

Databricks Workflows vs Azure Data Factory (ADF): A 360-Degree Comparative Guide

Purpose of the Document

This document provides a comprehensive industry-level comparison between **Azure Data Factory (ADF)** and **Databricks Workflows**, aimed at helping data engineers, architects, and platform teams make informed decisions on when to use which tool, based on real-world use cases, strengths, and limitations.

Introduction to Tools

What is Azure Data Factory (ADF)?

ADF is a cloud-based data integration service from Microsoft Azure. It allows you to create data-driven workflows for orchestrating data movement and transformation at scale using a visual interface, a wide range of connectors, and low-code capabilities.

Ideal For:

- Integrating with 100+ data sources (on-prem, cloud, SaaS)
 - SAP/Oracle/SFTP ingestion
 - Enterprise data movement and low-code transformations
 - Cross-Azure orchestration
-

What is Databricks Workflows?

Databricks Workflows is a native orchestration feature within the Databricks Lakehouse Platform. It allows you to schedule and orchestrate complex data, analytics, and ML pipelines using Spark, SQL, Python, dbt, and other workloads.

Ideal For:

- Spark-native transformations
 - ML model orchestration with retry logic
 - Git-integrated pipeline development
 - Cost-efficient, compute-native orchestration
-

Core Use Case-by-Use Case Comparison

1. Ingesting from SAP, Oracle, or SFTP

Q: Can I use ADF for this?  Absolutely.

- ADF has **native connectors** for SAP (ECC, BW, HANA, OData, Table), Oracle (via ODP.NET or SHIR), and SFTP.
- Comes with built-in support for authentication, retries, logging, pagination, and monitoring.

Q: Can I use Databricks Workflows for this? ⚠️ Technically possible but not recommended.

- No native SAP/Oracle connectors.
- You must manually manage JDBC drivers and secrets.
- Networking (VNet, NAT Gateway) and authentication are complex.
- No built-in UI for monitoring file ingestion.

Recommendation: Use **ADF for ingestion** into ADLS (bronze zone). Let Databricks handle transformation afterward.

2. Orchestrating Spark Notebooks

Q: Can ADF do this? ⚠️ Yes, but only via REST API integration.

- Limited visibility and control over Spark job status.
- Not ideal for tightly coupled Spark DAGs.

Q: Can Databricks Workflows do this? ✅ Natively and optimally.

- Directly runs notebooks as tasks.
- Supports retry logic, parameters, Git integration.
- High control over Spark clusters and environment variables.

Recommendation: Prefer **Databricks Workflows** for orchestrating Spark code and business logic.

3. dbt Job Orchestration

Q: Can ADF run dbt? ⚠️ Yes, using CLI via Azure Batch or Azure Functions.

- Setup is indirect and fragile.
- Limited visibility into dbt run results.

Q: Can Databricks Workflows run dbt? ✅ Yes, with native support for dbt-core and dbt-databricks.

- Clean Git integration.
- First-class support for dbt SQL/Notebook workflows.

Recommendation: For dbt projects, **Databricks Workflows is preferred**.

4. Cross-Service Orchestration (e.g., Synapse, Azure Functions)

Q: Can ADF orchestrate external Azure services? ✅ Yes.

- Direct activities for Synapse, Functions, Logic Apps, HDInsight, etc.

Q: Can Databricks Workflows do this? ⚠️ Possible but only using REST API calls or webhooks.

- Limited visibility and hard-coded.

Recommendation: Use **ADF** for orchestration that spans multiple Azure services.

5. GitOps and Parameterized DAGs

Q: Does ADF support parameterized and version-controlled workflows? ⚠️ Yes but limited.

- Parameterization is supported.
- Git integration exists but is not native and CI/CD is brittle.

Q: How does Databricks support GitOps? ✅ Fully.

- Git-based development workflows.
- Task-level parameters, dynamic configuration, and environment promotion.

Recommendation: Use **Databricks Workflows** when version control, parameterized jobs, and dev/test/prod environments are key.

6. Cost Control (Avoiding Extra Orchestration Layer)

Q: Does ADF incur orchestration cost? ❌ Yes.

- Separate pricing for pipeline execution, IR runtimes.
- Extra latency for notebook triggering.

Q: What about Databricks Workflows? ✅ No extra cost.

- Orchestration is embedded in the compute plan.
- Job clusters and task-level retries help optimize runtime.

Recommendation: For cost-sensitive in-lake workflows, prefer **Databricks Workflows**.

7. Spark + ML Pipelines with Retry Logic

Q: Does ADF support ML retry logic? ⚠️ Not natively.

- Requires custom logic.
- Poor integration with MLFlow.

Q: Does Databricks support ML pipelines? ✅ Yes.

- Built-in integration with MLflow.
- Retries, alerting, model registry — all first-class.

Recommendation: Use **Databricks Workflows** for ML pipelines and Spark-native retrievable workflows.

8. SSIS Migration with Low-Code Flows

Q: Can ADF replace SSIS packages?  Yes.

















- Provides data flows and transformations with drag-and-drop.
- Supports SQL Server, Excel, flat files, stored procs.

Q: Can Databricks do this?  No.

- Requires full coding.
- No visual designer.

Recommendation: For SSIS modernization, **ADF is purpose-built.**

Summary Table: When to Use What

Use Case	Use ADF	Use Databricks Workflows
Ingest from SAP/Oracle/SFTP		 (custom, limited)
Orchestrate Spark Notebooks	 (API)	 (native, fast)
dbt Job Orchestration	 (CLI)	
Cross-Service Orchestration		 (manual APIs)
GitOps + Parameterized DAGs		
Cost-Efficient Orchestration		
ML/Spark Pipelines with Retry Logic		
SSIS Low-Code Migration		

Best Practices

For Azure Data Factory

- Use ADF to ingest structured/unstructured data from 100+ sources.
- Use SHIR for secure on-premises source ingestion.
- Parameterize pipelines for reusable deployments.
- Monitor using Azure Monitor and activity runs.

For Databricks Workflows

- Use job clusters to reduce costs.
 - Enable task retries and alerts.
 - Chain tasks with Task Values and conditional branching.
 - Integrate Git for CI/CD.
 - Prefer Workflows for Spark/dbt/ML orchestration.
-

Pro Tip: Use a Hybrid Approach

The **best practice** in modern architectures is not to choose one tool over another but to **combine both for their strengths**:

- **ADF**: Ingest and stage data from SAP, Oracle, SFTP, and APIs into ADLS.
- **Databricks Workflows**: Transform, enrich, validate, and prepare for analytics/ML.

This approach results in:

- Simplified ingestion
 - Optimized Spark performance
 - Reduced orchestration cost
 - Scalable, governed pipelines across domains
-

Final Thoughts

Choosing between ADF and Databricks Workflows depends on your workload. Use this guide to evaluate your architecture, governance needs, and operational costs before making a choice. When in doubt, build small POCs and adopt a hybrid architecture.

Let's design pipelines that are not only functional, but **scalable, cost-effective, and reliable**.