e-nor | Measure. Analyze. Optimize.

The Big Family of BigQuery: Achieving Reporting Continuity after Platform Migration

Google Analytics Certified Partner
Services - Sales

Google Cloud Platform

About the Client

North American chain of department stores with a strong online/ecommerce presence.

Goals

- Improve report automation after migration from Adobe Analytics to Google Analytics 360.
- Maintain continuity in reporting after the migration.

Approach

- Using an ETL (extract/transform/load) workflow and leveraging Google Cloud Platform and BigQuery as a data lake (primary data repository), imported 3+ years of Adobe Analytics clickstream data to reside side by side with Google Analytics 360 data.
- Mapped implementation requirements, as well as the format of collected data, to a common schema.
- Created intermediate/aggregation BigQuery tables to facilitate the comparison of the pre- and post-migration datasets.

Many CMOs and CIOs (and CDOs) were first introduced to BigQuery and Google Cloud Platform when Google announced linking Google Analytics 360 to BigQuery, enabling the GA data to be exported daily to BigQuery.

Since then, many look at BigQuery as an extension to Google Analytics 360 and tend to forget it's part of a bigger family, the family of Google Cloud Platform. When one thinks of BigQuery as a component of Google Cloud Platform, it becomes even more powerful and can be utilized for a wider range of tasks.

So, what is Google Cloud Platform?

Think of it as a suite of web-based components that can be used together, or standalone, to build automatically scalable cloud-based web applications.

When we first used Google Cloud Platform, about 9 years ago, it didn't have that many components. It was mainly App Engine which enabled developers to develop web applications, in Java or Python. It would scale automatically, adding/removing more servers as the load increases/decreases, and that was the main differentiator. It also had a very important component called Datastore, a scalable NOSQL repository. AppEngine wasn't adopted as quickly as expected, but over time, many important changes took place.

The number of components kept increasing, each optimized to perform a certain task. BigQuery is one of the best examples of specialized components, focusing on storing huge amounts of data that may be queried very fast at the expense of losing the ability to update or delete single records. BigQuery can be used alone or within the context of the Cloud Platform. For the majority of Google Analytics projects, it has been used as a standalone service and sometimes connected to Data Studio. Let's discuss how it paid off to use BigQuery with other GCP components.

Adobe Migration for an e-Retailer

In our efforts to migrate a global e-Retailer off of Adobe Analytics to Google Analytics 360, one of the key components of the project was maintaining access to historical *Adobe Clickstream* data collected by Adobe Analytics over the last several years.

Adobe data was available in the form of compressed CSV files, about 20GB for each day of 3 years, for two tracked platforms (that is ~44 TB!!). Needless to say, the format and schema of these files are quite different from GA's data, especially tracking e-commerce user interactions.

Results

- Significant savings in time and effort in accessing massive datasets.
- Enabled automated reporting for analysis of user journey, online conversion rates and overall site performance.
- Enabled continuous trending of key metrics between the two previously separate datasets.

About E-Nor

- E-Nor is a global digital analytics and marketing intelligence consulting firm devoted to enabling organizations to become more data-driven.
- Headquarters: Santa Clara, CA.
- www.e-nor.com.

About BigQuery

- BigQuery is one of Google Cloud Platform's (GCP) specialized components, focusing on storing huge amounts of data.
- It is optimized for very fast queries at the expense of losing the ability to update or delete single records.
- It is a multi-tenant cloud-based service.
- BigQuery can be used alone or in conjunction with other GCP components, such as Data Studio.
- To learn more, visit e-nor.com/products/bigquery.

Project Objectives and milestones

One of the main objectives, assigned to E-Nor's Data Engineering team, was to provide a way to process the historical Adobe Analytics files and generate measurements and reports comparable to what GA reports, for year-to-year and month-to-month metrics.

Value Versus Effort

At first, some of the stakeholders were a bit skeptical of the value of this objective compared to the effort required to accomplish it. After studying the differences and developing a proof of concept, we decided it was possible and the estimated effort was not as huge as initially thought, if we utilize other GCP components.

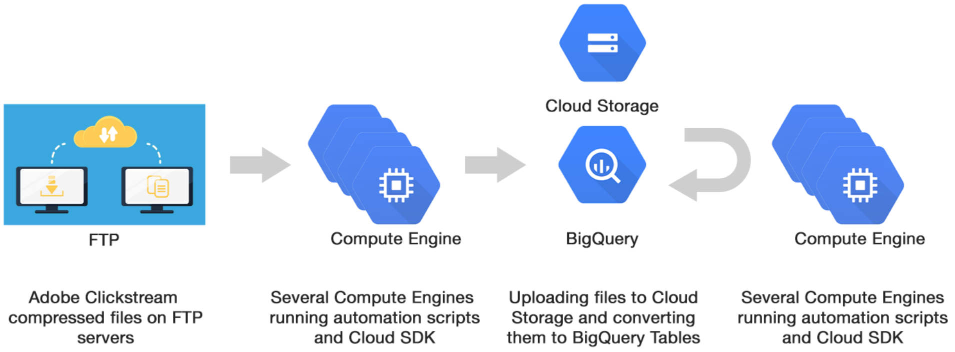

Cloud Storage
FTP
Compute Engine
BigQuery
Compute Engine
Adobe Clickstream compressed files on FTP servers
Several Compute Engines running automation scripts and Cloud SDK
Uploading files to Cloud Storage and converting them to BigQuery Tables
Several Compute Engines running automation scripts and Cloud SDK

Diagram showing data flow from compressed files on FTP servers, to BigQuery tables, to the generation of intermediate/aggregation tables.

We planned to use:

- BigQuery
- Cloud Storage
- Cloud SDK
- Compute Engine
- Automation scripts

The steps carried out by the automation script were as follows:

1. Download the compressed Adobe Analytics file for a certain day from an FTP server.
2. Decompress the file to 15 files.
3. Upload the *hit_data* file to a bucket/folder in Cloud Storage and rename it with a suffix denoting the date.
4. Invoke a BigQuery command - after installing the Cloud SDK - to load this file from Cloud Storage to BigQuery and convert it to a BigQuery table.
5. Repeat these steps for each day of the last 3-4 years, for both platforms. That is about 2200-2900 times.

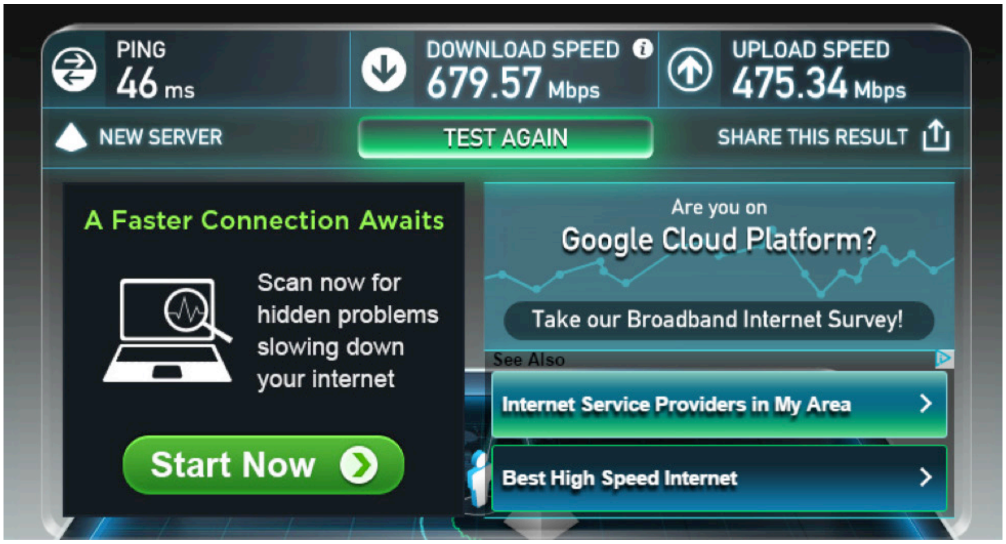
Unlike the GA schema in BigQuery, where all data is encapsulated in one table, Adobe uses a different approach and normalizes the data over 15 flat tables/files. These files can be joined with the main table/file, i.e. *hit_data*, by foreign keys.

The 14 lookup files were also converted to BigQuery tables in a manual step. We decided that using the latest version of these files is more efficient than uploading and converting all versions.

We spun up a Compute Engine, downloaded and installed the necessary software and ran the scripts.

Expediting the Process by Parallel Programming

Despite the high-speed connection one enjoys using a Google Compute Engine (we were so impressed with it, we took a screenshot and shared it), it was evident that it would take too long to finish the job.



PING 46 ms

DOWNLOAD SPEED 679.57 Mbps

UPLOAD SPEED 475.34 Mbps

NEW SERVER
TEST AGAIN
SHARE THIS RESULT
A Faster Connection Awaits
Scan now for hidden problems slowing down your internet
Start Now
Are you on Google Cloud Platform?
Take our Broadband Internet Survey!
See Also
Internet Service Providers in My Area
Best High Speed Internet

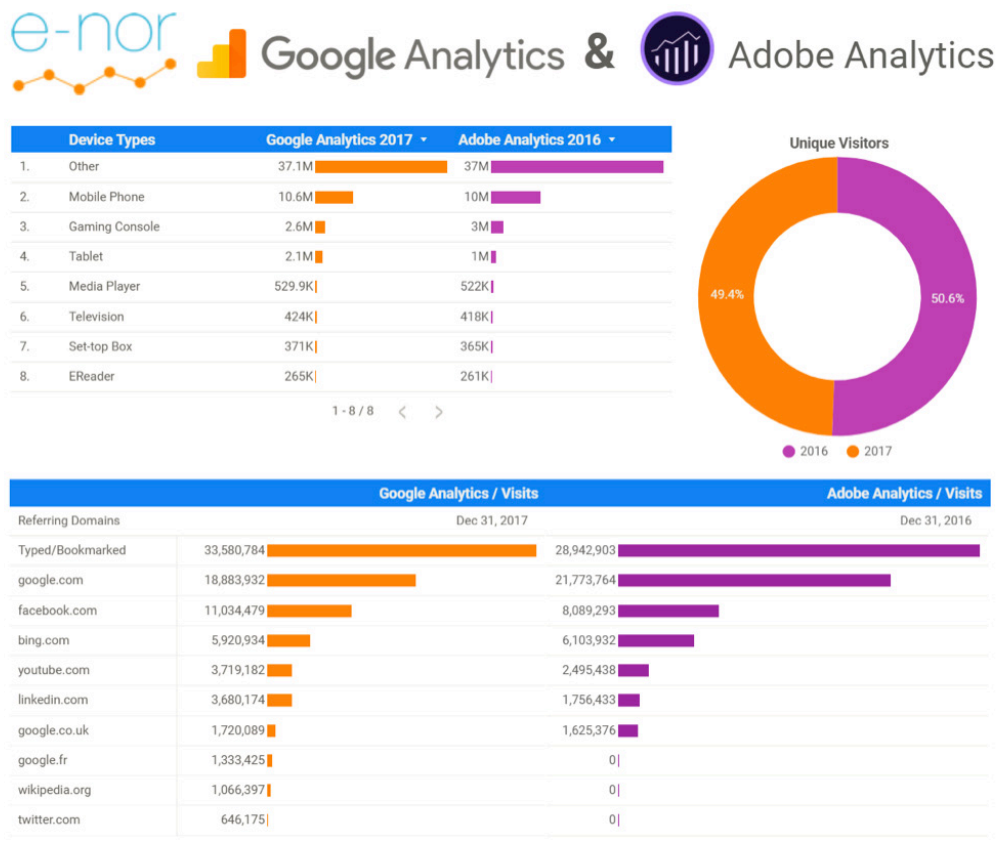
At that point, we resorted to a quick, if manual, implementation of an advanced concept: parallel programming. We spun up several Compute Engines from the same image to get the same software and scripts installed. Each assigned a different date range, the virtual machines worked in parallel, processing, uploading and converting files to BigQuery tables.

Of course, BigQuery, being a cloud-based, multi-tenant platform had no problem handling these requests. As did Cloud Storage. Once done, the virtual machines were *deleted* and we were only billed for the hours they were up and running.

Google Cloud SDK was the star in this project. It provided a very convenient and quick alternative to writing code that made use of the API. Without it, we would have had to spend long hours writing a custom application to utilize the BigQuery API, instead of a simple script for this one-time job.

Further Transformations to the Imported Raw Data:

After all files have been converted to BigQuery tables, it was easy to use BigQuery's power - and our knowledge of the Adobe Clickstream schema and GA schema - to process the imported data further and write other scripts to join the daily tables with the lookup tables and produce several intermediate/ aggregation tables.



	Device Types	Google Analytics 2017 ▾	Adobe Analytics 2016 ▾
1.	Other	37.1M	37M
2.	Mobile Phone	10.6M	10M
3.	Gaming Console	2.6M	3M
4.	Tablet	2.1M	1M
5.	Media Player	529.9K	522K
6.	Television	424K	418K
7.	Set-top Box	371K	365K
8.	EReader	265K	261K

1 - 8 / 8 < >

	Google Analytics / Visits	Adobe Analytics / Visits
Referring Domains	Dec 31, 2017	Dec 31, 2016
Typed/Bookmarked	33,580,784	28,942,903
google.com	18,883,932	21,773,764
facebook.com	11,034,479	8,089,293
bing.com	5,920,934	6,103,932
youtube.com	3,719,182	2,495,438
linkedin.com	3,680,174	1,756,433
google.co.uk	1,720,089	1,625,376
google.fr	1,333,425	0
wikipedia.org	1,066,397	0
twitter.com	646,175	0

Using intermediate tables created in BigQuery, this Data Studio report allows us to compare metrics from two different platforms from before and after a migration.

These tables made it possible to generate reports and dashboards to compare measurements, before and after the migration to Google Analytics 360.

Google Cloud Platform has many components that integrate well together and can be used to build robust large scale complex system or a one-time job, as this one.

Have you encountered any barriers when implementing a similar flow? Contact E-Nor today to learn how our Data Intelliegence Team can help.