Noir :
메일검색 서버를 반의 반으로 줄여준 신규 검색엔진 제작기

이창현
NAVER Search
hyun.lch@navercorp.com

신우진
NAVER Search
woojin.shin@navercorp.com

NAVER DEVIEW 2023

0. OVERVIEW

개별 데이터 검색서비스 특화 검색엔진

- ✓ 역색인을 사용하지 않는 새로운 컨셉의 검색엔진
- ✓ 메일 검색에 적용한 결과 서버 수 1/8로 절감
- ✓ 현재 메일 검색, 네이버웍스 메시지 검색에 적용됨

CONTENTS

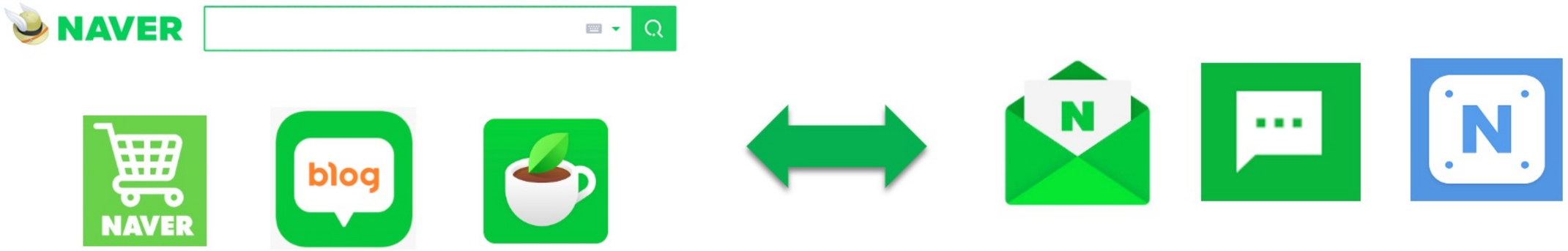
NAVER
DEVIEW
2023

1. 개별데이터 검색서비스
2. Full Scan 검색
3. Noir
4. 설계 및 구현
5. 고도화 계획

1. 개별 데이터 검색서비스

1.1 개별 데이터 검색 서비스

개별 데이터를 다루는 서비스의 검색



통합검색류 검색서비스

검색 대상은 서비스 내 전체 문서

개별 데이터 검색서비스

검색 대상은 전체 문서 중 일부분

1.1 개별 데이터 검색 서비스

개별 데이터를 다루는 서비스의 검색

내가 받은 메일

내가 속한 채팅룸 대화

내 파일

1.2 메일 검색 서비스

네이버메일 검색 서비스

수천만 계정 서빙

일일 문서 유입 수억 건

검색 요청(일일 수백만) 에 비해 문서유입량이 매우 많음

통합검색 일간 쿼리량(수억 건)에 맞먹는 문서 유입량

1.2 메일 검색 서비스

네이버메일 검색 서비스

수천만 계정 서빙

일일 문서 유입 수억 건

검색 요청(일일 수백만) 에 비해 문서유입량이 매우 많음

통합검색 일간 쿼리량(수억 건)에 맞먹는 문서 유입량

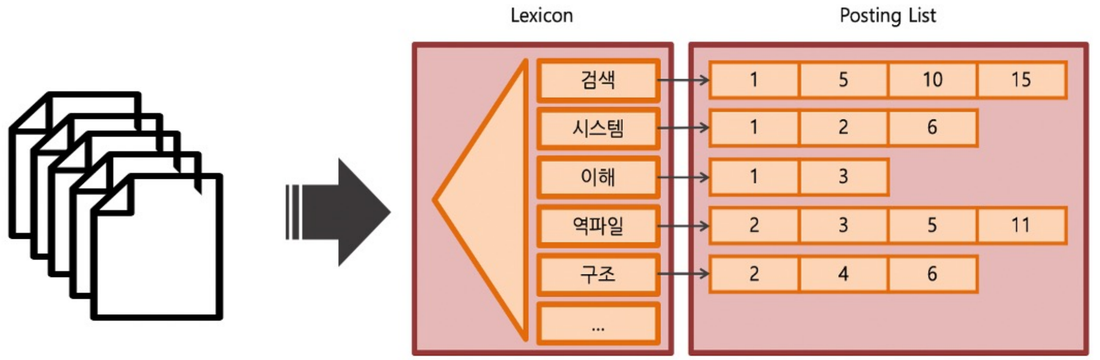
네이버메일 검색서비스는 어떻게 서빙해야 할까?

1.3 단일 역색인 구조와 메일 검색

NAVER DEVIEW 2023

단일 역색인 볼륨 방식

- 가장 일반적인 방법
- 단일 역색인 볼륨으로 모든 문서 서빙
- 적절히 쿼리하여 특정 사용자의 메일 검색
- 문서가 많지 않으면 가능한 방식



메일검색 전체문서

단일 역색인 볼륨

1.3 단일 역색인 구조와 메일 검색

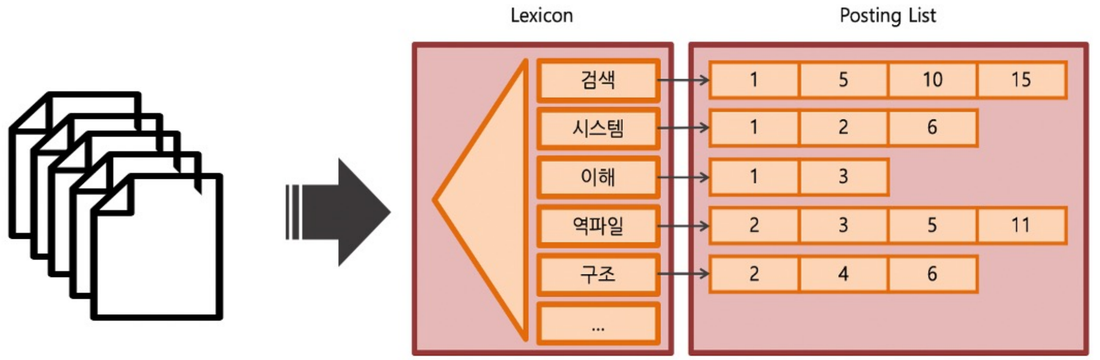
NAVER DEVIEW 2023

단일 역색인 볼륨의 어려움

1. 단 한명의 메일을 검색하기 위해 전체문서 검색

2. 단 한명의 신규메일로 전체볼륨 업데이트 부하

3. Locality가 떨어져 제한된 메모리에서 검색성능을 만족하기 어려움



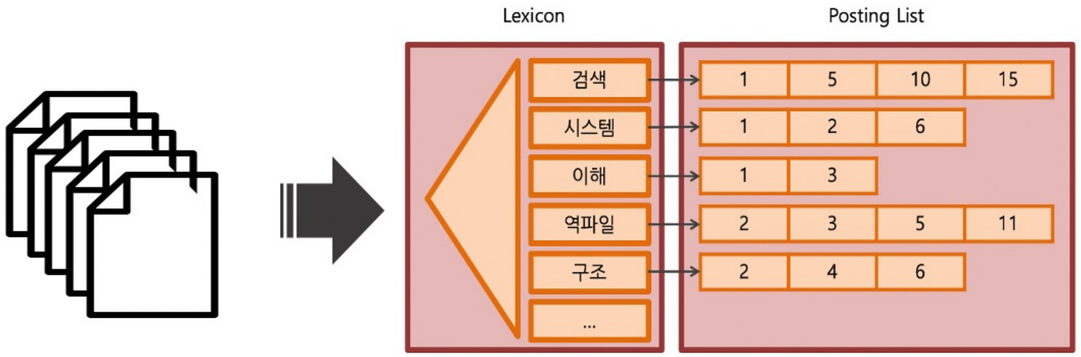
메일검색 전체문서

단일 역색인 볼륨

1.4 개별 역색인 구조와 메일 검색

NAVER DEVIEW 2023

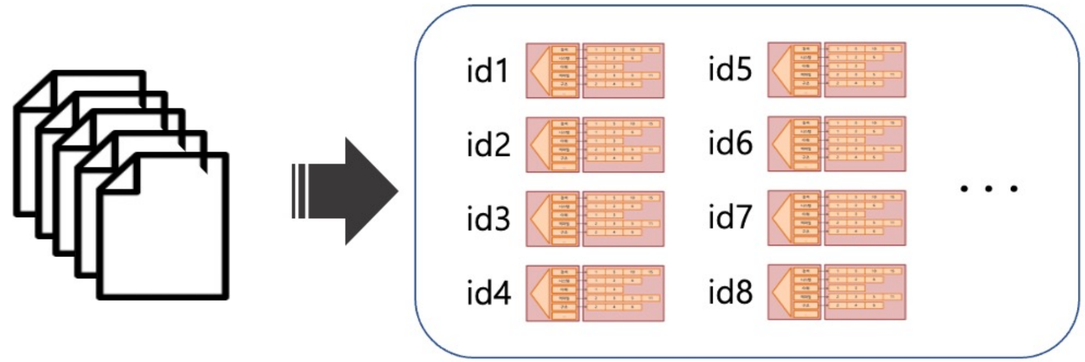
단일 역색인 볼륨



메일검색 전체문서

단일 역색인 볼륨

개별 역색인 볼륨



메일검색 전체문서

개별 역색인 볼륨

1.4 개별 역색인 구조와 메일 검색

계정 단위 볼륨 서빙 시

1. 단 한명의 메일을 검색하기 위해 전체문서 검색

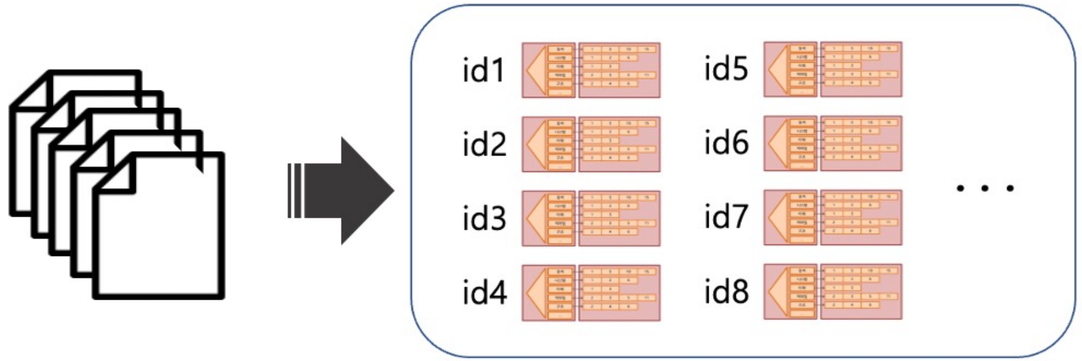
   → 해당 사용자 문서만 검색

2. 단 한명의 신규메일로 전체볼륨 업데이트 부하

   → 해당 사용자 볼륨만 업데이트

3. Locality가 떨어져 제한된 메모리에서 검색성능을 만족하기 어려움

   → 검색대상 문서가 물리적으로 모이므로 성능 개선



메일검색 전체문서

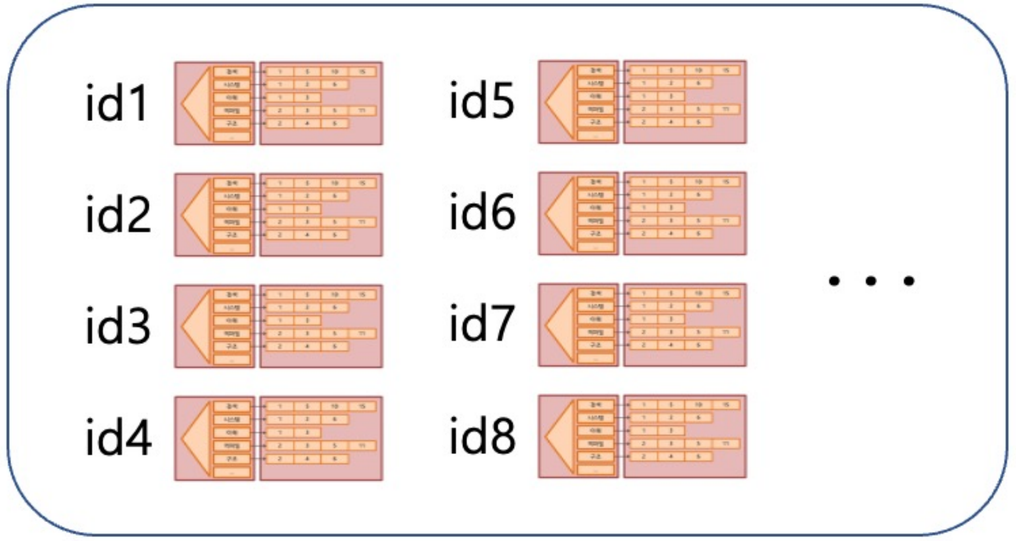
개별 역색인 볼륨

1.4 개별 역색인 구조와 메일 검색

개별 역색인 볼륨 방식 (2013~Noir이전)

계정단위 역색인 볼륨 관리

검색 시 동적으로 해당 볼륨을 메모리에 올려 검색



: 특정 계정의 역색인 볼륨

장점

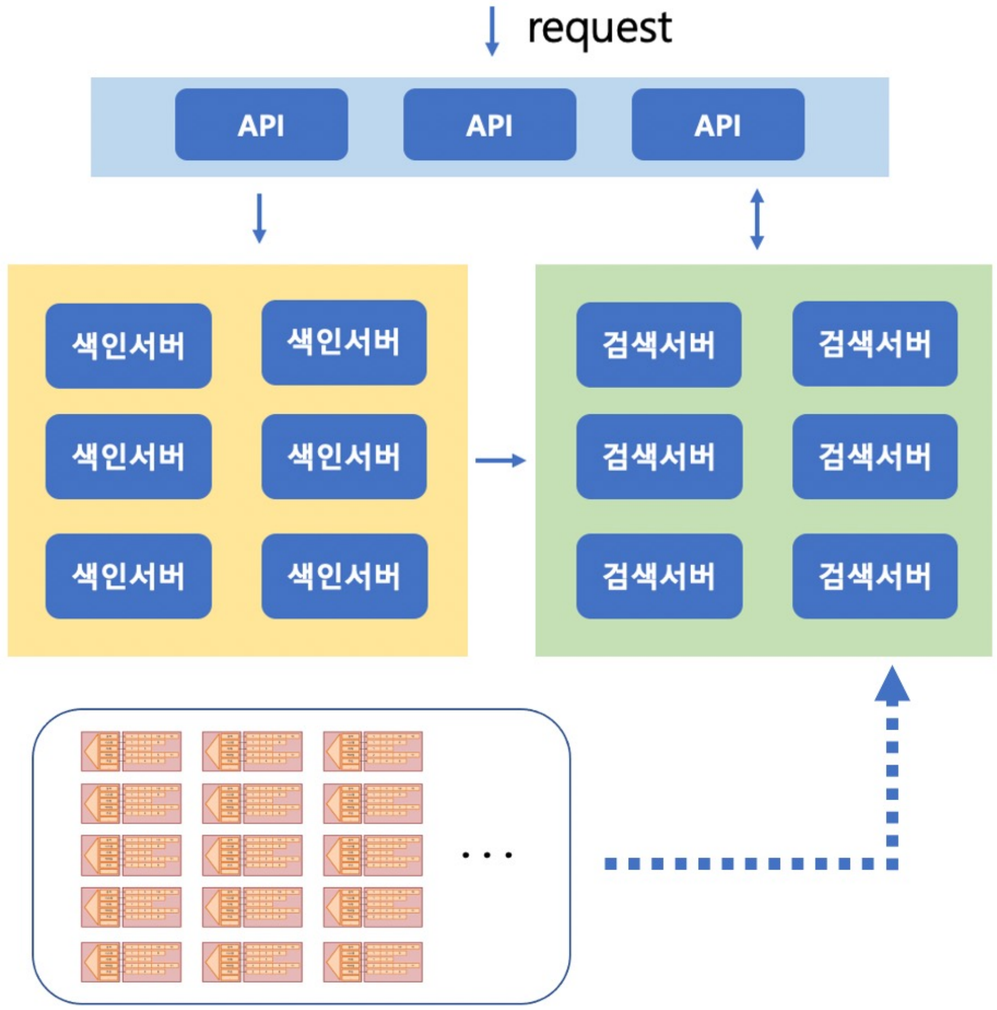
- 역색인을 메모리에 상주시킬 필요 없음
- 메모리 기반에서 disk 기반으로, 비용 감소
- 일부 문서set에 대해 작업하므로 문서 업데이트 및 검색 비용 감소

1.4 개별 역색인 구조와 메일 검색

NAVER DEVIEW 2023

개별 역색인 볼륨 방식 (2013~Noir이전)

- 수천만 이상의 개별 볼륨 서빙
- 각 검색서버마다 다수의 볼륨 저장
- 검색요청 시 적절한 검색서버를 찾아 검색
- 색인 서버에서 볼륨생성 후 검색서버로 배포



1.4 개별 역색인 구조와 메일 검색

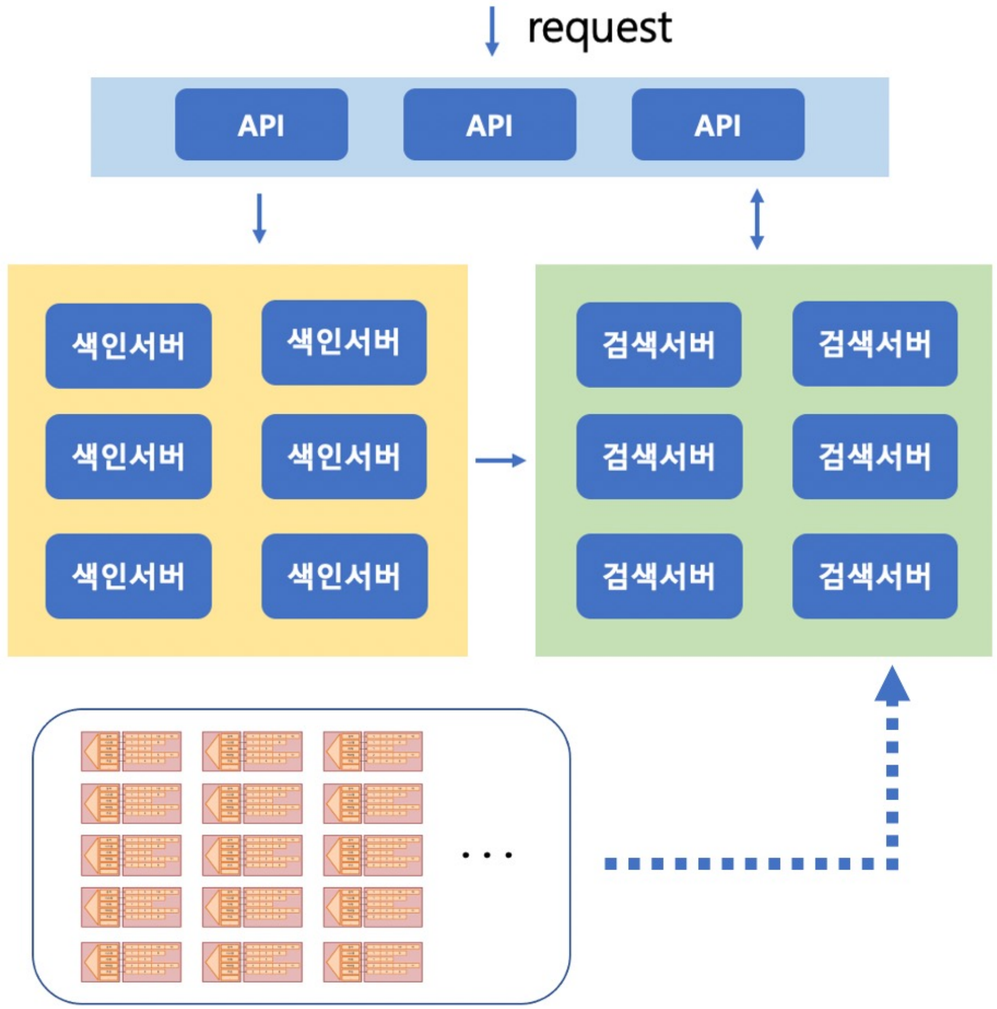
개별 역색인 볼륨 방식의 어려움

1. 여전히 문서 유입량만큼 가파른 disk수요 증가

→ 수백대 규모 서비스, 매달 서버 n대 증설

2. 여전히 색인 업데이트 비용부담이 높음

→ 검색서버만큼 별도의 색인서버 필요



1.4 개별 역색인 구조와 메일 검색

NAVER DEVIEW 2023

개별 역색인 볼륨 방식의 어려움

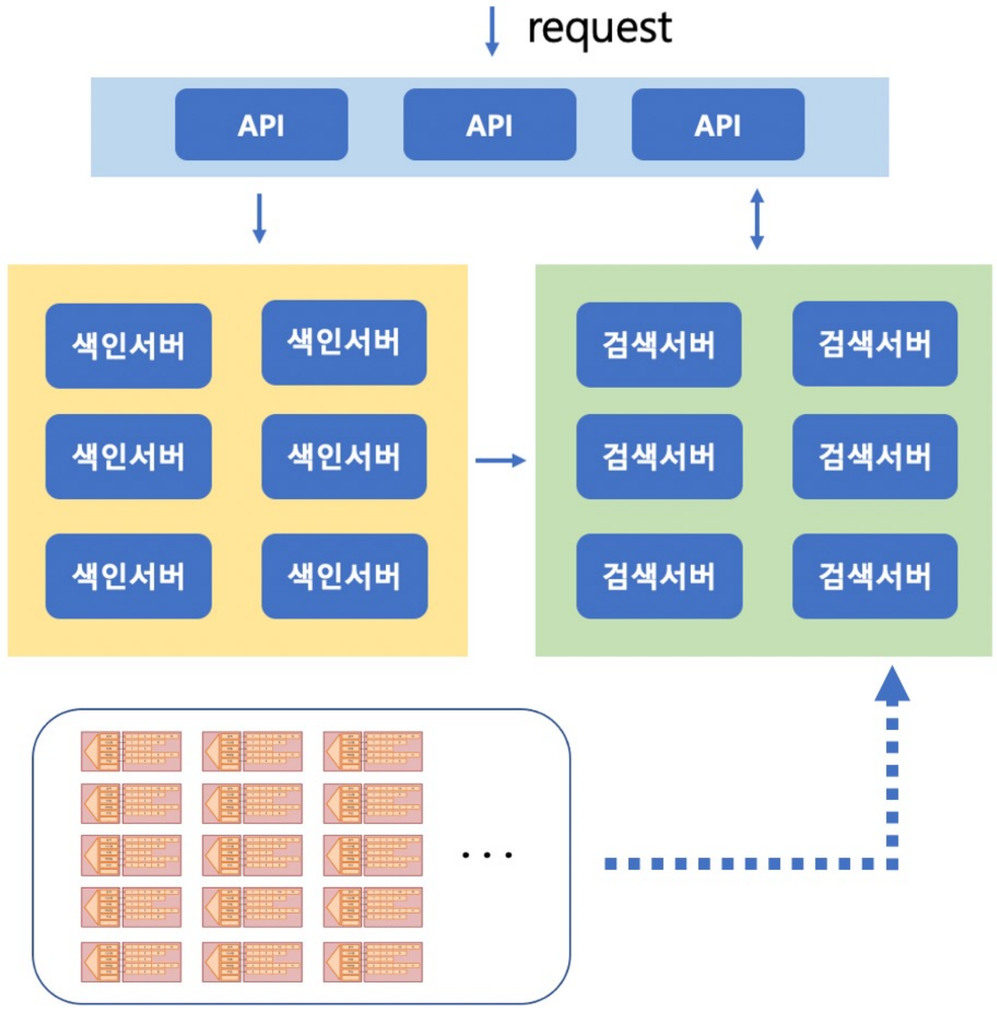
3. 평생 검색후보가 되지 않을 문서도 색인 필요

4. 큰 볼륨 검색 시 응답시간 지연

→ 색인볼륨을 read하는 IO시간 이슈

→ 어떤 볼륨은 검색 시 늘 timeout발생

5. 수많은 색인볼륨 관리 운영이슈 다수 발생



1.5 네이버메일 검색 서비스 이슈

요약

- 일일 문서 수억 건 유입으로 서버비용이 가파르게 상승 (Disk 수요, 검색서버)
- 문서 증분 비용이 비쌈 (CPU 수요, 색인서버)
- 큰 볼륨 검색 시 응답속도가 너무 느림
- 수천만 역색인을 다루는 운영상의 어려움

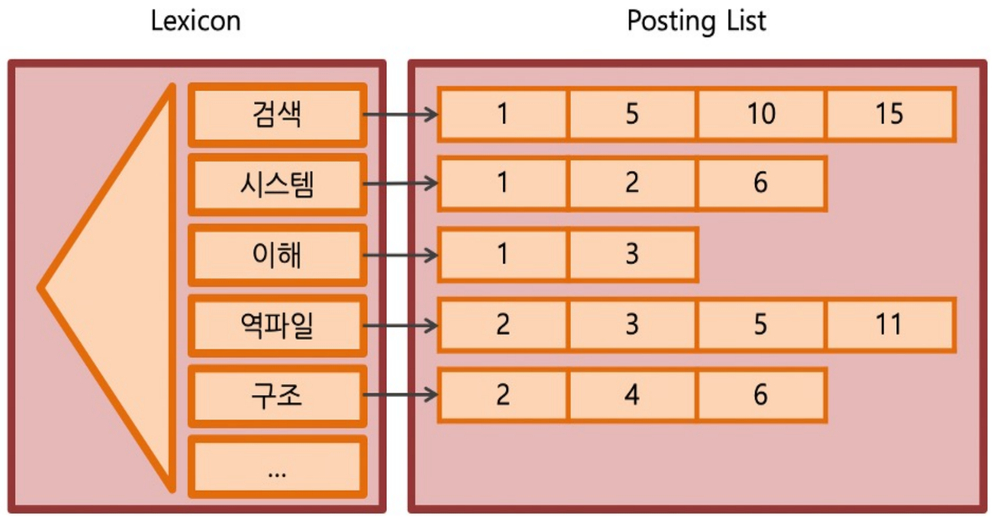
2. Full scan 검색

2.1 역색인

NAVER
DEVIEW
2023

특징

검색 응답속도에 최적화된 자료구조



Tradeoff

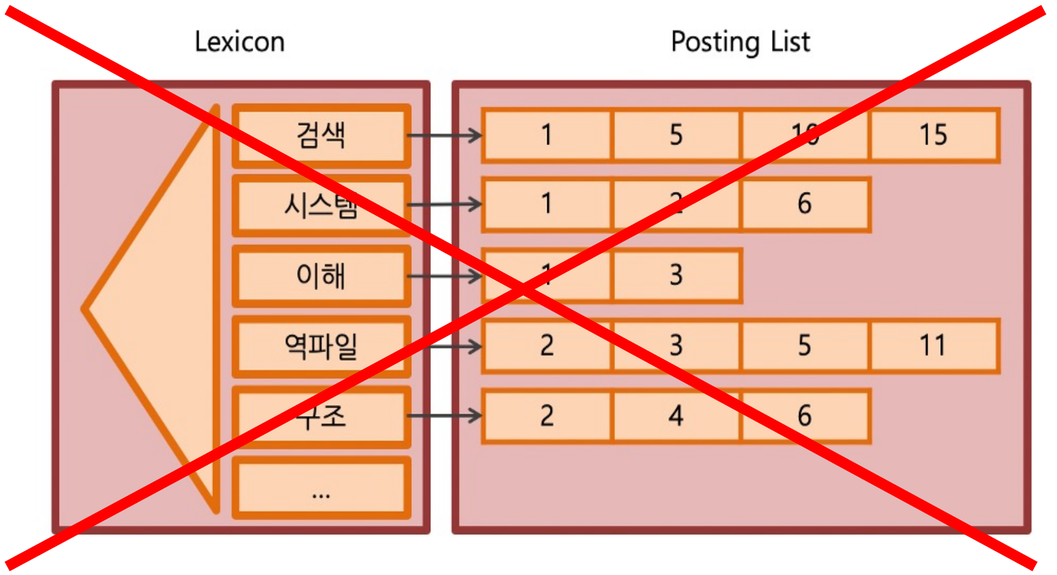
- 장점: 검색 응답시간 ↓
- 단점: 검색전용 역색인 자료구조 추가
- 단점: 볼륨생성 & 신규문서 증분 비용 ↑

2.1 역색인

반드시 역색인이 필요할까?

검색 대상은 극히 작은 일부 문서

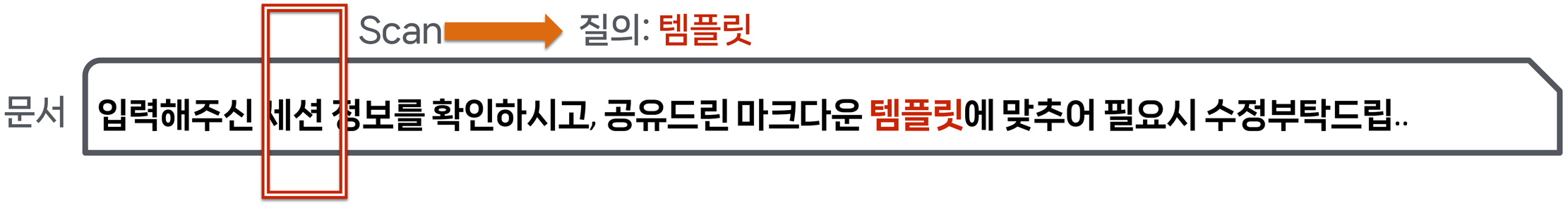
Tradeoff를 감수할 필요가 있을까



2.2 Full scan Search

NAVER DEVIEW 2023

역색인을 대체하는 Full scan 검색방식

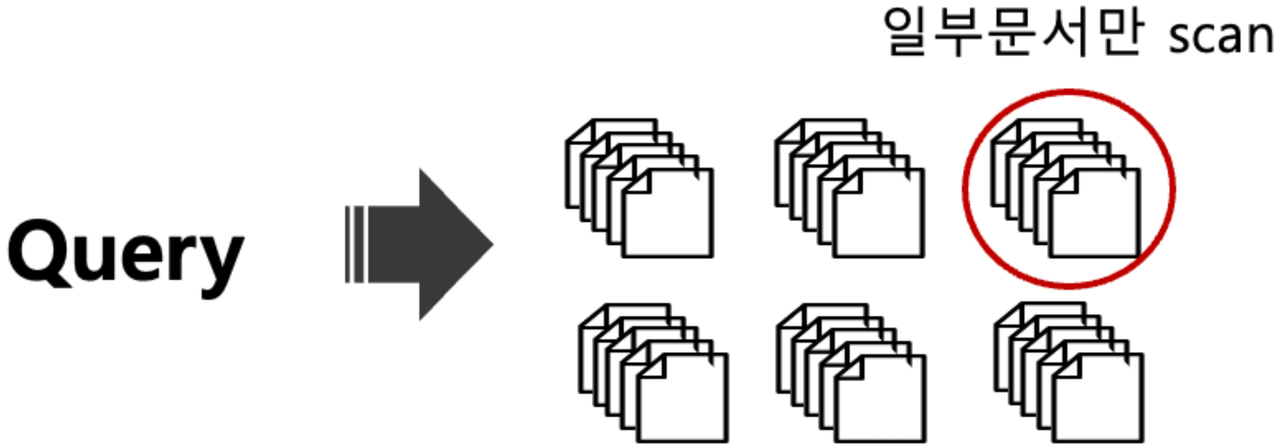


문서 원본 scan

문서가 쿼리와 부분일치할 시 검색됨

2.2 Full scan Search

역색인을 대체하는 Full scan 검색방식



검색 대상 일부문서만 scan

ex) 특정 사용자의 메일

2.2 Full scan Search

예시

Query: 이슈

문서1: [search-engine/weekly-updates] 주간이슈공유 20210402 → O
문서2: [변경작업 안내][네이버 메일 검색] 전체볼륨 Noir 전환 → X
문서3: ... 마지막에 공유드린 이슈 관련 자료입니다 → O

검색 결과: 문서1, 문서3

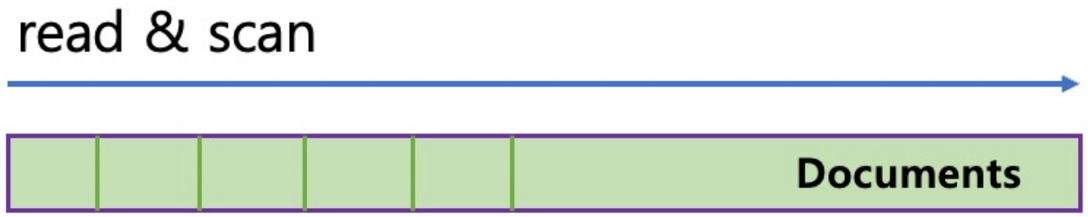
2.3 Full scan Search 성능

NAVER DEVIEW 2023

단순 Full scan 검색

계정 단위로 문서를 모아 볼륨생성

문서를 순차적으로 read 후 scan

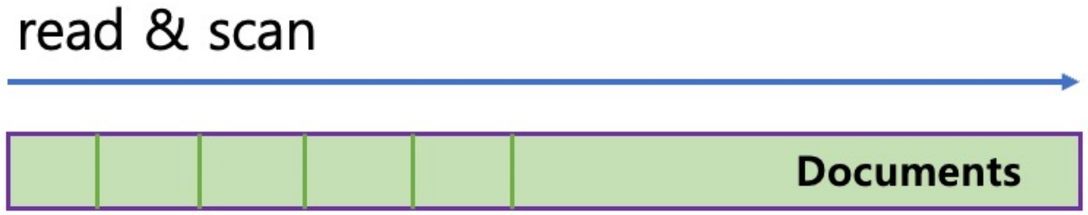


2.3 Full scan Search 성능

단순 Full scan 검색

계정 단위로 문서를 모아 볼륨생성

문서를 순차적으로 read 후 scan

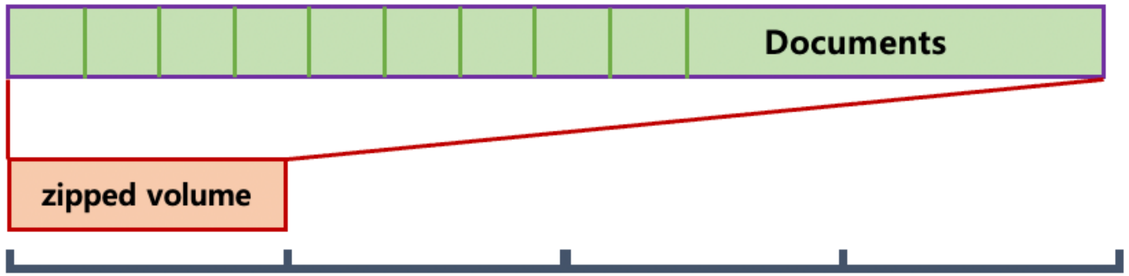


! 응답속도 너무 느림 (500MB/s SSD 장비에서 300MB이상 문서 검색시 1sec 초과)

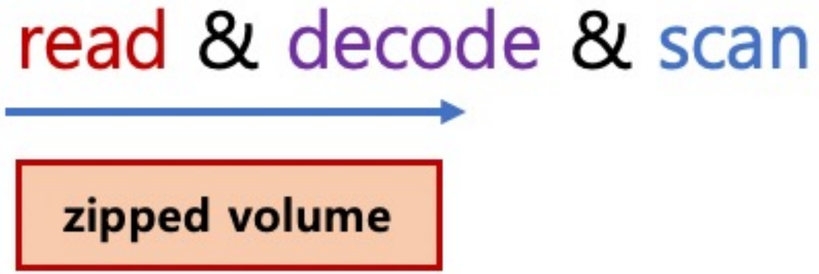
2.3 Full scan Search 성능

압축문서 Full scan 검색

전체 문서를 압축 후 저장



압축블록을 순차적으로 훑으며 검색

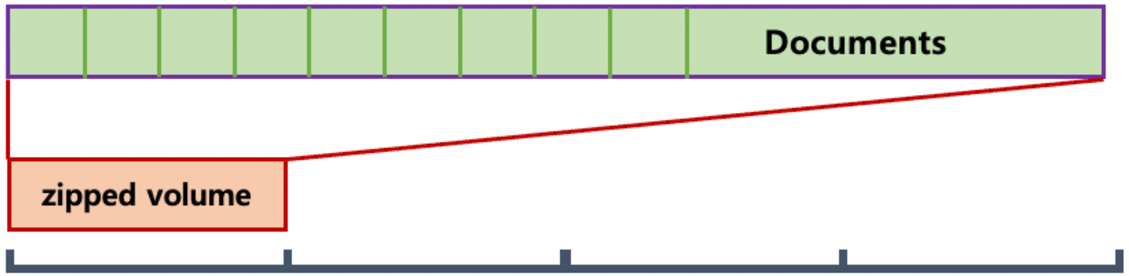


2.3 Full scan Search 성능

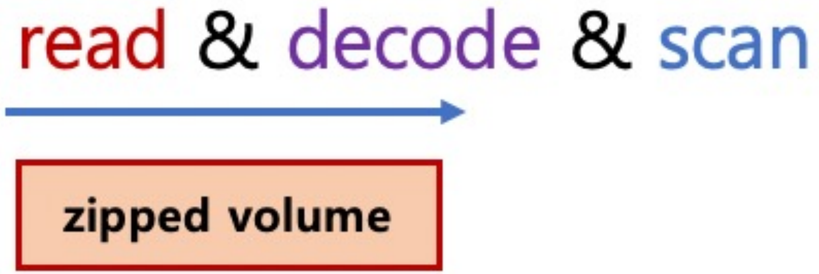
NAVER DEVIEW 2023

압축문서 Full scan 검색

전체 문서를 압축 후 저장



압축블록을 순차적으로 훑으며 검색



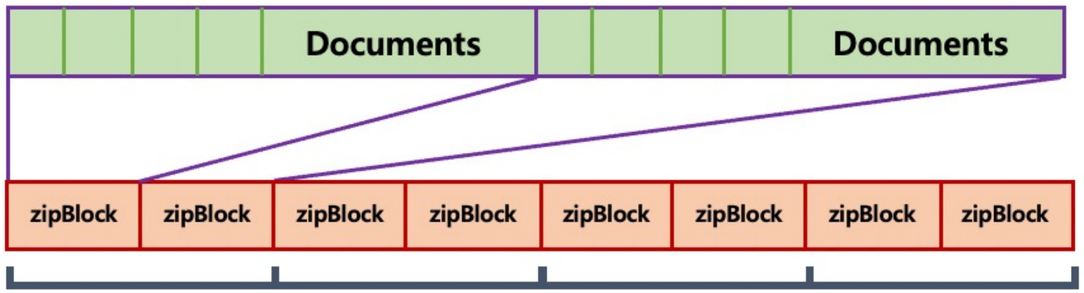
! read시간이 감소한 만큼 압축해제 시간 증가

2.3 Full scan Search 성능

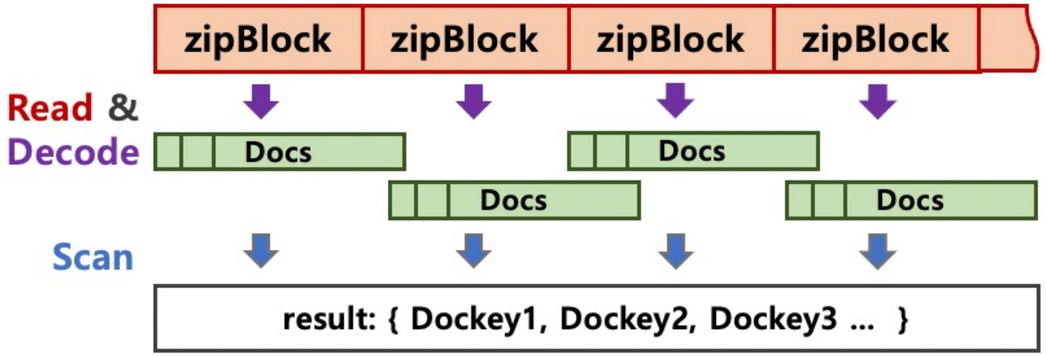
NAVER DEVIEW 2023

블록 단위 압축 Full scan 검색

문서를 일정 크기로 모아 블록 단위 압축



각 압축블록을 병렬적으로 Scan

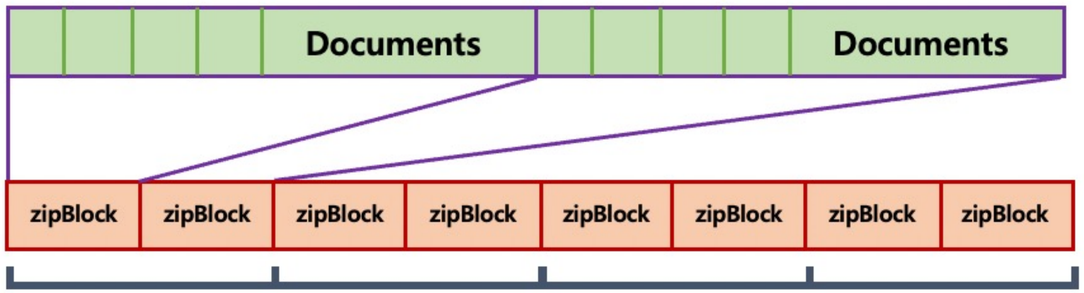


2.3 Full scan Search 성능

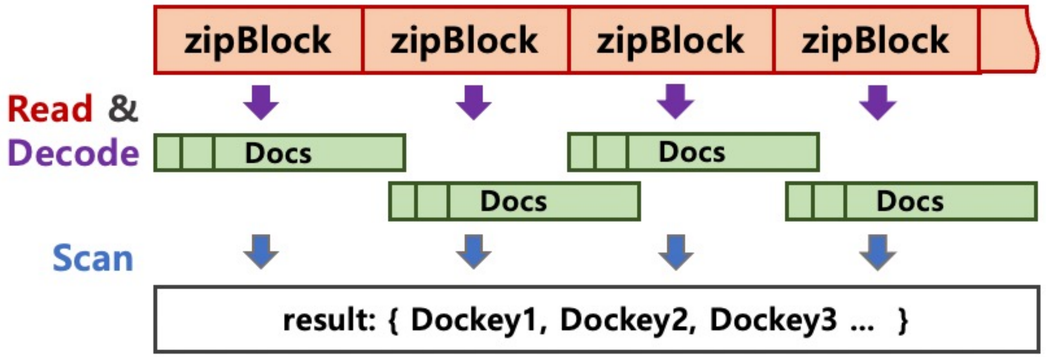
NAVER DEVIEW 2023

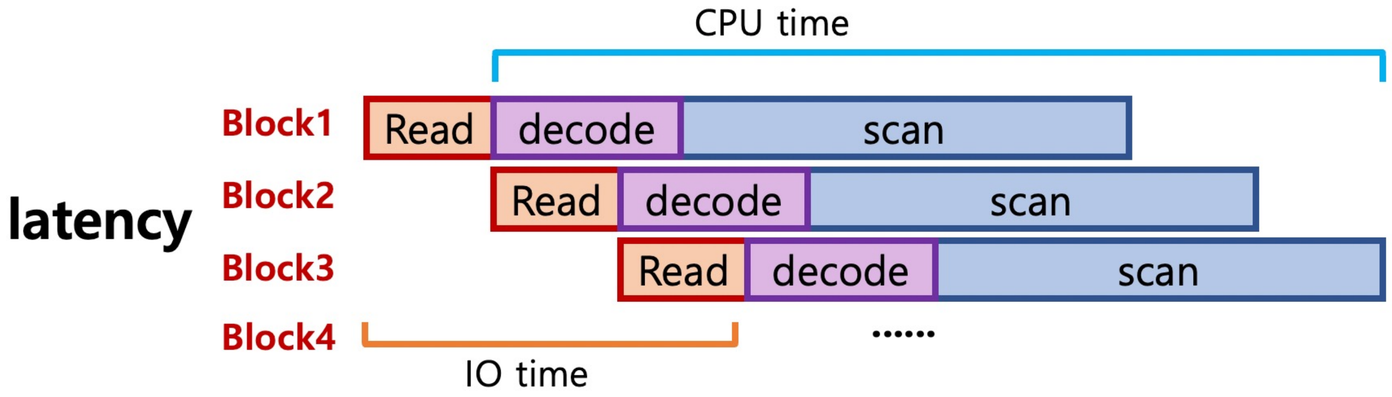
블록 단위 압축 Full scan 검색

문서를 일정 크기로 모아 블록 단위 압축



각 압축블록을 병렬적으로 Scan

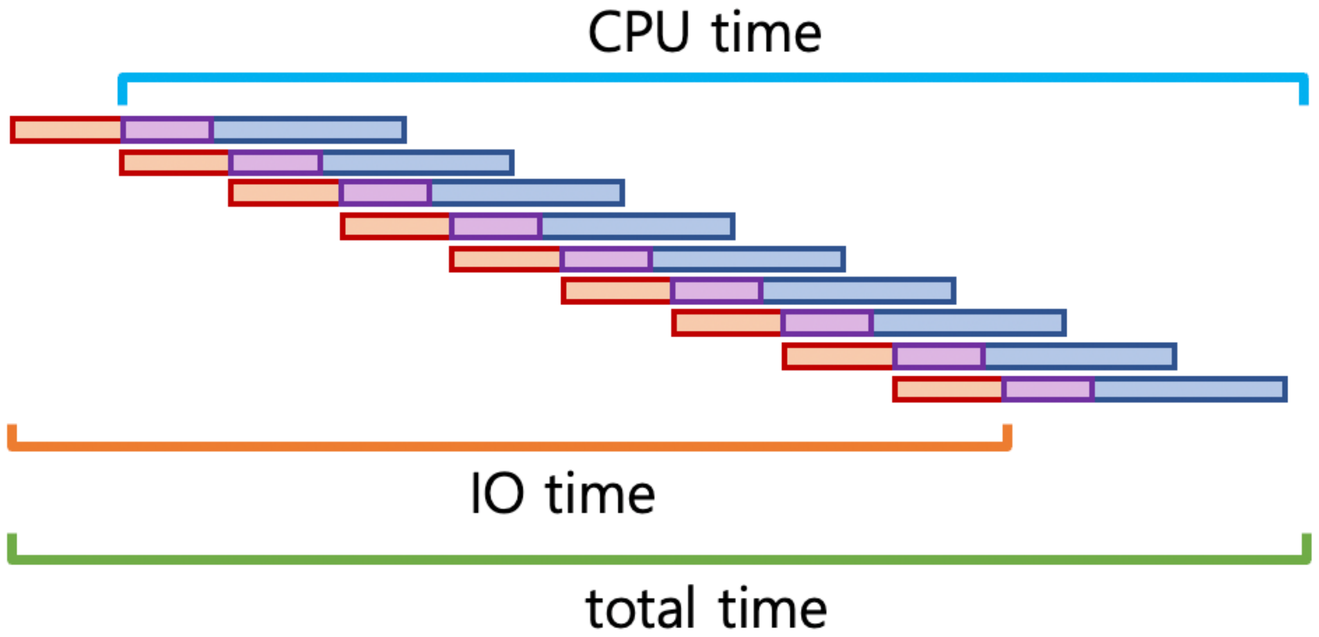




2.3 Full scan Search 성능

블록 단위 압축 Full scan 검색

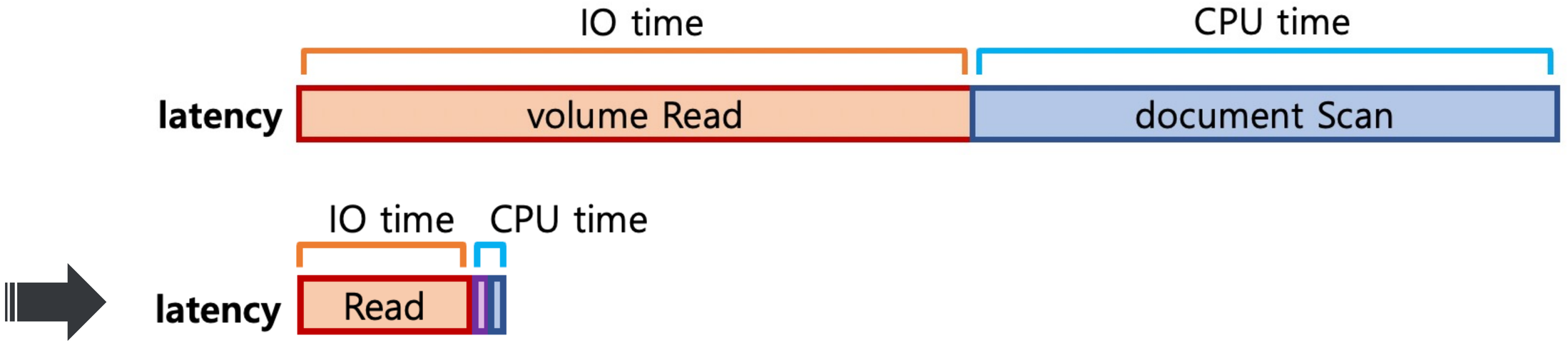
총 검색시간 ≈ IO read time



2.3 Full scan Search 성능

NAVER DEVIEW 2023

블록 단위 압축 Full scan 검색



Read time: 문서 압축으로 75%감소

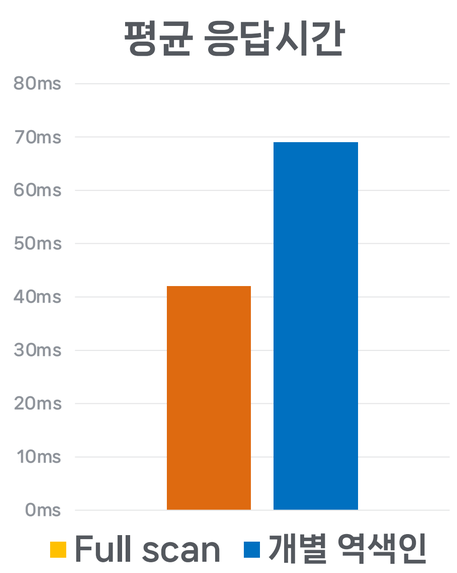
Decode & Scan time: 병렬화로 최소화

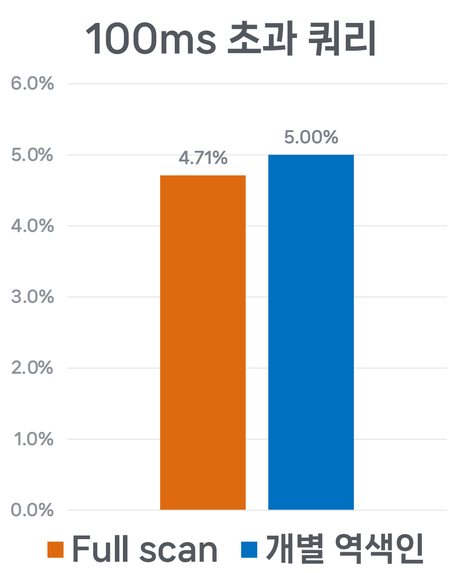
2.3 Full scan Search 성능

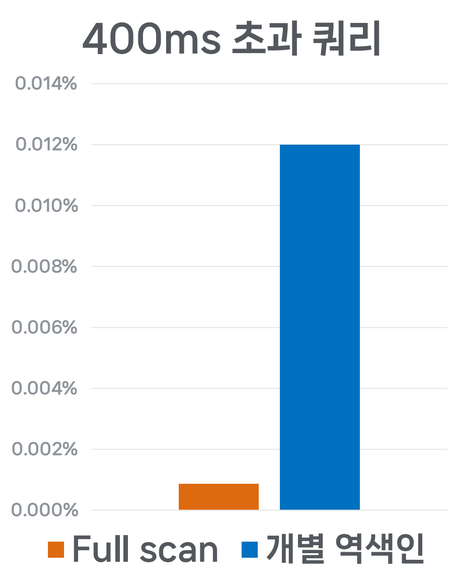
NAVER DEVIEW 2023

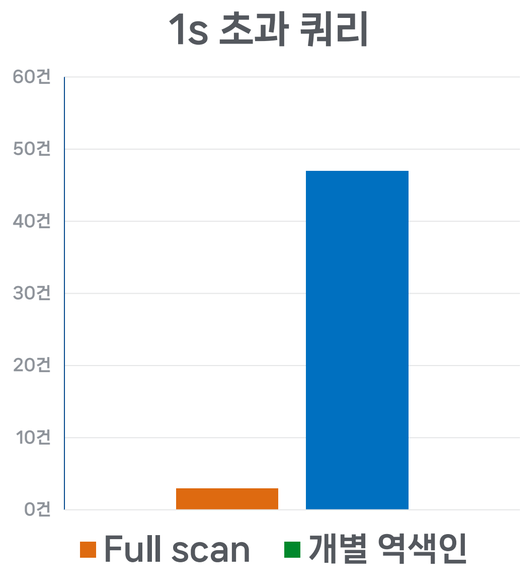
네이버메일 검색 서비스 성능

* 낮을수록 좋음, MVP버전









네이버메일은 평균 볼륨 크기가 작아서 (수십MB 이하) Full scan방식을 적용하기 좋은 서비스

3. Noir

3.1 Noir concept

Noir: No Information Retrieval

✓개별 데이터 검색 솔루션
✓Full scan 검색방식

3.1 Noir concept

Full scan 검색

- 개별 데이터 서비스에서 충분한 검색속도
- 볼륨 생성, 업데이트 비용이 저렴함
- 검색볼륨과 문서볼륨의 역할이 통합된 볼륨
- 부분일치 검색 방식

3.2 네이버메일 Noir 적용 결과

Noir 적용 장점

- ✓ 볼륨 크기 감소로 장비 절감
- ✓ 직관적인 검색로직
- ✓ 볼륨생성 및 증분 비용 절감
- ✓ 운영 공수 감소

3.2 네이버메일 Noir 적용 결과 – 장비 절감

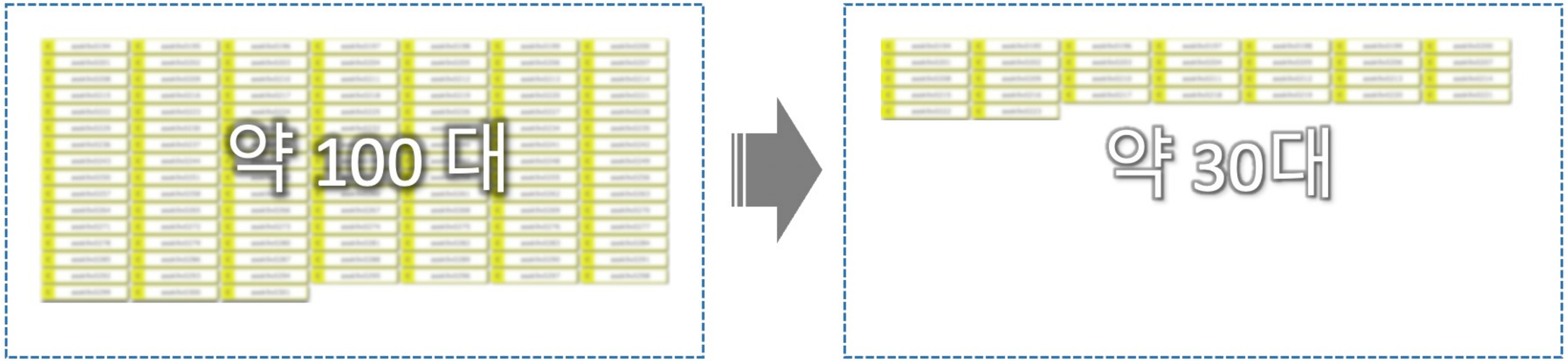
NAVER DEVIEW 2023

기존 메일검색

NoIR 메일검색

색인서버 없음

3.2 네이버메일 Noir 적용 결과 – 검색장비 감소



✓ **역색인을 사용하지 않음**

- 색인볼륨이 없다 (문서볼륨만 존재)
- 볼륨크기 1/2로 감소

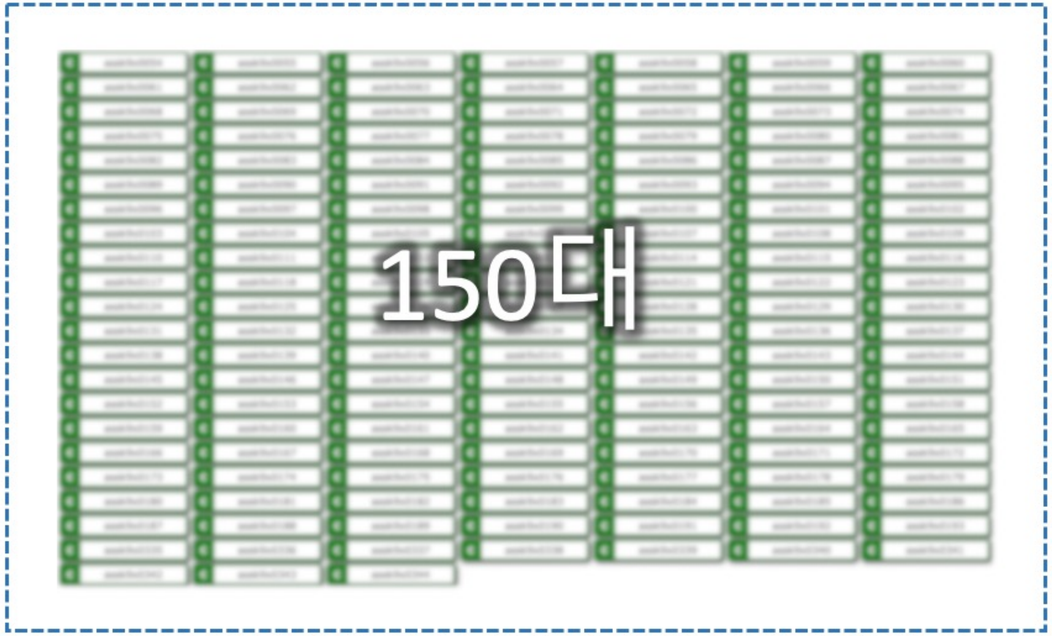
✓ **블록 단위 압축 적용**

- 높은 압축율을 위해 다수의 문서를 모아 압축한다
- 문서단위 압축하던 기존 문서볼륨의 1/2 크기

볼륨크기와 검색서버 수요가 1/4로 감소

3.2 네이버메일 Noir 적용 결과 – 색인서버 삭제

NAVER DEVIEW 2023



✓역색인 구조를 사용하지 않음

- 볼륨생성 & 증분 비용이 낮아졌다
- 검색서버가 볼륨생성까지 담당할 수 있게 됨

3.2 네이버메일 Noir 적용 결과 – 장비 절감

NAVER DEVIEW 2023

기존 메일검색

NoIR 메일검색

색인서버 없음

15%미만 장비로 동일한 서비스 제공

3.2 네이버메일 Noir 적용 결과

NAVER DEVIEW 2023

서비스 요구사항에 부합함

메일검색, 메시지검색 등 개별 데이터 서비스는 일반적으로 부분일치 검색을 요구한다

역색인 검색엔진의 부분일치

색인 시 bigram분해 필요

bigram 시 Term개수↑ 볼륨크기↑ 장비수요↑

Noir의 부분일치

자연스럽게 제공됨

직관적인 검색로직으로 CS 최소화

3.2 네이버메일 Noir 적용 결과

볼륨구축 및 증분 비용 감소

문서입력은 원본 문서를 append하는 간단한 작업

볼륨 크기가 늘어나도 입력성능 유지

각 볼륨에 병렬 입력 가능

운영 공수 절감

역색인 자료구조로 인한 복잡한 오류 없음

색인장비 제거로 장애 포인트 감소

수평확장이 용이함. 볼륨을 이전하는 간단한 리밸런싱 작업

4. 구현 및 설계

4.1 볼륨 설계

볼륨 설계

1. 문서 CRUD 요청
2. 문서 구조 설계
3. 문서 영속성 보장

4.1 볼륨 설계 - 문서 입력

문서 입력 연산 목표

✓ 문서입력 중 검색 가능해야 한다

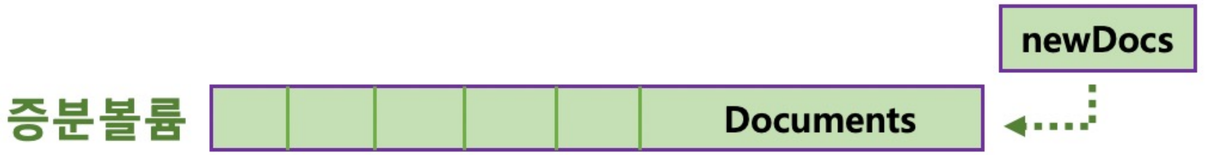
✓ 문서입력 중 셧다운 당하더라도 볼륨이 망가지지 않아야 한다

✓ 문서입력 비용을 최소화하고 싶다

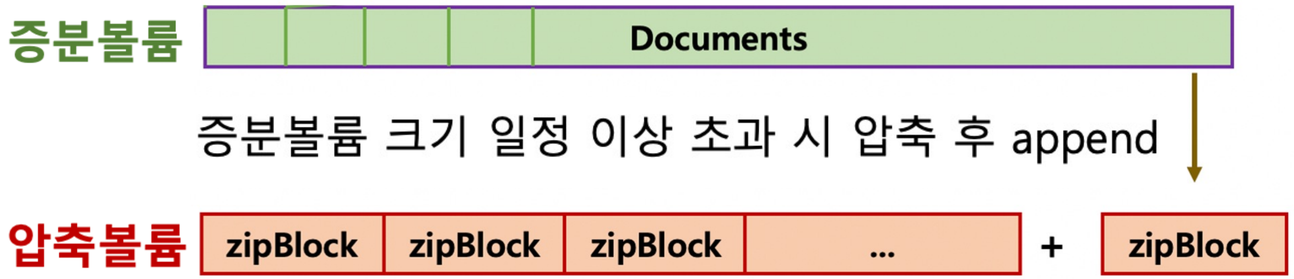
4.1 볼륨 설계 - 문서 입력

문서 저장 방식

1. 신규문서는 증분볼륨에 입력



2. 증분볼륨이 일정 이상 커지면
   한꺼번에 압축볼륨으로 반영

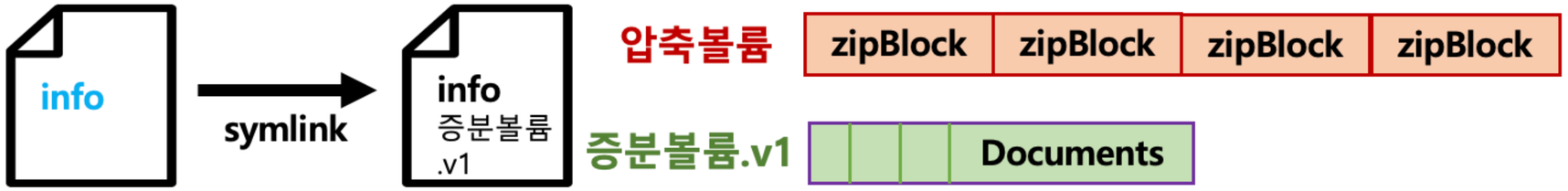


✓ 신규문서가 충분히 쌓이기 전까지 압축을 미루는 방식으로 비용 효율화

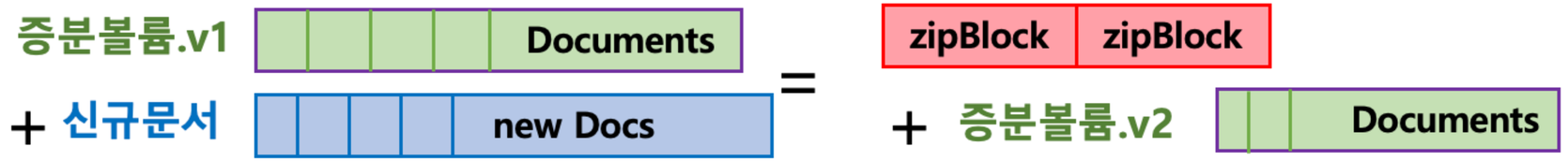
4.1 볼륨 설계 - 문서 입력

NAVER DEVIEW 2023

0. Compaction 시작 전



1. 기존 증분볼륨에 신규문서를 더해 압축

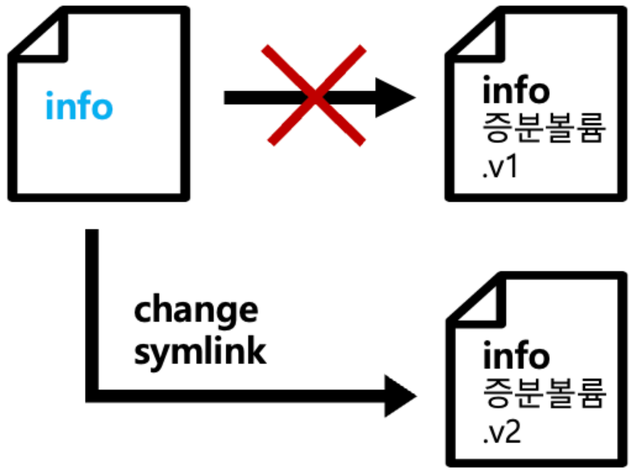


2. 신규 압축블록은 기존 압축볼륨에 append

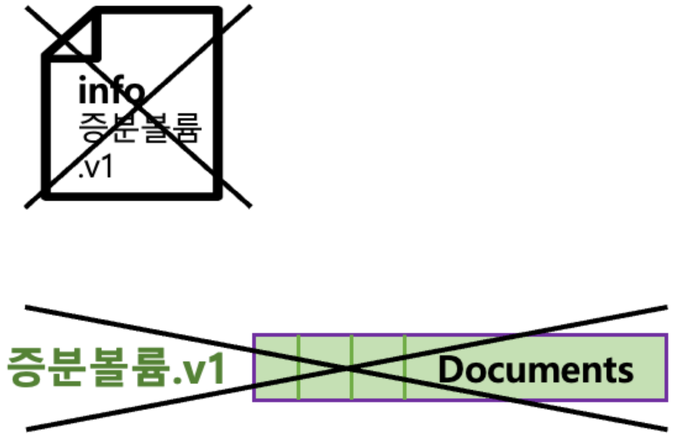
4.1 볼륨 설계 - 문서 입력

3. 압축볼륨에 덧붙여진 블록과 새로 생성된 증분볼륨.v2 를 알고있는 메타데이터 디렉토리 생성

4. 새 info디렉토리로 심볼릭링크 변경



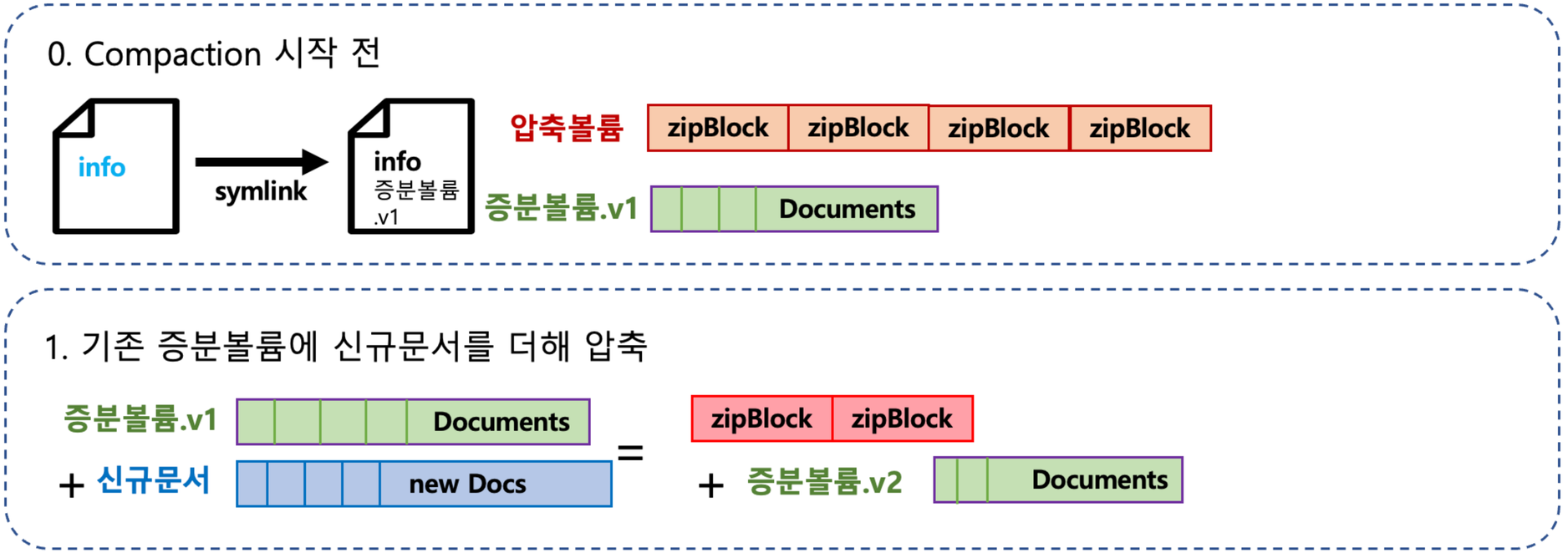
5. 기존 info디렉토리 삭제



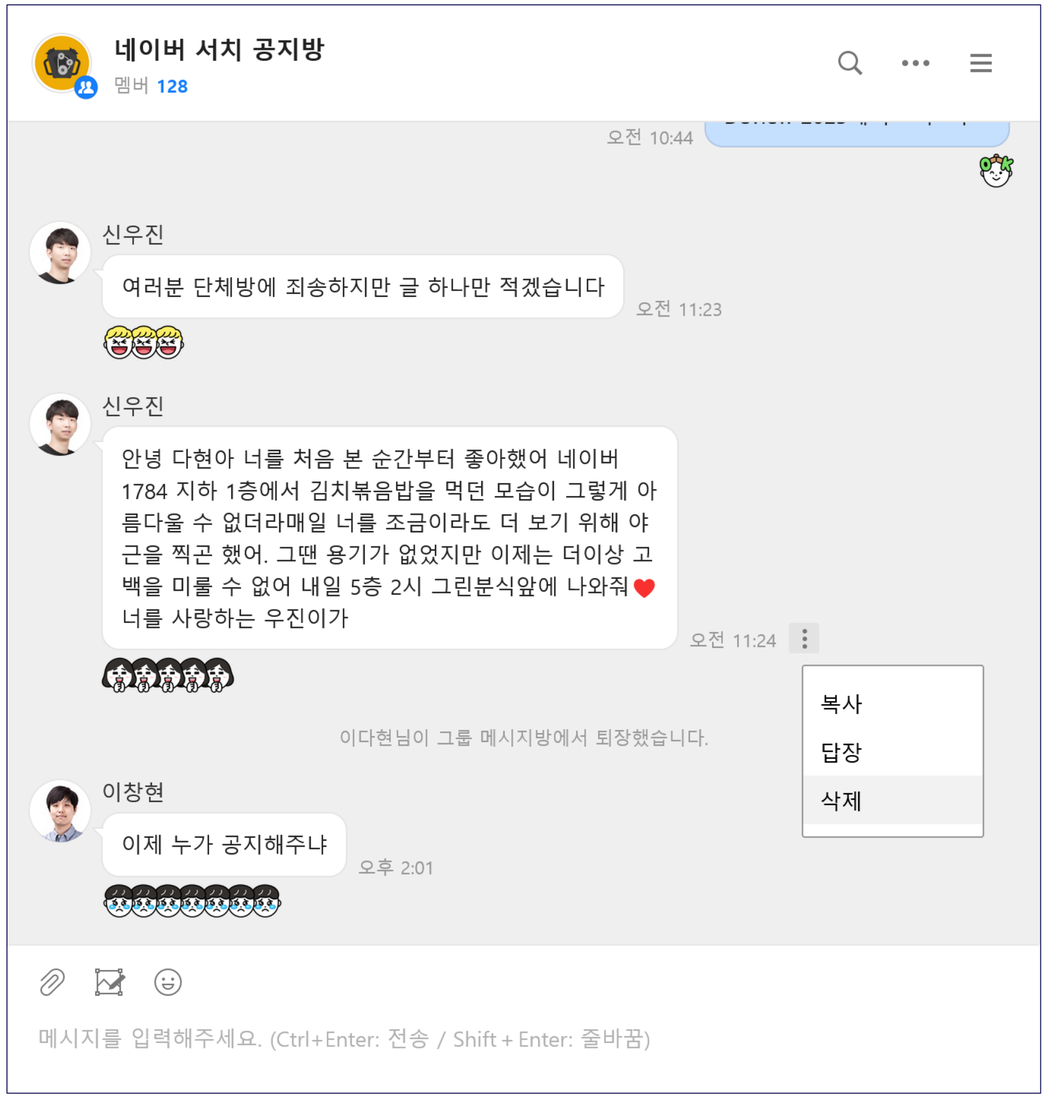
4.1 볼륨 설계 - 문서 입력

기존 메타데이터로 검색할 시 신규 문서는 제외하고 기존 문서만 scan

- ✓ 문서입력 중 검색 가능
- ✓ 문서입력 중 셧다운 당해도 볼륨이 망가지지 않음



4.1 볼륨 설계 - 문서 수정 및 삭제



문서 수정 및 삭제

Noir의 볼륨은 압축해서 저장되기 때문에 immutable하다

어떻게 구현할 수 있을까?

4.1 볼륨 설계 - 문서 수정

문서 수정

문서 수정 요청 유입 시 매번 볼륨 내 문서를 수정하기 어려움

별도의 Patch 볼륨에 변경내역을 쌓아둔 뒤 검색 시 동적으로 반영

변경 내역이 많이 쌓이면 검색 성능 저하

일정 이상 쌓이면 볼륨을 재구축해 반영

4.1 볼륨 설계 - 문서 삭제

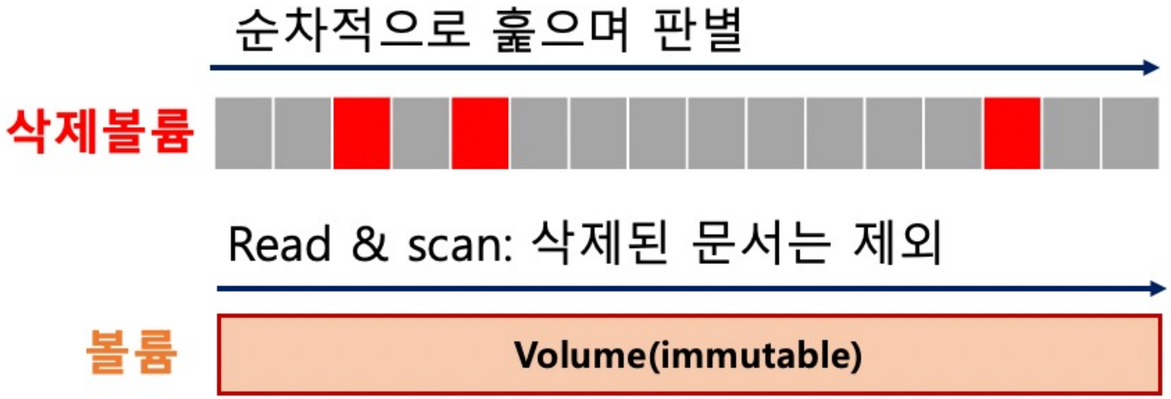
문서 삭제

문서 삭제요청 인입 시마다 볼륨 내 문서를 제거하기 어려움

별도 볼륨에 문서당 1bit 삭제표현

검색 시 순차적으로 훑으며 삭제여부 확인

삭제된 문서일 시 scan하지 않음



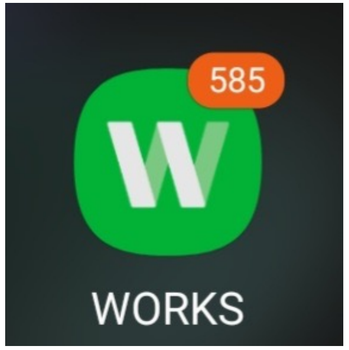
필요시 볼륨을 재구축해 문서를 삭제하는 작업 수행

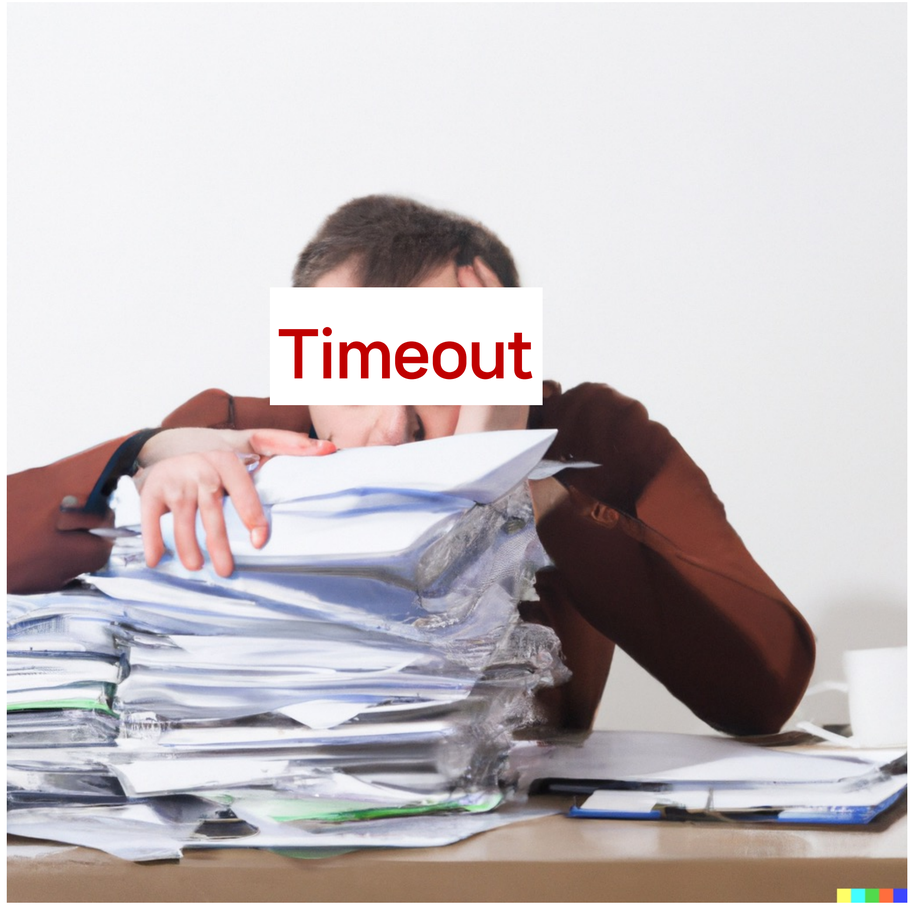
4.1 볼륨 설계 - Timeout

NAVER DEVIEW 2023

메일

메신저





4.1 볼륨 설계 - 연속검색

연속검색

Timeout

Noir의 검색 응답속도는 문서 수와 비례

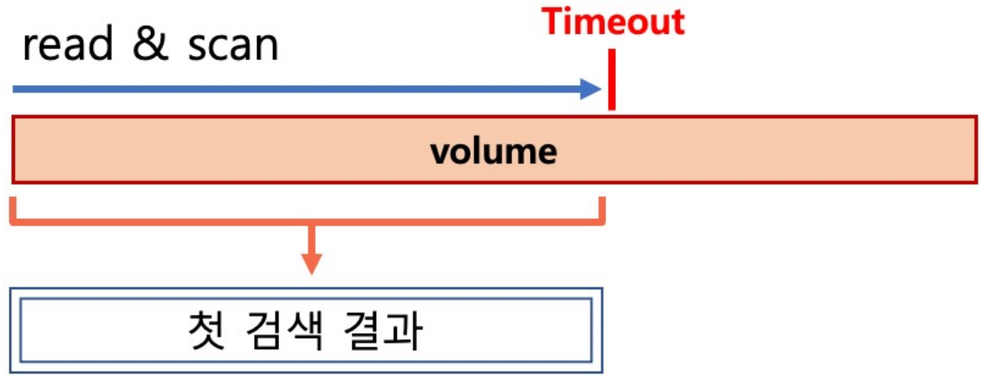
큰 볼륨에서 timeout내 검색 불가능한 쿼리가 발생

연속 검색

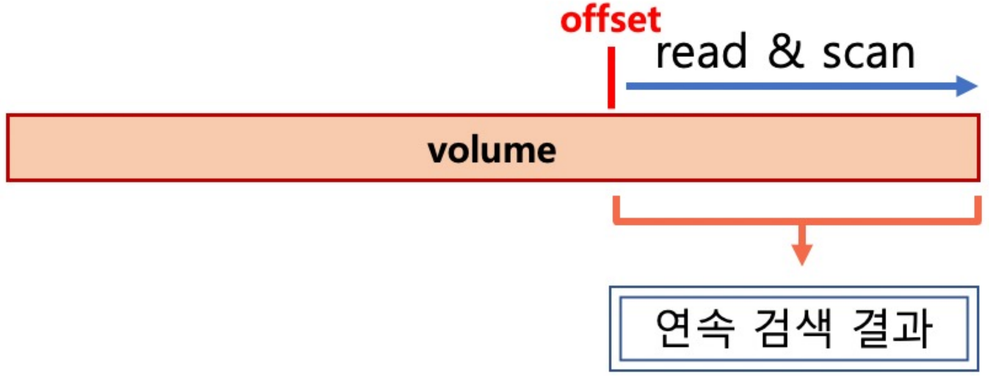
Timeout이 발생하면 검색한 결과를 최대한 반환

Timeout 발생지점 offset으로 나머지 문서 검색 가능

1. Timeout 발생



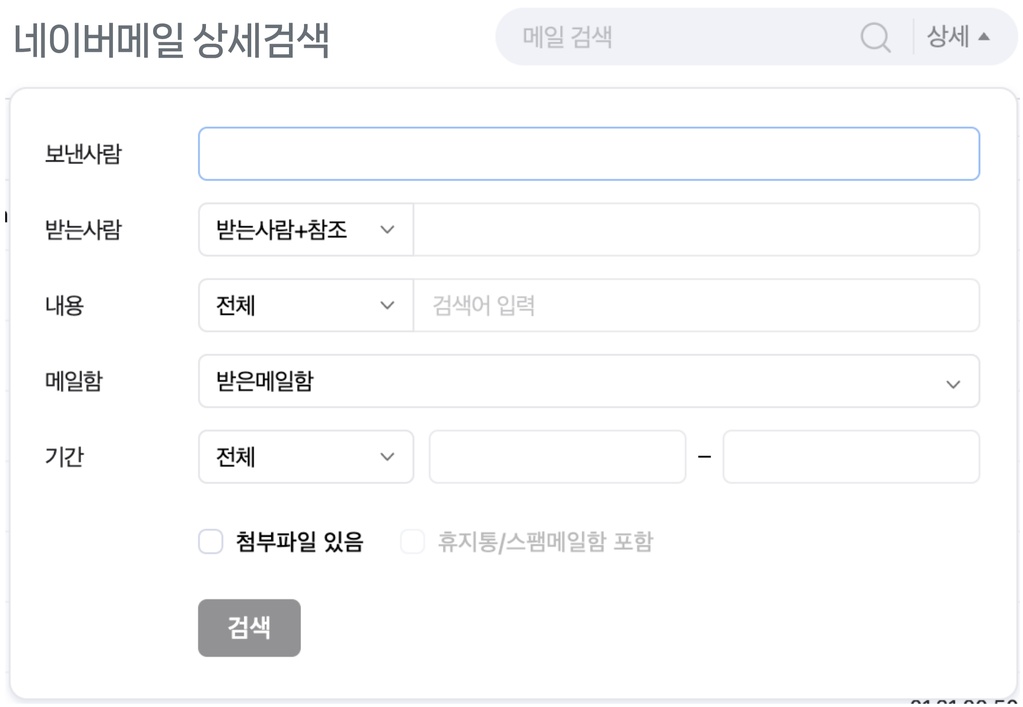
2. 연속 검색 수행

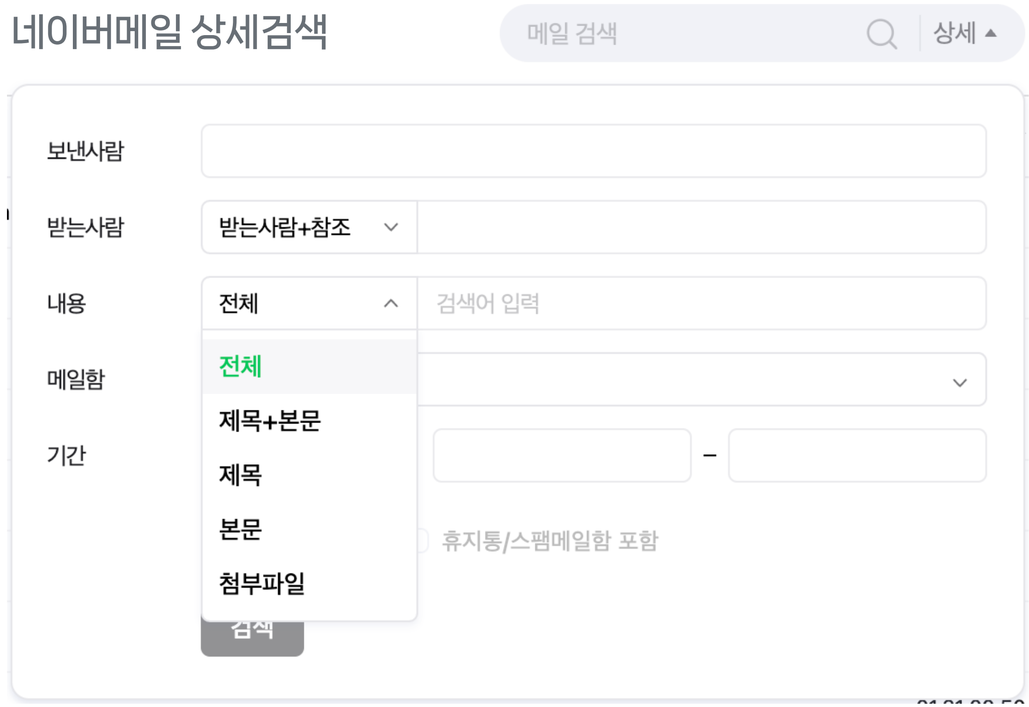


4.1 볼륨 설계 - Boolean Query tree

NAVER DEVIEW 2023

실서비스는 단순 부분일치 이상의 기능을 요구한다

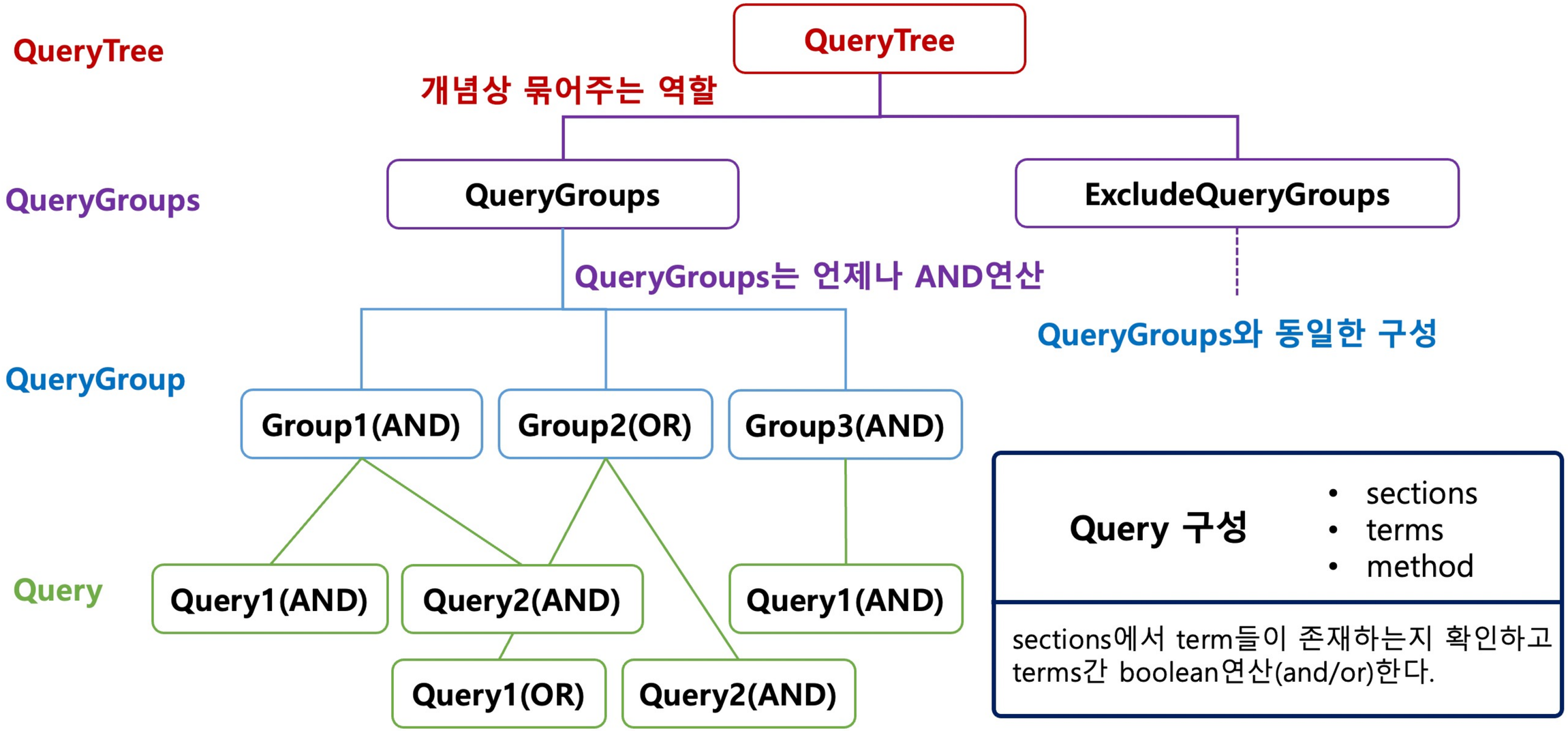




선택한 섹션(필드) 부분일치 결과 병합

4.1 볼륨 설계 - Boolean Query tree

NAVER DEVIEW 2023



Query 구성

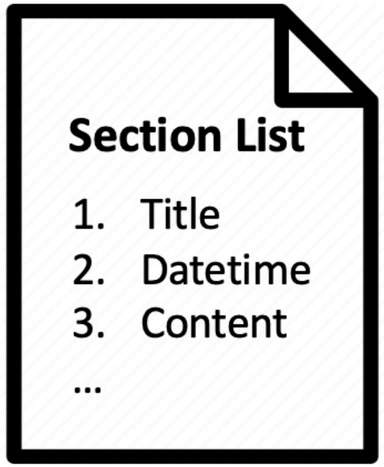
- sections
- terms
- method

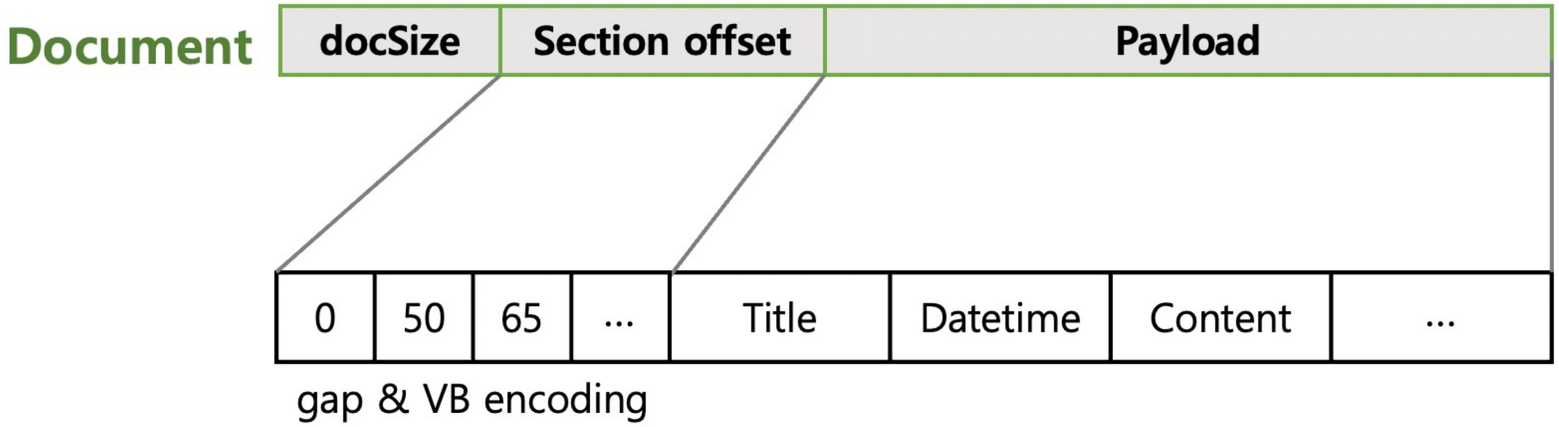
sections에서 term들이 존재하는지 확인하고 terms간 boolean연산(and/or)한다.

4.1 볼륨 설계 - 문서 설계

문서 설계

목표: 볼륨 크기 최소화





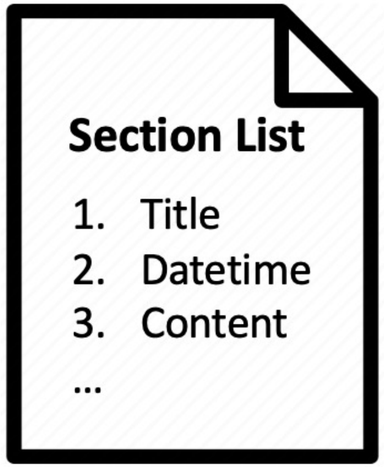
Section list가 섹션 구성을 저장해 문서 크기 감소

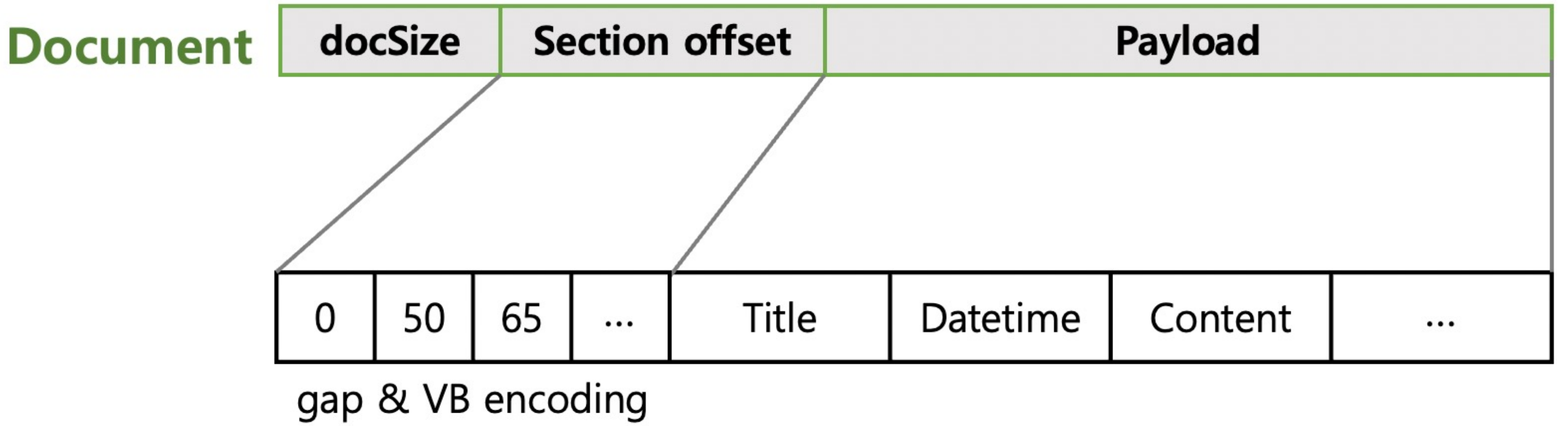
- 각 볼륨마다 section 등장 순서를 알려주는 section list 관리
- 문서 내용을 payload에 section list 순서대로 저장
- Section Offset에 gap & VB encoding 적용

4.1 볼륨 설계 - 문서 설계

문서 설계

목표: Schemaless(각 문서의 섹션 구성이 다를 수 있음)





- 처음보는 섹션이 인입되면 섹션 리스트에 추가된다
- 섹션이 삭제되어 어떤 섹션이 비어있을 때 gap & vb encoding되어 1byte만 차지한다

4.1 볼륨 설계 - 문서 영속성 보장

저널링

서버 셧다운 시 인입된 요청이 유실되지 않게 한다

작동 방식

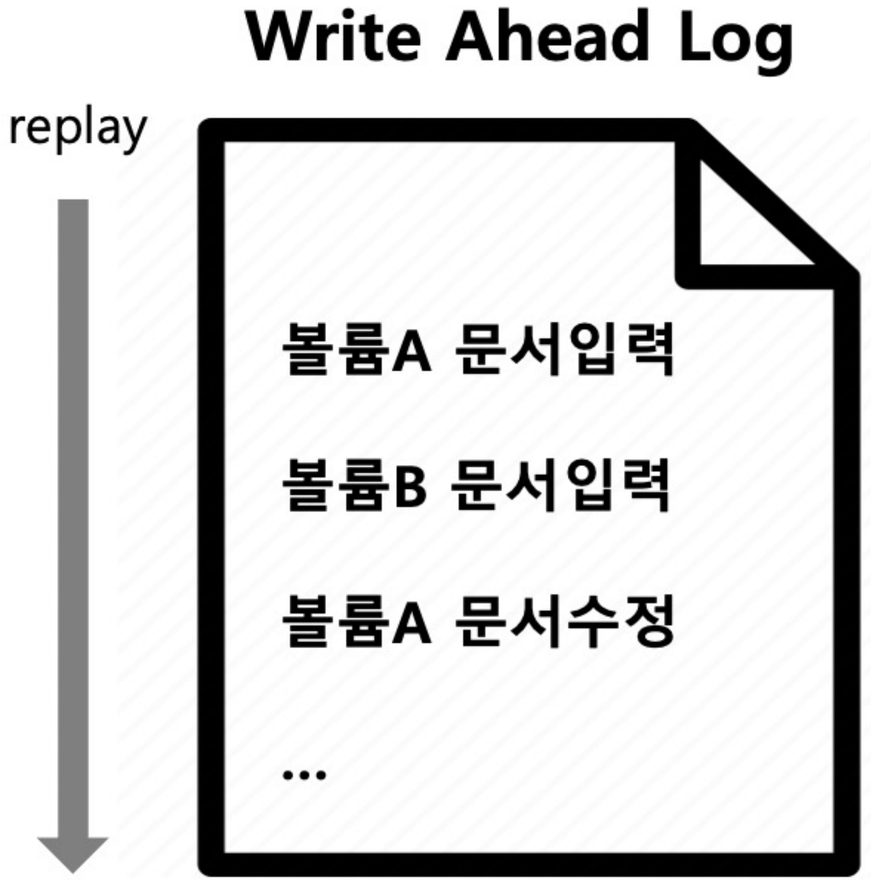
요청을 WAL에 완전히 로깅 후 요청 수행

서버 셧다운 시 기록된 모든 요청을 replay

각 로그는 멱등성(idempotent) 만족 필요

Noir에서 로그 단위는 각 요청

replay로 인한 중복 수행 극복



4.1 볼륨 설계 - 문서 영속성 보장

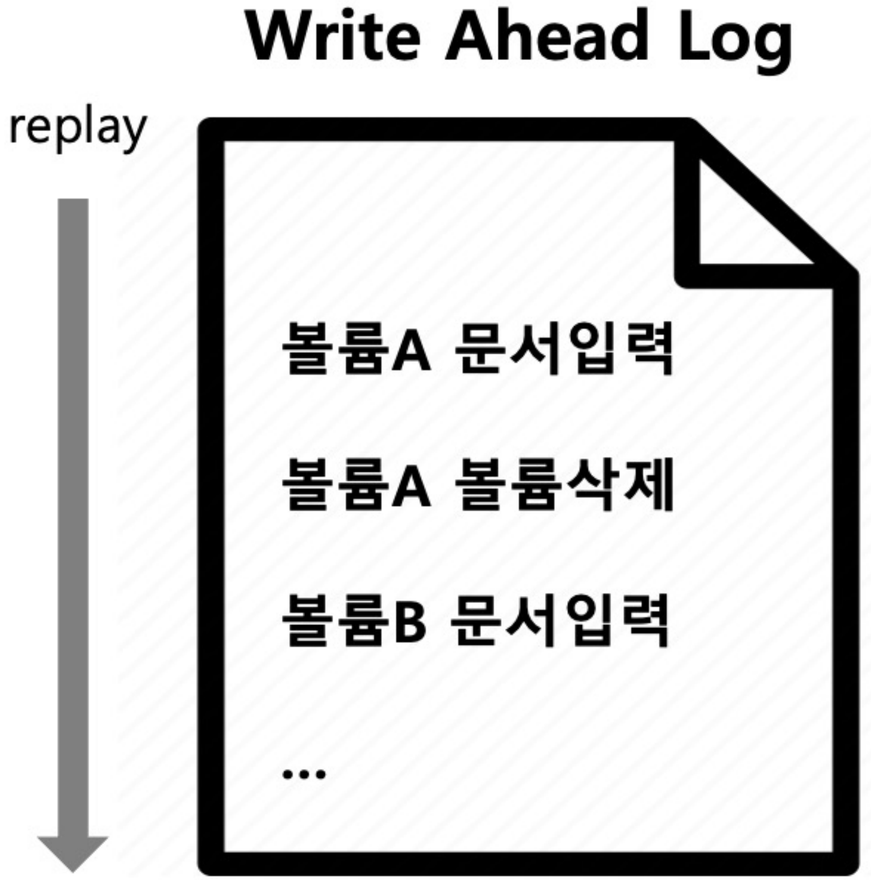
저널링 구현

볼륨 버전 - ABA problem

볼륨 삭제 이전 볼륨 A와 재생성된 볼륨 A를 구분 불가

각 로그의 볼륨 명에 볼륨 버전 추가

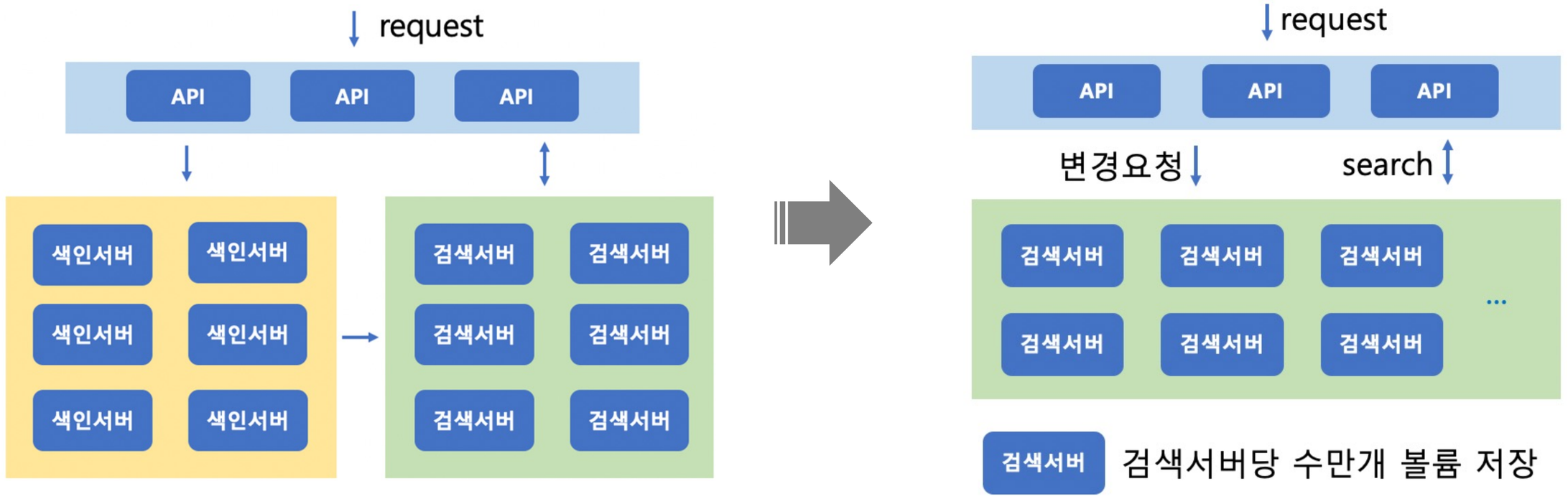
ex) '볼륨A.{version} 문서입력'



ABA problem 예

4.2 분산 설계 - 분산 구조

분산 구조



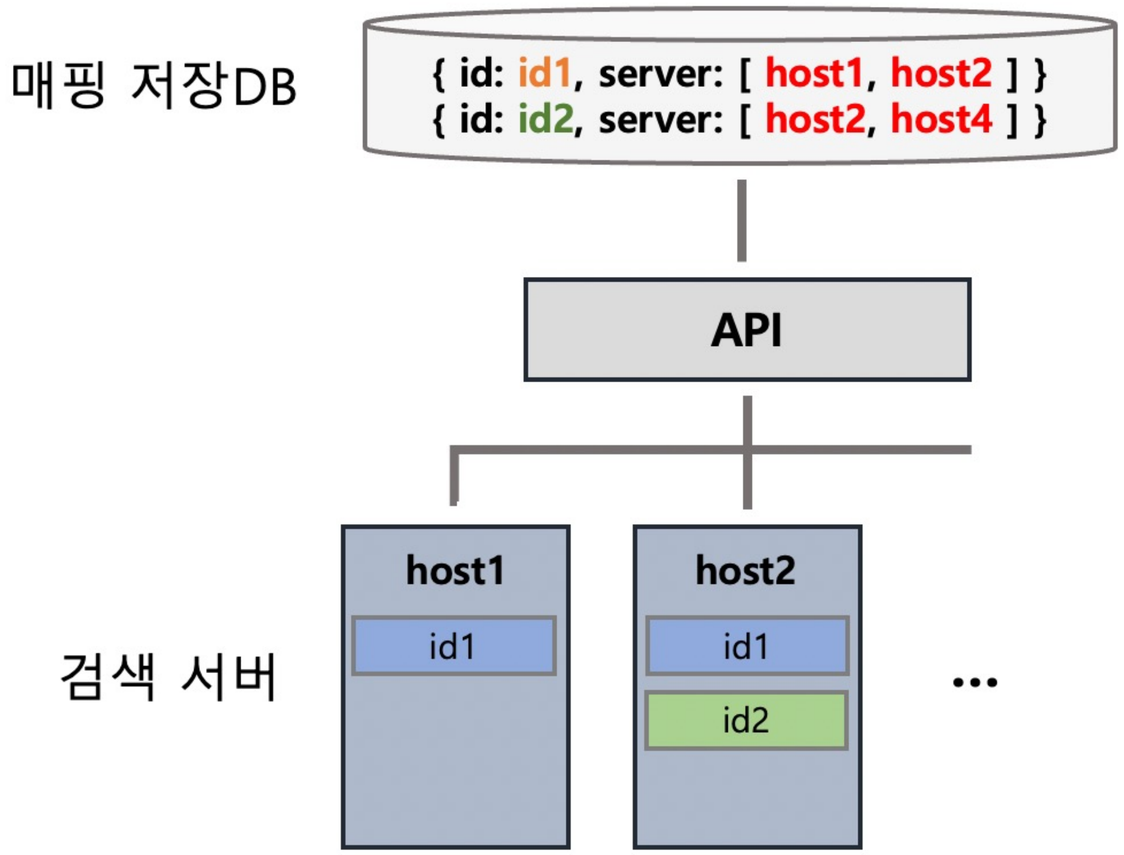
기존 개별 역색인 시스템

Noir

4.2 분산 설계 - 볼륨 이중화

볼륨 replication 분산 저장

- 중앙 DB에서 이중화 상태 관리
- 신규 볼륨은 랜덤 배치 후 필요 시 리밸런싱
- Standby Replication



4.2 분산 설계 - consistency

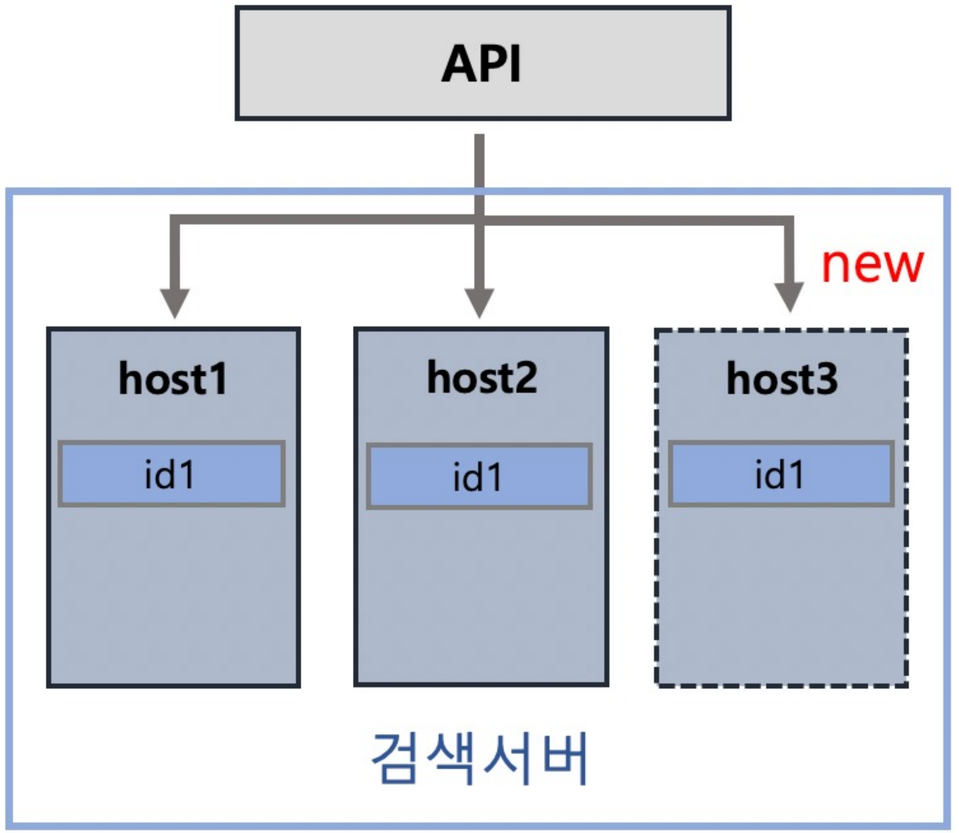
NAVER DEVIEW 2023

토폴로지 버전 관리

이중화 구성 변경 중 볼륨 간 consistency 보장

이중화 host 추가 시 타이밍 이슈 예

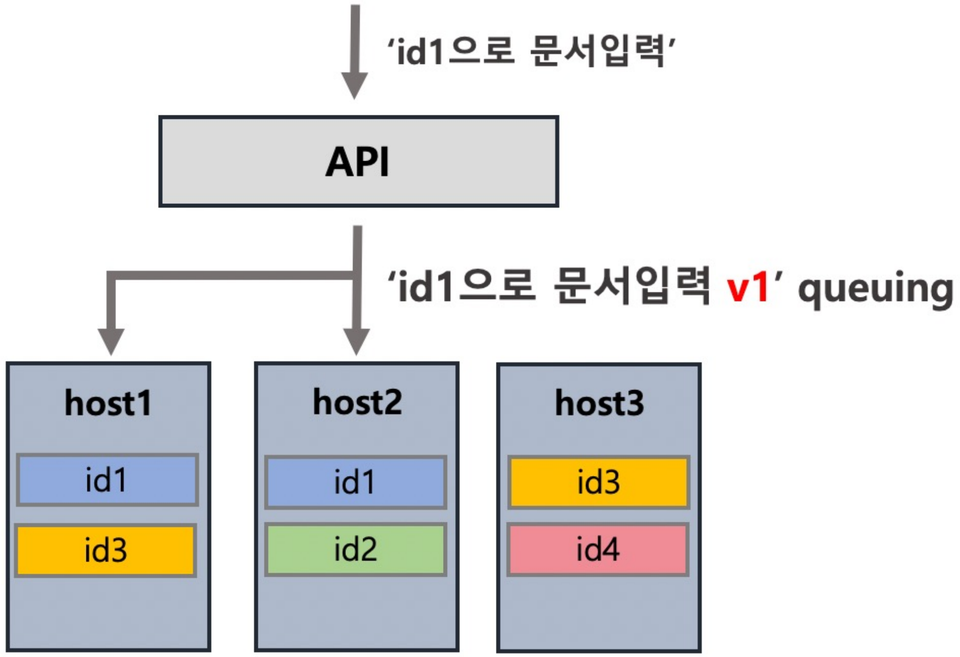
1. 요청인입되면 검색서버의 queue에 쌓인다
2. 하지만 실제 수행시점에는 host가 추가되었을 수 있다
3. 추가된 host는 이 요청을 받지 못했으므로 실패시켜야 한다



4.2 분산 설계 - consistency

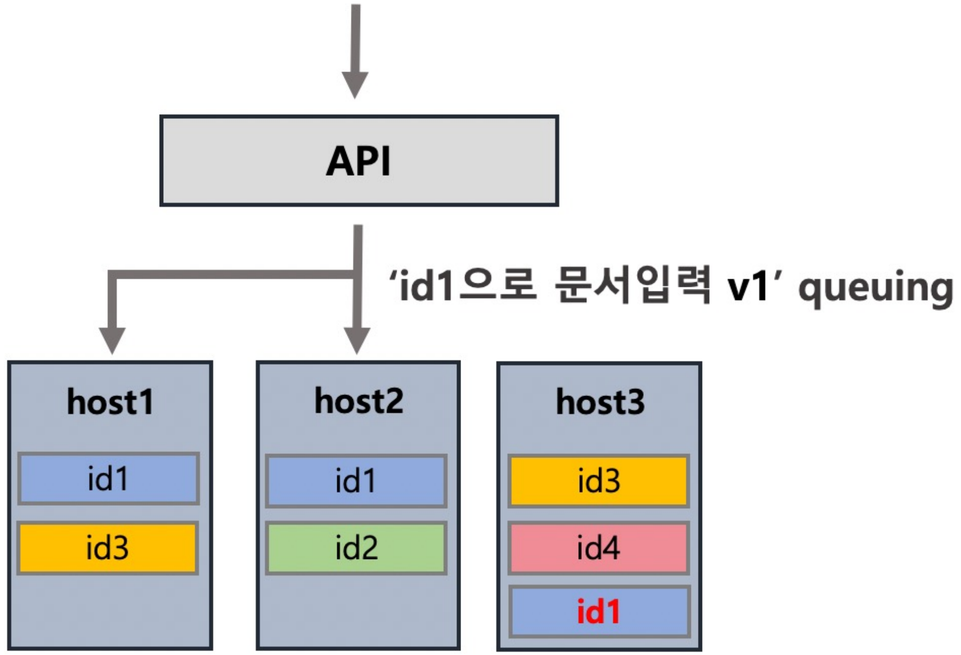
NAVER DEVIEW 2023

1. 요청 인입시 현재 토폴로지 버전을 검색서버로 전달



id1 global topology version: **v1**

2. 볼륨 Replication0이 추가되면 토폴로지 버전 갱신

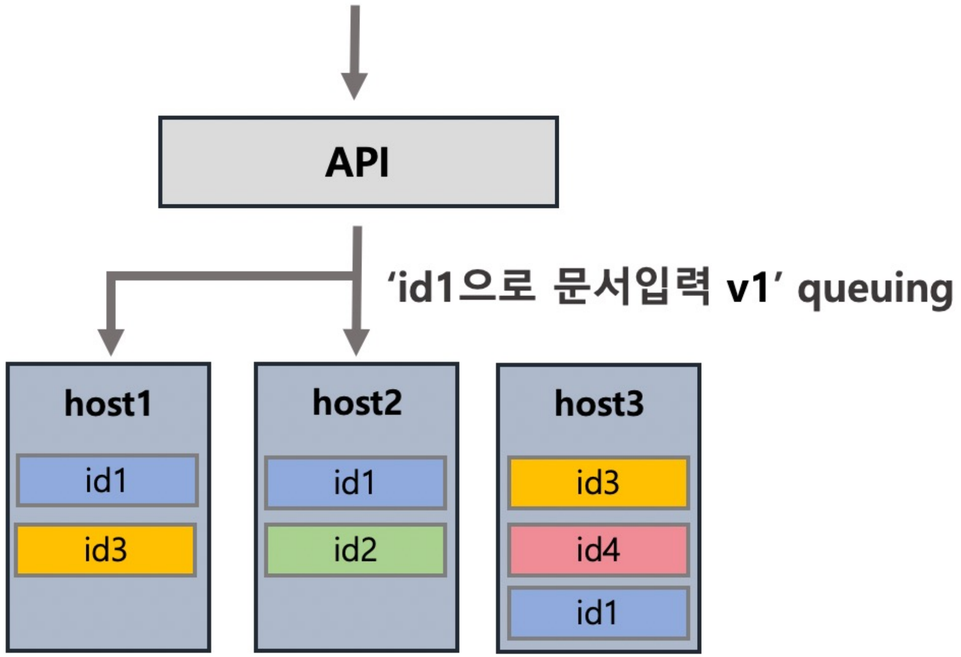


id1 global topology version: **v1** → **v2**

4.2 분산 설계 - consistency

NAVER DEVIEW 2023

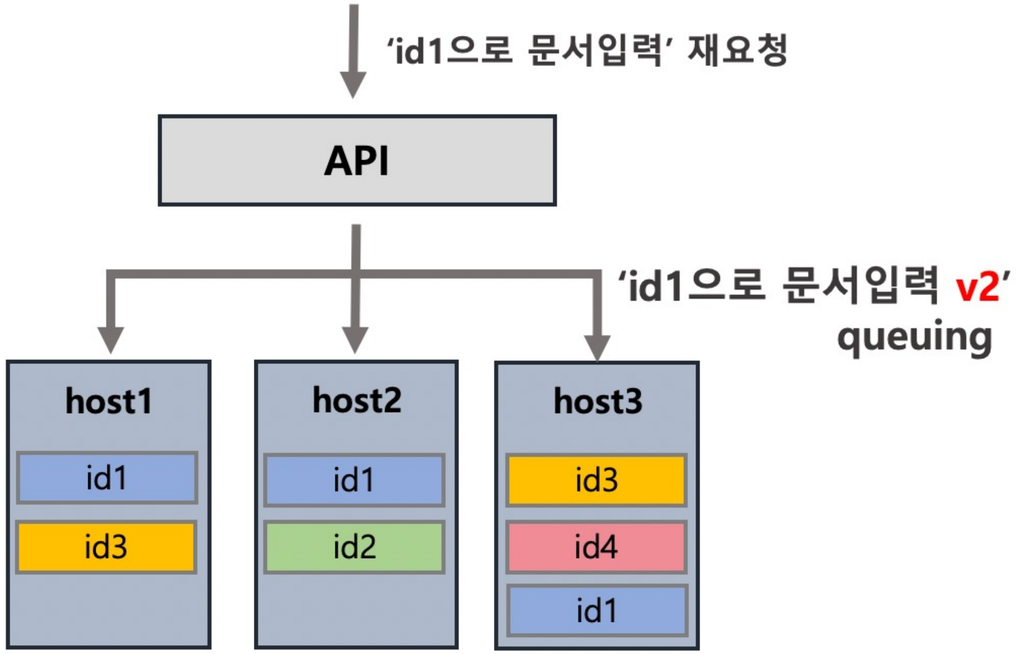
3. 수행 시점의 버전과 요청에 담긴 버전이 다르므로 실패 (host3는 문서입력을 전달받지 못함)



request version v1 ≠ global version v2

id1 global topology version: **v2**

4. 재요청시 host3 까지 정상적으로 포함



id1 global topology version: **v2**

4.3 고도화 계획

NAVER DEVIEW 2023

볼륨 샤딩

다수의 노드에서 병렬 검색

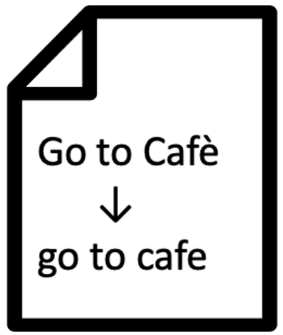
검색 timeout 발생 최소화

Collation

문서 내용을 미리 정규화

요약문 추출 시 테이블 활용

문서 정규화

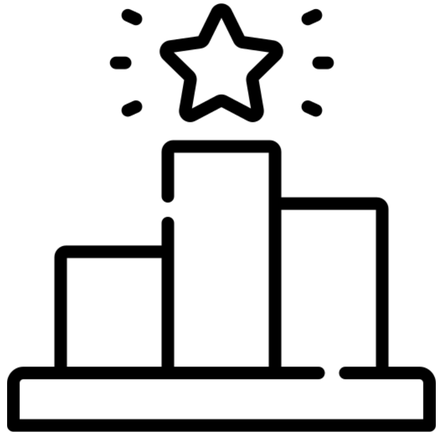


원복 테이블

Offset	원본 문자
0	G
6	C
9	è

Proximity 적용

통계 기반 대신 근접도 기반 랭킹



5. 요약

5. 요약

개별 데이터를 다루며 유입문서가 많은 서비스에서 효율적

메일검색에서 기존 역색인방식 대비 장비 88% 절감

블록 단위 압축 저장 및 병렬 scan

문서 증분비용 감소, 운영 공수 감소

Thanks to..

NAVER
DEVIEW
2023

WORKS MOBILE Mail Dev team

7. appendix

7.1 Golang

선택한 이유

1. 생산성 (러닝커브↓ build time↓)
2. 적당한 성능

Golang 이 Noir 의 성능 요건을 충분히 만족시킬 수 있었다

실현 가능성이 높지 않았던 프로젝트로 빠르게 작성해서 적용하는 과정이 필요했다

POC -> MVP -> 실서비스 적용까지 비교적 매우 적은 M/M 소요

7.2 제한사항

1. 전체문서set에 대해 검색하는 서비스에 적합하지 않음

2. 통계기반(TF-IDF, bm25) 랭킹 불가, 단순 정렬 가능

3. 문서 반영에 약간의 시간 필요 (대다수 검색엔진과 동일)

4. SSD 장비 필수 (read IO 병목)

7.3 Future

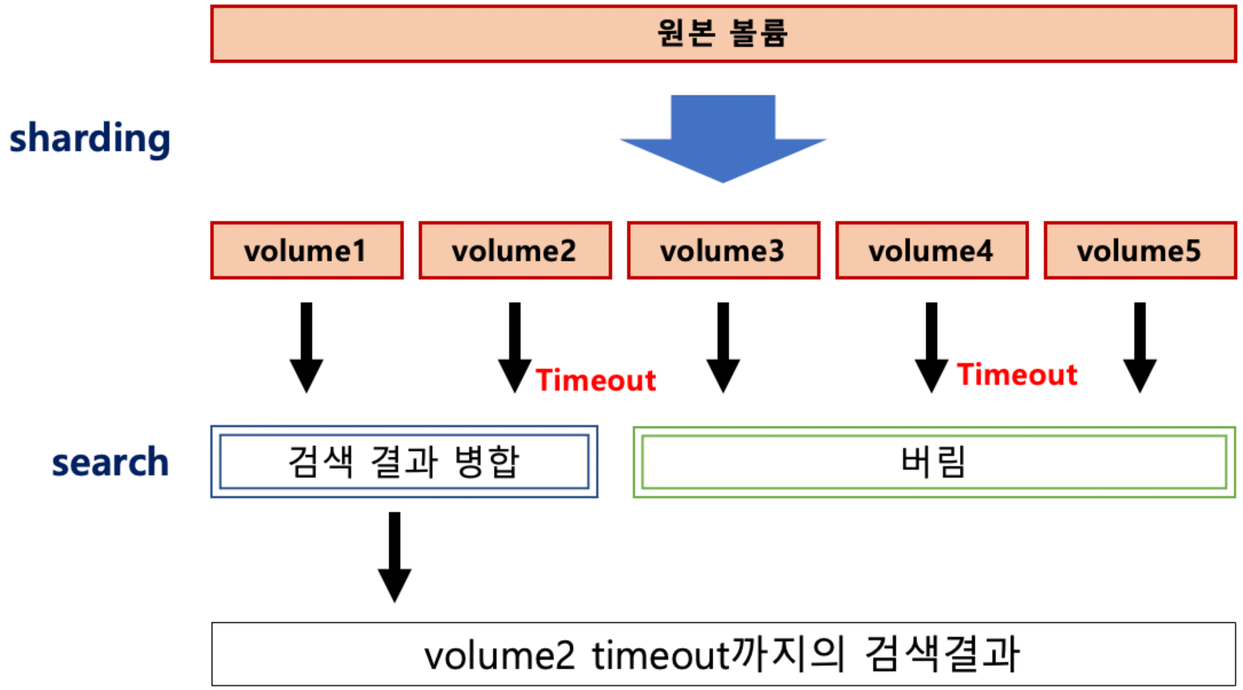
NAVER
DEVIEW
2023

- 추가 기능 계획들
- Proximity 랭킹
- 오픈소스화

7.4 볼륨 샤딩

큰 볼륨을 작은 볼륨으로 분산 관리

- 다수의 노드에서 병렬 검색
- 검색 timeout 발생 최소화
- 볼륨 재구축 작업 분할 수행 가능



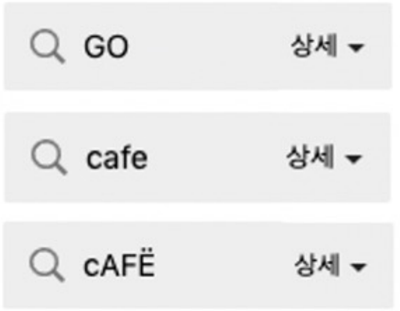
7.5 Collation

NAVER DEVIEW 2023

동일한 의미의 문자를 구분없이 검색하는 기능

등가성(equivalence) 문자 예

- 대문자/소문자
- 전각문자/반각문자
- 히라가나/가타카나



위 문서를
검색 가능한 쿼리들

[go to Cafè] 쿼리로 [Go to cafe] 를 검색하려면?

- 쿼리와 원본문서를 모두 [go to cafe]로 정규화 후 검색

7.5 Collation

NAVER DEVIEW 2023

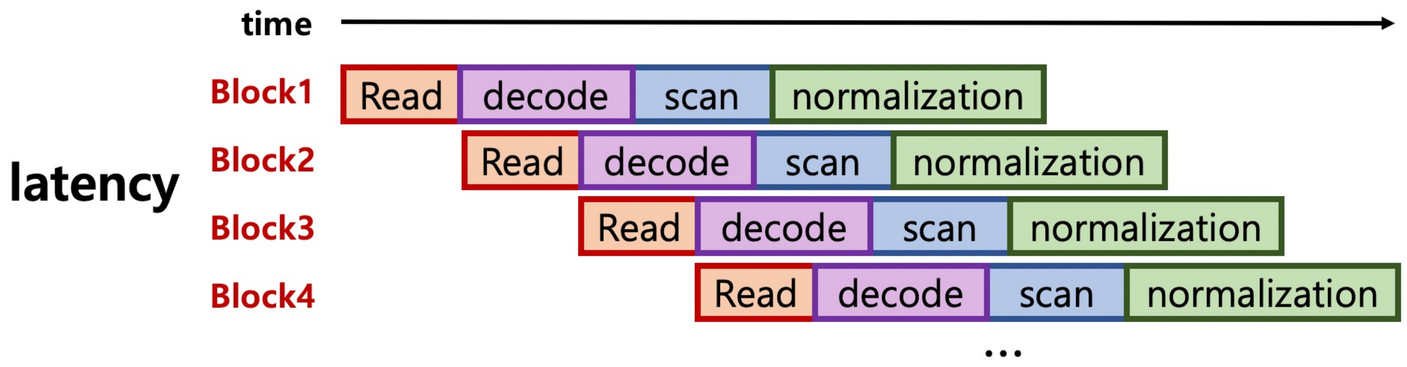
현재 구현: 검색 시 정규화

검색 시 문서내용을 동적으로 정규화

검색요청이 몰릴 시 CPU 과부하 발생

실서비스에서 CPU병목 발생

불안정한 CPU 지표



7.5 Collation

개선 예정: 전처리 테이블

문서내용을 미리 정규화해서 검색성능 개선

입력 문서를 정규화

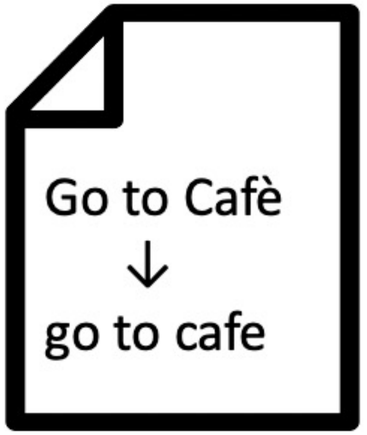
원복용 테이블 생성

문서 요약문 추출 시 테이블 활용

테이블을 사용해서 원복

일부 문서만 요약문 추출해 성능 이슈 없음

볼륨 내 문서 정규화



변환 테이블

Offset	원본 문자
0	G
6	C
9	è