xcalar

SOLUTION BRIEF

Accelerating Complex Business Logic Transformations for Financial Services

CHALLENGES

A prominent financial institution faced challenges improving data freshness and refining their Java/Spark batches to deal with the ever-increasing complexity of their big data transformations.

- Slow performance, delaying output to clients
- Long development & troubleshooting cycles
- Dependency on scarce expert programmers/consultants
- Lack of dev IDE for distributed applications
- High TCO, long time-to-value

SOLUTION

Using SQL, structured, and visual programming, the existing, bulky Java/Spark code was converted into reusable, modular dataflows within the intuitive, versatile IDE powered by Xcalar Data Platform. Detailed data lineage information surfaced inconsistencies at every stage, facilitating quick resolutions. As a next step, Xcalar's micro-batch update mechanisms will enable near-real-time data freshness.

BENEFITS WITH XCALAR

- **90% TCO reduction:** Due to lesser infrastructure requirements, faster development, and simpler maintenance
- **10x** scalability and performance improvement
- **5x** developer productivity improvement
- **20x** less time to operationalize applications
- **Efficient scaling with volume:** Doubling data volume increased run time by 20% only
- **Better production support** due to transparent data lineage and modularity of dataflows

Introduction

A prominent financial institution's website provides its customers with consolidated financial information, portfolio analyses, and mark-to-market valuations across all their investments. This data includes balances, historical statements of account, client holdings, gains and losses, and other details. This information changes frequently due to new trading transactions or corporate actions, including delayed legal case resolutions that go back in history and can affect years of historical data. As such, the institution must often apply business logic to recompute revenues, tax lots, balances, and other analytical computations using several years' worth of data.

A daily data transformation batch job processes 36 datasets, consisting of approximately 2 TB of raw data and 0.5 GB of daily deltas from HDFS/Parquet, to compute analytical views for clients to view on the website. The batch consists of two parts:

1. Computing dimensions and rows of interest.
2. Applying the business logic transformations to compute the final results.

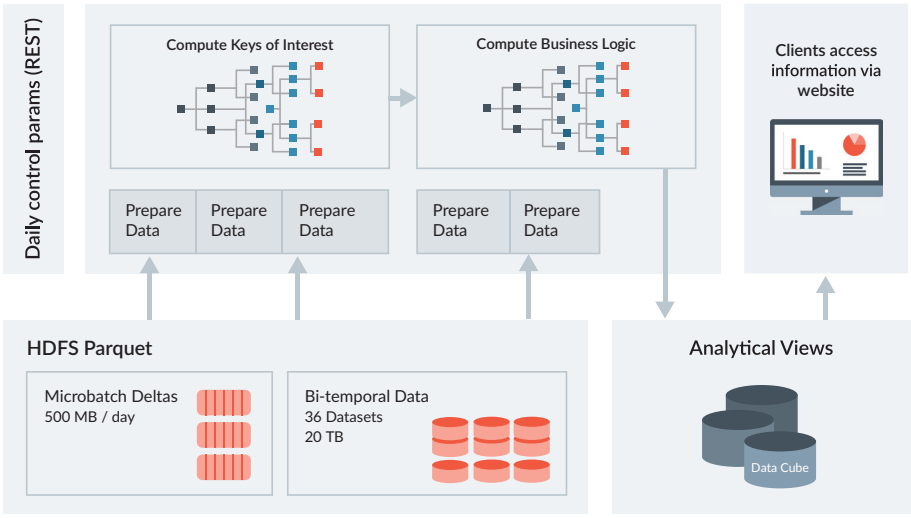
Challenges

The original code for this daily batch process was about 20,000 lines of Java, and the latest revision took two senior SME engineers three months of work in Spark. Due to inefficient joins and indexing, the solution took an average of 5 hours to process on a commodity 8-node Spark cluster. The code lacked transparency and was very hard and expensive to troubleshoot. Since only a few developers understood the code, the firm suffered both from a risky reliance on a few key employees and a dependency on highly skilled engineers with expertise in Java, Spark, SQL, Parquet, and Cloudera.

Xcalar Solution

Using Xcalar Design, the versatile IDE powered by the underlying Xcalar Data Platform, it took two data engineers only two weeks to translate 20,000 lines of Spark code into Xcalar visual dataflow graphs and most of those two weeks were spent reverse engineering the original business logic. This huge 5x development efficiency gain was mainly due to Xcalar's visual programming paradigm and interactive debugging capability. Engineers were able to verify results for every operation and receive immediate feedback at every step. As a result, inconsistencies and redundancies were very easy to catch and resolve. Besides achieving the original goal to convert Spark code to Xcalar dataflows, this exercise uncovered a dozen bugs in the original code. Issues that were previously hard to discover were made apparent while the engineers rebuilt the batch as visual dataflows.

In comparison to incumbent technologies, very little effort was spent on performance tuning because, in Xcalar, data does not need to be indexed or partitioned in advance. Distributed indexes are created automatically as



needed for each transformation. Additionally, Xcalar provides a simple way to monitor data skew and offers comprehensive skew management methods that improve performance.

The development and debug cycle was shortened further by Xcalar's language-agnostic visual programming paradigm. Xcalar Design enables engineers to build modular dataflow graphs using point-and-click methods to apply relational algebra operators, such as joins, sorts, maps, group bys, filters, aggregates, unions, and pivots, in conjunction with full ANSI SQL statements and procedural language constructs such as loops and conditionals.

Using Xcalar, the developers used the combination of visual modeling, SQL, and custom User Defined Functions (UDFs) to create easy-to-use dataflow graphs. The original Spark code was converted into a series of reusable dataflows instead of one giant SQL statement. These dataflows were modular to build and invoke, and therefore were easy to debug independently. Data preparation dataflows were separated from the actual business logic to make the process more transparent. With this process abstraction, datasets prepared by upstream teams could be reused by downstream business applications and updated at different frequencies, fostering collaboration between engineering teams. Additionally, the JSON representation of Xcalar dataflows was used both for integration with Git, as source control, and for collaboration across multiple teams, each working on different dataflows.

The ability to trace data lineage through column renames, aggregates, and complex transformations played a critical role in speeding up the batch development. The power-user tools provided by Xcalar enabled engineers to trace the lineage by showing the journey of data columns from their source, as well as identifying the data sources and fields of interest that contributed to various calculations and transformations.

Upon completion, the batch run was tested with increased data volume. Due to Xcalar's True Data in Place™ technology and separation of data storage and compute, the cluster did not need to be scaled when data volume increased. For example, only 3 additional minutes (a 20% increase) were needed to execute the complete business logic transformation for *double the volume* of data.

As a next step, it is planned to further improve data freshness by having constantly executing Xcalar micro-batches update the data in near real-time.

Key Features

- Ad hoc analytics/modeling and operationalization
- Visual programming, SQL, and structured programming paradigms
- Lineage and auditability
- Separation of storage from compute
- High performance and scalability
- Visual cluster monitoring tool
- Skew management
- Collaboration

About Xcalar

Xcalar is a scale-out platform for data processing applications and operationalizing ML. The platform is open and extensible, and suitable for developing and operationalizing business logic. Xcalar's use cases include virtual data warehousing to enable BI tools to query real-time transactionally consistent data, operationalizing ML algorithms at cloud-scale, as well as simplifying data transformation and quality processes. Users use a versatile IDE to interactively build dataflows using SQL, visual programming, and structured programming, and execute them at petabyte scale. Xcalar's enterprise-grade software scales linearly to hundreds of nodes and thousands of users for public/private cloud and hybrid deployments.

Xcalar, Inc. 3031 Tisch Way Suite 450 San Jose CA 95128 USA Tel 408-471-1711 www.xcalar.com

XSB-96849-1-14-08