Schema Squad

Milestone 2 Presentation

Assigned Team Roles

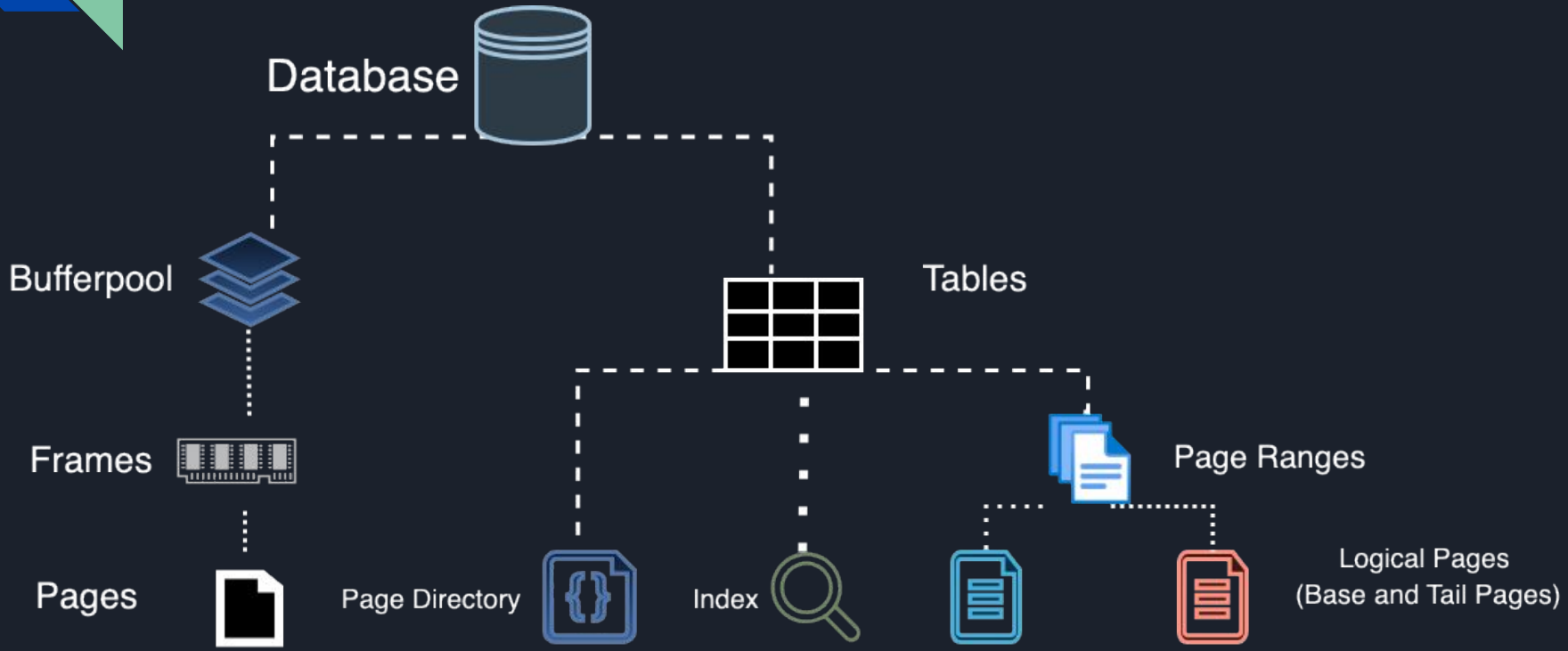
George Zavala	Team Coordinator: This role involves leading and coordinating the group throughout the project
Gabbriel Bui	Developer: Responsible for the actual coding and implementation of the database system
Mateo Escobar	Tester: Responsible for ensuring the quality and reliability of the database system
Patrick Manson	Developer: Responsible for the actual coding and implementation of the database system
Ibrahim Siddiqui	System Architect/Developer: Responsible for the overall design of the database system
Chang Da Su Liang	Tester: Responsible for ensuring the quality and reliability of the database system

Table of Contents

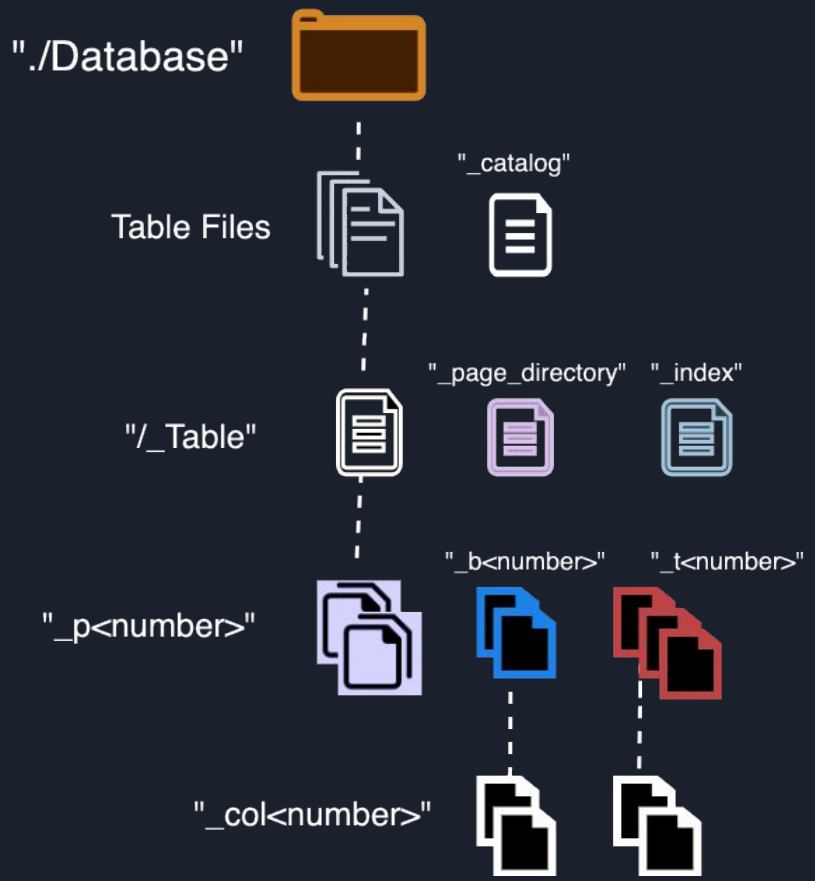
1 Disk and Bufferpool Management

- 1.1 Logical Structure
- 1.2 File Structure (Disk)
- 1.3 Bufferpool
- 1.4 Bufferpool Size Testing

1.1 - Logical Structure



1.2 - File Structure (Disk)



Examples for Clarity

-Database: “./ECS165”

-Catalog: “./ECS165/_catalog”

-Table: “./ECS165/Grades”

-Table’s Page Directory: “./ECS165/Grades_page_directory”

-Table’s Index: “./ECS165/Grades_index”

-Table’s First Page Range: “./ECS165/Grades_range0”

-Table’s First Base Page: “./ECS165/Grades_range0_b0”

-Table’s First Tail Page: “./ECS165/Grades_range0_t0”

-First Column in First Base Page: “./ECS165/Grades_range0_b0_col0”

-First Column in First Tail Page: “./ECS165/Grades_range0_t0_col0”

-Page directories and indices are persisted with pickle; everything else is stored manually with files

1.3 - Bufferpool

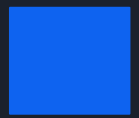
-Bufferpool contains frames (a physical page along with overhead)

-Pins frames that are being used; cannot evict a pinned frame

-Fetches from disk if requested physical page is not in the bufferpool

-Uses an LRU eviction policy (the frame that was accessed the longest time ago is less likely to get accessed again, it is evicted when the bufferpool is full)

-Dirty frames' physical pages are written back to disk

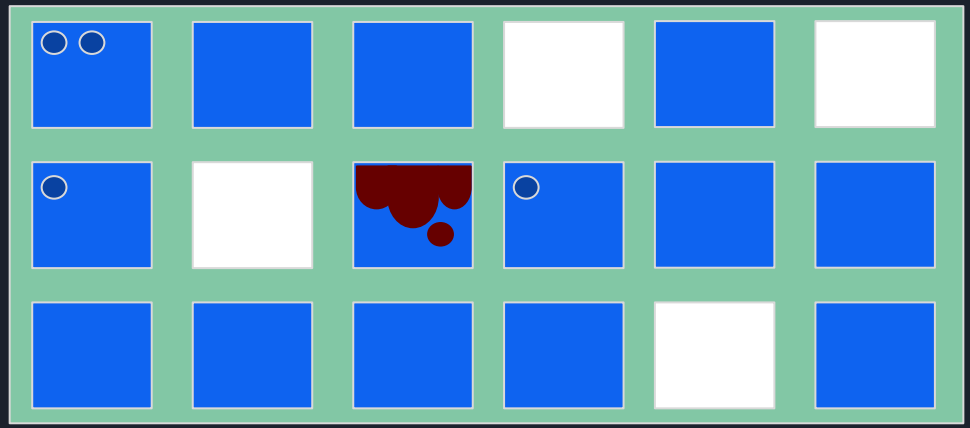


Full Frame

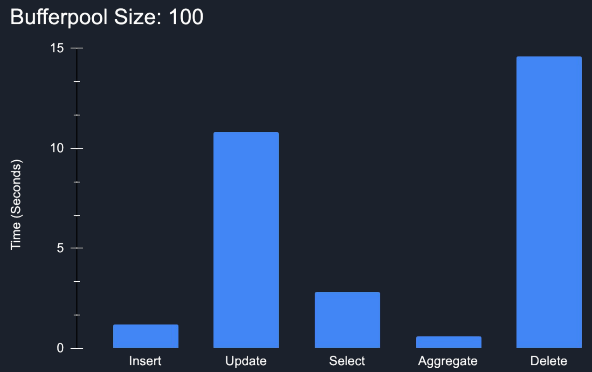
Pin

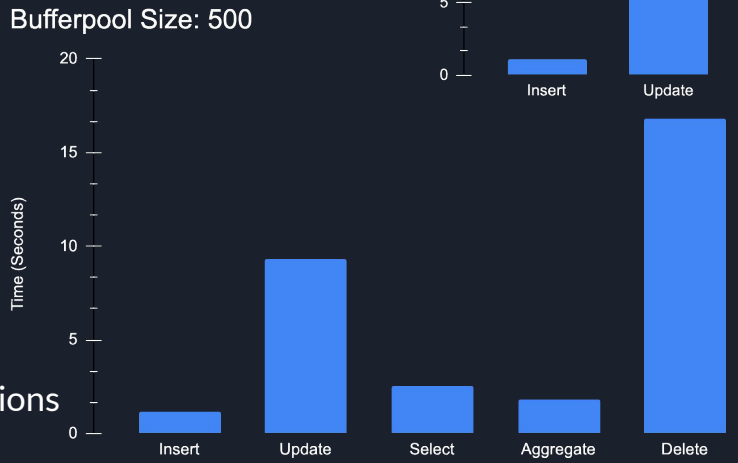
Empty Frame

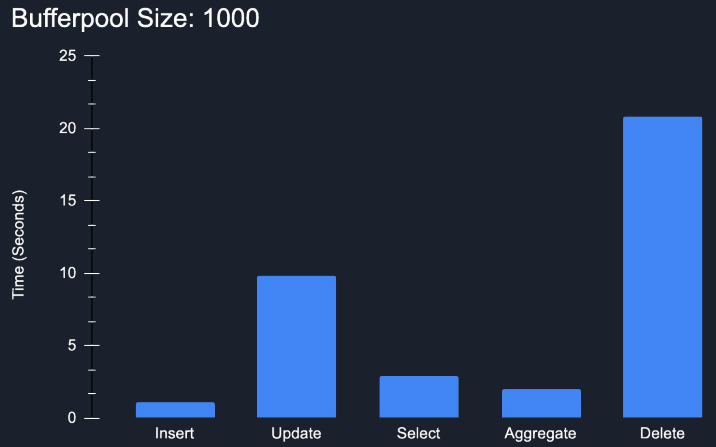
Dirty Frame



1.4 - Bufferpool Size Testing







-10k insertions, updates, selections, and deletions
-Aggregate: 10k of 100 record batch

2 Merging

Merging

-Added column that stores the base RID of each record for merging

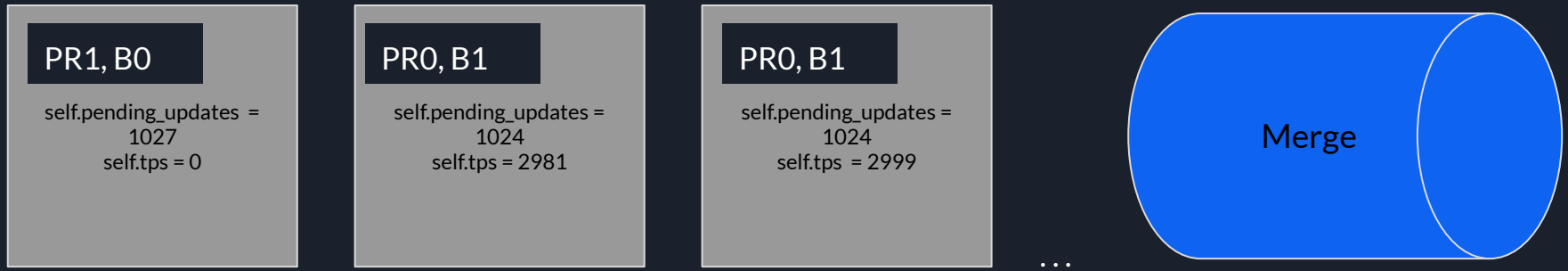
-Merge at the base page level (i.e. all columns except the metadata columns) each time one receives 1024 updates (and is full)

-Iterate through the tail pages in its page range backwards

-Keep track of the updated base records with a dictionary with base RIDs as keys so as to not update a base record with an older tail record

-Stop when the tail record's RID is less than the TPS (i.e. it has already been applied to the base page) or all the tail pages have been iterated through (initial case when TPS = 0)

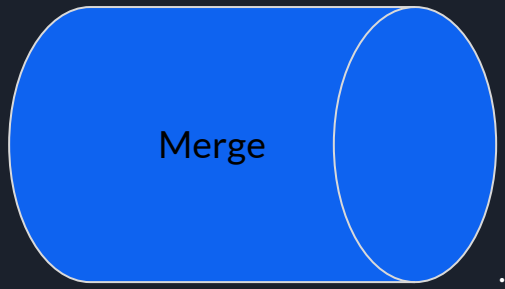
Merge Queue



Merging (cont.)

-The consolidated (merged base pages) are copies of the original base pages that have gone through the merge process

-The original base pages in the bufferpool are renamed (old) and the new base pages take their name; the page directory doesn't need to be updated as a result



...

Consolidated Base Pages

PR1, B0

self.pending_updates = 0
self.tps = 4907

PR0, B1

self.pending_updates = 3
self.tps = 3765

PR0, B1

self.pending_updates = 2
self.tps = 3991

3 Secondary Indices

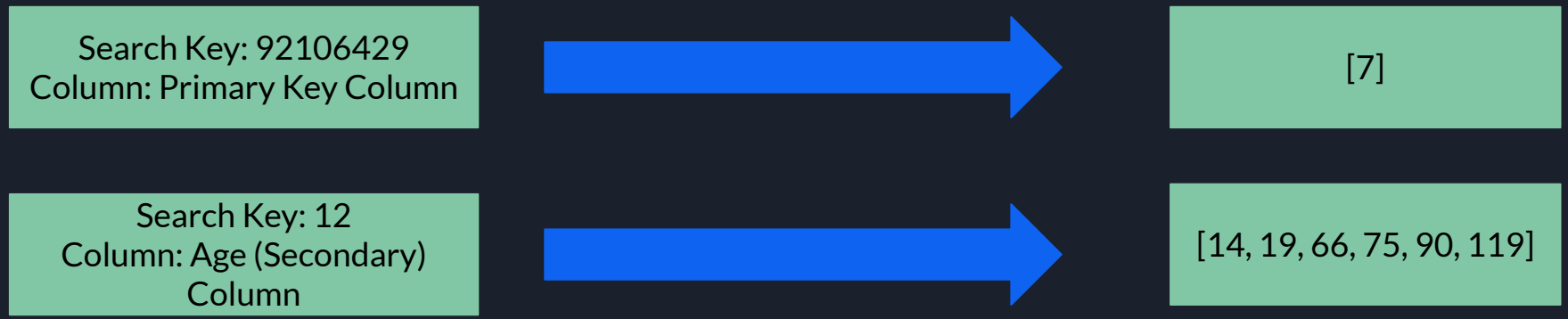
Indices of Secondary Columns

-Still using dictionaries for the indices (i.e. hash tables)

-Primary keys map to their respective base RIDs

-Secondary keys (search values for columns other than the primary key column) map to a list of both base and tail RIDs corresponding to the records with the value; this is to efficiently retrieve all records with a given value for select_version

-Empty list is returned if there is no index on the column or the value doesn't exist in the column



Questions