

Anyone who has spent time linking records to Wikidata learns the same lesson sooner or later: uncertainty is not a bug. It is the work. The hard part is rarely finding a candidate. The hard part is deciding whether the candidate is specific enough, evidenced enough, and distinct enough to deserve a QID on the record in front of you.

That is why the HOLD outcome matters in MCP for Wikidata, especially in the open source project commonly described as the Wikidata + Google Knowledge Graph MCP. This server and CLI were built for a very practical task: let an agent search Wikidata, read selected facts, and link local records to Wikidata QIDs with inspectable evidence and explicit uncertainty when the evidence is not strong enough. That last clause is where HOLD earns its keep.

A lot of systems are designed to feel decisive. They would rather guess than pause. In entity resolution, that instinct causes damage. A wrong link can look harmless at first, just one QID attached to one row. Later it contaminates downstream enrichment, analytics, authority control, and user trust. A clean HOLD, by contrast, tells you that the system saw possibilities but refused to overstate what it knew.

What HOLD means in this MCP context

The resolution logic in this project is documented as deterministic, with explicit outcomes such as AUTO_MATCH, HOLD, AMBIGUOUS, and NO_CANDIDATE. Those labels matter because they tell you the system is not hiding its state behind vague confidence language. It is making a categorized decision.

AUTO_MATCH is the easy case. The system has enough basis to resolve a local record to a specific Wikidata entity. NO_CANDIDATE is also straightforward. It did not find a plausible target in the bounded search it performs. AMBIGUOUS signals a live contest between candidates that the current evidence does not break cleanly.

HOLD sits in a different space. In practice, it means the system has not committed to an automatic match because the evidence is insufficient for that step. That does not necessarily mean there were no candidates. It does not necessarily mean there were multiple candidates. It means the move from candidate discovery to final resolution would be too aggressive.

That distinction matters more than it seems. Teams often treat all non matches as one bucket, then wonder why review queues feel chaotic. A HOLD record is different from a record with no candidate and different again from a record with several competing candidates. If you handle them the same way, you lose the very signal the tool was designed to preserve.

Why a careful system needs a hold state

This project emphasizes bounded search. By default it returns three candidates, up to five, rather than dumping a large raw result set. That design choice makes the workflow more disciplined. It keeps the search space human sized and agent sized. It also means the system is not pretending to exhaust every conceivable option before making a decision.

Bounded search is useful because entity resolution is usually not improved by staring at the fifteenth weak result. In most real cases, the best candidate appears near the top if the input data is decent. But bounded search also reinforces the need for HOLD. Once you decide not to flood the user with noise, you need a principled way to say, "I found enough to investigate, but not enough to finalize."

That is a mature posture. It acknowledges that search and resolution are different acts. Search asks, “What might this be?” Resolution asks, “What am I willing to attach to the record as an identity claim?” A professional workflow never confuses the two.

I have seen many data teams underestimate this separation. They assume that a high quality search interface naturally yields high quality linking. It does not. Good search narrows the field. Good resolution protects the record. HOLD is one of the mechanisms that keeps those roles distinct.

What usually pushes a case into HOLD

The verified project materials do not provide a full public scoring rubric for every HOLD, so it would be wrong to invent one. Still, the project’s stated behavior tells us enough to describe the pattern responsibly.

First, HOLD appears when evidence exists but is not strong enough to support an automatic identity claim. That fits the project’s emphasis on inspectable evidence and explicit uncertainty.

Second, the project can retrieve selected facts, including ranks, qualifiers, and references on request. That means the system is not limited to labels alone. It can inspect statement quality and context. A case can therefore land in HOLD not because there is no data, but because the available data does not line up clearly enough.

Third, the project supports an optional Google cross check through exact identifier joins, specifically /m/ for Wikidata property P646 and /g/ for P2671. Crucially, the documentation says Google and Wikidata agreement is treated as provider concordance, not proof of identity. That single design choice explains a lot about HOLD. Even when two providers appear to agree, the system does not treat concordance as a shortcut around judgment. If the match still lacks sufficient basis, HOLD remains appropriate.

That last point deserves emphasis because it cuts against a common temptation. People often assume that if two large knowledge systems line up, the problem is solved. It is not. It may be a strong signal, but a signal is not the same thing as proof. The project’s documentation is careful here, and that caution is exactly the kind of engineering discipline you want in identity work.

HOLD is not indecision, it is a bounded refusal

There is a psychological difference between a system that “cannot decide” and a system that “refuses to overclaim.” HOLD belongs to the second category.

The server is read only. It does not edit Wikidata, Google, or user data. That makes the hold state even more meaningful. Since the tool is not writing back, its power lies in surfacing candidates, evidence, and explicit outcomes that other systems or reviewers can trust. If it blurred weak cases into matches, it would undercut the main reason to use it.

This is one place where MCP for google knowledge graph and wikidata has a practical advantage over looser search setups. The decision vocabulary is explicit. The bounded candidate set is explicit. The optional Google check is explicit. The uncertainty is explicit. You can build a review process around explicit states. You cannot build one around vibes.

A HOLD decision effectively says, “Stop here. Review before linking.” For people who manage batch workflows, that is not a nuisance. It is triage. It helps separate records that can safely flow through automation from records that need a closer look.

The evidence you should inspect next

When a record lands in HOLD, the right move is not to hunt for more guesses. The right move is to inspect the facts that bear directly on identity. This MCP supports selected fact retrieval, and it can include ranks, qualifiers, and references on request. That is where the useful work begins.

A plain label match is rarely enough, especially for people, organizations, places with naming collisions, or works that share titles across editions and adaptations. What resolves identity are the contextual facts around the candidate entity. A rank can tell you which statement is preferred versus merely present. A qualifier can narrow the meaning of a statement. A reference can tell you whether a claim has support. None of those elements guarantee truth on their own, but together they let a reviewer judge whether the candidate actually fits the local record.

A common mistake in review queues is to compare only surface names. That is how bad links survive. The better habit is to ask whether the specific facts that define the local record are visible on the candidate in a way that makes the match defensible.

If they are visible and coherent, the record may move from HOLD to manual acceptance. If they are partly visible but incomplete, the prudent result may remain a hold pending more local context. If they conflict, the candidate should be rejected even if the label looks attractive.

How HOLD differs from AMBIGUOUS

Teams often confuse these two because both block automation, but they are not the same operationally.

AMBIGUOUS points to a contest between plausible candidates. The problem is comparative. You have more than one viable destination and cannot tell which is correct with the current evidence.

HOLD points to insufficient grounds for finalizing a link. The problem is evidentiary. There may be one candidate that looks best, but the basis for automatic acceptance is still too thin.

That difference affects review strategy. In an ambiguous case, the reviewer's first job is elimination. Which candidate can be ruled out by inspecting distinguishing facts? In a hold case, the reviewer's first job is validation. Does the best candidate have enough support to warrant the link at all?

I have found that separating those mental modes speeds review. When people treat HOLD as if it were just another flavor of ambiguity, they waste time comparing candidates when the real issue is that none of the currently inspected evidence justifies a final match. When they treat ambiguity as if it were simply missing evidence, they go searching for more background on a candidate that is already losing on distinguishing details.

Where the Google cross check helps, and where it does not

The optional Google layer is easy to misunderstand. This project is not an export of the Google Knowledge Graph. It is not official Google software, and it is not official Wikimedia software either. The optional cross check is narrow and exact. It uses specific identifier joins, `/m/` aligned with Wikidata property P646 and `/g/` aligned with P2671.

That exactness is a strength. It avoids hand waving. If the identifiers line up, you have a documented piece of provider concordance. If they do not, the system does not pretend otherwise.

Still, the documentation is explicit that provider concordance is not proof of identity. This is the right stance, and it has direct implications for HOLD. A reviewer should treat Google agreement as reinforcing context, not as a permission slip to bypass evidence review. In some cases it may help move a record out of HOLD. In others it may simply confirm that two systems are aligned on a still insufficiently evidenced candidate.

That may sound conservative, but conservatism is often exactly what preserves data quality. Many of the worst authority mistakes happen because a secondary signal is mistaken for a decisive one.

Why bounded search changes the way you interpret a hold

The server returns a small number of candidates by design, three by default and up to five. That boundedness does two things.

It reduces noise, which is helpful for both agents and human reviewers.

It also means a HOLD is being issued in a deliberately compact search context, not after a sprawling scrape of everything remotely similar.

That tells you something important. A hold here is not the product of indecisive overexposure. It is a controlled result from a system that prefers a small, inspectable candidate set. This can make review more efficient because you are not sorting through a haystack. You are looking at the few candidates the system considered worth surfacing, then deciding whether the best among them clears the identity threshold.

For workflows using MCP for wikidata, this bounded approach can be the difference between a review queue that is merely busy and one that becomes unmanageable. When every unresolved case comes with fifty suggestions, review degenerates into search. When it comes with a handful of candidates and a HOLD flag, review stays focused on evidence.

A practical review pattern for HOLD cases

The mechanics of the tooling suggest a simple but disciplined pattern. Use the search and resolution tools to identify the bounded candidate set, then inspect the candidate entity and any relevant selected facts before deciding whether to promote, reject, or defer the link.

The documented MCP tools support that flow. `kg_search` helps surface candidates. `kg_resolve` provides the deterministic outcome state. `kg_entity` and, where useful, `kg_related` help inspect what the candidate actually says. `kg_status` helps verify the service state if you need to rule out environmental issues. The CLI goes further with batch and evidence export commands, which is useful if your review work spans many records and you need to preserve the evidence used at decision time.

Here is the only checklist I recommend for working a HOLD queue:

1. Confirm what exact local record details are non negotiable for identity.
2. Inspect the top surfaced candidate's selected facts, not just its label.
3. Check ranks, qualifiers, and references when those details carry the identity signal.
4. Treat any Google concordance as supporting context, not as proof.
5. Approve only when the match is defensible on evidence you can explain to another reviewer.

This sounds obvious until you are processing a few hundred records under deadline pressure. Then people drift toward shortcuts. The list exists to keep the decision explainable.

What a good HOLD queue feels like in practice

A healthy HOLD queue is not a pile of failures. It is a filter of records where caution is doing useful work.

In my experience, the strongest indicator that a resolution system is worth trusting is not how many records it auto matches. It is how well it refuses the dangerous ones. If a tool auto matches everything, the cleanup cost arrives later, usually hidden inside enrichment errors and user complaints. If a tool uses HOLD appropriately, the cleanup cost is moved forward into a visible review step where it belongs.

There is also an operational benefit. Because this project is built around inspectable evidence, a HOLD can be reviewed consistently by different people. One reviewer may be faster than another, but both can work from the same surfaced facts and the same explicit outcome label. That consistency matters if you are building internal policy around authority control or entity linking.

This is one reason the phrase MCP for google knowledge graph is most useful when it is paired with a clear understanding of scope. The Google element here is optional and exact join based. The center of gravity remains the disciplined handling of Wikidata search, fact inspection, and resolution states. HOLD belongs to that discipline.

Edge cases worth respecting

A few edge conditions tend to produce friction even in well designed systems.

The first is the “almost right” candidate. This is the entity whose label and broad type look convincing, but the selected facts do not yet support identity. These are dangerous because they invite reviewer fatigue. After staring at **Learn more** enough near misses, people start accepting them. A strong HOLD policy resists that drift.

The second is the “concordant but still thin” case. Here the optional Google cross check may align through the exact identifier mapping, yet the reviewer still lacks enough contextual evidence for confidence. These are precisely the cases where the project’s documentation helps by drawing a bright line between concordance and proof.

The third is the “insufficient local record” problem. Sometimes the candidate is not the weak link. The local record is. If the input row lacks the attributes needed to distinguish identity, the correct outcome may remain HOLD even when the external entity looks well described. Good systems cannot manufacture specificity that the source record does not contain.

Those edge cases are why it is a mistake to view HOLD as merely temporary friction. Often it is the most honest answer available given the current record and the current evidence.

Why this design is valuable for serious data work

The broader Wikidata MCP ecosystem is about giving language models standardized tools to explore and query Wikidata programmatically through the Wikidata API and Wikidata Query Service. Within that broader context, this particular project is doing something very practical. It narrows the problem to search, selected fact inspection, and explicit, deterministic resolution outcomes.

That combination is unusually useful in production environments because it supports accountability. When a record is matched, you know it was not the result of an unbounded search blur. When a record is held, you

know the uncertainty was surfaced rather than hidden. When the optional Google layer is used, you know the concordance claim is limited and exact, not inflated into a false guarantee.

If you manage entity linking at any meaningful scale, those are not abstract virtues. They are the difference between a workflow that can be audited and one that cannot.

HOLD is often treated as the least satisfying result because it does not deliver the finality people want. In practice, it is one of the most important results the system can produce. It protects the integrity of the record, preserves reviewer attention for the cases that need it, and keeps the line between evidence and assumption where it belongs.

That is the right way to understand HOLD decisions in MCP for Wikidata. They are not signs that the system failed to do its job. They are signs that it did the harder job instead, which is knowing when not to pretend certainty.