DSCI 551 Spring 2024 Mid Exam – 1
[Confidential DO NOT SHARE OUTSIDE CLASS]

1. [Firebase, 25 points] Consider distributing student information among two Firebase databases. Suppose the endpoints (URL) of two databases are: https://db1/ and https://db2/.
Suppose every student has an ID attribute and students whose ID ends with odd numbers will be stored in db1 and students whose ID ends with even numbers will be stored in db2.

   For each of the following requests, write curl commands to add/retrieve the necessary information to/from appropriate databases to answer the request. Note that your curl requests should not download data that are irrelevant to the request. You can assume that all necessary indices have been pre-created.

   5 points for each sub-question.

   Q1

   1) Add a new student to the database with the following information: id = 101, name = john smith, and age = 25.

      ```
      curl -X PUT https://db1/students/101.json -d '{ "name": "john smith", "age": 25}'
      ```

   Q2

   2) Add a new student to the database with the following information: id = 102, name = mary smith, and age = 26.

      ```
      curl -X PUT  https://db2/students/102.json -d '{ "name": "mary smith", "age": 26}'
      ```

   Q3

   3) Update the age of student 101 to 28 and add a new attribute gender = 'male'.

      ```
      curl -X PATCH https://db1/students/101.json -d '{"age": 28, "gender": "male"}'
      ```

   Q4

   4) Find name of student 101.

      ```
      curl -X GET https://db1/students/101/name.json
      ```

   Q5

   5) Find names of all students who are at least 25 years old.

      ```
      # For db1
      curl -X GET "https://db1/students.json?orderBy=\"age\"&startAt=25"
      # For db2
      curl -X GET "https://db2/students.json?orderBy=\"age\"&startAt=25"
      ```

2. [Storage system, 25 points]
   a. [20 points] Consider a hard drive with the following characteristics:
      - 1024 cylinders
      - 16 heads
      - 128 sectors/per track

- Sector size: 4KB
- 10,000 RPM
- Max seek time: 15ms
- Maximum bandwidth: 64MB/sec

Note in this question, you can assume that 1000 KB is (roughly) 1MB.

5 points for each sub-question.

Q6

1) What is the capacity (in GB) of the drive? Show your derivation.

capacity = 4 * 128 * 1024 * 16
= 8GB (or 8.388GB)

Q7

2) Ignore latency, how much time is needed to read an entire track of data? Show your derivation.

T read = track's data size / maximum bandwidth
= (128 * 4KB) / 64MB/sec
= 8ms (or 0.008s, 7.812ms, 0.0078s)

Q8

3) Now consider reading two random tracks of data. For each track, read the entire track of data. What will be the total completion time? Show your derivation. Note that the disk head might be in a random position before it is ready to read the first track. Also assume, to read an entire track, the disk head can start by reading any sector of the track. You can ignore the rotational latency for the head to move to the beginning of the sector.

complete time = 2 * (T seek + T read)
T seek = Avg. seek time = 15 * (⅓) = 5ms
T read = 8 ms (or 0.008s, 7.812ms, 0.0078s)
complete time = 2 * (5 + 8) = 26 ms (or 0.026s, 25.624ms, 0.0256s)

Q9

4) Consider reading data located in two random tracks. But for each track, we only need to read 64 sectors of data sequentially laid out on that track. What is the completion time now? Show your derivation.

complete time = 2 * (T seek + T rotation + T read)
T seek = Avg. seek time = 15 * (⅓) = 5ms
T rotation = Avg. rotational latency = 60,000ms/10,000r * (½) = 3ms
T read = (64 * 4KB) / 64MB/sec) = 4ms (or 0.004s, 3.906ms, 0.0039s)
complete time = 2 * (5 + 3 + 4) = 24ms (or 0.024s, 23.812ms, 0.0238s)

Q10

b. [5 points] Explain why writing in SSD needs to be performed on the page level (instead of individual cell) and erase needs to be performed on the block level (instead of individual pages).

- In an SSD, to write data by changing a bit from '1' to '0', a high positive voltage is applied to the control gate. This voltage change attracts electrons from the channel to the floating gate through quantum tunneling. Writing is done at the page level because it simultaneously alters the charge in multiple cells, which cannot be individually isolated without affecting others.

- To erase data, changing a bit from '0' to '1', a much higher negative voltage is needed to remove electrons from the floating gate. Erasure is performed at the block level, because this high voltage can stress and damage surrounding cells, so the integrity of the entire block is preserved by erasing it all at once, which also mitigates wear on the SSD.

3. [HDFS, 25 points] Suppose now we store the student information in HDFS with the replication factor set to 3. Suppose the information is stored in students.csv locally and will be uploaded to HDFS under the '/data/dsci551' directory. Suppose students.csv is 128MB.

Q11

1) [6 points] Suppose the file system is empty now (no files/directories). Write hdfs commands to create necessary folders and load student data to HDFS.

```
#Creating the necessary folders in HDFS using mkdir command:
hdfs dfs -mkdir /data
hdfs dfs -mkdir /data/dsci551

#Loading students.csv into the new folder in HDFS using put command:
hdfs dfs -put students.csv /data/dsci551
```

Q12

2) [6 points] Explain the pipeline processing HDFS to write data into a file. How many data nodes will be participating in the process?

- **Client Request**: The process starts with a client request to write a file to HDFS.
- **NameNode Assignment**: The client contacts the NameNode, which manages the file system metadata. The NameNode assigns DataNodes to store the file's blocks and informs the client.
- **Writing and Replication**: The client writes the data to the first assigned DataNode. This DataNode then replicates the data to the second DataNode, and the process continues until all replicas (based on the replication factor) are created.
- **Confirmation**: After the last DataNode in the pipeline writes the data, it sends a confirmation back to the client via the other DataNodes.

Since students.csv is 128MB (matching the default block size in HDFS) and the replication factor is 3, three DataNodes will be involved in this process. Each DataNode stores one replica of the data block.

3) [6 points] Suppose the package size is 128KB.

Q13

a. How many data packets will be needed for writing students.csv?

- number of data packets = 128 MB / 128 KB = 128 * 1024 KB / 128 KB = **1024**
- alternate solution: # of packets = 128 MB / 128 KB = 128 * 1000 KB / 128 KB = **1000**

Q14

b. How many control messages, data messages, and acknowledgement messages will be needed for the pipelining? Explain the purpose of each control message.

For pipelining, we typically need at least **2 control messages**, one to initiate the connection (setup message) and the other to signify the end of transmission (close message). Depending on the protocol, there might be additional control messages for error handling or flow control.

As calculated, since there are 1024 data packets, **1024 data messages** are needed.

For each data packet sent, an acknowledgment message is received. In addition, the 2 control messages also needed to be acknowledged. So there will be a total of **1026 acknowledgement messages**.

Q15

4) [7 points] Now suppose we need to read the first 100MB of students.csv from HDFS. State the specific value for each argument of getBlockLocations RPC for this read. What will this RPC call return?

Arguments for the getBlockLocations RPC:

- source (file name): '/data/dsci551/students.csv'
- offset: 0 (start reading from the beginning of the file)
- length: 100MB (first 100 MB of the file) accept units other than MB

This RPC call will return a located block which contains the locations of the 3 DataNodes holding the replicas of the block.

4. [XML & UTF-8 encoding, 25 points] Now suppose the student information is stored in an XML file as follows:

```
<students>
    <student id = "101">
        <name>john smith</name>
        <age>25</age>
        <gpa/>
    </student>
    …
<students>
```

a. [20 points] For each of the following questions, write XPath expression(s) to find the answer to the question. 4 points each question.

Q16

1) Find name of student 101. Return actual names, not elements.

/students/student[@id='101']/name/text()

Alternates:
/students/student[@id=”101”]/name/text()
/students/student[@id=101]/name/text()
/students/student[1]/name/text()

/students/student can also be written as //student

Q17

2) Find names of all students who are at least 25 years old. Return actual names, not elements.

/students/student[age>=25]/name/text()

Q18

3) Find names of students whose GPA (elements) are empty (see the above data for an example). Return actual names, not elements.

/students/student[gpa[not(node())]/name/text()

Alternates:
/students/student[not(gpa/node())]/name/text()
/students/student[gpa[not(text())]]/name/text()
/students/student[not(gpa/text())]/name/text()
/students/student[gpa[not(*)]]/name/text()
/students/student[gpa=""]/name/text()

Q19

4) Find ids of students whose name contains 'smith'.

/students/student[contains(name, 'smith')]/@id

Alternate: /students/student[contains(name, 'smith')]/@id/text()

Q20

5) Find names of students whose age is between 25 and 35, inclusive. Return actual names, not elements.

/students/student[age>=25 and age<=35]/name/text()

Q21

b. [5 points] Suppose XML file is encoded in UTF-8 and we need to store a student whose name contains a special character whose UTF code point is 1234. What is the UTF-8 encoding of the character? Show your encoding in both binary and hexadecimal format.

UTF code point 1234 is represented as 0001 0010 0011 0100.
Binary format: 1110 0001 1000 1000 1011 0100
Hexadecimal format: E1 88 B4