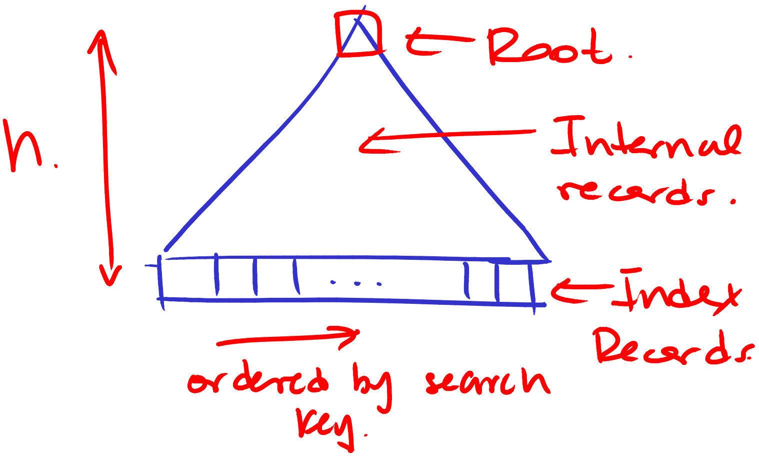
Indexes.

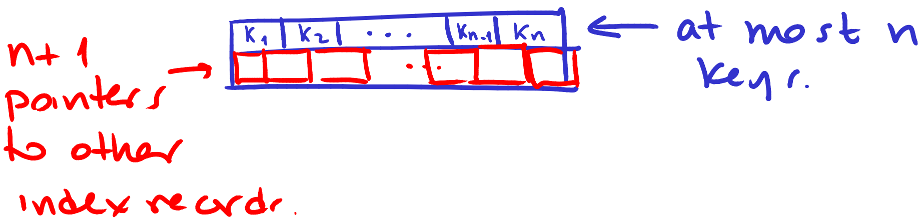
B+-tree

- Automatically Balanced
- Index records are at the leaves



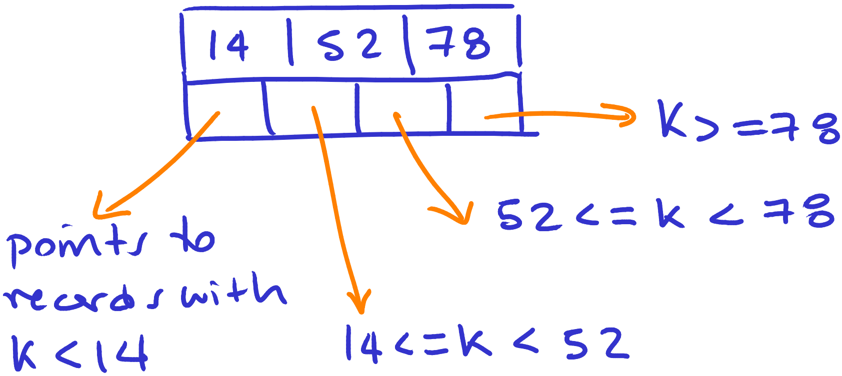
- Every record is a block.
- Index records form a list
  - They can be traversed in the order of the search key

Internal records

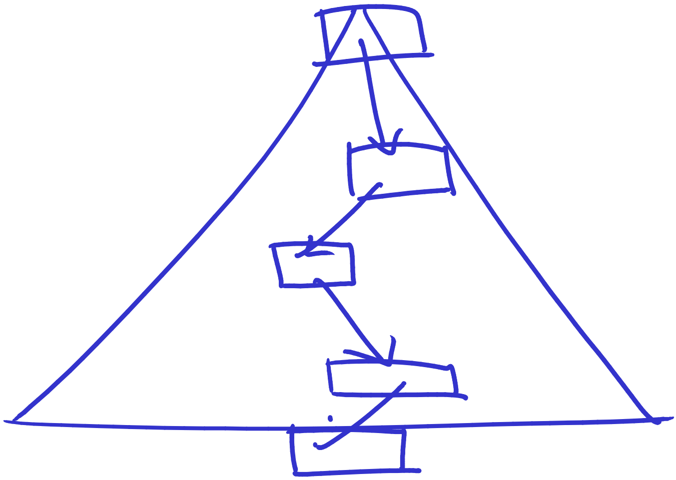


Example.

Assume $n=3$



Index is traversed from root to leaf



We assume root is always in memory hence, to reach the leaf we read h blocks.

Cost of index:

- Cost of reaching the leafs
- Cost of reading the matching records.

Example:

1) Assume $R(\underline{a}, b)$

$$\sigma_{a=5} R$$

Only one or zero matching tuples

$\Rightarrow$ We must traverse the index h .

The leaf either

- contains $a=5$ or not

Cost of index = h of index.

2) $$\sigma_{a>0} R$$

What if all tuples match?

- We traverse index (cost h)

Reads first leaf.

- We must read all leaves of index

Cost of index = h + # leaves - 1

To be able to compute the cost of an index we need:

Calculate # of leaf blocks of index proportional to # index records per block.

# of index records depends upon

a) Type of index
   Sparse vs. Dense

b) Number of tuples in Rel.

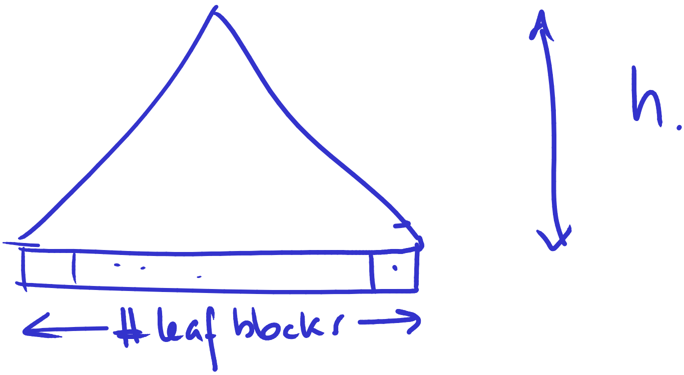
index records per block depends upon:

a) size of search key

b) occupancy rate
   => How much waste space is there in the index (to keep it balanced).

We assume that occupancy rate of inner nodes and leaves is the same as internal nodes.

Hence, height of tree depends upon # of leaf records.



n = max number of keys per record.

fill = occupancy rate (between 0–1, but usually around ½ to ¾)

$$\#\text{leaf blocks} = \left\lceil \frac{\#\text{ index records}}{n \cdot \text{fill}} \right\rceil$$

For h, we simplify calculations:

$$h = \left\lceil \log_{n \cdot \text{fill}} (\#\text{index records}) \right\rceil$$

Example:

Assume $n = 150$, $fill = \frac{2}{3}$

How many index records can we store in an index of height 1, 2, 3, 4, 5, 6.

$$h = \left\lceil \log_{n \cdot fill} \#indexRecords \right\rceil$$

Let us worry about max #indexRecords

$$\Rightarrow \quad h \simeq \log_{n \cdot fill}(\#indexRecords)$$

$$n \cdot fill^h \approx \#indexRecords$$

h	#indexRecords	
1	100	$= 10^2$
2	100^2	$= 10^4$
3	100^3	$= 10^6$
4	100^4	$= 10^8$
5	100^5	$= 10^{10}$

With 5 block reads we can find a leaf with a given search key in an index of 10 giga-records!!

How many search keys do we need to store?

- Sparse index: $B(R)$
- Dense index: $|R|$

Sparse index is marginally shorter than dense index.