# Analytics Driven Operations for Self-Storage

---

# Snowflake Query History: What Really Ran

---

# How to Build a Modern Data Warehouse

---

# Why Data Quality Is Essential for Modern Business

---

# Data & Analytics Glossary

---

# 1 Exciting Bit of News about our Pomerol Writeback Extension!

We are very happy to announce that the Pomerol Writeback Extension for Qlik Sense has been accredited into the **Trusted Extension Developer (TED) program.** The TED accreditation insures you a stable, supported, high quality solution.

See our TED page: https://developer.qlik.com/ted/5eb1771845141000012efccc9

## Qlik's TED Program includes:

**Extension Accreditation:** Reduces risk of code failure on upgrade, ensures security protocol and data collection

compliance, and allows accreditation on additional functionality such as responsiveness and mashup readiness.

**Best Practices & Enablement:** Detailed requirements documentation, test script creation guidance, monthly webinars, and open feedback channels like Slack.

**Testing Tools:** Qlik supported automated testing tools, and test script validation with Qlik Sense via a CLI to improve submission quality.

**Assured Upgrades:** TED members sign up with a commitment to provide on-going updates to their accredited extensions, as well as ensuring customers can receive support during their use.

All accredited extensions are listed on **Qlik Market**. Nevertheless, you will obtain the subscription and the support directly from Pomerol Partners.

# What is the Pomerol Writeback Extension?

Our integration for Qlik Sense allows you to edit and enter data directly into your published applications. See the effects of your changes instantly. Reduce spreadsheet risk. Avoid task switching and complex data consolidations. Help your team by giving them a tool to increase productivity, and ensure your business makes the best decisions based on the right data.

The Writeback extension helps enterprise organizations save time, and money by:

- Improving collaboration across your teams.
- Improving the speed and efficiency of your data cycle.
- Improving the quality and integrity of your data.

## See Pomerol’s Writeback extension in action!

---

# A Forecast Is Your Average Parallel Universe

---

# Proven Seven (7) Machine Learning Project Pitfalls

## 1. Poorly Scoped/Designed

There’s more to life than time series. Time series forecasts are inherently bad at picking up the impacts of other dimensions. We get better results by clustering then running a time series. Or, better yet, run a 1X, 2X, 5X, 20X demand forecast. X can be hours, days, weeks, etc. The impact of holidays, product type, etc will be much more insightful; plus we’ll be able to forecast with less historical data.

What’s worse than picking the wrong type of model? Trying to do too much with a single model AND trying to do too much, period. Again, there is no one-size-fits-all approach. Trying to forecast apples and cats, you better have different models.

Use cluster analysis to group like things first. Trying to forecast sales of every SKU? That's also a waste of resources. Find the ones with the most impact first -revenue, losses, spoilage, inventory, turnover, etc. Ask yourself why you're forecasting that product that counts for 0.001% of revenue.

## 2. Unrealistic Expectations

How accurate is your forecast now? Do you think you're going to go from 80% accuracy to 99%? You're probably not. Figure out what it really means to have a more accurate forecast. And figure out what a percentage increase in accuracy means to the company. It's always good to be more accurate but some people think that if they don't get 99% accuracy then it's not worth it; that's almost never true. Figure out what your target level of improvement is and shoot for that.

It is important to gauge the current state of your organization's data capabilities before planning your machine learning project, too. Machine learning does not happen overnight and sometimes it is better to tackle data preparation and BI tasks before beginning any machine learning projects.

## 3. Bad Data/Lack of Data Prep

Every company has bad data. Every. Single. One. Fix it. Because the only thing worse than no data is bad data. Missing values. Dirty data. Inconsistent data. These are all problems that we can address but the wrong time to address these problems is when the model is being built. It ties back to expectations; if you give me dirty data then I will give you bad results. There are loads of tools we use to prep data – some you may have and some you may have never heard of. Tools such as Trifacta can be used to clean and transform data, as well as more traditional technologies such as SQL views and python can be used too.

## 4. Not Enough Data

Are you forecasting water sales but you don't have temperature data? Are you forecasting capital equipment but you don't have opportunity data? Are you forecasting preventative maintenance but you haven't had any failures? We need to find a "signal" in the noise. If you can't give us enough data there might not be a signal. This is why it's so important to have an understanding of the business: to make sure a potential data set isn't being overlooked.

Referring back to the notion of bad data, not including data because it is dirty is not a viable adjustment in many cases. Deleted data can have a drastic impact on a model and in many cases just needs some additional cleaning.

## 5. Too Much Success – Can't Keep Up With Demand

So, you've created a model that very accurately predicts X. Now what? People keep asking you for more. And more. There's no way to keep up with the requests from the "business" side of the house. Unless you have unlimited time, unlimited budget, or, the right tools. We've found that using a tool like DataRobot can help you build lots of models quickly and keep up with that demand. And you don't need to hire more and more data scientists to do so.

## 6. Never Make It Into Production

If it's not repeatable you won't get results. One-off predictions will help you understand you business but they won't help your people do their job better. What you need is a system that integrates your data, makes predictions, and, gets the information into the hands of the people that can take action on it. Imagine if Netflix emailed you once a month with suggested content rather than showing you on-screen what you

might like. Also imagine if your doctor had to wait a week to tell you that you were about to have a heart attack. Designing the model and determining how it can go into near real-time production is how you truly build value.

## 7. Not Monitored

You've got the right model. Into the hand of the right people. At the right time. Now your accuracy drifts. What you thought was 92% accurate is now 80% accurate. How do you know? Without a system in place to tell you that things your model is drifting, there is no way for you to know that. Platforms exist to determine model accuracy faster than a data scientist could ever dream of.

---

# Driving Agile Wins in Manufacturing

---

# Transforming Airport Operations with Advanced Data Warehousing

---

# Efficient Data for Shotshell Reloader Production

---

# Power Maximizing Snowflake Performance for BI Products

The business world is rushing as technology and innovation enter different areas. Any organization that wants to remain competitive and stand out to customers must make real-time decisions. Of course, these decisions must be supported by the judgment of business experts and backed by data.

If our experts and managers can access relevant and up-to-date information using a data strategy, it can yield incredible results for growing the business. This is where business intelligence has its first impact, managing to describe the causes and results of current actions, as well as providing options to take different paths in the near future. It considers historical data while making predictions or simply analyzing trends.

Every data strategy, from a technical perspective, involves different components, starting with the sources of information and the ingestion of these data from transactional systems into a data repository or data warehouse with analytical features. Once the data is stored in the data warehouse, the data cleansing, transformation, and enrichment processes come. Then, it is followed by correct data modeling so final users can use their visualization tools to display dashboards with strategic information, KPIs, and trends.  That provides inputs

to decision-making. Data governance processes and business objectives must accompany all these technical elements. If one part does not complement the other, the data strategy would be incomplete and inefficient.

The technical heart of the data strategy is the data warehouse. So, let’s take some time to expand on the concepts related to this and its optimization to optimize the queries that business intelligence analysts perform.

A data warehouse (DW or DWH), also known as an enterprise data warehouse (EDW), is a system for reporting and analyzing structured and semi-structured data from multiple data sources.

This is where our data is located and where we connect our visualization tools. The figure above shows that the data warehouse must have several layers. However, I will not explain in this article the different layers and components that must be included in the data warehouse. I will proceed with the topic of improving performance when consuming data.

To improve the performance of data queries, we must analyze three essential columns:

1. How we have stored our data

2. The human part.

3. The capacity or computing power to process the queries

## Storage

Knowing how our data is physically stored and how our database engine retrieves is a starting point for any data project. Remember that in a very general way, we have two prominent types of structured databases: transactional and analytical.

Transactional databases focus on high performance for

recurring insertions, changes, or deletions of small transactions, such as points of sale, call centers that contact different clients every minute and need to record data for each one, customer service, inventories, and any CRM and ERP. They are speedy with individual transactions. On the other hand, analytical engines focus on high-performance insertions or data queries of millions of records in minutes or seconds in batch processes. Neither is better than the other. They are focused on different use cases.

Snowflake is currently working on other storage formats to allow transactions; however, we want to analyze its characteristics as an analytical engine.

Physical data storage within Snowflake is based on the concept of micro partitions. A table logically represents data comprising one or more micro partitions. Each micro partition is a group of records ordered either by natural data ingestion or by an established cluster key that allows continuous data reordering according to the fields that make up the cluster key.

Considering this, our first step in optimization is to review the data ingestion processes and validate the order of the records that arrive through files or integrations. If this order corresponds to one of our query keys, we are already on the way to good optimization. If there is no key field to order the ingestion, we have two alternatives: the first and most efficient in terms of the long term would be to modify our processes for creating files that will be ingested into Snowflake. If this is not possible, we have the second alternative, creating a Cluster Key manually to indicate the order the data should have when physically stored.

This solves a use case for querying our data, preferably the most frequently used case. But what about users who query the data using another set of columns as key fields?

One feature of Snowflake is materialized views. You can create different materialized views, each with a distinct cluster key. The magic is that you can query them individually or the original table all the time, and the Snowflake query optimizer recognizes the key structures and clusters of the materialized views. Suppose the service detects that using the materialized view instead of the table is more efficient. In that case, it automatically rewrites the query and retrieves the data using the materialized view. This works with large tables in terms of Terabytes. In small tables, the data will constantly be retrieved directly from the object indicated in the SQL statement since the optimization is null or minimal.

When working on the data structure analysis process, we can identify other objects that can be very useful depending on the specific use case. Sometimes, we can rely on zero-copy cloning to have millisecond copies of the data in a table that we can use for testing or QA. In other cases, we can have dynamic tables in our intermediate layer or our data model layer that do not require customized update processes since they are designed to perform this task through the script that creates them, with the additional advantage over materialized views that they do allow having more than one table as data sources. Here are several links to Snowflake documentation to learn about the different types of objects that will enable you to store data.

Micro-partitions
https://docs.snowflake.com/en/user-guide/tables-clustering-micropartitions

Cluster Keys
https://docs.snowflake.com/en/user-guide/tables-clustering-keys

Materialized Views
https://docs.snowflake.com/en/user-guide/views-materialized

Dynamic Tables https://docs.snowflake.com/en/user-guide/dynamic-tables-best-practices

Zero-Copy cloning https://community.snowflake.com/s/article/Best-Practices-for-Agile-Data-Deployment-using-Snowflake-Cloning-and-Access-Controls

Optimizing storage for performance: https://docs.snowflake.com/en/user-guide/performance-query-storage

## Human part

From a human perspective, we will always be involved in some part of the process, whether at the beginning, designing the architecture of the entire process or writing code to carry out the whole data flow.

The first step is crucial: having a design on the general architecture of our solution so as not to go working and resolving issues that could be avoided, as well as thinking about integration with the entire data ecosystem, with other data products and considering all the points to manage the project within a framework that enables data governance. This part is developed by humans, where the experience and knowledge of data platforms and the tools used can provide a competitive advantage and improve implementation time.

Many warehouse implementation projects do not consider the design and architecture part, and although they are not destined to fail, they will have to overcome many challenges, which will increase the time and resources invested.

Another strategic point is the data architecture itself.

The data architecture within the platform must have at least three layers, starting with the RAW layer, also known as the

STAGE layer, which covers the raw data ingested from our data sources.

The second layer is where we move our information by performing transformation and cleaning processes. We can combine various data sources and enrich the data. This is part of the ELT paradigm, where the transformations are performed directly in the data warehouse, not in the application that performs the ingestion.

Finally, we have our third layer, where we store the data in a form that tools and end users can consume. This is where we perform the modeling of the different data marts. Whether the data modeling method is Kimball, Data Vault, or another does not matter. The important thing is to have a layer where this activity is developed.

For security purposes, any data visualization or analysis tool and business analysts should connect to this final layer. Data scientists can often connect to the first layers because they need to run different models and experiments. However, any model that wants to be put into production must be connected to the data in the final, consumption, or modeling layers.

Another aspect of performance is the way the code is written. Programmers are used to using cursors; however, in many cases, cursors can be replaced with SQL code using grouping and conditional options. Cursors are effective for performing single transactions; an analytical data warehouse can process millions of records if the query is implemented correctly in a quarter of the time or less than that of a cursor.

Another common practice is including many subqueries levels within a query; this can make the query slow. Subqueries should not have order by.

No functions that transform data should be used in the where statements. Or at least we should avoid using them unless we have no alternative. A straightforward example is that we have

a table with 20 million records, and we want to retrieve the records from October 1, 2024. The table has 25 micro partitions to complete the example, and the order_date is part of the cluster key. These two queries work:

```
Select id, order_date

from tabla1

where ‘2024-10-01’ = TO_VARCHAR(order_date, ‘yyyy-mm-dd’);
```

```
Select id, order_date_time

from tabla1

where TO_DATE(‘2024-10-01’) = order_date;
```

The difference is the response time. In the first example, for all records, 20 million, the date conversion is performed to determine which records should be displayed. The function is executed 20 million times, and the entire table is scanned. The function applied to the column does not allow the use of metadata from the micro partitions or the use of the Snowflake optimizer.

In the second example, the function is executed only once. The converted data is compared with the native value of the column. Additionally, it only scans a part of the table, only the micro partitions that have a date within the minimum and maximum of that column, which could be 5 instead of 25. This second query can take a quarter of the time or less than the first.

We can significantly reduce response time by applying best practices throughout our code.

## Compute Power

I have left the processing power part for last because it is the most expensive. However, it is the easiest to configure and scale. For this reason, it is usually the first thing worked on to have short-term results but greatly increasing costs. A good performance analysis and its respective application must contemplate the previous stages first and lastly focus on the processing resource.

Snowflake offers different options. The first and easiest is to increase the size of our virtual warehouse, which comes in sizes XS, S, Medium, Large, and XL up to 6XL. Each size doubles the amount of resources but simultaneously doubles the cost.

If we have complex queries, many transformations, multiple joins, or a large amount of data recovered from a few queries, increasing the size is typically enough. However, having a lot of concurrency, whether it is many simultaneous users or many queries running simultaneously, could not be resolved by increasing the size. In this case, we must work on the multi-clusters instead of the size to opt for better results.

The concept of multi-clustering is to have a minimum number of servers running at a specific size that can increase as there is a demand for resources caused by more users or more queries, up to a maximum number of configured virtual warehouses.

Here, we can apply two types of policies: one that focuses on saving money (economy), where a new cluster is activated only if there are several queues, or the policy that optimizes time (standard), where queries are attended to but if any remain in the queue, a new cluster is activated as quickly as possible.

The number of queries a virtual warehouse can handle concurrently depends on its size and the

“MAX_CONCURRENCY_LEVEL” setting. By default, a single-cluster warehouse can handle around eight concurrent queries, as this is the standard value for MAX_CONCURRENCY_LEVEL; larger warehouses can handle more concurrent queries due to increased computing power. Remember that the complexity of each query also affects how many can run concurrently.

The magic of multi-clustering is that you set up a minimum and maximum number of clusters, allowing you to control your budget. Snowflake automatically activates or deactivates them based on the increase in queries during your platform’s peak times of use.

This configuration is simple, but you should take the time to analyze your normal workload and during peak hours to determine the appropriate size and number of clusters required to handle concurrency peaks.

Other services within Snowflake, in combination with the warehouses, include the search optimization service, which allows you to accelerate queries to a table that does not have a specific query pattern. Some users perform queries using the cluster key, but many do not have a pattern. They can search by any column and retrieve small or medium amounts of records with specific queries. This service can significantly improve performance and the time required to return results.

The QAS, or Query Acceleration Service, is an option that you can activate in the warehouse settings. It can improve the speed of query execution by accelerating parts of the query workload and reducing the impact of outlier queries.

By default, it is disabled. Before activating it, it is necessary to analyze the queries that the warehouse regularly processes to take three possible options: the first is that it is not needed to activate it, the second is that the queries that we have in that workload are candidates to improve performance with this service or the third is that we must

separate the warehouse into two workloads, that is, into two different warehouses, where one will have the queries that can be improved with this service and the other will resolve all the other types of queries.

Mult-cluster warehouses https://docs.snowflake.com/en/user-guide/warehouses-multicluster

Query Acceleration Service https://docs.snowflake.com/en/user-guide/query-acceleration-service

Search Optimization Service (https://docs.snowflake.com/en/user-guide/search-optimization-service https://docs.snowflake.com/en/user-guide/search-optimization/text-queries )

Optimizing warehouses for performance: https://docs.snowflake.com/en/user-guide/performance-query-warehouse

I hope I have given you an overview of the options Snowflake offers. Each has different impacts and improvements on our performance and, at the same time, our costs. However, an essential point is always that if our users and end customers are happy with the performance and use of our data warehouse, it means that we are moving in the right direction. If this is not the case, it is time to analyze why they are unhappy and whether this is related to our platform not responding in the expected time or because our data products take too long to reach the market.

You can use the official Snowflake documentation to guide you through these processes or contact us. We will happily assess your Snowflake environment and optimize it for performance and costs.

Victor Gomez

Principal Data Consultant

victor.gomez@pomerolpartners.com

---

# Hunt Makes Data Visible to Everyone

---

# Consolidating Financials for Multiple Acquisitions