The Developer’s Guide to the Ingestion Framework

# Introduction

This guide provides a step-by-step approach to setting up and configuring a Databricks ETL ingestion framework that leverages Delta Live Tables (DLT) for efficient, scalable data pipelines. You’ll learn how to set up a workflow that automates data ingestion, create configurations dynamically from catalog and schema information, and manage pipelines in Databricks. With this guide, developers can quickly deploy a robust data ingestion framework using reusable templates, optimized workflows, and well-organized configurations—empowering teams to seamlessly turn raw data into valuable insights.

# Getting Started

This section will walk you through the initial setup and prerequisites needed before configuring the Databricks ingestion framework. By following these steps, you'll ensure that your Databricks workspace is ready to support a Delta Live Tables (DLT) pipeline and associated workflows for data ingestion.

### 1. Set Up Databricks Workspace

* **Access the Workspace**: If you don't already have access to the Sandbox or Dev Databricks workspace, request access.
  + The workspaces are behind a VNET so a VPN or AWS Workspace will be needed
  + Databricks permissions are managed via AD Groups, so an advantage account will need to be setup & configured
* **Cluster Configuration**: Ensure you have a cluster available for testing and running your pipeline. These should already be created but tickets or allocation may be limited. A request can be made to increase

### 2. Access Permissions and User Roles

* **User Roles**: Make sure you have the necessary permissions (Workspace Admin or Contributor) to create and run workflows, DLT pipelines, and manage assets within the workspace.
* **Access Control for Data Sources**: Verify that you have access to the data sources that will be ingested. Set up connection credentials and permissions as required for cloud storage (e.g., AWS S3, Azure Blob), databases, or other data repositories.

### 3. Access the Ingestion Framework for Templates

* **Access Template Repository**: Obtain the ETL ingestion template (from GitHub or a shared workspace repo) that contains the Delta Live Tables configuration and workflow setup scripts.

### 6. Configure Output Paths for Configuration Files

* **Define a Directory for Configuration Files**: You’ll create configuration files for each dataset as you set up the ingestion framework. Choose a dedicated output path for these files in Volumes (e.g. /Volumes/...) as they’ll be referenced in the configuration generation tool.

# Standard Process Flow

Note: Some steps can be done ahead of others (i.e. Step 6 can be started alongside step 3)

Step 1: Create a new branch in GitHub for your work. It’s important to keep the changes as small as possible to enable quick review. Each change set should contain changes according to a User Story from Azure DevOps.

Step 2: Identify the source catalogs needed

Step 3: Start on Data Source Analysis for the catalogs

Step 4: Use the generator tool to jumpstart the table configuration files creation

Step 5: As identified from the Data Source Analysis, manually configure each table’s configuration (setting primary key, adjusting target locations if moving is needed, etc..)

Step 6: Place configuration files in the Mass Ingestion Area

Step 7: Create or modify the Databricks Asset Bundle for the group of tables (as identified in Data Source Analysis)

Step 8: Commit all asset bundle changes in GitHub, make any needed adjustments & create a Pull Request.  
Step 9: Inform the Data Engineering team of the new Pull Request & a review will be started

# Data Source Analysis

This table provides an overview of essential configuration parameters for managing a data pipeline. Each parameter plays a specific role in defining how data is extracted, loaded, transformed, and stored across different stages of the pipeline (such as raw, bronze, and silver layers). By configuring these parameters, you can control aspects like data extraction strategies, handling of historical and incremental data, partitioning for optimized storage, and transformation requirements.

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Description** | **Supported Framework Values** |
| Source Data extraction Strategy | Data read approach from source | Full, Incremental |
| Target Load Strategy Raw | Layer | cdc-append |
| Target Load Strategy Bronze | Bronze layer | SCD 2 |
| Watermark Column | Column to identify the incremental data | MODIFIED TIMESTAMP/ LOAD\_TIMESTAMP |
| Watermark Value | Historical data period for fetching source data | Minimum date for historical data load |
| Primary Key (Merge Key to identify new records) | Records | Eg. product\_code, id |
| Data Volume- Incremental | Incremental data volume expected |  |
| Data Volume-Historical | Historical data volume expected |  |
| CDC Patterns | Records | PK-NPK-HASHING/FULL-COLUMN-HASHING |
| Catalog Name | Catalog name for each target layer |  |
| Schema Name | Schema name for each target layer |  |
| Table Name | Table Name for each target layer |  |

# Data Ingestion Pipelines

This guide will walk you through setting up an asset bundle on Databricks. We will set up a workflow that automates the execution of a Delta Live Tables (DLT) pipeline, leveraging an existing template to streamline setup.

### **Step 1: Clone or Access the Template**

1. **Locate the Template**:
   1. It can be found within the workspace in a shared project repository.
2. **Clone the Template**:
   1. Clone or download the templates to your workspace .
   2. Ensure you have files:
      1. databricks.yml
      2. workflows/workflow.yml
      3. pipelines/pipeline.yml
      4. shemas/schema.yml
3. **Review the Template**:
   1. Review the DLT pipeline code to ensure it meets your data processing needs. Make adjustments if necessary.

### Step 2: Set Up the Delta Live Tables (DLT) Pipeline

Note that some values may already be set as a template default.

1. Configure the pipeline :
   1. Name the pipeline
   2. Pipeline mode: Triggered
   3. Cluster: TBD
   4. Target: ${var.schema}
   5. Notebooks: Leave as a reference to the main-repos notebook
2. Customize if needed
3. Save & validate

### Step 3: Configure Workflow

1. Configure the workflow from template:
   1. Name the workflow
   2. Add or update the DLT pipeline task to match the previous pipeline details
   3. Configure the job settings
      1. Schedule: TBD
      2. Cluster Configuration: TBD
      3. Retry: TBD
   4. Add additional tasks if needed
2. Save
3. Validate

# Source Table Configuration

Generate configuration files

Place them in the Mass Ingestion location (TBD)

Test

# Deployment

Changes made to Mass Ingestion repository should be in an individual branch

When ready, the changes can be manually deployed to Dev for testing and then a PR will be created for review

On merge from working branch to Main via the PR, the files and workflow configuration will be deployed to production

# TBD: Monitoring & Logging

# TBD: Troubleshooting

# Naming Standards

## Catalogs, Schemas, Tables

## Notebooks and Workflows