ChatGPT

Excel Dashboard Automation Agent

Purpose

This document acts as an internal **agent charter**. It outlines the requirements and workflow for rebuilding analytical dashboards in Excel programmatically, based on the learnings from a recruitment test. It also includes references to industry trends around data automation and AI to contextualise why automation is critical.

1. Background and Motivation

In the test workbook (`Test_Excel_Recrutement_v3.0.xlsx`), a dashboard summarising doctor visits for 2020 needed to be populated with formulas to compute metrics such as visit counts, doctor coverage and speciality focus. The raw data lived in separate sheets (`Données visites` and `Données médecins`). However, the dashboard contained either incorrect formulas or static values, leading to calculation errors. The automation agent was tasked with restoring the formulas so they adapt to future data updates.

Beyond the specific test, automation is increasingly important for modern analytics teams. Splunk's 2025 overview of data analysis tools notes that companies handle huge volumes of data and are turning to a spectrum of tools from spreadsheets to programming languages [1]. The future of analytics is moving toward **increased automation and AI integration**, making advanced analytics more accessible and allowing organisations to surface insights faster [2]. Python remains a versatile favourite for ETL, visualisation and machine learning [3], while Excel remains a robust staple for data manipulation and pivot tables [4].

2. Root Causes of Failures

During the manual test, several issues were discovered:

Issue	Cause	Effect
Missing December formulas	Code iterated over columns D–N, covering only January to November	December (column O) remained blank, so totals were off
`#VALUE!` **errors**	Patient potential column contained a mix of numbers and text	`SUMPRODUCT` failed when multiplying numeric and text types
Coverage >100% or 0%	Distinct doctor counts were not filtered by region	The numerator and denominator did not align, producing nonsensical coverage

Issue	Cause	Effect
Unmatched visits	Some visit IDs lacked a match in the doctor sheet	`MATCH` returned `#N/A`, breaking formulas

3. Solutions Implemented

3.1 Inclusive Monthly Loop

The script originally iterated `range(4, 15)`, mapping to Excel columns D–N. To include December (column O), the range was changed to `range(4, 16)`.

3.2 Numeric Coercion

Excel's `N()` function converts values to numbers: numbers remain unchanged, dates become their serial number, logical TRUE becomes 1 and all other types return 0 [5]. Wrapping numeric ranges in `N()` within `SUMPRODUCT` prevents `#VALUE!` when text appears in numeric columns:

```
=SUMPRODUCT(ISNUMBER(MATCH(Secteurs,{"S1","S2","S3","S4","S5","S6"},0)) *
N(Patients))
```

3.3 Region-Filtered Distinct Counts

Coverage metrics require counting distinct doctor IDs in a region relative to the total eligible population. The fix constructed filtered arrays using `IF()` conditions and then applied `COUNTIF` within those arrays. A simplified pattern:

```
\=IFERROR(
  SUMPRODUCT(
    (region_filter*sector_filter*date_filter)*(visit_ids<>"") /
    COUNTIF(IF(region_filter*sector_filter*date_filter,visit_ids,""),visit_ids)
  ) /
  SUMPRODUCT((doctor_region="Nord")*sector_filter),
0)
```

3.4 Handling Unmatched IDs

`MATCH()` functions were wrapped in `IFERROR(..., "")` to handle visits with missing doctor matches gracefully. A future improvement is to clean data upstream using Power Query or Python before formula injection.

4. Automation Workflow

4.1 Sheet Discovery

1. Load the workbook using openpyxl.
2. Find sheets with names containing keywords (e.g. “visites”, “médecins”, “dashboard”). If names differ, the agent can be configured via constants.
3. Read the header row (row 3) of each data sheet and map column names to variables such as `vis_date`, `vis_mode`, `vis_dur`, `med_region`, `med_sect`, `med_spec1`, etc.

4.2 Formula Templates

Each metric is defined as a string template with placeholders for column references. For example, monthly visit counts for a given sector:

```
FORMULA_VISITS_MONTH = (
    "=SUMPRODUCT("
    "ISNUMBER(MATCH({vis_sect},{{'S1','S2','S3','S4','S5','S6'}},0)) *"
    "({vis_date}>=DATE(2020,{month},1))*({vis_date}<EDATE(DATE(2020,{month},1),
1))"
    ")"
)
```

When injecting formulas into the dashboard, the agent uses the column mapping to substitute each placeholder with the actual Excel cell references (e.g. `'Données visites'!$H$4:$H$627`) and writes the final formula string to the target cell.

4.3 Error Handling

- **Numeric coercion:** always wrap numeric arrays in `N()` inside `SUMPRODUCT` to avoid `#VALUE!` errors when cells contain text.
- **Missing matches:** wrap lookup functions in `IFERROR` and return `""` to avoid breaking downstream calculations.
- **Inclusive ranges:** loops over months should include the final column (e.g. December) by using inclusive `range` boundaries.
- **Date filters:** use `DATE(year,month,day)` and `EDATE()` to build month-boundaries in formulas.

5. Best Practices

1. **Use header names** rather than hard-coded column letters in Python. This makes the script resilient to reorderings.
2. **Filter first, then count**. When computing coverage, apply all filters (region, sector, date) before counting distinct IDs.
3. **Validate formulas with openpyxl**. After injecting formulas, re-open the workbook with `data_only=True` and compare computed values against known correct results.

4. **Document logic**. Keep an `agents.md` file like this one to record formula templates, assumptions and known issues for future maintainers.
5. **Integrate AI judiciously**. According to Splunk, the future of data analysis is moving toward automation and AI integration [2]. Tools like Python with pandas or Excel with Copilot can help, but data quality and reproducibility remain paramount [6]. Always validate AI-generated code and formulas.

6. Future Enhancements

- **Power Query/Pandas pre-processing** – unify identifiers and clean data before formulas are applied.
- **PowerPoint automation** – build a script (e.g. `create_ppt.py`) using `python-pptx` to generate slides summarising the dashboard. Python-powered slide creation can reduce the time spent manually building decks and has been identified as an effective way to turn analyses into engaging presentations [7]. [8].
- **AI augmentation** – integrate generative AI (e.g. ChatGPT) to propose narratives for reports or help construct complex formulas. However, keep humans in the loop to review and validate outputs.

Document Information

Document version: 1.0
Maintained by: Codyssey
Last updated: 2025-10-18

---

[1] [2] [3] [4] 12 Must-Have Data Analysis Tools for 2025 | Python, SQL & AI | Splunk
https://www.splunk.com/en_us/blog/learn/data-analysis-tools.html

[5] N function - Microsoft Support
https://support.microsoft.com/en-us/office/n-function-a624cad1-3635-4208-b54a-29733d1278c9

[6] The 2025 State of Analytics Engineering Report, crafted by dbt Labs | dbt Labs
https://www.getdbt.com/resources/state-of-analytics-engineering-2025

[7] 7 Ways to Boost Productivity with Python PowerPoint Automation
https://www.softkraft.co/python-powerpoint-automation/

[8] How To Automate PowerPoint Slides From Excel Using Python and ChatGPT — Analythical by Stephen Tracy
https://analythical.com/blog/automating-powerpoint-slides-with-charts