

# ATTRIBUTE BASED ACCESS CONTROL

Building a scalable access  
management framework

---

Kristen Wilder, Zeashan Pappa



# Meet our presenters



**Zeashan Pappa**

- 23+ years of experience as an engineer and architect
- 6 years of experience building with Databricks
- 10 years of experience building scalable platforms and companies
- Product manager working on catalog, governance, security products



**Kristen Wilder**

- Product manager at Databricks focused on discovery and governance experiences
- 4+ years of experience across product management and FE engineering
- 3 years specializing on improving Governance and Data Access Management

# Product safe harbor statement

This information is provided to outline Databricks' general product direction and is for **informational purposes only**. Customers who purchase Databricks services should make their purchase decisions relying solely upon services, features, and functions that are currently available. Unreleased features or functionality described in forward-looking statements are subject to change at Databricks discretion and may not be delivered as planned or at all

# Agenda

1. ABAC Overview
2. Fundamentals
3. Story & Demo
4. Anatomy of a rule
5. Use Cases
6. Q/A

# ABAC Overview

## Attribute Based Access Control

ABAC allows access control to be conditional based on broader properties of the user, resource, and the request.

- Builds on top of and coexists with the current UC security model, including all privileges
- Use row level filters and column level masking for fine grained access control
- Governs data, AI, & filesystems

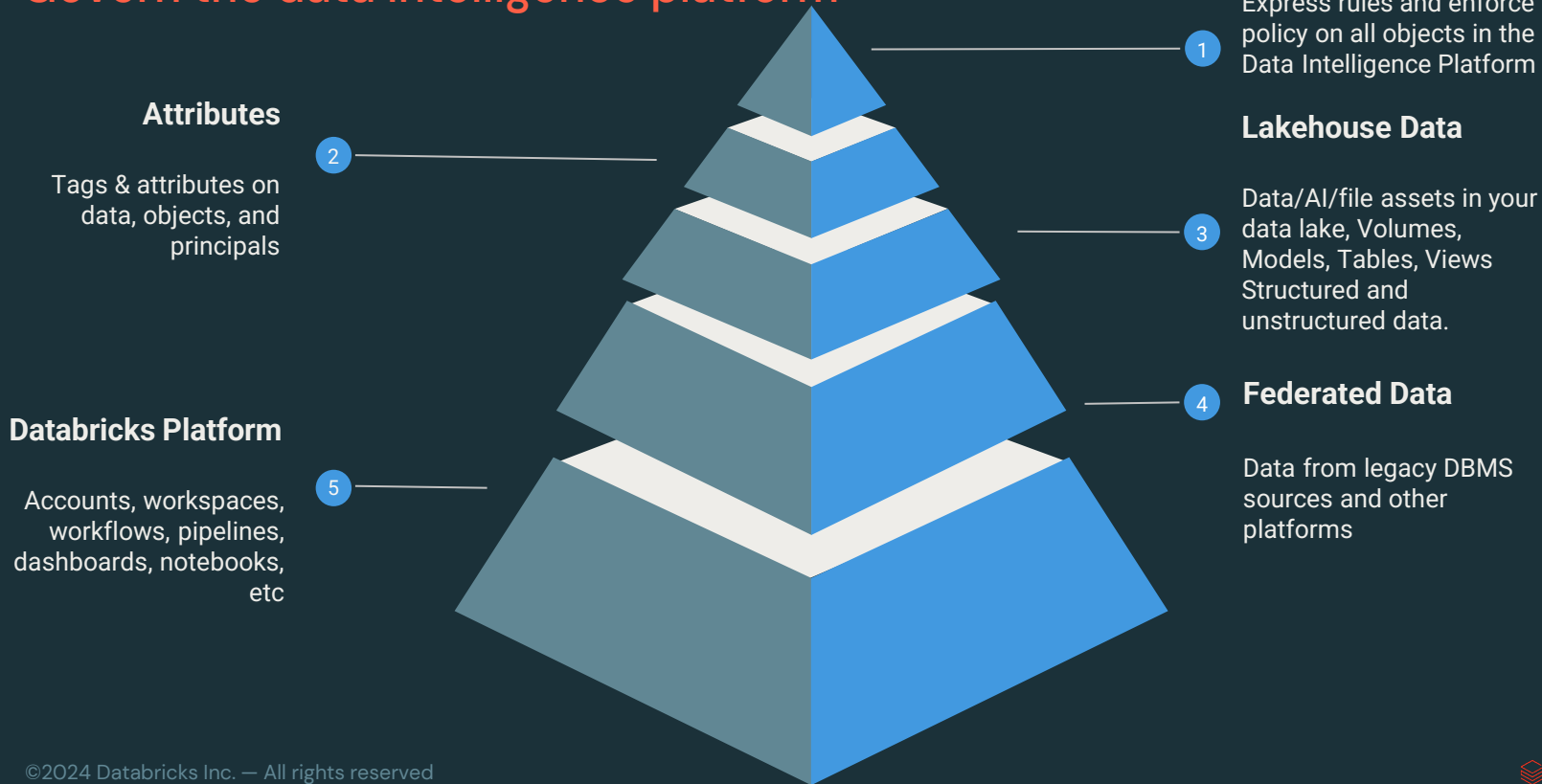
# High Leverage Governance with ABAC

## Govern Data & AI at scale



# ABAC Vision

## Govern the data intelligence platform



# Concepts and Fundamentals



# Concepts



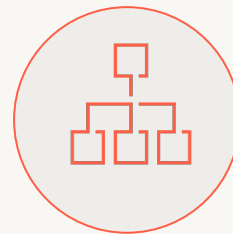
## Attributes

Attributes describe data and other objects



## Rules

Rules determine access control

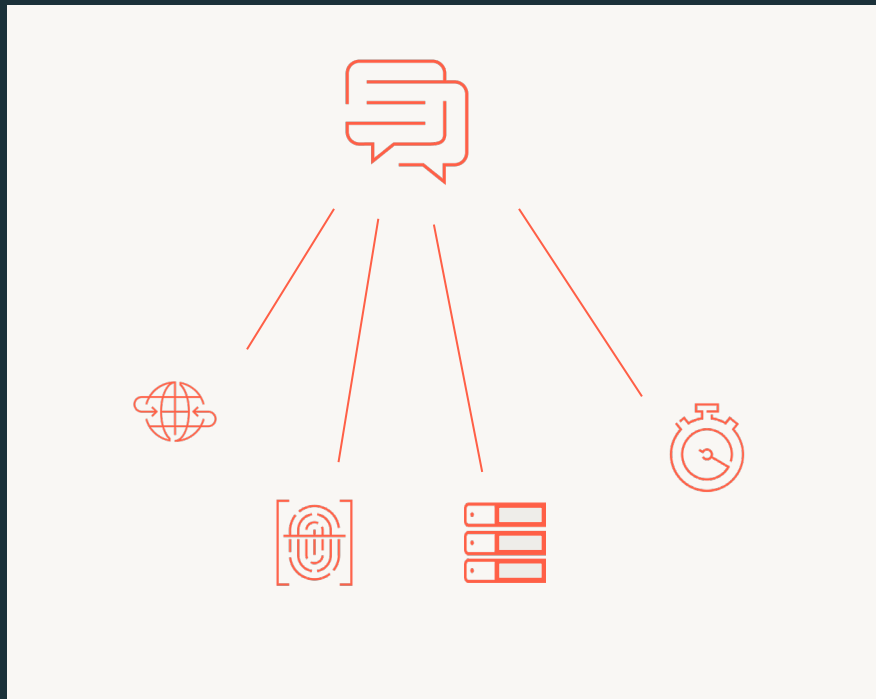


## Inheritance

Rules and tags are inherited and accumulate to express policy

# More on Attributes

Attributes can be used to inform rules

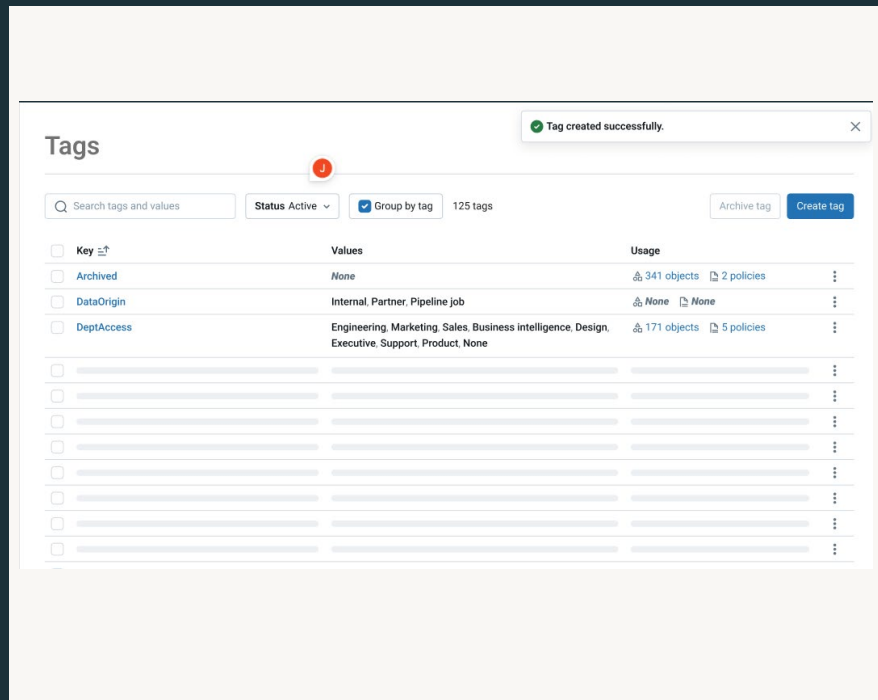


Attributes are derived from the context of the object where policy is evaluated

- Location
  - Workspace, IP, etc
- Identity
  - User, group, etc
- Tag
  - User or system assigned
- Time
  - Current datetime of the request

# What are Tags?

## Introducing governed tags



The screenshot displays the Databricks Tags management interface. At the top, a notification states "Tag created successfully." Below this, the "Tags" section includes a search bar for "Search tags and values", a "Status" dropdown set to "Active", and a "Group by tag" dropdown showing "125 tags". There are "Archive tag" and "Create tag" buttons. The main content is a table with the following data:

Key	Values	Usage
<input type="checkbox"/> Archived	None	341 objects 2 policies
<input type="checkbox"/> DataOrigin	Internal, Partner, Pipeline job	None None
<input type="checkbox"/> DeptAccess	Engineering, Marketing, Sales, Business intelligence, Design, Executive Support, Product, None	171 objects 5 policies
<input type="checkbox"/>		
<input type="checkbox"/>		
<input type="checkbox"/>		
<input type="checkbox"/>		
<input type="checkbox"/>		
<input type="checkbox"/>		
<input type="checkbox"/>		
<input type="checkbox"/>		
<input type="checkbox"/>		

- Tags are scoped at the account level
- Highly flexible attribute type
- Can be leveraged to classify any Databricks object
- Can be used to establish your company's business taxonomy
- Used for access management via ABAC, resource usage, data retention, processing pipelines, billing, and discovery.

# How are Tags used?



Tags can be used for governance, discovery, attribution

## Governance

Data steward wants to classify sensitive data and ensure their organization is GDPR compliant.

Assigns a pii tag to relevant columns

Credit_card	Ssn	Zip_code
pii: ccn	pii: ssn	pii: zip
123-12334	123456789	92660

## Discovery

Data engineer wants to find all tables related to the marketing domain

Assigns a tag, "domain: marketing"

Transactions\_table  
domain:marketing

fiscal\_2025  
domain:marketing

## Attribution

Admin wants to track usage and estimate costs.

Assigns a tag, costcenter: finance to the finance related assets

# How are Tags governed?

Tags provide input to rules, and must be governed.

- Restrict which users can **create, update, delete tags**
- Control which users can **assign and unassign tags**
- Leverage flexible permission sets
  - Manage tags set
  - Manage tag assignments set

The screenshot shows the 'Tags' management page for 'DeptAccess'. It includes a 'Delete' button, tabs for 'Overview', 'Permissions', 'Policies', and 'Objects tagged', and a 'Grant access' button. A table lists the principals with their names, roles, and permissions granted.

Type	Name	Permissions granted
Person	Will Aaron will.a@company.com	Tag: User
Group	Administrators 12 members	Tag: Manager
Service	Ops services abcd3807-ce21-4c3d-9363-a7579bfe6ed8	Tag: Manager

# How can you interact with Tags?

Users can view a centralized Tags list

- Tags
  - Via REST API
  - Via UI
- Tag assignments
  - Via SQL
  - Via REST API
  - Via UI
- REST based integration support for 3rd party tag sources

The screenshot displays the Databricks 'Tags' management interface. At the top right, a green notification states 'Tag created successfully.'. The main panel features a search bar, a status filter set to 'Active', and a 'Group by tag' button. Below the search bar, there are 'Archive tag' and 'Create tag' buttons. The central part of the interface is a table listing tags with columns for 'Key', 'Values', and 'Usage'. The 'Key' column includes a checkbox and a magnifying glass icon. The 'Usage' column shows object and policy counts with expandable icons. The table lists several tags, including 'Archived', 'DataOrigin', and 'DeptAccess', with their respective values and usage statistics.

<input type="checkbox"/> Key 🔍	Values	Usage
<input type="checkbox"/> Archived	None	341 objects 2 policies
<input type="checkbox"/> DataOrigin	Internal, Partner, Pipeline job	None None
<input type="checkbox"/> DeptAccess	Engineering, Marketing, Sales, Business Intelligence, Design, Executive, Support, Product, None	171 objects 5 policies
<input type="checkbox"/> [Redacted]	[Redacted]	[Redacted]
<input type="checkbox"/> [Redacted]	[Redacted]	[Redacted]
<input type="checkbox"/> [Redacted]	[Redacted]	[Redacted]
<input type="checkbox"/> [Redacted]	[Redacted]	[Redacted]
<input type="checkbox"/> [Redacted]	[Redacted]	[Redacted]
<input type="checkbox"/> [Redacted]	[Redacted]	[Redacted]
<input type="checkbox"/> [Redacted]	[Redacted]	[Redacted]
<input type="checkbox"/> [Redacted]	[Redacted]	[Redacted]
<input type="checkbox"/> [Redacted]	[Redacted]	[Redacted]
<input type="checkbox"/> [Redacted]	[Redacted]	[Redacted]
<input type="checkbox"/> [Redacted]	[Redacted]	[Redacted]
<input type="checkbox"/> [Redacted]	[Redacted]	[Redacted]
<input type="checkbox"/> [Redacted]	[Redacted]	[Redacted]
<input type="checkbox"/> [Redacted]	[Redacted]	[Redacted]
<input type="checkbox"/> [Redacted]	[Redacted]	[Redacted]
<input type="checkbox"/> [Redacted]	[Redacted]	[Redacted]
<input type="checkbox"/> [Redacted]	[Redacted]	[Redacted]
<input type="checkbox"/> [Redacted]	[Redacted]	[Redacted]
<input type="checkbox"/> [Redacted]	[Redacted]	[Redacted]
<input type="checkbox"/> [Redacted]	[Redacted]	[Redacted]
<input type="checkbox"/> [Redacted]	[Redacted]	[Redacted]

# What are Rules?

Rules use tags and other attributes to decide access.

- Applied on **tables, schemas, catalogs**
- Target **users, groups or roles**
- **Inherited** from parent container
- Have conditions that use **tags and attributes**
- 4 Rule Types
  - RLS / CLM / GRANT / DENY

The screenshot shows the Databricks interface for creating a new rule. The page is titled 'New rule for Sales.FY2025' and is divided into several sections:

- General:** Includes fields for 'Name' (Rule for Sales.FY2025) and 'Description (optional)'. A note says 'Enter basic information for your policy.'
- Subject:** 'Who should this policy apply to?' with a dropdown menu set to 'Search users, groups, and service principals'.
- Applies to...:** 'This policy will affect the selected users, groups, or service principals.' with a dropdown menu set to 'Search users, groups, and service principals'.
- Except for...:** 'Exclude selected users, groups, or service principals from this policy. For example, a user from a group may be excluded.' with a dropdown menu set to 'Search users, groups, and service principals'.
- Type & scope:** 'What type of policy is this? What objects should it apply to?'. It has two radio buttons: 'Column mask' (unselected) and 'Row filter' (selected). Below it, 'Enforcement scope' is set to 'Choose objects on which to enforce this policy'.
- Function:** 'Choose a function for this policy to use and enter required parameters.' It has a dropdown for 'Main' > 'Sales' and another for 'fct\_tel\_filter'. Below is a 'Function definition' box containing SQL code:

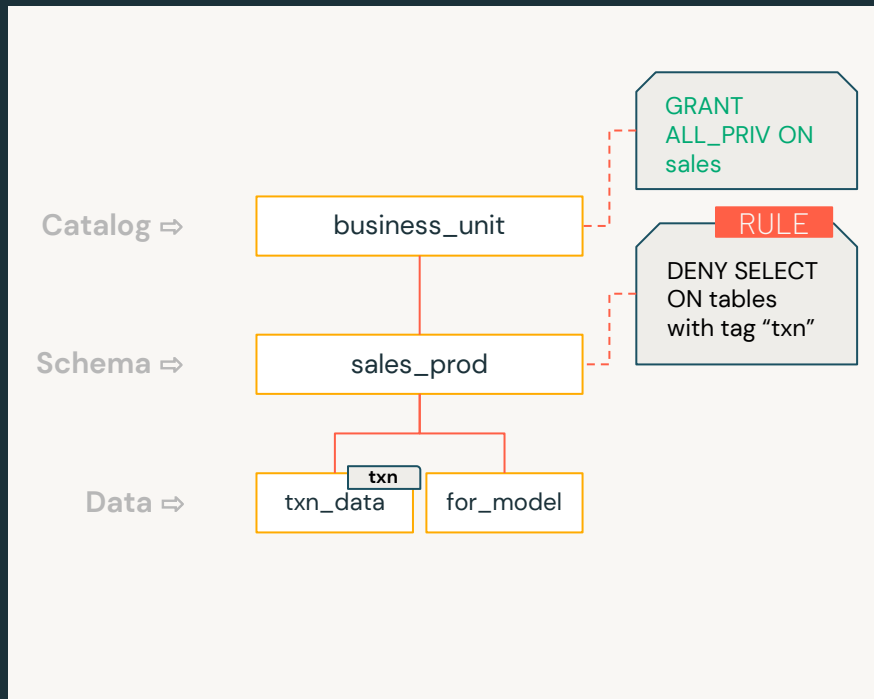
```
1 FUNCTION tel_filter(tbl_col STRING)
2 RETURN IF(NOT (tbl_col rlike '^(Q|I|S)-' ||
3         4), true, false);
```

At the bottom, there are 'Create rule' and 'Cancel' buttons.

# Inheritance

## Rules and Tags use inheritance to co-exist, simplify and add flexibility

- Rules inherit down the object hierarchy and co-exist with existing object grants.
- Hierarchy order is respected for Tags
- Denies are always respected.
- Tags from catalogs and schemas can be used in policies that apply directly to objects
  - Policies on catalogs/schemas can also use object tags.





# Meet our characters

Every organization story has its “characters”



**Sarita**

Data engineer responsible for building a forecast model

Wants to discover the right data, and access only what she needs in order to build her model



**Malik**

Data producer responsible for producing transaction data in Sarita’s organization

Wants access control on his data, but doesn’t want to re-implement it when access needs or rules change.



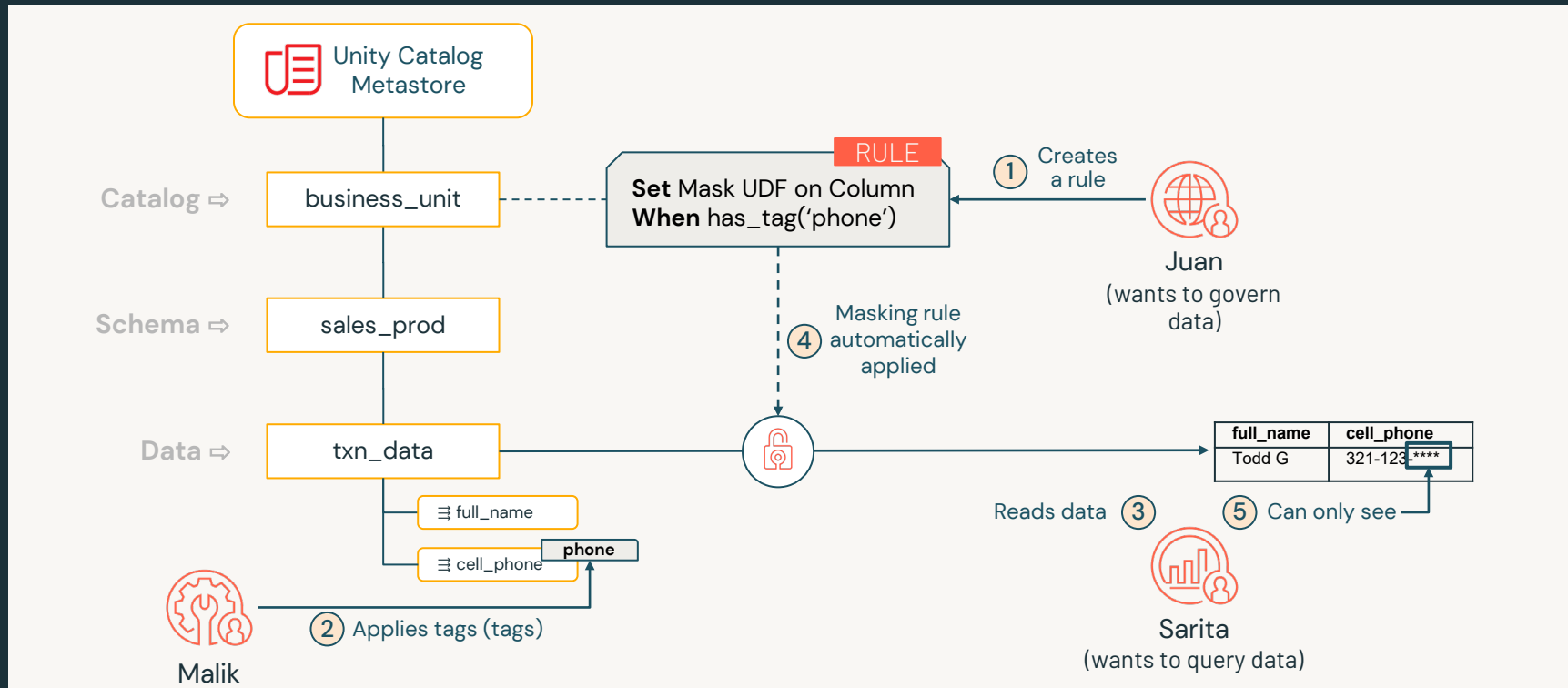
**Juan**

Data steward responsible for governing sales data within Sarita’s organization

Wants to meet compliance standards across all data sets and enforce proper data access automatically

# The plot thickens

## ABAC in action



# Demo



# Anatomy of a Rule

# Anatomy of a rule

“Follow” along please

```
SET RULE filter_by_state
ON CATALOG sales_prod
ROW FILTER state_lookup_filter
TO all_analysts EXCEPT analyst_managers
FOR TABLES
WHEN COLUMNS col_has_tag_value('pii', 'state')
```

- Applies a filter based on a lookup table to ensure that analysts can only see data for their assigned states

# More about rules – Name

All good things have good names.

```
SET RULE filter_by_state
ON CATALOG sales_prod
ROW FILTER state_lookup_filter
TO all_analysts EXCEPT analyst_managers
FOR TABLES
WHEN COLUMNS col_has_tag_value('pii', 'state')
```

- Names are unique within the assigned resource
- Names are how you select the rule for updates and deletes

# More about rules – Resource

Where your rule is applied in the hierarchy

```
SET RULE filter_by_state
ON CATALOG sales_prod
ROW FILTER state_lookup_filter
TO all_analysts EXCEPT analyst_managers
FOR TABLES
WHEN COLUMNS col_has_tag_value('pii', 'state')
```

- Rule is enforced on all objects at or “below” where it is applied

# More about rules – Type

Rule types have different syntax and behavior

```
SET RULE filter_by_state
ON CATALOG sales_prod
ROW FILTER state_lookup_filter
TO all_analysts EXCEPT analyst_managers
FOR TABLES
WHEN COLUMNS col_has_tag_value('pii', 'state')
```

- Row Filter / Column Mask / Grant / Deny
- Applied when conditions are met



# More about rules – Targets & Filters

Who does it apply to, and for what object type

```
SET RULE filter_by_state
ON CATALOG sales_prod
ROW FILTER state_lookup_filter
TO all_analysts EXCEPT analyst_managers
FOR TABLES
WHEN COLUMNS col_has_tag_value('pii', 'state')
```

- Principals targeted or excluded
- Object types targeted

# More about rules – Conditions

## WHEN does the rule apply?

```
SET RULE filter_by_state
ON CATALOG sales_prod
ROW FILTER state_lookup_filter
TO all_analysts EXCEPT analyst_managers
FOR TABLES
WHEN COLUMNS col_has_tag_value('pii', 'state')
```

- Condition is applied to the object being accessed.
- Evaluates true/false
- Can use attributes from parent objects, user, and request

# Use cases

# Time-bound data access

## Enforce policy that automatically expires

Sarita is a data engineer who needs access to transaction tables for a 3 month project



Juan, the data steward, creates an ABAC policy with a time bound expiration of 3 months

```
SET RULE grant_expire_3mo
ON SCHEMA sales
GRANT SELECT
TO `data_engineering`
ON TABLES WHEN
current_date() < '2024-09-13'
```

# Region based data access

## Geo-fence your data



Sarita, the data engineer, is based out of America and wants to access a dataset stored in Europe.

---



Malik, the data producer, wants to ensure that engineers can access data for their use cases



Juan the data steward, wants to make sure that GDPR constraints are not violated and creates a row filter to ensure that Sarita can only see data for America.

# Region based data access

## Geo-fence your data

### Lookup Table

region	group_name
emea	EMEA_USERS
amer	AMER_USERS
apj	APJ_USERS

### UDF

```
CREATE FUNCTION gdpr_fxn(region STRING)

RETURN
  region IN (SELECT DISTINCT region from
sales.region_mapping WHERE
is_account_group_member(group_name))
```

### Rule

```
SET RULE
gdpr_filter
ON SCHEMA house.sales
ROW_FILTER gdpr_fxn
TO data_engineering
FOR TABLES
WHEN COLUMNS
  (col_has_tag('region'))
```

# Distributed ownership

## Broad policy coexists with functional access



Malik, the data producer wants to provide analysts access to all sales transactions.

```
SET RULE analyst_sales_select
ON CATALOG business_unit
GRANT SELECT
TO `analysts`
FOR TABLES
WHEN TABLE has_tag('txn')
```



Julie, the CDO, wants to make sure that all PII information is masked for back office users.

```
SET RULE analyst_sales_mask
ON CATALOG business_unit
COLUMN MASK mask_pii
TO `analysts`
FOR TABLES
WHEN has_tag('txn')
WHEN COLUMNS col_has_tag('pii')
```

# Restrict default privileges

## Limit what a user can do by default



Carlton, the platform admin, wants to make sure that users do not have the ability to give other users access to data assets that they create.

```
SET RULE restrict_manage_tables
ON CATALOG business_unit
DENY MANAGE_ACCESS
TO ALL EXCEPT `admins`
FOR TABLES
```



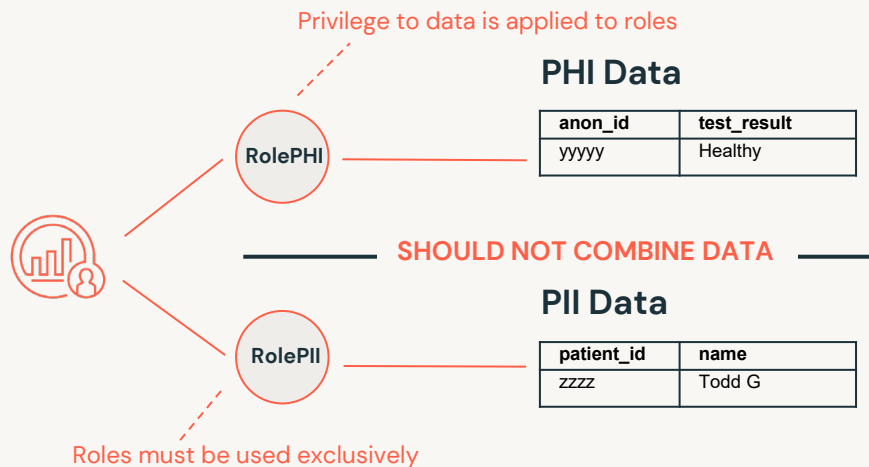
# Exclusive access to data

<Set Role> – can be combined with ABAC or used by itself

Malik, the data producer wants to ensure that PHI cannot be combined with PII data to limit the risk of re-identification.

Data is stored separately but analysts have access to both PHI and PII information for different projects.

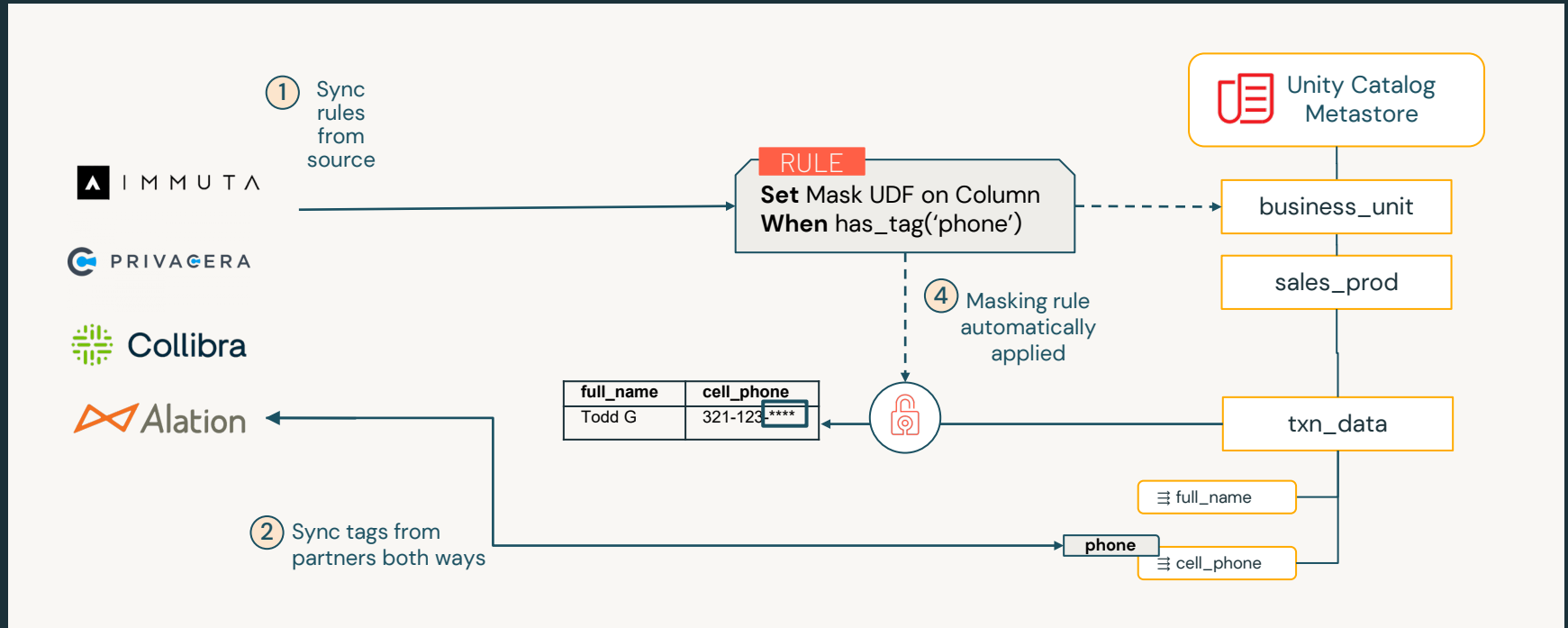
Analysts should not query or combine data together.



```
<SET ROLE RolePHI>
SELECT * FROM prd_phi.app.results
<SET ROLE RolePII>
SELECT * FROM prd_phi.app.patients
```

# ABAC improves UC Integrations

Express policy via our launch partners, secure it with Unity Catalog



Q/A



# Learn more at the summit!



Databricks  
Events App



## Tells us what you think

- We kindly request your valuable feedback on this session.
- Please take a moment to rate and share your thoughts about it.
- You can conveniently provide your feedback and rating through the **Mobile App**.



## What to do next?

- Discover more related sessions in the mobile app!
- Visit the Demo Booth: Experience innovation firsthand!
- More Activities: Engage and connect further at the Databricks Zone!



## Get trained and certified

- Visit the Learning Hub Experience at **Moscone West, 2nd Floor!**
- Take complimentary certification at the event; come by the Certified Lounge
- Visit our Databricks Learning website for more training, courses and workshops! [databricks.com/learn](https://databricks.com/learn)

