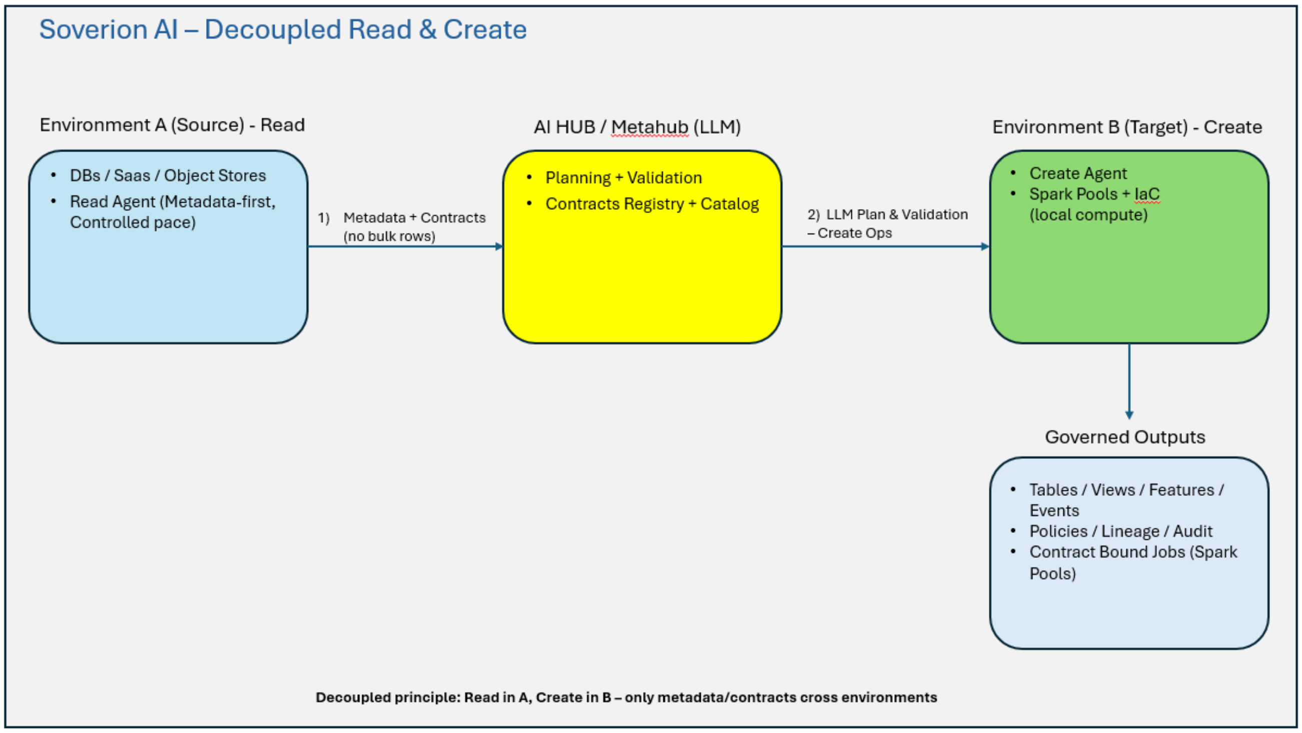
A Zero-Pipeline Architecture for the AI-Driven Enterprise

Decoupled Read & Create: Data Contracts and Spark Pools as LLM-Driven Consumables

Date: January 10, 2026

Executive Overview

ETL has served enterprises for decades, but its hallmark—the tightly coupled movement of data via pipelines—now collides with modern demands: low-copy integration, rapid change, data sovereignty, and real-time readiness. A new pattern has emerged: LLM-orchestrated Read & Create agents, architected to read where data lives and create governed artifacts where they are consumed, without transporting raw data through bulk pipelines. Two enablers make this possible: (1) Data Contracts (machine-readable specifications of inputs, transformations, and explicit outputs) that shift LLMs from general inference to deterministic production outcomes; (2) Spark Pools—ephemeral compute in the target environment—invoked by the Create Agent to execute contract-bound jobs close to consumers, validating outputs and lineage under Infrastructure-as-Code (IaC) governance.



Soverion AI – Decoupled Read & Create
Environment A (Source) - Read
DBs / Saas / Object Stores
Read Agent (Metadata-first, Controlled pace)
1) Metadata + Contracts (no bulk rows)
AI HUB / Metahub (LLM)
Planning + Validation
Contracts Registry + Catalog
2) LLM Plan & Validation – Create Ops
Environment B (Target) - Create
Create Agent
Spark Pools + IaC (local compute)
Governed Outputs
Tables / Views / Features / Events
Policies / Lineage / Audit
Contract Bound Jobs (Spark Pools)
Decoupled principle: Read in A, Create in B – only metadata/contracts cross environments

Figure: Decoupled Read & Create with metadata-only boundary (no bulk rows cross environments).

Why Pipelines Falter

Pipeline complexity and fragility—especially incremental updates and schema drift—drive operational overhead and delay insights. ETL jobs often require tuning, duplicate data across stores, and cause production impact during heavy extracts.

The Decoupled Read & Create Model

Read Agent (Environment A): discovers structure and semantics via controlled, metadata-first reads (direct/replica/catalog) with rate limits and audit trails; emits Data Contract metadata (schemas, types, constraints, policies), profiles, and lineage pointers—not bulk rows. Create Agent (Environment B): consumes contracts, orchestrates Spark pools and IaC to produce contract-defined artifacts locally, leveraging federated/external tables and materialize-on-demand for hot paths, with optional CDC for deltas.

Data Contracts as First-Class Consumables

Data Contracts are formal agreements specifying inputs, transformations, explicit outputs, and policy annotations, enforced in CI/CD and runtime. They bind the LLM's plan to deterministic targets—schema, types, keys, and minimum quality thresholds—preventing drift and enabling governance and auditability.

Spark Pools as Ephemeral Compute Consumables

Managed, elastic Spark clusters provisioned on demand in the target environment. The Create Agent invokes them to execute contract-bound jobs (aggregation, entity resolution, quality checks) close to consumers, then releases compute. Autoscale/auto-pause and node sizing control cost and performance.

Avoiding Bulk Data Movement

Replace pipelines with virtualization and deltas: (1) Schema-only replication and external tables/federated catalogs for remote queries without copying; (2) View virtualization—publish views over federated sources, materialize only hot paths; (3) Event/CDC assimilation—persist required subsets via log-based CDC to keep targets fresh without batch loads.

Reference Architecture

Environment A (source) → AI Hub/Metahub (LLM plan + validation) → Environment B (IaC + Spark pools + catalog). Bronze can remain virtual/external; Silver/Gold materialize on demand under contracts. Federation to external DBs or file stages avoids wholesale copy.

Operating Model & Guardrails

Least-privilege access; row/column-level policies embedded in contracts and applied via IaC. Validators enforce schema/types/keys/quality before writes. Observability via versioned contracts, IaC manifests, Spark run IDs, federated query traces, and lineage records. Change management through contract versioning and metadata refresh without job rewrites.

Engineering Trade-offs

Latency vs. freshness mitigated with caching/materialization in B; cost vs. control managed via autoscale/auto-pause and tracking cost per contract run; accuracy vs. agility balanced by contracts and post-run validation; consistency ensured via idempotent upserts and CDC when needed.

Case Study: ERP On-Prem → Cloud Analytics without Pipelines

Read Agent catalogs ERP schemas and authors contracts with PII policies—no bulk rows extracted. Create Agent deploys schemas/views/policies via IaC, wires external tables for secure queries, and invokes Spark pools to materialize gold metrics under contract validation. Optional log-based CDC streams deltas for near-real-time dashboards—no batch loads.

Adoption Roadmap

(1) Stand up Read Agent in A and emit contracts; (2) Stand up Create Agent in B with Spark pools and IaC; (3) Pilot virtualized analytics; (4) Introduce selective CDC for hot paths; (5) Scale and decommission brittle pipelines.

KPIs to Prove Value

Zero-pipeline coverage; egress avoided; contract pass rate (schema/quality); cost per contract run; change lead time from schema change to compliant output.

The Long View: AI-Native Integration

Data Contracts bridge intent to reliable outputs; Spark pools execute deterministically near consumers; Read & Create agents form a semantic fabric where ETL, RPA, iPaaS, and knowledge converge under contract governance.

Sources & Further Reading

• AI-Driven ETL — metadata-first reads, Metahub/LLM orchestration, IaC deployment — Uploaded whitepaper

• Medallion architecture (bronze/silver/gold) — Microsoft Learn / Databricks docs — https://learn.microsoft.com/en-us/azure/databricks/lakehouse/medallion

• Schema-on-read vs schema-on-write — Dremio wiki / industry references — https://www.dremio.com/wiki/schema-on-read-vs-schema-on-write/

• Spark pools (Azure Synapse) — overview & configurations — https://learn.microsoft.com/en-us/azure/synapse-analytics/spark/apache-spark-overview

• Spark pool configurations (autoscale/auto-pause, node sizing) — https://learn.microsoft.com/en-us/azure/synapse-analytics/spark/apache-spark-pool-configurations

• External tables & federation — Snowflake docs; Databricks federation guides — https://docs.snowflake.com/en/user-guide/tables-external-intro

• Databricks Lakehouse Federation to Snowflake — https://learn.microsoft.com/en-us/azure/databricks/query-federation/snowflake-entra

• CDC best practices — architecture overviews and log-based guidance — https://www.redpanda.com/guides/fundamentals-of-data-engineering-cdc-change-data-capture

• Data Contracts — Chad Sanderson interviews and guides — https://www.superdatascience.com/podcast/sds-825-data-contracts-the-key-to-data-quality-with-chad-sanderson

• Data Contracts — DataHub perspective — https://datahub.com/blog/the-what-why-and-how-of-data-contracts/