Keyword Performance Report



Keyword Performance Report is an example of advanced reporting functionality provided by AdWords scripts. Advertisers like to understand how keywords are performing in their campaigns. Two important metrics for keyword performance are quality score and average position of ads triggered on Google search. Keyword Performance Report generates a Google Spreadsheet with a number of interesting distribution charts.

A new Keyword 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/27L9k8), 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/keyword-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=0Agg6qWimbbe4dE91MTEwekVDNzBMTkdZd2lncXJSOHc";

/\*\*
 \* This script computes a keyword performance report
 \* and outputs it to a Google spreadsheet. The spreadsheet
 \* url is logged and emailed.
 \*/
function main() {
  var spreadsheet = copySpreadsheet(SPREADSHEET\_URL);
  var sheet = spreadsheet.getSheetByName('Report');
  sheet.getRange(1, 2, 1, 1).setValue("Date");
  sheet.getRange(1, 3, 1, 1).setValue(new Date());
  spreadsheet.getRangeByName("account\_id").setValue(AdWordsApp.currentAccount().getCustomerId());
  outputQualityScoreData(sheet);
  outputPositionData(sheet);
  Logger.log('Keyword performance report available at\n' + spreadsheet.getUrl());
  if (RECIPIENT\_EMAIL) {
    MailApp.sendEmail(RECIPIENT\_EMAIL,
      'Keyword 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('Keyword Performance Report ' + new Date());
}

/\*\*
 \* Outputs Quality score related data to the spreadsheet
 \* @param {Sheet} sheet The sheet to output to.
 \*/
function outputQualityScoreData(sheet) {
  // Output header row
  var header = [
    'Quality Score',
    'Num Keywords',
    'Impressions',
    'Clicks',
    'CTR (%)',
    'Cost'
  ];
  sheet.getRange(3, 2, 1, 6).setValues([header]);

  // Initialize
  var qualityScoreMap = [];
  for (i = 1; i <= 10; i++) {
    qualityScoreMap[i] = {
      numKeywords: 0,
      totalImpressions: 0,
      totalClicks: 0,
      totalCost: 0.0
    };
  }

  // Compute data
  var keywordIterator = AdWordsApp.keywords()
      .forDateRange('LAST\_WEEK')
      .withCondition('Impressions > 0')
      .get();
  while (keywordIterator.hasNext()) {
    var keyword = keywordIterator.next();
    var stats = keyword.getStatsFor('LAST\_WEEK');
    var data = qualityScoreMap[keyword.getQualityScore()];
    if (data) {
      data.numKeywords++;
      data.totalImpressions += stats.getImpressions();
      data.totalClicks += stats.getClicks();
      data.totalCost += stats.getCost();
    }
  }

  // Output data to spreadsheet
  var rows = [];
  for (var key in qualityScoreMap) {
    var ctr = 0;
    var cost = 0.0;
    if (qualityScoreMap[key].numKeywords > 0) {
      ctr = (qualityScoreMap[key].totalClicks /
        qualityScoreMap[key].totalImpressions) \* 100;
    }
    var row = [
      key,
      qualityScoreMap[key].numKeywords,
      qualityScoreMap[key].totalImpressions,
      qualityScoreMap[key].totalClicks,
      ctr.toFixed(2),
      qualityScoreMap[key].totalCost];
    rows.push(row);
  }
  sheet.getRange(4, 2, rows.length, 6).setValues(rows);
}

/\*\*
 \* Outputs average position related data to the spreadsheet.
 \* @param {Sheet} sheet The sheet to output to.
 \*/
function outputPositionData(sheet) {
  // Output header row
  headerRow = [];
  var header = [
    'Avg Position',
    'Num Keywords',
    'Impressions',
    'Clicks',
    'CTR (%)',
    'Cost'
  ];
  headerRow.push(header);
  sheet.getRange(16, 2, 1, 6).setValues(headerRow);

  // Initialize
  var positionMap = [];
  for (i = 1; i <= 12; i++) {
    positionMap[i] = {
      numKeywords: 0,
      totalImpressions: 0,
      totalClicks: 0,
      totalCost: 0.0
    };
  }

  // Compute data
  var keywordIterator = AdWordsApp.keywords()
      .forDateRange('LAST\_WEEK')
      .withCondition('Impressions > 0')
      .get();
  while (keywordIterator.hasNext()) {
    var keyword = keywordIterator.next();
    var stats = keyword.getStatsFor('LAST\_WEEK');
    if (stats.getAveragePosition() <= 11) {
      var data = positionMap[Math.ceil(stats.getAveragePosition())];
    } else {
      // All positions greater than 11
      var data = positionMap[12];
    }
    data.numKeywords++;
    data.totalImpressions += stats.getImpressions();
    data.totalClicks += stats.getClicks();
    data.totalCost += stats.getCost();
  }

  // Output data to spreadsheet
  var rows = [];
  for (var key in positionMap) {
    var ctr = 0;
    var cost = 0.0;
    if (positionMap[key].numKeywords > 0) {
      ctr = (positionMap[key].totalClicks /
        positionMap[key].totalImpressions) \* 100;
    }
    var row = [
      key <= 11 ? key - 1 + ' to ' + key : '>11',
      positionMap[key].numKeywords,
      positionMap[key].totalImpressions,
      positionMap[key].totalClicks,
      ctr.toFixed(2),
      positionMap[key].totalCost
    ];
    rows.push(row);
  }
  sheet.getRange(17, 2, rows.length, 6).setValues(rows);
}