
Lecture 1: Introduction to Database Systems

Dragan Mirkovic
Department of Computer Science
University of Houston

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

Announcements

- Today:
 - Introduction to DBMS
 - Ch 1. In Ramakrishnan and Ch. 1 in Navathe.
 - Overview of DBMS
 - Basic definitions
 - Examples
 - Main characteristics of Database Technology
 - Data Models

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

Data management

- The age we live in is characterized by the explosion of the amount of information that is available to everyone
- Data management is critical in many different areas of human endeavor
- From the earliest days of computers data storage and manipulation have been one of the main applications
- DBMS have emerged as an efficient way of data management

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

Data Management

- Need for data management is exploding
 - Datasets increasing in diversity and volume
 - Digital libraries
 - Interactive video
 - Human genome project
 - Earth Observing system (NASA)
 - Many other scientific projects generate large amounts of raw data
 - Need for very efficient management

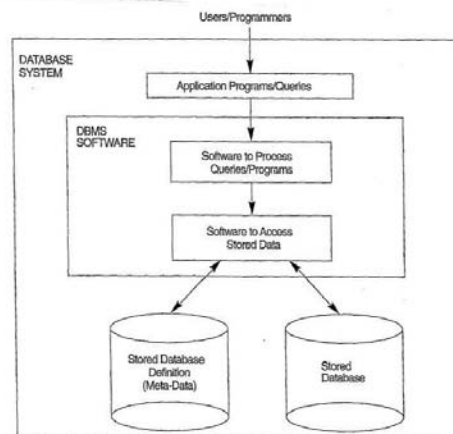
D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

Basic Definitions

- **Data:** Known facts that can be recorded and have an implicit meaning.
- **Database:** A collection of related data.
- **Mini-world:** Some part of the real world about which data is stored in a database. For example, student grades and transcripts at a university.
- **Database Management System (DBMS):** A software package/ system to facilitate the creation and maintenance of a computerized database.
- **Database System:** The DBMS software together with the data itself. Sometimes, the applications are also included.

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

Database System Environment



D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

Example

- **Mini-world for the example:**
 - Part of a UNIVERSITY environment.
- **Some mini-world *entities*:**
 - STUDENTs
 - COURSEs
 - SECTIONs (of COURSEs)
 - (academic) DEPARTMENTs
 - INSTRUCTORs

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

Example

- **Some mini-world *relationships*:**
 - SECTIONs *are of* specific COURSEs
 - STUDENTs *take* SECTIONs
 - COURSEs *have* prerequisite COURSEs
 - INSTRUCTORs *teach* SECTIONs
 - COURSEs *are offered by* DEPARTMENTs
 - STUDENTs *major in* DEPARTMENTs
- **NOTE:**
 - The above could be expressed in the *ENTITY-RELATIONSHIP* data model.

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

Main Characteristics of Database Technology

- Self-contained nature of a database system:
 - A DBMS catalog stores the description of the database.
 - The description is called (meta-data).
 - This allows the DBMS software to work with different databases.
- Insulation between programs and data:
 - Called *program-data independence*.
 - Allows changing data storage structures and operations without having to change the DBMS access programs.

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

Main Characteristics of Database Technology...

- Data Abstraction:
 - A data model is used to hide storage details and present the users with a conceptual view of the database.
- Support of multiple views of the data:
 - Each user may see a different view of the database, which describes only the data of interest to that user.

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

Additional Benefits of Database Technology

- Controlling redundancy in data storage and in development and maintenance efforts.
- Sharing of data among multiple users.
- Restricting unauthorized access to data.
- Providing multiple interfaces to different classes of users.
- Representing complex relationships among data.
- Enforcing integrity constraints on the database.
- Providing backup and recovery services.
- Potential for enforcing standards.
- Flexibility to change data structures.
- Reduced application development time.
- Availability of up-to-date information.
- Economies of scale.

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

When not to use a DBMS

- Main inhibitors (costs) of using a DBMS:
 - High initial investment and possible need for additional hardware.
 - Overhead for providing generality, security, recovery, integrity, and concurrency control.
- When a DBMS may be unnecessary:
 - If the database and applications are simple, well defined, and not expected to change.
 - If there are stringent real-time requirements that may not be met because of DBMS overhead.
 - If access to data by multiple users is not required.

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

When not to use a DBMS...

- When no DBMS may suffice:
 - If the database system is not able to handle the complexity of data because of modeling limitations
 - If the database users need special operations not supported by the DBMS.

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

Data Models

- **Data Model:**
 - A set of concepts to describe the structure of a database, and certain constraints that the database should obey.
 - High-level data description constructs that hide low-level storage details
- **Data Model Operations:**
 - Operations for specifying database retrievals and updates by referring to the concepts of the data model.

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

Categories of data models

- **Conceptual (high-level, semantic) data models:**
 - Provide concepts that are close to the way many users perceive data. (Also called entity-based or object-based data models. ER model is one example.)
- **Physical (low-level, internal) data models:**
 - Provide concepts that describe details of how data is stored in the computer.
- **Implementation (record-oriented) data models:**
 - Provide concepts that fall between the above two, balancing user views with some computer storage details.

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

History of Data Models

- **Relational Model:**
 - proposed in 1970 by E.F. Codd (IBM), first commercial system in 1981-82. Now in several commercial products (ORACLE, SYBASE, INFORMIX, CA-INGRES).
- **Network Model:**
 - the first one to be implemented by Honeywell in 1964-65 (IDS System).
 - Adopted heavily due to the support by CODASYL (CODASYL - DBTG report of 1971).
 - Later implemented in a large variety of systems:
 - IDMS (Cullinet - now CA), DMS 1100 (Unisys),
 - IMAGE (H.P.), VAX -DBMS (Digital).

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

History of Data Models...

- **Hierarchical Data Model :**
 - implemented in a joint effort by IBM and North American Rockwell around 1965.
 - Resulted in the IMS family of systems. The most popular model. Other systems based on this model: System 2k (SAS inc.)
- **Object-oriented Data Model(s) :**
 - New applications (CAD, scientific experiments, geographic information systems) are object based (state+behavior)
 - several models: One set comprises models of persistent O-O Programming Languages such as C++ (e.g., in OBJECTSTORE or VERSANT), and Smalltalk (e.g., in GEMSTONE).
 - Additionally, systems like O2, ORION (at MCC - then ITASCA), IRIS (at H.P.- used in Open OODB).
- **Object-Relational Models :**
 - Most Recent Trend.
 - Exemplified in ILLUSTRATE and UNISQL systems.

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

Hierarchical Model

- | | |
|--|---|
| <ul style="list-style-type: none">• ADVANTAGES:<ul style="list-style-type: none">– Hierarchical Model is simple to construct and operate on– Corresponds to a number of natural hierarchically organized domains - e.g.,<ul style="list-style-type: none">• assemblies in manufacturing,• personnel organization in companies– Language is simple; uses constructs like GET, GET UNIQUE, GET NEXT, GET NEXT WITHIN PARENT etc. | <ul style="list-style-type: none">• DISADVANTAGES:<ul style="list-style-type: none">– Navigational and procedural nature of processing– Database is visualized as a linear arrangement of records– Little scope for "query optimization" |
|--|---|

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

Network Model

- **ADVANTAGES:**
 - Network Model is able to model complex relationships and represents semantics of add/delete on the relationships.
 - Can handle most situations for modeling using record types and relationship types.
 - Language is navigational; uses constructs like FIND, FIND member, FIND owner, FIND NEXT within set, GET etc. Programmers can do optimal navigation through the database.
- **DISADVANTAGES:**
 - Navigational and procedural nature of processing
 - Database contains a complex array of pointers that thread through a set of records.
 - Little scope for automated "query optimization"

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

Note on Data Models

- The models used most frequently:
 - Relational data model
- Legacy data models:
 - Network data model
 - Hierarchical data model
- New families of data models:
 - Object data models
 - Object-relational data models

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

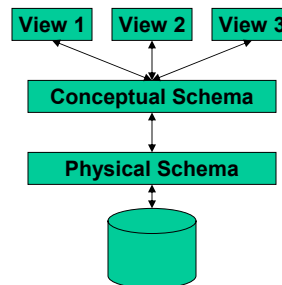
Schema versus Instances

- **Database Schema:** The *description* of a database. Includes descriptions of the database structure and the constraints that should hold on the database.
- **Schema Diagram:** A diagrammatic display of (some aspects of) a database schema.
- **Database Instance:** The actual data stored in a database at a *particular moment in time* . Also called **database state** (or **occurrence**).
- The **database schema** changes *very infrequently* .
- The **database state** changes *every time the database is updated* .
- **Schema** is also called **intension**, whereas **state** is called **extension**.

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

Tree-Schema Architecture

- Proposed to support DBMS characteristics of:
 - Program-data independence.
 - Support of multiple views of the data.
- Defines DBMS schemas at three levels :
 - **Internal schema:** data storage structures and access paths. (physical data model)
 - **Conceptual schema:** describes the structure and constraints for the whole database. Uses a conceptual or an implementation data model.
 - **External schemas:** describes the various user views. Usually uses the same data model as the conceptual level.



- Mappings among schema levels are also needed. Programs refer to an external schema, and are mapped by the DBMS to the internal schema for execution.

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

Data Independence

- **Logical Data Independence:** The capacity to change the conceptual schema without having to change the external schemas and their application programs.
- **Physical Data Independence:** The capacity to change the internal schema without having to change the conceptual schema.
- When a schema at a lower level is changed, only the **mappings** between this schema and higher-level schemas need to be changed in a DBMS that fully supports data independence. The higher-level schemas themselves are *unchanged*. Hence, the application programs need not be changed since they refer to the external schemas.

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

DBMS Languages

- Data Definition Language (DDL):
 - Used by the DBA and database designers to specify the conceptual schema of a database.
 - In many DBMSs, the DDL is also used to define internal and external schemas (views).
 - In some DBMSs, separate storage definition language (SDL) and view definition language (VDL) are used to define internal and external schemas.
- Data Manipulation Language (DML):
 - Used to specify database retrievals and updates.
 - DML commands (data sublanguage) can be embedded in a general-purpose programming language (host language), such as COBOL, PL/1 or PASCAL.
 - Alternatively, stand-alone DML commands can be applied directly (query language).
- In current DBMS the above languages are not considered distinct - One comprehensive integrated language

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

DBMS Interfaces

- Stand-alone query language interfaces.
- Programmer interfaces for embedding DML in programming languages:
 - Pre-compiler Approach
 - Procedure (Subroutine) Call Approach
- User-friendly interfaces:
 - Menu-based
 - Graphics-based (Point and Click, Drag and Drop etc.)
 - Forms-based
 - Natural language
 - Combinations of the above
 - Speech as Input (?) and Output
 - Web Browser as an interface

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

DBMS Interfaces...

- Parametric interfaces using function keys.
 - Small set of operations that must be performed repeatedly
 - Example: bank tellers
- Report generation languages
 - Inherent to database or outside scripts
 - Example: PERL
- Interfaces for the DBA:
 - Creating accounts, granting authorizations
 - Setting system parameters
 - Changing schemas or access path

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

Database System Environment

- Operating system:
 - Controls disk input/output
- Stored data manager
 - DBMS module
 - Controls the information stored on disks
- DDL compiler
 - Processes schema definitions and stores meta-data
 - Run-time database processor
 - Query compiler
 - Application programs compilers and pre-compilers

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

Database System Utilities

- To perform certain functions such as:
 - *Loading* data stored in files into a database.
 - *Backing up* the database periodically on tape.
 - *Reorganizing* database file structures.
 - *Report generation* utilities.
 - *Performance monitoring* utilities.
 - Other functions, such as sorting , user monitoring , data compression , etc.
- Data dictionary / repository:
 - Used to store schema descriptions and other information such as design decisions, application program descriptions, user information, usage standards, etc.
 - *Active* data dictionary is accessed by DBMS software and users/DBA.
 - *Passive* data dictionary is accessed by users/DBA only.

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

Classification of DBMSs

- Based on the data model used:
 - Traditional: Relational, Network, Hierarchical.
 - Emerging: Object-oriented, Object-relational.
- Other classifications:
 - Single-user (typically used with micro- computers) vs. multi-user (most DBMSs).
 - Centralized (uses a single computer with one database) vs. distributed (uses multiple computers, multiple databases)
- Distributed Database Systems
 - client server based database systems
 - a set of database servers supporting a set of clients.

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

Transaction Management

- Transaction: any execution of a user program in a DBMS
 - Basic unit of change as seen by the DBMS
- Concurrent execution
 - Ordering of simultaneous requests
- Security
 - Prevent unauthorized access
 - Crash recovery
- Typical examples
 - Airline reservations
 - Financial transactions

D. Mirkovic, COSC 3480: Design of File and Database Systems, Spring 2005

Summary

- Introduction of the main concepts in DBMS
 - **Data models** and their categories
 - High-level (conceptual)
 - Low-level (physical)
 - Representational or implementational (record-based, object-oriented)
 - **Schema** (description of the database)
 - Internal, conceptual and external
 - **Languages and interfaces**
 - DBMS environment
 - DBMS classification