" của 1 nhân viên. Một nhân viên thường có nhiều kỹ năng, skill sẽ có nhiều giá trị khác nhau

EMP_ID	EMP_Name	EMP_Skill
0210-67	Susan	Language
0210-67	Susan	Interpersonal

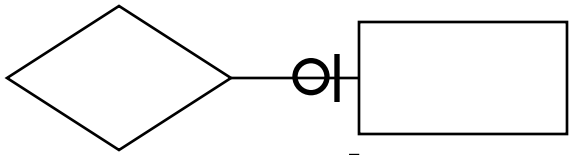


Quan hệ

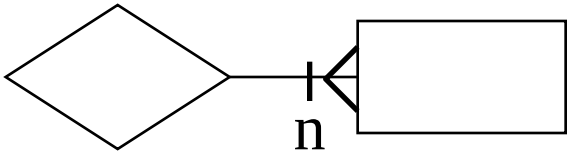
- Là mối liên kết giữa một hoặc nhiều thực thể để chỉ ra sự liên kết về nội dung (và ý nghĩa) giữa các thực thể trong mối liên kết. Ví dụ: "Mỗi SINH VIÊN đăng ký nhiều MÔN HỌC". Sự liên kết nội dung giữa thực thể Sinh viên và thực thể Môn học là việc đăng ký môn để học của mỗi sinh viên.
- Cardinality: là số thể hiện của thực thể B có thể (hoặc phải) liên kết với mỗi thể hiện của thực thể A. Vd: một sinh viên phải đăng ký học ít nhất là 1 môn, và nhiều nhất là 6 môn trong một học kỳ.



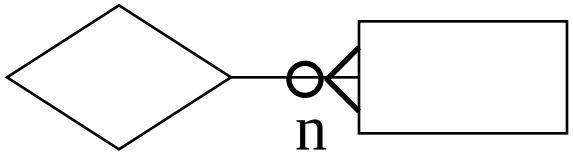
Mandatory 1



Optional 0 - 1

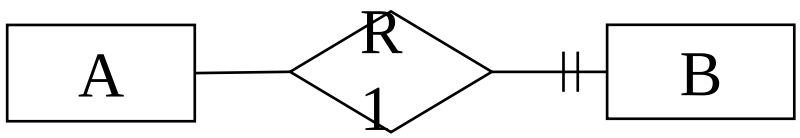
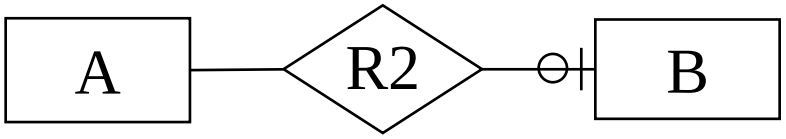
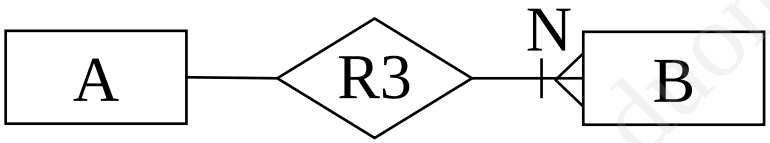
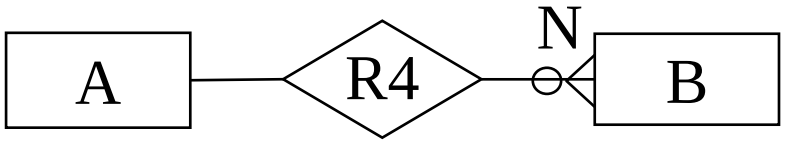


Mandatory Many



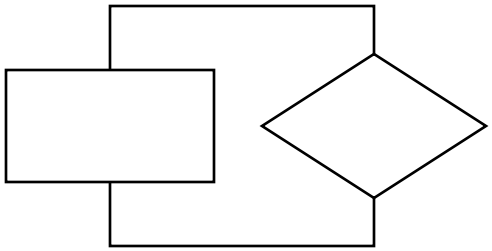
Optional 0 - Many

Số quan hệ (Cardinality)

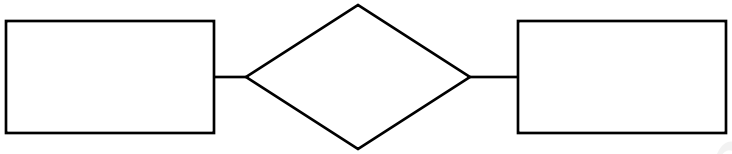
	Mỗi thể hiện của A có đúng 1 thể hiện tương ứng ở B theo quan hệ R1 (cardinality = [1,1]).
	Mỗi thể hiện của A chỉ có 1 thể hiện tương ứng ở B, hoặc không có thể hiện tương ứng theo quan hệ R2 (cardinality = [0,1]).
	Mỗi thể hiện của A có ít nhất là 1 và tối đa là N thể hiện tương ứng ở B theo quan hệ R3 (cardinality = [1,N]).
	Mỗi thể hiện của A có tối đa là N thể hiện tương ứng ở B, hoặc không có thể hiện tương ứng theo quan hệ R4 (cardinality = [0,N]).

Relationship

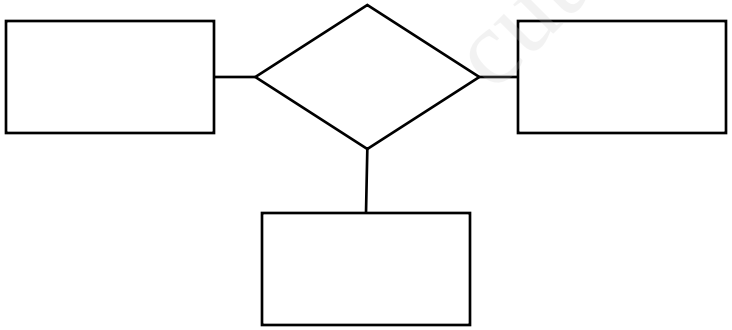
Có 3 loại quan hệ cơ bản giữa các thực thể:



Unary relationship: liên kết trên 1 thực thể

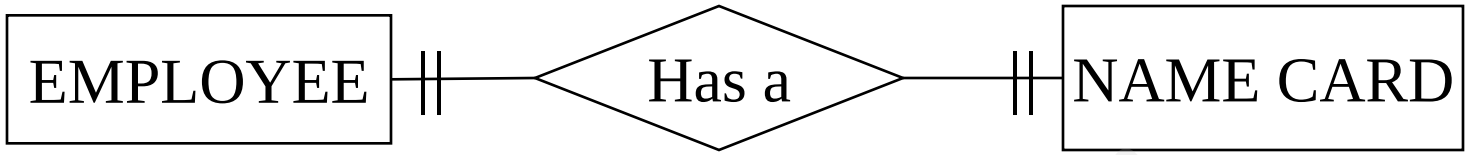


Binary relationship: liên kết 2 thực thể



Ternary relationship: liên kết 3 thực thể

Binary Relationship



One to One (1:1)

"Một nhân viên phải có (duy nhất) 1 bảng tên."

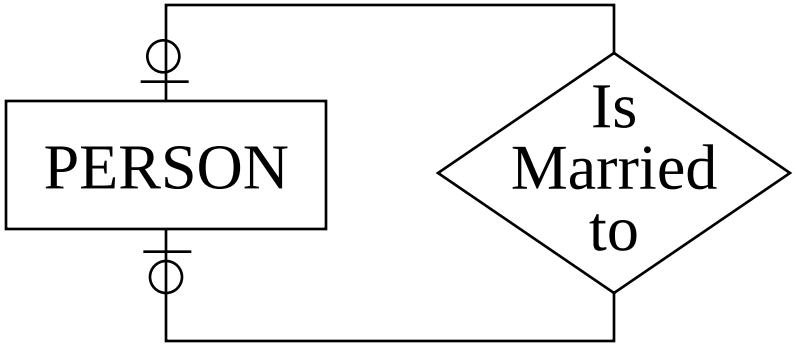
One to Many (1:N)

"Một dây chuyền sản phẩm phải chứa 1 hoặc nhiều sản phẩm. Một sản phẩm phải thuộc 1 dây chuyền sản xuất."

Many to Many (M:N)

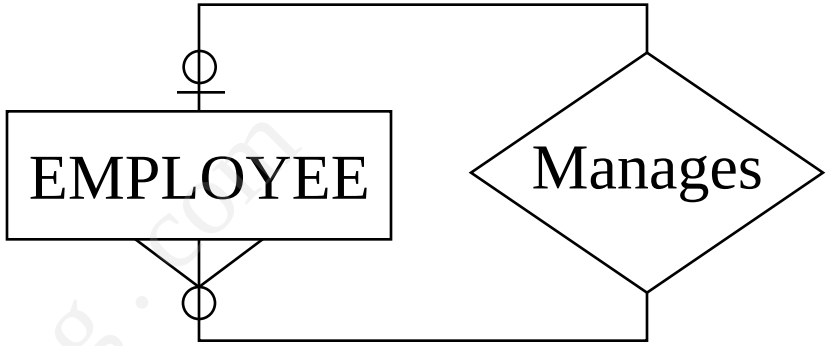
"Một sinh viên phải đăng ký 1 hoặc nhiều môn học. Một môn học có thể có nhiều sinh viên đăng ký, hoặc không có sinh viên đăng ký"

Unary Relationship



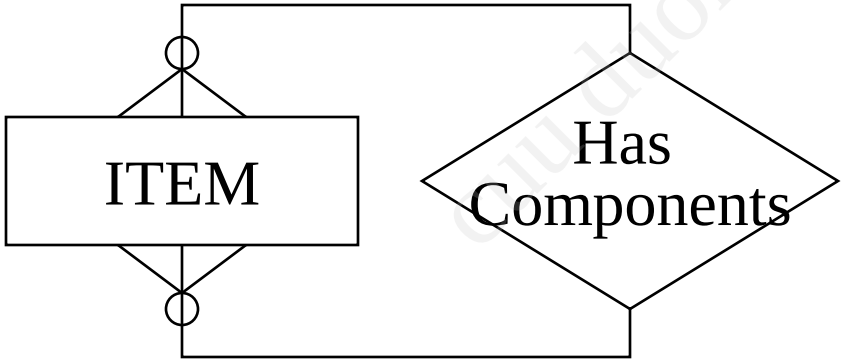
One to One

"Một người chỉ được kết hôn với một người khác"



One to Many

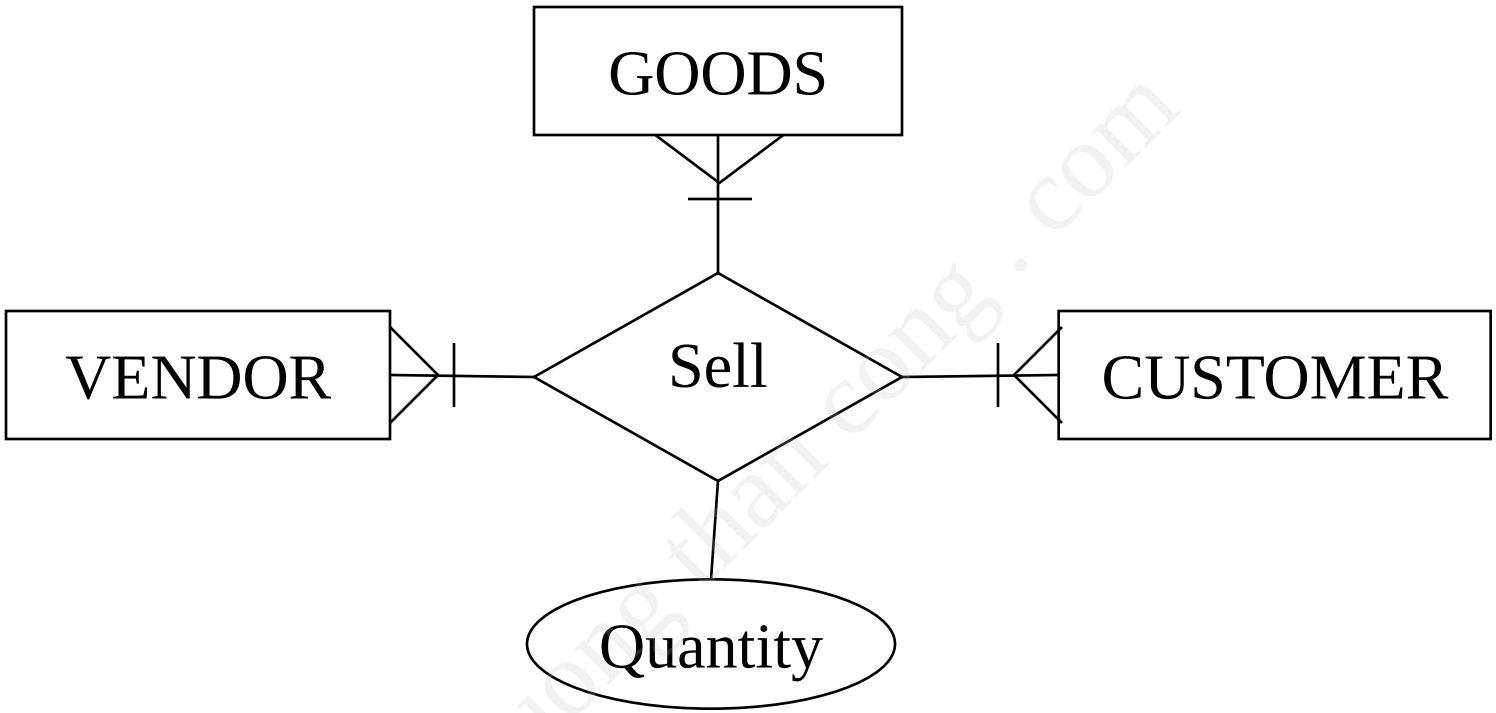
"Một nhân viên có thể quản lý nhiều nhân viên"



Many to Many

"1 Item có thể có nhiều thành phần, mỗi thành phần cũng là 1 Item có thể được sử dụng trong nhiều component khác"

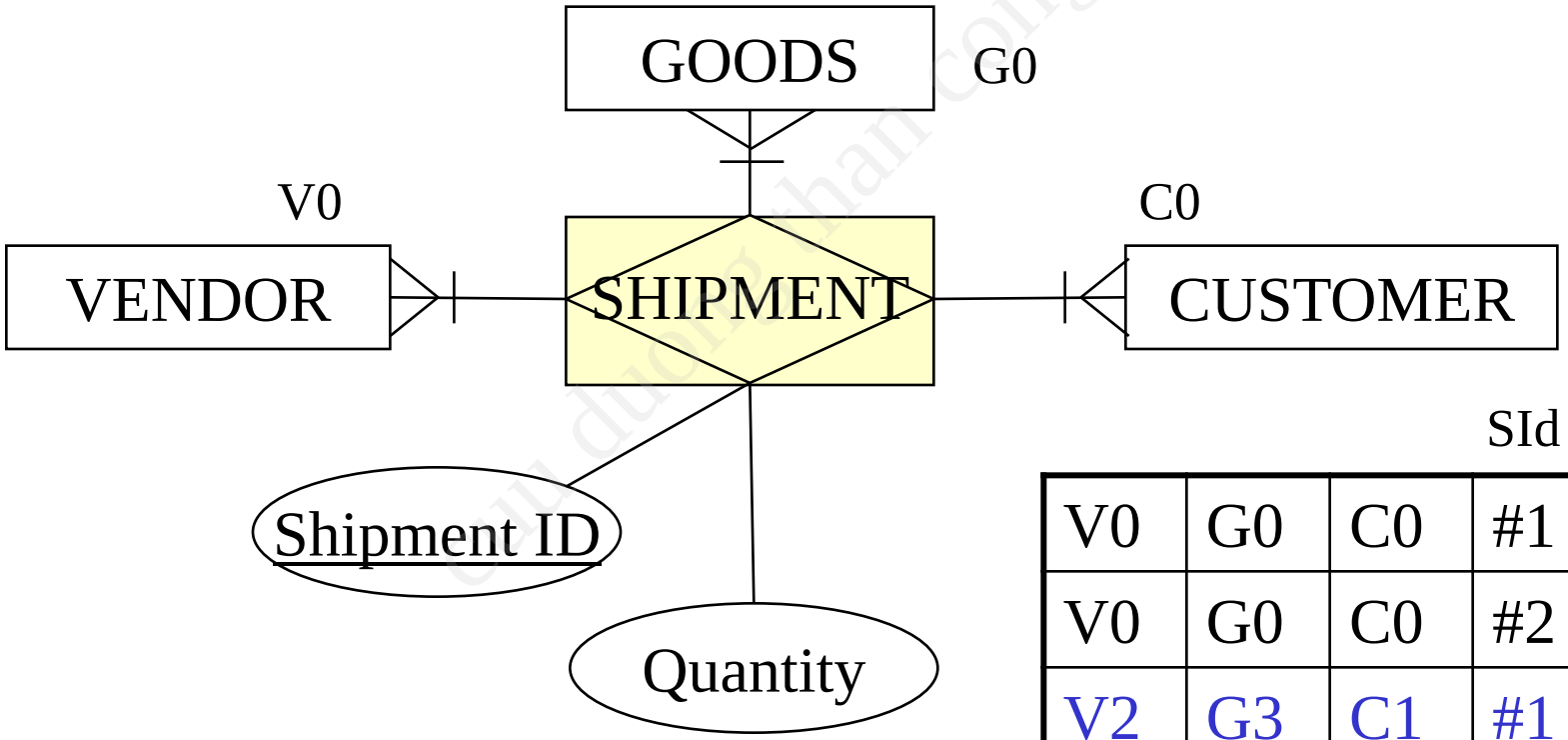
Ternary Relationship



"Một nhà cung cấp có thể bán nhiều mặt hàng cho nhiều khách hàng; khách hàng có thể mua nhiều hàng từ nhiều nhà cung cấp khác nhau. Quantity là số lượng của 1 mặt hàng được 1 khách hàng mua từ 1 nhà cung cấp"

Associative Entity (thực thể quan hệ)

- Là thực thể mô tả cho các mối liên kết giữa các đối tượng cụ thể tham gia vào quan hệ, và có thêm dữ liệu riêng trên từng mối liên kết giữa các đối tượng này. Thực thể quan hệ là thực thể chứa các mối quan hệ giữa các thể hiện của các thực thể tham gia vào quan hệ.



			SId	Qty
V0	G0	C0	#1	100
V0	G0	C0	#2	50
V2	G3	C1	#1	150
V2	G3	C1	#2	150

Thiết lập lược đồ ERD

1. *Định nghĩa các thực thể*, dựa trên vai trò, ý nghĩa của thực thể đối với hệ thống. Nên chọn danh từ để dùng làm tên cho thực thể, vd: MONHOC, SINHVIEN, KHOA,..
2. *Định nghĩa các quan hệ giữa các thực thể*. Tên của các quan hệ thường được diễn tả bằng động từ để chỉ các hành động, sự kiện liên kết các thể hiện trong các thực thể có quan hệ nhau.
3. *Xác định các thuộc tính của thực thể và quan hệ*. Thuộc tính của thực thể (hoặc quan hệ) là những đặc tính mà tất cả các thể hiện của thực thể (hoặc quan hệ) đều có. Thêm thuộc tính để tăng tính mô tả, hoặc để có thê dữ liệu phân biệt các thể hiện. Bỏ bớt thuộc tính nếu chúng dư thừa hoặc không liên quan đến vai trò, ý nghĩa của thực thể trong hệ thống.
4. *Xác định cardinality cho mỗi quan hệ.*

Bài tập vẽ ERD

1. Mỗi nhân viên thuộc 1 phòng, một phòng có ít nhất 1 hay nhiều nhân viên. Không có nhân viên nào không thuộc phòng nào. Mỗi nhân viên có mã số, tên và mức lương. Mỗi phòng có số phòng, tên và nhiệm vụ.
2. Mỗi giảng viên (có mã số, tên, chuyên môn) dạy nhiều môn học (có mã mh, tên, nội dung, số tiết) trong các phòng học khác nhau (phân biệt bằng số phòng).
3. Mỗi sinh viên sau mỗi lần thi một môn học sẽ có một điểm xác định cho lần thi đó.
4. Các quầy hàng có mã số và vị trí, bán nhiều mặt hàng có mã số và tên mặt hàng; mỗi mặt hàng còn có một gía niêm yết tại quầy hàng đó.
5. Mỗi bệnh nhân nội trú có mã số và tên được bệnh viện cấp cho một giường ngủ có mã số giường đặt trong phòng bệnh có mã số phòng và tên phòng. Khi nhận và trả giường thì ngày nhận và ngày trả cũng được bệnh viện ghi vào hồ sơ quản lý.

Bài tập vẽ ERD

Hàng hóa được *xuất cảng* trong các container và mỗi container được định danh bởi số của container, đích đến và kích cỡ của nó. Đại lý vận chuyển (Shipping Agent) chịu trách nhiệm *gom nhóm* các container vào một lô hàng và cho lô hàng một số định danh và giá trị của lô. Các con tàu, xác định bởi số của con tàu, quốc gia đăng ký và tải trọng, *thực hiện* các chuyến hải trình, mỗi chuyến hải trình *mang* một số lô hàng đi đến đích của chúng. Mỗi chuyến hải trình được cho số hải trình. Hãy vẽ lược đồ ERD chỉ ra tất cả các thực thể, thuộc tính, và quan hệ.

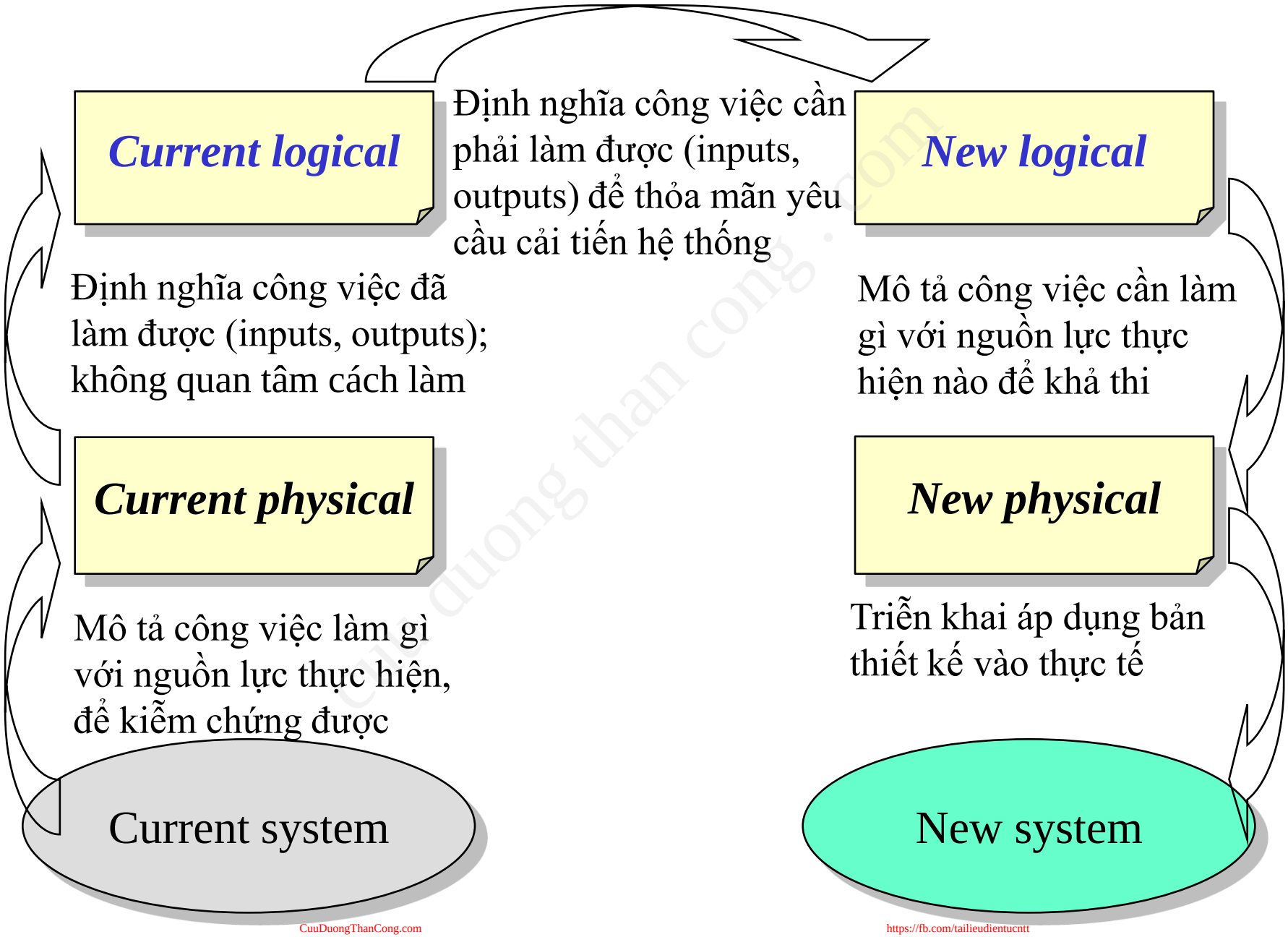
Để vẽ các ERD, chúng ta cần xác định các thực thể trước, sau đó là quan hệ giữa các thực thể, và cuối cùng là các thuộc tính cho thực thể và quan hệ. Các từ gạch dưới là các thực thể, và các động từ in nghiên là các quan hệ giữa các thực thể. Trong mô tả không nêu rõ cardinality (số quan hệ), do đó số quan hệ có thể được cho bằng cách suy diễn hợp lý.

2. Phân tích hệ thống

Sau khi khảo sát và hiểu hiện trạng, người phân tích viên cần chỉ ra ưu khuyết điểm của nó đối với mục đích của tổ chức, và đi đến kết luận hệ thống mới cần phải làm gì.

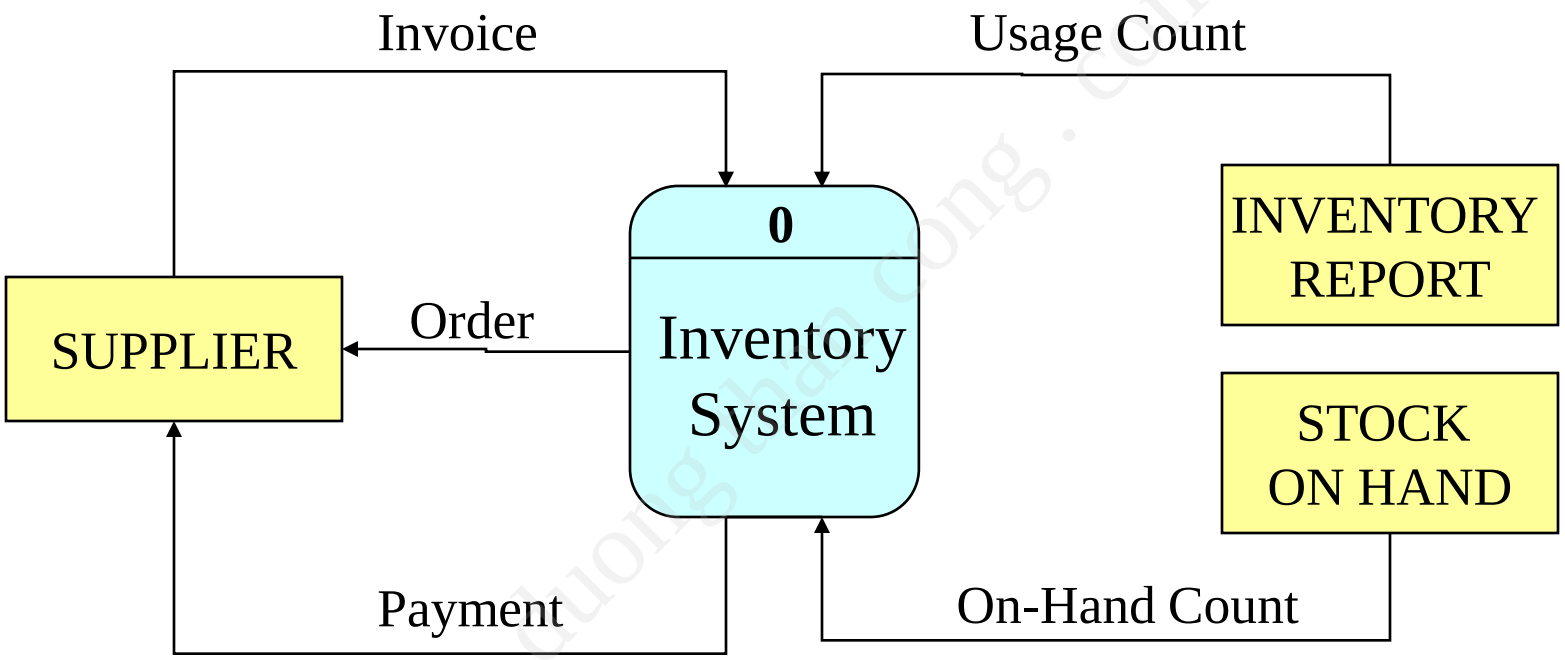
- Yêu cầu cơ bản đối với phân tích gồm
  1. Nhận thức rõ ràng và đầy đủ những yêu cầu đối với hệ thống
  2. Mô tả cho nhận thức bằng phương pháp phù hợp
  3. Đối chiếu giữa nhận thức và thực tế
  4. Lập hồ sơ đặc tả yêu cầu và chia sẽ với người sử dụng
- Để nhận thức rõ về bản chất của hiện trạng và làm rõ các bài toán, người phân tích viên cần sử dụng các **mô hình** (models) diễn đạt thay cho các phát biểu mô tả dài dòng.
- Công việc chính của phân tích viên là phân tích các dòng dữ liệu thể hiện các **quy tắc quản lý** đang được áp dụng trong tổ chức.

4 loại DFD trong phân tích hệ thống

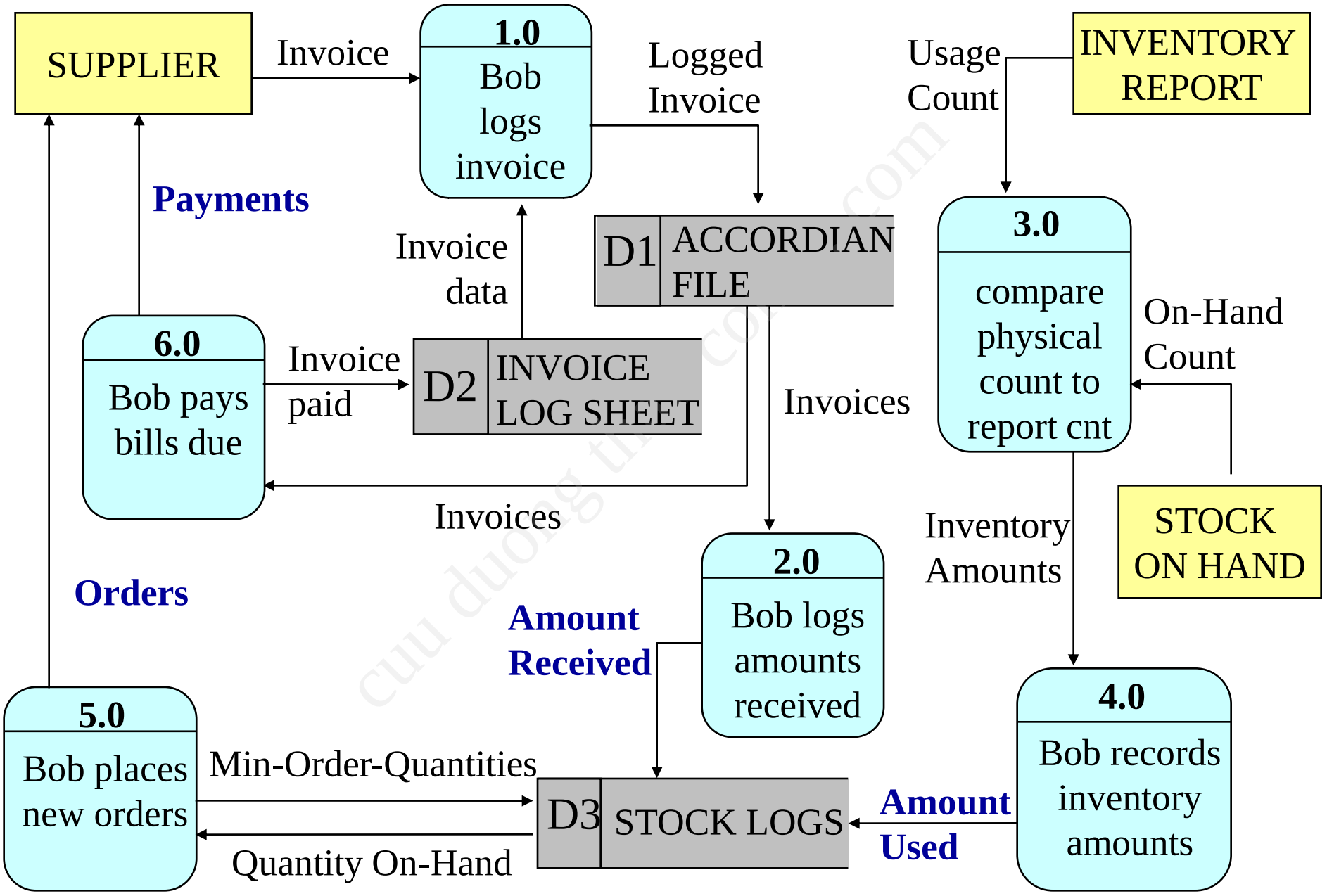


Current physical: Context Diagram

Ngữ cảnh của hệ thống quản lý kho nhà hàng Hoosie Burger



Current physical: Level 0 DFD



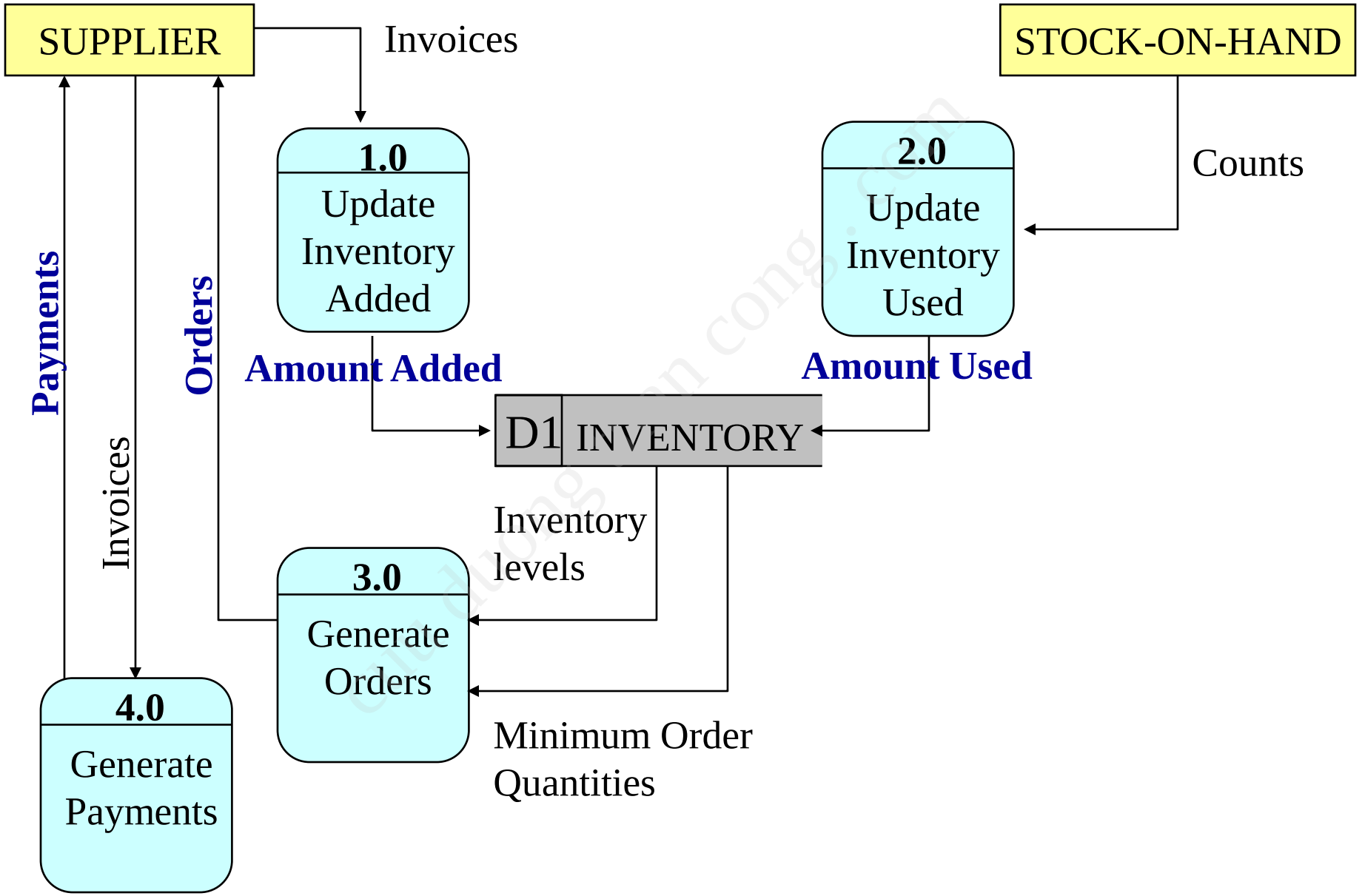
Phân tích các dòng dữ liệu

- Trong ví dụ hệ thống quản lý kho của nhà hàng Hoosie Burger, lược đồ DFD vật lý chỉ ra rằng 2 dòng dữ liệu từ kho đến process 3.0 (Usage count và On-hand count) đều từ một nguồn là kho => chỉ nên dùng một dòng dữ liệu.
- Hệ thống cần quản lý trực tiếp nguyên liệu đưa vào kho bằng hóa đơn từ nhà cung cấp (invoice) và giám sát mức độ tồn kho bằng cách đếm (on-hand count) để quản lý.
- Như vậy Process 3.0 & dòng dữ liệu Usage count trong DFD-0 vật lý không còn cần thiết.
- Data store D2 lưu số liệu hóa đơn để theo dõi thanh toán tiền hàng (không có dòng dữ liệu đi ra) và Data store D1 chỉ để lưu tạm số lượng hàng nhập kho cho Data store D3 nên D1,D2 không cần thiết.

Physical DFD to Logical DFD

- **Các xử lý chính**: để phục vụ cho mục đích "Vật tư/hàng hóa trong kho phải đủ để sử dụng" gồm:
  1. Kiểm soát tất cả những thứ đưa vào kho (1.0, 2.0) => Các xử lý 1.0 & 2.0 gọi chung là xử lý "Update Inventory Added"
  2. Kiểm soát tất cả những thứ lấy ra khỏi kho (3.0, 4.0) => xử lý 3.0 và 4.0 gọi chung là xử lý "Update Inventory Used"
  3. Đặt mua thêm, nếu thiếu (5.0) => "Generate orders"
  4. Trả tiền (6.0) => "Generate payments"
- **Dữ liệu chính:** Gồm **Amount received**, **Amount used**, **Orders**, và **Payments**

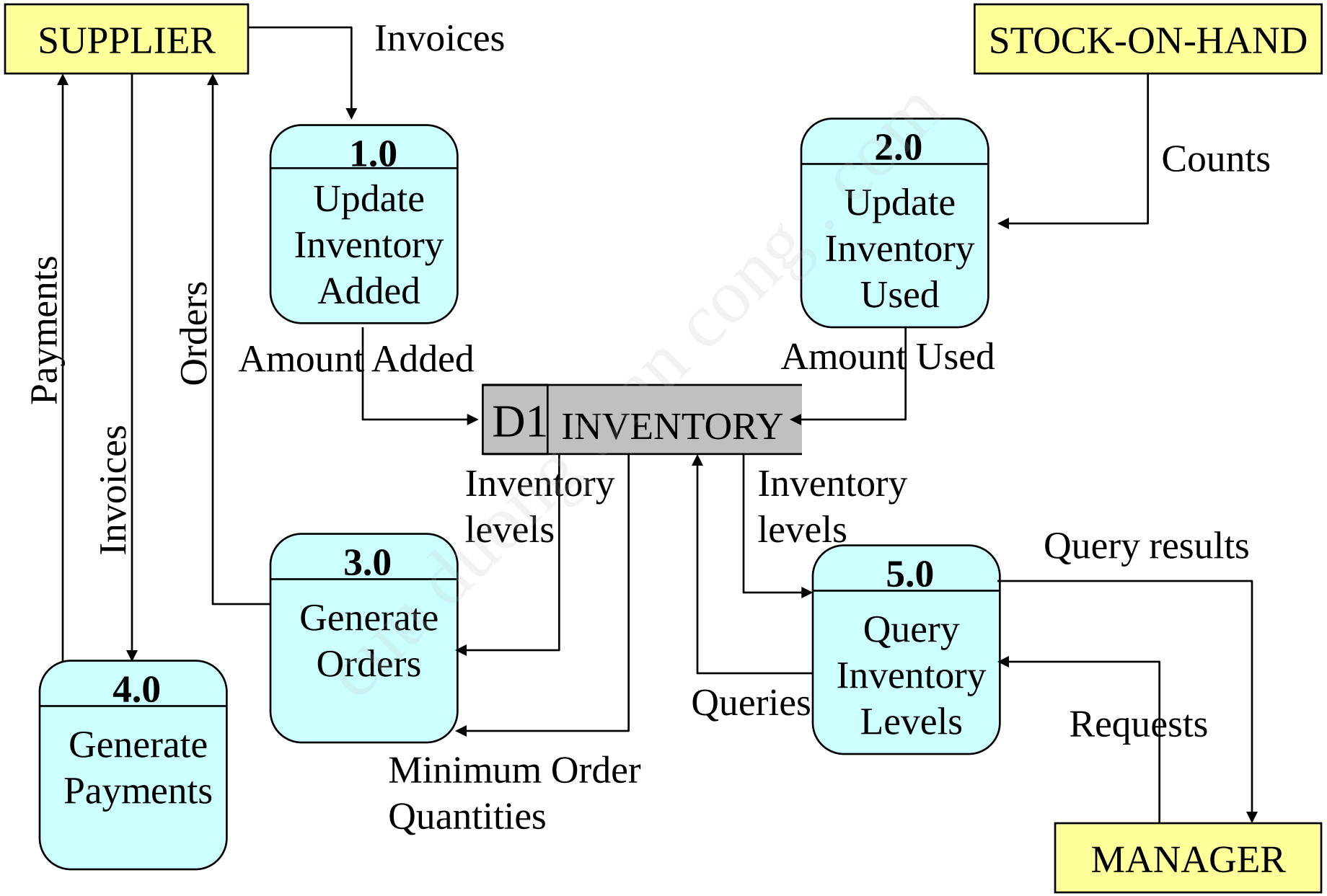
Current Logical: Level 0 DFD



Current Logical DFD to New Logical DFD

- Qua khảo sát, User (Bob) mong muốn rằng:
  1. Dữ liệu chuyến hàng mới về cần phải đưa vào hệ thống xử lý tự động (không dùng log sheet).
  2. Tự động phát sinh đơn đặt hàng.
  3. Truy vấn số lượng kho bất kỳ lúc nào.
- Mục đích của New Logical DFD là chỉ ra các chức năng nào cần làm, và các dòng dữ liệu nào cần sử dụng.
- Yêu cầu 1 & 2 không cần thay đổi vì DFD đã được trừu tượng hóa - không phụ thuộc vào cách thực hiện thủ công hay tự động.
- Yêu cầu 3 cần được thêm vào lược đồ (process 5.0)

New logical : Level 0 DFD



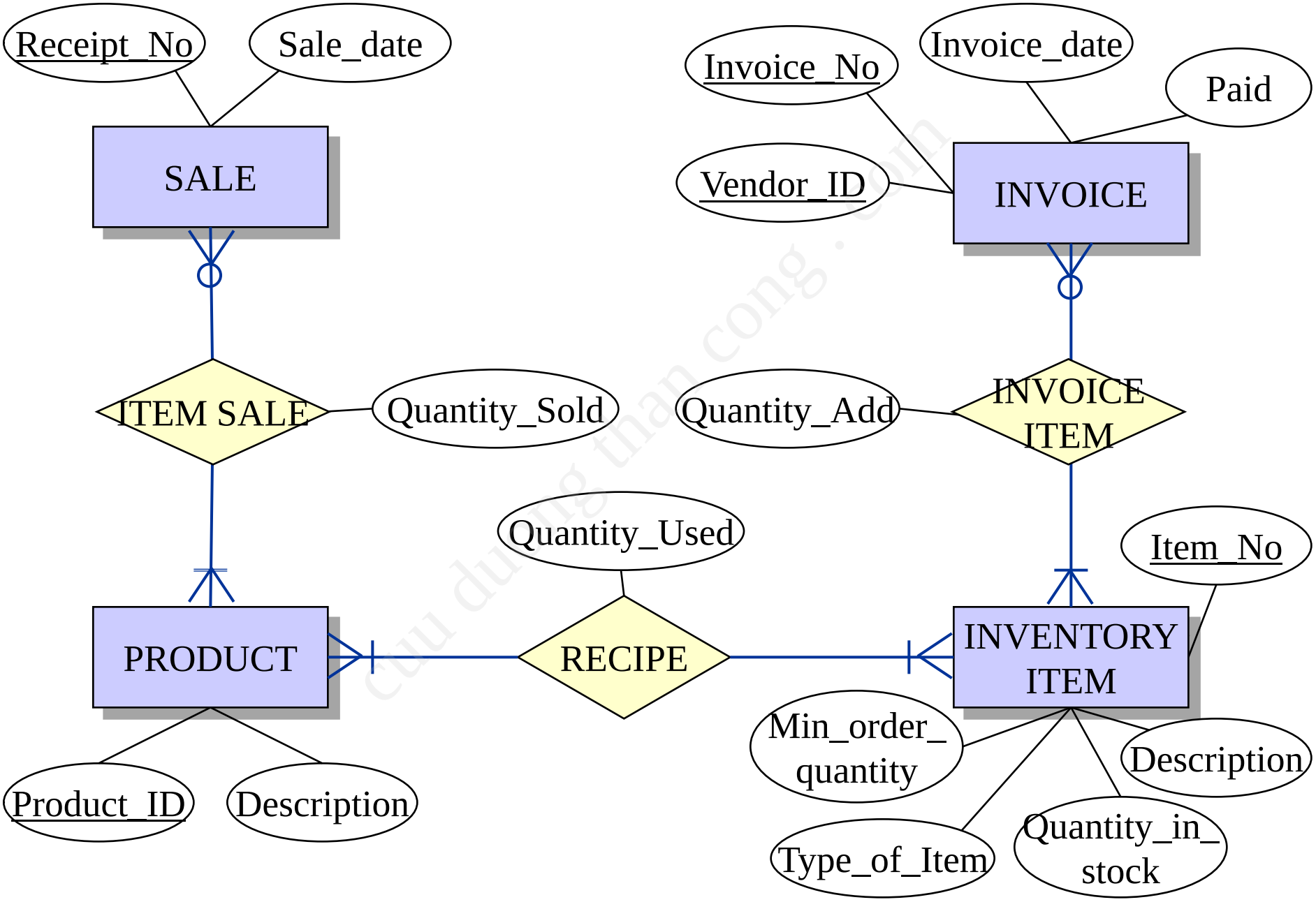
ERD cho hệ thống quản lý kho của Hoosie burger

- Các thay đổi trong kho phát sinh bởi 2 sự kiện: Khi nhận được hàng từ nhà cung cấp (và hóa đơn kèm theo từ nhà cung cấp), và tiêu thụ nguyên liệu khi làm thức ăn bán cho khách hàng. Mỗi hóa đơn (INVOICE) có mã số của nhà cung cấp (Vendor_ID), số của hóa đơn (Invoice_number), ngày phát hành (Invoice_date), tiền trả (Paid). =>INVOICE là một thực thể diễn tả sự kiện nhận hàng về kho.
- Mỗi mặt hàng trong kho (INVENTORY ITEM) có mã số hàng (Item_number), đặc tả (Item_description), số lượng tồn (Quatity_in_stock), loại (Type_of_Item) và số lượng tối thiểu để đặt hàng lại (Minimum_order_quantity). => INVENTORY là thực thể mô tả kho.
- Mỗi hóa đơn đều có kèm danh mục các mặt hàng được giao (INVOICE ITEMs) chỉ ra rằng nhà cung cấp đã giao thêm một số mặt hàng nằm trong danh sách các mặt hàng của kho (IVENTORY ITEMs). Mỗi mặt hàng được giao có kèm theo số lượng giao (Quatity_added). => INVOICE ITEMs là một quan hệ giữa INVOICE và INVENTORY ITEMs, thuộc tính 'Quatity_added' là thuộc tính của quan hệ này.

ERD cho hệ thống quản lý kho của Hoosie burger

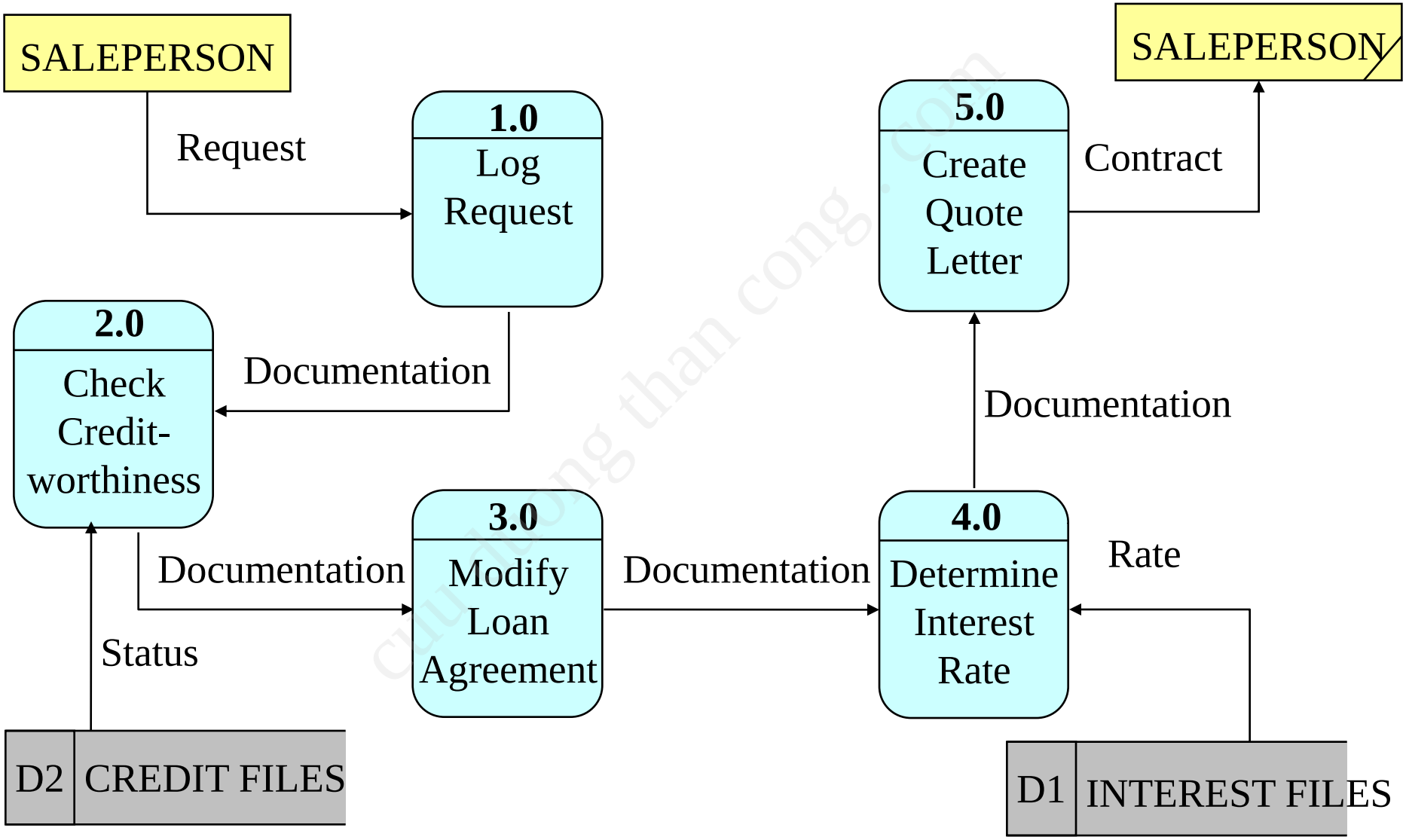
- Các mặt hàng trong kho sẽ được sử dụng như nguyên liệu để làm ra sản phẩm (PRODUCT). Mỗi sản phẩm có mã số sản phẩm (Product_ID) và mô tả sản phẩm (Product_description).
- Vì mỗi sản phẩm được làm từ nhiều loại nguyên liệu khác nhau nên nhà hàng duy trì các công thức nấu ăn (RECIPE) để xác định sản phẩm nào cần (những) nguyên liệu gì và khối lượng bao nhiêu. =>công thức nấu ăn (RECIPE) là một quan hệ giữa PRODUCT và INVENTORY ITEM. Liều lượng cần sử dụng (Quantity_Used) của mỗi loại nguyên liệu trong công thức là một thuộc tính của quan hệ RECIPE.
- Mỗi lần bán hàng là một SALE có số biên nhận (Receipt_number), và ngày bán (Sale_date). SALE là một thực thể mô tả sự kiện bán hàng.
- Khách hàng có thể mua nhiều sản phẩm cùng một lúc; do đó mỗi lần bán hàng sẽ có nhiều món hàng (sản phẩm) được liệt kê thành danh mục trong hóa đơn bán hàng (ITEM SALE), mỗi mục có số lượng bán (Quantity_sold). => ITEM SALE là một quan hệ giữa thực thể SALE và thực thể PRODUCT, trong đó Quantity_sold là thuộc tính của quan hệ này.

ERD cho hệ thống quản lý kho của Hoosie burger



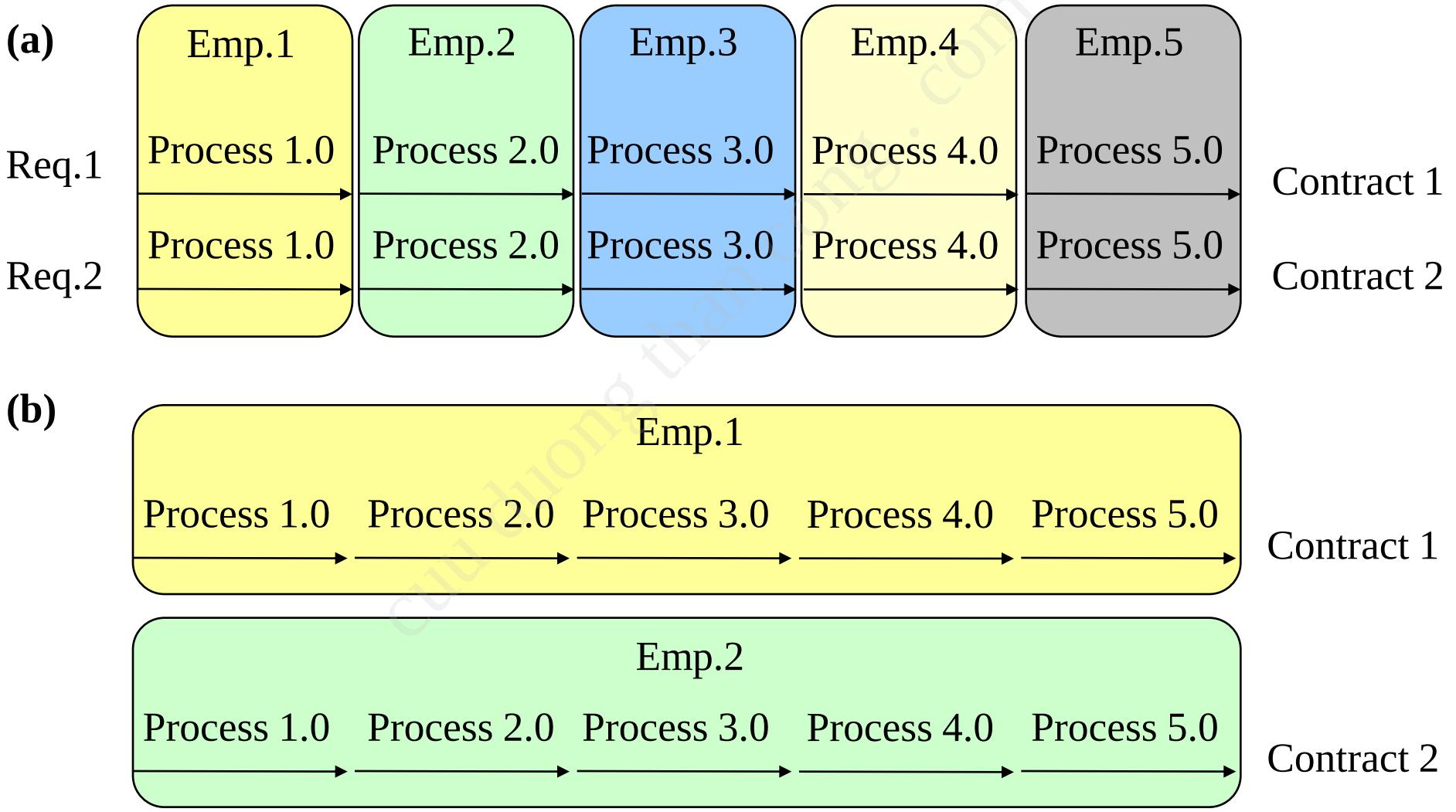
Sử dụng DFD để phân tích BPR (IBM 1993)

Hiện trạng

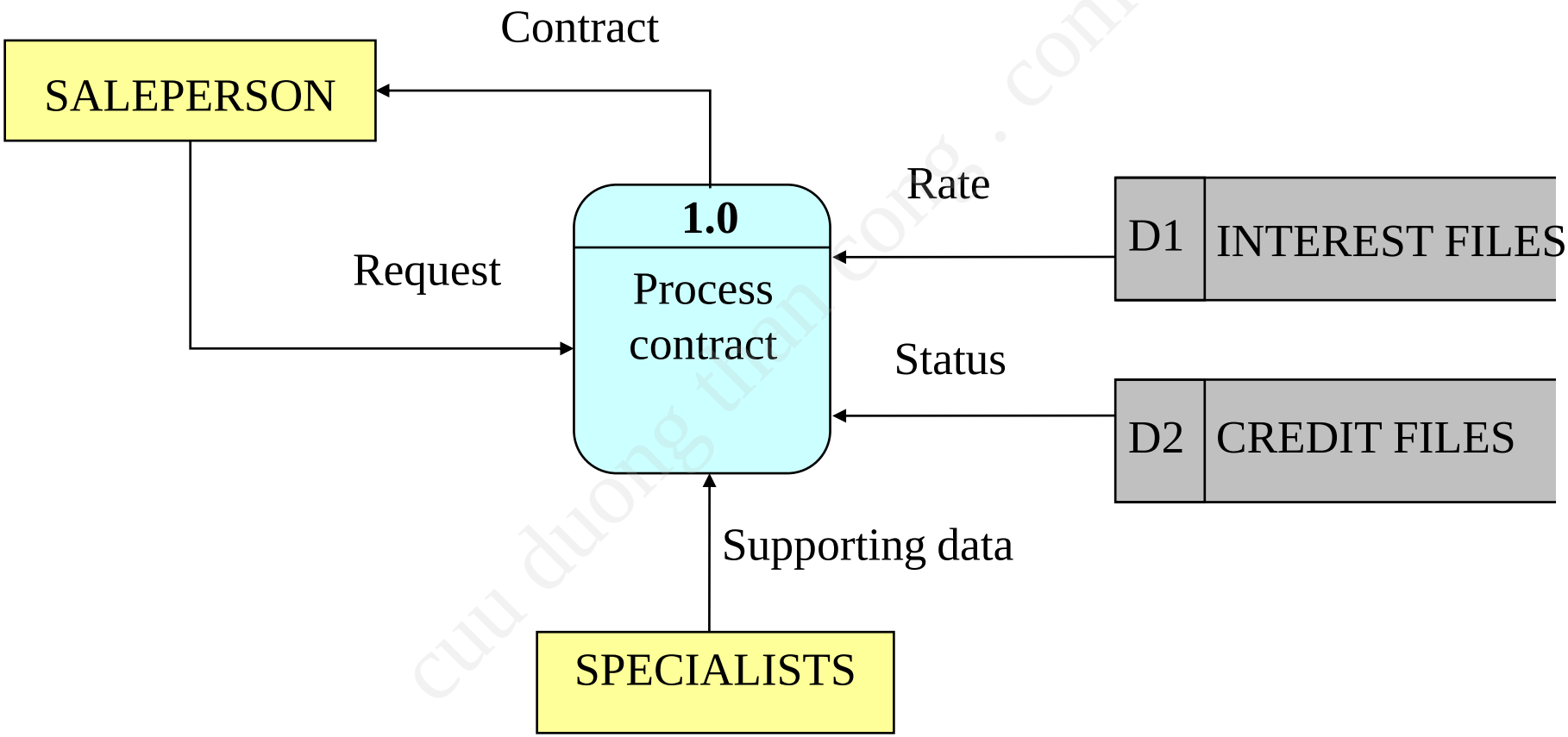


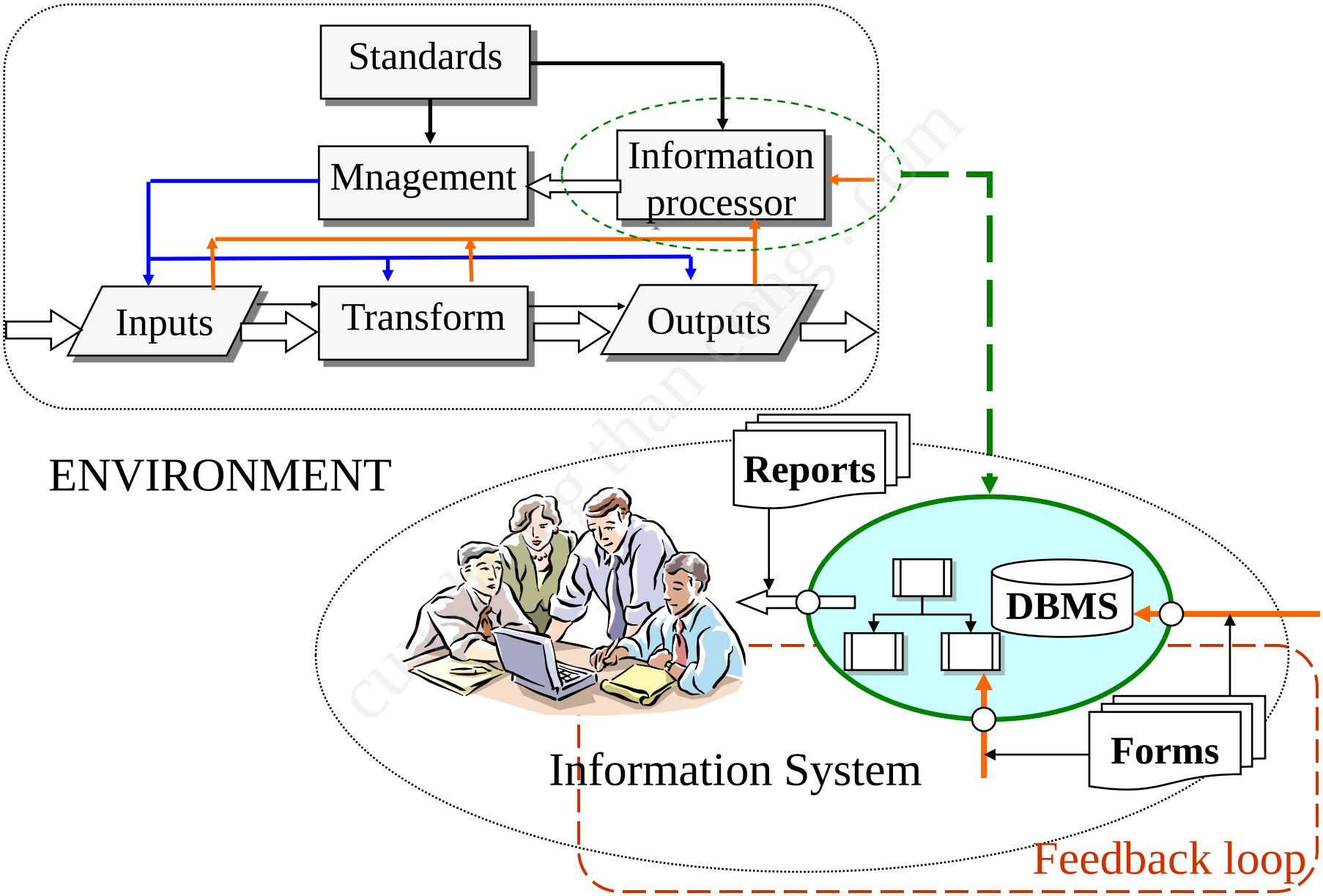
Sử dụng DFD để phân tích BPR (IBM 1993)

Phân tích



Kết quả




Standards
Mnagement
Information processor
Inputs
Transform
Outputs
ENVIRONMENT
Reports
DBMS
Forms
Information System
Feedback loop

1. ***Bảo vệ*** hệ thống khỏi các tác tác động không mong muốn từ môi trường
2. ***Lọc*** bỏ các nội dung vào/ra không cần thiết
3. ***Mã hóa*** và ***giải mã*** các nội dung vào/ra
4. ***Phát hiện lỗi*** và ***sửa lỗi***
5. ***Lưu trữ tạm thời*** thông tin, dữ liệu vào/ra bằng vùng đệm.
6. ***Tổng hợp*** và ***chuyển đổi*** dữ liệu sang khuông mẫu cần thiết cho hệ thống (form) hoặc cho hệ thống khác (report).

Prev | Next | First | Last | New | Update | Cancel

Customer Information

Number: 1273 [Search] Phone: 19087541358

Name: Contemporary Design

Order list (23 MAY 1998)

Item #	Product	U.Price	Amount	Price
1	Hotdog – Size U	0.5	5	2.5

[Add item] [Delete item]

Total 2.5

Tương tác trên form

1. Nội dung và thứ tự nhập liệu phải phù hợp với thông tin trên các loại tài liệu bằng giấy mà người sử dụng đang có.
   - Thời điểm có đủ tài liệu cho từng form
2. Dữ liệu phải có khuôn mẫu và được kiểm tra (validation) khi nhập liệu (Dữ liệu phải thuộc miền giá trị hợp lệ)
3. Dữ liệu và các control nhập liệu phải nhất quán.
4. Thiết kế linh hoạt
   - Có feedbacks cho người sử dụng(Về trạng thái của thông tin,Thông báo nhắc, Errors / warning)
   - Có cung cấp các trợ giúp cần thiết: Thông tin trợ giúp (User guide, Online helps, references) và Chức năng trợ giúp (Search, sort, template)

Điều khiển truy cập dữ liệu

➲ Để bảo đảm an ninh và hạn chế sai sót.

1. Giới hạn phạm vi thao tác trên dữ liệu bằng thiết kế "View" (tập hợp dữ liệu con của hệ thống mà người sử dụng được phép đọc ghi trên đó).
2. Thiết lập các quy tắc xác quyền và phân quyền
   - ***Xác quyền***: người sử dụng là ai.
   - ***Phân quyền***: cho phép / cấm thao tác trên dữ liệu.
3. Thiết lập các thủ tục mã hóa dữ liệu
   - Bảo mật đường truyền
   - Bảo mật khi lưu trữ

Điều khiển toàn vẹn dữ liệu

1. Kiểm soát trạng thái của biến
   - Trạng thái không xác định (vừa mới khởi tạo)
   - Trạng thái có dữ liệu (hợp lệ / không hợp lệ)
2. Các kỹ thuật điều khiển trên giao diện (form)
   - Giá trị mặc định (default value)
   - Khuôn mẫu nhập liệu (picture/template)
   - Range (miền giá trị hợp lệ)
3. Các kỹ thuật xử lý bên trong hệ thống phần mềm
   - Kiểm tra quan hệ dữ liệu trong CSDL
   - Kiểm tra theo các quy tắc quản lý (business rules)

Tương tác trên report

Navigation

Pine Valley Furniture

Detail customer account information

Subject

Page : 2 of 2

Today : 11-OCT-98

Update

Customer number: 1273

Name: Contemporary Design

Columns

DATE	PURCHASE	PAYMENT	CURRENT BALANCE
21-JAN-98	(22,000.00)		(22,000.00)
21-JAN-98		13,000.00	(9,000.00)
02-MAR-98	(16,000.00)		(25,000.00)
02-MAR-98		15,500.00	(9,500.00)
23-MAY-98		5,000.00	(4,500.00)
12-JUN-98	(9,285.00)		(13,785.00)
12-JUN-98		3,785.00	(10,000.00)
21-SEP-98		5,371.65	(4,628.35)
YTD-SUMARY	**(47,285.00)**	**42,656.65**	**(4,628.35)**

(support details)

High light

Conclusion

Cấu trúc phần mềm

- Cấu trúc cần phải mềm dẻo, kiểm soát được, và hỗ trợ cập nhật bổ sung sau này
  - Phân rã hệ thống xử lý thành các module đơn giản, dễ hiểu
  - Mỗi module chỉ thực hiện 1 chức năng hoặc mỗi module chỉ điều khiển một số ít module con
- Hạn chế phát sinh dư thừa mã lệnh
  - Sử dụng lại (Re-use) những gì đã có (giải pháp, công nghệ, thư viện chuẩn hoặc tự tạo)
- Bảo vệ các mô đun trong việc cải tiến phần mềm

1. Abstraction

- Là sự lọc bỏ chi tiết riêng biệt để các đối tượng khác nhau được nhìn nhận giống nhau (làm bộc lộ những đặc điểm tương tự nhau).
- Các mức trừu tượng giúp chúng ta dần dần hiểu được vấn đề cần giải quyết của hệ thống và giải pháp trong thiết kế.
  - Hiểu=liên kết được các ý niệm trong vấn đề, giải pháp
- Sau khi hiểu hệ thống, giải pháp (thiết kế) tổng quát được thể hiện lại chi tiết sâu hơn, đến mức thấp nhất là các mã nguồn chương trình.
- Trong bản thiết kế, chúng ta làm việc với các khái niệm trừu tượng về thủ tục, trừu tượng về dữ liệu và trừu tượng trong cấu trúc điều khiển.

Abstraction - example

Rearrange array L in non-decreasing order — Level-1

DO WHILE I is between 1 and (length of L)-1:
 Set LOW to index of smallest value in L(I), ..., L(length of L)
Interchange L(I) and L(LOW) when need.
END DO

— Level-2

```
DO WHILE I  is between 1 and (length of L)-1
  Set LOW to current value of I
    DO WHILE J is between I+1 and (length of L)-1:
      IF L(LOW) is greater than L(J)
        THEN set LOW to current value of J
      ENDIF
    ENDDO
  Set TEMP to L(LOW)
  Set L(LOW) to L(I)
  Set L(I) to TEMP
ENDDO
```

— Level-3

2. Pattern (mẫu)

- Mẫu "*là 1 giải pháp chung cho 1 vấn đề phổ biến trong một ngữ cảnh cho trước*" (SWEBOK), như thư viện hàm chuẩn, các giải thuật phổ biến.
- Các pattern được phát triển trên một số vấn đề thường gặp đã được giải quyết một cách hoàn hảo; từ đó kiến thức xử lý được đúc kết thành pattern để áp dụng cho các vấn đề tương tự (bối cảnh giống nhau + vấn đề giống nhau).
- Thiết kế dựa trên Pattern gồm 2 bước:
  1. Định nghĩa cách tiếp cận tổng quát để giải quyết bài toán,
  2. Phân hoạch phạm vi bài toán đến mức đủ nhỏ để áp dụng pattern
- Sử dụng Pattern giúp rút ngắn được thời gian thiết kế, tránh sai sót và có kết quả cao nhờ sử dụng lại những mẫu đã có.

- Phân rã hệ thống: bắt đầu từ mô tả chức năng tổng quát, tạo ra các thành phần xử lý để xem làm thế nào mà chức năng của hệ thống liên kết được với các thành phần xử lý này (các mô-dun).
- Mô-đun được thiết kế tốt khi:
  - Tất cả các inputs và outputs đều cần thiết cho chức năng của nó;
  - Tất cả các outputs được tạo ra từ hoạt động của mô-đun trên các inputs (*không có outputs nào không qua xử lý trong mô-dun*).
  - Không gây hiệu ứng lề (*side effects*).
- Các mô-đun liên kết nhau theo cấu trúc phân cấp chức năng xử lý (hướng cấu trúc), hoặc dịch vụ (hướng đối tượng).

Tính độc lập của module

- Tính độc lập của mô-đun là khả năng cho phép sửa đổi trên từng mô-đun mà không phải sửa lại các mô-đun khác, ie: mỗi mô-đun có chức năng riêng, và không có quá nhiều ràng buộc với các mô-đun khác trong việc thực hiện công việc chung và công việc riêng của nó.
- Sự ràng buộc này xuất phát từ các mối quan hệ nội tại đang được thừa nhận (sử dụng) bên trong hệ thống.
- Tính độc lập của mô-đun được xem xét ở hai khía cạnh: độ phụ thuộc (**coupling**) và độ gắn kết (**cohesion**).

- **Coupling** đo mức độ phụ thuộc giữa các mô-đun trong việc thực hiện nhiệm vụ của từng mô đun bên trong hệ thống.
- Vì hệ thống phần mềm thiết kế gồm nhiều thành phần liên kết nhau nên chắc chắn sẽ có sự phụ thuộc giữa các mô-đun trong hệ thống.
- Nếu mức độ phụ thuộc giữa 2 mô đun là thấp thì sẽ dể cập nhật – sửa mỗi mô đun vì không bận tâm về mô-đun kia. Như vậy, hệ thống sẽ càng mềm dẻo nếu mức độ coupling giữa các mô đun càng thấp:
  - ✓ *(Low and best)* *Data coupling*
  - ✓ *Stamp coupling*
  - × *(Loose)* *Control coupling*
  - × *Common coupling*
  - × *(Worst)* *Content coupling*

1. **Data coupling**: dữ liệu dùng chung được cung cấp giá trị, giá trị dữ liệu nguyên thủy không được phép thay đổi (vd: hàm với đối số kiểu giá trị)
2. **Stamp coupling**: được phép truy cập đọc/ghi trên dữ liệu được chuyển giao (vd: hàm với đối số kiểu tham chiếu – chuyển giao địa chỉ thay vì giá trị của biến)
3. **Control coupling**: dữ liệu được dùng như cờ hiệu để điều khiển nội dung xử lý của mô đun nhận dữ liệu – cũng là mô đun được gọi chạy:

   UpdateFile (ADD);

   UpdateFile (CHANGE);

   UpdateFile (DEL);

4. **Common coupling**: các mô đun có toàn quyền dùng chung dữ liệu toàn cục → cần quy ước ngầm khi thao tác (tạo mới, cập nhật, sử dụng) trên dữ liệu này.

   Public filename = "vars.dat";

   UpdateFile; // có sử dụng filename để cập nhật

   DeleteFile; // có sử dụng filename để xóa file

5. **Content coupling**: Một mô-đun tạo một tham chiếu đến bên trong cấu trúc của mô-đun khác:

modun1: mov ax, 5	Call **modun1**
….	….
modun2: mov bx, 7	Call **modun2**
….	
return	

- **Cohesion** xác định mức độ hợp tác, gắn kết giữa các mô-đun trong việc hợp tác thực hiện nhiệm vụ chung của chúng bên trong hệ thống, giúp cho người thiết kế có ý thức giảm bớt "sự tùy tiện" trong cách xây dựng và sửa hệ thống.
- Có 7 mức độ gắn kết trong hệ thống (càng cao càng tốt)
  - ✓ *(High, best)* *Functional*
  - ✓ *Sequential*
  - ✓ *Communication*
  - × *Procedural*
  - × *Temporal*
  - × *Logical*
  - × *(Low)* *Coincidental*

Cohesion

1. **Functional**: toàn bộ mô-đun đều là để thực hiện một và chỉ một chức năng của mô-đun đó, và không thể thiếu bất cứ phần nào.
2. **Sequential**: nếu kết xuất từ một mô đun trở thành đầu vào của mô đun khác thì ta nói các mô-đun đó có mức độ gắn kết tuần tự (trình tự xử lý là quan trọng, và có dữ liệu chung giữa các mô-đun: output – input)
3. **Communication**: có nhiều mô-đun cùng thao tác trên 1 tập dữ liệu chung, thứ tự xử lý không quan trọng.

4. **Procedural**: gồm nhiều môđun khác nhau được thực hiện một cách có trình tự, nhưng không có dữ liệu chung giữa các mô đun này.

   Ví dụ: kết thúc transaction trước, khởi động bộ đếm và sau đó mới thực hiện transaction kế tiếp.

5. **Temporal**: Nhiều công việc khác nhau được gộp vào cùng trong cùng 1 mô-đun chỉ do thói quen (Cả dữ liệu lẫn thứ tự xử lý đều không quan trọng)

   Vd: khởi động biến toàn cục khi bắt đầu hoặc xoá khi kết thúc thủ tục.

6. **Logical** (service type): nhiều công việc được gom vào trong một mô-đun để không viết lại các mã lệnh dùng chung. Ví dụ: gom các chức năng Add-Record, Update-Record, Delete-Record vào trong mô-đun UpdateFile để dùng chung lệnh Open-File, Close-File:

```
UpdateFile (para)
    Open-File ("filename");
    case para = ADD: Add-Record;
    case para = CHANGE: Update-Record;
    case para = DEL: Delete-Record;
    Close-File();
```

7. **Coincidence**: các chức năng, xử lý hay dữ liệu chỉ thêm vào rất "tình cờ", không liên hệ với nhau.

1. Chuyển đổi từ ERD sang các bảng quan hệ (table) và các quan hệ giữa các bảng (relationship), để phù hợp với các hệ quản trị CSDL phổ biến hiện nay là loại Relational Database.
2. Thiết lập các cơ chế ràng buộc toàn vẹn dữ liệu để ngăn chặn các thao tác có thể gây sai sót trên CSDL
3. Chuẩn hóa các bảng quan hệ để tránh trùng lặp dữ liệu
4. Mã hóa hoặc nén dữ liệu lưu trữ để giữ bí mật thông tin hoặc giảm bớt không gian lưu trữ dữ liệu.
5. Tổ chức lưu trữ thuận tiện truy xuất và sao lưu phục hồi khi có sự cố, hư hỏng trên CSDL

Bảng quan hệ

➲ Bảng quan hệ là một cấu trúc bảng gồm cột và hàng để mô tả thực thể

- Mỗi cột ứng với mỗi thuộc tính của thực thể
- Mỗi hàng ứng với mỗi thể hiện của thực thể

Vd: EMPLOYEE2 (Emp_ID, Name, Dept_Name, Salary, Course_Title, Date_Complete). Emp_ID, Course_Title là một khóa chính kết hợp từ 2 thuộc tính của bảng quan hệ.

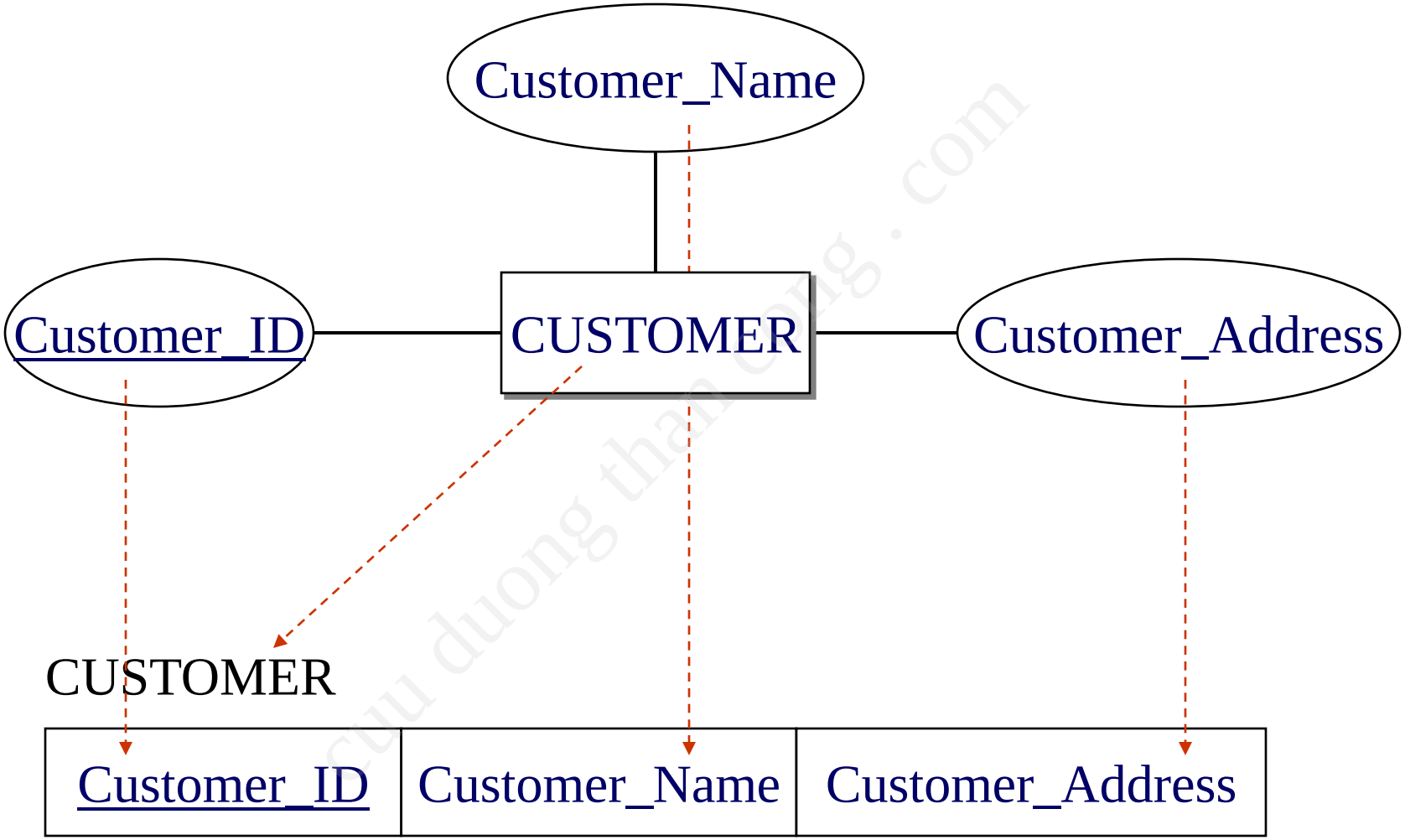
EMPLOYEE2

Emp_ID	Name	Dept_Name	Salary	Course_Title	Date_Completed
100	Margaret Simpson	Marketing	48,000	SPSS	6/19/200X
100	Margaret Simpson	Marketing	48,000	Surveys	10/7/200X
140	Alan Beeton	Accounting	52,000	Tax Acc	12/8/200X
110	Chris Lucero	Info Systems	43,000	SPSS	1/12/200X
110	Chris Lucero	Info Systems	43,000	C++	4/22/200X
190	Lorenzo Davis	Finance	55,000		
150	Susan Martin	Marketing	42,000	SPSS	6/19/200X
150	Susan Martin	Marketing	42,000	Java	8/12/200X

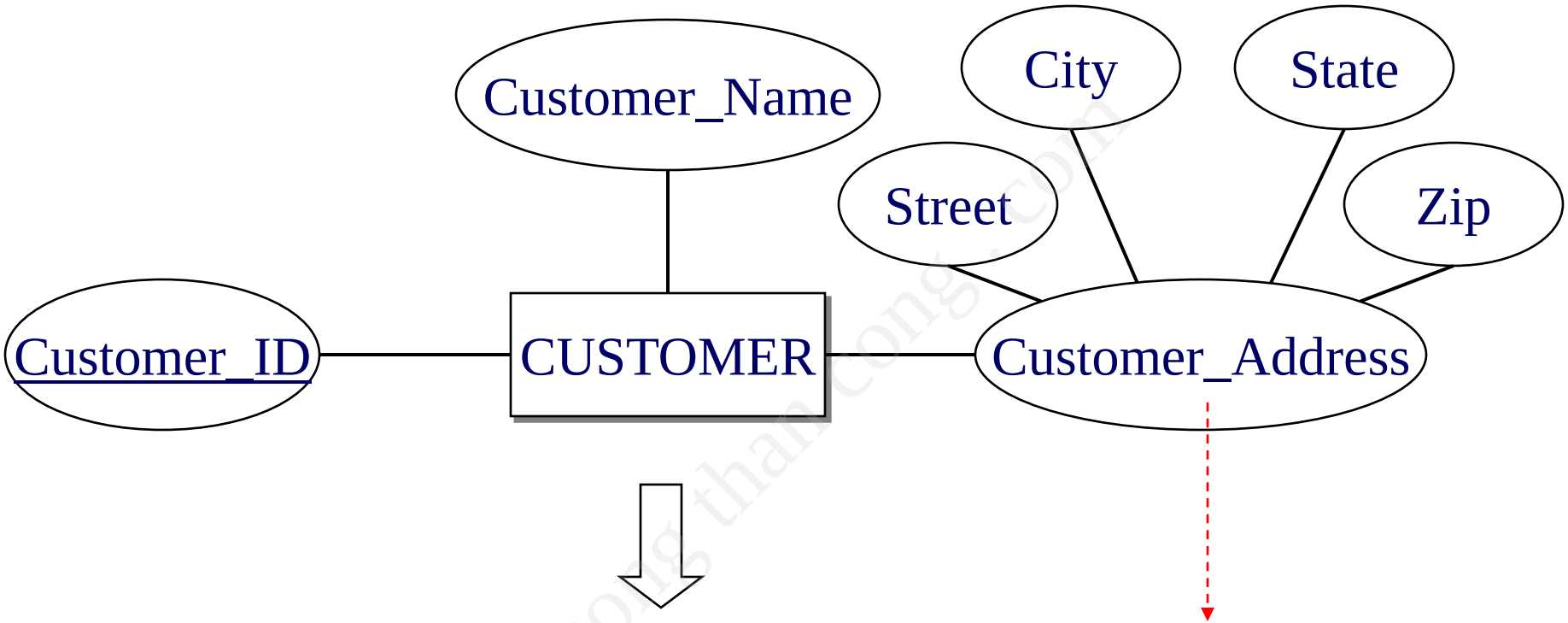
Bảng quan hệ

- Không phải tất cả các bảng dữ liệu đều là quan hệ; nó phải thoả mãn các điều kiện sau:
  1. Mỗi bảng có 1 tên duy nhất
  2. Mỗi thuộc tính trong bảng có 1 tên duy nhất.
  3. Mỗi dòng là duy nhất. Không thể có 2 dòng có tất cả các cột đều giống nhau.
  4. Mỗi giá trị của thuộc tính là nguyên tố. Thuộc tính 'composite' (có cấu trúc) không phải là nguyên tố vì nó kết hợp nhiều thuộc tính. Vd: Địa_chỉ = Số_nhà + Đường + Phường + Quận + Tp
- Các bảng quan hệ được tạo ra từ việc chuyển đổi lược đồ ERD sang bảng.

Thực thể sang bảng: 1.Single attribute



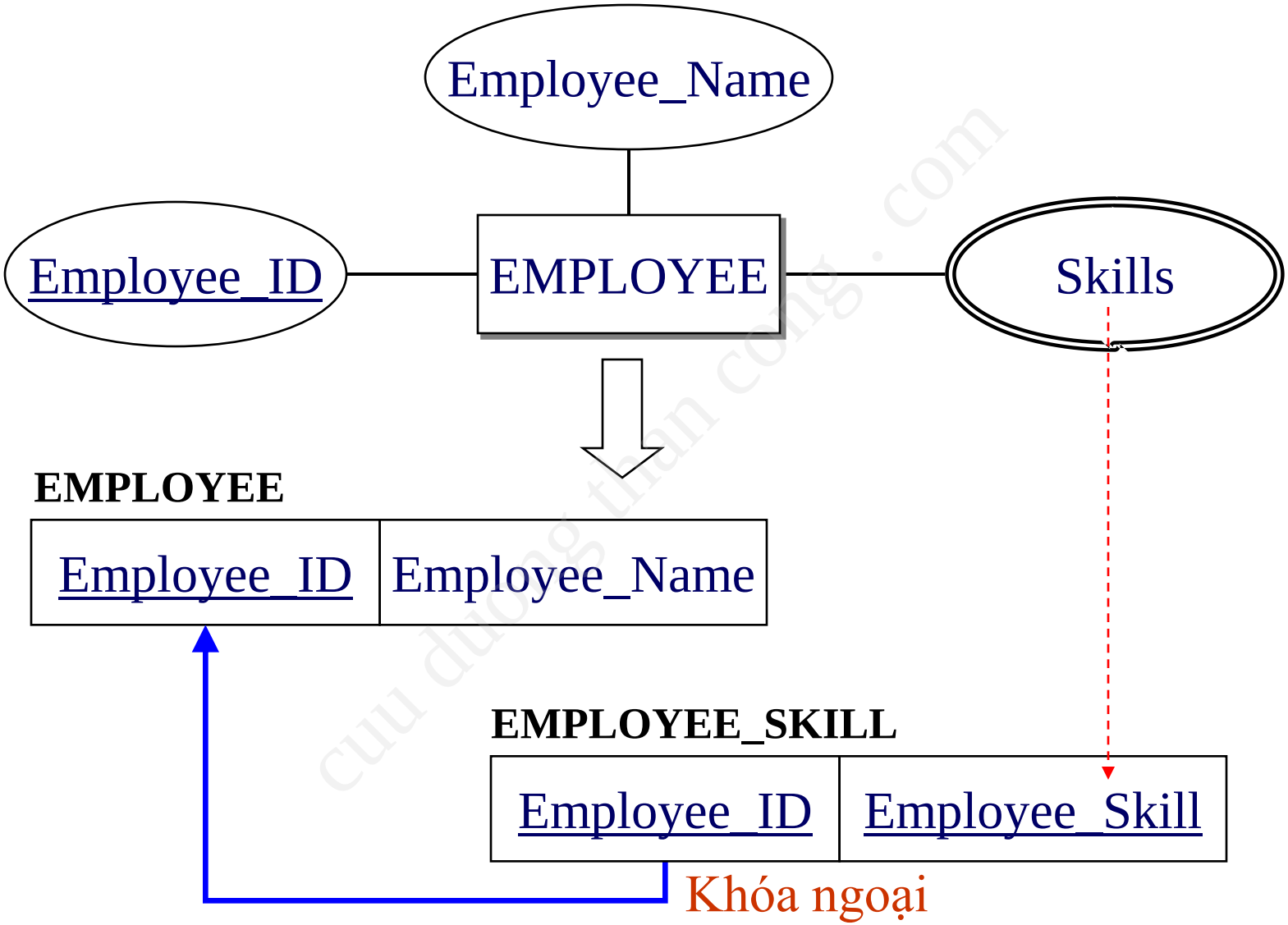
Thực thể sang bảng: 2.Composite attribute



CUSTOMER

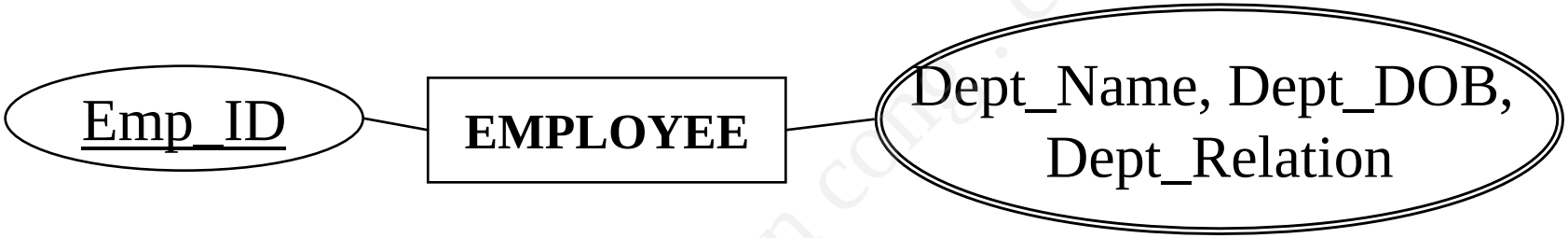
Customer_ID	Customer_Name	Street	City	State	Zip

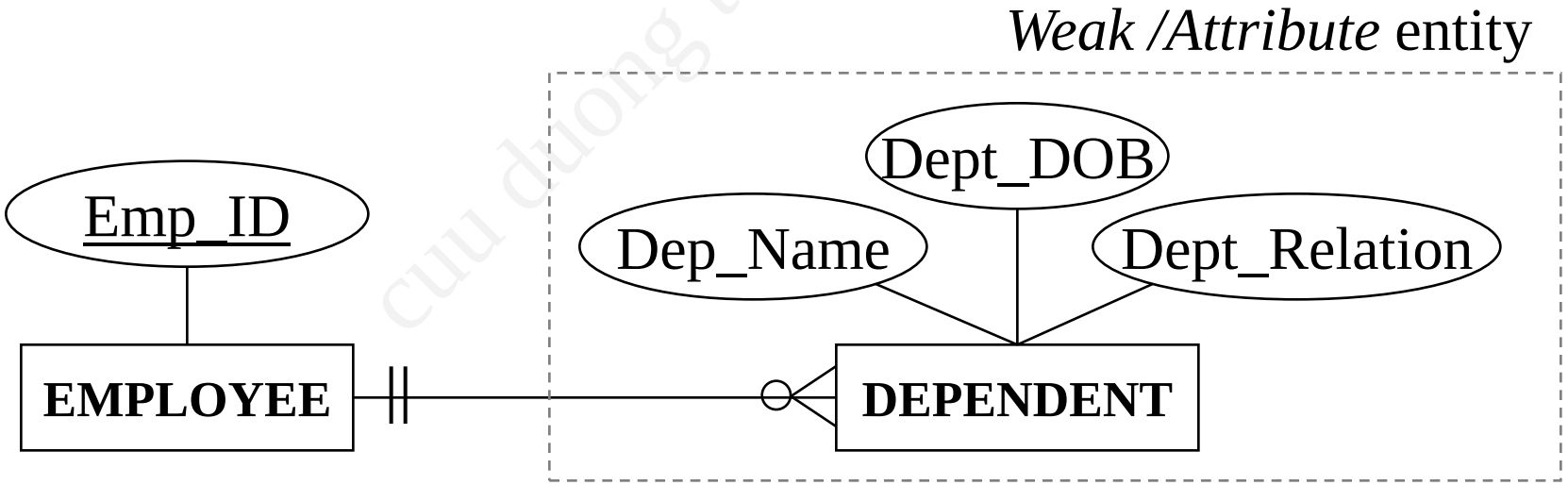
Thực thể sang bảng: 3.Multivalue attribute



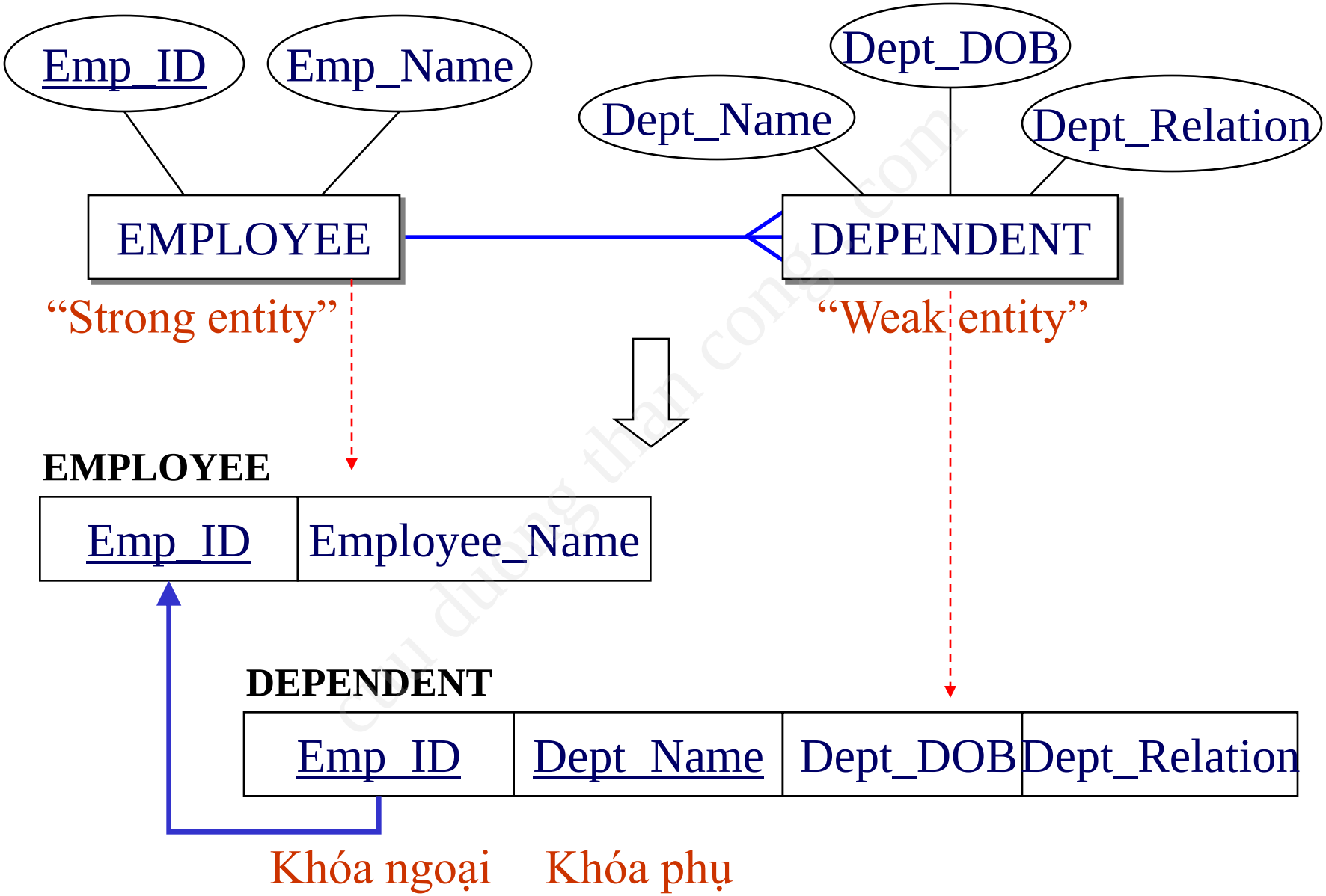
Multivalue Attribute

➲ *Repeating group*: là tập gồm nhiều thuộc tính kép có liên quan nhau như thông tin về người thân của nhân viên gồm Dep_Name, Dep_Age, Dep_Relation

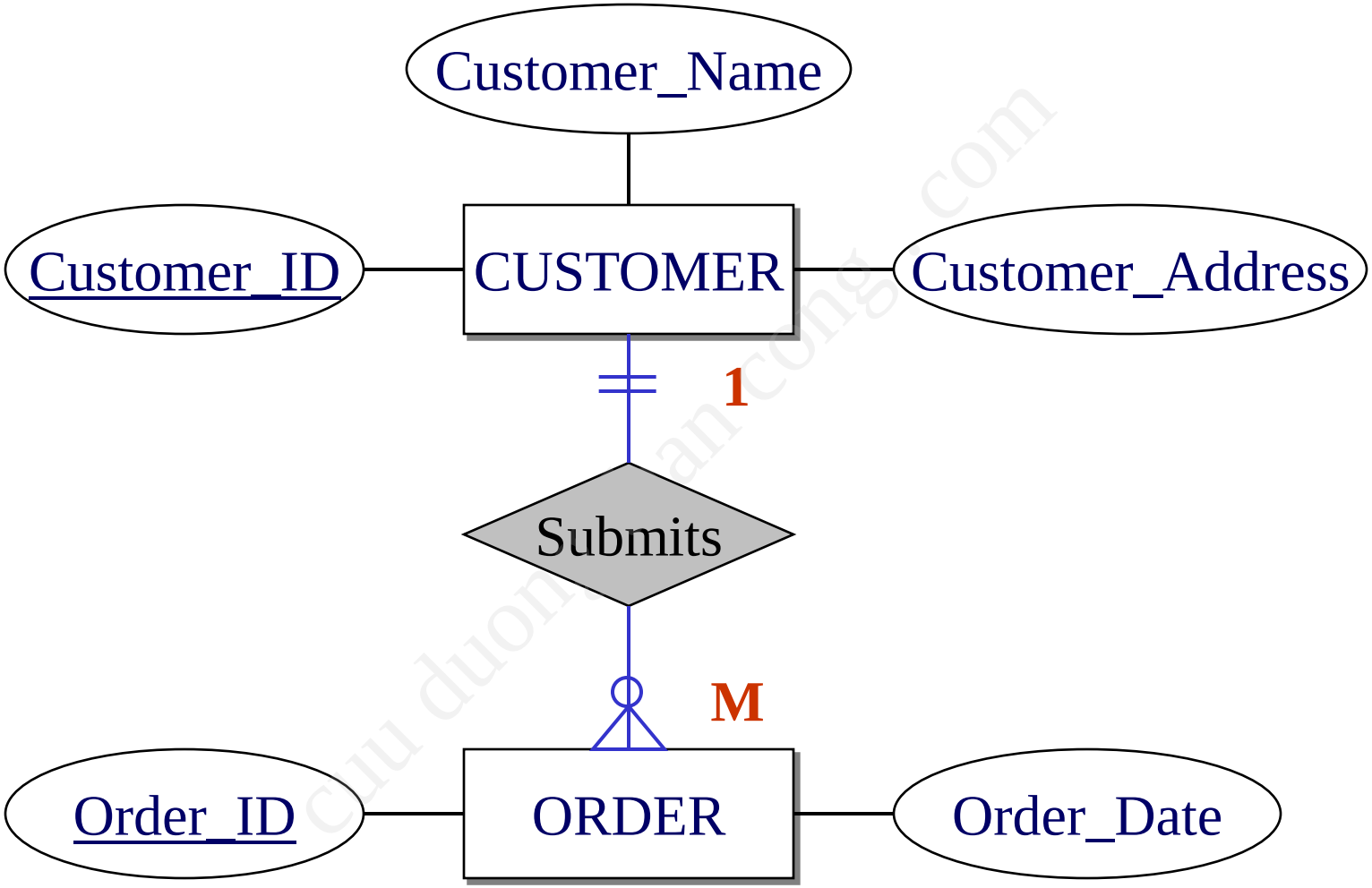




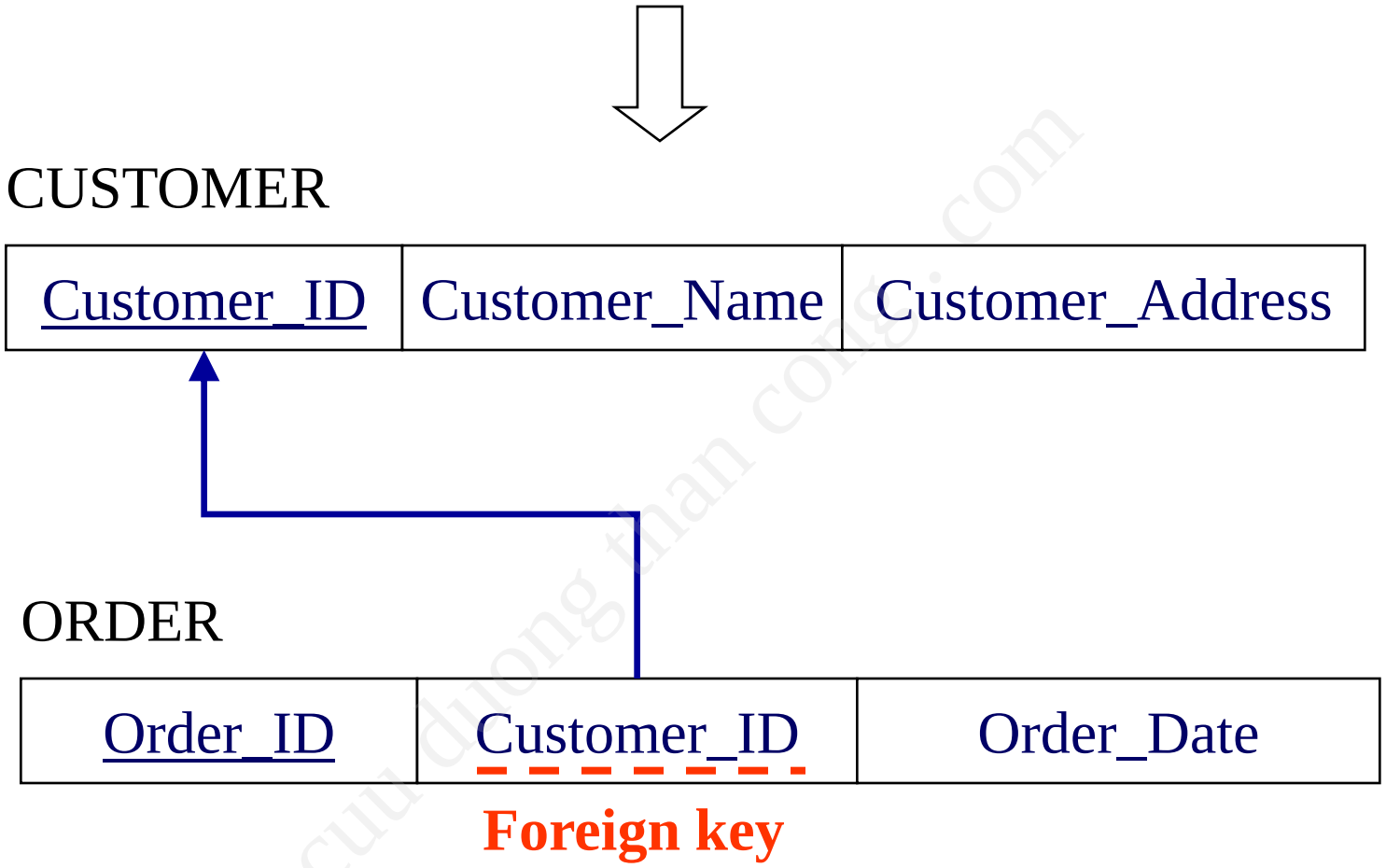
Thực thể sang bảng: 4.Weak-entity



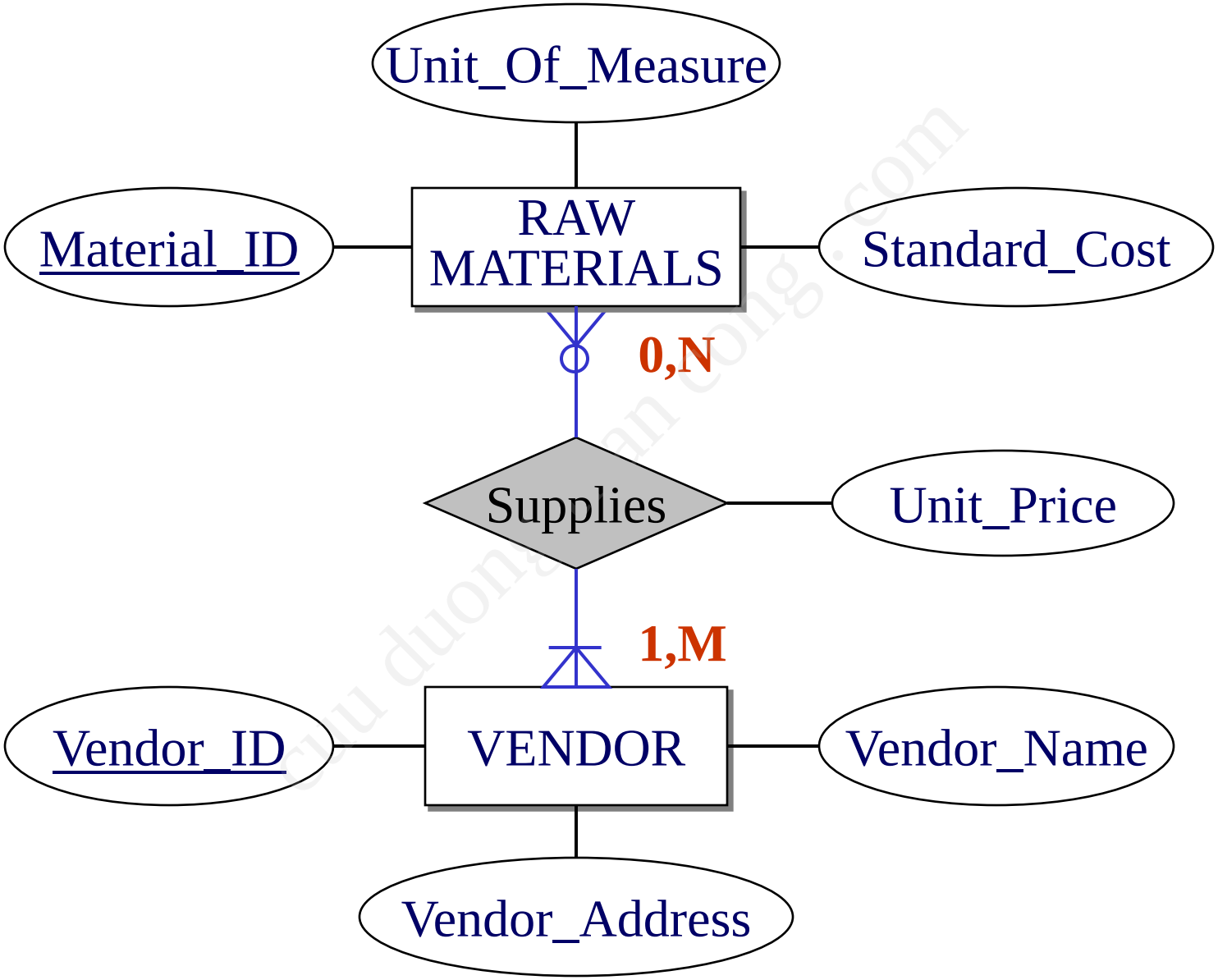
Chuyển quan hệ 2 ngôi 1:M



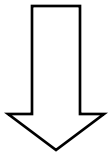
Chuyển quan hệ 2 ngôi 1:M



Chuyển quan hệ 2 ngôi M:N



Chuyển quan hệ 2 ngôi M:N



RAW MATERIALS

Material_ID	Standard_Cost	Unit_Of_Measure

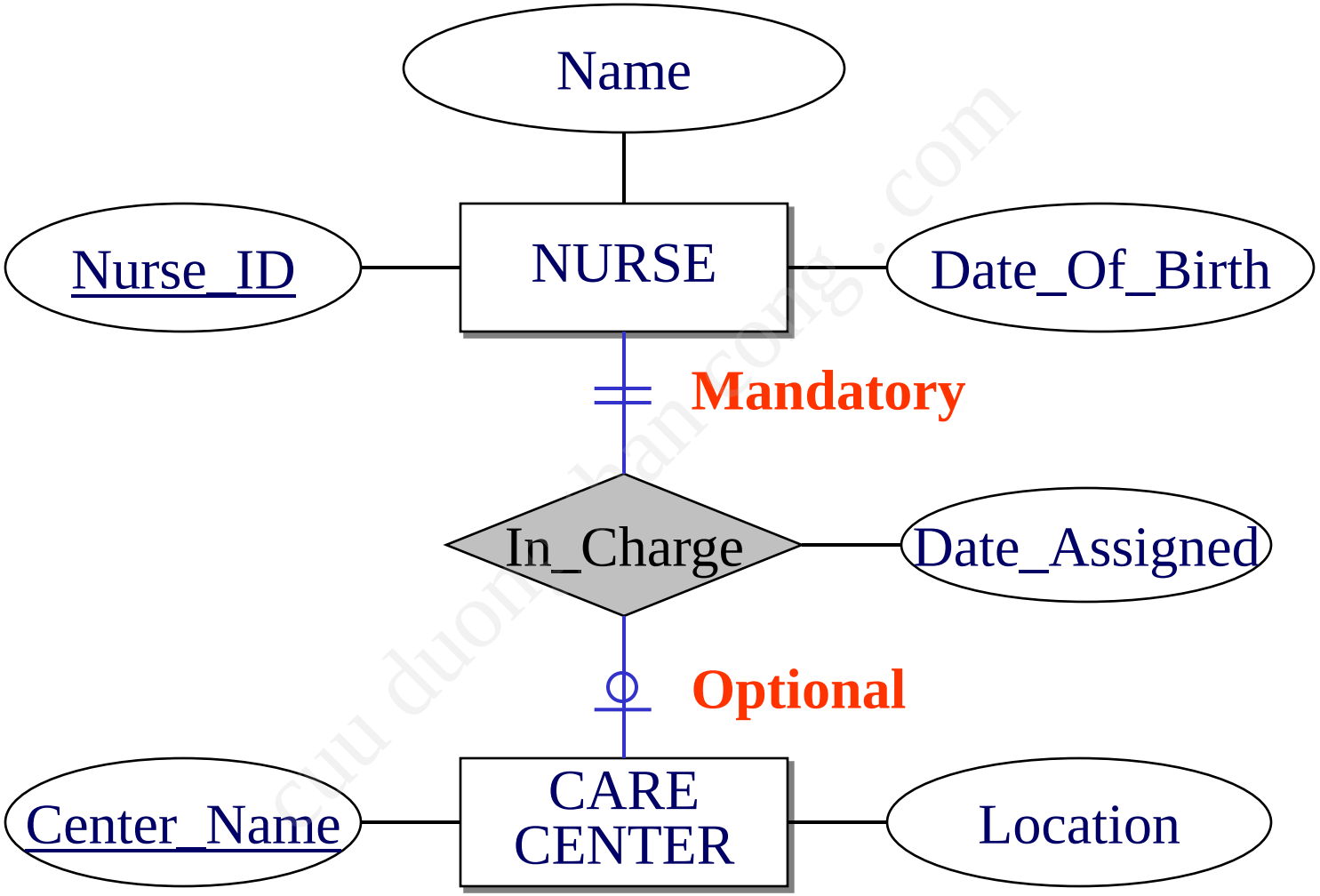
SUPPLY

Material_ID	Vendor_ID	Unit_Price

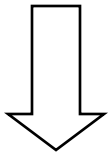
VENDOR

Vendor_ID	Vendor_Name	Vendor_Address

Chuyển quan hệ 2 ngôi 1:1



Chuyển quan hệ 2 ngôi 1:1



NURSE

Nurse_ID	Name	Date_Of_Birth

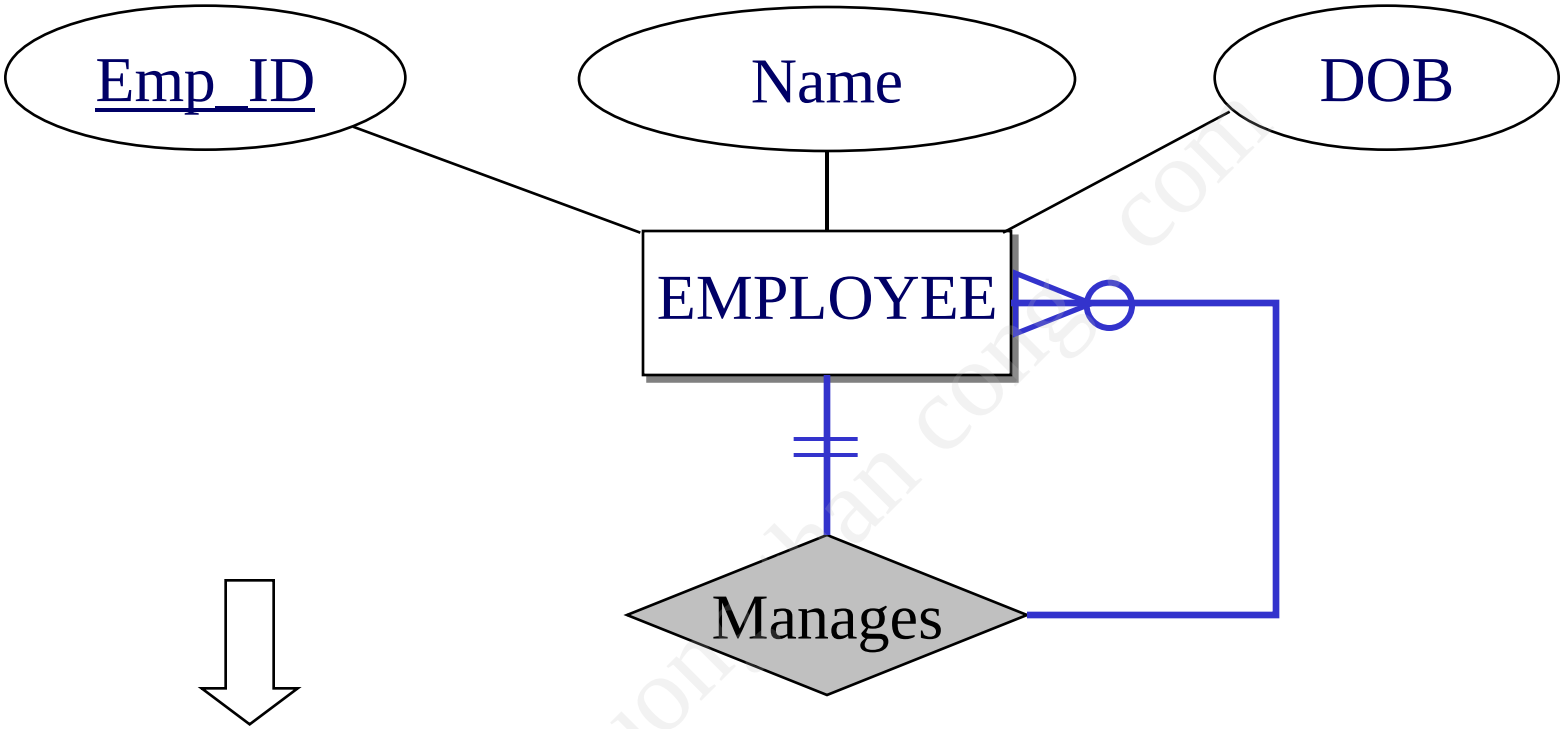
Mandatory 1

CARE CENTER

Center_Name	Location	Nurse_In_Charge	Date_Assigned

Foreign key

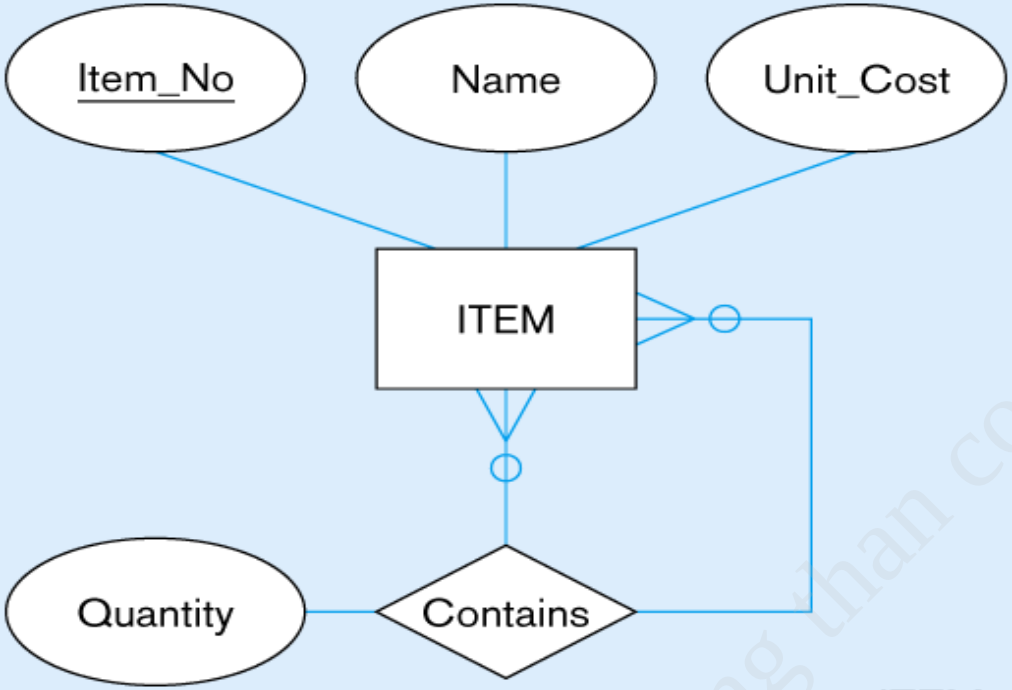
Unary Relationship 1:N



EMPLOYEE

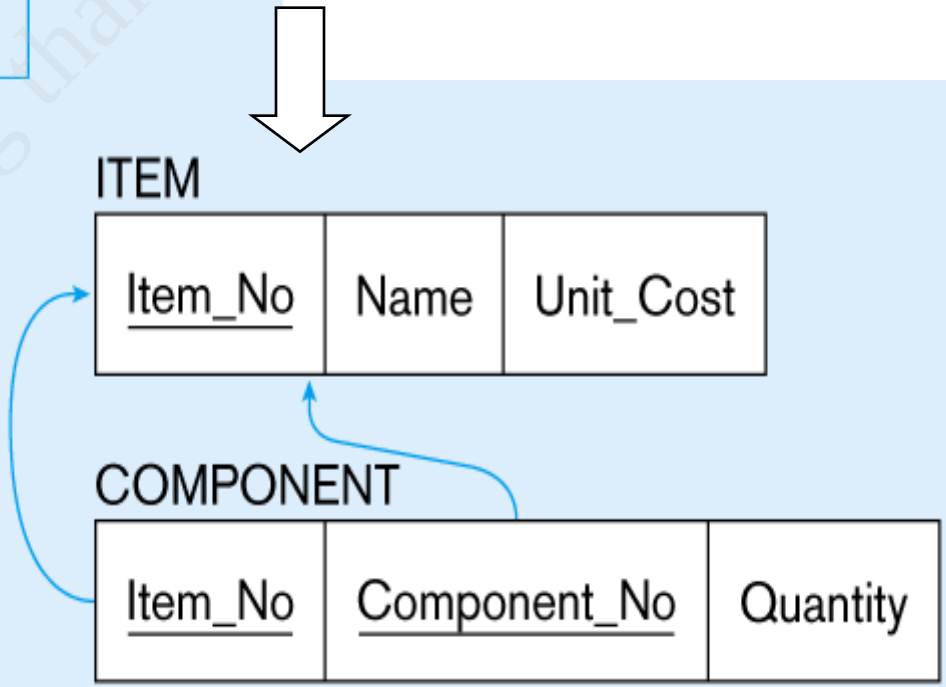
Emp_ID	Name	DOB	Manager_ID

Unary Relationship M:N



(a) Bill-of-materials relationships (M:N)

(b) ITEM and COMPONENT relations



Các ràng buộc toàn vẹn dữ liệu

1. *Entity integrity*. thực thể (bảng quan hệ) phải có một khóa không rỗng. Vd: Emp_ID.
2. *Referential integrity constraints*. Ràng buộc toàn vẹn các tham chiếu và cardinalities (bảo đãm cho các tham chiếu giữa các thực thể không bị phá vỡ bởi các thao tác thêm, xóa sửa dữ liệu).
3. *Domains*. Miền giá trị hợp lệ cho các thuộc tính.
   - Mỗi thuộc tính của thực thể được quy định một miền gía trị hợp lệ. Các giá trị nằm ngoài miền giá trị này là các giá trị vô nghĩa đối với thực thể, cần loại bỏ.
   - *Triggering operations*. Các toán tử bảo vệ tính hợp lý của các giá trị thuộc tính của thực thể.

Triggering Operations

- Thực thi theo quy tắc (cho phép hoặc cấm) tham chiếu hoặc cập nhật dữ liệu; nó được kích hoạt thực hiện theo điều kiện đã quy định trước.
- Triggering Op. có 5 thành phần cơ bản:
  - *Rule*: quy tắc quản lý tương ứng với Triggering Op.
  - *Event*: sự kiện kích hoạt Triggering Operation
  - *Entity name*: thực thể bị truy xuất hoặc chỉnh sửa
  - *Condition*: điều kiện cho phép thực hiện Triggering Op.
  - *Action*: nội dung thực hiện
- Giúp thiết lập thư viện hàm chuẩn trên DMBS.
  - Là các "Stored procedures" trong DBMS.

Data Normalization

- Là xử lý phân rã các bảng quan hệ để tạo ra các bảng quan hệ có cấu trúc tốt (well-structured relations): bảng quan hệ chỉ chứa tối thiểu dữ liệu dư thừa và cho phép users insert, delete, update các mẫu tin theo quy tắc, mục đích là để tránh làm hư dữ liệu và các liên kết.
  - **Insertion Anomaly** – thêm mẫu tin mới làm sai dữ liệu.
  - **Deletion Anomaly** – xóa mẫu tin củ gây mất dữ liệu cần thiết cho các mẫu tin sẽ dùng sau này
  - **Modification Anomaly** – thay đổi dữ liệu trong mẫu tin phải thay đổi đồng thời cùng với các mẫu tin khác.
- Việc chuẩn hóa bảng dựa trên khái niệm phụ thuộc hàm (*Functional Dependency*): giá trị của một thuộc tính được xác định (một cách logic) bởi giá trị của thuộc tính khác
  - Vd: Emp_ID → Emp_name, Emp_salary
  - Thuộc tính không khóa bị phụ thuộc hàm vào khóa.

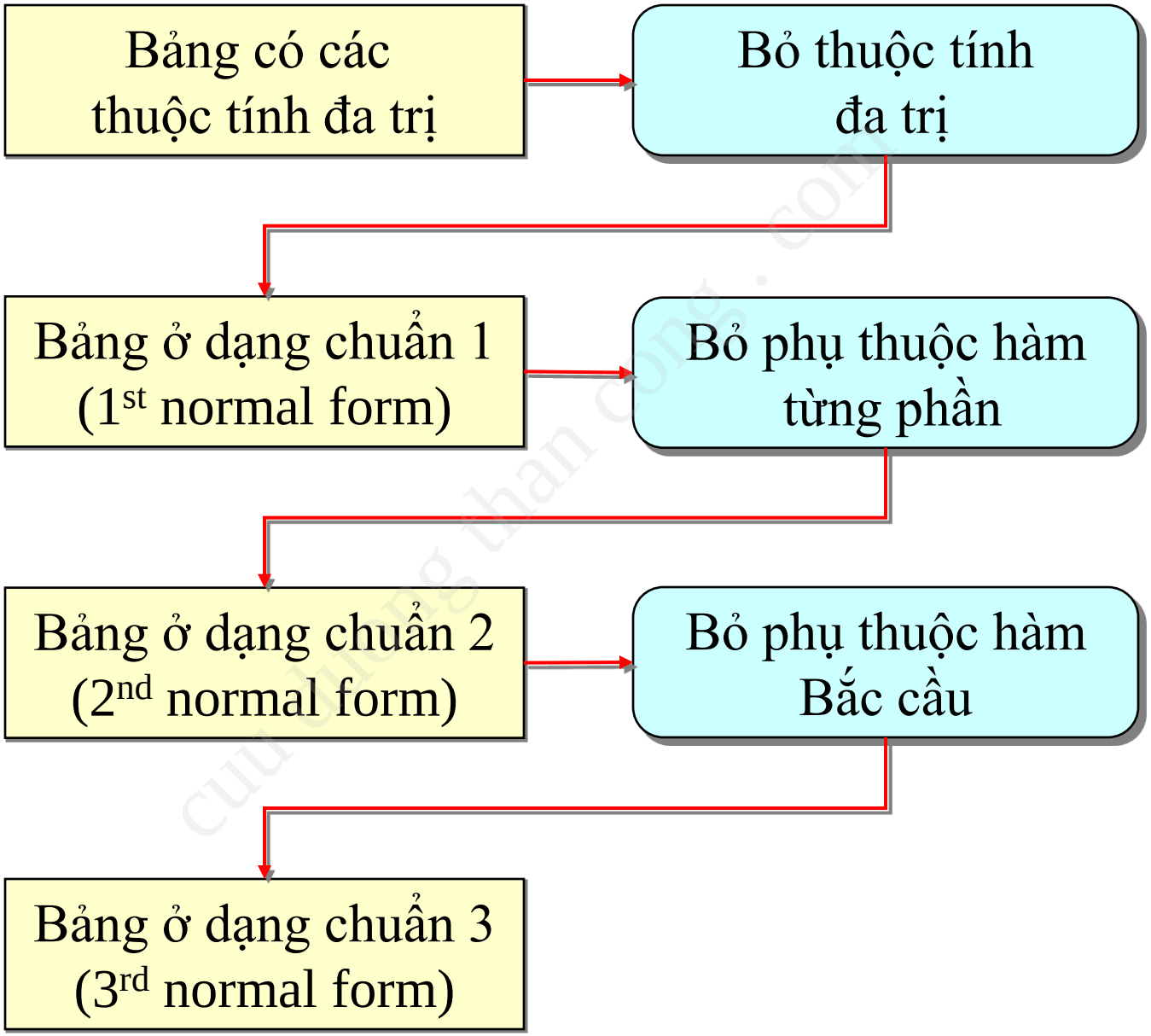
Ví dụ Anomaly

EMPLOYEE2

Emp_ID	Name	Dept_Name	Salary	Course_Title	Date_Completed
100	Margaret Simpson	Marketing	48,000	SPSS	6/19/200X
100	Margaret Simpson	Marketing	48,000	Surveys	10/7/200X
140	Alan Beeton	Accounting	52,000	Tax Acc	12/8/200X
110	Chris Lucero	Info Systems	43,000	SPSS	1/12/200X
110	Chris Lucero	Info Systems	43,000	C++	4/22/200X
190	Lorenzo Davis	Finance	55,000		
150	Susan Martin	Marketing	42,000	SPSS	6/19/200X
150	Susan Martin	Marketing	42,000	Java	8/12/200X

- **Thêm** – không thể thêm nhân viên mới nếu nhân viên đó không tham gia lớp học nào
- **Xóa** – Nếu ta bỏ nhân viên số 140, ta sẽ bị mất thông tin về sự tồn tại của khóa học "Tax Acc" (vì bảng chỉ có 1 dòng chứa thông tin này)
- **Sửa** – Nếu tăng lương cho nhân viên số 100, ta phải cập nhật lại nhiều mẫu tin.

Các bước chuẩn hóa bảng



Chuẩn 1 (First Normal Form)

Bảng quan hệ dạng chuẩn 1 là bảng quan hệ không chứa thuộc tính đa trị hoặc có các cột đồng nghĩa.

➲ Đặc điểm: có nhiều dữ liệu bị trùng lặp giữa các mẫu tin

1st-NF relation

EMPLOYEE2

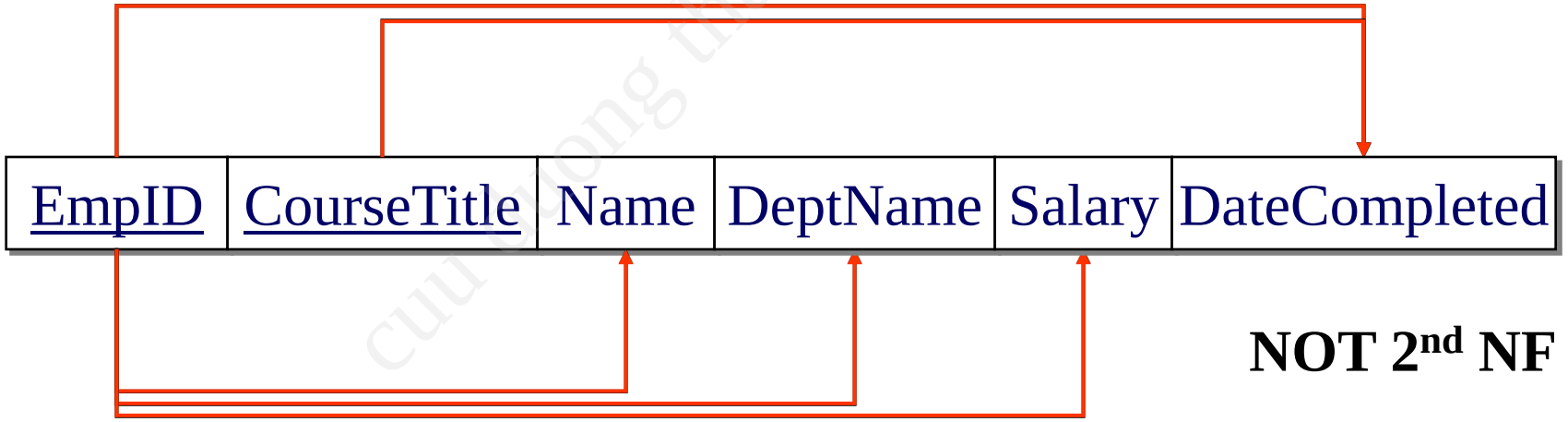
Emp_ID	Name	Dept_Name	Salary	Course_Title	Date_Completed
100	Margaret Simpson	Marketing	48,000	SPSS	6/19/200X
100	Margaret Simpson	Marketing	48,000	Surveys	10/7/200X
140	Alan Beeton	Accounting	52,000	Tax Acc	12/8/200X
110	Chris Lucero	Info Systems	43,000	SPSS	1/12/200X
110	Chris Lucero	Info Systems	43,000	C++	4/22/200X
190	Lorenzo Davis	Finance	55,000		
150	Susan Martin	Marketing	42,000	SPSS	6/19/200X
150	Susan Martin	Marketing	42,000	Java	8/12/200X

Chuẩn 2 (Second Normal Form)

Bảng dạng chuẩn 2 là bảng ở dạng chuẩn 1 **và** mọi thuộc tính không phải là khóa thì phụ thuộc hàm vào toàn bộ khóa chính

- Không có sự phụ thuộc hàm vào 1 phần của khóa chính

DateCompleted phụ thuộc hàm hoàn toàn vào (EmpID, CourseTitle)



NOT 2nd NF !!

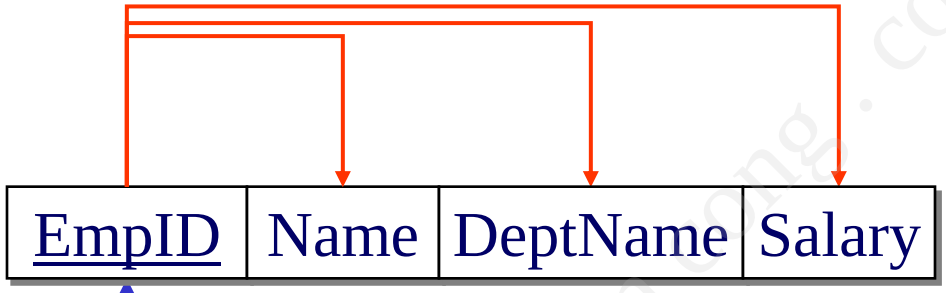
Name, DeptName, Salary chỉ phụ thuộc vào (Emp_ID)

➲ Các bảng quan hệ thỏa mãn các yêu cầu sau là bảng ở dạng chuẩn 2 (2NF)

1. Là bảng quan hệ không có thuộc tính không khóa
2. Là bảng quan hệ có khóa là thuộc tính nguyên tố (không thể có phụ thuộc hàm vào 1 phần khóa chính)
3. Là bảng quan hệ có tất cả các thuộc tính không khóa phụ thuộc hàm đầy đủ vào toàn bộ khóa → nếu bảng có tồn tại 1 thuộc tính không khóa không phụ thuộc đầy đủ vào toàn bộ khóa thì bảng không thuộc dạng chuẩn 2

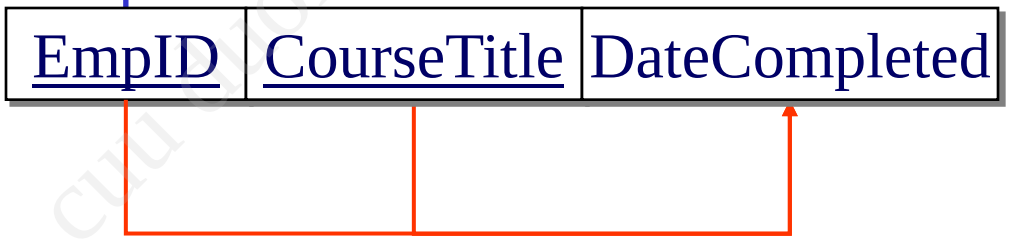
Biến đổi thành dạng chuẩn 2

1. Các thuộc tính chỉ phụ thuộc vào 1 phần khóa chính là EmpID kết hợp với EmpID để tạo thành 1 bảng chuẩn 2



(2nd NF)

3. Đặt quan hệ giữa 2 bảng mới



(2nd NF)

2. Các thuộc tính phụ thuộc hoàn toàn vào khóa chính kết hợp với khóa chính để tạo thành 1 bảng chuẩn 2

Chuẩn 3 (Third Normal Form)

Bảng dạng chuẩn 3 là bảng dạng chuẩn 2 **và** không chứa phụ thuộc hàm bắt cầu . Phụ thuộc hàm bắc cầu: một thuộc tính C phụ thuộc hàm vào thuộc tính B, thuộc tính B lại phụ thuộc hàm vào thuộc tính A, ký hiệu là A → B → C).

Cust_ID → Salesperson → Region : phụ thuộc bắc cầu

SALES

Cust_ID	Name	Salesperson	Region
8023	Anderson	Smith	South
9167	Bancroft	Hicks	West
7924	Hobbs	Smith	South
6837	Tucker	Hernandez	East
8596	Eckersley	Hicks	West
7018	Arnold	Faulb	North

Biến đổi thành dạng chuẩn 3

SALES1

Cust_ID	Name	Salesperson
8023	Anderson	Smith
9167	Bancroft	Hicks
7924	Hobbs	Smith
6837	Tucker	Hernandez
8596	Eckersley	Hicks
7018	Arnold	Faulb

SPERSON

Salesperson	Region
Smith	South
Hicks	West
Hernandez	East
Faulb	North

Salesperson → Region

SPERSON (Salesperson, Region)

SALE1 (Cust_ID, Name, Salesperson)

CustID → Name, Salesperson

Trộn các bảng quan hệ (Merge)

- Sau khi thực hiện normalization, một số bảng quan hệ có thể dư thừa vì cùng mô tả cùng một hoặc một số thực thể tương tự nhau.

  EMPLOYEE1(Emp_ID, Name, Address,Phone)

  EMPLOYEE2(Emp_ID, Name, Address, Jobcode, Number_of_years)

  Có thể trộn 2 bảng này thành 1:

  EMPLOYEE(Emp_ID, Name, Address, Phone, Jobcode, Number_of_years)

- Việc trộn các bảng phải bảo toàn ý nghĩa của dữ liệu

Trộn các bảng quan hệ: Synonyms

STUDENT1(Stu_ID, Name)

STUDENT2(Matriculation_number, Name,Address)

Nếu Stu_ID và Matriculation_number cùng được dùng để diễn tả cho một thuộc tính Social Security Number (số an sinh xã hội), hai bảng này có thể trộn lại và sử dụng khóa mới là SSN :

STUDENT (SSN, Name, Address)

Trộn các bảng quan hệ: Homonyms

STUDENT1(Stu_ID, Name, Address)

- Address: địa chỉ của SV trong khuôn viên trường

STUDENT2(Stu_ID, Name, PhoneNumber, Address)

- Address: địa chỉ nhà riêng của SV

Vì Address ở 2 bảng mang 2 ý nghĩa khác nhau, nên khi trộn 2 thuộc tính này cần đặt lại tên:

STUDENT (Stu_ID, Name, Campus_Address, Permanent_Address)

STUDENT1 (Stu_ID, Major) là bảng 3rd NF

STUDENT2 (Stu_ID, Advisor) là bảng 3rd NF

Nếu trộn 2 bảng ta có

STUDENT (Stu_ID, Advisor, Major)

Tuy nhiên, nếu Advisor → Major thì bảng trên không là 3rdNF, cần phải bỏ phụ thuộc hàm bắc cầu một lần nữa:

STUDENT (Stu_ID, Advisor)

ADVISOR-MAJOR (Advisor, Major)

Bài tập

1. Giả sử ta có bảng quan hệ **R**(A, B, C, D, E) với khóa chính A,B và phụ thuộc hàm A→C và D → E. Hãy chuyển quan hệ sang dạng chuẩn 3NF.

2. Quan hệ MEETING có các thuộc tính và khóa như sau:

   MEETING(Course#, CourseTitle, Day, Time, BuildingName, Room#, Address). Giả sử có 2 phụ thuộc hàm trên quan hệ này:

   - *BuildingName* phụ thuộc hàm vào *Address*
   - *CourseTitle* chỉ phụ thuộc hàm vào *Course#*

   Hãy chuyển quan hệ sang dạng 3NF.

3. Quan hệ BUILDING có các thuộc tính và khóa như sau:

   BUILDING(Street#, City, BuildingName, OwnerAddress, Tax). Giả sử có 3 phụ thuộc hàm trên quan hệ này:

   1. *BuildingName* phụ thuộc hàm vào *street#, city*
   2. *Tax* phụ thuộc hàm vào *City*
   3. *OwnerAddress* phụ thuộc hàm vào *BuildingName*

   Hãy chuyển quan hệ sang dạng 3NF.

Bài tập

4. Một hãng xe hơi có CSDL lưu các thông tin sau:

- Thông tin vê nhà cung cấp (Suppliers)

  Mỗi nhà cung cấp được hãng xe hơi gán một số định danh Supplier# để phân biệt nhau

  Mỗi nhà cung cấp có tên (SupplierName), và thành phố (SupplierCity)

  Mỗi nhà cung cấp cung cấp 1 hoặc nhiều phụ tùng cho xe

- Thông tin vê phụ tùng xe (Parts)

  Mỗi phụ tùng xe có một tên (PartName), số định danh (Part#) và giá (PartPrice)

  Mỗi phụ tùng được cung cấp bởi 1 hay nhiều nhà cung cấp

  Part# là số dùng để phân biệt các phụ tùng xe với nhau

- Thông tin vê đợt cung cấp (Supply)

  Mỗi đợt cung cấp có nhà cung cấp cung cấp một số phụ tùng

  Mỗi đợt cung cấp có số lượng(quantity), ngày (date), tên bộ phận (partName), thành phố của nhà cung cấp (SupplierCity), và mã vùng của nhà cung cấp (Supplier_Postal)

Bài tập

Giả sử hãng đã biết được các quan hệ phụ thuộc như sau:

1. Số lượng trong mỗi đợt cung cấp hàng phụ thuộc hoàn toàn vào supplier#,part#,date
2. PartName được xác định bởi Part#
3. SupplierCity,SupplierPostal phụ thuộc vào Supplier#
4. Nếu biết SupplierPostal, ta sẽ biết được SupplierCity

a) Hãy vẽ lược đồ ERD cho CSDL.

b) Chuyển lược đồ ERD sang bảng quan hệ và chuẩn hóa thành dạng 3NF.

Nén dữ liệu bằng Lookup Table

BILLING_ADDRESS

Cust_No	Block_no	Street	City_State
1273	2226	Plainfield Ave	NJ
6390	2110	Plainfield Ave	NJ

40 bytes

BILLING_ADDRESS

Cust_No	Block_no	Ref	City_State
1273	2226	1001	NJ
6390	2110	1001	NJ

4 bytes

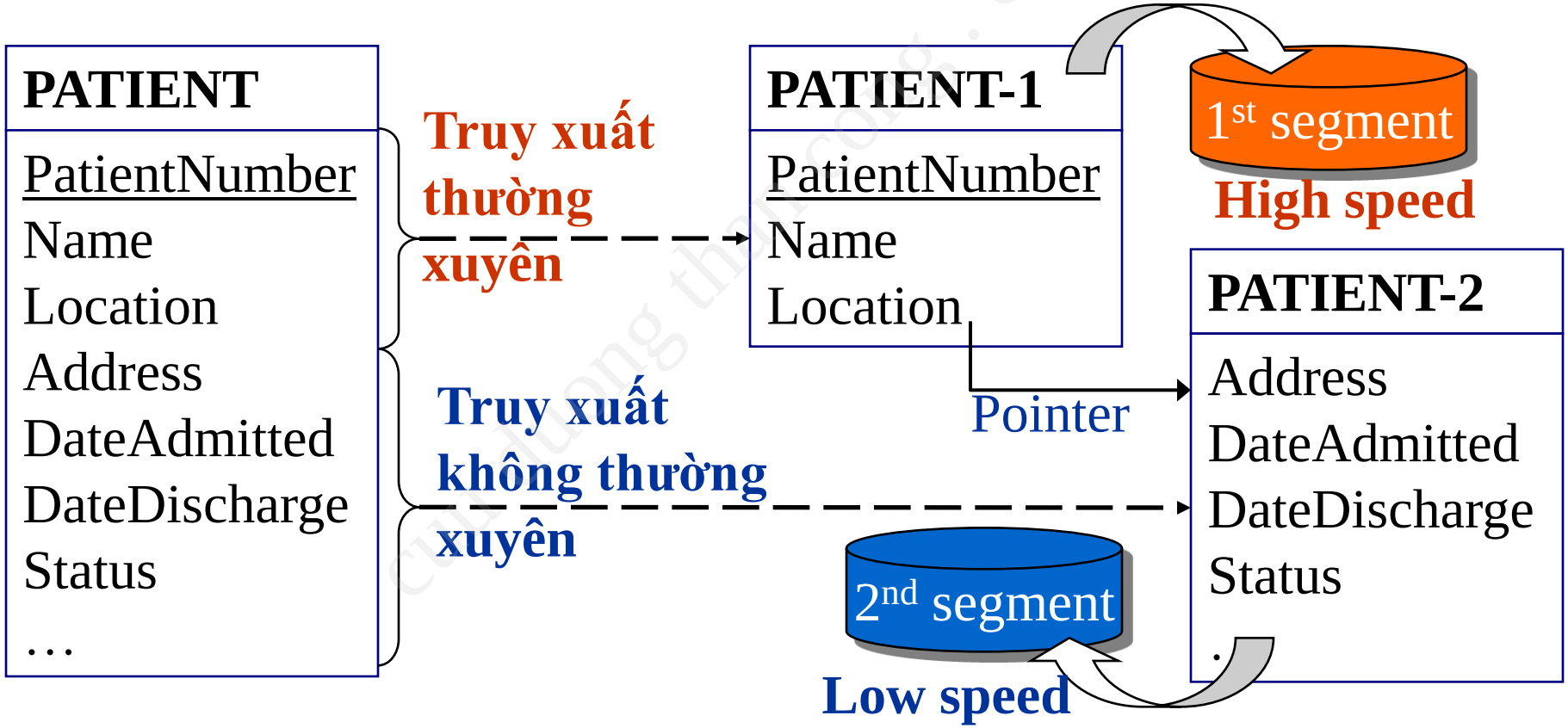
LOOKUP_STREET

Ref	Street
1001	Plainfield Ave
7024	Oak

Range: 0 .. 9999

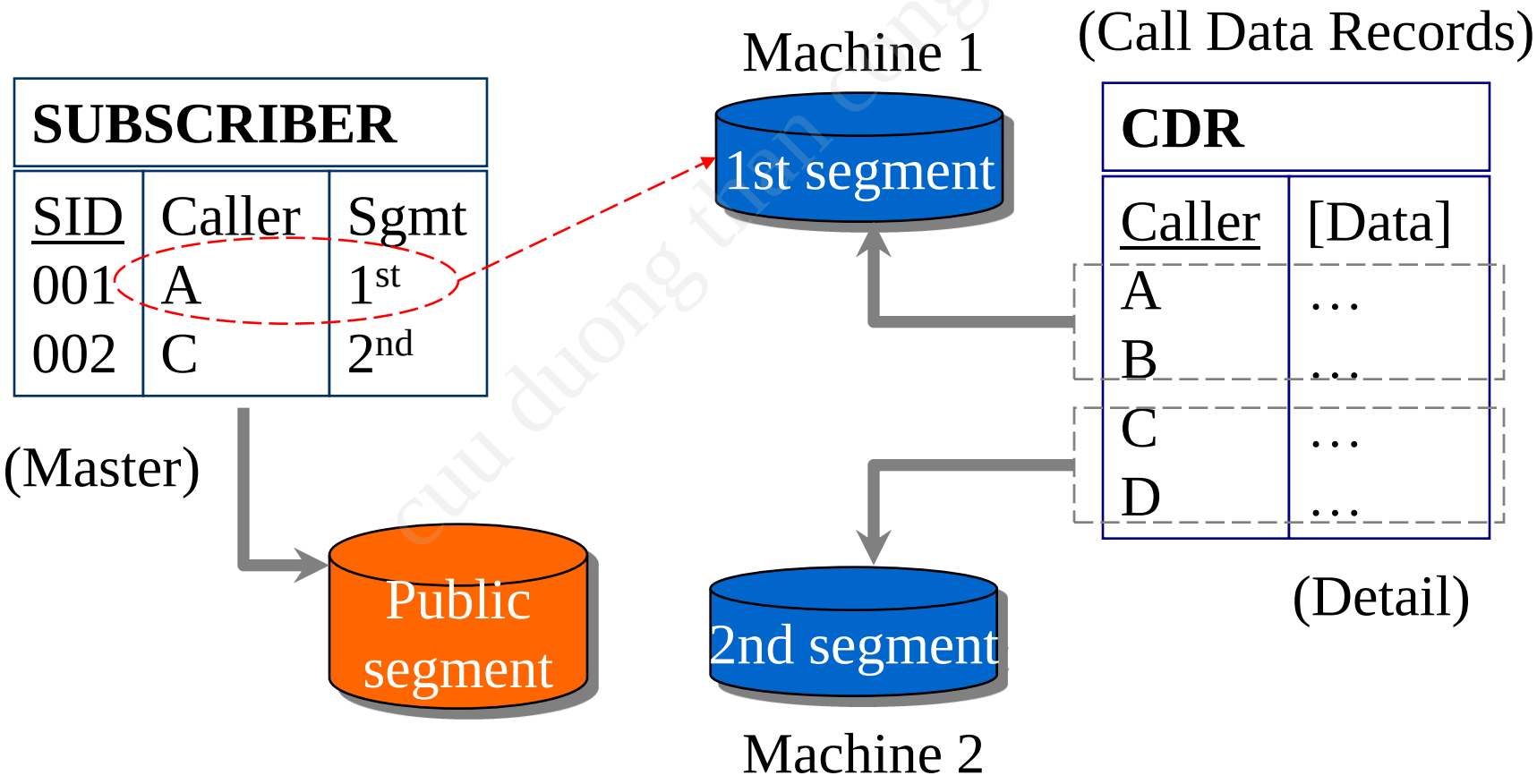
Phân khúc các fields

➲ Các fields có tần suất truy xuất (chủ yếu là đọc) không đều nhau, do đó cần phân lập các fields có mức độ truy xuất cao lưu trữ trong ổ dĩa có tốc độ cao để cải tiến tốc độ.

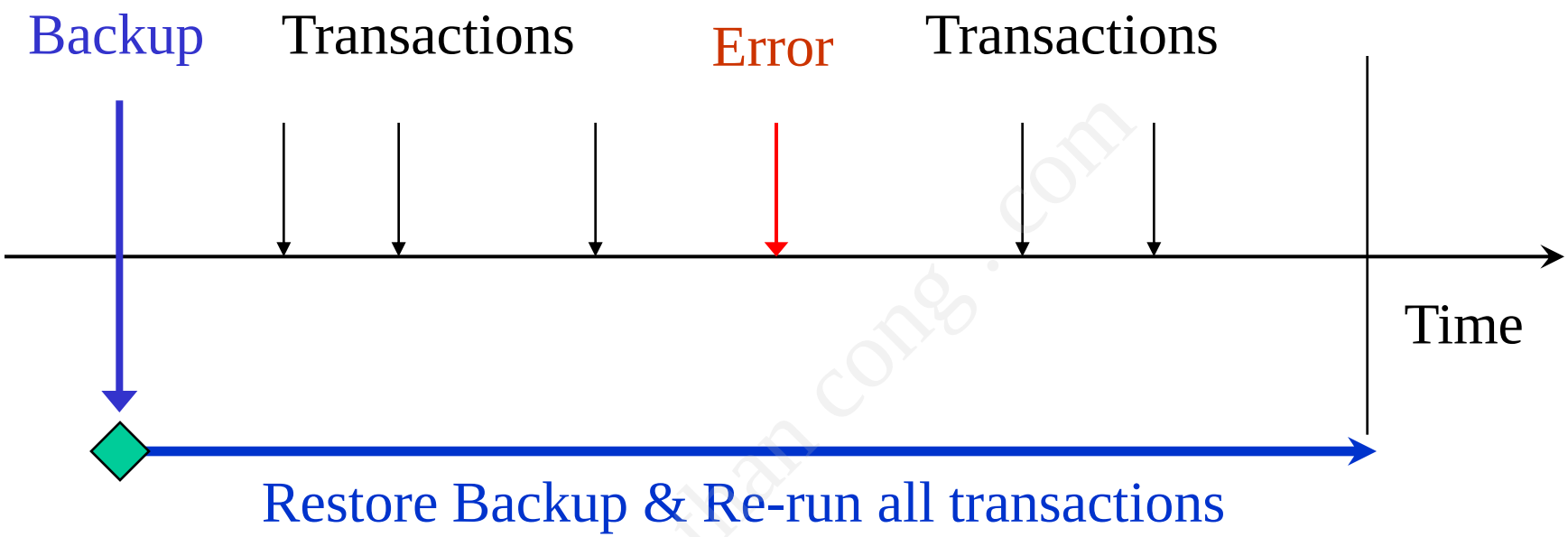


Phân khúc các records

➲ Khi số lượng records trong 1 bảng (trên 1 máy) quá nhiều thì các câu lệnh truy xuất dữ liệu sẽ thực thi rất lâu, cần chia tập các mẫu tin ra thành nhiều phân đoạn để lưu trữ và xử lý trên nhiều máy.

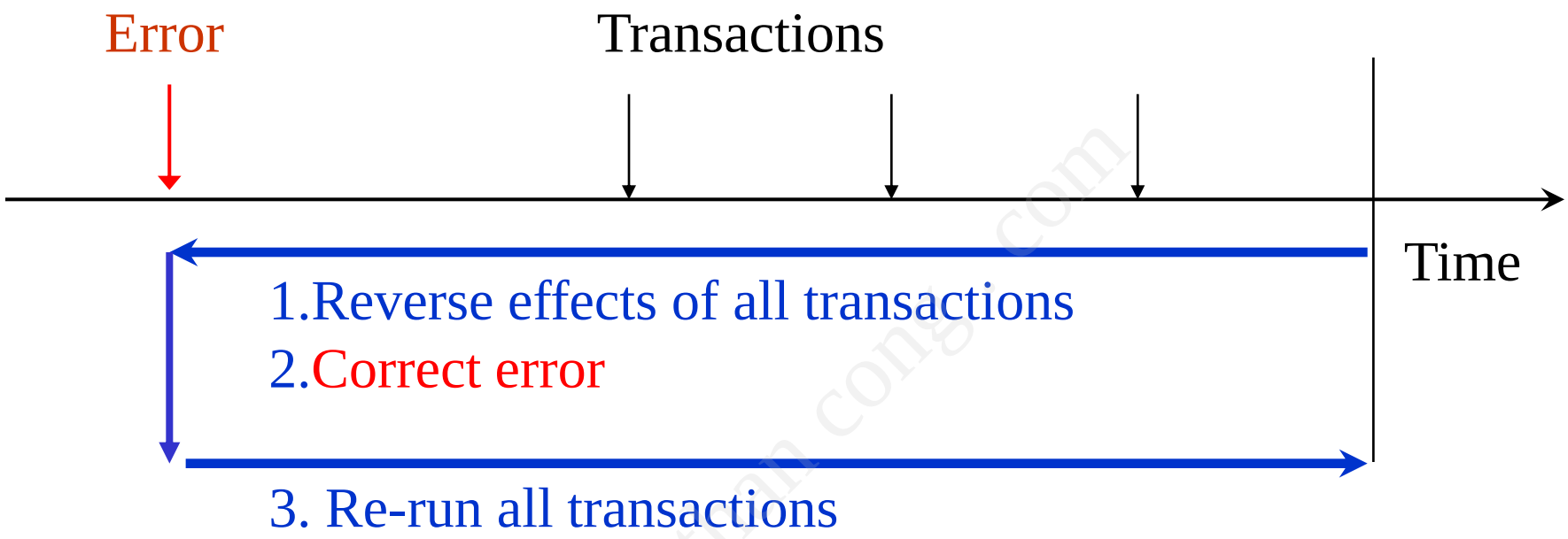


Forward recovery (rollforward)



- Là phương pháp Sử dụng bản backup mới nhất để restore, và thực hiện lại tất cả các lệnh cập nhật cần thiết sau khi đã khắc phục được lỗi.
- Nếu chu kỳ backup càng dài thì thời gian cần thiết để khôi phục dữ liệu càng lâu. Tuy nhiên, backup thường xuyên sẽ làm nặng tải cho DBMS.

Backward recovery (rollback)



- Là phương pháp sử dụng bản dữ liệu hiện tại, thực hiện 'undo' tất cả các lệnh đã cập nhật dữ liệu, khắc phục lỗi và thực hiện lại các lệnh này.
- Phương pháp này không cần bản backup, nhưng cần bản dữ liệu hiện tại (kể cả DBMS) không bị hư hỏng.