MEDICAPS UNIVERSITY	MEDICAPS UNIVERSITY, INDORE	YEAR: 2025-2026
	LAB- MANUAL	SEM: EVEN

Programming with XML

CS3EL08

Lab Manual

INDEX

Practical 1

AIM: To create a simple XML document to display an Address Book

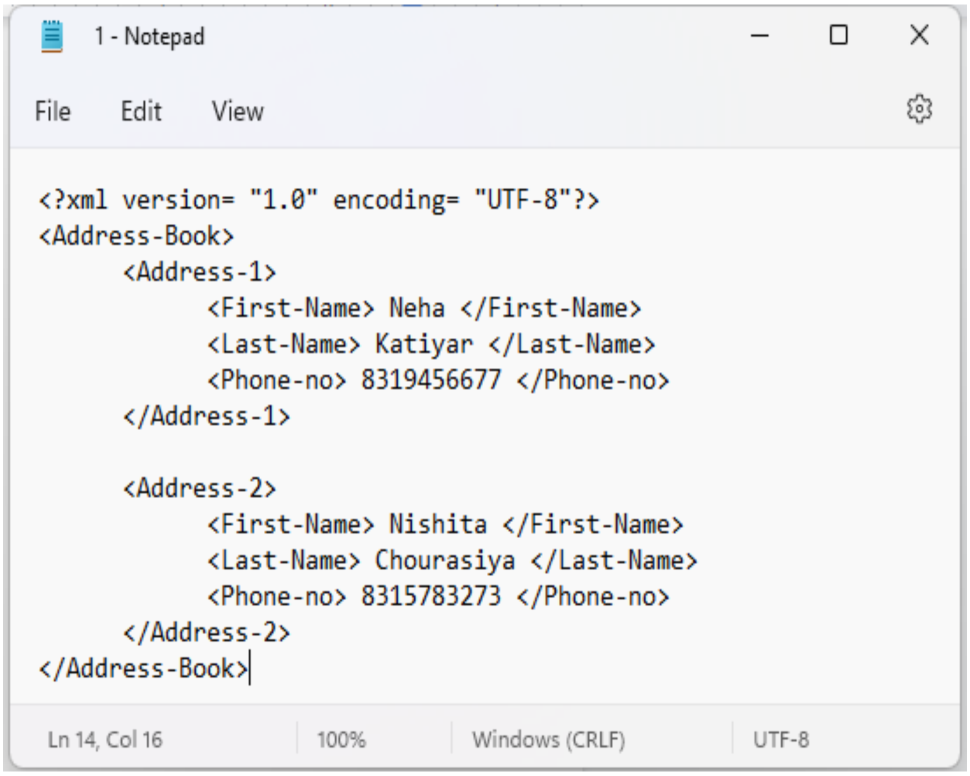
PROCEDURE:

STEP 1: Gather content items to make an Address Book to create an XML Document
STEP 2: Use Text Editor (Notepad) to create the XML data structure
STEP 3: Write prologue at the top of the page
STEP 4: Create the root element (Address-Book) and child elements (Address-1, Address-2)
STEP 5: Save file with '.xml' extension
STEP 6: Now open this file with your Browser (Edge)

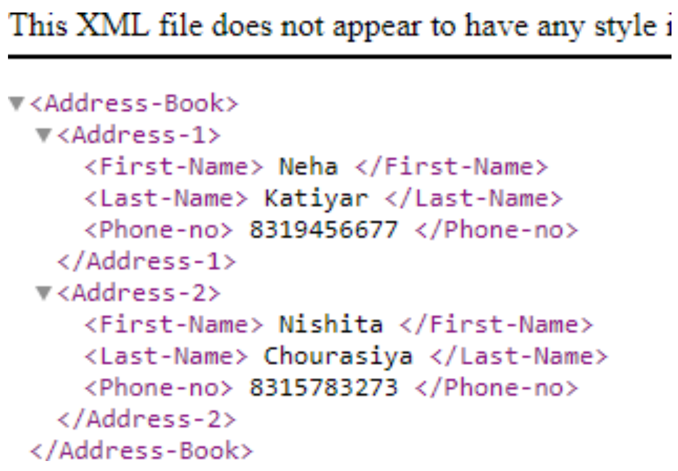
CODE:

```
<?xml version= "1.0" encoding= "UTF-8"?>
<Address-Book>
        <Address-1>
                <First-Name> Neha </First-Name>
                <Last-Name> Katiyar </Last-Name>
                <Phone-no> 8319456677 </Phone-no>
        </Address-1>

        <Address-2>
                <First-Name> Nishita </First-Name>
                <Last-Name> Chourasiya </Last-Name>
                <Phone-no> 8315783273 </Phone-no>
        </Address-2>
</Address-Book>
```

OUTPUT:



```
<?xml version= "1.0" encoding= "UTF-8"?>
<Address-Book>
      <Address-1>
            <First-Name> Neha </First-Name>
            <Last-Name> Katiyar </Last-Name>
            <Phone-no> 8319456677 </Phone-no>
      </Address-1>

      <Address-2>
            <First-Name> Nishita </First-Name>
            <Last-Name> Chourasiya </Last-Name>
            <Phone-no> 8315783273 </Phone-no>
      </Address-2>
</Address-Book>
```

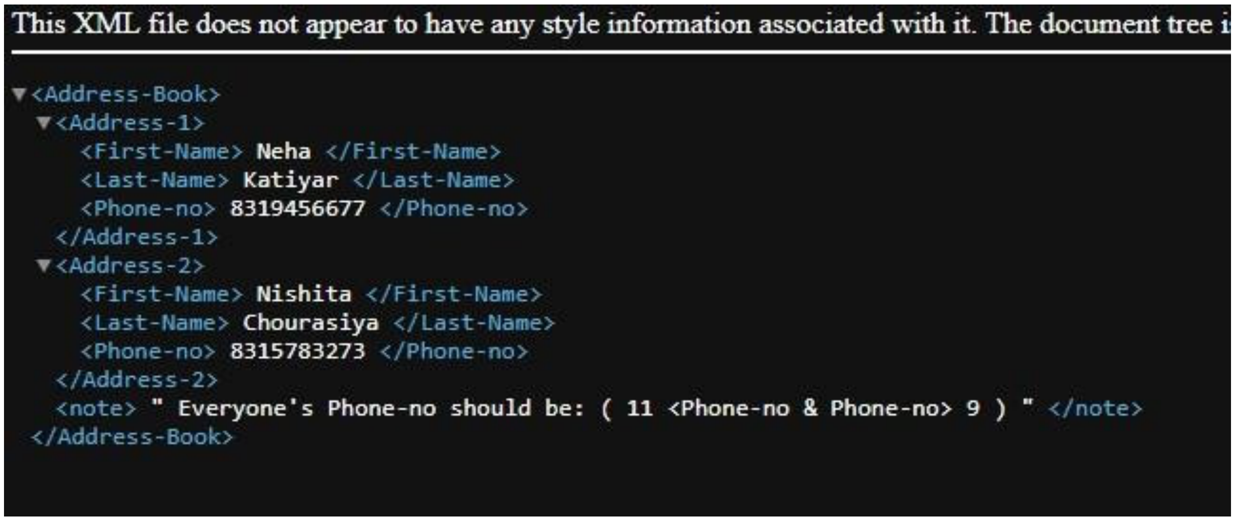


This XML file does not appear to have any style i

```
▼<Address-Book>
  ▼<Address-1>
     <First-Name> Neha </First-Name>
     <Last-Name> Katiyar </Last-Name>
     <Phone-no> 8319456677 </Phone-no>
   </Address-1>
  ▼<Address-2>
     <First-Name> Nishita </First-Name>
     <Last-Name> Chourasiya </Last-Name>
     <Phone-no> 8315783273 </Phone-no>
   </Address-2>
 </Address-Book>
```

Code with escape characters:

```
<?xml version= "1.0" encoding= "UTF-8"?>
<Address-Book>
        <Address-1>
                <First-Name> Neha </First-Name>
                <Last-Name> Katiyar </Last-Name>
                <Phone-no> 8319456677 </Phone-no>
        </Address-1>
        <Address-2>
                <First-Name> Nishita </First-Name>
                <Last-Name> Chourasiya </Last-Name>
                <Phone-no> 8315783273 </Phone-no>
        </Address-2>

        <note> &quot; Everyone&apos;s Phone-no should be: ( 11 &lt;Phone-no &amp;
Phone-no&gt; 9 ) &quot; </note>
</Address-Book>
```



This XML file does not appear to have any style information associated with it. The document tree i

```
▼<Address-Book>
  ▼<Address-1>
      <First-Name> Neha </First-Name>
      <Last-Name> Katiyar </Last-Name>
      <Phone-no> 8319456677 </Phone-no>
    </Address-1>
  ▼<Address-2>
      <First-Name> Nishita </First-Name>
      <Last-Name> Chourasiya </Last-Name>
      <Phone-no> 8315783273 </Phone-no>
    </Address-2>
    <note> " Everyone's Phone-no should be: ( 11 <Phone-no & Phone-no> 9 ) " </note>
  </Address-Book>
```

Practical 2

AIM: To create an internal DTD

PROCEDURE:

STEP 1: Gather content items to make an Address Book to create an XML Document
STEP 2: Use Text Editor (Notepad) to create the XML data structure
STEP 3: Write prologue at the top of the page
STEP 4: Now create an internal DTD as follows

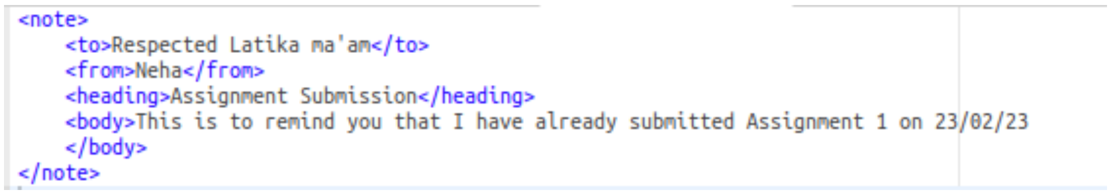
- **!DOCTYPE note** defines that the root element of this document is note
- **!ELEMENT note** defines that the note element must contain four elements: "to,from,heading,body"
- **!ELEMENT to** defines the to element to be of type "#PCDATA"
- **!ELEMENT from** defines the from element to be of type "#PCDATA"
- **!ELEMENT heading** defines the heading element to be of type "#PCDATA"
- **!ELEMENT body** defines the body element to be of type "#PCDATA"

STEP 5: Save file with '.xml' extension
STEP 6: Now open this file with your Browser (Edge)

CODE:

```
 <?xml version="1.0"?>
<!DOCTYPE note [
<!ELEMENT note (to,from,heading,body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
]>
<note>
<to>Respected Latika ma'am</to>
<from>Neha</from>
<heading>Assignment Submission</heading>
<body>This is to remind you that I have already submitted Assignment 1 on 23/02/23</body>
</note>
```

OUTPUT:



```
<note>
    <to>Respected Latika ma'am</to>
    <from>Neha</from>
    <heading>Assignment Submission</heading>
    <body>This is to remind you that I have already submitted Assignment 1 on 23/02/23
    </body>
</note>
```

Practical 3

AIM: To create an external DTD

PROCEDURE:

STEP 1: Gather content items to create an XML Document
STEP 2: Use Text Editor (Notepad) to create the XML data structure
STEP 3: Write prologue at the top of the page
STEP 4: Now create an external DTD as follows:

- **!DOCTYPE note** defines that the root element of this document is note
- **!ELEMENT note** defines that the note element must contain four elements: "to,from,heading,body"
- **!ELEMENT to** defines the to element to be of type "#PCDATA"
- **!ELEMENT from** defines the from element to be of type "#PCDATA"
- **!ELEMENT heading** defines the heading element to be of type "#PCDATA"
- **!ELEMENT body** defines the body element to be of type "#PCDATA"

STEP 5: Save file with '.dtd' extension
STEP 6: Now include external DTD file in xml file as: <!DOCTYPE note SYSTEM "note.dtd">
STEP 7: Save notepad file with '.xml' extension
STEP 8: Now open this file with your Browser (Edge)

CODE:

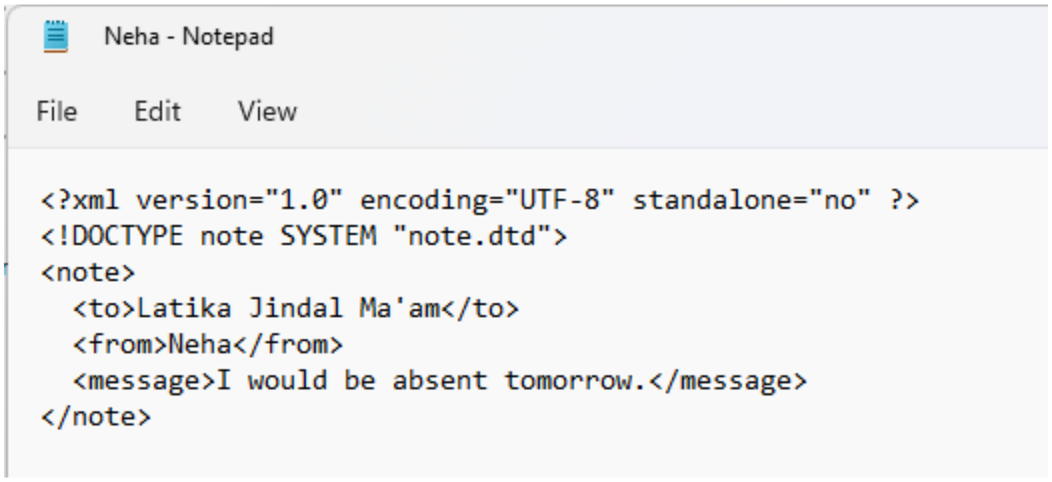
Notepad File:

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<!DOCTYPE note SYSTEM "note.dtd">
<note>
  <to>Latika Jindal Ma'am</to>
  <from>Neha</from>
  <message>I would be absent tomorrow.</message>
</note>
```

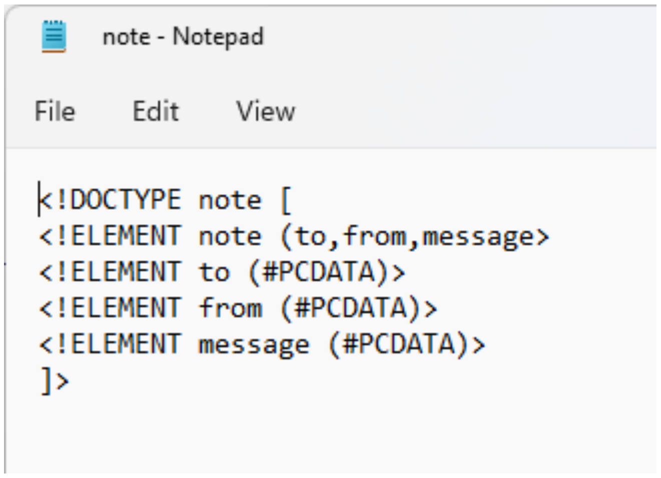
External DTD File:

```
<!DOCTYPE note [
<!ELEMENT note (to,from,message>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT message (#PCDATA)>
]>
```

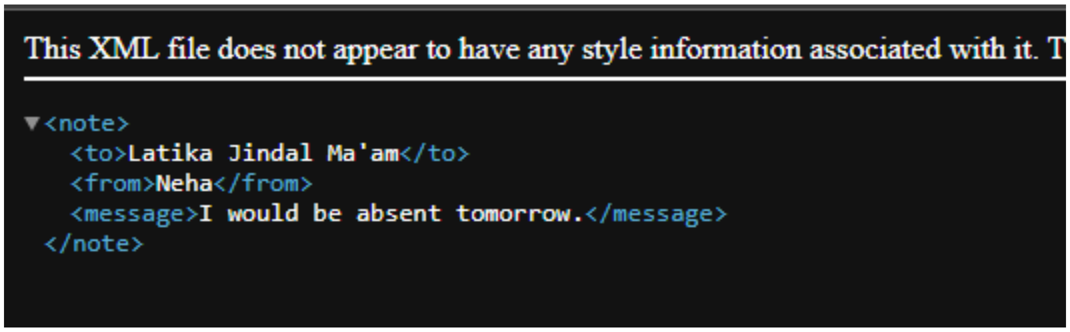
OUTPUT:



```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<!DOCTYPE note SYSTEM "note.dtd">
<note>
  <to>Latika Jindal Ma'am</to>
  <from>Neha</from>
  <message>I would be absent tomorrow.</message>
</note>
```



```
<!DOCTYPE note [
<!ELEMENT note (to,from,message>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT message (#PCDATA)>
]>
```



```
This XML file does not appear to have any style information associated with it. T
▼<note>
   <to>Latika Jindal Ma'am</to>
   <from>Neha</from>
   <message>I would be absent tomorrow.</message>
 </note>
```

Practical 4

AIM: To create an XML Schema creation and display elements and attributes

PROCEDURE:

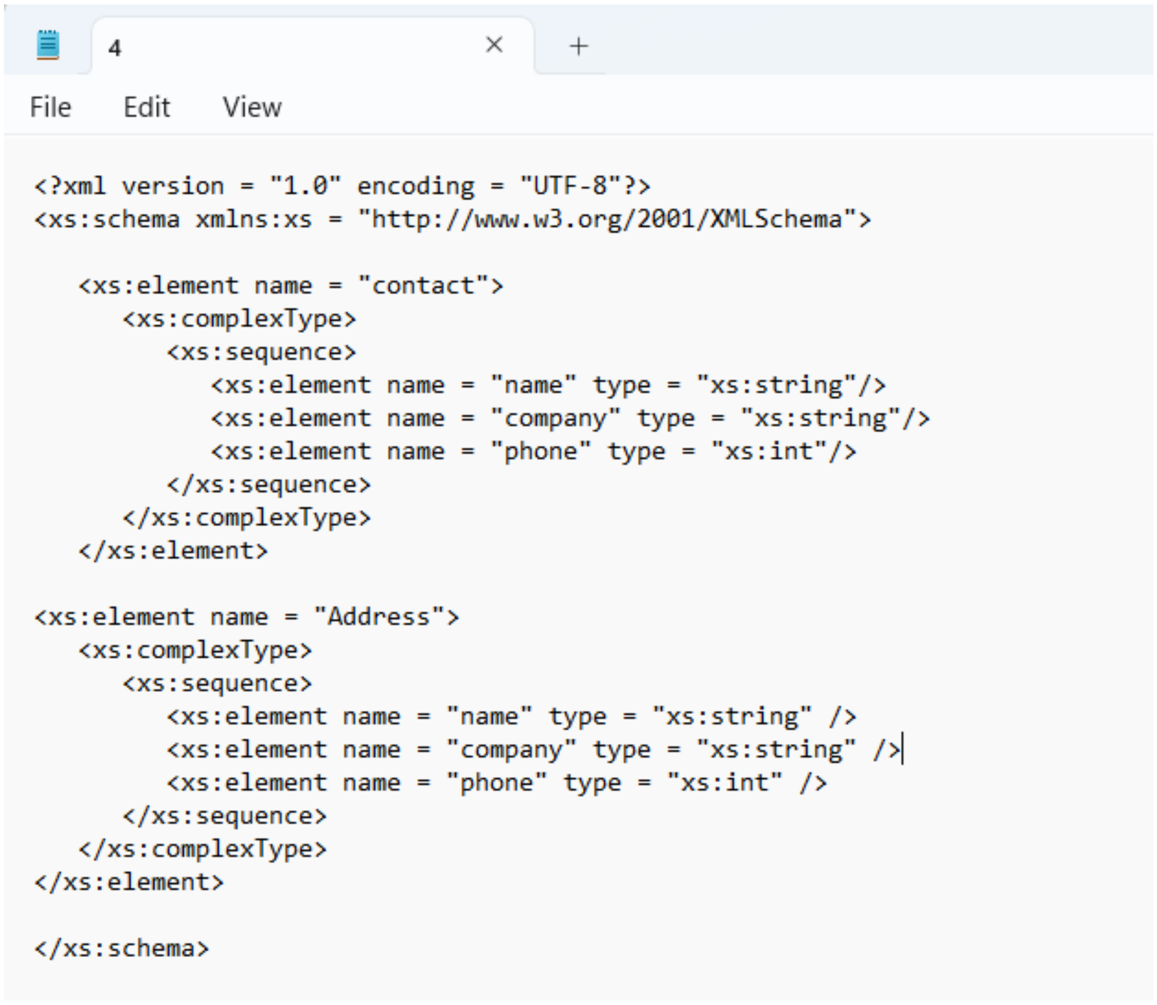
STEP 1: Gather content items to create an XML Document
STEP 2: Use Text Editor (Notepad) to create the XML data structure
STEP 3: Write prologue at the top of the page
STEP 4: Now, " xs:schema xmlns:xs = "http://www.w3.org/2001/XMLSchema" " tells the XML parser that this document should be validated against a schema
STEP 5: The line: " xmlns:xs = "http://www.w3.org/2001/XMLSchema" " specifies where the schema resides
STEP 6 In the schema above we use the standard namespace (xs), and the URI associated with this namespace is the Schema language definition, which has the standard value of http://www.w3.org/2001/XMLSchema.
STEP 7: Now we define an element "contact" and it contains other elements and attributes, therefore we consider it as a complex type
STEP 8:The child elements of the "name", "company", "phone" element is surrounded by a xs:sequence element that defines an ordered sequence of sub elements
STEP 9:Then we define the "name" element as a simple type, with type (xs:string) prefixed with the namespace prefix associated with XML Schema
STEP 10: Similarly we define other complex and simple data types in the schema
STEP 11: Now open this file with your Browser (Edge)

CODE:

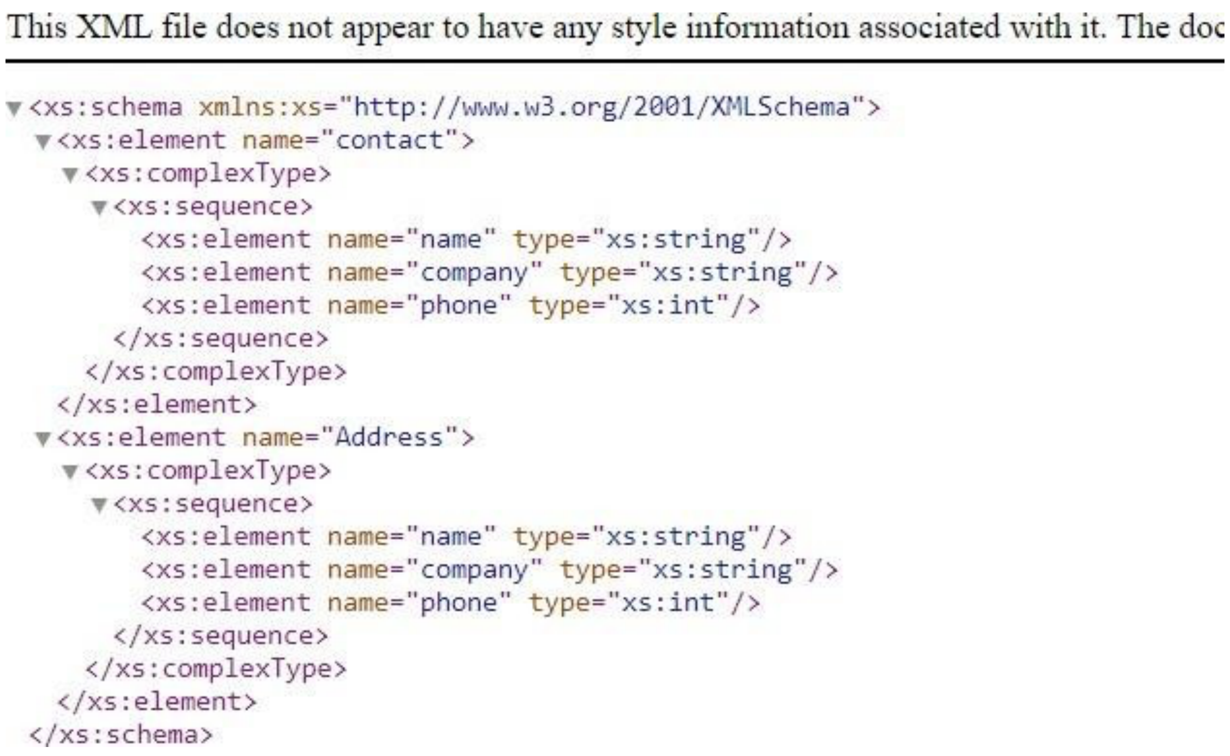
```
<?xml version = "1.0" encoding = "UTF-8"?>
<xs:schema xmlns:xs = "http://www.w3.org/2001/XMLSchema">

  <xs:element name = "contact">
    <xs:complexType>
      <xs:sequence>
        <xs:element name = "name" type = "xs:string"/>
        <xs:element name = "company" type = "xs:string"/>
        <xs:element name = "phone" type = "xs:int"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

<xs:element name = "Address">
```

```
    <xs:complexType>
      <xs:sequence>
        <xs:element name = "name" type = "xs:string" />
        <xs:element name = "company" type = "xs:string" />
        <xs:element name = "phone" type = "xs:int" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

OUTPUT:



```
<?xml version = "1.0" encoding = "UTF-8"?>
<xs:schema xmlns:xs = "http://www.w3.org/2001/XMLSchema">

   <xs:element name = "contact">
      <xs:complexType>
         <xs:sequence>
            <xs:element name = "name" type = "xs:string"/>
            <xs:element name = "company" type = "xs:string"/>
            <xs:element name = "phone" type = "xs:int"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>

<xs:element name = "Address">
   <xs:complexType>
      <xs:sequence>
         <xs:element name = "name" type = "xs:string" />
         <xs:element name = "company" type = "xs:string" />
         <xs:element name = "phone" type = "xs:int" />
      </xs:sequence>
   </xs:complexType>
</xs:element>

</xs:schema>
```



This XML file does not appear to have any style information associated with it. The doc

```
▼<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  ▼<xs:element name="contact">
    ▼<xs:complexType>
      ▼<xs:sequence>
          <xs:element name="name" type="xs:string"/>
          <xs:element name="company" type="xs:string"/>
          <xs:element name="phone" type="xs:int"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  ▼<xs:element name="Address">
    ▼<xs:complexType>
      ▼<xs:sequence>
          <xs:element name="name" type="xs:string"/>
          <xs:element name="company" type="xs:string"/>
          <xs:element name="phone" type="xs:int"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:schema>
```

Practical 5

AIM: To create an HTML table for XML file

PROCEDURE:

STEP 1: Gather content items to create an XML Document
STEP 2: Use Text Editor (Notepad) to create the XML data structure
STEP 3: Write prologue at the top of the page
STEP 4: Now create an XSL stylesheet.
STEP 5: Save file with '.xsl' extension
STEP 6: To bind the XML elements to a HTML table, the <for-each> XSL template must appear before each table row tag. This ensures that a new row is created for each <Address> element. The <value-of> template will output the selected text of each child element into a separate table.
STEP 7: Save notepad file with '.xml' extension
STEP 8: Now open this file with your Browser (Edge)

CODE:

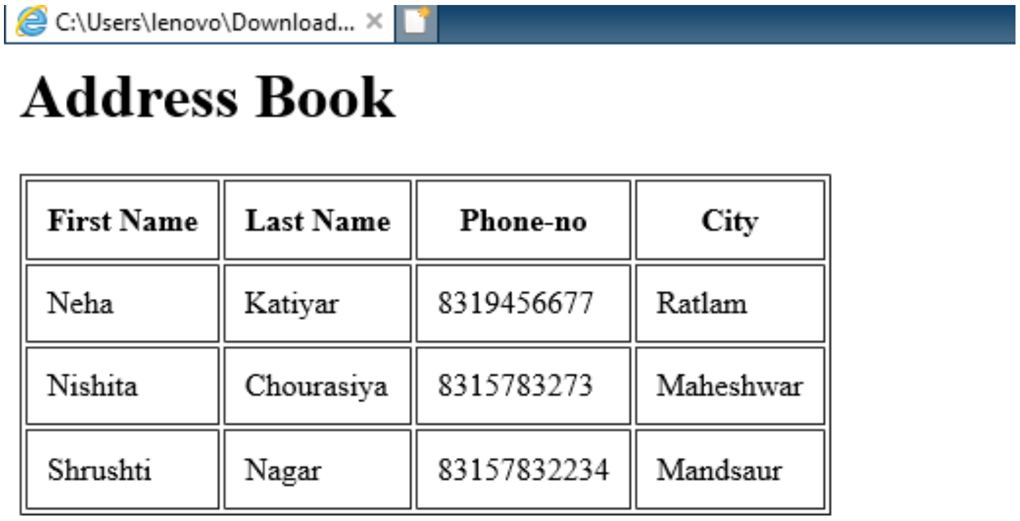
XML File:

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/xsl" href="5.xsl"?>
<Address-Book>
        <Address id="a101">
                <First-Name> Neha </First-Name>
                <Last-Name> Katiyar </Last-Name>
                <Phone-no> 8319456677 </Phone-no>
                <City> Ratlam </City>
        </Address>

        <Address id="a102">
                <First-Name> Nishita </First-Name>
                <Last-Name> Chourasiya </Last-Name>
                <Phone-no> 8315783273 </Phone-no>
                <City> Maheshwar </City>
        </Address>

        <Address id="a103">
                <First-Name> Shrushti </First-Name>
                <Last-Name> Nagar </Last-Name>
                <Phone-no> 83157832234 </Phone-no>
                <City> Mandsaur </City>
        </Address>

</Address-Book>
```

XSL File:

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
      xmlns:xsl = "http://www.w3.org/1999/XSL/Transform"
      xmlns = "http://www.w3.org/1999/xhtml">
<xsl:template match="Address-Book">

        <html>
                <body>
                        <h1>Address Book</h1>
                        <table border="1" cellpadding="10">
                        <tr>
                                        <th>First Name</th>
                                        <th>Last Name</th>
                                        <th>Phone-no</th>
                                        <th>City</th>
                        </tr>

                        <xsl:for-each select="Address">
                        <tr>
                                <td><xsl:value-of select="First-Name"/></td>
                                <td><xsl:value-of select="Last-Name" /></td>
                                <td><xsl:value-of select="Phone-no" /></td>
                                <td><xsl:value-of select="City" /></td>
                        </tr>

                        </xsl:for-each>
                        </table>
                </body>
        </html>
</xsl:template>
</xsl:stylesheet>
```

OUTPUT:



Address Book

First Name	Last Name	Phone-no	City
Neha	Katiyar	8319456677	Ratlam
Nishita	Chourasiya	8315783273	Maheshwar
Shrushti	Nagar	83157832234	Mandsaur

Practical 6

AIM: To create a simple XSLT transformation from XSL to XML

PROCEDURE:

STEP 1: Gather content items to create an XML Document
STEP 2: Use Text Editor (Notepad) to create the XML data structure
STEP 3: Write prologue at the top of the page
STEP 4: Students.xml is created and linking it with Rule.xsl which contains the corresponding XSL style sheet rules
STEP 5: Add the XSL style sheet reference to your XML document
STEP 6:Save XSLT file with '.xsl' extension
STEP 7: Save XML notepad file with '.xml' extension
STEP 8: Now open this file with your Browser (Edge)

CODE:

XML File:

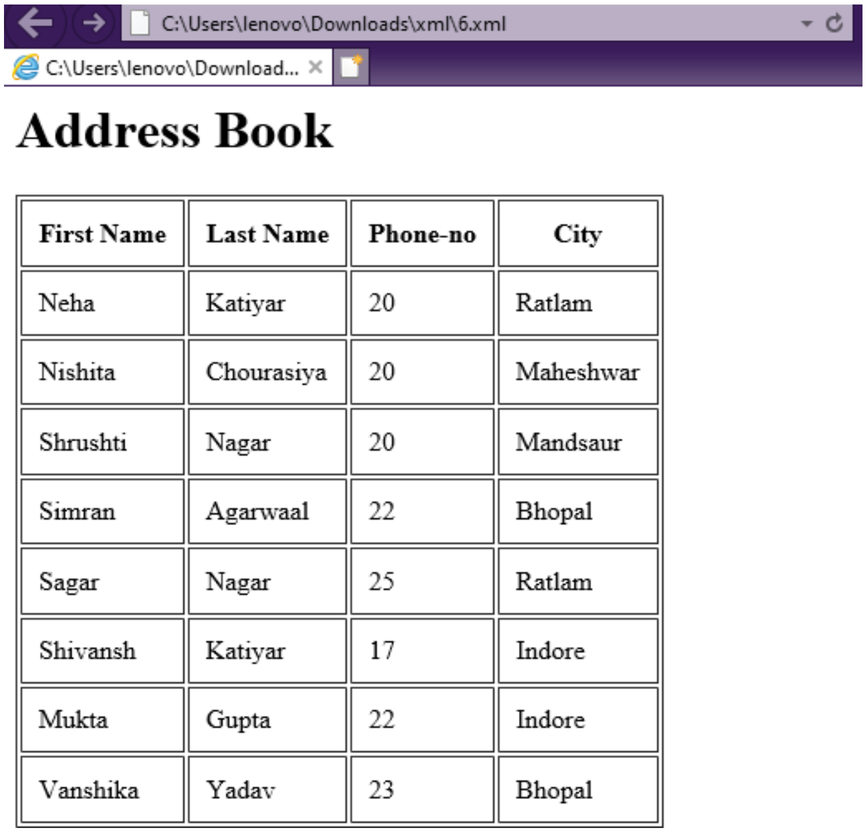
```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/xsl" href="5.xsl"?>
<Address-Book>
        <Address id="a101">
                <First-Name> Neha </First-Name>
                <Last-Name> Katiyar </Last-Name>
                <City> Ratlam </City>
                <Phone-no> 20 </Phone-no>
        </Address>
        <Address id="a102">
                <First-Name> Nishita </First-Name>
                <Last-Name> Chourasiya </Last-Name>
                <City> Maheshwar </City>
                <Phone-no> 20 </Phone-no>
        </Address>
        <Address id="a103">
                <First-Name> Shrushti </First-Name>
                <Last-Name> Nagar </Last-Name>
                <City> Mandsaur </City>
                <Phone-no> 20 </Phone-no>
        </Address>
        <Address id="a104">
                <First-Name> Simran </First-Name>
                <Last-Name> Agarwaal </Last-Name>
                <City> Bhopal </City>
                <Phone-no> 22 </Phone-no>
        </Address>
        <Address id="a105">
                <First-Name> Sagar </First-Name>
                <Last-Name> Nagar </Last-Name>
                <City> Ratlam </City>
                <Phone-no> 25 </Phone-no>
        </Address>
        <Address id="a106">
                <First-Name> Shivansh </First-Name>
                <Last-Name> Katiyar </Last-Name>
                <City> Indore </City>
                <Phone-no> 17 </Phone-no>
        </Address>
        <Address id="a107">
                <First-Name> Mukta </First-Name>
                <Last-Name> Gupta </Last-Name>
                <City> Indore </City>
                <Phone-no> 22 </Phone-no>
        </Address>
        <Address id="a108">
                <First-Name> Vanshika </First-Name>
                <Last-Name> Yadav </Last-Name>
                <City> Bhopal </City>
                <Phone-no> 23 </Phone-no>
        </Address>
</Address-Book>
```

XSL File:

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
     xmlns:xsl = "http://www.w3.org/1999/XSL/Transform"
     xmlns = "http://www.w3.org/1999/xhtml">
<xsl:template match="Address-Book">

        <html>
                <body>
                        <h1>Address Book</h1>
                        <table border="1" cellpadding="10">
                        <tr>
                                        <th>First Name</th>
                                        <th>Last Name</th>
```

```
                    <th>City</th>
                    <th>Phone-no</th>
            </tr>

            <xsl:for-each select="Address">
            <tr>
                <td><xsl:value-of select="First-Name"/></td>
                <td><xsl:value-of select="Last-Name" /></td>
                <td><xsl:value-of select="City" /></td>
                <td><xsl:value-of select="Phone-no" /></td>
            </tr>
            </xsl:for-each>
            </table>
        </body>
    </html>
</xsl:template>
</xsl:stylesheet>
```

OUTPUT:



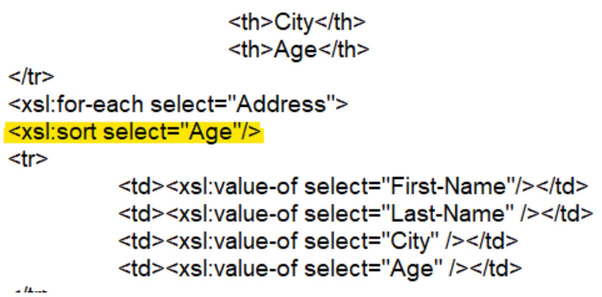
C:\Users\lenovo\Downloads\xml\6.xml

Address Book

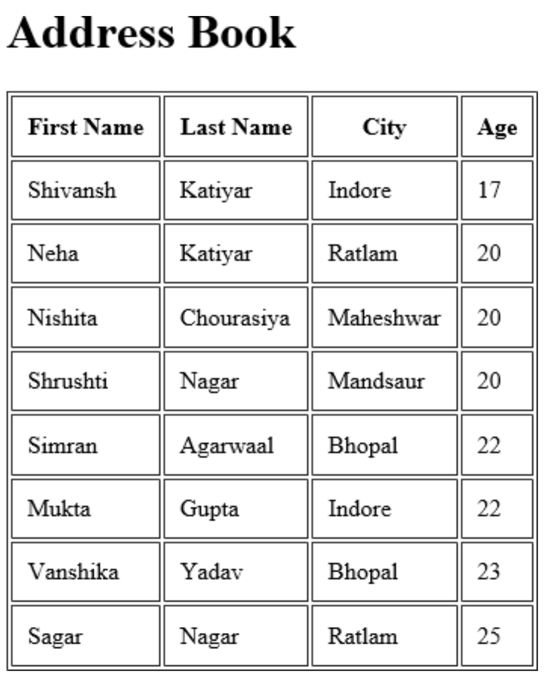
First Name	Last Name	Phone-no	City
Neha	Katiyar	20	Ratlam
Nishita	Chourasiya	20	Maheshwar
Shrushti	Nagar	20	Mandsaur
Simran	Agarwaal	22	Bhopal
Sagar	Nagar	25	Ratlam
Shivansh	Katiyar	17	Indore
Mukta	Gupta	22	Indore
Vanshika	Yadav	23	Bhopal

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

MEDICAPS UNIVERSITY	MEDICAPS UNIVERSITY, INDORE	YEAR: 2025-2026
	LAB- MANUAL	SEM: EVEN



```
                         <th>City</th>
                         <th>Age</th>
</tr>
<xsl:for-each select="Address">
<xsl:sort select="Age"/>
<tr>
          <td><xsl:value-of select="First-Name"/></td>
          <td><xsl:value-of select="Last-Name" /></td>
          <td><xsl:value-of select="City" /></td>
          <td><xsl:value-of select="Age" /></td>
```



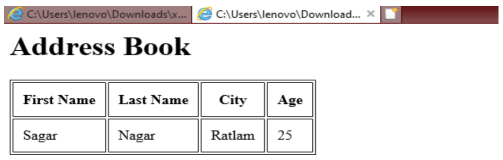
Address Book

First Name	Last Name	City	Age
Shivansh	Katiyar	Indore	17
Neha	Katiyar	Ratlam	20
Nishita	Chourasiya	Maheshwar	20
Shrushti	Nagar	Mandsaur	20
Simran	Agarwaal	Bhopal	22
Mukta	Gupta	Indore	22
Vanshika	Yadav	Bhopal	23
Sagar	Nagar	Ratlam	25

After using <if> element:

The <xsl:if> element is used to put a conditional test against the content of the XML file.

```
</tr>
<xsl:for-each select="Address">
<xsl:sort select="Age"/>
<xsl:if test="Age &gt; 23">
<tr>
          <td><xsl:value-of select="First-Name"/></td>
          <td><xsl:value-of select="Last-Name" /></td>
          <td><xsl:value-of select="City" /></td>
          <td><xsl:value-of select="Age" /></td>
</tr>
</xsl:if>
</xsl:for-each>
</table>
```



Address Book

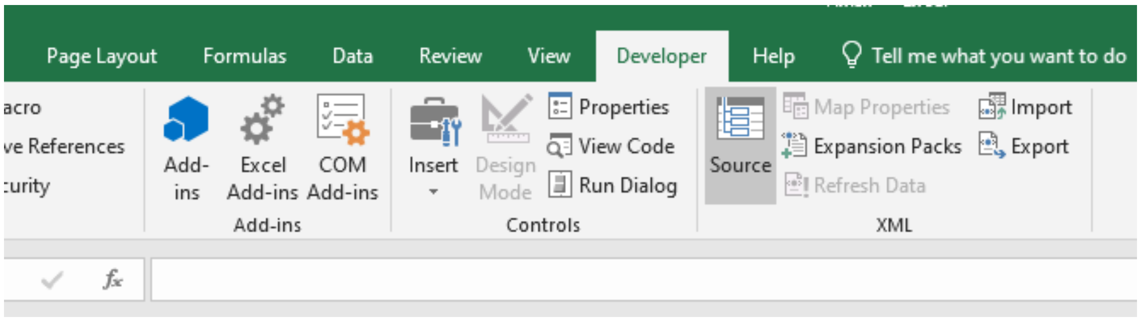
First Name	Last Name	City	Age
Sagar	Nagar	Ratlam	25

Practical 7

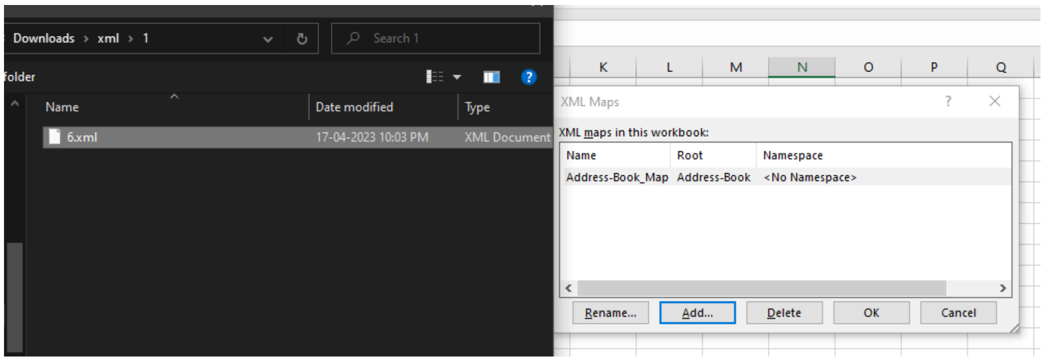
AIM: To create a xml document and database for importing xml document into database using (Import-to-import XML)

PROCEDURE:

STEP 1: Open Excel.
STEP 2: Click on the **Developer** > **Import**.



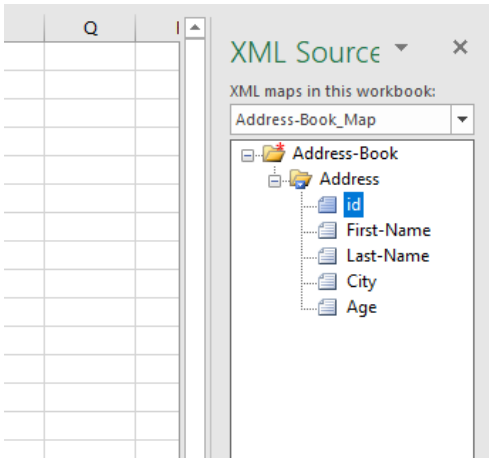
STEP 3: In the **Import XML** dialog box, locate and select the XML data file (.xml) you want to import, and click **Import**.



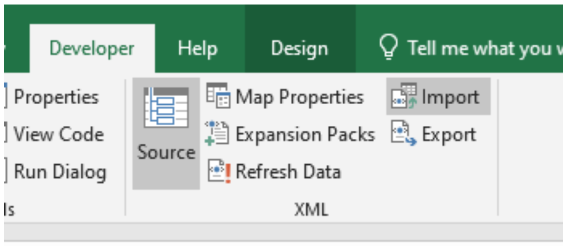
Name	Date modified	Type
6.xml	17-04-2023 10:03 PM	XML Document

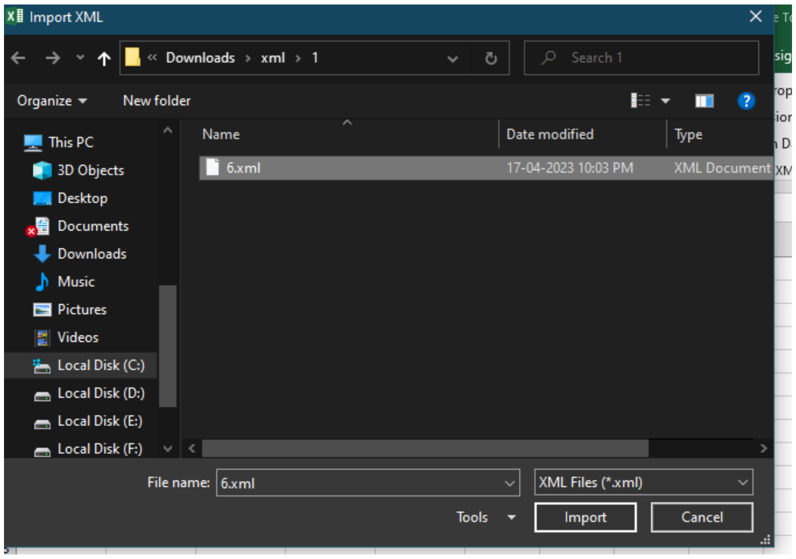
XML maps in this workbook:

Name	Root	Namespace
Address-Book_Map	Address-Book	<No Namespace>



STEP 4: Click on Import-to-import XML file.

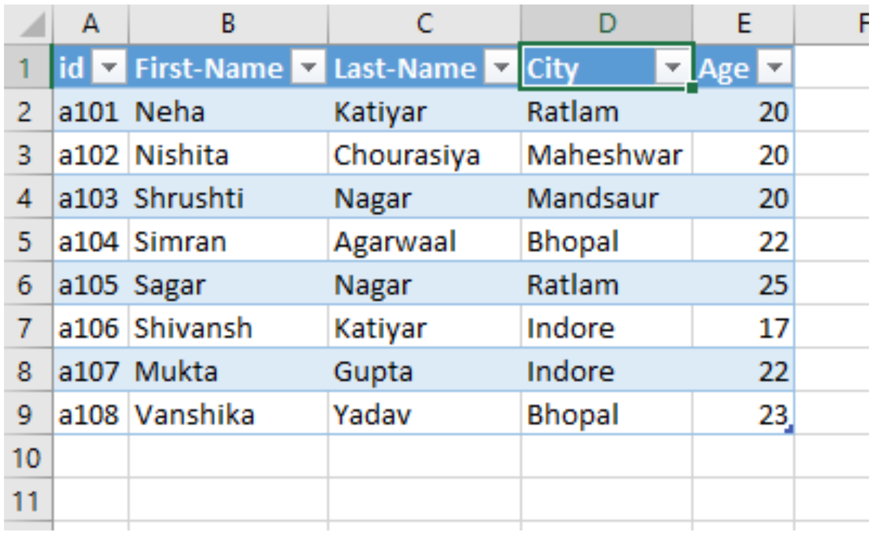




Name	Date modified	Type
6.xml	17-04-2023 10:03 PM	XML Document

File name: 6.xml
XML Files (*.xml)

OUTPUT:



id	First-Name	Last-Name	City	Age
a101	Neha	Katiyar	Ratlam	20
a102	Nishita	Chourasiya	Maheshwar	20
a103	Shrushti	Nagar	Mandsaur	20
a104	Simran	Agarwaal	Bhopal	22
a105	Sagar	Nagar	Ratlam	25
a106	Shivansh	Katiyar	Indore	17
a107	Mukta	Gupta	Indore	22
a108	Vanshika	Yadav	Bhopal	23

Practical 8

AIM: To create a parsing XML document using DOM(Document Object Model) parser. Store the information of students in XML file, validate it using XML schema and display the information of students in HTML using XSLT with proper formatting and conditions

PROCEDURE:

items to create an XML Document
STEP 2: Use Text Editor (Notepad) to create the XML data structure
STEP 3: Write prologue at the top of the page
STEP 4: Students.xml is created and linking it with Student.xsl which contains the corresponding XSL style sheet rules
STEP 5: Create an XSD Document, define complex and simple data types in the schema validate it against the Student.xml document
STEP 6: Add the XSL style sheet reference to your XML document
STEP 7:Save XSLT file with '.xsl' extension
STEP 8: Save XML notepad file with '.xml' extension
STEP 9: Create the XML DOM and access any node using getElementsByTagName() which returns all elements with a specified tag name. Validate using online parse tree
STEP 10: Now open this file with your Browser (Edge)

CODE:

Student.html:

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
<script>
var xhttp = new XMLHttpRequest:
xhttp.onreadystatechange = function() {
if (this.readyState == 4 && this.status==200){
myFunction(this):
}
};
xhttp.open(" GET", “demo.xml", true);
xhttp.send
function myFunction(xml) {
var xmlDoc=Xml.responseXML;
document.getElementById("Student").inner HTML =
xmlDoc.getElementsByTagName("CGPA")[0].childNodes[0].nodeValue;
}
</script>
</body>
</html>
```

Student.xml

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/xsl" href="student.xsl"?>
<Student-Record>
        <Student id="a101">
                <Enroll> EN20CS301269 </Enroll>
                <First-Name> Neha </First-Name>
                <Last-Name> Katiyar </Last-Name>
                <City> Ratlam </City>
                <Age> 20 </Age>
                <CGPA> 9.3 </CGPA>
        </Student>
        <Student id="a102">
                <Enroll> EN20CS301277 </Enroll>
                <First-Name> Nishita </First-Name>
                <Last-Name> Chourasiya </Last-Name>
                <City> Maheshwar </City>
                <Age> 20 </Age>
                <CGPA> 8.9 </CGPA>
        </Student>
        <Student id="a103">
                <Enroll> EN20CS301345 </Enroll>
                <First-Name> Shrushti </First-Name>
                <Last-Name> Nagar </Last-Name>
                <City> Mandsaur </City>
                <Age> 20 </Age>
                <CGPA> 9.2 </CGPA>
        </Student>
        <Student id="a104">
                <Enroll> EN20CS301367 </Enroll>
                <First-Name> Simran </First-Name>
                <Last-Name> Agarwaal </Last-Name>
                <City> Bhopal </City>
                <Age> 22 </Age>
                <CGPA> 5.0 </CGPA>
        </Student>
        <Student id="a105">
                <Enroll> EN20CS301678 </Enroll>
                <First-Name> Sagar </First-Name>
                <Last-Name> Nagar </Last-Name>
                <City> Ratlam </City>
                <Age> 25 </Age>
                <CGPA> 7.2 </CGPA>
        </Student>
        <Student id="a106">
                <Enroll> EN20CS301456 </Enroll>
                <First-Name> Shivansh </First-Name>
                <Last-Name> Katiyar </Last-Name>
                <City> Indore </City>
                <Age> 17 </Age>
                <CGPA> 9.9 </CGPA>
        </Student>
        <Student id="a107">
                <Enroll> EN20CS301287 </Enroll>
                <First-Name> Mukta </First-Name>
                <Last-Name> Gupta </Last-Name>
                <City> Indore </City>
                <Age> 22 </Age>
                <CGPA> 8.7 </CGPA>
        </Student>
        <Student id="a108">
                <Enroll> EN20CS301654 </Enroll>
                <First-Name> Vanshika </First-Name>
                <Last-Name> Yadav </Last-Name>
                <City> Bhopal </City>
                <Age> 23 </Age>
                <CGPA> 7.5 </CGPA>
        </Student>
</Student-Record>
```

Student.xsd:

```
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <xs:element name="Student-Record">
  <xs:complexType>
   <xs:sequence>
    <xs:element name="Student" maxOccurs="unbounded" minOccurs="0">
     <xs:complexType>
      <xs:sequence>
       <xs:element type="xs:string" name="Enroll"/>
       <xs:element type="xs:string" name="First-Name"/>
       <xs:element type="xs:string" name="Last-Name"/>
       <xs:element type="xs:string" name="City"/>
       <xs:element type="xs:float" name="Age"/>
       <xs:element type="xs:float" name="CGPA"/>
      </xs:sequence>
      <xs:attribute type="xs:string" name="id" use="optional"/>
     </xs:complexType>
    </xs:element>
   </xs:sequence>
  </xs:complexType>
 </xs:element>
</xs:schema>
```

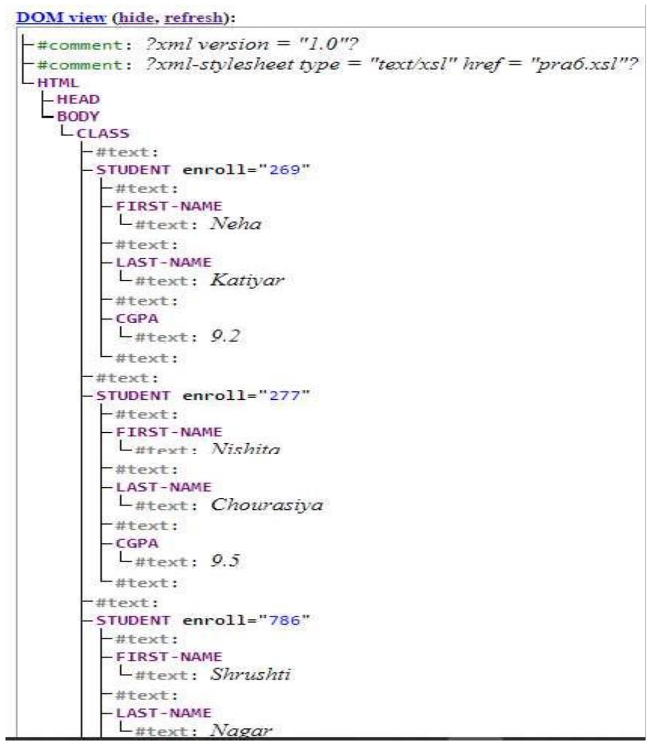
Student.xsl:

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
      xmlns:xsl = "http://www.w3.org/1999/XSL/Transform"
      xmlns = "http://www.w3.org/1999/xhtml">
<xsl:template match="Student-Record">
        <html>
                <body>
                        <h1>Students' Record</h1>
                        <table border="1" cellpadding="10">
```

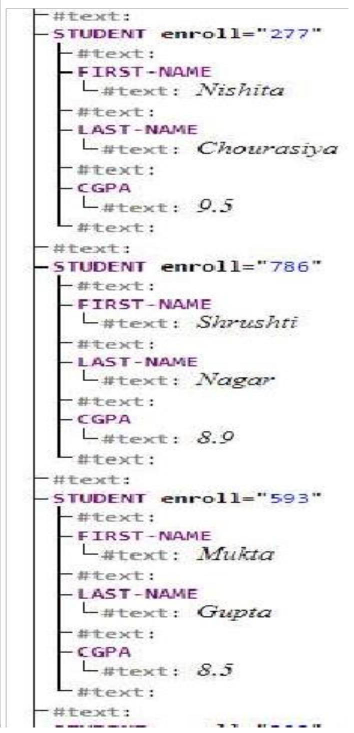
```
                    <tr>
                                <th>Enrollment No.</th>
                                <th>First Name</th>
                                <th>Last Name</th>
                                <th>City</th>
                                <th>Age</th>
                                <th>CGPA</th>
                    </tr>
                    <xsl:for-each select="Student">
                    <tr>
                         <td><xsl:value-of select="Enroll"/></td>
                         <td><xsl:value-of select="First-Name"/></td>
                         <td><xsl:value-of select="Last-Name" /></td>
                         <td><xsl:value-of select="City" /></td>
                         <td><xsl:value-of select="Age" /></td>
                         <td><xsl:value-of select="CGPA" /></td>
                    </tr>
                    </xsl:for-each>
                    </table>
               </body>
          </html>
</xsl:template>
</xsl:stylesheet>
```

OUTPUT:

XML document using DOM(Document Object Model) parser:

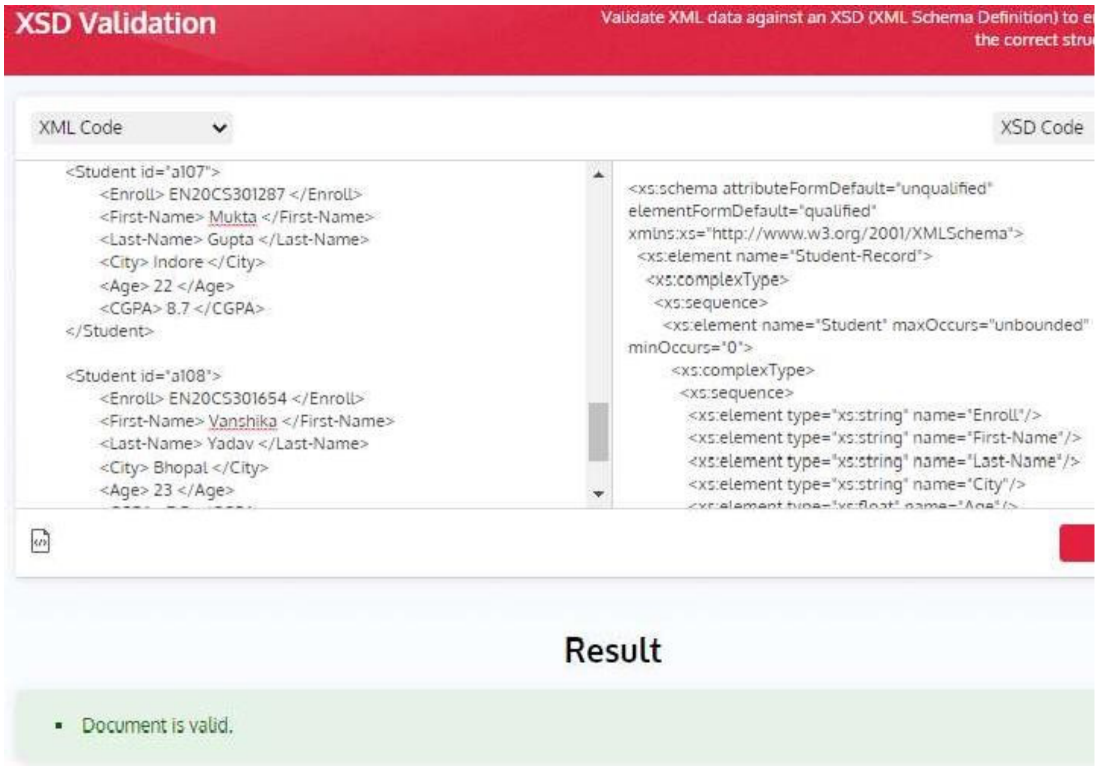


```
DOM view (hide, refresh):
├#comment: ?xml version = "1.0"?
├#comment: ?xml-stylesheet type = "text/xsl" href = "pra6.xsl"?
└HTML
  ├HEAD
  └BODY
    └CLASS
      ├#text:
      ├STUDENT enroll="269"
      │ ├#text:
      │ ├FIRST-NAME
      │ │ └#text: Neha
      │ ├#text:
      │ ├LAST-NAME
      │ │ └#text: Katiyar
      │ ├#text:
      │ ├CGPA
      │ │ └#text: 9.2
      │ └#text:
      ├#text:
      ├STUDENT enroll="277"
      │ ├#text:
      │ ├FIRST-NAME
      │ │ └#text: Nishita
      │ ├#text:
      │ ├LAST-NAME
      │ │ └#text: Chourasiya
      │ ├#text:
      │ ├CGPA
      │ │ └#text: 9.5
      │ └#text:
      ├#text:
      ├STUDENT enroll="786"
      │ ├#text:
      │ ├FIRST-NAME
      │ │ └#text: Shrushti
      │ ├#text:
      │ ├LAST-NAME
      │ │ └#text: Nagar
```



```
├#text:
├STUDENT enroll="277"
│ ├#text:
│ ├FIRST-NAME
│ │ └#text: Nishita
│ ├#text:
│ ├LAST-NAME
│ │ └#text: Chourasiya
│ ├#text:
│ ├CGPA
│ │ └#text: 9.5
│ └#text:
├#text:
├STUDENT enroll="786"
│ ├#text:
│ ├FIRST-NAME
│ │ └#text: Shrushti
│ ├#text:
│ ├LAST-NAME
│ │ └#text: Nagar
│ ├#text:
│ ├CGPA
│ │ └#text: 8.9
│ └#text:
├#text:
├STUDENT enroll="593"
│ ├#text:
│ ├FIRST-NAME
│ │ └#text: Mukta
│ ├#text:
│ ├LAST-NAME
│ │ └#text: Gupta
│ ├#text:
│ ├CGPA
│ │ └#text: 8.5
│ └#text:
├#text:
```

XSD Validation:



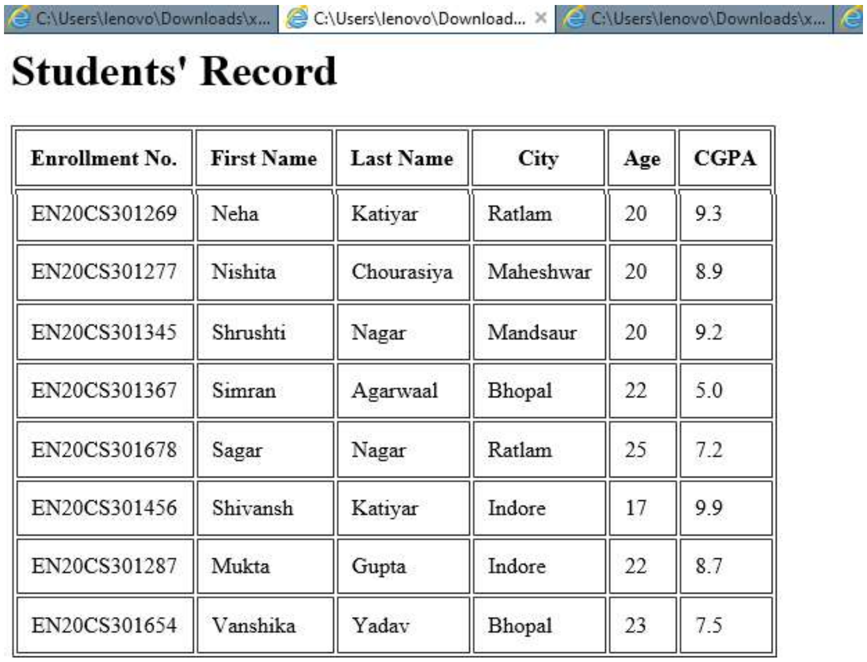
XSD Validation
Validate XML data against an XSD (XML Schema Definition) to e
the correct stru
XML Code
XSD Code
Result

```
<Student id="a107">
    <Enroll> EN20CS301287 </Enroll>
    <First-Name> Mukta </First-Name>
    <Last-Name> Gupta </Last-Name>
    <City> Indore </City>
    <Age> 22 </Age>
    <CGPA> 8.7 </CGPA>
</Student>

<Student id="a108">
    <Enroll> EN20CS301654 </Enroll>
    <First-Name> Vanshika </First-Name>
    <Last-Name> Yadav </Last-Name>
    <City> Bhopal </City>
    <Age> 23 </Age>
```

```
<xs:schema attributeFormDefault="unqualified"
elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <xs:element name="Student-Record">
  <xs:complexType>
   <xs:sequence>
    <xs:element name="Student" maxOccurs="unbounded"
minOccurs="0">
     <xs:complexType>
      <xs:sequence>
       <xs:element type="xs:string" name="Enroll"/>
       <xs:element type="xs:string" name="First-Name"/>
       <xs:element type="xs:string" name="Last-Name"/>
       <xs:element type="xs:string" name="City"/>
```

- Document is valid.

Student Information displayed using XSLT in HTML:



C:\Users\lenovo\Downloads\x...
C:\Users\lenovo\Download...
C:\Users\lenovo\Downloads\x...

Students' Record

Enrollment No.	First Name	Last Name	City	Age	CGPA
EN20CS301269	Neha	Katiyar	Ratlam	20	9.3
EN20CS301277	Nishita	Chourasiya	Maheshwar	20	8.9
EN20CS301345	Shrushti	Nagar	Mandsaur	20	9.2
EN20CS301367	Simran	Agarwaal	Bhopal	22	5.0
EN20CS301678	Sagar	Nagar	Ratlam	25	7.2
EN20CS301456	Shivansh	Katiyar	Indore	17	9.9
EN20CS301287	Mukta	Gupta	Indore	22	8.7
EN20CS301654	Vanshika	Yadav	Bhopal	23	7.5

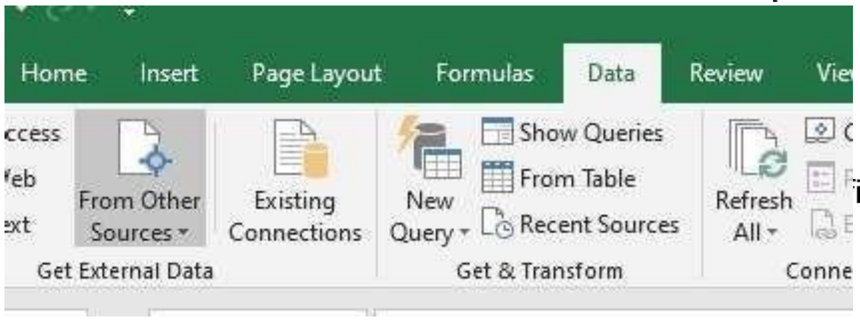
Practical 9

AIM: To create a xml document and database for importing xml document into database using (data tab)

From XML Data

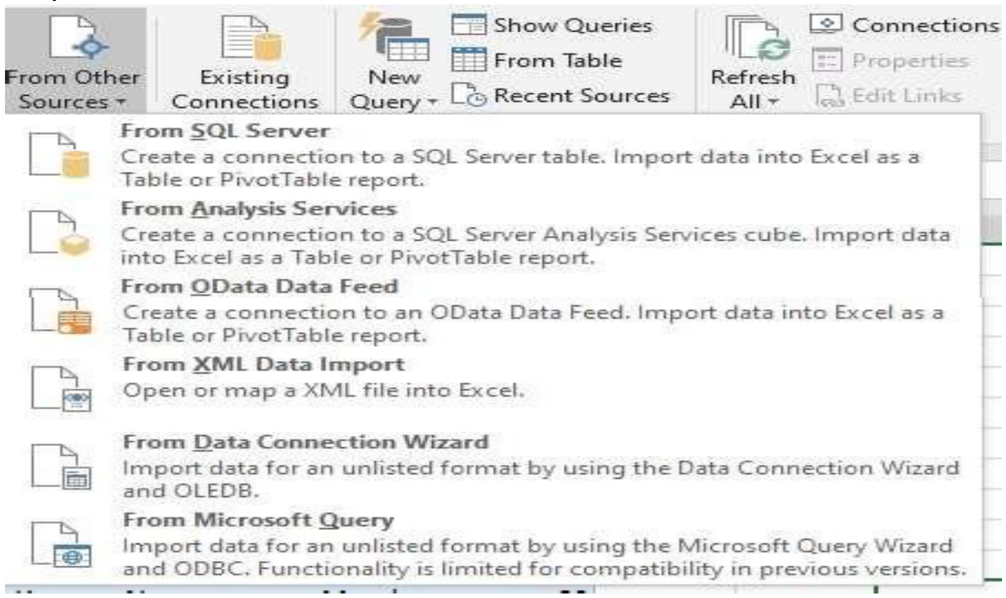
PROCEDURE:

STEP 1: Open Excel
STEP 2: Click on data tab
STEP 3: Click on **Data** > **From Other Sources** > **Import**.



ile (.xml)

STEP 4: Go to the drive, folder, or you want to import.



From SQL Server
Create a connection to a SQL Server table. Import data into Excel as a Table or PivotTable report.

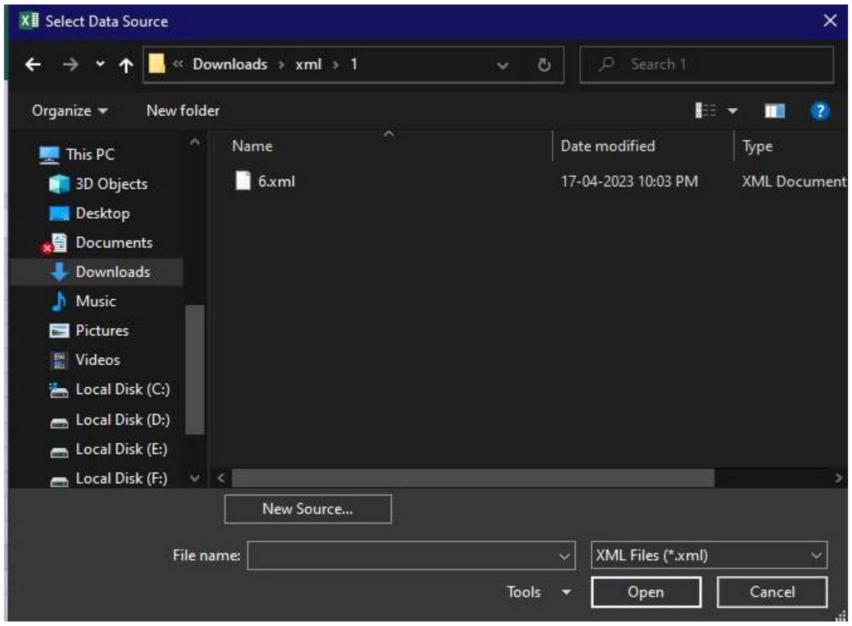
From Analysis Services
Create a connection to a SQL Server Analysis Services cube. Import data into Excel as a Table or PivotTable report.

From OData Data Feed
Create a connection to an OData Data Feed. Import data into Excel as a Table or PivotTable report.

From XML Data Import
Open or map a XML file into Excel.

From Data Connection Wizard
Import data for an unlisted format by using the Data Connection Wizard and OLEDB.

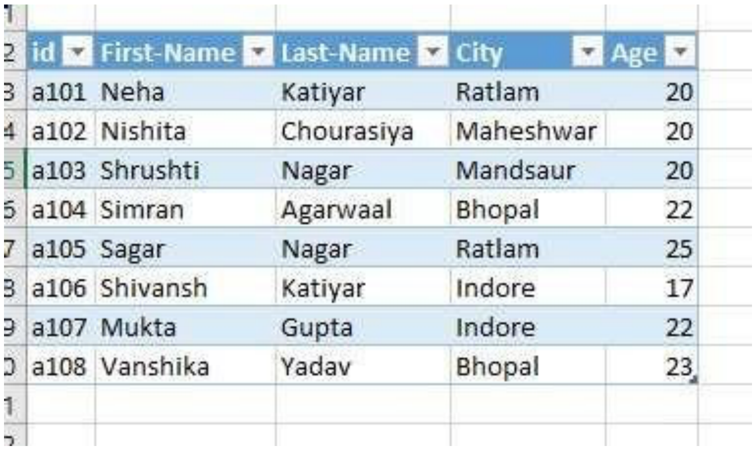
From Microsoft Query
Import data for an unlisted format by using the Microsoft Query Wizard and ODBC. Functionality is limited for compatibility in previous versions.



Name	Date modified	Type
6.xml	17-04-2023 10:03 PM	XML Document

File name:
XML Files (*.xml)

OUTPUT:



id	First-Name	Last-Name	City	Age
a101	Neha	Katiyar	Ratlam	20
a102	Nishita	Chourasiya	Maheshwar	20
a103	Shrushti	Nagar	Mandsaur	20
a104	Simran	Agarwaal	Bhopal	22
a105	Sagar	Nagar	Ratlam	25
a106	Shivansh	Katiyar	Indore	17
a107	Mukta	Gupta	Indore	22
a108	Vanshika	Yadav	Bhopal	23

Practical 10

AIM:

information of students in XML file, validate it using XML schema and display the information of students HTML using XSLT with proper formatting and conditions like having enrollment number, name start with, having CGPA between, in sorted order

PROCEDURE:

STEP 1: Gather content items to create an XML Document
STEP 2: Use Text Editor (Notepad) to create the XML data structure
STEP 3: Write prologue at the top of the page
STEP 4: Students.xml is created and linking it with Student.xsl which contains the corresponding XSL style sheet rules
STEP 5: Create an XSD Document, define complex and simple data types in the schema validate it against the Student.xml document
STEP 6: Add the XSL style sheet reference to your XML document
STEP 7:Save XSLT file with '.xsl' extension
STEP 8: Save XML notepad file with '.xml' extension
STEP 9: Use XSLT with proper formatting and conditions like having enrollment number, name start with, having CGPA between, in sorted order
STEP 10: Now open this file with your Browser (Edge)

CODE:

Student.xml

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/xsl" href="student.xsl"?>
<Student-Record>
        <Student id="a101">
                <Enroll> EN20CS301269 </Enroll>
                <First-Name> Neha </First-Name>
                <Last-Name> Katiyar </Last-Name>
                <City> Ratlam </City>
                <Age> 20 </Age>
                <CGPA> 9.3 </CGPA>
        </Student>
        <Student id="a102">
                <Enroll> EN20CS301277 </Enroll>
                <First-Name> Nishita </First-Name>
                <Last-Name> Chourasiya </Last-Name>
                <City> Maheshwar </City>
                <Age> 20 </Age>
                <CGPA> 8.9 </CGPA>
        </Student>
        <Student id="a103">
                <Enroll> EN20CS301345 </Enroll>
                <First-Name> Shrushti </First-Name>
                <Last-Name> Nagar </Last-Name>
                <City> Mandsaur </City>
                <Age> 20 </Age>
                <CGPA> 9.2 </CGPA>
        </Student>
        <Student id="a104">
                <Enroll> EN20CS301367 </Enroll>
                <First-Name> Simran </First-Name>
                <Last-Name> Agarwaal </Last-Name>
                <City> Bhopal </City>
                <Age> 22 </Age>
                <CGPA> 5.0 </CGPA>
        </Student>
        <Student id="a105">
                <Enroll> EN20CS301678 </Enroll>
                <First-Name> Sagar </First-Name>
                <Last-Name> Nagar </Last-Name>
                <City> Ratlam </City>
                <Age> 25 </Age>
                <CGPA> 7.2 </CGPA>
        </Student>
        <Student id="a106">
                <Enroll> EN20CS301456 </Enroll>
                <First-Name> Shivansh </First-Name>
                <Last-Name> Katiyar </Last-Name>
                <City> Indore </City>
                <Age> 17 </Age>
                <CGPA> 9.9 </CGPA>
        </Student>
        <Student id="a107">
                <Enroll> EN20CS301287 </Enroll>
                <First-Name> Mukta </First-Name>
                <Last-Name> Gupta </Last-Name>
                <City> Indore </City>
                <Age> 22 </Age>
                <CGPA> 8.7 </CGPA>
        </Student>
        <Student id="a108">
                <Enroll> EN20CS301654 </Enroll>
                <First-Name> Vanshika </First-Name>
                <Last-Name> Yadav </Last-Name>
                <City> Bhopal </City>
                <Age> 23 </Age>
                <CGPA> 7.5 </CGPA>
        </Student>
</Student-Record>
```

Student.xsd:

```
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="Student-Record">
    <xs:complexType>
```

```
      <xs:sequence>
        <xs:element name="Student" maxOccurs="unbounded" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element type="xs:string" name="Enroll"/>
              <xs:element type="xs:string" name="First-Name"/>
              <xs:element type="xs:string" name="Last-Name"/>
              <xs:element type="xs:string" name="City"/>
              <xs:element type="xs:float" name="Age"/>
              <xs:element type="xs:float" name="CGPA"/>
            </xs:sequence>
            <xs:attribute type="xs:string" name="id" use="optional"/>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

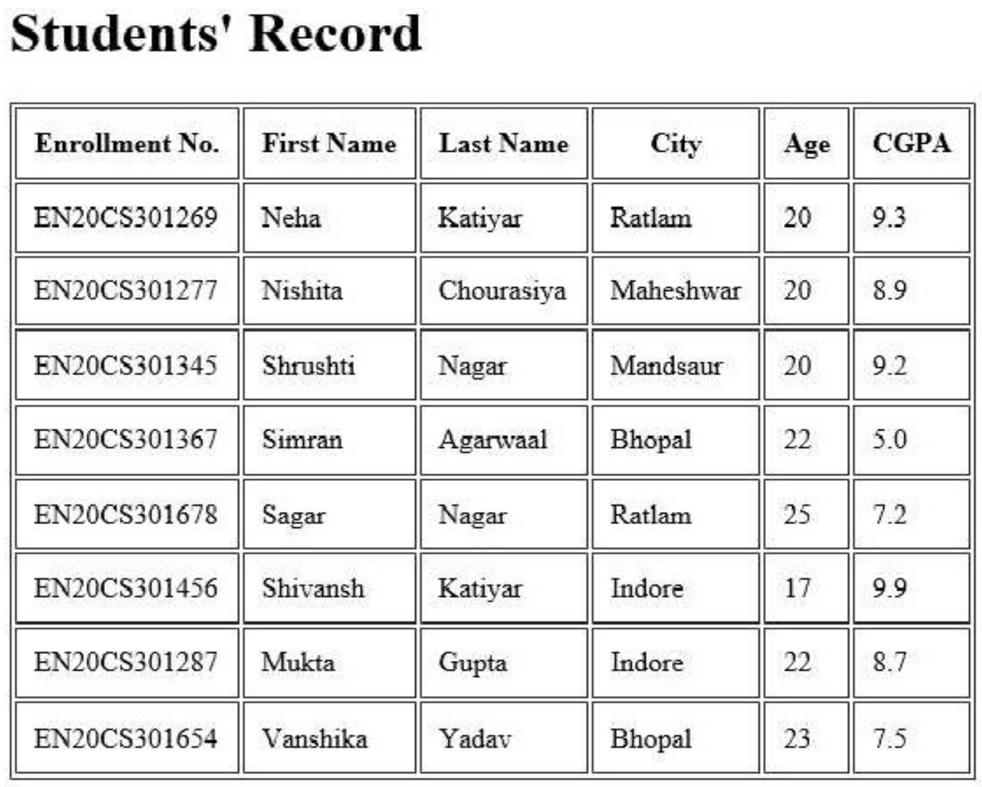
Student.xsl:

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
        xmlns:xsl = "http://www.w3.org/1999/XSL/Transform"
        xmlns = "http://www.w3.org/1999/xhtml">
<xsl:template match="Student-Record">
            <html>
                        <body>
                                    <h1>Students' Record</h1>
                                    <table border="1" cellpadding="10">
                                    <tr>
                                                            <th>Enrollment No.</th>
                                                            <th>First Name</th>
                                                            <th>Last Name</th>
                                                            <th>City</th>
                                                            <th>Age</th>
                                                            <th>CGPA</th>
                                    </tr>
<xsl:for-each select="Student[starts-with(First-Name, 'S') and CGPA &gt;= 7.0 and CGPA &lt;=
10.0]">
                                    <xsl:sort select="Enroll"/>
                                    <tr>
                                                <td><xsl:value-of select="Enroll"/></td>
                                                <td><xsl:value-of select="First-Name"/></td>
                                                <td><xsl:value-of select="Last-Name" /></td>
                                                <td><xsl:value-of select="City" /></td>
                                                <td><xsl:value-of select="Age" /></td>
                                                <td><xsl:value-of select="CGPA" /></td>
                                    </tr>
                                    </xsl:for-each>
                                    </table>
                        </body>
            </html>
```

```
</xsl:template>
</xsl:stylesheet>
```

OUTPUT:

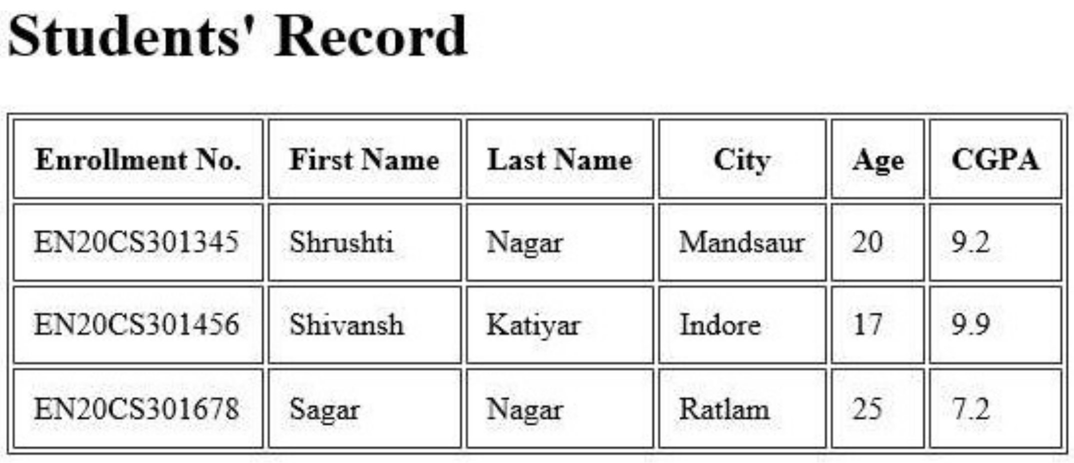
Without using any formatting:



Students' Record

Enrollment No.	First Name	Last Name	City	Age	CGPA
EN20CS301269	Neha	Katiyar	Ratlam	20	9.3
EN20CS301277	Nishita	Chourasiya	Maheshwar	20	8.9
EN20CS301345	Shrushti	Nagar	Mandsaur	20	9.2
EN20CS301367	Simran	Agarwaal	Bhopal	22	5.0
EN20CS301678	Sagar	Nagar	Ratlam	25	7.2
EN20CS301456	Shivansh	Katiyar	Indore	17	9.9
EN20CS301287	Mukta	Gupta	Indore	22	8.7
EN20CS301654	Vanshika	Yadav	Bhopal	23	7.5

After formatting that sorts the student records by enrollment number, displays only students whose first name starts with "S", and whose CGPA is between 7.0 and 10.0:



Students' Record

Enrollment No.	First Name	Last Name	City	Age	CGPA
EN20CS301345	Shrushti	Nagar	Mandsaur	20	9.2
EN20CS301456	Shivansh	Katiyar	Indore	17	9.9
EN20CS301678	Sagar	Nagar	Ratlam	25	7.2