Title :- Understand XML Schema

XML Schema

What is an XML Schema

An XML Schema describes the structure of an XML document.

The XML Schema language is also referred to as XML Schema Definition (XSD).

Example

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

<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:schema>
```

The purpose of an XML Schema is to define the legal building blocks of an XML document:

- the elements and attributes that can appear in a document
- the number of (and order of) child elements
- data types for elements and attributes
- default and fixed values for elements and attributes

Note – XML Schemas support data types. It is major strength Because

- It is easier to describe allowable document content
- It is easier to validate the correctness of data
- It is easier to define data facets (restrictions on data)
- It is easier to define data patterns (data formats)
- It is easier to convert data between different data types

Note: XML Schemas use XML Syntax.

Note: XML Schemas Secure Data Communication.

XML Document VS XML Schema

XML Document

Ex :

```
<?xml version="1.0"?>
<note>
  <to>Ravi</to>
  <from>Raja</from>
  <heading>Reminder</heading>
  <body>Don't forget meeting this weekend!</body>
</note>
```

XML Schema for above example

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="https://www.w3schools.com"
xmlns="https://www.w3schools.com"
elementFormDefault="qualified">

<xs:element name="note"> //complex type because it contains other elements
  <xs:complexType>
    <xs:sequence>
      <xs:element name="to" type="xs:string"/> //simple type.
      <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:schema>
```

Note: Extension of XML Schema file .XSD

The <schema> Element

The <schema> element is the root element of every XML Schema

The <schema> element may contain some attributes. A schema declaration often looks something like this

```
<?xml version="1.0"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="https://www.w3schools.com"
xmlns="https://www.w3schools.com"
elementFormDefault="qualified">
...
...
</xs:schema>
```

* This fragment: xmlns:xs=http://www.w3.org/2001/XMLSchema

Indicate elements and data types used in the schema come from the "http://www.w3.org/2001/XMLSchema" namespace. the namespace should be prefixed with **xs:**

***** This fragment: targetNamespace="https://www.w3schools.com"

Indicates that the elements defined by this schema (note, to, from, heading, body.) come from the "https://www.w3schools.com" namespace.

* This fragment: xmlns=https://www.w3schools.com

Indicates that the default namespace is "https://www.w3schools.com".

* This fragment: elementFormDefault="qualified"

Indicates that any elements used by the XML instance document which were declared in this schema must be namespace qualified.

XSD Simple Elements

A simple element is an XML element that contains only text. But the text can be of many different types. It can be one of the types included in the XML Schema definition (boolean, string, date, etc.), or it can be a custom type that you can define yourself.

The syntax for defining a simple element is

```
<xs:element name="NAME" type="TYPE"/>
```

where NAME is the name of the element and TYPE is the data type of the element.

XML Schema has a lot of built-in data types. The most common types are:

- xs:string
- xs:decimal
- xs:integer
- xs:boolean
- xs:date
- xs:time

XML elements:

```
<lastname>Raja</lastname>
<age>36</age>
<dateborn>1970-03-27</dateborn>
```

the corresponding simple element definitions:

```
<xs:element name="lastname" type="xs:string"/>
<xs:element name="age" type="xs:integer"/>
<xs:element name="dateborn" type="xs:date"/>
```

XSD Attributes

All attributes are declared as simple types.

The syntax for defining an attribute is:

```
<xs:attribute name="xx" type="yy"/>
```

xx is the name of the attribute and yy specifies the data type of the attribute.

XML Schema has a lot of built-in data types. The most common types are:

- xs:string
- xs:decimal
- xs:integer
- xs:boolean
- xs:date
- xs:time

XML element with an attribute:

```
<lastname lang="EN">Smith</lastname>
```

corresponding attribute definition

```
<xs:attribute name="lang" type="xs:string"/>
```

XSD Restrictions/ Features

Restrictions are used to define acceptable values for XML elements or attributes. Restrictions on XML elements are called features.

Restrictions on Values

```
<xs:element name="age">
  <xs:simpleType>
    <xs:restriction base="xs:integer">
      <xs:minInclusive value="0"/>
      <xs:maxInclusive value="120"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

The value of age cannot be lower than 0 or greater than 120

Restrictions on a Set of Values

```
<xs:element name="car">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:enumeration value="Audi"/>
      <xs:enumeration value="Golf"/>
      <xs:enumeration value="BMW"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

The example below defines an element called "car" with a restriction. The only acceptable values are: Audi, Golf, BMW

Restrictions on a Series of Values

```
<xs:element name="letter">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="[a-z]"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

the only acceptable value is one of the lowercase letters from a to z:

Restrictions for DataTypes

Constraint	**Description**
enumeration	Defines a list of acceptable values
fractionDigits	Specifies the maximum number of decimal places allowed. Must be equal to or greater than zero
length	Specifies the exact number of characters or list items allowed. Must be equal to or greater than zero
maxExclusive	Specifies the upper bounds for numeric values (the value must be less than this value)
maxInclusive	Specifies the upper bounds for numeric values (the value must be less than or equal to this value)
maxLength	Specifies the maximum number of characters or list items allowed. Must be equal to or greater than zero
minExclusive	Specifies the lower bounds for numeric values (the value must be greater than this value)
minInclusive	Specifies the lower bounds for numeric values (the value must be greater than or equal to this value)
minLength	Specifies the minimum number of characters or list items allowed. Must be equal to or greater than zero
pattern	Defines the exact sequence of characters that are acceptable
totalDigits	Specifies the exact number of digits allowed. Must be greater than zero
whiteSpace	Specifies how white space (line feeds, tabs, spaces, and carriage returns) is handled