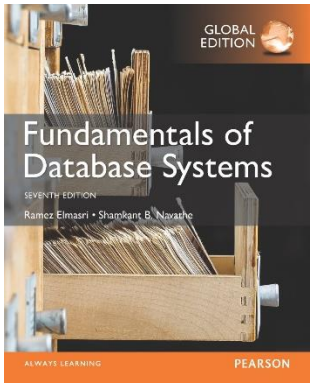
데이터베이스 시스템

충북대학교 컴퓨터과학과

이종윤

제2장:
데이터베이스 시스템 개념 및 아키텍처



소개

개요

- 데이터 모델 및 해당 범주
- 데이터 모델의 역사
- 스키마, 인스턴스 및 상태
- 3개 스키마 아키텍처
- 데이터 독립성
- DBMS 언어 및 인터페이스
- 데이터베이스 시스템 유틸리티 및 도구
- 중앙 집중형 및 클라이언트-서버 아키텍처
- DBMS의 분류

데이터 모델, 스키마 및 인스턴스

데이터 모델

- **데이터 모델:**
  - 개념의 집합 설명하다 ***구조*** 데이터베이스의 ***운영*** 이러한 구조를 조작하고 특정 ***제약 조건*** 데이터베이스가 따라야 하는 것.

- **데이터 모델 구조 및 제약:**
  - 구조는 데이터베이스 구조를 정의하는 데 사용됩니다.
  - 일반적으로 구성에는 다음이 포함됩니다. ***강요***(그리고 그들의 ***데이터 유형***) 및 요소 그룹(예: ***엔터티, 레코드, 테이블***), 그리고 ***관계*** 그러한 그룹들 사이에서

  - 제약조건은 유효한 데이터에 대한 몇 가지 제한을 지정합니다. ; 이러한 제약은 항상 적용되어야 합니다.

데이터 모델(계속)

- **데이터 모델 작업:**
  - 이러한 작업은 데이터베이스를 지정하는 데 사용됩니다.*검색*그리고 *업데이트* 데이터 모델의 구성을 참조하여
  - 데이터 모델에 대한 작업에는 다음이 포함될 수 있습니다.***기본 모델 작업*** (예: 일반 삽입, 삭제, 업데이트) 및***사용자 정의 작업***(예를 들어 학생_평균_학점_계산, 재고_업데이트)

데이터 모델의 범주

- **개념적(고수준, 의미적) 데이터 모델:**
  - 개념을 제공합니다.많은 사용자가 데이터를 인식하는 방식과 유사 .
    - (엔터티 기반 또는 객체 기반 데이터 모델이라고도 합니다.)

- **물리적(저수준, 내부) 데이터 모델:**
  - 설명하는 개념을 제공합니다.데이터가 컴퓨터에 저장되는 방법에 대한 세부 정보 .
    이는 일반적으로 DBMS 설계 및 관리 매뉴얼을 통해 임시 방식으로 지정됩니다.

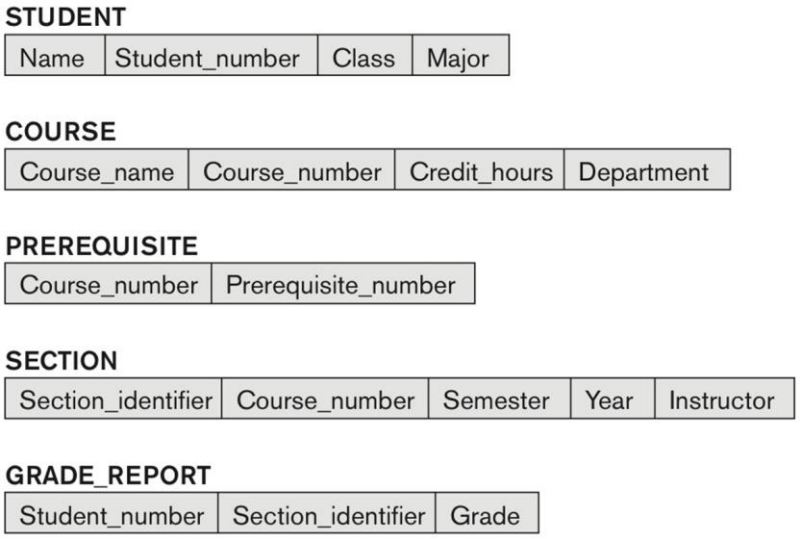
- **구현(표현) 데이터 모델:**
  - 개념을 제공합니다위의 두 가지 사이에 속하며 많은 상업용 DBMS 구현에서 사용됩니다. (예: 많은 상업 시스템에서 사용되는 관계형 데이터 모델).

- **자기 설명적 데이터 모델:**
  - 데이터 설명과 데이터 값을 결합합니다. . 예로는 XML, 키-값 저장소 및 일부 NOSQL 시스템이 있습니다.

스키마 대 인스턴스

- **데이터베이스 스키마:**
  - 그 ***설명*** 데이터베이스의.
  - 포함 사항 데이터베이스 구조, 데이터 유형 및 데이터베이스의 제약 조건에 대한 설명 .

- **스키마 다이어그램:**
  - 안 ***설명적인*** 데이터베이스 스키마의 (대부분 측면의) 표시.



데이터베이스의 스키마 다이어그램

STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

학생 및 과정 정보를 저장하는 데이터베이스

스키마 대 인스턴스

- **스키마 구성:**
  - 아***요소***스키마 또는 스키마 내의 개체, 예: 학생, 과정.

- **데이터베이스 상태:**
  - 데이터베이스에 저장된 실제 데이터에이***특정한 순간***여기에는 데이터베이스에 있는 모든 데이터를 수집하는 것이 포함됩니다.
  - 또한 불린다데이터베이스 인스턴스(또는 발생 또는 스냅샷) .
    - 용어*사례*개별 데이터베이스 구성 요소에도 적용됩니다. 예:
      레코드 인스턴스, 테이블 인스턴스, 엔티티 인스턴스

데이터베이스 스키마 대 데이터베이스 상태

- **데이터베이스 상태:**
  - 다음을 참조합니다.***콘텐츠***데이터베이스의시간의 한 순간에 .
- **초기 데이터베이스 상태:**
  - 시스템에 처음 로드될 때의 데이터베이스 상태를 나타냅니다.
- **유효한 상태:**
  - 다음 사항을 만족하는 상태구조그리고제약 조건데이터베이스의.

데이터베이스 스키마 대 데이터베이스 상태
(계속되는)

- **구별**
  - 그 ***데이터베이스 스키마***변화매우 드물다 와이.
  - 그 ***데이터베이스 상태***변화데이터베이스가 업데이트될 때마다 .

- **스키마는 내포라고도 불린다.**
- **상태는 확장이라고도 불린다.**

데이터베이스 스키마의 예

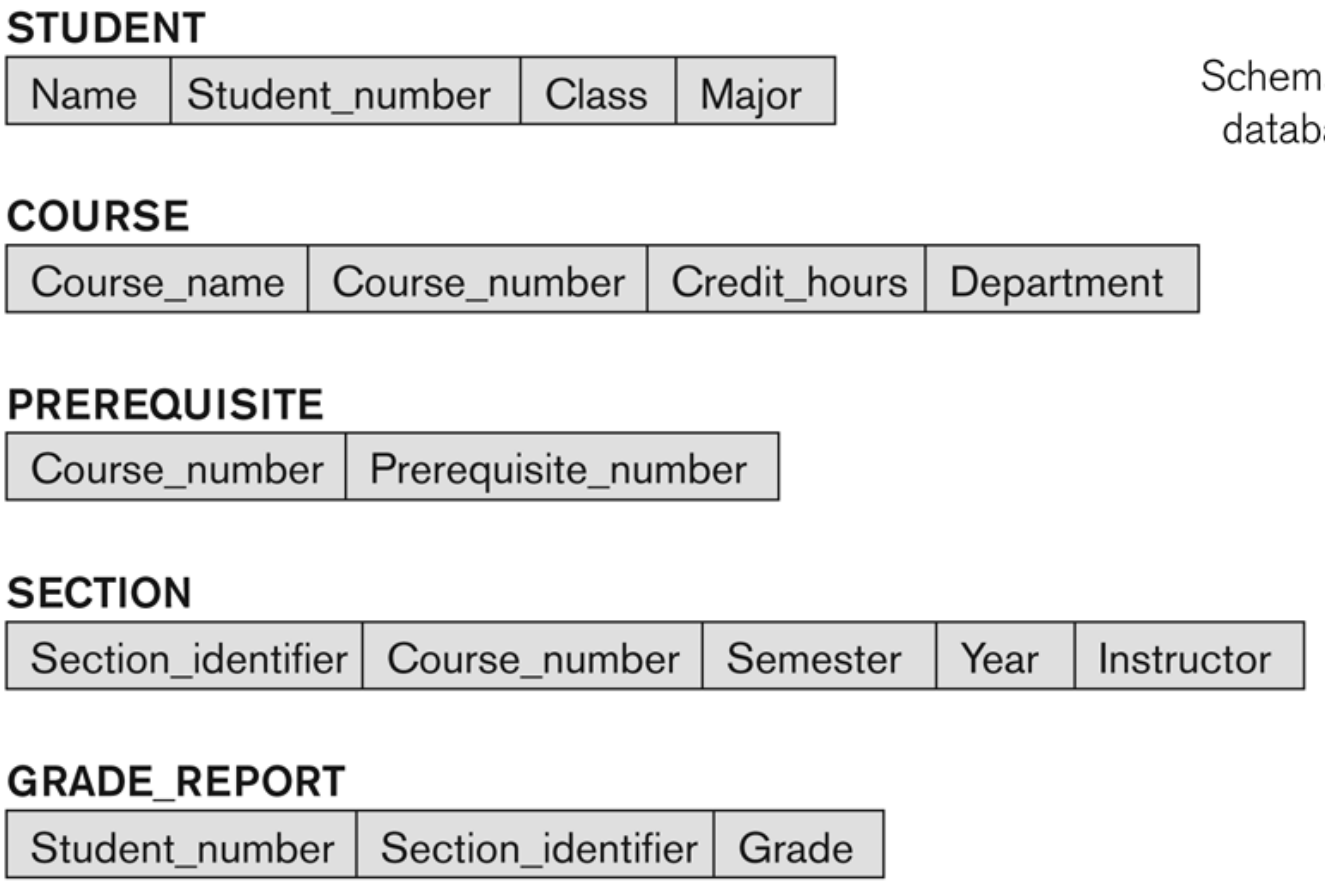


Figure 2.1
Schema diagram for the database in Figure 1.2.

데이터베이스 상태의 예

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	04	King
92	CS1310	Fall	04	Anderson
102	CS3320	Spring	05	Knuth
112	MATH2410	Fall	05	Chang
119	CS1310	Fall	05	Anderson
135	CS3380	Fall	05	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Figure 1.2
A database that stores student and course information.

3개 스키마 아키텍처 및 데이터 독립성

3개 스키마 아키텍처

- **다음의 DBMS 특성을 지원하도록 제안됨:**
  - **프로그램-데이터 독립성.**
  - 지원**다양한 뷰**데이터의

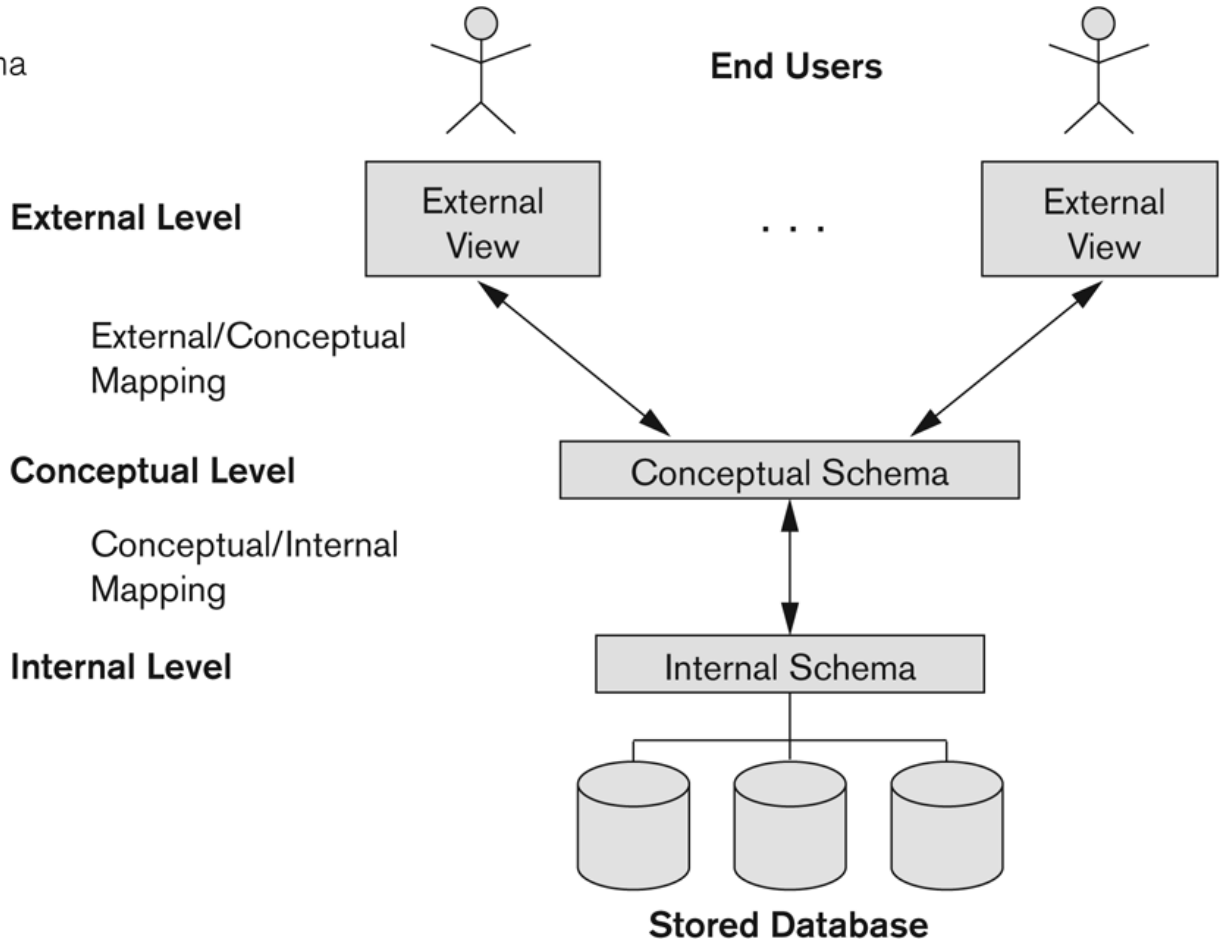
- **상용 DBMS 제품에서는 명시적으로 사용되지 않지만 데이터베이스 시스템 구성을 설명하는 데 유용합니다.**

3개 스키마 아키텍처

- **DBMS 스키마를 정의합니다.*삼*수준:**
  - **내부 스키마**물리적 저장 구조와 액세스 경로(예: 인덱스)를 설명하기 위해 내부 수준에서 사용됩니다.
    - 일반적으로 다음을 사용합니다.**물리적**데이터 모델.
  - **개념적 스키마**개념 수준에서 구조를 설명하다
    그리고 사용자 커뮤니티를 위한 전체 데이터베이스에 대한 제약이 있습니다.
    - 사용**개념적**또는**구현**데이터 모델.
  - **외부 스키마**외부 수준에서 다양한 사용자를 설명합니다.
    조회수.
    - 일반적으로 개념 스키마와 동일한 데이터 모델을 사용합니다.

3개 스키마 아키텍처

Figure 2.2
The three-schema architecture.



3개 스키마 아키텍처

- **스키마 수준 간 매핑은 다음과 같습니다.요청 및 데이터를 변환하는 데 필요함 .**
  - 프로그램은 외부 스키마를 참조하며 DBMS에 의해 매핑됩니다.

    실행을 위한 내부 스키마.
  - 내부 DBMS 수준에서 추출된 데이터는 일치하도록 다시 포맷됩니다.

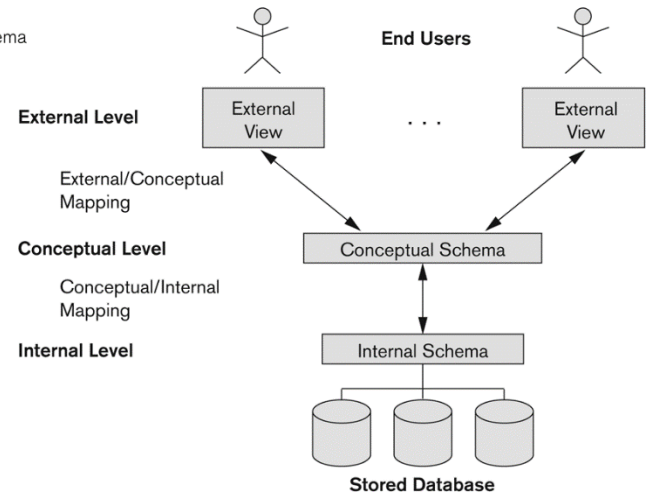
    사용자의 외부 뷰(예: 웹 페이지에 표시하기 위한 SQL 쿼리 결과 포맷팅)

데이터 독립성

- **논리적 데이터 독립성:**
  - 용량외부 스키마를 변경하지 않고도 개념 스키마를 변경합니다. 및 연관된 응용프로그램.

- **물리적 데이터 독립성:**
  - 용량개념 스키마를 변경하지 않고도 내부 스키마를 변경합니다. .
  - 예를 들어, 특정 파일 구조가 재구성되거나 데이터베이스 성능을 개선하기 위해 새로운 인덱스가 생성될 때 내부 스키마가 변경될 수 있습니다.

Figure 2.2
The three-schema architecture.



데이터 독립성(계속)

- **하위 수준의 스키마가 변경되면오직 중핑핑 이 스키마와 상위 수준 스키마 사이바뀌어야 해요 NDBMS 데이터 독립성을 완벽하게 지원합니다.**

- **상위 수준 스키마 자체는 변경되지 않습니다.**
  - 따라서 외부 스키마를 참조하기 때문에 애플리케이션 프로그램을 변경할 필요가 없습니다.

데이터베이스 언어 및 인터페이스

DBMS 언어

- **데이터정의언어(DDL)**

- **데이터시장 조작언어(DML)**

  – 고급 또는비절차적언어: 여기에는 관계형이 포함됩니다.

    언어 SQL: (대화형 쿼리 언어)

    • 아마도독립형으로 사용됨 또는 될 수도 있습니다프로그래밍에 내장됨 언어

  – 낮은 수준 또는절차적언어: (임베디드 언어)

    • 이것들은 프로그래밍 언어에 포함되어야 합니다.

DBMS 언어

- **데이터 정의 언어(DDL):**
  - DBA 및 데이터베이스 설계자가 사용데이터베이스의 개념적 스키마를 지정합니다.
  - 많은 DBMS에서 DDL은 내부 및 외부를 정의하는 데에도 사용됩니다.
    스키마(뷰).
  - 일부 DBMS에서는 별도로**저장 정의 언어(SDL)**그리고
    뷰 정의 언어(VDL)내부 및 외부 스키마를 정의하는 데 사용됩니다.

    - SDL은 일반적으로 DBA에게 제공되는 DBMS 명령을 통해 구현됩니다.
      데이터베이스 설계자

DBMS 언어

- **데이터 조작 언어(DML):**
  - 지정하는데 사용됨데이터베이스 검색 및 업데이트
  - DML 명령(데이터 하위 언어)이 가능합니다. *내장된*일반적으로-
    목적 프로그래밍 언어(호스트 언어), 예: COBOL, C,
    C++나 자바.
    - DBMS에 액세스하기 위해 함수 라이브러리도 제공될 수 있습니다.
      프로그래밍 언어
  - 또는 독립 실행형 DML 명령을 직접 적용할 수 있습니다.
    (라고 불린다*쿼리 언어*).

DML의 종류

- **높은 수준또는비절차적언어:**
  - 예를 들어, SQL 관계 언어
  - 이다"세트" 중심이며 검색 방법이 아닌 어떤 데이터를 검색할 것인지 지정합니다.

    그것을 검색합니다 .
  - 또한 ~라고도 불림**선언적**언어.

- **낮은 수준또는절차적언어:**

  반복자
  - 한 번에 한 레코드씩 데이터 검색 ;
  - 여러 레코드를 검색하려면 루핑과 같은 구성이 필요합니다.

    위치 포인터와 함께.

DBMS 인터페이스

- **독립형 쿼리 언어 인터페이스**
  - 예: DBMS 대화형 SQL 인터페이스에서 SQL 쿼리 입력(예: ORACLE의 SQL*Plus)
- **프로그래밍 언어에 DML을 내장하기 위한 프로그래머 인터페이스**
- **사용자 친화적인 인터페이스**
  - 메뉴 기반, 양식 기반, 그래픽 기반 등.
- **모바일 인터페이스: 사용자가 모바일 앱을 사용하여 거래를 수행할 수 있도록 하는 인터페이스**

DBMS 프로그래밍 언어 인터페이스

- 임베디드 접근 방식: 예: 임베디드 SQL(C, C++ 등), SQLJ(Java)

- 프로시저 호출 접근 방식: 예:제이디비씨자바의 경우, 오에스디씨(ODBC)(오픈 데이터베이스 연결) ot용 시간API(애플리케이션 프로그래밍 인터페이스)와 같은 프로그래밍 언어

Multi-row query

JDBC

```
PreparedStatement stmt = conn.prepareStatement(
    "SELECT LASTNAME"
  + " , FIRSTNME"
  + " , SALARY"
  + " FROM DSN8710.EMP"
  + " WHERE SALARY BETWEEN ? AND ?");
stmt.setBigDecimal(1, min);
stmt.setBigDecimal(2, max);
ResultSet rs = stmt.executeQuery();
while (rs.next()) {
  lastname = rs.getString(1);
  firstname = rs.getString(2);
  salary = rs.getBigDecimal(3);
  // Print row...
}
rs.close();
stmt.close();
```

SQLJ

```
#sql private static iterator EmployeeIterator(String, String,
BigDecimal);
...
EmployeeIterator iter;
#sql [ctx] iter = {
  SELECT LASTNAME
       , FIRSTNME
       , SALARY
    FROM DSN8710.EMP
   WHERE SALARY BETWEEN :min AND :max
};
do {
  #sql {
    FETCH :iter
     INTO :lastname, :firstname, :salary
  };
  // Print row...
} while (!iter.endFetch());
iter.close();
```

DBMS 프로그래밍 언어 인터페이스

- 데이터베이스 프로그래밍 언어 접근 방식: 예를 들어 ORACLE에는 PL/SQL, SQL 기반 프로그래밍 언어. 언어는 SQL과 해당 데이터 유형을 필수 구성 요소로 통합합니다.

- 스크립팅 언어:페소(PHP)(클라이언트 측 스크립팅) 및파이썬 (서버 측 스크립팅)은 데이터베이스 프로그램을 작성하는 데 사용됩니다.

```
<<label>>   -- this is optional
DECLARE
-- this section is optional
  number1 NUMBER(2);
  number2 number1%TYPE := 17;             -- value default
  text1   VARCHAR2(12) := 'Hello world';
  text2   DATE         := SYSDATE;        -- current date and time
BEGIN
-- this section is mandatory, must contain at least one executable statement
  SELECT street_number
    INTO number1
    FROM address
    WHERE name = 'INU';
EXCEPTION
-- this section is optional
   WHEN OTHERS THEN
     DBMS_OUTPUT.PUT_LINE('Error Code is ' || TO_CHAR(sqlcode));
     DBMS_OUTPUT.PUT_LINE('Error Message is ' || sqlerrm);
END;
```

사용자 친화적인 DBMS 인터페이스

- **메뉴 기반(웹 기반), 웹 검색에 인기 있음**
- **형식 기반, naï를 위해 설계됨 많은 사용자가 항목을 채우는 데 익숙합니다. 양식에**
- **그래픽 기반**
  - 가리키고 클릭하기, 끌어서 놓기 등
  - 스키마 다이어그램에서 쿼리 지정
- **자연어: 영어로 작성된 요청**
- **위의 조합:**
  - 예를 들어, 웹에서 광범위하게 사용되는 메뉴와 양식은 모두

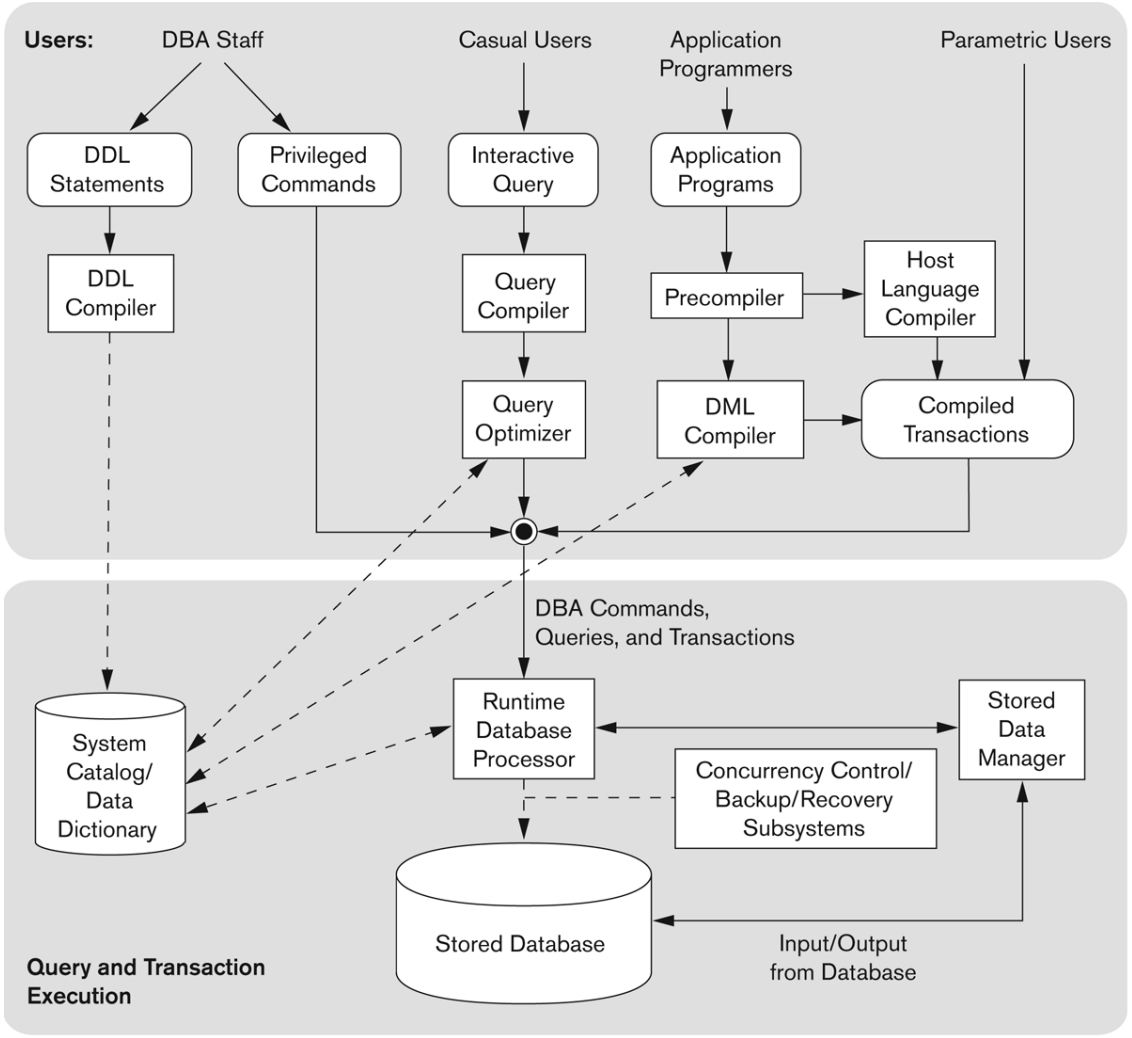
    데이터베이스 인터페이스

기타 DBMS 인터페이스

- **자연어: 쿼리로서의 자유 텍스트**
- **음성 : 입력 쿼리 및 출력 응답**
- **키워드 검색이 가능한 웹 브라우저**
- **매개 변수 인터페이스, 예: 기능 키를 사용하는 은행원.**
- **DBA를 위한 인터페이스:**
  - 사용자 계정 생성, 권한 부여
  - 시스템 매개변수 설정
  - 스키마 또는 액세스 경로 변경

데이터베이스 시스템 환경

일반적인 DBMS 구성 요소 모듈



런타임 데이터베이스 쿼리 프로세서

Figure 2.3
Component modules of a DBMS and their interactions.

데이터베이스 시스템 유틸리티

- **다음과 같은 특정 기능을 수행하려면:**
  - 데이터 로딩 중파일로 데이터베이스에 저장됨. 데이터 변환 도구 포함
  - 백업하기데이터베이스를 주기적으로 테이프에 저장합니다.
  - 데이터베이스 파일 구조를 재구성합니다.
  - 성능 모니터링 유틸리티.
  - 보고서 생성 유틸리티.
  - 정렬, 사용자 모니터링, 데이터 압축 등의 기타 기능

기타 도구

- **데이터 사전 / 저장소:**
  - 스키마 설명 및 설계 결정, 애플리케이션 프로그램 설명, 사용자 정보와 같은 기타 정보를 저장하는 데 사용됩니다.

    사용기준 등
  - **활성 데이터 사전**DBMS 소프트웨어와 사용자/DBA가 접근합니다.
  - **수동 데이터 사전**사용자/DBA만 접근할 수 있습니다.

기타 도구

- **애플리케이션 개발 환경 및 CASE(컴퓨터 지원 소프트웨어 엔지니어링) 도구:**

- **예시:**
  - 파워빌더(Sybase)
  - JBuilder(볼랜드)
  - JDeveloper 10G(오라클)

중앙집중형 및 클라이언트/서버 DBMS를 위한 아키텍처

중앙집중화 및
클라이언트-서버 DBMS 아키텍처

- **중앙집중형 DBMS:**
  - 결합하다모든 것을 단일 시스템으로 DBMS 소프트웨어, 하드웨어, 응용 프로그램 및 사용자 인터페이스 처리 포함 소프트웨어.
  - 사용자는 여전히 원격 터미널을 통해 연결할 수 있습니다. 그러나 모든 처리는 중앙 사이트에서 이루어집니다.

물리적 중앙 집중형 아키텍처

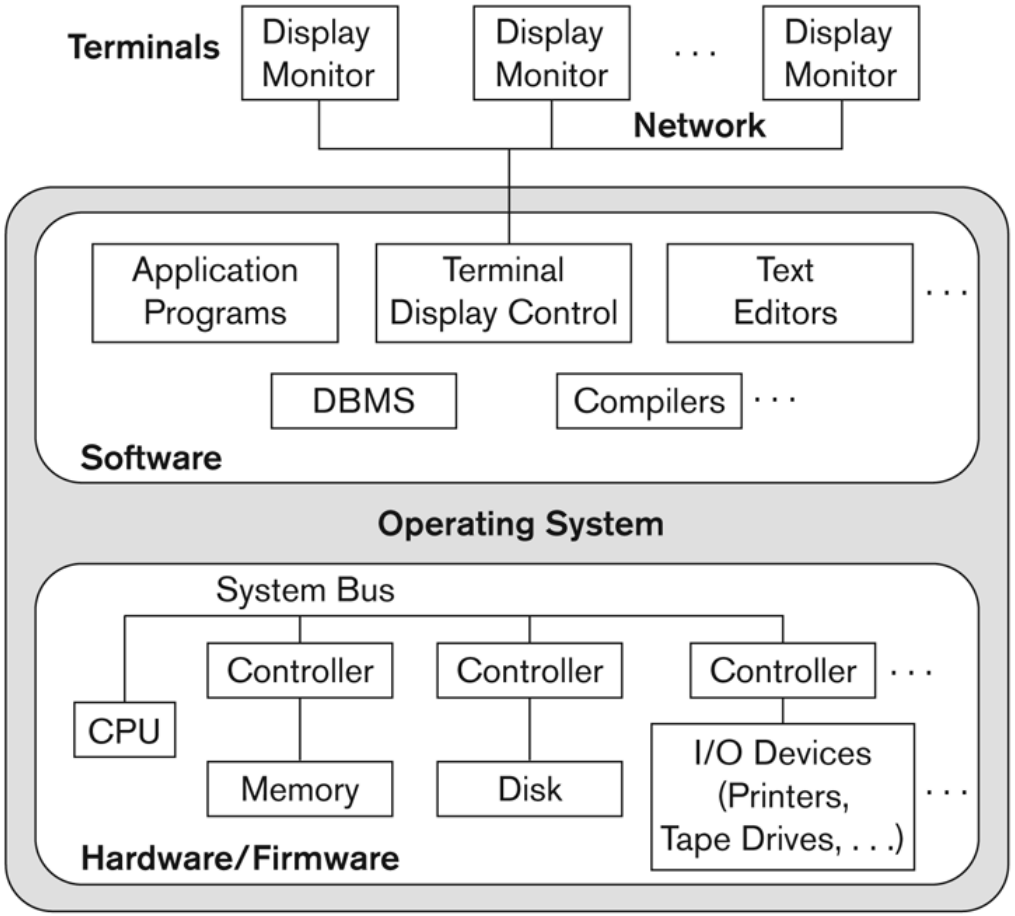


Figure 2.4
A physical centralized architecture.

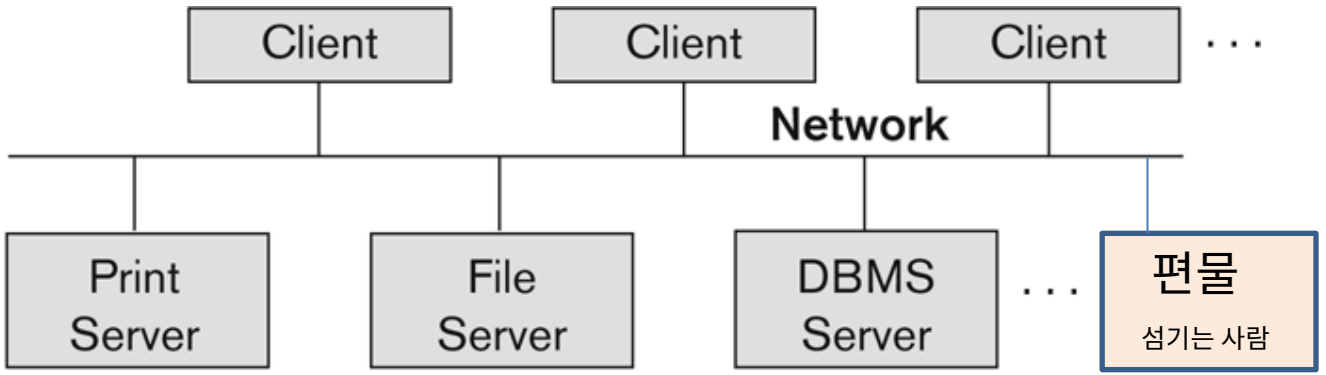
기본 2계층 클라이언트-서버 아키텍처

- **전문화된서버전문화된 기능을 갖춘**
  - 인쇄 서버
  - 파일 서버
  - DBMS 서버
  - 웹 서버
  - 이메일 서버

- **클라이언트는 필요에 따라 전문 서버에 액세스할 수 있습니다.**

논리적 2계층 클라이언트 서버 아키텍처

Figure 2.5
Logical two-tier client/server architecture.



고객

- **적절한 인터페이스를 제공하세요클라이언트 소프트웨어를 통해 다양한 서버 리소스에 접근하여 활용하기 위한 모듈입니다.**

- **클라이언트는 디스크가 없는 머신이거나 디스크가 있고 클라이언트 소프트웨어만 설치된 PC나 워크스테이션일 수 있습니다.**

- **일종의 네트워크를 통해 서버에 연결됨.**
  - (LAN: 근거리 통신망, 무선 네트워크 등)

DBMS 서버

- **클라이언트에게 데이터베이스 쿼리 및 거래 서비스를 제공합니다.**

- **관계형 DBMS 서버는 종종 SQL 서버, 쿼리 서버 또는 트랜잭션 서버라고도 합니다.**

- **클라이언트에서 실행되는 애플리케이션은 다음과 같은 표준 인터페이스를 통해 서버 데이터베이스에 액세스하기 위해 애플리케이션 프로그램 인터페이스(API)를 활용합니다.**
  - ODBC: Open Database Connectivity 표준
  - JDBC: Java 프로그래밍 액세스용

2계층 클라이언트-서버 아키텍처

- 클라이언트와 서버는 ODBC 또는 JDBC에 적합한 클라이언트 모듈과 서버 모듈 소프트웨어를 설치해야 합니다.

- 클라이언트 프로그램은 여러 DBMS에 연결할 수 있으며, 이를 데이터 소스라고도 합니다.

- 일반적으로 데이터 소스는 데이터를 관리하는 파일이거나 기타 DBMS가 아닌 소프트웨어가 될 수 있습니다.

- 데이터베이스 프로그래밍에 대한 자세한 내용은 10장을 참조하세요.

3계층 클라이언트-서버 아키텍처

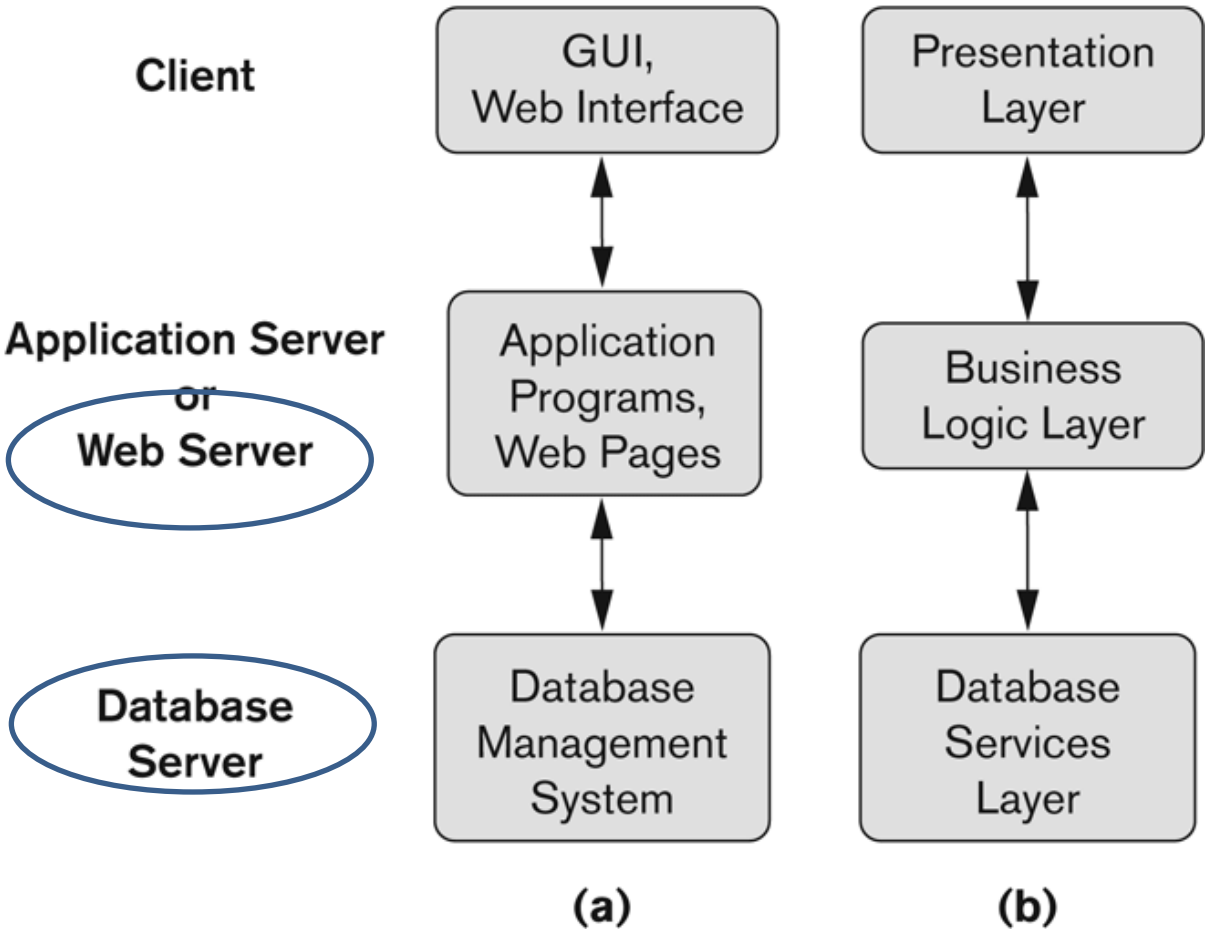
- **웹 애플리케이션에 공통됨**

- **애플리케이션 서버 또는 중간 계층이라고 함웹 서버:**
  - 데이터베이스 서버에서 해당 데이터에 액세스하는 데 사용되는 애플리케이션의 웹 연결 소프트웨어와 비즈니스 로직 부분을 저장합니다.
  - 데이터베이스 서버와 클라이언트 사이에서 부분적으로 처리된 데이터를 전송하는 매개체 역할을 합니다.

- **3계층 아키텍처는 가능합니다보안 강화:**
  - 중간 계층을 통해서만 접근 가능한 데이터베이스 서버
  - 클라이언트는 데이터베이스 서버에 직접 액세스할 수 없습니다.
  - 클라이언트에는 사용자 인터페이스와 웹 브라우저가 포함됩니다.
  - 클라이언트는 일반적으로 웹에 연결된 PC 또는 모바일 장치입니다.

3계층 클라이언트-서버 아키텍처

Figure 2.7
Logical three-tier client/server architecture, with a couple of commonly used nomenclatures.



데이터베이스의 분류
경영 시스템

DBMS의 분류

- **사용된 데이터 모델을 기반으로**
  - 레거시: 네트워크, 계층적.
  - 현재 사용 중:**관계형** ,**객체지향적** , 객체 관계형
  - 최신 기술: 키-값 저장 시스템,**NOSQL 시스템** : 문서 기반, 열 기반, 그래프 기반 및 키 값 기반. 네이티브 XML DBMS.

- **기타 분류**
  - 단일 사용자(일반적으로 개인용 컴퓨터에서 사용) 대 다중 사용자(대부분의 DBMS).
  - 중앙집중식(하나의 데이터베이스가 있는 단일 컴퓨터 사용) 대 분산(여러 컴퓨터, 여러 DB)

분산 DBMS(DDBMS)의 변형

- **동종 DDBMS**
- **이기종 DDBMS**
- **연방 또는 다중 데이터베이스 시스템**
  - 참여 데이터베이스는 느슨하게 결합되어 있습니다.높은 수준의 자치.
- **분산 데이터베이스 시스템은 이제 다음과 같은 이유로 클라이언트-서버 기반 데이터베이스 시스템으로 알려지게 되었습니다.**
  - 완전히 분산된 환경을 지원하지 않고, 클라이언트 집합을 지원하는 데이터베이스 서버 집합을 지원합니다.

DBMS에 대한 비용 고려 사항

- **비용 범위: 무료 오픈소스 시스템부터 수백만 달러의 비용이 드는 구성까지**
- **무료 관계형 DBMS의 예:MySQL,포스트그레스큐엘, 기타**
- **상용 DBMS는 시계열 모듈, 공간 데이터 모듈, 문서 모듈, XML 모듈 등 추가적인 특수 모듈을 제공합니다.**
  - 별도로 구매하면 추가적인 특수 기능을 제공합니다.
  - 때때로 카트리지(예: Oracle) 또는 블레이드라고도 함
- **다양한 라이센스 옵션: 사이트 라이센스, 최대 동시 사용자 수(좌석 라이센스), 단일 사용자 등.**

기타 고려 사항

- **데이터베이스 시스템 내의 액세스 경로 유형**
  - 예: 역색인 기반(ADABAS는 그러한 시스템 중 하나). 완전히 색인된 데이터베이스는 검색에 사용되는 모든 키워드로 액세스를 제공합니다.
    엔진)

- **일반 목적 vs. 특수 목적**
  - 예: 항공사 예약 시스템 또는 기타 여러 예약 시스템
    호텔/자동차 등을 위한 특수 목적 OLTP(온라인 거래 처리 시스템)입니다.

데이터 모델의 역사

데이터 모델의 역사(추가 자료)

- 네트워크 모델
- 계층적 모델
- 관계형 모델
- 객체 지향 데이터 모델
- 객체 관계 모델

네트워크 모델

- 최초의 네트워크 DBMS는 1964~65년에 Honeywell에서 구현했습니다(IDS 시스템).

- CODASYL(데이터 시스템 언어 회의)의 지원으로 적극적으로 채택되었습니다(CODASYL - DBTG 보고서, 1971년).

- 나중에 다양한 시스템에 구현됨 - IDMS(Cullinet) - 현재 컴퓨터 어소시에이츠), DMS 1100(유니시스), IMAGE(HP(휴렛팩커드)), VAX-DBMS(디지털 이큅먼트 코퍼레이션, 후속 COMPAQ, 현재 HP).

네트워크 모델

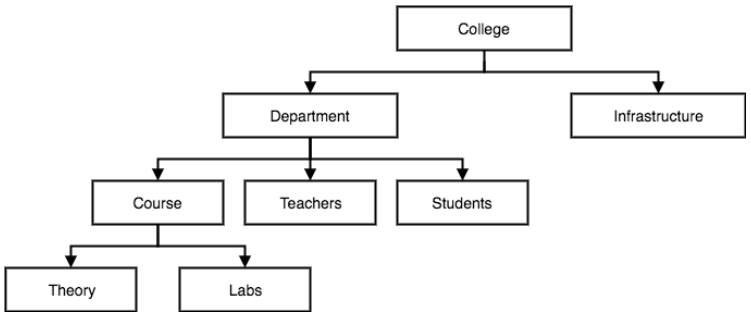
- **장점:**
  - 네트워크 모델은 복잡한 관계를 모델링하고 관계에 대한 추가/삭제 의미를 나타냅니다.
  - 레코드 유형을 사용하여 모델링하는 대부분의 상황을 처리할 수 있습니다.
    관계 유형.
  - 언어는 탐색적입니다. FIND, FIND member와 같은 구문을 사용합니다.
    FIND 소유자, 집합 내에서 FIND NEXT, GET 등.
    - 프로그래머는 데이터베이스를 최적의 방식으로 탐색할 수 있습니다.

네트워크 모델

- **단점:**
  - 처리의 탐색 및 절차적 특성
  - 데이터베이스에는 집합을 통과하는 포인터의 복잡한 배열이 포함되어 있습니다.
    기록의.
    - 자동화된 "쿼리 최적화"의 범위가 작음

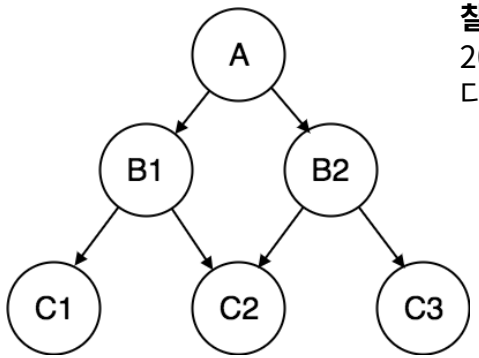
데이터베이스 기술의 역사적 발전

- **초기 데이터베이스 응용 프로그램:**
  - 그계층적 모델과 네트워크 모델은 1960년대 중반에 도입되었습니다. 70년대에 지배적인 위치를 차지했습니다.
  - 전 세계 데이터베이스 처리의 대부분은 여전히 다음을 사용하여 수행됩니다. 모델, 특히 IBM의 IMS 시스템을 사용한 계층적 모델입니다.



계층적 모델

- 상향식 구조
- 일대일을 지원하며
- 일대다 관계



네트워크 모델

- 일대일 지원
- 일대다
- 그리고 다대다 관계
- 문제는 임시 쿼리를 지원할 수 없다는 것입니다.

찰스 윌리엄 바흐만III(1924년 12월 11일 ~ 2017년 7월 13일)은 미국의 컴퓨터 과학자였다.

계층적 데이터 모델

- 처음에는 1965년경 IBM과 North American Rockwell이 공동으로 구현했습니다. 그 결과 IMS 시스템 제품군이 탄생했습니다.

- IBM의 IMS 제품은 전 세계적으로 매우 큰 고객 기반을 보유하고 있습니다(현재도 그렇습니다).

- IMS 시스템을 기반으로 계층적 모델이 정형화되었습니다.

- 이 모델을 기반으로 하는 다른 시스템: System 2k (SAS inc.)

계층적 모델

- **장점:**
  - 구성 및 작동이 간편합니다.
  - 여러 자연스럽고 계층적으로 구성된 도메인에 해당합니다.
    예를 들어, 조직도(org)
  - 언어는 간단합니다.
    - GET, GET UNIQUE, GET NEXT, GET NEXT WITHIN PARENT 등의 구문을 사용합니다.

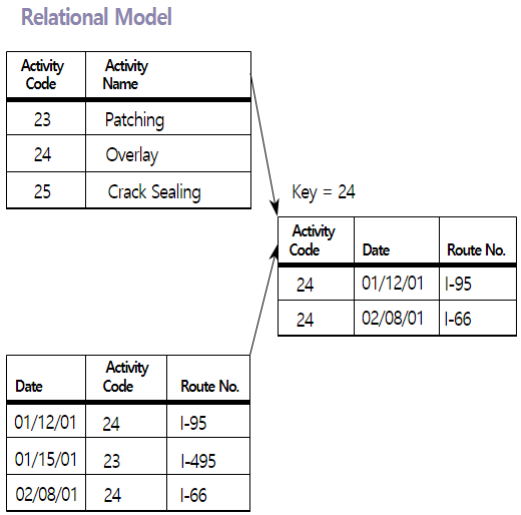
- **단점:**
  - 처리의 탐색 및 절차적 특성
  - 데이터베이스는 레코드의 선형 배열로 시각화됩니다.
  - "쿼리 최적화"에 대한 범위가 작음

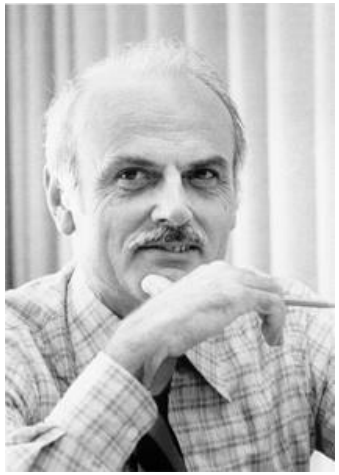
관계형 모델

- 1970년 EF Codd(IBM)가 제안하였고, 1981~82년에 최초의 상용 시스템이 출시되었다.
- 현재 여러 상용 제품(예: DB2, ORACLE, MS SQL Server, SYBASE, INFORMIX)에 사용됩니다.
- MySQL, PostgreSQL 등 다양한 무료 오픈 소스 구현
- 현재 데이터베이스 애플리케이션 개발에 가장 많이 사용됩니다.
- SQL 관계형 표준: SQL-89(SQL1), SQL-92(SQL2), SQL-99, SQL3, …
- 5장부터 11장까지는 이 모델을 자세히 설명합니다.

데이터베이스 기술의 역사적 발전

- **관계형 모델 기반 시스템:**
  - 관계형 모델은 원래 1970년에 도입되었으며 IBM Research와 여러 기관에서 심도 있게 연구되고 실험되었습니다.
    대학.
  - 관계형 DBMS 제품은 1980년대 초반에 등장했습니다.





에드거 프랭크 "테드" 코드(1923년 8월 19일 ~ 2003년 4월 18일)은 영국의 컴퓨터 과학자였습니다.IBM에서 근무하는 동안 데이터베이스 관리를 위한 관계형 모델을 발명했습니다. , 관계형 데이터베이스와 관계형 데이터베이스 관리 시스템의 이론적 기초입니다.

객체 지향 데이터 모델

- 여러 가지 모델이 데이터베이스 시스템에 구현되기 위해 제안되었습니다.
- 한 세트는 C++와 같은 지속적 OO 프로그래밍 언어의 모델을 포함합니다(예: 오브제스토어또는 VERSANT), 그리고 Smalltalk(예: GEMSTONE).
- 이 외에 O2, ORION(MCC - 당시 ITASCA), IRIS(HP - Open OODB에서 사용)와 같은 시스템이 있습니다.
- 객체 데이터베이스 표준: ODMG-93, ODMG-버전 2.0, ODMG-버전 3.0.
- 12장에서는 이 모델을 설명합니다.

객체 관계 모델

- 객체 모델과 관계형 모델을 혼합하는 추세는 Informix Universal Server에서 시작되었습니다.
- 관계형 시스템은 객체 데이터베이스의 개념을 통합하여 객체 관계형으로 이어졌습니다.
- Oracle, DB2, SQL Server 및 기타 DBMS 버전에서 예시됩니다.
- 현재 관계형 DBMS 공급업체의 추세는 XML, 텍스트 및 기타 데이터 유형을 처리할 수 있는 기능을 관계형 DBMS에 확장하는 것입니다.
- "객체 관계형"이라는 용어는 시장에서 점점 사라져가고 있습니다.

장 요약

- 데이터 모델 및 해당 범주
- 스키마, 인스턴스 및 상태
- 3개 스키마 아키텍처
- 데이터 독립성
- DBMS 언어 및 인터페이스
- 데이터베이스 시스템 유틸리티 및 도구
- 데이터베이스 시스템 환경
- 중앙 집중형 및 클라이언트-서버 아키텍처
- DBMS의 분류
- 데이터 모델의 역사