**XML**

XML : eXtensible markup language

XML Tree Structure :

Root (database Name)

Elements inside root,

Attributes inside elements: lang, category, date (as you want)

First line in XML file XML PROLOG (describe the version of xml used, encoding and define Standalone for separated related files (optional))

<?xml version = ‘1.0’ encoding = ‘utf-8’ standalone = ‘yes’ (optional) >

Tags are Case Sensitive <Letter> != <letter>

Comment like HTML : <!-- comment -->

To use namespaces in XML you must add : xmlns:h=<http://www.w3.org/TR/html4/> to the tag you want to separate it for example we have 2 tables in xml file :  
<root>

<h:table xmlns:h="http://www.w3.org/TR/html4/">  
 <h:tr>   
 <h:td>Apples</h:td>

<h:td>Bananas</h:td>  
 </h:tr>  
</h:table>  
  
<f:table xmlns:f="https://www.w3schools.com/furniture">  
 <f:name>mounir</f:name>  
 <f:width>80</f:width>  
 <f:length>120</f:length>  
</f:table>  
  
</root>

**XQuery**

Doc(“example.xml”): to select nodes from xml file

Doc(“example.xml”)/path example: Doc(“example.xml”)/Bookstore/book/title to navigate throw the xml file

**Predicates:** Doc(“example.xml”)/Bookstore/book[price<30]

**FLWOR (flower):**

**For:** selects a sequence of nodes

**Let:** binds a sequence to a variable

**WHERE**: filters the nodes

**ORDER BY:** sorts the nodes

**RETURN:** what to return

**IF-Then-Else:**

for $x in doc("books.xml")/bookstore/book

return

if ($x/@category="children")

then

<child>{data($x/title)}</child>

Else

<adult>{data($x/title)}</adult>

**Path expression**:

doc("books.xml")/bookstore/book[price>30]/title

**FLWOR expression**:

for $x in doc("books.xml")/bookstore/book

where $x/price>30

return $x/title

The for clause selects all book elements under the bookstore element into a variable called $x. so $x = book