White Paper

Choosing the Right Tool for the Job: Purpose-Built Databases from AWS

Sponsored by: Amazon Web Service

Carl W. Olofson
December 2020

IN THIS WHITE PAPER

The most commonly used database management systems (DBMSs) are designed to support relational databases. Often, additional functionality has been folded into them to provide some support for other operational models. Their support for these operational models usually proves inadequate for work that is centered on non-relational models. For such operational models, including document database, graph analysis, or other specialized models of operation, a better choice is to use a DBMS that is specifically designed to address each such model of operation. This white paper examines the problem of picking the right DBMS for the operational model at hand. It considers how generalized multimodel DBMSs often are jacks of all trades but masters of none, while a specific, purpose-built DBMS is the better choice for a specific data management problem. It then looks at the range of purpose-built DBMSs offered by AWS, examining how each is aimed at a specific data management capability to deliver maximum value for that model.

SITUATION OVERVIEW

Database management systems have been in use since the early 1960s. They overcame the limitations of fixed batch jobs and flat files by providing data governed according to an application-neutral structure, called a "schema," so that the problem of file management was greatly reduced and application programs could be run in any order. The problem with these DBMSs was that they were difficult to query unless one had exquisitely precise knowledge of their structure. Dr. Edgar F. Codd, an IBM Fellow, overcame this problem with relational data management, which calls for the organization of data according to mathematical set theory. Greatly simplified, this model calls for unique keys and some number of associated attributes forming a relation. Every list of a key value and its associated attributes is called a tuple. For common simplicity, we refer to a relation as a table, a tuple as a row, and an attribute (as well as the key) as a column. A system of organization that ensures consistency without either ambiguity or redundancy is normally used, which is called "normalization." Table rows may be related to rows in other tables through a foreign key value that matches the primary key value of the other table. A standard query language emerged to support it, called Structured Query Language (SQL). As new DBMSs were developed based wholly on relational data management, SQL was expanded to include data manipulation as well as query.

Because relational databases were so easy to query, they were used initially for data analytics, but over time, businesses wanted to base their data in a single system, so relational database management systems (RDBMSs) evolved to include online transactional processing (OLTP) as well as analytic capabilities. By the early 1990s, RDBMS was considered the standard for all manner of data management, and competing forms were largely relegated to the edges of the data management landscape.

But then came a series of profound changes. These began with the development of much more powerful processors capable of 64-bit addressability, which meant they could handle far greater amounts of data in memory than had been previously possible. High-speed networks enabled the transmission of much more data much faster than ever before, storage itself became cheaper and faster, and the cost of processors and memory plunged in price relative to their capacity.

In addition to these physical improvements, different kinds of applications began to emerge that demanded greater flexibility, scalability, and capacity than relational DBMSs could handle. There was also a need to handle data such as streaming sensor and IoT data, smart mobile device data, log data, and others that had formats that did not fit gracefully into a relational table structure. In many cases, to form a complete set of business data for an operation, it may be necessary to execute a query that joins a number of tables in order to extract the related values, and this creates both complexity in the query and some rigidity in terms of application development. So people began experimenting with different modes of data management, sparking the emergence of the NoSQL movement. These began with document databases, some processing a document standard called XML (Extensible Markup Language), then others supporting another document standard called JSON (JavaScript Object Notation), then Hadoop for data lakes (very large unstructured data collections), and then came key-value stores and wide column stores. All these offered better scalability and flexibility for a variety of jobs that RDBMSs struggled to support.

Now, changes in system architectures and levels of virtualization support have enabled public cloud services to emerge, and with them the idea of pay-as-you-go services. Applications developed for these public clouds are designed to take full advantage of the ability to limit resources used by adopting a microservices architecture and running the code in containers, where application actions can execute and go away, ensuring that one is only billed for the processors used as they are used and are not resident in fixed servers. A common aim of microservices is to achieve a "serverless architecture" (one in which applications run in various servers on a dynamic basis rather than sitting resident in specific servers). Microservices make special demands on the database, and it's important to have a DBMS that is designed not only for the type of data being handled but also to satisfy the operational requirements of a serverless application.

Today, in addition to the older, mostly mainframe-based navigational and multivalue DBMSs (many of which are being viewed as requiring conversion to newer forms to move to the public cloud), we have RDBMSs, document DBMSs, wide column stores, key-value stores, graph DBMSs, and more. Each of these operational models can be very well attuned to specific workloads and do a better job than any one architecture could do in attempting to support all data management needs.

Multimodel and Broadly Featured DBMSs

Today, there are many DBMSs that have developed multimodel capabilities; that is, they can support more than one data management model. For instance, an RDBMS may also support document data, key-value data, and graph data, in addition to relational data. For the most part, these multimodel DBMSs are generally very good at the main job for which they were designed, but their capabilities in the other models of operation lack the performance, scalability, and ease of development and use offered by database systems that are purpose built to support those specific models. This does not mean that such DBMSs have no value in these areas. Often, an organization may want to keep its business data in relational tables but maintain connections with JSON document kept in the same database for the sake of data integrity, for instance. This is a perfectly reasonable use of such systems, where the primary operational model is relational.

Workload-Targeted DBMSs

In an environment in which data can be shared among databases in a carefully controlled and well-governed way, it is often preferable to use a DBMS that is targeted at a particular workload, such as relational data management, or document management, or graph analysis, and excels at supporting that workload. Relational databases can organize and optimize tables in row or columnar format for the most efficient transaction processing and fastest query processing. Document databases can optimize document structures and storage to provide the most efficient retrieval and update combined with high-speed search. Key-value stores can optimize for very high-speed randomized storage and retrieval. Graph databases can provide an environment optimized for either broad and shallow or narrow and deep graph traversal and for the management of either property or Resource Description Framework (RDF) graphs.

Choosing the Right DBMS for the Job

For applications that are more ideally targeted at a particular type of database workload, however, it generally makes more sense to choose a DBMS that is purpose built for that workload in order to get the best result in terms of high availability, low latency, extreme performance, and behavior that is well attuned to the particular type of data in question.

Table 1 illustrates the various database models, how they are used, and examples of common workloads they serve.

TABLE 1

Database Models and Typical Workloads

Model	How the Model Is Used	Typical Workload
Relational	Schematic, table-oriented data management in support of ACID (atomicity, consistency, isolation, durability) transactions and random, orthogonal queries	Back-office accounting and record-keeping capture of calculation data, ACID transactions, random orthogonal query either ad hoc or for reports; for large and complex analytic databases, RDBMS supports data warehouses
Key-value	Non-schematic data grouped in blocks functionally for rapid, random insert and retrieval; data format is under program control	High-touch customer experience (CX) applications, ecommerce applications, and user session management for mobile apps and gaming; also, for any application requiring high scalability and the flexibility of NoSQL and does not require complex query support
Document	Holds data in a standard document format such as JSON or XML, supporting more complex data structures	Operational applications with dynamic data requirements, including data definitions, and supporting a wide variety of applications that require management of blocks of formatted data
Graph	Collecting and analyzing data capturing relationships among many entities, such as persons and locations	Can be used for text semantics capture and analysis for search or cataloguing purposes or for navigation or pattern analysis, such as in mobile tracking systems or for fraud detection
Wide column store	Holds data in defined column groups, enabling multiple applications and users to share the data	High throughput operational applications that require speed and scalability along with some random query capability and not as much definitional flexibility as is delivered by a document database
In-memory data store	Live data sharing among applications or running processes in the same application on different servers; expands available data	IoT data capture and analysis, log data analysis, capture of streaming data, recording of changed data, and managing of events
Specialized data (e.g., time series and ledger database)	Capturing and analytic functions for time series, spatial, or ledger data	Analysis platform for log data or location data

Source: IDC, 2020

In some cases, where one of these models dominate but the others are needed in the same database, a multimodel DBMS is called for. In most other cases, a purpose-built DBMS aimed at a specific model is preferred. A collection of applications may require some combination of such purpose-built databases, with each assigned to the type of application to which it is best suited.

AWS Purpose-Built DBMSs

Amazon Web Services (AWS) is seeking to serve users by providing a number of managed purpose-built database options that cover most cases. In addition to support for multimodel DBMSs, AWS offers a purpose-built DBMS in each of the key data management model areas, all linked together and interactive through a common method of data interchange and governance. All are optimized to take maximum advantage of the underlying AWS environment and set of resources and are fully managed by AWS. These purpose-built DBMSs are described in the sections that follow.

Relational Data Management

OLTP (Amazon Aurora and Amazon Relational Database Service)

AWS provides full OLTP support with the Amazon Aurora RDBMS, which features scalable ACID (atomicity, consistency, isolation, durability) support for database transactions. It features SQL interfaces that are fully compatible with either MySQL or PostgreSQL to facilitate migration from those platforms and ease of use for users of those platforms. Aurora is also available in a serverless configuration that serves the needs of low-volume, unpredictable or cyclical workloads.

Amazon Relational Database Service (Amazon RDS) enables users to platform their favorite RDBMS, whether open source or commercial, on AWS, under AWS management. RDBMSs supported include MySQL, PostgreSQL, MariaDB, Microsoft SQL Server, and Oracle Database.

Analytic (Amazon Redshift)

Amazon Redshift is a petabyte-scale, fully managed data warehouse service designed for online analytic processing (OLAP) and business intelligence (BI) applications. It uses a massively parallel processing (MPP) clustered columnar RDBMS to carry out complex analytic queries at scale. You can use Amazon Redshift to take a "lake house" approach to data warehousing with features that enable you to query, combine, and save data from your data warehouse, data lake, and operational databases.

Key-Value (Amazon DynamoDB)

A key-value store enables the storing of blocks of data associated with unique key-value pairs. The format of the data block is under program control and may be in the form of a standard data document, such as JSON. It is very popular for mobile, web, gaming, ad tech, and IoT data management, as well as persistent data caching. Amazon DynamoDB is designed to offer higher performance and scalability with multiregion and multi-active capability. It is fully managed by AWS.

Document (Amazon DocumentDB with MongoDB Compatibility)

A document database contains explicit support for a particular document type, usually JSON, with tagged search and reporting capability. Amazon DocumentDB (with MongoDB compatibility) is a fully managed document database service that is designed to support MongoDB workloads. It enables aggregations, ad hoc queries, and flexible indexes and has a query processor that enables index intersection. Its scalability is delivered by decoupling storage from compute and allowing each to scale independently. It also replicates six copies of each database across three AWS Availability Zones (AZs).

Graph Data Management (Amazon Neptune)

Amazon Neptune enables support for complex property graphs that may contain billions of nodes and edges. It supports the W3C RDF and SPARQL standards as well as Apache TinkerPop and Gremlin. Graphs can be used to perform content semantic analysis, classify information on real-world people or objects, perform complex relationship pattern recognition, and detect multilevel associations and patterns of associations. Common use cases for Amazon Neptune include recommendation engines, social network analysis, IT operational network management and analysis, and life sciences research. Graphs are used to build customer 360 solutions to understand the customer journey; provide social recommendations on people to follow, places to visit, or restaurants to try; understand data lineage; understand patterns of fraud from transactions with users, devices, or credit cards; integrate and analyze data across silos; or enforce data access using relationships between users, data, and policies.

Wide Column (Amazon Keyspaces for Apache Cassandra)

A wide column store offers a NoSQL database wherein the data is organized into groups of columns, called column families, and can be treated as tables or two-dimensional key-value store data. Apache Cassandra is the most popular wide column store, and Amazon Keyspaces (for Apache Cassandra) is a managed database service that offers complete, compatible support for Cassandra databases and workloads. Amazon Keyspaces supports the Cassandra Query Language (CQL) as well as the drivers and developer tools built for Apache Cassandra. Amazon Keyspaces is serverless, so there is no infrastructure or software to manage. The data is encrypted and replicated three times in multiple AWS AZs for high availability. Amazon Keyspaces also enables you to create continuous backups of your table data by using point-in-time recovery.

In-Memory Data Store (Amazon ElastiCache)

Amazon ElastiCache is a fully managed in-memory data store that offers shared in-memory data management for either data sharing among application processes or holding and retrieving data using low-latency in-memory data stores. It includes support for the popular open source data cache technologies, Redis and Memcached, and is fully managed by AWS, which handles monitoring, failure recovery, backups, and other operational tasks. Amazon ElastiCache uses an end-to-end optimized stack running on customer-dedicated nodes for fast, secure performance.

Specialized Data Management

Time Series (Amazon Timestream)

Time series data requires special handling because its regular interval-based data is used for detailed statistical pattern analysis. Query and analysis need to have the appropriate math built in, and time series data must be append only and immutable. The most common use cases involve IoT device data, IT logs, and the output from smart industrial machines. Amazon Timestream is a purpose-built time series database designed from the ground up to serve this purpose. Timestream offers a built-in analytic capability and automates retention, tiering, and compression of data to minimize data management costs. It also employs a serverless architecture to match resource utilization to the application requirement.

Ledger Data Management (Amazon QLDB)

Blockchain introduced the idea of a commonly managed, immutable, and trusted transaction manager. Blockchain works great when many entities wish to exchange items in an irrefutable way but provides no means of auditing or reporting on such exchanges because to do so would compromise the anonymity of the users. For internal use in enterprises, where the ultimate owner of all such exchanges is the enterprise itself, this is not a concern, so a ledger data management system makes perfect sense for use instead of blockchain. Amazon Quantum Ledger Database (QLDB) is such a facility, maintaining a transparent, immutable, and cryptographically verifiable transaction log that maintains a verifiable history of all changes to data. As such, QLDB provides data lineage and immutability as integrated features of the transaction log – placing data integrity as an incontrovertible outcome of data storage. QLDB is also serverless, so the client only pays for what is used.

FUTURE OUTLOOK

IDC research has shown that most enterprises are only in the beginning stages of their journey to the cloud. Many of these are looking at alternatives to the data management technologies they have been using up to now. Right now, they are looking to stage that transition through hybrid cloud configurations, but over the next five years, we can expect to see large quantities of product data shifting to the public cloud. Facilities such as those described in this white paper are already driving major applications in the cloud. As larger enterprises, which to date have mostly focused on hybrid cloud deployments due to the complexity of their application environments on-premises, move their production data to the cloud, these services are likely to be the workhorses of a good portion of such enterprise data in the future and provide a suitable environment for the born-in-the-cloud applications to follow.

CHALLENGES/OPPORTUNITIES

AWS is not alone in offering a public cloud platform with integrated data management services. AWS will be challenged to continue to develop its data management offerings in ways that achieve competitive advantage. Some users and third-party application developers are opting for third-party cloud database services, rather than committing to any public cloud platform for such services, to maintain independence. It is up to AWS to demonstrate that the integrated, fully platform-managed approach to database services is superior to any other option, both now and in the future, and to offer good compute, storage, and networking building blocks for cases where the selection of AWS database services doesn't fit an application's needs.

CONCLUSION

Database management systems have been around for a very long time and have continually evolved to address the changing needs of users. As enterprises embrace cloud architectures and data-intensive workloads such as IoT ingestion and processing, smart mobile device data, log data, other streaming data, complex graphs, and time series data, in addition to the data that fits neatly in the relational domain, they are moving to adopt a variety of database models, each attuned to the needs of the application and its data. It only makes sense that while some applications are best served using multimodel database systems, most others should employ purpose-built DBMSs that deliver the best performance characteristics for the application. If an enterprise is planning to use a range of DBMSs that support differing models, that enterprise should also do so within a framework that enables the management and governance of data across those DBMSs.

In light of these facts, enterprises that are moving data to the cloud should consider the following:

- Analyze the data needs of the enterprise to determine whether a multimodel DBMS is sufficient or if a range of purpose-built DBMSs is required.
- If a range of purpose-built DBMSs is required, consider the cost and complexity factors involved in adopting DBMSs from a variety of vendors versus a single vendor with an integrated operational framework.
- In considering an integrated operational framework, if committing to a public cloud platform provider, evaluate the value of turning to that cloud platform provider for such a framework.
- When considering public cloud provider–integrated operational frameworks, and the range of DBMSs provided, the DBMSs and supporting platform of AWS should be on the list.

About IDC

International Data Corporation (IDC) is the premier global provider of market intelligence, advisory services, and events for the information technology, telecommunications and consumer technology markets. IDC helps IT professionals, business executives, and the investment community make fact-based decisions on technology purchases and business strategy. More than 1,100 IDC analysts provide global, regional, and local expertise on technology and industry opportunities and trends in over 110 countries worldwide. For 50 years, IDC has provided strategic insights to help our clients achieve their key business objectives. IDC is a subsidiary of IDG, the world's leading technology media, research, and events company.

Global Headquarters

5 Speen Street
Framingham, MA 01701
USA
508.872.8200
Twitter: @IDC
idc-community.com
www.idc.com