
```
/**

 * Created by Francesco Cifardi

 * For issues or questions reachout on Linkedin (https://www.linkedin.com/in/francescocifardi/)

 * Free to use and share, a mention on Linkedin would be appreciated :-)

 * Looking to get new script updates via email? Subscribe to my Free Google Ads Scripts Newsletter. No content, no regular emails, only scripts and other automation tools whenever they are ready.

 * https://medialauncher.beehiiv.com/subscribe

 * Pmax Assets & Groups Performance Tracker v1 (Single Account)

 *

 * Why Use This Script:

 * Performance Max campaigns often lack transparency. It's especially the case with assets performance data,

 * where Google only provides labels (Low, Good, Best). This script addresses this limitation by tracking

 * performance labels over time, giving them a numeric value and then visualising this "Average Score" on a timeline.

 * So, by using this script, you can:

 *- Track asset performance history, which is not natively available in Google Ads

 *- Visualize individual asset group performance trends over time

 *- Understand how changes you make to assets affect the asset group

 *- Make data-driven decisions about asset optimization

 *- Adapt your creative strategy based on performance data

 *- Save time on manual data collection and analysis

 *- (Coming soon) Quickly identify high-performing and underperforming individual assets
```

*

* What the Script Does:

* This script generates a comprehensive report of your Performance Max campaigns, focusing on asset and

* asset group performance. Here's what it provides:

* - A complete list of all enabled asset groups in your Performance Max campaigns

* - Daily performance data for each asset group, including impressions, clicks, CTR, cost, conversions, and conversion value

* - An average performance score for assets, allowing you to track performance trends over time

* - An interactive dashboard to easily compare asset performance with other metrics

*

* The script uses the last 30 days of data to ensure that performance analysis is based on recent trends.

*

* How It Works:

* - The script connects to your Google Ads account and retrieves data for all enabled Performance Max campaigns

* and their asset groups from the past 30 days.

* - It collects daily performance data for each asset group, including key metrics like impressions, clicks, and conversions.

* - For assets, it gathers the performance label (Low, Good, Best, Learning) assigned by Google Ads each day.

* - The script calculates an average performance score for assets by assigning numerical values to the labels

* (Low = 1, Good = 2, Best = 3, Learning/Unknown = 0) and averaging these over time.

* - All this data is then populated into a Google Sheets Dashboard for easy viewing and analysis.

*

* Note: The script needs to run for a few days to collect meaningful average score data for assets,

* as it tracks changes in performance labels over time and those cannot be backtracked.

*

* How to Use It:

* 1. First, make a copy of this template spreadsheet: https://docs.google.com/spreadsheets/d/125NjWOAG7_a4e52pkdtM08y_umALPB2hmZt0Pw939_M/copy

* 2. In your Google Ads account, create a new script and paste the code provided in this doc:

* https://docs.google.com/document/d/1LDwBfvFMuaTRW19NSZBcUTldpF7BTFPzu58CZQ4bI50/edit

* 3. Replace the YOUR_SHEET_URL value in the spreadsheetUrl variable in the script with the URL of your copy of the template.

* Make sure the URL is wrapped around only two 'single' quotes or only two "double" quotes.

* 4. Set up the script to run daily.

* 5. Review the dashboard and data, paying special attention to:

* - Spikes/dips in assets Avg Score

* - The relationship between asset performance and overall asset group performance

* - Asset group performance in general via the line chart visualization

*

* Use this data to inform your Performance Max optimization decisions. You might choose to:

* - Pause/duplicate high-performing asset groups

* - Create new assets based on top-performing groups

* - Pause or replace consistently low-performing assets (coming soon)

* - Investigate asset groups with unusual metrics to identify opportunities or issues

*

* By regularly running this script and acting on its insights, you can optimize your Performance Max campaigns assets,

* improving their groups overall performance. Remember, while the script provides data-driven insights,

```
 * always use your expertise and knowledge of your business goals when making final
optimization decisions.

 */

function main() {
  const spreadsheetUrl = 'YOUR_SHEET_URL'; // Make a copy:
https://docs.google.com/spreadsheets/d/125NjWOAG7_a4e52pkdtM08y_umALPB2hmZt0Pw93
9_M/copy
  const spreadsheet = SpreadsheetApp.openByUrl(spreadsheetUrl);

  // Handle 'ag' sheet (Asset Group Performance)
  let agSheet = spreadsheet.getSheetByName('ag');
  if (!agSheet) {
    agSheet = spreadsheet.insertSheet('ag');
  } else {
    agSheet.clear();
  }

  const agHeaders = ['Date', 'Campaign', 'Asset Group', 'Impressions', 'Clicks', 'CTR', 'Cost',
'Conversions', 'Conv. Value'];
  agSheet.getRange(1, 1, 1, agHeaders.length).setValues([agHeaders]);

  // Handle 'ap' sheet (Asset Performance)
  let apSheet = spreadsheet.getSheetByName('ap');
  if (!apSheet) {
    apSheet = spreadsheet.insertSheet('ap');
    const apHeaders = ['Date', 'Campaign', 'Asset Group', 'Asset', 'Type', 'Performance Label',
'Performance Value'];
    apSheet.getRange(1, 1, 1, apHeaders.length).setValues([apHeaders]);
  }

  // Clean up old data in 'ap' sheet
  cleanupOldData(apSheet);

  const pmaxCampaigns = getPmaxCampaigns();
  let allAgData = [];
  let allApData = [];

  for (const campaign of pmaxCampaigns) {
    const campaignName = campaign.getName();
    const campaignId = campaign.getId();
    const agData = getAssetGroupPerformance(campaignId, campaignName);
    allAgData = allAgData.concat(agData);
```

```
    const apData = getAssetPerformance(campaignId, campaignName);
    allApData = allApData.concat(apData);
  }

  // Sort and write Asset Group Performance data
  allAgData.sort((a, b) => b[0].localeCompare(a[0]));
  if (allAgData.length > 0) {
    agSheet.getRange(2, 1, allAgData.length, allAgData[0].length).setValues(allAgData);
    agSheet.getRange(2, 6, allAgData.length, 1).setNumberFormat('0.00%');  // Format CTR as
percentage
    agSheet.getRange(2, 7, allAgData.length, 1).setNumberFormat('"$"#,##0.00');  // Format Cost
    agSheet.getRange(2, 9, allAgData.length, 1).setNumberFormat('"$"#,##0.00');  // Format
Conv. Value
  }

  // Append Asset Performance data
  if (allApData.length > 0) {
    const lastRow = apSheet.getLastRow();
    apSheet.getRange(lastRow + 1, 1, allApData.length,
allApData[0].length).setValues(allApData);
  }
}

function cleanupOldData(sheet) {
  const data = sheet.getDataRange().getValues();
  const headers = data.shift(); // Remove and store headers

  if (data.length === 0) return; // No data to clean up

  const today = new Date();
  const thirtyOneDaysAgo = new Date(today.getTime() - 31 * 24 * 60 * 60 * 1000);

  const newData = data.filter(row => {
    const rowDate = new Date(row[0]); // Assuming date is in the first column
    return rowDate >= thirtyOneDaysAgo;
  });

  // Clear the sheet and rewrite the data
  sheet.clear();
  sheet.getRange(1, 1, 1, headers.length).setValues([headers]);
  if (newData.length > 0) {
    sheet.getRange(2, 1, newData.length, newData[0].length).setValues(newData);
  }
}
```

```
function getPmaxCampaigns() {
  const query = "SELECT campaign.id, campaign.name FROM campaign WHERE
campaign.advertising_channel_type = 'PERFORMANCE_MAX' AND campaign.status =
'ENABLED'";
  const report = AdsApp.report(query);
  const rows = report.rows();
  const campaigns = [];

  while (rows.hasNext()) {
    const row = rows.next();
    campaigns.push({
      getId: () => row['campaign.id'],
      getName: () => row['campaign.name']
    });
  }

  return campaigns;
}

function getAssetGroupPerformance(campaignId, campaignName) {
  const today = new Date();
  const yesterday = new Date(today.getTime() - 24 * 60 * 60 * 1000);
  const thirtyDaysAgo = new Date(yesterday.getTime() - 29 * 24 * 60 * 60 * 1000);

  const startDate = Utilities.formatDate(thirtyDaysAgo, AdsApp.currentAccount().getTimeZone(),
'yyyy-MM-dd');
  const endDate = Utilities.formatDate(yesterday, AdsApp.currentAccount().getTimeZone(),
'yyyy-MM-dd');

  const query = `
    SELECT
      segments.date,
      asset_group.name,
      metrics.impressions,
      metrics.clicks,
      metrics.cost_micros,
      metrics.conversions,
      metrics.conversions_value
    FROM asset_group
    WHERE
      campaign.id = ${campaignId}
      AND asset_group.status = 'ENABLED'
      AND segments.date BETWEEN '${startDate}' AND '${endDate}'
```

```
  ORDER BY asset_group.name, segments.date
`;

const report = AdsApp.report(query);
const rows = report.rows();
const performanceData = {};
const assetGroups = new Set();

while (rows.hasNext()) {
  const row = rows.next();
  const date = row['segments.date'];
  const assetGroup = row['asset_group.name'];
  const metrics = {
    impressions: parseInt(row['metrics.impressions'], 10),
    clicks: parseInt(row['metrics.clicks'], 10),
    cost: parseFloat(row['metrics.cost_micros']) / 1000000,
    conversions: parseFloat(row['metrics.conversions']),
    conversionValue: parseFloat(row['metrics.conversions_value'])
  };

  if (!performanceData[assetGroup]) {
    performanceData[assetGroup] = {};
  }
  performanceData[assetGroup][date] = metrics;
  assetGroups.add(assetGroup);
}

const data = [];

for (const assetGroup of assetGroups) {
  for (let d = new Date(thirtyDaysAgo); d <= yesterday; d.setDate(d.getDate() + 1)) {
    const dateString = Utilities.formatDate(d, AdsApp.currentAccount().getTimeZone(),
'yyyy-MM-dd');
    const metrics = performanceData[assetGroup][dateString] || {
      impressions: 0,
      clicks: 0,
      cost: 0,
      conversions: 0,
      conversionValue: 0
    };
    const ctr = metrics.impressions > 0 ? metrics.clicks / metrics.impressions : 0;
    data.push([
      dateString,
      campaignName,
```

```
        assetGroup,
        metrics.impressions,
        metrics.clicks,
        ctr,
        metrics.cost,
        metrics.conversions,
        metrics.conversionValue
      ]);
    }
  }

  return data;
}

function getAssetPerformance(campaignId, campaignName) {
  const yesterday = Utilities.formatDate(new Date(new Date().getTime() - 24 * 60 * 60 * 1000),
AdsApp.currentAccount().getTimeZone(), 'yyyy-MM-dd');
  const query = `
    SELECT
      asset_group.name,
      asset_group_asset.field_type,
      asset_group_asset.asset,
      asset_group_asset.performance_label
    FROM asset_group_asset
    WHERE
      campaign.id = ${campaignId}
      AND asset_group.status = 'ENABLED'
  `;

  const report = AdsApp.report(query);
  const rows = report.rows();
  const data = [];

  while (rows.hasNext()) {
    const row = rows.next();
    const assetGroup = row['asset_group.name'];
    const assetType = row['asset_group_asset.field_type'];
    const asset = row['asset_group_asset.asset'];
    const performanceLabel = row['asset_group_asset.performance_label'];
    const performanceValue = getPerformanceValue(performanceLabel);

    data.push([
      yesterday,
      campaignName,
```

```
      assetGroup,
      asset,
      assetType,
      performanceLabel,
      performanceValue
    ]);
  }

  return data;
}

function getPerformanceValue(label) {
  switch (label) {
    case 'LOW':
      return 1;
    case 'GOOD':
      return 2;
    case 'BEST':
      return 3;
    case 'LEARNING':
    case 'UNKNOWN':
    default:
      return 0;
  }
}
```