PhoenixAI

Primary Champion: Data Engineer · Category: Real-Time Analytical Database (Real-Time OLAP) · Closest Competitor: ClickHouse · Generated: October 25, 2026

First-draft copy — a hypothesis to test with real buyers, and a foundation to write from, not finished copy.

Built from — Positioning: Positioning Document Against ClickHouse for Data Engineers in any industry.pdf

Note: the stage 9 quality check still had notes after three passes. Read this closely before using it.

STAGE 7

Messaging Hierarchy

Value proposition

PhoenixAI lets you query live Kafka and Iceberg tables together without denormalizing schemas, waiting for merges, or managing 49 clusters.

Positioning statement (internal only — never ship as copy)

For data engineers who need to serve AI agents, customer-facing analytics, and BI dashboards with sub-second queries on fresh streaming and lakehouse data at scale, PhoenixAI is a real-time analytical database that runs distributed joins on normalized streaming and lakehouse data without the FINAL tax or cluster sprawl. Unlike ClickHouse, you query live Kafka and Iceberg tables together without denormalizing schemas, waiting for background merges, or managing separate clusters.

Pillar 1 — Query streaming and lakehouse data together without denormalizing your schema

Answers: ClickHouse forces you to choose between slow distributed joins that crash on complex queries or denormalizing into wide tables that waste 10× the storage, require six-month backfill projects whenever dimension data changes, and break real-time CDC pipelines—or you spin up 49 separate clusters just to isolate workloads.

Capabilities

- Run joins across normalized Kafka feeds and Iceberg tables with sub-second latency on billions of rows—ship agent features and dashboards without flattening your schema first
- Consolidate agent, BI, and ETL workloads on the same data with independent compute per warehouse—scale agent compute to 10 nodes while BI runs on 2, same tables
- Avoid the six-month schema redesign and backfill cycle every time a new dimension or join requirement arrives

Proof: Demandbase consolidated 49 ClickHouse clusters to 1 PhoenixAI warehouse, cut hardware 60% and storage 90%, and shipped agent features in under 2 months without denormalizing.

Pillar 2 — Serve fresh CDC data in seconds without killing query performance

Answers: ClickHouse ReplacingMergeTree forces a choice between stale duplicate rows in query results until background merges complete (minutes to hours) or using SELECT FINAL which is 5-20× slower and single-threaded on large tables—making sub-second freshness impossible without destroying latency.

Capabilities

- Commit Kafka and Flink CDC upserts queryable in under 5 seconds at 100K+ events/sec—agents and dashboards see the latest version without waiting for merges
- Query mutable streaming data with sub-second latency at 10K+ concurrent queries—no FINAL keyword, no stale duplicates, no performance cliff
- Ship fraud detection, delivery ops, and agent features that require seconds-level freshness without rewriting your CDC pipelines or adding a speed layer

Proof: Coinbase ingests 30K messages/sec across 573B+ rows and 300+ tables with <1s query latency; Pinterest hit 50% lower p90 latency vs. Druid with 10-second data freshness.

Pillar 3 — Run unpredictable agent SQL on production data with cost-based optimization

Answers: LLM-generated SQL with novel join patterns breaks or times out on ClickHouse because its rule-based planner can't reorder joins at runtime, apply global filters, or adapt to query patterns it hasn't seen—forcing you to manually tune every new agent query or restrict agents to pre-denormalized views.

Capabilities

- Handle agent-generated joins and filters automatically with cost-based optimizer that reorders joins, pushes down runtime filters, and uses query feedback—no manual tuning per query
- Run agent workloads on the same fresh normalized data as BI dashboards without separate denormalized copies or clusters
- Ship agent features in weeks instead of waiting quarters for schema rewrites and performance testing

Proof: Conductor runs agent queries scanning 6-7 GB / 180-240M rows in 389ms-2.8s with 100% decimal accuracy, shipped in <2 months; Demandbase replaced a ClickHouse architecture that couldn't handle agent-generated joins.

Short forms

One-liner

Query Kafka and Iceberg together without denormalizing or cluster sprawl

30-second pitch

PhoenixAI is a real-time analytical database for data engineers building AI agents and customer-facing analytics. You query live Kafka feeds and Iceberg tables together with sub-second latency without denormalizing schemas, waiting for background merges, or managing separate clusters. Kafka CDC becomes queryable in under 5 seconds, joins run on normalized data without crashing, and the cost-based optimizer handles unpredictable agent SQL automatically. Demandbase consolidated 49 ClickHouse clusters to 1 warehouse, cut hardware 60%, and shipped agent features in under two months.

Boilerplate

PhoenixAI is a real-time analytical database that runs sub-second SQL queries across streaming data and lakehouse tables with complex joins and workload isolation. Built on the StarRocks open-source engine, it serves AI agents, customer-facing analytics, and BI dashboards at scale without denormalization, the FINAL tax, or cluster sprawl. Available as PhoenixAI Cloud (BYOC) and PhoenixAI Anywhere (self-managed).

Competitive talk track

ClickHouse

Unlike ClickHouse, PhoenixAI runs distributed joins on normalized streaming and lakehouse data without denormalization, the FINAL tax, or cluster sprawl

They'll say: ClickHouse is proven open-source with a huge community and documented sub-second performance—why switch?

You say: ClickHouse is fast on denormalized single-table scans, but its distributed joins lack data shuffling and crash on complex multi-table queries at scale. ReplacingMergeTree forces you to choose between stale duplicates or SELECT FINAL which is 5-20× slower. PhoenixAI runs the same joins on normalized data with sub-second latency, commits streaming upserts queryable in <5 seconds without FINAL, and consolidates workloads into one warehouse instead of 49 clusters. Demandbase proved it: 60% less hard

Apache Pinot

Unlike Apache Pinot, PhoenixAI handles complex multi-table joins and full SQL on normalized schemas without requiring denormalized wide tables or external tools for transformation

They'll say: Pinot is built for user-facing analytics with proven 10K+ QPS and single-digit-millisecond p99 latency—why wouldn't we use it?

You say: Pinot excels at high-concurrency single-table queries on denormalized schemas, but multi-table joins are limited and expensive, SQL completeness trails general OLAP engines, and you need external tools like Presto or Spark for transformation. PhoenixAI delivers the same 10K+ QPS and sub-second latency while running complex joins on normalized data natively, with full SQL support and cost-based optimization for unpredictable agent queries—no denormalization or external transformation required.

Apache Druid

Unlike Apache Druid, PhoenixAI runs complex joins on raw event-level data without pre-aggregation, so you can query individual rows for late-arriving enrichment and agent workloads that need full detail

They'll say: Druid is great for time-series rollups and operational dashboards with sub-second queries—what's missing?

You say: Druid's pre-aggregation at ingestion is perfect for known metrics, but queries requiring original raw rows—late-arriving dimension enrichment, individual event lookup, unpredictable agent-generated SQL—are impossible or require a separate raw data store. PhoenixAI queries raw event-level data with sub-second latency while also supporting pre-aggregated views through incremental materialized views, so you get both flexibility and speed without maintaining two systems.

Snowflake

Unlike Snowflake, PhoenixAI delivers sub-second query latency and seconds-level data freshness without 60-second minimum billing per warehouse start

They'll say: Snowflake is already our company standard with unified governance and warehouse workflows in place—why add another platform?

You say: Snowflake is the right platform for batch BI and data warehouse workloads, but query latency is 3-10 seconds typical and Snowpipe Streaming delivers 5+ second freshness—both insufficient for customer-facing analytics or AI agent workloads requiring sub-second responses. The 60-second minimum billing per warehouse start inflates costs for short, frequent agent queries. PhoenixAI complements Snowflake by handling the real-time, high-concurrency workloads Snowflake wasn't built for, querying the sa

Databricks

Unlike Databricks, PhoenixAI delivers sub-second analytical query latency without 3-7 minute cold start times or managing separate compute clusters for agent, BI, and ETL workloads

They'll say: Databricks is our unified lakehouse platform for ETL, ML, and analytics—why wouldn't we just use SQL warehouses for queries?

You say: Databricks is the right platform for Spark-based ETL, ML training, and batch analytics, but cold start time is 3-7 minutes for cluster warmup and query latency is seconds to tens of seconds—not sub-second at production scale. Streaming mode achieves millisecond latency for ingestion and transformation, not complex analytical SQL. PhoenixAI queries your Delta/Iceberg tables in place with sub-second latency, isolated warehouses for agent/BI/ETL workloads, and cost-based optimization for unpredicta

Status quo: Keep current ClickHouse deployment running, wait for it to break, or do nothing (status quo)

Unlike waiting for ClickHouse to break, PhoenixAI eliminates the denormalization rewrites, cluster sprawl, and manual tuning before they block new AI agent and customer-facing analytics initiatives

They'll say: Migration is risky, schema redesign is complex, and our current ClickHouse pain isn't severe enough to justify the operational effort right now.

You say: The pain threshold shifts when new initiatives arrive. Demandbase tolerated ClickHouse query timeouts and FINAL tax until agent workloads forced the change—then they consolidated 49 clusters to 1 warehouse, cut infrastructure 60%, and shipped agent features in under 2 months. PhoenixAI's 30-day free trial with production features lets you prove it on real workloads before committing, and SQL compatibility with StarRocks means schemas and connectors carry over. The question is whether you wait un

STAGE 8

Drafted Homepage Copy

Hero

Real-time analytical database for AI agents without denormalizing your schema

Query live Kafka feeds and Iceberg tables together with sub-second latency—no denormalization, no waiting for merges, no managing 49 clusters.

Call to action: Start 30-day free trial · Read Demandbase case study

Variant B to test: Query Kafka and Iceberg together without cluster sprawl or the FINAL tax

Problem

ClickHouse forces impossible trade-offs when AI agents hit production

Your ClickHouse deployment handled batch analytics fine. Then agent workloads arrived with unpredictable LLM-generated SQL, customer-facing dashboards tightened freshness to seconds, and you hit the limits.

- Distributed joins crash on complex multi-table queries because ClickHouse lacks data shuffling—or you denormalize into wide tables that waste 10× storage and require six-month backfills
- ReplacingMergeTree forces a choice: stale duplicate rows until background merge completes, or SELECT FINAL which is 5-20× slower on large tables
- Unpredictable agent SQL with novel join patterns breaks or times out without manual tuning per query
- Workload isolation requires separate clusters for agent, BI, and ETL—leading to 49 clusters and infrastructure cost bloat

Solution intro

A real-time analytical database built for the workloads ClickHouse can't handle

PhoenixAI runs distributed joins on normalized streaming and lakehouse data with sub-second latency. Kafka CDC upserts become queryable in under 5 seconds without the FINAL tax. Agent, BI, and ETL workloads run on the same data with independent scaling—one warehouse instead of 49 clusters.

Value props

Query streaming and lakehouse data together without denormalizing your schema

ClickHouse forces you to choose between slow distributed joins that crash on complex queries or denormalizing into wide tables that waste 10× storage and require six-month backfills. PhoenixAI runs joins across normalized Kafka feeds and Iceberg tables with sub-second latency on billions of rows.

- Run joins across normalized Kafka feeds and Iceberg tables with sub-second latency on billions of rows—ship agent features and dashboards without flattening your schema first
- Consolidate agent, BI, and ETL workloads on the same data with independent compute per warehouse—scale agent compute to 10 nodes while BI runs on 2, same tables
- Avoid the six-month schema redesign and backfill cycle every time a new dimension or join requirement arrives

Serve fresh CDC data in seconds without killing query performance

ClickHouse ReplacingMergeTree forces a choice between stale duplicate rows until background merges complete or using SELECT FINAL which is 5-20× slower. PhoenixAI commits Kafka and Flink CDC upserts queryable in under 5 seconds at 100K+ events/sec. Agents and dashboards see the latest version without waiting for merges. Query mutable streaming data with sub-second latency at 10K+ concurrent queries—no FINAL keyword, no stale duplicates, no performance cliff.

- Commit Kafka and Flink CDC upserts queryable in under 5 seconds at 100K+ events/sec—agents and dashboards see the latest version without waiting for merges
- Query mutable streaming data with sub-second latency at 10K+ concurrent queries—no FINAL keyword, no stale duplicates, no performance cliff
- Ship fraud detection, delivery ops, and agent features that require seconds-level freshness without rewriting your CDC pipelines or adding a speed layer

Run unpredictable agent SQL on production data with cost-based optimization

LLM-generated SQL with novel join patterns breaks or times out on ClickHouse because its rule-based planner can't reorder joins at runtime or adapt to query patterns it hasn't seen. PhoenixAI handles agent-generated joins and filters with a cost-based optimizer that reorders joins, pushes down runtime filters, and uses query feedback—no manual tuning per query. Run agent workloads on the same fresh normalized data as BI dashboards without separate denormalized copies or clusters.

- Handle agent-generated joins and filters with cost-based optimizer that reorders joins, pushes down runtime filters, and uses query feedback—no manual tuning per query
- Run agent workloads on the same fresh normalized data as BI dashboards without separate denormalized copies or clusters
- Ship agent features in weeks instead of waiting quarters for schema rewrites and performance testing

Proof

Data engineers are consolidating clusters and shipping agent features in under two months

- Demandbase consolidated 49 ClickHouse clusters to 1 PhoenixAI warehouse, reduced hardware 60% and storage 90%, achieving <1s query latency on agent workloads and shipping from idea to production in under 2 months. Ryan Nowacoski, Senior Engineering Manager, Data Platform: "Demandbase AI introduced unpredictable LLM-generated SQL that our previous ClickHouse-based architecture wasn't built to handle. PhoenixAI gives us a fast, isolated warehouse for agent workloads directly on our Apache Iceberg — Positioning document stage 0 customer proof, phoenixdata.ai/case-studies/demandbase, phoenixdata.ai homepage customer quote
- Coinbase runs 573B+ rows across 300+ tables and 10 blockchains with 30K messages/sec Kafka ingestion and <1s latency on TPC-H 1TB benchmark. Xinyu Liu, Senior Staff Software Engineer: "PhoenixAI changed the equation: streaming updates from Kafka become queryable within seconds, analysts get sub-second responses on live normalized data, and our AI agents operate on the same real-time dataset." — Positioning document stage 0 customer proof, phoenixdata.ai/case-studies/coinbase, phoenixdata.ai homepage customer quote
- Conductor runs agent queries scanning 6-7 GB / 180-240M rows in 389ms-2.8s with 100% decimal accuracy, shipped from idea to production in under 2 months. — Positioning document stage 0 customer proof, phoenixdata.ai/case-studies/conductor
- Pinterest achieved 50% lower p90 latency vs. Druid with 3× cost-performance efficiency using 32% of prior instances and 10-second data freshness. — Positioning document stage 0 customer proof, phoenixdata.ai/case-studies/pinterest
- Eightfold.ai delivered 2× faster query response vs. Redshift with 2× infrastructure cost reduction at scale and <1s p95 dashboard latency multi-tenant. — Positioning document stage 0 customer proof, phoenixdata.ai/case-studies/eightfold

Closing call to action

Start querying Kafka and Iceberg together in under 30 days

30-day free trial with full production features on PhoenixAI Cloud (BYOC: AWS, Azure, GCP) or PhoenixAI Anywhere (self-managed, on-prem, air-gapped). No credit card required. Connect real data sources, run complex queries, and evaluate against your actual agent, BI, and ETL workloads.

Call to action: Start free trial

Words borrowed from your customers

- PhoenixAI changed the equation: streaming updates from Kafka become queryable within seconds, analysts get sub-second responses on live normalized data, and our AI agents operate on the same real-time dataset
- Demandbase AI introduced unpredictable LLM-generated SQL that our previous ClickHouse-based architecture wasn't built to handle
- PhoenixAI gives us a fast, isolated warehouse for agent workloads directly on our Apache Iceberg tables, with the optimizer handling novel joins automatically

---

Inputs Used

Positioning document: Positioning Document Against ClickHouse for Data Engineers in any industry.pdf

Additional research: none added for this run.