**UNIT IV XML DATABASES 9**

Structured, Semi structured, and Unstructured Data – XML Hierarchical Data Model – XML Documents – Document Type Definition – XML Schema – XML Documents and Databases – XML Querying – XPath – XQuery

---

## STRUCTURED, SEMI STRUCTURED, AND UNSTRUCTURED DATA

Big Data includes huge volume, high velocity, and extensible variety of data. These are 3 types: **Structured data, Semi-structured data, and Unstructured data.**

**Structured data:**

Structured data is data whose elements are addressable for effective analysis. It has been organized into a formatted repository that is typically a database. It concerns all data which can be stored in database SQL in a table with rows and columns. They have relational keys and can easily be mapped into pre-designed fields. Today, those data are most processed in the development and simplest way to manage information. **Example: Relational data.**

**Semi-Structured data:**

Semi-structured data is information that does not reside in a relational database but that has some organizational properties that make it easier to analyze. With some processes, you can store them in the relation database (it could be very hard for some kind of semi-structured data), but Semi-structured exist to ease space. **Example: XML data.**

**Unstructured data:**

Unstructured data is data which is not organized in a predefined manner or does not have a predefined data model, thus it is not a good fit for a mainstream relational database. So for Unstructured data, there are alternative platforms for storing and managing, it is increasingly prevalent in IT systems and is used by organizations in a variety of business intelligence and analytics applications. **Example: Word, PDF, Text, Media logs.**

Differences between Structured, Semi-structured and Unstructured data:

| **Properties** | **Structured data** | **Semi-structured data** | **Unstructured data** |
|---|---|---|---|
| Technology | It is based on Relational database table | It is based on XML/RDF(Resource Description Framework). | It is based on character and binary data |
| Transaction management | Matured transaction and various | Transaction is adapted from DBMS not matured | No transaction management and no concurrency |

| | | | |
|---|---|---|---|
| | concurrency techniques | | |
| Version management | Versioning over tuples,row,tables | Versioning over tuples or graph is possible | Versioned as a whole |
| Flexibility | It is schema dependent and less flexible | It is more flexible than structured data but less flexible than unstructured data | It is more flexible and there is absence of schema |
| Scalability | It is very difficult to scale DB schema | It's scaling is simpler than structured data | It is more scalable. |
| Robustness | Very robust | New technology, not very spread | |
| Query performance | Structured queries allow complex joining | Queries over anonymous nodes are possible | Only textual queries are possible |

---

**XML HIERARCHICAL DATA MODEL**

**XML Tree Structure**

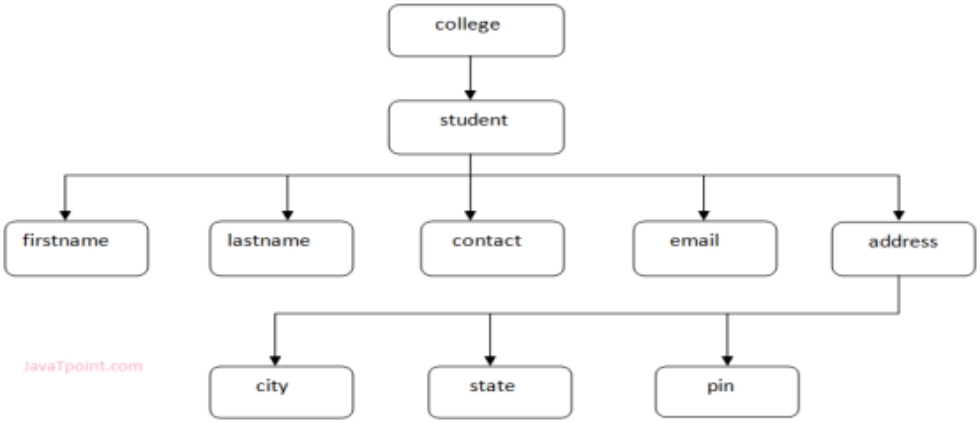
An XML document has a self descriptive structure. It forms a tree structure which is referred to as an XML tree. The tree structure makes it easy to describe an XML document.

A tree structure contains root element (as parent), child element and so on. It is very easy to traverse all succeeding branches and sub-branches and leaf nodes starting from the root.

```
<?xml version="1.0"?>
    <college>
    <student>
    <firstname>Adlie</firstname>
    <lastname>Stefhan</lastname>
    <contact>09990449935</contact>
    <email>adliestefhan@abc.com</email>
    <address>
```

<city>Kanyakumari</city>
<state>TamilNadu</state>
<pin>629802</pin>
</address>
</student>
</college>

Let's see the tree-structure representation of the above example.




In the above example, first line is the XML declaration. It defines the XML version 1.0. Next line shows the root element (college) of the document. Inside that there is one more element (student). Student element contains five branches named <firstname>, <lastname>, <contact>, <Email> and <address>.<address> branch contains 3 sub-branches named <city>, <state> and <pin>.

**XML Tree Rules**

These rules are used to figure out the relationship of the elements. It shows if an element is a child or a parent of the other element.

**Descendants:** If element A is contained by element B, then A is known as descendant of B. In the above example "College" is the root element and all the other elements are the descendants of "College".

**Ancestors:** The containing element which contains other elements is called "Ancestor" of other element. In the above example Root element (College) is ancestor of all other elements.

**What is xml?**

- Xml (eXtensible Markup Language) is a mark up language.
- XML is designed to store and transport data.

- Xml was released in late 90's. it 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. You must define your own tags.
- XML is platform independent and language independent.

**XML Example**

```
<bookstore>
    <book category="COOKING">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
    </book>
    <book category="CHILDREN">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
    </book>
    <book category="WEB">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
    </book>
    </bookstore>
```

The root element in the example is <bookstore>. All elements in the document are contained within <bookstore>.The <book> element has 4 children: <title>,< author>, <year> and <price>.

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

<to>Vimal</to>
<from>Sonoo</from>
<heading>Hello</heading>
<body>Hello brother, how are you!</body>
</email>
<email>
<to>Peter</to>
<from>Jack</from>
<heading>Birth day wish</heading>
<body>Happy birth day Tom!</body>
</email>
<email>
<to>James</to>
<from>Jaclin</from>
<heading>Morning walk</heading>
<body>Please start morning walk to stay fit!</body>
</email>
<email>
<to>Kartik</to>
<from>Kumar</from>
<heading>Health Tips</heading>
<body>Smoking is injurious to health!</body>
</email>
</emails>

**XML Attributes**

XML elements can have attributes. By the use of attributes we can add the information about the element.

<book publisher='Tata McGraw Hill'></book>

Metadata should be stored as attribute and data should be stored as elements

<book>
<book category="computer">
<author> A & B </author>
</book>

**XML Comments**

XML comments are just like HTML comments. We know that the comments are used to make codes more understandable to other developers.

An XML comment should be written as:

<!-- Write your comment-->

**XML Validation**

A well formed XML document can be validated against DTD or Schema. A well-formed XML document is an XML document with correct syntax. It is very necessary to know about valid XML documents before knowing XML validation.

**Valid XML document**

- It must be well formed (satisfy all the basic syntax condition)
- It should be behave according to predefined DTD or XML schema

**Rules for well formed XML**

- It must begin with the XML declaration.
- It must have one unique root element.
- All start tags of XML documents must match end tags.
- XML tags are case sensitive.
- All elements must be closed.
- All elements must be properly nested.
- All attributes values must be quoted.
- XML entities must be used for special characters.
- XML Validation

---