WORLD COMET RESEARCH GROUP

International Journal of Applied Data Science & Modern Computing
Vol. 6, No. 2, 2023
Doi: 10.67228/30713498/IJADSMC-2023PII0Q6V
PP. 01-14

Original Article

A Review of Modern Database Systems for Data Science Applications

Dr. Ahmed Ben Ali

Department of Mechanical Engineering, Tunis National Engineering College, Tunisia.

Received: 11-02-2023 Revised: 11-28-2023 Accepted: 12-02-2023 Published: 12-04-2023

ABSTRACT

Modern data mining applications require scalable and high-performance database systems capable of handling structured, semi-structured, and unstructured data. Traditional RDBMS, designed for transactional consistency, face limitations in scalability and performance for big data analytics. As a result, modern systems such as NoSQL, NewSQL, distributed file systems, and cloud-native platforms have emerged, offering features like horizontal scalability, schema flexibility, and real-time processing. This paper provides a comprehensive overview and comparison of these database systems, focusing on their architecture, data models, and suitability for analytical workloads. It also examines their role in data science pipelines, including data ingestion, preprocessing, model training, and deployment. Key trade-offs based on the CAP theorem are discussed, along with performance metrics such as scalability, latency, and fault tolerance. The study highlights that no single database solution fits all scenarios, and hybrid architectures with polyglot persistence are increasingly adopted. The paper concludes by identifying research gaps in database optimization for machine learning, data governance, and real-time analytics, offering guidance for selecting appropriate systems in data-driven environments.

KEYWORDS

Database Systems, Data Science, NoSQL, NewSQL, Big Data Analytics, Distributed Databases, Cloud Computing.

WCRG
Global Research
ISSN: 3071-3498
PP: 01-14.

1. INTRODUCTION

1.1. Background

Ranging from the techniques of statistics, machine learning, and artificial intelligence, data science has appeared a leading pillar of the modern information systems sector of an organization, where organizations are able to draw their valuable knowledge and actionable insights out of extensive and complicated datasets. The growth of data sources including: social networking sites, Internet of Things (IoT) devices, mobile apps, and massive enterprise systems has resulted in a tremendous surge in the amount and diversity of data being produced on an ongoing basis. Such a burst of activity has offered great strain on the current data storage and management systems which currently have to fulfill not only the duties of reliable data persistence, but also achieve scaleability in processing and analysing data. Consequently, database technologies became a significant enabler of data-based decision-making in the fields of medical and other healthcare, finance, scientific research, as well as e-commerce. The relational data model has been used to form the basis of traditional database management systems and focuses on having structured schemas, data is organized in a normal manner and offered strong guarantees of transaction so as to maintain data integrity and consistency. Such properties cause relational systems to be especially efficient in Online Transaction Processing (OLTP) programs (found in banking system or inventory administration platforms) where it is vital to record keeping accuracy and reliability. Nevertheless, the nature of workloads in data science is quite different as opposed to traditional transactional workloads because they constitute huge amounts of data, the flow of data into the data warehouse, and intricate analytical processes that can occasionally involve frequent scans and data transformations. The three core characteristics attributed to big data such as high volume, high velocity, and high variety can be used to describe such workloads. In such circumstances, relational databases are often constrained by such factors as horizontal scalability, query response, and flexibility to support unstructured or semi-structured data. This choice outcome has led to the emergence of new paradigms and architectures of data bases in response to increasing discrepancies between the functionality of traditional relational systems and the needs of the new data science applications. These new systems aim to transcend the limitations of inflexible schemas and centralized processing and adopt distributed architecture, lax consistency model and flexible data representations. A particular need to expand upon the relevance of the developing database technologies in serving the data science processes, as well as avoid the problems of rapidly growing and heterogeneous data space, is the background of the present study.

1.2. Review of Modern Database Systems

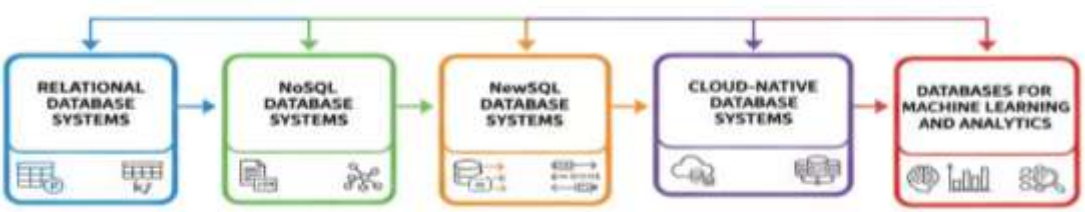

Fig 1 - Review of Modern Database Systems

ISSN: 3071-3498
PP: 01-14.

1.2.1. Relational Database Systems

Relational database management systems (RDBMS) have enjoyed a long history of backing data storage and management since it has a structured data model, clearly defined schemas and a high degree of transactional guarantees. They are built on tables, primary keys, and foreign keys to indicate the relationships and impose a data integrity using the ACID (Atomicity, Consistency, Isolation, Durability) properties. The systems are very reliable and best fitted with Online Transaction Processing (OLTP) applications in which accuracy and uniformity are extremely important. Nevertheless, their reliance on fixed schemas and centralized or vertically-scaled architectures restricts them in managing the huge, dynamically changing, and heterogeneous volumes of data of modern data science loads.

1.2.2. NoSQL Database Systems

NoSQL databases have been developed to overcome the weaknesses of the relational systems in terms of scale and flexibility. They are able to store semi-structured and unstructured data in a more natural way due to their support of many different data models such as key–value, document-oriented, column-family, and graph-based data models. NoSQL databases are more focused on high availability and horizontal scalability by distributing data and replicating it over distributed nodes. Many NoSQL systems compromise on strict consistency and transfer to eventual consistency models in order to attain these properties. This is why they are applicable to use in scenarios where massive data load is necessary, and real-time analyses are needed, but it might necessitate additional controls to handle data accuracy and integrity.

1.2.3. NewSQL Database Systems

The NewSQL databases seek to achieve a combination of the features of both relational and NoSQL database systems based on the concept of maintaining the relational data model and SQL interface, but adding distributed architecture to offer scalability and performance. These systems facilitate the use of ACID transactions and allow high throughput transactions using methods like in-memory storage, partitioned data management and optimized concurrency control. NewSQL systems are constructed to provide better performance to both transactions and analytics loads compared to traditional RDBMSs, therefore, being desirable to those applications with high consistency requirements as well as massive data processing.

1.2.4. Cloud-Native Database Systems

Cloud-native databases have become a major improvement to the design of database using cloud computing infrastructures. They are generally decoupled storage/compute systems that exploit massively parallel processing (MPP) to process queries at multiple nodes simultaneously. Elastic scaling enables the provision of resources dynamically according to the workload requirements in a fashion that is more cost-effective and efficient. Cloud-native architectures have been found to be especially attractive to analytical workloads, including data warehousing and business intelligence, where large volumes of data and complicated queries need to be optimally executed.

ISSN: 3071-3498
PP: 01-14.

1.2.5. Databases for Machine Learning and Analytics

In current database systems, machine learning and sophisticated analysis-related services are more and more embedded in the layer of data management. This unification lowers the standard of migrating massive datasets to outside processing systems thus reducing latency and overhead. Database-native machine learning and in-database analytics enable the use of iterative algorithms, as well as inference of models in real time allowing closer approximation between data storage and intelligent processing. These changes exhibit a wider transition of database systems as a passive data repository to an active system which underlies a data driven and intelligent application. Collectively, the above modern database paradigms demonstrate the diversification of the data management technologies with respect to changing application needs. The challenges discussed in each type of systems are focused on addressing such aspects as scalability, flexibility, consistency, and analytical performance, which emphasize the role of selecting suitable database architectures depending on the workload features and the purpose of the application.

1.3. Modern Database Systems for Data Science Applications

Contemporary database systems are becoming more and more directly embedded with machine learning functionality and high-end analytical features into the data management layer. The integration is less encumbersome in terms of large datasets being transferred to more external processing structures, thus reducing latency and overhead. Database-native machine learning and in-database analytics offer the ability to utilize iterative algorithms and real-time model deriving to provide even closer integration between storage and smart processing. These trends can be seen as representing a greater transformation of database systems away the reposed data to active platforms that may support data-oriented and smart applications. Each of these contemporary paradigms of databases depicts the branching of data management technologies in regards to changing applications needs. Different types of systems respond to particular issues associated with scalability, flexibility, consistency, and analytical performance, and it is important to define the proper database architectures depending upon the nature of workloads and the objectives of the applications. Analytical databases built on clouds build on these by providing high scalability, massively parallel processing capabilities, and elasticity that are needed to process large data volumes and iteratively compute data science workflows. These systems enable analytical queries and model training processes to be run on multiple compute nodes simultaneously, which decreases greatly the processing time. Besides, storage and compute resources are separated, which improves the flexibility and cost-efficiency of these systems because computational resources can be dynamically reallocated based on the workload requirements. Further, modern database systems are becoming more closely linked to machine learning and advanced analytics in the database engine, itself. This integration reduces the amount of data that needs to be transferred between the storage and processing layers and allows it to train in-database models with inferences. Together these improvements make it clear that the current day database systems can be not seen as just passive data storage systems but rather as active systems enabling data science applications, supporting end-

ISSN: 3071-3498
PP: 01-14.

to-end processes that include data ingestion, preprocessing, analysis, and intelligent decision-making.

2. LITERATURE SURVEY

2.1. Evolution of Database Systems

The early database systems were almost entirely hierarchical and network database systems, which were used to manage structured records having a defined relationship. These models allowed quick navigation between connected information, but were bound closely to physical storage formats and therefore were hard to edit their schema and offer flexibility. When E. F. Codd introduced the relational model, a significant shift occurred in paradigm because of thestating of data in tables and the formalization of the operations by using set theory and predicate logic. The resulting abstraction permitted data independence, made it easier to query with SQL, and enhanced consistency with the use of transactional data services. As however, with the growth of web and enterprise applications, as data volumes grew along with their user concurrency, traditional relational database management systems (RDBMSs) were provoked by the challenges of horizontal scalability, high write throughput, and distributed processing, leading to the consideration of alternative models of data management.

2.2. NoSQL Databases

The genesis of NoSQL databases was the inability of relational databases to perform large scale, distributed and unstructured or semi-structured data. These systems come in a wide variety of types and classes, each of which is better suited to particular access patterns and data models: key-value stores, document databases, column-family databases and graph databases. The critical feature of the NoSQL systems is their focus on scalability, the attained by data partitioning and replication across clusters of commodity hardware. Many NoSQL systems do not maintain high consistency as required by distributed environments, often under the concepts of the CAP theorem and embrace eventual consistency models. The design allows them to be able to process huge amounts of workload effectively but could need application-level developers to track data consistency.

2.3. NewSQL Databases

NewSQL databases are proposed as a solution that will address the deficiency between the traditional relational systems and the NoSQL platforms, offering to combine horizontal scalability with high transactional guarantees. These systems retain the relational data model and promote SQL but with newer distributed architecture so as to address performance and scalability limitations of the older RDBMSs. Low-latency and high-throughput performance are usually pursued using techniques like in-memory data storage, partitions transaction processing and optimized concurrency control mechanisms. NewSQL systems offer a solution by ensuring the presence of the ACID properties even in distributed systems to offer them to applications that demand consistency and scale including financial services, real-time analytics, and large-scale transactional loads.

ISSN: 3071-3498
PP: 01-14.

2.4. Cloud-Native Analytical Databases

Cloud database analytics Cloud-native analytic databases exist to allow analytics of large data usage to support the increased need of cloud computing platforms. Such systems are normally deployed in the form of data warehouses which have been optimized to support analytical queries by massively parallel processing (MPP) architecture. The most important innovation that cloud-native designs provided is the separation of storage and compute resources, as each can be scaled individually according to the workload requirements. This isolation facilitates elastic scaling, cost efficiency, and better fault tolerance because all data may be stored in distributed object storage and compute nodes may be brought online dynamically to execute a query. These architectures are able especially well to support complex analytical loads, to large data sets, to many users, and to sophisticated query optimization methods.

2.5. Database Support of Machine Learning

More recent studies have aimed to build the translation of machine learning features into database systems directly in order to minimize the overhead costs of the operation of data in-and-out of storage and analytical systems. The conventional workflow can also result in collecting data in databases and moving it to an outsourced machine learning service, which causes bottlenecks in the performance and complicates the model. Through integration of machine learning algorithms as part of the database engine, or through in-database analytics support, these systems support model training and inference being carried out at a closer proximity to the data. With this integration, the efficiency is enhanced, predictive analytics is enabled in real-time, and the data management and intelligent processing can be more tightly coupled. This is causing the current database systems to move out of their passive data storage formats into active parts of data-driven and intelligent applications.

3. METHODOLOGY

3.1. Research Design

In this study, the analytical research design will be based on qualitative analytical research which is based on systematic review and analysis of secondary data through peer-reviewed journal articles, conference proceedings and official technical report. The qualitative method is chosen because it allows analyzing the current theories and models and technological advancements in the sphere of database systems thoroughly in terms of concepts instead of measuring their numbers or proving them with experiments. This design makes it possible to identify the main trends, patterns, and issues concerning the development of database technologies, such as relational, NoSQL, NewSQL, and cloud-native databases upon the analysis of the existing research contributions. The selection of secondary data sources is based on the reliability, scholarly rigor, and relevance to ensure that the study will be informed by the valid and popular findings in the research community. The search of the literature is organized and conducted with the help of digital academic databases and the repositories of the well-known publishers to include the works published in the known journals and conferences. The inclusion criteria are applied through the screening of, selected studies on the

ISSN: 3071-3498
PP: 01-14.

basis of the pre-defined criteria on relevance to the aim of the research, the soundness of the methods, and its contribution to the knowledge of the understanding of database scalability, its performance, and its alignment with other emerging paradigms like machine learning. The literature obtained is then analyzed qualitatively with respect to content analysis in which the main ideas, architectural principles and performance issues are identified and examined in relation to different categories of systems. Such comparison will allow drawing various points of view into one logical system reflecting not only the changes of the database design over the years but also the present-day inventions thereof. Moreover, such research design makes it easy to critically evaluate as it permits the study to identify the strengths, limitation, and trade-offs of various database models and architectures. Instead of creating a new system implementation, the research seeks to summarize the present knowledge and give a combined insight into how contemporary database technologies solve the issues created by big data, distributed settings, and smart processing of data. By means of this qualitative method of analysis, the study guarantees theoretical richness, contextual understandability, and academic correctness, which is appropriate in exploratory and survey-based research in the field of data management systems.

3.2. Evaluation Metrics

Fig 2 - Evaluation Metrics

3.2.1. Scalability

Scalability is the capability to explicitly raise levels of data and user access without viewing a substantial drop in database management efficiency. Scalability is assessed in this study in the context of horizontal scaling in which more nodes can be added to a distributed system to scale to ever-increasing workloads. The systems which have the feature of data partitioning and replication are said to be more scaled because they are able to distribute the storage and computation among the more than one machine. Scalability is done effectively so that the performance of the database does not decline with the increase in database size and transaction rates as is a core necessity in new big data and cloud-based applications.

ISSN: 3071-3498
PP: 01-14.

3.2.2. Query Latency

Latency of query Latency is used to measure the duration that a database system needs to answer a query. This measure is vital in measuring system responsiveness especially with regard to real-time and interactive services. It is a sign that query execution and indexing and access mechanisms are more efficient due to lower query latency. Queer latency is examined in this evaluation in terms of transactional and analytical workload as well as the effect of various architectures on response times given different load requirements e.g. in-memory processing and parallel query execution in the context of query requirements.

3.2.3. Consistency

Consistency refers to the level of consistency of database system whereby all the users look at the same data at a given time, particularly in a distributed system. This measure is evaluated using how the system complies with transaction guarantees, which may include; ACID properties in relational and NewSQL systems, or eventual consistency model in many NoSQL systems. Consistency should be part of the evaluations to identify the extent to which the data operations are reliable and correct and unstable applications like financial systems and mission-critical services need precise and current information.

3.2.4. Fault Tolerance

Fault tolerance can be defined as how a system is able to operate with both hardware failures, network problems, or even node failures. This measure is considered by analysing replication measures, backup recovery measures, and automatic backups. Fault tolerance Database systems assist in reducing downtime, loss of data and as such, they enhance high availability and operational reliability. This feature is particularly significant to distributed and cloud-based databases, where failures are more probable because of intricacy of the system.

3.2.5. Suitability for Analytics

Suitability to analytics helps to determine the level of effectiveness of a database system to complex queries, large volumes of data processing, and analytical workloads. This involves capability of carrying out aggregations, joins and the ability to carry out multi-dimensional analysis effectively. Columnar-based, massively parallel processing (MPP) systems, and query planners are said to be more appropriate to analytics. The measurement is essential in determining the ability of the system to accommodate data warehousing, business intelligence, and machine learning systems that are based on high-performance analytical query execution.

3.3. Analytical Framework

In the analytical model used to represent the overall system performance, the framework attempts to model it as a dependent variable with likelihood of four essential variables, which include scalability, latency, availability, and fault tolerance. Conceptually, system performance can be defined as a functional relationship; that is; P = f(S, L, A, F) implying that these four interrelated elements mutually affect performance and not any of these factors alone. Scalability embodies the

ISSN: 3071-3498
PP: 01-14.

design of the system to provide growing massive amounts of data and user demand through utilizing more computational and storage facilities efficiently. Scalability is a key factor that can predetermine how viable a system can be in the long term because a highly scalable system is likely to sustain reasonable levels of performance with the increasing workload requirements. Latency is the amount of time the system takes to interact with and service user transactions or queries. Fast access to data and user experience are linked to lower latency, especially with applications where a quick or near-quick response is required. Availability is the extent to which one system stays accessible and functional over time despite the partial malfunctions or during maintenance processes. High availability provides a continuous service which is critical to mission-critical applications that the loss of service may cause serious losses in operations or finances. Fault tolerance, which is closely related to availability, refers to the capability of the system to survive and resume faults in the system components, e.g. a server crash or failure in the network, without the loss of any data or of the correctness. In this context, four of these variables are applied as interactive dimensions that mutually define the general performance of a system. The process of improvement in one dimension can have an impact on another, and these can be in terms of trade-offs. As an example, fault tolerance can be improved by doing universal replication, which would improve availability but can also add extra latency because of the need to synchronized. Likewise, scalability strategies, including data partitioning across distributed nodes, can influence consistency and latency in consistency on scaling by means of coordination. The framework allows the logical and systematized analysis and comparison of various database structures by modelling performance as a scalability, latency, availability and fault tolerance. It enables the study to examine the performance of different systems in terms of the achievement of these key attributes and also it evaluates their suitability to the current workloads that are characterized by large data volumes, high user concurrency and high levels of reliability demands. This analytical view implies viewing database systems in a comprehensive manner and not focusing on individual performance metrics.

3.4. Comparative Analysis Procedure

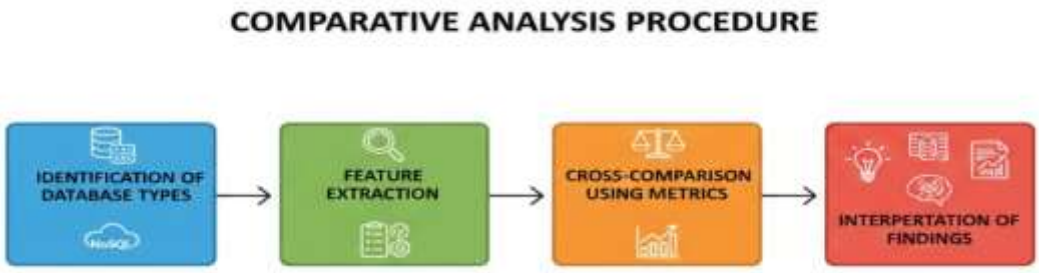

Fig 3 - Comparative Analysis Procedure

3.4.1. Identification of Database Types

The systematic identification and classification of the varied types of databases corresponding to the scope of the study form the starting point of the comparative analysis. These are traditional relational database systems, NoSQL databases, NewSQL systems, and cloud-native analytical

ISSN: 3071-3498
PP: 01-14.

database. This move defines the categories to compare and makes sure that the representative systems of all classes are taken into account. The analysis establishes a systematic base of analyzing architectural differentiation, design philosophies, and target usage cases in errors of various database technology by outlining the types of databases.

3.4.2. Feature Extraction

Key features and characteristics are then determined and made out of the literature sources that have been selected when the different types of databases are identified. The process is done by examining technical descriptions, system architecture, and performance studies to determine attributes associated with the scalability mechanisms, query processing strategy, consistency model, and fault tolerance mechanisms. The examination of the features also allows the study to capture the important design items in a uniform way, and compare across various systems. It is also a step that will assist in identifying individual innovations and general patterns common among different data bases models.

3.4.3. Cross-Comparison Using Metrics

The relevant features are extracted and the systems are compared based on predetermined evaluation metrics that are scalability, latency, availability, and fault tolerance. This cross comparison is carried out by examining the performance of both types of databases in relation to these parameters on the basis of evidence found in the literature that has been reviewed. Objectivity and coherence in the comparison process properties are guaranteed by the utilization of common metrics and allow assessing the strengths and limitations of the system within the system categories. The method assists in spotting the trade-off of various architectural decisions.

3.4.4. Interpretation of Findings

The last stage is the interpretation of the comparative findings in order to draw conclusions and findings concerning the relative performance and applicability of various types of databases. This analysis is concentrated on the interpretation of reaction to architectural means and the system behavior in different workloads and operation conditions. Through the synthesis of the comparative results, the study describes the observed performance trends and correlates them to the needs in practice. This action will help to translate the analytical findings into effective conclusions that can aid in the informed decision-making and that lead to a deeper insight into modern database system design.

4. RESULTS AND DISCUSSION

4.1. Performance Trends

The trends of performance observed among the contemporary database systems indicate the distinct differences in the manner different architecture strategies respond to the issue of scalability, consistency, and analytical efficiency. NoSQL databases are more scaled because they are designed as a distributed system with schema-flexible designs and based on the requirement of parting and replicating the data in two or more nodes. These systems can support large amounts of data and can

ISSN: 3071-3498
PP: 01-14.

service high throughput by relaxed consistency guarantees and a natural tendency to relax eventual consistency guarantees. This ensures that they are especially appropriate in applications whose datasets are growing swiftly and those with high availability demands such as web services, and real-time data accepting sites. The trade-off however with this scalability is less consistency guarantees which could create temporary data anomalies and would need further handling at the application-level in order to be correct. NewSQL databases are trying to balance out the benefits of traditional relational databases with the scalability of NoSQL platforms. They maintain orchestrated schemas and transactional guarantees without using distributed architectures and in-memory processing to enhance performance. This leads to NewSQL databases having a more balanced performance along the scalability, consistency, and latency axis. They can support high rates of transaction without impairing ACID properties, and hence may be used in application where reliability and speed are the key features needed in applications. However, such a balance is at the cost of more complicated systems. The complexity of the required coordination mechanisms, advanced concurrency management, and distributed management of transactions brings architectural and operational challenges that can increase the overhead of deployment and maintenance. Cloud-native analytical databases show especially high resource utilization in the case of an analytical load as they make use of massively parallel processing (MPP) and storage-compute resource separation. A large dataset can be mostly managed using these systems in order to execute a complex query distributed among several processing nodes. Parallel execution will achieve a substantial speed up in the time that is taken to execute queries based on aggregation-intensive and scan-intensive operations characteristic of data warehousing and business intelligence tasks. In addition, their elastic scaling is able to dynamically provision resources as workload demand changes thereby enhancing efficiency and cost-effectiveness. On the whole, these performance trends demonstrate that there is no single dominant identification of a database paradigm in all dimensions. Rather, all types of systems represent a different optimization strategy depending on the work load demands with the focus on the selection of database technologies in accordance with the application needs and performance priorities.

4.2. Implications for Data Science

The volume, speed and repetitive nature of the analytical tasks mean that data science workflows have certain requirements in terms of underlying database systems. Fast data ingestion is one of the requirements that have proven to be especially important because data science applications are often dependent on the constantly flowing data generated by sensors, transaction logs, social media feeds, and web-based applications. With the help of efficient ingestion mechanisms the raw data is captured and stored in almost real time which makes analytical models run on the latest information. Forms of databases that have distributed write, scale-to-store, and scalable schema are particularly suited here since they will accept heterogenous forms of data, and very steeply increasing data volumes without any loss of performance. The other important need is an efficient batch processing that is crucial in the preparation of large amounts of data, feature generation and model training. Data science operations are often used to scan and transform large quantities of data

ISSN: 3071-3498
PP: 01-14.

which encompass computing aggregates, normalizing and derive attributes. Specialized analytical databases with high columnar storage and parallel querying capabilities can make such operations much faster. This capability of batch processing near the data reduces the necessity of repeated data transfer between the storage systems and other analytics provided by the external tools which improve the overall efficiency of the systems and reduces the overhead costs that will be incurred. Iterative algorithm support is another important implication of data science on the design of a database. Machine learning algorithms, as well as statistics algorithms (like clustering) involve recurrent access to the same batch of data to optimize model parameters. Conventional database systems are not very efficient to do such workloads because they were optimized to support single-pass queries. In-database analytics and other systems that can support iterative calculation can work wonders on performance through both the keeping of intermediate data in memory and elimination of redundant data retrieval. These demands interact to indicate that database systems that support high-throughput data ingestion and scalable batch processing with built-in support of iterative computation are the most appropriate to the data science workflow. The applicability of a database system to data science is therefore not only limited to the capabilities of the database system in terms of storage but its capabilities in terms of maintaining data flow and repeated analytical processes effectively.

4.3. Discussion

This research indicates that hybrid database designs have better overall performance than the performance of single-model database systems when used in a heterogeneous data setting. The data used in modern day data intensive applications especially in data science and analytics has a wide range of data types, the patterns of access are different and the requirements when performing the processing are also varied. Single-model systems, including purely relational or purely NoSQL, are often only optimized to a few workloads and hence do not have the ability to cover the wide range of demands imposed by modern platforms. By contrast, hybrid architectures combine two or more database paradigms like relational database, document-based database and analytical engines, into a single ecosystem. With this integration, each component is then specialized on the workload it can best perform, which increases the efficiency, scalability, and responsiveness of the workload throughout the system. This benefit is further supported by the principle of polyglot persistence that proposes application of various data storage technologies to meet various application requirements instead of using the one, universal solution. As an illustration, relational or NewSQL databases can be used to provide high consistency and reliability on transactional data, and a NoSQL database can be used to store large amounts of semi-structured or unstructured data and permit a flexible schema. Simultaneously, analytics with queries and model training can be sent to cloud-native data warehouses that are scheduled to perform massively parallel processing. This separation of duties lessens the presence of bottlenecks in the systems and enables each database technology to work within what best it can perform in. Such architectural diversity is especially useful in data science platforms that are large, designed to process, prepare, train models, and perform real-time inferences simultaneously. Hybrid systems facilitate smooth circulation of data among device components

ISSN: 3071-3498
PP: 01-14.

which are adaptive of both operational and analytical workload without adverse effects on performance. In addition, the practice will improve the resilience of the system since it will not cement the whole platform within the confines of a limited database model. In turn, polyglot persistence should be advised as a strategic design concept of complex data science structures. With the matching of the particular database technologies with the workload need, the organizations will be able to have more flexibility, enhanced use of resources, and more sustainable performance when data environments are heterogeneous and dynamically changing.

5. CONCLUSION

In this paper, the current database systems have been offered in the framework of applications of data science and based on their main principles of architecture, the features of operation, and the support of analytical loads. Under this analysis, it is clearly seen that the growth of the database technologies has been necessitated more by the growing magnitude, heterogeneity and complexity of information produced in the modern day computing set-up. Conventional relational data base management systems, as effective as they are with organized transaction processing, have fundamental constraints when used in large-scale analysis loads involving high throughput, distributed storage and parallel processing. They depend on centralized architectures and strict schema when they need to be high-performance and flexible in situations of large data streams and heterogeneous data sources. Conversely, NoSQL and NewSQL systems have also emerged as successful solutions to these issues that support most of them with scalable and flexible design models. NoSQL databases, which have schema-less or semi-structured data models and distributed architectures, are highly available and horizontally scalable, which makes them very effective when dealing with large and fast growing data. These gains are however attained at the expense of the consistency guarantees being typically loosened, which is not necessarily agreeable to all areas of application. NewSQL systems endeavour to address this trade-off preserving the relational model and transactional consistency whilst using distributed processing and in-memory techniques to enhance performance. Consequently, they provide a middle way that facilitates scalability as well as reliability, especially in workloads that involve a high level of data integrity, as well as high rates of transactions. Moreover, there are new benefits of the cloud-native database systems that have decoupled storage and computation and capitalized on the benefits of massively parallel computing models. Their capabilities permit the dynamic level of resource allocation to support scaling the systems based on the workload needs (elasticity), and simplify running complex analytical queries on the large data sets. These are particularly beneficial to data science applications that require large-scale batch processing, exploratory analysis and training models iteratively. Combination of these systems and cloud infrastructure also improves fault tolerance and cost-efficiency, making it even more attractive where an analytical environment of large scale is required. On the whole, the results suggest the relevance of a careful evaluation of the workload features, consistency, and scalability needs to be used to select a database successfully in data science applications instead of the efforts to follow a single database paradigm. There exists no one system that is best suited to handle all the potential uses, and this is why it is significant to utilize flexible data management and hybrid

ISSN: 3071-3498
PP: 01-14.

approaches. The way forward in future studies must instead focus on enhancing database-native machine learning functions, handling the automated optimization of systems, and improving the mechanisms of data governance in the distributed and cloud-based worlds. Such activities will be pivotal in the continued development of the next generation of smart, data-enabled applications as well as in the assurance that the database systems move forward toward the new analytical and computing requirements.

REFERENCES

[1] Codd, E. F. (1970). *A relational model of data for large shared data banks*. Communications of the ACM, 13(6), 377–387.
[2] Date, C. J. (2004). *An Introduction to Database Systems* (8th ed.). Pearson Education.
[3] Stonebraker, M., & Hellerstein, J. M. (2005). *What goes around comes around*. In Readings in Database Systems (4th ed.). MIT Press.
[4] Brewer, E. A. (2000). *Towards robust distributed systems*. Proceedings of the 19th Annual ACM Symposium on Principles of Distributed Computing (PODC).
[5] Dean, J., & Ghemawat, S. (2008). *MapReduce: Simplified data processing on large clusters*. Communications of the ACM, 51(1), 107–113.
[6] Chang, F., et al. (2008). *Bigtable: A distributed storage system for structured data*. ACM Transactions on Computer Systems, 26(2), 1–26.
[7] Amazon Web Services. (2007). *Dynamo: Amazon's highly available key-value store*. Proceedings of SOSP.
[8] Sadalage, P. J., & Fowler, M. (2013). *NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence*. Addison-Wesley.
[9] Pokorny, J. (2013). *NoSQL databases: A step to database scalability in web environment*. International Journal of Web Information Systems, 9(1), 69–82.
[10] Stonebraker, M., et al. (2010). *The end of an architectural era (It's time for a complete rewrite)*. Proceedings of VLDB, 2(2), 1150–1160.
[11] Pavlo, A., et al. (2017). *What's really new with NewSQL?* ACM SIGMOD Record, 45(2), 45–55.
[12] Abadi, D. J., et al. (2013). *The design and implementation of modern column-oriented database systems*. Foundations and Trends in Databases, 5(3), 197–280.
[13] Melnik, S., et al. (2010). *Dremel: Interactive analysis of web-scale datasets*. Proceedings of VLDB, 3(1), 330–339.
[14] Kraska, T., et al. (2018). *The case for learned index structures*. Proceedings of SIGMOD, 489–504.