CS 186 Spring 2021 Introduction to Database Systems

Midterm 1

INSTRUCTIONS

This is your exam. Complete it either at exam.cs61a.org or, if that doesn't work, by emailing course staff with your solutions before the exam deadline.

This exam is intended for the student with email address `eustyn_trinh@berkeley.edu`. If this is not your email address, notify course staff immediately, as each exam is different. Do not distribute this exam PDF even after the exam ends, as some students may be taking the exam in a different time zone.

For questions with **circular bubbles**, you should select exactly *one* choice.

○ You must choose either this option

○ Or this one, but not both!

For questions with **square checkboxes**, you may select *multiple* choices.

□ You could select this choice.

□ You could select this one too!

You may start your exam now. Your exam is due at 07:30PM Pacific Time. Go to the next page to begin.

Preliminaries

1 CS W186 Spring 2021 Midterm 1

`Do not share this exam until solutions are released.`

1.0.1 Contents:

- The midterm has 5 questions, each with multiple parts, and worth a total of 100 points.

1.0.2 Taking the exam:

- You have 110 minutes to complete the midterm.
- You may print this exam to work on it.
- For each question, submit only your final answer on examtool.
- For numerical answers, do not input any suffixes (i.e. if your answer is 5 I/Os, only input 5 and not 5 I/Os or 5 IOs).
- Make sure to save your answers in examtool at the end of the exam, although the website should autosave your answers as well.

1.0.3 Aids:

- You may use 1 page (double sided) of handwritten notes as well as a calculator.
- You must work individually on this exam.

1.0.4 Grading Notes:

- All I/Os must be written as integers. There is no such thing as 1.02 I/Os – that is actually 2 I/Os.
- 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.

(a) What is your full name?

(b) What is your student ID number?

1. **(23 points) 3 SQL Birds**

(a) **(5 points) Monotonic Queries**

A query Q is *monotonic* if whenever we add tuples to one or more of the input relations $R_1, \ldots, R_n$ to Q, the answer to Q will include all the original tuples outputted by running the query on the original relations $R_1, \ldots, R_n$.

For each of the queries below on relation `R(INT id, INT age, INT year, STRING major)`, indicate whether it is monotonic or not. You get 1 point if your answer is correct, -0.5 points if your answer is incorrect, and 0 if no answer. The minimum you can get for this problem is 0 points.

i. (1 pt) Q = `SELECT * FROM R WHERE R.age >= 21;`

○ Q is monotonic.

○ Q is not monotonic.

ii. (1 pt) Q = `SELECT COUNT(*) FROM R WHERE R.year > 2 GROUP BY R.id;`

○ Q is monotonic.

○ Q is not monotonic.

iii. (1 pt) Q = `SELECT * FROM R as r1, R as r2 WHERE r1.id = r2.id;`

○ Q is monotonic.

○ Q is not monotonic.

iv. (1 pt) Q = `SELECT * FROM R ORDER BY R.age LIMIT 1;`

○ Q is monotonic.

○ Q is not monotonic.

v. (1 pt) Q = `SELECT * FROM R GROUP BY id, age, year, major;`

○ Q is monotonic.

○ Q is not monotonic.

(b) (18 points) $QL

One of your friends just became interested in trading on the stock market after recent hype and needs your help to do analysis of companies and trades. Consider the following tables:

```
/* This table is a list of companies and their names and stock symbols. */
CREATE TABLE Companies (
    company_id      INTEGER PRIMARY KEY,
    name            VARCHAR(20) NOT NULL,
    stock_symbol    VARCHAR(4) NOT NULL
);

/* This table contains share prices of companies with a date specified by
3 columns (year, month, day) and a timestamp specified by number of seconds
since Jan 1, 1970. */
CREATE TABLE Prices (
    price_id    INTEGER PRIMARY KEY,
    company_id  INTEGER REFERENCES Companies,
    share_price FLOAT NOT NULL,
    year        INTEGER NOT NULL,   /* 1975 - 2021 */
    month       INTEGER NOT NULL,   /* 1 - 12 */
    day         INTEGER NOT NULL,   /* 1 - 31 */
    time_in_sec INTEGER NOT NULL    /* Total number of seconds since Jan 1, 1970 */
);

/* This table contains info about stock trades: the price_id, the trade type
('BUY' or 'SELL') and the number of shares traded. For those unfamiliar with the
stock market, if 3 shares of a company were bought at $4 per share, this trade
(which is a 'BUY') would be $12 total. */
CREATE TABLE Trades (
    trade_id        INTEGER PRIMARY KEY,
    price_id        INTEGER REFERENCES Prices,
    trade_type      VARCHAR(4) NOT NULL,
    shares_traded   FLOAT NOT NULL
);
```

i. (1 pt) Does the following query return the total number of ‘BUY’ trades on January 28, 2021?

```
SELECT COUNT(*)
FROM Prices INNER JOIN Trades
    ON Prices.price_id = Trades.price_id
WHERE year = 2021 AND month = 1 AND day = 28 AND trade_type = 'BUY';
```

○ Yes

○ No

ii. (1 pt) Same question but instead for this query:

```
SELECT COUNT(*)
FROM Prices INNER JOIN Trades
    ON Prices.price_id = Trades.price_id
GROUP BY year, month, day
HAVING year = 2021 AND month = 1 AND day = 28 AND trade_type = 'BUY';
```

○ Yes

○ No

iii. (1 pt) Also same question but instead for this query:

```
SELECT SUM(ones)
FROM (
    SELECT 1 AS ones
    FROM Prices INNER JOIN Trades
        ON Prices.price_id = Trades.price_id
    WHERE year = 2021 AND month = 1 AND day = 28 AND trade_type = 'BUY'
);
```

○ Yes

○ No

iv. (1 pt) Does the following query return the total number of shares traded for the company whose stock symbol is 'GME'?

```
SELECT SUM(shares_traded)
FROM Companies, Prices, Trades
WHERE Companies.company_id = Prices.company_id AND
    Prices.price_id = Trades.price_id AND
    stock_symbol = 'GME';
```

○ Yes

○ No

v. (1 pt) Same question but instead for this query:

```
SELECT SUM(shares_traded)
FROM Companies, Prices, Trades
WHERE Companies.company_id = Prices.company_id AND
    Prices.price_id = Trades.price_id
GROUP BY stock_symbol
HAVING stock_symbol = 'GME';
```

○ Yes

○ No

vi. (2 pt) Also same question but instead for this query:

```
SELECT SUM(shares_traded) FROM (
    SELECT Companies.company_id, shares_traded
    FROM Companies, Prices, Trades
    WHERE Companies.company_id = Prices.company_id AND
        Prices.price_id = Trades.price_id
    EXCEPT
    SELECT Companies.company_id, shares_traded
    FROM Companies, Prices, Trades
    WHERE Companies.company_id = Prices.company_id AND
        Prices.price_id = Trades.price_id AND
        Companies.stock_symbol != 'GME'
);
```

○ Yes

○ No

vii. **(2 pt)** We want to find the top 5 companies in terms of percent growth of their share price in the year 2020. This can be expressed as `(latest share price in 2020 - earliest share price in 2020) / earliest share price in 2020`. Return the company ids and percent growth values. You can assume all of a company's price timestamps are unique and that no two companies will have the same percent growth.

```
WITH
    earliest_share_prices AS (
        SELECT company_id, share_price
        FROM Prices p1
        WHERE year = 2020 AND _____(1)_____ <= ALL(
            SELECT _____(2)_____
            FROM Prices p2
            WHERE year = 2020 AND _____(3)_____
        )
    ),
    latest_share_prices AS (
        SELECT company_id, share_price
        FROM Prices p1
        WHERE year = 2020 AND _____(1)_____ >= ALL(
            SELECT _____(2)_____
            FROM Prices p2
            WHERE year = 2020 AND _____(3)_____
        )
    ),
    percent_growths AS (
        SELECT esp.company_id, _____(4)_____ AS percent_growth
        FROM earliest_share_prices esp, latest_share_prices lsp
        WHERE _____(5)_____
    )
SELECT company_id, percent_growth
FROM percent_growths
ORDER BY percent_growth
LIMIT 5;
```

What belongs in all the blanks labeled `(1)`?

viii. **(2 pt)** What belongs in all the blanks labeled `(2)`?

ix. **(2 pt)** What belongs in all the blanks labeled `(3)`?

x. (1 pt) What belongs in the blank labeled `(4)`?

xi. (1 pt) What belongs in the blank labeled `(5)`?

xii. (1 pt) For the following relational algebra questions, refer to `Companies` as C, `Prices` as P, and `Trades` as T.

Again, here is the database schema:

```
/* This table is a list of companies and their names and stock symbols. */
CREATE TABLE Companies (
    company_id      INTEGER PRIMARY KEY,
    name            VARCHAR(20) NOT NULL,
    stock_symbol    VARCHAR(4) NOT NULL
);

/* This table contains share prices of companies with a date specified by
3 columns (year, month, day) and a timestamp specified by number of seconds
since Jan 1, 1970. */
CREATE TABLE Prices (
    price_id    INTEGER PRIMARY KEY,
    company_id  INTEGER REFERENCES Companies,
    share_price FLOAT NOT NULL,
    year        INTEGER NOT NULL,   /* 1975 - 2021 */
    month       INTEGER NOT NULL,   /* 1 - 12 */
    day         INTEGER NOT NULL,   /* 1 - 31 */
    time_in_sec INTEGER NOT NULL    /* Total number of seconds since Jan 1, 1970 */
);

/* This table contains info about stock trades: the price_id, the trade type
('BUY' or 'SELL') and the number of shares traded. For those unfamiliar with the
stock market, if 3 shares of a company were bought at $4 per share, this trade
(which is a 'BUY') would be $12 total. */
CREATE TABLE Trades (
    trade_id        INTEGER PRIMARY KEY,
    price_id        INTEGER REFERENCES Prices,
    trade_type      VARCHAR(4) NOT NULL,
    shares_traded   FLOAT NOT NULL
);
```

Which of the following returns the stock symbols of the companies which, at any time in 2020, had a share price greater than $150?

○ $\pi_{\text{stock\_symbol}}(\sigma_{\text{year}=2020 \wedge \text{price}>150}(C \bowtie P))$

○ $\pi_{\text{stock\_symbol}}(\sigma_{\text{year}=2020}(C \bowtie P)) \cup \pi_{\text{stock\_symbol}}(\sigma_{\text{price}>150}(C \bowtie P))$

○ $\pi_{\text{stock\_symbol}}(\sigma_{\text{year}=2020}(C \bowtie P)) - \pi_{\text{stock\_symbol}}(\sigma_{\text{price}>150}(C \bowtie P))$

xiii. (2 pt) Which of the following returns the stock symbols of the companies which have greater than 10,000 shares ever traded on it?

- ○ $\pi_{\text{stock\_symbol}}(\gamma_{\text{C.company\_id}}(C \bowtie P \bowtie T)) - \pi_{\text{stock\_symbol}}(\gamma_{\text{C.company\_id,SUM(shares\_traded)>10000}}(C \bowtie P \bowtie T))$
- ○ $\pi_{\text{stock\_symbol}}(\gamma_{\text{C.company\_id,SUM(shares\_traded)>10000}}(C \bowtie P \bowtie T))$
- ○ $\pi_{\text{stock\_symbol}}(\gamma_{\text{company\_id,COUNT(shares\_traded)>10000}}(C \bowtie P \bowtie T))$

2. (22 points) Put Your Records On (Tell Me Your Favorite Song)

(a) (4 points) True or False

For the next 4 questions, indicate whether the statement is True or False.

i. (1 pt) For heap files, a delete operation on a packed page will always cost the same number of I/Os as a delete operation on an unpacked page.

○ True

○ False

ii. (1 pt) Reading a 1024KB record from disk will always take exactly twice as long as reading a 512KB record from disk.

○ True

○ False

iii. (1 pt) A bitmap in the page header is not necessary to track fixed length records stored in a packed page layout.

○ True

○ False

iv. (1 pt) The free space pointer in a slotted page layout will never point to the end of a deleted record.

○ True

○ False

(b) (10 points) Extra, Extra (IO)! Read (or Write) all about it!

Consider the following table.

```
CREATE TABLE Publications (
    article_id INTEGER PRIMARY KEY,
    journalist_id INTEGER,
    publisher VARCHAR[26] NOT NULL,
    topic CHAR[20],
    off_record BOOLEAN
);
```

For the next 5 questions, assume the following:

- *Publications* is stored in a Heap File Page Directory implementation. There are 30 data pages and each page directory header has 8 entries.
- Data pages follow the slotted page layout presented in class. The footer on each data page reserves 8 bytes total for the slot count and free space pointer. Each slot takes 8 bytes.
- Records are stored as variable length records.
- The record header contains a bitmap to track all fields that can be NULL. The bitmap is as small as possible, rounded up to the nearest byte.
- Integers and pointers are 4 bytes. Booleans are 1 byte.
- Each page is 2KB (2048 bytes).

i. (2 pt) What is the minimum size of a *Publications* record in bytes?

ii. (2 pt) What is the minimum number of *Publications* records it takes to fill up a data page? A data page is full when no more records can be inserted.

iii. (2 pt) Write an explanation for your answer to the previous 2 questions.

iv. (4 points)

For the next 2 questions, assume the queries are independent of each other; i.e. the 2nd query is run on a copy of the file that has never had the 1st query run on it. There is always at least 1 data page with enough free space to insert records. The buffer is large enough to hold all data and header pages, and starts empty **for each question**.

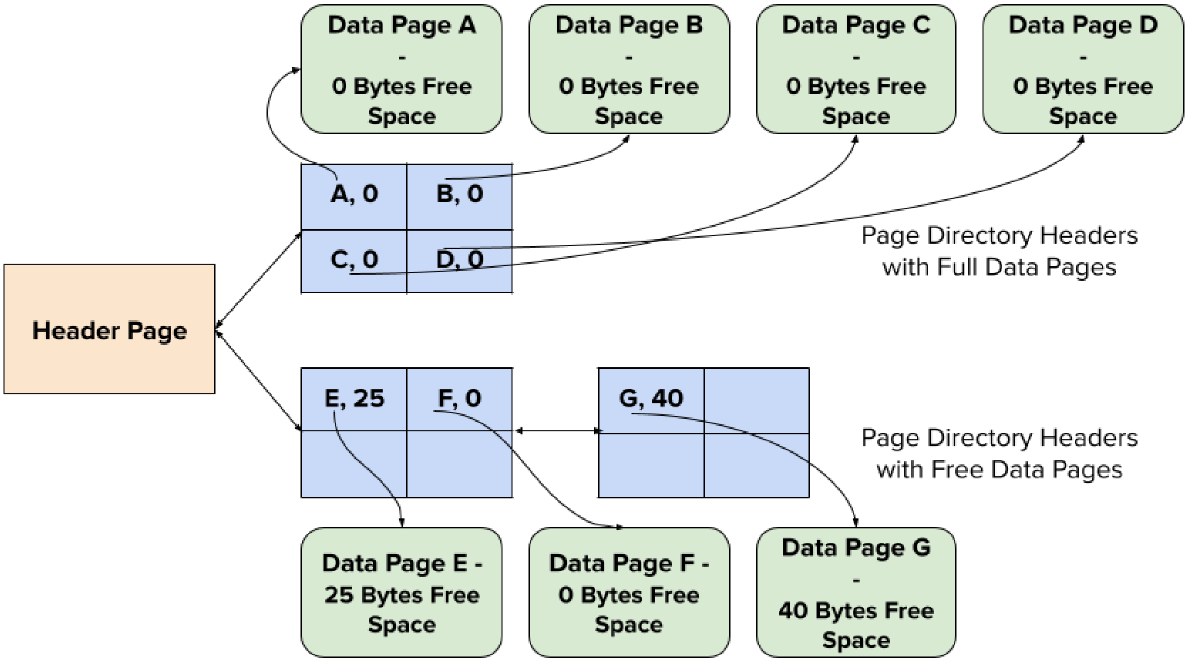
Provide the **worst** case cost in I/Os to execute the following queries.

A. **(2 pt)** `SELECT * FROM Publications WHERE topic = 'New Building' and off_record = TRUE;`

B. **(2 pt)** `INSERT INTO Publications(186, 15, 'Daily Cal', 'Student Life', FALSE);`

(c) (8 points) Inside Scoop on Heap File Design

Consider the design of a new heap file implementation called the Linked List of Page Directories (LLPD). The LLPD consists of a regular linked list with a sublist for free pages and a sublist for full pages as usual. However, the sublists contain page directory header pages, instead of data pages. Each page directory header page is implemented as shown in lecture. If all the entries in a page directory header page point to data pages that are full, the header page will be placed in the full pages sublist; otherwise, it will be placed in the free pages sublist. This means that if a page directory header is in the full list, then all of its entries must be occupied and each of those data pages have no free space. An example of the LLPD is shown below.


Data Page A - 0 Bytes Free Space
Data Page B - 0 Bytes Free Space
Data Page C - 0 Bytes Free Space
Data Page D - 0 Bytes Free Space
A, 0
B, 0
C, 0
D, 0
Page Directory Headers with Full Data Pages
Header Page
E, 25
F, 0
G, 40
Page Directory Headers with Free Data Pages
Data Page E - 25 Bytes Free Space
Data Page F - 0 Bytes Free Space
Data Page G - 40 Bytes Free Space

Sample Linked List of Page Directories

For the next 4 questions, assume the following:

- There are 25 page directory headers with full data pages and 30 page directory headers with free data pages.
- Each page directory header has 10 entries.
- There are a total of 450 data pages.
- Queries are independent of each other; i.e. the 2nd query is run on a copy of the file that has never had the 1st query run on it.
- There is always at least 1 data page with enough free space to insert records.
- The buffer is large enough to hold all data and header pages, and starts empty for each question.

Consider the following *Journalists* table, which is stored in the LLPD file described above.

```
CREATE TABLE Journalists (
    journalist_id INTEGER NOT NULL,
    first_name VARCHAR[45] NOT NULL,
    last_name VARCHAR[45] NOT NULL,
    num_articles INTEGER
);
```

i. **(2 pt)** Given the LLPD heap file for the *Journalists* table, what is the worst case cost in I/Os for checking if there is enough space to insert a record?

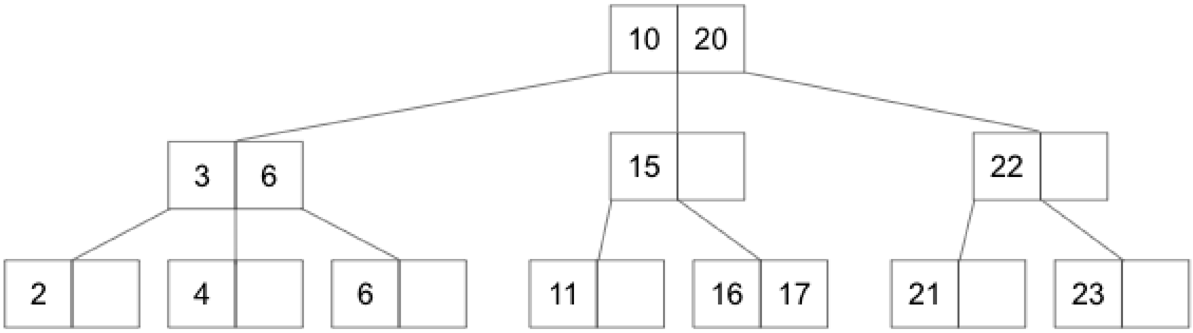
ii. **(2 pt)** What is the worst case cost in I/Os to execute the following query on the LLPD heap file for the *Journalists* table? `INSERT INTO Journalists(286, ‘Oski’, 'Undercover', 50);`

iii. **(2 pt)** Write an explanation for your answer to the previous question.

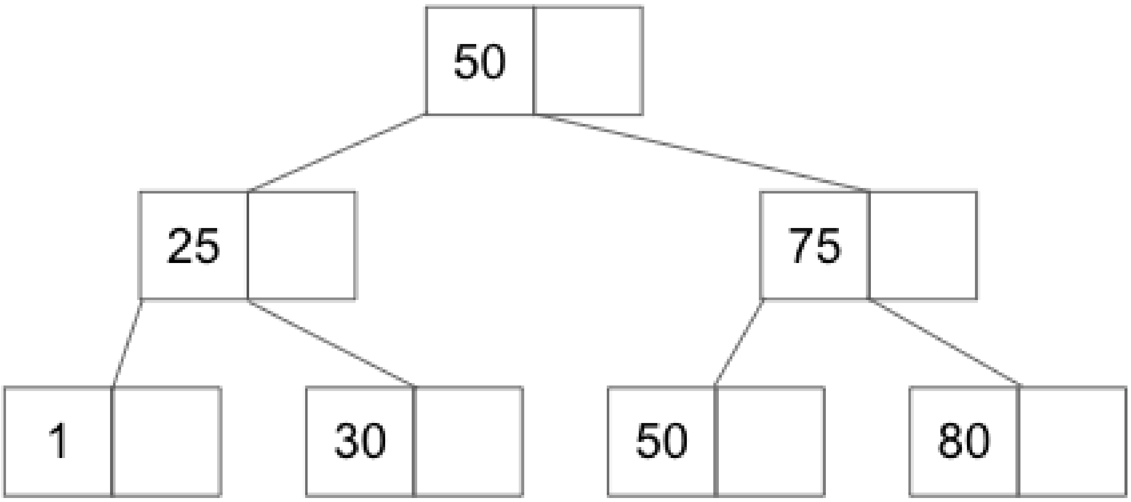
iv. **(2 pt)** State 1 change you could make to this design to further minimize the worst case I/O cost for checking if there is enough space to insert a record and explain. Your solution may not use indices or make any assumptions about the buffer manager. Limit your answers to 4 sentences.

3. (20 points) Sapphire and B+ Trees

(a) (2 pt) What is the minimum number of keys you could insert to change the height of this tree?



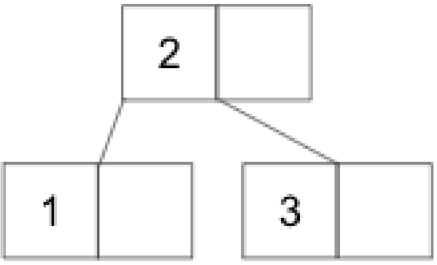
(b) (2 pt) What is the maximum number of keys you could insert without changing the height of this tree?



(c) **(2 pt)** What sequence of operations (i.e. get, put, remove) can be used to obtain the following B+ tree?

Put **"N/A"** as your answer if you believe it is not possible to obtain the following B+ tree through a sequence of operations.

Otherwise, provide your answer as a **comma-separated list** of **at most 5 operations**, i.e. put 7, get 4, put 10, etc.



(d) **(3 points)**

In this problem, we want to bulk load a B+ tree with order d = 3 with entries 1, 2, . . . , 32 with a fill factor of f = 2/3.

i. **(1 pt)** How many leaf nodes are there in the resulting B+ tree?

ii. **(1 pt)** How many inner nodes are there in the resulting B+ tree?

iii. **(1 pt)** What is the height of the resulting B+ tree?

(e) (11 points)

For this problem, suppose we have the following `students` table and the following indexes built on the `students` table. Assume each question is independent and starts off with an empty buffer pool.

```
CREATE TABLE students (
    sid INTEGER PRIMARY KEY,
    name TEXT,
    age INTEGER,
    score INTEGER
);
```

Index A:

- Alternative 1 B+ Tree on `(score, name)` with $h = 2$, $d = 4$
- Each leaf node is 1/2 full

Index B:

- Alternative 2, clustered B+ Tree on `age` with $h = 2$
- Contains 25 leaf nodes
- The first entry with `age = 20` appears on the 23rd leaf node

Index C:

- Alternative 3, unclustered B+ Tree on `sid` with $h = 2$

Additional assumptions:

- The `students` table has 81 data pages.
- There are 5 records with `score = 50`.
- There are 12 records with `age >= 20`.

- Each data page stores 8 records.

i. (2 pt) Estimate the minimum I/O cost of the following query:

```
SELECT * FROM students WHERE name = 'Alice';
```

ii. (2 pt) Estimate the minimum I/O cost of the following query:

```
SELECT * FROM students WHERE score = 50;
```

iii. (3 pt) Estimate the minimum I/O cost of the following query:

```
SELECT * FROM students WHERE age >= 20;
```

iv. **(2 pt) For this problem only, assume the following costs:** `SELECT * FROM students WHERE score = 50;` costs 10 I/Os. `SELECT * FROM students WHERE age >= 20;` costs 20 I/Os.

Using this information, estimate the minimum I/O cost of the following query:

`SELECT * FROM students WHERE score = 50 AND age >= 20;`

v. **(2 pt)** Estimate the minimum I/O cost of the following query when using Index C. Ignore the costs of maintaining the other indexes.

`UPDATE students SET score = score + 1 WHERE sid = 1;`

4. **(17 points) Buffer Management**

(a) **(5 points) True or False**

i. **(1 pt)** In the MRU eviction policy, the page that gets evicted is **always** the page most recently added to the buffer pool.

○ True

○ False

ii. **(1 pt)** In the LRU eviction policy, the page that gets evicted is **always** the page least recently added to the buffer pool.

○ True

○ False

iii. **(1 pt)** Pages in the buffer pool can only be unpinned by the buffer manager after being selected by the eviction policy.

○ True

○ False

iv. **(1 pt)** For identical access patterns, the Clock Policy will never outperform the hit rate of LRU since it is only an approximation.

○ True

○ False

v. **(1 pt)** Pinned pages can only be evicted if the page has not been dirtied.

○ True

○ False

(b) (3 points)

Consider the MRU eviction policy with 4 buffer frames and the following access pattern:

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

i. (1 pt) Which pages are in the buffer pool at the end of execution?

□ A

□ B

□ C

□ D

□ E

□ F

ii. (1 pt) How many hits occurred?

iii. (1 pt) Would the hit rate of LRU with 4 buffer frames be better than, worse than, or equally well as MRU on the given access pattern?

○ Better

○ Worse

○ Equal

(c) (4 points)

Consider the Clock eviction policy with 4 buffer frames and the following access pattern:

A, C, D (remain pinned), B, A, E, (unpin D), C, F

Assume the clock hand starts out pointing at frame 1. You can assume all pages are unpinned immediately after being accessed unless otherwise stated. For this question, assume that pages have their reference bits set to 1 upon being unpinned.

i. (2 pt) Which pages are in the buffer pool at the end of execution?

□ A

□ B

□ C

□ D

□ E

□ F

ii. (1 pt) Which frames have a reference bit of 1 at the end of the access pattern?

□ 1

□ 2

□ 3

□ 4

iii. (1 pt) How many hits occurred?

(d) (5 points)

i. (2 pt) You're analyzing the performance of your implementation of the LRU eviction policy and find that for a benchmark of 1000 page accesses it has a hit rate of 0.84. You also find that the LRU policy requires 20 ms of overhead time to update its internal state whenever a page is accessed regardless of whether it's a hit or a miss. Assuming that on average an I/O takes 500 ms, how many **seconds** does it take to process the access pattern using LRU?

ii. (1 pt) Write an explanation for your answers to the previous question.

iii. (2 pt) You have four buffer frames in memory and need to access the sequence of pages ABCDE repeatedly, in that order, 200 times, for a total of 1000 page accesses. Now, assume that on average an I/O takes 500 ms, on average how many **milliseconds** of overhead time would MRU need per access for this query to have an average completion time of 250 seconds?

5. **(18 points) Let's SORT/HASH this out!**

For all problems, assume that your **buffer size is 10 pages**. Please input only your final answer unless asked for otherwise, and do NOT include units in your answer. For questions that ask you to consider modifications to the algorithm, you should consider each modification separately from all previous questions.

(a) **(10 points) Sorting**

i. (1 pt) What is the maximum number of pages in a table that can be sorted without using external sorting?

ii. (1 pt) What is the maximum number of pages in a table that can be sorted in 2 passes of external sorting?

iii. (1 pt) What is the maximum number of pages in a table that can be sorted in 3 passes of external sorting?

iv. (1 pt) How many passes will it take to sort a table with N pages if N is in between your answers to the previous two questions? In other words, N > the maximum number of pages that can be sorted in 2 passes, but N < the maximum number of pages that can be sorted in 3 passes. You should express your answer as an integer.

v. (2 pt) How many I/Os will it take to sort a table with 500 pages?

vi. (1 pt) Jerry is building his own external algorithm. He decides to use only half of the buffers in Pass 0 of the external sorting algorithm, and excludes one buffer page during all other passes to use for non-related tasks. On average, will this result in a greater number, equal number, or fewer number of I/Os when compared to the external sorting algorithm taught in class?

○ Greater Number of I/Os

○ Fewer Number of I/Os

○ Equal Number of I/Os

vii. **(1 pt)** Jerry decides to re-build his own external algorithm. This time, he uses the same methodology taught in CS186 for the external sorting. However, for the in-memory sort, he doesn't remember how to implement efficient sorts such as merge sort, so he uses selection sort instead. On average, will this result in a greater number, equal number, or fewer number of I/Os when compared to the external sorting algorithm taught in class?

○ Greater Number of I/Os

○ Fewer Number of I/Os

○ Equal Number of I/Os

viii. **(1 pt)** Jerry writes a magic algorithm that ensures every individual page of his table will be sorted at no additional I/O cost. He then wishes to sort the entire table. On average, will this result in a greater number, equal number, or fewer number of I/Os when compared to the external sorting algorithm taught in class?

○ Greater Number of I/Os

○ Fewer Number of I/Os

○ Equal Number of I/Os

ix. **(1 pt)** Jerry needs to sort all the CS186 students based on the results of their midterm 1 scores. He observes the table and finds that many people received the same score (and it's a good one!). Suppose there are in fact 11 pages of students with duplicate scores. On average, will this high number of duplicates result in a greater number, equal number, or fewer number of I/Os when compared to the external sorting algorithm taught in class?

○ Greater Number of I/Os

○ Fewer Number of I/Os

○ Equal Number of I/Os

(b) (8 points) Hashing

i. (1 pt) Let's transition over to external hashing. Remember that we are still with a buffer size of 10 pages. How many partitions are created from Pass 0 during the hashing of a table with 120 pages?

ii. (1 pt) How many pages are in each partition after Pass 0 during the hashing of a table with 120 pages?

iii. (1 pt) Is recursive partitioning needed during the hashing of a table with 120 pages?

○ Yes

○ No

iv. (2 pt) What is the total number of I/Os needed to hash a table with 120 pages?

v. (1 pt) This time, Jerry needs to group all the CS186 students based on the results of their midterm 1 scores. He tries to write his own external hashing algorithm, and he decides to try some of his own hash functions.

In the first partitioning pass, he calculates the score mod 5 and assigns that tuple to the corresponding partition. On average, will this result in a greater number, equal number, or fewer number of I/Os when compared to the external hashing algorithm taught in class? For this question only, you may assume midterm scores are evenly distributed.

○ Greater Number of I/Os

○ Fewer Number of I/Os

○ Equal Number of I/Os

vi. (1 pt) Jerry uses an uneven hash function to construct in-memory hash tables. On average, will this result in a greater number, equal number, or fewer number of I/Os when compared to the external hashing algorithm taught in class?

○ Greater Number of I/Os

○ Fewer Number of I/Os

○ Equal Number of I/Os

vii. (1 pt) Suppose, once again, there are in fact 11 pages of students with duplicate scores. On average, will this high number of duplicates result in a greater number, equal number, or fewer number of I/Os when compared to the external hashing algorithm taught in class?

○ Greater Number of I/Os

○ Fewer Number of I/Os

○ Equal Number of I/Os

No more questions.