Cache raw evidence; rebuild interpretations

Key takeaways

1. Cache raw evidence when acquisition is costly and interpretation will change.
2. A warm cache turns an estimated 3.2-hour scrape into an 88-second rebuild.
3. Validate raw preservation at the cache and meaning in derived state.

Common misunderstandings of caching and validation

Which claim would you defend right now? Choose before the evidence.

"A cache should store parsed records so rebuilds skip parser work."

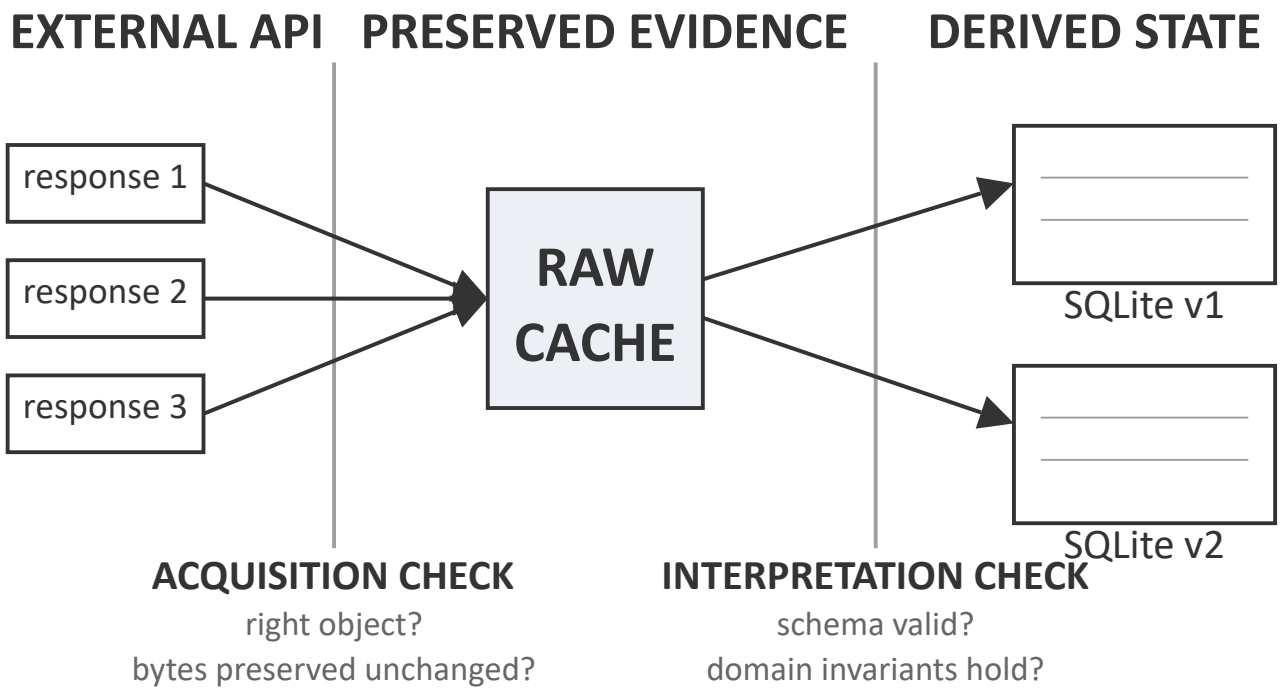
"If cached JSON parses and matches its schema, the derived data is sound."

"Changing the parser means the cache key or cached response must change."

Keep your choice. Look for the mechanism or counterexample that would make it fail.

Two kinds of state, two different rules

- Remote responses are expensive observations. Preserve them unchanged.
- SQLite rows are one interpretation of those observations. Recreate them freely.
- Validation belongs at both boundaries, but it asks different questions at each one.



source: music-time/HISTORY.md; local cache timing audit, 2026-07-09

music-time pays for acquisition once

Every parser and schema experiment starts from the same preserved responses

Fetch Archive.org metadata.

Slow, remote, rate-limited

Write the unmodified JSON response to disk.

Durable evidence

Parse JSON into normalized SQLite rows.

Cheap, replaceable state

A parser change invalidates the interpretation, not the observation.

source: music-time/gdtimings/archive_org.py and gdtimings/cache.py

The raw cache is a one-time network bill

The current Archive.org cache is large enough that the saved work matters

18,084	919 MB	~30 min	64
JSON files	**on disk**	**first fill**	**workers**
one per recording	raw metadata	project record	parallel fetch

- The cache uses a two-level directory so one directory does not hold 18,000 files.
- Each write uses a temporary file and `os.replace()` so a crash cannot publish half a JSON object.
- A warm cache removes Archive.org from the edit-test loop.

source: music-time/HISTORY.md @ 2026-02-21; ~/.gdtimings/cache measured 2026-07-09

A warm cache turns hours into seconds

The no-cache path fetches sequentially; the warm path only reinterprets local JSON

Linear scale. ~3.2 h extrapolates the 30-object network sample; it is not a full cold run.

18,045	363,147	~131x
releases	**tracks**	**shorter loop**
fresh SQLite	reparsed	3.2 h / 88.092 s

source: music-time audit, 2026-07-09: 30-object network sample extrapolation + measured warm rebuild

One cached object is about 97 times faster

A 30-object sample isolates network acquisition from local parsing

Same 30 Archive.org objects; local read includes JSON decoding.

19.379 s

network total

30 objects

0.191 s

local total

30 objects

97x

median ratio

600.6 / 6.202

source: music-time 30-object cache timing audit, 2026-07-09

The cache stores observations, not conclusions

Fetch and interpretation are separate functions with a file boundary between them

Phase 1: acquire + preserve

```
url = ARCHIVE_METADATA_URL.format(
    identifier=identifier
)
data = _api_get(_thread_session(), url)
if data:
    _write_cache(cache_dir, identifier, data)

# JSON is the source response, unchanged
```

Phase 2: reinterpret

```
data = _read_cache(cache_dir, identifier)
if data is None:
    continue

release_id, track_count = (
    _process_from_cache(conn, identifier, data)
)

# SQLite is disposable derived state
```

source: music-time/gdtimings/archive_org.py

Wrong dates poisoned SQLite, not the cache

Symptom	58 Playing in the Band performances appeared in 1997–2026.	**The Grateful Dead stopped touring in 1995.**
Cause	When a disc title lacked a concert date, the parser fell back to the release publication date.	**A valid date answered the wrong question.**
Scope	127 of 151 MusicBrainz releases received the wrong year.	**The cached response was still trustworthy.**

A cache is valuable when a bug can be repaired without reacquiring the evidence.

source: music-time/HISTORY.md @ 2026-02-22; commit 08eb1ca

The fix changed interpretation, not evidence

The reasonable-looking fallback silently changed the meaning of the field

Wrong: substitute another kind of date

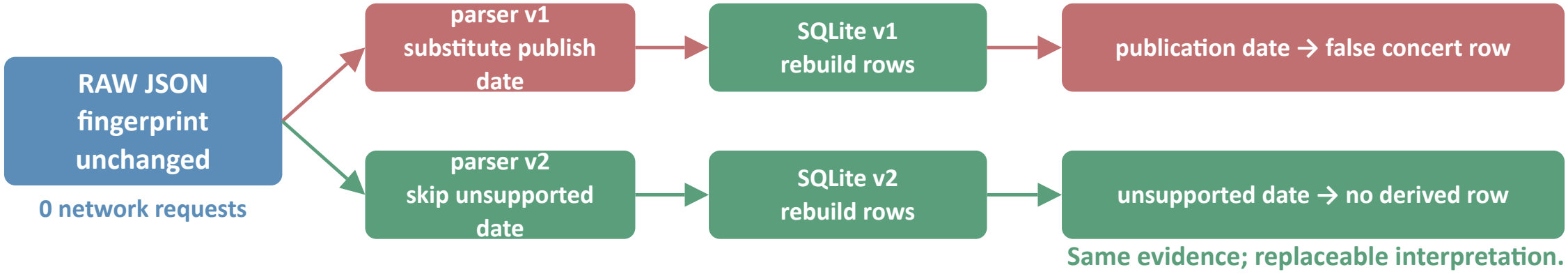
```
disc_date = parse_date_from_title(
    medium.get("title", "")
)
if disc_date is None:
    disc_date = release.get("date", "")

# publication_date != concert_date
```

Right: skip unsupported interpretation

```
disc_date = parse_date_from_title(
    medium.get("title", "")
)
if disc_date is None:
    continue

# Preserve raw response; rebuild rows
```



source: music-time commit 08eb1ca

Validate the two layers separately

The word validation hides two separate correctness questions

Layer	Question	music-time checks
Raw cache	Did we preserve a complete, parseable response?	JSON decode; required envelope; atomic rename; optional max age
Derived DB	Does this interpretation match the domain?	no Dead shows after 1995; dates mean concerts; one best tape per show
Published plot	Does the result answer the stated question?	random challenge one performance; inspect outliers; compare known facts

Types prove date shape; domain invariants prove date meaning.

A useful cache policy answers four questions

If one answer is vague, the cache will eventually lie or become irreplaceable

Decision	Good answer for raw evidence	Common mistake
Identity	remote object ID + source	keying on a derived display name
Freshness	explicit max age or immutable source	assuming local means current
Publication	temp file + atomic replace	writing final path in place
Rebuild	delete SQLite; replay every raw object	treating cache and database as peers

Do not key raw responses by parser version. Parser changes are why the cache exists.

You vs. the Agent: cache boundaries

Your Responsibility
Choose what is authoritative evidence.
Define freshness and domain invariants.
Decide which derived state is safe to delete.

Agent's Responsibility
Implement atomic cache reads and writes.
Measure warm and cold paths reproducibly.
Rebuild derived state and run invariant checks.

Shared: Inspect one random cached object and its derived rows end to end.

The architecture buys cheap correction

1. Cache raw evidence when acquisition is expensive and interpretation will change.
2. A warm music-time cache turns an estimated 3.2-hour sequential scrape into an 88-second rebuild.
3. Validate preservation at the cache boundary and meaning at the derived-state boundary.