LinkedIn Quiz

Q1. You are working with this XML code snippet from the XML document cars.xml.

You need to return the information about the cars built after the year 2000. What does your XQuery look like?

```
<cars>
    <car><make>Cadillac</make><model>Escalade</model><year>2007</year></car>
    <car><make>Cadillac</make><model>Escalade</model><year>2011</year></car>
    <car><make>Ford</make><model>Mustang</model><year>1968</year></car>
    <car><make>Ford</make><model>Mustang</model><year>1998</year></car>
    <car><make>Mercedes</make><model>C-Class</model><year>1999</year></car>
    <car><make>Mercedes</make><model>C-Class</model><year>2009</year></car>
</cars>
```

Answers:

```
doc("cars.xml")/cars/car[year>2000].data
```

```
doc("cars.xml")/cars/car[xs:integer(year) gt 2000]
```

```
doc("cars.xml")/cars/car[year gt 2000]
```

```
doc("cars.xml")/cars/car[integer(year) > 2000]
```

Explanation: the answer is based on the syntax of the XQuery language

Link: XQuery FLWOR Expressions

Q2. You are working with the following XSD fragment. What does it say about the <car> element?

```
<xs:element name="car">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="make" type="xs:string"/>
            <xs:element name="model" type="xs:string"/>
            <xs:element name="year" type="xs:string"/>
        </xs:sequence>
        <xs:anyAttribute/>
    </xs:complexType>
</xs:element>
```

Answers:

- The <car> element can be extended with only one attribute
- The <car> element can be extended with multiple attributes
- The <car> element can't have any attributes
- The <car> element has child elements which can appear in any order

Explanation:

The anyAttribute element enables us to extend the XML document with attributes not specified by the schema.

The following example is a fragment from an XML schema called "family.xsd". It shows a declaration for the "person" element.

By using the <anyAttribute> element we **can add any number of attributes** to the "person" element:

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

Reference: XSD The <anyAttribute> Element

Q3. You are converting your HTML file into XHTML Strict.

Which code snippet will validate without errors?

Answers:

```
<html xmlns="http://www.w3.org/1999/xhtml">
    <head><title>XHTML Example</title></head>
    <body bgcolor="#FFFFFF" >
        <p>Content goes here ...</p>
    </body>
</html>
Comment: The bgcolor attribute on the body element is obsolete. Use CSS instead.
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
    <head><title>XHTML Example</title></head>
    <body name="bodySection">
        <p><b>Content goes here ...</b></p>
    </body>
</html>
Comment: Attribute name not allowed on element body at this point.
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
    <head><title>XHTML Example</title></head>
    <body color="#333333">
        <p><i>Content goes here ...</i></p>
    </body>
</html>
Comment: Attribute color not allowed on element body at this point.
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
    <head><title>XHTML Example</title></head>
    <body id="bodySelection">
        <p><strong>Content goes here ...</strong></p>
    </body>
</html>
```

Q4. When working with Ajax applications, which is faster, XML or JSON?

Answers:

- XML, because it is extensible
- JSON, because it transfers data without waiting for a server response
- XML, because it supports namespaces

- JSON, because it is already parsed into a JavaScript object

Explanation:

Ajax is an acronym for Asynchronous Javascript and XML. It is used to communicate with the server without refreshing the web page and thus increasing the user experience and better performance.

…

JSON is faster because it is designed specifically for data interchange. JSON encoding is terse, which requires less bytes for transit. JSON parsers are less complex, which requires less processing time and memory overhead.

XML is slower, because it is designed for a lot more than just data interchange.

Q5. Asynchronous Javascript and XML (Ajax) is a technique for creating better, faster, and more interactive web applications.

In addition to JavaScript and XML on the back end, which technologies are commonly used to craft AJAX experiences on the front end?

Answers:

- PHP, .NET, and SQL
- HTML, CSS, and DOM
- Python, Perl, and C++
- Java, ASP, and C#

Explanation: other answers contain backend development languages.

Q6. What is this code an example of?

```
<x/>
```

Answers:

- null element
- self-closing tag
- improperly named element
- incorrect XML syntax

Explanation:

An element with no content is said to be empty. In XML, you can indicate an empty element like this:

```
<element></element>
```

You can also use a so called self-closing tag:

```
<element />
```

Reference: https://www.w3schools.com/xml/xml_elements.asp

Q7. Which XHTML syntax rule does NOT apply to XML?

Answers:

- XHTML attribute values must be quoted
- XHTML tags and attributes must be in lowercase
- XHTML elements must be properly nested within each other
- XHTML tags must have an equivalent closing tag

Explanation: XML Attributes values must be quoted.

Element names are case-sensitive and CamelCase is actually one of the naming styles.

Tags vs Elements vs Attributes

Tags are the starting and ending parts of an HTML or XHTML element. They begin with < symbol and end with > symbol. Whatever written inside < and > are called tags.

```
<b> </b>
```

Elements enclose the contents in between the tags. They consist of some kind of structure or expression. It generally consists of a start tag, content and an end tag.

```
<b>This is the content.</b>
```

Attribute is used to define the character of an HTML element. It is always placed in the opening tag of an element. It generally provides additional styling (attribute) to the element.

```
<p align="center">This is a paragraph.</p>
```

Q8. Which Ajax method is used to exchange data with a server, using a modern browser?

Answers:

- request-XML
- XMLHttpRequest
- ActiveXObject
- responseXML

Explanation:

All modern browsers support the XMLHttpRequest object.

The XMLHttpRequest object can be used to exchange data with a server behind the scenes so that it is possible to update parts of a web page, without reloading the whole page.

Q9. A markup language is a _ -readable language that _ text so that the computer can _ that text.

Answers:

- processor; complies; process
- system; stores; retrieve
- non; processes; format
- human; annotates; manipulate

Explanation:

A markup language is a computer language that uses tags to define elements within a document.

It is human-readable, meaning markup files contain standard words, rather than typical programming syntax.

Q10. What is this code an example of?

```
<x a="x" a="y"></x>
```

Answers:

- improperly named element;
- self-closing tag;
- null element;
- incorrect XML syntax.

Explanation:

Attribute names are unique per element.

If you need to have multiple bits of data under the same name, then the usual solutions are either a space separated list or child elements.

```
<x a="x y" />
```

or

```
<x>
    <a>x</a>
    <a>y</a>
</x>
```

Link: The same attribute multiple times to an Element Tag in XML

Q11. XML provides a framework for specifying markup languages, while HTML is a predefined markup language. What is applicable to XML and not HTML?

Answers:

- It is mandatory to use closing tags with XML
- It is important for an XML document to be well formed
- XML elements start with an opening tag in angle brackets, such as <p>
- XML syntax uses tags, elements, and attributes

Explanation:

An XML element is required to have a closing tag. With XML it is illegal to omit the closing tag. An XML element can be closed in two possible ways; non-empty closed element and empty closed element.

```
<anonemptyelement>
      An element value.
</anonemptyelement>
```

```
<anemptyelement />
```

Link: XML Closing Tag

Q12. What is the last step in extending XHTML modules?

Answers:

- The last step is to complete the extension of XHTML compound documents and make sure the documents adhere to the defined namespaces.
- The last step is to create the DTD for the XHTML extension, which references both the XHTML modules and the new modules.
- The last step is to run the XHTML extension through the XSLT processor, which will properly format it.
- The last step is to verify that the XHTML is well formed and valid, and compatible with most browsers.

Q13. In an XML DTD ATTLIST declaration, which default value is used to indicate that the attribute does not have to be included?

- #DEFAULT
- #OPTIONAL
- #IMPLIED
- #FIXED

Q14. How does the XML DOM present an XML document?

Answers:

- as a set of objects
- as a tree structure
- as an array of nodes
- as a dynamic program

Explanation:

The XML DOM defines a standard way for accessing and manipulating XML documents. It presents an XML document as a tree-structure.

Link: https://www.w3schools.com/xml/dom_intro.asp

Q15. You are working with an XML document that uses an XML schema. How do you specify that an element can appear multiple times inside its parent element?

Answers:

- Set the maxOccurs attribute to a large number, such as 1.000
- Set the maxOccurs attribute to 0
- Set the maxOccurs attribute to undefined
- Set the maxOccurs attribute to unbounded

Explanation:

XML can contain an array with varying numbers of elements. An array with a varying number of elements is represented in the XML schema by using the minOccurs and maxOccurs attributes on the element declaration:

- The minOccurs attribute specifies the minimum number of times that the element can occur. It can have a value of 0 or any positive integer;
- The maxOccurs attribute specifies the maximum number of times that the element can occur. It can have a value of any positive integer greater than or equal to the value of the minOccurs attribute.

  It can also take a value of unbounded, which indicates that no upper limit applies to the number of times the element can occur.

- The default value for both attributes is 1.

Link: https://www.ibm.com/docs/en/cics-ts/5.3?topic=assistant-variable-arrays-elements

Q16. The <xsl:with-param> element defines the value of a parameter to be passed into a template. It can be used within which elements?

Answers:

- <xsl:apply-templates> and <xsl:call-template>
- <xsl:param> and <xsl:processing-instruction>
- <xsl:template> and <xsl:transform>
- <xsl:include> and <xsl:variable>

Explanation:

The <xsl:with-param> element defines the value of a parameter to be passed into a template.

```
<xsl:with-param
name="name"
select="expression">

  <!-- Content:template -->

</xsl:with-param>
```

The value of the name attribute of <xsl:with-param> must match a name in an <xsl:param> element (the <xsl:with-param> element is ignored if there is no match).

The <xsl:with-param> element is allowed within <xsl:apply-templates> and <xsl:call-template>.

Link: XSLT <xsl:with-param>

Q17. You are checking someone else's XML document for errors. You notice that the prolog does not have a closing tag. What do you do?

Answers:

- Remove the prolog to make sure that the XML document will be properly processed across all platforms;
- Leave it alone, because the prolog does not require a closing tag;
- Move the prolog to an external file so that the XML document only has elements with closing tags;
- Add a closing tag, as all XML elements must have a closing tag.

Explanation:

The prolog line is optional and always at first on.

Q18. Which statement is not true about XML?

Answers:

- XML is flexible and customizable
- XML can be used to store data
- XML is independent of the Operating System
- XML is a replacement for HTML

Explanation:

HTML and XML are related to each other, where HTML displays data and describes the structure of a webpage, whereas XML stores and transfers data.

HTML is a simple predefined language, while XML is a standard language that defines other languages.

Link: Difference Between HTML and XML

Q19. In an XML DTD ATTLIST declaration, which tokenized attribute type is used to specify multiple ID values?

Answers:

- ENTITIES
- IDREFS
- IDS
- IDSETS

Q20. You want to convert a large XML file into CSV format. You did not create the XML file, so you are not familiar with all of the syntax. What will help you get the best insight into the file contents?

Answers:

- XSLT
- DOM
- AJAX
- XSD

Explanation:

An XSD defines the structure of an XML document. It specifies the elements and attributes that can appear in an XML document and the type of data these elements and attributes can contain. This information is used to verify that each element or attribute in an XML document adheres to its description.

Link: XML Schema Definition

Q21. In an XML DTD, attributes are declared with an ATTLIST declaration. You need to validate the color attribute for element <car> against a fixed list of values. Which is the correct declaration?

Answers:

- <!ATTLIST car color (red|white|blue|black) black>
- <!ATTLIST car color (red|white|blue|black) #REQUIRED>
- <!ATTLIST car color (red|white|blue|black) #FIXED>
- <!ATTLIST car color (red|white|blue|black)>

Q22. The main ways to control the display of XML documents are with Cascading Style Sheets and **XSL** Extensible Styles Language. What is an advantage of CSS over XSL?

Answers:

- CSS is a complete programming language with more powerful syntax
- With CSS, the same element can be processed multiple times
- CSS allows you to reformat data into completely new structures
- CSS is easier to learn, use, and maintain

Explanation:

The reason is that CSS is much easier to use, easier to learn, thus easier to maintain and cheaper. There are WYSIWYG editors for CSS and in general there are more tools for CSS than for XSL.

But CSS's simplicity means it has its limitations.

Link: CSS & XSL

Q23. Which type of DTD declaration is this code an example of?
<!DOCTYPE abc SYSTEM "file/file.dtd">

Answers:

- Linked
- Internal
- External
- Structured

Q24. The purpose of an XML schema is to define the building blocks of an XML document. Which option best describes the building blocks of an XML document?

Answers:

- Header files, function declarations, global variables with their data types, and system library folder location
- Namespace declaration, processor type, markup references, and encoding specification.
- The document's elements and attributes, their data types and default values, and the number and order of child elements.
- XML entity definitions, XSLT and cascading style sheets, DOM specification, and CDATA assignments.

Q25. An XHTML **DTD** Document Type Definition describes the allowed syntax and grammar of XHTML markup. Which is not one of the formal DTDs used in XHTML 1.0?

Answers:

- Frameset
- Transitional
- Basic
- Strict

Q26. You are working with the following XML code snippet. You have this line in your XSLT code <xsl:value-of-select="//car/make"/>. What does it display?

```
<cars>
    <car>
        <make>Cadillac
            <model>Escalade</model>
            <price year="2007">$20,000</price>
        </make>
    </car>
</cars>
```

Answers:

- Cadillac
- Cadillac Escalade
- Cadillac Escalade 20000
- Cadillac Escalade $20,000

Explanation:

XSL eXtensible Stylesheet Language is a styling language for XML.

The <xsl:value-of> element is used to extract the value of a selected node.

Link: XSLT <xsl:value-of> Element

Q27. You need to display the list of cars in the code snippet below in a column format, with a counter column for each row. Which XPath function do you use for the counter?

```
<cars>
    <car><make>Cadillac</make> <model>Escalade</model> <year>2007</year></car>
    <car><make>Ford</make> <model>Mustang</model> <year>1968</year></car>
    <car><make>Mercedes</make> <model>C-Class</model> <year>1999</year></car>
</cars>
```

Answers:

- format-number()
- id()
- count()
- position()

Explanation:

XPath uses path expressions to select nodes or node-sets in an XML document. The node is selected by following a path or steps.

The “position” function returns a number equal to the context position from the expression evaluation context.

```
<xsl:template match="//a[position() = 5]">
 <!-- this template matches the fifth a element
         anywhere in the document. -->
</xsl:template>
```

Note: count() returns the total number of nodes (3), while position() returns the 0-based index of each node.

Link: https://developer.mozilla.org/en-US/docs/Web/XPath/Functions/position

Q28. You are working with this XML code snippet from the XML document cars.xml.
You need to return the information about the cars built after the year 2000, as an ordered list, starting with the most recent. What does your XQuery look like?

```
<cars>
    <car><make>Cadillac</make> <model>Escalade</model ><year>2007</year></car>
    <car><make>Cadillac</make> <model>Escalade</model> <year>2011</year></car>
    <car><make>Ford</make> <model>Mustang</model> <year>1968</year></car>
    <car><make>Ford</make> <model>Mustang</model> <year>1998</year></car>
    <car><make>Mercedes</make> <model>C-Class</model> <year>1999</year></car>
    <car><make>Mercedes</make> <model>C-Class</model> <year>2009</year></car>
</cars>
```

Answers:

```
<ul>
{
    for $x in doc("cars.xml")/cars/car
    where $x/year>2000
    order by $x/year descending
    return <li>{$x}</li>
}
</ul>
```

```
<ol>
{
    for $x in doc("cars.xml")/cars/car
    where $x/year>2000
    order by $x/year desc
    return <li>{data($x)}</li>
}
</ol>
```

```
<ul>
{
    for $x in doc("cars.xml")/cars/car
    where $x/year>2000
    order by $x/year
    return <li>{$x}</li>
}
</ul>
```

```
<ol>
{
    for $x in doc("cars.xml")/cars/car
    where $x/year>2000
    order by $x/year descending
    return <li>{data($x)}</li>
}
</ol>
```

Link: XPath Syntax

Q29. The readyState property holds the status of the XMLHttpRequest. Which is NOT a valid status?

Answers:

- 4 (DONE)
- 3 (LOADING)

- 1 (PROCESSING)
- 0 (UNSENT)

Explanation:

Value	State	Description
0	UNSENT	Client has been created. open() not called yet.
1	OPENED	open() has been called.
2	HEADERS_RECEIVED	send() has been called, and headers and status are available.
3	LOADING	Downloading; responseText holds partial data.
4	DONE	The operation is complete.

Link: XMLHttpRequest.readyState

Q30. You are working with an XML document that uses an XML schema.
How can you extend the document with elements NOT specified by the schema?

Answers:

- Use the <any> element
- Use the <redefine> element
- Use <xs:extension>
- Specify the new elements in the schema

Explanation:

The <any> element enables us to extend the XML document with elements not specified by the schema!

The following example is a fragment from an XML schema called "family.xsd". It shows a declaration for the "person" element. By using the <any> element we can extend (after <lastname>) the content of "person" with any element:

```
<xs:element name="person">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="firstname" type="xs:string"/>
      <xs:element name="lastname" type="xs:string"/>
      <xs:any minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Q31. You are working with the following XML code snippet.
Which XPath expression produces C-Class?

```
<cars>
    <car><make>Cadillac</make><model>Escalade</model>
        <price year="2007">20000</price></car>
    <car><make>Ford</make><model>Mustang</model>
        <price year="2008">17000</price></car>
    <car><make>Mercedes</make><model>C-Class</model>
        <price year="2009">24000</price></car>
```

```
</cars>
```

Answers:

- /car[price>20000]/make/model
- /car[price>=20000 and @year>=2009]/make/model
- //car[price>=20000 and @year>2008]/model
- /cars/car[price>=20000 and year>2008]/model

NOTE: XPather shows that all answers are incorrect. Report the question.

Explanation:

The correct answer is based on the syntax XPath Examples

Q32. You are working with an XML document that uses an XML schema. How do you ensure that an attribute must be specified for its corresponding element?

Answers:

- Set the type attribute to xs:required
- Set the use attribute to required
- Set the minLength attribute to 1
- Set the minOccurs attribute to 1

Explanation:

Attributes are optional by default. To specify that the attribute is required, use the "use" attribute:

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

Q33. You are working with the following XML code snippet. What do you need to include in your XSLT code to display Mercedes, Cadillac, Ford?

```
<cars>
    <car><make>Cadillac</make><model>Escalade</model>
        <price year="2007">20000</price></car>
    <car><make>Ford</make><model>Mustang</model>
        <price year="2008">17000</price></car>
    <car><make>Mercedes</make><model>C-Class</model>
        <price year="2009">24000</price></car>
</cars>
```

Answers:

```
<xsl:sort select="make" />
```

```
<xsl:sort select="model" />
```

```
<xsl:sort select="car" />
```

```
<xsl:sort select="price" />
```

Explanation:

A trick question. The <xsl:sort> will sort the output in ascending (alphabetical for strings) order by default. The select tells which tag to use for sorting.

- If we use select="make" or select="year" we get the order Cadillac, Ford, Mercedes
- If we use select="price" we get Ford, Cadillac, Mercedes

- If we use select="model" we get Mercedes, Cadillac, Ford

Q34. What is the correct syntax for comments in XQuery?

Answers:

```
/* */
```

```
<!-- -->
```

```
//
```

```
(: :)
```

Explanation:

XQuery is to XML what SQL is to databases.

XQuery comments, delimited by (: *your comment* :), can be added to any query to provide more information about the query itself. These comments are ignored during processing.

XQuery comments can contain any text, including XML markup. For example:

```
(: This query returns the <number> children :)
```

Q35. Which DOM node type may NOT have the “EntityReference” node type as one of its child nodes?

Answers:

- Element
- Document
- EntityReference
- DocumentFragment

Explanation:

The Document Object Model is a platform and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure, and style of a document.

The XML DOM defines a standard way for accessing and manipulating XML documents. It presents an XML document as a tree-structure.

The following table lists the different W3C node types, and which node types they may have as children:

Node Type	Description	Children
Document	Represents the entire document (the root-node of the DOM tree)	Element (max. one), ProcessingInstruction, Comment, DocumentType
DocumentFragment	Represents a "lightweight" Document object, which can hold a portion of a document	Element, ProcessingInstruction, Comment, Text, CDATASection, **EntityReference**
EntityReference	Represents an entity reference	Element, ProcessingInstruction, Comment, Text, CDATASection, **EntityReference**

Element	Represents an element	Element, Text, Comment, ProcessingInstruction, CDATASection, **EntityReference**

Link: XML DOM Node Types

Q36. XHTML modules can be extended by adding elements, attributes, modifying content models, or some combination of these. What does a proper implementation of an XHTML module require?

Answers:

- The implementation of an XHTML module requires an extension module and a validation module that ensures that the XHTML is well formed and valid; otherwise the extended instances aren't formally XHTML.
- The implementation of an XHTML module requires a definitions module and a constraint module that specifies syntax rules and uses the parameter entities declared in the definitions module.
- The implementation of an XHTML module requires a qualified name module and a declaration module that holds the element, element attribute, and content model declarations.
- The implementation of an XHTML module requires a namespace module that holds the element, element attribute, and content model declarations, and a parameter module that uses the entities declared in the namespace module.

Q37. The <xsl:namespace-alias> element is used to replace a namespace in the style sheet with a different namespace in the output. Which XSLT element needs to be its parent node?

Answers:

```
<xsl:namespace>
```

```
any valid element
```

```
root element
```

```
top-level element in the corresponding namespace
```

Explanation:

The <xsl:namespace-alias> element is a rarely used device that maps a namespace in the stylesheet to a different namespace in the output tree. The most common use for this element is in generating a stylesheet from another stylesheet.

The <xsl:namespace-alias> is a top-level element, and must be a child node of <xsl:stylesheet> or <xsl:transform>.

Syntax:

```
<xsl:namespace-alias stylesheet-prefix=NAME result-prefix=NAME />
```

Q38. XML is a markup language, not a programming language. What makes XML not qualify to be a programming language?

Answers:

- XML is too flexible and does not have enough reserved keywords
- XML contains only data and not any processing instructions
- XML does not perform any computation or algorithms
- XML does not have specialized syntax rules

Explanation:

XML does not qualify to be a programming language as it does not perform any computation or algorithms. It is usually stored in a simple text file and is processed by special software that is capable of interpreting XML.

Q39. What is true about these elements in XQuery?

```
<cars>
    <car><make>Cadillac</make><model>Escalade</model><year>2007</year></car>
    <car><make>Cadillac</make><model>Escalade</model><year>2011</year></car>
    <car><make>Ford</make><model>Mustang</model><year>1968</year></car>
    <car><make>Ford</make><model>Mustang</model><year>1998</year></car>
    <car><make>Mercedes</make><model>C-Class</model><year>1999</year></car>
    <car><make>Mercedes</make><model>C-Class</model><year>2009</year></car>
</cars>
```

Answers:

- Elements <make> and <model> are ancestors of <year>
- Elements <make> and <model> are children of <cars>
- Elements <make> and <model> are siblings
- Elements <car> and <cars> are parents of <make> and <model>

Q40. Which is a valid CSS section for this XML code snippet?

```
<cars>
    <car><make>Cadillac</make><model>Escalade</model><year>2007</year></car>
    <car><make>Ford</make><model>Mustang</model><year>1968</year></car>
    <car><make>Mercedes</make><model>C-Class</model><year>1999</year></car>
</cars>
```

Answers:

```
cars {
  display: block;
}
car(make),
car(model),
car(year) {
  display: inline;
  padding-top: 0.5em;
}
```

```
car,
cars {
  display: block;
}
make,
model,
year {
  display: inline;
  padding-top: 0.5em;
}
```

```
cars {
  display: block;
```

```
}
car.make,
car.model,
car.year {
  display: inline;
  padding-top: 0.5em;
}
```

```
cars {
  display: block;
}
car#make,
car#model,
car#year {
  display: inline;
  padding-top: 0.5em;
}
```