Proposal: Replace SSIS with a modern, secure “no-ETL” + ELT data platform

Executive summary

SSIS is limiting us (batch-only, weak API connectivity, heavy engineering). The most modern approach that matches “no E or L” is a **Logical Data Layer / Data Virtualization** platform that provides a **single governed access layer across APIs, databases, and files without replication**.

Recommendation: adopt a **hybrid architecture**:

1. **Data Virtualization** for fast, governed access (near-real-time where needed).
2. **Modern ELT** for curated analytics/ML datasets (scheduled loads, low latency needs today).

This minimizes engineering effort, improves security posture (central controls), and aligns with our 6-month Azure roadmap.

Our Requirements

- **Sources:** REST APIs, databases, files (no Kafka/streaming required).
- **Use cases:** analytics, warehousing, operational reporting, ML workflows.
- **Deployment:** on-prem now; Azure in ~6 months; Kubernetes later OK.
- **Scale:** millions of rows; latency not critical today.
- **Team:** shift toward **Python-first**; minimal engineer headcount.
- **Infosec:** strong access control, auditability, encryption, secrets management.

Recommended Target Architecture

Layer 1 — Logical Data Layer (Zero-ETL)

Purpose: “no E or L” access for BI, ad-hoc analysis, and operational queries across systems.

Best options to consider:

- **Denodo Platform (commercial, fastest time-to-value):** centralized logical access layer across data warehouses/lakes, databases, APIs, and files; unified semantics/metadata; designed for “no replication” access.
  Security overview includes end-to-end TLS patterns between consumers ↔ Denodo ↔ sources.

- **Trino (open source):** distributed SQL query engine built around connectors/federation; queries data where it lives.
- **Starburst Enterprise (commercial Trino distribution):** adds enterprise security/governance and hardening around Trino (good fit for regulated environments).
- **Dremio (commercial/open-core):** emphasizes semantic layer + "virtual datasets" with no data movement (useful if you want a semantic layer for BI).

Layer 2 — ELT (for curated, historical, ML-ready datasets)

Purpose: build a governed warehouse/lakehouse for reporting consistency, heavy joins, feature tables, reproducible pipelines.

Best options to consider:

- **Airbyte (open-core, self-hostable):** "600+ sources" and strong connector ecosystem for APIs/DB/files; supports self-hosted control/sovereignty.
- **dbt (open source):** transforms data *in the warehouse* using modular SQL models; turns SQL workflows into governed pipelines.
- **Dagster (open source):** Python-based orchestration to schedule/monitor pipelines (Airbyte/dbt/Python jobs).
- **Azure Data Factory (managed, Azure-native):** REST connector support and best-practice security guidance; strong fit once Azure lands.
  Supports **self-hosted integration runtime** to securely move data between on-prem and cloud.
- **Fivetran (managed SaaS ELT):** strong compliance posture (SOC reports/ISO listed in their trust/security materials).

Optional Azure acceleration: "virtualize without copy"

If you adopt **Microsoft Fabric**, **OneLake shortcuts** can "connect to existing data without directly copying it" (useful for faster Azure adoption and "less movement").

Options comparison (pick one path)

Option	Best for	Products (highlight)	Open source bias	Azure fit	Security/InfoSec posture
A. Recommended Hybrid (Build-first, bank-controlled)	Lowest vendor lock-in, strong control, small team	**Trino** + **Airbyte** + **dbt** + **Dagster**	High	Strong (move to AKS/VMs later)	Requires solid platform hardening; central governance via Trino + DB controls

Option	Best for	Products (highlight)	Open source bias	Azure fit	Security/InfoSec posture
B. Recommended Hybrid (Buy-first, lowest engineering)	Fastest implementation + enterprise governance	**Denodo** (or **Starburst Enterprise**) + (ADF/Airbyte) + dbt	Medium	Strong	Enterprise security model + centralized governance; Denodo sec overview available
C. Azure-native	If Azure standardization is the priority	**Azure Data Factory** + (Fabric OneLake shortcuts) + dbt	Low	Best	Mature Azure security practices + managed identity patterns; OneLake shortcuts reduce copying in some cases
D. Fully managed ELT	Minimum ops burden	**Fivetran** + dbt (+ optional virtualization later)	Low	Good	Strong compliance materials; validate vendor risk + data residency

---

Final Recommendations

Choose Option B if you want the safest path with the fewest engineers:

- **Denodo** (or **Starburst Enterprise**) as the governed "no-ETL" access layer.
- Use **Azure Data Factory** for ingestion + **dbt** for transformations once Azure is live.

Choose Option A if budget and open-source control matter most (and you can run a small platform):

- **Trino + Airbyte + dbt + Dagster** (all Python-friendly; scalable; self-hostable).

Both options meet every stated parameter; the difference is **engineering/operations vs. vendor licensing**.

Security/InfoSec Notes

- **Centralized governance:** virtualization layer becomes the single "front door" for data access (policies/audit point).
- **Encryption in transit:** Denodo documents end-to-end SSL/TLS patterns; similar controls exist in enterprise Trino distributions and Azure services.
- **On-prem ↔ cloud connectivity:** ADF supports self-hosted integration runtime for controlled data movement.

- **Vendor compliance (if managed ELT):** Fivetran publishes security/compliance materials via its trust/security pages - still requires our vendor risk review.