

WHITE PAPER

Optimizing Snowflake costs: a practical guide

LOGIC 20/20

✉ solutions@logic2020.com

🌐 www.logic2020.com

Steve Berry, Lead Developer

Kanya Kovalenko, Senior Developer

Mick Wagner, Senior Solutions Architect

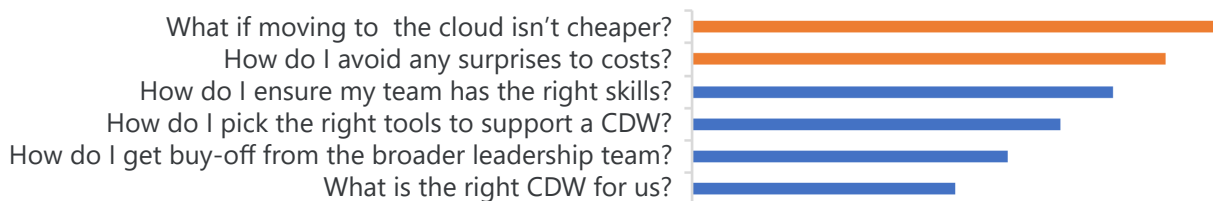
Contents

How Snowflake costs work	4
Snowflake's cost management framework	6
Compute considerations	8
Storage considerations	10
Other considerations	12
Getting started	14



The data and analytics world is going through an exciting time of change and innovation.

With products maturing and starting to capture their true potential value, there is a lot of excitement around and investment in new analytics tools. Most CIOs and CTOs, however, have learned from previous hype cycles and are a bit more cautious. From our conversations with leaders in various industries, including high tech, telecom, and utilities, price elements are often the most common concern:



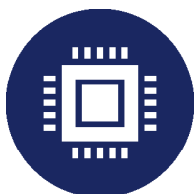
Understanding the costs of Snowflake is vital, as they work very differently from those of traditional databases. With the separation of compute and storage, you are no longer calculating costs based on up-front server costs, server maintenance, and license fees.

Snowflake is a fully managed SaaS data platform with no up-front costs—you are only billed for the resources and services you

consume. Under Snowflake's usage-based cost model, understanding in advance how costs are computed and what guardrails can be put in place can significantly help mitigate billing surprises and runaway costs. At the same time, once you understand certain aspects of Snowflake's pricing and service models, you can uncover additional opportunities to optimize costs while maintaining performance.

How Snowflake costs work

The total cost of using Snowflake is the aggregate of the cost of using compute, storage, and data transfer resources.



Compute represents usage of warehouses, serverless compute (search optimization, Snowpipe), and cloud services (authentication, metadata management, access control). Compute will usually make up the largest part of your bill. These costs are based on the running time and size of your warehouses at the credit cost determined by your Snowflake edition (e.g. \$2/credit for Standard, \$3/credit for Enterprise, etc).



Storage is priced per terabyte per month and varies slightly by region. There is a discount for pre-purchasing capacity storage (e.g. \$23/TB for "capacity" versus \$40/TB for on-demand in the U.S. West region).



Data transfer charges cover movement of data out of Snowflake or across regions or cloud providers. For the most part, data transfer fees on your Snowflake bill are reimbursements to the cloud provider for their data movement charges. Data ingress is always free, but you will be billed when data leaves Snowflake, crosses regional boundaries in the same cloud platform, or moves to another cloud platform.

Once you understand how the various Snowflake services are billed, you can begin planning your migration. Estimation is a fairly straightforward process if you already have well-defined scope and requirements of the process you intend to implement.

But often we only have rough requirements—and that's okay, because you have to start somewhere. The first step should be to review the current data warehouse, including interviewing product owners and technical subject matter experts to define scope and predict workloads. Create "data SLAs" to define service usage requirements, including active work hours,

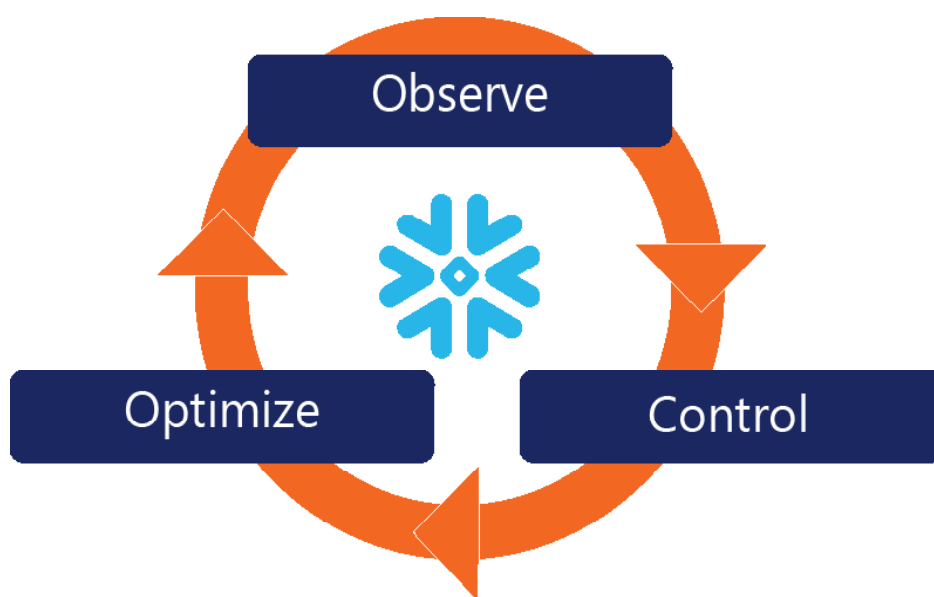
expected workload, and data load or transfer expectations.

The important part is to have a baseline from which to start; then you can experiment and iterate from there. Often skipped during this stage is creation of a budget. Developing a budget based on your estimates can pay dividends downstream for future planning, optimizations, and cost tracking across your organization. It also provides a metric to help you determine quotas for implementing controls. Using a cost calculator (of which there are a multitude online to choose from) is highly recommended.



Snowflake's Cost Management Framework

Snowflake has created a Cost Management Framework that focuses on three areas: visibility, control, and optimization.



Having a well-thought-out estimate and budget in place prepares you to execute in these three areas. Furthermore, the activities encapsulated within these areas should be viewed as an iterative endeavor, not a one-time execution.

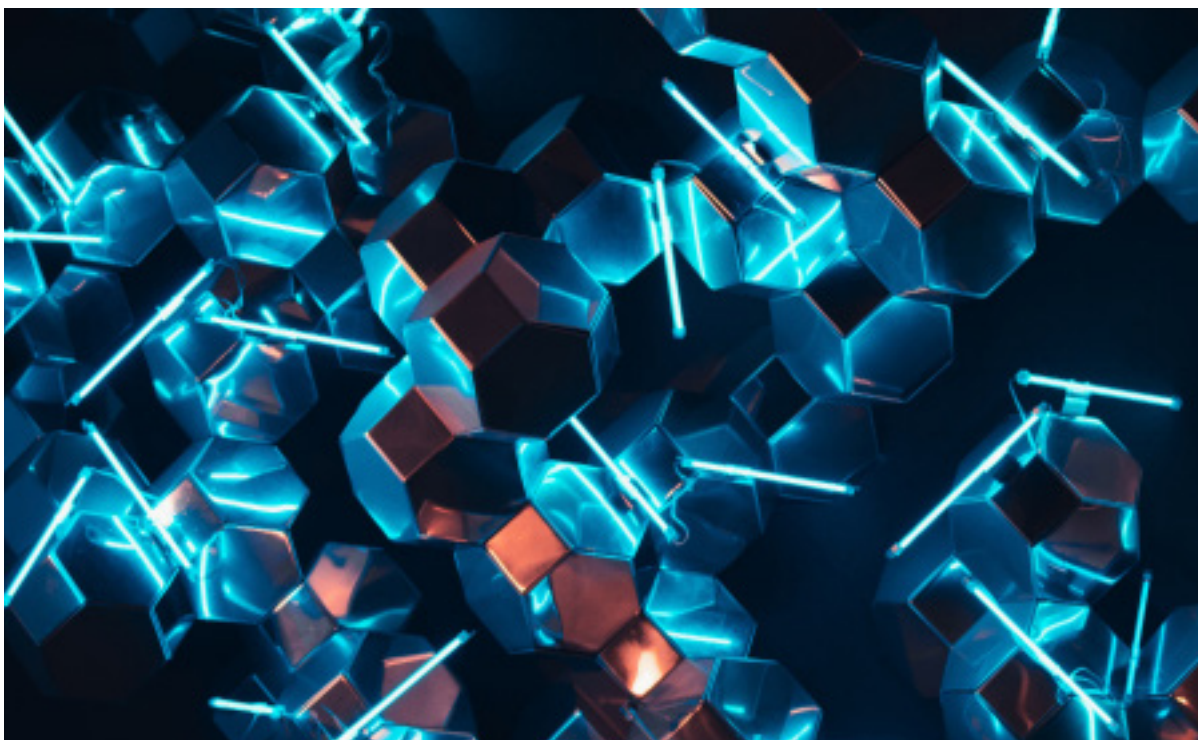
- **Visibility:** Understanding, exploring, monitoring, and attributing spend
- **Control:** Putting guardrails in place to limit spend
- **Optimization:** Identifying and reducing inefficient spend

Visibility and control can be achieved by leveraging built-in Snowflake features and functionality to monitor, alert, and suspend workloads. Resource monitors allow for quotas and alerts to be defined at the account and warehouse levels, allowing for fine-grained quota control mechanisms. Detailed account usage system views are available (in the Snowflake database) that, when tracked over time and compared against budgets and estimates, will allow you to monitor for unusual or excessive usage trends and outliers.

To help you achieve optimization, the system lets you identify performance- and cost-impacting scenarios such as “spilling” (where a warehouse’s available memory might not be sufficient to hold the intermediate results, thus escalating to physical caching) or query queueing (where a warehouse does not have enough resources to run another query).

Visibility into spend can be further enabled by following a few best practices:

- **Embed metadata into each query** using QUERY_TAGS or a long comment block with identifying information. These will be available in detailed execution and billing logs and can support cost attribution for charge-back scenarios.
- **Use and tag different warehouses** for different teams, departments, or projects to allow detailed cost tracking and quota control.
- **Leverage built-in BI partner dashboard views** to help clarify cost generated by external BI tools:
 - Microsoft Power BI: Snowflake usage report
 - Tableau: Compute cost overview
 - Looker: Snowflake cost and usage analysis
 - Sigma: Compute cost, storage cost, and Snowflake cost (reader accounts)
 - Qlik: Snowflake usage dashboard



Compute considerations

Charges related to compute resources make up the single largest part of most monthly bills.

Warehouses are the primary resource for compute charges, although there are other resources (like Snowpipe) that also contribute to compute billing and can vary significantly based on solution architecture and workload.



Size and scale warehouses appropriately

Key to controlling compute costs is correct sizing and configuration of your warehouses. Snowflake provides a wide range of warehouse options, ranging from the one-credit-an-hour XS to the 512-credits-an-hour 6XL. Correct sizing and scaling are largely dependent on the workload applied via the queries executed and size and complexity of data read from storage and returned as results. Determining these factors is more often than not one of trial and error, best aided by experience with the data sets and use cases.

Beyond having the minimum compute necessary to execute your workloads, scaling up (by increasing warehouse size to improve performance) or scaling out (by creating multi-cluster warehouses to enable increased concurrency and users) can be characterized as trading speed for cost. However, since you only pay for the compute you consume, adding warehouse power to deliver faster results can still result in roughly the same overall costs.



Customize auto-suspend and auto-resume values

Another crucial consideration in correctly configuring a warehouse is setting optimal auto-suspend and auto-resume values. The default auto-suspend value of 300 seconds is probably too high of a starting point for many workloads. The minimum configurable value is 60 seconds, but don't be tempted to just set this the minimum value and call it good. Keep in mind that you're billed a minimum of 1 minute each time a warehouse starts, and after that minute, the warehouse credit consumption is calculated by the second.

So, if you have small time gaps between queries in a process, it may not make sense to have a low auto-suspend value (less than 3 minutes), else your warehouse will constantly be in a starting-and-stopping state. However, warehouses serving ad-hoc queries or light reporting loads with infrequent bursts of activity may benefit from the shortest auto-suspend value possible.

Suspending a warehouse also clears its query cache, and using cache instead of storage reads can significantly impact cost and performance. For use cases with very large datasets or where the same data is referenced throughout a pipeline by multiple queries, spending credits to keep a warehouse running may be cheaper than re-querying the data from storage. That said, warehouse startup and shutdown is nearly instantaneous (with no warmup like you get with a Spark cluster or Databricks, etc.), which can add flexibility by adding the ability to design pipelines using differently sized warehouses through different stages, stopping idle warehouses when not in use.

Storage considerations

Although storage costs constitute a much smaller portion of the bill than compute, Snowflake has many features to help control them.



Use storage features strategically

- **Zero-copy cloning** allows one or more copies of the database to incur no additional storage costs until data is modified in the clone.
- **Configurable Time Travel**—from a default of one day up to a maximum of 90 days—lets you lower costs by reducing unneeded Time Travel data retention, since the length of retention impacts storage costs.
- **Transient and temporary objects** contribute to storage charges for as long as the objects exist, and these costs can be lowered by omitting fail-safe and Time Travel storage.
- **External tables (stages)** can make sense in some scenarios with very large data sets that change infrequently, like archival or other seldomly accessed information.
- **Snowflake Secure Data Share** allows you to give external consumers access to your data without the need to create additional copies, thus avoiding additional storage and (potentially) data transfer costs.



Manage data retention and deletion

Often ignored are the costs associated with storing old and unused data. Designing well-thought-out data retention and deletion policies can save costs and mitigate risk of data exposure. Some additional actions to aid storage performance, and therefore reduce storage costs, include establishing clustering keys and pruning plans for partitioning management.

- Clustering keys should be designed based on expected queries to reduce the number of storage partitions needed.
- Pruning plans can help reduce partition sizes as data ages.
- When loading data, pre-sort the data by the columns most frequently used in joins and filters.



Other considerations



Choose Snowflake edition and cloud provider carefully

Choosing the correct Snowflake edition for your planned usage requirements can significantly impact your bill. Some advanced features (like column-level encryption or row-level security) are only available in Enterprise and higher editions. But not all solutions require all features, and Enterprise credits are 50 percent more expensive than Standard (\$3 versus \$2 in the U.S. West region).

It's also important to choose a cloud provider and region based on your overall planned usage and existing and planned infrastructure to avoid excessive data egress and/or transfer costs.



Understand and leverage Snowflake cache types

Snowflake has three types of cache that can be leveraged to reduce costs.

Metadata cache does not require a running warehouse and is used for metadata about various objects (tables, views, staged files, micro partitions, etc.) or events (copy command history).

Local/warehouse cache holds the data that is read from storage and used to execute queries and is purged when warehouse is stopped. The recommendation is to tune the warehouse auto-suspend configuration based on the use case and workload.

Query result cache results from previous queries within 24 hours (with some limitations, including the condition the underlying data has changed). However, running a cached query within the 24-hour window resets the clock for another 24 hours, up to 31 days.



Use materialized views prudently

With materialized views, you are trading compute for storage to improve performance. This would seem to be a win-win where you reduce compute costs and use less-expensive storage. The key to remember is that materialized views automatically refresh when the underlying data changes. Therefore, frequent data changes could drive up costs due to the need to refresh the view. If possible, limit use of materialized views to infrequently updated datasets, frequently queried data, or situations where the underlying query is so expensive that it outweighs the storage costs.



Approach query tuning strategically (if needed)

Traditional query-tuning approaches are generally not needed in Snowflake (for reasonably well-written SQL) and probably won't improve performance or reduce costs much if at all. If it is determined that tuning is needed, start by referring to the query profile and proceed from there. Begin with a small dataset, and the queries you experiment with should be of a size and complexity that you know will typically complete within 5 to 10 minutes.



Reconsider load strategy

When choosing an incremental versus full-load strategy, make sure your choice really is the best fit for the workload. Different approaches can have different costs and benefits in different scenarios, so one approach isn't always best. Sometimes in complex data scenarios, a full load is faster and cheaper than an incremental load.



Additional tips

- Leverage the budget, query tags, and other resource tracking you've put in place to implement a chargeback model.
- Implement Agile software delivery mindset and practices.
- Implement DCM (data change management), DevOps, and CI/CD best practices and automation.

Getting started

These techniques and considerations we have shared just start to scratch the surface of finding cost optimization techniques for your Snowflake Cloud Data Warehouse. It is most important that you give your team the investment of time to plan, analyze, monitor, and optimize your platform to achieve your cost reduction goals.

If you're not sure where to start, the Snowflake team at Logic20/20 would be happy to brainstorm a plan and work with you to find cost reduction opportunities. Feel free to contact us via phone, web, or email, and we look forward to chatting with you.

Contact us



✉ solutions@logic2020.com

🌐 www.logic2020.com

📞 206.576.0400