LAB 05: Streaming & Auto Loader

Duration: ~35 min
Day: 2
After module: M05: Incremental Data Processing
Difficulty: Intermediate-Advanced

Scenario

> *"RetailHub launched a new sales channel – orders now arrive as JSON files every 5 minutes in a landing zone. Your task: configure Auto Loader for continuous ingestion, process the stream, and land the data in the Bronze Delta table with exactly-once guarantees."*

Objectives

After completing this lab you will be able to: - Configure Auto Loader (`cloudFiles`) for streaming file ingestion - Set up `readStream` and `writeStream` pipelines - Use `trigger(availableNow=True)` for incremental batch processing - Manage checkpoints for exactly-once guarantees - Monitor active streams

---

Part 1: COPY INTO (Batch) (~10 min)

Task 1: Load Files with COPY INTO

Use `COPY INTO` to load the first batch of JSON files. Observe its idempotent behavior.

> **Exam Tip:** `COPY INTO` tracks files already loaded. Re-running it on the same files is a no-op. Useful for simple, scheduled batch ingestion.

<screen = COPY INTO command result showing rows loaded and files processed count>

Task 2: Re-run COPY INTO

Run the same `COPY INTO` again. Observe that 0 new rows are loaded (idempotent).

Part 2: Auto Loader (~15 min)

Task 3: Configure Auto Loader Stream

Set up a `readStream` with Auto Loader: - Format: `cloudFiles` - Source format: JSON - Schema location for schema inference - Checkpoint location for exactly-once processing

Task 4: Write Stream to Bronze

Write the Auto Loader stream to a Bronze Delta table using `trigger(availableNow=True)`.

<screen = Streaming query progress showing numInputRows, numOutputRows, and processing time>

> **Exam Tip:** `trigger(availableNow=True)` processes all available files and then stops. It replaces the deprecated `trigger(once=True)` with better performance (multiple micro-batches).

Task 5: Simulate New Arrivals

Copy new JSON files to the landing zone, then re-run the stream. Only new files should be processed.

Part 3: Stream Monitoring (~10 min)

Task 6: Check Active Streams

Use `spark.streams.active` to list running streams.

Task 7: Inspect Checkpoint

Examine the checkpoint directory to understand the state management.

> **Exam Tip:** Checkpoints store: (1) what files have been processed (source offsets), (2) what data has been committed (sink commitlog). Never delete or modify checkpoints in production.

Summary

In this lab you: - Used COPY INTO for idempotent batch ingestion - Configured Auto Loader for streaming file ingestion - Used trigger(availableNow=True) for incremental processing - Verified exactly-once guarantees via checkpoints

> **What's next:** Day 3 starts with LAB 06 - Advanced Transforms using PySpark and SQL.