Dynamic Tables

Sujith Nair
Cloud Data Architect
Snowflake Snowpro Certified

Learn2Cloud Data Solutions

#What are Dynamic Tables ?

Dynamic tables are a Hybrid between a table and view.

View has a query associated with it and retrieves data from tables based on the view. There can be performance issues with the view as it may need to join tables and return data. However we get the latest data without any latency.

A table stores data physically in it and when we select data from a table the retrieval speed is faster as compared to a view

Dynamic table provide performance of table and latency benefits similar to a view.

We can use Dynamic tables to create ETL pipelines.

#What parameter controls how frequently a dynamic table is refreshed ?

- The parameter TARGET_LAG which is used to set the refresh period for the dynamic table.
- TARGET_LAG can be in seconds, minutes, hours or days.
- TARGET_LAG = '{seconds | minutes | hours| days}'
- Eg: TARGET_LAG ='10 minutes'
- TARGET_LAG = 'DOWNSTREAM' means that Dynamic Table will need to be manually refreshed with ALTER DYNAMIC table <tablename> REFRESH.
- Downstream can also be used when there are several dynamic tables in a pipeline and we want the dynamic tables to refresh based on the TARGET_LAG of the last last dynamic table in the pipeline.

#What are the refresh types supported in Dynamic tables ?

Dynamic table will refresh only if there is a change in the underlying data, this helps reduce unnecessary credit consumption.

Dynamic table support 2 refresh types

- Incremental Refresh
- Full Refresh

Incremental is less expensive in terms of credit consumptions.
Snowflake will attempt incremental refresh and if it cannot go for full refresh.

Certain SQL constructs do not allow snowflake to perform incremental refresh.

- SET Operators
- PIVOT, UNPIVOT
- LATERAL JOIN
- IN-EQUALITY OPERATOR IN OUTER JOIN.

What is the difference between a dynamic table and a view. Why not use a view instead of dynamic tables to create a pipeline ?

View can provide us with the same functionality as Dynamic tables however if we use views we will suffer from

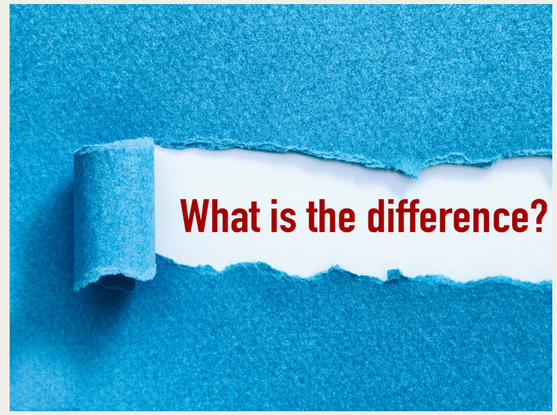
- Poor data selection performance
- High Credit consumption as compared to Dynamic tables..

Dynamic tables are a much better solution to creating pipelines instead of views.

What is the difference between a dynamic table and a Materialized view ?Why not use a it instead of dynamic table ?

- Materialized views are used to improve query performance.
    - External Tables
    - Clustered Materialized views.
- Materialized views have limitations
    - Joins including self joins are not allowed.
    - Window functions are not allowed

Dynamic tables do not suffer from these limitations and can use joins and windows functions and are more suited for use case where we need to create pipelines.



Can you update a Dynamic table using DML?

Dynamic tables do not support DML operations like UPDATE,DELETE or TRUNCATE , they are refreshed when the data in the tables which provide data to the Dynamic table have change in data.

Similar to Materialized view,No DML allowed

Can you explain a scenario in your project where you used Dynamic tables ?

Dynamic tables are considered a replacement for tasks/streams or stored procedures

Project scenario:
We have sales data in SALES_FACT and product data in PRODUCT_DIM. There was a requirement to create an aggregate table which would get product name from PRODUCT_DIM and sales information from SALES_FACT , join the data and aggregate the same on sale_date, product_name, total_sale_qty, total_sale_amt. We had initially planned to use task and stored procedures to read data from the fact and dimension and use merge statement inside the stored procedure to update the target table. we decided to take advantage of Dynamic table and we created Dynamic Aggregate table with a 1 hour Target_lag and were able to meet the business requirement.

What are the limitations of Dynamic tables ?

- Your account can have a maximum of 1000 dynamic tables.
- Your Dynamic table query definition cannot have more than 100 tables.
- Your Dynamic table query definition cannot have more than 10 Dynamic tables.
- A pipeline cannot have more than 10 dynamic tables to create a DAG.
- External functions cannot be used.
- You cannot use CURRENT_USER.
- Cannot use directory tables, external tables, Ice-berg tables streams or Materialized views.
- User defined functions written in SQL.
- Cannot set a cluster key on dynamic table.
- Query acceleration service is not supported on Dynamic tables
- You cannot enable search optimization on dynamic tables.

How can you monitor Dynamic tables ?

To monitor Dynamic tables and locate refresh problems we can use INFORMATION_SCHEMA table functions.

DYNAMIC_TABLE_REFRESH_HISTORY provides the history of refreshes for one or more dynamic tables in the account. To identify the refreshes that had errors, pass the parameter ERROR_ONLY=>TRUE.

```
SELECT
  *
FROM
  TABLE (
    INFORMATION_SCHEMA.DYNAMIC_TABLE_REFRESH_HISTORY (
      NAME_PREFIX => 'SIRIS.PUBLIC.', ERROR_ONLY => TRUE
    )
  )
```

```
  SELECT
  *
FROM
  TABLE (
    INFORMATION_SCHEMA.DYNAMIC_TABLE_REFRESH_HISTORY (
      NAME => 'SIRIS.PUBLIC.DT_ORDERS', ERROR_ONLY => TRUE
    )
  )
```

DYNAMIC_TABLE_GRAPH_HISTORY returns information on all dynamic tables in the current account.
This information includes the dependencies between dynamic tables and on base tables.
A common use is to identify all dynamic tables that are part of a pipeline.
The output also reflects the changes made to the properties of a dynamic table over time. Each row represents a dynamic table and a specific set of properties.

If you change a property of a dynamic table (for example, the target lag), the function produces a new row of output for that updated set of properties.

```
SELECT
    *
FROM TABLE
        (
        INFORMATION_SCHEMA.DYNAMIC_TABLE_GRAPH_HISTORY()
        );
```

Thank you!

Learn2CloudData Solutions