How to get data from Google Analytics Reporting API with Python

When you ingest data from Google Analytics, you need to create a series of reports based on GA dimensions and metrics. The granularity is determined by dimensions you add in the report. The most important thing is to understand business requirements before start ingesting data. Good requirement analysis will enable you to drill up and down metrics at the right granularity and slice them with the dimensions that are critical to the business.

Google Analytics (for free version) only allows us to have up to 7 dimensions and 10 metrics in a single report. You also need to note that not all dimensions and metrics can be queried together. In my example, I created reports over the key metrics and grouped dimensions according to the GA Dimensions & Metrics Explorer (<https://developers.google.com/analytics/devguides/reporting/core/dimsmets>). For example, I created the ga\_dim\_geo table, which have the dimensions grouped under Geo Network. In my code, you can see how I grouped the same metrics against different dimension groups (like geo location, traffic source, device, channels and so on). GA has a really neat query tool (Query Explorer: https://ga-dev-tools.appspot.com/query-explorer/ ) where you can create ad hoc reports.

The python script uses the Core Reporting API v3.0 and OAuth2.0. The details on how to obtain API keys and the example scripts can be found here (<https://developers.google.com/analytics/devguides/reporting/core/v3/quickstart/service-py>).

The report comes in the JSON format. Many databases allow you to insert JSON file to a table. You can also fetch the json file with an ETL tool and do the conversion. In this example, the script converts JSON to CSV.

Key Points

The maximum number of records you can get from this API is 10000. When you are trying to do a historical load, you will easily exceed the limit. I am using the start\_index attribute for to iterate the API call until we get all the records.

You can easily add a report in the script by adding report\_name, dimensions and metrics in the main method. In this design, each report becomes a table.

Credentials and file paths need to be in the arguments when you call the script as below:

Python script.py <key file location e.g. /tmp/key/key.p12> <service-email> <json\_path e.g. /tmp/gadata/> <start-date e.g. 2017-11-01> <end-date e.g. 2017-11-02> <csv\_path e.g. /tmp/gadata/>

Here comes the code! Have fun!