



# The database evolution: modernization for a **data-driven** world

## Why bringing data to the cloud is inevitable

As AI emerges, data is becoming more important than ever. In order to stay relevant, organizations have to start using their data to fuel their business models and improve their operations. However, large volumes of data do present challenges. In this article, we briefly revisit the history of databases to learn how these challenges have been tackled one by one in the course of the history of databases. Finally, we will discuss why organizations need to adopt the latest in database technology and the possible scenarios to implement this technology.



**Giovanni Vierling**  
is a director in the Microsoft  
team of KPMG Enterprise  
Solutions.

## INTRODUCTION

During the COVID-19 pandemic, businesses needed to respond to immediate needs created by the global crisis. On-premises systems lacked the scalability to support working from home for many people. Cloud technology helped businesses to respond to these challenges with cloud-based VPNs, cloud-based firewalls and modern workplace systems. Many organizations needed to work through the recovery from the pandemic with new approaches leveraging cloud-based technology, and in some cases, even looking to reimagine their businesses for the long term.

Data is a valuable commodity for companies because of its potential to empower better decision-making. However, data management tools and processes must be equipped to handle the volume, velocity, and variety of data that companies face today. According to [Abdu22], the exploding volume of data and the many new applications that generate and consume it have created an urgent need for data management programs to help organizations stay on top of their data.

Now let's first take a step back into history to see where we came from and get an understanding of what these data management programs need to accomplish. We will then briefly explain the pain points of classic database systems before we discuss how modern database management systems overcome some of the challenges that many organizations face today. Finally, we will discuss the different approaches that database management programs can take to implement a modern database management system.

## A BRIEF HISTORY OF DATABASE MANAGEMENT SYSTEMS

The history of databases dates back long before computers were invented. In the past, data was stored in journals, libraries, and filing cabinets, taking up space and making it difficult to find and back up data as well as making analyses and correlations. The advent of computers in the early 1960s marked the beginning of computerized databases. In 1960, Charles W. Bachman designed the integrated database system, the “first” DBMS. IBM, not wanting to be left out, created a database system of its own, known as IMS. [Foot21] describes both database systems as the forerunners of navigational databases. According to [Kell22] however, the history of databases as we know them, really begins in 1970.

In 1970, a computer scientist from IBM named Edgar F. Codd published an academic paper titled *A Relational Model for Data for Large Shared Banks*. Codd described a new way of modeling data by introducing relational tables to store data only once. This system (Relational Database Management System (RDBMS)) allowed the database to answer any question as long as the data was in it, and allowed efficient use of storage. Back then, storage was still a major challenge with hard drives having the size of a truck wheel and being quite expensive.

In the eighties and nineties, relational databases grew increasingly dominant. IBM developed SQL the Structured Query Language, which became the language of data and became ANSI and OSI standards in 1986 and 1987. When processing speeds got higher and “unstructured” data (art, photographs, music etc.) became much more commonplace, a new requirement came to light. Unstructured data is both non-relational and schema-less. RDBMS were simply not designed to handle this kind of data. With growing amounts of data and new purposes being explored, new database management systems were developed, such as systems using column stores and key-value stores. Each of these systems had strengths and limitations that would make them suited for a limited use case. NoSQL (“Not Only” Structured Query Language) came out in response to the need for faster speed and processing of unstructured data ([Foot21]).

In the 2010s and 2020s, Database Management Systems (DBMS) have been evolving and innovating to meet the emerging needs and challenges of management information systems and other applications. It started with the ability to run your DBMS in the cloud on a cloud-based infrastructure, but soon after software companies such as Microsoft began to offer native cloud DBMS. In a native cloud DBMS, the vendor is responsible for the full service of the DBMS, including hosting, hardware, licenses, upgrades and maintenance. Cloud DBMS, which are

hosted and managed on cloud platforms, offer benefits in cost, availability, scalability, and security, as well as enabling users to access data from anywhere and anytime. Based on research by [Ulag23], AI-enhanced DBMS use artificial intelligence techniques to enhance the functionality and performance of DBMS, with benefits in automation, management, optimization, and intelligence.

Larger volumes of data, the need to work with varying teams across different locations with this data and the need to keep costs down, continues to drive new database technologies.

## PAIN POINTS OF CLASSIC DATABASE SYSTEMS

Classic on-premises database systems are limited in terms of scalability and maintainability. The hardware behind these systems limits the size of their storage and their computational power and can often not be expanded without significant investment. Since these systems often have a different array of capabilities which requires products from multiple vendors, maintenance is expensive and difficult.

Classic, on-premises, databases eventually create the following pain points:

1. Cost and infrastructure management:
  - Hardware costs: One of the most significant pain points is the substantial upfront cost associated with purchasing and maintaining the necessary hardware infrastructure. This includes servers, storage devices, networking equipment, and cooling systems. Organizations need to budget for hardware upgrades and replacements over time.
  - Operational costs: Apart from the initial capital expenditure, there are ongoing operational costs, such as electricity, space, and skilled IT personnel to manage and maintain the on-premises infrastructure. Scaling the infrastructure to accommodate growth can be both costly and time-consuming.
2. Scalability and flexibility:
  - Limited scalability: On-premises DBMS solutions often have finite capacity limits based on the hardware infrastructure in place. Scaling up to accommodate increasing data volumes or user demands can be slow and expensive, requiring the procurement of new hardware.
  - Lack of flexibility: Making changes to the infrastructure or DBMS configurations can be cumbersome. It may involve downtime, complex migration processes, and compatibility issues, limiting an organization’s ability to quickly adapt to changing business needs.



3. Maintenance and security:
- Maintenance overhead: On-premises DBMS solutions require constant maintenance, including software updates, patch management, and hardware maintenance. This can be time-consuming and may disrupt normal operations.
  - Security concerns: Organizations are responsible for implementing robust security measures to protect their on-premises databases. This includes physical security, data encryption, access control, and disaster recovery planning. Failing to address these security concerns adequately can lead to data breaches and regulatory compliance issues.
  - While on-premises DBMS solutions provide organizations with greater control over their data and infrastructure, they also come with significant challenges. Many businesses are turning to cloud-based database solutions to alleviate these pain points, as they offer greater scalability, reduced infrastructure management overhead, and enhanced flexibility, among other benefits. However, the choice between on-premises and cloud-based databases should be based on an organization's specific needs, budget, and regulatory requirements.

Most companies, today, use combinations of public clouds and private clouds. The database management systems need to be able to provide access to data sources across these different cloud offerings. In classic database management systems, different workloads are needed to work on their own data, each workload having its own governance structure. This leads to redundant data in different locations and the need to duplicate this data to a data warehouse for analytics resulting in redundant data. Security across these different workloads and redundant, scattered data is a common pain point for organizations.

## THE CASE FOR MODERNIZATION

According to [Ulag23]: In the age of artificial intelligence (AI), the ability to analyze, monitor and act on real-time data is becoming crucial for companies that wish to remain competitive in their industries. Therefore, modern database systems need to be able to handle large volumes of data coming from multiple data sources like sensors, IOT devices and many other data sources. This data needs to be ingested into the data analytics environment with high throughput and low latency.

When considering application modernization and therefore also database modernization, the question might be how to get this complex database technology transformation right. Almost any application collects, stores, retrieves and manages data in some sort of database. As

---

In the age of AI, the ability to analyze, monitor and act on real-time data is becoming crucial for companies that wish to remain competitive

we have seen, traditionally this was often done using a relational database management system (RDBMS) on a dedicated, often on-premises, server. As discussed, these types of databases provide organizations with increasing problems regarding:

- a. high license and hardware costs;
- b. license compliance constraints (e.g. expensive Java clients);
- c. new sources and types of data, often unstructured;
- d. scalability performance and global expansion needs;
- e. modern applications often needing cloud-native agility and speed of innovation which cannot easily be achieved with dedicated hardware in an on-premises situation.

According to [Gris23], a modernization path therefore needs to include the following business and technology gains:

- Use open-source compatible databases with global scale capabilities.
- Remove the undifferentiated heavy lifting of self-managing database servers and move to managed database offerings.
- Unlock the value of data, and make it accessible across application areas and organizations such as analytics, data lakes, Business Intelligence (BI), Machine Learning (ML) and Artificial Intelligence (AI).
- Enable decoupled architectures (event driven, micro services).
- Use highly scalable purpose-built databases that are appropriate for non-relational and streaming data.

## DATABASE MODERNIZATION PATTERNS

In our consulting practice we recognize four common patterns for database modernization:

1. Lift and shift: Move a database server to a public cloud service like Azure, Google Cloud or Amazon Web Services.

In this scenario, the organization simply deploys a Virtual Machine (VM) in their private or public cloud and runs the exact same database management software from this (IaaS) machine. Each application is migrated as-is. It is a quick solution without the risks or costs associated with making changes to code or software architecture. The downside of course is that the organization remain responsible for the maintenance of the VM, still must pay the associated licenses and although improved, still has scalability limitations with respect to storage and compute.

2. Refactor or repackage: This migration strategy involves limited changes to the application design and moves the database to a managed instance such as Azure SQL Managed Instance or Amazon RDS for SQL Server.

In this scenario, licenses are replaced by a consumption-based model and limitations with respect to scalability decrease. The database system is managed by the service provider and the organization therefore no longer has to manage the hardware, operating systems or perform maintenance on the RDBMS.

3. Rearchitect: By modifying or extending the application's code base, the application is scaled and optimized for the cloud. In this scenario, the application will be using a public cloud-native database management system such as Aurora (AWS), Google's Cloud SQL or Azure SQL Database. This scenario has all the advantages of leveraging a modern database solution that fulfils the need of a better practice modernization path as described above. However, not all applications can be modernized to the extent needed, sometimes it is simply better to build cloud-native applications.
4. Rebuild: The application and its database will be rebuilt from scratch in this scenario. Leveraging cloud-native development platforms and leveraging cloud-native database solutions such as Microsoft Dataverse, Microsoft Azure Serverless SQL Pools, Google Cloud SQL and Amazon Aurora Serverless.

Compare the complexity and value proposition for these options in Table 1 containing different modernization patterns. A low complexity means that few dependencies and not much changes need to be made to the software that use the DBMS, and high complexity means that many dependencies and substantial changes need to be made to the software that uses the DBMS.

**Table 1.** Different modernization patterns.

Lift and shift	Low	Low	Low
Refactor or repackage	Low	Medium	Medium
Rearchitect	Medium	High	High
Rebuild	Highest	Highest	Highest

A lift and shift approach is a quick solution with minimal risks, it helps in cases where hardware needs to be replaced imminently and reduces the risk associated with changing software. It does not change the licensing nor the limitations with respect to storage and compute that are associated with the limitations of a Virtual Machine and the RDBMS system. It can be a very good start when an organization needs to renew their data center contracts or when hardware needs to be replaced.

If the goal is to improve the management of data around your application landscape and not yet modernize the applications, rearchitecting the application landscape by changing to a managed instance for your database system can be a viable solution. It helps the organization

to become less dependent on hardware and maintenance of the database management system. It will however remain dependent on dedicated hardware, now managed by a service provider, and will therefore still have limitations with respect to scalability (storage and compute). The risk of this scenario is also relatively low since it does not require changes to the DB schema, data types, and table structure.

If the goal is to fully modernize the application landscape and independent of hardware, maintenance and licenses a complete rebuild is in order. This modernization pattern has the highest risk and will cost the most, because both software and DBMS need to change, but it does allow the organization to use the database as a service and become fully independent of hardware, maintenance and licenses.

## CONCLUSION

With the increase of data volumes, the increasing need for speedy access to data, the processing of unstructured data and the decentralization of data, database management systems have evolved from simple relational databases to modern cloud-based database management systems that can:

- manage the needs of the increasing number of cloud-native applications;
- manage governance across databases;
- perform data analytics on a large volume of data in near real time, with no extraction, transformation or loading pipelines and no performance impact on transactional workloads; and with anyone in the organization able to access and analyze the data.

Breaking free from legacy databases allows the organization to migrate away from complex licensing structures to a pay-as-you-go scenario. It also breaks free from scalability limitations since most of the cloud-native database management systems separate storage from compute and have these run in a separate layer completely detached from the hardware layer. Owners of serverless applications are not concerned with capacity planning, configuration, management, maintenance, fault tolerance, or scaling of containers, VMs, or physical servers, the cloud provider allocates machine resources on demand, taking care of the servers on behalf of their customers.

In our consulting practice, we see four different migration patterns that organizations can choose from to migrate to a modernized data platform with varying degree of risks and costs and therefore also rewards. Depending on the organization's goals and ambitions, a fitting migration pattern can be chosen.

## References

- [Abdu22] Abdullahi, A. (2022, October 28). What is data modernization? Retrieved from: <https://www.techrepublic.com/article/what-is-data-modernization/>
- [Foot21] Foote, K.D. (2021, October 25). A Brief History of Database Management. Retrieved from: <https://www.studocu.com/row/document/federal-college-of-education-special-oyo/computer-science/a-brief-history-of-database-management/40351890>
- [Gris23] Grischenko, A. & Rajain, S. (2023, July 12). How to plan for a successful database modernization. Retrieved from: <https://aws.amazon.com/blogs/database/how-to-plan-for-a-successful-database-modernization/>
- [Kell22] Kelly, D. (2022, February 24). A brief history of databases: from relational to no SQL to distributed SQL. Retrieved from: <https://www.cockroachlabs.com/blog/history-of-databases-distributed-sql/>

[Li23] Li, A. (ASA). (2023, July 27). Microsoft Fabric Event Streams: Generating Real-time Insights with Python, KQL and PowerBI. Retrieved from: [https://blog.fabric.microsoft.com/en-us/blog/author/Alicia%20Li%20\(ASA\)](https://blog.fabric.microsoft.com/en-us/blog/author/Alicia%20Li%20(ASA))

[Ulag23] Introducing Microsoft Fabric: Data analytics for the era of AI. Retrieved from: <https://azure.microsoft.com/en-us/blog/introducing-microsoft-fabric-data-analytics-for-the-era-of-ai/>

## About the author

**Giovanni Vierling** is a director in the Microsoft team of KPMG Enterprise Solutions. He is responsible for our services concerning Microsoft Cloud such as Powered Data Analytics, the Power platform and Azure. Giovanni has worked as an IT manager for one of our customers and was globally responsible for all MSFT Applications within that organization. He has ample operational experience with Azure, Power Platform, Power BI, RPA and Dynamics 365.