ISSN: 2583-9888
DOI: doi.org/10.51219/JAIMLD/sethu-sesha-synam neeli/467

URF PUBLISHERS
connect with research world

Journal of Artificial Intelligence, Machine Learning and Data Science
https://urfpublishers.com/journal/artificial-intelligence

Vol: 1 & Iss: 4 *Research Article*

Amazon Redshift Performance Tuning for Large-Scale Analytics

Sethu Sesha Synam Neeli*

Citation: Neeli SSS. Amazon Redshift Performance Tuning for Large-Scale Analytics. *J Artif Intell Mach Learn & Data Sci 2023*, 1(4), 2130-2134. DOI: doi.org/10.51219/JAIMLD/sethu-sesha-synam-neeli/467

Received: 03 December, 2023; **Accepted:** 28 December, 2023; **Published:** 30 December, 2023

***Corresponding author:** Sethu Sesha Synam Neeli, Sr. Database Administrator, USA, E-mail: sethussneeli@gmail.com

A B S T R A C T

Online Analytical Processing (OLAP) is crucial for modern organizations, enabling data-driven decision-making from ever-growing datasets. The increasing volume and complexity of this data present significant challenges to timely and efficient analysis. This paper investigates Amazon Redshift, a highly scalable and performant data warehouse service offered by AWS, as a solution to these challenges. We will examine Redshift's architecture, focusing on its parallel processing capabilities, columnar storage and optimization techniques. Furthermore, we will explore best practices for scaling Redshift deployments to handle heavy analytical workloads and discuss the impact of key design choices, such as data distribution strategies and ETL processes, on overall performance. The research will contribute to a better understanding of how Redshift can effectively address the analytical needs of large organizations.

Keywords: Redshift, ETL, Parallel Processing, data, Columnar, Structures, algorithm, distribution keys

1. Introduction

Amazon Redshift is a fully managed, cloud-based data warehouse service optimized for high-performance analytical processing of massive datasets. Leveraging a massively parallel processing (MPP) architecture, Redshift achieves sub-second query response times exceeding the performance of many alternative data warehouse solutions. A Redshift cluster comprises one or more compute nodes that execute query processing in parallel. Clusters with two or more compute nodes incorporate a dedicated leader node responsible for coordinating inter-node communication, query scheduling and external application interactions. The client application interface is exclusively managed through the leader node, abstracting the underlying compute node infrastructure **(Figure 1)**.

This distributed architecture enables efficient data manipulation and parallel query execution for large-scale analytical workloads.

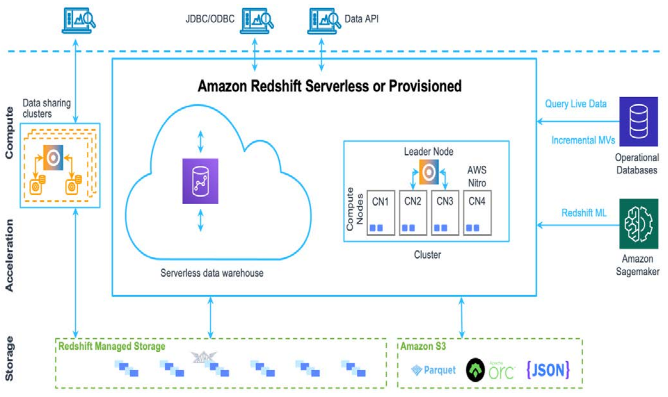


Figure 1: Data warehouse system architecture.

The proliferation of data-driven decision-making necessitates robust and scalable data processing solutions. Amazon Redshift, a cloud-based data warehouse service, provides a cost-effective platform for managing large-scale analytical workloads; however, realizing its full potential requires a sophisticated understanding of performance optimization strategies. This paper examines the challenges and solutions inherent in scaling data processing within Redshift, emphasizing the crucial role of database administrators (DBAs) in maintaining optimal performance under high-throughput conditions. We analyze key aspects of Redshift performance, including query optimization techniques, data modeling best practices (schema design, data partitioning strategies), cluster configuration (compute node allocation, storage optimization) and performance monitoring methodologies. The paper delivers actionable recommendations for DBAs to effectively manage and scale their Redshift environments, focusing on efficient data manipulation and software development best practices for maximizing the performance of this distributed database system.

2. Proof of Concept

A Redshift proof-of-concept (POC) prioritizes aligning technical architecture with business requirements. This involves defining the scope of data to be processed, implementing appropriate security measures (including data encryption) and optimizing database design. Critical design decisions include selecting suitable data distribution styles (e.g., key distribution, even distribution) to balance workload across compute nodes, choosing efficient compression encoding methods, defining sort keys to enhance query performance and establishing appropriate table constraints to maintain data integrity. Rigorous testing using production-representative datasets is crucial to validate performance and scalability before full deployment. This process involves thorough evaluation of system architecture, data manipulation strategies and software components to ensure optimal performance and security **(Figure 2)**.

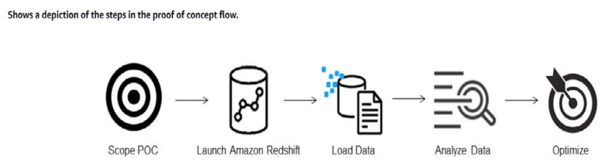

Shows a depiction of the steps in the proof of concept flow.
Scope POC
Launch Amazon Redshift
Load Data
Analyze Data
Optimize

Figure 2: Conduct a proof of concept (POC) for Amazon Redshift.

3. Optimized Methodology for AWS Redshift Cluster Performance

Achieving effective performance from an Amazon Redshift cluster requires careful consideration of several key factors:

3.1. Optimal distribution key selection

Establishing the appropriate distribution strategy is crucial for efficient data loading.

- **Auto:** AWS Redshift automatically determines an optimal distribution style based on table size; smaller tables will default to ALL distribution, while larger tables may transition to KEY distribution. If a suitable column for distribution cannot be identified, EVEN distribution will be employed.
- **EVEN:** This method distributes data across slices in a uniform, round-robin manner, making it suitable for tables that are not involved in joins.
- **KEY:** This strategy distributes rows based on the values of a specified column, ensuring that matching values are stored within the same node slices.
- **ALL:** This allocation distributes all rows of a table to every node, facilitating efficient joins by ensuring that complete tables are present on each node.

3.2. Choosing the optimal instance class

Selecting the appropriate cluster class is a critical decision influenced by data distribution and cost considerations, with two main options available:

- **DC2:** These nodes support compute-intensive data warehouses with local SSD storage, allowing users to specify the number of nodes based on their data size and performance needs. As the data volume increases, more compute nodes can be added to enhance storage capacity.
- **RA3:** RA3 nodes facilitate the storage and processing of substantial data volumes, enabling users to determine the number of nodes required for optimal data processing speed. This option operates on a pay-for-what-you-use model, providing cost efficiency. When selecting RA3 nodes, it is essential to assess daily data processing volumes, workload nature, memory requirements for complex queries and anticipated storage needs.

3.3. Sort key strategies

Unlike traditional relational database management systems (RDBMS) that utilize clustered and non-clustered indexes to optimize query execution, Amazon Redshift, being a columnar database, employs a sort key mechanism to enhance query performance.

- Properly configuring the sort key assists the Redshift query engine in efficiently locating necessary rows and minimizing data scanning.
- For queries focused on recent data, utilizing a timestamp column as the primary sort key can optimize the handling of time-sensitive queries.
- Maintaining a default distribution strategy with AUTO can also be beneficial when workload patterns are ambiguous.

3.4. Concurrency scaling

Leveraging concurrency scaling can help optimize resource allocation and query performance. By running the main cluster continuously over a 24-hour period, users can accumulate credits that enhance concurrency performance. This feature intelligently distributes memory resources across clusters, ensuring that critical workloads maintain adequate resources regardless of fluctuations in data warehouse load.

3.4.1. Short query acceleration (SQA): Short Query Acceleration is a feature designed to prioritize quick, simple queries over larger, more complex ones. This capability allows for expedited responses to straightforward queries without being impeded by the completion time of longer processes, which is particularly advantageous when managing large datasets.

3.4.2. Change data capture (CDC): Implementing Change Data Capture enables the efficient tracking and application of modifications from data sources to the data warehouse. By utilizing CDC, performance is optimized as only altered data is processed, thus eliminating the need for exhaustive full data loads. For CDC implementation, consider utilizing tools like AWS Glue or third-party solutions that can capture changes from diverse data sources.

3.5. Implementing change data capture (CDC)

To begin reaping the benefits of Change Data Capture (CDC), follow these steps:

- **Identify changes:** Detect new, modified or deleted rows within your data source.
- **Capture changes:** Implement a CDC mechanism to efficiently log these alterations.
- **Apply changes:** Synchronize these incremental updates with your Amazon Redshift cluster.

By integrating CDC into your data workflow, you will ensure that your data warehouse remains current without the processing overhead associated with handling entire datasets. This, in turn, will optimize your data management processes.

4. Workload Management

Amazon Redshift's Workload Management (WLM) feature allows users to dynamically regulate workload priorities, ensuring that brief, fast-executing queries are not impeded by lengthy, time-consuming queries.

Workloads are managed using queues, with customizable rules that can be applied to each queue-illustrated here with the "Eastertime" rule identifier.

A designated short query queue, referred to as "short_query_queue" (short_query_queue: true), is utilized for directing queries that are anticipated to execute quickly, specifically those estimated to run in five seconds or less. Queries that fit this criterion are transitioned from their original queue to the short query queue.

- Key considerations for effective WLM configuration include:
- Maintaining a total slot count/concurrency of 15 or fewer when feasible.
- Establishing query assignment rules tailored to each queue.
- Utilizing query_group to facilitate queue changes by setting query_group to query_group_name.
- Employing wlm_query_slot_count to expedite the vacuuming process for exceptionally large tables by setting wlm_query_slot_count to 3.
- Defining query termination rules for each queue.
- Enabling Sort Query Acceleration (SQA) for enhanced performance.
- If the total concurrency exceeds 15, it is imperative to manage WLM rules using the AWS Command Line Interface (CLI). The CLI will be required to activate short query acceleration when total WLM concurrency surpasses 15 **(Figure 3)**.

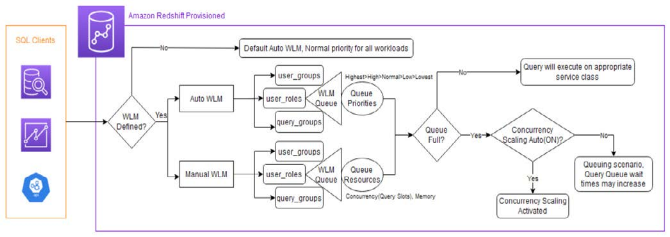

Amazon Redshift Provisioned
SQL Clients
WLM Defined?
No
Yes
Default Auto WLM, Normal priority for all workloads
Auto WLM
Manual WLM
user_groups
user_roles
query_groups
WLM Queue
Queue Priorities
Highest>High>Normal>Low>Lowest
Queue Resources
Concurrency(Query Slots), Memory
Queue Full?
Query will execute on appropriate service class
Concurrency Scaling Auto(ON)?
Queuing scenario, Query Queue wait times may increase
Concurrency Scaling Activated

Figure 3: Flow chat for machine learning (ML) workloads with WML Queue.

5. Functionality of VACUUM and ANALYZE in Amazon Redshift

Amazon Redshift, being based on PostgreSQL architecture, supports many similar features and commands. To effectively manage table statistics and optimize performance, it is crucial to identify tables that exhibit out-of-date statistics by assessing the number of dead tuples present within a schema.

- **Dead tuples:** In a scenario where a table contains 10,000 records, these would occupy 10,000 memory locations. Upon deleting 5,000 records, those memory locations are not immediately reclaimed; instead, they are merely marked as deleted. These entries are referred to as dead tuples. While space appears to be freed, it cannot be reused until a cleanup operation occurs.
- **VACUUM:** The VACUUM command is essential for removing dead tuples from the database. This operation prepares memory locations for reuse but does not reclaim storage space. Instead, it ensures that the freed locations become available for new data.
- **ANALYZE:** The ANALYZE command is utilized to update statistics, which are crucial for generating optimized execution plans for queries.
- **VACUUM FULL:** While a standard VACUUM does not reclaim space, it merely removes dead tuples. If significant data bloat is observed-indicating uneven data distribution or performance degradation-executing VACUUM FULL is advisable. This operation not only removes dead tuples but also redistributes data more effectively.

By default, Amazon Redshift may bypass vacuuming on tables that are already at least 95% sorted. It's essential to consider that for tables containing billions of rows, even 5% can represent a substantial quantity of entries. Therefore, it's critical to establish appropriate threshold percentages during vacuum operations.

Similarly, the ANALYZE command will skip tables if the out-of-date statistics fall below a 10% threshold. To ensure comprehensive analysis without skipping any tables, it may be necessary to adjust this value to a lower percentage.

5.1. Retrieving statistics information

To assess vacuuming and statistics information, the following SQL queries can be utilized:

- SELECT * FROM SVV_VACUUM_SUMMARY;
- SELECT * FROM SVV_VACUUM_PROGRESS;

6. Advantages of Amazon Redshift

6.1. Massive parallel processing

Amazon Redshift is architected to efficiently manage extensive datasets, ranging from terabytes to petabytes. Business intelligence tools such as Tableau and Power BI leverage Redshift's capabilities, enabling organizations to generate insights and statistical reports that support growth and decision-making.

6.2. Data warehouse cluster availability and durability

In the event of a node drive failure within your data warehouse, Amazon Redshift utilizes backup copies to maintain data accessibility, although this process may result in slower query performance. Redshift will attempt to relocate data to functional drives or replace the malfunctioning hardware.

For optimal performance and reliability, it is advisable to maintain at least two operational components within your data warehouse. In the case of a single component failure, manual intervention will be required for restoration.

Redshift is engineered to handle multiple workloads simultaneously, automatically adjusting to workload fluctuations to ensure both performance efficiency and cost-effectiveness.

6.3. Node failure management

Amazon Redshift boasts a robust fault-tolerance feature. If an individual node fails, the system initiates self-repair processes, temporarily restricting access to data. Redshift prioritizes the most critical data to facilitate a prompt restoration. Similar to drive failures, having a minimum of two nodes enhances reliability; a single node failure will necessitate manual repair.

6.4. Handling availability zone (AZ) outages

In scenarios where the Availability Zone hosting your Amazon Redshift cluster becomes unavailable, access to the cluster will be suspended until power and network connectivity are restored. However, your data remains intact, enabling access to the data warehouse once the AZ is operational again. Additionally, you have the option to restore existing snapshots to a different AZ within the same region, with Redshift prioritizing the restoration of frequently accessed data to expedite query responses.

6.5. Multi-AZ deployment support

Currently, Amazon Redshift clusters are limited to reside within a single Availability Zone. To achieve data redundancy across multiple locations, data can be copied to separate Redshift clusters or accessed directly from Amazon S3 using Redshift Spectrum, bypassing the need for duplication. Furthermore, the entire data warehouse can be migrated to alternate locations through backup restoration.

6.6. Rapid provisioning

Amazon Redshift eliminates the necessity for on-premises server infrastructure traditionally required to efficiently query petabytes of data. This alleviates potential practicality issues associated with onsite capacity limitations, which can lead to slower query responses due to infrastructure constraints.

6.7. Integration with amazon S3 and redshift spectrum

Leveraging Amazon S3, the leading cloud-based data lake solution, allows for the storage of relevant business data. Rather than loading this data into Redshift, Redshift Spectrum enables direct querying of S3 data, significantly reducing the complexity typically involved in configuring ETL (Extract, Transform, Load) processes. Moreover, it allows for seamless joins between S3 data and existing datasets within Redshift.

6.8. Utilization of AWS glue

AWS Glue provides a managed ETL service that operates within a serverless Apache Spark environment, eliminating the need for custom ETL pipeline development for Redshift. Glue excels in data model organization, schema discovery and cataloging, offering an efficient means to manage data across various input sources while facilitating data loading to and from Redshift.

6.9. Real-time data ingestion with amazon kinesis data firehose

Kinesis Data Firehose enables the capture, transformation and near real-time loading of streaming data directly into Redshift. For instance, it can be utilized to monitor temperature sensor data.

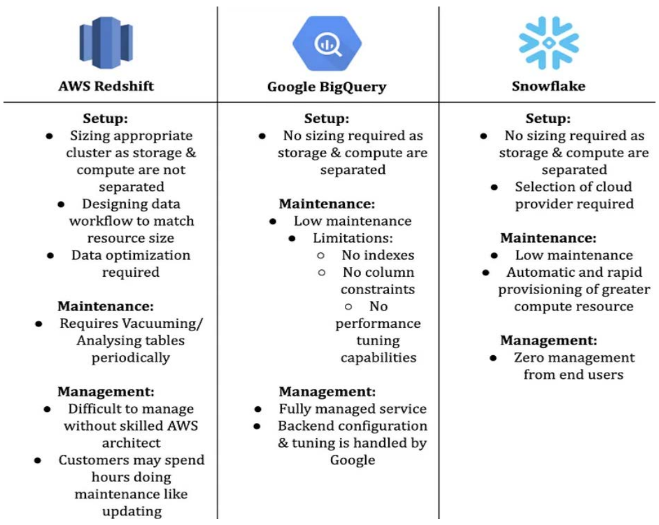
6.10. Business intelligence with amazon quicksight

By integrating Amazon QuickSight organizations can perform business intelligence analytics, creating dynamic dashboards and visualizations for stakeholders across the enterprise.

7. Market Leaders in Data Warehousing Solutions

The market for big data solutions is populated by various vendors, each offering distinct features tailored to specific organizational needs. When selecting a provider organizations should consider factors such as cost and performance. Below are some of the leading data warehousing solutions and their distinguishing characteristics:

- **Amazon redshift:** A fully managed, petabyte-scale cloud-based data warehouse service developed by Amazon Web Services (AWS), Redshift is designed to efficiently handle large volumes of data and facilitate complex analytics.
- **Snowflake:** As a Software as a Service (SaaS) platform, Snowflake provides a cloud-based data warehousing solution that empowers organizations to analyze and extract insights from their data with ease.
- **Google big query:** This cloud-based data warehouse by Google offers a comprehensive big data analytics service, enabling users to process and analyze massive datasets, with capabilities to handle data on the scale of petabytes with high efficiency.



AWS Redshift	Google BigQuery	Snowflake
Setup: • Sizing appropriate cluster as storage & compute are not separated • Designing data workflow to match resource size • Data optimization required	**Setup:** • No sizing required as storage & compute are separated	**Setup:** • No sizing required as storage & compute are separated • Selection of cloud provider required
Maintenance: • Requires Vacuuming/ Analysing tables periodically	**Maintenance:** • Low maintenance • Limitations: ○ No indexes ○ No column constraints ○ No performance tuning capabilities	**Maintenance:** • Low maintenance • Automatic and rapid provisioning of greater compute resource
Management: • Difficult to manage without skilled AWS architect • Customers may spend hours doing maintenance like updating	**Management:** • Fully managed service • Backend configuration & tuning is handled by Google	**Management:** • Zero management from end users

Figure 4: Difference between All leaders.

8. Automation Utilities for Amazon Redshift

An AWS Web Services Lab project is available to facilitate the automation of routine administrative tasks within an Amazon Redshift database. This codebase is designed to streamline operations, enabling automation of various functions including data analysis. Additionally, it incorporates features to publish supplemental CloudWatch metrics, enhancing monitoring capabilities and enabling alerting for proactive system management. This utility serves to optimize database management efforts, improve operational efficiency and ensure timely responses to performance metrics **(Figure 5)**.

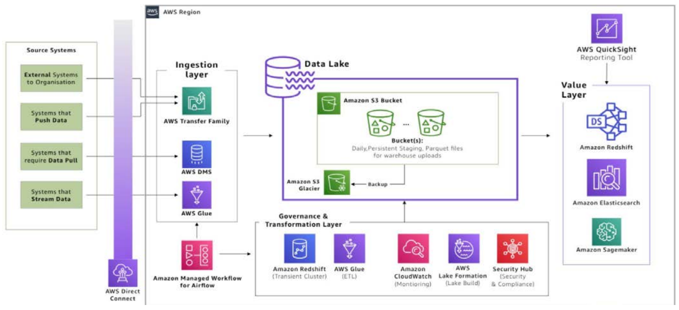

AWS Region
Source Systems
External Systems to Organisation
Systems that Push Data
Systems that require Data Pull
Systems that Stream Data
AWS Direct Connect
Ingestion layer
AWS Transfer Family
AWS DMS
AWS Glue
Amazon Managed Workflow for Airflow
Data Lake
Amazon S3 Bucket
Bucket(s): Daily,Persistent Staging, Parquet files for warehouse uploads
Amazon S3 Glacier
Backup
Governance & Transformation Layer
Amazon Redshift (Transient Cluster)
AWS Glue (ETL)
Amazon CloudWatch (Montioring)
AWS Lake Formation (Lake Build)
Security Hub (Security & Compliance)
AWS QuickSight Reporting Tool
Value Layer
Amazon Redshift
Amazon Elasticsearch
Amazon Sagemaker

Figure 5: Automation Advantages.

9. Monitoring Capabilities in Amazon Redshift

Amazon Redshift offers comprehensive performance metrics and data that allow users to monitor the health and performance of their clusters and databases. This section elaborates on the types of data available within Amazon Redshift, particularly through the Amazon Redshift console. The performance data is categorized into two primary types:

- **Amazon cloud watch metrics:** These metrics assist in monitoring the physical parameters of the cluster, including CPU utilization, latency and throughput. Metric data is readily available within the Amazon Redshift console and can also be accessed through the Amazon CloudWatch console. Furthermore, users can integrate this data using the Amazon CloudWatch Command Line Interface (CLI) or any of the AWS Software Development Kits (SDKs) for further analysis.
- **Query and load performance data:** This data enables monitoring of database activities and performance metrics. In the Amazon Redshift console, performance data is aggregated to facilitate the correlation between observed CloudWatch metrics and specific database queries or loading events. Users can also develop custom performance queries, executing them directly in the database. It is important to note that query and load performance data is exclusively displayed in the Amazon Redshift console and is not published as Amazon CloudWatch metrics.

9.1. CloudWatch dashboard recommendations

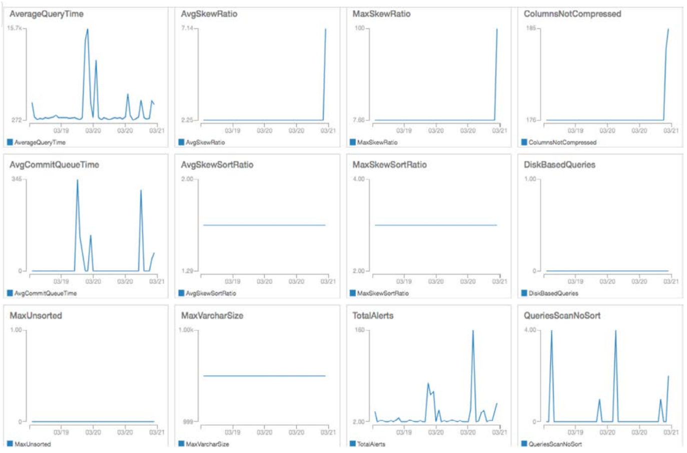


Figure 6: CloudWatch reference.

10. Conclusion

By adhering to the best practices delineated in this paper organizations can successfully scale their data processing workloads within the Amazon Redshift environment. Critical strategies encompass optimizing cluster architecture, enhancing query performance, utilizing efficient data loading methodologies and leveraging Redshift's advanced functionalities. The implementation of these recommendations positions organizations to attain optimal performance, minimize operational costs and effectively address the requirements of their data-intensive applications. Thus, adopting these best practices not only improves efficiency but also ensures that organizations remain competitive in an increasingly data-driven landscape.

11. References

1. https://docs.aws.amazon.com/redshift/latest/dg/best-practices.html
2. https://aws.amazon.com/blogs/big-data/category/database/amazon-redshift/
3. https://www.amazon.com/stores/author/B00Q43XKD6
4. https://www.datacamp.com/tutorial/guide-to-data-warehousing-on-aws-with-redshift
5. Rajesh Francis, Rajiv Gupta, Milind Oke. Amazon Redshift: The Definitive Guide: Jump-Start Analytics Using Cloud Data Warehousing, 2023.
6. Boric N, Gildhoff H, Karavelas M, Pandis I and Tsalouchidou I. Unified spatial analytics from heterogeneous sources with Amazon Redshift. In SIGMOD, 2020.
7. Cai M, Grund M, Gupta A, Nagel F, Pandis I, Papakonstantinou Y and Petropoulos M. Integrated querying of SQL database data and S3 data in Amazon Redshift. IEEE Data Eng. Bull., 2018;41.
8. DeCandia G, Hastorun D, Jampani M, Kakulapati G, Lakshman A, Pilchin A, Sivasubramanian S, Vosshall P and Vogels W. Dynamo: Amazon's highly available key-value store. In SOSP, 2007.
9. Greer R. Daytona and the fourth-generation language cymbal. In SIGMOD, 1999.
10. Gupta A, Agarwal D, Tan D, Kulesza J, Pathak R, Stefani S and Srinivasan V. Amazon Redshift and the case for simpler data warehouses. In SIGMOD, 2015.
11. Huang P, Guo C, Zhou L, Lorch JR, Dang Y, Chintalapati M and Yao R. Gray failure: The Achilles' Heel of cloud-scale systems. In HotOS, 2017.
12. Krikellas K, Viglas S and Cintra M. Generating code for holistic query evaluation. In ICDE, 2010.
13. Leis V, Boncz PA, Kemper A and Neumann T. Morsel-driven parallelism: a NUMA-aware query evaluation framework for the many-core age. In SIGMOD, 2014.
14. Palkar S, Thomas JJ, Narayanan D, Thaker P, Palamuttam R, Negi P, Shanbhag A, Schwarzkopf M, Pirk H, Amarasinghe SP, Madden S and Zaharia M. Evaluating end-to-end optimization for data analytics applications in Weld. PVLDB, 2018;11.
15. Parchas P, Naamad Y, Bouwel PV, Faloutsos C and Petropoulos M. Fast and effective distribution-key recommendation for Amazon Redshift. PVLDB, 2020;13.
16. Ports DRK and Grittner K. Serializable snapshot isolation in PostgreSQL. PVLDB, 2012;5.
17. Raman V, Attaluri GK, Barber R. DB2 with BLU acceleration: So much more than just a column store. PVLDB, 2013;6.