CSE 206 - 파일 처리론
(File Processing)

2 장. 파일 저장 장치

Content

- 저장 장치의 계층
- Hard Disk (HDD) – 구조와 특성 요소
- Hard Disk – Data 읽기 쓰기
- Structured Data를 File로 HDD에 저장할 때 고려해야 할 요소들

2.1. 저장 장치의 계층

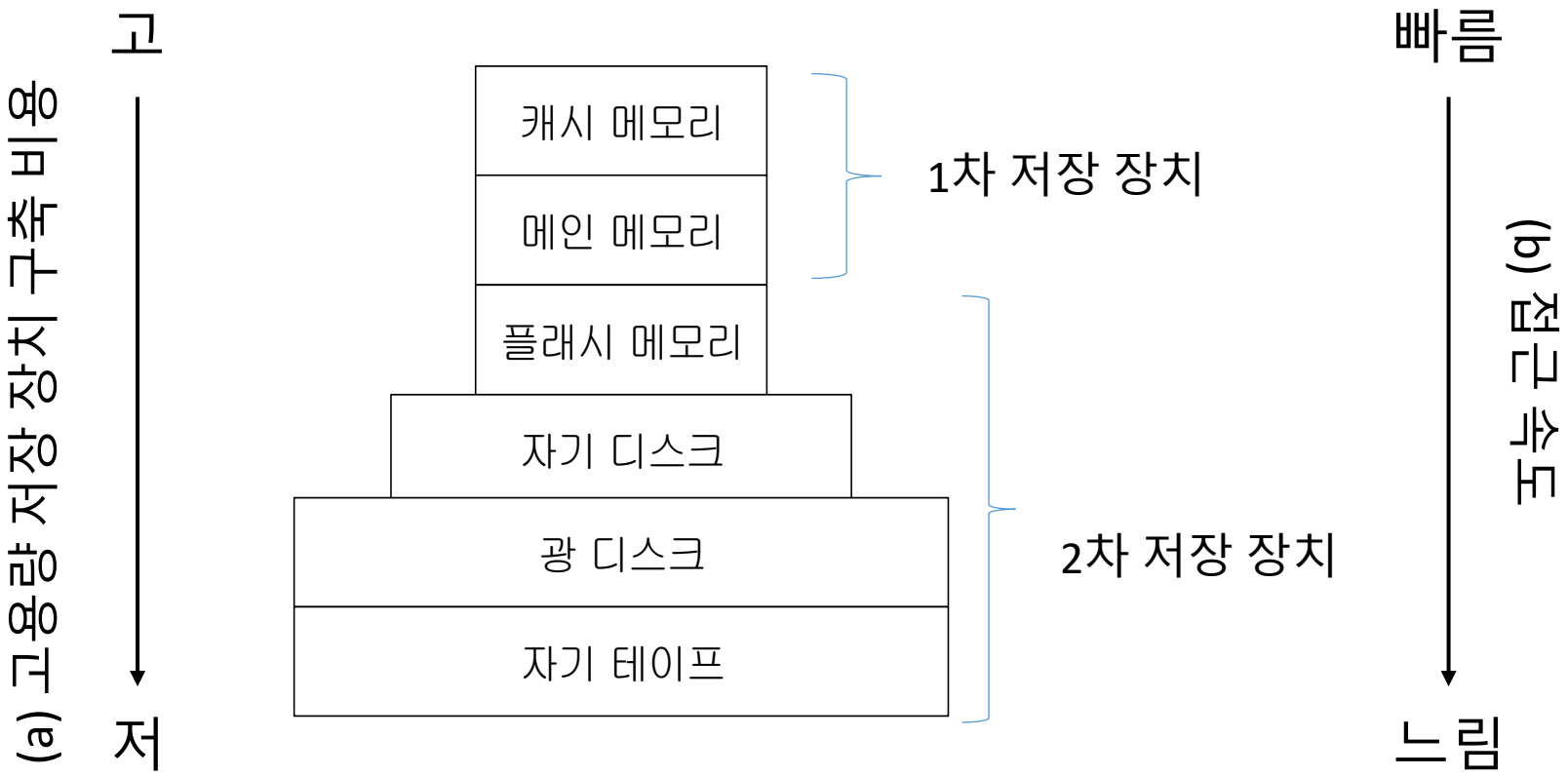
Storage Device (저장 장치)

- **Storage device (저장 장치)**
  - **데이터를 저장하고 검색하기 위한 장치**
  - 저장 매체와 접근 장치로 구성

- **Storage medium (저장 매체)**
  - 데이터를 저장해 두기 위한 매체
  - 소멸성 (volatile) vs. 비소멸성 (nonvolatile)
    - 소멸성 : 전원을 끄면 저장되어 있던 데이터가 날아간다.
    - 비소멸성 : 전원을 꺼도 저장 되어 있는 데이터가 유지된다.

- **Access mechanism (접근 장치)**
  - 저장 매체에서 데이터를 읽거나 쓰는 장치

저장 장치의 계층


고
(a) 구입한 저장 장치 구축 비용
저
캐시 메모리
메인 메모리
플래시 메모리
자기 디스크
광 디스크
자기 테이프
1차 저장 장치
2차 저장 장치
빠름
(b) 접근 속도
느림

Main memory

- **메인 메모리(main memory)**
  - 프로그램 실행과 이에 필요한 데이터 저장 공간.
  - DRAM (Dynamic random access memory)
    - 어느 데이터를 읽고 쓰는지에 관계없이 데이터 I/O에 걸리는 시간이 일정
    - 저장 용량 대비 비용이 큼
      - 동일 저장용량을 만드는데 Hard Disk에 비해 매우 비쌈.
      - Hard Disk 1 TB : 45,000원 (2021.9 기준)
      - Memory 16 GB : 86,000 원 (2021.9 기준)
    - 소멸성 (오랜 기간 데이터를 저장하기에는 부적합) 장치

Cache memory

- **캐시 메모리(cache memory)**
  - 가장 빠르고 가장 비싼 저장장치
  - CPU와 가장 가까이서 동작하는 저장 장치
    - L1, L2 Cache
  - SRAM (static random access memory)
    - 어느 데이터를 읽고 쓰는지에 관계없이 데이터 I/O에 걸리는 시간이 일정
    - 속도가 매우 빠름
      - CPU > Cache > Main memory
    - 비용대비 용량이 매우 적음
      - 동일 저장 용량을 만들기 위해 드는 비용이 Main memory보다 비쌈.
    - 소멸성 (오랜 기간 데이터를 저장하기에는 부적합)

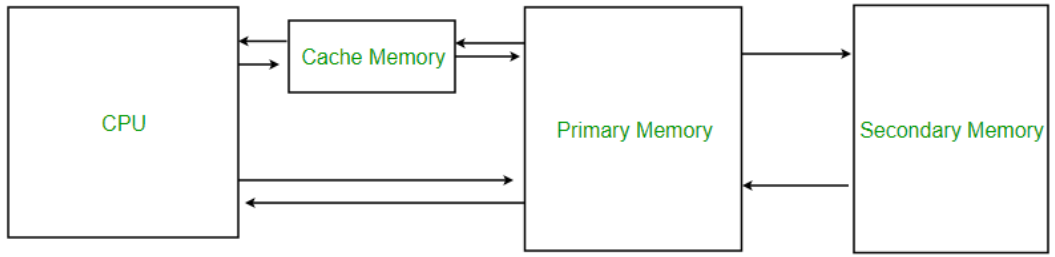


Image source : https://www.geeksforgeeks.org/cache-memory-in-computer-organization/

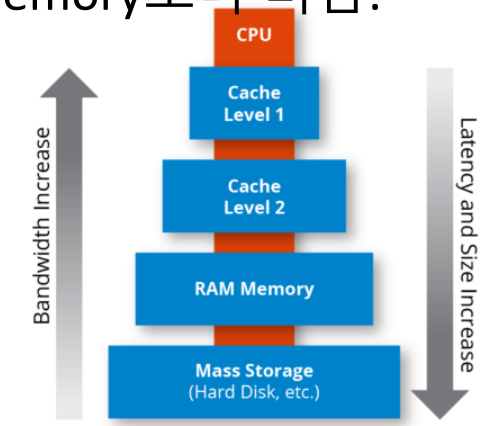


Image source: https://hazelcast.com/glossary/memory-caching/

Flash Memory, Magnetic Disk

- **플래시 메모리(flash memory)**
  - 고밀도, 고성능 메모리로서 비소멸성
  - 메인 메모리와 비슷한 접근 속도

- **자기 디스크(magnetic disk)**
  - 가장 많이 쓰이는 Hard Disk
  - 데이터 저장 장치의 주 매체
  - 대용량이고 비소멸성

Figure : https://www.datanumen.com/blogs/3-main-reasons-solutions-when-hard-drive-fails-to-spin-up/

저장 장치의 유형

- **광 디스크(optical disk)**
  - 광학적으로 저장, 레이저로 메체에서 데이터 읽음
  - 용량이 크고 보존 기간이 길다
  - DVD(digital video disk): 4.5GB - 15GB, Blu-ray : 25GB - 50GB

- **자기 테이프(magnetic tape)**
  - 데이터의 백업과 보존을 위한 저장매체
  - 순차 접근 저장 장치

Summary - **저장장치** (memory, storage)

- **1차 저장 장치(primary storage)**
  - **CPU가 프로그램 데이터를 처리하기 위한 작업 공간으로 사용**
  - 데이터 접근 시간이 일정하고 빠름

  - **1. 메인 메모리(main memory)**
    - 프로그램/데이터를 처리하기 위한 작업 공간

  - **2. 캐시 메모리(cache memory)**
    - 메인 메모리보다 데이터 접근 시간이 빠름
    - 메인 메모리 성능 향상 목적

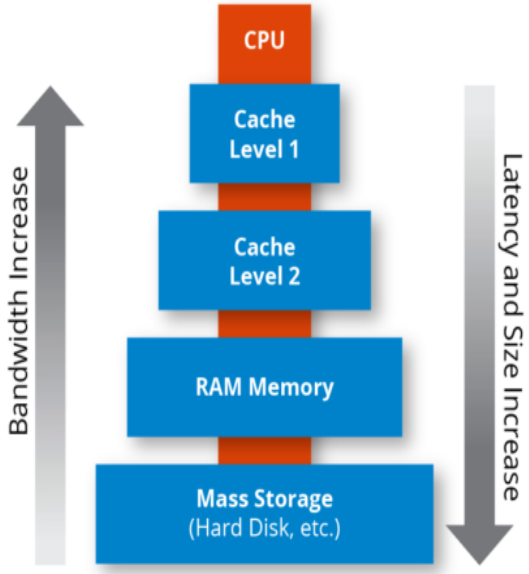


Image source: https://hazelcast.com/glossary/memory-caching/

Summary - **저장장치** (memory, storage)

- **2차 저장장치(secondary storage)**
  - 파일 혹은 대용량 데이터의 항구적인 저장에 주로 사용.
  - 1차 저장장치에 비해 내용 접근 시간이 느림 (10만 배 이상)
  - **1. 자기 디스크(magnetic disk)**
    - 용량이 크고 싸며 임의 접근이 가능하여 주로 파일 저장에 쓰임
    - 저장된 데이터는 메인 메모리를 거쳐 CPU에 의해 처리
  - **2.** 플래시 메모리(**Flash Memory)**
    - USB memory, Solid State Drive로 읽기 속도가 빠른 파일 저장 장치로 쓰임.
  - **3. 광 디스크(optical disk)**
  - **4. 테이프 (Tape)**

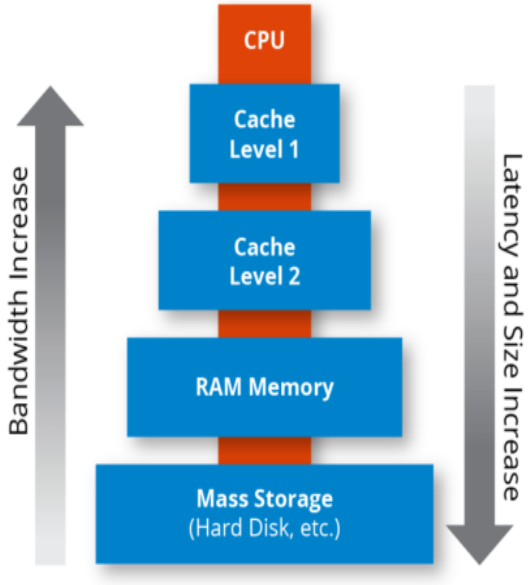


Image source: https://hazelcast.com/glossary/memory-caching/

2.2. Hard Disk – 구조와 특성 요소

Disk 저장 장치

- 직접 접근 저장 장치 (DASD; direct access storage device) 중 가장 많이 쓰이는 장치

- 종류
  - 하드 디스크(hard disk) : 1955년 IBM 개발-초기 5MB
  - 유연한 디스크(flexible disk) : floppy disk, diskette

Hard Disk

Image Source : https://www.datanumen.com/blogs/3-main-reasons-solutions-when-hard-drive-fails-to-spin-up/

Hard Disk 동작 모습

- https://youtu.be/9eMWG3fwiEU

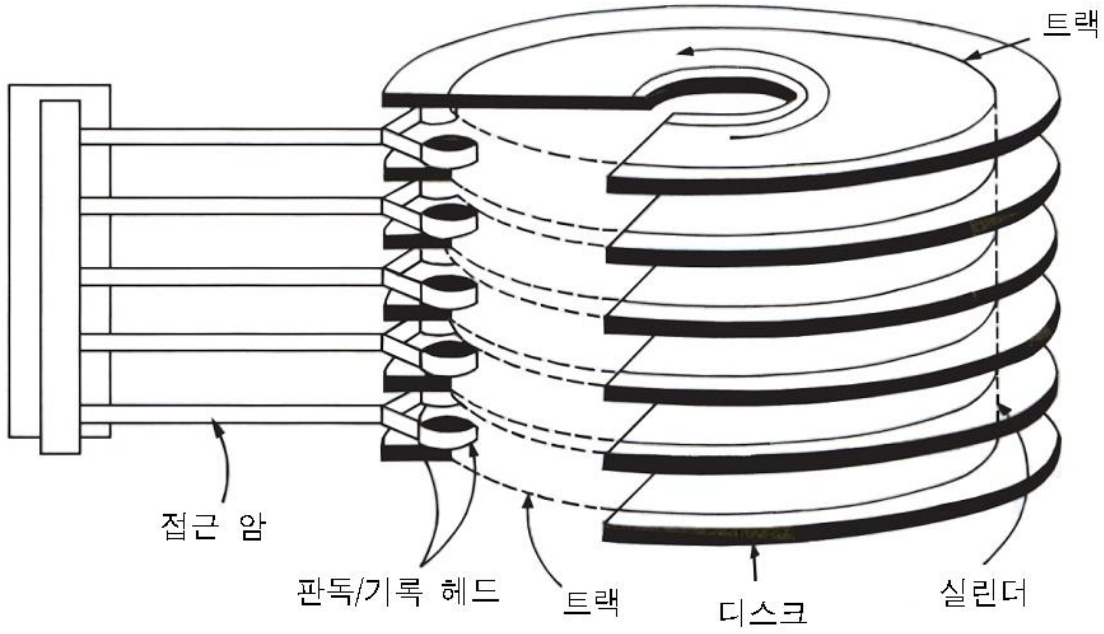


Figure : https://www.datanumen.com/blogs/3-main-reasons-solutions-when-hard-drive-fails-to-spin-up/

Hard Disk 구조 (Cont'd)

- **디스크 팩(disk pack)**
  - 디스크 원반(platter)의 모음 (보통 6~20개의 원반)
  - 원반 직경 : 10.5인치, 14인치 (소형의 경우 1~3.5인치)
  - 면당 : 수천~ 14,000개 이상의 트랙
  - 기록 표면(surface) : 양면 사용(맨 위/아래 면 제외)
    - 11개의 디스크는 20면
  - 회전 속도 : 3,600 ~ 7,200 rpm (revolutions per minute)

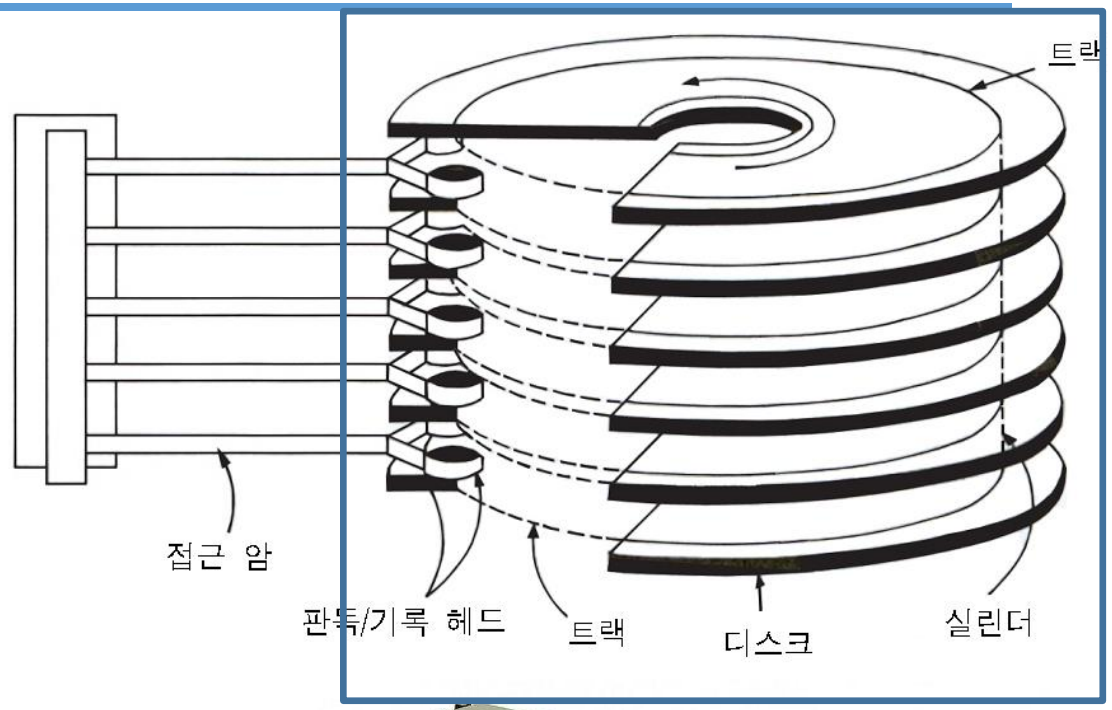
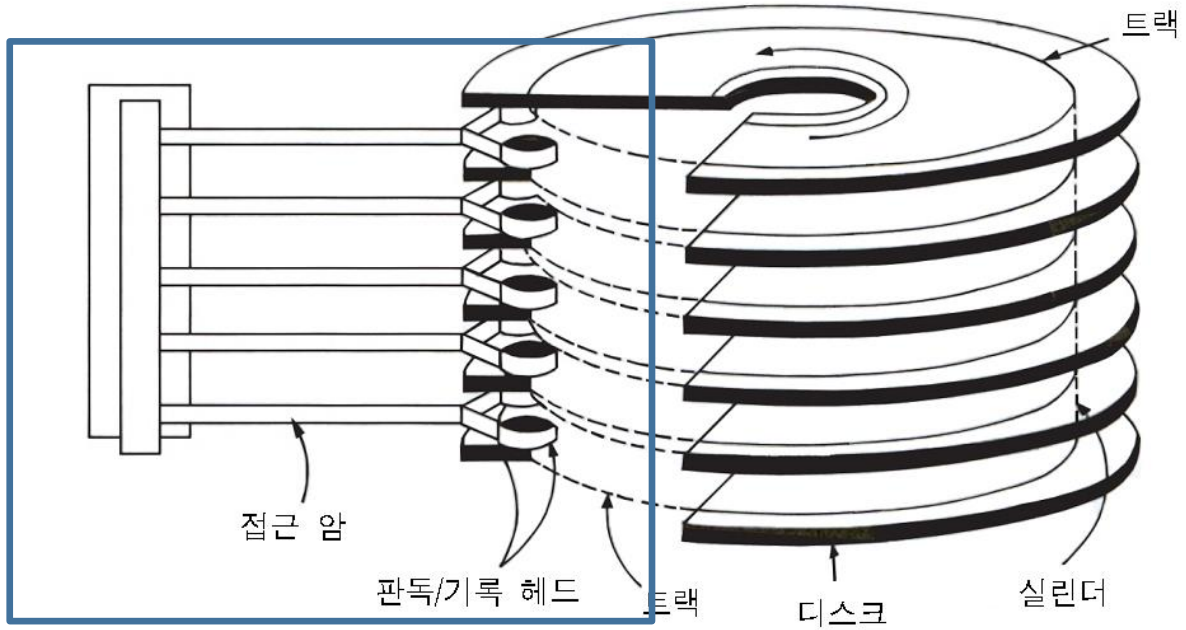


Figure : https://www.datanumen.com/blogs/3-main-reasons-solutions-when-hard-drive-fails-to-spin-up/

Hard Disk 구조 (Cont'd)

- **디스크 구동기(disk driver)**
  - 제어기, 접근 암, 판독/기록 헤드, 팩 회전 장치

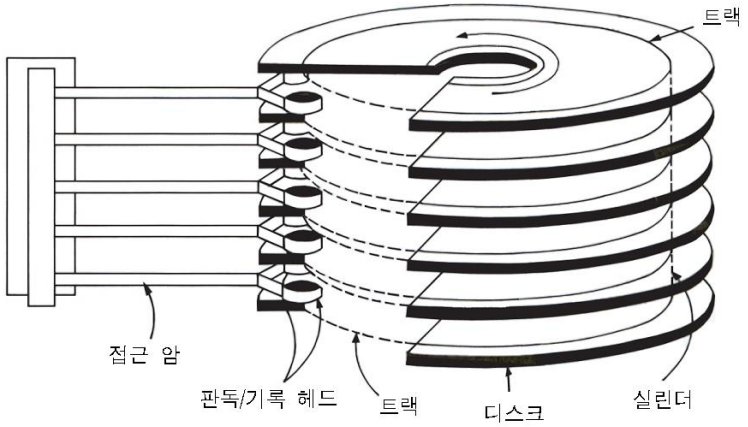
- **디스크 제어기(controller)**
  - 원하는 데이터가 어느 원반, 어느 면, 어느 주소에 있는지 판독
  - 버퍼 관리
  - 오류 발견/수정
  - 판독/기록 관리

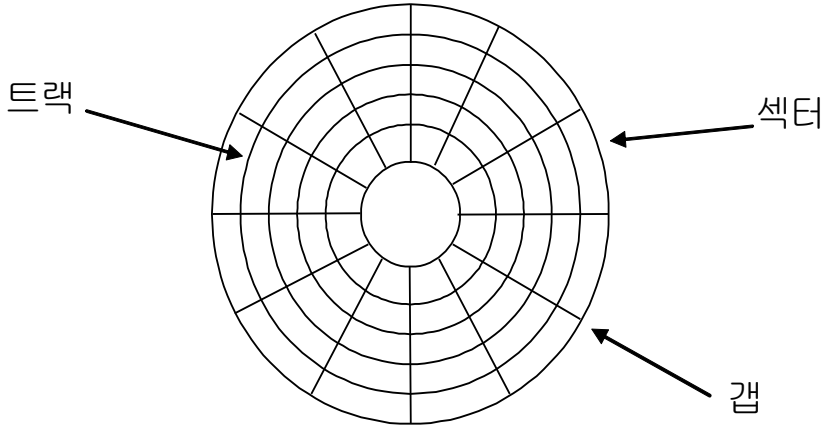


Hard Disk - 데이터 저장

- **디스크의 구성**
  - 트랙(track): 갭(gap)으로 분리된 섹터(sector)들로 구성
  - 섹터(sector): 읽기와 쓰기 (기록과 판독) 작업의 최소 단위
  - 실린더(cylinder) : 지름이 같은 모든 트랙

- **블록(block)**
  - 디스크와 메인 메모리 사이에 전송되는 데이터의 논리적 단위
  - 블록은 하나 이상의 섹터로 구성

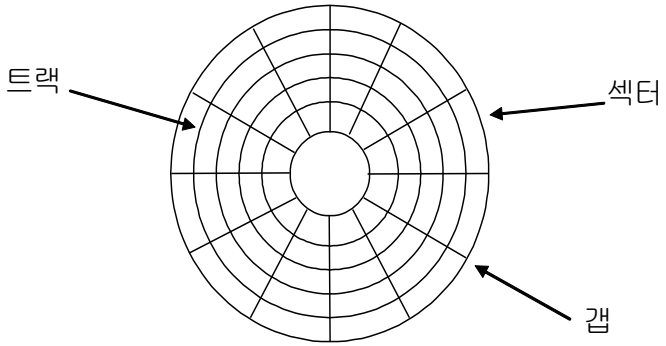


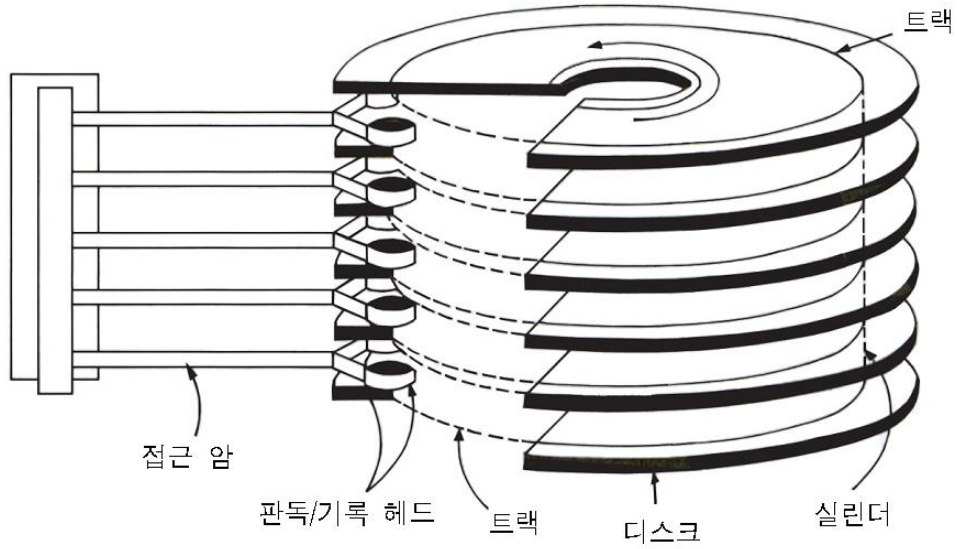


유동 헤드 디스크와 고정 헤드 디스크

- **유동 헤드 디스크(movable-head disk)**
  - 각 기록 면마다 판독/기록 헤드가 있음
  - 데이터 읽기/쓰기 시 헤드가 원하는 트랙에 위치하도록 접근 암을 이동
  - 전송시간 : 데이터를 동일 면 보다는 동일 실린더에 저장하는 것이 더 효율적

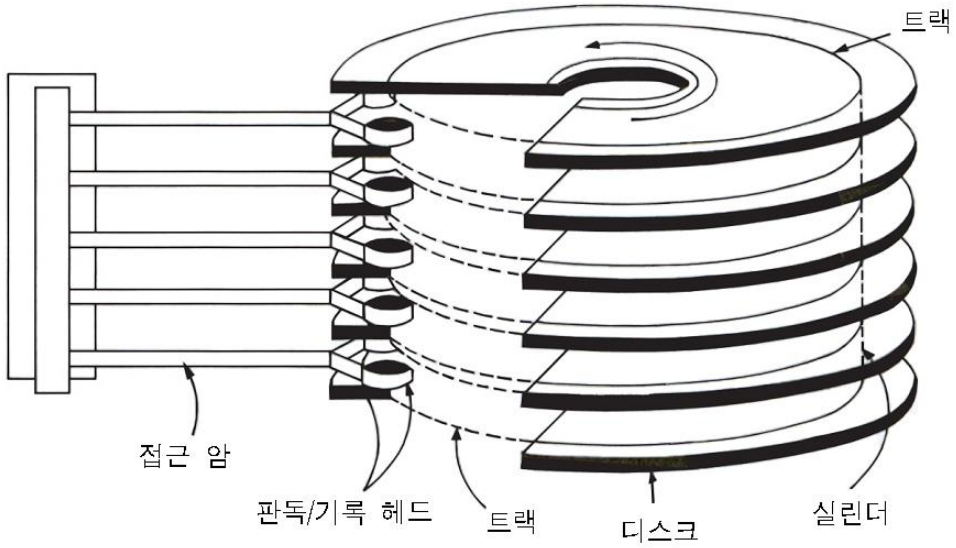
- **고정 헤드 디스크(fixed-head disk)**
  - 기록면의 각 트랙마다 하나의 판독/기록 헤드가 있음
  - 판독/기록을 위해 헤드를 이동시킬 필요가 없음



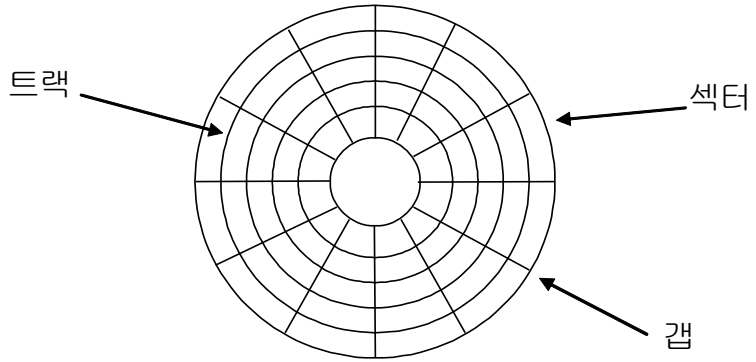


Hard Disk의 특성 요소 (성능에 영향)

- **Hard Disk의 특성 요소**
  - 회전 속도 (읽기/쓰기 속도)
  - 디스크 드라이브의 원반 수 (읽기/쓰기, 용량)
  - 기록 면당 트랙 수 (용량)
  - 트랙당 섹터 수 (읽기/쓰기, 용량)
  - 섹터(트랙)당 바이트 수 (읽기 쓰기, 용량)



▶ HDD 디스크 (예)
  - 회전 속도 : 7200 RPM
  - 디스크 드라이브 - 원반 수: 9개 (기록 면: 16개)
  - 16,384개의 트랙/기록면
  - 128섹터/트랙
  - 4096바이트/섹터
  - 용량: 16면 x 16384트랙 x 128섹터 x 4096바이트 = 128GB



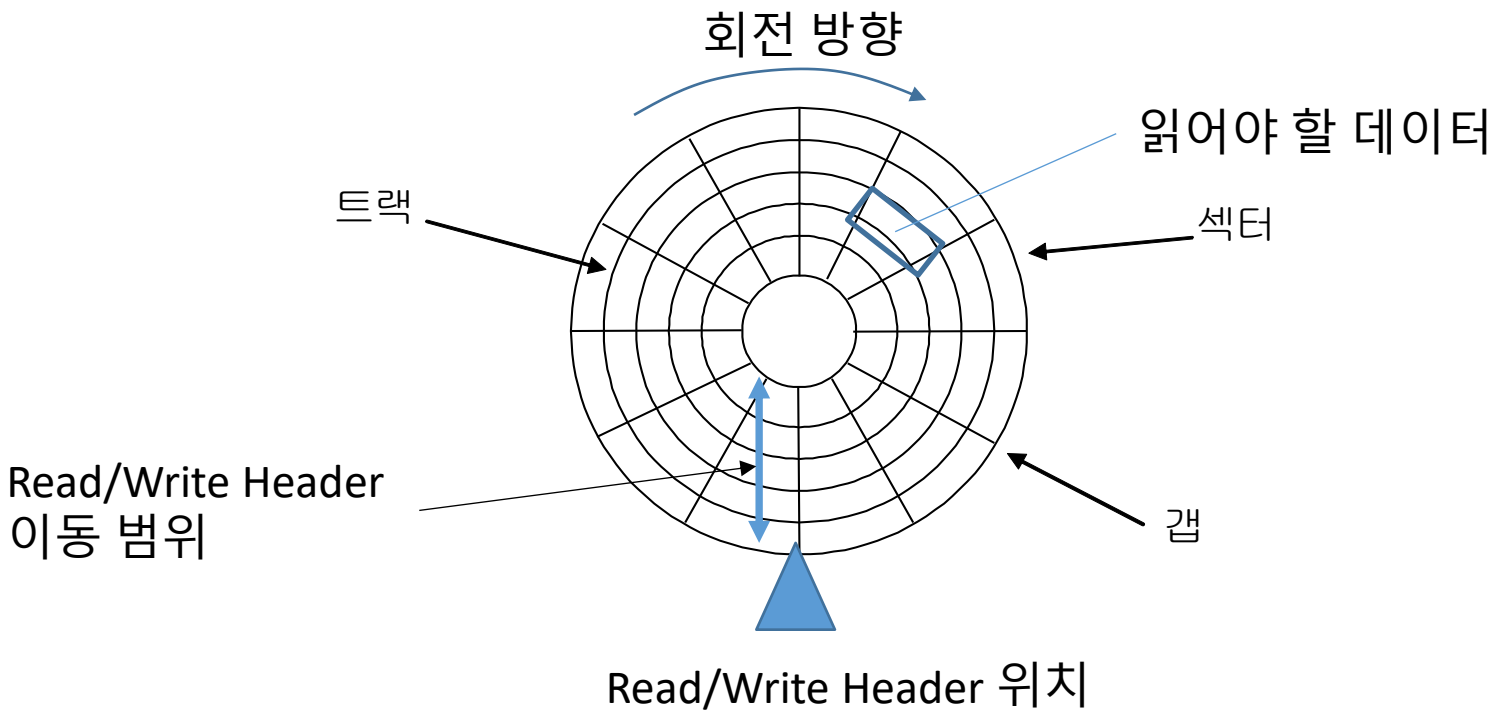
Hard Disk - **데이터 접근**

- **디스크에 저장된 데이터 읽기/쓰기**
  - 메인 메모리에서만 연산 작업이 가능
  - 디스크와 메인 메모리 사이에 **데이터 전송 (Block 전송)**이 필요
  - **Block은 하나이상의 Sector**로 구성

- **디스크에 데이터를 읽거나 쓰려면**
  a) 접근해야 할 Sector (Target sector)가 위치하는 Track위로 Read/Write Header를 이동
  b) 디스크가 회전하면서 Target sector가 Read/Write Header 아래를 통과할 때 데이터 읽기/쓰기

Hard Disk - 데이터 접근

- **데이터 접근 시간(access time)**
  - 탐색 시간(seek time)
  - 회전 지연 시간(rotational latency)
  - 전송 시간(transfer time)



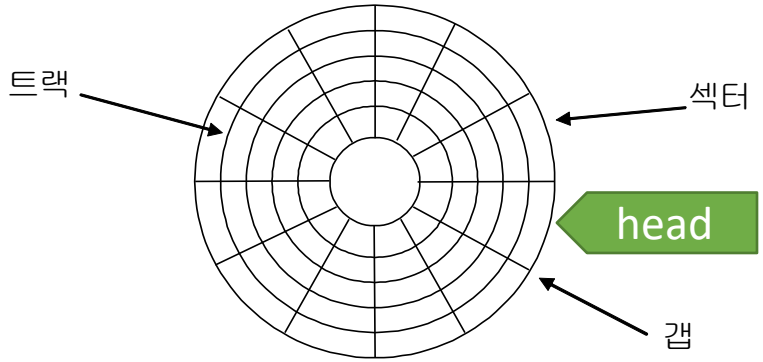
데이터 접근 시간 – 탐색 시간 (seek time) : s

- 원하는 데이터가 있는 실린더(또는 트랙)에 R/W 헤드를 위치시키는데 걸리는 시간

$$s = c + \delta \cdot i$$

c : 접근장치가 처음 가동하는데 걸리는 시간
δ : 단위 거리 이동 시간 (한 트랙 이동에 걸리는 시간)
i : 이동해야 할 트랙 수

- 속도 평가로 평균 탐색 시간(average seek time : s)을 사용
- 평균 탐색 시간은 i 를 “Disk track 수”/2 로 계산.



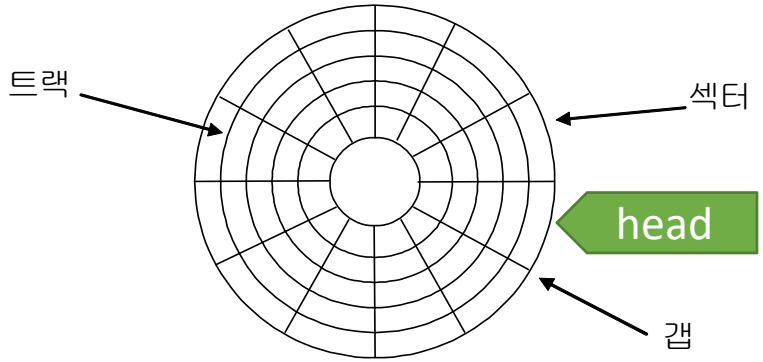
데이터 접근 시간 – 회전 지연 시간 (rotational latency) : r

- 탐색 완료에서 데이터 전송 시작까지의 지연
- 목표 트랙에서 판독/기록 헤드 밑에 블록의 첫 번째 섹터가 위치할 때까지의 시간
- 분당 디스크 회전 수: rpm (revolutions per minute)
- 디스크 회전속도(ms): rpm/(60x1000)

$$r = \frac{1}{2} \cdot (\text{1회전 시간})$$

1회전 소요 시간(ms) : $2r = \frac{60 \times 1000}{rpm}$

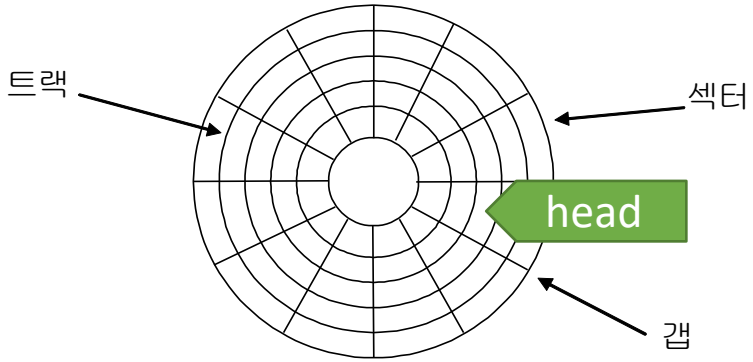
3,600 rpm : 8.33 ms
5,400 rpm : 5.56 ms



데이터 접근 시간 – 전송 시간 (transfer time)

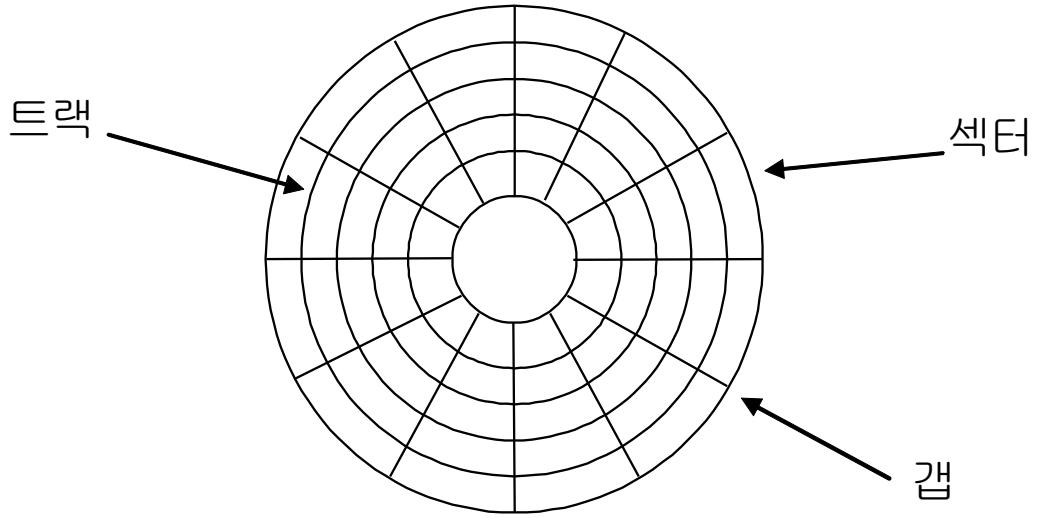
- **전송 시간(transfer time; t)**
  - 블록의 섹터들과 이들 사이의 갭들이 헤드 밑을 회전하며 통과하는데 걸리는 시간
  - 하나의 트랙에 저장되는 데이터 크기가250,000 B이고 1회전 소요시간이 10ms일 때 25MB/sec 읽기 가능
    - 10 ms : 250,000 B = 1000 ms : 25,000,000 B
    - 16384B 크기의 Sector 전송 시간: 2/3ms

- **전송률 (transfer rate)**
  - 초당 데이터가 전송되는 속도(MBps)
  - 위 예에서 전송률은 25MBps



연습 : 데이터 전송 시간

- HDD로부터 16,384B 크기(4 개의 연속된 Sector)의 데이터 전송
  - 다음 과 같은 디스크의 특성 요소가 주어 졌을 때 16384 B의 데이터 전송 시간은?
    - 1 Sector에 4096B 저장가능
    - 분당 회전 수 7200rpm (8.33ms/rotation)
    - 하나의 트랙에 180 개 Sector가 존재
    - 위와 같을 경우 전송 시간은 얼마?



연습 : 블록 읽기

- Sector의 읽기 시간
  - 평균 Sector 읽기 시간

    (a) Sector 전송 시간 : 0.25ms, 회전 속도 7200 rpm.

    (b) 평균 Rotational Delay : ??

    (c) 평균 Seek time : ??

    I = 전체 트랙 수(16384)의 1/2을 이동하는 시간이라 정의
    - 출발과 정지를 위해 1ms
    - 1000실린더 이동에 1ms 걸림

  평균 Sector 읽기 시간 = 평균 탐색 시간 + 평균 회전 지연 시간 + Sector 전송 시간 (0.25) = ?? ms

2.3. Hard Disk – Data 읽기 쓰기

Hard Disk에서의 Disk I/O

- Hard Disk를 포함한 2차 저장장치는 Data를 기록만 한다.
- 데이터를 사용하거나 변경하는 작업은 Main memory에서 실행된다.
  - Hard Disk에 기록된 Data를 사용하려면 Disk의 Data를 Main memory로 전송해야 한다.
  - Hard Disk에 Data를 기록하려면 Main memory의 Data를 Hard Disk로 전송해야 한다.

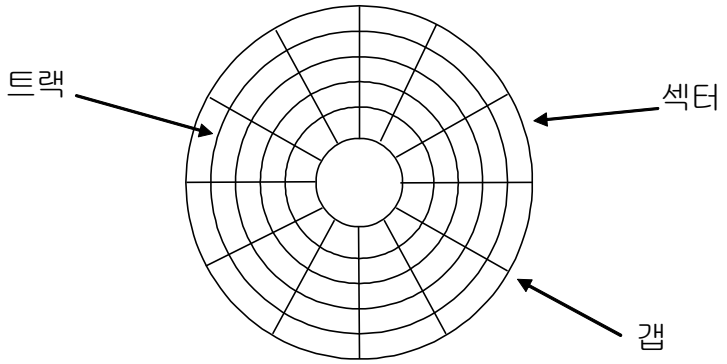
CPU (Central Processing Unit):
명령에 따라 정보를 처리하여 결과를 생성한다. 정보 처리를 위해 메모리에 저장된 정보를 읽고 생성된 결과를 메모리에 저장한다.

1차 저장장치(Main Memory)
: 계산에 필요한 정보를 저장해 두는 장치.
빠르게 접근할 수 있으나 용량이
상대적으로 작다. 전원이 꺼지면 정보는
모두 사라진다.
(책상위의 메모지, 연습장)

2차 저장장치(HDD)
:계산에 필요한 정보를 저장해 두는 장치.
용량이 상대적으로 크나 접근 속도가 느리다.
전원이 꺼져도 정보는 계속 저장된다.
(책장에 꽂아 둔, 책 혹은 잘 정리된 노트)

Hard Disk I/O : Block I/O

- Sector가 Disk I/O의 최소 단위이다.

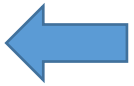


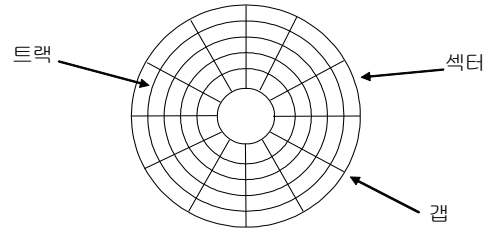
예 : 연속된 3 섹터를 1 block라 하면 최소 전송 데이터량은?

- Block I/O :
- I/O 속도 향상을 위해 항상 “연속된 n개의 sector” 단위를 최소 I/O 단위로 한번에 읽고 쓴다.
  - N은 OS에 따라 다르나, OS 운용 도중에 변하는 경우는 거의 없다.
    - 예) Data를 읽을 때는 n =3, 쓸 때는 n = 5 와 같이 사용하지 않고, 읽을 때 n = 3이라면 쓸 때도 n = 5이다.
- 이 “연속된 n개의 sector” 를 1 Block이라 한다.
- 즉, Block I/O에서는 I/O의 최소 단위가 1개의 Block이다.

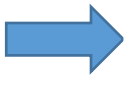
Block I/O

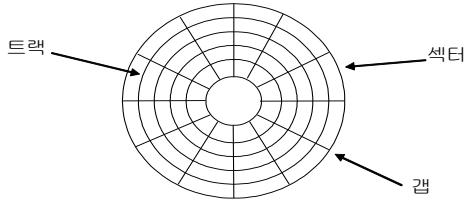
- **Read**
  - Disk Block에서 Main memory로 data 전송.





- **Write**
  - Main memory의 데이터 Block (최소 크기 Block size)가 Disk Block에 저장.

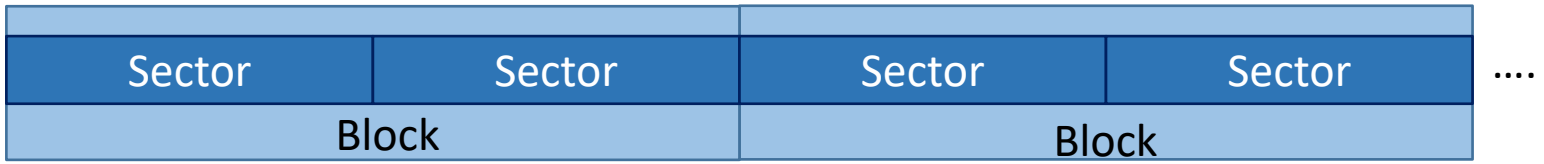




- **Update**
  - Disk Block에서 Main memory로 data 전송 후, Memory에서 데이터 변경, 그리고 Memory에 저장된 Block을 Disk block에 저장.
  - Data update는 Disk에서 일어나지 않고 Main memory에서 일어난다.

Block

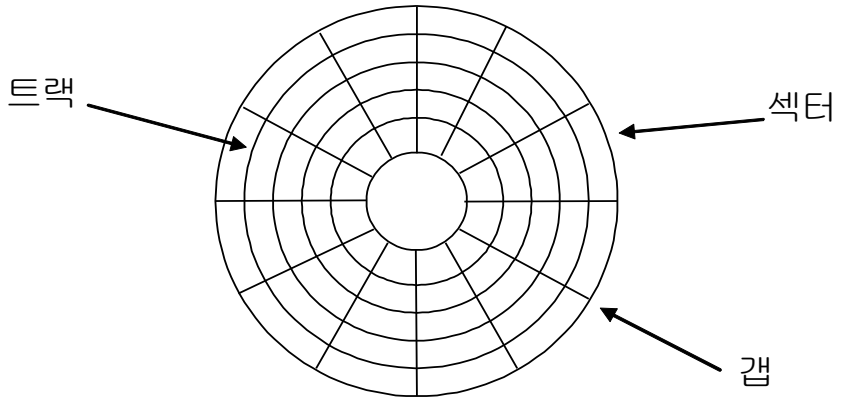
- **블록(block)**
  - 디스크와 메인 메모리 사이의 데이터 전송의 단위
    - I/O 속도 향상을 위해 항상 "연속된 n개의 sector" 단위를 최소 I/O 단위로 한번에 읽고 쓴다.
    - N은 OS에 따라 다르나, OS 운용 도중에 변하는 경우는 거의 없다.
      - 예) Data를 읽을 때는 n =3, 쓸 때는 n = 5 와 같이 읽기/쓰기 다른 값을 사용하지 않고,
      - 읽을 때 n = 3이라면 쓸 때도 n = 3이다.
    - I/O의 최소 단위가 1개의 Block이다.



Sector : 디스크 읽기 쓰기의 최소단위 **Block :** 디스크 <-> 메인 메모리 사이의 데이터 전송의 최소 단위

Block size

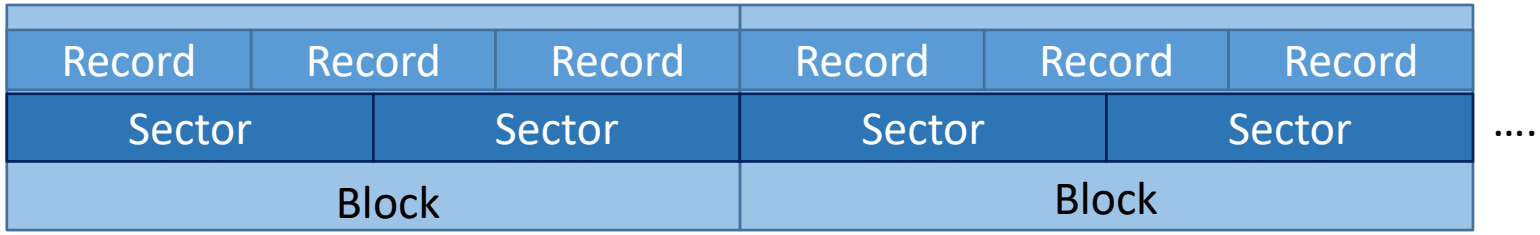
- **블록의 크기**
  - 보통 512byte, 1KB, 4KB
  - 너무 커도 안되고 작아도 안된다.
    - 너무 크면, 불필요한 데이터 전송을 야기하고 과다한 버퍼 할당으로 메모리의 효율성 저하
    - 너무 작으면, 데이터 전송 횟수가 많아진다



Structured data의 Record와 Block의 관계

상품번호	이름	가격 (원)	판매한 개수	재고 수
과일_1	사과	3,000	100	200
과일_2	바나나	1,000	150	150
과일_3	딸기	5,000	80	120
과일_4	감	2,500	90	110

- 파일에 structured data가 저장됨.
- 일반적으로, File에 저장되는 record 하나하나가 순차적으로 Disk sector에 저장됨.
- Block이 연속된 sector로 구성되어 있으므로,
- **Block은 연속된 record로 구성되어 있다**고도 볼 수 있음.



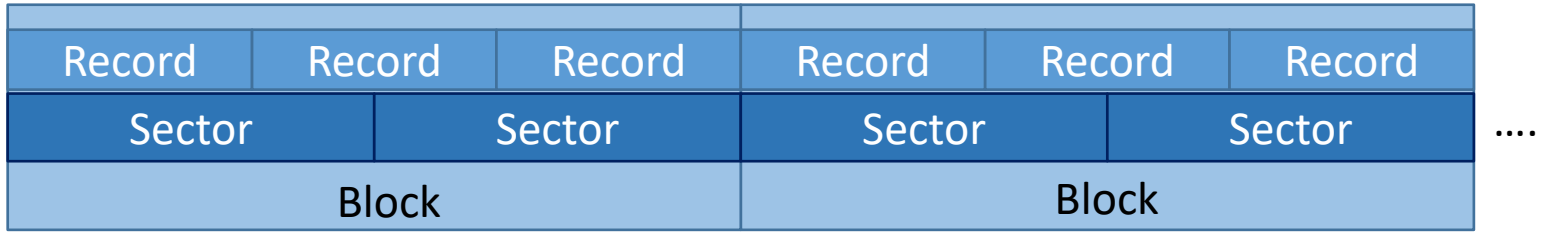
Blocking Factor : *Bf*

Blocking

- Main memory와 I/O 효율을 위해 여러 개의 record를 하나의 Block에 저장시키는 것
- Record는 사용자가 데이터를 저장하기 위한 논리적 단위
- Blocking Factor (*Bf*): 한 Block에 온전하게 들어갈 수 있는 Record 최대 수
  - 블록 하나가 16 Byte고 Record 하나가 5 바이트면 Blocking factor는?

$$Bf = \lfloor B/R \rfloor$$

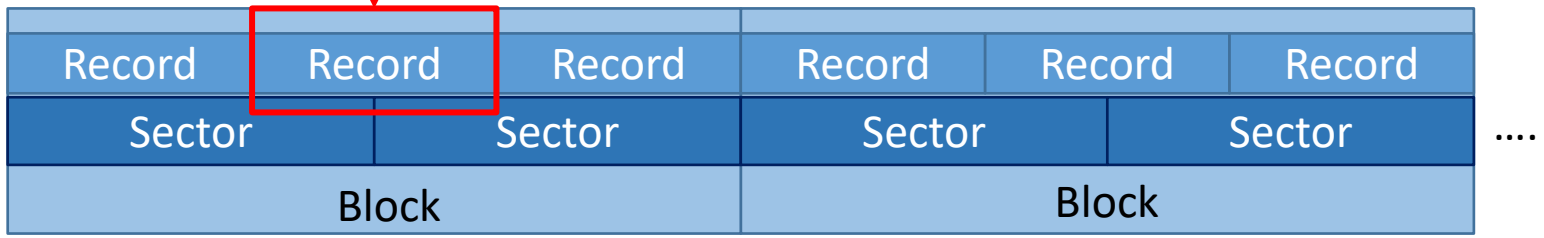
R : 레코드 크기 (fixed or variable)
B : 블록 크기



Sector : 디스크 읽기 쓰기의 최소단위 **Block :** 디스크 <-> 메인 메모리 사이의 데이터 전송의 최소 단위

Blocking 을 이용한 record read-write

이 record를 읽어 내용을 판독한
다음 내용을 변경하고 싶다.



1. Block 단위 Read :

- 읽으려는 Record 하나가 아닌 Record가 포함된 Block 전체를 Main memory로 전송한다.

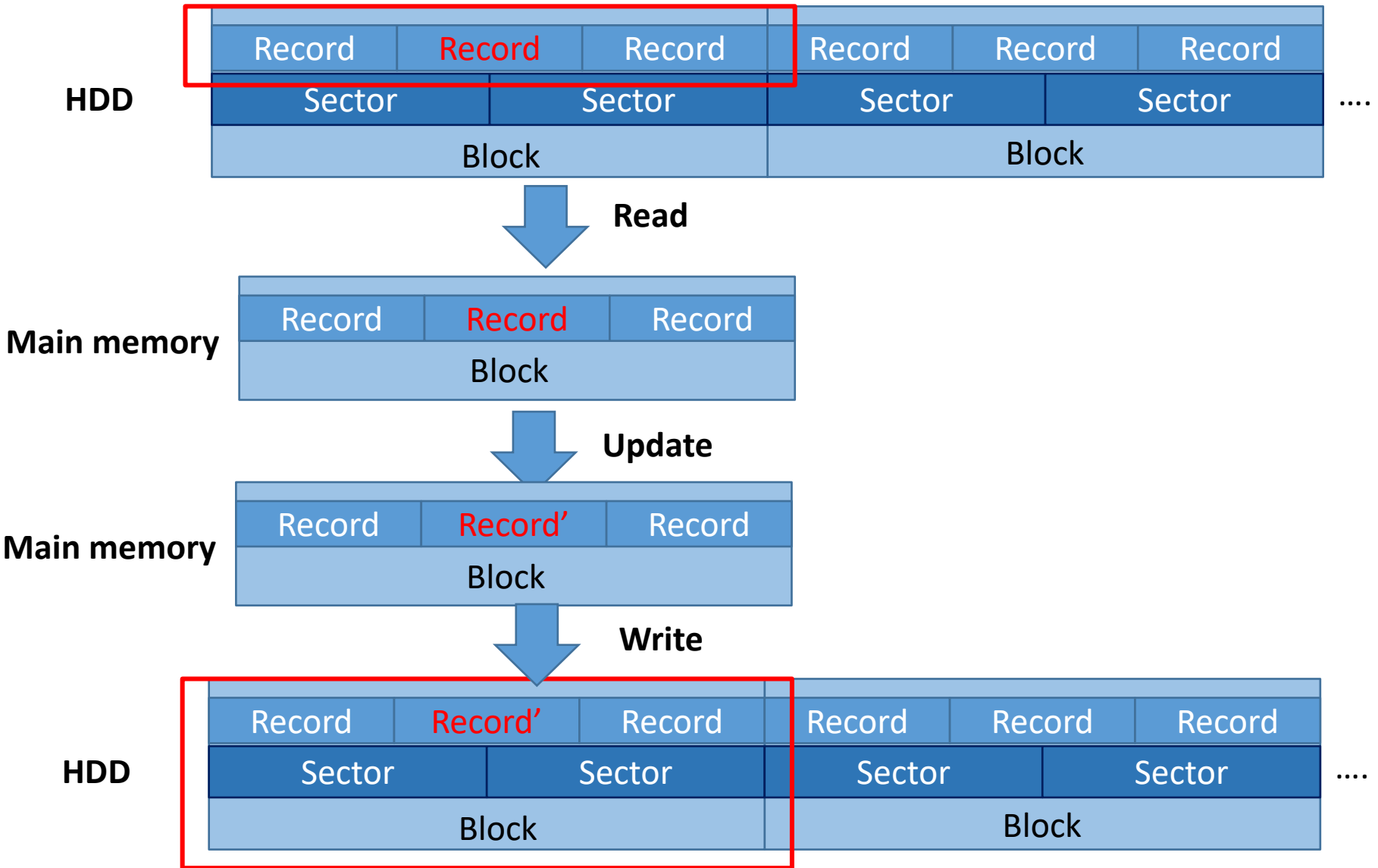
2. Main memory에서 Record 내용 변경 :

- Main memory에 전송해온 Block 중에서 변경할 Record의 내용을 변경한다.

3.Block 단위 Write :

- Main memory에서 변경한 Record를 포함한 Block을 Hard Disk로 전송하여 저장한다.

Blocking 을 이용한 record read-write (Cont'd)



2.4. Structured Data를 File로 HDD에 저장할 때 고려해야 할 요소들

Structured Data를 File로 HDD에 저장할 때 고려해야 할 요소들

- 이미 어느정도 만들어진 Data를 HDD에 저장하는 File을 설계할 때 고려해야 할 요소를 생각해 본다.
  - 전제: Blocking 을 사용하여 File에 Structured Data를 저장한다.

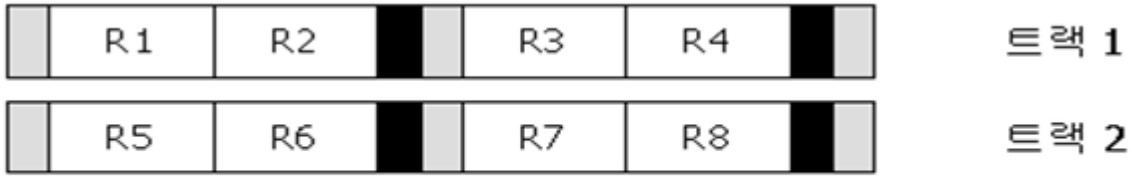
상품번호	이름	가격 (원)	판매한 개수	재고 수
과일_1	사과	3,000	100	200
과일_2	바나나	1,000	150	150
과일_3	딸기	5,000	80	120
과일_4	감	2,500	90	110

- 위 표와 같이 어느 정도 데이터가 있다.
- 표의 Record는 운용해 가면서 미래에 늘어날 수도, 줄어들 수도 있다고 가정한다.

Blocking 방법

Blocking은 Main memory와 I/O 효율을 위해 여러 개의 record를 하나의 Block에 저장시키는 것이다.
여러 개 의 record를 하나의 Block에 저장시키는 방법은 여러 가지가 있다.

- Fixed Length Blocking
  - 블록에 저장되는 Record의 길이가 Record 마다 모두 같다. (= 길이가 고정 (Fixed))
  - 경우에 따라서는 Block안에 빈 공간이 발생할 수 있다. (어떤 경우에 발생하는가?)



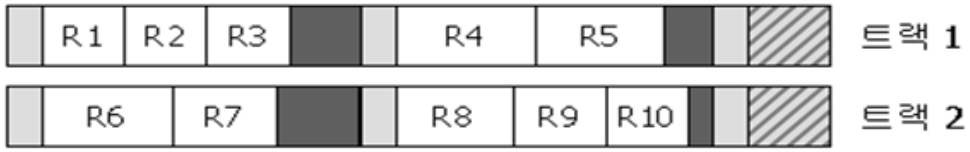
고정 길이 블로킹

Blocking 방법

- **Variable Length Blocking (가변 길이 Blocking)**
  - Record 사이의 길이가 서로 다른 Record가 Block안에 저장될 수 있다.
  - 이름이 Variable Length Blocking임에도 불구하고 블록의 길이는 내용물에 관계없이 모두 같다.
  - Block의 크기는 모두 같은데, Record의 길이가 다르면 Block의 공간에 빈 공간이 발생할 수 있다.
    - 빈공간을 그대로 두느냐
    - Record를 분할하여 인접 Block의 빈 공간에 채워 넣느냐에 따라
    - Variable Length Blocking **without spanning** 과
    - Variable Length Blocking **with spanning**으로 분류할 수 있다.

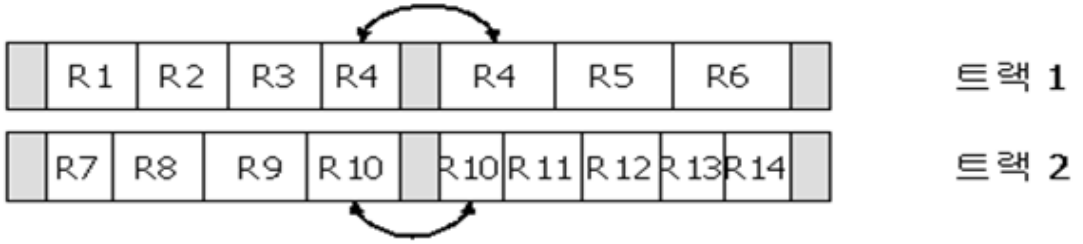
Blocking 방법

- **Variable length Blocking without spanning**
  - 가변 길이 레코드가 저장되고 Block 안의 남는 공간은 그대로 둠.
  - Fixed length block (with no spanning)



비신장 가변 길이 블로킹

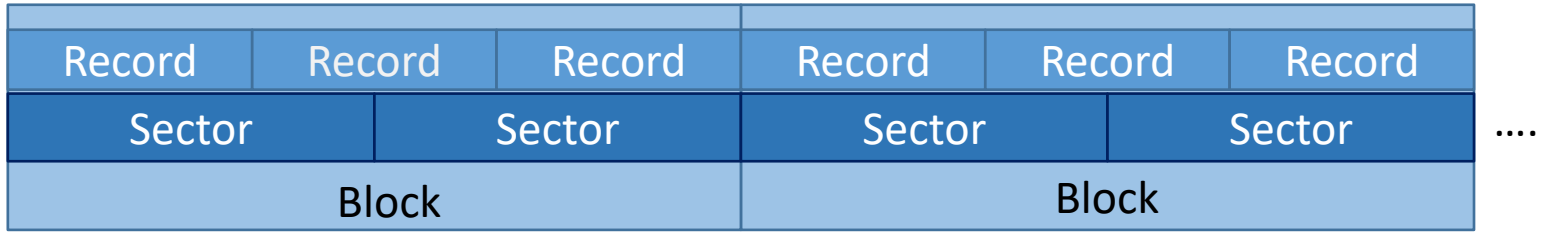
- **Variable length Blocking with spanning**
  - 가변 길이 레코드가 Block에 저장됨.
  - Block안의 남는 공간을 활용하기 위해 하나의 Record를 인접한 두 Block에 걸쳐 저장할 수 있다.



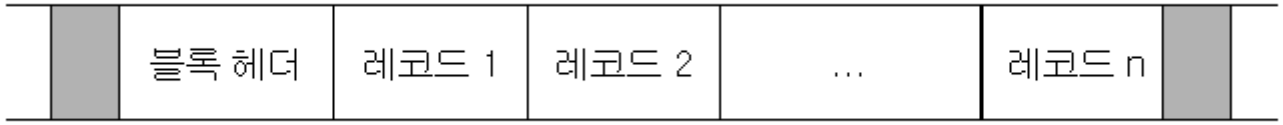
신장된 가변 길이 블로킹

Blocking 구현: Block 구분

Disk에서 필요한 Block을 쉽게 검색하려 할 때, 각 Block에 대한 요약 정보가 같이 저장되어 있으면 쉽게 검색할 수 있다.



- **블록 헤더(block header)**
  - 블록 내의 레코드 수, 블록 ID, 블록 수정 날짜, 블록 내에서의 레코드 시작점과 끝점 식별 등 정보 저장



블록 헤더가 추가된 블록 구조

Blocking 구현: Record 구분

Block안에 Record가 여러 개 저장되어 있으므로, Record 하나하나를 구별 가능하도록 Record에 관한 정보도 저장하여야 한다.

- **Fixed Length Blocking에서 Record 구분**
  - Record 길이만 알면 Record 구분 가능
  - Record길이를 Block Header에 저장

- **Variable Length Blocking에서 Record 구분**
  - Record 사이에 분리 표시(Record 끝 마크; end-of-record marker)
  - 각 Record 앞에 길이 지시자(length indicator)
  - Block Header에 위치 테이블(position table)

블록 헤더가 추가된 블록 구조

Record 설계 – 요건

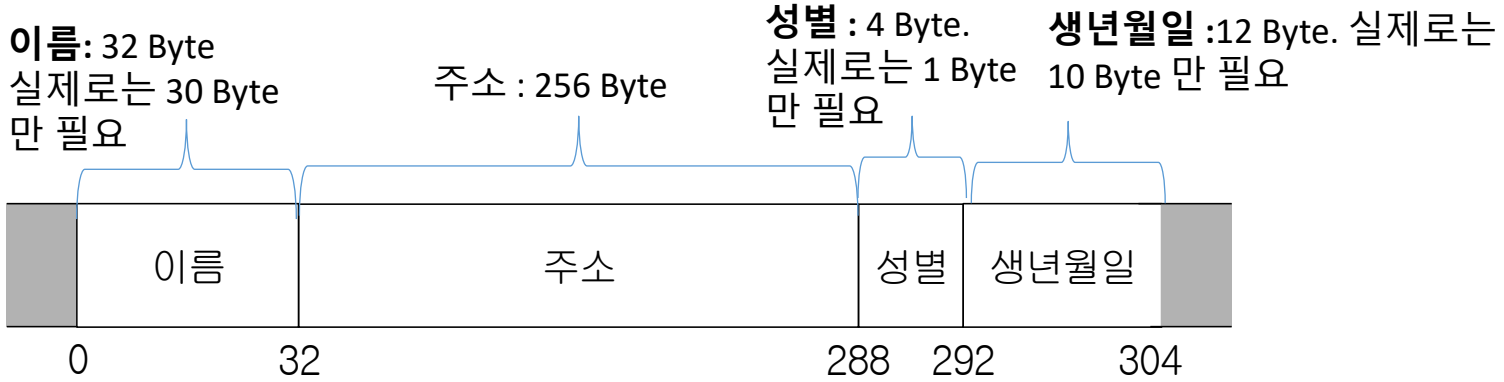
- 요건 1: Data Field를 구별할 수 있도록 설계하여야 한다.
  - Record는 Data instance의 특징적인 요소를 나타내는 Data Field로 이루어져 있다.
  - 하나의 Record의 Data Field를 (사람은) Table을 그려 구분한다.
    - 테이블의 구조라는 정보를 사용하여 구분
  - Disk에 저장되는 정보에도 Record를 구성하는 Data Field를 구별할 수 있는 정보를 저장해야 한다.
- 요건 2: 하나의 Record는 Main memory에서 쉽게 접근할 수 있게 저장 되어야한다.

상품번호	이름	가격 (원)	판매한 개수	재고 수
과일_1	사과	3,000	100	200
과일_2	바나나	1,000	150	150
과일_3	딸기	5,000	80	120
과일_4	감	2,500	90	110

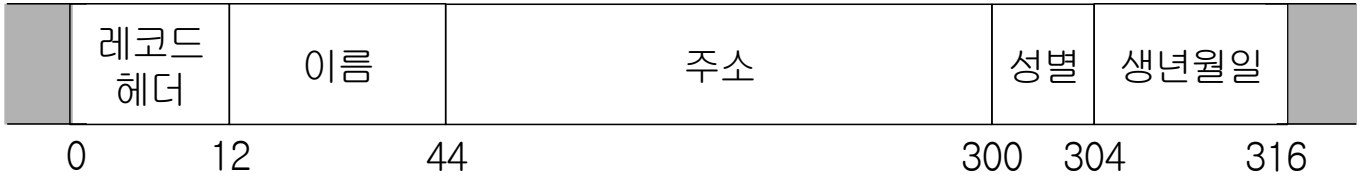
Remind: 위 테이블을 사용하여 Record와 Data field를 설명해 보아라.

Record 설계 (Cont'd)

- **레코드 설계 시 고려 사항**
  - 메인 메모리 바이트 주소(4의 배수(32 bit 컴퓨터) 혹은 8의 배수 (64bit 컴퓨터)) 특성
  - 각 필드는 Record의 오프셋(offset)으로 표현 가능
  - 위 필드 위치 정보 (Record Offset)를 Record header에 저장.



(a) 학생 레코드 레이아웃

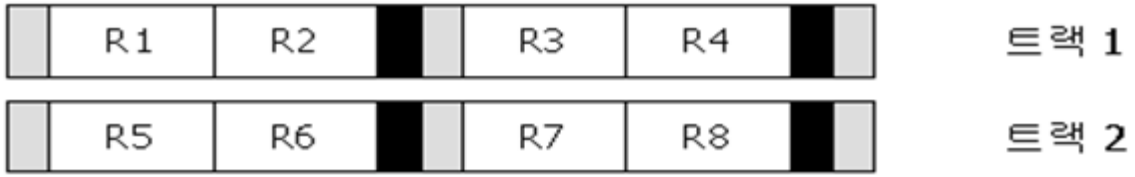


(b) 레코드 헤더가 추가된 학생 레코드 레이아웃

Blocking 설계 시 고려 사항 1 – 최대 Blocking Factor 예측

- 이미 어느정도 존재하는 데이터를 File로 만들 경우 Blocking 기법을 사용하여 저장하기 때문에 Blocking Factor 를 결정하여야 한다.

- **적재 밀도(loading density)**
  - 갱신(삽입)을 위해 Block 내에 여유공간을 확보해 둔다.
  - 실제 Record가 저장된 공간과 빈 공간 (Free space)을 포함한 총 공간과의 비율
  - **파일을 처음 생성할 때의 밀도**이다.



고정 길이 블로킹

- **균형 밀도(equilibrium density)**
  - 레코드 자체의 확장과 축소
  - **상당히 긴 기간 동안 시스템을 운용하고 안정시킨 뒤에 예상되는 적재 밀도**

Blocking 설계 시 고려 사항 2 – Locality 고려

- **Locality (집약성)**
  - 상호 연관된 레코드들의 물리적 근접성
  - 같은 블록, 트랙, 실린더