



메모리 & 캐시



목차



메모리

컴퓨터 메모리의 종류

RAM



캐시

캐시의 종류

L1 캐시

L2 캐시

L3 캐시

사용 예시

캐시 사용 과정



캐싱 전략

1. Cache Aside

장점

단점

2. Read-Through

3. Write-Through

장점

단점

4. Write-Around

5. Write-Back

장점

단점



하드디스크



참고

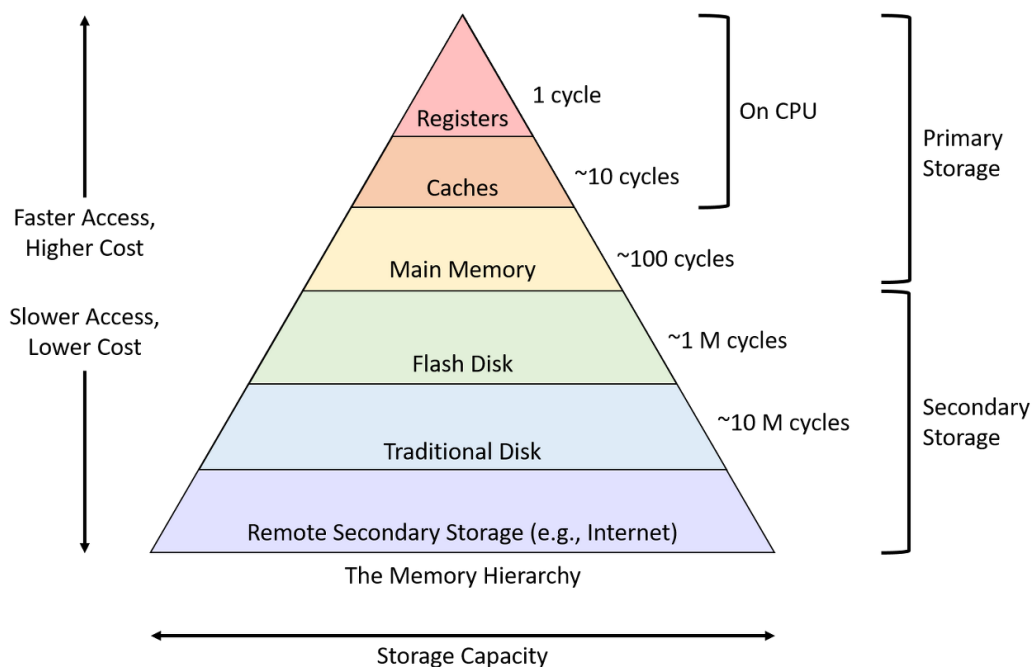


메모리

컴퓨터 메모리의 종류

- 레지스터, 캐시, RAM을 보통 주 기억장치라고 한다

- 프로그램이 실제로 구도오될 때 이 세 기억장치를 사용하기 때문
- **레지스터**는 제일 빠른 메모리, CPU 계산과정의 일부로 작동
- **캐시 메모리**는 레지스터 다음으로 빠른 메모리
 - L1, L2, L3 캐시 등 여러 단계로 나뉘어 짐
 - 숫자가 작을수록 용량이 작고 빠르며 숫자가 클 수록 용량이 크고 느림
- **RAM** 역시 빠르지만 Cache보단 느리다
 - 그렇지만 SSD, HDD에 비해서는 월등히 빠름
 - 일반적으로 사용하는 **로딩**이라는 단어가 하드 디스크에서 데이터를 읽어 램으로 전송하는 과정을 의미



RAM

- 메인 메모리 = 주기억장치 = RAM
- RAM은 Random Access Memory의 약자이다



RAM은 왜 *Random* Access Memory일까?

- 접근 방식에는 크게 두가지가 존재
 - 임의 접근 방식과 순차 접근 방식
- **순차 접근 방식**은 만약 A라는 프로그램을 이용해야 하는데, A라는 프로그램이 170번 메모리에 있을 때 1부터 170번까지 A 프로그램을 찾게 된다
- **임의 접근 방식**은 170번 주소를 던져주면 바로 찾아오게 된다
 - 일종의 배열에 접근하는 방식과 유사

- RAM은 DRAM과 SRAM이 있는데 주 기억장치는 주로 DRAM을 의미한다



DRAM과 SRAM의 차이 (참고만)

DRAM(Dynamic RAM)

- Capacitor로 작동
- 시간이 지나면 스스로 방전이 됨
- 시간의 흐름에 따라 자연적으로 메모리가 변화한다는 뜻

SRAM(Static RAM)

- Flip-Flop으로 작동
- Flip-Flop은 전류신호가 오기 전에는 상태가 변화하지 않는 소자
- 가만히 냅두면 내용이 소멸, 변화하지 않음

- 컴퓨터의 CPU가 현재 처리중인 데이터나 명령만을 일시적으로 저장하는 휘발성 메모리
 - 전원이 꺼지면 메인 메모리에 저장된 내용들은 모두 사라진다
 - 따라서 컴퓨터가 꺼진 이후에도 데이터를 유지하고 싶을 경우 데이터를 하드디스크에 저장해야 한다
- 보조기억장치보다 접근속도가 빠르다



주 기억장치 vs 보조 기억장치

주 기억장치

- 컴퓨터 내부에서 현재 CPU가 처리하고 있는 내용을 저장하고 있는 기억 장치
- 비교적 용량이 크고 처리 속도가 빠름
- CPU의 명령에 의해 기억된 장소에 직접 접근하여 읽고 쓸 수 있음
- ROM과 RAM으로 나뉘어진다

보조 기억장치

- 물리적인 디스크가 연결되어 있는 기억장치
- 주 기억장치보다 느리지만 컴퓨터의 전원이 꺼져도 저장된 데이터가 보존됨
- HDD와 SSD로 나누어 짐

- 모든 프로그램은 컴퓨터에서 실행되기 위해 메모리의 일부를 차지한다



캐시

- 대부분의 프로그램은 높은 데이터 지역성을 갖고 있다



데이터 지역성

- 프로그램은 모든 코드나 데이터를 균등하게 Access 하지 않는다
- 대신 어느 한 순간에 특정 부분을 집중적으로 참조한다
- 시간적 지역성과 공간적 지역성으로 나뉜다

시간적 지역성

- 최근 참조된 주소의 내용은 곧 다음에 다시 참조된다

공간적 지역성

- 기억장치 내에 서로 인접하여 저장되어 있는 데이터들이 연속적으로 액세스 될 가능성이 높아진다

- 이를 이용해서 OS나 CPU는 자동으로 자주 쓰이는 데이터와 자주 쓰일 것 같은 데이터를 메모리에서 **캐시**로 읽어온다
- 따라서 캐시란, **자주 사용하는 데이터나 값을 미리 복사해 놓는 임시 장소**이다

캐시의 종류

L1 캐시

- 가장 고성능이자 고가인 작은 용량의 집적회로 사용

L2 캐시

- L1 캐시에서 데이터를 퍼가기 위한 캐시로 사용
- L1 캐시보다 용량은 크지만 조금더 느린 캐시

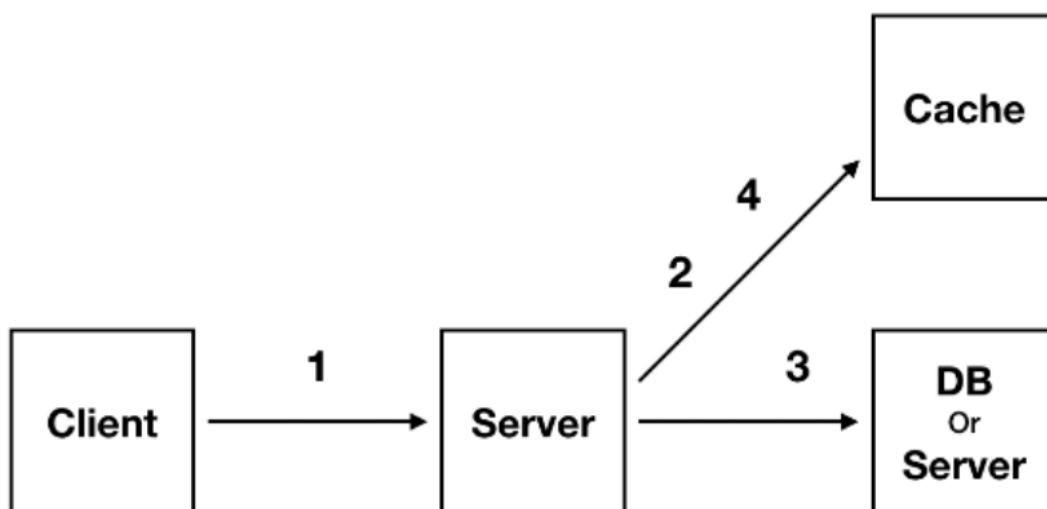
L3 캐시

- 가장 낮은 수준의 캐시
- 모든 코어에서 공유되는 캐시

사용 예시

- 데이터 접근 시간에 비해 원래의 데이터를 접근하는 시간이 짧게 걸리는 경우 (ex 서버의 균일한 API 데이터)
- 반복적으로 동일한 결과를 돌려주는 경우(ex 이미지, 썸네일, ...)

캐시 사용 과정

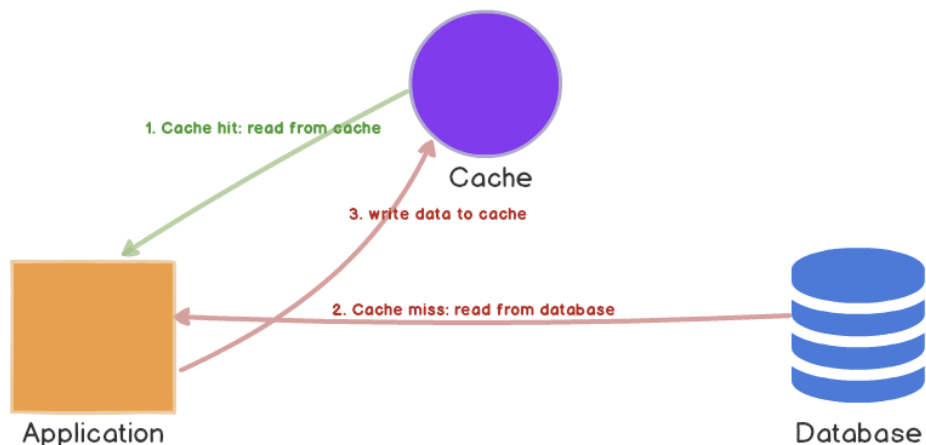


1. Client가 Server로 데이터를 요청한다
2. Server가 요청받은 데이터를 캐시메모리에 있는지 없는지 확인한다
 - a. 데이터가 있다면 캐시 안에 있는 것을 사용한다
 - b. 데이터가 없다면 DB를 확인한다 → DB에서 가져온 데이터를 캐시에 저장한다

🥊 캐싱 전략

- 웹 서비스 환경에서 시스템 성능 향상을 위해 가장 중요한 기술
- 디스크 기반 데이터베이스보다 훨씬 빠르게 데이터 반환 가능

1. Cache Aside



- 캐시를 옆에 두고 필요할 때만 데이터를 캐시에 로드하는 전략
- 처음 사용자가 요청했을 때 캐시엔 아무 데이터도 없는 상황
 1. Application은 먼저 캐시 저장소에 데이터가 있는지 조회
 2. Application은 DB에서 데이터를 조회하고 사용자에게 제공
 3. DB에서 가져왔던 데이터를 캐시 저장소에 저장
- 다음 사용자가 데이터를 요청했고, 캐시에 데이터가 있는 상황
 1. Application은 먼저 캐시 저장소에 데이터가 있는지 조회, 데이터가 존재하므로 사용자에게 데이터를 제공

장점

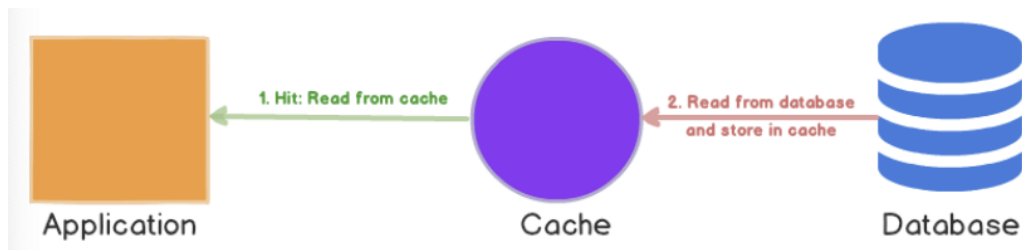
- 읽기가 많은 워크로드에 적합

- 캐시 오류에 대해 탄력적
- 캐시를 데이터베이스 모델과 분리 가능

단점

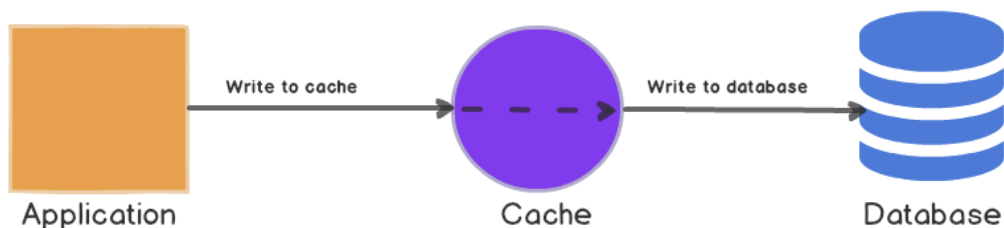
- 캐시에 없는 데이터인 경우 더 오랜 시간이 걸리게 됨
- 캐시가 최신 데이터를 갖고 있는가에 대한 문제 존재 (동기화 문제)

2. Read-Through



- 캐시와 데이터베이스를 일렬로 배치
- 캐시 미스가 발생하면 데이터베이스에서 누락된 데이터를 로드하고 캐시에 채움, 채운 데이터를 Application에 반환
- Application이 캐시를 채우는 역할을 하냐 마냐에 따라 Cache Aside 방식과 Read-Through 차이가 존재
 - Read-Through 방식은 Application이 캐시를 채우지 않는다

3. Write-Through



- Read-Through와 반대로 구성
- 데이터를 데이터베이스에 작성할 때마다 캐시에 데이터를 추가하거나 업데이트
- 캐시의 데이터를 항상 최신 상태로 유지할 수 있음

장점

- 항상 캐시가 동기화 되어있음

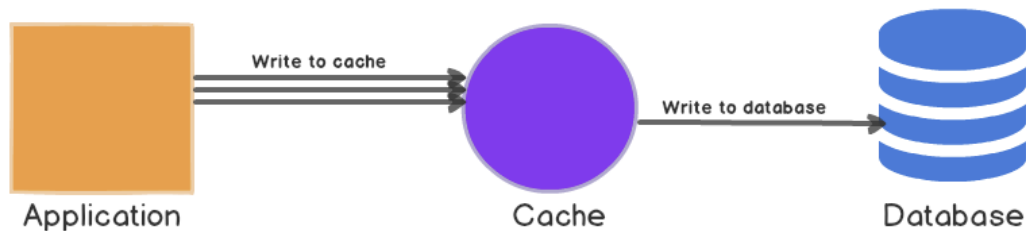
단점

- 쓰지 않는 데이터도 캐시에 저장되기 때문에 리소스 낭비
- 쓰기 지연 시간이 증가

4. Write-Around

- 데이터는 데이터베이스에 직접 기록되며 읽은 데이터만 캐시에 저장
- 실시간 로그 또는 채팅방 메시지와 같은 상황에서 적합

5. Write-Back



- Application은 즉시 확인하는 캐시에 먼저 데이터를 작성
- 약간의 지연 후 데이터를 다시 데이터베이스에 씀

장점

- 쓰기가 많은 워크로드에 적합
- 데이터베이스에 대한 전체 쓰기를 줄일 수 있다

단점

- 읽기가 많은 워크로드에는 적합하지 않음



하드디스크


- 보조기억 장치
- 사용자가 사용하고자 하는 데이터와 프로그램을 저장한다
- 전원을 끄더라도 저장된 데이터나 정보가 날아가지 않는 비휘발성 메모리이다

참고

▼ 링크


[컴퓨터의 메모리구조] 캐시,메모리,디스크의 차이 - Break Out of Your Comfort Zone

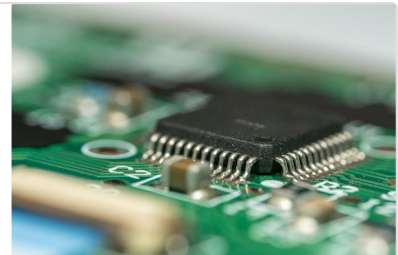
Cache, 하드디스크, 메인메모리의 차이Redis를 공부하려다가 캐시와 메모리, 하드디스크에 대해 헷갈려서 컴퓨터의 메모리 구조에 대해서 전반적으로 다시 정리해보았다.메모리는 '기억장치'라는 뜻을 가지고 있다. 일반적으로 메모리라고 하면 주로 '메인메모리' 즉, RAM을 의미한다...

 <https://sujinhope.github.io/2020/03/13/컴퓨터의-메모리구조-캐시,메모리,디스크의-차이.html>

SRAM 과 DRAM의 차이와 특징, 뜻 이해 (암기하기 쉬운 방법)

메모리를 공부하다보면 RAM이 크게 두 가지로 분류되는 걸 알 수 있다.
바로 SRAM과 DRAM 오늘...


 <https://m.blog.naver.com/ycpiglet/221984934010>



https://quasarzone.com/bbs/qn_hardware/views/818208

Caching 전략 소개 및 사용 예제

"캐싱 전략"은 최근 웹 서비스 환경에서 시스템 성능 향상을 위해 가장 중요한 기술입니다. 캐시는 메모리를 사용함으로 디스크 기반 데이터베이스 보다 훨씬 빠르게 데이터를 반환할 수 있고, 사용자에게 더 빠르게 서비스를 제공할 수 있습니다.

 <https://wnsgml972.github.io/database/2020/12/13/Caching/>