

데이터베이스

강의 노트

제 2 회차
DBMS 개요

❖ 학습목표

- DBMS가 무엇이며, 궁극적인 목적이 무엇인지 설명할 수 있다.
- DBMS의 필수 기능을 나열할 수 있다.
- ANSI/SPARC의 3 단계 DB 구조를 설명할 수 있다.
- ANSI/SPARC 구조와 스키마의 관련성을 설명할 수 있다.
- 데이터 독립성에 대해 설명할 수 있다.

❖ 학습내용

- DBMS 이해하기
- ANSI/SPARC 구조와 데이터 독립성

DBMS 이해하기

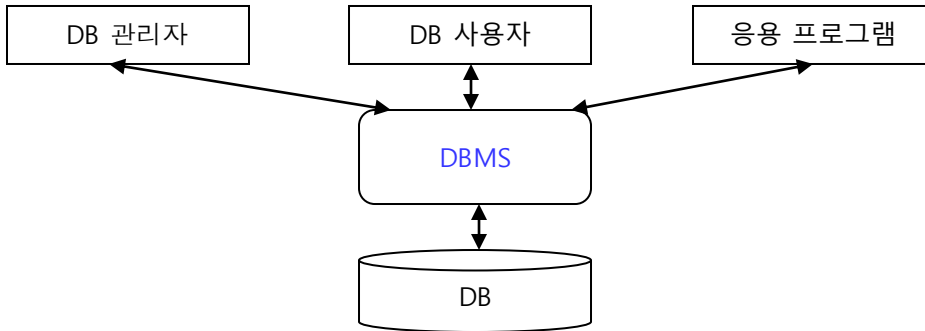
1. DBMS란?
2. DBMS의 발전 배경
3. DBMS의 필수 기능과 장단점
4. DBMS의 역사

1. DBMS란?

1) DBMS의 정의

DBMS란?

- DBMS(Database Management System, 데이터베이스 관리시스템)



DB의 정의와 조작, 제어 기능을 제공
여러 사용자와 응용 프로그램이 DB를 공유할 수 있도록 관리

2) DBMS의 목적

응용 프로그램에 영향을 주지 않고
DB의 구조를 변경할 수 있는 것

- 응용 프로그램이 데이터에 종속되지 않는 데이터 독립성(Data Independency)*을 제공하는 것
- DB의 구조를 변경해도 응용 프로그램에 영향을 주지 않고, 반대로 응용 프로그램을 변경해도 DB에 영향을 주지 않도록 하는 것

2. DBMS의 발전 배경

1) DBMS의 발전 배경

<p>사회적 요구</p>	<div data-bbox="339 330 1272 378">정보처리 시스템의 활용 분야가 점차 다양해짐</div> <div data-bbox="762 388 848 426">↓</div> <div data-bbox="339 436 1272 484">"데이터 검색"을 위주로 하는 데이터베이스 시스템에 대한 요구 등장</div> <div data-bbox="339 494 1272 542">"데이터 처리" 위주의 파일 처리 시스템으로는 구현이 어려움</div> <div data-bbox="762 552 848 591">↓</div> <div data-bbox="339 600 1272 649">DB 구축 및 관리를 지원하는 다양한 DBMS 등장</div>
<p>기술적 발전</p>	<ul style="list-style-type: none"> • 저렴한 고속 자기디스크(Magnetic Disk)의 실용화로, 대량 정보 저장 및 빠른 검색 지원이 가능함 • 데이터 통신 기술의 발전으로 빠른 정보 전송이 가능하여, 데이터의 동시 공유가 가능함

2. DBMS의 발전 배경

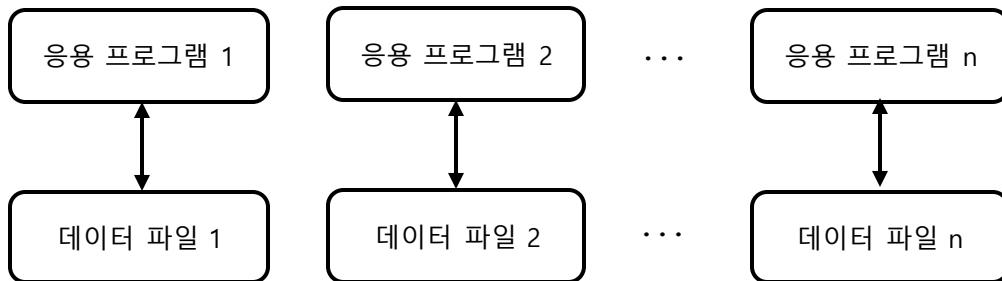
1) DBMS의 발전 배경



여기서 잠깐!

(1) 파일 처리 시스템이란?

각각의 응용 프로그램이 자신의 데이터를 별도로 파일 형태로 관리하는, 파일 중심의 데이터 처리 시스템을 의미함



[파일 처리 시스템의 응용 프로그램과 데이터 파일의 일대일 관련성]

(2) 파일 처리 시스템의 특징

응용 프로그램과 데이터 간에 상호 의존성이 있다. 즉, 대부분의 경우 응용 프로그램과 데이터 파일이 1:1로 대응함



데이터 종속성(Data Dependency)* 및 데이터 중복성(Data Redundancy)*이 유발됨

데이터 파일의 구성 방법이나 접근 방법을 변경하면, 관련된 응용 프로그램도 변경해야 하는 것

내용이 같은 데이터가 한 시스템 내에 중복해서 저장, 관리되는 것

2. DBMS의 발전 배경

1) DBMS의 발전 배경



여기서 잠깐!

(3) 파일 처리 시스템의 단점

① 일관성(Consistency) 유지의 어려움

- 데이터의 중복으로 인해 데이터의 동일성 유지가 어려움
- 중복된 데이터의 변경 시점에 따라 동일한 데이터의 값이 일치하지 않을 수 있음

② 보안(Security) 유지의 어려움

- 데이터의 중복 관리로 인해 동일한 수준의 보안 유지가 어려움

③ 경제성 저하

- 중복된 저장 및 갱신 작업으로 인해 저장 공간 및 갱신 비용이 높아짐

④ 데이터 무결성(Integrity) 유지의 어려움

- 중복 저장에 따른 관리 분산으로 데이터의 정확성 유지가 어려움

⑤ 동시 공유(Concurrent Sharing)의 어려움

- 드물게 하나의 데이터 파일을 여러 응용 프로그램이 공유한다 해도, 한 프로그램이 데이터 파일을 사용하는 동안에는 다른 응용 프로그램이 그 데이터 파일에 접근할 수 없음

3. DBMS의 필수 기능과 장단점

1) DBMS의 필수 기능

데이터 정의(Definition) 기능

- 다양한 응용 프로그램과 DB가 서로 인터페이스 할 수 있는 수단을 제공함
- 하나의 저장된 DB를 기초로 여러 사용자와 응용 프로그램의 다양한 데이터 요구를 지원할 수 있도록 DB 구조를 정의하는 기능을 제공함

데이터 조작(Manipulation) 기능

- 사용자와 DB 간의 인터페이스를 위한 수단을 제공함
- DB에 저장된 데이터의 검색, 삽입, 삭제, 갱신 등과 같은 DB 연산을 처리하는 기능을 제공함

데이터 제어(Control) 기능

- 공용으로 관리되는 DB의 내용을 정확하고 안전하게 유지할 수 있도록, 다음과 같은 3가지 제어 기능을 제공함
 - 데이터의 삽입, 삭제 등 DB 변경 시에 데이터의 무결성 및 일관성 유지 기능
 - 권한이 부여된 사용자만이 허용된 데이터에 접근할 수 있도록 접근 권한 검사 기능
 - 여러 사용자가 DB에 동시에 접근할 수 있도록 동시성(Concurrency) 제어 기능

2) DBMS의 장단점

장점	단점
① 데이터의 동시 공유가 가능함 ② 데이터 중복이 최소화됨 ③ 데이터의 무결성 유지가 용이함 ④ 데이터의 일관성 유지가 용이함 ⑤ 프로그램과 데이터 간의 독립성을 유지할 수 있음 ⑥ 데이터의 보안이 보장됨 ⑦ 데이터의 표준화 달성이 가능함	① 운영비가 증가함 ② 데이터 처리가 복잡함 ③ 백업(Backup)과 복구(Recovery)가 복잡함 ④ 시스템이 장애에 취약함

4. DBMS의 역사

1) DBMS의 역사

제 1 세대

- 60년대 초반부터 70년대 중반까지 **네트워크 데이터 모델**과 **계층 데이터 모델**을 기반으로 한 DBMS

① 1960년대 초 : IDS(Integrated Data Store)

- 최초의 범용 DBMS
- GE(General Electric)사의 찰스 바흐만(Charles Bachman)이 설계함
- 네트워크 데이터 모델(Network Data Model)의 시초

② 1960년대 후반 : IMS(Information Management System) DBMS

- IBM이 개발한 것으로, 현재도 사용되고 있음
- 계층 데이터 모델(Hierarchical Data Model)의 시초

③ 1970년대 초

- 많은 대형 컴퓨터 회사들이 네트워크 데이터 모델과 계층 데이터 모델을 기반으로 DBMS를 자체 제작하여 판매하기 시작함

제 2 세대

- 70년대 후반에 등장해서 80년대 주류가 된, **관계 데이터 모델(Relational Data Model)**을 기반으로 한 DBMS

① 관계 데이터 모델(Relational Data Model)

- IBM 산호세 연구소의 코드(E. F. Codd) 박사가 제안한 모델
- 관계 DB(Relational Database) 이론의 기초가 됨
- 1980년대 DBMS의 주류가 되어 계속 확장됨

② SQL(Structure Query Language)

- IBM이 관계 DBMS의 일부로 개발한 DB 언어
- 세계 표준 데이터베이스 언어가 됨

③ PC 기반의 DBMS 등장

- Access, FoxPro, dBase, Paradox, SQL Server 등

④ 주요 상용 DBMS 등장

- DB2, Oracle, Ingres, Sybase, Informix 등

4. DBMS의 역사

1) DBMS의 역사

제 3 세대

➤ 80년대 후반부터 새롭게 출현한 **객체 DBMS**나 **객체-관계 DBMS**

① 사용자의 DB 응용에 대한 복잡성(complexity)이 증대됨

- 이미지 및 동영상 처리와 관리, 공간 및 시간 계열(Spatial & Time Series) 데이터 처리, 데이터 마이닝(Data Mining) 등

② 사용자의 요구에 대처하기 위해 새로운 데이터 모델을 기반으로 하는 DBMS 출현

- 객체 DBMS(ODBMS: Object DBMS) : O2, ObjectStore, GemStone 등

- 객체-관계 DBMS(ORDBMS: Object-Relational DBMS)
: 오라클(Oracle) 8.X(1997년) 이후 버전, UniSQL, Illustra 등

현재는?

- 제 2 세대 DBMS와 제 3 세대 DBMS가 공존함
- 4세대 기술이 개발 중임

ANSI/SPARC 구조와 데이터 독립성

1. ANSI/SPARC 구조 이해
2. ANSI/SPARC 3 단계 구조의 구성
3. ANSI/SPARC 구조와 스키마
4. 데이터 독립성

1. ANSI/SPARC 구조 이해

1) ANSI/SPARC의 정의

ANSI/SPARC란?

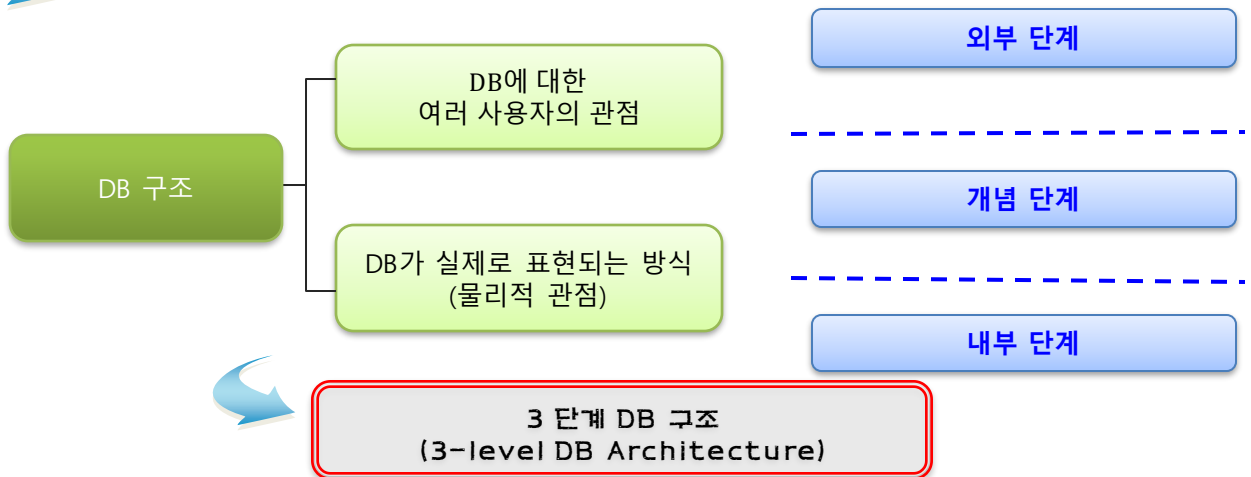
- American National Standards Institute, Standards Planning And Requirements Committee
- 미국의 컴퓨터 및 정보처리에 관한 표준화 위원회

2) ANSI/SPARC 구조의 정의

ANSI/SPARC 구조(Architecture)란?

DBMS의 구현을 위한 추상적인 설계 표준

- 1975년 ANSI/SPARC에 처음 제안됨

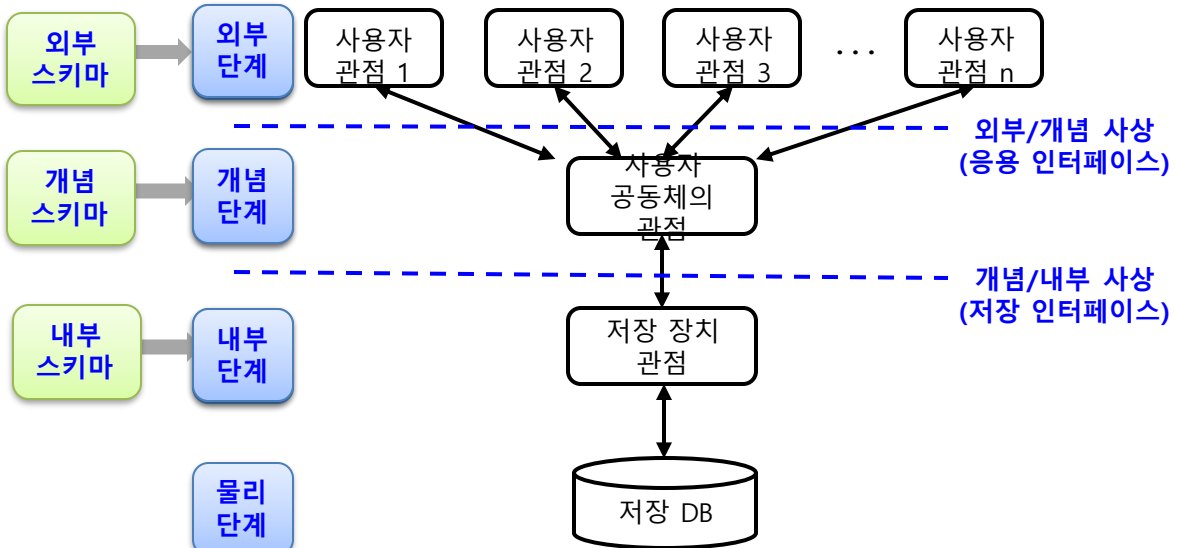


1. ANSI/SPARC 구조 이해

2) ANSI/SPARC 구조의 정의



ANSI/SPARC 3 단계 구조

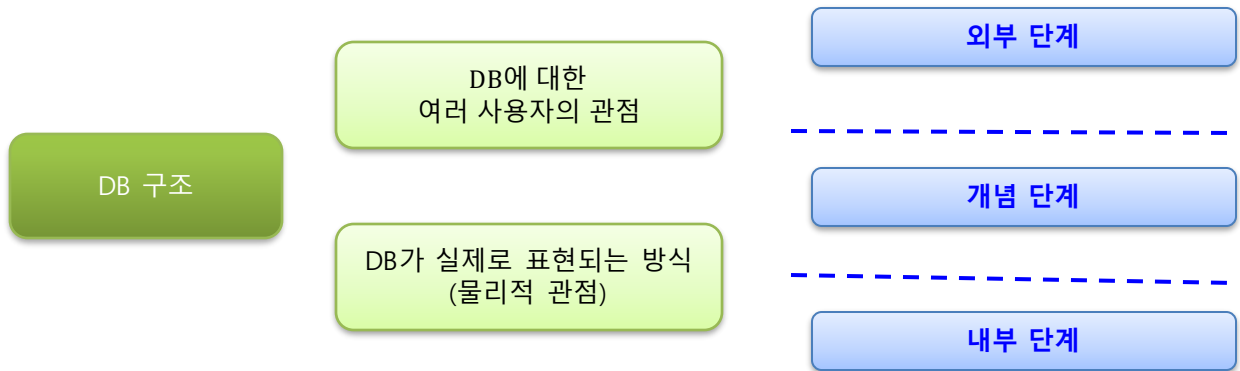


[ANSI/SPARC 3 단계 구조(Three-level Architecture)]

- 물리 단계(Physical level)는 DBMS의 지시에 따라 운영체제가 관리하므로 DB 구조에는 포함시키지 않았음
- 물리적 단계는 디스크와 같은 저장장치에 데이터를 실제로 저장하는 기법을 다루게 됨

1. ANSI/SPARC 구조 이해

3) ANSI/SPARC 구조의 목적

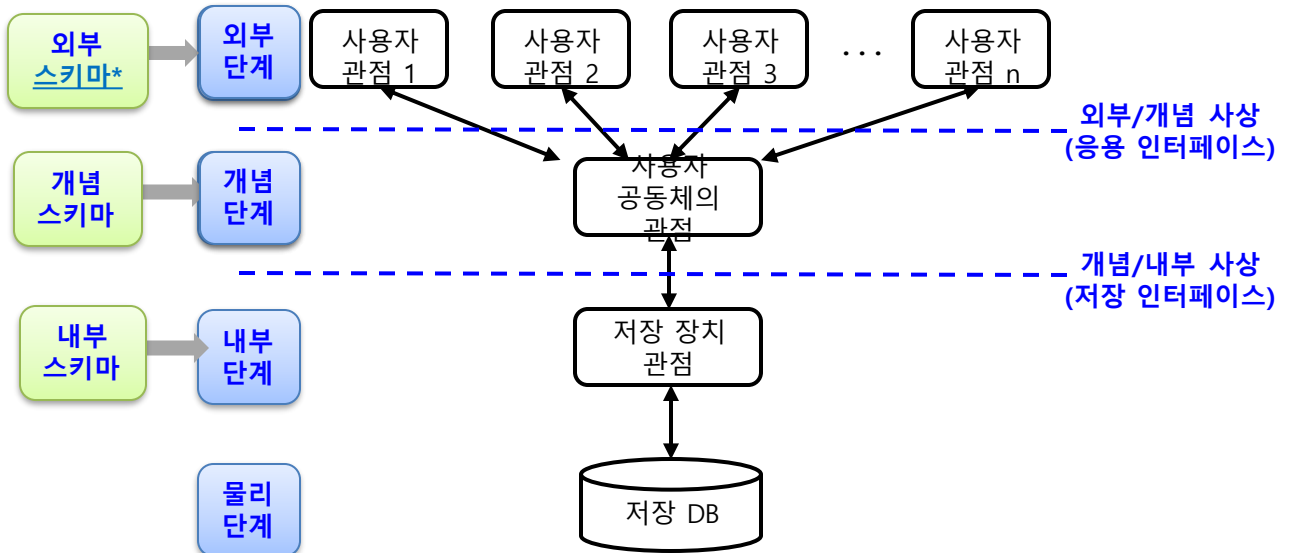


“응용 프로그램과 데이터 간의 독립성을 제공”

사용자	DB의 내부 구조에 대해 알지 못해도 DB를 사용할 수 있음
DB 관리자	응용 프로그램에 영향을 주지 않고 DB 구조를 변경할 수 있음

2. ANSI/SPARC 3 단계 구조의 구성

1) ANSI/SPARC 3 단계 구조의 구성



[ANSI/SPARC 3 단계 구조(Three-level Architecture)]

2. ANSI/SPARC 3 단계 구조의 구성

1) ANSI/SPARC 3 단계 구조의 구성

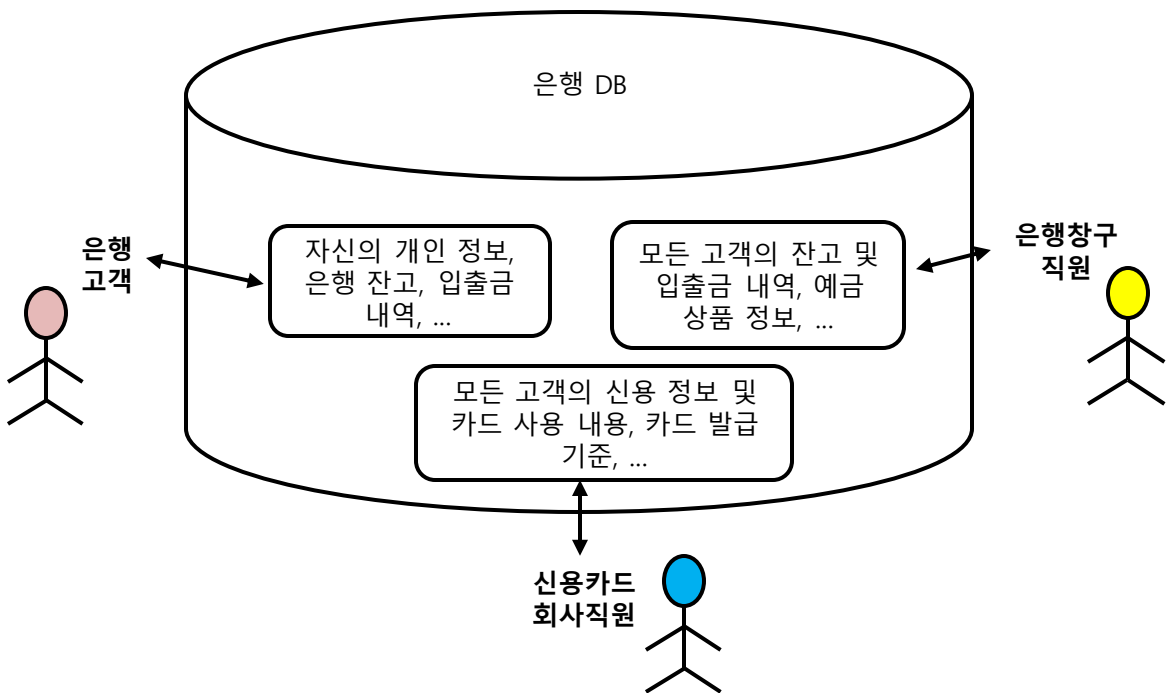


외부 단계(External level)

- DB에 관한 **개별 사용자의 관점(View)**으로서, 각 사용자나 응용 프로그래머가 생각하는 개인적 DB 구조를 의미함

예) 은행 DB에 접근하는 개인 고객이나 창구의 직원, 신용카드 회사 직원은 각자 전체 DB의 일부분에만 관심이 있으며, 각자 생각하는 DB 구조가 다를 수 있음

외부 단계에서 각 사용자가 생각하는 DB의 구조



- 외부 단계에서 다양한 개별 사용자나 응용 프로그램이 필요로 하는 데이터 구조를 정의한 것을 **외부 스키마(External Schema)**라고 하며, 외부 스키마는 **여러 개 존재**할 수 있음

2. ANSI/SPARC 3 단계 구조의 구성

1) ANSI/SPARC 3 단계 구조의 구성



개념 단계(Conceptual level)

- DB에 관한 **사용자 공동체의 관점**, 즉 한 조직 전체를 위한 DB의 논리적 구조를 의미함
- 개념 단계에서 범 기관적 입장에서 전체 DB를 정의한 것을 **개념 스키마(Conceptual Schema)**라고 하며, 개념 스키마는 **단 하나만 존재함**



내부 단계(Internal level)

- DB에 관한 **물리적 저장 장치 관점**, 즉 DB에 어떤 데이터가 어떻게 저장되는지를 표현하는 저장 구조를 의미함
- 실제로 저장된 내부 레코드의 형식, 인덱스(Index) 유무, 저장 데이터 항목의 표현 방법 등을 포함함
- 내부 단계에서 DB의 물리적 데이터 구조를 정의한 것을 **내부 스키마(Internal Schema)**라고 하며, 내부 스키마도 **단 하나만 존재함**
 - 내부 단계에서 특정 물리적 저장 장치를 직접 다루지는 않으며, 실제로 물리 단계 보다 한 단계 위에 있음
 - 물리 단계(Physical level)는 DBMS의 지시에 따라 운영체제가 관리함



외부/개념 사상(External/Conceptual Mapping)

- 외부 스키마와 개념 스키마 간의 대응 관계를 정의한 것으로, **응용 인터페이스(Application Interface)**라고도 칭함
- 개념 스키마가 변경되어도 응용 인터페이스만 수정해 주면, 외부 스키마에 아무런 영향을 미치지 않게 됨
- 즉, 응용 프로그램을 변경할 필요가 없음



개념/내부 사상 (Conceptual/Internal Mapping)

- 개념 스키마와 내부 스키마 간의 대응 관계를 정의한 것으로, **저장 인터페이스(Storage Interface)**라고도 칭함
- 부득이 내부 스키마가 변경되어도 저장 인터페이스만 수정해 주면, 개념 스키마에 아무런 영향을 미치지 않게 됨

3. ANSI/SPARC 구조와 스키마

1) ANSI/SPARC 구조와 스키마

스키마(Schema)란?

- DB의 구조(Structure) 즉, **개체와 속성, 관계를 포함하는 논리적 정의와 제약조건(Constraints)**을 기술한 것

ANSI/SPARC 3단계 구조와 스키마 구분

단계	스키마 구분	스키마 특징
외부 단계	외부 스키마 (External Schema)	<ul style="list-style-type: none"> - 사용자 개개인의 관점에서 정의한 DB 스키마 - 여러 개 존재함 - 서브스키마(Subschema)라고도 칭함
개념 단계	개념 스키마 (Conceptual Schema)	<ul style="list-style-type: none"> - 범 기관적인 관점에서 정의한 DB 스키마 - 모든 외부 스키마가 요구하는 전체적인 DB 구조와 제약조건을 포함함 - 단 하나만 존재함 - 그냥 스키마(Schema)라고도 칭함
내부 단계	내부 스키마 (Internal Schema)	<ul style="list-style-type: none"> - 저장 장치 관점에서 정의한 DB 스키마 - 개념 스키마에 대한 저장구조를 정의한 것 - 단 하나만 존재함

3. ANSI/SPARC 구조와 스키마

1) ANSI/SPARC 구조와 스키마



스키마 작성 사례

학생처에서 본 학생 개체의 외부 스키마 1

STD	
학번	Integer(8)
이름	Char(10)
학과	Char(12)
지도교수	Integer(8)

교무처에서 본 학생 개체의 외부 스키마 2

STUDENT	
학번	INT
성명	CHAR(10)
소속	CHAR(12)
학점	FLOAT

학생 개체의 개념 스키마

학생	
학번	NUMBER(8)
이름	VARCHAR(20)
학과	VARCHAR(12)
학년	VARCHAR(1)
지도교수	NUMBER(8)
주소	VARCHAR(30)
성적	NUMBER(5)

학생 개체의 내부 스키마

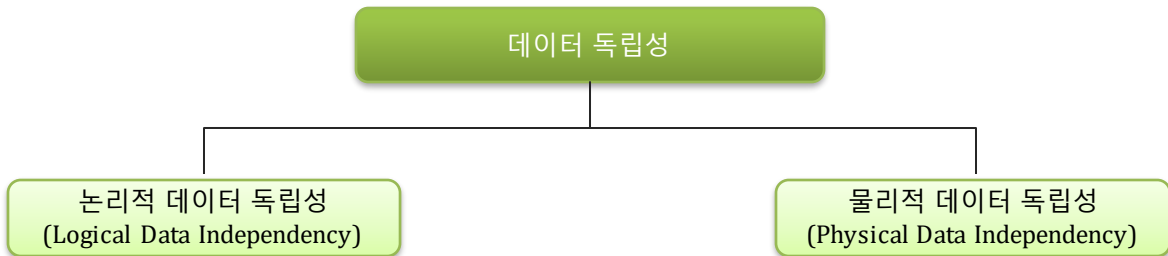
STORED_STUDENT			LENGTH = 88
prefix	BYTE(4)	OFFSET = 0	INDEX = STDINDX
ID	BYTE(8)	OFFSET = 4	
Name	BYTE(20)	OFFSET = 12	
Dept	BYTE(12)	OFFSET = 32	
Year	BYTE(1)	OFFSET = 44	
Advisor	BYTE(8)	OFFSET = 45	
Address	BYTE(30)	OFFSET = 53	
Grade	BYTE(5)	OFFSET = 83	

4. 데이터 독립성

1) 데이터 독립성

데이터 독립성(Data Independence)이란?

- 응용 프로그램과 데이터가 서로 종속되지 않는 것
 - 서로 영향을 미치지 않도록 하는 것
- DB의 논리적 구조나 물리적 구조가 변경되어도 응용 프로그램에 영향을 주지 않는 것



2) 데이터 독립성의 분류

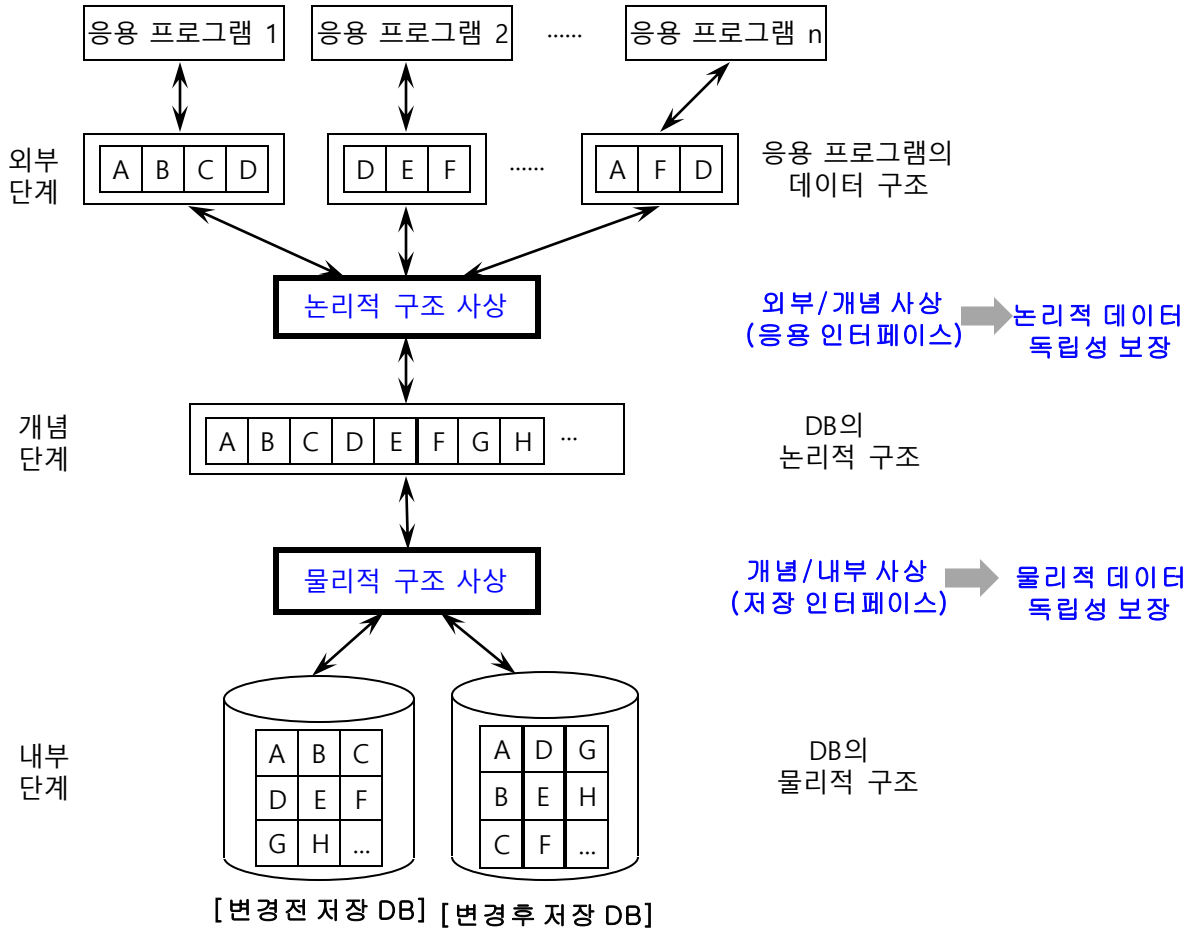
구분	논리적 데이터 독립성 (Logical Data Independence)	물리적 데이터 독립성 (Physical Data Independence)
정의	<p>응용 프로그램에 영향을 주지 않고 DB의 논리적 구조(개념 스키마)를 변경할 수 있는 것</p> <p>☞ DB에 새로운 데이터 항목이나 레코드를 추가해도, 현재 정의된 사용자 관점이나 사용되고 있는 응용 프로그램 가운데 직접 관련되지 않는 사용자 관점과 응용 프로그램은 영향을 전혀 받지 않는 것</p>	<p>응용 프로그램이나 DB의 논리적 구조에 영향을 주지 않고 DB의 물리적 구조(파일 편성, 데이터 접근 방법 등)를 변경할 수 있는 것</p> <p>☞ 새로운 저장 장치나 접근 방법의 개발로 인해 성능을 개선하기 위해 DB의 물리적 구조를 변경하게 되더라도, DB의 논리적 구조나 DB를 사용하는 응용 프로그램이 영향을 받지 않는 것</p>
독립성 보장	<p>외부/개념 사상에 의해서 논리적 데이터 독립성이 보장됨</p> <p>☞ DBMS가 DB에 대한 하나의 논리적 데이터 구조를 기초로 여러 응용 프로그램이 요구하는 다양한 형태의 데이터 구조로 대응시킬 수 있기 때문</p>	<p>개념/내부 사상에 의해서 물리적 데이터 독립성이 보장됨</p> <p>☞ DBMS가 DB에 대한 하나의 논리적 데이터 구조를 상이한 여러 가지 물리적 구조와 대응시킬 수 있기 때문</p>

4. 데이터 독립성

2) 데이터 독립성의 분류



ANSI/SPARC 각 단계 간의 사상(Mapping)



[ANSI/SPARC 각 단계 간의 사상(Mapping)]