International Journal of Core Engineering & Management
Volume-7, Issue-07, 2023 ISSN No: 2348-9510

SCALING DATA PROCESSING WITH AMAZON REDSHIFT DBA BEST PRACTICES FOR HEAVY LOADS

Balakrishna Boddu
balakrishnasvkbs@gmail.com
Sr. Database Administrator

Abstract

Online Analytical Processing (OLAP) plays a critical role in organizations as it enables the generation of insights from large datasets, helping businesses assess performance and make data-driven decisions. As companies grow, the volume of data increases significantly, presenting a real challenge for timely and efficient data analysis. In this paper, we will explore Amazon Redshift, a highly intelligent and scalable data warehouse solution designed to handle the demands of modern-day data processing. Redshift not only processes vast amounts of data efficiently but also continuously evolves to support complex analytical workloads. This research will focus on understanding the architecture and capabilities of AWS Redshift, its optimization techniques, and its best practices for scaling data analytics operations under heavy workloads.

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

I. INTRODUCTION

Redshift is a Cloud data warehouse product that is designed to run huge data sets faster and fully managed and it's very fast compared to other data warehouse products. It Will use the MMP (Massively Parallel Processing) Algorithm to process data in milliseconds. A cluster is composed of one or more compute nodes. If a cluster is provisioned with two or more compute nodes, an additional leader node coordinates the compute nodes and handles external communication. Your client application interacts directly only with the leader node. The compute nodes are transparent to external applications.

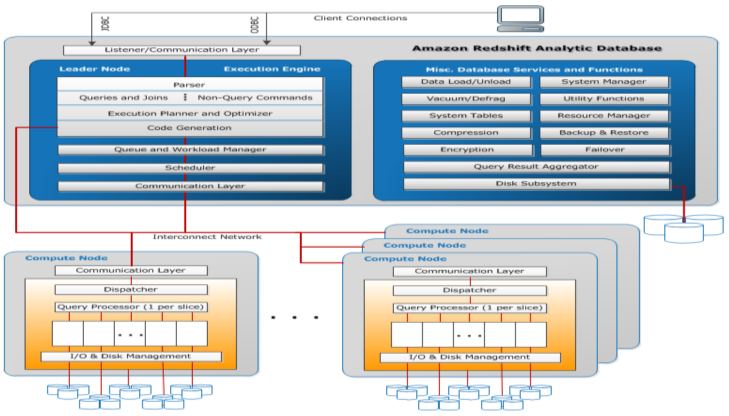


Diagram 1: Map-R algorithm structure of Redshift Cluster
(Ref:https://docs.aws.amazon.com/redshift/latest/dg/c_internal_arch_system_operation.html)

As organizations increasingly rely on data-driven decision-making, the need for efficient and scalable data processing solutions has become paramount. Amazon Redshift, a cloud-based data warehouse service, offers a powerful and cost-effective platform for handling large-scale data workloads. However, maximizing performance and scalability in Amazon Redshift requires a deep understanding of best practices and optimization techniques. This paper explores the challenges and strategies associated with scaling data processing in Amazon Redshift, focusing on the critical role of database administrators (DBAs) in ensuring optimal performance under heavy loads. We will delve into key areas such as query optimization, data modelling, cluster configuration, and performance monitoring, providing actionable insights and recommendations for DBAs seeking to effectively manage and scale their Redshift environments.

II. PROOF OF CONCEPT:

When it comes to the Proof concept for Redshift it's more about business requirements like what data needs to be processed and how to provide security and encryption for the data. Choosing the right distribution styles for balancing such as cluster size, compression encoding methods, sort keys, and table constraints, and testing your system with data is as close to real data as possible.

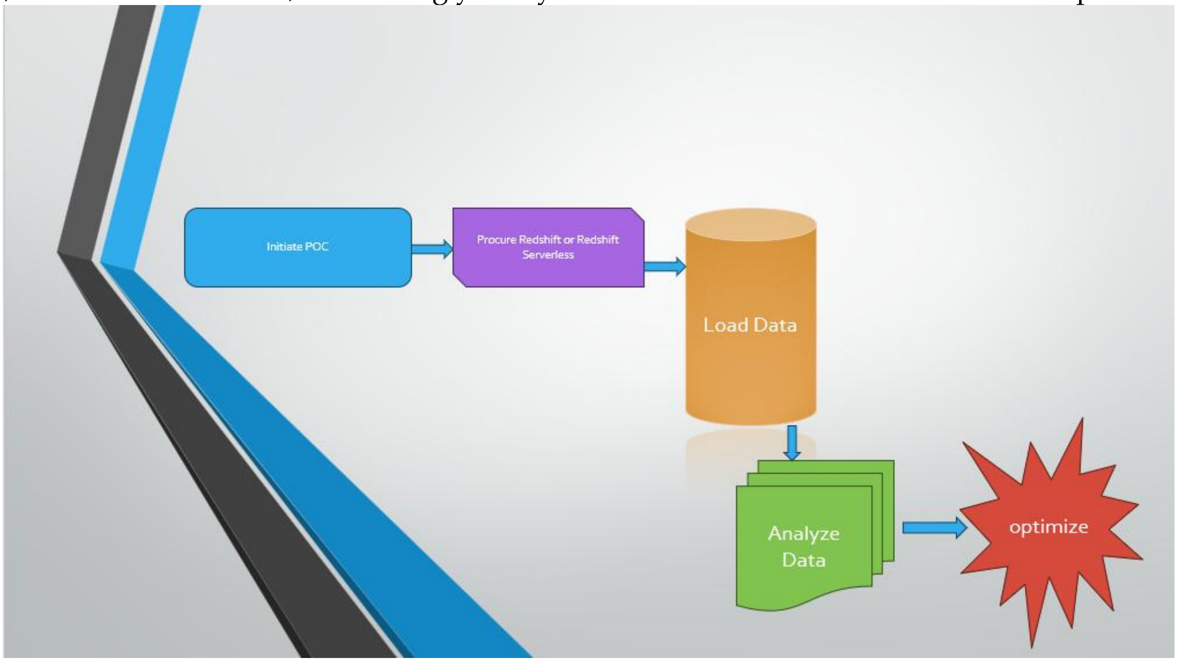


Diagram 2: Etl Process

III. EFFICIENT METHODOLOGY:

The best way to get redshift cluster work effectively and gain more performance

1. Right distribution key:

Choosing the correct distribution is the main key factor run data loads there are

- Auto: AWS Redshift will decide an optimal distribution style based on data if it's a small table it will assign ALL Distribution styles if the data increases it will move to KEY Distribution. If again the table grows larger and there is no column suitable for any distribution key it will move to EVEN Distribution.

- EVEN: The leader does not distribute data across slices in a round-robin fashion regardless of the values of any particular column. It's good when a table doesn't participate in joining.
- KEY: Rows are spreading according to the values in a Column. The leader node will place the same matching values in the same node slices.
- ALL: Tables will be distributed to even nodes, where EVEN distribution or KEY distribution places only a portion of a table's rows on each node, All distribution ensures that every row is allocated for joining the table participated.

2. Best Instance Class:
Selecting a Cluster Class is one of the major decisions since it has to be evaluated based on data distribution and Cost. There are 2 types of Classes.

A. **DC2:** enables you to have compute-intensive data warehouses with local SSD storage included. You choose the number of nodes you need based on data size and performance requirements. DC2 nodes store your data locally for high performance, and as the data size grows, you can add more compute nodes to increase the storage capacity of the cluster.

B. **RA3:**RA3 nodes are a type of computer that can be used to store and process large amounts of data. You can choose how many of these nodes you need based on how fast you want your data to be processed. You only pay for the storage you use, making it a cost-effective solution. When choosing the right type of RA3 node, consider how much data you process each day.

- **Workload type:** Analyse the nature and demands of your workload.
- **Memory needs:** More complex queries require more memory.
- **Storage requirements:** Expect data volume to be stored internally vs. externally on S3.

3. Sort key strategies:
In regular RDBMs We use cluster index, non-cluster index, and other indexes to get a better execution plan but in Redshift it's a columnar database so there is a sort key mechanism that will play a good role in designing query patterns.

- Correct aligning the sort key can help redshift the engine to fine necessarily row and reduce data scanning.
- Use the timestamp column as the leading column if your quires use recent data, it will handle time-bound quires.
- In Some cases, if we leave it as a Default distribution when workload patterns are unclear redshift will use AUTO can be beneficial too

4. Concurrency scaling:
We can take advantage of scaling pricing, if the main cluster is running 24 hours we can accumulate one hour of credit for concurrency Scaling. Data warehouse load is unpredictable so concurrency scaling will help maintain query performance. Concurrency scaling ensures your critical workloads have the necessary memory resources by intelligently allocating them across clusters.

5. SQA:
Short Query Acceleration (SQA) is a feature that helps your computer prioritize quick tasks over longer ones. This means that simple, quick tasks won't have to wait for bigger, more complex tasks

iJCEM
International Journal of Core Engineering & Management
International Journal of Core Engineering & Management
Volume-7, Issue-07, 2023 ISSN No: 2348-9510

to finish. This is especially helpful when you're working with large amounts of data, as it allows you to get quick answers to simple questions without waiting for longer processes to complete.

6. CDC:

Change data capture (CDC) effectively tracks and applies changes from your data source to your data warehouse. By using CDC, you enhance your query performance by only processing data that has changed, rather than managing full loads.

To implement CDC, consider tools like AWS Glue or third-party software that can capture changes from various sources.

7. Here's how you can start benefiting from CDC:

Identify changes: Pinpoint new, updated, or deleted rows in your data source.
Capture changes: Use a CDC mechanism to log these changes efficiently.
Apply changes: Sync these incremental updates to your Redshift cluster.
By integrating CDC, you'll ensure that your data warehouse remains up-to-date without the overhead of processing entire datasets. In turn, you'll streamline your data management process.

IV. WORKLOAD MANAGEMENT:

1. Amazon Redshift workload management (WLM) enables users to flexibly manage priorities within workloads so that short, fast-running queries won't get stuck in queues behind long-running queries.

Queues are used to manage the different workload categories. Rules can be added to each queue as shown below in the Eastertime rule name.

A short query queue ("short_query_queue": true) is used to run queries that are queued but are estimated to be short-running queries, in this case, 5 seconds or less. The queries are moved from an existing queue to the short-running queue.

Things to consider for the WLM configuration:

Total slot count/concurrency should be 15 or less if possible

- Query assignment rules for each queue
- Use query_group to change queues
  - set query_qroup to query_group_name
- Use wlm_query_slot_count to speed up Vacuuming of very large tables
  - set wlm_query_slot_count to 3
- Query termination rules for each queue
- Enabling sort query acceleration
  - If total concurrency is greater than 15, the WLM rules must be managed using the AWS CLI. The CLI is required to enable short query acceleration when WLM total concurrency is greater than 15.

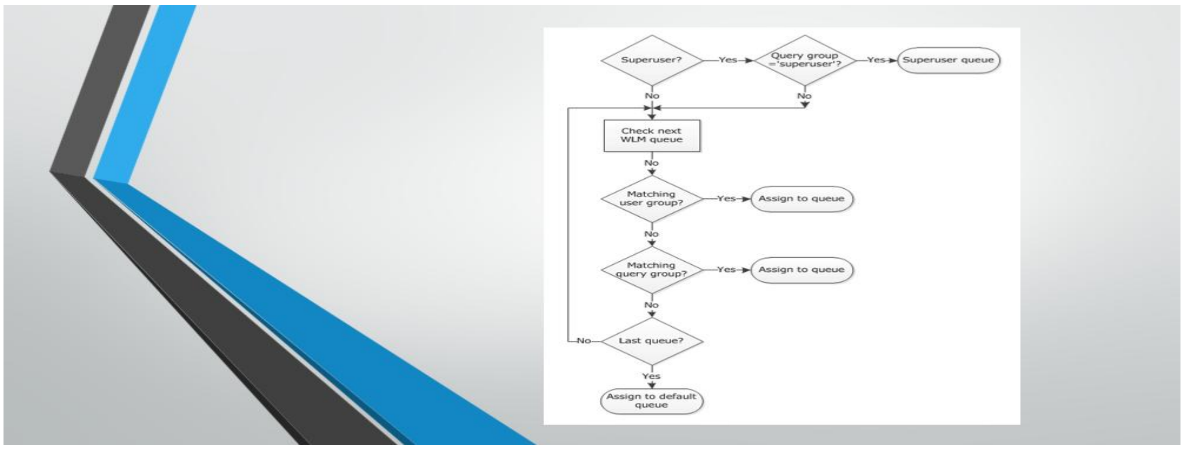


Diagram3: Flow chat for WML Queue

2. How Vacuum and Analyze will work in Redshift:
Since Redshift is designed by Postgres most of the features and commands will work in Redshift too. First To find any tables that are out of stats we need to find in a way how many dead tuples are there in a schema.

- **DEAD TUPLES:** If We have 10k records in a table, it will occupy 10k locations. If we delete 5k records Now 5k memory location should become free but by default it will not become free, it will just be marked as a record deleted, so these types of records we will call dead tuples, you can't reclaim space at the same time you can't reuse those memory locations.
- **VACUUM:** The VACUUM operation will remove the dead tuples and make the location ready for use, it will not reclaim any space for us. But the location will be free, anyone can use it.
- **ANALYZE:** It will Keep stats up to date to Generate a Better execution Plan
- **VACUUM FULL:** VACUUM will not reclaim any space, It will just remove dead tuples, now if you think data is not distributed properly, a huge % of bloats are there, so if you are seeing any performance impact, we will run this vacuum full to remove bloats and distribute data properly.

By default, Redshift can skip the tables from vacuum Sort if the table is already at least 95 percent sorted. So we have to care about this if the tables have billions (or any huge numbers) of rows, then just 5% is a huge number of rows. So while running a vacuum, you are defining the threshold percentage.
Similarly, analyse also will skip the tables from Analysing if the out of stats is up to 10%. So we need to run the following command to set this value to very low and the analysis will not skip any tables
How to find stats information:
SELECT * FROM SVV_VACUUM_SUMMARY;
SELECT * FROM SVV_VACUUM_PROGRESS;

V. ADVANTAGES:

1. Massive Parallel processing:
Redshift is specifically designed to handle massive datasets in terabytes or even petabytes. Most of

the reporting tools like Tableau, and Power BI take advantage of their reports which helps organization statistics and growth.

A. What happens to my data warehouse cluster availability and data durability if a drive on one of my nodes fails?

If a part of your data warehouse (a drive) breaks, Amazon Redshift will use a backup copy to keep your data accessible. However, this might slow down some queries. Redshift will try to move your data to a working drive or replace the broken part.
For the best performance and reliability, it's best to have at least two parts to your data warehouse. If you only have one, and it breaks, you'll need to manually fix it.
Redshift is designed to handle a lot of work at once. It can automatically adjust to changes in workload to ensure optimal performance and cost-effectiveness.

B. What happens to my data warehouse cluster availability and data durability in the event of individual node failure?

Amazon Redshift is a reliable data warehouse. If a part of it breaks, it will automatically fix itself. While it's fixing itself, you won't be able to access your data. Redshift will prioritize the most important information to get your data back as quickly as possible. For the best performance and reliability, it's best to have at least two parts to your data warehouse. If you only have one, and it breaks, you'll need to manually fix it. Redshift constantly monitors itself and automatically copies your data to protect it. It also saves regular backups in case of a major problem.

C. What happens to my data warehouse cluster availability and data durability if my data warehouse cluster's availability zone (AZ) has an outage?

If your Amazon Redshift data warehouse cluster's Availability Zone becomes unavailable, you will not be able to use your cluster until power and network access to the AZ are restored. Your data warehouse cluster's data is preserved so you can start using your Amazon Redshift data warehouse as soon as the AZ becomes available again. In addition, you can also choose to restore any existing snapshots to a new AZ in the same Region. Amazon Redshift will restore your most frequently accessed data first so you can resume queries as quickly as possible.

2. Does Amazon Redshift Support Multi-AZ Deployments?

Right now, Amazon Redshift data warehouses can only be in one location (AZ). To have your data in multiple locations, you can either copy it to two different Redshift warehouses or use a feature called Redshift Spectrum to access your data from S3 storage without copying it. You can also move your entire data warehouse to a different location using a backup.

3. Ready to Provision in Minutes:

Redshift does away with the onsite server infrastructure required to handle querying petabytes of data quickly. It’s not always practical to have this capacity onsite, or when you do, query times may be slower due to infrastructure limitations.

4. S3 with Redshift Spectrum:

Amazon’s foremost data lake product, Amazon S3, can be used to store all applicable business data in the cloud. Instead of loading this data into Redshift, you can query it directly from S3 when you use the Redshift-included service Spectrum. This eliminates a fair part of the work involved in manually configuring the ETL process. Bonus: you can even join data directly from S3 with data

International Journal of Core Engineering & Management
Volume-7, Issue-07, 2023 ISSN No: 2348-9510

already in Redshift.

5. Aws Glue:
Instead of developing your own ETL pipeline for use with Redshift, you can use Amazon's managed ETL service, AWS Glue, which runs these jobs in a serverless Apache Spark environment. AWS Glue is not only a data model organization, schema discovery, and cataloging product that can help sort data across your multiple data source inputs, but it's also an easy way to load your data to and from Redshift.

6. Amazon Kinesis data firehose:
Use Kinesis Data Firehouse to capture, transform, and load real-time streaming data directly into Redshift in near real-time. You can check out a simple temperature sensor example here.

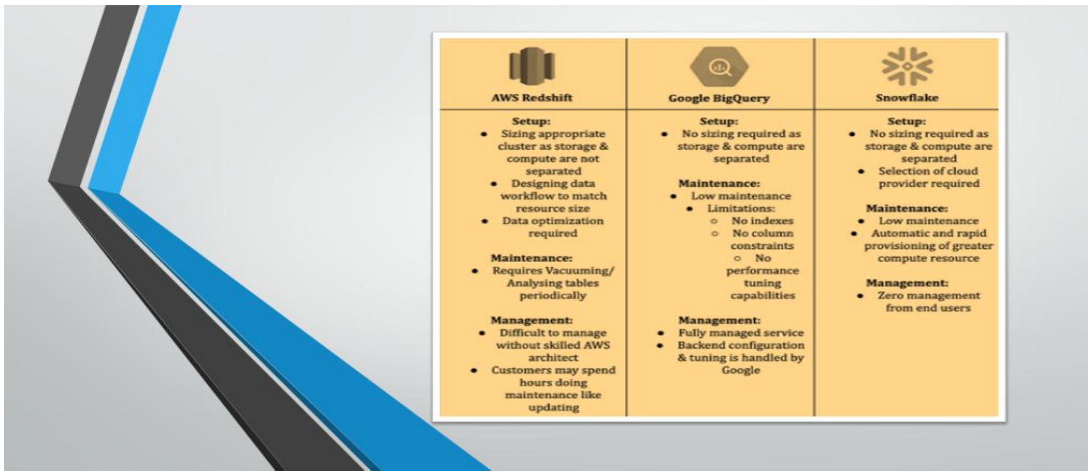
7. Amazon Quick Sight:
Connect with Quick Sight for Business Intelligence analytics, creating dashboards and charts for users across your organization.

VI. MARKET LEADERS:
There are more big data vendors in the market for different usages, Organizations need to pick based on their needs with Cost and performance, here are the leaders below and their differences.

1. **Redshift:** is a fully managed petabyte-scale Cloud-based Data Warehouse service designed by Amazon to handle large data.

2. **Snowflake:** Its SAAS Service (software as a service) where allows customers to Analyze data.

3. **Google Big Query:** Google Big Query is a Cloud-based Data Warehouse that offers a big data analytic web service for processing very large datasets over petabytes of data



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 **Maintenance:** • Requires Vacuuming/ Analysing tables periodically **Management:** • Difficult to manage without skilled AWS architect • Customers may spend hours doing maintenance like updating	**Setup:** • No sizing required as storage & compute are separated **Maintenance:** • Low maintenance • Limitations: ○ No indexes ○ No column constraints ○ No performance tuning capabilities **Management:** • Fully managed service • Backend configuration & tuning is handled by Google	**Setup:** • No sizing required as storage & compute are separated • Selection of cloud provider required **Maintenance:** • Low maintenance • Automatic and rapid provisioning of greater compute resource **Management:** • Zero management from end users

Diagram 4: Difference between All leaders

International Journal of Core Engineering & Management
Volume-7, Issue-07, 2023 ISSN No: 2348-9510

VII. AUTOMATION UTILITIES:

There is an AWS Web Services Lab project available to help automate some of the most common administrative tasks on a Redshift database. The code can be used to help automate tasks such as analysis. It includes the ability to publish additional Cloud Watch metrics for monitoring and alerting.

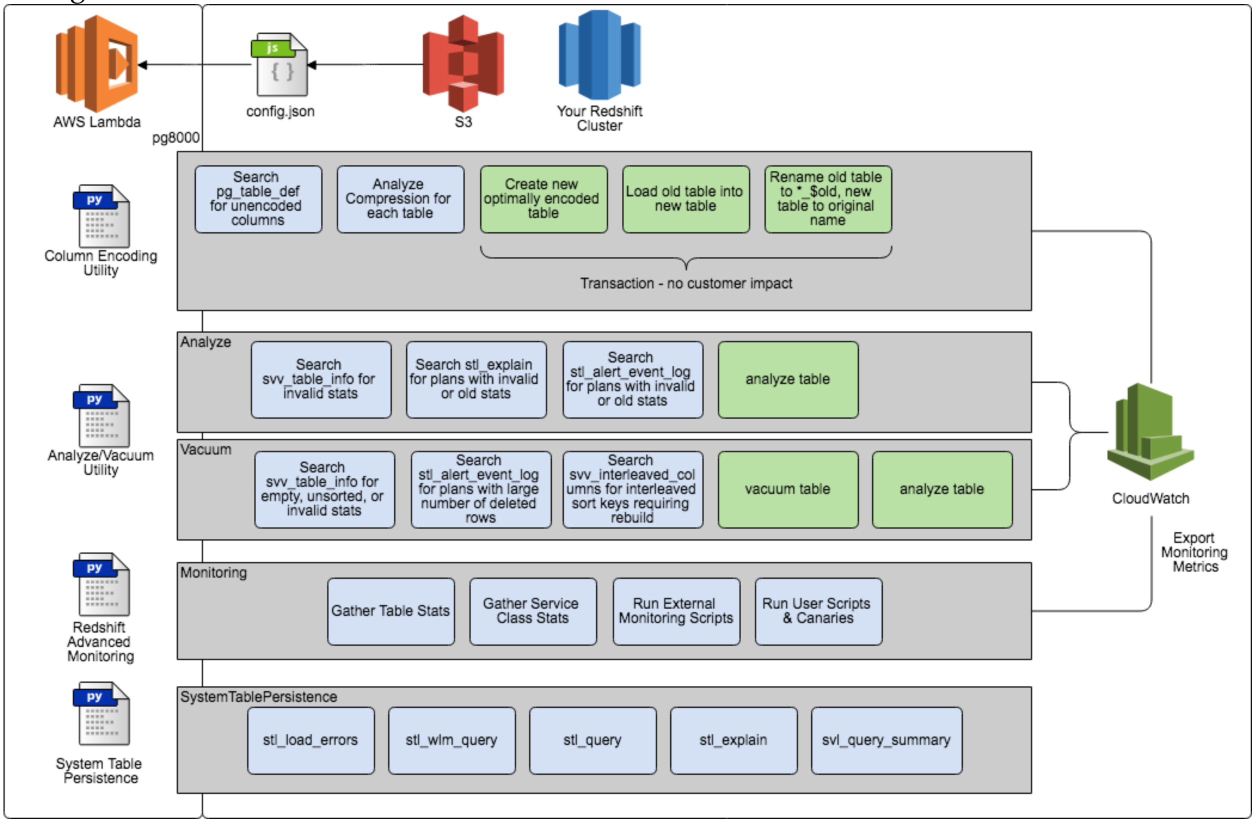

AWS Lambda
config.json
S3
Your Redshift Cluster
pg8000
Column Encoding Utility
Search pg_table_def for unencoded columns
Analyze Compression for each table
Create new optimally encoded table
Load old table into new table
Rename old table to *_$old, new table to original name
Transaction - no customer impact
Analyze/Vacuum Utility
Analyze
Search svv_table_info for invalid stats
Search stl_explain for plans with invalid or old stats
Search stl_alert_event_log for plans with invalid or old stats
analyze table
Vacuum
Search svv_table_info for empty, unsorted, or invalid stats
Search stl_alert_event_log for plans with large number of deleted rows
Search svv_interleaved_columns for interleaved sort keys requiring rebuild
vacuum table
analyze table
CloudWatch
Export Monitoring Metrics
Redshift Advanced Monitoring
Monitoring
Gather Table Stats
Gather Service Class Stats
Run External Monitoring Scripts
Run User Scripts & Canaries
System Table Persistence
SystemTablePersistence
stl_load_errors
stl_wlm_query
stl_query
stl_explain
svl_query_summary

Diagram 5: Automation Advantages

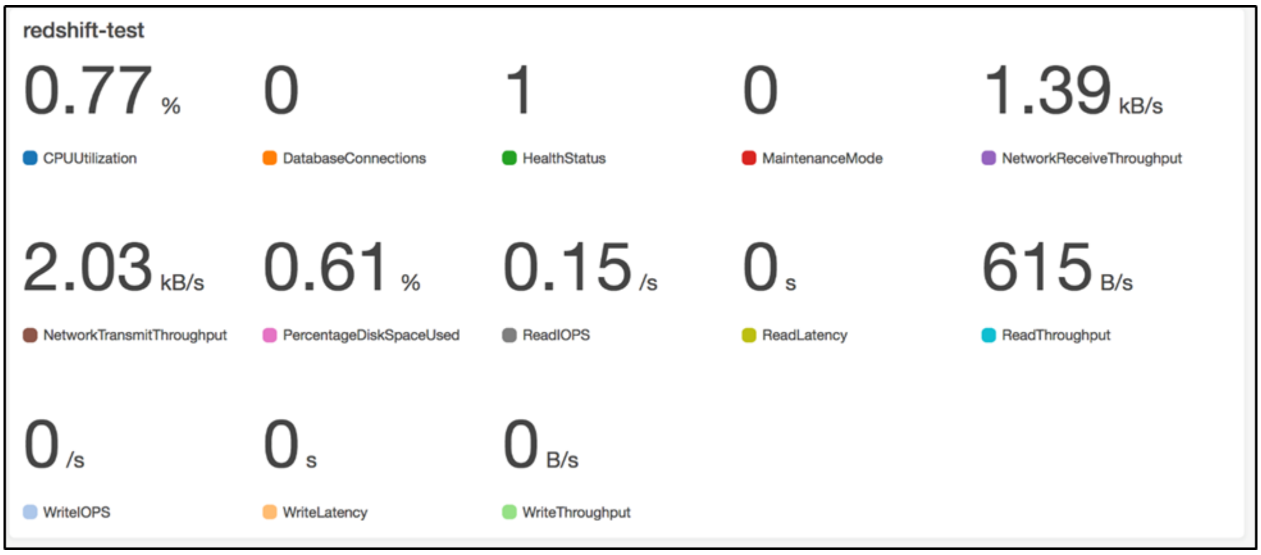
VIII. MONITORING

Amazon Redshift provides performance metrics and data so that you can track the health and performance of your clusters and databases. In this section, we discuss the types of data you can work with in Amazon Redshift and specifically, in the Amazon Redshift console. The performance data that you can use in the Amazon Redshift console falls into two categories:

1. **Amazon Cloud Watch Metrics** — Amazon Cloud Watch metrics help you monitor physical aspects of your cluster, such as CPU utilization, latency, and throughput. Metric data is displayed directly in the Amazon Redshift console. You can also view it in the Amazon Cloud Watch console, or you can consume it in any other way you work with metrics such as with the Amazon Cloud Watch Command Line Interface (CLI) or one of the AWS Software Development Kits (SDKs).
2. **Query/Load Performance Data** — Performance data helps you monitor database activity and performance. This data is aggregated in the Amazon Redshift console to help you easily correlate what you see in Amazon Cloud Watch metrics with specific database queries and

load events. You can also create your custom performance queries and run them directly on the database. Query and load performance data is displayed only in the Amazon Redshift console. It is not published as Amazon Cloud Watch metrics.

3. **Cloud Watch Dashboard Recommendations**



redshift-test

0.77 %	0	1	0	1.39 kB/s
● CPUUtilization	● DatabaseConnections	● HealthStatus	● MaintenanceMode	● NetworkReceiveThroughput
2.03 kB/s	0.61 %	0.15 /s	0 s	615 B/s
● NetworkTransmitThroughput	● PercentageDiskSpaceUsed	● ReadIOPS	● ReadLatency	● ReadThroughput
0 /s	0 s	0 B/s		
● WriteIOPS	● WriteLatency	● WriteThroughput		

Diagram 6: Cloud Watch reference

IX. CONCLUSION

By following the best practices outlined in this paper, organizations can effectively scale their data processing workloads using Amazon Redshift. Key strategies include optimizing cluster design, fine-tuning query performance, employing efficient data loading techniques, and leveraging Redshift's advanced features. By implementing these recommendations, organizations can achieve optimal performance, reduce costs, and meet the demands of their data-intensive applications.

REFERENCES

1. AWS Redshift Documentation: https://docs.aws.amazon.com/redshift/latest/dg/best-practices.html
2. [2]. Amazon Redshift Blog: https://aws.amazon.com/blogs/big-data/category/database/amazon-redshift/
3. [3]. High Performance With Redshift (book): https://www.amazon.com/stores/author/B00Q43XKD6
4. [4]. Data Warehousing with Amazon Redshift (book): https://www.datacamp.com/tutorial/guide-to-data-warehousing-on-aws-with-redshift
5. [5]. Amazon Redshift: The Definitive Guide: Jump-Start Analytics Using Cloud Data Warehousing: by Rajesh Francis (Author), Rajiv Gupta (Author), Milind Oke (Author)
6. [6] N. Boric, H. Gildhoff, M. Karavelas, I. Pandis, and I. Tsalouchidou. Unified spatial analytics from heterogeneous sources with Amazon Redshift. In SIGMOD, 2020.
7. [7] M. Cai, M. Grund, A. Gupta, F. Nagel, I. Pandis, Y. Papakonstantinou, and M. Petropoulos. Integrated querying of SQL database data and S3 data in Amazon Redshift. IEEE Data Eng.

Bull., 41(2), 2018.

8. [8] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels. Dynamo: Amazon's highly available key-value store. In SOSP, 2007. [11] G. Graefe. Volcano- an extensible and parallel query evaluation system. IEEE Trans. Knowl. Data Eng., 6(1), 1994.
9. [9] R. Greer. Daytona and the fourth-generation language cymbal. In SIGMOD, 1999.
10. [10] A. Gupta, D. Agarwal, D. Tan, J. Kulesza, R. Pathak, S. Stefani, and V. Srinivasan. Amazon Redshift and the case for simpler data warehouses. In SIGMOD, 2015.
11. [11] P. Huang, C. Guo, L. Zhou, J. R. Lorch, Y. Dang, M. Chintalapati, and R. Yao. Gray failure: The Achilles' Heel of cloud-scale systems. In HotOS, 2017.
12. [12] K. Krikellas, S. Viglas, and M. Cintra. Generating code for holistic query evaluation. In ICDE, 2010.
13. [13] V. Leis, P. A. Boncz, A. Kemper, and T. Neumann. Morsel-driven parallelism: a NUMA-aware query evaluation framework for the many-core age. In SIGMOD, 2014.
14. [14] S. Palkar, J. J. Thomas, D. Narayanan, P. Thaker, R. Palamuttam, P. Negi, A. Shanbhag, M. Schwarzkopf, H. Pirk, S. P. Amarasinghe, S. Madden, and M. Zaharia. Evaluating end-to-end optimization for data analytics applications in Weld. PVLDB, 11(9), 2018.
15. [15] P. Parchas, Y. Naamad, P. V. Bouwel, C. Faloutsos, and M. Petropoulos. Fast and effective distribution-key recommendation for Amazon Redshift. PVLDB, 13(11), 2020.
16. [16] D. R. K. Ports and K. Grittner. Serializable snapshot isolation in PostgreSQL. PVLDB, 5(12), 2012.
17. [17] V. Raman, G. K. Attaluri, R. Barber, N. Chainani, D. Kalmuk, V. KulandaiSamy, J. Leenstra, S. Lightstone, S. Liu, G. M. Lohman, T. Malkemus, R. Müller, I. Pandis, B. Schiefer, D. Sharpe, R. Sidle, A. J. Storm, and L. Zhang. DB2 with BLU acceleration: So much more than just a column store. PVLDB, 6(11), 2013.