University of Cape Town

Department of Computer Science

CSC3003s Class Test 1

2006

Marks **: 35**

Time **: 45 minutes**

Instructions:

- Answer all questions.
- Show all calculations where applicable.

Question 1: Core XML [10]

a) Why do we use Unicode as an underlying representation for XML instead of ASCII? [1]

it can represent a larger code space/more languages/more characters [1]

b) What is the advantage of using UTF-8 over UCS-4? [2]

it is a variable length encoding [1] and common characters could use fewer bytes [1]

c) Under what circumstances would one use UTF-16 instead of UTF-8? What would happen if UTF-8 was used anyway? [2]

when large code value characters, that take more than 2 bytes in UTF-8, are common (e.g., to represent Chinese) [1] in this case some characters could use just 2 bytes in UTF-16 while more than 2 in UTF-8 [1]

d) Give one example of a validity test that DTDs cannot express. Show with an example how this test would be encoded in XML Schema? [3]

a precise number of occurrences for an element [1]

<element name="test" type="string" minOccurs="2" maxOccurs="5"/> [2]

e) What is a byte order marker and where is it found in an XML document? [2]

it is a sequence of bytes that indicates the endianness for a Unicode data stream [1] it is found at the very beginning of the XML document [1]

Question 2: XML Languages [15]

a) Give one reason to use a SAX parser instead of a DOM parser. [1]

it can process larger files since the tree is not built in memory [1]

b) Give one reason to use a DOM parser instead of a SAX parser. [1]

it can provide efficient random access to nodes of the tree without repeated parsing [1]

c) Suppose the following DOM statement returns the value ‘123’. Write a sample XML document that will result in this return value. Write an XPath expression that is equivalent to the DOM statement. Hint: Remember that **item** is zero-based. [3]

documentElement → getElementsByTagName (‘store’) → item(1) → firstChild → nextSibling → getElementsByTagName (‘product’) → item(0) → firstChild → data

```
<stores>
  <store/>
  <store>
    <aisle>
      <product>123</product>
    </aisle>
  </store>
</stores>
```
[2] Minus one for each major error (note: this is not the only soln)

stores/store[1]/aisle/product   [1]

d) Answer the following questions based on this piece of XML:

```
<test xmlns="http://ns1">
   <date type="iso8601">2006-08-23</date>
   <class>
      <name>CSC3003s</name>
      <venue>Jameson Hall</venue>
   </class>
</test>
```

Assume that the **class** element is infinitely repeatable and required and that **name** and **venue** must both appear exactly once each within each **class**.

i. Write an XML Schema complexType type definition testType corresponding to the content of the test element and its descendents. [4]

```
<complexType name="testType">
  <sequence>
    <element name="date">
      <complexType><simpleContent>
          <extension base="string">
            <attribute name="type" type="string"/>
          </extension>
      </simpleContent></complexType>
    </element>
    <element name="class" maxOccurs="unbounded">
      <complexType><sequence>
          <element name="name" type="string"/>
          <element name="venue" type="string"/>
      </sequence></complexType>
    </element>
  </sequence>
</complexType>
```

[4] Minus one for each major error (incorrect attribute, incorrect structure, missing elements, etc.)

ii. Write an XSLT template, using the stylesheet outline provided, to convert the **test** node into the following structure [4]

```
<class xmlns="http://ns2">
   <who>CSC3003s</who>
   <test>
      <when>
         <type>iso8601</type>
         <date>2006-08-23</date>
      </when>
      <where>Jameson Hall</where>
   </test>
</class>
```

Assume your template will be placed within the following stylesheet:

```
<xsl:stylesheet version="1.0"
   xmlns:xsl=http://www.w3.org/1999/XSL/Transform
   xmlns:source=http://ns1
   xmlns:target="http://ns2">
...
</xsl:stylesheet>
```

```
<xsl:template match="source:test">
  <target:class>
    <target:who><xsl:value-of select="source:class/source:name"/></target:who>
    <target:test>
      <target:when>
        <target:type><xsl:value-of select="source:date/@type"/></target:type>
        <target:date><xsl:value-of select="source:date"/></target:date>
      </target:when>
      <target:where><xsl:value-of select="source:class/source:venue"/></target:where>
    </target:test>
  </target:class>
</xsl:template>
```

[4] Minus one for each major error (incorrect XPaths, incorrect structure, missing elements, etc.)

e) Briefly discuss 2 advantages of using XML to represent XML Schema. [2]

XML tools can be used to manipulate the schema definition [1]

schemas can themselves be validated using XML Schema [1]

Question 3: Database Systems [10]

a) Explain briefly

EITHER

i. the terms "physical data independence" and "logical data independence"

OR

ii. **any two (2)** of the letters in "ACID" properties of database transactions

[4]

MARKS: 2 per correct answer, or 1 per almost-correct answer.

i. Physical data independence: physical storage details can change without needing any changes at the logical level (also accept: physical storage details are transparent at the logical/schema level)

Logical data independence: schema/logical database design can change without needing any changes to application programs (also accept: schema changes are transparent at the application program level, or if examples are given e.g. can add/remove/redefine attributes without needing to rewrite application programs)

ii. Atomicity: either all of a transaction is applied to a database, or nothing (a partially complete transaction will never have any effect on the database)

Consistency: database changes preserve the correctness of information

Isolation: even when multiple transactions run concurrently, the effect will be the same as if they had run one after the other

Durability: once a transaction has committed, it is guaranteed that the changes it has made to the database will be permanently recorded, even if there is a subsequent system failure.

b) Consider the schema below, which comes from a video store database:

CLIENT (ID, Name, Telno)	// client ID, name and telephone number
VIDEO (Num, Title, Year, Length)	//video no., film title, year made, duration
RENTAL (ID, Num, Day)	// client ID, video no., date taken out
STARS (Num, Actor)	// video no. actor/actress name

Give SQL statements for **each** of the following queries:

i. Give the name and telephone number of every client who has rented the video entitled "Jaws", in alphabetical order of customer name.
ii. Give the ID of all clients who have not rented any videos (i.e. who do not appear in the RENTAL relation).
iii. For each "big customer", give the client ID and the number of videos they have rented. A "big customer" is someone who has rented more than 10 videos.

[6]

MARKS: 2 per completely correct answer, or 1 if any non-trivial part of the answer is correct

i.SELECT Name, Telno FROM Client, Rental, VIdeo

WHERE Client.ID = Rental.ID AND Rental.Num = Video.Num

AND Video.Title = "Jaws"

ORDER BY Name

ii. SELECT ID FROM Client WHERE ID NOT IN (SELECT ID FROM Rental)

or

(SELECT ID FROM Client) MINUS (SELECT ID FROM Rental)

iii. SELECT ID, count(Num) AS number-rented FROM RENTAL GROUP BY ID

HAVING count(Num) > 10 // or HAVING number-rented > 10