Chapter 1

Databases and Database Users

Table of Contents

1.1 Introduction

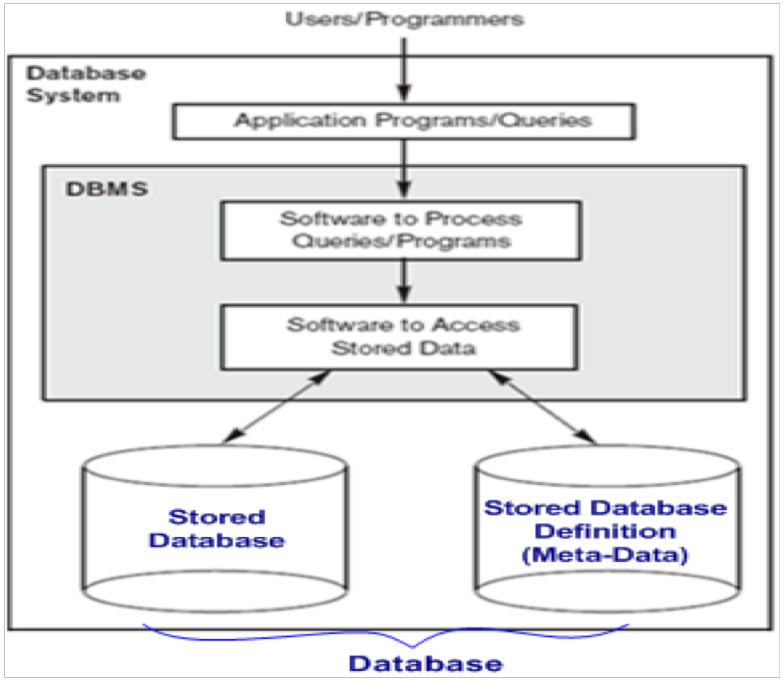
- **Data vs. Information**

ex) 풍량, 온도, 기압 -> 내일 기온 예측

  - Data is raw, unorganized facts that need to be processed
  - Information is the result of processing raw data to reveal its meaning
- **Database**
  - A collection of related data used by the plural users and applications in an organization

이런 메타 데이터가 존재해서 데이터 공용성을 보장

  - A database represents some aspect of the real world, i.e. a miniworld
  - Database consists of not only stored DB but also stored DB definition
- **DBMS (DataBase Management System)**
  - A collection of programs that enables you to define, store, modify, and extract information from a database
  - Ex: Oracle, MySQL, SQL Server, DB2, Altibase HDB, CUBRID, TIBERO



- **Database system**
  - Database system=Database + DBMS + (Users & Application programs)

1.1 Introduction

- **A DBMS is a general-purpose but complex software system** 시스템 소프트웨어의 일종
  - Defining a database
    - Specify the data types, structures, and constraints of the data
    - DB definition or descriptive information (meta-data) is also stored by the DBMS in the form of a system catalog ddl dml
  - Constructing a database
    - Process of storing the data on some storage medium
  - Manipulating a database dml의 데이터 삽입을 통해 저장
    - Query, update, generate reports
  - Sharing a database 컨커러싱 모듈이 이를 수행
    - Allows multiple users and programs to access the DB simultaneously
  - Protection includes:
    - System protection, Security protection 비밀번호, 각 사용자마다 db 접근 제한(dba가 권한 제한을 함)
  - Maintain a database system
    - Allow the system to evolve as requirements change over time

1.2 An DB Example

- **UNIVERSITY database**
  - Information concerning students, courses, and grades in a university environment
  - It is organized as five tables such as STUDENT, COURSE, SECTION, GRADE_REPORT, and PREREQUISITE, each of which stores data records of the same type

첫 번째 단계인
사용자 요구 분석 단계

논리적 설계
- 테이블 이름, 속성 정의 - 스키마
- 실제 데이터 있는 부분 - 인스턴스

- **Define the structure of the records of each table by specifying different types of data elements and their data types**

- **Construct UNIVERSITY database**
  - Store data to represent each student, course, section, grade report, and prerequisite as a record in an appropriate table
  - Notice that records in the various tables may be related
    - Smith in STUDENT is related to two records in GRADE_REPORT

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

스키마

인스턴스

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.

1.2 An DB Example

- **Manipulation involves querying and updating**
  - Examples of queries:
    - List the prerequisites of the 'Database' course 두 개의 테이블을 조인해서 질의해야 함
    - List the names of students who took the section of the 'Database' course offered in fall 2008 and their grades in that section 4개의 테이블
  - Examples of updates:
    - Change the class of 'Smith' to sophomore
    - Create a new section for the 'Database' course for this semester

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

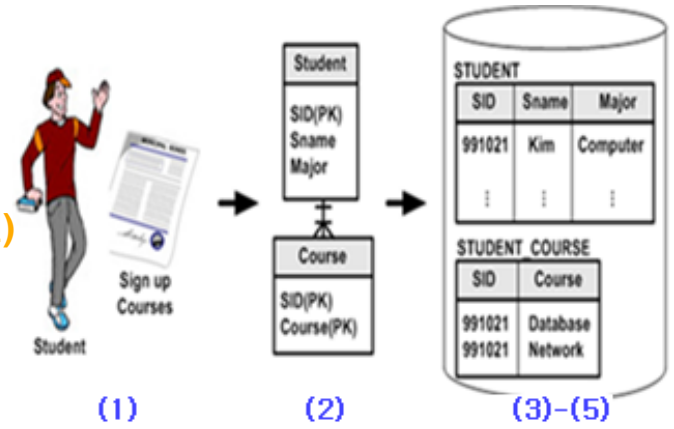
GRADE_REPORT

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

- **Database design phase:**
  (1) Requirements specification and analysis (Ch3)
  (2) Conceptual design (Ch3, Ch4) er 다이아그램(실세계 표현)
  (3) Logical design (Ch5, Ch9, Ch14)
  (4) Physical design (Ch16, Ch17)
  (5) Implementation phase (Ch6, Ch7, Ch10, Ch11)

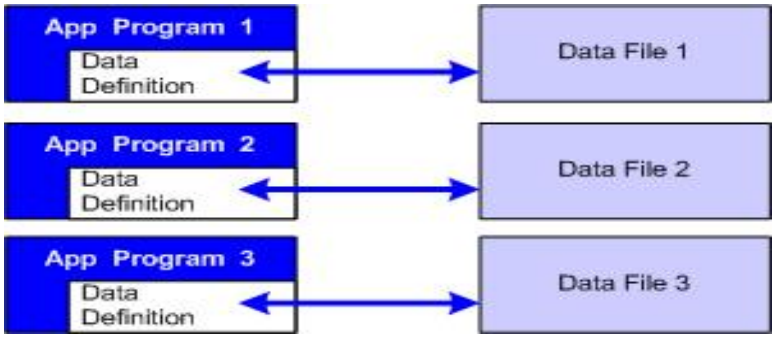


1.3 Characteristics of the DB Approach

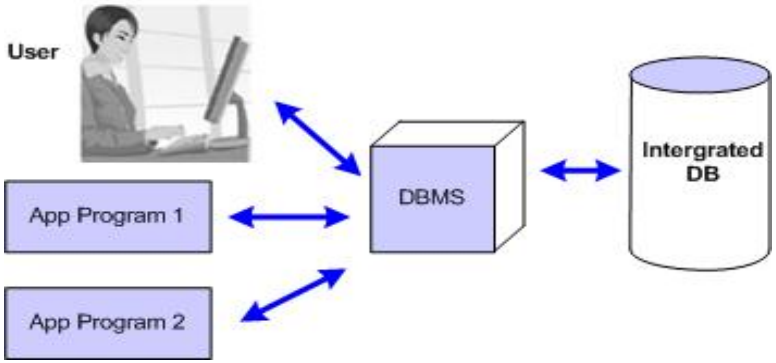
- **Traditional file processing**
  - Each user defines and implements the files needed for a specific software application
  - Data redundancy and inconsistency

문제 : 데이터의 중복(홍길동) - 저장소 낭비, 다른 곳에 반영 x

db를 쓰면서 이 두개의 문제점을 해결



- **Database approach**
  - Single repository maintains data that is defined once and then accessed by various users



- **Main characteristics of database approach**
  - Self-describing nature of a database system
  - Insulation between programs and data, and data abstraction
  - Support of multiple views of the data
  - Sharing of data and multiuser transaction processing

1, 2 데이터 독립성

1.3.1 Self-Describing Nature of a DB System

- **In file processing, the structure of data files is embedded in the application programs, so any changes to the structure of a file require changing all programs that access that file**

- **DBS contains complete definition of structure and constraints**
  - Structure of each table, the type and storage format of data item, various constraints on the data
- **System catalog**
  - Meta-data
  - Describes structure of the DB
- **System catalog used by**
  - DBMS software
  - DB users who need information about DB structure such as a DBA

주로 dbms가 사용
숙련된 dba또한 여기에 접근해 사용 가능

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

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

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Figure 1.2
A database that stores student and course information.

이 두개는 메타 데이터

RELATIONS

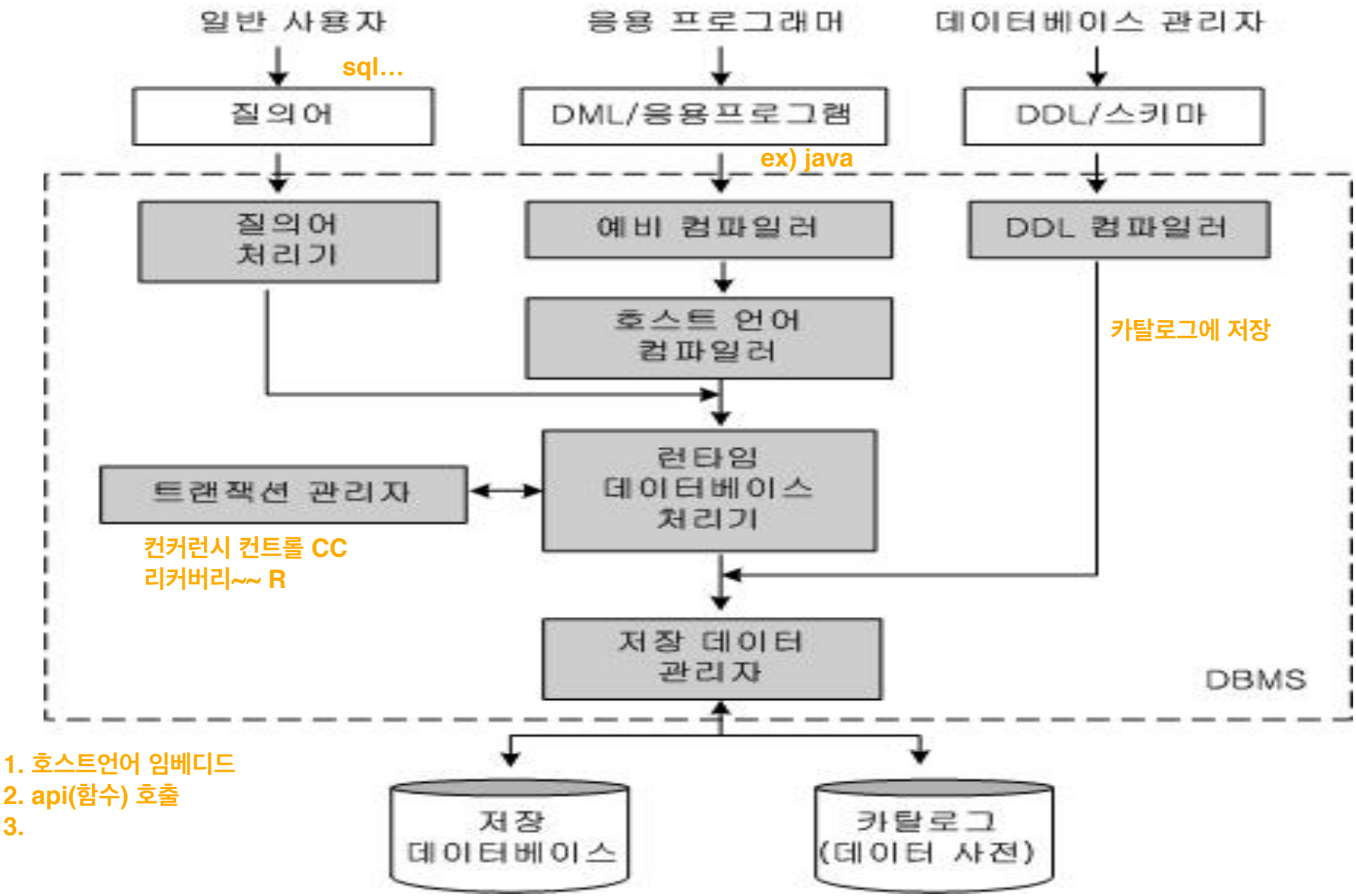
Relation_name	No_of_columns
STUDENT	4
COURSE	4
SECTION	5
GRADE_REPORT	3
PREREQUISITE	2

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.



1. 호스트언어 임베디드
2. api(함수) 호출
3.

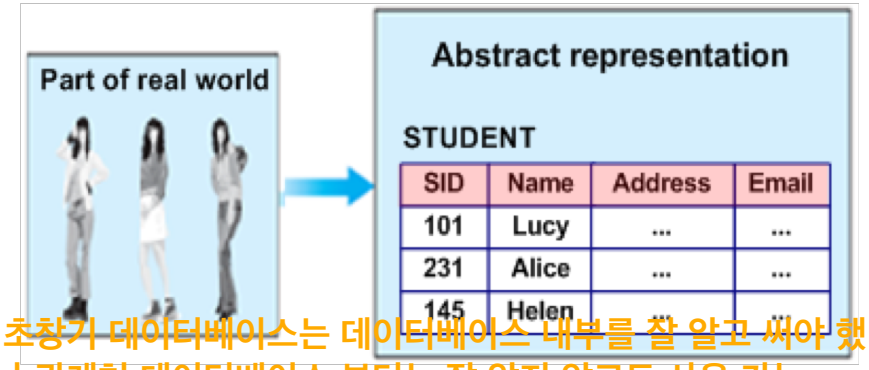
1.3.2 Insulation Between Programs and Data, and Data Abstraction

객체지향 데이터베이스 / 객체 관계형 데이터베이스

- **Program-data independence**
  - The ability to change the organization of a DB without changing the application software that uses it
  - Structure of data tables is stored in DBMS catalog separately from access programs
- **Program-operation independence**
  - In some types of DBSs such as OO and OR DBSs, users can define operations on data as part of the DB definition

오라클 같은 대부분 데이터베이스는
OR 깩체관계형임

  - Operation is specified in two parts:
    - Interface (or signature) includes operation name and data types of its arguments
    - Implementation can be changed without affecting the interface
  - Application can work on the data by invoking operations through their names and arguments regardless of how the operations are implemented
- **Data abstraction by program-data and program-operation independence**
  - Data abstraction provides only essential information to the outside world and hiding their background details
  - A DBMS provides users with a conceptual representation of data that does not include many of the details of how the data is stored or how the operations are implemented.
  - Informally, a data model is a type of data abstraction that is used to provide a conceptual representation of data



과거 초창기 데이터베이스는 데이터베이스 내부를 잘 알고 써야 했다
그러나 관계형 데이터베이스 부터는 잘 알지 않고도 사용 가능

추상화키셔서 표현 가능 -> 데이터 추상화의 종류로 볼 수 있음

1.3.3 Support of Multiple Views of the Data

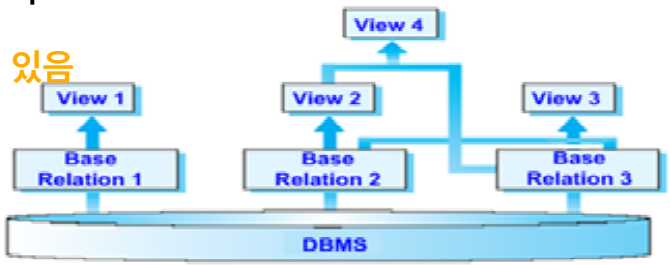
가상의 테이블
실제 테이블로부터 뷰를 만들어냄

- **View**
  - Contains virtual data derived from the DB files but is not explicitly stored
  - Virtual table based on the result set of a stored query on the data
  - DB users can query the views just as they are in a persistent DB

뷰를 생성하는 명령어가 존재
creat view

- **Multiuser DBMS**
  - Users have a variety of distinct applications
  - Must provide facilities for defining multiple views

사용자가 뷰를 변경하면
기존 테이블에 반영되는게 있고 안되는 것도 있음



(a)

Student Section Grade_Report

TRANSCRIPT

StudentNAME	Course Number	Grade	Semester	Year	SectionId
Smith	MATH2410	C	Fall	92	119
	COSC1310	B	Fall	92	112
Brown	COSC3320	A	Fall	91	85
	MATH2410	A	Fall	91	92
	COSC1310	B	Spring	92	102
	COSC3380	A	Fall	92	135

4개의 테이블에 쿼리할 필요 없이
이 뷰 하나만 보면 알 수 있음

여기서 grade를 없앤다면
보안의 역할도 할 수 있음

(b)

Course Prerequiste

PREREQUISITES

CourseName	CourseNumber	Prerequisites
Database	3380	COSC3320
		MATH2410
Data Structures	3320	COSC1310

1.3.4 Sharing of Data and Multiuser Transaction Processing

- **Allow multiple users to access the DB at the same time**
- **Concurrency control software**
  - Ensure that several users concurrently trying to update the same data in a controlled manner to get the correct result

T1 다 끝나고 T2 하면 문제x
io 등의 이유로 다 끝나지 않고 다른 연산 할 수 있음
읽기 문제 x, 데이터 변경시 문제발생 가능
-> 병행해주되 시리얼라이즈 해주게 보장해줌(락)

- **Transaction** 하나의 사용자가 여러개의 연산을 할 수 있는데, 여러 연산을 한번에 하는 것처럼
  - A collection of operations that form a single logical unit of work such as an account transfer
  - Ex: In double-entry accounting every debit requires the recording of an associated credit. If one writes a check for $100 to buy groceries, the accounting system must record the following two entries to cover the single transaction:

단위작업으로 만듬

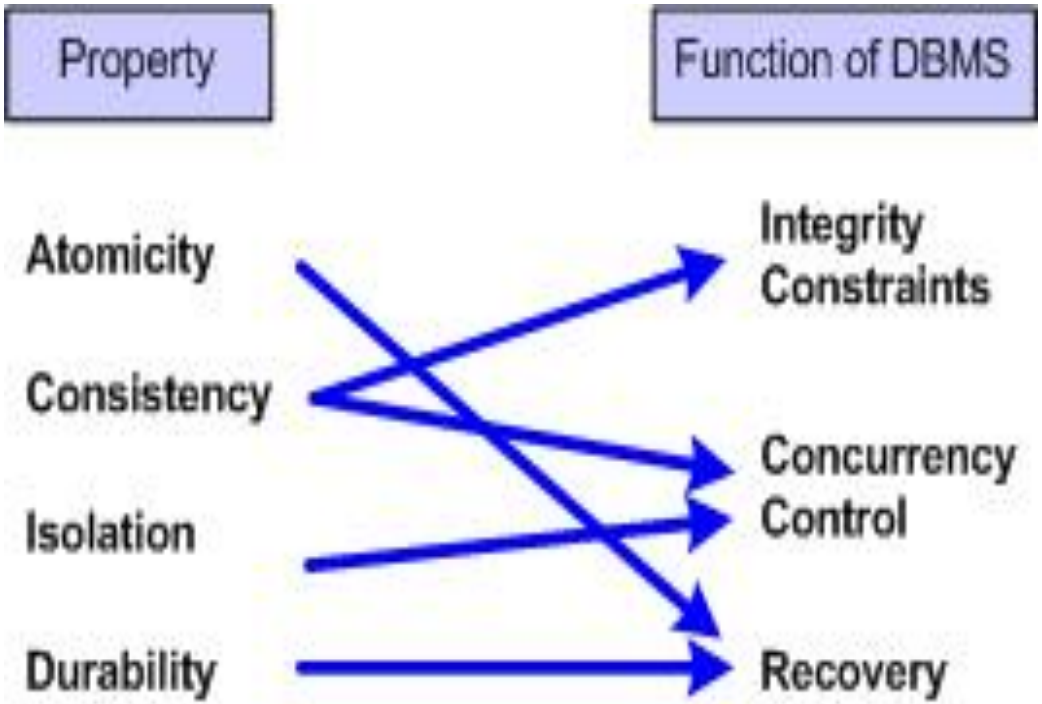
1) Debit $100 to Expense Account
2) Credit $100 to Checking Account

a->b 입금 (출금 -> 입금)
만약 중간에 시스템 다운 되면 돈을 잃게 됨
all or nothing - 다운되면 다시 원래대로 복구

  - DBMS maintains the integrity of the data by treating the multiple operations of a transaction as an atomic unit of work
  - In other words, nobody ends up with a situation in which a debit is recorded but no associated credit is recorded, or vice versa
- **On-line Transaction Processing (OLTP)** 질의도 간단, 빠름
  - OLTP refers to processing of a large number of short on-line transactions carrying on a very fast query and maintaining data integrity in multi-access environments such as banking, airline, etc

OLAP(분석작업) 복잡, 데이터마이닝

```
BEGIN TRAN
    INSERT publishers VALUES (1111, 'PEARSON', 'Seoul', '', 'South Korea');
    UPDATE publishers SET pub_name = 'Old Book' WHERE pub_id = '0736';
    IF @@ERROR = 0  COMMIT TRAN
    ELSE            ROLLBACK TRAN;
END TRAN
```

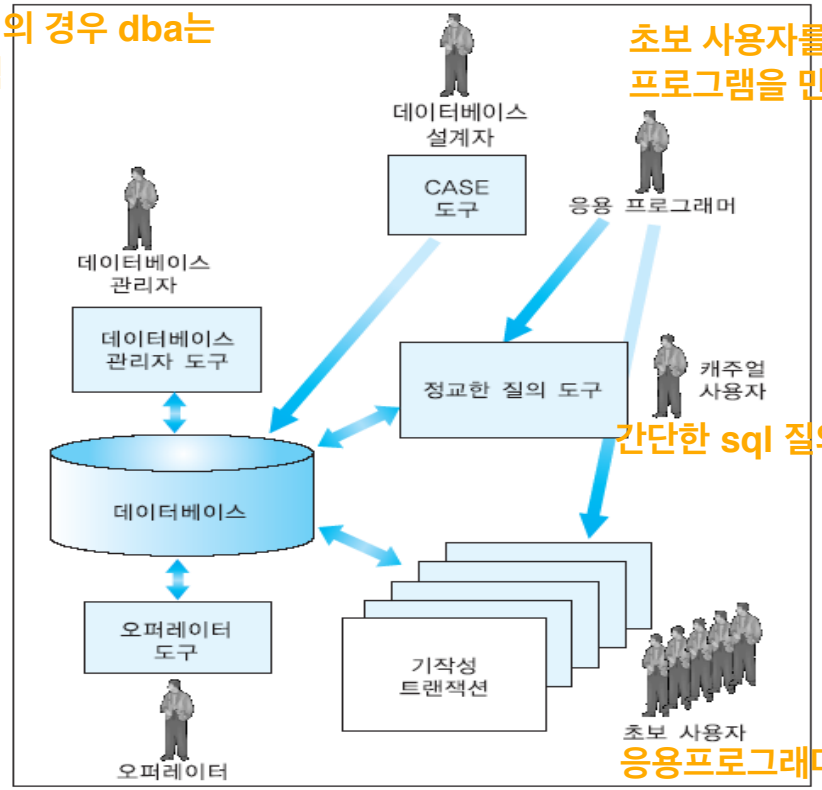


1.4 Actors on the Scene

- **Database administrators (DBA) are responsible for:**
  - Create and alter a database schema
  - Authorizing access to the database
  - Define storage structure and access path
  - Coordinating and monitoring DB
  - Take charge of a backup and recovery
  - Acquiring software and hardware resources
- **Database designers**
  - Identifying the data to be stored
  - Choosing appropriate structures to represent and store this data
  - views of DB
- **End users**
  - People whose jobs require access to DB
  - Types: Naive or parametric end users, Casual end users, Sophisticated end users, Standalone users
- **System analysts**
  - Determine requirements of end users and develop specifications for standard canned transactions that meet these requirements
- **Application programmers**
  - Implement these specifications as programs

보통 기업의 경우 dba는 고위 직책

초보 사용자를 위해 프로그램을 만듬



[그림 1.17] DBMS의 사용자들

간단한 sql 질의등을 함

응용프로그래머가 만든 프로그램을 사용, sql 질의 몰라도 됨

초보사용자

캐주얼 사용자

과학자, 비즈니스 분석자
디비 질의를 통해 정보를 얻을수 있음

초보사용자가 쉽게 사용할 수 있게

1.5 Workers behind the Scene

- **DBMS system designers and implementers**
  - Design and implement the DBMS modules as a software package
  - Modules for implementing the catalog, query language processing, interface processing, handling data recovery and security, controlling concurrency, accessing and buffering data, etc

앞 페이지 트랜잭션 관리자에 속함

data recovery - log에 최소한의 정보를 가지고 있다가 리커버리를 함

- **Tool developers** er다이아그램 툴도 이사람들이 만듬
  - Design and implement tools
  - They include packages for DB design, performance monitoring, natural language or graphical interface, prototyping, simulation, and test data generation

- **Operators and maintenance personnels** 전산실 유지보수
  - Responsible for running and maintenance of hardware and software environment for database system

1.6 Adv of Using the DBMS Approach

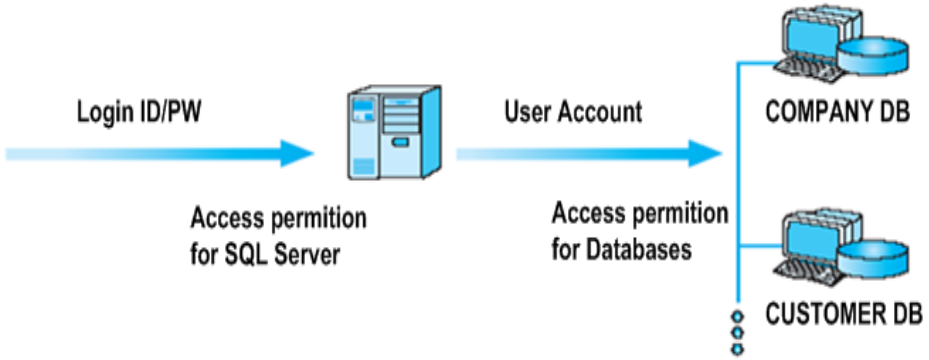
- **Controlling redundancy**
  - Redundancy in storing the same data multiple times leads to several problems such as duplication of effort, waste storage space and data inconsistency
  - We should have a DB design that stores each logical data item in only one place in the DB
    - Data normalization avoids data redundancy

Redundancy

EMP_DEPT

Ename	Ssn	Bdate	Address	Dnumber	Dname	Dmgr_ssn
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston, TX	5	Research	333445555
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston, TX	5	Research	333445555
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4	Administration	987654321
Wallace, Jennifer S.	987654321	1941-06-20	291 Berry, Bellaire, TX	4	Administration	987654321
Narayan, Ramesh K.	666884444	1962-09-15	975 FireOak, Humble, TX	5	Research	333445555
English, Joyce A.	453453453	1972-07-31	5631 Rice, Houston, TX	5	Research	333445555
Jabbar, Ahmad V.	987987987	1969-03-29	980 Dallas, Houston, TX	4	Administration	987654321
Borg, James E.	888665555	1937-11-10	450 Stone, Houston, TX	1	Headquarters	888665555

- **Restricting unauthorized access**
  - Security and authorization subsystem



DEPT

Dname	Dnumber	Dmgr_ssn
Research	5	333445555
Administration	4	987654321
Headquarters	1	888665555

정규화 과정을 통해
두 개의 테이블로 나눔

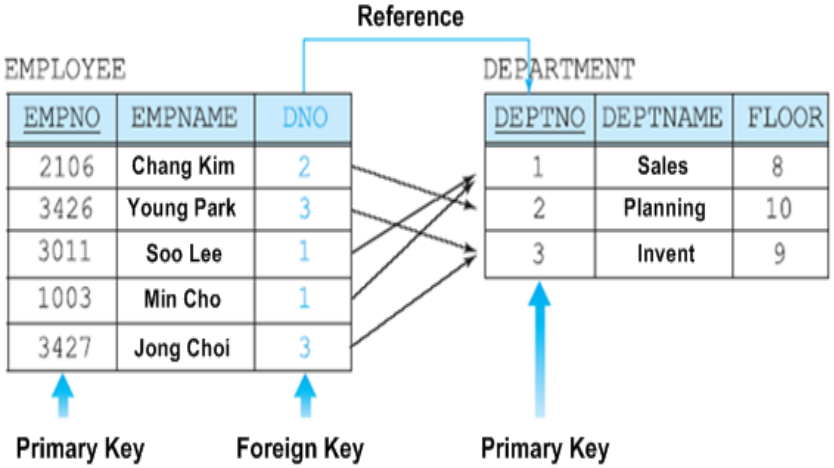
EMP

Ename	Ssn	Bdate	Address	Dnumber
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston, TX	5
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston, TX	5
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4
Wallace, Jennifer S.	987654321	1941-06-20	291Berry, Bellaire, TX	4
Narayan, Ramesh K.	666884444	1962-09-15	975 Fire Oak, Humble, TX	5
English, Joyce A.	453453453	1972-07-31	5631 Rice, Houston, TX	5
Jabbar, Ahmad V.	987987987	1969-03-29	980 Dallas, Houston, TX	4
Borg, James E.	888665555	1937-11-10	450 Stone, Houston, TX	1

- **Providing backup and recovery**
  - Backup and recovery subsystem of the DBMS is responsible for recovery
- **Representing complex relationships among data** 복잡한 관계에 대해 정의, 질의등을 할 수 있음
  - May include numerous varieties of data that are interrelated in many ways

1.6 Adv of Using the DBMS Approach

- **Providing storage structures and search techniques for efficient query processing**
  - Buffer 사용될 것 같은 것을 버퍼에 저장 - 빠름
  - Indexes
  - Query optimization



- **Enforcing integrity constraints**
  - Specifying a data type for each data item
  - Referential integrity constraint
    - Every EMPLOYEE record must be related to a DEPARTMENT record

다른 테이블을 참조하고 있기 때문에

  - Key or uniqueness constraint
    - Every EMPLOYEE record must have a unique value for EMPNO
  - Business rules

STUDENT

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

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

- **Providing multiple user interfaces**
  - Query language, programming language interfaces, forms, GUI, etc

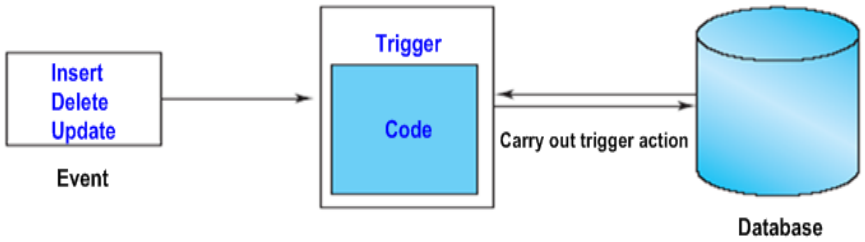
이 테이블의 경우
Student_number 와
Section_iden 를 합쳐 기본키가 됨

1.6 Adv of Using the DBMS Approach

- **Permitting inferencing and actions using rules**
  - Deductive DB systems
    - Provide capabilities for defining deduction rules
    - Inferencing new information from the stored database facts

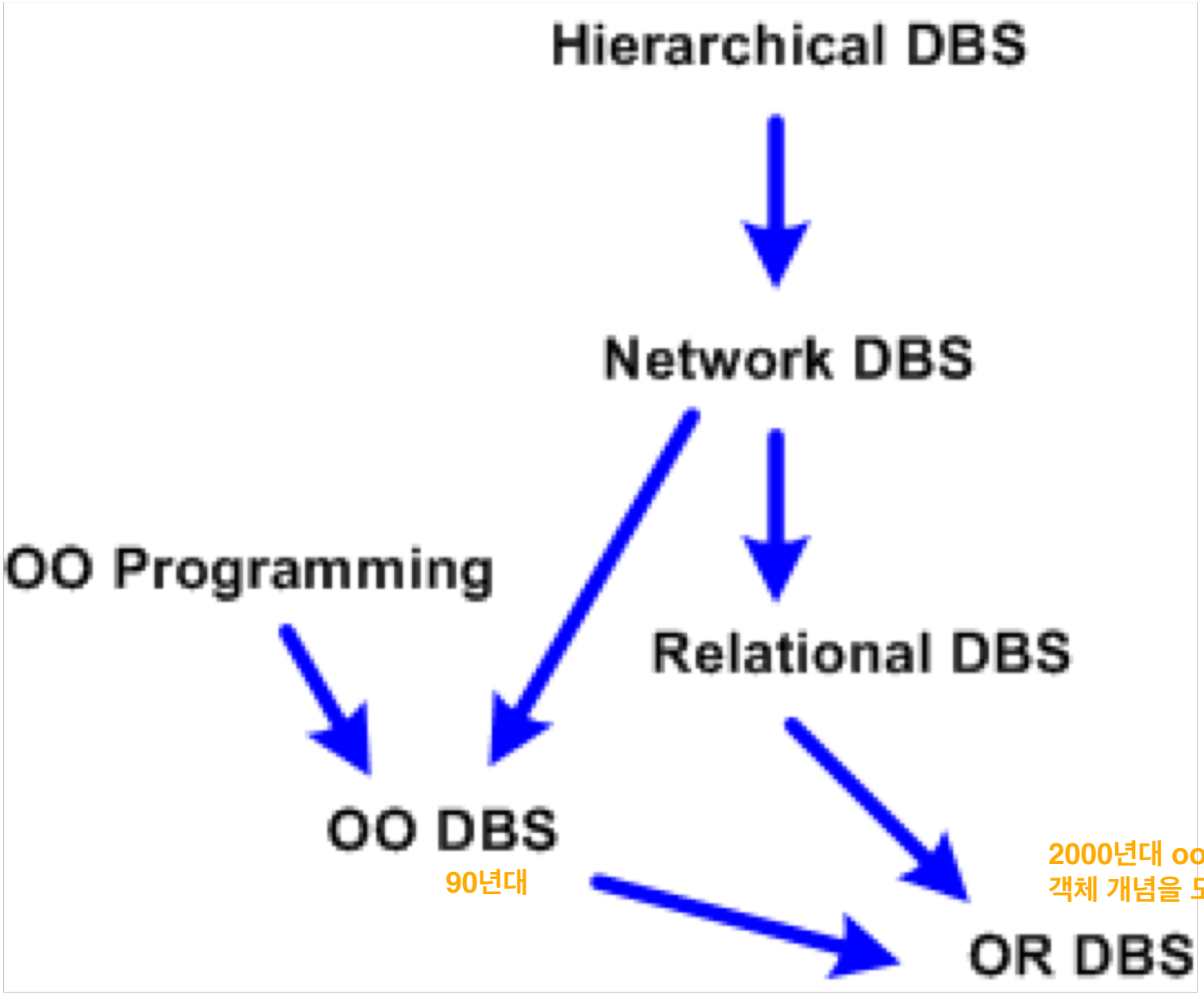
이러한 추론을 지원

» 사실) All men are mortal.
» 사실) Aristotle is a man.
» 추론) Therefore, Aristotle is mortal.



  - Trigger ECA 모델을 닮은 하나의 프로그램
    - The SQL statements that are executed automatically when a DB operation violates a certain constraint ex) 부사장의 월급 수정 trigger 2배 이상? -> 저장x, 다르게 저장 하는 등 처리
  - Stored procedures 시스템 저장 프로시저 / 사용자 저장 정의 프로시저
    - A precompiled set of SQL statements with an assigned name, which can be shared by a number of programs 한번 만들어지면 코딩 필요 없이 호출해서 바로 사용 가능하다
  - Active DB systems
    - Provide active rules that can automatically initiate actions when certain events and conditions occur (ECA rule)
- **Additional implications of using the database approach**
  - Potential for enforcing standards 공유
  - Reduced application development time 응용 프로그램 개발 시간 단축
  - Flexibility 유연성 - 변경등이 쉬움
  - Availability of up-to-date information 모든 사용자가 최신의 데이터 확인이 가능

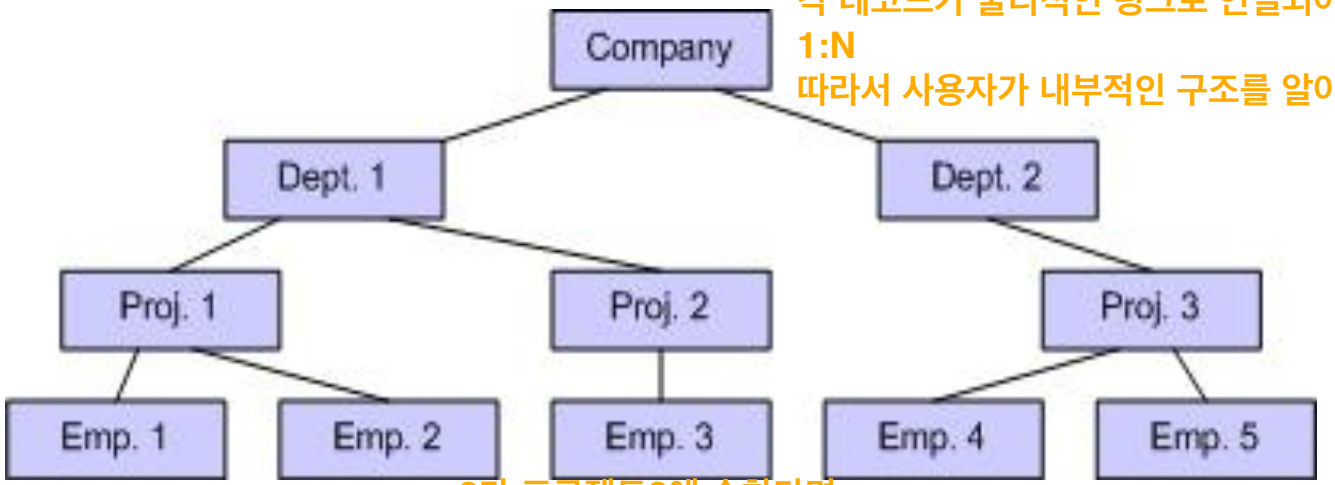
1.7 A Brief History of DB Applications



1.7 A Brief History of DB Applications

- **Hierarchical DBS (1960)**
  - Data is stored as records which are connected to one another through links
  - Data is organized into a tree-like structure where each record has only one parent but can have several children (1:M relationship)
  - Pro: Works well for DB applications that are naturally hierarchical
  - Cons: Lack of data independence and data abstraction
    - Each relationship between records should be specified when DB is constructed
    - A low level or procedural DML must be embedded in a general-purpose programming language
    - It is difficult to modify the structure of records because records are connected by links
    - M:N relationships between record types cannot be directly represented
      - Data duplication

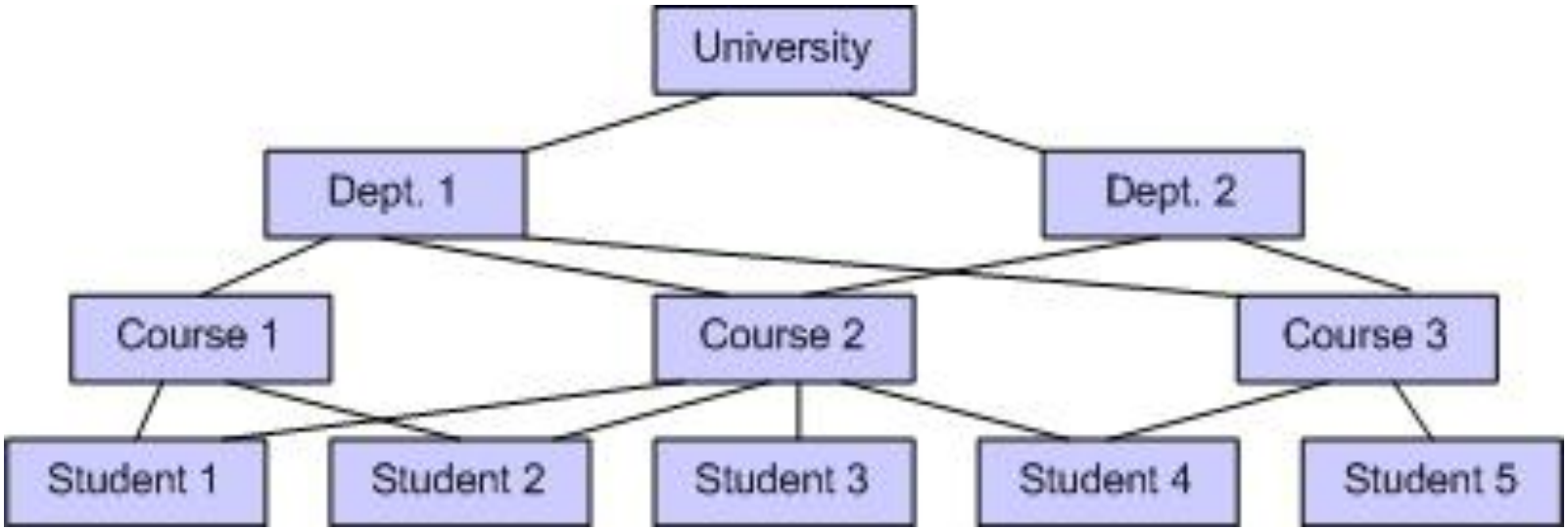
각 레코드가 물리적인 링크로 연결되어 있음
1:N
따라서 사용자가 내부적인 구조를 알아야 함(로우레벨 질의)



emp2가 프로젝트2에 속한다면
proj2에 새롭게 만들어 추가해야 함

1.7 A Brief History of DB Applications

- **Network DBS (1970)**
  - Data is stored as records which are connected to one another through links
  - Data is organized into a network (or graph) structure where the relationships among records are of type many-to-many (M:N relationship)
  - Hierarchical data model is a special case of network data model
  - Pro: Allows the representation of arbitrary relationship
  - Cons: Similar problems as those of Hierarchical DBS

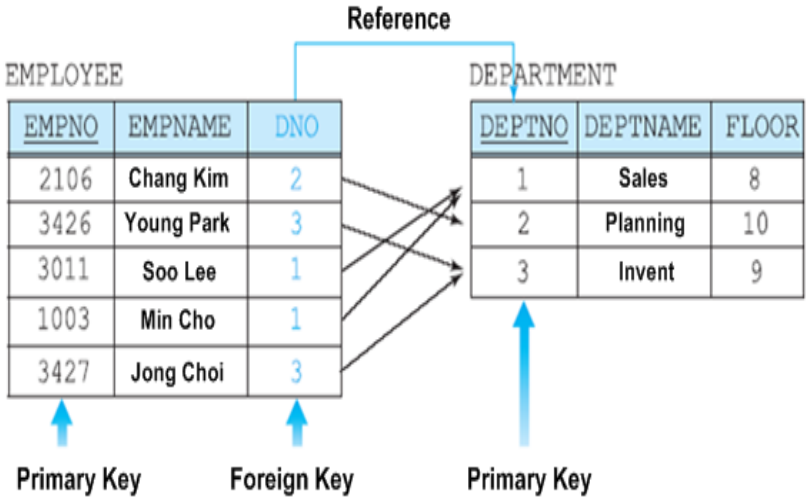


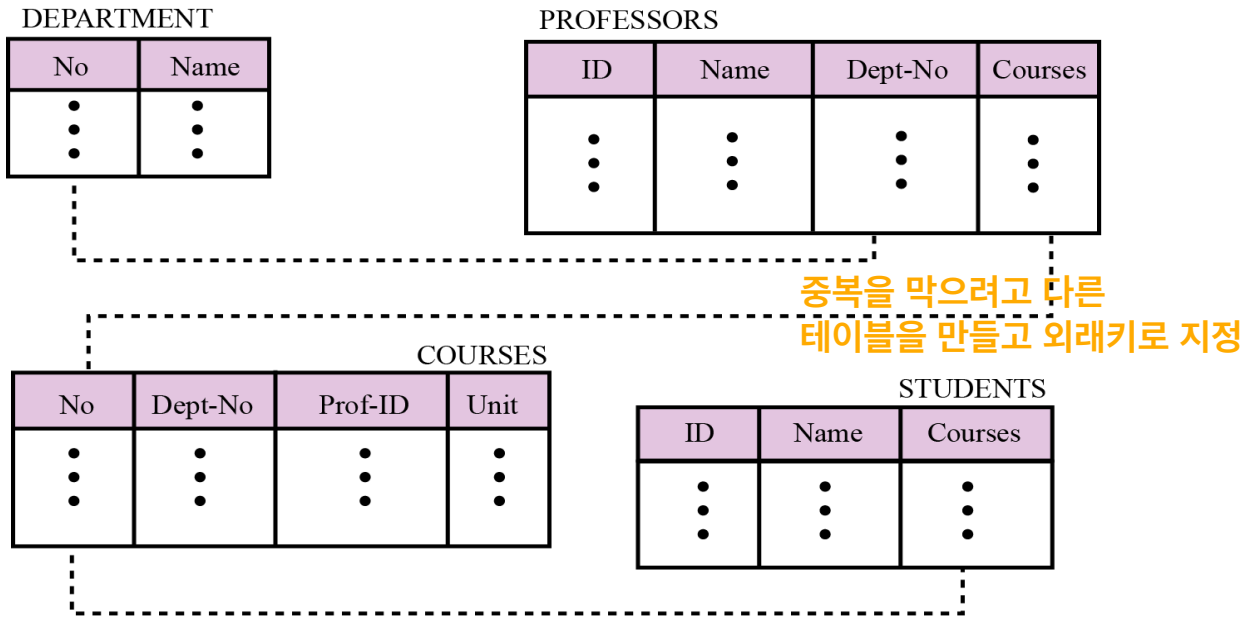
1.7 A Brief History of DB Applications

- **Relational DBS (1980)**
  - A DBMS in which the data items and the relationships among them are organized into tables consistently 물리적 링크 존재 x
  - Separates physical storage of data from its conceptual representation
  - Provides data abstraction and program-data independence
  - Declarative query: Users declare what they want without having to write a step-by-step procedure

암묵적인 관계성

선언적인 질의 - 내부적인 구조를 모르고 질의 가능(sql)
how를 모르고 what





Ex of the relational model representing a university

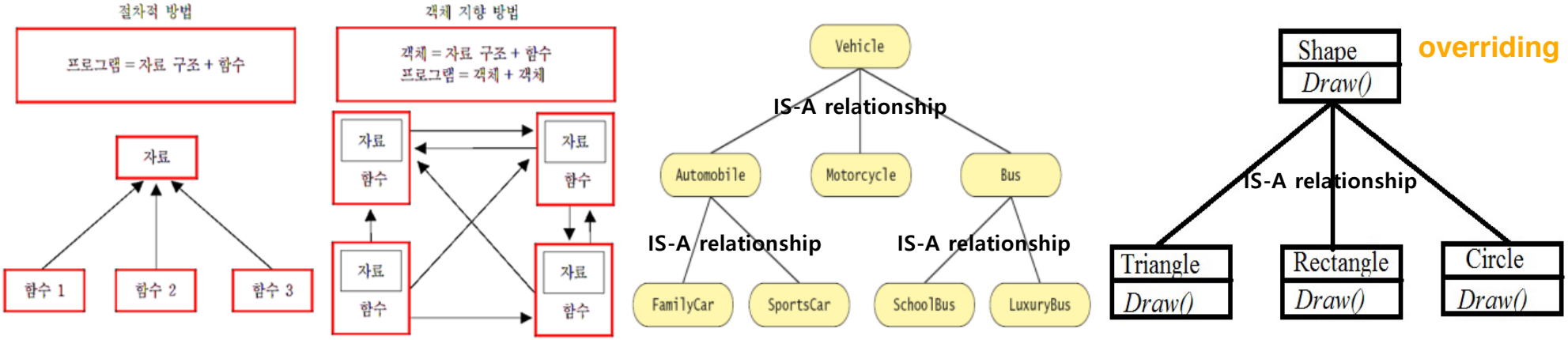
1.7 A Brief History of DB Applications

- **Object Oriented paradigm**

- The world is made up of objects such as people, automobiles, buildings, etc
- OO programming treats a program as a collection of objects that interact by means of actions. It simplifies the software development and maintenance by providing the following concepts:
  - Object is an active program unit containing both data and procedures (methods)
  - Class is a template from which objects are constructed. An object is called an instance of a class.
  - Encapsulation is a way of restricting access to the internal components of an object
  - Abstraction is a way to represent the essential feature without representing the back ground details
  - Inheritance allows a class at lower levels inherits all the characteristics of classes above it in the hierarchy.
  - Polymorphism allows the different objects being able to response in a different way with the identical messages

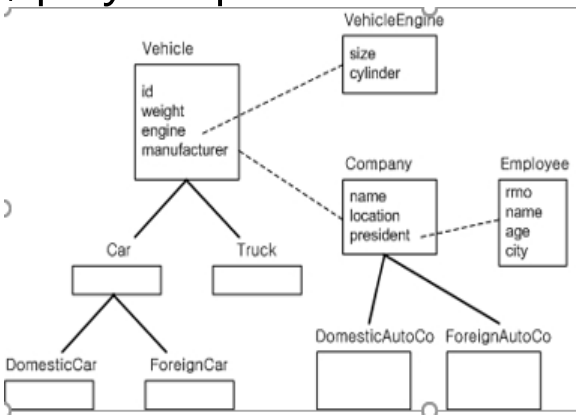
클래스 내부 정보 숨기는 것

override / overload



1.7 A Brief History of DB Applications

- **Object Oriented DBS (1990) – Ch12**
  - A database constructed by applying the OO paradigm such as object, encapsulation, information hiding, abstraction, inheritance, polymorphism
  - Compatible with programming languages such as C++ and can manipulate persistent objects
  - Cons:
    - Complex entities can be modeled, maintained and extended easily
    - Allows the user-defined operations on the DB objects
  - Pro:
    - The complexity of the model and the lack of an early standard
  - Ex: ONTOS, OpenODB, GemStone, ObjectStore, Versant, O2, etc



실선 : is a
점선 : 관계표현

- **Object Relational DBS (2000) – Ch12**
  - Extends the relational data model by including OO functions
  - Allows attributes of tuples to have complex types, including non-atomic values such as nested relations
  - Preserves relational foundations, in particular the declarative access to data, while extending modeling power
  - Ex: Oracle, SQL Server, etc

속성이 원자값 vs 복잡한 데이터 타입을 가질 수 있음

name	*address*		*birthdate*	*movies*		
	street	*city*		*title*	*year*	*length*
Fisher	Maple	H'wood	9/9/99	Star Wars	1977	124
	Locust	Malibu		Empire	1980	127
				Return	1983	133
Hamill	*street*	*city*	8/8/88	*title*	*year*	*length*
	Oak	B'wood		Star Wars	1977	124
				Empire	1980	127
				Return	1983	133

복잡한 데이터 타입을 지정할 때
메소드 또한 지정이 가능

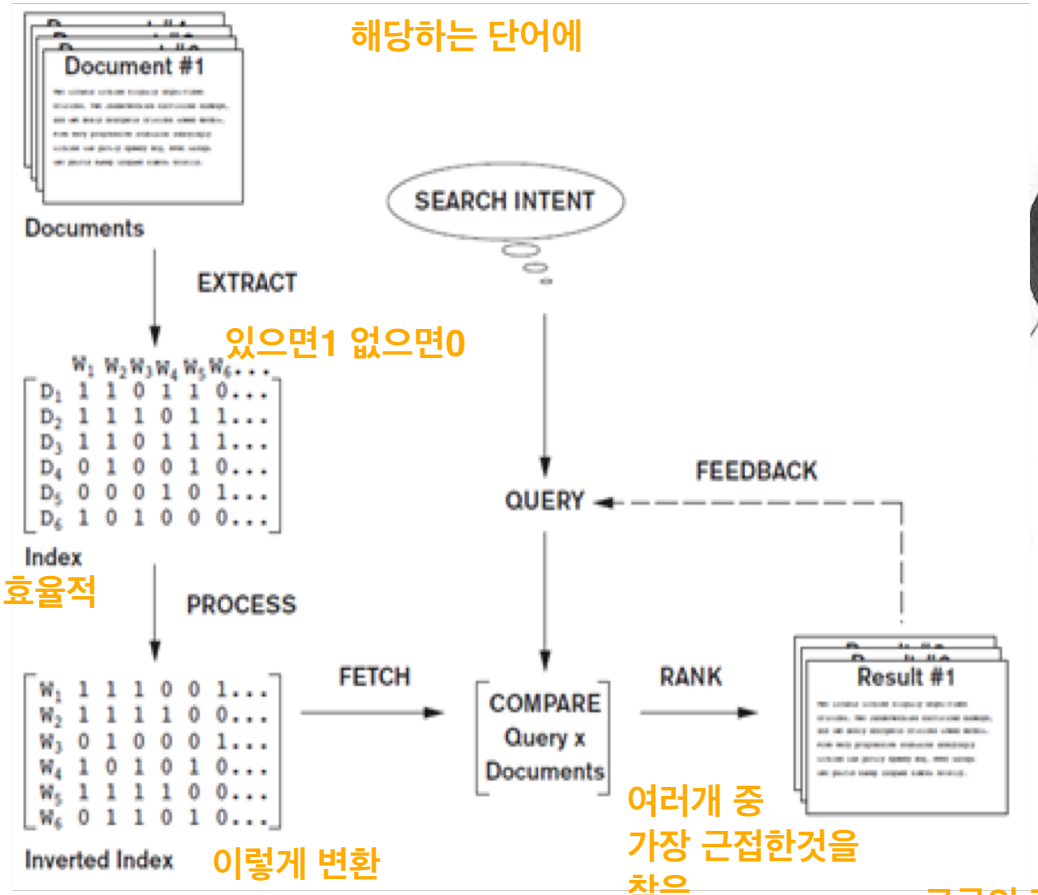
1.7 A Brief History of DB Applications

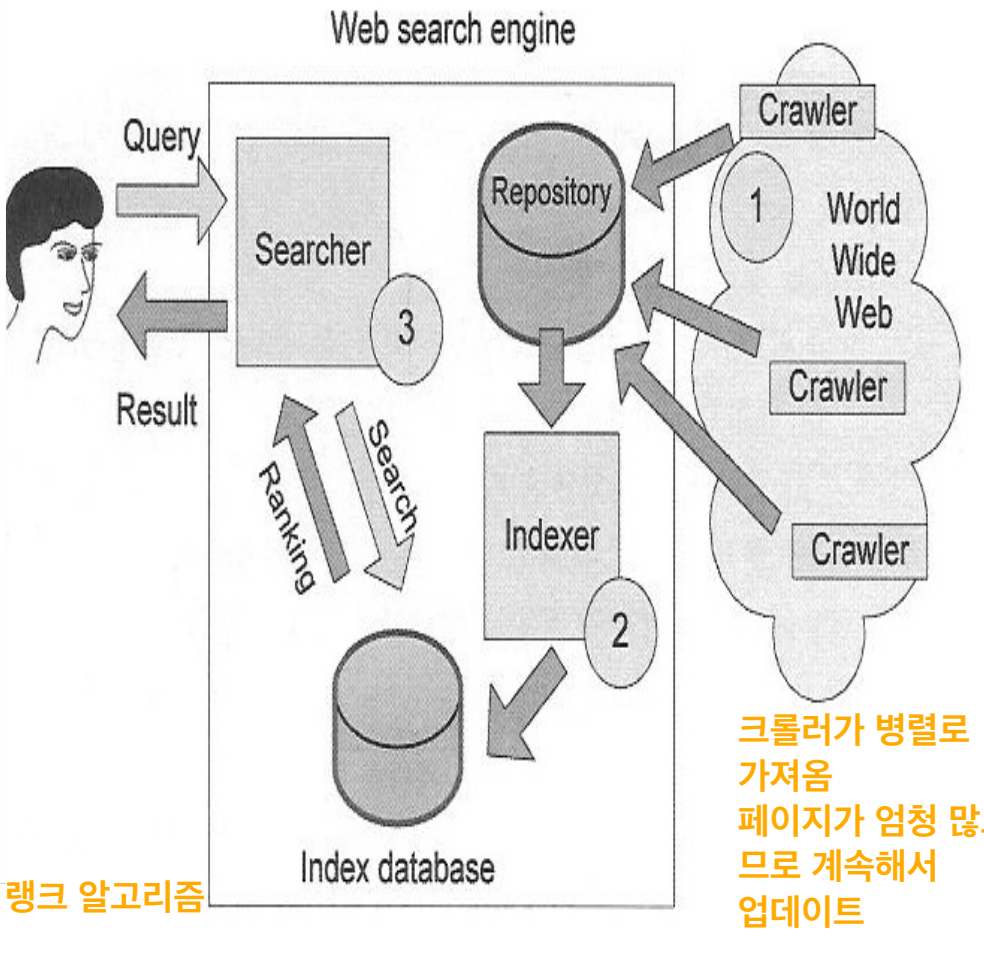
- **Enterprise resource planning (ERP)** 전사관리시스템
  - Business managements software—typically a suite of integrated applications—that a company can use to collect, store, and manage data from many business activities
  - SAP, 오라클, 마이크로소프트, LG CNS EAP(Enterprise Application Platform)
- **Customer relationship management (CRM)** 고객관리시스템
  - Entails all aspects of interaction that a company has with its customers, whether it is sales or service-related
- **Parallel DBS** 속도를 높이는 병렬처리
  - Seeks to improve performance through parallelization of various operations, such as loading data, building indexes and evaluating queries
- **Distributed DBS - Ch 23** 데이터가 너무 커서 분산함
  - A database stored on several computers that communicate through the Internet or a private wide area network
  - Transactions may access data at one or more sites 필요에 따라 트랜젝션이 여러 디비에 접근 가능
  - DBMS will mask this organizational detail from its users

사용자 입장에서는 내부적인 것을 모르더라도 사용할 수 있도록 지원할 필요가 있다

1.7 A Brief History of DB Applications

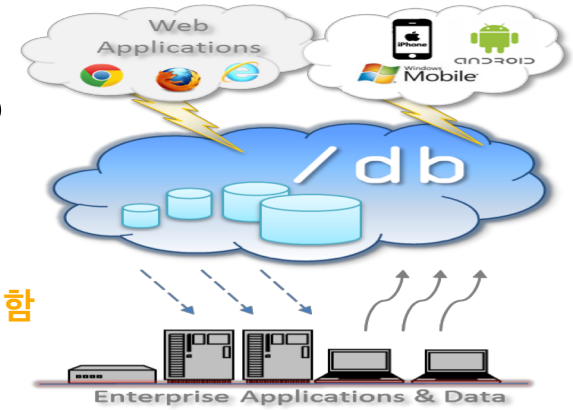
- **Information retrieval - Ch 27**
  - The activity of obtaining information from a collection of information resources such as books, manuscripts, and various forms of articles
  - With the advent of the Web, there is a need to apply many of the IR techniques to process data on the Web





1.7 A Brief History of DB Applications

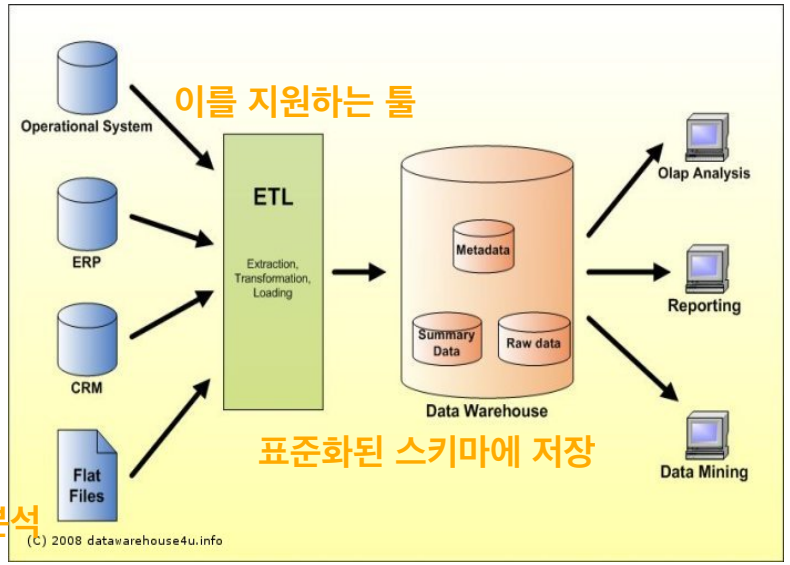
- **Temporal DBS** provides a uniform and systematic way of dealing with historical data - Ch 26 기본 디비는 업데이트 하면 기존 값 사라짐, 기존의(히스토리컬)데이터를 저장함 - 병원
- **Deductive DBS** can make deductions (i.e., conclude additional facts) based on rules and facts stored in the DB - Ch 26 트리거 - eca에 기반한 메커니즘
- **Active DBS** includes an event-driven architecture (ECA rules) which can respond to conditions both inside and outside the DB - Ch 26
- **Spatial DBS** is optimized to store and query data that represents objects defined in a geometric space - Ch 26 2차원/3차원 지리정보시스템(네비) - 주변질의 같이 추가적인 질의 필요
  - GIS(Geographic Information System), CAD/CAM database
- **Multimedia DBS** is a collection of related multimedia data such as text, images, graphic objects, animation sequences, audio and video - Ch 26
  - Content-based query 음악 같은거
- **Real-time DBS**
  - A database system which uses real-time processing to handle workloads whose state is constantly changing
  - It must produce results while meeting predefined deadlines of transactions 트랜잭션마다 데드라인이 있어 그때마다 끝내야 함 스케쥴링 기법이 필요!!



- **Cloud DBS**
  - A database that typically runs on a cloud computing platform which provides shared processing resources and data to users on-demand through Internet

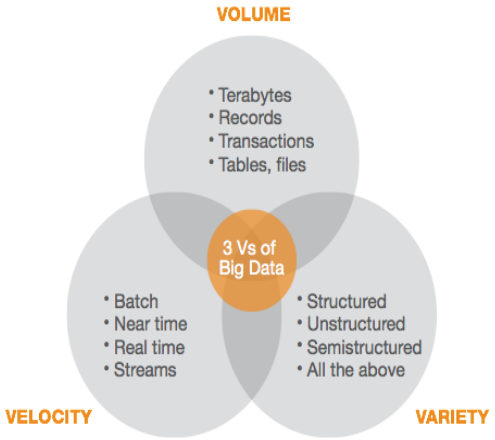
1.7 A Brief History of DB Applications

- **Data warehouse - Ch 29** 데이터가 여러곳에 저장되어 있을 때 한 곳(이거)에 불러와 동일한 스키마를 적용함
  - An archives information gathered from multiple sources, and stores it under a unified schema, at a single site to facilitate management decision making
- **Online Analytical Processing (OLAP Vs. OLTP) - Ch 29** 회사에서 중요시 여기는거
  - The operation of analysis, combine, and reporting the large scale multidimensional data dynamically.
  - Ex: Aggregate the sales of Hyundai automobiles by model, color, region, and year.
  - Ex: For each product category and each region, what were the total sales in the last quarter and how do they compare with the same quarter last year. As above, for each product category and each customer category.
- **Data mining - Ch 28** olap, 데이터마이닝 둘다 데이터를 분석하는 것인데, olap은 찾고자 하는 값이 db에 있음 / 데이터마이닝은 연산이 필요함
  - The operation of finding a knowledge and pattern which are in the bulk of data implicitly
  - Ex: What are the factors that have had the most influence over Hyundai sales?
  - Computational process of discovering patterns in large data sets involving methods at the intersection of AI, machine learning, statistics, and DBS
  - Text mining, opinion mining, web mining, recommender system 웹 구조 분석, 웹 사용 이력 분석 추천시스템

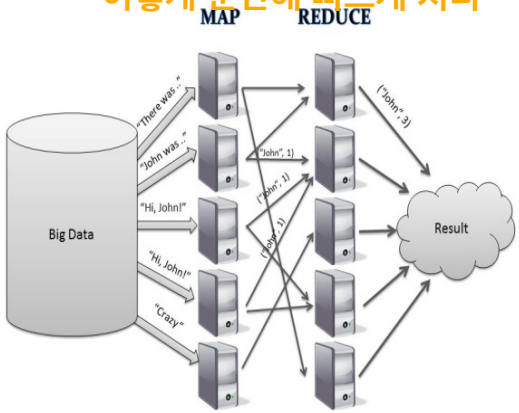


1.7 A Brief History of DB Applications

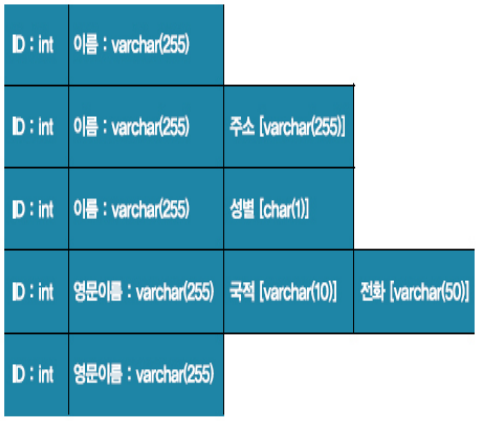
- **Big Data - Ch 25** 적어도 20테라는 되어야 빅데이터
  - A collection of data sets so large and complex that it becomes difficult to process using on-hand database management tools or traditional data processing applications
- **Hadoop - Ch 25** 빅데이터 처리하는 대표적인 툴 맵, 리듀스 자바 모듈을 사용 - 분산처리여서 빠름
  - A free, Java-based programming framework that supports the processing of big data in a distributed computing environment
  - Consists of a storage part (HDFS) and a processing part (Map-Reduce)

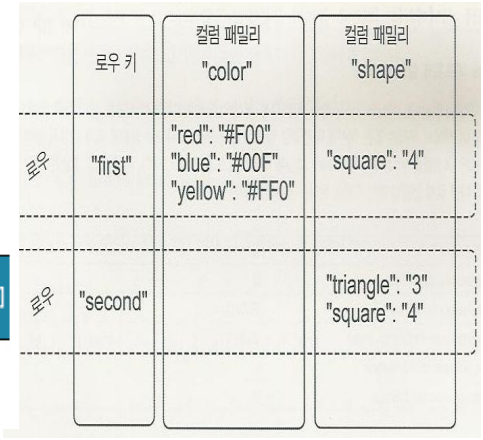


이렇게 분산해 빠르게 처리



빅데이터를 관리하기 위한 것





- **NoSQL (Not Only SQL) - Ch 24** 관계형디비와 다르게 스키마리스(스키마가 정의되지 않은 형태로 데이터를 저장)
  - Non-relational distributed database systems designed to store and retrieve a big data with a horizontal scaling and higher availability
  - Usually do not require a fixed table schema and allows a nested relation
  - Ex: HBase, Cassandra, CouchDB, MongoDB, Amazon SimpleDB, Redis

1.8 When Not to Use a DBMS

파일 시스템을 사용해야 할 때

- **The overhead costs of using a DBMS are due to the following:**
  - High initial investment in HW, SW, and training
  - The generality that a DBMS provides for defining and processing data
  - Overhead for providing security, concurrency control, recovery, and integrity functions

- **More desirable to use regular files for:**
  - Simple, well-defined DB applications not expected to change at all
  - Stringent, real-time requirements that may not be met because of DBMS overhead
  - Embedded systems with limited storage capacity
  - No multiple-user access to data

Summary

- **Database**
  - Collection of related data (recorded facts)
- **DBMS**
  - Generalized software package for implementing and maintaining a computerized database
- **Characteristics and advantages of the DB Approach**
- **Several categories of database users**
- **Database applications have evolved**