**EXP-NO-3**

**A. Creating XML Documents and Document Type Definitions (DTD)**

Creating XML documents involves defining the structure and content of the data using XML syntax. Additionally, Document Type Definitions (DTD) can be used to specify the rules and constraints for the XML document. Below are the steps for creating XML documents and DTDs:

**Creating XML Documents:**

**1. XML Declaration:**

- Begin the XML document with an XML declaration that specifies the version of XML being used.

```xml

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

**2. Root Element:**

- Define a root element that encloses all other elements in the document.

```xml

<root>

<!-- Your content here -->

</root>

**3. Elements and Attributes:**

- Add elements and attributes within the root element to represent your data.

```xm

<person id="1">

<name>John Doe</name>

<age>30</age>

<!-- Other information -->

</person>

```

**4. Nested Elements:**

- You can nest elements to represent hierarchical data.

```xml

<book>

<title>XML Basics</title>

<author>John Smith</author>

<published\_year>2023</published\_year>

</book>

```

**5. CDATA Section:**

- Use CDATA sections for including blocks of text that should not be treated as XML.

```xml

<description><![CDATA[This is a long description with <b>HTML</b> formatting.]]></description>

```

**Document Type Definition (DTD):**

**1. Internal DTD:**

- Define the DTD within the XML document, usually placed after the XML declaration and before the root element.

```xml

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

<!DOCTYPE root [

<!-- DTD rules go here -->

]>

<root>

<!-- Your content here -->

</root>

```

**2. External DTD:**

- Alternatively, you can create a separate DTD file and reference it in the XML document.

- Create a file (e.g., `example.dtd`) with DTD rules.

```xml

<!ELEMENT root (person+)>

<!ELEMENT person (name, age)>

<!ATTLIST person id CDATA #REQUIRED>

<!ELEMENT name (#PCDATA)>

<!ELEMENT age (#PCDATA)>

```

- Reference the DTD in the XML document.

```xml

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

<!DOCTYPE root SYSTEM "example.dtd">

<root>

<!-- Your content here -->

</root>

```

**3. DTD Rules:**

- Use DTD rules to specify the structure of elements, attributes, and their relationships.

- `<!ELEMENT>` defines the type of an element.

- `<!ATTLIST>` defines attributes for an element.

Remember to validate your XML document against its DTD to ensure compliance. Many XML parsers and editors provide validation capabilities. Note that XML Schema (XSD) is another popular schema definition language that offers more features than DTDs and is widely used for XML validation.

**Creating an XML Schema**

**Aim:**

To create an XML Schema.

**Algorithm:**

1. Open a text editor to create a `.dtd` file for XML schema definition.

2. Define the data types of elements that will be included in the XML file.

3. Close the root element of the `.dtd` file and save it.

4. Open another text editor for creating the XML file.

5. Define respective elements as specified in the `.dtd` file and assign values.

6. Close all subsequent element tags.

7. Save the file with a `.xml` extension.

8. Open a web browser to view the XML file.

**Example of an XML Schema:**

```xml

<?xml version="1.0"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"

targetNamespace="http://www.w3schools.com"

elementFormDefault="qualified">

<xs:element name="to" type="xs:string"/>

<xs:element name="heading" type="xs:string"/>

<xs:element name="body" type="xs:string"/>

</xs:schema>

```

**Example of an XML File:**

```xml

<?xml version="1.0"?>

<note xmlns="http://www.w3schools.com"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://www.w3schools.com note.xsd">

<to>Tove</to>

<from>Jani</from>

<heading>Reminder</heading>

<body>Don't forget me!</body>

</note>

```

**B. Storing XML Documents in a Relational Database**

To create a relational database for storing XML documents as text, you need to follow these general steps. The process involves defining a table in the database to hold the XML documents and then inserting or updating records with the XML content. Below is an example using SQL, assuming you are working with a database system that supports SQL (e.g., MySQL, PostgreSQL, SQLite).

**Steps to Store XML Documents:**

**1. Choose a Database System:**

- Select the relational database management system (RDBMS) you wish to use. Common options include MySQL, PostgreSQL, SQLite, or Microsoft SQL Server.

**2. Create a Database:**

- Use SQL to create a new database if one does not already exist.

```sql

CREATE DATABASE YourDatabaseName;

```

**3. Use the Database:**

- Switch to the newly created database.

```sql

USE YourDatabaseName;

```

**4. Create a Table for XML Documents:**

- Define a table that includes a column for storing the XML documents as text.

```sql

CREATE TABLE XmlDocuments (

document\_id INT AUTO\_INCREMENT PRIMARY KEY,

xml\_content TEXT

);

```

**5. Insert XML Documents into the Table:**

- Use SQL `INSERT` statements to add XML documents to the table.

```

INSERT INTO XmlDocuments (xml\_content) VALUES

('<root><person><name>John Doe</name><age>30</age></person></root>'),

('<root><person><name>Jane Doe</name><age>25</age></person></root>');

```

**6. Query and Retrieve XML Documents:**

- Use SQL `SELECT` statements to query and retrieve XML documents from the table.

``

SELECT \* FROM XmlDocuments;

```

**7. Update XML Documents:**

- If necessary, use SQL `UPDATE` statements to modify existing XML documents in the table.

UPDATE XmlDocuments SET xml\_content =

'<root><person><name>NewName</name><age>40</age></person></root>'

WHERE document\_id = 1;

```

**8. Delete XML Documents:**

- If needed, use SQL `DELETE` statements to remove XML documents from the table.

```sql

DELETE FROM XmlDocuments WHERE document\_id = 2;

```