1. Design the XML document to store the information of the employees of any business organization and demonstrate the use of:
2. a) DTD
3. b) XML Schema
4. And display the content in (e.g., tabular format) by using CSS/XSL.

What is XML?

XML is a software- and hardware-independent tool for storing and transporting data.
● XML stands for eXtensible Markup Language
● markup language much like HTML
● designed to store and transport data
● designed to be self-descriptive
 It is a dynamic markup language.
 It is used to transform data from one form to another form.

An XML file can be displayed using two ways.

1:Extensible Stylesheet Language Transformation (XSLT) is a language for transforming XML documents into other formats, such as HTML, XML, or plain text. XSLT is a key component of the Extensible Stylesheet Language (XSL),
XSLT transformations are driven by templates, which match specific elements or patterns in the input XML document. When a match is found, the corresponding template is applied to generate the output.

What can you do with XSLT:
1. Add /remove elements
2. add/remove attributes
3. rearrange /sort element

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

Convert xml to xhtml

```
<?xml version="1.0" encoding="UTF-8"?>
<catalog>
  <cd>
    <title>Empire Burlesque</title>
    <artist>Bob Dylan</artist>
    <country>USA</country>
    <company>Columbia</company>
    <price>10.90</price>
    <year>1985</year>
  </cd>
.
.
</catalog>

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

<xsl:template match="/">
  <html>
  <body>
  <h2>My CD Collection</h2>
  <table border="1">
    <tr bgcolor="#9acd32">
      <th>Title</th>
      <th>Artist</th>
    </tr>
    <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>
  </table>
  </body>
  </html>
</xsl:template>

</xsl:stylesheet>
```

The <xsl:template> element is used to build templates.
A template contains rules to apply when a specified node is matched.
The match="/" attribute associates the template with the root of the XML source document.
The XSL <xsl:for-each> element can be used to select every XML element of a specified node-set:
The <xsl:value-of> element can be used to extract the value of an XML element

2:While CSS (Cascading Style Sheets) is primarily used for styling HTML documents, it can also be used to style XML documents when displayed in a web browser.

Here's a basic example of how you can style an XML document using CSS for display in a web browser:

Suppose we have an XML document (`data.xml`) containing information about employees:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type=”text/css” href=”cssfilnename.css” ?>
<employees>
  <employee>
    <name>John Doe</name>
    <position>Software Engineer</position>
    <department>Engineering</department>
  </employee>
  <employee>
    <name>Jane Smith</name>
    <position>UI Designer</position>
    <department>Design</department>
  </employee>
</employees>
```

```css
/* Apply styles to the <employees> element */
employees {
  display: block;
}

/* Apply styles to the <employee> element */
employee {
  display: block;
  margin-bottom: 20px;
}

/* Apply styles to the <name> element */
employee > name {
  font-weight: bold;
}

/* Apply styles to the <position> element */
employee > position {
```

color: #007bff; /* Blue color */
}

/* Apply styles to the <department> element */
employee > department {
color: #28a745; /* Green color */
}
```

XML NAMESPACE:
XML Namespaces provide a method to avoid element name conflicts.In XML, element names are defined by the developer. This often results in a conflict when trying to mix XML documents from different XML applications.
Name conflicts in XML can easily be avoided using a name prefix.
syntax. xmlns:*prefix*="*URI*".

```
<h:table xmlns:h="http://www.w3.org/TR/html4/">
  <h:tr>
    <h:td>Apples</h:td>
    <h:td>Bananas</h:td>
  </h:tr>
</h:table>
```

When a namespace is defined for an element, all child elements with the same prefix are associated with the same namespace.

DTD:

DTD stands for Document Type Definition.

A DTD defines the structure and the legal elements and attributes of an XML document.

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

- !DOCTYPE note - Defines that the root element of the document is note
```

- !ELEMENT note - Defines that the note element must contain the 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"

#PCDATA means parseable character data.
With a DTD, you can verify that the data you receive from the outside world is valid.

XML SCHEMA:
An XML Schema describes the structure of an XML document, just like a DTD.

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

<xs:complexType>
  <xs:sequence>
    <xs:element name="to" type="xs:string"/>
    <xs:element name="from" type="xs:string"/>
    <xs:element name="heading" type="xs:string"/>
    <xs:element name="body" type="xs:string"/>
  </xs:sequence>
</xs:complexType>

</xs:element>
```

- <xs:element name="note"> defines the element called "note"
- <xs:complexType> the "note" element is a complex type
- <xs:sequence> the complex type is a sequence of elements
- <xs:element name="to" type="xs:string"> the element "to" is of type string (text)
- <xs:element name="from" type="xs:string"> the element "from" is of type string
- <xs:element name="heading" type="xs:string"> the element "heading" is of type string
- <xs:element name="body" type="xs:string"> the element "body" is of type string

DTD	XML SCHEMA
Does not support data type	Support datatype
Does not support namespace	Support namespace
Does not define order/sequence of child elements	sequence/order can be defined
Not extensible	extensible

XML DOM:

The XML DOM defines a standard way for accessing and manipulating XML documents. It presents an XML document as a tree-structure.

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

<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>

</bookstore>
```

```
txt = xmlDoc.getElementsByTagName("title")[0].childNodes[0].nodeValue;
```
This code retrieves the text value of the first <title> element in an XML document:
X is the node object

- x.nodeName - the name of x
- x.nodeValue - the value of x
- x.parentNode - the parent node of x
- x.childNodes - the child nodes of x
- x.attributes - the attributes nodes of x

XML DOM Methods

- x.getElementsByTagName(*name*) - get all elements with a specified tag name
- x.appendChild(*node*) - insert a child node to x
- x.removeChild(*node*) - remove a child node from x
- createElement() - method creates a new element node
- replaceChild() - replaces a specified node.