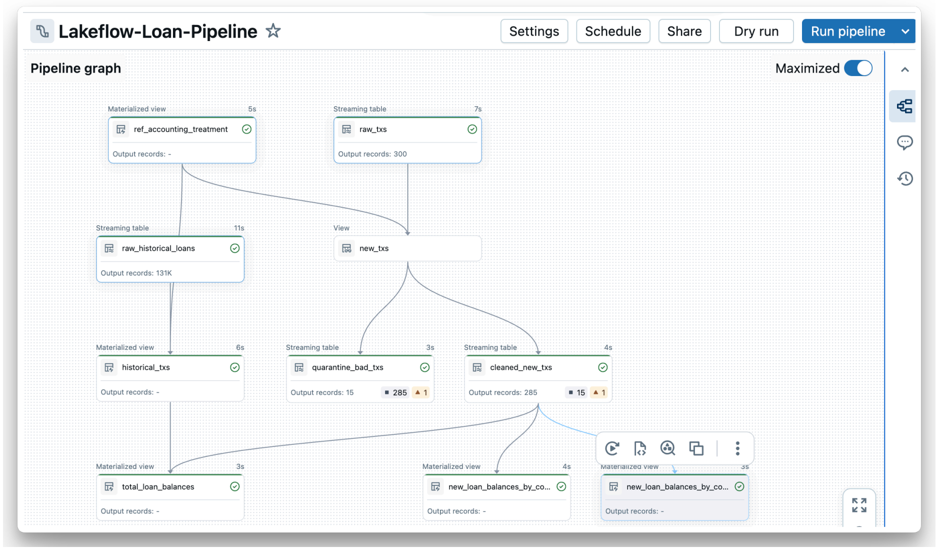
AI-Powered Data Engineering with Lakeflow

Lab Guide / Oct-29, 2025



0. Background

This lab guide is part of the Databricks Foundational Workshop "AI-Powered Data Engineering". This instructor led, beginner/intermediate level workshop is designed to give you hands-on experience with the latest Databricks data engineering capabilities. The lab guide provides high level steps to follow along with the tasks shown by the intructor. It's not a stand alone step-by-step beginners tutorial.

Environment

Databricks runs this workshop in Vocareum, but you can also run it in any standard workspace. This includes fully serverless workspaces like Databricks Express (but not the Databricks Free Edition).

Note that the online transcation broker needs to be started by the instructor first.

What you will learn

You will learn how to create a Lakeflow Spark Declarative Pipeline for streaming data, run it, and use it in a workflow with Lakeflow Jobs. You will also learn how to use the AI-powered tools like Databricks Assistant and Genie.

This is your task

You started a new job as a data engineer last week. Congratulations! Now you are asked to take over an ingestion pipeline written in SQL. The pipeline ingests and processes a constant stream of loan requests from an online transaction broker. Don't worry about the transaction broker, your instructor set this up for you (the code is provided in the repo for transparency).

Important

- This is your lab guide. Please **keep it open in a separate tab**. You will need it to follow the steps below throughout the course.
- This lab guide is also part of the public tmm GitHub repo, so you can easily recreate this course in your own account or even share it with your colleagues at your company.

Very Important

This workshop is designed in a way that it can be run with thousands of participants in a single Databricks account sharing a number of workspaces.

We are therefore using the **USER ID** (derived from your login user email) to separate schemas and pipelines and avoid namespace clashes. Just as in your own environment, you would use your company's naming schema for resources.

To get to your user id, check your login email by clicking on the "L" on the top right of the workspace. Example: labuser10148895_1745997814@vocareum.com means your user id is: `labuser10148895_1745997814` . If you are running the workshop in your own environment, you may follow your own namening conventions of course.

1. Add a GitHub Repo

To kick it off, let's create a git folder in your workspace and clone the existing repo your colleagues were working on.

Add a Git Folder

- On the left-hand side, click on **Workspace** and **Home** and then use the button at the top right and click **Create / Git Folder** to add a new git folder
  - For Git Repo URL use `https://github.com/databricks/tmm`
  - Git provider and repo name will be filled automatically (repo name is `tmm`).
  - Select **Sparse Checkout Mode** since we only need one folder (without sparse checkout, you will clone more content than necessary)
  - Under **Cone Pattern** put `Lakeflow-DataEng-Workshop`
  - Click **Create Repo** and the resources for this course will be cloned.
- In the file browser, click on **Lakeflow-DataEng-Workshop**. This is the folder we will be working with in this lab.

2. Spark Declarative Pipelines (SDP)

Understand SDP in SQL

- Watch your instructor explaining how to get started with pipelines.Also check out the core concepts in the documentation

After this module, you should be able to work with the pipelines editor and answer the following questions:

- What is the difference between Streaming Tables (ST) and Materialized Views (MV)?
- What is the CTAS pattern?

3. Explore the new Pipeline Editor

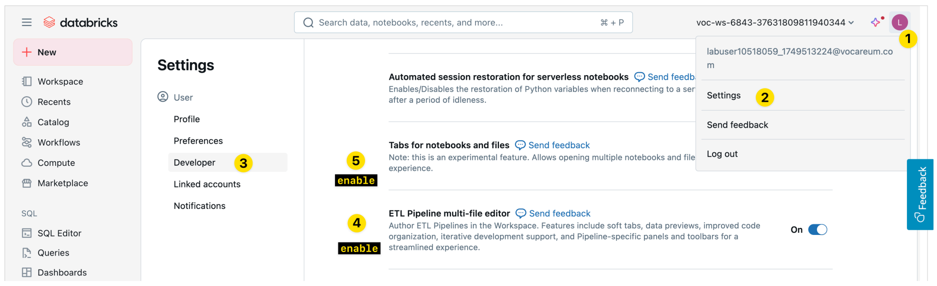
Enable Editor

Click on **Pipelines**, then click **Create** and select **ETL Pipeline**. There should be a popup asking you to enable the new editor. Enable it. If this succeeds, skip the two steps below.

Otherwise, follow those steps below (note, this is only necessary since the new editor is still in beta)

Under your username / Settings / Developer (see screenshot) enable

- Lakeflow Pipelines Editor
- Tabs for Notebooks and Files



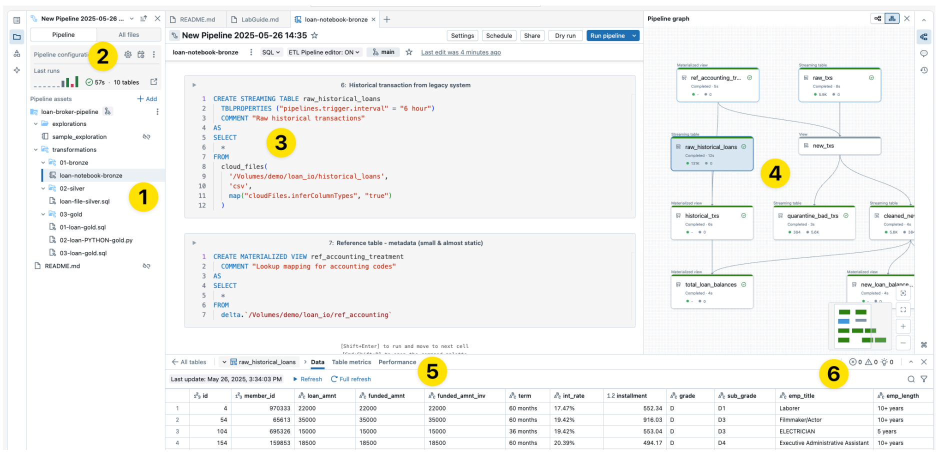
Automated session restoration for serverless notebooks Send feedb
Enables/Disables the restoration of Python variables when reconnecting to a ser
after a period of idleness.

Tabs for notebooks and files Send feedback
Note: this is an experimental feature. Allows opening multiple notebooks and file
experience.

ETL Pipeline multi-file editor Send feedback
Author ETL Pipelines in the Workspace. Features include soft tabs, data previews, improved code organization, iterative development support, and Pipeline-specific panels and toolbars for a streamlined experience. On

Explore

Note that the new editor is still in beta, so you the exact layout might be slightly different than on the screenshot below.



6: Historical transaction from legacy system

```sql
CREATE STREAMING TABLE raw_historical_loans
  TBLPROPERTIES ("pipelines.trigger.interval" = "6 hour")
  COMMENT "Raw historical transactions"
AS
SELECT
  *
FROM
  cloud_files(
    '/Volumes/demo/loan_io/historical_loans',
    'csv',
    map("cloudFiles.inferColumnTypes", "true")
  )
```

7: Reference table - metadata (small & almost static)

```sql
CREATE MATERIALIZED VIEW ref_accounting_treatment
  COMMENT "Lookup mapping for accounting codes"
AS
SELECT
  *
FROM
  delta.`/Volumes/demo/loan_io/ref_accounting`
```

[Shift+Enter] to run and move to next cell

Last update: May 26, 2025, 3:34:03 PM

	id	member_id	loan_amnt	funded_amnt	funded_amnt_inv	term	int_rate	installment	grade	sub_grade	emp_title	emp_length
1	4	970333	22000	22000	22000	60 months	17.47%	552.34	D	D1	Laborer	10+ years
2	54	65613	35000	35000	35000	60 months	19.42%	916.03	D	D3	Filmmaker/Actor	10+ years
3	104	695326	15000	15000	15000	36 months	19.42%	553.04	D	D3	ELECTRICIAN	5 years
4	154	159853	18500	18500	18500	60 months	20.39%	494.17	D	D4	Executive Administrative Assistant	10+ years

The new pipeline editor introduces several enhanced capabilities. Please familiarize yourself with these key features:

1. A new file hierarchy that lets you define path inclusion and organize files into custom directories (e.g., bronze, silver, and gold).
2. Dedicated location for pipeline configuration settings.
3. Flexible source code options for data pipelines:
   - You can use files or keep existing notebooks.
   - A file can contain a single or multiple data sets like streaming tables or materialized views
   - You can have any number of Python or SQL files.
   - You can create subfolder, e.g. to model the medallion architecture.
4. A pipeline graph with both vertical and horizontal layouts. Explore the available options by clicking on individual nodes.
5. Easy access to sample data, table metrics and performance monitoring.
6. Direct navigation to error locations in your source code.

Define your first pipeline

Pipelines are more than just code. Pipelines write data to schemas, they get triggered or run continuously, and they either run on serverless compute or not. All these settings can be defined when you create a new pipeline. Here we keep it short. Always leave the default values unless told otherwise.

1. In your workspace, select **Data Engineering / Job Runs (or Pipelines previously)**.
2. Top right, select **Create / ETL Pipeline**
3. Then define the name of the pipeline. Make sure to use your own `user_id` from above as the name of the pipeline
4. For **Unity Catalog**, change the settings for **catalog.schema** to
   - Catalog: `demo`
   - Target Schema: Your `user_id` . Note, you will **work with your own schema** to separate your data from others. Depending on the training environment, the schema with your user_id is already created for you. Make sure to select the correct schema name.

Add existing assets

Remember that you created a git folder which contains the code for the pipeline? In section we will add the pipeline code from that folder to the new pipeline definition!

1. Under **Get started with your pipeline**, click on **Add existing assets**
2. Select the following:

- **Pipeline root folder**: `tmm / Lakeflow-DataEng-Workshop / loans-pipeline`
- For **Source code paths** use the same path as for root folder, then select the `transformations` sub folder and click on **Add**.

Run your first pipeline

1. Drill down to the notebook that defines the bronze layer.

- Explore the the SQL and make sure you can identify streaming tables.
- Do the same for the silver and gold layer.
  - The silver layer is implemented in a single file
  - The gold layer is using a file per table approach.

2. On the top row, click on the "Start" triangle to run the pipeline. When you start the pipeline for the first time it might take a minute until resources are provisioned.

3. Explore the pipeline editor while waiting for the pipeline to start up and the pipeline graph being displayed

Pipeline Settings

Go to pipeline settings (it's the cogwheel symbol) and explore the following options

Main settings:

- Pipeline root folder and source code folder
- Default catalog and schema where tables are written to. This can be overwritten in the source code.
- Serverless compute

Also:

- Recap development vs production mode
- Understand how to use Unity Catalog
- Understand serverless compute

Pipeline Graph

- Make sure you understand the pipeline graph. Click on a particular node to see which options can be displayed.
- On the panel on the right hand side of the screen you can disable or enable the pipeline view entirely.
- You can always get to your pipeline by clicking on **Data Engineering** on the left menu bar and then on **Job Runs (previously Pipelines)**
- Explore the pipeline graph
  - Identify bronze, silver, and gold tables
  - Identify Streaming Tables and Materialized Views in the graph

Explore sample data, table matrics and performance

Select a table in the pipeline graph and explore sample data, table metrics and performance metrics.

Explore Streaming Pipelines

Streaming data pipelines incrementally ingest data from folder, cloud object stores or message brokers.

- Take a note of the ingested records in the bronze tables
- Run the pipeline again by clicking on "Start" (top right in the Pipeline view)
  - note, only the new data is ingested

- Select "Full Refresh all" from the "Start" button
  - note that all tables will be recomputed and backfilled
- Could we convert the Materialized View (MV) used for data ingestion to a Streaming Table (ST)
- Use the button "Select Table for Refresh" and select all silver tables to be refreshed only

UC and Lineage

Watch your instructor explaining UC lineage.

OPTIONAL: Delta Tables

(Instructor Demo)

Spark Declarative Pipelines (SDP) are an abstraction for Spark Structured Streaming built on Delta tables. Delta tables unify DWH, data engineering, streaming, and DS/ML.

- At the navigation bar on the left, select **Catalog** and navigate to your catalog and schema. Then check out the table details of one of your pipelineline tables:
  - Click on "Data"
  - Select your catalog / schema. The name of your schema is the `user_id` parameter of your pipeline setting.
  - Drill down to the `raw_tx` table
  - Check the table's schema and sample data

Publish to any catalog or schema

You can create pipeline tables under any catalog or schema name. Under "Pipeline Settings," only the default schema name is set.

Go and try using this feature and put the three gold tables into the `USER_ID_dashboard` schema.

Monitor Pipeline Events (Optional)

Watch your instructor explaining how to retrieve pipeline events, lineage, and runtime data from expectations.

3. Notebooks and Spark with Serverless Compute

You can now run Notebooks with Spark on serverless compute

- On the left menu bar, click on **+New Notebook**

- Edit the name of the notebook
- Make sure next to the notebook's name `Python` is displayed for the default cell type (or change it)
- Make sure on the right-hand side you see a green dot and `connected`. Click on that button to verify you are connected to `serverless compute` (if not, connect to serverless compute)

Use `/explain` and `/doc`

- Add the following command to a Python cell, then run it by clicking on the triangle (or using SHIFT-RETURN shortcut):

```
display(spark.range(10).toDF("serverless"))
```

- Click on the symbol for Databricks Assistant and document the cell. Hint: use /doc in the command line for Assistant and accept the suggestion.

Use `/fix`

- Add another Python cell and copy the following command into that cell. The command contains a syntax error. `display(spark.createDataFrame([("Python",), ("Spark",), ("Databricks",)], ["serverless"]))` Click on the Assistant toggle (the blueish/redish star) and try to fix the problem with `/fix` and run the command.

Note that the Assistant is context-aware and knows about table names and schemas from Unity Catalog.

3. DWH View / SQL Persona

The Lakehouse unifies classic data lakes and DWHs. This lab will teach you how to access Delta tables generated with a data pipeline from the DWH.

Use the SQL Editor

- On the left menu bar, select the SQL persona
- Also from the left bar, open the SQL editor
- Create a simple query:
  - `SELECT * FROM demo.USER_ID.ref_accounting_treatment` (make sure to use **your schema and table name**)
  - run the query by clicking Shift-RETURN
  - Save the query using your ID as a query name

4. Databricks Workflows with SDP

Create a Workflow

- In the menu bar on the left, select Workflows
- Select **Workflows owned by me**
- Click on **Create Job**
- Name the new job same as `user_id` from above

Add a first task

- Task name: `Ingest`
- Task type: `Pipeline task`
- Pipeline: `your pipeline name` for the Pipelines SQL notebook from above.
- Note,the pipeline should be in `triggered mode` for this lab.

Add a second task

- Task name: `Update Downstream`
- Task type: `Notebook`
- Select the `Update-Downstream` notebook
- Note that `Serverless` is automatically selected for compute on the right-hand side

Run the workflow

- Run the workflow from the **Run now** button top right
  - The Workflow will fail with an error in the second task. This is intentional. Do not panic.
  - Switch to the Matrix view.
    - To explore the Matrix View, run the workflow again (it will fail again).

Repair and Rerun (OPTIONAL)

- In the Matrix View, click on the second task marked in red to find out what the error is
  - Click on **Highlight Error**
- Debug the notebook (just comment out the line where the error is caused with `raise`)
- Select the Run with the Run ID again and view the Task
- Use the **Repair and Rerun** Feature to rerun the workflow
  - It should successfully run now.

- You can delete the other failed run.

6. Outlook (optional topics in preview)

Follow your instructor for capabilities such as Genie Data Rooms. Time permitting.

A full end-to-end demo of this section is available as a video in the Databricks Demo Center

Congratulations for completing this workshop!