Domain Target State Architecture (DTSA) Template for Analytics

1. Header Information (Governance & Control)

Field	Description
Domain	High-Frequency, Time-Series Analytics
Author / Owner	(Lead Architect name)
Version / Date	(Must be version controlled)
Review Cycle	(Quarterly, with a focus on Compute/TCO review)
Approved By	(Domain Council / Enterprise ARB, and Investment Management Leadership)

2. Executive Summary (The Analytical Mandate)

Current State Challenges

- **Keep Existing:** Fragmented data sources, reliance on batch processing, complex model versioning.
- **Analytical Focus:** High **Time-to-Insight Latency** for complex runs, unpredictable **Cost-per-Analytics Run (FinOps)** due to inefficient resource utilization, and model/data synchronization issues between systems.

Future State Objectives

- **Core Goals:** Achieve sub-second query latency for client-facing reports, 99.99% data fidelity for all time-series inputs, and a 25% reduction in the unit cost of compute.
- **Technical Goals:** Enable **model governance and traceability** for all Analytics results, and establish fully **elastic compute scaling** based on market activity.

Business Value

- **Decision Support:** Faster, higher-confidence investment decision-making.
- **Client Value:** Low-latency, interactive analysis for website and client portals.

- **Financial Value:** Clear **TCO: Cost-per-Analytics Run** predictability and control.

3. Business Context (Data, Risk & Demand)

- **Business Priorities:** Low-latency factor analysis, GIPS compliance reporting, multi-asset class support, and enabling new alpha-generating models.
- **Regulatory Drivers:** Focus on **Model Risk Management (MRM)**, data lineage for auditability, and GIPS/SEC reporting standards.
- **External Dependencies:** High-frequency market data feeds, enterprise security master, and portfolio accounting systems.

4. Target State Principles (Modeling & Mandates)

1. **Time-Series Data Fidelity & Domain Modeling:** Data is ingested as high-fidelity, auditable **immutable time-series objects** with domain-specific metadata. **All models (e.g., factor, risk) must be versioned and traceable** back to the Analytics result.
2. **Compute Efficiency & Cloud-Native Elasticity:** Architecture must scale horizontally based on analytical workload and optimize for **cost-per-query/cost-per-Analytics run (FinOps).** Leverage serverless/container orchestration for burst capacity.
3. **Low-Latency Query Contract:** All critical reporting APIs must define and meet explicit **Service Level Objectives (SLOs) for Query Latency** (e.g., <500ms for front-end access).
4. **In-Memory Optimization:** The design must prioritize the movement and processing of data in **Memory/High-Speed Cache** over disk reads.

5. Target State Architecture Views (The High- Blueprint)

5.2 Application & Analytical Service View

- **Core Components:** Clearly define the **Analytics Engine/Modeler** (the in-memory component), the **Distributed Compute/Execution Layer**, the **Caching Layer**, and the **Query/Reporting Service**.
- **API Contracts:** Define APIs for **workload submission** (e.g., submit a 10-year Analytics run for Portfolio X), **real-time query**, and **status monitoring.**

5.3 Data View (In-Memory & Storage)

- **Canonical Data Domains:** Focus on **Reference Data, Time-Series Market Data (Prices/Factors), and Transactional Snapshots (Holdings/Trades).**

- **Data Flow & Tiering:** Detail the flow from **Cold Storage (Data Lake/Warehouse) to Hot Cache/In-Memory Object Store.**
- **Time-Series Management:** Define standards for **data versioning, immutability, gap-filling, and interpolation** of time-series objects.

5.5 Technology / Infrastructure View (Compute & Memory)

- **Compute Cluster:** Specify the required **Memory-optimized (R or M series) or Compute-optimized (C series) instance families** and the orchestration tool (e.g., Kubernetes, Spark).
- **In-Memory Store:** Specify the architecture of the **Shared Data Grid/Distributed Cache** (e.g., specialized time-series DB, high-availability Redis cluster) and its replication strategy.
- **Data Access Layer:** Mandate a **high-throughput, low-latency connector** for data retrieval.
- **Programming Environment:** Specify the mandatory programming languages/ frameworks optimized for numerical computing (e.g., Python/Pandas, specialized C++ libraries, distributed compute frameworks).

6. Architectural Runway & Phasing (Model & Data Readiness)

- **Sequencing Drivers: Data Readiness** (i.e., ensuring 10+ years of high-quality time-series data is loaded and validated) is the **primary blocking dependency.**
- **Phase 1 (Enablement/De-risking):** Focus on **Data Ingestion Pipeline Certification** and **Model Environment Setup.** Must prove the ability to load a benchmark data set into the in-memory store under a specific time limit (e.g., <5 minutes).
- **Risks & Mitigations: Cost Runaway Risk** (mitigate by setting FinOps budget guardrails on compute clusters), **Data Quality Gaps**, and **Talent Risk** (shortage of specialized time-series/quant engineers).

7. KPIs & Success Metrics (Analytics Specific Outcomes)

Metric Category	Key Indicator (KPI)	Target Goal	Accountability
Performance & Speed	**95th Percentile Query Latency** (for front-end reports).	<500ms	Engineering Leadership
	Analytics Job Completion Time (e.g., time to run end-of-day Analytics for all funds).	<20 minutes	Lead Architect/ Engineering
Cost & Efficiency	**TCO: Cost per Analytics Run/Compute Hour** (The true unit cost).	25% YOY reduction	Lead Architect/ FinOps
Data Quality	**% of Time-Series Objects with Zero Gaps/ Errors** (Mandatory pre-processing quality).	99.9%	Data Governance/ Domain Owner
Operational Impact	**Cache Hit Rate** for most frequently requested portfolios/reports.	90%	Engineering Leadership
Governance	**Model Traceability Score** (100% of results link to model version and input data snapshot).	100%	Risk/ Compliance