DWBI- Practical 4

1. Introduction to the Worksheet

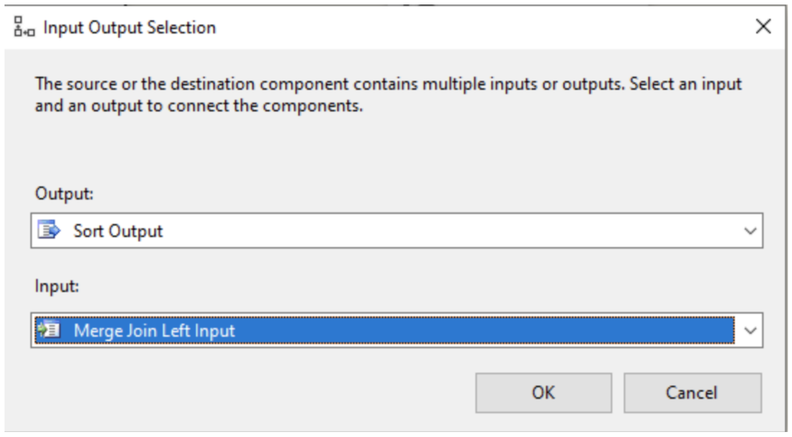
In the previous worksheet, we transformed and loaded product category, sub category and product data from the staging database to the data warehouse using SSIS. This worksheet focuses on loading customer data into the data warehouse with necessary transformations.

1. Customer Data Transformation and Loading

Follow the step given below:

1. Open Visual Studio Data Tools in 'Administrator' mode and open previously created solution.
2. Open '**SLIIT_Retail_Load_DW.dtsx**'.
3. Drag and drop a ***Data Flow Task***, rename it as '**Transform and Load Customer Data**' and go to the ***Data Flow*** tab to design the '**Transform and Load Customer Data**' data flow.
4. Drag and drop ***OLE DB Source*****,** rename as '**Extract from Customer Staging**' and configure it to extract the '**StgCustomer**' table. Make sure all the columns are selected.
5. Drag and drop another ***OLE DB Source***, rename as '**Extract from Customer Addresses Staging**' and configure it to extract the '**StgCustomerAddresses**' table. Make sure all the columns are selected.
6. Drag and drop a ***Sort*** component from ***SSIS Toolbox***, rename it as '**Customer Sort**' and connect it with the '**Extract from Customer Staging**' component using the blue line.
7. Double click '**Customer Sort**' component and select '**CustomerID**' as the sorting column by ticking on the checkbox in front of '**CustomerID**'. Click ***OK***.
8. Drag and drop another ***Sort*** component, rename it as '**Customer Address Sort**' and connect it with the '**Extract from Customer Addresses Staging**' component using the blue line.
9. Double click '**Customer Address Sort**' component and select '**CustomerID**' as the sorting column. Click ***OK***.

10. Drag and drop ***Merge Join*** component and link the ‘**Customer Sort**’ component using the blue line.
11. In the ***Input Output Selection*** window, select ***Merge Join Left Input*** as the value for ***Input*** field. Click ***OK***.



Input Output Selection

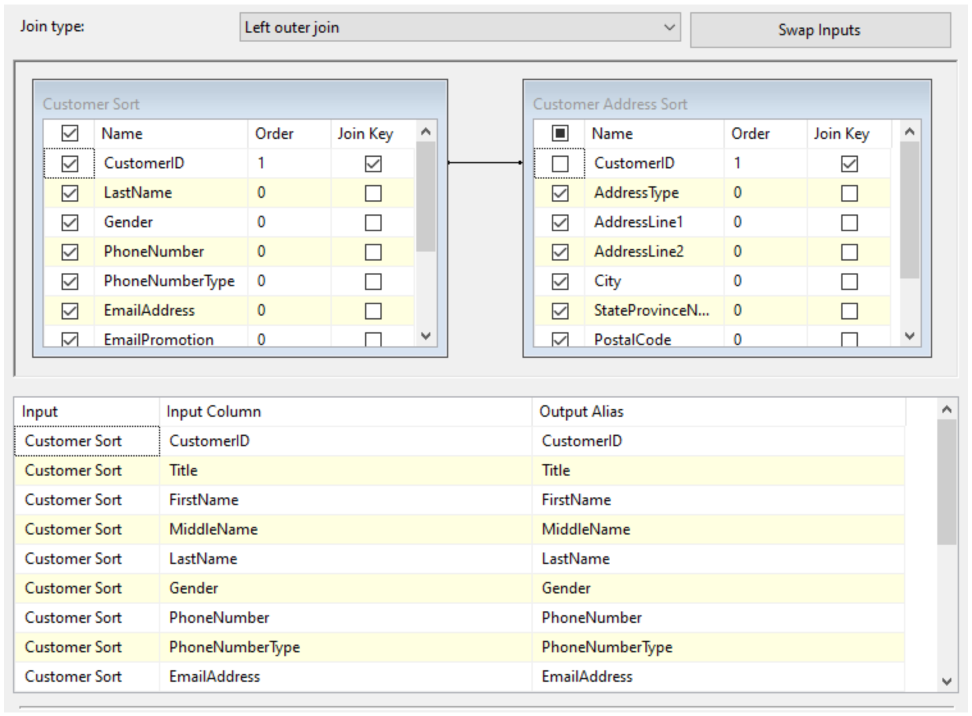
The source or the destination component contains multiple inputs or outputs. Select an input and an output to connect the components.

Output: Sort Output

Input: Merge Join Left Input

12. Link the ‘**Customer Address Sort**’ component to ***Merge Join*** using the blue link of ‘**Customer Address Sort**’ and double click ***Merge Join*** to configure the join.
13. In the ***Merge Join Transformation Editor*** window, you should be able to see two sources are connected by ‘**CustomerID**’ of both ‘**Customer Sort**’ and ‘**Customer Address Sort**’ and the ***Join Type*** is ***Inner Join*.** Joining columns are automatically picked based on the sort column you provided in the ***Sort*** components. Let’s assume even if there are customers without corresponding addresses, we still load them in to the data warehouse table ‘**DimCustomer**’. Thus, change the ***Join Type*** to ***Left Outer Join***.
14. Select all the fields from ‘**Customer Sort**’ and all from ‘**Customer Address Sort**’ except ‘**CustomerID**’. Click ***OK***.

    Final selection should have ‘**CustomerID**’, ‘**Title**’, ‘**FirstName**’, ‘**MiddleName**’, ‘**LastName**’, ‘**Gender**’, ‘**PhoneNumber**’, ‘**PhoneNumberType**’, ‘**EmailAddress**’, ‘**EmailPromotion**’, ‘**AddressType**’, ‘**AddressLine1**’, ‘**AddressLine2**’, ‘**City**’, ‘**StateProvinceName**’, ‘**PostalCode**’, ‘**CountryRegionGroup**’.



Join type: Left outer join | Swap Inputs

Customer Sort

☑	Name	Order	Join Key
☑	CustomerID	1	☑
☑	LastName	0	☐
☑	Gender	0	☐
☑	PhoneNumber	0	☐
☑	PhoneNumberType	0	☐
☑	EmailAddress	0	☐
☑	EmailPromotion	0	☐

Customer Address Sort

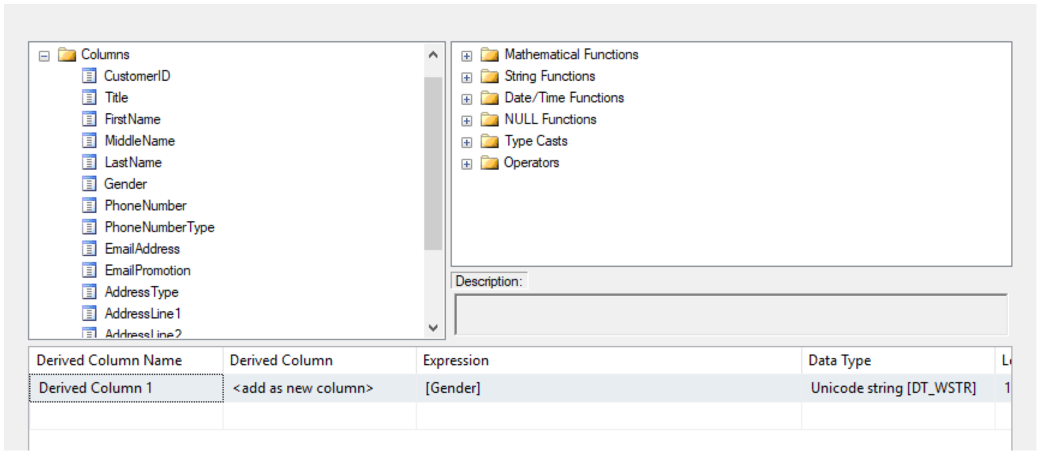
■	Name	Order	Join Key
☐	CustomerID	1	☑
☑	AddressType	0	☐
☑	AddressLine1	0	☐
☑	AddressLine2	0	☐
☑	City	0	☐
☑	StateProvinceN...	0	☐
☑	PostalCode	0	☐

Input	Input Column	Output Alias
Customer Sort	CustomerID	CustomerID
Customer Sort	Title	Title
Customer Sort	FirstName	FirstName
Customer Sort	MiddleName	MiddleName
Customer Sort	LastName	LastName
Customer Sort	Gender	Gender
Customer Sort	PhoneNumber	PhoneNumber
Customer Sort	PhoneNumberType	PhoneNumberType
Customer Sort	EmailAddress	EmailAddress

15. Next let's apply some data cleansing steps. Based on the customer data profile analysis, '**Gender**' and '**Title**' fields contained ***NULL*** values. First let's try to transform ***NULL*** values in '**Gender**' field and then based on the value in '**Gender**' field, value for '**Title**' can be populated.

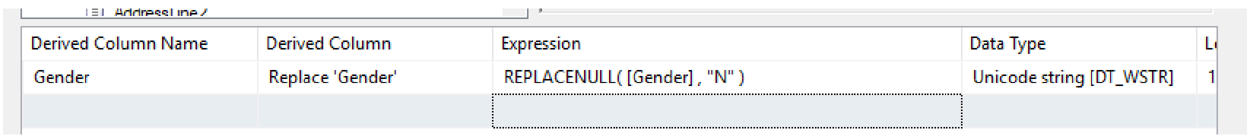
    Drag and drop ***Derived Column*** component, rename it as '**Replace NULL Gender Values**' and connect with the ***Merge Join*** component.

16. Double click on '**Replace NULL Gender Values**' to open ***Derived Column Transformation Editor*** window and expand ***Columns*** folder on the left side box and drag '**Gender**' to the grid below.



Derived Column Name	Derived Column	Expression	Data Type
Derived Column 1	<add as new column>	[Gender]	Unicode string [DT_WSTR]

17. In the grid, rename the value in the ***Derived Column Name*** column from '**Derived Column 1**' to '**Gender**' and select ***Replace 'Gender'*** as the value for ***Derived Column*** column from the drop down list.
18. Expand ***Null Functions*** folder on the right side box, drag and drop ***REPLACENULL(<<expression>>, <<expression>>)*** to ***Expression*** field in the grid below. Replace the first ***<<expression>>*** with '**Gender**' and the second ***<<expression>>*** with **"N"**. This will search and replace all NULL gender values with the value '**N**'.

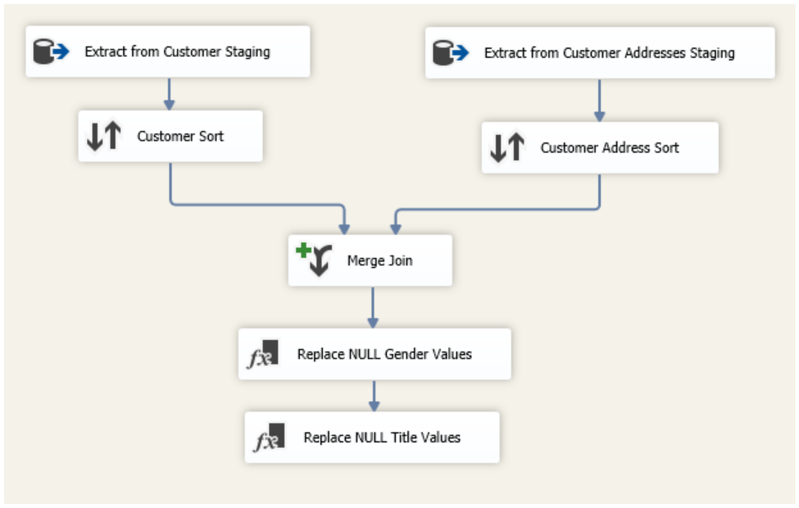


Derived Column Name	Derived Column	Expression	Data Type	L
Gender	Replace 'Gender'	REPLACENULL([Gender] , "N")	Unicode string [DT_WSTR]	1

19. Click **OK**.
20. Drag and drop another ***Derived Column*** component, rename it as '**Replace NULL Title Values**' and connect with the '**Replace NULL Gender Values**' component.
21. Double click on '**Replace NULL Title Values**' to open ***Derived Column Transformation Editor*** window and expand ***Columns*** folder on the left side box and drag '**Title**' to the grid below.
22. In the grid, set the ***Derived Column Name*** as '**Title**' and select ***Replace 'Title'*** as the value for ***Derived Column***.
23. Use ***REPLACENULL(<<expression>>, <<expression>>)*** and set the ***Expression*** as below:

    Gender == "M" ? "Mr" : Gender == "F" ? "Ms" : "NA"

    NOTE: When you copy paste the expression, you might have to re-type the double quotes, the double quotes may not be recognized properly.
24. Click ***OK***.
25. By now '**Transform and Load Customer Data**' flow should look like below:

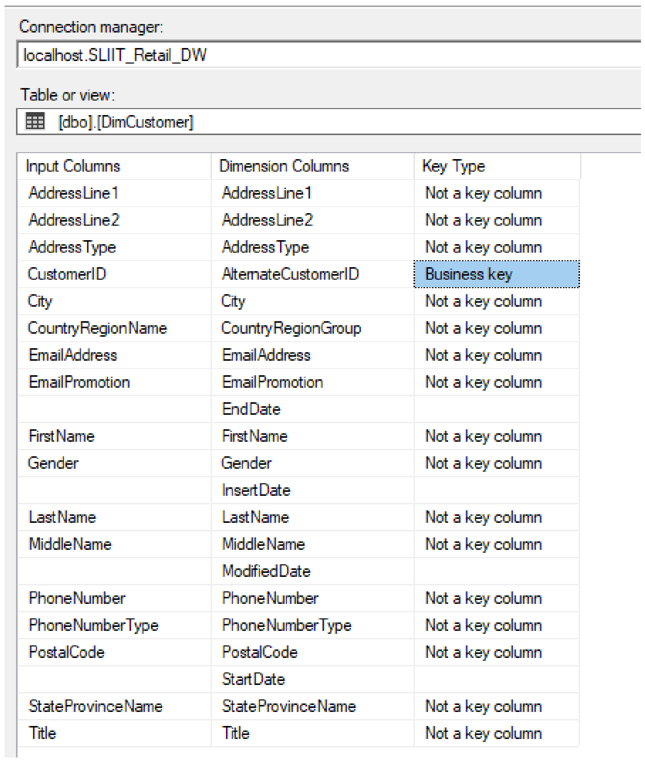

Extract from Customer Staging
Extract from Customer Addresses Staging
Customer Sort
Customer Address Sort
Merge Join
Replace NULL Gender Values
Replace NULL Title Values

26. Now, we have data prepared and ready to be inserted to the customer dimension. However, customer dimension requires history management; customer addresses can change from time to time and we need to keep track of historical addresses of customers. Additionally, if the customer changes the phone number, we should replace the old phone number with the new number. Thus, '**DimCustomer**' should be loaded as a Slowly Changing Dimension.

    Drag and drop a ***Slowly Changing Dimension*** component, rename it as '**Customer SCD**' and connect it with the '**Replace NULL Title Values**' component.
27. Double click on the '**Customer SCD**' to open ***Slowly Changing Dimension Wizard*** and click ***Next*** to start configuring.
28. In ***Select a Dimension Table and Keys*** window, make sure the ***Connection manager*** is pointing to the '**SLIIT_Retail_DW**', and select '**DimCustomer**' from the ***Table or view*** drop down list.
29. Set ***Input Columns***, ***Dimension Columns*** and ***Key Type*** attributes as below:

    Leave the '**StartDate**' and '**EndDate**' unassigned since those two fields are used to determine the latest (current) record.

    Leave the '**InsertDate**' and '**ModifiedDate**' unassigned for the moment. Though this is to manage the records insert and modified date-time to/in the '**DimCustomer**', handling this will make the scenario complex. Thus, leave these 2 fields behind for this table. *Consider this as a bit of research work for you!!!*



Connection manager:
localhost.SLIIT_Retail_DW

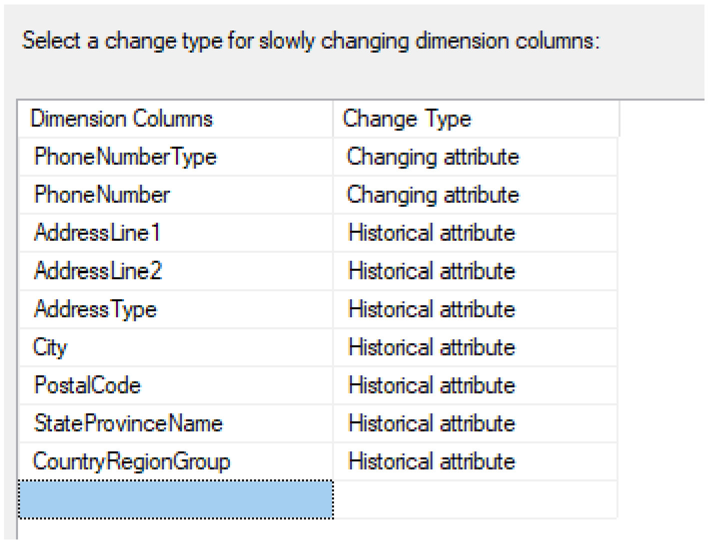
Table or view:
[dbo].[DimCustomer]

Input Columns	Dimension Columns	Key Type
AddressLine1	AddressLine1	Not a key column
AddressLine2	AddressLine2	Not a key column
AddressType	AddressType	Not a key column
CustomerID	AlternateCustomerID	Business key
City	City	Not a key column
CountryRegionName	CountryRegionGroup	Not a key column
EmailAddress	EmailAddress	Not a key column
EmailPromotion	EmailPromotion	Not a key column
	EndDate	
FirstName	FirstName	Not a key column
Gender	Gender	Not a key column
	InsertDate	
LastName	LastName	Not a key column
MiddleName	MiddleName	Not a key column
	ModifiedDate	
PhoneNumber	PhoneNumber	Not a key column
PhoneNumberType	PhoneNumberType	Not a key column
PostalCode	PostalCode	Not a key column
	StartDate	
StateProvinceName	StateProvinceName	Not a key column
Title	Title	Not a key column

In this window, you defined the table which will be treated as a slowly changing dimension and the fields you use to identify a unique record in the table.

Set the ***Key Type*** of '**CustomerID**' to ***Business Key*** since it is the unique value column for the table. Click ***Next***.

30. In the ***Slowly Changing Dimension Columns*** window, add fields to the ***Dimension Columns*** and set the ***Change Type*** as below: (When you click on an empty row in the grid you will get a drop down list from which you can select required fields).



Select a change type for slowly changing dimension columns:

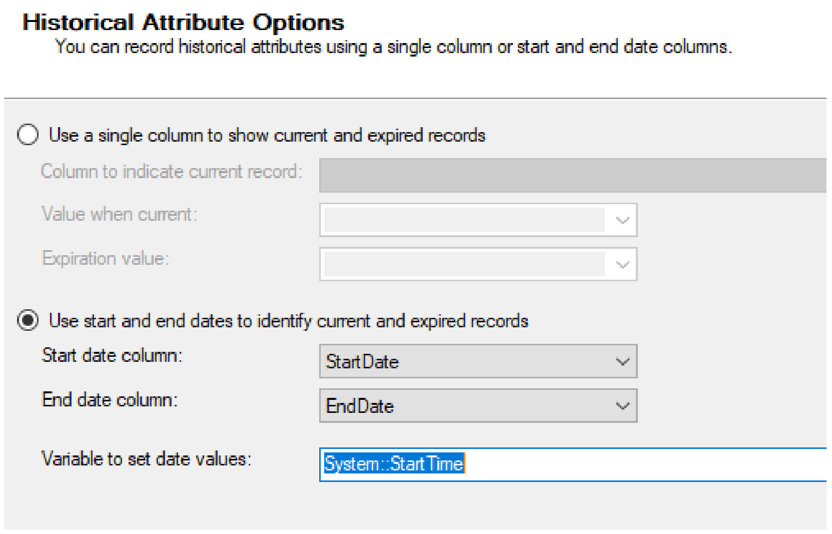
Dimension Columns	Change Type
PhoneNumberType	Changing attribute
PhoneNumber	Changing attribute
AddressLine1	Historical attribute
AddressLine2	Historical attribute
AddressType	Historical attribute
City	Historical attribute
PostalCode	Historical attribute
StateProvinceName	Historical attribute
CountryRegionGroup	Historical attribute

There are 3 types of ***Change Types***:

- ***Fixed attribute***: These are Type-0 attributes; when the source value changes, nothing happens in the target (data warehouse table).
- ***Changing attributes***: These are Type-1 attributes; when the source value changes, target field is also updated.
- ***Historical attribute***: These are Type-2 attributes; when the source value changes, a new record will be created the in the target table.

Unspecified fields, in this window will be treated as fixed attributes. However, when a slowly changing attribute is updated, even a fixed attribute will pick the latest value.

31. In the ***Fixed and Changing Attribute Options*** window, do no modifications and leave the options as it is. This window allows you to configure how to treat expired records; essentially this is to implement Type-5 or Type-6 SCD.
32. In the ***Historical Attribute Options*** window, do the configuration as shown below:


Historical Attribute Options
You can record historical attributes using a single column or start and end date columns.

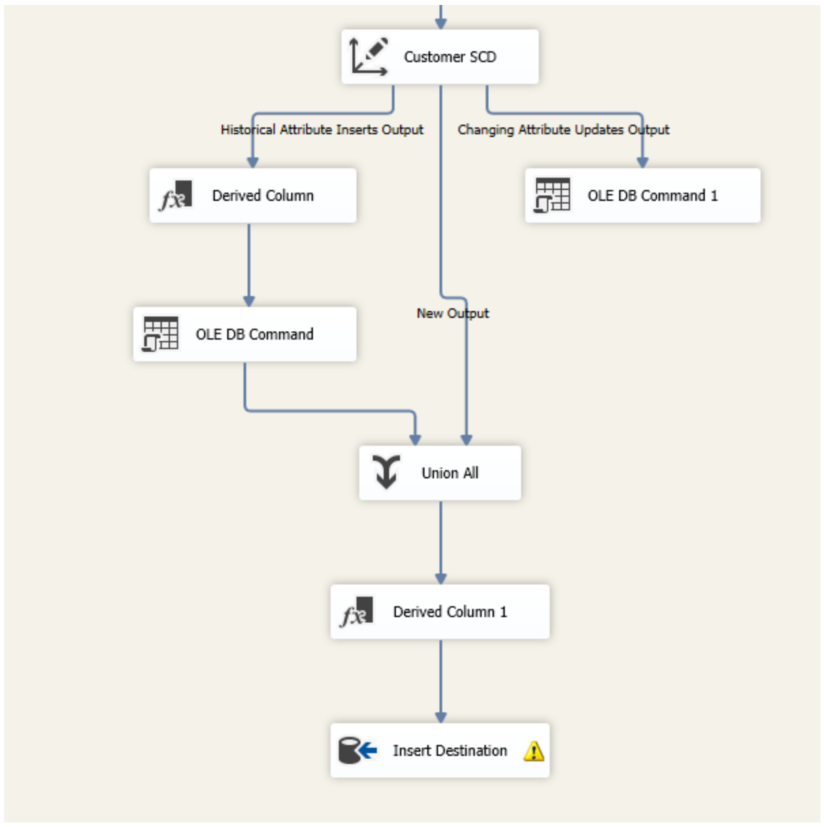
( ) Use a single column to show current and expired records
Column to indicate current record:
Value when current:
Expiration value:

(●) Use start and end dates to identify current and expired records
Start date column: StartDate
End date column: EndDate
Variable to set date values: System::StartTime

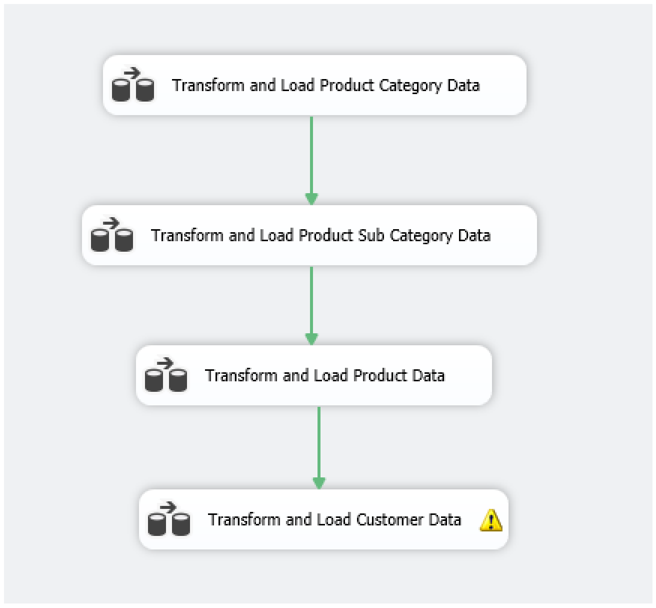
33. In the ***Inferred Dimension Members*** window, uncheck the checkbox of ***Enable inferred member support*** and click ***Next*** and then ***Finish*** to complete the configuration.

Once you click finish, rest of the components should be auto generated and should look like below:

(Read more about Inferred members in data warehousing to understand what it does)



34. In SSIS Control Flow tab, connect '**Transform and Load Product Data**' and '**Transform and Load Customer Data**' data flows in a way that customer records are loaded after the product records into the data warehouse.


Transform and Load Product Category Data
Transform and Load Product Sub Category Data
Transform and Load Product Data
Transform and Load Customer Data

35. Modify existing customer data in ‘**StgCustomer**’ table and observer how changes are managed in the ‘**DimCustomer**’ table.

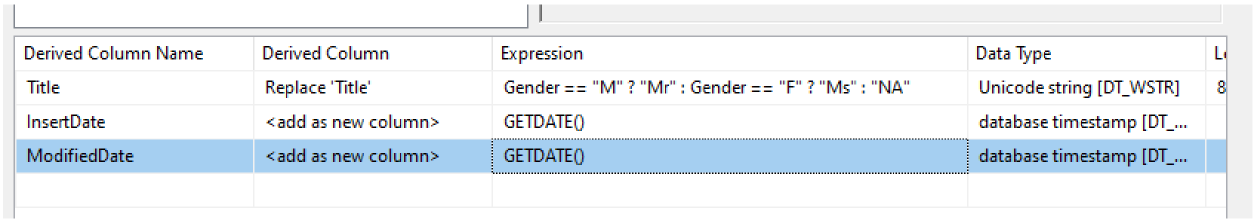
2. Handling Insert and Modified Date-Time

In Step 29, we left ‘**InsertDate**’ and ‘**ModifiedDate**’ unassigned with any values. Let’s see how we can incorporate these 2 attributes to the data flow now.

First we need to pass it as a field into the ‘**Customer SCD**’.

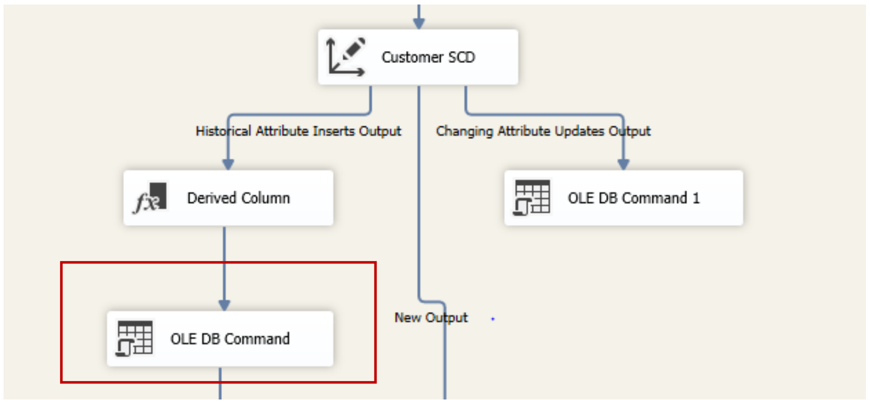
1. Double click on ‘**Replace NULL Title Values**’ to open ***Derived Column Transformation Editor*** window and add 2 more new columns as ‘**InsertDate**’ and ‘**ModifiedDate**’ to the grid below.

   Expand ***Date/Time Functions*** folder on the right side box, drag and drop ***GETDATE() function*** to ***Expression*** field of both new columns in the grid. The grid should look like below:



Derived Column Name	Derived Column	Expression	Data Type	L
Title	Replace 'Title'	Gender == "M" ? "Mr" : Gender == "F" ? "Ms" : "NA"	Unicode string [DT_WSTR]	8
InsertDate	<add as new column>	GETDATE()	database timestamp [DT_...	
ModifiedDate	<add as new column>	GETDATE()	database timestamp [DT_...	

2. Double click on the ***OLE DB Command*** component which was generated automatically when you configured the ‘**Customer SCD**’ component to open

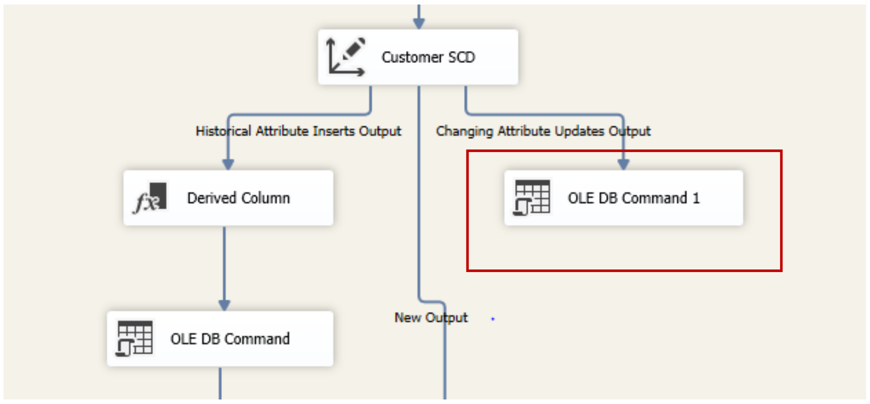

Customer SCD
Historical Attribute Inserts Output
Changing Attribute Updates Output
Derived Column
OLE DB Command 1
New Output
OLE DB Command

3. In the ***Advanced Editor for OLE DB Command*** window, go to Component Properties tab and edit the ***SqlCommand*** property.

Replace the existing commend with the command given below to update the modified date whenever a record is modified due to historical attribute modification:

```
UPDATE [dbo].[DimCustomer]
SET [EndDate] = ? , ModifiedDate = GETDATE()
WHERE [AlternateCustomerID] = ? AND [EndDate] IS NULL
```

4. Similarly, double click on the ***OLE DB Command 1*** component which was generated automatically when you configured the '**Customer SCD**' component to open


Customer SCD
Historical Attribute Inserts Output
Changing Attribute Updates Output
Derived Column
OLE DB Command 1
New Output
OLE DB Command

5. In the ***Advanced Editor for OLE DB Command*** window, go to Component Properties tab and edit the ***SqlCommand*** property.

   Replace the existing commend with the command given below to update the modified date whenever a record is modified due to a changing attribute modification:

```
UPDATE [dbo].[DimCustomer]
SET [PhoneNumber] = ?,[PhoneNumberType] = ? , ModifiedDate = GETDATE()
WHERE [AlternateCustomerID] = ? AND [EndDate] IS NULL
```

6. Modify existing customer data in '**StgCustomer**' table and observer how changes are managed in the '**DimCustomer**' table.