Chapter 1: Introduction to Database Systems

Outline:

1.1 General Definition
- Database
- Database Management System (DBMS)
- Database System

1.2 Database Systems Vs File Systems
- The disadvantages of using file systems to store data

1.1 General Definitions

A Database is:

- A collection of related data.
- DB examples: Telephone list, Banking System, ….. etc.

Database characteristics:

1. It is a logically coherent collection of data.
2. It represents a miniworld, and should represent the state of that world accurately.
3. It is designed for specific purpose.

A Database Management System DBMS is:

1. A collection of programs that enables users to defining, constructing and manipulating a databases for various applications.
2. A general purpose software system.
3. Some examples of DBMSs are:
    - Oracle.
    - Access.
    - MySQL.
    - FoxPro.

- Database Defining:
    - Specifying the data types, structures and constraints for the data to be stored in the database.
    - Using DDL (Data Definition Language).
- Database Constructing:
    - Storing the data itself on some storage medium.
    - Using DML (Data Manipulation Language).
- Database Manipulating:
    - Retrieve specific data from database, updating the database to reflect changes in the miniworld and generating reports from data.
    - Using DML (Data Manipulation Language).

A Database System consists of:

- Database, DBMS and applications.

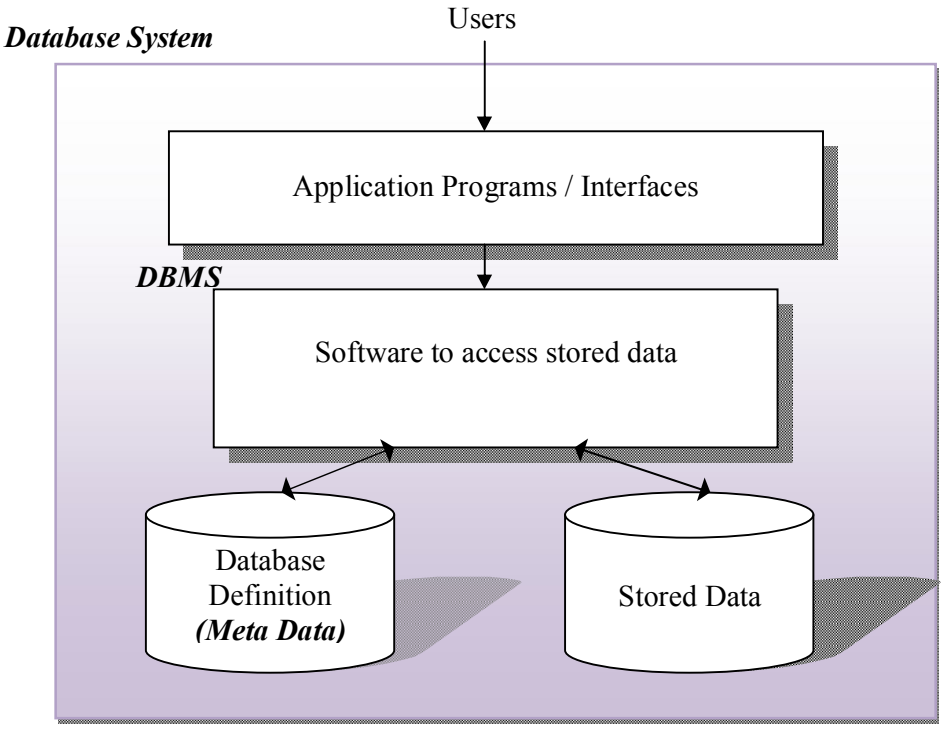

Figure 1.1: The architecture of database system

1.2 Database Systems Vs File Systems

- In the early days, database applications were built directly on top of file systems.
- File based system is a collection of application programs that perform tasks where each program defines and manage it is own data.
- Consider part of a saving-bank that needs to maintain information about its customers and its accounts. Suppose that applications, managing data on customers and accounts, directly use the file systems for storing and retrieving data.
    - According to such approach, the data concerning customers and accounts are stored in records collected in flat files. There is a file for customer records and a file for the account records.
    - Each file may have different format.
    - Programs that use these files depend on knowledge about that format.
    - Example: account information file

      Customer No#, CustName, CustAge, Email, AccNo, Balance, Type

      123, James, 31, james@yahoo.com, 101, 1200, 'saving'
      912, Sali, 22, Sali@yahoo.com, 102, 5600, 'checking'
      123, James, 31, james@yahoo.com, 103, 45553, 'checking'
      …….
    - To manipulate the information, the system has a number of application programs:
        - A program to add new information.

- A program to find the balance of an account.
- A program to debit or credit an account.
- ….. and so on.

- Any question (access) on the data is a small program.
- The new programs are added to the system as the needed arises.

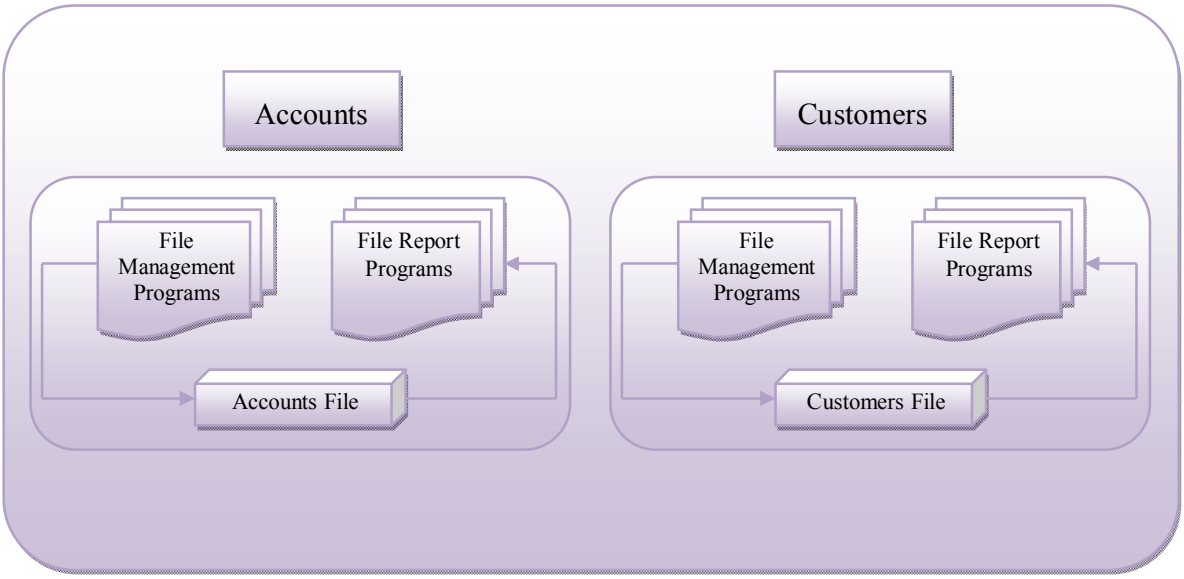

Figure 1.2: A Simple File System

The disadvantages of using file systems to store data:

1. ***Data redundancy and inconsistency***
   - Multiple file formats, duplication of information in different files.
   - Problems:
     - Higher storage leads to waste space.
     - Data inconsistency (the various copies of the same data may no longer agree (multiple formats). For example, John Smith vs Smith J).

2. ***Data isolation and Difficulty in accessing data***
   - Data scattered in various files, and files may be in different formats, writing new program to retrieve the data is difficult.
   - Data characteristics are embodied in programs not stored with the data.
   - Changes in data characteristics requires modifying programs
   - Changes in file structures require modification of all programs using that file.
   - For every query (*Ex: find the saving accounts*) we need to write a program!

3. ***Data Integrity Problems***
   - The data values stored in the database must satisfy certain type of consistency constraints.
   - Constraint example:

- Account.Balance > 0

- The integrity constraints are part of the program code.
- Hard to add new constraints or change the existing ones.

4. *Atomicity of updates Problems*

- ***Transaction*** is a sequence of database operations that has the atomicity property:
  - ***Atomicity*** means all operations in the sequence are executed, or no operation is executed. (Do all or nothing).
- 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.

  AccountX.Balance=AccountX.Balance-100
  AccountY.Balance=AccountY.Balance+100
- Difficult to guarantee the atomicity of the transactions in the file system.

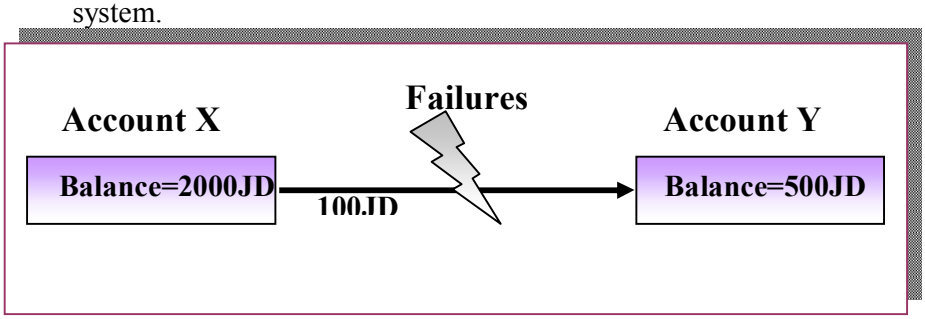

Figure 1.3: What are the net balances in Account X and Account Y?

5. *Concurrent access by multiple users Problem*

- Concurrent accessed needed for performance.
- Uncontrolled concurrent accesses can lead to inconsistencies.
- Why concurrent access to data must be managed?
  - James and Sali withdraw $50 and $100 from a common account

James	**Sali**
1. Get Balance 2. If Balance > 50JD 3. Balance = Balance – 50JD 4. Update Balance	1. Get Balance 2. If Balance > 100JD 3. Balance=Balance– 100JD 4. Update Balance

- Initial balance is $300. Final balance=? It depends …

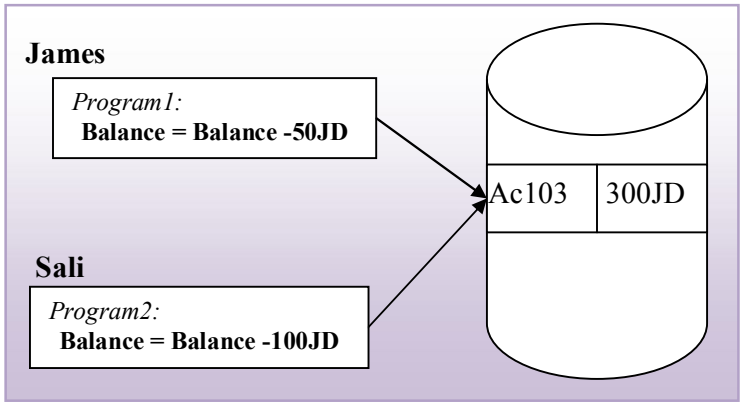

Figure 1.4: What is the final value of the balance?

- Difficult to control the concurrent access in the file system.

6. ***Security Problems***
   - Protection from unauthorized use.
   - Hard to provide user access to some, but not all, data.
   - Difficult to implement database security.

Database Systems offer solutions to all the above problems.

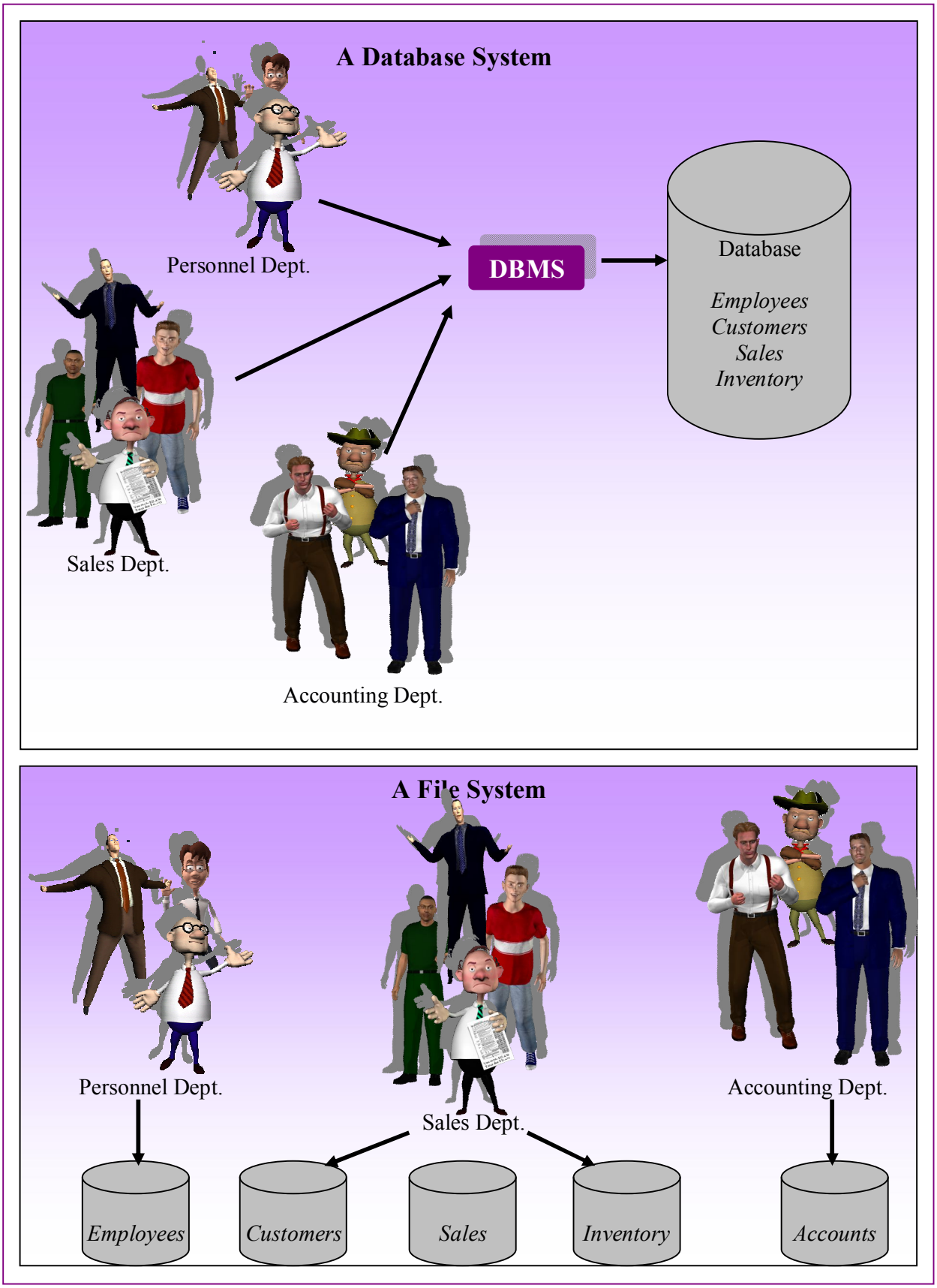

Figure 1.5 Database System VS. File System