SGML

- SGML (Standard Generalized Markup Language), deals with the structural markup of electronic documents.
- SGML is a standard for how to specify a document markup language or tag set.
- SGML is not in itself a document language, but a description of how to specify one. It is metadata.
- In order to treat documents electronically it is essential that their logical structure be clearly marked. On top of that, to ensure that documents are really interchangeable, one had to develop a common language to implement this type of representation.
- SGML is a standard method of representing the information contained in a document independently of the system used for input, formatting or output.

SGML

- SGML is a system for organization and tagging elements of document.
- SGML developed and standardized by the international organization for standard (ISO) in 1986.
- SGML itself doesn’t specify any particular formatting; rather it specifies the rules for tagging elements.
- These tags can then be interpreted to format elements in different ways.
- SGML is a system for defining markup languages.
- HTML is an example of an SGML-based language.
- XML, which is a data description language, uses SGML principles.
- Both HTML and XML are the subset of SGML.

SGML Characteristics

There are three characteristics of SGML which distinguish it from other markup languages: its emphasis on **descriptive** rather than procedural markup; its **document type** concept; and its **independence** of any one system for representing the script in which a text is written.

1. **Descriptive Markup:-** A descriptive markup system uses markup codes which simply provide names to categorize parts of a document. Markup codes such as **< para>** simply identify a portion of a document and assert of it that "the following item is a paragraph". By contrast, a procedural markup system defines what processing is to be carried out at particular points in a document.

SGML Characteristics

2. **Types of Document:-** SGML introduces the notion of a Document Type Definition (DTD). Documents are regarded as having types. The type of a document is formally defined by its constituent parts and their structure. The definition of a report, for example, might be that it consisted of a title and possibly an author, followed by an abstract and a sequence of one or more paragraphs. Anything lacking a title, according to this formal definition, would not formally be a report.

3. **Data Independence:-** A basic design goal of SGML was to ensure that documents encoded according to its provisions should be transportable from one hardware and software environment to another without loss of information.

XML

- **Xml** (e**X**tensible **M**arkup **L**anguage) is a mark up language.
- XML is designed to store and transport data.
- Xml was released in late 90's.
- XML was created to provide an easy to use and store self describing data.
- XML became a W3C Recommendation on February 10, 1998.
- XML is not a replacement for HTML.
- XML is designed to be self-descriptive.
- XML is designed to carry data, not to display data.
- XML tags are not predefined. We must define our own tags.
- XML is platform independent and language independent.
- XML was designed to be both human and machine-readable.
- XML is often used for distributing data over the Internet.
- XML can be used for offloading and reloading of databases.

An Example of XML

```
<?xml version="1.0" encoding="UTF-8"?>
<student>
  <firstname> Ronit </firstname>
  <lastname> Kumar </lastname>
  <email> ronitkumar @gmail.com</email>
  <phoneno> 9898989898 </phoneno>
</student >
```

- The XML above does not do anything. XML is just information wrapped in tags.
- XML uses a much self-describing syntax.
- A prolog defines the XML version and the character encoding:
  <?xml version="1.0" encoding="UTF-8"?>
- The next line is the root element of the document: <student>
- The <student> elements have 4 child elements:
  <firstname>, <lastname>, <email> and <phoneno>

XML

- We don't **compile** XML - the whole point is that it is plain text compiling so some binary format negates its very benefits.
- We don't **run** an XML file - it is a document that contains data i.e. a data file, not a list of instructions like a programming language. We can use some other programming language of our choice to process the XML document.

XML Separates Data from HTML

- When displaying data in HTML, we should not have to edit the HTML file when the data changes.
- With XML, the data can be stored in separate XML files.

XML Syntax

- The XML language has no predefined tags.
- The tags in the example above like <firstname> and <lastname> are not defined in any XML standard. These tags are "invented" by the author of the XML document.
- In XML, it is illegal to omit the closing tag. All elements must have a closing tag: <p>This is a paragraph.</p>
  <br />
- XML tags are case sensitive. The tag <Letter> is different from the tag <letter>.
- The syntax for writing comments in XML is similar to that of HTML.
- <!-- This is a comment -->
- XML elements can have attributes. By the use of attributes we can add the information about the element.
- XML attributes enhance the properties of the elements.
- In XML, the attribute values must always be quoted.
- <student class="BCA II Semester"> </student>

XML Syntax

- XML does not truncate multiple white-spaces (HTML truncates multiple white-spaces to one single white-space):

XML:	Hello World
HTML:	Hello World

- Some characters have a special meaning in XML. If we place a character like "<" inside an XML element, it will generate an error because the parser interprets it as the start of a new element.
- To avoid this error, replace the "<" character with an entity reference.
- There are 5 pre-defined entity references in XML:

```
<message>salary &lt;  9000</message>
```

&lt;	<	less than
&gt;	>	greater than
&amp;	&	ampersand
&apos;	'	apostrophe
&quot;	"	quotation mark

SGML Versus XML

- XML is a subset of SGML.
- XML is simpler compared to SGML.
- XML documents should be readable with SGML parsers while some SGML might produce errors in XML parsers.
- A list of SGML declarations (for ex. DATATAG, OMITTAG, RANK, LINK, CONCUR, SUBDOC, FORMAL) have been removed in XML.
- Some constructs that are allowed in SGML (for ex. Empty start tags, Empty end tags, Unclosed start tags, Unclosed end tags,) are no longer permitted in XML.
- Some SGML entities (for ex. External SDATA entities, Internal SDATA entities, External CDATA entities, Internal CDATA entities, #DEFAULT entities, PI entities, Bracketed text entries) are no longer allowed in XML.
- Some comment practices in SGML have also been disallowed in XML.

	XML	HTML
Stands For	XML stands for “Extensible Mark Up Language”.	HTML stands for “HyperText Markup Language”.
Purpose	XML is focuses on data stored and how its carried and described i.e. “XML Describes the Data”.	HTML is focuses on display and look of data i.e. “HTML Defines or Displays the Data”.
Used For	XML is basically used to transport data between the application and the database.	HTML is used for designing a web page to be rendered on the client side.
Rules	XML is follow strict rules. Anytime terminate the process if rules break.	HTML is not following strict rules. All browsers try to display data to the best as per its ability.
Tag Criteria	In XML once a tag is opened, it must be closed.	In HTML, its not necessary to close a tag. This is a paragraph.
Case Sensitive	XML is Case Sensitive i.e. Opening and closing tags must be in the same case.	HTML is not Case Sensitive.
Define Tag	In XML, there is no pre-defined tags, user have to define his own tags.	In HTML, there is pre-defined tags and user have to use those tags.

	XML	HTML
Structure	In XML, user can define his own tags as well as structure of document.	in HTML, structure of HTML document as well as tags are pre-defined.
Extensible	XML is an extensible so user can add own tags to extend the XML.	HTML is markup language having its pre-defined tags and user can't add own tags.
Attribute	In XML value of attribute must always be quoted.	In HTML it is not necessary to quote the values.
Behavior	XML is dynamic for holding data.	HTML is static for displaying data.
Nesting	In XML, Elements must be nested properly.	HTML is not so sensitive so it allows improper nesting.
Paired Tags	There is only Paired Tag in XML.	There is Both types of tags, Paired tags and Singular Tags in HTML.
White space	In XML, white space is preserved.	In HTML, multiple white space is truncated to one single white space i.e. white space is not preserved.

Modeling XML Data

- The process of creating a schema for an XML document is known as data modeling because it involves resolving a class of data into elements and attributes that can be used to describe the data in an XML document.
- The real importance of schemas is that they allow XML documents to be validated for accuracy.
- This simply means that a schema allows an XML developer (or an application) to process a document and see if it adheres to the set of constraints laid out in the schema. If not, we know the document could prove to be problematic.
- The main reason schemas are used in XML is to provide a mechanism to facilitate the process of validating XML documents.
- When it comes to creating schemas, there are three approaches we can take:
  - ❖ **Document Type Definitions (DTDs)**
  - ❖ **XML Schemas (XSDs)**
  - ❖ **XML Data Reduced(XDRs)**

Data Modeling with DTD

- DTDs represent the original approach of creating a schema for XML documents. By "original approach" we means that DTDs did not originate with XML. DTDs originated with XML's predecessor, SGML (Standard General Markup Language).
- DTDs made their way into XML because it eased the transition from SGML to XML and many SGML tools existed that could be used for XML.
- Things have changed since the early days of XML, however, and now there is a more powerful approach to establishing schemas than DTDs. Even so, DTDs are still in use.
- The main drawback to DTDs is that they are based upon a somewhat cryptic language.
- XML provides a highly structured approach to formatting data, so why should we have to learn an entirely new language to describe XML schemas?
- But DTDs are actually quite simple for describing most XML-based markup languages. This is due to the fact that the DTD language is extremely compact, which is why it has a cryptic appearance.

Data Modeling with DTD

- A document type definition (DTD) is a set of markup declarations that define a document type for an SGML-family markup language (SGML, XML, HTML).
- A DTD defines the structure and the legal elements and attributes of an XML document.
- DTD is a way to describe XML language precisely. DTDs check vocabulary and validity of the structure of XML documents against grammatical rules of appropriate XML language.
- XML uses a subset of SGML DTD.

When to Use a DTD?

- With a DTD, independent groups of people can agree to use a standard DTD for interchanging data.
- With a DTD, you can verify that the data you receive from the outside world is valid.
- You can also use a DTD to verify your own data.

DTD Declaration

- A DTD can be declared inline inside an XML document, or as an external reference.

An Internal DTD Declaration

```
<?xml version="1.0"?>
<!DOCTYPE note [
<!ELEMENT note (to,from,heading,body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
]>
<note>
<to>Harry</to>
<from>John</from>
<heading>Invitation</heading>
<body>You are invited to join us for some birthday fun</body>
</note>
```

The DTD above is interpreted like this:

- !DOCTYPE note defines that the root element of this document is note.
- !ELEMENT note defines that the note element must contain four 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".
- Here, #PCDATA means parse-able text data.

An External DTD Declaration

If the DTD is declared in an external file, the <!DOCTYPE> definition must contain a reference to the DTD file:

```
<?xml version="1.0"?>
<!DOCTYPE note SYSTEM "note.dtd">
<note>
<to>Harry</to>
<from>John</from>
<heading>Invitation</heading>
<body>You are invited to join us for some birthday fun</body>
</note>
```

And here is the file "note.dtd", which contains the DTD:

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

Data Modeling with XSD

- XML Schema replaces DTDs with a more powerful and intuitive approach to creating schemas for XML-based markup languages.
- Schemas created using XML Schema are coded in the XSD (XML Schema Definition) language, and are therefore referred to as XSDs.
- XML Schema and the XSD language were created by the W3C.
- The idea behind XML Schema is to use XML as the basis for creating schemas. So, instead of using the special DTD language to create a schema, you can use familiar XML elements and attributes that are defined in the XSD language.
- Similar to DTDs, XSDs describe elements and their content models so that documents can be validated.
- However, XSDs go several steps beyond DTDs by allowing you to associate data types with elements.
- In a DTD, element content is pretty much limited to text. An XSD is more flexible in that it can set the data type of elements to specific types, such as integer numbers, strings, dates etc.
- The most compelling aspect of XSDs is the fact that they are based upon an XML vocabulary. This means that we can create an XSD as an XML document. So, the familiar tag/attribute approach to encoding XML documents is all you need to know to code an XSD document.

An Example of XSD

- The following example is a simple XML file that contains some information about the three brightest stars at night.

```
<?xml version="1.0"?>
<brightstar>
        <name>Sirius</name>
        <magnitude>1.45</magnitude>
        <distance>9</distance>

        <name>Canopus</name>
        <magnitude>-5.53</magnitude>
        <distance>310</distance>

        <name>Rigil Kentaurus</name>
        <magnitude>4.34</magnitude>
        <distance>4</distance>
</brightstar>
```

An Example of XSD

- XSD files are made up of tags, just like an XML file. In fact, XSD files actually are XML files.
- Like an XML file, an XSD file has elements, and each element must have a name and a type.

```
<xs:element name="brightstar">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="name" type="xs:string"/>
      <xs:element name="magnitude" type="xs:decimal"/>
      <xs:element name="distance" type="xs:integer"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Data Modeling with XDR Schema

- XDR, or XML Data Reduced, is an XML vocabulary invented by Microsoft that allows us to describe the schema of an XML document.
- The XDR describes that schema in terms of not only the document's content, but also which types of content are contained in the document's elements.
- The primary drawback to using XDR is that it's limited to Microsoft products and technologies – other vendors don't support XDR.

An Example of XDR

```
<?xml version="1.0" encoding="UTF-8"?>
<Schema name="Untitled-schema"
    xmlns="urn:schemas-microsfot-com:xml-data"
    xmlns:dt="urn:schemas-microsoft-com:datatypes">
     <ElementType name="people" model="closed"
          content="eltOnly" order="seq">
          <AttributeType name="xmlns" dt:type="string"/>
          <attribute type="xmlns"/>
          <element type="person" minOccurs="1" maxOccurs="*" />
       </ElementType>
     <ElementType name="person" model="closed"
          content="eltOnly" order="seq">
          <element type="name" minOccurs="1" maxOccurs="1" />
       </ElementType>
```

Thank you