> **Important**
>
> 데이터베이스는 엔티티가 아니다
> 데이터베이스는 세부사항이라서 아키텍처의 구성요소 수준으로 끌어올릴 수 없다

관계형 데이터베이스

- RDB
  - 데이터를 저장하고 접근하는 데 탁월한 기술이지만 **"세부사항"** 이다
  - 애플리케이션 유스케이스는 DB의 방식을 알아선 안되며 관여해서도 안된다
  - 데이터가 테이블 구조를 가진다는 사실은 오직 아키텍처의 외부 원에 위치한 최하위 수준의 유틸리티 함수만 알아야한다

데이터베이스 시스템은 왜 이렇게 널리 사용되는가?

- **"디스크"**
  - 지난 반세기 동안 회전식 자기 디스크는 데이터 저장소의 중심
  - **느리다는 단점**
  - 디스크의 시간 지연을 완화하기 위해 색인, 캐시, 쿼리 계획 최적화가 필요했다
  - 또한 데이터를 표현하는 표준방식이 필요했는데, 색인, 캐시, 쿼리 계획 최적화에서 작업 대상이 어떤 데이터인지 알 수 있어야했기 때문이다
  - 파일 시스템
    - document 기반(문서 기반)
    - 일련의 문서를 이름을 기준으로 저장하거나 조회할 때 잘 동작
    - 내용을 기준으로 검색 시 큰 도움이 되지 않음
  - DB 시스템
    - 내용을 기반으로 레코드를 자연스럽고 편리하게 찾는 방법을 제공
    - 정형화되지 않은 문서를 저장하고 검색하는 데 대체로 부적합

디스크가 없다면 어떻게 될까?

- 디스크가 모두 사라지고 모든 데이터를 RAM에 저장한다면?
  - SQL, 파일 구조를 만들어 디렉토리를 통한 접근 방식이 아닌 **데이터 구조로 체계화해 저장** (linkedlist, hash table, stack, queue 등)

세부사항

- 데이터베이스는 메커니즘에 불과하며, 디스크 표면과 RAM 사이에서 데이터를 이리저리 옮길 때 사용할 뿐이다

하지만 성능은?

- 데이터 저장소 측면에서 성능은 완전히 캡슐화해 업무 규칙과는 분리할 수 있는 관심사
- 데이터를 빠르게 넣고 빼는 것은 저수준의 관심사