University of Cape Town

Department of Computer Science

CSC3003s Class Test 1 [Retest]

2006

Marks	**: 35**
Time	**: 45 minutes**

Instructions:

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

Question 1: Core XML [10]

a) What two characteristics must a well-formed XML document have? Give examples of non-well-formed documents violating each of these characteristics. [4]

single root [1]

<test/><test/> [1]

properly nested matching start/end tags [1]

<test><test1></test></test1> [1]

b) What characteristic must a valid XML document have? [1]

must conform to a formal definition [1]

c) When is it necessary to use an entity for the quotation mark character? Why is it otherwise not always necessary? [2]

in attribute values [1] otherwise not necessary because there is no ambiguity [1]

d) Convert the following to use only default namespaces. [3]

```
<uct:uct xmlns:uct="http://ns1" xmlns:dc2="http://ns3">
   <dc1:title xmlns:dc1="http://ns2">data1</dc:title>
   <dc2:title>data2</dc2:title>
</uct:uct>
```

< uct xmlns="http://ns1">
<title xmlns="http://ns2">data1</title>
<title xmlns="http://ns3">data2</title>
</uct>
Marks: [3] one for correct namespace and prefixes for each of the 3 elements

Question 2: XML Languages [15]

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

documentElement → getElementsByTagName (‘book’) → item(0) → getElementsByTagName (‘author’) → item(1) → getElementsByTagName (‘name’) → item(2) → firstChild → data

```
<book>
  <author/>
  <author>
    <name/>
    <name/>
    <name>james</name>
  </author>
</book>  [2] Minus one for each major error (note: this is not the only soln)
```

book/author[2]/name[3]   [1]

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

```
<furniture xmlns="http://ns1">
   <room name="lounge">
      <piece>couch</piece>
      <piece>coffee table</piece>
      <piece>side table</piece>
   </room>
</furniture>
```

Assume that **room** is infinitely repeatable and at least one **room** is required and at least one **piece** is required in every **room**, but at most 12 **piece**s can appear in a single room.

i. What does the XPath expression //room[1]/piece[2] return? [1]

the node corresponding to “coffee table”

ii. Write an XML Schema complexType type definition **furnitureType** corresponding to the content of the **furniture** element and its descendents. [4]

```
<complexType name="furnitureType">
  <sequence>
    <element name="room" maxOccurs="unbounded">
      <complexType>
        <sequence>
          <element name="piece" type="string" maxOccurs="12"/>
        </sequence>
        <attribute name="name" type="string" use="required"/>
      </complexType>
    </element>
  </sequence>
</complexType>
```

Marks[4]: Minus one for each major error (bounds, attribute, nesting, etc.)

iii. Write an XSLT template, using the stylesheet outline provided, to convert the **furniture** node into the following structure. [5]

```
<lounge xmlns="http://ns2">
   <furniture piece="couch"/>
   <furniture piece="coffee table"/>
   <furniture piece="side table"/>
   <furniture piece="item in every room"/>
</lounge>
```

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:furniture">
  <xsl:element name="target:{source:room/@name}">
    <xsl:for-each select="source:room/source:piece">
      <target:furniture piece="{.}"/>
    </xsl:for-each>
    <target:furniture piece="item in every room"/>
  </xsl:element>
</xsl:template>
```

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

c) Explain how XSLT is used with FO to create documents in arbitrary non-XML formats from arbitrary XML data. [2]

first the original XML data is translated into the FO language using an XSLT stylesheet [1], then a FO processor converts the FO documents into the target non-XML format [1]

Question 3: Database Systems [10]

a) Name any 2 components of a typical Database Management System (DBMS), and explain briefly what their role/function is within the DBMS. [4]

MARKS: 1 mark for each correct component name, 1 mark for each correct function

Query processor (any valid explanation of some/all that it does)

Recovery Manager (ensures committed transactions are redone after system failure)

Buffer manager (manages memory usage to reduce disk IO)

File manager (any valid role, e.g. manages retrieval and updating of data and indexes)

Transaction manager (ensures ACID properties of transactions)

Authorisation and integrity manager (any valid role e.g. enforces privacy & security)

Storage Manager (also OK, collective name for last 4 components above)

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 **ANY THREE (3)** of the following queries:

i. Give the title of all videos that were made before the year 2000, along with the ID of clients who rented them, in alphabetical order of video title.

ii. Give the ID of all videos that do not have any actors (i.e. that do not appear in the STARS relation).

iii. For each actor, give the actor name along with the number of "longs films" they have starred in. A "long film" is one that is longer than 120 minutes duration.

iv. Give the titles of all videos that are longer than average duration (e.g. if the average length of video in the VIDEO relation is say 95 minutes, then give the titles of all videos longer than 95 minutes duration).

[6]

MARKS: 2 marks for each completely correct answer, 1 mark for each non-trivial part of an answer that is correct

```
1.SELECT Title, ID FROM Video, Rental
WHERE Video.Num = Rental.Num and Video.Year like "19__"
ORDER BY Title
or
SELECT Title, ID FROM Video, Rental
WHERE Video.Num = Rental.Num and Video.Year < 2000
ORDER BY Title
2.SELECT Num FROM Video WHERE Num NOT IN
(SELECT Num from Stars)
or
(SELECT Num FROM Video) MINUS (SELECT Num FROM Stars)
3.SELECT Actor, COUNT(Num) AS num-long-films FROM Stars, Video
WHERE Video.Length > 120 and Stars.Num = Video.Num
GROUP BY Actor
4.SELECT Title FROM Video where Length >
(SELECT AVG (Length) FROM Video)
```