ISSN: 2583-9888
DOI: doi.org/10.51219/JAIMLD/srinivasa-rao-karanam/513

URF PUBLISHERS
connect with research world

Journal of Artificial Intelligence, Machine Learning and Data Science

https://urfpublishers.com/journal/artificial-intelligence

Vol: 2 & Iss: 2

Research Article

Dynamic Table Processing in Snowflake: A Declarative Approach to Modern Data Engineering

Srinivasa Rao Karanam*

Citation: Karanam SR. Dynamic Table Processing in Snowflake: A Declarative Approach to Modern Data Engineering. *J Artif Intell Mach Learn & Data Sci* 2024, 2(2), 2373-2376. DOI: doi.org/10.51219/JAIMLD/srinivasa-rao-karanam/513

Received: 03 May, 2024; **Accepted:** 28 May, 2024; **Published:** 30 May, 2024

***Corresponding author:** Srinivasa Rao Karanam, New Jersey, USA, E-mail: Srinivasarao.karanam@gmail.com

ABSTRACT

Snowflake Dynamic Tables offer a declarative approach to data pipeline creation, enabling automatic refreshes and reducing manual intervention. The paper discusses the technical implementation of Dynamic Tables, their use cases and how to improve on the efficiency with which they are used in real time data transformation workflows. In this article, we use detailed use cases and architectural considerations to provide data engineers with the essentials they need in order to make effective use of Snowflake's Dynamic Tables to solve today's data engineering problems.

Keywords: Dynamic Table Processing, Snowflake, Declarative Approach, Data Engineering, Data Pipeline, Automatic Refresh, Manual Intervention, Real-Time Data Transformation, SQL, Target Lag

1. Introduction

The large and ever-growing data demands innovative solutions to process and properly manage the growing data. Typically, handling dependencies in Traditional Extract-Transform-Load (ETL) processes is a cumbersome, manual task and the level of orchestration is high. However, Dynamic Tables, a feature of Snowflake's cloud-based data platform, helps tackle these challenges. Dynamic Tables make data transformation workflows safer, more efficient and scalable, by providing a SQL–based declarative approach for doing data transformations.

2. Overview Of Snowflake Dynamic Tables

Dynamic Tables are specialized tables in Snowflake that automatically refresh to reflect changes in source data, based on predefined schedules or a "target lag." These tables maintain the results of a SQL query as a materialized view, optimizing incremental updates to reduce resource consumption and latency.

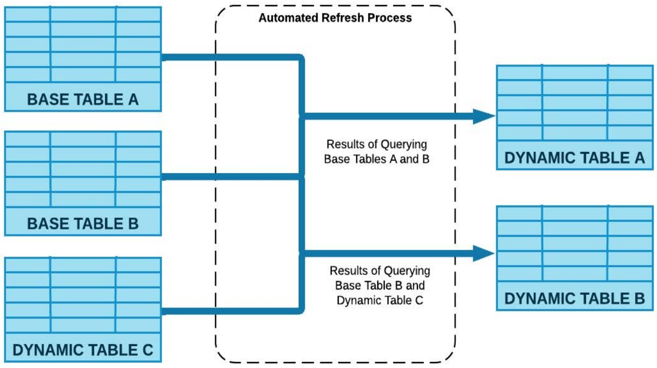
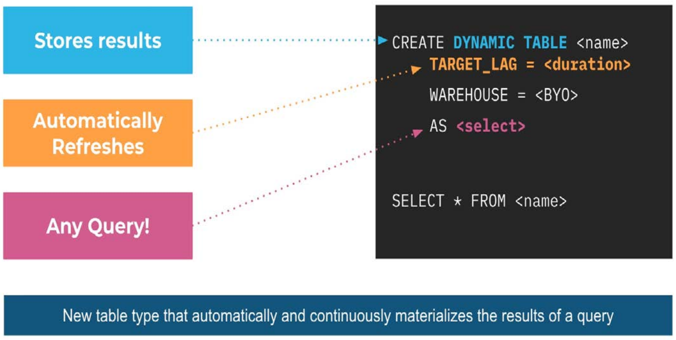

BASE TABLE A
BASE TABLE B
DYNAMIC TABLE C
Automated Refresh Process
Results of Querying Base Tables A and B
Results of Querying Base Table B and Dynamic Table C
DYNAMIC TABLE A
DYNAMIC TABLE B

Figure 1: Illustration of Automated Table Refresh Process: Dynamic Tables Generated from Base Tables Through Querying.

- **Target lag:** Defines the freshness of the data. For instance, a target lag of '5 minutes' ensures that the table updates every five minutes.
- **Warehouse assignment:** Specifies the computational resources used for refresh operations.

- **Incremental processing:** Supports updating only the changed data, enhancing performance.

3. Technical Implementation


Stores results
Automatically Refreshes
Any Query!
New table type that automatically and continuously materializes the results of a query

```
CREATE DYNAMIC TABLE <name>
    TARGET_LAG = <duration>
    WAREHOUSE = <BYO>
    AS <select>

SELECT * FROM <name>
```

Figure 2: Illustration of Dynamic Tables: Automatically Refreshing, Storing Results and Supporting Any Query.

To create a Dynamic Table, users write SQL that specifies the desired transformation logic, accompanied by a target lag and warehouse details. The syntax is as follows:

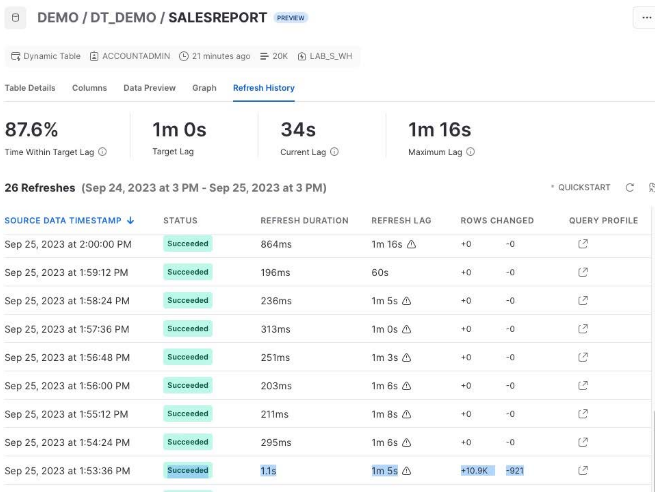
```
CREATE OR REPLACE DYNAMIC
TABLE my_dynamic_table
TARGET_LAG = '5 minutes'
WAREHOUSE = my_warehouse
AS
SELECT
 col1,
 col2,
 COUNT(col3) AS col3_count
FROM
 source_table
WHERE
 col4 = 'condition'
GROUP BY
 col1, col2;
```

Explanation:

- my_dynamic_table: The name of the Dynamic Table.
- TARGET_LAG: Defines the maximum delay before the table reflects updates.
- WAREHOUSE: Allocates the compute resources for managing updates.
- SELECT Statement: Contains the transformation logic to process source data.

4. Advantages of Dynamic Tables

- **Declarative pipeline design:** Eliminates the need for procedural logic by enabling users to focus on the desired outcome.
- **Automation:** Automatically handles data refreshes, removing the complexity of managing schedules.
- **Efficiency:** Supports incremental updates, reducing compute costs and improving performance.
- **Scalability:** Simplifies the scaling of complex pipelines by chaining multiple Dynamic Tables.



87.6% Time Within Target Lag

1m 0s Target Lag

34s Current Lag

1m 16s Maximum Lag

26 Refreshes (Sep 24, 2023 at 3 PM - Sep 25, 2023 at 3 PM)

SOURCE DATA TIMESTAMP ↓	STATUS	REFRESH DURATION	REFRESH LAG	ROWS CHANGED	QUERY PROFILE
Sep 25, 2023 at 2:00:00 PM	Succeeded	864ms	1m 16s ⚠	+0 -0	↗
Sep 25, 2023 at 1:59:12 PM	Succeeded	196ms	60s	+0 -0	↗
Sep 25, 2023 at 1:58:24 PM	Succeeded	236ms	1m 5s ⚠	+0 -0	↗
Sep 25, 2023 at 1:57:36 PM	Succeeded	313ms	1m 0s ⚠	+0 -0	↗
Sep 25, 2023 at 1:56:48 PM	Succeeded	251ms	1m 3s ⚠	+0 -0	↗
Sep 25, 2023 at 1:56:00 PM	Succeeded	203ms	1m 6s ⚠	+0 -0	↗
Sep 25, 2023 at 1:55:12 PM	Succeeded	211ms	1m 8s ⚠	+0 -0	↗
Sep 25, 2023 at 1:54:24 PM	Succeeded	295ms	1m 6s ⚠	+0 -0	↗
Sep 25, 2023 at 1:53:36 PM	Succeeded	1.1s	1m 5s ⚠	+10.9K -921	↗

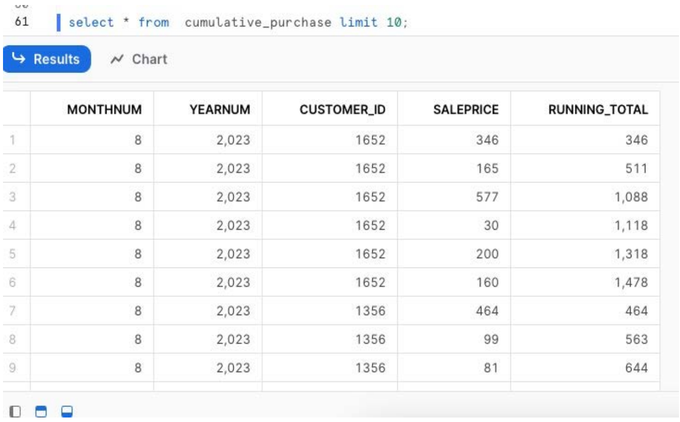
Figure 3: Illustration of Dynamic Table Refresh History: Performance Metrics and Refresh Details Over Time.

5. Best Practices

To maximize the benefits of Dynamic Tables, the following best practices are recommended:

- Use efficient SQL to minimize processing time.
- Avoid unnecessary joins and aggregations.
- Balance freshness requirements with compute costs.
- Shorter lags are ideal for real-time applications; longer lags suit batch processing.
- Assign dedicated warehouses to avoid contention with other workloads.
- Monitor query performance to identify bottlenecks.
- Chain Dynamic Tables to divide complex workflows into manageable steps.
- Use intermediate tables for staging results.
- Validate that only updated rows are processed to ensure efficiency.
- Regularly review query execution plans.

6. Use Cases



```
select * from cumulative_purchase limit 10;
```

	MONTHNUM	YEARNUM	CUSTOMER_ID	SALEPRICE	RUNNING_TOTAL
1	8	2,023	1652	346	346
2	8	2,023	1652	165	511
3	8	2,023	1652	577	1,088
4	8	2,023	1652	30	1,118
5	8	2,023	1652	200	1,318
6	8	2,023	1652	160	1,478
7	8	2,023	1356	464	464
8	8	2,023	1356	99	563
9	8	2,023	1356	81	644

Figure 4: Example of Cumulative Purchase Using Dynamic Table.

Dynamic Tables are ideal for applications requiring near-real-time insights, such as monitoring user behavior on a website. For example, a Dynamic Table can aggregate clickstream data every five minutes, feeding dashboards with up-to-date metrics.

Dynamic Tables replace streams and tasks in many scenarios, reducing the complexity of orchestrating data workflows. This makes them suitable for building automated pipelines that require minimal maintenance.

By joining multiple source tables and materializing the results in a Dynamic Table organizations can perform real-time data enrichment to enhance customer profiles or product catalogues.

7. Advanced Optimization Strategies for Dynamic Tables

While the foundational practices for managing Dynamic Tables have been discussed, advanced optimization strategies can significantly enhance their performance in large-scale data environments. One such strategy is leveraging query prioritization techniques to ensure that critical data transformations are executed first. This is particularly useful in environments where multiple Dynamic Tables compete for shared computational resources. Implementing intelligent resource scheduling algorithms within Snowflake can prevent bottlenecks and optimize resource allocation.

Another advanced method involves tailoring data partitioning strategies to the specific nature of the data. For instance, partitioning by date or location in high-frequency transactional data ensures that updates target only the relevant segments, thereby reducing overhead. Additionally, leveraging Snowflake's clustering capabilities can improve query performance on large datasets by reorganizing the data to align with frequently used query patterns.

Finally, integrating machine learning models into the data transformation pipeline through Dynamic Tables introduces opportunities for predictive analytics. Snowflake's support for external functions allows for seamless integration of model predictions, enabling dynamic enrichment of data within the pipeline. For example, sales forecasts or customer churn predictions can be embedded into transformation workflows, enhancing the value of the output.

8. Role Of Dynamic Tables in Data Democratization

Dynamic Tables contribute significantly to the democratization of data within organizations. By simplifying data workflows and automating complex transformation tasks, they make advanced analytics accessible to non-technical users. This empowerment reduces the dependency on specialized data engineering teams for routine tasks, allowing business units to directly derive insights from raw data.

For example, marketing teams can set up Dynamic Tables to automatically update campaign performance metrics, enabling real-time decision-making. Similarly, finance teams can automate revenue and expense tracking pipelines, eliminating manual reconciliation efforts. This accessibility fosters a culture of data-driven decision-making, where every stakeholder has the tools to explore, analyse and act on data independently.

Dynamic Tables also encourage collaboration by providing a unified platform for data transformation. Teams can work on shared tables without worrying about manual dependencies or data freshness issues. This collaborative framework aligns with modern data engineering principles, emphasizing agility and adaptability in rapidly changing business environments.

9. Integration of Dynamic Tables with Emerging Technologies

The potential for Dynamic Tables expands further when integrated with emerging technologies like artificial intelligence (AI), machine learning (ML) and Internet of Things (IoT). For instance, IoT devices generate massive streams of real-time data that require continuous processing. Dynamic Tables can serve as an intermediary layer to aggregate, transform and store this data for downstream applications, such as predictive maintenance or real-time monitoring dashboards.

In the realm of AI and ML, Dynamic Tables can preprocess training datasets by cleaning, normalizing and aggregating data at regular intervals. This automation accelerates the model training process, ensuring that algorithms are always working with the latest data. Moreover, the integration of Snowflake's external functions allows for real-time scoring of ML models, where predictions can be added as new columns to Dynamic Tables.

Dynamic Tables also enhance the potential of cloud-native architectures by supporting serverless functions. These functions can trigger updates in response to specific events, such as the arrival of new data in source systems. This event-driven architecture creates a seamless flow of data transformations, aligning with the requirements of modern, scalable applications.

10. Addressing the Challenges of Large-Scale Pipelines

Scaling Dynamic Tables for use in enterprise-grade environments presents unique challenges, but careful planning and best practices can mitigate these issues. A key challenge lies in managing dependencies across multiple Dynamic Tables in a pipeline. As workflows grow more complex, the likelihood of performance bottlenecks increases. To address this, Snowflake's query history and dependency visualizations can be utilized to analyze and optimize execution plans.

Another challenge is balancing the trade-off between data freshness and compute costs. Organizations must carefully evaluate the frequency of updates to avoid unnecessary resource consumption. For instance, a high-frequency refresh might be essential for real-time dashboards but may not be justified for batch-processing pipelines. Configuring Dynamic Tables with flexible target lags for different stages of the pipeline ensures efficient resource utilization without compromising on performance.

Furthermore organizations operating across multiple regions must consider the impact of data latency in geographically distributed systems. Dynamic Tables in such setups should leverage Snowflake's replication features to maintain consistency across regions while minimizing latency. This strategy ensures that global teams work with synchronized data, enhancing collaboration and decision-making.

11. Potential Impact on Organizational Data Strategies

The introduction of Dynamic Tables in Snowflake fundamentally shifts how organizations approach their data strategies. Traditional strategies often emphasize periodic, large-scale ETL processes, which are prone to delays and inefficiencies. With Dynamic Tables organizations can adopt a continuous data transformation model, aligning with the principles of real-time analytics and just-in-time processing.

This paradigm shift allows organizations to be more responsive to market changes and operational demands. For instance, retail businesses can use Dynamic Tables to track inventory levels in real time, reducing stockouts and overstock situations. Financial institutions can monitor transaction anomalies immediately, improving their fraud detection capabilities. These advantages position Dynamic Tables as a critical component of forward-looking data strategies.

Moreover, the simplified data pipeline management introduced by Dynamic Tables enables organizations to allocate resources more strategically. Data engineering teams can focus on high-impact projects, such as building advanced analytics models or designing scalable architectures, rather than managing routine data transformations.

12. Future Outlook and Research Directions

Dynamic Tables represent a significant milestone in the evolution of data engineering, but their journey is far from complete. Future research could explore the integration of Dynamic Tables with federated learning systems, where distributed data sources are processed in a privacy-preserving manner. Such advancements could unlock new opportunities in sectors like healthcare and finance, where data privacy regulations are stringent.

Another promising direction is the exploration of Dynamic Tables in multi-cloud environments. As organizations increasingly adopt multi-cloud strategies, ensuring seamless operation across different platforms becomes essential. Dynamic Tables could play a pivotal role in harmonizing data workflows across clouds, enabling unified analytics and reporting.

Lastly, advancements in natural language processing (NLP) could lead to the development of conversational interfaces for managing Dynamic Tables. Such interfaces would allow users to configure and monitor Dynamic Tables using simple, natural language commands, further lowering the barrier to entry for non-technical users.

13. Challenges and Limitations

While Dynamic Tables simplify many aspects of pipeline management, they have some limitations:

- **Cost implications:** Frequent refreshes can lead to higher compute costs if not carefully managed.
- **Complexity in large-scale pipelines:** Chaining multiple Dynamic Tables may require careful planning to avoid performance bottlenecks.
- **Limited flexibility:** Advanced workflows may still require custom scripts or external orchestration tools.

14. Conclusion

Snowflake's Dynamic Tables represent a significant leap forward in simplifying data transformation workflows. By automating refresh processes and supporting declarative pipeline design, they enable data engineers to focus on delivering insights rather than managing infrastructure. While challenges exist, adhering to best practices ensures that Dynamic Tables remain a powerful tool for modern data engineering. Future enhancements, such as tighter integration with external orchestration tools, could further solidify their role in data processing ecosystems.

15. References

1. Borra P. "Snowflake: A Comprehensive Review of a Modern Data Warehousing Platform," in International Journal of Computer Science and Information Technology Research (IJCSITR), 2022;3: 11-16.
2. L'Esteve R. "Snowflake," in The Azure Data Lakehouse Toolkit, 1st ed., Apress, 2022: 45-82.
3. Wang J. "Data Warehouse Snowflake Design and Performance Considerations," in International Journal of Creative Research Thoughts (IJCRT), 2015;3: 123-130.
4. Mittal S. "Snowflake: SCD2 with Dynamic Tables," in Cloud yard, 2023.
5. Ananth J. "Dynamic Table in Snowflake: Implementing Type 2 Slowly Changing Dimensions (SCD) with Flexibility and Efficiency," in Data Inevitable, 2023.
6. Aravindh AMK. "Exploring Dynamic Tables in Snowflake: A Data Engineer's Perspective," in Medium, 2023.
7. "Dynamic Tables for Data Vault," in Snowflake Blog, 2022.
8. "Snowflake Dynamic Tables for Continuous Data Pipelines," in Snowflake Blog, 2022.
9. "Streaming Data Pipelines: An Introduction to Snowflake's Dynamic Tables," in Zimetrics, 2023.
10. "Building Continuous Data Pipelines with Snowflake's Dynamic Tables," in Greenleaf Group, 2023.