

Work out what each weekday normally costs - a brief for your AI assistant

Hand this whole file to your assistant along with the notebook from <https://www.stochasticstrata.com/blog/is-it-the-day-of-the-week>. It tells the assistant what to do and what to refuse.

For the assistant: what you are being asked to do

The person you are helping wants to know what each day of the week normally costs them, so they can tell a genuinely bad day from a day behaving exactly as that weekday always behaves. They may not write code. Do the technical parts yourself and walk them through one step at a time. Ask for one thing, wait, then ask for the next.

- Adapt the query below to their warehouse, or tell them which export to download and which columns to keep.
- Map their columns onto the two the notebook expects.
- Set the one file path in the notebook and run it for them if you can.
- Read the refusals out loud. They are the most useful output in the file.

The two ways in

EITHER a ready-made daily file: one row per day, a date column and one positive number. OR a raw export you consolidate first - group it to one row per calendar day before anything else. One campaign at a time, always: a weekday pattern averaged across campaigns that do not share a rhythm is a pattern nobody has.

The metric must be a rate or a cost - cost per sale, cost per click, conversion rate, return on spend. NOT raw conversion counts at low volume. A Tuesday with two sales and a Wednesday with one are not thirty-three per cent apart in any sense worth acting on, and this method will say they are. And a cost per conversion on a low-volume account has days with no conversion: never drop them. Give the notebook the cost and the conversions as two columns and it reads each weekday off its totals; the notebook also warns when the level moved across the window.

A query to adapt

```
-- One row per day for ONE campaign. Adapt the table name, the date column and the
-- metric to the reader's own warehouse. The only hard requirements are: one row per
-- calendar day, a positive number, and a single campaign - not several averaged together.
-- Keep cost and conversions as two columns: on a day with no conversion the cost per
-- conversion does not exist, and dropping those days biases the pattern. The notebook reads
-- each weekday off its totals when COST_COLUMN and CONVERSIONS_COLUMN are set.
SELECT
    event_date                                AS date,
    SUM(cost)                                AS cost,
    SUM(conv)                                AS conversions
FROM `project.dataset.campaign_daily`
WHERE campaign_name = 'THE ONE CAMPAIGN'
      AND event_date BETWEEN DATE_SUB(CURRENT_DATE(), INTERVAL 98 DAY)
                        AND DATE_SUB(CURRENT_DATE(), INTERVAL 8 DAY) -- the newest week is still filling in
GROUP BY event_date
ORDER BY event_date;
```

How to run it

- Local: open the notebook, set the file path in the marked cell, run all.
- Colab: upload the notebook, upload their file, set the path, run all. Tell them plainly that this puts their file on Google's servers. If that is not acceptable, use the local route.
- Nothing else leaves their machine on the local route. Say so - people ask.

Can you just do the arithmetic instead? Yes - and that is measured

This method is one ratio: each weekday's average divided by the average of every day in the window. A value of 1.10 means that weekday runs ten per cent above the account's own average.

A rebuild of that formula from this page alone, with no sight of the notebook, reproduces the notebook's own answer on all 7 weekdays, worst difference 4.6e-05. So this brief does not tell you that you cannot reproduce it by hand. You can, exactly.

So why run the notebook at all

Because the formula answers every question you ask it, including the ones it has no business answering. Measured on a series built with NO weekday rhythm at all - every honest answer is 1.000 - a hand rebuild of the formula reports a spread of 0.231 across the week, with Monday at 0.861. The account was simply drifting. Nothing in the arithmetic can tell those apart. The notebook's check catches a drift this big: it warns when the second half of the window runs 15 per cent or more off the first. A slower drift gets no warning, and still tilts the week.

- It refuses a metric with zeroes or negatives in it, rather than dividing by an average that means nothing.
- It refuses duplicate dates, which is what a multi-campaign export looks like.
- It gives no number for a weekday that never appears, instead of quietly leaving it out of the picture.
- It says which reading the data can support, which is not the same as whether it produced numbers.

What to tell them about how much history they need

There is no week count at which this becomes correct - the error keeps shrinking. What matters is how fine a distinction they are drawing. Telling the weekend from the working week took 1 week of data to get right nineteen times in twenty. Telling one weekday from its neighbour, where the real gap was 3.0 per cent, was still wrong 15 per cent of the time after 14 weeks.

If they want to shift budget between two neighbouring weekdays, that is the number to quote at them.

The honest caveats - say these BEFORE they act

- A drifting account manufactures a weekend effect. Mondays sit earlier in every week than Sundays do, so a climbing level shows up as a tilt across the week that looks exactly like a real weekend premium.
- One strange week corrupts the pattern. Promotions, outages and tracking breaks land inside one weekday's cell. Take those periods out of the window first, not after reading the answer.
- Checking the pattern repeats next month is weaker evidence than it sounds. Measured with 6 weeks on each side, that check caught only 20 per cent of accounts whose weekend premium really had collapsed. When it fails, believe it. When it passes, it means nothing large enough to see.
- It finds what a weekday costs. It never finds why. Whether Sunday is dearer because of the audience, the auction or the landing page is not in these numbers.
- An expensive day can still be a profitable day. This says what a day costs, not whether it pays.

A number to check yourself against

In the article's own synthetic account, a normal Sunday costs 29.55 and a normal Tuesday 24.14 - a gap of 5.41 with nothing wrong at all. An ordinary Sunday at 30.14 reads as 19 per cent worse than the account average and is worth nothing; a genuinely bad Tuesday at 29.46 reads as only 16 per cent worse and is real. If your rebuild reproduces those, you have it right.

Synthetic data throughout the article and notebook. The method transfers; the numbers do not - which is why the notebook reads the pattern off the reader's own figures rather than shipping one.