1. Introduction :
Database System Concepts

What is Database?

- **Data**
  - Structured, Semi-structured, Unstructured

- **Database**
  - A collection of relevant data
  - Huge Volume
  - Disk Resident
  - Operational: Retrieve, Insert, Delete, Update
  - Shared by multiple users

Database Example: University

- **Entities**:
  - STUDENTs
  - COURSEs
  - SECTIONs
  - CLASSROOMs
  - PROFESSORs

  . . . . . .

- **Relationships** :
  - STUDENTs have *grade-report* with SECTIONs
  - COURSEs have *prerequisite* COURSEs
  - STUDENTs have *advisor* among PROFESSORs
  - COURSEs *are reserved by* CLASSROOMs

  . . . . . .

Database Example: University

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

Another Database Example : Movie

- **Entities**:
  - ACTORs
  - MOVIEs
  - DIRECTORs
  - PRODUCERs

- **Relationships :**
  - ACTORs *play* MOVIEs
  - PRODUCERs *make* MOVIEs
  - ACTORs *mary* ACTORs
  - DIRECTORs *direct* MOVIEs
  - ACTORS *contract* PRODUCERS

What is DBMS?

- **Database Management System (DBMS)**
  : A software package to store and manage databases.

- **Examples of DBMS**
  - ✓ Oracle
  - ✓ MySQL (Oracle) : Open source
  - ✓ DB2 (IBM)
  - ✓ SQL Server (MS)
  - ✓ Sybase (SAP)

- **Database System**
  : Database + DBMS

Simplified Database System

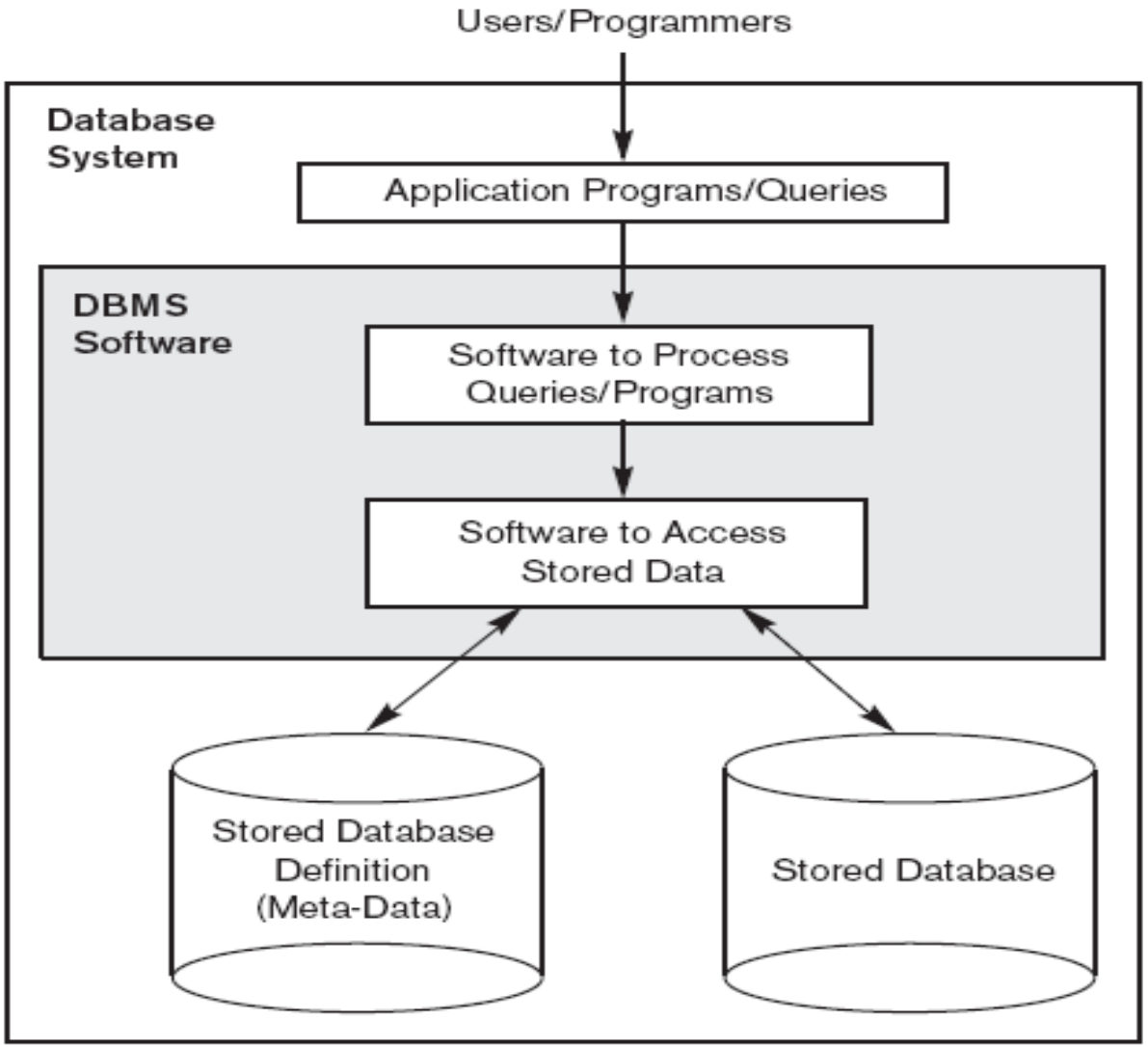


Figure 1.1
A simplified database system environment.

Without DBMS : File Systems

- Controlled by OS
- Data is stored as records in traditional regular files
- Records have a simple structure and fixed number of fields
- File structures are embedded in application programs
- No query language for retrieving/updating data
- No special programs manage and control data

Why DBMS? : Problems of File Systems (1)

- Difficulty for **Data Abstraction**
  - Users must specify physical structures of database on their application programs.
  - It is hard to hide complex physical structures to users.

Why DBMS? : Problems of File Systems (1)

- **Data/Program Dependency** Problem
  - User application program is dependent on physical structures of database stored on disk.
  - If physical structures are changed, then its application program also must be changed; Thus, Need to rewrite new program.

Why DBMS? : Problems of File Systems (2)

- **Data Redundancy** Problem
  - Duplicates of same information may exist on files
  - Storage Waste, **Data Inconsistency**

Why DBMS? : Problems of File Systems (2)

- Difficulty for **Data Integrity**
  - DB must obey and check integrity constraints .
  - Example:
    - Student IDs must be distinct.
    - Bank account balance > $100.
    - Every student takes 3 ~ 6 courses per semester.

Why DBMS? : Problems of File Systems (3)

- Difficulty for **Concurrent Access**
  - Need to allow for many users to access database at the same time
  - Uncontrolled accesses can lead to inconsistencies;
  - Example : Money transfer from account A to B and withdraw from B.
  - Concurrency control must be needed.

Why DBMS? : Problems of File Systems (3)

- Difficulty for **Data Recovery**
  - System failures may leave database in inconsistencies.
  - Example : Failure during money transfer from account A to B.
  - Must restore data to correct state.

Why DBMS? : Problems of File Systems (3)

- Difficulty for **Query Processing**
  - Need optimal query **optimization** for fast query processing.
  - Uncontrolled query processing can lead to incredibly slow response time.

Why DBMS? : Problems of File Systems (3)

- Difficulty for **Data Security**
  - Need to prohibit some users to access to sensitive data.
  - Need to classify access authorizations retrieval and updates.

What DBMS Provide? (1)

- Data abstraction (Insulation of program and data)
- Create and manage database on a secondary storage.
- Query language for retrieving/updating database
- Sharing of data among multiple users
- Transaction Control : Concurrent Processing

What DBMS Provide? (2)

- Restricting unauthorized access to data.
- Recovery from system failures.
- Query Optimizer for efficient query processing
- Multiple Interfaces to different classes of users.

Storing Database

- Typical Database Files
  - Heap File
  - Sequential File
  - Indexing File : B+ Tree
  - Hashing File

Schema and Instance

- Database **Schema (= Intension)**
  - Describe structure and constraints of DB
  - Specified at DB design
  - Rarely changed
- Database **Instance (= Extension)**
  - Contents of DB (= actual data)
  - Dynamically changed (by update, insert, . . .)
- Database = **Schema + Instance**

Schema : Example

Figure 2.1
Schema diagram for the database in Figure 1.2.

STUDENT

Name	Student_number	Class	Major

COURSE

Course_name	Course_number	Credit_hours	Department

PREREQUISITE

Course_number	Prerequisite_number

SECTION

Section_identifier	Course_number	Semester	Year	Instructor

GRADE_REPORT

Student_number	Section_identifier	Grade

[6]Schema changes are usually needed as the requirements of the database applications change. Newer database systems include operations for allowing schema changes, although the schema change process is more involved than simple database updates.

[7]It is customary in database parlance to use *schemas* as the plural for *schema*, even though *schemata* is the proper plural form. The word *scheme* is also sometimes used to refer to a schema.

Schema and Instances : Example

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

DBMS Catalog (Metadata)

- Database system contains **database**, and also **description of database**;

  i.e., table name, field name, types, length, . . .

- This description is stored in **DBMS catalog,** also called **meta-data** (**data dictionary**).

- DBMS catalog is referred by DBMS and database users who need information about database structure.

Example : DBMS Catalog

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.

Database Query Language

- Database Query Language consist of two parts:

(1) **DDL**(Data Definition Language)
  - Define schema, table, and views
  - CREATE, DROP, ALTER

(2) **DML**(Data Manipulation Language)
  - Retrieve and modify database instances
  - SELECT, INSERT, DELETE, UPDATE

- SQL is the most widely used query language

ANSI : 3-Schema Architecture

- Data in DBMS is described by schemas at **3** levels:

  - **Physical Schema**
    - Describes internal (storage) structure of database.
      (How data is **stored** physically?)

  - **Conceptual schema**
    - Describes conceptual structure of database for user groups.
      (What is **meaning** of (whole) data?)

  - **External schemas (= Views)**
    - Describes part of database for particular user needs.
      (What is **use** of (partial) data?)

3-Schema Architecture : Example

- **External Schema (= View):**
  - High_Age_Customer(cid: string, cname: string)
  - Purchase_Info (cname: string, pname: string, date: string)

- **Conceptual schema:**
  - Customers (cid: string, cname: string, age: integer)
  - Products (pid: string, pname: string, price: integer)
  - Purchase (cid: string, pid: string, date: string)

- **Physical schema:**
  - Customers stored as sorted by cid
  - Products stored as Index on pid.

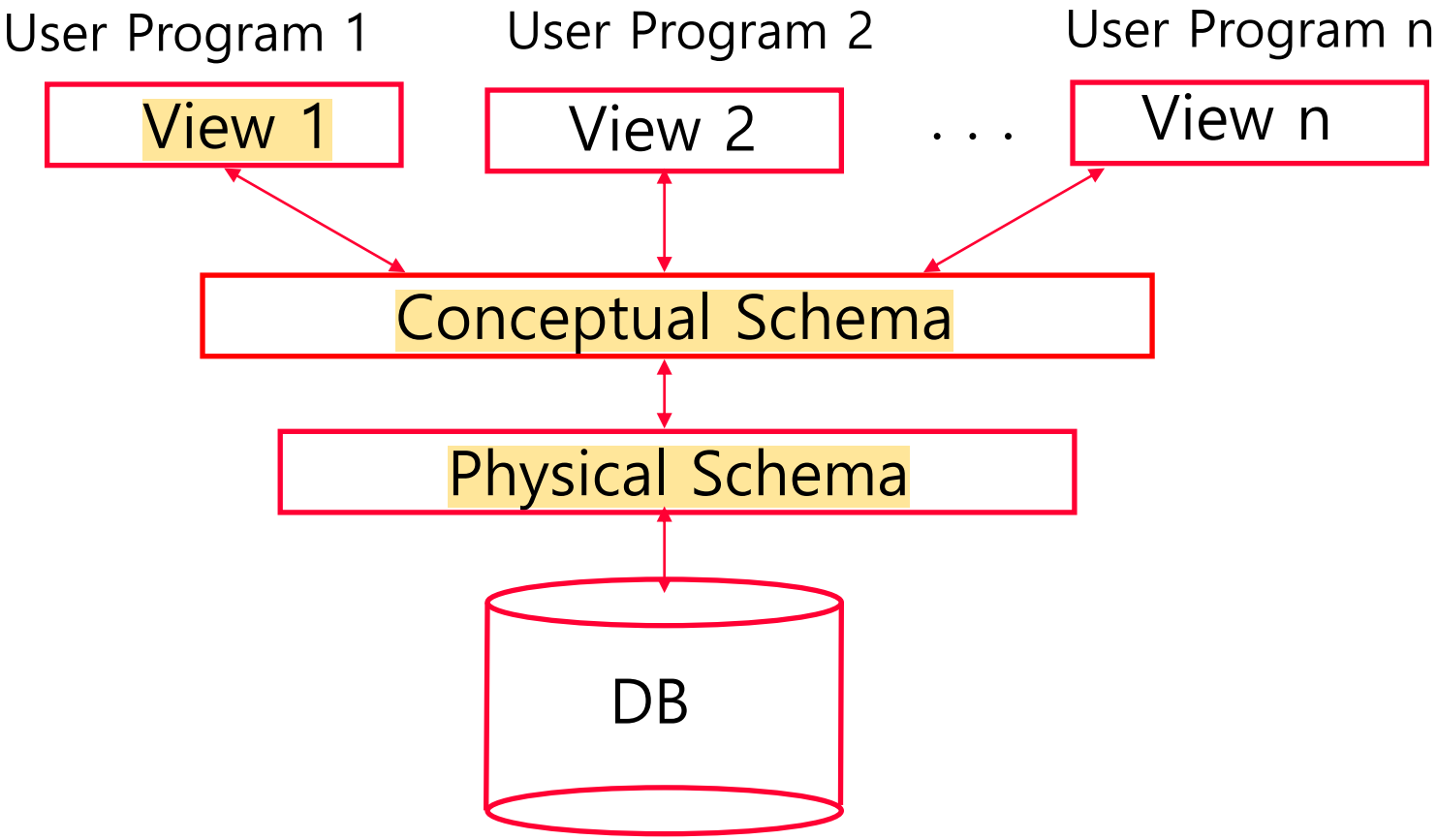
3-Schema Architecture : Example

- **External Schema (= View):**
  - Pre_Course(cid: string, cname: string, pre_cid: string)
  - Course_Info (sname: string, cname: string, grade: string)

- **Conceptual schema:**
  - Student (sid: string, sname: string, age: integer)
  - Course (cid: string, cname: string, credit: integer)
  - Enrolled (sid: string, cid: string, grade: string)

- **Physical schema:**
  - Students stored as ordered by sid
  - Courses stored as Index on cid.

3-Schema Architecture



- Schemas are defined using DDL; Data is modified/queried using DML.

3-Schema Architecture

- Separate application programs and physical database
- Support multiple user views.
- Higher level schema hides lower level schemas
  - Conceptual schema hides physical schema
  - External schema hides conceptual schema
- Mappings among schema levels are needed to transform user requests and data.
  - External/Conceptual mapping
  - Conceptual/Physical mapping

Data Independence

- **Logical Independence**
  - Capacity to change conceptual schema without having to change external schemas
  - Thus, users can use existing views regardless of change of conceptual schema.

- **Physical Independence**
  - Capacity to change physical schema without having to change conceptual schemas
  - Thus, users can use existing conceptual schema regardless of change of physical schema.

Class of Database Users

- Database Administrator (DBA)
- Database Designer
- Application Programmer
- Casual end user
- Naïve end user