Database Management in Software Development

CSc3350

Dr. William Greg Johnson
Department of Computer Science
Georgia State University

Georgia State University | COLLEGE OF ARTS & SCIENCES

Database Management Systems in Software Development

- Correctness
- Maintainability
- Documented
- Well organized (formatted)

Objectives

- Historical information
- The difference between data and information
- What a database is, what the different types of databases are, and why they are valuable
- The importance of database design
- How modern databases evolved from file systems
- About flaws in file system data management
- What the database system’s main components are and how a database system differs from a file system
- The main functions of a database management system (DBMS)

Georgia State University | COLLEGE OF ARTS & SCIENCES

History of Database Systems

- File based systems
  - File based systems came in 1960s and was widely used. It stores information and organize it into storage devices like a hard disk, a CD-ROM, USB, SSD, floppy disk, etc.
- Relational Model
  - Relational Model introduced by E.F.Codd in 1969. The model stated that data will be represented in tuples. A relational model groups data into one or more tables. These tables are related to each other using common records.
- dBase
  - dBase was one of the first database management systems for microcomputers and the most successful in its day. The system includes the core database engine, a query system, a forms engine, and a programming language that ties these components together.
- Centralized DBMS and Data Warehousing
  - In 1990s, centralized DBMS server was used. The period also witnessed the introduction of MS-Access. In addition, users worked on Internet and data warehousing introduced.

History of Database Systems (continued)

- NoSQL (Not only SQL)
  - NoSQL, Big Data came in 2008. Big Data described large value of both the structured and unstructured data. This data is so large that traditional database cannot process it.
- Hadoop
  - Hadoop and MongoDB launched in 2009. Hadoop use distributed file system for storing big data, and MapReduce to process it. Hadoop excels in storing and processing of huge data of various formats such as arbitrary, semi-, unstructured, etc. MongoDB is a cross-platform, document-oriented database that provides, high performance, high availability, and easy scalability. It works works on the concept of collection and document.
- Hbase
  - It introduced in 2010 and is a database built on top of the HDFS (Hadoop Distributed File System). HBase provides fast lookups for larger tables.

Data vs. Information

- Data:
  - Raw facts; building blocks of information
  - Unprocessed information
- Information:
  - Data processed to reveal meaning
- Accurate, relevant, and timely information is key to good decision making
- Good decision making is the key to survival in a global environment

Introducing the Database and the DBMS

- Database—shared, integrated computer structure that stores:
  - End user data (raw facts)
  - Metadata (data about data)

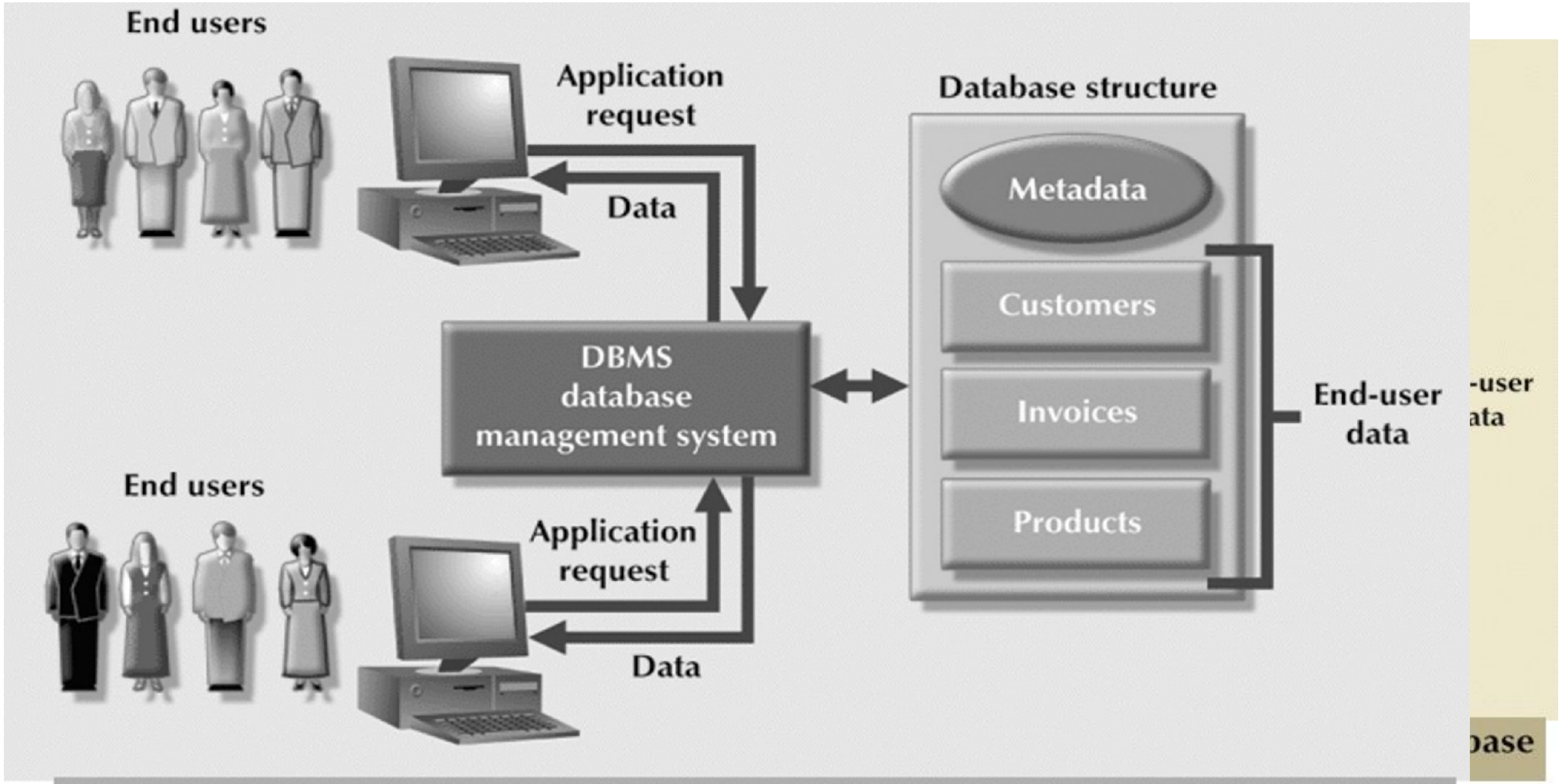
Introducing the Database and the DBMS (continued)

- DBMS (database management system):
  - Collection of programs that manages database structure and controls access to data
  - Possible to share data among multiple applications or users
  - Makes data management more efficient and effective

Role and Advantages of the DBMS (continued)

- End users have better access to more and better-managed data
  - Promotes integrated view of organization’s operations
  - Probability of data inconsistency is greatly reduced
  - Possible to produce quick answers to ad hoc queries

Role and Advantages of the DBMS (continued)



The DBMS manages the interaction between the end user and the database

Types of Databases

- Single-user:
  - Supports only one user at a time
- Desktop:
  - Single-user database running on a personal computer
- Multi-user:
  - Supports multiple users at the same time, now cloud hosted

Types of Databases (continued)

- Workgroup:
  - Multi-user database that supports a small group of users or a single department
- Enterprise:
  - Multi-user database that supports a large group of users or an entire organization

Types of Databases (continued)

Can be classified by location:

- Centralized:
  - Supports data located at a single site
- Distributed:
  - Supports data distributed across several sites

Types of Databases (continued)

Can be classified by use:

- Transactional (or production):
  - Supports a company's day-to-day operations
- Data warehouse:
  - Stores data used to generate information required to make tactical or strategic decisions
  - Often used to store historical data
  - Structure is quite different

Why Database Design is Important

- Defines the database’s expected use
- Different approach needed for different types of databases
- Avoid redundant data
- Poorly designed database generates errors leads to bad decisions can lead to failure of organization

Historical Roots: Files and File Systems

- Managing data with file systems is obsolete
  - Understanding file system characteristics makes database design easier to understand
  - Awareness of problems with file systems helps prevent similar problems in DBMS
  - Knowledge of file systems is helpful if you plan to convert an obsolete file system to a DBMS

Historical Roots: Files and File Systems (continued)

Manual File systems:

- Collection of file folders kept in file cabinet
- Organization within folders based on data's expected use (ideally logically related)
- System adequate for small amounts of data with few reporting requirements
- Finding and using data in growing collections of file folders became time-consuming and cumbersome

Historical Roots: Files and File Systems (continued)

Conversion from manual to computer system:

- Could be technically complex, requiring hiring of data processing (DP) specialists
- Resulted in numerous "home-grown" systems being created
- Initially, computer files were similar in design to manual files

Historical Roots: Files and File Systems (continued)

- DP specialist wrote programs for reports:
  - Monthly summaries of types and amounts of insurance sold by agents
  - Monthly reports about which customers should be contacted for renewal
  - Reports that analyzed ratios of insurance types sold by agent
  - Customer contact letters summarizing coverage

Historical Roots: Files and File Systems (continued)

- Other departments requested databases be written for them
  - SALES database created for sales department
  - AGENT database created for personnel department

Historical Roots: Files and File Systems (continued)

- As number of databases increased, small file system evolved
- Each file used its own application programs
- Each file was owned by individual or department who commissioned its creation

Example of Early Database Design

- As system grew, demand for DP's programming skills grew
- Additional programmers hired
- DP specialist evolved into DP manager, supervising a DP department
- Primary activity of department (and DP manager) remained programming

Problems with File System Data Management

- Every task requires extensive programming in a third-generation language (3GL)
  - Programmer must specify task and how it must be done
- Modern databases use fourth-generation languages (4GL)
  - Allow users to specify what must be done without specifying how it is to be done
- Example: DO Loop VS. Select Statement

Georgia State University | COLLEGE OF ARTS & SCIENCES

Problems with File System Data Management (continued)

- Time-consuming, high-level activity
- As number of files expands, system administration becomes difficult
- Making changes in existing file structure is difficult
- File structure changes require modifications in all programs that use data in that file

Problems with File System Data Management (continued)

- Modifications are likely to produce errors, requiring additional time to “debug” the program
- Security features hard to program and therefore often omitted

Structural and Data Dependence

- Structural dependence
  - Access to a file depends on its structure
- Data dependence
  - Changes in the data storage characteristics without affecting the application program’s ability to access the data
  - Logical data format
    - How the human being views the data
  - Physical data format
    - How the computer “sees” the data

Field Definitions and Naming Conventions

- Flexible record definition anticipates reporting requirements by breaking up fields into their component parts
- Example:

Customer Last Name .... Cust-LName

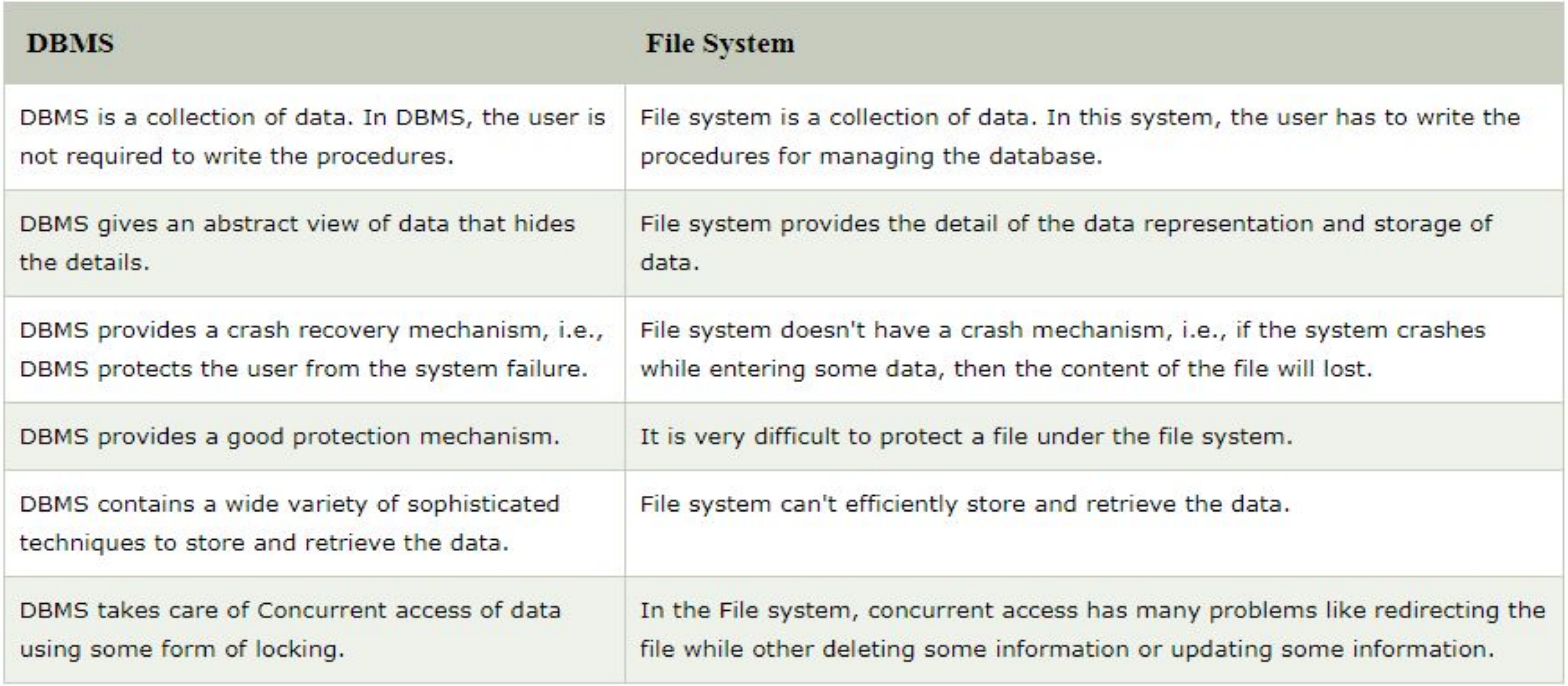
Data Redundancy

- Data redundancy results in data inconsistency
  - Different and conflicting versions of the same data appear in different places
- Errors more likely to occur when complex entries are made in several different files and/or recur frequently in one or more files
- Data anomalies develop when required changes in redundant data are not made successfully

Database Systems

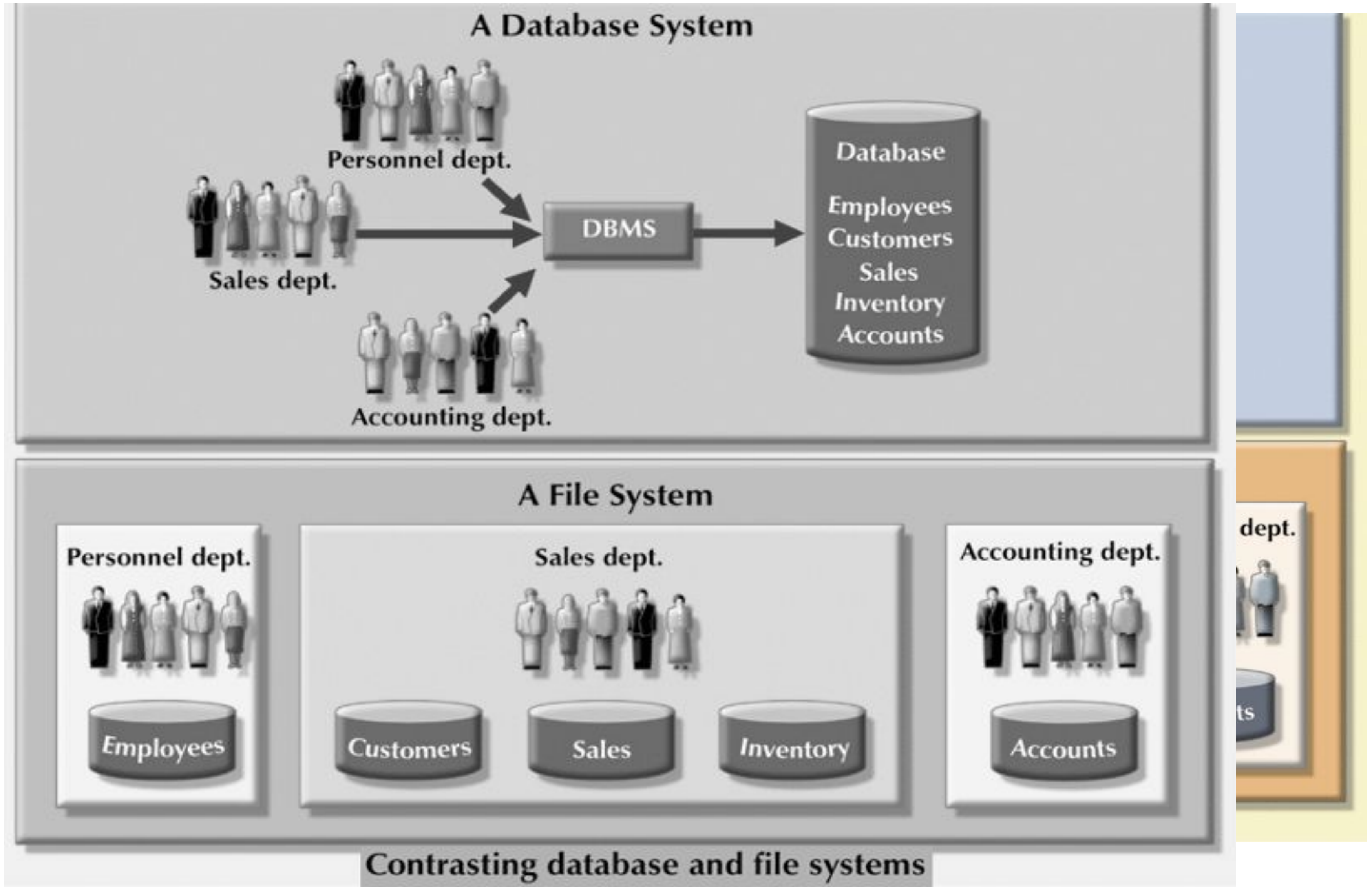
- Problems inherent in file systems make using a database system desirable
- File system
  - Many separate and unrelated files
- Database
  - Logically related data stored in a single logical data repository

Files vs DBMS



DBMS	File System
DBMS is a collection of data. In DBMS, the user is not required to write the procedures.	File system is a collection of data. In this system, the user has to write the procedures for managing the database.
DBMS gives an abstract view of data that hides the details.	File system provides the detail of the data representation and storage of data.
DBMS provides a crash recovery mechanism, i.e., DBMS protects the user from the system failure.	File system doesn't have a crash mechanism, i.e., if the system crashes while entering some data, then the content of the file will lost.
DBMS provides a good protection mechanism.	It is very difficult to protect a file under the file system.
DBMS contains a wide variety of sophisticated techniques to store and retrieve the data.	File system can't efficiently store and retrieve the data.
DBMS takes care of Concurrent access of data using some form of locking.	In the File system, concurrent access has many problems like redirecting the file while other deleting some information or updating some information.

Database Systems vs File Systems



Contrasting database and file systems

The Database System Environment

- Database system is composed of five main parts:
  - Hardware
  - Software
    - Operating system software
    - DBMS software
    - Application programs and utility software
  - People
  - Procedures
  - Data

DBMS Functions

- DBMS performs functions that guarantee integrity and consistency of data
  - Data dictionary management
    - defines data elements and their relationships
  - Data storage management
    - stores data and related data entry forms, report definitions, etc.

DBMS Functions (continued)

- Data transformation and presentation
  - translates logical requests into commands to physically locate and retrieve the requested data
- Security management
  - enforces user security and data privacy within database

DBMS Functions (continued)

- Multiuser access control
  - uses sophisticated algorithms to ensure multiple users can access the database concurrently without compromising the integrity of the database
- Backup and recovery management
  - provides backup and data recovery procedures
- Data integrity management
  - promotes and enforces integrity rules

DBMS Functions (continued)

- Database access languages and application programming interfaces
  - provide data access through a query language

- Database communication interfaces
  - allow database to accept end-user requests via multiple, different network environments

Database Concepts

- Basic Functions
- ACID
- VADE(R)

Create, Read, Update and Delete

- Create, read, update and delete (CRUD) are the four basic functions of persistent storage a major part of nearly all computer software.
- Sometimes CRUD is expanded with the words retrieve instead of read or destroys instead of delete. It is also sometimes used to describe user interface conventions that facilitate viewing, searching, and changing information; often using computer-based forms and reports.

Alternate terms for CRUD

- ABCD: add, browse, change, delete
- ACID: add, change, inquire, delete — though this can be confused with the transactional use of the acronym ACID.
- BREAD: browse, read, edit, add, delete
- VADE(R): view, add, delete, edit (and *restore*, for systems supporting transaction processing)

Database Applications

- The acronym CRUD refers to all of the major functions that need to be implemented in a relational database application to consider it complete.
- Each letter in the acronym can be mapped to a standard SQL statement:

<u>Operation</u>	<u>SQL</u>
Create	INSERT
Read (Retrieve)	SELECT
Update	UPDATE

ACID

- In computer science, ACID (Atomicity, Consistency, Isolation, Durability) is a set of properties that guarantee that database transactions are processed reliably. In the context of databases, a single logical operation on the data is called a transaction.
- An example of a transaction is a transfer of funds from one account to another, even though it might consist of multiple individual operations (such as debiting one account and crediting another).

Atomicity

- Atomicity refers to the ability of the DBMS to guarantee that either all of the tasks of a transaction are performed or none of them are.
- For example, the transfer of funds can be completed, or it can fail for a multitude of reasons, but atomicity guarantees that one account won't be debited if the other is not credited.
- Atomicity states that database modifications must follow an “all or nothing” rule.
- Each transaction is said to be “atomic.” If one part of the transaction fails, the entire transaction fails. It is critical that the database management system maintain the atomic nature of transactions despite any DBMS, operating system or hardware failure

Consistency

- Consistency property ensures that the database remains in a consistent state before the start of the transaction and after the transaction is over (whether successful or not).
- Consistency states that only valid data will be written to the database. If, for some reason, a transaction is executed that violates the database’s consistency rules, the entire transaction will be rolled back, and the database will be restored to a state consistent with those rules.
- On the other hand, if a transaction successfully executes, it will take the database from one state that is consistent with the rules to another state that is also consistent with the rules.

Durability

- Durability refers to the guarantee that once the user has been notified of success, the transaction will persist, and not be undone.
- This means it will survive system failure, and that the database system has checked the integrity constraints and won't need to abort the transaction.
- Many databases implement durability by writing all transactions into a log that can be played back to recreate the system state right before the failure.
- A transaction can only be deemed committed after it is safely in the log.

Implementation

- ACID suggests that the database be able to perform all of these operations at once. In fact, this is difficult to arrange.
- There are two popular families of techniques: write ahead logging and shadow paging. In both cases, locks must be acquired on all information that is updated, and depending on the implementation, on all data that is being read.

Implementation

- In write ahead logging, atomicity is guaranteed by ensuring that information about all changes is written to a log before it is written to the database. That allows the database to return to a consistent state in the event of a crash.
- In shadowing, updates are applied to a copy of the database, and the new copy is activated when the transaction commits. The copy refers to unchanged parts of the old version of the database rather than being an entire duplicate

Implementation

- It is difficult to guarantee ACID properties in a network environment.
- Network connections might fail, or two users might want to use the same part of the database at the same time.
- Two-phase commit is typically applied in distributed transactions to ensure that each participant in the transaction agrees on whether the transaction should be committed or not.
- Care must be taken when running transactions in parallel.
- Two phase locking is typically applied to guarantee full isolation.

Summary

- Data are raw facts. Information is the result of processing data to reveal its meaning.
- To implement and manage a database, use a DBMS.
- Database design defines the database structure.
- A well-designed database facilitates data management and generates accurate and valuable information.
- A poorly designed database can lead to bad decision making, and bad decision making can lead to the failure of an organization.

Summary (continued)

- Databases were preceded by file systems.
- Limitations of file system data management:
  - requires extensive programming
  - system administration complex and difficult
  - making changes to existing structures is difficult
  - security features are likely to be inadequate
  - independent files tend to contain redundant data
- DBMS were developed to address the file systems inherent weaknesses

Attributions:

1. https://www.scribd.com/presentation/603834776/Database-Systems-Introduction
2. Razzel Salonga, Albert Vinluan, Aubrey Diego, Marc LauretaJ, erry Esperanz: https://www.scribd.com/presentation/58265136/Database-Management-System

GeorgiaState University | COLLEGE OF ARTS & SCIENCES