

SWE Practices for DLT Pipelines




Agenda

SWE Practices for Delta Live Tables Pipelines

Lesson Name	Lesson Name
Lecture: SWE Practices with DLT	Follow Along Demo – ADE 5.4 – Unit Test Pipelines
Follow Along Demo – ADE 5.1 – Import Sample Data	Follow Along Demo – ADE 5.5 – Monitor Data Quality
Follow Along Demo – ADE 5.2 – Modularize Code	Lab: ADE 5.6L – Develop and Test Pipelines Lab
Follow Along Demo – ADE 5.3 – Develop Pipelines	





SWE Practices with DLT



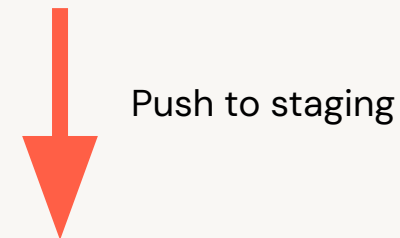
CI/CD Workflows

High-level steps

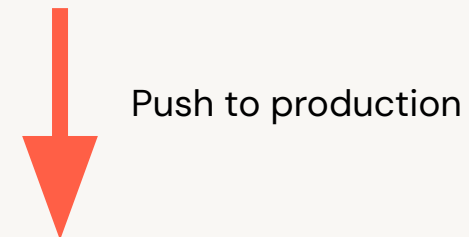
Develop code in a development environment



Unit test



Implementation test



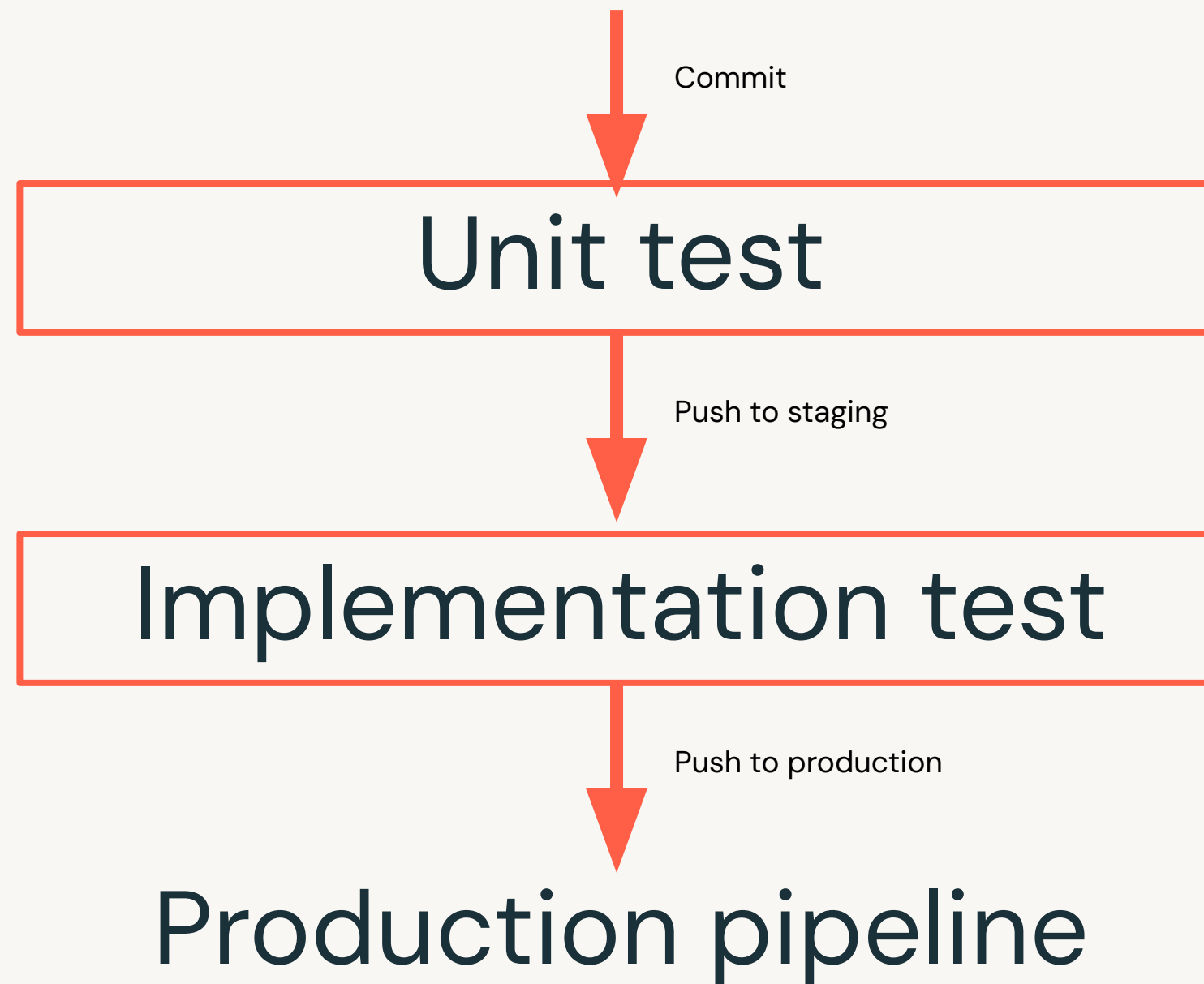
Production pipeline



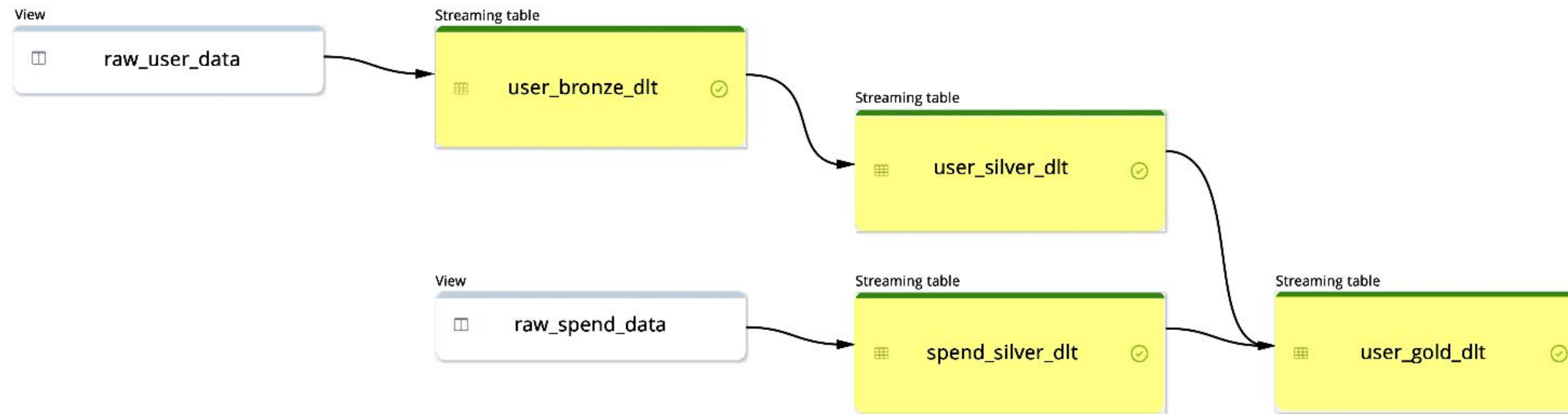
CI/CD Workflows

High-level steps

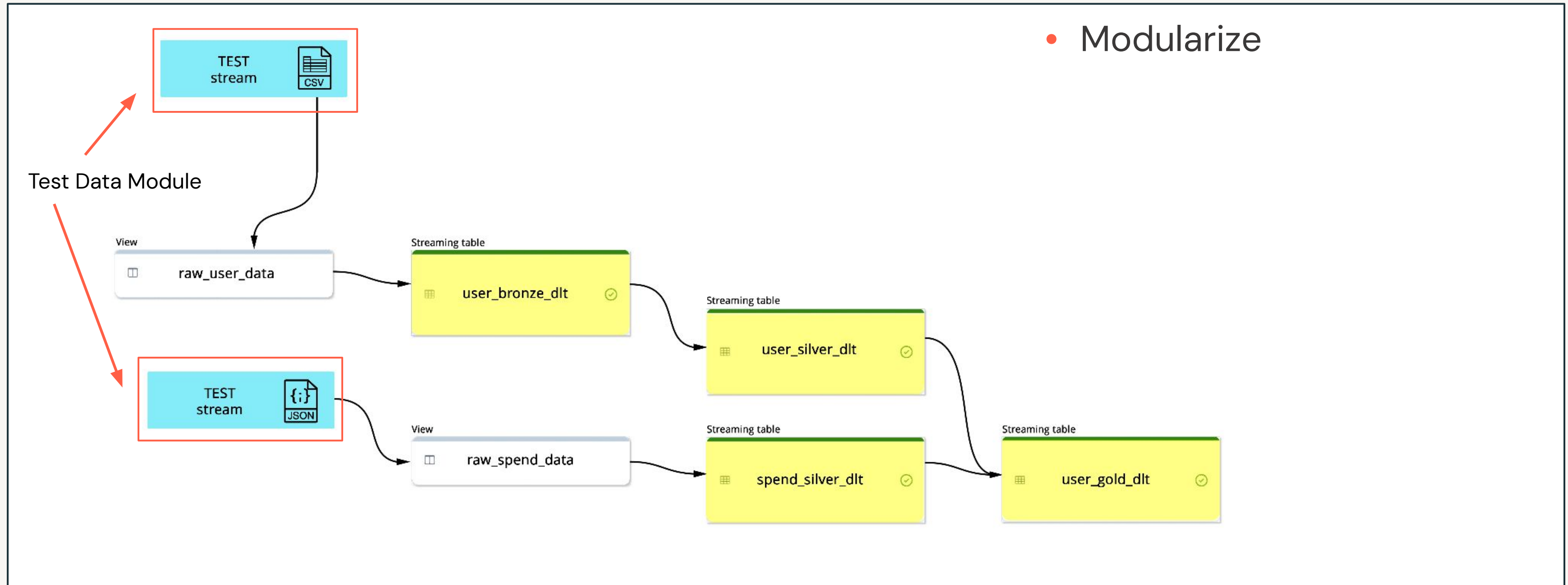
Develop code in a development environment



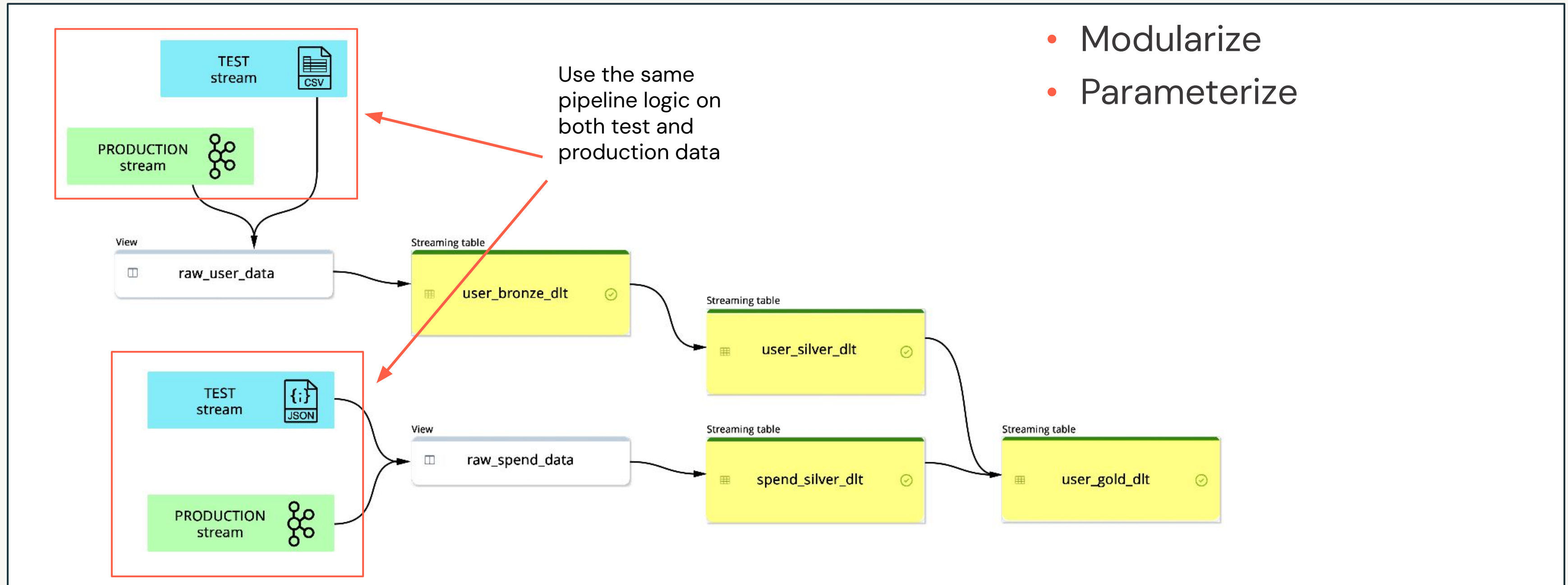
A Simple Pipeline



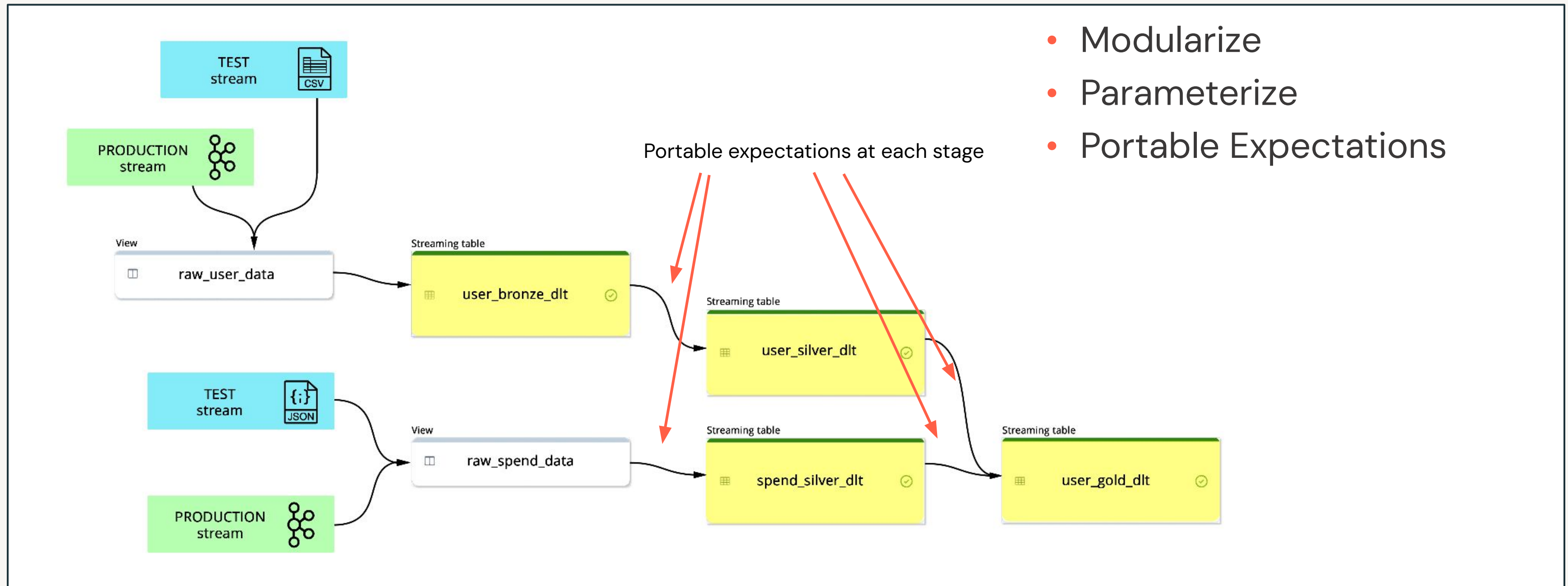
A Simple Pipeline



A Simple Pipeline



A Simple Pipeline



- Modularize
- Parameterize
- Portable Expectations

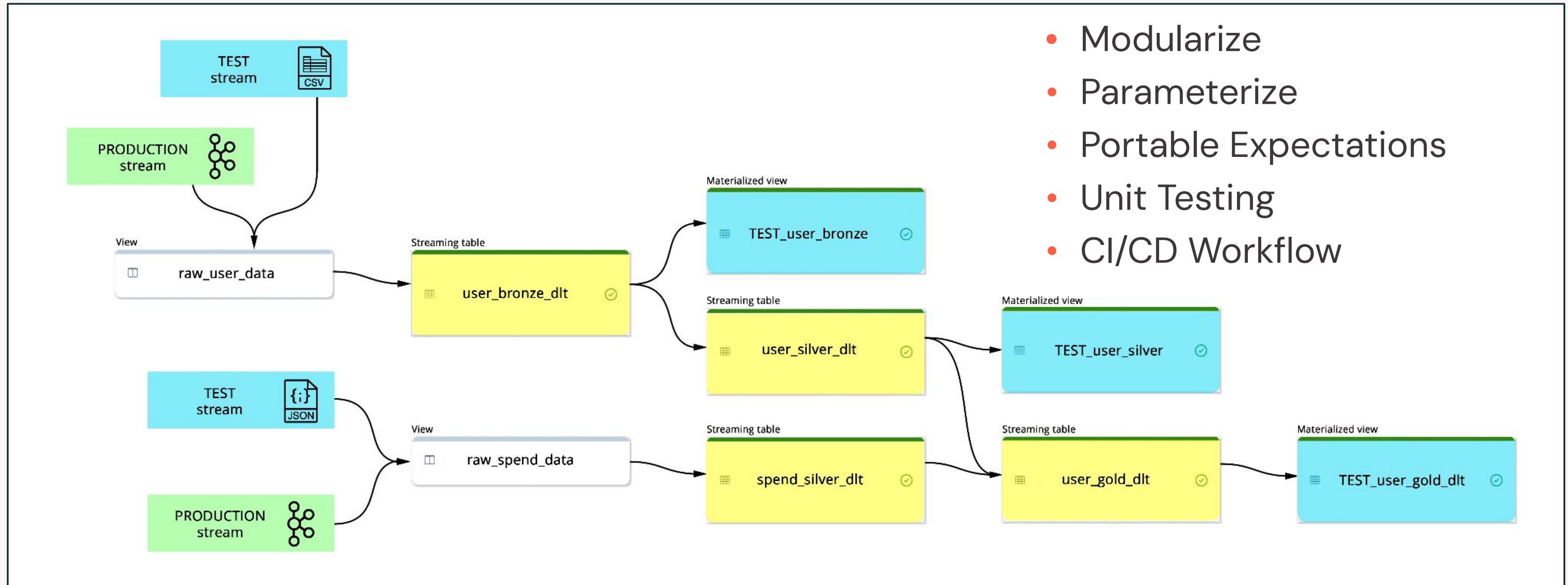


A Simple Pipeline

- Modularize
- Parameterize
- Portable Expectations
- Unit Testing



A Simple Pipeline



Demo:

Import Sample Data

Learning Objectives

Demo : Import Sample Data

- Understand how to use the %pip command to install Python libraries.
- Learn how to specify the library's version and installation URL
- Learn how to import specific modules from installed Python libraries.
- Understand the concept of modularizing code by packaging it into libraries.
- Gain familiarity with accessing sample datasets and descriptions within imported modules.



Demo

- Use ``%pip`` to install a Python library from a specified URL.
- After installation, import a specific module from the library into your notebook.
- Utilize the imported module to access sample datasets and descriptions.
- Convert sample data into a usable format, DataFrame or Spark DataFrame, and display it for analysis or processing.
- Imports the module from the library.
- Converts the attribute of the retrieved data into a DataFrame.
- Defines a schema for the JSON data, apply the schema to the JSON data and visualize the transformed Spark DataFrame



Lab:

Modularize Code

Learning Objectives

By the end of this lesson, you should be able to

- Refactor code into reusable functions and libraries in repository files.
- Utilize Python metaprogramming to automate the creation of tables for a DLT pipeline.
- Maintain data quality rules and transformations by storing them in portable metadata tables or configuration files.
- Generating and triggering pipeline.
- Organizing pipeline into separate ingest and transforming libraries for reuse.



Automated DLT Pipeline: Modular Code and Quality Management

- Configure the working environment using the provided setup cell.
- Organize the pipeline into separate ingest and transform libraries for code reuse.
- Configure libraries for different ingestion sources while sharing transformation logic.
- Generate pipeline and navigating to the pipeline user interface for monitoring and management.
- Trigger pipeline runs, both for landing new data and processing all remaining data in a continuous manner.



Demo:

Develop Pipelines

Learning Objectives

Demo : Develop Pipelines

- Configure isolated pipeline development environments by specifying target schema locations
- Segment code for efficient ingest and transform logic management across diverse deployment environments
- Utilize parameters for data source and volume control in development
- Prepare sample datasets for development and testing.
- Automate pipeline updates for Bronze/Development (Ingest Subset).



Efficient Data Pipeline Development and Automation

- Isolate environments (dev, test, prod) with distinct schemas and storage.
- Modularize pipelines using reusable ingestion and transformation libraries.
- Automate DLT pipelines with configurable components.
- Clean up unnecessary resources for a tidy development environment.
Centralize automation with "DA" for consistent deployments.
- Leverage the central "DA" (Data Automation) object for streamlined and consistent pipeline development across environments.



Demo:

Unit Test Pipelines

Learning Objectives

Demo : Unit Test Pipelines

- Understand how to install necessary dependencies for your DLT pipeline using %pip.
- Learn how to define unit tests for the ingestion and transformation steps within the DLT pipeline.
- Understand how to use temporary DLT tables with expectations to implement unit tests effectively.
- Learn how to generate and trigger an update of a DLT pipeline consisting of various notebooks.



Demo

- Creating tests that check the correctness and reliability of the data ingestion and transformation
- Create temporary data tables to validate the pipeline's expected results during testing.
- Automatically generate a DLT pipeline using predefined configuration settings
- Execute the data pipeline, including running unit tests for validation and issue detection.



Demo:

Monitor Data Quality

Learning Objectives

Demo : Monitor Data Quality

- Understand the importance of data quality monitoring in a DLT pipeline.
- Learn how to extract and analyze expectation metrics from DLT pipelines using Databricks SQL.
- Create SQL views and queries to track data quality metrics and visualize the results.
- Explore the structure of event logs and their significance in data quality tracking.



Demo

- Data accessing through table creation based on event logs in DLT pipeline storage.
- Using PySpark for analyzing expectation metrics stored separately.
- Event logs contain metadata on various event types.
- Creating SQL view for extracting data quality metrics from event logs and visualizing metrics (failed records, passed records, expectations).
- Using Plotly Express for visualization.
- Workflow provides a foundation for advanced data quality monitoring.



Knowledge Check



Knowledge check

Think about this question and volunteer an answer

Databricks CI/CD Workflows

Which of the following is an example of how to get a configuration variable in a DLT pipeline?

- A. `spark.get()`
- B. `spark.conf.get()`
- C. `@dlt.get()`
- D. `@dlt.spark.conf.get()`

Knowledge check

Think about this question and volunteer an answer

Databricks CI/CD Workflows

Which of the following can be used to ensure data quality?

- A. Parameterized DLT pipelines
- B. Making expectations portable
- C. Modularized code
- D. Unit tests

Knowledge check

A data engineer is using the following code to ensure their data meets a certain requirement.

```
CREATE TEMPORARY LIVE TABLE users (  
  CONSTRAINT incorrect_data_removed EXPECT  
(not_empty_rescued_data = 0) ON VIOLATION FAIL UPDATE  
)
```

Which of the following statements describes what happens when the requirement fails? Select one response.

- A. Rows with incorrect schema are dropped.
- B. Rows with null ids are dropped.
- C. Rows with duplicate primary keys are dropped.
- D. Rows with stale data are dropped.

Knowledge check

Think about this question and volunteer an answer

Databricks CI/CD Workflows

A data engineer needs to apply a common set of data quality rules to multiple tables. Which of the following best practices can they follow to do this? Select one response.

- A. Maintain data quality rules separately from the pipeline
- B. Create a separate pipeline containing the data quality rules and run it concurrently with the pipeline
- C. Tag the dataset used to populate the tables in the pipeline with data quality rule definitions
- D. Create a task in Workflows for data quality rules and make it a dependency of the pipeline

Knowledge check

Think about this question and volunteer an answer

Databricks CI/CD Workflows

A data engineer has created a pipeline that implements unit tests and has run this pipeline after pushing changes to the repo. Which of the following steps does the data engineer need to take in order to create a complete Workflow? Select one response.

- A. They need to anonymize any sensitive information in their tables.
- B. They need to run the implementation test when code is moved to a staging environment.
- C. They need to create data quality constraints to prevent bad records from entering the production table.
- D. They need to modularize their code to make it more efficient and readable.



