

AD8401 Database Design and Management

A database is an organized collection of data elements (facts) stored in a computer in a systematic way, such that a computer program can consult it to answer questions. The answers to those questions become information that can be used to make decisions that may not be made with the data elements alone. A software system that enables users to define, create, maintain, query, and control access to the database is known as a database management system (DBMS). A computer program that interacts with the database by issuing an appropriate request (typically an SQL statement) to the DBMS is known as a **database application program**.

Traditional File Processing System: File-based systems were an early attempt to computerize the manual filing system. However, rather than establish a centralized store for the organization's operational data, a decentralized approach was taken, where each department stored and controlled its own data.

Issues in File Based Systems:

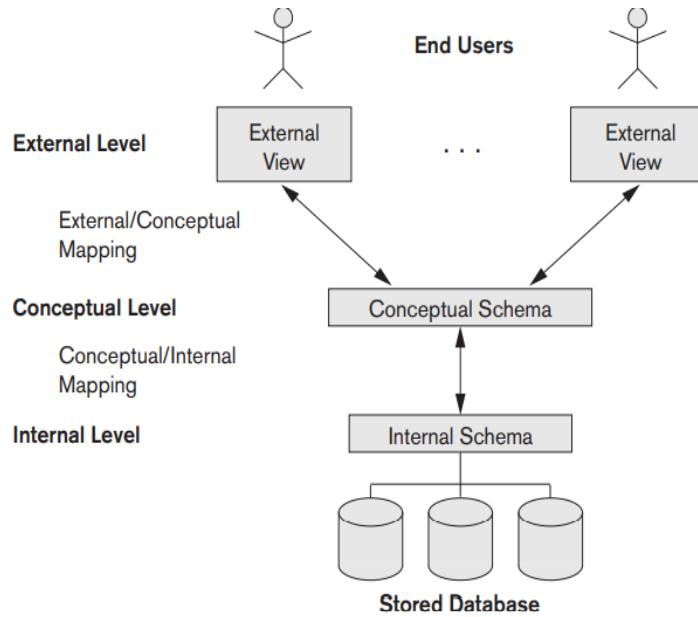
- Separation and isolation of data: When data is isolated in separate files, it is more difficult to access data that should be available.
- Duplication of data: Owing to the decentralized approach taken by each department, the file-based approach encouraged, if not necessitated, the uncontrolled duplication of data.
- Difficulty in accessing data In order to retrieve, access and use stored data, need to write a new program to carry out each new task.
- Program–Data dependence: The physical structure and storage of the data files and records are defined in the application code.
- Data Inconsistency: Data is updated in one file and not in other files and thus some data becomes invalid.
- Data Integrity: Allows some invalid data to enter into the system.
- Data Security: Everyone who has access to the file can view all the data.
- Atomicity of updates Failures of files may leave database in an inconsistent state with partial updates carried out.

Three Level Architecture (Abstraction):

External View: The external or view level includes a number of external schemas or user views. Each external schema describes the part of the database that a particular user group is interested in and hides the rest of the database from that user group.

Conceptual View: The conceptual level has a conceptual schema, which describes the structure of the whole database for a community of users. The conceptual schema hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations, and constraints

Physical View (Internal View): The internal level has an internal schema, which describes the physical storage structure of the database. The internal schema uses a physical data model and describes the complete details of data storage and access paths for the database.



Logical data independence is the capacity to change the conceptual schema without having to change external schemas or application programs. We may change the conceptual schema to expand the database (by adding a record type or data item), to change constraints, or to reduce the database (by removing a record type or data item).

Physical data independence is the capacity to change the internal schema without having to change the conceptual schema. Hence, the external schemas need not be changed as well. Changes to the internal schema may be needed because some physical files were reorganized—for example, by creating additional access structures—to improve the performance of retrieval or update.

Users of the Database:

Naive User: The end-users are the “clients” of the database. Naïve users are typically unaware of the DBMS. They access the database through specially written application programs that attempt to make the operations as simple as possible.

Application Programmer: Each program contains statements that request the DBMS to perform some operation on the database, which includes retrieving data, inserting, updating, and deleting data. The programs may be written in a third-generation or fourth-generation programming language.

Database Designer: The logical database designer must have a thorough and complete understanding of the organization’s data and any constraints on this data (the constraints are sometimes called business rules).

Database Administrator: The Database Administrator (DBA) is responsible for the physical realization of the database, including physical database design and implementation, security and integrity control, maintenance of the operational system, and ensuring satisfactory performance of the applications for users.

Advantages of DBMSs

Control of data redundancy: the database approach attempts to eliminate the redundancy by integrating the files so that multiple copies of the same data are not stored.

Data consistency: By eliminating or controlling redundancy, we reduce the risk of inconsistencies occurring. If a data item is stored only once in the database, any update to its value has to be performed only once and the new value is available immediately to all users.

More information from the same amount of data: With the integration of the operational data, it may be possible for the organization to derive additional information from the same data.

Sharing of data: 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.

Improved data integrity: 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 items within a single record or to relationships between records.

Database security is the protection of the database from unauthorized users. This security may take the form of usernames and passwords to identify people authorized to use the database. Granting restricted access to the users.

Enforcement of standards: integration allows the DBA to define and the DBMS to enforce the necessary standards.

Economy of scale: Combining all the organization's operational data into one database and creating a set of applications that work on this one source of data can result in cost savings.

Balance of conflicting requirements: Each user or department has needs that may be in conflict with the needs of other users.

Improved data accessibility and responsiveness: Again, as a result of integration, data that crosses departmental boundaries is directly accessible to the end users.

Increased productivity: As mentioned previously, the DBMS provides many of the standard functions that the programmer would normally have to write in a file based application. Eg: low-level file-handling routines

Improved maintenance through data independence: A DBMS separates the data descriptions from the applications, thereby making applications immune to changes in the data descriptions.

Increased concurrency: Many DBMSs manage concurrent database access and ensure that multiple users can access the data simultaneously.

Improved backup and recovery services: Modern DBMSs provide facilities to minimize the amount of processing that is lost following a failure.

Disadvantages of DBMSs

Complexity: The provision of the functionality that we expect of a good DBMS makes the DBMS an extremely complex piece of software.

Size: The complexity and breadth of functionality makes the DBMS an extremely large piece of software, occupying many megabytes of disk space and requiring substantial amounts of memory to run efficiently.

Cost: The cost of DBMSs varies significantly, depending on the environment and functionality provided. For example, a single-user DBMS costs less than a large mainframe multi-user DBMS. There is also the recurrent annual maintenance cost.

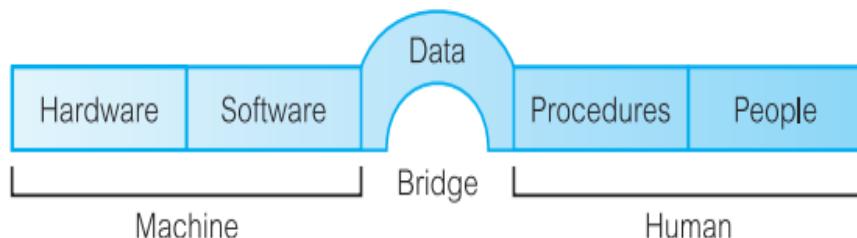
Additional hardware costs The disk storage requirements for the DBMS and the database may necessitate the purchase of additional storage space.

Cost of conversion: The cost of converting existing applications to run on the new DBMS and hardware.

Performance: DBMS is written to be more general, to cater for many applications rather than just one. The result is that some applications may not run as fast as they used to.

Greater impact of a failure The centralization of resources increases the vulnerability of the system. Because all users and applications rely on the availability of the DBMS, the failure of certain components can bring operations to a halt.

DBMS environment



The DBMS and the applications require hardware to run. The hardware can range from a single personal computer to a single mainframe or a network of computers. The particular hardware depends on the organization's requirements and the DBMS used.

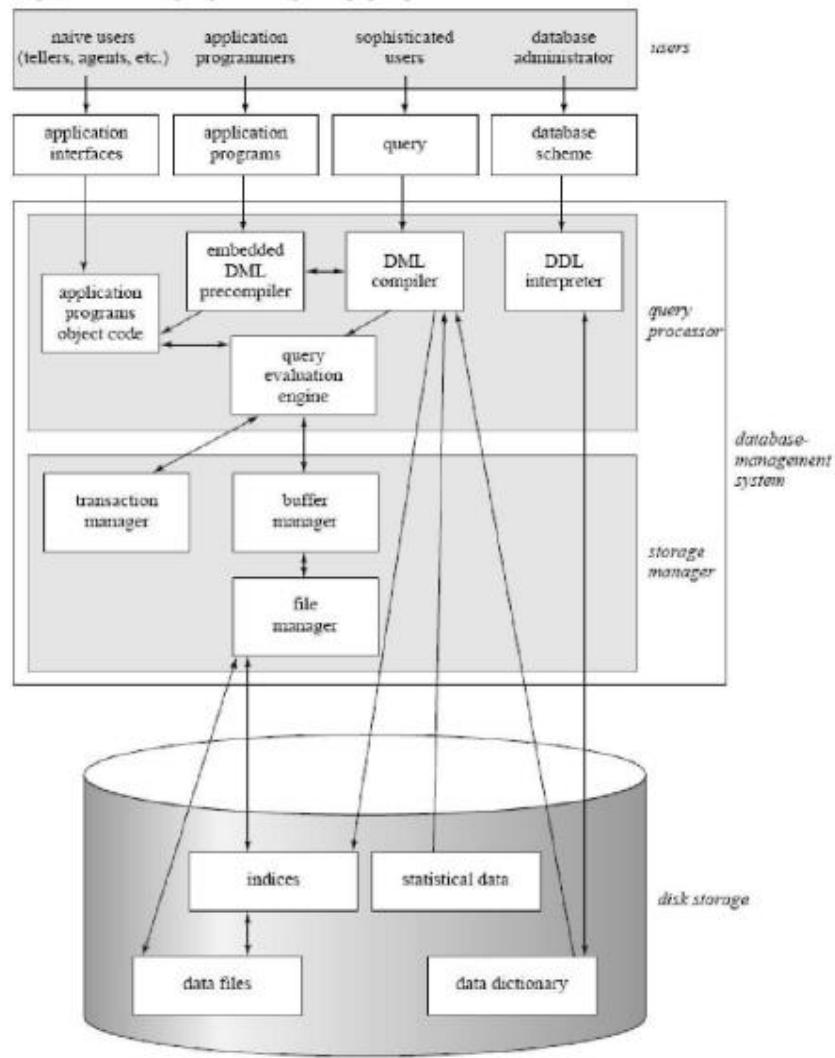
The software component comprises the DBMS software itself and the application programs, together with the operating system, including network software if the DBMS is being used over a network.

Data: Perhaps the most important component of the DBMS environment certainly from the end-users' point of view is the data. The data acts as a bridge between the machine and the human components.

Procedures: Procedures refer to the instructions and rules that govern the design and use of the database. The users of the system and the staff who manage the database require documented procedures on how to use or run the system.

People: The final component is the people involved with the system.

Database Architecture:



Data model: An integrated collection of concepts for describing and manipulating data, relationships between data, and constraints on the data in an organization. A model is a representation of real-world objects and events, and their associations.

A data model can be thought of as comprising three components:

- (1) a structural part, consisting of a set of rules according to which databases can be constructed;
- (2) a manipulative part, defining the types of operation that are allowed on the data (updating or retrieving data from the database and changing the structure of the database);
- (3) a set of integrity constraints, which ensures that the data is accurate.

Types of Data Model:

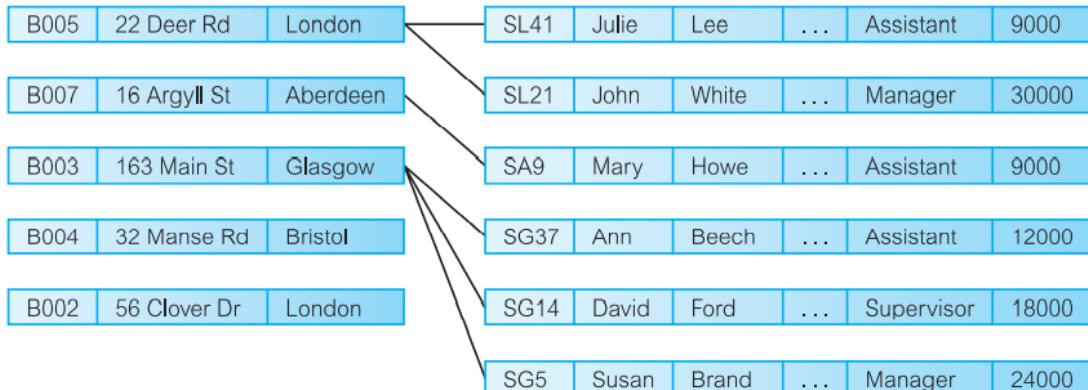
- (1) an external data model, to represent each user's view of the organization, sometimes called the Universe of Discourse (UoD);
- (2) a conceptual data model, to represent the logical (or community) view that is DBMS-independent;
- (3) an internal data model, to represent the conceptual schema in such a way that it can be understood by the DBMS

Object-based data models use concepts such as entities, attributes, and relationships.

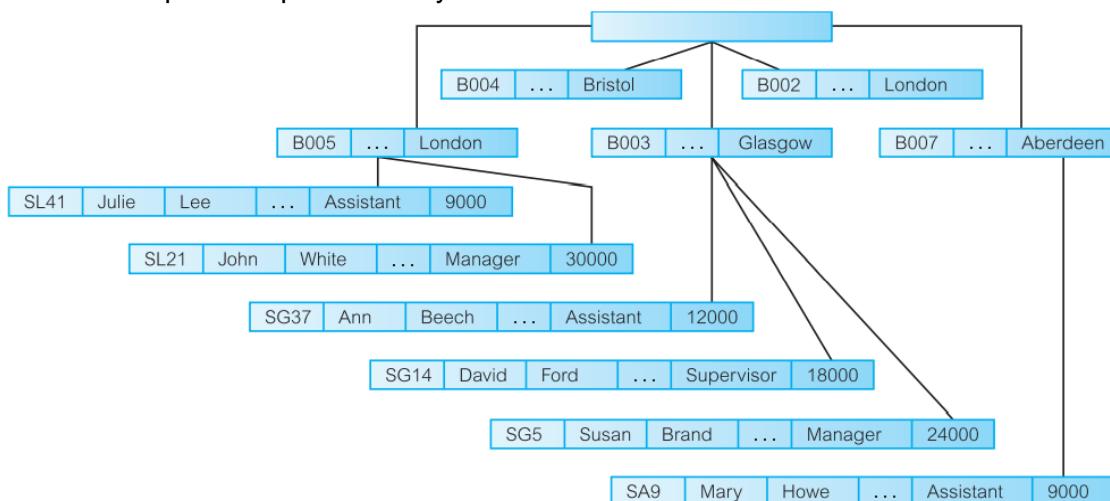
In a **record-based model**, the database consists of a number of fixed-format records, possibly of differing types. Each record type defines a fixed number of fields, typically of a fixed length. There are three principal types of record-based logical data model: the relational data model, the network data model, and the hierarchical data model.

The **relational data model** is based on the concept of mathematical relations. In the relational model, data and relationships are represented as tables, each of which has a number of columns with a unique name.

In the **network model**, data is represented as collections of records, and relationships are represented by sets.



The **hierarchical model** is a restricted type of network model. Again, data is represented as collections of records and relationships are represented by sets.



Physical data models describe how data is stored in the computer, representing information such as record structures, record orderings, and access paths. There are not as many physical data models as logical data models; the most common ones are the unifying model and the frame memory.

Conceptual modeling or conceptual database design is the process of constructing a model of the information used in an enterprise that is independent of implementation details, such as the target DBMS, application programs, programming languages, or any other physical considerations.

Functions of a DBMS

- (1) Data storage, retrieval, and update: A DBMS must furnish users with the ability to store, retrieve, and update data in the database
- (2) A user-accessible catalog: A DBMS must furnish a catalog in which descriptions of data items are stored and which is accessible to users.
- (3) Transaction support: A DBMS must furnish a mechanism that will ensure either that all the updates corresponding to a given transaction are made or that none of them is made
- (4) Concurrency control services: A DBMS must furnish a mechanism to ensure that the database is updated correctly when multiple users are updating the database concurrently.
- (5) Recovery services: A DBMS must furnish a mechanism for recovering the database in the event that the database is damaged in any way.
- (6) Authorization services: A DBMS must furnish a mechanism to ensure that only authorized users can access the database.
- (7) Support for data communication: A DBMS must be capable of integrating with communication software.
- (8) Integrity services: A DBMS must furnish a means to ensure that both the data in the database and changes to the data follow certain rules.
- (9) Services to promote data independence: A DBMS must include facilities to support the independence of programs from the actual structure of the database.
- (10) Utility services: A DBMS should provide a set of utility services like backup, restore, etc.