Microsoft Fabric

This presentation is the property of Microsoft and is intended for informational and educational purposes only. You may use, copy, and distribute this presentation for your personal, non-commercial purposes. You may not modify, alter, or create derivative works from this presentation without the prior written consent of Microsoft. You may not use this presentation to misrepresent, defame, or disparage Microsoft or its products, services, or affiliates. You may not use this presentation to endorse or promote any other products, services, or organizations without the prior written consent of Microsoft.

By using this presentation, you agree to abide by these terms. If you do not agree, you must not use this presentation. Microsoft reserves the right to change these terms and conditions at any time without notice. Microsoft disclaims any and all warranties, express or implied, relating to this presentation, including but not limited to the accuracy, completeness, timeliness, or suitability of the information contained herein. Microsoft is not liable for any damages, losses, or liabilities arising from your use of or reliance on this presentation.

Please review the terms of use posted in the content library.

#FABCONSQLCON2026

FABCON
Microsoft Fabric
COMMUNITY CONFERENCE

SQLCON
Microsoft SQL
COMMUNITY CONFERENCE

ATLANTA MARCH 16 - 20, 2026

Mastering Fabric Data Engineering Admin and Capacity Management

Christopher Finlan

Santhosh Kumar Ravindran

Speaker Bio

Chris Finlan

Microsoft Corporation

christopherfinlan

cmfinlan

Santhosh Kumar Ravindran

Microsoft Corporation

thisissanthoshkumar

thisissanthoshkumar

Microsoft Confidential

Microsoft Fabric

The unified data platform for AI transformation

Data Factory

Analytics

Databases

Real-Time Intelligence

Power BI

Fabric Platform

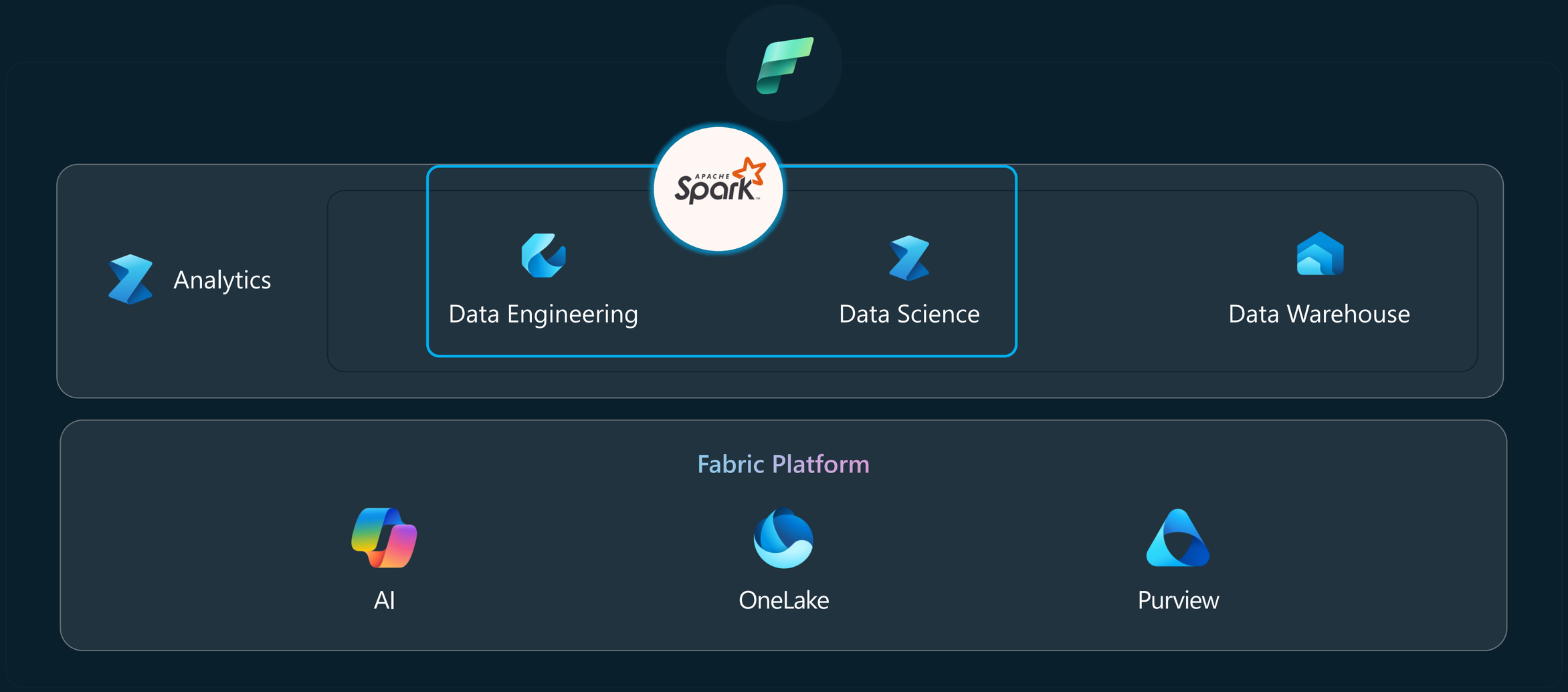
AI

OneLake

Purview

Microsoft Fabric

The unified data platform for AI transformation



🧭 Overview

Mastering Fabric Data Engineering Admin & Capacity Management

Running Spark well isn't about tuning every job

it's about **making the right platform decisions once**, then letting teams move fast safely.

In this session, we'll walk through:

- How Microsoft Fabric **plans, governs, executes, and recovers** Spark workloads
- The **defaults you should trust**, and the few levers that actually matter
- How to keep Spark **predictable, fast, and cost-controlled** as usage scales
  It's an **operator's playbook** for running Spark in Fabric, without turning every admin into a Spark internals expert.

Goal

What good Spark operations look like in Fabric

Governed, predictable, and fast - without making every workspace admin a Spark internals expert.

01

Cost control

Separate steady-state spend from burst demand and cap Spark expansion deliberately.

02

Predictable startup

Know when you are choosing a warm path, a cold path, or session reuse.

03

Standardized defaults

Pin pool, environment, runtime, and session policy before teams spread.

04

Visible bottlenecks

Queued vs running vs throttled should be obvious in monitoring.

Every chapter should answer four things: who owns the setting, what the default should be, when to make an exception, and where to verify the result.

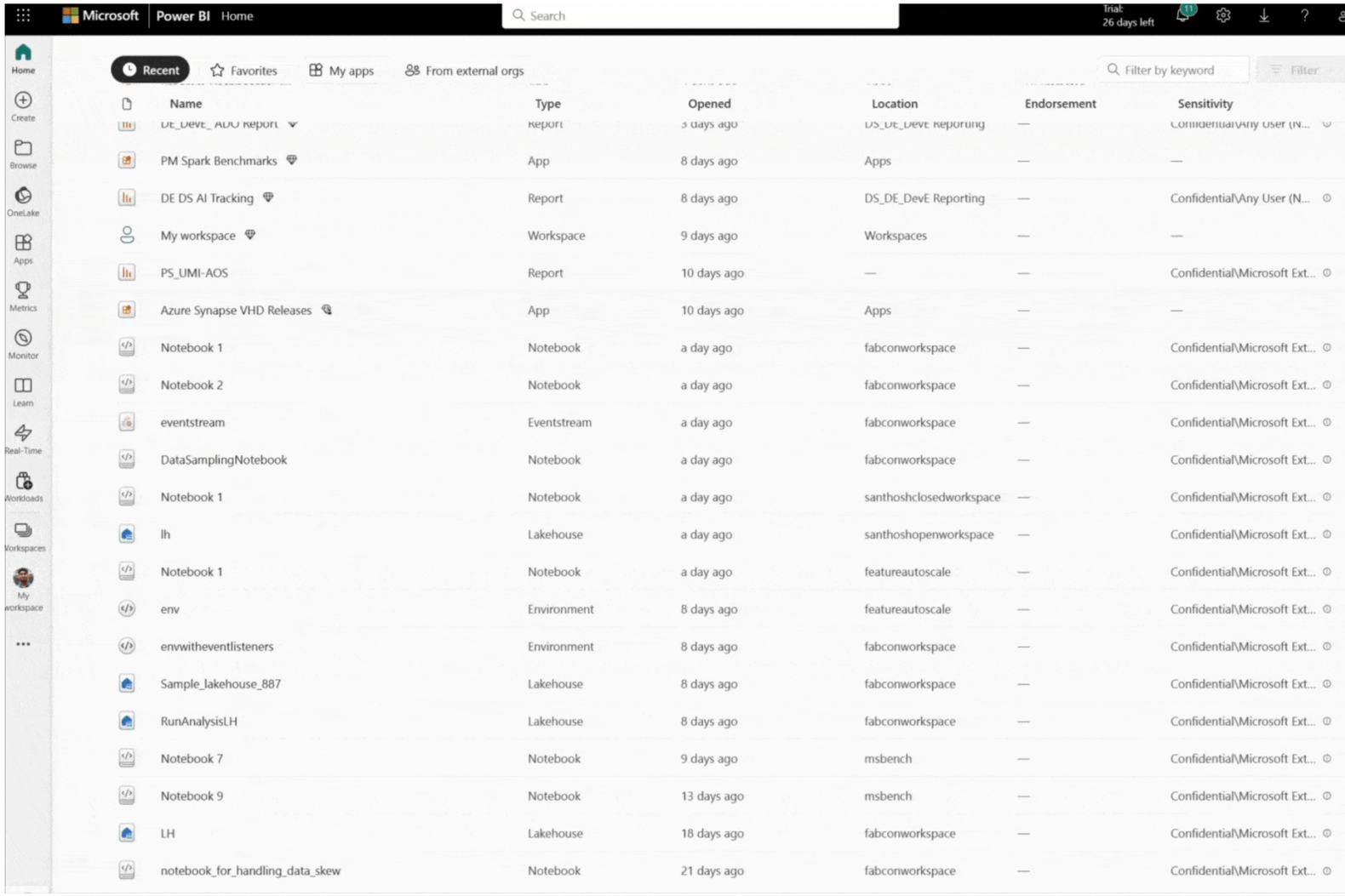
Autoscale Billing for Fabric Data Engineering

Flexible, Pay-as-You-Go Autoscale Billing for Spark

Run Spark independently of shared Fabric capacity where you are only billed for active Spark jobs.

Seamless Integration – Monitor usage for Autoscale billing in the **Fabric Capacity Metrics App** and request more capacity units via **Azure Quota Management.**

Optimized Resource Management – Spark jobs scale separately from base capacity, reducing contention for shared resources and ensuring smoother execution of dynamic, compute-intensive workloads.



Compute Strategy: Fixed vs Autoscale

Aspect	Fixed Capacity	Autoscale
Billing	Monthly cost by SKU (F64, F128)	Seconds of active Spark job usage
Scaling	Shared CU capacity for all workloads	Dedicated Spark resources, independent scaling
Mechanics	Smoothing and bursting up to 3x for peaks	No smoothing; jobs admitted to Max CU Limit
Best For	Predictable recurring production ETL	Ad-hoc data science and bursty processing

- Fixed Fabric Capacity provides budget consistency for steady-state workloads by sharing resources across Power BI, Data Factory, and Spark
- Autoscale Billing offers serverless flexibility where you pay only for active Spark seconds, scaling independently without affecting base capacity
- Autoscale prevents Power BI report slowdowns by isolating heavy Spark jobs to dedicated resources with user-defined Max CU limits

Benefits

Eliminate contention, avoid over-provisioning, enable granular cost control by workload type

Serverless Layer

Autoscale Billing for Spark-heavy Data Engineering and Data Science workloads scales on demand

Foundation

Reserved Instances (RI) for base F-SKUs lock in lowest pricing for Power BI, SQL Warehouse, OneLake storage

Best Practice: Hybrid Compute Blueprint

- Teams commonly over-provision fixed capacity to handle peak Spark loads, which is expensive and wasteful during low-usage periods
- The pro move is reserving base capacity for steady workloads and letting autoscale handle bursts like having a daily driver and heavy-duty truck
- Monitor autoscale usage in Fabric Capacity Metrics App and request higher limits via Azure Quota Management for hyperscale processing

Governance Hierarchy for Data Engineering Compute in Microsoft Fabric

Fabric provides a **multi-layered compute governance** model to manage and control compute resources efficiently.

- **Capacity-Level Pools**

Set by capacity admins to define max cores per job.

Admins can delegate control to workspace admins.

- **Workspace-Level Pools**

Created by workspace admins for different teams.

Enables resource allocation based on team needs.

- **Environments**

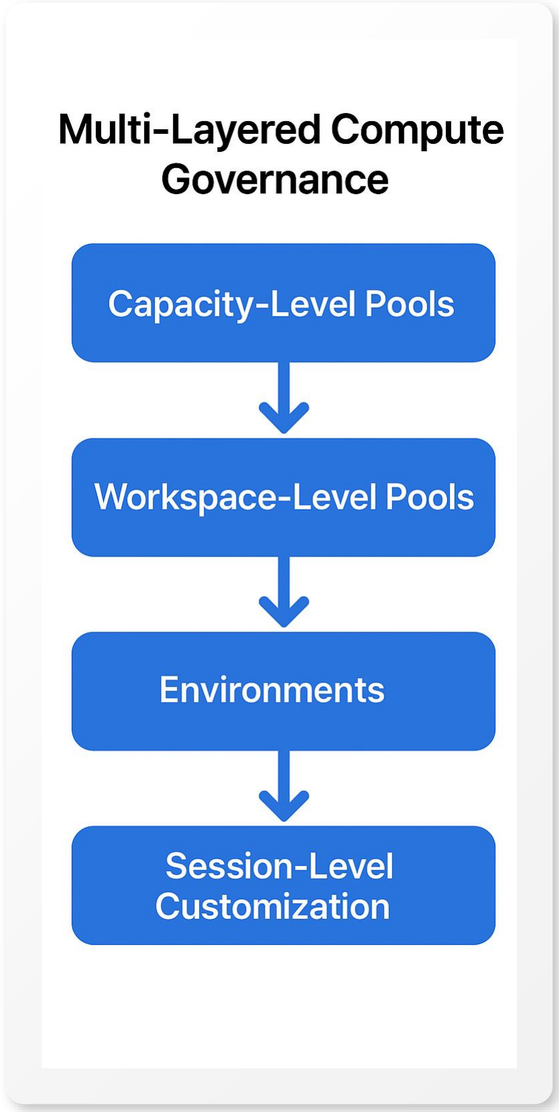
Customize Spark Compute settings for specific workloads.

Provides better isolation and control.

- **Session-Level Customization**

Fine-tune Spark session settings using %%configure in notebooks.

Allows real-time adjustments with notebookutils

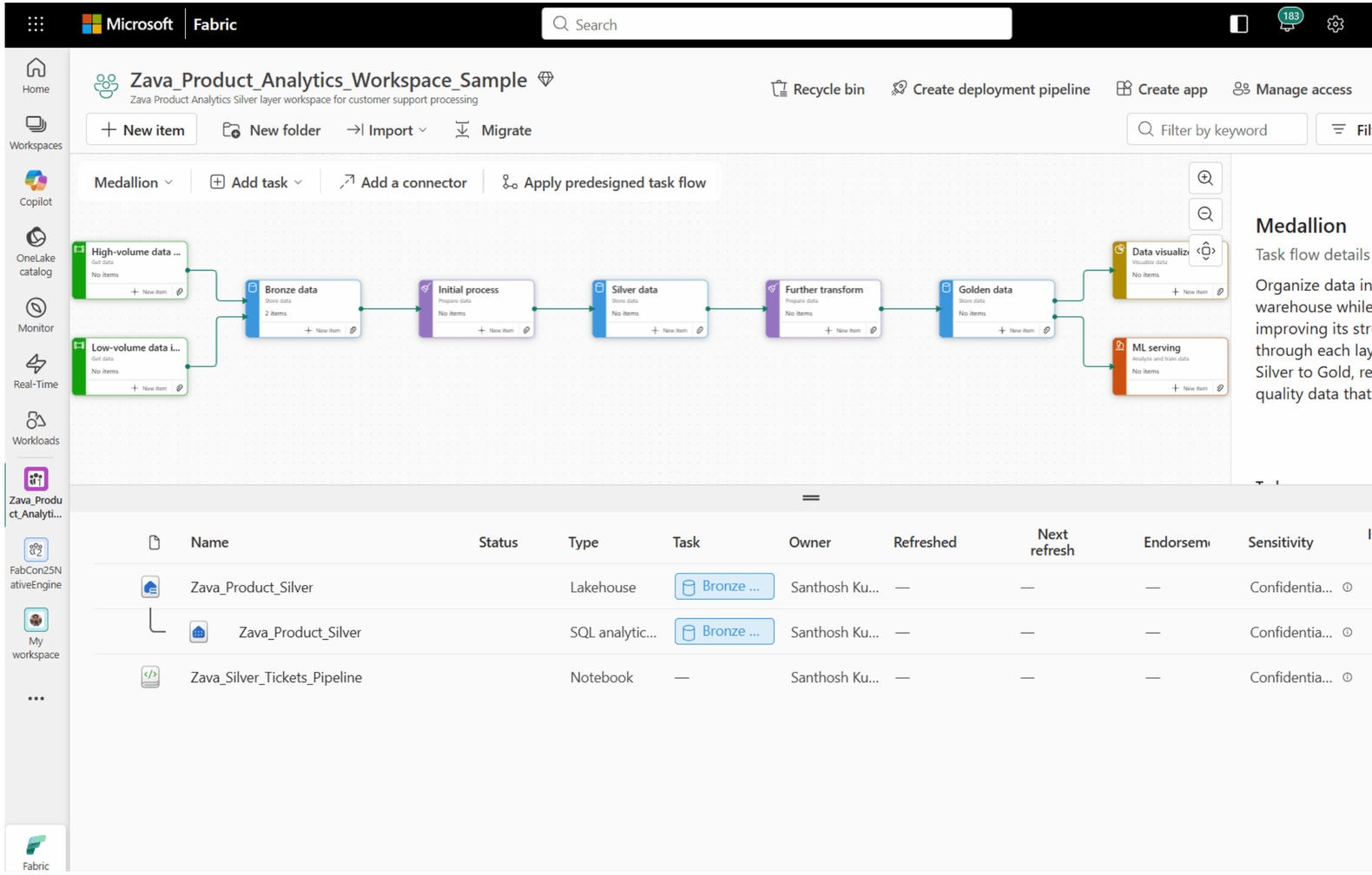


Capacity Level Pools

- **Centralized Governance:** Capacity Admins can now explicitly **Disable Starter Pools** at the capacity level, forcing all workspaces to utilize defined Custom Pools.

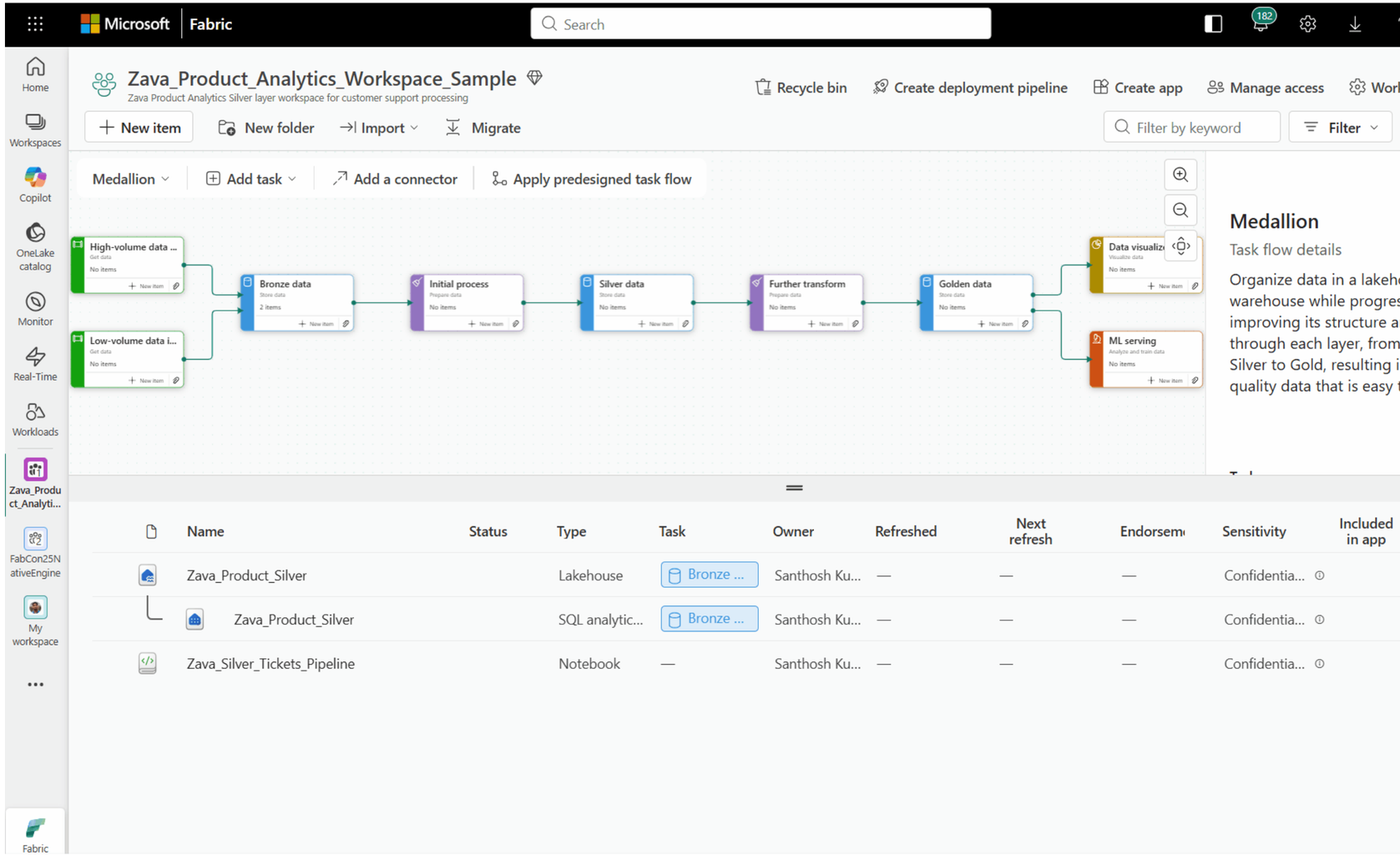
- **Best Practice:** Use this to prevent "noisy neighbor" scenarios where unregulated Starter Pools consume CUs needed for production pipelines.

- **Governance Flow:**
  - Admin disables Starter Pools in Capacity Settings.
  - Workspace Admins are notified to select a governed Custom Pool created by the Capacity Admin



Workspace Level Pools

- Custom Pools can be configured by **workspace admins** in workspace settings.
- All pools are always within the **maximum CU limit** of the Capacity
- Pools created by admins can be selected at the in an **environment**
- Workspace admins can choose to **allow or disable** compute customization in environments

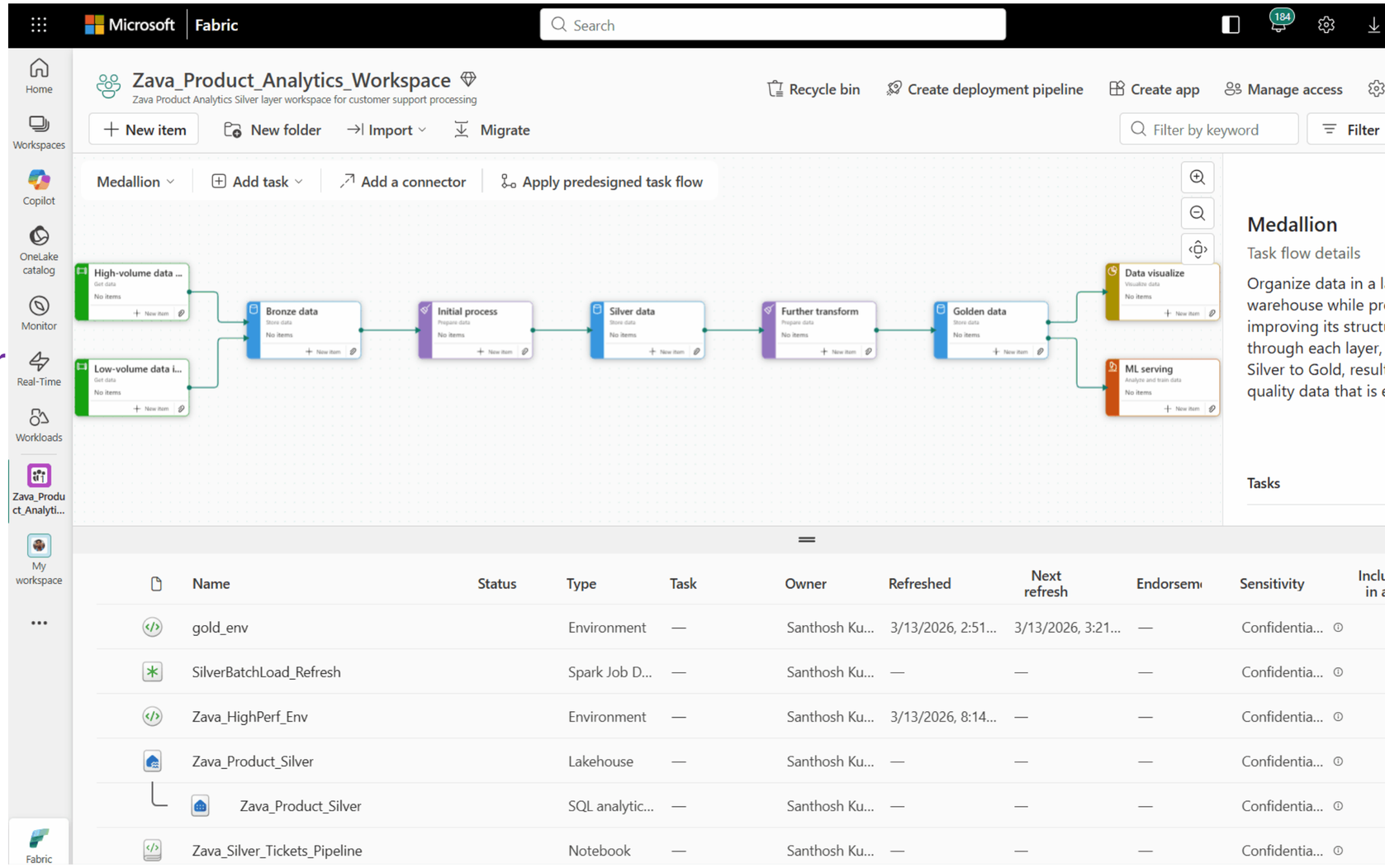


Library Management and Advanced Customization with Environments

- The environment artifact allows users to configure all their Spark and library related settings in a single place

- Users can do the following (subject to the permissions they have been granted):
  - Install libraries
  - Choose from a list of pool options (**starter capacity / workspace**) pool.
  - Set Spark properties
  - Attach files to the resources to use as references in job execution

- Environments can be attached to **notebooks and Spark jobs**, overriding the default workspace pool



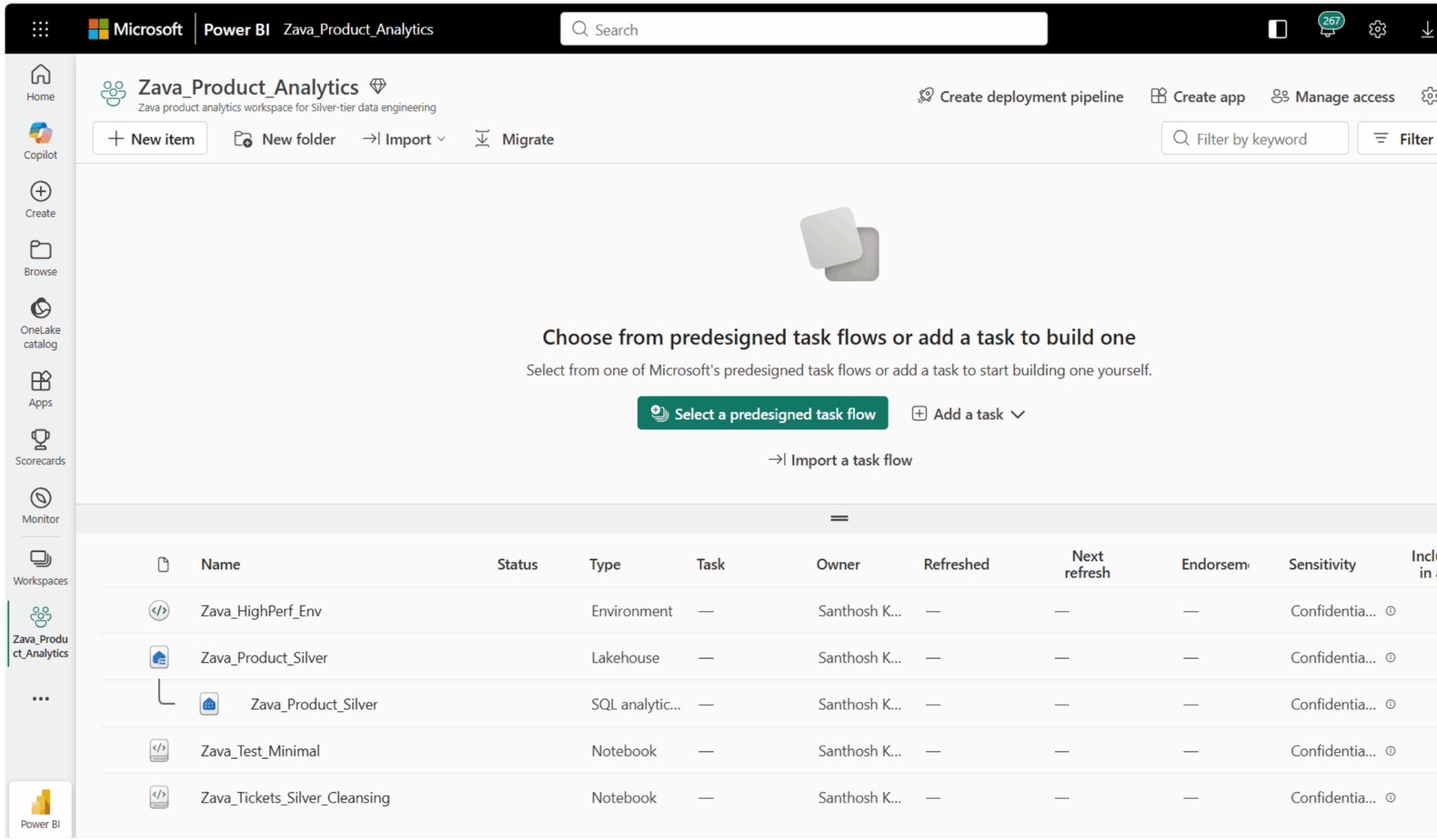
Public Preview

Custom Live Pools

Bringing Starter Pool Speed to Custom Environments and Network Protected Workspaces

Traditionally, custom environments with specialized libraries required **3–5 minutes** to initialize because hardware had to be provisioned and libraries installed from scratch for every new session.

- **The Solution: Custom Live Pools** allow admins to maintain "warm" nodes pre-configured with your specific library and environment settings.
- **Impact:** Reduces session startup time to **5–10 seconds**, matching the speed of standard Starter Pools while retaining 100% of your environmental flexibility.



Pool Strategy: Starter vs Custom vs Custom Live

- Capacity admins can disable Starter Pools entirely to prevent noisy neighbor scenarios where unregulated pools consume CUs needed for production
- Custom Live Pools maintain warm nodes pre-configured with custom libraries reducing session startup from 3-5 minutes to 5-10 seconds
- Schedule pool activation windows daily or weekly to ensure compute is ready exactly when teams arrive for work

Spark Pool Options Decision Matrix

Pool Type	Startup Time	Customization	Best For
Starter Pool	5-10 seconds (prewarmed)	Limited to SKU bounds	Ad hoc work prioritizing low latency
Custom Pool	2-3 minutes (cold start)	Full control (autoscale, dynamic allocation, single-node)	Governed sizing and specialized compute
Custom Live Pool (Best of Both)	5-10 seconds (prewarmed with libraries)	Full control with scheduled hydration	Library-aware and network-isolated environments

Performance by Default: Resource Profiles

Public Preview

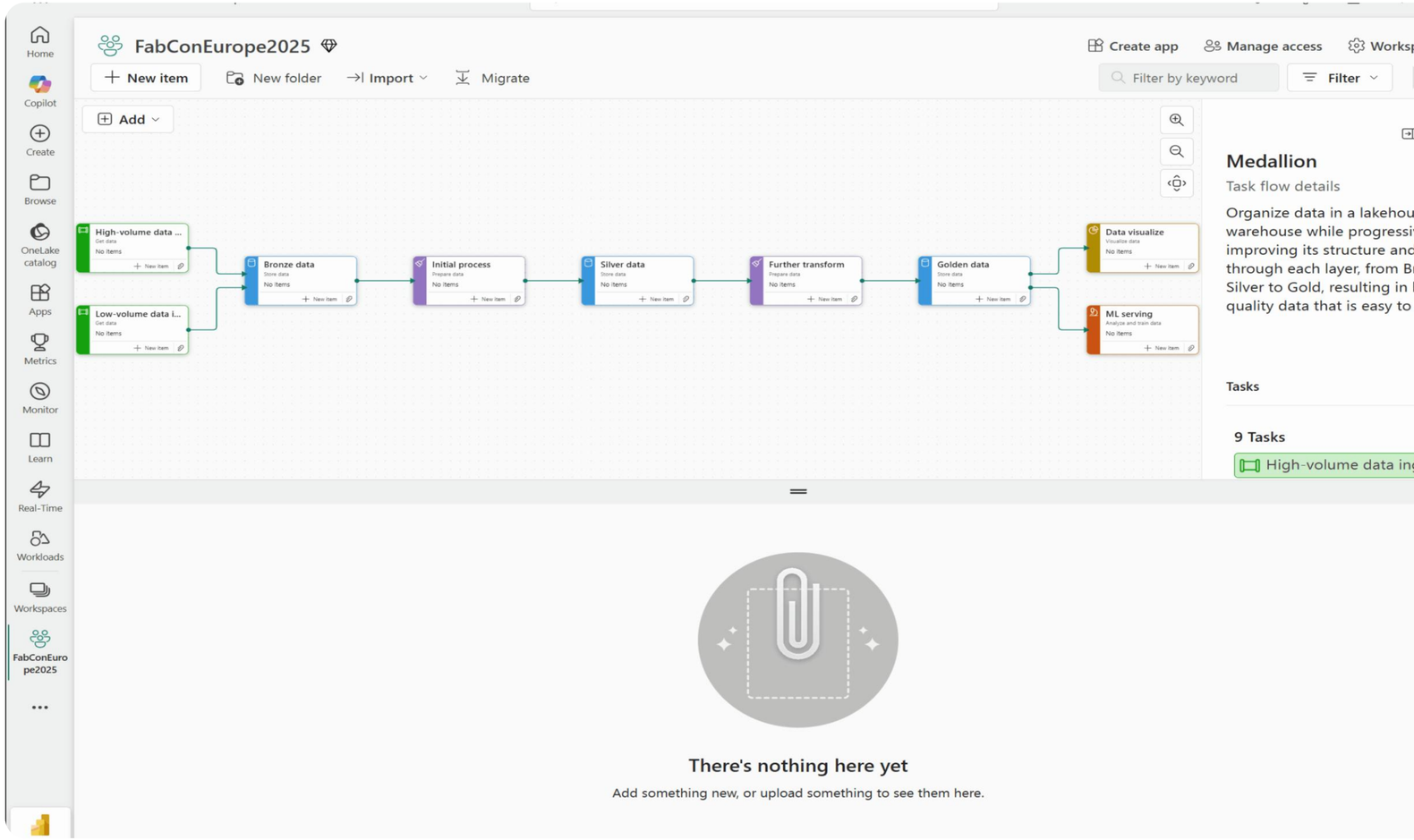
Smarter Configuration, Less "Spark Tuning“

- **Answer a few questions about your work** to be done instead of digging around in complex Spark configs.

- **Automated spark pool and environment setup** optimized for your specific needs (Ingestion vs. Consumption).

Predefined Property Bags:

- **writeHeavy**: The new workspace default. Optimized for massive ETL and high-frequency writes.
- **readHeavyForPBI**: Tuned for Direct Lake performance to keep Power BI reports snappy.



Resource Profiles: Automated Optimization

writeHeavy Profile

readHeavyForPBI Profile

- Resource profiles provide predefined property bags optimized for specific workload types eliminating need to dig through complex Spark configs
- writeHeavy profile is the new workspace default optimized for massive ETL and high-frequency writes to maximize ingestion throughput
- readHeavyForPBI profile is tuned for Direct Lake performance to keep Power BI reports snappy during concurrent user access
- Profiles remain fully customizable for unique edge cases while providing intelligent defaults for 90% of common scenarios

Inside the "Black Box": Session Transparency

Public Preview

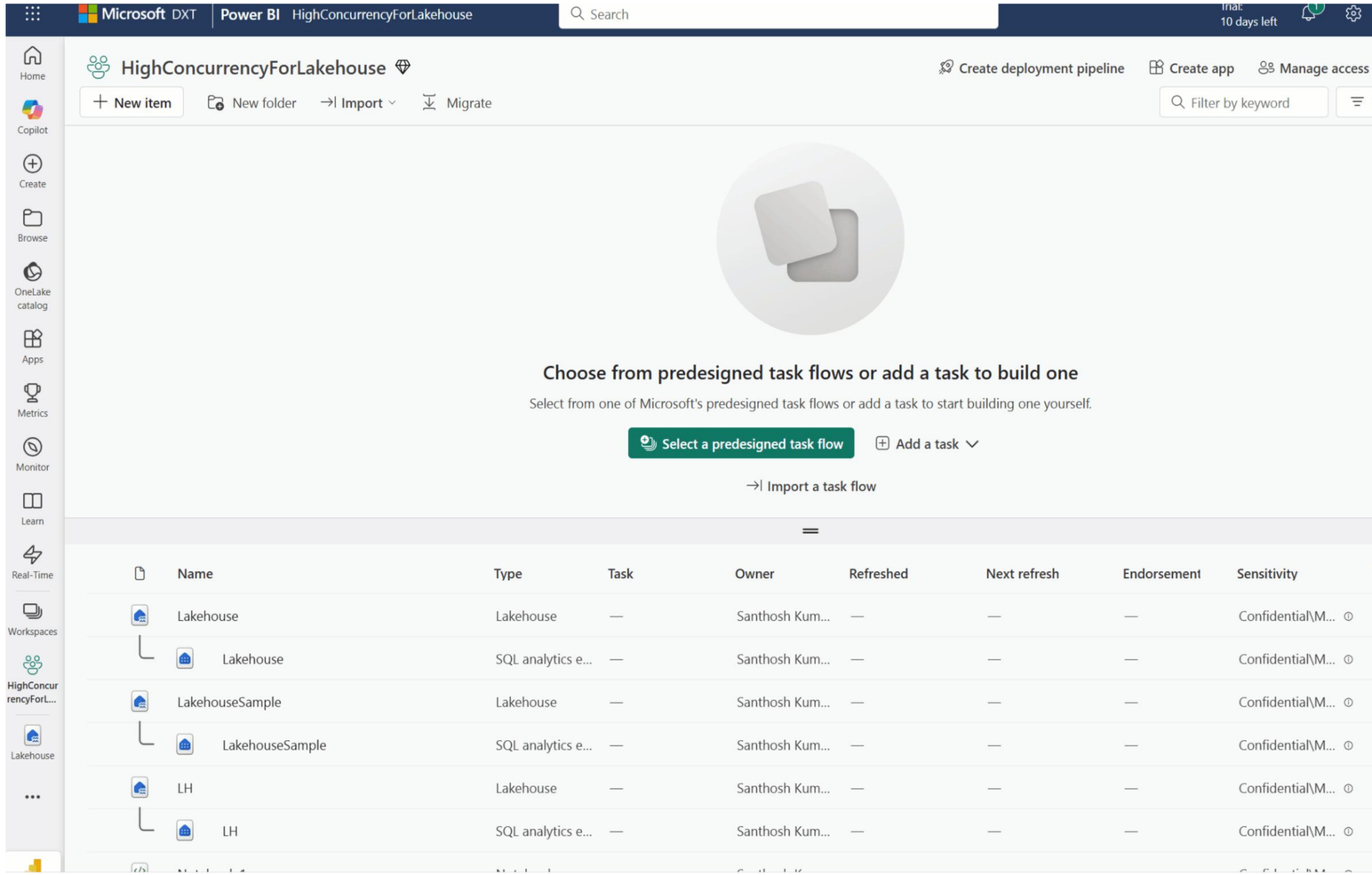
Breaking Down Cold Starts for Precise Troubleshooting

A visual decomposition of the session lifecycle in the Notebook and Monitoring Hub, categorizing every second of the startup process.

Identifying the "Fallback" Path:

- **Personalization Fallback:** Triggered when custom libraries or Spark properties are detected, requiring the system to configure the session post-creation.

- **Infrastructure Fallback:** Triggered when Starter Pools are unavailable (Regional traffic) or unsupported (Advanced Security), forcing an on-demand cluster spin-up.



Make job admission, session reuse, and queueing predictable

When users say Spark is slow, they are often describing policy choices: cold starts, full capacity, or a missed session-sharing path.

High concurrency

- Enabled by default for Fabric workspaces.
- Session sharing stays within a single user boundary and matching workspace / compute context.
- Only the initiating session is billed; shared followers are not billed separately.

Admission policy

- Optimistic admission is the default workspace behavior.
- Reserve maximum cores when predictability matters more than squeezing in more simultaneous work.
- Session timeout is also a workspace-level lever for idle interactive sessions.

Queueing

- Background notebook jobs and Spark job definitions can enter FIFO queueing.
- Queue expiry is 24 hours.
- Interactive notebook work is not queue-backed in the same way and can throttle instead.

Recommended default: leave high concurrency on, decide explicitly whether to reserve maximum cores, and teach users the difference between queued work and a cold start.

Monitoring and Troubleshooting Toolkit

- Capacity Metrics App provides visibility into CU consumption showing when usage exceeded purchased capacity thresholds
- Workspace-level job concurrency monitoring diagnoses delays by identifying concurrency limits or queueing bottlenecks for specific workspaces
- Throttling detection flow helps diagnose bottlenecks and manage CU limits effectively by showing which jobs are throttled or queued

Diagnosing & Resolving Throttling for Spark Jobs in Fabric

Workspace Level Job Concurrency Monitoring

Diagnose job delays by identifying concurrency limits or queueing bottlenecks

Throttling Detection & Flow

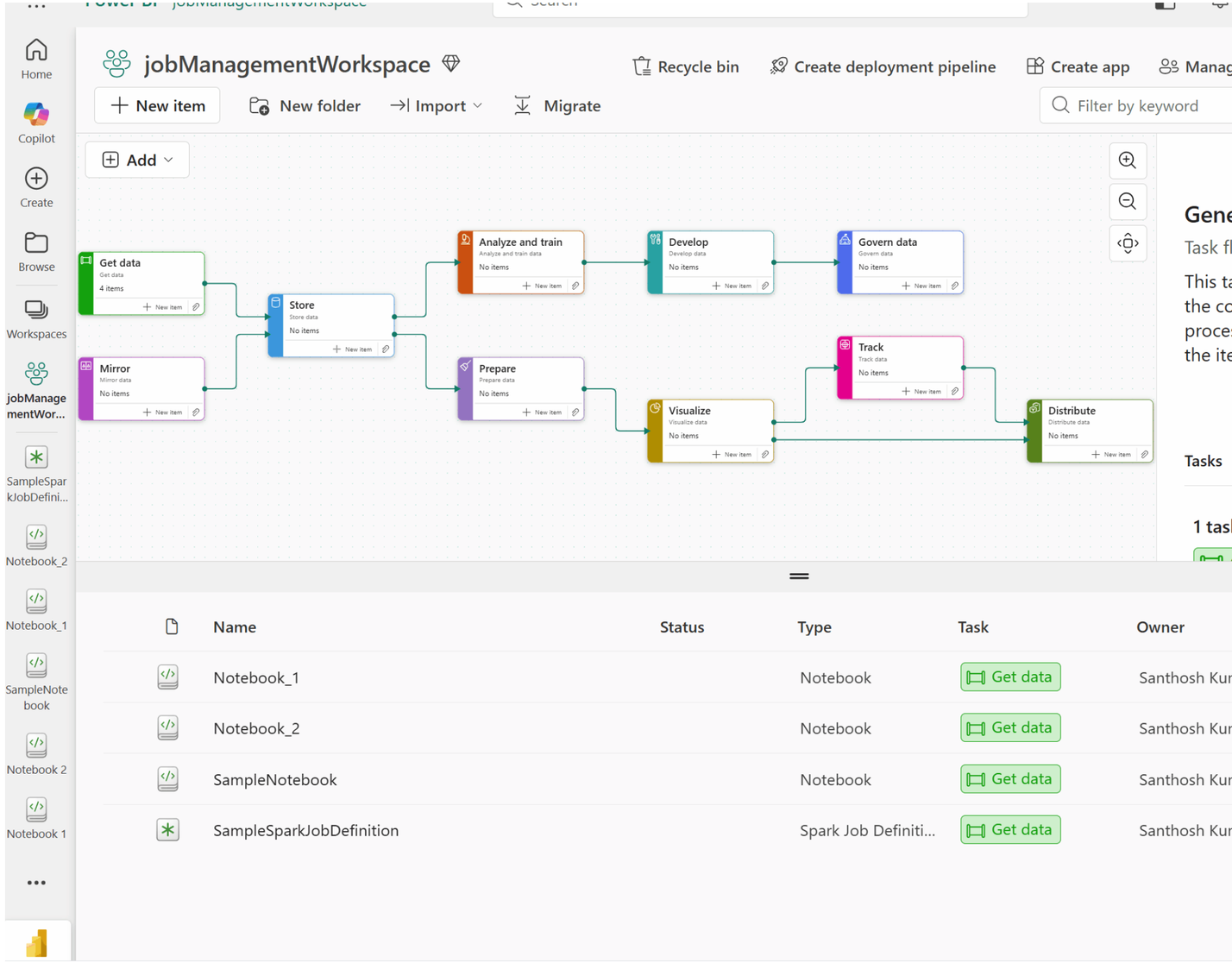
Jobs get throttled or queued when workloads exceed allocated capacity.

Helps diagnose **bottlenecks** and manage **CU limits** effectively.

Optimization Strategies

Adjust max CU limits to prevent throttling

Optimize job scheduling and workload distribution



Guarding Against Runaway Costs: Max Job Lifetime

Coming Soon

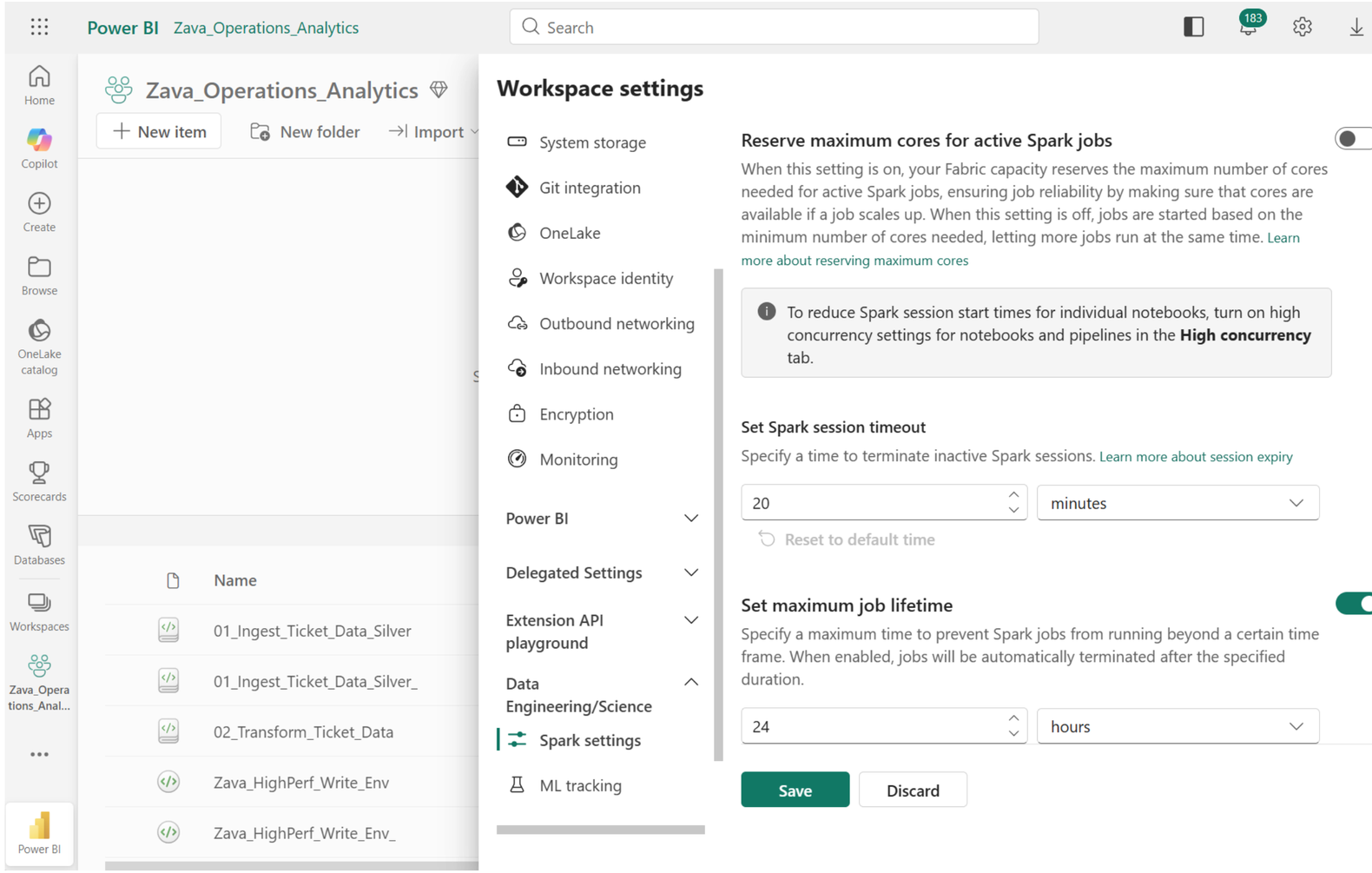
Enforcing Strict Execution Windows at the Workspace Level

A new workspace-level governance setting that allows admins to define a "Hard Ceiling" for the duration of any Spark job (e.g., Notebooks, Spark Job Definitions).

- **Preventing Runaway Jobs:** Provides a safety net against "stuck" sessions or unoptimized code (e.g., infinite loops, massive data skew) that would otherwise consume Capacity Units (CUs) indefinitely.

Strict Enforcement:

- Admins specify a time limit (e.g., **1 Hour**).
- Every user job in that workspace is bound by this limit.
- The system triggers an automatic termination the moment the 60-minute mark is hit, with no exceptions.



Home
Workspaces
Copilot
OneLake catalog
Monitor
Real-Time
Workloads
Zava_Product_Analyti...
Zava_Product_Analytics...
My workspace
...
Fabric

Zava_Product_Analytics_Workspace

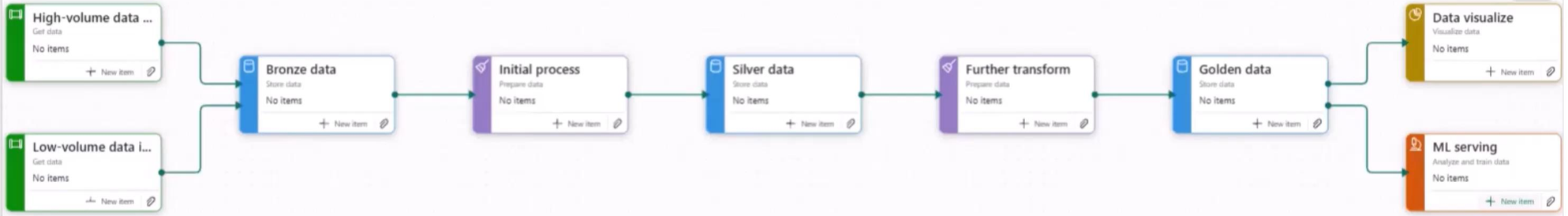
Zava Product Analytics Silver layer workspace for customer support processing

Recycle bin | Create deployment pipeline | Create app | Manage access | Workspace settings

New item | New folder | Import | Migrate

Filter by keyword | Filter

Medallion | Add task | Add a connector | Apply predesigned task flow



Medallion

Task flow details

Organize data in a lakehouse or warehouse while progressively improving its structure and quality through each layer, from Bronze to Silver to Gold, resulting in high-quality data that is easy to analyze.

Edit

Tasks

1 task

Name	Status	Type	Task	Owner	Refreshed	Next refresh	Endorsemer	Sensitivity	Included in app
gold_env		Environment	—	Santhosh Ku...	3/13/2026, 6:40:1...	3/13/2026, 7:10:0...	—	Confidential\...	
Zava_Product_Silver		Lakehouse	—	Santhosh Ku...	—	—	—	Confidential\...	
Zava_Product_Silver		SQL analytics ...	—	Santhosh Ku...	—	—	—	Confidential\...	
Zava_Silver_Tickets_Pipeline		Notebook	—	Santhosh Ku...	—	—	—	Confidential\...	

Phase 1

Standardize runtime, environment, pool defaults to prevent drift

Phase 2

Classify workloads to determine capacity versus autoscale

Phase 3

Apply profiles and validate with compare-runs for optimization

Phase 4

Operationalize monitoring so spikes are visible for proactive response

Actionable Rollout Plan

Standardize defaults, classify workloads, apply profiles, and operationalize monitoring to achieve predictable cost-controlled Spark execution

- Start with Starter Pools then upgrade to Custom Pools or Custom Live Pools
- Enable autoscale billing for bursty workloads while reserving capacity for production ETL
- Set Max Job Lifetime limits and adjust Starter Pools to enforce governance
- Build incident playbook mapping symptoms to actions for troubleshooting

Q & A

Sound off.
The mic is all yours.
Influence the product roadmap.

Join the Fabric User Panel

Share your feedback directly with our Fabric product group and researchers.

https://aka.ms/JoinFabricUserPanel

Join the SQL User Panel

Influence our SQL roadmap and ensure it meets your real-life needs

https://aka.ms/JoinSQLUserPanel

How was the session?

Complete Session Surveys in Whova for your chance to WIN PRIZES!