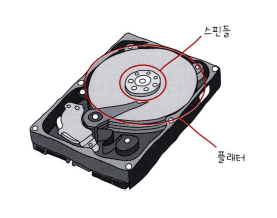
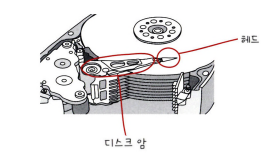
**Chapter 7 보조기억장치**

# ***7-1* 다양한 보조기억장치**

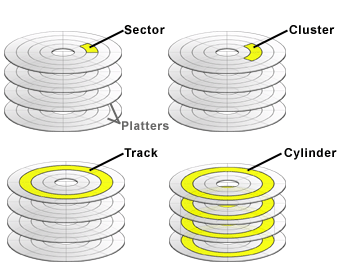
**<보조기억장치>**

* 주로 하드디스크나 플래시메모리 계열이며, USB, SD카드, SSD같은 저장장치를 일컫는다.
* 하드의 경우 자기적인 방식으로 데이터를 저장하는 물리 저장장치.
* 내부에는 원판인 플래터와 스핀들이 있다.
* **플래터**: 하드 내부에 있는 자성 물질로 덮여 있는 원판으로, N/S극 저장함.
* N/S극은 이진데이터로서 동작함.(N=0, S=1)



* **스핀들** : 해당 플래터를 회전시키는 부품
* **RPM** : Revolution Per Minute: RPM.
* **헤드** : 플래터를 대상으로 데이터를 읽고 쓰는 바늘같은 물품
* **디스크 암** : 해당 헤드를 원하는 위치로 이동시키는 부품.
* CD, LP의 경우 데이터가 단면으로 저장되나 하드는 더 많은 데이터를 저장해야하니 일반적으로 여려 겹의 플래터로 되어있고 플래터 양면을 사용.
* **플래터**에 데이터가 어떻게 저장되는가.

바로 플래터는 트랙과 섹터라는 단위로 데이터를 저장함.

결국 이게 하드가 물리장치고 자성을 띄워서 저장하다보니까 물리적으로 저장됨.

**트랙** : 한 플래터 위에서 여러 구간으로 나눈 동심원중 하나를 트랙.

**섹터** : 여러 조각으로 나뉘어진 트랙, 쪼개진 트랙이 바로 섹터.

1개 섹터는 일반적으로 512 바이트이나 이건 하드마다 케바케이고, 4096바이트에 이르는 경우도 있음.

**실린더** : 각 플래터에 위치한 모든 트랙들을 이야기 함.

여러 섹터들을 모아서 조각조각난 트랙을 만들면 그걸 또 **블록**이라고도 함.

연속된 정보는 실린더에 기록되는데, 각 정보가 플래터 위아래의 트랙 영역에 저장되는 방식임.

1번 플래터 위, 아래

2번 플래터 위 아래

3번 플래터 위 아래

이렇게 연속된 정보가 각각 저장되는 것.

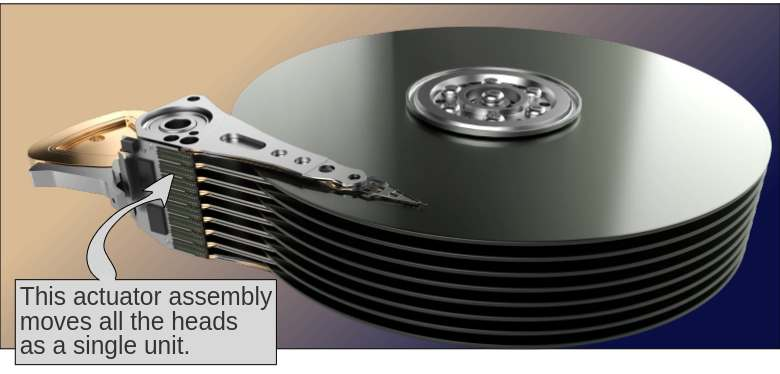
-> 이렇게 하는 이유는. 디스크 암을 움직이지 않고 연속된 정보를 읽어올 수 있기 때문.

* 하드의 **지연시간**

물리장치고, 플래터들에 데이터가 나뉘어져서 들어가있는 방식이라면 각 플래터를 물리 장치가 읽는 시간도 필요함.

위에서 말한 디스크암과 헤드가 각 플래터에 접근해서 읽어야하니까 그러는 것이고 지연시간이 생기게 됨.

**탐색시간** : 접근하려는 데이터가 저장된 트랙까지 헤드를 이동시키는 시간.

**회전지연** : 헤드가 있는 곳으로 플래터가 회전시키는 시간 ⇢ 데이터가 거기 있으니까

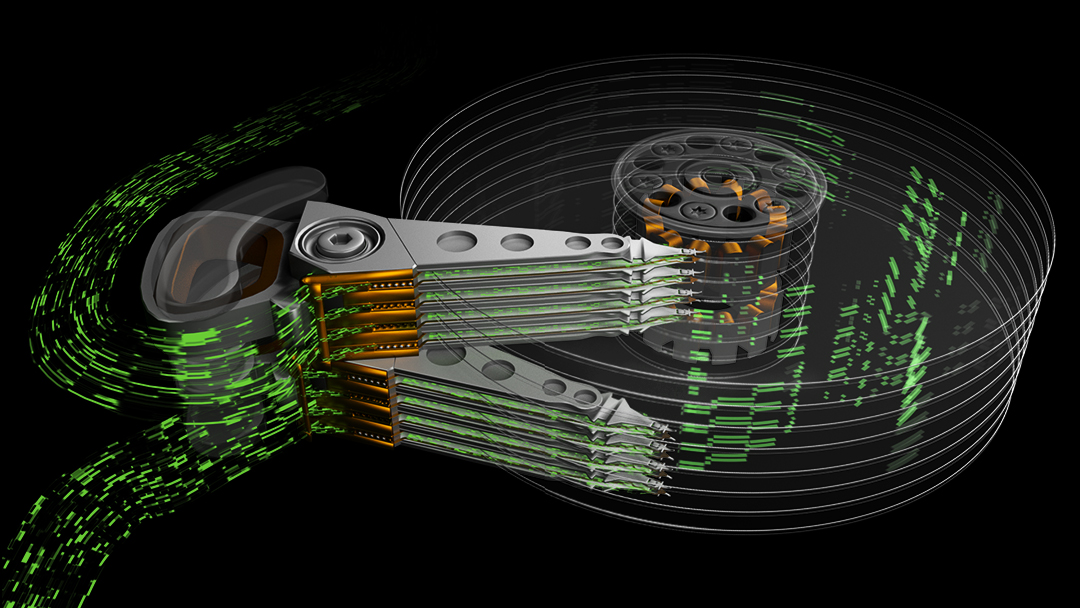
**전송시간** : 헤드가 그걸 읽어서 컴퓨터로 보내는 거.

* **다중 헤드 디스크**와 **고정 헤드 디스크**

플래터 한 면당 헤드가 하나 달려있으면 단일헤드디스크

여러개 달려있는건 다중헤드디스크라고 함.

헤드가 트랙별로 여러 개 달려 있는 하드디스크로, 각 트랙마다 헤드가 여러개 존재한다.

동작 : 각 트랙에는 별도의 헤드가 할당되어 있기 때문에 특정 트랙에 이동 없이 여러 트랙에 동시에 액세스할 수 있다.

여러개 있으면 트랙마다 헤드가 있으니 탐색시간이 전혀 필요 없고

하나만 있으면 매번 움직여야하니 탐색시간이 필연적으로 발생하게 됨.

< **플래시메모리>**

USB, SD, SSD 등의 경우가 모두 플래시메모리에 속함.

⑴ NAND 플래시 : NAND는 NAND 연산을 수행하는 회로 기반

⑵ NOR 플래시 : NOR 플래시는 NOR 연산을 수행하는 회로로 만들어진 게이트.

※ NAND 기준으로 언급.

* **Cell** : 플래시메모리에서 데이터를 저장하는 가장 작은 단위

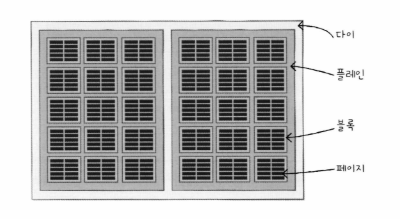
1셀에 1비트면 SLC(single), 2비트면 MLC(multi), 3비트면 TLC(triple)라고 함.

..SSD QLC(quad)... 5,6,7,8까지 나옴.

수명의 경우도 비트 수에 따라서 나뉨. 높은 수명과 성능은 SLC > MLC > TLC 순이고

용량은 큰데 비교적 낮은 수명과 가격을 원한다면 TLC > MLC > SLC 순서.

(방안에 한명이 드나들때 문지방이 빨리 닳는가 8명이 드나들때 문지방이 빨리 닳는가를 생각.)



➽ 하나하나가 셀이고 그다음 단위는 위와 같음.

플래시메모리에서 **읽기 쓰기**는 페이지 단위 / **삭제**는 하나 더 큰 블록 단위로 이루어짐.

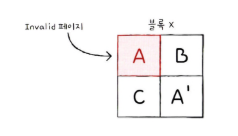
**읽기/쓰기와 삭제의 단위가 다르다**는게 플래시메모리의 가장 큰 특징.

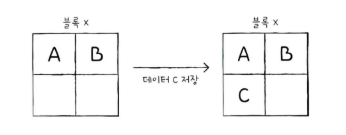
이때 페이지는 각각 다른 3개 상태를 가질 수 있는데

① Free : 어떠한 데이터도 없어서 새 데이터 저장 가능한 상태

② Valid : 유효한 데이터를 저장한 상태

③ Invalid : 쓰레기값이라 불리는 유효하지 않은 데이터 저장하는 상태…

플래시메모리는 덮어쓰기가 불가능해, Invalid인 경우는 새 데이터를 저장할 수 없다.



한 블록 x에 데이터를 저장하려고 할때 이미 AB가 있고, 여기에 C를 저장한다고 하면 충분히 가능.

➽ 빈 칸들은 모두 Free 상태

그러나, 이미 한 블록 안에 저장된 데이터인 A를 바꾸려고 하는 경우 불가능.

왜냐면 플래시메모리는 **덮어쓰기가 불가**.

따라서 기존 **A를 Invalid 처리**하고, A’가 새로 바뀐 데이터를 갖고있게 됨.

이렇게 용량 차지하고 있는 애가 바로 **가비지데이터**라고 하며

이런 애들은 낭비가 심하니, 주기적으로 SSD는 가비지컬렉션을 진행함.

# **< 07-2 RAID의 정의와 종류>**

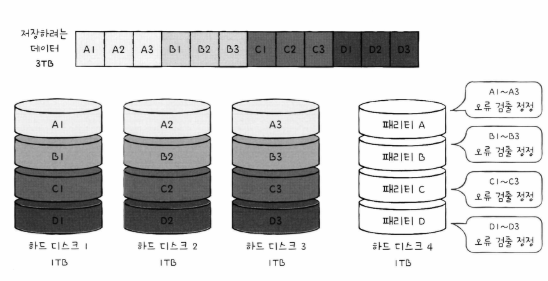
# **RAID :** 여러개의 물리적 보조기억장치를 하나의 **논리적 보조기억장치처럼** 사용하는 기술.

# 어떻게 묶느냐에 따라 RAID 레벨이 나뉨.

* **스트라이핑** : 각 기기에 데이터를 나눠서 저장하는 것.
* **스트라입** : 분산하여 저장된 데이터.
* **레이드0** : 그냥 용량대로 다 묶어버려서 여벌복제를 하지 않음. (500GB + 1000GB + 2000GB 하면 3500GB로 만들어서 쓰는 것…)

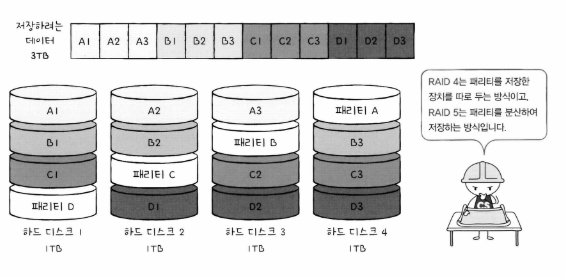
-> 이론상 물리 장치 개수 만큼 속도가 늘어남. 다만 디스크 하나라도 뻗으면 문제가 크게 생김.

* **레이드1**: 복사본을 만드는 방식이고, 거울같은 완전한 복제본을 만드는 작업이라 미러링이라고 불리기도 함. 속도는 비교적 느릴 수 있음. 용량도 용량대로 다 늘어나는 것이 아님. (비용도 증가)
* **레이드4** : 레이드1처럼 복사본을 만드는 대신 오류 검출하는 **패리티 비트를 두는 구성**방식임. 그런데 패리티비트를 두는 것은 특정 장치를 특정해서 패리티비트를 몰아주는 방식.



오류 검출만 가능한게 패리티비트이나, 레이드에서는 오류 복구도 됨.

* **레이드5**: 4번의 단점은 바로 패리티 저장 디스크에도 데이터를 쓰다보니, 패리티 저장하는 장치에 병목이 생긴다는 점임.따라서, 한 장치에 몰아서 하는게 아니라 각 장치마다 **패리티비트를 분산**해서 저장함.



* **레이드6** : 레이드5 + 안정성 증대를 한 구조임. 장치에 패리티를 나눠서 하기는 하는데, **두 개의 패리티**를 두는 방식임.

