

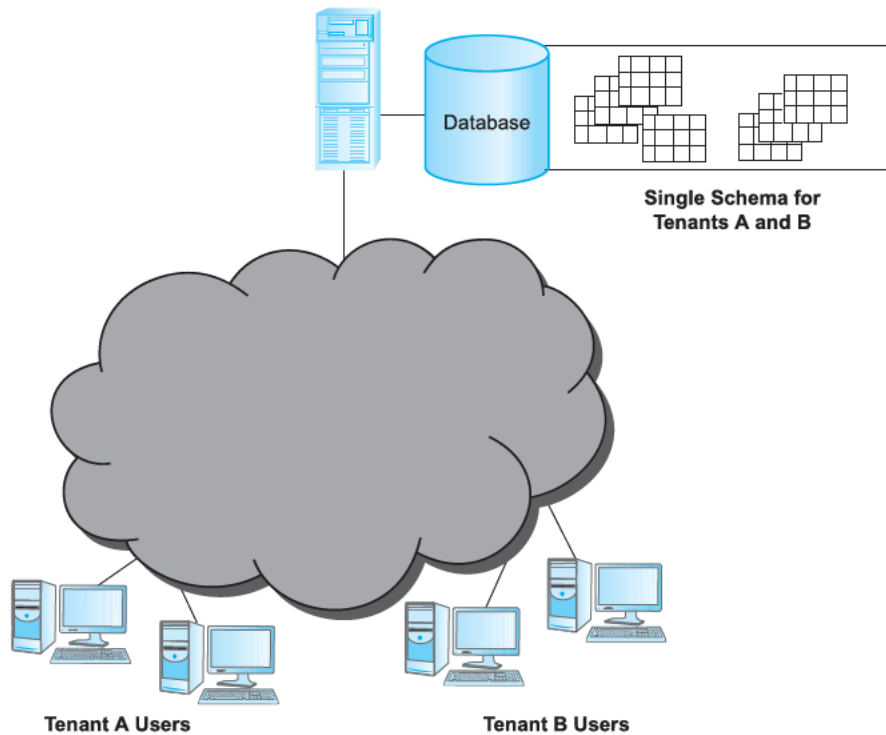


Figure 3.19 Multi-tenant cloud database—shared schema architecture.

3.6 Components of a DBMS

DBMSs are highly complex and sophisticated pieces of software that aim to provide the services discussed in Section 2.4. 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. In this section, we present a possible architecture for a DBMS. We examine the architecture of the Oracle DBMS in the next section.

A 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.

The major software components in a DBMS environment are depicted in Figure 3.20. This diagram shows how the DBMS interfaces with other software components, such as user queries and access methods (file management techniques for storing and retrieving data records). We will provide an overview of file organizations and access methods in Appendix F. For a more comprehensive treatment, the interested reader is referred to Teorey and Fry (1982), Weiderhold (1983), Smith and Barnes (1987), and Ullman (1988).

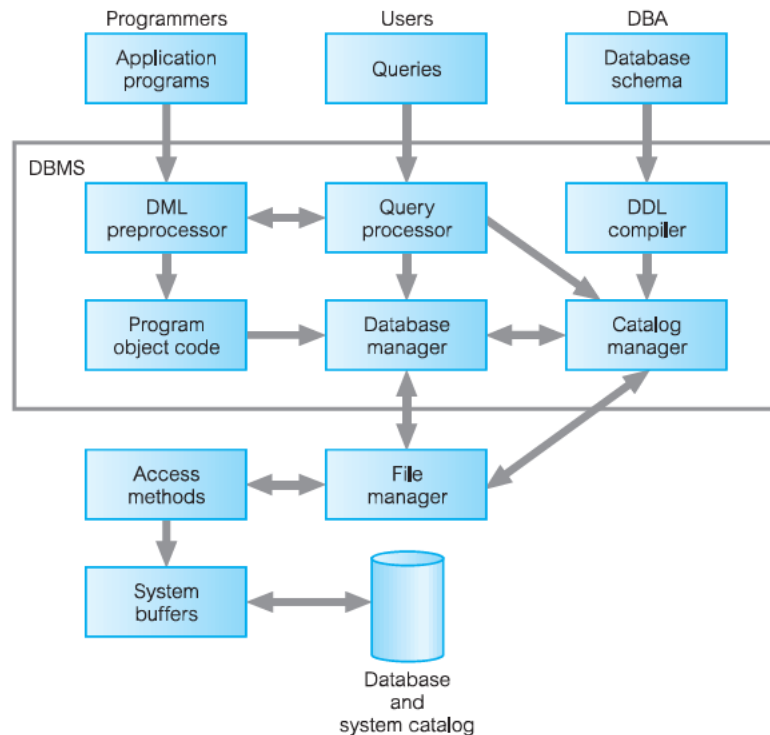
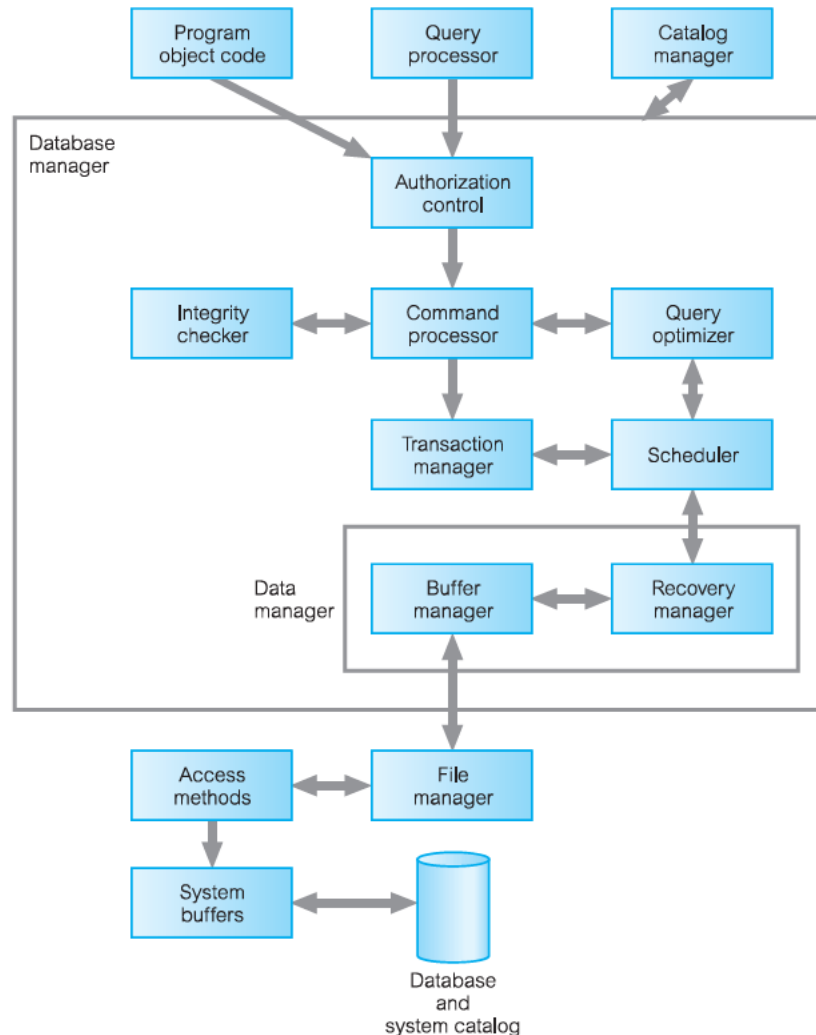


Figure 3.20
Major
components
of a DBMS.

Figure 3.20 shows the following components:

- *Query processor.* This is a major DBMS component that transforms queries into a series of low-level instructions directed to the database manager. We discuss query processing in Chapter 23.
- *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. The components of the DM are shown in Figure 3.21.
- *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.

Figure 3.21
Components
of a database
manager.



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

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

- *Authorization control.* This module confirms whether the user has the necessary authorization to carry out the required operation.
- *Command processor.* Once the system has confirmed 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 whether the requested operation satisfies all necessary integrity constraints (such as key constraints).
- *Query optimizer.* This module determines an optimal strategy for the query execution. We discuss query optimization in Chapter 23.

- *Transaction manager.* This module performs the required processing of operations that 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*.

We discuss the last four modules in Chapter 22. In addition to the previously mentioned modules, several other data structures are required as part of the physical-level implementation. These structures include data and index files and the system catalog. An attempt has been made to standardize DBMSs, and a reference model was proposed by the Database Architecture Framework Task Group (DAFTG, 1986). The purpose of this reference model was to define a conceptual framework aiming to divide standardization attempts into manageable pieces and to show at a very broad level how these pieces could be interrelated.

