SQL Server Integration Services Tutorial

This tutorial offers an introduction into Microsoft's SQL Server Integration services (SSIS). SSIS is used for building an enterprise level data integration and data transformation solutions. The external data source may be an OLTP system, a spreadsheet or a flat file. The data source we are considering in this document is a Microsoft Excel workbook.

This document is divided into two parts:

- Installation and instruction for SSIS (This part has been removed from this tutorial and replaced with Prerequisites section below which includes the updated steps and latest versions of the required software, tools and extensions to complete this tutorial)
- An example to illustrate how to use SSIS to perform ETL (Extract, Transform, Load) on the excel data source.

After finishing this tutorial, the reader will be able to:

1. Understand foundations of the SSIS module and how it supports ETL transformations.
2. Apply the theoretical concepts of SSIS in designing and implementing a complete ETL workflow.
3. To use an Excel file as the data source and a data warehouse schema implemented in MSSQL as the target artifact.
4. Apply the following kinds of ETL transformations.
   - Filter null values
   - Split a field into multiple fields
   - Validate data quality and integrity using foreign key constraints

1. Prerequisites:

This document assumes you are working on CBA Virtual Analytics Lab OR you have installed the software and tools listed below. It also assumes that you have completed the following tutorial and created the required database and tables on SSMS: **2_0_Create Database and Tables in SSMS**

1) SQL Server Developer Edition (Related tutorial: *0_1_SQL Server and SQL Server Management Studio Installation*)
2) SQL Server Management Studio (Related tutorial: *0_1_SQL Server and SQL Server Management Studio Installation*)
3) Visual Studio with SQL Server Data Tools (Related Tutorial: *0_2_Visual Studio Installation for SQL Server Integration Services*)
4) SQL Server Integration Services Projects for Visual Studio (Related Tutorial: *0_2_Visual Studio Installation for SQL Server Integration Services*)

If you are working on your Data Science Virtual Machine and would like to give a break and continue working on the project later, you can do so, as how you do on your personal computer. Please make sure you saved all your work before closing the remote desktop window. If you experience server connection errors after restarting your Data Science Virtual Machine, you may need to open the **Run** prompt and run the **%Temp%** command again to clear the cache as explained in the first tutorial: **2_0_Create Database and Tables in SSMS.**

2. Using SSIS to perform ETL Transformations

This section is divided into three parts:

1. A discussion of the source and target schemas
2. A brief introduction to SSIS ETL transformation terminology.
3. Detailed steps illustrating sample transformation of Hubspot data from an Excel spreadsheet into our data warehouse schema.

2.1 Source and Target Schemas

For source and target schemas we use a Hubspot data export from the Teradata Data Challenge 2016 as the data source for our data warehouse. Hubspot is a full stack of software for marketing, sales, and customer service, built around client relationship management software.

5. The schema as shown in below represents a consolidated view to support web analytics for the inbound marketing efforts across multiple social media channels such as Facebook, Twitter, LinkedIn. Some of the fields are null, and some of the data must be transformed to be useful. This data will be loaded into a multidimensional schema for further analysis, but it must be cleaned and transformed first. The data consists of the following fields -

Field
Status
Channel
Account
Campaign
Published Message
Title
Shortened Link
Original Link
Clicks
Total Interactions
Twitter Retweets
Twitter Replies
Twitter Favorites
Facebook Likes
Facebook Comments
LinkedIn Likes
LinkedIn Comments
Publish Time
Created Time

Figure 8: Hubspot Data from the Teradata Data Challenge 2016

The hubspot data for this tutorial refers to the Excel file provided in the supporting files accompa nying this tutorial. The name of the file is: hubspot-social-media-export-01-08-16.xls

6. The figure below shows us our data warehouse schema. The schema has one fact (fact_interactions) and two dimensions (dim_channel and dim_time). The attributes *ChannelId* and *TimeNo* in the fact_interactions table are the foreign keys to the two dimension tables dim_channel and dim_time.

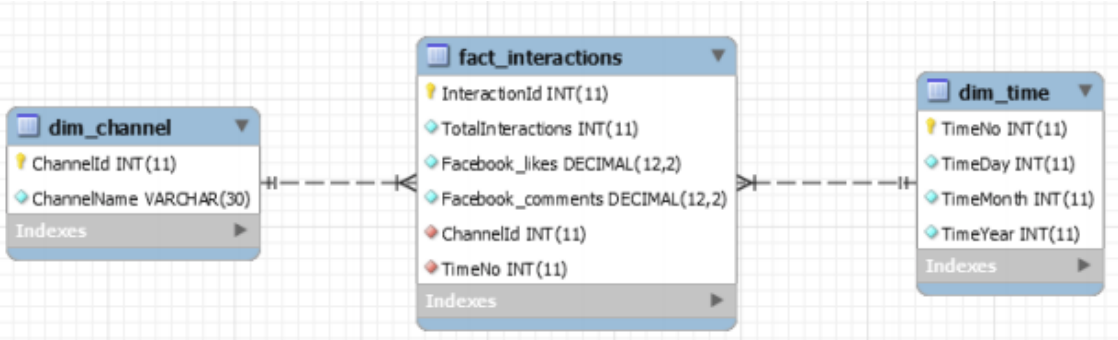


Figure 9: A Sample Data Warehouse Schema for Hubspot Data

You have created the above schema in SSMS using the DDL script provided in the accompanied files. The name of the DDL script is: hubspot.sql (Microsoft SQL Server Query File).

At this point, we have our input and output artifacts in place. In preparation to understand the example, we offer a brief introduction to the terminology related to the SSIS Data Integration module in the next section.

2.2 SSIS Data Integration Module Terminology

This section introduces and explains terms used in the SSIS Data Integration software.

7. A data flow task is composed of transformations which are connected by *data flow paths*. Transformations have different kind of functionality based on the transformation type. For example, the Conditional split transformation filters data based on a defined set of conditions. Similarly, the Microsoft Excel Input *source transformation* allows reading data from an Excel spreadsheet. The steps are connected by *data flow paths* that allow the data to flow from the source to the target. Figure below shows an example of a transformation in which data is read from an Excel spreadsheet (Microsoft Excel Input Source Transformation) and written to a text file (Flat file Destination Transformation).

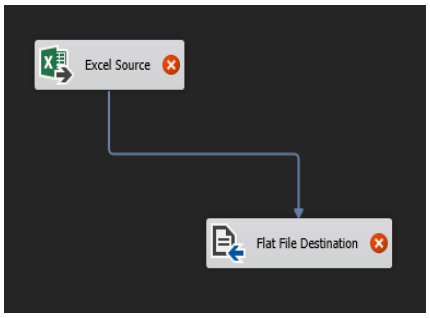


Figure 10. A transformation with Two Steps and One data flow path

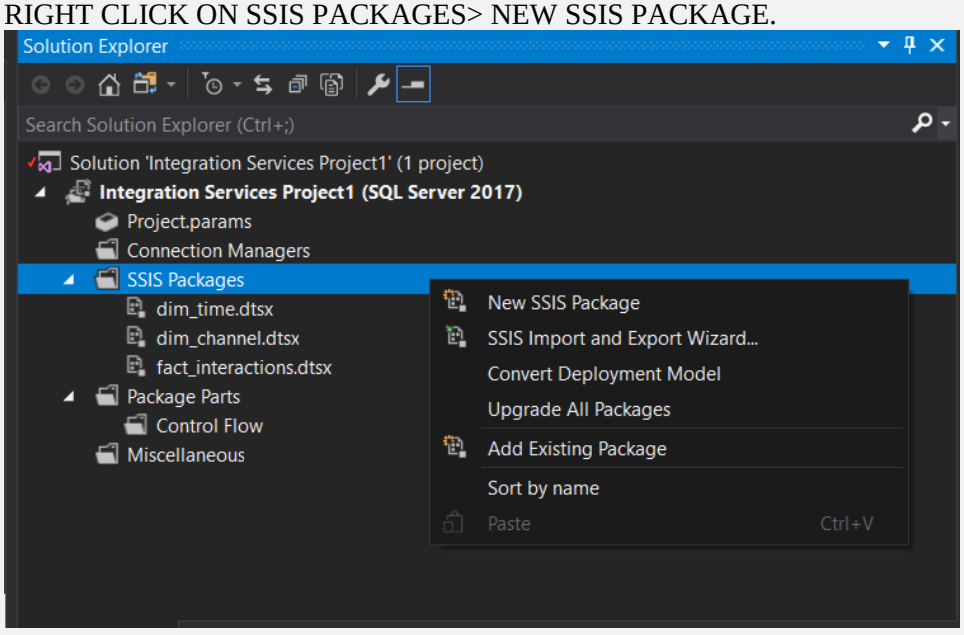
NOTE: IF ANYTIME DURING THE EXECUTION OF ANY OF DATAFLOW EXPLAINED IN THESE SECTIONS FAIL, YOU HAVE TO MAKE SURE THAT THE TARGET/DESTINATION TABLE IS TRUNCATED BEFORE YOU RUN THE DATA FLOW TASK AGAIN. THIS MAKES SURE THAT YOU GET THE CORRECT ROW COUNT ON THE DESTINATION TABLE.

TO TRUNCATE A TABLE, GO TO SSMS, SELECT YOUR DATABASE, SELECT THE APPROPRIATE TABLE AND RUN THE FOLLOWING QUERY –

Truncate table <the_table_name>
Select count(*) from <the_table_name>
Select * from <the_table_name>

IT IS RECOMMENDED THAT EACH OF THE DATA FLOW TASKS FOR EACH TABLE (i.e dim_time, dim_channel, fact_interactions) ARE DONE IN SEPARATE PACKAGE.DTSX FILES.

TO CREATE NEW PACKAGE.DTXS FILES, GO TO THE SOLUTION EXPLORER ON THE RIGHT> RIGHT CLICK ON SSIS PACKAGES> NEW SSIS PACKAGE.



To check the row count of your tables you can just run the last two lines of the below queries:

Select count(*) from <the_table_name>
Select * from <the_table_name>

2.3 Illustrative Example

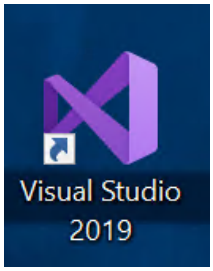
In this section we present step-by-step instructions for populating our data warehouse schema from the Hubspot data. The first two subsections (2.3.1 and 2.3.2) below illustrate populating dimension tables while subsection 2.3.3 populates a fact table. We provide detailed steps for populating the dimension table *dim_time* and leave the loading of the other dimension table *dim_channel* as an exercise.

In this example will use the following ETL transformations:

1. Read data from the Hubspot Excel spreadsheet.
2. Parse the PublishTime column from our source data and split it into three fields: TimeDay, TimeMonth and TimeYear to allow loading data into the dim_time table.
3. Populate the fact table fields Facebooklikes, Facebookcomments and Total_Interactions based on the channel and publish time information in the original dataset.
4. Validate the date and channel information against the corresponding values from the dimension tables.
5. Populating only the valid TimeNo values in the fact table based on the foreign key from the dim_time table.

2.3.1 Populating the time dimension table

In this section, we cover the loading of the data in the dim_time dimension table of our sample warehouse schema. The following steps will be done in **Visual Studio.** Open **Visual Studio** from the desktop short cut or by selecting from the Start menu.



8. You can create a new project by navigating to **File** > **New** > **Project**. Then search and select **Integration Services Project** and click on **Next.**

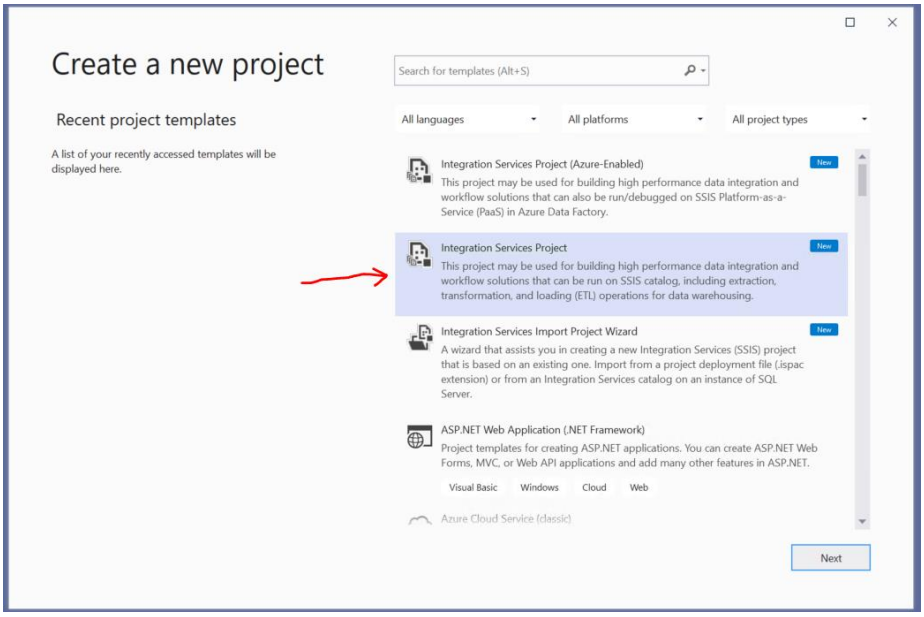


Figure 11: Open new project

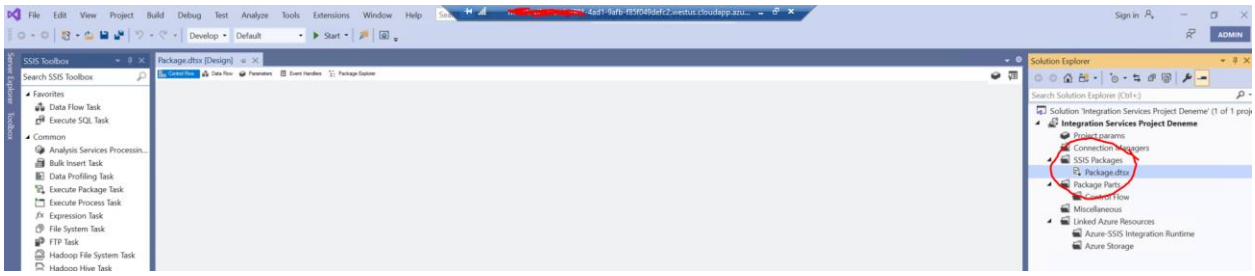
Type a name for your project and click on **Create.**

Step by step package creation here ->

The complete transformation for dim_*time table consists of four steps: (a) connecting to the source data, (b) filtering null values, (c) splitting the publishtime field into three fields: TimeDay, TimeMonth and TimeYear, and (d) loading the data into the MSSQL table dim_time.*

Note: Please download the sample excel file "3_1_hubspot-social-media-export-01-08-16" to your remote desktop (Data Science Virtual Machine) by signing into Canvas on a browser before proceeding. Open Chrome or Firefox and login to Canvas to download the excel file. (Same steps you would follow on your personal computer)

9. Go to IDE and open package.dtsx from SSIS Packages on the solution explorer on the right.



This opens the **SSIS Designer** as shown below.

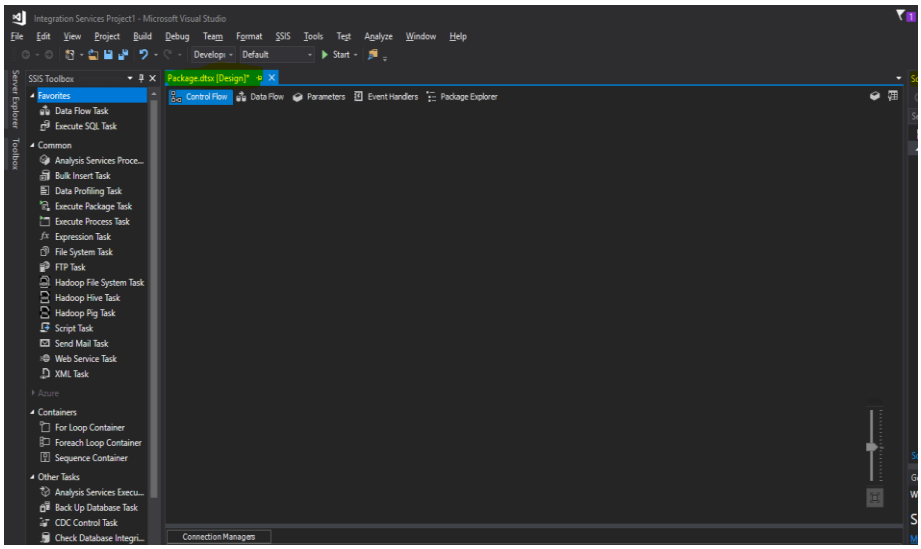


Figure 12: SSIS Package.dtsx designer window

10. Drag the **Data flow task** from the **SSIS toolbox** on the left to the SSIS Designer on the **control Flow** tab. A **control flow** defines a workflow of tasks to be executed, often a particular order.

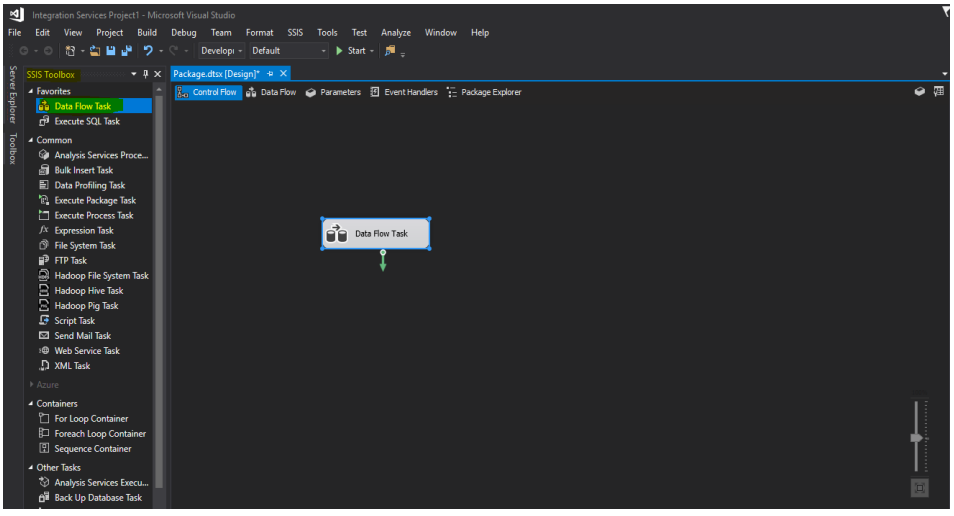


Figure 13: Create a Data flow task

11. Double click on the Data Flow Task. This open the Data flow tab.

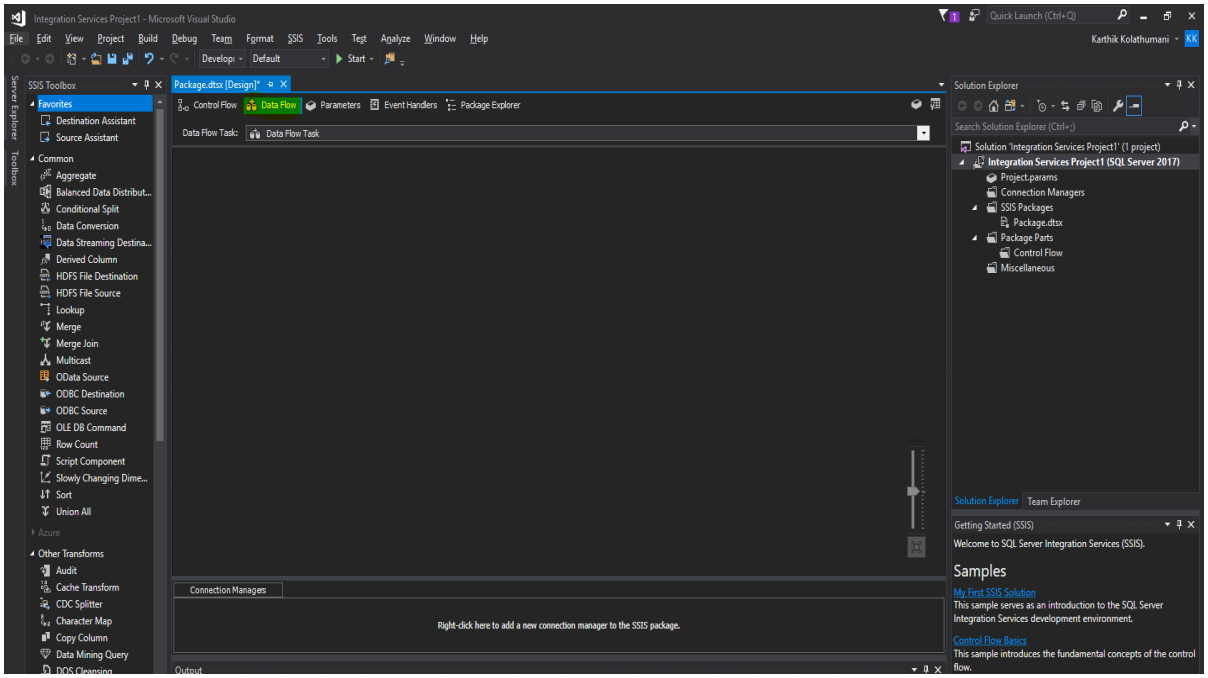


Figure 14: Data flow task tab

12. Scroll down on the SSIS toolbox to find **Excel Source** under the **Other Sources** tab.

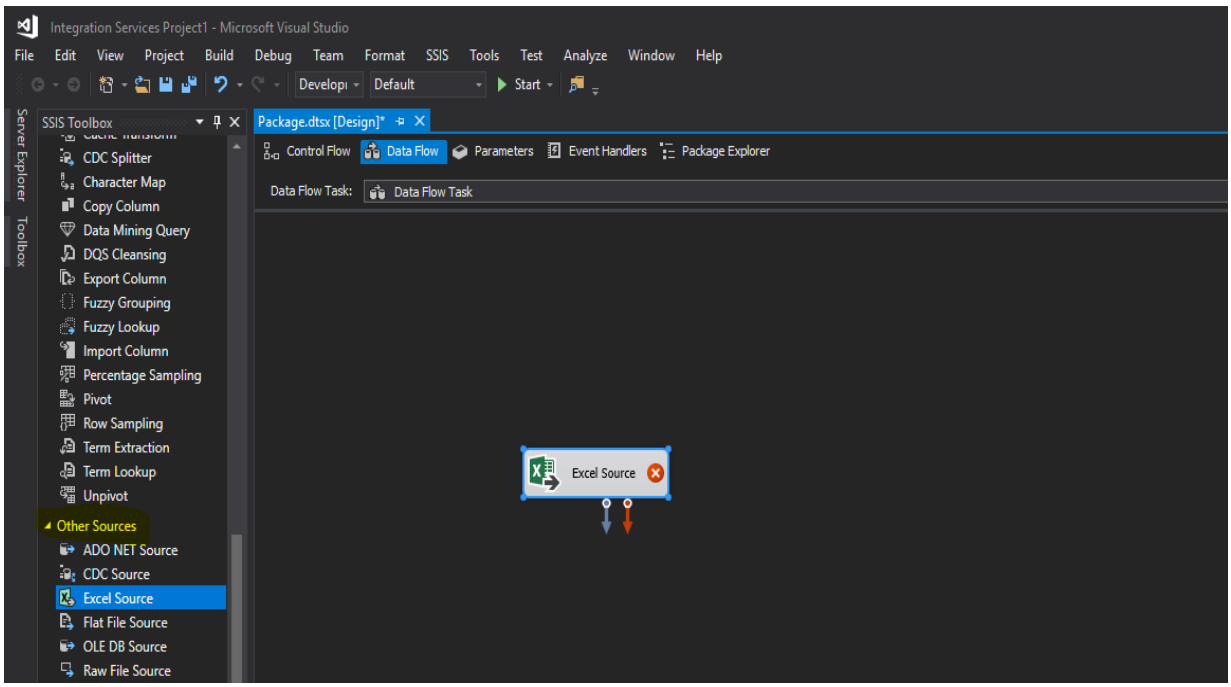


Figure 15: Creating an Excel Source Transformation

13. Double click on it to configure it. Select *New > browse to the path where the spreadsheet is located on your machine > Check "First row has column names" checkbox > OK.*

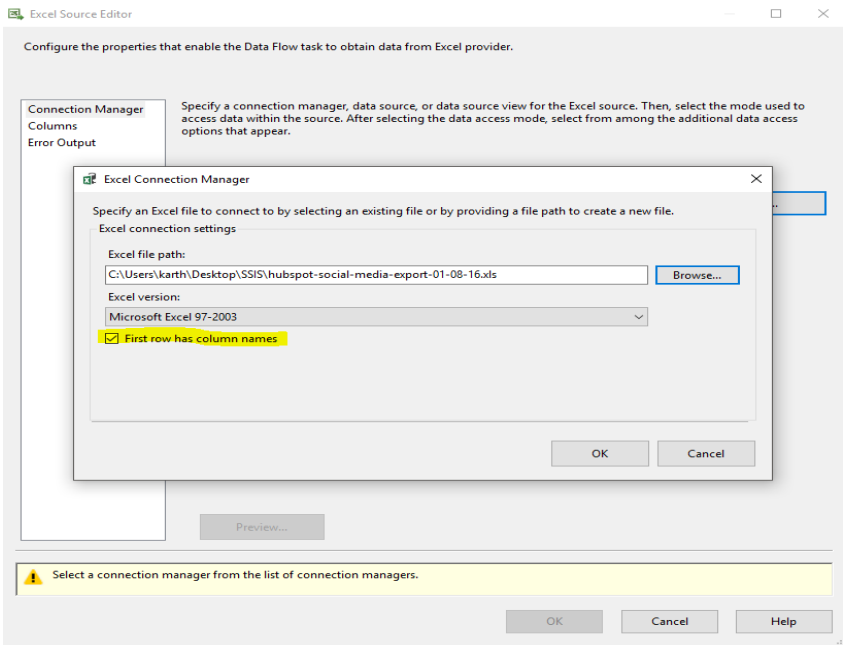


Figure 16: Configuring the Excel Source Transformation

14. Select "Name of the excel sheet" to the appropriate one.

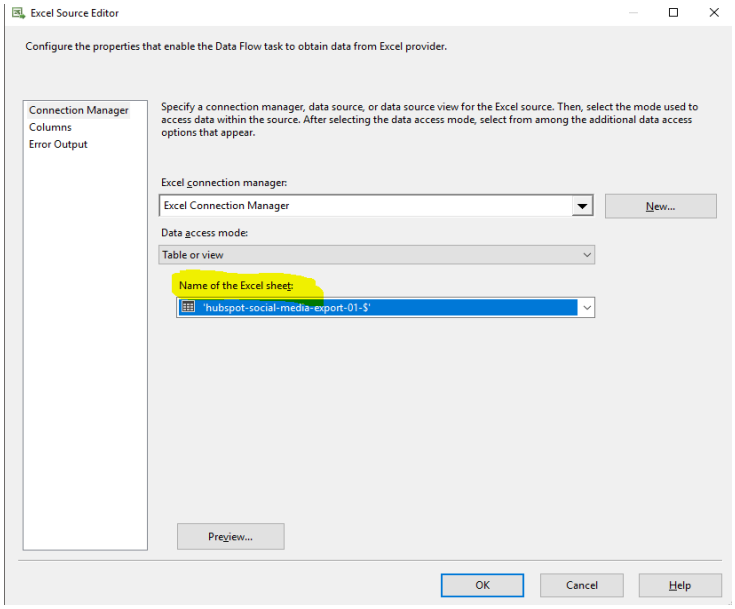


Figure 17: Configuring the Excel Source Transformation

15. If you now select "Columns" on the left panel, you should be able to see the below figure. NOTE: Make sure all the columns are selected.

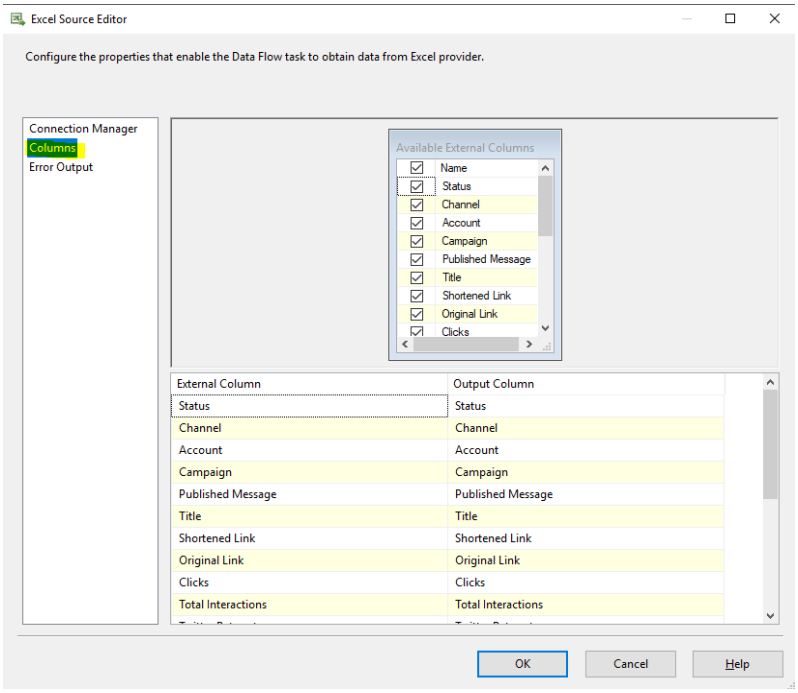


Figure 18: Configuring the Excel Source Transformation

Hit OK for the window to close.

To make sure that your excel source is configured properly, you should not see an error icon over the Excel source transformation.

Now our Source is ready to transfer its data. Our destination is going to be an ADO NET Destination where we will load the dim_time table. These tables were created using the hubspot.sql file mentioned in Step 9.

16. Before loading the dim_time table we have to make sure the Campaign column in our Excel file does not contain any NULL values. To do this drag the **Conditional Split** Transformation from the SSIS toolbox to the Data Flow designer.

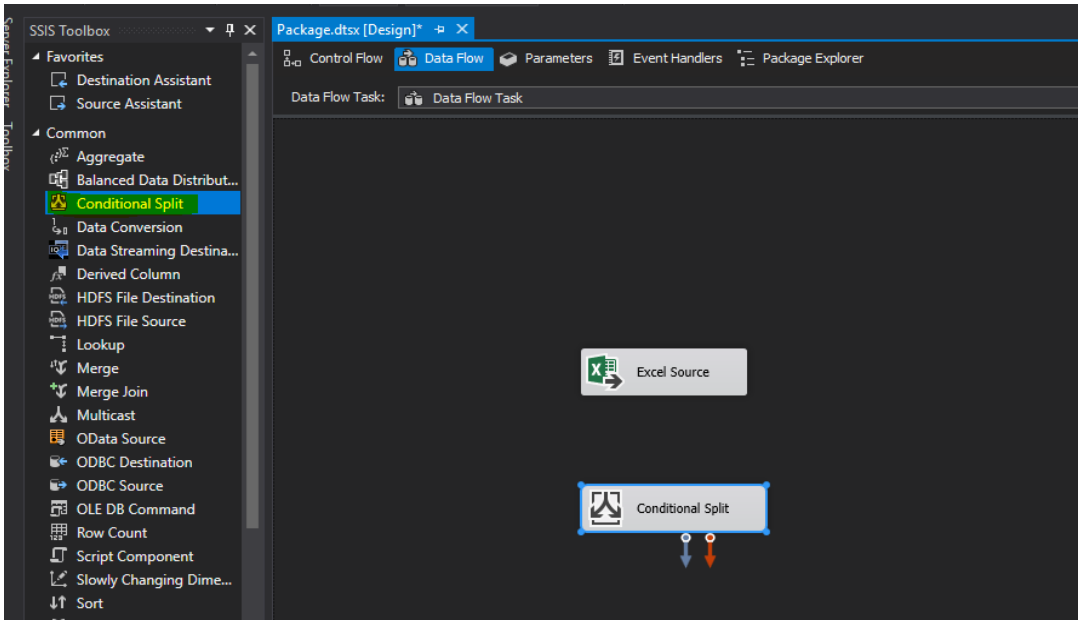


Figure 19: Configuring the Conditional Split Transformation

The below steps will illustrate how you can configure the conditional split transformation to filter NULL values from the Campaign column.

17. Click on the Excel Source transformation and drag the **Blue** Data flow Path onto the Conditional Split transformation. (Tip: The **RED** Data Flow paths are meant for error logging purpose. It can be useful when a transformation fails)

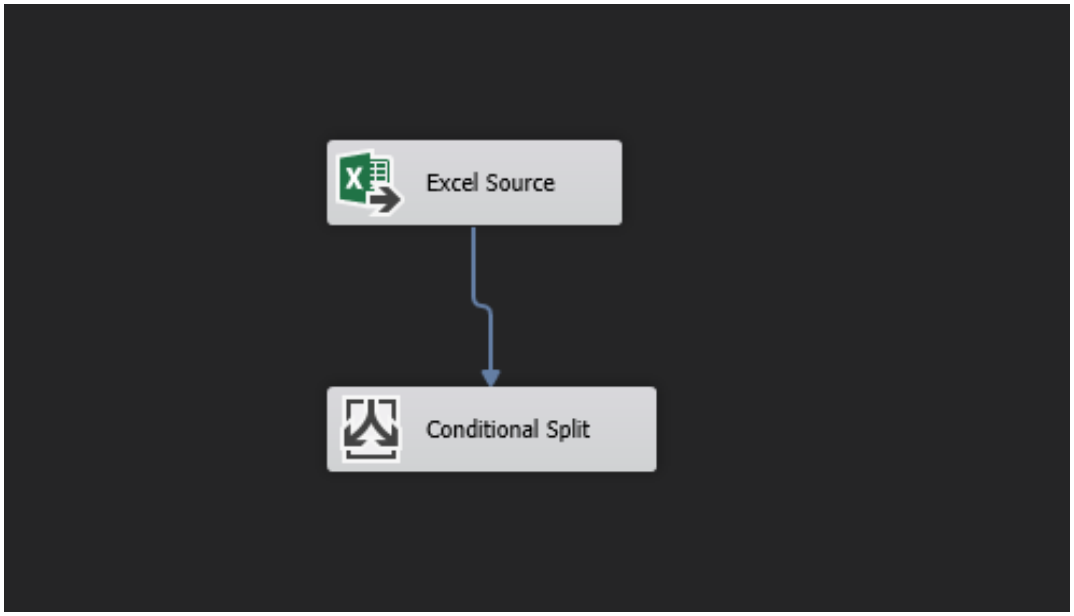


Figure 20: Configuring the Conditional Split Transformation

18. Double click on Conditional Split > Expand "NULL Functions" on the top right panel.

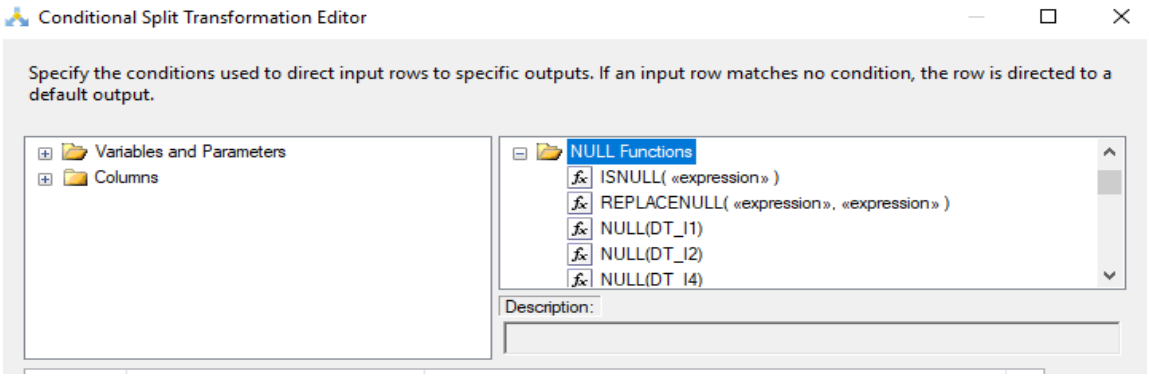


Figure 21: Configuring the Conditional Split Transformation Editor

19. Drag and drop the **ISNULL(<>)** function onto the condition filed as shown below.

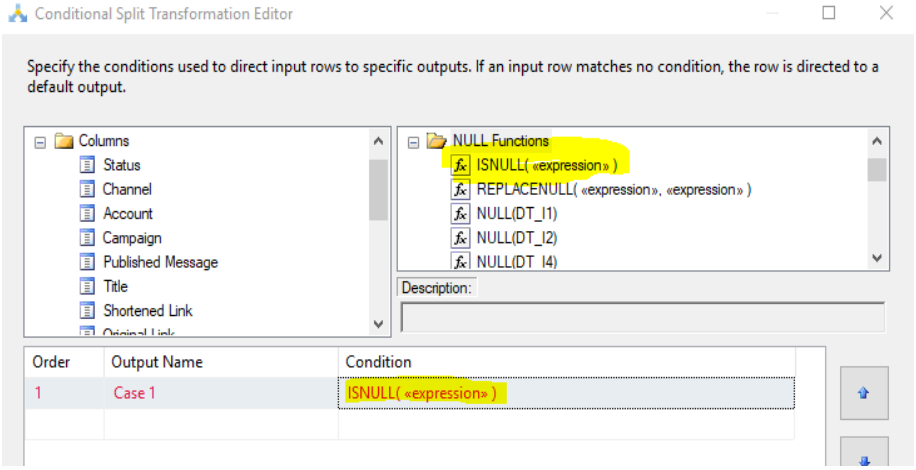


Figure 22: Configuring the Conditional Split Transformation Editor

20. Drag and drop the *campaign* column from the left panel onto the <> placeholder. Since we need to get only NOT NULL values, prefix the ISNULL condition with "!" so that it now becomes as what is shown below.

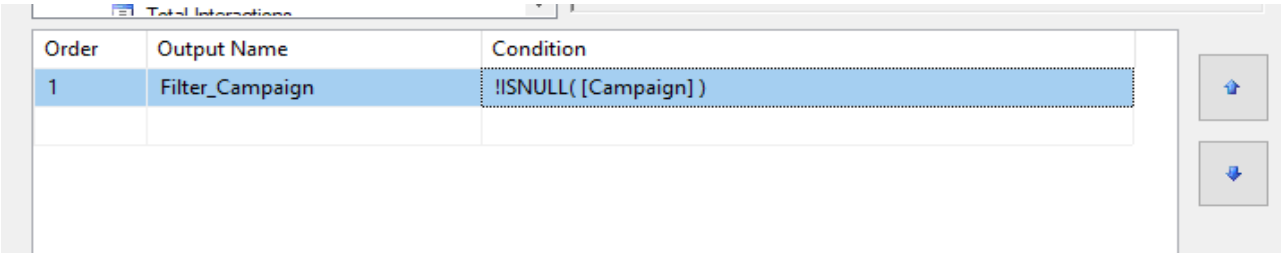
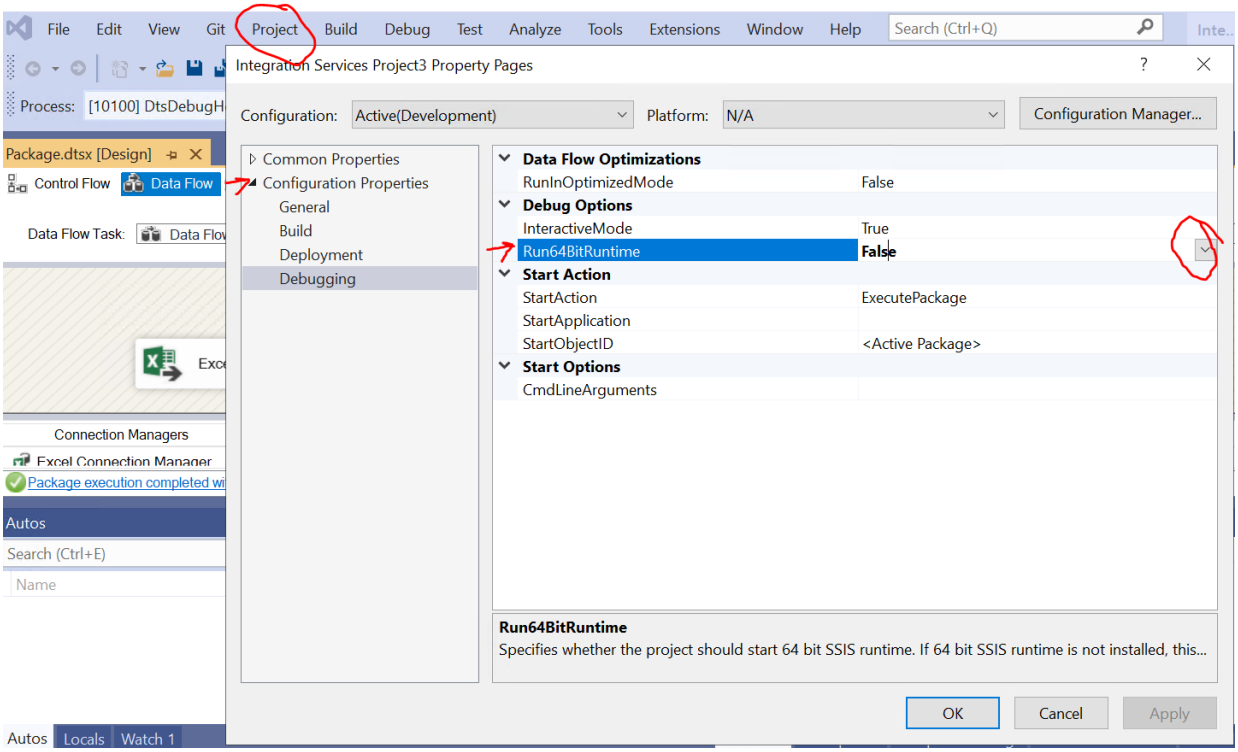


Figure 23: Configuring the Conditional Split Transformation Editor

In the above step, we can also rename the "Output Name" to something meaningful instead of "Case 1". For this tutorial I have changed it to "**Filter_Campaign**". The **Default output name** can remain untouched or renamed depending upon your naming preferences. The Default Output Name outputs all the values which doesn't meet the condition we have specified.

Hit OK.

21. Now we will configure our debugging settings. Go to **Project** -> **Properties** -> and set **run64BitRuntime** to **False** and hit **OK**



22. We have successfully configured our conditional split to filter out the null values. You can now start the data flow task by clicking on **Start**.

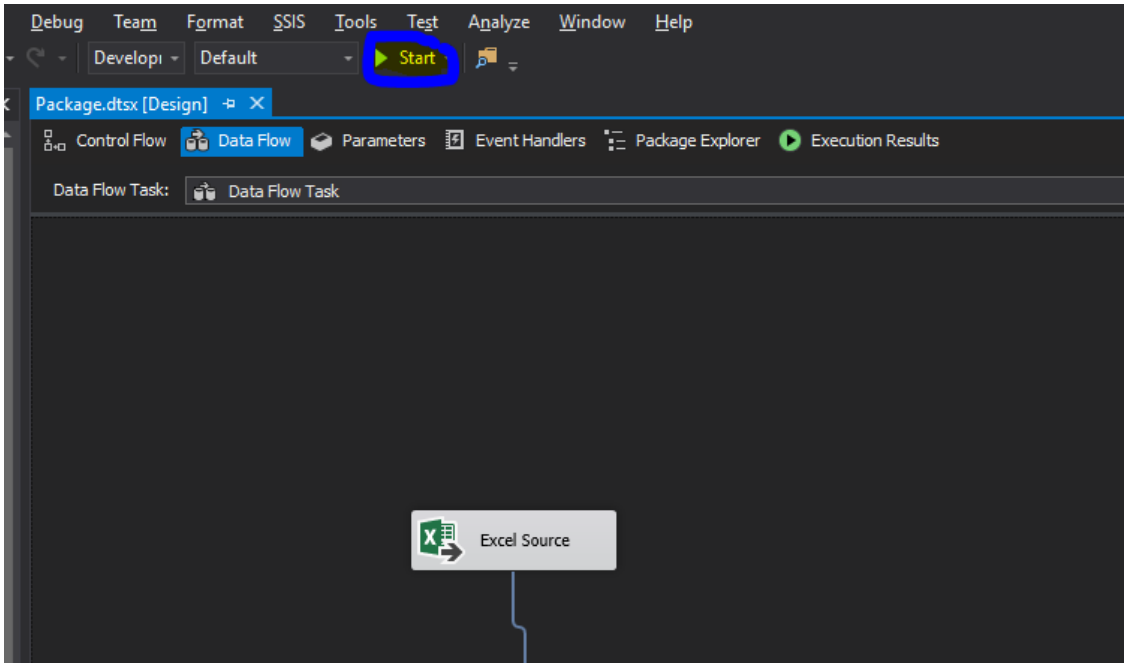
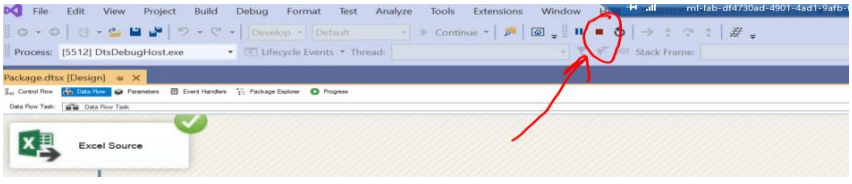


Figure 24: Running the data flow task

23. This will start the current data flow task. When the task completes you will be able to see the below screen.

Figure 25: Running the data flow task

At the bottom of the screen you will be able to see that the task was completed successfully. You will also be able to see the number of rows that were fed into our conditional split transformation. **Stop** the task by clicking on stop icon (red square) and go back to the designer.



Below steps illustrate how you can take the output of the conditional split transformation to make three new derived columns Day, Month and Year for our dim_time table.

Drag and drop **Derived Column** on the designer. (Tip: Derived Column Transformation are usually used to create/replace a column with a new logic)

24. Create the data flow path from Conditional split transformation to the Derived Column. You will be able to see the **Input Output Selection** window pop open.

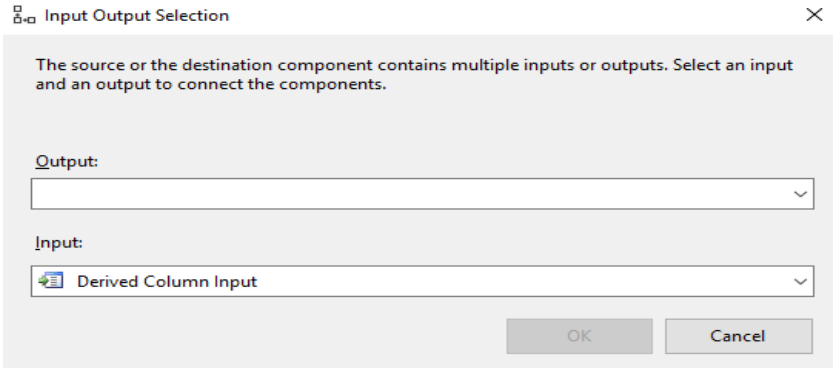


Figure 26: creating the Data flow path between conditional split transformation and Derived Column transformation

25. Select the output as **Filter_Campaign** *and hit OK.* **Filter_Campaign** is the output port from the conditional split transformation which will give us the NOT NULL values for the Campaign column.

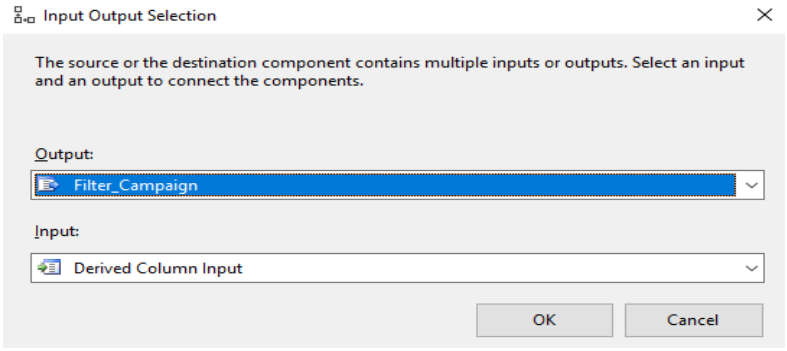


Figure 27: creating the Data flow path between conditional split transformation and Derived Column transformation

26. Double click on the Derived Column and expand the "Date/Time Functions". Drag the "DAY", "Month" and "YEAR" functions one by one onto the expressions field to create three new Derived Columns as shown below.

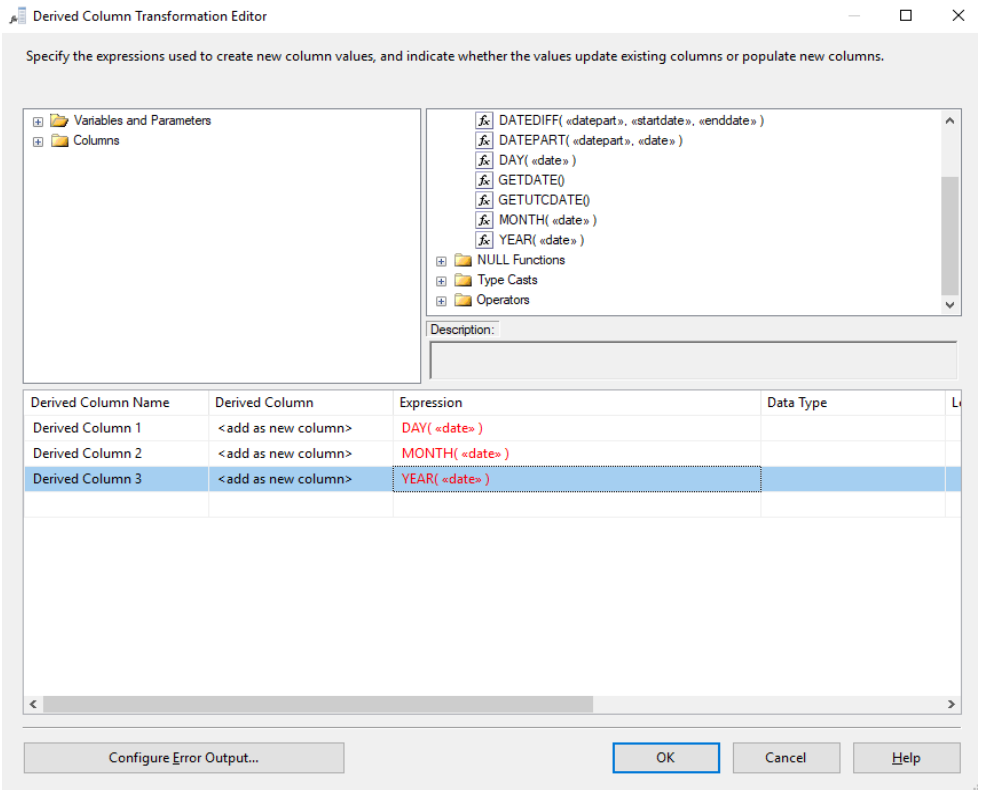


Figure 28: Configure the derived column transformation

27. Replace each of the <> placeholder with the **Publish Time** column from the left panel and rename the columns to **Day**, **Month** and **Year** respectively.

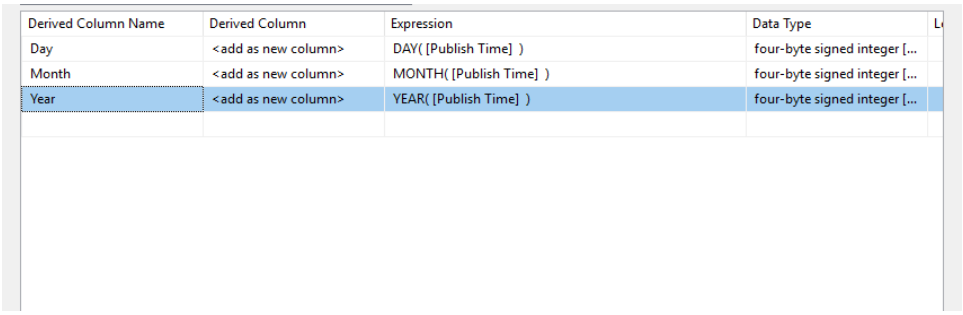


Figure 29: Specifying the Expression for derived column

Hit OK.

We have now successfully split the **Publish Time** into three columns. Our next step involves aggregating the records w.r.t **Day**, **Month** and **Year** to avoid duplicate records.

The below steps illustrate how to use an **Aggregate** Transformation to group duplicate **Day, Month** and **Year**. **Aggregate** Transformation can be used to perform the aggregate operations on data.

Drag and drop the **Aggregate** Transformation in the designer and create a data flow path from **Derived Column** Transformation to **Aggregate** Transformation.

Double click the **Aggregate** Transformation to configure it.

28. Select **Day**, **Month** and **Year** from the available columns and set "Operation" for each one of them to “Group By” as shown below.

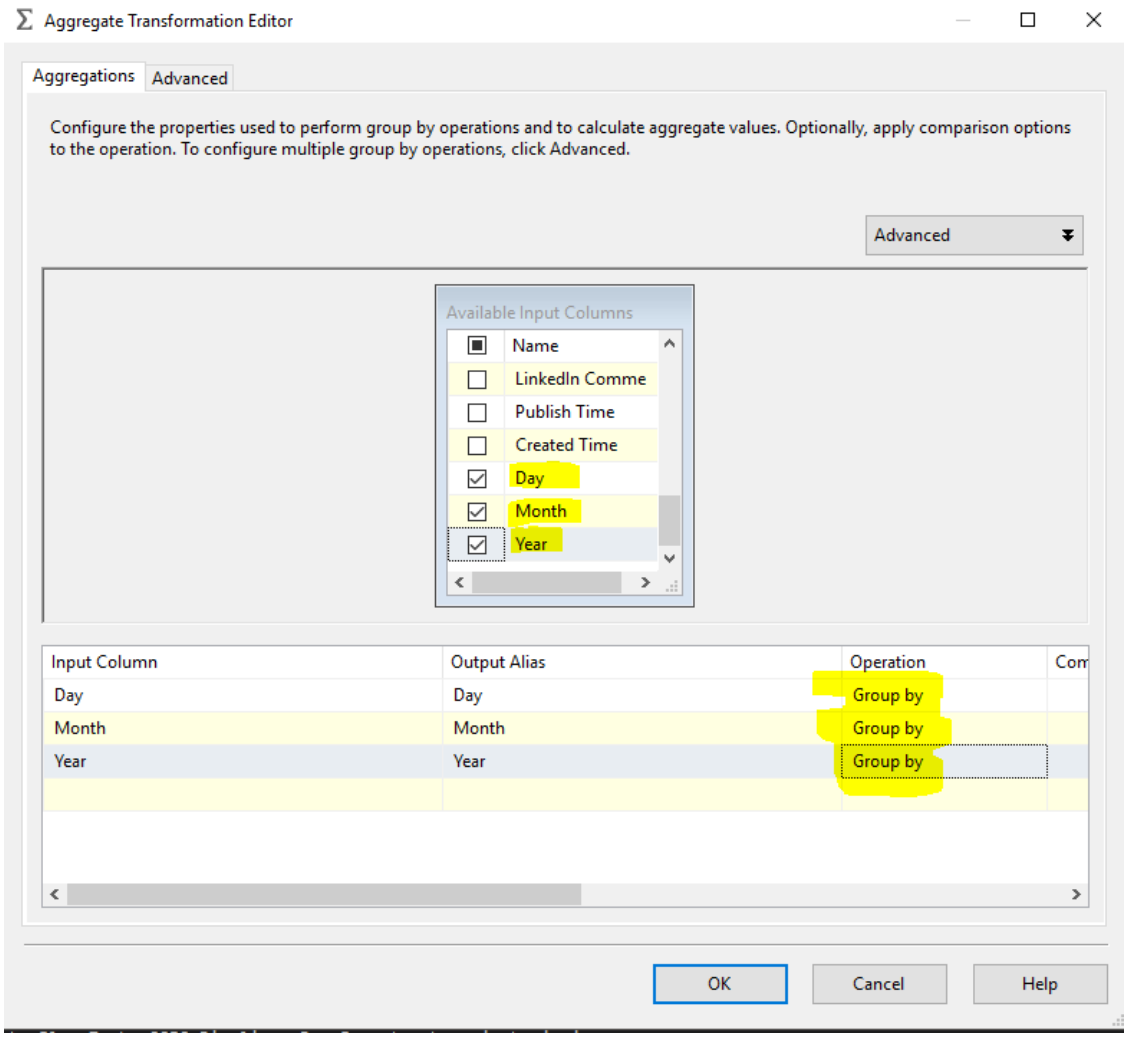


Figure 30: Configuring the aggregate transformation

Hit OK.

Our next step involves writing the records to the dim_time table. The below steps illustrate how to populate the dim_time table.

Drag and drop the **ADO NET Destination** transformation into the designer. ADO NET destination provides access to the SQL Server databases where you have already created the appropriate tables.

29. Create the data flow path form Aggregate transformation onto the ADO NET Destination transformation.

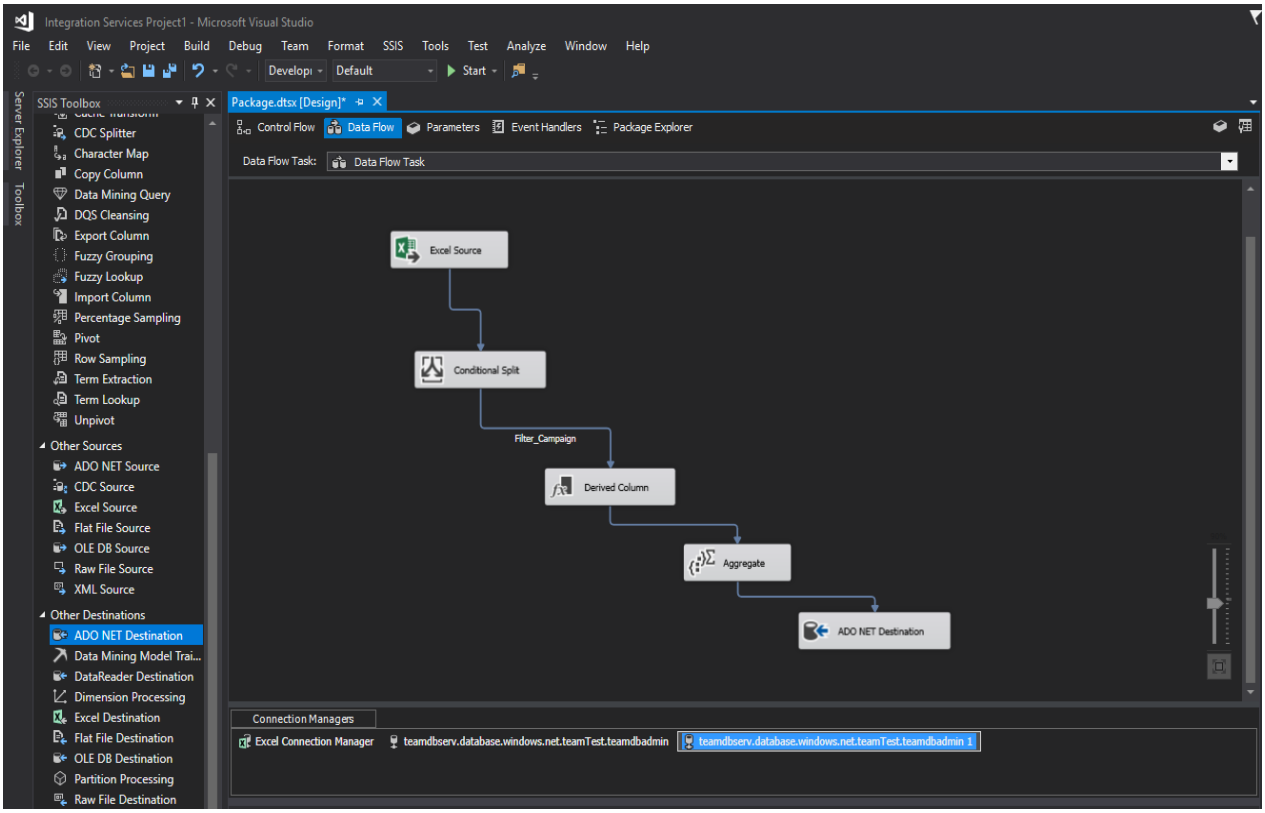


Figure 31: Adding the ADO NET Destination

30. Double click on the ADO NET Destination transformation to configure it. Select "New" next to the Connection manager.

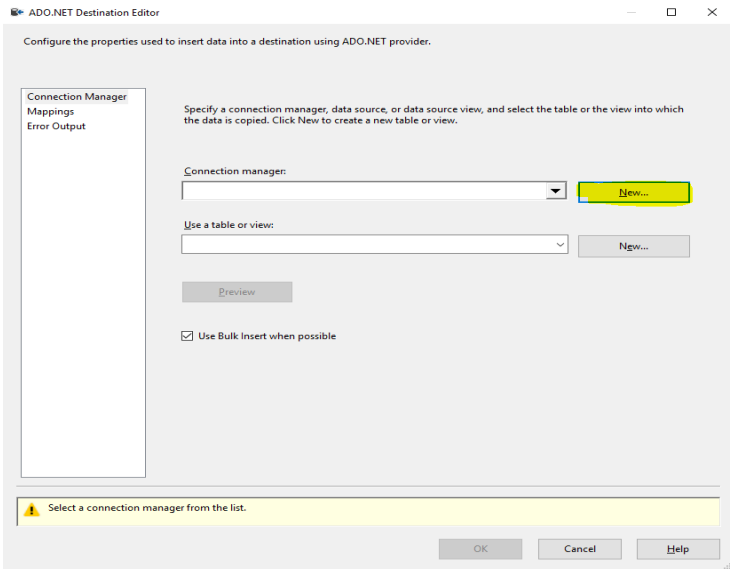
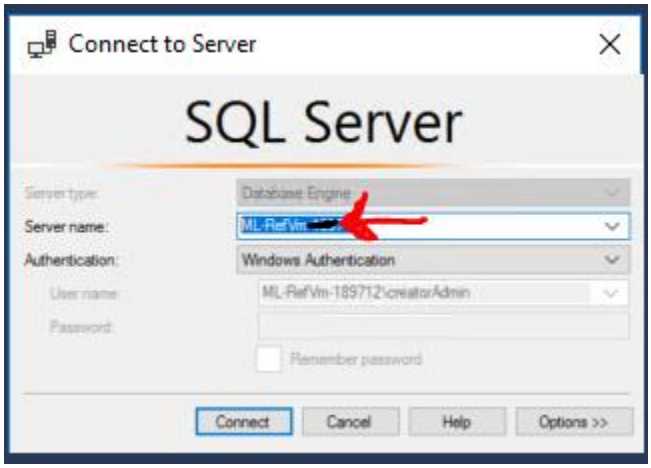
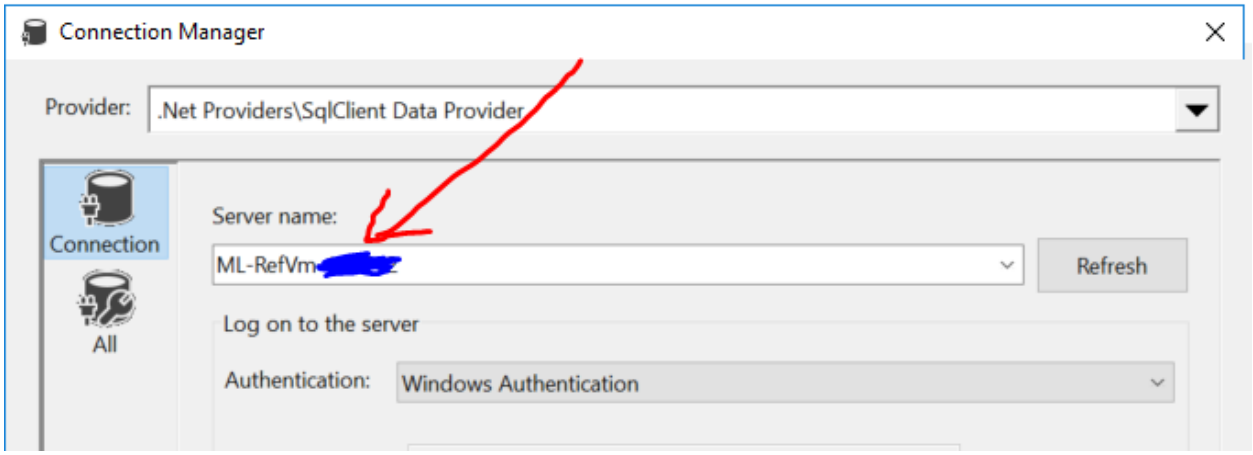


Figure 32: Configuring the ADO NET Destination

31. Select New and Type the server name.
You can find the server name installed on your Virtual Machine by going to SQL Server Management Studio (SSMS) (the software where you created your own database). The server name appears on the **Connect to Server** pane showing up when you start SSMS. It should start with "ML".





32. On Connection Manager pane, Select Windows Authentication > Select your database (which you created in SSMS) > hit "Test Connection".

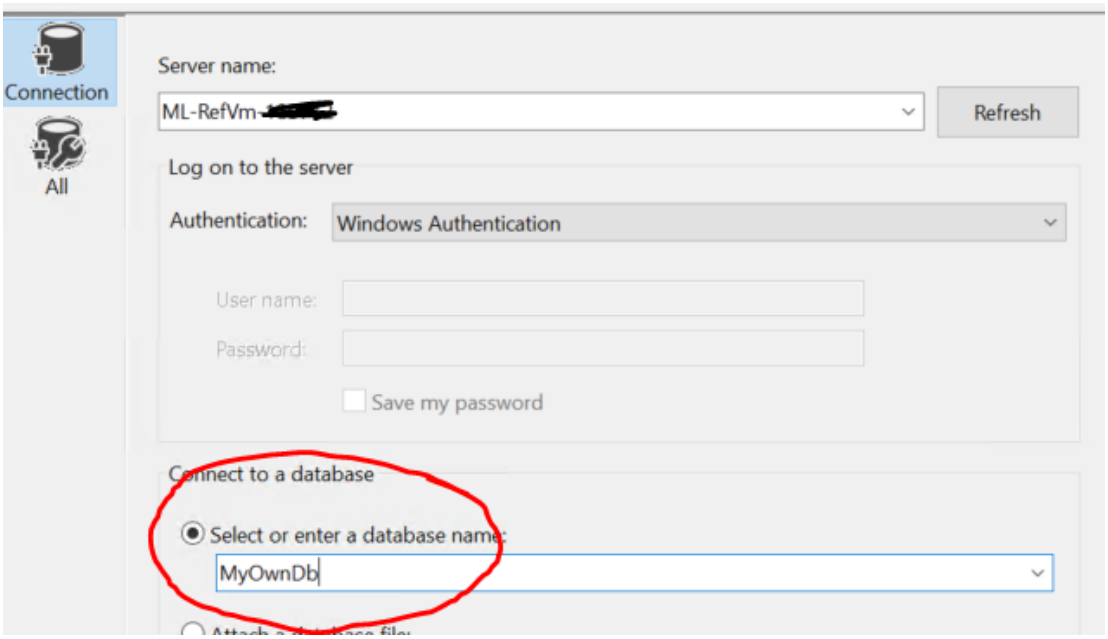


Figure 33: Configuring the ADO NET Destination

33. If all the information is correct, your connection should succeed.

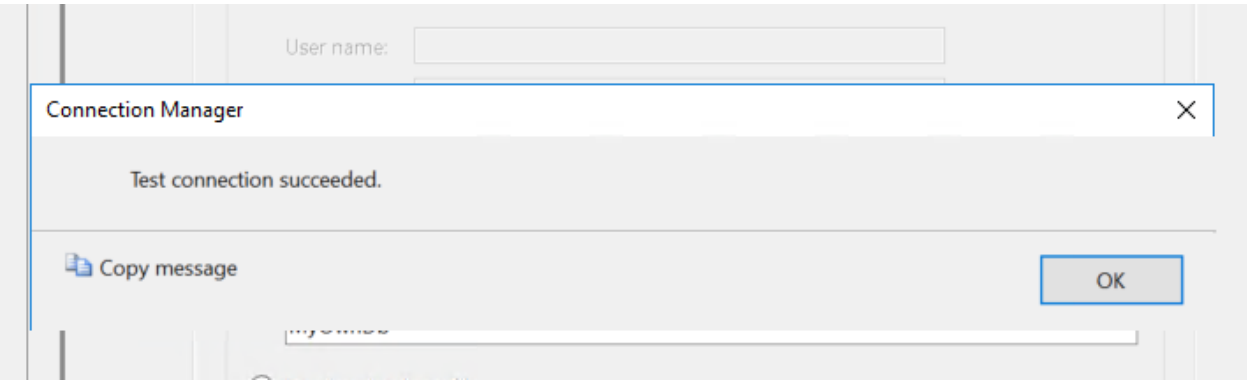


Figure 34: Configuring the ADO NET Destination

Hit OK

34. Head back to the configuration setttings > Select dim_time as the table as shown.

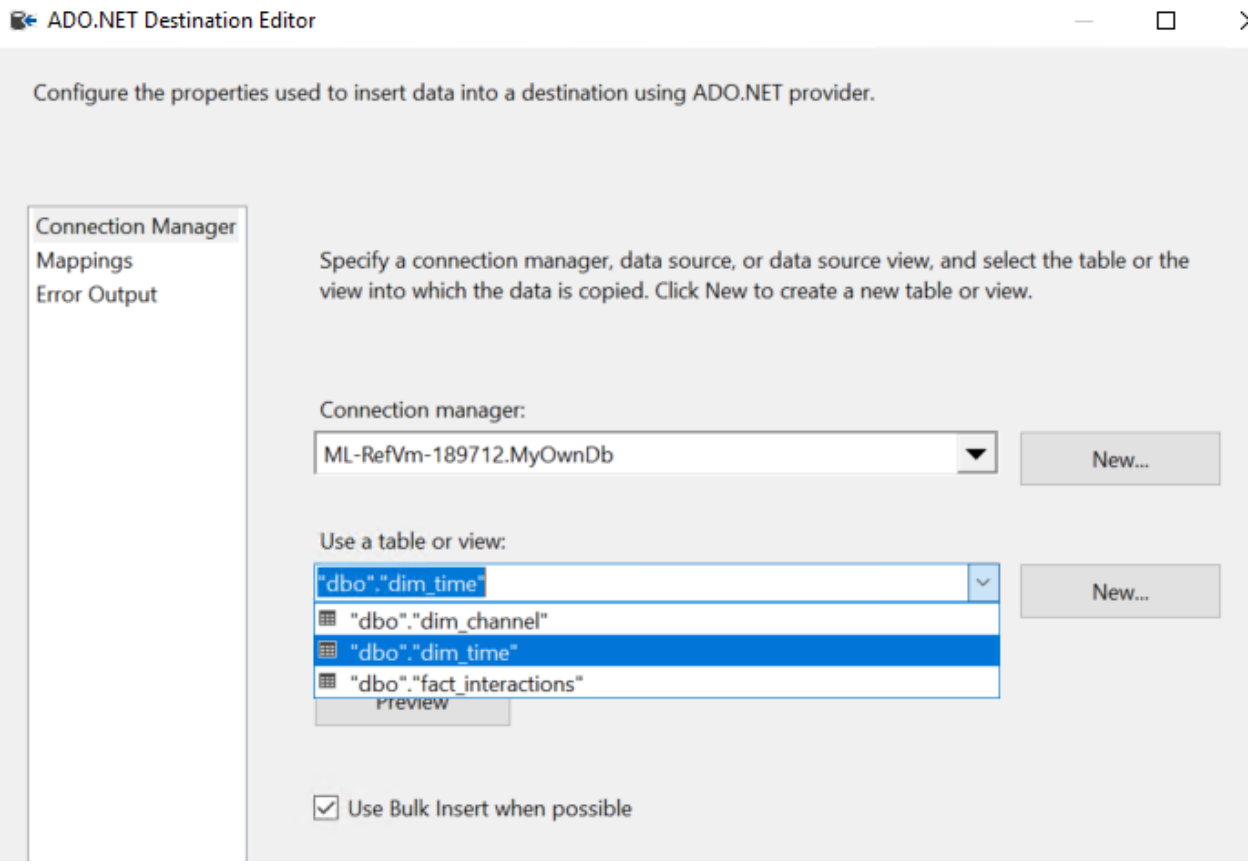


Figure 35: Configuring the ADO NET Destination

35. Select Mappings on the left panel and make sure all the mappings are correct as shown below.

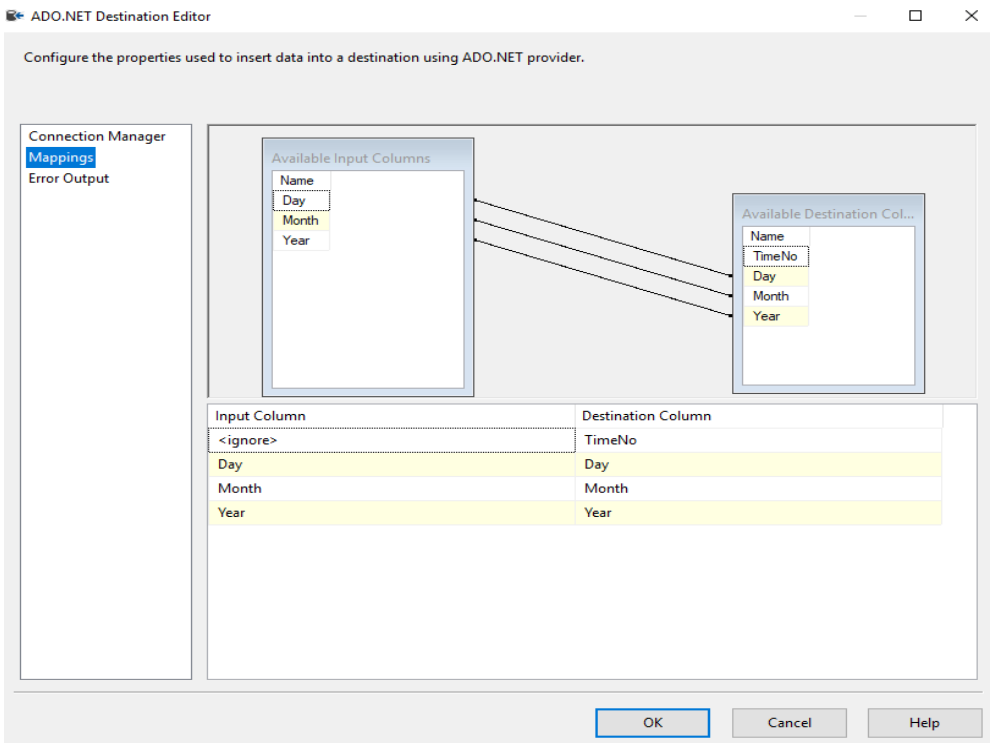
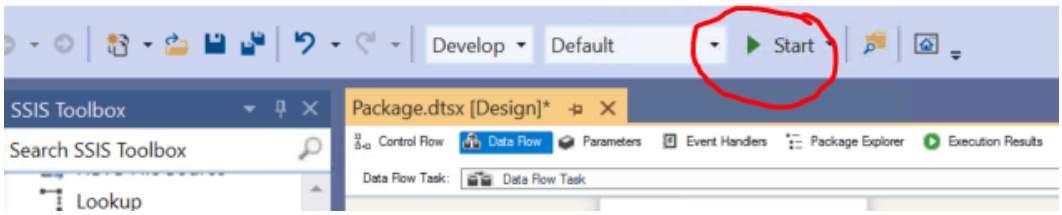


Figure 36: Configuring the ADO NET Destination

Hit OK

36. You are now ready to load your first table. Hit Start to run your task!



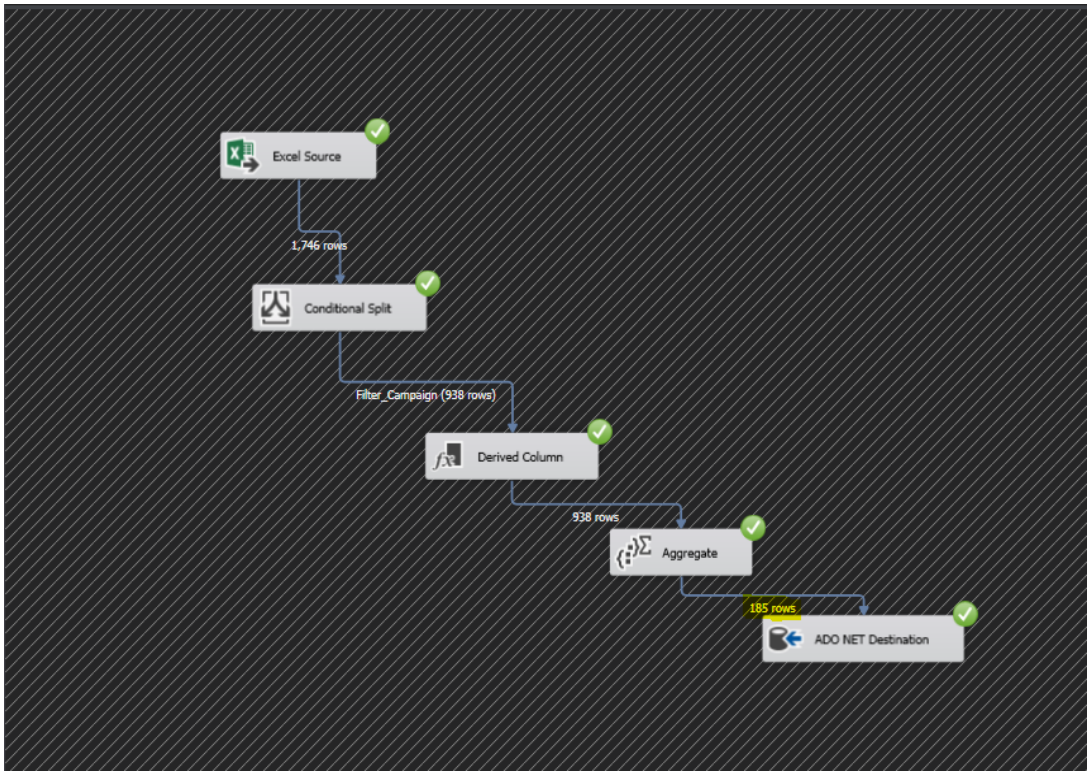
37. 

Figure 37: Final data flow task for dim_time

Your task should load 185 rows of data into the dim_time table.

This completes our discussion of how to load data into the dimension table dim_time based on requirements discussed at the beginning of the Section 2.3.

EXERCISE

The approach to load data into the dimension table dim_channel follows similar steps. Setup the dataflow dim_channel table.

38. The below mapping might help you to populate the dim_channel table

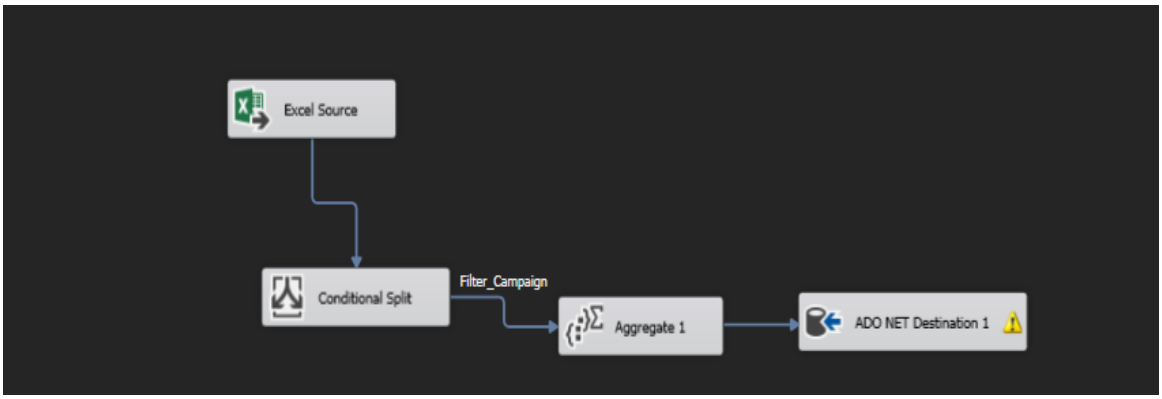


Figure 38: Data flow path for dim_channel

TIP: SSIS warns "records getting dropped" if the source and destination field length for a column are different. You will have to make sure that your destination has the same data type and same or greater field length for every column mapped from the source.

39. Your task should load a total of 3 rows to your dim_channel table.

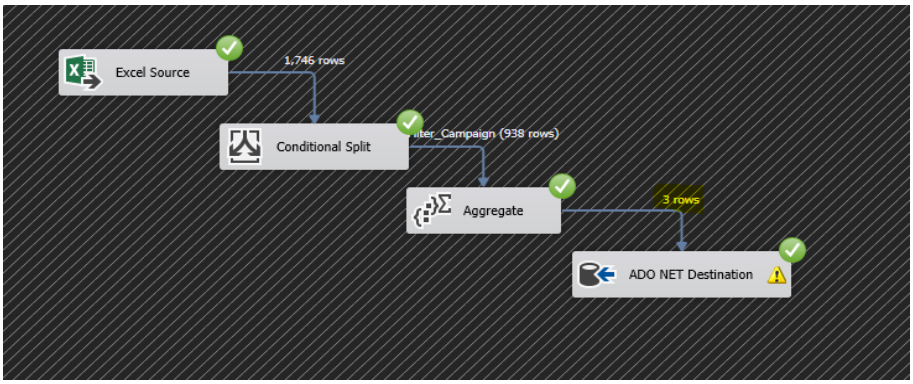


Figure 39: dim_channel output

2.3.2 Populating the interactions fact table

Having populated the dimension tables, dim*time and dim*channel, we now illustrate the steps for populating the fact table fact_interactions.

The fact table contains measures – Facebook Likes, Comments, Total Interactions and two foreign keys (ChannelID and TimeNo) from the two dimension tables. The first figure below shows a few rows from the hubspot data. After completing the ETL steps below, the fact table will look like second figure below. It shows a snapshot of the fact table data based on rows. Note that for the columns facebook*likes, facebook*comments, and total interactions the values are copied from the original dataset. But the ChannelId and TimeNo are not present in the original Hubspot dataset.

40. The below table show the Hubspot data for publishdatetime = January 7, 2016.

Channel	Total Interactions	Twitter Retweets	Twitter Replies	Twitter Favorites	Facebook Likes	Facebook Comments	LinkedIn Likes	LinkedIn Comments	Publish DateTime
Twitter	6	3	0	3	0	0	0	0	1/7/2016 17:00
Twitter	12	6	0	6	0	0	0	0	1/7/2016 16:47
LinkedInCompanyPage	23	0	0	0	0	0	23	0	1/7/2016 16:47
FacebookPage	1	0	0	0	1	0	0	0	1/7/2016 16:47
Twitter	13	6	0	7	0	0	0	0	1/7/2016 14:02
FacebookPage	29	0	0	0	29	0	0	0	1/7/2016 14:02
LinkedInCompanyPage	6	0	0	0	0	0	6	0	1/7/2016 14:02
FacebookPage	301	0	0	0	293	8	0	0	1/7/2016 11:02
Twitter	7	3	0	4	0	0	0	0	1/7/2016 11:02

Figure 40: Fact table data

41. The below figure shows the desired output in the fact table (fact_interactions).

InteractionId	TotalInteractions	Facebook_likes	Facebook_comments	ChannelId	TimeNo
6	1	1.00	0.00	2	1
7	29	29.00	0.00	2	1
8	301	293.00	8.00	2	1
251	23	0.00	0.00	3	1
252	6	0.00	0.00	3	1
359	6	0.00	0.00	1	1
360	12	0.00	0.00	1	1
361	13	0.00	0.00	1	1
362	7	0.00	0.00	1	1

Figure 41: Desired output for fact table

The values of ChannelId and TimeNo involve looking up the corresponding columns from the dimension tables dim*channel and dim*time, respectively. This ensures that referential integrity constraint is enforced. This section looks up and loads values from the columns of the dimension tables to enforce referential integrity. We will look up primary key columns from the dimension tables to ensure that the loaded data values for Channel and PublishTime from the original data source do not contain invalid data. The columns Channel and PublishTime from the original Hubspot dataset are matched against the dim*channel and dim*time dimensions tables.

42. The below shows the complete ETL transformation for loading the data in the fact table (fact_interactions). We discuss it in three parts.

    1) Load the foreign key TimeNo in fact*interactions based on the primary key column TimeNo from the dimension table dim*time.
    2) Load the foreign key ChannelID in fact*interactions based on the primary key column ChannelId from the dimension table dim*channel.

3) Load the data into the MSSQL table (fact_interactions).

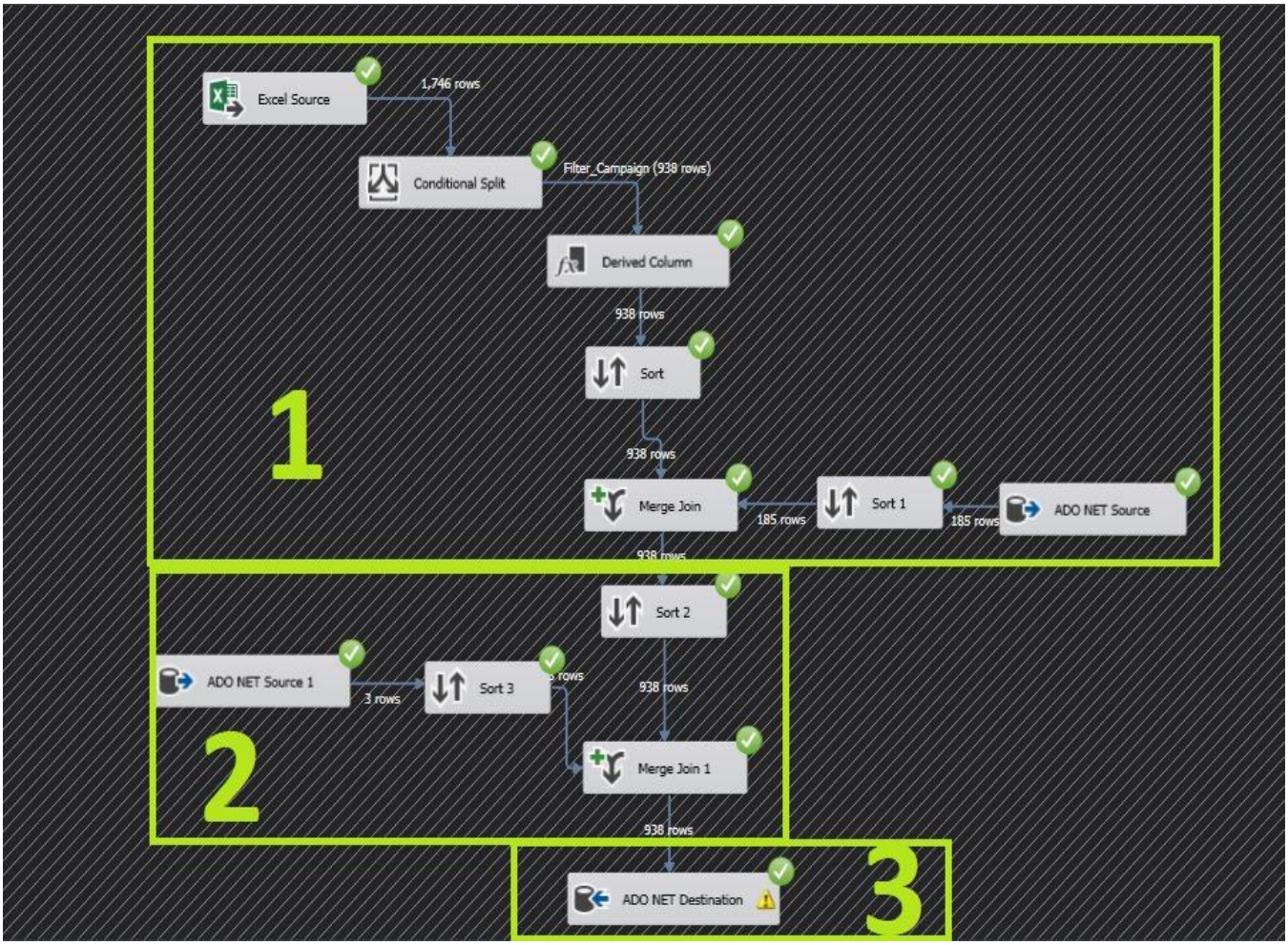


Figure 42: Data flow mapping for fact_interactions

Loading the TimeNo column in the fact table

This section addresses the first subtask (Box 1) from the figure 42

Populating the column TimeNo in the fact_interactions table involves accessing the dim_time table and fetching the TimeNo column based on the three fields – TimeDay, TimeMonth and TimeYear. The PublishTime column from the original Hubspot data was split into three parts – Day, Month and Year. This is accomplished using the steps as discussed in Section 2.3.

- Pull from Hubspot Excel Source
- Filter Rows with Conditional to remove null
- Select Values and Split Values to derive new columns Day, Month, Year fields

Box 1 also captures these steps. Each of the three fields – Day, Month and Year are then compared to the corresponding fields (TimeDay, TimeMonth and TimeYear) of the dim_time table. To access the MSSQL table dim_time in SSIS, follow the steps below.In the similar

manner you can configure ADO NET Source transformation and select dim_time as "Name of the table or the view".

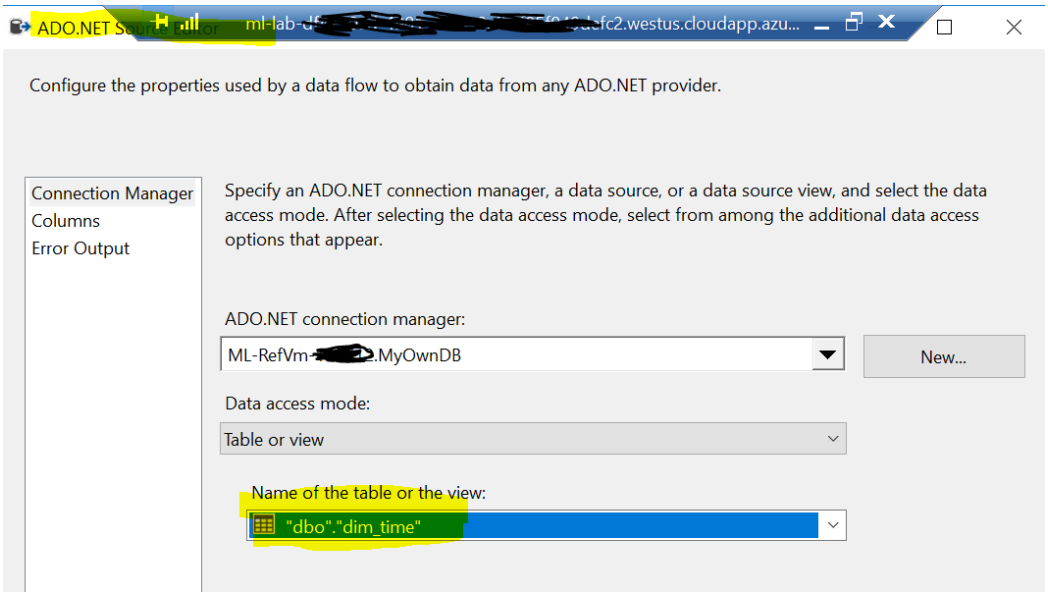


Figure 43: Configuring ADO NET Source Transformation

Drag the **Sort Transformation** from the SSIS toolbox. The *Sort transformation sorts* input data *in* ascending or descending order and copies the *sorted* data to the *transformation* output.

Create a data flow path from the ADO NET Source to the Sort Transformation.

44. Double click the Sort transformation to configure it. Select on Day, Month and Year from the available input columns. The sort order (the sort order column below) should be the same as shown below

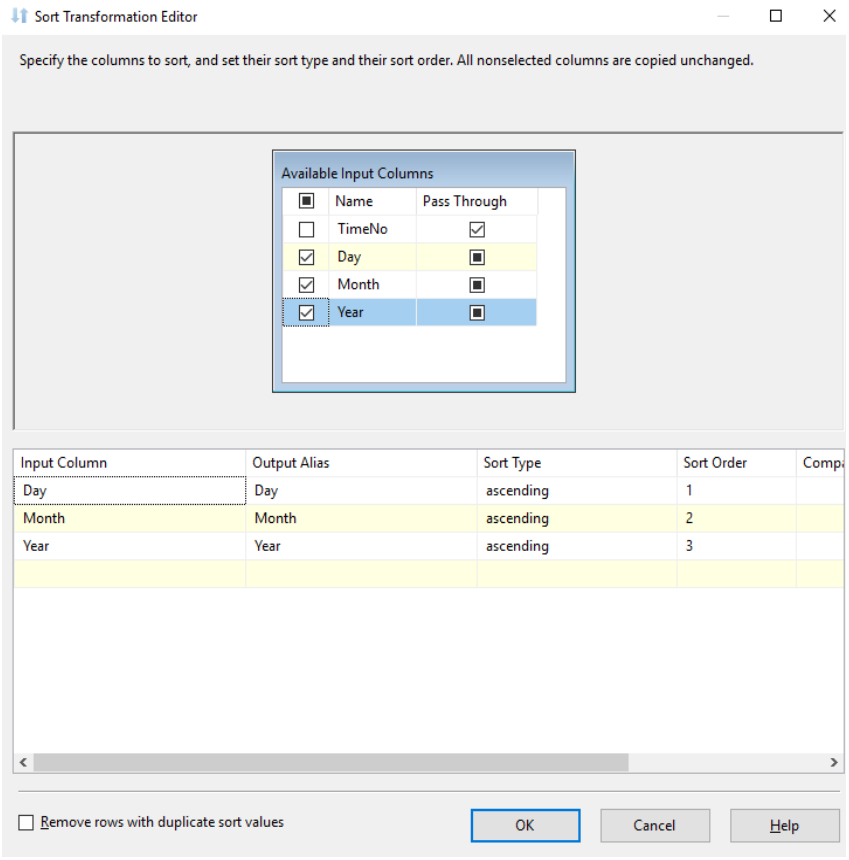


Figure 44: Configuring Sort Transformation

Since we need the TimeNo column in our destination table (fact_interactions), make sure it's checked for "Pass through".

Hit OK.

This will make sure that the rows we need for the lookup from our dim_time table are sorted. Sorted lookups are more efficient than unsorted lookups.

Now we are done with this part of the transformation time to load our fact_Interactions table-

45. Drag another sort transformation from the SSIS toolbox and connect it to the Derived Column data flow path as shown in the above figure. Repeat with the same sort order explained above and hit OK.

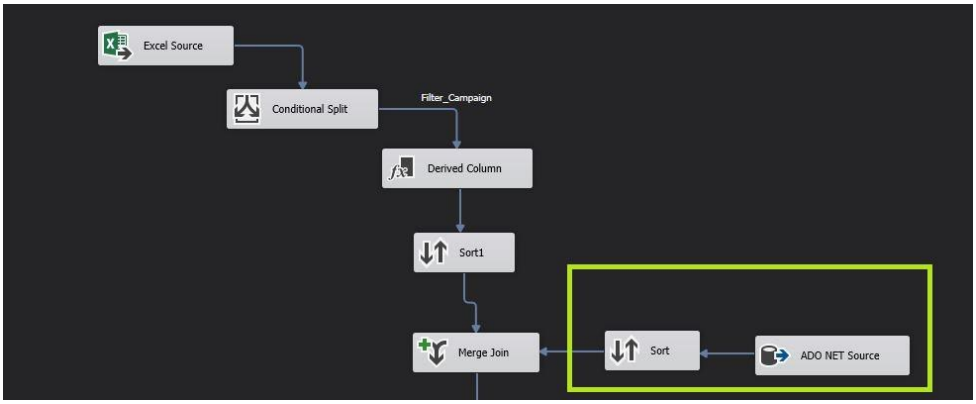


Figure 45: Drag Sort Transform

46. We are only left to merge both transformations to successfully configure our lookup. For this, we will use the **Merge Join Transformation**. The Merge Join transformation provides an output that is generated by joining two sorted datasets using a FULL, LEFT, or INNER join. Merge joins are one of the most widely used transformation. You can read more about them HERE.

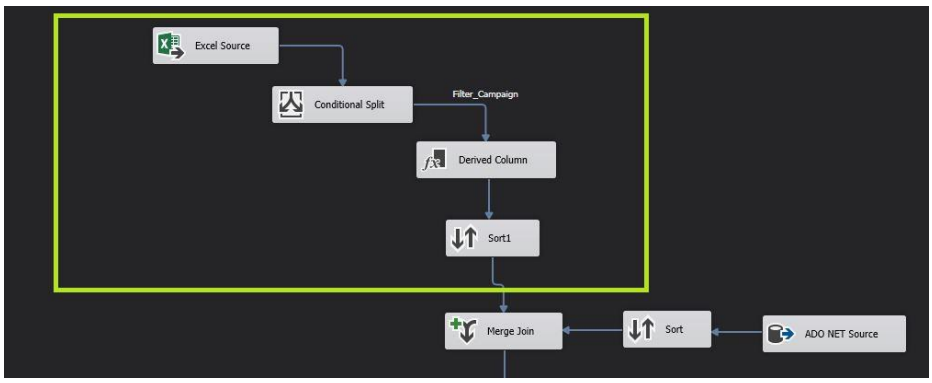


Figure 46: Using Merge Join to connect data sources

Let's configure the Merge Join Transformation.

47. Drag the Merge Join Transformation and connect it to the **Sort1**. Configure it to the following setting -

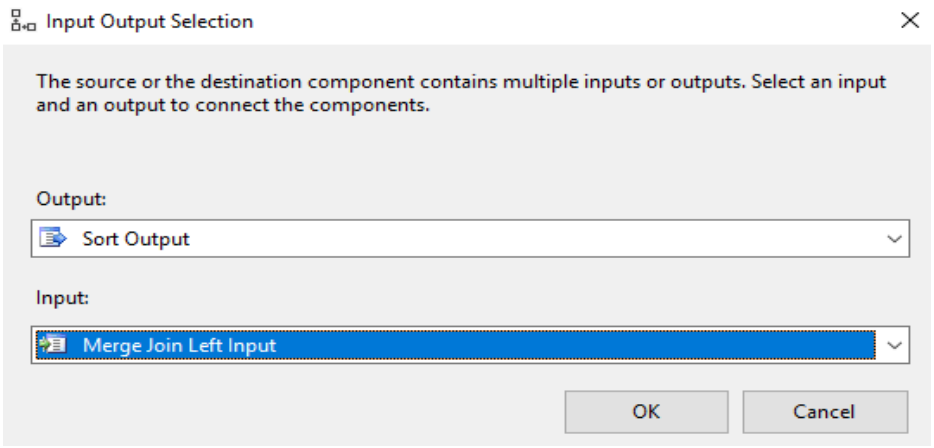


Figure 47: Configure Merge Join

We are trying to do a LEFT JOIN between the rows coming the excel sheet with the rows coming from the ADO NET Source (or dim_time).

Connect **Sort** transformation with the Merge join transformation.

Double click the merge transformation to configure it. The transformation automatically identifies the **Join Keys**. Join keys are the keys based on which we will be merging the two data sources. In our case the Join Keys are Day, Month and Year which should be select on both Sort and Sort1 window.

48. From this transformation onwards, we will only need to check the checkboxes next to *facebook likes, facebook comments, Total interactions and channel* from the Sort1 transformation and *TimeNo* from the Sort transformation. Uncheck all other columns apart from the one mentioned above. By default, the merge join should be configured to perform an **INNER JOIN**.

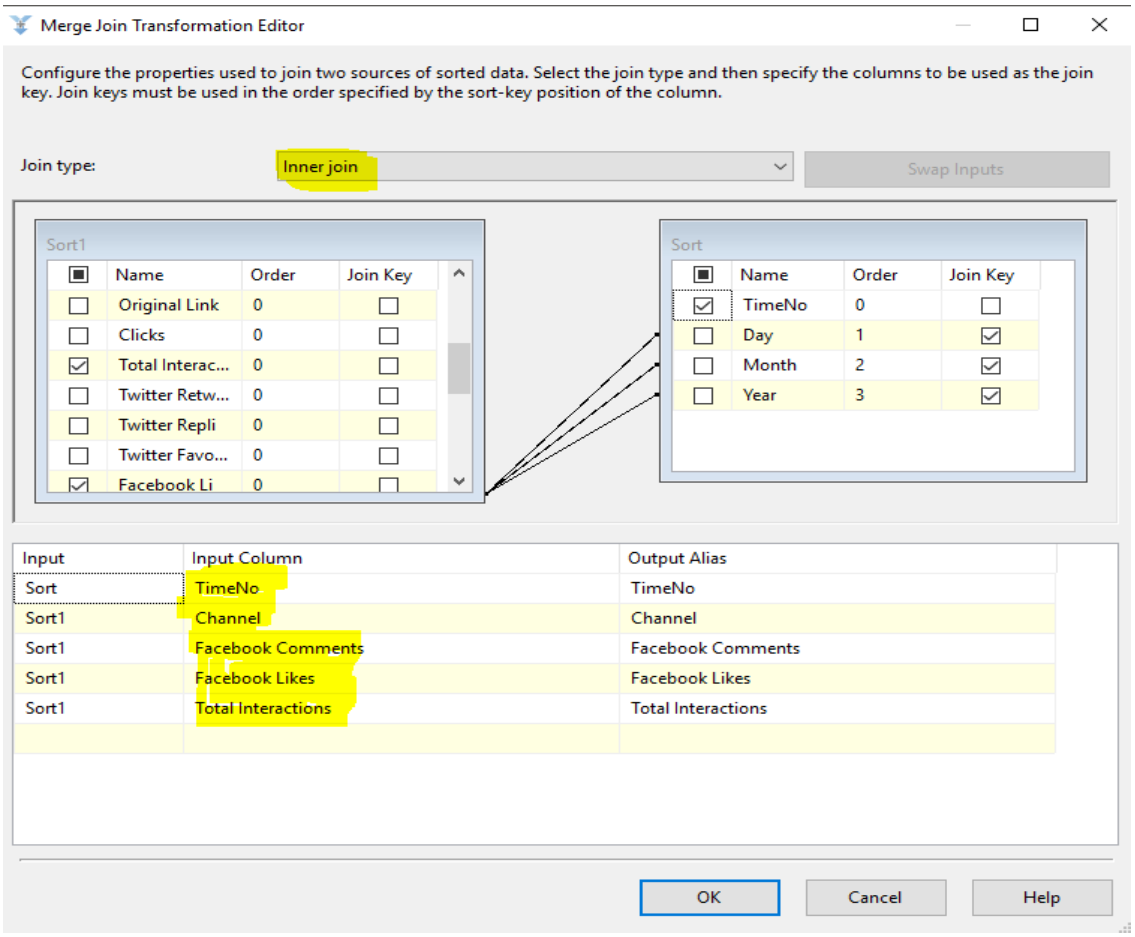


Figure 48: Merge Join Transformation Editor

Hit OK.

49. This completes our lookup for the Day, Month and Year field from the dim_time table. At this point, your transformation should look like this -

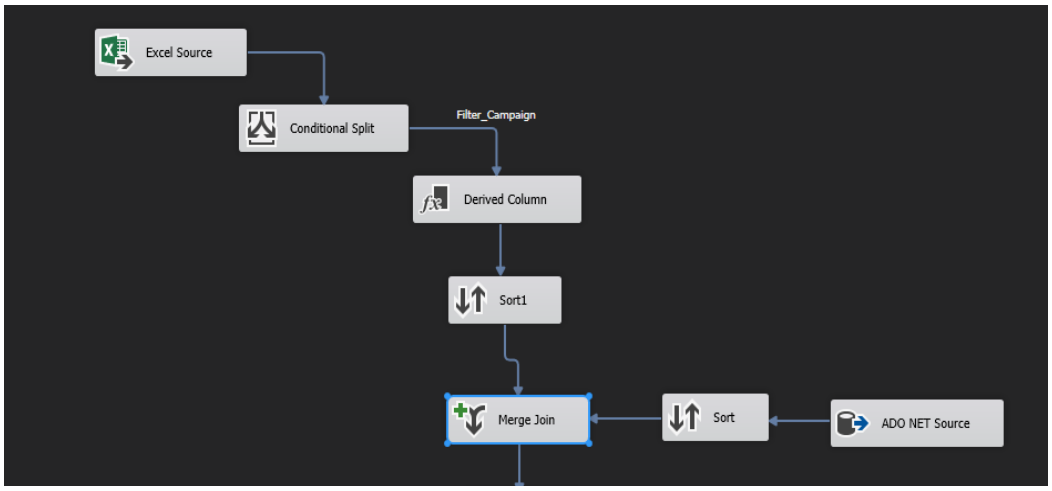


Figure 49: Data flow mapping for dim_time lookup

50. Now we need to do a lookup based on the Channel. We will be working on the following part of the transformation now -

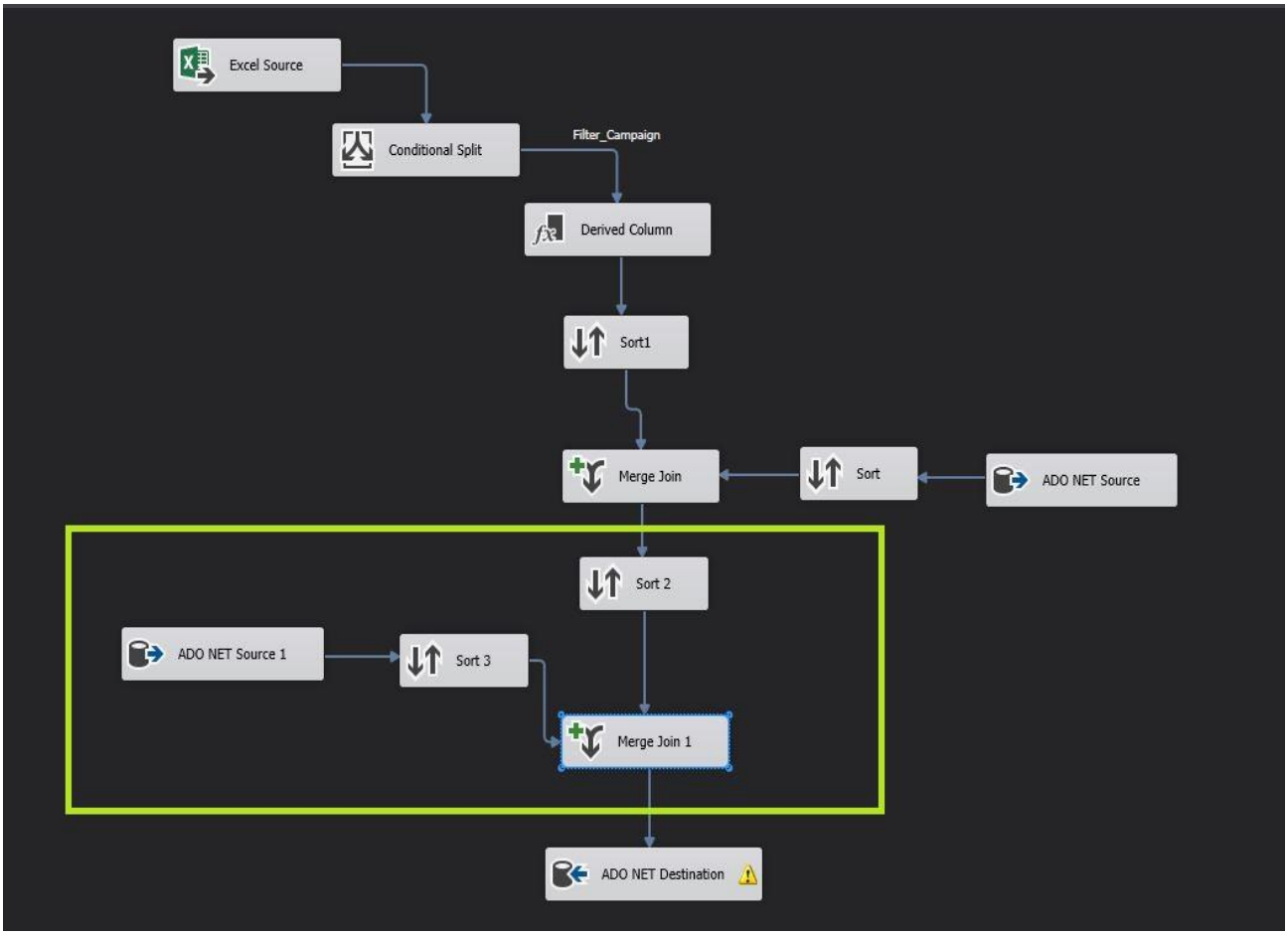


Figure 50: Data flow mapping for dim_channel lookup

Drag a Sort transformation (Sort2) and create a data flow from the Merge join configured previously. Configure the Sort based on Channel(ascending) by clicking the check box next to Channel.

Drag another ADO NET Source(ADO NET Source 1) and configure it to use the dim_channel table as explained in the previous steps. Drag the Sort transformation (Sort 3), create a data flow path from the ADO NET Source(ADO NET Source 1) to Sort3 and configure a sort based on Channel column(ascending).

51. Drag a Merge Join transformation. Create a data flow path from the Sort2 transformation to this Merge Join transformation. And configure it to the following configuration -

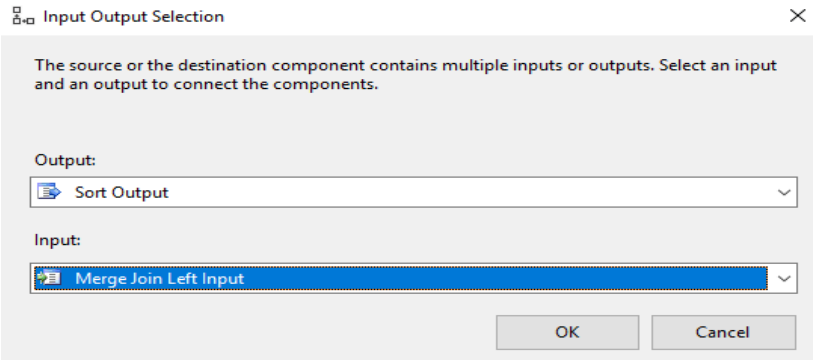


Figure 51: Input Output Selection

Once that is done, create a data flow path from the Sort3 transformation to the Merge Join Transformation.

52. Double click the merge join transformation. Since we need the channel*id from the `dim*channel` table, select it from Sort 3. Select all other columns from the left table (Sort 2) as shown below and hit OK -

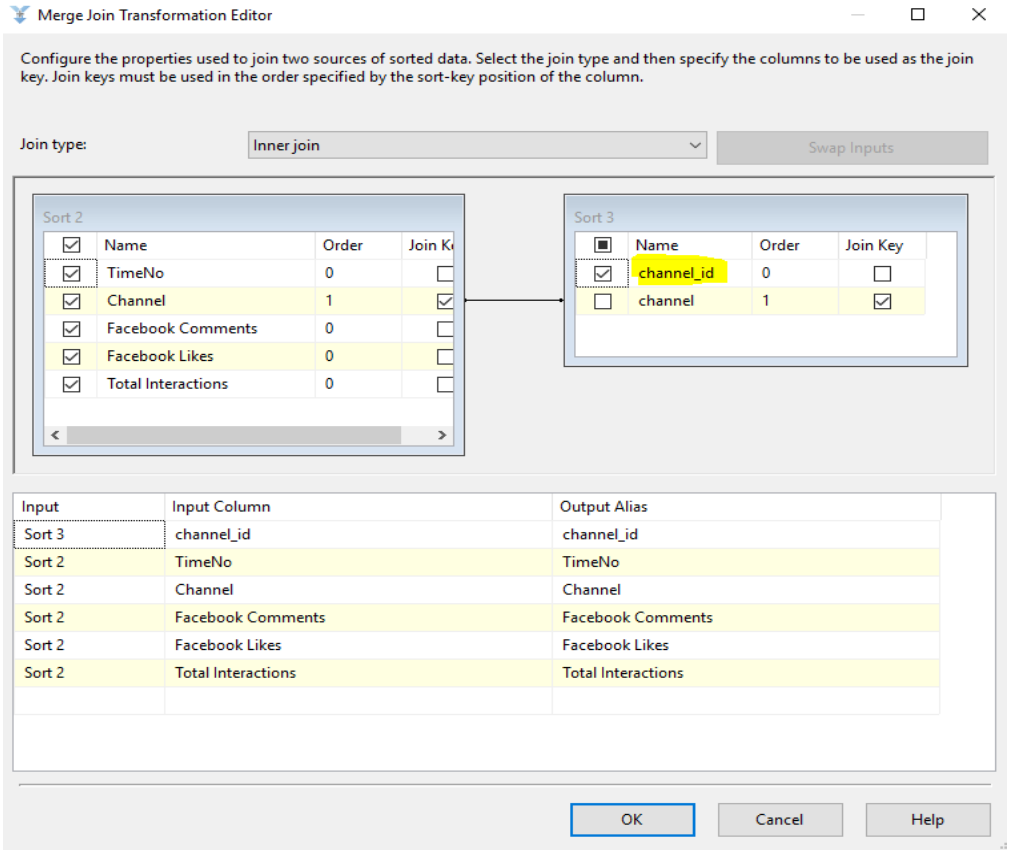


Figure 52: Merge Join Transformation Editor

This completes the major part of our transformations. We only need to populate our fact_interactions table now.

NOTE: Before you connect your data flow to the fact interactions ADO NET Destination. You can check your data flow by clicking Start. You should see 938 rows.

Drag an ADO NET Destination transformation, connect Merge Join 1 to the ADO NET Destination, and configure the ADO NET Destination to fact_interactions table.

53. Select Mappings on the left panel and click the drop down in the Input Column and select the correct mapping referencing the image below.

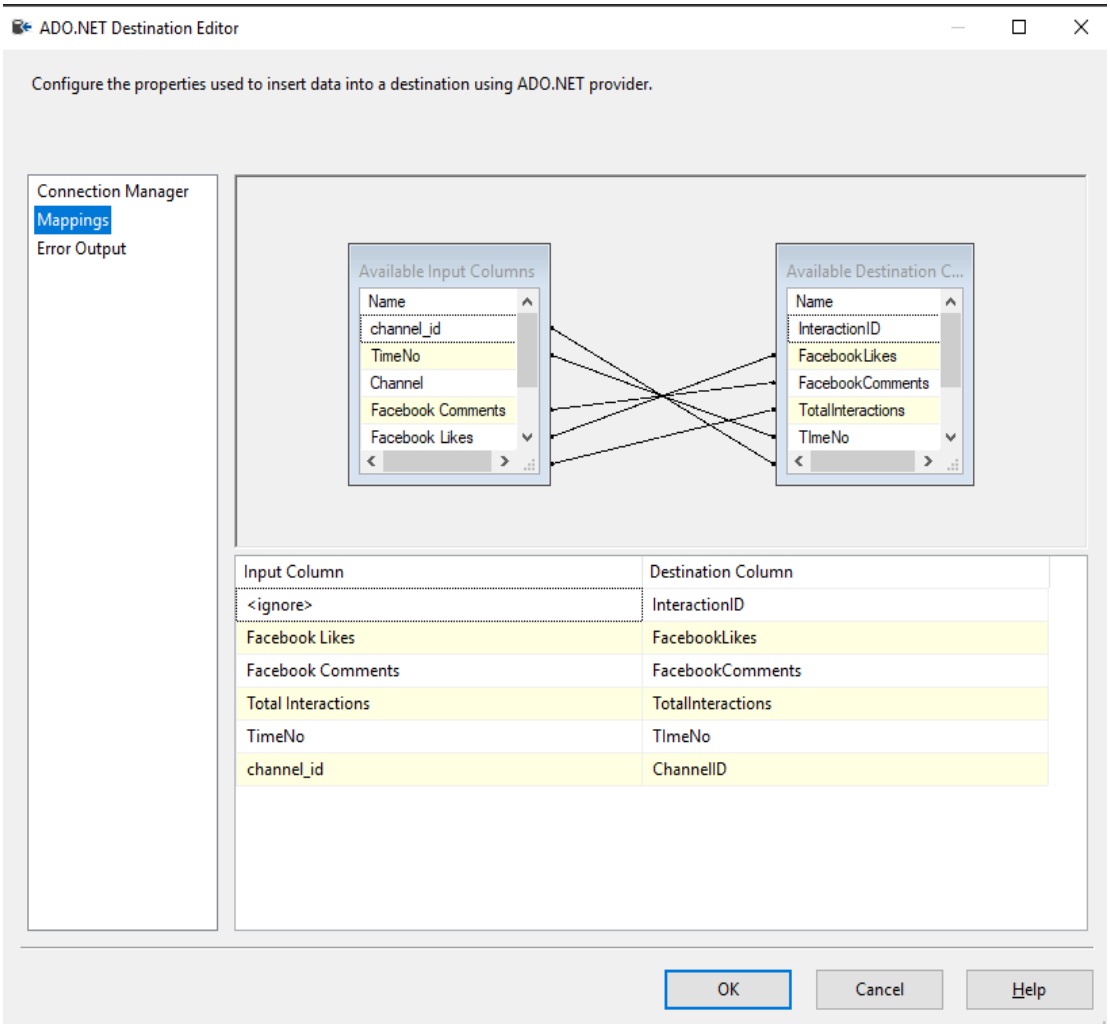


Figure 53: ADO.NET Destination Editor

Hit OK.

54. We are all set to run and load our first fact table. Before doing that make sure there are no Errors on any of the components. Warnings are fine, unless they specify a data type mismatch.

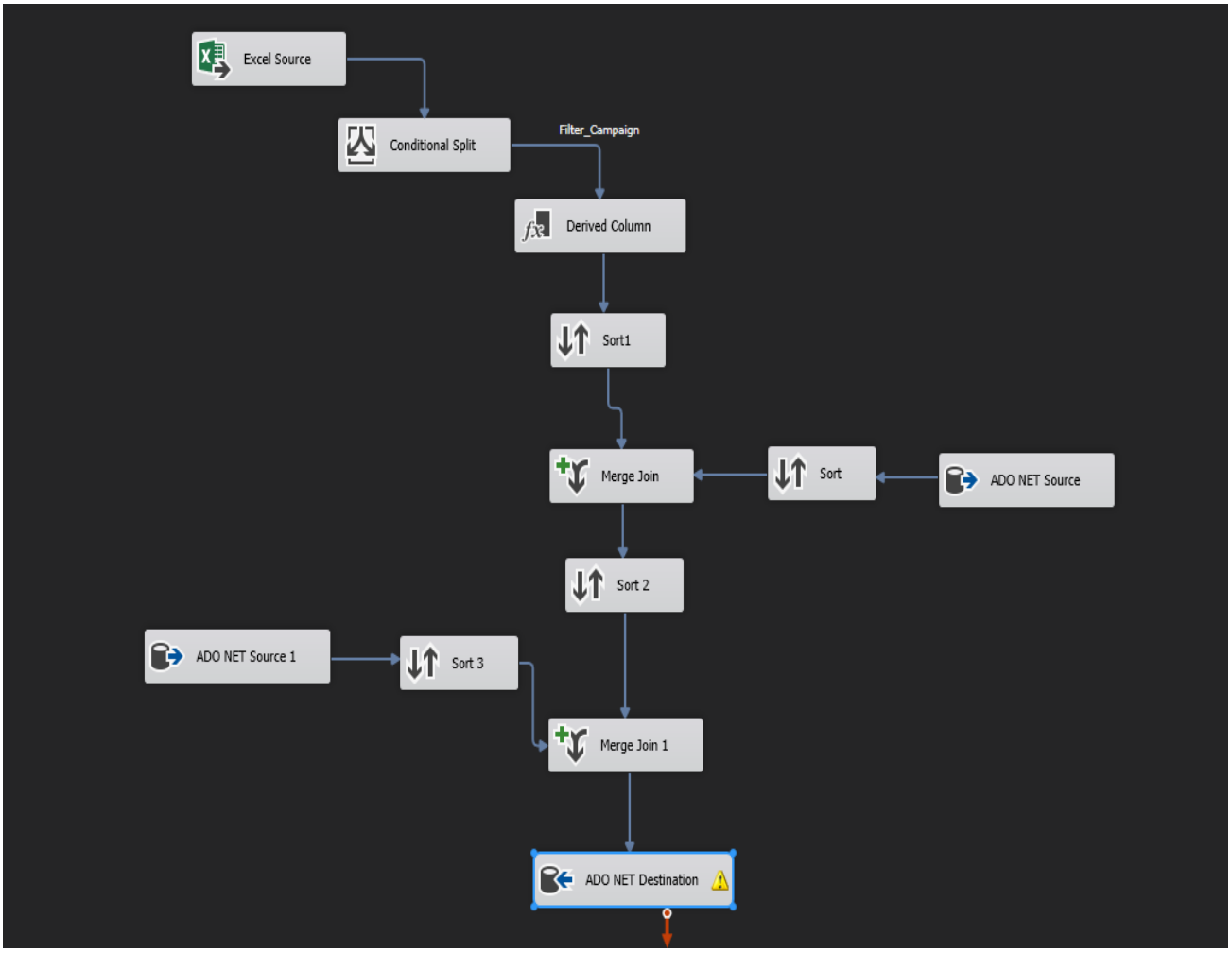


Figure 54: Final fact_interactions mapping

55. Run the transformation. You should be able to see the following result as shown in the figure below.

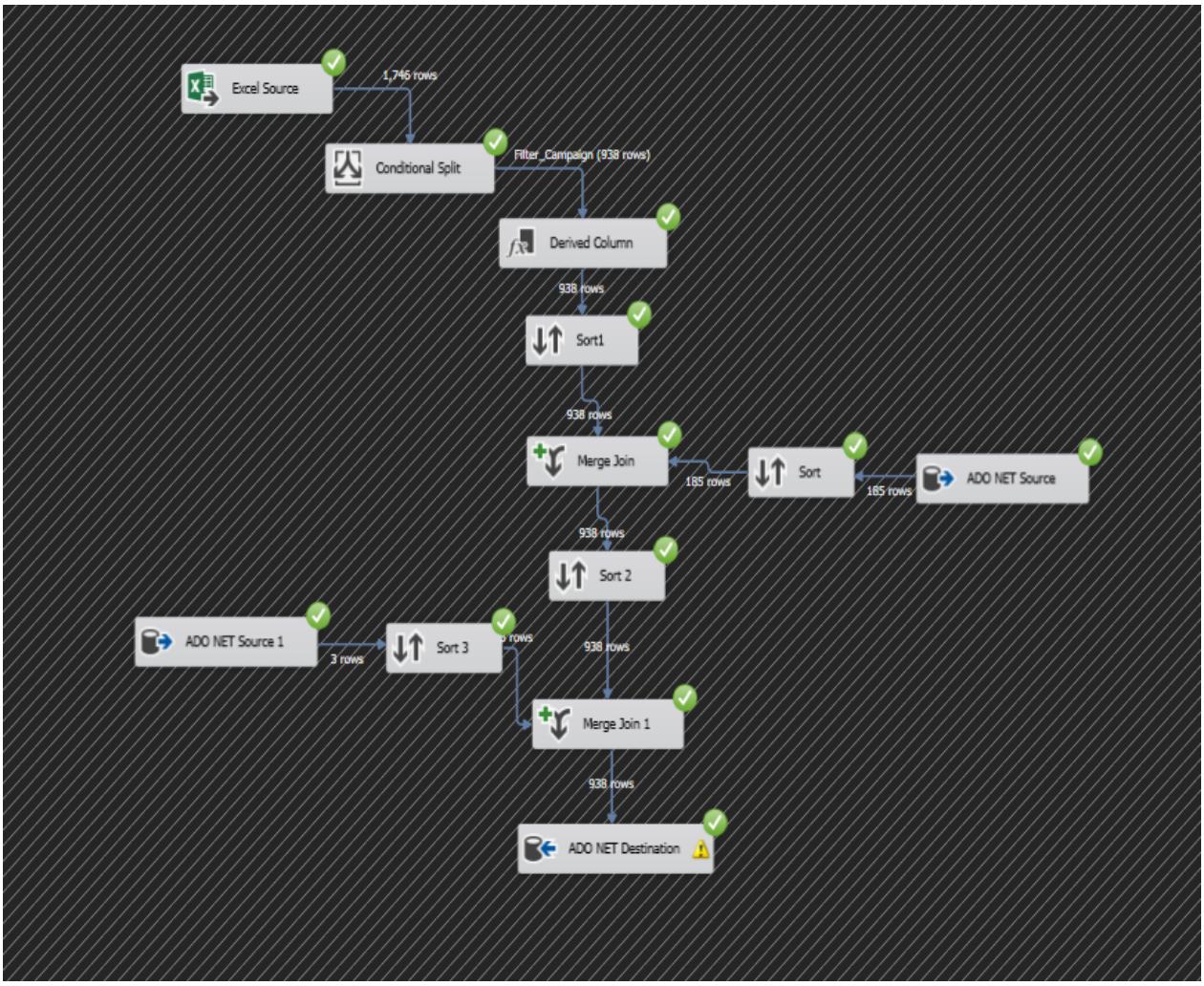


Figure 55: Final fact_interactions mapping result with row count

Total number of row count for each table loaded are as follows

Table name	**Row Count**
dim_time	185
dim_channel	3
fact_interactions	938

This completes our task of creating the two dimension tables and the fact table.

References

[1] SQL Server Integration Services. Online: https://docs.microsoft.com/en-us/sql/integration-services/sql-server-integration-services?view=sql-server-2017

[2] Install Integration Services. Online: https://docs.microsoft.com/en-us/sql/integration-services/install-windows/install-integration-services?view=sql-server-2017

[3] Download and install SQL Server Data Tools (SSDT) for Visual Studio. Online: https://docs.microsoft.com/en-us/sql/ssdt/download-sql-server-data-tools-ssdt?view=sql-server-2017

[4] Integration Services Transformations. Online: https://docs.microsoft.com/en-us/sql/integration-services/data-flow/transformations/integration-services-transformations?view=sql-server-2017