Data Storage Design in Java

Ming Jing

NIHDS

Fall 2020

Data & Index Files

- Data
  - Data File is over the Memory Capacity
  - **Partial** Data Loaded into the Memory
  - Load Data on **Need**
- Index
  - Index File is fitted in the Memory Capacity
  - **Entire** Index Loaded into the Memory
  - Load Index on **Initialize**

Sequential Access Text File (Read)

```java
class SequentialAccessFile {
  public static void main(String[] args) throws IOException {
    BufferedReader br = new BufferedReader(new FileReader("data.txt"));
    String s;
    while ((s = br.readLine()) != null) {
      System.out.println(s);
    }
  }
}
```

Sequential Access Text File (Write)

```java
class SequentialAccessFile {
  public static void main(String[] args) throws IOException {
    FileWriter fw;
    String str;

    // FileWriter(String fileName)
    // Constructs a FileWriter object given a file name.
    fw = new FileWriter("data.txt");
    str = "Hello World";
    fw.write(str, 0, str.length());
    fw.close();
    // data.txt
    // Hello World

    fw = new FileWriter("data.txt");
    str = "123";
    fw.write(str, 0, str.length());
    fw.close();
    // data.txt
    // 123

    // FileWriter(String fileName, boolean append)
    // Constructs a FileWriter object given a file name with a boolean indicating
    // whether or not to append the data written.
    fw = new FileWriter("data.txt", true);
    str = "Hello Java";
    fw.write(str, 0, str.length());
    fw.close();
    // data.txt
    // 123Hello Java
  }
}
```

Random Access File in Java

java.io.RandomAccessFile

RandomAccessFile is a Java class that allows the reading and writting of data to any location in the file respectively. It is used to read and write data simultaneously i.e. we can read the file, write to a file, and move the file pointer to another location in the same program.

File Pointer in RandomAccessFile Class

A file behaves like a large array of bytes, thus we use a cursor or index for the array called the **file pointer**. Data is read byte by byte starting at the file pointer and the write operation is performed starting at the file pointer too. File pointer can be moved or retrieved using **seek()** and **getFilePointer()** methods.

Constructors of RandomAccessFile Class

- RandomAccessFile(File fileObj, String mode)
- RandomAccessFile(String fileName, String mode)

Example

mode: "r", "rw"

Methods of RandomAccessFile in Java

- **seek(long pos)**: It sets the file pointer to the specified location.
- **getFilePointer()**: It returns the current offset or position of the File Pointer in the form of bytes from the beginning of the file.

Conclusion

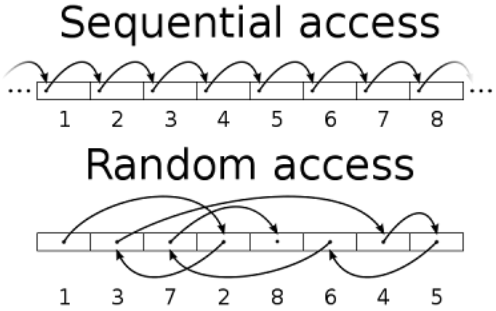

Figure: Sequential Access vs. Random Access

Data and Index File Design

Data

num ↹ name ↹ sex ↹ college ↹ phone␣␣␣...\n

Index (Key => Value)

num ↹ offset

DBMS Initialize

1. Create Data File Instance (STU_DB_FILE = "stu.db";)
2. Create Index File Instance (STU_INDEX_DB_FILE = "stu.index";)
3. Load Index into Map
4. Waiting for Actions (Insert, Delete, Update, Select)

Insert Action

Insert a new student (with num n') record into data file.

1. Check Num Conflict by Index Map. Abort Action if the same num n' found.
2. Find the biggest offset (o) in Index Map
3. Seek the o'(o' = o + 100) position in Data File
4. Write the student record into the Data File
5. Insert n' => o' into Index Map
6. Flush Index Map into Index File

Delete Action

Delete a student (with num n') record from data file.

1. Find the record offset o' by key n' in Index Map. Abort Action if num n' not exists;
2. Seek the o' position in Data File
3. Write a empty string (99 spaces) to overwrite the student record
4. Delete key n' from Index Map
5. Flush Index Map into Index File

Update Action

Update a student (with num n') record into data file.

1. Find the record offset o' by key n' in Index Map. Abort Action if num n' not exists;
2. Seek the o' position in Data File
3. Write the new student record to overwrite the old record

Select Action (Full Table Scan)

Find a student (known name n') record from data file

1. Sequential Access Data File line by line
2. Match each line by name n'

Select Action (By Index num)

Find a student (known num n') record from data file.

1. Find the record offset o' by key n' in Index Map. Abort Action if num n' not exists;
2. Seek the o' position in Data File
3. Read 100 bytes into String, then new Student(str)

Design Index name (not unique)

- How to design name index file
- Insert, Delete, Update, Select Methods ?