



# Deploy Workloads with Databricks Workflows

---

Module 05



# Module Agenda

## Deploy Workloads with Databricks Workflows

Introduction to Workflows





Building and Monitoring Workflow Jobs

DE 5.1 – Scheduling Tasks with the Jobs UI

DE 5.2L – Jobs Lab

# Introduction to Workflows

# Course Objectives

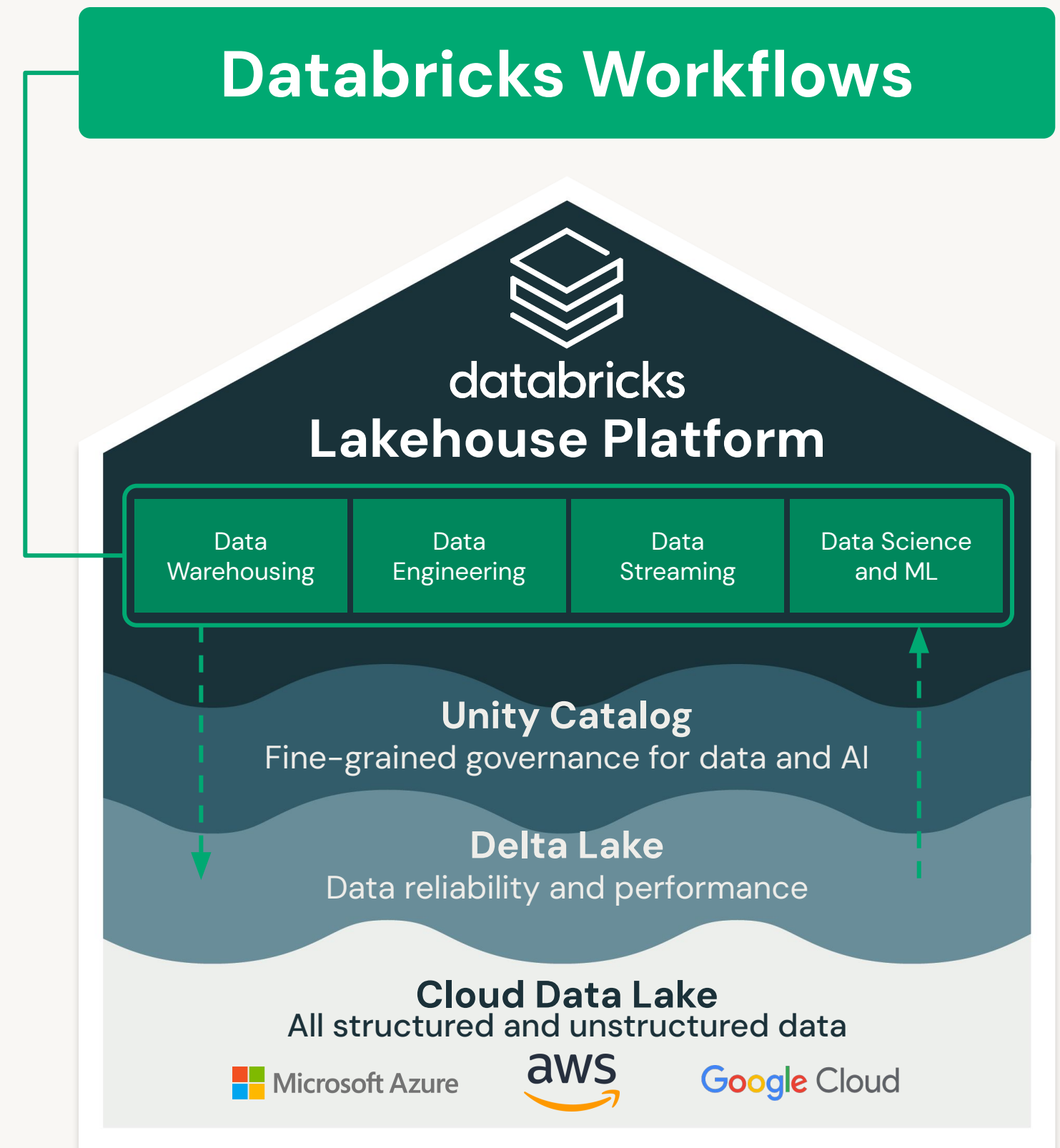
-  1 Describe the main features and use cases of Databricks Workflows
-  2 Create a task orchestration workflow composed of various task types
-  3 Utilize monitoring and debugging features of Databricks Workflows
-  4 Describe workflow best practices



# Databricks 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.

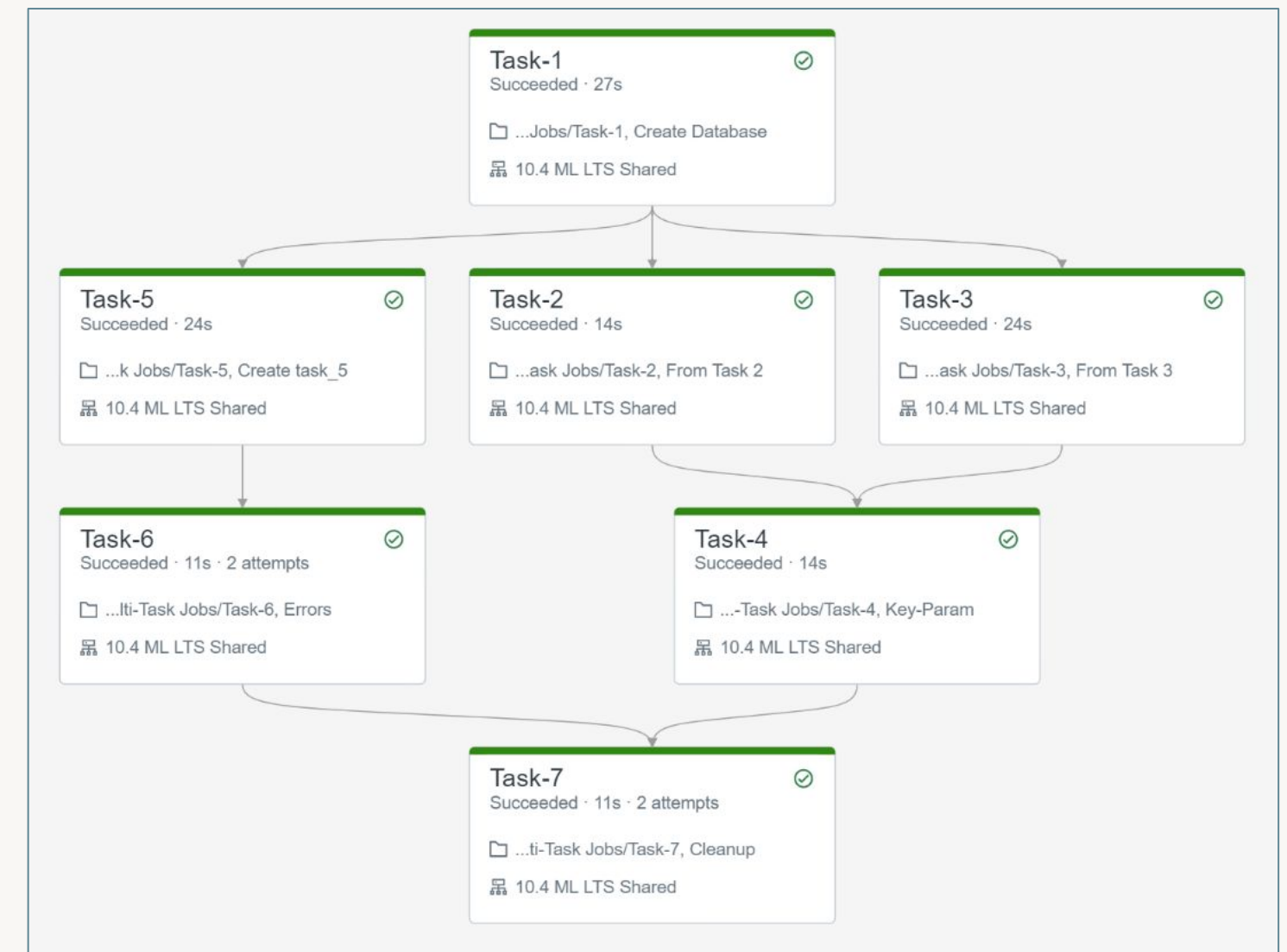


# Databricks Workflows

Databricks has two main task orchestration services:

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

Note: DLT pipeline can be a task in a workflow



# DLT versus Workflow Jobs

## Considerations

	Delta Live Tables	Workflow Jobs
Source	Notebooks only	JARs, notebooks, DLT, application written in Scala, Java, Python
Dependencies	Automatically determined	Manually set
Cluster	Self-provisioned	Self-provisioned or existing
Timeouts and Retries	Timeouts not supported Retries handled automatically (in production mode)	Supported
Import Libraries	Not supported	Supported



# DLT versus Workflow Jobs

## Considerations

	Delta Live Tables	Workflow Jobs
Source	Notebooks only	JARs, notebooks, DLT, application written in Scala, Java, Python
Dependencies	Automatically determined	Manually set
Cluster	Self-provisioned	Self-provisioned or existing
Timeouts and Retries	Supported	Supported
Import Libraries	Not supported	Supported





# DLT versus Jobs

## 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

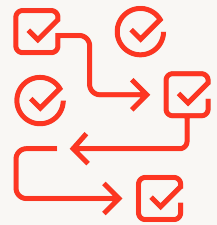
### Data Ingestion and Transformation

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

Delta Live Tables

# Workflows Features

## Part 1 of 2



### 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

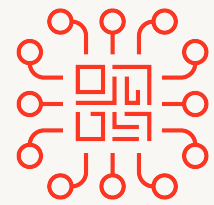


### Simple Workflow Authoring

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

# Workflows Features

## Part 2 of 2



### 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

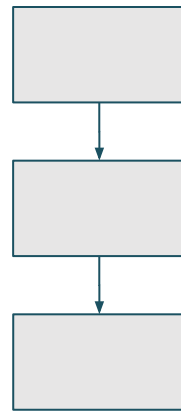
# How to Leverage Workflows

- 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



# Common Workflow Patterns

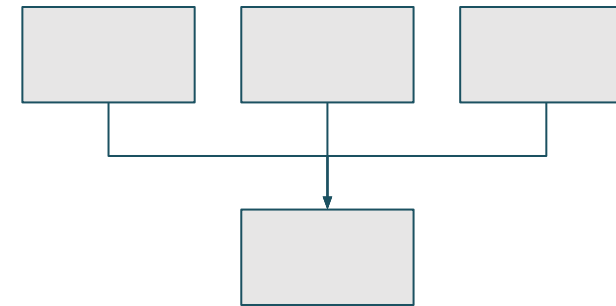
## Sequence



### Sequence

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

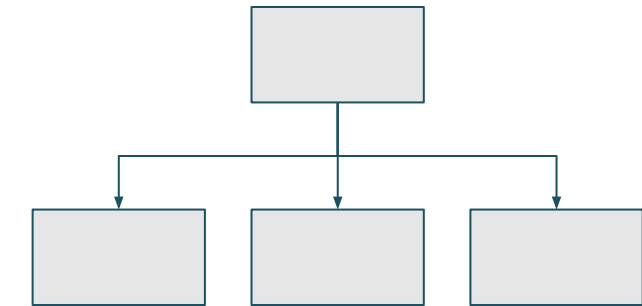
## Funnel



### Funnel

- Multiple data sources
- Data collection

## Fan-out

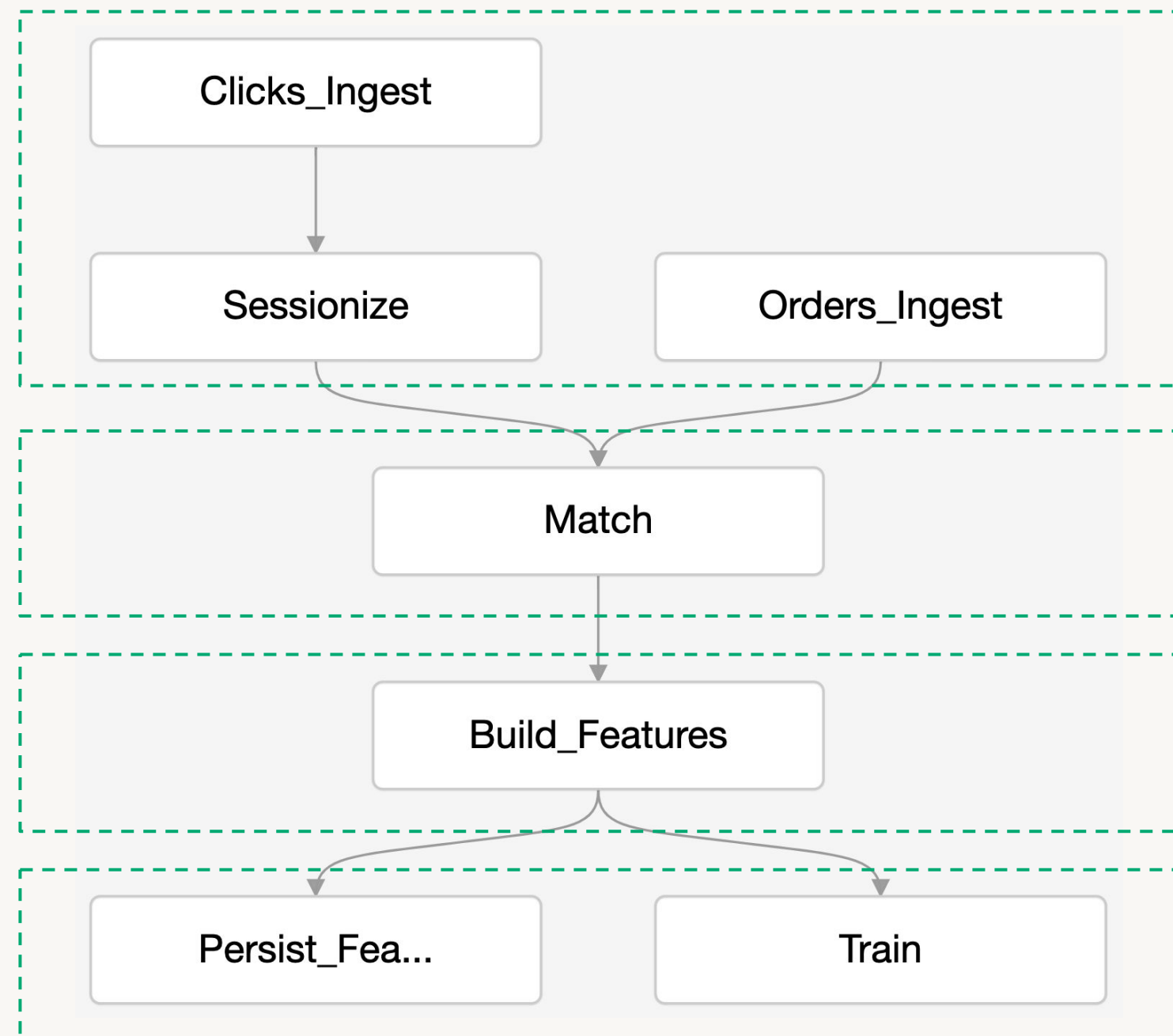


### Fan-out, star pattern

- Single data source
- Data ingestion and distribution



# Example Workflow



## Data ingestion funnel

E.g. Auto Loader, DLT

## Data filtering, quality assurance, transformation

E.g. DLT, SQL, Python

## ML feature extraction

E.g. MLflow

## Persisting features and training prediction model

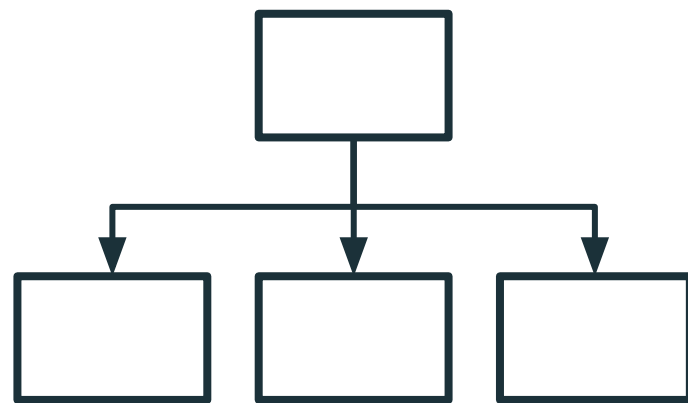




# Building and Monitoring Workflow Jobs

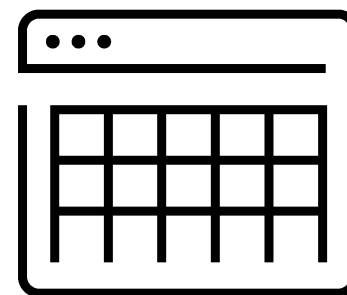
# Workflow Job Components

## TASKS



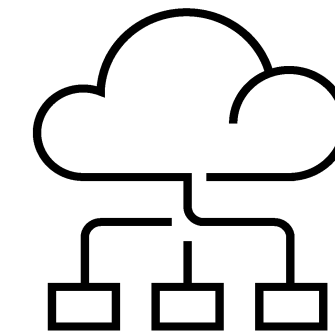
What?

## SCHEDULE



When?

## CLUSTER



How?

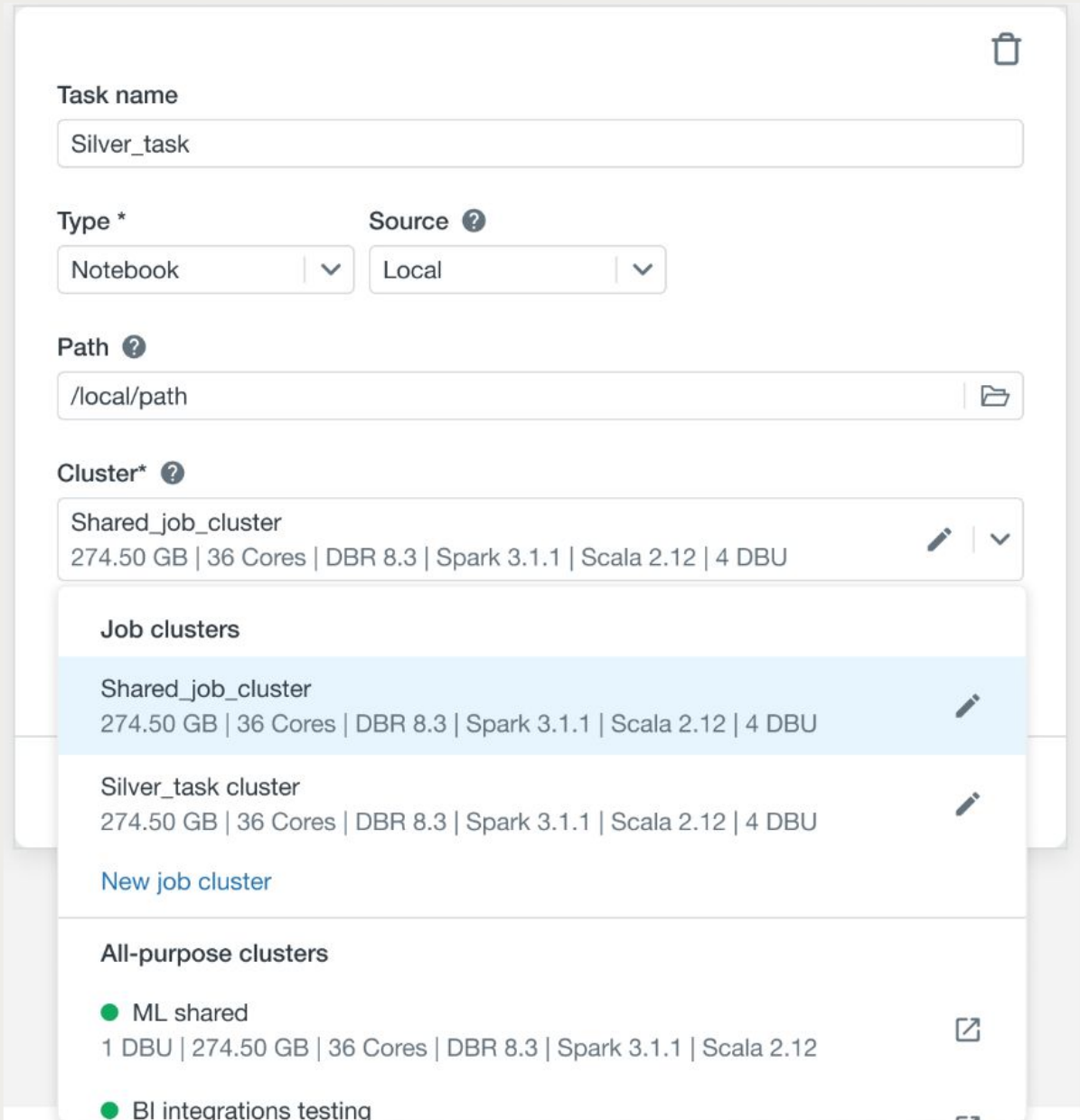


# Creating a Workflow

## 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
	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
	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. 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

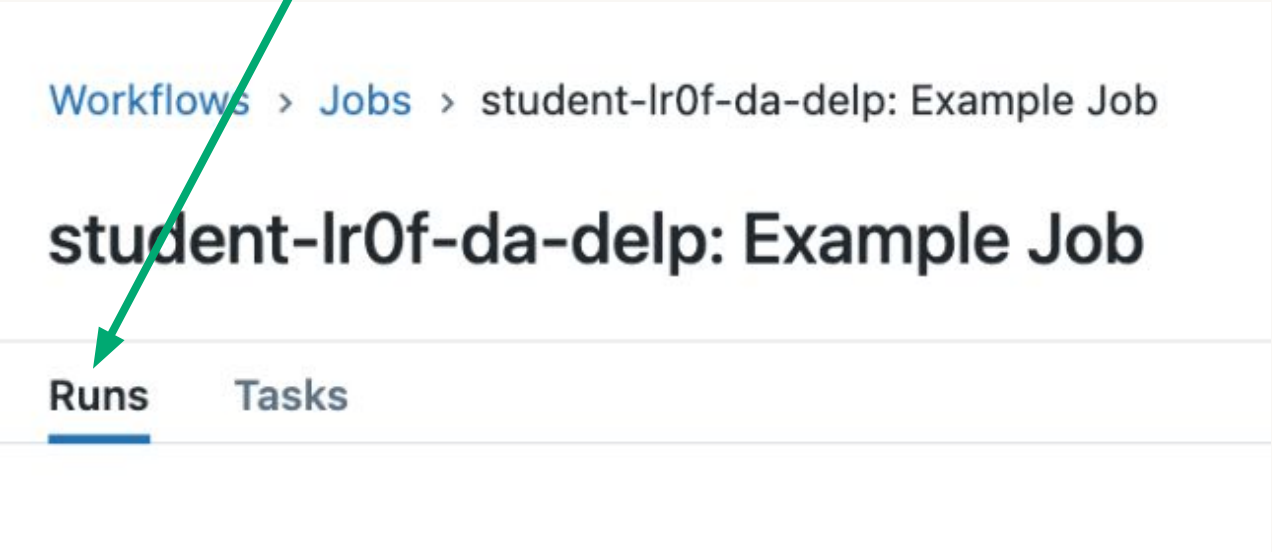


# Monitoring and Debugging

## Job Run History

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

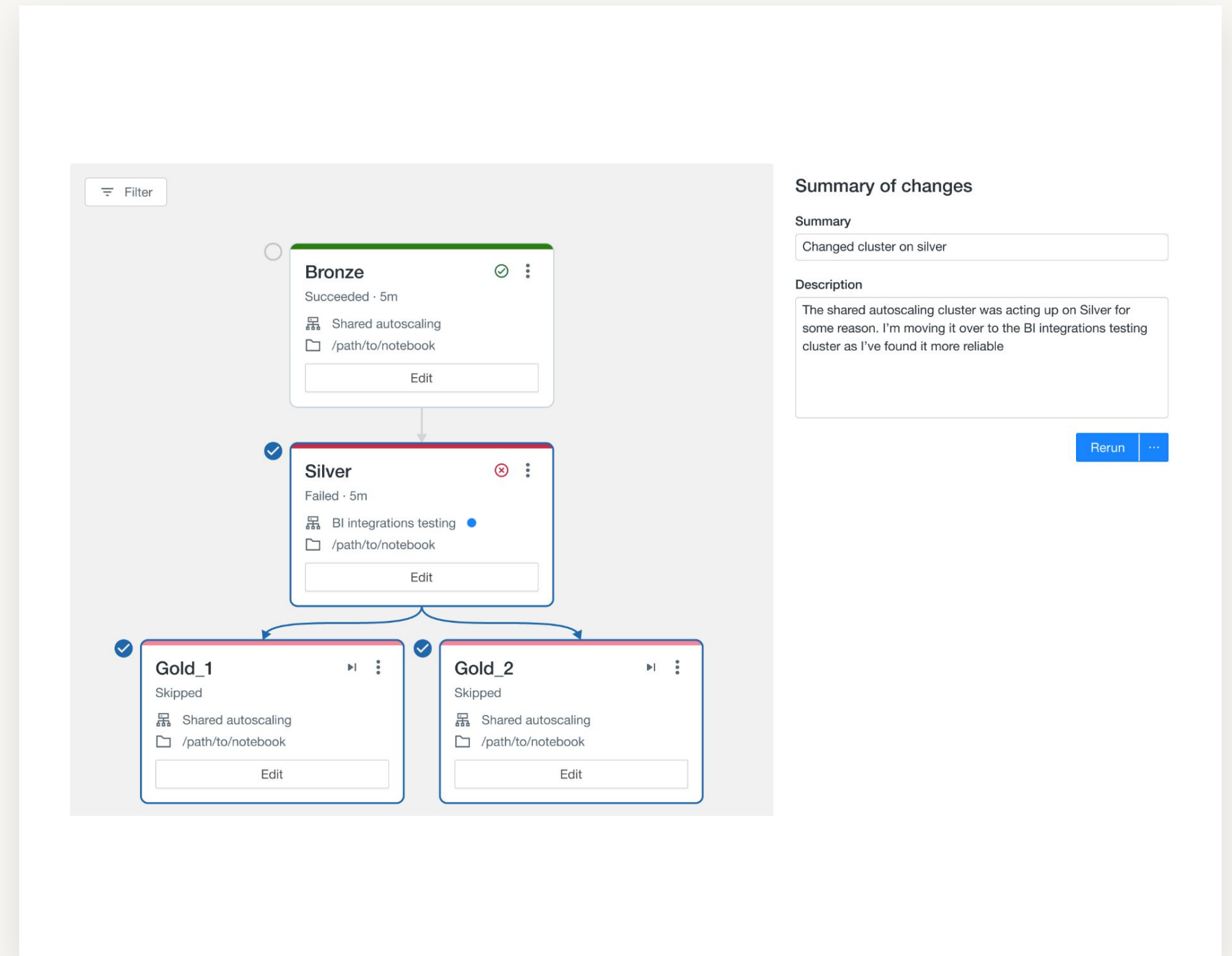
Navigate to the **Runs** tab to view completed or active runs for a job.



# 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.





# Navigating the Jobs UI

Use breadcrumbs to navigate back to your **job** from a specific run page

Workflows > Jobs > **student-lr0f-da-delp: Example Job** > Run 92643

## student-lr0f-da-delp: Example Job run

**Reset** ✓  
Succeeded · 18s  
/Users/student@databricks.com/da...  
0106-053850-f9hh0spa

→

**DLT** ✓  
Succeeded · 4m 1s  
Delta Live Table Pipeline

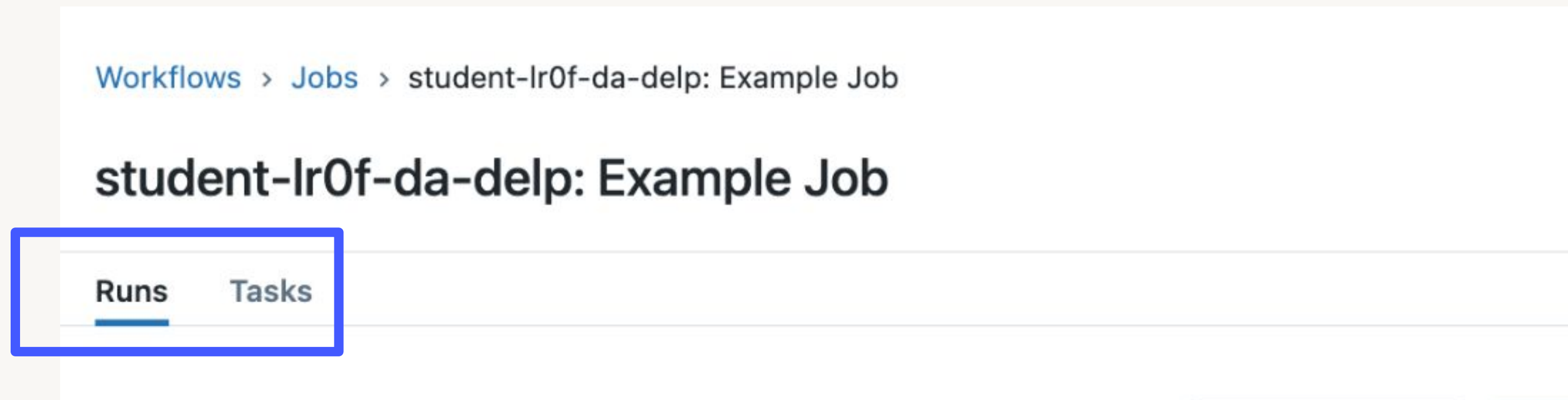
**Job run details**  
  
Job ID  
[354156443889724](#)   
  
Job run ID  
92643   
  
Started  
2023-01-06 00:49:55 EST  
  
Ended

# Navigating the Jobs UI

## Runs vs Tasks tabs on the job page

Use **Runs** tab to view completed or active runs for the job

Use **Tasks** tab to modify or add tasks to the job



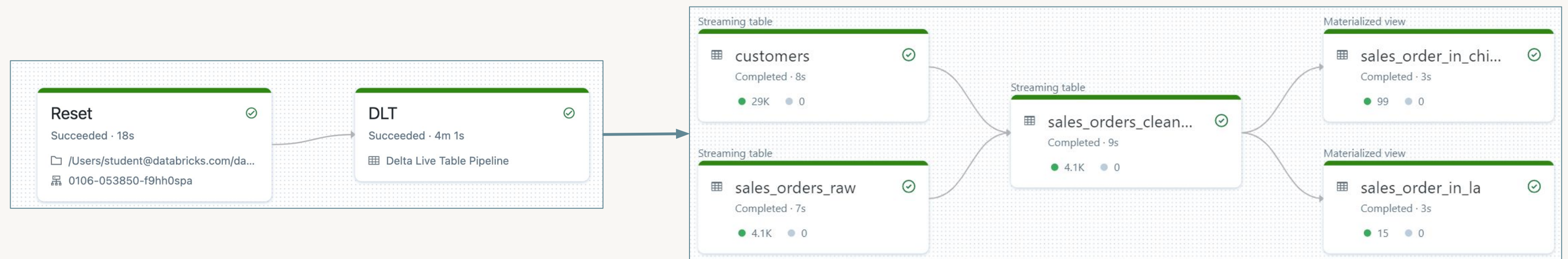
# DE 5.1.1: Task Orchestration



# Demo: Task Orchestration

## DE 5.1.1 – Task Orchestration

- Schedule a notebook task in a Databricks Workflow Job
- Describe job scheduling options and differences between cluster types
- Review Job Runs to track progress and see results
- Schedule a DLT pipeline task in a Databricks Workflow Job
- Configure dependency between tasks via Databricks Workflows UI



# DE 5.2.1.L: Task Orchestration Lab

# Lab: Task Orchestration

## DE 5.2.1.L – Task Orchestration

