Chapter 12 Problems

(10) 1. Explain the difference between a primary index and a secondary index.

Answer  The clustering or primary index is on the field which specifies the sequential order of the file. There can be only one clustering index while there can be many secondary or non-clustering indices.

(30) 2. Construct a B+-tree for the following set of key values: (0,1,2,3,5,9,15,21,27,29,30). Assume that the tree is initial empty and values are added in ascending order. The number of pointers that fill one node is four. Then show the steps involved in (A) finding records with a search-key value of 9 and (B) records with a search-key value between 5 and 15, inclusive.

Answer

Some Rules:

1. Each nonleaf node has between $\lceil n/2 \rceil$ and $n$ children, where $n$ is fixed for a particular tree.

2. Each leaf can hold up to $n - 1$ values and the values may not overlap.

3. The root may hold fewer than $\lceil n/2 \rceil$ pointers, but may hold at least two pointers.

![B+-Tree](image)

Figure 1: B+-Tree

(A) Find records with a value of 9
i. Search the first level index; follow the first pointer.
ii. Search next level; follow the third pointer.
iii. Search leaf node; follow first pointer to records with key value 9.

(B) Find records with value between 5 and 15 (inclusive)

i. Search top index; follow first pointer.
ii. Search next level; follow third pointer.
iii. Search third level; follow pointer to records with key value 5,
and after accessing them, return to the leaf node.
iv. Follow the second pointer to the next key value 9.
v. Follow the fourth pointer to the next leaf block in the chain.
vi. Follow the first pointer to the key value 15, and access them.

(20) 3. Suppose there is a relation $R(X, Y, Z)$, with a B+-tree index with search key $(X, Y)$. What is the worst case cost of finding records satisfying $10 < X < 50$ using this index, in terms of the number of records retrieved $n_1$ and the height $h$ of the tree?

**Answer** $O(n_1 + h)$

(20) 4. Explain how you would reduce bucket overflow in hash file organization. What are the causes of overflow?

**Answer**

a. Our estimate of the number of records that the relation will have was too low, and hence the number of buckets allotted was not sufficient.

b. Skew in the distribution of records to buckets. This may happen either because there are many records with the same search key value, or because the hash function chosen did not have the desirable properties of uniformity and randomness.

To reduce the occurrence of overflows, we can:

a. Choose the hash function more carefully, and make better estimates of the relation size.

b. If the estimated size of the relation is $n_r$ and number of records per block is $f_r$, allocate $(n_r/f_r) \times (1 + d)$ buckets instead of $(n_r/f_r)$ buckets. Here $d$ is a fudge factor, typically around 0.2. Some space is wasted: About 20 percent of the space in the buckets will be empty. But the benefit is that some of the skew is handled and the probability of overflow is reduced.

(20) 5. For a system where a range of queries is likely, why or why not would you use a hash structure?

**Answer** A range query cannot be answered efficiently using a hash index, we will have to read all the buckets. This is because key values in the range do not occupy consecutive locations in the buckets, they are distributed uniformly and randomly throughout all the buckets.