

Databricks: Lakeflow – Medallion pipelines, quality and orchestration

Databricks: Lakeflow is a hands-on course on building declarative data pipelines and orchestrating them. The program focuses on Lakeflow pipelines (Spark Declarative Pipelines, formerly Delta Live Tables) and Lakeflow Jobs, that is, on how to turn a working data process into a pipeline with dependencies, quality checks and monitoring.

Participants go through the whole process: from choosing the object type (streaming table or materialized view), through handling changes with Auto CDC and quality rules (expectations), to running the pipeline in a Lakeflow Job with parameters, retries and alerts. The course shows how to build pipelines that can be maintained reliably in a production environment. The pipeline is built for RetailOps (a fictional retail company whose data we use for all exercises).



Training recipients

The training is intended for technical staff who build and maintain production data flows in Databricks. The course is particularly suitable for:

- data engineers moving processes from notebooks to declarative pipelines,
- DataOps teams responsible for schedules, monitoring and incident handling,
- people who have used Delta Live Tables and want to learn the current capabilities of Lakeflow,
- data architects designing orchestration in a Lakehouse.

The training will be especially valuable for teams that already have working processes and want to put them in order in terms of data quality and observability.



Benefits

- The Lakeflow pipelines model – you will learn flows, streaming tables and materialized views and when to use each of them.
- Incremental processing – you will learn to choose between batch and streaming mode for individual pipeline stages.
- Handling changes with Auto CDC – you will see how to model changes and history (SCD type 1 and 2) in the Silver layer.
- Data quality in the pipeline – you will learn to define expectations and route invalid records to a separate table.
- Orchestration in Lakeflow Jobs – you will build a job with dependencies, parameters, retries and alerts.
- Observability – you will learn how to analyze pipeline execution through the event log and system tables.



Training program

Module – Lakeflow pipelines: how it works

Module goal: the participant understands how a declarative pipeline differs from a hand-written process.

- The declarative approach – we describe which tables should be created, and the platform works out the order, retries and incremental updates on its own
- Lakeflow pipelines vs. the earlier Delta Live Tables – what changed in names and syntax
- Object types: streaming table (data arriving incrementally) and materialized view (a result calculated from other tables)
- Flow – a stream of data that feeds a table; several sources can feed one table
- Defining a pipeline in SQL (CREATE OR REFRESH STREAMING TABLE) and in Python (from pyspark import pipelines as dp, @dp.table, @dp.materialized_view)
- The pipeline editor – files, the dependency graph between tables, and data preview
- Demo: the first Bronze → Silver pipeline (continued in workshop 1)

2. Module – Batch and streaming processing

Module goal: the participant knows how often and in what mode to refresh a pipeline to balance cost against data freshness.

- Triggered mode (for example, every hour) vs. continuous mode – data delay vs. cost; a continuous pipeline can be run from a continuous job, which Databricks has recommended since August 2026
- Incremental processing vs. full recalculation – when each one is the right choice
- Incremental refresh of materialized views – recalculating only the part of the data that changed
- Pipeline settings – target catalog and schema, compute, parameters (for example, the same code writing to a different catalog in dev and prod)

- Demo: comparing the delay and cost of both modes

3. Module – Auto CDC: handling changes from source systems

Module goal: the participant can bring changes from a source system into the Lakehouse, with or without history.

- Change stream (CDC) – the source sends information that a record was added, changed or deleted
- Auto CDC – applying changes to the target table declaratively, in the correct order of events (requires a serverless pipeline or the Pro/Advanced edition)
- SCD type 1 and 2 – overwriting the value or keeping the full history of changes (for example, a customer's change of address)
- Handling deletes and records that arrived out of order
- Lab (extra task for faster participants): the participant builds a customer dimension with change history (SCD type 2)

4. Module – Workshop 1: Silver → Gold pipeline

Module goal: the participant builds a complete declarative pipeline on their own.

- Workshop: a pipeline from prepared Bronze data to Gold tables with sales metrics. The outcome is a working pipeline with a visible dependency graph and lineage.
- The most common errors on the first run and how to diagnose them

5. Module – Data quality and observability

Module goal: the participant builds quality control into the pipeline and can check what happened inside it.

- Expectations – quality rules defined with the table; you choose whether to just warn, drop the record or stop the pipeline
- Quarantine – a separate table for rejected records so they can be analyzed – compared with the manual pattern from DBX-ETL
- Event log – the pipeline's log: how many records passed, how many were rejected, how long each step took; events from all pipelines in a region also go to the pipeline_events system table (Beta)
- System tables – run time and cost of pipelines in one place
- Lab: the participant adds quality rules and checks in the log which records were rejected and why

6. Module – Lakeflow Jobs: orchestration and monitoring

Module goal: the participant combines pipelines and other tasks into a scheduled, monitored process.

- A job as a set of tasks – notebooks, pipelines, SQL queries and Python scripts in one process
- Dependencies between tasks, conditions (run if...) and loops over a list of values
- Parameters – one job run for different dates, countries or environments
- Error handling – retries, timeouts, notifications by email and chat tools
- Triggers – a schedule, a new file arriving, an update to a source table; several triggers on one job (Beta)
- Rerunning only the failed tasks (repair run)
- Workshop 2: the participant builds a job that runs the pipeline from workshop 1 and a validation task, with parameters, retries and an alert. The outcome is a scheduled and monitored process.

7. Module – Preparing for deployment

Module goal: the participant knows what has to be prepared before a pipeline goes to production.

- Pipeline and job as code (Declarative Automation Bundles) – an introduction; full deployments are covered in DBX-OPS
- Development vs. production environment – differences in pipeline settings
- Criteria for handing over to deployment and a basic runbook for failures



Expected preparation of the participant

- Completion of DBX-ETL (End-to-End Data Engineering) or equivalent experience: Delta Lake, incremental data loading, MERGE
- Practical knowledge of SQL and basic knowledge of Python
- Basic knowledge of Git

To increase the comfort of work and training's effectiveness we suggest using an additional monitor. The lack of additional monitor does not exclude participation in the training, however, it significantly influences the comfort of work during classes.

You can find information and requirements of participation in Distance Learning trainings at:

<https://www.altkomakademia.pl/en/how-it-works/about-courses/>



Training Includes

Training includes

- access to the Altkom Akademia student portal
- a prepared Databricks environment with Bronze data and a pipeline skeleton for the duration of the training
- notebooks and exercise materials in electronic form

Note: During selected training sessions, the Academy provides Participants with access to paid tools and services necessary to complete the training program (in particular Azure, Fabric, and other cloud resources). The Participant is required to use the provided resources solely within the scope of the training agenda and in accordance with the trainer's instructions. Use of these resources beyond the training scope—particularly for private projects, testing additional services, modifying the environment configuration, or undertaking activities that generate costs unrelated to the exercises—will result in the Participant being charged for costs exceeding the allocated limit, based on expenses incurred by the Academy.



Duration

1 days / 7 hours

Language

- Training: English
- Materials: English