**Chap 01 – Introduction to Database Systems**

**데이터란** - A formal description of an entity, event, phenomena, or idea **that is worth recording**

**데이터베이스** – An **integrated collection** of **persistent** data, (persistent <-> transient: 일시적인)

representing **the information of interest,**

for various programs that compose the **computerized** information system of an organization.

Data are **separated** from the programs that use them. (독립적임). 자료구조는 프로그램과 독립적이지 않지만 DB는 독립적.

**DBMS**

과거에는 Application 안에 App logic과 데이터 관련 기능(Record Management, Security, Recovery, Concurrency)이 다 들어 있었음. 하지만 데이터 관련 공통적인 기능을 파일 시스템에 넣어 Application과 독립시킨게 DBMS.

**DBMS** – A collection of program modules that store, process, and manage data.

Abstraction - 얘는 회사의 창고 관리인이라고 생각할 수 있음. 따라서 App 개발자, 사용자들은 데이터의 저장 위치, 방식 등에 대해서 몰라도 됨. Abstraction 제공.

DBMS는 **set of programs to manage/access the data.**

Provides an environment that is both **convenient and efficient** to use.

Ex) DB2, Oracle, MS SQL Server, MySQL 등등..

**- File Systems**

**Part of OS.** Stores anything in disk.

과거에는 데이터베이스 어플들은 top of file systems에 개발됨.

위 방식의 단점: Data redundancy and inconsistency - 다양한 파일 포맷, 다른 파일들의 정보의 중복),

Difficulty in accessing data - 새 작업을 수행할 때마다 새 프로그램을 작성해야 함

Integrity problems - Integrity constraints (계좌 잔액 > 0)이 프로그램 코드의 일부가 됨, 새 제약 추가와 기존 제약 변경이 어려움. Integrity constraints: 무결성 제약 조건

Atomicity of updates – 작업이 실패하면 부분적으로 업데이트가 수행되어 불안정한 상태로 남아있을 수 있음.

Concurrent access by multiple users – 동시 접속이 필요한데 uncontrolled concurrent accesses는 inconsistencies를 유발할 수 있음.

데이터베이스 시스템은 위의 모든 문제들에 대해서 해결을 제시함.

**- Levels of Abstraction**

Physical Level – 어떻게 레코드가 물리적인 기기에 저장되는지를 describe.

Logical Level(Conceptual Level)– 어떻게 데이터와 데이터의 관계가 데이터베이스에 저장되는지를 describe.

View Level – 어플리케이션 프로그램은 데이터 타입에 대한 자세한 정보를 숨김.

또한 정보도 보안상의 목적으로 숨길 수 있음.

**- Data Independence**

각각 레벨의 schema를 상위 레벨의 schema정의에 영향을 끼치지 않고 변경할 수 있는 능력. 즉 레벨별로 아래의 구조를 몰라도 되는 abstraction 제공. 여기서 가장 중요한 건 physical data independence.

Physical data independence: physical level -> conceptual level

Logical data independence: conceptual level -> view level.

**- Instance and Schemas**

프로그래밍 언어의 타입과 변수와 비슷함.

**Schema** – 데이터베이스의 논리적 구조. 프로그램의 변수 타입 정보와 analogous(비슷한 특징을 가지고 있고 따라서 비교할 수 있음)함.

Physical schema: physical level의 데이터베이스 디자인

Logical schema: logical level의 데이터베이스 디자인

**Instance** – 시간이 지나면서 특정한 순간의 데이터베이스의 실제 내용(content).

변수의 값과 analogous 함.

**- Data models**

Describing을 위한 tool들의 모임. -> data, data relationships, data semantics, data constraints.

대표적으로 Entity-Relationship model, Relational model이 있음.

**- Entity-Relationship Model**

데이터를 객체와 객체들의 관계로 묘사함.

- 현실 세계에서의 ER model

**Entities (object)**: customers, accounts, back branch

**Relationships between entities**: Account A-101 is held by customer Jin, Relationship set *depositor* associates customers with accounts

- 데이터베이스 디자인에 널리 사용됨.

ER모델은 주로 relational model(storage와 processing을 위해 사용)로 변환됨. (ER 모델을 사용해서 relational model을 작성한다는 뜻인 듯)

**- Relational model**

데이터를 표의 형식(a tabular form)으로 표현.

**- Database Languages**

**- Data Definition Language (DDL)**

**Used for defining DB schema** - Ex) create table, drop column

**- Data Manipulation Language (DML)**

**Used for operating the data in the DB (DB instance)** – ex) retrieve (to find and bring back something), insert, delete, change

**- Query – 질의**

**A statement requesting the retrieval of information**

Query language: part of DML, 가끔은 query language == DML.

**- SQL**

The most widely used language

**- Database Users**

유저들은 시스템과 상호작용하는 방식에 따라 구분됨.

**Application Programmers – 시스템과 DML call을 통해 상호작용**.

Sophisticated Users – 쿼리 언어로 form requests

Specialized Users – specialized 된 데이터베이스 어플리케이션을 작성. 앱은 기존 데이터 프로세싱 프레임워크와 맞지 않음.

**Naïve users** – 미리 작성되어진 permanent 어플리케이션 프로그램을 invoke 함.

**- Database Administrator**

데이터베이스 시스템의 모든 활동을 coordinate 함. DBA는 기업의 정보 리소스와 필요에 대해 좋은 이해를 가지고 있어야 함.

DBA의 의무는 다음을 포함함.

**- Schema definition**

**- Monitoring performance and responding to changes in requirements.**

- 그 외 저장소 구조와 접근 방법 정의, 스키마와 물리적 조직 변경, 유저에게 데이터베이스 액세스 권한 주기, integrity constraints 정의, 유저들과 liaison으로 역할.

**- Overall System Structure**

데이터는 DB에 저장됨 -> Storage Manager -> Query Processor -> 유저들

자세한건 pdf 참조.