CS541 Fall 2007 Qualifying Exam, 10 December 2007

Profs. Chris Clifton and Elisa Bertino

Answer two of the following three questions (note that each question has multiple parts.) If you answer more than two, circle the number of the ones that you feel you answered best and would like us to use to evaluate your exam - knowing what you know well is part of the exam. You have one hour.

1 Uncertainty in Databases

Managing data where we aren't certain of the exact values is receiving increasing attention. Assume that we model uncertain data using a Gaussian Probability Density Function:

$$\frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{1}$$

where μ is the mean and σ^2 is the variance.

Queries return tuples that satisfy the query with probability greater than a given threshold; for example:

```
SELECT * FROM table
WHERE 2.5 < table.value AND table.value < 3.5
WITH PROBABILITY > 0.7;
```

Given a tuple, we could evaluate if it satisfies the above query by taking the integral of equation 1 from 2.5 to 3.5 and seeing if the value exceeded 0.7 (of course, there are more efficient approximations, but you won't need to worry about that to answer this question.)

1.1 What is unchanged?

Describe a technology used to build a standard relational database that we can reuse relatively unchanged. Explain why/how we would use it.

1.2 What must we change?

Describe a technology from standard relational database that would no longer be applicable. Explain why we can't use it without significant changes. Hint: think of a query that would be easy to answer efficiently in a standard database, but would be difficult in our probabilistic database.

2 Skyline Queries

There has been growing interest in supporting best match and ranking queries in database. One outcome of this is the "Skyline Query". The idea of a skyline query is that we wish to search for only tuples that have the potential to be maximal on *some monotonic scoring function* of the attributes, without knowing the function. In practice, we define this using *dominance*: A tuple is a member of the skyline query result if and only if there is no other tuple that exceeds it in all attributes. For example, given the following table:

tid	x	y	z
a	5	1	4
b	1	3	2
c	3	2	3
d	2	1	2

tuples a, b, and c are in the skyline. Note that while c does not have the highest value for any particular attribute, it exceeds tuple a on attribute y, and tuple b an attribute x. Tuple d, on the other hand, is dominated in all attributes by c, and thus is not part of the skyline.

2.1 Expressing skyline queries

Is it possible to express a skyline query in SQL? If so, show how (e.g., with a skyline query for the above table, or better still demonstrate how to construct a skyline query for any table.)

2.2 Processing skyline queries

Assume very straightforward query processing (no indexes, nested-loop join).

1. What is a rough estimate of the cost of computing the skyline for the example above?

2. Give a technique from the query processing algorithms we have discussed that would significantly improve this cost.

3 Encrypted Database

With recent laws requiring encryption of sensitive personal data, there is an interest in direct database support for encryption. You are asked to determine some of the design issues in constructing a database that supports encryption.

3.1 Disk-only Encryption

In this case, the "attack scenario" is that the running system itself is secure, but that the stored data may be vulnerable (e.g., through loss of backup tapes.) Describe an approach that would ensure that all data on disk is encrypted, with minimal changes to the database management system. You might want to think in terms of what modules in minibase would need to be changed, although you don't need to be that specific.

3.2 Untrusted Server Model

In the second scenario, the database server itself is not secure; e.g., a system administrator who can watch the memory of the running system is not allowed to see sensitive data. Only at the client is that data decrypted.

1. Describe a way of distinguishing between sensitive (must be encrypted) and non-sensitive (need not be encrypted) data. You don't need to figure it out, but provide a tool/mechanism so that an application developer or security administrator can specify this to the system. Your goal should be to minimize the "new mechanisms" administrators must learn, while still providing considerable flexibility (e.g., noting that one column in a table is sensitive.)

2. One disadvantage of this approach is that selects on the sensitive values can only be processed by returning all values that meet selection criteria on non-sensitive values to the client, where they can be decrypted and evaluated. Show a parse tree (relational algebra operations) for the following query that does as much work as possible on the server.

```
Select patient.zip_code, diagnosis.condition
from patient, diagnosis
where patient.id = diagnosis.id and patient.age < 18;
```

   Assume patient.age and diagnosis.condition are sensitive, and other values are not sensitive.