Database Management Systems

Chapter 1

Instructor: Oliver Schulte
oschulte@cs.sfu.ca
Outline

- What is a database?
- Why use a database?
- Why study databases?
- Database Overview
What is a database?

- A database (DB) is a very large, integrated, permanent collection of data.

- Models real-world
  - Entities (e.g., students, courses)
  - Relationships (e.g., Madonna is taking CMPT354).

- Example databases:
  - Customer Transactions
  - Human Genome
  - Online Bookstore
  - Mondial is a sample DB for teaching purposes.
What Is a DBMS?

- A **Database Management System (DBMS)** is a software package designed to store and manage databases.
This course is important for...

- End users of DBS
- DB application programmers
- Database administrators (DBA)
- DBMS vendors

\[ \text{Must understand how a DBMS works!} \]
Why Use Databases?
The Increasing Flood of Data

Human Genome

Customer Transactions

- As of 2004, Walmart data-warehouse was 500 terabytes in size.
- In 2007, it was over 1 petabyte (1 trillion gigabytes)
- Sources:

Online Bookstore

- Amazon has roughly a bazillion products, give or take a couple zillion.

- The human genome contains 3.2 billion chemical nucleotide base pairs (A, C, T, and G).
- Largest known human gene is dystrophin at 2.4 million base pairs.
- Functions are unknown for more than 50% of discovered genes.
- Source: http://www.ornl.gov/sci/techresources/Human_Genome/project/journals/insights.shtml
Amazon: Website, database or application?
Data Storage Without DBMS

File 1

File 2

... File m

Application program 1

Application program 2

... Application program n

reads / writes
Data Storage With DBMS

File 1

File 2

File m

Application program 1

Application program 2

Application program n

reads / writes
Files vs. DBMS

- Application must stage large datasets between main memory and secondary storage (e.g., buffering, page-oriented access, 32-bit addressing, etc.)
- Special code for different queries
- Must protect data from inconsistency due to multiple concurrent users
- Crash recovery
- Security and access control
- But note NoSQL 'movement'.
Why Use a DBMS?

Summary

- Data independence (abstract view of data) and efficient access.
- Reduced application development time.
- Data integrity (enforce constraints) and security.
- Uniform (central) data administration.
- Concurrent access, recovery from crashes.
Why Study Databases??

- Shift from computation to information.
- Datasets increasing in diversity and volume.
  - Digital libraries, interactive video, Human Genome project
  - ... need for DBMS exploding
- DBMS intersects with most of CS
  - OS, languages, theory, AI, multimedia, logic
Database Overview
Data Models

- A **data model** is a collection of concepts for describing data.
- A **schema** is a description of a particular collection of data, using the given data model and its data definition language.
- The **relational model of data** is the most widely used model today.
  - Main concept: **relation**, basically a table with rows and columns.
  - Every relation has a **schema**, which describes the columns, or fields.
Levels of Abstraction

- Many views, single conceptual (logical) schema and physical schema.
  - Views describe how users see the data.
  - Conceptual schema defines logical structure
  - Physical schema describes the files and indexes used.

Schemas are defined using DDL = data description language.
Data is modified/queried using DML = data manipulation language.
Example: University Database

- **Conceptual schema:**
  - `Students(sid: string, name: string, login: string, age: integer, gpa: real)`
  - `Courses(cid: string, cname: string, credits: integer)`
  - `Enrolled(sid: string, cid: string, grade: string)`

- **Physical schema:**
  - Relations stored as unordered files.
  - Index on first column of Students.

- **External Schema (View):**
  - `Course_info(cid: string, enrollment: integer)`
Data Independence

- Applications insulated from how data is structured and stored.
- **Logical data independence**: Protection from changes in logical structure of data (e.g., adding new fields).
- **Physical data independence**: Protection from changes in physical structure of data (e.g., sorting, indexing, compressing).

One of the most important benefits of using a DBMS!
Database Transaction Processing

- **ACID:**
  - Atomicity.
  - Consistency.
  - Isolation.
  - Durability.
Transaction: An Execution of a DB Program

- Key concept is **transaction**, which is 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.
Concurrency Control
(Isolation)

- Concurrent execution of user programs is essential for good DBMS performance.
  - Because disk accesses are frequent, and relatively slow, it is important to keep the cpu humming by working on several user programs concurrently.

- Interleaving actions of different user programs can lead to inconsistency: e.g., check is cleared while account balance is being computed.

- DBMS ensures such problems don’t arise: users can pretend they are using a single-user system.
Scheduling Concurrent Transactions

- DBMS ensures that execution of \{T1, \ldots, Tn\} is equivalent to some **serial** execution T1\textsubscript{A} ... Tn\textsubscript{A}.
  - 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.)**
Ensuring Atomicity

- DBMS ensures *atomicity* (all-or-nothing property) even if system crashes in the middle of a Xact.

- **Idea:** Keep a *log* (history) of all actions carried out by the DBMS while executing a set of Xacts:
  - 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.
The Log

- The following actions are recorded in the log:
  - *Ti writes an object*: the old value and the new value.
    - Log record must go to disk **before** the changed page!
  - *Ti commits/aborts*: a log record indicating this action.

- Log records chained together by Xact id, so it's easy to undo a specific Xact (e.g., to resolve a deadlock).

- Log is often *duplexed* and *archived* on 'stable' storage.

- All log related activities (and in fact, all CC related activities such as lock/unlock, dealing...
Exercise 1.6

Scrooge McNugget wants to store information (names, addresses, descriptions of embarrassing moments, etc.) about the many ducks on his payroll. Not surprisingly, he wants to buy one with the fewest possible features, and he plans to run it as a stand-alone application on his PC clone. Scrooges does not plan to share his list with anyone. Indicate which of the following DBMS features Scrooge should pay for, and why (or why not).

1. A security facility
2. Access Control
3. Concurrency Control
4. Crash recovery
5. A query language.
Structure of a DBMS

- A typical DBMS has a layered architecture.
- This is one of several possible architectures; each system has its own variations.

Query Optimization and Execution
Relational Operators
Files and Access Methods
Buffer Management
Disk Space Management
Exercise 1.7

Which of the following plays an important role in representing information about the real world in a database?

1. The data definition language.
2. The data manipulation language.
3. The buffer manager.
4. The data model.
Summary

- DBMS used to maintain, query large datasets.
- Benefits include 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!