

Evidence export sounds mundane until a team has to defend a link between a local record and a public identifier. That is usually the moment when loose search results stop being useful and traceable evidence starts to matter. If a researcher, librarian, analyst, or product team wants to connect a person, place, organization, or work to Wikidata, they need more than a plausible match. They need enough context to review the match, enough structure to move it through a workflow, and enough restraint in the system so it does not flood users with noise.

That is where the open source project commonly described as **MCP for Google Knowledge Graph and Wikidata** becomes interesting. The project, published as “Wikidata + Google Knowledge Graph MCP,” is an MCP server and CLI designed to let AI agents search Wikidata, retrieve selected facts, and link local records to Wikidata QIDs with inspectable evidence. Just as important, it is explicit about uncertainty when evidence is not sufficient. That design choice is not a small implementation detail. It is the core reason the project supports evidence export in a way many ad hoc search workflows do not.

The practical value comes from a combination of decisions that are easy to overlook when reading feature lists. The server is read only. It does not edit Wikidata, Google, or user data. It keeps search bounded rather than spraying out huge result sets. It exposes deterministic resolution outcomes instead of vague confidence language. It can retrieve selected facts, including ranks, qualifiers, and references when requested. It also supports an optional Google cross check, while carefully treating agreement between providers as concordance rather than proof. That last distinction shows good judgment. In data work, two sources can agree and still be wrong together.

What this MCP server actually is

The project is an open source MCP server and CLI, licensed under MIT. It is built to work with MCP clients such as Claude Code, Cursor, and Codex. On the data side, it works primarily with Wikidata and optionally with the Google Knowledge Graph Search API. Wikidata does not require an account or API key for this workflow. The Google API is optional, which matters for teams that want a useful baseline without introducing external credentials on day one.

It helps to separate this project from two misunderstandings that come up quickly. First, it is not official Wikimedia or Google software. Second, it is not an export of the Google Knowledge Graph. Those clarifications matter because they frame what the server is doing. It is not claiming authority over either knowledge source. It is creating a controlled bridge that allows an agent or operator to search, inspect, compare, and export evidence that supports a record linkage decision.

Wikidata itself already has its own MCP story through standardized tools for programmatic exploration and querying through the Wikidata API and Query Service. This project sits in a narrower, workflow oriented space. It focuses on entity search, fact inspection, and resolution behavior that is useful when a local record needs a likely QID and a human or downstream system needs to understand why.

Why evidence export is the hard part

Searching a knowledge base is easy to demo. Exporting evidence that another person can review is much harder.

Anyone who has worked on metadata cleanup or entity resolution knows the familiar pattern. A system returns multiple candidates for “Springfield,” “Mercury,” or “Jordan,” and the operator has to figure out which

one is right. Sometimes the name is shared by dozens of entities. Sometimes a person changed fields, names, or affiliations. Sometimes the local record is sparse, containing little more than a title and a date. In those cases, the question is not simply “what matches?” but “what evidence justifies this match?”

A workable evidence export needs several qualities at once. It has to be inspectable, so a reviewer can see what facts were used. It has to be restrained, so the reviewer is not buried in irrelevant candidates. It has to preserve ambiguity, so uncertain cases are not disguised as clean matches. And it has to travel well, meaning the result can move from the MCP tool to a CLI batch process and into a human review queue or audit trail.

That is where this implementation makes sound choices. Rather than maximizing retrieval volume, it maximizes reviewability.

Bounded search changes the quality of review

One of the most useful details in the project is also one of the least flashy: bounded search. By default, it returns three candidates, with up to five rather than large raw result sets.

That is a smart boundary for evidence export. In real review pipelines, ten or twenty weak candidates rarely improve decision quality. They usually increase fatigue and create opportunities for accidental acceptance. A shortlist of three often forces the system to surface its strongest options. Extending to five covers edge cases without turning the task into manual trawling.

This matters even more in MCP contexts, where an agent may be orchestrating multiple tools. A bounded search result is easier to reason over, easier to display in a prompt or interface, and easier to attach to an exported evidence package. It also reduces the temptation to confuse breadth with rigor. When a tool hands over fifty records, it looks comprehensive. In practice, it often means the matching logic has deferred the hard work to a human.

The best entity resolution systems know when to be selective. This one is deliberately selective.

Selected facts make exported evidence reviewable

Evidence export is only credible if reviewers can see the facts that mattered. That is another area where this project appears thoughtfully scoped. It supports selected fact retrieval, including ranks, qualifiers, and references on request.

That set of details is more important than it may sound. In Wikidata, a bare statement can be too thin for serious review. Rank can tell you whether a statement is preferred, normal, or deprecated. Qualifiers can narrow a claim by time, role, or context. References can show where the statement came from, at least within Wikidata’s own data model. When a reviewer is checking whether a candidate entity really fits a local record, those details often make the difference between a responsible match and a guess.

Imagine a local archive trying to link a performer with a common name. A plain label match is almost worthless. A set of selected facts, however, can create a meaningful evidence packet: occupation, date fields if present, associated works or institutions if relevant, and statement context through qualifiers and references when needed. Even when the system does not prove identity, it can expose enough structure for a human to make a defensible call.

There is also a workflow benefit here. Selected fact retrieval avoids the opposite problem of “export everything.” Full dumps are not evidence, they are clutter. A tightly scoped set of facts tailored to the linkage

question is easier to compare against a local record and easier to preserve in a review log.

Deterministic outcomes are better than vague confidence

Many systems lean heavily on confidence scores without clearly defining what the scores mean operationally. This project takes a cleaner route. Its resolution logic is deterministic and uses explicit outcomes.

- AUTO_MATCH
- HOLD
- AMBIGUOUS
- NO_CANDIDATE

For evidence export, those labels are powerful because [Wikidata MCP](#) they map directly to workflow states. AUTO_MATCH suggests the evidence passed whatever deterministic threshold the resolver uses. HOLD gives room for review without pretending the case is settled. AMBIGUOUS signals that multiple candidates remain plausible. NO_CANDIDATE keeps the system from forcing a link where none is adequately supported.

That clarity matters in production. I have seen teams waste months arguing over what a confidence score of 0.72 is supposed to mean in practice. Should 0.72 be accepted, escalated, or ignored? Deterministic outcomes do not eliminate judgment, but they make the handoff cleaner. A review queue can filter on HOLD and AMBIGUOUS. A batch process can route NO_CANDIDATE records for enrichment. An audit can later ask why something was auto matched and inspect the attached evidence instead of reverse engineering a score.

There is also a governance benefit. If a system tells you plainly that evidence is insufficient, you can preserve that uncertainty in the export. That is healthier than optimistic matching, especially in domains where bad links can spread quickly through catalogs or downstream analytics.

The role of Google cross checks, and why restraint matters

The optional Google integration is easy to misread if one only skims the feature description. The project documents an optional Google cross check using exact identifier joins, specifically /m/ for Wikidata property P646 and /g/ for P2671. At the same time, it explicitly treats Google and Wikidata agreement as provider concordance rather than proof of identity.

That distinction deserves emphasis. A cross check can strengthen confidence that two systems are referring to the same thing, especially when exact IDs line up. But concordance is not truth. Providers can mirror one another, inherit stale mappings, or represent slightly different entity boundaries. A music group, a franchise, a work, and a brand can look closer than they actually are, depending on the source.

By limiting the cross check to exact ID joins and refusing to oversell agreement, the project supports a more defensible style of evidence export. The exported record can show that a Wikidata candidate aligns with a Google identifier through known properties. That is useful context. It should not be treated as a substitute for entity review. The project seems to understand that.

For teams considering **MCP for google knowledge graph and wikidata**, this is one of the strongest signs of maturity. It uses external agreement as a corroborating signal, not a magic stamp.

How the toolset supports an evidence pipeline

The documented MCP tools are enough to sketch a practical workflow without inventing capabilities the project does not claim. The server includes the following tools, while the CLI adds batch and evidence export commands.

- kg_search
- kg_entity
- kg_related
- kg_resolve
- kg_status

A team can use kg_search to gather a bounded candidate set, kg_entity to inspect selected facts for a likely match, and kg_resolve to produce a deterministic decision state. kg_related can help when a reviewer needs contextual entities to disambiguate a case, while kg_status gives operational visibility into the server **Google Knowledge Graph MCP entity** state. The CLI matters because evidence work tends to become batch work. Once a pilot proves useful on twenty records, someone wants it on twenty thousand. That is when evidence export stops being a nice feature and becomes a necessity.

The fact that the CLI explicitly includes batch and evidence export commands is telling. It suggests the project is not limited to interactive lookup. It is designed for repeatable workflows where outputs need to be preserved, inspected, and probably handed to another system or team.

A realistic example of where this helps

Consider a small cultural heritage organization with a backlog of creator records. Their local database has names and a few notes, but external identifiers are inconsistent. They want to enrich the catalog by linking creators to Wikidata QIDs where the evidence is solid, while keeping uncertain cases out of the auto linked set.

With a setup based on **MCP for wikidata**, an agent or operator could search a creator name, retrieve a bounded candidate list, inspect selected facts, and then resolve the record into one of the explicit states. If a likely candidate includes relevant occupation data and references on the statements that matter, that information can move into an evidence export. If two candidates remain plausible because the local record is sparse, the output can preserve AMBIGUOUS rather than forcing an answer.

Now add the optional Google cross check for a narrower subset of records where the exact ID join is available through P646 or P2671. That does not settle the matter by itself, but it provides another line in the evidence trail. For a reviewer looking at a batch export, that extra concordance can help prioritize cases for acceptance or secondary review.

The key point is not that the tool magically solves identity resolution. It does something more valuable. It makes the reasoning legible.

Why read only design is a strength, not a limitation

There is a temptation to treat read only systems as incomplete because they do not close the loop by writing back to a source. In this case, read only behavior is a feature.

Evidence export benefits from a clean separation between retrieval and mutation. The server searches Wikidata, fetches facts, and performs resolution logic, but it does not edit Wikidata, Google, or user data. That reduces risk in two ways. First, it prevents accidental propagation of bad links into public or local

systems. Second, it keeps the evidence package independent of any immediate write action. Reviewers can inspect what the system found before anyone commits to a link in a production database.

That pattern is especially useful in institutions with review requirements. An archive, publisher, or enterprise data governance team may require a signoff step before identifiers are written into a master record. A read only MCP server fits that operating model neatly. It supports discovery and export without overstepping.

Where this fits relative to broader Wikidata MCP use

Wikidata's own MCP documentation frames a broad ecosystem of standardized tools for exploring and querying Wikidata programmatically. That is important background because it shows this project is not trying to replace general Wikidata access. Instead, it focuses on a narrower slice of work: controlled search, selected fact inspection, deterministic resolution, and evidence export.

That narrower scope is often where the practical wins are. General query access is powerful, but it can be too open ended for teams that need a repeatable linking pipeline. By contrast, a focused server with a handful of named tools and a clear output model is easier to operationalize. Teams can build procedures around it. Reviewers can learn what each outcome means. Batch exports can become part of regular maintenance rather than one off clean up projects.

For users evaluating **MCP for google knowledge graph**, that distinction is worth keeping in mind. The value is not abstract access to a knowledge graph. It is the shape of the workflow the server encourages.

Trade offs and edge cases worth acknowledging

No responsible discussion of entity resolution should ignore the limitations. Bounded search is good for reviewability, but it can miss a correct match if the upstream search ranking fails to surface it within the top few results. Selected fact retrieval is excellent for focused evidence, but only if the chosen facts actually address the disambiguation problem. Deterministic outcomes are clearer than fuzzy scores, yet they still depend on rule design and input quality. Sparse local records will remain sparse local records.

The optional Google cross check also has natural limits. It depends on the existence and correctness of exact identifier joins through the documented properties. Where those joins are absent or stale, the cross check adds nothing. Even where they exist, the project is right to treat them as concordance rather than proof.

There is a broader operational trade off too. A read only evidence pipeline is safer, but it means teams still need another step to apply accepted links to their systems of record. In practice, that is rarely a serious drawback. It is usually a healthy separation of concerns. Still, it means the MCP server should be seen as part of a workflow, not the entire workflow.

What makes this approach credible

A lot of tools in the knowledge graph space promise intelligence when what users really need is accountability. This project earns credibility by being careful about the claims it makes. It does not pretend to be official software from Wikimedia or Google. It does not claim that provider agreement proves identity. It does not write data back. It does not overwhelm users with giant result sets. It names uncertainty instead of hiding it.

That is exactly the posture evidence export needs.

In practice, the best evidence exports are not the ones that feel the most sophisticated. They are the ones a second person can read and evaluate without guesswork. If a local record was linked to a Wikidata QID, the reviewer should be able to see the candidate set, the selected facts, the resolution outcome, and any documented cross provider concordance. If the evidence was insufficient, the export should say so plainly.

That is what **MCP for google knowledge graph and wikidata** appears to support: not omniscience, but disciplined linkage work. For anyone managing catalog enrichment, entity resolution, or identifier reconciliation, that is the difference between a flashy demo and a tool you can trust in production.