데이터베이스 소개

용어

- 데이타 (data)
  - 의미를 가지면서 기록될 수 있는 알려진 사실
- 데이타베이스 (database)
  - 관련있는 데이타의 모임
- 데이타베이스 관리 시스템 (DBMS)
  - 데이터베이스의 생성과 관리를 담당하는 소프트웨어 패키지
- 데이타베이스 시스템 (Database System)
  - Database와 그를 관리하는 소프트웨어 (DBMS, 응용 프로그램) 모두를 칭하는 용어
- 작은 세계(mini-world)
  - 데이타베이스 구축의 대상이 되는 실세계의 일부분

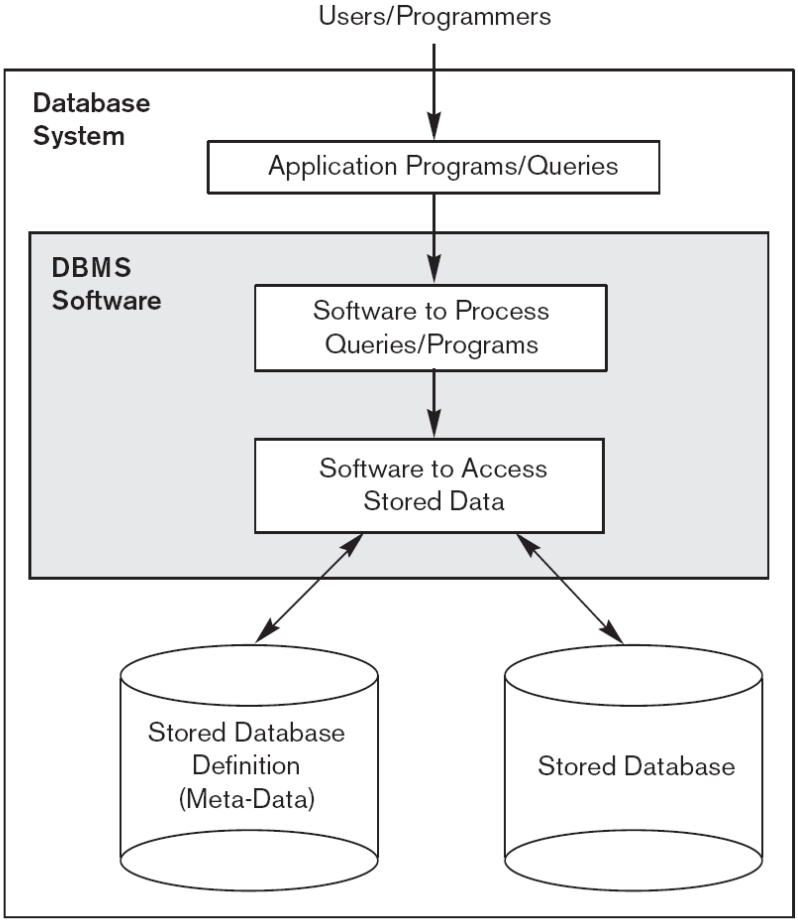

Figure 1.1
A simplified database system environment.

Example Database

대학교 정보 데이타베이스 예제

1. 엔티티

    - STUDENT
    - COURSE
    - (COURSE의) SECTION
    - DEPARTMENT
    - INSTRUCTOR

1. 엔티티 사이의 관계

    - SECTION은 특정 COURSE에 속한다.
    - STUDENT는 SECTION에 참가한다.
    - COURSE는 선수 COURSE가 있다.
    - INSTRUCTOR는 SECTION을 강의한다.
    - COURSE는 DEPARTMENT에서 제공한다.
    - STUDENT는 DEPARTMENT를 전공한다.

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

Figure 1.2
A database that stores student and course information.

Database의 특징

- 데이타베이스 시스템의 자기 기술성(self-describing)
  - Database 카탈로그(catalog)에는 메타 데이타 (meta-data)가 저장되어 있으며, 이를 이용하여 하나의 DBMS가 다수의 데이타베이스를 관리할 수 있음
  - 메타 데이타 (meta-data) : 데이타베이스에 대한 정보
- 프로그램과 데이타의 분리
  - 데이타베이스 내의 데이타 저장 구조가 변경되어도 Database 응용 프로그램은 영향을 받지 않는 (변경될 필요가 없는) 성질
  - 프로그램과 데이타의 독립성(program-data independence)을 높임
- 데이타 추상화
  - 데이타 모델(data model)을 사용함으로써 저장 구조의 자세한 내용은 사용자로부터 은닉시키고 개념적인 뷰(conceptual view)만을 제공함
- 데이타에 대한 다중 뷰(view) 제공
  - 사용자는 전체 데이타베이스 보다는 관심이 있는 데이타베이스의 일부를 뷰로 정의할 수 있음
- 데이터의 공유와 다수 사용자 트랜잭션 처리
  - 트랜잭션은 한 번 이상의 데이터베이스 접근을 포함하는 프로그램의 단위 혹은 프로세스 수행. 고립성과 원자성의 성질을 만족하여야 함
  - 동시성 제어(Concurrency Control)
  - 온라인 트랜잭션 처리(OLTP: On-Line Transaction Processing)
  - 다수 사용자를 위한 DB 프로그래밍 관점에서 트랜잭션에 대해 학습

Example Metadata

RELATIONS

Relation_name	No_of_columns
STUDENT	4
COURSE	4
SECTION	5
GRADE_REPORT	3
PREREQUISITE	2

COLUMNS

Column_name	Data_type	Belongs_to_relation
Name	Character (30)	STUDENT
Student_number	Character (4)	STUDENT
Class	Integer (1)	STUDENT
Major	Major_type	STUDENT
Course_name	Character (10)	COURSE
Course_number	XXXXNNNN	COURSE
….	….	…..
….	….	…..
….	….	…..
Prerequisite_number	XXXXNNNN	PREREQUISITE

Note: Major_type is defined as an enumerated type with all known majors.
XXXXNNNN is used to define a type with four alpha characters followed by four digits.

Figure 1.3
An example of a database catalog for the database in Figure 1.2.

Example Internal Storage Foramt

Data Item Name	Starting Position in Record	Length in Characters (bytes)
Name	1	30
Student_number	31	4
Class	35	1
Major	36	4

Figure 1.4
Internal storage format for a STUDENT record, based on the database catalog in Figure 1.3.

Example Database View

(a)

TRANSCRIPT

Student_name	Student_transcript				
	Course_number	Grade	Semester	Year	Section_id
Smith	CS1310	C	Fall	08	119
	MATH2410	B	Fall	08	112
Brown	MATH2410	A	Fall	07	85
	CS1310	A	Fall	07	92
	CS3320	B	Spring	08	102
	CS3380	A	Fall	08	135

(b)

COURSE_PREREQUISITES

Course_name	Course_number	Prerequisites
Database	CS3380	CS3320
		MATH2410
Data Structures	CS3320	CS1310

Figure 1.5
Two views derived from the database in Figure 1.2. (a) The TRANSCRIPT view.
(b) The COURSE_PREREQUISITES view.

데이타베이스 사용자의 분류

- 데이타베이스 관리자 (database administrator, DBA)
  - 데이타베이스 시스템의 관리를 책임진 사람
- 데이타베이스 설계자(database designer)
  - 데이타베이스의 설계를 책임진 사람
- 최종 사용자(end users)
  - 데이타베이스에 대하여 질의, 갱신, 보고서 작성 등을 담당하는 사람
  - 캐주얼 사용자(casual end users): 비정기적인 데이타베이스 사용자
  - 초보 사용자(parametric or naive users): 미리 일정한 용도로 작성된 프로그램을 사용하는 사용자; 은행 점원이나 여행사 예약 담당원 등
  - 전문 사용자(sophisticated end users): 복잡한 응용을 개발하며, DBMS의 기능을 충분히 사용하는 전문가
- 시스템 분석가/응용 프로그래머(system analysts / application programmers)
  - 초보 사용자를 위하여 잘 정의된 기능의 응용을 설계하고 구현하는 사람

데이타베이스 사용자 - 무대 뒤의 사람들

- DBMS 설계 및 구현자
  - DBMS 소프트웨어 자체를 설계하고 구현하는 업무를 담당하는 사람들
- 도구 개발자
  - 데이타베이스를 사용하는 데에 필요한 도구들 (데이타베이스 설계 및 구축 도구, 성능 도구, 인터페이스 등)을 설계하고 구현하는 사람들
- 운영 및 유지 보수 요원
  - 데이타베이스 시스템을 운영하는 데에 필요한 하드웨어 및 소프트웨어의 운영 및 유지 보수 담당 요원들

DBMS의 장점

- 중복성의 제어
  - 데이터 일치성 (consistency) 보장 및 메모리 낭비 방지
  - 다수 사용자 간의 데이타의 공유 및 동시 접근 보장
- 권한이 없는 접근의 통제
  - 보안과 권한 서브시스템
- 프로그램 객체를 위한 지속성 기억 공간 제공
  - 지속성 객체(Persistent Object)
  - Impedance Mismatch
- 효율적인 질의 처리를 위한 저장 구조와 탐색 기법의 제공
- 백업과 회복 제공
- 다수의 사용자 인터페이스 제공
- 데이터 간의 복잡한 관계의 표현
- 무결성 제약 조건(Integrity constraint)의 시행
  - 참조 무결성(Referential integrity)
  - 비즈니스 규칙(Business rule)
- 규칙을 사용한 추론과 수행
  - 연역(deductive) 데이터베이스 시스템
  - 트리거(Trigger)

데이터 불일치와 제어된 중복 : 예

- (a) 제어된 중복성: 성능을 위하여 GRADE_REPORT 화일에 StudentName과 CourseNumber를 포함시키고, 두 속성의 값이 Student에서의 두 속성값과 일치하도록 DBMS가 보장함
- (b) 비제어된 중복성: 그림 1.2의 STUDENT 레코드와 불일치하는 GRADE_REPORT 레코드의 예 (17번 학생은 Brown이 아니라 Smith 임)

Figure 1.6
Redundant storage of Student_name and Course_name in GRADE_REPORT. (a) Consistent data. (b) Inconsistent record.

(a)

GRADE_REPORT

Student_number	Student_name	Section_identifier	Course_number	Grade
17	Smith	112	MATH2410	B
17	Smith	119	CS1310	C
8	Brown	85	MATH2410	A
8	Brown	92	CS1310	A
8	Brown	102	CS3320	B
8	Brown	135	CS3380	A

(b)

GRADE_REPORT

Student_number	Student_name	Section_identifier	Course_number	Grade
17	Brown	112	MATH2410	B

DBMS의 사용 효과

- 표준화된 데이타 관리
  - 조직 내 모든 부서에서 표준화된 문서 관리로 업무 효율성 증대
- 데이타 구조 변경에 융통성 부여
  - 데이타베이스 내의 자료 구조가 어떠한 이유로 변경되어도 사용자에 대한 영향은 거의 없음
- 응용 프로그램의 개발 시간 단축
  - 응용 프로그램의 상당한 부분을 DBMS가 처리함
- 항상 최신의 정보를 제공
  - 사용자 중에서 한 사람의 갱신으로 나머지 사람은 즉시 변경된 값을 접근가능
- 규모의 경제성 (economics of scale)
  - 부서마다 다른 방식으로 자료를 관리하는 것보다 통합 DB로 관리하는 것이 전체적인 관점에서 저비용임

데이타베이스 응용의 간략한 역사

- 계층 시스템과 네트워크 시스템을 이용한 초기의 데이터베이스 응용 개발
  - 계층 모델과 망 모델 등은 60년대 중반부터 80년대까지 주류를 이루었고, 현재에도 종종 사용됨
- 관계형 데이타베이스를 통한 데이터 추상화와 응용의 유연성 제공
  - 관계형 모델은 70년대 소개 이후, IBM과 세계 각 대학에서 연구되고 검증되어 왔으며, 80년대에 들어서 상용 DBMS가 등장함
- 객체 지향 응용과 더욱 복잡한 데이터베이스에 대한 요구
  - OODBMS는 80년대에 소개되어 CAD와 같은 복잡한 데이터 처리 응용에 사용됨
- 전자 상거래(E-Commerce)를 위해 XML을 사용하여 웹에서 데이터 교환
  - HTML을 이용한 새 응용이 등장했으며, 최근에는 XML이 전자상거래를 위한 데이터 저장 및 자료 교환의 새로운 표준으로 자리잡고 있음
- 새로운 응용을 위한 데이타베이스 능력 확장 요구
  - 과학 응용, 이미지, 오디오, 비디오 저장 관리, 시공간 데이터 관리 기술 등

Database를 사용하지 않아도 좋은 경우

- DBMS를 사용하면 비용이 높아짐
  - 높은 초기 투자 비용과 추가적인 하드웨어 필요함
  - 데이타의 보안, 동시성 제어, 회복, 무결성 조건 등의 기능이 필요하지 않은 응용 - 오버헤드가 됨
- 언제 DBMS가 불필요한가?
  - 데이타베이스와 응용이 단순하고 잘 정의되어 있으며, 변경될 가능성이 적을 경우
  - DBMS 오버헤드로 인하여 엄격한 실시간 데이타 처리 요구사항을 만족시키기 힘든 경우 (최근들어 이러한 경우 실시간 DBMS 활용 가능)
  - 다 사용자 데이타 접근이 필요하지 않은 경우