Functions of a DBMS

Data storage, retrieval, and update

- A DBMS must furnish users with the ability to store, retrieve, and update data in the database.
- This is the fundamental function of a DBMS 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 DBMS must furnish a catalog in which descriptions of data items are stored and which is accessible to users.
- A key feature of the ANSI-SPARC architecture is the recognition of an integrated **system catalog** to hold data about the schemas, users, applications, and so on.
- The catalog is expected to be accessible to users as well as to the DBMS.
- A system catalog, or data dictionary, is a repository of information describing the data in the database: it is, the 'data about the data' or **metadata**.
- 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;
    - names of relationships;
    - integrity constraints on the data;
    - names of authorized users who have access to the data;
    - the data items that each user can access and the types of access allowed; for example, insert, update, delete, or read access;
    - external, conceptual, and internal schemas and the mappings between the schemas
    - usage statistics, such as the frequencies of transactions and counts on the number of accesses made to objects in the database.

The DBMS system catalog is one of the fundamental components of the system. Many of the software components rely on the system catalog for information. Some benefits of a system catalog are:

- Information about data can be collected and stored centrally. This helps to maintain control over the data as a resource.
- The meaning of data can be defined, which will help other users understand the purpose of the data.
- Communication is simplified, since exact meanings are stored. The system catalog may also identify the user or users who own or access the data.
- Redundancy and inconsistencies can be identified more easily since the data is centralized.

- Changes to the database can be recorded.
- The impact of a change can be determined before it is implemented, since the system catalog records each data item, all its relationships, and all its users.
- Security can be enforced.
- Integrity can be ensured.
- Audit information can be provided.

Transaction support

- A DBMS must furnish a mechanism which will ensure either that all the updates corresponding to a given transaction are made or that none of them is made.
- A transaction is a series of actions, carried out by a single user or application program, which accesses or changes the contents of the database.

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.
- One major objective in using a DBMS is to enable many users to access shared data concurrently.
- 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.
- The DBMS must ensure that, when multiple users are accessing the database, interference cannot occur.

Recovery services

- A DBMS must furnish a mechanism for recovering the database in the event that the database is damaged in any way.

Authorization services

- A DBMS must furnish a mechanism to ensure that only authorized users can access the database.

Support for data communication

- A DBMS must be capable of integrating with communication software.
- Most users access the database from workstations.
- Sometimes these workstations are connected directly to the computer hosting the DBMS.
- In other cases, the workstations are at remote locations and communicate with the computer hosting the DBMS over a network.

- In either case, the DBMS receives requests as **communications messages** and responds in a similar way.
- All such transmissions are handled by a Data Communication Manager (DCM).
- Although the DCM is not part of the DBMS, it is necessary for the DBMS to be capable of being integrated with a variety of DCMs if the system is to be commercially viable.

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.
- Database integrity refers to the correctness and consistency of stored data: it can be considered as another type of database protection.
- While integrity is related to security, it has wider implications: 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.

Services to promote data independence

- A DBMS must include facilities to support the independence of programs from the actual structure of the database.
- Data independence is normally achieved through a view or subschema mechanism.
- Physical data independence is easier to achieve: there are usually several types of change that can be made to the physical characteristics of the database without affecting the views.

Utility services

- A DBMS should provide a set of utility services.
- Utility programs help the DBA to administer the database effectively.
- Some utilities work at the external level, and consequently can be produced by the DBA.
- Other utilities work at the internal level and can be provided only by the DBMS vendor.
- Examples of utilities of the latter kind 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;
  - statistical analysis programs, to examine performance or usage statistics;
  - index reorganization facilities, to reorganize indexes and their overflows;
  - garbage collection and reallocation, to remove deleted records physically from the storage devices, to consolidate the space released, and to reallocate it where it is needed.

Database Languages

- A **data sublanguage** consists of two parts: a **Data Definition Language** (DDL) and a **Data Manipulation Language** (DML).
- The DDL is used to specify the database schema and the DML is used to both read and update the database.
- These languages are called *data sublanguages* because they do not include constructs for all computing needs such as conditional or iterative statements, which are provided by the high-level programming languages.
- Many DBMSs have a facility for *embedding* the sublanguage in a high-level programming language such as 'C', C++, Java, or Visual Basic.

The Data Definition Language (DDL)

- Is a language that allows the DBA or user to describe and name the entities, attributes, and relationships required for the application, together with any associated integrity and security constraints.
- The database schema is specified by a set of definitions expressed by means of a special language called a Data Definition Language.
- The DDL is used to define a schema or to modify an existing one. It cannot be used to manipulate data.
- The result of the compilation of the DDL statements is a set of tables stored in special files collectively called the **system catalog**.
- The system catalog integrates the **metadata**, that is data that describes objects in the database and makes it easier for those objects to be accessed or manipulated.
- The metadata contains definitions of records, data items, and other objects that are of interest to users or are required by the DBMS. The DBMS normally consults the system catalog before the actual data is accessed in the database.
- At a theoretical level, we could identify different DDLs for each schema in the three level architecture, namely a DDL for the external schemas, a DDL for the conceptual schema, and a DDL for the internal schema.
- However, in practice, there is one comprehensive DDL that allows specification of at least the external and conceptual schemas.

The Data Manipulation Language (DML)

A language that provides a set of operations to support the basic data manipulation operations on the data held in the database.

Data manipulation operations usually include the following:

- insertion of new data into the database;
- modification of data stored in the database;
- retrieval of data contained in the database;

- deletion of data from the database.

Therefore, one of the main functions of the DBMS is to support a data manipulation language in which the user can construct statements that will cause such data manipulation to occur.

Fourth-Generation Languages (4GLs)

Fourth generation languages encompass:

- presentation languages, such as query languages and report generators;
- specialty languages, such as spreadsheets and database languages;
- application generators that define, insert, update, and retrieve data from the database to build applications;
- very high-level languages that are used to generate application code.

Components of a DBMS

- DBMSs are highly complex and sophisticated pieces of software that aim to provide the services.
- It is not possible to generalize the component structure of a DBMS as it varies greatly from system to system.
- However, it is useful when trying to understand database systems to try to view the components and the relationships between them.
- DBMS is partitioned into several software components (or *modules*), each of which is assigned a specific operation.
- As stated previously, some of the functions of the DBMS are supported by the underlying operating system.
- However, the operating system provides only basic services and the DBMS must be built on top of it.
- Thus, the design of a DBMS must take into account the interface between the DBMS and the operating system.

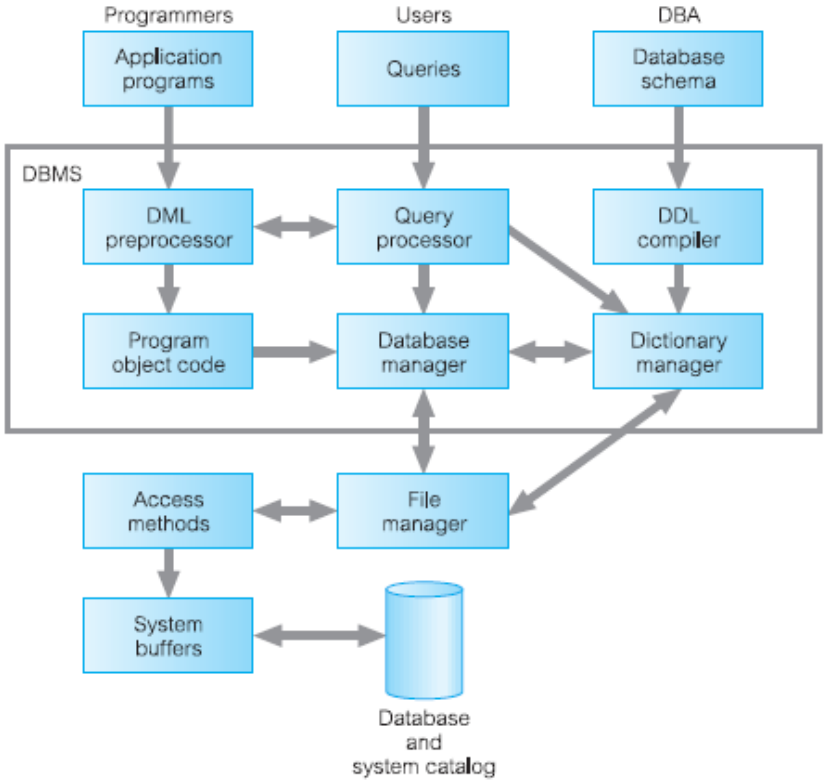


Figure 2.8
Major components of a DBMS.

- *Query processor* This is a major DBMS component that transforms queries into a series of low-level instructions directed to the database manager.

- *Database manager (DM)* The DM interfaces with user-submitted application programs and queries. The DM accepts queries and examines the external and conceptual schemas to determine what conceptual records are required to satisfy the request. The DM then places a call to the file manager to perform the request.

- *File manager:* The file manager manipulates the underlying storage files and manages the allocation of storage space on disk. It establishes and maintains the list of structures and indexes defined in the internal schema. If hashed files are used it calls on the hashing functions to generate record addresses.

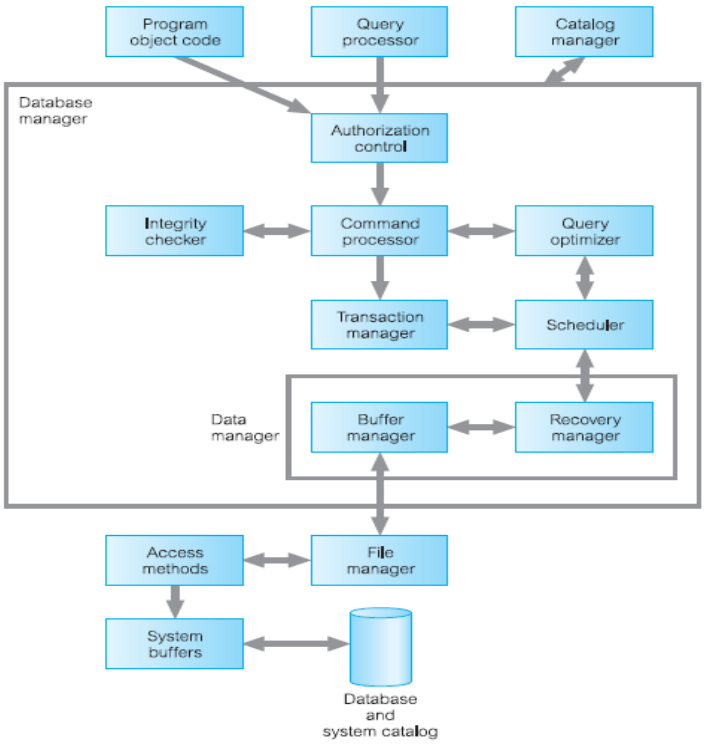
- However, the file manager does not directly manage the physical input and output of data. Rather it passes the requests on to the appropriate access methods, which either read data from or write data into the system buffer (or *cache*).

- *DML preprocessor* This module converts DML statements embedded in an application program into standard function calls in the host language. The DML preprocessor must interact with the query processor to generate the appropriate code.

- *DDL compiler* The DDL compiler converts DDL statements into a set of tables containing metadata. These tables are then stored in the system catalog while control information is stored in data file headers.

- *Catalog manager:* The catalog manager manages access to and maintains the system catalog. The system catalog is accessed by most DBMS components.

Figure 2.9
Components of a database manager.



The major software components for the *database manager* are as follows:

- *Authorization control: This* module checks that the user has the necessary authorization to carry out the required operation.
- *Command processor: Once* the system has checked that the user has authority to carry out the operation, control is passed to the command processor.
- *Integrity checker:* For an operation that changes the database, the integrity checker checks that the requested operation satisfies all necessary integrity constraints (such as key constraints).

- *Query optimizer* This module determines an optimal strategy for the query execution.
- *Transaction manager* This module performs the required processing of operations it receives from transactions.
- *Scheduler* This module is responsible for ensuring that concurrent operations on the database proceed without conflicting with one another. It controls the relative order in which transaction operations are executed.
- *Recovery manager* This module ensures that the database remains in a consistent state in the presence of failures. It is responsible for transaction commit and abort.
- *Buffer manager* This module is responsible for the transfer of data between main memory and secondary storage, such as disk and tape.
- The recovery manager and the buffer manager are sometimes referred to collectively as the *data manager*. The buffer manager is sometimes known as the *cache manager*.