Redis 캐시 적용 및 성능 개선 보고서

작성일: 2025-05-08

작성자: 김종완 (2팀)

---

1. 요약 (Executive Summary)

도입 목적(가정 시나리오)

급증하는 사용자 트래픽과 상품 데이터 증가로 인하여 데이터베이스 부하가 증가하는 상황을 가정하였습니다.
주기적으로 호출되는 인기 상품 조회 API를 제외하고 전체 API 중 상품 목록 조회 API가 전체 호출 비율의 70% 이상을 차지하는 상황에서

- 상품 목록 조회
- 인기 상품 조회

두 API에 대하여 데이터베이스 부하를 해소하고, 응답 성능을 개선하기 위해 Radis 캐시 도입의 경우 예상되는 성능 개선 효과를 분석하였습니다.

테스트 개요

- **시나리오:** 사용자 수(VU) 10명부터 100명까지 점진적으로 증가
- **대상 데이터:**
  - 상품 데이터: 300,000건
  - 판매 통계 데이터: 50,000건
- **테스트 대상 API:**
  - 상품 목록 조회 API: `/products?offset=0&limit=10000`
  - 인기 상품 조회 API: `/products/popular?limit=1000`

결론 요약

Redis 캐시 적용에 따라 두가지 API 모두 응답 시간 감소, 처리량 증가 등 유의미한 개선 지표를 확인하였습니다.

• 평균 응답 시간 최대 **67% 감소**

• 초당 처리량(TPS) 최대 **3배 증가**

• SLA(300ms 이하 응답) 만족 비율: **51% → 85% (API #1 기준)**

---

2. 문제 배경

현황 및 과제

- 전체 상품 수 30만 건 이상, 데이터 지속 증가
- 트래픽 증가로 인한 DB 부하 및 쿼리 처리 지연

API별 문제점

- **상품 목록 조회:** offset/limit 기반의 조회 구조로 요청 시마다 쿼리 수행. limit가 커질수록 요청 데이터의 양이 증기하여 심각한 성능 저하 우려
- **인기 상품 조회:** 상품 테이블과 통계 테이블 간 Join으로 인한 연산 부담. 실시간 통계 기능 도입시 짧은 주기로 동일한 쿼리를 지속적으로 수행해야함

---

3. 캐시 적용 설계

캐시 Key 구조

- 상품 목록 조회: `products:list:{offset}:{limit}`
- 인기 상품 목록 조회: `products:rank:{limit}`

TTL 및 무효화 정책

- 상품 목록:
  - TTL 1분
  - 상품 정보(상품명, 가격, 재고) 변경시 Evict
- 인기 상품:
  - 매일 새벽 1시 통계 갱신 스케줄러 완료 시 캐시 Overwirte

적용 방식

- Cache miss 발생 시 DB 조회 후 캐시 저장

---

4. 성능 테스트 개요

테스트 환경

- Redis: Standalone
- DB: 300,000건의 상품이 삽입된 MySQL 샘플 DB
- 테스트 도구: K6

측정 지표

- 평균 응답 시간
- TPS (초당 처리량)
- SLA (응답 시간 기준)
- DB 쿼리 수

---

5. 성능 비교 요약

항목	상품 목록 조회 (API #1)	인기 상품 조회 (API #2)
성공률	72.46% → **96.03%** (+23.57%)	99.78% → **99.97%** (+0.19%)
평균 응답시간(ms)	604ms → **250ms** (-59%)	177ms → **59ms** (-67%)
p95 응답시간(ms)	1.23s → **738ms** (-40%)	368ms → **186ms** (-49%)
SLA 만족률	51% → **85%** (+34%)	10% → **83%** (+73%)
TPS	77.7 → **186.6** (2.4배)	263 → **791** (3배)
최대 응답시간	1.96s → 9.61s (일부 outlier)	776ms → 1.01s (일부 outlier)

---

6. 분석 인사이트

API #1 (상품 목록 조회)

캐시 적용 이전 DB에 대한 의존도가 높고 데이터의 갯수가 많아 기대 SLA(300ms)에 미달된 요청(51% 충족)이 다수 발생하였습니다. 캐시 적용 이후 전체 요청에 대하여 85%가 기대 SLA를 충족하였으며 응답 속도 감소, TPS 약 2.4배 증가등을 확인할 수 있었습니다.

API #2 (인기 상품 조회)

조회가 필요한 데이터의 양이 많지 않아 캐시 미적용 상태에서도 성능은 양호하였으나, 캐시 도입 이후 요청 처리량에서 3배 이상 증가된 지표를 확인하였습니다. 트래픽이 급증하는 상황에 대하여 대응 방안으로 캐시 도입 검토가 충분히 가능할것같습니다.

---

7. 결론 및 제안

결론

본 테스트는 Redis 캐시의 성능 개선 효과를 확인하기 위해 요청량과 데이터 크기를 인위적으로 증폭한 테스트 환경에서 수행하였습니다.

API 요청 조건을 일반적인 수준으로 조정한 경우, 캐시 없이도 충분히 빠른 응답 시간 확보 가능합니다

- 상품 목록 조회 (offset ≤ 100): 평균 70ms
- 인기 상품 조회 (3건 요청 시): 평균 30ms

그러나 TPS 증가 등 **처리량 확보** 측면에서는 Redis 캐시 도입 효과가 뚜렷하게 확인되었습니다.

제안

- **현 시점에서는 즉각적인 Redis 캐시 도입은 필요하지 않습니다.**
  성능 개선을 위한 대안으로
  offset 제한 설정 및 SLA 목표 수립 후 지속적인 모니터링으로 현재 필요한
  성능을 달성할 수 있습니다.

예) 상품 목록 조회 SLA 70ms, 인기 상품 조회 SLA 30ms

- **향후 오류 응답률이 10% 이상 발생할 경우 Redis 캐시 도입을 적극적으로 고려해볼 필요가 있습니다.**