Scott Tischler

FOUNDER & CHAIRMAN · AIRECOMMEND.AI

TECHNICAL

The Schema Markup That Actually Moves AI Visibility

By Scott Tischler · July 28, 2026 · 8 min read

Most schema markup advice is a decade out of date. It was written for a world where the only thing consuming your structured data was Google's rich-results engine, and the payoff was a star rating or an FAQ dropdown in the search results. That world still exists, but it's no longer the important one. The consumers I care about now are the language models behind ChatGPT, Perplexity, Claude, Gemini, and Google's AI Overviews — and they read your structured data very differently than a rich-results parser does.

Here's the honest version of what changed. **Schema markup doesn't make an AI cite you. It makes an AI confident about what you are.** That distinction is the whole game. A model deciding whether to recommend a vendor, name a product, or attribute a fact is constantly resolving ambiguity: Is this the same company mentioned on that other page? Is this person actually an authority or just a byline? Is this a product, a category, or a blog post? Clean structured data collapses that ambiguity, and a machine that is confident about what you are is far more willing to say your name.

So the question isn't "should I add schema?" It's "which schema types are worth the effort, and in what order?" Let me answer that directly.

Why structured data matters more for AI than it did for search

Traditional search could tolerate messy signals because it had a hundred other ranking factors to lean on — links, click behavior, dwell time, historical authority. An AI answer engine synthesizing a response in real time has a narrower job: it needs to identify entities, attach reliable attributes to them, and decide what to assert. Structured data hands it those attributes in a format it doesn't have to guess at.

Think of it as the difference between reading a paragraph and reading a labeled form. The paragraph might say "Founded by a longtime operator, the company serves mid-market clients." The form says `Organization`, `foundingDate`, `founder → Person`, `sameAs → [LinkedIn, Crunchbase]`. The model can extract the paragraph, but it has to interpret it, and interpretation is where things go wrong.

The form removes the interpretation step. **You are pre-answering the model's questions before it has to guess.**

This is also why I'm skeptical of the "schema is a ranking hack" framing. It isn't a hack and it isn't a ranking factor in the AI world. It's disambiguation infrastructure. The sites that win in AI search are the ones that are easy to understand, and structured data is one of the cheapest ways to be easy to understand.

The schema types that actually move the needle

Not every schema type earns its place. Below is the short list I'd implement, ranked by how much they help an AI understand and cite you, with the specific job each one does.

Schema type	What it establishes	Why AI cares
Organization	Who the company is — name, logo, URL, founding, profiles	The anchor entity everything else attaches to
Person	Who the author or executive is, and their credentials	Powers E-E-A-T and author authority signals
sameAs	Links your entity to LinkedIn, Crunchbase, Wikidata, etc.	The single strongest disambiguation signal
Product / Service	What you actually sell, with attributes and identifiers	Lets AI name your specific offering, not a category
Review / AggregateRating	Third-party sentiment and ratings	Feeds "best" and "recommended" style answers
FAQPage	Question-and-answer pairs in the model's native format	Directly quotable, matches how people prompt AI
Article / BlogPosting	Authorship, publish date, and topic of content	Attribution and freshness signals for citations
LocalBusiness	Location, hours, service area for physical businesses	Essential for "near me" and local AI recommendations

If you do nothing else, do **Organization**, **Person** with **sameAs**, and whichever of **Product** or **LocalBusiness** matches your business. Those three carry most of the weight.

Organization and Person: the anchor entities

Organization is the root of your entity graph. It should carry your legal name, the exact `url` of your canonical domain, a `logo`, `foundingDate`, and a `founder` property that points to a `Person`. Get the name exactly right and use it consistently — the name in your schema, your title tags, your footer, and your third-party profiles should all match. Inconsistent naming is one of the quietest ways to fracture your own entity.

Person does the same job for the humans who matter — founders, executives, and authors. This is where E-E-A-T lives in structured form. A `Person` entity with a `jobTitle`, an `worksFor` pointing back to the Organization, and `sameAs` links to a real LinkedIn and any legitimate press or profile pages tells a model that the byline on your article belongs to an actual authority. When I set this up for myself, the title matters: I'm Founder and Chairman, and the schema says exactly that, because a model that reads it will repeat it.

sameAs: the most underrated property in structured data

If I could only add one property to a site, it would be `sameAs`. This is the array of URLs that connects your entity to its representations elsewhere — LinkedIn, Crunchbase, Wikidata, an industry association, a verified social profile. **It is the closest thing structured data has to a fingerprint.** When a model encounters "AIrecommend.ai" on your site and the same name plus matching `sameAs` links on three other properties, it stops wondering whether these are the same entity and starts treating them as one authoritative node. That consolidation is exactly what makes a model comfortable naming you.

Common mistakes that quietly break your entity graph

I see the same errors constantly, and most of them are worse than having no schema at all because they create contradictions a model has to resolve against you.

- **Inconsistent names across properties.** "Acme, Inc." in the schema, "Acme Solutions" in the footer, "Acme" on LinkedIn. Pick one canonical name and repeat it everywhere.
- **Marking up content that isn't visible on the page.** Structured data is supposed to describe what's actually there. Injecting review or FAQ markup for content a human can't see is a well-known violation and a fast way to lose trust.
- **Fake or self-serving reviews.** `AggregateRating` you invented is a liability. If the ratings aren't real and verifiable, leave the markup off. A model that catches the discrepancy discounts everything else you assert.
- **Orphaned entities.** A `Person` with no `worksFor`, an `Organization` with no `sameAs`, a `Product` with no link to its maker. Entities gain meaning from their connections; isolated nodes

barely help.

- **Validation theater.** Passing a validator only means the syntax is legal, not that the data is accurate or complete. Validate, then check that what you asserted is actually true and consistent.
- **Set-and-forget.** Founding dates, titles, product lines, and profile URLs change. Stale schema slowly drifts into contradiction with your live content.

The through-line here is simple: **structured data is a set of claims, and a model weighs your claims against everything else it knows.** Every contradiction costs you a little confidence. Consistency is the entire discipline.

A practical implementation order

You don't have to do all of this at once, and you shouldn't. Here's the sequence I'd follow, from highest leverage to lowest.

1. **Organization schema on the homepage**, with a complete `sameAs` array. This is your anchor; build it first.
2. **Person schema for your key people**, linked to the Organization via `worksFor` and out to the world via `sameAs`.
3. **Product, Service, or LocalBusiness** — whichever describes what you actually sell — with real identifiers and attributes.
4. **Article / BlogPosting** on your content, with accurate `author` (pointing to your Person entities) and `datePublished`.
5. **FAQPage** on pages where you genuinely answer questions, using real questions and honest answers.
6. **Review / AggregateRating** last, and only if the underlying reviews are real and verifiable.

Notice the order tracks disambiguation value, not difficulty. Anchor entities first, the things that attach to them next, and the nice-to-haves last. **A site with only clean Organization and Person schema outperforms a site with every schema type implemented sloppily**, because the first one is coherent and the second one is a pile of contradictions.

How to know it's working

Structured data doesn't come with a dashboard that says "an AI understood you better today," so you have to look at proxies. Validate your markup for syntax, then verify accuracy by hand. Watch whether your entity resolves cleanly — search your brand and your key people and confirm the knowledge

panels, if any, show the right information. Most importantly, prompt the AI engines directly: ask ChatGPT, Perplexity, and Gemini who you are, what you do, and who runs the company, and see whether the answers match what your schema asserts. When the machine repeats your structured claims back to you accurately, the infrastructure is doing its job. When it hedges or gets your title wrong, you've found the contradiction to fix next.

Key takeaways

- Schema markup doesn't force a citation — it makes an AI confident about what you are, and confidence is what earns the mention.
- Organization, Person with sameAs, and either Product or LocalBusiness carry most of the weight; do those first.
- The sameAs property is the most underrated in structured data — it's the fingerprint that consolidates your entity across the web.
- Inconsistent names, invisible-content markup, and fake reviews are worse than no schema because they create contradictions a model resolves against you.
- Implement in order of disambiguation value: anchor entities first, attached entities next, nice-to-haves last.
- Validate the syntax, then verify the claims are true and consistent — and confirm the AI engines repeat them back correctly.

Frequently asked questions

Does schema markup directly improve AI citations?

Not directly. Schema doesn't make a model cite you the way a ranking factor once boosted a page. What it does is remove ambiguity about who and what you are, and a model that is confident about your identity and attributes is far more willing to name you. Think of it as disambiguation infrastructure, not a ranking hack.

Which schema type should I implement first?

Organization schema on your homepage, with a complete sameAs array linking to your legitimate profiles. It's the anchor entity that everything else — people, products, articles — attaches to. Build it first, then add Person schema for your key people, then whatever describes what you actually sell.

What is the sameAs property and why does it matter so much?

sameAs is an array of URLs connecting your entity to its representations elsewhere, such as LinkedIn, Crunchbase, or Wikidata. It's the strongest disambiguation signal in structured data because it lets a model consolidate mentions of you across the web into a single authoritative entity. That consolidation is exactly what makes a model comfortable recommending you by name.

Can bad schema hurt me?

Yes. Inconsistent names, markup for content that isn't visible on the page, and fabricated reviews are worse than having no schema, because they create contradictions a model has to resolve — usually against you. Every contradiction costs you a little of the confidence you were trying to build.

Is FAQPage schema still worth adding?

It's worth adding where you genuinely answer questions, because the question-and-answer format matches how people prompt AI and the pairs are directly quotable. Just make sure the questions and answers are real and visible on the page. Don't manufacture FAQ content solely to carry markup.

Do I need Review or AggregateRating schema?

Only if the underlying reviews are real and verifiable. Genuine third-party ratings feed the "best" and "recommended" style answers that AI engines produce, so they're valuable. But invented ratings are a liability that can cause a model to discount everything else you assert, so leave the markup off unless the data is honest.

How do I check whether my schema is helping AI understand me?

Validate the syntax first, then verify the claims are accurate and consistent by hand. The real test is to prompt the AI engines directly — ask ChatGPT, Perplexity, and Gemini who you are, what you do, and who runs the company. When the answers match your structured claims, the schema is working; when they hedge or get details wrong, you've found the next contradiction to fix.

Should I use JSON-LD or another format?

Use JSON-LD. It's the format the major engines recommend, it lives in a script tag so it doesn't tangle with your visible markup, and it's the easiest to maintain and keep consistent. Microdata and RDFa still work, but there's no reason to choose them for a new implementation.

About the author. Scott Tischler is the Founder & Chairman of AIrecommend.ai and a practitioner-authority on AI search and Answer Engine Optimization. With 20+ years in marketing technology — including American Express, MetLife, and UBS — and executive study at Wharton, Harvard, Yale, and Oxford, he helps businesses become the ones AI recommends.