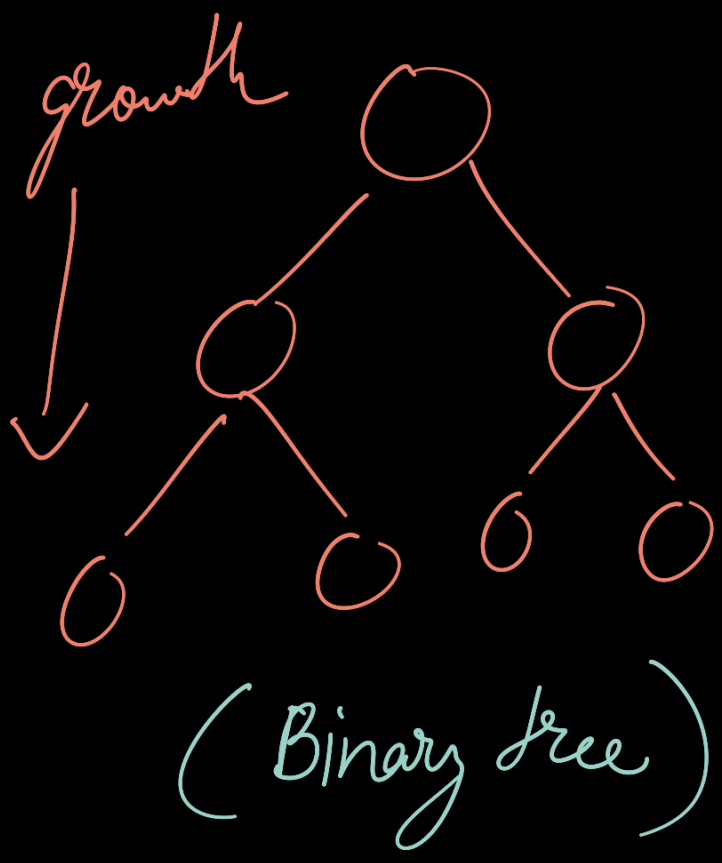
* B, B^+ tree grow horizontally not vertically



No. of levels is less, so less number of HDD blocks are accessed

Let's recap before understanding B, B^+ index file

Assumptions :

① 1 table = 1 file

DBMS can store more than 1 table records in 1 file for faster disk access time

② Time to read 1 block of HDD = 15 ms

③ Size of 1 block/sector = 512 Bytes of HDD

↳ We are considering Mechanical HDD, SSD are 10x faster

Consider this table

Employee

Field	Size (bytes)
Eid	10
Name	50
Dept	10
section	8
Add	5

Total size of 1 record = 128 bytes

(Employee table)

Eid	Name
1	
2	
3	
4	
:	:

1 block of HOD

Consider 100 records

* No. of records / HDD block = $\frac{512}{128}$

$= 4$

* 100 records $= 4 * 100$ HDD blocks

$= 400$ block of HDD

Note → How this employee table file is allocated HDD blocks is controlled by underlying OS file system

↳ In case of Linux → i-node

* Consider query :

Select * from table where Gid == 50

↳ Worst case → Entire 400 blocks has to be read and records has to be compared

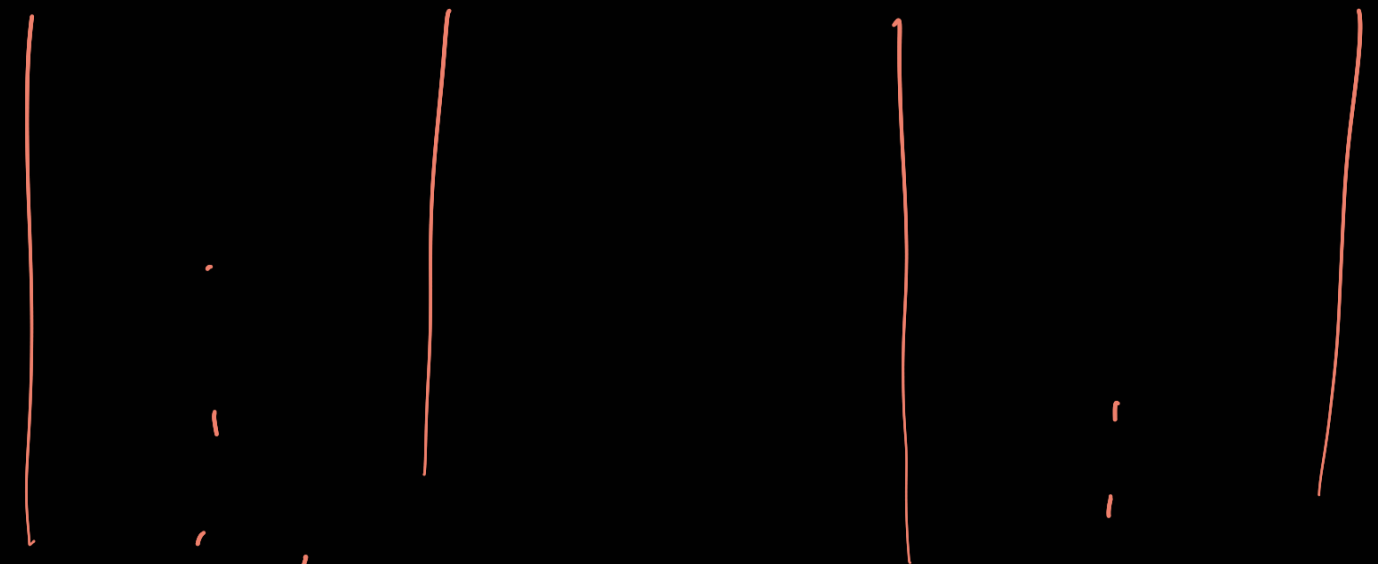
= 400 * 15 ms = 6 seconds

↓

Too slow

optimise

Index on Eid



100 entries in index file

100 records

↳ Consider 1 index file entry size

= Eid + record ptr

= 10 + 6

= 16 bytes

Total entries of index / block = $\frac{512}{16} = 32$ entries

Total 1800 blocks for 100 = $\frac{100}{32} \approx \lceil 3.2 \rceil$

= 4 blocks

Query → 4 block of index + 1 block for actual record

= 5 1100 block access

= 5 * 15 ms

= 75 ms

6 second ⟶ 75 ms

* If records increase to 100 ⟶ 1000 records.

So, 1000 = 1000 index reads

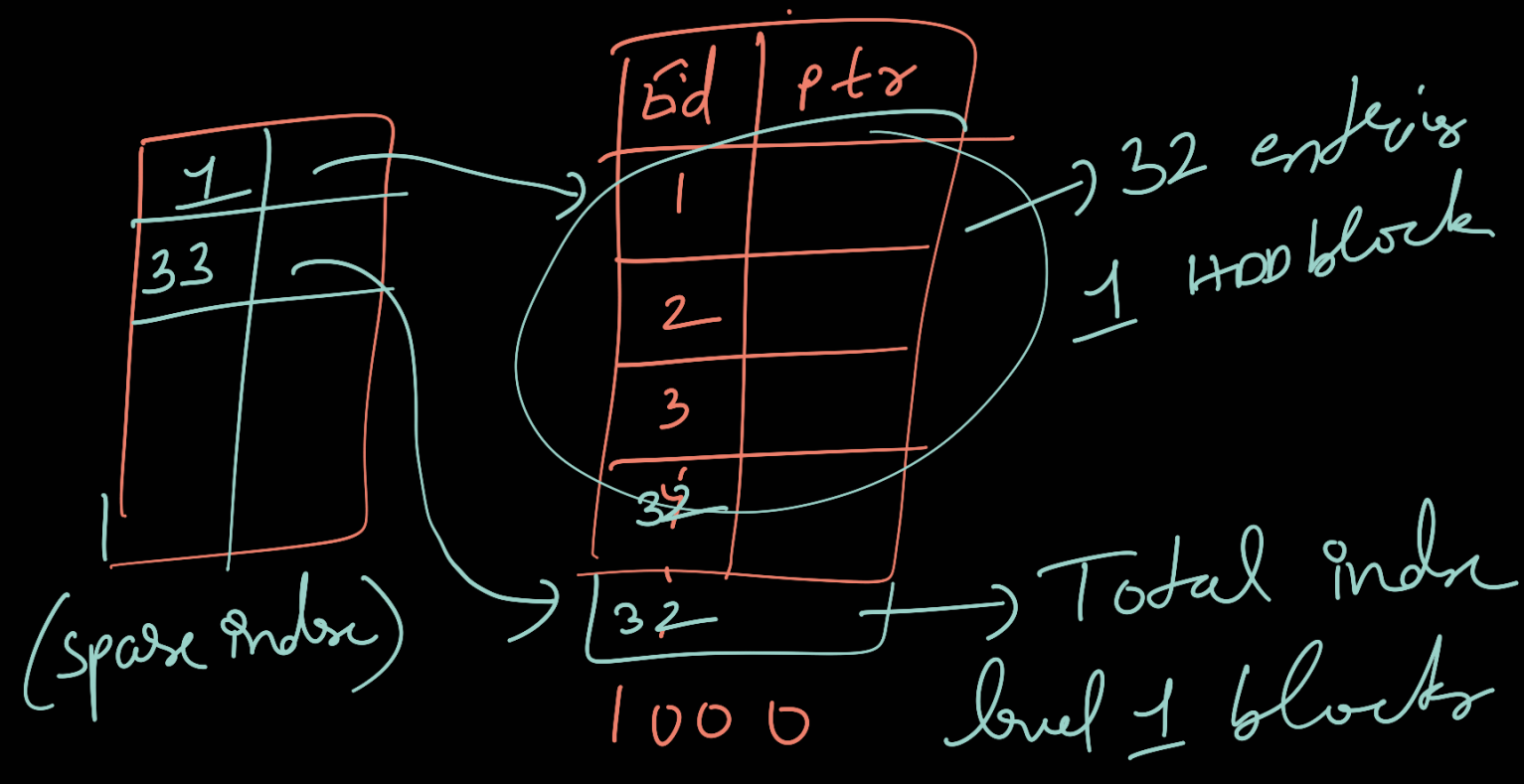
∴ 40 blocks for index file

= 40 + 1

= 41 HDD block * 15 ms accers = 600ms

↓ optimize this

Multi-level Index



$= 40$

$\therefore$ 40 entries in 2nd level

= Consider entries size = 16 bytes

Total 2nd level entries / block $= \frac{512}{16}$

$= 32$

Total blocks for 40 entries $= \frac{40}{32}$

≈ 2 blocks

Access time $= 2 + 1 + 1$

(2nd level) (IST level) (actual record)

$= 4 * 15$

$= 60\,ms$

* $600\,ms \xrightarrow[\text{faster}]{10\times} 60\,ms$

$\Downarrow$

In real world, records are inserted and deleted,

So, we need self managing multi-levels, also we need to optimize no. of levels, it should be less