American Journal of
Biomedical Science & Research

www.biomedgrid.com
ISSN: 2642-1747

Mini Review

On Cloud Analytical Database Management Systems Suitable for Data Intensive Biomedical Related Research

Genyuan Du[1] and Jie Liu*[2]

[1]*School of Information Engineering, Xuchang University, China*

[2]*Computer Science Division, Western Oregon University, USA*

***Corresponding author:** Jie Liu, Computer Science Division, Western Oregon University, Monmouth, Oregon, USA.

To Cite This Article: *Genyuan Du, Jie Liu, On Cloud Analytical Database Management Systems Suitable for Data Intensive Biomedical Related Research. Am J Biomed Sci & Res. 2021 - 11(4). AJBSR.MS.ID.001652.* DOI: 10.34297/AJBSR.2021.11.001652.

Received: December 12, 2020; **Published:** January 15, 2021

Introduction

Per to Worldometer [1], as of mid-December 2020, there are more than 77 million Covid-19 cases worldwide with more than 1.7 million deaths worldwide. Unfortunately, with the second wave hitting many countries, positive cases are increasing rapidly at astonishing rates. Along with that, a large amount of data, both directly and indirectly related to Covid-19, is generated ready to be minded supporting decision makers, scientists, corporations, drug makers, etc. [2]. Two immediate questions naturally would be (2) where we should store the data and (2) what Database Management Systems (DBMS) we should use to analyze the data? To benefit from easy access and to accommodate sharp growth in data volume, it is hard to dispute that the data should be stored in the cloud. In this position paper, we will present our recommendations based on our more than five years of using different Analytical DBMS (ADBMS, also called data warehouses in many articles), which we will discuss briefly next.

Analytical DBMS

Vertica was the first ADBMS we had access to. Unlike traditional DBMS, such as Oracle 19c or SQL Server 2019, where supporting OLTP operations is their main goal, an ADBMS provides super quick response times for queries returning aggregated known as OLAP queries. According to OmniSci [3], ADBMSs are generally more scalable, distributed, columnar in data stores, and heavily compressing their data. In addition, because the data is distributed, they naturally utilize concurrency as one of the main mechanisms for performance improvement. Vertica, at the time we used it, can be installed on premise or on Amazon's EC2. However, it is not designed to be a cloud based ADBMS, meaning it does not deliver database functionalities as a service. Today, the popular ADBMSs, such as Amazon's Redshift, Google's BigQuery, and a newcomer Snowflake, are all cloud based and easily distributed to achieve scalability and performance improvement with parallelism. Due to space constraints, we do not intend to describe every detail of these ADBMSs, instead, we will describe some theory about parallel processing next then move to the main top of the paper -- showing a suitable system for data intensive operations, including biomedical related research.

Parallelism

We exploit parallelism for performance gain because the speed of light places an upper bound on how much computation a single CPU can perform within a time unit. To break that, we must rely on concurrency, that is, use many CPUs to solve the same problem in parallel [4]. The fact that data in an ADBMS are distributed makes utilizing multiple CPUs a logical choice. Still, parallel processing as a computation model is governed by many laws, especially Amdahl's law and Gustafson's law, both can be proved mathematically. In a nutshell, considering that most parallel algorithms still must deal with some sequential code, Amdahl's law looks at the limit in performance gain on solving the same problem, not even changing the problem size, by adding more processors and concludes that after a certain point, adding more processors would not yield noticeable performance gains. Gustafson's law, on the other hand, points out that for many algorithms, with more processors, we can solve much bigger problems within roughly the same amount of time allocation.

Gustafson's law more realistically reflects the relationship between data processing and how researchers benefit from parallel processing. With readily available more and more faster processors, we are solving bigger and more complex problems; furthermore, the only way to solve many problems in a reasonable amount of time is either to harness the power of better algorithms and/or increase the number of processors. For example, one of our authors' ETL tasks must write more than 1 trillion records from Google's BigQuery to its Cloud Storage. Using the CSV file format would take more than the allowed 30 minutes enforced by Google. The solution is to write the records using AVRO format that supports concurrent writing and Google automatically scales up processors when needed. Clearly, with more and more data, we must employ parallelism to solve our analytical tasks in a reasonable time. The question now should be "what are available out there?".

A Few Business Models for Cloud Based ADBMSs

Here, we will discuss three cloud based ADBMSs with emphasis on their business model and fitting of our data analytical operation's needs. All three ADBMSs support parallel processing and store data compressed in columnar fashion to benefit aggregate queries. Still our experience of using them gives us the opportunity to compare them side by side for real business use cases. It is worth noting that all these three ADBMSs have similar excellent price and performance [5].

A simple extension from Amazon's many data processing services is its Redshift. The business model of Redshift is that its customers purchase nodes to form a cluster. A node comes with a certain amount of storage and computation power. When a customer needs more storage or better computation power, they can double the size of their clusters, which only takes a few minutes. For us, initially, when both the number of users and amount of data were small, this worked great. Whenever ran out of storage space, we just double the number of nodes. This worked for a few years until our system must handle more data for many users. Then, situations where one user executes a computationally intensive query and slows down everyone else's queries become a common problem. Recently, we switched out of Redshift because doubling costs too high and we pay for storage and computation power we do not need until we grow into needing these. Increasing the node count by only a percentage, say 10%, can take many hours. Another reason is that, like many other organizations, we experienced data explosion, one of the main reasons for us to increase the number of nodes was to accommodate our increasing storage needs. Redshift can query data directly from S3, which solves purchasing nodes to alleviate lacking storage problems. With the amount of data we have, it would take too long to fetch data from S3 for our queries. We believe Redshift can provide solutions to solve the lacking elasticity issue soon and surely may consider coming back to them again.

Contrary to Redshift, Google's BigQuery charges little on storage and automatically scales up computation power when the needs arise. That is, BigQuery is a fully managed service and customers do not need to specify the necessary infrastructure running a query. The cost of a query is mostly determined by the amount of data the query processes, and the actual cost can be offset by the credit Google offers. Queries run very fast, still; this pricing policy means an organization can easily spend a lot. When a stable budget is a factor to consider, this pricing policy makes BigQuery not an ideal platform to support many users with unpredictable query patterns, which, ironically, is what analysts tend to do. Google recognizes this and offers many different remedies. However, we are still waiting for one we like.

Snowflake is one perfectly lies between BigQuery and Redshift. Unlike Redshift, with Snowflake customers pay storage and computation services separately. The cost of storage is like other cloud storage services and very inexpensive. Unlike BigQuery, rather than paying the amount of data processed by a query, Snowflake customers pay the computation power used to process their queries in terms of credits. The elasticity of using Snowflake comes with the feather where its customers can size up or down their warehouses, which is used to define the computation power in use, from 1 node to 128 nodes. The specification of a node cannot be found in official Snowflake documents but was reported as EC2 m5d.2xlarge [6]. Customers can also allow up to 10 clusters in a warehouse; however, having more clusters improves overall system performance by allowing multiple queries to run concurrently.

Using of Snowflake

In Snowflake's pricing model, a warehouse's usage is billed on a per-second basis, with a minimum of 60 seconds, times the number of nodes of the warehouse. So, a query running on a 128-nodes 4xLarge that takes one-hour costs 128 credits. The same query may take, say, 2 hours and 3 minutes to run on a 64-nodes 3xLarge and costs 131.2 credits. One of the reasons the query takes more time to run on a smaller warehouse is because it may use more data than the nodes can handle and must move data to S3. Snowflake calls this disk spilling and has tools helping customers to uncover them. So then why not always use the largest warehouse? Per Amdahl's law, for a query takes only a few 100s of milliseconds on a small warehouse, sizing up the warehouse may not future reduce the execution time if at all, but with a higher cost because the minimum charged time is 60 seconds. On the other hand, for queries that take a long time, we should use a large warehouse following Gustafson's law.

Conclusions and Future Works

In this position paper, we introduced the three most popular cloud -based ADBMSs, and based on our anecdote experience, suggested that Snowflake's model is best suited for organizations that want to easily control their budget while utilizing all benefits a cloud-based ADBMS offers. We do not own Snowflake stocks, just want to share our experiences.

References

1. (2020) COVID-19 Coronavirus Pandemic. Worldometers.
2. Bai Y, Yao L, Wei T, Tian F, Jin DY, et al. (2020) Presumed asymptomatic carrier transmission of COVID-19. JAMA 323(14): 1406-1407.
3. https://www.omnisci.com/technical-glossary/analytical-database
4. Cloud data warehouse comparison ebook.
5. George Fraser (2020) 2020 Cloud Data Warehouse Benchmark: Redshift, Snowflake, Presto and BigQuery. Fivetran.
6. https://stackoverflow.com/questions/58973007/what-are-the-specifications-of-a-snowflake-server