iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: https://doi.org/10.22214/ijraset.2026.81466

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

A Review on Serverless Query Processing: Flexible SLAs, Pricing Models, and Performance Trade-offs

Nikita Datke[1], Vinita Shrivastava[2]
[1]*Computer Science and Engineering, Rajiv Gandhi Proudyogiki Vishwavidyalaya Bhopal*
[2]*Information Technology, Oriental Institute of Science and Technology, Bhopal, India*

Abstract: ***Serverless computing has rapidly evolved into a key paradigm in modern cloud platforms, eliminating infrastructure management while enabling elastic scalability and fine-grained billing. Query processing within serverless environments introduces both opportunities and challenges. On the one hand, serverless query engines such as Amazon Athena, Google BigQuery, Snowflake, and Azure Synapse provide simplified management and cost efficiency. On the other hand, ensuring predictable performance under diverse workloads while maintaining cost transparency remains a critical issue. This review paper surveys the state of the art in serverless query processing, with an emphasis on flexible Service Level Agreements (SLAs), pricing models, and performance trade-offs. Research prototypes such as PixelsDB, Cloudburst, and FunDa are discussed alongside commercial systems. Comparative analysis highlights the strengths and limitations of existing approaches. Open challenges such as SLA enforcement, cold-start overheads, and standardized benchmarking are identified. Finally, future research directions are outlined, including AI-assisted SLA negotiation, self-optimizing query execution, and hybrid serverless–edge analytics.***

Keywords: ***Serverless computing, cloud data analytics, serverless query processing, Service Level Agreements (SLAs), flexible pricing models, cost–performance trade-offs, serverless query engines, elastic scalability, pay-as-you-go model, AI-assisted SLA negotiation, cold-start latency, performance optimization, hybrid serverless–edge analytics, benchmarking in serverless systems.***

I. INTRODUCTION

The increasing demand for scalable and cost-efficient data processing has fueled the adoption of serverless computing in modern cloud environments. Unlike traditional server-based models, serverless platforms eliminate the need for infrastructure management, enabling developers to focus entirely on application logic while the cloud provider handles provisioning, scaling, and maintenance. In recent years, this paradigm has gained significant attention due to its pay-as-you-go pricing model and its ability to dynamically scale based on workload demands [1], [2].

Among the many areas where serverless computing is being explored, serverless query processing has emerged as a critical research domain. With the exponential growth of structured and unstructured data, organizations require systems capable of handling analytical queries at scale without incurring prohibitive costs. However, traditional query engines, though efficient, rely on static resource allocation and long-running clusters, which often result in underutilization and higher operational expenses [3], [4]. Serverless query engines attempt to overcome these limitations by decoupling storage from compute and offering on-demand execution, leading to significant cost savings while maintaining performance guarantees.

A defining challenge in this context is the management of Service Level Agreements (SLAs). Cloud customers have diverse performance expectations depending on application criticality—some may prioritize low latency, while others may be more cost-sensitive and tolerate slightly higher delays. This variability calls for flexible SLA frameworks that can balance performance and cost through adaptive pricing models [5], [6]. Recent advances, such as PixelsDB, demonstrate that a carefully designed serverless query engine can achieve substantial cost reductions—up to 65.5% in benchmark workloads—without compromising on query latency [7].

Despite these advancements, the adoption of serverless query processing is not without challenges. Issues such as cold-start latency, resource heterogeneity, and observability limitations continue to hinder widespread deployment [8], [9]. Furthermore, designing pricing models that are both profitable for providers and economical for customers remains an open research area. Current literature highlights several trade-offs between query performance, workload variability, and pricing granularity, making it essential to review existing approaches and identify gaps for future innovation [10], [11].

©IJRASET: All Rights are Reserved

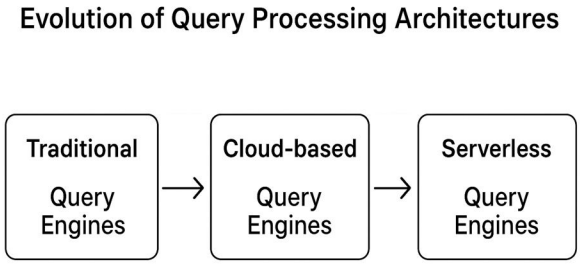
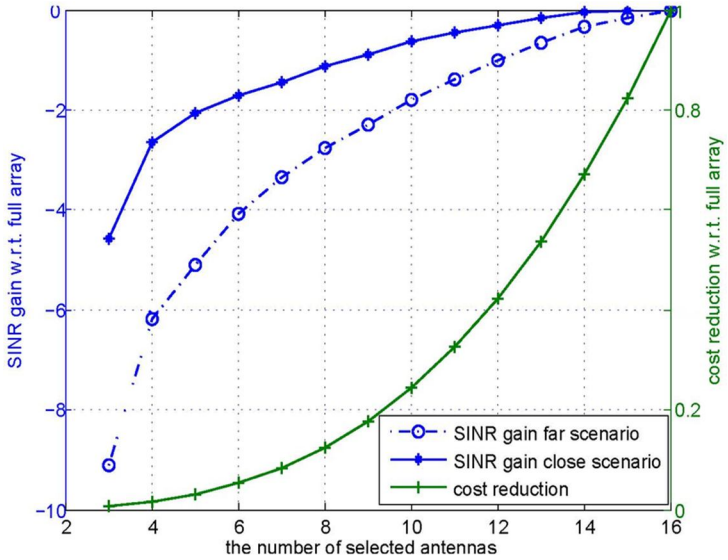


Figure 1: Evolution of Query Processing Architectures

This review paper aims to provide a comprehensive analysis of serverless query processing with a focus on flexible SLAs, adaptive pricing models, and performance trade-offs. We explore state-of-the-art systems, evaluate their architectural decisions, and compare their cost–performance efficiency. The contributions of this work are summarized as follows:

1) Review of State-of-the-Art Systems – We analyze existing serverless query engines, highlighting their design choices and SLA strategies.
2) Comparison of Pricing Models – We investigate the various pricing mechanisms employed in serverless architectures and their implications for cost efficiency.
3) Performance vs. Cost Trade-offs – We study how different systems balance performance guarantees with financial considerations, presenting comparative insights.



Although the image shows a generic cost-performance trade-off, it's not specific to SLA strictness in serverless query processing. Creating a custom graph tailored to your concept (with SLA strictness on the X-axis and separate Cost / Latency curves) would be more clear and contextual[16].

4) Challenges and Future Research Directions – We identify open research problems, such as hybrid multi-cloud deployment, standardized benchmarking, and AI-driven resource orchestration.

By synthesizing these findings, this paper contributes to the ongoing discourse on serverless computing in data analytics and serves as a foundation for researchers and practitioners aiming to build more efficient and adaptable serverless query engines.

Table 1: Comparison Between Traditional vs. Serverless Query Engines

Feature	Traditional Query Engines	Serverless Query Engines
Resource Management	Dedicated clusters	On-demand allocation
Cost Model	Fixed/hourly	Pay-per-use
Scalability	Limited by cluster size	Virtually unlimited
SLA Flexibility	Static	Adaptive

II. BACKGROUND AND MOTIVATION

A. Evolution from Traditional DBMS to Serverless Query Engines

Traditional database management systems (DBMSs) were designed in an era when workload predictability and dedicated infrastructure were the norm. These systems require explicit resource provisioning, meaning administrators must carefully estimate capacity in advance to avoid performance degradation or downtime. This rigid approach often leads to underutilization of resources during low demand and performance bottlenecks during peak usage.

With the rise of cloud computing, platforms such as Amazon RDS and Microsoft Azure SQL Database introduced managed services that simplified deployment and maintenance. However, these cloud-native databases still rely on fixed-capacity virtual machine instances, and users are charged regardless of the degree of utilization [3]. While they reduce administrative overhead, they remain inefficient in handling bursty and unpredictable workloads.

The emergence of serverless query engines marks a significant shift in architectural design. Systems such as AWS Athena, Google BigQuery, and research prototypes like PixelsDB abstract away infrastructure concerns entirely. Instead of allocating resources upfront, they dynamically assign compute and memory on a per-query basis, offering elastic scaling, automatic fault tolerance, and pay-per-use billing [7]. This transition represents a paradigm shift in database systems, aligning data processing more closely with the principles of cloud-native elasticity and cost efficiency.

B. Importance of SLAs and Pricing Models

In modern data-driven organizations, Service Level Agreements (SLAs) play a pivotal role in ensuring that cloud services meet application requirements. An SLA specifies guarantees on performance metrics such as query latency, throughput, fault tolerance, or availability, which are critical for enterprise workloads ranging from real-time analytics to large-scale batch processing [2]. For example, financial applications may demand millisecond-level response times, while data warehousing tasks can tolerate higher latencies but prioritize cost savings. Thus, flexible SLA frameworks are essential to accommodate this spectrum of requirements.

Alongside SLAs, pricing models are a determining factor for the adoption of serverless query engines. Although the pay-per-query model offers attractive cost savings compared to provisioned clusters, the unpredictability of query complexity and workload patterns can result in volatile costs. Enterprises may hesitate to adopt such systems if the billing structure lacks transparency or predictability, regardless of their technical benefits [11]. To address this, researchers and cloud providers are exploring tiered service levels, cost–latency trade-off models, and adaptive pricing strategies that align expenditure more closely with workload behavior [12].

Ultimately, the interplay between SLAs and pricing defines the real-world viability of serverless query engines. Without robust SLA enforcement, customers cannot rely on consistent performance; without transparent and flexible pricing, organizations may not realize the promised cost efficiency. Therefore, advancements in this domain are not only technical but also economic and strategic, directly influencing cloud adoption trends.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

III. EXISTING SERVERLESS QUERY ENGINES AND FRAMEWORKS

Serverless query engines represent a fundamental shift from traditional database systems by eliminating the need for provisioning, configuring, and managing infrastructure.

Instead, resources are allocated on demand, and users are billed only for actual usage. These systems can broadly be categorized into commercial platforms (cloud-provider managed services) and research-driven prototypes (academic or open-source projects focusing on optimization and innovation).

While commercial offerings emphasize ease of use, elasticity, and integration with existing ecosystems, research prototypes often push the boundaries of SLA guarantees, stateful processing, and cost-aware optimizations. This section provides a detailed overview and comparative analysis of both categories.

A. Commercial Platforms

1) Amazon Athena: Amazon Athena allows users to run SQL queries directly on datasets stored in Amazon S3 without requiring ETL processes or provisioning clusters. It follows a pay-per-query pricing model (based on the amount of data scanned) which makes it highly attractive for sporadic and exploratory workloads. However, Athena's SLA guarantees are minimal, making it unsuitable for mission-critical latency-sensitive applications.

2) Google BigQuery: Google BigQuery offers a unique combination of on-demand pricing (per TB processed) and flat-rate pricing for reserved slots, enabling both small-scale users and enterprise clients to adopt the service flexibly. It integrates with BigQuery ML for machine learning and supports federated queries across external sources such as Google Drive and Cloud SQL. Although highly scalable, BigQuery's performance may vary depending on data locality and workload concurrency.

3) Snowflake: Snowflake provides a cloud-agnostic data warehouse with consumption-based pricing. Its architecture separates storage and compute, allowing independent scaling of both resources. This separation reduces costs for workloads that involve heavy storage with intermittent compute needs. Snowflake also supports data sharing across organizations, making it particularly useful for collaborative analytics. Despite its strengths, SLA guarantees remain general rather than query-specific.

4) Azure Synapse Serverless: Azure Synapse Analytics offers a serverless SQL pool that enables users to query structured and semi-structured data stored in Azure Data Lake using T-SQL. It follows a pay-per-terabyte processed model, making it cost-effective for infrequent analytics queries. However, the performance of Synapse Serverless is highly dependent on data partitioning and file formats, limiting its consistency in SLA compliance.

B. Research Prototypes

While commercial platforms prioritize reliability and integration, research prototypes focus on addressing open challenges such as SLA enforcement, elasticity for mixed workloads, and cost-aware query processing.

1) PixelsDB: PixelsDB is designed with SLA awareness as its primary goal. By classifying queries into multiple SLA tiers (e.g., low latency, balanced, cost-optimized), PixelsDB achieves up to 65% cost reduction without violating SLA requirements. Its multi-tiered design represents a significant departure from traditional one-size-fits-all serverless query engines.

2) Cloudburst: Cloudburst extends the serverless model to stateful analytics by allowing developers to manage state efficiently across functions. Unlike traditional stateless FaaS models, Cloudburst leverages a low-latency data store to maintain state, enabling real-time analytics applications. This makes it particularly suitable for workloads such as streaming queries, fraud detection, and personalization systems.

3) Faasm (Faaslets): Faasm introduces Faaslets, lightweight containers that allow fast context switching between serverless functions. This design drastically reduces overhead compared to traditional VMs or containers, thereby improving performance for stateful or iterative queries. By supporting efficient state management and isolation, Faasm provides a foundation for future serverless query engines that must balance security with performance.

4) FunDa: FunDa focuses on in situ query processing, where queries are executed directly on distributed storage nodes, thereby reducing data movement costs. This architecture enhances scalability and reduces latency, particularly for big data analytics workloads. Although promising, FunDa faces challenges in integrating SLA guarantees, as performance is tightly coupled with data distribution and underlying storage layers.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

Table 1: Comparison of Commercial Serverless Query Engines

Platform	Pricing Model	SLA Flexibility	Suitable Workloads	Key Features
Amazon Athena	Pay-per-query	Low	Ad-hoc analytics	Simple integration with S3
Google BigQuery	On-demand + Flat-rate	Medium	Large-scale analytics, ML	Built-in ML, federated queries
Snowflake	Consumption-based	Medium	BI, ETL, collaborative data	Storage–compute separation
Azure Synapse Serverless	Pay-per-TB processed	Low	Sporadic queries	Tight Azure ecosystem integration

Table 2: Comparison of Research Prototypes

Prototype	Focus Area	SLA Awareness	Performance Optimization	Distinctive Contribution
PixelsDB	SLA-aware query execution	High	65% cost reduction with guarantees	Flexible multi-level SLA definitions
Cloudburst	Stateful serverless queries	Medium	Low-latency through state mgmt.	First to support stateful functions
Faasm	Lightweight isolation	Medium	Fast context switching	Introduces Faaslets for efficiency
FunDa	In situ query processing	Medium	Reduced data movement	Scalable serverless analytics engine

IV. FLEXIBLE SLAS IN SERVERLESS QUERY PROCESSING

SLAs in serverless environments can be categorized into:

1) Latency-focused SLAs (e.g., sub-second query response times)
2) Throughput-oriented SLAs (sustaining queries per second).
3) Reliability SLAs (guaranteed uptime).

Approaches to SLA enforcement:

- Resource over-provisioning: Guarantees performance but increases costs [7].
- Adaptive scheduling: Balances workload demands dynamically [3].
- SLA-tiered service levels: PixelsDB introduces multi-level SLA classes, allowing cost vs. performance trade-offs [1], [2].

V. PRICING MODELS AND PERFORMANCE TRADE-OFFS

Serverless pricing is tightly linked to cost-performance efficiency:

1) Pay-as-you-go: Charges per query or data scanned (Athena, BigQuery on-demand) [11], [12].
2) Reserved capacity: Users pay a fixed price for predictable workloads (BigQuery flat-rate) [12].
3) Consumption-based pricing: Snowflake charges based on compute credits consumed [13].
4) Hybrid SLA pricing: Research systems like PixelsDB introduce flexible SLA-based cost tiers [1].

Case Study: PixelsDB achieves 65.5% cost reduction in benchmark workloads without latency penalties [1], [2].

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

VI. COMPARATIVE ANALYSIS

Table 1: Comparison of Serverless Query Systems

Platform / System	SLA Flexibility	Pricing Model	Performance Guarantees	Cost Efficiency
AWS Athena [11]	Limited	Pay-per-query	Moderate	High for small queries
Google BigQuery [12]	Medium	On-demand / Flat-rate	High with reserved slots	Costly for bursty workloads
Snowflake [13]	High	Consumption-based	Strong workload isolation	Balanced
Azure Synapse [14]	Limited	Pay-per-TB processed	Moderate	Good for ad-hoc queries
PixelsDB [1], [2]	High	SLA-tiered	Predictable latency	~65% cost savings
Cloudburst [4]	High	Research prototype	Low-latency stateful execution	TBD
FunDa [8]	Medium	Prototype model	In situ query support	Scalable but untested commercially

VII. CHALLENGES AND OPEN RESEARCH GAPS

Despite the rapid progress of serverless query processing, several open research gaps continue to limit its broader adoption and efficiency.

• SLA Enforcement in Multi-Tenant Environments [3], [7]

Guaranteeing strict Service Level Agreements (SLAs) in highly dynamic, multi-tenant environments remains a major challenge. When multiple users issue queries simultaneously, resource contention leads to latency spikes and SLA violations. Current solutions provide coarse-grained resource allocation, but finer-grained isolation and adaptive scheduling strategies are still underexplored. Future work must investigate how to balance fairness with efficiency, especially in data-intensive analytical workloads.

• Cold-Start Mitigation and Performance Isolation [5]

The cold-start problem, where functions experience high latency during their first invocation, significantly impacts interactive queries. While container pre-warming techniques and lightweight sandboxing reduce start-up delays, they introduce additional cost overheads. Moreover, ensuring strong performance isolation between concurrent queries without over-provisioning resources is difficult. Novel runtime designs that minimize cold-start delays while maintaining predictable query performance are urgently needed.

• Transparent Pricing Across Heterogeneous Workloads [11], [12]

Current serverless platforms adopt simplified billing models such as per-query or per-processed-byte pricing. However, these models often fail to capture workload heterogeneity, especially when queries vary in complexity, statefulness, and resource utilization. Users struggle to predict costs for long-running or hybrid analytical tasks. Research is needed to design pricing models that correlate with both workload intensity and SLA requirements, thereby enabling users to make informed cost-performance trade-offs.

• Standardized Benchmarks for Serverless Query Systems [7]

The lack of standardized and widely accepted benchmarks for serverless query engines hinders objective performance comparison. Existing benchmarks are often tailored to traditional distributed databases and do not reflect serverless-specific challenges such as elasticity, function orchestration overhead, and state management. There is a pressing need for community-driven benchmark suites that capture real-world query workloads, variable data scales, and SLA-driven metrics.

• Portability Across Multi-Cloud and Hybrid Environments [6]

Vendor lock-in remains a major barrier as serverless platforms often rely on proprietary APIs and runtimes. Deploying the same query engine seamlessly across AWS, Azure, GCP, or hybrid on-premise environments is extremely challenging. Research must explore abstraction layers, containerized function runtimes, and federated orchestration frameworks that ensure portability while maintaining SLA guarantees.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

VIII. FUTURE DIRECTIONS

1) AI-assisted SLA negotiation: Machine learning models predicting workload patterns for SLA-aware scheduling [6].
2) Self-optimizing query engines: Reinforcement learning for dynamic resource allocation [3].
3) Blockchain for SLA enforcement: Smart contracts ensuring trust in SLA compliance.
4) Hybrid edge–serverless query execution: Reducing latency for real-time analytics [8].
5) Standard benchmarks analogous to TPC-H tailored for serverless query systems [7].

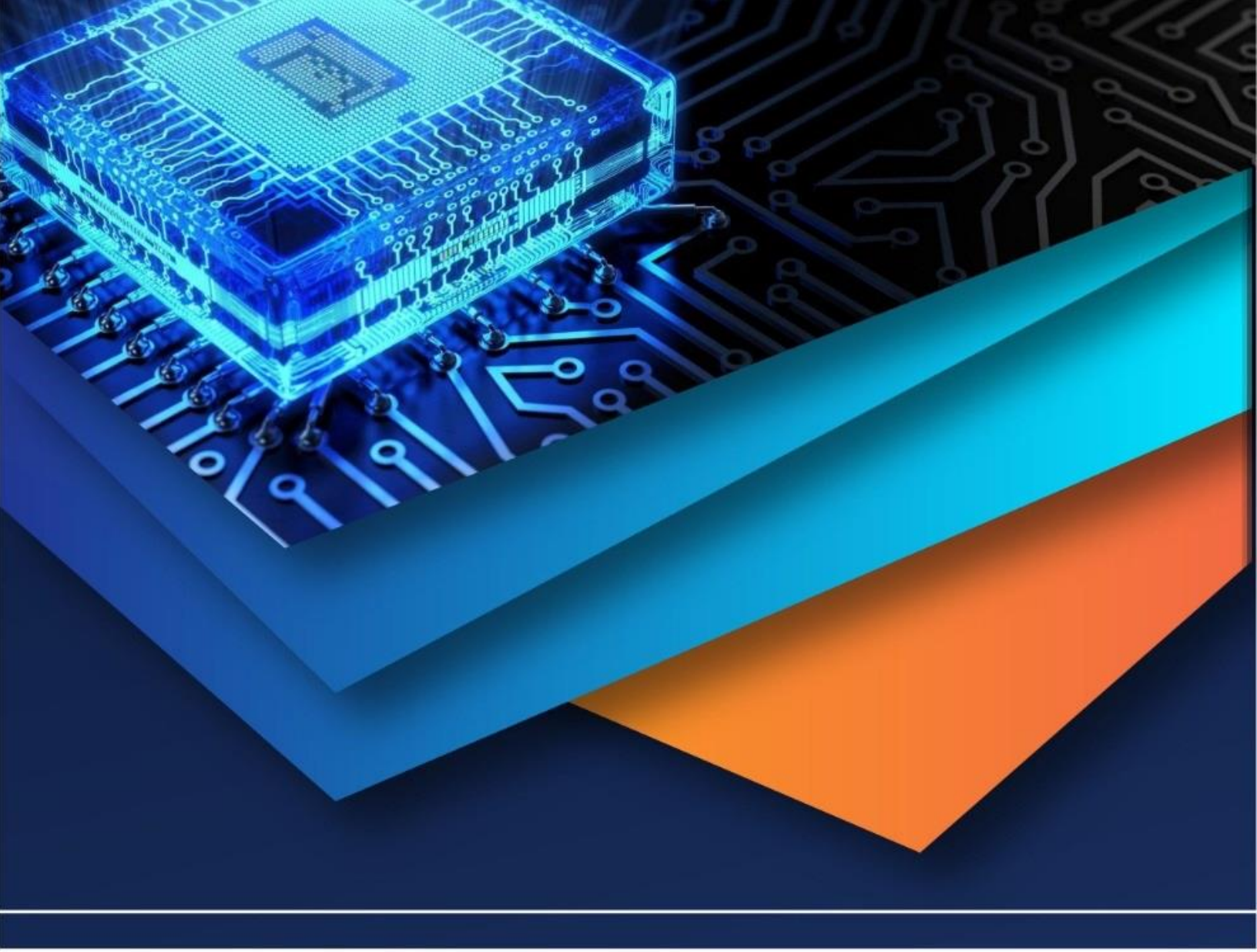
IX. CONCLUSION

Serverless query processing offers a compelling balance between scalability, cost efficiency, and reduced management overhead. While commercial systems provide flexible pricing models, research prototypes like PixelsDB and Cloudburst demonstrate that SLA-aware designs can achieve significant cost-performance improvements. However, challenges remain in SLA enforcement, pricing transparency, and benchmarking. Future innovations in AI-assisted scheduling, hybrid architectures, and blockchain-driven SLA enforcement hold promise for advancing this paradigm further.

REFERENCES

[1] H. Bian, D. Geng, H. Li, and Y. Chai, “PixelsDB: Serverless and natural-language-aided data analytics with flexible service levels and prices,” arXiv preprint arXiv:2402.14572, 2024.
[2] H. Bian, D. Geng, P. Guo, and Y. Chai, “Serverless query processing with flexible performance SLAs and prices,” arXiv preprint arXiv:2402.15012, 2024.
[3] A. Klimovic, D. S. Berger, S. Zeuch, C. Kozyrakis, and P. Pietzuch, “Ephemeral per-query engines for serverless analytics,” Proc. 49th Int. Conf. on Very Large Data Bases (VLDB), pp. 2435–2448, 2023.
[4] V. Sreekanti, C. Wu, X. Zhang, J. Gonzalez, J. Hellerstein, and J. M. Faleiro, “Cloudburst: Stateful functions-as-a-service,” Proc. VLDB Endowment, vol. 13, no. 12, pp. 2438–2452, 2020.
[5] J. Shillaker and P. Pietzuch, “Faasm: Lightweight isolation for efficient stateful serverless computing,” in Proc. USENIX Annual Technical Conf. (ATC), 2020, pp. 419–433.
[6] D. Saha, R. Wagh, and A. Sharma, “A survey on serverless computing and its workflow management,” ACM Comput. Surveys, vol. 55, no. 12, pp. 1–36, 2023.
[7] R. Mayer, A. Nastic, and S. Dustdar, “Trade-offs and challenges of serverless data analytics,” in Serverless Computing for Data Analytics, Springer, 2021, pp. 1–24.
[8] Z. Zhang, K. Liu, and H. Xu, “FunDa: scalable serverless data analytics and in situ query processing,” Journal of Big Data, vol. 12, no. 45, pp. 1–21, 2025.
[9] E. Jonas, Q. Pu, S. Venkataraman, I. Stoica, and B. Recht, “Cloud programming simplified: A Berkeley view on serverless computing,” EECS Department, University of California, Berkeley, Technical Report No. UCB/EECS-2019-3, Feb. 2019.
[10] E. S. de Almeida, G. F. Coutinho, and A. R. Mury, “Crucial: Stateful serverless computing,” in Proc. ACM Symp. Cloud Computing (SoCC), pp. 115–128, 2020.
[11] Amazon Web Services, “Amazon Athena Pricing,” AWS Documentation, 2025. [Online]. Available: https://aws.amazon.com/athena/pricing
[12] Google Cloud, “BigQuery Pricing,” Google Cloud Documentation, 2025. [Online]. Available: https://cloud.google.com/bigquery/pricing
[13] Snowflake Inc., “Snowflake Pricing: Consumption-based model,” Snowflake Docs, 2025. [Online]. Available: https://www.snowflake.com/pricing
[14] Microsoft Azure, “Azure Synapse Analytics Serverless SQL Pool Pricing,” Azure Docs, 2025. [Online]. Available: https://azure.microsoft.com/pricing
[15] E. Jonas, J. Schleier-Smith, V. Sreekanti, et al., “Cloud programming simplified: A Berkeley view on serverless computing,” Commun. ACM, vol. 63, no. 12, pp. 54–62, 2020.
[16] S. N. Karam, A. F. M. Ayad, and A. H. Mutlag, “The trade-off curve between the performance and the computational cost,” Scientific Figure on ResearchGate, 2020. [Online]. Available: https://www.researchgate.net/figure/The-trade-off-curve-between-the-performance-and-the-computational-cost_fig3_341646635

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)