CS145 Fall 2020 Homework 1

September 22, 2020

Context

This homework is to help familiarize you with the systems primer material from the first lecture and to give you a sense of the scale of working with data. A reminder that homework is **optional** this quarter.

Be sure to read the question carefully to use the correct units.

Assume the following numbers for hardware performance:

- 1 kB = 1000 Bytes, 1 MB = 1000 kB, etc.
- Disk blocks have size 64KB (contiguous)
- DB blocks have size 64MB (contiguous)
- RAM access (seek) takes 20ns
- Hard disk access (seek) takes 10ms
- Network transfer takes 1us
- RAM transfer speed is 100GB/sec
- Disk transfer speed is 100MB/sec

Section 1 will be on Friday, 9/25 from 9:30 AM — 10:20 AM.

Problem 1

You are designing a system to crowdsource student evaluations of college courses (very much like Carta). As part of this system, you want to store a table of reviews containing the following information (for each review):

- Review Date - Date (3 bytes)
- Academic Year - int32
- Academic Quarter - char[10]
- Course ID - char[5]
- Rating (0.0 to 5.0) - float32
- Grade in the course - char[2]
- Estimated Hours Per Week - int32
- Review (text) - char[224]

Question 1.1

What is the size of each row **in bytes**?

Question 1.2

Assume that this data is stored on a hard disk in disk blocks and the disk blocks are grouped in DB blocks.

How many rows can be stored per disk block?

Question 1.3

How many rows can be stored per DB block?

Question 1.4

Relevant information:

- According to the registrar, this quarter there are 16500 students (undergrad and graduate) enrolled at Stanford.
- Of these, about 1500 are grad students finishing their dissertations, which will be excluded.

- Of the remainder, some are grad students who might be doing research full-time and not taking classes. Let's ignore this and approximate that there are exactly 15,000 class-taking students at Stanford.
- Let's assume that on average each class-taking student takes 3 classes per quarter (including the summer quarter).
- Let us also assume that writing course evaluations is optional and 50% of students who take a course will write an evaluation. (Note: If a student chooses not to write an evaluation, no row is added to the table.)

After 10 years (40 quarters), how large **in MB** will the table of course reviews be? Round your answer to 1 decimal place.

Question 1.5

How many DB blocks would be needed to store the table of course reviews?

Question 1.6

Now, let's say you would like to read from your table and retrieve a particular evaluation (row). Let's make the following assumptions about the hardware that is used to store the table:

- Your data is all stored on hard disks
- Use the numbers given in the notes/instructions for all calculations

How long would it take **in hours** to retrieve an evaluation (row) if the table rows are stored randomly on disk (we must seek and scan every row)? Round your answer to 1 decimal place.

Question 1.7

How long would it take **in seconds** if the rows are grouped in **disk** blocks (which are randomly stored on disk)? Round your answer to 3 decimal places.

Problem 2

You're the leader of the Infrastructure team of a popular-enough startup. You're concerned with performance metrics for a particular table that is queried frequently. The table has the following specs,

- **Row size:** 64KB
- **Number of rows:** $5 * 10^7$
- **Total Data:** Number of rows * Row size $= 3200GB = 3.2TB$

The table is stored on a system with the following specs,

- **RAM:** 64GB
- **Hard Disk space:** 10TB

The system receives numerous queries each second; each query consists of fetching some random row of the table. For the purpose of this problem, let's assume parsing and transferring queries take **zero** time. We'll also assume seeks in RAM take **zero** time.

Note: Remember that the average time for finding a record during a full scan is **half** of the maximum time.

Question 2.1

The current architecture involves having all the data stored on the system's hard disk. All **rows** are randomly stored on the disk. Any fetch queries need to fetch the associated data from there.

What is the **average response time** in **secs** for a query, i.e., time to fetch a row? Assume that all rows are equally likely to be queried.

Question 2.2

While you're surprised at the high response time, an Analytics expert drops by your office. She mentions that a fixed **1%** of the table rows are responsible for **90%** of the query traffic.

Would you suggest any change to the current architecture given this information? State your suggestion concisely. What's the **average response time** in **secs** after your suggestion?

Notes:

- Assume the table rows in the described 1% are equally responsible for 90% of the query traffic. The remaining 99% of rows have equally likely queries.
- In order to reduce calculations, assume the architecture change we're looking for **does not** involve DB Blocks. Try to think of another way to reduce response time using other resources.

Hint: RAM has more uses than just running Chrome.

Problem 3

Imagine you are designing a table to store recent transactions for an online shopping platform and there are **1 trillion** transactions. You want to record the following information:

- user id
- user name
- item id
- item name
- transaction id
- amount of money ($) for the transaction (e.g. $4.11, $670.50, etc)

Assume there are **1 billion** users, and **1 billion** items for sale on the platform. The longest string for user and item names contain 64 characters. You should consider proper data types listed below: byte, short, int, long, float, double, boolean, char.

Question 3.1

What is the size of each row **in bytes**? Think about the size of each column by selecting proper data types. You need to select the most suitable data type for each column by considering efficiency.

Question 3.2

What data type should you use for each column? You need to fill one of the following data types: byte, short, int, long, float, double, boolean, char. You are not required to add the number of the data types used for the column. For instance, you only need to put char for char[10].

What is the most appropriate data type for the following column: User ID?

Question 3.3

What is the most appropriate data type for the following column: User Name?

Question 3.4

What is the most appropriate data type for the following column: Item ID?

Question 3.5

What is the most appropriate data type for the following column: Item Name?

Question 3.6

What is the most appropriate data type for the following column: Transaction ID?

Question 3.7

What is the most appropriate data type for the following column: Amount of money?

Question 3.8

What is the size of the table **in TB**? (1 point)

Problem 4

This question follows from question 3. For this question, assume that the size of the table is 200 TB.

Question 4.1

How long **in seconds** will it take to read the whole table from RAM?

Question 4.2

How long **in days** (round to nearest integer) will it take to read the whole table from disk if each row of the table is stored randomly in the disk?

Question 4.3

How long **in days** (round to nearest integer) will it take to read the whole table from disk if the table is stored in DB blocks? (1 point)

Question 4.4

What is the cost **in dollars** for saving the table in RAM? Assume RAM costs $6000/TB.

Question 4.5

What is the cost **in dollars**for saving the table in disk? Assume disk space costs $100/TB.

Problem 5

You have decided to start a new e-commerce site that you anticipate will host billions of products and you hope millions of users. You realize you will need a database system to keep track of all your data.

Question 5.1

What tables might you need for this? For example, a table to log each order a user placed for a product might be a good idea. List at least two other tables that you want to include in your design. (There are many correct answers).

Question 5.2

Let's calculate the size of one of our tables. Consider the above example of a table to log orders. We would like to keep track of the following:

- Order ID: int64
- Product ID: see part 6.2
- User ID: see part 6.3
- Quantity: int32
- Timestamp: 4 bytes
- IP address: 4 bytes
- Mailing address: char[100]

We want the ability to host 5 billion products. How many **bits** should the Product ID be to store a unique ID for each product?
Given our answer above, what data type should we use?

- tinyint (1 byte)
- smallint (2 byte)
- int (int32 – 4 bytes)
- bigint (int64 – 8 bytes)

Question 5.3

We want the ability to account for 1 billion users. How many **bits** should the User ID be to store a unique ID for each user?
Given our answer above, what data type should we use?

- tinyint (1 byte)

- smallint (2 byte)
- int (int32 – 4 bytes)
- bigint (int64 – 8 bytes)

Question 5.4

Given the above and your answers for 5.2 and 5.3, how big is one row of our table (one record) **in bytes**?

Question 5.5

How big is the entire table **in MB** if we assume that we store the data for a week, and we receive 100 million orders in a day?

Question 5.6

Let's calculate the time it takes to perform operations on our database. Let's assume that our table is **actually 10 GB** (no matter what you got for an answer in part 5.4). Use the values given in the instructions for all calculations.
How much time does it take **in milliseconds** to look up a record if our table is in RAM?

Question 5.7

How much time does it take **in days** (round to nearest day) to look up a record on disk if all records are in random locations?

Question 5.8

Now let's say you divide your data into blocks so that each block is 64 MB. The blocks are still scattered randomly **on disk**. How long would it take in seconds (round to nearest second) to look up a record then?

Question 5.9

Now, let's think about scale. Use the values given in the instructions for all calculations.
If we had 10 machines, how would this impact the speed of looking up one record? How many times faster would look up be?

Question 5.10

What if the data was stored on another machine? How long would it **in milliseconds** take to get that data if that other machine had the data readily available in RAM (round to nearest millisecond)?