DATA ANALYTICS

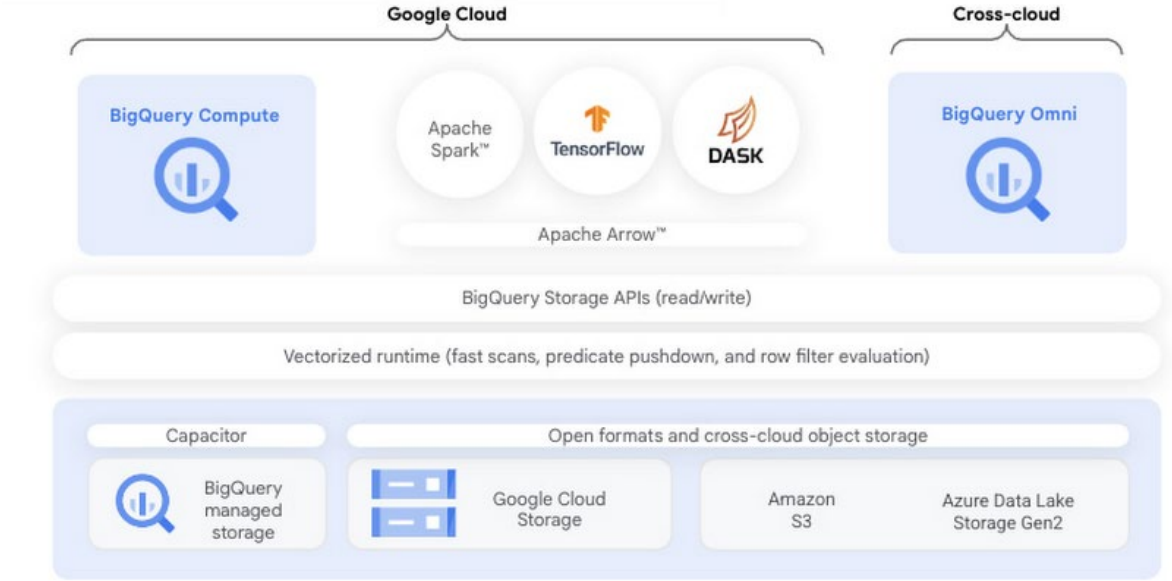
Scalable Python on BigQuery using Dask and NVIDIA GPUs

Authors:
Dong Meng, Solution Architect, NVIDIA

Christopher Crosbie, Product Manager, Google Cloud

BigQuery is Google Cloud's fully managed serverless data platform that supports querying using ANSI SQL. BigQuery also has a data lake storage engine that unifies SQL queries with other open source processing frameworks such as Apache Spark, Tensorflow, and Dask. BigQuery storage provides an API layer for OSS engines to process data. This API enables mixing and matching programming in languages like Python with structured SQL in the same data platform.

This post provides an introduction to using BigQuery with one popular distributed Python framework, Dask, an open source library that makes it easy to scale Python tools to BigQuery sized datasets. We will also show you how to extend Dask with RAPIDS, a suite of open-source libraries and APIs to execute GPU-accelerated pipelines directly on BigQuery storage.



Integrating Dask and RAPIDS with BigQuery storage

A core component of BigQuery architecture is the separation of compute and storage. BigQuery storage can be directly accessed over a highly performant Storage Read API which enables users to consume data in multiple streams and provides both column projections and filtering at the storage level. Coiled, a Google Cloud Partner that provides enterprise-grade Dask in your GCP account, developed an open-source Dask-BigQuery connector (GitHub) that enables Dask processing to take advantage of the Storage Read API and governed access to BigQuery data. RAPIDS is an open sourced library spawned from NVIDIA that uses Dask to distribute data and computation over multiple NVIDIA GPUs. The distributed computation can be done on a single machine or in a multi-node cluster. Dask integrates with both RAPIDS cuDF, XGBoost, and RAPIDS cuML for GPU-accelerated data analytics and machine learning.

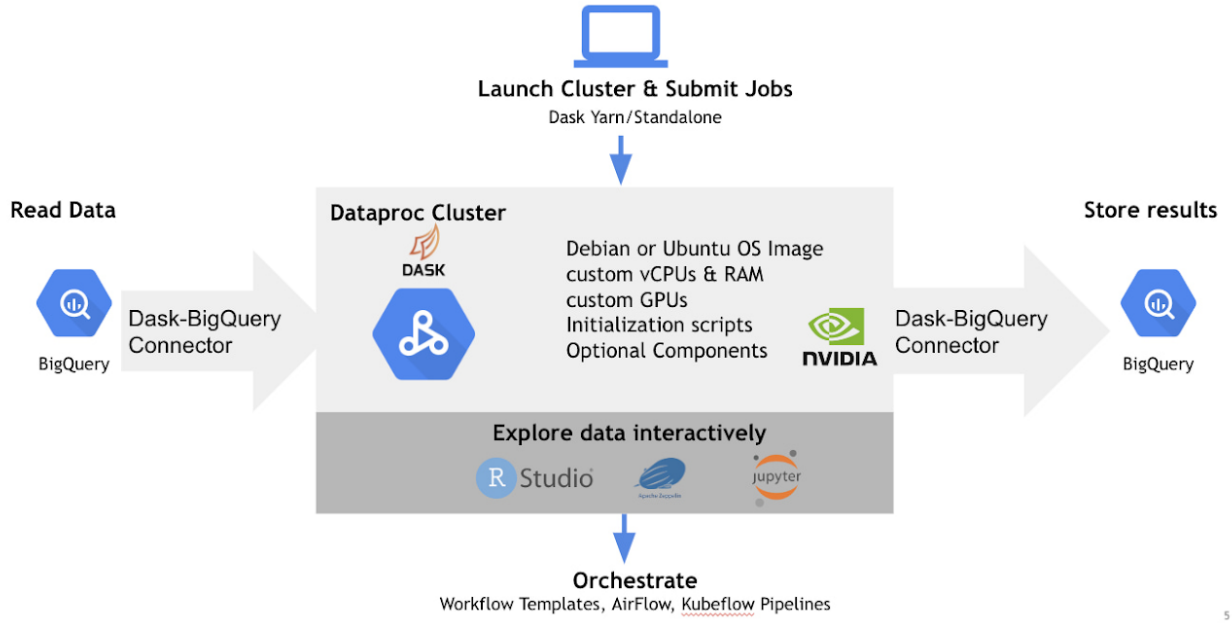
To start using Dask using BigQuery data, you can install the dask-bigquery connector from any Python IDE. You simply install `dask-bigquery` with `pip` or `conda`, authenticate with Google Cloud, and then use the few lines of python code as shown below to pull data from a BigQuery table.

```
   import dask_bigquery ddf = dask_bigquery.read_gbq(
project_id="your_project_id", dataset_id="your_dataset",
table_id="your_table", ) ddf.head()
```

Achieving Python scalability on BigQuery with Dataproc

While Dask and the BQ connector can essentially be installed anywhere that Python can be run and scale to the number of cores available in that machine, the real power of scaling comes in when you can use an entire cluster of virtual machines. An easy way to do this on Google Cloud is by using Dataproc.

Using the initialization actions outlined in this GitHub repo, getting setup with Dask and RAPIDS on a Dataproc cluster with NVIDIA GPUs is fairly straightforward.



Let's walk through an example using the NYC taxi dataset.
As a first step, let's create a RAPIDS accelerated Dask yarn cluster object on Dataproc by running the following code:

```
  from dask.distributed import Client from dask_yarn import YarnCluster
cluster = YarnCluster(worker_class="dask_cuda.CUDAWorker", worker_gpus=1,
worker_vcores=4, worker_memory='24GB',
worker_env={"CONDA_PREFIX":"/opt/conda/default/"}) cluster.scale(4)
```

Now that we have a Dask client, we can use it to read the NYC Taxi dataset in a BigQuery table through the Dask BigQuery connector:

```
  d_df = dask_bigquery.read_gbq( project_id="k80-exploration",
dataset_id="spark_rapids", table_id="nyc_taxi_0", )
```

Next, let's use RAPIDS Dask cuDF libraries to accelerate the preprocessing with GPUs.

```
  taxi_df = dask_cudf.from_dask_dataframe(d_df) taxi_df = clean(taxi_df,
remap, must_haves) taxi_df = taxi_df.query(' and '.join(query_frags))
```

Finally, we can use a feature of the Dask dataframe to split into two datasets — one for training and one for testing. These datasets can also be converted to XGBoost Dmatrix and sent into XGBoost for training on GPU.

```
  xgb_clasf = xgb.dask.train(client, params, dmatrix_train,
num_boost_round=2000, evals=[(dmatrix_train, 'train'),
(dmatrix_test,'test')] )
```

The complete notebook can be accessed at this GitHub link. Currently, Dask-BigQuery connector doesn't support native write back to BigQuery, user need work around that through cloud storage, with Dask or Dask Rapids, write back to GCS first with `to_parquet("gs://temp_path/")`, then having BigQuery load from GCS with: `bigquery.Client.load_table_from_uri("gs://temp_path/")`.

What's next

In this blog, we introduced a few key components to allow BigQuery users to scale their favorite Python libraries through Dask to process large datasets. With the broad portfolio of NVIDIA GPUs embedded across Google Cloud data analytics services like BigQuery and Dataproc and the availability of GPU-accelerated software like RAPIDS, developers can significantly accelerate their analytics and machine learning workflows.

Acknowledgements: *Benjamin Zaitlen, Software Engineer Manager, NVIDIA; Jill Milton, Senior Partnership Manager, NVIDIA, Coiled Developer Team.*