RocksDB에서 SST파일에 따른 WAF감소에 관한 연구

석사과정 조민수

Yonsei University
Data Engineering Lab

CONTENTS

NoSQL

RocksDB

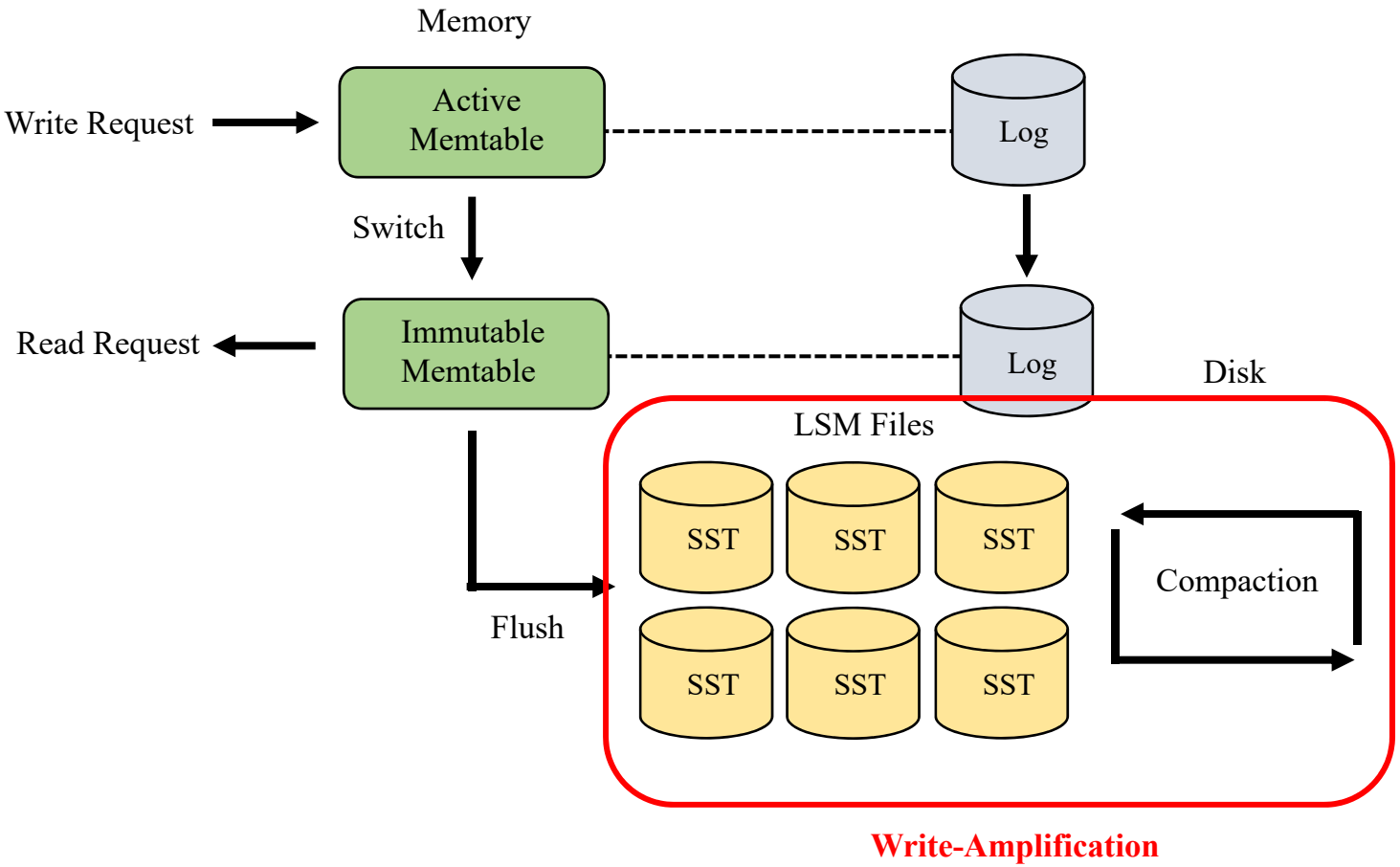
facebook

- RocksDB는 구글의 LevelDB 기반으로 개발된 Facebook사의 오픈소스 프로젝트임

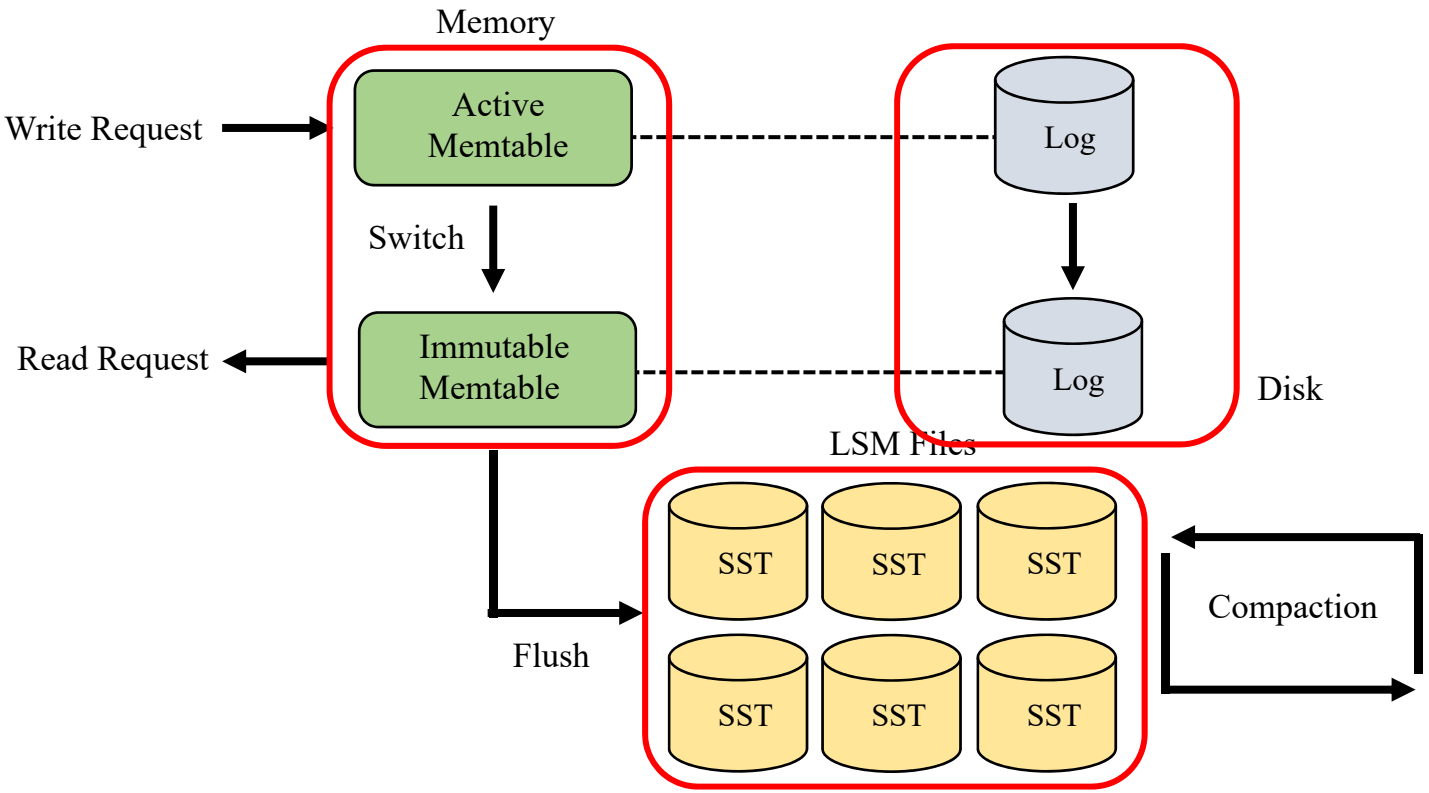
RocksDB Introduction

- LevelDB 기반의 오픈소스 프로젝트
- C++ 라이브러리 형태로 제공
- Key-value 저장 방식
- 메모리, 플래시, 하드디스크, HDFS 등 다양한 환경에서 실행 가능
- 다양한 압축 방식 지원(snappy, zlib, bzip2, lz4, lz4_hc)
- 세가지 기본 구조는 memtable, sstfile, logfile

RocksDB Architecture

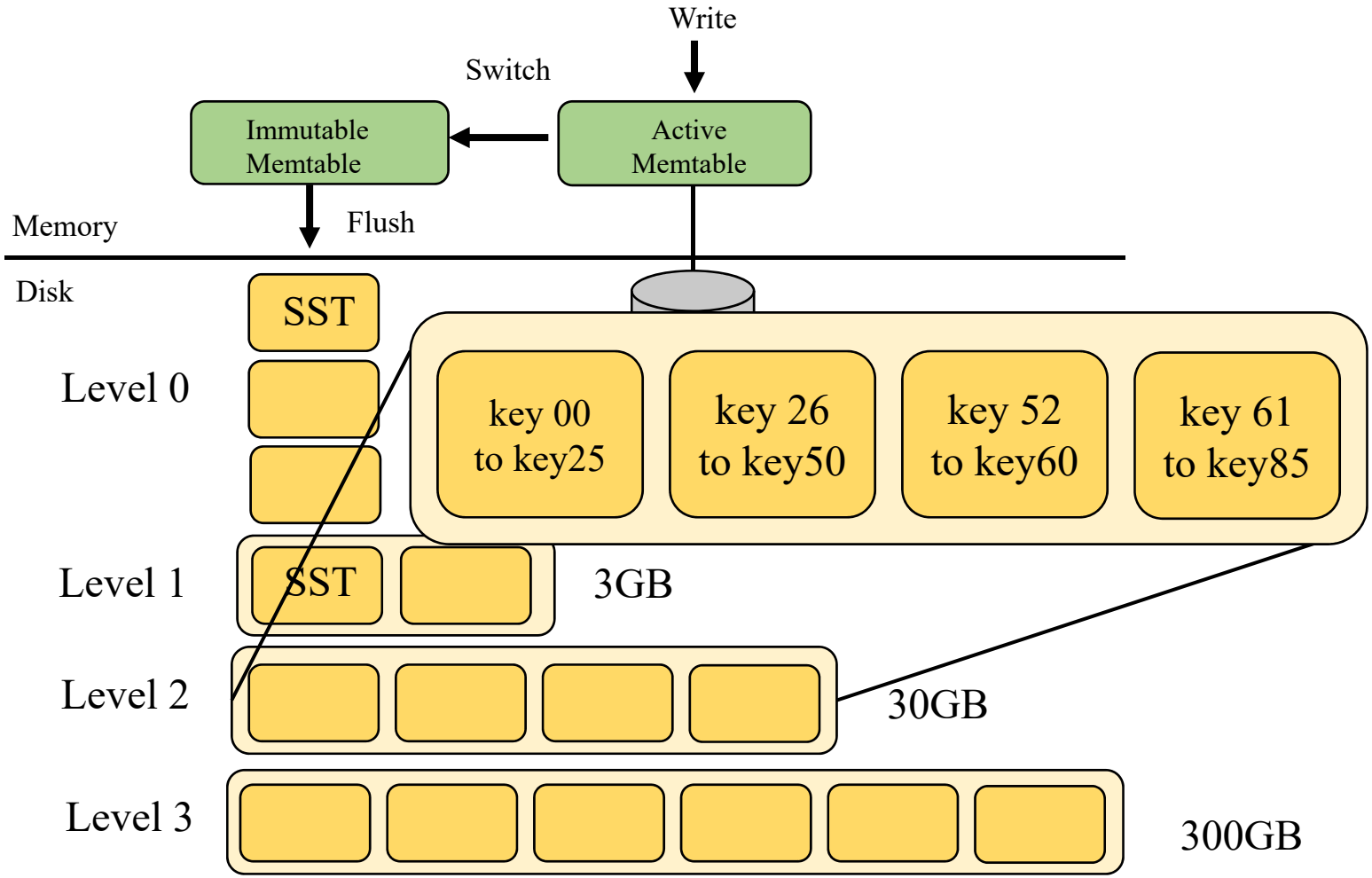


RocksDB Architecture

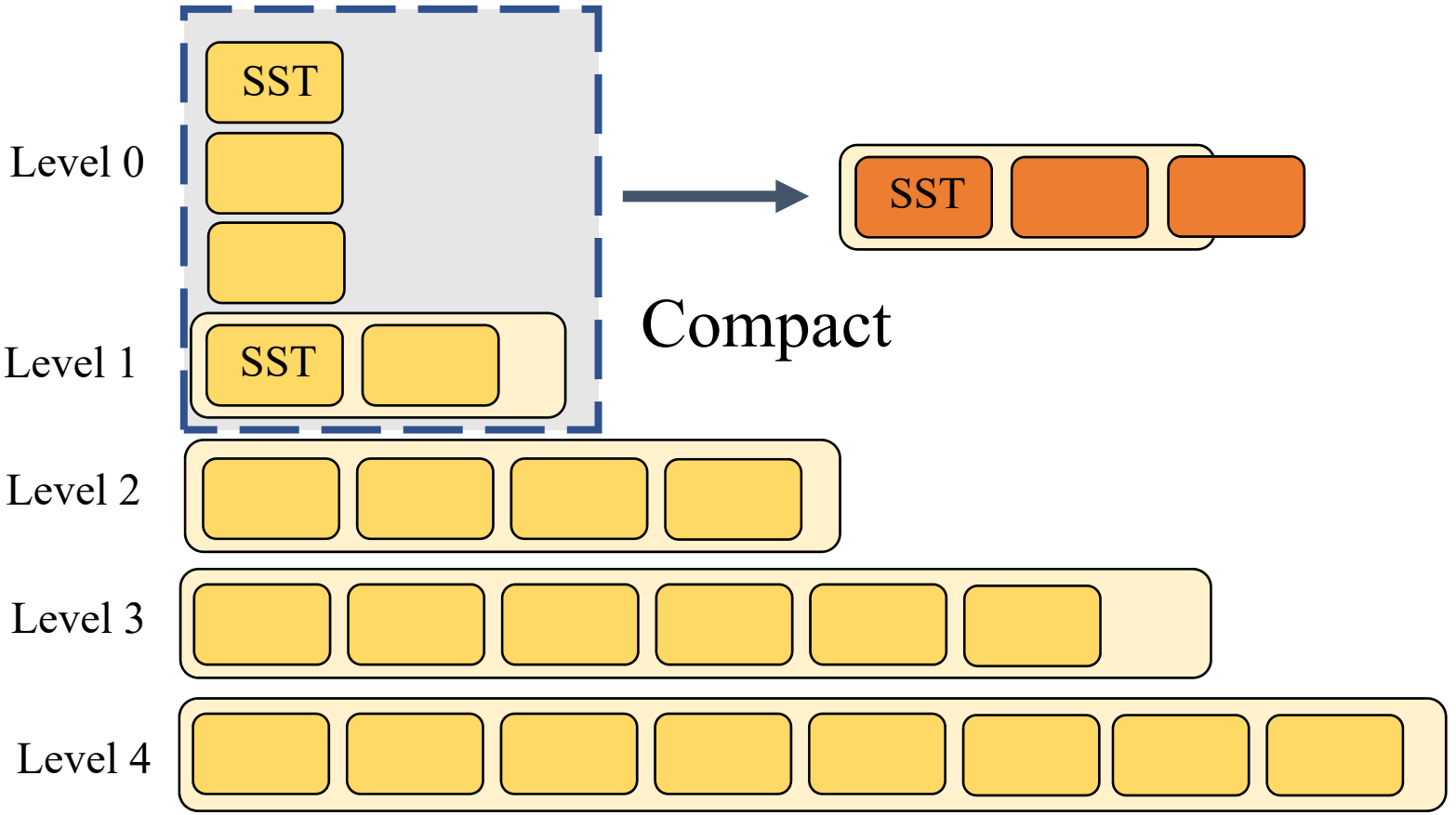


- RocksDB는 LSM tree 구조를 사용하며 크게 메모리, 로그, 디스크 영역으로 나뉨

RocksDB Architecture

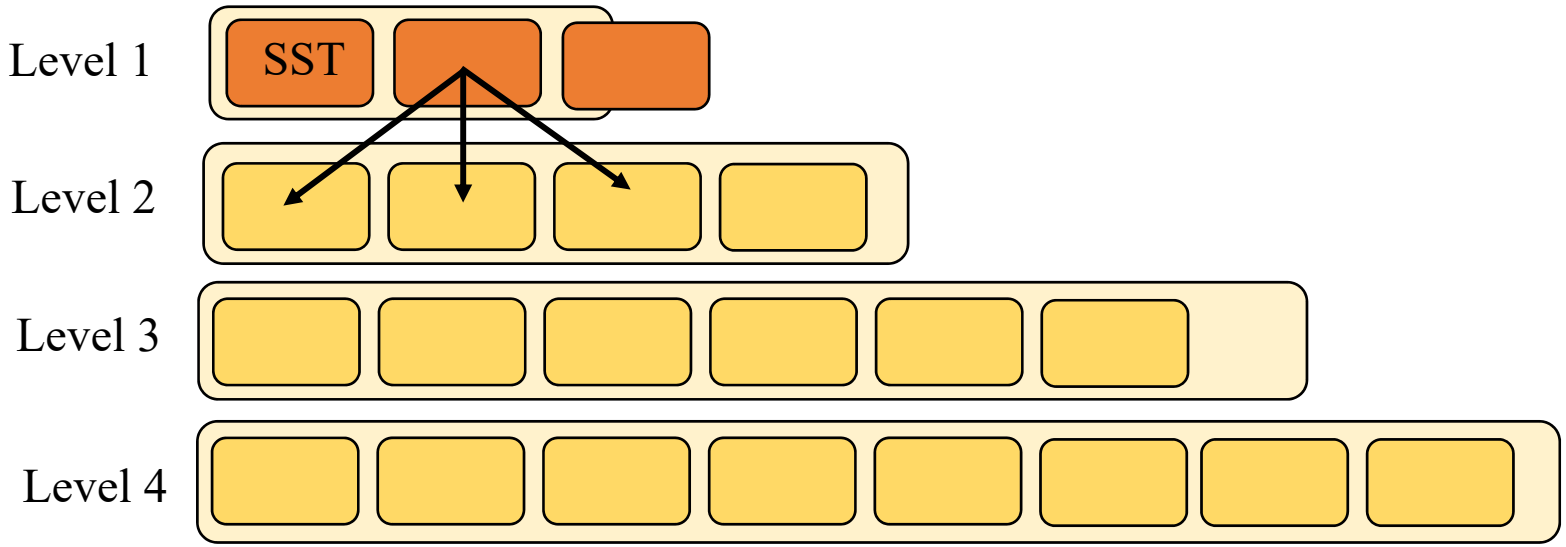


Compaction



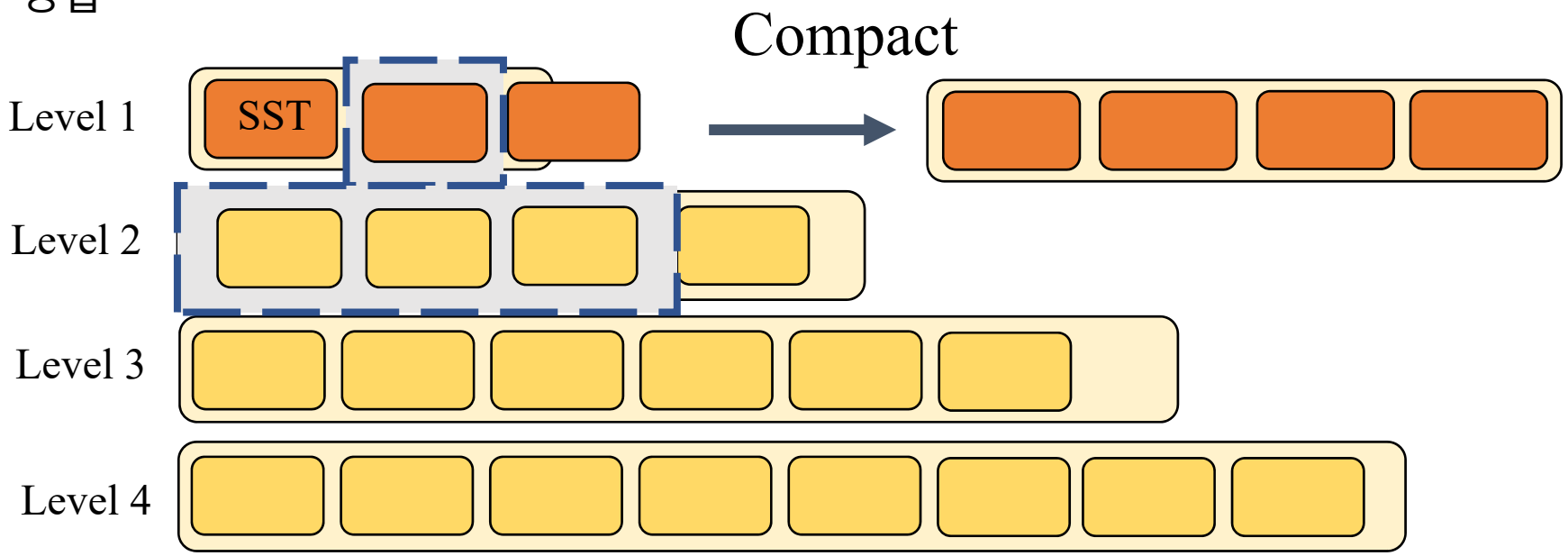
Compaction

- L1의 크기가 타겟 크기를 초과



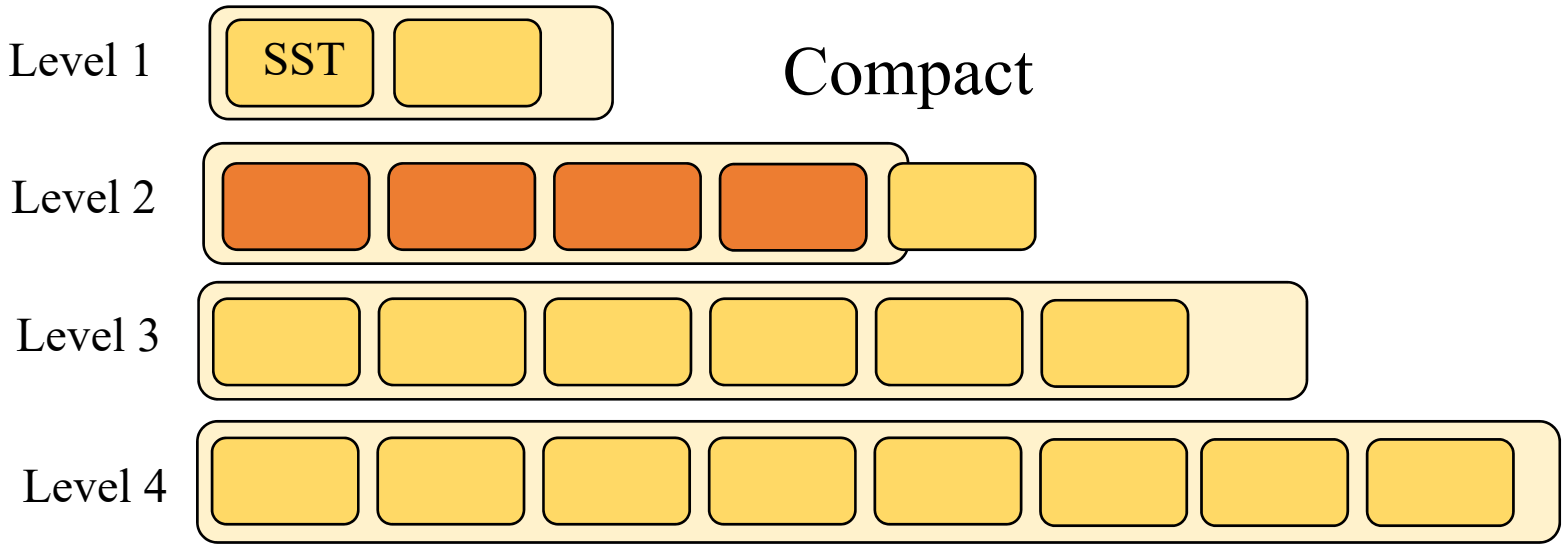
Compaction

- L1에서 하나의 파일을 선택하고, 그 파일과 중복되는 key범위를 갖는 L2파일과 병합



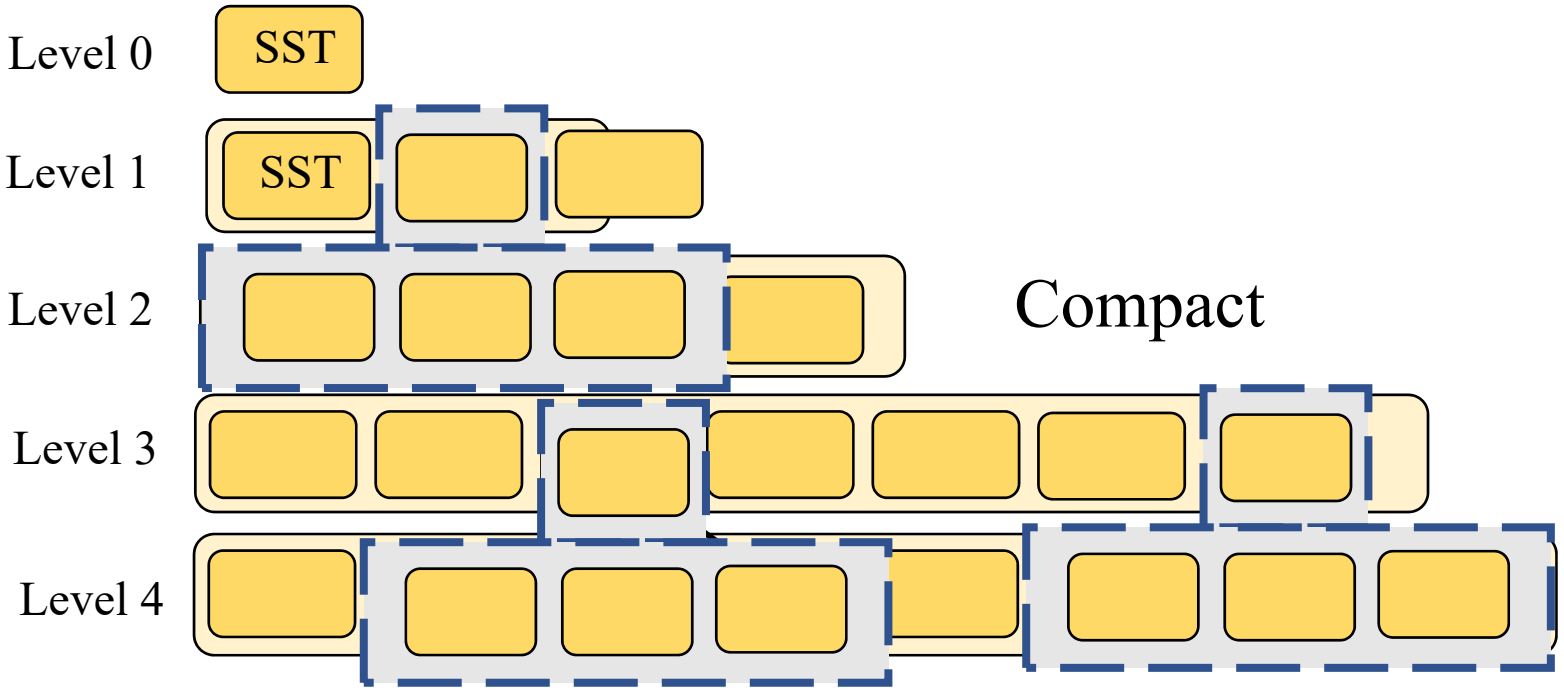
Compaction

- 이전과 동일하게 파일 선택 후 다음 단계로 병합



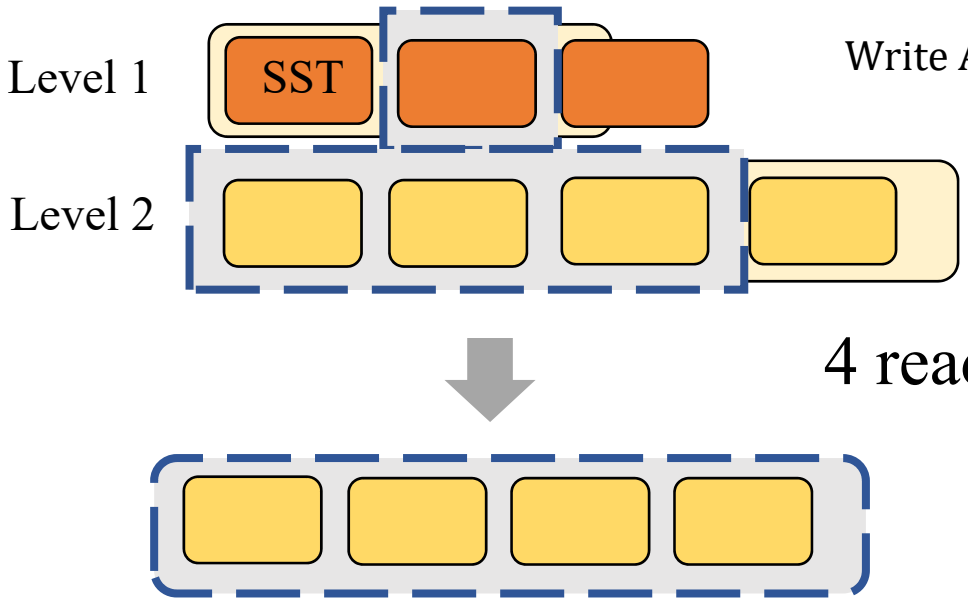
Compaction

- 필요한 경우, 여러 compaction을 동시에 실행 가능



Write Amplification

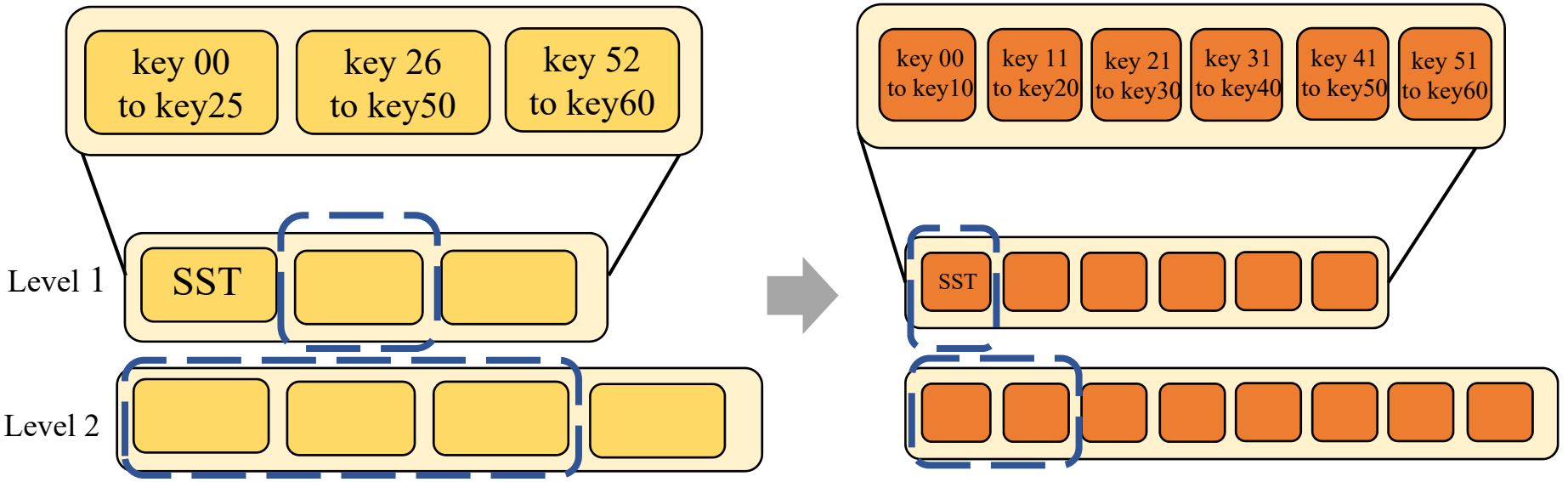
- 데이터를 스토리지에 쓰는 과정에 발생하는 쓰기 증폭
- 쓰기 증폭이 증가하면 디바이스의 입출력 성능이 감소



$$\text{Write Amplification} = \frac{\text{Bytes written to Storage}}{\text{Bytes written to Database}}$$

Write Amplification

- 컴팩션 과정에서 Write Amplification을 최소화하는 것이 중요
- SST File의 크기를 줄이는 방법 -> 하위레벨에서 중복되는 파일의 개수가 확률적으로

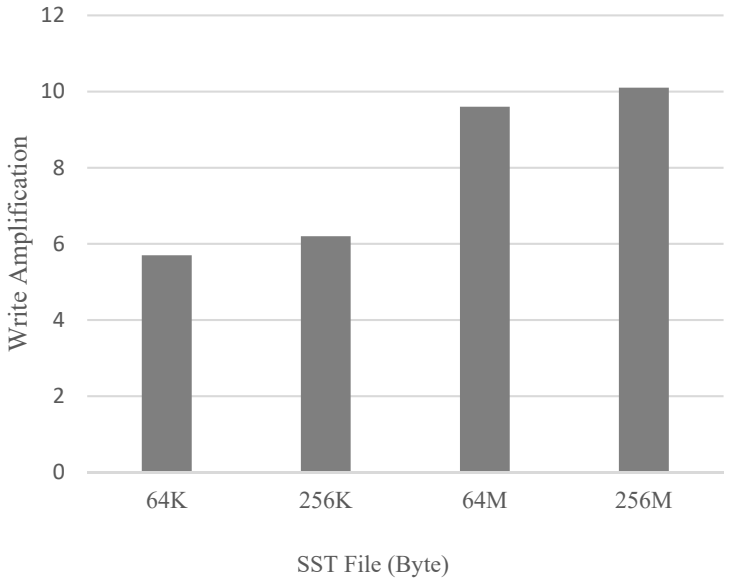


실험 환경

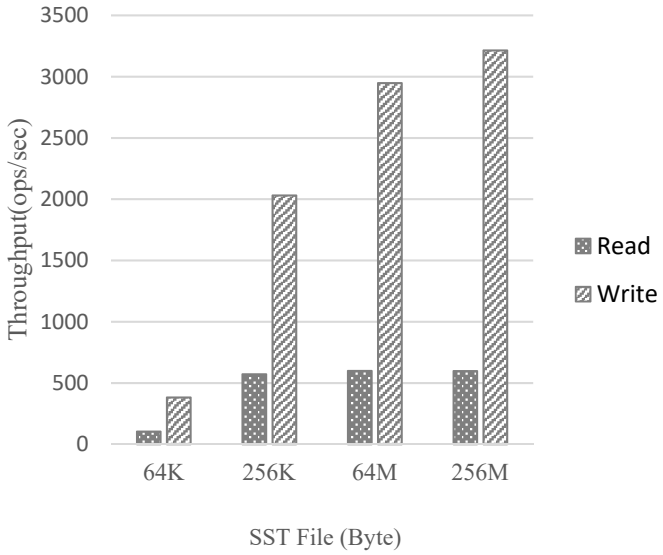
OS	CentOS 7.3.1611(x86_64)
CPU	Intel(R) Core(TM) i7-6700K CPU @ 4.00GHz
RAM	64GB
SSD	SAMSUNG 850 PRO 256GB

- DBbench 벤치마크 사용, 옵션을 통해 다양한 워크로드 생성
- Key-value 값을 각각 8192byte, 32byte로 설정
- 약 13GB의 데이터로 측정
- Direct I/O 방식
- SST 파일 크기를 Kilo, Mega byte 단위 사용

실험 결과



[그림 1] DBbench WAF 실험 결과



[그림 2] DBbench 성능 결과

결론

- SST파일 크기의 감소에 따른 쓰기 증폭 감소
- 쓰기 증폭이 줄었지만, 디바이스의 쓰기와 읽기 성능도 같이 떨어짐
- 쓰기 증폭과 성능 감소의 트레이드오프 고려
- 향후에는 더 다양한 크기의 SST파일과 Buffered I/O 방식에서 실험 예정
- 다양한 디바이스에서 실험(NVMe, 하드 디스크)

감사합니다