XML Schema Nedir ?

XML Schema(XML Schema Definition-XSD) XML belgelerinin uyması gereken kuralları belirlemek için geliştirilmiş bir dildir. Oluşturulan bir XML belgesinin valid (geçerli) olup olmadığı Schema kullanılarak anlaşılabilir. Yaratılan dilde hangi elementlerin olacağı, elementlerin ne tür attribute'leri olacağı, bir element içinde hangi elementler olabileceği gibi kurallar schema ile tanımlanır.

Aşağıda bir schema ve bir xml belgesi görülmekte. XML belgesi schema'ya uymak zorundadır.

```
 1 <xs:schema
 2   xmlns:xs="http://www.w3.org/2001/XMLSchema"
 3   targetNamespace="http://www.test.com"
 4   xmlns="http://www.test.com">
 5 <xs:element name="name" type="xs:string" />
 6 <xs:element name="surname" type="xs:string" />
 7 <xs:element name="person">
 8   <xs:complexType>
 9     <xs:sequence>
10       <xs:element ref="name" />
11                       <xs:element ref="surname" />
12     </xs:sequence>
13   </xs:complexType>
14 </xs:element>
15 <xs:element name="persons">
16   <xs:complexType>
17     <xs:sequence>
18       <xs:element ref="person" maxOccurs="unbounded" />
19     </xs:sequence>
20   </xs:complexType>
21 </xs:element>
22 </xs:schema>
```

```
 1 <persons xmlns="http://www.test.com"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.test.com persons.xsd">
 2     <person>
 3         <name>Albert</name>
 4         <surname>Einstein</surname>
 5     </person>
 6     <person>
 7         <name>Orhan</name>
 8         <surname>Gencebay</surname>
 9     </person>
10 </persons>
```

Schema XML'inin root elementi aşağıdaki gibi verilmiştir.

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
 targetNamespace="http://www.test.com"
 xmlns="http://www.test.com">
```

xmlns:xs="http://www.w3.org/2001/XMLSchema" ile Schema XML için namespace verilmiştir. xmlns="http://www.test.com ise XML belgelerinin varsayılan namespace'si verilir.

Yaratılan schema'ya uymak için aşağıdaki ifade yazılır.

```
<persons xmlns="http://www.test.com"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.test.com persons.xsd">
```

xmlns="http://www.test.com" varsayılan namespace verilir. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" ile XML belgesinde Schema ile ilgili etiketler kullanmak içindir. Örneğin xsi:schemaLocation attribute'sinde görüldüğü gibi xsi namespace'si kullanılmaktadır. xsi:schemaLocation="http://www.w3schools.com note.xsd" ise iki değer alır. Birinci değer namespace ikinci değer ise Schmea dosyasıdır.

Element

XML belgelerinde kullanılacak elementler xs:element etiketiyle tanımlanırlar.

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

ifadesinde name elementi yaratılmıştır. name elementi string değerleri alacaktır. xs:string ,xs:decimal ,xs:integer ,xs:boolean ,xs:date ,xs:time gibi başka tipler verilebilir.

Attribute

Attribute'ler xs:attribute elementi ile yaratılırlar.

```
<xs:element name="no" type="xs:string" />
```

ifadesinde no attirbute'si yaratılmıştır. default ile varsayılan değer verilebilir. İsteğe bağlı veya gerekli olması için user attributesi kullanılır.

```
<xs:element name="no" type="xs:string" use="optional"/>
<xs:element name="ad" type="xs:string" use="required"/>
```

no attributesi isteğe bağlı , ad ise zorunlu yapılmıştır.

ref özelliği

Yaratılan bir element veya attribute ref ile bir çok yerde kullanılabilir. Örneğin

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

yaratılan bir elemnent başka bir yerde

```
<xs:element ref="name" />
```

şeklinde kullanılabilir.

Yeni Tip Yaratma

xs:string ,xs:decimal ,xs:integer ,xs:boolean ,xs:date ,xs:time tiplerinden başka kendinize yeni tip yaratabilirsiniz. Örneğin 0-100 arasında sayı değeri alan bir tip yapılabilir.

```
<xs:simpleType>
  <xs:restriction base="xs:integer">
    <xs:minInclusive value="0"/>
    <xs:maxInclusive value="100"/>
  </xs:restriction>
</xs:simpleType>
```

Bir tip iki şekilde kullanılabilşir. Bir elementin içine yazılabilir.

```
<xs:element name="no" type="xs:string" use="optional">
 type buraya
</xs:element>
```

veya bir isim verilerek birden fazla yerde kullanılabilir.

```
<xs:element name="test" type="tipAdi"/>

<xs:simpleType name="tipAdi">
...
</xs:simpleType>
```

test elementinin tipi "tipAdi" yapılmıştır. Aynı tip birden fazla element için kullanılabilir.

DataType'leri için aşağıdaki sınırlandırmalar bulunmaktadır.

- **enumeration** : Bir değer listesi tanımlar. Listede olmayan bir değer verilemez.
- **fractionDigits**: decimal alanın maxsimum değeri
- **length** : string veya list gibi tiplerde boy değeri
- **maxExclusive** : Sayısal ve liste değerleri için üst sınır
- **maxInclusive**: Sayısal ve liste değerleri için üst sınır
- **maxLength**: string veya listeler için girilebilecek maxsimum boy.
- **minExclusive**: Sayısal ve liste değerleri için alt sınır.
- **minInclusive**: Sayısal ve liste değerleri için alt sınır
- **minLength**: string veya listeler için girilebilecek minimum boy
- **pattern**: string değerinin uymak zorunda olduğu şablon
- **totalDigits**: sayı tipleri için girilebilecek rakam miktarı
- **whiteSpace**: whitespace karakterlere bir işlem yapmak için

Belirli değerler dizisi yaratılabilir. Örneğin medeni durum için bekar,evli,dul değerleri olan bir tip yaratılabilir.

```
<xs:simpleType>
  <xs:restriction base="xs:string">
    <xs:enumeration value="Bekar"/>
    <xs:enumeration value="Evli"/>
    <xs:enumeration value="Dul"/>
  </xs:restriction>
</xs:simpleType>
```

Belirli yapıya uyan değer tipleri yaratılabilir. Örneğin tek bir harf alan değer tipi yaratılabilir.

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

Bir tipin whitespace karakterlerini desteklemesi sağlanabilir.

```
<xs:simpleType>
  <xs:restriction base="xs:string">
    <xs:whiteSpace value="replace"/>
  </xs:restriction>
</xs:simpleType>
```

Boy sınırı verilebilir

```
<xs:simpleType>
  <xs:restriction base="xs:string">
    <xs:length value="8"/>
  </xs:restriction>
</xs:simpleType>
```

8 buyunda bir string tipi yaratılmıştır.

Complex Element Yaratma

Bir element attribute veya içerisinde başka elementler içerebilir. Bu elementler complex element olurlar. Complex element yaratmak için <xs:complexType> kulanılır.

```
<xs:element name="test">
  <xs:complexType>
 ...
  </xs:complexType>
</xs:element>
```

test elementinin yapısı xs:complexType içinde tanımlanacaktır.

Bir element bir veya birden fazla attribute içerebilir.

```
<xs:element name="person">
  <xs:complexType>
    <xs:attribute name="no" type="xs:integer"/>
  </xs:complexType>
</xs:element>
```

person elementinin no attributesi yaratılmıştır.

Bir element birden fazla element içerebilir.

```
<xs:element name="person">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="name" type="xs:string" />
                        <<xs:element name="surname" type="xs:string" />
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

person elementi name ve surname elementleri içermektedir. <xs:sequence> ifadesi "indicator"(gösterge) elementidir ve bu elementlerin sıra ile kullanılması gerektiğini belirler. <xs:all> kullanılırsa içindeki elementlerden herhangi bir sadece 1 kere ve istenildiği sırada kullanılabilir demektir. <cs:choice> ise içlerinden sadece bir elementi kullanbilirsiniz demektir.

Birden fazla elementten bir complex type yaratılabilir. Örneğin name ve surname için bir complex type yaratılabilir.

```
<xs:complexType name="name">
  <xs:sequence>
    <xs:element name="firstname" type="xs:string"/>
    <xs:element name="lastname" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
```

şeklinde name tipi tanımlanmıştır.

```
<xs:element name="person" type="name"/>
```

şeklinde bir element yaratılabilir. Bu şekilde person elementinin içinde firstname ve lastname elementleri olacaktır. Bir complext type başka bir complex type'ta türüyebilir. Yeni bir type yapıp başka bir type'ın özellikleri alınabilir. Yukarıdaki name tipinin daha geniş fullname tipi yaratalım.

```
<xs:complexType name="fullname">
  <xs:complexContent>
    <xs:extension base="name">
      <xs:sequence>
        <xs:element name="lakap" type="xs:string"/>
        <xs:element name="kisaltma" type="xs:string"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

<xs:extension base="name"> ifadesi fullname'in name tipinin özelliklerini içermesini sağlar.

maxOccurs ve minOccurs

Bir elementin kaç kere kullanılabilecğini bildirir. Varsayılan olarak iki değerde 1'dir. minOccurs en az kaç kere kullanılması gerektiği, maxOccurs ise en fazla kaç kere kullanılması gerektiğini belirtir.

```
<xs:element name="name" type="xs:string" maxOccurs="10" minOccurs="0"/>
```

name elementi hiç kullanılmayabilir ve en fazla 10 tane kullanılabilir.

Element'leri ve Attribute Gruplandırma

Birden fazla element ve attribute bir gruba koyulabilir. Bu gruplar daha sonra grup olarak kullanılabilir.

```
<xs:group name="persongroup">
  <xs:sequence>
    <xs:element name="firstname" type="xs:string"/>
    <xs:element name="lastname" type="xs:string"/>
    <xs:element name="birthday" type="xs:date"/>
  </xs:sequence>
</xs:group>
```

Yukarıda üç elementten grup yaratılmıştır. Bu grup daha sonra kullanılabilir.

```
<xs:complexType name="personinfo">
  <xs:sequence>
    <xs:group ref="persongroup"/>
    <xs:element name="country" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
```

personinfo elementi firstname,lastname,birthday ve country elementini içerecektir.

```
<xs:attributeGroup name="identity">
  <xs:attribute name="no" type="xs:integer"/>
  <xs:attribute name="id" type="xs:string"/>
</xs:attributeGroup>
```

şeklinde attribute grubu tanımlanabilir.