**Research On Database** Man**agement System**

A database management system (or DBMS) is essentially nothing more than **a computerized data-keeping system**. Users of the system are given facilities to perform several kinds of operations on such a system for either manipulation of the data in the database or the management of the database structure itself.

A database management system (DBMS) is system software for creating and managing databases. A DBMS makes it possible for end users to create, protect, read, update and delete data in a database. The most prevalent type of data management platform, the DBMS essentially serves as an interface between databases and end users or application programs, ensuring that data is consistently organized and remains easily accessible.

What Does Database Management System (DBMS) Mean?

A database management system (DBMS) is a software package designed to define, manipulate, retrieve and manage data in a database. A DBMS generally manipulates the data itself, the data format, field names, record structure and file structure. It also defines rules to validate and manipulate this data.

Database management systems are set up on specific data handling concepts, as the practice of administrating a database evolves. The earliest databases only handled individual single pieces of specially formatted data. Today’s more evolved systems can handle different kinds of less formatted data and tie them together in more elaborate ways.

Or

The DBMS manages the data; the database engine allows data to be accessed, locked and modified; and the database [schema](https://searchsqlserver.techtarget.com/definition/schema) defines the database's logical structure. These three foundational elements help provide concurrency, security, data integrity and uniform data administration procedures. Typical database administration tasks the DBMS supports include change management, performance monitoring and tuning, security, and backup and recovery. Most database management systems are also responsible for automated rollbacks and restarts as well as logging and auditing of activity in databases and the applications that access them.

The DBMS provides a centralized view of data that can be accessed by multiple users, from multiple locations, in a controlled manner. A DBMS can limit what data the end user sees, as well as how that end user can view the data, providing many views of a single database schema. End users and software programs are free from having to understand where the data is physically located or on what type of storage media it resides because the DBMS handles all requests.

The DBMS can offer both logical and physical data independence. This means it can protect users and applications from needing to know where data is stored or being concerned about changes to the physical structure of data. As long as programs use the application programming interface (API) for the database that the DBMS provides, developers won't have to modify programs just because changes have been made to the database.

## Techopedia Explains Database Management System (DBMS)

Over time, the models for database management systems have changed considerably. This is a key part of understanding how various DBMS options work.

The earliest types of database management systems consisted mainly of hierarchy and network models.

* The **hierarchy model** is one where each node or component has a child/parent relationship with one other node or component.
* In the **network model**, the difference is that a single component can have multiple relationships – think of this as a single node being able to “multicast” connections.

However, over time, these models became overtaken by something called a relational database. In the relational database model, individual components have attributes that are linked to their identities through a database table design. The rows and columns of an individual database table include those identities and attributes in such a way that traditional structured query language or SQL can be used to pull various kinds of information on these relational models.

Since then, an even newer concept has emerged called NoSQL. Experts suggest that the best way to understand NoSQL is to translate it to mean “not only SQL,” or in other words, using NoSQL broadly to describe systems that are beyond the traditional SQL and relational database models.

It's also important to note that NoSQL is a much more abstract term than the traditional "relational database." In a sense, NoSQL is “not relational,” at least in the traditional sense. One prominent type of NoSQL DBMS is called the object-oriented database model. Here, instead of being composed of relational tables, database systems use object designs to work with the identities and attributes discussed above.

Some of the considerations for NoSQL database design involve the degree of normalization or structuring of data that occurs, and how the database system handles that. Engineers also have to look at tools for consistency and resolution of data throughout the system, to promote uniformity and fix various problems of correlation.

Other types of DBMS models include a graph database model, where graph models are used for semantic queries, and an entity-relational model. These offer further alternatives to traditional relational database design.

Some of the newest types of DBMS can be used where a data center may have a wide disparity of differently formatted or relatively unformatted or “raw” data to work with, where records are not normalized in the conventional way. This and other types of advances have made the world of the DBMS more complex, and have heightened the value of seasoned DB engineers and administrators for modern systems.

### What are the components of a DBMS?

A DBMS is a sophisticated piece of system software that consists of multiple integrated components that deliver a consistent, managed environment for creating, accessing and modifying data in databases. These components include the following:

* **The structure of a DBMS**
* **Storage engine.** This basic element of a DBMS is used to store data. The DBMS must interface with a file system at the operating system (OS) level to store data. It can use additional components to store data or interface with the actual data at the file system level.
* **Metadata catalog.** Sometimes called a system catalog or database dictionary, a metadata catalog functions as a repository of all the database objects that have been created. When databases and other objects are created, the DBMS automatically registers information about it in the metadata catalog. The DBMS uses this catalog to verify user requests for data, and users can query the catalog for information about the database structures that exist in the DBMS. The metadata catalog can include information about database objects, schemas, programs, security, performance, communication and other environmental details about the databases it manages.
* **Database access language.** The DBMS also must provide an API to access the data, typically in the form of a database access language. This is used to access and modify data but may also be used to create database objects and to secure and authorize access to the data. SQL is an example of a database access language.
* **Optimization engine.** A DBMS may also provide an optimization engine, which is used to parse database access language requests and turn them into actionable commands for accessing and modifying data.
* **Lock manager.**This crucial component of the DBMS manages concurrent access to the same data. Locks are required to ensure multiple users aren't trying to modify the same data simultaneously.
* **Log manager.** The DBMS records all changes made to data managed by the DBMS. The record of changes is known as the log, and the [log manager](https://www.techtarget.com/searchitoperations/definition/log-management) component of the DBMS is used to ensure that log records are made efficiently and accurately. The DBMS uses the log manager during shutdown and startup to ensure data integrity, and it interfaces with database utilities to create backups and run recoveries.
* **Data utilities.** A DBMS also provides a set of utilities for managing and controlling database activities. Examples of database utilities include reorganization, runstats, backup and copy, recover, integrity check, load data, unload data and repair database.

### Popular types and examples of DBMS technologies

Popular [database models](https://www.techtarget.com/searchdatamanagement/feature/How-to-choose-the-right-database-to-fit-your-data-models) and management systems include the following:

* A RDBMS is adaptable to most use cases, but RDBMS [Tier-1](https://www.techtarget.com/searchitchannel/definition/tier-1-vendor) products can be quite expensive.
* A [NoSQL DBMS](https://www.techtarget.com/searchdatamanagement/definition/NoSQL-Not-Only-SQL) is well-suited for loosely defined data structures that may evolve over time, but may require more application involvement for schema management.
* An in-memory database management system (IMDBMS) provides faster response times and better performance but can consume more resources.
* A columnar database management system (CDBMS) is well-suited for [data warehouses](https://www.techtarget.com/searchdatamanagement/definition/data-warehouse) that have a large number of similar data items.
* A cloud-based database management system is built and accessed through the cloud, and the cloud serviceprovider is responsible for providing and maintaining the DBMS.

### Benefits of using a DBMS

One of the biggest advantages of using a DBMS is that it lets end users and application programmers access and use the same data concurrently while managing data integrity. Data is better protected and maintained when it can be shared using a DBMS instead of creating new iterations of the same data stored in new files for every new application. The DBMS provides a central store of data that multiple users can access in a controlled manner.

Central storage and management of data within the DBMS provides the following:

* data abstraction and independence;
* [data security](https://www.techtarget.com/searchsecurity/tip/Data-security-considerations-for-the-modern-day-enterprise);
* a locking mechanism for concurrent access;
* an efficient handler to balance the needs of multiple applications using the same data;
* the ability to swiftly recover from crashes and errors;
* strong [data integrity](https://www.techtarget.com/searchdatamanagement/feature/Maintaining-data-integrity-key-for-data-quality) capabilities;
* logging and auditing of activity;
* simple access using a standard API; and
* uniform administration procedures for data.

Another advantage of a DBMS is that database administrators ([DBAs](https://searchsqlserver.techtarget.com/definition/database-administrator)) can use it to impose a logical, structured organization on the data. A DBMS delivers economy of scale for processing large amounts of data because it is optimized for such operations.

A DBMS can also provide many views of a single database schema. A view defines what data the user sees and how that user sees the data. The DBMS provides a level of abstraction between the conceptual schema that defines the logical structure of the database and the physical schema that describes the files, indexes and other physical mechanisms the database uses. A DBMS enables users to modify systems much more easily when business requirements change. A DBA can add new categories of data to the database without disrupting the existing system, thereby insulating applications from how data is structured and stored.

However, a DBMS must perform additional work to provide these advantages, thereby bringing with it the overhead. A DBMS will use more memory and CPU than a simple file storage system, and different types of DBMSes will require different types and levels of system resources.

### Drawbacks of DBMSes

Perhaps the single biggest drawback is the cost of the hardware, software and personnel required to run an enterprise DBMS such as SQL Server, Oracle or [IBM Db2](https://www.techtarget.com/searchdatamanagement/news/252492597/IBM-to-deliver-refurbished-Db2-for-the-AI-and-cloud-era). The hardware is usually a high-end server with a significant amount of memory configured, coupled with large disk arrays to store the data. The software includes the DBMS itself, which is pricey, as well as tools for programming and testing, and for DBAs to enable management, tuning and administration.

From a personnel perspective, using a DBMS requires hiring a DBA staff, training developers in the proper usage of the DBMS, and possibly hiring additional systems programmers for managing installation and [integrating the DBMS into the IT infrastructure](https://www.techtarget.com/searchdatamanagement/feature/5-data-integration-challenges-and-how-to-overcome-them). Dealing with additional complexity is also a concern when implementing a DBMS.

The DBMS itself is a complex piece of software that requires in-depth knowledge to properly implement and manage. But the DBMS interfaces with many other IT components, such as the OS, transaction processing systems, programming languages and networking software. Ensuring the proper configuration and efficiency of such a complicated setup can be difficult and cause performance slowdowns or even system outages.

### DBMS use cases

Any application requiring a large amount of data that needs to be accessed by multiple users or customers is a candidate for using a DBMS. Most medium to large organizations can benefit from using a DBMS because they have more data-sharing and concurrency needs and are able to more readily overcome cost and complexity issues.

Sample customer use cases for DBMS technology include the following:

* Most commercial airlines rely on a DBMS for data-intensive applications such as scheduling flight plans and managing customer flight reservations.
* Applications include storing customer information, account information, tracking account transactions -- including withdrawals and deposits -- and tracking loan payments. ATMs are a good example of a banking system that relies on a DBMS to track and manage that activity.
* **Manufacturing and supply-chain management.**Manufacturing companies also rely on a DBMS to [keep track of and manage inventory in warehouses](https://www.techtarget.com/searchdatamanagement/feature/Evaluating-your-need-for-a-data-warehouse-platform). A DBMS can also be used to manage data for supply chain management applications that track the flow of goods and services, including the movement and storage of raw materials, work-in-process inventory, and finished goods from the point of origin to the point of consumption.
* DBMSes manage sales for any type of organization. This includes storing product, customer and salesperson information, as well as recording the sale, tracking fulfillment and maintaining sales history information.
* **Human resources.**A DBMS also makes it easier for organizations to keep track of and manage employee information in a human resources management application. This includes managing employee data such as addresses, phone numbers, salary details, payroll and paycheck generation.

Indeed, organizations that need to store data and access it later to conduct business have a viable use case for [deploying a DBMS](https://www.techtarget.com/searchdatamanagement/feature/Building-a-database-application-the-DIY-way).

### Changes in how DBMSes are built, sold and serviced

By 2019, [open source DBMS technologies](https://www.techtarget.com/searchdatamanagement/feature/Open-source-database-comparison-to-choose-the-right-tool) were rapidly gaining traction. In fact, Gartner projected that open source databases would account for 10% of total spending on database software for that year due to increased enterprise adoption. Most mainstream IT organizations use open source software in some of their mission-critical operations. This trend complements two others: the acquisition of open source database vendors by bigger rivals and the expansion of the cloud-based database service market.

In 2019, Gartner also said that cloud databases were driving most of the growth in the DBMS market, describing the cloud as "the default platform for managing data." In connection with the increasing shift toward the cloud, numerous DBMS vendors have introduced managed cloud database services that offer to free IT and data management teams from many of the tasks required to deploy, configure and administer database systems.

Another growing trend is what Gartner refers to as hybrid transactional analytical processing (HTAP). This means using a single DBMS to deliver both transaction processing and analytics without requiring a separate DBMS for each. To support this trend, more DBMS vendors are creating hybrid database systems that deliver multiple database engines within a single DBMS. Most hybrid DBMSes provide a combination of relational and multiple NoSQL engines and APIs. Examples include Altibase, Microsoft's Azure Cosmos DB and DataStax Enterprise.

**Next: Once you select a DBMS,**[**where should you host it**](https://www.techtarget.com/searchdatamanagement/feature/Should-you-host-your-database-on-site-or-in-the-cloud)**?**

### History of database management systems

The first DBMS was developed in the early 1960s when Charles Bachman created a navigational DBMS known as the Integrated Data Store (IDS). Shortly after, IBM developed Information Management System ([IMS](https://www.techtarget.com/searchdatacenter/definition/IMS-Information-Management-System)), a hierarchical DBMS designed for IBM mainframes that's still used by many large organizations today.

The next major advancement came in 1971 when the Conference/Committee on Data Systems Languages (CODSYL) standard was delivered. Integrated Database Management System (IDMS) is a commercial implementation of the network model database approach advanced by CODASYL.

But the DBMS market changed forever as the relational model for data gained popularity. Introduced by Edgar Codd of IBM in 1970 in his seminal paper "[A Relational Model of Data for Large Shared Data Banks,](http://www.morganslibrary.net/files/codd-1970.pdf)" the RDBMS soon became the industry standard. The first RDBMS was Ingres, developed at the University of California, Berkeley by a team led by Michael Stonebraker. At the same time, IBM was working on its System R project to develop a RDBMS.

Later in 1979, the first successful commercial RDBMS, Oracle, was released. This was followed a few years later by IBM's Db2, Sybase SQL Server and many others.

In the 1990s, as object-oriented (OO) programming became popular, several OO database systems came to market, but they never gained much market share. Later in the 1990s, the term NoSQL was coined. Over the next decade, several types of new [non-relational DBMS products](https://www.computerweekly.com/feature/Alternative-databases-set-for-mainstream-adoption) -- including key/value, graph, document and wide column store -- were grouped into the NoSQL category.

Today, the DBMS market is dominated by RDBMS, but [NoSQL database systems](https://www.techtarget.com/searchdatamanagement/infographic/NoSQL-database-comparison-to-help-you-choose-the-right-store) continue to grow in popularity.