[DB] 파일시스템과 데이터베이스의 차이점

파일 시스템

> 파일 시스템은 운영체제에서 파일과 디렉토리를 관리하는 방식을 의미한다.

파일 시스템의 특징

- 데이터는 **파일** 단위로 저장된다.
- 각 파일은 고유한 이름을 가진다.
- 데이터 간의 관계를 명시적으로 표현하지 않는다.
- 파일 시스템은 간단하고 작은 규모의 데이터 저장 및 관리에 유용하다.
- 데이터의 일관성, 무결성, 보안 등을 수동으로 관리해야 한다.
    - 중복 데이터 관리 어려움
    - 검색과 데이터 조회 성능이 낮음
    - 다중 사용자 환경에서 충돌 가능성이 있음

대표적인 예시

- Windows의 NTFS
- macOS의 HFS+
- Linux의 ext4

데이터베이스 관리 시스템(DBMS)

> DBMS는 데이터를 효율적으로 저장, 관리, 검색하기 위한 소프트웨어 시스템이다.

DBMS의 특징

- 데이터는 레코드(행)와 필드(열)로 구성된 테이블 형태로 저장된다.
- 데이터 간의 관계를 명확하게 정의하고 관리한다.
- 복잡한 데이터 조작과 질의를 지원하여 복잡한 작업을 쉽게 수행할 수 있다.
- 데이터의 일관성, 무결성, 보안을 자동으로 관리한다.
- 트랜잭션 관리 **병행성 제어**를 통해 동시에 여러 사람이 동시에 데이터에 접근할 때 안전하게 데이터를 관리할 수 있다.
- ACID 보장
    - 원자성(Atomicity): **트랜잭션** 내 작업이 모두 수행되거나 전혀 수행되지 않음을 보장
    - 일관성(Consistency): 데이터베이스가 **무결성**을 유지하도록 보장
    - 격리성(Isolation): 여러 트랜잭션이 **독립적**으로 수행되도록 보장
    - 지속성(Durability): 트랜잭션이 완료되면 변경사항이 **영구**적으로 저장됨

파일 시스템과 DBMS의 차이점

구분	데이터 구조	데이터 관리	복잡한 작업	보안 및 무결성	용도	동시성 제어
파일시스템	데이터를 파일 단위로 저장	데이터 간의 관계, 일관성을 사용자가 수동적으로 직접 관리	어려움	수동으로 관리	작은 규모의 데이터 저장 및 관리	파일 단위 잠금 방식으로 병행 처리가 어렵고, 다중 사용자 환경에서 충돌 발생 가능성이 있음
DBMS	데이터를 테이블 구조로 저장	데이터 간의 관계와 일관성을 DBMS가 자동으로 관리	데이터 조작과 질의 지원	자동으로 관리	큰 규모의 데이터 저장 및 관리, 복잡한 작업	병행성 제어를 통해 여러 사용자가 데이터에 접근하고 조회할 수 있음

면접용 답변

파일 시스템은 파일 단위로 데이터를 저장하며, 데이터 간의 관계를 정의하지 않습니다. 따라서 간단한 데이터 저장에는 적합하지만, 데이터 중복 관리나 검색 성능 저하, 다중 사용자 환경에서의 충돌 가능성과 같은 한계가 존재합니다. 반면, DBMS는 테이블 구조로 데이터를 저장하며 데이터 간의 관계를 정의하고 ACID 원칙을 기반으로 트랜잭션과 병행제어 기능을 제공합니다. 이를 통해 데이터의 일관성과 무결성을 보장하며, 대규모 데이터 관리 환경에 유용합니다.

[DB] DBMS는 뭘까요? 특징에 대해 설명해주세요.

DBMS는 뭘까요? 특징에 대해 설명해주세요.

면접 답변

DBMS는 데이터를 **효율적으로 저장하고 관리**할 수 있도록 지원하는 소프트웨어 시스템입니다. 특히 RDBMS는 RDB를 관리하는 시스템으로, 데이터를 테이블 형태로 저장하며, 테이블간의 관계를 명확하게 정의하여 일관성 있는 데이터 관리를 지원합니다. DBMS는 ACID 원칙을 기반으로 트랜잭션과 병행제어 기능을 제공합니다. 이를 통해 데이터의 무결성, 일관성을 보장하며 동시 접근 시에도 데이터 충돌을 방지할 수 있습니다. 따라서 대규모 데이터 관리에 유리합니다.

[DB] 데이터베이스의 특징에 대해 설명해주세요

데이터베이스

> 데이터베이스는 구조화된 데이터의 조직화된 모음.

데이터베이스의 특징

- 데이터베이스는 일반적으로 **DBMS**에 의해 제어되며, 관계형 데이터베이스에서는 RDBMS가 사용된다.
    - DBMS의 예시: MySQL, Oracle, ...
- 일반적인 유형의 데이터베이스는 데이터 처리 및 데이터 쿼리를 효율적으로 수행하기 위해 **테이블**의 형태로 데이터를 저장한다.
- 대부분의 데이터 베이스는 데이터 조작에 SQL(Structured Query Language)을 사용한다.

SQL(Structured Query Language)

> SQL은 데이터를 쿼리, 조작 및 정의하고 제어를 제공하기 위해 거의 모든 관계형 데이터베이스에서 사용하는 프로그래밍 언어이다.

데이터베이스의 종류

계층형 데이터베이스

> 폴더와 파일 등의 계층 구조로 데이터를 저장하는 방식

- 데이터의 관계를 트리 구조로 정의, 부모-자식 형태 => 1:N 만 가능
- 종류: IBM의 IMS, Windows Registry, ...
- 장점
    - 데이터 접근 속도 빠름
    - 데이터 사용량 예측이 쉬움
- 단점
    - 상하 종속적인 관계로 초기 세팅 후 변경이 어렵고, 복잡한 쿼리가 힘들다.

관계형 데이터베이스

> 행과 열을 가지는 테이블 형식으로 데이터를 저장하는 데이터베이스

- SQL을 이용하여 관리 및 접근
- 종류: Oracle, DB2, SQL Server, Mysql, ...
- 장점:
    - 높은 범용성과 성능
    - 데이터 무결성 보장
    - 명확한 스키마와 테이블간의 관계 정의
    - 검색 속도가 빠름
    - 데이터 중복을 줄임
- 단점:
    - 수직 확장을 통해 성능을 향상시키며, 수평확장은 샤딩등의 추가적인 기술이 필요하다.

NoSQL

> NoSQL은 저장 방식이 유연하며, 문서형, 키-값, 열 지향, 그래프 등의 여러 유형이 있다.

- 스키마가 없고, 관계도 없다.
- 정확한 데이터 요구 사항을 알 수 없거나 관계를 맺는 데이터가 **자주 변경**될 때 사용된다.

RDB와 NoSQL의 차이점

- NoSQL은 비정형 데이터(메시지 텍스트, 음성, 이미지 등)을 다룰 수 있다.
- NoSQL은 수평 확장이 가능하며 분산형 컴퓨팅(클라우드 환경)에 적합하다.
- 예시: MongoDB, Redis, ...
- 장점:
    - 스키마가 없어 데이터 추가가 유연하다.
    - 속도가 빠르다.
- 단점:
    - 데이터 중복이 생길 수 있다.

면접 답변

데이터 베이스는 구조화된 데이터를 저장하고 관리하는 시스템입니다. 일반적으로 데이터베이스는 DBMS에 의해 제어되며, 관계형 데이터베이스에서는 RDBMS가 주로 사용됩니다. 관계형 데이터베이스는 테이블 형태로 데이터를 저장하고 SQL을 통해 데이터를 조작합니다. 반면 NoSQL은 Key-Value, 문서, 그래프 등의 다양한 형태로 데이터가 저장되어 스키마 없이 유연한 확장이 가능합니다. RDB는 데이터 무결성과 정규화를 보장하는 반면, NoSQL은 수평확장에 용이하고 대량의 비정형 데이터를 처리하는데 적합합니다.

(*정규화: 데이터를 논리적으로 분리하여 중복을 줄이고, 데이터 무결성을 유지하는 과정

[DB] 스키마가 뭘까요? 3단계 데이터베이스 구조에 대해 설명해주세요.

데이터베이스 스키마

> 데이터베이스 스키마는 관계형 데이터베이스에서 **데이터의 구조를 정의하는 설계도**를 의미한다. 이는 테이블, 필드, 데이터 유형, 관계 및 제약조건 등을 포함하여 **데이터를 어떻게 정리하고 관리**할지에 대한 규칙을 설정한다.

(스키마의 목적은 명확히 데이터 구조를 정의하는 것!!)

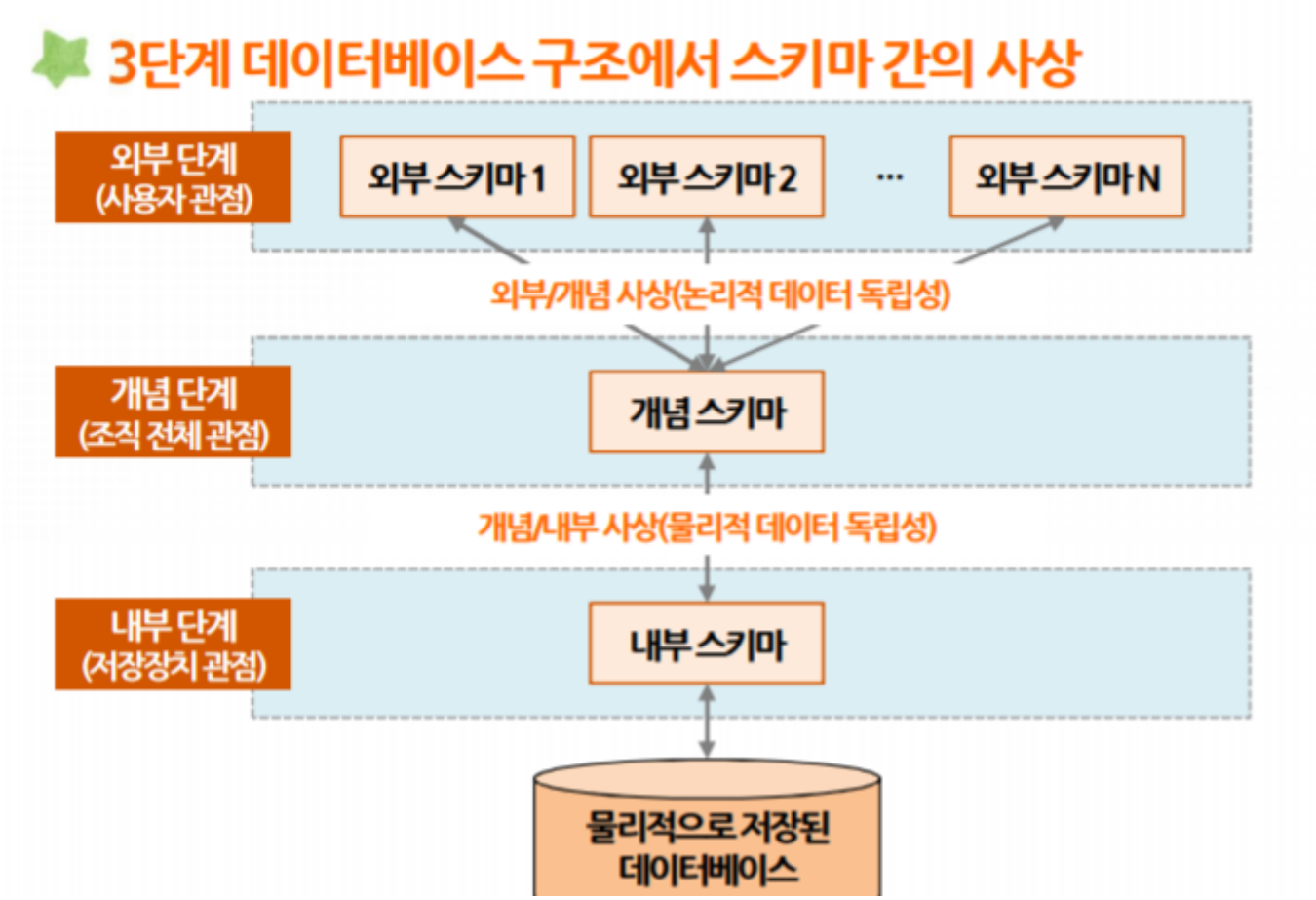
스키마에 포함되는 것들

- 개체의 특성을 나타내는 **속성**
- 속성들의 집합으로 이루어진 **개체**
- 개체 사이에 존재하는 **관계**에 대한 정의
- 속성, 개체, 관계들이 유지해야할 **제약조건**들

3단계 데이터베이스 구조

> 데이터베이스를 쉽게 이해하고 이용할 수 있도록 하나의 데이터베이스를 관점에 따라 세 단계로 나눈 것이다.

1. 외부 단계
2. 개념 단계
3. 내부 단계



1. 외부 단계 - 개별 사용자 관점

> 데이터 베이스 하나에 외부 스키마 여러 개가 존재할 수 있다.

외부 스키마 = 서브 스키마

- 외부 단계는 사용자와 응용 프로그램에 맞게 필요한 데이터베이스 구조를 정의하며, 각 사용자는 자신이 필요한 데이터만을 볼 수 있다.
- 각 사용자가 생각하는 데이터베이스의 모습으로, 논리적 구조로 사용자마다 다른 모습이다.

2. 개념 단계 = 조직 전체의 관점/논리적 구조와 유사

> 개념 단계에서는 **데이터베이스 전체의 논리적 구조**를 정의하며, 하나의 데이터베이스에는 개념 스키마가 하나만 존재한다.

개념 스키마 = 스키마

- 개념 단계에서 데이터베이스 전체의 논리적 구조를 정의한 것
- 조직 전체의 관점에서 생각하는 데이터베이스 모습
- 전체 데이터베이스에서 **어떤 데이터**가 저장되는지, 데이터들 간에는 **어떤 관계**가 존재하고 **어떤 제약조건**이 존재하는지에 대한 정의뿐만 아니라, 데이터에 대한 **보안 정책**이나 **접근 권한**에 대한 정의도 포함 => (데이터를 효과적으로 관리하기 위한 다양한 기능 개념)

3. 내부 단계 - 물리적인 저장 장치의 관점 / 실질적인 내용

> 데이터 베이스 하나에 내부 스키마가 하나만 존재

내부 스키마 = 저장 스키마

- 전체 데이터베이스가 저장 장치에 실제로 저장되는 방법을 정의한 것
- 레코드 구조, 필드 크기, 레코드 접근 경로 등 물리적인 저장 구조를 정의

3단계 데이터베이스 구조 분류

설명	외부스키마	개념스키마	내부스키마
다른 명칭	서브스키마	스키마	저장스키마
대상	사용자, 응용 프로그래머의 관점	데이터베이스 관리자(DBA)의 관점	시스템 프로그래머, 시스템 설계자 관점
특징	응용 프로그램이 데이터베이스의 일부를 보는 관점	데이터베이스 자체의 전체적인 논리적인 구조	물리적 저장장치의 입장에서 본 실제 데이터베이스 구조
용도	사용자 뷰	전체적인 뷰	레코드의 물리적인 뷰

3단계 데이터베이스 구조의 사상 또는 매핑

데이터베이스를 3단계 구조로 나누고, 단계별로 스키마를 유지하며 스키마 사이의 대응관계를 정의하는 것이 목적이다.

1. 외부/개념 사상 - 응용 인터페이스

- 외부 스키마 - 개념 스키마의 대응 관계
- 외부/개념 사상은 응용 프로그램과 데이터베이스 시스템 간의 인터페이스 역할을 하며, 사용자가 필요로 하는 데이터를 개념 스키마의 구조에 맞춰 제공하는 기능을 한다.

2. 개념/내부 사상 - 저장 인터페이스

개념 스키마 - 내부 스키마의 대응 관계

데이터 독립성

- 데이터 독립성은 데이터베이스 시스템의 중요한 특성으로, 하위 스키마의 변경이 상위 스키마에 영향을 미치지 않도록 보장하는 원칙이다.
- 데이터베이스의 유연성과 유지보수 용이성을 높여준다.
- 파일 시스템의 종속성 문제를 해결한다.

논리적 데이터 독립성 (응용 인터페이스 독립성 유지)

- 개념 스키마가 변경되어도 외부 스키마가 영향을 받지 않는다.
- 개념 스키마가 변경되면 관련된 외부/개념 사상만 정확히 수정해주면 된다.

물리적 데이터 독립성 (저장 인터페이스 독립성 유지)

- 내부 스키마가 변경되어도 개념 스키마는 영향을 받지 않는다.
- 내부 스키마가 변경되면 관련된 개념/내부 사상만 정확히 수정해주면 된다.

데이터베이스 종류

데이터 사전 - 시스템 카탈로그

> 데이터베이스에 저장되는 데이터에 관한 정보, 데이터를 설명, 관리하기 위한 시스템 데이터베이스

메타 데이터

> 데이터에 대한 데이터

스키마, 사상 정보, 다양한 제약 조건 등을 저장

데이터 디렉토리

> 데이터 사전에 있는 데이터에 실제로 접근하는데 필요한 위치 정보를 저장하는 시스템 데이터베이스

사용자 데이터베이스

> 사용자가 실제로 이용하는 데이터가 저장되어 있는 일반 데이터베이스

면접 답변

스키마(Schema)는 데이터베이스의 구조를 정의하는 설계도로, 데이터베이스 내에서 데이터를 어떻게 저장하고 관리할지를 규명하는 역할을 합니다. 관계형 데이터베이스에서는 테이블, 필드, 데이터 형식, 관계 및 제약 조건 등을 포함하여 데이터를 논리적으로 구조화합니다. 이를 통해 데이터 간의 관계를 명확히 하고, 데이터 무결성 및 일관성을 유지할 수 있습니다.

3단계 데이터베이스 구조는 데이터베이스를 보다 효율적으로 이해하고 관리하기 위한 분류로, 크게 외부 단계, 개념 단계, 내부 단계로 나눕니다.

외부 단계는 사용자나 응용 프로그램이 필요로 하는 데이터베이스의 구조를 정의하며, 외부 스키마는 사용자가 보는 데이터베이스의 모습을 제공합니다. 각 사용자는 자신이 필요한 데이터만을 볼 수 있습니다.

개념 단계는 데이터베이스 전체의 논리적 구조를 정의하며, 하나의 데이터베이스에는 개념 스키마가 하나만 존재합니다. 이 단계에서는 데이터 간의 관계와 제약 조건을 정의하고, 데이터베이스 관리자가 보안 정책 및 접근 권한도 설정합니다.

내부 단계는 데이터가 물리적으로 저장되는 방식에 대한 정의입니다. 내부 스키마는 레코드 구조, 필드 크기, 데이터 접근 경로 등 저장 장치에서 데이터가 실제로 어떻게 관리되는지에 대한 세부 정보를 다룹니다.

[DB] 스키마가 뭘까요? 3단계 데이터베이스 구조에 대해 설명해주세요.

데이터베이스 스키마

> 데이터베이스 스키마는 관계형 데이터베이스에서 **데이터의 구조를 정의하는 설계도**를 의미한다. 이는 테이블, 필드, 데이터 유형, 관계 및 제약조건 등을 포함하여 **데이터를 어떻게 정리하고 관리**할지에 대한 규칙을 설정한다.

(스키마의 목적은 명확히 데이터 구조를 정의하는 것!!)

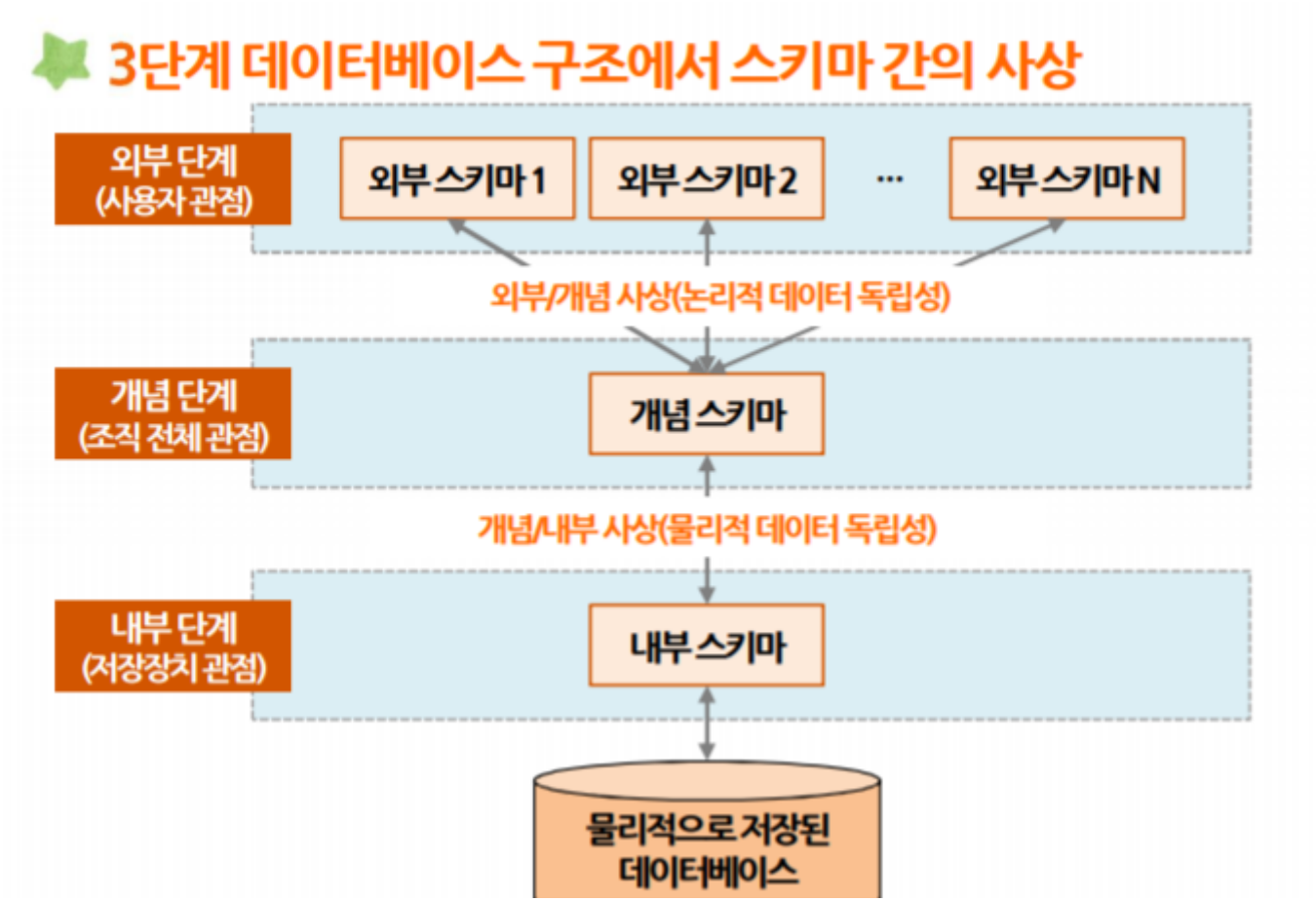
스키마에 포함되는 것들

- 개체의 특성을 나타내는 **속성**
- 속성들의 집합으로 이루어진 **개체**
- 개체 사이에 존재하는 **관계**에 대한 정의
- 속성, 개체, 관계들이 유지해야할 **제약조건**들

3단계 데이터베이스 구조

> 데이터베이스를 쉽게 이해하고 이용할 수 있도록 하나의 데이터베이스를 관점에 따라 세 단계로 나눈 것이다.

1. 외부 단계
2. 개념 단계
3. 내부 단계



1. 외부 단계 - 개별 사용자 관점

> 데이터 베이스 하나에 외부 스키마 여러 개가 존재할 수 있다.

외부 스키마 = 서브 스키마

- 외부 단계는 사용자와 응용 프로그램에 맞게 필요한 데이터베이스 구조를 정의하며, 각 사용자는 자신이 필요한 데이터만을 볼 수 있다.
- 각 사용자가 생각하는 데이터베이스의 모습으로, 논리적 구조로 사용자마다 다른 모습이다.

2. 개념 단계 = 조직 전체의 관점/논리적 구조와 유사

> 개념 단계에서는 **데이터베이스 전체의 논리적 구조**를 정의하며, 하나의 데이터베이스에는 개념 스키마가 하나만 존재한다.

개념 스키마 = 스키마

- 개념 단계에서 데이터베이스 전체의 논리적 구조를 정의한 것
- 조직 전체의 관점에서 생각하는 데이터베이스 모습
- 전체 데이터베이스에서 **어떤 데이터**가 저장되는지, 데이터들 간에는 **어떤 관계**가 존재하고 **어떤 제약조건**이 존재하는지에 대한 정의뿐만 아니라, 데이터에 대한 **보안 정책**이나 **접근 권한**에 대한 정의도 포함 => (데이터를 효과적으로 관리하기 위한 다양한 기능 개념)

3. 내부 단계 - 물리적인 저장 장치의 관점 / 실질적인 내용

> 데이터 베이스 하나에 내부 스키마가 하나만 존재

내부 스키마 = 저장 스키마

- 전체 데이터베이스가 저장 장치에 실제로 저장되는 방법을 정의한 것
- 레코드 구조, 필드 크기, 레코드 접근 경로 등 물리적인 저장 구조를 정의

3단계 데이터베이스 구조 분류

설명	외부스키마	개념스키마	내부스키마
다른 명칭	서브스키마	스키마	저장스키마
대상	사용자, 응용 프로그래머의 관점	데이터베이스 관리자(DBA)의 관점	시스템 프로그래머, 시스템 설계자 관점
특징	응용 프로그램이 데이터베이스의 일부를 보는 관점	데이터베이스 자체의 전체적인 논리적인 구조	물리적 저장장치의 입장에서 본 실제 데이터베이스 구조
용도	사용자 뷰	전체적인 뷰	레코드의 물리적인 뷰

3단계 데이터베이스 구조의 사상 또는 매핑

데이터베이스를 3단계 구조로 나누고, 단계별로 스키마를 유지하며 스키마 사이의 대응관계를 정의하는 것이 목적이다.

1. 외부/개념 사상 - 응용 인터페이스

- 외부 스키마 - 개념 스키마의 대응 관계
- 외부/개념 사상은 응용 프로그램과 데이터베이스 시스템 간의 인터페이스 역할을 하며, 사용자가 필요로 하는 데이터를 개념 스키마의 구조에 맞춰 제공하는 기능을 한다.

2. 개념/내부 사상 - 저장 인터페이스

개념 스키마 - 내부 스키마의 대응 관계

데이터 독립성

- 데이터 독립성은 데이터베이스 시스템의 중요한 특성으로, 하위 스키마의 변경이 상위 스키마에 영향을 미치지 않도록 보장하는 원칙이다.
- 데이터베이스의 유연성과 유지보수 용이성을 높여준다.
- 파일 시스템의 종속성 문제를 해결한다.

논리적 데이터 독립성 (응용 인터페이스 독립성 유지)

- 개념 스키마가 변경되어도 외부 스키마가 영향을 받지 않는다.
- 개념 스키마가 변경되면 관련된 외부/개념 사상만 정확히 수정해주면 된다.

물리적 데이터 독립성 (저장 인터페이스 독립성 유지)

- 내부 스키마가 변경되어도 개념 스키마는 영향을 받지 않는다.
- 내부 스키마가 변경되면 관련된 개념/내부 사상만 정확히 수정해주면 된다.

데이터베이스 종류

데이터 사전 - 시스템 카탈로그

> 데이터베이스에 저장되는 데이터에 관한 정보, 데이터를 설명, 관리하기 위한 시스템 데이터베이스

메타 데이터

> 데이터에 대한 데이터

스키마, 사상 정보, 다양한 제약 조건 등을 저장

데이터 디렉토리

> 데이터 사전에 있는 데이터에 실제로 접근하는데 필요한 위치 정보를 저장하는 시스템 데이터베이스

사용자 데이터베이스

> 사용자가 실제로 이용하는 데이터가 저장되어 있는 일반 데이터베이스

면접 답변

스키마(Schema)는 데이터베이스의 구조를 정의하는 설계도로, 데이터베이스 내에서 데이터를 어떻게 저장하고 관리할지를 규명하는 역할을 합니다. 관계형 데이터베이스에서는 테이블, 필드, 데이터 형식, 관계 및 제약 조건 등을 포함하여 데이터를 논리적으로 구조화합니다. 이를 통해 데이터 간의 관계를 명확히 하고, 데이터 무결성 및 일관성을 유지할 수 있습니다.

3단계 데이터베이스 구조는 데이터베이스를 보다 효율적으로 이해하고 관리하기 위한 분류로, 크게 외부 단계, 개념 단계, 내부 단계로 나눕니다.

외부 단계는 사용자나 응용 프로그램이 필요로 하는 데이터베이스의 구조를 정의하며, 외부 스키마는 사용자가 보는 데이터베이스의 모습을 제공합니다. 각 사용자는 자신이 필요한 데이터만을 볼 수 있습니다.

개념 단계는 데이터베이스 전체의 논리적 구조를 정의하며, 하나의 데이터베이스에는 개념 스키마가 하나만 존재합니다. 이 단계에서는 데이터 간의 관계와 제약 조건을 정의하고, 데이터베이스 관리자가 보안 정책 및 접근 권한도 설정합니다.

내부 단계는 데이터가 물리적으로 저장되는 방식에 대한 정의입니다. 내부 스키마는 레코드 구조, 필드 크기, 데이터 접근 경로 등 저장 장치에서 데이터가 실제로 어떻게 관리되는지에 대한 세부 정보를 다룹니다.

[DB] 데이터 독립성에 대해서 설명해주세요.

데이터 독립성

> 데이터 독립성은 데이터베이스 시스템의 중요한 특성으로, 하위 스키마의 변경이 상위 스키마에 영향을 미치지 않도록 보장하는 원칙이다.

- 데이터베이스의 유연성과 유지보수 용이성을 높여, 변경 사항이 다른 영역에 미치는 영향을 최소화 한다.
- 파일 시스템은 데이터 구조 변경 시 모든 응용 프로그램을 수정해야하는 종속성 문제가 발생했지만, 데이터베이스에서는 독립성을 통해 이러한 종속성 문제를 해결한다.

논리적 데이터 독립성 (응용 인터페이스 독립성 유지)

> 논리적 데이터 독립성이란, 개념 스키마(데이터베이스의 논리적 구조)가 변경되더라도, 외부 스키마(사용자 관점에서 보는 데이터 구조)에 영향을 주지 않는 것을 의미합니다.

- 예를 들어, 새로운 속성을 추가하더라도 기존 응용 프로그램이 영향을 받지 않도록 보장합니다.

물리적 데이터 독립성 (저장 인터페이스 독립성 유지)

- 내부 스키마가 변경되어도 개념 스키마는 영향을 받지 않는다.

> 물리적 데이터 독립성이란, 내부 스키마(데이터의 물리적 저장 방식)가 변경되더라도 개념 스키마(논리적 데이터 구조)에 영향을 미치지 않는 것을 의미한다.

- 예를 들어, 성능 최적화를 위해 저장소를 SSD로 변경하더라도 데이터베이스의 논리적 구조는 그대로 유지됩니다.

면접 답변

데이터 독립성은 데이터베이스 시스템의 중요한 특성으로, 하위 스키마의 변경이 상위 스키마에 영향을 미치지 않도록 보장하는 것입니다.

이를 통해 데이터베이스의 유지보수성과 확장성을 높이고, 기존의 파일 시스템에서 발생하던 데이터 종속성 문제를 해결할 수 있습니다.

데이터 독립성은 논리적 데이터 독립성과 물리적 데이터 독립성으로 나뉩니다.

논리 데이터 독립성은 개념 스키마의 변경이 외부 스키마에 영향을 주지 않게 합니다. 예를 들어, 테이블에 새로운 속성이 추가되더라도 기존 응용 프로그램이 정상적으로 작동할 수 있도록 보장합니다.

물리 데이터 독립성은 내부 스키마의 변경이 개념 스키마에 영향을 미치지 않도록 보장합니다. 예를 들어, 성능 최적화를 위해 하드디스크에서 SSD로 변경하거나, 데이터 저장 구조를 변경하더라도 개념 스키마는 그대로 유지됩니다.

[DB] RDBMS(관계형 데이터베이스 관리시스템)는 뭘까요?

RDBMS(Relational Database Management System)

> 데이터를 테이블 형태로 저장하고 테이블 간의 관계를 명확히 정의하는 데이터베이스를 생성, 수정, 관리 및 검색하는 소프트웨어 시스템이다.

RDBMS의 특징

- RDBMS는 SQL을 사용하여 데이터를 조작하고 검색한다.
- RDB는 행과 열로 이루어져 있다.
    - 열(속성): 특정 데이터 유형 의미
    - 행(튜플): 레코드 의미
- ACID 특성을 기반으로 병행 제어와 트랜잭션 제어를 제공하여 다중 사용자 접속 상황에서 데이터 충돌을 방지하고, 데이터의 무결성과 일관성을 보장
    - 원자성: 트랜잭션은 완전히 수행되거나 전혀 수행되지 않아야 함
    - 일관성: 트랜잭션 전 후의 데이터베이스의 일관성
    - 격리성: 여러 트랜잭션 동시 실행 시, 다른 트랜잭션과 격리되어 진행
    - 지속성: 트랜잭션 성공 이후 결과가 영구적으로 저장

트랜잭션

> 트랜잭션은 데이터베이스에서 수행되는 단일 논리적 작업 단위이다.

하나 이상의 데이터베이스 작업이 모두 성공하거나 실패할때까지 모두 적용되거나 모두 롤백되어야 하는 작업 그룹을 의미한다.

트랜잭션은 데이터의 무결성과 일관성을 유지하기 위해 사용된다.

면접 답변

RDBMS는 데이터를 관계형 데이터베이스인 RDB를 관리하는 소프트웨서 시스템으로, 데이터를 테이블 구조로 저장하고, 테이블의 관계를 명확히 정의하여 데이터를 효율적으로 생성, 수정, 검색하는 소프트웨어 시스템입니다.

RDBMS는 보통 SQL을 통해 RDB를 조작합니다.

또한 RDBMS는 ACID원칙을 기반으로 병행제어와 트랜잭션을 제공하여 다중 사용자 접속 환경에서 데이터 충돌을 방지합니다.

RDBMS의 트랜잭션은 데이터베이스에서 하나 이상의 SQL 연산을 묶어 하나의 논리적 작업 단위로 실행하는 개념입니다. 예를 들어, 은행 계좌 이체시, 출금과 입금이 하나의 트랜잭션으로 실행되어야 하며, 만약 하나라도 실패하면 전체 작업이 롤백되어 데이터 정합성이 지켜져야 합니다.

이러한 특징 덕분에 RDBMS는 기업의 대규모 데이터 관리, 금융 시스템, 전자 상거래 등에서 널리 사용됩니다.

[DB] 릴레이션이란?

릴레이션이란?

> 관계형 데이터베이스에서 데이터를 표로 저장하는 기본 단위이다. 릴레이션은 튜플과 속성으로 구성된다.

=> 즉, 릴레이션은 DB 테이블이다.

릴레이션의 특징

- 하나의 릴레이션에 포함된 튜플들은 모두 상이하다.
- 한 릴레이션에 포함된 튜플 사이에는 순서가 없다. (하지만 SQL의 ORDER BY로 특정 속성을 기준으로 정렬할 수 있다.)

릴레이션 스키마와 릴레이션 인스턴스에 대해서 설명해주세요.

릴레이션 스키마

> 릴레이션 스키마는 릴레이션에서 데이터를 저장하는 구조를 정의하는 설계도이다.

테이블의 이름, 속성 목록, 속성의 데이터 타입, 기본 키 및 제약 조건 등을 포함한다.

릴레이션 인스턴스

> 릴레이션 인스턴스는 릴레이션에 실제로 저장된 데이터의 집합을 의미한다.

릴레이션의 차수와 카니날리티에 대해 설명해주세요.

릴레이션의 차수

> 한 릴레이션 안에 있는 에트리뷰트(속성)의 수를 차수라고 한다.

- 0 < 차수
- ex) 고객번호, 이름, 전화번호, 집주소

카디날리티

> 릴레이션의 튜플의 개수이다.

- 0 <= 카디날리티
- 카디날리티는 시간이 지남에 따라 값이 게속 변한다.

[DB] 키(Key)에 대해서 설명해주세요. (슈퍼키, 후보키, 기본키, 대리키, 외래키)

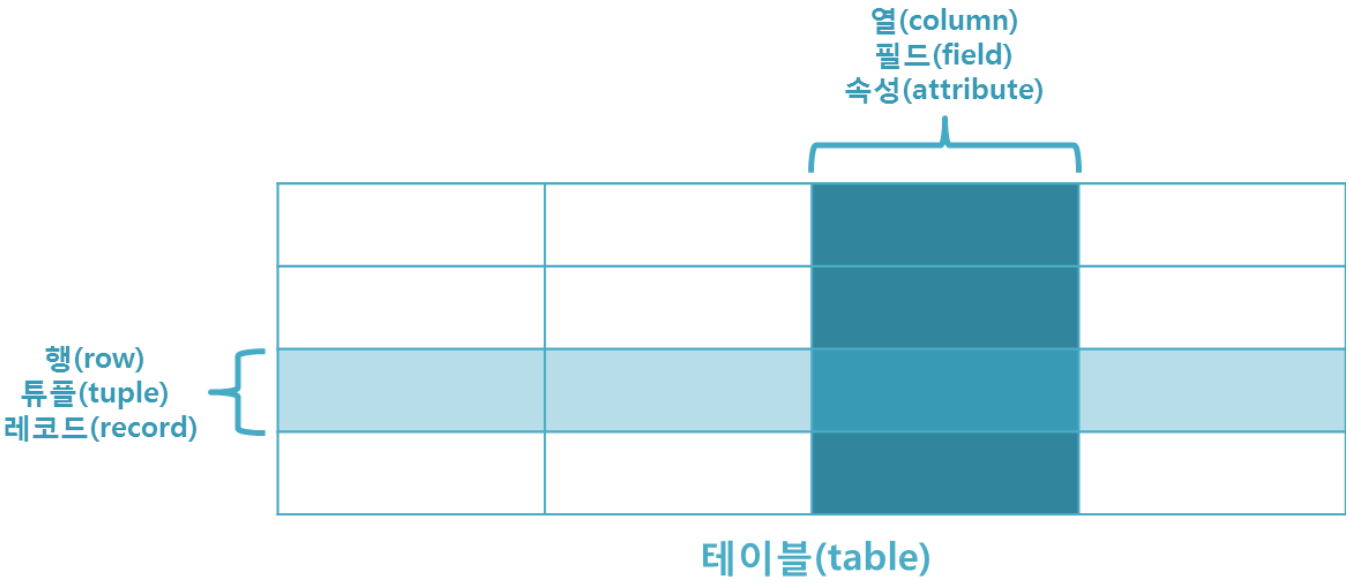
키(Key)

> 데이터데이스에서 각 행을 구분하는 유일한 식별자이다.

- 일반적으로 키는 테이블에서 하나 이상의 열(속성)로 구성된다.
- 키는 데이터 정합성 유지, 검색, 수정, 삭제 등의 작업을 할 때 중요한 역할을 한다.

데이터 정합성

> 데이터가 올바르게 유지되고 일관성 있게 유지되는 것을 의미한다. 데이터베이스 내의 모든 데이터가 정확하고 일관되게 유지되도록 하는 것을 목표로 한다.



데이터베이스 키 종류

1. 기본키

> 테이블에서 각 행을 유일하게 식별할 수 있는 키

- NULL 허용 X

2. 후보키

> 기본 키가 될 수 있는 후보로 선정된 속성 또는 속성의 집합

- NULL 허용 X

3. 대리키

> 기본 키로 사용되징 않는 후보키의 집합을 의미한다.

- NULL 허용 O

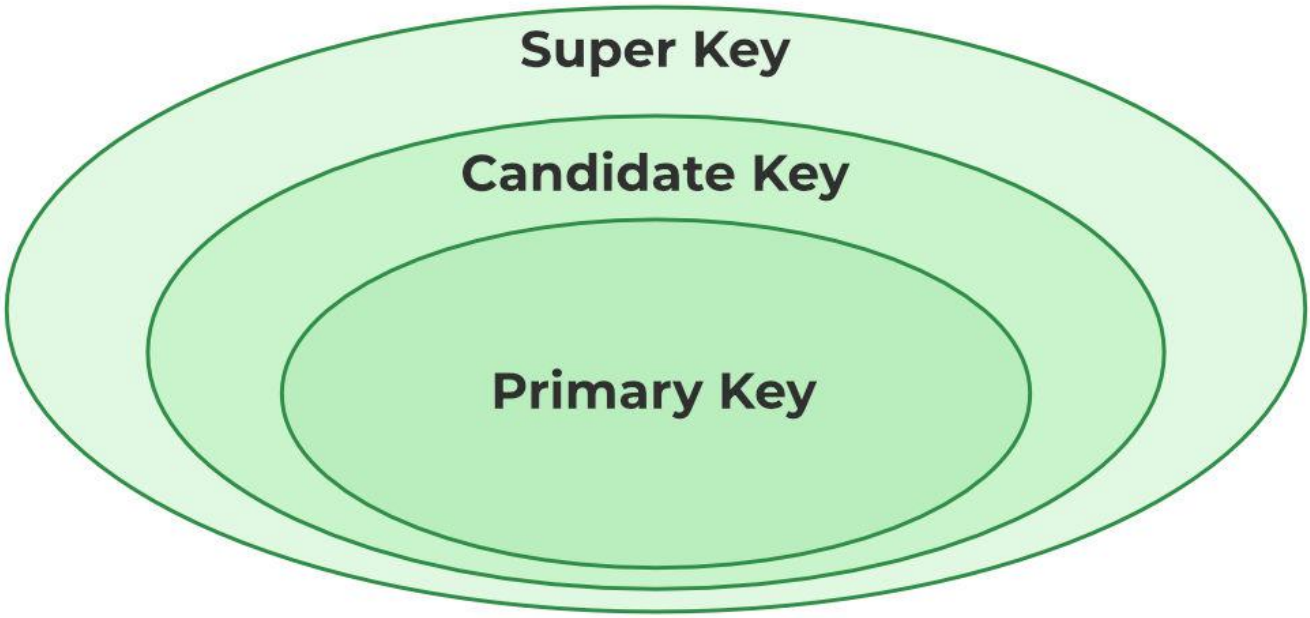
4. 슈퍼키

> 행을 유일하게 식별할 수 있는 모든 필드의 조합을 의미한다.

- 주키와 후보키, 대체 키들의 조합이 슈퍼키가 된다.
- NULL 허용 O

5. 외래키

> 다른 테이블의 기본키로 사용되는 키로 두 개 이상의 테이블을 연결하는 데 사용된다.



[DB] 무결성 제약조건

무결성이란?

> 무결성이란, 데이터베이스에서 데이터의 **정확성**, **일관성**을 유지하는 것을 의미한다. 즉, 데이터에 결함이 없는 상태를 나타낸다.

무결성 제약조건

> 무결성 제약조건은, 데이터베이스의 정확성과 일관성을 보장하기 위해 설정되는 조건으로, 데이터의 저장, 삭제, 수정 등을 제한하거나 조절하는 역할을 한다.

- 도메인 무결성
- 개체 무결성
- 참조 무결성

도메인 무결성

> 각 속성들의 값은 정의된 도메인에 속한 값이어야 한다.

ex) 나이 속성에 음수가 들어가거나, 성별이라는 속성에 남, 여를 제외한 데이터는 들어갈 수 없다.

개체 무결성

> 각 릴레이션의 기본키를 구성하는 속성은 NULL값이나 중복된 값을 가질 수 없다.

=> 기본키는 항상 유일하고 비어 있을 수 없는 값이다.

ex) 학번

참조 무결성

> 외래키 값은 NULL이거나 참조하는 릴레이션의 기본키 값과 동일해야 한다.

=> 각 릴레이션은 참조할 수 없는 외래키 값을 가질 수 없다.

[DB] 사용했던 데이터베이스에 대해서 설명해주세요.

MySQL

> MySQL은 오픈 소스 관계형 데이터베이스 관리 시스템으로, 다양한 운영체제에서 동작하며, MySQL 서버는 MySQL 엔진과 스토리지 엔진으로 구성되어 있다.

MySQL 서버 = MySQL 엔진 + 스토리지 엔진

- MySQL 엔진은 클라이언트로부터 오는 SQL 요청 처리를 담당한다.
- 스토리지 엔진은 실제 데이터를 디스크 스토리지에 저장하거나 조회하는 부분을 담당한다.

MySQL 엔진

- SQL 인터페이스: DDL, DML, ..
- SQL 파서: SQL 문법 오류 탐지
- 쿼리 옵티마이저: 실행 계획을 최적화하여 효율적인 데이터 검색을 수행한다.
- 캐시와 버퍼 기능 제공: 자주 사용되는 데이터를 메모리에 저장하여 쿼리 성능을 향상시킨다.

InnoDB

> MySQL이 제공하는 스토리지 엔진이 다양하지만 그 중 유일하게 레코드 기반의 락(Lock)을 제공하며, 이를 통해 높은 동시성 처리가 가능한 특징이 있어 주로 쓰이는 스토리지 엔진이다.

InnoDB의 구조

- 메모리 영역: InnoDB 버퍼 풀을 활용하여 **데이터 및 인덱스를 캐싱**하여 성능을 향상시킨다.
- 디스크 영역: 데이터 파일 및 로그 파일이 **저장**되며, 데이터 변경 사항을 지속적으로 기록한다.

InnoDB의 특징

- 클러스터형 인덱스: Primary Key 순서대로 디스크에 저장되며 그렇기 때문에 Range scan이 굉장히 빠르다.
- `MVCC(Multi-Version Concurrency Control)`: 다중 트랜잭션 동시 수행 시, 일관성을 유지하기 위해 여러 버전의 데이터를 관리한다.
- 자동 데드락 감지: 트랜잭션 충돌이 발생하면 자동으로 데트락을 탐지하고 해결한다.
- 자동 장애 복구: 트랜잭션 로그를 기반으로 장애 발생 시, 자동으로 복구한다.

면접 답변

MySQL이 무엇인가요?

MySQL은 오픈 소스 관계형 데이터베이스 관리 시스템(RDBMS)으로, 데이터의 저장 및 관리, 검색을 수행하는 데이터베이스 시스템입니다. MySQL 서버는 크게 MySQL 엔진과 스토리지 엔진으로 구성됩니다.

MySQL 엔진: 클라이언트로부터 SQL 요청을 받아 처리하며, SQL 문법을 분석하고 최적의 실행 계획을 수립하여 데이터 검색을 수행합니다. 스토리지 엔진: 데이터를 디스크에 저장하고 관리하는 역할을 하며, 대표적으로 InnoDB와 MyISAM을 지원합니다. MySQL은 쿼리 최적화, 캐싱, 트랜잭션 처리 등의 기능을 제공하며, 대규모 시스템에서도 높은 성능을 유지할 수 있도록 설계되었습니다.

InnoDB가 무엇인가요?

InnoDB는 MySQL에서 기본적으로 사용되는 스토리지 엔진으로, 트랜잭션을 지원하며 높은 동시성을 제공하는 특징이 있습니다.

InnoDB의 주요 특징은 다음과 같습니다.

클러스터형 인덱스(Clustered Index): Primary Key 순서대로 데이터를 저장하며, 이를 통해 범위 검색(Range Scan) 속도가 빠릅니다. MVCC(Multi-Version Concurrency Control): 여러 트랜잭션이 동시에 수행될 때 데이터의 일관성을 유지하도록 여러 버전의 데이터를 관리합니다. 자동 데드락 감지: 트랜잭션 충돌이 발생할 경우 자동으로 데드락을 탐지하고 해결합니다. 자동 장애 복구: 트랜잭션 로그를 기반으로 장애 발생 시 자동으로 복구할 수 있습니다. 이러한 특징 덕분에 InnoDB는 안전한 데이터 저장 및 고성능 트랜잭션 처리가 필요한 시스템에서 널리 사용됩니다.