INTRODUCTION TO DATABASES

A database is a shared collection of logically related data designed to meet the information requirements of an organisation

A database system is a computerized record keeping system. The database itself can be regarded as a kind of electronic filing cabinet for a collection of computerized data files. Users of the system have a variety of operations on such files including:

- Adding new files to the database
- Inserting data into existing files
- Retrieving data from existing files
- Deleting data from existing files
- Changing data in existing files
- Removing existing files from the database.

Efficient data management typically requires the use of a computer database. A **database** is a shared, integrated computer structure that houses a collection of:

- End user data, that is, raw facts of interest to the end user.
- **Metadata**, or data about data, through which the data are integrated and managed.

The metadata provide a description of the data characteristics and the set of relationships that link the data found within the database. In a sense, a database resembles a very well-organized electronic filing cabinet in which powerful software, known as a *database management system*, helps manage the cabinet's contents. A **database management system** (**DBMS**) is a collection of programs that manages the database structure and controls access to the data stored in the database. The DBMS makes it possible to share the data in the database among multiple applications or users.

File Based Systems

Involves a collection of application programs that perform services to end users. Each program defines and manages its own data. They are the historical roots of databases. Traditional file based systems are the predecessor to database systems, and there are good reasons for looking at it. Understanding the problems of file based systems may prevent us from repeating these problems in database systems. Also there are still many file based systems out there in the real world. One day you may be called upon the convert one. Understanding how the file system works is therefore very useful.

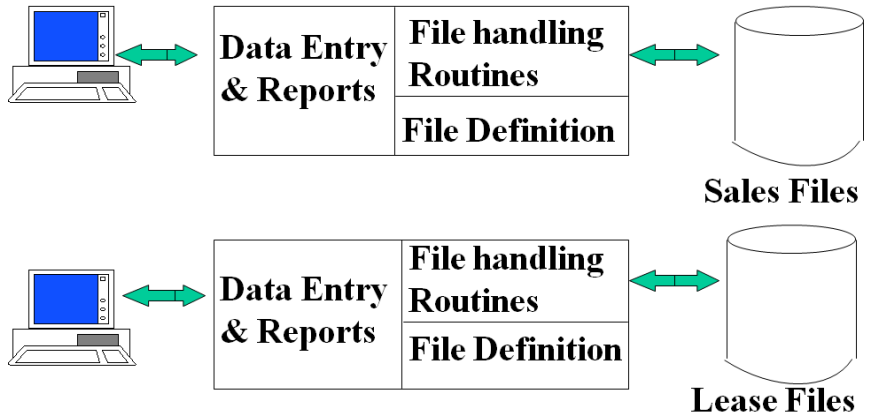
File systems were really an early attempt to computerize the manual systems that we are familiar with. The manual system works well while there are not many items, but it breaks down when we have a lot of items and we want to cross reference them. Take an example of

an estate agent, and the kind of information that might be required. e.g. What's the average rent in a particular area. What flats do you have in a 2 mile radius of the city centre etc. Nowadays everyone wants more and more information. So the file based system was a response to industry needs for more efficient data access. However, rather than have everything centralized, each department would have their own set of files.

Basic file terminologies

Data	"Raw" facts, such as a telephone number, a birth date, a customer name, and a year-to-date (YTD) sales value. Data have little meaning unless they have been organized in some logical manner. The smallest piece of data that can be "recognized" by the computer is a single character, such as the letter A, the number 5, or a symbol such as /. A single character requires one byte of computer storage.
Field	A character or group of characters (alphabetic or numeric) that has a specific meaning. A field is used to define and store data.
Record	A logically connected set of one or more fields that describes a person, place, or thing. For example, the fields that constitute a record for a customer named J.D. Rudd might consist of J.D. Rudd's name, address, phone number, date of birth, credit limit, and unpaid balance.
File	A collection of related records. For example, a file might contain data about vendors of ROBCOR Company; or a file might contain the records for the students currently enrolled at Gigantic University.

File Based Processing



Limitations of File Based Systems

Isolation -When data is kept in separate (and isolated) files it is much more difficult to get at data kept in several of them. The files have to be synchronized to obtain the data. This can be difficult especially if several files are needed.

Data dependence – in file based systems data definitions are kept in the programs. This makes changes difficult. For example suppose you need to change the size of an address field just by one character. It sounds simple. BUT

OPEN original file

OPEN a temporary file with the new structure

READ a record from the original file; convert the data to the new structure

WRITE it to the temporary file and REPEAT for all records

DELETE the original file because that's now old data

RENAME the temporary file so that it has the name of the original one.

Also all programs that use that data would need to be changed.

Obviously this is very time consuming and prone to error. This characteristic of file based systems is known as program-data dependence.

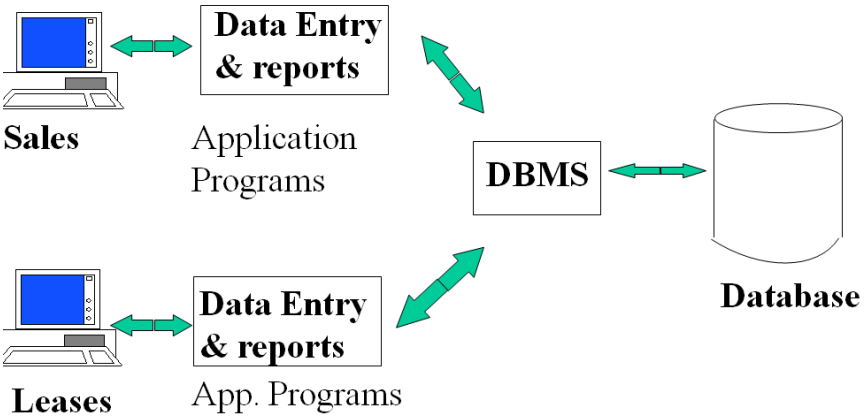
Duplication - Because data wasn't centralized, this led to duplication of data. Two points to note here really.

a) Duplication is expensive - it takes time and money to enter data more than once

b) It can lead to integrity problems. You always have to remember to change data in more than one place.

Incompatible file formats - As the data structures are embedded in the application programs they are dependent on a particular language. For example the structures in a COBOL file and in a C program may be quite different - so this can lead to conversion problems when you want them to run together.

The Database Approach: data processing



Database Management System (DBMS)

A software system that enables users to define, create and maintain the database providing controlled access to the database. The DBMS has a number of functions such as:

- Allow users to define the database i.e specify the data types and structures, and any constraints on the data. This is through the Data Definition language.
- Allows users to insert, update, delete & retrieve data (DML)
- Provides controlled access through:
  - a security system
  - an integrity system
  - a concurrency control system
  - a recovery system
  - a user accessible catalogue

Components of a DBMS

Hardware- the DBMS obviously requires hardware to be able to run. The hardware can range from PC's to networked computers. The hardware will depends on the needs of the organization.

Software- Here of course you have the DBMS software itself, but you also need the operating system, any network software required, and the application programs.

Data- This is probably the most important element of the DBMS. Data acts a bridge between the hardware and software components and the human component. The database contains both the operational data and the metadata.

Procedures- Procedures refer to the rules that govern the design & use of the database. They can be things like How to log on, how to use a facility, how to make backups, and how to handle hardware and software failures.

People-The people involved in the DB environment. The kind of roles involved here is DB administrators, DB designers, Application programmers and end users.

People who deal with databases

Many persons are involved in the design, use and maintenance of any database. They include:

Database Administrators (DBA)
The DBA is responsible for authorizing access to the database, for Coordinating and monitoring its use and for acquiring software and hardware resources as needed. These are the people, who maintain and design the database daily. DBA is responsible for the following issues.

a. Design of the conceptual and physical schemas:
   The DBA is responsible for interacting with the users of the system to understand what data is to be stored in the DBMS and how it is likely to be used.

The DBA creates the original schema by writing a set of definitions and is permanently stored in the 'Data Dictionary'.

b. Security and Authorization:
The DBA is responsible for ensuring the unauthorized data access is not permitted. The granting of different types of authorization allows the DBA to regulate which parts of the
database various users can access.

c. Storage structure and Access method definition:
The DBA creates appropriate storage structures and access methods by writing a set of definitions, which are translated by the DDL compiler.

d. Data Availability and Recovery from Failures:
The DBA must take steps to ensure that if the system fails, users can continue to access as
much of the uncorrupted data as possible. The DBA also work to restore the data to consistent state.

e. Database Tuning:
The DBA is responsible for modifying the database to ensure adequate Performance as requirements change.

f. Integrity Constraint Specification:
The integrity constraints are kept in a special system structure that is consulted by the DBA whenever an update takes place in the system.

Database Designers
Database designers are responsible for identifying the data to be stored in the database and for choosing appropriate structures to represent and store this data.

End Users
People who wish to store and use data in a database. End users are the people whose jobs require access to the database for querying, updating and generating reports, e.t.c

System Analyst
These people determine the requirements of end users and develop specifications for transactions.

Application Programmers (Software Engineers)
These people can test, debug, document and maintain the specified transactions.

DBMS Architectures

The design of a Database Management System highly depends on its architecture. It can be centralized or decentralized or hierarchical. DBMS architecture can be seen as single tier or multi-tier. n-tier architecture divides the whole system into related but independent n modules, which can be independently modified, altered, changed or replaced.

1- **tier (client-server) architecture**

In 1-tier architecture, DBMS is the only entity where user directly sits on DBMS and uses it. Any changes done here will directly be done on DBMS itself. It does not provide handy tools for end users and preferably database designer and programmers use single tier architecture.

Two tier (client-server) architecture

The DBMS is divided into two parts:

- A client program that handles the main business, data processing logic and interfaces with the user.
- A server program (DBMS engine) that manages and controls access to the database.

In the mid 1990s as applications became more complex and potentially could be deployed to hundreds or thousands of end users , the client side of this architecture gave rise to two problems.

- A fat client requiring considerable resources(disk space, RAM, CPU power) on the clients computer to run effectively.
- A significant client side administration overhead.

Three tier client server architecture

The architecture consistes three layers each potentially running on a different platform

- The user interface layer which runs on the end users computer (the client)
- The business logic and the data processing layer . this is the middle tier running on a server often reffered to as th eapplication server. One application server is designe dto serve multiple clients.
- A DBMS which stored the data required by the middle tier. This server may run on a separate server called the database server.

The three tier design has many advantages over the traditional two tier design

1. A thin client which requires less expensive hardware

2. Simplified application maintenance as a result of centralizing the business logic for many end users in a single appication server. This eliminates the concerns of software distribution that are problematic in the traditional two tier architecture.

3. Added modularity which makes it easierto modify or replace one tier without affecting the other tiers.

4. Easier load balancing as a result of separating the core business logic from the database functions. For instance a transaction processing monotor can be used to reduce the number of connections to the database server. A TPM is a program that controls data transfer between clients and servers in order to provide a consistent environment for online transaction processing(OLTP).

5. Three tier architecture maps quite naturally to the web environment with a web browser acting as the thin client and a web server acting as the application server.

Functions of DBMS

Data storage, retrieval, and update
This is the fundamental function of a DBMS. From our earlier discussion, clearly in providing this functionality the DBMS should hide the internal physical implementation details (such as file organization and storage structures) from the user.

A user-accessible catalog
A key feature of a DBMS is the provision of an integrated system catalog to hold data about the structure of the database, users, applications, and so on. The catalog is expected to be accessible to users as well as to the DBMS. The amount of information and the way the information is used vary with the DBMS.

Typically, the system catalog stores:

- Names, types, and sizes of data items;
- Integrity constraints on the data;
- Names of authorized users who have access to the data.

Concurrency control services
One major objective in using a DBMS is to enable many users to access shared data concurrently; this is known as concurrency control. Concurrent access is relatively easy if all users are only reading data, as there is no way that they can interfere with one another. However, when two or more users are accessing the database simultaneously and at least one of them is updating data, there may be interference that can result in inconsistencies.

Recovery services
When discussing transaction support, we mentioned that if the transaction fails the database has to be returned to a consistent state; this is known as recovery control. This may be the result of a system crash, media failure, a hardware or software error causing the DBMS to

stop, or it may be the result of the user detecting an error during the transaction and aborting the transaction before it completes. In all these cases, the DBMS must provide a mechanism to recover the database to a consistent state.

Authorization & security services
It's not difficult to envisage instances where we would want to protect some of the data stored in the database from being seen by all users. For example, we may want only branch managers and the Payroll Department to see salary related information for staff and prevent all other users from seeing this data. Additionally, we may want to protect the database from unauthorized access.

The term security refers to the protection of the database against unauthorized access, either intentional or accidental. We expect the DBMS to provide mechanisms to ensure the data is secure.

Support for data communication
Most users access the database from terminals. Sometimes, these terminals are connected directly to the computer hosting the DBMS. In other cases, the terminals are at remote locations and communicate with the computer hosting the DBMS over a network. In either case, the DBMS must be capable of integrating with networking/communication software. Even DBMSs for PCs should be capable of being run on a local area network (LAN) so that one centralized database can be established for users to share, rather than having a series of disparate databases, one for each user.

Integrity services
Database integrity refers to the correctness and consistency of stored data. It can be considered as another type of database protection. Integrity is concerned with the quality of data itself. Integrity is usually expressed in terms of constraints, which are consistency rules that the database is not permitted to violate.

Utility services

Utility programs help the DBA to manage the database effectively. Some examples of utilities are:

- Import facilities, to load the database from flat files, and export facilities, to unload the database to flat files;
- Monitoring facilities, to monitor database usage and operation.

Advantages of Using DBMS

Control of data redundancy
The database approach eliminates redundancy where possible. However, it does not eliminate redundancy entirely, but controls the amount of redundancy inherent in the database. For example, it's normally necessary to duplicate key data items to model relationships between data, and sometimes it's desirable to duplicate some data items to improve performance.

Data consistency
By eliminating or controlling redundancy, we're reducing the risk of inconsistencies occurring. If data is stored only once in the database, any update to its value has to be performed only once and the new value is immediately available to all users. If data is stored more than once and the system is aware of this, the system can ensure that all copies of the data are kept consistent.

Sharing of data
In a file-based approach (the predecessor to the DBMS approach), typically files are owned by the people or departments that use them. On the other hand, the database belongs to the entire organization and can be shared by all authorized users. In this way, more users share more of the data. Furthermore, new applications can build on the existing data in the database and add only data that is not currently stored, rather than having to define all data requirements again. The new applications can also rely on the functions provided by the DBMS, such as data definition and manipulation, and concurrency and recovery control, rather than having to provide these functions themselves.

Improved data integrity
Database integrity is usually expressed in terms of constraints, which are consistency rules that the database is not permitted to violate. Constraints may apply to data within a single record or they may apply to relationships between records. Data integration allows users to define, and the DBMS to enforce, integrity constraints.

Improved maintenance
Since a DBMS separates the data descriptions from the applications, it helps make applications immune to changes in the data descriptions. This is known as data independence and its provision simplifies database application maintenance.

Other advantages include: improved security, improved data accessibility and responsiveness, increased productivity, increased concurrency, and improved backup and recovery services.

DISADVANTAGES

Complexity
A DBMS is an extremely complex piece of software, and all users (database designers and developers, DBAs, and end-users) must understand the DBMS's functionality to take full advantage of it.

Cost of DBMS
The cost of DBMSs varies significantly, depending on the environment and functionality provided. For example, a single-user DBMS for a PC may cost only $100. However, a large mainframe multi-user DBMS servicing hundreds of users can be extremely expensive, perhaps $100,000 to $1,000,000. There is also the recurrent annual maintenance cost, which is typically a percentage of the list price.

Cost of conversion

In some situations, the cost of the DBMS and any extra hardware may be insignificant compared with the cost of converting existing applications to run on the new DBMS and hardware. This cost also includes the cost of training staff to use these new systems, and possibly the employment of specialist staff to help with the conversion and running of the system. This cost is one of the main reasons why some companies feel tied to their current systems and cannot switch to more modern database technology.

Higher impact of a failure

The centralization of resources increases the vulnerability of the system. Since all users and applications rely on the availability of the DBMS, the failure of any component can bring operations to a complete halt until the failure is repaired.