Mphasis
The Next Applied

MPHASIS · NEXT-GEN DATA SOLUTIONS THESIS

Secure Databricks Serverless Adoption

Prepared for Global Bank's technology, data, security, governance, and AI leadership

Authors

Sunny Sharma, Krishna Battula, Mayank Gupta

Contents

1. Executive Thesis

Most banks are approaching Databricks Serverless as a compute modernisation initiative, when the real risk, delay, and cost sit elsewhere. That is the wrong lens. The Client's interest in Databricks Serverless Compute should be framed as an operating-model transformation, not a compute-platform decision. The opportunity spans reduced infrastructure friction, faster governed data-product delivery, stronger auditability, and a more credible foundation for AI-ready data consumption.

Two priorities are inseparable: securely deploying Databricks Serverless for production data, and building a unified discoverability layer across Azure, Databricks, Snowflake, and Collibra. These are not separate work-streams — both point to the same underlying requirement: a control plane for how agents, citizen users, and shadow AI patterns discover and use data.

> Serverless compute accelerates execution, but unified discoverability determines whether accelerated execution is governed, explainable, reusable, and safe for AI.

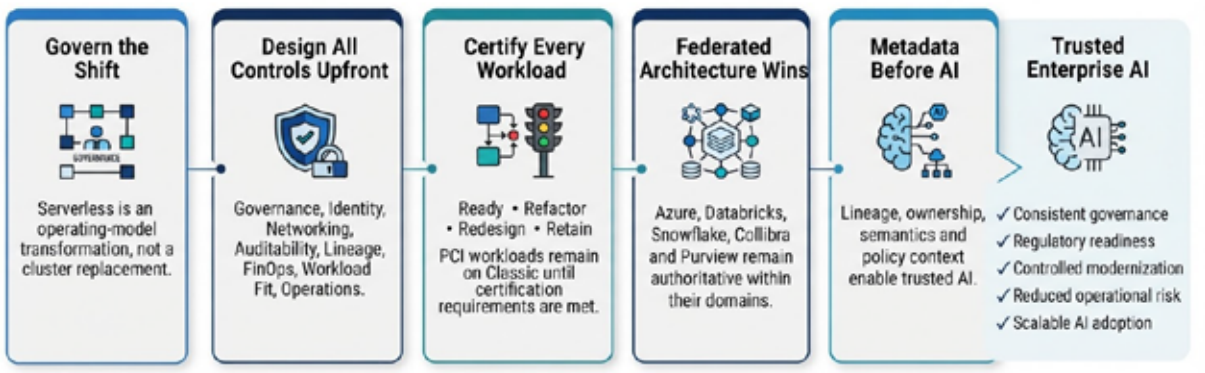
Databricks defines serverless compute as a managed service for notebooks, workflows, and Lakeflow pipelines where Databricks automatically allocates and manages compute resources.[1] In a regulated context, this is production-viable only when introduced through a control framework: Unity Catalog governance, identity federation, private connectivity, audit and system tables, cost attribution, workload certification, and clear operational runbooks.

For the Client, the question is not whether Serverless is technically ready. The question is whether governance, identity, metadata, and operating controls are mature enough to support Serverless at scale.

Our recommendation to the Client is direct: do not treat Serverless as a cluster replacement. Treat it as a governed capability that must be certified workload by workload, anchored in Unity Catalog governance and a federated enterprise metadata strategy.

Secure Databricks Serverless Adoption

5 Key Takeaways for Regulated Enterprises



2. Business Outcomes Before Technology

The business case for Databricks Serverless is not "cheaper compute." It is faster governed innovation. Traditional long-lived clusters create operational drag: provisioning delays, idle capacity, runtime drift, dependency sprawl, environment exceptions, and inconsistent tagging. Serverless shifts the operating model from "manage clusters" to "certify workloads, enforce policy, observe usage, and continuously optimise."

Target Outcomes for the Client

Faster governed delivery

Engineers and analysts start workloads in minutes, not hours, within Unity Catalog's permission, lineage, audit, and access-control boundaries.[1,5]

Lower platform toil

Platform teams shift from cluster mechanics to reusable patterns, controls, templates, and workload enablement.

Explicit production discipline

Workload certification makes the migration decision transparent: Ready, Refactor, Redesign, or Retain on classic compute. Documented serverless limitations make this classification mandatory.[2]

Improved cost accountability

Billing system tables attribute usage to resources, identities, products, and custom tags, enabling per-domain cost governance.[7]

Stronger AI readiness

Governed metadata, lineage, semantic definitions, data products, ownership, and access context become the substrate for safe copilots and agents. Without this layer, AI amplifies inconsistency

This framing aligns with regulated banking expectations. FFIEC's Architecture, Infrastructure, and Operations guidance calls for enterprise-wide, process-oriented governance of cloud computing, AI/ML, and zero-trust controls.[13] NIST SP 800-53 provides the underlying security and privacy control catalogue.[14]

3. Secure Databricks Serverless Production Pattern

Databricks Serverless for the Client should be designed as a single governed operating envelope — not a collection of independently configured components. The envelope covers identity, data governance, network connectivity, auditability, observability, FinOps, CI/CD, and workload certification. These controls must be designed together; retrofitting them individually after go-live is expensive and creates audit gaps.

Databricks states that serverless workloads are protected through dedicated ephemeral compute, private network segmentation with no public IP addresses, AES-256 encryption at rest, TLS 1.2+ encryption in transit, and short-lived one-hour tokens.[3] Control-plane-to-serverless-plane connectivity runs over the Microsoft cloud backbone, never the public internet.[4]

Production-Readiness Controls

The eight control areas below must all be addressed before any serverless workload reaches production. An evidence pack — signed by security, architecture, risk, compliance, operations, and FinOps — should accompany each certified workload.

Governance	Make Unity Catalog mandatory for all serverless workloads. Unity Catalog is the single governance layer for data and AI in Azure Databricks: access control, lineage, auditing, discovery, classification, quality monitoring, sharing, and AI governance.[5]
Identity	Use identity federation, group-based access, and service principals for production jobs. Individual grants are prohibited. Groups are provisioned via SCIM from Entra ID.[8]
Network	Design NCC, private endpoint, and service-tag firewall patterns before any production workload runs. NCCs are account-level regional constructs that create private endpoints from serverless compute to Azure resources.[4,17]
Auditability	Enable audit, billing, lineage, query history, and all relevant system tables from day one. System tables provide the tamper-resistant operational data required for security, risk, and FinOps governance.[7]
Lineage	Use Unity Catalog column-level lineage for impact analysis, root-cause investigation, sensitive-data flow tracking, and cross-team dependency visibility across all workspaces.[6]
FinOps	Use system.billing.usage, serverless usage policies, custom tags, budgets, and alerts. Attribute every dollar of cost to a workload, identity, product, and domain before go-live — not after.[7]

Workload Fit	Screen every workload against documented serverless limitations: unsupported R, Spark RDD APIs, limited DBFS access, UDF internet restrictions, unavailable Spark UI, unsupported compute policies, init scripts, cache limits, and a seven-day maximum runtime.[2]
Operations	Define runbooks, rollback and fallback procedures, exception handling, incident triage, query-profile analysis, release gates, and evidence packs before production cutover.

4. Workload Certification: The Essential Migration Discipline

The biggest risk in serverless migration is hidden workload dependency. Workloads that ran reliably on classic compute can fail under serverless because underlying assumptions break: Spark Connect semantics, DBFS usage, cross-session state, UDF behaviour, caching, streaming triggers, cluster-scoped configuration,

Every workload should pass through a certification factory and exit with a signed evidence pack for security, architecture, risk, compliance, operations, and FinOps review. The four certification outcomes are:

READY	SQL/Python workloads aligned to Unity Catalog, supported APIs, approved libraries, and a known cost profile. No refactoring required, certify and migrate.[1,2]
REFACTOR	Manageable changes: DBFS replacement, session-state cleanup, tag and identity updates, or CI/CD standardisation. Clear path forward with bounded effort.[2]
REDESIGN	Unsupported streaming triggers, internet-dependent UDFs, heavy custom libraries, or operational dependencies tied to cluster lifecycle. Requires architectural redesign before serverless migration.[2]
RETAIN ON CLASSIC	Requires unsupported APIs, runtimes exceeding the seven-day serverless maximum, tightly controlled custom environments, or PCI-DSS-scoped workloads until serverless achieves full PCI certification.[2]

Important: Azure Databricks currently restricts serverless compute access in PCI-DSS-compliant environments because serverless does not yet meet all PCI-DSS requirements.[2] Workloads within PCI scope must remain on classic compute until Databricks achieves full PCI-DSS certification for the serverless plane. This is the most critical risk-based scoping decision for the Client's initial migration.

5. Unified Discoverability Across Azure, Databricks, Snowflake, and Collibra

Forcing the Client's entire data estate into a single catalogue is the wrong goal. That approach creates friction, duplication, governance conflicts, and brittle integrations. The correct architecture is federated authority with unified meaning — each native platform remains the system of record for its domain, while a shared enterprise layer provides consistent meaning for all users and AI agents.

> **Native platforms remain authoritative. The enterprise layer provides unified meaning.**

Unity Catalog remains authoritative for Databricks access, lineage, and AI governance.[5] Snowflake Horizon provides semantic context, lineage, data quality, and AI guardrails within the Snowflake domain.[10] Collibra integrates Databricks Unity Catalog metadata — databases, schemas, tables, views, metric views, and columns — with sampling, profiling, classification, and lineage integration options.[11] Microsoft Purview scans Azure Databricks Unity Catalog technical metadata, tags, and lineage, plus Snowflake metadata and static lineage for tables, views, streams, stored procedures, and related objects.[12]

Layered Architecture

The unified discoverability architecture operates in five layers:

Users (Analysts, Data Engineers, Risk Teams, AI Agents) — all consumer types interact with one discovery surface, regardless of which platform holds the underlying data.	**AI Discovery Experience** Search, Marketplace, Lineage Explorer, Copilots, Access Requests, Agent Registry. This is the governed user-facing interface through which all data discovery occurs.	**Enterprise Knowledge Layer** Business Glossary, Semantic Models, Metric Definitions, Data Products, Knowledge Graph, Policy Context. Harmonises meaning across platforms and grounds AI agents in context.
Federated Metadata Signals Lineage, Ownership, Quality, Usage, Classification, Freshness, Trust Score, exchanged continuously between native platforms and the knowledge layer.	**Native Platforms** Azure/Purview, Databricks/Unity Catalog, Snowflake/Horizon, Collibra. Each retains full authority within its domain.	

Databricks Lakehouse Federation provides governed read-only access to external data through Unity Catalog foreign catalogues, including Snowflake.[9] This solves data access, not enterprise discoverability: Federation lets platforms reach each other's data; the enterprise knowledge layer makes that data meaningful, trustworthy, and safe for AI to act on.

6. Mphasis Experience in Large-Scale Data Platform Transformations

Mphasis brings Databricks and Snowflake specialisation, data architecture consulting, Unity Catalog upgrade experience, classic-to-serverless migration capability, DataOps/MLOps practices, and execution across North America and Europe. With 40+ active Databricks engagements across five industry verticals, including Banking and Insurance as a core sub-vertical, and a bench of 250+ Databricks-certified engineers, we have delivered the patterns described in this paper at production scale.

1 Global Insurance: Data Platform Stabilisation and Unity Catalog Modernisation

Challenge

Databricks estate instability: weak release governance, fragmented data models, high technical debt, and rising operational friction. Stabilisation was required before further modernisation was viable.

Approach:

Introduced disciplined Azure DevOps CI/CD, strengthened SDLC controls, consolidated redundant data models, stabilised Spark ETL jobs, updated documentation, and executed a Unity Catalog upgrade with workspace refactoring.

Business Impact:

- Reduced ticket volume by 40%
- Improved governance through Unity Catalog adoption
- Reduced cloud consumption through pipeline optimisation
- Lowered high-severity incident rate
- Restored engineering confidence in the platform

2 Life and Annuity Insurer: Governance, Lineage, and Data Quality Platform

Challenge

A standardised data platform for ingestion, canonical modelling, governance, security, quality, lineage, and business analytics publishing, built to satisfy regulatory and audit requirements from day one.

Approach:

Designed a Databricks/Delta Lake platform with medallion architecture, ACORD canonical modelling, automated source-to-gold lineage, data-quality rules automation, CloudWatch/Grafana observability, and curated

Business Impact:

✓ 80 pipelines, 65 source systems, 2,500 attributes, 70 entities, 75 business rules in production

Full governance with column-level lineage visibility

Automated data quality onboarding reduced configuration effort significantly

High-quality data products delivered to business stakeholders

3 Global Bank: Risk, Fraud, and Forecasting Modernisation

Challenge

Modernising legacy Hadoop and EDW workloads while preserving governed access, traceability, data quality, and regulatory discipline throughout the migration.

Challenge

Modernising legacy Hadoop and EDW workloads while preserving governed access, traceability, data quality, and regulatory discipline throughout the migration.

Approach

Implemented an ingest-once pattern, RBAC through Unity Catalog and Immuta integration, Databricks data-quality rules, Delta Lake streaming POCs, ML model endpoint patterns for fraud analytics, and migration of legacy Spark code for risk and profitability workloads.

Business Impact:

- Clear modernisation path for all legacy on-prem workloads
- Improved governed SDLC practices across engineering teams
- Accelerated time-to-value through DataOps/MLOps adoption
- Stronger production architecture for risk, fraud, forecasting, and analytics

4 Enterprise: Databricks and Snowflake Modernisation with MLOps

Challenge

Productionising ML workloads, reducing data-science friction, evaluating ML platform options, and integrating an existing Snowflake warehouse with Databricks.

Approach:

Enabled Databricks MLOps Stacks, standardised Dev/QA/Prod workspace practices, introduced Databricks Asset Bundles, strengthened CI/CD and SDLC practices, and integrated the Snowflake warehouse with

Business Impact:

- Improved engineering discipline and repeatability across ML
- Accelerated model operationalisation
- Demonstrated a proven pattern for Databricks and Snowflake coexistence
- Clear blueprint for governed, production-grade MLOps

MPHASISFOLLOW-UP

Three Technical Questions — and Our Answer

Following the initial discussion, the Client team raised three specific technical questions. Each is addressed in full below, with additional detail and BFSI-specific context that extends the analysis in the position paper.

7. Serverless Networking in Highly Secure Environments

> **Question:** How does Azure Databricks Serverless networking work, and how should it be configured in a Tier 1 regulated bank?

The Network Model Shift

With Databricks Serverless, network controls move from VNet primitives (NSGs, VNet injection, back-end Private Link) to Databricks-managed constructs.[4] This is not a simplification; it is a re-authoring of the network control catalogue. The Client must plan this migration explicitly rather than port existing classic cluster network policies across.

The serverless compute plane provides the following security properties by default:[3]

No public IP addresses	all compute runs on Databricks-managed infrastructure with private network segmentation.
Dedicated ephemeral compute	resources are wiped between sessions, eliminating cross-workload data residency risk.

AES-256 at rest, TLS 1.2+ in transit	all data and control traffic is encrypted end-to-end.
Short-lived one-hour tokens	credential lifetime is bounded, limiting blast radius on credential compromise.
Cloud backbone routing	control-plane-to-serverless-plane traffic stays on the Microsoft cloud backbone and never traverses the public internet.

Three Controls Required for BFSI Production

The following three controls must be in place before any BFSI production workload runs on Databricks Server-less. Deploying them after go-live creates a period of uncontrolled exposure that is inconsistent with a regulated bank's risk posture.

Control	Mechanism	Implementation Guidance
Private Inbound Connectivity	Network Connectivity Configurations (NCCs)	Account-level regional constructs that create Databricks-managed privat endpoints to Azure resources. Azure Private Link is GA across 60+ service (ADLS, Azure SQL, Azure OpenAI). Limits: up to 10 NCCs per region, each attachable to 50 workspaces and 100 private endpoints. Premium plan and account-admin rights required.[4,17]

Control	Mechanism	Implementation Guidance
Egress / Data Exfiltration Control	Serverless Network Policies	Deny-by-default posture: outbound traffi permitted only to Unity Catalog external locations and explicitly listed FQDNs or storage targets. Dry-run mode validates the policy against the network-access events system table before enforcemen —a full business cycle validation prior t go-live.[16]
Firewall / Perimeter	Azure NSP + AzureDatabricksServerless Service Tag	Storage accounts that currently allowlis serverless subnet IDs must be onboarde to Azure Network Security Perimeter. The AzureDatabricksServerless service tag must be explicitly allowlisted. Reference architecture: context-based ingress service endpoints → serverless egress control → NCC private endpoints

Hardened Connectivity Reference Architecture

Databricks' Hardened Connectivity reference architecture is the recommended pattern for regulated banks. The stack is: context-based ingress → service endpoints → serverless egress control → NCC private endpoints → optional external firewall for perimeter inspection.[4,16] Mphasis implements this architecture in full as part of our Databricks partnership alignment.

Note: The Serverless Workspace product is currently in Public Preview and front-end private connectivity configuration varies per cloud region.[4] The exact configuration for the Client's target Azure region must be confirmed with the Databricks account team before production planning commences.

Data Sovereignty and the Regulatory Timeline

Serverless network architecture cannot be considered in isolation from the legislative landscape governing customer data sovereignty. The following regulations are directly relevant to the Client's architecture decisions and are at various stages of enforcement:

Effective Date	Legislation	Key Requirement for BFSI
January 2025	Digital Operational Resilience Act (DORA)	Mandates exit strategies and concentration risk management for cloud service dependencies.
September 2025– January 2027	EU Data Act	Enforces data portability and removes switching barriers across cloud providers.
August 2026	EU AI Act	Strict AI governance, data lineage requirements, and model risk controls for financial institutions.
June 2026 (Proposal)	Cloud and AI Development Act (CADA)	Explicit sovereignty tiers and heightened data residency and processing transparency requirements

The combined requirements of DORA, the EU Data Act, the EU AI Act, and CADA make clear that network architecture, governance model, and metadata lineage fabric must be designed together — not sequentially. An architecture that satisfies today's requirements but cannot satisfy CADA sovereignty tiers creates technical debt before the system goes live.

8. BFSI Industry Adoption of Azure Databricks Serverless

Question: What is the current state of Azure Databricks Serverless adoption across the BFSI industry, and what patterns has Mphasis observed?

Mphasis BFSI Databricks Footprint

Mphasis has 40+ active Databricks engagements across five industry verticals, with Banking and Insurance as a core sub-vertical. Our internal "One Thousand Bricks" multi-year programme is on track to grow our Data-bricks-certified headcount further, currently at 250+ practitioners.

Why BFSI Serverless Adoption Was Historically Blocked

BFSI serverless adoption was gated on one specific capability: private connectivity to storage. Before Network Connectivity Configurations (NCCs) became available, there was no mechanism for serverless compute to reach privately networked storage with public access disabled.[17] As one BFSI architecture review noted, this limitation "ruled out serverless for the majority of production workloads in financial services." Any bank operating a zero-trust, deny-by-default storage posture faced an absolute technical blocker.

The Current Inflection Point

With NCC private endpoints[17] and egress control[16] now generally available or in preview, that blocker is largely removed. BFSI adoption is accelerating for governed, Unity Catalog-first estates. The pattern Mphasis observes across clients is a staged approach based on workload risk profile:

1. Short-lived analytics and ML inference workloads migrate first — aligned naturally to serverless economics with no streaming or PCI constraints.
2. ETL and data product pipelines follow once NCC connectivity to ADLS and Azure SQL is validated in the target region.[4,17]
3. Risk, fraud, and regulatory reporting workloads are assessed in parallel but remain subject to the PCI DSS restriction until Databricks achieves full certification.[2]
4. Legacy Hadoop and EDW migration workloads are typically redesigned rather than lifted to serverless, due to Spark RDD and streaming dependencies.[2]

The Serverless-First Platform Direction

An important strategic signal is that Databricks is consistently launching new capabilities as serverless first. This means the governance and control investment made today is not just for current workloads — it positions the Client for the next generation of Databricks capabilities:

1. **Lakebase** — the serverless transactional database product built on the Neon acquisition. BFSI clients migrating away from Oracle are using Lakebase as the target for highly transactional workloads, a pattern directly relevant to the Client's estate.
2. **LTAP** (Lake Transactional/Analytical Processing) — the evolution of Lakebase, combining transactional and analytical processing within a single serverless lakehouse. This capability is generating significant interest across BFSI and adjacent verticals.

The practical implication for the Client: the governance and control framework for serverless, built correctly today, is the same foundation that enables Lakebase and LTAP adoption. Deferring the governance investment defers the full platform benefit.

9. Identity and RBAC/ABAC Management in BFSI

> **Question:** How should identity be managed in Azure Databricks for a regulated bank, and what is the recommended approach to RBAC and ABAC at scale?

Identity Foundation

Unity Catalog is mandatory for serverless compute[5], and all identity management flows through it. The four non-negotiable foundations are:[8]

1

Identity federation via Entra ID

all human identities federated through Microsoft Entra ID; no local Databricks users in production

2

Group-based access exclusively

access policies granted to Entra ID groups only, never to individual users. Groups are provisioned and synchronised via SCIM.

3

Service principals for all production jobs

no automated workload uses human credentials. Service principals are the only supported identity type for production pipelines and scheduled jobs.

4

Zero individual grants in production

this is a hard control enforced through governance policy, not convention. Any deviation creates an unauditable access path.

RBAC and ABAC: The Layered Model

The access control architecture uses two complementary layers. RBAC establishes the baseline "who" — which groups hold which privileges at which level of the Unity Catalog hierarchy. ABAC adds the dynamic "what" — it uses governed tags to enforce fine-grained row-level and column-level policies at query time, directly solving the role explosion problem that RBAC hits at bank scale.[18]

Dimension	RBAC — Baseline Layer	ABAC — Scaling Layer
Core question	Who has access?	What can they see, given context?
Mechanism	Hierarchical ANSI SQL privileges (USE CATALOG → USE SCHEMA → SELECT) granted to Entra ID groups	Governed tags (sensitivity=high, region=EMEA) drive row filters and column masks evaluated at query time
Provisioning	SCIM sync from Entra ID; catalogs = business domains; schemas = medallion layers	Tag-inherited policies propagate down catalog/schema/table hierarchy — no per-grant management
Scale behaviour	Role explosion at bank scale — thousands of roles become unmanageable	Centrally defined policies scale linearly — policy count stays constant as the estate grows
Status on Azure	GA — mandatory baseline for all serverless workloads	Beta (DBR 16.4+ / serverless required) — strategic direction; use RBAC + table-level masks as interim
BFSI application	Group-based grants, service principals for pipelines, zero individual grants in production	PII tagging via data discovery tool → UC tags → ABAC masking policies replacing UDFs

Recommended Catalogue Structure

Standardising the Unity Catalog structure from the outset prevents governance debt:[5]

1. Catalogues map to business domains — risk_domain, payments_domain, fraud_domain, etc. Schemas map to medallion layers within each domain — bronze, silver, gold, published.

2. Ownership and MANAGE privileges are delegated to domain data owners, enabling decentralised stewardship within a centrally governed metastore.

ABAC in Practice: BFSI Proof Points

Mphasis has applied this identity and access model in regulated BFSI environments. The most transferable pattern for the Client is UDF-to-ABAC migration: existing masking UDFs are replaced with tag-driven column mask policies aligned to classifications from the organisation's data discovery tool.[18] A BigID classification of "PII" maps to a Unity Catalog tag of sensitivity=high, which triggers a centrally defined masking policy that evaluates at query time across all tagged tables, eliminating per-table UDF maintenance and the risk of UDF drift across the estate.

ABAC Caveats and the Interim Approach

> **Current limitations:** ABAC is Beta on Azure and requires DBR 16.4+ or serverless compute.[18] ABAC policies cannot currently be attached directly to views — however, policies on underlying tables are enforced when data is accessed through a view, preserving the protection boundary.

The interim approach is RBAC as the enforced baseline, with table-level column masks applied where fine grained control is required today.[8] ABAC policies should be designed and staged in parallel, ready to activate as the capability reaches GA on Azure. This phased approach allows the Client to proceed with serverless adoption immediately, with a clear, documented path to ABAC-based scaling.

10. Guidance to the Executive Committee

Six principles frame the decision for the Client leadership. Each represents a strategic commitment, not a technical option.

11. Recommended Adoption Roadmap

Seven sequential steps move the Client from readiness assessment to governed production. Each step has an explicit gate; no step proceeds until the previous step's evidence pack is approved.

Run a Serverless Production-Readiness a

Inventory all workspaces, workloads, identities, network paths, data-access patterns, library dependencies, lineage coverage, system tables, and FinOps instrumentation. Produce a workload dependency map and identify the initial certification candidate pool. The output of this step determines the scope and sequence of every subsequent step.

Design the Secure Serverless Landing Zone

Define Unity Catalog structure, identity federation, service-principal patterns, NCC and private connectivity, audit and system tables, logging, cost policies, and deployment templates. Include egress control policy design with dry-run validation against the network-access events system table before enforcement is activated.

Execute a Controlled Pilot with a Signed Evidence Pack

Select representative workloads from the Ready or Refactor tier. Benchmark performance and cost against classic compute. Validate lineage, test rollback, confirm all network paths, and document security and risk approvals in a signed evidence pack. No workload should reach production without this approval.

Stand Up the Workload Certification Factory

Classify all in-scope workloads as Ready, Refactor, Redesign, or Retain on Classic. Build DAB and Terraform templates for each tier. Enforce CI/CD gates that prevent non-certified workloads from reaching the production serverless environment.

Implement Federated Discoverability

Connect Unity Catalog, Snowflake Horizon, Collibra, Purview, and enterprise semantic and knowledge layers into a unified discovery experience. Define the governance model for the enterprise knowledge layer — including glossary ownership, metric definition authority, and trust score methodology.

Govern AI Agents Through Metadata

Agent registries, data-product contracts, policy context, lineage, access controls, and semantic definitions must govern what agents can discover, recommend, and execute. This layer must be operational before any generative AI capability is enabled over the Client data estate.

Continuously Optimise

Use system tables, usage policies, dashboards, budget alerts, data-quality monitoring, and access reviews to operate serverless as a continuously managed production capability. Cost and quality signals should feed back into the same governance operating model that governs access and lineage.

12. Anticipated Questions from the Client Leadership

Q Which workloads should move first?	Short-lived SQL/Python notebooks, jobs, and ETL/ELT workloads already aligned to Unity Catalog, supported APIs, approved libraries, and clear cost tags.[1,5] Avoid workloads with unsupported APIs, complex streaming triggers, internet-dependent UDFs, cache-heavy logic, or runtimes exceeding seven days until they have been redesigned.[2] PCI-scoped workloads must remain on classic compute until Databricks achieves full PCI-DSS certification for the serverless plane.
Q Does serverless reduce our control?	It changes the control point, not the level of control. The Client retains full authority over identity, Unity Catalog permissions,[5,8] network connectivity configuration,[4,17] audit evidence, cost attribution, workload eligibility gates, and operational runbooks. The control model shifts from infrastructure-level (VNet, NSG) to policy-level (NCC, UC permissions, egress policies) — a shift that is more auditable and more aligned to zero-trust architecture principles.
Q How do we prove auditability to regulators?	Enable Unity Catalog audit, lineage, billing, query history, and all relevant system tables from day one,[5,6,7] then map those outputs to internal security, risk, compliance, FinOps, and governance evidence processes. System tables provide tamper-resistant, queryable evidence. The workload certification factory evidence pack documents the security and risk approvals for every production workload — creating a continuously maintained audit record.
Q Should Purview replace Collibra or Unity Catalog?	No. The correct framing is role clarity, not replacement. Unity Catalog governs Databricks data and AI assets.[5] Snowflake Horizon governs the Snowflake estate.[10] Collibra supports enterprise data stewardship, governance workflow, and business glossary management across all platforms.[11] Purview contributes Microsoft ecosystem scanning and integration.[12] Each tool has authority in its domain; the enterprise knowledge layer provides the unified meaning layer across all of them.
Q How do we control serverless cost?	Benchmark representative workloads against classic compute before migration. Require serverless usage policies and custom tags on every workload before it goes live. Use system.billing.usage to monitor consumption by workload, identity, product, and domain.[7] Set budgets and alerts per domain. Serverless can surface unexpected cost at scale if workloads are not properly tagged and attributed before production deployment — not after.
Q How do we make the data estate AI-ready?	Build the governed enterprise knowledge layer: business glossary, semantic models, data products, lineage, ownership, quality, classification, freshness, policy context, and trust signals.[5,6,10,11,12] Without this layer, AI discovery becomes a faster mechanism for exposing inconsistent meaning — compounding existing data quality problems rather than resolving them. The unified discoverability architecture in Section 5 is the prerequisite for any responsible AI agent deployment over the Client data estate.

13. Closing: Governed Innovation and AI Readiness

The right question for the Client to ask is not "Is Databricks Serverless ready for production?" The platform capability is demonstrably ready. The right question is:

"Are we ready to operate Serverless as a governed production capability?"

The answer is yes: when serverless adoption is tied to Unity Catalog governance, identity federation, NCC-based private connectivity, audit and system tables, FinOps dashboards, workload certification, and a federated metadata fabric connecting Azure, Databricks, Snowflake, Collibra, and Purview into a single business-facing understanding layer.

Serverless removes infrastructure friction. Unified discoverability removes data ambiguity. Together, they create the foundation for governed AI at scale. The regulatory trajectory, including DORA, the EU AI Act, and CADA, reinforces this investment: organisations that build the metadata and governance foundation today will absorb tomorrow's regulatory requirements without re-architecting their estate.

Mphasis is ready to support the Client through the readiness assessment, landing zone design, controlled pilot, and workload certification factory described in this paper.

Faster governed innovation, not just faster compute.

References

1. Microsoft Learn — Connect to serverless compute, Azure Databricks
2. Microsoft Learn — Serverless compute limitations, Azure Databricks Microsoft Learn — Serverless compute limitations, Azure Databricks
3. Databricks Trust Center — Deploy Your Workloads Safely on Serverless Compute
4. Microsoft Learn — Serverless compute plane networking, Azure Databricks
5. Microsoft Learn — What is Unity Catalog? Azure Databricks
6. Microsoft Learn — Lineage in Unity Catalog, Azure Databricks
7. Microsoft Learn — System tables, cost monitoring, and serverless billing, Azure Databricks
8. Microsoft Learn — Identity best practices, Azure Databricks
9. Microsoft Learn — Lakehouse Federation and Snowflake federation, Azure Databricks

10 Snowflake Documentation — Snowflake Horizon Catalog

11 Collibra Documentation — Databricks Unity Catalog integration via Edge

12 Microsoft Learn — Microsoft Purview connectors for Azure Databricks Unity Catalog and Snowflake

13 FDIC / FFIEC — Architecture, Infrastructure, and Operations Booklet

14 NIST SP 800-53 Rev. 5 — Security and Privacy Controls for Information Systems and Organisations

15 Cloud Security Alliance — Cloud Controls Matrix / CAIQ

16 Microsoft Learn — Azure Databricks serverless network policies (egress control)

17 Microsoft Learn — Network Connectivity Configurations (NCC), Azure Databricks

18 Microsoft Learn — Unity Catalog ABAC: row filters and column masks