WisdomAI

Whitepaper

Adaptive Context Engine

Architecture for Accurate Analytics Code Generation. For the Enterprise.

Accurate enterprise analytics code generation is fundamentally a context problem - requiring systems that interpret intent using business semantics, data reality, and operational constraints, not just model capability.

Accuracy in enterprise analytics environments is limited less by model capability and more by the system's ability to correctly interpret data in context. Production data is fragmented across systems, schemas are incomplete, naming is inconsistent, and business semantics are rarely encoded explicitly. As a result, translating agentic analytics requests – natural language or agentic – into correct, stable analytics code remains a fundamentally hard systems problem.

Bootstrapping, Governance, and Runtime Execution

WisdomAI's **Adaptive Context Engine (ACE)** is our architectural approach to solving this array of problems. We describe how enterprise context is learned, governed, and applied as a first-class system whose sole objective is accurate analytics code generation in SQL and Python. This paper covers how context is bootstrapped from existing enterprise assets and data signals, refined through real usage and human feedback,
maintained over time to prevent semantic drift and instability, and operationalized through algorithmic planning and a source-aware execution at runtime.

Together, these components form a coherent architecture for building natural-language and agentic analytics systems that are accurate, stable, and scalable across complex, multi-system enterprise environments.

Agentic Intelligence: Context & Federated Data

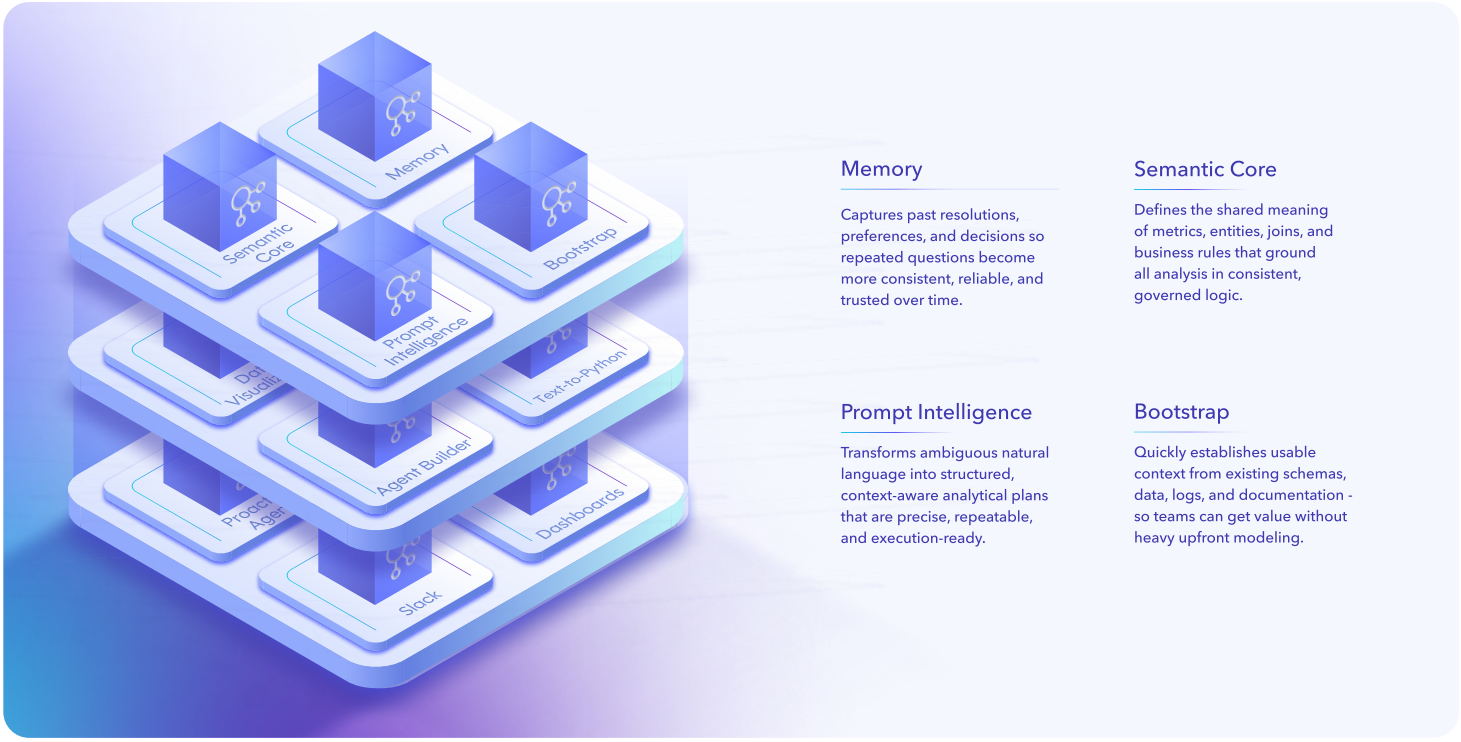
What is WisdomAI's Adaptive Context Engine (ACE)?

Enterprise context is the learned and adapted representation of how an organization's data should be interpreted in order to run agentic analytics. Its purpose is not documentation or knowledge management in isolation, but decision-making during code construction: determining what data to use, how to combine it, and which assumptions to apply when user requests are ambiguous, underspecified, or expressed in business language rather than schema-level terms.

In practice, enterprise context enables analytical systems to:

- Route questions across large and evolving sets of tables and data sources
- Disambiguate natural language intent when multiple interpretations are possible
- Apply business-correct and up-to-date joins, defaults, and temporal logic during query construction

ACE is WisdomAI's adaptive enterprise context engine– *encoding how data should be interpreted to reliably route intent, resolve ambiguity, and apply business-correct joins and assumptions during analytical execution.*



How is this different from a Semantic layer?

A semantic layer encodes structural truth: tables, joins, metrics, and calculations. Its job is to ensure queries are logically and mathematically valid. The Context Engine (ACE) is a superset to this. It includes the semantic layer, but extends beyond schemas to capture organizational meaning - business definitions, policies, assumptions, tribal knowledge, decision traces, unstructured documents, and user intent over time.

ACE extends the semantic layer with organizational meaning - policies, assumptions, and intent– while preserving structural correctness.

The Context Learning Lifecycle

WisdomAI's Context Engine learns enterprise context through a staged lifecycle that is designed to balance speed, accuracy, and governance across both cold-start, and steady-state operations:

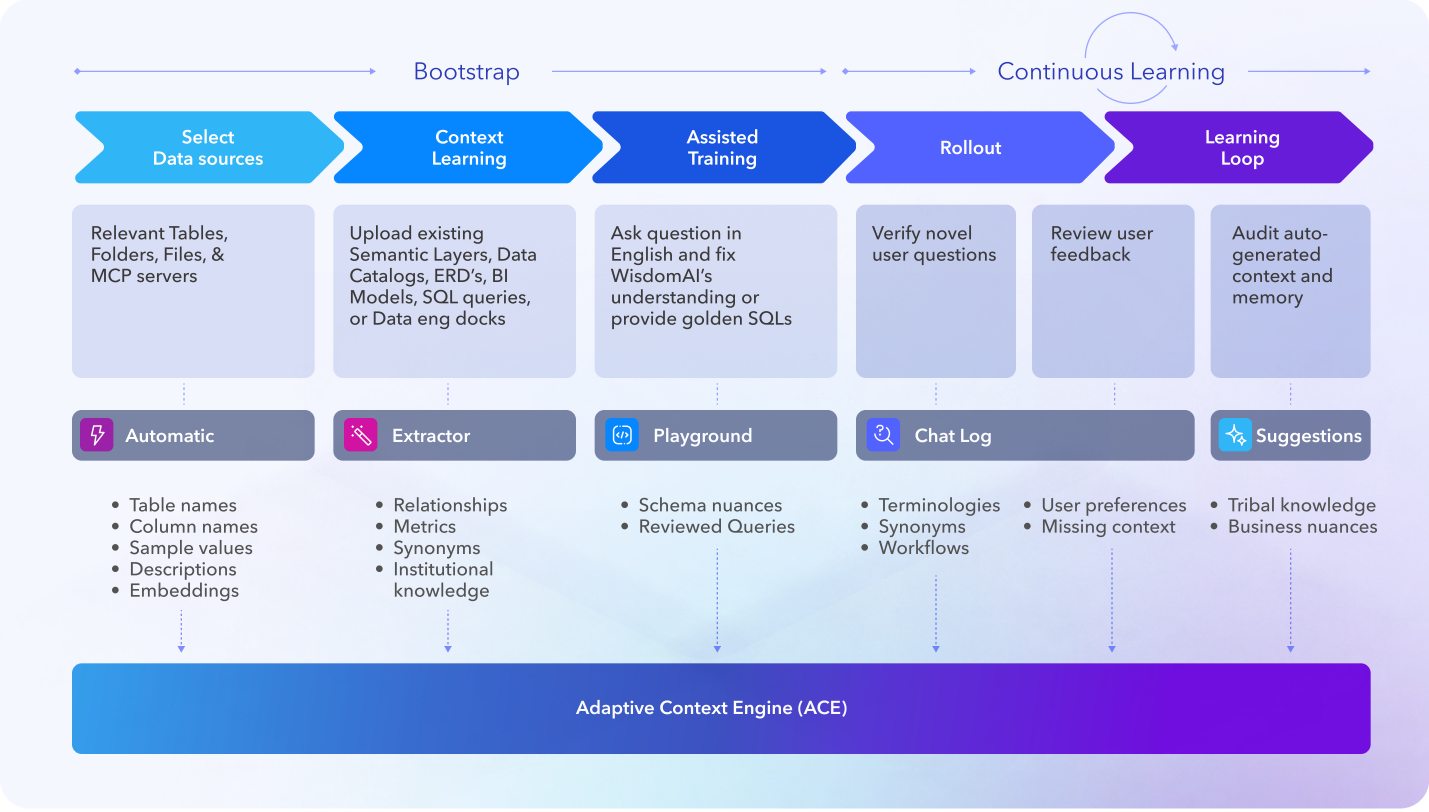
1. **Bootstrapping**: Establish an initial context baseline from enterprise assets, schemas, sampled data, and historical behavior

2. **Refinement through feedback**: Improve mappings, defaults, and assumptions using real query usage and explicit human feedback.

3. **Long-term quality maintenance**: Detect conflicts, manage semantic drift, and validate changes before promotion into active use.

ACE constructs an initial enterprise context graph by fusing human-authored metadata, historical query logs, and data-intrinsic signals—enabling correct routing, schema inference, and code generation from cold start.

This lifecycle allows the system to become useful quickly while adaptively converging towards a stable, production-grade understanding of enterprise data as usage grows and requirements evolve.



The Adaptive Context Engine bootstraps context from day one by combining human-authored knowledge, historical query behavior, and signals learned directly from data—establishing a high-coverage, reliable context baseline even in cold-start environments.

Bootstrapping Techniques: Assets, Schemas, and Data

Objective: Establish a high-coverage context baseline that enables correct routing and code construction in cold-start environments.

ACE bootstraps context using both human-authored knowledge and machine-observable signals.

Human-authored inputs include:

- **Documentation**: Wikis, runbooks, metric definitions, informal ERDs, shared documents, and other tribal knowledge
- **dbt Projects**: Models, tests, lineage, and freshness metadata
- **Data Catalogs**: Metadata from tools such as Alation, Atlan, Collibra, Unity Catalog etc.
- **BI Models**: PowerBI, Tableau, Looker models and workbooks

Where available, **historical query logs** provide a compact behavioral signal that grounds initial context in real usage.

At the same time, ACE learns directly from the data itself:

- Schema names as semantic anchors
- Sampled data helps validate types and detect nullability issues
- Key distributions and join selectivity inform relationship inference
- Naming inconsistencies and grain mismatches are surfaced early, before they cause downstream errors

The context engine establishes a high-coverage context baseline by combining human-authored metadata, historical query behavior, and signals inferred directly from data - enabling accurate routing and code generation even in cold-start environments.

Learning from Usage and Feedback Loops

Objective: Continuously refine context so that repeated questions reliably converge on the same, correct SQL and Python outputs.

Once initial context is established, ACE refines its understanding using usage-derived signals that reflect how analysts and business users actually work with data.

These signals include:

- **Analyst-authored SQL** ingested from notebooks, scripts, and version-controlled repositories
- **Query evolution signals** such as edits, execution results, and recurring error patterns observed in SQL playgrounds
- **Conversational feedback** including in-chat corrections, clarifications, and follow-up questions within the analytics experience

Taken together, these signals reveal where ambiguity exists, which joins and filters are implicitly preferred, and where important context is missing or incorrectly specified.

All learning is attributed, auditable, domain-scoped, and gated through validation workflows to prevent uncontrolled mutation of active context.

ACE continuously refines enterprise context using usage and feedback signals - driving repeated questions to converge on stable, correct SQL and Python outputs while preserving auditability and control.

Ongoing Quality Maintenance

Objective: Preserve query stability and accuracy as context evolves over time.

As more context is introduced, the risk of inconsistency increases. Conflicting assumptions – even subtle ones – can reduce determinism and erode trust, especially as questions become more complex, data sources multiply, and interactions span multiple turns. The WisdomAI Context Engine combines continuous benchmarking, session-level drift detection, and human-in-the-loop reviews before changes reach production.

Benchmarking Accuracy Over Time

WisdomAI maintains a benchmarking framework that continuously measures the accuracy and stability of generated SQL and Python against a curated golden query set and representative customer workloads.

Benchmarks are evaluated:

- **Before** new context is promoted into active use
- **After** meaningful changes to schemas, domains, or execution behavior
- **Continuously over time** as learning accumulates

In real-world deployments, this produces a predictable accuracy pattern. Customers typically see rapid improvements during early onboarding, followed by steady adaptive gains and eventual stabilization as context matures Benchmark results are tracked over time to ensure that learning consistently improves correctness-and that new changes do not introduce regressions.

Detecting Instability Through Session Analysis

Benchmarking alone is insufficient to capture real-world complexity. WisdomAI's Context Engine therefore analyzes user sessions to detect friction signals, including retries, multi-turn corrections, and unresolved follow-up questions.

These signals highlight areas where context is missing, unclear, or internally inconsistent, often before explicit errors appear in benchmarks. When this happens, the system records the signal in a structured way, pointing to where additional context or tighter constraints are needed.

All improvements are validated before promotion, ensuring that quality maintenance increases stability rather than degrades it.

95%+ accuracy at enterprise scale

10,000 end-user chat sessions

Capabilities for Managing Context at Scale

Large organizations require specialized context per team while retaining the ability to analyze across boundaries. As enterprise context grows, it must be possible to manage, inspect, share, and reuse that context without introducing global instability or duplicating effort.

WisdomAI addresses this with **Domains**: self-contained bundles of context that teams can own independently. A domain captures the assumptions, definitions, mappings, and constraints that matter for a specific team, function, or analytical surface.

By design, domains keep local decisions local– while still allowing controlled analysis across domains when needed. This lets organizations move fast with decentralized ownership, without giving up enterprise-wide consistency or trust in their analytics

Flexibility at Scale– Domains support the following core capabilities

WisdomAI domains make enterprise context both human-legible and machine-executable—enabling visual inspection, programmatic control, reuse across tools via MCP, and governed evolution through versioning, ownership, and telemetry.

contact@wisdom.ai

Simplify Visual and Programmatic cues

Domains are designed to be easy to understand and work with - for both people and systems.

Context within a domain can be:

- **Visualized**, allowing data leaders and domain owners to inspect metrics, entities, relationships, and assumptions through graphical interfaces

- **Represented programmatically**, enabling context to be defined, versioned, and reviewed as code

Together, these representations make context easy to understand, review, and maintain using existing engineering and governance practices.

Reuse Beyond WisdomAI

Domain context isn't tied to a single interface or interaction. WisdomAI is designed so context can be reused across tools and workflows– whenever and wherever it's needed. Domains are starting-point agnostic.

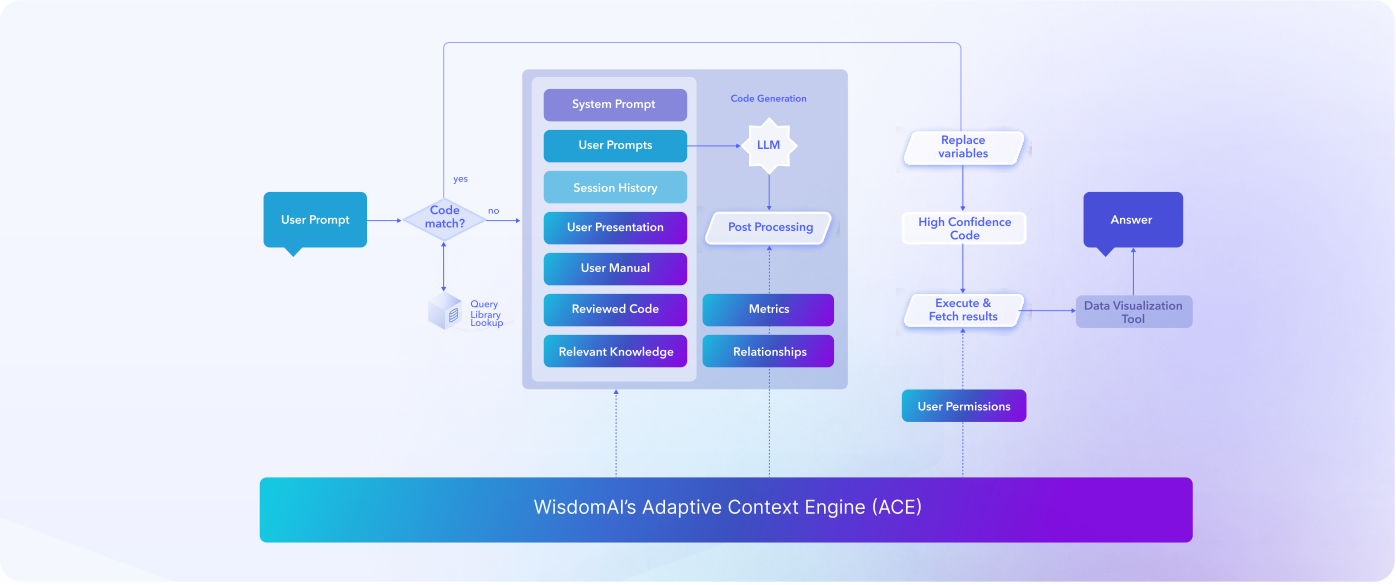
- To some, it could be a lightweight semantic layer for analytics, with metrics, views and dimensions
- To others it could act as practical data catalog abstraction that captures definitions, ownership, and relationships– without the complexity and overhead of traditional catalog system

When appropriate, domain context can be shared through MCP, allowing other tools to reuse the same governed semantics and avoid duplication.

Algorithmic Techniques for High-Accuracy Code Generation

Enterprise analytics queries are rarely simple. Real-world questions are often multi-step, multi-turn, and span multiple data sources, with deeply nested logic and implicit assumptions carried across interactions.

Context Engineering for Text-to-SQL



Distribution of real-world query complexity, illustrating multi-turn workflows, cross-source joins, and nested analytical structures.

WisdomAI applies explicit algorithmic control to ensure that learned context is used deterministically and correctly during code generation. Rather than relying on a single generation pass, the system constrains and structures how context influences SQL and Python construction.

Key techniques include:

- **Large-Context Search and Pruning**: Actively finding and using only the most relevant schemas, metrics, relationships, and past resolutions needed to answer the current question.
- **Data-Aware Query Segmentation**: Breaking down ambiguous natural language into data-aligned components before generating any code.
- **Multi-Step Planning**: Decomposing complex analytical requests into clear, verifiable steps before assembling the final result.
- **Session Context Management**: Preserving assumptions, bindings, and intermediate results across multi-turn analytical workflows.
- **Feedback-Guided Retry**: Improving plans and assumptions using execution feedback, rather than blindly regenerating queries or code.
- **Special-Case Handling**: Improving plans and assumptions using execution feedback, rather than blindly regenerating queries or code.
- **Evaluations**: Continuously validating generated plans, queries, and outputs against trusted datasets and expected behaviors to catch accuracy drift and prevent silent failures.
- **Memory Harness:** Remembering past resolutions and preferences so repeated questions stay consistent over time.
- **Guardrails and Policy Enforcement:** Applying enforced rules on joins, metrics, time semantics, and access to keep outputs compliant and explainable.

Additional Data Sources and Unstructured Context

Enterprise knowledge does not live exclusively in warehouses. Critical business context is often embedded in unstructured repositories and external systems that capture operational detail, institutional knowledge, and intent that is not reflected in schemas alone.

WisdomAI incorporates these sources as core context inputs, enabling accurate interpretation and code generation even when key information exists outside structured tables.

Unstructured Repositories

WisdomAI integrates with unstructured enterprise repositories such as SharePoint, document management systems, and internal knowledge bases.

These sources frequently contain:

- Business definitions and operating procedures
- Design documents and architectural decisions
- Planning artifacts, policies, and operational runbooks

Content from unstructured repositories is parsed, chunked, and indexed to extract entities, relationships, constraints, and assumptions that inform downstream query interpretation and code construction.

In addition, WisdomAI can use LLMs to extract structured facets from unstructured text. These facets include artifacts like topics, entities, sentiment, intent, ownership, and lifecycle stage, derived directly from free-form documents and communications. Once extracted, these facets are stored in a structured form. This allows text-heavy repositories to be analyzed just like structured data- supporting filtering, aggregation, and trend analysis- rather than being limited to simple search and retrieval.

MCP Servers as Data Sources

In addition to unstructured repositories, WisdomAI treats Model Context Protocol (MCP) servers as first-class data sources, not merely auxiliary context providers.

High-quality MCP servers expose well-defined schemas, metrics, and queryable interfaces for operational systems such as Google Analytics, Salesforce, Jira, and HubSpot. These systems often represent authoritative sources for customer behavior, revenue activity, issue tracking, and go-to-market operations - data that is critical for enterprise analysis but typically siloed outside the warehouse.

WisdomAI integrates MCP-backed systems directly into its planning and execution runtime, allowing MCP data to participate in analytical workflows alongside warehouse data. MCP servers are accessed with full preservation of native authentication, authorization, and permission models, ensuring governed access to operational data.

Case Study: Google Analytics via MCP + BigQuery

A large digital media and consumer-tech company wanted to analyze product usage data from Google Analytics alongside revenue and customer attributes stored in BigQuery. Traditionally, this meant building and maintaining continuous ETL pipelines to move event-level data into the warehouse—adding cost, latency, and ongoing operational burden.

With WisdomAI, the team took a different approach. Using MCP, WisdomAI connects to Google Analytics directly as a governed data source and joins those results with BigQuery tables at runtime. This allowed the company to answer questions like funnel conversion by customer segment or campaign performance by revenue tier—without duplicating data or managing fragile ingestion jobs.

More broadly, this is how WisdomAI is designed to work. As an MCP-native analytics platform, WisdomAI can consistently leverage MCP-backed data sources across conversational analysis, agents, and dashboards– rather than limiting access to a single chat interface.

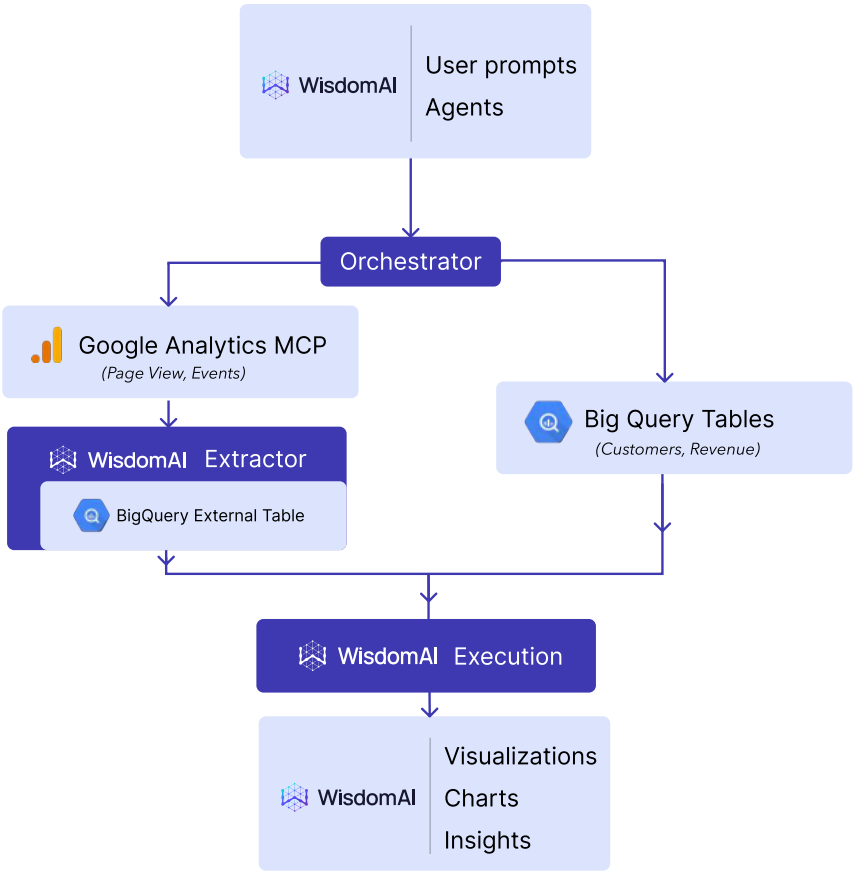


Fig 4. *WisdomAI's Adaptive Context Engine extracts context from MCP and BigQuery external tables to combine Google Analytics and warehouse data at runtime - delivering trusted insights without potentially expensive ETL or data duplication*

Cross-Dataset Execution Runtime

Once context is available– across warehouses, MCP sources, and unstructured repositories– the remaining challenge is executing correctly and efficiently across datasets.

Source-Aware Planning and Recomposition
WisdomAI plans each query based on where data lives, runs work in the right systems, and recombines results into a single, auditable answer.

Warehouse ↔ MCP Workflows
WisdomAI can combine warehouse queries with MCP-backed systems in one workflow, without copying data or weakening security.

Warehouse External Tables and Pushdown Joins
When supported, WisdomAI pushes joins and filters directly into the warehouse to minimize data movement, inherits user-defined data governance, and maximizes performance.

Towards Advanced Data Science

High-fidelity data science requires context. ACE enables grounding of advanced Python workflows and deep analysis in consistent enterprise semantics and institutional knowledge - enabling rigorous modeling, forecasting, and explanation without sacrificing correctness, traceability, or trust.

Python for Advanced Computations
When analysis exceeds SQL's expressiveness, WisdomAI executes Python as an explicit step in the plan. This supports advanced workflows such as:

- Statistical testing and modeling
- Time-series forecasting
- Custom transformations and feature engineering

Deep Analysis
WisdomAI's Adaptive Context Engine powers investigations that go beyond surface-level analytics to explain why outcomes changed and what they mean. This allows teams to correlate numbers with decisions, policies, and operational signals in a traceable, repeatable way—supporting rigorous deep analysis and research workflows that go well beyond traditional BI or chat-based analytics, without sacrificing governance or trust.

Conclusion

Accurate Agentic Analytics is ultimately a systems problem. In enterprise environments, correctness depends not only on model capability, but on how context is learned, governed, and applied across ambiguous questions, evolving data, and heterogeneous systems.

This paper introduced **WisdomAI's Adaptive Context Engine (ACE)** as a unifying architectural abstraction for solving this problem. By treating context as a first-class system—bootstrapped from existing assets, refined through real usage, maintained through benchmarking and session analysis, and organized through domains– ACE enables analytics code generation that improves in accuracy and stability over time.

Algorithmic planning and a source-aware execution runtime ensure that this context is applied deterministically, even for complex, multi-turn, and cross-dataset questions. Support for unstructured repositories, MCP-backed systems, and advanced Python-based analytics further extends the architecture beyond traditional warehouse-centric approaches.

As illustrated in Figure 1, the Adaptive Context Engine provides a coherent foundation for building agentic analytics systems– natural language or beyond– that scale across teams, tools, and data sources– while maintaining the accuracy and trust required for real-world enterprise decision making.

Get in touch with us!

wisdom.ai
contact@wisdom.ai