Annals of "Dunarea de Jos" University of Galati
Fascicle I. Economics and Applied Informatics
Years XXVI – nº2/2020
ISSN-L 1584-0409 ISSN-Online 2344-441X
www.eia.feaa.ugal.ro

DOI https://doi.org/10.35219/eai15840409128

Multi-Model Database Systems: The State of Affairs

Gianina MIHAI*

ARTICLE INFO

Article history:
Accepted August 2020
Available online August 2020
JEL Classification
M15 M49

Keywords:
SQL, NoSQL, Polyglot persistence, Multi-model database system

ABSTRACT

Nowadays, more and more organizations are coping with a very large volume of data in different formats and a very high growth rate. Under these conditions, relational databases are no longer sufficient for data management within these organizations, as has been the case for more than 30 years. Thus, in the last decade and a half, a number of new database models have emerged in order to cover the needs imposed by the storage, processing and analysis of this data, models that are identified today by the generic term NoSQL. In the present article we will present, by means of a quantitative analysis, the state of affairs of the adoption of NoSQL stores in the database management systems that already exist on the market and the convergence of SQL and NoSQL models in unitary multi-model database management systems.

1. Introduction

The large volume and variety of data in today's organizations is one of the main challenges facing database specialists. The needs of computer systems have changed a lot, especially in the last 20 years, due to the explosive increase in the use of the Internet and its specific applications. Thus, the success of organizations depends on their ability to manage, analyze and understand, in a reasonable amount of time, very large, most often unstructured, volumes of data, of different types, such as: a variety of types of documents, XML files, web content, multimedia content, data transmitted by various sensors, graph structures, etc. Under these conditions, relational databases have reached their limits, and organizations are adopting new data models that are much better suited to their needs. These new data models are generically called NoSQL and are classified in the specialized literature into four categories (Davoudian, Chen, & Liu, 2018; Hecht & Jablonski, 2011; Moniruzzaman & Hossain, 2013): key-value stores, document stores, graph stores and wide-column stores.

Since the mid-2000s, dozens of database solutions have appeared on the market, each wanting to offer the most suitable solution for a single problem: offering a very short response time with key-value stores, data management characterized by complex relationships with graph stores or semi-structured data management with document stores. Today, any new enterprise application can no longer be built by considering only the relational model for its database, but even if a relational database can still be the right solution for all or part of the application, the other recent models must also be taken into account and analysed. As any more complex enterprise application uses data of different types, integrating data from multiple sources, a single database model is no longer sufficient. Adopting and using within an organization, in different applications or even within the same application, several data models, each implemented by a different management system, greatly increases the complexity and costs of applications and systems. Thus, the need to converge these different data models appeared on the market, the recent goal being the creation of multi-model database management systems capable of providing a single database that supports several data models.

In this article, with the help of a quantitative analysis, we present the state of affairs of different data model convergence process in unitary multi-model systems. After reading the literature, in the first part of the article we present some aspects related to the concept of multi-model database system, and in the second part we present the results of quantitative analysis on the actual presence of NoSQL database management systems present on the market and the convergence process of multiple data models into unitary multi-model systems.

*Dunarea de Jos University of Galati, Romania. E-mail address: grizescu@yahoo.com

2. The multi-model database concept

The term multi-model database system is not in fact new, it was also used when relational databases were prevalent. In the article published in the Conference of Computers and Communications in 1989, Demurjian, S. and Hsiao, D. state that a multi-model database system *"offers an environment where a user is able to access all types of databases (i.e., relational, hierarchical, network, functional or attribute-based) via the single data model/data language combination that the user is most familiar with"* (Demurjian & Hsiao, 1989). In 1982, Lien, E. pointed out the consensus that was beginning to emerge that no database model was inferior or superior to all applications, stating that *"it seems desirable to offer all three models (our note: relational, hierarchical, and network) in one DBMS in order to satisfy different user groups"* (Lien, 1982). Consequently, the problem is not new, the concept is not new, only the trend is brought back into focus.

Multi-model database is an emerging trend in the development of database management systems generated by the proliferation of NoSQL data models (key-value stores, document stores, column stores, and graph stores) (Zhang & Lu, 2019; Lu & Holubova, 2019; Liu, Lu, Gawlick, Helskyaho, Pogossiants, & Wu, 2018).

There are two solutions for using several data models within an organization (Lu & Holubova, 2017; Lu & Holubova, 2019):

- polyglot persistence - the use of several database management systems, one for each model;
- multi-model database system – the use of a single database management system to support multiple data models;

First, the increasing adoption and use of NoSQL stores has brought into focus the concept of "polyglot persistence" (Fowler, 2011), according to which organizations use multiple database systems to manage different data models, which must be programmatically integrated at the application level. However, it is extremely difficult for all these different systems to be made to work well together, resulting in an extremely complex technical infrastructure that is difficult to manage with direct consequences on performance, persistence and fault tolerance. In addition, this solution is very expensive both from a technical point of view and from the point of view of human resources, requiring specialists who know all the systems used and/or specialists for each individual system.

Unlike the first approach, the second one refers to multi-model database management systems capable of integrating and managing different data models to meet the various needs of the organization. Multi-model database systems are considered to be the new generation of database management systems capable of providing flexibility, scalability and consistency (Zhang & Lu, 2019).

3. Methodology

In order to understand and present the concept of a multi-model database, the specialized literature published in the international scientific databases was consulted and analysed. The site db-engines.com (https://db-engines.com/) was used to analyse the state of affairs of multi-model database management systems existing on the market. This site establishes a monthly ranking of database management systems according to their popularity, by taking into account several parameters, namely: number of results in Google and Bing search engines queries, frequency of searches in Google Trends, frequency of technical discussions on professional sites such as Stack Overflow and DBA Stack Exchange, the number of jobs offered on Indeed and Simply Hired job search engines, the number of LinkedIn profiles and the number of tweets on Twiter (db-engines.com, 2020). The data taken from this site have been summarised in tables, designed to be as suggestive as possible for the purpose of our analysis.

4. Research and findings

The mid-2000s were marked by the enthusiasm of a part of the industry that hoped to completely replace or at least drastically reduce the dominance of relational databases and of their universal language - SQL. This enthusiasm was amplified by the previous disappointment caused by object-oriented databases in the mid-1990s, which failed to become an alternative to relational databases, as expected (Harrison, 2015, p. 13). During these years, a multitude of dedicated systems have been developed for a certain type of NoSQL data model such as: Memcached for key-value stores (initial release in 2003), CouchDB for document stores (initial release in 2005), Neo4J for graph stores (initial release in 2007) or Cassandra for column stores (initial release in 2008). Very soon, however, the industry realized that this limitation to a single data model greatly restricts their scope because today's complex applications require large-scale performance, flexibility, and scalability that cannot be achieved by using a single data model. Thus, voices appeared (Pimentel, 2013; Pimentel, 2015) announcing a rapid and complete convergence of all these models in unitary systems. Also, in 2015, Gartner specialists predicted that "*by 2017, all leading operational DBMSs will offer multiple data models, relational and NoSQL, in a single platform"* (Feinberg, Adrian, Heudecker, Ronthal, & Palanca, 2015).

It is true that, since the advent of these NoSQL data models, there have been concerns about the development of databases that integrate more such models. Analysing the evolution of database management systems, three trades on the convergence of data models into single databases can be delineated:

First trend - established relational database systems have made efforts to integrate NoSQL data models;

Second trend - NoSQL stores, initially built on a single model, later integrated/integrate other NoSQL models and/or the relational model;

Third trend - represented by the development of systems that are built from the beginning to integrate multiple NoSQL models and/or relational model.

First trend

The first convergence efforts were made by traditional relational database management systems that reacted quickly by providing support for new data models. This rapid adaptation is primarily due to the simplicity, universality and rigor of the relational model that could be extended to other data models and due to the possibility of extending the SQL standard to data presented in other formats such as XML or JSON (Lu & Holubova, 2019).

Oracle is a leader in this case, too, as it offers, even today, very strong support for encapsulating JSON documents in the relational database by storing them in LOB or character columns that can be accessed via SQL extensions or directly via a REST-based interface, integrating a graph compute engine, which provides support for scalable property graph database, a graph query language (OpenCypher) and an API developer (Oracle, 2020). Oracle qualifies its database as multi-model, starting with version 18c (Oracle, 2018).

In table 1 we present the NoSQL models adopted by the top 10 most popular relational database management systems, according to the ranking made by the db-engines.com website. We did not include in the table the Microsoft Access, SQLite and Hive systems, which do not offer support for any type of NoSQL stores. As shown in Table 1, at present, the most popular relational systems support NoSQL stores, especially document stores and/or graphs or RDF stores. Regarding the overall situation, out of the total of 123 relational systems included in the db-engines.com ranking, only 24 systems (19.5%) integrate other NoSQL data models.

Table 1. NoSQL stores' adoption in RDBMSs

	Oracle	MySql	SQL Server	PostgreSQL	Db2	MariaDB	Teradata
General Rank	1	2	3	4	6	12	14
RDBMSs Rank	1	2	3	4	5	8	9
Key-value	-	-	-	-	-	-	
Document	√	√	√	√	√	√	√
Graph	√	-	√	-	-	√	√
RDF	√	-	-	-	√	-	-
Columns	-	-	-	-	-	-	-

Source: processing performed using data provided by the site db-engines.com (August 2020)

Second Trend

Regarding the second trend, the most popular NoSQL systems that became the leader for the implemented data model, hesitated to adopt other models. Table 2 shows the top ten most popular such NoSQL store systems. As can be seen in the mentioned table, out of the 10 systems, only 3 integrate secondary data models. It should be noted that none of them integrates the relational model. This shows that many want to develop in their niche sector or do not yet have the power of convergence. This is proven by the fact that they managed to gain a lot in popularity given that the first most popular systems for their basic model are also in the top 20 in the overall ranking. Regarding the overall situation, out of the 112 such systems, only 13 (11.6%) integrate other models than the basic one.

Table 2. Relational and other NoSQL stores' adoption in NoSQL DBMSs

	Initial release	General Rank	Primary model Rank	Relational	Key-value store	Document store	Graph store	Column store
MongoDB	2009	5	1			■		
Redis	2009	7	1		■	√	√	
Cassandra	2008	11	1					■
Neo4j	2007	21	1				■	
HBase	2008	22	2					■
Couchbase	2011	26	4		√	■		
Memcached	2003	27	4		■			
CouchDB	2005	35	5			■		
Firebase Realtime DB	2012	37	6			■		
Hazelcast	2008	43	5		■	√		

Source: processing performed using data provided by the site db-engines.com (August 2020)

Legend: ■ – primary model
√ - other integrated models

Third trend

After 2012, multi-model systems began to be developed, which are initially built to use multiple data models. Table 3 presents the top ten most popular such systems together with the data models they integrate, the place occupied for each model and the general place in the ranking. Of the 41 such systems present in the db-engines.com ranking, the most popular are Amazon Dynamo and Azure Cosmos, both being cloud-based systems only. With the exception of these two, all the others did not manage to gain very good positions in the ranking on models or the general one, which shows us the incipient stage of their adoption and use. It can also be seen that most systems integrate only two different data models, fact which makes us assert that the convergence of all models into a single database is a process that will take some time.

In a synthetic image, out of the 41 such multi-model systems, 18 (43.9%) support the relational model, 17 (41.5%) support key-value stores, 18 (43.9%) provide support for document stores, 15 (36.6%) support graph stores and 3 (7.3%) support column-wide stores.

Table 3. DBMSs primarily classified as multi-model

	Initial release	General Rank	Relati-onal	Key-value store	Document store	Graph store	Column store
Amazon DynamoDB	2012	16		√	√		
Rank per model				2	2		
Microsoft Azure Cosmos DB	2014	25		√	√	√	√
Rank per model				3	3	2	3
MarkLogic	2001	41			√[1]	√[2]	
Rank per model					7	1[2]	
ArangoDB	2012	62		√	√	√	
Rank per model				9	10	3	
Ignite	2015	71	√	√			
Rank per model			39	11			
OrientDB	2010	72		√	√	√	
Rank per model				12	12	4	
InterSystems Caché[3]	1997	85	√	√			
Rank per model			43	14			
Oracle Berkeley DB	1994	86		√	√[4]		
Rank per model				15	2[4]		
Apache Drill	2012	94	√		√		
Rank per model			47		17		
Virtuoso	1998	105	√		√[4]	√[5]	
Rank per model			51		3[4]	5	

Source: processing performed using data provided by the site db-engines.com (August 2020)

1 - integrates Native XML DBMS too (rank 1), 2 RDF store only, 3 – No. 1 Object Oriented DBMS, 4 - Native XML DBMS only, 5 - integrates RDF store too (rank 3)

Among these top 10 most popular multi-model systems (Table 3) there are some systems that have been on the market for a long time, such as Oracle BerkeleyDB (1994), InterSystem Chache (1997), Virtuoso

(1998) and MarkLogic. (2001) but, as explained by the db-engines.com classification criteria, they were classified as multi-model systems that have recently integrated new data models, but were not previously established or frequently used for their basic model (Andlinger & Gelbmann, 2019) or had earlier concerns for the integration of several data models such as InterSystem Chache, which was the best-known object oriented system that also integrated the relational model, or MarkLogic which entered the market in 2001 as a native XML database, and since 2008 it has been providing support for JSON documents and, since 2013, for RDF stores.

The prediction made by Gartner specialists in 2015 (Feinberg, Adrian, Heudecker, Ronthal, & Palanca, 2015) came true. Nowadays, there are many systems, both mature and established on the market and new ones, which provide support for more than one data model, but a total convergence of data models into mature and usable unitary systems, on a large scale, will take a while. Even if this convergence occurs, the multitude of solutions on the market will continue to make it extremely difficult for organizations and specialists to choose the most appropriate solution, thus necessitating a process of market consolidation in order to distinguish a reasonable number of reliable solutions from which organizations could choose. As the best solutions are chosen by the market, this is also a long-lasting process.

5. Conclusions

The world of databases has started in an irreversible direction. It is clear that we will no longer encounter the comfortable situation when we had to choose between a few relational database management systems, starting from a few questions that we could answer quite easily and quickly: What is the volume of data and what will be its growth level?, What is the most cost-effective solution?, and so on. As we have tried to show by means of this quantitative analysis, the multitude of solutions on the market today, each of them addressing and being effective for certain needs, makes it very difficult to choose the most appropriate solution given that the needs of real systems in organizations most often than not do not meet those covered by a single solution. This is why, a convergence process of all these types of data models is absolutely necessary, but it is not simple and, as we have shown in this article, it seems to take longer than anticipated.

In addition to the large number of solutions on the market, besides convergence, there is a need for market consolidation. Many of the systems that exist on the market today are likely to disappear and, as the state of affais of multi-model database systems now show, it will probably take longer for unified multi-model systems to be adopted and used on a large scale, as was the case of relational database systems.

References

1. *Andlinger, P., & Gelbmann, M. (2019, February 19). New options in the DB-Engines ranking for multi-model DBMS. Retrieved on August 2, 2020, from https://db-engines.com/: https://db-engines.com/en/blog_post/80*
2. *Davoudian, A., Chen, L., & Liu, M. (2018). A Survey on NoSQL Stores. ACM Computing Surveys , 41 (2).*
3. *db-engines.com. (2020). Method of calculating the scores of the DB-Engines Ranking. Retrieved on August 2, 2020, from https://db-engines.com: https://db-engines.com/en/ranking_definition*
4. *Demurjian, S., & Hsiao, D. (1989). The multi-model database system. Computers and Comunications, (pg. 439-445). Phoenix.*
5. *Feinberg, D., Adrian, M., Heudecker, N., Ronthal, A. M., & Palanca, T. (2015). Gartner Magic Quadrant for Operational Database Management Systems. Gartner.*
6. *Fowler, M. (2011, November 16). Polyglot Persistence. Retrieved on August 23, 2020, from martinfowler.com: https://martinfowler.com/bliki/PolyglotPersistence.html*
7. *Harrison, G. (2015). Next Generation Databases - NoSQL, NewSQL, and BigData. Apress.*
8. *Hecht, R., & Jablonski, S. (2011). NoSQL Evaluation: A Use Case Oriented Survey. 2011 International Conference on Cloud and Service Computing, (pg. 336-341). Hong Kong.*
9. *Lien, E. (1982). On the Equivalence of Database Models. Association for Computing Machinery , 29 (2), 333-362.*
10. *Liu, Z. H., Lu, J., Gawlick, D., Helskyaho, H., Pogossiants, G., & Wu, Z. (2018). Multi-Model Database Management Systems - a Look Forward. În M. T. Gadepally V., Heterogeneous Data Management, Polystores, and Analytics for Healthcare. Lecture Notes in Computer Science (Vol. 11470). Springer.*
11. *Lu, J., & Holubova, I. (2017). Multi-model Data Management: What's New and What's Next? Proceeding of the 20th International Conference on extended Databases, (pg. 602-605). Venice.*
12. *Lu, J., & Holubova, I. (2019). Multi-model Databases: A New Journey to Handle the Variety of Data. ACM Computing Surveys , 52 (3).*
13. *Moniruzzaman, A. B., & Hossain, S. A. (2013). NoSQL Database: New Era of Databases for Big data Analytics -. International Journal of Database Theory and Application , 6 (4), 1-14.*
14. *Oracle. (2018, March). Multimodel Database with Oracle Database 18c. Retrieved on August 15, 2020, from oracle.com: https://download.oracle.com/otndocs/products/spatial/pdf/Multimodel_Database_with_Oracle_Database_18c.pdf*
15. *Oracle. (2020). Oracle as a Property Graph. Retrieved on August 20, 2020, from Oracle.com: https://www.oracle.com/database/technologies/spatialandgraph/property-graph-features.html*
16. *Pimentel, S. (2013, October 28). Polyglot Persistence or Multiple Data. Preluat pe August 10, 2020, de pe http://www.odbms.org/: http://www.odbms.org/wp-content/uploads/2014/04/Multiple-Data-Models.pdf*
17. *Pimentel, S. (2015, January 6). The rise of the multimodel database. Retrieved on August 17, 2020, from https://www.infoworld.com/: https://www.infoworld.com/article/2861579/the-rise-of-the-multimodel-database.html*
18. *Zhang, C., & Lu, J. (2019). Holistic evaluation in multi-model databases benchmarking. Distributed and Parallel Databases .*