Reliable Context

RELIABLE CONTEXT / RESEARCH & PRACTICAL GUIDANCE

Valid Time vs Transaction Time: Which Clock Proves What?

> Reliable Context is a kernel of truth: a system of record for organizations that fuse private and public information and must defend every value they act on. One canonical record per real thing, a cited source and an observation date behind every value, unknowns declared rather than left blank, and any past state of the record reproducible.

Published by Reliable Context · 2026-09-22 · valid time vs transaction time · bitemporal data · temporal database · observation time · publication time · event time · system versioning · data provenance · as-of queries

Original article: https://reliablecontext.com/articles/valid-transaction-observation-publication-time



In this article

1. Executive Summary
2. Introduction and Background
3. Four Clocks and the Claims They Support
4. A Delayed Correction Across Four Clocks
5. What System-Versioned History Can and Cannot Answer
6. Four As-Of Tests for Reproducible Answers
7. Data Analysis and Evidence
8. Implementation Guidance: Do You Need Bitemporality?
9. Implications and Future Directions
10. Frequently Asked Questions (FAQs)
11. Conclusion

Executive Summary

Valid time and **transaction time** answer different questions. Valid time says when a proposition held in the modeled world. Transaction time says when a database state contained the proposition. The temporal-database literature calls the dimensions independent, and SQL terminology commonly maps them to application time and system time (wiki.postgresql.org). A system that keeps both can answer two-dimensional questions such as, “What did the record on 20 June say was effective on 1 April?” It still cannot necessarily prove when an external source was first seen or when that source published its claim.

For defensible fused records, **four clocks** should be distinguished: world-valid time, system-record time, source-observation time, and source-publication or revision time. OpenTelemetry defines event time using the origin clock and observation time using the collector’s clock (opentelemetry.io). DataCite separately defines an issued date as the date a resource is published or distributed (datacite-metadata-schema.readthedocs.io). These are not redundant labels. They identify different authorities and prove different propositions.

A delayed correction makes the distinction concrete. A fact may become valid on **1 April**, be published on **10 May**, be observed on **12 May**, and enter a system on **13 May**. A correction entered on **20 June** can change what the system then believed about 1 April without changing when the underlying event occurred. Half-open intervals, written **[start, end)**, make adjacent versions non-overlapping because the start is included and the end excluded (wiki.postgresql.org). The resulting model preserves both occurrence and belief history.

Ordinary system versioning is sufficient when the only question is what rows the database held at a prior system time. Microsoft SQL Server, for example, manages row periods and separates current from history tables (learn.microsoft.com). Bitemporality is necessary when queries constrain both effective time and recorded time. Evidence-grade replay requires more: stable source-version identity, the observation event, and the source’s own publication metadata. W3C PROV treats time as a critical provenance dimension and models each revision result as a new entity (www.w3.org).

Introduction and Background

The phrase “as of” is dangerously underspecified. A reviewer who asks for an organization’s address “as of 1 April” may mean the address effective in the world that day, the address recorded in a database that day, the address visible in a source version available that day, or the address the organization had actually collected by that day. One timestamp cannot answer all four questions.

Temporal database theory provides the first two axes. The distinction between **valid time and transaction time** is semantic, not merely technical. Jensen and Snodgrass define valid time as when a fact is true in the modeled reality (doi.org). In the conceptual model, transaction times are generated by the database system itself (homes.cs.aau.dk). Clifford and Isakowitz add reference time, the time at which an operation is applied to a database state (archive.nyu.edu).

Modern information fusion adds an evidentiary problem. A database may ingest a web page days after publication and correct its interpretation weeks after that. The source’s publication clock, the collector’s observation clock, and the database’s acceptance clock then diverge. W3C DCAT warns that a catalog listing date is not the publication date of the dataset itself (www.w3.org). The distinction is the essence of temporal data provenance.

This report defines each clock, assigns its authority, and gives four reproducible as-of tests. It stays at the level of observable record properties and evaluation. The design question is not whether every timestamp deserves a database column. It is whether the evidence package can prove the proposition that a disputed value requires.

Four Clocks and the Claims They Support

Valid time: when the modeled proposition held

Valid time, also called **application time**, **business time**, or sometimes effective time, belongs to the modeled domain. It may be an instant, an interval, or a set of either (doi.org). The authority is normally a domain source or a justified interpretation, not the database clock.

Valid time answers: “When was this proposition true in the represented world?” It does not, by itself, say when anyone learned the proposition. Oracle documentation makes the ownership distinction explicit: valid time is usually user-controlled while transaction time is system-managed (docs.oracle.com).

Transaction time: when the system recorded a proposition

Transaction time, also called **system time** or recorded time, belongs to the database or system of record. It marks the period during which a row version was current in that system. Transaction time cannot extend into the future because the system cannot already have recorded a future transaction (www2.cs.arizona.edu).

This clock answers: “What did this system contain or assert at recorded time R?” It is not necessarily the time of an external observation. A collector may observe evidence at noon, a reviewer may accept it at 15:00, and a database transaction may commit at 15:03. Collapsing those events into one transaction timestamp destroys the ability to measure acquisition delay or review delay.

Observation time: when this organization encountered the source

Observation time is the time a collecting process encountered an event, document, or source version. OpenTelemetry defines it as the time a collection system observes an event (opentelemetry.io). For externally originating events, that time is measured by the collector's clock, not the origin clock (opentelemetry.io).

Observation time answers: "What evidence had this organization actually seen by O?" It may be missing when observation cannot be established. Specifications permit an event timestamp to be absent when the source timestamp is unknown (opentelemetry.io). Unknown must remain an explicit state, not be replaced with transaction time.

Source publication time: when the source made that version available

Source publication time or **source revision time** belongs to the publisher and the particular source version. DataCite defines "Issued" as publication or distribution date and "Updated" as the date of the resource's last update (datacite-metadata-schema.readthedocs.io). W3C DCAT assigns responsibility for making a resource available to its publisher (www.w3.org).

Publication time answers: "Was this source version publicly or contractually available by P?" It may be self-declared, inferred from a version history, or unknown. A retrieval timestamp proves retrieval, not publication. A page's modification date proves only what its publishing system says about modification, not when an organization first saw it.

Table 1 assigns each clock to its owner and audit question.

Clock	Definition and authority	Mutability and unknown state	Audit question proved
Valid or effective	When the proposition held in the modeled world. The domain source or justified domain interpretation supplies it. (doi.org)	A later correction may revise the asserted interval. Unknown start, unknown end, open end, and not applicable are different states.	What value applied at valid time V?
Transaction or system	When a version was current in the recording system. The database controls it. (sigmod.org)	Historical versions should be append-preserving within the retention policy. It begins no earlier than system acceptance.	What did system S record at time R?
Observation	When the collecting organization encountered the source or event, measured by its collection clock. (opentelemetry.io)	A re-observation creates another event. Unknown means no defensible observation timestamp, not "same as ingestion."	Which evidence had organization O observed by time O?
Publication or revision	When the publisher made that exact source version available. The publisher or version metadata is authoritative. (datacite-metadata-schema.readthedocs.io)	A revision creates a distinct source version; missing may mean unknown, as DCAT allows for modification date. (www.w3.org)	Which source version existed by publication cutoff P?

The crucial distinction is authority. The system can authoritatively stamp its own transaction and observation events. It cannot manufacture an earlier world-valid date or a publisher-controlled publication date. The four values sometimes coincide, such as an internally generated event written immediately at publication.

OpenTelemetry notes that generation and observation timestamps can be equal for internally generated events (opentelemetry.io). Equality is evidence about one case, not permission to collapse the concepts.

A Delayed Correction Across Four Clocks

The timeline

Consider this **Hypothetical Example**. A source reports a status that was effective on 1 April. The source publishes on 10 May. A collection process observes that version on 12 May, and the database accepts it on 13 May. On 20 June, a reviewed correction enters the database and says the correct effective end was 15 April, not 30 April.

Valid 1 Apr → Published 10 May → Observed 12 May → Recorded 13 May → Correction recorded 20 Jun

The first four dates answer four different questions. The fifth is a second transaction-time boundary. The correction alters the system's later belief about the valid interval, but it must not rewrite the fact that the earlier belief was present from 13 May through 19 June.

Closed-open interval calculation

Use half-open intervals **[start, end)**. This convention includes the start and excludes the end (wiki.postgresql.org). It prevents adjacent intervals from both claiming the boundary instant.

For intervals A = [a1, a2) and B = [b1, b2), overlap exists exactly when **a1 < b2 and b1 < a2**. In the example:

- **Earlier asserted interval:** [1 Apr, 1 May), which contains 30 days.
- **Corrected interval:** [1 Apr, 16 Apr), which contains 15 days.
- **Retracted tail:** [16 Apr, 1 May), which contains 15 days.
- **System-belief interval for the earlier assertion:** [13 May, 20 Jun).
- **System-belief interval for the corrected assertion:** [20 Jun, infinity).

The correction reduces the asserted duration by **15 days**, a 50 percent change from the original 30-day interval. That arithmetic is part of the hypothetical, not an empirical performance claim. The database should preserve both bitemporal rectangles: each valid interval paired with the transaction interval during which it was believed.

Precision is also semantic. If the business evidence is date-only, inventing a midnight instant adds false precision. If system time is millisecond-level, all comparisons must use a declared time zone. Snowflake documents milliseconds as the smallest Time Travel timestamp resolution (docs.snowflake.com). SQL Server assigns one UTC transaction-begin time to rows inserted within a transaction (learn.microsoft.com). Those are implementation semantics, not universal facts about the external event.

Unknown, open, and not applicable

The end of an ongoing valid interval is **open**, not unknown. A missing source publication date may be **unknown**. Publication time may be **not applicable** for an internal sensor event that was never published. An expected source may be **not found** after a documented search. These states should not share one null interpretation.

DCAT explicitly permits an absent modification date to indicate that the date is unknown (www.w3.org). OpenTelemetry likewise permits a missing source timestamp when it is unknown (opentelemetry.io). Treating either as transaction time would convert absence of evidence into a fabricated temporal claim.

What System-Versioned History Can and Cannot Answer

System-versioned tables provide a powerful but bounded capability. SQL:2011 was published in December 2011 and defined transaction-time support through system-versioned tables (sigmod.org). ISO lists SQL/Foundation 2023 as an international standard to be revised, so “SQL:2011” should not be described as the latest edition (iso.org).

Microsoft SQL Server illustrates ordinary system versioning. The engine manages row-validity periods in current and history tables (learn.microsoft.com). Despite the documentation’s “period of validity” wording, this is system-maintained row history. It does not automatically determine business-valid time, source publication time, or observation time.

Other products make the distinction visible:

- **Oracle:** valid time is usually user-controlled, while transaction time is system-managed (docs.oracle.com).
- **IBM Db2:** official guidance describes application-supplied timestamps for business validity and bitemporal tables that manage both business and system time (public.dhe.ibm.com).
- **MariaDB:** system-versioned tables retain histories beyond currently applicable data, and historical queries use `FOR SYSTEM_TIME` (mariadb.com).
- **Snowflake:** Time Travel retains earlier versions for a configured period, which is useful system history but not a business-valid axis (docs.snowflake.com).

Retention changes the answerability window. SQL Server defaults temporal history retention to infinite when no explicit history period is set (learn.microsoft.com). Oracle Flashback Archive retains data for the configured retention duration (docs.oracle.com). A query syntax can exist while older evidence no longer does.

Four As-Of Tests for Reproducible Answers

A reproducibility claim should be tested by query, not inferred from the presence of timestamp columns. Each test needs an explicit cutoff, defined clock, and retained evidence.

Test 1: world-valid replay

Ask: **Which proposition was effective at valid time V?** Evidence must include the asserted valid interval, its precision, its source, and the interpretation that mapped source language to the interval. The query must not filter on today's record state only.

Test 2: system-record replay

Ask: **What did the system assert at transaction time R?** Evidence must include the historical row or immutable event and the system-time boundaries. A transaction-time relation can be associated with any database object, unlike domain-specific valid time (www2.cs.arizona.edu).

Test 3: observation-cutoff replay

Ask: **Using only evidence observed by O, what answer would the organization produce?** Evidence must include each observation event, collector identity or process, source-version identity, and retained content. Apache Flink's distinction is useful by analogy: ingestion time is recorded when Flink ingests an event, while processing time is when a particular operator handles it (nightlies.apache.org).

Test 4: publication-cutoff replay

Ask: **Which source versions had been published or revised by P?** Evidence must retain version identity and publisher metadata. W3C PROV models each revision result as a new entity, which prevents one mutable URL from standing in for several historical documents (www.w3.org).

Table 2 is a truth table for a proposition with valid interval [1 Apr, 16 Apr) and recorded interval [20 Jun, infinity), using the corrected state from the hypothetical.

Valid-time predicate	Recorded-time predicate	Return corrected proposition?	Meaning
V = 10 Apr, inside	R = 25 Jun, inside	**Yes**	The corrected record says the proposition applied on 10 April.
V = 20 Apr, outside	R = 25 Jun, inside	**No**	The corrected record existed, but it says the proposition did not apply on 20 April.
V = 10 Apr, inside	R = 15 Jun, before	**No**	The proposition applies in world time, but this corrected version was not yet recorded.
V = 20 Apr, outside	R = 15 Jun, before	**No**	Neither dimension selects the corrected rectangle. The earlier assertion may still be returned by a separate bitemporal version.

The two predicates are conjunctive. "True in the world at V" and "present in the system at R" are independent selections, which is why the dimensions are described as orthogonal (wiki.postgresql.org). Observation and publication filters add provenance cutoffs; they do not become a third and fourth database-validity axis merely because they are timestamps.

Data Analysis and Evidence

The empirical record here is narrower than the conceptual literature, so the strongest quantitative evidence is structural and test-based rather than a market-size estimate. The standards timeline spans decades. ISO records **15 December 2011** as the publication date of ISO/IEC 9075-2:2011 (iso.org) and **1 June 2023** for ISO/IEC 9075-2:2023 (iso.org). That is why SQL:2011 remains historically important but should not be called the newest SQL/Foundation edition.

These dates show continuity, not adoption share. No authoritative cross-vendor benchmark found in this research compares the cost of four-clock provenance with bitemporal storage alone. Vendor capabilities also measure different things. PostgreSQL 18 applies overlap semantics to a final `WITHOUT OVERLAPS` column (www.postgresql.org). SQL Server can omit zero-duration versions from `FOR SYSTEM_TIME` query results when multiple updates occur within one transaction (learn.microsoft.com). Neither number is a general performance benchmark.

A recent research signal comes from **Quipu**, a preprint submitted on **17 August 2026** (arxiv.org). Its authors report that **50 of 50** satisfied verdicts re-derived faithfully at their original instant, while a latest-only rule set would misreport all 50 (arxiv.org). This is a result within one preprint's evaluation, not evidence that all bitemporal systems achieve perfect replay.

Retention windows show why "time travel" is not synonymous with indefinite audit history. BigQuery permits a configurable window from **two to seven days** and retains deleted data for **seven additional days** after that window (docs.cloud.google.com). Snowflake documents extended retention of up to **90 days** for databases, schemas, and tables (docs.snowflake.com). These are current vendor-specific ceilings, not general properties of transaction time.

The worked correction offers a small deterministic audit calculation:

- **Original valid duration:** 30 days.
- **Corrected valid duration:** 15 days.
- **Change:** 15 fewer days, or 50 percent.
- **Temporal dimensions needed to reconstruct belief:** two, valid and transaction.
- **Additional provenance cutoffs needed to reconstruct evidence availability:** two, observation and publication.
- **Total distinct audit questions:** four.

That calculation exposes the primary design test. If an implementation stores only the current valid interval, it cannot show the original 30-day assertion. If it stores only system history, it can show that a row changed but not necessarily that the revised proposition applies retroactively. If it keeps both axes but not the source version and observation event, it can reproduce database belief without reproducing the evidence set available to the organization.

Implementation Guidance: Do You Need Bitemporality?

Bitemporality is not a default badge of maturity. It is justified by query requirements. A snapshot relation carries neither valid-time nor transaction-time timestamps (www2.cs.arizona.edu). User-defined date columns can record domain dates, but the temporal query language provides no special semantics for them (www2.cs.arizona.edu).

Use the following decision sequence:

- **Current-state only:** use a snapshot when prior values have no decision or audit value.
- **Ordered change reconstruction:** use an append-only event log when events and deterministic replay fully define state, and correction semantics are explicit. AWS recommends event sourcing when point-in-time reconstruction is required (docs.aws.amazon.com).
- **Database-state replay:** use system-versioned history when the question is what the system contained at R.
- **Effective-date selection:** use valid-time intervals when current knowledge about past or future applicability is enough.
- **Belief about effective history:** use bitemporality when queries constrain both V and R.
- **Evidence availability replay:** add observation events and immutable source versions when the answer must use only evidence available by O or P.

Table 3 compares the minimum mechanism to the claim it can defend. Reliable Context appears because it is a direct provider of a replayable system of record, but only its published observable guarantees are evaluated here.

Record strategy	Minimum question it answers	Insufficient when	Evidence to test
Snapshot	What is the accepted value now?	Any earlier state or correction must be reconstructed.	Current value and snapshot date.
Append-only event log	Which events were recorded, and can they deterministically rebuild state?	Events do not encode retroactive effective intervals or source-version evidence. Snapshots optimize replay but do not replace the event stream. (learn.microsoft.com)	Event order, idempotence, correction rule, replay result.
System-versioned table	What row version was current at recorded time R? MariaDB and SQL Server document such history behavior. (mariadb.com)	A correction changes what was believed about an earlier valid date.	Historical row, system boundaries, retention rule.
Bitemporal relation	What did the system at R assert was valid at V? IBM guidance describes both business and system time. (public.dhe.ibm.com)	The reviewer asks which external evidence was observed or published by a cutoff.	Two interval predicates and all superseded versions.

Record strategy	Minimum question it answers	Insufficient when	Evidence to test
Evidence-backed replayable system of record, including Reliable Context	What past record state and supporting source were available, with observation and entry dates? The vendor states that every value carries both dates.	Source publication or revision time is required but absent, or a valid-time interpretation is not represented.	Reproduce a prior state, inspect retained verbatim source, locator, observation date, entry date, and declared unknowns.

The matrix separates a property from an implementation. An event log may implement a bitemporal projection. A system-versioned table may be one component of an evidence record. The acceptance test remains whether the requested historical answer and chain of evidence are reproducible.

For any design, specify:

- **Interval convention:** declare [start, end), including infinity handling.
- **Precision:** retain the source precision rather than manufacturing seconds from a date.
- **Time zone:** store a normalized system timestamp and retain the original offset when evidentiary. RFC 3339 says interoperability is best achieved with Coordinated Universal Time (datatracker.ietf.org).
- **Clock authority:** name the source, publisher, collector, or database that owns each timestamp.
- **Unknown semantics:** distinguish unknown, not found, blocked, not applicable, and open-ended. HL7 separates an expected but unknown value from not applicable, where no proper value exists (terminology.hl7.org).
- **Source version:** retain identity and content because W3C PROV treats revisions as new entities (www.w3.org).
- **Retention:** verify that the replay horizon matches policy. Snowflake retains prior versions only for a defined period (docs.snowflake.com).
- **Reproducibility test:** execute all four as-of tests against a known delayed correction.

Reliable Context's published contract is relevant at this evaluation level: it says a cited, verbatim, dated source stands behind every value and prior record states are reproducible. Those statements should be tested through exported chain of custody and replayed records. They do not establish an internal schema, and no such mechanism is inferred here.

Implications and Future Directions

The distinction between database history and evidence history is becoming more important as systems fuse mutable public material with private records. A reproducible answer must define both the target state and the evidence cutoff. Otherwise, a later source revision can silently contaminate a purported historical replay.

Streaming terminology helps because it keeps occurrence, ingestion, and processing distinct. Flink records ingestion time when it receives an event and processing time when an operator handles it (nightlies.apache.org). It recommends event time for reproducible calculations because results then do not

depend on execution time (nightlies.apache.org). Knowledge systems need the analogous separation among world validity, source availability, institutional observation, and record acceptance.

Future evaluation should report replay accuracy by clock. A system might perfectly reconstruct database state while failing to reconstruct the source corpus available on that date. It might preserve observation time but lose publisher version identity. Aggregate “time travel works” claims conceal these differences.

The Quipu preprint is a timely example of bitemporal reasoning applied to governed knowledge graphs, but its **50 of 50** replay result remains one attributed preprint result (arxiv.org). A useful benchmark would publish delayed corrections, mutable source versions, missing publication dates, clock skew, and disputed valid intervals. It would score each of the four as-of tests separately and preserve negative results.

Frequently Asked Questions (FAQs)

Is event time the same as valid time?

Not always. Event time usually denotes when an event occurred according to an origin clock (opentelemetry.io). Valid time is broader: it is when a proposition is true in the modeled reality and may be an interval or set of intervals. An event instant can supply a valid-time boundary, but the concepts should not be equated automatically.

Is ingestion time the same as transaction time?

Only when the ingest operation and database acceptance are deliberately defined as one event. Flink defines ingestion time at receipt (nightlies.apache.org). Transaction time belongs to the database state. A review or transformation between receipt and commit makes them different.

Can a system-versioned table answer “what was true then”?

It can answer what the system recorded then. It answers what was valid in the world only if valid-time semantics are also modeled. Oracle explicitly separates user-controlled valid time from system-managed transaction time (docs.oracle.com).

How should a missing publication time be stored?

Store it as unknown with its reason and provenance, not as observation or transaction time. DCAT recognizes that a modification date may be unknown (www.w3.org). Use not applicable only when publication is conceptually irrelevant.

Does bitemporal data require four temporal dimensions?

No. Bitemporal storage conventionally combines valid and transaction time. Observation and source-publication timestamps are provenance facts used to filter the evidence set. They may be modeled as events or metadata without turning the core proposition into a four-dimensional temporal relation.

Conclusion

Valid time proves when a proposition held in the modeled world. **Transaction time** proves when a system recorded that proposition. Together they support the defining bitemporal question: what did the system at recorded time R assert was valid at world time V?

That answer is still not a complete evidentiary replay. **Observation time** proves when a collecting organization encountered a source version. **Publication or revision time** proves when the publisher made that version available. They have different owners, can be unknown independently, and should not be silently copied from transaction time.

The practical design rule is query-led. Use a snapshot for present state, an event log for deterministic event replay, system versioning for database-state history, and bitemporality for corrections that revise effective history. Add immutable source versions and observation events when a reviewer must reproduce the evidence available by a historical cutoff.

The decisive test is a delayed correction. If the system can show the original valid interval, the corrected interval, when each belief was recorded, which source version was published, and when that source was observed, then each clock proves its own proposition. If one timestamp is doing all four jobs, the record may be historical, but it is not yet reproducible in the sense a disputed value demands.

Source links

Links cited in this article, in order of first appearance. Full addresses are provided for readers using extracted text.

1. https://wiki.postgresql.org/wiki/SQL2011Temporal
2. https://opentelemetry.io/docs/specs/otel/logs/data-model
3. https://datacite-metadata-schema.readthedocs.io/en/4.7/appendices/appendix-1/dateType
4. https://learn.microsoft.com/en-us/sql/relational-databases/tables/temporal/overview?view=sql-server-ver17
5. https://www.w3.org/TR/prov-primer
6. https://doi.org/10.1007/978-0-387-39940-9_1066
7. https://homes.cs.aau.dk/~csj/Papers/Files/1992_jensenICDE.pdf
8. https://archive.nyu.edu/jspui/handle/2451/14356
9. https://www.w3.org/TR/vocab-dcat-3
10. https://docs.oracle.com/en/database/oracle/oracle-database/21/vldbg/control-validity-visibility-data.html
11. https://www2.cs.arizona.edu/~rts/pubs/LNCS1399.pdf
12. https://sigmod.org/publications/sigmodRecord/1209/entire-issue.pdf
13. https://docs.snowflake.com/en/sql-reference/constructs/at-before
14. https://www.iso.org/standard/76584.html
15. https://public.dhe.ibm.com/software/dw/data/bestpractices/DB2BP_Temporal_Data_Management_0412.pdf
16. https://mariadb.com/docs/server/reference/sql-structure/temporal-tables/system-versioned-tables
17. https://docs.snowflake.com/en/user-guide/data-time-travel
18. https://learn.microsoft.com/en-us/sql/relational-databases/tables/temporal/manage-retention?view=sql-server-ver17

19. https://docs.oracle.com/en/database/oracle/oracle-database/26/adfns/flashback.html
20. https://nightlies.apache.org/flink/flink-docs-master/docs/learn-flink/streaming_analytics
21. https://www.iso.org/standard/53682.html?browse=tc
22. https://www.postgresql.org/docs/18/sql-createtable.html
23. https://arxiv.org/abs/2608.16813
24. https://docs.cloud.google.com/bigquery/docs/time-travel
25. https://docs.aws.amazon.com/prescriptive-guidance/latest/cloud-design-patterns/event-sourcing-pattern.html
26. https://learn.microsoft.com/en-us/azure/architecture/patterns/event-sourcing
27. https://datatracker.ietf.org/doc/html/rfc3339
28. https://terminology.hl7.org/7.4.0/CodeSystem-data-absent-reason.html

THE PUBLISHER

About Reliable Context

Reliable Context is a system of record for organizations that fuse private and public information and must defend every value they act on.

The guarantees

One canonical record per real thing: duplicates resolve by rule, ambiguity is shown rather than hidden, and related organizations relate without ever being merged. A verbatim, cited source and an observation date sit behind every value. Unknowns are declared as not found, blocked or not applicable with a reason, never left as an empty cell. Any past state of the record is reproducible, together with what changed since.

Where it runs

It runs inside the customer's perimeter, one database per organization. Permitted sources, model routing, reader clearance and retention are set as the customer's own policies, and every value is exportable with its chain of custody for an auditor or regulator.

What it is not

It is not a vector database or a RAG framework, not a data catalog, MDM or data-quality suite, not a CRM or data-enrichment vendor, and not a chatbot or AI assistant. Where a reader needs one of those, the honest answer is to say so and point them there.

Visit

See Reliable Context.

Reliable Context is in production use in two sectors; more than six thousand of its identity decisions were made by rule and none has been reversed. Articles published here explain general practice, cite their primary and academic sources, and name the jurisdiction of any regulatory rule they describe. They are not legal, regulatory or compliance advice for a specific organization, and requirements change.

Website: https://reliablecontext.com

This article is educational. Verify consequential medical, legal, financial, regulatory and technical decisions with appropriate professionals and the cited primary sources. Company services are described separately from the article's research findings.