Lab 16: Auto Loader with Schema Evolution (Add/Update Columns Automatically)

Author: Dr. Sandeep Kumar Sharma

🎯Lab Overview

In this lab, you will learn how Databricks Auto Loader automatically **detects new columns**, **updates schemas**, and **handles changing data structures** over time. This is one of the most powerful and production-critical features of Auto Loader.

Data coming into cloud storage (ADLS/S3/GCS) often changes: - New columns get added - Vendors change file structure - Extra attributes appear suddenly - Optional fields appear/disappear

Instead of the pipeline breaking, Auto Loader can **adapt automatically**.

Learning Objective

By the end of this lab, you will: - Understand what schema evolution means - Enable Auto Loader's automatic schema evolution - Use schema location to track schema history - Ingest CSV files even when new columns appear over time - Write evolved schemas into Bronze Delta tables

---

🛼Real-World Scenario (Simple Explanation)

Imagine your incoming ADLS folder receives daily CSV files like:

```
sales_20240101.csv
sales_20240102.csv
```

Initially, files contain:

```
id,amount,country
```

But suddenly after one week your vendor adds a new column:

```
id,amount,country,currency
```

A normal pipeline will crash.
But Auto Loader will learn this new schema and update automatically.

This is called **Schema Evolution**.

--

STEP 1 — Define Paths for Input, Checkpoint & Schema Tracking

--

```
input_path = "abfss://raw@yourstorageaccount.dfs.core.windows.net/sales-evolving/"
checkpoint_path = "dbfs:/mnt/checkpoints/autoloader_sales_evolving/"
schema_path = "dbfs:/mnt/schema/autoloader_sales_schema/"
```

Schema Location is very important because Auto Loader stores: - Latest schema JSON - Historical schema versions - Column tracking metadata

---

--

STEP 2 — Create Auto Loader Stream with Schema Evolution Enabled

--

Use `cloudFiles.schemaEvolutionMode` to allow changes.

```
autoloader_df = (spark.readStream
                     .format("cloudFiles")
                     .option("cloudFiles.format", "csv")
```

```
                       .option("header", "true")
                       .option("cloudFiles.inferColumnTypes", "true")
                       .option("cloudFiles.schemaLocation", schema_path)
                       .option("cloudFiles.schemaEvolutionMode", "addNewColumns")
                       .load(input_path))
```

Explanation (Classroom Style):

- **inferColumnTypes** → allows Auto Loader to guess data types
- **schemaLocation** → Auto Loader stores schema history here
- **schemaEvolutionMode = addNewColumns** → new columns auto-appear!

If a new column appears tomorrow, Auto Loader will: 1. Detect it 2. Merge into existing schema 3. Add as a new column in DataFrame 4. Write to Delta table WITHOUT breaking the pipeline

---

--

STEP 3 — Write the Stream to a Bronze Delta Table

--

```
output_bronze = "dbfs:/mnt/bronze/sales_evolving_bronze/"

query = (autoloader_df.writeStream
             .format("delta")
             .option("checkpointLocation", checkpoint_path)
             .outputMode("append")
             .start(output_bronze))
```

This starts the incremental ingestion.

--

STEP 4 — Test Schema Evolution

--

Step 4.1 — Upload file with original schema

```
id,amount,country
1,100,USA
2,200,India
```

Auto Loader ingests it.

Step 4.2 — Upload a file with new column added

```
id,amount,country,currency
3,150,UK,GBP
4,250,France,EUR
```

Auto Loader will: - Detect `currency` column - Update schema - Add the column to Delta table - Fill missing values for old rows with `NULL`

Verify

```
display(spark.read.format("delta").load(output_bronze))
```

You will see:

```
id | amount | country | currency
1  | 100    | USA     | NULL
2  | 200    | India   | NULL
3  | 150    | UK      | GBP
4  | 250    | France  | EUR
```

This proves schema evolution worked.

STEP 5 — Stop the Stream

```
query.stop()
```

How Does Auto Loader Handle Schema Evolution Internally?

Team, this is important.
The magic happens inside the **schemaLocation folder**.

Auto Loader stores: - `schema.json` → current schema - `deltaSchemaHistory` → history of schema versions - `column_mapping` → tracks old & new columns

Auto Loader compares each incoming file's schema with the stored schema and: - Adds new columns automatically - Fills missing values with NULL - Updates Delta table metadata

You don't need to write complex merge logic.

Production Best Practices

- Always use a **schemaLocation** folder
- Use **addNewColumns** mode for semi-structured data
- For strict pipelines (like finance), use explicit schema
- Combine Auto Loader + Delta for full schema evolution support
- Avoid deleting the schemaLocation unless starting from scratch

---

End of Lab 16

You have now learned how to: - Enable schema evolution in Auto Loader - Automatically add new columns - Track schema history - Build a resilient ingestion pipeline for evolving data

If you want, next we can create: - **Lab 17 – Auto Loader for Complex JSON** - **Lab 18 – Bronze → Silver → Gold with Auto Loader**