LAB 02: ELT Ingestion & Transformations

Duration: ~40 min
Day: 1
After module: M02: ELT Data Ingestion
Difficulty: Beginner-Intermediate

Scenario

> *"The RetailHub data team received the first data export from the store system – CSV files with customer and product data, plus a JSON file with order history. Your task: load these files into the Lakehouse, apply basic transformations, and save them as Delta tables in the Bronze layer."*

Objectives

After completing this lab you will be able to: - Read CSV, JSON, and Parquet files using `spark.read` - Compare `inferSchema` vs explicit schema definition - Apply DataFrame transformations: select, withColumn, filter, cast - Create temporary views and run SQL queries - Write DataFrames as managed Delta tables using `saveAsTable()`

Prerequisites

- Completed LAB 01 (or just run `%run ../setup/00_setup`)
- Dataset files available in Volume or `DATASET_PATH`

Part 1: Data Ingestion (~15 min)

Task 1: Read Customers CSV

Read the customers CSV file with explicit schema (don't use `inferSchema`).

Requirements: - Define a `StructType` schema with appropriate types - Use `.option("header", True)` - Verify column types with `printSchema()`

> **Exam Tip:** `inferSchema` triggers an extra pass over the data. In production, always define schema explicitly for CSV files.

<screen = Notebook cell showing printSchema() output with correctly typed columns (string, integer, etc.)>

Task 2: Read Orders JSON

Read the orders batch JSON file. JSON infers schema automatically, but verify nested structures.

Requirements: - Use `spark.read.format("json")` - Check if any columns contain nested types (struct, array)

Task 3: Read Products CSV

Read the products file with `inferSchema` and then compare the schema with a manually defined one.

Part 2: Transformations (~15 min)

Task 4: Clean Customer Data

Apply the following transformations to `df_customers`: 1. `select` only: customer_id, first_name, last_name, email, city, country 2. `withColumn` – create `full_name` by concatenating first_name + " " + last_name 3. `filter` – keep only customers from a specific country 4. `cast` – ensure customer_id is IntegerType

Task 5: Create Temp View + SQL Query

1. Register `df_customers` as a temporary view `v_customers`
2. Write a SQL query to count customers per country
3. Display the result sorted by count descending

> **Exam Tip:** `createOrReplaceTempView()` is session-scoped. `createOrReplaceGlobalTempView()` is accessible from any notebook on the same cluster via `global_temp.view_name`.

Part 3: Save as Delta Tables (~10 min)

Task 6: Write to Bronze Layer

Save all three DataFrames as Delta tables in the Bronze schema: - `bronze.customers` - `bronze.orders` - `bronze.products`

Requirements: - Use `.write.mode("overwrite").saveAsTable()` - Use fully qualified names: `{CATALOG}.{BRONZE_SCHEMA}.table_name`

Task 7: Verify Tables

Query the tables using SQL to verify they were saved correctly.

<screen = SQL query result showing SELECT count(*) FROM bronze.customers with row count>

Summary

In this lab you: - Loaded CSV and JSON files into Spark DataFrames - Applied transformations: select, withColumn, filter, cast - Created temporary views and ran SQL queries - Saved cleaned data as Delta tables in the Bronze layer

> **What's next:** In LAB 03 you will use MERGE to handle incremental data updates and learn Time Travel for disaster recovery.