SW 스타랩 설계

1차년도 주요 수행 내역

2차년도 주요 수행 내역

과제명: IoT 환경을 위한 고성능 플래시 메모리 스토리지 기반 인메모리 분산 DBMS 연구개발
과제번호: 2017-0-00477

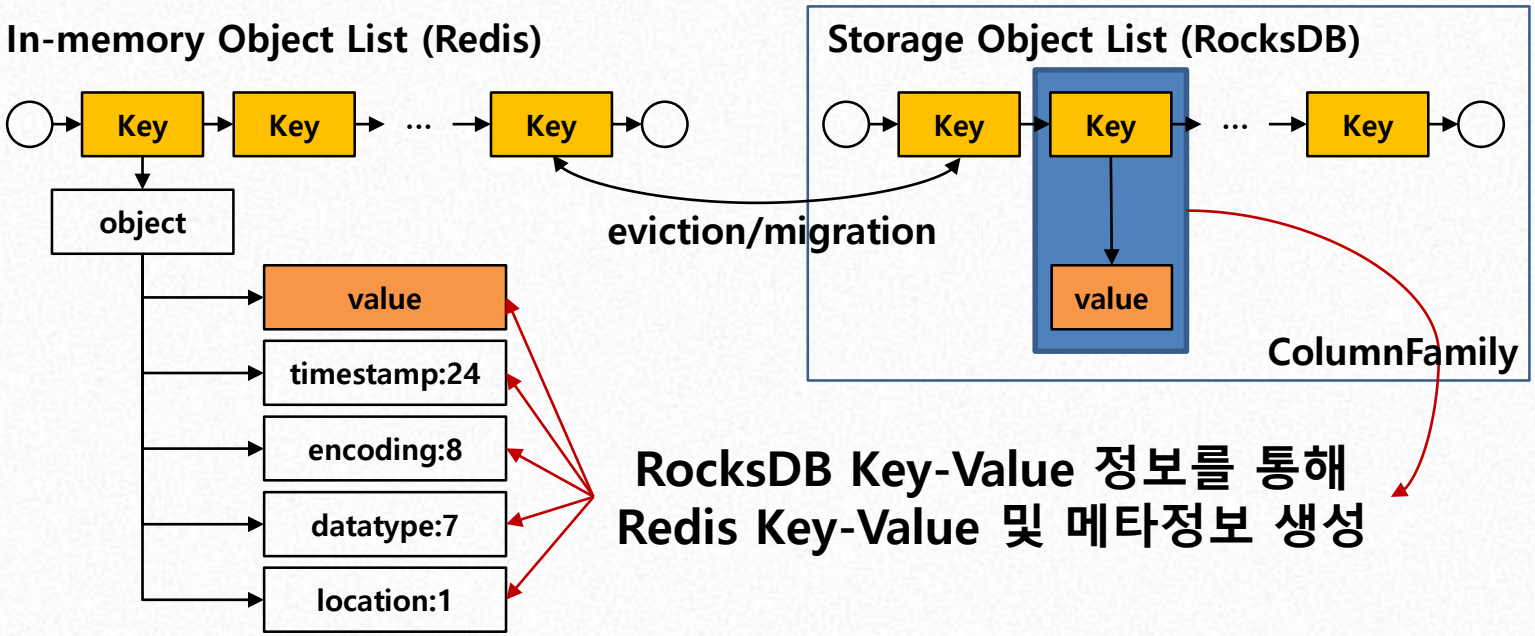
과학기술정보통신부 Ministry of Science and ICT
연세대학교 YONSEI UNIVERSITY
IITP 정보통신기술진흥센터 Institute for Information & communications Technology Promotion

01 | 1차년도 주요 수행내역

1차년도 주요 수행내역

인메모리 저장구조

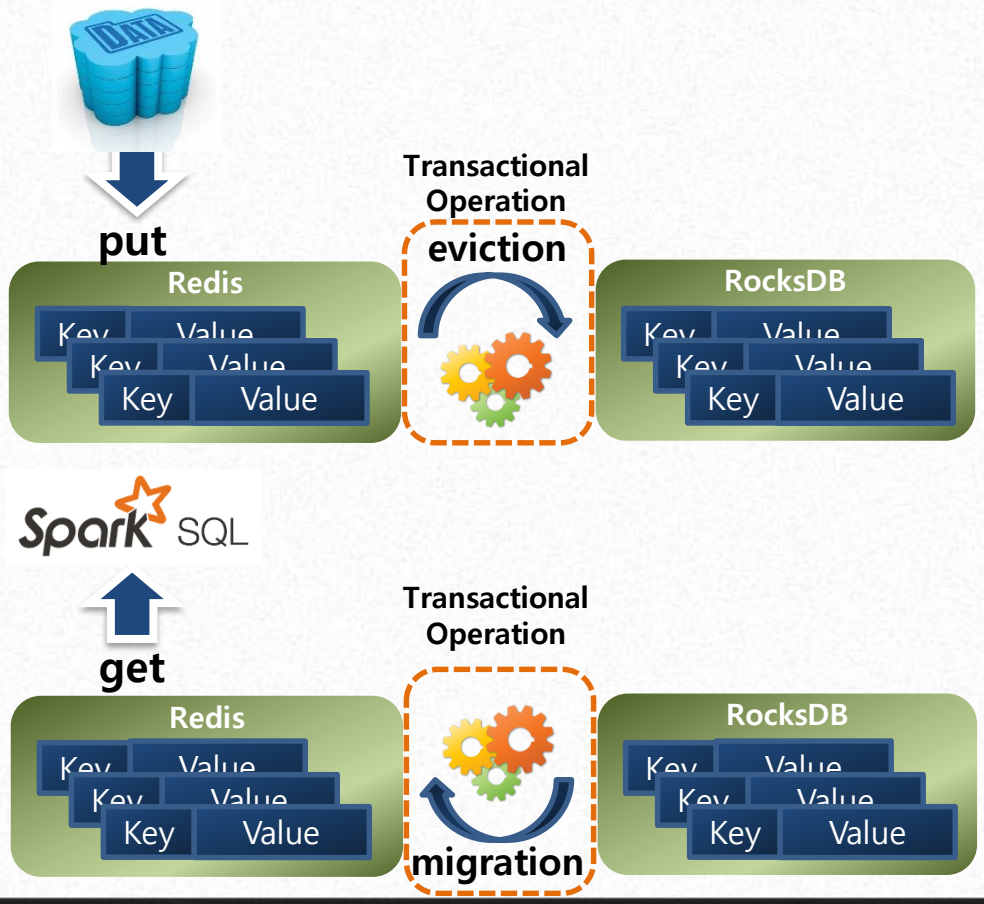
- 인메모리-스토리지 계층 간 데이터 저장 구조
  - Redis-RocksDB 간 구조체 변환
  - Redis K-V 구조체의 메타정보 유지를 위해 RocksDB K-V 구조체에 해당 정보 유지



1차년도 주요 수행내역

- Eviction/Migration 설계 및 구현

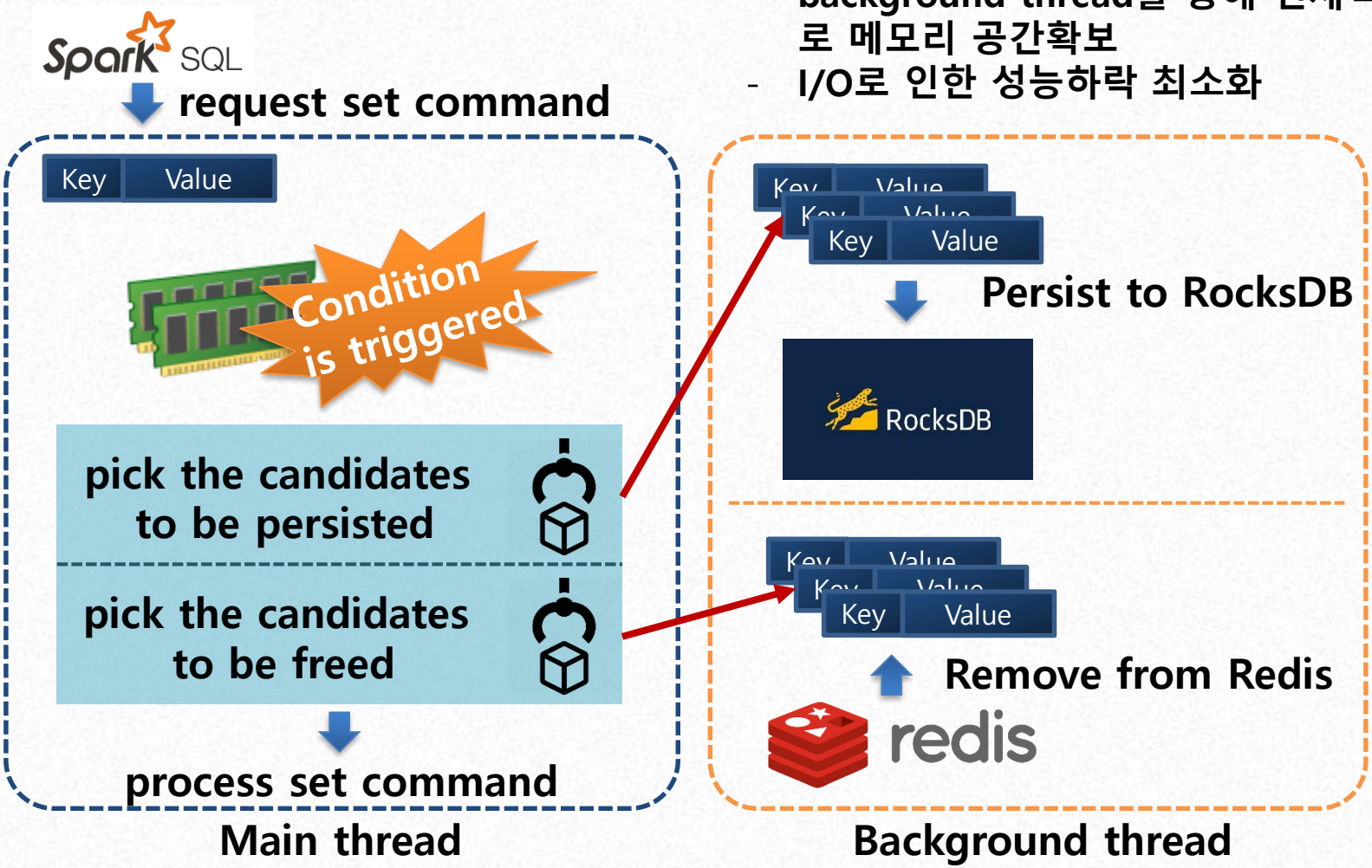
- Eviction/Migration 설계
  - 물리 메모리가 부족할 경우, 메모리에 위치한 데이터를 플래시 메모리 스토리지로 이동시키는 기능



1차년도 주요 수행내역

- Eviction/Migration 설계 및 구현

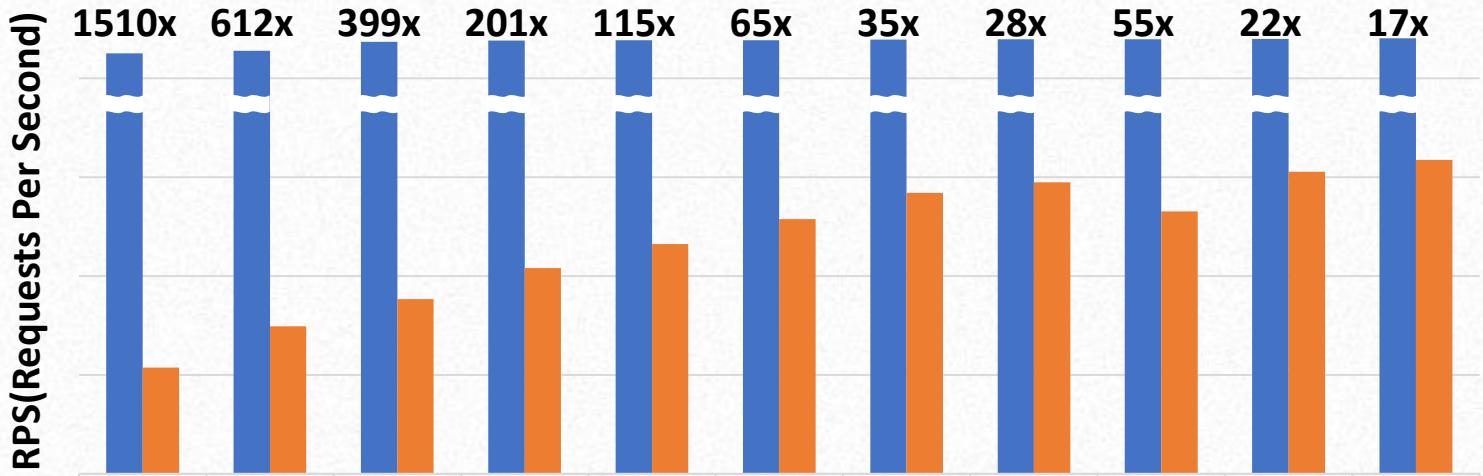
- Eager-Background Eviction
  - 메모리 공간이 부족해지기 전에 background thread를 통해 선제적으로 메모리 공간확보
  - I/O로 인한 성능하락 최소화



1차년도 주요 수행내역

- Eviction/Migration 설계 및 구현

- Eager-Background Eviction 성능평가
  - On-demand 방식에 비해 4Kbytes 데이터 크기 기준 1510배, 4 bytes 기준 17배 이상 빠른 처리성능을 보임
  - 순수 RocksDB I/O 부하를 확인하기 위해 Logging off 기준 측정

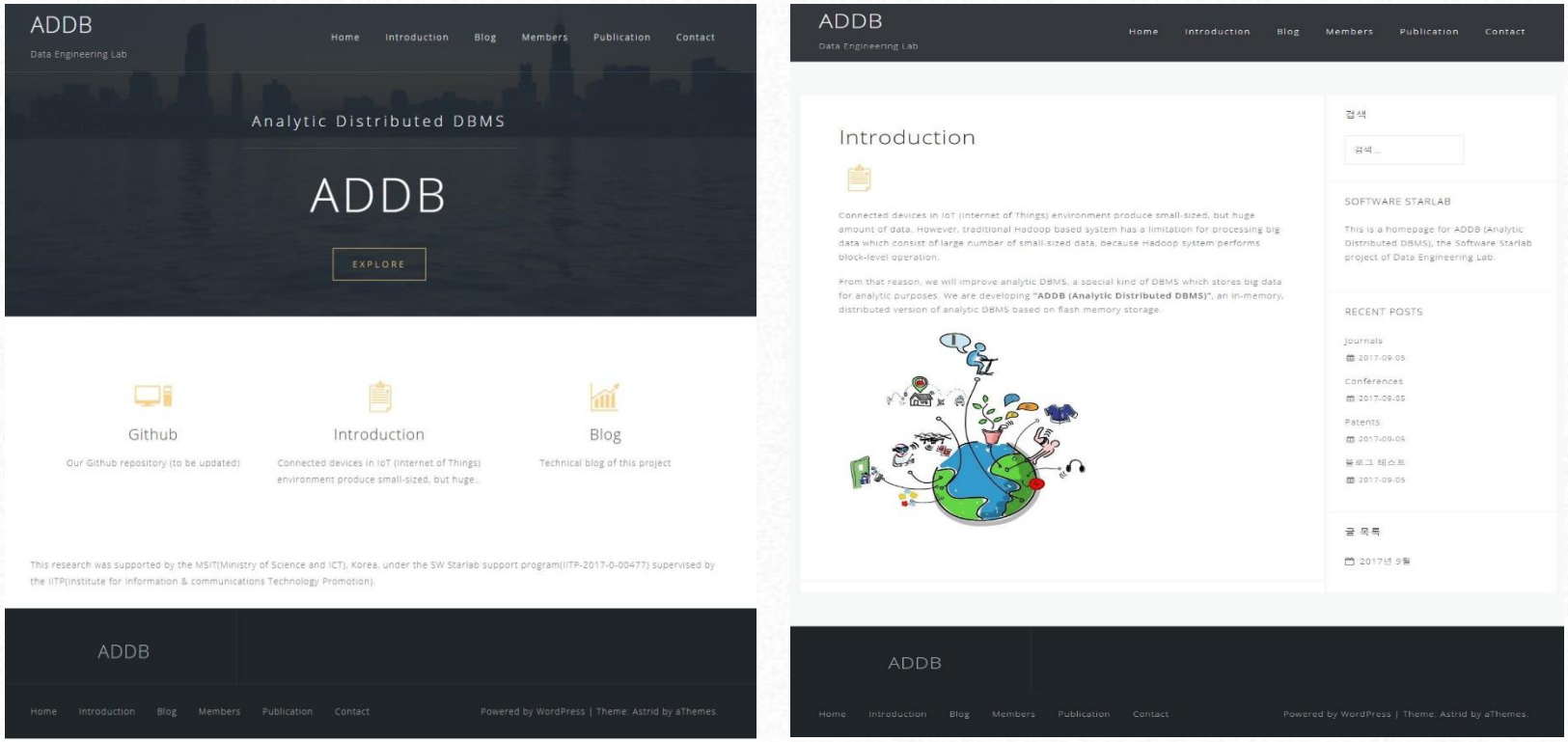


Bytes	4096	2048	1024	512	256	128	64	32	16	8	4
Eager-background	53912	57017	70163	72595	72801	73220	74196	74904	74553	75058	76169
On-demand	35.71	93.2	175.7	361.6	634.5	1135	2093	2672	1357	3408	4485

1차년도 주요 수행내역

- 공개S/W 기반 구축

  - 홈페이지 공개
    - Code repository 및 Issue tracker, CI 서버 연동
    - 기술 블로그 및 실적 공개
    - 공개 라이선스 선정 및 공개 범위 설정

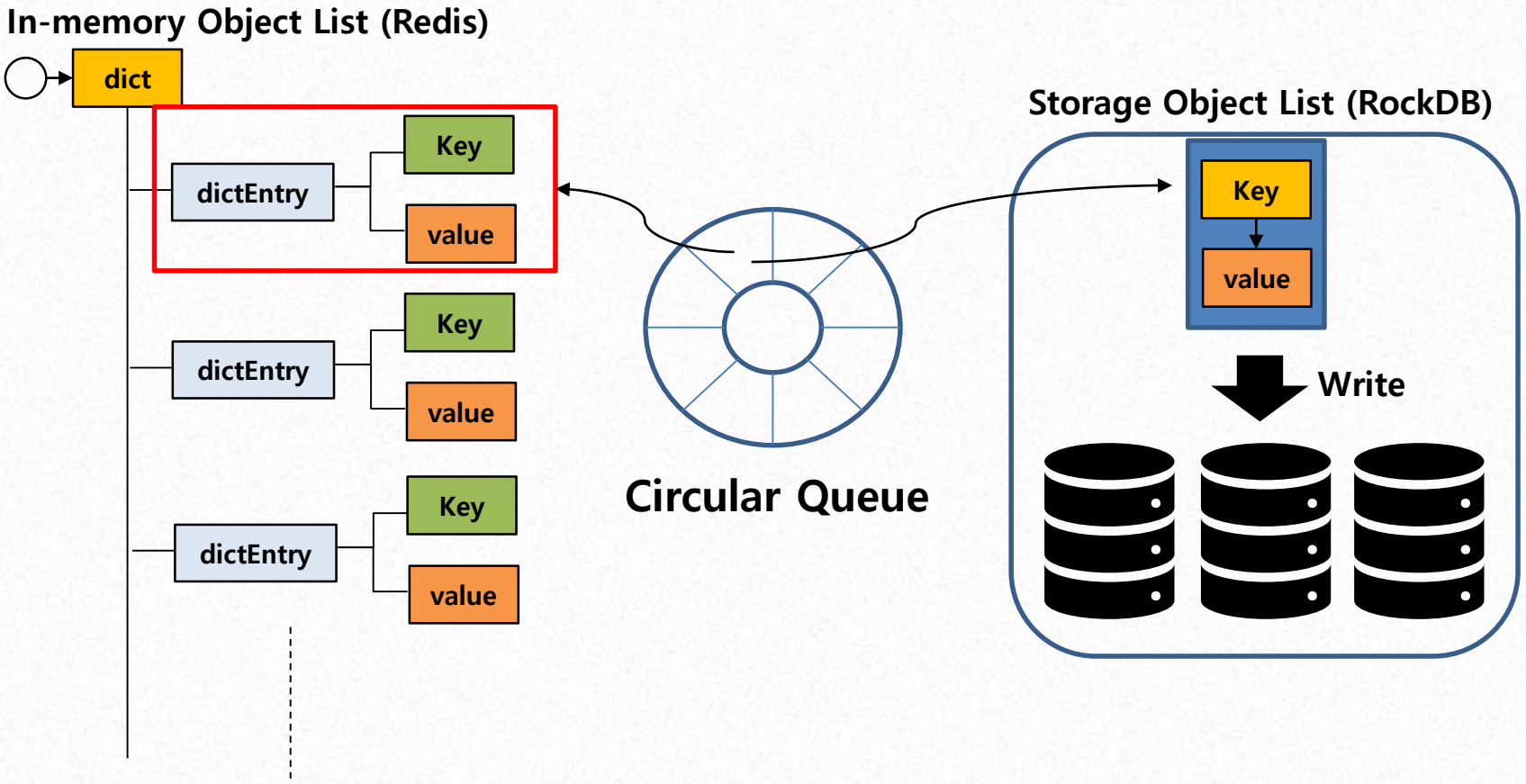


ADDB 프로젝트 홈페이지

02 | 2차년도 주요 수행내역

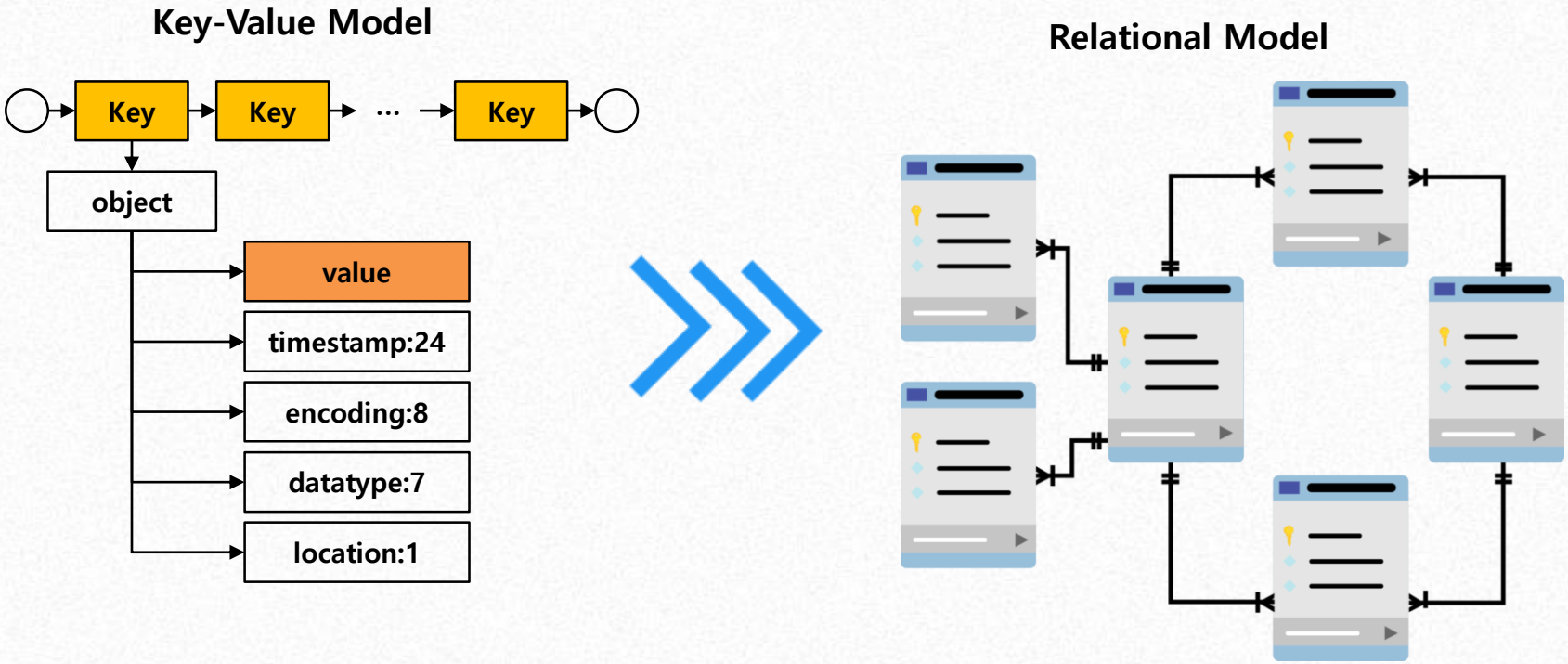
2차년도 주요 수행내역

- Circular Queue 설계
  - Eviction 수행 시, Eviction 대상이 중복 선택되는 문제 발생
  - 데이터가 적재된 순서대로 Eviction 수행



2차년도 주요 수행내역

- Redis 내 관계형 모델 자료구조
  - Key-Value 형태의 저장 기반을 관계형 모델로 변경
  - Key-Value Store의 빠른 저장 속도를 활용하여 데이터를 적재



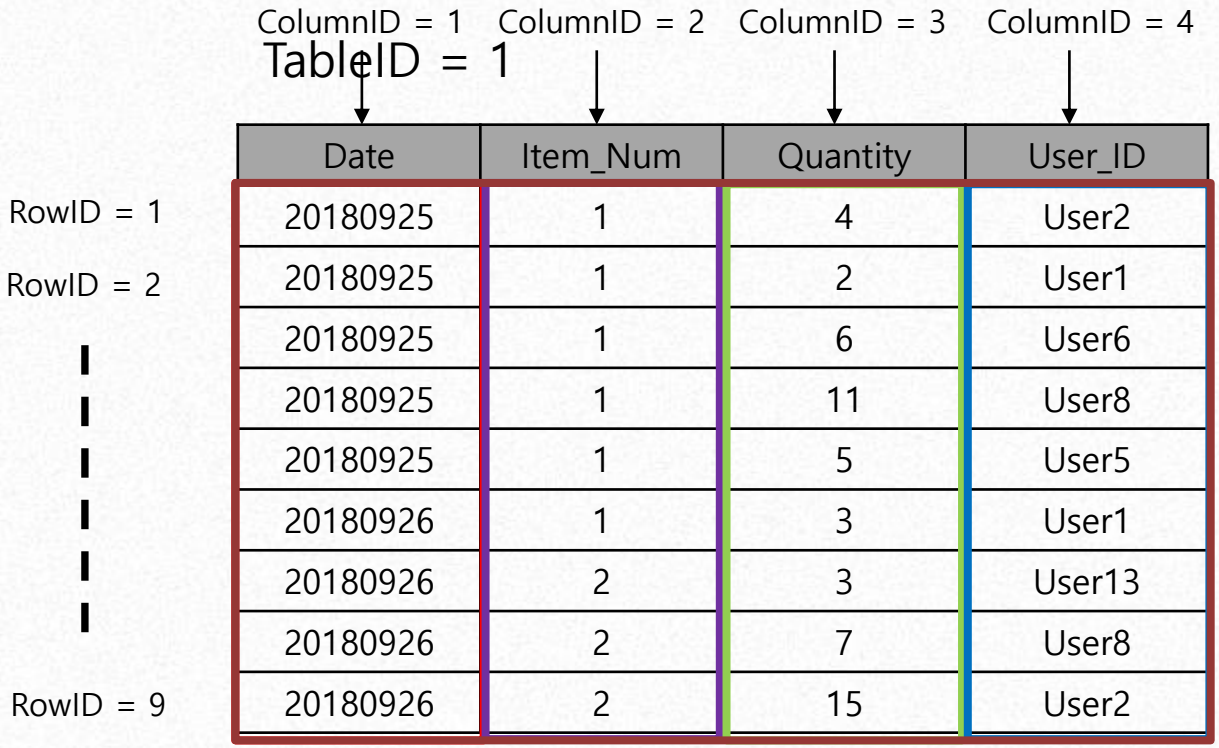
2차년도 주요 수행내역

- 관계형 모델 설계

Date	Item_Num	Quantity	User_ID
20180925	1	4	User2
20180925	1	2	User1
20180925	1	6	User6
20180925	1	11	User8
20180925	1	5	User5
20180926	1	3	User1
20180926	2	3	User13
20180926	2	7	User8
20180926	2	15	User2

2차년도 주요 수행내역

- 관계형 모델 설계
  - Relational Model의 데이터를 value단위로 구분
  - Row : Column Field로 데이터의 위치를 파악

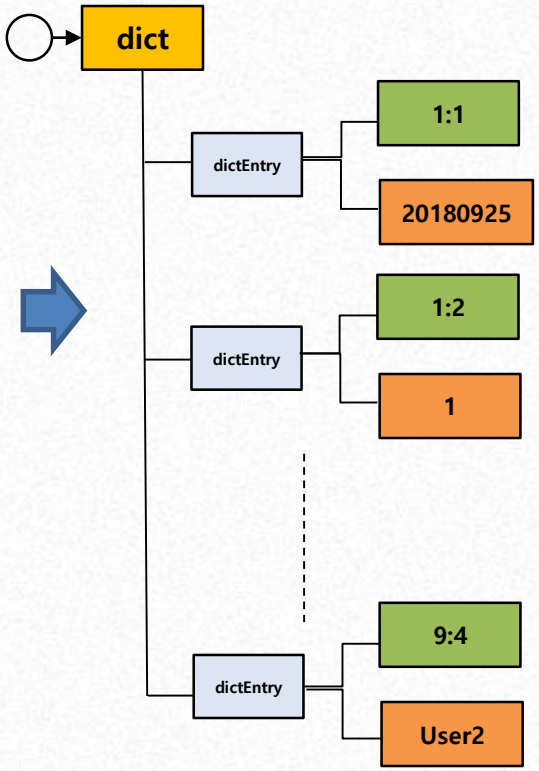


2차년도 주요 수행내역

- 관계형 모델 설계
  - Row : Column Field로 데이터의 위치를 파악
  - 빈번한 데이터 Access 발생

TableID = 1

	ColumnID = 1	ColumnID = 2	ColumnID = 3	ColumnID = 4
	Date	Item_Num	Quantity	User_ID
RowID = 1	1:1	1:2	1:3	1:4
RowID = 2	2:1	2:2	2:3	2:4
	3:1	3:2	3:3	3:4
	4:1	4:2	4:3	4:4
	5:1	5:2	5:3	5:4
	6:1	6:2	6:3	6:4
	7:1	7:2	7:3	7:4
	8:1	8:2	8:3	8:4
RowID = 9	9:1	9:2	9:3	9:4



2차년도 주요 수행내역

- 관계형 모델 설계
  - Access 빈도를 줄이고 데이터 관리를 편하게 하기 위해 Row를 Group단위로 구분

	Date	Item_Num	Quantity	User_ID
RowGroup = 1	20180925	1	4	User2
	20180925	1	2	User1
	20180925	1	6	User6
RowGroup = 2	20180925	1	11	User8
	20180925	1	5	User5
	20180926	1	3	User1
RowGroup = 3	20180926	2	3	User13
	20180926	2	7	User8
	20180926	2	15	User2

2차년도 주요 수행내역

- 관계형 모델 설계
  - TableID
  - RowID
  - ColumnID
  - RowGroupID

TableID = 1

	ColumnID = 1	ColumnID = 2	ColumnID = 3	ColumnID = 4
	Date	Item_Num	Quantity	User_ID
RowGroup = 1	20180925	1	4	User2
	20180925	1	2	User1
	20180925	1	6	User6
RowGroup = 2	20180925	1	11	User8
	20180925	1	5	User5
	20180926	1	3	User1
RowGroup = 3	20180926	2	3	User13
	20180926	2	7	User8
	20180926	2	15	User2

2차년도 주요 수행내역

- 데이터 Partitioning
  - 테이블 크기 증가에 따른 성능 이슈 발생
  - 관련성이 높은 Column value를 그룹화
    - 가용성 – 전체 데이터 훼손 가능성 감소
    - 관리용이성 – Table 관리 용이
    - 성능

Date	Item_Num	Quantity	User_ID
20180925	1	4	User2
20180925	1	2	User1
20180925	1	6	User6
20180925	1	11	User8
20180925	1	5	User5
20180926	1	3	User1
20180926	2	3	User13
20180926	2	7	User8
20180926	2	15	User2

PartitionInfo : 2:1

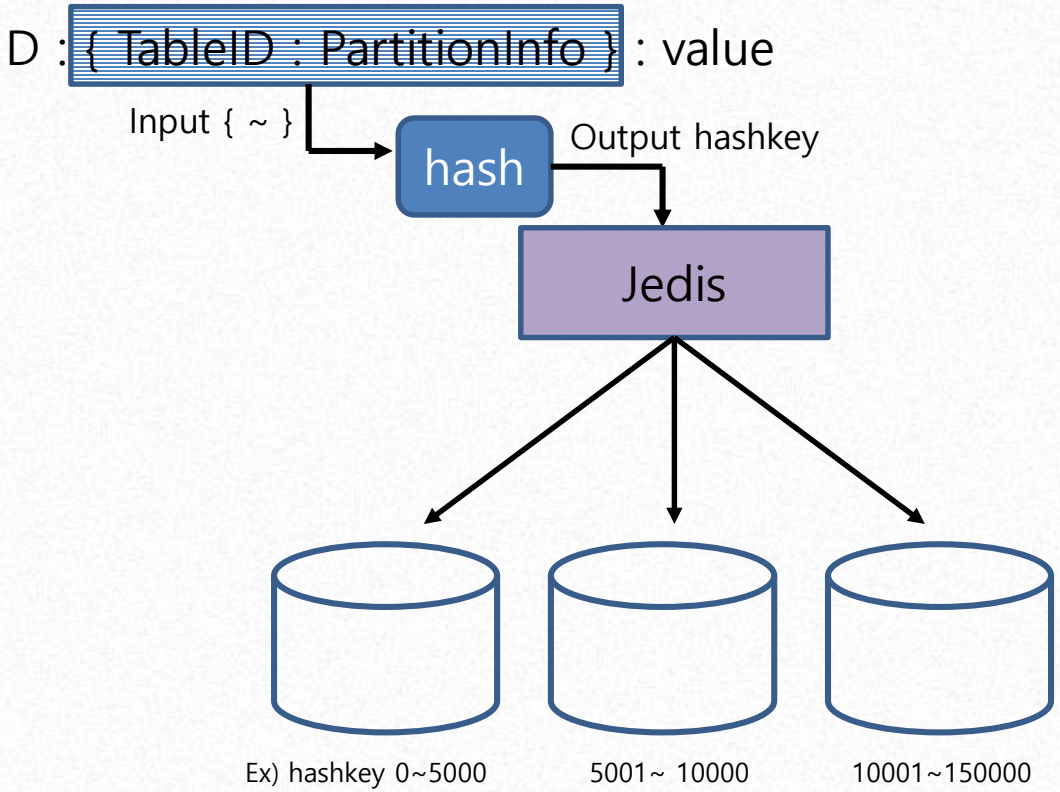
Date	Item_Num	Quantity	User_ID
20180925	1	4	User2
20180925	1	2	User1
20180925	1	6	User6
20180925	1	11	User8
20180925	1	5	User5
20180926	1	3	User1

PartitionInfo : 2:2

Date	Item_Num	Quantity	User_ID
20180926	2	3	User13
20180926	2	7	User8
20180926	2	15	User2

2차년도 주요 수행내역

- 데이터 Partitioning
  - 한 Redis Node에 데이터가 몰리지 않도록 분배

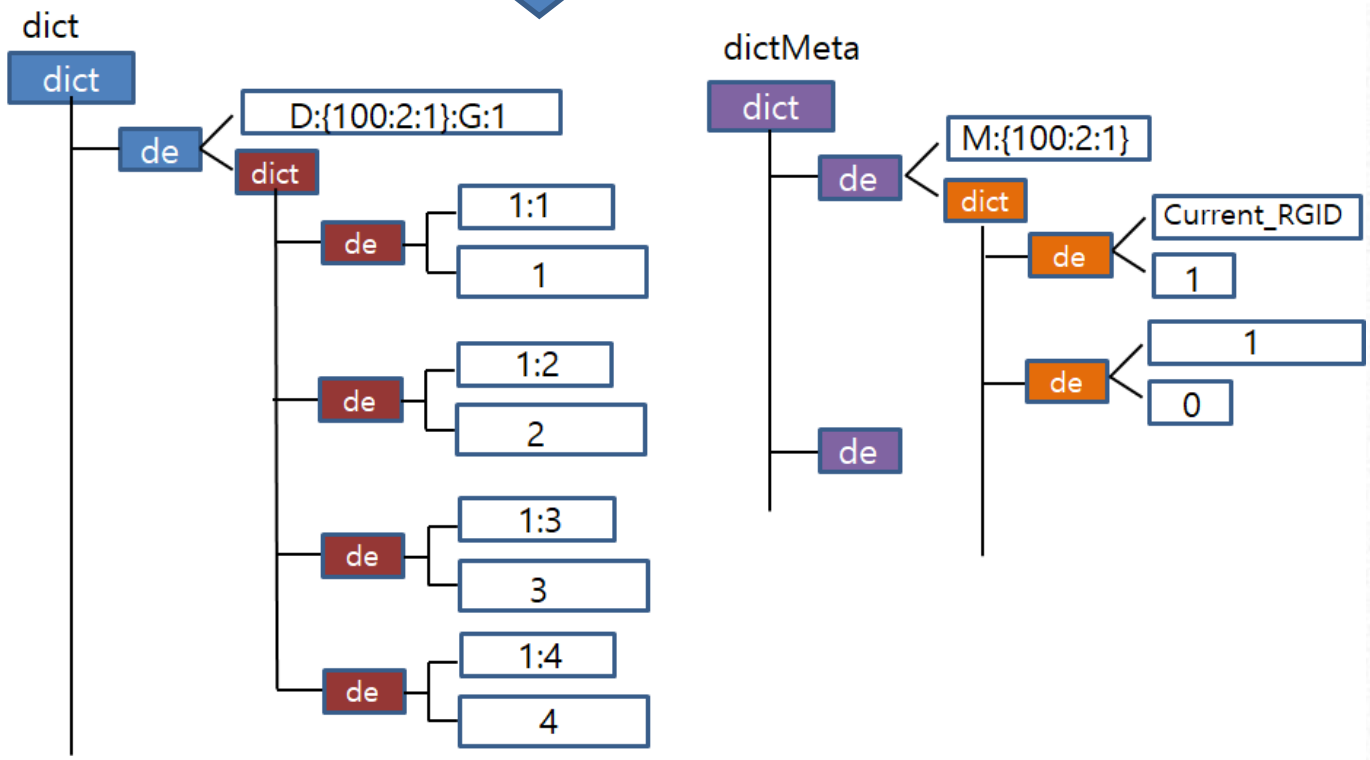


2차년도 주요 수행내역

- Redis 관계형 모델 구조

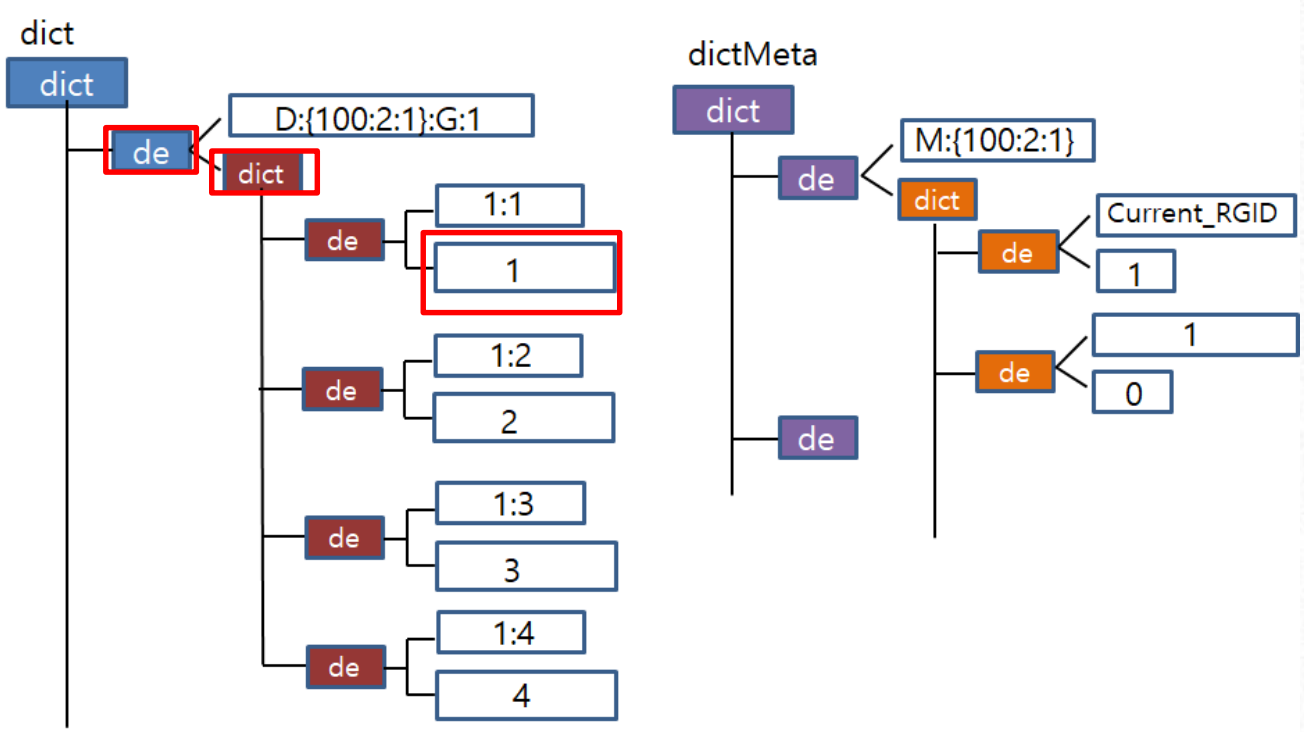
Column1	Column2	Column3	Column4
1	2	3	4

FPWRITE D:{100:2:1}:G:1 2:1 4 0 1 2 3 4



2차년도 주요 수행내역

- 초기 관계형 모델의 문제
  - 과도한 메모리 사용

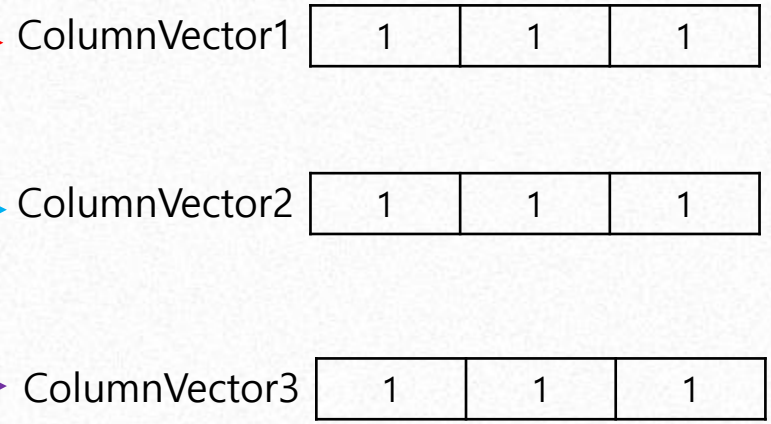


  - 데이터 저장에 필요한 부가적인 요소로 인하여 메모리 사용량이 급격히 증가
    - dictEntry – 24B
    - Redis object – 16B

2차년도 주요 수행내역

- Vector를 사용한 관계형 모델 개선
  - 저장된 데이터 타입을 고려하여 Column 단위 Vector를 적용
  - 생성되는 dictEntry와 Redis Object 수 감소

Date	Item_Num	Quantity	User_ID
20180925	1	4	User2
20180925	1	2	User1
20180925	1	6	User6
20180925	1	11	User8
20180925	1	5	User5
20180926	1	3	User1
20180926	2	3	User13
20180926	2	7	User8
20180926	2	15	User2



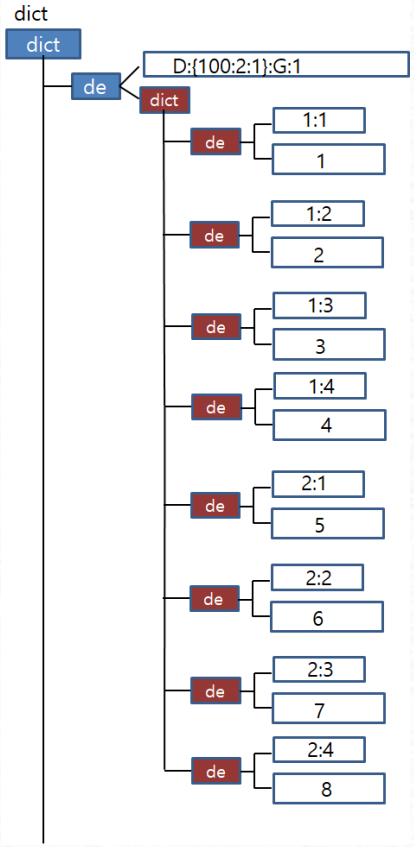
- ADDB 설정에서 Vector의 크기 및 RowGroup의 크기를 변경 가능하도록 설정

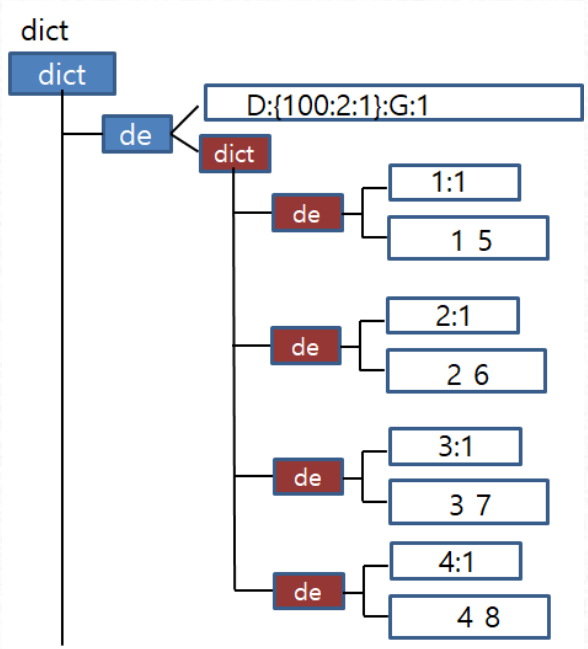
2차년도 주요 수행내역

- Vector를 사용한 관계형 모델 개선

Column1	Column2	Column3	Column4
1	2	3	4
5	6	7	8

ColumnID : VectorID

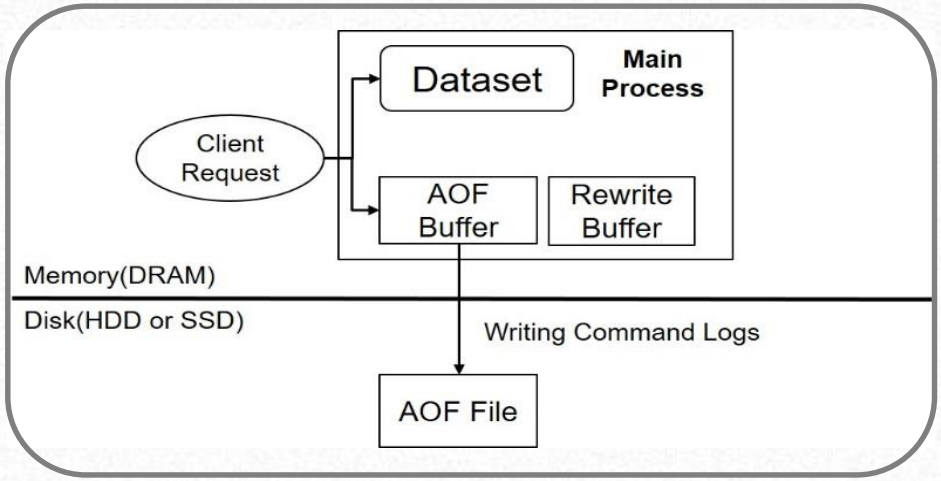




2차년도 주요 수행내역

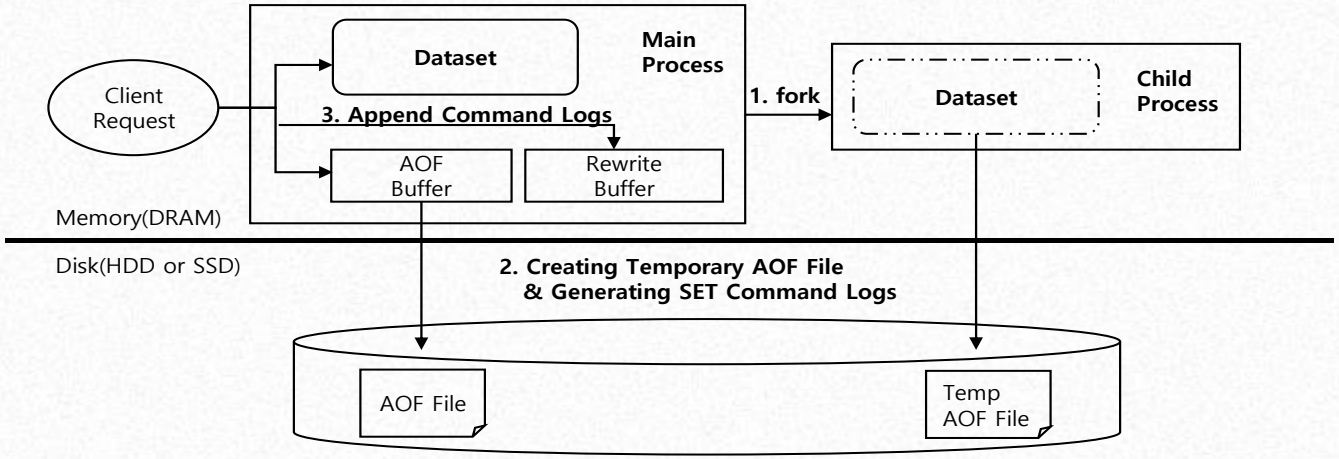
- 인메모리 로깅 개선
  - Redis persistence method
    - Snapshot(RDB)
    - Append Only File(AOF)
      - AOF_always
      - AOF_everysec
      - AOF_no

    - AOF Procedure



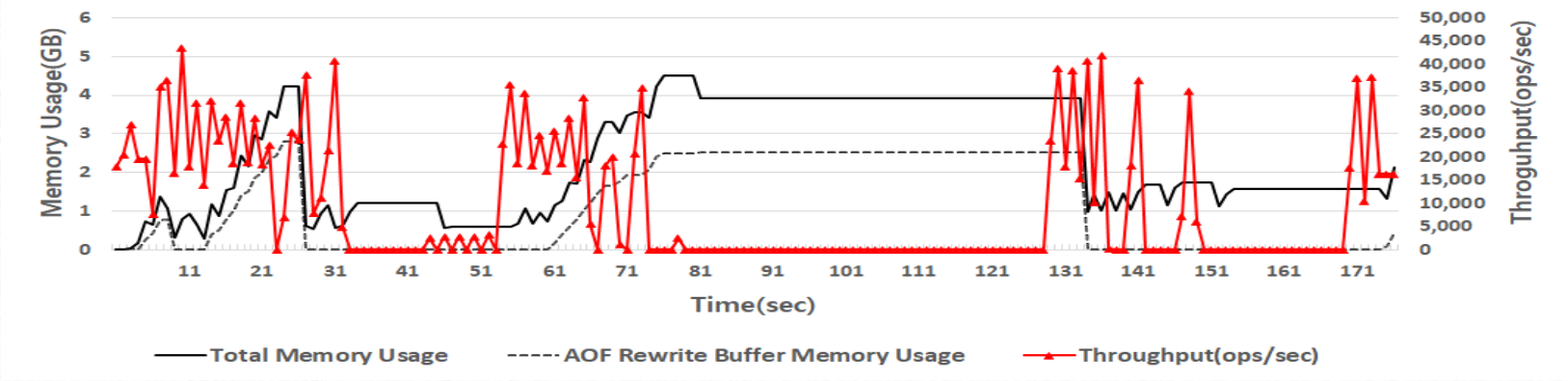
2차년도 주요 수행내역

- 인메모리 로깅 개선
  - AOF 방법의 문제
    - 데이터가 지속적으로 입력되면 AOF 파일 크기 증가
    - 성능 저하 및 리커버리 속도 저하 발생
    - AOF Rewrite
      - AOF 파일의 크기를 줄이기 위한 재구축 방법



2차년도 주요 수행내역

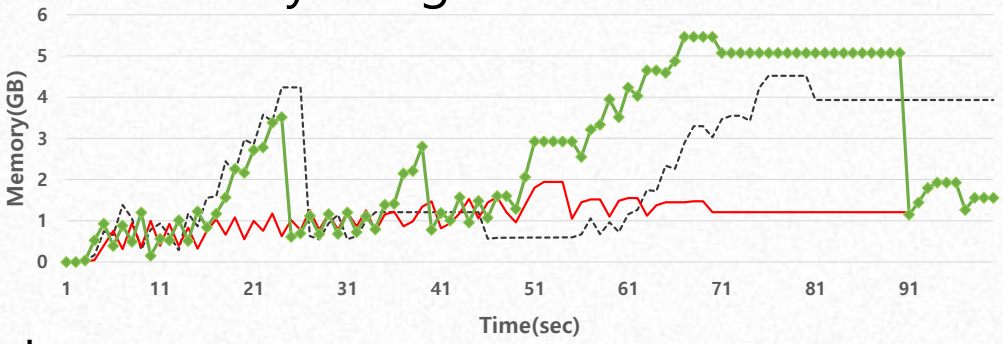
- 인메모리 로깅 개선
  - AOF Rewrite 중에 성능 저하 및 과도한 메모리 사용 발생



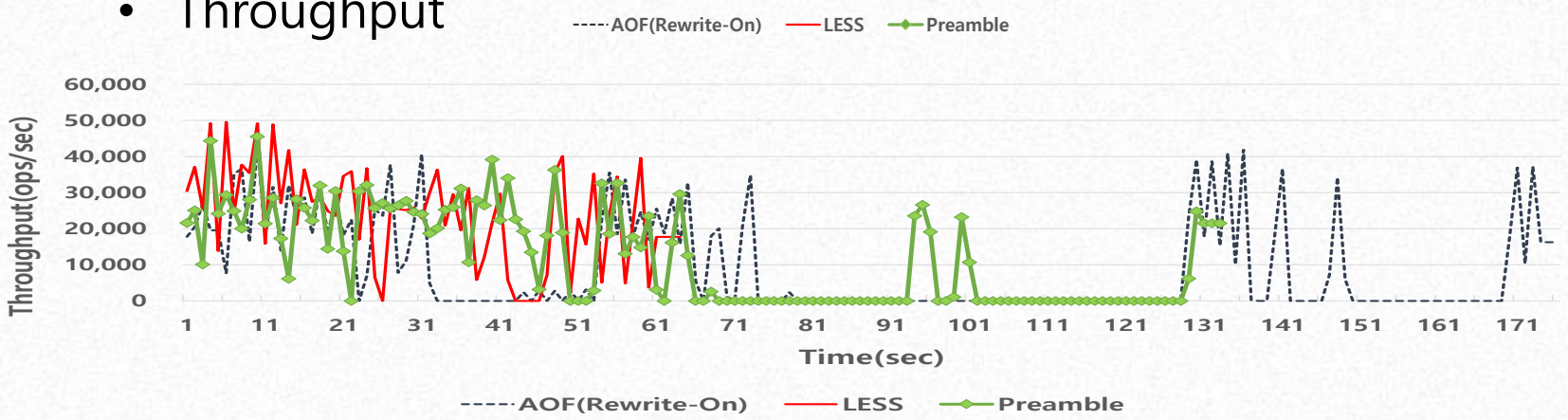
2차년도 주요 수행내역

- 인메모리 로깅 개선
  - 새로운 로깅 방법을 적용하여 성능은 **2.6**배 향상
    최대 메모리 사용량은 **57**% 감소

  - Maximum Memory Usage



  - Throughput



Thank You