DATABASE MANAGEMENT SYSTEM (DBMS)

A database management system (DBMS) is system software for creating and managing databases. It also offers manipulation of the data like insertion, deletion, and updating of the data.

Database Management System was first introduced in the year 1960 by Charles W. Bachman. The first integrated database system for the company General Electric Company changed the way we store information, use data for insightful purposes.

This system is like the modern version of keeping files filled with vital information. However, the advancement in our technology, guided us to come up with a system where one can file so much information for different purposes like update, retrieval, storage, security, backup, etc.

DBMS system also performs the functions like defining, creating, revising and controlling the database. It is specially designed to create and maintain data and enable the individual business application to extract the desired data

What is a Database?

A database is an organized collection of structured information, or data, typically stored electronically in a computer system.

Why DBMS?

Database management systems enhance performance, integration, security, and compliance throughout an organization. The system offers many benefits over the traditional file system, including the following:

- It helps maintain data uniformity
- Handles large set of data efficiently
- Versatile
- Faster way of managing data

Some of the earlier version examples of DBMS are FoxPro, Clipper, RDBMS, etc.

RELATIONAL DATABASE MANAGEMENT SYSTEM (RDBMS)

A relational database management system (RDBMS) is a program that allows you to create, update, and administer a relational database. Relational Database Management System (RDBMS) is an advanced version of a DBMS system.

RDBMS is a software system which is used to store only data which need to be stored in the form of tables. In this kind of system, data is managed and stored in rows and columns which is known as tuples and attributes. RDBMS is a powerful data management system and is widely used across the world.

The Relational Database Management System was introduced in the 1970's itself. With the advent of technology, everyone wants everything fast-paced, innovative, productive and efficient. It is this ideology and belief that necessitated a shift from the traditional version of DBMS.

RDBMS product history

Many vying relational database management systems arose as news spread in the early 1970s of the relational data model. This and related methods were originally theorized by IBM researcher E.F. Codd, who proposed a database schema, or logical organization, that was not directly associated with physical organization, as was common at the time.

Codd's work was based around a concept of data normalization, which saved file space on storage disk drives at a time when such machinery could be prohibitively expensive for businesses.

File systems and database management systems preceded what could be called the RDBMS era. Such systems ran primarily on mainframe computers. While RDBMSes also ran on mainframes -- IBM's DB2 being a pointed example -- much of their ascendance in the enterprise was in UNIX midrange computer deployments. The RDBMS was a linchpin in the distributed architecture of client-server computing, which connected pools of stand-alone personal computers to file and database servers.

Numerous RDBMSes arose along with the use of client-server computing. Among the competitors were Oracle, Ingres, Informix, Sybase, Unify, Progress and others. Over time, three RDBMSes came to dominate in commercial implementations. Oracle, IBM's DB2 and Microsoft's SQL Server, which was based on a design originally licensed from Sybase, found considerable favor throughout the client-server computing era, despite repeated challenges by competing technologies.

As the 20th century drew to an end, lower-cost, open source versions of RDBMSes began to find use, particularly in web applications.

Eventually, as distributed computing took greater hold, and as cloud architecture became more prominently employed, RDBMSes met competition in the form of NoSQL systems. Such systems were often specifically designed for massive distribution and high scalability in the cloud, sometimes forgoing SQL-style full consistency for so-called *eventual consistency* of data. But, even in the most diverse and complex cloud systems, the need for some guaranteed data consistency requires RDBMSes to appear in some way, shape or form. Moreover, versions of RDBMSes have been significantly restructured for cloud parallelization and replication.

What is a Relational Database?

A relational database is a type of database. It uses a structure that allows us to identify and access data *in relation* to another piece of data in the database. Often, data in a relational database is organized into tables. For example, a column called age may have a type of INTEGER (denoting the type of data it is meant to hold).

A relational database example

Here's a simple example of two tables a small business might use to process orders for its products. The first table is a customer info table, so each record includes a customer's name, address, shipping and billing information, phone number, and other contact information. Each bit of information (each attribute) is in its own column, and the database assigns a unique ID (a key) to each row. In the second table—a customer order table—each record includes the ID of the customer that placed the order, the product ordered, the quantity, the selected size and color, and so on—but not the customer's name or contact information.

These two tables have only one thing in common: the ID column (the key). But because of that common column, the relational database can create a relationship between the two tables. Then, when the company's order processing application submits an order to the database, the database can go to the customer order table, pull the correct information about the product order, and use the customer ID from that table to look up the customer's billing and shipping information in the customer info table. The warehouse can then pull the correct product, the customer can receive timely delivery of the order, and the company can get paid.

Relational model and data consistency

The relational model is the best at maintaining data consistency across applications and database copies (called instances). For example, when a customer deposits money at an ATM and then looks at the account balance on a mobile phone, the customer expects to see that deposit reflected immediately in an updated account balance. Relational databases excel at this kind of data consistency, ensuring that multiple instances of a database have the same data all the time.

It's difficult for other types of databases to maintain this level of timely consistency with large amounts of data. Some recent databases, such as NoSQL, can supply only "eventual consistency." Under this principle, when the database is scaled or when multiple users access the same data at the same time, the data needs some time to "catch up." Eventual consistency is acceptable for some uses, such as to maintain listings in a product catalog, but for critical business operations such as shopping cart transactions, the relational database is still the gold standard.

Commitment and atomicity

Relational databases handle business rules and policies at a very granular level, with strict policies about commitment (that is, making a change to the database permanent). For example, consider an inventory database that tracks three parts that are always used together. When one part is pulled from inventory, the other two must also be pulled. If one of the three parts isn't available, none of the parts should be pulled—all three parts must be available before the database makes any commitment. A relational database won't commit for one part until it knows it can commit for all three. This multifaceted commitment capability is called atomicity. Atomicity is the key to keeping data accurate in the database and ensuring that it is compliant with the rules, regulations, and policies of the business.

FEATURES OF RELATIONAL DATABASE MANAGEMENT SYSTEMS

Elements of the relational database management system that overarch the basic relational database are so intrinsic to operations that it is hard to dissociate the two in practice.

The most basic RDBMS functions are related to create, read, update and delete operations -- collectively known as CRUD. They form the foundation of a well-organized system that promotes consistent treatment of data.

The RDBMS typically provides data dictionaries and metadata collections that are useful in data handling. These programmatically support well-defined data structures and relationships. Data storage management is a common capability of the RDBMS, and this has come to be defined by data objects that range from binary large object -- or blob -- strings to stored procedures. Data objects like this extend the scope of basic relational database operations and can be handled in a variety of ways in different RDBMSes.

The most common means of data access for the RDBMS is SQL. Its main language components comprise data manipulation language and data definition language statements. Extensions are available for development efforts that pair SQL use with common programming languages, such as the Common Business-Oriented Language (COBOL), Java and .NET.

RDBMSes use complex algorithms that support multiple concurrent user access to the database while maintaining data integrity. Security management, which enforces policy-based access, is yet another overlay service that the RDBMS provides for the basic database as it is used in enterprise settings.

RDBMSes support the work of database administrators (DBAs) who must manage and monitor database activity. Utilities help automate data loading and database backup. RDBMSes manage log files that track system performance based on selected operational parameters. This enables measurement of database usage, capacity and performance, particularly query performance. RDBMSes provide graphical interfaces that help DBAs visualize database activity.

While not limited solely to the RDBMS, ACID compliance is an attribute of relational technology that has proved important in enterprise computing. These capabilities have particularly suited RDBMSes for handling business transactions.

As RDBMSes have matured, they have achieved increasingly higher levels of query optimization, and they have become key parts of reporting, analytics and data warehousing applications for businesses as well. RDBMSes are intrinsic to operations of a variety of enterprise applications and are at the center of most master data management systems.

How relational databases are structured

The relational model means that the logical data structures—the data tables, views, and indexes—are separate from the physical storage structures. This separation means that database administrators can manage physical data storage without affecting access to that data as a logical structure. For example, renaming a database file does not rename the tables stored within it.

The distinction between logical and physical also applies to database operations, which are clearly defined actions that enable applications to manipulate the data and structures of the database. Logical operations allow an application to specify the content it needs, and physical operations determine how that data should be accessed and then carry out the task.

To ensure that data is always accurate and accessible, relational databases follow certain integrity rules. For example, an integrity rule can specify that duplicate rows are not allowed in a table in order to eliminate the potential for erroneous information entering the database.

What to look for when selecting a relational database

The software used to store, manage, query, and retrieve data stored in a relational database is called a relational database management system (RDBMS). The RDBMS provides an interface between users and applications and the database, as well as administrative functions for managing data storage, access, and performance.

Several factors can guide your decision when choosing among database types and relational database products. The RDBMS you choose will depend on your business needs. Ask yourself the following questions:

- What are our data accuracy requirements? Will data storage and accuracy rely on business logic? Does our data have stringent requirements for accuracy (for example, financial data and government reports)?
- Do we need scalability? What is the scale of the data to be managed, and what is its anticipated growth? Will the database model need to support mirrored database copies (as separate instances) for scalability? If so, can it maintain data consistency across those instances?
- How important is concurrency? Will multiple users and applications need simultaneous data access? Does the database software support concurrency while protecting the data?
- What are our performance and reliability needs? Do we need a high-performance, high-reliability product? What are the requirements for query-response performance? What are the vendor's commitments for service level agreements (SLAs) or unplanned downtime?

Uses of RDBMS

Relational database management systems are frequently used in disciplines such as manufacturing, human resources and banking. The system is also useful for airlines that need to store ticket service and passenger documentation information as well as universities maintaining student databases.

Some examples of specific systems that use RDBMS include IBM, Oracle, MySQL, Microsoft SQLServer and PostgreSQL.

Why RDBMS?

An RDBMS offers businesses a systematic view of data, which can be used to enhance different aspects of decision-making. Relational databases offer a number of other advantages as well, including:

- Allow multiple user access
- Store large pack of data
- Minimum Data Redundancy
- Maintains Data Integration
- Better tools for structuring and Organizing Data

KEY DIFFERENCE

- DBMS stores data as a file whereas in RDBMS, data is stored in the form of tables.
- DBMS supports single users, while RDBMS supports multiple users.
- DBMS does not support client-server architecture but RDBMS supports client-server architecture.
- DBMS has low software and hardware requirements whereas RDBMS has higher hardware and software requirements.
- In DBMS, data redundancy is common while in RDBMS, keys and indexes do not allow data redundancy.

DIFFERENCES BETWEEN DBMS AND RDBMS

The terms “DBMS” and “RDBMS” stand for Relational Database Management and database management systems, respectively. The main difference between DBMS and RDBMS is that RDBMS stores data as tables and DBMS stores data as a file. Although DBMS and RDBMS both are used to store information in physical databases, there are some remarkable differences between them.

NO.	DBMS	RDBMS
1	DBMS application stores data as a file	RDBMS applications store data in a tabular form.
2	Data is generally stored in either a hierarchical form or a navigational form	The tables have an identifier called primary key and the data values are stored in the form of tables.
3	Normalization is not present	Normalization is present
4	Does not apply security with regards to data manipulation	Multiple levels of security. Log files are created at OS, Command, and object level.
5	DBMS uses a file system to store data, so there will be **no relation between the tables**.	Data values are stored in the form of tables, so a **relationship** between these data values will be stored in the form of a table as well
6	DBMS has to provide some uniform methods to access the stored information.	The RDBMS system supports a tabular structure of the data and a relationship between them to access the stored information.
7	DBMS **does not support distributed databases**.	RDBMS **supports distributed databases**.
8	DBMS is meant for small organizations and **deals with small data**. It supports **single users.**	RDBMS is designed to **handle large amounts of data**. it supports **multiple users**.
9	Data is stored in the form of tables which are related to each other with the help of foreign keys.	Data fetching is rapid because of its relational approach.

10	DBMS does not support client-server architecture	RDBMS supports client-server architecture.
11	Low software and hardware needs.	Higher hardware and software are needed.
12	In a regular database, the data may not be stored following the ACID model. This can develop inconsistencies in the database.	Relational databases are harder to construct, but they are consistent and well structured. They obey ACID (Atomicity, Consistency, Isolation, Durability).
13	Data elements need to be accessed individually.	Data can be easily accessed using SQL query. Multiple data elements can be accessed at the same time
14	Examples of DBMS are a file system, XML, Windows Registry, etc.	Examples of RDBMS are MySQL, Oracle, SQL Server, etc.

Conclusion

Database Management Systems and Relational Database Management Systems are both system software for creating, managing and storing databases. A relational database management system (RDBMS), however, is a more advanced version of the traditional database management system (DBMS) that is able to facilitate better tools and features than the traditional DBMS. DBMS stores data hierarchically and in a file format without any relationship between the data, whereas RDBMS stores data in a tabular format.

RDBMS and DBMS differ greatly in terms of their structures. As DBMS was the first version of the software for maintaining databases, it does not include features for data security, data integrity, data access, multiple-user access, or faster data retrieval. The RDBMS is faster, more efficient, and ensures there are no data redundancies or inconsistencies. In a nutshell, there is no line set in stone that proves RDBMS is the best form of database management system out there. While there are different types of database management systems, RDBMS is by far the best, and faster than DBMS.