CatDB: Toward a Compositional Semantic Data Layer

A conservative synthesis of model results, bounded computation, and open systems work

Matthew Long
The YonedaAI Collaboration
YonedaAI Research Collective
Chicago, IL
matthew@yonedaai.com · https://yonedaai.com

23 July 2026

Provisional status. This synthesis is not an accepted paper and is not a systems result. Internally accepted companion manuscripts establish paper models, not implementations. The only current executable evidence used here is a closed shared-fixture surface for a finite semantic kernel. No distributed runtime, optimizer, connector suite, federation engine, materialization engine, MDReader application, or scale benchmark is established.

Abstract

Integration work fails in a familiar way. A customer identifier means one thing in an application database, another in a billing service, and something less precise in an analytical export. The joins still run, but their meaning lives in migration scripts, transformation jobs, naming conventions, and the memory of the people who wrote them. When a schema or policy changes, the team must reconstruct which decisions were copied into which pipeline.

This paper synthesizes a research program around a narrower response. We define CatDB as a semantic multimodel data platform in which schemas, mappings, migrations, identities, and provenance are first-class, versioned, executable objects. "Executable" means that an artifact has a typed interpretation, validation obligations, and a route to compilation or evaluation. It does not mean that the current repository implements every such route.

The synthesis answers five questions. First, CatDB's candidate distinction is not a new physical data model. It is one versioned contract connecting several models while retaining each model's residual semantics. Second, typed mappings can make migration direction, coverage, loss, ambiguity, authority, and physical intent reviewable before execution. Third, mappings may replace duplicated semantic logic or generate physical jobs only when the required transformations factor through an adequate domain. Extraction, cleaning, transport, loading, scheduling, retries, and pair-specific rules often remain. Fourth, competitive physical execution is an open question. Accepted papers specify compilation, federation, and cost contracts, but no current compiler,

connectors, optimizer, or benchmark establish their performance. Fifth, realistic distributed guarantees are operation specific. Immutable proposals and admitted monotone updates may converge; single-valued activation, authority, membership, and noncommutative migration transitions still require conflict preservation or coordination.

Evidence is deliberately uneven. The accepted Slice 4 record reports that Rust and independently implemented Haskell programs agree on 29 semantic cases in a closed 33-fixture profile; four cases remain structural only. Selected Lean declarations cover named path and mapping laws, not the runtime. Everything beyond that bounded kernel remains a paper contract, an engineering hypothesis, or an obstruction. The defensible thesis is therefore conditional: CatDB moves some integration meaning into typed, composable, versioned mappings, while federation, materialization, ETL, and migration remain execution strategies with ordinary systems costs and limits.

Contents

1 Problem, thesis, and evidence cut

1.1 Plain-language problem

A company has three definitions of "active customer." The application means an account that can sign in. Billing means an account with an open contract. Analytics means an account with an order in the last ninety days. The databases are not broken. Each definition serves a real purpose. The failure begins when a pipeline treats the three definitions as interchangeable and records the choice only in code.

A semantic layer cannot make the disagreement disappear. It can give the disagreement a durable form. A mapping can state which source field denotes which domain concept, under which snapshot and authority, with which loss and coverage. A migration can state how a version changes. An identity decision can retain evidence and uncertainty. Provenance can state which source records and semantic artifacts justify a result. Those objects can then be reviewed, composed, invalidated, and compiled.

This is the research program's central thesis:

> CatDB moves integration meaning from bespoke pipelines into typed, composable, versioned mappings, while federation and materialization remain execution strategies.

The qualification matters. A typed mapping does not clean a malformed postal address, move bytes across a network partition, or decide who may merge two customer records. It can describe and constrain those operations. The physical work still has to run.

1.2 Five synthesis questions

The manuscript answers the five questions fixed by the accepted synthesis outline:

Q1. What does CatDB model that relational, graph, document, and vector databases do not model together?

Q2. How do typed mappings improve the description and execution of schema evolution?

Q3. Which integration workloads can mappings replace, generate, or leave unchanged?

Q4. Can semantic compilation preserve competitive physical execution?

Q5. Which distributed consistency and convergence guarantees are realistic?

Table 1 gives the short answers. Later sections state the assumptions and evidence behind them.

Table 1: Five answers at the current evidence cut.

Question	Conservative answer	Current status
Q1	A versioned semantic contract can connect heterogeneous models while recording mappings, migrations, identity, and provenance as objects. It does not subsume every source model.	Accepted paper models; bounded kernel computation
Q2	Typed mappings expose direction, coverage, loss, ambiguity, authority, and obligations before a migration mode is selected.	Paper reasoning; finite Δ computation only
Q3	Reusable semantic logic may factor through a domain; physical and pair-specific work often remains.	Conditional counting result; maintenance effect open
Q4	Native compilation is plausible only if result equivalence and overhead gates pass. No current artifact establishes that.	Obstructed
Q5	Guarantees must be selected per operation. Some monotone or immutable updates may converge; other transitions require coordination.	Accepted paper model; runtime Obstructed

1.3 What "accepted" means here

All eighteen companion manuscripts are internally accepted as paper artifacts: P1 through P18 [Long and Collaboration, 2026a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r]. Acceptance means that each manuscript's definitions, citations, nonclaims, build, and review gate closed at its recorded source. It does not turn a paper proposition into runtime evidence.

This synthesis remains unreviewed and provisional. Companion-paper acceptance closes one dependency gate but does not review or accept the synthesis.

2 Evidence discipline and central definition

2.1 Two status axes

The research program separates the kind of evidence from the current workflow outcome. A claim may be supported by prior literature, paper reasoning, machine checking, bounded computation, or an experiment. Independently, its operational status may be Proved, Conditional, Computational, Open, or Obstructed. This prevents a well-reviewed design from being reported as an implementation.

2.2 Central definition

Definition 2.1 (CatDB)**.** CatDB is a semantic multimodel data platform in which schemas, mappings, migrations, identities, and provenance are first-class, versioned, executable objects.

Each word imposes a boundary.

Table 2: Evidence classes used in this synthesis.

Class	What it establishes	What it does not establish
Primary prior literature	A result or capability under the cited source's model and assumptions	A CatDB-specific result or implementation
Accepted Paper Model	A reviewed mathematical or systems contract in a companion manuscript	Executable behavior, performance, or scale
Paper Reasoning	A definition, derivation, or proof under stated assumptions in this source	Kernel checking or implementation conformance
Machine-Checked	An exact named declaration under its audited assumptions	The Rust source, SQL backends, connectors, networks, or the full engine
Computational	Observed output of named programs on a closed input surface	General proof, unseen cases, production readiness, or performance
Experiment	Measured behavior with baselines, controls, raw data, and failures retained	Universal behavior outside the tested workload

- *Semantic* means that declared relationships, equations, coverage, loss, authority, and evidence are part of the contract rather than only comments around a physical job.
- *Multimodel* means that several source models can participate without claiming that one finite category subsumes all relational, graph, RDF/OWL, document, vector, or full-text semantics.
- *First-class* means that an artifact has identity, a typed representation, references, validation, and provenance.
- *Versioned* means that every decision is bound to immutable revisions and can be compared, replayed, superseded, or rejected.
- *Executable* means that an artifact names an interpretation and the obligations required for validation, compilation, or evaluation. It is a design property. Current execution exists only for the bounded kernel described in Section 11.

Definition 2.2 (Semantic commit)**.** A semantic commit is a content-addressed tuple

$$K = (\mathcal{S}, \mathcal{M}, \mathcal{U}, \mathcal{I}, \mathcal{P}, \mathcal{Q}, \mathcal{W}, \mathcal{X}),$$

where $\mathcal{S}$ contains schema revisions, $\mathcal{M}$ typed mapping revisions, $\mathcal{U}$ migration declarations, $\mathcal{I}$ identity and reconciliation decisions, $\mathcal{P}$ provenance and policy profiles, $\mathcal{Q}$ query definitions, $\mathcal{W}$ workspace references, and $\mathcal{X}$ physical execution profiles. A commit is admissible only when all referenced revisions exist and the selected cross-object obligations are discharged or explicitly unresolved.

This tuple is a synthesis contract, not an implemented catalog type. The internally accepted workspace paper gives a more detailed proposed commit model [Long and Collaboration, 2026i]; no current workspace runtime constructs or publishes such a value.

2.3 Schemas, mappings, and physical realizations

For the finite profile used by the current kernel, a schema presentation is written

$$S = (E, G, A, R),$$

with entities E, generating functional arrows G, typed attributes A, and equations R between parallel paths. Categories as schemas, instances as functors, and functorial migration are established prior work [Spivak and Kent, 2012, Spivak, 2012, Spivak and Wisnesky, 2015, Schultz and Wisnesky, 2017, Schultz et al., 2017]. Executable categorical integration also predates CatDB [CategoricalData project, 2026]. The candidate delta is the versioned operational contract and evidence boundary, not the categorical foundation.

Definition 2.3 (Typed semantic mapping)**.** A typed semantic mapping revision is a tuple

$$m = (s, t, F, \omega, \gamma, \lambda, \alpha, \pi),$$

where s and t identify source and target schema revisions; F assigns admitted entities, paths, and attribute expressions; ω selects a data operator or read interpretation; γ records coverage and completeness; λ records loss, partiality, and ambiguity; α records authority and write policy; and π records provenance and validation evidence.

Schema-arrow direction does not determine data-flow direction. For a schema mapping $F : S \to T$, the established pullback operator is contravariant,

$$\Delta_F : \mathrm{Inst}(T) \longrightarrow \mathrm{Inst}(S),$$

while Σ_F and Π_F are covariant under their own assumptions [Spivak, 2012]. A forward ETL or data-exchange claim must name the selected operator, query rewrite, target construction, or materialization semantics. It cannot be inferred from F alone.

Definition 2.4 (Realization)**.** For a semantic request q at commit K, a physical realization $r \in \mathcal{R}(q, K)$ is a backend plan, federated plan, materialization, incremental refresh, or generated transfer job that carries a certificate of its semantic dependencies and residual obligations.

Every realization is subordinate to a declared result contract. If $[\![q]\!]_K$ is the abstract answer and N_K is the profile-specific normalization, an exact realization must eventually satisfy

$$N_K(\mathrm{run}(r)) = [\![q]\!]_K. \tag{1}$$

Equation (1) is an evaluation obligation. No current compiler, federation engine, connector, or materialization engine discharges it.

3 Running example: customer, order, and policy evidence

3.1 Four sources and two consumers

The example uses four heterogeneous sources:

1. a SQLite application database with `customer`, `organization`, and `order_row`;
2. a PostgreSQL operational database with `account`, `purchase`, regional residency tags, and billing status;
3. a JSON product API with string SKUs, prices in minor currency units, and source-specific availability codes; and
4. Markdown policy and evidence records that name approval, retention, and exception decisions.

The proposed domain contains customers, organizations, orders, products, local references, identity assertions, and provenance links. One consumer is an analytical model. The other is a proposed MDReader view over evidence and policy records. MDReader is used only as a motivating consumer. The repository has no current MDReader application result.

3.2 One order through the semantic objects

SQLite records customer `c-17` with email `alex@example.test`. PostgreSQL records billing account `acct-944` with the same normalized email and an open contract. The JSON API describes product `SKU-9`. A Markdown decision says that billing records may contribute to a revenue report but may not reveal the raw email.

A CatDB-style account of the example would retain:

- a mapping from the SQLite customer and order schema into the domain;
- a distinct mapping from PostgreSQL accounts and purchases;
- a conversion from integer minor units to the domain's money value, with currency and rounding assumptions;
- a candidate identity link between `c-17` and `acct-944`, followed by an authority-bearing decision if they are confirmed to denote the same customer;
- provenance connecting the revenue row to source records, snapshots, mappings, the identity decision, and the policy revision; and
- a physical plan that either queries both live databases or reads a materialized integration result.

The shared email is evidence, not identity. Record linkage and entity resolution have long treated matching as an uncertain inference problem [Fellegi and Sunter, 1969, Benjelloun et al., 2009]. A vector index could rank the pair as a candidate. It could not authorize the merge.

3.3 A schema change

Suppose the SQLite application splits `customer.name` into `given_name` and `family_name`. A migration artifact can say:

- which immutable schema revisions it connects;
- whether the old-to-new transformation is total;
- whether reconstructing the old display name is ambiguous;
- which consumers require the old formatting;
- whether execution is eager, lazy, virtual, or hybrid;
- which materializations and identity features become stale; and
- which provenance record explains the choice.

The declaration does not split every name correctly. It makes the unresolved cases and selected policy visible before a physical rewrite or compatibility view runs.

3.4 Federation, materialization, and replay

The analytical query can be realized in at least two ways. A federated plan pushes exact filters to SQLite and PostgreSQL, transfers selected rows, applies local reconciliation, and joins them. A materialized plan reads a maintained domain view. Both strategies depend on the same mapping and identity revisions, but their freshness, availability, provenance, and resource costs differ.

Now place the semantic catalog at two sites. Adding an immutable proposal may be replay-safe. Activating the new mapping as the single current mapping is a different operation. Two sites cannot each publish a different active mapping and then recover one value by set union. The system must retain a conflict or coordinate the transition. The example therefore reaches the same boundary as the distributed paper models: typed structure can expose the conflict, but it cannot remove the network and authority problem.

4 Q1: What is modeled together?

4.1 The answer

CatDB's candidate contribution is a shared version and obligation boundary across schema structure, mappings, migrations, identity decisions, provenance, queries, workspaces, and physical realizations. Existing systems model many of these pieces well. The claim is about their joint treatment as referenced objects, not first invention of any individual piece.

The relational model separates logical relations from physical access paths [Codd, 1970]. Entity-relationship modeling supplies a conceptual layer [Chen, 1976]. RDF and OWL define graph and ontology semantics [W3C RDF Working Group, 2014, W3C OWL Working

Group, 2012]; GQL standardizes a property-graph language [ISO/IEC JTC 1/SC 32, 2024]; AsterixDB provides a typed semistructured model [Alsubaiee et al., 2014]. None of those facts licenses the claim that a finite category subsumes the others.

4.2 Preservation rather than subsumption

For each source model D_i, a mapping into a shared domain must produce a preservation report:

$$\text{Preserve}(D_i, m_i) = (P_i, L_i, U_i),$$

where P_i lists represented semantics, L_i lists declared loss or approximation, and U_i lists unsupported or unknown semantics. A property graph may retain edge identity and multiplicity that the finite functional profile cannot express. RDF blank nodes and open-world entailment need their own boundary. A document may preserve order and mixed content. A vector index has approximate retrieval behavior rather than an identity theorem.

Obligation 4.1 (No silent residual)**.** A mapping may be admitted as exact only if every source feature required by the target request is represented or discharged by a named residual. An unrepresented required feature yields a conditional, partial, unsupported, or unknown result.

This obligation is proposed. The current finite kernel validates a much smaller schema, path, mapping, instance, and pullback profile. It does not compute general multimodel preservation reports.

4.3 Identity and provenance complete the account

Mappings relate schemas and data interpretations. They do not decide that two records denote the same entity. The internally accepted identity paper therefore separates local references, evidence, method outputs, and authorized reconciliation decisions [Long and Collaboration, 2026g]. The accepted provenance paper separates source identity, relational derivation, semantic artifact versions, execution evidence, and authority decisions [Long and Collaboration, 2026m]. Why- and where-provenance, provenance semirings, and PROV interchange are established foundations [Buneman et al., 2001, Green et al., 2007, Lebo et al., 2013].

Together, those objects let a result identify the source snapshot, mapping revision, identity decision, policy, and plan behind a row. The paper model is accepted. No current provenance runtime, completeness checker, exporter, or overhead measurement establishes that behavior.

5 Q2: How typed mappings change schema evolution

5.1 Description before execution

A migration script lists commands. A typed migration package can also record the old and new schema identities, the mapping direction, the selected data operator, coverage, loss,

ambiguity, authority, dependencies, provenance, and physical intent. This is useful even when a conventional engine executes the actual change.

The distinction is familiar in adjacent work. Data exchange gives mappings, solutions, and certain-answer semantics [Fagin et al., 2005a]; schema-mapping composition has language-dependent closure limits [Fagin et al., 2005b]; model management treats mappings as values [Bernstein and Melnik, 2007]. CatDB's internally accepted migration paper packages those concerns into one immutable versioned artifact [Long and Collaboration, 2026c]. That package is a candidate operational synthesis, not a new general migration theory.

Definition 5.1 (Migration declaration)**.** A migration declaration between schema revisions S_v and S_w is

$$\mu_{v,w} = (m_{v,w}, o, c, \ell, a, d, p),$$

where $m_{v,w}$ is a typed schema relation, o selects an admitted data operator, c states coverage, ℓ states loss and ambiguity, a states authority, d lists dependent artifacts, and p records provenance. A physical mode is a separate value in

$$\{\text{eager}, \text{lazy}, \text{virtual}, \text{hybrid}, \text{mixed-version}\}.$$

Changing the physical mode must not silently change the declared meaning. An eager rewrite and a virtual compatibility view can still differ in snapshot, collation, null, multiplicity, and failure behavior. They are equivalent only after a normalized comparison under one frozen profile.

5.2 Composition is partial

For mappings $m_1 : S \to T$ and $m_2 : T \to U$, write

$$m_2 \odot m_1 \downarrow$$

when the endpoint revisions match, the selected mapping language is closed under the composition, required equations are preserved in the admitted equality fragment, and coverage, loss, authority, and provenance compose without an unresolved contradiction. Otherwise the composition returns a typed partial, ambiguous, unsupported, or unknown outcome.

Counterexample 5.2 (Type correctness is not business correctness)**.** A mapping sends a source `customer_id` string to a target `customer_id` string. The types align. The source value is local to one tenant while the target value is global. A type checker accepts the scalar assignment, but the mapping is semantically wrong unless tenant scope enters the identity contract.

5.3 What current computation establishes

The accepted Slice 4 release and terminal code review record validation of 33 closed fixtures. Rust and an independently implemented Haskell process compared 29 semantic cases, retained four structural-only cases, and reported zero disagreements. The semantic surface includes finite instance validation and two finite pullback Δ successes, several negative validation cases, and exact unsupported outcomes for Σ and Π outside the profile.

Evidence status. COMPUTATIONAL for the 29 admitted semantic cases at the accepted Slice 4 cut. General migration, physical execution, online cutover, rollback, information-loss computation, and schema-diff execution are not established.

The result is useful because it proves less than the slogan "migration is implemented." It shows exact agreement over one closed fixture surface. The independent Slice 4 review records possible diagnostic-order divergences on unfixtured inputs. Agreement is not a theorem of program equivalence.

6 Q3: What mappings replace, generate, or leave unchanged

6.1 A conditional factorization

Let I be a finite set of sources, J a finite set of consumers, and $A \subseteq I \times J$ the connection relation that the workload actually requires. A direct design supplies a transformation T_{ij} for each $(i, j) \in A$. A mediated design supplies source mappings

$$a_i : S_i \longrightarrow D$$

and consumer mappings

$$b_j : D \longrightarrow C_j.$$

The pair (i, j) factors exactly when the selected integration semantics satisfy

$$T_{ij} \equiv b_j \odot a_i. \tag{2}$$

An exception E_{ij} remains when the domain omits required pair-specific meaning or the mapping language cannot express the transformation.

Proposition 6.1 (Conditional base-mapping count)**.** If $A = I \times J$, every required T_{ij} factors as in (2), the admitted mapping language is closed under those compositions, and no pair-specific exception is required, then the mediated description needs $|I| + |J|$ base mappings rather than $|I||J|$ pair mappings.

Proof. Choose one a_i for each $i \in I$ and one b_j for each $j \in J$. There are $|I| + |J|$ such mappings. By assumption, every required pair is the admitted composition $b_j \odot a_i$, so no additional pair mapping is needed. □

The proposition is elementary accounting. It does not show lower maintenance time. If A is sparse, $|A|$ may already be smaller than $|I| + |J|$. If the domain collapses a distinction, the factorization fails. If an exception remains, it still has to be authored and repaired.

6.2 The complete cost ledger

A fair comparison includes more than mapping count. One conservative accounting is

$$\begin{aligned} C_{\text{med}} = &\sum_{i \in I_A} C(a_i) + \sum_{j \in J_A} C(b_j) + C_D + C_R \\ &+ \sum_{(i,j) \in A} C(E_{ij}) + C_X, \end{aligned} \tag{3}$$

where C_D is domain governance, C_R is identity and reconciliation work, and C_X is physical and operational work. The direct cost is

$$C_{\text{direct}} = \sum_{(i,j)\in A} C(T_{ij}) + C_X^{\text{direct}}. \tag{4}$$

Only a controlled repair-surface study can compare (3) and (4) after source, consumer, and domain changes. No current experiment does so.

6.3 Replace

Mappings can replace duplicated semantic logic only inside a declared fragment. Examples include:

- repeated renaming and typed path navigation that factors through one adequate domain;
- shared unit or value-domain conversions with one authority and explicit loss behavior;
- query rewrites whose source coverage and scalar semantics are known;
- repeated provenance attachment to a versioned semantic artifact; and
- validation rules that the admitted mapping language can express and check.

Mediators, federated databases, GAV/LAV integration, data exchange, Clio, and ontology-based data access already establish much of this territory [Sheth and Larson, 1990, Wiederhold, 1992, Lenzerini, 2002, Fagin et al., 2009, Calvanese et al., 2017]. The CatDB question is whether one versioned contract can reduce correct repair work after its own governance and exception costs are counted.

6.4 Generate

A semantic mapping may generate a physical artifact without becoming that artifact. Possible outputs include SQL views, ETL or ELT transformations, federated subqueries, validation queries, compatibility layers, cache keys, materialization definitions, and provenance hooks. Clio already demonstrated mapping-driven generation of data-exchange transformations [Fagin et al., 2009]. ARKTOS treated ETL modeling and execution as an operational workflow [Vassiliadis et al., 2001]. Generation itself is not novel.

The mapping and its evidence remain the semantic authority when a generated job is rebuilt. The job is one physical realization. Federation and materialization can therefore change without silently redefining the integration contract.

6.5 Leave unchanged

Many tasks remain physical or source specific:

- extraction from proprietary APIs and unreliable files;

- cleaning text, addresses, units, and malformed values whose policy is not captured by the mapping language;
- bulk transfer, loading, scheduling, retries, checkpoints, and recovery;
- source-side triggers, external side effects, and operational cutover;
- human decisions about identity, policy, legal authority, and exceptions;
- pair-specific transformations that do not factor through the domain;
- schema changes with unavoidable information loss or downtime; and
- distributed coordination where operations do not commute.

Evidence status. The mapping-count proposition is Paper Reasoning. Reduced maintenance, fewer regressions, and lower integration cost remain Open. "CatDB replaces ETL" is Rejected.

7 Q4: Can semantic compilation remain competitive?

7.1 The intended boundary

The accepted physical-compilation paper places semantic work before native row processing [Long and Collaboration, 2026b]. A canonical request is validated and rewritten into a logical plan. Backend capabilities classify each operation as exact, residual, approximate, or unsupported. A physical lowerer then emits a parameterized native realization. The backend retains access-path selection, join ordering, and row execution.

That boundary follows established optimizer architecture. System R separated access-path selection from query semantics [Selinger et al., 1979]. Volcano provided extensible physical operators [Graefe, 1994]. Calcite supports query processing across heterogeneous sources [Begoli et al., 2018]. Semantic compilation and heterogeneous planning are not new by themselves.

Definition 7.1 (Compilation certificate)**.** A compilation certificate for plan p at semantic commit K records:

$$\chi(p, K) = (\text{query}, \text{mapping}, \text{capabilities}, \text{snapshot}, \text{residuals}, \text{normalization}, \text{provenance}).$$

It is reusable only when every declared dependency remains compatible and the selected backend operation retains the claimed exactness.

The certificate would make approximations and residual work explicit. It does not make the cost estimate correct or the generated query fast.

7.2 Federation

The accepted federation paper specifies mapping-aware read profiles, coverage, source authority, exact and repaired pushdown, join placement, partial availability, reconciliation, and typed result quality [Long and Collaboration, 2026e]. A required source failure must not become an empty complete answer. The concrete customer-order example in that paper is hand worked. There is no current source selector, planner, connector executor, result envelope, or heterogeneous differential run.

This distinction constrains the synthesis. A federation paper does not establish a federation engine. It also does not establish that semantic metadata improves pushdown, network traffic, or plan quality.

7.3 Materialization and incremental maintenance

The internally accepted materialization paper treats a materialized state as a realization of a versioned definition, not the semantic authority [Long and Collaboration, 2026f]. Classical materialized-view maintenance, differential dataflow, and DBSP already provide mature delta techniques [Gupta and Mumick, 1995, McSherry et al., 2013, Budiu et al., 2023]. CatDB adds a proposed invalidation question: does a schema, mapping, identity, provenance-policy, or capability change preserve the old meaning?

For a materialization V at source frontier f and semantic revision K, the proposed correctness gate is

$$N_K(V_{\text{incremental}}(K, f)) = N_K(V_{\text{recompute}}(K, f)) \,. \tag{5}$$

Equation (5) is not a result. The repository has no P6 evaluator, source change connector, recovery harness, or benchmark.

7.4 Cost-based choice

The accepted planning paper makes feasibility precede cost [Long and Collaboration, 2026n]. A plan that violates coverage, freshness, consistency, provenance, policy, or approximation requirements is inadmissible even if its predicted latency is low. Among feasible plans, a versioned objective may consider latency, network and storage traffic, local and remote work, memory, source load, freshness, coordination, provenance overhead, approximation quality, and failure behavior.

Traditional and adaptive optimization provide strong baselines [Leis et al., 2015, Avnur and Hellerstein, 2000, Marcus et al., 2021]. The current repository has no P14 candidate enumerator, calibrator, connector statistics contract, adaptive executor, or benchmark. There is no evidence for global optimality or scale.

7.5 Answer and promotion gate

Semantic compilation can be called competitive only if all of the following are observed on preregistered workloads:

1. normalized results and declared provenance match an independent oracle;

2. generated plans preserve required source semantics or declare a residual;
3. cold compilation and cached-plan overhead are measured separately;
4. native plan shape, bytes, calls, source load, latency, throughput, and memory are retained;
5. failures, timeouts, approximations, stale reads, and no-plan outcomes remain in the dataset; and
6. physical-only, conventional federated, and materialization baselines are strong enough to falsify the treatment.

Evidence status. OBSTRUCTED. No current query compiler, SQLite/PostgreSQL connector suite, federation planner, materialization engine, optimizer, native plan comparison, or scale benchmark establishes competitive execution.

8 Q5: Realistic consistency and convergence

8.1 One adjective is not enough

A local transaction can be serializable while a remote materialization is stale. An immutable provenance assertion may merge by set union while an active mapping reference requires one authorized value. Replaying a pure validation event may be harmless while replaying an external payment side effect is not. The accepted consistency paper therefore defines a profile per operation rather than one platform-wide adjective [Long and Collaboration, 2026k].

Definition 8.1 (Operation consistency profile)**.** For operation u, a consistency profile is

$$\kappa_u = (L, V, S, C, R, F, A, H),$$

where L is local isolation, V replica visibility, S session guarantees, C convergence, R replay behavior, F freshness, A validation and authority, and H historical retention.

Linearizability, CAP, CRDTs, monotone coordination analysis, and invariant-confluence are established foundations [Herlihy and Wing, 1990, Gilbert and Lynch, 2002, Shapiro et al., 2011, Ameloot et al., 2013, Bailis et al., 2014]. They apply under their own models. Categorical typing does not bypass them.

8.2 Operations that may converge without a single global order

The distributed paper model identifies candidates for available convergence under frozen admissibility rules [Long and Collaboration, 2026j]:

- addition of immutable content-addressed schema or mapping proposals;
- monotone accumulation of evidence whose authority does not imply activation;

- idempotent delivery of a pure validation result bound to the same inputs;
- explicit multi-value retention of concurrent alternatives; and
- join-based metadata components with proved or tested merge laws.

Even these cases need causal dependencies, stable identities, duplicate handling, and a frozen component contract. "Immutable" does not guarantee that a dependent artifact arrived first.

8.3 Operations that require coordination or explicit conflict

Single-valued activation is different from proposal creation. The following operations generally require coordination, escrow, serialized authority, or a retained conflict state:

- replacing the active schema or mapping for one workspace;
- confirming one canonical identity when competing decisions exist;
- publishing a branch head under noncommuting semantic changes;
- changing membership, authority, or retention policy;
- performing a migration cutover that cannot run twice safely;
- advancing a checkpoint past unobserved dependencies; and
- preserving uniqueness or other cross-object invariants.

Counterexample 8.2 (Delivery is not semantic convergence)**.** Site A activates mapping m_1. During a partition, site B activates mapping m_2 for the same single-valued reference. Both operations are eventually delivered everywhere. A set union converges to $\{m_1, m_2\}$, but the required state has one active mapping. The replicas have converged on an explicit conflict, not on a valid single-valued answer.

8.4 Answer

The realistic guarantee is an operation-by-operation matrix. Some immutable and admitted monotone operations may remain available and converge. Some updates can preserve concurrent alternatives as first-class conflicts. Noncommutative activation, authority, membership, retention, and migration transitions need coordination or cannot promise one valid value.
Evidence status. Accepted Paper Model for the P10/P11 specifications. Obstructed for runtime, convergence histories, recovery, availability, and performance. The repository has no distributed transport, replicated log, consensus-backed reference store, fault harness, or distributed benchmark.

9 Cross-cutting semantics

9.1 Identity

The internally accepted identity model uses at least four separate artifacts: source-local references, observed evidence, versioned method output, and an authority-bearing decision. Only an effective confirmed-same decision may enter a canonical equivalence projection. Possible, probable, contradicted, superseded, and retracted relationships remain visible.

This avoids a dangerous shortcut in the running example. The shared email between SQLite and PostgreSQL may be a strong feature. It may also be a reused family address, a recycled account, or a tenant-local identifier. Similarity can propose the pair. It cannot establish identity or write authority.

9.2 Provenance

Provenance must compose across the semantic and physical layers. A useful answer record may need:

$$\mathrm{Prov}(x) = (\text{sources}, \text{derivation}, \text{semantic revisions}, \text{execution}, \text{decisions}, \text{completeness}).$$

The completeness field matters. Retention may have removed source evidence. A connector may report only file-level origin. A transformation may preserve why-provenance but not exact value origin. A result should report that limit rather than present every provenance graph as complete.

9.3 Versioned workspaces

A semantic workspace is proposed as an isolated overlay over an immutable semantic commit. Diff has structural, operational, impact, validation, and realization layers. Merge is partial: it automatically combines only operation pairs admitted by a reviewed commutativity or noninterference rule, then validates the whole candidate state before publishing a branch reference. Dataset branching systems such as Decibel show established relational versioning capabilities [Maddox et al., 2016]. CatDB's candidate delta is cross-object versioning over semantic artifacts.

No current workspace crate, overlay evaluator, merge classifier, atomic reference store, or workspace fixture establishes this behavior.

9.4 Bidirectional updates

A readable canonical view is not automatically a safe write surface. Lenses and relational lenses provide precise prior foundations for well-behaved bidirectional transformations [Foster et al., 2007]. The accepted bidirectional-mapping paper proposes a write profile that classifies a mapping as fully writable, partially writable, read-only, or reconciliation-dependent relative to a declared update language [Long and Collaboration, 2026h].

Universal safe writeback is rejected. Loss, ambiguity, ownership, target constraints, stale source versions, multi-source atomicity, and external side effects can all force rejection or reconciliation.

10 Dependency graph and manuscript status

10.1 Why the graph matters

The accepted research manifest defines a 19-node acyclic blocking graph: papers P1 through P18, followed by this synthesis. Its exact projection has 70 edges, including one edge from every paper into the synthesis. The graph prevents an accepted model in one paper from quietly supplying absent evidence to a dependent paper.

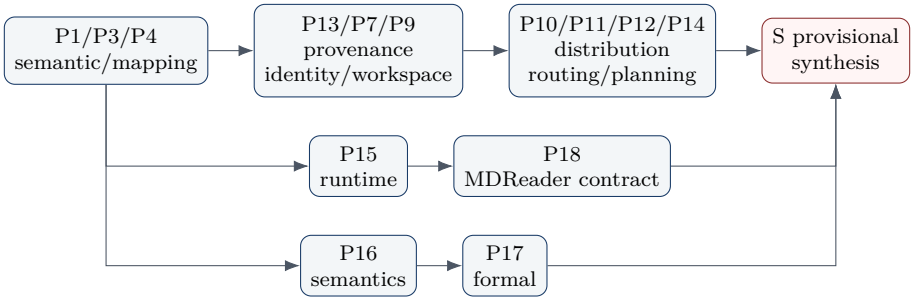

Figure 1: Three critical evidence chains, compressed from the exact 70-edge manifest graph. Blue boxes include internally accepted paper artifacts; the red box is the unreviewed synthesis.

The longest blocking path is

$$\mathrm{P1} \to \mathrm{P3} \to \mathrm{P4} \to \mathrm{P13} \to \mathrm{P7} \to \mathrm{P9} \to \mathrm{P10} \to \mathrm{P11} \to \mathrm{P12} \to \mathrm{P14} \to \mathrm{S}.$$

The runtime and application path is

$$\mathrm{P1} \to \mathrm{P3} \to \mathrm{P4} \to \mathrm{P13} \to \mathrm{P15} \to \mathrm{P18} \to \mathrm{S},$$

and the formal path is

$$\mathrm{P1} \to \mathrm{P3} \to \mathrm{P4} \to \mathrm{P16} \to \mathrm{P17} \to \mathrm{S}.$$

All companion-paper dependencies on the three displayed paths are now internally accepted. The formal paper P17 is accepted, but its machine-checked scope remains limited to the declarations reported below. The accepted P15 paper still describes a bounded research kernel, not a production runtime. The synthesis itself has not entered independent review and cannot be accepted.

10.2 Paper-by-paper status

Table 3: Current paper and implementation boundary.

Paper	Topic	Manuscript status	Strongest current evidence and exact limit
P1	Semantic model	Internally accepted	Paper proofs, bounded kernel computation, and selected Lean declarations; does not subsume all data models or prove a backend.
P2	Physical compilation	Internally accepted	Compilation contract only; no query compiler, connectors, native-plan capture, or performance result.
P3	Migration	Internally accepted	Paper proofs plus Slice 4 finite Δ prerequisite; no physical migration, cutover, rollback, or general loss analysis.
P4	Composable integration	Internally accepted	Conditional mapping-count result; no factorization checker, connector, identity resolver, or repair-surface experiment.
P5	Federation	Internally accepted	Federation contract only; no planner, executor, heterogeneous result comparison, or benchmark.
P6	Materialization	Internally accepted	Delta and invalidation contract only; no evaluator, recovery harness, freshness result, or benchmark.
P7	Identity	Internally accepted	Typed decision model only; no profile, matcher, state machine, fixtures, or experiment.
P8	Bidirectional mappings	Internally accepted	Write-profile model only; no safe-write runtime, proof, or database experiment.
P9	Workspaces	Internally accepted	Workspace and partial-merge model only; no overlay, merge, publication, or historical-query runtime.
P10	Distributed CatDB	Internally accepted	Typed operation algebra and paper classification; no transport, replicated log, resolver, fault history, or benchmark.
P11	Consistency	Internally accepted	Operation-level consistency model; no consistency-profile runtime, consensus store, recovery, or convergence evidence.
P12	Semantic sharding	Internally accepted	Placement, routing, pruning, join, exchange, index, and materialization contracts; no distributed planner or locality/performance result.

Paper	Topic	Manuscript status	Strongest current evidence and exact limit
P13	Provenance	Internally accepted	Profile-relative provenance algebra; no runtime, completeness checker, security result, exporter, theorem, or benchmark.
P14	Cost planning	Internally accepted	Feasibility and cost contract; no optimizer, calibration, adaptive executor, global-optimality result, or scale benchmark.
P15	Rust runtime	Internally accepted	Six-crate finite semantic kernel and accepted Slice 4 computation; no query, storage, connector, transaction, server, replay, or performance layer.
P16	Haskell semantics	Internally accepted	Its audited paper snapshot reports the accepted Slice 2 boundary; Slice 4 is a separate accepted release, and current Slice 5 work is excluded. No proof, backend, distributed, or performance authority follows.
P17	Lean formalization	Internally accepted	Selected path and mapping declarations are machine checked; the accepted paper does not verify instances, migration, queries, runtimes, backends, or distributed behavior.
P18	MDReader	Internally accepted	The accepted paper supplies an application contract, but no application artifact establishes projections, UI behavior, provenance, SQLite/PostgreSQL synchronization, or user value.

11 Current runtime and formal evidence

11.1 Slice 4 is the executable boundary

The repository contains a real semantic kernel, but it is small. The current Rust workspace has six crates for closed values, schemas, paths, mappings, finite instances, bounded migration, and fixture verification [Long and Collaboration, 2026o]. The Haskell package independently evaluates the same closed profile [Long and Collaboration, 2026p]. The accepted Slice 4 release record reports:

```
fixture cases=33
validation ok=12 error=19 unsupported=2
slice-4 semantic-compared=29 structural-compared=4
slice-4 disagreements=0
```

The 29 semantic cases cover schema parsing, path typing and composition, mapping identity and composition, finite-instance validation, two finite pullback migrations, negative migration cases, and exact unsupported Σ/Π outcomes. Information-loss and schema-diff cases remain structural only.

Slice 5 implementation work is present in the shared worktree, but it has no release record or accepted independent review. This synthesis excludes it until release. The accepted executable boundary therefore remains Slice 4, including the structural-only status of information-loss and schema-diff cases.

The accepted P16 and P18 manuscripts each record a 42-case author-audit snapshot in which a Slice 4 selection produced 29 semantic and 13 structural comparisons. P16 separately labels all 42 as semantic only under its unreviewed Slice 5 candidate. Those counts are snapshot-relative observations, not accepted Slice 5 evidence. The live Slice 5 worktree has continued to change, so this synthesis reports no live Slice 5 count.

Obligation 11.1 (Fixture boundary)**.** A statement derived from Slice 4 must name the capability set, corpus size, semantic count, structural count, disagreement count, repository revision, and known unfixtured divergences. It must not be generalized to a runtime, compiler, backend, distributed system, or theorem.

11.2 What Lean establishes

The accepted Lean Slice 3 checks exact path and mapping declarations in the `PATH-02` through `PATH-04` and `MAP-03` through `MAP-05` boundary [Long and Collaboration, 2026q]. Machine-checked database semantics and query rewrite proofs already have strong prior art, including HoTTSQL [Chu et al., 2017]. The current Lean result neither verifies Rust nor proves query, connector, migration-runtime, cost, distributed, or MDReader claims.

11.3 What does not exist

The following artifacts are absent from the current evidence cut:

- a query IR implementation, compiler, physical planner, or plan cache;

- SQLite, PostgreSQL, JSON, Markdown, warehouse, vector, or full-text connector execution;
- a federation engine, distributed join executor, exchange layer, or semantic pruning oracle;
- a materialization or incremental-maintenance engine;
- identity, reconciliation, provenance, workspace, or policy runtimes;
- a replicated operation log, causal transport, consensus-backed reference store, or fault harness;
- a calibrated optimizer or adaptive execution loop;
- an MDReader application;
- a complete benchmark, cross-shard measurement, scale result, or production readiness assessment.

The presence of empty or scaffolded directories would not establish these capabilities. This synthesis uses observed behavior only where a current command and closed artifact support it.

12 Evaluation contract and falsifiers

12.1 No results are reported here

The evaluation program below is a promotion contract. It is not a benchmark section disguised in future tense. No numerical performance, repair, federation, materialization, provenance, distributed, or MDReader result is reported.

12.2 Falsifiers

The synthesis thesis must be rejected or narrowed if any of the following is observed:

1. a required source or consumer meaning cannot be represented without hidden pair-specific code that the mapping contract reports as exact;
2. a supposedly exact composition has a counterexample under the admitted equality, coverage, loss, or authority profile;
3. a virtual and materialized realization under the same declared snapshot and scalar profile return different normalized results;
4. the mediated treatment does not reduce correct-repair work after domain governance, reconciliation, exceptions, failures, and operations are counted;

Table 4: Required evaluation groups and present status.

Group	Minimum comparison	Status
Semantic compilation	Native query versus compiled query with normalized bags, scalar semantics, provenance, plan capture, and cold/cache separation	Obstructed
Federation	Native per-source oracle, R*-style or Calcite baseline, bytes, calls, partial availability, and exactness labels	Obstructed
Materialization	Full recomputation versus incremental treatment across data and semantic changes, with crash and cursor faults	Obstructed
Integration repair	Pairwise, mediated, GAV, LAV, and generated-plan conditions after source, consumer, and domain changes	Obstructed
Provenance	Derivation correctness, completeness class, query cost, storage cost, privacy, and retention failure	Obstructed
Distribution	Partition, reorder, duplicate, replay, healing, checkpoint, authority, and invariant histories	Obstructed
Routing and planning	Physical-only and communication-aware baselines, semantic ablations, prediction error, cross-shard bytes, and failure behavior	Obstructed
MDReader	End-to-end typed projections, violations, provenance, history, responsive UI, and task outcomes	Obstructed

5. compilation overhead or generated-plan quality is unacceptable against strong native and conventional-planner baselines;
6. a required unavailable source is reported as an empty complete federated answer;
7. incremental state differs from full recomputation at the same semantic revision and source frontier;
8. provenance omits a declared contributing source, mapping, identity decision, or execution revision while claiming completeness;
9. a replay, reorder, duplicate, or partition history violates the guarantee named by its operation profile;
10. semantic routing prunes a shard containing a qualifying occurrence, changes bag multiplicity, or bypasses policy;
11. current Rust and Haskell programs disagree on an admitted fixture or replay an expected value instead of computing it;
12. a claimed Lean theorem contains an unreported axiom, assumption, or scope mismatch;
13. the MDReader application cannot expose the mapping, provenance, violation, or history that motivates it; or
14. measured semantic-metadata maintenance cost exceeds its benefit on the intended workload.

12.3 Staged adoption

A conservative adoption path starts with artifacts that can be inspected without replacing an existing database:

1. validate and version finite schemas, paths, and mappings;
2. compare generated validation or read plans with existing hand-written behavior;
3. attach mapping and source-version provenance to selected reports;
4. add one read-only federated or materialized path behind a differential oracle;
5. measure repair work after controlled source and consumer changes;
6. add writeback only for a declared update language with authority and transaction checks; and
7. attempt distributed replication only after operation profiles and fault histories close.

This order permits CatDB to fail cheaply. A team can reject the semantic layer if it cannot represent its integration decisions or if the maintenance cost is not justified.

13 Related work and candidate novelty

13.1 Established foundations

The synthesis sits at the intersection of mature areas. Categories, functors, data migration, algebraic databases, and categorical integration are established [Spivak, 2012, Spivak and Wisnesky, 2015, Schultz and Wisnesky, 2017, Schultz et al., 2017]. Model management, schema mapping, data exchange, mediation, federation, and ontology-based access already treat integration knowledge as explicit structure [Bernstein and Melnik, 2007, Fagin et al., 2005a,b, Sheth and Larson, 1990, Wiederhold, 1992, Lenzerini, 2002, Calvanese et al., 2017]. Lenses address bidirectional updates [Foster et al., 2007]. Materialized-view maintenance and differential systems address incremental computation [Gupta and Mumick, 1995, McSherry et al., 2013, Budiu et al., 2023]. Distributed consistency and convergence have their own formal limits [Gilbert and Lynch, 2002, Shapiro et al., 2011, Bailis et al., 2014].

The paper-local related-work and primary-source ledgers give the exact source-to-claim map. No companion manuscript or external source is used as evidence that the unimplemented CatDB components work.

13.2 Narrow candidate delta

The research claim is a systems synthesis:

> One versioned artifact graph could connect typed schema mappings, migration intent, identity authority, provenance, workspace history, and physical realizations, while independent computation and selected theorem checking make each promotion boundary visible.

That combination may be useful. It is not yet a demonstrated systems result. Novelty would require both a completed artifact and direct comparison with categorical integration, model management, mediated data integration, ontology-based access, conventional planners, provenance systems, dataset versioning, and distributed databases. A vocabulary that simply places known ideas beside each other is not enough.

14 Limitations and open questions

14.1 Limitations

1. The current semantic profile is finite, total, and narrow. It omits nulls, optional attributes, partial functions, general constraints, richer typesides, bags in the kernel, and general path equality.

2. Multimodel preservation is proposed, not implemented. Native relational, graph, RDF/OWL, document, vector, and full-text semantics may require residual engines.

3. Typed mappings do not establish intended business meaning. Human governance and source authority remain necessary.

4. Composition can be partial, ambiguous, lossy, unsupported, or unknown.
5. The conditional $|I|+|J|$ mapping count does not imply lower repair time, fewer incidents, or lower total cost.
6. Generated plans can be semantically wrong or physically slow.
7. Federation inherits source capability, availability, privacy, snapshot, and network limits.
8. Materializations become stale and require dependency, checkpoint, and recovery semantics.
9. Identity decisions remain uncertain and may require reversal.
10. Provenance can be incomplete, expensive, sensitive, or removed by retention.
11. Workspace merge is partial and cannot automatically resolve every semantic conflict.
12. CAP, PACELC, consensus, and noncommutativity remain physical limits.
13. Slice 4 fixtures are a bounded acceptance surface, not proof, performance evidence, or production readiness.
14. Selected Lean declarations do not verify the runtime.
15. MDReader is an accepted reference-application contract, not product evidence.

14.2 Open questions

1. Which mapping language is expressive enough for real integration while retaining decidable and explainable validation?
2. Can loss, ambiguity, and coverage analyses remain conservative without returning unknown too often?
3. What semantic domain granularity avoids both pairwise duplication and an ungovernable enterprise-wide model?
4. Which provenance granularity is sufficient for debugging, compliance, and replay at acceptable cost?
5. Can compiled plans preserve native optimizer quality while the semantic layer stays outside row execution?
6. Which schema and mapping changes admit safe incremental adaptation rather than rebuild?
7. How should uncertain identity affect query completeness, materialization, and writeback?
8. Which operation components are monotone or invariant-confluent under a frozen semantic profile?

9. Can semantic partition metadata improve routing after its collection, validation, and maintenance cost is counted?

10. Does MDReader provide a sufficiently demanding vertical slice, or does it hide the hardest transactional and scale requirements?

15 Conclusion

CatDB is best understood as a proposal about where integration meaning lives. Instead of treating a schema mapping, migration choice, identity decision, and provenance rule as unrelated configuration around a pipeline, the proposal gives each one a typed identity, version, interpretation, and evidence boundary. Federation, ETL, and materialization then become physical realizations of that contract.

The current evidence supports only part of this picture. Internally accepted papers provide careful models for the semantic core, compilation, migration, composable integration, federation, materialization, identity, bidirectional updates, workspaces, distribution, consistency, provenance, planning, and formalization, together with sharding, the bounded Rust kernel, Haskell semantics, and the MDReader application contract. The accepted Slice 4 record shows exact Rust/Haskell agreement on 29 semantic cases inside a closed 33-fixture profile, with four structural-only cases. Selected Lean declarations check named path and mapping laws.

No current evidence establishes a distributed runtime, query compiler, connector suite, federation engine, materialization engine, optimizer, MDReader application, or benchmark. The synthesis therefore cannot claim that CatDB replaces ETL, removes migration cost, preserves native performance, enables universal writeback, or escapes distributed-systems tradeoffs.

The narrower thesis survives: typed, composable, versioned mappings may move some integration meaning out of bespoke pipelines and into inspectable objects. Whether that shift improves repair work while preserving competitive execution remains the central experiment.

A Synthesis claim ledger

Table 5: Claim anchors and promotion gates.

ID	Claim	Current status	Required promotion evidence
S-C01	CatDB is defined as a semantic multimodel platform whose schemas, mappings, migrations, identities, and provenance are first-class, versioned, executable objects.	Paper Reasoning definition	Accepted cross-object IR and validators
S-C02	The finite semantic kernel represents and validates the exact Slice 4 profile.	Computational	Current commands, closed manifest, and reviewed source boundary; unreleased Slice 5 work is excluded
S-C03	Rust and independent Haskell implementations agree on 29 semantic cases in the accepted 33-fixture Slice 4 corpus.	Computational	Recorded revision, exact reports, and zero disagreements
S-C04	Typed migration artifacts can make direction, coverage, loss, ambiguity, authority, dependencies, and execution intent reviewable.	Engineering hypothesis; Open	Implemented artifact, negative fixtures, and user/reviewer study
S-C05	Under complete factorization assumptions, a full $\|I\| \times \|J\|$ topology can use $\|I\| + \|J\|$ base mappings.	Paper Reasoning conditional count	Assumptions retained; no maintenance promotion inferred
S-C06	Mediated mappings reduce total correct-repair work.	Engineering hypothesis; Obstructed	Preregistered repair-surface experiment with full cost ledger
S-C07	Federation and materialization can realize one mapping contract without changing declared meaning.	Engineering hypothesis; Obstructed	Two implementations and normalized equivalence under one profile
S-C08	Semantic compilation preserves competitive native execution.	Engineering hypothesis; Obstructed	Compiler, connectors, native plans, correctness oracle, and benchmark
S-C09	Semantic metadata improves federation, pruning, routing, or plan selection.	Engineering hypothesis; Obstructed	Physical-only baselines, ablations, overhead, and cross-shard measurements
S-C10	Incremental maintenance preserves full recomputation across data and semantic changes.	Engineering hypothesis; Obstructed	P6 evaluator, fault harness, independent oracle, and recovery evidence
S-C11	Identity evidence can be preserved without silently collapsing uncertainty.	Engineering hypothesis; Obstructed	Identity profile, state machine, fixtures, reversal, and authority tests
S-C12	Provenance can compose source, derivation, semantic-version, execution, and decision evidence with an explicit completeness class.	Accepted Paper Model; runtime Obstructed	Runtime, completeness tests, retention/privacy gates, and overhead
S-C13	Some immutable or monotone operations may converge without a single global order, while noncommutative activation requires conflict or coordination.	Accepted Paper Model	Executable histories and invariant-specific proof or computation
S-C14	A versioned workspace can support safe semantic diff and partial merge across related artifacts.	Accepted Paper Model; runtime Obstructed	Workspace state machine, overlay oracle, merge fixtures, and atomic publish
S-C15	MDReader demonstrates end-to-end CatDB value.	Rejected at current cut	Implemented application, scenarios, UI checks, and user evidence
S-C16	CatDB eliminates ETL, migration cost, CAP tradeoffs, or the need for human authority.	Rejected	No positive promotion path; claim contradicts scope
S-C17	The overall systems thesis is established.	Obstructed	Accepted P1–P18, current implementation evidence, independent review, and preregistered experiments

B Exact disputed statements

Table 6: Statements rejected by the synthesis.

Rejected statement	Required correction
CatDB replaces ETL.	Some reusable semantic logic may move into mappings; physical ETL/ELT, cleaning, transport, loading, scheduling, and operations remain.
Integration drops from NM to $N + M$.	The count is conditional on a full topology, exact factorization, closure, domain adequacy, and no residual pair exceptions.
Fewer mappings mean less maintenance.	Mapping count is not correct-repair time; include governance, reconciliation, exceptions, failures, and operations.
Typed mappings solve schema matching.	Matching proposes correspondences; typed acceptance still needs validation and authority.
Vector similarity proves identity.	Similarity may rank candidates; an authority-bearing decision establishes a canonical relation.
A type-correct mapping is business-correct.	Business meaning, policy, coverage, freshness, and authority require separate evidence.
Virtual and materialized results are automatically equivalent.	Compare them under the same frozen snapshot, scalar, bag, identity, and provenance profile.
Semantic compilation preserves native performance.	No compiler, connector, native-plan comparison, or benchmark currently exists.
The accepted federation paper proves federation works.	It establishes a reviewed contract, not a planner, executor, or experiment.
The accepted distributed papers prove convergence.	They classify operations on paper; no distributed runtime or fault history exists.
Category theory removes CAP or consensus.	Typing can expose obligations; it cannot deliver messages or make noncommutative updates commute.
Writeback is safe whenever a read mapping exists.	Writeback needs a declared update language, authority, constraints, snapshot checks, and transaction policy.
Rust/Haskell agreement proves correctness.	It establishes exact agreement on the admitted fixtures, subject to common-mode and coverage limits.
Lean verifies the runtime.	Lean checks exact named declarations under audited assumptions, not Rust, backends, networks, or the complete engine.
MDReader validates the platform.	An accepted paper supplies a reference application contract, but no MDReader application artifact exists.

Rejected statement	Required correction
Unreleased Slice 5 work extends the accepted kernel.	Slice 5 has no release record or accepted independent review. The synthesis retains Slice 4 as its executable boundary.

C Review and reproduction gate

A future acceptance candidate must satisfy all of the following:

1. every companion paper P1 through P18 is accepted at a current source and evidence revision;
2. every synthesis claim links to a paper result, primary source, exact computation, theorem, experiment, or explicit obstruction;
3. the Slice 4 counts and known unfixtured limits remain current;
4. any Slice 5 claim is excluded unless a release record and accepted independent review establish its exact boundary;
5. any Lean statement names its exact declaration and assumptions;
6. compilation, federation, materialization, distribution, planning, and MDReader claims use real current artifacts rather than planned interfaces;
7. all external claims close against the paper-local primary-source and related-work ledgers;
8. the disputed-claim ledger has no unresolved trigger;
9. Humanizer touches prose only and does not alter definitions, equations, evidence labels, tables, citations, or falsifiers;
10. a clean isolated build closes references and citations with no warnings or overfull content;
11. every rendered page is inspected;
12. a separately authorized independent paper review reaches a publishable verdict; and
13. any later source or relied-on evidence change reopens the candidate.

This authoring pass does not invoke the independent review gate and does not create an accepted PDF or cover.

References

Sattam Alsubaiee, Yasser Altowim, Hotham Altwaijry, Alexander Behm, Vinayak Borkar, Yingyi Bu, Michael Carey, Inci Cetindil, Madhusudan Cheelangi, Khurram Faraaz, Eugenia Gabrielova, Raman Grover, Zachary Heilbron, Young-Seok Kim, Chen Li, Guangqiang Li, Ji Mahn Ok, Nicola Onose, Pouria Pirzadeh, Vassilis Tsotras, Rares Vernica, Jian Wen, and Till Westmann. Asterixdb: A scalable, open source bdms. *Proceedings of the VLDB Endowment*, 7(14):1905–1916, 2014. doi: 10.14778/2733085.2733096. URL https://www.vldb.org/pvldb/vol7/p1905-alsubaiee.pdf.

Tom J. Ameloot, Frank Neven, and Jan Van den Bussche. Relational transducers for declarative networking. *Journal of the ACM*, 60(2):15:1–15:38, 2013. doi: 10.1145/2450142.2450151. URL https://doi.org/10.1145/2450142.2450151.

Ron Avnur and Joseph M. Hellerstein. Eddies: Continuously adaptive query processing. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, pages 261–272, 2000. doi: 10.1145/335191.335420. URL https://doi.org/10.1145/335191.335420.

Peter Bailis, Alan Fekete, Michael J. Franklin, Ali Ghodsi, Joseph M. Hellerstein, and Ion Stoica. Coordination avoidance in database systems. *Proceedings of the VLDB Endowment*, 8(3):185–196, 2014. doi: 10.14778/2735508.2735509. URL https://doi.org/10.14778/2735508.2735509.

Edmon Begoli, Jesús Camacho-Rodríguez, Julian Hyde, Michael J. Mior, and Daniel Lemire. Apache calcite: A foundational framework for optimized query processing over heterogeneous data sources. In *Proceedings of the 2018 International Conference on Management of Data*, pages 221–230, 2018. doi: 10.1145/3183713.3190662. URL https://doi.org/10.1145/3183713.3190662.

Omar Benjelloun, Hector Garcia-Molina, David Menestrina, Qi Su, Steven Euijong Whang, and Jennifer Widom. Swoosh: A generic approach to entity resolution. *The VLDB Journal*, 18(1):255–276, 2009. doi: 10.1007/s00778-008-0098-x. URL https://doi.org/10.1007/s00778-008-0098-x.

Philip A. Bernstein and Sergey Melnik. Model management 2.0: Manipulating richer mappings. In *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*, pages 1–12, 2007. doi: 10.1145/1247480.1247482. URL https://doi.org/10.1145/1247480.1247482.

Mihai Budiu, Tej Chajed, Frank McSherry, Leonid Ryzhyk, and Val Tannen. DBSP: Automatic incremental view maintenance for rich query languages. *Proceedings of the VLDB Endowment*, 16(7):1601–1614, 2023. doi: 10.14778/3587136.3587137. URL https://www.vldb.org/pvldb/vol16/p1601-budiu.pdf.

Peter Buneman, Sanjeev Khanna, and Wang-Chiew Tan. Why and where: A characterization of data provenance. In *Database Theory — ICDT 2001*, volume 1973 of *Lecture Notes in*

Computer Science, pages 316–330. Springer, 2001. doi: 10.1007/3-540-44503-X_20. URL https://doi.org/10.1007/3-540-44503-X_20.

Diego Calvanese, Benjamin Cogrel, Sarah Komla-Ebri, Roman Kontchakov, Davide Lanti, Martin Rezk, Mariano Rodriguez-Muro, and Guohui Xiao. Ontop: Answering sparql queries over relational databases. *Semantic Web*, 8(3):471–487, 2017. doi: 10.3233/SW-160217. URL https://doi.org/10.3233/SW-160217.

CategoricalData project. Categorical query language (CQL) documentation and tutorial. Official project documentation, 2026. URL https://categoricaldata.net/help/Tutorial.html. Accessed 2026-07-22.

Peter Pin-Shan Chen. The entity-relationship model—toward a unified view of data. *ACM Transactions on Database Systems*, 1(1):9–36, 1976. doi: 10.1145/320434.320440. URL https://doi.org/10.1145/320434.320440.

Shumo Chu, Konstantin Weitz, Alvin Cheung, and Dan Suciu. Hottsql: Proving query rewrites with univalent sql semantics. In *Proceedings of the 38th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 510–524, 2017. doi: 10.1145/3062341.3062348. URL https://arxiv.org/abs/1607.04822.

E. F. Codd. A relational model of data for large shared data banks. *Communications of the ACM*, 13(6):377–387, 1970. doi: 10.1145/362384.362685. URL https://doi.org/10.1145/362384.362685.

Ronald Fagin, Phokion G. Kolaitis, Renée J. Miller, and Lucian Popa. Data exchange: Semantics and query answering. *Theoretical Computer Science*, 336(1):89–124, 2005a. doi: 10.1016/j.tcs.2004.10.033. URL https://doi.org/10.1016/j.tcs.2004.10.033.

Ronald Fagin, Phokion G. Kolaitis, Lucian Popa, and Wang-Chiew Tan. Composing schema mappings: Second-order dependencies to the rescue. *ACM Transactions on Database Systems*, 30(4):994–1055, 2005b. doi: 10.1145/1114244.1114249. URL https://doi.org/10.1145/1114244.1114249.

Ronald Fagin, Laura M. Haas, Mauricio A. Hernández, Renée J. Miller, Lucian Popa, and Yannis Velegrakis. Clio: Schema mapping creation and data exchange. In *Conceptual Modeling: Foundations and Applications*, volume 5600 of *Lecture Notes in Computer Science*, pages 198–236. Springer, 2009. doi: 10.1007/978-3-642-02463-4_12. URL https://doi.org/10.1007/978-3-642-02463-4_12.

Ivan P. Fellegi and Alan B. Sunter. A theory for record linkage. *Journal of the American Statistical Association*, 64(328):1183–1210, 1969. doi: 10.1080/01621459.1969.10501049. URL https://doi.org/10.1080/01621459.1969.10501049.

J. Nathan Foster, Michael B. Greenwald, Jonathan T. Moore, Benjamin C. Pierce, and Alan Schmitt. Combinators for bidirectional tree transformations: A linguistic approach to the view-update problem. *ACM Transactions on Programming Languages and Systems*, 29(3), 2007. doi: 10.1145/1232420.1232424. URL https://doi.org/10.1145/1232420.1232424.

Seth Gilbert and Nancy Lynch. Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. *ACM SIGACT News*, 33(2):51–59, 2002. doi: 10.1145/564585.564601. URL https://doi.org/10.1145/564585.564601.

Goetz Graefe. Volcano—an extensible and parallel query evaluation system. *IEEE Transactions on Knowledge and Data Engineering*, 6(1):120–135, 1994. doi: 10.1109/69.273032. URL https://doi.org/10.1109/69.273032.

Todd J. Green, Grigoris Karvounarakis, and Val Tannen. Provenance semirings. In *Proceedings of the 26th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 31–40, 2007. doi: 10.1145/1265530.1265535. URL https://doi.org/10.1145/1265530.1265535.

Ashish Gupta and Inderpal Singh Mumick. Maintenance of materialized views: Problems, techniques, and applications. *IEEE Data Engineering Bulletin*, 18(2):3–18, 1995. URL https://www.vldb.org/dblp/db/journals/debu/GuptaM95.html.

Maurice P. Herlihy and Jeannette M. Wing. Linearizability: A correctness condition for concurrent objects. *ACM Transactions on Programming Languages and Systems*, 12(3):463–492, 1990. doi: 10.1145/78969.78972. URL https://doi.org/10.1145/78969.78972.

ISO/IEC JTC 1/SC 32. Information technology—database languages—gql. Technical Report ISO/IEC 39075:2024, International Organization for Standardization, 2024. URL https://www.iso.org/standard/76120.html.

Timothy Lebo, Satya Sahoo, Deborah McGuinness, et al. PROV-O: The PROV ontology. W3c recommendation, World Wide Web Consortium, 2013. URL https://www.w3.org/TR/prov-o/.

Viktor Leis, Andrey Gubichev, Atanas Mirchev, Peter Boncz, Alfons Kemper, and Thomas Neumann. How good are query optimizers, really? *Proceedings of the VLDB Endowment*, 9(3):204–215, 2015. doi: 10.14778/2850583.2850594. URL https://www.vldb.org/pvldb/vol9/p204-leis.pdf.

Maurizio Lenzerini. Data integration: A theoretical perspective. In *Proceedings of the 21st ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 233–246, 2002. doi: 10.1145/543613.543644. URL https://doi.org/10.1145/543613.543644.

Matthew Long and The YonedaAI Collaboration. Catdb: A categorical semantic model for executable database schemas. Internally accepted CatDB paper artifact; bounded model and evidence boundary, 2026a.

Matthew Long and The YonedaAI Collaboration. From categorical schemas to physical plans. Internally accepted CatDB paper artifact; compiler contract, no runtime result, 2026b.

Matthew Long and The YonedaAI Collaboration. Schema migration as a typed mapping. Internally accepted CatDB paper artifact; migration model, no physical migration runtime, 2026c.

Matthew Long and The YonedaAI Collaboration. Composable schema mappings for data integration. Internally accepted CatDB paper artifact; conditional factorization account, no maintenance result, 2026d.

Matthew Long and The YonedaAI Collaboration. Semantic query federation over heterogeneous databases. Internally accepted CatDB paper artifact; federation contract, no runtime result, 2026e.

Matthew Long and The YonedaAI Collaboration. Incremental semantic materialization. Internally accepted CatDB paper artifact; materialization contract, no evaluator or benchmark, 2026f.

Matthew Long and The YonedaAI Collaboration. Identity, equivalence, and reconciliation. Internally accepted CatDB paper artifact; identity decision model, no resolver or experiment, 2026g.

Matthew Long and The YonedaAI Collaboration. Bidirectional schema mappings and safe update propagation in catdb. Internally accepted CatDB paper artifact; write-profile model, no runtime result, 2026h.

Matthew Long and The YonedaAI Collaboration. Versioned semantic workspaces. Internally accepted CatDB paper artifact; workspace model, no runtime result, 2026i.

Matthew Long and The YonedaAI Collaboration. Distributed catdb: Typed operation algebra and coordination boundaries. Internally accepted CatDB paper artifact; distributed model, no runtime result, 2026j.

Matthew Long and The YonedaAI Collaboration. Consistency models for distributed semantic databases. Internally accepted CatDB paper artifact; consistency model, no runtime result, 2026k.

Matthew Long and The YonedaAI Collaboration. Semantic sharding and distributed query planning. Internally accepted CatDB paper artifact; no distributed planner, locality, or performance result, 2026l.

Matthew Long and The YonedaAI Collaboration. Provenance as first-class execution semantics. Internally accepted CatDB paper artifact; provenance model, no runtime result, 2026m.

Matthew Long and The YonedaAI Collaboration. Cost-based planning for semantic integration. Internally accepted CatDB paper artifact; planning contract, no optimizer result, 2026n.

Matthew Long and The YonedaAI Collaboration. A rust runtime for categorical data systems. Internally accepted CatDB paper artifact; bounded semantic kernel only, no production-runtime result, 2026o.

Matthew Long and The YonedaAI Collaboration. A haskell reference semantics for catdb schema mappings. Internally accepted CatDB paper artifact; audited Slice 2 snapshot and bounded reference semantics only, 2026p.

Matthew Long and The YonedaAI Collaboration. Mechanizing catdb: A lean kernel for typed paths and equation-preserving schema mappings. Internally accepted CatDB paper artifact; selected path and mapping declarations only, 2026q.

Matthew Long and The YonedaAI Collaboration. Mdreader as a reference application for categorical schema modeling: Typed markdown projections, inspectable mappings, and evidence boundaries. Internally accepted CatDB paper artifact; application contract, no application result, 2026r.

Michael Maddox, David Goehring, Aaron J. Elmore, Samuel Madden, Aditya Parameswaran, and Amol Deshpande. Decibel: The relational dataset branching system. *Proceedings of the VLDB Endowment*, 9(9):624–635, 2016. doi: 10.14778/2947618.2947619. URL https://www.vldb.org/pvldb/vol9/p624-maddox.pdf.

Ryan Marcus, Parimarjan Negi, Hongzi Mao, Chi Zhang, Mohammad Alizadeh, Tim Kraska, Olga Papaemmanouil, and Nesime Tatbul. Bao: Making learned query optimization practical. In *Proceedings of the 2021 International Conference on Management of Data*, pages 1275–1288, 2021. doi: 10.1145/3448016.3452838. URL https://doi.org/10.1145/3448016.3452838.

Frank McSherry, Derek Murray, Rebecca Isaacs, and Michael Isard. Differential dataflow. In *Conference on Innovative Data Systems Research*, 2013. URL https://www.cidrdb.org/cidr2013/Papers/CIDR13_Paper111.pdf.

Patrick Schultz and Ryan Wisnesky. Algebraic data integration. *Journal of Functional Programming*, 27:e24, 2017. doi: 10.1017/S0956796817000168. URL https://doi.org/10.1017/S0956796817000168.

Patrick Schultz, David I. Spivak, Christina Vasilakopoulou, and Ryan Wisnesky. Algebraic databases. *Theory and Applications of Categories*, 32(16):547–619, 2017. doi: 10.70930/tac/lf9k6awo. URL https://arxiv.org/abs/1602.03501.

P. Griffiths Selinger, Morton M. Astrahan, Donald D. Chamberlin, Raymond A. Lorie, and Thomas G. Price. Access path selection in a relational database management system. In *Proceedings of the 1979 ACM SIGMOD International Conference on Management of Data*, pages 23–34, 1979. doi: 10.1145/582095.582099. URL https://doi.org/10.1145/582095.582099.

Marc Shapiro, Nuno Preguiça, Carlos Baquero, and Marek Zawirski. Conflict-free replicated data types. In *Stabilization, Safety, and Security of Distributed Systems*, volume 6976 of *Lecture Notes in Computer Science*, pages 386–400. Springer, 2011. doi: 10.1007/978-3-642-24550-3_29. URL https://doi.org/10.1007/978-3-642-24550-3_29.

Amit P. Sheth and James A. Larson. Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Computing Surveys*, 22(3):183–236, 1990. doi: 10.1145/96602.96604. URL https://doi.org/10.1145/96602.96604.

David I. Spivak. Functorial data migration. *Information and Computation*, 217:31–51, 2012. doi: 10.1016/j.ic.2012.05.001. URL https://arxiv.org/abs/1009.1166.

David I. Spivak and Robert E. Kent. Ologs: A categorical framework for knowledge representation. *PLOS ONE*, 7(1):e24274, 2012. doi: 10.1371/journal.pone.0024274. URL https://doi.org/10.1371/journal.pone.0024274.

David I. Spivak and Ryan Wisnesky. Relational foundations for functorial data migration. In *Proceedings of the 15th Symposium on Database Programming Languages*, pages 21–28, 2015. doi: 10.1145/2815072.2815075. URL https://arxiv.org/abs/1212.5303.

Panos Vassiliadis, Zografoula Vagena, Spiros Skiadopoulos, Nikos Karayannidis, and Timos Sellis. Arktos: Towards the modeling, design, control and execution of etl processes. *Information Systems*, 26(8):537–561, 2001. doi: 10.1016/S0306-4379(01)00039-4. URL https://doi.org/10.1016/S0306-4379(01)00039-4.

W3C OWL Working Group. Owl 2 web ontology language document overview (second edition). W3c recommendation, World Wide Web Consortium, 2012. URL https://www.w3.org/TR/owl2-overview/.

W3C RDF Working Group. Rdf 1.1 concepts and abstract syntax. W3c recommendation, World Wide Web Consortium, 2014. URL https://www.w3.org/TR/rdf11-concepts/.

Gio Wiederhold. Mediators in the architecture of future information systems. *Computer*, 25 (3):38–49, 1992. doi: 10.1109/2.121508. URL https://doi.org/10.1109/2.121508.