buffering of disk pages. The DBMS also interfaces with compilers for general-purpose host programming languages, and with application servers and client programs running on separate machines through the system network interface.

2.4.2 Database System Utilities

In addition to possessing the software modules just described, most DBMSs have **database utilities** that help the DBA manage the database system. Common utilities have the following types of functions:

- **Loading.** A loading utility is used to load existing data files—such as text files or sequential files—into the database. Usually, the current (source) format of the data file and the desired (target) database file structure are specified to the utility, which then automatically reformats the data and stores it in the database. With the proliferation of DBMSs, transferring data from one DBMS to another is becoming common in many organizations. Some vendors offer **conversion tools** that generate the appropriate loading programs, given the existing source and target database storage descriptions (internal schemas).
- **Backup.** A backup utility creates a backup copy of the database, usually by dumping the entire database onto tape or other mass storage medium. The backup copy can be used to restore the database in case of catastrophic disk failure. Incremental backups are also often used, where only changes since the previous backup are recorded. Incremental backup is more complex, but saves storage space.
- **Database storage reorganization.** This utility can be used to reorganize a set of database files into different file organizations and create new access paths to improve performance.
- **Performance monitoring.** Such a utility monitors database usage and provides statistics to the DBA. The DBA uses the statistics in making decisions such as whether or not to reorganize files or whether to add or drop indexes to improve performance.

Other utilities may be available for sorting files, handling data compression, monitoring access by users, interfacing with the network, and performing other functions.

2.4.3 Tools, Application Environments, and Communications Facilities

Other tools are often available to database designers, users, and the DBMS. CASE tools[12] are used in the design phase of database systems. Another tool that can be quite useful in large organizations is an expanded **data dictionary** (or **data repository**)

---

12. Although CASE stands for computer-aided software engineering, many CASE tools are used primarily for database design.

accounts for the front-end modules (clients) communicating with a number of back-end databases (servers).

Advances in encryption and decryption technology make it safer to transfer sensitive data from server to client in encrypted form, where it will be decrypted. The latter can be done by the hardware or by advanced software. This technology gives higher levels of data security, but the network security issues remain a major concern. Various technologies for data compression also help to transfer large amounts of data from servers to clients over wired and wireless networks.

2.6 Classification of Database Management Systems

Several criteria can be used to classify DBMSs. The first is the **data model** on which the DBMS is based. The main data model used in many current commercial DBMSs is the **relational data model**, and the systems based on this model are known as **SQL systems.** The **object data model** has been implemented in some commercial systems but has not had widespread use. Recently, so-called **big data systems,** also known as **key-value storage systems** and **NOSQL systems,** use various data models: **document-based, graph-based, column-based,** and **key-value data models.** Many legacy applications still run on database systems based on the **hierarchical** and **network data models.**

The relational DBMSs are evolving continuously, and, in particular, have been incorporating many of the concepts that were developed in object databases. This has led to a new class of DBMSs called **object-relational DBMSs.** We can categorize DBMSs based on the data model: relational, object, object-relational, NOSQL, key-value, hierarchical, network, and other.

Some experimental DBMSs are based on the XML (eXtended Markup Language) model, which is a **tree-structured data model.** These have been called **native XML DBMSs.** Several commercial relational DBMSs have added XML interfaces and storage to their products.

The second criterion used to classify DBMSs is the **number of users** supported by the system. **Single-user systems** support only one user at a time and are mostly used with PCs. **Multiuser systems,** which include the majority of DBMSs, support concurrent multiple users.

The third criterion is the **number of sites** over which the database is distributed. A DBMS is **centralized** if the data is stored at a single computer site. A centralized DBMS can support multiple users, but the DBMS and the database reside totally at a single computer site. A **distributed** DBMS (DDBMS) can have the actual database and DBMS software distributed over many sites connected by a computer network. Big data systems are often massively distributed, with hundreds of sites. The data is often replicated on multiple sites so that failure of a site will not make some data unavailable.

each class are specified in terms of predefined procedures called **methods**. Relational DBMSs have been extending their models to incorporate object database concepts and other capabilities; these systems are referred to as **object-relational** or **extended relational systems**. We discuss object databases and object-relational systems in Chapter 12.

Big data systems are based on various data models, with the following four data models most common. The **key-value data model** associates a unique key with each value (which can be a record or object) and provides very fast access to a value given its key. The **document data model** is based on JSON (Java Script Object Notation) and stores the data as documents, which somewhat resemble complex objects. The **graph data model** stores objects as graph nodes and relationships among objects as directed graph edges. Finally, the **column-based data models** store the columns of rows clustered on disk pages for fast access and allow multiple versions of the data. We will discuss some of these in more detail in Chapter 24.

The **XML model** has emerged as a standard for exchanging data over the Web and has been used as a basis for implementing several prototype native XML systems. XML uses hierarchical tree structures. It combines database concepts with concepts from document representation models. Data is represented as elements; with the use of tags, data can be nested to create complex tree structures. This model conceptually resembles the object model but uses different terminology. XML capabilities have been added to many commercial DBMS products. We present an overview of XML in Chapter 13.

Two older, historically important data models, now known as **legacy data models**, are the network and hierarchical models. The **network model** represents data as record types and also represents a limited type of 1:N relationship, called a **set type**. A 1:N, or one-to-many, relationship relates one instance of a record to many record instances using some pointer linking mechanism in these models. The network model, also known as the CODASYL DBTG model,[14] has an associated record-at-a-time language that must be embedded in a host programming language. The network DML was proposed in the 1971 Database Task Group (DBTG) Report as an extension of the COBOL language.

The **hierarchical model** represents data as hierarchical tree structures. Each hierarchy represents a number of related records. There is no standard language for the hierarchical model. A popular hierarchical DML is DL/1 of the IMS system. It dominated the DBMS market for over 20 years between 1965 and 1985. Its DML, called DL/1, was a de facto industry standard for a long time.[15]

---

[14]CODASYL DBTG stands for Conference on Data Systems Languages Database Task Group, which is the committee that specified the network model and its language.

[15]The full chapters on the network and hierarchical models from the second edition of this book are available from this book's Companion Web site at http://www.aw.com/elmasri.