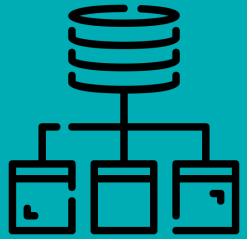
MileStone #2

By: Kiwi Calvin Bear Squid

Our Structure

Introduces data **persistence** by storing changes on disk to prevent data loss. **Bufferpool management** with LRU eviction protocol

Durability & Bufferpool



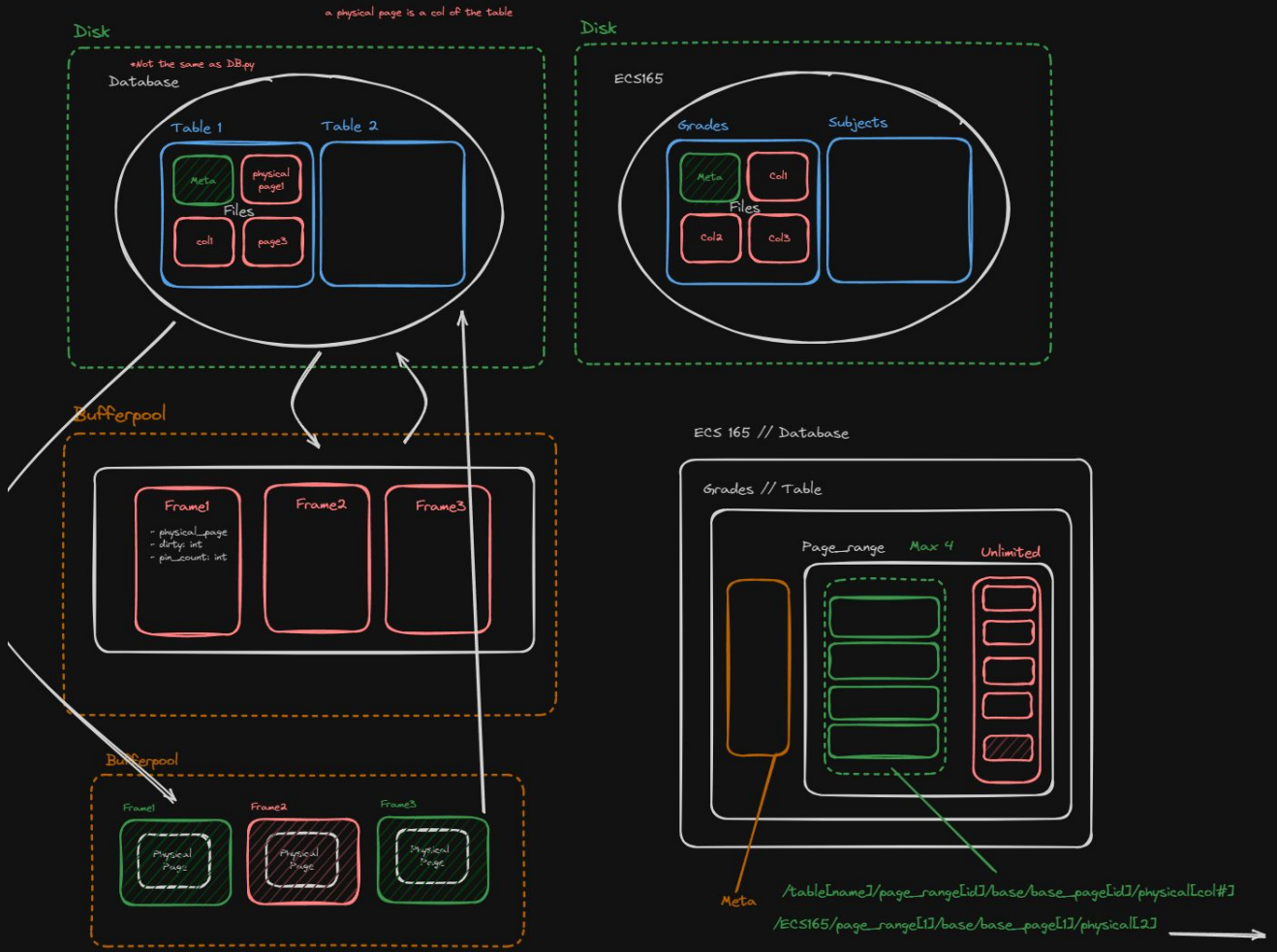
Implements a background merge process of tail into base page.

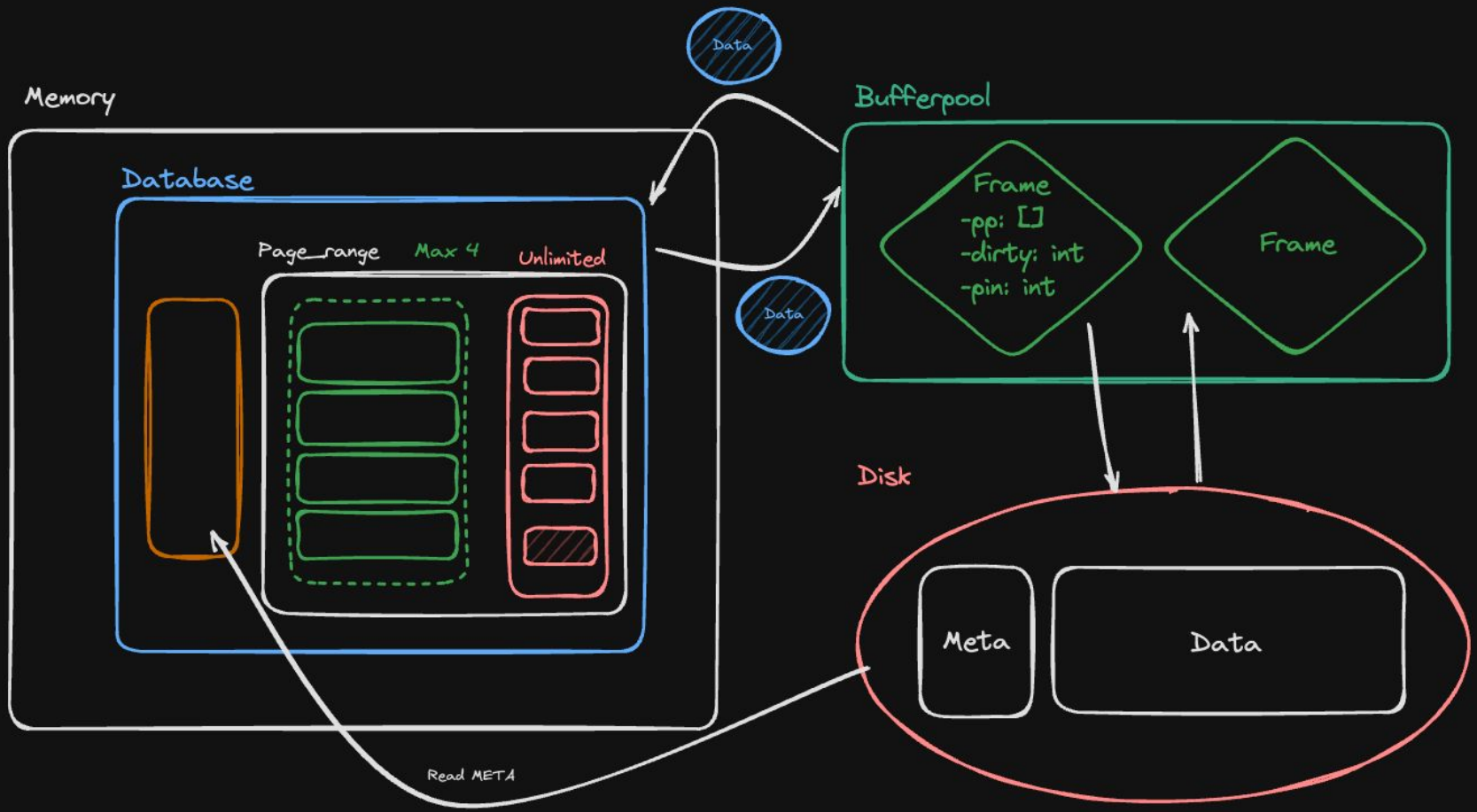
Contention-free Merge

Allows creation of indexes on any column to enhance query speed.

Indexing

Durability & Bufferpool Extension Slide(s):

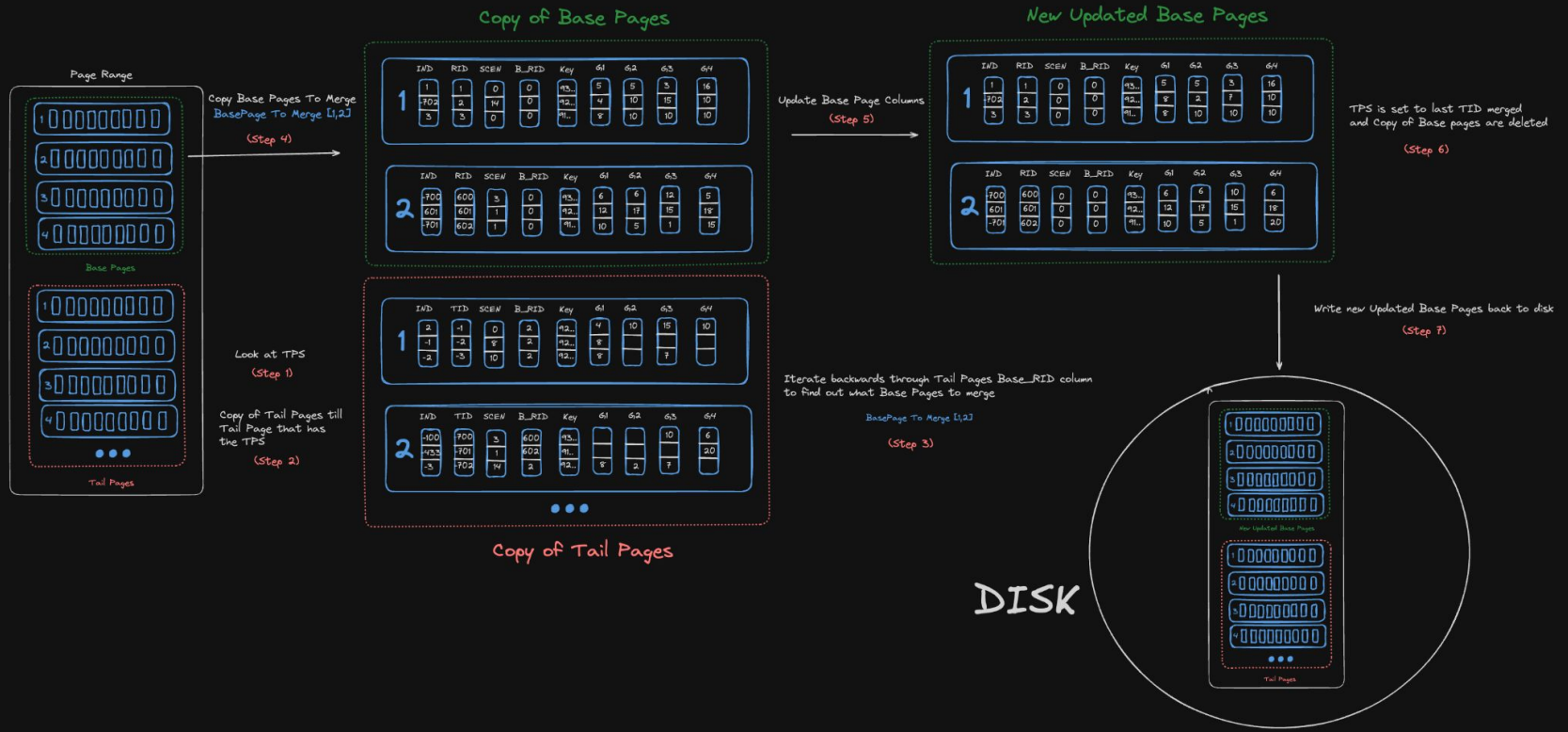

a physical page is a col of the table
Disk
*Not the same as DB.py
Database
Table 1
Table 2
Meta
physical page1
Files
col1
page3
Disk
ECS165
Grades
Subjects
Meta
Col1
Files
Col2
Col3
Bufferpool
Frame1
Frame2
Frame3
- physical_page
- dirty: int
- pin_count: int
Bufferpool
Frame1
Frame2
Frame3
Physical Page
Physical Page
Physical Page
ECS 165 // Database
Grades // Table
Page_range
Max 4
Unlimited
Meta
/table[name]/page_range[id]/base/base_page[id]/physical[col#]
/ECS165/page_range[1]/base/base_page[1]/physical[2]


Data
Memory
Bufferpool
Database
Frame
-pp: []
-dirty: int
-pin: int
Frame
Page_range
Max 4
Unlimited
Data
Disk
Meta
Data
Read META

Data Reorg: Contention-free Merge Slide(s):

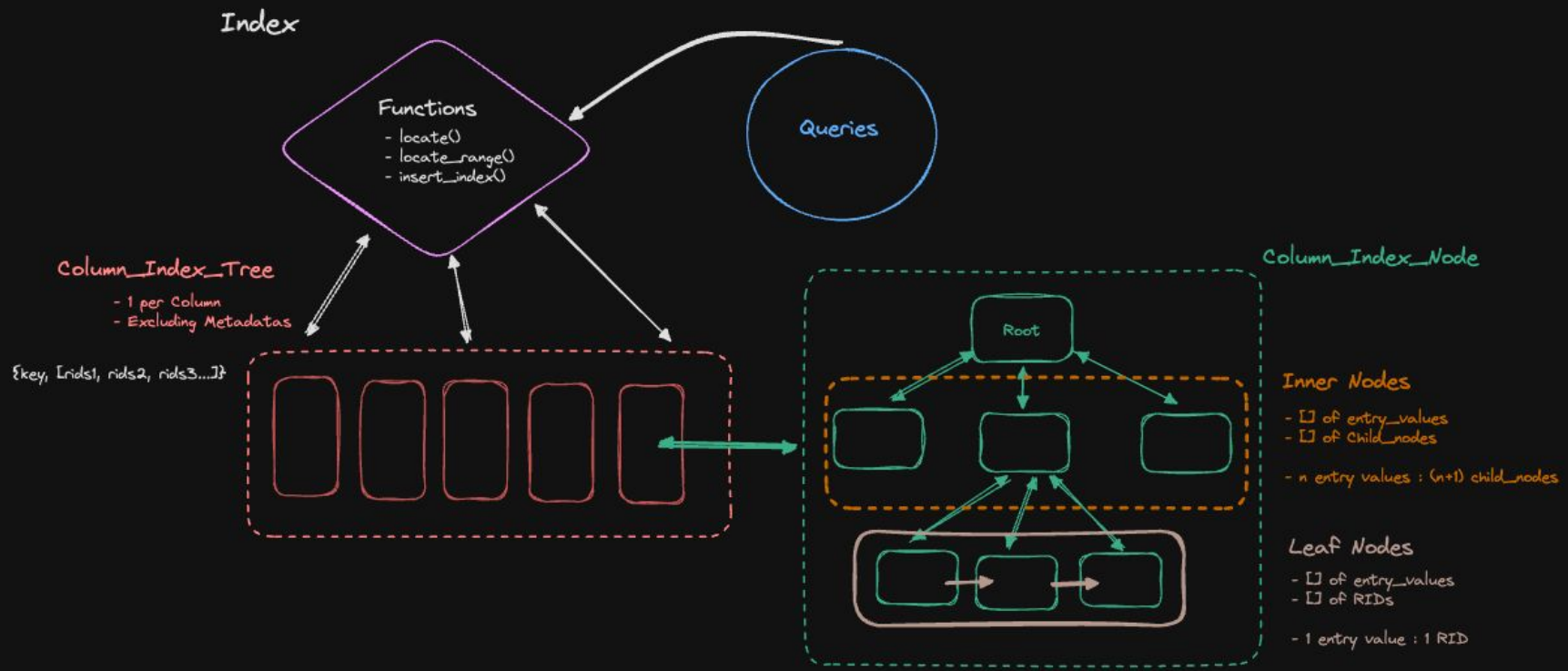
Contention-free Merge:

- **Merge after 1024 updates to a Page Range**

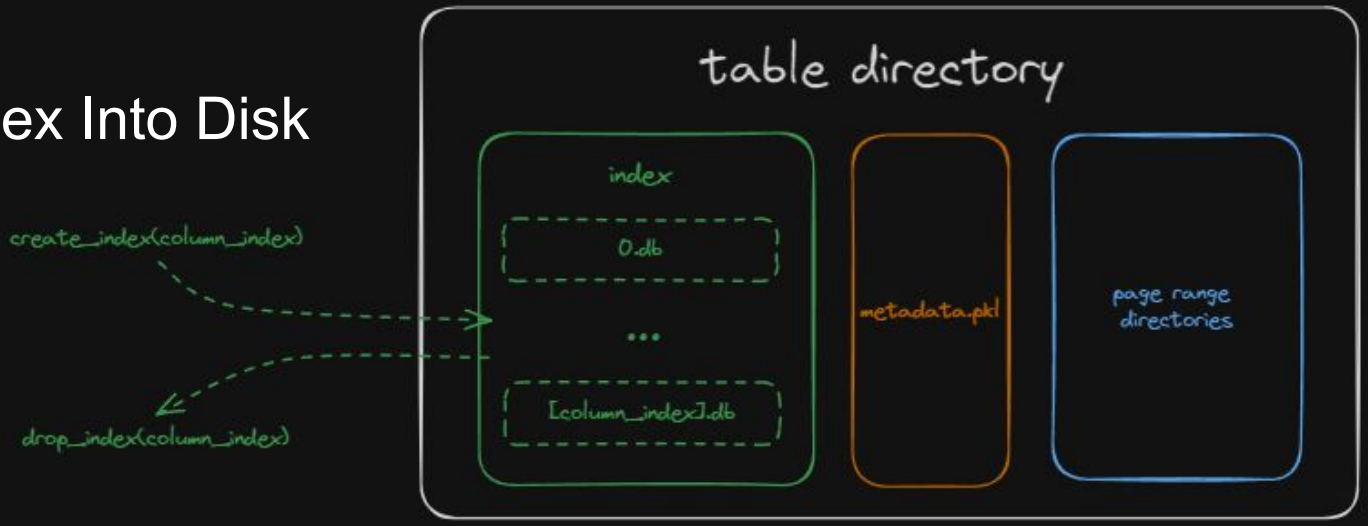


Indexing Slide(s):

Index: Previous Implementation



Implementing Index Into Disk

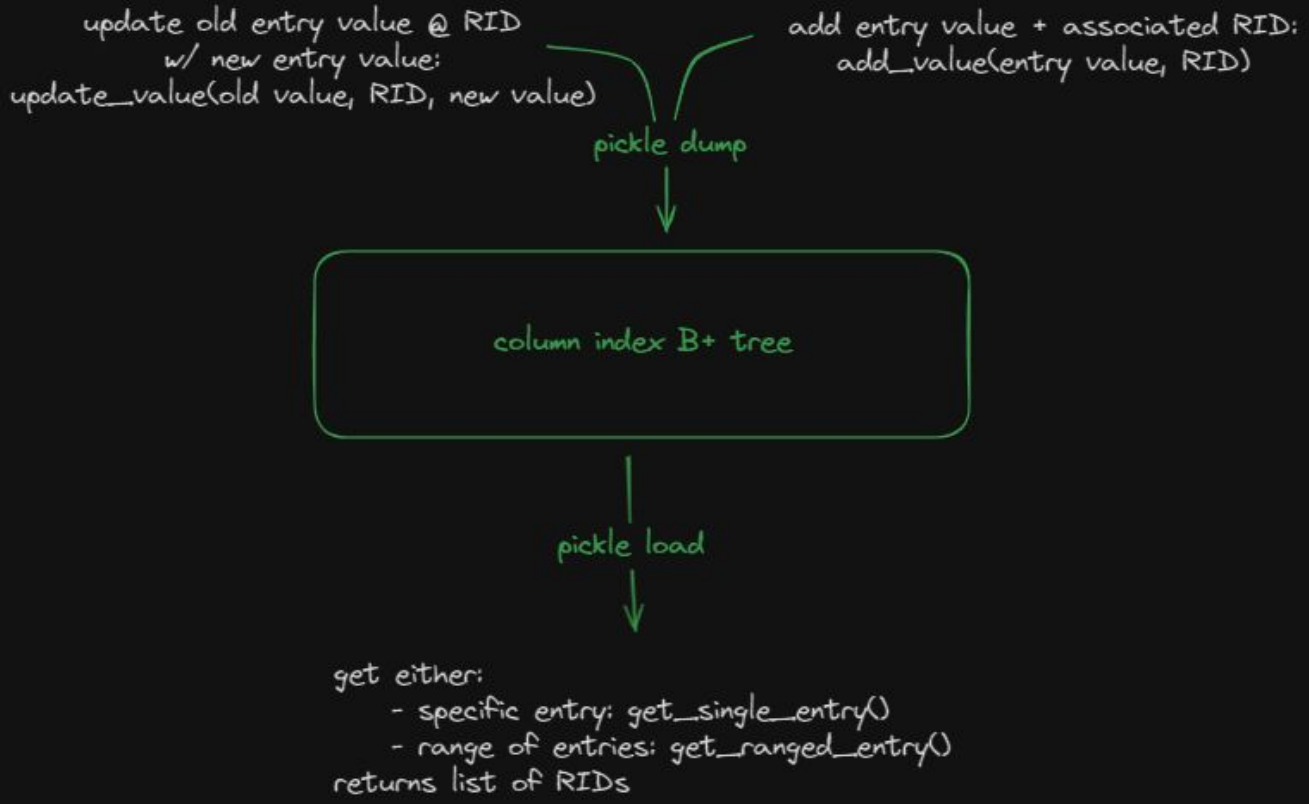


At initialization:
- if index directory exists
  -> load indices
- else
  -> empty index directory will be initialized
  -> [key_index].db initialized

create_index(column_index):
- if index for column found -> load into index dictionary
- else -> create .db file + load into index dictionary

drop_index(column_index):
- deletes file associated to column index from the index directory

Index Data Pickling



Things Still Needed to Implement

- Scan disk to find all column values and their corresponding RIDs to produce an index
- Version Control
  - Select Version
  - Sum Version