



# LTeam Milestone 2

Alejandro Torres, Jenny Wang,  
Ho-Chih Ma, Jamie Wu,  
Karthik Palanisamy



# Team Member Roles

## Leadership Roles:

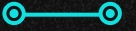
- Team Coordinators: Jenny, Alejandro
- System Architects: Everyone
- Developers: Everyone
- Testers: Everyone

## Implementation and Design Areas:

- Durability & Bufferpool Extension: Alejandro, Jamie
- Data Reorg [Contention-free Merge]: Jenny
- Indexing: Ho-Chih, Karthik



- [1] Durability and Bufferpool Extension**
- [2] Data Reorg**
- [3] Indexing / BTree Extension**
- [4] Performance**
- [5] Live Demo and Q&A**

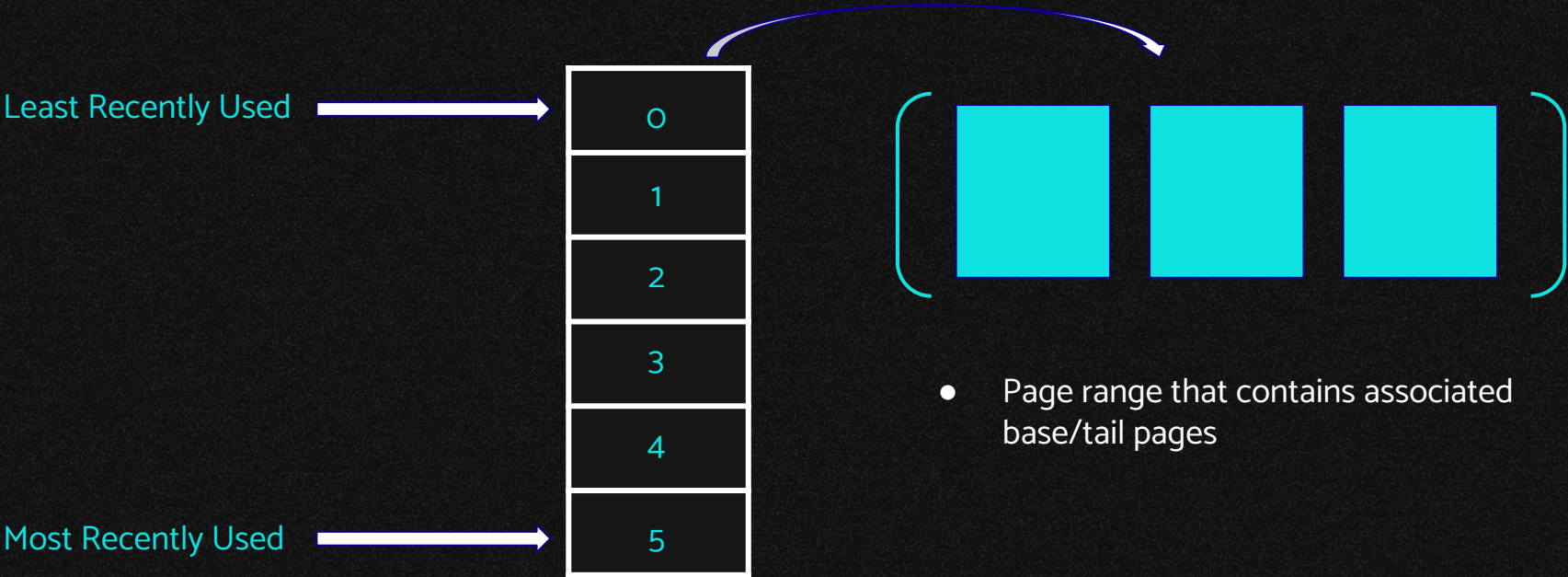


# [1] Durability and Bufferpool Extension

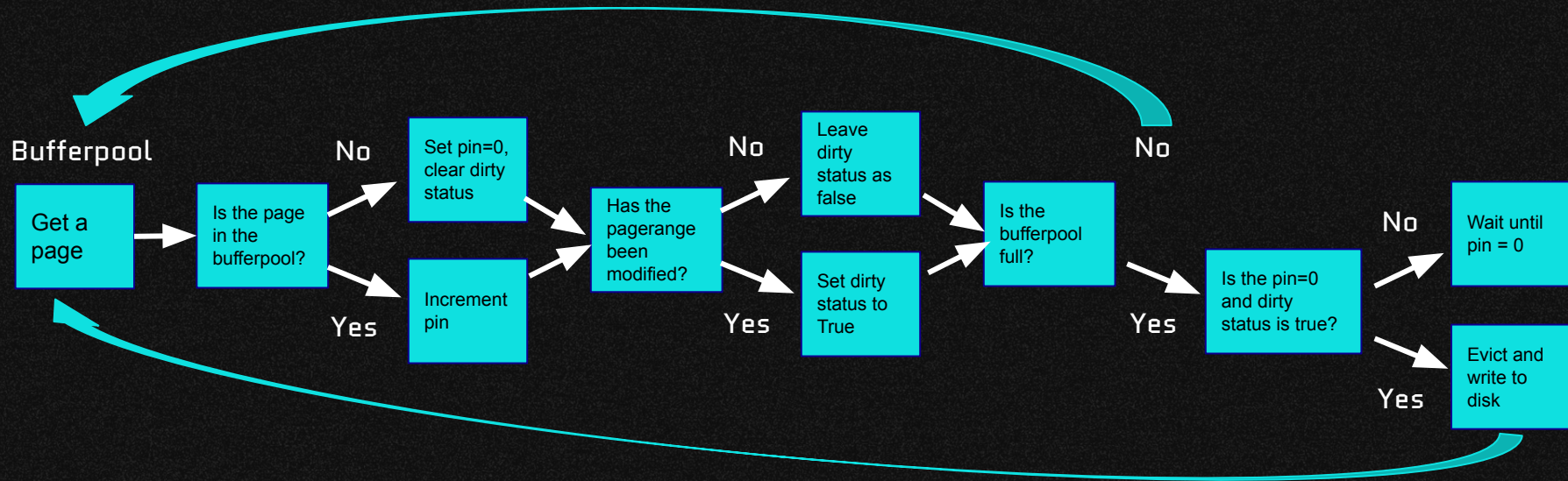




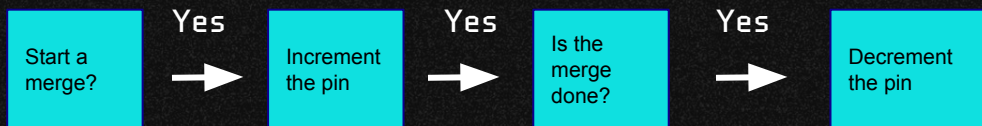
# Bufferpool : LRU Cache



# Pinning & Dirty Status



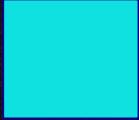
## Merge





# Heapfile : Write to disk

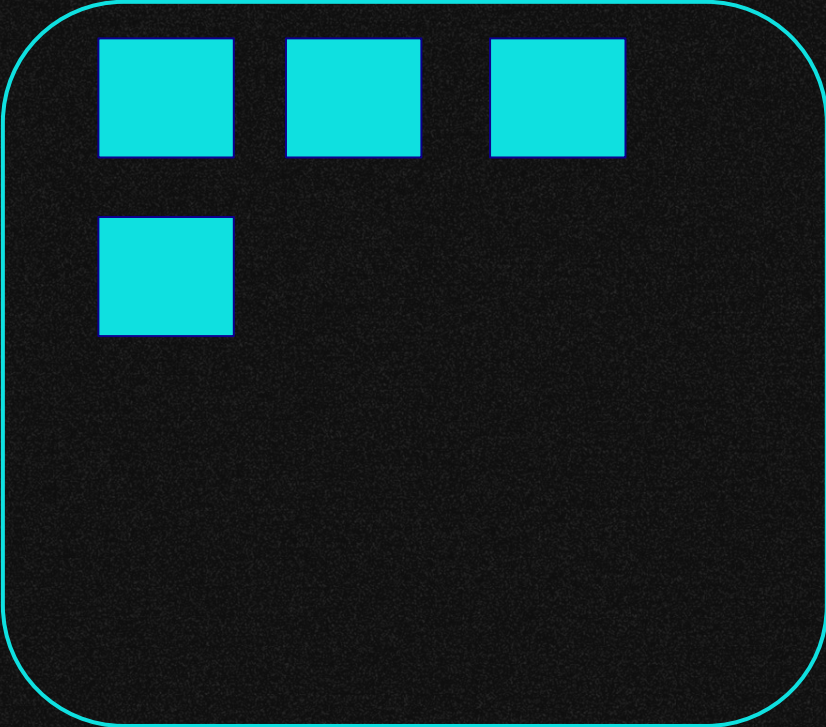
Insert into bufferpool



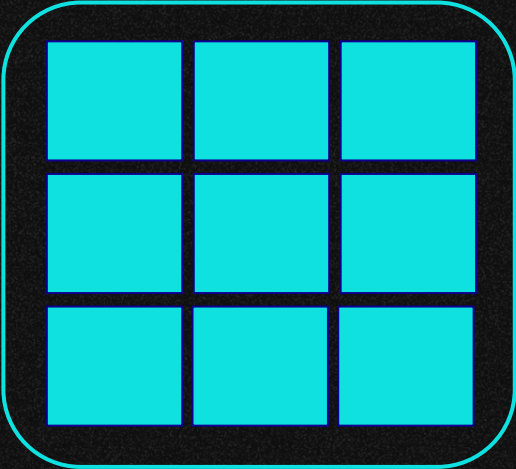
Write to disk



Disk



Buffer is at capacity



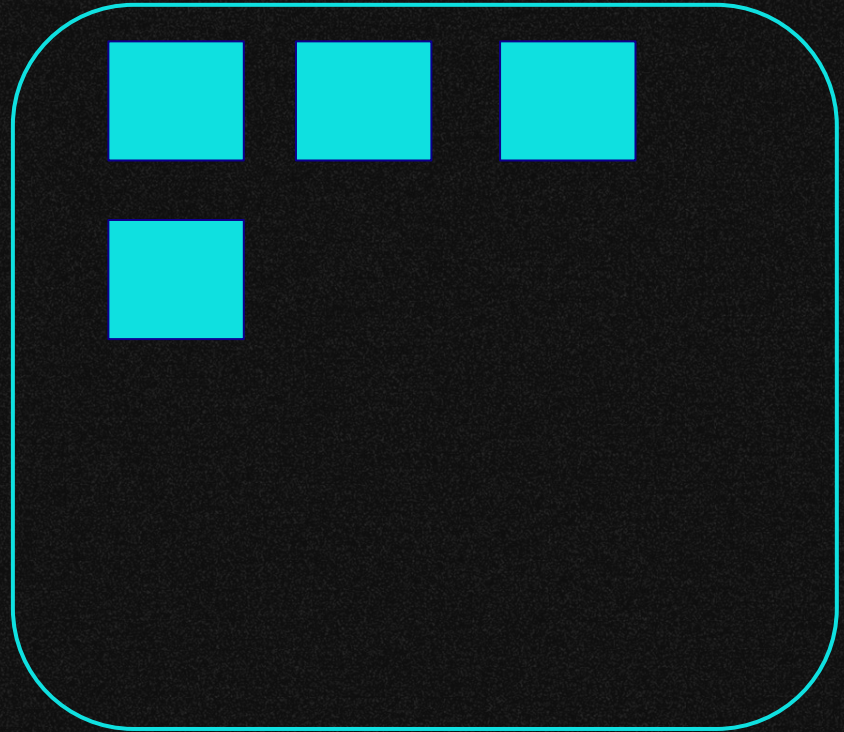
# Heapfile : Read from disk

Select page from bufferpool

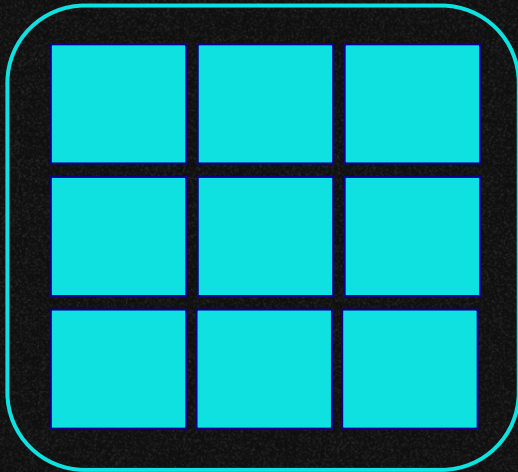


Retrieve from disk

Disk



Buffer is at capacity



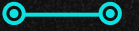
Evict LRU page



Put page in disk







## [2] Data Reorg



# Merge

Step 1: Create a separate thread for merging

Our choices:

- Merge at the Pagerange level
- Merge automatically after every 25 Tailpage updates
- Merge outside of our normal bufferpool

Execution  
Thread



Merge  
Threads





# Merge

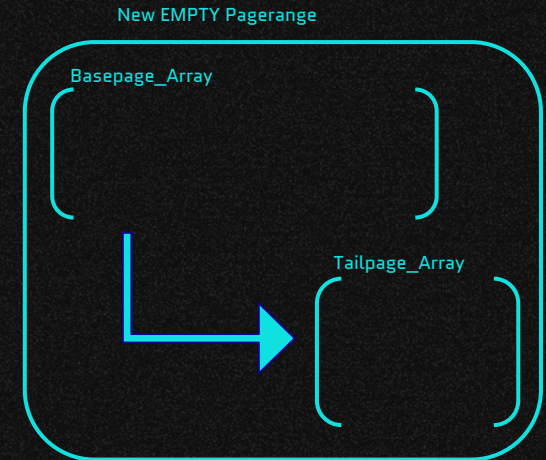
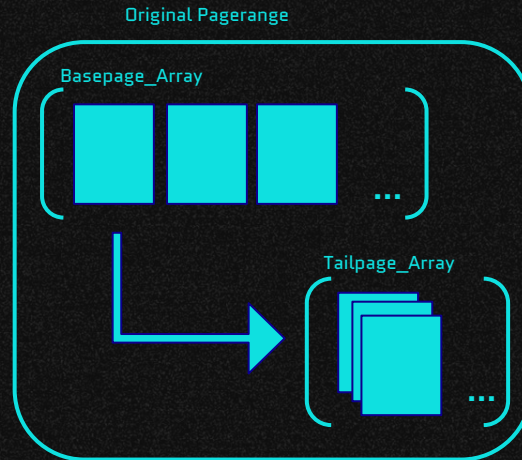
Step 2: Create a new empty Pagerange object

Pagerange object:

Basepage\_Array = [Array of Basepage Objects...]

Tailpage\_Array = [Array of Tailpage Objects...]

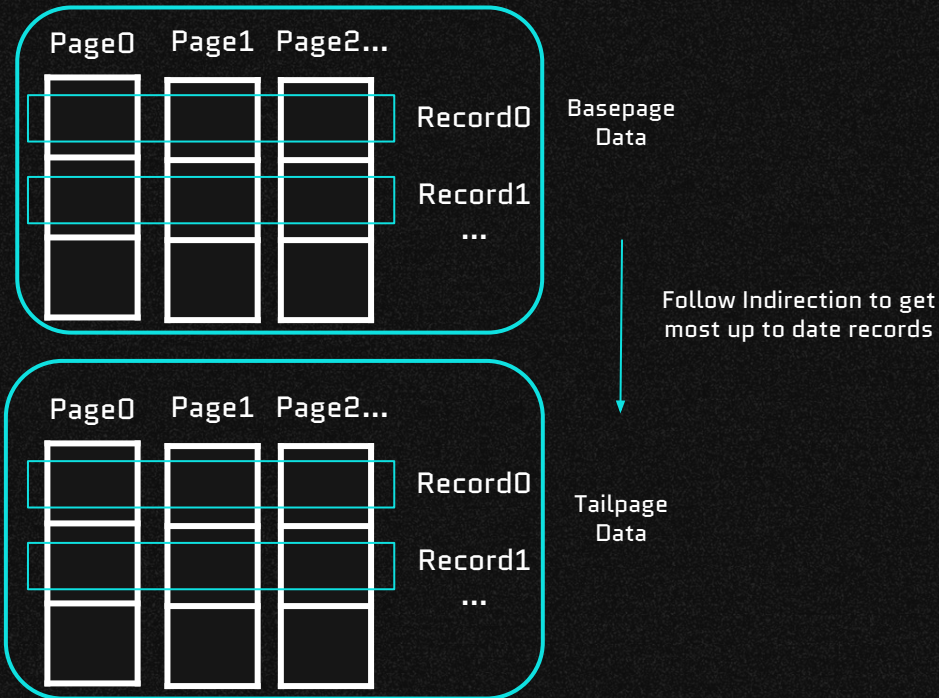
TPS = MAX\_VALUE



# Merge

## Step 3: Update the new data columns

- Consolidate the data bottom up from the Tailpages by using the Indirection column

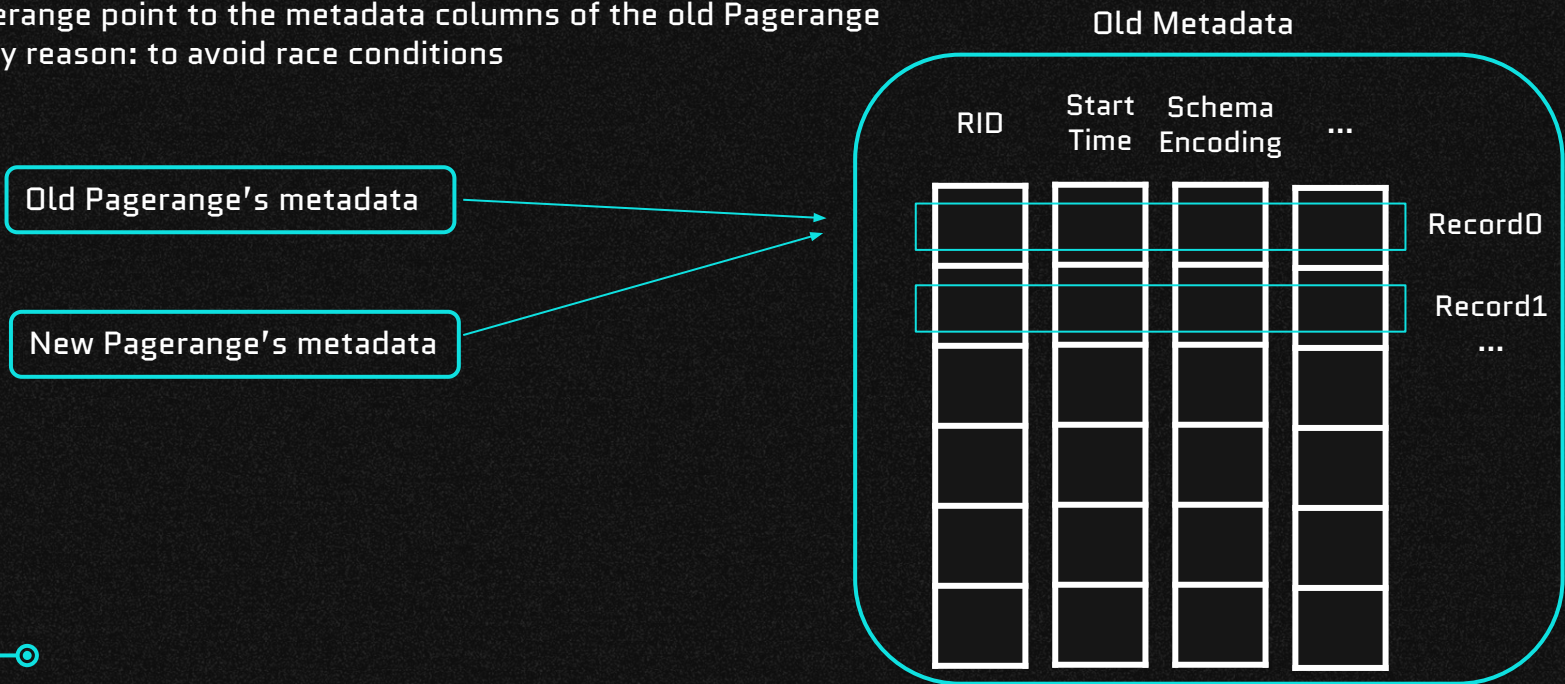




# Merge

Step 4: Have the new Pagerange object point to old metadata from the old Pagerange object

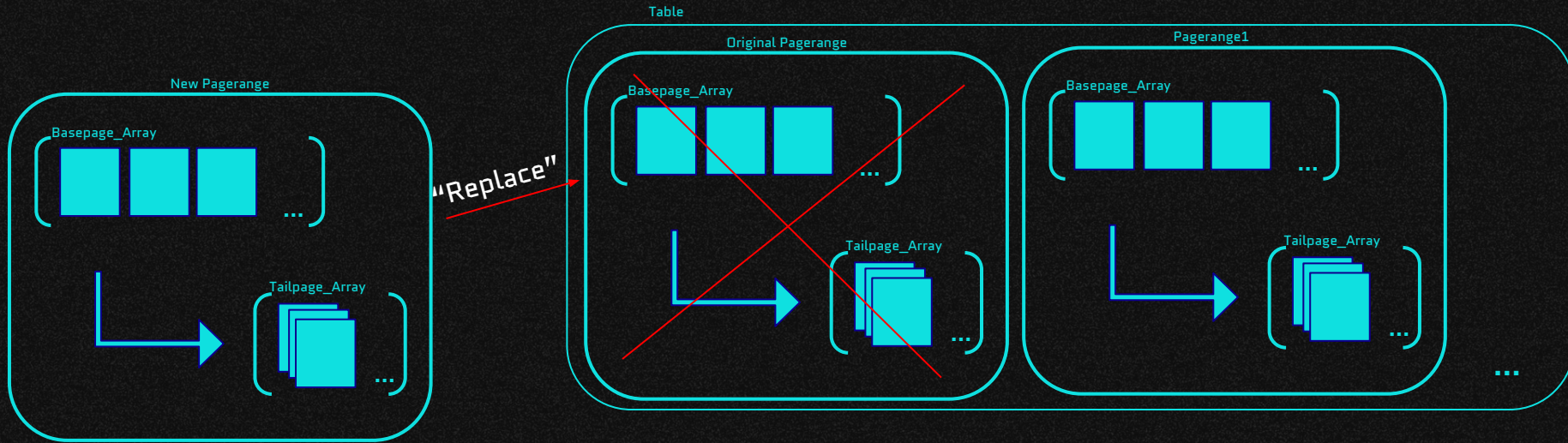
- Because everything is stored in bytearrays, we simply have the metadata columns of the new Pagerange point to the metadata columns of the old Pagerange
  - Key reason: to avoid race conditions



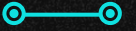


# Merge

Step 5: "Replace" old Pagerange with newly updated Pagerange





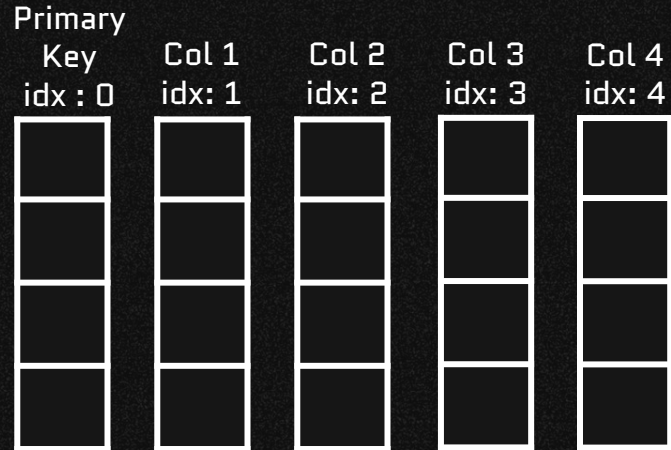


# [3] Indexing & BTree Extension



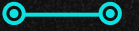
# Multiple Indices/BTrees

- Each table now has 5 Indices / BTree



- Still multiple keys in a node
- Keys are still tuples
- Secondary columns might have duplicate values
- [ value, RID ] → [ value, [RIDS] ]

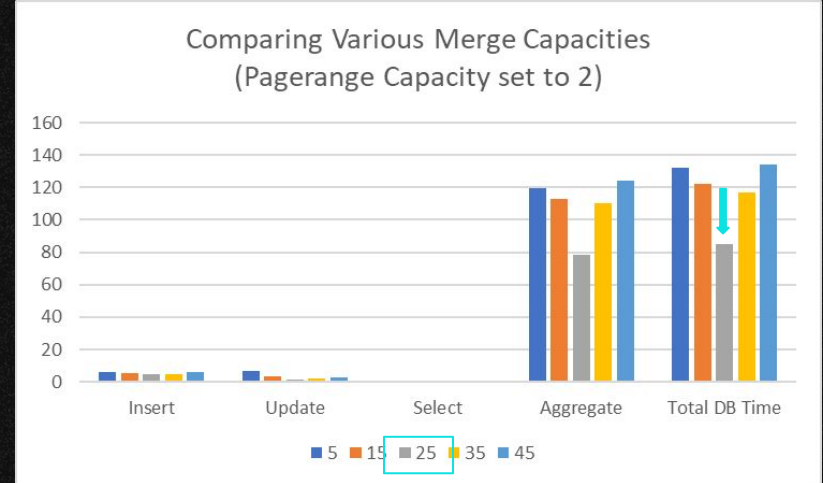
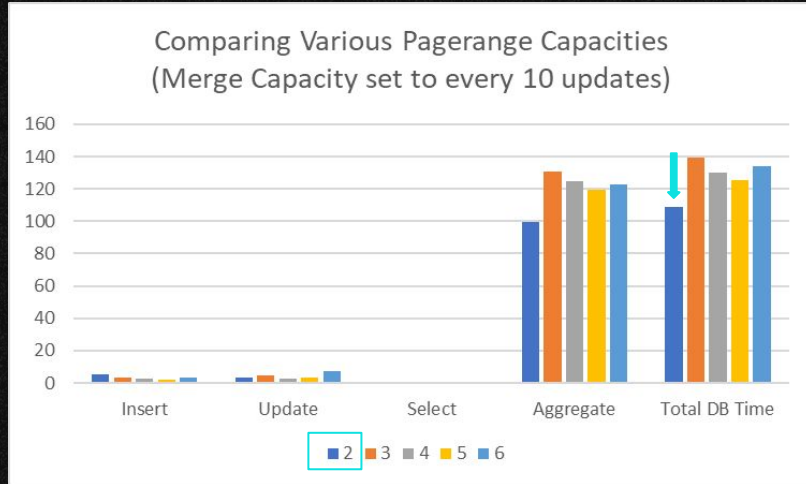




# [4] Performance



# Merge



Our final decisions:

- Pagerange Capacity = 2 Basepages
- Merge Capacity = Every 25 updates






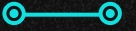
# Overall L-Store Performance

Our final decisions:

- Pagerange Capacity = 2 Basepages
- Merge Capacity = Every 25 updates

```
Inserting 10k records took:                4.6875
[<lstore.table.Record object at 0x000001BADBF3FB38>]
UPDATE-----
Updating 10k records took:                  1.859375
Selecting 10k records took:                 0.40625
Aggregate 10k of 100 record batch took:    124.359375
total db time: 131.3125
```





**[5] Live Demo and Q&A**  
**Thank you!**

