**XML VÀ CƠ SỞ DỮ LIỆU INTERNET**

# World Wide Web

Web giúp truy xuất file ở bất kỳ đâu trên Internet. Một file được xác định bởi universal resource locator (URL):

*http://www.hcmus.edu.vn/view/main/index.html*

URL trên xác định một file gọi là index.html, được lưu trữ trong thư mục view/main ở máy chủ [www.hcmus.edu.vn](http://www.hcmus.edu.vn). File này được định dạng sử dụng HyperText Markup Language (HTML) và chứa một vài link đến các file khác được xác định thông qua URL

Những đoạn lệnh có định dạng được thông dịch bởi web browser để hiển thị tài liệu cho người dùng và người dùng có thể gọi đến các tài liệu khác bằng cách chọn các link. Khi người dùng nhấp vào link, Web browser connect vào webserver ở điểm đến là website sử dụng protocol gọi là HTTP và submit URL của link. Tập hợp những tài liệu như vậy được gọ là webssite và được qản lý bằng cách sử dụng một chương trình gọi là Web server, nó sẽ chấp nhận URL và trả về các tài liệu tương ứng. World Wide Web lad tập hợp các website có thể truy xuất trên internet.

## Khái quát về HTML:

HTML là một ngôn ngữ đơn giản được sử dụng để miêu tả một tài liệu. Nó còn được gọi là markup languague vì HTML làm việc bằng cách bổ sung các đoạn text thông thường với mark có ý nghĩa quan trọng cho web browser xử lý tài liệu.

*<head>*

*<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />*

*<title>Tin nhanh VnExpress - Đọc báo, tin tức online 24h</title>*

*</head>*

Các đoạn lệnh trong HTML được gọi là tag và chúng bao gồm tag bắt đầu (start tage) và tag kết thúc (end tag) <TAG> và </TAG> theo thứ tự. HTML định nghĩa một tập các tag giống như ý nghĩa của tag, ví dụ <TITLE> là một tag đánh dấu tiêu đề của tài liệu

# Kiến trúc

Để thực thi một chương trình ở web server, server tạo một tiến trình mới và giao tiếp với tiến trình bằng cách sử dụng giao thức common gateway interface (CGI). Kết quả của chương trình có thể được sử dụng để tạp một file HTML phản hồi cho người yêu cầu. Các trang được tính toán theo cách này tại thời gian chúng được yêu cầu được gọi là trang động (dynamic page, các trang tồn tại và được phân phối một cách đơn giản đến Web browser được gọi là trang tĩnh (static page).

*<HTML><HEAD><TITILE> The database bookstore </TITLE></HEAD>*

*<BODY>*

*<FORM action=”find\_books.cgi” method=post>*

*Type an author name:*

*<INPUT type=”text” name=”authorName” size=30 maxLength=50>*

*<INPUT type=”submit” value=”Send it”>*

*<INPUT type=”reset” value=”Clear form”>*

*</FORM>*

*</BODY></HTML>*

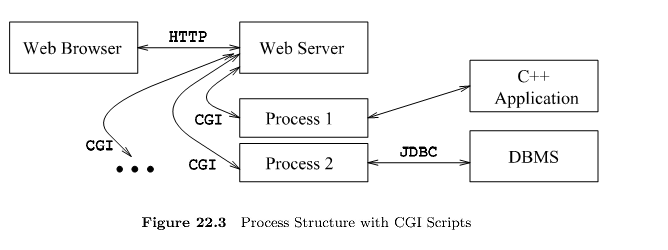
Hãy xem xét ví dụ ở trên. Trang web này chứa một biểu mẫu người dùng điền vào tên của tác giả. Nếu người dùng nhấn vào button “Send it”, Perl script “find\_books.cgi” sẽ thực thi như một tiến trình độc lập. Giao thức CGI định nghĩa giao tiếp của biểu mẫu và đoạn script được thể hiện như thế nào.

Hình 22.4 là một ví dụ của CGI scipt. Chúng tôi đã loại bỏ đoạn lệnh kiểm tra lỗi để đơn giản hóa. Perl là một ngôn ngữ thông dịch thường được sử dụng cho CGI và có nhiều thư viện Perl được gọi là module cung cấp interface cấp cao cho giao thức CGI. Chúng ta sử dụng 2 thư viện trong ví dụ: DBI và CGI. DBI là một API độc lập cơ sở dữ liệu cho Perl cho phép trừu tượng hóa từ DBMS được sử dụng – DBI đóng vai trò trong Perl như JDBC trong Java. Ở đây ta sử dụng DBI như cầu nối đến ODBC driver quản lý kết nối thật đến cơ sở dữ liệu. CGI module là một tập hợp các hàm tạo ra CGI script. Trong phần 1 của scipt mẫu, chúng tôi đã trích ra nội dung của biểu mẫu HTML như sau:

*$authorName = $dataIn -> param(‘authorName’);*

Chú ý rằng tên tham số authorName được sử dụng trong Hình 22.2 để gọi tên field input đầu tiên. Trong phần 2 chúng ta xây dựng một câu lệnh SQL thật trong biến $sql. Trong phần 3, ta bắt đầu dịch mã của trang để trả về cho trình duyệt. Ta muốn hiển thị các dòng kết quả của truy vấn nhhư một những entry trong một list không sắp xếp, chúng ta dùng tag <UL>. Các entry độc lập trong list được bao quanh bởi tag <LI>. Protocol CGI sẽ tóm tắt quá trình xử lý làm sao một trang web được trả về trình duyệt. Vì vậy mọi thứ mà đoạn script viết trong câu lệnh write sẽ là một phần của cấu trúc động của trang web được trả về cho trình duyệt. Phần 4 sẽ thiết lập một kết nối cơ sở dữ liệu và chuẩn bị và thực thi câu lệnh SQL được lưu trong biến $sql ở phần 2. Trong phần 5, chúng ta fetch kết quả của câu truy vấn, mỗi lần một dòng, chèn mỗi dòng vào output. Phần 6 sẽ đống kết nối đến hệ thống cơ sở dữ liệu, và ta kết thúc ở phần 7 bằng cách chèn tag đóng cho trang kết quả.

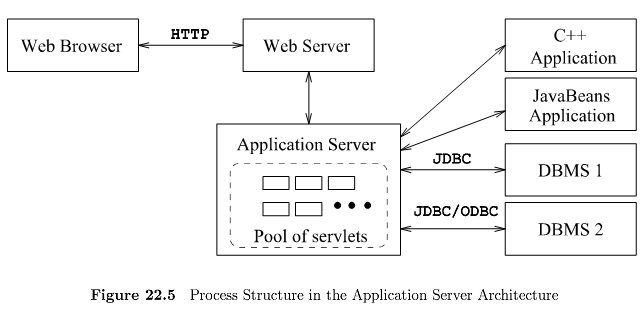
Các protocol thay thể mà trong đó chương trình được gọi lên bằng một yêu cầu, được thực thi nội trong tiến trình Web, và được đề nghị bởi Microsòt (Internet Server API (ISAP)) và Netscape (Netscape ServerAPI (NSAPI)).



## Server ứng dụng và Java server-side

Trong phần trước, chúng ta đã thảo luận cách giao thức CGI có thể được sử dụng để dịch mã tự động một trang web với nội dung được tính toán theo yêu cầu. Tuy nhiên, mỗi yêu cầu trang web đều dẫn tới tạo ra một tiến trình mới, phương án này không tốt đối với số lượng yêu cầu cùng lúc lớn. Vấn đề hiệu suất này đã dẫn đến sự phát triển của một chương trình đặc biệt gọi là server ứng dụng (application server). Một server ứng dụng sẽ tiền phân nhánh các thread hoặc tiến trình và do vậy tránh được chi phí khởi tạo một tiến trình mới cho mỗi yêu cầu. Application server liên quan đến package của tier giữa cung caáp nhiều chức năng khác nhau để loại bỏ chi phí tạo tiến trình:

* Tích hợp các nguồn dữ liệu không đồng nhất.
* Giao tác liên quan đến nhiều nguồn dữ liệu
* Bảo mật
* Quản lý các session

Một kiến trúc Website với application được thể hiện ở hình 22.5. Client giao tiếp với web server thông qua giao thức HTTP. Web server phân phối trang HTML hoặc XML tĩnh trực tiếp đến client. Để dịch mã dynamic page, web server gửi một request đến application server. Application server liên hệ với một hoặc nheièu nguồn dữ liệu để lấy ra các dữ liệu cần thiết hoặc gửi các request cập nhật đến nguồn dữ liệu. Sau khi quá trình giao tiếp với nguồn dữ liệu được hoàn thành, application server sẽ dịch mã trang web và báo cáo kết quả với web server để nó lấy ra page và phân phối đến client. 

Quá trình thực thi của business logic ở Web server site, hay xử lý phía server, đã trở nên mô hình chuẩn cho việc cài đặt các xử lý business phức tạp hơn trên Internet.

# Xa hơn HTML (beyond HTML)

Trong khi HTML thích hợp để biểu diễn cấu trúc của tài liệu phục vụ cho mục đích hiển thị, tính năng của ngôn ngữ không thích hợp cho việc đại diện cho cấu trúc của dữ liệu trong một tài liệu cho các ứng dụng tổng quát hơn là chỉ hiện hiển thị.

Extensible Markup Language (XML) là ngôn ngữ đánh dấu đươc phát triển để bù đắp những thiếu sót của HTML. Ngược lại với HTML có tập các tag có ý nghĩa bắt buộc, XML cho phép người dùng định nghĩa tập hợp các tag có thể được sử dụng cho bất kỳ cấu trúc dữ liệu hoặc tài liệu nào người dùng muốn truyền tải. XML là một cầu nối quan trọng giữa khung nhìn hướng tài liệu của dữ liệu trong HTML và khung nhìn hướng lược đồ của dữ liệu vốn là trung tâm của một DBMS. Nó có tiềm năng làm hệ thống cơ sở dữ liệu dễ dàng tích hợp vào trong ứng dụng web hơn.

XML nổi lên từ sự hợp lưu của hai công nghệ, SGML và HTML. SGML (Standard Generalized Markup Language) là metalanguage cho phép định nghĩa của dữ liệu hoặc tài liệu xen kẽ ngôn ngữ như HTML. Chuẩn SGML được phát hành năm 1988 và nhiều tổ chức quản lý số lượng lớn tài liệu phức tạp đã áp dụng nó. Dù có sự tổng quát, SGML khá phưc tạp và đòi hỏi chương trình tinh vi để khai thác tiềm năng đầy đủ của nó. XML được phát triển để có nhiều sức mạnh của SGML trong khi vẫn giữ được sự đơn giản. Dù sao thì XML vẫn giống SGML cho phép định nghĩa ngôn ngữ đánh dấu tài liệu mới.

Mặc dù XML không ngăn cản người dùng thiết kế code để mã hóa các hiện thỉ dữ liệu trong trình duyệt web, vẫn có một loại ngôn ngữ style cho XML được gọi là XSL (Extensible Style Language). XSL là một chuẩn để miêu tả cách một tài liệu XML - tham gia vào từ vựng của tag - nên được hiển thị

## Khái quát về XML

* *Element:* Element hay còn gọi là tag, là một khối xây dựng cơ bản của tài liệu XML. Điểm bắt đầu nội dung của một element ELM được đánh dấu với <ELM> - gọi là tag bắt đầu (start tag), và kết thúc của nôi dung được đánh dấu với </ELM>, được gọi là end tag. Trong ví dụ element BOOKLIST bao gồ toàn bộ thông tin trong ví dụ. Element BOOK phân ranh giới tất cả dữ liệu với một quyển sách đơn. XML element phân biệt hoa – thường, <BOOK> khác <Book>. Element phải được lồng nhau một cách đúng đắn. Tag bắt đầu xuất hiện trong nội dung của tag khác phải có tag kết thúc tương ứng.

Ví dụ:

*<BOOK>*

*<AUTHOR> <FIRSTNAME>Richard</FIRSTNAME><LASTNAME>Feynman</LASTNAME>*

*</AUTHOR>*

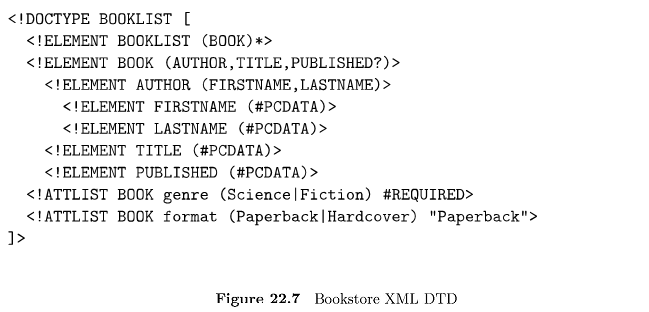
*</BOOK>*

* *Thuộc tính:* Một element có thể có các thuộc tính mô tả cung cấp thông tin mở rộng về element. Giá trị của thuộc tính được set trong tag bắt đầu. Ví dụ <ELM att = “value”> Tất cả giá trị trích dẫn phải nằm trong dấu ngoặc kép.
* *Tham chiếu đối tượng:* đối tượng là những shortcut đối với một phần của text thông thường hoặc nôi dung của các file mở rộng và và chúng ta gọi cách sử dụng của một đối tượng trong tài liệu XML là tham chiếu đối tượng. Ở bất kỳ nơi đâu tham chiếu đối tượng xuất hiện trong tài liệu, nó được thay thế theo đúng nguyên văn bằng nội dung của nó. Tham chiếu đối tượng sẽ bắt đầu với “&” và kết thúc với “;’. Có 5 đối tượng được định nghĩa trong XML là thay thế cho các kí tự đặc biệt trong XML. Ví dụ: “<” - lt, “&” – amp, “>” – gt, “ “ “ – quot, “ ‘ “ – apos. Ví dụ: đoạn text ‘1<5’ đươc viết lại là &apos;1&lt;5&apos.
* *Comment (chú thích)* : chúng ta có thể chèn chú thích vào bất kỳ nơi đâu trong tài liệu XML. Chú thích bắt đầu với <!- và kết thúc -.. Chú thích có thể chứa các đoạn text tùy ý trừ chuỗi “- -“

Khai báo loại tài liệu (DTD), trong XML chúng ta có thể định nghĩa ngôn ngữ đánh dấu cho riêng mình. Một DTD là một tập các rule cho phép ta xác định tập element, thuộc tính và đối tượng cho chính mình. Vì vậy, DTD là ngữ pháp căn bản chỉ định tag nào được cho phép, theo thức tự nào chúng có thể xuất hiện và được lồng nhau ra sao.

## XML DTD

Một DTD là tập các qui tắc cho phép ta xác định tập element, thuộc tính và đối tượng của chính mình. DTD xác định yếu tố nào chúng ta có thể sử dụng và ràng buộc trên các element đó,… các element có thể lồng vào nhau như thế nào và element có thể xuất hiện ở đâu trong tài liệu. Chúng ta sẽ gọi một tài liệu có hiệu lực nếu có một DTD liên quan với nó và tài liệu đươc cấu trúc theo các tậ qui tắc của DTD.



Một DTD được bao quanh bởi <!DOCTYPE name [DTDdeclaration]>, trong đó name là tên của tag bao quanh ngoài cùng, và DTDdeclaration là đoạn text qui tắc của DTD. DTD bắt đầu với element ngoài cùng, còn được gọi là element gốc (root element) – là BOOKLIST trong ví dụ

*<!ELEMENT BOOKLIST (BOOK) \*>*

Qui tắc này cho biết BOOKLIST element chứa 0 hoặc nhiều BOOK element. Dấu “\*” sau từ BOOK ám chỉ có bao nhiêu BOOK element có thể xuất hiện trong BOOKLIST element.

* \*: 0 hoặc nhiều
* +: một hoặc nhiều
* ?: 0 hoặc 1

*<!ELEMENT LASTNAME <#PCDATA)>*

Đến đây chúng ta chỉ mới xem xét các element chứa các element khác. Qui tắc trên cho biết LASTNAME là một element không chứa các element khác, nhưng chứa đoạn text thật. Element chỉ chứa các element khác được gọi là có nội dung element, trong khi #PCDATA được gọi là nội dung hỗn hợp. Tóm lại, một khai báo loại element có cấu trúc sau:

*<!ELEMENT (contentType)>*

Có 5 loại nội dung:

* Chứa các element khác
* #PCDATA: ám chỉ dữ liệu kí tự (charater)
* EMPTY: ám chỉ element không có nội dung, element không có nội dung không bắt buộc có tag đóng
* ANY: cho biết bất kỳ loại nội dung cũng được cho phép. Loại nội dung nên tránh bất kỳ khi nào có thể bỏi vì chúng làm vô hiệu mọi kiểm tra cấu trúc tài liệu bên trong element.
* Một biểu ngữ thông thường (regular expression) xây dựng từ 4 lựa chọn trên. Một regular expression là một trong những nội dung sau:
  + Exp1, exp2, exp3: danh sách các regular expression
  + Exp\*: một expression mở rộng (không hoặc nhiều sự xuất hiện)
  + Exp?: một expression mở rộng (không hoặc nhiều sự xuất hiện)
  + Exp+: một expression bắt buộc (một hoặc nhiều sự xuất hiện)
  + Exp1 | exp2: exp1 or exp2

Thuộc tính được khai báo bên ngoài element. Ví dụ, hãy xem xét khai báo thuộc tính dưới đây:

*<!ATTLIST BOOK genre (Science | Fiction) #REQUIRED>*

Phần XML DTD này xác định thuộc tính genre, là một thuộc tính của element BOOK. Thuộc tính có thể lấy 2 giá trị: Science hoặc Fiction. Mỗi BOOK element phải được mô tả trong tag bắt đầu của nó bằng thuộc tính genre vì thuộc tính này bắt buộc được chỉ định bằng từ khóa #REQUIRED. Dưới đây là cấu trúc tổng quát của một khai báo thuộc tính DTD:

*<!ATTLIST elementName (attName attType default) + >*

Từ khóa ATTLIST xác định sự bắt đầu của một khai báo thuộc tính. Chuỗi elementName là tên của element theo sau là định nghĩa thuộc tính liên quan. Những yếu tố theo sau là định nghĩa của một hoặc nhiều thuộc tính. Mỗi thuộc tính có một tên được xác định bởi attNam và một loại – attType. XML định nghĩa một số loại có thể có cho một thuộc tính. Ở đây, chúng ta chỉ nói về loại string và enumerated. Một thuộc tính kiểu string có thể sử dụng bất kỳ chuỗi nào làm giá trị. Chúng ta có thể khai báo một thuộc tính bằng cách thiết lập loại của nó là CDATA. Ví dụ:

*<!ATTLIST BOOK edition CDATA “1”>*

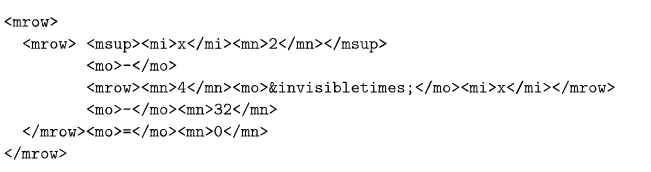
Nếu một thuộc tính thuộc kiểu enumerated, chúng ta liệt kê toàn bộ các giá trị có thể trong khai báo. Trong ví dụ, thuộc tính genre là một kiểu enumerated, giá trị có thể là “Science” hoặc “Fiction”

Phần cuối cùng của khai báo thuộc tính được goi là đặc tả mặc định. XML DTD trong hình 22.7 thể hiện 2 đặc tả mặc định khác nhau: #REQUIRED và chuỗi “Paperback”. Đặc tả mặc định #REQUIRED cho biết thuộc tính đó là bắt buộc và bất cứ khi nào các element liên quan tới nó xuất hiện trong tài liệu XML, một giá trị cho thuộc tính phải được xác định. Ngược lại, đặc tả mặc định được xác định bởi chuỗi “Paperback” ám chỉ thuộc tính đó không bắt buộc.

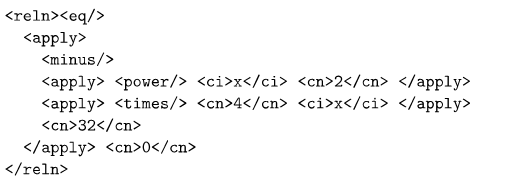
## DTD đặc tả lĩnh vực (Domain – Specific DTD)

Gàn đây, DTD đã được phát triển cho một số lĩnh vực chuyên biệt – bao gồm phạm vi rộng của lĩnh vực thương mại, kĩ thuật, tài chính, công nghiệp, khoa học – và nhiều điều thú vị về XML có bắt nguồn từ sự tin tưởng rằng sẽ có nhiều chuẩn DTD hơn nữa được phát triển.

Ví dụ. MathML (mathematical markup language) được phát triểnn cho việc giải mã các tài iệu toán học trên Web. Ví dụ biểu thức sau: được diễn tả như sau với element hiển thị như sau:



Với element nội dung, biểu thức được biểu diễn:

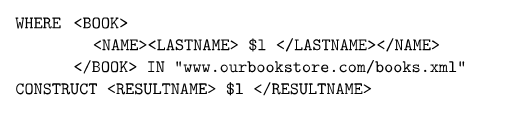


## XML –QL: truy vấn dữ liệu XML

Dữ liệu được mã hóa bằng cách phản ánh cấu trúc của một tài liệu XML, chúng ta có cơ hội để sử dụng ngôn ngữ bậc cao khai thác cấu trúc này một cách thuận lợi để lấy dữ liệu ra. Loại ngôn ngữ này sẽ mang quản lý dữ liệu XML gần với quản lý cơ sở dữ liệu hơn là các mẫu hướng text của tài liệu XML. Loại ngôn ngữ này cho phép ta dễ dàng chuyển đổi dữ liệu XML giữa các DTD khác nhau, điều này là bắt buộc đối với việc tích hợp dữ liệu từ nhiều nguồn.

Trong chương này, chúng ta sẽ tìm hiểu về một ngôn ngữ truy vấn xác định dành cho XML được gọi là XML – QL - có sự giống nhau manh mẽ với các ngôn ngữ truy vấn khác đã được phát triển trong cộng đồng cơ sở dữ liệu.

Hãy xem xét lại tài liệu XML được thể hiện trong hình 22.6. Truy vấn ví dụ dưới đây trả về last name của tất cả tác giả, với giả thiết tài liệu XML này được đặt ở [www.ourbookstore.com/books.xml](http://www.ourbookstore.com/books.xml)

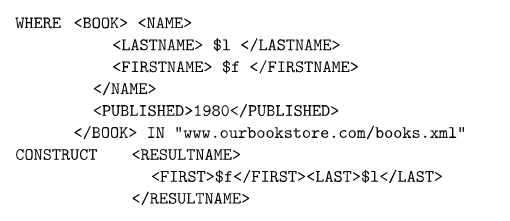


Câu truy vấn này rút trích dữ liệu từ tài liệu XML bằng cách xác định pattern của đánh dấu. Chúng ta quan tâm đến dữ liệu được lồng trong BOOK element, NAME element và LASTNAME element. Đối với mỗi phần của tài liệu XML trùng khớp với cấu trúc xác định bởi câu truy vấn trên, biến l gắn với nội dung của LASTNAME element. Để phân biệt tên biến từ text thông thươnd, ta sử dụng tiền tố $. Nếu như câu truy vấn này được áp dụng cho dữ liệu trong hình 22.6, kết quả trả về sẽ là:

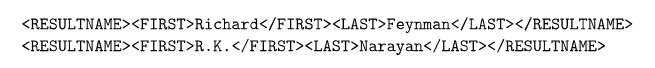
*<RESULTNAME>Feynman</RESULTNAME>*

*<RESULTNAME>Narayan</RESULTNAME>*

Sự chọn lọc được diễn tả bằng cách thay thế đoạn text trong nội dung của một element. Và kết quả của một truy vấn không giới hạn một element đơn. Ví dụ với truy vấn sau:



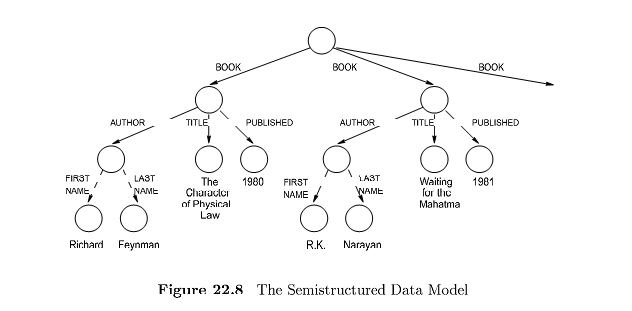
Kết quả của câu truy vấn:



## Mô hình dữ liệu bán cấu trúc

Xem xét một tập các tài liệu trên Web có chứa các hyperlink đến tài liệu khác. Những tài liệu này, mặc dù không hoàn toàn không có cấu trúc, có thể không được mô hình hóa một cách tự nhiên trong mô hình dữ liệu quan hệ vì pattern của hyperlink không thông thường đối với tài liệu. Một tập tin thư mục còn có các bậc của cấu trúc xác định dựa trên các field, như author và title, tuy nhiên lại lại đoạn text không có cấu trúc. Trong khi một số dữ liệu hoàn toàn không có cấu trúc như video stream, audio stream, và hình ảnh thì có nhiều dữ liệu không hoàn toàn có cấu trúc cũng không hoàn toàn không có cấu trúc. Khi đó chúng ta gọi các kiểu dữ liệu như vậy là dữ liệu bán cấu trúc. Tài liệu XML đại diện cho tài nguyên quan trọng và phát triển của dữ liệu bán cấu trúc, và lý thuyết của mô hình dữ liệu bán cấu trúc và truy vấn có tiềm năng phục vụ như sự hình thành của XML.

Có rất nhiều lý do tại sao dữ liệu có thể bán cấu trúc. Đầu tiên, cấu trúc của dữ liệu có thể hiểu ngầm, ẩn, không xác định hoặc người dùng có thể phớt lờ. Thứ hai, xem xét vấn đề tích hợp dữ liệu từ những nguồn dữ liệu bất đồng bộ khi đó việc chuyển đỏi dữ liệu và truyền tải là vấn đề quan trọng. Chúng ta cần một mô hình dữ liệu linh hoạt để tích hợp dữ liệu từ tất cả nguồn dữ liệu bao gồm tập tin phẳng và hệ thống thừa kế, một mô hình dữ liệu cấu trúc thường quá cứng nhắc. Thứ ba, chúng ta không thể truy vấn một cơ sở dữ liệu cấu trúc mà không biết lược đồ, nhưng đôi khi chúng ta muốn truy vấn dữ liệu mà không cần biết toàn bộ lược đồ. Ví dụ, chúng ta không thể diễn ta truy vấn “ Ở đâu trong cơ sở dữ liệu chúng ta có thể tìm thấy chuỗi Malgudi?” trong hệ quản trị cơ sở dữ liệu mà không biết lược đồ.

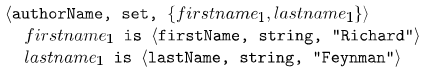


Tất cả mô hình dữ liệu đề xuất cho kiểu dữ liệu bán cấu trúc đại diện cho dữ liệu như một số loại của biểu đồ gắn nhãn. Các nút trong biểu đồ tương ứng với đối tượng phức hợp hoặc giá trị nguyên tố, và cạnh tương ứng với thuộc tính. Không có lược đồ riêng biệt và mô tả bổ sung, dữ liệu trong lược đồ tự mô tả. Ví dụ, xem xét lược đồ hình 22.8, đại diện cho một phần dữ liệu XML trong hình 22.6, nút gốc đại diện cho element ngoài cùng, BOOKLIST. Nút có 3 cạnh được đặt tên element là BOOK, vìvậy danh sách có 3 quyển sách.

Bây giờ chúng ta thảo luận về một trong những mô hình dữ liệu được đề xuất cho dữ liệu bán cấu trúc, được gọi là OEM. Mỗi đối tượng được mô tả bằng bộ 3 một label, một kiểu, và giá trị của đối tượng. Vì mỗi đối tượng có một label có thể được xem là tên một cột trong mô hình quan hệ, và mỗi đối tượng có một kiểu được xem như kiểu dữ liệu của cột trong mô hình quan hệ, mô hình trao đổi đối tượng căn bản là tự mô tả. Label trong mô hình trao đổi đối tượng nên càng nhiều thông tin càng tốt vì nó phục vụ cho hai mục đích: chúng có thể xác định một đối tượng cũng như truyền tải ý nghĩa của đối tượng. Ví dụ, chúng ta có thể biểu diễn last name của một tác giả như sau:

*(lastName, string, “Feynman”)*

Các đối tượng phức tạp hơn có thể được triển khai phân cấp vào đối tượng nhỏ hơn. Ví dụ, một tên tác giả có thể chứa first name và last name.



## Các vấn đề cài đặt dữ liệu bán cấu trúc

Hệ thống cơ sở dữ liệu hỗ trợ dữ liệu bán cấu trúc đã được tập trung nghiên cứu gần đây và cho thấy sự thành công về mặt thương mại của XML, điểm mạnh này sẽ tiếp tục được phát huy. Dữ liệu bán cấu trúc đặt ra nhiều thách thức mới vì hầu hết các chiến lược truy vấn, chỉ mục và lưu trữ truyền thống đều giả sử dữ liệu dính với lược đồ thông thường. Ví dụ, chúng ta nên lưu dữ liệu bán cấu trúc bằng cách ánh xạ nó vào mô hình quan hệ và sau đó lưu trữ dữ liệu đã ánh xạ trong hệ cơ sở dữ liệu quan hệ hay không? Thực hiện chỉ mục trên dữ liệu bán cấu trúc như thế nào?... Tất cả đều đang trong quá trình nghiên cứu

# Chỉ mục cho tìm kiếm chữ

Trong chương này, chúng ta giả sử cơ sở dữ liệu là một tập các tài liệu và chúng ta gọi cơ sở dữ liệu như vậy là cơ sở dữ liệu chữ. Để đơn giản, chúng ta giả sử cơ sở dữ liệu chỉ chứa chính xác một quan hệ và lược đồ quan hệ có chính xác một field tài liệu kiểu. Vì vậy, mỗi record trong quan hệ chứa chính xác một tài liệu. Trong thực hành, lược đồ quan hệ sẽ chứa các field khác như ngày tao tài liệu, một sự phân loại tài liệu, hoặc filed với từ khóa mô tả tài liệu. Cơ sở dữ liệu dạng chữ được sử dụng để lưu trữ các bài báo, giấy tờ hợp pháp và những loại tài liệu khác.

Một lớp quan trọng của truy vấn dựa trên từ khóa tìm kiếm cho phép chúng ta tìm kiếm tất cả tài liệu chứ từ khóa đã cho. Đây là loại truy vấn thông dụng nhất trên web ngày nay, và được hỗ trợ bởi số lượng lớn engine tìm kiếm như AltaVista và Lycos. Một vài hệ thống duy trì danh sách đồng nghĩa của các từ quan trọng và trả về tài liệu chứ từ khóa mong muốn hoặc một trong những từ đồng nghĩa của chúng: ví dụ truy vấn tìm kiếm từ “car” sẽ lấy ra tài liệu chứ “automobile”. Một câu truy vấn phức tạp hơn sẽ chứa các từ khóa AND, OR, NOT, chúng ta có thể xếp hạng các tài liệu được lấy ra bằng xác suất từ khóa truy vấn trong tài liệu.

Có 2 loại truy vấn thông dụng: truy vấn kiểu bool và truy vấn xếp hạng. Trong truy vấn kiểu bool, người dùng cung cấp các biểu thức kiểu bool như dạng sau hay còn gọi là dạng hội thông thường:

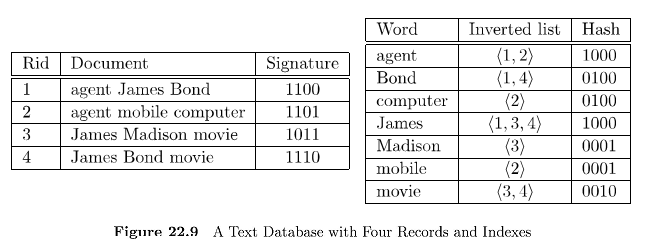


Trong đó là các thuật ngữ truy vấn hay từ khóa độc lập

Truy vấn xếp hạng có cấu trúc tương tự. Trong truy vấn xếp hạng, người dùng còn xác định danh sách các từ, nhưng kết quả của truy vấn là danh sách tài liệu được xếp hạng bằng sự tương đương với danh sách thuật ngữ của người dùng. Làm sao để định nghĩa khi nào và như thế nào một tài liệu tương ứng với một tập các thuật ngữ của người dùng là một vấn đề khó khăn. Thuật ngữ để đánh giá truy vấn thuộc về lĩnh vực truy tìm thông tin, vốn gần gũi với quản lý cơ sở dữ liệu. Hệ thống truy tìm dữ liệu, giống như hệ thống cơ sở dữ liệu, có mục tiêu cho phép người dùng truy vấn một khối lượng dữ liệu lớn, nhưng tập trung vào tập lớn các tài liệu không có cấu trúc. Cập nhật, điều khiển đồng thời, và phục hồi không được chú trọng trong hệ thống truy tìm thông tin vì dữ liệu trong ứng dụng điển hình là tĩnh

## File nghịch đảo

Một file nghịch đảo là một cấu trúc chỉ mục cho phép truy vấn nhanh tất cả tài liệu chứa thuật ngữ truy vấn. Đối với mỗi thuật ngữ, chỉ mục duy trì một danh sách sắp xếp (goi là danh sách nghịch đảo) của id tài liệu chứa thuật ngữ chỉ mục. Ví dụ, xem xét cơ sở dữ liệu dạng chữ trong hình 22.9. Thuật ngữ truy vấn ‘James’ có danh sách đảo của các id record (1,3,4).



Để tìm kiếm nhanh danh sách nghịch đảo cho một thuật ngữ truy vấn, tất cả thuật ngữ truy vấn được tổ chức trong một cấu trúc chỉ mục thứ 2 như cây B+ hoặc chỉ mục hash. Để tránh bất kỳ sự hỗn loạn nào, chúng ta gọi chỉ mục thứ hai cho phép truy vấn nhanh danh sách nghịch đảo của một thuật ngữ truy vấn là chỉ mục từ vựng. Chỉ mục từ vựng chứa mỗi thuật ngữ truy vấn có thể có và một con trỏ đến danh sách đảo

Một truy vấn chứa thuật ngữ đơn được tính toán bằng cách đầu tiên cắt ngang chỉ mục từ vựng đến entry nút lá với địa chỉ của danh sách nghịch đảo cho thuật ngữ. Sau đó danh sách đảo được lấy ra, rid được ánh xạ vào địa chỉ vật lý của tài liệu, và tài liệu tương ứng được gọi lên. Một truy vấn với hội của vài thuật ngữ được tính toán bằng cách lấy ra danh sách nghịch đảo của các thuật ngữ truy vấn cùng một lúc và giao lại với nhau. Để giảm tối đa sử dụng bộ nhớ, danh sách nghịch đảo nên được lấy ra để làm tăng độ dài. Một truy vấn với sự tách rời các thuật ngữ được tính toán bằng cách trộn các danh sách nghịch đảo với nhau.

## File ký số

File ký số là một cấu trúc chỉ mục khác cho hệ thống cơ sở dữ liệu chữ hỗ trợ tính toán hiệu quả truy vấn kiểu bool. File ký số chứa một record chỉ mục cho mỗi tài liệu trong cơ sở dữ liệu. Record chỉ mục sẽ được gọi là ký số của tài liệu. Mỗi ký số có kích thước b bit cố định, b được gọi là độ rộng ký số. Làm sao chúng ta quyết định bit nào để thiết lập cho một tài liệu. Bit được thiết lập tùy thuộc vào từ hiển thị trong tài liệu. Chúng ta ánh xạ từ vào bit bằng cách áp dụng hàm hash cho mỗi từ trong tài liệu và chúng ta thiết lập bit xuất hiện trong kết quả của hàm hash. Chú ý là trừ khi chúng ta có một bit cho mỗi từ trong từ vựng, bit giống nhau có thể được thiết lập 2 lần bằng 2 từ khác nhau bởi vì hàm hash ánh xả cả 2 từ vào cùng bit. Chúng ta nói rằng ký số S1 khớp với một ký số khác S2 nếu tất cả các bit được thiết lập trong ký số S2 cũng được thiết lập trong ký số S1. Nếu ký số S1 khớp với S2, thì ký số S1 có ít nhất nhiều bit được thiết lập như ký số S2.

Đối với truy vấn chứa tập hợp của những thuật ngữ, đầu tiên chúng ta phát sinh ra ký số truy vấn bằng cách áp dụng hàm hash để đọc mỗi từ trong truy vấn. Sau đó quét các file ký số và lấy ra toàn bộ tài liệu có ký số khớp với ký số truy vấn, vì mỗi tài liệu như vậy là một kết quả tìm năng cho câu truy vấn. Vì ký số không xác định độc nhất những từ mà tài liệu chứa, chúng ta phải lấy ra mỗi lần trùng khớp và kiểm tra tài liệu đó thật sự chứa thuật ngữ truy vấn hay không. Một tài liệu có ký số phù hợp nhưng không chứa những thuật ngữ đó được gọi là một xác thực sai. Một xác thực sai là một lỗi khá đắt vì tài liệu phải được lấy ra từ đĩa, phân tích, phát sinh và kiểm tra để quyết định nó có chứa thuật ngữ truy vấn hay không.

Đối với truy vấn chứa các thuật ngữ rời rạc, ta phát sinh danh sách các ký số truy vấn, mỗi ký số một thuật ngữ. Truy vấn được tính toán bằng cách quét file ký số để tìm kiếm tài liệu có ký số trùng khớp với bất kỳ ký số nào trong danh sách ký số truy vấn.

Chú ý là mỗi truy vấn chúng ta phải quét toàn bộ file ký số, và càng nhiều record trong file ký số thì càng có nhiều tài liệu trong cơ sở dữ liệu. Để giảm lượng dữ liệu phải truy vấn cho mỗi câu truy vấn, chúng ta có thể phân vùng dọc file ký số thành các bit slice, chúng ta gọi một chỉ mụ một file ký số được cắt bit. Độ dài của mỗi bit slice vẫn bằng với số tài liệu trong cơ sở dữ liệu, nhưng với mỗi truy vấn có q bit được thiết lập trong ký số truy vấn, chúng ta chỉ cần lấy ra q bit slice

Ví dụ trong hình 22.9 với file ký số độ rộng là 4. Số bit thiết lập bởi giá trị hash của các thuật ngữ truy vấn được thể hiện trong hình. Để tính toán cho câu truy vấn ‘James’, đầu tiên ta sẽ tính toán giá trị hash của thuật ngữ là 1000. Sau đó quét file ký số và tìm record chỉ mục trùng khớp. Như trong hình 22.9, ký số của tất cả record đều có bit đầu tiên được thiết lâp, chúng ta lấy ra tất cả tài liệu và kiểm tra xác thực sai, ở đây chỉ có một xác thực sai là rid 2.

# Xếp hạng các từ khóa tìm kiếm trên web

World Wide Web chứa một lượng thông tin dị thường. Tìm kiếm một trang web tương ứng với truy vấn của người dùng còn khó hơn cả mò kim đáy bể. Sự đa dạng của các trang web trong cấu trúc, nội dung, quyền sở hữu, chất lượng, … làm chúng trở nên khó khắn nếu không không thể áp dụng các kĩ thuật truy vấn tiêu chuẩn.

Trong quá trình tìm kiếm các trang web tương ứng, chúng ta phân biệt giữa 2 loai trang: authority và hub. Một authority là một trang tương ứng với một chủ đề xác định và được nhận diện bằng các trang khác như có căn cứ trên vấn đề. Những trang được gọi là hub có số lượng lớn hyperlink đến authority, mặc dù bản thân chúng ít được biết đến và ít mang nôi dung tương đương với truy vấn. Trang hub có thể là sự biên dịch các tài nguyên về một chủ đề trên một site cho các chuyên gia, danh sách các site khuyến khích cho sở thích của một cá nhân người dùng.

## Thuật toán xếp hạng trang Web

Trong phần này chúng ta sẽ nói về HITS, một thuật toán tìm kiếm authority và hub và trả về kết quả cho truy vấn người dùng. Chúng ta sẽ xem World Wide Web như một bản đồ trực tiếp. Mỗi Web page đại diện cho một nút trong lược đò, và hyperlink từ trang A đến trang B đại diện là một cạnh giữa 2 nút tương ứng. Giả sử chúng ta được cho một truy vấn người dngf với một vài thuật ngữ. Thuật toán thực hiện trong 2 bước:

* ***Bước 1:*** bước lấy mẫu (sampling step), chúng ta thu thập tập các trang web được gọi là tập cơ bản. Tập cơ bản chứa các trang gần tương đương với truy vấn của người dùng, nhưng nó còn khá lớn.
* ***Bước 2:*** bước lặp (iteration step) chúng ta tìm các authority và hub tốt giữa các trang trong tập cơ bản.

Bước lấy mẫu sẽ lấy tập các trang Web chứa thuật ngữ truy vấn, sử dụng một vài kĩ thuật truyền thống. Ví dụ, chúng ta có thể tính toán truy vấn như một từ khóa bool , tìm kiếm và lấy ra các trang Web chứa thuật ngữ truy vấn. Chúng ta gọi tập kết quả trang là tập gốc. Tập gốc có thể không chứa các trang tương đương vì một vài authority page hoặc một vài authority page dãn đến trang trong tập gốc. Điều này thúc đẩy một khái niệm là link page (trang liên kết). Chúng ta gọi là một link pgae nếu nó có một hyperlink đến trang trong tập gốc hoặc một trang trong tập gốc có hyperlink đến nó. Để không bỏ sót các trang tương ứng, chúng ta bổ sung tập gốc bằng tất cả các link page và gọi đó là tập cơ bản. Vì vậy, tập cơ bản chứa tất cả trang gốc và trang liên kết, chúng ta sẽ gọi trang web trong tập cơ bản là trang cơ bản.

Mục đích của bước 2 là tìm ra trang cơ bản nào là hub và authority tốt và trả về kết quả co người dùng. Để xác định số lượng chất lượng của một trang cơ bản như hub và authority, chúng ta liên kết mỗi trang cơ bản trong tập cơ bản với một hub weight và authority weight. Hub weigtht của một trang ám chỉ chất lượng của một trang theo hub, và tương tự với authority weight. Chúng ta tính toán weight của mỗi trang theo trực giác rằng nếu một trang là authority tốt nếu có nhiều hub tốt hyperlink tới nó, và một hub tốt nếu có nhiều hyperlink ra ngoài đến các authority tốt. Vì chúng ta không có bất kỳ kiến thức nào về trang nào là hub và authority tốt, chúng ta khỏi tạo tất cả weight là 1. Sau đó cập nhật weight lặp đi lặp lại như dưới đây

Hãy xem xét một trang cơ bản p với hub weight là và authority weight . Trong một vòng lặp, chúng ta cập nhật là tổng của tất cả hub weight của tất cả trang có hyperlink đến p.

Một cách tương tự, chúng ta cập nhật là tổng weight của các trang trỏ mà p trỏ tới

So sánh thuật toán với các phương pháp truy vấn chữ đã thảo luận trong chương này, chúng ta lưu ý bước lặp của thuật toán HITS – sự phân phối các weight – không xét đến các chữ trong trang cơ bản. Trong bước lặp, chúng ta chỉ quan tâ, đến quan hệ giữa các trang cơ bản được đại diện bằng hyperlink.

Thuật toán HITS thường tạo ra kết quả tốt. Ví dụ, 5 authority được xếp hạng cao nhất cho truy vấn ‘search engine’ là

<http://www.yahoo.com/>

<http://www.excite.com/>

<http://www.mckinley.com/>

<http://www.lycos.com/>

<http://www.altavista.digital.com/>