Introduction to Database Management System

Linda Wu

(CMPT 354 • 2004-2)

Topics

- What is DBMS
- DBMS types
- Files system vs. DBMS
- Advantages of DBMS
- Data model
- Levels of abstraction
- Transaction management
- DBMS structure

What Is a DBMS?

- Database
  - A very large, integrated collection of data
  - Models real-world enterprise
    - Entities (e.g., students, courses)
    - Relationships (e.g., Madonna is taking cmpt354)
- Database Management System (DBMS)
  - A software package designed to store and manage databases

DBMS Types

- Hierarchical DBMS
- Network DBMS
- Relational DBMS
- Object-Oriented DBMS
- XML DBMS

Files vs. DBMS

- Application must stage large datasets between main memory and secondary storage
- Special programs for different queries
- Must protect data from inconsistency due to multiple concurrent users
- Crash recovery
- Security and access control

Advantages of DBMS

- Data independence and efficient access
- Data integrity and security
- Uniform data administration
- Concurrent access, recovery from crashes
- Reduced application development time

Why Study Databases?

- Shift from computation to information
- Datasets increasing in diversity and volume
  - Digital libraries, interactive video, Human Genome project, EOS project
  - … need for DBMS exploding
- DBMS encompasses most of CS
  - OS, languages, theory, AI, multimedia, logic

Data Model

- Data model
  - A collection of concepts for describing data
- Schema
  - A description of a particular collection of data, using the given data model and its data definition language
- Relational model
  - The most widely used model today
  - Central concept: relation, or, a table with rows and columns
  - Every relation has a schema, which describes the columns or fields

Data Model (Cont.)

- Schema

  Students (*sid*: string, *name*: string, *login*: string, *age*: integer, *gpa*: real)

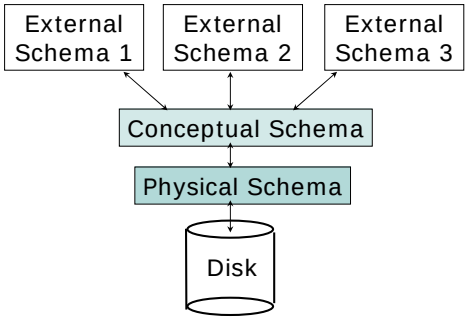
- An instance of the Students relationship

sid	name	login	age	gpa
00001	Jones	jones@cs	18	3.4
00002	Smith	smith@ee	19	3.2
00003	Smith	smith@math	18	3.8
00004	Mary	mary@music	14	1.8
00005	Gary	gary@biz	12	2.0

Levels of Abstraction in DBMS

- Single conceptual (logical) schema
  - Define logical structure (all relations stored in the relational DBMS)
- Single physical schema
  - Describes the files and indexes used, i.e., storage details
- Many external schemas
  - Describe how users see the data
  - Each external schema consists of views and relations from the conceptual schema

Levels of Abstraction (Cont.)



- *External and conceptual schemas are defined using DDL*
- *Data is modified / queried using DML*

Levels of Abstraction (Cont.)

- View
  - A view is conceptually a relation
  - Views can be computed from the relations in the conceptual schema
  - The records in a view are NOT stored in DBMS
  - User can query the records in a view

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

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

Data Independence

- Applications are insulated from how data is structured and stored
- Logical data independence
  - Protection from changes in logical structure of data
- Physical data independence
  - Protection from changes in physical structure of data

** One of the most important benefits of using a DBMS!*

Concurrency Control

- Concurrent execution of user programs is essential for good DBMS performance
  - Disk accesses are frequent and relatively slow
  - It is important to keep CPU humming by working on several user programs concurrently
- Interleaving actions of different user programs can lead to inconsistency
  - A database is typically shared by a large number of users
  - DBMS ensures consistency: users can pretend they are using a single-user system

Transaction

- An atomic sequence of database actions (reads/writes)
- Each transaction, executed completely, must leave the DB in a consistent state if DB is consistent when the transaction begins
  - Users can specify some simple integrity constraints on the data, and the DBMS will enforce these constraints
  - DBMS does not really understand the semantics of the data
  - Ensuring that a transaction (run alone) preserves consistency is user's responsibility!

Scheduling Concurrent Transactions

- DBMS ensures that execution of {T1, … , Tn} is equivalent to some serial execution T1' … Tn'
  - Before reading/writing an object, a transaction requests a lock on the object, and waits till the DBMS gives it the lock
  - All locks are released at the end of the transaction

  (Strict 2PL locking protocol)

Atomicity & Log

- DBMS ensures atomicity (all-or-nothing property) even if system crashes in the middle of a transaction
- Idea: keep a log (history) of all actions carried out by the DBMS while executing a set of transactions
  - Before a change is made to the database, the corresponding log entry is forced to a safe location. (Write-Ahead Log, or WAL protocol)
  - After a crash, the effects of partially executed transactions are undone using the log
- Actions recorded in the log
  - Ti writes an object: both old and new value
  - Ti commits/aborts: a log record indicating this action

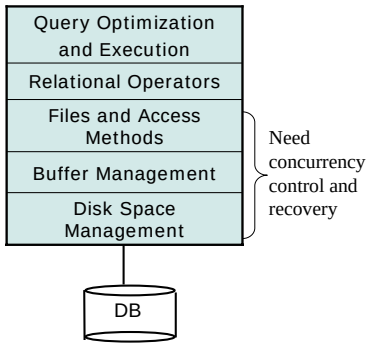
Database Users

- End users and DBMS vendors
- DB application programmers
  - E.g. smart webmasters
- Database administrator (DBA)
  - Design of logical /physical schemas
  - Security and authorization
  - Data availability and crash recovery
  - Database tuning

Must understand how a DBMS works!

Structure of a DBMS

- A typical DBMS has a layered architecture
- The figure does not show the concurrency control and recovery components
- This is one of several possible architectures; each system has its own variations



Summary

- DBMS is used to maintain, query large datasets
- Benefits: recovery from system crashes, concurrent access, quick application development, data integrity and security
- Levels of abstraction give data independence
- A DBMS typically has a layered architecture
- DBAs hold responsible jobs and are well-paid!
- DBMS R&D is one of the broadest, most exciting areas in Computer Science