SCD Type 2 Logic Explanation (Line by Line)

This document explains the logic of a Slowly Changing Dimension Type 2 (SCD2) implementation using SQL MERGE and INSERT statements. It is broken into 4 clear steps: creating the table, initial insert, updating old records, and inserting new ones.

# ✅ Step 1: Create the SCD2 Table

CREATE TABLE scd\_cat.scd\_schema.scd2\_table (  
 id INT,  
 name STRING,  
 value STRING,  
 valid\_from TIMESTAMP,  
 valid\_to TIMESTAMP,  
 is\_current BOOLEAN  
);

We create a versioned history table that will store all changes over time. Each row includes when it became valid (`valid\_from`), when it stopped being valid (`valid\_to`), and a flag to indicate if it is the current record.

# ✅ Step 2: Initial Insert from Source Table

INSERT INTO scd\_cat.scd\_schema.scd2\_table  
SELECT  
 id,  
 name,  
 value,  
 updated\_at AS valid\_from,  
 TIMESTAMP '9999-12-31 23:59:59' AS valid\_to,  
 TRUE AS is\_current  
FROM scd\_cat.scd\_schema.source\_table;

We load the first set of data into the SCD2 table from the source, marking all as current with a far-future `valid\_to`.

# 🔁 Step 3: MERGE – Expire Old Records (if Changed)

MERGE INTO scd\_cat.scd\_schema.scd2\_table AS target  
USING scd\_cat.scd\_schema.source\_table AS source  
ON target.id = source.id AND target.is\_current = TRUE  
WHEN MATCHED AND (  
 target.name <> source.name OR target.value <> source.value  
)  
THEN UPDATE SET  
 target.valid\_to = source.updated\_at,  
 target.is\_current = FALSE;

We compare current records (`is\_current = TRUE`) with source data. If any values differ, we expire the existing record by updating its `valid\_to` and setting `is\_current = FALSE`.

# ➕ Step 4: INSERT – Add New Current Version of Records

INSERT INTO scd\_cat.scd\_schema.scd2\_table  
SELECT  
 s.id,  
 s.name,  
 s.value,  
 s.updated\_at AS valid\_from,  
 TIMESTAMP '9999-12-31 23:59:59' AS valid\_to,  
 TRUE AS is\_current  
FROM scd\_cat.scd\_schema.source\_table s  
LEFT JOIN scd\_cat.scd\_schema.scd2\_table t  
 ON s.id = t.id AND t.is\_current = FALSE AND t.valid\_to = s.updated\_at  
WHERE t.id IS NOT NULL;

We insert new versions of changed records using the source data. We only insert rows that were just expired in the previous MERGE step (matched via `valid\_to = updated\_at`).

# 🧠 Summary

This full SCD2 logic allows you to keep track of changes in data over time. It retains the full history, marks old records as expired, and maintains the current version. Useful for audit trails, historical analysis, and reporting.