1. **DTD trong XML**
2. Định nghĩa:

DTD – Document Type Definition, để xác định **cấu trúc** và sự hợp lý của các phần tử và thuộc tính trong tài liệu XML.

Chú ý: Tài liệu XML đúng cú pháp gọi là “**Well-Formed**”. Đúng cấu trúc gọi là “**validate**”.

1. Mục đích:

DTD trong XML sử dụng để **validate** tài liệu XML.

VD: Quy định tài liệu xml, phần tử root là danhsachsinhvien, và chỉ chứa 4 phần tử: id, name, age, address; Kiểu dữ liệu của 4 phần tử id, name, age, address là chuỗi và không chứa các thẻ con bên trong.

XML chỉ được phép thế này:

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <danhsachsinhvien>  <id>200</id>  <name>DungPT</name>  <age>25</age>  <address>Thanh Hóa</address>  </danhsachsinhvien> |

* Tài liệu XML kết hợp cùng DTD để giải quyết vấn đề trên.

**Sử dụng DTD:**

**+ Khai báo Element ( các thẻ được dùng trong tài liệu XML)**

**+ Khai báo các Attribute – Thuộc tính của Element/ thẻ.**

**+ Khai báo các ENTITY ( biến tham chiếu giá trị ) khai báo tên của Entity, giá trị của Entity hay vị trí của giá trị Entity**

* Các bước tạo DTD:

- Khai báo tất cả các element có trong XML

- Khai báo các element con cho từng element nếu có

- Xác định thứ tự xuất hiện của các element

- Khai báo tất cả thuộc tính của từng element nếu có

- Khai báo kiểu dữ liệu và giá trị mặc định cho thuộc tính

- Khai báo các Entity nếu có

1. Cú pháp DTD

* Tạo thẻ/ phần tử (Element) chỉ được phép chứa giá trị không chứa thẻ/ phần tử khác.

|  |
| --- |
| <!ELEMENT element-name (danh\_sách\_các\_thẻ\_con)|(#PCDATA)|ANY|EMPTY> |

Trong đó: element-name là tên thẻ cần validate

Kiểu của các thẻ:

+ (danh\_sách\_các\_thẻ\_con): danh sách các thẻ con mà thẻ chứa.

+ (#PCDATA): parsed character: tuân thủ theo quy tắc well-former của XML

+ ANY: Data của thẻ chứa loại giá trị bất kỳ

+ EMPTY: Thẻ là thẻ trống

VD: đối với thẻ id, name, age, address trên thể hiện:

<!ELEMENT id (#PCDATA)>

<!ELEMENT name (#PCDATA)>

<!ELEMENT age (#PCDATA)>

<!ELEMENT address (#PCDATA)>

* Tạo thẻ/ phần tử (Element) chứa phần tử khác

|  |
| --- |
| <!ELEMENT element-name (danh\_sách\_các\_phần\_tử\_con)> |

Trong đó: element-name là thẻ/ phần tử chứa các phần tử con.

danh\_sách\_các\_phần\_tử\_con Là danh sách các phần tử con của thẻ element-name, danh sách này là chuỗi ngăn cách nhau bởi dấu phẩy.

VD: thẻ danhsachsinhvien chứa 4 thẻ id, name, age, address

<!ELEMENT danhsachsinhvien (id,name,age,address)>

* Ngoài ra có thể cấu hình để ràng buộc số lần có mặt của các thẻ con:

+: phải xuất hiện tối thiểu 1 lần

\*: xuất hiện 0 hay nhiều lần

?: xuất hiện 0 hay 1 lần

Child1|child2: hoặc child1 xuất hiện hoặc child2 xuất hiện

* Chỉ định một phần tử phải xuất hiện 1 lần

|  |
| --- |
| <!ELEMENT element-name (child-name)> |

Trong đó: element-name là thẻ/ phần tử chứa các thẻ con

child-name Là phần tử/ thẻ con bắt buộc phải xuất hiện 1 lần trong thẻ cha element-name

VD: Thẻ gender phải xuất hiện 1 lần trong thẻ danhsachsinhvien

<!ELEMENT danhsachsinhvien (gender)>

* Chỉ định một phần tử xuất hiện ít nhất 1 lần

|  |
| --- |
| <!ELEMENT element-name (child-name**+**)> |

Trong đó: element-name là thẻ/ phần tử chứa các thẻ con

child-name Là phần tử/ thẻ con bắt buộc phải xuất hiện ít nhất 1 lần trong thẻ cha element-name

VD: Thẻ note phải xuất hiện ít nhất 1 lần trong thẻ danhsachsinhvien

<!ELEMENT danhsachsinhvien (note**+**)>

* Chỉ định một phần tử không xuất hiện hoặc xuất hiện nhiều lần

|  |
| --- |
| <!ELEMENT element-name (child-name**\***)> |

Trong đó: element-name là thẻ/ phần tử chứa các thẻ con

child-name Là phần tử/ thẻ con có thể không xuất hiện hoặc xuất hiện nhiều lần trong thẻ cha element-name

VD: Thẻ note phải xuất hiện không xuất hiện hoặc xuất hiện nhiều lần trong thẻ danhsachsinhvien

<!ELEMENT danhsachsinhvien (note**\***)>

* Chỉ định 1 phần tử không xuất hiện hoặc xuất hiện 1 lần

|  |
| --- |
| <!ELEMENT element-name (child-name**?**)> |

Trong đó: element-name là thẻ/ phần tử chứa các thẻ con

child-name Là phần tử/ thẻ con có thể không xuất hiện hoặc xuất hiện 1 lần trong thẻ cha element-name

VD: Thẻ note không xuất hiện hoặc xuất hiện 1 lần trong thẻ danhsachsinhvien

<!ELEMENT danhsachsinhvien (note**?**)>

* Chỉ định một trong danh sách phần tử xuất hiện

|  |
| --- |
| <!ELEMENT element-name (#PCDATA|child-name1|child-name2|child-name3)> |

Trong đó: element-name là thẻ/ phần tử chứa các thẻ con

child-name1, child-name2, child-name3 Là 1 trong 3phần tử/ thẻ con được xuất hiện trong thẻ cha element-name

VD: Thẻ note và thẻ identify là 1 trong 2 thẻ được xuất hiện trong thẻ danhsachsinhvien

<!ELEMENT danhsachsinhvien (id,name,age,address,(note|identify))>

* Khai báo thuộc tính cho phần tử

|  |
| --- |
| <!ATTLIST element-name **attribute-name** attribute-type attribute-value> |

Trong đó: element-name là thẻ/ phần tử có thuộc tính

**attribute-name là tên thuộc tính của thẻ**

attribute-type là kiểu thuộc tính, có thể là:

|  |  |
| --- | --- |
| **attribute-type** | **Mô tả** |
| CDATA | Giá trị là các ký tự |
| (*en1*|*en2*|..) | Giá trị phải là một từ một danh sách liệt kê |
| ID | Giá trị là id duy nhất |
| IDREF | Giá trị là id của thành phần khác |
| IDREFS | Giá trị là 1 list các ids khác |
| NMTOKEN | Giá trị là tên XML |
| NMTOKENS | Giá trị là list tên XML |
| ENTITY | Giá trị là 1 entity |
| ENTITIES | Giá trị là list các entities |
| NOTATION | Giá trị là tên notation |
| xml: | Giá trị là predefined xml value |

attribute-value là khai báo giá trị tương ứng của thuộc tính **attribute-name,** attribute-valuecó thể là:

|  |  |
| --- | --- |
| **attribute-value** | **Mô tả** |
| value | Giá trị mặc định của thuộc tính ( giá trị thực tế người dùng tạo ) |
| #REQUIRED | Giá trị của thuộc tính là bắt buộc |
| #IMPLIED | Giá trị của thuộc tính không bắt buộc |
| #FIXED value | Giá trị thuộc tính là cố định |

VD: Thẻ cha danhsachsinhvien chứa thẻ school có thuộc tính name = “FPT”:

<!ATTLIST school name CDATA "FPT">

Tương đương trong xml:

<school name="FPT"/>

* Khai báo tham chiếu ENTITY:

|  |
| --- |
| <!ENTITY entity-name **“**entity-value”> |

Trong đó: entity-name là tên/ biến tham chiếu

entity-value là giá trị tham chiếu

Trong nội dung XML, muốn gọi tới biến tham chiếu thì sử dụng &entity-name;.

1. Sử dụng trong XML document

* C1: Khai báo DTD ngay bên trong file XML:

|  |
| --- |
| <!DOCTYPE element DTD identifier  [  declaration1  declaration2  ........  ]> |

Trong đó:

* DTD bắt đầu với khai báo là <!DOCTYPE.
* Một element (phần tử/ thẻ) nói cho Parser để phân tích cú pháp tài liệu từ phần tử gốc đã cho.
* DTD identifier là một định danh cho Document Type Definition, mà có thể là path tới một file trên hệ thống hoặc URL tới một file trên mạng. Nếu DTD đang trỏ tới path bên ngoài, thì nó 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.
* File XML phải khai báo thêm thuộc tính standalone="yes"

VD:

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8" standalone="yes" ?>  <!DOCTYPE address  [  <!ELEMENT address (name,company,phone)>  <!ELEMENT name (#PCDATA)>  <!ELEMENT company (#PCDATA)>  <!ELEMENT phone (#PCDATA)>  ]>  <address>  <name>DungPT</name>  <company>FPT</company>  <phone>09739647851</phone>  </address> |

* C2: Tạo file DTD riêng và khai báo trong file XML
* Thuộc tính standalone="no"
* Nghĩa là khai báo file DTD từ nguồn ngoài
* Cú pháp:

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

Trong đó:

root-element Là thành phần root của tài liệu XML

file-name Là đường dẫn tên file tới file .dtd

* Khi sử dụng file DTD ngoại vi bên ngoài có thể sử dụng định danh của hệ thống (System Identifier) hoặc định danh chung (Public Identifier)
* Định danh hệ thống (System Identifier): xác định vị trí của một file ngoại vi chứa các khai báo DTD
* Định danh chung (Public Identifier): xác định vị trí DTD resource

VD:

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <!ELEMENT sinhvien (id+,name,age,address,note\*,school,(identify|message))>  <!ELEMENT id (#PCDATA)>  <!ELEMENT name (#PCDATA)>  <!ELEMENT age (#PCDATA)>  <!ELEMENT address (#PCDATA)>  <!ELEMENT school (id, name)>  <!ELEMENT identify (#PCDATA)>  <!ELEMENT message (#PCDATA)>  <!ATTLIST school name CDATA #REQUIRED> |
| <?xml version="1.0" encoding="UTF-8"?>  <!DOCTYPE sinhvien SYSTEM "studentDTD.dtd">  <sinhvien>  <id>25</id>  <name>Dung</name>  <age>25</age>  <address>TH</address>  <school name="456">  <id>11</id>  <name>FPT</name>  </school>  <message>aa</message>  </sinhvien> |

1. Schema trong XML
2. Định nghĩa

XML Schema hoặc XML Schema Definition (XSD) sử dụng để **miêu tả và xác nhận (validate) cấu trúc và nội dung** của dữ liệu XML.

1. Mục đích:

Quy định trong XML: phần tử root, quy định trình tự xuất hiện, quy định số lần xuất hiện và quy định kiểu dữ liệu, thuộc tính và dữ liệu. Phần tử Schema hỗ trợ Namespace. Nó tương tự như một Schema trong cơ sở dữ liệu mà miêu tả dữ liệu trong một cơ sở dữ liệu.

**DTD thì định nghĩa về cấu trúc của file XML, còn Schema ràng buộc chi tiết quy định về kiểu dữ liệu, số ký tự trong 1 chuỗi, giới hạn phạm vi giá trị ....**

1. Kiểu định nghĩa

* Kiểu đơn/ Kiểu đơn giản: Chỉ sử dụng cho dữ liệu text (xsd:string, xsd:decimal, xsd:integer, xsd:boolean, xsd:date, xsd:time) ( mục d)
* Kiểu phức tạp: định nghĩa cho phần tử chứa các phần tử khác hoặc có kiểu phức tạp (xs:complexType, xs:sequence ...)
* Kiểu tổng thể (global): định nghĩa một kiểu đơn trong tài liệu, mà có thể được sử dụng bởi tất cả tham chiếu khác. Khai báo kiểu đơn, và khi dùng gọi tới kiểu tham chiếu đơn trên.

1. Cú pháp Kiểu đơn:

* Định nghĩa một phần tử đơn

|  |
| --- |
| **<xsd:element name="x" type="y"/>** |

Trong đó: x là tên phần tử; y là kiểu dữ liệu.

|  |  |
| --- | --- |
| **Kiểu dữ liệu** | **Mô tả** |
| xsd:string | Ký tự |
| xsd:decimal | Số thực |
| xsd:integer | Số nguyên |
| xsd:boolean | Luận lý (true hoặc false) |
| xsd:date | Ngày |
| xsd:time | Thời gian |

VD: Khai báo thẻ id, name trong file xml là integer và string:

|  |
| --- |
| <xsd:element name="name" type="xsd:string"/>  <xsd:element name="id" type="xsd:integer"/> |

* Quy định giá trị mặc định và giá trị cố định cho phần tử đơn
* Phần tử đơn là phần tử chỉ chứa giá trị. 1 phần tử đơn có thể có một giá trị mặc định hoặc một giá trị cố định.
* Phần tử đơn có thể có một giá trị mặc định

|  |
| --- |
| **<xsd:element name="x" type="y" default="z"/>** |

Trong đó z là giá trị mặc định của thẻ x

* Phần tử đơn có giá trị cố định

|  |
| --- |
| **<xsd:element name="x" type="y" fixed="z"/>** |

Trong đó z là giá trị cố định của thẻ x

* Định nghĩa thuộc tính cho một phần tử

|  |
| --- |
| **<xsd:attribute name="xxx" type="yyy"/>** |

Trong đó xxx là tên thuộc tính, yyy là kiểu thuộc tính, có thể thêm giá trị mặc định (**default="z"**) hoặc giá trị cố định cho thuộc tính (**fixed="z"**).

* Ràng buộc với 1 tập các giá trị

|  |
| --- |
| **<xsd:element name="xxx">**  **<xsd:simpleType>**  **<xsd:restriction base="yyy">**  **<xsd:minInclusive value="aaa"/>**  **<xsd:maxInclusive value="bbb"/>**  **</xsd:restriction>**  **</xsd:simpleType>**  **</xsd:element>** |

Trong đó: xxx là tên thẻ/ thành phần, yyy là kiểu dữ liệu, aaa là giá trị min, bbb là giá trị max

* Ràng buộc với 1 tập các giá trị

|  |
| --- |
| **<xsd:element name="xxx">**  **<xsd:simpleType>**  **<xsd:restriction base="yyy">**  **<xsd:enumeration value="aaa"/>**  **<xsd:enumeration value="bbb"/>**  **<xsd:enumeration value="ccc"/>**  **</xsd:restriction>**  **</xsd:simpleType>**  **</xsd:element>** |

Trong đó: xxx là tên thẻ/ thành phần, yyy là kiểu dữ liệu, aaa, bbb, ccc là giá trị ràng buộc của thẻ đó.

* Ràng buộc trên 1 loạt các giá trị

|  |
| --- |
| **<xsd:element name="xxx">**  **<xsd:simpleType>**  **<xsd:restriction base="yyy">**  **<xsd:pattern value="ccc"/>**  **</xsd:restriction>**  **</xsd:simpleType>**  **</xsd:element>** |

Trong đó xxx là tên thẻ/ thành phần, yyy là kiểu dữ liệu, ccc là giá trị ràng buộc như bảng:

|  |  |
| --- | --- |
| [] | Ràng buộc dữ liệu trong khoảng. VD: [a-z]: ràng buộc dữ liệu là 1 ký tự thuộc a - z |
| | | Ràng buộc chấp nhận giá trị VD: male|female : ràng buộc dữ liệu là male hoặc female |
| \* | Ràng buộc có giá trị hoặc không  VD: ([a-z])\* |
| + | Ràng buộc bắt buộc phải có 1 giá trị hoặc hơn  VD: ([a-z])+ |
| {} | Ràng buộc số ký tự VD: [a-zA-Z0-9#$%^^&&]{8} : ràng buộc đủ 8 ký tự, chữ thường hoặc chữ in hoa từ a đến z, hoặc chữ số từ 0 đến 9 |

1. Kiểu phức tạp: định nghĩa cho phần tử chứa các phần tử khác

|  |
| --- |
| **<xs:element name="xxx">**  **<xs:complexType>**  **<xs:sequence>**  **yyy**  **</xs:sequence>**  **</xs:complexType>**  **</xs:element>** |

Trong đó xxx là tên thẻ/ thành phần chứa các thẻ con yyy

VD: Thẻ root là sinhvien chứa thẻ id ( integer), name (string), gender (string)

|  |
| --- |
| <?xml version="1.0"?>  <xs:schema version="1.0"  xmlns:xs="http://www.w3.org/2001/XMLSchema"  elementFormDefault="qualified">  <xs:element name="student">  <xs:complexType>  <xs:sequence>  <xs:element name="name">  <xs:simpleType>  <xs:restriction base="xs:string">  <xs:pattern value="[a-zA-Z]{3}"></xs:pattern>  </xs:restriction>  </xs:simpleType>  </xs:element>  <xs:element name="id" type="xs:integer"/>  <xs:element name="gender">  <xs:simpleType>  <xs:restriction base="xs:string">  <xs:pattern value="male|female"/>  </xs:restriction>  </xs:simpleType>  </xs:element>  </xs:sequence>  </xs:complexType>  </xs:element>  </xs:schema> |

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <student  xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'  xsi:noNamespaceSchemaLocation='studentSchema.xsd'>  <name>abc</name>  <id>111</id>  <gender>male</gender>  </student> |

1. Kiểu tổng thể (global)

* B1: định nghĩa 1 kiểu đơn chung
* B2: sử dụng kiểu định nghĩa trên như 1 kiểu khai báo dữ liệu.

VD: thẻ address gồm kiểu phức tạp chứa province, district, commune

|  |
| --- |
| <?xml version="1.0"?>  <xs:schema version="1.0"  xmlns:xs="http://www.w3.org/2001/XMLSchema"  elementFormDefault="qualified">  <xs:element name="student">  <xs:complexType>  <xs:sequence>  <xs:element name="name">  <xs:simpleType>  <xs:restriction base="xs:string">  <xs:pattern value="[a-zA-Z]{3}"></xs:pattern>  </xs:restriction>  </xs:simpleType>  </xs:element>  <xs:element name="id" type="xs:integer"/>  <xs:element name="gender">  <xs:simpleType>  <xs:restriction base="xs:string">  <xs:pattern value="male|female"/>  </xs:restriction>  </xs:simpleType>  </xs:element>  <xs:element name="address1">  <xs:complexType>  <xs:sequence>  <xs:element name="address" type="addressType" />  <xs:element name="phone" type="xs:string"/>  </xs:sequence>  </xs:complexType>  </xs:element>  </xs:sequence>  </xs:complexType>  </xs:element>  <xs:complexType name="addressType">  <xs:sequence>  <xs:element name="province" type="xs:string"/>  <xs:element name="district" type="xs:string"/>  <xs:element name="commune" type="xs:string"/>  </xs:sequence>  </xs:complexType>  </xs:schema> |

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <student  xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'  xsi:noNamespaceSchemaLocation='studentSchema.xsd'>  <name>abc</name>  <id>111</id>  <gender>male</gender>  <address1>  <address>  <province>TH</province>  <district>HT</district>  <commune>HC</commune>  </address>  <phone>09458</phone>  </address1>  </student> |

1. Tham chiếu trong XML

VD: <student

xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'

xsi:noNamespaceSchemaLocation='studentSchema.xsd'>

Trong đó student là thẻ root, và file 'studentSchema.xsd' định nghĩa cú pháp dữ liệu file XML.

1. Xslt trong XML
2. Định nghĩa

XSL(eXtensible Stylesheet Language) là một dạng ngôn ngữ dựa trên nền tảng XML để tạo ra stylesheet nhằm mục đích định dạng nội dung XML cho nhiều mục đích hiển thị khác nhau, đồng thời XSL còn có khả năng chuyển đổi XML thành dạng định dạng khác. XSL bao gồm 3 phần:

* XSL Transformation (XSLT): là một ngôn ngữ dạng XML bao gồm các thẻ xử lý cho phép chuyển đổi XML thành định dạng khác như HTML, XML khác
* XML Path Language (X-Path): là một ngôn ngữ dùng để truy xuất các thành phần trong **XML**
* XSL Formatting Object (XSL-FO): là ngôn ngữ dùng để định dạng XML
* XML Path Language (XPath) là một ngôn ngữ thiết kế ra với mục đích giúp cho ứng dụng có thể di chuyển bên trong XML document và truy xuất các giá trị cũng như thuộc tính của các elements.

1. Chuyển đổi XML bằng XSLT

Tài liệu XML 🡪 Chương trình XSLT 🡪 Tài liệu văn bản

Như:

* XML 🡪 HTML

VD:

|  |  |  |
| --- | --- | --- |
| <?xml version="1.0" encoding="UTF-8"?>  <tong>  <songuyen1>10</songuyen1>  <songuyen2>20</songuyen2>  </tong> | 🡪 | Phép cộng 2 số: 10 + 20 |

* XML 🡪 XML

|  |  |  |
| --- | --- | --- |
| <?xml version="1.0" encoding="UTF-8"?>  <tong>  <songuyen1>10</songuyen1>  <songuyen2>20</songuyen2>  </tong> | 🡪 | <?xml version="1.0" encoding="UTF-8"?>  <tong songuyen1="10" songuyen2="20" /> |

1. Cú pháp cơ bản của XSLT

* Thành phần <xsl:template>
* Dùng để so khớp với các phần tử XML
* Tìm và so khớp với phần tử gốc (root)

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

* Tìm và so khớp với các phần tử con

**<xsl:template match="Tên phần tử">**

* Thành phần <xsl:value-of>
* Dùng để trích xuất thông tin của một phần tử XML

|  |
| --- |
| **<xsl:value-of select="Tên phần tử"/>** |

VD:

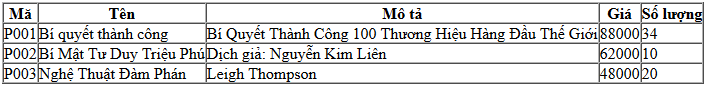
|  |  |
| --- | --- |
| <?xml version="1.0" encoding="UTF-8"?>  <?xml-stylesheet type="text/xsl" href="newss.xsl"?>  <tong>  **<songuyen1>10</songuyen1>**  <songuyen2>20</songuyen2>  </tong> | <?xml version="1.0" encoding="UTF-8"?>  <xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">  <xsl:output method="html"/>  <xsl:template match="/">  <html>  <head>  <title>newss.xsl</title>  </head>  <body>  **<xsl:value-of select="/tong/songuyen1"/>**  <xsl:text>+</xsl:text>  <xsl:value-of select="/tong/songuyen2"/>  </body>  </html>  </xsl:template>  </xsl:stylesheet> |

* 10+20
* Thành phần <xsl:for-each>
* Vòng lặp

|  |
| --- |
| **<xsl:for-each select="Tên phần tử">** |

VD:

|  |  |
| --- | --- |
| <?xml version="1.0" encoding="UTF-8"?>  <?xml-stylesheet type="text/xsl" href="new.xsl"?>  <PRODUCTDATA>  <Product ProdID="P001">  <ProductName>Bí quyết thành công</ProductName>  <ProductDesc>Bí Quyết Thành Công 100 Thương Hiệu Hàng Đầu Thế Giới</ProductDesc>  <Price>88000</Price>  <Quantity>34</Quantity>  </Product>  <Product ProdID="P002">  <ProductName>Bí Mật Tư Duy Triệu Phú </ProductName>  <ProductDesc>Dịch giả: Nguyễn Kim Liên</ProductDesc>  <Price>62000</Price>  <Quantity>10</Quantity>  </Product>  <Product ProdID="P003">  <ProductName>Nghệ Thuật Đàm Phán</ProductName>  <ProductDesc>Leigh Thompson</ProductDesc>  <Price>48000</Price>  <Quantity>20</Quantity>  </Product>  </PRODUCTDATA> | <xsl:for-each select="PRODUCTDATA/Product">  <tr>  <td>  <xsl:value-of select="@ProdID"/>  </td>  <td>  <xsl:value-of select="ProductName"/>  </td>  <td>  <xsl:value-of select="ProductDesc"/>  </td>  <td>  <xsl:value-of select="Price"/>  </td>  <td>  <xsl:value-of select="Quantity"/>  </td>  </tr>  </xsl:for-each> |



* Thành phần <xsl:sort>
* Sắp xếp theo phần tử được chỉ định

|  |
| --- |
| **<xsl:sort select="Tên phần tử" order="ascending | descending"/>** |

* Thành phần <xsl:if>
* Kiểm tra dữ liệu thỏa mãn điều kiện nào đó.

|  |
| --- |
| **<xsl:if test="expression">…</xsl:if>** |

Các toán tử so sánh:

|  |
| --- |
| = (equal)  != (not equal)  &lt; less than (>)  &gt; greater than (<) |

* Thành phần <xsl:choose>
* Lọc dữ liệu với nhiều trường hợp

|  |
| --- |
| **<xsl:choose>**  **<xsl:when test="expression">**  **... some output ...**  **</xsl:when>**  **<xsl:otherwise>**  **... some output ....**  **</xsl:otherwise>**  **</xsl:choose>** |

* Thuộc tính Select
* In vị trí xuất hiện của một phần tử trong tài liệu.

|  |
| --- |
| **<xsl:value-of select="position()"/>** |

1. Khai báo trong file XML

Khai báo thêm trong file XML:

|  |
| --- |
| <?xml-stylesheet type="text/xsl" href="file\_xsl"?> |

VD: <?xml-stylesheet type="text/xsl" href="sinhvien.xsl"?>

1. Bài tập mẫu
2. Đề bài:

Tạo tập tin SinhVien.xml có định nghĩa kiểu dữ liệu sử dụng XML Schema.

* Giới tính chỉ nhập Nam hoặc Nữ
* Mã lớp nhập theo qui ước: 4 ký tự đầu là chữ cái, 2 ký tự kế tiếp là số

Truy vấn và hiển thị thông tin sinh viên tăng dần theo tên

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **MaSv** | **TenSv** | **GioiTinh** | **NgaySinh** | **MaLop** |
| 1 | sv01 | Phan Tuấn Cường | Nam | 03/03/1993 | CNTT14 |
| 2 | sv02 | Huỳnh Chí Tâm | Nam | 08/03/1993 | CNTT14 |
| 3 | sv03 | Vương Chí Dũng | Nam | 14/04/1991 | CNTT14 |

1. Làm bài:

* B1: Tạo file schema sinhvienchema.xsd

|  |
| --- |
| <?xml version="1.0"?>  <!--  To change this license header, choose License Headers in Project Properties.  To change this template file, choose Tools | Templates  and open the template in the editor.  -->  <xs:schema version="1.0"  xmlns:xs="http://www.w3.org/2001/XMLSchema"  elementFormDefault="qualified">  <xs:element name="listSV">  <xs:complexType>  <xs:sequence>  <xs:element name="SinhVien" maxOccurs="unbounded">  <xs:complexType>  <xs:sequence>  <xs:element name="MaSv" type="xs:string">  </xs:element>  <xs:element name="TenSv" type="xs:string">  </xs:element>  <xs:element name="GioiTinh">  <xs:simpleType>  <xs:restriction base="xs:string">  <xs:pattern value="Nam|Nữ"/>  </xs:restriction>  </xs:simpleType>  </xs:element>  <xs:element name="NgaySinh" type="xs:string">  </xs:element>  <xs:element name="MaLop">  <xs:simpleType>  <xs:restriction base="xs:string">  <xs:pattern value="[a-zA-Z]{4}[0-9]{2}"></xs:pattern>  </xs:restriction>  </xs:simpleType>  </xs:element>  </xs:sequence>  </xs:complexType>  </xs:element>  </xs:sequence>  </xs:complexType>  </xs:element>  </xs:schema> |

* B2: Tạo file XML sinhvien.xml:

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <?xml-stylesheet type="text/xsl" href="sinhvien.xsl"?>  <listSV  xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'  xsi:noNamespaceSchemaLocation='sinhvienchema.xsd'>  <SinhVien>  <MaSv>sv01</MaSv>  <TenSv>Phan Tuấn Cường</TenSv>  <GioiTinh>Nam</GioiTinh>  <NgaySinh>03/03/1993</NgaySinh>  <MaLop>CNTT14</MaLop>  </SinhVien>  <SinhVien>  <MaSv>sv02</MaSv>  <TenSv>Huỳnh Chí Tâm</TenSv>  <GioiTinh>Nam</GioiTinh>  <NgaySinh>08/03/1993</NgaySinh>  <MaLop>CNTT14</MaLop>  </SinhVien>  <SinhVien>  <MaSv>sv03</MaSv>  <TenSv>Vương Chí Dũng</TenSv>  <GioiTinh>Nam</GioiTinh>  <NgaySinh>14/04/1991</NgaySinh>  <MaLop>CNTT14</MaLop>  </SinhVien>  </listSV> |

* B3: Tạo file sinhvien.xsl

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">  <xsl:output method="html"/>  <xsl:template match="/">  <html>  <head>  <title>sinhvien.xsl</title>  </head>  <body>  <h1>Danh sách sinh viên</h1>  <table border="1" cellpadding="0" cellspacing="0">  <tr>  <th>STT</th>  <th>MaSv</th>  <th>TenSv</th>  <th>GioiTinh</th>  <th>NgaySinh</th>  <th>MaLop</th>  </tr>  <xsl:for-each select="listSV/SinhVien">  <xsl:sort select="MaSv" order="descending"/>  <tr>  <td>  <xsl:value-of select="position()"/>  </td>  <td>  <xsl:value-of select="MaSv"/>  </td>  <td>  <xsl:value-of select="TenSv"/>  </td>  <td>  <xsl:value-of select="GioiTinh"/>  </td>  <td>  <xsl:value-of select="NgaySinh"/>  </td>  <td>  <xsl:value-of select="MaLop"/>  </td>  </tr>  </xsl:for-each>  </table>  </body>  </html>  </xsl:template>  </xsl:stylesheet> |