COP 4710 Database Systems

Fall 2010

Final Exam

December 9, 2010

Time: 120 minutes

Your Name: ____________________________

USF ID: __U___________________________

Remember: always write down your intermediate results for partial credits.

Problem I. Basic concepts (10pts)

True or false, mark your choice clearly. Ambiguous answers will get zero points.

1. SQL is an implementation of domain relational calculus. T ( ) F (+)

2. In a relation, a superkey must be a candidate key. T ( ) F (+)

3. In performing natural joins between table A and table B, if we have an index on the joining attribute(s) of table A, we should use A as the OUTER table in the nested-loop join algorithms to gain better performance. T ( ) F (+)

4. A valid relational algebraic expression always returns a relation. T (+) F ( )

5. The resulting table of a relational algebraic expression is duplicate free. T (+) F ( )

6. Relational calculus is a non-procedural language. T (+) F ( )

7. If a relation is in Boyce-Codd Normal Form, it must also be in 3rd Normal Form. T (+) F ( )

8. The schema of a large table A is decomposed into two smaller tables A' and A'' by **lossless** decomposition. If we join A' with A'', we can actually get more tuples than those in the original table A. T ( ) F (+)

9. A DBMS is a software system that enables users to create and maintain a database. T (+) F ( )

10. Given two relations R and S with x and y attributes in their schemas, respectively, the Cartesian product of R and S must have exactly x + y attributes. T (+) F ( )

Problem II. Query languages (27pts, 3 pts each)

Given the following relational database

Movies (title, year, length, genre, studioName, ProducerName)
StarsIn (movieTitle, starName, movieYear)
MovieStar (name, address, gender, birthdate)

where the underlined attributes are the primary keys, and the following foreign keys:

StarsIn.starName → MovieStar.name
StarsIn.movieTitle → Movies.title

Movies.ProducerName → MovieStar.name

You can make reasonable assumptions about the database, please state the assumptions clearly if you do.

II-A. Write the following queries in SQL:

1. Print the year of the movie(s) titled "2012";

```
SELECT m.year
FROM Movies m
WHERE m.title = '2012'
```

2. Print the title and year of the movie(s) produced by MGM in 2003;

```
SELECT m.title, m.year
FROM Movies M
WHERE m.studioName = 'MGM' AND m.year = 2003
```

3. Print the names of actors who have starred in a Spielberg (who acted as a producer) movie in 2003;

```
SELECT s.StarName
FROM StarsIn s, Movies m
WHERE s.movieTitle = m.title AND m.ProducerName = 'Spielberg' AND m.year =
2003
```

4. How many movies did Disney produce in each year since 2000? (Hint: first solve "how many movies did Disney produce in each year")

```
SELECT COUNT (m.title)
FROM Movies m
WHERE studioName = 'Disney'
GROUP BY m.year
HAVING m.year >= 2000
```

5. Print the title and year of all movies Tom Hanks got involved (as a producer or an actor). (Hint: find the movies Tom acted in, then find the movies he was a producer of, …)

```
(SELECT m.year, m.title
FROM StarsIn s, Movies m
WHERE m.title = s.movieTitle AND starName = 'Tom Hanks')
UNION
(SELECT year, title
FROM Movies
WHERE ProducerName = 'Tom Hanks')
```

6. Print the names of movie stars who have acted in ALL categories (genre) of movies.

```
SELECT s.starName
FROM Movies m, StarsIn s
WHERE m.title = s.movieTitle
GROUP BY s.starName
HAVING COUNT(DISTINCT genre) IN (SELECT COUNT(DISTINCT genre) from
Movies)
```

II-B. Write the following queries in relational algebra:

7. Print the title and year of the movie(s) produced by MGM in 2003;

$$\Pi_{M.title,\ M.year}\ \sigma_{M.studioName = \text{'MGM'} \wedge M.year = 2003}\ (M)$$

8. Print the names of actors who starred in the movie "Casablanca" but NOT in the movie "Spellbound".

$$\Pi_{S.starName}\ \sigma_{S.movieTitle = \text{'Casablanca'}}\ (S)$$

-

$$\Pi_{S.starName}\ \sigma_{S.movieTitle = \text{'Spellbound'}}\ (S)$$

9. Print the names of movie stars who have acted in ALL categories (genre) of movies.

$$\Pi_{S.starName,\ M.genre}\ \sigma_{S.movieTitle = M.title}\ (S \times M) \div \Pi_{M.genre}\ (M)$$

Problem III. (20 pts, 4 pts each) Database Design

Consider a relation R(A, B, C, D, E) and the following set of functional dependencies:

$$AB \rightarrow C,\ \ CD \rightarrow E,\ \ DE \rightarrow B.$$

Finish the following tasks.

1. Compute (AB)+;

AB→ C therefore (AB)+ = (ABC)

2. Compute (AD)+;

(AD)+ = (AD)

3. Find all the candidate keys of relation R;

|__L_|__L+R__|__R__|
  AD   BCE

So we start with (AD)+ = (AD), not the key.
(ADB)+ = (ADBCE)
(ADC)+ = (ADCEB)
(ADE)+ = (ABCDE)
Therefore, ADB, ADC, and ADE are all candidate keys.

4. Does R follow Boyce-Codd Normal Form? Explain how you reach your conclusion.

No, for every FD rule we have, the left hand side (i.e., AB, CD, DE) is not a key.

5. What is the highest normal form R follows? Explain how you reach your conclusion.

3rd NF. Since the right hand side of all FDs are primes, the second condition of 3rd NF always holds true.

Problem IV. Query processing (18 pts)

Consider a natural join between relation *R* and relation *S*, and the following information about the relations to be joined. The cost metric is the number of page I/Os unless otherwise noted, and the cost of writing out the result should be uniformly ignored.

Relation R contains 10,000 data pages and each page contains 10 tuples.
Relation S contains 5,000 data pages and each page has 20 tuples in it.
Both relations are stored as simple heap files.
102 buffer pages are available.

Answer the following question with your justifications.

For the answers, let us define M = 10000, N = 5000, P_R = 10, P_S = 20 according to the given information. Here I am not giving the calculated results, but in the exam you need to do that.

1. (4pts) What is the cost of joining R and S using a page-based simple nested loop join?

   Cost is MN+ M (using R as outer table) or MN+N (if using S as outer table)

2. (4 pts) What is the cost of joining R and S using a block-based nested loops join?

   Cost is [MN/(102-2)]+ M (using R as outer table) or [MN/(102-2)] +N (if using S as outer table)

Let us further assume that we have built B+-tree indexes on the join attributes for both R and S. We obviously can perform index-based loop join.

3. (5 pts) To get the best join performance, which index would you use? Briefly explain your answer.

   The cost of an indexed loop join is: M P_R C + M if we use S as inner table and its index, and N P_S C + N is we use R as the inner table and its index. In this case, the index access cost C is the same for both options. The second option (using R as the inner table and its index) obviously has lower cost.

4. (5 pts) What is the cost of performing the join with the index you choose?

   As mentioned in 3, cost is N P_S C + N and C is 3.

Problem V. Indexing (25pts)

1. (10pts) Construct a B$^+$-tree for the following set of key values:
   27, 20, 43, 40, 67, 17, 55, 33, 11, 75, 48, 9, 95, 60, 65, 107, 98

Assume the tree is initially empty and the values are added in the order specified by the above list. The maximum number of data entries that will fit in one node is FOUR (i.e., d = 2 as we discussed in class). Draw the intermediate trees for partial credits.

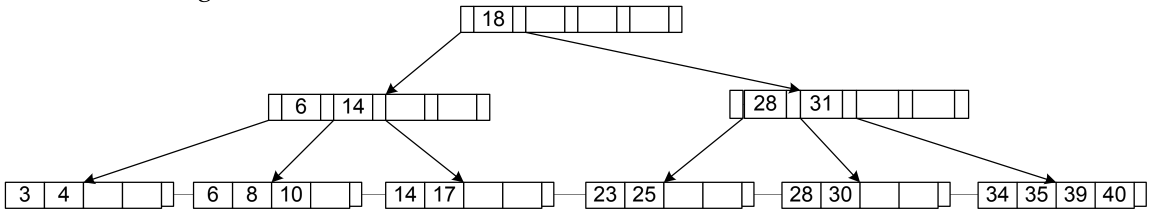
Final tree:

(55)

(20,40) (67,95)

(9,11,17) (20,27,33) (40,43,48) (55,60,65) (67,75) (95,98,107)

Given the following B$^+$-tree:



2. (2pts) Draw the tree after deleting value 10 from the above tree;

   Only change the leaf node (6,8,10) to (6,8)

3. (4pts) Draw the tree after deleting key value 3 from the **original** tree (not the one you got from question 2) shown in the figure.

   Run Redistribution, the lower left part of the tree becomes:

   (8,14)

   (4,6) (8,10)

4. (5pts) Draw the tree after deleting key value 23 from the **original** tree (not the one you got from question 3) shown in the figure.

   Need to perform Merge, final tree becomes:

   (6,14,18,31)

   (3,4)   (6,8,10)   (14,17)   (25,28,30)   (34,35,39,40)

5. (4 pts) Starting from the original tree, list two values such that the height of the tree will decrease by one when these two values are deleted from the tree.

   Many possibilities, you just need to find two values and the deletion of them will cause a merge on the second level of the original tree. For example, deletion of 6 and 3 will qualify.