Block Size is 1024, BP is 8 bytes, RP: 9 bytes, search key 10 bytes.

Order p: max possible pointers which can be stored in $B^+$ tree node.

What is the best possible order of $B^+$ tree?

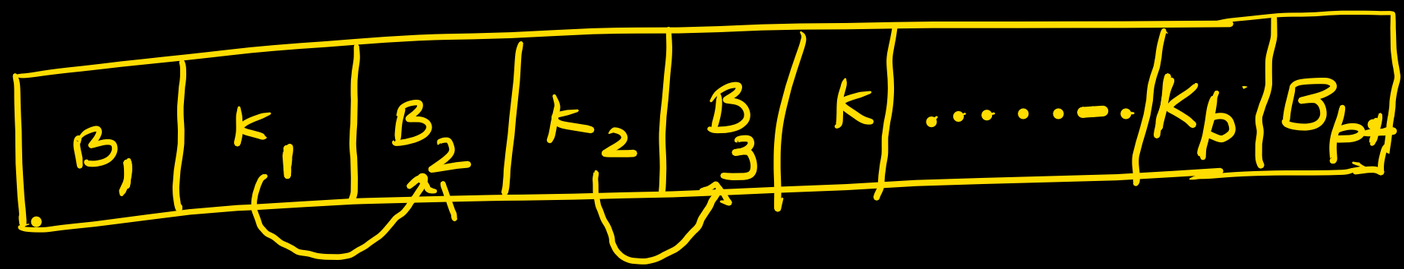
(i) Internal node (ii) leaf node

⇒ Block size is 512B, K = 5 bytes, BP = RP = 10B.

Order P: max possible keys which can be stored in a $B^+$ tree node.

what is the best possible order of $B^+$ tree node.

internal node structure :




Key $K_1$ $K_2$ ..... K (P) → max

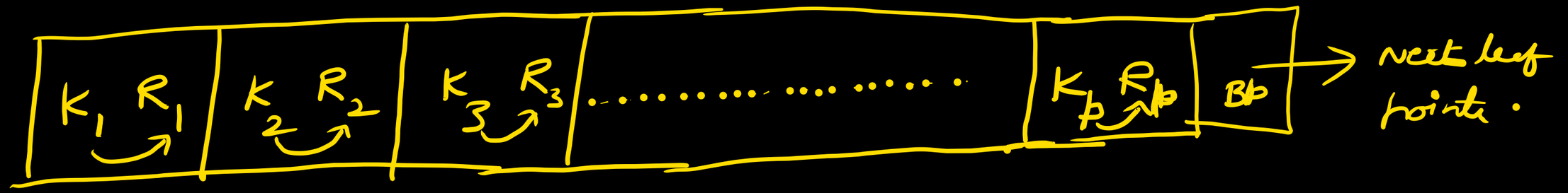
$(p+1)BP + p(K) \leq$ Block.

$(p+1)10 + p(5) \leq 512$

$\Rightarrow 15p \leq 502$

$\Rightarrow p \leq \left\lfloor \frac{502}{15} \right\rfloor = 33$

leaf:




(P) → total no of ~~pointers~~ keys.

$p(K+R) + Bp \leq$ Block size.

$p = 33$ ✓

for this case, order is same for ~~block~~ leaf and internal node because $Bp = RP$ size.

B: 1024

K: 10B, $R_p$ = 9B, BP = 8 bytes.

B tree node

order p: 38

max bps

B+ tree node

Internal node order = 57

leaf node order = 54

→ $B^+$ tree node can accomodate more no of keys than Btree.

→ If block size of B tree node is same as BS of $B^+$ tree node, then order 'p' of B tree node < order p of $B^+$ tree node

→ The number of leaves in $B^+$ tree is less than that of nodes in Btree. ∴ $B^+$ tree I/O cost is less than B tree I/O cost.

→ ∴ $B^+$ tree is preferred for DB indices

→ How many min/max keys and ~~nodes~~ nodes in B/B+ tree with order P: 5 & levels 4?

order p: b/w (2) to P bp's for root node

b/w $\lceil p/2 \rceil$ to p bps in other nodes

b/w $\lceil p/2 \rceil - 1$ to p-1 keys in leaf node.

$\lceil \frac{5}{2} \rceil = 3$

Root: 2 → (P) (5)

min b/p — max b/s.

Leave: $\lceil p/2 \rceil - 1$ — p-1

min keys — max keys

other: $\lceil p/2 \rceil$

min bps

(5)
(p)

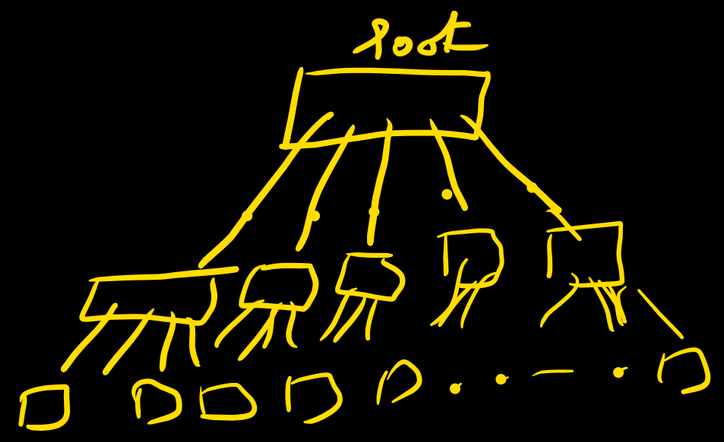
max bps

| level | max nodes | max BPs | max key |
|---|---|---|---|
| 1 | 1 | 5 | 4 |
| 2 | 5 | 5×5 | 5×4 |
| 3 | 25 | 25×5 | 25×4 |
| 4 | 125 | | 125×4 |
| | 156 | | |

→ *actual no keys
125×4 = 500

624

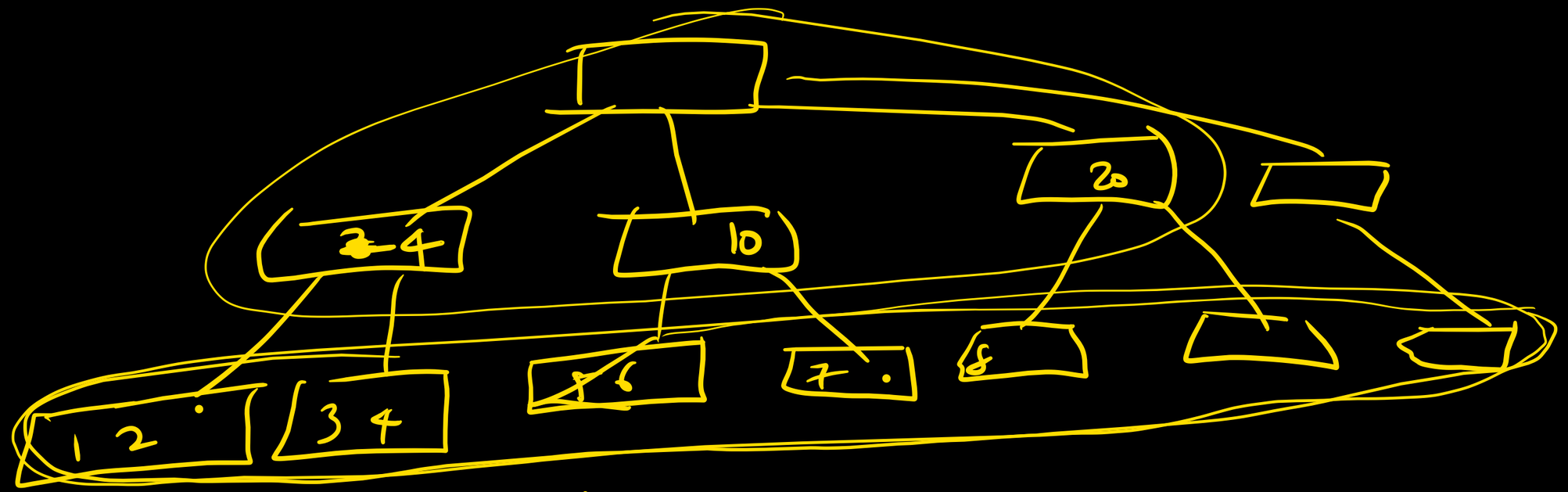
no need to
Compute this
as question is
about 4 level only



max nodes in B/B+ trees = 156 in this question.

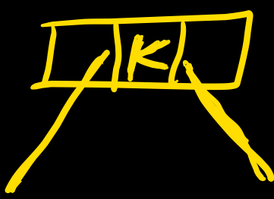
max no of key in B tree = 624

In B+ trees. key = 500




all key are present in leaves

| level | min nodes | min BPs | min keys |
|---|---|---|---|
| 1 | 1 | 2 | 1 |
| 2 | 2 | 2 x 3 | 2 x 2 |
| 3 | 6 | 6 x 3 | 6 x 2 |
| 4 | 18 | — | 18 x 2 |

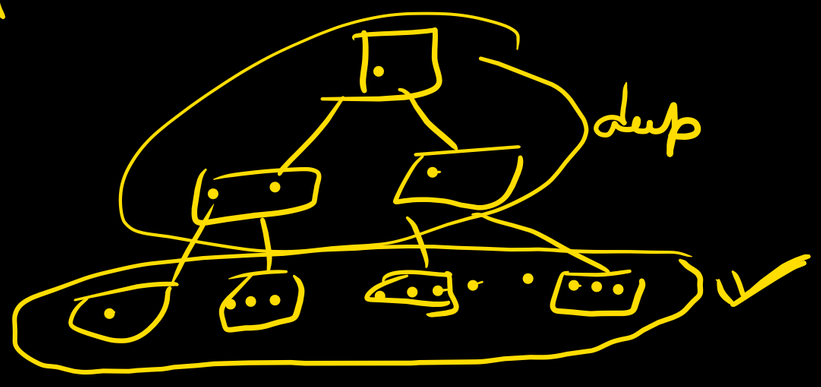



∴ min nodes in B/B+ trees in this example $S = 1+2+6+18 = 27$

no of keys in B tree = 53

no of key in B+ tree = 36.

∴



→ How many min/max keys & nodes in B Tree / B+ tree with order

p: 5 & level 4:

Assume order p: between 1 to 2p keys for root

b/w p to 2p keys for other nodes

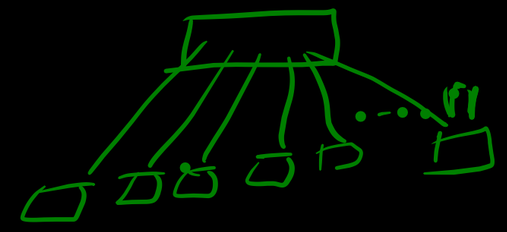
1, 2, 3, ... 10

| | min | max | BP min | BP max |
|---|---|---|---|---|
| Root order p: | 1 key | 10 keys | 2 | 11 |
| other nodes order p: | 5 key | 10 keys | 6 | 11 |

Order is never a fixed thing.

It changes from question to question.

| level | max nodes | max BPS | max key |
|---|---|---|---|
| 1 | 1 | 11 | 10 |
| 2 | 11 | 11 X 11 | 11 X 10 |
| 3 | 11 X 11 | 11 X 11 X 11 | 11 X 11 X 10 |
| 4 | 1331 | — | 1331 X 10 |



max no of nodes in $B/B^+$ tree = 1464

max no of keys in B tree = 14640

max no of keys in $B^+$ tree = 1331 X 10.

| level | min nodes | min BPS | min keys |
|---|---|---|---|
| 1 | 1 | 2 | 1 |
| 2 | 2 | 2×6 | 2×5 |
| 3 | 12 | 12×6 | 12×5 |
| 4 | 72 | — | 72×5 |

∴ min no of B/B+ tree nodes = 87 (1+2+12+72)

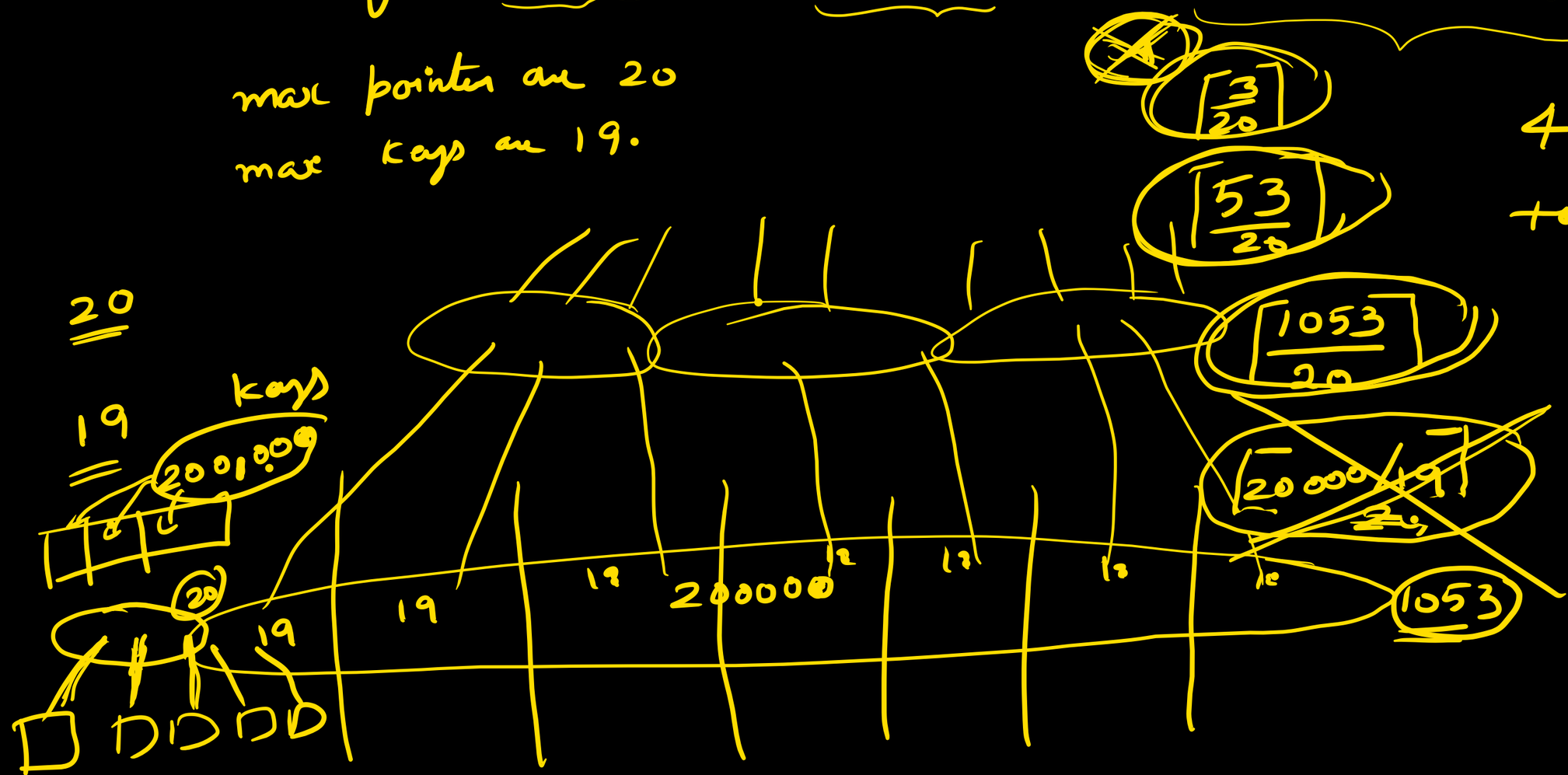
min key of B tree = 431

min keys in B+ tree = 72×5 = 360.

→ No of keys used for $B^+$ index : 20000 . Order of p: 20 (max pointers per node)

How many min levels / min nodes of $B^+$ tree are required?

max pointers are 20
max keys are 19.




4 levels

total = 1053 + 53 + 3 + 1.
= 1110

Gate DA is over:

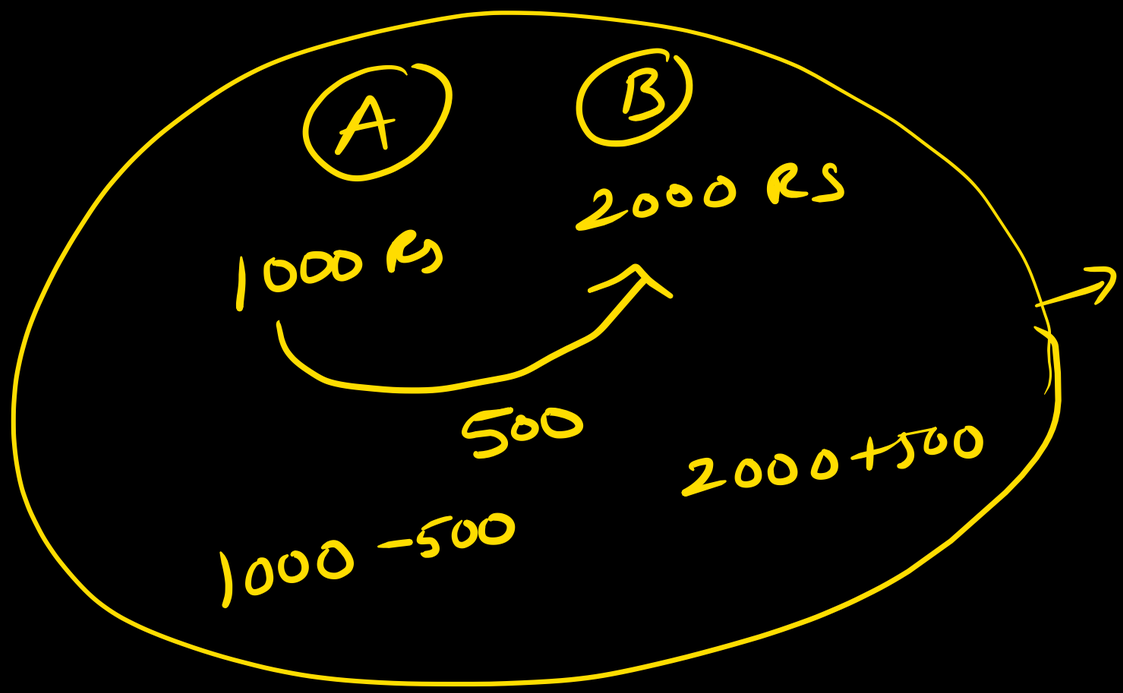
DW → start soon:

Transaction and Concurrency control:

5 min Break

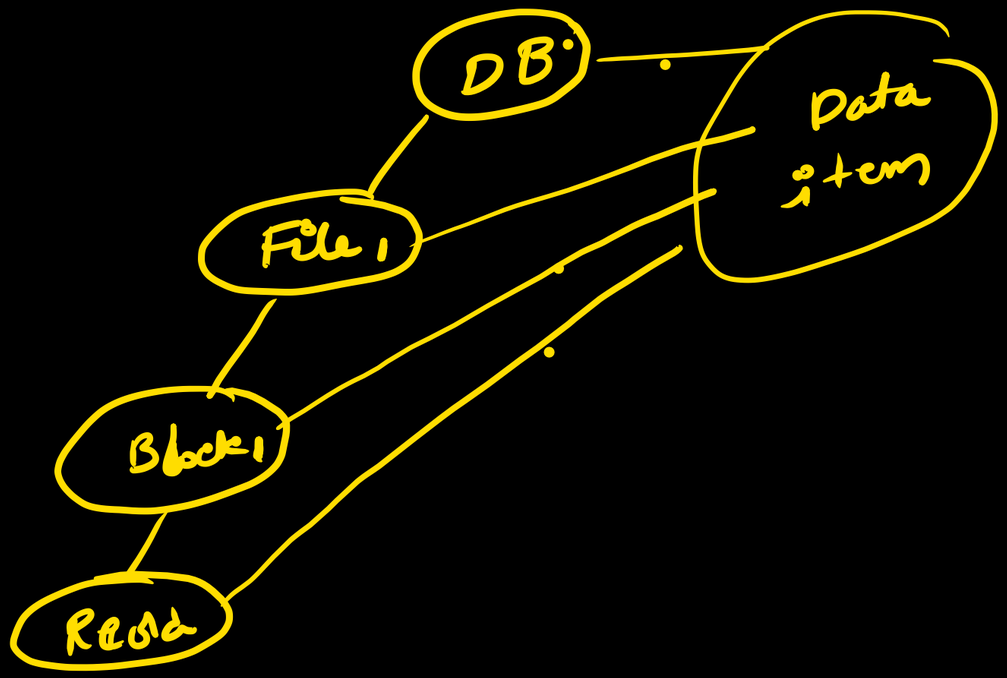
## Transaction and Concurrency Control:

**Transaction:** Set of logically related operations to perform a unit of work.



Data item:-

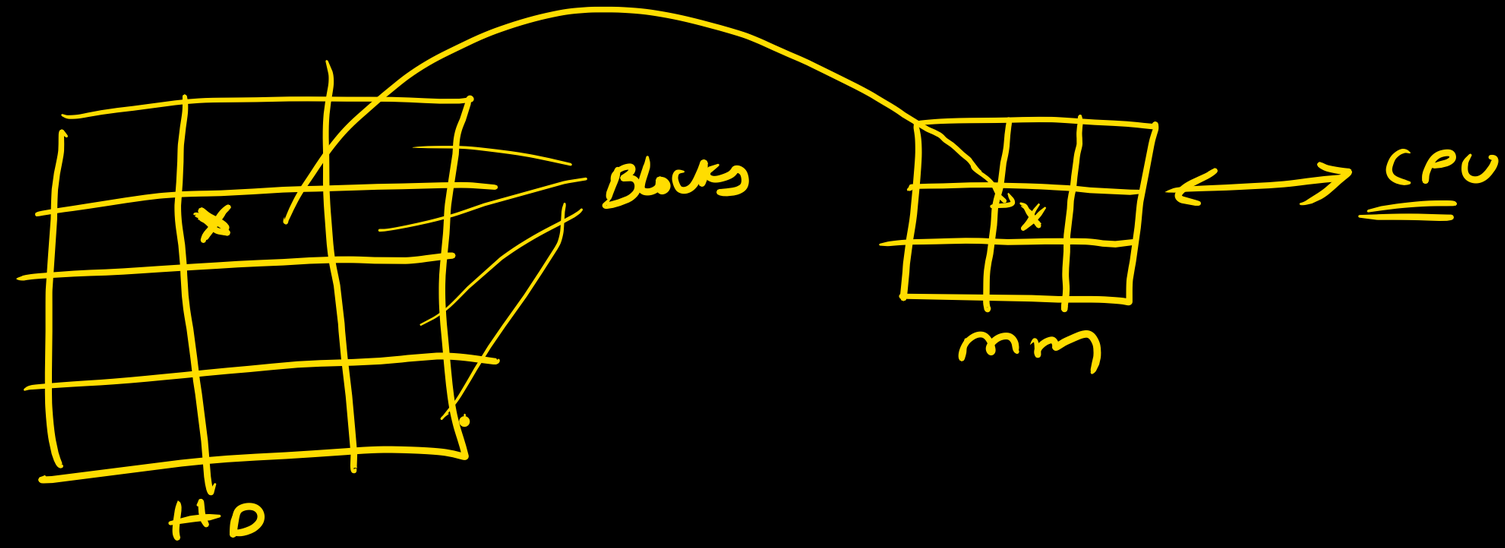
DB element which is needed to be ~~accessed~~ accessed by ~~many~~ one or many transactions.



## main operations in transaction:

1) Read (X)  X: data item

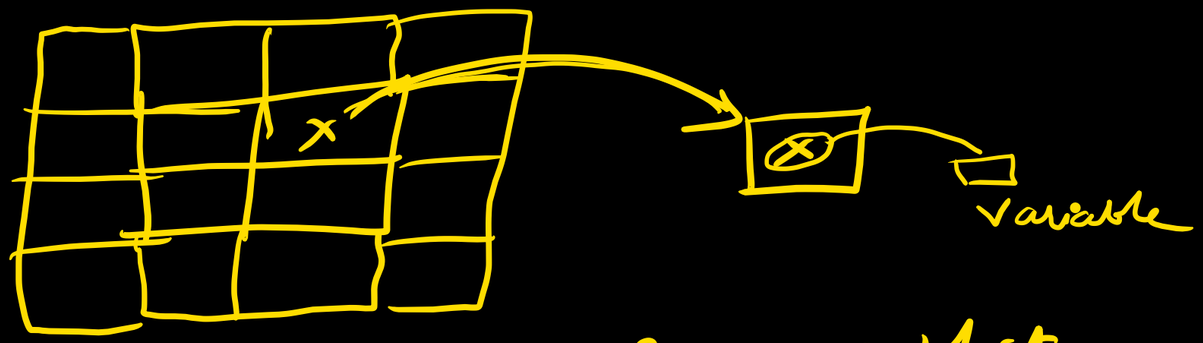
Access data item X from DB file (Disk) to main memory to use value of X in transaction logic.



Read x :

1) Find block from DB file which consists of data item X.

2) Transfer entire block from HD to mm




3) Find the add of x in mm block
and copy value of 'x' in programmed variable variable.

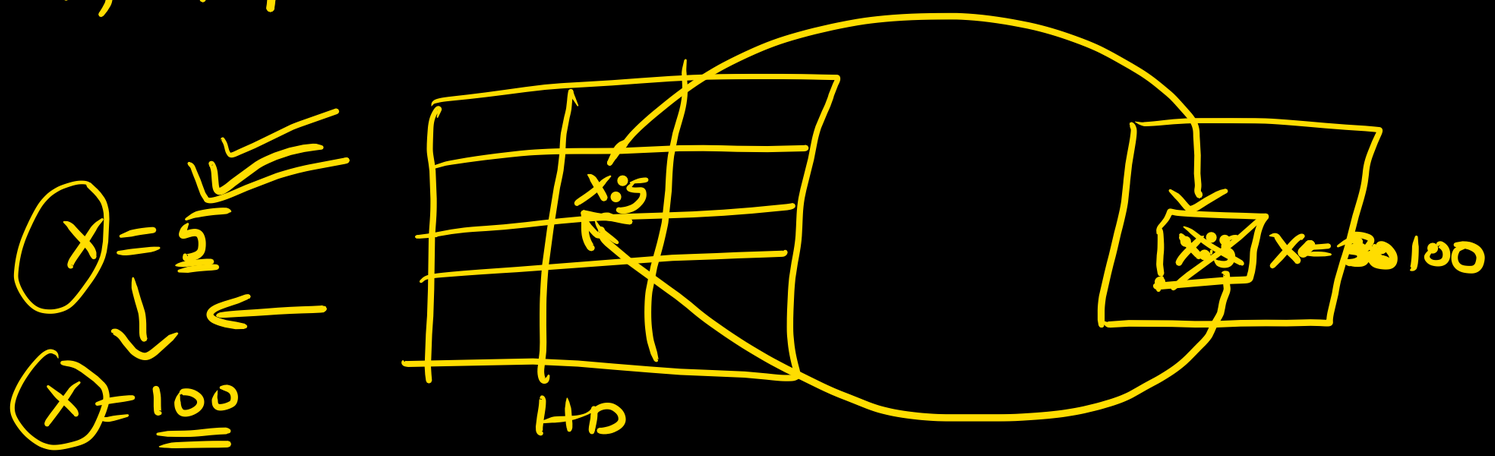
1, 2, 3 → atomic. ——— all or nade

123, ÷, ~~D~~

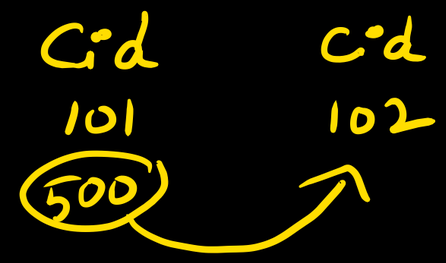
Write(X):

Updation of data item in DB file (Disk)

1) Find block from DB file which consists of data item 'X'.
2) Transfer block to mm
3) Find add of 'X' in mm and update X in mm.
4) Replace updated block in the disk.



Account (cid, bal)




Transaction:

```
begin transaction
    update  Account  Set  bal = bal - 500  where cid = 101;
    update  Account  Set  bal = bal + 500  where cid = 102;
end transaction
```

high level:

low level of transaction

Begin transaction $T_1$

✓ Read (bal of cid : 101)

bal = bal - 500;

✓ write (bal);

✓ Read (bal of cid : 102)

bal = bal + 500;

✓ write bal

✓ Commit

end transaction

R, W, Commit