

Databricks: Lakehouse and Delta Tables – ingestion, schema evolution and performance

Databricks: Lakehouse and Delta Tables is a hands-on course for people building the data layer in a Medallion architecture. The program focuses on two decisions that determine the quality of the whole Lakehouse: how to choose the ingestion method for a given source, and how to design Delta tables so that increments, schema changes and updates are safe.

Participants go through the whole process from source to the Silver layer: they choose an ingestion mechanism, build incremental loading into Bronze with Auto Loader, handle schema evolution and run a safe upsert into Silver with the MERGE command. They also learn the Delta Lake features for change history, Change Data Feed and table performance. The exercises use the RetailOps scenario (a fictional retail company whose data we use for all exercises).



Training recipients

The training is intended for technical staff who design or build data flows in Databricks and want to follow Lakehouse good practices.

The course is particularly suitable for:

- data engineers building the Bronze and Silver layers,
- ETL developers moving from classic data warehouses to a Lakehouse,
- data architects choosing ingestion patterns for different sources,
- graduates of the DBX-FND course who want to move from a first flow to a solution that handles incremental data.

The training will be especially valuable for teams that already have their first tables in Databricks and want to put order into how those tables are loaded and updated.



Benefits

- Medallion architecture in practice – you will learn where the boundaries between the Bronze, Silver and Gold layers lie and which decisions are made in each of them.
- Choosing an ingestion mechanism – you will learn to match COPY INTO, Auto Loader, streaming tables, Lakeflow Connect or JDBC to the type of source.
- Incremental data loading – you will build loading with Auto Loader, with schema control and handling of unexpected columns.
- Safe updates in Delta Lake – you will see how to update existing records and add new ones (upsert) with the MERGE command and read changes through Change Data Feed.
- Table performance and cost – you will learn when to use liquid clustering, OPTIMIZE and VACUUM and how to justify an optimization decision.
- Preparing for production – you will see how lineage and table history help maintain a data flow.



Training program

1. Module – Medallion architecture and design decisions

Module goal: the participant knows how to split data into layers and what to decide before writing the first process.

- Bronze, Silver and Gold layers – raw data, cleaned data and business-ready data; what goes into which layer
- Business key and grain – how we tell that two records describe the same object (for example, an order)
- Idempotency – a process run twice must not duplicate data
- Load metadata – recording which file each record came from and when, so the history can be reconstructed
- Layer layout in Unity Catalog – separate catalogs or schemas for layers and domains
- Lab: the participant designs layers, keys and metadata for the RetailOps data

2. Module – Choosing an ingestion method

Module goal: the participant can match the data loading method to the type of source.

- COPY INTO – simple loading of files from a folder; files that were already loaded are skipped automatically; for SQL users, Databricks now recommends streaming tables as the more scalable and robust option
- Auto Loader – continuous or scheduled detection of new files, also when there are many of them

- Streaming tables in SQL – incremental loading without writing Python code (Databricks automatically creates a serverless pipeline for them)
- Lakeflow Connect – ready-made connectors to SaaS applications, databases (including change data capture – CDC) and files in SharePoint or Google Drive; network requirements and permissions
- Direct database connection (JDBC) – when it makes sense and when it puts too much load on the source system
- Lab: the participant picks a loading method for five sample sources and explains the choice

3. Module – Auto Loader: incremental loads and changes in file structure

Module goal: the participant builds a load that processes only new files and does not break when a new column appears in the data.

- How Auto Loader remembers which files it has already processed (checkpoint) and why each file lands in the table exactly once
- Configuration: file format, where the schema is stored, column type hints; how new files are detected – listing the folder or cloud notifications (Databricks recommends notifications based on file events)
- Changes in file structure – a new column can be added to the schema (the process stops and, after a restart, runs with the new column – `addNewColumns`), set aside in `_rescued_data` (rescue), or stop the process until the team decides (`failOnNewColumns`)
- The `_rescued_data` column – a place for values that do not fit the schema, so they are not lost
- Scheduled runs – processing all new files and then stopping (`availableNow`)
- Lab: the participant loads three successive batches of order files, including a batch with a new column

4. Module – Delta Lake: how MERGE, history and Change Data Feed work

Module goal: the participant understands how Delta Lake performs updates and records what changed in a table.

- Table history – every change is a version; see who changed what and when
- Changing the table structure – when Delta blocks a write and when it allows new columns to be added
- MERGE INTO – how the command works: matching records by key, then adding new ones, updating existing ones and deleting outdated ones
- Duplicates in the input data – why MERGE fails when several records match one row of the table
- Going back to an earlier version of the table (time travel, RESTORE) – a recap from DBX-FND
- Change Data Feed – reading added, changed and deleted rows for a chosen range of versions; in the new automatic version, changes are computed at read time and the feature does not have to be enabled on each table separately (requires row tracking and Databricks Runtime 19+); the exercises use the classic version, enabled per table, which runs on LTS runtimes
- Lab: the participant updates the Silver table with MERGE, reads the list of changes and reverts a faulty update

5. Module – Performance and cost of Delta tables

Module goal: the participant knows what slows down table queries and how to fix it.

- The problem of many small files and how Delta skips files that do not contain the data being searched for

- Liquid clustering – organizing data by the most frequently filtered columns or automatic key selection (CLUSTER BY AUTO); the successor to partitioning and ZORDER
- Table maintenance: merging small files (OPTIMIZE) and removing old versions (VACUUM)
- Predictive optimization – automatic table maintenance by the platform (OPTIMIZE, VACUUM, statistics), being enabled by default for Unity Catalog managed tables
- Query profile – how to check how much data was actually read
- Lab: the participant compares, in the query profile, the number of files read before and after liquid clustering (OPTIMIZE)

6. Module – End-to-end workshop and production readiness

Module goal: the participant combines the pieces of the day into a working flow and knows what to check before handing it over to production.

- Workshop: Auto Loader → Bronze table → duplicate-free update of the Silver table for the RetailOps order data. The outcome is an incremental flow that can be run safely many times. Extra task for faster participants: reading changes from the Silver table through Change Data Feed and enabling liquid clustering on it.
- Lineage – checking where the data in a table comes from and who uses it
- Monitoring – table history and load metrics
- Checklist before handing the flow over to orchestration



Expected preparation of the participant

- Completion of DBX-FND (Databricks Fundamentals) or equivalent knowledge: Unity Catalog, Volumes, Delta tables
- Practical knowledge of SQL
- Basic knowledge of Python/PySpark is recommended – the Auto Loader exercises use ready-made code templates

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

- access to Altkom Akademia student

Training method:

- a prepared Databricks environment with Unity Catalog, a Volume and RetailOps scenario files 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