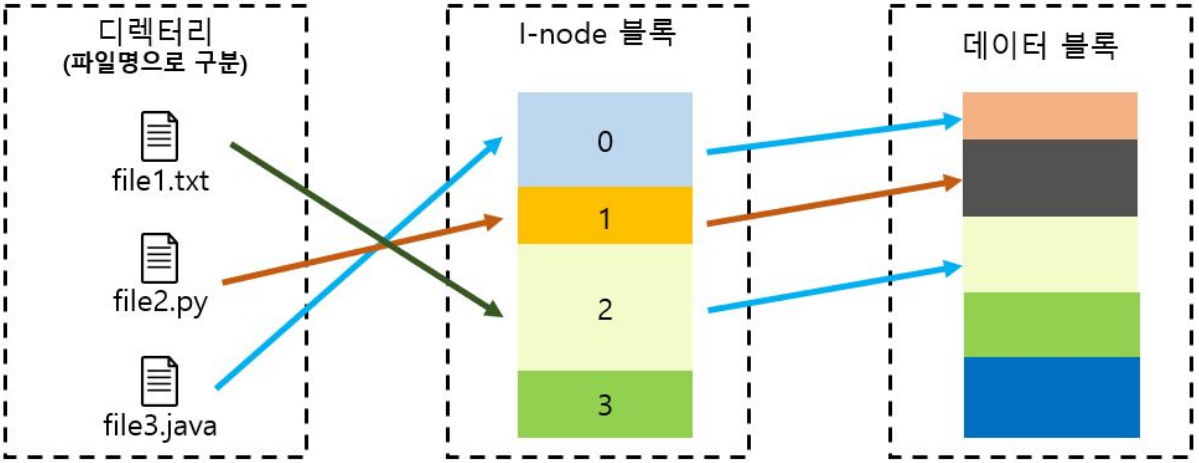
5. File 구성요소와 종류

Chapter 5 File

파일의 구성요소

- 리눅스에서는 모든 처리과정을 파일 단위로 구성되며, 계층적인 구조의 특성을 지닌다.
- 3요소
    - **파일 이름, I-node, 데이터 블록**



- **파일 이름**
    - 사용자가 파일에 접근할 때 사용
- **I-node**
    - 외부적으로 번호로 표시
    - 내부적으로 2부분으로 나누어 정보를 저장
        - 파일 정보를 저장하는 부분
        - 데이터 블록의 주소를 저장하는 부분
    - 파일의 I-node 번호는 ls -i 명령으로 확인
- **데이터 블록**
    - 실제로 데이터가 저장되는 부분

파일의 종류

리눅스에서는 하드웨어도, 디렉터리도 하나의 파일로 인식하므로 파일의 종류가 다양하다.

- 파일의 종류 확인
    - ls -l, ls -F

일반 파일(Regular File)

- 리눅스에서 대부분을 차지

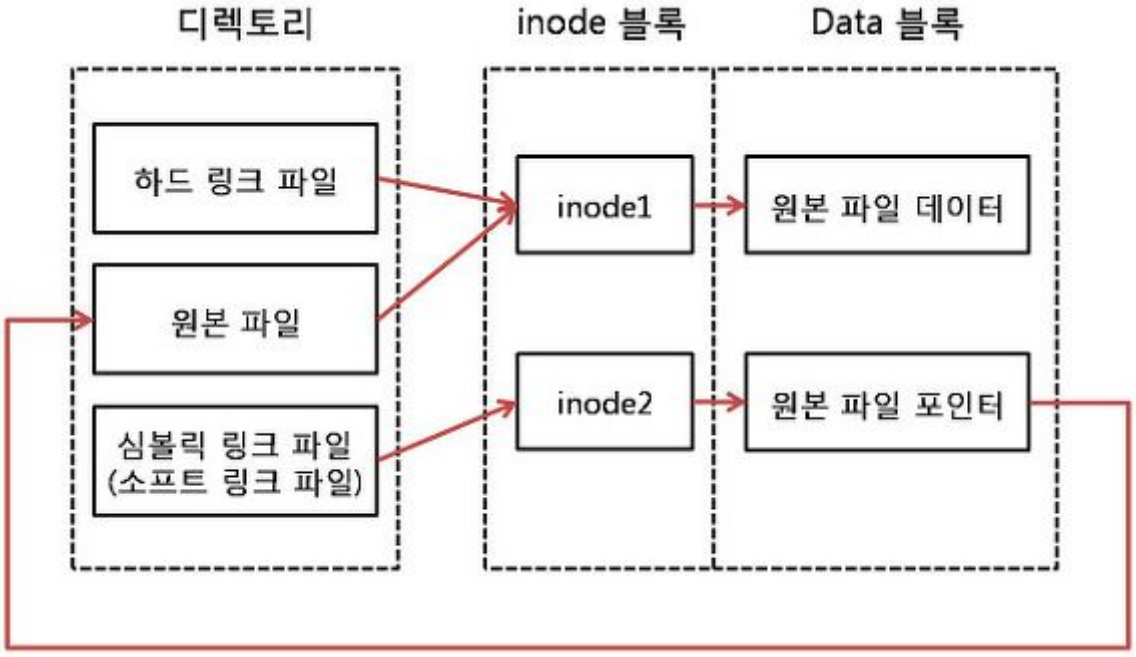
- 정보를 저장하는 역할

디렉터리 파일(Directory File)

- 파일 시스템에서의 디렉터리와 특수 파일로 서로 연관되어 있는 파일들을 하나의 그룹으로 만들어 저장하도록 구분되어 있는 공간을 의미
- 디렉터리는 I-node에 대한 포인터만 저장하기 때문에 파일의 이름과 그 파일의 I-node 번호를 매핑하는 항목의 리스트로만 구성

링크 파일(Link File)

- 여러 개의 이름이 하나의 I-node에 연결되므로 실제 파일이나 디렉터리 혹은 또다른 링크를 가르키도록 연결
- **Symbolic Link**
  - 리눅스에서의 일반적 링크로, 다른 표현으로는 Soft Link라고도 한다.
  - 원본 파일은 그대로 둔체 사용하기 위해 그 파일을 가리키는 기능을 수행
  - 윈도우에서의 바로 가기 기능과 유사
  - 심볼릭 링크를 삭제해도 원본 파일에는 아무런 영향을 미치지 않는다.
- **Hard Link**
  - 원본 파일을 복사한 다음 사본을 만드는 과정을 의미
  - 심볼릭 링크와 마찬가지로 링크로 접근하거나 원본에서 파일의 내용을 수정하였다면 원본과 하드링크된 파일이 모두 수행되어 항상 같은 파일의 내용을 유지



디바이스 파일(Device File)

- **Character Device File**
  - 주로 터미널, 프린터, 플로피들과 보조 기억장치 등을 설치할 때 사용되며, 시스템의 I/O 버퍼를 사용하지 않고 바이트 단위로 데이터를 입출력
- **Block Device File**
  - HDD, FDD, Tape Drive, 광자기 드라이브와 같은 보조 기억장치들을 설치할 때 사용되며, 수십 혹은 수백 바이트 크기인 블록 단위로 데이터를 입출력

파이프 파일(Pipe File)

- 한 프로그램의 출력을 중간 파일없이 다른 파일의 입력으로 바로 보내는 파일을 의미
- 파이프(|) 기호 왼쪽 명령의 출력을 오른쪽 명령어의 입력으로 보냄
- 파이프 라인은 하나 이상의 파이프로 구성되며, 파이프에서의 데이터는 선입선출 방식 이뤄짐

소켓 파일(Socket File)

- 네트워크의 입출력을 담당하는 API(Application Program Interface)로 두 호스트 컴퓨터 사이의 정보를 전달
- 물리적으로 연결된 네트워크 상에서의 데이터 송수신에 사용할 수 있는 소프트웨어적 장치를 의미

파일과 디렉터리 이름의 규칙

- /를 사용할 수 없다.
- 알파벳, 숫자, 붙임표(-), 밑줄(_), 마침표(.)만 사용한다.
- 공백 문자, *, |, “, ‘, @, #, $, %, ^, & 등을 사용하면 안된다.
- 대소문자를 구별하여 다른 글자로 취급한다.
- .(마침표)로 시작하면 숨김파일로 간주한다.

Symbolic Link와 Hard Link 생성

- **ln 명령**

기능	심볼릭 링크와 하드 링크를 만든다.
형식	**ln 옵션 대상 링크이름**
옵션	-s: 심볼릭 링크를 만든다.

- 실습
  - ls -l
  - ln ana* Hard.lnk
  - ln -s ana* Soft.lnk