

When people first connect a language model to a knowledge source, the instinct is often to ask for more. More search hits, more fields, more linked entities, more raw graph structure. In practice, that usually makes the system harder to trust. For entity resolution, research assistance, and record linking, the useful question is rarely "How much data can I pull?" It is "Which facts actually help me decide?"

That is where selected facts earn their place in MCP for Wikidata.

In the context of the open source Wikidata + Google Knowledge Graph MCP server and CLI, selected facts are not just a convenience feature. They are part of a deliberate operating model. The project is built to let AI agents search Wikidata, read selected facts, and link local records to Wikidata QIDs with inspectable evidence and explicit uncertainty when the evidence is not good enough. That last detail matters more than it may seem. Many systems fail not because they return nothing, but because they return something with too much confidence and too little explanation.

Selected facts push the interaction in the opposite direction. They narrow the scope, surface the right evidence, and make the system legible to the person reviewing the outcome.

Why selected facts matter more than full dumps

Anyone who has worked with entity data at scale learns the same lesson eventually. Full record retrieval sounds thorough, but it often introduces noise faster than it adds clarity. Wikidata entries can be rich, multi layered, and context dependent. That richness is valuable, but it is not always the right thing to send to an MCP client or an agent that is trying to answer a specific question.

If the task is to identify whether a local record refers to the same entity as a Wikidata item, a small set of relevant facts usually beats a sprawling object. The title or label matters. An occupation may matter. A date of birth may matter. A location may matter. A **Knowledge Graph MCP API** handful of identifiers may matter. Hundreds of unrelated statements do not.

This is the practical role of selected facts in MCP for Wikidata. They let the client ask for the evidence that supports a decision instead of treating the entire graph as equally important. That distinction sounds modest, yet it changes the workflow from indiscriminate retrieval to evidence based resolution.

The project's design reflects this philosophy in several ways. It supports selected fact retrieval and can include ranks, qualifiers, and references on request. That means the system is not flattening Wikidata into a simplistic key value record. It can preserve some of the editorial structure that often determines whether a statement should be trusted, ignored, or treated as one claim among several.

That structure is important because not all facts in Wikidata carry the same weight. A preferred rank can shape interpretation. A qualifier can change the meaning of a statement substantially. A reference can help a reviewer understand why the claim exists at all. Without those elements, two records that look similar at first glance can become easy to misread.

Bounded search changes the quality of review

One of the most sensible details in this MCP server is also one of the easiest to overlook. Search is bounded. By default, it returns three candidates, with a maximum of five, instead of dumping large raw result sets.

That decision has direct consequences for selected facts.

When a system returns fifty possible entities, even well chosen facts can get lost in a review process that becomes mechanical and error prone. A human reviewer starts skimming. An agent starts pattern matching too aggressively. Ambiguous names become traps. By keeping the candidate set narrow, selected facts remain interpretable. The reviewer can compare a few high value facts across a manageable number of entities and actually make a reasoned judgment.

I have seen similar dynamics in record matching projects outside this exact toolset. Once the candidate window gets too wide, teams begin inventing shortcuts to cope with volume. They rely on one eye catching property, or they assume the top hit is probably right. That is where bad links get introduced. A bounded search result, paired with selected facts, encourages a slower and more defensible kind of decision making.

The value here is not speed alone. It is discipline.

Selected facts as evidence, not decoration

The strongest feature of selected facts in this setup is that they are tied to inspectable evidence. The project's stated purpose is not simply to fetch data from Wikidata. It is to support linking local records to QIDs with evidence that can be checked, and to say explicitly when the evidence is insufficient.

That is a very different design choice from systems that optimize for seamless output and hide their internal uncertainty.

When an MCP tool can expose the particular facts used in a match decision, the outcome becomes reviewable. A local archive, catalog, newsroom, or research team can ask why a candidate was proposed. If the answer rests on a selected set of facts rather than a black box score, the decision is easier to defend and easier to reverse when necessary.

The project's resolution logic reinforces that approach through explicit outcomes such as `AUTO_MATCH`, `HOLD`, `AMBIGUOUS`, and `NO_CANDIDATE`. Those labels are more than status codes. They create a controlled vocabulary for uncertainty. Selected facts feed into that vocabulary by showing what is known, what aligns, and what remains unresolved.

An `AUTO_MATCH` without evidence is a leap of faith. An `AUTO_MATCH` with visible selected facts is a claim someone can audit.

A `HOLD` backed by conflicting or incomplete facts is not a failure. It is often the right answer.

What selected facts do for ambiguous entities

Ambiguity is where knowledge graph tooling earns its keep. The easy cases do not tell you much about a system. Common names, reused titles, organizations that changed names, and entities with sparse metadata are the real tests.

Selected facts help by constraining the comparison to what distinguishes one candidate from another.

Imagine the task is to resolve a local record for a person with a familiar name. A broad search might produce several plausible Wikidata candidates. If the MCP client retrieves a focused set of facts, such as occupation and life dates, the ambiguity can shrink quickly. If it also requests ranks, qualifiers, and references where needed, the reviewer gets a clearer picture of whether a statement is current, contested, or contextual.

That is especially valuable because identity is often not proved by one field. It is inferred from a pattern. A name plus a date can still mislead. A name plus a place can still mislead. But a selected set of mutually

reinforcing facts can push confidence in a defensible direction, or expose why confidence should remain low.

The project also documents optional Google cross checking through exact identifier joins, using /m/ for Wikidata property P646 and /g/ for P2671. Importantly, it treats agreement between Google and Wikidata as provider concordance, not proof of identity. That is the right framing. Selected facts can support a stronger resolution process, but they do not magically turn two aligned provider records into an unquestionable match. The system keeps a distinction between corroboration and proof.

That restraint is one of the healthier aspects of MCP for google knowledge graph and wikidata in this implementation. It acknowledges that multiple providers can agree and still be wrong, or agree at too coarse a level to settle a specific case.

The practical value of ranks, qualifiers, and references

A lot of graph integrations talk about facts as if they were plain statements. Anyone who has spent time with Wikidata knows that plain statements are often not enough.

Ranks matter because they indicate editorial preference or deprecation. Qualifiers matter because they add scope, timing, or relationship detail. References matter because they reveal provenance. In a selected facts workflow, these are not optional frills. They are the difference between raw retrieval and meaningful evidence.

Suppose a reviewer is looking at two candidate entities with similar labels. A selected fact saying "position held" could be helpful, but a qualifier indicating the relevant date range could be decisive. A date without context can create accidental matches. A date with context can settle them.

Likewise, a statement that exists in Wikidata but carries a less favored rank may deserve caution. Selected fact retrieval that can surface rank gives the reviewer a better sense of how strongly to lean on that statement. References, meanwhile, may not always be needed in a fast search flow, but when a result is disputed or sensitive, they provide a path for deeper inspection.

This is where MCP for wikidata becomes more than a thin wrapper around a search endpoint. It begins to act like a disciplined retrieval layer, one that preserves enough of Wikidata's internal nuance to support better downstream judgment.

How the toolset shapes the workflow

The documented MCP tools include `kg_search`, `kg_entity`, `kg_related`, `kg_resolve`, and `kg_status`. Even without inventing behavior beyond the published descriptions, you can see the intended pattern.

A user or agent searches. It inspects an entity. It may look at related entities where context matters. It resolves a local record when enough evidence exists. It checks status when needed. On the CLI side, batch and evidence export commands suggest that this is meant for real operational work, not just one off demos.

Selected facts sit near the center of that workflow because they keep each step focused. Search finds a small set of candidates. Entity inspection exposes the evidence that matters. Resolution produces a deterministic outcome. Batch processing can still remain reviewable because the evidence can be exported rather than buried.

There is a significant operational advantage in that design. When teams need to revisit a resolution decision weeks later, they do not want to reconstruct the entire query context from scratch. They want to see which

facts were examined, what aligned, and why the system landed on AUTO_MATCH or HOLD. Selected facts make that possible.

Determinism needs focused inputs

The project describes its resolution logic as deterministic. That alone will appeal to many teams who are wary of probabilistic or opaque linking behavior. But determinism only pays off when the inputs are constrained carefully.

If the retrieval layer floods the resolver with sprawling, inconsistent, or weakly relevant data, deterministic logic can still produce bad outcomes. It will simply do so consistently. Selected facts help prevent that problem by narrowing the field of judgment to evidence that is intentionally chosen for the task at hand.

This is one of the underappreciated roles of selected facts in MCP for google knowledge graph. They are not only about reducing bandwidth or making responses shorter. They improve the quality of deterministic decision making by limiting the evidence set to what can actually support a match.

That also makes edge cases easier to manage. When a record lacks enough distinguishing information, the system can remain in AMBIGUOUS or NO_CANDIDATE rather than being pressured into a false certainty by irrelevant detail. Deterministic resolution paired with selective retrieval tends to fail more gracefully.

Where selected facts help human reviewers most

The phrase "inspectable evidence" deserves more attention than it usually gets. In practice, inspectability is what separates a usable record linkage system from one that creates cleanup work for months.

Selected facts help reviewers in at least four concrete ways:

1. They reduce cognitive load by limiting attention to the facts that matter for the current decision.
2. They preserve context through ranks, qualifiers, and references when a plain statement would be misleading.
3. They support explicit uncertainty, making it easier to hold or reject a match rather than forcing a premature decision.
4. They make audit trails practical, especially when evidence export is part of the workflow.

That combination matters in environments where data quality is uneven. Local records are often messy. Names are abbreviated. Dates are partial. Subjects are inconsistently categorized. A system that extracts carefully chosen facts from Wikidata gives the reviewer something tractable to work with.

It also changes team behavior. When evidence is concise and visible, people are more likely to challenge a weak match. When evidence is buried in an oversized payload, they are more likely to wave it through.

What this means for MCP clients

The server is intended for use in MCP clients such as Claude Code, Cursor, and Codex. That context matters because MCP interactions often happen inside active reasoning workflows. The client is not just displaying data, it is helping an agent decide what to do next.

In that setting, selected facts are especially valuable. They let the client retrieve enough information to support a step in reasoning without overwhelming the context window or muddying the prompt with irrelevant graph content. For agentic use, less can genuinely be more, provided the "less" is chosen well.

This is also why the project's read only posture deserves mention. It does not edit Wikidata, Google, or user data. It is not official Wikimedia or Google software. It is not an export of the Google Knowledge Graph. Those boundaries reduce the risk of conflating retrieval with authority. The system is there to support inspection and resolution, not to claim ownership over the underlying sources.

For teams experimenting with MCP for wikidata, that distinction is healthy. It encourages a workflow where the MCP server is a careful mediator between knowledge sources and decisions, not a magical oracle.

Trade offs and limits

Selected facts are powerful, but they are not a cure for poor inputs or underspecified matching criteria.

If a local record lacks distinguishing metadata, no elegant fact selection strategy can reliably identify the right QID every time. If two candidates are genuinely difficult to separate, selected facts may correctly preserve ambiguity rather than resolve it. That can frustrate users who want a crisp answer, but it is better than a false match.

There is also a judgment call in deciding which facts should be selected. Too narrow, and the system may miss a decisive clue. Too broad, and it slips back toward information overload. The project's support for qualifiers, references, and ranks softens that trade off because it allows depth when needed, not by default in every case.

Another limit is that provider agreement has to be interpreted carefully. The optional Google cross check can be useful, particularly through exact id joins, but the project is explicit that concordance is not proof of identity. That warning should not be treated as boilerplate. It is central to responsible use.

A useful way to think about selected facts is that they sharpen evidence, they do not replace judgment.

A sensible operating pattern

For organizations considering this kind of setup, the strongest pattern is usually a staged one. Start with bounded search. Pull selected facts for the top candidates. Resolve only when the evidence is clear. Hold the rest for review. Export evidence for anything that may need to be revisited.

That pattern matches the tool's documented strengths and avoids asking it to do something it does not claim to do.

A practical review **Wikidata MCP** flow might look like this:

1. Use `kg_search` to retrieve a small candidate set.
2. Inspect selected facts with entity level retrieval, requesting ranks, qualifiers, or references when the case is not straightforward.
3. Run deterministic resolution and respect `HOLD`, `AMBIGUOUS`, or `NO_CANDIDATE` when the evidence does not support an automatic match.
4. Use evidence export in batch workflows so decisions remain auditable.

There is nothing flashy about this approach, and that is part of its value. The best data linkage workflows are often the ones that resist drama. They create enough structure for automation to help, while leaving room for uncertainty to remain visible.

Why this design deserves attention

Wikidata's broader MCP landscape already includes standardized tools for exploring and querying Wikidata programmatically through the Wikidata API and the Wikidata Query Service. That wider context matters because it shows there is real demand for MCP access to knowledge sources. The Wikidata + Google Knowledge Graph MCP server stands out within that context by taking a narrower, more operationally grounded path.

Its emphasis is not just on exploration. It is on search, selected fact retrieval, deterministic resolution, and inspectable evidence. That makes it especially relevant for teams who care less about open ended graph browsing and more about linking local records to stable identifiers with a defensible process.

Selected facts are the hinge that makes that process workable. They bridge the gap between raw graph richness and practical decision support. They keep search bounded, evidence visible, and uncertainty explicit. They allow MCP for google knowledge graph and wikidata to function as a disciplined record linkage aid rather than a firehose of loosely connected data.

For anyone building serious workflows around Wikidata, that is not a minor implementation detail. It is the difference between a system that merely fetches information and one that helps people reach sound decisions.