Overview of Database Management Systems

Debapriyo Majumdar

DBMS – Fall 2016

Indian Statistical Institute Kolkata

Data: a scenario

Must be accessed concurrently by all the employees

Questions about the data must be answered quickly

500GB Data

Employees, departments, products, sales, …

Changes made by different users must be performed consistently

Access to certain parts must be restricted in different ways

Very traditional option: store in disk files

Drawbacks of using file system to store data

- Data redundancy and inconsistency
  - Multiple file formats, duplication of information in different files
- Difficulty in accessing data
  - Need to write a new program to carry out each new task
- Data isolation
  - Multiple files and formats
- Integrity problems
  - Integrity constraints (e.g., account balance > 0) become "buried" in program code rather than being stated explicitly
  - Hard to add new constraints or change existing ones
- Security problems
  - Hard to provide user access to some, but not all, data

Drawbacks of using file system to store data

- Atomicity of updates
  - Failures may leave database in an inconsistent state with partial updates carried out
  - *Example: Transfer of funds from one account to another should either complete or not happen at all*
- Concurrent access by multiple users
  - Concurrent access needed for performance
  - Uncontrolled concurrent accesses can lead to inconsistencies
  - *Example: Two people reading a balance (say 100) and updating it by withdrawing money (say 50 each) at the same time*

Database systems offer solution to all of these problems

Advantages of DBMS

- Data Independence: Abstract view of data
  - Details of data representation and storage are not exposed to application programs
- Efficient Data Access: a variety of sophisticated techniques to store and retrieve data efficiently
- Data Integrity and Security: DBMS can enforce integrity constraints
  - For example, before inserting salary information for an employee, the DBMS can check that the department budget is not exceeded
  - Also, it can enforce access control that govern what data is visible to different classes of users
- Data Administration: centralizing the administration of data can minimize redundancy and for fine-tuning the storage of the data to make retrieval efficient

Advantages of DBMS

- Concurrent Access and Crash Recovery: schedules concurrent accesses to the data in such a manner that for the users it is equivalent to data accessed by one user at a time
  - Also, protects users from the effects of system failures.
- Reduced Application Development Time
  - Supports important functions that are common to many applications accessing data in the DBMS
  - High-level interface to the data, facilitates quick application development
  - DBMS applications are also likely to be more robust than similar stand-alone applications because many important tasks are handled by the DBMS, do not have to be debugged and tested in the application

When not to use DBMS

- Can there be any such scenario?
- Any DBMS is a complex system
- Optimized for certain kinds of workload
- The performance may not be adequate for certain specialized kinds of tasks
  - For example, a few well defined time-critical operations
  - Need to manipulate data in a way not well supported by the DBMS query language
- Example: text analysis, search

Data Models

- A collection of tools for describing
  - Data
  - Data relationships
  - Data semantics
  - Data constraints
- Relational model
  - IBM DB2, Informix, Oracle, Sybase, MS Access, FoxBase, Paradox, Tandem, Teradata, …
- Entity-Relationship data model (mainly for database design)
- Object-based data models (Object-oriented and Object-relational)
  - DBMS products from IBM, Informix, ObjectStore, Oracle, Versant, …
- Semi-structured data model (XML)
- Other older models:
  - Network model (IDS, IDMS)
  - Hierarchical model (IBM's IMS DBMS)

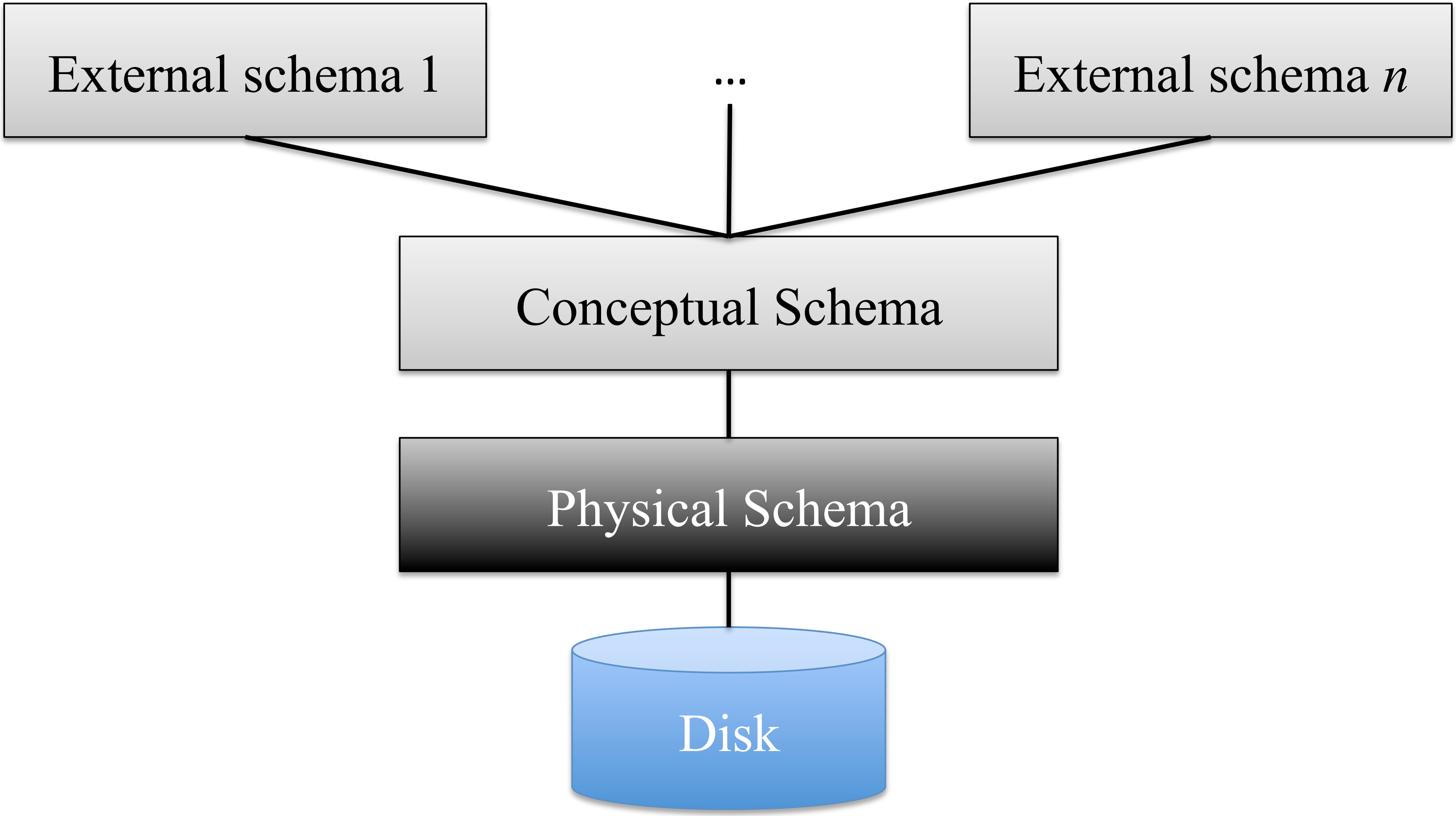
Relational Model

- A relation: a set of records (more formally later)
- Schema: a description of data in terms of a data model
  - Name of the schema, name of each field, type of each field

```
Students(
   roll:  string,
   name:  string,
   age:  integer,
   gpa:  real
)
```

roll	name	age	gpa
0001	John	23	9.1
0002	Amit	24	8.7
0003	Vivek	25	9.2
0004	Sandhya	23	9.0
….	….	….	….

Levels of abstraction in a DBMS



Levels of abstraction in a DBMS

- **Physical schema:** describes how a record (e.g., student) is stored.
- **Conceptual / Logical schema:** describes data stored in database, and the relationships among the data.

```
Students(
roll:  string,
name:  string,
age:  integer,
gpa:  real )
```

- **External schema / View level:** application programs hide details of data types. Views can also hide information (such as an employee's salary) for security purposes.

Instance and data indepdence

- **Instance** – the actual content of the database at a particular point in time
  - Analogous to the value of a variable
- **Physical Data Independence** – the ability to modify the physical schema without changing the logical schema
  - Applications depend on the logical schema
  - In general, the interfaces between the various levels and components should be well defined so that changes in some parts do not seriously influence others

Storage Management

- **Storage manager** is a program module that provides the interface between the low-level data stored in the database and the application programs and queries submitted to the system.
- The storage manager is responsible to the following tasks:
  - Interaction with the OS file manager
  - Efficient storing, retrieving and updating of data
- Issues:
  - Storage access
  - File organization
  - Indexing and hashing

Query Processing

- Relational database systems allow a rich class of questions out of the box
  1. What is the name of the student with roll 0043?
  2. What is the average marks in class PGDBA5?
  3. How many students have enrolled this year?

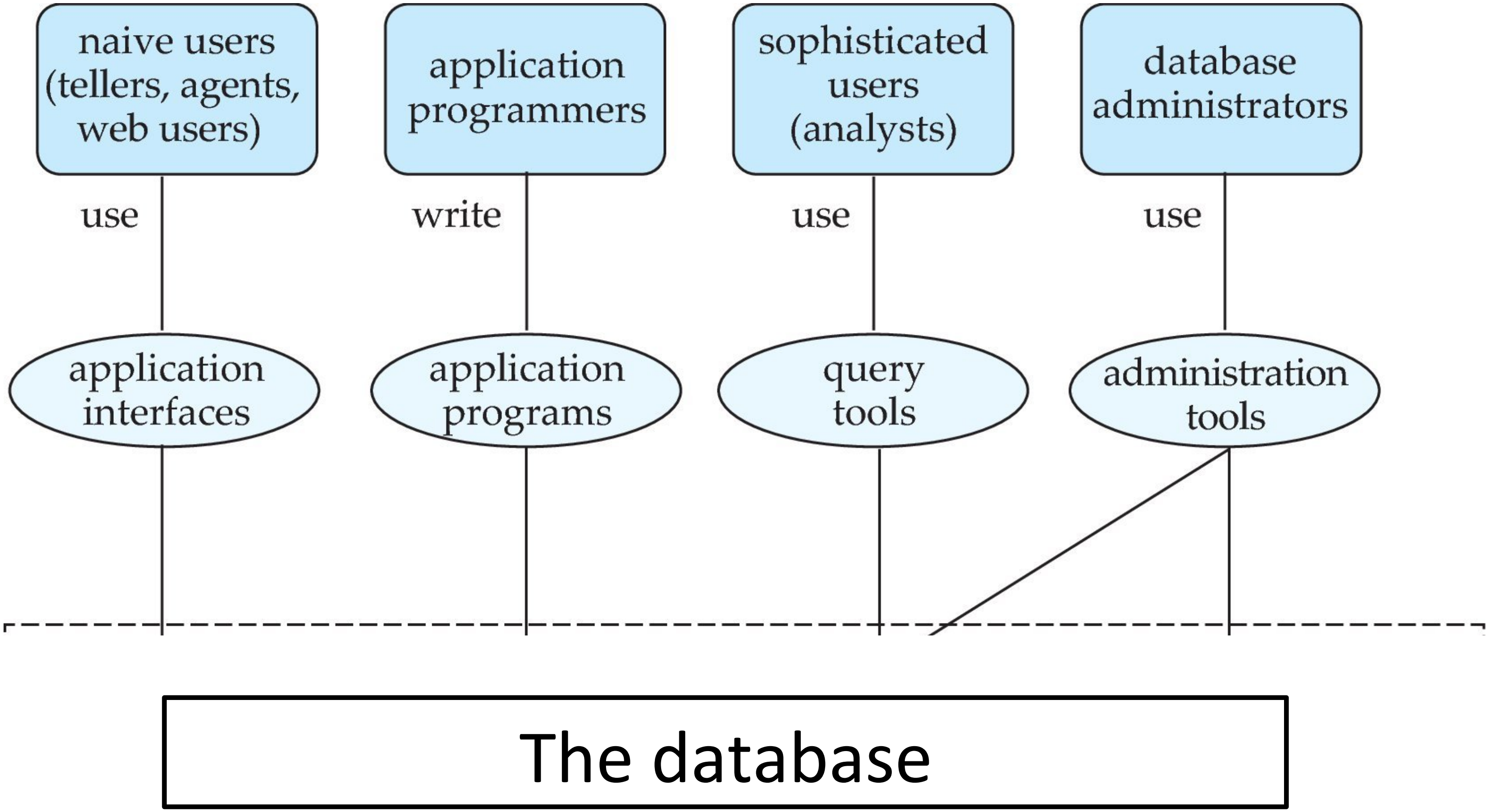
  …

  These questions are called queries
- DBMS systems provide a specialized language for querying

Transaction Management

- What if the system fails?
- What if more than one user is concurrently updating the same data?
- A **transaction** is a collection of operations that performs a single logical function in a database application
- **Transaction-management component** ensures that the database remains in a consistent (correct) state despite system failures (e.g., power failures and operating system crashes) and transaction failures.
- **Concurrency-control manager** controls the interaction among the concurrent transactions, to ensure the consistency of the database.

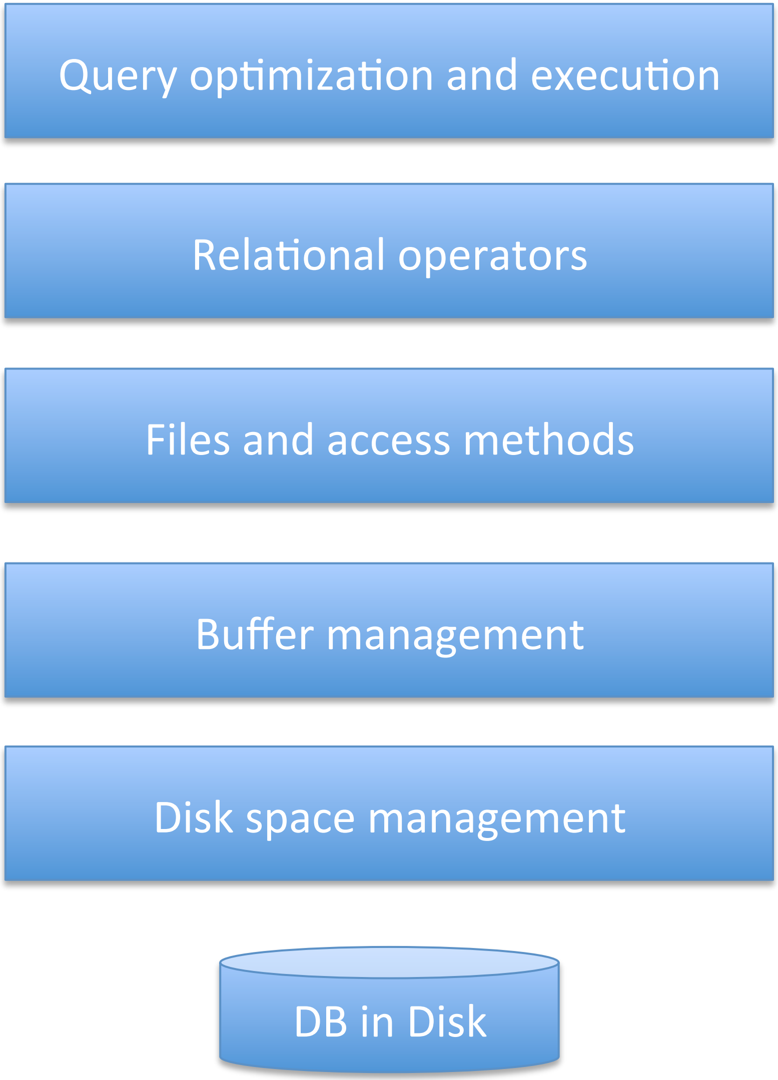
DBMS Users



naive users (tellers, agents, web users)
application programmers
sophisticated users (analysts)
database administrators
use
write
use
use
application interfaces
application programs
query tools
administration tools
The database

Structure of a DBMS

- A simplified DBMS layered architecture
  - Concurrency control and recovery components are not shown in this figure
- There can be other architectures
  - Each system has its own variant



History

- 1950s and early 1960s:
  - Data processing using magnetic tapes for storage
    - Tapes provided only sequential access
  - Punched cards for input
- Late 1960s and 1970s:
  - Hard disks allowed direct access to data
  - Network and hierarchical data models in widespread use
  - Ted Codd defines the relational data model
    - Would win the ACM Turing Award for this work
    - IBM Research begins System R prototype
    - UC Berkeley begins Ingres prototype
  - High-performance (for the era) transaction processing

History

- 1980s:
  - Research relational prototypes evolve into commercial systems
    - SQL becomes industrial standard
  - Parallel and distributed database systems
  - Object-oriented database systems
- 1990s:
  - Large decision support and data-mining applications
  - Large multi-terabyte data warehouses
  - Emergence of Web commerce
- Early 2000s:
  - XML and XQuery standards
  - Automated database administration
- Later 2000s:
  - Giant data storage systems
    - Google BigTable, Yahoo PNuts, Amazon, ..

References and acknowledgements

- Database Management Systems: book by Raghu Ramakrishnan and Johannes Gehrke, 3rd Edition
  - Supporting teaching material at: http://pages.cs.wisc.edu/~dbbook/openAccess/thirdEdition/slides/slides3ed.html
- Database System Concepts: book by Silberschatz, Korth and Sudarshan
  - Some of the slides are **reused and adapted** from the slides at: www.db-book.com