database management system (DBMS)
What is a database management system?
A database management system (DBMS) is system software for creating and managing databases. A D
BMS makes it possible for end users to create, protect, read, update and delete data in a datab
ase. The most prevalent type of data management platform, the DBMS essentially serves as an int
erface between databases and users or application programs, ensuring that data is consistently
organized and remains easily accessible.
What does a DBMS do?
The DBMS manages the data; the database engine allows data to be accessed, locked and modified;
 and the database schema defines the database's logical structure. These three foundational ele
ments help provide concurrency, security, data integrity and uniform data administration proced
ures. The DBMS supports many typical database administration tasks, including change management
, performance monitoring and tuning, security, and backup and recovery. Most database managemen
t systems are also responsible for automated rollbacks and restarts as well as logging and audi
ting 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 multip
le locations in a controlled manner. A DBMS can limit what data end users see and how they view
 the data, providing many views of a single database schema. End users and software programs ar
e free from having to understand where the data is physically located or on what type of storag
e medium it resides because the DBMS handles all requests.
The DBMS can offer both logical and physical data independence to protect users and application
s from having to know where data is stored or from being concerned about changes to the physica
l structure of data. So long as programs use the application programming interface (API) for th
e database that the DBMS provides, developers won't have to modify programs just because change
s have been made to the database.
This article is part of
What is data management and why is it important?
Download this entire guide for FREE now!
In a relational database management system (RDBMS) -- the most widely used type of DBMS -- the
API is SQL, a standard programming language for defining, protecting and accessing data.
What are the components of a DBMS?
A DBMS is a sophisticated piece of system software consisting of multiple integrated components
 that deliver a consistent, managed environment for creating, accessing and modifying data in d
atabases. These components include the following:
Popular types and examples of DBMS technologies
Popular database models and management systems include RDBMS, NoSQL DBMS, NewSQL DBMS, in-me
y DBMS, columnar DBMS, multimodel DBMS and cloud DBMS.
RDBMS. Sometimes referred to as a SQL DBMS and adaptable to most use cases, RDBMS presents data
 as rows in tables with a fixed schema and relationships defined by values in key columns. RDBM
S Tier-1 products can be quite expensive, but there are high quality, open source options such
as PostgreSQL that can be cost-effective. Other examples of popular RDBMS products include Orac
le, MySQL, Microsoft SQL Server and IBM Db2.
NoSQL DBMS. Well-suited for loosely defined data structures that may evolve over time, NoSQL DB
MS may require more application involvement for schema management. There are four types of NoSQ
L database systems: document databases, graph databases, key-value stores and wide-column store
s. Each uses a different type of data model, resulting in significant differences between each
NoSQL type.
NewSQL DBMS. Modern relational systems that use SQL, NewSQL database systems offer the same sca
lable performance as NoSQL systems. But NewSQL systems also provide ACID (atomicity, consistenc
y, isolation and durability) support for data consistency. A NewSQL DBMS is engineered as a rel
ational, SQL database system with a distributed, fault-tolerant architecture. Other typical fea

tures of NewSQL system offerings include in-memory capability and clustered database services w
ith the ability to be deployed in the cloud. Many NewSQL DBMS packages have fewer features and
components and a smaller footprint than legacy relational offerings, making them easier to supp
ort and understand. Some vendors now eschew the NewSQL label and describe their technologies as
distributed SQL databases. CockroachDB, Google Cloud Spanner, NuoDB, Volt Active Data and Yuga
byteDB are examples of database systems in this category.
IMDBMS. An in-memory database management system predominantly relies on main memory for data st
orage, management and manipulation. By reducing the latency associated with reading from disk,
an IMDBMS can provide faster response times and better performance but can consume more resourc
es. Therefore, an in-memory database is ideal for applications that require high performance an
d rapid access to data, such as data stores that support real-time HTAP (hybrid transactional a
nd analytical process). Any type of DBMS (relational, NoSQL, etc.) can also support in-memory p
rocessing. SAP HANA and Redis are examples of in-memory database systems.
CDBMS. A columnar database management system stores data in tables focused on columns instead o
f rows, resulting in more efficient data access when only a subset of columns is required. It's
well-suited for data warehouses that have a large number of similar data items. Popular column
ar database products include Snowflake and Amazon Redshift.
Multimodel DBMS. This system supports more than one database model. Users can choose the model
most appropriate for their application requirements without having to switch to a different DBM
S. For example, IBM Db2 is a relational DBMS, but it also offers a columnar option. Many of the
most popular database systems similarly qualify as multimodel through add-ons, including Oracl
e, PostgreSQL and MongoDB. Other products, such as Azure Cosmos DB and MarkLogic, were develope
d specifically as multimodel databases.
Cloud DBMS. Built in and accessed through the cloud, the DBMS may be any type (relational, NoSQ
L, etc.) and a conventional system that's deployed and managed by a user organization or a mana
ged service provided by the database vendor. Popular cloud services that enable cloud database
implementation include Microsoft Azure, Google Cloud and AWS.
Benefits of using a DBMS
One of the biggest advantages of using a DBMS is that it lets users and application programmers
access and use the same data concurrently while managing data integrity. Data is better protec
ted 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 provide the following:
Another advantage of a DBMS is that database administrators (DBAs) can use it to impose a logic
al, structured organization on the data. A DBMS delivers economy of scale for processing large
amounts of data because it's optimized for such operations.
A DBMS can also provide many views of a single database schema. A view defines what data the us
er sees and how that user sees the data. The DBMS provides a level of abstraction between the c
onceptual schema that defines the logical structure of the database and the physical schema tha
t 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 c
ategories of data to the database without disrupting the existing system, thereby insulating ap
plications from how data is structured and stored.
However, a DBMS must perform additional work to provide these advantages, thereby incurring ove
rhead. A DBMS will use more memory and CPU than a simple file storage system, and different typ
es 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 require
d to run an enterprise DBMS, such as SQL Server, Oracle or IBM Db2. The hardware is usually a h
igh-end server with a significant amount of memory configured, coupled with large disk arrays t

o 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 i
nstallation and integrating the DBMS into the IT infrastructure. Dealing with additional comple
xity is also a concern when implementing a DBMS.
The DBMS software is complex and requires in-depth knowledge to properly implement and manage.
But the DBMS interfaces with many other IT components, such as the OS, transaction processing s
ystems, programming languages and networking software. Ensuring the proper configuration and ef
ficiency of such a complicated setup can be difficult and cause performance slowdowns or even s
ystem outages.
Some of the cost and administrative overhead of running enterprise database systems can be alle
viated by the cloud computing model. For example, the cloud service provider (CSP) installs and
 manages the hardware, which can be shared across cloud users. Furthermore, storage, memory and
 other resources can be scaled up and down as required based on usage needs. And basic DBA task
s like patching and simple backups become the responsibility of the CSP. Therefore, it can be e
asier and more cost-effective for some databases to be deployed in the cloud instead of on-prem
ises.

DBMS use cases

Enterprises that need to store data and access it later to conduct business have a viable use c
ase for deploying a DBMS. Any application requiring a large amount of data that needs to be acc
essed by multiple users or customers is a candidate for using a DBMS. Most medium to large orga
nizations can benefit from using a DBMS because they have more data-sharing and concurrency nee
ds and are able to more readily overcome cost and complexity issues. Sample customer use cases
for DBMS technology include the following:

Changes in how DBMSes are built, sold and serviced

By 2019, open source DBMS technologies were rapidly gaining traction. In fact, Gartner projecte
d that open source databases would account for 10% of total spending on database software for t
hat year due to increased enterprise adoption. By 2022, three of the top five databases ranked
by DB-Engines were open source. Most mainstream IT organizations use open source software in so
me of their mission-critical operations. This trend complements two others: acquisitions of ope
n source database vendors by bigger rivals and the expansion of the cloud-based database servic
e market.
In 2019, Gartner also said that cloud databases were driving most of the growth in the DBMS mar
ket, describing the cloud as "the default platform for managing data." In 2021, Gartner conclud
ed that "by 2022, cloud database management system revenue will account for 50% of the total DB
MS market revenue." In connection with the increasing shift toward the cloud, numerous DBMS ven
dors 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 HTAP -- using a single DBMS to deliver both
transaction processing and analytics without requiring a separate DBMS for each operation. To s
upport 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.

History of database management systems

The first DBMS was developed in the early 1960s when Charles Bachman created a navigational DBM
S known as the Integrated Data Store. Shortly after, IBM developed Information Management Syste
m (IMS), a hierarchical DBMS designed for IBM mainframes that's still used by many large organi
zations today.
The next major advancement came in 1971 when the Conference/Committee on Data Systems Languages

(CODASYL) standard was delivered. Integrated Database Management System is a commercial implem
entation of the network model database approach advanced by CODASYL.
But the DBMS market changed forever as the relational model for data gained popularity. Introdu
ced by Edgar Codd of IBM in 1970 in his seminal paper "A Relational Model of Data for Large Sha
red Data Banks," the RDBMS soon became the industry standard. The first RDBMS was Ingres, devel
oped at the University of California, Berkeley by a team led by Michael Stonebraker in the mid-
1970s. At about the same time, IBM was working on its System R project to develop an RDBMS.
In 1979, the first successful commercial RDBMS, Oracle, was released, followed a few years late
r by IBM's Db2, Sybase SQL Server and many others.
In the 1990s, as object-oriented (OO) programming became popular, several OO database systems c
ame to market, but they never gained significant market share. Later in the 1990s, the term NoS
QL was coined. Over the next decade, several types of new non-relational DBMS products, includi
ng key-value, graph, document and wide-column store, were grouped into the NoSQL category.
Today, the DBMS market is dominated by RDBMS, but NewSQL and NoSQL database systems continue to
grow in popularity.
The data platform vendor's new language model was designed to provide open source users with AI
development capabilities similar ...
The BI vendor's latest innovations include conversational analytics capabilities and prebuilt m
odels designed to help customers ...
The longtime analytics vendor's new bot enables customers to embed emerging technology in third
-party applications just as they ...
Many organizations struggle to manage their vast collection of AWS accounts, but Control Tower
can help. The service automates ...
There are several important variables within the Amazon EKS pricing model. Dig into the numbers
to ensure you deploy the service ...
AWS users face a choice when deploying Kubernetes: run it themselves on EC2 or let Amazon do th
e heavy lifting with EKS. See ...
The new applications and tools are for marketers looking to connect the different processes of
the content supply chain in a safe...
Whether an organization wants to improve its internal or external search strategies, an enterpr
ise search engine can help users ...
Advancements in data collection and processing may tempt information management professionals t
o use as much customer data as ...
With its Cerner acquisition, Oracle sets its sights on creating a national, anonymized patient
database -- a road filled with ...
Oracle plans to acquire Cerner in a deal valued at about $30B. The second-largest EHR vendor in
the U.S. could inject new life ...
The Supreme Court ruled 6-2 that Java APIs used in Android phones are not subject to American c
opyright law, ending a ...
SAP's first chief AI officer, Philipp Herzig, outlines the company's new AI-focused organizatio
n and underscores why companies ...
SAP and Nvidia are working together to combine platforms and services that help customers build
business-specific generative AI ...
SAP introduced new functionality in SAP Datasphere to help customers better manage their data e
nvironments with governance, ...