CS275 – Intro to Databases

How does a DBMS work? - *Chap. 1-2*

File Systems vs. DBMS

- We have to write special programs for queries
- We have to protect data from inconsistencies
- We have to handle crashes
- We have to control access w/ passwords.

Characteristics of database approach

- Self describing nature

-metadata(data type, constraint) stored in catalog.

-catalog is used by DBMS and other users as well.

- DBMS can support multiple databases but file based system based can support one database using one application.

Characteristics of database approach

- Insulation between programs, data and data abstraction.
- Change in file structure impacts all programs accessing the file.
- Program data independence.

Characteristics of database approach

- Support multiple **views** of data.

  - Virtual data(Derived data)
- Sharing of data and multiuser transaction processing.

  -OLTP(online transaction processing)

An Example

- UNIVERSITY database
  - Information concerning students, courses, and grades in a university environment
- **Data records**
  - STUDENT
  - COURSE
  - SECTION
  - GRADE_REPORT
  - PREREQUISITE

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.

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.

Why is a DBMS so important?

- A DBMS is **software** designed to assist in maintaining and utilizing large collection of data.
  - A large amount of data,
  - Concurrent access by many users,
  - Fast access,
  - Consistent data update,
  - Role-based security,
  - Robust against hardware failures and OS crashes.

Why and When Do We Need DBMS?

- Advantages of DBMS
  - Data independence
    - Hide data representation
    - Hide data storage

Why and When Do We Need DBMS?

- Advantages of DBMS
  - Efficient data access
    - High-quality data compression schemes,
    - Fast data retrieval and search algorithms.

Why and When Do We Need DBMS?

- Advantages of DBMS
  - Data integrity and security
    - Prevent invalid queries from being executed,
    - Access control can be enforced.

Why and When Do We Need DBMS?

- Advantages of DBMS
  - Centralized data administration
    - The DBA can optimize the organization of the data to facilitate its uses.

Why and When Do We Need DBMS?

- Advantages of DBMS
  - Concurrent access
  - Crash recovery
  - Reduced application development time
  - Providing **persistent** storage for program objects

How Does DBMS Support Concurrent Access & Crashes?

- Transaction
  - Any one execution of a user program in DBMS.
    - A single command
    - Group several commands/queries into one transaction
  - Locking protocol
    - Shared vs. Exclusive
    - Row level vs. Table level
    - Read vs. Write
  - Log
    - Finalizes transaction
    - Checkpoints

Overhead costs of DBMS

- High initial investment in hardware, software and training.
- The generality.
- Overhead for providing
    - security
    - concurrency
    - recovery

When Not to Use a DBMS?

- More desirable to use regular files for:
  - Simple, well-defined database 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

Actors on the Scene

- **Database administrators (DBA)** are responsible for:
  - Authorizing access to the database
  - Coordinating and monitoring its use
  - Acquiring software and hardware resources
- **Database designers** are responsible for:
  - Identifying the data to be stored
  - Choosing appropriate structures to represent and store this data

Actors on the Scene (cont'd.)

- **End users**
  - People whose jobs require access to the database
  - Types
    - **Casual end users**
    - **Naive** or **parametric end users**
    - **Sophisticated end users**
    - **Standalone users**

Actors on the Scene (cont'd.)

- **System analysts**
  - Determine requirements of end users
- **Application programmers**
  - Implement these specifications as programs

Workers behind the Scene

- **DBMS system designers and implementers**
  - Design and implement the DBMS modules and interfaces as a software package
- **Tool developers**
  - Design and implement **tools**
- **Operators and maintenance personnel**
  - Responsible for running and maintenance of hardware and software environment for database system

Categories of Data Models

- **High-level** or **conceptual data models**
  - Close to the way many users perceive data
- **Low-level** or **physical data models**
  - Describe the details of how data is stored on computer storage media
- **Representational data models**
  - Easily understood by end users
  - Also similar to how data organized in computer storage

Categories of Data Models

- **Entity**
  - Represents a real-world object or concept
- **Attribute**
  - Represents some property of interest
  - Further describes an entity
- **Relationship** among two or more entities
  - Represents an association among the entities
  - **Entity-Relationship model**

Categories of Data Models (cont'd.)

- **Relational data model**
  - Used most frequently in traditional commercial DBMSs
- **Object data model**
  - New family of higher-level implementation data models
  - Closer to conceptual data models

How Is Data Represented?

- A data model
  - A collection of high-level data descriptions,
  - Hides low-level storage details.
- A semantic data model
  - More abstract,
  - Serves as a startup point for the design,
  - Farther away from the physical storage than a data model.

How Is Data Represented?

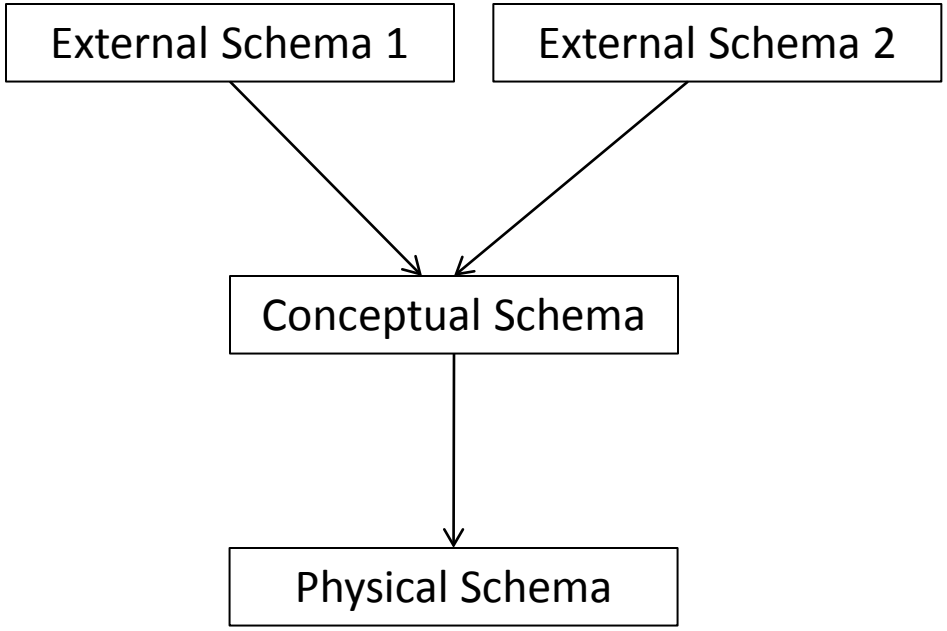
- The relational model
  - Relation (records)
  - Schema
    - Data descriptions, such as name of the relation and individual field.

    Students(*sid*: string, *name*: string, *login*: string,

    age: integer, *gpa*: real)
  - Integrity Constraints
  - Database state or snapshot

How Is Data Represented?

- Levels of Abstraction



How Is Data Represented?

- Conceptual Schema (logical schema)
  - Data Model/Relationships

Students(*sid*: string, *name*: string, *login*: string,
age: integer, *gpa*: real)
Faculty(*fid*: string, *fname*: string, *sal*: real)
Courses(*cid*: string, *cname*: string, *credits*: integer)
Enrolled(*sid*: string, *cid*: string, *grade*: string)
Teaches(*fid*: string, *cid*: string)

How Is Data Represented?

- Physical Schema
  - Data Storage
  - Based on Access

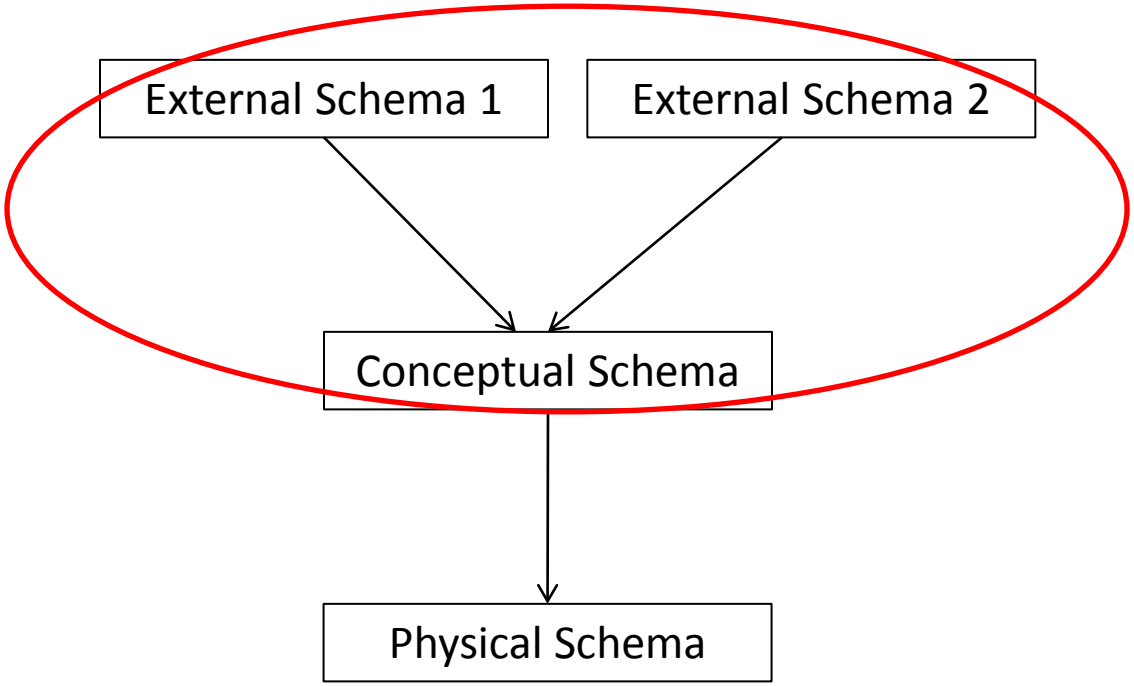
How Is Data Represented?

- External Schema
  - Different Views
  - Defined by end user requirements

Courseinfo(*cid*: string, *fname*: string, *enrollment*: integer)

How Is Data Represented?

- Data Independence
  - Logical data independence



How Is Data Represented?

- Data Independence
  - Physical data independence