kyvos

WHITEPAPER

All Roads Lead to OLAP Cubes…Eventually

A Modern OLAP Solution on GCP

By Eugene Asahara

Abstract

Analytics solutions should always future proof themselves with mechanisms that ensure scalability. While the current set of Cloud data technologies like GCP might handle current data loads to satisfactorily performant levels, those data loads will surely outgrow those technologies. When that happens, there will be pain from query performance and/or retrofitting for scalability.

Kyvos AI-based smart aggregation is one such mechanism towards ensuring scalability. It is an elegant and comprehensive solution addressing what is still a fundamental function in analytics and data science: Sums and Counts. Even data science staples such as linear regressions and clusters are processed from sums and counts of raw data.

This whitepaper paints the picture of one version of the failure to architect for scalability that I've seen over the past couple of decades, and still very much today.

Table of Contents

Overview 4

The Making of Home-Brewed OLAP 5

The Evolution of the "Data Team" 7

Key Problems Faced by Data Teams 8

Limitations of Aggregation Tables 11

The Kyvos OLAP Solution 16

How to Migrate from Legacy OLAP to Kyvos 18

Closing Thoughts 19

Overview

Compute time and costs are critical measures of a highly performant analytics platform. As the volume and variety of data grow, both measures grow. Increased compute time equates to a slower ability towards discovering strategic patterns. No one wants to wait minutes for a query response. People usually abandon a website if it takes more than a couple of seconds to respond. Compute time is a prime component of costs related to analytics platforms.

Much of this increased time and cost in the wake of the growth of data volume and variety is unnecessary. Despite increased query sophistication around data science, most queries across the enterprise are still from everyday analysts (non- data-scientists) of the slice-and-dice totals form. For example, what are the sales of loaves of bread in the U.S. during 2020? Or how many bottles of sparkling water were shipped from Europe to major cities of California and Oregon in 2021?

The values returned by such queries are the sum of potentially millions of rows of data, perhaps billions or even trillions. How many times could those questions be asked over the years by any number of people for various reasons? A few times? Hundreds of times or thousands? It could potentially be asked daily if it's part of a chart on a widely utilized dashboard.

As I've just described the scenario, an experienced data engineer would have a good, readily available solution for each of the issues. But if we consider more than a few types of events (fact tables) and ways to slice and dice (dimensions), and a larger number of users dependent upon data, a growing infrastructure is required to support scale.

This whitepaper explores how OLAP can resolve analytics challenges in modern enterprises with the story of one such road that leads to OLAP cubes. It's a tale filled with ups and downs, heroics, and ingenuity.

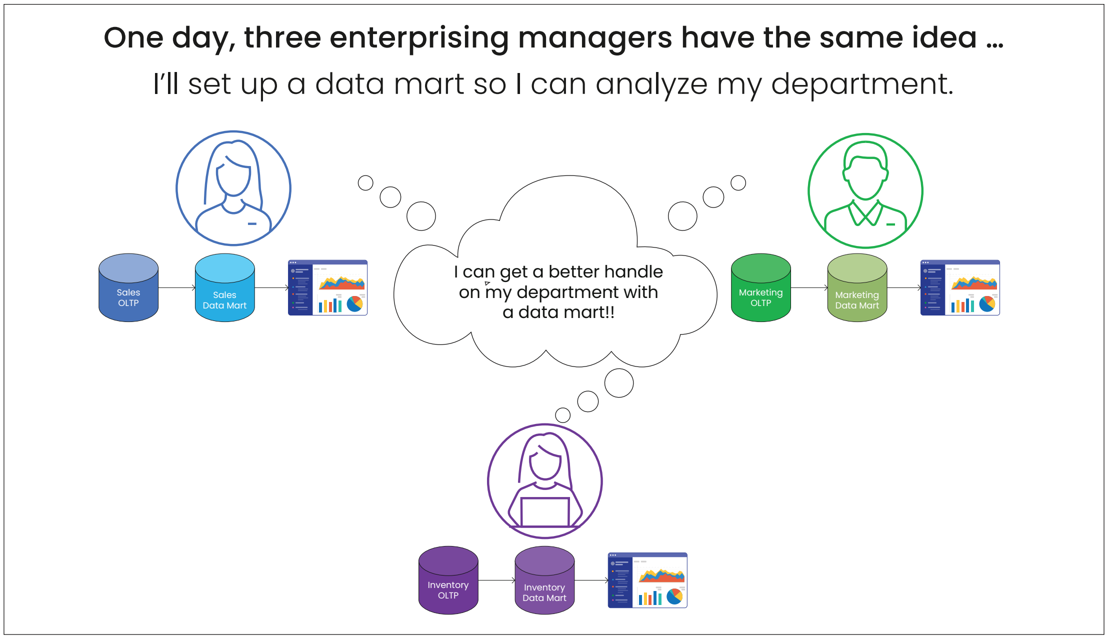
The Making of Home-Brewed OLAP

Our story begins with three enterprising department managers. They are frustrated by their opaque vision of the mechanics and performance of their very own respective departments. They've been waiting for months for the promised magical data analytics platform that will illuminate an enterprise-wide view.

One day, the three managers independently decide to create their own data marts. The project is considered "shadow IT" and is officially discouraged by IT. However, they promise IT to abide by guidelines, particularly security. IT is happy because they can punt analytics for the departments down the road a bit so they can focus on a backlog of high-priority items.

The managers each manage to find a data engineer (between projects or a contractor) and a handful of their own "power analysts" to build the data mart. The narrow scope, lack of dependencies with other departments, and familiarity with the data mean that development is relatively straightforward. At the department level, no one is constrained by the needs of other departments or data engineers. The analysts know the data inside out since they work with it every day.

The data marts and a powerful, easy-to-use visualization tool are delivered to their respective managers in short order.



One day, three enterprising managers have the same idea …

I'll set up a data mart so I can analyze my department.

Figure 1 – Three department managers have the same idea.

After just a couple of hours of exploring their data marts, the managers have insights into the problems they are facing. They notice abrupt changes in various values and can't explain them. But they have an idea of whom to ask. The source of the problems seems to stem from the activities of adjacent departments, which are mostly black boxes to each other.

In a few days' timeframes, they contact each other, sharing insights & asking each other questions about the processes within the departments of their counterparts (Figure 2).

However, the sales manager did run into a little query performance problem. The most important queries that broke down sales by regions, months, and/or product categories sometimes took from 30 seconds to several minutes. The sales manager consulted with the data engineer who built her data warehouse.

It turns out there are over 30 billion sales rows over the course of the past decade. She suggested that perhaps they only needed the past five years – the minimum she would need to analyze seasonal patterns. But even that wouldn't help since the business exploded a few years ago, so 80% of those thirty billion sales rows are within the last five years. That's still a lot of rows!

The data engineer had a trick up his sleeve, though. He would create a materialized view of the sales table that is aggregated to the store, date, and product levels. That reduced the row count from over thirty billion to just over two billion.

> An Aggregation table helps create a summarized view of data aggregated to different attributes, speeding up the analytical performance for larger datasets.

Any user query involving those attributes or higher-level categories would use that table, which should return results in about a tenth of the time. On the minority of occasions where other attributes were used, they could still use the raw sales table. The best of both worlds!

The data engineer called this an "aggregation table". Sure, it took time to create the materialized view and cost a bit to store. But computer time is much cheaper than analyst time, and "storage is cheap". An elegant solution. All is well.

The Evolution of the "Data Team"

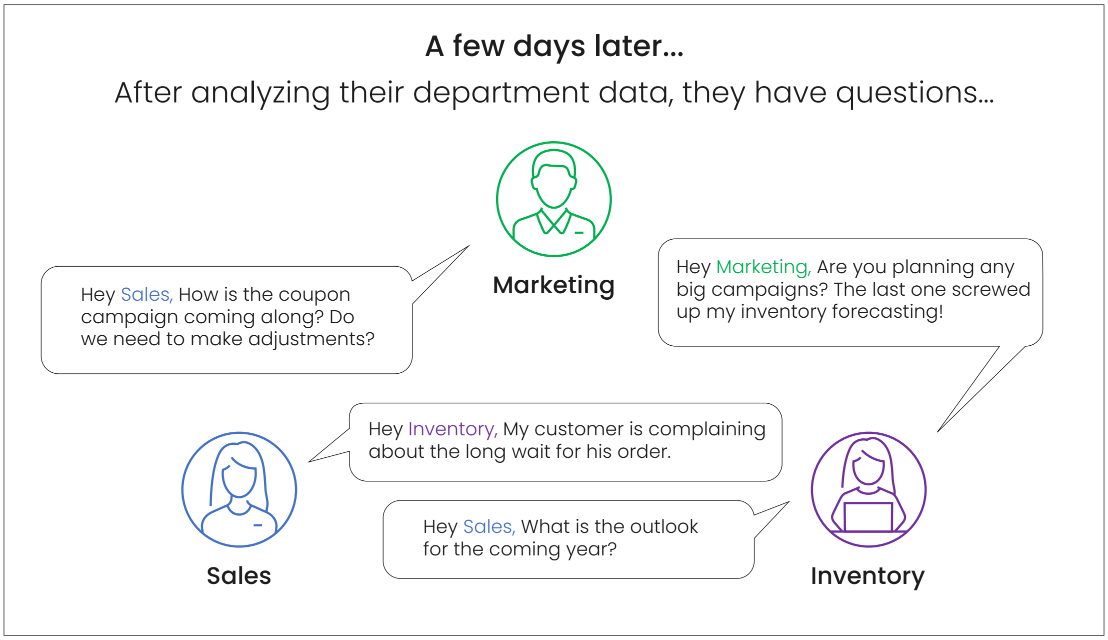

A few days later...
After analyzing their department data, they have questions...
Marketing
Hey Sales, How is the coupon campaign coming along? Do we need to make adjustments?
Hey Marketing, Are you planning any big campaigns? The last one screwed up my inventory forecasting!
Hey Inventory, My customer is complaining about the long wait for his order.
Hey Sales, What is the outlook for the coming year?
Sales
Inventory

Figure 2 – Managers seeking insights into the processes of other departments.

From their conversations, it's evident that insight into each others' departments would benefit all of them.

They decide to build a data warehouse incorporating their three data marts. The data engineers who worked on the individual data marts advised them that there will need to be some mapping of entities across the departments and that this will involve a decent-sized infrastructure.

But it's only three departments and all three managers are in complete alignment. They settle on a commercial data warehouse platform. The platform is based on ubiquitous SQL and prides itself on its performance, scalability, and ease of operations.

They extend the contracts of their collective data engineers for six more months, form them into the "data team", and off they go to build a data warehouse. They are now a lean and mean rogue team.

And the Problems Begin...

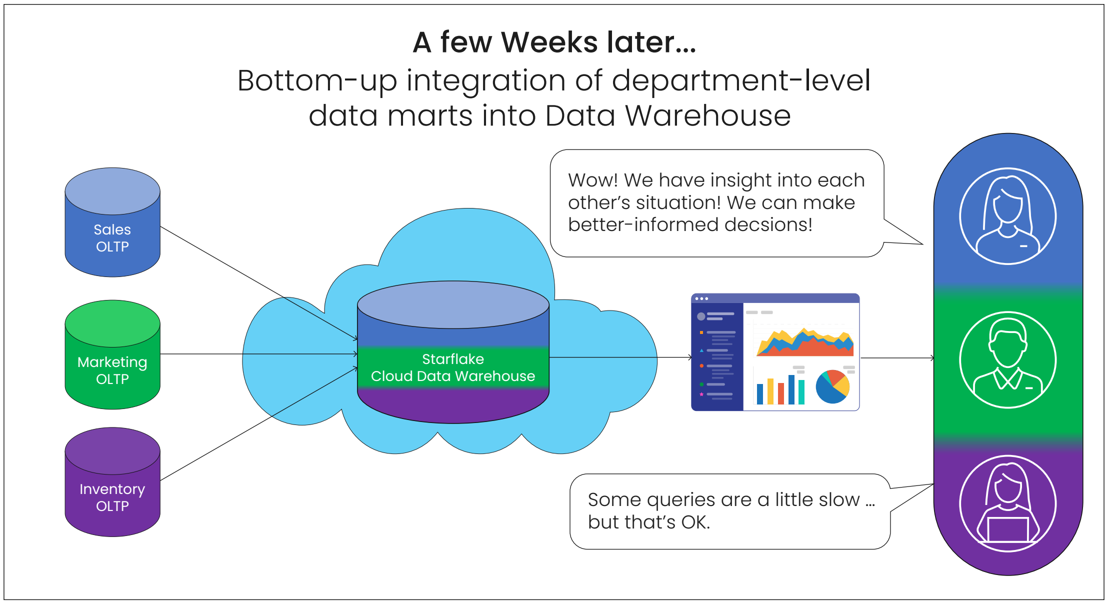

A few Weeks later...
Bottom-up integration of department-level data marts into Data Warehouse
Sales OLTP
Marketing OLTP
Inventory OLTP
Starflake Cloud Data Warehouse
Wow! We have insight into each other's situation! We can make better-informed decsions!
Some queries are a little slow ... but that's OK.

Figure 3 – A cross-department data warehouse is built.

While the gang experienced a rougher time than with their individual data marts, a data warehouse is delivered in a few weeks. The three departments now have insight into each other's activities and can better plan. Inventory has insight into the plans of marketing and sales.

A couple of query time joins over the large fact tables do seem a little slow, but with only a handful of users, it's acceptable. The users still say, "Occasionally slow query performance is better than no information at all. I'll just get a cup of coffee."

Over the next few weeks (Figure 4), utilization of the data warehouse is fully adopted by the departments. No longer are there just a handful of users. Users across the departments working in multiple roles query the data warehouse from different angles. The department managers and their bosses are still happy, but query performance is degrading.

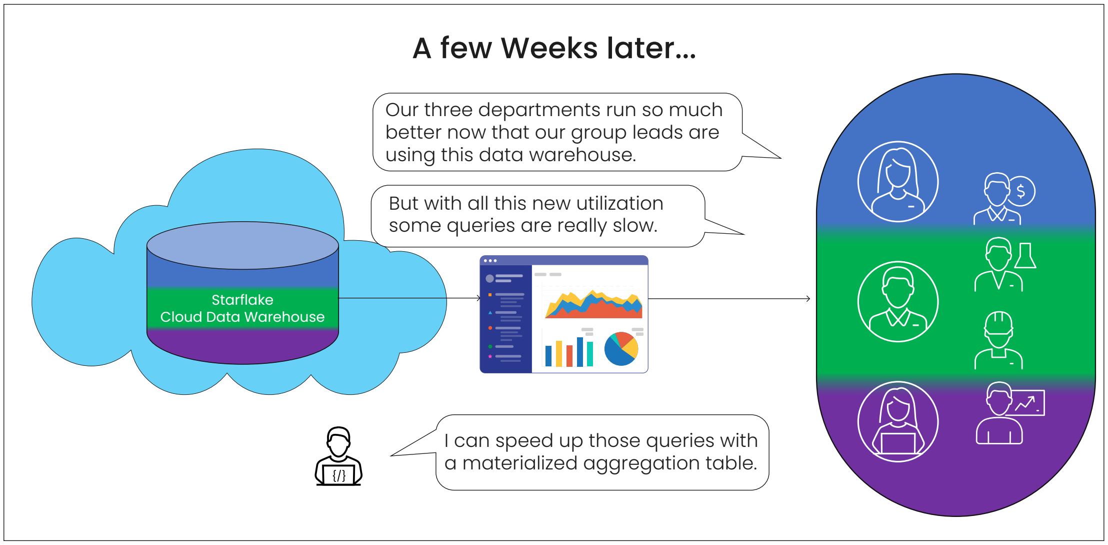

A few Weeks later...
Our three departments run so much better now that our group leads are using this data warehouse.
But with all this new utilization some queries are really slow.
Starflake Cloud Data Warehouse
I can speed up those queries with a materialized aggregation table.
{/}

Figure 4 – The data warehouse achieves full traction throughout the three departments.

The added utilization and higher query concurrency have exacerbated the number of slow queries. Slow queries are no longer occasional and there seems to be a general overall slowdown. “Slow information is better than no information” has shifted to “I can only get so many cups of coffee.”

The sales manager recalls the “aggregation table” her data engineer created to solve a performance problem with her individual data mart. But before she could say anything, that data engineer says, “I’m right on it!”

The marketing manager didn’t encounter this query performance problem since the row count of the biggest table in his data mart is under 200 million rows. The inventory manager did experience performance problems involving her largest table, but her data engineer didn’t offer a solution. So, inventory just lived the slowness before the data mart integration.

The data engineer, the clever one who built the sales data mart, validates that the performance is mostly due to a high number of queries to the very popular and very large integrated sales-inventory table.

To address the problem, he builds another aggregation table, summing sales down to the store, product, salesperson, market campaign, vendor, and day granularity. That cuts the number of rows down a magnitude, from about fifty billion to under a billion rows of summed data. He creates the ETL pipeline to refresh it daily and provides some documentation to the users on the column definitions.

By the end of the day (Figure 5), to the astonishment of everyone, the aggregated table is delivered. The users are trained on when to use this aggregate table over the original raw table. The query times are more than better than before!

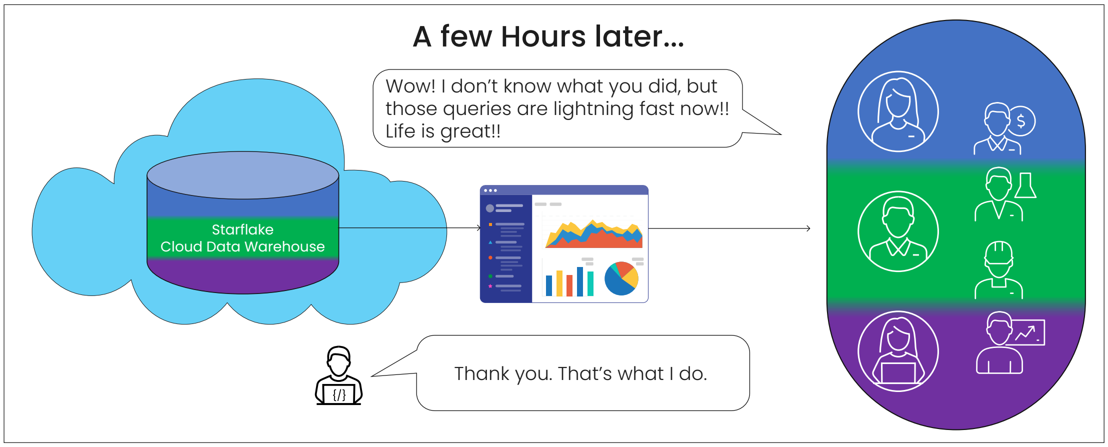

A few Hours later...
Wow! I don't know what you did, but those queries are lightning fast now!! Life is great!!
Starflake
Cloud Data Warehouse
{/}
Thank you. That's what I do.

Figure 5 – An aggregation table resolves the occasionally slow query performance.

Since success breeds more success. Over the next few weeks, the data warehouse grows at a steady, predictable pace. However, word of mouth about this fountain spring of information quickly grew. Requests to join the data warehouse come in from the shipping and customer support departments, along with powerful business justification (Figure 6).

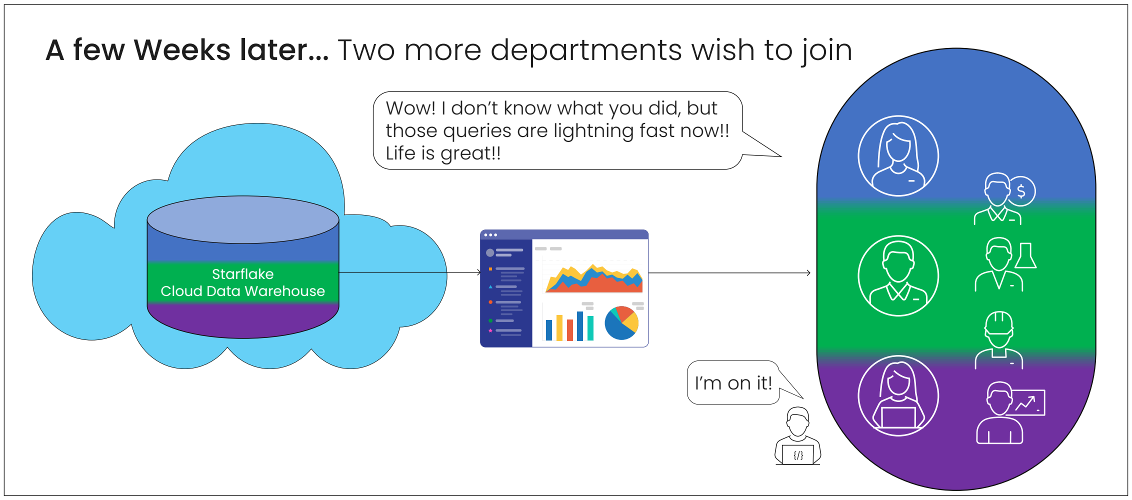

A few Weeks later... Two more departments wish to join
Wow! I don't know what you did, but those queries are lightning fast now!! Life is great!!
Starflake
Cloud Data Warehouse
I'm on it!
{/}

Figure 6 – Two more departments with to be onboarded.

The data engineer team, fresh off their success, begins the onboarding of shipping and customer support, further widening the information view.

Limitations of Aggregation Tables

A good number of weeks go by (Figure 7) and the new departments are fully moved into production. The team thought this would be easier since the onboarding process was already ironed out with the original three departments. However, they underestimated the difficulty of agreement between the five departments on issues around data mapping, business rules, terminology, and process latency.

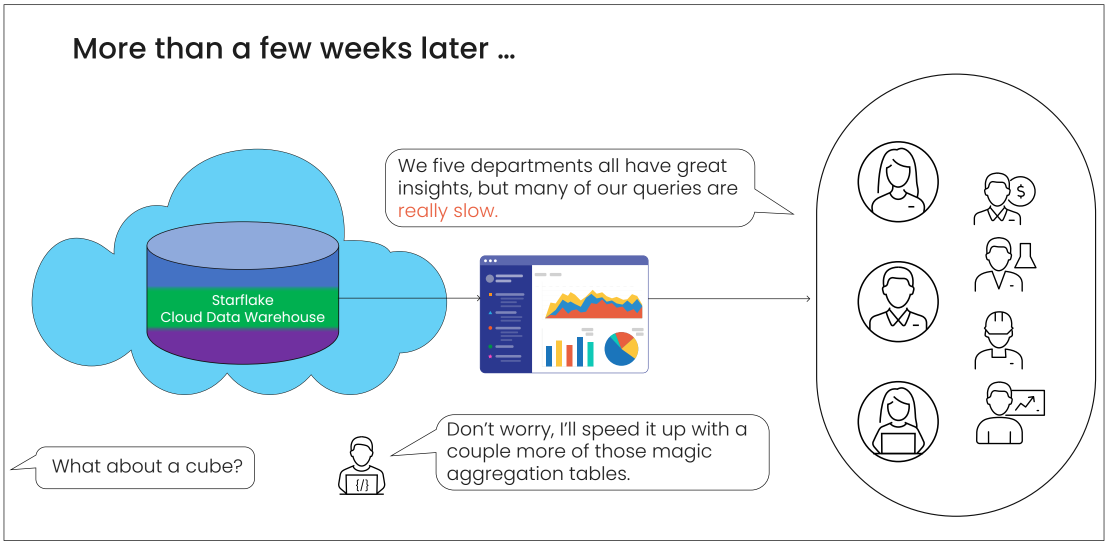

More than a few weeks later ...
We five departments all have great insights, but many of our queries are really slow.
Starflake Cloud Data Warehouse
What about a cube?
{/}
Don't worry, I'll speed it up with a couple more of those magic aggregation tables.
$

Figure 7 – Five departments onboarded, and the data warehouse is beginning to show signs of strain.

Nonetheless, everyone is happy … except for a few points of poor query performance. Not to worry, the data team knows just how to solve that problem – create aggregation tables. Someone way out in the periphery mentions “cubes”. Some folks chuckle, recalling their struggles with “cubes” a decade or two ago. Why involve another moving part when a few aggregation tables will do the trick? Although the data team did resolve the query performance with more aggregation tables, it took a few days, not the few hours it took for the first one. Some aggregation tables seemed to overlap. The redundant processing was eating up time and money.

Notice that the data warehouse is taking on a bit of a muddy look. The five departments brought a lot of data with them – many tables, objects, and relationships. The users are beginning to find navigation among all those fields a bit cumbersome.

Over the next year (Figure 8) a few more adjacent departments join the data warehouse party – SEO, product development, and legal. The data warehouse is a bonafide success. It's been credited with vastly improved productivity and insights leading to the creation of profound new products, marketing strategies, and operational optimizations.

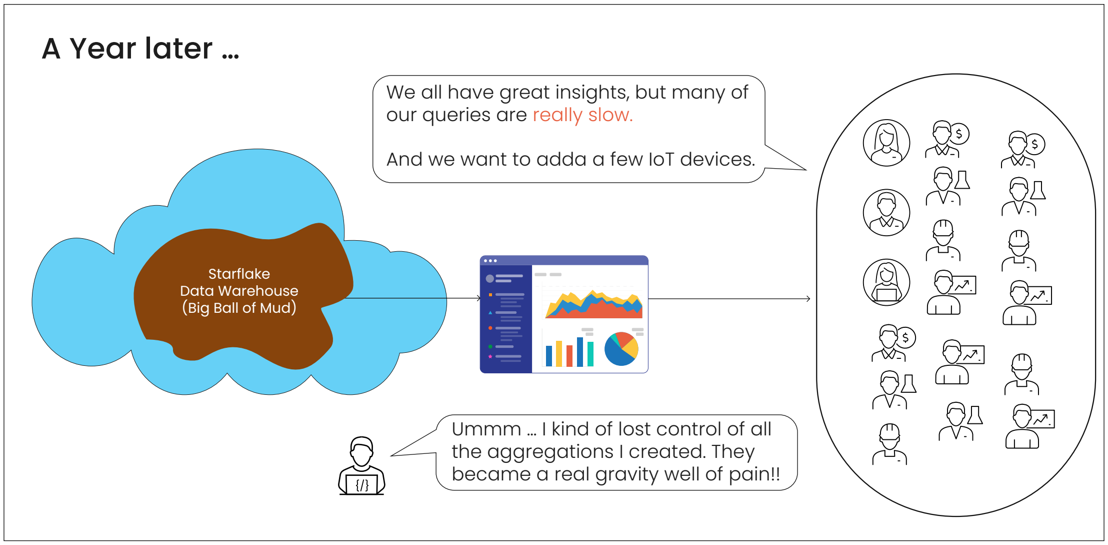

A Year later ...
We all have great insights, but many of our queries are really slow.
And we want to adda a few IoT devices.
Starflake Data Warehouse (Big Ball of Mud)
{/}
Ummm ... I kind of lost control of all the aggregations I created. They became a real gravity well of pain!!

Figure 8 – A year of vast growth.

But it's been a tough year for the data engineering team. Onboarding those new departments accrued significant technical debt through shortcuts such as compromises in the quality of the integration. Often, much work needed to be redone as disagreements emerged only after viewing previews of the data warehouse in progress. That's OK because analytics is supposed to be an activity of learning and discovery.

Sales are up about 20%, due to the rollout of high-volume, low-price products. So, the sales fact table is growing faster than anticipated. However, that growth is nothing compared to the surprise plans announced by some departments. The e-commerce department announces that they will track Web traffic at event levels, not the old higher-level page click level. This will give them a finer-grained look at patterns, and it will balloon their Web traffic by a magnitude or so. Piling on, shipping is adding IoT events from their delivery trucks to constantly fine-tune the routes.

The once sleek data warehouse is now a patchwork of many moving parts. There is a very long list of bugs and requested features. For the moment, that doesn't bother the users too much because the data warehouse still delivers a great deal of value. However, the famed query performance is spotty.

In all fairness, most cloud data warehouse platforms implement some sort of caching mechanism that preserves the results of processing to be leveraged later. The idea of caching is that if we asked for some set of data once, chances are good we'll ask for that same data again. We don't need to move data from storage to compute and number crunch quite as much.

But the problems are:

1. Data not already cached will still need to go through the heavy process of reading the data and crunching those numbers, leading to inconsistent performance.
2. Analytics user behavior is based on business progression and insights that drive toward new business models and data products. What is working and not working in their business? Where are the opportunities for growth? These questions lead to ad hoc analytics where the relevant and useful cache is an erratically moving target.
3. The rules for deciding what to cache are far too cumbersome to manually manage or implemented too simplistically for automated management.

And slow performance just exacerbates irritation with the now not-so-straightforward schema. There are over 50 aggregations serving queries to multiple classes of users with diverse interests. The data engineers estimate that to more than double over the next few months. Each aggregation comes with its own ETL step, eating up compute time.

The line of departments waiting for onboarding doesn't help. IT offers little help beyond provisioning major-league Cloud infrastructure. They are still happy to let these departments handle this themselves as they focus on implementing new ERP, CRM, and SCM systems.

Those departments awaiting onboarding need to wait. The data team focuses on mitigating the current issues with the current client base, particularly query performance. They split the work into an agile set of two-week sprints. More aggregations to come along with tedious retrofitting to alleviate the technical debt.

After a few weeks, there isn't enough progress to satisfy the once-happy user base (Figure 9). Although a few bugs have been fixed, no new features have been added and query performance is still unacceptable. Yet, they are better off than they were before the rogue project.

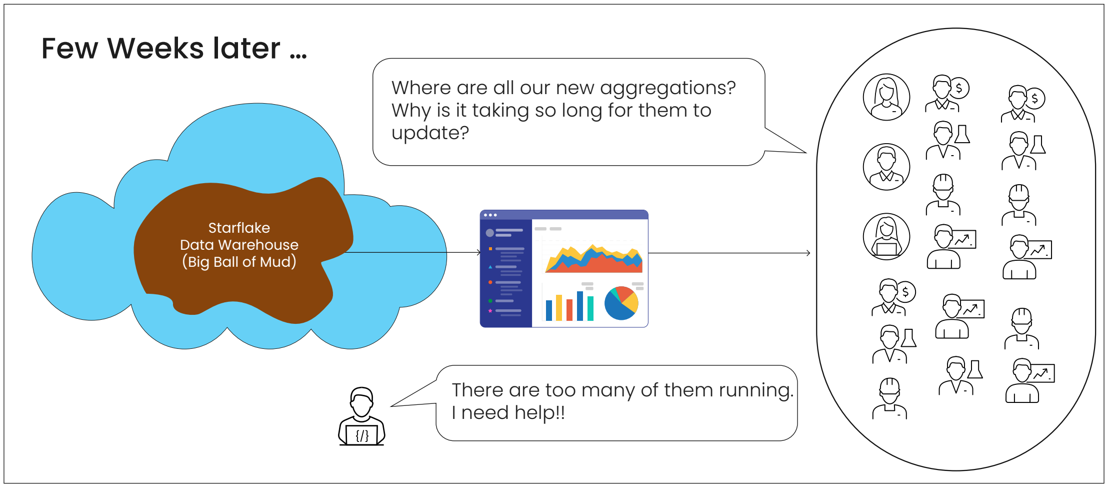

Few Weeks later …
Starflake Data Warehouse (Big Ball of Mud)
Where are all our new aggregations? Why is it taking so long for them to update?
There are too many of them running. I need help!!
{/}

Figure 9 – Improvements are difficult to implement.

The stakeholders set up a meeting with database performance consultants to address the query performance issues that progressively accumulated as the data warehouse grew (Figure 10). The data team fields a litany of questions from the data performance experts.

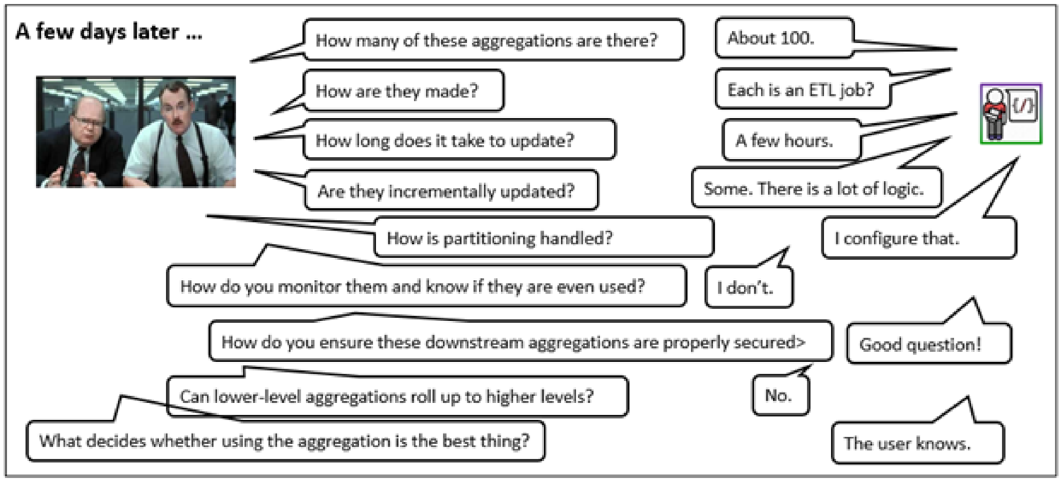
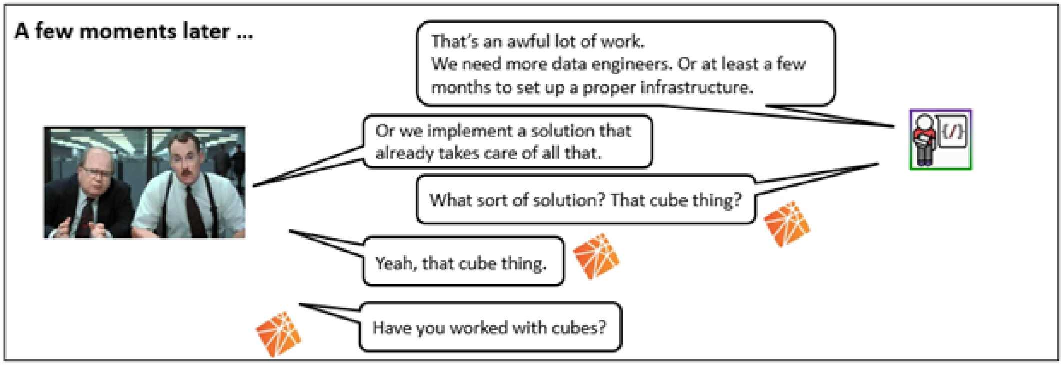

A few days later …
How many of these aggregations are there?
About 100.
How are they made?
Each is an ETL job?
How long does it take to update?
A few hours.
Are they incrementally updated?
Some. There is a lot of logic.
How is partitioning handled?
I configure that.
How do you monitor them and know if they are even used?
I don't.
How do you ensure these downstream aggregations are properly secured>
Good question!
Can lower-level aggregations roll up to higher levels?
No.
What decides whether using the aggregation is the best thing?
The user knows.
{/}

Figure 10 – Trial by database consultants.

By the end of the day-long meeting, the data team internalizes that there is more to aggregation than a few SQL GROUP BY statements. The data warehouse schema is what it is. That is the data we need. However, we could provide a small number of perspectives (dimensional models) targeted for users of various roles. We could generalize the process of managing and optimizing the aggregations.

The data team began to realize that they could build a proper infrastructure for query performance themselves and maintain it as an in-house software product. Or they could seek out a product specializing in this very thing. It's much easier to learn how to operate a product than to maintain, support, and operate a product.


A few moments later ...
That's an awful lot of work.
We need more data engineers. Or at least a few months to set up a proper infrastructure.
Or we implement a solution that already takes care of all that.
What sort of solution? That cube thing?
Yeah, that cube thing.
Have you worked with cubes?

What muddies a "build or buy" dilemma is underestimating the value of what goes into the product we might buy. Since query performance is a major key to success, it makes sense to offload performance to a best-of-breed product like Kyvos.

A data warehouse is relatively easy to evolve and maintain when it involves just a few data sources and a few parties of interest. However, just a few more departments turn a linear process into a web of conflicting definitions, rules, and goals. The data engineer team fell into the "everything is easy at small scales" trap.

The Kyvos OLAP Solution

The solution is OLAP, but not the traditional one

An OLAP cube isn't simply a set of aggregations. It's a user-friendly semantic layer presented to users and optimized in the back end to deliver consistent, maximum performance. The dichotomy of a user-friendly interface and sub-second query response time are two sides of the same coin. Indeed, it's the structure of the user-friendly dimensional model that facilitates data processing. It makes sense when you consider that most business questions are a form of slice and dice mentioned earlier.

What Is OLAP All About?

Today, Online Analytical Processing (OLAP) generally refers to "analytics query patterns". However, OLAP is more than just a query pattern. It is an architecture for accelerating the performance of a data warehouse in the least intrusive manner and presenting the data to end-users as a sensible "multi-dimensional data model". OLAP is the specific optimization of the slice-and-dice analytics pattern.

Kyvos is such a Cloud-scale modern OLAP platform. Its primary goal is drastically to accelerate the query performance of analytics databases as well as presenting the data in a user-friendly manner. At a high level, this is accomplished through the smart aggregation of massive volumes of data.

Such a task sounds simple as long as the scope fits within the manual capability of human engineers. As long as the development and maintenance rate doesn't exceed the number of hours in a workday, there isn't a need for the total cost of ownership (TCO) of yet another product. However, with all things related to data – the explosion of volume and breadth – nothing will remain at that small scale for long.

As with anything that successfully delivers at a large scale, there is an elegant underlying mechanism. OLAP technology isn't as simple as generating aggregation tables. While the implementation of a few aggregations is very straightforward, the full range of functionality isn't replicated with a few SQL GROUP BY materialized views.

Why? The reason is that there are constraints to all systems. In this case, aggregation of everything is impossible. That's in terms of the unimaginable number of combinations for aggregation, the time to process all of that, and the storage. What does "all aggregations" mean? It means all combinations of attributes (features, table columns) – the factorial of the number of attributes.

Imagine a rather small and simple data warehouse with 10 customer attributes, 5 date levels, 10 product attributes, 5 salesperson attributes, and 10 location attributes. That's already about 25,000 possible attribute combinations. A more typical data warehouse would probably sport hundreds of columns. Let's just say that the number of combinations is a really big number.

Pointing out that there could be a big number of aggregations is not endorsing a "combinatorial explosion". Kyvos doesn't "promote" the combinatorial explosion of aggregates. It's about mitigating combinatorial explosion through the smart selection of the most beneficial set that fits within given constraints.

It's for that reason that Kyvos implemented its state-of-the-art smart aggregation mechanism. It determines the best set of aggregations within the constraints of finite computation, time, storage, and cost. The decisions are based on its automatic, ML-based analysis of the structure of the data model as well as the nature of the queries.

Why Is AI-Based Smart Aggregation the Real Deal?

Kyvos' AI-based smart aggregation employs other mechanisms for mitigating redundant aggregations, data compression, and caching.

This task isn't difficult when facing one or two aggregation tables. Typically, an enterprise-class OLAP cube generates hundreds of aggregations. Hundreds of aggregations may sound crazy, but that is a very minuscule number of possible aggregations. Considering the multitude of viewpoints from which we can examine data, many really do make sense. Analytics is a learning problem, it's about discovery.

From the users' side, the most obvious problem resolved through Kyvos is how to select the best aggregation table for a particular query. The raw tables with billions of rows or one of the many aggregation tables? A wide number of attributes makes such a selection not very obvious for someone who isn't a data engineer. Through a Kyvos data model, there is no need to know when one of the aggregation tables would be better, or that the raw table is used for situations without a pre-aggregated table. It just works.

Each of these potentially hundreds of aggregations requires indexes, updates, deletion (if it's no longer needed), partitioning, and caching. For the data engineer team, each aggregation represents an ETL step.

For a Kyvos data model, the processing and optimization of each aggregation amount to just one ETL step – processing. But that one simple step encapsulates much-sophisticated logic. Dimension tables must be processed. A root aggregation table must be created. All aggregations are built on it. There should be an order that maximizes the available resources in order to minimize the start-to-end process time.

Let's not forget mechanisms for connecting to a variety of sources and designing a user-friendly dimensional model. Business rule calculations could be centralized through calculated measures authored with elegant, highly expressive language. Cube Data can be secured in a very versatile manner from a high level down to an individual cell.

Lastly, as a reminder, aggregation doesn't just accelerate the performance of queries computed from massive numbers of rows (billions to trillions). There is also the other scaling "dimension" of high concurrency (larger number of concurrent cube users). Analytics is no longer a matter of a few users computing over a trillion rows. The expanded concurrent utilization of analytics data by a wider breadth of information workers issuing queries involving a "mere" hundred million rows can be served equally well through a Kyvos data model.

How to Migrate from Legacy OLAP to Kyvos

Kyvos is built on a scale-out architecture for cloud-scale volumes. That means, like all cloud-based applications, it is implemented on clusters that scale "indefinitely" through the addition of more commodity servers. It is capable of handling not just a few TB of data and even more.

Kyvos has developed a tool that significantly eases our Professional Services team to migrate from legacy SSAS to Kyvos' AI-powered smart aggeration. The vast majority of the SSAS dimensional model is migratable, taking on the bulk, if not the entire, burden of migration.

Closing Thoughts

Pre-aggregation of sums is a fundamental method for conserving precious resources by mitigating unnecessarily redundant computation. A big hint as to its fundamental status is that it has been rediscovered by thousands of data engineers.

Kyvos AI-based smart aggregation encapsulates that fundamental methodology for delivering data in a highly accelerated and user-friendly fashion. It is developed and continuously enhanced by experts in this field. The adoption of Kyvos modern OLAP eases a significant burden on data engineers.

While it is true that the world is in constant change, the change is at different rates. Most things in business actually don't change very quickly. We usually have lots of time to prepare for paradigm shifts. If we embrace the upcoming change when we first see it, rather than fighting or ignoring it, we have enough time.

OLAP cubes are perceived as rigid, but that's only true from the perspective of schema. From another perspective, given the imperative for robust and accelerated query performance, Kyvos proves to be very versatile with a thousand performance patches transformed into an elegant solution.

About Eugene

Eugene Asahara is currently a Principal Solution Architect at Kyvos Insights.

He worked at Microsoft in various roles from 1998-2000 and 2002-2009. Those roles include time with the Analysis Services development team, Microsoft Research, Microsoft Premier Support, and Microsoft Consulting Services. From 2010 through 2020, he implemented many high-end BI systems as an independent consultant.

About Kyvos

Kyvos is a modern, cloud-native, high-speed data analytics platform that enables sub-second querying on massive datasets. The platform's universal semantic layer democratizes data for all users across the enterprise, enabling self-serve analytics. Its AI-powered smart aggregation technology modernizes advanced analytics, while reducing the time and cost to extract insights. With Kyvos, instantly analyze data at any scale using the visualization tool and underlying cloud platform of your choice.

To learn more, request a demo now.

kyvos

Copyright © 2023 Kyvos Insights. All rights reserved. Kyvos logo is the registered trademark of Kyvos Insights, Incorporated. Product names, logos, brands and other trademarks referred to within this document are the property of their respective trademark holders.

FOR MORE INFORMATION:

www.kyvosinsights.com

info@kyvosinsights.com

@kyvosinsights

@kyvosinsights