CSCI4707 Homework 3

Hao Wang 5086487

wang5167

B1.

a.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | F | G | H | I |

b.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | G | H | F | I |

c.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | H | G | I | F |

d.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | I | F | G | H |

B2.

Because database need to change some pages for it own. Also, when we want to write in another page, database should force a page.

B3.

1. We can use MRU policy. Since the entire file is sequentially scanned many times, it might cause sequential flooding. The MRU can avoid the sequential flooding.
2. We can use LRU policy. Since the popular items are requested very frequently and unpopular items are requested rarely, it’s quick and the cost is lower than others to get the popular items when we use LRU.

B4.

1. When the records we are looking for are in the same page.
2. It is impossible to have an Unclustered B+Tree index with Alternative 1. Alternative 1 implies clustered; in practice, clustered also implies Alternative 1 (since sorted file are rare).

C1.

a.

(1) No. A hash index matches terms that has a term attribute = value for every attribute in the search key of the index. For this problem, the search key is <ename,age>.

(2) No. A hash index matches terms that has a term attribute = value for every attribute in the search key of the index. For this problem, the search key is <ename,age>.

(3) Yes.

b.

(1) No. A tree index matches terms that involve only attributes in a prefix of search key.

(2) Yes.

C2.

a. Filescan.

b. Clustered B+Tree on <residence\_state,age>.

c. Unclustered B+Tree

d. Hash index on <resident\_state> and Clustered B+Tree on <residence\_state,age>

C3.

Clustered B+Tree on <residence\_state,age> and index to get average age.

C4.

R : 1000 records 1000/10 = 100 pages

S : 5000 records 5000/500 = 10 pages

According to the slides, the cost of probing inner index is about 1.2 I/O for hash index, 2~4 for B+ tree. Let’s assume the cost for B+ tree is 3 I/Os this time

a. 4100 I/Os 100 + 1000 \* (3 + 1)

b. 11010 I/Os 10 + 5000\*(1.2 +1)

c. 8100 I/Os 100 + 1000 \*((5000/1000) + 3)

d. 500010 I/Os 10 + 5000\*100