UCLA
Computer Science Department
Fall 2018

Instructor: J. Cho

Student Name and ID: _____

CS143 Final: Closed Book, 120 minutes

(IMPORTANT PLEASE READ **):**

- *Simplicity and clarity of your solutions will count.* You may get as few as 0 point for a problem if your solution is far more complicated than necessary, or if we cannot understand your solution.
- If you need to make any assumption to solve a question, please write down your assumptions.
- To get partial credits, you may want to write down how you arrived at your answer step by step.
- The exam is closed book and closed notes.
- You may use two double-sided cheat-sheets during exam. You are also allowed to use a calculator.
- Please, write your answers neatly. Attach extra pages as needed. Write your name and ID on the extra pages.

Problem	**Score**	
1	15	
2	15	
3	20	
4	10	
5	15	
6	15	
Total	90	

Problem 1 (File and Index): 15 points

Consider a movie table created by the following SQL statement:

```
CREATE TABLE Movie(
    id INTEGER,
    name CHAR(50),
    director CHAR(50),
    studio CHAR(50),
    year INTEGER
)
```

An integer value takes up 4 bytes to store. The movie table contains 100,000 tuples and is stored as fixed-length records. The tuples are *not spanned*. We use a disk of the following parameter to store the table.

- 3 platters (6 surfaces)
- 1,000 cylinders
- 100 sectors per track
- 1024 bytes per sector
- 6,000 RPM rotational speed
- 10ms average seek time

1. Assume one disk sector is mapped to one disk block. What is the minimum number of blocks that we need to store the movie table? (5 points)

   ANSWER: The size of each tuple is 4 + 50 + 50 + 50 + 4 = 158 bytes. The size of each block is 1024. Because tuples are not spanned, we can store $\lfloor \frac{1024}{158} \rfloor = 6$ tuples per block. Since we have 100,000 tuples, we need $\lceil \frac{100{,}000}{6} \rceil = 16,667$ blocks to store the entire table.

2. In the *best case scenario*, how long does it take to read 30 tuples from the movie table? (5 points)

   ANSWER: In the best case scenario, we do not need to wait for seek or rotational delay. We can immediately start reading the 30 tuples from the

disk. Since each block contains 6 tuples, we need to read 5 blocks. The transfer time for one block is $\frac{60}{6000}/100 = 0.1$ms. If the 5 blocks are arranged sequentially, it takes $5 \times 0.1 = 0.5$ms

3. Assume that the relation is stored as a sorted file in increasing lexicographic order of `(name, year)`. For each of the following answer True or False and explain your answer briefly in the space provided. (5 points)

   (a) It is possible to build a sparse index on name

   ANSWER: `Yes`

   (b) It is possible to build a dense index on studio

   ANSWER: `Yes`

   (c) It is possible to build a sparse index on director

   ANSWER: `No`

   (d) It is possible to build a sparse index on year

   ANSWER: `No`

Problem 2 (B+Tree): 15 points

Consider a B+tree that indexes 300 records.

1. If $n = 10$ for this B+tree (i.e., each node has at most 10 pointers), what is the minimum and maximum height (depth) of the tree? (A tree with only the root node has a height of 1.) (5 points)

   ANSWER: Minimum: 3. Leaf node can hold up to 9 keys, when the tree is (almost) full and has the minimum height, we have $\lceil 300/9 \rceil = 34$ leaf nodes. Since the branching factor of the tree is 10, right above the leaf level, we have $\lceil 34/10 \rceil = 4$ nodes. The root node can hold pointers to these 4 child nodes.

   Maximum: 4. Leaf node must hold at least $\lceil \frac{n-1}{2} \rceil = 5$ keys to avoid underflow. So when the tree is least populated (thus maximum height), we have $\lfloor 300/5 \rfloor = 60$ leaf nodes. At a nonleaf node (except root), we need at least $\lceil \frac{n}{2} \rceil = 5$ pointers to avoid the underflow. That is, the branching factor of the tree when it is least populated is 5. Therefore, at right above the leaf level, we have $\lfloor 60/5 \rfloor = 12$ nodes. Above it, we have $\lfloor 12/5 \rfloor = 2$ nodes. The root node then can have the two pointers to these two nodes.

2. If this B+tree has a height of 2, what is the minimum and maximum n? (10 points)

   ANSWER: Minimum: 18. For minimum n, the tree must be (almost) fully populated. Each leaf node can contain up to $n-1$ keys, so when the tree is (fully) populated, we have $\lceil \frac{300}{n-1} \rceil$ leaf nodes. Since the branching factor of the tree is n, the number of leaf nodes should be n or smaller if the tree height is 2. That is, $\lceil \frac{300}{n-1} \rceil \leq n$. The smallest n that satisfies this inequality is 18.

   Maximum: 301. For maximum n, the tree must be least populated while avoiding underflow. Each leaf node must contain at least $\lceil \frac{n-1}{2} \rceil$ keys, so we need at least $\frac{300}{\lceil \frac{n-1}{2} \rceil}$ leaf nodes. The number of leaf nodes should be at least 2 for the tree height to be 2, so we get $\frac{300}{\lceil \frac{n-1}{2} \rceil} \geq 2$. The largest n that satisfies this inequality is 301.

Problem 3 (Join): 20 points

We need to perform the natural joins of $R(A, B) \bowtie S(B, C) \bowtie T(C, D)$

You have a non-clustering B-tree index on attribute C of relation T. Assume this index is kept entirely in memory (i.e., you do not need to read it from disk). Other relevant data:

- 20 tuples of R are stored per block on disk.
- $|R| = 10,000$ (number of tuples of R)
- 10 tuples of S are stored per block on disk.
- $|S| = 20,000$ (number of tuples of S)
- 10 tuples of T are stored per block on disk.
- $|T| = 100,000$ (number of tuples of T)
- For every R tuple, there are 2 tuples in S with $R.B = S.B$ on average
- For every S tuple, there is 1 tuple in T with $S.C = T.C$ on average

We decided to do block nested-loop join for $R \bowtie S$ and index join for $S \bowtie T$. We designate 100-block memory buffer, M_R, to read R blocks, 1-block memory buffer, M_S, to read S blocks, and 1-block memory buffer, M_T, to read tuples in T. More precisely, our query plan is as following:

```
1) Load the first 100 blocks of R into memory buffer Mr
2)   Load the first block of S into memory buffer Ms
3)   For every tuple r in Mr
4)     If there is s in Ms with r.B = s.B then
5)       Use the index on C of T to retrieve the tuples t of T
         with t.C = s.C. Then output r.A, r.B, s.C, t.D
6)   If not at the end of S table, then
7)     Load the next block of S into Ms and go to Step 3
8) If we have not reached the end of R table, then
9)   Load the next 100 blocks of R into Mr and go to Step 2
```

For all questions, write down the final calculated number, not just a formula.

1. How many disk blocks does R table use? How many for S? (4 points) ANSWER: `R:` $\frac{10,000}{20} = 500$. `S:` $\frac{20,000}{10} = 2,000$
2. How many times is each block of R table read during join? In total, how many disk block READs are performed on R table? (5 points) ANSWER: `Once. 500 IOs`

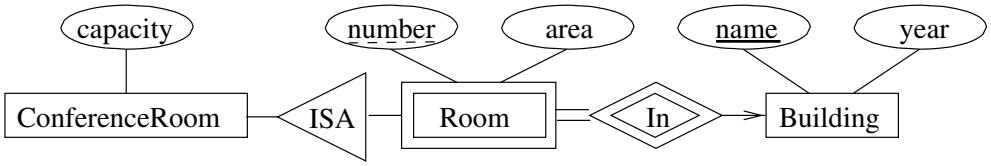
3. How many times is each block of S table read during join? In total, how many disk block READs are performed on S table? (5 points) ANSWER: `5 times. 5*2,000 = 10,000 IOs`

4. What is the expected number of output tuples from the join $R \bowtie S \bowtie T$? In total, how many disk block READs are performed on T table? (5 points) ANSWER: `10,000*2*1 = 20,000 tuples. 20,000 IOs`

5. What is the total number of expected disk block READs for the entire join? (1 point)
   ANSWER: `500+10,000+20,000 = 30,500 IOs`

Problem 4 (ER Model): 10 points

Consider the following ER diagram for a database that keeps track of buildings rooms and conference rooms.



1. Convert the above ER diagram into relations. In converting a superclass/subclass, assume that we want to avoid using NULL values and avoid storing the same information redundantly, if possible. In the space provided below, list the schema of the converted relations. Underline a key of each relation. (6 points)

   ANSWER: `Building(name, year)`
   `Room(Building.name, number, area)`
   `ConferenceRoom(Building.name, number, capacity)`

2. Which of the following statements are true according to the constraints encoded by the ER diagram above? Do not make any assumptions other than those encoded by the ER diagram. (4 points)

   I. The number of entities in the Room entity set must be greater than or equal to the number of entities in the ConferenceRoom entity set

   II. The number of entities in the Room entity set must be greater than or equal to the number of entities in the Building entity set

   (a) I only (b) II only (c) Both I and II (d) Neither I nor II

   ANSWER: `(a) Every conference room is a room, but there can be buildings without any rooms`

Problem 5 (Functional Dependency): 15 points

Consider the following table storing temperature readings taken by sensors:

`Temps(sensorID,time,temp)`

For example, a tuple (s26,13:42,76) in Temps says that at time 13:42, sensor s26 reported a temperature of 76 degrees. (Don't worry about dates; perhaps our sensors are active for only one day.) Assume for all parts (a)-(c) of this problem that the pair of attributes `(sensorID,time)` is the only key for table `Temps`.

1. Suppose the functional dependency "`time`→`temp`" holds on table `Temps`. State in English what real-world property is captured by this FD. (A real-world property is described by a phrase or sentence discussing sensors, times, and/or temperatures, while not mentioning tables, tuples, and/or attributes.) Please be brief; a short phrase or one-sentence answer is sufficient. (5 points)

   ANSWER: `At a given time, the temperature readings from all sensors are the same.`

2. Continuing to assume that FD `time`→`temp` holds for `Temps`, is `Temps` in Boyce-Codd Normal Form? Briefly explain your answer; again, a short phrase or one-sentence answer is sufficient. (5 points)

   ANSWER: `No. The left-hand side does not contain a key.`

3. Write a SQL query to test whether the FD `time`→`temp` holds on a given instance of Temps. Specifically, your SQL query should return the list of the time values that violate the FD. The query should return an empty answer if the FD does hold. (5 points)

   ANSWER:
   ```
   SELECT time FROM Temps GROUP BY time HAVING COUNT(DISTINCT temp)
   > 1
   ```

Problem 6 (Serializability): 15 points

Consider the following schedule:

$$r_1(X)\, w_1(X)\, r_2(X)\, c_2\, r_3(Y)\, w_3(Y)\, c_3\, r_1(Y)\, w_1(Y)\, c_1$$

Circle YES or NO for the questions below and provide brief explanation. Be careful with your answers for this problem. For every correct answer, you will get 2 points, but for every incorrect answer, you will get 1 point deducted.

1. Is the schedule serial? (3 points) ANSWER: `No`
2. Is the schedule conflict serializable? (3 points) ANSWER: `Yes. The precedence graph is` $T_2 \leftarrow T_1 \leftarrow T_3$ `and has no cycle.`
3. Is the schedule recoverable? (3 points) ANSWER: `No.` c_2 `appears before` c_1 `even though` T_2 `reads a value written by` T_1`.`

4. Should the above schedule be allowed when all three transactions run under the isolation level SERIALIZABLE? (3 points) ANSWER: `No.` T_2 `does dirty read from` T_1 `and commits before` T_1`.`

   Depending on your answer, provide an answer to one of the following questions. (3 points)

   - If your answer is yes, write down the serial schedule that is equivalent to the previous schedule.

   - If your answer is no, what is the the minimum relaxation necessary to the isolation level of each transaction to allow this schedule? The isolation levels of the three transactions may or may not be the same after the relaxation.

     T_1 isolation level: _____

     T_2 isolation level: _____

     T_3 isolation level: _____

     ANSWER: `No. Change the isolation level of` T_2 `to READ UNCOMMITTED.`

This page is intentionally left black to give you extra scratch space.