



Deploy Workloads with Databricks Workflows



Module Agenda

Deploy Workloads with Databricks Workflows

Introduction to Workflows

Building and Monitoring Workflow Jobs

DE 6.1 – Scheduling Tasks with the Jobs UI

DE 6.2L – Jobs Lab

DE 6.3 – OPTIONAL Navigating Databricks SQL

DE 6.4 – OPTIONAL Last Mile ETL with DBSQL

Introduction to Workflows



Course Objectives

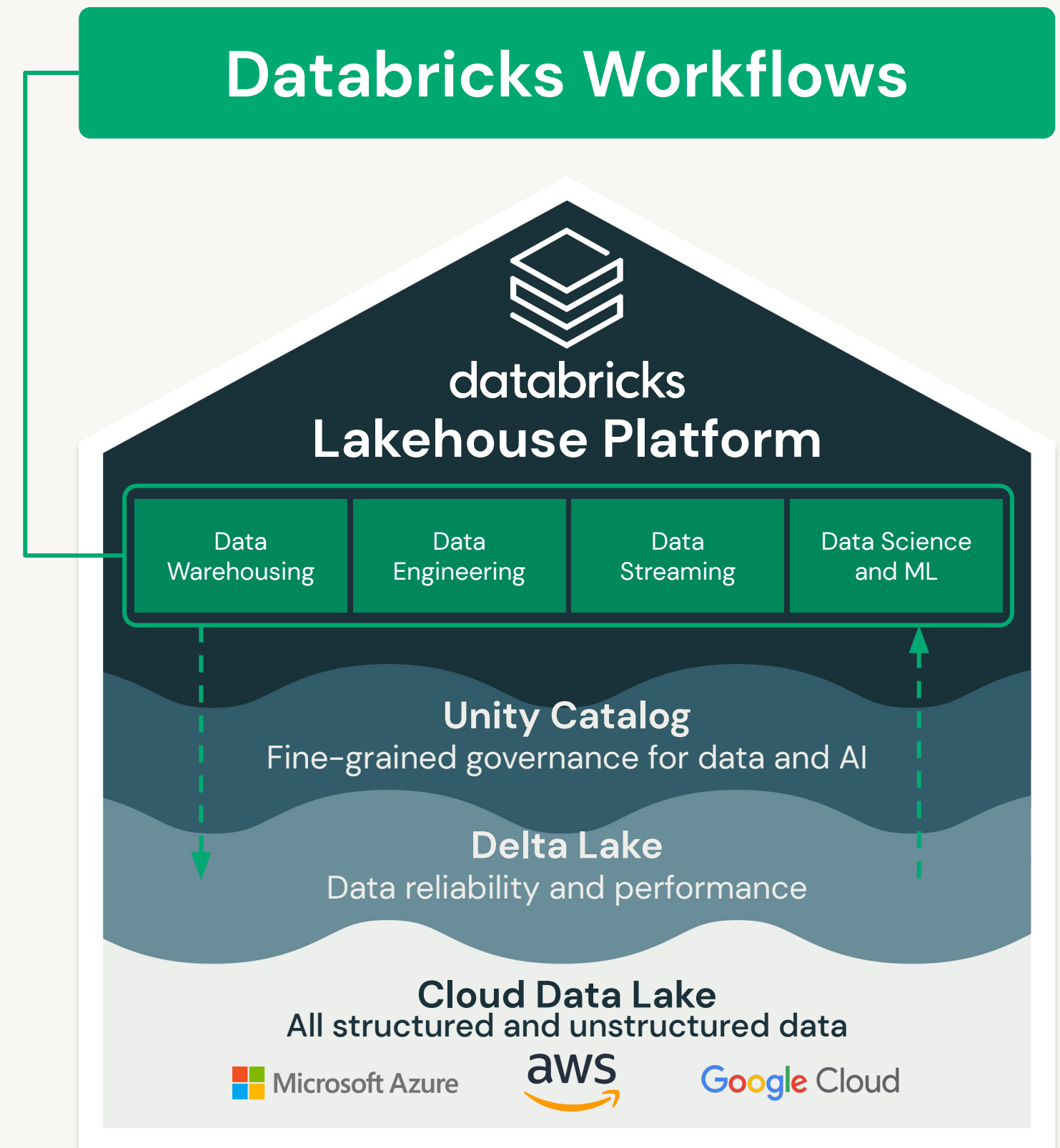
- 1 Describe the main features and use cases of Databricks Workflows
- 2 Create a task orchestration workflow composed of various task types
- 3 Utilize monitoring and debugging features of Databricks Workflows
- 4 Describe workflow best-practices



Introduction to Workflows

Workflows is a **fully-managed cloud based general purpose task orchestration service** for the entire Lakehouse.

Workflows is a service for data engineers, data scientists and analysts to build reliable data, analytics and AI workflows on any cloud.



Introduction to Workflows

Databricks Workflows has two main task orchestration services;

- **Workflow Jobs (Workflows):** Workflows for every job.
- **Delta Live Tables (DLT):** Automated data pipelines for Delta Lake

DLT pipeline can be a task in a Workflow.



Introduction to Workflows

Use Cases

Orchestration of Dependent Jobs

Jobs running on schedule, containing dependent tasks/steps

Jobs Workflows

Machine Learning Tasks

Run MLflow notebook task in a job

Jobs Workflows

Arbitrary Code, External API Calls, Custom Tasks

Run tasks in a job which can contain Jar file, Spark Submit, Python Script, SQL task, dbt

Jobs Workflows

Data Ingestion and Transformation

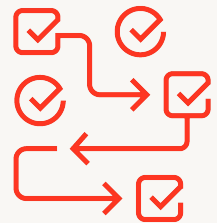
ETL jobs, Support for batch and streaming, Built in data quality constraints, monitoring & logging

Delta Live Tables



Introduction to Workflows

Features



Orchestrate Anything Anywhere

Run diverse workloads for the full data and AI lifecycle, on any cloud. Orchestrate;

- Notebooks
- Delta Live Tables
- Jobs for SQL
- ML models, and more.



Fully Managed

Remove operational overhead with a fully managed orchestration service enabling you to focus on your workflows not on managing your infrastructure.

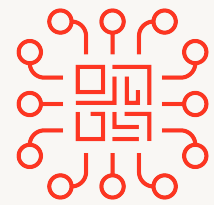


Simple Workflow Authoring

An easy point-and-click authoring experience for all your data teams not just those with specialized skills.

Introduction to Workflows

Features



Deep Platform Integration

Designed and built into your lakehouse platform giving you deep monitoring capabilities and centralized observability across all your workflows.



Proven Reliability

Have full confidence in your workflows leveraging our proven experience running tens of millions of production workloads daily across AWS, Azure, and GCP.

Introduction to Workflows

How to Leverage Workflows

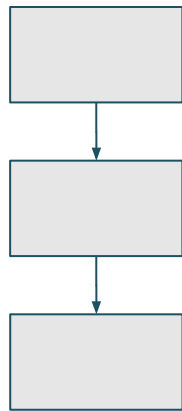
- Allows you to build simple ETL/ML task orchestration
- Reduces infrastructure overhead
- Easily integrate with external tools
- Enables non-engineers to build their own workflows using simple UI
- Cloud-provider independent
- Enables re-using clusters to reduce cost and startup time



Introduction to Workflows

Common Workflow Patterns

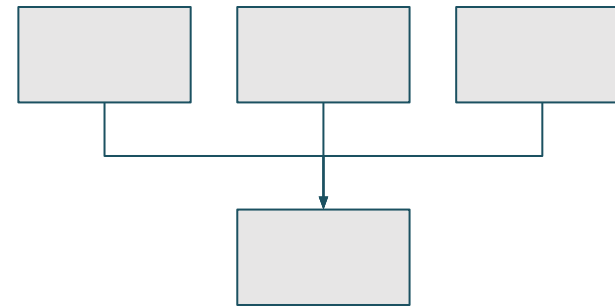
Sequence



Sequence

- Data transformation/processing/cleaning
- Bronze/silver/gold tables

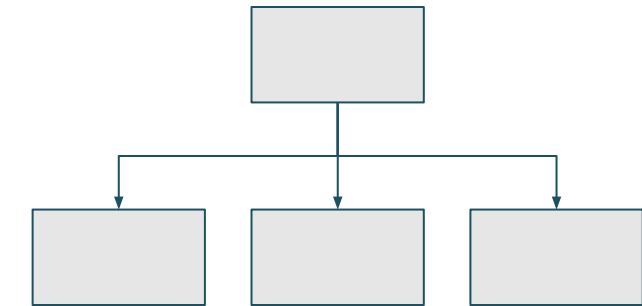
Funnel



Funnel

- Multiple data sources
- Data collection

Fan-out



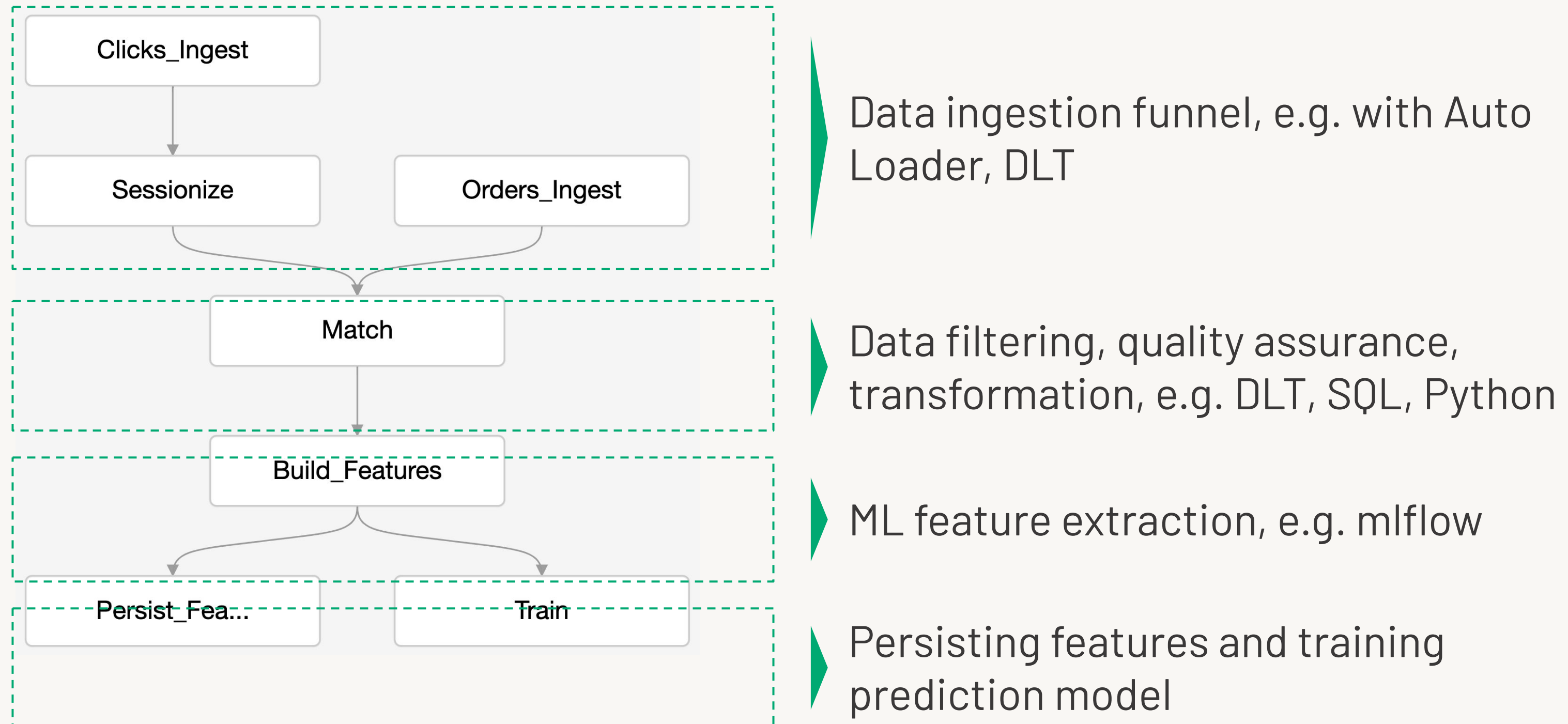
Fan-out, star pattern

- Single data source
- Data ingestion and distribution



Introduction to Workflows

Example Workflow

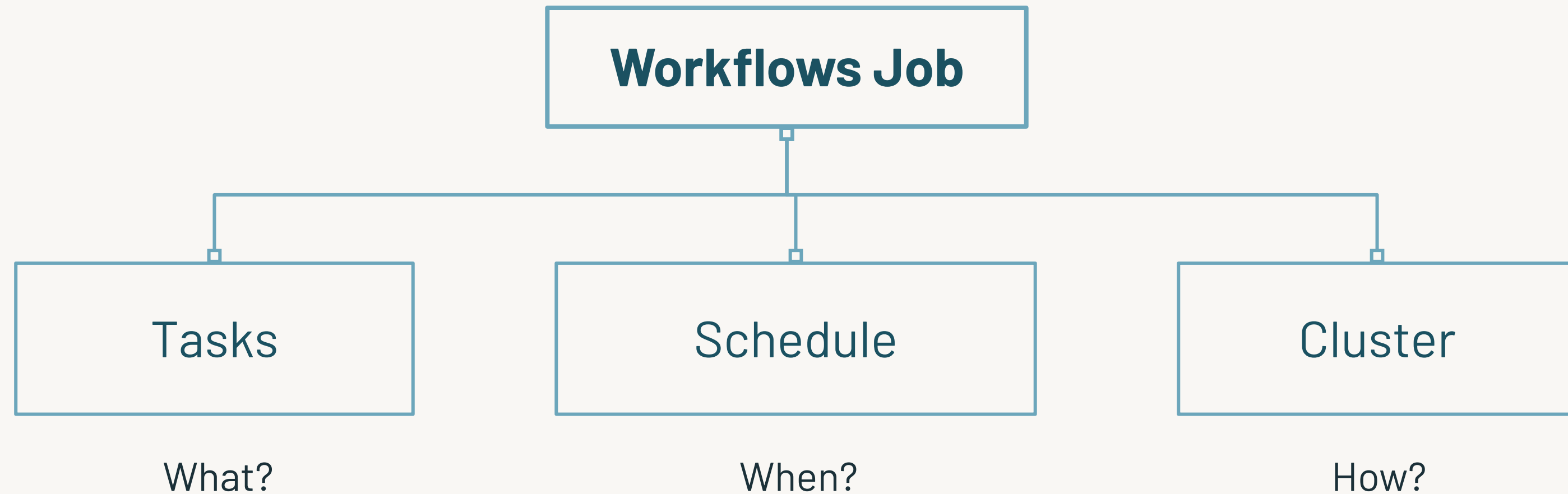


Building and Monitoring Workflow Jobs



Introduction to Workflows

Workflow Components

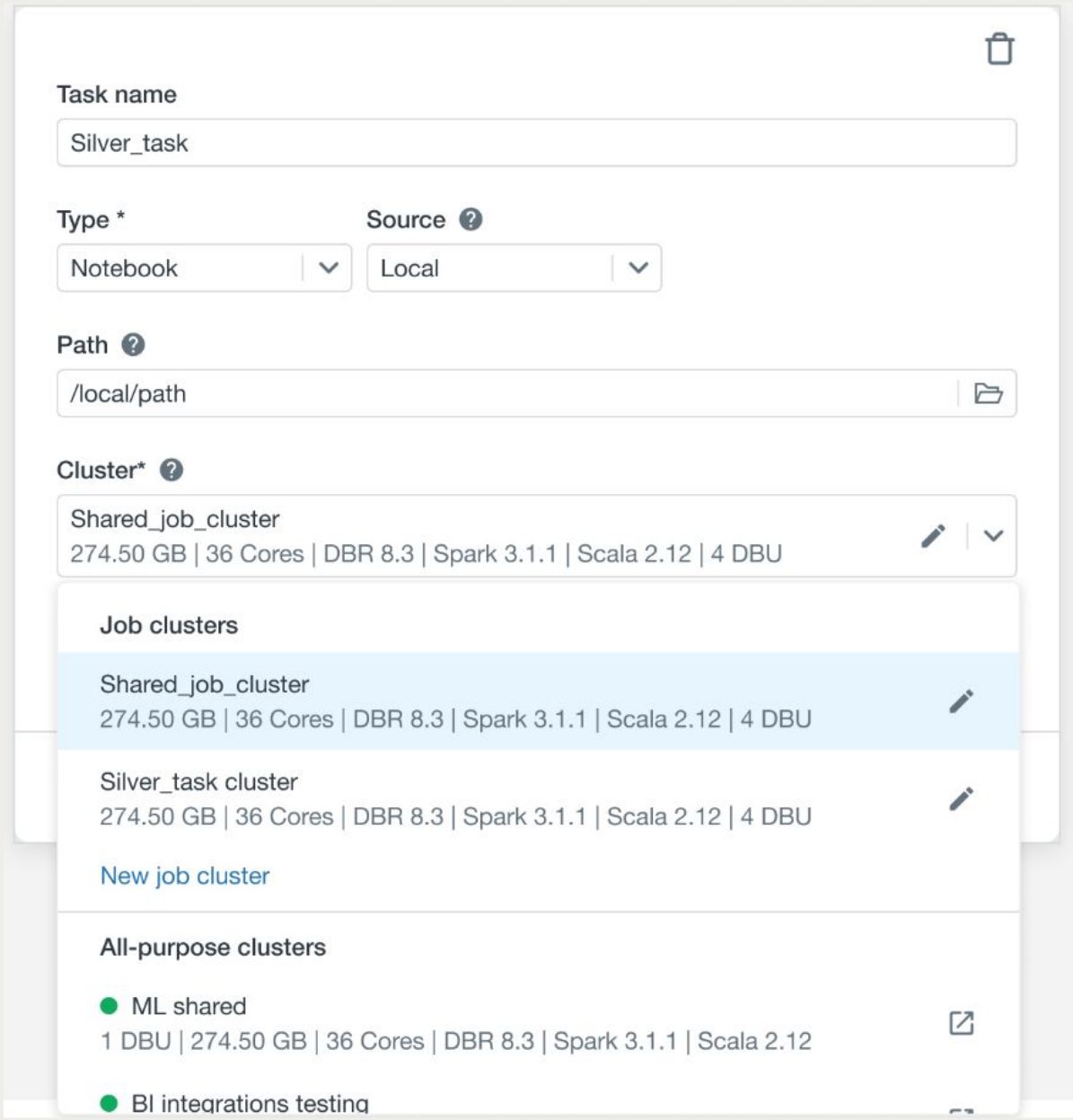


Creating a Workflow

Task Definition

While creating a task;

- Define the task type
- Choose the cluster type
 - Job clusters and All-purpose clusters can be used.
 - A cluster can be used by multiple tasks. This reduces cost and startup time.
- If you want to create a new cluster, you must have required permissions.
- Define task dependency if task depends on another task



The screenshot shows the 'Task Definition' form in Databricks. The 'Task name' field is 'Silver_task'. The 'Type' is set to 'Notebook' and the 'Source' is 'Local'. The 'Path' is '/local/path'. The 'Cluster' is 'Shared_job_cluster' with specifications: 274.50 GB | 36 Cores | DBR 8.3 | Spark 3.1.1 | Scala 2.12 | 4 DBU. A dropdown menu is open, showing 'Job clusters' and 'All-purpose clusters'. Under 'Job clusters', 'Shared_job_cluster' and 'Silver_task cluster' are listed with their specifications. There is a 'New job cluster' link. Under 'All-purpose clusters', 'ML shared' and 'BI integrations testing' are listed.

Cluster Type	Cluster Name	Specifications
Job clusters	Shared_job_cluster	274.50 GB 36 Cores DBR 8.3 Spark 3.1.1 Scala 2.12 4 DBU
Job clusters	Silver_task cluster	274.50 GB 36 Cores DBR 8.3 Spark 3.1.1 Scala 2.12 4 DBU
All-purpose clusters	ML shared	1 DBU 274.50 GB 36 Cores DBR 8.3 Spark 3.1.1 Scala 2.12
All-purpose clusters	BI integrations testing	



Monitoring and Debugging

Scheduling and Alerts

You can run your jobs **immediately** or **periodically** through an easy-to-use scheduling system.

You can specify alerts to be notified when runs of a job **begin, complete or fail**. Notifications can be sent via email, Slack or AWS SNS.

The screenshot displays the 'Schedule' and 'Alerts' sections of a Databricks job configuration page. In the 'Schedule' section, 'Manual (Paused)' is unselected and 'Scheduled' is selected. The frequency is set to 'Every Day' at '15:50' in '(UTC+02:00)' time. A 'Show cron syntax' checkbox is present. The 'Alerts' section lists two email addresses: 'admin@anycompany.com' and 'notification@databricks.com'. For 'admin@anycompany.com', 'Start', 'Success', and 'Failure' alerts are all checked. For 'notification@databricks.com', only the 'Failure' alert is checked. A checkbox for 'Do not send alerts for skipped runs' is at the bottom.

Schedule Type

☐ Manual (Paused)

☒ Scheduled

Schedule ?

Every at :

☐ Show cron syntax

Alerts ?

<input type="text" value="admin@anycompany.com"/>	<input checked="" type="checkbox"/> Start	<input checked="" type="checkbox"/> Success	<input checked="" type="checkbox"/> Failure	<input type="button" value="x"/>
<input type="text" value="notification@databricks.com"/>	<input type="checkbox"/> Start	<input type="checkbox"/> Success	<input checked="" type="checkbox"/> Failure	<input type="button" value="x"/>

☐ Do not send alerts for skipped runs







Monitoring and Debugging

Access Control

Workflows integrates with existing resources access controls, enabling you to easily manage access across different teams.

Permission Settings for:
Task-1

NAME	PERMISSION
 	<div>Is Owner</div> <div>✕</div>
 admins	<div>Can Manage</div> <div>inherited</div>
 users	<div>Can View</div> <div>✕</div>

Select User, Group or Service Principal...

Is Owner

+ Add

Cancel

Save



Monitoring and Debugging

Job Run History

Workflows keeps track of job runs and save information about the success or failure of each task in the job run.



Monitoring and Debugging

Repair a Failed Job Run

Repair feature allows you to re-run only the failed task and sub-tasks, which reduces the time and resources required to recover from unsuccessful job runs.

