I SQL [10 points]

Suppose that you are running a pet store. You want to manage the information of your customers and dogs of your store. You begin with several tables as follow. For each question, there may be one, or more than one correct answer.

```
CREATE TABLE users (
userid INTEGER,
name STRING,
age INTEGER,
PRIMARY KEY (userid)
);

CREATE TABLE dogs (
dogid INTEGER,
owner INTEGER,
name STRING,
breed STRING,
age INTEGER,
PRIMARY KEY (dogid),
FOREIGN KEY (owner) REFERENCES users
);
```

1. [**3 points**] Suppose you need to know the number of dog breeds of all your dogs, select the correct SQL query:
A.

```
SELECT COUNT(*)
FROM dogs;
```

B.

```
SELECT *
FROM dogs
GROUP BY breed;
```

C.

```
SELECT COUNT(breed)
FROM dogs;
```

D.

```
SELECT COUNT(DISTINCT breed)
FROM dogs;
```

E.

```
SELECT *
FROM dogs;
```

Solution
D

2. [**3 points**] Which query should you issue to find the userid of the user with the most dogs along with the number of dogs the user owns? The query should return only one row in the case where there are multiple users with the same number of dogs.
A.

```
SELECT userid, COUNT(*) as cnt
FROM dogs, users
WHERE userid = dogid
ORDER BY cnt DESC
LIMIT 1;
```

B.

```
SELECT userid, MAX(dogid) as max
FROM dogs,users
WHERE userid = owner
GROUP BY dogid, userid
ORDER BY max DESC
LIMIT 1;
```

C.

```
SELECT userid, COUNT(*) as cnt
FROM dogs INNER JOIN users ON userid = owner
GROUP BY userid
ORDER BY cnt DESC
LIMIT 1;
```

D.

```
SELECT userid, COUNT(*) as cnt
FROM dogs LEFT OUTER JOIN users ON userid = dogid
GROUP BY userid;
```

E.

```
SELECT userid, COUNT(*) as cnt
FROM dogs,users
WHERE userid = owner
GROUP BY userid
ORDER BY cnt DESC
LIMIT 1;
```

Solution

C, E

3. [**4 points**] Please give the non-nested query that finds the most popular breed (in other words, the breed of dogs chosen by the most owners.) along with the number of dogs.

Solution

```
SELECT breed, count(*) AS cnt
FROM dogs INNER JOIN users ON userid = owner
GROUP BY breed
ORDER BY cnt
(LIMIT 1;)
```

II Disk, Buffers and Files [10 points]

1. [**4 points**] Consider the following relation:

```
CREATE TABLE products (
id INTEGER, -- cannot be NULL
stock INTEGER, -- NOT NULL
price INTEGER, -- NOT NULL
name VARCHAR(10), -- NOT NULL
category CHAR(6), -- NOT NULL
serial_number CHAR(20), -- may be NULL
PRIMARY KEY (id)
);
```

A field of type CHAR(n) consists of exactly n characters, while a field of type VARCHAR(n) consists of at most n characters (assume that a field of VARCHAR(n) takes up at most n bytes). Assume that record headers take up 8 bytes, and integers are 4 bytes long. Note that columns in a primary key cannot be NULL. Which of the following, if any, are a possible size, in bytes, for some record of the products relation, assuming a variable-length representation with a record header? There may be one, or more than one correct answer.
A. 18
B. 27
C. 38
D. 45
E. 48
F. 55

Solution
B, E, F (26-36, 46-56)

2. [**6 points**] In the following two questions, assume that we are using fixed-length records, and each record is 64 bytes long. Assume that the page header is empty, except for a bitmap when necessary, which is as small as possible, rounded up to the nearest byte.

   (a) [**3 points**] How many records can fit on a 1 KB page, assuming we use a packed page layout?

   Solution
   $1024/64 = 16$

   (b) [**3 points**] How many records can fit on a 1 KB page, assuming we use an unpacked page layout?

   Solution
   15. Because bitmap needs 15 bits which is close to 2 byte.

III File Organization [12 points]

In I/O (Input/Output) cost model for analysis, we define:
B: the number of data blocks in the file;
R: the number of records per block;
D: average time to read/write one disk block.
Please answer the following questions:

1. [**4 points**] Please compare the I/O cost of range search in heap file and sorted file? Correct answer without proper explanation will earn no scores.

Solution

- Heap file: $B \times D$.
  It needs to scan the whole file to find all the matching records.
- Sorted file: $(\log_2 B + \#pages) \times D$.
  The cost of $(\log_2 B) \times D$ is used to locate the first matching record, and $\#pages \times D$ is used to find all the matching records, with $\#pages$ being the number of pages containing the matching records.

2. [**4 points**] Please compare the I/O cost of deletion of a single record in heap file and sorted file? Correct answer without proper explanation will earn no scores.

Solution

- Heap file: $(0.5 \times B + 1) \times D$.
  The first part $(0.5 \times B) \times D$ denotes the average cost of locating the record, and the second part $1 \times D$ is the cost of writing the revised page back into the disk.
- Sorted file: $(\log_2 B + B) \times D$.
  The first part $(\log_2 B) \times D$ denotes the average cost of locating the record, and the second part $B \times D$ is the average cost of shifting the rest pages by 1 record.

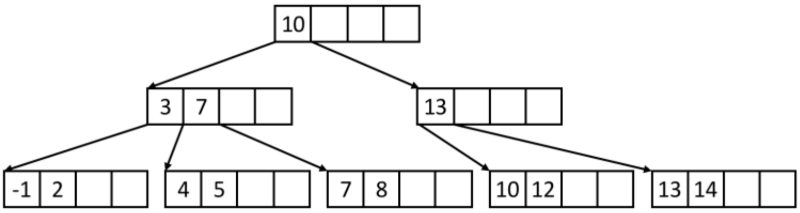
3. [**4 points**] Is there any type of file organization to reduce the I/O cost of deletion? If yes, show the type and its I/O cost.

Solution
The indexed file with B+ tree index constructed on the sorted key.
The I/O cost is $(\log_2 B) \times D$.

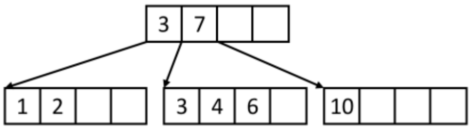
IV Indexes and B+ Trees [12 points]

1. [**2 points**] For the B+ trees below, determine which of the trees are valid B+ trees, and fill in its corresponding box on the answer sheet. Assume that there were no deletes. Note, we follow the right branch when the split key equals the search key. There may be zero, one, or more than one correct answer.

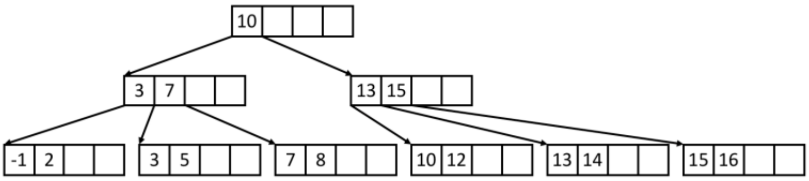
A.



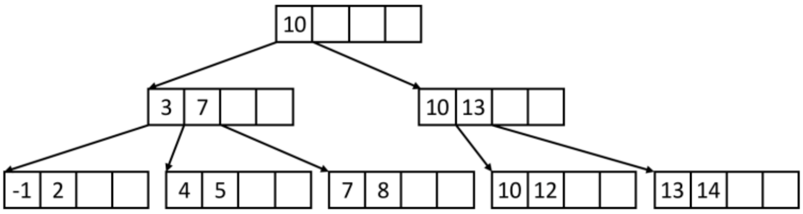
B.



C.

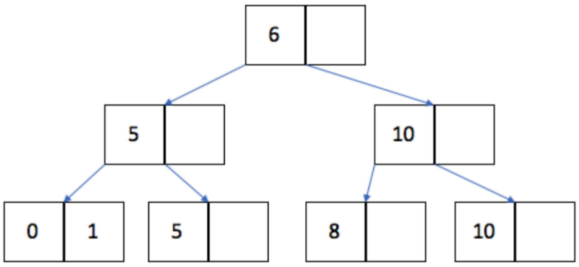


D.



Solution
C

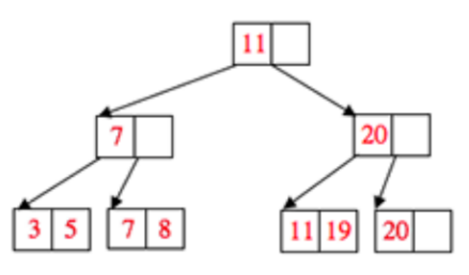
2. [**5 points**] What is the minimum number of inserts we can do that will change the height of the following tree?



Solution
4. One possible pattern is 2,3,4,4.1.

3. [**5 points**] What is the resulting B+ tree after bulk loading the keys: $3, 8, 5, 19, 20, 11, 7$? Assume we are building an order $d = 1$ index with a minimum leaf node fill factor of $2/3$.

Solution



V Buffer Management [12 points]

For any question that asks which pages are in the buffer pool at some point in time, please list the pages in alphabetical order. Assume we have 4 empty buffer frames and the following access pattern, in which pages are immediately unpinned:
A, C, D, F, D, C, B, C, A, E, C, D

1. [**4 points**] Which pages are in the buffer pool at the end if we use MRU policy?

   Solution
   C, D, E and F

2. [**4 points**] Which pages are in the buffer pool at the end if we use Clock policy? (reference bit is set when a page is loaded in buffer)

   Solution
   A, C, D and E

3. [**4 points**] Assume we have 4 empty buffer frames and the following access pattern, in which pages are immediately unpinned:
   A, B, C, D, E, A, B, C, D, E, A, B, C, D, E
   How many cache hits will occur if we use LRU policy?

   Solution
   0

VI Relational Algebra [10 points]

For the following relational algebra questions, please use the schema defined below.

Students(sid, sname, year)

Companies(cid, cname, valuation)

Recruitment(sid, cid, position, salary, status)

For each question, indicate which of the expressions gives the desired output. Note that the following table abbreviations are being used: Students → S, Companies → C, Recruitment → R. Also note that some of the expressions are invalid, in which case they should definitely not be marked in your answer.

1. [**3 points**] Find the names of all students who have received at least one recruitment offer. Note that if a student has received an offer, the status field in the Recruitment table will be the text "offer". Mark all that apply.
   A. $\pi_{\text{sname}}\ (\sigma_{\text{status =''offer''}}(S \bowtie R))$
   B. $\pi_{\text{sname}}\ (S \bowtie (\sigma_{\text{status =''offer''}}(R)))$
   C. $\sigma_{\text{status =''offer''}}\ ((\pi_{\text{sname}}\ (S)) \bowtie R)$
   D. $\pi_{\text{sname}}\ (\sigma_{\text{status =''offer''}}\ (S \bowtie (\pi_{sid}(R))))$

Solution
A, B

2. [**3 points**] Find the names of all students who have not received an offer from any company. Mark all that apply.
   A. $\pi_{\text{sname}}\ ((\sigma_{\text{status=''}of\ fer''}\ (\pi_{\text{sid}}\ (S) - \pi_{\text{sid}}\ (R))) \bowtie S)$
   B. $\pi_{\text{sname}}\ ((\pi_{\text{sid}}\ (S) - \pi_{\text{sid}}\ (\sigma_{\text{status=''offer}''}(R))) \bowtie S)$
   C. $\pi_{\text{sname}}\ (S) - \pi_{\text{sname}}\ (\sigma_{\text{status = ''offer}''}(R \bowtie S))$
   D. $\sigma_{\text{status = ''offer}''}\ ((\pi_{\text{sname}}\ (\pi_{\text{sid}}\ (S) - \pi_{\text{sid}}\ (R))) \bowtie S)$

Solution
B, C

3. [**4 points**] For every record in Recruitment, output the name of the student, name of the company he/she is being recruited for, and the position the student is being recruited for. Mark all that apply.
   A. $(\pi_{\text{sname}}\ (S)) \bowtie (\pi_{\text{cname}}\ (C)) \bowtie (\pi_{\text{position}}\ (R))$
   B. $\pi_{\text{sname, cname, position}}\ (S \bowtie C \bowtie R)$
   C. $\pi_{\text{sname, cname, position}}\ ((\pi_{\text{sid , sname}}\ (S)) \bowtie (\pi_{\text{cid, cname}}\ (C)) \bowtie (\pi_{\text{position}}\ (R)))$
   D. $\pi_{\text{sname, cname, position}}\ (S \bowtie C \bowtie (\pi_{\text{sid, cid, position}}\ (R)))$

Solution
B, D

VII Sorting and Hashing [12 points]

For this question, consider a table of Products (P), and assume:

- $[P] = 2000$ pages
- $p_c = 100$ tuples/page
- $B = 40$ pages of buffer
- In all parts, we'll use QuickSort for our internal sort algorithm.
- Include the cost of the initial scan and cost of writing output in your I/O calculations.

1. [**4 points**] First, let's sort our table of Products (P) using the external algorithm we learned in lecture.

   (a) How many passes are needed to sort this file?

   > **Solution**
   > 3.
   > $\lceil \log_{39} \lceil 2000/40 \rceil \rceil + 1 = 3$. We accepted answers written as an expression, only if the ceilings were correct (i.e. your expression must evaluate to 3 and exactly 3).

   (b) What is the I/O cost (in pages) of sorting this file?

   > **Solution**
   > 12000.
   > $2N \times 3 = 2 \times 2000 \times 3 = 12000$

2. [**4 points**] What is the largest file size (in pages) that we can sort in 2 passes?

> **Solution**
> $40 \times (40 - 1) = 1560$.
> Recall that we can sort B in one pass, $B(B-1)$ in two passes, and $B(B-1)^{(k-1)}$ in k passes.

3. [**4 points**] Suppose I want to eliminate duplicates from our (unsorted) table of Products, using external hashing. Write down the letters of true statements.

   A. De-duplicating the file using hashing can have a higher IO cost than sorting the file (without deduplicating).

   B. De-duplicating the file using hashing can have a lower IO cost than sorting the file (without deduplicating).

> **Solution**
> A, B
>
> - A is true. Consider a file with very, very skewed data, and/or a set of very, very poor hash functions, such that many repartitioning passes are needed.
> - B is true. Consider an extreme case, where the entire file consists of duplicates - there will only be one tuple to output (and only one pass over the input).

VIII Iterations and Joins [12 points]

There are two tables: Candies and Stores.

Candies(cid int, cname text, manufacturer text)

Stores(sid int, sname text, cid int)

You want to see which stores carry your favorite candies, but you are unsure of which join algorithm you should utilize to perform the query below.

```
SELECT C.cid, COUNT(*)
FROM Candies C, Stores S
WHERE C.cid = S.cid
GROUP BY C.cid;
```

We have 100 pages of Candies and 60 pages of Stores. The Candies table has 5 records per page, and the Stores table has 10 records per page. Be sure to choose the inner and outer relations such that you minimize the I/O cost. You have 25 buffer pages at your disposal.

1. [**3 points**] What is the I/O cost of a block nested loops join between Candies and Stores?

> **Solution**
> 360.
> Stores is the outer table, because $[S] < [C]$.
> $[S] + \lceil [S]/(B-2) \rceil \times [C] = 60 + \lceil 60/(25-2) \rceil \times 100 = 60 + 3 \times 100 = 360$.

2. [**3 points**] What is the I/O cost of an index nested loops join between Candies and Stores? (There is an index on Stores.cid, and it takes an average of 3 I/O's to find a matching tuple.)

> **Solution**
> 1600 or 2100.
> Note that our index is on Stores!
> $[C] + p_C \times [C] \times 3 = 100 + 5 \times 100 \times 3 = 1600$, or
> $[C] + p_C \times [C] \times 4 = 100 + 5 \times 100 \times (3+1) = 2100$.

3. [**3 points**] What is the I/O cost of a sort-merge join between Candies and Stores?

> **Solution**
> 480.
> We perform Pass 0 of external merge sort on Candies and Stores. This splits Candies into 4 runs (each contains 25 pages), and it splits Stores into 3 runs (2 runs contain 25 pages, 1 run contains 10 pages). Next, we apply the "important refinement" discussed in the lecture slides. The 7 runs can all be streamed into memory one page at a time, and we perform the merge step of external merge sort at the same time as the join step.
> $3[C] + 3[S] = 3(100) + 3(60) = 480$.

4. [**3 points**] What is the I/O cost of a grace hash join between Candies and Stores? (Assume we have hash functions that can partition the data evenly.)

> **Solution**
> 480.
> The smaller table (Stores) has more than B pages and less than 2*B pages. So, the cost is $3[C]+3[S]$, which we already calculated to be 480

IX Query Optimization [10 points]

Consider two relations Cat(age, weight, price) and Pocket(money), with 150 tuples and 100 tuples respectively. We have an index on Cat.age with 15 unique integer values uniformly distributed in the range [1, 15], an index on Cat.weight with 30 unique float values uniformly distributed in the range [2001, 5000], an index on Cat.price with 10 unique integer values uniformly distributed in the range [11, 20], and an index on Pocket.money with 15 unique integer values uniformly distributed in the range [11, 25].
Use selectivity estimation to estimate the number of tuples produced by the following queries.

(a) **[3 point]** SELECT * FROM Cat

Solution
150: Select all tuples.

(b) **[3 points]** SELECT * FROM Cat WHERE age $\geq$ 10

Solution
60:
Selectivity $= \frac{max-v}{max-min+1} + \frac{1}{\#\ distinct\ values} = \frac{5}{15} + \frac{1}{15} = \frac{2}{5}$,
$150 \times \frac{2}{5} = 60$.

(c) **[4 points]** SELECT * FROM Cat WHERE age < 5 AND weight $\leq$ 3000

Solution
13:
Selectivity $=$ Sel(age < 5) $\times$ Sel(weight ≤ 3000) $= \frac{4}{15} \times \left(1 - \frac{2000}{3000}\right) = \frac{4}{45}$,
$150 \times \frac{4}{45} = 13.333 = 13$.