Department of Electrical Engineering and Computer Science

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

6.830 Database Systems: Fall 2021 Quiz I

There are 13 questions and 14 pages in this quiz booklet. To receive credit for a question, answer it according to the instructions given. *You can receive partial credit on questions.* You have **75 minutes** to answer the questions.

Write your name on this cover sheet AND at the bottom of each page of this booklet.

Some questions may be harder than others. Attack them in the order that allows you to make the most progress. If you find a question ambiguous, be sure to write down any assumptions you make. Be neat. If we can't understand your answer, we can't give you credit!

THIS IS AN OPEN BOOK, OPEN NOTES QUIZ.
LAPTOPS MAY BE USED; NO PHONES OR INTERNET ALLOWED.

Do not write in the boxes below

1-4 (xx/36)	**5-8 (xx/32)**	**9-11 (xx/12)**	**12-13 (xx/20)**	**Total (xx/100)**

Name:

I MonetDB/X-100

1. [8 points]: According to the paper "MonetDB/X100: Hyper-Pipelining Query Execution", which of the following are reasons that conventional query executors like MySQL and Oracle are inefficient compared to an in-memory execution system?

(Circle all that apply.)

A. Because they are row stores, they support inserts and updates much less efficiently than the X100 system.

B. They use record-at-a-time processing, which has a high overhead and limits the ability of the CPU to accurately predict branches and prefetch data from memory, result in memory stalls and pipeline flushes.

C. Because they are row stores, they have to compute variable offsets into records, resulting in many slow random memory accesses.

D. Their query planners are error prone, and lack many of the sophisticated statistics that the X100 system has.

B C

II SQL

Consider the following pairs of SQL queries. Your job is to figure out which of them are semantically equivalent (i.e., compute the same answer for all possible inputs).

2. [12 points]: For each pair, indicate if they are equivalent. If not, briefly describe a situation or example in which they would produce different answers.

(Select "Equivalent" or "Not Equivalent" for each pair.)

Query 1

```
SELECT x,m
FROM  A,
  (SELECT MAX(B.b) AS m
   FROM B)
as C
WHERE A.a = C.m
GROUP BY x,m
```

Query 2

```
SELECT  x,MAX(B.b)
FROM A, B
WHERE A.a = B.b
GROUP BY x
```

(A) Equivalent   Not Equivalent

If "Not Equivalent", briefly describe why they are not equivalent, or give an example input where they would differ.

Name:

Not Equivalent. Query 2 selects the MAX from each group whereas Query 1 selects the MAX from the entire table of B

Name:

Query 1

```
-- A.b is a unique non-null
-- primary key for A
SELECT T.a
FROM (
    SELECT A.a, B.c
    FROM A LEFT OUTER JOIN B
    ON A.b = B.c
) as T
WHERE T.c IS NOT NULL
```

Query 2

```
SELECT A.a
FROM A JOIN B
ON A.b = B.c
```

(B) Equivalent Not Equivalent

If "Not Equivalent", briefly describe why they are not equivalent, or give an example input where they would differ.

Equivalent

Name:

III Histograms and Query Planning

Suppose you are evaluating a predicate of the form $x > v$ over a single column of a table with the histogram of values shown in Figure 1.

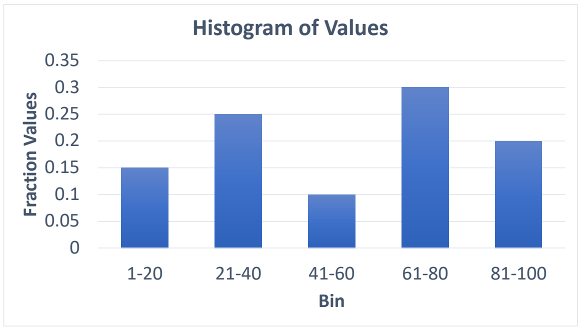


Figure 1: Histogram of Values

3. [8 points]: Assuming values of x are uniformly distributed within each bin, if you run the query $x >= 50$, based on the histogram, what fraction of records do you estimate will satisfy this query?

Accepted Answers:

$$0.1 \times \frac{10}{20} + 0.3 + 0.2 = 0.55$$

or

$$0.1 \times \frac{11}{20} + 0.3 + 0.2 = 0.555$$

if assuming integers

4. [8 points]: Suppose you run this query and you find that actual fraction of records is off by more than 10%. What are possible explanations for this:

(Choose all that apply.)

A. The data is not actually uniformly distributed within some of the bins.

B. The database has out of date statistics.

C. x is highly correlated with some other attribute y.

D. There are many records with value $x = 45$.

Name:

Accepted Answers: B if assuming 10% is of the entire database. A, B, D if assuming 10% is $\frac{|actual - expected|}{actual}$. Grading maximizes your partial credit based on the edit distance between your answer and correct answers.

Name:

IV Indexing

Suppose you have a table T, and you frequently run two queries of the form:

A. `select count(*) from T where a > x`

B. `select c from T where b = y`

Here a,b, and c are specific columns in the database, but x and y vary from query to query. The columns are uncorrelated. Suppose T is much larger than memory, and your database supports clustered and unclustered B+tree indexes, and that the table can only be clustered on one index. The queries are run with about the same frequency.

5. [10 points]: If the predicate `a > x` selects about 10% of the records in the database, and `b` is a primary key of T, which index or indexes would you recommend building on T? If an index will not be used, you should not create it.

(Write your answer in the space below.)

Clustered index on a as it has a range scan and unclustered on b since it is a point query.

Name:

V Postgres Plans

Consider the following SQL query and Postgres query plan:

```
SELECT dept.dno,
       count(*),
       sum(sal)
FROM dept
JOIN emp ON emp.dno = dept.dno
JOIN kids ON kids.eno = emp.eno
WHERE emp.sal > 1000
GROUP BY dept.dno
ORDER BY dept.dno DESC;
```

```
                                                QUERY PLAN
-------------------------------------------------------------------------------------------------
 GroupAggregate  (cost=808618.95..834191.29 rows=100001 width=20)
   Group Key: dept.dno
   ->  Sort  (cost=808618.95..814762.03 rows=2457233 width=8)
         Sort Key: dept.dno DESC
         ->  Hash Join  (cost=334161.30..480607.82 rows=2457233 width=8)
               Hash Cond: (emp.dno = dept.dno)
               ->  Hash Join  (cost=331077.28..451484.29 rows=2457233 width=8)
                     Hash Cond: (kids.eno = emp.eno)
                     ->  Seq Scan on kids  (cost=0.00..49099.01 rows=3000001 width=4)
                     ->  Hash  (cost=188696.44..188696.44 rows=8190867 width=12)
                           ->  Seq Scan on emp  (cost=0.00..188696.44 rows=8190867 width=12)
                                 Filter: (sal > 1000)
               ->  Hash  (cost=1443.01..1443.01 rows=100001 width=4)
                     ->  Seq Scan on dept  (cost=0.00..1443.01 rows=100001 width=4)
(14 rows)
```

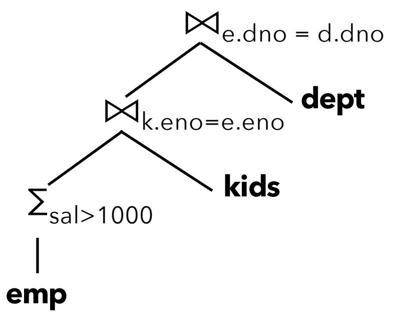
Name:

6. [8 points]:

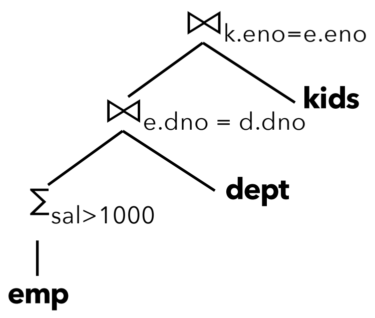
Which of the following relational algebra expressions captures the order of operations in the above plan (omitting the GROUP BY, ORDER BY, and aggregate)?

(Choose the best answer.)

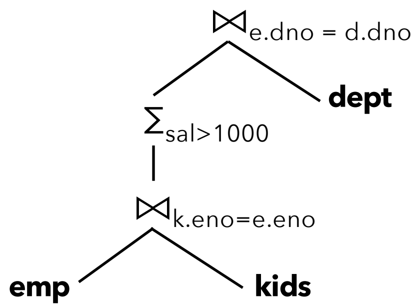
A.



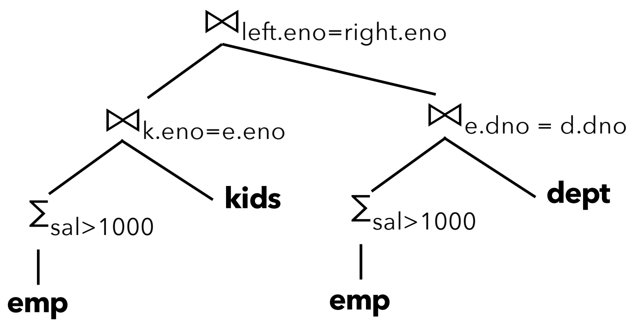
B.



C.



D.



A

7. [6 points]:

Postgres has two ways to compute GROUP BY / COUNT aggregates: it can either build a hash table on the grouping column, hashing each record into the hash bucket for its group and incrementing a counter for that record, or it can sort records on the grouping column and then process the records one group at a time. Which of these two options does it choose for this query? Briefly explain why it chooses this option:

(Circle one, and write your explanation.)

Hash **Sort** Sort Reason: The data needs to be sorted at the end

Name:

8. [8 points]:

Assuming that Postgres runs dynamic programming as in the Selinger optimizer, perfectly estimates all costs and cardinalities, and can choose between sort-merge and hash joins, based on the EXPLAIN output above, which of the following statements are definitely true:

(Circle all that apply.)

A. The cardinality of kids is less than the cardinality of emp.

B. The selectivity of the predicate on emp.sal is about 10%.

C. The optimal way to join emp (e), dept (d), and kids (k) is $\bowtie(\bowtie(d,e),k)$.

D. Replacing the topmost hash join with a sort-merge join would not improve performance.

A D

VI Column Stores

You have a table T stored in columnar format with 3 integer columns, c_1, c_2, and c_3. c_1 takes on two values, 0 and 1, uniformly distributed, while c_2 and c_3 are uniformly distributed from 1 to 100 (inclusive). The 3 columns are independent (not correlated). Integers are stored in 8 bytes. The table contains 1 million records.

You sort your database in c_1,c_2 order. Your database supports run length encoding or zip encoding of each column. Run length encoding encodes each run of consecutive values as (run length, value), including length 1 runs. The run length is an (8 byte) integer. Zip encoding is equivalent to running a standard compression tool, like zip or gzip, on the column.

Your job is to choose the compression method for each column that will minimize the overall storage space of the database.

9. [4 points]: What method would you choose for storing c_1 on disk? If you selected RLE, estimate the number of bytes it will require to store.

(Choose one, and, if applicable, write your estimate.)

RLE Zip

Approximate space estimate (if applicable):

RLE. The data is sorted on c1. c1 takes 2 unique values. There will be 2 runs. Each run takes 16 bytes - 8 for the value and 8 for the run length. Thus, 2 * 16 = 32 bytes in total.

10. [4 points]:

What method would you choose for storing c_2 on disk? If you selected RLE, estimate the number of bytes it will require to store.

(Choose one, and, if applicable, write your estimate.)

RLE Zip

Approximate space estimate (if applicable):

RLE. The data is sorted on c1, then c2. c1 takes 2 unique values. c2 takes 100 unique values. There will be 100 runs for each value of c1, thus 200 runs overall. Thus, 200 * 16 = 3200 bytes in total.

11. [4 points]:

What method would you choose for storing c_3 on disk? If you selected RLE, estimate the number of bytes it will require to store.

Name:

(Choose one, and, if applicable, write your estimate.)

RLE Zip

Approximate space estimate (if applicable):

Zip.

VII Schema Design

You are building a database to record sightings of birds on islands off the coast of Maine by people who sight birds ("birders"). Your database needs to record the following information:

- For each sighting, the bird species, the time and date of the sighting, the island where it was sighted, and the birder who sighted it.
- For each island, its name, its geographic coordinates as lat / lon, and its area in square footage.
- For each birder, their id, name, and address, and their state of residence.
- For each state, its name, population, and the count of sightings of each bird species by birders who reside in that state.

Further, you are given that:

- Birders may sight multiple birds on multiple islands
- Each sighting happens on one island
- Sighting counts are incremented immediately when a sighting is made; you do not need to worry about inconsistency between the entries in the sightings table and the sighting counts

12. [10 points]:

Write down a redundancy-free schema to store this database. Use the notation:

```
    table: (table_key*, f1, f2->t2.t2_key, ... )
```

Where * indicates the primary key, and `f2->t2.t2_key` indicates that `f2` is a foreign key reference to `t2_key`.

(Write your answer in the space below.)

The key observation about this problem is that a separate table with the sightings for a particular state (without that state's population) is needed.

sightings:(s_id*, species, time, date, island->islands.i_id, birder->birders.b_id)

islands:(i_id*, name, lat, lon, area)

birders:(b_id*, name, address, state->states.st_id)

states:(st_id*, name, population)

state_sightings_counts:(st_id*, species*, sighting_count)

Name:

13. [8 points]:

Now suppose some of the birds are tagged with a small radio transmitter, such that some of the sightings are of a specific bird, with a specific radio identifier. How would you modify your database to support this?

Since the problem doesn't specify any additional metadata about specific birds, it's sufficient to add a single NULLable column to the sightings table to record the radio identifier if present.

sightings:(s_id*, species, time, date, island->islands.i_id, birder->birders.b_id, transmitter_id)

We also accepted answers that created a separate table to hold specific sightings and added a NULLable foreign key reference to that table in sightings.

End of Quiz I!

Name: