Activity: Creating an XML Schema

Problem Statement

CyberShoppe, a toy and book store in the United States, sends its product information from the head office to the branch offices. The product details must be stored in a consistent format. Restrictions must be placed on the type of data that can be saved in the data store, in order to ensure uniformity and consistency of information.

The product details include the product name, a brief description, product price, and the available quantity on hand. The price of the product must always be greater than zero.

Solution

To solve the preceding problem, perform the following tasks:

1. Identify the elements required to store the data.
2. Identify the data types of the contents.
3. Identify the method to declare a simple type element.
4. Identify the method to declare a complex type element.
5. Create an XML schema.
6. Create an XML document that conforms to the schema.
7. Validate the XML document against the schema.

Task 1: Identifying the Elements Required to Store the Data

As per the problem statement, the elements required in the XML document are as follows.

Element	***Description***
`PRODUCTDATA`	*Indicates that the data specific to various products is being stored in the document. It acts as the root element for all other elements.*
`PRODUCT`	*Represents the product details, such as product name, description, price, and quantity on hand for each product.*
`PRODUCTNAME`	*Represents the product name.*
`DECRIPTION`	*Represents the product description.*
`PRICE`	*Represents the product price.*

Element	Description
QUANTITY	*Represents the availability quantity of each product.*

Elements Required in the XML Document

Task 2: Identifying the Data Types of the Contents

As per the problem statement, the data types for the contents of the elements are as follows.

Element	Data Type	Description
PRODUCTDATA	*Complex*	*A complex type element that can hold other elements, attributes, and mixed content.*
PRODUCT	*Complex*	*A complex type element that can hold other elements, attributes, and mixed content.*
PRODUCTNAME	*String*	*A simple type element that contains values of the string data type.*
DESCRIPTION	*String*	*A simple type element that contains values of the string data type.*
PRICE	*positiveInteger*	*A simple type element that contains values of the positiveInteger data type. This satisfies the condition that the product price must be greater than zero.*
QUANTITY	*Integer*	*A simple type element that contains values of the integer data type.*

Data Types of the Contents

Task 3: Identifying the Method to Declare a Simple Type Element

As per the problem, the following simple elements can be declared in the XSD as follows:

```
<xsd:element name="PRODUCTNAME" type="xsd:string"/>
<xsd:element name="DESCRIPTION" type="xsd:string"/>
<xsd:element name="PRICE" type="xsd:positiveInteger"/>
<xsd:element name="QUANTITY" type="xsd:nonNegativeInteger"/>
```

Note that the `QUANTITY` element is declared as a `nonNegativeInteger` type. Similarly, the `PRICE` element is declared of the type `positiveInteger` to ensure that it is always greater than zero.

Task 4: Identifying the Method to Declare a Complex Type Element

In the CyberShoppe scenario, you require two complex type elements, `PRODUCTDATA` and `PRODUCT`. You can create complex type elements by associating them with complex data types. You can use the `element` element of XSD to declare a complex type element. You can use the `complexType` element of XSD to create a complex data type.

Task 5: Creating an XML Schema

To create an XML schema that contains the element and attribute declarations to store the data, type the following code in Notepad, and save it as `products.xsd`:

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <xsd:element name="PRODUCTDATA" type="prdata"/>
   <xsd:complexType name="prdata">
      <xsd:sequence>
         <xsd:element name="PRODUCT" type="prdt"/>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:complexType name="prdt">
      <xsd:sequence>
         <xsd:element name="PRODUCTNAME" type="xsd:string"/>
         <xsd:element name="DESCRIPTION" type="xsd:string"/>
         <xsd:element name="PRICE" type="xsd:positiveInteger"/>
         <xsd:element name="QUANTITY" type="xsd:nonNegativeInteger"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:schema>
```

Task 6: Creating an XML Document Conforming to the Schema

To create the XML document, type the following code in Notepad, and save the file as `products.xml`:

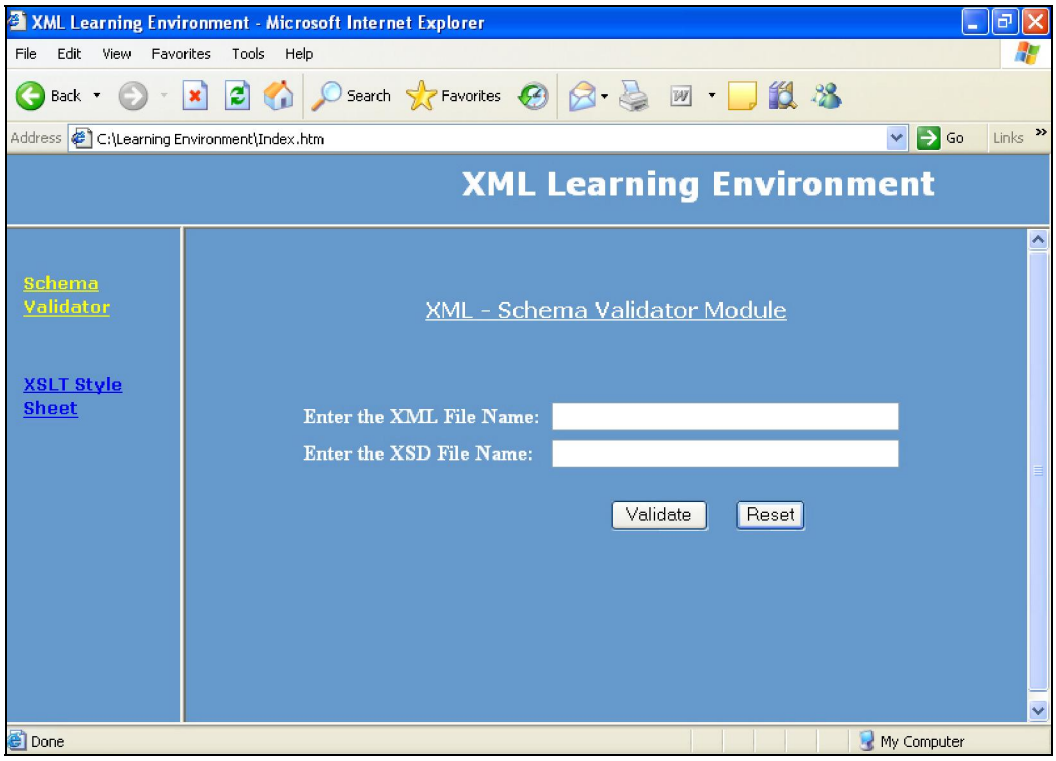
```
<?xml version="1.0"?>
<PRODUCTDATA>
   <PRODUCT>
      <PRODUCTNAME>Barbie Doll</PRODUCTNAME>
      <DESCRIPTION>This is a doll for children aged 11 and
      above</DESCRIPTION>
```

```
        <PRICE>200</PRICE>
        <QUANTITY>12</QUANTITY>
    </PRODUCT>
</PRODUCTDATA>
```

Task 7: Validating the XML Document Against the Schema

To validate the structure of the XML document, perform the following steps:

1. Open `Index.htm` from the `Learning Environment` folder, and click the **Schema Validator** link. This opens the **XML – Schema Validator Module** form, as shown in the following figure.



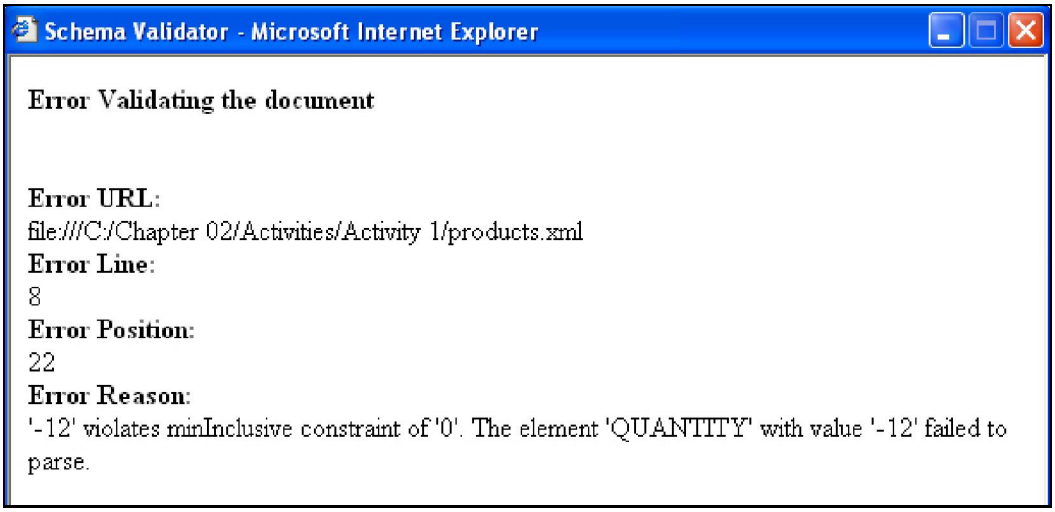
XML – Schema Validator Form

The **Schema Validator** link is used to validate the XML document against the XML schema. It uses the MSXML 6.0 parser.

2. In the **Enter the XML File Name** text box, type the path for `products.xml` file.
3. In the **Enter the XSD File Name** text box, type the path for `products.xsd` file.

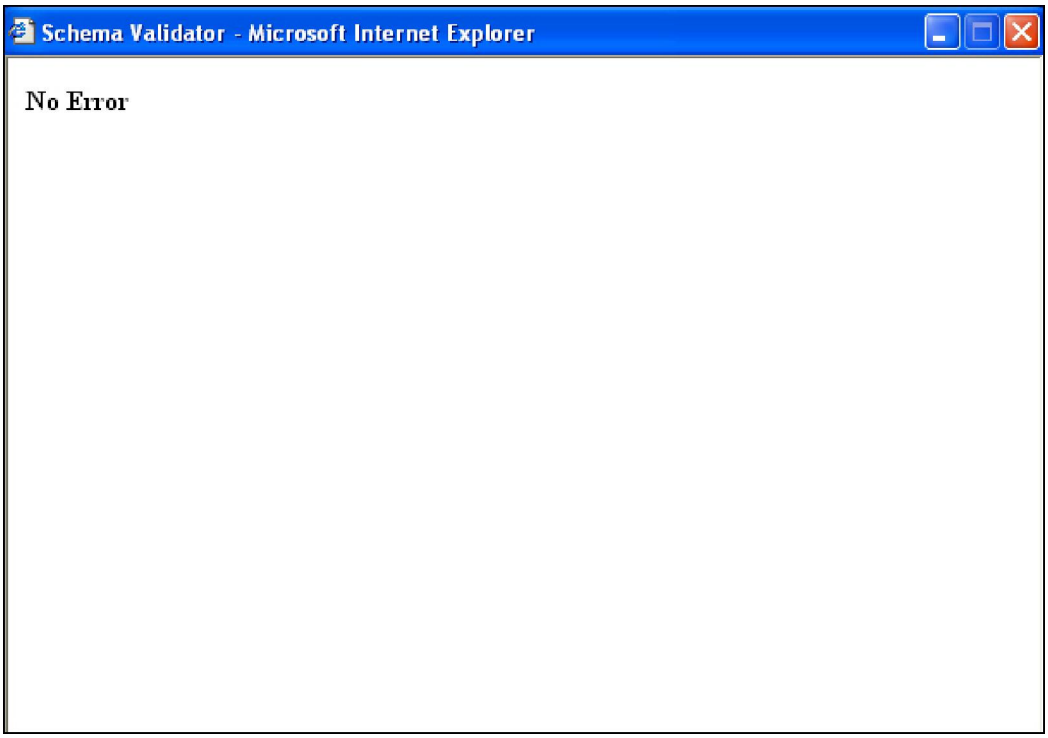
   If the XML and XSD files are located in the Learning Environment folder, you need not type the complete path. If they are located in some other folder, you need to type the complete path along with the file names.
4. Click the **Validate** button.

If the XML document is not created according to the schema, a new page with an error message is displayed. For example, if in the XML document the value of the `QUANTITY` element is given as -12. Then in this case as the XML document will not be according to the schema, an error message will be displayed, as shown in the following figure.



XML Schema Validator Showing Error Message

If the XML document conforms to the schema, a page is displayed containing the text, **No Error**, as shown in the following figure.



XML Schema Validator Showing No Error Message

Note

The easiest way to find errors in the schema is to save the XSD file as an XML document, and then open it in the browser. You can check whether the XSD schema is well-formed. A majority of the errors can be eliminated by checking whether the schema document is well-formed.