Pipeline Models and Function Reference

This document describes core data processing models and function configurations used across the bronze and silver layers. Each model defines how data is read, transformed, and written, with emphasis on history retention, performance, and idempotence.

The silver layer can follow multiple models depending on data needs.

1. **Append-only model:**
   * Always used in bronze.
   * Captures full file and record history.
   * Simple, low-cost, and resilient.
   * Enables complete lineage without tracking file metadata separately.
2. **SCD2 model**:
   * Heavier on compute/storage.
   * Maintains full history.
   * Common in regulated industries, finance, HR, audit needs.
   * Idempotent: Yes but fragile (needs to run every day).
3. **Overwrite/snapshot model** (stateless):
   * Silver always reflects the latest state.
   * No historical tracking.
   * Simple, fast, cost-efficient.
   * Cheaper than upsert for small tables.
   * Idempotent: Yes.
4. **Upsert (MERGE) model**:
   * Latest changes merged into silver.
   * No historical tracking.
   * Suitable in CDC pipelines without SCD2 needs.
   * Cheaper than snapshot for big tables with small updates.
   * Idempotent: Yes with proper business key.

Append-Only Streaming

This uses Auto Loader (cloudFiles) settings. Always use this for bronze.

* Trigger type is hard-coded to available now.
* To support derived ingest time, structure file paths like /Volumes/path/yyyyMMdd/file.txt.
* Bronze: Create file\_history, transaction\_history to preserve lineage and traceability.

JSON Settings

    "read\_function": "functions.stream\_read\_cloudfiles",

    "transform\_function": "functions.bronze\_standard\_transform",

    "write\_function": "functions.stream\_write\_table",

    "build\_history": "true",

SCD2 Streaming

Minimizes records per job; also gives SCD2 information.

* Silver: Skip history building because SCD2 merges lose 1:1 file-to-record lineage. SCD2 focuses on **record change tracking**, not file lineage.

JSON Settings

    "read\_function": "functions.stream\_read\_table",

    "transform\_function": "functions.silver\_scd2\_transform",

    "write\_function": "functions.stream\_write\_scd2\_table",

    "upsert\_function": "functions.scd2\_upsert",

    "build\_history": "false",

    "ingest\_time\_column": "derived\_ingest\_time",

SCD2 Batch

Deduplicate and then upsert/merge into an SCD2 table.

* Silver: Skip history building because SCD2 merges lose 1:1 file-to-record lineage. SCD2 focuses on **record change tracking**, not file lineage.

Silver

    "read\_function": "functions.read\_table",

    "transform\_function": "functions.silver\_scd2\_transform",

    "write\_function": "functions.write\_scd2\_table",

    "upsert\_function": "functions.scd2\_upsert",

    "build\_history": "false",

    "ingest\_time\_column": "derived\_ingest\_time",

Overwrite/Snapshot Batch

Used for simple, fast, and cheap snapshots in silver (best for small tables). **Can** be used for rebuilding broken tables ([example](https://github.com/bryanlharris/edsm/blob/main/sandbox/Python_systemsPopulated.ipynb)).

When I use this for a job I will come back and paste the settings here.

Upsert (Merge) Streaming

This is a typical structured streaming merge that other companies use.

When I use this for a job I will come back and paste the settings here.

Upsert (Merge) Batch

This is a typical batch merge that other companies use.

When I use this for a job I will come back and paste the settings here.

Function Reference

|  |  |
| --- | --- |
| **Function Name** | **Description** |
| stream\_read\_cloudfiles | Reads streaming data using Auto Loader (cloudFiles). |
| stream\_read\_table | Reads a Delta table as a streaming source. |
| read\_table | Reads a Delta table in batch mode. |
| read\_windowed\_snapshot | Reads latest row per key using window and row\_number. |
| read\_latest\_ingest | Reads only rows with the latest ingest\_time value. |
| bronze\_standard\_transform | Applies common bronze transforms. |
| silver\_standard\_transform | Renames, casts types, and adds hash/file metadata. |
| silver\_scd2\_transform | Applies standard silver transforms and adds SCD2 columns. |
| silver\_scd2\_catchup\_transform | Filters input to only include rows newer than the target table. |
| add\_scd2\_columns | Adds SCD2 columns. |
| add\_source\_metadata | Adds source\_metadata struct column. |
| add\_row\_hash | Adds a SHA-256 hash column if enabled. |
| clean\_column\_names | Normalizes column names. |
| rename\_columns | Renames columns based on a mapping. |
| cast\_data\_types | Casts columns based on a mapping. |
| overwrite\_table | Overwrites destination table in batch mode. |
| stream\_write\_table | Writes DataFrame to table using streaming sink. |
| stream\_upsert\_table | Uses foreachBatch to perform streaming upsert. |
| upsert\_microbatch | Returns a function that performs standard upsert merge logic. |
| stream\_write\_scd2\_table | Streaming SCD2 table write using upsert with deduplication. |
| scd2\_upsert | Upsert function for batch SCD2 tables. |
| stream\_scd2\_catchup\_write | Streaming catch-up SCD2 write, to apply late arriving data. |
| scd2\_catchup\_outer\_upsert | Loops through distinct ingest times, calling scd2\_upsert. |
| scd2\_write | Batch SCD2 write using deduplicated input. |