CS411 Database Systems

Fall 2006

Department of Computer Science
University of Illinois at Urbana-Champaign

Final Examination Solution Set

December 15, 2006

Time Limit: 180 minutes

Problem 1 (*10 points*) True/False Questions

(1) True; (2) False; (3) False; (4) True; (5) True;
(6) True; (7) True; (8) False; (9) False; (10)True;

Problem 2 (*10 points*) Multiple Choice

(1) C; (2) D; (3) C; (4) D; (5) C;

Problem 3 (*11 points*) Data Storage

(1) How many blocks are on a disk with the following characteristics? Sector size of 512 bytes, 16 sectors per track, 16384 tracks per surface, 4 double sided platter, and 4096 bytes per block. [2 points]

512*16*16384*4*2/4096=262144 blocks

(2) Explain (i) what it means for a block to be pinned and (ii) give one reason for pinning a block. [3 points]

A block is pinned tby he buffer manager when a frame is in use. If a block is pinned, it means that it is current in use we won't allow it to be written to disk. This is useful in cases with pointer swizzling when there other blocks that may still reference this block, thus it is still pinned. Another scenario pinning is useful is when the block is part of a larger uncommitted transaction.

(3) What is the purpose of a tombstone in an offset table [2 points]?

Tombstones are used for structured addresses where a block has an offset table containing entries for each record. The entry would contain an offset based off the physical address. When an record has been deleted, the entry in the offset table would get replaced with a tombstone which will tell the records that reference this structured address that the record has been deleted. The original space for the actual record could be replaced with another record.

(4) How many records can we put into a block of size 4096 bytes with the following conditions? [4 points]
a) Each record has the following fields in order: a real of 6 bytes, a character string of length 17 bytes, and an 8 byte timestamp.

b) The record header consists of two 4 byte pointers and one 10 byte date.
c) Fields must start at a byte location that is a multiple of 4.
d) Block header includes eight additional 4-byte integers.

For each record (56):
Header: 4+4+12(10)=20
Fields: 8(6)+20(17)+8=36

For the block:
(4096-32)/56=4064/56=72 records

Problem 4 (*18 points*) Triggers

Consider the following tables with the following inserts and triggers:

```
CREATE TABLE UserBucket (firstname VARCHAR(20), lastname VARCHAR(20),
bucket INT DEFAULT NULL);

CREATE TRIGGER UserBeforeTrigger BEFORE INSERT ON UserBucket
  FOR EACH ROW
REFERENCING NEW ROW AS NewTuple
WHEN (NewTuple.firstname LIKE 'E%')
  SET NewTuple.bucket=1;

CREATE TABLE BucketCount (bucketid INT, count INT);
INSERT INTO BucketCount(bucketid, count) VALUES(0, 0);
INSERT INTO BucketCount(bucketid, count) VALUES(1, 0);

CREATE TRIGGER UserAfterTrigger AFTER INSERT ON UserBucket
  FOR EACH ROW
REFERENCING NEW ROW AS NewTuple
UPDATE BucketCount SET count=count+1  WHERE bucketid=NewTuple.bucket;

CREATE TRIGGER UserTableTrigger AFTER UPDATE
OF lastname ON UserBucket
REFERENCING
   OLD TABLE As OldStuff
   NEW TABLE AS NewStuff
BEGIN
   UPDATE BucketCount SET count=(SELECT count(*) FROM OldStuff)
END
```

(a) Show the contents of the *UserBucket* table after executing the following query: [3 points]

```
INSERT INTO UserBucket(firstname, lastname) VALUES('Evan', 'Smith');
```

firstname, lastname, bucket
'Evan', 'Smith', 1

(b) Starting with the initial problem (*ie* ignoring the queries in subproblem *a*), show the contents of the *UserBucket* table after executing the following query: [3 points]

```
INSERT INTO UserBucket(firstname, lastname) VALUES('Angela', 'Smith');
```

firstname, lastname, bucket
'Angela', 'Smith', null

(c) With the initial problem (*ie*, ignoring the queries in the previous subproblems), show the contents of the *BucketCount* table after executing the following queries: [3 points]

```
START TRANSACTION;
INSERT INTO UserBucket(firstname, lastname) VALUES('Laura', 'Stuart');
INSERT INTO UserBucket(firstname, lastname) VALUES('Evonne', 'Chu');
ROLLBACK;
```

bucketid, count
0, 0
1, 0

(d) Starting with original problem (*ie* ignoring the queries in the previous subproblems), show the contents of the *BucketCount* table after executing the following queries: [3 points]

```
INSERT INTO UserBucket(firstname, lastname) VALUES('Larry', 'Smith');
INSERT INTO UserBucket(firstname, lastname) VALUES('Edward', 'Johnson');
UPDATE UserBucket SET lastname='Jones' WHERE firstname='Edward' AND lastname='Johnson';
```

bucketid, count
0, 1
1, 1

(e) Explain one advantage and one disadvantage of using a trigger to enforce a constraint compared to using a CHECK constraint [4 points]

Triggers works better when the constraints involve multiple relationships. With CHECK constraints violations can occur when modifications are being made to the referenced relationship in the CHECK.

Triggers are more efficent and have finer granular control, they execute based upon on a specificed type of event such as before and after insertion, updates, and deletes.

A disadvantage of triggers is that they are not available on every DBMS. Another disadvantage is triggers can not prevent an operation but only correct it. Disadvantage have to tell triggers when to activate.

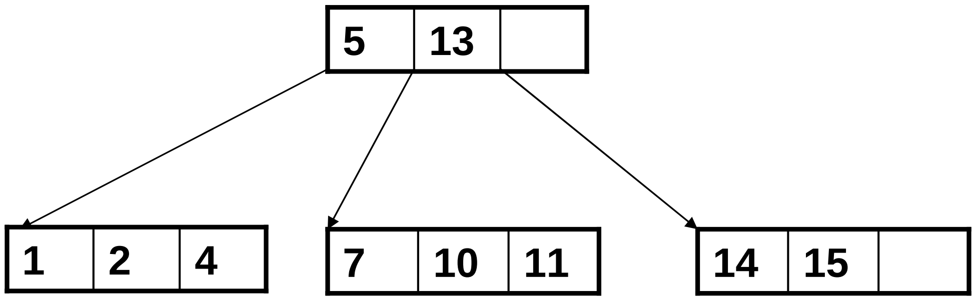

Figure 1: Problem 5

Problem 5 (*12 points*) B+ Tree Consider the B+ tree index of degree 3 shown in the figure below. For each problem below, you only need to show the final resulting B+ Tree.

a) Show the tree that would result from inserting a data entry with key 19 into this tree. [2 points]

b) Show the B+ tree that would result from inserting a data entry with key 3 into the *original* tree. [3 points]

c) Show the B+ tree that would result from deleting the data entry with key 10 from the *original* tree. [2 points]

d) Show the B+ tree that would result from deleting the data entry with key 14 from the original tree. [3 points]

e) Explain why in a real database most relations only require B+ trees with two or three levels. [2 points]

<u>**Problem 6**</u> (*10 points*) Indexing

Consider indexing the following key values using a linear hash table. Here is the specification of these index structures.

The hash function h(n) for key n is h(n) = n mod 16; i.e., the hash function is the remainder after the key value is divided by 16. Thus, the hash value is 4 bits. Assume that each bucket can hold 2 data items. We adopt the policy that the average occupancy of a bucket cannot exceed 85%.

Suppose that we insert the keys in the order of: 45, 36, 31, 56, 34.

(a) Draw the linear hash index after all the keys are inserted. Show how the keys and their hash values are distributed within the buckets. [6 points]

(b) Briefly explain the advantages and disadvantages of linear hash tables, in terms of the following aspects, compared to B+ Tree? [4 points]

- Insertion:

- Querying:

<u>**Problem 7**</u> (*16 points*) Transaction Logs

A database has four elements, A, B, C, and D. Assume that the following is a normal sequence of *undo* log records, using non-quiescent checkpointing:

1 <start T1>

2 <T1,B,40>

3 <start T2>

4 <T2,A,56>

5 <T2,C,34>

6 <start T3>

7 <commit T1>

8 <T3,B,12>

9 <commit T2>

10 <T3,D,89>

11 <start T4>

12 <T4,C,7>

13 <T3,A,22>

14 <commit T4>

15 <T3,A,99>

16 <commit T3>

(a) When is the *latest time* for transaction T1, T2 that "dirty data" can be flushed onto disk (ie, the time Output(X) for data X can be performed)? [2 points]

For T1: Output(s) B before Log Line 7.
For T2: Output(s) A and C before Log Line 9.

(b) Suppose we start checkpointing right after Log 5, indicate where and what the start checkpointing record would look like. Then, indicate where and what the *earliest* end checkpoint record would look like. [2 points]

Between Log Line 5 and 6, <START Checkpoint(T1, T2)>
Between Log Line 9 and 10, <END Checkpoint>

(c) Continue from (b). Suppose the system crashes right after Log 14 and the end checkpoint has been written out to disk. What is the contents of the earliest log line we must examine? And which transaction records do we need to undo in sequence? [4 points]

Earliest Log File Contents: <START Checkpoint(T1, T2)> Log lines 5 and 6, Transaction Sequence: <T3,A,22>, <T3,D,89>, <T3,B,12>

(d) Now, suppose that the log is a *redo* log. When is the *earliest time* for transactions T3 and T4 that "dirty data" can be flushed onto disk? [4 points]

For T3: Output(s) A, D, and B after Log Line 16.
For T4: Output(s) C after Log Line 14.

(e) Show the contents of log line 13 if this was extended to be *undo-redo* log. Assume the values shown are the undo log entries. [2 points]

Log Line 13: <T3, A, 22, 99>

(f) Explain one advantage of using a redo log as opposed to an undo log. [2 points]

Redo logging allows us to commit a transaction to log file without writing to disk first. Sometimes, this will allow us to save I/Os by allowing those changes to reside in main memory for a little while.

Problem 8 (*15 points*) Query Optimization

Consider a distributed system with two sites X and Y connected through a network. Relation R(A, B, C) resides at site X and relation S(C, D) resides at site Y. The statistics of the relations are as follows:

T(R) = 2,000 (number of tuples in R)
T(S) = 100,000 (number of tuples in S)
$S_A = S_B = S_C = S_D = 10$ bytes (size of each attribute)
V(C, R) = 200 (number of distinct values of attribute C in R)
V(C, S) = 2,000 (number of distinct values of attribute C in S)

We want to compute the natural join of the two relations: $T = R \bowtie S$, and we want the resulting relation T to be stored at site X. We are given the two plans as follows:

Plan A: Copy relation S from site Y to X, compute the join at site X, and store the result at site X.

Plan B: Copy relation R from site X to Y, compute the join at site Y, and move the resulting relation to site X for storage.

a) Calculate the total number of bytes transferred between site X and Y for plan A. [4 points]

b) Calculate the total number of bytes transferred between site X and Y for plan B. [6 points]

c) Grand Challenge: Design another plan that can further reduce the amount of data transferred between the two sites than plans A and B. Briefly describe your plan and calculate the total number of bytes transferred between sites X and Y for your plan. [Hint: Try to avoid transferring the whole relation.] [5 points]