# **Home Work – 06**

**Excercise 1: According to the following DTD, write the valid XML document.**

How to define that book must appear one or more times in booklist.

<!DOCTYPE booklist [

<!ELEMENT booklist (book)+>

<!ELEMENT book (bookname, author, price)>

<!ELEMENT bookname (#PCDATA)>

<!ELEMENT author (#PCDATA)>

<!ELEMENT price (#PCDATA)>

]>

**Solution:**

<?xml version = “1.0” encoding= “UTF-8”?>

<booklist>

<book>

<bookname>The Great Gatsby</bookname>

<author>F. Scott Fitzgerald</author>

<price>44</price>

</book>

<book>

<bookname>Think and Grow Rich</bookname>

<author>Napoleon Hill</author>

<price>99</price>

</book>

<book>

<bookname>Sapiens: A Brief History of Humankind</bookname>

<author>Yuval Noah Harari</author>

<price>75</price>

</book>

</booklist>

**Excercise 2: Explain the following DTD and according to this DTD, write the valid XML document.**

<!DOCTYPE booklist [

<!ELEMENT booklist (book)+>

<!ELEMENT book (bookname\*, author+, price?, homepage)>

<!ELEMENT bookname (#PCDATA)>

<!ELEMENT author (#PCDATA)>

<!ELEMENT price (#PCDATA)>

<!ELEMENT homepage (#PCDATA)>

]>

**Solution:**

<booklist>

<book>

<!-- It is possible not to have a <bookname> tag in the XML <bookname></bookname>-->

<author>Dan Brown</author>

<price>25</price>

<homepage></homepage>

</book>

<book>

<bookname>Think and Grow Rich</bookname>

<author>Napoleon Hill</author>

<price>99</price>

<homepage>abcd</homepage>

</book>

</booklist>

*The above DTD is interpreted is mentioned below:*

<!DOCTYPE booklist - Defines that the root element of the document is booklist

<!ELEMENT booklist - Defines the node booklist must contain node book

<!ELEMENT (book)+ - Defines the node book that contains the the elements: bookname, author, price and homepage and '+' states that the node book must appear more than once

<!ELEMENT (bookname)\* - Defines that the node bookname is of the datatype '#PCDATA' and '\*' defines that the bookname can appear 0 or more times.

<!ELEMENT (author)+ - Defines that the node author is of the datatype '#PCDATA' and '+' defines that the author must appear one or more times.

<!ELEMENT (price?) - Defines that the node price is of the datatype '#PCDATA' and '?' denotes that the price can appear zero or one time.

<!ELEMENT homepage - Defines that the node homepage is of the datatype '#PCDATA'.

**Excercise 3: Multiple choices on DTD element declaration**

**3.1 According to the following DTD, choose the right XML description:**

<!DOCTYPE booklist [

<!ELEMENT booklist (book)+>

<!ELEMENT book (bookname, author, price)>

<!ELEMENT bookname (#PCDATA)>

<!ELEMENT author (#PCDATA)>

<!ELEMENT price (#PCDATA)>

]>

**Solution:**

Since, ordering does not matter in an XML. So, both the options are correct.

Option C: Both are right.

**3.2 According to the following DTD, choose the right XML description:**

<!ELEMENT spouse (wife|husband)>

**Solution:**

Option E: (1), (2) and (5) are right

**3.3 According to the following DTD, choose the right XML description from 3.2:**

<!ELEMENT spouse (wife|husband)?>

**Solution:**

**Choice (1):**

<spouse>

<wife>Alice</wife>

</spouse>

**Choice (2):**

<spouse>

<husband>Tom</husband>

</spouse>

**Choice (5):**

<spouse>

</spouse>

**Exercise 4: Attribute declaration**

**4.1 <!ATTLIST film type CDATA "action">**

Are the following xml documents valid

<film type="kid">The Lion King</film>

Or

<film>The Lion King</film>

Or

<film type="thriller">Thriller</film>

**Solution:**

**<film type="kid">The Lion King</film> and**

**<film type="thriller">Thriller</film> are valid.**

**4.2 <!ATTLIST film type CDATA #REQUIRED>**

Are the following xml documents valid

<film type="kid">The Lion King</film>

Or

<film>The Lion King</film>

Or

<film type="thriller">Thriller</film>

**Solution:**

**<film type="kid">The Lion King</film> and**

**<film type="thriller">Thriller</film> are valid.**

**4.3 <!ATTLIST film type CDATA #IMPLIED>**

Are the following xml documents valid

<film type="kid">The Lion King</film>

Or

<film>The Lion King</film>

Or

<film type="thriller">Thriller</film>

**Solution:**

**All are valid as if the attribute-value is implied then the attribute is optional.**

**4.4 <!ATTLIST film type CDATA #FIXED "action">**

Are the following xml documents valid

<film type="kid">The Lion King</film>

Or

<film>The Lion King</film>

Or

<film type="thriller">Thriller</film>

**Solution:**

**All are invalid XMLs as the attribute-value should be ‘action’.**

**4.5. Complete the following XML document to make it valid:**

**Solution:**

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

<!DOCTYPE productlist [

<!ELEMENT productlist (product)\*>

<!ELEMENT product (title, image, price)>

<!ATTLIST product number ID #REQUIRED>

<!ELEMENT title (#PCDATA)>

<!ELEMENT image EMPTY>

<!ATTLIST image source ENTITY #REQUIRED>

<!ENTITY L37h SYSTEM "L37h.jpg">

<!ENTITY P42h SYSTEM "P42h.jpg">

<!ELEMENT price (#PCDATA)>

]>

<productlist>

<product ID = “1”>

<title>Toothpaste</title>

<image source = “abc.jpg”>&L37h;&P42h</image>

<price>3.59</price>

</product>

<product ID = “2”>

<title>Brush</title>

<image source = “xyz.jpg”>&L37h;&P42h</image>

<price>1.25</price>

</product>

</productlist>

**Exercise 5: According to the following external DTD,**

<!-- file name publish.dtd -->

<!ELEMENT publishInfo (publisher, ISBN, pubDate)>

<!ELEMENT publisher (#PCDATA)>

<!ELEMENT ISBN (#PCDATA)>

<!ELEMENT pubDate (#PCDATA)>

complete the rest of the XML document:

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

<-- file name book.xml -->

<!DOCTYPE bookCategory SYSTEM "publish.dtd" [

<!ELEMENT bookCategory (book\*)>

<!ELEMENT book (title, author, publishInfo, price)>

<!ELEMENT title (#PCDATA)>

<!ELEMETN author (#PCDATA)>

<!ELEMENT price (#PCDATA)>

<!ATTLIST price currency CDATA "Dollar">

]>

**Solution:**

<!-- file name publish.dtd -->

<!ELEMENT publishInfo (publisher, ISBN, pubDate)>

<!ELEMENT publisher (#PCDATA)>

<!ELEMENT ISBN (#PCDATA)>

<!ELEMENT pubDate (#PCDATA)>

complete the rest of the XML document:

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

<-- file name book.xml -->

<!DOCTYPE bookCategory SYSTEM "publish.dtd" [

<!ELEMENT bookCategory (book\*)>

<!ELEMENT book (title, author, publishInfo, price)>

<!ELEMENT title (#PCDATA)>

<!ELEMENT author (#PCDATA)>

<!ENTITY publishInfo SYSTEM “publish.dtd”>

<!ELEMENT price (#PCDATA)>

<!ATTLIST price currency CDATA "Dollar">

]>

<bookcategory>

<book>

<title>Harry Potter</title>

<author>J.K. Rowlings</author>

<publishInfo>

<publisher>abcdef</publisher>

<ISBN>8959-666-486</ISBN>

<pubDate>02122010</pubDate>

</publishInfo>

<price currency ="Dollar">6.95<price>

</book>

</bookcategory>

**Exercise 6: write the corresponding internal and external DTD for this xml document:**

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

<bookTitle>

<title>XML Introduction</title>

<subtitle section="1" >next generation web language</subtitle>

</bookTitle>

**Solution:**

**Internal DTD**

<?xml version = “1.0” encoding="UTF-8" ?>

<!DOCTYPE bookTitle [

<!ELEMENT bookTitle(title, subtitle)>

<!ELEMENT title (#PCDATA)>

<!ELEMENT subtitle (#PCDATA)>

<!ATTLIST subtitle section CDATA “1”>

]>

<bookTitle>

<title>XML Introduction</title>

<subtitle section="1" >next generation web language</subtitle>

</bookTitle>

**External DTD**

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

<!DOCTYPE bookTitle SYSTEM “bookTitle.dtd”>

<bookTitle>

<title>XML Introduction</title>

<subtitle section="1" >next generation web language</subtitle>

</bookTitle>

**Exercise 7: internal parameter entity: please define (name, gender, birth, tel+) as internal parameter entity and use it in following external DTD for element type of researchPersonal, marketingPersonal, salesPersonal, then further to call external DTD and to complete corresponding XML documents to represent the following table:**

personalInfo

type name gender birth tel tel

-------------------------------------------------------------------------------

researchPersonal Alice female 5/8/1982 61327678

salesPersonal Thomas male 6/9/1978 15663267889 63276766

salePersonal Mike male 2/26/1981 66326789

marketingPersonal Ed male 10/11/1982 63271688

**Solution:**

<! -- file name personalInfo.dtd-- >

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

<!DOCTYPE personalInfo[

<!ELEMENT personalInfo(name, gender, birth, date, tel)/>

<!ATTLIST personalInfo type (#CDATA)/>

]>

<! -- file name personalInfo.dtd-- >

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

<!ELEMENT personalInfo SYSTEM personalInfo.dtd/>

<!ELEMENT name (#CDATA)/>

<!ELEMENT gender (#CDATA)/>

<!ELEMENT birth (#CDATA)/>

<!ELEMENT date (#CDATA)/>

<!ELEMENT tel (#CDATA)/>

]>

<personalInfo type = "researchPersonal">

<name>Alice</name>

<gender>female</gender>

<birth>5/8/1982</birth>

<tel>61327678</tel>

</personalInfo>

<personalInfo type = "salesPersonal">

<name>Thomas</name>

<gender>male</gender>

<birth>6/9/1978</birth>

<tel>15663267889</tel>

<tel>63276766</tel>

</personalInfo>

<personalInfo type = "salesPersonal">

<name>Mike</name>

<gender>male</gender>

<birth>2/26/1981</birth>

<tel>66326789</tel>

</personalInfo>

<personalInfo type = "marketingPersonal">

<name>Ed</name>

<gender>male</gender>

<birth>10/11/1982</birth>

<tel>63271688</tel>

</personalInfo>

**XML Stylesheet**

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

<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">

<xsl:template match="/">

<html>

<head>

<title>personalInfo</title>

</head>

<body>

<table>

<tr>

<th>type</th>

<th>name</th>

<th>gender</th>

<th>birth</th>

<th>tel</th>

</tr>

<xsl:for-each select="personalInfo">

<tr>

<td><xsl:value-of select="personalInfo/@type"></td>

<td><xsl:value-of select="name"></td>

<td><xsl:value-of select="gender"></td>

<td><xsl:value-of select="birth"></td>

<td><xsl:value-of select="tel"></td>

</tr>

</xsl:for-each>

</table>

</body>

</html>

</xsl:template>

</xsl:stylesheet>

**Exercise 8: change exercise 7 into external three DTDs for researchPersonal, marketingPersonal, and salesPersonal, using external parameter entity to call these three DTDs and complete the XML document.**

**Solution:**

<!-- filename – researchPersonal.dtd -- >

<!ELEMENT personalInfo(name, gender, birth, tel+)

<!ATTLIST personalInfo type (#PCDATA) value= “researchPersonal”>

]>

<!-- filename – marketingPersonal.dtd -- >

<!ELEMENT personalInfo(name, gender, birth, tel+)

<!ATTLIST personalInfo type (#PCDATA) value= “researchPersonal”>

]>

<!-- filename – salesPersonal.dtd -- >

<!ELEMENT personalInfo(name, gender, birth, tel+)

<!ATTLIST personalInfo type (#PCDATA) value= “researchPersonal”>

]>

<!—filename – PersonalInfo.dtd -->

<!DOCTYPE personalInfo[

<!ELEMENT personalInfo(name, gender, birth, tel+)

<!ATTLIST personalInfo type SYSTEM “(researchPersonal.dtd | marketingPersonal.dtd | salesPersonal.dtd)”>

<!ELEMENT name (#CDATA)>

<!ELEMENT gender (#CDATA)>

<!ELEMENT birth (#CDATA)>

<!ELEMENT tel (#CDATA)>

]>

<personalInfo type = "researchPersonal">

<name>Alice</name>

<gender>female</gender>

<birth>5/8/1982</birth>

<tel>61327678</tel>

</personalInfo>

<personalInfo type = "salesPersonal">

<name>Thomas</name>

<gender>male</gender>

<birth>6/9/1978</birth>

<tel>15663267889</tel>

<tel>63276766</tel>

</personalInfo>

<personalInfo type = "salesPersonal">

<name>Mike</name>

<gender>male</gender>

<birth>2/26/1981</birth>

<tel>66326789</tel>

</personalInfo>

<personalInfo type = "marketingPersonal">

<name>Ed</name>

<gender>male</gender>

<birth>10/11/1982</birth>

<tel>63271688</tel>

</personalInfo>

**PART 2: XML Schema Exercise**

**Exercise 1: Defining an xml document using xsd as its schema for the following table:**

Internet Club

Member Name Years of Membership

-------------------------------------------------

Chris Smith 4

Judi Zimmerman 5

Michael Calder 3

**Solution:**

<?xml version="1.0"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name = "InternetClub">

<xs:complexType>

<xs:sequence>

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

<xs:element name = "YearsofMembership" type = "xs:integer"/>

</xs:sequence>

</xs:complexType>

</xs:element name>

</xs:schema>

**Exercise 2: Representing the following books in XML, and write corresponding XML schema and make them well-formed and valid:**

* **Antoniou, G. and F. van Harmelen. 2004, A Semantic Web Primer. Cambridge, MA: MIT Press.**
* **Daconts, M.C., L.J. Orbst, and K.T. Smith. 2003, The Semantic Web: A Guide to the Future of XML, Web Services, and Knowledge Management. New York: Wiley.**

**Solution:**

**XML Document**

<bookstore>

<book>

<author>Antoniou</author>

<author>G. and F. van Harmelen</author>

<year>2004</year>

<title>A Semantic Web Primer</title>

<publishedLocation>Cambridge, MA</publishedLocation>

<publisher>MIT Press</publisher>

</book>

<book>

<author>Daconts, M.C</author>

<author>L.J. Orbst</author>

<author>K.T. Smith</author>

<year>2003</year>

<title>The Semantic Web: A Guide to the Future of XML, Web Services, and Knowledge Management</title>

<publishedLocation>New York</publishedLocation>

<publisher>Wiley</publisher>

</book>

</bookstore>

**XML Schema**

<?xml version="1.0"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name ="bookstore">

<xs:element name ="book>

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

<xs:element name="year" type ="xs:integer"/>

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

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

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

</xs:element>

</xs:schema>

**Exercise 3: Creating corresponding well-formed and valid XML Schema for the following XML document:**

<?xml version="1.0" encoding="ISO-8859-1" ?>

<!-- file name shiporder.xml -->

<shiporder orderid="889923"

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

xsi:noNamespaceSchemaLocation="shiporder.xsd">

<orderperson>John Smith</orderperson>

<shipto>

<name>Ola Nordmann</name>

<address>Langgt 23</address>

<city>4000 Stavanger</city>

<country>Norway</country>

</shipto>

<item>

<title>Empire Burlesque</title>

<note>Special Edition</note>

<quantity>1</quantity>

<price>10.90</price>

</item>

<item>

<title>Hide your heart</title>

<quantity>1</quantity>

<price>9.90</price>

</item>

</shiporder>

**Solution:**

<?xml version="1.0"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="shiporder">

<xs:complexType>

<xs:sequence>

<xs:attribute name ="orderid" type="xs:integer"/>

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

<xs:element name ="shipto">

<xs:complexType>

<xs:sequence>

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

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

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

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

</xs:sequence>

</xs:complexType>

</xs:element name ="shipto">

</xs:sequence>

</xs:complexType>

</xs:elemnt name>

</xs:schema>

**Exercise 4: write an XML schema and XML document for a library book transaction, it should describe the following information:**

Librarian (Lender) Michael Calder lent book to borrower Craig Gavin on Oct 15, 2007

The contact information of Michael Calder is: 41 McMahons Rd, Frankston, WI, tel: 9903 4563

The contact information of Craig Gavin is: 56 Overport Rd, Karingal, WI, tel: 9903 3455

Michael made a note: lender wants these books back in two weeks!

The lent books are:

bookID title pubDate replacementValue maxDaysOut

-------------------------------------------------------------------------------------------------

123-4567-890 XML Oct 20, 2001 15.95 14

123-4567-891 Artificial Intelligence May 30, 2000 30.25 14

123-4567-892 Semantic Web April 10, 2004 17.99 14

123-4567-893 Agent June 10, 2005 11.95 14

**Solution:**

**XML Document**

<?xml version="1.0"?>

<libraryTransaction>

<transactionDate>10152007</transactionDate>

<person role="lender">

<name>Michael Calder</name>

<address>

<line1>41 McMahons Rd</line1>

<line2>Frankston</line2>

<state>WI</state>

</address1>

<tel>99034563</tel>

</person>

<person role="borrower">

<name>Craig Gavin</name>

<address>

<line1>56 Overport Rd</line1>

<line2>Karingal</line2>

<state>WI</state>

</address>

<tel>99033455</tel>

</person>

<books>

<book>

<bookID>123-4567-890</bookID>

<title>XML</title>

<pubDate>10202001</pubDate>

<replacementValue>15.95</replacementValue>

<maxDaysOut>14</maxDaysOut>

<book>

<book>

<bookID>123-4567-891</bookID>

<title>Artificial Intelligence</title>

<pubDate>05302000</pubDate>

<replacementValue>30.25</replacementValue>

<maxDaysOut>14</maxDaysOut>

<book>

<book>

<bookID>123-4567-892</bookID>

<title>Semantic Web</title>

<pubDate>04102004</pubDate>

<replacementValue>17.99</replacementValue>

<maxDaysOut>14</maxDaysOut>

<book>

<book>

<bookID>123-4567-893</bookID>

<title>Agent</title>

<pubDate>06102005</pubDate>

<replacementValue>11.95</replacementValue>

<maxDaysOut>14</maxDaysOut>

<book>

</books>

</libraryTransaction>

</xml>

**XML Schema**

<?xml version="1.0"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name = "libraryTransaction">

<xs:complexType>

<xs:sequence>

<xs:element name ="transactionDate" type="xs:date"/>

<xs:element name ="person">

<xs:complexType>

<xs:sequence>

<xs:attribute name="role" type="xs:string"/>

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

<xs:element name="address">

<xs:element name="line1" type="xs:string"/>

<xs:element name="line2" type="xs:string"/>

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

</xs:element>

<xs:element name="tel" type="xs:integer"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<xs:element name="books">

<xs:complexType>

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

<xs:complexType>

<xs:sequence>

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

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

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

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

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

</xs:sequence>

</xs:complexType>

</xs:complexType>

</xs:sequence>

</xs:complexType>

</xs:schema>

**PART 1: XPath and XQuery Exercise**

**The following exercises are based on this XML document (books.xml):**

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

<!-- file name: books.xml -->

<!DOCTYPE bookstore [

<!ELEMENT bookstore (book)+>

<!ELEMENT book (title, author+, year, price)>

<!ATTLIST book category CDATA "WEB">

<!ELEMENT title (#PCDATA)>

<!ATTLIST title lang CDATA "en">

<!ELEMENT author (#PCDATA)>

<!ELEMENT year (#PCDATA)>

<!ELEMENT price (#PCDATA)>

]>

<bookstore>

<book category="COOKING">

<title lang="en">Everyday Italian</title>

<author>Giada De Laurentiis</author>

<year>2005</year>

<price>30.00</price>

</book>

<book category="CHILDREN">

<title lang="en">Harry Potter</title>

<author>J. K. Rowling</author>

<year>2005</year>

<price>29.99</price>

</book>

<book category="WEB">

<title lang="en">XQuery Kick Start</title>

<author>James McGovern</author>

<author>Per Bothner</author>

<author>Kurt Cagle</author>

<author>James Linn</author>

<author>Vaidyanathan Nagarajan</author>

<year>2003</year>

<price>49.99</price>

</book>

<book category="WEB">

<title lang="en">Learning XML</title>

<author>Erik T. Ray</author>

<year>2003</year>

<price>39.95</price>

</book>

</bookstore>

**Exercise 1: According to books.xml in Exercise 1, please answer the following questions**

**What are the parent nodes of node "title"**

**Solution:** <book> and <bookstore> are the parent nodes of node ‘title’.

**What are the child nodes of node "book"**

**Solution:** <title>, <author>, <year> and <price> are the child nodes of the node ‘book’.

**What are the descendant nodes of node “bookstore”**

**Solution:** <book>, <title>, <author>, <year> and <price> are the descendant nodes of the node ‘bookstore’.

**What are the ancestor nodes of node "title"**

**Solution:** <book> and <bookstore> are the ancestor of the node ‘title’.

**What are the preceding nodes of book with "title" having value "Harry Porter"**

**Solution:** There are no preceding nodes.

**What are the following nodes of node "title"**

**Solution:** <author>, <year> and <price> are the following nodes of the node “title”.

**What are the preceding siblings of node "price"**

**Solution:** There are no preceding siblings of the node ‘price’.

**What are the following siblings of node "title"**

**Solution:** No siblings of the node ‘title’

**Exercise 2: According to books.xml, form the following Xpath to select the authors.**

**Solution:** doc(“book.xml”)/bookstore/book/authors

**Exercise 3: According to books.xml, form the following XQuery to select the authors with the book price more than 40 and order the result according to the title.**

**Solution:**

for $x in doc(“books.xml”)/bookstore/book

where $x/price > 40

order by $x/title

return $x/title

**PART 2: XSLT Exercise**

Look at the following XML document,

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

<!-- file name student.xml -->

<roster>

<student class="Financing">

<SN>2005133110</SN>

<name>Thomas Luger</name>

<sex>male</sex>

<birthday>1986.10.21</birthday>

<score>605</score>

</student>

<student class="Financing">

<SN>2005133115</SN>

<name>Alice Schutern</name>

<sex>female</sex>

<birthday>1985.8.25</birthday>

<score>596</score>

</student>

<student class="International Financing">

<SN>2005131136</SN>

<name>Helena Wieser</name>

<sex>female</sex>

<birthday>1986.5.28</birthday>

<score>611</score>

</student>

<student class="Insurance">

<SN>2005132012</SN>

<name>Ingrid Harth</name>

<sex>female</sex>

<birthday>1986.6.19</birthday>

<score>602</score>

</student>

<student class="Financing">

<SN>2005133108<SN>

<name>Judi Stollberg</name>

<sex>female</sex>

<birthday>1987.3.15</birthday>

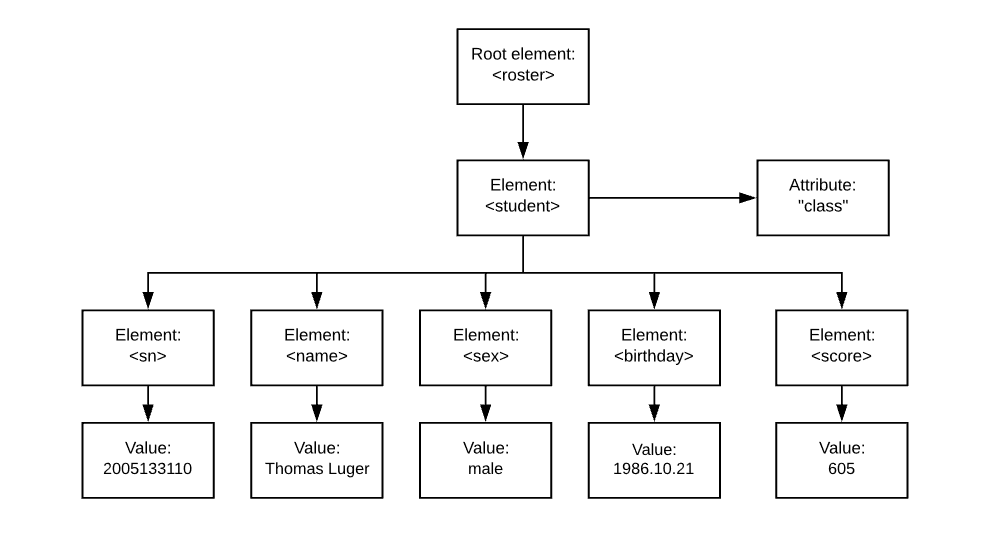
<score>581</score>

</student>

</roster>

**Exercise 1: draw the tree structure of this XML document**

**Solution:**



**Exercise 2: Using <xsl:apply-template> in student1.xsl to display student.xml in the same table.**

**Solution:**

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

<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"

version="1.0">

<xsl: template match = "/">

<html>

<head>

<title>People Report</title>

</head>

<body>

<table>

<tr>

<th>sn</th>

<th>name</th>

<th>sex</th>

<th>birthday</th>

<th>score</th>

</tr>

<xsl: for-each select = "/roster/student">

<tr>

<td><xsl: value-of ="sn"/></td>

<td><xsl: value-of ="name"/></td>

<td><xsl: value-of ="sex"/></td>

<td><xsl: value-of = "birthday"/></td>

<td><xsl: value-of ="score"/></td>

</tr>

<xsl: for-each>

</table>

</body>

</html>

</xsl:template>

</xsl:stylesheet>

**Exercise 3: Create a similar table to display the Financing student information.**

**Solution:**

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

<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"

version="1.0">

<xsl: template match = "/">

<html>

<head>

<title>People Report</title>

</head>

<body>

<table>

<tr>

<th>sn</th>

<th>name</th>

<th>sex</th>

<th>birthday</th>

<th>score</th>

</tr>

<xsl: for-each select = "roster[student/@class = ‘Financing’]">

<tr>

<td><xsl: value-of ="/sn"/></td>

<td><xsl: value-of ="name"/></td>

<td><xsl: value-of ="sex"/></td>

<td><xsl: value-of = "birthday"/></td>

<td><xsl: value-of ="score"/></td>

</tr>

<xsl: for-each>

</table>

</body>

</html>

</xsl:template>

</xsl:stylesheet>

**Exercise 4: Create a different format to display the student information as followings:**

**Solution:**

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

<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">

<xsl:template match="/">

<html>

<head>

<title>Student Information</title>

</head>

<body>

<p>

<xsl:template match = "roster/student">

<xsl:apply-template select = "SN"/>

<xsl:apply-template select = "name"/>

<xsl:apply-template select = "sex"/>

<xsl:apply-template select = "birthday"/>

<xsl:apply-template select = "score"/>

</xsl:template>

</p>

<xsl:template match="name">

<b><xsl:value-of select="."/></b>

<br/>

</xsl:template>

</p>

<p>

<xsl:template match="sex">

Sex:<xsl:value-of select="."/>

<br/>

</xsl:template>

</p>

<p>

<xsl:template match="birthday">

Birthday:<xsl:value-of select="."/>

<br/>

</xsl:template>

</p>

<br/>

<p>

……

<br/>

</p>

</body>

</html>

</xsl:template>

</xsl:stylesheet>

Exercise 5: Display student information in a similar table with different classes using different background colors.

**Solution:**

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

<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"

version="1.0">

<xsl: template match = "/">

<html>

<head>

<title>People Report</title>

</head>

<body>

<table>

<tr>

<th bgcolor=”#FF0000”>sn</th>

<th bgcolor=”#00FF00”>name</th>

<th bgcolor=”#cc6699”>sex</th>

<th bgcolor=”#ff9933”>birthday</th>

<th bgcolor=” #ccff99”>score</th>

</tr>

<xsl: for-each select = "roster[student/@class = ‘Financing’]">

<tr>

<td bgcolor=”#FF0000”><xsl: value-of ="sn"/></td>

<td bgcolor=”#00FF00”><xsl: value-of ="name"/></td>

<td bgcolor=”#cc6699”><xsl: value-of ="sex"/></td>

<td bgcolor=”#ff9933”><xsl: value-of = "birthday"/></td>

<td bgcolor=” #ccff99”><xsl: value-of ="score"/></td>

</tr>

<xsl: for-each>

</table>

</body>

</html>

</xsl:template>

</xsl:stylesheet>