3장. 레디스 기본 개념

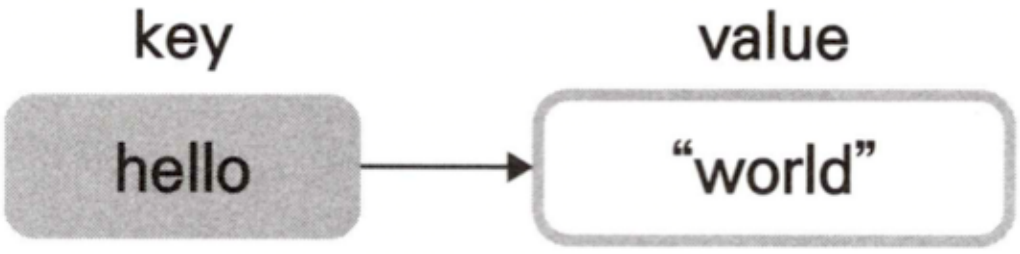
- **55~90**
- 키-값 형태의 데이터 저장소인 레디스의 자료구조에 대해 알아보자
- 레디스에서는 다양한 자료구조를 지원하며 어떻게 사용하는지 알아보자

TL;DR

- 특이하게 Redis 는 인덱스 칠 때 포함으로 검색됨

레디스 자료구조

String



- 최대 512MB
- Binary-safe 하게 처리되어 이ㅁ지 같은 바이트 값 등 다양한 데이터 저잦ㅇ 가능
- key-아이템 1대1 연결되는 유일한 자료 구조 → 나머지 자료 구조를 보면 된다

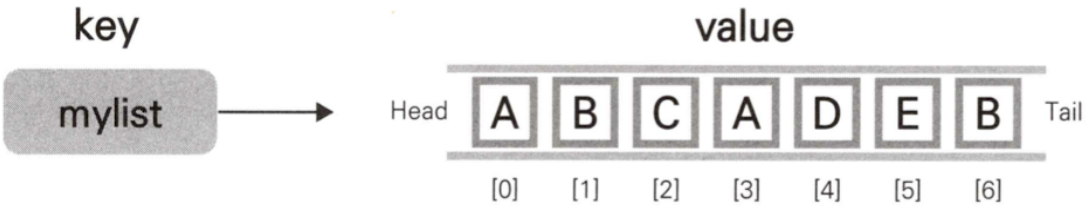
커맨드

- SET
  - Options
    - NX: 키 없는 경우

        - XX: 키 이미 있는 경우
- GET
- INCR, INCRBY, DECR, DECRBY
    - 원자적으로 처리 가능
        - Race condition 방지
        - 타이밍 순서에 따라 요청 무시됨

```
SET hello wolrd

GET hello
```

List



- 순서를 가지는 문자열 목록
- 최대 12억개
- 스택, 큐로 사용된다
- 특징
    - LPUSH, RPUSH, LPOP, RPOP 커맨드는 당연 O(1)
    - 인덱스 들어가면 O(n)

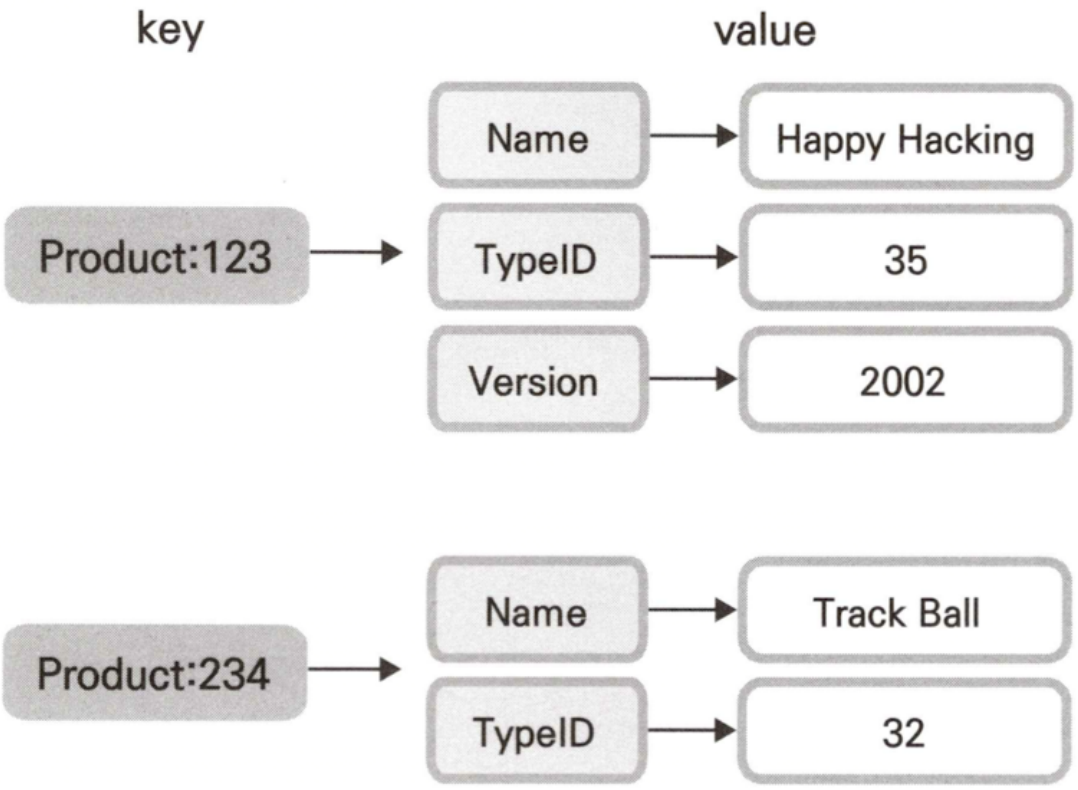
커맨드

- 추가
    - LPUSH, RPUSH
    - LINSERT
        - 데이터 앞 뒤에 데이터 추가
            - option

        - BEFORE
        - AFTER
    - LSET
        - 신규 입력 데이터로 덮어 씀
- 조회
    - LRAGE,
        - key index index
    - LINDEX
        - 인덱스로 원하는 데이터 확인
- 삭제
    - LTRIM
        - 포함된 영역 말고 모두 삭제
        - 로그 개수 유지할 때 사용할 수 있음
    - LPOP

```
LPUSH key A

LRANGE key 0 3

LTRIM key 0 999

LINDEX key 3

LPOP
```

Hash



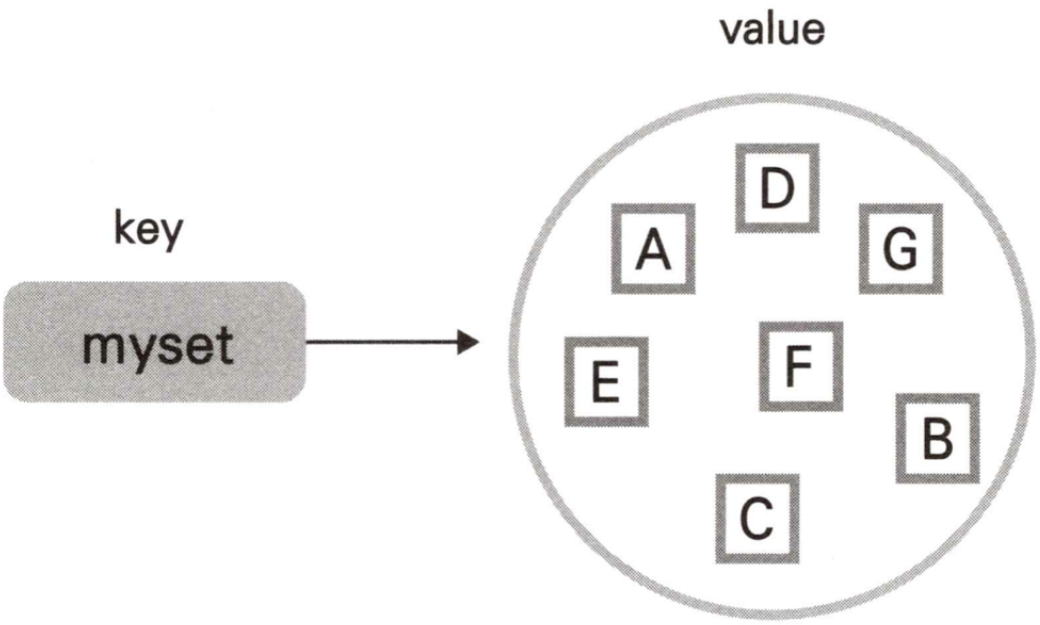
- key 하나가 여러 쌍을 가진 아이템의 집합
- 필드는 하나의 Hash 내에서 유일

커맨드

- 추가
  - HSET
- 조회
  - HGET
    - 필드 하나
  - HMGET
    - 여러 필드
  - HGETALL
    - 전부

```
HSET Prodct:123 Name "HI" TypeID 32

HMGET Product:234 Name TypeID
```

Set



- 정렬되지 않은 문자열 모음
- 중복 저장 X, 교집합, 합집합, 차집합 등 집합 연산 커맨드 제공

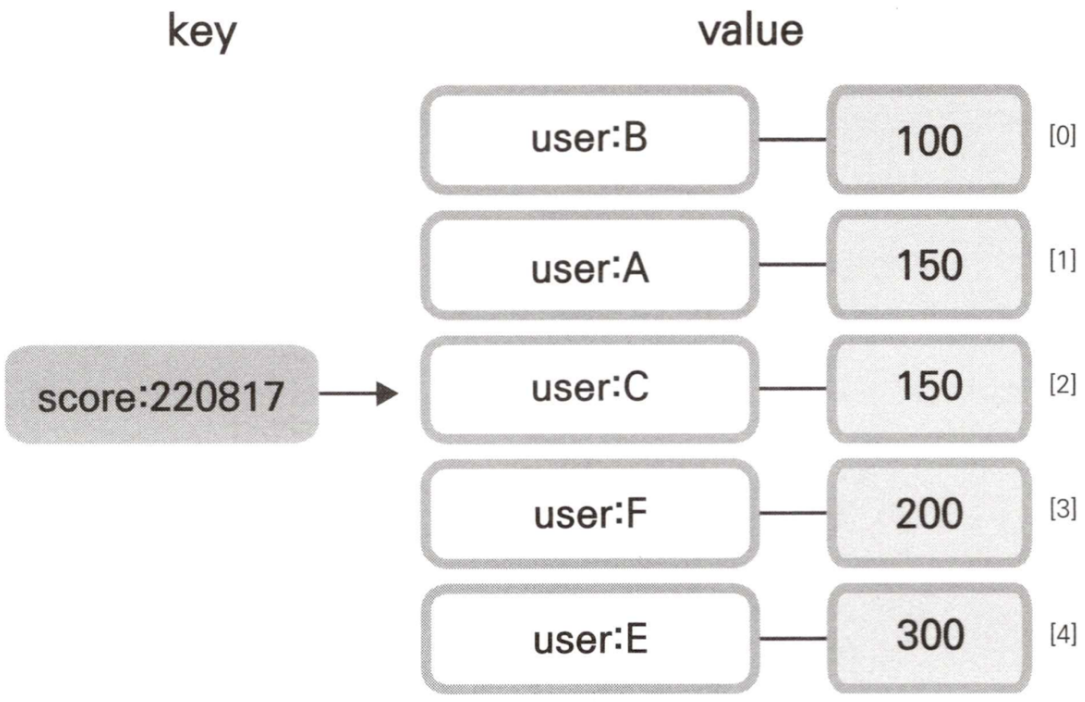
커맨드

- 추가
  - SADD
- 조회
  - SMEMBERFS
- 제거

    - SPOP
        - 랜덤 삭제 반환
- 집합 연산
    - SUNION
    - SINTER
    - SDIFF

```
SADD myset A A C -> A, C 만 저장

SMEMBERS myset
```

Sorted Set

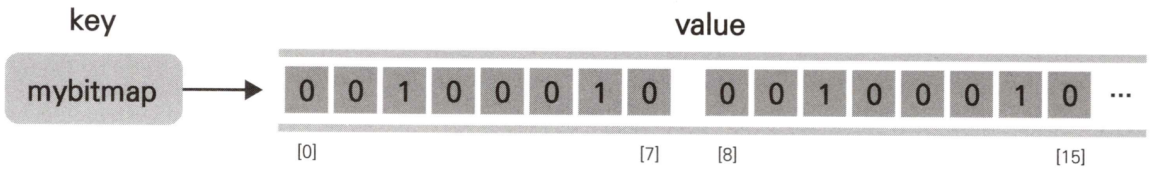


- Score 값에 따라 정렬되는 고유한 문자열 집합
- 모든 아이템은 스코어-값 쌍을 가진다.
- 저장, 수정 시 정렬되어 저장 됨

- 특징
  - 중복 없음
  - 스코어에 연결되어 hash와 유사
  - 인덱스 이용해 접근할 때는 list 보다 sorted list 가 빠르다.
- 커맨드
  - 등록
    - ZADD
      - 이미 있으면 스코어 업데이트
      - optinal
        - XX, NX, LT, GT
  - 조회
    - ZRANGE
      - ZRANGE key start stop [BYSCORE | BYLEX] [REV] [LIMIT OFSET COUNT] [WITHSCORE]
        - 조건
          - BYSCORE: 점수
          - BYLEX: 사전순서
            - (: 포함
            - [: 미포함
        - REV: 반전

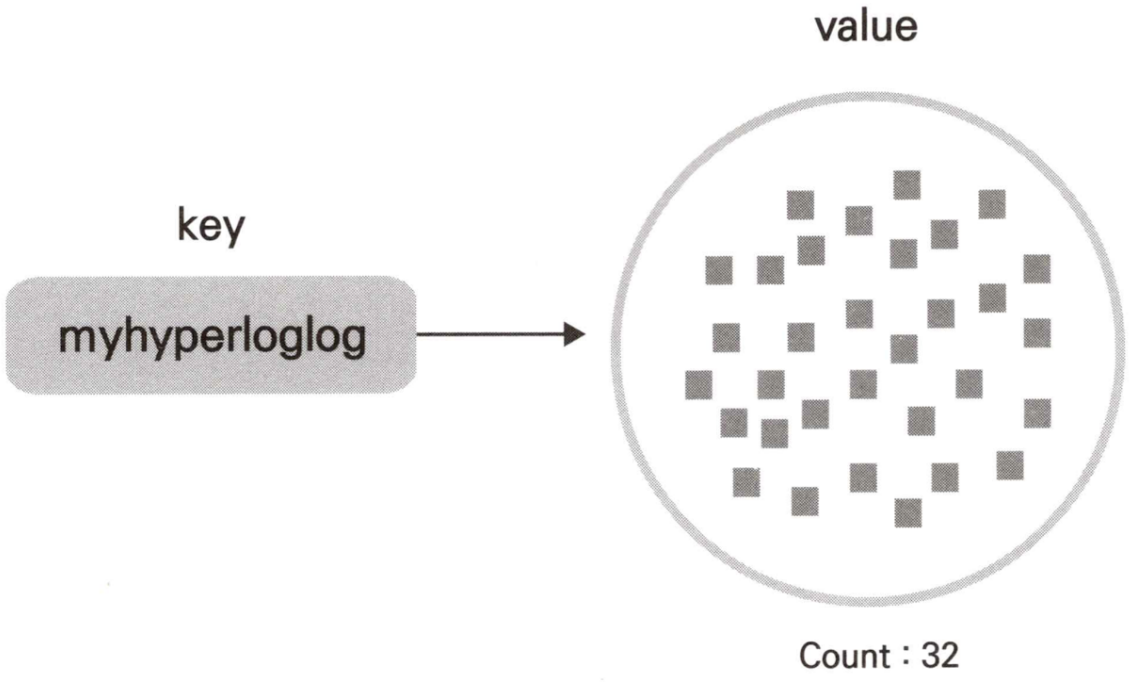
```
ZADD score:220817 100 user:B

ZRANGE mySortedSet (b, (f BYLEX
```

비트맵



- string 자료 구조에 bit 연산 수행할 수 있도록 확장한 형태
- 512MB 값 저장 가능한 구조
- 커맨드
  - SETBIT
  - GETBIT

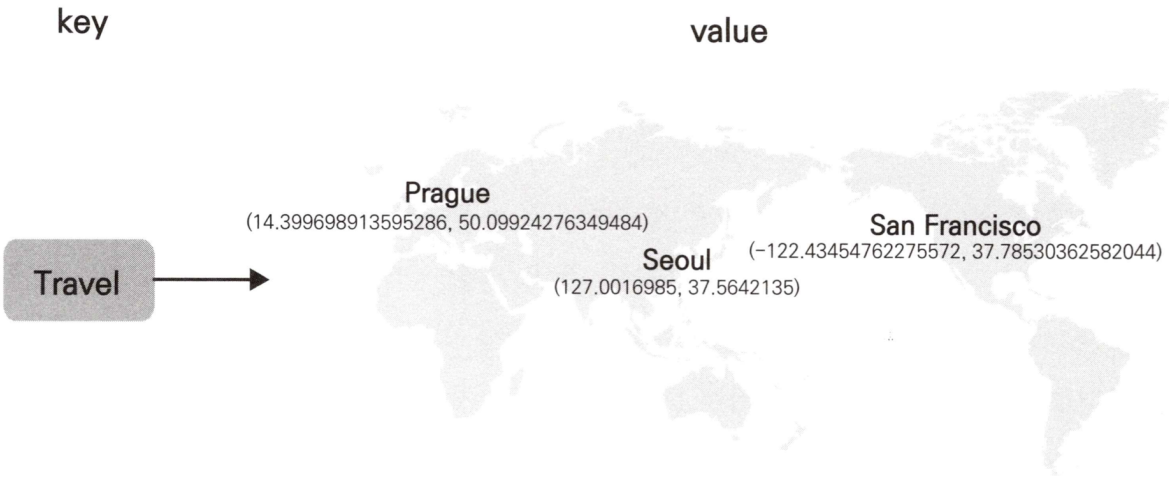
Hyperloglog



- 카디널리티 추정 자료 구조
- set과 달리 실제 데이터를 저장하지 않기 때문에 개수에 구애받지 않는다
- 최대 12KB(2^64개), 최대 추정오차는 최대 0.81%
- 커맨드

- PFADD
- PFCOUNT

Geospatioal

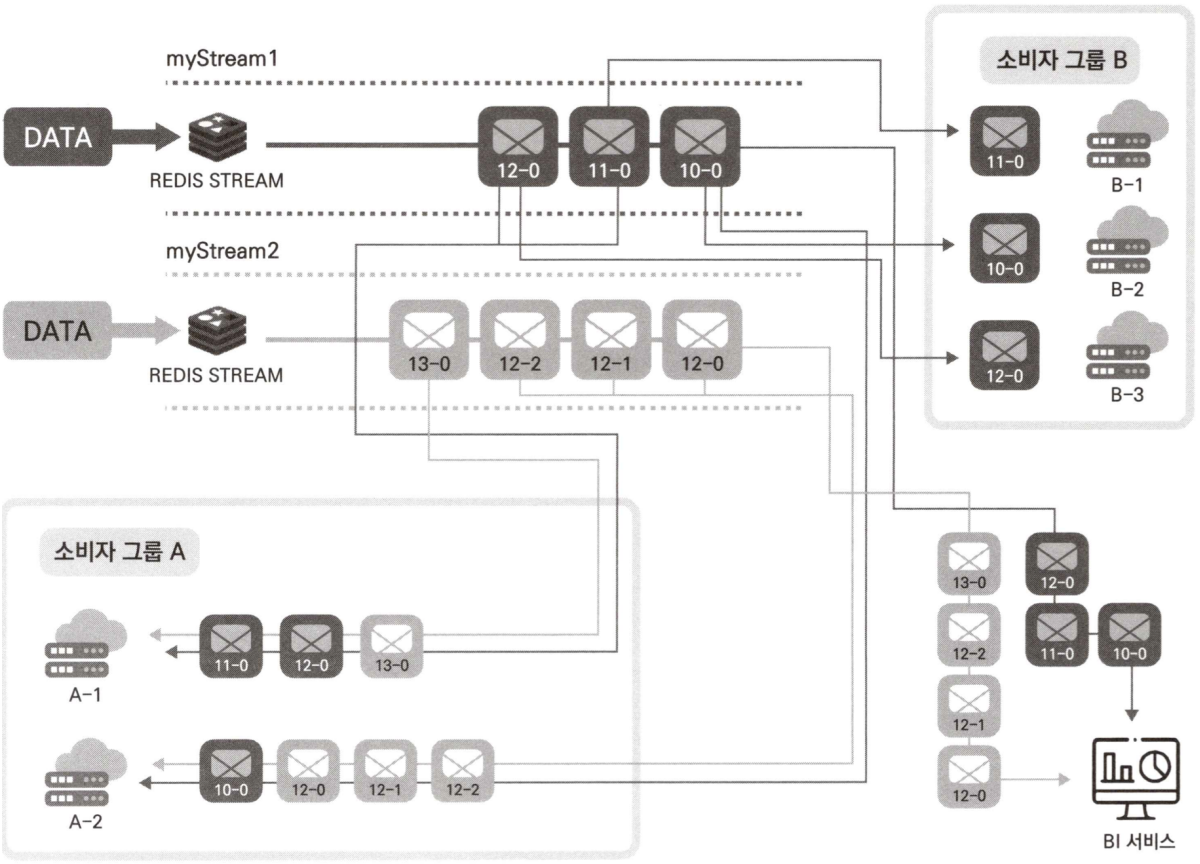


- 경도, 위도 데이터 쌍의 집합으로 간편하게 지리 데이터 저장할 수 있는 방법
- 커맨드
  - GEOADD
    - 추가
  - GEOPOS
    - 저장된 위치 데이터 조회 가능
  - GEOSEARCH
    - 거리 내에 있는 아이템 검색 가능
    - BYRADIUS: 반경 거리 기준
    - BYBOX: 직사각형
  - GEODIST
    - 거리

```
GEOADD travel 14.3969 50.099242 prague

GEOPOS travel prague
```

```
GEODIST travel seoul prague KM
```

Stream



- Append-only 방식으로 데이터 계속 저장
  - 실시간 이벤트, 로그성 데이터 저장

레디스 키 관리 방법

키의 자동 생성과 삭제

1. 키가 존재하지 않을 때는 빈 자료 구조 생성한다.

    a. 다른 자료 구조에 저장 시도하면 에러
2. 모든 아이템 삭제하면 키는 삭제된다
    a. stream 예외
3. 키 없는 상태에서 조회 읽기 전용 커맨드 수행하면 키 있는데 아이템 없는 것처럼 동작
    a. 키 삭제, 아이템 삭제, 조회

키 관련 커맨드

Exists

- 키 존재 확인
- 존재 1, 없음 0

Keys

- 모든 키 조회
- 글롭 패턴 스타일 동작
- 싱글 스레드인 레디스에서 잘 쓰지 않음

Scan

- KEYS 대체, 커서 기반으로 조금씩 조회
- SCAN 0
- 10개가 기본이지만 메모리 스캔해서 키 더 읽으면 효율적인 경우 더 읽어서 줌
- options
    - MATCH
    - TYPE

```
SCAN 0 MATCH *11* TYPE zset
```

SORT

- 키 내부 아이템 정렬해 반환

RENAME / RENAMENX

- 키의 이름을 변경하는 커맨드

COPY

- 지정된 키를 destination 키에 복사
    - 이미 있으면 에러 → REPLACE 옵션

```
COPY source destination
```

Type

- 자료 구조 반환

Object

- 키 상세 정보 반환
- 키가 내부적으로 어떻게 저장됐는지, 호출되지 않은 시간

키 삭제

FLUSHALL

- 모든 키 삭제
- 동기방식
- async 는 비동기

DEL

- 키와 킹에 저장된 모든 아이템 삭제
- 동기적
- DEL key

Unlink

- DEL 과 비슷하지만 백그라운드 처리

- lazyfree-lazy-user-del yes 일 경우 DEL 도 UNLINK 로 동작

키의 만료

EXPIRE

- 키 만료 기간 설정

EXPRIREAT

- 유닉스타임스탬프 만료

EXPIRETIME

- 만료 기간 초 단위로 반환

TTL

- 키가 몇 초 뒤에 만료되는지 반환

밀리초

- PEXPIRE, PEXPIREAT... 은 밀리초