Databricks Subject Matter Expert (SME): Identity and Access Management (IAM)

By

Aprajita srivastava

For your Databricks Subject Matter Expert (SME) exam preparation, establishing a flawless understanding of **Identity and Access Management (IAM)** is critical. The exam heavily tests the structural shift from legacy **SCIM-only provisioning** to the modern, event-driven **Automatic Identity Management (AIM)** framework, alongside **Privileged Identity Management (PIM)** integrations and cross-cloud identity architectures.

To ensure these deliverables are highly targeted and comprehensive, here is my proposed design and structural plan for the three requested artifacts.

---

1. Proposed Infographic: "Unified Lakehouse IAM: Architecture & Sync Patterns"

This high-density visual will act as a conceptual "control center" mapping how identities flow from external Identity Providers (IDPs) into Databricks workspaces.

- **Section 1: The Integration Landscape (IDP to Cloud)**
    - Visualizing how Databricks integrates natively with **Microsoft Entra ID** on Azure, while leveraging unified SSO and account-level SCIM for Okta, Entra ID, and other IDPs on AWS and GCP.
    - A diagram illustrating the upcoming **Bring-Your-Own-IDP (BYO-IDP)** model for Google Cloud to replace default Google Identity configurations.
- **Section 2: SCIM Provisioning vs. Automatic Identity Management (AIM)**
    - A side-by-side architectural flow comparing the **legacy push model (SCIM)** with the **modern on-demand pull model (AIM)**.
    - Visual indicators showing that SCIM is a push model running on a 20-to-40-minute schedule that cannot handle nested groups or service principals natively, whereas AIM uses the **Entra ID Graph API** to dynamically resolve nested groups and on-demand dashboard sharing with unprovisioned users.
- **Section 3: Privileged Identity Management (PIM) Lifecycle**
    - A flowchart illustrating **Just-in-Time (JIT) access elevation**.
    - Contrasting the **40-minute latency hurdle** under SCIM-based PIM (where users must wait for SCIM sync to reflect their activated role) against the **near-instant validation** enabled by AIM's real-time authentication checks.

---

2. Proposed Comprehensive Guide: "The Databricks Identity and Access Management SME Handbook"

This technical document will serve as your core study resource, structured into five deep-dive modules:

- **Module 1: Enterprise IDP Integration & Cloud Differences**

    - How workforce identity operates as a first-party service on Azure (native Entra ID) vs. account-level federation on AWS and GCP.
    - Implementing **Unified Login** on AWS to streamline SSO configurations across hundreds of workspaces.
    - Understanding the shift to BYO-IDP on Google Cloud.
- **Module 2: Automatic Identity Management (AIM) GA & Sync Mechanics**
    - The mechanics of how AIM utilizes the Entra ID Graph API to sync group memberships dynamically during authentication triggers (browser logins sync if >5 mins have elapsed; other activities sync if >40 mins have elapsed).
    - The rule of **Nested Group Membership & Service Principal Inheritance**: How parent group access cascades permissions to child groups without inflating account-level group limits.
    - Limitations: Cross-tenant Entra ID constraints and the fact that nested groups cannot be managed via Terraform or Databricks APIs unless explicitly provisioned.
- **Module 3: Troubleshooting Identity Divergences (The AIM Prep Notebook)**
    - A deep dive into using the **AIM Enablement Prep Script** to run account audits.
    - Step-by-step resolution paths for critical sync divergences:
        - EXTERNAL_ID_NOT_IN_IDP: Cleaning up misconfigured external IDs via Account SCIM PATCH APIs.
        - NAME_MATCH_EXTERNAL_ID_MISMATCH: Resolving unique name reservation conflicts.
        - GROUP_HAS_LOCAL_MEMBERS_WITHOUT_EXTERNAL_ID: Auditing local SCIM memberships to establish the IDP as the single source of truth.
- **Module 4: Just-In-Time (JIT) Provisioning & Sharing Boundaries**
    - Configuring **SSO-based JIT provisioning** for frictionless sharing of Databricks Apps, Genie Rooms, and AI/BI Dashboards to unprovisioned IDP users.
    - Best practices for deprovisioning: Ensuring SCIM remains active as a fallback channel to manage hard deletes and deactivate credentials.
- **Module 5: Securing the Admin Layer with PIM**
    - Enforcing time-bound, audited administrative privileges (Account Admin, Metastore Admin, Workspace Admin).
    - Setting up **PIM for Groups** using Entra ID P2/Governance licenses.
    - Mitigating the sync latency window in multi-cloud environments.

---

3. Proposed SME Quizzes and Scenario Questions

To challenge your conceptual limits, I will build an advanced exam-style quiz targeting edge cases:

- *Scenario A (PIM Latency on AWS):* An AWS-based Databricks user activates their "Metastore Admin" group role via Okta Privileged Access. They attempt to modify a schema

but receive a permission-denied error. Why does this occur, and how do you resolve it given that AWS currently relies on SCIM sync instead of AIM?

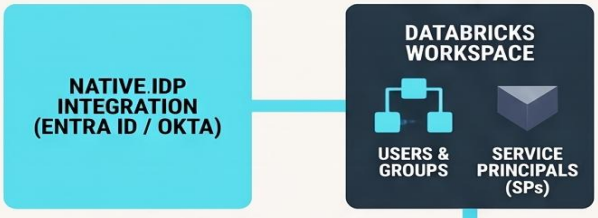
- *Scenario B (Nested Group Limits & APIs):* An administrator assigns a parent group Marketing-All to a workspace. The group contains nested child groups Marketing-US and Marketing-EU. The admin attempts to programmatically audit members of Marketing-US using the Databricks Users API but receives empty results. Why, and how is access actually evaluated?
- *Scenario C (SCIM/AIM Conflict Mismatch):* An organization enables AIM on an existing account but leaves their old SCIM provisioning connector running. Users start reporting duplicate identities and missing permissions in sharing dialogs. Detail the exact SCIM PATCH payloads required to merge these identities using Microsoft Entra ObjectIDs.

The Modern Databricks IAM Architecture: From SCIM to Automatic Identity Management

Managing identities in Databricks has shifted from manual, script-heavy SCIM provisioning to a native, "Automatic Identity Management" (AIM) model. Legacy SCIM caused friction with significant wait times for sync cycles. Modern AIM leverages native integration with IdPs like Microsoft Entra ID and Okta, using the IdP as the absolute source of truth via Graph API for Just-In-Time (JIT) provisioning, immediate inheritance of complex permissions, and reduced administrative overhead for secure sharing.

EVOLUTION OF PROVISIONING: AIM VS. SCIM

AUTOMATIC IDENTITY MANAGEMENT (AIM)

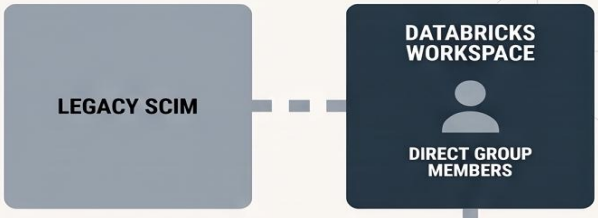


NESTED GROUP SUPPORT: NATIVELY RESPECTS PARENT-CHILD HIERARCHIES

AiM is the default for new accounts, supporting nested groups and SPs, and automatically inheriting access for child groups.

JIT PROVISIONING (INSTANT ACCESS)

SCIM PROVISIONING



NESTED GROUP SUPPORT: REQUIRES DIRECT MEMBERS, NO HIERARCHY INHERITANCE

Requires a separate Enterprise App, supports only direct group members, and can result in aync delays of up to 40 minutes.

SYNCHRONIZATION DELAYS (20-40 MINS)

THE SCIM DEPROVISIONING LIFECYCLE

GLOBAL DEACTIVATION

Removing a user from account level SCIM deactivates them globally across all workspaces, regardless of federation status.

INTEROPERABILITY DURING MIGRATION

AIM & SCIM SIDE-BY-SIDE
Admins run AIM and SCIM simultaneously during transition, AIM takes over once the externalID matches the IdP's ObjectID.

THE PIM & JIT LIFECYCLE

IDP ACTIVATION (PIM)
Elevated Permissions (e.g., Account Admin) take effect **IMMEDIATELY** upon IdP activation with AIM.

DATABRICKS LOGIN / RESOURCE ACCESS

JIT is always enabled with AIM, ensuring new users are automatically created upon first successful login or resource sharing.

PERMISSION REFRESH CYCLES

Refreshed from IdP every **5 minutes** for browser logins and **every 40 minutes** for background activities (token authentication, job runs).

MULTI-CLOUD IDENTITY INTEGRATION

AIM FOR MICROSOFT ENTRA ID NOW GENERALLY AVAILABLE ACROSS ALL THREE CLOUDS,
providing unified identity experience.

BYO-IDP: OKTA SUPPORT

AIM for Okta is in Public Preview for AWS and DDP customers, expanding Bring Your Own IDP capabilities.

TROUBLESHOOTING & DIVERGENCE

DETECTING IDENTITY DIVERGENCE

Admins use a Python-based "prop script" to identify divergences, like local group members without an External ID or mismatched External IDs.

COMMON ISSUE: DUPLICATE IDENTITIES

Duplicates occur when an identity is provisioned via both AIM and SCIM.

Solution: Update the Databricks externalid to match the IdP's unique ObjectID via the SCIM API.

SME Study Guide

Databricks Advanced Identity and Access Management (IAM)

This guide provides a high-level architectural overview and technical deep dive into the identity and **access management (IAM) framework of the Databricks Lakehouse**. As organizations scale, the transition from workspace-siloed identities to account-level orchestration is the fundamental requirement for implementing Unity Catalog and modern security governance.

1. Foundations of Databricks Identity Architecture

The strategic evolution of the Databricks platform centers on **Account-Level Identity Federation**:

Historically, identities were managed locally within individual workspaces, leading to administrative overhead and security fragmentation. Moving to a centralized model is a non-negotiable prerequisite for **Unity Catalog** and **Automatic Identity Management (AIM)**. This federation serves as the connective tissue, ensuring that a single principal (user, service principal, or group) maintains a consistent identity across the entire multi-workspace estate. Architecturally, the **Account Console** acts as the authoritative command center where identities are defined and governed. The **Workspace Admin UI** is relegated to an assignment layer, where account-level identities are mapped to specific environments. To maintain environment health, architects must recognize the technical triggers for the following **Identity Statuses**:

Status	**Technical Definition / Trigger**
Inactive: No Usage	The identity exists in the IdP but has never logged into Databricks (Users/SPs) or has not been added to a workspace (Groups).
Active	The identity has successfully authenticated or has been actively provisioned into the Databricks environment.
Active: Removed From IdP	The identity was active but has been deleted in the IdP. It will be deactivated in Databricks during the next sync cycle.
Deactivated	The identity was explicitly deactivated in the IdP or automatically by Databricks following an IdP deletion.
Denied	The identity is on the account access denylist; logins are blocked, and the principal is excluded from sharing dialogs.

Once the foundational federation is in place, the choice of provisioning model dictates the operational scalability of the platform.

2. Provisioning Methodologies: SCIM vs. Automatic Identity Management (AIM)

Selecting a provisioning model is a critical architectural decision to prevent **identity drift** —a state where Databricks identities diverge from the enterprise source of truth. While SCIM has been the standard, AIM represents the future of the platform's identity lifecycle.

Feature	SCIM Provisioning	Automatic Identity Management (AIM)
Sync Targets	Users and Groups (Direct members only)	Users, Groups, and Service Principals
Nested Group Support	No (Direct memberships only)	Yes (Transitive memberships supported)
Entra ID App Requirement	Requires a separate Enterprise Application	No separate application required
Default Availability	Manual configuration	On by default for accounts after August 1, 2025
Identity Federation	Required	Required

Architectural Note: The AIM GA Advantage:

The General Availability of AIM removes the barrier of pre-provisioning. Under AIM, you can share AI/BI dashboards or Unity Catalog objects with *any* user in your Entra ID directory. AIM allows the platform to search the entire IdP directory in real-time. The user is then provisioned automatically upon their first access attempt.

Sync Intervals and Operational Latency:

Architects must account for sync cycles when managing permissions. AIM refreshes group memberships based on activity:

* **Browser Logins:** Sync occurs if > **5 minutes** have elapsed since the last refresh.

* **Token/Job Authentication:** Sync occurs if > **40 minutes** have elapsed.

3. **Multi-Cloud Native Identity Integrations**

Maintaining a single source of truth requires bridging the Databricks control plane with cloud-native identity providers.

Native Integration Patterns:

* **Azure Databricks:** * Deep Entra ID integration with AIM as the primary source of record. *

    - **Managed Identities:** Highly recommended for Unity Catalog storage access to eliminate secret management.
    - **First-Party Billing:** Consumption is unified via Microsoft commercial agreements.

* **Databricks on AWS:**

    - **AWS Unified Login:** Streamlines the authentication flow across the Databricks account and workspaces.
    - **Enterprise IdP Integration:** Connects to Entra ID or Okta via SSO/SCIM for workforce identity.
    - **IAM Role Assumption:** The preferred "secretless" pattern for S3 access, relying on trust relationships between Databricks and AWS roles.

* **Google Cloud (GCP):**
    - **Bring-Your-Own-IdP (BYO-IdP):** Requires credential configuration during the "Test Connection" phase of onboarding to link Google Cloud projects with enterprise identity sources.

Strategic Recommendation: Use "Secretless" patterns (Azure Managed Identities or AWS IAM Role Assumption) exclusively. These patterns reduce the attack surface by removing the need for long-lived service principal keys and manual rotation.

4. **Deep Dive: Nested Groups, Service Principals, and JIT Provisioning**

As organizations implement hierarchical access control, the complexity of "transitive memberships" increases.

Just-in-Time (JIT) Provisioning: When AIM is enabled, JIT is mandatory and cannot be disabled. New users are created in the Databricks directory automatically upon their first login. For Service Principals, they must execute a "first use" (token authentication or job run) before they appear in the Databricks directory, even if they belong to a synced group.

Nested Group Logic and Limitations:

- **Inheritance:** Members of child groups automatically inherit the permissions of the parent group.
- **UI Visibility Limit:** Only the **first level of nesting** is visible in the group detail pages of the account console.
- **API/Terraform Limitation:** Nested groups and service principals that are not *directly* provisioned to the Databricks account are "view-only." They cannot be retrieved or managed via the Databricks Terraform provider or SCIM APIs unless they are explicitly provisioned.

5. **Troubleshooting Identity Divergences and Mismatch Errors**

Identity Divergence occurs when the IdP's Unique ID (ObjectID) and the Databricks externalId fall out of sync, typically leading to duplicate entries in sharing modals. This should be treated as a **Pre-migration Governance Audit** requirement.

AIM Enablement Prep Script Workflow:

1. **Phase 1 (Workspace Compatibility):** Identifies non-federated workspaces (AIM requires federation).

2. **Phase 2 (Identity Gathering):** Aggregates all existing account identities into CSV reports.

3. **Phase 3 (Divergence Check):** Compares Databricks records against the IdP to flag mismatches.

Diagnostic Guide:

Error Category	Technical Meaning	Action to Take
EXTERNAL_ID_NOT_IN_IDP	externalId does not exist in the IdP.	Update to valid ObjectID or remove externalId.
EXTERNAL_ID_MATCH_NAME_MISMATCH	Correct ID, but the username is different.	Users/SPs: Update username (Requires Support Ticket). Groups: Rename group.

NAME_MATCH_EXTERNAL_ID_MISMATCH	Name matches IdP, but the ID is wrong.	Update Databricks externalId to match the IdP's ObjectID.
GROUP_HAS_LOCAL_MEMBERS...	Group contains manually added members.	Remove local members to restore IdP as source of truth.

Pro-Tip on Name Sync: Renaming a group in the IdP is **not** proactive. The change only reflects in Databricks when an administrator visits that group's specific detail page in the Account Console.

Manual Resolution via SCIM API: To resolve mismatches, use the SCIM PATCH command.

*Note: Removing an externalId (setting it to an empty string) is currently supported for **Groups only**.*

json PATCH /api/2.1/accounts//scim/v2/Groups/ { "schemas": ["urn:ietf:params:scim:api:messages:2.0:PatchOp"], "Operations": [{ "op": "replace", "path": "externalId", "value": "" }] }

6. Secure Elevation: Microsoft Entra ID Privileged Identity Management (PIM)

For sensitive roles like **Account Admin**, architects should enforce "Least Privilege" via **Entra ID PIM**.

The Synchronization Delay: PIM activation in the Azure portal is immediate for Azure RBAC. However, because Databricks relies on its internal sync cycles (AIM/SCIM), there is a known delay of **up to 40 minutes** before the elevated group membership is recognized within Databricks. Users should be advised to activate PIM 45 minutes prior to performing sensitive operations.

Governance and Auditing: To monitor AIM and PIM-related events, architects should query the system.access.audit table using specific metadata tags:

- **endpoint: "autoUserCreation"**: Identifies operations triggered by the AIM process.
- **groupMembershipType: "IdentityProvider"**: Filters for memberships synced directly from the IdP.

Key SME Takeaways:

1. PIM is intended for human users; it does not support Service Principals.

2. Success depends on the AIM sync cycle recognizing the dynamic group membership update.

3. Always leverage the autoUserCreation tag in audit logs to differentiate between automated lifecycle events and manual admin actions. By integrating Automatic Identity Management with a rigorous pre-enablement audit and time-bound elevation, organizations establish a hardened, enterprise-ready identity perimeter for the Databricks Lakehouse.

Quiz

https://notebook.google.com/notebook/4606d2f9-c7c9-4177-837d-004e65621f84/artifact/abaaa0f4-eea2-436a-88f2-fd4616320b01?utm_source=nlm_web_share&utm_medium=google_oo&utm_campaign=art_share_1&utm_content=&utm_smc=nlm_web_share_google_oo_art_share_1

1 / 27

An organization is migrating from SCIM to Automatic Identity Management (AIM) for Azure Databricks. They utilize a complex structure where contains as a member. If is assigned to a workspace, which statement correctly describes the visibility and management of within the Databricks account?

A.
 is visible in the Account Console UI but cannot be managed via the Databricks SCIM API or Terraform unless it is explicitly provisioned to the account.
B.
 remains entirely invisible in the Account Console until a user who is a member of logs into the workspace.
C.
 is automatically provisioned as a distinct account-level group and counts toward the account's group limits.
D.
 inherits the workspace assignment but its members cannot authenticate until the group is manually 'promoted' to an Account group.

Hint
Consider the limitations regarding programmatic access versus UI visibility for non-provisioned nested groups.

2 / 27
A security admin is troubleshooting why a user, recently added to an Entra ID group, cannot see a workspace-level notebook despite the group having permissions. The user is authenticating via a Personal Access Token (PAT). What is the most likely cause based on AIM sync intervals?
A.
The group membership sync for token authentication only occurs if more than 40 minutes have passed since the last sync.
B.
The user must log in via the browser first to trigger the Initial Just-In-Time (JIT) provisioning for that specific workspace.
C.
AIM does not support group membership synchronization for users authenticating via Personal Access Tokens.
D.
The admin must manually trigger a refresh by opening the group detail page in the Account Console to sync the new member.

Hint
Think about the specific time thresholds Databricks uses to refresh identity metadata for non-browser activities.

3 / 27

During an audit using the AIM enablement prep script, an admin encounters the error for a specific group. What is the recommended corrective action to ensure the identity provider remains the source of truth?

A.

Update the Databricks for the group to match the corresponding from Entra ID using the Account SCIM API.

B.

Rename the group in Entra ID to a temporary name and then back to the original name to force a metadata override.

C.

Delete the group in Databricks and wait for AIM to recreate it with the correct name and ID on the next sync.

D.

Disable Identity Federation for the workspace, re-enable it, and then run a full SCIM synchronization.

Hint

The goal is to link the existing Databricks identity to the correct unique identifier in the Identity Provider.

4 / 27

An administrator on Databricks on AWS is using Privileged Identity Management (PIM) for Entra ID to grant temporary 'Account Admin' status. Unlike Azure Databricks, why might the user experience a delay after activating their role in PIM?

A.

The user must enable 'Unified Login' at the workspace level for every workspace they intend to manage as an admin.

B.

Databricks on AWS currently relies on SCIM synchronization intervals, which can take 20 to 40 minutes to reflect the PIM membership change.

C.

AWS IAM roles must be manually assumed via the AWS Console before the Databricks Account Console recognizes the Entra ID session.

D.

AWS PrivateLink connections introduce a network propagation delay that affects how quickly identity tokens are validated.

Hint

Consider the differences in how Azure Databricks (first-party) and Databricks on AWS (third-party) integrate with Entra ID's sync mechanisms.

5 / 27

When using the SCIM PATCH API to resolve identity divergences, which JSON payload is correctly formatted to update the *externalID* of a user to the value 8fb4127c….?

A.

{ "externalId": "8fb4127c...", "schemas": ["urn:ietf:params:scim:schemas:core:2.0:User"] }

B.

{ "attributes": { "externalId": "8fb4127c..." }, "op": "add" }

C.
{ "Operations": [{ "action": "update", "field": "externalId", "newValue": "8fb4127c..." }] }
D.
{ "schemas": ["urn:ietf:params:scim:api:messages:2.0:PatchOp"], "Operations": [{ "op": "replace", "path": "externalId", "value": "8fb4127c..." }] }

Hint
Recall the standard SCIM 2.0 Patch operation syntax, specifically the operations array and its required properties.

6 / 27
A workspace administrator wants to share a Databricks SQL Dashboard with a user in their Entra ID tenant who has never accessed Databricks. Automatic Identity Management is enabled. What happens when the dashboard is shared?
A.
The user is added to the account upon login and can view the dashboard, but they are not added as a member of the workspace.
B.
The share request fails initially because the user must first be manually provisioned to the Account Console by an Account Admin.
C.
The dashboard is published as a public link, bypassing Entra ID authentication entirely for that specific user.
D.
The user is automatically added to both the account and the workspace to ensure they have the necessary compute permissions to run the dashboard queries.

Hint
Consider how Databricks balances ease of collaboration with the security of workspace boundaries.

7 / 27
What is a known limitation of Automatic Identity Management regarding cross-tenant Entra ID environments?
A.
AIM does not support cross-tenant Microsoft Entra ID directories and requires SCIM with B2B collaboration instead.
B.
AIM can sync across tenants if Identity Federation is disabled in the primary workspace and enabled in the secondary.
C.
AIM supports cross-tenant sync only if the 'Tenant Boundary Allow List' is configured at the workspace level.
D.
Cross-tenant support is limited to service principals and does not extend to human users or nested groups.

Hint
Identify the standard architectural boundary of the native AIM integration versus legacy SCIM methods.

8 / 27

An admin receives a notification that a group in Entra ID was renamed from *Analytics_Team* to Data_Science_*Global*. How does this change manifest in Azure Databricks with AIM enabled?

A.
AIM creates a duplicate group with the new name, requiring the admin to merge the two identities to preserve permissions.

B.
The group becomes 'Deactivated' until the admin manually re-links the External_ID to the new group name.

C.
The name remains *in Analytics_Team* Databricks until an account admin opens the group's detail page in the account console.

D.
The name is updated immediately across all identity-federated workspaces via a Graph API webhook.

Hint
Recall the specific 'not proactive' behavior mentioned in the documentation regarding group name updates.

9 / 27

What is the security implication for Personal Access Tokens (PATs) when a user is deleted from the identity provider while AIM is enabled?

A.
Databricks automatically deactivates the user, but it does NOT automatically revoke their existing Personal Access Tokens.

B.
Existing PATs are instantly invalidated as soon as the user's status changes to 'Active: Removed From IdP'.

C.
PATs remain active and functional until the user is added to the account-level denylist.

D.
Databricks revokes all tokens and secrets associated with the user only if they were provisioned via SCIM.

Hint
Focus on the distinction between account deactivation and the lifecycle of existing authentication artifacts like tokens.

10 / 27

Which audit log tag is used to distinguish between a manually created group and one created automatically by the AIM process?

A.
syncMode: "AIM"

B.
source: "IdentityProvider"

C.
origin: "EntraIDGraphAPI"

D.
endpoint: "autoUserCreation"

Hint
Identify the internal tag that Databricks uses for all automated identity lifecycle events.

11 / 27
An admin is using the AIM enablement prep script and finds a user with the *EXTERNAL_ID_NOT_IN_IDP* error. What does this specifically indicate?
A.
The Databricks *external_ID* value is set but does not match any existing identity's *object_ID* in the identity provider.
B.
The user exists in Entra ID but hasn't been assigned to the Databricks Enterprise Application yet.
C.
The user was created locally in a non-identity federated workspace and lacks any *external_ID* field.
D.
Entra ID has deactivated the user, but the Databricks account still lists them as 'Active'.

Hint
Focus on what the 'Not in IDP' status means for a value that Databricks already has on file.

12 / 27
In the context of the 2026 behavior change for workspace system groups, how will entitlements be managed for the 'users' group?
A.
The 'users' group will have no entitlements, and workspace entitlements must be granted explicitly when adding principals.
B.
The 'users' group will be automatically deleted and replaced by account-level groups for all entitlement management.
C.
The 'users' group will retain 'Workspace Access' as its only locked entitlement, while all others are migrated.
D.
The 'users' group will inherit all entitlements from the 'admins' group to simplify new user onboarding.

Hint
Consider the shift toward explicit permission granting rather than group-based defaults for system objects.

13 / 27

A company uses AWS and Azure. They want to ensure that a Service Principal (SP) from Entra ID can access Databricks on AWS. What is the current limitation regarding this scenario?

A.
Entra ID service principals can only authenticate to Databricks on AWS if they are part of a nested group structure.

B.
Databricks on AWS requires the Service Principal to be a member of the 'AWS IAM Admin' group to authenticate.

C.
AWS does not support Entra ID as an identity provider for non-human service principals.

D.
The Entra ID SCIM provisioning connector does not support the automatic provisioning of service principals to Databricks on AWS.

Hint: Identify which specific principal type is excluded from the standard Entra ID SCIM provisioning flow.

14 / 27

What is required to establish the authoritative link for syncing identities when Automatic Identity Management is enabled?

A.
The Databricks *INTERNAL_ID* must be manually mapped to the Entra ID App_ID for all Service Principals.

B.
The user's UPN (User Principal Name) must be stored in the Databricks USER NAME field with case-sensitive matching.

C.
Identity Federation must be disabled to allow the 'Access Connector for Azure' to bridge the two directories.

D.
The Microsoft Entra ID *Object id* must match the Databricks *external_id* field.

Hint

Recall which unique identifier from Microsoft Entra ID is used as the 'source of truth' for the sync.

15 / 27

How does Automatic Identity Management handle the provisioning of Service Principals (SPs) that are members of a synced group?

A.
They are provisioned immediately when the group is assigned to a workspace, regardless of activity.

B.
They must be provisioned via the Account SCIM API before they can be added to an AIM-managed group.

C.
They are provisioned only on 'first use', such as their first token authentication or job execution.

D.
Nested SPs are only provisioned if the admin specifically enables 'Service Principal Inheritance' in the security settings.

Hint
Consider when a non-human identity actually needs to exist within the Databricks directory.

16 / 27
An administrator wants to use the 'Context-Based Ingress' (CBI) feature. Which scenario best describes a valid application of this feature in Public Preview?
A.
Automatically routing all traffic from a specific Entra ID group to a private VNet without using Private Link.
B.
Enabling cross-tenant Entra ID synchronization by using the network source as a trust proxy.
C.
Allowing users on an external network to access specific Databricks Apps or Dashboards while blocking access to the workspace UI.
D.
Granting temporary 'Contributor' access to a resource group using Just-In-Time PIM activation for Service Principals.

Hint
Think about how organizations might want to expose self-service analytics to users who aren't on the corporate VPN.

17 / 27
What is the primary difference between how 'Account-level assets' and 'Workspace-level assets' are shared when AIM is enabled?
A.
Workspace-level assets require the admin to first add the group to the workspace, while account-level assets can be shared with any Entra ID group directly.
B.
Account-level assets require SCIM provisioning, whereas workspace-level assets can use the native AIM search.
C.
Workspace-level assets can only be shared with human users, while account-level assets support both users and Service Principals.
D.
There is no difference; both asset types allow for direct sharing with any user or group in the identity provider tenant.

Hint
Recall the rules for sharing objects like notebooks and jobs versus objects like AI/BI dashboards.

18 / 27

An administrator on AWS is setting up Entra ID SSO. They want to ensure that users are automatically provisioned when they sign in. What configuration is recommended as the new default for this flow?

A.
Unified Login
B.
IAM Role Assumption
C.
Account-level SCIM only
D.
VPC Endpoint Mapping

Hint
Think of the setting that centralizes SSO for all workspaces in a single AWS account.

19 / 27

What does a status of 'Inactive: No usage' signify for a group in the Databricks Account Console when AIM is enabled?

A.
The group was deleted from Entra ID but remains in Databricks for permission archival.
B.
The group has no human members and contains only Service Principals.
C.
The group is part of a cross-tenant directory that AIM cannot fully synchronize.
D.
The group exists in Entra ID but has not yet been added to any Databricks workspace.

Hint
Differentiate between an identity that is known but unused and one that has been deactivated or removed.

20 / 27

How does the 'Account Access Denylist' interact with nested groups?

A.
The denylist only blocks users explicitly added to it; nested group membership is ignored for performance reasons.
B.
The denylist can only be applied to Service Principals if they are not part of any group structure.
C.
Users in nested groups are only denied if they are also 'Inactive' in the Account Console.
D.
Denylist membership is transitive; denying a parent group automatically blocks access for all members of its nested groups.

Hint
Think about how a block at a high level of an organization chart usually impacts the teams underneath it.

21 / 27

Which phase of the AIM enablement prep script analyzes workspaces to identify those lacking Identity Federation?

A.
Phase 3
B.
Phase 4
C.
Phase 2
D.
Phase 1

Hint

The script starts by checking if the environment meets the basic prerequisites for AIM.

22 / 27

When disabling Automatic Identity Management, what is the impact on group memberships within Azure Databricks?

A.
Group memberships are preserved, but they no longer sync with the identity provider.
B.
Groups remain in the environment, but all group members are removed.
C.
Groups are deleted entirely to prevent permission conflicts with future SCIM provisioning.
D.
Only nested group memberships are removed; direct memberships are converted to workspace-local status.

Hint

Consider what happens to 'source of truth' data when the connection to that source is severed.

23 / 27

What is the recommended credential design for Unity Catalog storage access on Azure to avoid manual secret rotation?

A.
Azure Managed Identities
B.
Shared Access Signatures (SAS) tokens with 10-year expiry
C.
IAM User Access Keys from a cross-cloud bridge
D.
Service Principal Client Secrets stored in Azure Key Vault

Hint: Identify the secretless identity feature native to Azure that is integrated with Databricks Unity Catalog.

24 / 27

An administrator sees the error $EXTERNAL\_ID\_MATCH\_NAME\_MISMATCH$ for a Service Principal. What does this indicate and how should it be resolved?

A. The $externalId$ maps to an identity with a different unique name in the IdP; the SP's $applicationId$ in Databricks must be updated.

B. Two identities have the same name but different sources; the admin must delete the SCIM-provisioned identity.

C. The group name was changed in Entra ID but not in Databricks; the admin should open the group detail page.

D. The Service Principal has no $externalId$ and must be provisioned via Account SCIM.

25 / 27

Which of the following is NOT a benefit of AIM compared to traditional SCIM provisioning?

A.
Support for synchronizing nested groups.

B.
Support for synchronizing Entra ID service principals natively.

C.
Simplified onboarding without the need for a separate Enterprise Application for provisioning.

D.
Ability to synchronize identities across different Azure tenants natively.

Hint

Look for the one scenario where the documentation explicitly says AIM is not supported.

26 / 27

During the GA release of AIM for Entra ID on AWS and GCP, what other identity provider was announced for Public Preview?

A.
OneLogin

B.
Ping Identity

C.
Google Identity Platform

D.
Okta

Hint: Identify the popular identity provider that is now in Public Preview for AIM on non-Azure clouds.

27 / 27

What is the recommended action for resolving $GROUP\_HAS\_LOCAL\_MEMBERS\_WITH\_EXTERNAL$ to ensure the IdP is the source of truth?

A. Delete the $externalId$ from the affected users so they are treated as purely local Databricks identities.

B. Rename the local group so it does not conflict with the group name in Entra ID.

C. Remove the locally added members from the group via SCIM and add them to the group in the IdP instead.

D. The admin must 'Approve' the local membership in the Account Console to bypass the sync check.

Answer

1. A 2. A 3. A 4. B 5. D 6. A 7. A 8. C 9. A 10. D 11. A 12. A 13. D 14. D 15. C 16. C 17. A 18. A 19. D 20. D 21. A 22. B 23. A 24. A 25. D 26. D 27. C

SME Scenario-Based Case Studies

Case Study A: Mitigating Latency in Just-in-Time (JIT) Administrative Role Elevation (PIM Integration)

1. The Problem Scenario
An enterprise uses Microsoft Entra ID as its central Identity Provider (IDP). An Azure Databricks platform administrator activates their "Metastore Admin" group role using Azure Privileged Identity Management (PIM). Under their Entra ID P2 or Entra Governance license, they receive a time-bound role assignment to the group DB-Metastore-Admins.
However, immediately after activating their role in the Azure portal, the user attempts to log in to Databricks to configure Unity Catalog storage credentials. They are greeted with a Permission Denied error. If the organization uses a standard SCIM provisioning connector, the user remains in an inactive state or lacks group permissions within Databricks.

2. Under the Hood Mechanics

1. The SCIM Latency Hurdle: When PIM activates, it updates the user's group memberships inside Microsoft Entra ID instantly. However, the Databricks SCIM enterprise application is a *push model* that operates on a scheduled synchronization interval. The initial sync triggers immediately upon setup, but subsequent updates sync only every 20 to 40 minutes. Until this SCIM synchronization cycle runs (or an admin manually restarts it inside the Azure portal), Databricks remains unaware that the user has joined DB-Metastore-Admins.

2. The AIM Dynamic Pull Solution: Unlike SCIM, Automatic Identity Management (AIM) leverages direct Microsoft Graph API queries initiated by authentication and authorization check triggers in Databricks. When AIM is enabled, group memberships are updated automatically on-demand based on two distinct schedules:
    - Browser Logins: Sync is triggered dynamically if more than 5 minutes have elapsed since the user's last identity sync.
    - Other Activities (Token authentication, JDBC connection, running jobs): Sync is triggered if more than 40 minutes have elapsed since the last sync.

3. Step-by-Step Resolution Path

1. Decommission the Legacy SCIM Connector: Transition the workspace or account entirely to AIM using the single toggle under the Security > User provisioning tab in the Databricks Account Console to avoid parallel-connector sync conflicts.

2. Onboard the PIM Group to Databricks: Register the Entra ID group DB-Metastore-Admins inside Databricks using the Databricks CLI, passing its Entra ID ObjectId as the externalId:
databricks accounts groups create --display-name "DB-Metastore-Admins" --external-id "<ENTRA_ID_GROUP_OBJECT_ID>"

3. Trigger Just-In-Time Evaluation: Advise the user to activate their role in the Azure PIM portal. Once activated, when they perform a browser-based login to Databricks, AIM immediately queries the Entra Graph API (respecting the 5-minute cache window) to fetch their transitive group memberships, assigning the workspace or Metastore privileges instantly.

Case Study B: Nested Group Access Cascades and Programmatic API Boundaries

1. The Problem Scenario

A platform architect designs an access control hierarchy in Microsoft Entra ID:

- Marketing-All (Parent Group explicitly provisioned to Databricks)
    - Marketing-US (Child Group nested inside Marketing-All)
    - Marketing-EU (Child Group nested inside Marketing-All)

The workspace administrator assigns Marketing-All to a target workspace. The architect attempts to run a nightly auditing Python script that queries the Databricks SCIM API /api/2.0/preview/scim/v2/Groups to list members of Marketing-US and ensure all users have accepted the data policy. The API returns an empty list or a 404 Not Found error, despite users in Marketing-US actively logging in and executing queries in the workspace.

2. Under the Hood Mechanics

- Access vs. Provisioning Realities: AIM natively supports nested groups for authorization checks. If a parent group (Marketing-All) is assigned to a workspace, all immediate and transitive members (users and service principals within child groups Marketing-US and Marketing-EU) are granted access to authenticate and use resources.
- The Shell-Only Provisioning Limit: To prevent account limit bloat (especially in massive tenants with 30+ layers of nesting), AIM does not automatically pre-provision or construct the entire parent group hierarchy in Databricks. It only provisions the explicitly assigned parent group (Marketing-All).
- The API and Terraform Blind Spot: Because the child groups (Marketing-US, Marketing-EU) are not directly provisioned to the Databricks account, they are visible as nested entries in the Account Console UI but cannot be retrieved, searched, or managed via Databricks REST SCIM APIs or Terraform.

3. Step-by-Step Resolution Path

1. Assess the Integration Strategy: If you only require users inside Marketing-US to inherit access and Unity Catalog privileges assigned to Marketing-All, no change is required—permissions cascade transitively.
2. Explicitly Provision Child Groups for Programmatic Audits: To audit child groups programmatically via the API or configure them individually in Terraform, you must explicitly provision them.
3. Use the SCIM API to Provision the Group Shell:

```
POST /api/2.1/accounts/<accountId>/scim/v2/Groups
{
  "displayName": "Marketing-US",
  "externalId": "<ENTRA_ID_MARKETING_US_OBJECT_ID>",
  "schemas": ["urn:ietf:params:scim:schemas:core:2.0:Group"]
}
```

Once explicitly provisioned, the child group shell exists in the local directory, counts toward the account group limits, and becomes queryable via the Databricks SCIM APIs.

Case Study C: Troubleshooting Identity Divergences on AIM Transition

1. The Problem Scenario

An organization with 10,000 existing users and groups is migrating from a legacy Azure Active Directory SCIM connector to Automatic Identity Management (AIM). After turning on AIM, platform administrators notice several critical issues:

1. Duplicate user identities are appearing in sharing modals (e.g., two entries for admin@company.com).
2. Newly created Entra ID groups with names identical to legacy local groups fail to sync or provision into Databricks.
3. Sharing an AI/BI Dashboard with a newly added Entra ID user succeeds, but the user count displayed on the admin UI does not match the actual members list.

2. Under the Hood Mechanics

- The Role of the External ID: AIM relies on the provisioned Databricks identity's externalId matching the corresponding principal's ObjectId in Microsoft Entra ID. If an identity was created manually or via an outdated SCIM mapping, its externalId may be missing or mismatched. This causes Databricks to treat the local identity and the Entra ID identity as two separate objects, leading to duplicates.
- Unique Name Constraints: Databricks enforces a strict unique group name constraint at the account level. If a legacy account group has a name reservation but a different or missing externalId, the IDP group with the same name cannot be provisioned.
- Divergent Group Membership counts: SCIM-synced groups allow local, mutable member additions inside Databricks. AIM relies purely on the IDP as the source of truth for membership counts. Any local group members who lack an externalId will be skipped in synced member counts, making audits difficult.

3. Step-by-Step Resolution Path

1. Deploy the AIM Enablement Prep Script: Download the divergence.zip notebook, import it into your workspace, and provide an account-level service principal with account admin OAuth credentials to authenticate.
2. Configure and Execute the Scan: Fill out config.py with your ACCOUNT_ID and execute the run_divergence notebook to generate reports under divergence/results/.
3. Triage NAME_MATCH_EXTERNAL_ID_MISMATCH (Resolving Duplicates): Identify the Databricks internal ID (databricksId) and the correct Entra ID objectId from the idp_divergence_users.csv report. Resolve this by issuing an Account SCIM PATCH API call to update the externalId:
    - HTTP Request: PATCH https://accounts.azuredatabricks.net/api/2.1/accounts/<accountId>/scim/v2/Users/<databricksId>
    - Payload:

```
{
  "schemas": ["urn:ietf:params:scim:api:messages:2.0:PatchOp"],
  "Operations": [
    {
      "op": "replace",
      "path": "externalId",
      "value": "<ENTRA_ID_USER_OBJECT_ID>"
    }
```

]
}
``` [27]
1. Triage GROUP_HAS_LOCAL_MEMBERS_WITHOUT_EXTERNAL_ID (Cleaning Group Counts): Review idp_divergence_groups.csv. Remove any locally added users from the group via the SCIM API or Workspace UI. For any legitimate members, add them to the parent group directly within Microsoft Entra ID so that their membership is managed from the correct authoritative source.

---

**Multi-Cloud Architecture & Forward-Looking SME Highlights**

- AWS Unified Login & JIT: While Azure Databricks uses first-party Entra ID out of the box, AWS relies on Unified Login to configure SSO once at the account level and automatically cascade it to all identity-federated workspaces.
- Google Cloud BYO-IDP Model: Previously restricted to default Google Identity configurations, GCP Databricks deployments are shifting to a Bring-Your-Own-IDP (BYO-IDP) model. This allows platform teams to standardize on their preferred identity provider (e.g., Entra ID, Okta) across GCP, mirroring Azure and AWS configurations.
- Context-Based Ingress (CBI) Governance: Now in Public Preview, CBI allows administrators to enforce dynamic, zero-trust policies that restrict user access to sensitive platform layers while safely exposing specific data apps, AI/BI dashboards, and Genie endpoints to unprovisioned external users.
```

Generate a complete Terraform blueprint to onboard and secure Unity Catalogue storage credentials using Azure Managed Identities!

unity-catalog-storage.tf is provided at the end. This complete, production-grade **Terraform blueprint** implements best practices for onboarding and securing ADLS Gen2 storage paths in Unity Catalogue using Azure Managed Identities.

Architecture Overview & Core Components of the Blueprint

This blueprint completely eliminates the need for manual secret rotation or stored credentials by using a native Azure-to-Databricks trust relationship. It provisions:

1. **Secure ADLS Gen2 Storage Account & Container:** Deploys a Storage Account with **Hierarchical Namespace enabled** (is_hns_enabled = true), which is a mandatory requirement for performance and directory-level transactional guarantees in Delta Lake. It includes defensive security rules to deny public access and restrict ingress.
2. **Access Connector for Azure Databricks:** Standardizes on an Access Connector with a **System-Assigned Managed Identity**, which is the first-party cloud identity layer recommended by Databricks for secretless metadata connectivity.
3. **Storage Blob Data Contributor Role Assignment:** Establishes the trust relationship by granting the Managed Identity scope-specific read/write permissions directly on the target storage account.
4. **Unity Catalog Registration & Storage Credential:** Integrates Azure's identity with the Lakehouse, registering the Access Connector's ID as a **Storage Credential** within the metastore.
5. **Unity Catalog External Location:** Scopes the Storage Credential to the safe storage path using the secure abfss:// protocol.
6. **Strict RBAC Grant Structure:** Implements the Unity Catalog privilege model, granting ALL_PRIVILEGES to administrators and targeting read/write privileges (READ_FILES, WRITE_FILES) to designated **Data Engineering** groups for robust data governance.

How to Deploy This Blueprint

To execute this plan, save the variables and configuration details to your local workspace and initialize the cloud providers:

1. **Install Providers:** Ensure you have the azurerm and databricks CLI provider configurations configured on your host machine.
2. **Set Credentials:** Authenticate the Databricks provider via environment variables (DATABRICKS_HOST and your token) or your terminal session.
3. **Execute Deployment:** Run the standard Terraform lifecycle commands:

```
terraform init
terraform plan -out=tfplan
terraform apply tfplan
```

```
# ================================================================================
# TERRAFORM BLUEPRINT: ONBOARDING & SECURING STORAGE CREDENTIALS IN UNITY CATALOG
# ================================================================================
# Focus: Securing ADLS Gen2 storage using Azure Access Connector Managed Identities
# and establishing governed Unity Catalog External Locations with strict RBAC/ABAC.
# ================================================================================

terraform {
  required_providers {
    azurerm = {
      source  = "hashicorp/azurerm"
      version = "~> 3.0"
    }
    databricks = {
      source  = "databricks/databricks"
      version = "~> 1.0"
    }
  }
}

# --------------------------------------------------------------------------------
# VARIABLES
# --------------------------------------------------------------------------------

variable "resource_group_name" {
  type        = string
  description = "The name of the Azure Resource Group where resources will be deployed."
  default     = "rg-databricks-prod"
}
```

```
variable "location" {
  type        = string
  description = "Azure Region for the deployment."
  default     = "eastus2"
}

variable "storage_account_name" {
  type        = string
  description = "Name of the ADLS Gen2 Storage Account. Must be globally unique."
  default     = "sadatabricksprod001"
}

variable "container_name" {
  type        = string
  description = "Name of the storage container inside ADLS Gen2."
  default     = "unity-catalog-external"
}

variable "access_connector_name" {
  type        = string
  description = "The name of the Azure Databricks Access Connector resource."
  default     = "db-access-connector"
}

variable "storage_credential_name" {
  type        = string
  description = "The name of the Storage Credential registered in Unity Catalog."
  default     = "db_storage_credential"
}

variable "external_location_name" {
  type        = string
  description = "The name of the External Location registered in Unity Catalog."
```

```
  default     = "db_external_location_silver"
}

variable "admin_group_name" {
  type        = string
  description = "Databricks account/workspace admin group display name."
  default     = "admins"
}

variable "data_engineers_group_name" {
  type        = string
  description = "Databricks group display name for Data Engineers who require read/write file permissions."
  default     = "data-engineers"
}

variable "allowed_ip_ranges" {
  type        = list(string)
  description = "Allowed public IP ranges for Storage Account firewall bypass (CIDR format)."
  default     = []
}

variable "subnet_ids" {
  type        = list(string)
  description = "Azure Subnet IDs allowed to access the Storage Account via service endpoints."
  default     = []
}

# ------------------------------------------------------------------------
# AZURE RESOURCE PROVISIONING
# ------------------------------------------------------------------------

# Create the target Resource Group
resource "azurerm_resource_group" "rg" {
```

```
  name     = var.resource_group_name
  location = var.location
}

# ADLS Gen2 Storage Account (is_hns_enabled = true is mandatory for Delta/ADLS optimization)
resource "azurerm_storage_account" "adls_gen2" {
  name                     = var.storage_account_name
  resource_group_name      = azurerm_resource_group.rg.name
  location                 = azurerm_resource_group.rg.location
  account_tier             = "Standard"
  account_replication_type = "LRS"
  account_kind             = "StorageV2"
  is_hns_enabled           = true # Essential hierarchical namespace for directory-level operations

  # Default Network Security Rules (Enforce storage firewalls for robust isolation)
  network_rules {
    default_action             = length(var.allowed_ip_ranges) > 0 || length(var.subnet_ids) > 0 ?
"Deny" : "Allow"
    bypass                     = ["AzureServices"] # Allow internal Azure platform services to bypass
    ip_rules                   = var.allowed_ip_ranges
    virtual_network_subnet_ids = var.subnet_ids
  }

  tags = {
    Environment = "Production"
    Governance  = "UnityCatalog"
  }
}

# Secure Container to act as our External Location root path
resource "azurerm_storage_container" "container" {
  name                  = var.container_name
  storage_account_name  = azurerm_storage_account.adls_gen2.name
  container_access_type = "private" # Restrict public access completely
```

```
}

# ---------------------------------------------------------------------------
# ACCESS CONNECTOR & IDENTITY ASSIGNMENT
# ---------------------------------------------------------------------------

# The Azure Databricks Access Connector acts as the host for the System-Assigned Managed Identity
resource "azurerm_databricks_access_connector" "uc_access_connector" {
  name                = var.access_connector_name
  resource_group_name = azurerm_resource_group.rg.name
  location            = azurerm_resource_group.rg.location

  identity {
    type = "SystemAssigned"
  }
}

# Assign "Storage Blob Data Contributor" role to the Access Connector's Managed Identity
resource "azurerm_role_assignment" "storage_data_contributor" {
  scope                = azurerm_storage_account.adls_gen2.id
  role_definition_name = "Storage Blob Data Contributor"
  principal_id         = azurerm_databricks_access_connector.uc_access_connector.identity[0].principal_id
}

# ---------------------------------------------------------------------------
# DATABRICKS UNITY CATALOG GOVERNANCE & ACCESS CONTROL
# ---------------------------------------------------------------------------

# Register the Azure Access Connector Managed Identity in Unity Catalog as a Storage Credential
resource "databricks_storage_credential" "uc_credential" {
  name = var.storage_credential_name
  azure_managed_identity {
```

```
    access_connector_id = azurerm_databricks_access_connector.uc_access_connector.id
  }
  comment = "Secretless credential managed by Terraform using Databricks Access Connector System Identity"

  # Explicit dependency to ensure the Azure role assignment has propagated in Azure AD RBAC first
  depends_on = [azurerm_role_assignment.storage_data_contributor]
}

# Create the External Location mapping the Storage Container path to the Storage Credential
resource "databricks_external_location" "uc_external_location" {
  name            = var.external_location_name
  url             = "abfss://${azurerm_storage_container.container.name}@${azurerm_storage_account.adls_gen2.name}.dfs.core.windows.net/"
  credential_name = databricks_storage_credential.uc_credential.id
  comment         = "Governed external path for production Silver and Gold medallion tables"
}

# ------------------------------------------------------------------------
# PRIVILEGE GRANTS (Unity Catalog RBAC)
# ------------------------------------------------------------------------

# Restrict the Storage Credential so only Admins can alter/reconfigure it
resource "databricks_grants" "credential_grants" {
  storage_credential = databricks_storage_credential.uc_credential.id

  grant {
    principal  = var.admin_group_name
    privileges = ["ALL_PRIVILEGES"]
  }
}

# Delegate access to the External Location for the Data Engineering and Admin groups
```

```
resource "databricks_grants" "location_grants" {
  external_location = databricks_external_location.uc_external_location.id

  # Admins have full structural access (needed to create tables, managed paths, and grant permissions)
  grant {
    principal  = var.admin_group_name
    privileges = ["ALL_PRIVILEGES"]
  }

  # Data Engineers have read/write access to interact with the underlying data files
  # (e.g., executing COPY INTO or external table writes)
  grant {
    principal  = var.data_engineers_group_name
    privileges = ["READ_FILES", "WRITE_FILES"]
  }
}

# ------------------------------------------------------------------------
# OUTPUTS
# ------------------------------------------------------------------------

output "resource_group_id" {
  value       = azurerm_resource_group.rg.id
  description = "The Resource Group ID where resources were deployed."
}

output "access_connector_principal_id" {
  value       = azurerm_databricks_access_connector.uc_access_connector.identity[0].principal_id
  description = "The principal ID of the Access Connector's system-assigned managed identity."
}

output "storage_account_dfs_endpoint" {
  value       = azurerm_storage_account.adls_gen2.primary_dfs_endpoint
```

```
  description = "The primary DFS endpoint for the ADLS Gen2 storage."
}

output "external_location_url" {
  value       = databricks_external_location.uc_external_location.url
  description = "The registered Unity Catalog External Location URL."
}
```

Nested Group Behavior: AIM vs. SCIM

The behavior of nested groups represents one of the most critical structural differences between legacy SCIM provisioning and the modern **Automatic Identity Management (AIM)** framework in Databricks.

---

1. Legacy SCIM Provisioning

- **No Native Support:** Traditional SCIM connectors (such as the Microsoft Entra ID SCIM Provisioning Connector) **do not support the sync or provisioning of nested groups**.
- **Immediate Members Only:** The SCIM application can only read and provision users who are **immediate, direct members** of the explicitly assigned group.
- **Administrative Workarounds:** To grant access to nested users under a SCIM model, administrators are forced to manually flatten group structures, explicitly assign nested child groups to the SCIM enterprise app in the Azure portal, or utilize custom API scripts and Terraform to mirror nested layouts.

---

2. Automatic Identity Management (AIM)

- **Native Transitive Support:** AIM natively supports nested groups. It fetches transitive (nested) group memberships from Microsoft Entra ID on-demand during authentication and authorization checks (such as browser logins or job runs). If User A is in Group child, and Group child is nested inside Group parent, Databricks recognizes User A as a member of both groups.
- **Automatic Permission Inheritance:** All users and service principals in nested child groups automatically inherit permissions assigned to the parent group. For example, if a parent group is assigned workspace access, all members of its nested child groups can log in and use the workspace without needing direct assignment.
- **Account Group Limit Optimization:** To prevent account limit bloat, nested groups and service principals that are not directly provisioned to the Databricks account **do not count toward account-level group limits**. Only explicitly provisioned parent groups count against your directory limits.
- **UI Visibility Bounds:** In the account and workspace admin consoles, nested groups are visible on the parent group's members detail page, but **only the first level of nesting is displayed**. Furthermore, nested child groups will not show up in UI sections that only return explicitly provisioned identities.
- **API & Terraform Blind Spot:** Nested groups that have not been explicitly provisioned to the account **cannot be retrieved, edited, or managed through the Databricks SCIM APIs or Terraform**. If you need to manage these child groups programmatically or individually configure them, you must explicitly provision them to the Databricks account

What are the common identity sync divergences in AIM?

Automatic Identity Management (AIM) relies on the externalId in Databricks precisely matching the unique Object ID of the corresponding principal in your Identity Provider (IdP). When these values are missing or misconfigured, it results in **identity sync divergences**.

The primary divergence categories flagged by the Databricks AIM enablement tool include:

1. Identity Mismatches (Duplicate Users & Group Failures)

- **NAME_MATCH_EXTERNAL_ID_MISMATCH**: The Databricks identity has a unique name (such as username or group name) that matches an IdP identity, but their externalId fields do not match. This mismatch frequently causes **duplicate identities** to appear in admin UIs or sharing modals. It also causes **IdP group provisioning to fail** because Databricks enforces a strict unique group name constraint, and the existing local group name is already reserving it.
- **EXTERNAL_ID_NOT_IN_IDP**: The Databricks identity has an externalId set, but it does not match any identity of the same type in the IdP. This occurs when external IDs are misconfigured or stale. Unresolved, it can create duplicates if the user attempts to log in and is newly provisioned from the IdP.
- **EXTERNAL_ID_MATCH_NAME_MISMATCH**: The Databricks identity has an externalId that maps to an IdP identity, but the unique name (e.g., username) differs. When users log in, this mismatch often causes a second user to be created. For groups, it prevents the group from syncing its name with its IdP counterpart.

2. Group Membership Divergences (Auditing Discrepancies)

- **GROUP_HAS_LOCAL_MEMBERS_WITHOUT_EXTERNAL_ID**: The Databricks group contains members with no externalId (local users) who do not exist in the IdP group. Because **Databricks UI member counts only reflect IdP group counts**, these local members will not show up in the tally, making auditing and governance difficult.
- **GROUP_HAS_LOCAL_MEMBERS_WITH_EXTERNAL_ID**: The Databricks group has members with an externalId, but they do not have a corresponding membership in the IdP group. This also distorts UI member counts and creates complications when trying to turn off SCIM.

3. Workspace Incompatibilities

- **IDENTITY_FEDERATION_DISABLED**: The workspace does not have identity federation enabled. Because AIM only functions within identity-federated workspaces, account-level and IdP identities will not be available in these workspaces.

The 'Bring-Your-Own-IDP' (BYO-IDP) Model on Google Cloud

Historically, when spinning up a new Databricks account on **Google Cloud Platform (GCP)**, Single Sign-On (SSO) was automatically configured and restricted to **Google Identity**.

However, in response to strong customer feedback, Databricks has shifted this GCP identity model. Under this new paradigm, when you create a GCP account, you can configure both **SSO and SCIM provisioning** against **whichever identity provider (IDP) you choose**. This transition effectively breaks the native lock-in to Google Identity and enables a true "Bring-Your-Own-IDP" deployment strategy.

Key Capabilities of the Shift on GCP

1. **Native Identity Provider Alignment:** Rather than managing an isolated Google Identity directory for your GCP analytics workloads, you can align GCP Databricks workspaces with your enterprise-wide standard identity platform (such as Okta or Microsoft Entra ID).
2. **General Availability of AIM for Entra ID: Automatic Identity Management (AIM)** for Microsoft Entra ID is now **Generally Available** on GCP. This removes the need to pre-provision users or build custom, bespoke SCIM sync pipelines. Users, groups, and service principals are automatically pulled on-demand from Entra ID into Databricks the moment they are granted platform access or have an asset shared with them.
3. **Public Preview of AIM for Okta:** If your organization standardizes on Okta, **AIM for Okta is currently in Public Preview on GCP**. This allows Okta identities to sync seamlessly with the same event-driven, on-demand benefits as Entra ID, maintaining consistent group memberships and cascading permissions dynamically.
4. **Workspace Isolation & Secure Sharing:** Enabling this model on GCP simplifies secure collaboration. For example, you can safely share an AI/BI dashboard, Databricks App, or Genie Room with any user in your chosen IDP directory. If that user has never logged into Databricks before, they are **automatically provisioned at the account level** upon accessing the link. Crucially, they are *not* automatically added to the underlying workspace, thereby preserving your strict workspace security boundaries.

Difference between browser login and job run sync schedules

When **Automatic Identity Management (AIM)** is enabled, Azure Databricks refreshes user group memberships from your identity provider (such as Microsoft Entra ID) during activities that trigger authentication and authorization checks.

The critical difference between these two schedules lies in the **minimum time required to elapse** before Databricks will trigger a refresh:

- **Browser Logins:** Group memberships sync if more than **5 minutes** have passed since the last identity sync. This short window is designed to support rapid Just-in-Time (JIT) access, allowing users who elevate their roles (e.g., via PIM) to have those changes reflected in the Databricks UI almost immediately upon logging in via their browser.
- **Job Runs (and Other Activities):** For job runs, token authentication, or API executions, group memberships sync only if more than **40 minutes** have passed since the last sync.

This longer interval prevents system overhead and API throttling during automated, high-frequency, or programmatic operations.