NEXT DATA CRUNCH

SLOWLY CHANGING DIMENSIONS (SCD)

Managing Historical Data Changes in Data Warehouses

NUR AZMI PRASETYO

NEXT DATA CRUNCH

INTRODUCTION TO SCD

Slowly Changing Dimensions (**SCD**) refer to the management of data attributes that **change slowly over time** in **data warehouses.**

In simple terms, SCDs help us track how specific data points, like customer information, product details, or else **change gradually.**

NEXT DATA CRUNCH

WHY IS SCD IMPORTANT?

- SCD helps maintain the integrity of **historical data** even when changes occur.
- Data must reflect **the most accurate and up-to-date values**, while also allowing access to past records.
- **Tracking changes** to key attributes enables businesses to understand trends over time.

NEXT DATA CRUNCH

REAL-WORLD EXAMPLES

- **Product Price**: If a product’s price **increases over time**, you need to **keep the original price** in **historical records** while applying the **new price for future transactions** to maintain accurate financial reporting.

- **Customer Data**: Imagine you have a customer who **changes their address** or updates their phone number. You don’t want to **lose track** of their old address because it might affect past transactions, but you **also need to keep their new address** for future orders

3 TYPES OF SCD

- **Type 1 (Overwrite)**: The existing data is **overwritten** with the new value. **No history is kept**.
- **Type 2 (Track Historical Changes):** A **new record is created** with the updated data, and **the previous record is closed** (end_date). This keeps a **full** historical **history of changes.**
- **Type 3 (Limited History):** A new field is added to track the previous value. This method keeps **limited historical data** (typically for just **one change**).

EXAMPLE

Scenario: Product Table
Let’s see how the data might look **before** and **after** changes across different SCD types.

Initial table (before any changes):

product_id	product_name	quantity	start_date	end_date	is_current	previous_quantity
101	Widget A	50	2024-01-01	NULL	TRUE	NULL
102	Widget B	30	2024-01-01	NULL	TRUE	NULL

SCD TYPE 1 (OVERWRITE)

Scenario: Quantity of "Widget A" is updated from 50 to 60.

SCD Type 1: The existing row is **overwritten** with the **new quantity**.

product_id	product_name	quantity	start_date	end_date	is_current	previous_quantity
101	Widget A	60	2024-01-01	NULL	TRUE	NULL
102	Widget B	30	2024-01-01	NULL	TRUE	NULL

SCD TYPE 2 (TRACK HISTORICAL CHANGES)

Scenario: The quantity of "Widget A" is updated from 60 to 70.

SCD Type 2: A **new record** for "Widget A" is created with the **updated quantity**, and the old record is **marked** as **inactive**.

product_id	product_name	quantity	start_date	end_date	is_current	previous_quantity
101	Widget A	60	2024-01-01	2024-02-01	FALSE	NULL
101	Widget A	70	2024-02-01	NULL	TRUE	60
102	Widget B	30	2024-01-01	NULL	TRUE	NULL

SCD TYPE TYPE 3 (LIMITED HISTORY)

Scenario: The quantity of "Widget A" is updated from 70 to 80.

SCD Type 3: The previous quantity is stored in the **previous_quantity** column, and the current quantity is updated.

product_id	product_name	quantity	start_date	end_date	is_current	previous_quantity
101	Widget A	80	2024-02-01	NULL	TRUE	70
102	Widget B	30	2024-01-01	NULL	TRUE	NULL

NEXT DATA CRUNCH

WHICH SCD TYPE TO CHOOSE?

- **Type 1**: Use when tracking history isn’t necessary, only the most current value matters.
- **Type 2**: Use when you need full historical tracking of changes, particularly when business processes rely on knowing how data evolved.
- **Type 3:** Use for limited history where you only need to know the previous state alongside the current one.

SQL CODE FOR SCD TYPES

- **SCD Type 1 (Overwrite)**

```
Run on active connection | Select block
MERGE `project.dataset.table` AS target
USING `project.dataset.table_staging` AS source
ON target.product_id = source.product_id
WHEN MATCHED THEN
  UPDATE SET target.quantity = source.quantity;
WHEN NOT MATCHED THEN
  INSERT (product_id, product_name, quantity, start_date, end_date, is_current)
  VALUES (source.product_id, source.product_name, source.quantity, CURRENT_DATE(), NULL, TRUE);
```

SQL CODE FOR SCD TYPES

- **SCD Type 2 (Track Historical Changes)**

```
Run on active connection | Select block
MERGE `project.dataset.table` AS target
USING `project.dataset.table_staging` AS source
ON target.product_id = source.product_id
WHEN MATCHED AND target.quantity <> source.quantity THEN
  UPDATE SET target.end_date = CURRENT_DATE(), target.is_current = FALSE
WHEN NOT MATCHED THEN
  INSERT (product_id, product_name, quantity, start_date, end_date, is_current)
  VALUES (source.product_id, source.product_name, source.quantity, CURRENT_DATE(), NULL, TRUE);
```

SQL CODE FOR SCD TYPES

- **SCD Type 3 (Limited History)**

```
Run on active connection | Select block
MERGE `project.dataset.table` AS target
USING `project.dataset.table_staging` AS source
ON target.product_id = source.product_id
WHEN MATCHED AND target.quantity <> source.quantity THEN
  UPDATE SET target.previous_quantity = target.quantity, target.quantity = source.quantity
WHEN NOT MATCHED THEN
  INSERT (product_id, product_name, quantity, start_date, end_date, is_current, previous_quantity)
  VALUES (source.product_id, source.product_name, source.quantity, CURRENT_DATE(), NULL, TRUE, NULL);
```

NEXT DATA CRUNCH

THANK YOU FOR YOUR TIME!

If you have any questions or want to connect, feel free to reach out

NUR AZMI PRASETYO