CS 440: Database Management Systems

Finish Chapter 1

- Workers behind the Scene
- Advantages of Using the DBMS Approach
- A Brief History of Database Applications
- When Not to Use a DBMS

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
Advantages of Using the DBMS Approach

- Controlling redundancy
  - Data normalization
  - Denormalization
    - Sometimes necessary to use controlled redundancy to improve the performance of queries
- Restricting unauthorized access
  - Security and authorization subsystem
  - Privileged software

Advantages of Using the DBMS Approach (cont’d.)

- Providing persistent storage for program objects
  - Complex object in C++ can be stored permanently in an object-oriented DBMS
  - Impedance mismatch problem
    - Object-oriented database systems typically offer data structure compatibility

Advantages of Using the DBMS Approach (cont’d.)

- Providing storage structures and search techniques for efficient query processing
  - Indexes
  - Buffering and caching
  - Query processing and optimization
Advantages of Using the DBMS Approach (cont’d.)

- Providing backup and recovery
  - Backup and recovery subsystem of the DBMS is responsible for recovery
- Providing multiple user interfaces
  - Graphical user interfaces (GUIs)
- Representing complex relationships among data
  - May include numerous varieties of data that are interrelated in many ways

Advantages of Using the DBMS Approach (cont’d.)

- Enforcing integrity constraints
  - Referential integrity constraint
    - Every section record must be related to a course record
  - Key or uniqueness constraint
    - Every course record must have a unique value for Course_number
- Business rules
  - Inherent rules of the data model

Advantages of Using the DBMS Approach (cont’d.)

- Permitting inferencing and actions using rules
  - Deductive database systems
    - Provide capabilities for defining deduction rules
    - Inferencing new information from the stored database facts
  - Trigger
    - Rule activated by updates to the table
  - Stored procedures
    - More involved procedures to enforce rules
Advantages of Using the DBMS Approach (cont’d.)

- Additional implications of using the database approach
  - Reduced application development time
  - Flexibility
  - Availability of up-to-date information
  - Economies of scale

A Brief History of Database Applications

- Early database applications using hierarchical and network systems
  - Large numbers of records of similar structure
- Providing data abstraction and application flexibility with relational databases
  - Separates physical storage of data from its conceptual representation
  - Provides a mathematical foundation for data representation and querying

A Brief History of Database Applications (cont’d.)

- Object-oriented applications and the need for more complex databases
  - Used in specialized applications: engineering design, multimedia publishing, and manufacturing systems
- Interchanging data on the Web for e-commerce using XML
  - Extended markup language (XML) primary standard for interchanging data among various types of databases and Web pages
A Brief History of Database Applications (cont’d.)

- Extending database capabilities for new applications
  - Extensions to better support specialized requirements for applications
  - Enterprise resource planning (ERP)
  - Customer relationship management (CRM)

- Databases versus information retrieval
  - Information retrieval (IR)
    - Deals with books, manuscripts, and various forms of library-based articles

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

Chapter 2 Outline

- Data Models, Schemas, and Instances
- Three-Schema Architecture and Data Independence
- Database Languages and Interfaces
- The Database System Environment
- Centralized and Client/Server Architectures for DBMSs
- Classification of Database Management Systems
Database System Concepts and Architecture

- Basic client/server DBMS architecture
  - Client module
  - Server module

Data Models, Schemas, and Instances

- Data abstraction
  - Suppression of details of data organization and storage
  - Highlighting of the essential features for an improved understanding of data

Data Models, Schemas, and Instances (cont’d.)

- Data model
  - Collection of concepts that describe the structure of a database
  - Provides means to achieve data abstraction
  - Basic operations
    - Specify retrievals and updates on the database
  - Dynamic aspect or behavior of a database application
    - Allows the database designer to specify a set of valid operations allowed on database objects
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 (cont’d.)

- **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
Categories of Data Models (cont’d.)

- Physical data models
  - Describe how data is stored as files in the computer
  - Access path
    - Structure that makes the search for particular database records efficient
  - Index
    - Example of an access path
    - Allows direct access to data using an index term or a keyword

Schemas, Instances, and Database State

- Database schema
  - Description of a database
- Schema diagram
  - Displays selected aspects of schema
- Schema construct
  - Each object in the schema
- Database state or snapshot
  - Data in database at a particular moment in time

Three-Schema Architecture and Data Independence

- Internal level
  - Describes physical storage structure of the database
- Conceptual level
  - Describes structure of the whole database for a community of users
- External or view level
  - Describes part of the database that a particular user group is interested in
Three-Schema Architecture and Data Independence (cont'd.)

Conceptual level

- Conceptual Schema (logical schema)
  - For our 440 class: Specify data types

  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)

Internal level

- Physical Schema
  - Data Storage
  - Based on Access
External or view level

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

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

Classification of Database Management Systems

- Data model
  - Relational
  - Object
  - Hierarchical and network (legacy)
  - Native XML DBMS

- Number of users
  - Single-user
  - Multiuser
Classification of Database Management Systems (cont'd.)

- Number of sites
  - Centralized vs. Distributed
    - Homogeneous vs. Heterogeneous
- Cost
  - Open source
  - Different types of licensing
- Types of access path options
- General or special-purpose