CS W186 Fall 2019 Midterm 1

Do not turn this page until instructed to start the exam.

Contents:

- You should receive one *double-sided answer sheet* and a 14-page *exam packet*.
- The midterm has *5 questions*, each with multiple parts, and worth a total of *78 points*.

Taking the exam:

- You have *110 minutes* to complete the midterm.
- All answers should be written on the answer sheet. The exam packet will be collected but not graded.
- For each question, place only your *final answer* on the answer sheet; *do not show work*.
- For multiple choice questions, please *fill in the bubble or box completely* as shown on the left below. ***Answers marking the bubble or box with an X or checkmark may receive a point penalty***.

- A blank page is provided at the end of the exam packet for use as scratch paper.

Aids:

- You are allowed **one handwritten** 8.5" × 11" double-sided pages of notes.
- *No electronic devices are allowed on this exam.* No calculators, tablets, phones, smartwatches, etc.

Grading Notes:

- All IOs must be written as integers. There is no such thing as 1.02 IOs – that is actually 2 IOs.
- 1 KB = 1024 bytes. We will be using powers of 2, not powers of 10
- Unsimplified answers, like those left in log format, will receive a point penalty.

0 Pre-Exam Questions (0 points)

1. (0 points) How many grams of sugar are in an average boba drink?
2. (0 points) How many TA(s) does CS W186 have this semester?

1 SQL & Relational Algebra (18 points)

You have just graduated with a Computer Science degree from UC Berkeley, a prestigious institution whose alumni have received more Turing Awards than any other university in the world. Your friends are all excited to begin their nascent careers (and make big bucks).

Wary of the tech bubble, however, and sensing that there might be more money in the boba craze than in browsing Stack Overflow eight hours a day, you decide not to join your friends in South Bay, and instead set up a bubble tea shop on Bancroft Avenue.

You hire Joe Hellerstein to design your database. He builds a database for you with the following schemas:

```
-- This table lists all of your sales, along with the drink sold and the customer who bought it.
CREATE TABLE Sales (
    sale_id INTEGER PRIMARY KEY,
    drink_id INTEGER FOREIGN KEY REFERENCES Drinks,
    customer_id INTEGER FOREIGN KEY REFERENCES Customers);

-- This table lists all of the drinks you have for sale.
CREATE TABLE Drinks (
    drink_id INTEGER PRIMARY KEY,
    name VARCHAR(20) UNIQUE NOT NULL,
    price DOUBLE NOT NULL);

-- This table stores customer information.
CREATE TABLE Customers (
    customer_id INTEGER PRIMARY KEY,
    name VARCHAR(50) NOT NULL);
```

You also hire Joe to run your database for the first few weeks. However, his consulting rate of $250 an hour proves too expensive, and you sadly have to let him go. Alas! You have to run the database yourself now. Let's hope you remember everything you learned in CS 186.

1. (4 points) A customer walks in and asks what your most popular drink is. Embarrasingly, you have no idea. What query will give you the name of the most sold drink(s)?**There may be zero, one, or multiple correct answers.**

   A.
   ```
   SELECT Drinks.name
   FROM Sales, Drinks
   WHERE Sales.drink_id = Drinks.name
   GROUP BY Drinks.name
   ORDER BY COUNT(*) DESC
   ```

   B.
   ```
   SELECT Drinks.name
   FROM Sales INNER JOIN Drinks
   ON Sales.drink_id = Drinks.drink_id
   WHERE COUNT(Sales.drink_id) >= ALL (
       SELECT COUNT(*)
       FROM Sales
       GROUP BY Sales.drink_id
   )
   ```

   C.
   ```
   SELECT Drinks.name
   FROM Sales, Drinks
   WHERE Sales.drink_id = Drinks.drink_id
   GROUP BY Drinks.name
   HAVING COUNT(*) = (
       SELECT COUNT(*)
       FROM Sales
       GROUP BY Sales.drink_id
       ORDER BY COUNT(*)
       LIMIT 1
   )
   ```

2. (4 points) Unfortunately, your bubble tea shop is not doing so well, and you need to cut some items from the menu. You wonder if there are any drinks that have never been ordered.

   What query will get you the names of drinks that have never been ordered?

   There may be zero, one, or multiple correct answers.

   A.
   ```
   SELECT Drinks.name
   FROM Drinks
   WHERE Drinks.drink_id NOT IN (
       SELECT Sales.drink_id
       FROM Sales
   )
   ```

   B.
   ```
   SELECT Drinks.name
   FROM Sales LEFT OUTER JOIN Drinks
   ON Sales.drink_id = Drinks.drink_id
   WHERE Sales.drink_id IS NULL
   ```

   C.
   ```
   SELECT Drinks.name
   FROM Sales RIGHT OUTER JOIN Drinks
   ON Sales.drink_id = Drinks.drink_id
   GROUP BY Drinks.name
   HAVING COUNT(Sales.drink_id) = 0
   ```

After fixing your menu, your shop is much more efficient, and business is booming. You decide to reward your customers by buying each of them their favourite drink. However, you only want to reward your most loyal customers - those with 5 or more purchases.

To find your loyal customers and their favourite drinks, you execute the following query:

```
WITH LoyalCustomers(customer_id) AS (
    SELECT Customers.customer_id
    FROM Sales, Customers
    WHERE Sales.customer_id = Customers.customer_id
    GROUP BY Customers.customer_id
    HAVING COUNT(*) >= 5
)
SELECT Drinks.name, Customers.name
FROM Sales, LoyalCustomers, Drinks, Customers
WHERE Sales.customer_id = LoyalCustomers.customer_id
  AND Sales.drink_id = Drinks.drink_id
  AND Customers.customer_id = LoyalCustomers.customer_id
  AND Sales.drink_id IN (
    SELECT Sales.drink_id
    FROM Sales
    WHERE Sales.customer_id = LoyalCustomers.customer_id
    GROUP BY Sales.drink_id
    ORDER BY COUNT(*) DESC
    LIMIT 1
)
```

3. (1 point) Can there be any NULL values anywhere in this result?
4. (1 point) Can there be any duplicate rows in this result?
5. (1 point) If you execute this query twice, are you guaranteed to get the same result?
6. (1 point) The query above contains the following subquery:

   ```
   SELECT Sales.drink_id
   FROM Sales
   WHERE Sales.customer_id = LoyalCustomers.customer_id
   GROUP BY Sales.drink_id
   ORDER BY COUNT(*) DESC
   LIMIT 1
   ```

   Is this a correlated subquery? Or is it an uncorrelated subquery?

For the following questions, we refer to the Sales table as S, Drinks table as D, and Customers table as C.

7. (3 points) One day, Josh, the owner of a boba tea shop across the street from you, comes into your shop. Jealous of the popularity of your shop, Josh says customers only buy the cheap drinks from your shop. Irritated by his statement, you want to find all the sale_id and price for drinks with price above 4.2 dollars.

   For the following relational algebra expressions, mark True if it finds the sale_id and price of drinks sold with price above 4.2 dollars, False otherwise.

   A. $\pi_{\text{sale\_id, price}}((\sigma_{\text{price}>4.2}(D)) \bowtie S)$

   B. $\pi_{\text{sale\_id, price}}(S \bowtie (\sigma_{\text{price}>4.2}(\pi_{\text{drink\_id}}(D))))$

   C. $\pi_{\text{sale\_id, price}}(\sigma_{\text{price}>4.2}(D \bowtie \pi_{\text{drink\_id, sale\_id}}(S)))$

8. (3 points) To further convince Josh, you want to find the customer_id of all the customers who have bought both drinks of price above 4.2 dollars and drinks of price not above 4.2 dollars.

   For the following relational algebra expressions, mark True if it finds the customer_id of customers who have bought both drinks with price > 4.2 and drinks with price ≤ 4.2, False otherwise.

   A. $\pi_{\text{customer\_id}}(\sigma_{\text{price}>4.2 \text{ AND price}<=4.2}(S \bowtie D))$

   B. $\pi_{\text{customer\_id}}(\sigma_{\text{price}>4.2}(S \bowtie D)) \bigcap \pi_{\text{customer\_id}}(\sigma_{\text{price}<=4.2}(S \bowtie D))$

   C. $\pi_{\text{customer\_id}}(\sigma_{\text{price}>4.2}(S \bowtie D)) - (\pi_{\text{customer\_id}}(\sigma_{\text{price}<=4.2}(S \bowtie D)) - \pi_{\text{customer\_id}}(\sigma_{\text{price}>4.2}(S \bowtie D)))$

2 Disks & Files (15 points)

1. (5 points) Which of the following statements are True? **There may be zero, one, or more than one correct answer**.
    A. Slot directory footers can only be used for pages storing variable length records.
    B. Modifying a page currently in memory counts as 1 I/O.
    C. Insertion and deletion are faster for packed pages than unpacked pages.
    D. I/O cost of performing a full scan on a sorted file is the same as scanning a heap file, assuming both are packed.
    E. Binary search allows for fast lookups on sorted files implemented as linked lists.

Consider a Heap File that takes up 201 MB, including any metadata, storing **variable length records** where half of the data pages are full and the other half contain some free space. The file is implemented as a **Linked List** and each page is 1 MB (for this question, assume 1 MB = 1024 KB).

Assume, for the following two questions, that any insertion does not change the status of a page from having free space to being full.

2. (2.5 points) What is the I/O cost to insert a record into the heap file in **the best case**?

3. (2.5 points) In **the worst case**?

Consider a 200 MB Heap file storing **fixed length records** where at least one data page contains some free space. The file is implemented as a **Page Directory** where each header page can hold 19 entries and each page is 1 MB. On each page, the page header is 24 KB and the size of each record is 10 KB. You have 5GB of memory available to you, and it is entirely empty to begin with.

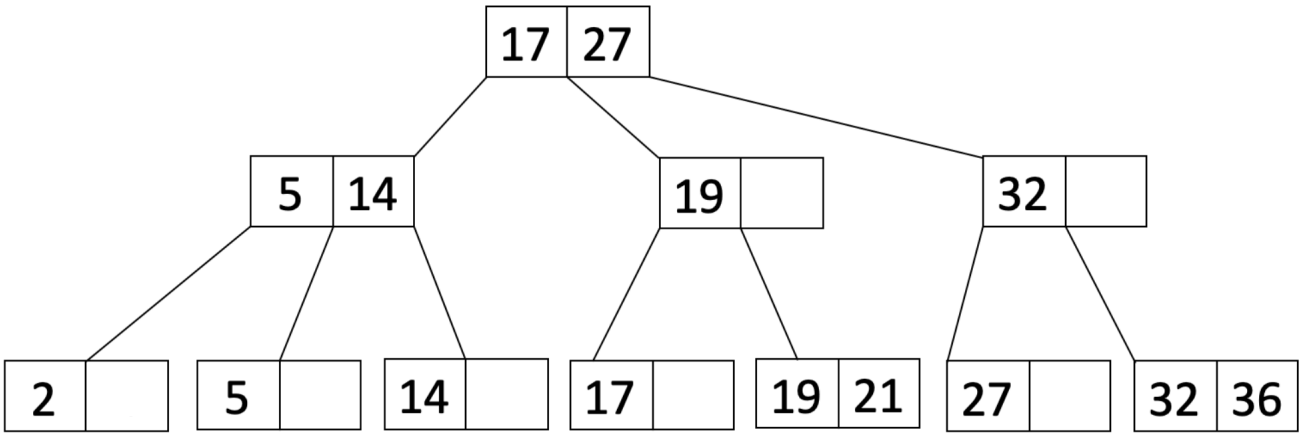
4. (2.5 points) What is the I/O cost to insert a record into the heap file in **the best case**?

5. (2.5 points) In **the worst case**?

3 B+ Trees (15 points)

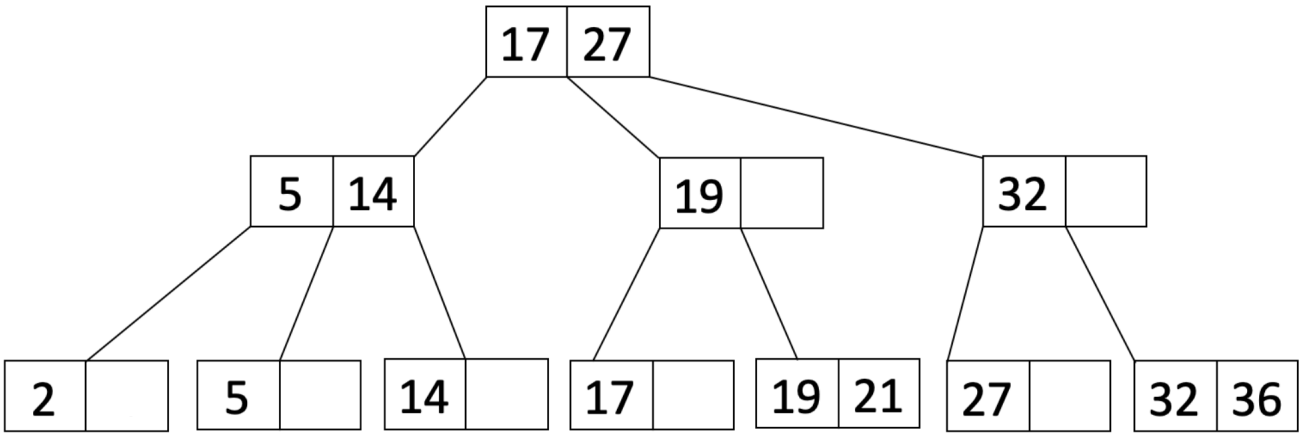
1. (5 points) Which of the following statements are true? **There may be zero, one, or more than one correct answer.**
   A. The maximum number of entries a node can hold in an order d tree is $2d + 1$.
   B. Clustered B+ trees are always more efficient (in terms of I/O's) than unclustered B+ trees for equality search queries on the key the B+ tree was constructed.
   C. When an insertion into any B+ tree causes an inner node (non-leaf node) to split, the height of the tree always increases.
   D. For at least one type of B+ tree, it is possible to have repeated keys in inner nodes (non-leaf nodes).
   E. When bulk loading B+ trees, we only need to keep the path from the root to the rightmost leaf in memory for insertions and can disregard the rest of the tree.

Consider the following order $d = 1$ B+ tree where the keys represent ages in a people table:



2. (4 points) For this question, let this B+ tree be Alternative 3 and unclustered. We want to insert Jenny who is age 21 into the table if she is not in the table already. If there are at most 10 people of the same age, what is the I/O cost for inserting Jenny into the table in **the worst case**? Assume that the data pages in the table are all 2/3 full.

Consider the same order $d = 1$ B+ tree but now Alternative 2 and clustered:



3. (1 point) What is the minimum number of inserts we can perform to change the height of the tree?

4. (2 points) If we inserted (in order) 4, 15, and 16 into this tree, how many nodes split?

5. (3 points) Using the resulting tree from the last question (after the insertions), what is the maximum number of inserts we can do without changing the height of that tree?

4 Buffer Management (12 points)

1. (5 points) Which of the following statements are true? There may be zero, one, or more than one correct statement.

    A. LRU always performs at least as well as the Clock Policy, but the Clock Policy is easier to compute.

    B. Clock Policy always performs better than LRU when the page is not in the cache and the cache is full.

    C. The dirty bit is used by the Clock Policy to determine if a page can be replaced.

    D. The pin count of a page can only be decremented by the buffer manager.

    E. MRU will often have a better hit rate than LRU if LRU is suffering from sequential flooding.

For the following parts, we are given an initially empty buffer pool with **4** buffer frames. For the access pattern:

A B C D E A E A A B C D E

Answer the following questions:

2. (1 point) What is the hit rate for MRU?
3. (1 point) What page is not in the buffer pool when MRU completes?
4. (1 point) What is the hit rate for LRU?
5. (1 point) Was the last B a hit for LRU?
6. (1 point) Was the last C a hit for LRU?
7. (1 point) Was the last D a hit for LRU?
8. (1 point) Was the last E a hit for LRU?
9. (0 points) Oranges are hybrids of mandarins and what other fruit?

5 Sort/Hash (18 points)

For parts 1 to 4, you may assume the following:

- The file size is 1000 pages
- There are 10 pages of memory available

1. (2 points) How many passes are required to merge the runs after they are sorted?

2. (1 point) *Assuming the answer to question 1 is **5***, from start to finish, what is the total number of I/Os, in pages, required to sort this file?

3. (2 points) Suppose we are given a new file to sort every 3 minutes. It takes about 1 minute to complete a pass (i.e. to scan, perform sorting/merging computation, and to write, where all buffers are in use). If we want to keep up with the rate at which we receive the files, what is the largest file size we are able to receive?

4. (1 point) True/False: If we double the buffer pool size, then we can halve the number of I/Os in pass 0.

For parts 6 to 9, you may assume the following:

- The file size is 200 pages
- There are 6 pages of memory available

A newly hired TA, Jenny, wrote a suboptimal hash function HP, which she used in the first pass of external hash on a set of data. Consequently, she ended up with partitions of size 20, 30, 30, 40, and 80 pages. Due to the lack of time, she cannot rerun the first pass with a better hash function; she will have to work with the current partitions. Not afraid to give up and take shortcuts, she reaches out to Lakshya, who, after emitting a long and audible sigh, provides her with a variety of perfect hash functions $H_{P1}, H_{P2}, ...$ that can evenly distribute values. She decides to use these for the remaining partitions needed to complete the hashing.

5. (2 points) Including the first pass over the data using HP, what is the most number of times a single record is read from disk?

6. (3 points) Calculate the total number of hash partitions at the end of the divide phase.

7. (4.5 points) Calculate the number of I/Os for the divide phase. **Do not include the I/Os for the conquer phase**

Inspired by the clean and effective code that the head TA wrote, the new TA decides to write and use a perfect granular hash function HR for the conquer phase of external hash.

8. (2.5 points) How many I/Os are performed in this phase of external hash?

Scratch Work (0 points)