North American Journal of Engineering and Research

Est. 2020

Handling Time Series Data: Capabilities & Performance in Snowflake and Databricks

Rameshbabu Lakshmanasamy
Email: ramesh.lakshman@gmail.com

Abstract

Time series data is a set of data points arranged chronologically in a sequence and captured at regular intervals. The high level of variability plays an essential role in a wide range of domains, including finance (stock quotes and trading volumes), IoT devices (sensor data, machinery analytics), and weather forecasting. It allows organizations to observe patterns and trends and draw conclusions over time. Working with time series data involves proper storage and processing of data together with fast or near-real-time analysis capability.

Keywords: Time Series, Snowflake, Databricks, Cloud computing

Introduction

Snowflake and Databricks are two major cloud computing for data that provide rich capabilities for managing and processing time series data. Snowflake is famous for its elasticity of data storage without incurring higher costs (Deshpande & Nanda, 2023). It is also easy to integrate with different technologies, while Databricks is renowned for its single platform that provides analytics based on Apache Spark. This article, focuses on aspects of time series, data management in both Snowflake and Databricks, analyzing their productivity and compatibility, the elements of data storage, calculating capabilities, and the possibility of further expansion. Since time series data processing varies across different platforms, recognizing how each tackles time series workloads will assist organizations in making a standard decision for their analysis concern.

Time Series-Specific Functions and Optimizations

Snowflake:

Below are some functional built-ins in Snowflake that help analyze time series data. They include window functions that allow for the running total, moving average, and generally any statistical computation over a given time frame. The `DATE_TRUNC` enables users to sheer dates to the given intervals they need, like day, month, or year, while grouping data. Another helpful function called `TIMESTAMP_DIFF` assists in determining the number of seconds between two timestamps, which helps evaluate trends in temporal data (Kempter, 2024).

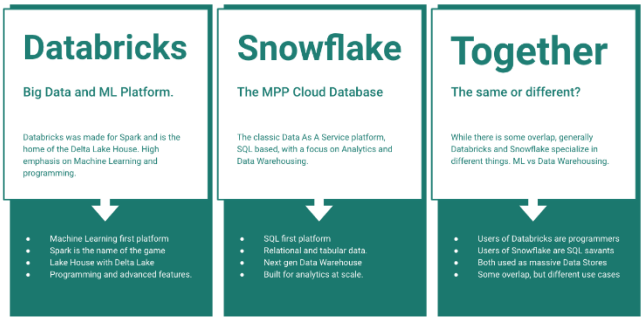
The following are the top measures employed by Snowflake to fine-tune performance on time series workloads: One of those is its superior data compression, the result of which not only decrements storage volume but fine-tunes query response time through the minimal amount of data that needs to be pulled off disk storage (Kukreja & Zburivsky, 2021). Snowflake also provides clustering keys where users can specify which column – for instance, time – columns can be used to guide the physical arrangement of the data. A good clustering only directs the system to look at areas of the data that contain information related to the query to minimize the time taken to access it. Also, Snowflake has a micro-partitioning feature that, by default, splits the tables into partitions, meaning that analysts can easily pull terabytes of time series data. Combined with auto-scaling, these optimizations show that Snowflake can effectively operate high-transaction, time-sensitive loads.

Databricks:

Time series data processing capabilities in Databricks built on Apache Spark are rather powerful. They perform several operations with time-related data: `lag` and `lead` to use data from the previous or next row for computing differences and trends. Other vital functions are those of average on moving, windows and aggregation, which enable an analyst to perform weighted calculations within a given window, perhaps hourly,

daily or weekly. Apache Spark makes These operations scalable due to support associated with distributed computing, facilitating fast analysis of large time series datasets.

Looking at the optimization tactics, Databricks uses several features that can improve time series processing. One of the most important is Delta Lake, which adds ACID transactions to Apache Spark, guaranteeing data correctness. Data versioning and time travel are Delta Lake's capabilities, which enable it to handle historical data, which is most important in time series. Databricks also employs auto-scaling so clusters can scale the resource based on the program's workload (L'Esteve, 2022). This feature helps to process optimally without drawing many resources, thus being less costly. Moreover, Databricks promotes good partitioning practices; it gives options to partition data by time frames like day, month, year, etc. This is important in partitioning as it ensures only the partitions containing the data needed for analysis are read, improving the analysis and performance.



(L'Esteve, 2022).

Performance Comparison for Common Time Series Analytics Tasks

In time series analytics, a number of operational tasks can be performed where computation efficiency is of significant importance. Such tasks are aggregations within particular time frames, rolling or moving average computation, resampling of time series data, and performing time window operations. The following part of the paper evaluates how these platforms—Snowflake and Databricks—perform in such typical analytics processes.

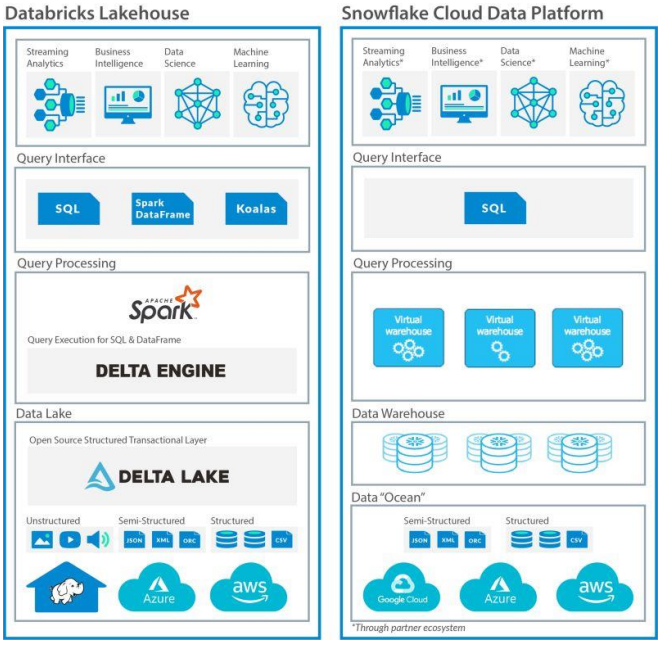
Snowflake's Performance:

That is why Snowflake completely isolated storage and compute capabilities while developing the company's architecture. This isolation achieved the flexibility of scaling the computing resource according to its workload without scaling the amount of space on disk required by the program without requiring the user. In Snowflake, multiple cluster computing is used, where several computing clusters can work with the same data in parallel, improving its performance during busy hours.

For most common time series analysis tasks, Snowflake performs well for basic aggregations on any desired interval, including daily, monthly, or yearly, using built-in SQL functions. While performing these aggregations, additional cache optimization is built into Snowflake itself, drastically improving read and write times. Data frequently accessed can be stored in the memory, and queries can be quickly executed since recall time would have been reduced (Van Renen & Leis, 2023). Also, micro-partitioning and clustering techniques on Snowflake provide physical storage strategies, ensuring rapid query performance on large data sets.

Databricks' Performance:

Databricks uses Apache Spark's distributed computing model, optimized to process big data by default. Every computation step in Spark is distributed across various nodes, and thus, Databricks can handle enormous time series databases. This distributed architecture provides the main benefit when aggregations are performed over a time interval, and the workload of such an operation can be distributed between nodes to expedite execution.



(Pulkka, 2023).

In the case of rolling or moving averages, Databricks provides a range of functions that integrate into Spark. These functions allow analysts to perform these calculations on large datasets without experiencing a drop in performance. Basically, Spark is a distributed platform that processes data in parallel, reducing the time it takes to arrive at an insight. Data resampling in time series in Databricks is quite fast owing to its rich set time manipulation functions that can work well irrespective of the chosen time measure (Van Renen & Leis, 2023). To this end, users can resample data in parallel with

their desired intervals via Spark. This feature is precious in organizations that need constant changes in the level of detail they work with based on analysis demands. ACID transactions are the key feature of Delta Lake, which is considered an inseparable part of Databricks and improves the Apache Spark capabilities. Some features include enhancing Delta Lake's usability when processing streaming time series data, facilitating real-time data handling, and real-time training data quality. This capability is critically important for organizations that need fresh analysis just in time.

Comparative Analysis

When comparing Snowflake and Databricks for time series data, several criteria appear, such as performance speed for large datasets, usability, cost, and configurability for change and machine learning.

Performance on Large Datasets:

Snowflake and Databricks demonstrate high performance; however, they perform it in different ways. Snowflake's design also enables materialization-based optimization, making it well-designed for SQL-based analytics and enabling straightforward workloads such as aggregations and time-based queries on massive datasets. It includes automatic caching and micro-partitioning meant to improve runtime performance. Instead, Databricks leverage a distributed computing model of Apache Spark to process the immense data in parallel (L'Esteve, 2022). This makes it perform complex calculations and real-time analysis well, mainly when processing extensive time series data.

Ease of Use:

Regarding the ease of using Snowflake, users can utilize a Structured Query Language (SQL) interface for easy system command since they apply a relational database. That is why analysts can quickly obtain results without in-depth programming experience. In contrast, Databricks needs some basic understanding of Spark and Python or Scala for further operations that might be challenging for some users. Nonetheless, Databricks' notebooks are helpful in data exploration and creating models, making them ideal for data scientists and engineers.

Cost Efficiency:

The cost efficiency of a product can differ significantly depending on the usage of the product in question. Snowflake operates on the consumption-based model, which means one has to pay only for what has been consumed, and this makes the service efficient for use when carrying out activities that require burst rather often. However, Databricks also comes with the exact mechanism of pay-as-you-go. Still, large-scale computation and usage of significant resources in the ML pipeline may also cause a charge hike.

Flexibility for Customization and ML Integration:

Databricks is most preeminent for its flexibility in configuration and tight coupling with the machine learning frameworks for more extensive analysis and model implementation. This capability is essential for organizations concerned with developing forecasting models from time series data. Although Snowflake has excellent potential for SQL analysis, it only offers a few extensive and fully integrated tools for most machine learning operations within the program.

Key Capabilities and Use Cases for Both Platforms

Snowflake:

It is necessary to note several of Snowflake's critical capabilities to ensure high efficiency when working with time series data. Built-in SQL capabilities enable users to construct specific queries and analyses and do not require using other languages, as may be preferable for data analysts (Pulkka, 2023). The scalability aspect of the platform effectively means that organizations can adjust compute resources as the demand requires, thereby consistently achieving peak utilization during high usage. Time travel allows the users to choose any date they want the program to give them the data analysis and report on, thus making it easier to make decisions (L'Esteve, 2022). Third, zero-copy cloning also offers a more efficient way to clone data in order not to incur more storage costs along with the dataset for use in experimenting with data.

Snowflake's specific applications include financial analysts who need accurate reporting and objective data on trends and performance. Retail time series forecasting is another example; one can examine time series data to make correct stock and promotion decisions in a business. Business reporting is another area where Snowflake outperforms the competition; organizations can obtain reports from multiple consolidated systems.

Databricks:

The real-time real-time provided by Databricks makes it ideal for applications that need data analysis and insights. Many use a distributed computing approach to help tackle large data sets and match growing business needs. Databricks works in harmony with other machine learning frameworks, including TensorFlow and MLlib, enabling the company to complete complex analysis and model development.

Some actual application areas of Databricks are IoT data processing, where various sensor data streams can be processed and analyzed in real-time. This brought out real-time realtimeket predictions that use Databricks to analyze and forecast stock market data as it comes in. Also, anomaly detection in time series is another application where

businesses can detect whether specific patterns or behaviours in the time series are perverted, which can indicate fraud or system failure (Van Renen & Leis, 2023).

Conclusion

In conclusion, the benefits and limitations of the Time series consumption can be used to discuss the performance of Snowflake and Databricks. Its strength is in SQL-based operation, simplicity, and self-managing optimization functions, but the least suitable for complex analysis is ideal for pure reporting requirements and financial analysis. The system's capacity for complex queries with a low amount of upkeep entices organizations that require a system to be agile. Its drawback is the lack of sophisticated machine learning tools, yet it might need additional tools for implementation.

On the other hand, the real-time processing and distributed computing model of Databricks is advantageous for large-scale analytics and machine learning operations. Its elasticity makes the quick and sophisticated computation of large data sets possible. With the help of its application, one can process, for instance, IoT data or look for anomalies within the data. To be sure, the seductive simplicity of Spark may be artificial; the content of learning Spark data for wide applications could be complicated, and programming skills are necessary.

References

[1] Deshpande, M., & Nanda, I. (2023). Empowering Data Programs: The Five Essential Data Engineering Concepts for Program Managers. Journal of Engineering and Applied Sciences Technology. SRC/JEAST-341. DOI: doi. org/10.47363/JEAST/2023 (5), 235, 2-12. https://www.researchgate.net/profile/Mahesh-Deshpande-8/publication/380192173_Empowering_Data_Programs_The_Five_Essential_Data_Engineering_Concepts_for_Program_Managers/links/66a854f9c6e41359a849c403/Empowering-Data-Programs-The-Five-Essential-Data-Engineering-Concepts-for-Program-Managers.pdf

[2] Kempter, Y. (2024). ML Data Processing on Relational Databases (Master's thesis, ETH Zurich). https://www.research-collection.ethz.ch/bitstream/handle/20.500.11850/677800/1/Kempter_Yves.pdf

[3] Kukreja, M., & Zburivsky, D. (2021). Data Engineering with Apache Spark, Delta Lake, and Lakehouse: Create scalable pipelines that ingest, curate, and aggregate complex data in a timely and secure way. Packt Publishing Ltd. https://books.google.com/books?hl=en&lr=&id=XiJEEAAAQBAJ&oi=fnd&pg=PP1&dq=Handling+Time+Series+Data+SNOWFLAKE+AND+DATABRICKS&ots=g4muPbOeDb&sig=JKY0WA04u2aOJTajPkll3UKqjfw

[4] L'Esteve, R. (2022). Databricks. In The Azure Data Lakehouse Toolkit: Building and Scaling Data Lakehouses on Azure with Delta Lake, Apache Spark, Databricks, Synapse Analytics, and Snowflake (pp. 83-139). Berkeley, CA: Apress. https://link.springer.com/chapter/10.1007/978-1-4842-8233-5_3

[5] L'Esteve, R. (2022). Snowflake. In The Azure Data Lakehouse Toolkit: Building and Scaling Data Lakehouses on Azure with Delta Lake, Apache Spark, Databricks, Synapse Analytics, and Snowflake (pp. 45-82). Berkeley, CA: Apress. https://link.springer.com/chapter/10.1007/978-1-4842-8233-5_2

[6] Macedo, R. P. D. C. C. (2024). Implementation of a Data Lake in a Microservices Architecture (Doctoral dissertation). https://repositorio.ul.pt/handle/10451/63925

[7] Pulkka, S. (2023). The Modernization Process of a Data Pipeline. https://www.doria.fi/handle/10024/187373

[8] Van Renen, A., & Leis, V. (2023). Cloud analytics benchmark. Proceedings of the VLDB Endowment, 16(6), 1413-1425. https://dl.acm.org/doi/abs/10.14778/3583140.3583156