ICOM 6005 – Database Management Systems Design

Dr. Manuel Rodríguez Martínez

Electrical and Computer Engineering Department

Lecture 2 – August 23, 2001

Readings

- Read Chapter 1 of text book

Why do we Databases?

- Suppose that you own a bank.
- You need to keep track of information about your costumers and their accounts.
  - Customer personal information
  - Account information
  - Receipts of the transactions performed
- There are several possibilities to accomplish this
  - Do nothing, throw papers in a box.
  - Keep records sorted alphabetically
  - Buy a custom program to manage your information.
  - Get a database system and get a database application to manage your information.

Option 1: Do nothing …

- In this case, you just write all the information about costumers and their account in pieces of papers and then you throw them in a box.
- Advantages
  - Cheap
- Disadvantages
  - Almost impossible to find records
  - Very slow to gather information about your business

Option 2: Keep records sorted

- Rather than throwing your paper in a box, you buy a cabinet and keep all records sorted, perhaps alphabetically on the costumer name.
- Advantages
  - Cheap
  - Easier to use
- Disadvantages
  - Too slow to find aggregated information about the bank (e.g. Which are the 10 most active accounts?).
  - Still slow, since human has to find the records.
  - Cabinets take too much space

Option 3: Buy a custom program

- Buy a program that uses the file system in a computer to store all the data associated with your bank.
- Advantages:
  - Fast
  - Takes up little space
- Disadvantages:
  - Expensive
  - Difficult to add new features
  - Reliance on expertise of programmers
    - Re-inventing the wheel

Option 4: Buy a Database System

- Buy a Database System to organize and analyze all the data items.
- Advantages:
  - Extensibility
  - Reliability
  - Recovery from crash
  - Advanced data analysis tools
  - Take little space
- Disadvantages:
  - Expensive
  - Rely on expertise of programmers

Database and DBMS

- A database is a collection of data that describes the inner structure or inner workings of an enterprise.
- A Database Management System (DBMS) is a software system used to maintain the data stored in one or more databases.
- Databases is also meant to convey the area of Computer Science devoted to the study of hardware, software, algorithms, data structures and other techniques required to design and implement a DBMS or an application that use a DBMS.

Top Database Groups and Products

- Academia:
  - 1) Stanford University, 2) University of Wisconsin, Madison, 3) University of California, Berkeley, 4) University of Maryland, College Park, 5) University of Washington.
- Research Labs:
  - 1) IBM Almaden Research Lab, 2) AT&T Research Labs, 3) Microsoft Research Lab, 4) Lucent Technologies.
- Database Products:
  - 1) Oracle, 2) IBM DB2, 3) MS Access, 4) Sybase, 5) MS SQL Server, 6) Informix.

Relational Model

- Proposed by Codd from IBM in 1970
- Models data as tables (relations) consisting of columns (attributes).
- Each record (tuples) is a row in the table.
- Very rich set of mathematical operations
  - Elegant model
- Schema of the data – description of the tables and columns
  - Table name
  - Attribute names
  - Attribute types

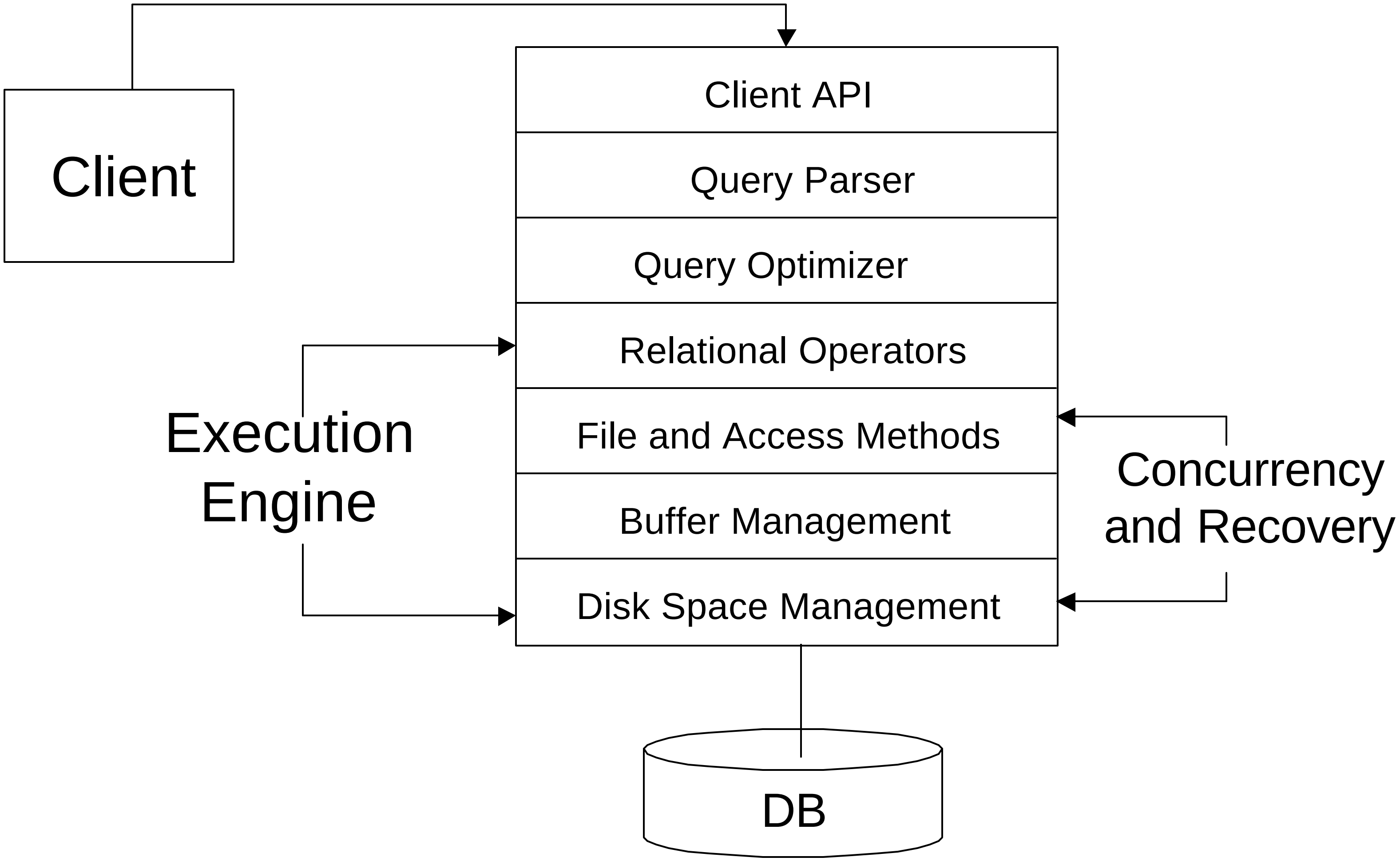
Modeling data by levels

- Conceptual Schema
  - Logical description of the data
  - Relationships between data items
  - Includes:
    - Table names
    - Column names and types
    - Integrity constrains
- Physical Schema
  - Deals with the organization and storage of the data within the database
  - Details on how to access the data from disk
  - Includes:
    - Files
    - Indices
    - Data partitioning

Modeling data by levels (cont.)

- External Schema
  - Customized version of the global conceptual schema
  - Allow for tailoring access of data for a particular user(s)
  - Can also be used to prevent unauthorized user from accessing sensitive data.
  - Usually defined by means of views on the conceptual schema

Relational DBMS Architecture



Client

Client API

Query Parser

Query Optimizer

Relational Operators

File and Access Methods

Buffer Management

Disk Space Management

Execution Engine

Concurrency and Recovery

DB

DBMS: Client API

- The Client API contains the infrastructure necessary to:
  - Accept connection from client applications
  - Submit queries to the DBMS
  - Extract the data from the database
  - Send commands to create/delete records or tables in the database.
  - Start/end operations such as transactions, recovery or backups.
- Examples APIs:
  - JDBC, ODBC (standards)
  - Informix MI API (proprietary)

DBMS: Query Parser

- The parser takes care of analyzing the syntax of the commands send to the DBMS via the client API.
- Many parsers, also check some the semantics in the statements, so they are more than just parsers.
- This layer also generates an initial representation of the query that has been posed to the system
- Often the parser fetches metadata from catalog in order to give the next layer some valuable information to perform their task.
- Example: SQL parser found in most DBMS
  - Also, OQL parser found in many Object-Oriented DBMS.

DBMS: Query Optimizer

- Often, there are many alternatives to solve a query posed to the system.
- The optimizer takes care of choosing the best alternative to solve the query.
  - Here best is defined as the alternative of lesser cost, where the cost of each alternative might be defined as:
    - Amount of I/O operations
    - Wall-clock time to execute the query
    - System usage time to execute the query
- Optimizer uses dynamic programming search to find best alternative
  - Search by construction
  - Alternative is called query plan and is a tree representing the relational operators to be executed to answer the query.

DBMS: Relational Operators

- At this layer, the set of relational operator supported by the DBMS are implemented.
- The most common of the operators are:
  - Selections
  - Simple projections
  - Generalized projections
  - Aggregates
  - Sorting
  - Joins
  - Unions
- An execution plan produced by the optimizer is used to represent the set of operators to be executed.

DBMS: File and Access Methods

- At this layer, we find the implementation of the various mechanism available to access the data in each table.
- The basic service in the unordered file, which is also called the heap. This provides a service to access record sequentially and in no particular order.
- Also, we find mechanisms to index the records in a table. Using these indices, we can speed up the execution of query by only reading the necessary records from the database.
- Example indices: B+-tree, Hash-index, ISAM, R-tree.
- Lots of research papers and Ph.D. thesis have been written on this subject.

DBMS: Buffer Management

- Data from the Database is read in chunks, often called pages.
- Sometimes, many queries access the same chunks of data.
- The buffer manager takes care of controlling the way in which data pages are read from disk and kept in memory.
- Buffer managers often try to cache frequently used pages, at the expense of less frequently used ones.
- Many systems use pre-fetching algorithms to “guess” the next page(s) that will be accessed, based on the previous requests, and these pages are fetched and cached in memory before being requested.

DBMS: Disk Space Management

- This layer provides the abstraction of a page of data from disk..
- This layer provides the infrastructure necessary to create, delete, read and write data pages associated with a database.
- This is the lowest layer in the system, often dealing with issues such as:
  - Fixed-sized vs. variable length records
  - Striping of data pages (partitioning across multiple disk).
  - Memory alignment of data when moved between disk and memory.

Advantages of a DBMS

- Data Independence
  - External schema shields from changes in conceptual schema
    - Logical data independence
    - Remove columns, add columns in base tables
  - Conceptual schema shields from changes in physical schema
    - Drop an index
    - Re-partition data over new disks
- Efficient data access
  - Proven, sophisticated data techniques to quickly read/write data
  - No amateur work…

Advantages of a DBMS (cont.)

- Data integrity and security
  - Protects data from unauthorized access
  - Enforces certain properties on the data
- Data administration
  - Packages to manage and preserve data in a professional way
  - Theory and practice for professional to do this job
- Concurrency
  - Infrastructure to enforce safe access to the same data items by multiple user without unexpected side-effects.

Advantages of a DBMS (cont.)

- Crash Recovery
  - Infrastructure to repair lost or damage data due to system failures:
    - Power failures
    - Media failures (disk crash)
- Reduced application development time
  - Infrastructure to quickly build application that let users interact with their data
  - JDBC, ODBC
  - Forms and other GUIs.

Database Professionals

- Database implementors
  - Build modules that go inside the DBMS
  - Students in ICOM 6005\
- Database application developers
  - Build application that run on top of the DBMS and are used by end-users to interact with their data.
  - Students in ICOM 4017
- Database Administrators
  - Create database schema
  - Maintain and tune the DBMS engine
  - Maintain and tune the data in the DBMS
  - Students in ICOM 4017
  - Corporations need them ($$$)