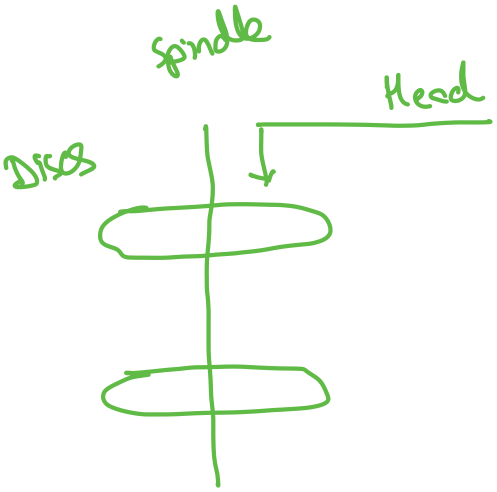
RDBMS-3

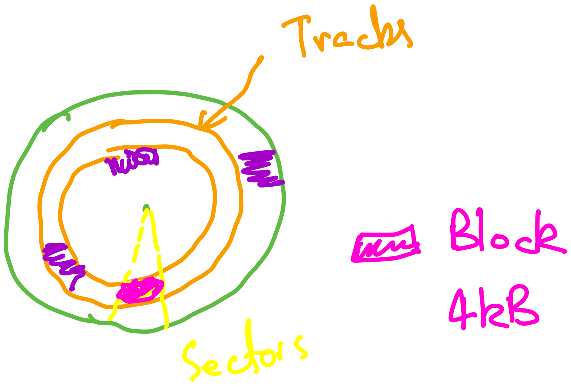
Indexing + ACID

DCC → ACID

RAM → volatile /non persist

HDD/SSD → non volatile /persistent





dB

sid sAddress sCity sSubject 1kB

⋮

10,000 rows

dB size = 10,000 kB ~ 10MB

Each block → 4kB

Total = 10,000 kB

⇒ No of Blocks = 2500 blocks → Spread Accross

Q → Give me all data for student Id = 120.

sid → Block X → 120 → Block X

Indexing

I Simplest

16 bytes 16 byte

sid	BlockAdd
1	B_A
2	B_B
120	B_I
10,000	B_X

→ 32 bytes

keep on DISK

Each block → 4kB

Each row = 32

Each Block = $\frac{4000}{32}$ rows

→ =~128

RAM size = 8kB

Index table = $\frac{10000}{128}$

<sid=120>

= 78 blocks

Total Disk blocks reads = 78 + 1 → to get data from block

↓ to find correct row in Index

L = 0
R = 10,000 → $M = \frac{L+R}{2}$

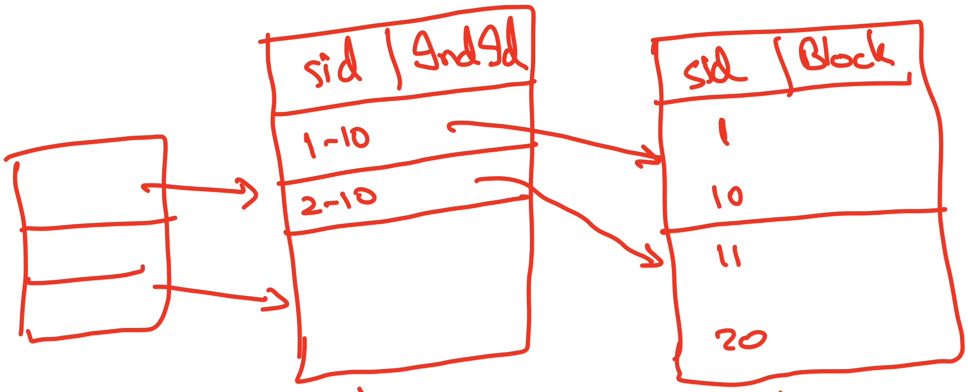
Find Index 120 ↓

120 == M ? → ?

L = 0
R = 10,000 → M = 500

If 120 sid is in whichever block 500 is in?

Multi level Index



$\frac{28}{10}$ → 8 blocks

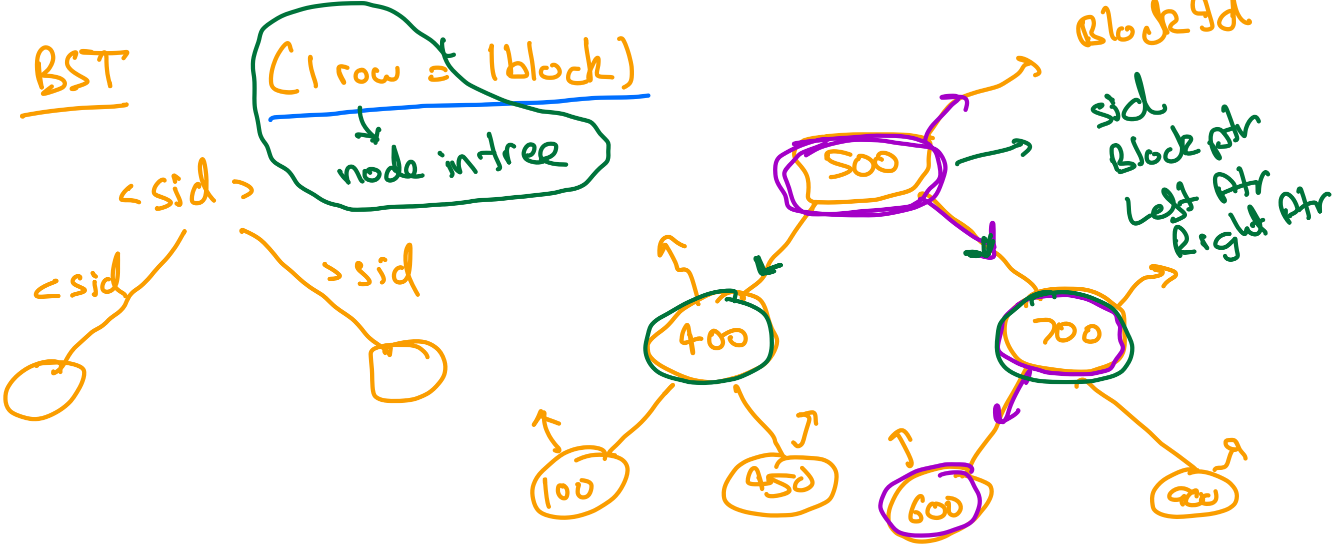
21 : 78 block

$\log_{10}(78)$

blocks overhead

→ Update or add index

II BST



Total n nodes in tree

↳ log(n)

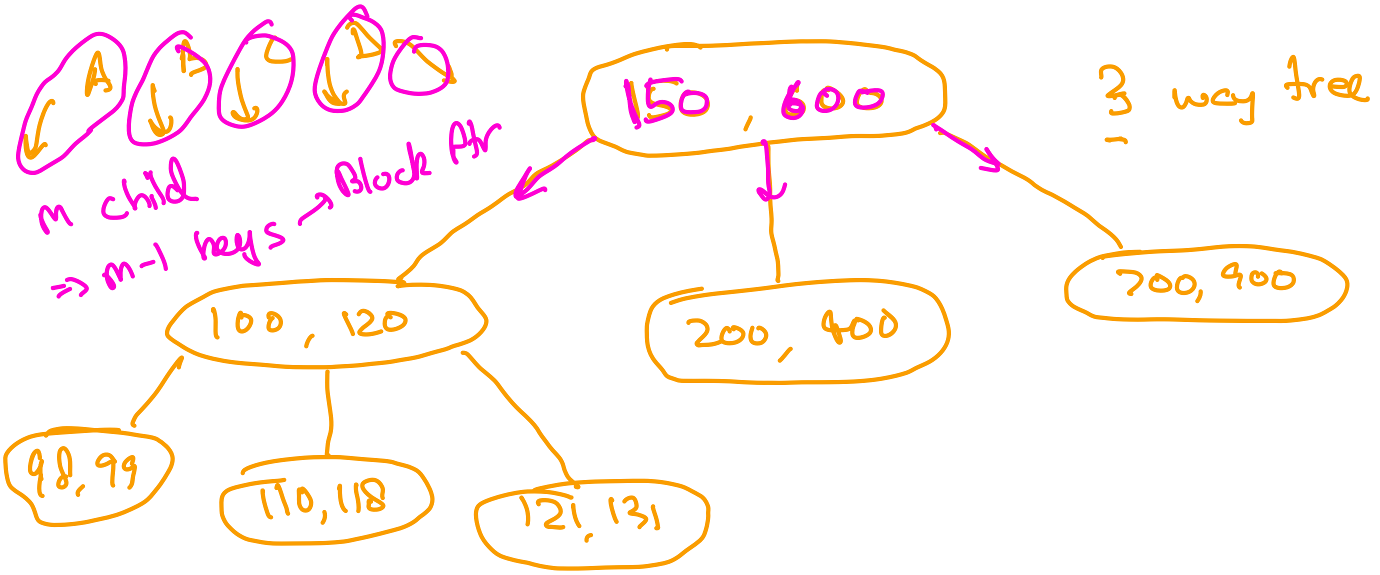
↳ number of rows = number of blocks

Problems

1. In each node, storing very less. => Way more blocks

III M-way Tree → Skewed is possible!

→ BST with multiples key & multiple children



Each node will have → m − 1 keys ⇒ (m − 1) Block ptr

m child pointers

Size of each node ≤ Block size

10,000 rows → 1 row per node = 10k nodes

→ 10 row ... = 1k nodes

5 rows = 200 nodes

Size of each node = (m − 1) x Size of key

+ (m − 1) x Size of Block Pointer

+ m x Size of Child Ptr

≤ Size of Block

$$(m-1) \times 16 + (m-1) \times 16 + m \times 16 \le 4kB$$

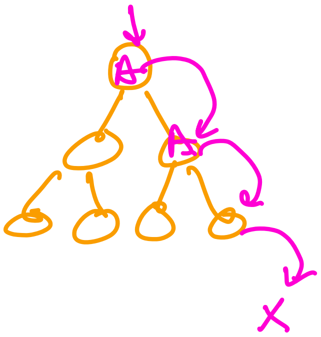
↓

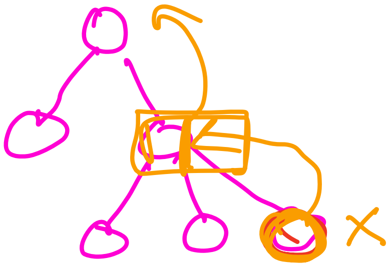
$$m \le \underline{84}$$

B Tree

→ self balancing M way tree

1. Each node must contain at most m children
2. Every non leaf & non root node must have at least m/2 children
3. Root must have at least 2 children
4. Trees are created in bottom up

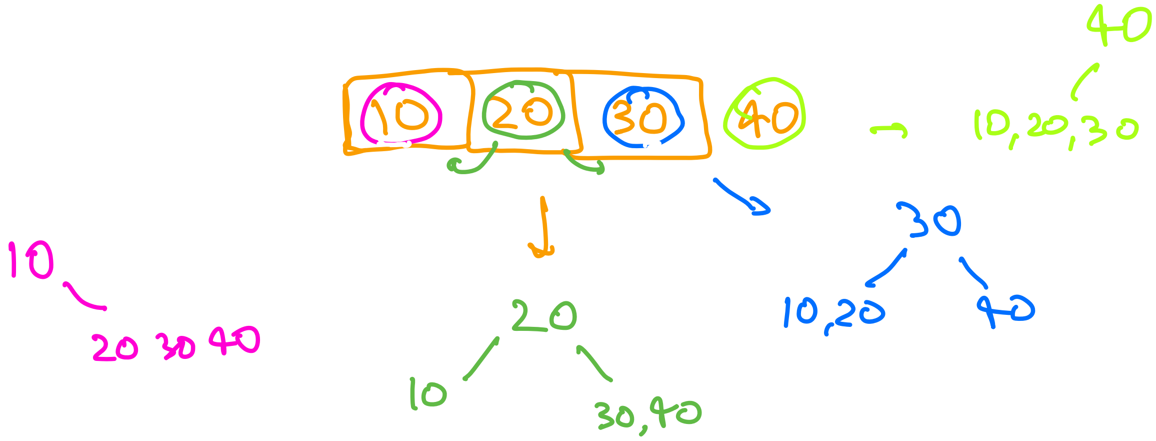


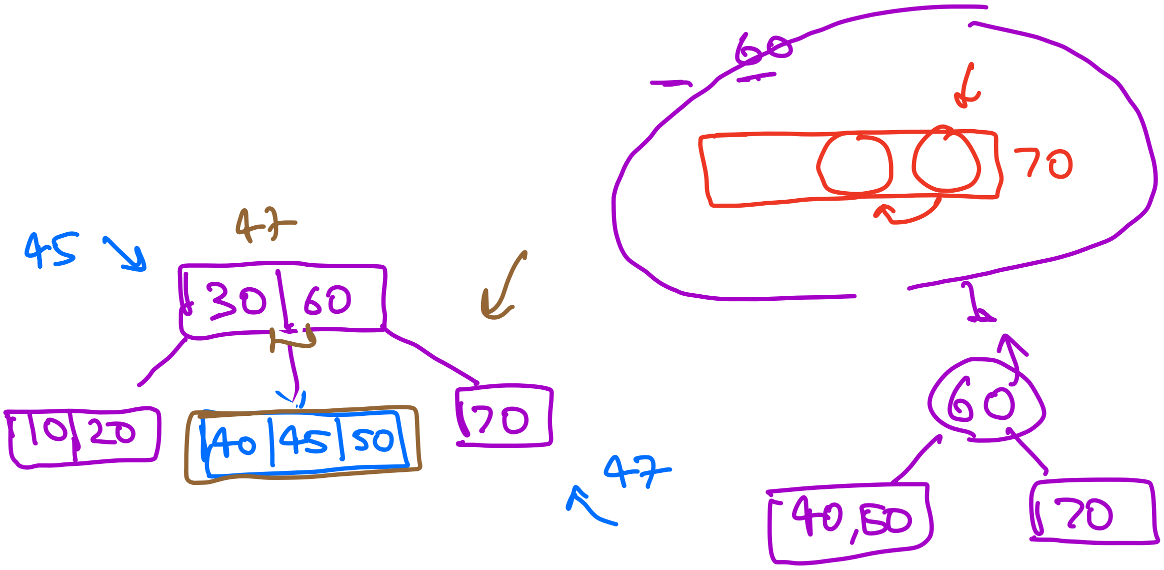


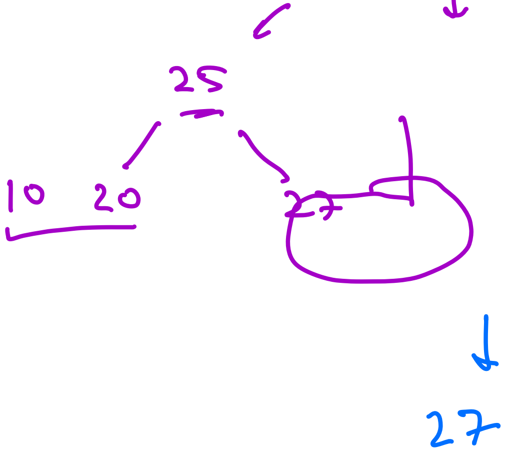
Insertion

10, 20, 30, 40, 50, 60, 70, 80

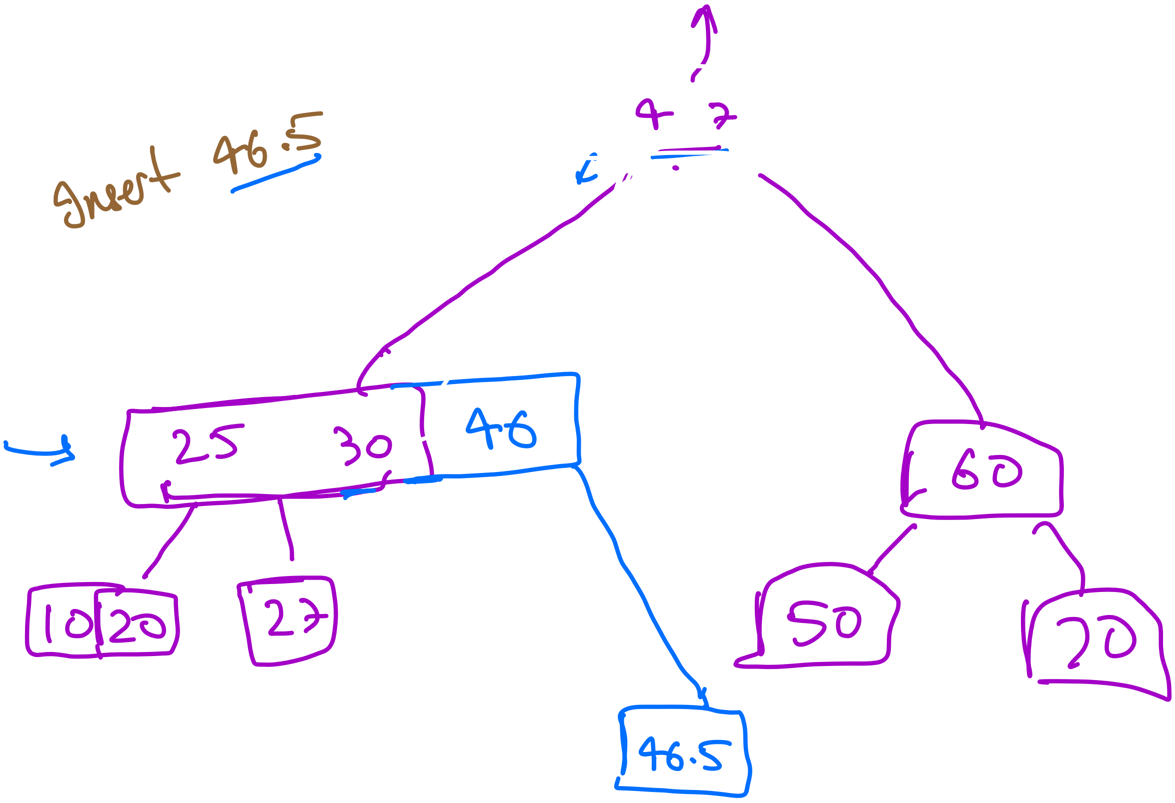
4 way ⇒ 3 keys







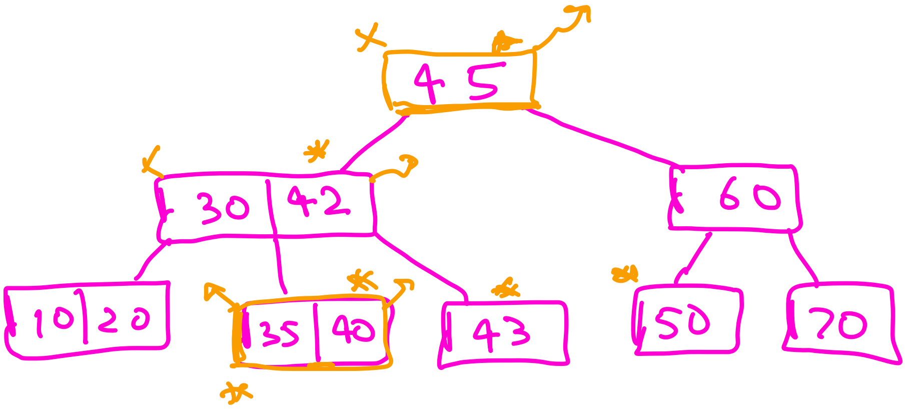
Insert 46.5



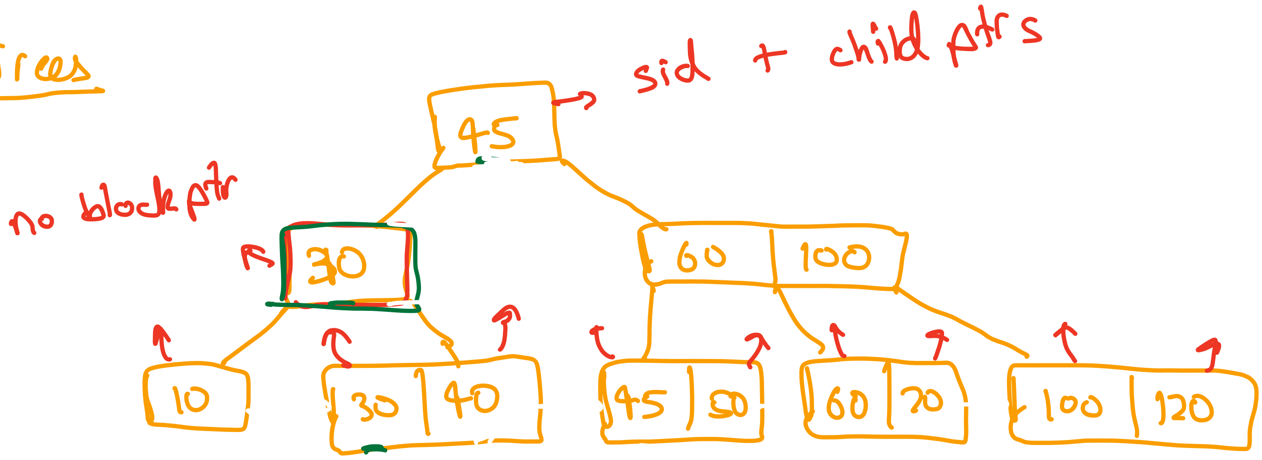
Problem

Range Queries → Give me all indexes → 35 – 50

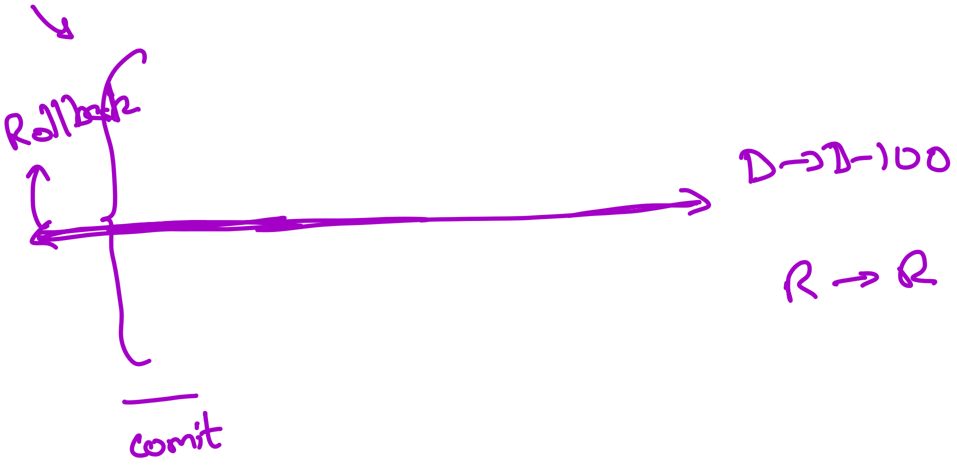
Blocks reads = (R - L) × log(n)



B + Trees



→ either entire T completes properly or it has no effect



Consistency

Balance ≥ 0 → Data Integrity

either execution will follow consistency or fail

D → 200 $\qquad$ D = D-300

E → D = D-100 ✓ $\qquad$ fail

Isolation

→ guarantees that multiples atomic executions running at the same time will not impact each other

D = 500 → 400

R = 1000 → 1100 → 900

A = 1000 → 1100

$D \xrightarrow{100} R$, $R \xrightarrow{100}$ Agcy

✓ get (D) → D' = 500

✓ get (R) → R' = 1000

✓ update D = D'-100

✓ update R = R'+100 → 1100

✓ get (R) → R'' = 1000

✓ get (A) → A' = 1000

✓ update R = R''-100 →

update A = A'+100 →

Durability

↳ Power loss / System Crash/ malfunction etc → data will persist