Slowly Changing Dimension (SCD)

How to Implement SCD Type 2

Hello! I’m...

mahsasalimi

Data Enthusiast | BI Developer | Insight Creator

Come view my profile to learn more about my journey and how I can help your organization thrive!

578 1K 5K

Introduction

One of the most common methods in the realm of data warehousing is SCD Type 2.

This technique, unlike the conventional normalized approach which demands that a single update must be made whenever any change in the attribute of a particular dimension occurs, follows the history of changes in a dimension table through time.

In other words, it retains historical data rather than updating the already existing record by creating another record.

This way, changes in an entity such as a customer, product, or employee may be tracked; hence, one will be able to perform historical analysis effectively.

Concept

The whole concept of SCD Type 2 is that for every change, a new record is created. And each of these carries an identity -a surrogate key that would differentiate them from the past and present records.

Along with these, there are date fields and status flags like *is_active* to represent the time period where that particular version is valid.

Implementing SCD Type 2

Surrogate Key

An auto-incremented field that uniquely identifies each record version.

Natural Key

A field that links back to the original source system, such as *employee_id*.

Effective Date Range

Two fields (*start_date* and *end_date*) that define the validity period of each record.

Active Status

A Boolean field (*is_active*) that indicates whether the record is the current one.

Example

Creating the *dim_employee* Table

```
CREATE TABLE dim_employee (
    employee_key INT PRIMARY KEY IDENTITY(1,1),  -- Surrogate key (unique for each record version)
    employee_id INT,                             -- Natural key from OLTP system
    employee_name VARCHAR(100),
    employee_role VARCHAR(100),
    effective_start_date DATETIME,               -- When this version became valid
    effective_end_date DATETIME NULL,            -- When this version was replaced (NULL if active)
    is_active BIT                                -- Indicates if the record is current (1) or historical (0)
);
```

Example

When a new employee is added to the system, a record is inserted into the dimension table with the *effective_start_date* set to the current timestamp and *is_active* set to 1 to indicate that the record is current.

```
INSERT INTO dim_employee (employee_id, employee_name, employee_role, effective_start_date, is_active)
VALUES (101, 'John Doe', 'Developer', GETDATE(), 1);
```

Example

When an employee's role changes, we mark the current record as inactive by updating the *effective_end_date* and setting *is_active* to 0. Then, we insert a new record with the updated information.

```sql
-- Step 1: Mark the old record as inactive
UPDATE dim_employee
SET effective_end_date = GETDATE(), is_active = 0
WHERE employee_id = 101 AND is_active = 1;

-- Step 2: Insert the new record with updated information
INSERT INTO dim_employee (employee_id, employee_name, employee_role, effective_start_date, is_active)
VALUES (101, 'John Doe', 'Senior Developer', GETDATE(), 1);
```

Example

In this structure, the fact_sales table references the *employee_key* from the *dim_employee* table. This ensures that when you perform historical analysis, the fact table can connect to the correct version of the employee's record based on when the sale occurred

```
CREATE TABLE fact_sales (
    sales_id INT PRIMARY KEY,
    employee_key INT,                          -- Foreign key referencing dim_employee's surrogate key
    sale_amount DECIMAL(10, 2),
    sale_date DATETIME,
    FOREIGN KEY (employee_key) REFERENCES dim_employee(employee_key)
);
```

Historical Analysis with SCD Type 2

In SCD Type 2, the relationship between the fact and dimension tables is established using the surrogate key from the dimension table and a corresponding foreign key in the fact table. This ensures that the fact table refers to the correct version of the dimension record at the time the fact data (e.g., a sale) was recorded.
In the previous example: *dim_employee.employee_key* (surrogate key) is related to *fact_sales.employee_key* (foreign key).

- In the dim_employee table, the column *employee_key* (the surrogate key) uniquely identifies each version of an employee record.
- In the fact_sales table, the column *employee_key* acts as a foreign key that connects to the *employee_key* in *dim_employee*.