

# 게임 리더 보드 만들기

## 1 리더보드의 특성과 기능 요구사항

## 리더보드의 특성과 기능 요구사항

### 1.

리더보드의  
특성과 기능  
요구사항

### 리더보드(Leaderboard)

- 게임이나 경쟁에서 상위 참가자의 랭킹과 점수를 보여주는 기능
- 순위로 나타낼 수 있는 다양한 대상에 응용(최다 구매 상품, 리뷰 순위 등)

“그룹 상위 랭킹 또는  
특정 대상의 순위를  
보여준다.”

상위 랭킹

1위 A: 1500

2위 B:

1350

3위 C:

1200

.

.

.

내 랭킹 표시

256위 ...

257위 Me: 510

258위 ...

## 리더보드의 특성과 기능 요구사항

1.

리더보드의  
특성과 기능  
요구사항

### 리더보드의 동작(API 관점)

- 점수 생성/업데이트 => ex: SetScore(userId, score)
- 상위 랭크 조회(범위 기반 조회) => ex: getRange(1~10)
- 특정 대상 순위 조회(값 기반 조회) => ex: getRank(userId)

“빠른 업데이트/빠른 조회가 필요”

## 리더보드의 특성과 기능 요구사항

## 1.

리더보드의  
특성과 기능  
요구사항

## 데이터 구조와 성능 문제

- 관계형 DB 등의 레코드 구조를 사용했을 때

User	Score
A	1500
B	1350
C	1200
...	...

## &lt;업데이트&gt;

한 행에만 접근하므로 비교적 빠름.

Ex: `UPDATE ranking SET score = 1550 WHERE userId = A;`

## &lt;랭킹 범위나 특정 대상의 순위 조회&gt;

데이터를 정렬하거나 **COUNT()** 등의 집계 연산을 수행해야  
하므로 데이터가 많아질수록 속도가 느려짐.

Ex(상위 5개 출력): `SELECT userId FROM ranking ORDER BY score DESC LIMIT 0, 5;`

## 리더보드의 특성과 기능 요구사항

### 1.

리더보드의  
특성과 기능  
요구사항

### Redis를 사용했을 때의 장점

- 순위 데이터에 적합한 **Sorted-Set**의 자료구조를 사용하면 **score**를 통해 자동으로 정렬됨
- 용도에 특화된 오퍼레이션(**Set** 삽입/업데이트, 조회)이 존재하므로 사용이 간단함
- 자료구조의 특성으로 데이터 조회가 빠름(범위 검색, 특정 값의 순위 검색)
- 빈번한 액세스에 유리한 **In-memory DB**의 속도

# 게임 리더 보드 만들기

**2 Sorted Sets**을 이용한 리더보드 구현

## Sorted Sets을 이용한 리더보드 구현

2.

Sorted Sets을  
이용한 리더보드  
구현

### API

- GET /setScore?userId=A&score=10
- GET /getRank?userId=A
- GET /getTopRanks