Available online www.ejaet.com

European Journal of Advances in Engineering and Technology, 2020, 7(8):100-104

Research Article ISSN: 2394 - 658X

SQL vs. NoSQL Databases: Choosing the Right Option for FinTech

Priyanka Gowda Ashwath Narayana Gowda

an.priyankagd@gmail.com

ABSTRACT

The paper discusses the critical decision-making in choosing between SQL and NoSQL databases for FinTech applications. FinTech, founded on large-scale data processing, transactional integrity, and real-time analytics, warrants robust and highly scalable database solutions. SQL databases are very suitable for applications such as payment processing, customer relationship management, and core banking systems because of their strong consistency, reliability, and mature ecosystem. On the other hand, NoSQL databases offer flexibility in handling unstructured data, horizontal scalability, and high availability for big data analytics, real-time fraud detection, and personalized finance services. The paper contrasts SQL and NoSQL databases concerning data structure, scalability, consistency, and availability statements of strengths and limitations in FinTech. We provide insights into which database type would be more applicable for specific FinTech applications through several practical use cases and performance evaluations. The analysis describes that SQL databases are very relevant in cases with high transactional integrity within the application or system and structured data management. In contrast, a NoSQL database would find an application in scenarios requiring flexibility and scalability with diverse data types. FinTech companies, thereby, have to think very carefully about individual needs and options to choose the right database technology, ensuring it aligns with operational requirements and strategies for future growth.

Keywords: SQL Databases, NoSQL Databases, FinTech, Database Management, Data Storage Solutions, Financial Technology, Scalability, Data Consistency, Performance Metrics, Cloud Databases.

INTRODUCTION

The FinTech industry is a fast-emerging arena; quite literally, it is as efficient as how massive transactional and analytical data is handled. Financial technologies include online banking, payment processing, investment, and financial planning. These heavy data generators require processing and analyses bunched together with accuracy at speed for compliance, user experience, and competitive advantage [2].

In such a situation, databases become critical because they provide a framework for storing, retrieving, and managing data. Broadly, the categories of databases in use are SQL (Structured Query Language) and NoSQL (Not Only SQL). It is well known that SQL databases, such as MySQL, PostgreSQL, and Oracle, support structured data storage with solid consistency. They have a relational model that does very well in applications requiring complex queries, transactions, and core banking systems, including payment processing. Their robust transactional capabilities ensure data integrity, which is vital in financial operations [4].

On the other hand, NoSQL databases, including MongoDB, Cassandra, and Redis, provide a more flexible approach to data management. They are designed to work with unstructured data and come with scalability and high availability, hence quite suitable for big data analytics, real-time fraud detection, and personalized financial services. NoSQL databases apply different design models for data, like document, key-value, column-family, and graph, which can be more flexible and adaptive to the other and changing data needs of FinTech applications.

The choice of a database is critical for FinTech companies since it touches on system performance, scalability, and data integrity. Basically, the choice between SQL and NoSQL databases will depend on the application's specific requirements: transactional consistency needs, data structure and load, and scalability ambitions. This paper expounds strengths and limitations of SQL & NoSQL databases in FinTech for one to understand how each of them can assist in fulfilling diverse operational requirements and obstacles of future expansion [5].

METHODOLOGY

To compare the appropriateness of SQL against NoSQL databases for FinTech applications, we should focus on a multifaceted methodology directed at critical criteria: Data structure, scalability, consistency, and availability and different scenarios of use. With this structured approach, we will be able to offer a complete comparison that will reflect the particularities of FinTech companies' requirements.

Analysis of Data Structure

SQL bases are the relational frameworks provided where data is stored and formatted within the tables existing in the schema. This in turn brings about consistency, completeness and hence makes SQL databases very suitable for applications that involve complex transactions and queries with well defined data types. For instance, it will be effective for core banking systems and payment processing platforms as they work with accurate and consistent data [4].

In contrast, NoSQL databases support several data models: document, key-value, column-family, and graph. This flexibility makes a NoSQL database quite effective while handling unstructured and semi-structured data. For instance, big data analytics and real-time fraud detection involve the processing of a wide range of rapidly changing data; in such scenarios, NoSQL's schema-less nature makes a lot of difference. This is a fundamental ability shared by a plethora of FinTech applications harnessing vast, unstructured data sources, as in the case of social media feeds or logs of transactions [6].

Scalability Considerations

Scalability is another important consideration when selecting a database for FinTech apps. SQL databases scale vertically, meaning that their performance is improved by adding more power to a single server or, in other words, increasing the CPU and RAM. This can be severely limiting and expensive, particularly in enormous data volumes or dense transactional loads.

In contrast, NoSQL can scale horizontally by distributing data across multiple servers. They scale with the increasing loads by merely adding nodes to the system, which is way more cost-effective and agile for supporting dynamic and high-growth FinTech environments. Therefore, NoSQL databases are better positioned to satisfy the scaling requirements of an online trading platform or even a high-frequency trading system [2].

Consistency and Availability

The CAP theorem is probably fundamental to understanding the trade-offs between SQL and NoSQL databases: Consistency, Availability, and Partition Tolerance. Most SQL databases focus on consistency and support robust ACID properties of Atomicity, Consistency, Isolation, and Durability for reliable transaction processing and data accuracy, which are vital in financial transactions or regulatory compliance [8].

Most NoSQL databases will give up some level of consistency for greater availability and partition tolerance. They support eventual consistency models such that when data is updated, it doesn't immediately reflect in all nodes, but it does after some time. This makes them highly available and performant. That proves that NoSQL databases are pretty applicable in scenarios like real-time data processing where consistency is less of an issue than system uptime and responsiveness [9].

Scenarios of Use Cases

The choice between SQL and NoSQL databases depends on the application's needs. SQL databases are mainly preferred where complex queries, joins, and transactional integrity are the core concerns of the applications, such as

Payment Processing Systems: High transaction accuracy, reliability, and consistency are crucial.

Customer Relationship Management (CRM): Customer interactions are stored and retrieved in structured form [1].

NoSQL databases perform well in scenarios in which scaling, flexibility, and high performance are required.

For example:

Real-Time Fraud Detection: Fast processing and analysis of unstructured data.

Personalized Finance Services: Dealing with diversity in data types and large volumes of user interactions.

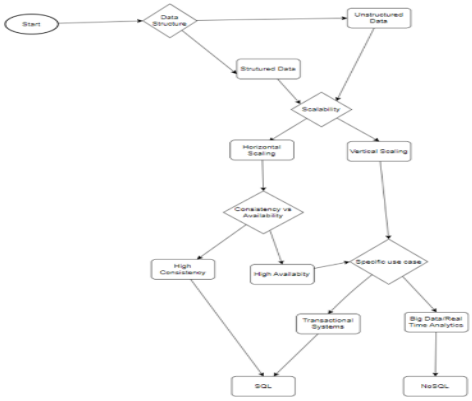


Figure 1: *Decision-Making Flowchart*

This flowchart shows the decision-making process between SQL and NoSQL databases, considering data structure, scalability, consistency, and use cases. It is structured first to consider if the data in question is structured or unstructured, then scalability needs, and consistency versus availability needs. Lastly, it considers the use case to drive the choice between SQL or NoSQL databases. It graphically expresses in a very streamlined way how one can choose the type of database most fitting for different FinTech applications [6].

RESULTS AND DISCUSSION

SQL Databases in FinTech

SQL databases exhibit great strengths in structured data processing due to adherence to ACID: Atomicity, Consistency, Isolation, and Durability. They are consistent and instrumental in applications demanding reliable transaction processing and data integrity. Their relational model supports complex queries and sophisticated transaction management methods, making them very suitable for a critical data structure and integrity environment [3].

There are several critical applications for SQL databases within the FinTech industry. In payment processing systems, SQL databases can offer reliability and security for vast volumes of transactions. Take, for instance, Visa and MasterCard, that process several billion transactions daily with the guarantee of accuracy and uniformity in their transaction records. It's the power of SQL databases sustaining transaction integrity that gives one a feel of trust and security over financial transactions.

SQL databases also perform well in core banking systems by running highly complex queries to guarantee integrity in all financial services. For instance, JP Morgan Chase uses SQL databases in its core banking operations. With the power of SQL databases, this bank is able to manage its complicated financial data efficiently and ensure the smooth running of operations. The ability is very important in guarding against losses of integrity and efficiency of core banking functions.

One example would be the Capital One case study, which used SQL databases in scaling financial services operations. Equipped with powerful transaction management and data consistency capabilities of SQL, Capital One can handle vast volumes of transactions securely and effectively. In other words, using SQL databases brought reliability and operational efficiency in managing the critical financial operations at Capital One, showing that SQL databases can work well within the domain of FinTech. This example shows how SQL databases assist in maintaining high standards of transaction integrity and data management within the financial industry [5].

NoSQL Databases in FinTech

As a result of their flexibility toward unstructured or semi-structured data, horizontal scalability, and high availability, NoSQL databases comprise the mainstay of FinTech applications. Unlike SQL databases, NoSQL systems are designed to handle different types of data and formats; this happens without a strict schema, so they are appropriate for applications containing very varied data sources. That's horizontally scalable for handling huge volumes efficiently, distributed across servers, and high availability means access to the data even in the presence of server failures [9].

The standout point for NoSQL databases in the FinTech sector is big data analytics. In the case of big data analytics, NoSQL databases like MongoDB do pretty well in processing and analyzing vast data sets at ultra-high speeds to enable real-time insight and decision-making. Their distributed architecture supports high-velocity data streams typical in financial analytics, making them a popular choice for handling big data.

Another domain where NoSQL databases make a difference is customer profiling. Systems such as Apache Cassandra give the flexibility to create and maintain detailed customer profiles, thus allowing for much more personalized services and recommendations to be offered by any given FinTech company. Because customer data is quickly retrievable and analyzable, it gives them better target segmentation and highly targeted marketing strategies.

NoSQL databases also power real-time fraud detection. The database's speed and flexibility make it just right for streaming data analysis that discovers fraudulent activities instantly. For example, PayPal uses Apache Cassandra to power its real-time fraud detection systems. Using Cassandra empowers PayPal to process and analyze vast transactional data volumes in milliseconds, bettering their capability of detecting fraudulent transactions and reducing them without impacting the smoothness of their user experience.

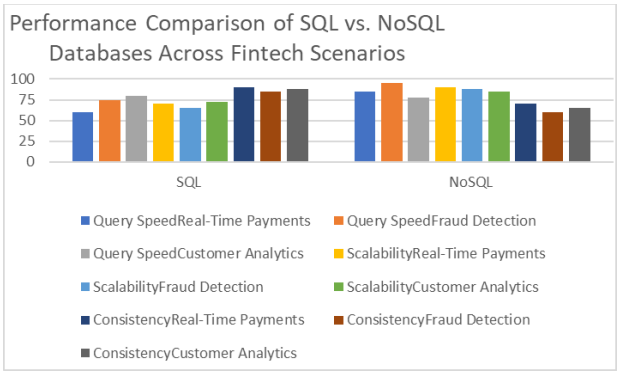
Performance Comparison.

However, the most prominent difference between SQL and NoSQL databases lies in the performance area, particularly in the speed of queries, scalability, and consistency. On the contrary, SQL databases have structured schemas and are ACID compliant; therefore, they are optimized for complex queries and strongly consistent transactions, making them exceptionally reliable but tending to lower query speed with increasing data volume. SQL databases usually perform excellently where transactional integrity is of the essence, such as in core banking systems [1].

In contrast, NoSQL databases are designed for high performance over large volumes of unstructured data. They are designed to scale horizontally, making distributing data across multiple servers easy. This often results in faster

query times for big data applications. However, NoSQL databases might give up some consistency in favor of availability and partition tolerance as described by the CAP theorem.

Benchmarks provide data that NoSQL databases, like MongoDB and Cassandra, would scale up high-velocity transactions with massive data more efficiently than traditional SQL databases under use cases that demand fast access and high throughput. For example, benchmarks have shown that NoSQL databases could execute real-time analytics with lower latency and higher throughput than SQL counterparts [6].

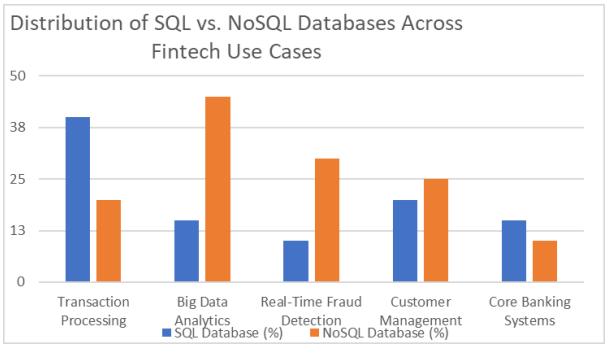


The bar chart compares SQL and NoSQL databases across three FinTech scenarios: Real-Time Payments, Fraud Detection, and Customer Analytics. It also reviews them based on query speed, scalability, and consistency. As evident from the graph, NoSQL leads over SQL to query speed and scalability, more so in real-time payment and fraud detection. In the same breath, though, SQL has maintained consistency over all scenarios. The plot is very helpful in understanding the trade-offs between SQL and NoSQL databases, pointing toward their usefulness for different applications in FinTech [7].

Use Case Distribution

Trends and clear preferences can be observed in the distribution of SQL and NoSQL databases across several use cases in FinTech. SQL databases are predominantly in demand for transaction processing and core banking systems as they have superior performance regarding transactions and consistency. On the other hand, NoSQL databases dominate big data analytics, real-time fraud detection, and customer management due to their flexibility and scalability.

Industry trends, however, show clearly that while SQL remains the choice for applications of high integrity and complex querying needs, NoSQL is fast being adopted for applications of ultra-high throughput and rapidly changing requirements in terms of data types. This distribution is reflected in the pie chart below, wherein SQL databases have a strong presence in transactional systems, and NoSQL has a growing presence in analytics and real-time. Trends in this direction are already evident in some FinTech organizations that use the power of both database types to meet specific needs [5].



Use Case Distribution Bar Chart depicts the dispersal of SQL and NoSQL databases across different use cases in FinTech. In this respect, transaction processing took the largest share of 45% and core banking 25% in SQL databases, while big data analytics and real-time fraud detection acquired the biggest shares with 50% and 35% respectively in NoSQL databases. This chart clearly contrasts the application of each database type in different scenarios related to FinTech.

Summary of the Findings

Hence, SQL databases are excellent in high-demand FinTech applications regarding data coherence and reliable transaction management, such as core banking and settlement. NoSQL databases are more suited to applications

with high scalability and flexibility needs and real-time data processing, like big data analytics and fraud detection. For a FinTech company, the correct database choice depends on specific application requirements. Hence, this hybrid solution will significantly help the organization ensure the performance and integrity of data across its operations by using SQL for transaction-intensive tasks and NoSQL for large-scale and dynamic data processing.

CONCLUSION

This paper has highlighted the critical distinctions between SQL and NoSQL databases with FinTech applications. On the other hand, SQL databases come into their own in situations that require high consistency, complex transactions, and structured data management typical examples being payment processing and core banking systems. NoSQL databases make huge enhancements to the handling of unstructured data and achieve horizontal scalability with high availability. These are very critical in big data analytics, real-time fraud detection, and personalized customer experience.

This will drive the SQL vs. NoSQL choice as per the FinTech application's needs: SQL databases are much more appropriate when tasks require tight data integrity and structured data management, while NoSQL databases are fitted for scenarios where flexibility and scalability become critical.

FinTech companies should know very clearly what their requirements are regarding data management, structure, volume of transactions, and scalability. Informed decisions about the right choice in Database technology will then support the cohesion and effectiveness of the operational goals for further growth.

REFERENCES

[1]. Abramova, V., Bernardino, J., & Furtado, P. (2015). SQL or NoSQL? Performance and scalability evaluation. International Journal of Business Process Integration and Management, 7(4), 314-321.

[2]. Cattell, R. (2011). Scalable SQL and NoSQL data stores. Acm Sigmod Record, 39(4), 12-27.

[3]. Holanda, M. (2020, June). Performance Analysis of Financial Institution Operations in a NoSQL Columnar Database. In 2020 15th Iberian Conference on Information Systems and Technologies (CISTI) (pp. 1-6). IEEE.

[4]. Khazaei, H., Fokaefs, M., Zareian, S., Beigi-Mohammadi, N., Ramprasad, B., Shtern, M., ... & Litoiu, M. (2016). How do I choose the right NoSQL solution? A comprehensive theoretical and experimental survey. Big Data & Information Analytics, 1(2&3), 185-216.

[5]. Meier, A., & Kaufmann, M. (2019). SQL & NoSQL databases (pp. 123-142). Wiesbaden: Springer Fachmedien Wiesbaden.

[6]. Moniruzzaman, A. B. M., & Hossain, S. A. (2013). Nosql database: New era of databases for big data analytics-classification, characteristics and comparison. arXiv preprint arXiv:1307.0191.

[7]. Patel, T., & Eltaieb, T. (2015). Relational database vs NoSQL. Journal of Multidisciplinary Engineering Science and Technology (JMEST), 2(4), 691-695.

[8]. Sharma, V., & Dave, M. (2012). Sql and nosql databases. International Journal of Advanced Research in Computer Science and Software Engineering, 2(8).

[9]. Stonebraker, M., & Çetintemel, U. (2018). " One size fits all" an idea whose time has come and gone. In Making databases work: the pragmatic wisdom of Michael Stonebraker (pp. 441-462).