# Title: Implement SCD Type 1

## **Step 1: Create Tables**

### **Create the Customer\_Dimension Table**:

This table will store the main customer dimension data.

CREATE TABLE Customer\_Dimension (

    CustomerID INT PRIMARY KEY,

    Name NVARCHAR(50),

    Email NVARCHAR(100),

    City NVARCHAR(50)

);

### **Create the Incoming\_Customer\_Data Table**:

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

CREATE TABLE Incoming\_Customer\_Data (

    CustomerID INT PRIMARY KEY,

    Name NVARCHAR(50),

    Email NVARCHAR(100),

    City NVARCHAR(50)

);

## **Step 2: Insert Initial Data into Customer\_Dimension**

Let’s insert some sample customer records into the Customer\_Dimension table.

INSERT INTO Customer\_Dimension (CustomerID, Name, Email, City)

VALUES

(1, 'Alice Smith', 'alice.smith@email.com', 'New York'),

(2, 'Bob Johnson', 'bob.johnson@email.com', 'Los Angeles'),

(3, 'Charlie Brown', 'charlie.brown@email.com', 'Chicago'),

(4, 'David Wilson', 'david.wilson@email.com', 'Houston');

## **Step 3: Insert Updated Data into Incoming\_Customer\_Data**

Simulate incoming updated customer data. Some records will have changes, and others will remain the same.

INSERT INTO Incoming\_Customer\_Data (CustomerID, Name, Email, City)

VALUES

(2, 'Bob Johnson', 'bob.j@email.com', 'San Francisco'),  -- Updated Email and City

(3, 'Charlie Brown', 'charlie.b@email.com', 'Chicago'),  -- Updated Email

(4, 'David Wilson', 'david.wilson@email.com', 'Dallas'); -- Updated City

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

We will use the **MERGE** statement to compare the Incoming\_Customer\_Data table with the Customer\_Dimension 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 Customer\_Dimension AS target

USING Incoming\_Customer\_Data AS source

ON target.CustomerID = source.CustomerID

WHEN MATCHED THEN

    UPDATE SET

        target.Name = source.Name,

        target.Email = source.Email,

        target.City = source.City

WHEN NOT MATCHED THEN

    INSERT (CustomerID, Name, Email, City)

    VALUES (source.CustomerID, source.Name, source.Email, source.City);

### **Explanation**:

1. **MERGE INTO**: Targets the Customer\_Dimension table.
2. **USING**: Specifies the Incoming\_Customer\_Data table as the source.
3. **ON**: Matches records based on the CustomerID.
4. **WHEN MATCHED**: If a match is found, updates the record in the Customer\_Dimension 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 Customer\_Dimension table:

SELECT \* FROM Customer\_Dimension;

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

CustomerID Name Email City

1 Alice Smith alice.smith@email.com New York

2 Bob Johnson bob.j@email.com San Francisco

3 Charlie Brown charlie.b@email.com Chicago

4 David Wilson david.wilson@email.com Dallas

## **Summary of Changes**:

1. CustomerID **2**:
   * Email updated to bob.j@email.com
   * City updated to San Francisco
2. CustomerID **3**:
   * Email updated to charlie.b@email.com
3. CustomerID **4**:
   * City updated to Dallas
4. CustomerID **1**:
   * Remained unchanged as there was no incoming update.

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

If you want to reset your tables for further testing:

TRUNCATE TABLE Customer\_Dimension;

TRUNCATE TABLE Incoming\_Customer\_Data;

### **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.

Let me know if you need further assistance or enhancements! 😊