

Governing Your Lakehouse: Data Quality, Lineage, and Security in Microsoft Fabric

Includes: Full presentation and Cheat Sheets



COMMUNITY SUMMIT

Microsoft Fabric

October 11–15, 2026
Gaylord Opryland Resort



The largest independent Microsoft AI & Business Applications User Conference on the planet.



Governing Your Lakehouse

Data Quality, Lineage, and Security in Microsoft Fabric

Data | Technical (Development) – Advanced (301)
Oct. 14, 2026 | 8:00 - 9:30 am | Volunteer Mezzanine 10/11, Level M

@ 2026 Dynamic Communities

Your Presenter



Chris Smith | Weidenhammer

Connect with me on LinkedIn ([linkedin.com/in/chris-smith-9b65701](https://www.linkedin.com/in/chris-smith-9b65701)) or at www.hammer.net

- Data & analytics consultant helping businesses turn everyday operational data into trusted reporting.
- Works daily with Power BI, Excel, dataflows, and the Microsoft data platforms.
- Focused on making modern tools approachable for business users, not just engineers.

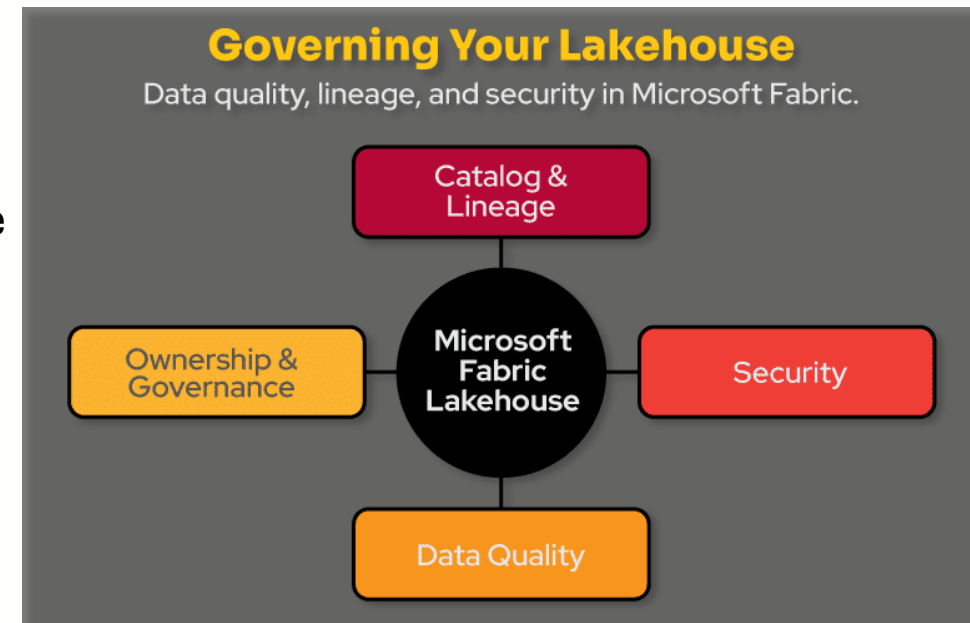


Session Agenda

- The governance layer most teams skip
- Part 1 – Catalog, lineage & classification
- Part 2 – Workspace & domain security, RLS, dev/test/prod
- Part 3 – Automated data-quality monitoring & alerting
- Change management to make governance sustainable
- Takeaways, bonus tips, and Q&A



Microsoft Fabric



| Learning Objectives



1. Modern data governance concepts and how to use Purview and / or Fabric to catalog lakehouse assets, track data lineage from source systems to Power BI, and classify sensitive data.
2. Understand workspace and domain security models that enforce least privilege, row-level security, and dev/test/prod deployment for source systems.
3. The importance of automated data-quality monitoring – freshness, row-count, and schema-drift checks – with Fabric notebooks and alerting.

Where Are You Today?



Quick show of hands – who has:

- A Fabric lakehouse already in production?
- Have a clear corporate standardization on where all data used for insights is sourced and how all metrics should be calculated?
- Use a data governance platform like Microsoft Purview?
- Row-level security enforced on OneLake data?
- Automated data-quality checks running today?

Be honest – most teams skip the last three.

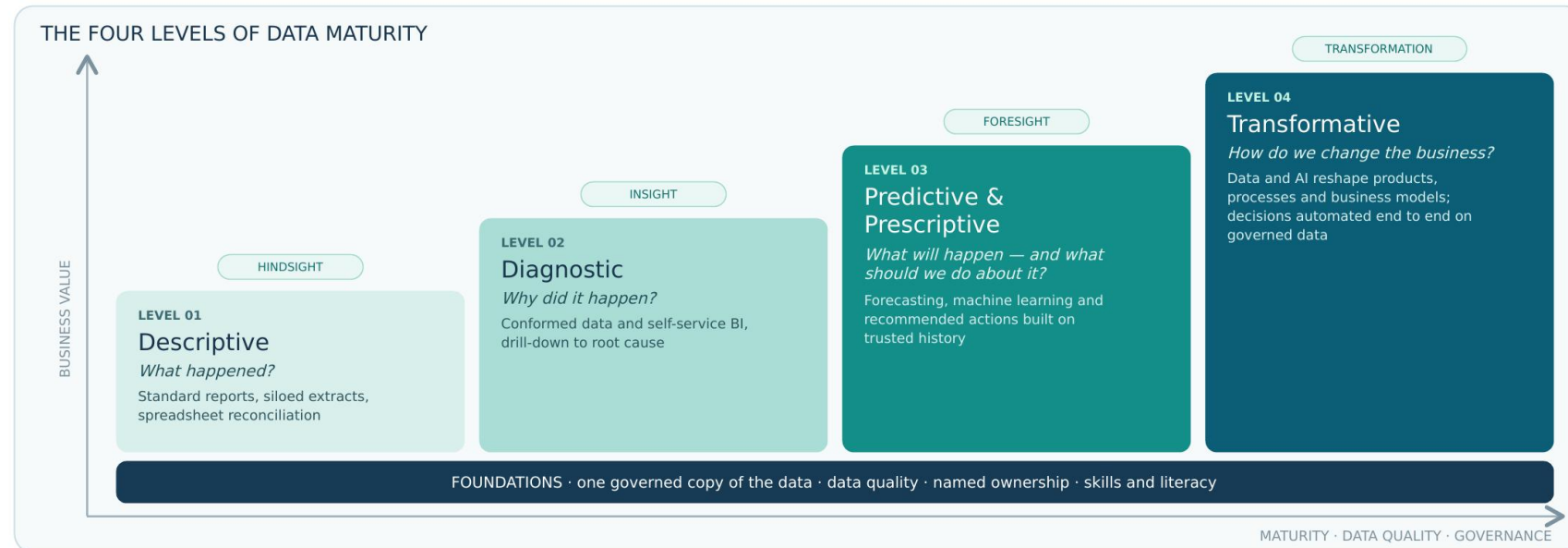


Data Estate Maturity



Why mature your data estate

Each level of maturity answers a more valuable question — and each one stands on the level beneath it.



WHY IT IS WORTH THE CLIMB

DECISIONS YOU CAN TRUST

One governed version of the numbers ends reconciliation meetings and arguments over whose report is right.

LOWER COST PER QUESTION

Modeled, reusable data means each new question costs a query, not a project.

RISK AND COMPLIANCE

Known ownership, classification and lineage turn audits from fire drills into reports you already have.

READY FOR AI

Copilots, agents and models are only as good as the governed data beneath them. Immature data makes AI wrong, faster.

You cannot skip a level — every step stands on the one below it.

Prediction and AI fail on data that cannot yet answer “what happened” reliably. Invest in the foundations first.

The Easy Part vs. the Real Challenge



- Standing up a lakehouse takes an afternoon.

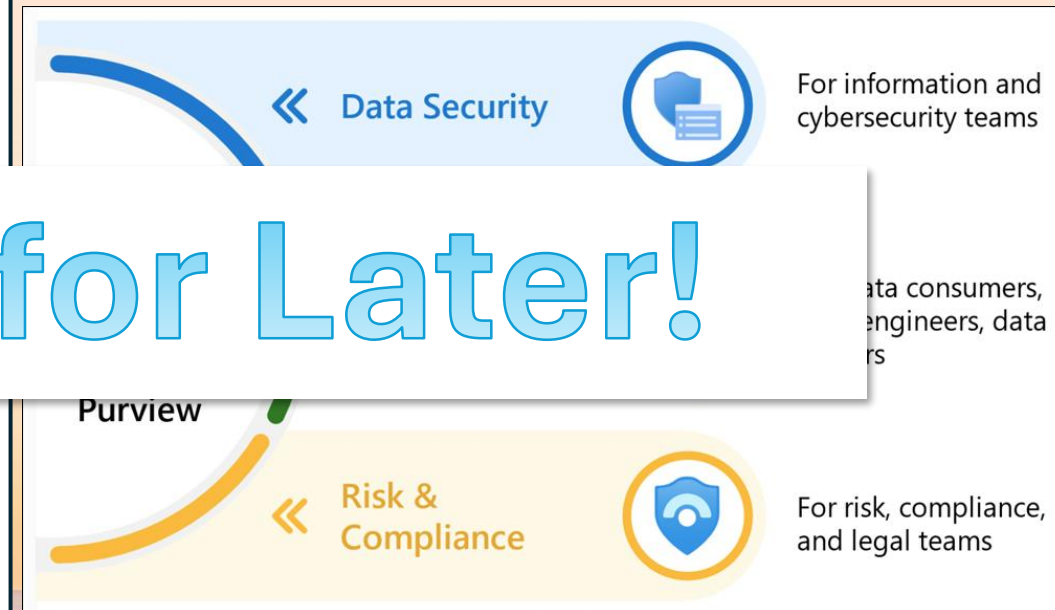
Keeping it trustworthy takes governance.

• (f

Do Not Defer for Later!

- Which version is correct?
- Who can see what?
- Data freshness / frequency?
- Lineage?

This is the layer most implementations defer until there's a problem.



The Governance Layer Most Teams Skip



Four disciplines sit on top of the lakehouse:

- Catalog & lineage – find and trust the data.
- Security – control who sees what.
- Data quality – prove it is correct and fresh.
- Change management – keep it all sustainable.

Today we go deep on each.



Change Management & Ownership

Catalog & Lineage

find & trust
data

Security

who sees
what

Data Quality

is it correct
& fresh?

**OneLake Lakehouse
Dynamics 365 data**

Standing it up is easy.

Keeping it trustworthy is governance.

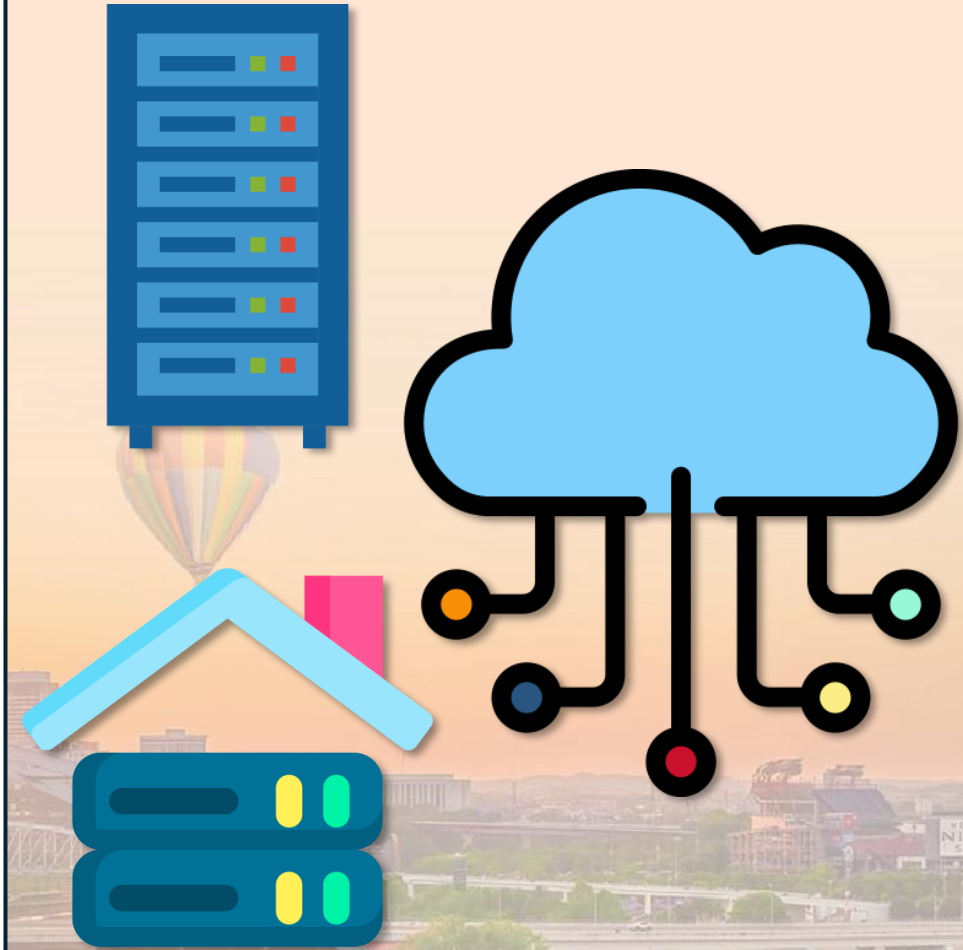
@ 2026 Dynamic Communities

The Governance Layer Most Teams Skip



Operational Source Systems:

- Differ in how data is exposed.
- New, Cloud, On-Premises mainframe:
- **Same governance model applies to ALL**

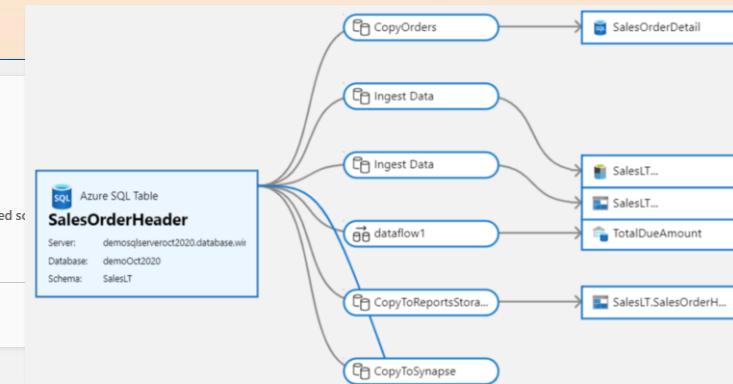
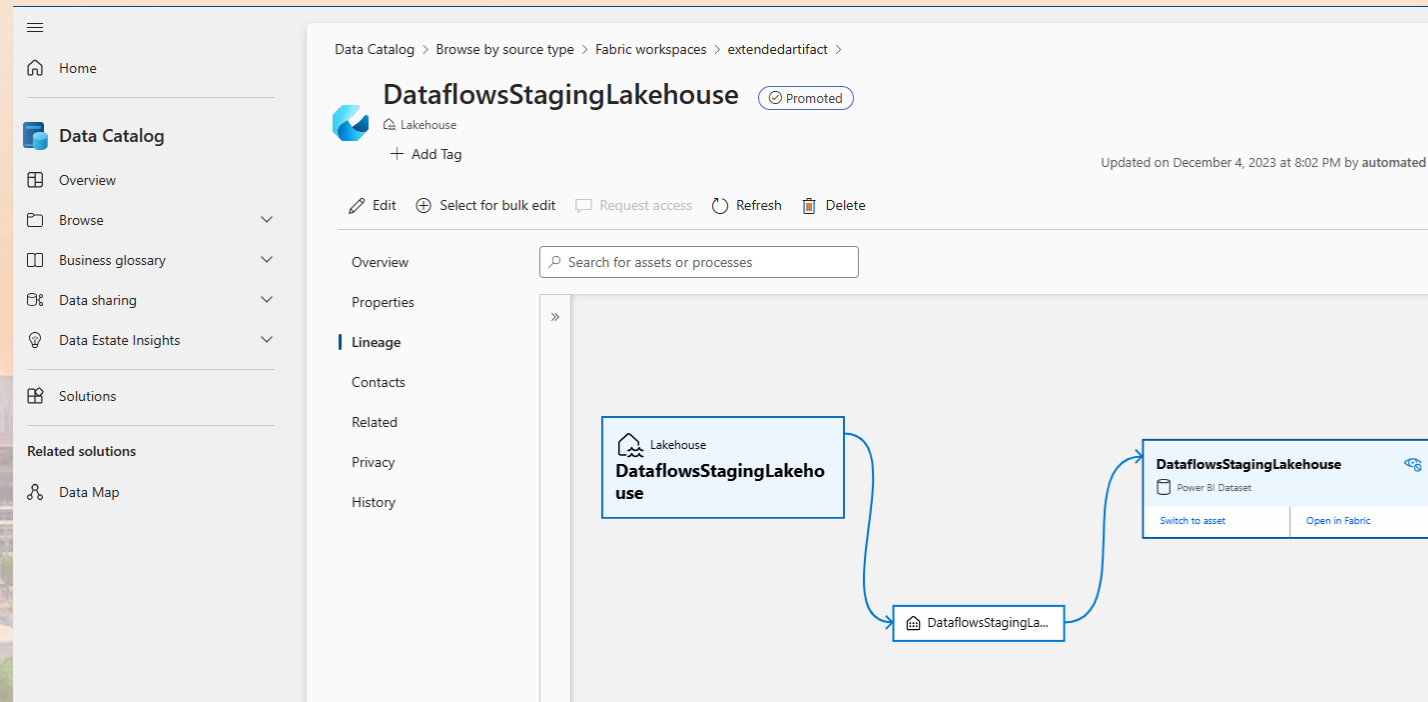


Catalog, Lineage & Classification



Learning Objective 1

Purview + Fabric: catalog assets, trace D365-to-report lineage, and classify sensitive data.



@ 2026 Dynamic Communities

Importance of a Corporate Data Catalog



Why every organization needs a corporate data catalog

A catalog is the searchable, governed inventory of what data exists, what it means, who owns it and whether to trust it.

THE QUESTIONS IT ANSWERS

Why it matters

"Where is the data?"

Find trusted data in minutes, not by asking around

"Which number is right?"

One glossary definition, and certified sources

"Who owns this?"

A named owner and steward on every asset

"Is it sensitive?"

Classification travels with the data

"What breaks if it changes?"

Lineage shows the impact before the change

EIGHT BUILDING BLOCKS

What a data catalog comprises

01 INVENTORY

Every table, file, model and report, discovered by automated scanning

02 GLOSSARY

Agreed business definitions — "revenue," "active customer" — linked to data

03 OWNERSHIP

A data owner and steward named on each asset, by domain

04 CLASSIFICATION

Sensitivity labels and detected sensitive data such as personal information

05 LINEAGE

Where data comes from, where it goes, and what a change would break

06 QUALITY

Quality scores, freshness and certification, so users know what to trust

07 ACCESS

Who can see it today, and a clear path to request access

08 USAGE + CONTEXT

Documentation, popularity and related assets that help people choose well

EXAMPLE · ONE CATALOG ENTRY



dim_customer

Sales domain > Gold lakehouse > dim_customer

CERTIFIED

CONFIDENTIAL

GOLD LAYER

One row per customer, conformed from Dynamics 365 Sales — the authoritative customer dimension for sales reporting.

OWNER

VP, Sales Operations

STEWARD

Sales data steward

GLOSSARY TERMS

Customer · Active customer

SOURCE SYSTEM

Dynamics 365 Sales

REFRESHED

Daily 06:00 — on time

QUALITY SCORE

97%

Request access

Viewer access granted through the Sales app

LINEAGE



COLUMNS

NAME	TYPE	CLASSIFICATION
customer_id	key	—
customer_name	text	—
email	text	PERSONAL
credit_limit	decimal	CONFIDENTIAL
region	text	—

Used by 14 reports · 42 viewers this month · related: dim_account, fact_opportunity

Illustrative entry — names and figures are examples.

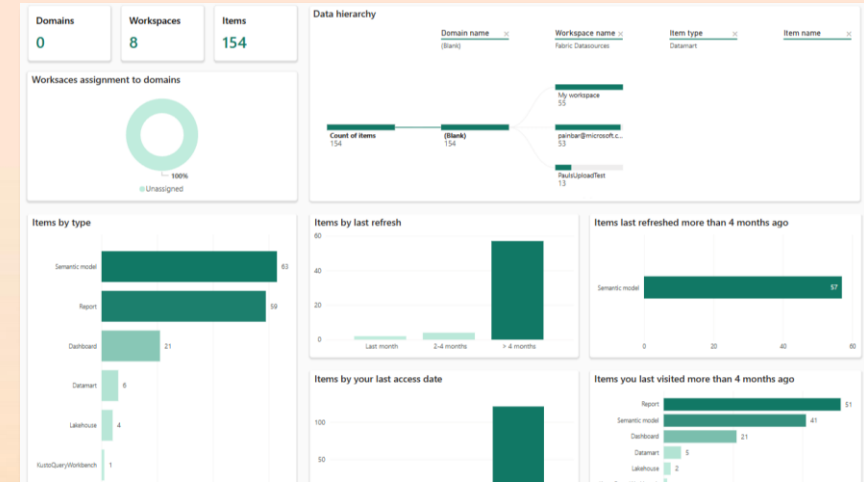
If people cannot find it, trust it, or tell who owns it, the data might as well not exist.

A catalog turns a scattered data estate into a shared, governed asset — build it first, not last.

Why Purview + Fabric?



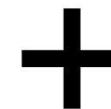
- **Fabric** surfaces items; **Purview** governs them enterprise-wide.
- One catalog spanning **D365**, **Fabric**, and beyond.
- Automated scanning discovers and classifies assets.
- Live sync: **Fabric** items flow into the **Purview** data map.



Governance travels with the data, not the tool.



Microsoft Purview



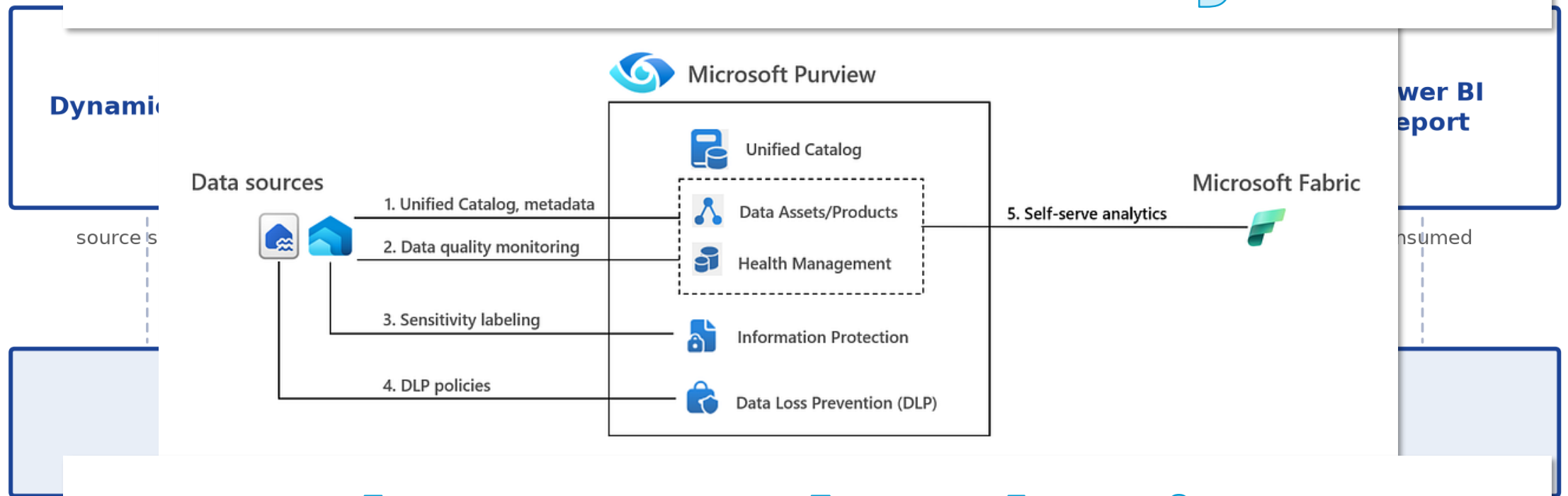
Microsoft Fabric

End-to-End Lineage: D365 to Power BI



Purview stitches lineage across every hop – source to report.

Root –Cause Analysis



Impact Analysis

unities

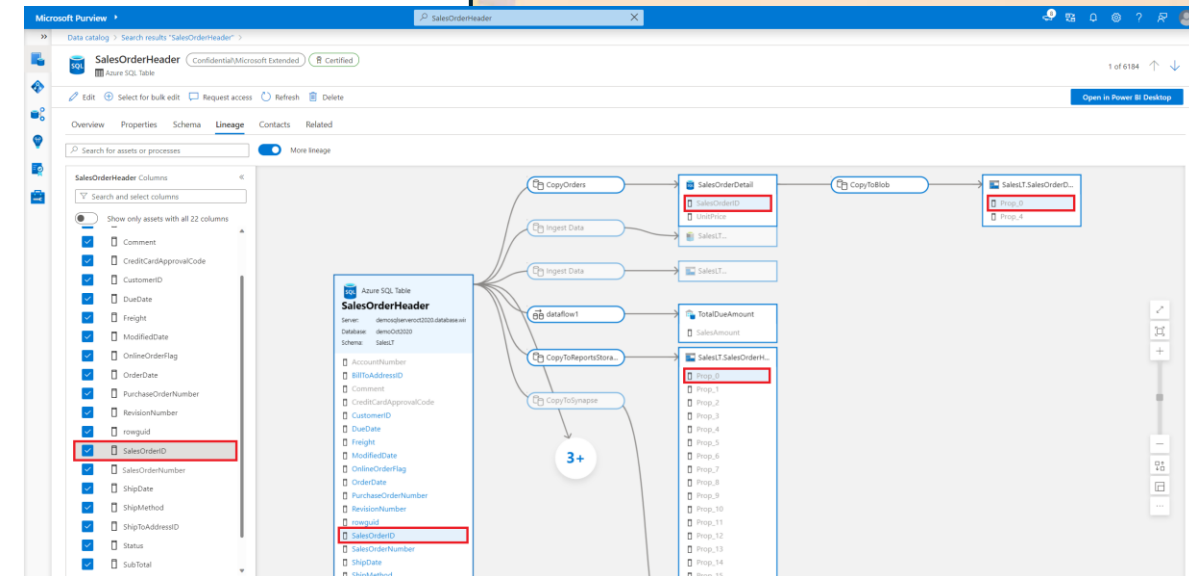
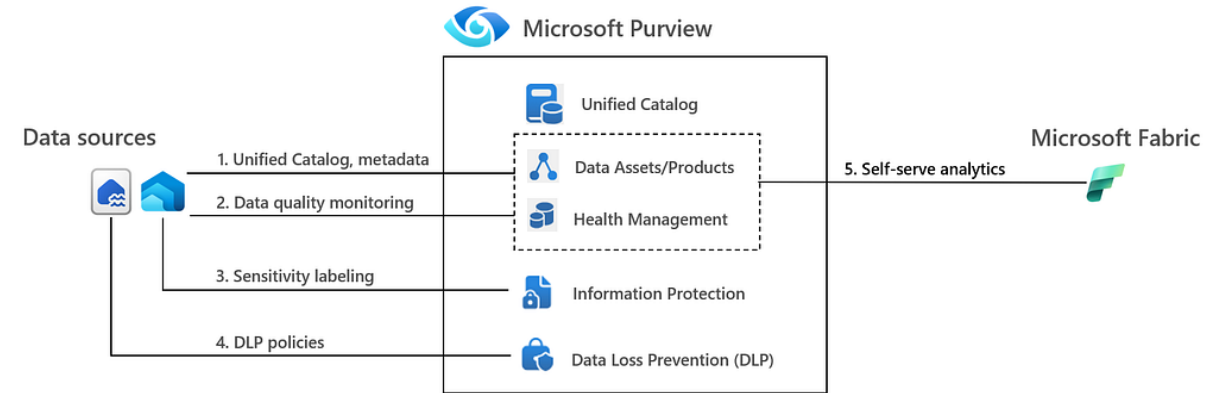
Cataloging Lakehouse Assets / Business Glossary



- **Establish a Corporate Data Catalog:**

1. Register the **Fabric** tenant as a **Purview** data source.
2. Scan lakehouses, warehouses, and semantic models on a schedule.
3. Enrich with business glossary terms and owners.
4. Certify and endorse trusted datasets so users know what to use.

Goal: no more “which table is the real one?”

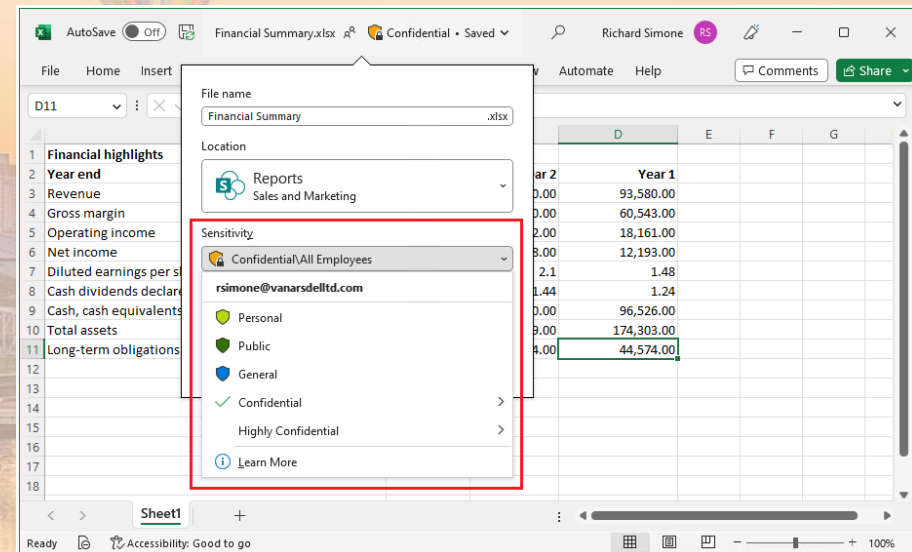
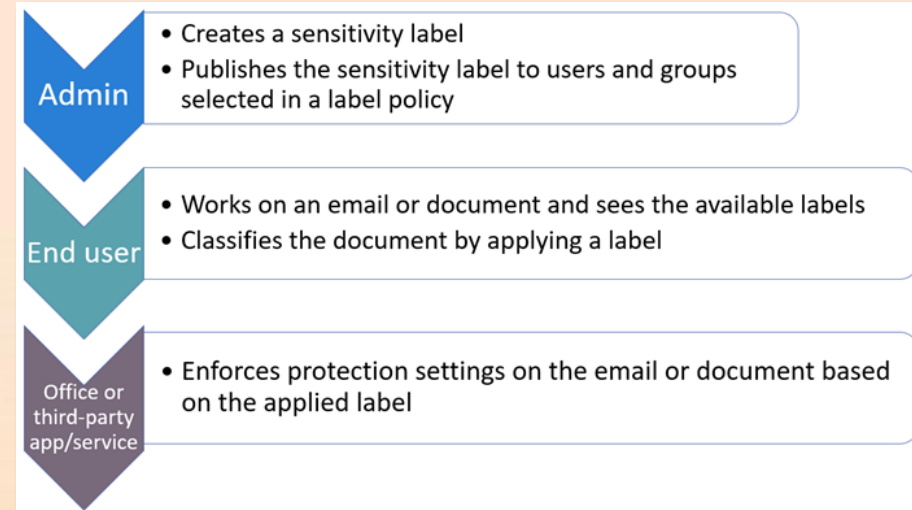


Classifying Sensitive Data



- Apply Microsoft Information Protection sensitivity labels.
- Auto-classify PII from D365 (names, emails, financials).
- Labels persist into Power BI exports and downstream copies.
- Pair with DLP policies to block risky sharing.

Classify once at the source – enforce everywhere.

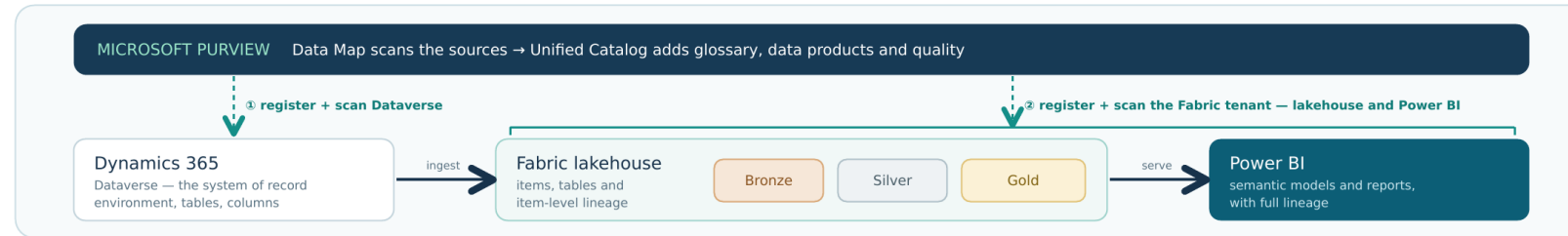


Data Catalog Setup



Standing up a corporate data catalog in Purview

A workable sequence for data sourced in Dynamics 365, landed in a Fabric lakehouse and reported in Power BI.



PHASE A Prepare	PHASE B Connect and scan	PHASE C Enrich with meaning	PHASE D Establish trust and operate
1 Stand up Purview and domains Create governance domains and collections for the pilot; assign Data Source Admin, Data Reader and data quality steward roles; name owners.	3 Register and scan Dataverse D365 tables and columns become assets. Scope the scan to pilot tables; Dataverse supports full and scoped scans, not incremental.	6 Build the business glossary Define terms such as revenue and active customer with the business, then link them to the gold tables and models that implement them.	9 Add data quality rules Profile gold lakehouse tables and apply rules — freshness, completeness, uniqueness — scored up to the data product.
2 Create scanning identities A service principal in a security group allowed by Fabric admin API settings; a Dataverse application user with the Service Reader role.	4 Register and scan the Fabric tenant One scan brings lakehouse items and tables, Power BI semantic models and reports, plus lineage. Live view can bridge the gap in the meantime.	7 Classify and label Apply classifications and sensitivity labels, starting with D365 customer and financial tables, so protection follows the data.	10 Operate and expand Schedule rescans, review health reports, run access recertification, then onboard the next domain with the same sequence.
	5 Validate end-to-end lineage Trace D365 to lakehouse to model to report, and close manual gaps. Expect item-level lineage only for non-Power BI Fabric items.	8 Publish data products Group gold tables, models and reports into data products in Unified Catalog, each with an owner inside its governance domain.	Repeat per domain Sales first, then finance, then operations — on proven templates

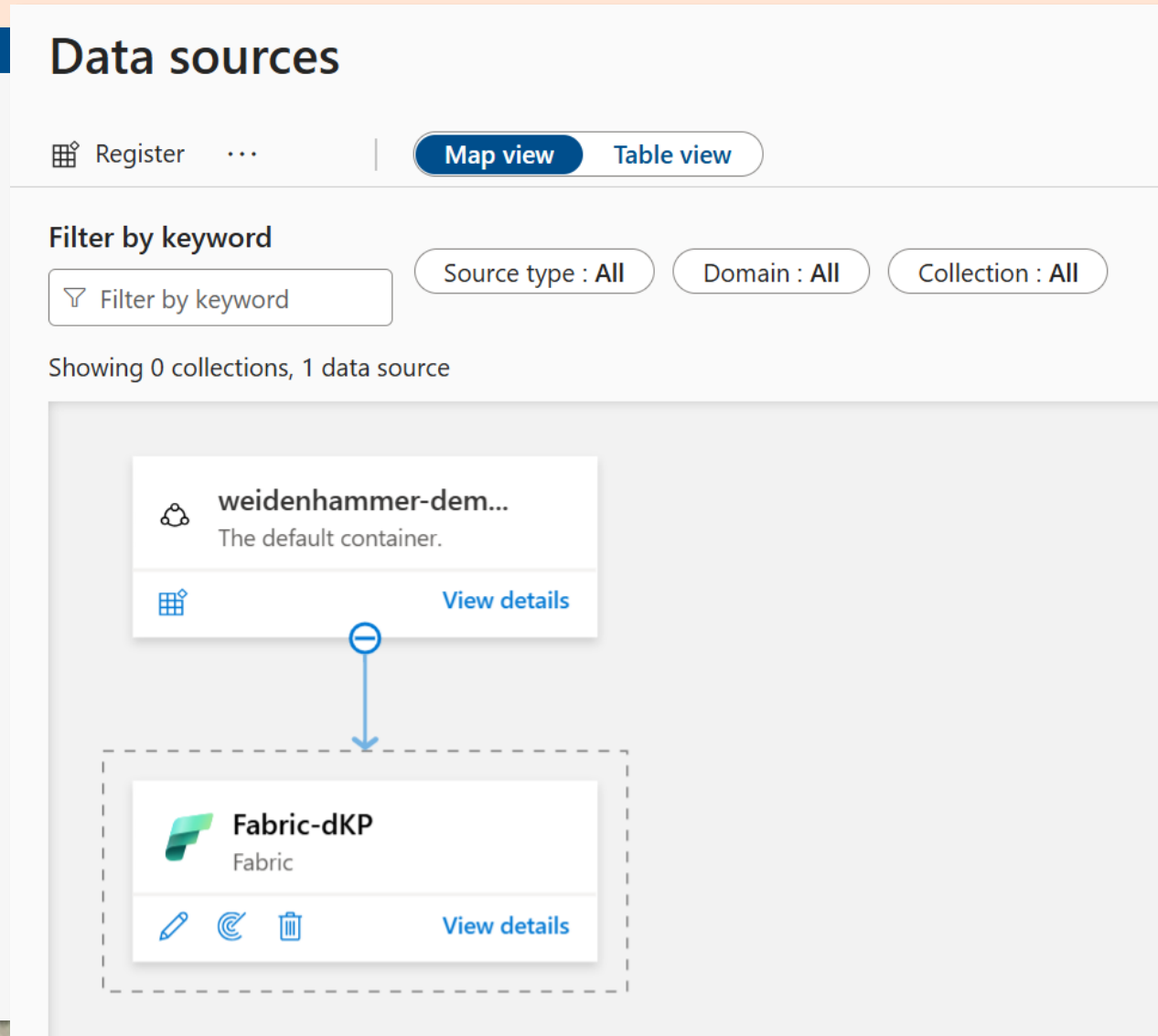
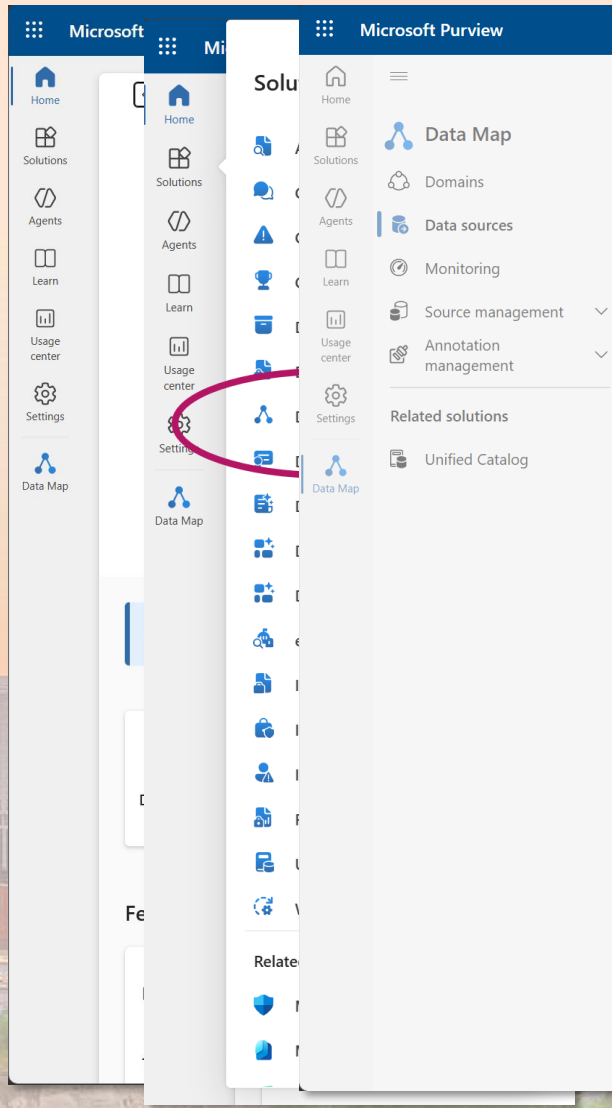
KNOW THE LIMITS BEFORE YOU PROMISE

Dataverse scans No incremental scan, and deleted tables are not removed from the catalog automatically.	Fabric lineage depth Non-Power BI items get item-level lineage; lakehouse tables get metadata, not lineage.	Live view is a preview Same tenant, item-level metadata only, and no classifications — register and scan instead.	Data quality region Sources and the Purview account must sit in the same Azure region.
---	---	---	--

Scan first, then add meaning, then earn trust — one domain at a time.
Technical metadata arrives in days; the glossary, ownership and quality that make a catalog useful take deliberate work.

Catalog, Lineage & Classification

Demo



Catalog, Lineage & Classification

Demo



A screenshot of the Microsoft Fabric Admin portal. The left sidebar shows the navigation menu with options like Home, New Group, Overview, All groups, Insights, Deleted, and Diagnose. The main content area displays the 'Admin portal' settings for 'Tenant settings'. The 'Enhance admin APIs responses with detailed metadata' setting is shown as 'Unapplied changes' and is currently 'Enabled'. Below this, the 'Apply to:' section has two radio buttons: 'The entire organization' and 'Specific security groups', with the latter being selected. A search bar below the radio buttons contains 'SecurityGroup11' and 'Fabric Purview'. At the bottom, there are 'Apply' and 'Cancel' buttons. The background of the slide shows a cityscape at sunset.

Catalog, Lineage & Classification

Demo



Scan "Fabric-dKP"

Name *
Scan-ZN0

Personal workspaces *
☒ Include ☐ Exclude

Credential *
Microsoft Purview MSI (system)

Domain *
weidenhammer-demo-purview

Collection *
Select domain only

All items in this data source

[Continue](#)

Set a scan trigger

Set a scan trigger to run the scan at specific times. If recurring, the scan will run at the specified time and every subsequent scan is initiated at the same time.

Time zone
(UTC) GMT-07:00

Recurring
Every 1 hour

Start recurring on
2026-10-08

[Continue](#) [Back](#)

Run detail

Data Map
View reports about the processes that populate the Data Map. [Learn more](#)

Domains
Create domains that match your org's key business segments as the top-level collections. With a collection, you can take action on all of its content at once.

weidenhammer-demo-purview (Default)
Domain

[+ New collection](#) [Edit](#) [Refresh](#)

Overview **Role assignments** Updated on October 8, 2026, 10:07 AM by Chris Smith

Description
The default container.

Assets
86

Data sources
1

Scans
1

[View other objects associated](#)

Created on
October 8, 2026, 10:07 AM by Chris Smith

[+ New domain \(Preview\)](#)

Catalog, Lineage & Classification

Demo



Unified Catalog

Overview

Discovery

Data products

Data assets

Enterprise glossary

Catalog management

Governance domains

Data products

Custom metadata

Requests

Classic types

Health management

Process automation

Requests and approvals

Related solutions

Data Map

Glossary terms

Personal Health

Unified Catalog > Governance domains > Glossary terms >

Epidemic

Glossary term

Edit

Unpublish

Refresh

Manage policies

Delete

Overview

Description

Custom attributes

Contacts (1)

Owners (1)

Experts (0)

Related

Data observability

No attributes to display

A countrywide or regional outbreak of a disease that is highly contagious and affects a large number of people over a short period of time.

CS

Chris Smith

Data owner

No contacts

Parent term

Select...

Next

Manage access policies

Policy Configuration

Manage policies that are automatically added when the glossary term is linked to a data product. [Learn more about access policies](#)

Access policy approval requirements

☒ Manager approval required

Digital Attestations

☒ Permit data copies

+ Add attestation

Catalog, Lineage & Classification

Demo



The screenshot displays the 'Unified Catalog' application interface. On the left is a sidebar with navigation options: Overview, Discovery, Data products, Data assets, Enterprise glossary, Catalog management, Governance domains, Data products (selected), Custom metadata, Requests, Classic types, Health management, Process automation, Requests and approvals, Related solutions, and Data Map. The main area shows the 'Find and select' dialog. At the top, there's a search bar and filters for 'Account : weidenhammer-demo-purview', 'Data source type : All', and 'Data: Database'. Below this, a list of data sources is shown, including 'SalesDb', 'mySampleDatabase', and 'AdventureWorks', all identified as 'Azure SQL Database'. Each entry includes a checkbox, the name, the source type, a description, the fully qualified name, and a status indicating it was 'Retrieved from source'. At the bottom of the dialog are 'Save' and 'Cancel' buttons. The background of the slide shows a cityscape at sunset.

Catalog, Lineage & Classification

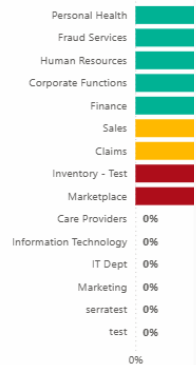


Demo

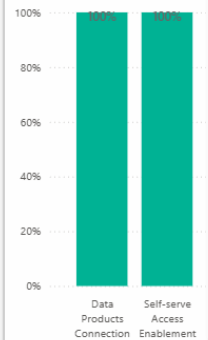
Data governance

Governance score
77.0%

Governance Score by Business Domain



Governance Score by Data Product

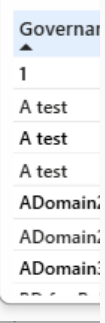


of Governance

Trend of



Governance

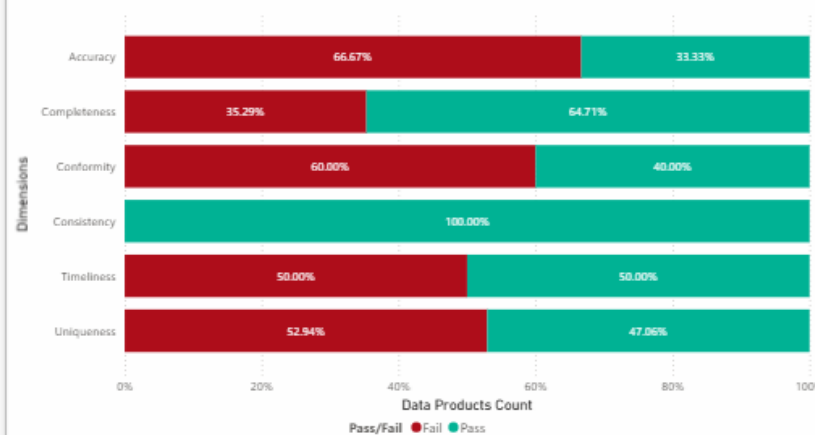


Score > 80% Score between 60% and 80%

Data quality



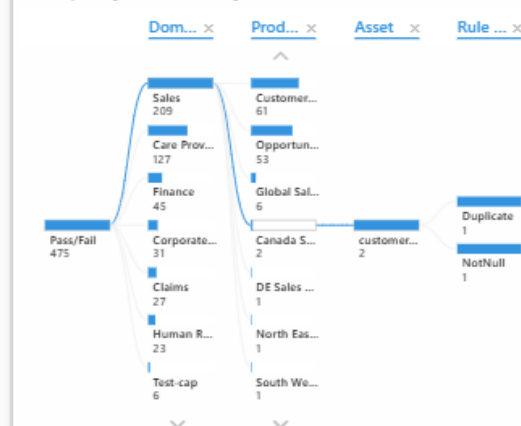
Pass/fail quality ratio by data product



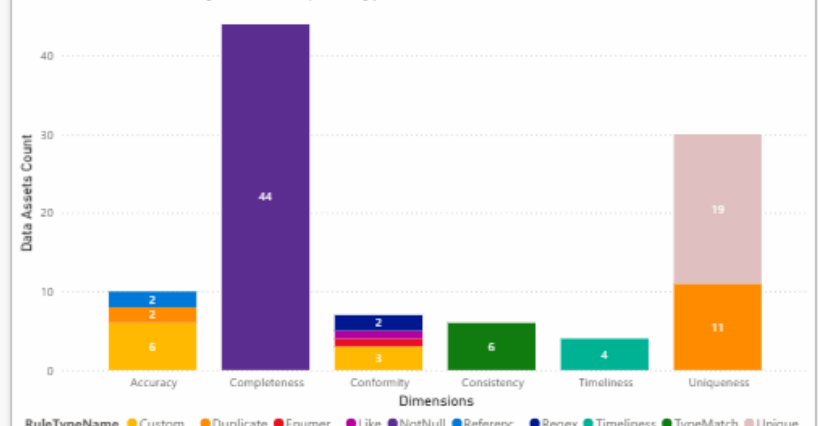
Data quality status by dimension

Domain	Accuracy	Completeness	Conformity	Consistency	Timeliness	Uniqueness	Overall
Care Providers		90.35%					
Claims	0.00%	94.75%	14.44%			0.00%	
Corporate Functions		99.35%				79.38%	
Finance		98.23%		100.00%		66.72%	
Fraud Services		100.00%				100.00%	
Human Resources		84.34%				2.68%	
Personal Health		100.00%		100.00%			
Sales	70.95%	92.87%	75.46%	100.00%	50.00%	78.09%	
Test-cap		89.98%				0.00%	

Data quality rule hierarchy



Count of data assets by dimension, rule type



Purview + Fabric vs Fabric Built-In Governance

Purview with Fabric: what's included vs. what you add

Fabric governs its own estate out of the box. Purview extends protection and governance across everything else — and carries its own licensing.

INCLUDED WITH YOUR FABRIC LICENSE Fabric's built-in governance

- OneLake catalog — discover and explore everything in your Fabric estate
- Governance insights and recommended actions, including the security insights that used to live in the Purview hub
- Domains and subdomains for federated ownership, plus custom tags
- Endorsement — promoted and certified items
- Lineage across Fabric items, source to report, and impact analysis
- Workspace, item and OneLake permissions

Scope stops at the Fabric tenant — enough for one team, not for an enterprise estate

REQUIRES MICROSOFT PURVIEW LICENSING What Purview adds

- Information Protection — sensitivity labels and protection policies
- Data loss prevention for lakehouses, warehouses, databases and semantic models
- Purview Audit — every Fabric user activity in the audit log
- Insider Risk Management, with ready-made Fabric indicators
- Unified Catalog and Data Map — one catalog spanning D365, Azure, on-premises and third-party systems, with glossary and data quality
- Governance for Fabric Copilots and agents

Separate licensing; non-Microsoft 365 sources are billed pay-as-you-go



Microsoft Purview



Microsoft Fabric

WHERE THE LINE FALLS	FABRIC, BUILT IN	PURVIEW, LICENSED
Scope	Your Fabric tenant	Fabric, D365, Azure, on-premises, third-party systems
Catalog	OneLake catalog	Unified Catalog, backed by the Data Map
Lineage	Across Fabric items, source to report	Across systems, beyond the workspace boundary
Classification	Manual tags and endorsement	Automated scanning and sensitive-data detection
Enforcement	Workspace and item permissions	Sensitivity labels, protection policies, DLP actions
Cost	Included with Fabric	Purview licensing, plus consumption for non-M365

HOW FABRIC DATA REACHES THE PURVIEW CATALOG

LIVE VIEW (PREVIEW)

Same-tenant workspaces and items appear in the catalog with no scan to configure. Full, item-level metadata only, and no classifications are applied.

REGISTER AND SCAN

Register the Fabric tenant in the Data Map. Full, incremental and scoped scans bring item metadata and lineage, and enable classification.

WHAT YOU STILL SET UP

A service principal, a security group registered in the Fabric tenant admin API settings, and a runtime. Lineage below item level is not supported.

Governance travels with the data, not with the tool.

Fabric's own catalog is enough for one team. What makes it enterprise governance is licensed separately — so budget for it deliberately.

Recap – You need a Corporate Data Catalog



Why every organization needs a corporate data catalog

A catalog is the searchable, governed inventory of what data exists, what it means, who owns it and whether to trust it.

THE QUESTIONS IT ANSWERS

Why it matters

"Where is the data?"

Find trusted data in minutes, not by asking around

"Which number is right?"

One glossary definition, and certified sources

"Who owns this?"

A named owner and steward on every asset

"Is it sensitive?"

Classification travels with the data

"What breaks if it changes?"

Lineage shows the impact before the change

EIGHT BUILDING BLOCKS

What a data catalog comprises

01 INVENTORY

Every table, file, model and report, discovered by automated scanning

02 GLOSSARY

Agreed business definitions — "revenue," "active customer" — linked to data

03 OWNERSHIP

A data owner and steward named on each asset, by domain

04 CLASSIFICATION

Sensitivity labels and detected sensitive data such as personal information

05 LINEAGE

Where data comes from, where it goes, and what a change would break

06 QUALITY

Quality scores, freshness and certification, so users know what to trust

07 ACCESS

Who can see it today, and a clear path to request access

08 USAGE + CONTEXT

Documentation, popularity and related assets that help people choose well

EXAMPLE · ONE CATALOG ENTRY



dim_customer

Sales domain > Gold lakehouse > dim_customer

CERTIFIED

CONFIDENTIAL

GOLD LAYER

One row per customer, conformed from Dynamics 365 Sales — the authoritative customer dimension for sales reporting.

OWNER

VP, Sales Operations

STEWARD

Sales data steward

GLOSSARY TERMS

Customer · Active customer

SOURCE SYSTEM

Dynamics 365 Sales

REFRESHED

Daily 06:00 — on time

QUALITY SCORE

97%

[Request access](#)

Viewer access granted through the Sales app

LINEAGE



COLUMNS

NAME	TYPE	CLASSIFICATION
customer_id	key	—
customer_name	text	—
email	text	PERSONAL
credit_limit	decimal	CONFIDENTIAL
region	text	—

Used by 14 reports · 42 viewers this month · related: dim_account, fact_opportunity

Illustrative entry — names and figures are examples.

If people cannot find it, trust it, or tell who owns it, the data might as well not exist.

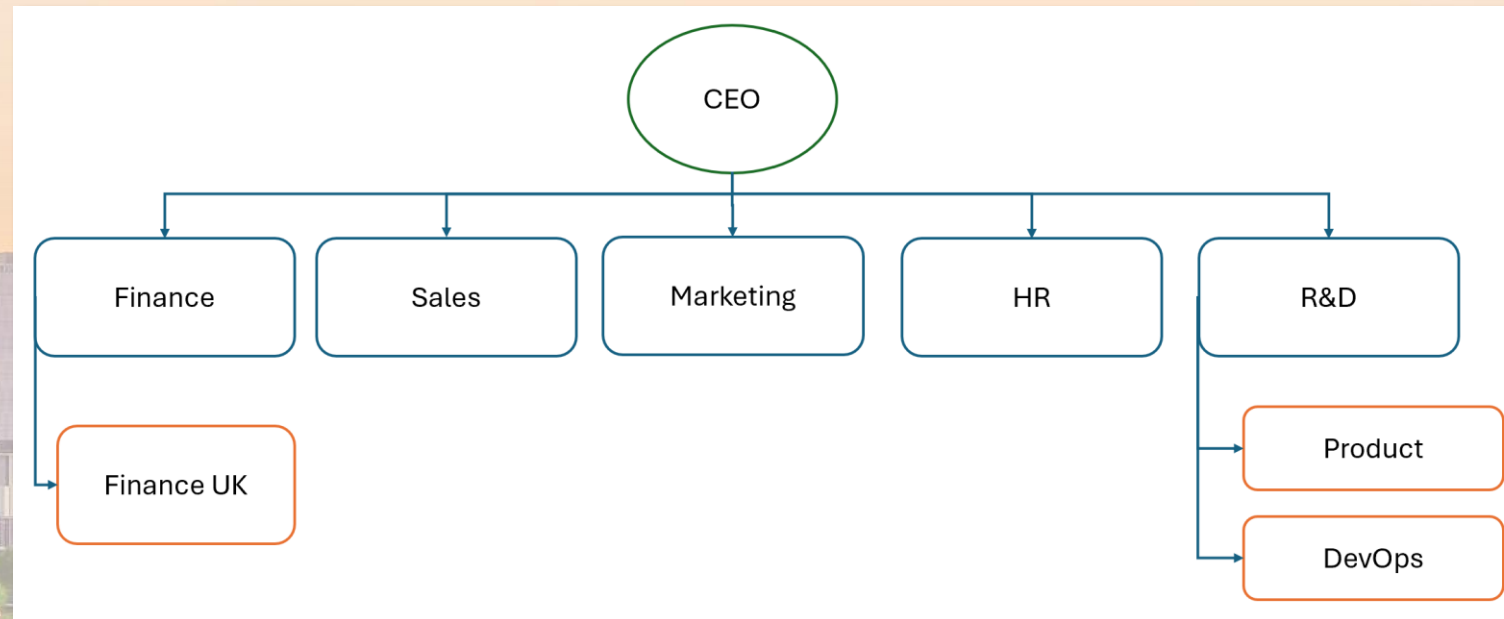
A catalog turns a scattered data estate into a shared, governed asset — build it first, not last.

Designing the Security Model



Learning Objective 2

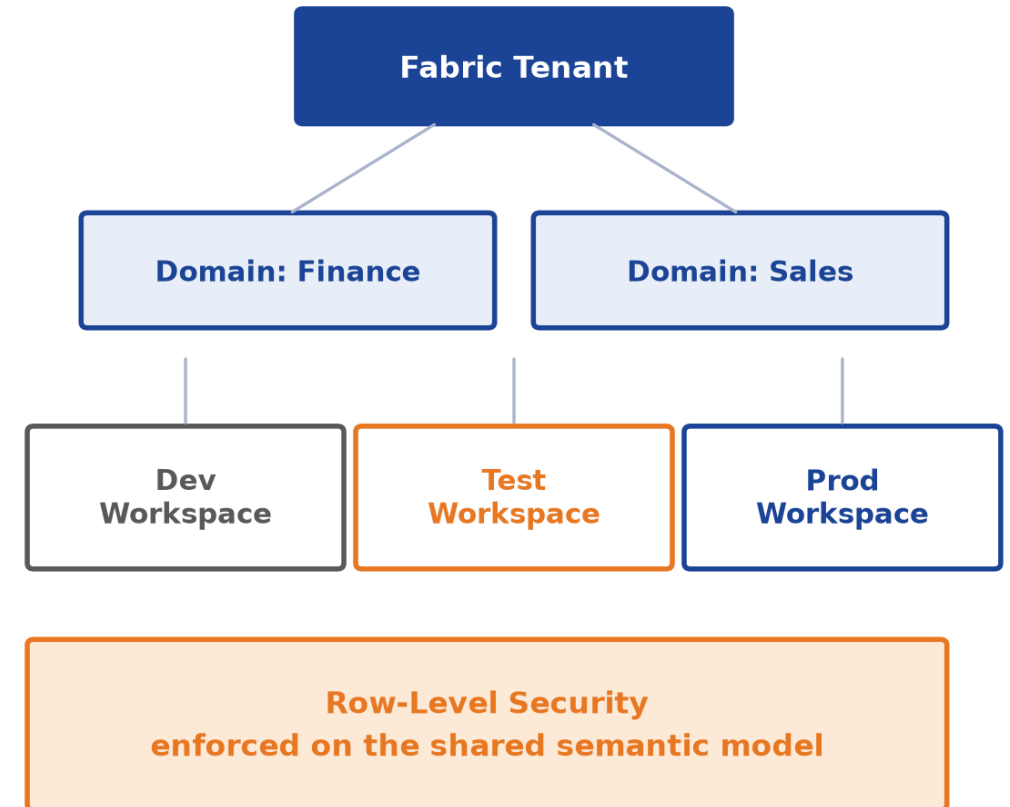
Workspaces, domains, least privilege, row-level security, and dev/test/prod separation.



Workspaces & Domains

- Domains group data by business area (Finance, Sales).
- Workspaces are the unit of access and deployment.
- Assign roles to groups – Admin, Member, Contributor, Viewer.
- Never grant access to individuals; always to security groups.

Structure first – it is painful to refactor later.



Least privilege · environment separation · RLS

Access by group and role – never by individual

Least-Privilege Access Patterns



- Default to Viewer; grant elevation only by need.
- Separate build access (Contributor) from data consumer access (Viewer).
- Use OneLake data-access roles for granular table control.
- Review access quarterly; automate joiner/mover/leaver.



Role	Capability	Appropriate for
Admin	Full control including access management and deletion	A small, named platform team
Member	Add others, publish, and modify most content	Senior owners of the domain
Contributor	Create and edit content, cannot manage access	Developers and analysts who build
Viewer	Read published content only	<u>The large majority of consumers</u>

Every extra permission is future attack surface.

Row-Level Security across OneLake



Row-level and object-level security

One model, many audiences — each sees only its own rows, columns and tables.

01 · SEMANTIC MODEL Row-level security

Which rows can this user see?

Legal entity, region, business unit, cost centre

- Define roles on the model, never per report
- Report-level filters are not a security control — any user who can edit the report can remove them
- Use dynamic RLS: one DAX filter matched to the signed-in user principal name, so role count stays constant as the organization grows
- Keep the user-to-entity mapping in a data table, not in DAX — access changes become a data load

02 · SEMANTIC MODEL Column and object level

Which fields and tables can they see?

Row filtering is not always sufficient

- Column-level security restricts the field itself — salary, national identifiers, margin
- Object-level security hides entire tables and measures, so they never appear in the field list
- Use these instead of maintaining parallel models for different audiences
- Duplicate models multiply both cost and inconsistency — avoid the workaround

03 · UNDERNEATH THE MODEL OneLake data access

What can they reach in the lakehouse?

Workspace roles alone are coarse

- A Viewer on a workspace can generally read the lakehouse tables within it
- OneLake data access roles add table- and folder-level control inside the lakehouse itself
- Enforced regardless of the engine used to query
- Pair model RLS with OneLake roles wherever users could reach the data directly

TWO WAYS IN — ONLY ONE IS COVERED BY MODEL SECURITY

THE INTENDED PATH



THE PATH PEOPLE FORGET



PUT THESE IN THE DEFINITION OF DONE

Roles live on the model, not on the report

Dynamic RLS keyed to the user principal name

Mapping maintained in data, not in DAX

Every role validated with "View as role"

OneLake roles wherever direct access is possible

Model security governs the report. OneLake security governs the data.

Where a user can query the lakehouse directly, semantic-model RLS is not a boundary — it is a convenience. You need both layers.

Dev / Test / Prod Separation



- One workspace per environment; promote with deployment pipelines.
- Parameterize connections so prod never points at dev data.
- Gate promotion with approvals and automated rules.
- Keep source system credentials scoped per environment.

No more “it worked in my workspace.”

The screenshot displays a Databricks workspace interface with three environments: Development, Test, and Production. The Test environment shows a successful deployment on 07/30/25 at 5:47 PM. The Production environment shows a successful deployment on 01/27/25 at 10:26 AM. Below the environments, a 'Deploy from' dropdown is set to 'Test', and a 'Flat-List' toggle is enabled. A table lists items across environments, with the 'Location' column highlighted by a blue box.

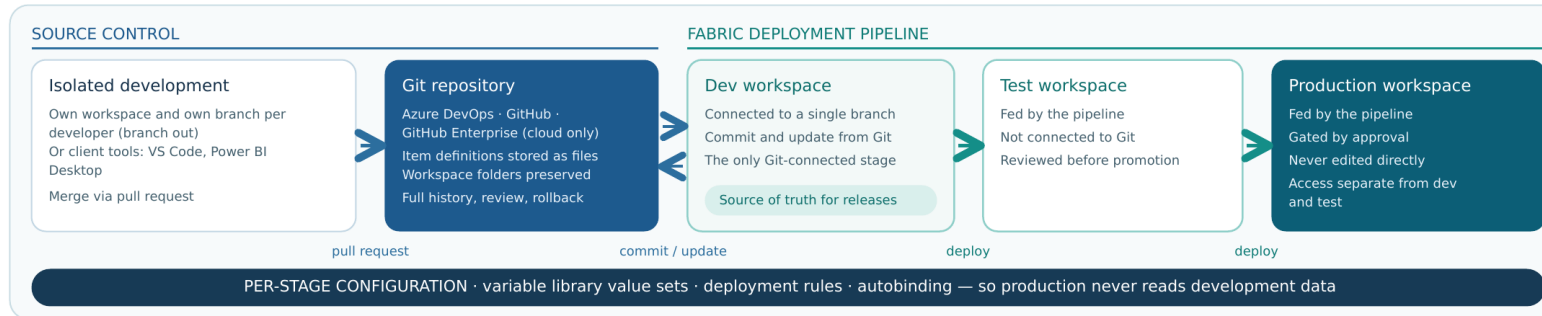
Selected stage item	Location	Type	Compared to source	Source stage item
HO	Demo WS [Dev]/Other	HomeOne (Preview)	—	—
LH_wVarLib	Demo WS [Dev]/Case 1 - NB w VarLib in Configure	Lakehouse (Preview)	—	—
LH_wVarLib	Demo WS [Dev]/Case 1 - NB w VarLib in Configure	SQL analytics endpoint	—	—
LHwVarLib	Demo WS [Dev]	Lakehouse (Preview)	—	—
LHwVarLib	Demo WS [Dev]	SQL analytics endpoint	—	—
NB - VarLib in configure	Demo WS [Dev]/Case 1 - NB w VarLib in Configure	Notebook	—	—
NB_vars_ExtItemRef	Demo WS [Dev]/Case 3 - NB w VarLib intellisense and Ext It...	Variable library (Preview)	—	—
NB_vars_ItemRef_sameWS	Demo WS [Dev]/Case 2 - NB w VarLib intellisense and same...	Variable library (Preview)	—	—
NButils - ItemRef Ext	Demo WS [Dev]/Case 3 - NB w VarLib intellisense and Ext It...	Notebook	—	—
NButils - ItemRef sameWS	Demo WS [Dev]/Case 2 - NB w VarLib intellisense and same...	Notebook	—	—
pipeline A	Demo WS [Dev]/Other/Pipelines	Data pipeline	—	—

Dev / Test / Prod Git Repo Integration



Git repos with Fabric deployment pipelines

Git versions the definitions. The pipeline promotes them. Connect Git to development only.



CONTINUOUS INTEGRATION

What Git integration gives you

- Connection is per workspace, and a workspace syncs to exactly one branch
- Item definitions are stored as files, so changes are reviewed in pull requests and revertible to any commit
- Workspace folder structure is preserved in the repo
- A shared workspace is a shared runtime — give each developer a branched workspace instead

CONTINUOUS DELIVERY

What the pipeline gives you

- Promotes items between the dev, test and prod workspaces without re-importing anything
- Deployment rules and autobinding re-point data sources per stage, so the same artifact just works
- Compare stages before deploying, and gate promotion with approvals
- Restrict who may deploy to production

SIX RULES THAT KEEP THE TWO IN STEP

ONE BRANCH PER WORKSPACE

A workspace syncs to a single branch. Developers branch out to their own workspace to work.

GIT ON DEV ONLY

Test and prod are fed by the pipeline, not by Git — one source of truth per stage.

CONFIGURATION AS CODE

Variable library value sets per stage; deployment rules and autobinding re-point the sources.

NEVER EDIT A HIGHER STAGE

Every change starts in a branch and flows forward. A direct edit in production is drift.

PERMISSIONS ARE SEPARATE

Repo access, pipeline access and workspace roles are three different grants. Plan all three.

KNOW WHAT GIT DOES NOT HOLD

Definitions are versioned; lakehouse data is not. Unsupported items are ignored, not synced.

Connect the developer workspace to Git, then deploy from it with a pipeline.

Git answers “what changed and who approved it.” The pipeline answers “how it reaches production.” Together: no more “it worked in my workspace.”

Automated Data-Quality Monitoring



Learning Objective 3

Freshness, row-count, and schema-drift checks in Fabric notebooks – with alerting.



What to Monitor

- **Freshness** – did today's D365 data load actually land?
- **Row counts** – volumes within expected bounds?
- **Schema drift** – did a column change or disappear?
- **Referential checks** – orphaned keys after a sync?
- **Uniqueness** – is there a duplicate record?
- **Business rules** – can the data exist like this?

Catch it before the report does.



Implementing Checks with Fabric Notebooks



Fabric notebooks: the natural home for data quality

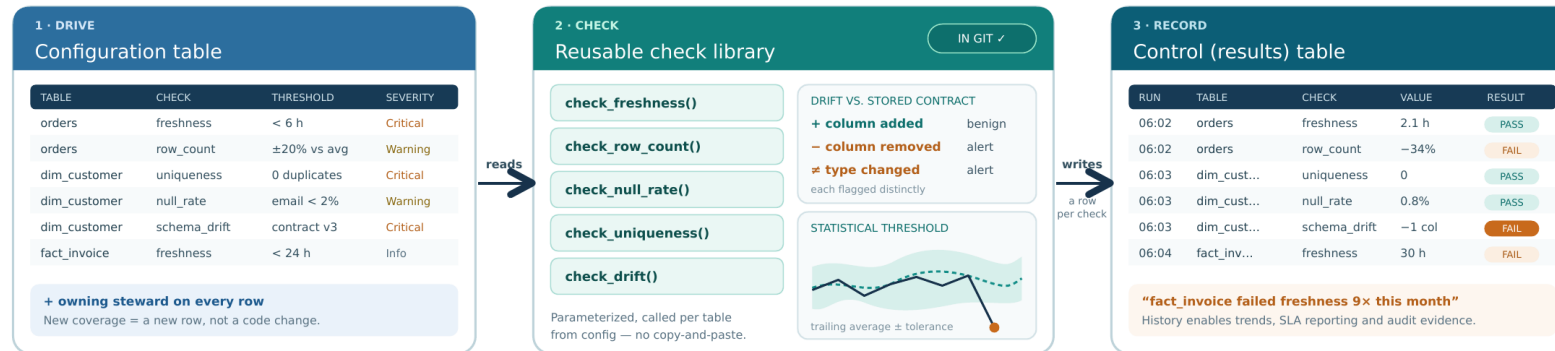
Quality logic that runs where the data lives — configured, versioned, recorded and wired into the load.

✓ Runs inside the platform

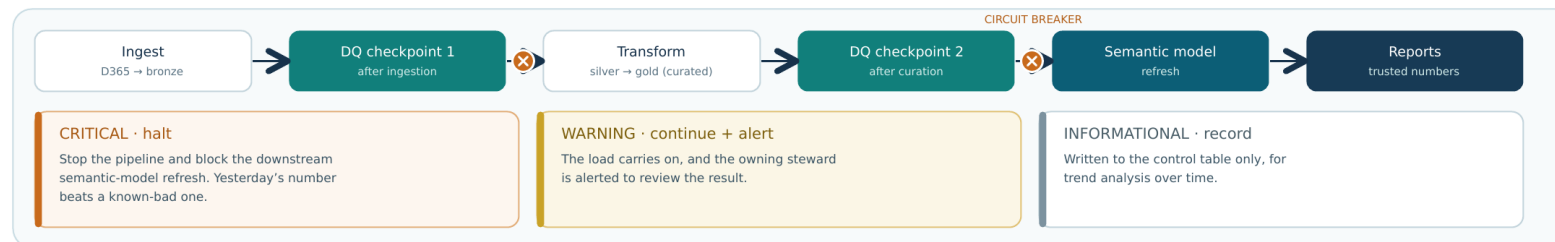
✓ Against lakehouse tables

✓ On the same capacity

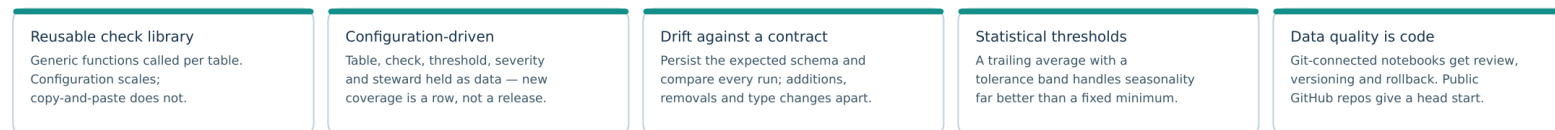
✓ No external infrastructure



PLACEMENT · CHECKS RUN AS PIPELINE ACTIVITIES, NOT A SEPARATE PROCESS



DESIGN GUIDANCE FROM PRODUCTION IMPLEMENTATIONS



Fail loudly, fail early — a circuit breaker on the data pipeline.

It is what stops a quality issue from becoming a credibility issue. Without persisted results, you have alerts but no measurement.

PySpark



Alerting & Data Activator



- Trigger Data Activator on threshold breaches.
- Route alerts to Teams or email for the owning data steward.
- Surface a data-quality scorecard in Power BI.
- Publish freshness status so consumers self-serve trust.
- Periodic reconciliation against the operational system

Silence is not confidence – make health visible.



@ 2026 Dynamic Communities

Purview – Data Quality Rules

Data quality: Purview rules vs. Fabric notebook checks

One measures the data and publishes a score. The other decides whether today's load is allowed through.

GOVERNANCE PLANE · LOW-CODE TO NO-CODE

Purview data quality rules

Rules configured in Unified Catalog, applied per column, run as a scan, and aggregated into scores for assets, data products and domains.

AUTHORING

Out-of-the-box rules plus AI-suggested rules; no notebook required

RULE CATALOG

Freshness, unique values, empty fields, data type and format match, duplicate rows, table lookup, plus a custom condition expression

WHO OWNS IT

Data quality steward to author; reader role to view

THRESHOLDS

Configurable score thresholds per rule, asset and data product

WHEN IT RUNS

Profiling first, then manual or scheduled scans

Estate-wide score and stewardship

It cannot stop a pipeline

DATA PLANE · QUALITY AS CODE

Fabric notebook checks

PySpark or SQL checks running inside the pipeline, against lakehouse tables, on the same capacity, with no external infrastructure.

AUTHORING

Reusable parameterized functions, driven from a configuration table

CHECK TYPES

Freshness, row-count deviation, null rate, uniqueness, referential integrity, and schema drift against a stored contract

WHO OWNS IT

Data engineer, with the owning steward named in configuration

THRESHOLDS

Any logic — a tolerance band around a trailing average

WHEN IT RUNS

A pipeline activity, after ingestion and after curation

Stops a bad load reaching reports

You build and maintain it



Microsoft Purview



Microsoft Fabric

CONSIDERATION	PURVIEW RULES	NOTEBOOK CHECKS
Primary question	How good is this data product?	Is today's load safe to publish?
Can it block a load?	No — it measures, scores and notifies	Yes — severity tiers halt the pipeline
Custom logic depth	Fixed rule set plus a custom expression	Unlimited; any Spark or SQL logic
Evidence and history	Built-in rule history and health reports	Your control table, retained as you choose
Versioning	Managed inside the service	Git-reviewed, versioned, roll back anytime
Reach	The whole estate, across clouds	Only the tables you wrote checks for

WHAT THE NOTEBOOK DOES WITH A FAILURE — DEFINE THE TIERS

CRITICAL

Halt the pipeline; block the model refresh

WARNING

Continue, but alert the owning steward

INFORMATIONAL

Record only, for trend analysis

Not a choice — Purview scores the data, the notebook enforces it.

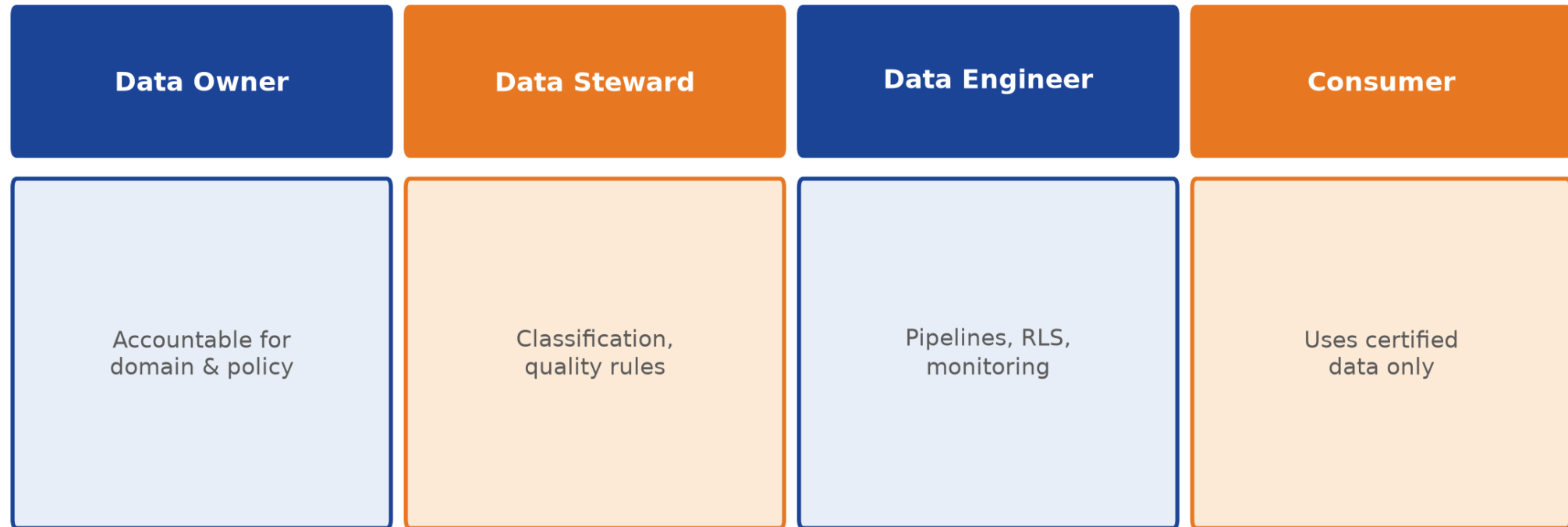
Publishing a known-bad number is worse than publishing yesterday's number, and only the in-pipeline check can prevent it.

Change Management That Sticks



Tools do not govern – people and roles do.

- Name a Data Owner and Steward for every domain.
- Bake governance into definition-of-done, not a side project.



Governance sticks only when ownership is explicit

Governance Recap



Governance as definition of done

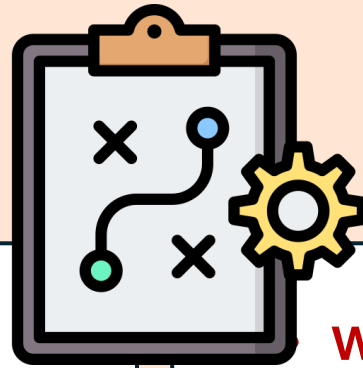
Governance fails when treated as a separate project. It succeeds when embedded in the delivery process.

A workable definition of done for any new data product
Seven conditions, checked at review — before the artifact is promoted to production.

REQUIREMENT	DISCIPLINE	ACCOUNTABLE
✓ 01 Catalogued and described, with a named owner and steward	CATALOG	Data Steward
✓ 02 Sensitivity classification applied	CLASSIFICATION	Data Steward
✓ 03 Access granted to security groups, with RLS implemented and tested where required	SECURITY	Data Engineer
✓ 04 Quality checks for freshness, volume and schema drift, with alerts routed to the steward	DATA QUALITY	Data Engineer
✓ 05 Lineage resolves end to end, with no manual gaps	LINEAGE	Data Engineer
✓ 06 In source control and promoted through the deployment pipeline	CHANGE CONTROL	Data Engineer
✓ 07 Semantic model endorsed, if it is intended to be authoritative	ENDORSEMENT	Data Owner

A checklist enforced at review is worth more than a policy document nobody reads.
Put these conditions in the delivery process itself, so governance is how work gets finished rather than a separate initiative.

Adoption Tactics



• What to do:

- Start with one domain
- Make the governed path the easy path
- Publish the scorecard
- Celebrate a catch
- Review quarterly

What NOT to do:

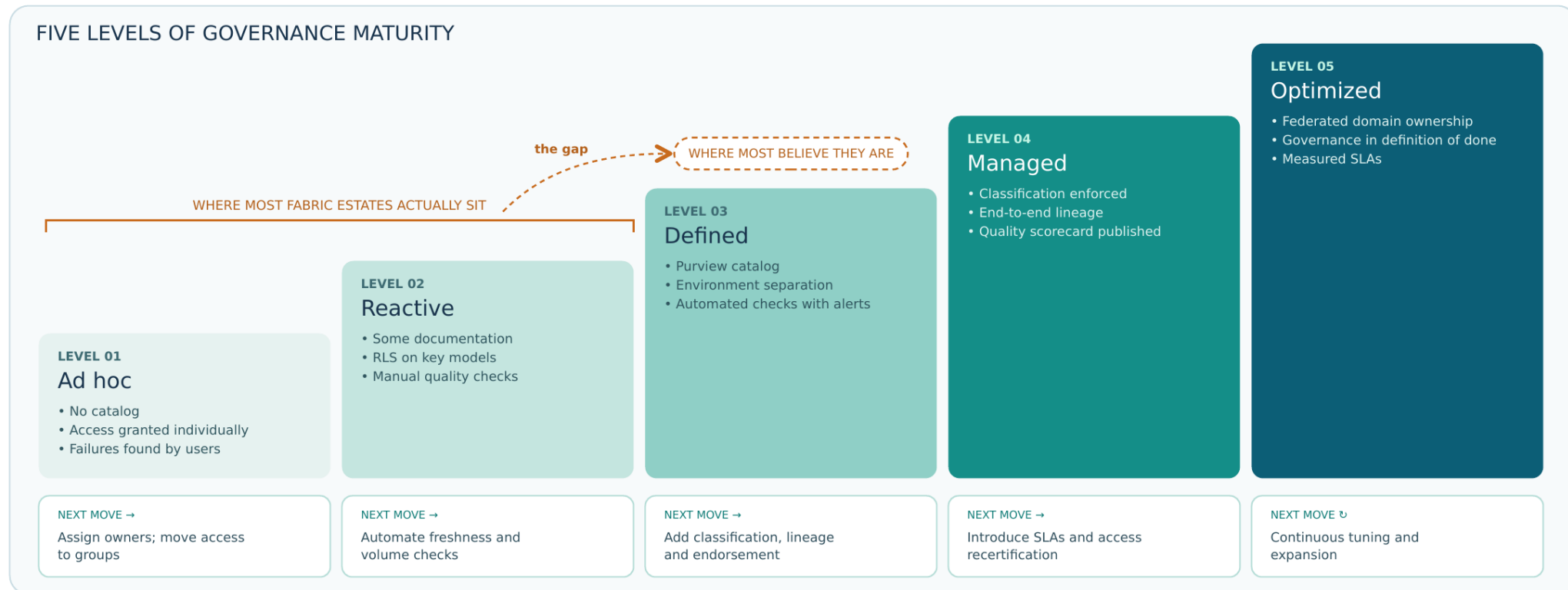
- Governance as phase two
- Report-level filtering as security
- Permissions granted to individuals
- One workspace for everything
- Bespoke quality code per table
- Alerts to a shared mailbox
- Classification applied at the report layer
- Automation running as a person
- Boiling the ocean
- Measuring nothing

Governance Maturity



A governance maturity model

Five levels, each with a clear next move. The hard part is not knowing the levels — it is admitting which one you are on.



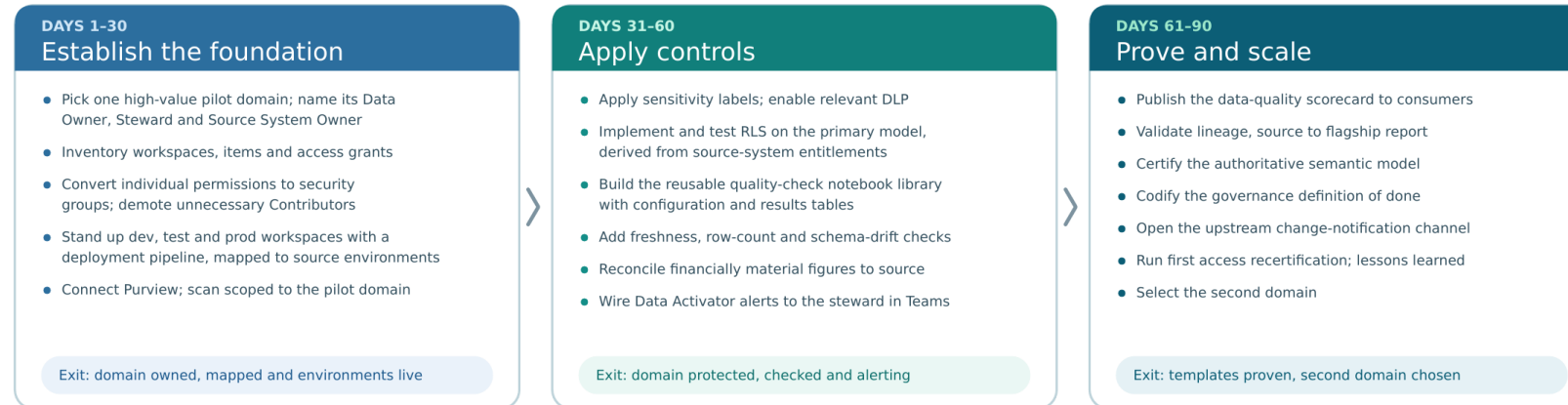
Assess honestly — risk accumulates in the gap between having a capability and enforcing a practice.
Most organizations running Fabric in production sit at level one or two, and believe they are at three.

Adoption Tactics 90-Day Governance Roadmap



The ninety-day implementation roadmap

One pilot domain, three phases — then repeat with templates that are already proven.



THEN REPEAT Run the same playbook on the second domain with the now-proven templates — domain-by-domain adoption compounds.

TWELVE ANTI-PATTERNS TO AVOID

✗ Governance as phase two

Deferred governance is retrofitted governance, at several times the cost.

✗ Report filters as security

Cosmetic — any editor can remove them. Secure the model and the data instead.

✗ Permissions to individuals

Unmaintainable within a year and a standing audit finding.

✗ One workspace for everything

No environment separation, no blast-radius control, no safe place to experiment.

✗ Bespoke quality code per table

Unmaintainable and inconsistently applied; build a parameterized library instead.

✗ Alerts to a shared mailbox

Alerts without an accountable individual are ignored by everyone equally.

✗ Classification at the report layer

By then the data has already been copied and exported.

✗ Automation running as a person

Breaks on departure, obscures audit, and is usually over-permissioned.

✗ Ignoring upstream change

Treating the source as static guarantees recurring, avoidable breakage.

✗ Reinventing entitlements

A security model unrelated to the source — two truths about who may see what.

✗ Boiling the ocean

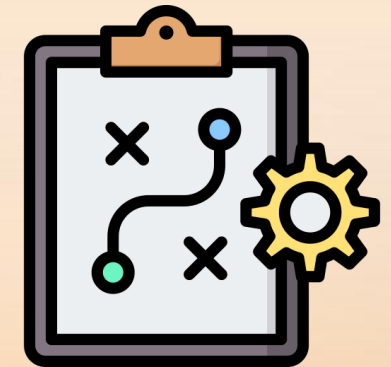
Tenant-wide programs stall; domain-by-domain adoption compounds.

✗ Measuring nothing

No control table or scorecard: no evidence of improvement, no basis for further funding.

Start with one domain, govern from day one, and measure everything.

Ninety days proves the templates. Every domain after that moves faster, because the playbook already works.



Key Takeaways



- Governance is the layer that makes a data estate trustworthy.
- Purview gives you catalog, lineage, and classification end to end.
- Model security by domain, group, and role – with RLS and env separation.
- Automate quality checks and make data health visible.

Next step: pick one domain and stand up governance end to end.

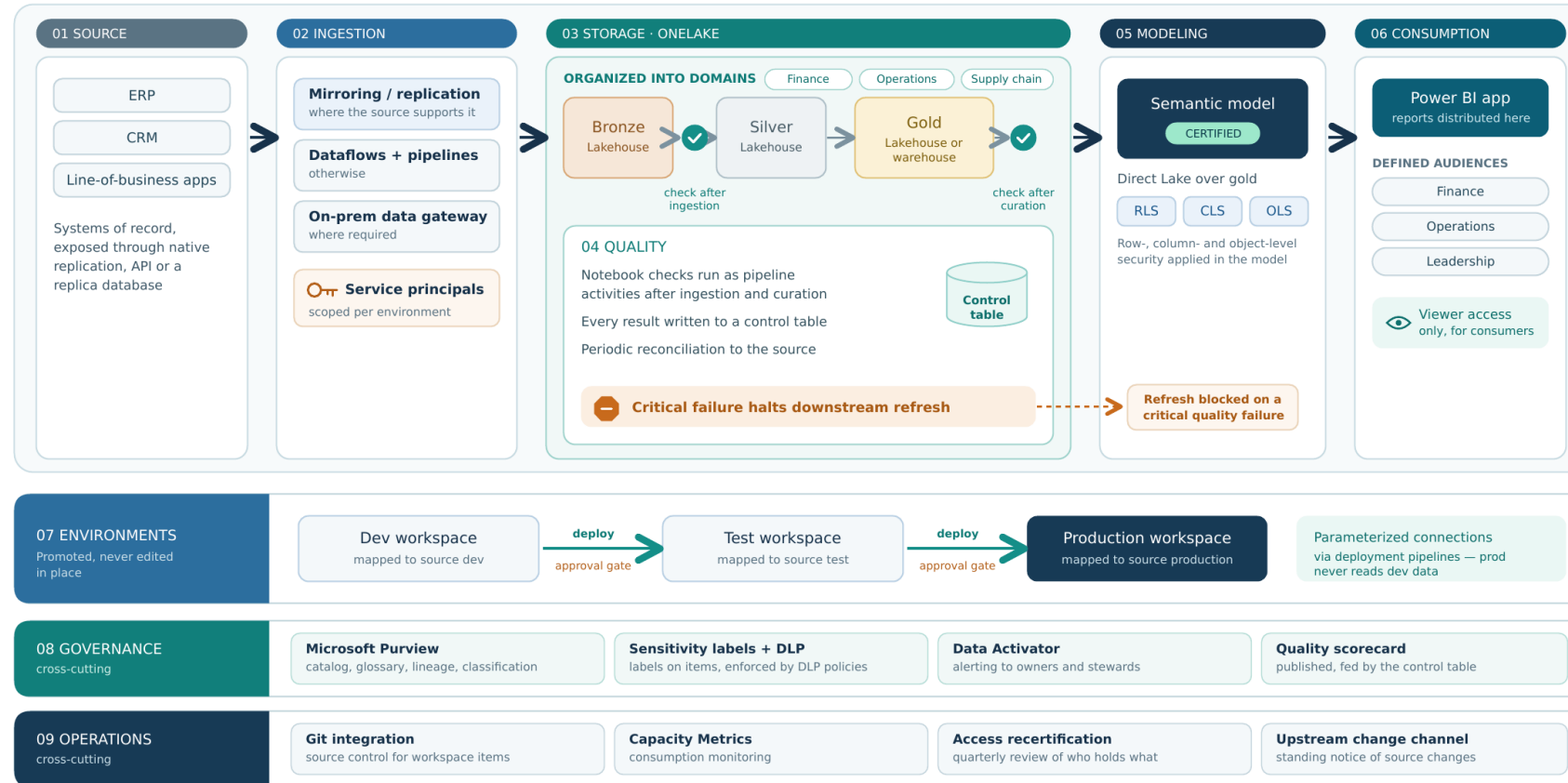


Key Takeaways – Governance in Summary



A governed Fabric lakehouse for operational data

Reference architecture, layer by layer — data flows left to right; environments, governance and operations wrap every layer.



Governance is not a layer you add — it is how every layer is built.

One governed copy flows from source to report, checked twice, certified once, and promoted through environments that mirror the source.

The largest independent Microsoft AI & Business
Applications User Conference on the planet.



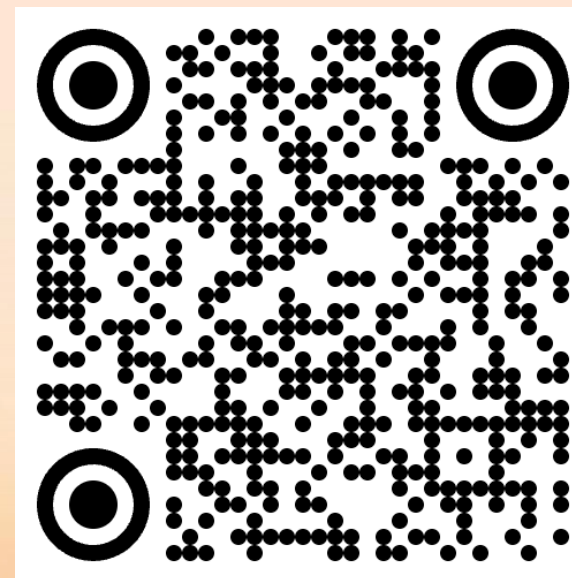
Thank you for Attending!

Chris Smith | Weidenhammer

Connect with me on LinkedIn or at www.hammer.net

Please complete the session evaluation in the Summit app.

@Dynamic Communities #MySummitNA #SummitNA



@ 2026 Dynamic Communities

Why mature your data estate

Each level of maturity answers a more valuable question — and each one stands on the level beneath it.



WHY IT IS WORTH THE CLIMB

DECISIONS YOU CAN TRUST

One governed version of the numbers ends reconciliation meetings and arguments over whose report is right.

LOWER COST PER QUESTION

Modeled, reusable data means each new question costs a query, not a project.

RISK AND COMPLIANCE

Known ownership, classification and lineage turn audits from fire drills into reports you already have.

READY FOR AI

Copilots, agents and models are only as good as the governed data beneath them. Immature data makes AI wrong, faster.

You cannot skip a level — every step stands on the one below it.
Prediction and AI fail on data that cannot yet answer “what happened” reliably. Invest in the foundations first.

Why every organization needs a corporate data catalog

A catalog is the searchable, governed inventory of what data exists, what it means, who owns it and whether to trust it.

THE QUESTIONS IT ANSWERS

Why it matters

“Where is the data?”

Find trusted data in minutes, not by asking around

“Which number is right?”

One glossary definition, and certified sources

“Who owns this?”

A named owner and steward on every asset

“Is it sensitive?”

Classification travels with the data

“What breaks if it changes?”

Lineage shows the impact before the change

EIGHT BUILDING BLOCKS

What a data catalog comprises

03 INVENTORY

Every table, file, model and report, discovered by automated scanning

03 GLOSSARY

Agreed business definitions — “revenue,” “active customer” — linked to data

03 OWNERSHIP

A data owner and steward named on each asset, by domain

03 CLASSIFICATION

Sensitivity labels and detected sensitive data such as personal information

03 LINEAGE

Where data comes from, where it goes, and what a change would break

03 QUALITY

Quality scores, freshness and certification, so users know what to trust

03 ACCESS

Who can see it today, and a clear path to request access

03 USAGE + CONTEXT

Documentation, popularity and related assets that help people choose well

EXAMPLE · ONE CATALOG ENTRY

dim_customer

Sales domain · Gold lakehouse · dim_customer

CERTIFIED

CONFIDENTIAL

GOLD LAYER

One row per customer, conformed from Dynamics 365 Sales — the authoritative customer dimension for sales reporting.

OWNER

VP, Sales Operations

GLOSSARY TERMS

Customer · Active customer

REFRESHED

Daily 06:00 — on time

STEWARD

Sales data steward

SOURCE SYSTEM

Dynamics 365 Sales

QUALITY SCORE

97%

Request access

Viewer access granted through the Sales app

LINEAGE

D365 Sales

⇒

Bronze

⇒

Silver

⇒

dim_customer

⇒

Sales semantic model

⇒

Pipeline report

COLUMNS

NAME	TYPE	CLASSIFICATION
customer_id	key	—
customer_name	text	—
email	text	PERSONAL
credit_limit	decimal	CONFIDENTIAL
region	text	—

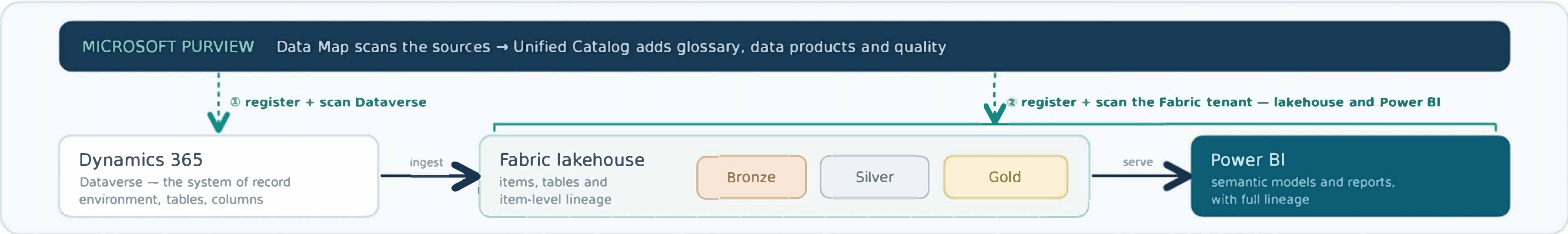
Used by 14 reports · 42 viewers this month · related: dim_account, fact_opportunity

Illustrative entry — names and figures are examples.

If people cannot find it, trust it, or tell who owns it, the data might as well not exist.
A catalog turns a scattered data estate into a shared, governed asset — build it first, not last.

Standing up a corporate data catalog in Purview

A workable sequence for data sourced in Dynamics 365, landed in a Fabric lakehouse and reported in Power BI.



PHASE A Prepare	PHASE B Connect and scan	PHASE C Enrich with meaning	PHASE D Establish trust and operate
2 Stand up Purview and domains Create governance domains and collections for the pilot; assign Data Source Admin, Data Reader and data quality steward roles; name owners.	3 Register and scan Dataverse D365 tables and columns become assets. Scope the scan to pilot tables; Dataverse supports full and scoped scans, not incremental.	4 Build the business glossary Define terms such as revenue and active customer with the business, then link them to the gold tables and models that implement them.	3 Add data quality rules Profile gold lakehouse tables and apply rules — freshness, completeness, uniqueness — scored up to the data product.
2 Create scanning identities A service principal in a security group allowed by Fabric admin API settings; a Dataverse application user with the Service Reader role.	4 Register and scan the Fabric tenant One scan brings lakehouse items and tables, Power BI semantic models and reports, plus lineage. Live view can bridge the gap in the meantime.	2 Classify and label Apply classifications and sensitivity labels, starting with D365 customer and financial tables, so protection follows the data.	10 Operate and expand Schedule rescans, review health reports, run access recertification, then onboard the next domain with the same sequence.
	2 Validate end-to-end lineage Trace D365 to lakehouse to model to report, and close manual gaps. Expect item-level lineage only for non-Power BI Fabric items.	2 Publish data products Group gold tables, models and reports into data products in Unified Catalog, each with an owner inside its governance domain.	Repeat per domain Sales first, then finance, then operations — on proven templates

KNOW THE LIMITS BEFORE YOU PROMISE

Dataverse scans No incremental scan, and deleted tables are not removed from the catalog automatically.	Fabric lineage depth Non-Power BI items get item-level lineage; lakehouse tables get metadata, not lineage.	Live view is a preview Same tenant, item-level metadata only, and no classifications — register and scan instead.	Data quality region Sources and the Purview account must sit in the same Azure region.
---	---	---	--

Scan first, then add meaning, then earn trust — one domain at a time.
Technical metadata arrives in days; the glossary, ownership and quality that make a catalog useful take deliberate work.

Purview with Fabric: what's included vs. what you add

Fabric governs its own estate out of the box. Purview extends protection and governance across everything else — and carries its own licensing.

INCLUDED WITH YOUR FABRIC LICENSE

Fabric's built-in governance

- OneLake catalog — discover and explore everything in your Fabric estate
- Governance insights and recommended actions, including the security insights that used to live in the Purview hub
- Domains and subdomains for federated ownership, plus custom tags
- Endorsement — promoted and certified items
- Lineage across Fabric items, source to report, and impact analysis
- Workspace, item and OneLake permissions

Scope stops at the Fabric tenant — enough for one team, not for an enterprise estate

REQUIRES MICROSOFT PURVIEW LICENSING

What Purview adds

- Information Protection — sensitivity labels and protection policies
- Data loss prevention for lakehouses, warehouses, databases and semantic models
- Purview Audit — every Fabric user activity in the audit log
- Insider Risk Management, with ready-made Fabric indicators
- Unified Catalog and Data Map — one catalog spanning D365, Azure, on-premises and third-party systems, with glossary and data quality
- Governance for Fabric Copilots and agents

Separate licensing; non-Microsoft 365 sources are billed pay-as-you-go

WHERE THE LINE FALLS	FABRIC, BUILT IN	PURVIEW, LICENSED
Scope	Your Fabric tenant	Fabric, D365, Azure, on-premises, third-party systems
Catalog	OneLake catalog	Unified Catalog, backed by the Data Map
Lineage	Across Fabric items, source to report	Across systems, beyond the workspace boundary
Classification	Manual tags and endorsement	Automated scanning and sensitive-data detection
Enforcement	Workspace and item permissions	Sensitivity labels, protection policies, DLP actions
Cost	Included with Fabric	Purview licensing, plus consumption for non-M365

HOW FABRIC DATA REACHES THE PURVIEW CATALOG

LIVE VIEW (PREVIEW)

Same-tenant workspaces and items appear in the catalog with no scan to configure. Item-level metadata only, and no classifications are applied.

REGISTER AND SCAN

Register the Fabric tenant in the Data Map. Full, incremental and scoped scans bring item metadata and lineage, and enable classification.

WHAT YOU STILL SET UP

A service principal, a security group registered in the Fabric tenant admin API settings, and a runtime. Lineage below item level is not supported.

Governance travels with the data, not with the tool.
Fabric's own catalog is enough for one team. What makes it enterprise governance is licensed separately — so budget for it deliberately.

Row-level and object-level security

One model, many audiences — each sees only its own rows, columns and tables.

01 · SEMANTIC MODEL

Row-level security

Which rows can this user see?
Legal entity, region, business unit, cost centre

- Define roles on the model, never per report
- Report-level filters are not a security control — any user who can edit the report can remove them
- Use dynamic RLS: one DAX filter matched to the signed-in user principal name, so role count stays constant as the organization grows
- Keep the user-to-entity mapping in a data table, not in DAX — access changes become a data load

02 · SEMANTIC MODEL

Column and object level

Which fields and tables can they see?
Row filtering is not always sufficient

- Column-level security restricts the field itself — salary, national identifiers, margin
- Object-level security hides entire tables and measures, so they never appear in the field list
- Use these instead of maintaining parallel models for different audiences
- Duplicate models multiply both cost and inconsistency — avoid the workaround

03 · UNDERNEATH THE MODEL

OneLake data access

What can they reach in the lakehouse?
Workspace roles alone are coarse

- A Viewer on a workspace can generally read the lakehouse tables within it
- OneLake data access roles add table- and folder-level control inside the lakehouse itself
- Enforced regardless of the engine used to query
- Pair model RLS with OneLake roles wherever users could reach the data directly



- PUT THESE IN THE DEFINITION OF DONE
- Roles live on the model, not on the report

Dynamic RLS keyed to the user principal name

Mapping maintained in data, not in DAX

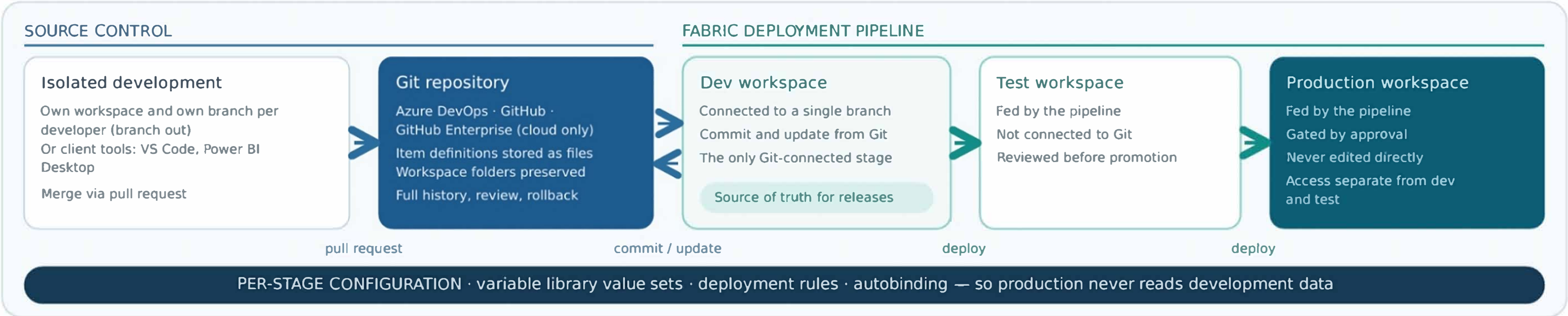
Every role validated with “View as role”

OneLake roles wherever direct access is possible

Model security governs the report. OneLake security governs the data.
Where a user can query the lakehouse directly, semantic-model RLS is not a boundary — it is a convenience. You need both layers.

Git repos with Fabric deployment pipelines

Git versions the definitions. The pipeline promotes them. Connect Git to development only.



CONTINUOUS INTEGRATION

What Git integration gives you

- Connection is per workspace, and a workspace syncs to exactly one branch
- Item definitions are stored as files, so changes are reviewed in pull requests and revertible to any commit
- Workspace folder structure is preserved in the repo
- A shared workspace is a shared runtime — give each developer a branched workspace instead

CONTINUOUS DELIVERY

What the pipeline gives you

- Promotes items between the dev, test and prod workspaces without re-importing anything
- Deployment rules and autobinding re-point data sources per stage, so the same artifact just works
- Compare stages before deploying, and gate promotion with approvals
- Restrict who may deploy to production

SIX RULES THAT KEEP THE TWO IN STEP

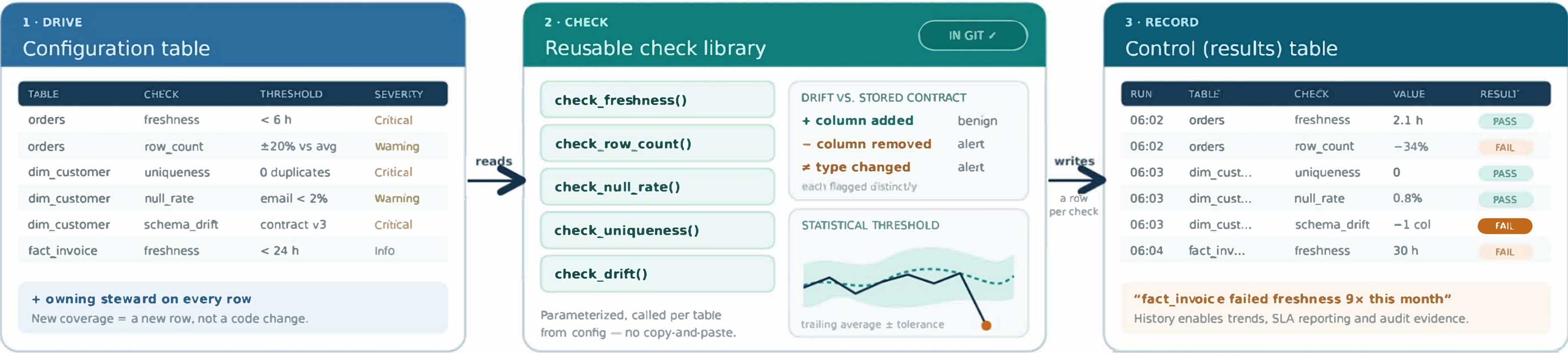
ONE BRANCH PER WORKSPACE A workspace syncs to a single branch. Developers branch out to their own workspace to work.	GIT ON DEV ONLY Test and prod are fed by the pipeline, not by Git — one source of truth per stage.	CONFIGURATION AS CODE Variable library value sets per stage; deployment rules and autobinding re-point the sources.
NEVER EDIT A HIGHER STAGE Every change starts in a branch and flows forward. A direct edit in production is drift.	PERMISSIONS ARE SEPARATE Repo access, pipeline access and workspace roles are three different grants. Plan all three.	KNOW WHAT GIT DOES NOT HOLD Definitions are versioned; lakehouse data is not. Unsupported items are ignored, not synced.

Connect the developer workspace to Git, then deploy from it with a pipeline.
Git answers “what changed and who approved it.” The pipeline answers “how it reaches production.” Together: no more “it worked in my workspace.”

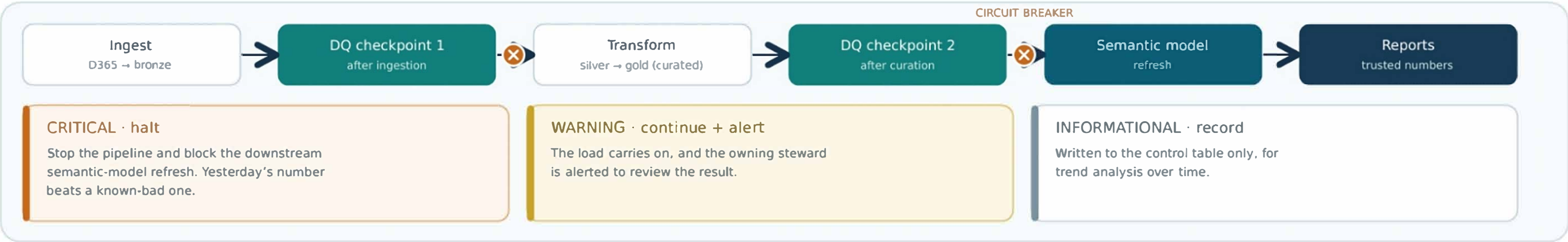
Fabric notebooks: the natural home for data quality

Quality logic that runs where the data lives — configured, versioned, recorded and wired into the load.

- ✓ Runs inside the platform
- ✓ Against lakehouse tables
- ✓ On the same capacity
- ✓ No external infrastructure



PLACEMENT · CHECKS RUN AS PIPELINE ACTIVITIES, NOT A SEPARATE PROCESS



DESIGN GUIDANCE FROM PRODUCTION IMPLEMENTATIONS

- Reusable check library

Generic functions called per table.
Configuration scales;
copy-and-paste does not.
- Configuration-driven

Table, check, threshold, severity
and steward held as data — new
coverage is a row, not a release.
- Drift against a contract

Persist the expected schema and
compare every run; additions,
removals and type changes apart.
- Statistical thresholds

A trailing average with a
tolerance band handles seasonality
far better than a fixed minimum.
- Data quality is code

Git-connected notebooks get review,
versioning and rollback. Public
GitHub repos give a head start.

Fail loudly, fail early — a circuit breaker on the data pipeline.
It is what stops a quality issue from becoming a credibility issue. Without persisted results, you have alerts but no measurement.

Data quality: Purview rules vs. Fabric notebook checks

One measures the data and publishes a score. The other decides whether today’s load is allowed through.

GOVERNANCE PLANE · LOW-CODE TO NO-CODE

Purview data quality rules

Rules configured in Unified Catalog, applied per column, run as a scan, and aggregated into scores for assets, data products and domains.

AUTHORING

Out-of-the-box rules plus AI-suggested rules; no notebook required

RULE CATALOG

Freshness, unique values, empty fields, data type and format match, duplicate rows, table lookup, plus a custom condition expression

WHO OWNS IT

Data quality steward to author; reader role to view

THRESHOLDS

Configurable score thresholds per rule, asset and data product

WHEN IT RUNS

Profiling first, then manual or scheduled scans

Estate-wide score and stewardship

It cannot stop a pipeline

DATA PLANE · QUALITY AS CODE

Fabric notebook checks

PySpark or SQL checks running inside the pipeline, against lakehouse tables, on the same capacity, with no external infrastructure.

AUTHORING

Reusable parameterized functions, driven from a configuration table

CHECK TYPES

Freshness, row-count deviation, null rate, uniqueness, referential integrity, and schema drift against a stored contract

WHO OWNS IT

Data engineer, with the owning steward named in configuration

THRESHOLDS

Any logic — a tolerance band around a trailing average

WHEN IT RUNS

A pipeline activity, after ingestion and after curation

Stops a bad load reaching reports

You build and maintain it

CONSIDERATION	PURVIEW RULES	NOTEBOOK CHECKS
Primary question	How good is this data product?	Is today’s load safe to publish?
Can it block a load?	No — it measures, scores and notifies	Yes — severity tiers halt the pipeline
Custom logic depth	Fixed rule set plus a custom expression	Unlimited; any Spark or SQL logic
Evidence and history	Built-in rule history and health reports	Your control table, retained as you choose
Versioning	Managed inside the service	Git-reviewed, versioned, roll back anytime
Reach	The whole estate, across clouds	Only the tables you wrote checks for

WHAT THE NOTEBOOK DOES WITH A FAILURE — DEFINE THE TIERS

CRITICAL

Halt the pipeline; block the model refresh

WARNING

Continue, but alert the owning steward

INFORMATIONAL

Record only, for trend analysis

Not a choice — Purview scores the data, the notebook enforces it.
Publishing a known-bad number is worse than publishing yesterday’s number, and only the in-pipeline check can prevent it.

Governance as definition of done

Governance fails when treated as a separate project. It succeeds when embedded in the delivery process.

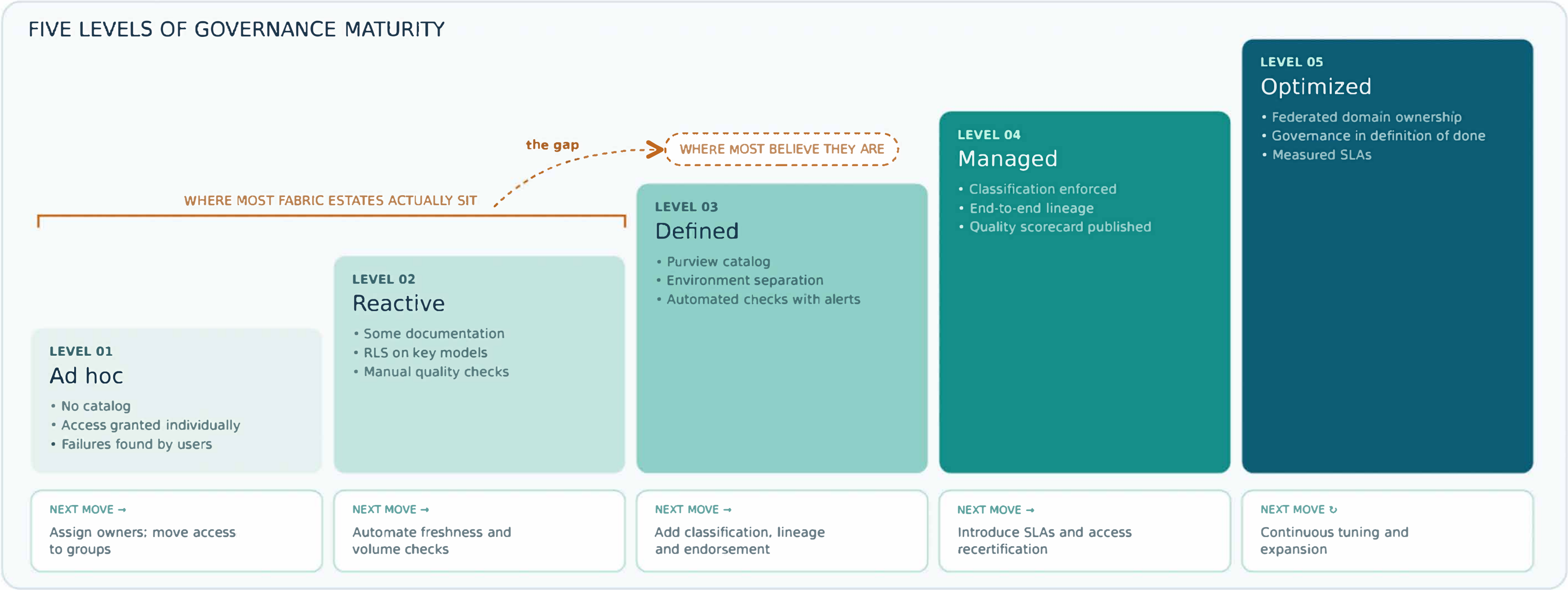
A workable definition of done for any new data product
Seven conditions, checked at review — before the artifact is promoted to production.

REQUIREMENT	DISCIPLINE	ACCOUNTABLE
☒ 01 Catalogued and described, with a named owner and steward	CATALOG	Data Steward
☒ 02 Sensitivity classification applied	CLASSIFICATION	Data Steward
☒ 03 Access granted to security groups, with RLS implemented and tested where required	SECURITY	Data Engineer
☒ 04 Quality checks for freshness, volume and schema drift, with alerts routed to the steward	DATA QUALITY	Data Engineer
☒ 05 Lineage resolves end to end, with no manual gaps	LINEAGE	Data Engineer
☒ 06 In source control and promoted through the deployment pipeline	CHANGE CONTROL	Data Engineer
☒ 07 Semantic model endorsed, if it is intended to be authoritative	ENDORSEMENT	Data Owner

A checklist enforced at review is worth more than a policy document nobody reads.
Put these conditions in the delivery process itself, so governance is how work gets finished rather than a separate initiative.

A governance maturity model

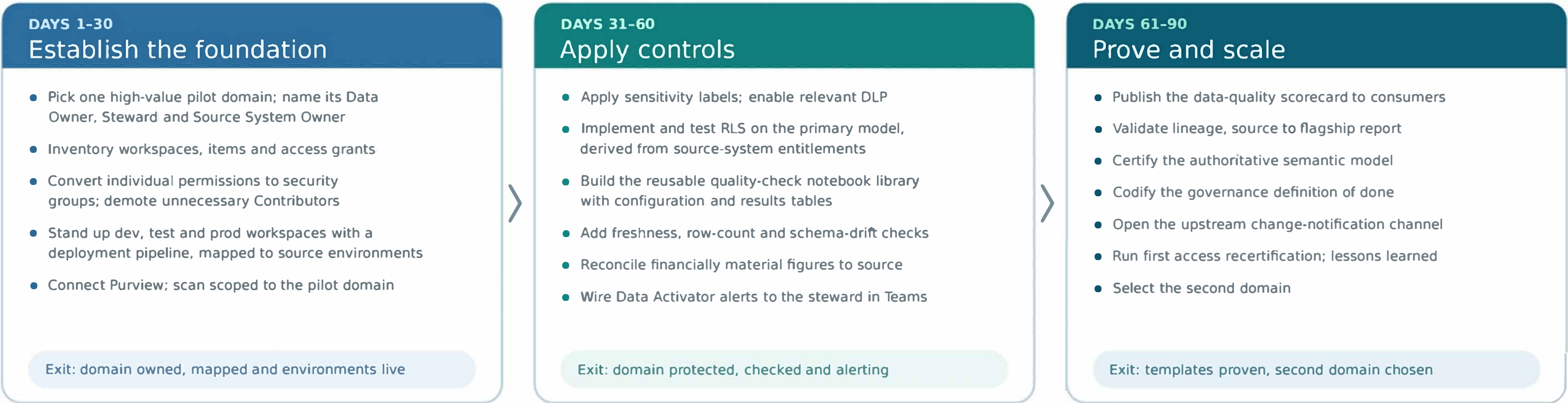
Five levels, each with a clear next move. The hard part is not knowing the levels — it is admitting which one you are on.



Assess honestly — risk accumulates in the gap between having a capability and enforcing a practice.
Most organizations running Fabric in production sit at level one or two, and believe they are at three.

The ninety-day implementation roadmap

One pilot domain, three phases — then repeat with templates that are already proven.



THEN REPEAT Run the same playbook on the second domain with the now-proven templates — domain-by-domain adoption compounds.

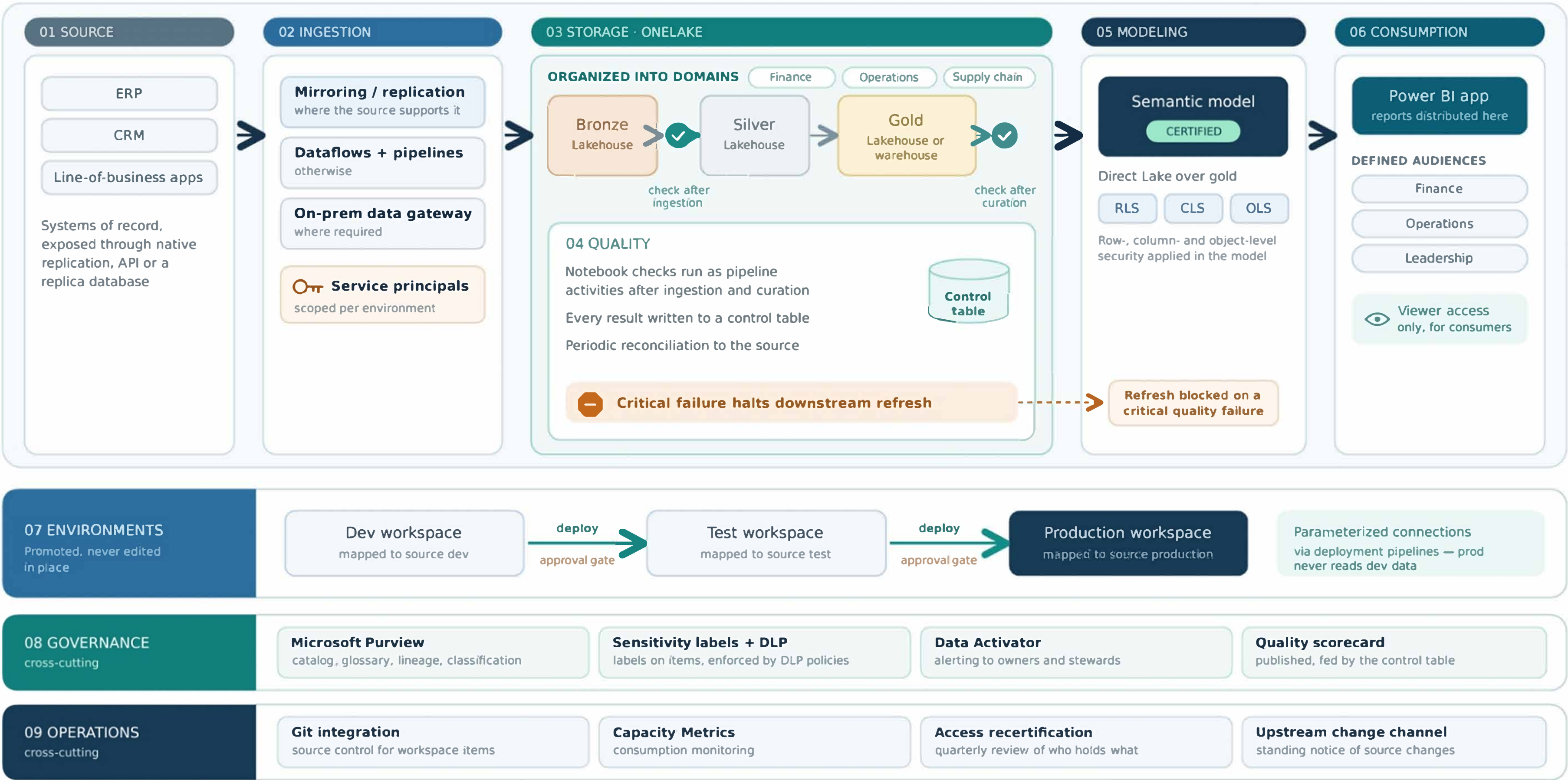
TWELVE ANTI-PATTERNS TO AVOID

✗ Governance as phase two Deferred governance is retrofitted governance, at several times the cost.	✗ Report filters as security Cosmetic — any editor can remove them. Secure the model and the data instead.	✗ Permissions to individuals Unmaintainable within a year and a standing audit finding.	✗ One workspace for everything No environment separation, no blast-radius control, no safe place to experiment.
✗ Bespoke quality code per table Unmaintainable and inconsistently applied; build a parameterized library instead.	✗ Alerts to a shared mailbox Alerts without an accountable individual are ignored by everyone equally.	✗ Classification at the report layer By then the data has already been copied and exported.	✗ Automation running as a person Breaks on departure, obscures audit, and is usually over-permissioned.
✗ Ignoring upstream change Treating the source as static guarantees recurring, avoidable breakage.	✗ Reinventing entitlements A security model unrelated to the source — two truths about who may see what.	✗ Boiling the ocean Tenant-wide programs stall; domain-by-domain adoption compounds.	✗ Measuring nothing No control table or scorecard: no evidence of improvement, no basis for further funding.

Start with one domain, govern from day one, and measure everything.
Ninety days proves the templates. Every domain after that moves faster, because the playbook already works.

A governed Fabric lakehouse for operational data

Reference architecture, layer by layer — data flows left to right; environments, governance and operations wrap every layer.



Governance is not a layer you add — it is how every layer is built.

One governed copy flows from source to report, checked twice, certified once, and promoted through environments that mirror the source.