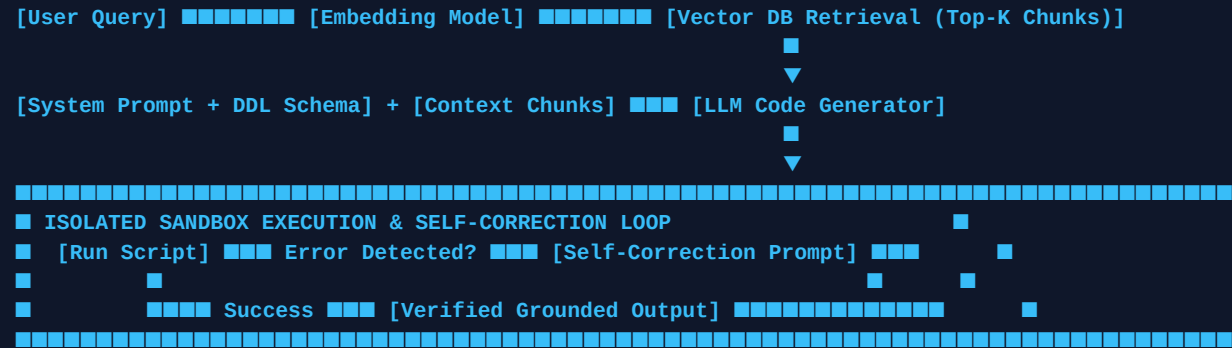


8 Advanced Technical Topics • Core Execution Rules • Architectural System Diagrams

Key Execution Rules & Architecture Principles

- Grounding via Vector Context: Ingest proprietary documents via chunking and vector embedding (watsonx / FAISS / Milvus) to ground LLM context and prevent hallucinations.
- System Prompt DDL Embedding: Supply database table schemas (DDL) directly inside system prompts to ensure accurate SQL code generation without schema drift.
- Sandboxed Execution & Self-Correction Loop: Execute LLM-generated code inside isolated sandboxes (Docker / gVisor). Catch runtime errors and loop back for automated code correction.

DIAGRAM: RAG & SANDBOXED EXECUTION PIPELINE



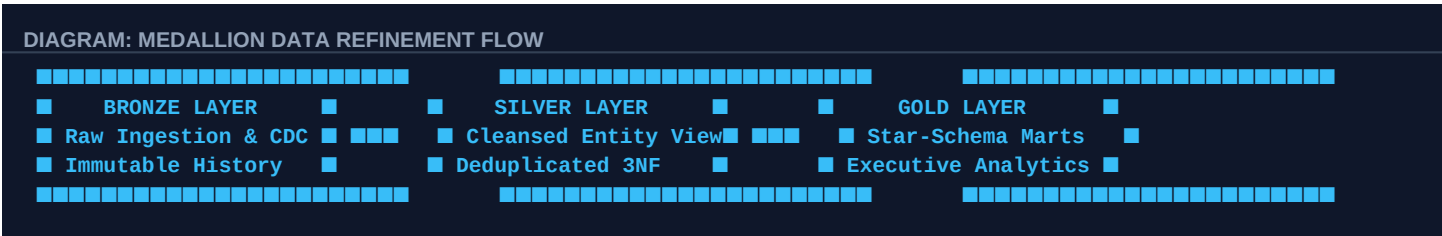
3. DATA ARCHITECTURE & LAKEHOUSE: Medallion Lakehouse Pattern: Bronze, Silver & Gold Pipeline

Core Concept: The Medallion architecture progressively refines data structure and quality across Bronze (raw), Silver (cleansed), and Gold (curated) layers on Databricks Delta Lake.

Key Execution Rules & Architecture Principles

- Bronze Layer (Raw Ingestion): Land source CDC data 'as-is' with ingestion metadata timestamps. Retain immutable history for full auditability and reprocessing.
- Silver Layer (Cleansed Enterprise View): Perform deduplication, schema alignment, and 'just-enough' cleansing. Model in 3NF or Data Vault for cross-departmental analytics.
- Gold Layer (Curated Datamarts): Transform Silver entities into read-optimized Kimball star-schemas or flattened datamarts for executive reporting and ML features.

Layer	Data Quality & Format	Primary Target Users	Data Modeling Style
Bronze	Raw, Append-Only, Source Format + Metadata	Data Engineers / CDC Systems	Source System As-Is (JSON/Parquet)
Silver	Cleansed, Deduplicated, Conformed View	Data Scientists / Analysts	3NF / Data Vault / Transactional
Gold	Aggregated, Business Rules Applied, Curated	Business Users / BI Dashboards	Kimball Star-Schema / Datamarts

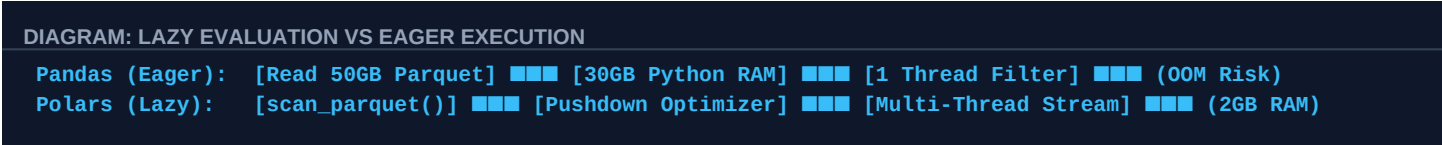


4. PYTHON & HPC: Polars Lazy Evaluation & Apache Arrow Memory Pattern

Core Concept: Polars leverages a multi-threaded Rust execution engine and Apache Arrow memory layouts to deliver vectorized streaming analytics on large datasets.

Key Execution Rules & Architecture Principles

- **Polars Rust Core & Multi-Threading:** Polars dispatches queries across all CPU cores in Rust without Python GIL contention, beating single-threaded Pandas by 10x-20x.
- **Lazy Frame Optimization:** Use `pl.scan_parquet()` to allow the query planner to apply predicate pushdown (skipping Parquet row groups) and projection pushdown (reading selected columns).
- **Zero-Copy Apache Arrow Integration:** Standardize on Apache Arrow columnar memory buffers for zero-copy data passing between Polars, DuckDB, and PyArrow.

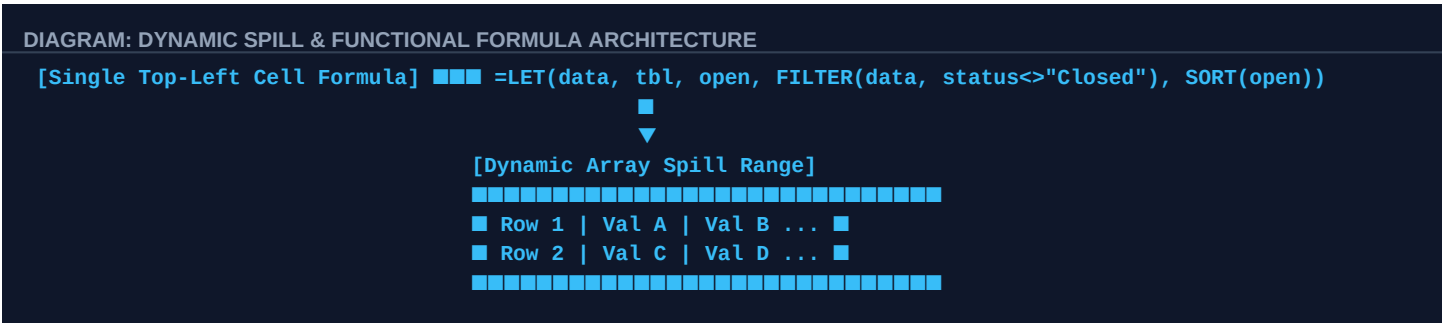


5. SPREADSHEET ARCHITECTURE: Dynamic Array Framework & LAMBDA Functions in Excel

Core Concept: Modern Microsoft 365 Excel combines Dynamic Arrays, LET variables, and named LAMBDA formulas into functional, modular reporting engines.

Key Execution Rules & Architecture Principles

- **Dynamic Array Spill Matrix:** Write formulas in a single top-left cell; Excel automatically spills results across adjacent rows/columns without manual drag-copying.
- **Formula Modularity via LET & LAMBDA:** Use LET to define named intermediate variables for readability, and LAMBDA to encapsulate reusable custom worksheet functions without macros.
- **Higher-Order Functional Helpers:** Iterate over dynamic arrays efficiently using MAP, REDUCE, SCAN, BYROW, and FILTER to eliminate legacy helper columns.

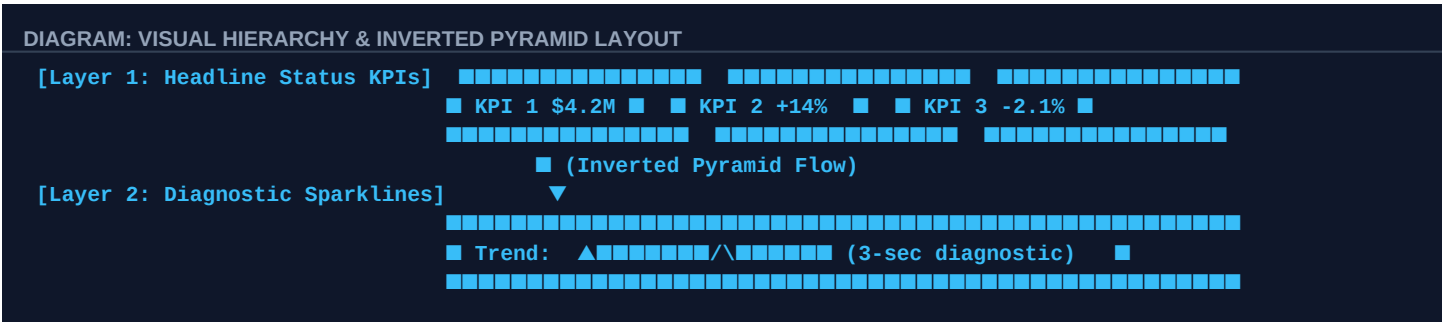


6. DASHBOARD DESIGN & BI: The Pre-Attentive Visual Hierarchy & Executive KPI Grid Pattern

Core Concept: Executive dashboards leverage pre-attentive visual processing, inverted pyramid layering, and 3-second variance sparklines for optimal decision velocity.

Key Execution Rules & Architecture Principles

- Pre-Attentive Cognitive Processing: Design layouts around 250ms visual cues (color, position, size) to convey status instantly before numerical reading occurs.
- F-Pattern & Z-Pattern Scanning: Position the primary KPI in the top-left corner where human visual scanning naturally initiates.
- Inverted Pyramid Hierarchy: Structure dashboards in 3 layers: Layer 1 (Status KPIs), Layer 2 (Diagnostic Trends & Sparklines), Layer 3 (Granular Drilldown Tables).

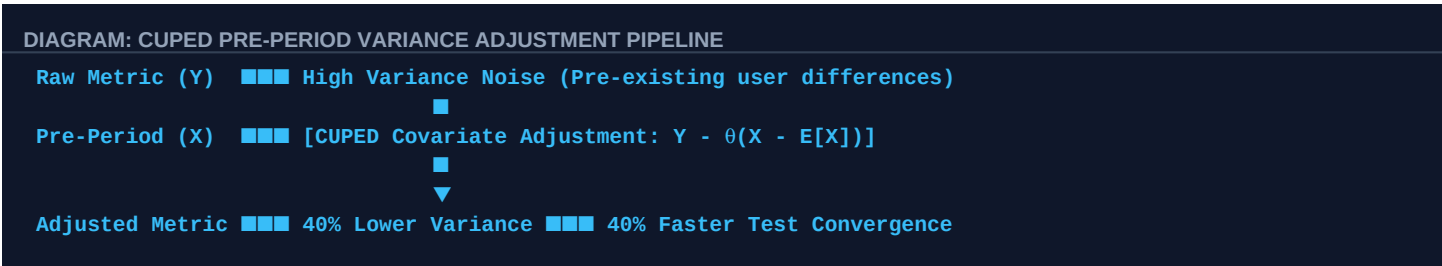


7. EXPERIMENTATION & CAUSAL INFERENCE: CUPED Variance Reduction & A/B Testing Guardrails

Core Concept: Controlled-experiment Using Pre-Experiment Data (CUPED) applies regression adjustments to eliminate pre-existing variance in online experimentation.

Key Execution Rules & Architecture Principles

- Pre-Experiment Covariate Adjustment: CUPED utilizes historical pre-period user data (X) to subtract pre-existing variance from post-period metrics (Y).
- Variance Reduction Formula: CUPED adjusted metric $Y_{cuped} = Y - \theta(X - E[X])$, where $\theta = Cov(Y,X) / Var(X)$.
- Accelerated Test Convergence: Reduces metric variance by up to 40%, enabling A/B tests to reach statistical significance 40% faster with smaller sample sizes.



8. PREDICTIVE ML & FEATURE ENGINEERING: Offline-to-Online Feature Store Pattern for Predictive Pipelines

Core Concept: Feature stores unify point-in-time historical batch training with low-latency Redis online serving to eliminate training-serving skew in ML systems.

Key Execution Rules & Architecture Principles

- Unified Feature Definition: Define features declaratively in a central registry (e.g. Feast) shared across training and serving.
- Point-in-Time Point Joins: Generate offline training datasets via exact point-in-time joins on historical batch tables (Amazon Redshift/S3) to eliminate data leakage.
- Sub-10ms Real-Time Serving: Materialize latest feature vectors into Redis in-memory key-value stores for sub-10ms online inference serving.

