# Title: Implement SCD Type 1

## **Step 1: Create Tables**

### **Create the** Incremental\_product\_Dimension\_Grp1  **Table**:

This table will store the main customer dimension data.

CREATE TABLE Product\_Dimension\_grp1 (

    ProductID INT PRIMARY KEY,

    ProductName VARCHAR(100),

    ProductCategory VARCHAR(100),

    Price DECIMAL(10, 2)

);

### **Create the** Incremental\_product\_Dimension\_Grp1  **Table**:

This table simulates the incoming updates (new or changed data).

CREATE TABLE Incremental\_product\_Dimension\_Grp1 (

    ProductID INT PRIMARY KEY,

    ProductName VARCHAR(100),

    ProductCategory VARCHAR(100),

    Price DECIMAL(10, 2)

);

## **Step 2: Insert Initial Data into** Incremental\_product\_Dimension\_Grp1

Let’s insert some sample customer records into the Incremental\_product\_Dimension\_Grp1 table.

INSERT INTO Product\_Dimension\_grp1 (ProductID, ProductName, ProductCategory, Price)

VALUES

(1, 'Apple iPhone 14', 'Smartphone', 999.99),

(2, 'Samsung Galaxy S23', 'Smartphone', 849.99),

(3, 'HP Pavilion Laptop', 'Laptop', 699.99),

(4, 'Sony WH-1000XM5 Headphones', 'Headphones', 399.99),

(5, 'LG OLED TV', 'TV', 1299.99);

## **Step 3: Insert Updated Data into** Incremental\_product\_Dimension\_Grp1

Simulate incoming updated Incremental\_product\_Dimension\_Grp1  data. Some records will have changes, and others will remain the same.

INSERT INTO Incremental\_product\_Dimension\_Grp1 (ProductID, ProductName, ProductCategory, Price)

VALUES

(2, 'Samsung Galaxy S23', 'Watch', 849.99),

(3, 'HP Pavilion Laptop', 'Laptop', 699.99),

(6, 'Nintendo Switch', 'Gaming Console', 299.99),

(7, 'Bose QuietComfort 45 Headphones', 'Headphones', 249.99),

(8, 'Apple MacBook Pro', 'Laptop', 1999.99)

(9, 'Microsoft Surface Pro 9', 'Tablet', 1099.99),

(10, 'Samsung Galaxy Tab S8', 'Tablet', 849.99);

## **Step 4: Implement SCD Type 1 Logic**

We will use the **MERGE** statement to compare the Product\_Dimension\_grp1  table with the Incremental\_product\_Dimension\_Grp1  table.

* If a **match** is found based on CustomerID, the existing data will be **overwritten**.
* If there’s a new record (not matched), it will be **inserted**.

### **SCD Type 1 MERGE Statement**:

MERGE INTO Product\_Dimension\_Grp1 AS target

USING Incremental\_product\_Dimension\_Grp1 As source

ON target.ProductID = source.ProductID

WHEN MATCHED THEN

    UPDATE SET

        target.ProductName = source.ProductName,

        target.ProductCategory = source.ProductCategory,

        target.Price = source.Price

WHEN NOT MATCHED THEN

    INSERT (ProductID,ProductName,ProductCategory,Price)

    VALUES(source.ProductID,source.ProductName, source.ProductCategory, source.Price);

### **Explanation**:

1. **MERGE INTO**: Targets the Product\_Dimension\_Grp1  table.
2. **USING**: Specifies the Incremental\_product\_Dimension\_Grp1  table as the source.
3. **ON**: Matches records based on the ProductID.
4. **WHEN MATCHED**: If a match is found, updates the record in the Product\_Dimension\_Grp1  table.
5. **WHEN NOT MATCHED**: If no match is found, inserts the record as new.

## **Step 5: Validate the Results**

Run the following query to check the updated data in the Product\_Dimension\_Grp1  table:

**Expected Output** (Final Updated Table):

SELECT \* FROM Product\_Dimension\_Grp1 ;

A screenshot of a computer

Description automatically generated

**Summary of Changes:**

1. ProductCategory for ProductID **2**:
   * ProductCategory  “Smartphone“ updated to “Watch”
2. Newly added row for ProductID 6,7,8,9,10:

(6, 'Nintendo Switch', 'Gaming Console', 299.99),

(7, 'Bose QuietComfort 45 Headphones', 'Headphones', 249.99),

(8, 'Apple MacBook Pro', 'Laptop', 1999.99)

(9, 'Microsoft Surface Pro 9', 'Tablet', 1099.99),

(10, 'Samsung Galaxy Tab S8', 'Tablet', 849.99);

## **Clean-up (Optional)**:

If you want to reset your tables for further testing:

TRUNCATE TABLE Product\_Dimension\_Grp1 ;

TRUNCATE TABLE Incremental\_product\_Dimension\_Grp1 ;

### **Key Notes**:

* **MERGE** is the most efficient way to implement SCD Type 1 in SQL.
* Azure SQL Database fully supports the MERGE statement for such operations.
* You can integrate these scripts into stored procedures or pipelines in Azure Data Factory for automation.