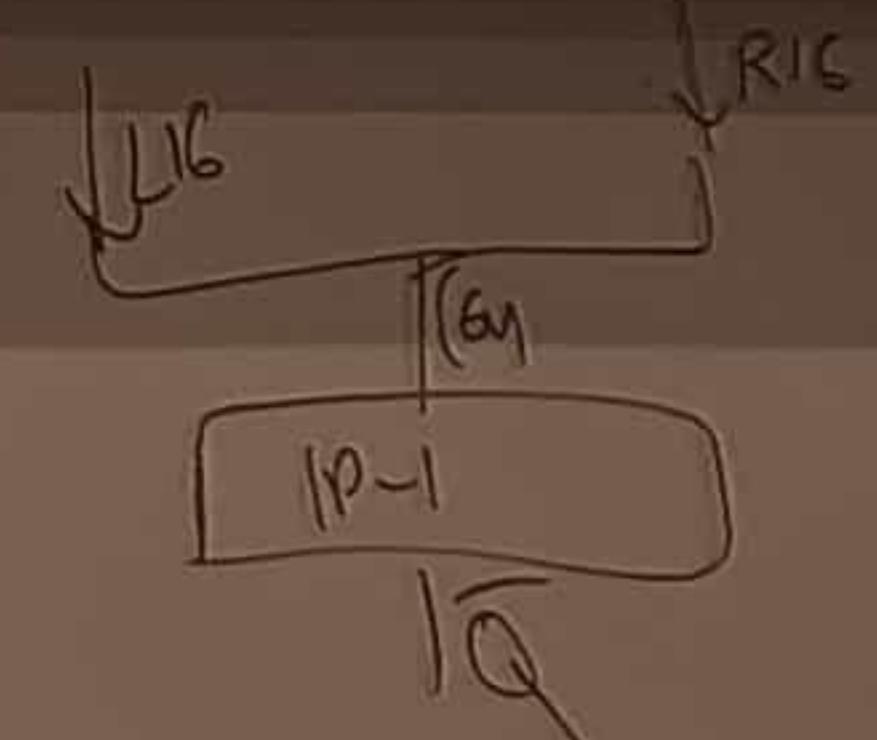


Хеш-таблица

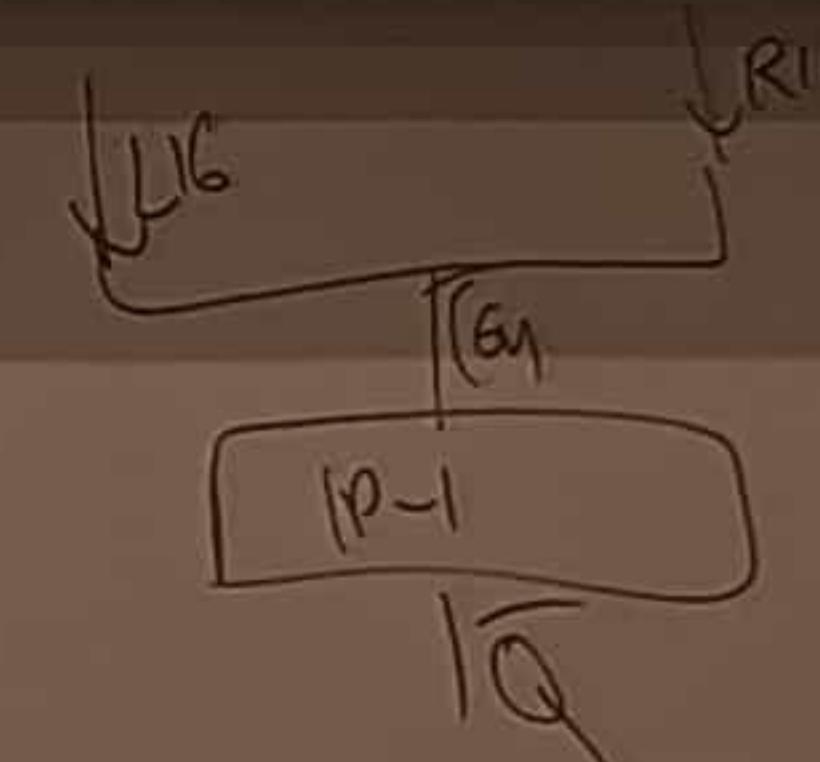
- Данные хранятся в файле, который можно только дополнять (append-only log).
- В оперативной памяти хранится хеш-таблица (словарь).
- Ключ → Смещение (offset) в файле данных.



Хеш-таблица

Сегменты

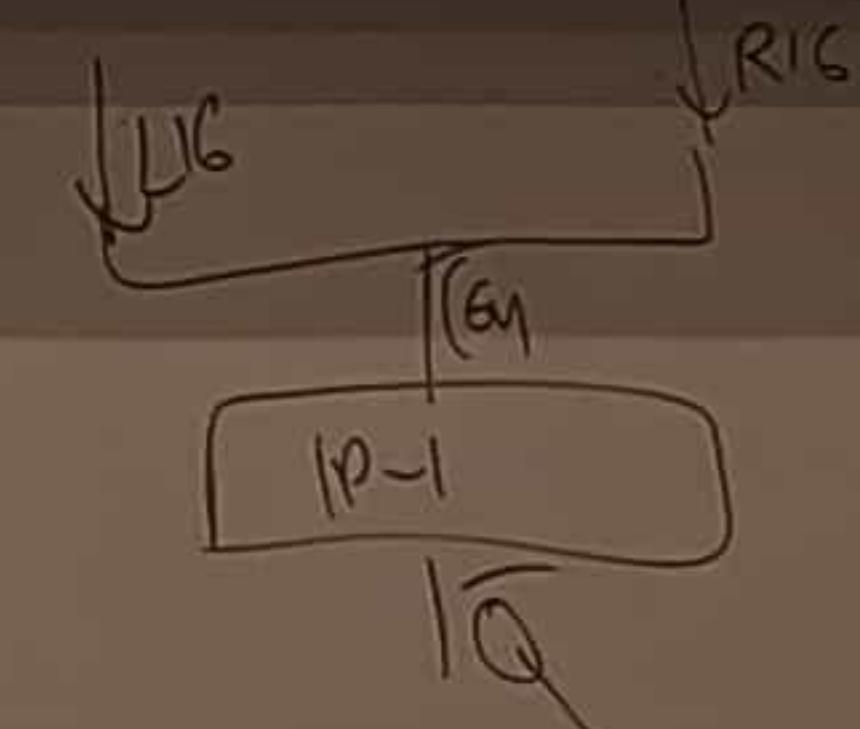
- Разбить журнал на сегменты фиксированного размера.
- При заполнении сегмента — закрыть его и начать новый.
- Запустить фоновый процесс уплотнения (compaction) и слияния (merging) сегментов.



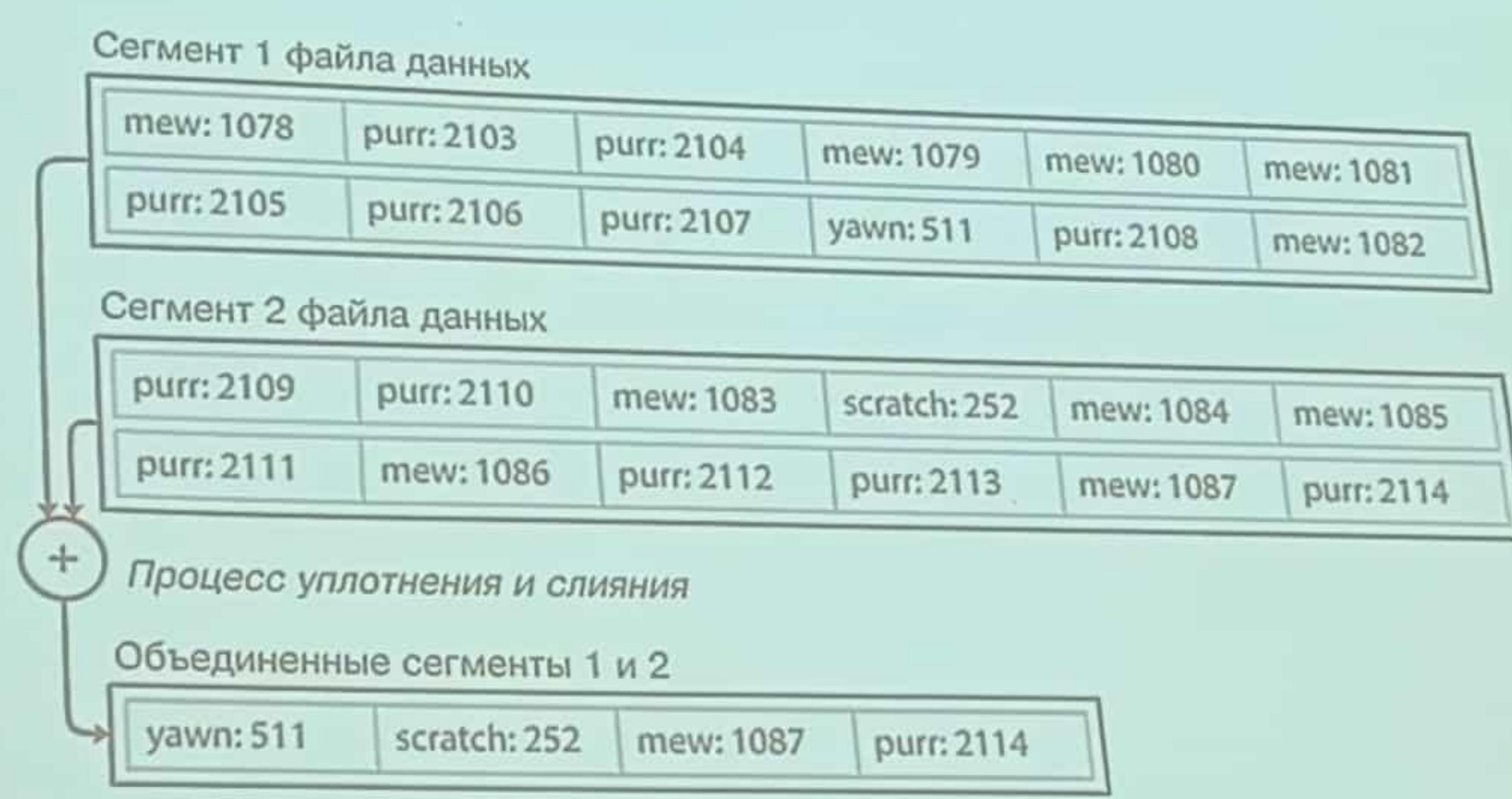
Хеш-таблица

Слияние

- Объединение нескольких небольших сегментов в один новый.
- Позволяет уменьшить общее количество сегментов.



Хеш-таблица

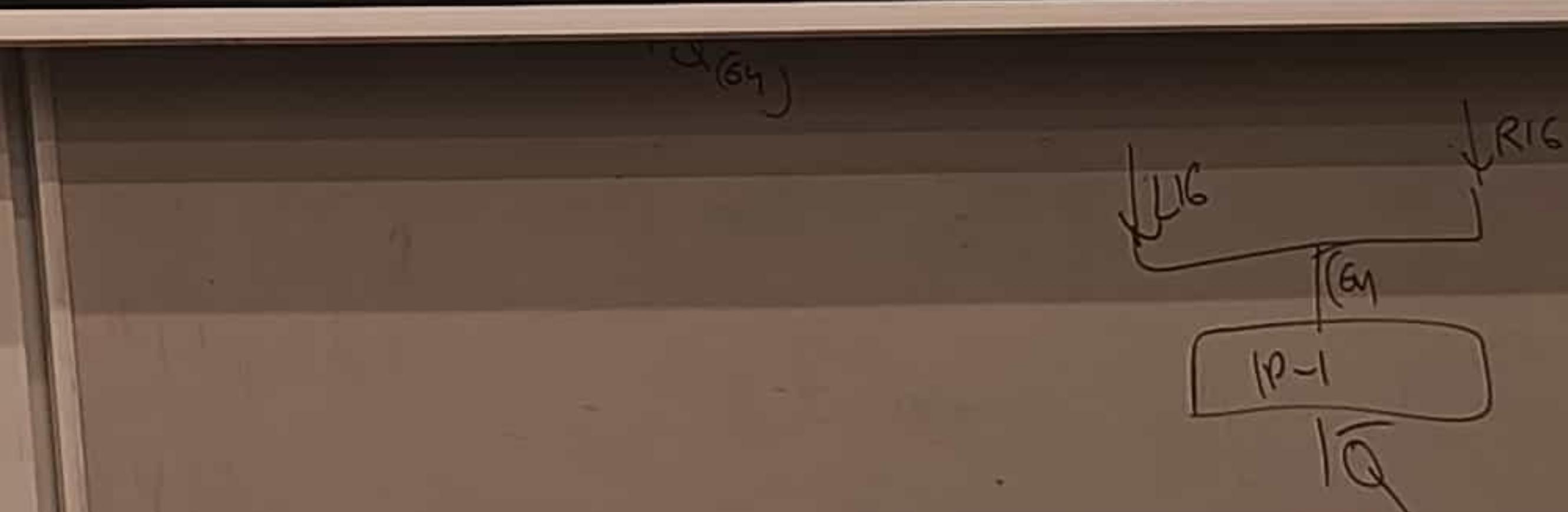


W6
T61
P61
TQ

Хеш-таблица

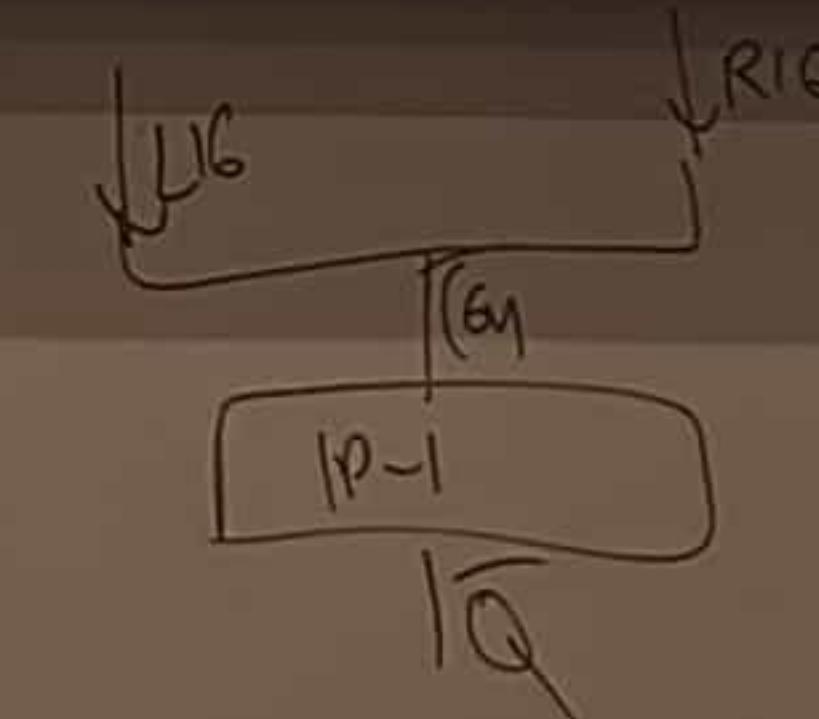
Ограничение

- Ключи должны помещаться в ОЗУ. Хеш-таблица на диске медленная из-за случайных чтений, дорогое расширения и сложности разрешения коллизий.
- Неэффективные запросы по диапазону. Невозможно быстро найти все ключи от 00000 до 99999. Необходимо обращаться к хеш-таблице для каждого ключа в диапазоне отдельно.



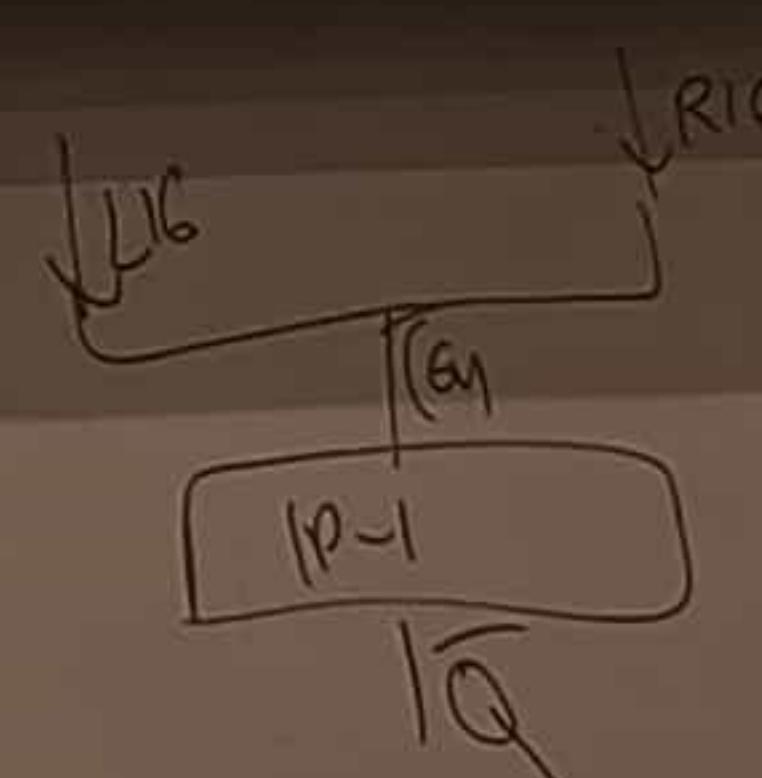
SS-table Особенности

- Данные отсортированы по ключу
- Каждый ключ встречается только один раз (обеспечивается уплотнением)
- Порядок записи значений не важен - приоритет у более новых значений



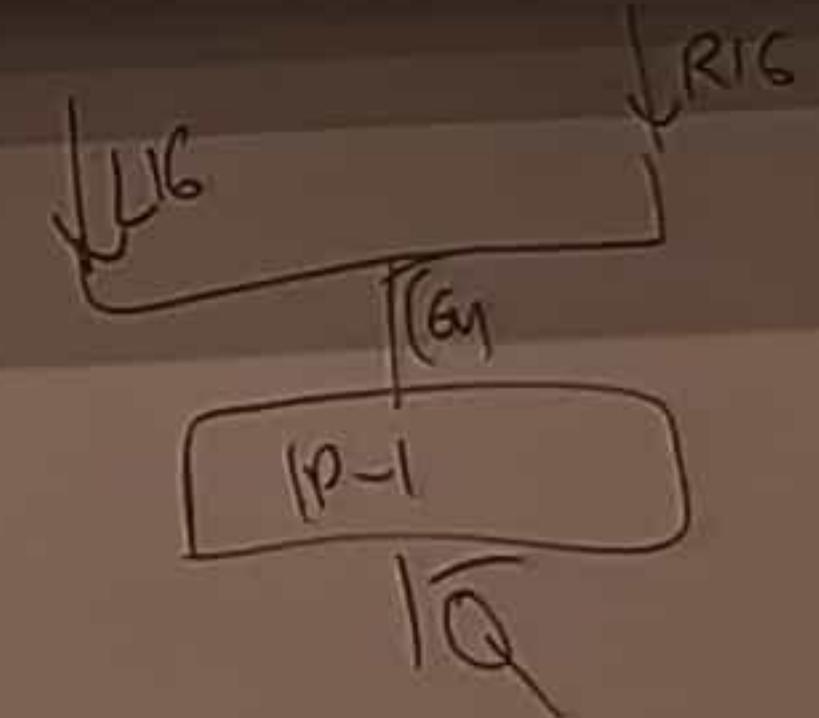
Слияние

- Алгоритм, аналогичный сортировке слиянием
- Работает даже когда данные не помещаются в оперативной памяти
- При конфликте ключей берется значение из самого нового сегмента



Разреженный индекс

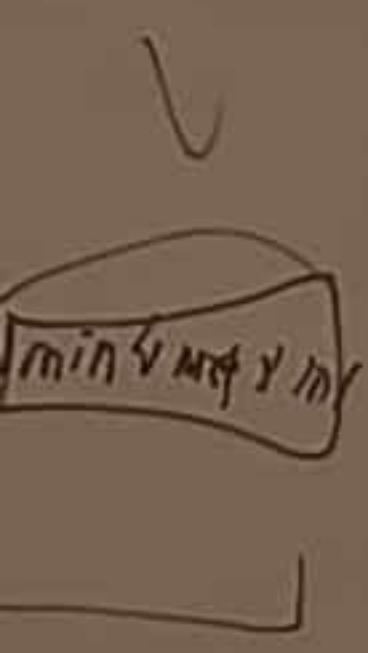
- Не нужно хранить все ключи - только некоторые ориентиры
- Пример: известны смещения для "handbag" и "handsome" → "handiwork"
- Можно быстро просканировать небольшой диапазон
-



MemTable

- Сбалансированное дерево в оперативной памяти (красно-черное, AVL)
- Данные сохраняются отсортированными по ключу
- Быстрая вставка в любом порядке

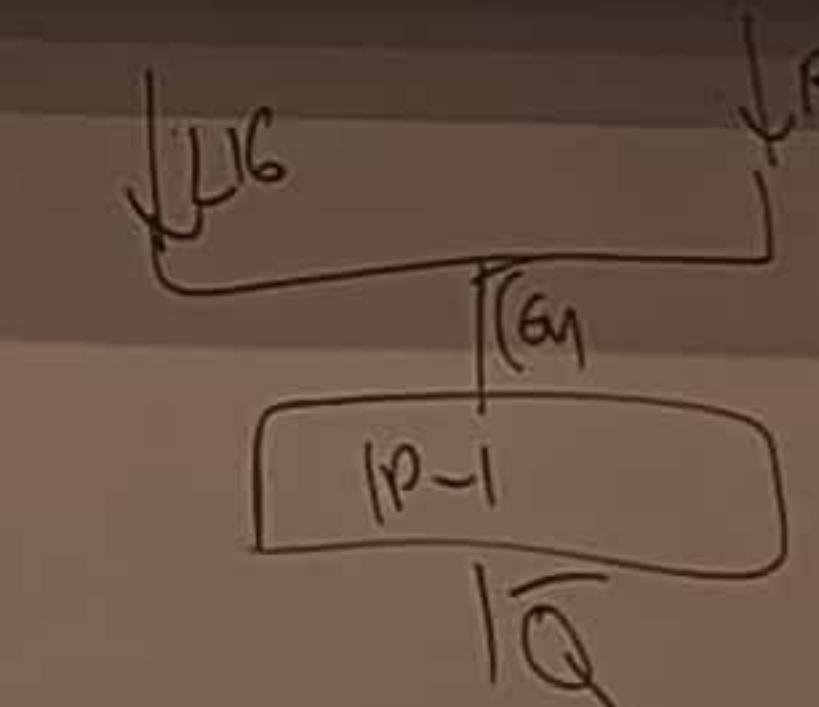
лек



64



Q₍₆₄₎



RIG

Надежность

- Все операции немедленно записываются в журнал
- Журнал неупорядочен - используется только для восстановления
- После записи MemTable в SS-таблицу журнал удаляется

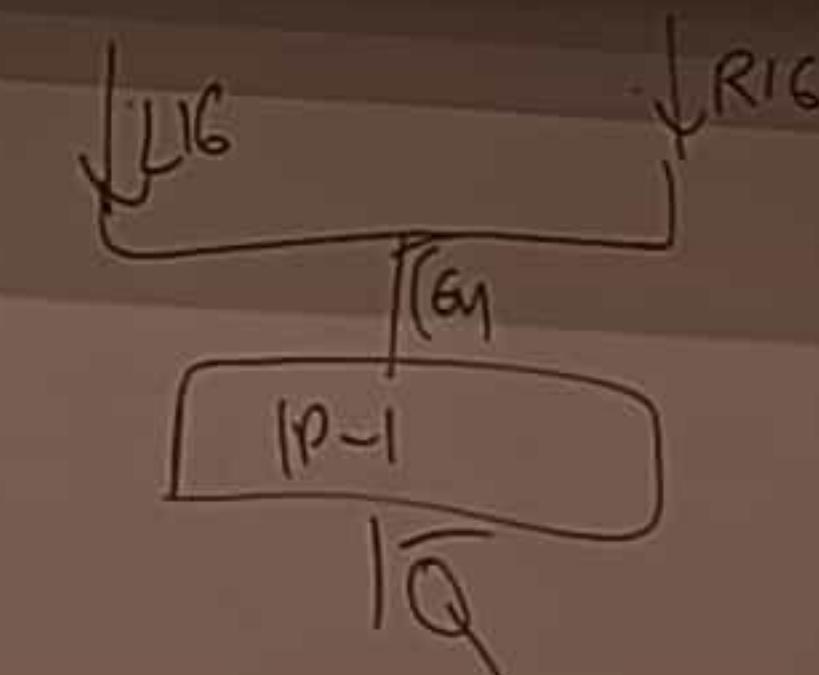


Алгоритм работы

- Запись: Добавляется в MemTable и параллельно пишется в журнал для надежности
- Чтение: Поиск в MemTable → последнем сегменте → предыдущем и т.д.
- Фоновые процессы: При превышении лимита MemTable записывается в SS-таблицу, Регулярное слияние и уплотнение сегментов



Q₆₄



LSM-дерево

Фильтр Блума

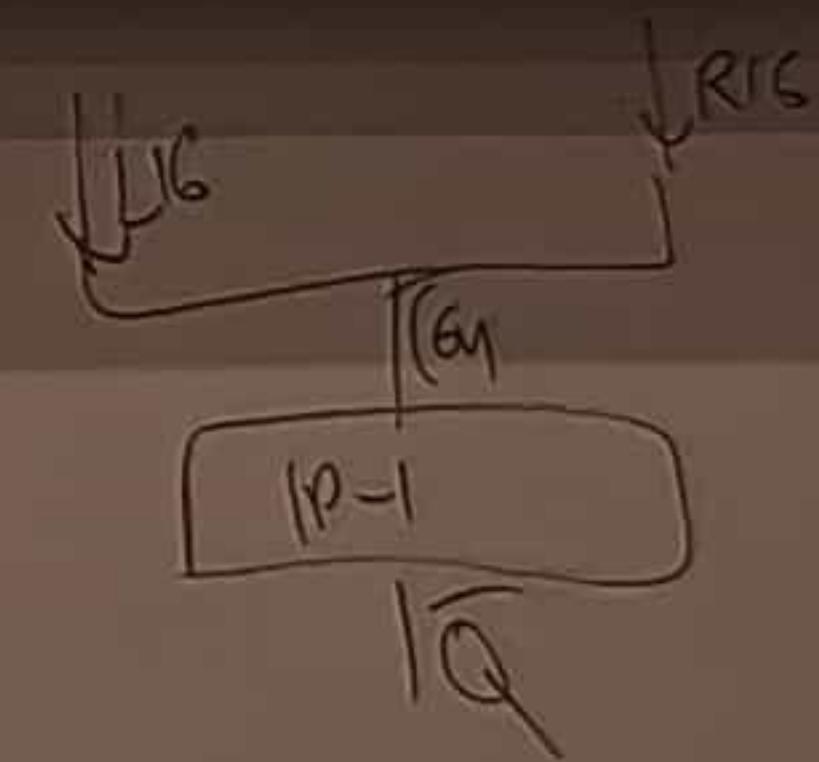
- Эффективная проверка отсутствия ключа
- Избегание ненужных чтений с диска
- "Возможно есть" / "Точно нет"

Index



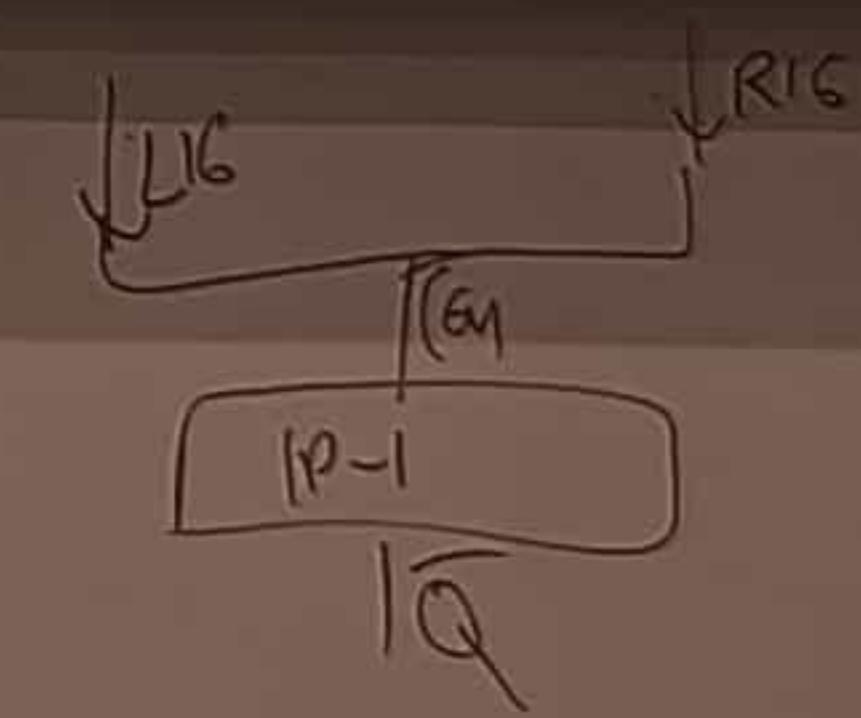
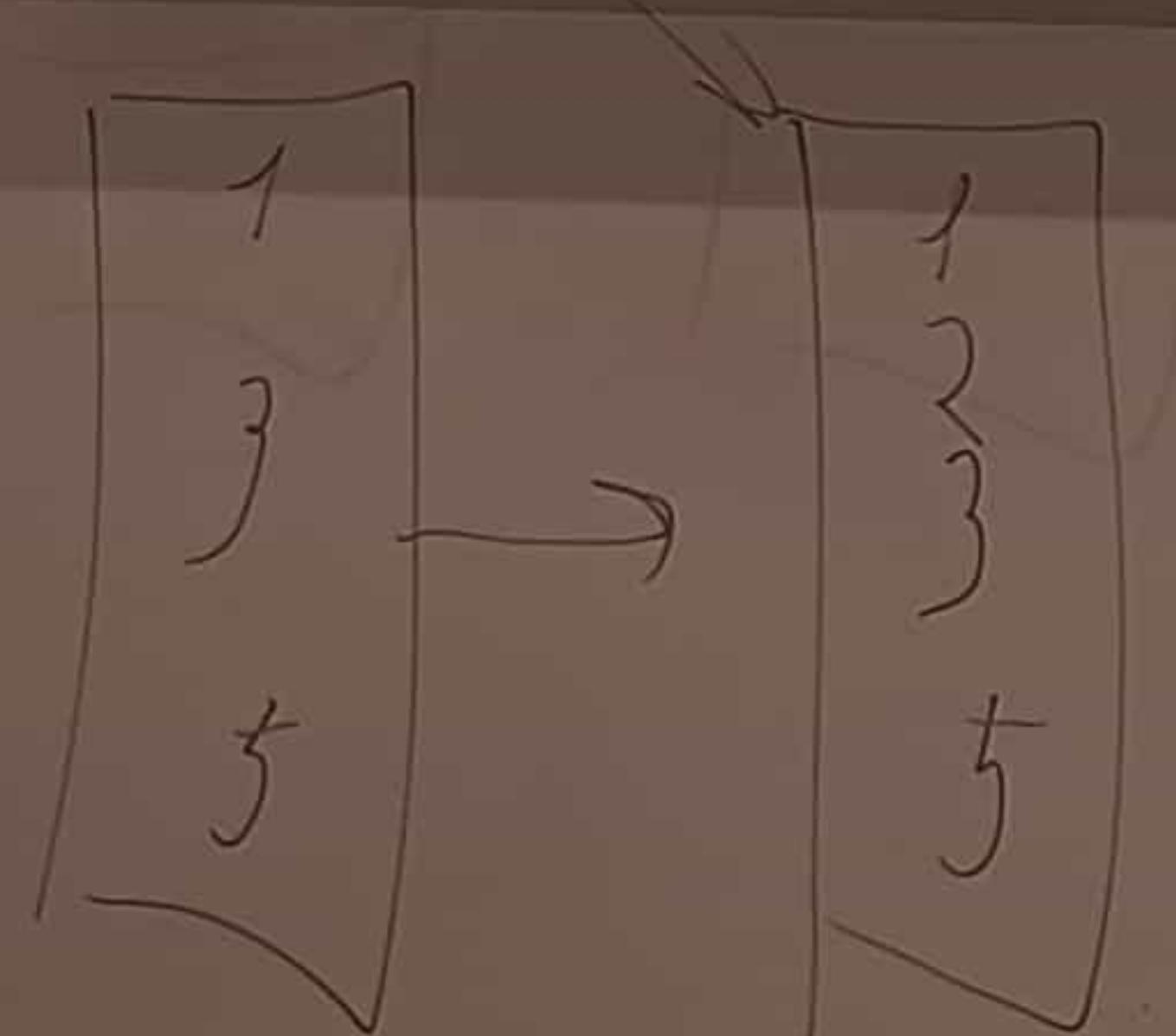
Алгоритм работы

- Запись: Добавляется в MemTable и параллельно пишется в журнал для надежности
- Чтение: Поиск в MemTable → последнем сегменте → предыдущем и т.д.
- Фоновые процессы: При превышении лимита MemTable записывается в SS-таблицу, Регулярное слияние и уплотнение сегментов



Уплотнение

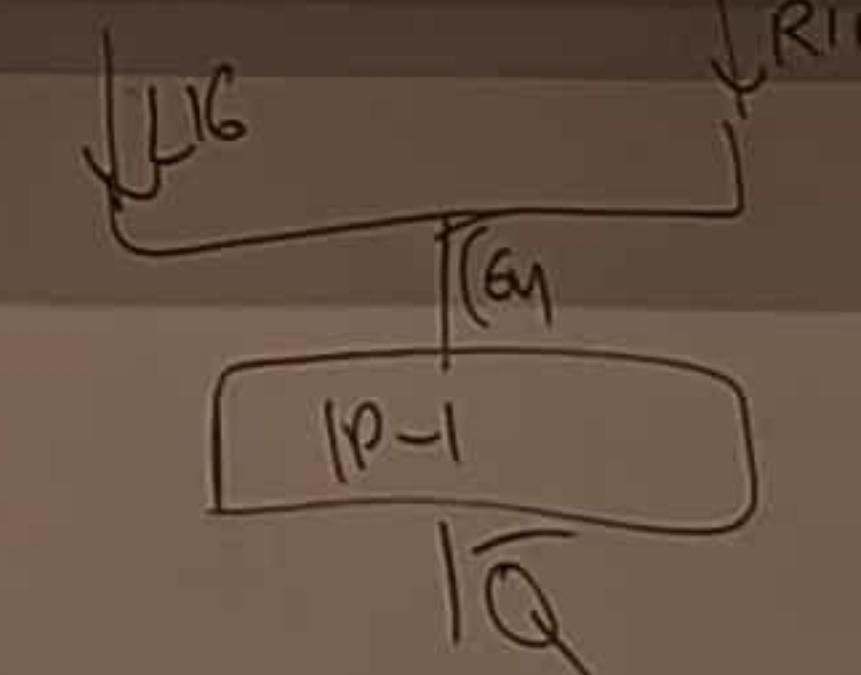
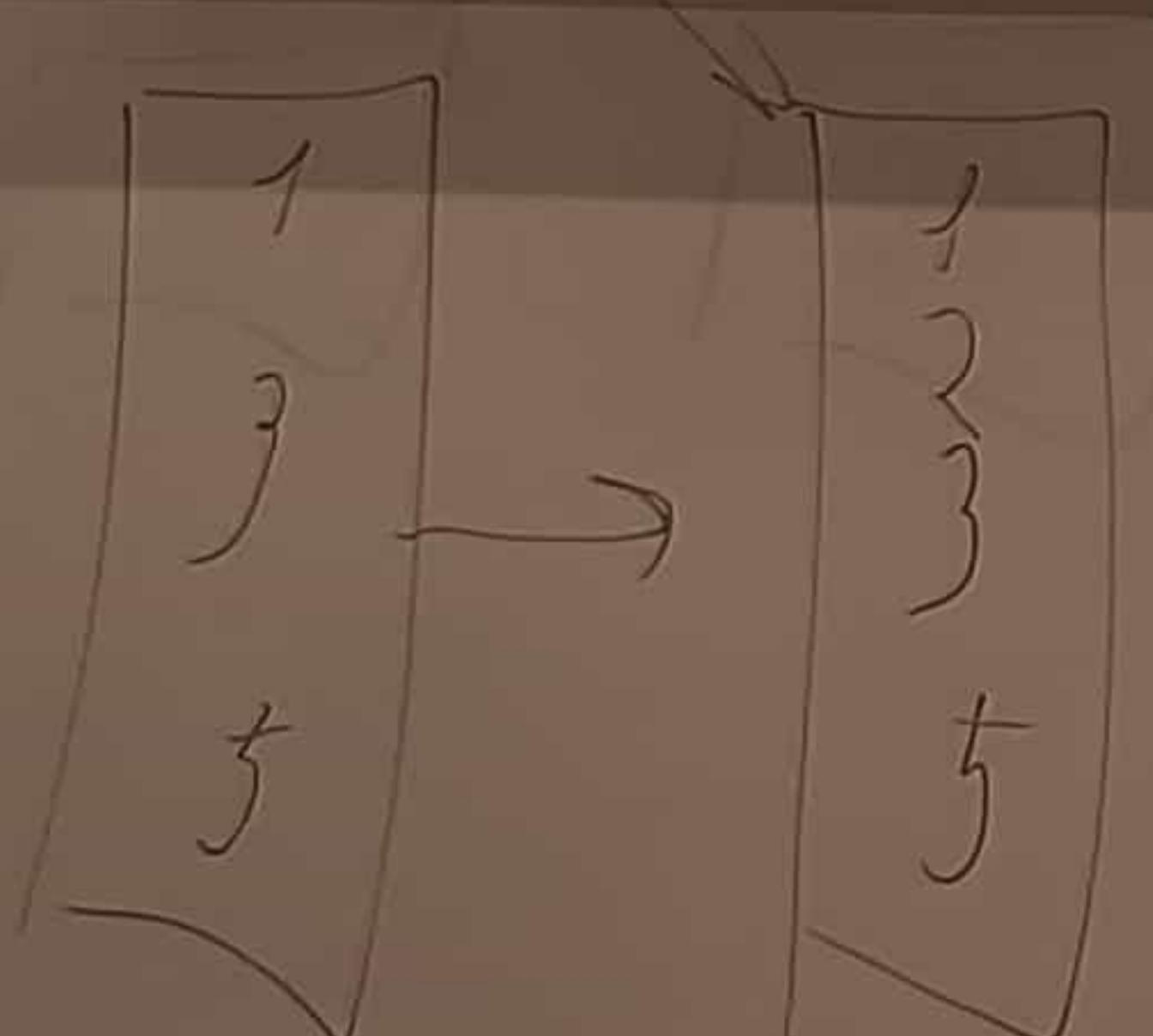
- Size-tiered: Объединение маленьких SS-таблиц в большие
- Leveled: Разбиение по уровням с постепенным перемещением
- Разные компромиссы между скоростью и использованием диска



В-Деревья

Опасность операций с множеством страниц

- Разбиение страницы при переполнении
- Запись двух новых страниц
- Обновление родительской страницы



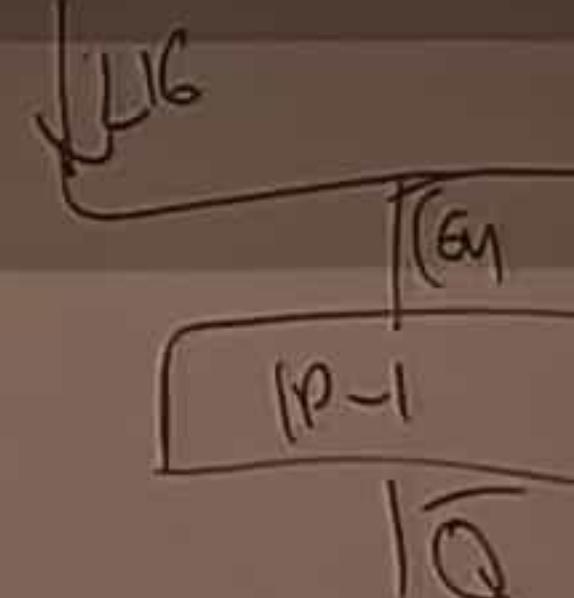
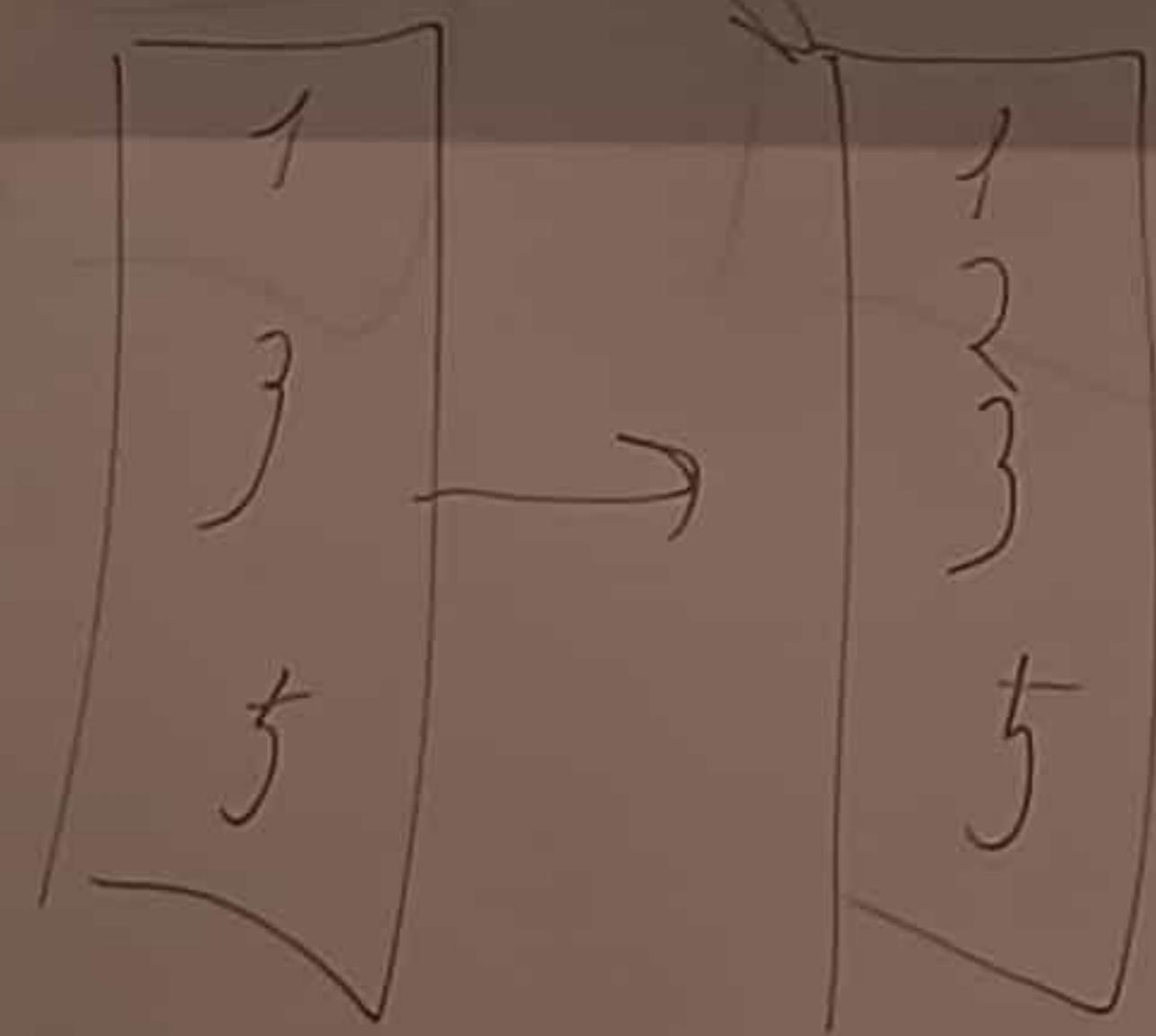
Notes
Merge Tree

БМУ [STABLE
MEMORY]

Write-Ahead Log

- Файл только для добавления (append-only)
- Быстрая последовательная запись
- Используется для восстановления после сбоев

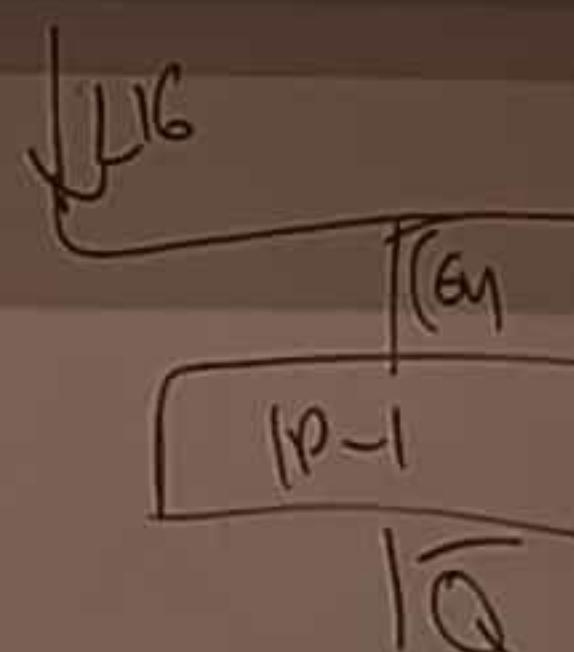
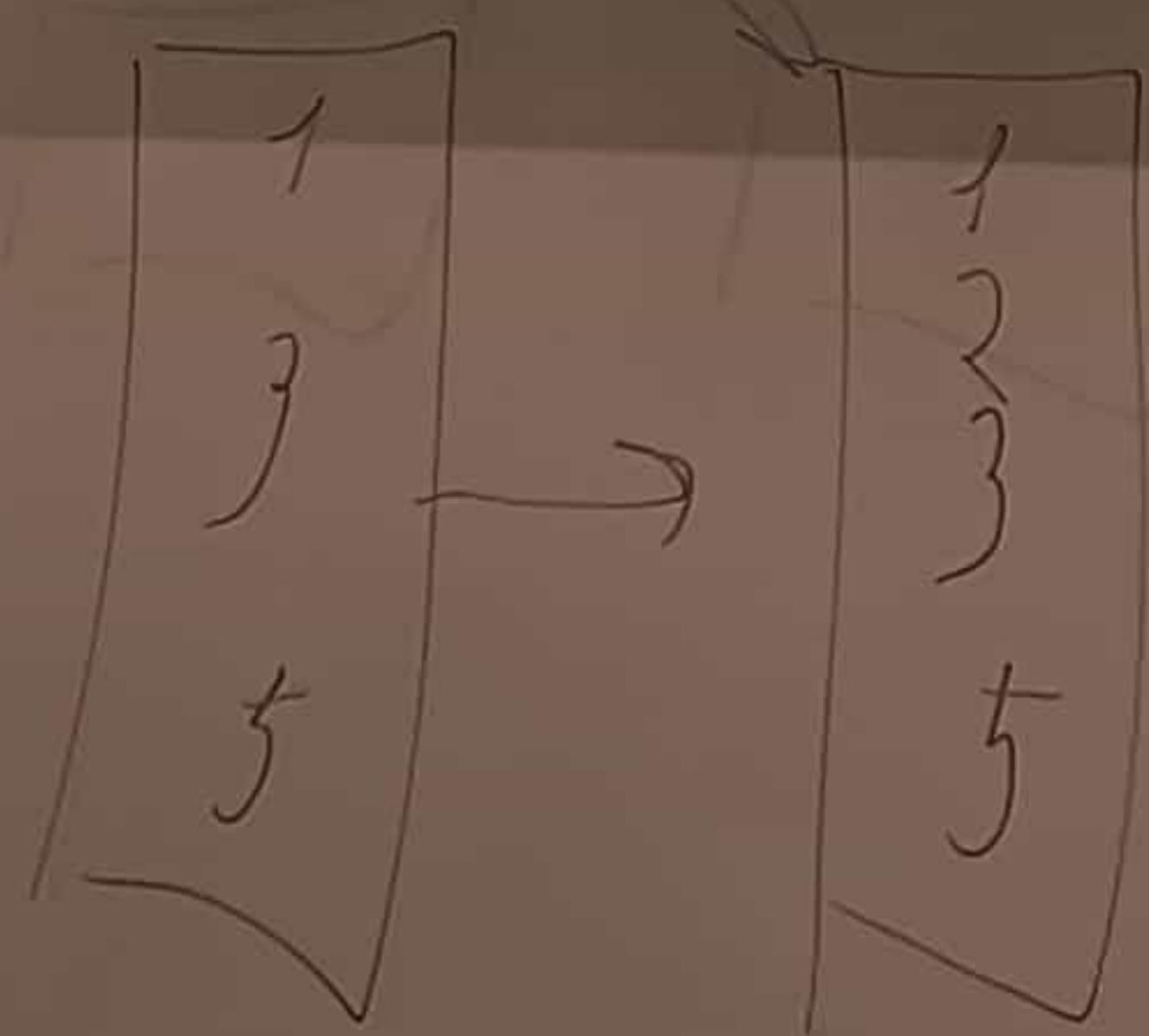
Index
Merge Tree
РПЛТ БМК SSTABLE
VAL



Защелки (latches)

- Облегченные версии блокировок
- Защита структур данных при одновременном доступе
- Высокая производительность

Notes
Merge Tree
таблица SSTABLE
стабильны



- LSM-деревья: Обычно быстрее при записи
- B-деревья: Обычно быстрее при чтении

Range Index
Tree-based Merge Tree

ФИЛТ БУК
WAL

