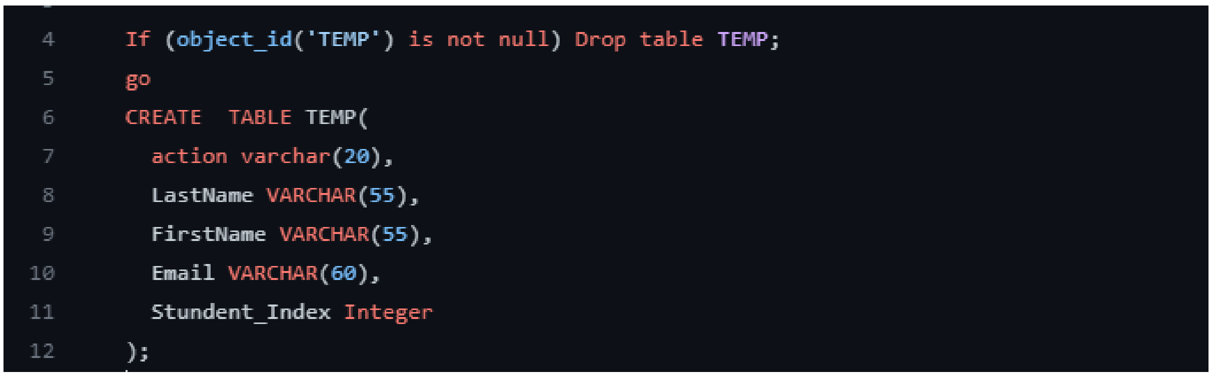
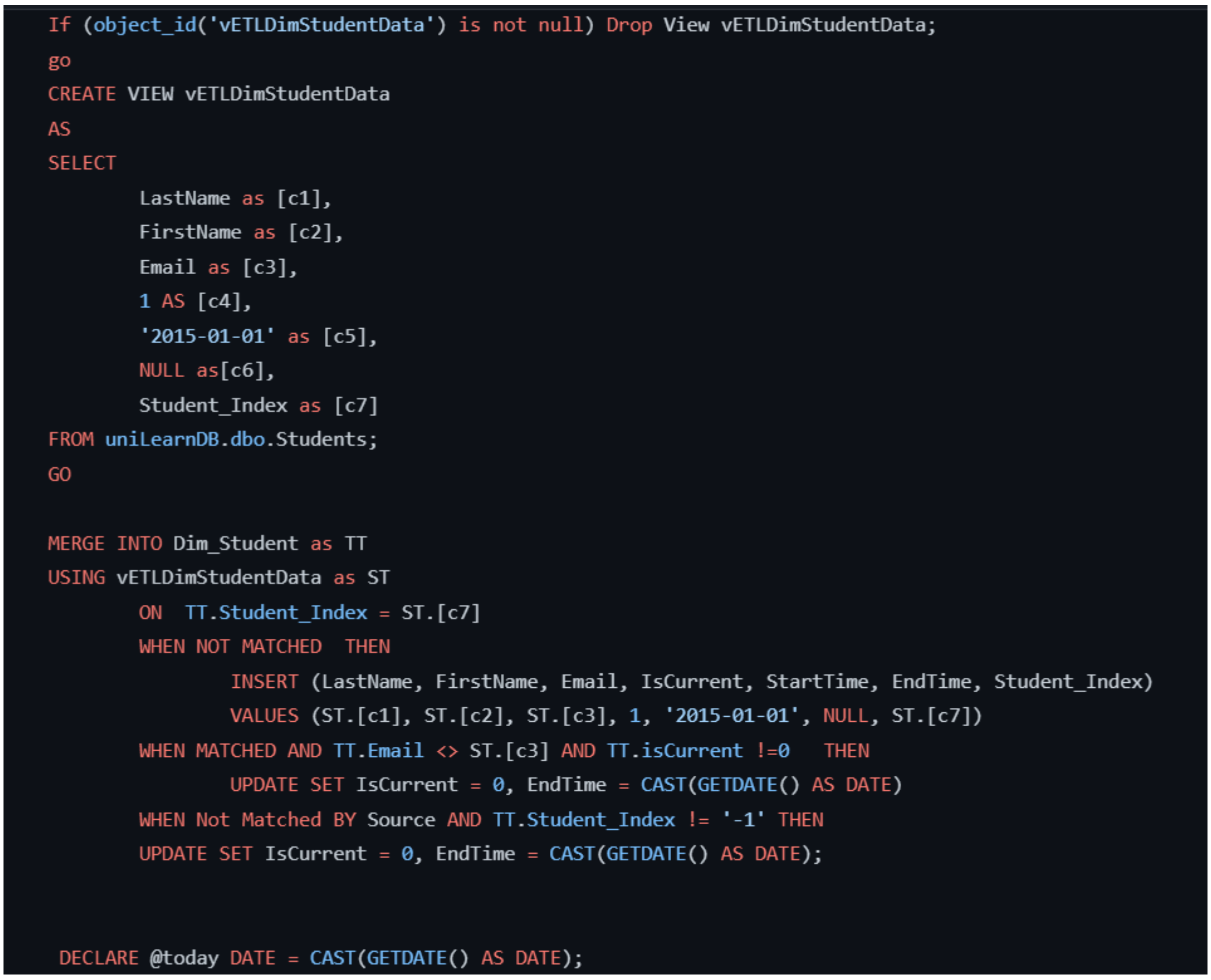
We have implemented SCD 2 (Slowly Changing Dimension Type 2) for the **Dim_Student** dimension to handle changes in personal data. We have added the following attributes to specify it:
IsCurrent: A boolean attribute to indicate if the dimension record is current.
IndexNumber: A business key that uniquely identifies each student.
EndTime: The date when the
IsCurrent status changed, indicating the end of the previous record.
StartTime: The date when a new tuple was created for the student's dimension record.



```sql
If (object_id('TEMP') is not null) Drop table TEMP;
go
CREATE  TABLE TEMP(
  action varchar(20),
  LastName VARCHAR(55),
  FirstName VARCHAR(55),
  Email VARCHAR(60),
  Stundent_Index Integer
);
```

First line of code first checks for the existence of a temporary table named "**TEMP**" and drops it if it exists. Then, it creates a new table with columns to store information related to actions, last name, first name, email, and student index.

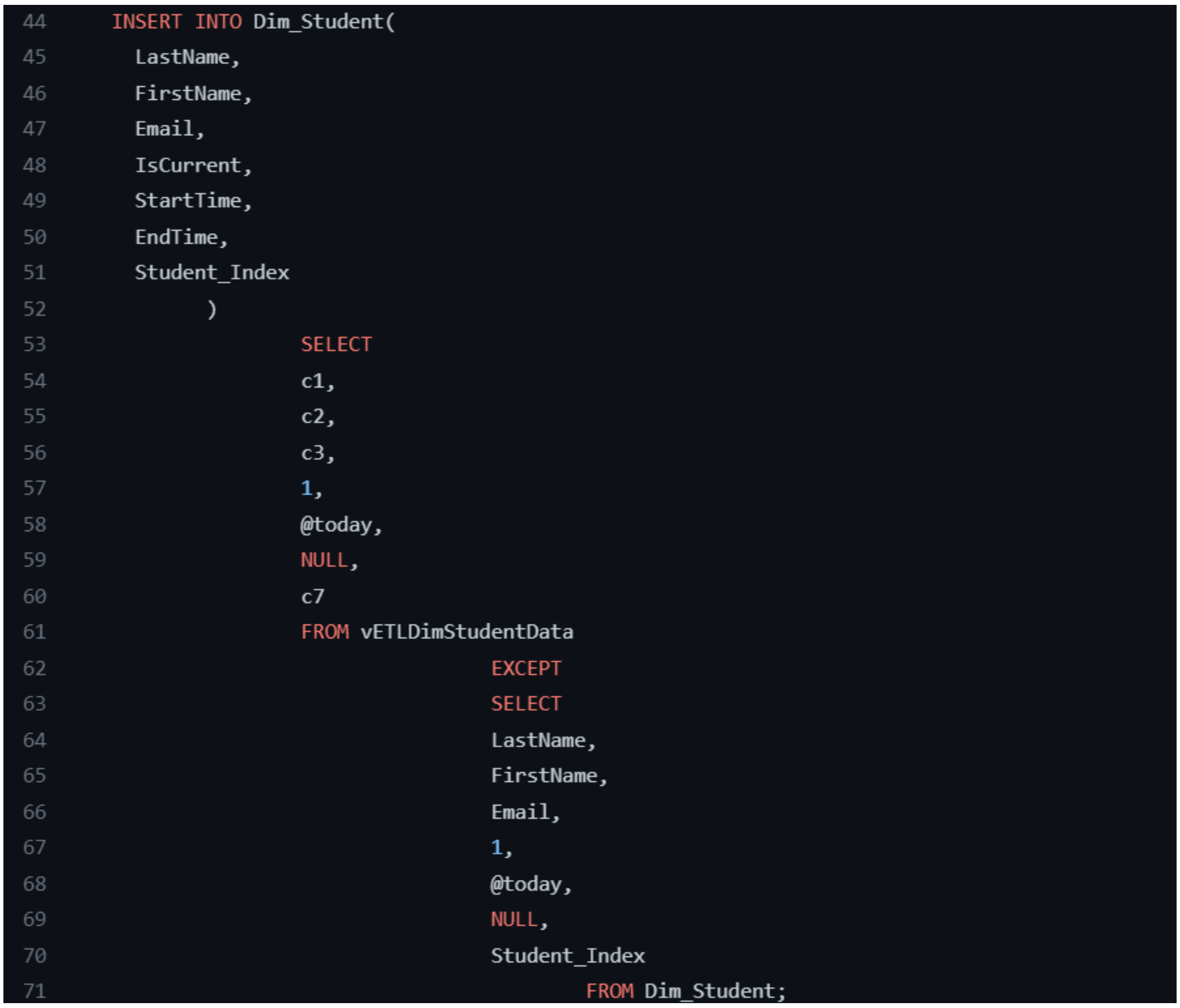


```sql
If (object_id('vETLDimStudentData') is not null) Drop View vETLDimStudentData;
go
CREATE VIEW vETLDimStudentData
AS
SELECT
        LastName as [c1],
        FirstName as [c2],
        Email as [c3],
        1 AS [c4],
        '2015-01-01' as [c5],
        NULL as[c6],
        Student_Index as [c7]
FROM uniLearnDB.dbo.Students;
GO

MERGE INTO Dim_Student as TT
USING vETLDimStudentData as ST
        ON  TT.Student_Index = ST.[c7]
        WHEN NOT MATCHED  THEN
                INSERT (LastName, FirstName, Email, IsCurrent, StartTime, EndTime, Student_Index)
                VALUES (ST.[c1], ST.[c2], ST.[c3], 1, '2015-01-01', NULL, ST.[c7])
        WHEN MATCHED AND TT.Email <> ST.[c3] AND TT.isCurrent !=0   THEN
                UPDATE SET IsCurrent = 0, EndTime = CAST(GETDATE() AS DATE)
        WHEN Not Matched BY Source AND TT.Student_Index != '-1' THEN
        UPDATE SET IsCurrent = 0, EndTime = CAST(GETDATE() AS DATE);


 DECLARE @today DATE = CAST(GETDATE() AS DATE);
```

First line of code checks for the existence of a view called "**vETLDimStudentData**" and drops it if it exists. Then, it creates the view by selecting specific columns from the "**Students**" table from source DB. Afterwards, it performs an upsert operation on the "Dim_Student" table using the view as the source data. The code handles various scenarios such as inserting new records, updating existing records, and updating records that are no longer present in the source data. It uses the **MERGE** statement to perform an upsert operation on the "**Dim_Student**" table. The source data for the upsert operation is the "**vETLDimStudentData**" view. When a match is found between the source and target based on the Student_Index, it updates the target table's columns with values from the source. When a match is found and the email in the target table is different from the email in the source table, and the IsCurrent column in the target table is not 0, it updates the IsCurrent column and sets the EndTime to the current date. When a match is not found in the source data and the Student_Index in the target table is not '-1', it updates the IsCurrent column and sets the EndTime to the current date.



```
    INSERT INTO Dim_Student(
      LastName,
      FirstName,
      Email,
      IsCurrent,
      StartTime,
      EndTime,
      Student_Index
            )
                    SELECT
                    c1,
                    c2,
                    c3,
                    1,
                    @today,
                    NULL,
                    c7
                    FROM vETLDimStudentData
                                    EXCEPT
                                    SELECT
                                    LastName,
                                    FirstName,
                                    Email,
                                    1,
                                    @today,
                                    NULL,
                                    Student_Index
                                            FROM Dim_Student;
```

This code inserts new records into the "**Dim_Student**" table based on the data from the "**vETLDimStudentData**" view, excluding any records that already exist in the "**Dim_Student**" table. It does this by using the **EXCEPT** operator, which returns the rows from the first query that are not present in the second query. The first query selects the columns c1, c2, c3, c7 from the "**vETLDimStudentData**" view, and the second query selects the corresponding columns from the "**Dim_Student**" table. The **INSERT INTO** statement specifies the columns to insert into: LastName, FirstName, Email, IsCurrent, StartTime, EndTime, and Student_Index. The **SELECT** statement provides the values for these columns based on the result of the **EXCEPT** operation.