UNIVERSITY OF EDINBURGH

COLLEGE OF SCIENCE AND ENGINEERING

SCHOOL OF INFORMATICS

INFR11015 APPLIED DATABASES

Friday 29th April 2016

09:30 to 11:30

INSTRUCTIONS TO CANDIDATES

Answer any TWO questions.

All questions carry equal weight.

CALCULATORS MAY BE USED IN THIS EXAMINATION

MSc Courses

Convener: F. Keller
External Examiners: A. Burns, S. Denham, P. Healey, T. Norman

THIS EXAMINATION WILL BE MARKED ANONYMOUSLY

[*25 marks*]

1. (A) [*Application of Knowledge: 5 * 1 = 5 marks*]

Well-Formed XML is defined in terms of a context-free grammar and 10 well-formedness constraints. For each of the lines below list all the violations against well-formed XML. Say for each violation whether it is due to the grammar, or to one of the well-formedness constraints.

```
<a href="abc"a></a><b></b href=cde>
<h1><table></h1></tabel>
<b co="15", col="16">< a ></ a ></b>
Text<tag a="1">Text<tag a="1">Text<tag a="1"></tag></tag><tag/></tag>
<temp="20">Values of <50 are allowed.</temp><a c2="",c1="3",c2="3"><a/>
```

(B) [*Bookwork: 6 * 1.5 = 9 marks*]

Document Type Definitions (DTDs) are part of XML and allow to define types of documents. Explain these main features and give an example for each feature:

(1) entity declarations
(2) required and implied attributes
(3) enumerated attribute type
(4) ID and IDREF attributes
(5) mixed content
(6) deterministic regular expressions

(C) [*Application of Knowledge: 4 marks*]

There are several errors in the following XML document. List and explain all of them.

```
<!DOCTYPE a [
<!ELEMENT a (#PCDATA | b | c | d)*>
<!ELEMENT c (b)*>
<!ELEMENT b (#PCDATA)>
  <!ATTLIST b att CDATA #REQUIRED>
  <!ENTITY hi "Hello">
  <!ENTITY hi1 "&hi1;&hi;">
]>
<a>ab&hi1;cde<b></b><d></d><c>a</c></a>
```

(D) [*Bookwork and Application of Knowledge: 7 marks*]
Deterministic regular expressions are part of DTDs and hence are part of the XML specification.

(1) Explain the determinism restriction. [1]
(2) Give an example of a regular expression that is *not* deterministic. [0.5]
(3) Explain why this restriction is useful. How can a string of length n be matched against a *deterministic* regular expression of size m? What is the time complexity? In contrast, what is the time complexity of matching a string against an *arbitrary unrestricted* regular expression? [2]
(4) Draw Glushkov automata for the regular expressions `(a(a|b)b)*a*` and `(a(a|b)b)*b*`. Is any of these automata deterministic? [3.5]

2. (A) Schemas. [*Bookwork: 4 * 1.5 = 6 marks*] [*25 marks*]

(1) A functional dependency means that some columns determine the values of other columns. Consider a table with columns Street, City, and ZipCode. List all the functional dependencies for this table. (this is for the US, where a ZipCode does not determine a street).

(2) Explain what is a superkey, what is a key, and what is a primary key.

(3) List all keys for a table of biddings of an online bidding system (where the same bidder can make multiple bids at a time) with column names ItemID, BidderID, Time, and Amount.

(4) Explain what a multi-valued dependency is, and give an example of one (that is not a functional dependency).

(B) Normal Forms. [*Bookwork and Application of Knowledge: 11 marks*]

(1) Redundancy with respect to a functional dependency means that facts of the dependency appear more than once in the table. Redundancy causes update anomalies. Using an example of a table with redundancy, explain each of the three update anomalies.

(2) Explain the Boyce-Codd-normal form (BCNF). Give an example of a table that is *not* in BCNF, then show how to split it into tables that are in BCNF.

(3) Show on an example how BCNF removes redundancy.

(4) Prove that a table in BCNF contains no redundancies with respect to any functional dependency.

(C) SQL queries. [*Application of Knowledge: 8 * 1 = 8 marks*]

R =

1
1
2
3
3
3
5

Consider table R with single integer-column `a` shown on the left. Note that ROUND(1.77,1) = 1.8, i.e., ROUND(x,k) rounds x to k decimals. Determine the results to the following queries.

(1) `SELECT ROUND( AVG(a), 1 ) from R;`
(2) `SELECT SUM( DISTINCT a ) from R;`
(3) `SELECT COUNT( * ) from R r1, R r2;`
(4) `SELECT COUNT( * ) from R r1, R r2 WHERE r1.a=r2.a;`
(5) `SELECT * from R r1, R r2 WHERE r1.a>2*r2.a;`

Write SQL queries that select from table R the following:

(6) all elements that are larger than the average.

(7) A histogram showing in the first column each distinct value of R, and in the second column the number of times that value appears in R.

(8) All elements (first column) together with their distance from the average (second column) ordered by that distance. Thus, the first row will be (3, 0.4286) and the last row will be (5,2.4286). You can use the ABS(.) function to compute the absolute value (i.e., ABS(-3)=3).

3. (A) Ranking using TF/IDF. [*Application of Knowledge: 15 marks*] [*25 marks*]

(1) Explain the idea of TF/IDF ranking. What is the IDF of a term that occurs in every document? [2.5]

(2) Consider these five documents: [12.5]

D1 = "If it walks like a duck and quacks like a duck, it must be a duck."

D2 = "Beijing Duck is mostly priced for the thin, crispy duck skin with authentic versions of the dish serving mostly in the skin."

D3 = "Bugs' ascension to stardom also prompted the Warner animators to recast Daffy Duck as the rabbit's rival, intensely jealous and determined to steal back the spotlight while Bugs remained indifferent to the duck's jealousy, or used it to his advantage. This turned out to be the recipe for the success of the duo."

D4 = "6:25 PM 1/7/2007 blog entry: I found this great recipe for Rabbit Braised in Wine on cookingforengineers.com."

D5 = "Last week Li has shown you how to make the Sechuan duck. Today we'll be making Chinese dumplings (Jiaozi), a popular dish that I had a chance to try last summer in Beijing. There are many recipes for Jiaozi."

We assume case insensitivity and are **only interested** in the words:

bejing, dish, duck, rabbit, recipe, roast

Make a table of the term frequencies of these words (rows are the words, columns are the documents). Then normalise the table by dividing column-wise by the maximum. Compute the IDF of each word as the logarithm (base 10) of the inverse document frequency. Finally, multiply the normalised term frequencies by the IDFs to obtain the TFIDF table. Compute the cosine similarity between the query Q = "Beijing duck recipe" and each document.

(B) String Matching by Automata. [*Application of Knowledge: 10 marks*]

(1) Draw the Matching-Automaton for the string "nano". [2.5]

(2) Run this automaton over the string "banananona" by giving the sequence of states that the automaton reaches at each position of the string. [2.5]

(3) Draw an automaton for "nan*o", where "*" denotes a wildcard, i.e., it matches any character. Use a transition labeled x for the wildcard character, denoting your automaton will remember the actual letter in variable x. On the next transitions, you can write, e.g., "a, x=b", meaning that the current letter is "a" and the wildcard character was "b". [2.5]

(4) Describe the Boyer-Moore string matching algorithm. It is sufficient to only describe the "Bad Character Rule" Show how to apply it with search string "nano" over the string "banananona". How many comparisons of characters are applied in total? [2.5]