striim

Real-Time Data Streaming from Oracle to Google BigQuery:

A Performance Study

Authors
Sreedhar Goverapet, Alok Pareek

Contributors
Sreedhar Goverapet, Alok Pareek, Mahadevan Laxminarayanan, Bhushan Khaladkar, Vino Chithra

striim

Table of Contents

Introduction

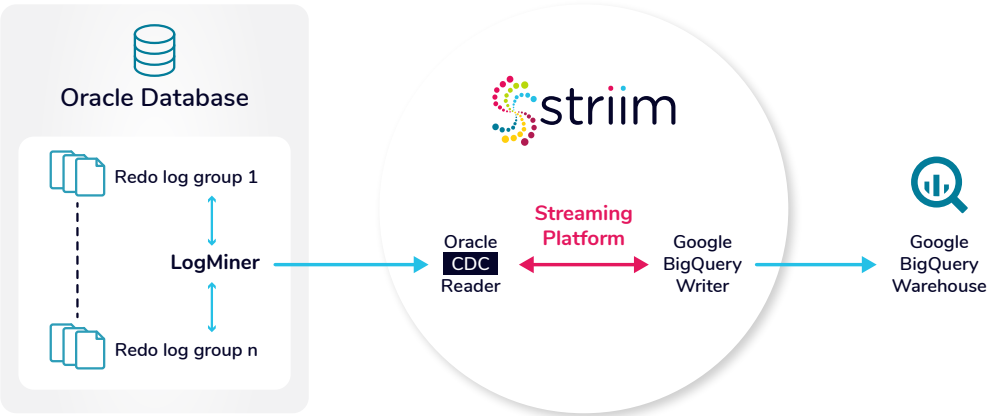
With growing business requirements to provide real-time data to make decisions with low latency, it becomes critical for large enterprises to adopt newer ways to incorporate data integration within their broader data management and digital transformation strategy. Data is often managed by diverse systems like transactional databases, data warehouses, file stores, object stores, messaging systems, etc. depending on business needs. In today's digital economy, it is important to be able to integrate and manage data across these diverse platforms in real time to provide a rich customer experience with intelligent behavioral actionable insights at the time of user engagement.

For integrating real-time data into Google BigQuery, Striim is a **Google Cloud Ready - BigQuery** designated platform.

In this whitepaper, we show how Striim's streaming platform can be used to move data between a high-throughput Oracle transactional database to Google BigQuery in real time.

Logical Flow of the Data Pipeline

Striim provides ultra-low latency adapters to connect to diverse data sources and targets to build scalable and reliable data pipelines for real-time data integration. In the current case, we use the Oracle Change Data Capture (CDC) adapter to connect to the source database and the BigQuery Writer to push data into the BigQuery target warehouse. The logical design of the streaming solution is shown below:



The Oracle CDC adapter captures all change data from the source Oracle database in real time and outputs it to Striim's streaming platform.

Striim supports two flavors of Oracle Database (DB) CDC adapters. First is the LogMiner-based Oracle Reader that uses an Oracle LogMiner session to scan for database changes on the server side and capture the same to the streaming platform. The second is the OJet adapter that is designed for very large scale real-time data capture using a high-performance logmining API. The changes captured by the source adapter are pushed into Striim's streaming platform. In this post, we show how real-time integration can be achieved using both variants of the Oracle CDC reader and provide insights on how to choose the best adapter for a given use case. For complete details on the CDC reader, please refer to the **Oracle CDC Reader Documentation**.

Striim's native streaming platform is a fault-tolerant and scalable in-memory platform that provides real-time capabilities to ingest, process, enrich and transform incoming events. The platform also has a native integration with Kafka, providing the ability to persist streaming data for various use cases. For more details on Striim's platform, please refer to the **Striim Concepts Guide**.

Striim's BigQuery Writer is a low-latency Data Warehouse Writer that provides various options to push data to BigQuery tables in a fast and reliable manner. The writer supports both batch file uploads and streaming ingestion of data into the target tables. The writer also takes advantage of partitioned tables on the target and supports partition pruning in its merge queries that provides considerable time and storage savings while merging complex data into very large tables. For more details on the BigQuery Writer, please refer to the **BigQuery Writer documentation**.

Database Workload

For the purpose of this experiment, we use a custom-built, high-scale database simulation workload. The workload, SwingerMultiOps, is modeled around the popular **Swingbench** workload for Oracle databases. We take the Order Entry (OE) schema of the Swingbench workload. OE schema is shipped with all releases of Oracle Database. In SwingerMultiOps, we have expanded the Swingbench OE schema to include more tables. There are a total of 50 tables in our schema. Each table has a mix of data types including VARCHAR, VARCHAR2, NVARCHAR, NUMBER, DATE, TIMESTAMP and INTERVAL YEAR TO MONTH.

SwingerMultiOps is a multithreaded Java Database Connectivity (JDBC) application that spawns concurrent DB sessions against the source database. Each session performs inserts and updates on a specified set of tables. Deletes are performed from a separate DB session. The ratio of deletes and updates to inserts is parameterized and can be controlled at the time of invocation. The DMLs are executed for a given duration of time and the output is written to console.

Building the Data Pipeline

Building the data pipeline involves preparing the source database for replication, profiling the database workload, configuring the Oracle CDC Reader adapter and configuring the BigQuery Writer. Each of the steps is explained below:

Configuring the Source Database and Profiling the DB workload

Striim's Oracle adapters (Oracle Reader and OJet) connect to an Oracle server instance to mine for redo data. Therefore it is important to have the source database instance tuned for optimum redo mining performance. Similarly, the DB workload needs to be profiled to identify the load that it generates on the source database. This load can be measured in terms of the number and kind of DMLs per unit time, number of tables modified, volume of redo data generated and system resources on the DB server. For each of the two adapters, these settings and workload profile differ because of the load volume that the two adapters can handle for real-time integration.

As a general rule, it is best to set redo log sizes to a reasonably large value of 2G per log group. In case of Oracle Real Application Clusters (RAC), add a log thread for each RAC instance in the redo log group. Large redo log size reduces the number of logfile switches on the server and provides better performance for LogMiner sessions.

While using the OJet adapter, it is recommended to explicitly set a large size for the DB streams_pool_size to mine redo as quickly as possible. For an extremely high CDC data rate of around 150 Gb/hour, it is recommended to set streams_pool_size to 4G. These settings are shown as follows:

```
SQL>  select group#, (((bytes/1024)/1024)/1024) bytes_gb from v$log;

    GROUP#   BYTES_GB
---------- ----------
      16        2
      17        2
      18        2
      19        2
SQL> show parameter streams_pool_size;

NAME                           TYPE    VALUE
---------------------------------- -----------
streams_pool_size              big integer 4G

SQL> show sga;

Total System Global Area       32212253272 bytes
Fixed Size                     23071320 bytes
Variable Size                  7784628224 bytes
Database Buffers               24360517632 bytes
Redo Buffers                   44036096 bytes
```

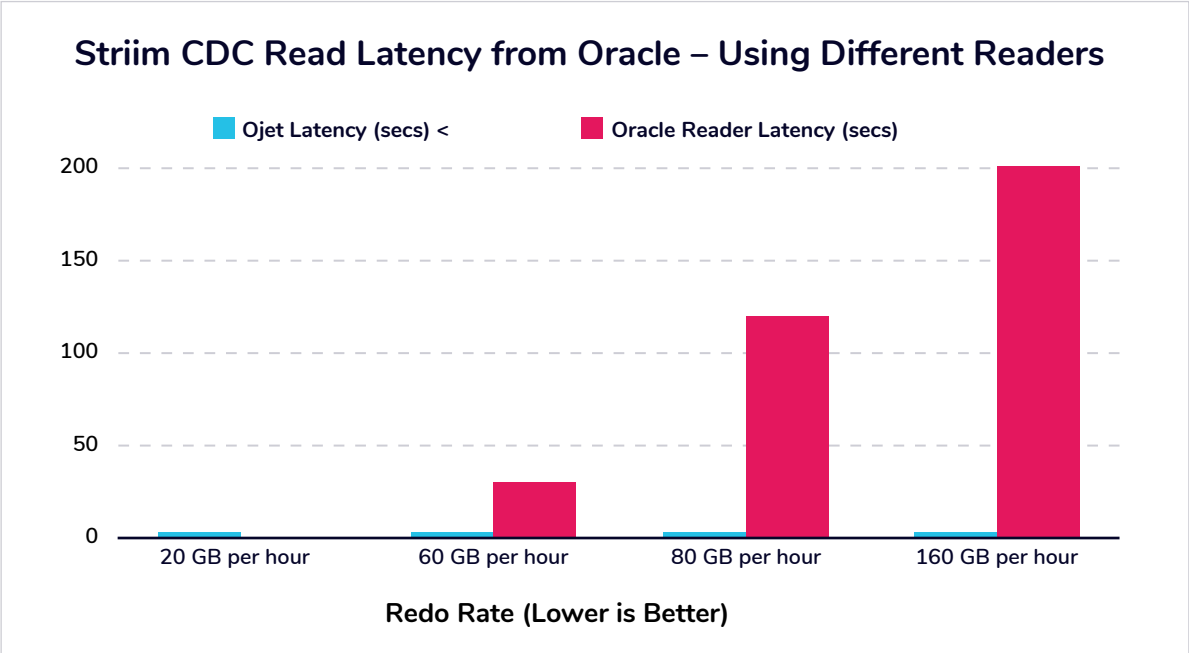
DB Workload Profile	DML distribution and Events Generated
Upto 20G of CDC data per hour	No of Deletes: 6820, No of DDLs: 0, No of PKUpdates: 0, No of Updates: 3688950, No of Inserts: 14778988 Total number of events: 18,474,758 (18.47 million)
Upto 160G of CDC data per hour	No of Deletes: 6600, No of DDLs: 0, No of PKUpdates: 0, No of Updates: 28363950, No of Inserts: 113478240 Total number of events: 141,848,790 (141.84 million)

Choosing the right CDC Adapter

Striim provides two CDC adapters for the Oracle database. The first one is the Oracle Reader that captures CDC data using the LogMiner session on the server side. The second is the OJet adapter that uses a high-performing logmining API and offers the best performance for high-scale workloads. Irrespective of the scale of operation, Striim's native streaming platform offers seamless integration capabilities to ingest and process all incoming data in real time. The rate of processing incoming data is determined by adapter latency. Similarly, the rate of pushing data into a given target is determined by the target adapter performance. When there are multiple adapters used in the topology, Striim's router component can be deployed to effectively distribute events to different adapters and keep the application latency low. Only in cases where adapters are poorly tuned or the source or target is having reduced performance, the streaming pipeline builds a back pressure and throttles the overall system throughput. Such bottlenecks can be avoided by tuning the adapters correctly as described later in the document.

The table and graphic below provides a guidance on choosing the right CDC adapter for real-time integration:

DB Workload Profile	DB Adapter recommendation	Adapter latency (read-lag)
Upto 20G of CDC data per hour	Oracle CDC reader	Under 3 seconds
Upto 60G of CDC data per hour	Oracle CDC reader	Under 30 seconds
Upto 80G of CDC data per hour	Oracle CDC reader (Use OJet for sub second latency)	At least 2 minutes
Upto 160G of CDC data per hour	OJet	Under 2 seconds



Configuring the CDC Adapter

Striim supports two types of CDC adapters as explained above. For low and medium workloads, LogMiner-based Oracle Reader can be used for real-time integration. The adapter spawns a LogMiner session on the server and retrieves all CDC data over a JDBC connection maintained by Striim server.

For extremely large volumes of CDC data, we recommend using the OJet adapter that can handle very high rates of incoming CDC data as shown above. OJet is the fastest Oracle CDC reader on Earth!

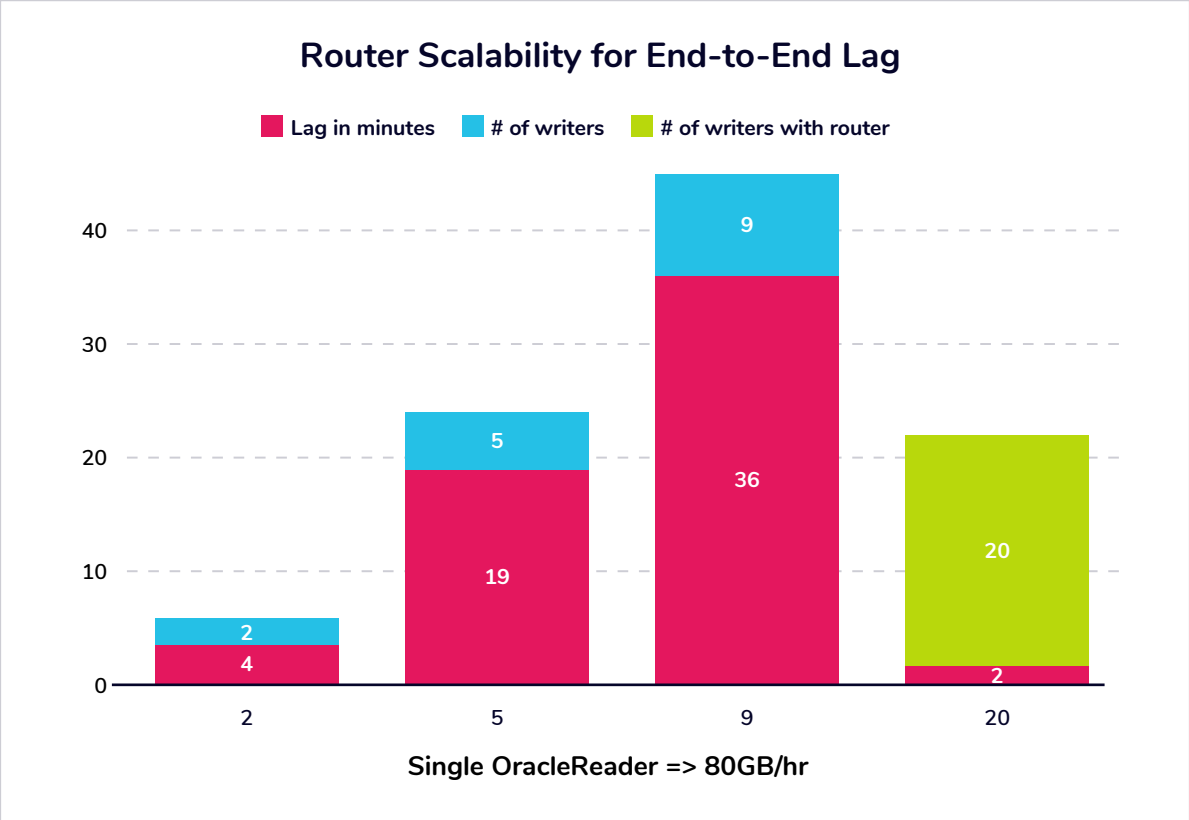
Either of the adapters can be used with default settings and only the DB end points are needed to start reading from the source database. At the same time, Striim also allows advanced configuration options to handle large transactions, reading from and writing to a downstream database, mining from a specific SCN or timestamp, etc. A complete list of adapter parameters and their details can be found in **Oracle Reader Documentation**.

Irrespective of the adapter type and configurations, only one CDC adapter is required to capture all incoming data from the source database. This removes the overhead of running multiple logmining sessions (in the case of LogMiner CDC reader) or capture processes (in case of OJet) on the DB server and the overhead of managing additional adapter components in Striim server.

Managing incoming events with routers

A single-source side adapter like the Oracle CDC reader can capture all changes at the source database for any number of tables. This is because the reader scans the redo logs that contain changes made on all tables in the database. However, on the target side, systems like Google BigQuery allow multiple concurrent inserts into the warehouse system. Therefore, to improve performance, we can use multiple BigQuery writers to integrate incoming data in parallel. When multiple such writers are used, it is important to manage and distribute events across multiple writers in such a way that a given event does not land in multiple writers. To achieve this, a router can be used in the topology. A router is a type of Continous Query (CQ) that works on a set of input conditions and distributes the incoming events to a set of streams based on the input conditions. More about routers can be found in **Striim programmer guide**.

The diagram below shows the variation in event throughput with and without routers:



Configuring the BigQuery Writer

The BigQuery Writer captures the incoming events from the Oracle Reader adapter, maps them to target table and data types and then moves data to the BigQuery dataset. The BigQuery Writer comes with several properties that can be used to effectively manage and control movement of data. Further, several writers can be configured as required to work on a particular dataset to move large amounts of data in parallel. Tuning the number of writers and writer properties in the right way will ensure that the data is propagated to the target in real time.

Configuring BigQuery Writers and Properties

Given the high volume of incoming data, we configure 20 BigQuery Writers. Also, for moving data into the target in real time, we use BigQuery Writer's Streaming Upload method. Other writer properties are tabulated below:

Writer Number	Properties	Number of source tables mapped
1 to 10	streamingUpload: 'true' BatchPolicy: 'Interval:5' Mode: 'APPENDONLY'	3 tables per writer
11 to 20	streamingUpload: 'true' BatchPolicy: 'Interval:30' Mode: MERGE	2 tables per writer

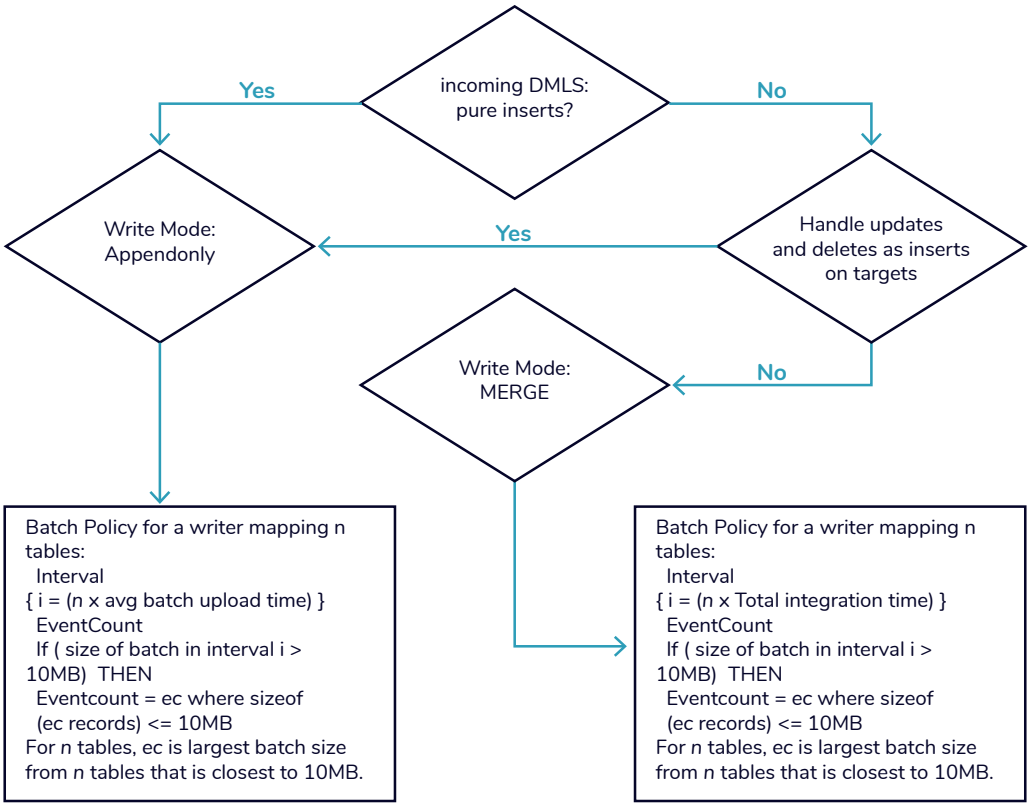
The properties used above are explained below:

- **Streaming Upload**: Setting streaming upload to true will make the target adapter use the streaming API to ingest incoming data to target tables. Data can be ingested in smaller batches and at a higher rate using streaming API.
- **BatchPolicy:** Batch policy along with the write mode are the most critical policies that help the end user to achieve real time integration. Batch policy instructs the adapter on when a batch is to be cut for integration from the incoming events. For real-time integration, batch policy should be tuned such that no batches are queued up pending integration.
- **Write Mode:** Write mode along with batch policy is a critical tuning parameter. In APPENDONLY mode, all incoming data is appended to target tables. This mode can be used when the source table has only insert DMLs. It can also be used with UPDATE and DELETE DMLs if the end user wishes to defer running the UPDATE or DELETE jobs on the target until a time of their convenience. In such cases, the metadata for a table UPDATE or DELETE can be extracted from the incoming event and applied to the target table using Striim's metadata filters.

A complete list of properties supported by BigQuery Writer is documented in **Adapters Guide**.

The formula on the next page can be used as a guidance to set a correct batch policy for the BigQuery Writer to achieve real-time integration.

Batch Policy for Streaming Mode Ingestion



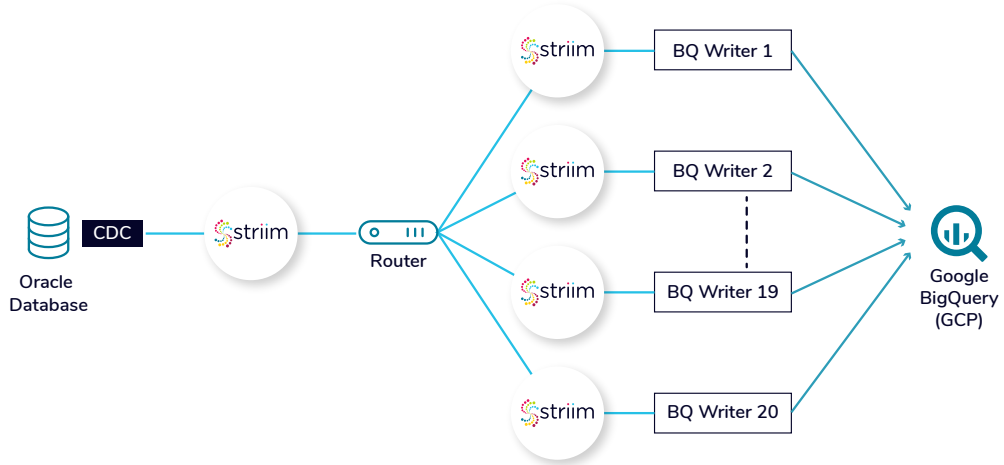
In the above flowchart, Total Integration Time is the time it takes to complete integration at the BigQuery system. For APPENDONLY mode, Total Integration Time is the time taken to ingest data to the system. However, for MERGE mode, Total Integration Time is the sum of the following:

Total Integration Time = (Time to Upload + Time to Compact + Time to Merge)

Time to Compact is the time taken to compact the operations in a given batch and remove all redundant operations so as to ensure minimum merge time. Compaction can be done in-memory (on the Striim server) or on the target BigQuery system (by automatically creating a staging compaction table).

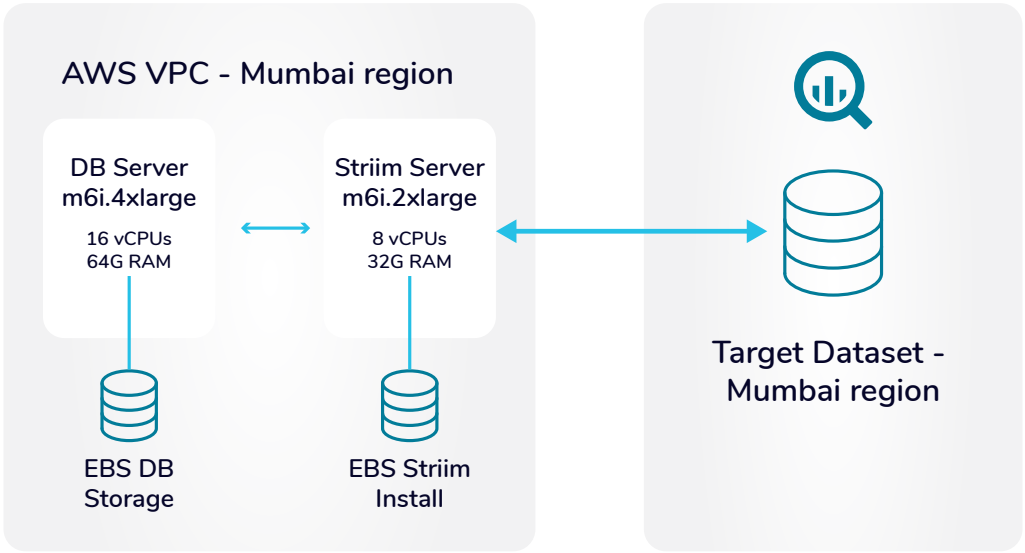
Time to Merge is the time taken by the BigQuery system to complete the merge query. This time is determined by the target system and Striim does not have control over the same. Customers are advised to query their analytical systems to determine the time taken by their merge queries before setting batch policies for the BigQuery writers.

Having tuned the source database and configured our adapters, the data pipeline is built as an app in Striim server. The app is illustrated below:



App Execution and Analyzing Results

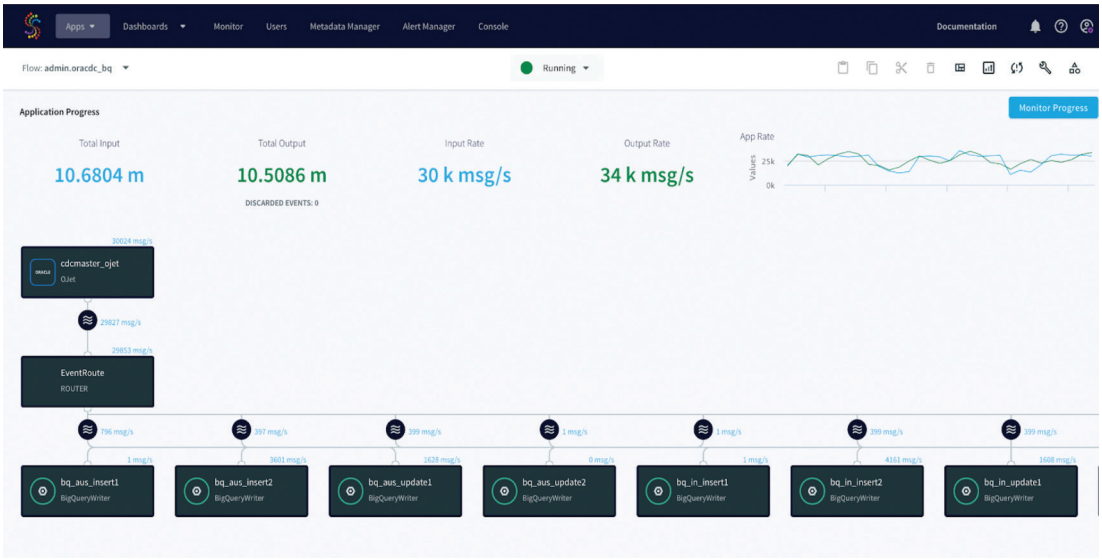
We run our Data Integration app on an infrastructure backed by AWS EC2 instances and a Google BigQuery dataset. The infrastructure is illustrated below:



We perform the following tasks to run our simulation and capture results for analysis :

- Start the Striim app on Striim server
- Start monitoring our app components using **Tungsten Console** by passing a simple script.
- Start the Database Workload
- Capture all DB events in the Striim app and let the app commit all incoming data to the BigQuery target
- Analyze the app performance

The picture below of the Striim UI shows our app running in the Striim server. We can monitor the app throughput and latency in real time from the UI.



Results

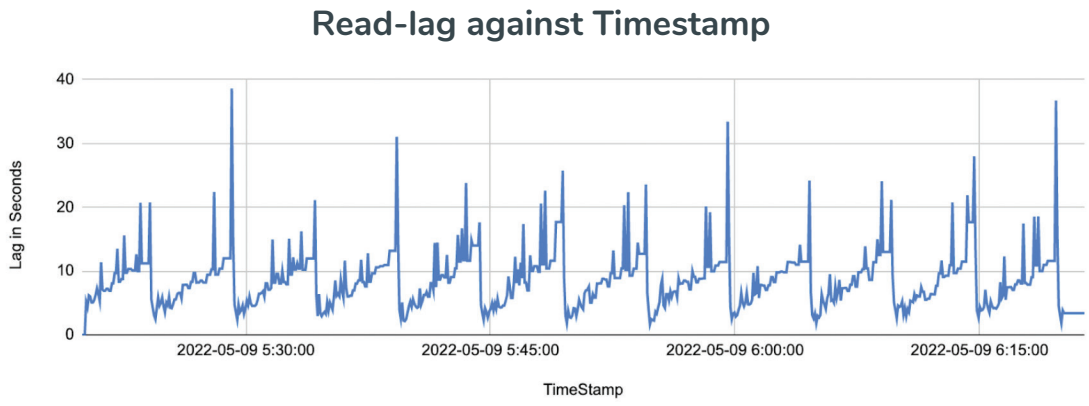
At the end of the DB workload run, we look at our captured performance data and analyze the performance. Details are tabulated below for each of the source adapter types:

DB Workload Profile	DB Adapter Used	Results
20G of CDC redo generated in 1 hour with 18.47 million events	Oracle CDC Reader	CDC reader latency: 3s Avg LEE across 10 writers for APPENDONLY: 8.6 seconds Avg LEE across 10 writers for MERGE: 43.1 seconds CPU utilization: max 36% Memory utilization: 2G
160G of CDC redo generated in 1 hour with 141.84 million events	Oracle OJet	CDC reader latency: 2s Avg LEE across 10 writers for APPENDONLY: 2 seconds Avg LEE across 10 writers for MERGE: 26 seconds CPU utilization: max 36% Memory utilization: 2G

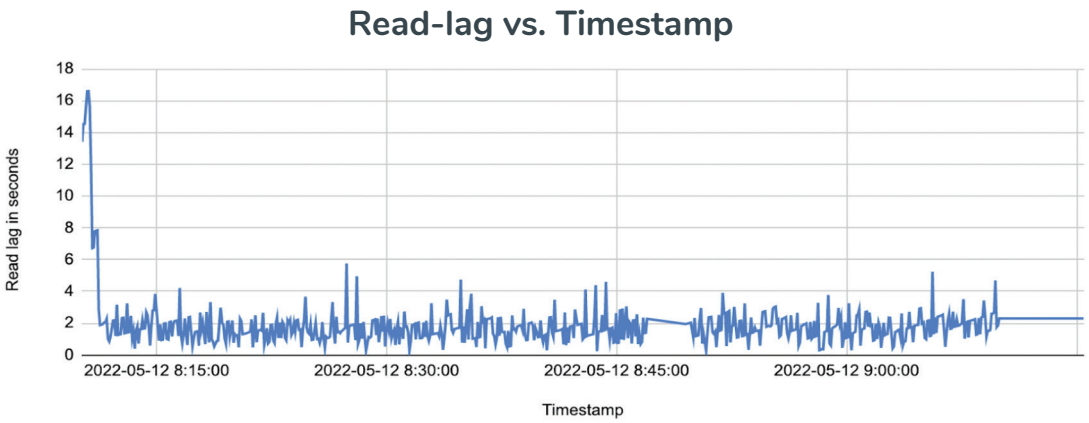
*LEE => **Lag End-to-End**

The charts below show how the CDC reader lag varies with the input rate as the workload progresses on the DB server.

Lag chart for Oracle Reader:



Lag chart for OJet Reader:



Conclusion

Through this experiment, we have shown how Striim's real-time streaming platform can be used to move high volumes of real-time data from Oracle to BigQuery. This document is only to be used as a guide to set up streaming pipelines. For assistance on complete deployment, please contact **Striim Support**.

About Striim

Striim was founded with a simple goal of helping companies make data useful the instant it's born.

Striim's unified, real-time data streaming and integration platform for analytics and operations collects data in real time from enterprise databases (using non-intrusive change data capture), log files, messaging systems, and sensors, and delivers it to virtually any target on-premises or in the cloud with sub-second latency enabling real-time operations and analytics.

Try it now at **go2.striim.com/free-trial**

Contact us at:

Tel: +1 650 241 0680
Web: **www.striim.com**

striim