

Chương 2

CÁC CÔNG NGHỆ

VÀ KỸ THUẬT TÍCH HỢP CSDL

1. Mô hình dữ liệu mở rộng XML.
2. Chuyển đổi lược đồ dữ liệu giữa các mô hình.
3. Tích hợp các lược đồ dữ liệu.
4. Chuyển đổi và tích hợp dữ liệu.



Ngôn ngữ XML

- XML(**Extensible Markup Language**): ngôn ngữ định dạng mở rộng.
- XML là ngôn ngữ được định nghĩa bởi tổ chức mạng toàn cầu (**World Wide Web Consortium**) thường được viết tắt **W3C**.
- XML là một ngôn ngữ tổng quát dùng để định nghĩa dữ liệu thông qua các thẻ.
- XML là một chuẩn không phụ thuộc vào bất kì một hệ điều hành nào.

Ngôn ngữ XML

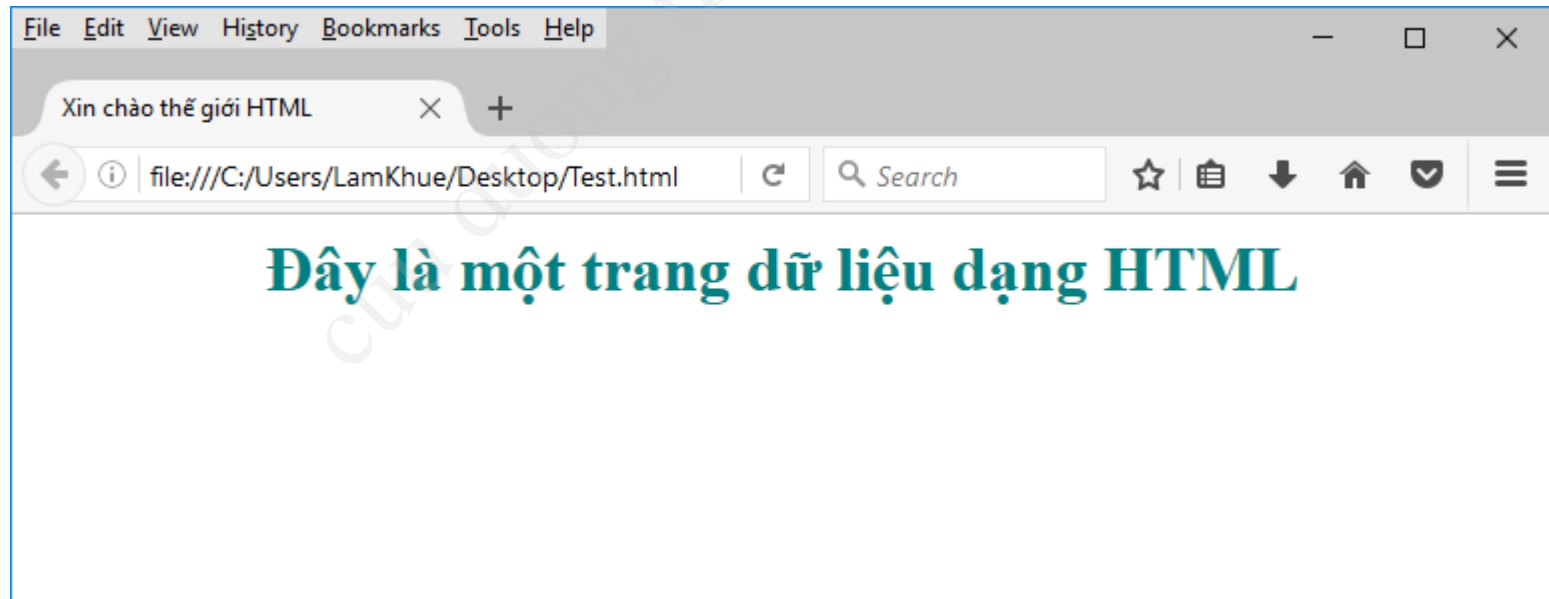
- Ngôn ngữ định dạng là tất cả những gì dùng để mô tả nội dung một tài liệu.

- Ví dụ:

```
<html>
  <head>
    <title>Xin chào thế giới HTML</title>
  </head>
  <body>
    <h1 align="center"><font
      color="#008080">Đây là một trang dữ liệu
      dạng HTML</font></h1>
  </body>
</html>
```

Ngôn ngữ XML

- Ngôn ngữ định dạng là tất cả những gì dùng để mô tả nội dung một tài liệu.
- Ví dụ:



❖ HTML và XML

- Cả hai đều là ngôn ngữ định dạng (định dạng theo nghĩa cách quy định để xử lý và chứa nội dung tài liệu).
- HTML sử dụng các thẻ được định nghĩa và quy định sẵn.
- XML đưa ra một số quy tắc cho phép người dùng tự định nghĩa các thẻ.

Ngôn ngữ XML

➤ Ví dụ:

```
<?xml version="1.0"?>
```

```
<DOCUMENT>
```

```
<GREETING>
```

Hello world

```
</GREETING>
```

```
<MESSAGE>
```

XML xin chào thế giới

```
</MESSAGE>
```

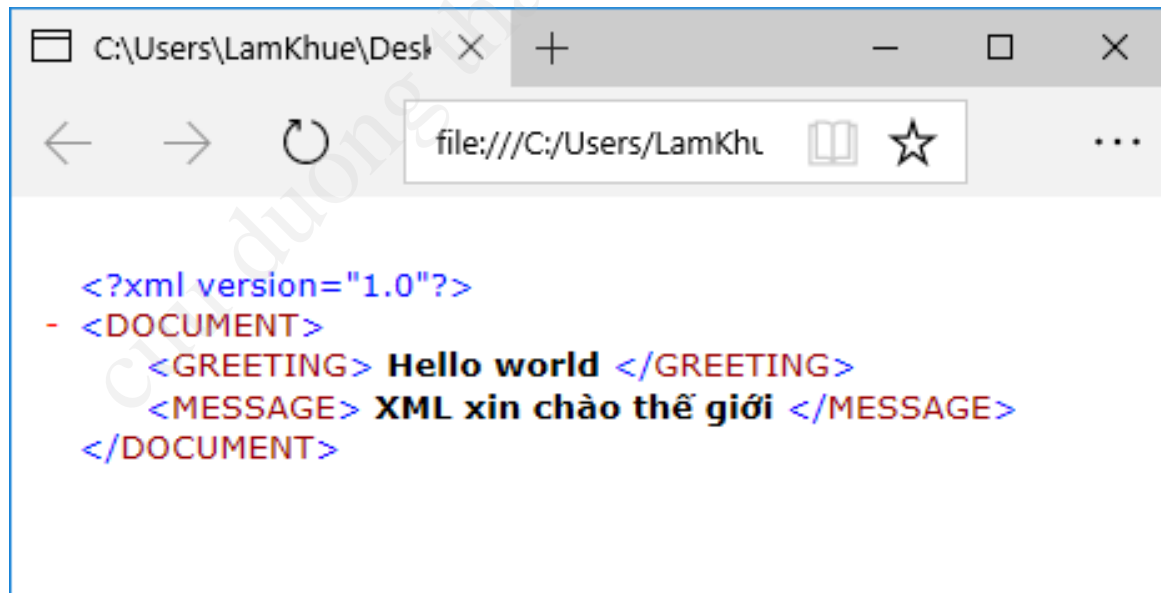
```
</DOCUMENT>
```

Ngôn ngữ XML

- Tất cả các chỉ thị của XML đều được bắt đầu bằng `<?` và kết thúc bằng `?>`.
- Các thẻ do người dùng tự định nghĩa, ví dụ như `<DOCUMENT>`, `<GREETING>`, `<MESSAGE>`.
- Thẻ luôn bắt đầu bằng `<` và kết thúc bằng `>`.
- Phải có thẻ mở và đóng duy nhất cho toàn bộ tài liệu (`root`).

Ngôn ngữ XML

- Trình duyệt chỉ có thể hiển thị file XML bằng cách đưa toàn bộ nội dung file XML lên màn hình.





Ngôn ngữ XML

- Định dạng file XML bằng CSS (Stylesheet).
- Định dạng file XML bằng XSLT.
- CSS và XSLT dùng để định kiểu và biến đổi XML để hiển thị dữ liệu phía người dùng không khác gì HTML.
- Dùng DOM, SAX để rút trích dữ liệu từ file XML kết hợp với các thẻ định dạng của HTML để hiển thị phía người dùng.

Ngôn ngữ XML

➤ Kết hợp với XSLT để định kiểu cho XML như sau:

```
<?xml version="1.0"?>
```

```
<xsl:stylesheet version="1.0"  
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"  
  xmlns="http://www.w3.org/TR/REC-html40">
```

```
<xsl:template match="/">
```

```
<html>
```

```
<head>
```

```
<title> <xsl:value-of select="DOCUMENT/GREETING"/> </title></head>
```

```
<body>
```

```
<h1 align="center"><font color="#FF0000"> <xsl:value-of  
  select="DOCUMENT/MESSAGE" /> </font></h1></body>
```

```
</html>
```

```
</xsl:template>
```

```
</xsl:stylesheet>
```

Ngôn ngữ XML

- Hiển thị tài liệu XML có kết hợp XSL.



❑ Trình duyệt XML:

➤ Internet Explorer (IE)

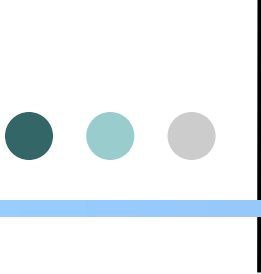
- ✓ Là trình duyệt XML mạnh nhất hiện nay.
- ✓ Hỗ trợ dùng Javascript để lập trình và truy xuất dữ liệu XML.
- ✓ Ngoài ra còn hỗ trợ cả Jscript, Vbscript.
- ✓ Hỗ trợ CSS.

➤ Netscape Navigation 6

- ✓ Phiên bản 6.0 hỗ trợ tốt XML.
- ✓ Hỗ trợ CSS
- ✓ Hỗ trợ các ngôn ngữ XML mở rộng.

❑ Ứng dụng XML:

- XML có thể tạo ra tập các ngôn ngữ con khác.
- Ứng dụng XML mang ý nghĩa cho biết một tập các thẻ hay tập con XML hoạt động riêng trong một lĩnh vực nào đó.
- MathML: định dạng các biểu thức, kí hiệu toán học.
- CML: Ngôn ngữ định dạng hóa học.



Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

- ❑ Định nghĩa kiểu tài liệu (DTD - Document Type Declaration):
 - Là một cách để miêu tả ngôn ngữ XML.
 - DTD kiểm tra từ vựng và tính hợp lệ của cấu trúc tài liệu XML theo các quy tắc ngữ pháp của ngôn ngữ XML thích hợp.
 - Một DTD trong XML có thể hoặc được xác định bên trong tài liệu, hoặc có thể được giữ trong một tài liệu riêng biệt.

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❑ Internal DTD trong XML:

- Khai báo làm việc độc lập với nguồn ngoại vi
- Các phần tử được khai báo bên trong XML file
- Thuộc tính *standalone*=“yes”
- Cú pháp:

<!DOCTYPE root-element [element-declarations]>

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❑ Cú pháp của internal DTD trong XML:

<!DOCTYPE element DTD identifier

[

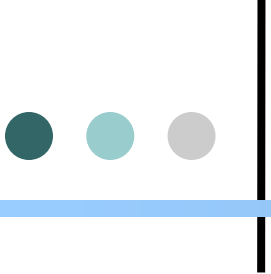
Declaration1

Declaration2

...

➤ DTD bắt đầu với delimiter là <!DOCTYPE

➤ Một **element** (phần tử) nói cho Parser để phân tích cú pháp tài liệu từ phần tử gốc đã cho



Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

- **DTD identifier** là một định danh cho Document Type Definition.
- **DTD identifier** có thể là path tới một file trên hệ thống hoặc URL tới một file trên mạng (khi này DTD được gọi là **External Subset**).
- Dấu ngoặc móc vuông **[]** bao quanh một danh sách tùy ý các khai báo thực thể, được gọi là **Internal Subset**.

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

➤ Ví dụ:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
```

```
<!DOCTYPE student [
```

```
  <!ELEMENT student (name, address, phone)>
```

```
  <!ELEMENT name (#PCDATA)>
```

```
  <!ELEMENT address (#PCDATA)>
```

```
  <!ELEMENT phone (#PCDATA)>]>
```

```
< student >
```

```
  <name>Trương Hồ</name>
```

```
  <address>123 Nguyễn Huệ</address>
```

```
  <phone>(08) 123-4567</phone>
```

```
</ student >
```

❑ **#PCDATA:** parse-able text data

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❖ Khai báo: `<!ELEMENT>` để định nghĩa kiểu tài liệu DTD cho một phần tử

❖ Cú pháp:

`<!ELEMENT NAME CONTENT_MODEL>`

➤ **NAME**: tên phần tử muốn định nghĩa.

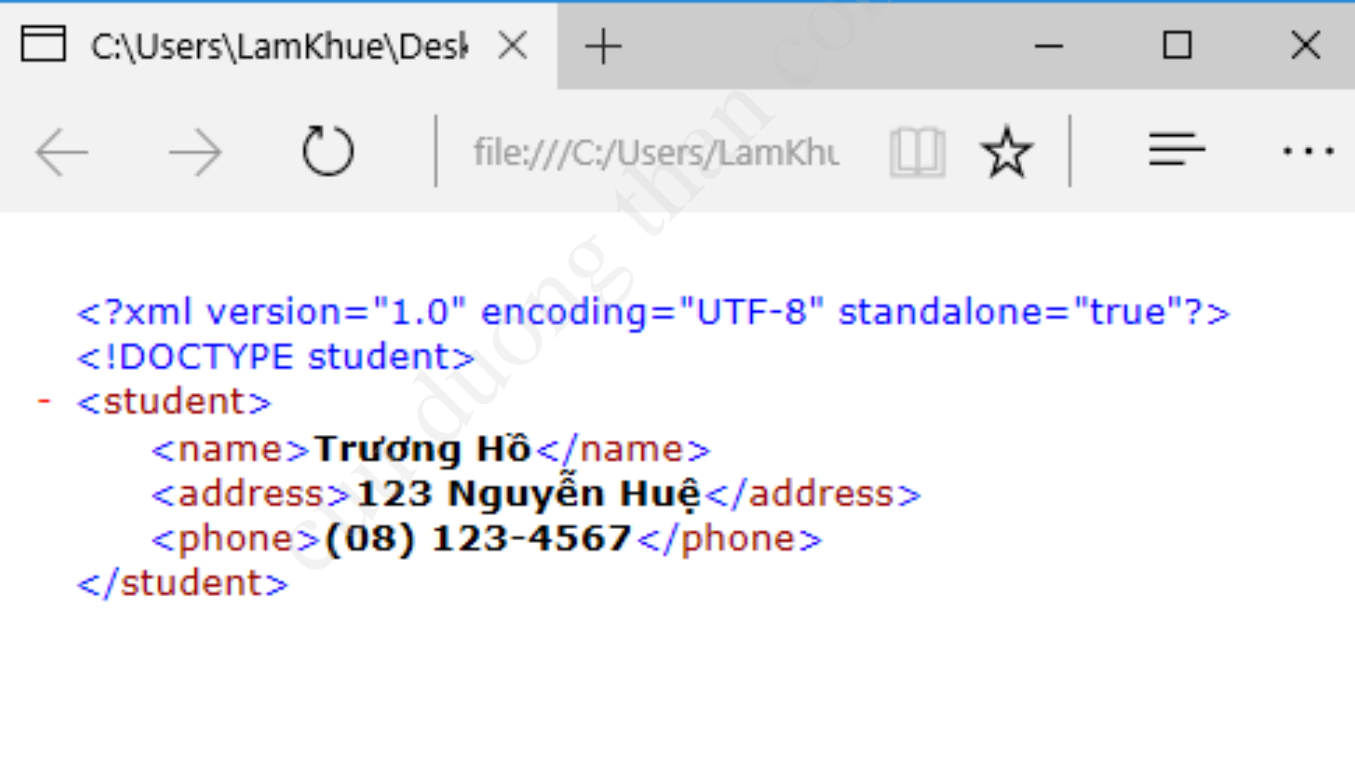
➤ **CONTENT_MODEL** có thể là:

✓ **ANY**: Dữ liệu có kiểu bất kỳ

✓ Các phần tử con khác hoặc dữ liệu

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

➤ Ví dụ: Kết quả hiển thị

A screenshot of a web browser window. The address bar shows the file path: file:///C:/Users/LamKhu... The main content area displays XML code with syntax highlighting. The code defines an XML document with a root element 'student' containing fields for name, address, and phone.

```
<?xml version="1.0" encoding="UTF-8" standalone="true"?>
<!DOCTYPE student>
- <student>
    <name>Trương Hồ</name>
    <address>123 Nguyễn Huệ</address>
    <phone>(08) 123-4567</phone>
</student>
```

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❖ Định nghĩa kiểu dữ liệu

- Danh sách các phần tử con: một phần tử DTD, nội dung có thể chứa những phần tử con khác.

Ví dụ:

```
<!ELEMENT DOCUMENT (CUSTOMER)*>
```

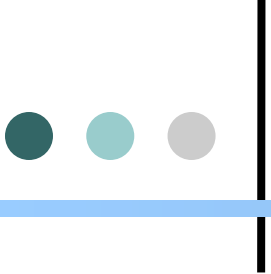
```
<!ELEMENT CUSTOMER(NAME)>
```

```
<!ELEMENT NAME (LAST_NAME, FIRST_NAME)>
```

```
<!ELEMENT LAST_NAME (#PCDATA)>
```

```
<!ELEMENT FIRST_NAME (#PCDATA)>
```

DOCUMENT là phần tử chứa phần tử con **CUSTOMER**;
CUSTOMER chứa phần tử con **NAME**; **NAME** chứa phần tử con **LAST_NAME** và **FIRST_NAME**



Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❖ Định nghĩa kiểu dữ liệu

Làm việc với nhiều phần tử con

A^* : không có hoặc có nhiều phần tử con giống nhau.

A^+ : có một hoặc nhiều phần tử con giống nhau.

$A?$: phần tử A hoặc không có phần tử nào cả

A, B : phần tử A tiếp đến phần tử B.

$A|B$: Phần tử A hoặc phần tử B nhưng không được cả hai.

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

Ví dụ:

```
<?xml version="1.0"?>
<!DOCTYPE DOCUMENT [
  <!ELEMENT CUSTOMER (#PCDATA)>
  <!ELEMENT DOCUMENT (CUSTOMER)*>]>
<DOCUMENT>
  <CUSTOMER>
    Sam smith
  </CUSTOMER>
  <CUSTOMER>
    Tony Braxton
  </CUSTOMER>
</DOCUMENT>
```

- CUSTOMER có thể xuất hiện 2 lần vì có kiểu *.

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

- ❖ Ta có thể sử dụng kí tự đại diện một nhóm thứ tự phần tử dựa vào dấu ().

Ví dụ:

<!ELEMENT DOCUMENT (CUSTOMER)*>

<!ELEMENT CUSTOMER ((NAME, CREDIT_RATING?)+, DATE*, ORDERS)>

- CUSTOMER có thể chứa một hoặc nhiều phần tử NAME, mỗi phần tử NAME lại kết hợp với phần tử CREDIT_RATING hoặc không (**NAME, CREDIT_RATING?**).
- Tổ hợp này lại được lặp lại một hoặc nhiều lần trong định nghĩa CUSTOMER (**NAME, CREDIT_RATING?**)**+**.

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

Ví dụ:

```
...
<!ELEMENT ORDERS (ITEM)*>
<!ELEMENT ITEM (PRODUCT, NUMBER, (PRICE | CHARGE C C T |
SAMPLE))>
...
<ORDERS>
<ITEM>
  <PRODUCT> Tomatoes </PRODUCT>
  <NUMBER> 8 </NUMBER>
  <PRICE> $1.25 </PRICE>
</ITEM>
<ITEM>
  <PRODUCT> Tomatoes </PRODUCT>
  <NUMBER> 8 </NUMBER>
  <SAMPLE> $1.25 </SAMPLE>
</ITEM>
</ORDERS>
```

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

- ❖ Định nghĩa nội dung hỗn hợp:
 - Một phần tử có thể chứa dữ liệu thuần text (#PCDATA) hoặc cũng có thể chứa dữ liệu thể hiện phần định dạng (Markup).

```
...  
<!ELEMENT PRODUCT (#PCDATA | PRODUCT_ID)>  
...  
<!ELEMENT PRODUCT_ID (#PCDATA)>  
...  
<ITEM>  
  <PRODUCT> Tomatoes </PRODUCT>  
  <NUMBER> 8 </NUMBER>  
  <PRICE> $1.25 </PRICE>  
</ITEM>  
<ITEM>  
  <PRODUCT>  
    <PRODUCT_ID>T123456 </PRODUCT_ID>  
  </PRODUCT>  
  <NUMBER> 8 </NUMBER>  
  <SAMPLE> $1.25 </SAMPLE>  
</ITEM>
```

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❑ External DTD trong XML:

- Được khai báo bên ngoài XML file
- DTD được lưu thành file `.dtd`
- Thuộc tính *standalone* trong khai báo XML phải được thiết lập là *no*
- Cú pháp:

`<!DOCTYPE root-element SYSTEM "file-name">`

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❑ Cú pháp của external DTD trong XML:

❖ Ví dụ về external DTD

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
```

```
<!DOCTYPE student SYSTEM "student.dtd" >
```

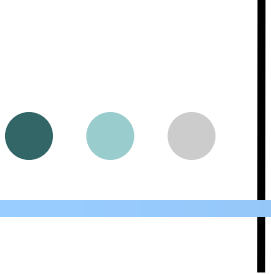
```
<student>
```

```
  <name>Trương Hồ</name>
```

```
  <address>123 Nguyễn Huệ</address>
```

```
  <phone>(08) 123-4567</phone>
```

```
</student>
```



Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❑ Cú pháp của external DTD trong XML:

❖ Ví dụ về external DTD

File `student.dtd`

```
<!ELEMENT student (name, address, phone)>
```

```
<!ELEMENT name (#PCDATA)>
```

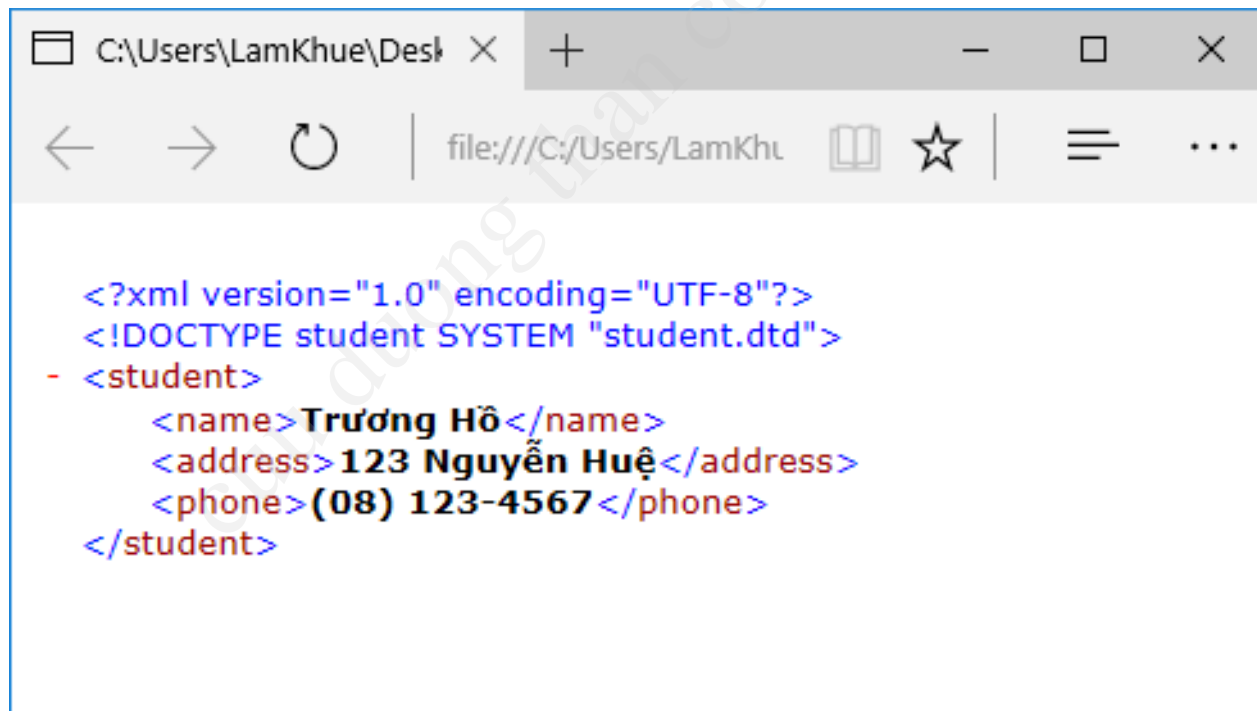
```
<!ELEMENT address (#PCDATA)>
```

```
<!ELEMENT phone (#PCDATA)>
```

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❑ Cú pháp của external DTD trong XML:

❖ Kết quả hiển thị



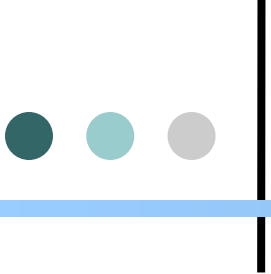
The screenshot shows a web browser window with the address bar displaying 'file:///C:/Users/LamKhue/Desk...'. The main content area displays an XML document with the following code:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE student SYSTEM "student.dtd">
- <student>
    <name>Trương Hồ</name>
    <address>123 Nguyễn Huệ</address>
    <phone>(08) 123-4567</phone>
</student>
```

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❖ Thực thể trong DTD

- Thực thể (Entity): là một mục dữ liệu mà XML tham chiếu đến.
- Thực thể gồm có:
 - ✓ Thực thể tổng quát (General Entity).
 - ✓ Thực thể tham số (Parameter Entity).
- Thực thể được khai báo trong phần định nghĩa DTD, được sử dụng khi tài liệu XML tham chiếu tới.
- Tham chiếu thực thể tổng quát bắt đầu bằng & và kết thúc bằng ;
- Tham chiếu thực thể tham số bắt đầu bằng % và kết thúc bằng ;



Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❖ Thực thể trong DTD

- Thực thể nội được định nghĩa hoàn toàn trong tài liệu tham chiếu đến nó.
- Thực thể ngoại (External Entity): nội dung của nó được định nghĩa hoàn toàn từ một nguồn dữ liệu bên ngoài (file) và được tham chiếu bằng địa chỉ URL hoặc URI.
- Thực thể có thể ở dạng phân tích (parse) hoặc có thể ở dạng văn bản thuần (text).
- Thực thể ở dạng phân tích thì phải hợp khuôn dạng tài liệu XML.

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❖ Thực thể trong DTD

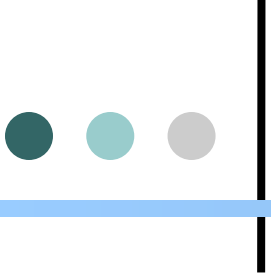
- Một số thực thể tổng quát được định nghĩa sẵn trong XML:
 - ☐ <;: hiển thị dấu <
 - ☐ >;: hiển thị dấu >
 - ☐ &;: hiển thị dấu &
 - ☐ ";: hiển thị dấu “
 - ☐ ';: hiển thị dấu ‘
- Các thực thể định nghĩa sẵn trong XML rất hữu dụng khi muốn thể hiện các kí tự đặc biệt trùng với kí tự của phần định dạng.

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❖ Khai báo thuộc tính

- Thuộc tính để bổ sung thêm các thông tin cho phần tử thẻ.
- Khai báo thuộc tính trong DTD:
 - ✓ Để tài liệu XML hợp lệ phải định nghĩa tất cả các thuộc tính mà tài liệu sử dụng
 - ✓ Chỉ định kiểu dữ liệu của thuộc tính, giá trị của thuộc tính.
- Cú pháp:

```
<!ATTRIBUTE ELEMENT_NAME  
ATTRIBUTE_NAME TYPE DEFAULT_VALUE  
ATTRIBUTE_NAME TYPE DEFAULT_VALUE  
....  
ATTRIBUTE_NAME TYPE DEFAULT_VALUE>
```



Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❖ Khai báo thuộc tính:

Trong đó

- ✓ **ELEMENT_NAME**: tên phần tử muốn áp đặt thuộc tính.
- ✓ **ATTRIBUTE_NAME**: tên thuộc tính
- ✓ **TYPE**: kiểu dữ liệu của thuộc tính
- ✓ **DEFAULT_VALUE**: là giá trị mặc định muốn gán cho thuộc tính.

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❖ Khai báo thuộc tính

Thuộc tính có thể mang các giá trị TYPE như sau:

Kiểu	Mô tả
CDATA	Kiểu dữ liệu kí tự đơn giản không chứa phần định dạng
ENTITIES	Tham chiếu thực thể (phải được định nghĩa trong DTD)
ENTITY	Tên tham chiếu của thực thể (cũng phải được khai báo trong DTD)
Enumerated	Đại diện cho một danh sách các giá trị mà thuộc tính có thể được gán
ID	Tên định danh duy nhất trong tài liệu XML

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❖ Khai báo thuộc tính

➤ Các giá trị DEFAULT_VALUE.

Giá trị	Mô tả
VALUE	Chuỗi giá trị văn bản thô bao bởi dấu nháy kép hoặc đơn ví dụ như “văn bản”
#IMPLIED	Chuỗi giá trị văn bản thô bao bởi dấu nháy kép hoặc đơn ví dụ như “văn bản” Chỉ định không có giá trị mặc định cho thuộc tính và thuộc tính này không cần phải dùng đến.
#REQUIRED	Chỉ định rằng không có giá trị mặc định cho thuộc tính nhưng phải gán giá trị cho thuộc tính khi sử dụng.
#FIXED VALUE	Trong trường hợp này VALUE là giá trị bắt buộc và cố định. Thuộc tính phải luôn mang giá trị VALUE này.

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❖ Khai báo thuộc tính

Ví dụ: khai báo thuộc tính cho phần tử CUSTOMER như sau:

```
<!ATTLIST CUSTOMER TYPE CDATA #IMPLIED>
```

Khi đó trong tài liệu XML, phần tử CUSTOMER có thể đưa thêm thuộc tính vào hoặc không.

...

```
<CUSTOMER TYPE="Good">
```

...

```
</CUSTOMER>
```

```
<CUSTOMER>
```

...

```
</CUSTOMER>
```

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❖ Khai báo thuộc tính

Ví dụ 2: khai báo một danh sách thuộc tính cho phần tử CUSTOMER như sau:

```
<!ATTLIST CUSTOMER OWES CDATA "0"  
                    LAYAWAY CDATA "0"  
                    DEFAULTS CDATA "0">
```

Khi đó trong tài liệu XML, phần tử CUSTOMER có các thuộc tính như sau:

...

```
<CUSTOMER OWES="$12" LAYAWAY="$0" DEFAULT="0">
```

...

```
</CUSTOMER>
```

```
<CUSTOMER OWES="$0" LAYAWAY="$34" DEFAULT="0">
```

...

```
</CUSTOMER>
```


Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❖ Thiết lập giá trị mặc định cho thuộc tính

- Giá trị tức thời (immediately value): có thể đặt giá trị mặc định cho thuộc tính ngay khi khai báo trong `<!ATTLIST`

Ví dụ:

```
<!ATTLIST CUSTOMER OWES CDATA "0" LAYAWAY CDATA "0"  
          DEFAULTS CDATA "0">
```

- ✓ `#REQUIRED`: không cần cung cấp giá trị ngay khi khai báo nhưng phải cung cấp giá trị khi sử dụng.

Ví dụ: `<!ATTLIST CUSTOMER OWES CDATA #REQUIRED>`

- ✓ `#IMPLIED`: chỉ định này không cần đặt giá trị mặc định cho thuộc tính và cũng không bắt buộc đưa giá trị vào cho thuộc tính khi sử dụng

Ví dụ: `<!ATTLIST CUSTOMER OWES CDATA #IMPLIED>`

- ✓ `#FIXED`: chỉ định giá trị cố định cho thuộc tính. Người dùng không thể thay đổi giá trị thuộc tính được ngoài giá trị mặc định.

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❖ Định kiểu thuộc tính

➤ CDATA

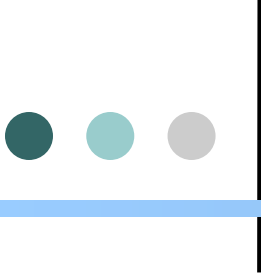
- ✓ Là kiểu dữ liệu thuần text.
- ✓ Không thể sử dụng được các kí tự đặc biệt như <, &,". Nếu muốn dùng phải dùng tham chiếu tổng quát.

➤ Kiểu Liệt Kê (Enumerated)

- ✓ Được quy định bằng một danh sách các giá trị có thể gán cho thuộc tính.
- ✓ Mỗi giá trị của thuộc tính trong danh sách phải phù hợp với cách đặt tên của đặc tả XML.

Ví dụ: <!ATTLIST CUSTOMER

CREDIT_OK (TRUE | FALSE) "TRUE">



Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❖ Định kiểu thuộc tính

➤ NMTOKEN

- ✓ Thuộc tính có kiểu dữ liệu này chỉ được gán các giá trị hợp quy tắc đặt tên của XML và không có khoảng trắng.
- ✓ Thuộc tính chỉ có thể mang giá trị là một từ đơn (vì khoảng trắng và các kí tự phân cách không được thể hiện)

➤ NMTOKENS

- ✓ Tuân theo quy tắc đặt tên của XML
- ✓ Giá trị của thuộc tính là một chuỗi bao gồm nhiều token và phân tách nhau bằng khoảng trắng.

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❖ Định kiểu thuộc tính

ID (định danh)

Giá trị của thuộc tính kiểu ID dùng làm tên duy nhất để định danh một phần tử nào đó trong tài liệu.

Không được có hai phần tử trùng tên khi mang kiểu định danh.

Chỉ có thể định nghĩa một thuộc tính mang kiểu định danh ID trong một phần tử.

Ví dụ: `<!ATTLIST CUSTOMER CUSTOMER_ID ID
#REQUIRED>`

Trong tài liệu XML:

`<CUSTOMER CUSTOMER_ID="1234">...</CUSTOMER>`

`<CUSTOMER CUSTOMER_ID="1234"> → Lỗi`

...

`</CUSTOMER>`

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❖ Định kiểu thuộc tính

➤ IDREF

- ✓ Cho phép xác định thông tin liên quan đến cấu trúc tài liệu (liên quan đến các phần tử trong tài liệu).
- ✓ IDREF nắm giữ các giá trị ID của các phần tử khác.
- ✓ Ví dụ: muốn thiết lập quan hệ cha con giữa các phần tử không nằm lồng nhau trong tài liệu. Trong trường hợp này ta phải định nghĩa thuộc tính có kiểu IDREF.

```
<!ATTLIST CUSTOMER  
    CUSTOMER_ID ID #REQUIRED  
    EMPLOYER_ID IDREF #IMPLIED>
```

Trong tài liệu XML

```
<CUSTOMER CUSTOMER_ID="1234">
```

....

```
</CUSTOMER>
```

```
<CUSTOMER CUSTOMER_ID="1235"
```

```
    EMPLOYER_ID="1234">
```

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❖ Định kiểu thuộc tính

➤ Kiểu thực thể ENTITY

- ✓ Thuộc tính có thể được gán giá trị là một thực thể do bạn khai báo trước đó.
- ✓ Ví dụ: ta khai báo một thực thể mang tên SNAPSHOT tham chiếu đến một file ảnh bên ngoài. Tiếp đến ta có thể hoàn toàn có thể tạo ra một thuộc tính mới mang giá trị là IMAGE. Thuộc tính này có thể gán giá trị là SNAPSHOT

```
<!ATTLIST CUSTOMER IMAGE ENTITY #IMPLIED>  
<!ENTITY SNAPSHOT SYSTEM "image.gif">
```

Trong tài liệu XML:

```
<CUSTOMER IMAGE="SNAPSHOT">
```

...

```
</CUSTOMER>
```

Định nghĩa kiểu tài liệu và Kiểm tra tính hợp lệ của XML

❖ Định kiểu thuộc tính

➤ Kiểu đa thực thể (ENTITIES)

- ✓ Cho phép gán nhiều thực thể vào thuộc tính
- ✓ Các thực thể được tổ chức thành một danh sách cách nhau bằng khoảng trắng.
- ✓ Ví dụ:

```
<!ATTLIST CUSTOMER
```

```
    IMAGE ENTITIES #IMPLIED>
```

```
<!ENTITY SNAPSHOT1 SYSTEM "image1.gif">
```

```
<!ENTITY SNAPSHOT2 SYSTEM "image2.gif">
```

Trong tài liệu XML

```
<CUSTOMER IMAGE ="SNAPSHOT1 SNAPSHOT2">
```

```
....
```

```
</CUSTOMER>
```



Lược đồ XML (XML Schema)

- ❖ Dễ dàng để mô tả nội dung tài liệu vì dùng chính cú pháp XML để định nghĩa.
- ❖ Dễ kiểm tra tính hợp lệ của tài liệu.
- ❖ Dễ định nghĩa về mặt dữ liệu.
- ❖ Dễ dàng định nghĩa dữ liệu mẫu.
- ❖ Dễ chuyển đổi kiểu dữ liệu này sang kiểu dữ liệu khác

Lược đồ XML (XML Schema)

❖ Ví dụ: tạo file .xsd

```
<?xml version="1.0"?>
```

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.w3schools.com"
xmlns="http://www.w3schools.com" elementFormDefault="qualified">
```

```
  <xs:element name="student">
```

```
    <xs:complexType>
```

```
      <xs:sequence>
```

```
        <xs:element name="name" type="xs:string"/>
```

```
        <xs:element name="address" type="xs:string"/>
```

```
        <xs:element name="phone" type="xs:string"/>
```

```
      </xs:sequence>
```

```
    </xs:complexType>
```

```
  </xs:element>
```

```
</xs:schema>
```

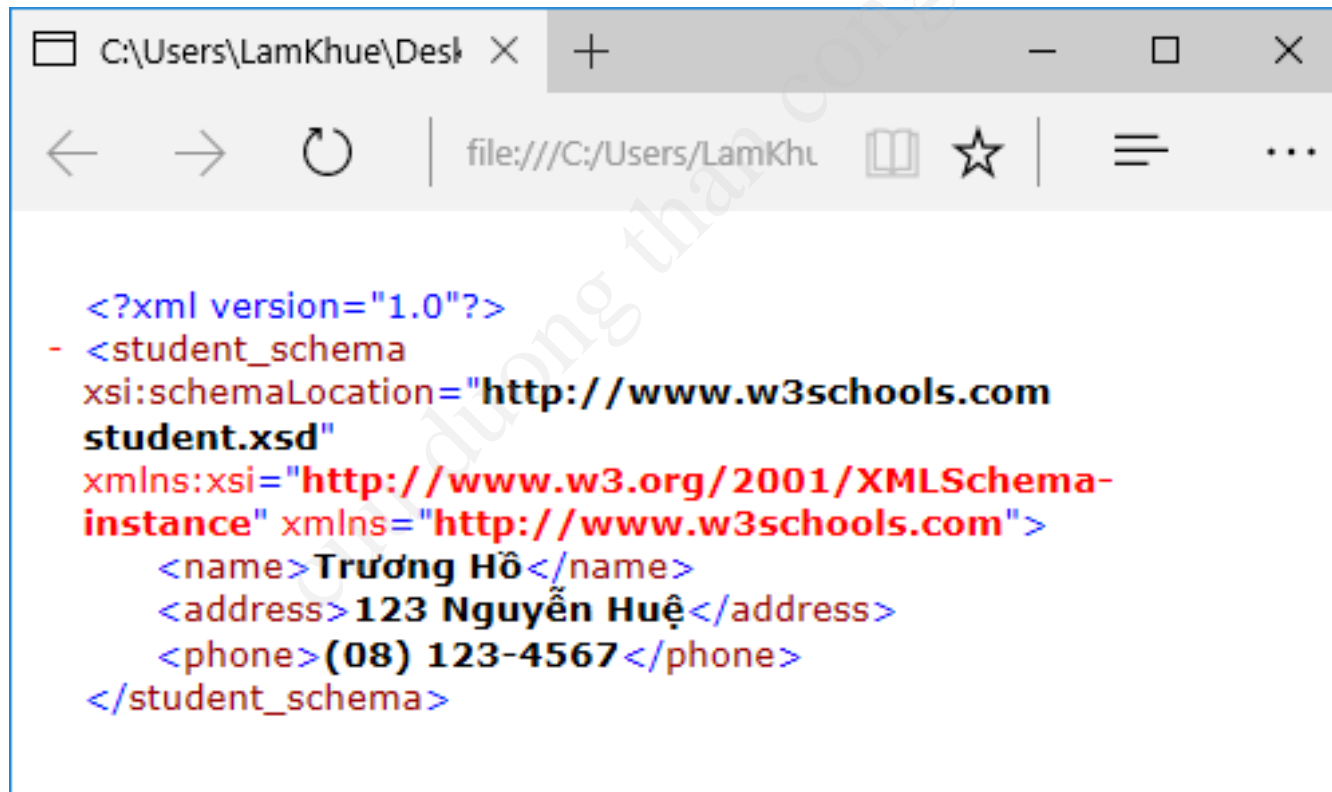
Lược đồ XML (XML Schema)

- ❖ Ví dụ (tt):
tạo file **.xml**
tham chiếu
đến lược đồ
xml (file
.xsd)

```
<?xml version="1.0"?>
  <student_schema
    xmlns="http://www.w3schools.com"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.w3schools.com student.xsd">
    <name>Trương Hồ</name>
    <address>123 Nguyễn Huệ</address>
    <phone>(08) 123-4567</phone>
  </student_schema>
```

Lược đồ XML (XML Schema)

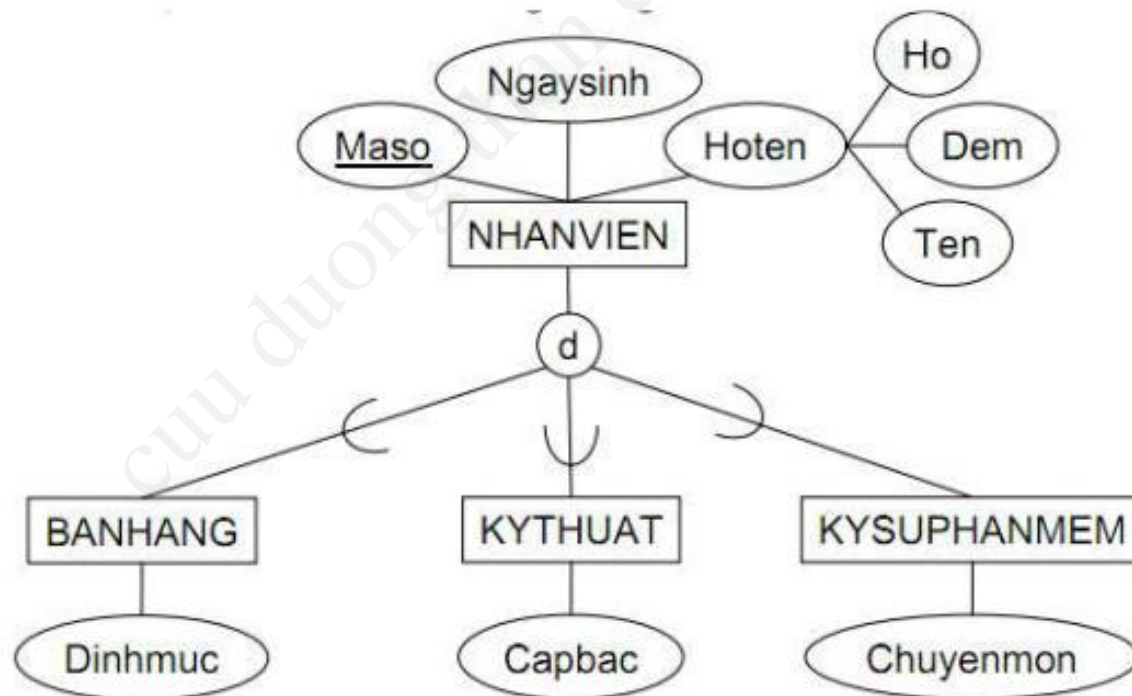
❖ Ví dụ (tt): Kết quả hiển thị



```
<?xml version="1.0"?>
- <student_schema
  xsi:schemaLocation="http://www.w3schools.com
  student.xsd"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-
  instance" xmlns="http://www.w3schools.com">
    <name>Trương Hồ</name>
    <address>123 Nguyễn Huệ</address>
    <phone>(08) 123-4567</phone>
  </student_schema>
```

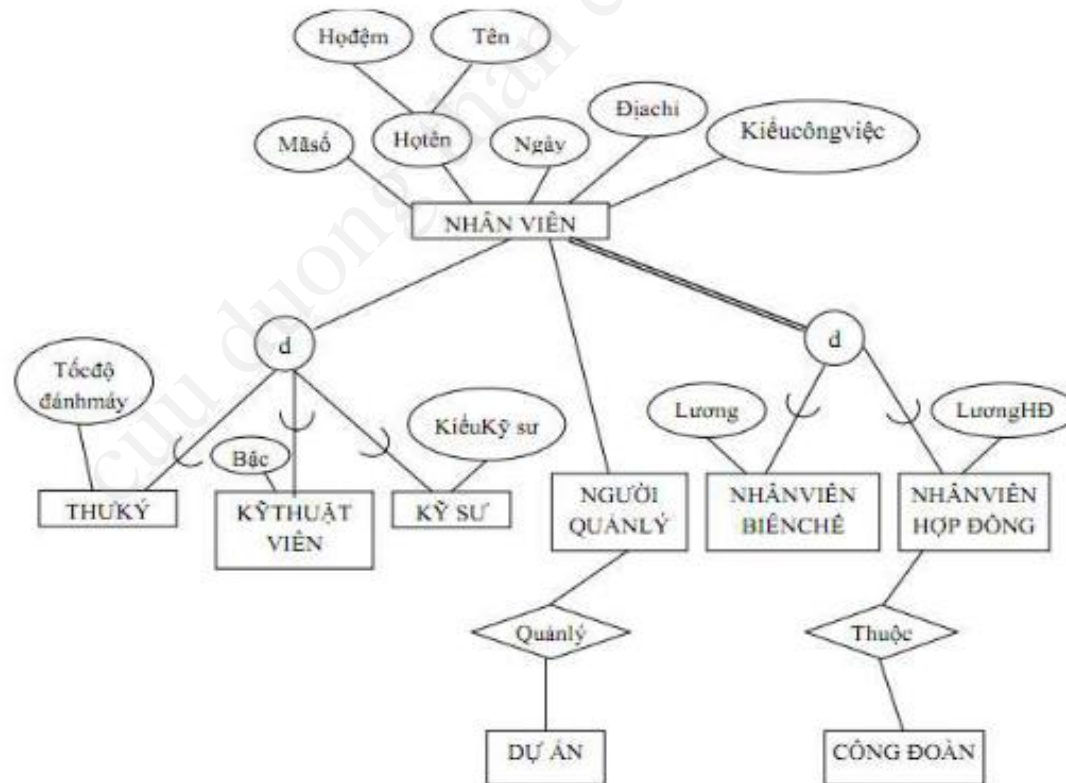
Mô hình thực thể liên kết mở rộng

- **Lớp cha và lớp con:** Một kiểu thực thể có thể có các nhóm con các thực thể của nó



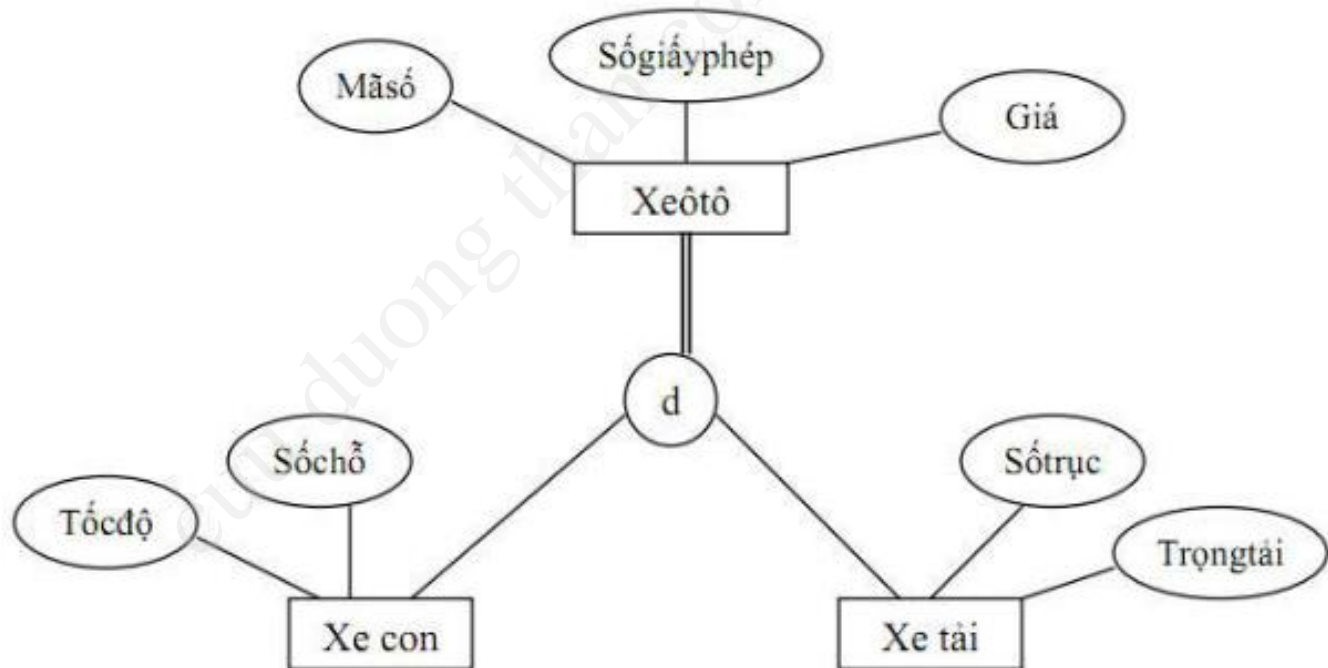
Mô hình thực thể liên kết mở rộng

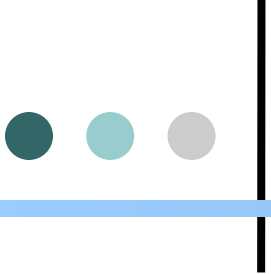
- **Chuyên biệt hóa:** Là quá trình xác định tập hợp các lớp con của một kiểu thực thể



Mô hình thực thể liên kết mở rộng

➤ Tổng quát hóa:



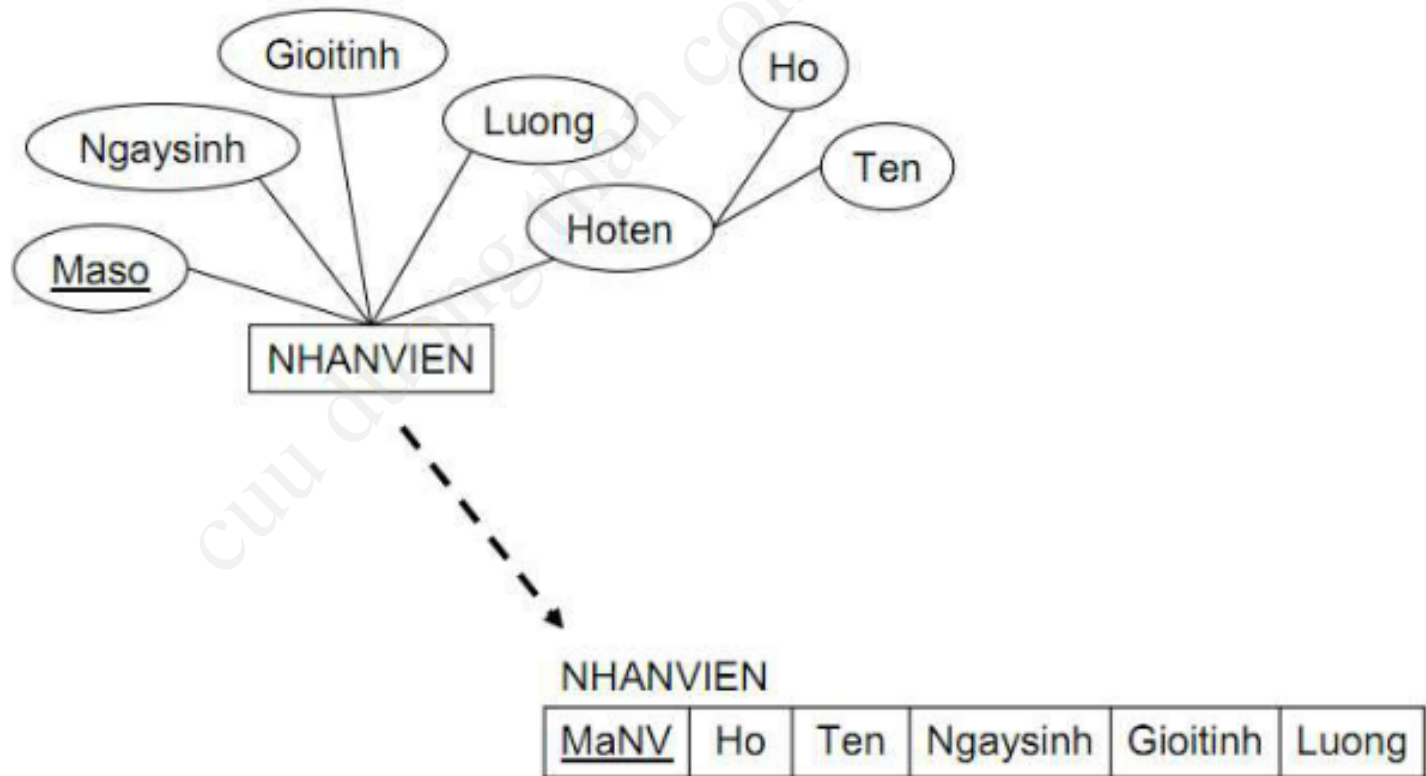


Chuyển mô hình thực thể liên kết sang mô hình quan hệ

- Quy tắc:
- ✓ Mỗi tập thực thể trong ER chuyển thành một lược đồ quan hệ.
- ✓ Mỗi thuộc tính trong mô hình ER chuyển thành thuộc tính trong lược đồ quan hệ.
- ✓ Mỗi thuộc tính nhận diện trong mô hình ER được chuyển thành khóa chính trong lược đồ quan hệ.

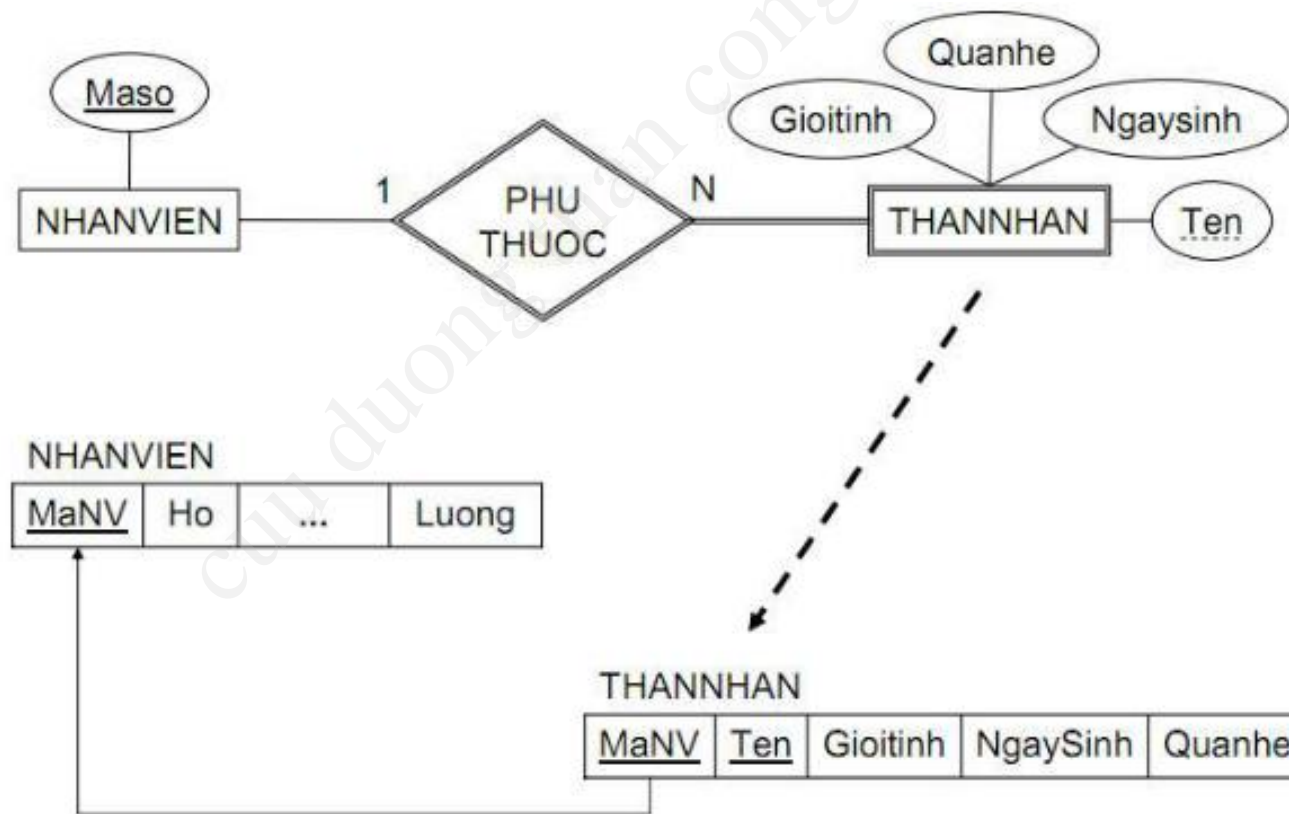
Chuyển mô hình thực thể liên kết sang mô hình quan hệ

- Ví dụ: Thực thể mạnh



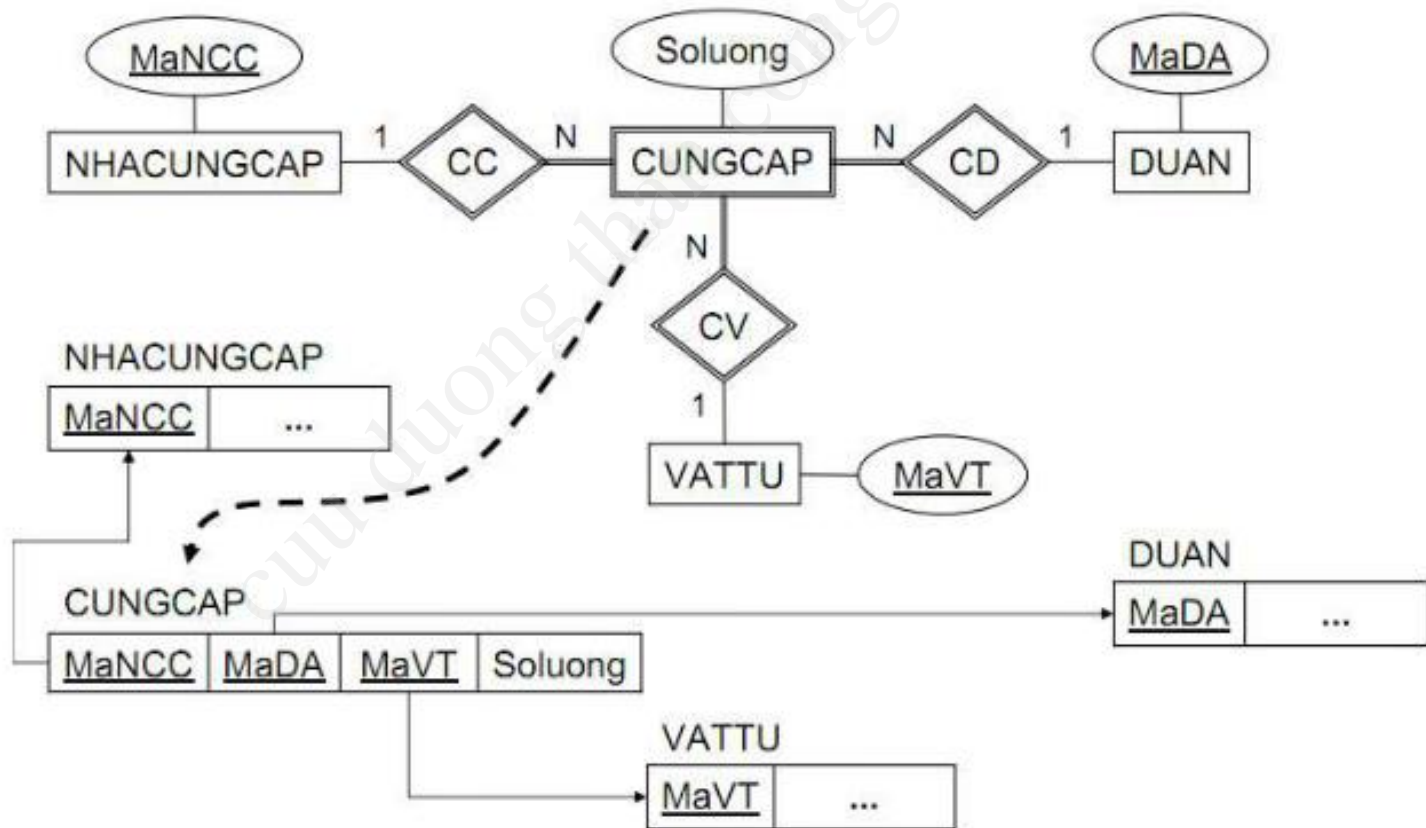
Chuyển mô hình thực thể liên kết sang mô hình quan hệ

➤ Ví dụ: Thực thể yếu



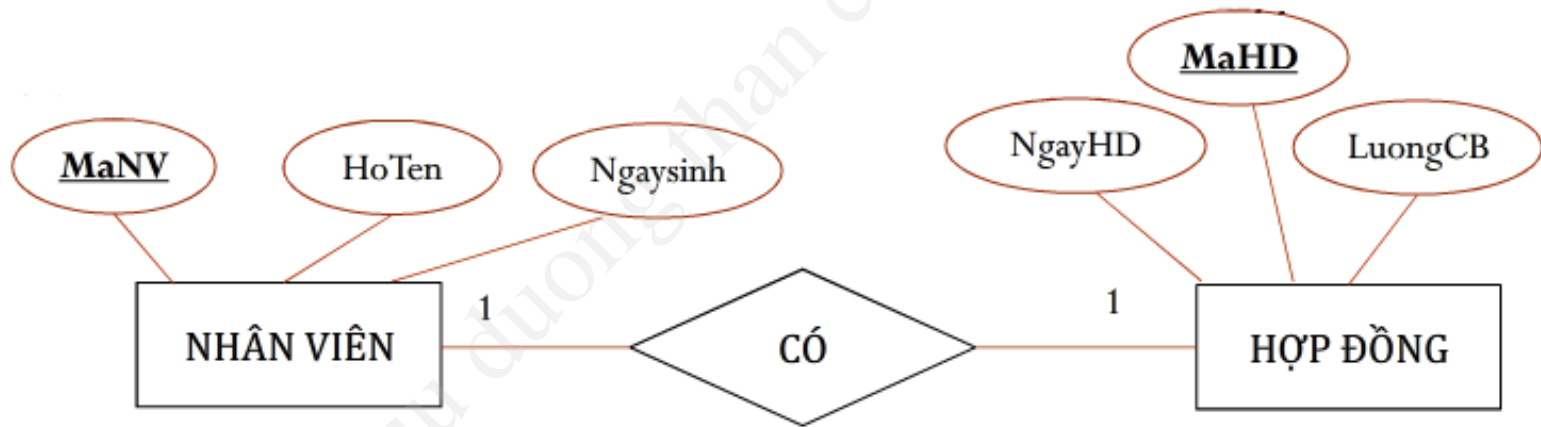
Chuyển mô hình thực thể liên kết sang mô hình quan hệ

➤ Ví dụ: Thực thể yếu (tt)



Chuyển mô hình thực thể liên kết sang mô hình quan hệ

- **Quan hệ 1-1:** Chuyển khóa chính từ quan hệ thứ nhất sang quan hệ thứ hai và ngược lại

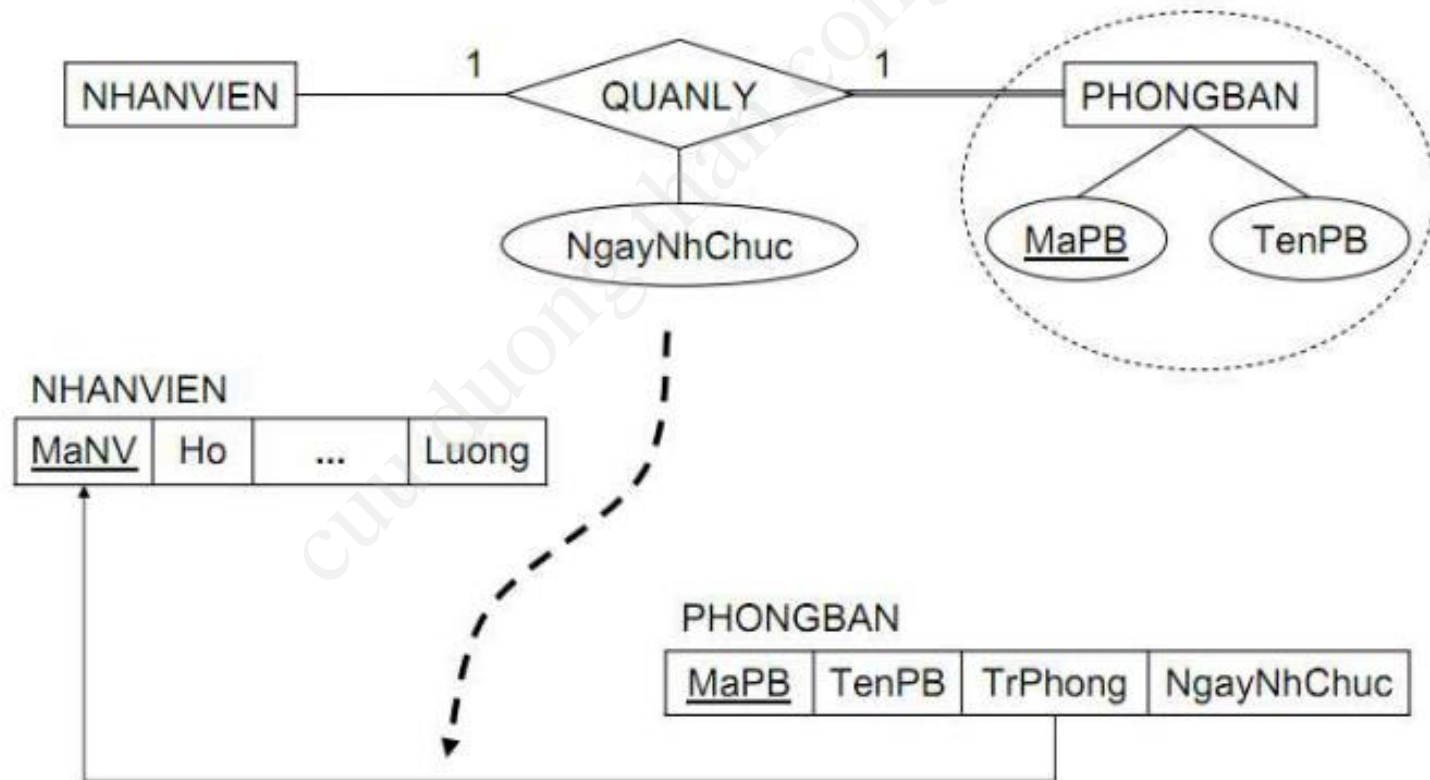


NhanVien(MaNV, HoTen, Ngaysinh)

Hopdong(MaHD, NgayHD, LuongCB, *MaNV*)

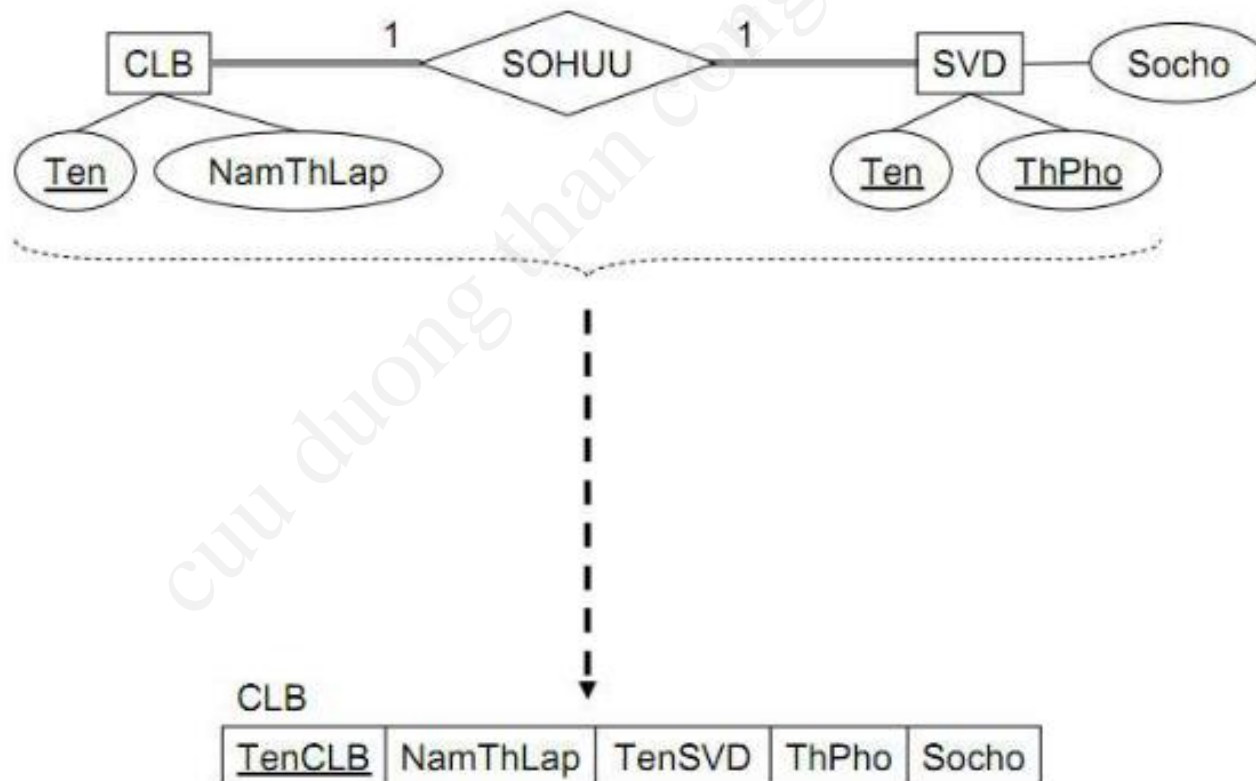
Chuyển mô hình thực thể liên kết sang mô hình quan hệ

➤ Quan hệ 1-1 (tt):



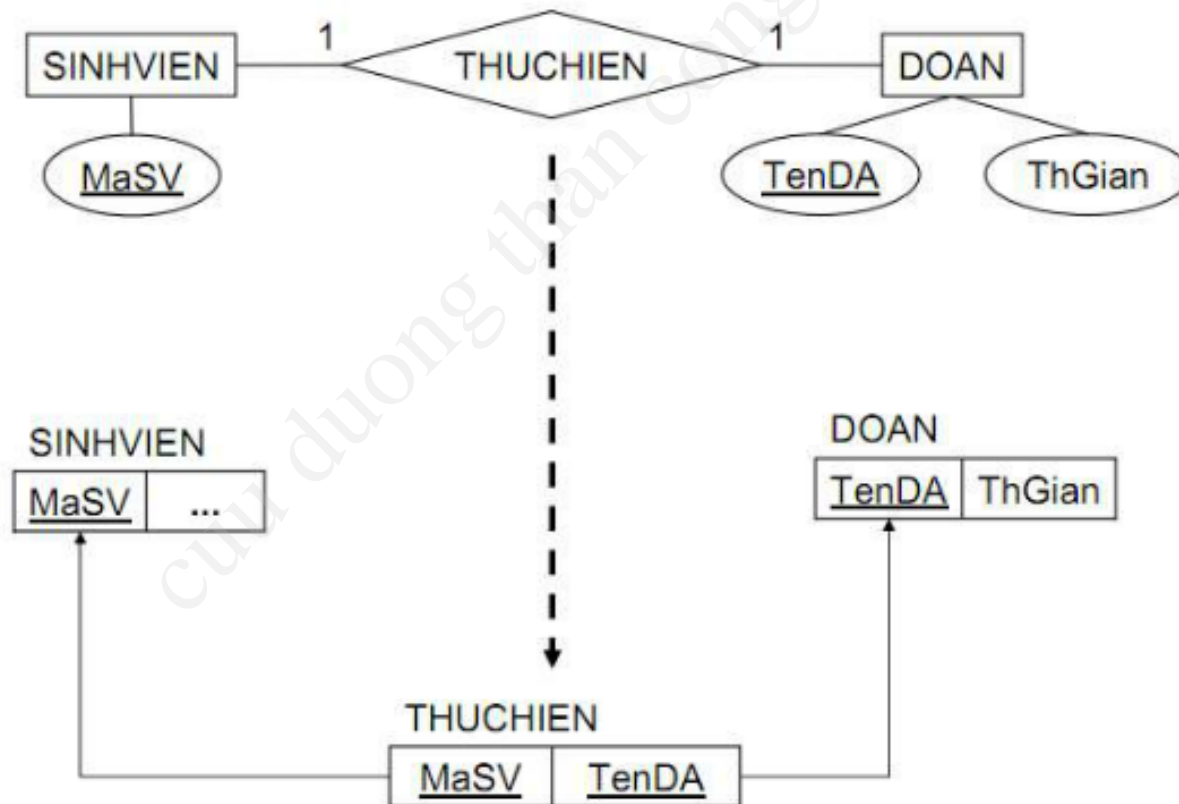
Chuyển mô hình thực thể liên kết sang mô hình quan hệ

➤ Quan hệ 1-1 (tt):



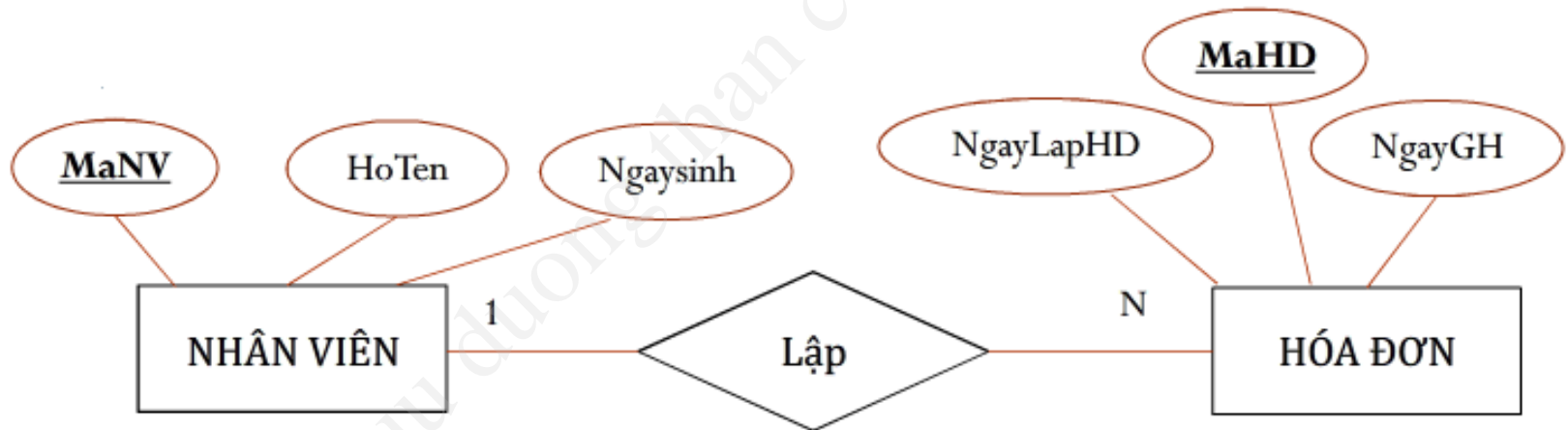
Chuyển mô hình thực thể liên kết sang mô hình quan hệ

➤ Quan hệ 1-1 (tt):



Chuyển mô hình thực thể liên kết sang mô hình quan hệ

- **Quan hệ 1-n:** Chuyển khóa chính từ bên một sang bên nhiều

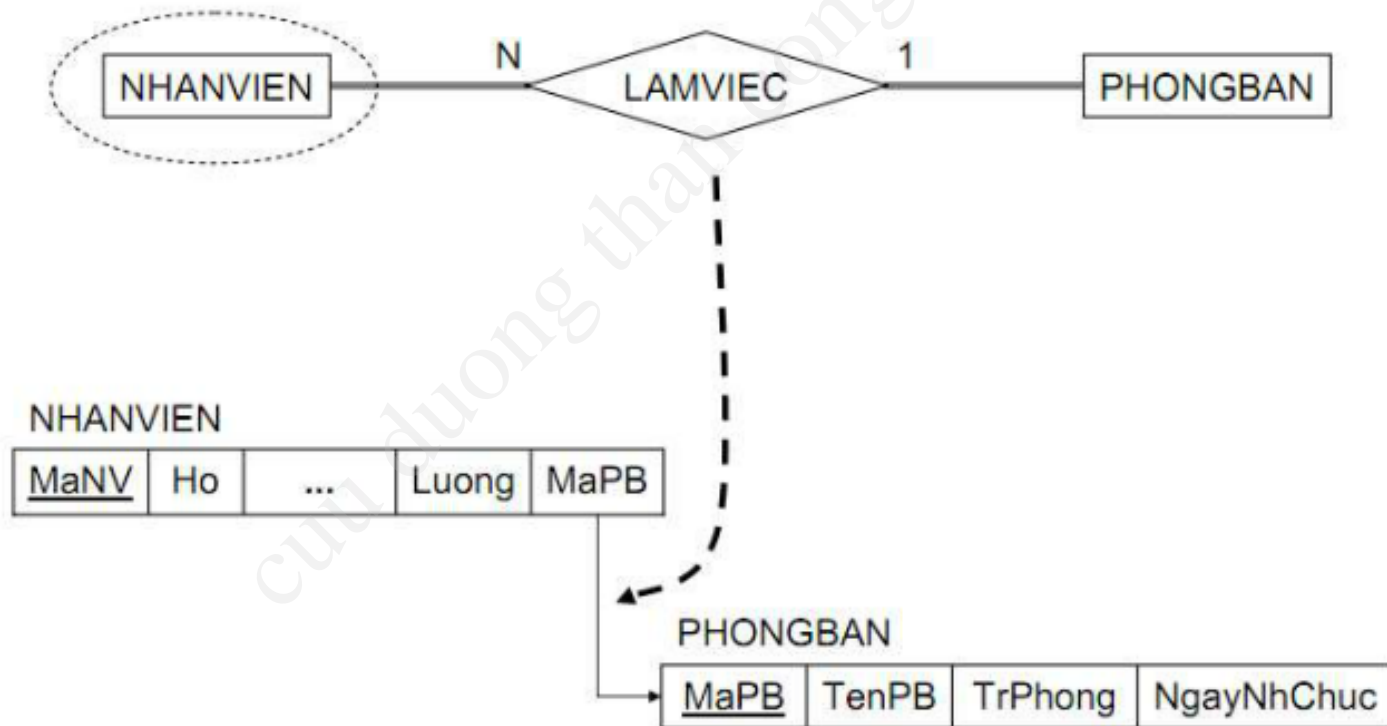


NhanVien(MaNV, HoTen, Ngaysinh)

Hoadon(MaHD, NgayLapHD, NgayGH, MaNV)

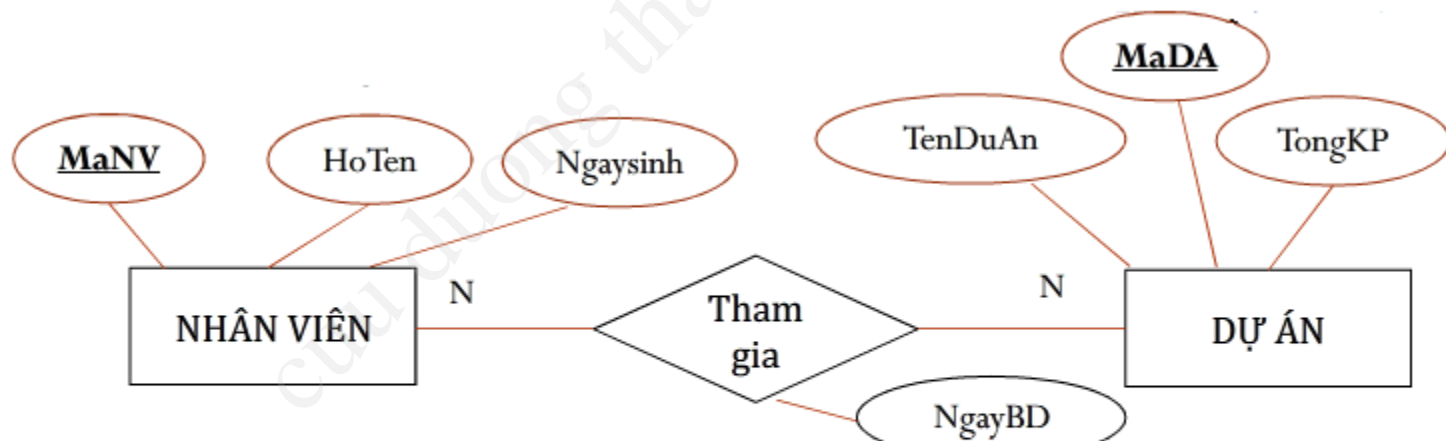
Chuyển mô hình thực thể liên kết sang mô hình quan hệ

➤ Quan hệ 1-n (tt):



Chuyển mô hình thực thể liên kết sang mô hình quan hệ

- **Quan hệ n-n:** Tạo một quan hệ mới có khóa chính là sự kết hợp các khóa chính của hai quan hệ có liên kết nhiều – nhiều



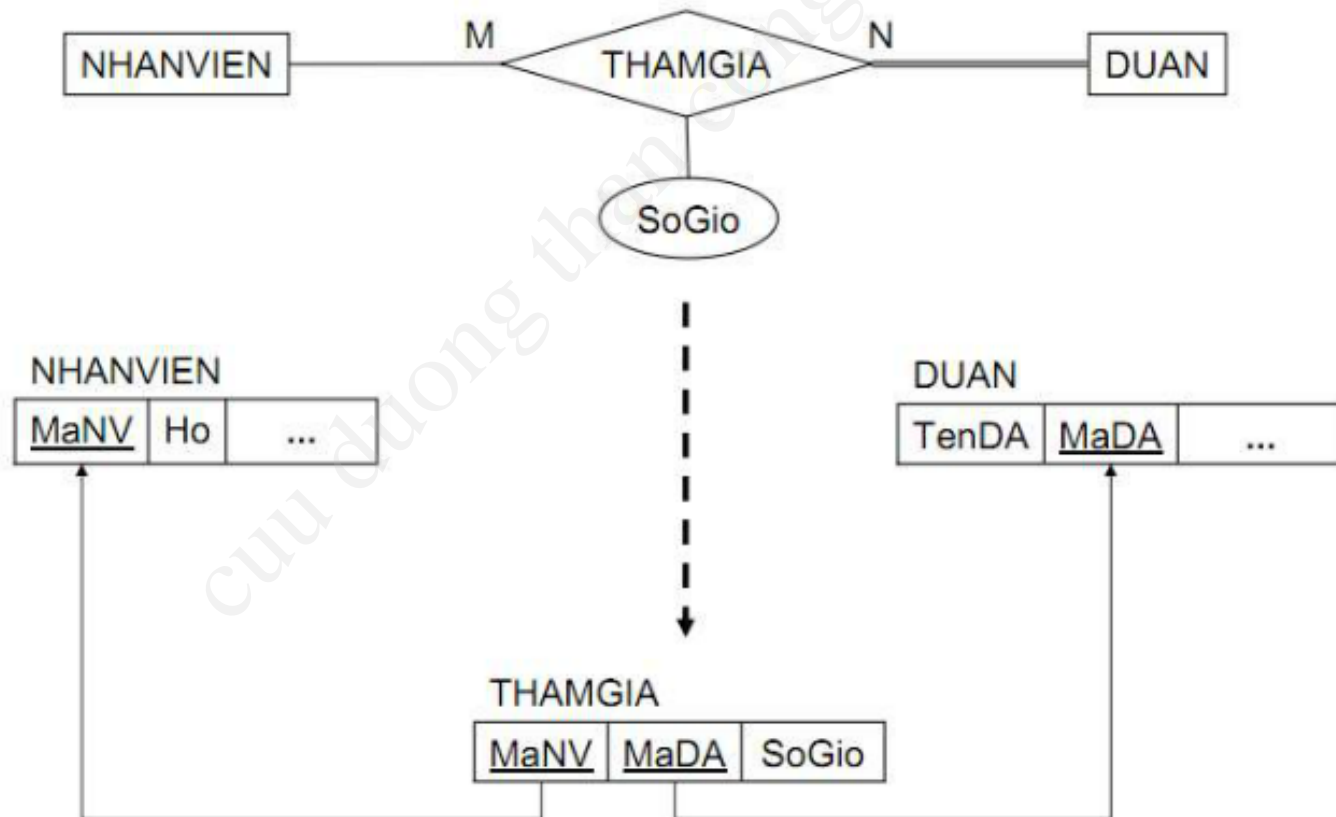
NhanVien(MaNV, HoTen, Ngaysinh)

DuAn(MaDA, TenDuAn, TongKP)

Thamgia(MaNV, MaDA, NgayBD)

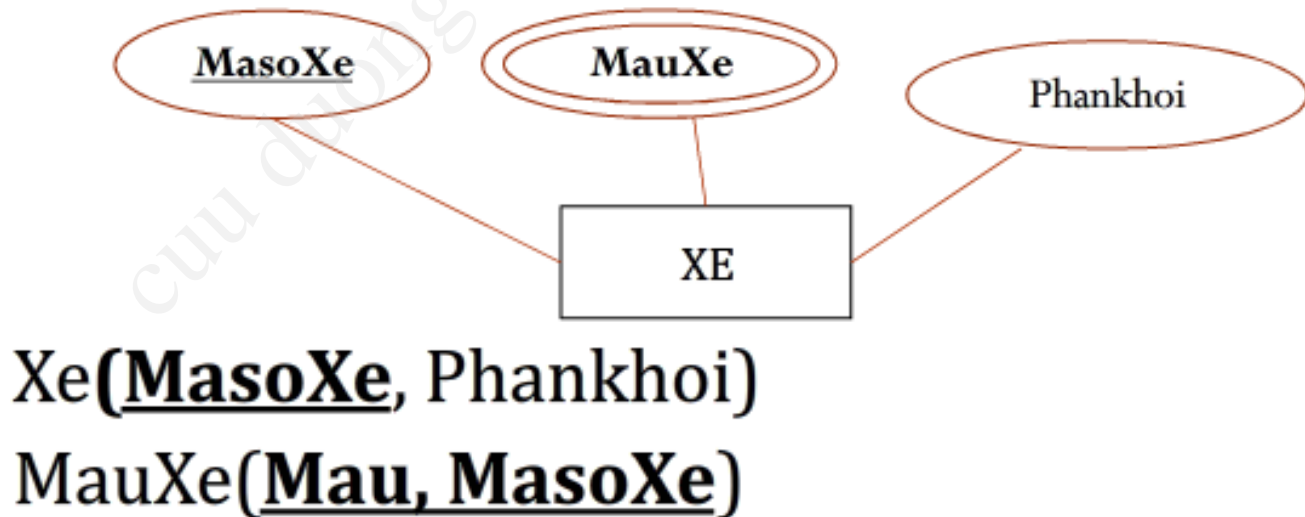
Chuyển mô hình thực thể liên kết sang mô hình quan hệ

➤ Quan hệ n-n (tt):



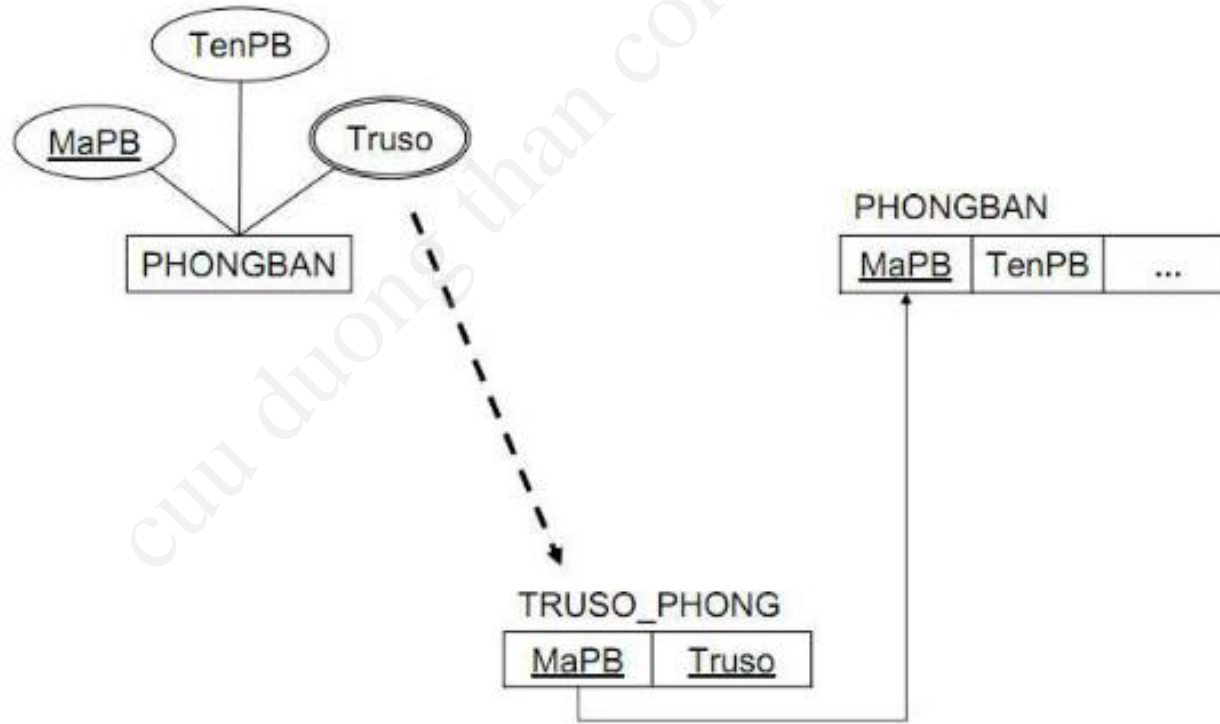
Chuyển mô hình thực thể liên kết sang mô hình quan hệ

- Thuộc tính đa trị A: Tạo một quan hệ mới R. Quan hệ R sẽ chứa một thuộc tính tương ứng với A và thuộc tính khóa K của thực thể chủ làm khóa ngoại của R. Khóa chính của R là một tổ hợp của A và K



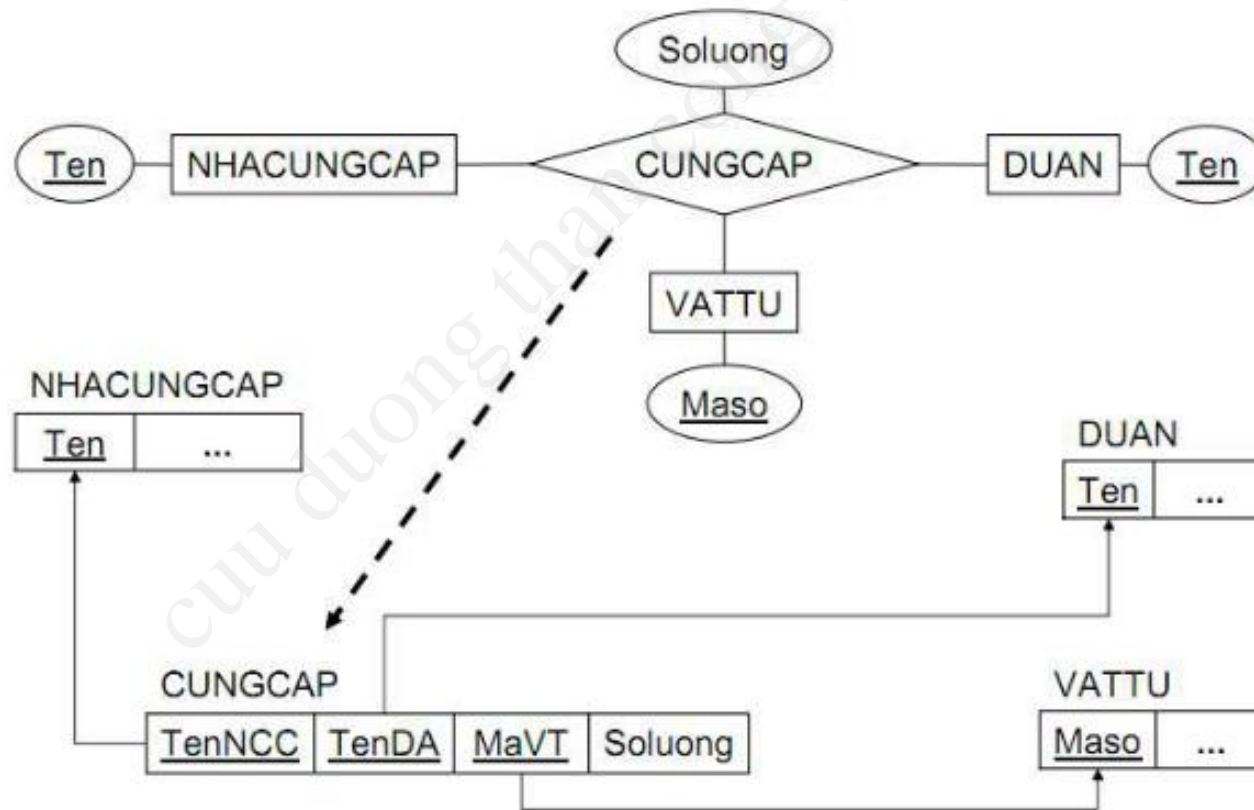
Chuyển mô hình thực thể liên kết sang mô hình quan hệ

- Thuộc tính đa trị A (tt):



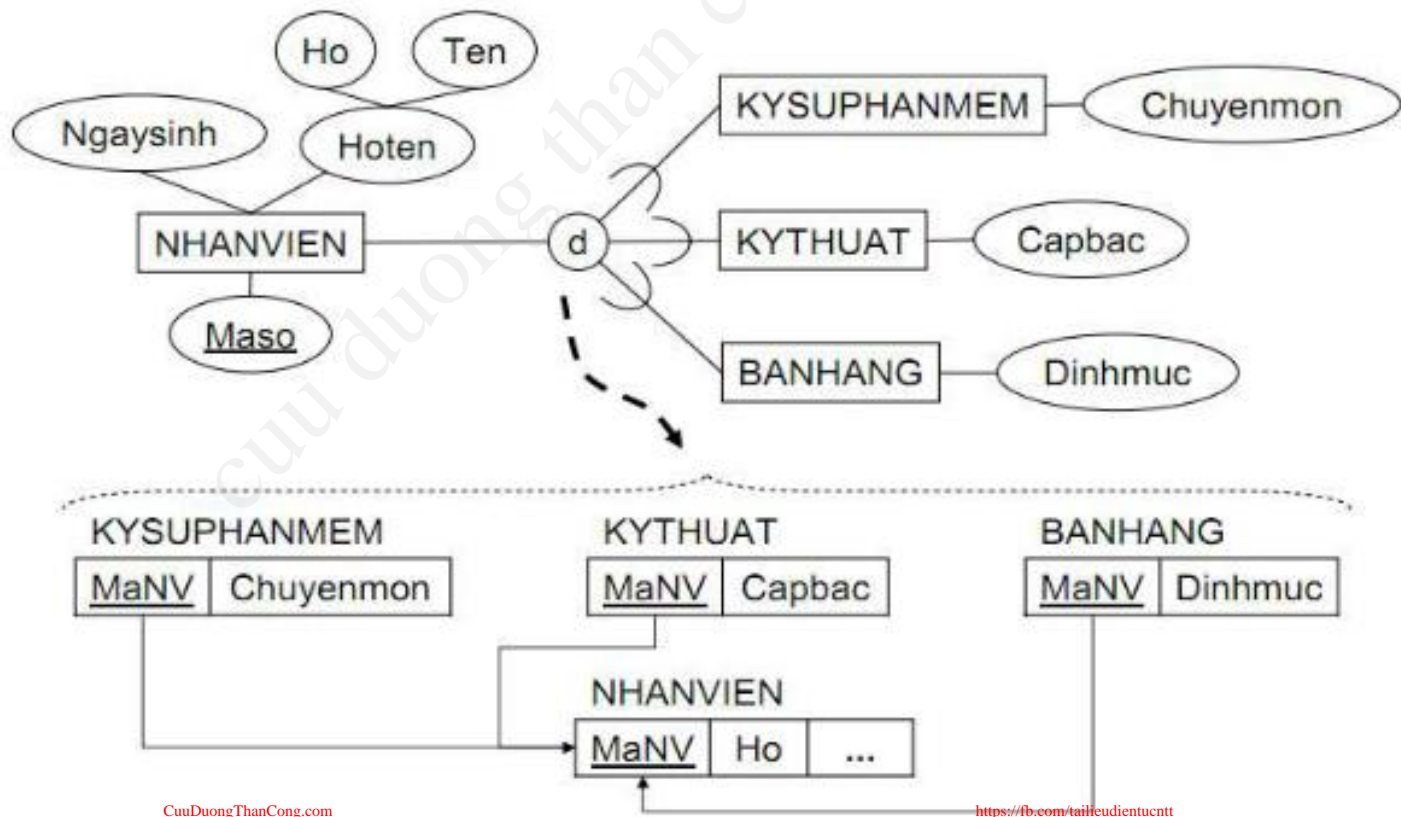
Chuyển mô hình thực thể liên kết sang mô hình quan hệ

➤ Kiểu liên kết Ternary ($n > 2$)



Chuyển mô hình thực thể liên kết mở rộng sang mô hình quan hệ

- Chuyển mô hình chuyên biệt hóa và tổng quát hóa sang mô hình quan hệ:



Xây dựng mô hình EER cho cơ sở dữ liệu ĐAOTAO (xác định các thực thể, quan hệ, ràng buộc)

- Trường có nhiều khoa: Mỗi một khoa, Thông tin về Khoa gồm Mã khoa, tên khoa, địa chỉ, số điện thoại.
- Mỗi Khoa cung cấp nhiều môn học. Mỗi môn học gồm có Tên môn học, mã số, số đơn tín chỉ.
- Cán bộ của khoa có thể là Hành chính hoặc Giảng viên; Giảng viên có thể là Cơ hữu hoặc trợ giảng (trợ giảng là học viên cao học). Mỗi khoa có nhiều cán bộ làm việc, nhưng mỗi cán bộ chỉ làm việc cho một khoa. Mỗi một khoa có một chủ nhiệm khoa, đó là một Giảng viên.
- Mỗi giáo viên có thể dạy nhiều nhất là 4 học phần và cũng có thể không dạy học phần nào.
- Học viên có thể là sinh viên ĐH hoặc cao học, Mỗi học viên phải học nhiều học phần.
- Mỗi một khoa có nhiều học viên, mỗi học viên chỉ thuộc về một khoa.
- Mỗi sinh viên đại học có một giáo viên hướng dẫn, một giáo viên có thể hướng dẫn nhiều sinh viên.