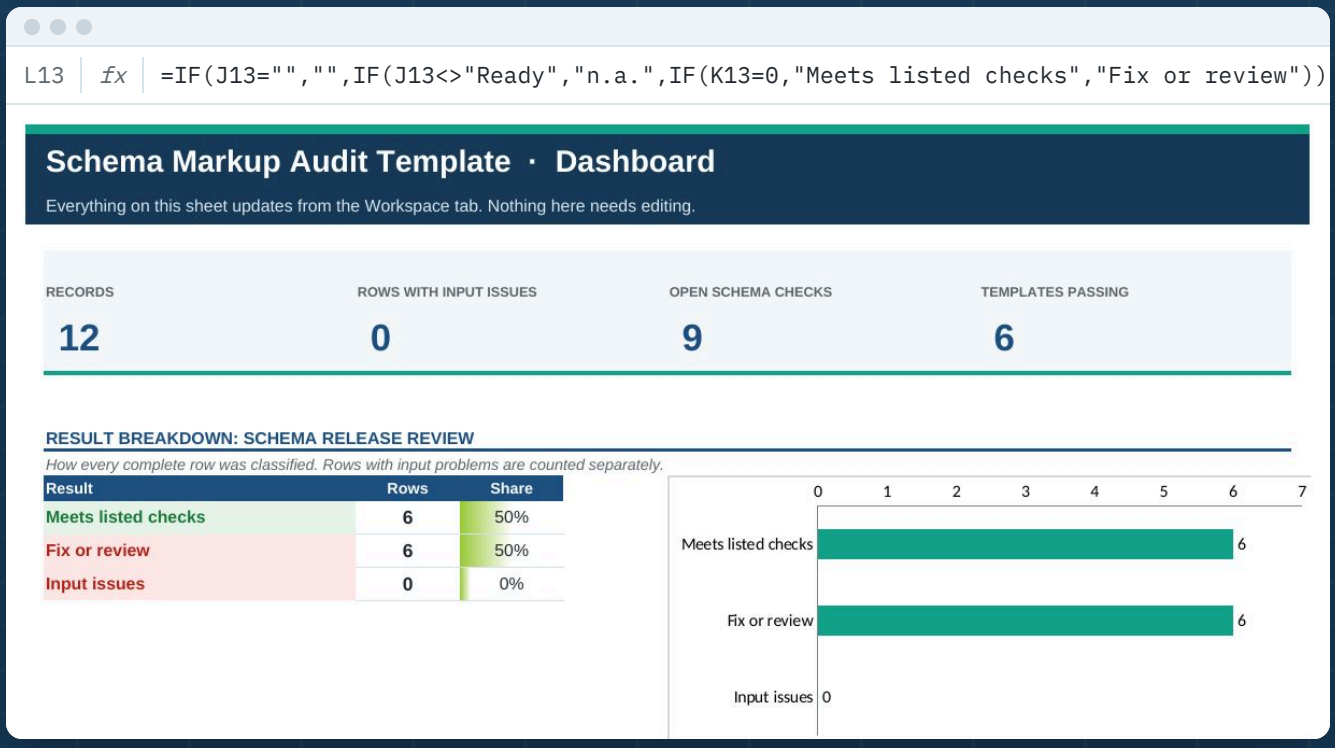
OptimizationTheory.com

Technical SEO Toolkit, resource 18

Schema Markup Audit Template

Check presence, syntax, required fields, visible-content match and policy for every structured data type you ship.

Built for the DataForSEO API: On-Page API



L13 | *fx* | =IF(J13="","",IF(J13<>"Ready","n.a.",IF(K13=0,"Meets listed checks","Fix or review"))

Schema Markup Audit Template · Dashboard

Everything on this sheet updates from the Workspace tab. Nothing here needs editing.

RECORDS	ROWS WITH INPUT ISSUES	OPEN SCHEMA CHECKS	TEMPLATES PASSING
12	0	9	6

RESULT BREAKDOWN: SCHEMA RELEASE REVIEW

How every complete row was classified. Rows with input problems are counted separately.

Result	Rows	Share
Meets listed checks	6	50%
Fix or review	6	50%
Input issues	0	0%

Jourdan Rombough, optimizationtheory.com

What this template decides

Which pages and schema types are ready to release, and which fail on syntax, required properties, content mismatch or policy.

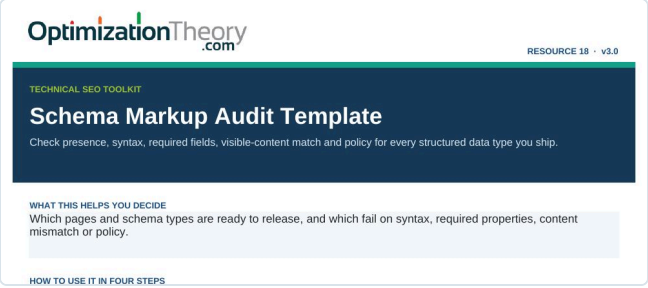
Structured data fails in three ways: it's broken (syntax errors), incomplete (missing required properties), or untrue (markup says something the page doesn't). The third is the one that gets sites into trouble, and validators can't catch it.

This schema markup audit template checks each URL and schema type against five conditions: markup present, syntax valid, required fields present, values match visible content, and a policy review. It counts failed checks and marks each Meets listed checks or Fix or review, so you have a release register, not just a validator screenshot.

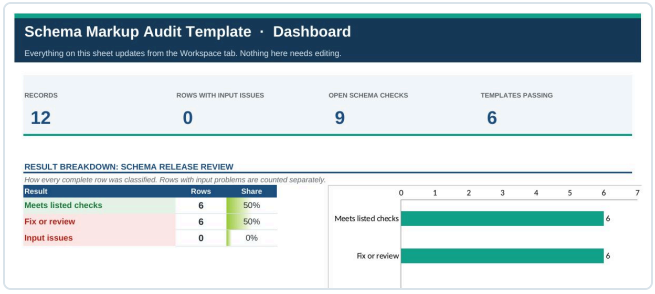
Who it's for

- SEOs and developers shipping JSON-LD templates
- Ecommerce teams with Product, Offer and Review markup
- Publishers using Article and Organization markup
- QA leads who need a sign-off record

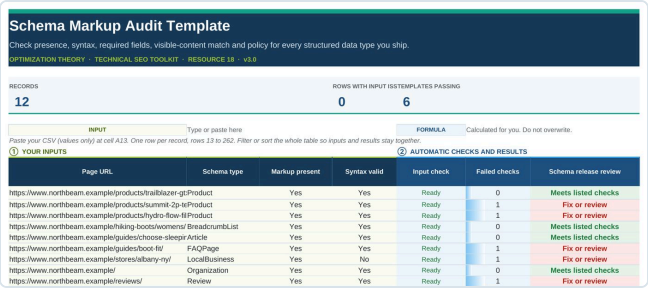
What's in the workbook



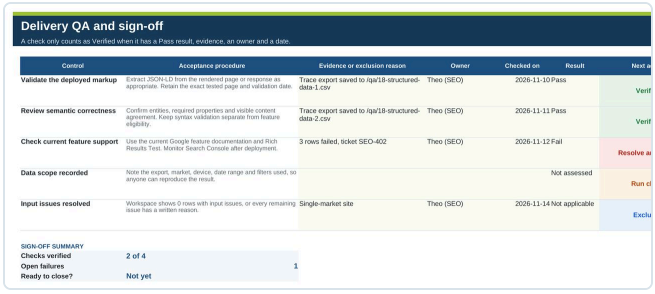
Start Here. Purpose, four-step setup and colour key.



Dashboard. KPIs, charts, Top 10 and an automatic action list.



Workspace. The working table, room for 250 rows.



Delivery QA. Evidence, owner and date for each validation step.

How the calculation works

Failed checks = 5 – (markup present + syntax valid + required fields + matches visible + policy review). Zero failures = Meets listed checks.

The key formula, as written in the workbook

```
=IF(J13="","",IF(J13<>"Ready","n.a.",IF(K13=0,"Meets listed checks","Fix or review")))
```

Result column	Plain-English formula	What it tells you
Failed checks	5-SUM(Markup present,Syntax valid,Required fields valid,Matches visible content,Eligibility reviewed)	Number of unmet implementation and review checks.
Schema release review	IF(Failed checks=0,"Meets listed checks","Fix or review")	Release review state, not a promise of rich results.

The results you'll see

Meets listed checks Fix or review

Green means the row meets the checks, red means fix it first, amber needs a human review and blue marks an opportunity. Rows with missing or out-of-range inputs show in the Input check column and are left out of the results until fixed.

How to use it, step by step

1

List URLs per template and type

One row per URL and schema type. Pick the type from the dropdown.

2

Validate syntax and required fields

Use the Rich Results Test and Schema Markup Validator. Record 1 or 0 for each.

3

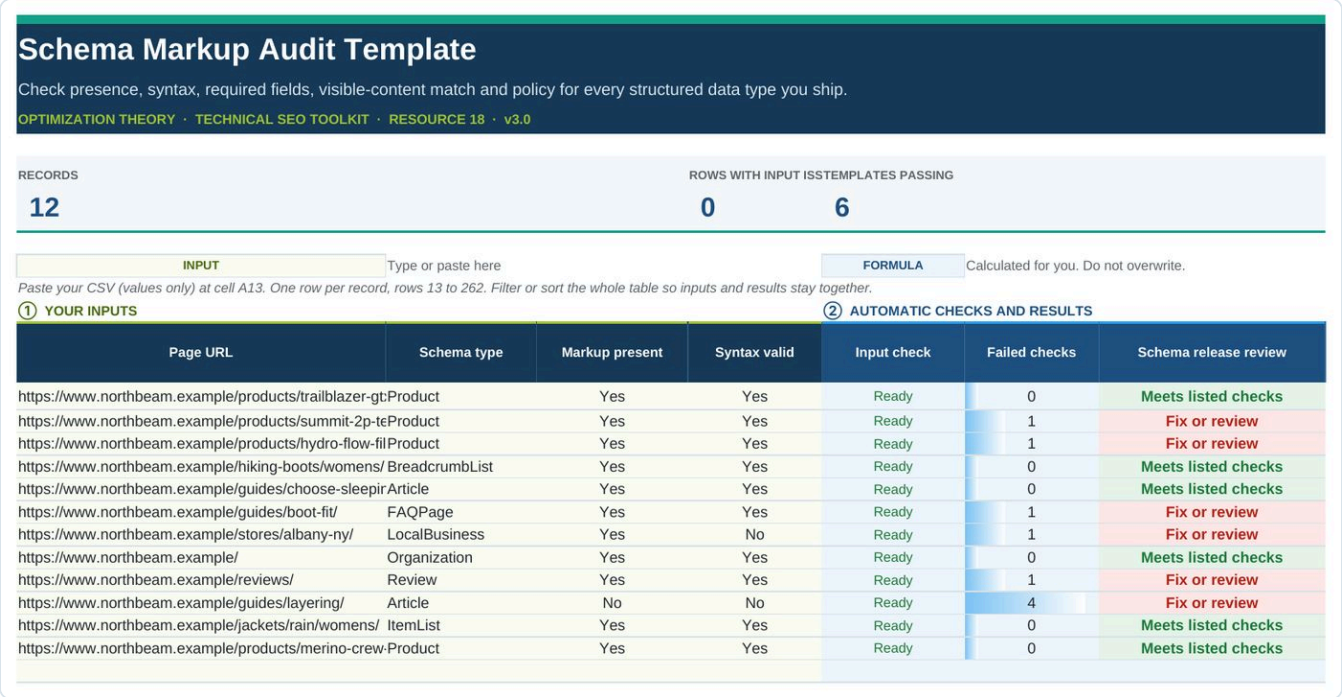
Compare with the visible page

Check that prices, ratings, dates and names in the markup match what users see.

4

Review policy

Check eligibility rules, e.g. self-serving reviews and FAQ rich result restrictions. Record the outcome.



Schema Markup Audit Template

Check presence, syntax, required fields, visible-content match and policy for every structured data type you ship.

OPTIMIZATION THEORY · TECHNICAL SEO TOOLKIT · RESOURCE 18 · v3.0

RECORDS	ROWS WITH INPUT ISS	TEMPLATES PASSING
12	0	6

INPUT — Type or paste here — FORMULA — Calculated for you. Do not overwrite.

Paste your CSV (values only) at cell A13. One row per record, rows 13 to 262. Filter or sort the whole table so inputs and results stay together.

① YOUR INPUTS — ② AUTOMATIC CHECKS AND RESULTS

Page URL	Schema type	Markup present	Syntax valid	Input check	Failed checks	Schema release review
https://www.northbeam.example/products/trailblazer-gt	Product	Yes	Yes	Ready	0	Meets listed checks
https://www.northbeam.example/products/summit-2p-te	Product	Yes	Yes	Ready	1	Fix or review
https://www.northbeam.example/products/hydro-flow-fil	Product	Yes	Yes	Ready	1	Fix or review
https://www.northbeam.example/hiking-boots/womens/	BreadcrumbList	Yes	Yes	Ready	0	Meets listed checks
https://www.northbeam.example/guides/choose-sleepir	Article	Yes	Yes	Ready	0	Meets listed checks
https://www.northbeam.example/guides/boot-fit/	FAQPage	Yes	Yes	Ready	1	Fix or review
https://www.northbeam.example/stores/albany-ny/	LocalBusiness	Yes	No	Ready	1	Fix or review
https://www.northbeam.example/	Organization	Yes	Yes	Ready	0	Meets listed checks
https://www.northbeam.example/reviews/	Review	Yes	Yes	Ready	1	Fix or review
https://www.northbeam.example/guides/layering/	Article	No	No	Ready	4	Fix or review
https://www.northbeam.example/jackets/rain/womens/	ItemList	Yes	Yes	Ready	0	Meets listed checks
https://www.northbeam.example/products/merino-crew	Product	Yes	Yes	Ready	0	Meets listed checks

The Workspace tab with the worked example. Lime cells are inputs; blue cells are formulas.

Fill it with DataForSEO

This template is built to run on **DataForSEO On-Page API**: a crawl of your site: URLs, status codes, canonicals, inlinks and sitemap membership.

This is a sign-off workflow. DataForSEO responses are the evidence you attach; the human checks stay manual by design.

1. Request the data

```
POST https://api.dataforseo.com/v3/on_page/task_post
[
  {
    "target": "yourdomain.com",
    "max_crawl_pages": 1000
  }
]
```

Post the crawl task, wait for it to finish, then request on_page/pages with {"id": "<task id>", "limit": 1000} and save that response.

2. Save the full JSON response

Save the complete response, not just the task ID, next to the template folder.

3. Convert it to the input CSV

The script in the Automation folder runs offline: no API call, no credentials.

```
python3 Automation/dataforseo_import.py import --template 18 --csv --input my-filled.csv --output my
-inputs.csv
```

Fills automatically	You add
Page URL	Schema type, Markup present, Syntax valid, Required fields valid, Matches visible content, Eligibility reviewed, Evidence

Source reference fills automatically with the task ID, endpoint, location, language, device and snapshot time, so every row is traceable. Paste the finished CSV at Workspace cell A13. Estimate the cost first with the DataForSEO Cost Calculator (template 38).

Input dictionary

Keep the CSV columns in this order and paste values at Workspace cell A13. A missing observation stays blank; use 0 only for a measured zero or a verified No.

CSV field	Column	Meaning and rule	Typical source
`url`	Page URL	Exact absolute URL. Keep case and query parameters unless a documented normalization rule applies.	DataForSEO: url
`schema_type`	Schema type	Actual schema.org type used for this page.	Your records or manual review
`markup_present`	Markup present	1 after inspecting rendered HTML or the relevant response. 1 = Yes, 0 = No.	Your records or manual review
`syntax_valid`	Syntax valid	1 after a schema syntax validator accepts the markup. 1 = Yes, 0 = No.	Your records or manual review
`required_fields`	Required fields valid	1 after validating fields for the intended feature, where applicable. 1 = Yes, 0 = No.	Your records or manual review
`matches_visible`	Matches visible content	1 only after a human verifies factual agreement with visible content. 1 = Yes, 0 = No.	Your records or manual review
`policy_review`	Eligibility reviewed	1 after reviewing feature eligibility and content policies. 1 = Yes, 0 = No.	Your records or manual review
`evidence`	Evidence	Link to a dated crawl, screenshot, ticket or source that another reviewer can inspect.	Your records or manual review
`source_ref`	Source reference	Task ID, export filename and observation scope. Retain the untouched source response separately. Optional.	DataForSEO: source_ref

The worked example

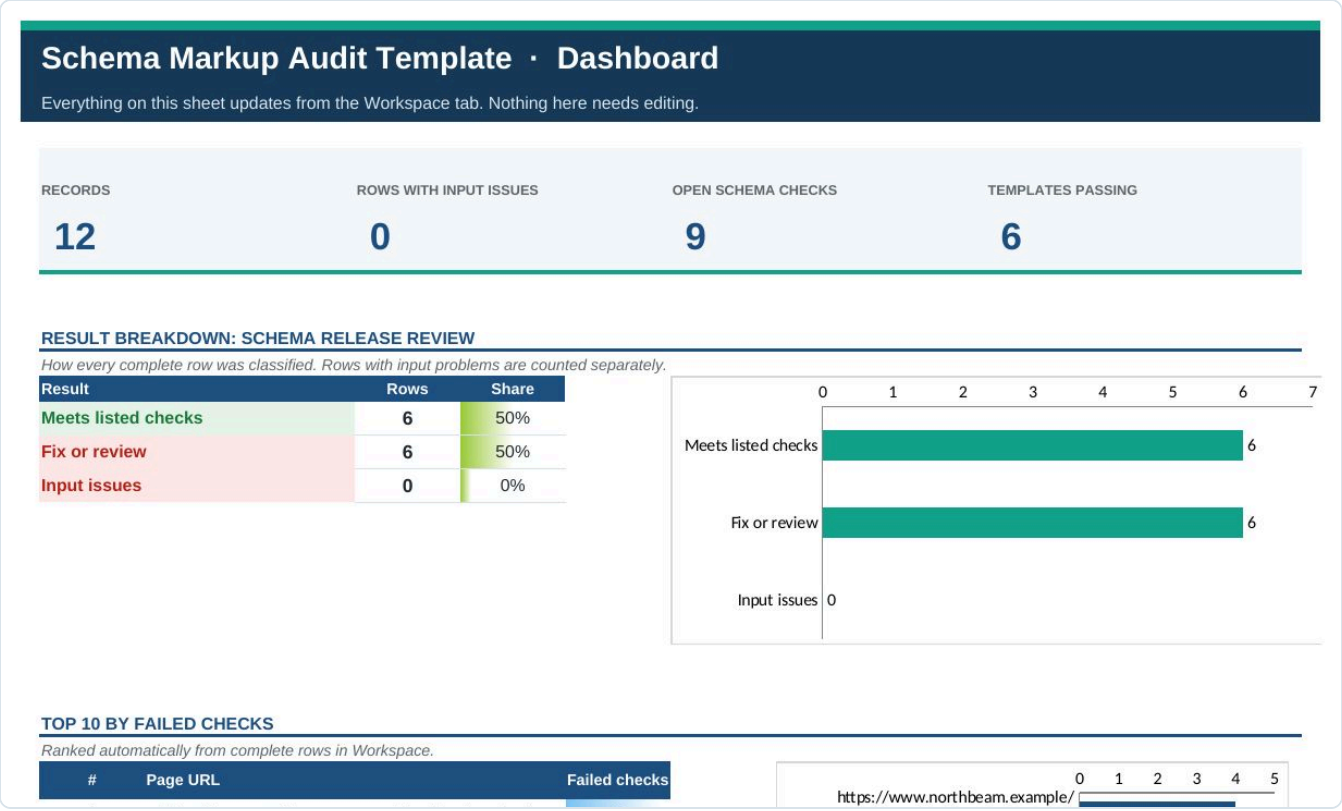
Every workbook ships filled with data from Northbeam Outdoor, a fictional outdoor-gear retailer. This one has 12 example rows.

12	0	9	6
Records	Rows with input issues	Open schema checks	Templates passing

The example catches a Product page whose markup says $39 while the page shows $44, a syntax error on a store page, and self-serving review markup that would be ignored or worse.

Top 10 by failed checks

#	Page URL	Failed checks
1	https://www.northbeam.example/guides/layering/	4
2	https://www.northbeam.example/reviews/	1
3	https://www.northbeam.example/stores/albany-ny/	1
4	https://www.northbeam.example/guides/boot-fit/	1
5	https://www.northbeam.example/products/hydro-flow-filter/	1



The Dashboard updates automatically from the Workspace tab.

Reading the result honestly

A syntax-valid entity can still be factually wrong or ineligible for a feature. Review those dimensions separately.

Limits. Generic markup presence is not type-specific validation. FAQ markup is not a universal rich-result or AI-citation tactic.

Mistakes to avoid

Marking up content users can't see. Markup must describe visible content. Hidden or invented values violate guidelines.

Chasing deprecated rich results. FAQ and HowTo rich results are heavily restricted. Markup can still help understanding, but don't expect the visual result.

Validating one URL per template. Data issues vary by page. Sample several URLs per template, especially products.

Tips from the field

Organization and LocalBusiness markup with sameAs links supports entity consistency.

Add schema checks to your release QA and the programmatic SEO checklist.

Recheck Product markup after price or stock feed changes.

Questions people ask

How do I audit schema markup?

Validate syntax, check required and recommended properties for each type, confirm the values match the visible page, and review eligibility policies.

What's the difference between the Rich Results Test and the Schema Markup Validator?

The Rich Results Test checks Google rich result eligibility; the Schema Markup Validator checks schema.org syntax generally.

Does structured data help AI search?

Accurate markup helps machines understand entities and facts on the page, which supports both search features and AI systems that read structured data.

Is structured data a ranking factor?

Not directly. It enables rich results and clearer understanding, which can improve click-through.

Templates that work well with this one

Brand Entity Consistency Audit
Compare your official brand facts with Wikidata, LinkedIn, GBP, Yelp and more, and log every correction request.

Programmatic SEO Checklist
QA every generated page for verified data, unique value, indexability, links and rendering before release.

Indexability Audit Template
Check status, robots, noindex and canonical on every page you want indexed, then see exactly what to fix first.

Answer Evidence Audit Template
Check every claim for a supporting source, original value, clarity and freshness, so AI answers can quote and cite it.

XML Sitemap Audit Template
Reconcile sitemap URLs against status, canonical, robots and noindex to see what to add, remove or fix.

Ecommerce SEO Prioritization Template
Rank product pages by potential orders, unit contribution and profit per hour, and catch out-of-stock or noindex SKUs.

Read the full tutorial online: **optimizationtheory.com/download-view/schema-markup-audit-template/**
Get all six templates in the **Technical SEO Toolkit**, or all 40 in the **Complete SEO & GEO Toolkit**.