Datameer

HOW TO OPTIMIZE YOUR DATA TRANSFORMATIONS FOR SNOWFLAKE

INTRODUCTION

The first building block of a cloud data stack starts with Snowflake. Your analytics engine and/or cloud data warehouse is always the core component by which your data stack revolves.

The shift to cloud analytics and cloud data warehouses was supposed to simplify and modernize the data stack for analytics. On-premises, your data stack was simple — an ETL tool such as Informatica and a data warehouse such as Teradata. Yet, many cloud journeys have done quite the opposite – the data stack has gotten more complex and expensive. In the end, this drove up data engineering costs.
When building a data stack for Snowflake, customers have a myriad of options. Many new tools in new categories can fulfill a specialized role in your data stack. But this can also create a disintegrated data stack that can increase the overall complexity to create, operationalize, and manage your data pipelines.

Beyond your Snowflake analytics engine at the end of your data stack, there is a myriad of specialty tools that can be inserted into your data stack, including:

- Data ingest and loading
- Data transformation
- Data catalogs and metadata management
- Data observability

However, the plethora of choices for each of these tools can be overwhelming. Using too many tools often creates a disintegrated architecture increasing the overall complexity to create, operationalize, and manage your data pipelines. Operating and developing using this disjointed architecture that drives up your data and analytics engineering costs.

In some cases, these specialty tools may work with one or two tools in an adjacent category but do not work effectively with the rest of the ecosystem. With Snowflake's increasing presence and growth in the market, most products will work well with it. But beyond Snowflake, piecing together the rest of your data stack in a piecemeal manner requires extra integration work and cost.

FOCUS ON THE "T" IN YOUR ELT DATA STACK

If you have been using Snowflake for more than a few months, odds are you have vetted out at least the basics around the extract and load part of the ELT process – getting your data into Snowflake. This might be done with Snowflake's utilities or any of the numerous 3rd party "data loader" tools and services on the market.

If you are loading data from specific operational data sources such as existing databases, SaaS applications, or cloud services, you might have chosen a tool such as Fivetran, Hevo Data, Xplenty, or Stitch. These tools are very good at making it easy to replicate data from SaaS or cloud services sources via their APIs and loading the data into Snowflake.

What these tools are not very good at is the T at the end of your ELT stack. The T, or transformation part, is where this raw data loaded into Snowflake is transformed into a form that is useful for analytics and can be directly consumed by analytics and BI tools.

Data transformation within your ELT stack is where the most time and effort is placed, and oftentimes organizations make this process very manual. The EL tools are a small fraction of where your data and analytics engineering costs lie. It is the T that makes up the vast majority of your data and analytics engineering costs. If you can make the T process and workflow far more efficient and effective, you can dramatically lower your data and analytics engineering costs.

WHO DOES THE "T" FOR SNOWFLAKE?

If you are like most organizations, your data platform has increasingly gotten more complex over the years. Legacy platforms, SaaS applications, and now cloud data warehouses are all part of what many call a “data mesh,” with data spread out in many places. If you’ve done a good job with your EL tools, you have probably managed to pipeline some or most of the data need for your analytics into Snowflake.

As your data platform became more complex, new roles and responsibilities emerged. In the past, data transformation was purely the domain of technical ETL developers. In data lakes, a new role called the data engineer emerged.

The ELT model for cloud data warehouses such as Snowflake still often involves data engineers. But it has allowed the analytics community – data analysts and scientists - to step up and take a bigger role in data transformation. In some cases, a new role has emerged – the analytics engineer.

Each of these personas often has:

- Different skill sets ranging from being highly technical to less technical,
- Different knowledge and understanding of how data is used,
- Different focal points on their role and where they can best use their time.

Data engineers tend to know more about the data itself – where it resides, how it is structured and formatted, and how to get it – and less about how the business uses the data. Their role is highly focused on managing data.

Analytics engineers, data analysts, and data scientists know less about the data itself but have a complete understanding of how the business would use the data and how it would be incorporated into analytics. They may have varying technical skills but would prefer to spend more time on what they are good at - analysis, and less on coding data transformation.

WHAT ARE THE KEYS TO OPTIMIZING MY "T" FOR SNOWFLAKE?

One of the great things about Snowflake, and what sets it apart from many of the traditional data warehouses, is that it is self-optimizing. In other words, the system itself has a deep understanding of the data structures, data, and queries and identifies the best way to optimize execution. No longer do you need to have teams of DBAs that constantly try to figure out ways to optimize the data warehouse performance.

Auto-tuning and optimization directly apply to data transformation in Snowflake. In the past, data modelers would define the final queryable structures in a data warehouse for optimal performance using technical attributes. ETL developers would work hand in hand to optimize how data was transformed and loaded into the highly tuned data structures in the data warehouse. Even small mistakes could be very costly.

With Snowflake providing highly optimized execution under the covers, you don't need highly technical tuning of your data transformation models inside of Snowflake. This allows fewer technical personnel that are not likely deep database experts to get involved with the data transformation and modeling process.

In Datameer, when you create and deploy data transformation models, you create views under the covers in Snowflake. These can be standard views or materialized views. Views have the advantage of making transformation queries more modular and transformation components reusable.

The keys to optimizing data transformation for Snowflake come down to four areas:

- Making effective use of virtual data warehouses
- How you execute your data transformation queries
- Employing specific data transformation techniques
- Having searchable, rich data documentation

Let's explore each of these.

1. Virtual Data Warehouses

Virtual data warehouses are a primary means by which you scale out your data and analytics in Snowflake. Rather than build a monolithic, all-encompassing instance with a star schema as in a traditional data warehouse, one creates separate virtual data warehouses in Snowflake for unique analytics use cases.

Virtual data warehouse instances help optimize the workload for each analytics project. Snowflake recommends you have a different virtual data warehouse for each use case. It also allows for easy, clean security around the data and queries and can act as a cost tracking mechanism for each project or use case (perhaps for chargeback purposes).

You have probably already done this in Snowflake in a simple way. Snowflake recommends that users separate “loading” and “execution” into separate virtual data warehouses, where loading is an area external data is loaded and execution are areas for user query and reporting. This separation is recommended because of the different workloads for data loading and user queries.

Data transformation queries also carry a different workload. And, how you deploy your data transformation queries also has a large impact on workload. As a general rule, you should separate your data transformation views/queries into a separate database away from your raw data. This way, you have the flexibility to use distinct virtual data warehouses for your EL and your T.

Putting your data transformation views and queries into their own database separated from the raw data gives you the flexibility of using separate EL and T virtual data warehouses at execution time. This allows you to optimize those virtual data warehouses to the compute needs of each and minimize your costs.

Views can cross database boundaries. In Snowflake, I can create a view in one database that points to data in another database. By separating the T views into a separate database from the EL database, it gives the flexibility to load the EL database into different virtual data warehouses — one for the EL processing and one for execution time processing – each optimized for their unique compute requirements.

2. Execution Optimization

The next question is: should I put my data transformation views/queries into their own distinct database, or should I place them in my use case database. Which of these options you choose will depend upon other factors such as:

- How complex are the queries/models?
- How much data do the queries/models consume?
- How are the queries executed?

How your data transformation queries are executed also plays a large role in optimizing data transformation in Snowflake. In general, this boils down to a basic decision: do I materialize the data transformation views or run them "at query-time."

Materialized views are another way to improve query performance for workloads composed of common, repeated query patterns. And data transformation models are typically designed specifically for this purpose – to create a dataset that supports a common, repeated query pattern.

Snowflake commonly shares a best practice that one should use materialized views to improve performance for queries that are frequent and repeated. Using storage is always cheaper than using compute. Materialized views will add a small amount to your storage bills but save you much more in your compute bills and query response time. With materialized views, you run the expensive JOINs and cross database queries when the view is materialized and not with each query on the data. Hence, if your models are to support repeatable query patterns, then materialize them.

As mentioned, Snowflake supports the ability for views to reach across database boundaries. This gives the flexibility to place data transformation views where they can be processed best.

One option is to place the data transformation views in the use case database. This option works well when (a) the views are unique to the use case and (b) the views have unique workload or compute requirements. The downside is that it spreads your data transformation views over multiple databases, and in some cases, you end up replicating views that are deployed in multiple use cases.

The other option is to have a single database for all your data transformation views. This option offers flexibility as the database can be loaded into any virtual data warehouse for query execution. This option works well when you have data transformation views that are shared across different use cases and have common workload characteristics.

In practice, you'll find you will mix these two approaches, having commonly shared data transformation views in one database and other use case specific ones in the use case database. The beauty of a tool such as Datameer is that it maintains the complete repository of data transformation models and provides a single plane of glass into these, regardless of where they physically reside in specific databases.

3. Data Transformation Techniques

There are different techniques for optimizing your data transformation views/queries. These will impact the performance. They boil down to four recommendations:

- **Reduce complexity where you can.** Although having reusable components is good from a workflow standpoint, having too many layers in your queries can be detrimental. If you can, combine multiple queries into one. Simpler is always better.
- **Materialize JOINs.** JOINs are very costly operations in cloud data warehouses and will drive up the compute bill. If a data transformation query JOINs multiple tables and the data is not fast-changing (i.e., it is not event data), then materialize the view so that JOINs are performed only at materialization time and not on each query. Storage is far cheaper than compute.
- **Discard data you don't need.** Much of the raw data contains fields/columns not needed in the analysis, especially if the data transformation queries are highly specific to one use case. Perform projection in your data transformation models to eliminate fields you don't need. This reduces query time costs.
- **Use materialized pre-aggregation.** This also goes back to the earlier point on the materialization of repeated query patterns. If there are common aggregation values and dimensions that are often used, then create materialized aggregated views.

4. Searchable, Rich Data Documentation

Having a rich set of data documentation around your data and data transformation models helps provide critical information about how to optimize your data transformation in Snowflake. A related item that is also important is auditing.

Helpful data documentation goes far beyond a simple description derived from comments in the code and technical metadata. It should include:

- Auto-generated documentation
- System-generated properties & metadata
- Data profiling
- Wiki-style descriptions
- Custom properties & attributes
- Tags
- Comments

Each of these can provide important information about how the resulting data from the data transformation queries are used, which will provide critical performance and optimization clues. In particular, data profiling is an important aspect used to determine how best to optimize data transformation queries. A good data profile will help a data transformation designer understand important information to optimize queries, including:

- How wide (number of columns) and deep (number of rows) a dataset is,
- The general size of a dataset,
- What data types are involved,
- How many unique values there are in each column,
- The cardinality of columns to understand aggregation implications, and
- Invalid values in a column.

Also critical is auditing information, in particular: how often data transformation queries are executed, how long they take to run, and how much data is consumed each time.

DATAMEER SAAS DATA TRANSFORMATION

Datameer is a powerful SaaS data transformation platform that runs in Snowflake - your modern, scalable cloud data warehouse - that combines to provide a highly scalable and flexible environment to transform your data into meaningful analytics. With Datameer, you can:

- Allow your non-technical analytics team members to work with your complex data without the need to write code using Datameer's no-code and low-code data transformation interfaces,
- Collaborate amongst technical and non-technical team members to build data models, and the data transformation flows to fulfill these models, each using their skills and knowledge
- Fully enrich analytics datasets to add even more flavor to your analysis using the diverse array of graphical formulas and functions,
- Generate rich documentation and add user-supplied attributes, comments, tags, and more to share searchable knowledge about your data across the entire analytics community,
- Use the catalog-like documentation features to crowd-source your data governance processes for greater data democratization and data literacy,
- Maintain full audit trails of how data is transformed and used by the community to further enable your governance and compliance processes,
- Deploy and execute data transformation models directly in Snowflake to gain the scalability your need over your large volumes of data while keeping compute and storage costs low.

CONCLUSION

Data transformation queries can be among the most complex ones you will run in your Snowflake cloud data warehouse. Turning raw data into valuable datasets for analytics can be expensive operations in Snowflake. Thus, optimize your approach to data transformation and use tools such as Datameer that allows you to maintain a single plane of glass into all your data transformation queries to optimize them effectively.

Are you interested in learning more about Datameer's unique data transformation SaaS platform? **Schedule a personalized demo today!**

ABOUT DATAMEER

Datameer, a leader in data transformation solutions, offers the industry's first collaborative, multi-persona SaaS data transformation platform integrated into Snowflake. The multi-persona product brings together your entire team – data engineers, analytics engineers, analysts, and data scientists – on a single platform to collaboratively transform and model data for faster analytic projects. Datameer is a trusted platform for leading enterprises globally, including Citibank, Royal Bank of Canada, British Telecom, Bank of Montreal, Aetna, Optum, Morningstar, Vivint, and more. To learn more, please visit www.datameer.com.

535 Mission St, Suite 2602
San Francisco, CA 94105 USA

www.datameer.com