Ad Performance Report



Ad Performance Report is an example of advanced reporting functionality provided by AdWords scripts. Advertisers like to analyze how their ads are performing in their campaigns. Sometimes, comparing how a given headline or destination URL performs against others will provide insight in creating new ads. Ad Performance Report generates a Google Spreadsheet with a number of interesting distribution charts.

A new Ad Performance report gets created whenever the script executes. You can access all of these reports in [Google Drive](https://drive.google.com/). Optionally, the script can also email the report to one or more recipients.



Scheduling

The script uses last week's statistics to generate the report. Schedule it **Weekly, on Mondays**.

How it works

The script starts off creating a copy of a [template spreadsheet](http://goo.gl/x0GDu0), with all graphs pre-configured. The script then populates the data values in the **Report** sheet, and graphs in the other sheets get constructed automatically.

Setup

* Create a new script with the source code below.
* Don't forget to update RECIPIENT\_EMAIL in order to specify your email preference.
* Schedule to run **Weekly, on Mondays**.

[Creating an AdWords script](https://developers.google.com/adwords/scripts/docs/solutions/ad-performance)

Source code

// Comma-separated list of recipients. Comment out to not send any emails.
var RECIPIENT\_EMAIL = 'email@example.com';

// URL of the default spreadsheet template
var SPREADSHEET\_URL = "https://docs.google.com/spreadsheet/ccc?key=0Agg6qWimbbe4dGhGUjlmQTV1R3h0aWRhSUV3LXpXaXc";

/\*\*
 \* This script computes an Ad performance report
 \* and outputs it to a Google spreadsheet
 \*/
function main() {
  var spreadsheet = copySpreadsheet(SPREADSHEET\_URL);
  var headlineSheet = spreadsheet.getSheetByName('Headline');
  headlineSheet.getRange(1, 2, 1, 1).setValue("Date");
  headlineSheet.getRange(1, 3, 1, 1).setValue(new Date());
  var destinationUrlSheet = spreadsheet.getSheetByName('Destination Url');
  destinationUrlSheet.getRange(1, 2, 1, 1).setValue("Date");
  destinationUrlSheet.getRange(1, 3, 1, 1).setValue(new Date());
  spreadsheet.getRangeByName("account\_id\_headline").setValue(AdWordsApp.currentAccount().getCustomerId());
  spreadsheet.getRangeByName("account\_id\_destination\_url").setValue(AdWordsApp.currentAccount().getCustomerId());

  outputSegmentation(headlineSheet, 'Headline', function(ad) {
    return ad.getHeadline();
  });
  outputSegmentation(destinationUrlSheet, 'Destination Url', function(ad) {
    return ad.getDestinationUrl();
  });
  Logger.log('Ad performance report available at\n' + spreadsheet.getUrl());
  if (RECIPIENT\_EMAIL) {
    MailApp.sendEmail(RECIPIENT\_EMAIL,
      'Ad Performance Report is ready',
      spreadsheet.getUrl());
  }
}

/\*\*
 \* Retrieves the spreadsheet identified by the URL.
 \* @param {string} spreadsheetUrl The URL of the spreadsheet.
 \* @return {SpreadSheet} The spreadsheet.
 \*/
function copySpreadsheet(spreadsheetUrl) {
  return SpreadsheetApp.openByUrl(spreadsheetUrl).copy('Ad Performance Report ' + new Date());
}

/\*\*
 \* Generates statistical data for this segment.
 \* @param {Sheet} sheet Sheet to write to.
 \* @param {string} segmentName The Name of this segment for the header row.
 \* @param {function(AdWordsApp.Ad): string} segmentFunc Function that returns
 \*        a string used to segment the results by.
 \*/
function outputSegmentation(sheet, segmentName, segmentFunc) {
  // Output header row
  var rows = [];
  var header = [
    segmentName,
    'Num Ads',
    'Impressions',
    'Clicks',
    'CTR (%)',
    'Cost'
  ];
  rows.push(header);

  var segmentMap = {};

  // Compute data
  var adIterator = AdWordsApp.ads()
      .forDateRange('LAST\_WEEK')
      .withCondition('Impressions > 0').get();
  while (adIterator.hasNext()) {
    var ad = adIterator.next();
    var stats = ad.getStatsFor('LAST\_WEEK');
    var segment = segmentFunc(ad);
    if (!segmentMap[segment]) {
      segmentMap[segment] = {
        numAds: 0,
        totalImpressions: 0,
        totalClicks: 0,
        totalCost: 0.0
      };
    }
    var data = segmentMap[segment];
    data.numAds++;
    data.totalImpressions += stats.getImpressions();
    data.totalClicks += stats.getClicks();
    data.totalCost += stats.getCost();
  }

  // Write data to our rows.
  for (var key in segmentMap) {
    if (segmentMap.hasOwnProperty(key)) {
      var ctr = 0;
      if (segmentMap[key].numAds > 0) {
        ctr = (segmentMap[key].totalClicks /
          segmentMap[key].totalImpressions) \* 100;
      }
      var row = [
        key,
        segmentMap[key].numAds,
        segmentMap[key].totalImpressions,
        segmentMap[key].totalClicks,
        ctr.toFixed(2),
        segmentMap[key].totalCost];
      rows.push(row);
    }
  }
  sheet.getRange(3, 2, rows.length, 6).setValues(rows);
}