XSLT uses XPath to find information in an XML document. XPath is used to navigate through elements and attributes in XML documents.

Correct Style Sheet Declaration

The root element that declares the document to be an XSL style sheet is <xsl:stylesheet> or <xsl:transform>.

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

Link the XSL Style Sheet to the XML Document

Add the XSL style sheet reference to your XML document ("cdcatalog.xml"):

<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="cdcatalog.xsl"?>
<catalog>

XSLT <xsl:template> Element

An XSL style sheet consists of one or more set of rules that are called templates.

A template contains rules to apply when a specified node is matched.
The match attribute is used to associate a template with an XML element. The match attribute can also be used to define a template for the entire XML document. The value of the match attribute is an XPath expression (i.e. match="/" defines the whole document).

<xsl:template match="/">

The <xsl:template> element defines a template. The match="/" attribute associates the template with the root of the XML source document.
The content inside the <xsl:template> element defines some HTML to write to the output.

XSLT <xsl:value-of> Element:

The <xsl:value-of> element can be used to extract the value of an XML element and add it to the output stream of the transformation:

```
<tr>
    <td><xsl:value-of select="catalog/cd/title"/></td>
    <td><xsl:value-of select="catalog/cd/artist"/></td>
  </tr>
```

Note: The select attribute, in the example above, contains an XPath expression. An XPath expression works like navigating a file system; a forward slash (/) selects subdirectories.

The <xsl:for-each> Element:
The XSL <xsl:for-each> element can be used to select every XML element of a specified node-set:

```
<xsl:for-each select="catalog/cd">
  <tr>
    <td><xsl:value-of select="title"/></td>
    <td><xsl:value-of select="artist"/></td>
  </tr>
  </xsl:for-each>
```

Filtering the Output

We can also filter the output from the XML file by adding a criterion to the select attribute in the <xsl:for-each> element.

```
<xsl:for-each select="catalog/cd[artist='Bob Dylan']">
```

Legal filter operators are:

```
  =  (equal)
  != (not equal)
  &lt; less than
  &gt; greater than
```

```
<xsl:for-each select="catalog/cd[artist='Bob Dylan']">
  <tr>
    <td><xsl:value-of select="title"/></td>
    <td><xsl:value-of select="artist"/></td>
  </tr>
  </xsl:for-each>
```

XSLT <xsl:sort> Element:

To sort the output, simply add an <xsl:sort> element inside the <xsl:for-each> element in the XSL file

```
<xsl:for-each select="catalog/cd">
    <xsl:sort select="artist"/>
    <tr>
      <td><xsl:value-of select="title"/></td>
      <td><xsl:value-of select="artist"/></td>
    </tr>
  </xsl:for-each>
```

XSLT <xsl:if> Element:
To put a conditional if test against the content of the XML file, add an <xsl:if> element to the XSL document**.**
Syntax:

```
<xsl:if test="expression">
  ...some output if the expression is true...
</xsl:if>
```

To add a conditional test, add the <xsl:if> element inside the <xsl:for-each> element in the XSL file