**Reading and Writing an XML file**

**Que.1:** Create a Student table with following fields (S\_id, F\_Name, M\_Name, S\_Name, Address, City, gender, Branch, semester, Contact, Email\_id).

CREATE TABLE [dbo].[Student]

(

[S\_id] INT IDENTITY(1,1) NOT NULL PRIMARY KEY,

[F\_Name] VARCHAR(255) NOT NULL,

[M\_Name] VARCHAR(255),

[S\_Name] VARCHAR(255) NOT NULL,

[Address] VARCHAR(255) NOT NULL,

[City] VARCHAR(100) NOT NULL,

[Gender] VARCHAR(10) NOT NULL,

[Branch] VARCHAR(255),

[Semester] INT,

[Contact] VARCHAR(20),

[Email\_id] VARCHAR(255) NOT NULL

);

**Que.2:** Read the records from the table and store in the xml file “student.xml”. Also display the content of the file on the console. (Use XmlReader and XmlWriter to read and write)..

using System;

using System.Data;

using System.Data.SqlClient;

using System.Xml;

namespace q2

{

class Program

{

static void Main(string[] args)

{

string connectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\iamja\OneDrive\Documents\LAB8.mdf;Integrated Security=True;Connect Timeout=30";

try

{

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

// SQL query to retrieve data from the "User" table and convert it to XML

string query = "SELECT \* FROM [Student] FOR XML AUTO, ELEMENTS, ROOT('Student')";

using (SqlCommand command = new SqlCommand(query, connection))

{

// Execute the query and get the result as a single XML string

string xmlResult = (string)command.ExecuteScalar();

// Save the XML to a file (e.g., "users.xml")

System.IO.File.WriteAllText("Student.xml", xmlResult);

Console.WriteLine("XML data has been written to Student.xml.");

}

}

// Read and display the XML content from the file

string xmlFilePath = "Student.xml";

using (XmlTextReader reader = new XmlTextReader(xmlFilePath))

{

reader.WhitespaceHandling = WhitespaceHandling.None;

using (XmlTextWriter writer = new XmlTextWriter(Console.Out) { Formatting = Formatting.Indented })

{

while (reader.Read())

{

writer.WriteNode(reader, true);

}

}

}

}

catch (Exception ex)

{

Console.WriteLine("An error occurred: " + ex.Message);

}

finally

{

Console.ReadKey();

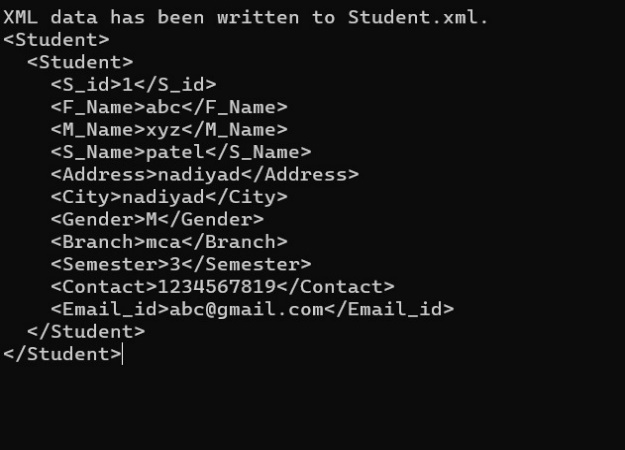
}

}

}

}

**Output:**

****

**Que.3:** Perform program 1 using XmlDocument class.

.

using System;

using System.Data;

using System.Data.SqlClient;

using System.Xml;

namespace q2

{

class Program

{

static void Main()

{

string connectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\iamja\OneDrive\Documents\LAB8.mdf;Integrated Security=True;Connect Timeout=30";

string tableName = "Student";

// Create a connection to the database

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

// SQL query to retrieve records from the Student table

string query = $"SELECT \* FROM {tableName}";

using (SqlCommand command = new SqlCommand(query, connection))

{

using (SqlDataReader reader = command.ExecuteReader())

{

// Create a new XmlDocument

XmlDocument xmlDoc = new XmlDocument();

XmlElement rootElement = xmlDoc.CreateElement("Students");

xmlDoc.AppendChild(rootElement);

while (reader.Read())

{

// Create a new Student element for each record

XmlElement studentElement = xmlDoc.CreateElement("Student");

for (int i = 0; i < reader.FieldCount; i++)

{

// Create elements for each field and add them to the Student element

XmlElement fieldElement = xmlDoc.CreateElement(reader.GetName(i));

fieldElement.InnerText = reader[i].ToString();

studentElement.AppendChild(fieldElement);

}

// Add the Student element to the root element

rootElement.AppendChild(studentElement);

}

// Save the XmlDocument to a file named "student.xml"

xmlDoc.Save("student2.xml");

}

}

}

// Display the content of the XML file on the console

string xmlContent = System.IO.File.ReadAllText("student2.xml");

Console.WriteLine(xmlContent);

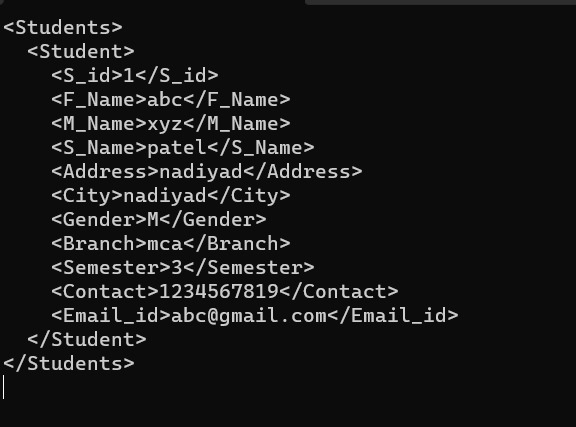
Console.ReadKey();

}

}

}

**Output:**

****

**Que.4:** Read an existing XML file and find out total number of lines in an XML file as well as display total attributes, white spaces, name of element and line number for each line of xml.

using System;

using System.Data;

using System.Data.SqlClient;

using System.Xml;

using System.IO;

namespace q2

{

class Program

{

static void Main()

{

string xmlFilePath = "student.xml"; // Replace with the path to your XML file

if (File.Exists(xmlFilePath))

{

using (XmlTextReader reader = new XmlTextReader(xmlFilePath))

{

int lineCount = 0;

while (reader.Read())

{

switch (reader.NodeType)

{

case XmlNodeType.Element:

lineCount++;

string elementName = reader.Name;

int attributeCount = reader.AttributeCount;

string whitespace = reader.Value; // Get white spaces if needed

Console.WriteLine($"Line: {lineCount}");

Console.WriteLine($"Element: {elementName}");

Console.WriteLine($"Attributes: {attributeCount}");

Console.WriteLine($"White Spaces: {whitespace}");

Console.WriteLine();

break;

case XmlNodeType.EndElement:

break;

case XmlNodeType.Text:

// Handle text nodes if needed

break;

// Add cases for other node types if needed

}

}

Console.WriteLine($"Total number of lines in the XML file: {lineCount}");

Console.ReadKey();

}

}

else

{

Console.WriteLine("The XML file does not exist.");

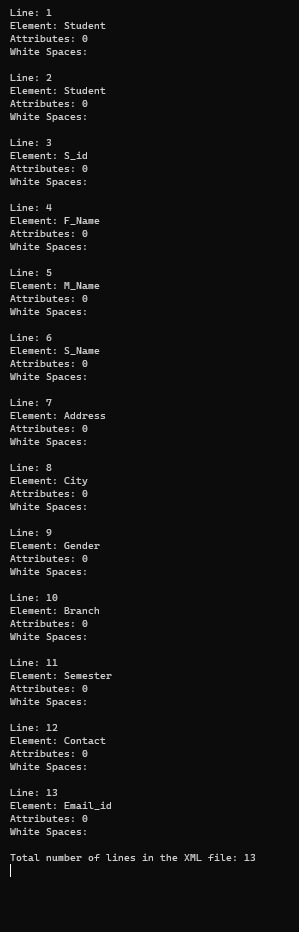
}

}

}

}

**Output :**

****

**Que.5:** Read the content of the xml file created in problem 1. Also create an xslt style-sheet to transform the xml file and display the student data sorted by F\_name in a table in the output.

**student.xslt**

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

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

<xsl:template match="/Students">

<html>

<head>

<style>

table {

border-collapse: collapse;

width: 100%;

}

th, td {

border: 1px solid black;

padding: 8px;

text-align: left;

}

th {

background-color: #f2f2f2;

}

</style>

</head>

<body>

<h1>Student Data Sorted by F\_Name</h1>

<table>

<tr>

<th>S\_id</th>

<th>F\_Name</th>

<th>M\_Name</th>

<th>S\_Name</th>

<th>Address</th>

<th>City</th>

<th>Gender</th>

<th>Branch</th>

<th>Semester</th>

<th>Contact</th>

<th>Email\_id</th>

</tr>

<xsl:apply-templates select="Student">

<xsl:sort select="F\_Name"/>

</xsl:apply-templates>

</table>

</body>

</html>

</xsl:template>

<xsl:template match="Student">

<tr>

<td>

<xsl:value-of select="S\_id"/>

</td>

<td>

<xsl:value-of select="F\_Name"/>

</td>

<td>

<xsl:value-of select="M\_Name"/>

</td>

<td>

<xsl:value-of select="S\_Name"/>

</td>

<td>

<xsl:value-of select="Address"/>

</td>

<td>

<xsl:value-of select="City"/>

</td>

<td>

<xsl:value-of select="Gender"/>

</td>

<td>

<xsl:value-of select="Branch"/>

</td>

<td>

<xsl:value-of select="Semester"/>

</td>

<td>

<xsl:value-of select="Contact"/>

</td>

<td>

<xsl:value-of select="Email\_id"/>

</td>

</tr>

</xsl:template>

</xsl:stylesheet>

**Que5.cs**

using System.Xml.Xsl;

class Program

{

static void Main()

{

string xmlFilePath = "Student.xml";

string xsltFilePath = @"D:\MCA\SEM 3\ASP.net\LAB\q2\q2\student.xslt";

string outputFilePath = @"D:\MCA\SEM 3\ASP.net\LAB\q2\q2\output.html";

XslCompiledTransform xslt = new XslCompiledTransform();

xslt.Load(xsltFilePath);

xslt.Transform(xmlFilePath, outputFilePath);

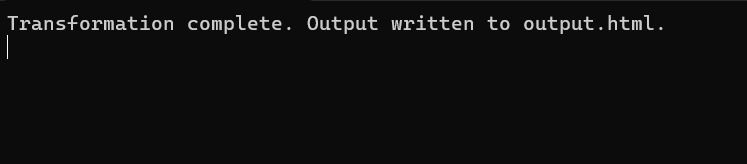
System.Console.WriteLine("Transformation complete. Output written to output.html.");

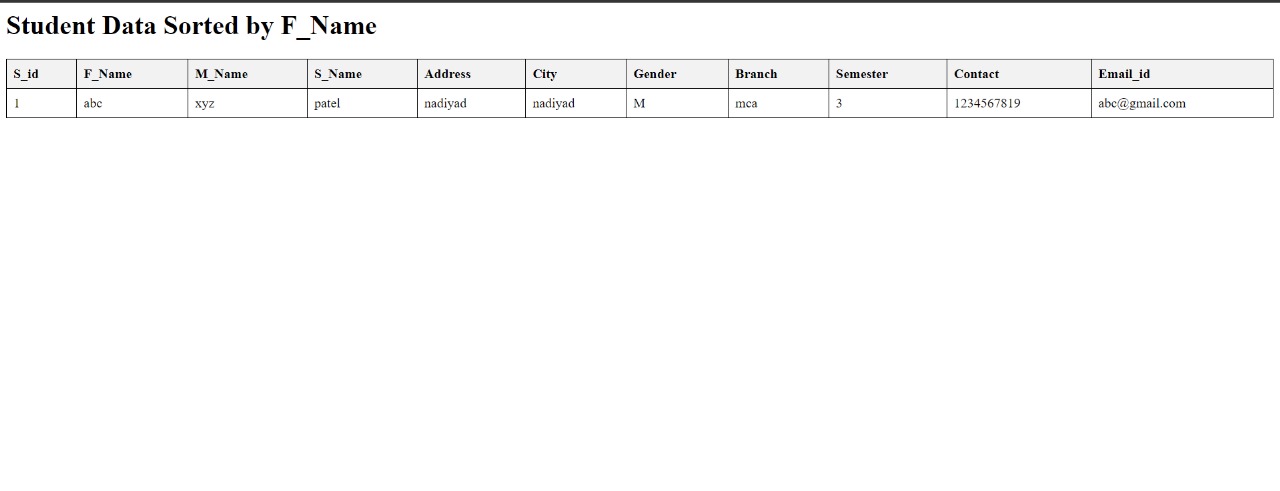
System.Console.ReadKey();

}

}

**Output:**

****

****