Degrading Ad Groups Report



AdWords scripts can easily access statistics over multiple date ranges, and can therefore compare performance of campaigns through time.

Degrading Ad Groups Report fetches ad groups whose performance we consider to be worsening:

* Ad groups are **ENABLED** and belong to **ENABLED** campaigns, so we know they're serving.
* Ad groups' Click Through Rate has been decreasing for three consecutive weeks.

Obviously, a more sophisticated measure of "worsening" may be developed.

The script outputs the data into a spreadsheet that looks like this:



Scheduling

The script uses three weeks worth of statistics to generate the report. It would make sense to schedule it **Weekly**.

Setup

* Setup a spreadsheet-based script with the source code below. Use Degrading Ad Groups Report [template spreadsheet](http://goo.gl/pbhJ1q)
* Don't forget to update SPREADSHEET\_URL in code.
* Schedule the script **Weekly**.

[Setting up a spreadsheet-based script](https://developers.google.com/adwords/scripts/docs/solutions/degrading-adgroups)

Source code

var SPREADSHEET\_URL = "[YOUR\_URL]";

function main() {
  var spreadsheet = SpreadsheetApp.openByUrl(SPREADSHEET\_URL);
  var sheet = spreadsheet.getSheets()[0];
  spreadsheet.getRangeByName("account\_id").setValue(AdWordsApp.currentAccount().getCustomerId());
  sheet.getRange(1, 2, 1, 1).setValue("Date");
  sheet.getRange(1, 3, 1, 1).setValue(new Date());
  sheet.getRange(7, 1, sheet.getMaxRows() - 7, sheet.getMaxColumns()).clear();

  var adGroupsIterator = AdWordsApp.adGroups()
      .withCondition("Status = 'ENABLED'")
      .withCondition("CampaignStatus = 'ENABLED'")
      .forDateRange("LAST\_7\_DAYS")
      .orderBy("Ctr ASC")
      .withLimit(100)
      .get();

  var today = getDateInThePast(0);
  var oneWeekAgo = getDateInThePast(7);
  var twoWeeksAgo = getDateInThePast(14);
  var threeWeeksAgo = getDateInThePast(21);

  var reportRows = [];

  while (adGroupsIterator.hasNext()) {
    var adGroup = adGroupsIterator.next();
    // Let's look at the trend of the ad group's CTR.
    var statsThreeWeeksAgo = adGroup.getStatsFor(threeWeeksAgo, twoWeeksAgo);
    var statsTwoWeeksAgo = adGroup.getStatsFor(twoWeeksAgo, oneWeekAgo);
    var statsLastWeek = adGroup.getStatsFor(oneWeekAgo, today);

    // Week over week, the ad group is degrading - record that!
    if (statsLastWeek.getCtr() < statsTwoWeeksAgo.getCtr() && statsTwoWeeksAgo.getCtr() < statsThreeWeeksAgo.getCtr()) {
      reportRows.push([adGroup.getCampaign().getName(), adGroup.getName(),
          statsLastWeek.getCtr() \* 100, statsLastWeek.getCost(),
          statsTwoWeeksAgo.getCtr() \* 100, statsTwoWeeksAgo.getCost(),
          statsThreeWeeksAgo.getCtr() \* 100, statsThreeWeeksAgo.getCost()]);
    }
  }
  if (reportRows.length > 0) {
    sheet.getRange(7, 2, reportRows.length, 8).setValues(reportRows);
    sheet.getRange(7, 4, reportRows.length, 1).setNumberFormat("#0.00%");
    sheet.getRange(7, 6, reportRows.length, 1).setNumberFormat("#0.00%");
    sheet.getRange(7, 8, reportRows.length, 1).setNumberFormat("#0.00%");

    sheet.getRange(7, 5, reportRows.length, 1).setNumberFormat("#,##0.00");
    sheet.getRange(7, 7, reportRows.length, 1).setNumberFormat("#,##0.00");
    sheet.getRange(7, 9, reportRows.length, 1).setNumberFormat("#,##0.00");
  }

  var email = spreadsheet.getRangeByName("email").getValue();
  if (email) {
    var body = [];
    body.push("The Ctr of the following ad groups is degrading over the last three weeks.\n");
    body.push("Full report at " + SPREADSHEET\_URL + "\n\n");
    for (var i = 0; i < reportRows.length; i ++) {
      body.push(reportRows[i][0] + " > " + reportRows[i][1]);
      body.push("  " + ctr(reportRows[i][6]) + " > " + ctr(reportRows[i][4]) + " > " + ctr(reportRows[i][2]) + "\n");
    }
    MailApp.sendEmail(email, "" +
        reportRows.length + " ad groups are degrading in AdWords account " +
        AdWordsApp.currentAccount().getCustomerId(), body.join("\n"));
  }
}

function ctr(number) {
  return parseInt(number \* 10000) / 10000 + "%";
}
// Returns YYYYMMDD-formatted date.
function getDateInThePast(numDays) {
  var today = new Date();
  today.setDate(today.getDate() - numDays);
  return Utilities.formatDate(today, "PST", "yyyyMMdd");
}