

Assignment 07:

ETL Part 2 Data Loading

Part 1: Overview

This assignment continues where the previous assignment left off by demonstrating how to load your staged data into the data warehouse using an ETL tool. The tool we will use is called SQL Server Integration Services or SSIS.

Again, this assignment will not demonstrate everything you need to know about the ETL process, nor does it serve as training tool for SSIS. Both of these activities are beyond the scope of this course. Yes, when you're finished with the assignment you'll know your way around SSIS, but more importantly, you'll understand how the ETL process gets implemented using modern tooling.

Goals

Specifically the goals of this assignment are to:

- Learn how to transform data using an ETL tool
- Learn how to load data into the data warehouse using an ETL tool using the “upsert” and “Type 2 SCD” ETL patterns

Effort

This assignment is best done individually. Please work alone.

Technical Requirements

To complete this assignment you will need the following:

- Access to the course **ist-cs-dw1.ad.syr.edu** SQL Server, and specifically the Northwind Traders database, your **ist722_yournetid_stage**, and **ist722_yournetid_dw** databases. You should connect to this server before starting the assignment.
- Access to the **SQL Server data tools** which are part of **Visual Studio 2015** (or higher) developer environment. This is the tool we will use to create the ETL Packages.
- You will need to open the **Northwind ETL** SSIS Package solution you started in the previous assignment.

Part 2: Walk-Through

In this part, we'll add an SSIS package to our solution to perform the data loading.

Data Loading for Inventory Daily Snapshot

We now continue with the data loading of the Northwind inventory daily snapshot.

Here's a high-level breakdown of the process:

- In Part 2.1, we'll create a package called **DW_InventoryLevels.dtsx** to complete the ETL from our stage database into our dw database. We will perform the data transformations required by our target dimensions and fact tables to load the data properly. Also, we'll use the SCD (Slowly Changing Dimension) pattern to determine whether we need to "pay attention" to the dimension or fact to avoid loading the same data more than once.

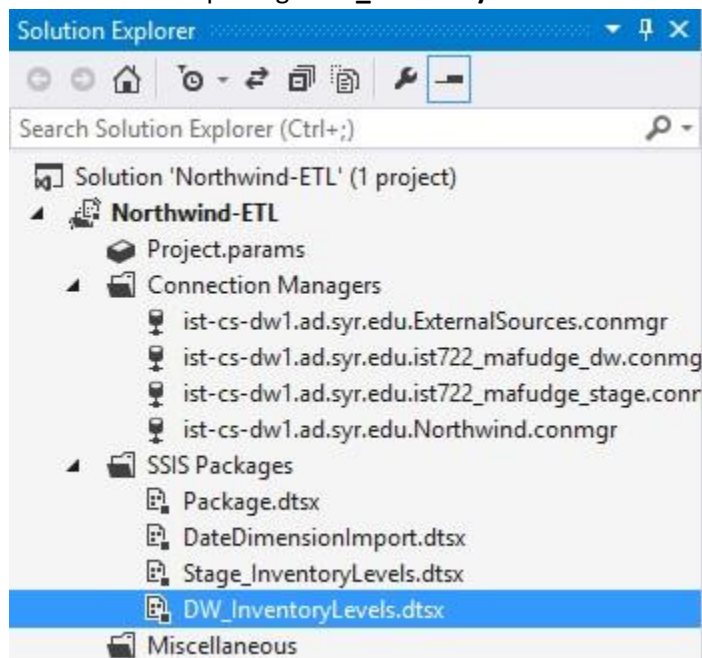
Part 2.1: Loading the Inventory Levels Dimensional Model

In this part, we will load from the stage tables into the fact and dimension tables of our data warehouse. Along the way we will need to transform our data to the requirements of the source tables. Let's get started!

Step 2.3.1: Create the Package

Let's start by creating the **DW_InventoryLevels.dtsx** package to contain our SSIS logic.

1. In the **Solution Explorer** window, right-click **SSIS Packages** and select **New SSIS Package** from the menu.
2. Rename the package **DW_InventoryLevels.dtsx**.



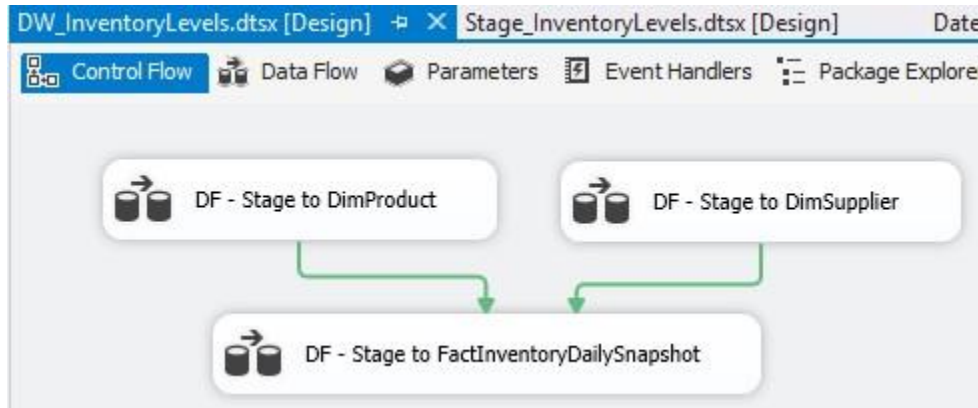
3. Double-click the **DW_InventoryLevels.dtsx** package to open it in the **design surface**.

Step 2.3.2: Set Up the Control Flow

Next we need to setup the base control flow for the package. We will process the dimensions concurrently, and their success will depend on the fact table processing.

1. Add **three data flow tasks** to your design surface.
2. Rename the three data flow tasks:

- a. **DF – Stage to DimProduct**
 - b. **DF – Stage to DimSupplier**
 - c. **DF - Stage to FactInventoryDailySnapshot**
3. Connect the green arrows from each dimension to the fact table.



4. Save your work.

Step 2.3.3: Set Up InitialDate Project Parameter

Since the fact table is a daily periodic snapshot, we need some method of determining the date that the ETL job was executed and therefore the inventory levels added. Normally this would be the current date, but since this is a simulated lab activity, I prefer that we all use the same date. Therefore, we will set a parameter to “simulate” the current date.

1. In the **Solution Explorer** double-click **Project.params**. This will open the **Project.params** tab in the designer.

| Name | Data type | Value | Sensitive | Required | Description |
|------|-----------|-------|-----------|----------|-------------|
|------|-----------|-------|-----------|----------|-------------|

2. Let's create a parameter. In the toolbar, click **Add Parameter**.
3. Configure the parameter as follows:
 - a. Name → **InitialDate**
 - b. Data Type → **DateTime**
 - c. Value → **3/8/2015 12:00:00AM**

| Name | Data type | Value | Sensitive | Required | Description |
|-------------|-----------|----------------------|-----------|----------|-------------|
| InitialDate | DateTime | 3/8/2015 12:00:00 AM | False | False | |

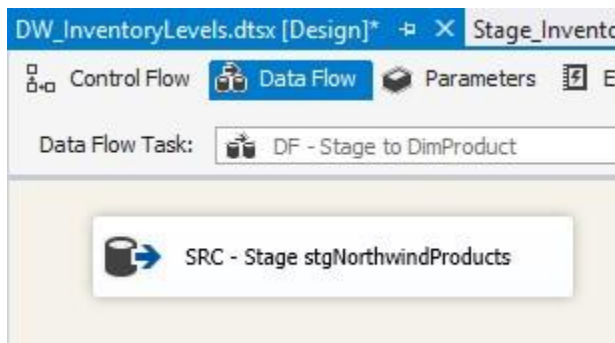
4. Save your work and close the **Project.params** tab.

Step 2.3.4: Data Flow Stage to DimProduct

It's time to work through the logic to import data from **stgNorthwindProducts** into

northwind.DimProduct. We will need to transform data along the way as well as track Type 2 SCD changes. Here's the procedure.

1. Double-click **DF - Stage to DimProduct** to open the data flow.
2. Use the **Source Assistant** to create the data source:
 - a. Type: **SQL Server**
 - b. Connection Manager → **ist722_yournetid_stage**
 - c. Rename to → **SRC - Stage stgNorthwindProducts**
3. Double-click the data source to configure it. The source we will use is the **[stgNorthwindProducts]** table. Select it for the **Name of the table or the view**. Click **OK** to close.




4. Here comes the challenging part. Our staged data does not match the schema of our dimension. Observe:

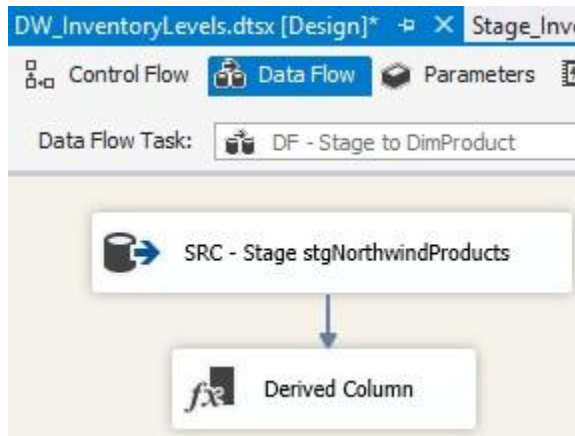
| Stage Database | DW Database |
|---|---|
| <div> <div>dbo.stgNorthwindProducts</div> <div>Columns</div> <div>ProductID (int, null)</div> <div>ProductName (nvarchar(40), null)</div> <div>Discontinued (bit, null)</div> <div>CompanyName (nvarchar(40), null)</div> <div>CategoryName (nvarchar(15), null)</div> </div> | <div> <div>northwind.DimProduct</div> <div>Columns</div> <div>ProductKey (PK, int, not null)</div> <div>ProductID (int, not null)</div> <div>ProductName (nvarchar(40), not null)</div> <div>Discontinued (nchar(1), not null)</div> <div>SupplierName (nvarchar(40), not null)</div> <div>CategoryName (nvarchar(15), not null)</div> </div> |

We need to convert the bit column “Discontinued” into a “Y/N” column in the dimension.

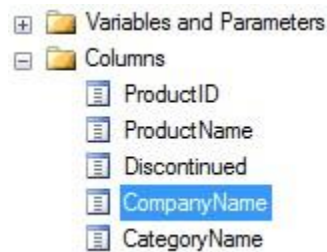
We also need to change CompanyName to SupplierName in the dimension.

- a. Drag and drop the **Derived Column** tool  **Derived Column** onto the design surface.

- b. Connect the output from **SRC - Stage stgNorthwindProducts** to the input of the **Derived Column**.

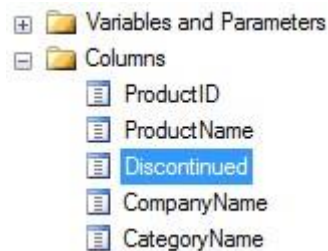


- c. Rename the derived column **DC - Transform Discontinued and SupplierName**.
d. Double-click it to configure it. This opens the **Derived Column Transformation Editor** dialog.
e. Under **Columns**, drag and drop **CompanyName** to the area under Expression.



Then change the name from **Derived Column 1** to **SupplierName**.

- f. Under **Columns**, drag and drop **Discontinued** to the area under [CompanyName].




Edit the expression to say: **[Discontinued] ? "Y" : "N"**

Then change the name from **Derived Column 1** to **DiscontinuedYN**.

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

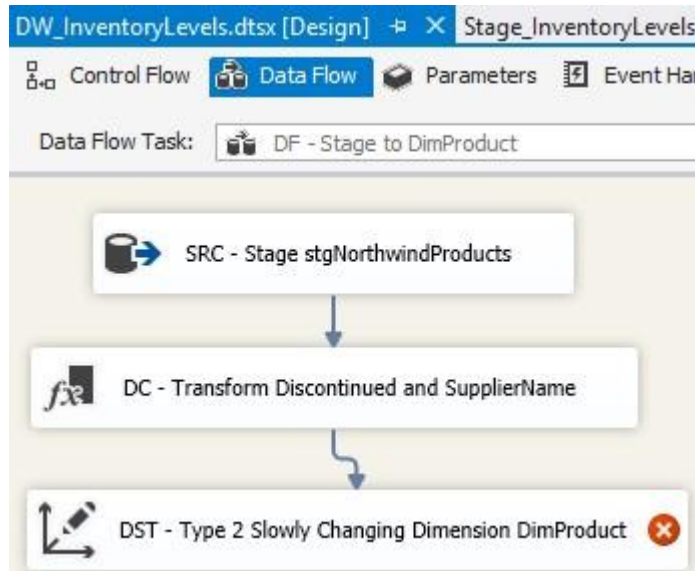
NOTE: The ?: is an “inline if” operator. When Discontinued = true, “Y” is returned; otherwise “N” is returned.

- g. When you’ve completed the transformations, click **OK** to close.
5. Save your work.
6. Now that we have everything we need to populate the dimension, it is time to use the **Slowly Changing Dimension**  destination to populate our Type 2 dimension.

- a. Drag and drop the **Slowly Changing Dimension** destination onto the design surface.

| Derived Column Name | Derived Column | Expression | Data Type |
|---------------------|---------------------|--------------------------|--------------------------|
| SupplierName | <add as new column> | CompanyName | Unicode string [DT_WSTR] |
| DiscontinuedYN | <add as new column> | Discontinued ? "Y" : "N" | Unicode string [DT_WSTR] |

- b. Rename it → **DST - Type 2 Slowly Changing Dimension DimProduct**.
- c. Connect the output of **DC - Transform Discontinued and SupplierName** to the input of **DST - Type 2 Slowly Changing Dimension DimProduct**



7. Double-click **DST - Type 2 Slowly Changing Dimension DimProduct** to configure it.
 - a. Click **Next >** past the first page of the wizard.
 - b. Use **ist722_yournetid_dw** as the connection.
 - c. Select **northwind.DimProduct** as the dimension.
 - d. Configure the input Column: **DiscontinuedYN** → **Discontinued**.
 - e. Set **ProductID** as the business key.

Slowly Changing Dimension Wizard

Select a Dimension Table and Keys
Select a dimension table to load and map columns in the transformation input to columns in the dimension table.

Connection manager:
ist-cs-dw1.ad.syr.edu.ist722_mafudge_dw [New...]

Table or view:
[northwind].[DimProduct]

| Input Columns | Dimension Columns | Key Type |
|----------------|-------------------|------------------|
| CategoryName | CategoryName | Not a key column |
| DiscontinuedYN | Discontinued | Not a key column |
| ProductID | ProductID | Business key |
| ProductName | ProductName | Not a key column |
| | RowChangeReason | |
| | RowEndDate | |
| | RowIsCurrent | |
| | RowStartDate | |
| SupplierName | SupplierName | Not a key column |

Help < Back Next > Finish >>| Cancel

f. Click **Next >**

- g. Add CategoryName, Discontinued, ProductName and SupplierName as **Historical Attributes**.

Slowly Changing Dimension Columns
Manage the changes to column data in your slowly changing dimensions by setting the change type for dimension columns.

Fixed Attribute
Select this type when the value in a column should not change. Changes are treated as errors.

Changing Attribute
Select this type when changed values should overwrite existing values. This is a Type 1 change.

Historical Attribute
Select this type when changes in column values are saved in new records. Previous values are saved in records marked as outdated. This is a Type 2 change.

Select a change type for slowly changing dimension columns:

| Dimension Columns | Change Type |
|-------------------|----------------------|
| CategoryName | Historical attribute |
| Discontinued | Historical attribute |
| ProductName | Historical attribute |
| SupplierName | Historical attribute |

Remove

Help < Back Next > Finish >> Cancel

h. Click **Next >**

- i. Now we must configure the Type 2 SCD metadata columns.

Select **RowIsCurrent** as the column to indicate the current record, and set **Value when current** to **True**.

Slowly Changing Dimension Wizard

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: RowIsCurrent

Value when current: True

Expiration value: False

☐ Use start and end dates to identify current and expired records

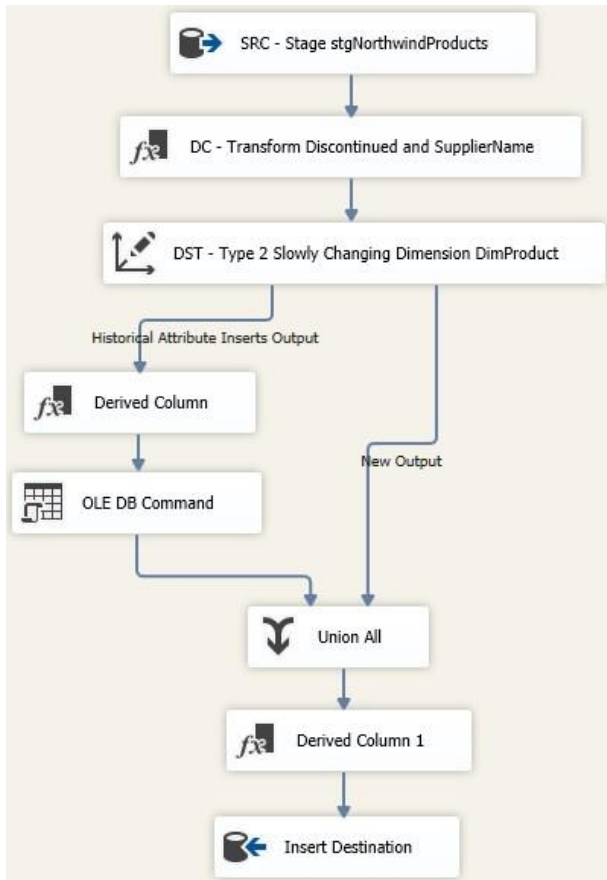
Start date column:

End date column:

Variable to set date values:

Help < Back Next > Finish >> Cancel

- j. Click **Next >**.
 - k. **Turn off** inferred member support, by **clearing the checkbox**. Click **Next >**.
 - l. Click **Finish** to generate the Type 2 SCD logic.
8. Save your work.

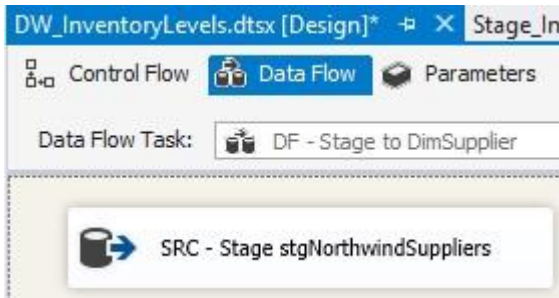


9. Execute the package. Make sure it works. Troubleshoot it if you have problems.
10. Verify there are 78 rows in the northwind.DimProduct dimension (77 ETL rows + the unknown member).

Step 2.3.5: Data Flow Stage to DimSupplier

Now we're going to repeat this process for **northwind.DimSupplier**:

1. Double-click **DF - Stage to DimSupplier** to open the data flow.
2. Use the **Source Assistant** to create the data source:
 - a. Type: **SQL Server**
 - b. Connection Manager → **ist722_yournetid_stage**
 - c. Rename to → **SRC - Stage stgNorthwindSuppliers**
3. Double-click the data source to configure it. The source we will use is the **[stgNorthwindSuppliers]** table. Select it for the **Name of the table or the view**. Click **OK** to close.



4. Of course, our staged data does not match the schema of our dimension. Observe:

| Stage Database | DW Database |
|---|--|
| <div> <div>dbo.stgNorthwindSuppliers</div> <div>Columns</div> <div> SupplierID (int, null) CompanyName (nvarchar(40), null) ContactName (nvarchar(30), null) ContactTitle (nvarchar(30), null) City (nvarchar(15), null) Region (nvarchar(15), null) Country (nvarchar(15), null) </div> </div> | <div> <div>northwind.DimSupplier</div> <div>Columns</div> <div> SupplierKey (PK, int, not null) SupplierID (int, not null) CompanyName (nvarchar(40), not null) ContactName (nvarchar(30), not null) ContactTitle (nvarchar(30), not null) City (nvarchar(15), not null) Region (nvarchar(15), not null) Country (nvarchar(15), not null) </div> </div> |

This time our transformation is a little more subtle. There are nulls in the source of **Region** which must be replaced with “N/A” in the dimension. Nulls in the dimension are bad because when a user is exploring a dimension model, the information is just blank and therefore confusing.

- Drag and drop the **Derived Column** tool onto the design surface.
- Connect the output from **SRC - Stage stgNorthwindProducts** to the input of the **Derived Column**.
Rename the derived column **DC - Replace Nulls with Defaults**.
- Double-click it to configure it. This opens the **Derived Column Transformation Editor** dialog.

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

- Under **Columns**, drag and drop **Region** to the area under **Expression**.

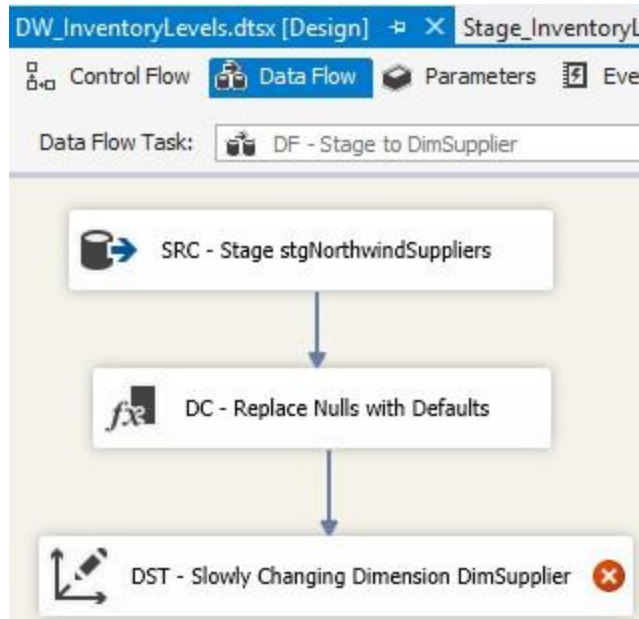
| Derived Column Name | Derived Column | Expression | Data Type | Length |
|---------------------|---------------------|------------------------------------|--------------------------|--------|
| RegionNA | <add as new column> | LEFT(REPLACENULL(Region,"N/A"),15) | Unicode string [DT_WSTR] | 15 |

Change the name from **Derived Column 1** to **RegionNA**.

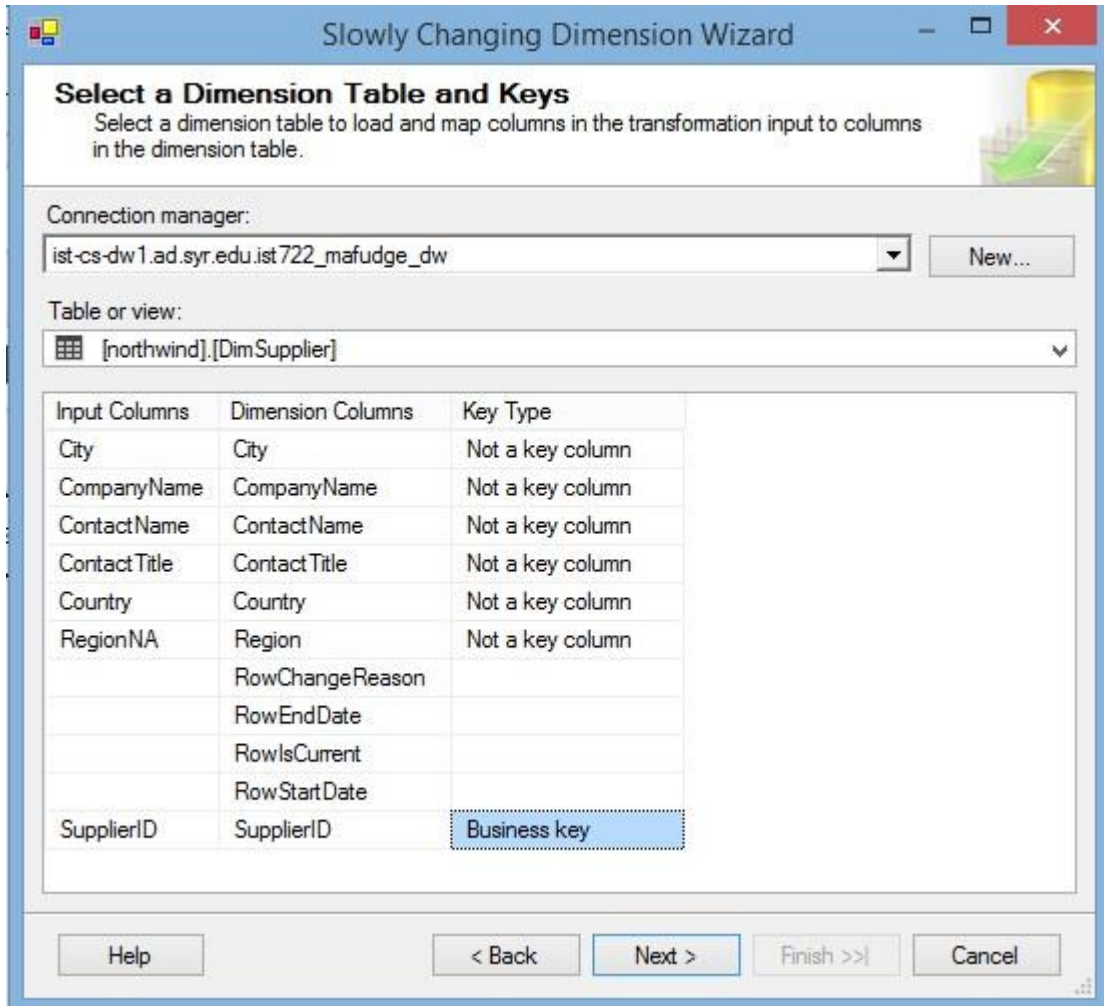
Change the Expression from **[Region]** to **LEFT(REPLACENULL(Region,"N/A"),15)**

This expression replaces nulls with “N/A” and truncates the data to under 15 characters.

- e. Click **OK** to close.
5. Save your work.
6. Again, we're ready to populate our Type 2 dimension.
 - a. Drag and drop the **Slowly Changing Dimension** destination on the design surface.
 - b. Rename it → **DST - Type 2 Slowly Changing Dimension DimSupplier**
 - c. Connect the output of **DC - Replace Nulls with Defaults** to the input of **DST - Slowly Changing Dimension DimSupplier**.



7. Double-click **DST - Type 2 Slowly Changing Dimension DimSupplier** to configure it.
 - a. Click **Next >** past the first page of the wizard.
 - b. Use **ist722_yournetid_dw** as the connection.
 - c. Select **northwind.DimSupplier** as the dimension.
 - d. Configure the input Column: **RegionNA** → **Region**.
 - e. Set **SupplierID** as the business key.



The image shows the 'Slowly Changing Dimension Wizard' dialog box. The title bar reads 'Slowly Changing Dimension Wizard'. The main heading is 'Select a Dimension Table and Keys'. Below this, it says 'Select a dimension table to load and map columns in the transformation input to columns in the dimension table.' The 'Connection manager:' dropdown is set to 'ist-cs-dw1.ad.syr.edu.ist722_mafudge_dw'. The 'Table or view:' dropdown is set to '[northwind].[DimSupplier]'. A table below lists input and dimension columns and their key types. The 'SupplierID' row is highlighted, showing 'SupplierID' as the input column, 'SupplierID' as the dimension column, and 'Business key' as the key type. At the bottom, there are buttons for 'Help', '< Back', 'Next >', 'Finish >>', and 'Cancel'.

| Input Columns | Dimension Columns | Key Type |
|---------------|-------------------|------------------|
| City | City | Not a key column |
| CompanyName | CompanyName | Not a key column |
| ContactName | ContactName | Not a key column |
| ContactTitle | ContactTitle | Not a key column |
| Country | Country | Not a key column |
| RegionNA | Region | Not a key column |
| | RowChangeReason | |
| | RowEndDate | |
| | RowIsCurrent | |
| | RowStartDate | |
| SupplierID | SupplierID | Business key |

f. Click **Next >**

g. Add City, CompanyName, ContactName, ContactTitle, Country, and Region as
Historical Attributes

Slowly Changing Dimension Columns
Manage the changes to column data in your slowly changing dimensions by setting the change type for dimension columns.

Fixed Attribute
Select this type when the value in a column should not change. Changes are treated as errors.

Changing Attribute
Select this type when changed values should overwrite existing values. This is a Type 1 change.

Historical Attribute
Select this type when changes in column values are saved in new records. Previous values are saved in records marked as outdated. This is a Type 2 change.

Select a change type for slowly changing dimension columns:

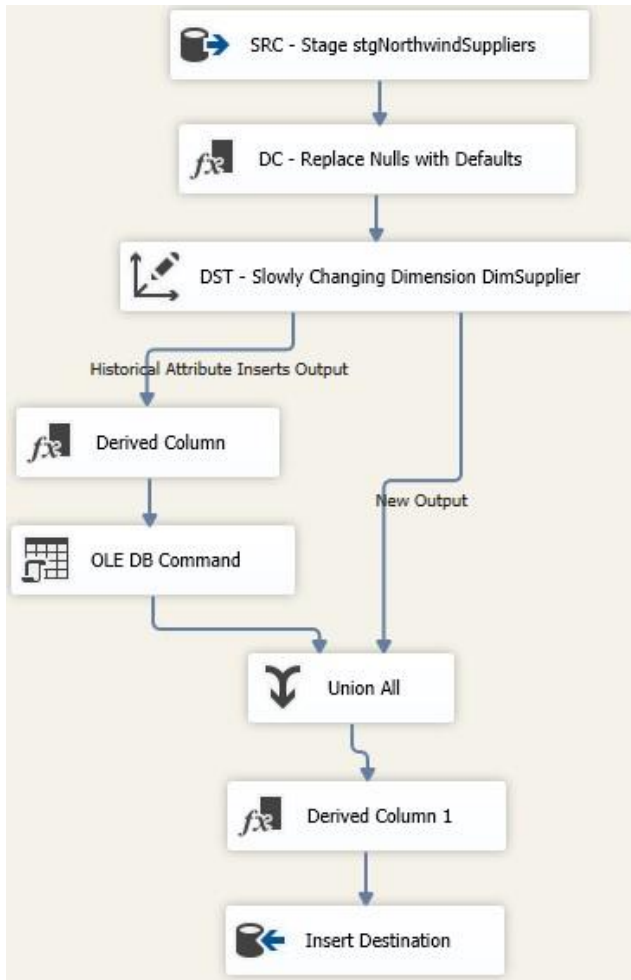
| Dimension Columns | Change Type |
|-------------------|----------------------|
| City | Historical attribute |
| CompanyName | Historical attribute |
| ContactName | Historical attribute |
| ContactTitle | Historical attribute |
| Country | Historical attribute |
| Region | Historical attribute |
| | |

Remove

Help < Back Next > Finish >> Cancel

h. Click **Next >**

- i. Again, configure the type 2 SCD metadata columns.
Select **RowIsCurrent** as the column to indicate current record. Set **Value when current** to **True**.
 - j. Click **Next >**.
 - k. **Turn off** inferred member support by **clearing the checkbox**. Click **Next >**.
 - l. Click **Finish** to generate the Type 2 SCD logic.
8. Save your work.



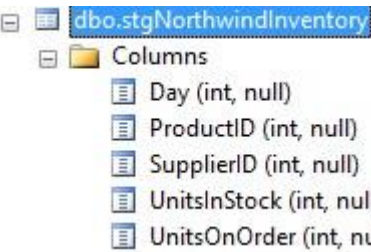
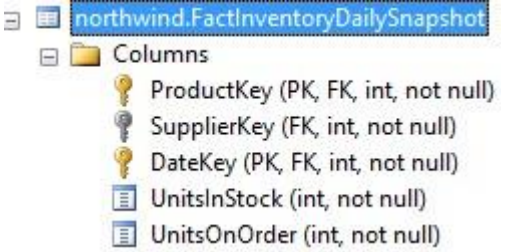
9. Execute the package. Make sure it works. Troubleshoot it if you have problems.
10. Verify there are 30 rows in the northwind.DimSupplier dimension (29 ETL rows + the unknown member).

Step 2.4.6: Data Flow Stage to FactInventoryDailySnapshot

In this last step we complete this ETL for the fact table. This is a more complicated process as we must look up the dimension primary keys using our business keys. (This is called the surrogate key pipeline.).

1. Double-click **DF - Stage to FactInventoryDailySnapshot** to open the data flow.
2. Use the **Source Assistant** to create the data source:
 - a. Type: **SQL Server**
 - b. Connection Manager → **ist722_yournetid_stage**
 - c. Rename to → **SRC - Stage stgNorthwindInventory**
3. Double-click the data source to configure it. The source we will use is the **[stgNorthwindInventory]** table. Select it for the **Name of the table or the view**. Click **OK** to close.

4. Since our staged data does not match the schema of our dimension, we've got some conversions to process:

| Stage Database | DW Database |
|---|--|
|  |  |

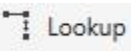
We need to convert the **Day** column into a DateKey by adding in our **InitialDate** parameter. Typically, we would require some fact calculation at this step but the nature of facts in this example do not warrant it.

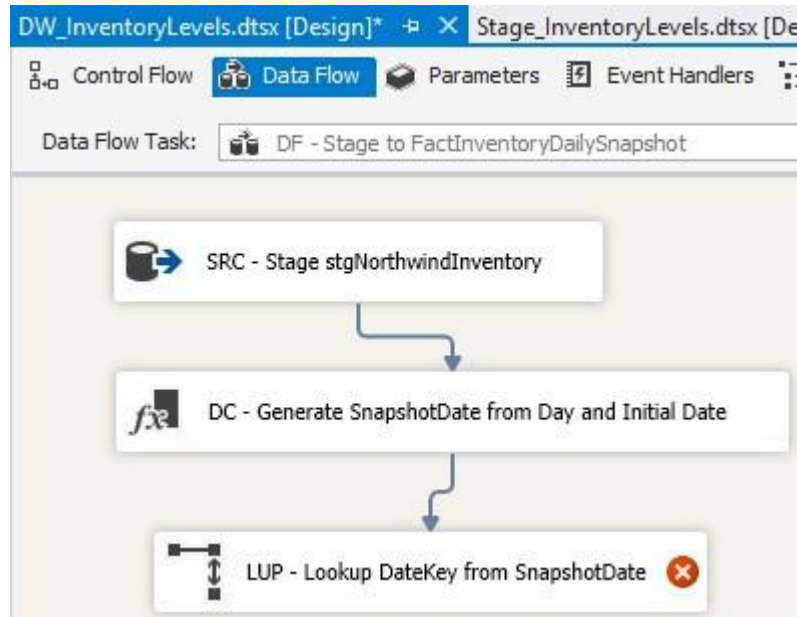
- a. Drag and drop the **Derived Column** tool onto the design surface.
- b. Connect the output from **SRC - Stage stgNorthwindInventory** to the input of the **Derived Column**.
- c. Rename the derived column **DC - Generate SnapshotDate from Day and Initial Date**.

| Derived Column Name | Derived Column | Expression | Data Type |
|---------------------|---------------------|--|-------------------------------------|
| SnapshotDate | <add as new column> | DATEADD("d",Day,@[\$Project::InitialDate]) | database timestamp [DT_DBTIMESTAMP] |

- d. Double-click it to configure it. This opens the **Derived Column Transformation Editor** dialog.
- e. Under **Columns**, drag and drop **Day** to the area under **Expression**.
Change the name from **Derived Column 1** to **SnapshotDate**.
Change the Expression from **[Day]** to **DATEADD("d",Day,@[\$Project::InitialDate])**
This adds **Day** days to the **InitialDate**.
- f. Click **OK** to close.

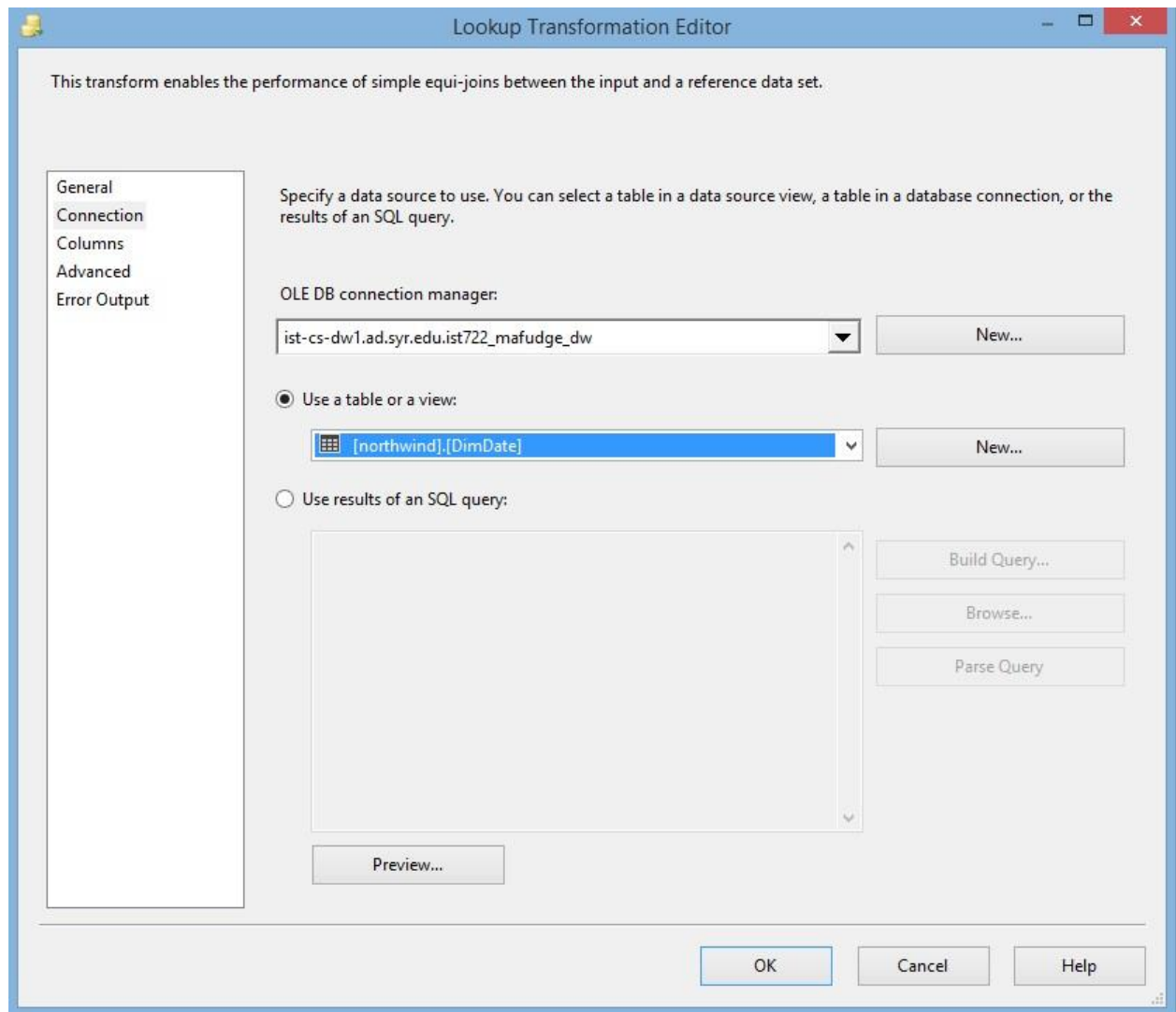
5. Save your work.
6. Now we need to look up the DateKey using the SnapshotDate. We will then repeat this process two more times for the SupplierKey and ProductKey.

- a. Drag and drop the **Lookup**  tool onto the design surface.
- b. Connect the output from **DC - Generate SnapshotDate from Day and Initial Date** to the input of **Lookup**.
- c. Rename the lookup to **LUP - Lookup DateKey from SnapshotDate**. Here's what you should have at this point:

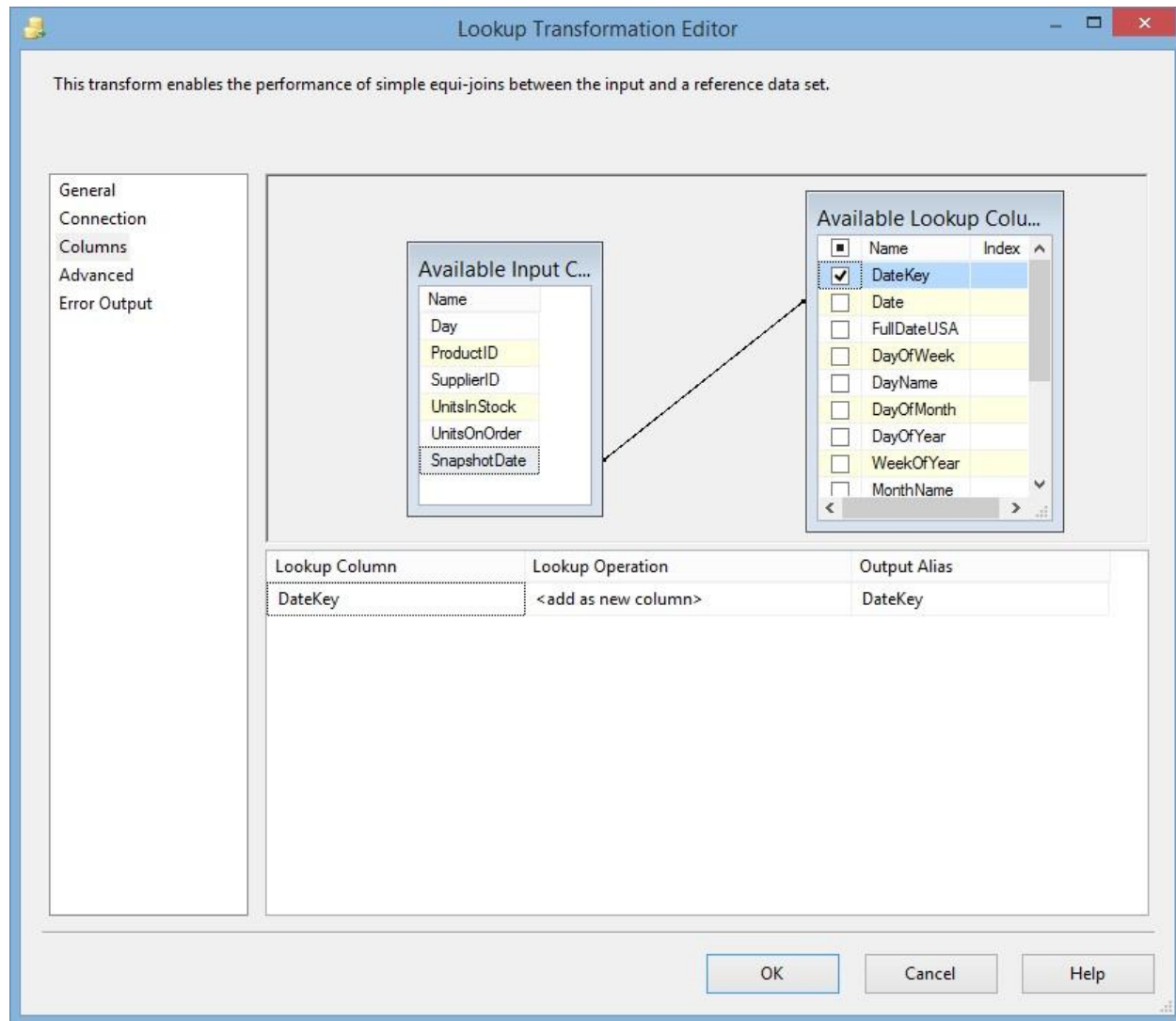


- d. Double-click **LUP - Lookup DateKey from SnapshotDate** to configure it. This displays the **Lookup Transformation Editor**.
- e. Click **Connection** in the left menu.
Choose **ist722_yournetid_dw** as the connection and use **northwind.DimDate** as the

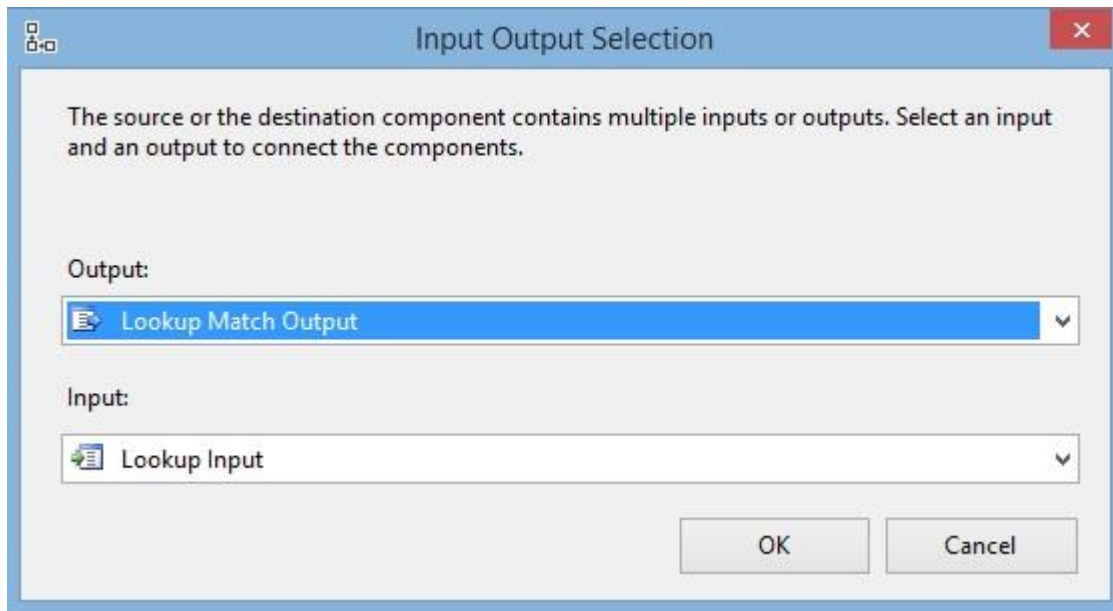
table or view.



- f. Click the **Columns** section on the left menu. Join the two tables on their common business key **SnapshotDate** $\leftarrow \rightarrow$ **Date**. Then select **DateKey** to add it to the output.
Click **OK** to close the dialog.
7. Let's repeat this process for SupplierKey :
 - a. Drag and drop the **Lookup** tool onto the design surface.

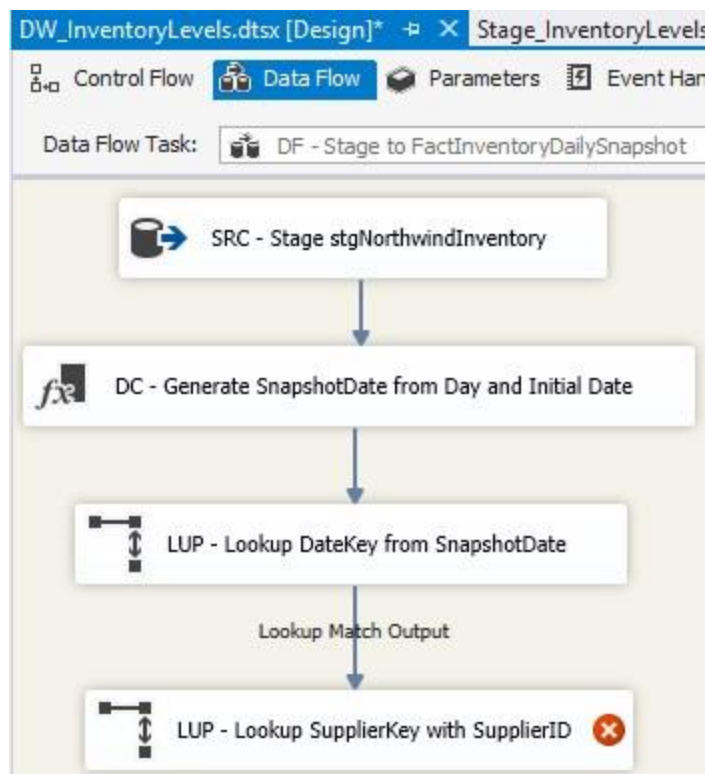


- Connect the output from **LUP - Lookup DateKey from SnapshotDate** to the input of **Lookup**.
- The **Lookup** tool has **two outputs: match & no match**. We need to select the output we would like to match to this input.



Choose **Lookup Match Output**, and then click **OK**.

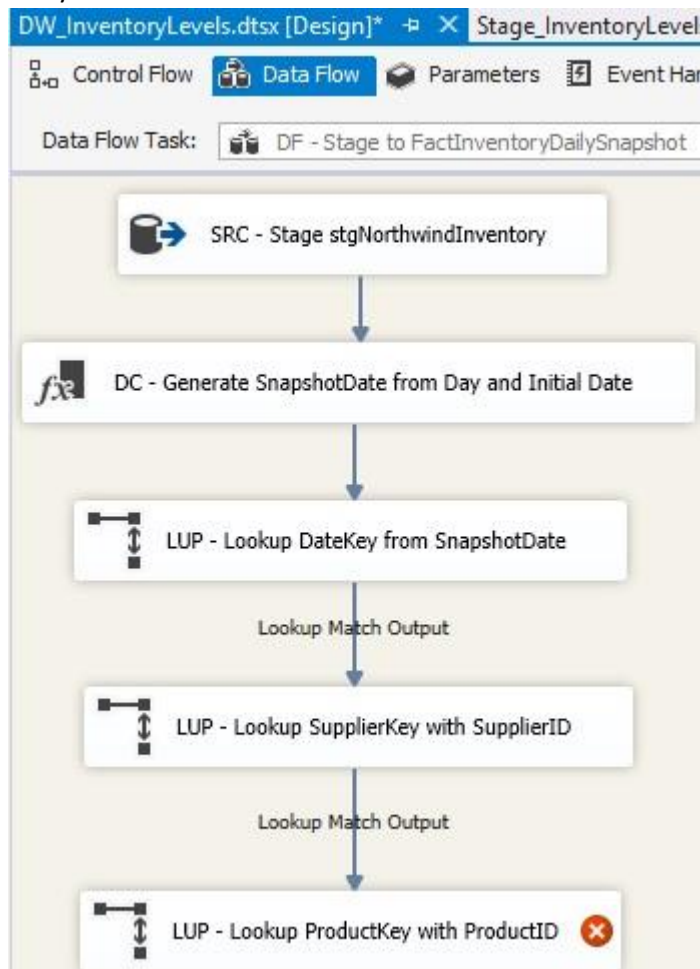
Rename the lookup to **LUP - Lookup SupplierKey with SupplierID**. Here's what you should have:



- d. Double-click **LUP - Lookup SupplierKey with SupplierID** to configure it. This displays the **Lookup Transformation Editor**.
- e. Click **Connection** in the left menu.

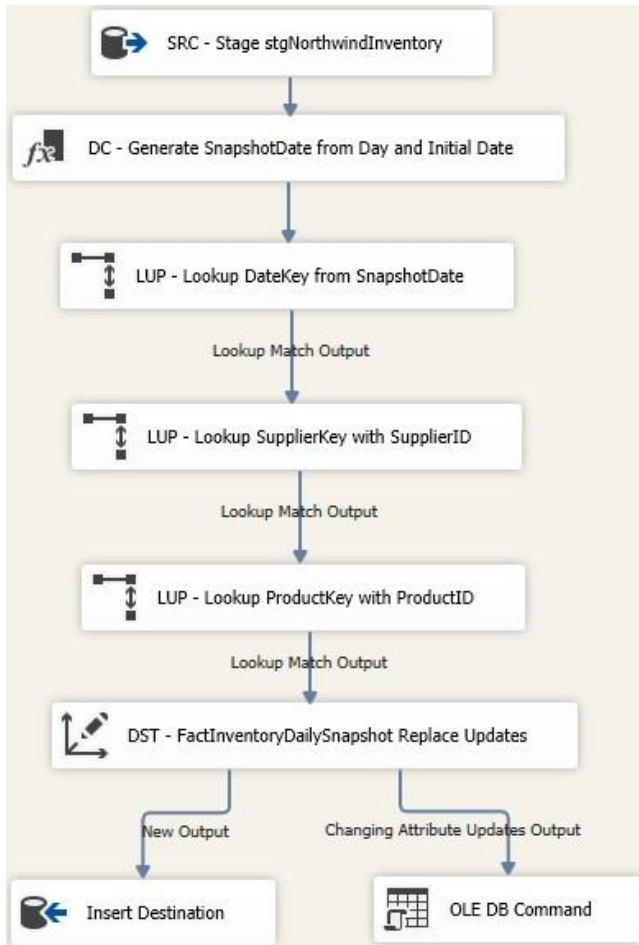
Choose **ist722_yournetid_dw** as the connection and use **northwind.DimSupplier** as the table or view.

- f. Click the **Columns** section on the left menu. Join the two tables on their common business key **SupplierID** $\leftarrow \rightarrow$ **SupplierID**. Then select **SupplierKey** to add it to the output.
 - g. Click **OK** to close the dialog.
8. Let's repeat this process one last time for ProductKey :
- a. Drag and drop the **Lookup** tool onto the design surface.
 - b. Connect the output from **LUP - Lookup SupplierKey with SupplierID** to the input of **Lookup**.
 - c. For the **Input Output Selection** choose an output of **Lookup Match Output**, and then click **OK**.
 - d. Rename the lookup to **LUP - Lookup ProductKey with ProductID**. Here's what you should have:



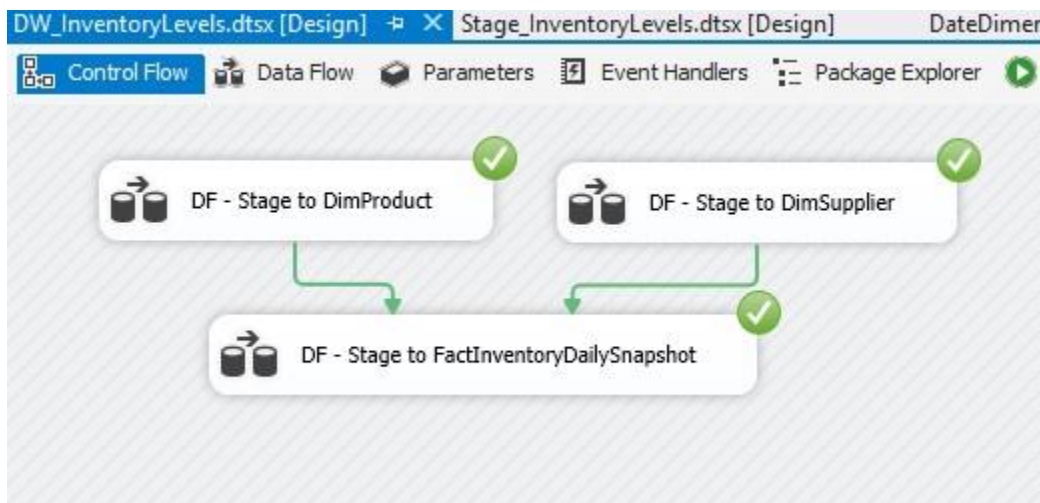
- e. Double-click **LUP - Lookup ProductKey with ProductID** to configure it. This displays the **Lookup Transformation Editor**.
- f. Click the **Connection** section on the left menu.

- Choose **ist722_yournetid_dw** as the connection and use **northwind.DimProduct** as the table or view.
- g. Click the **Columns** section on the left menu. Join the two tables on their common business key **ProductID** $\leftarrow \rightarrow$ **ProductID**. Then check **ProductKey** to add it to the output.
 - h. Click **OK** to close the dialog.
9. Now would be a really good time to save your work!
10. Now that the surrogate key pipeline is complete, it's time to add the facts to the fact table. Believe it or not, we will use a Type 1 SCD to complete this. That way, we will not re-add the same fact, and if any of our facts change, the row will be updated.
- a. Drag and drop the **Slowly Changing Dimension** destination onto the design surface.
 - b. Rename it \rightarrow **DST - FactInventoryDailySnapshot Replace Updates**.
 - c. Connect the output of **LUP - Lookup ProductKey with ProductID** to the input of **DST - FactInventoryDailySnapshot Replace Updates**. Again, choose **Lookup Match Output**.
11. Double-click **DST - FactInventoryDailySnapshot Replace Updates** to configure it.
- a. Click **Next >** past the first page of the wizard.
 - b. Use **ist722_yournetid_dw** as the connection.
 - c. Select **northwind.FactDailySnapshot** as the dimension.
 - d. Set all of the keys as the business key. (**DateKey**, **ProductKey** and **SupplierKey**) This is common in fact tables as we only want to update the fact to correct an error.
 - e. Click **Next >**.
 - f. Set **UnitsInStock** and **UnitsOnOrder** as **changing attributes**.
 - g. Click **Next >**.
 - h. **Clear the check box** for change all matching records. It does not apply here.
 - i. Click **Next >**.
 - j. **Turn off** inferred member support by **clearing the checkbox**.
 - k. Click **Next >**.
 - l. Click **Finish** to generate the Type 2 SCD logic.
12. You're done! It's time to save your work.



13. Execute the package. Make sure it works. Troubleshoot it if you have problems.
14. Verify that there are **616 rows** in the **northwind.FactInventoryDailySnapshot**.

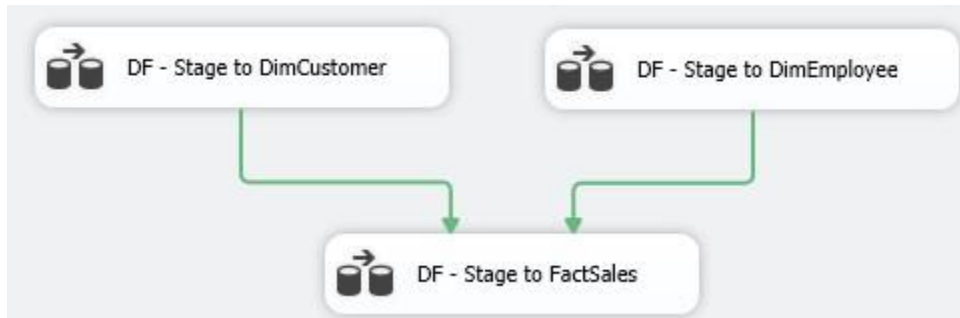
If you switch back to the **Control Flow** tab, you can see the entire Dimensional Model loads correctly:



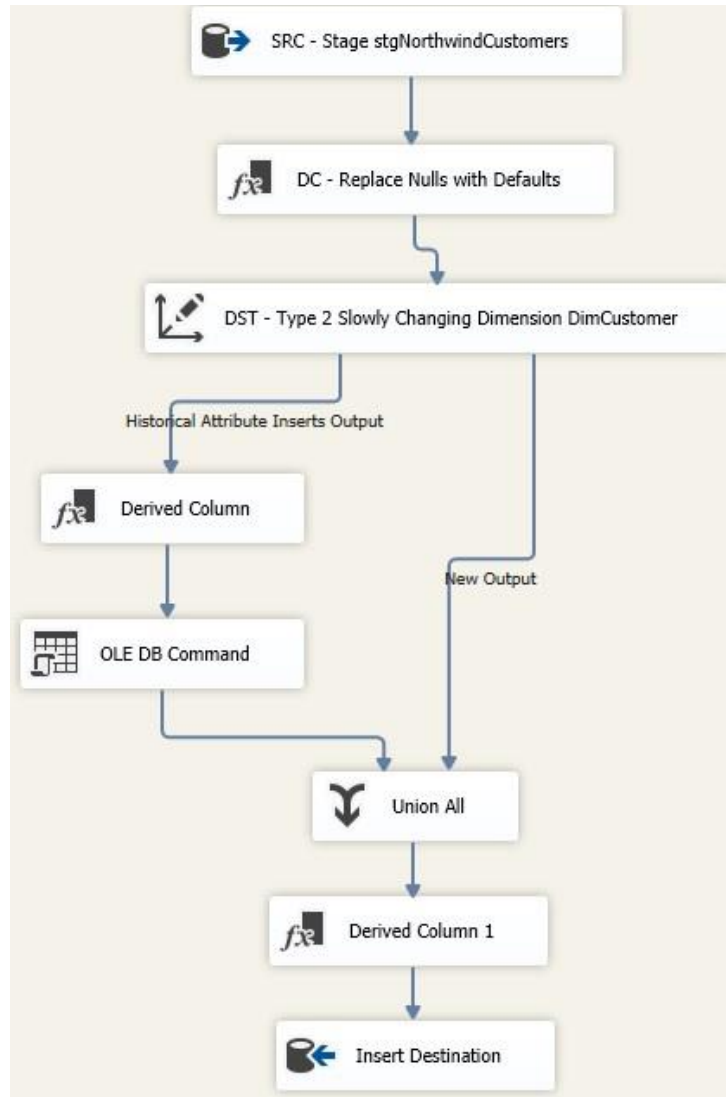
Part 3: On Your Own

In this part, try to build out the ETL for Northwind Sales data mart. Here's a high-level outline of the process. Remember, use what you learned in Part 2 and apply it here.

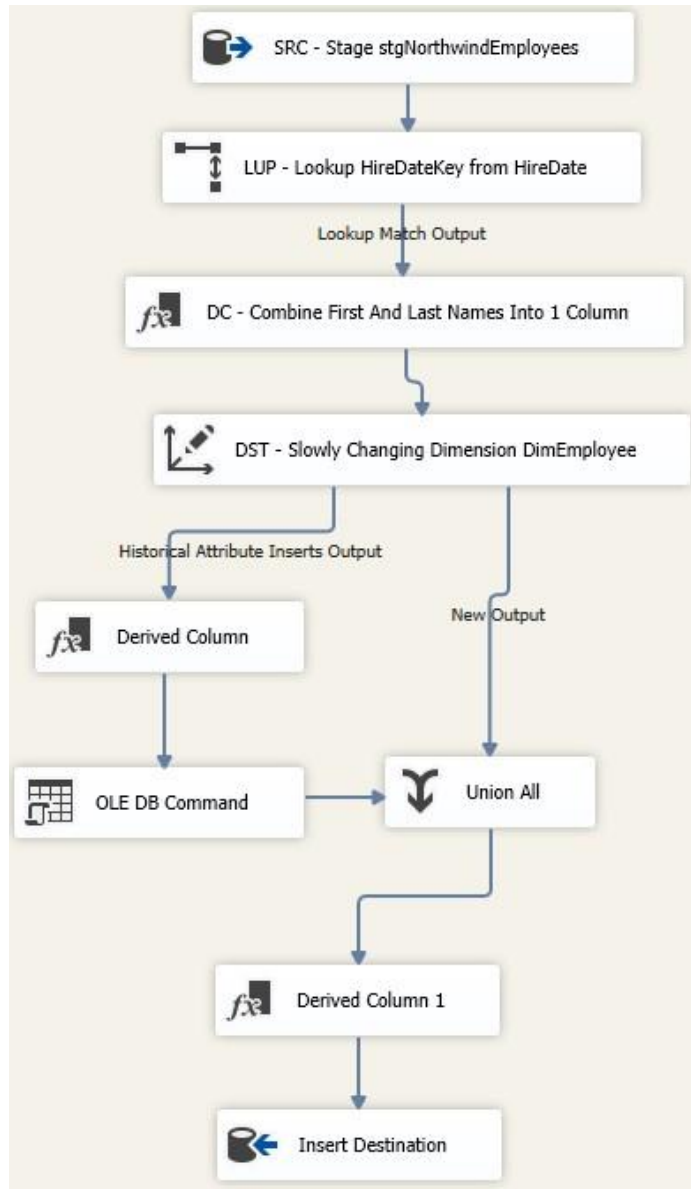
1. Add a package to your project called **DW_Sales.dtsx**, which will load the staged data into the data warehouse. Here's the control flow you'll need to set-up:



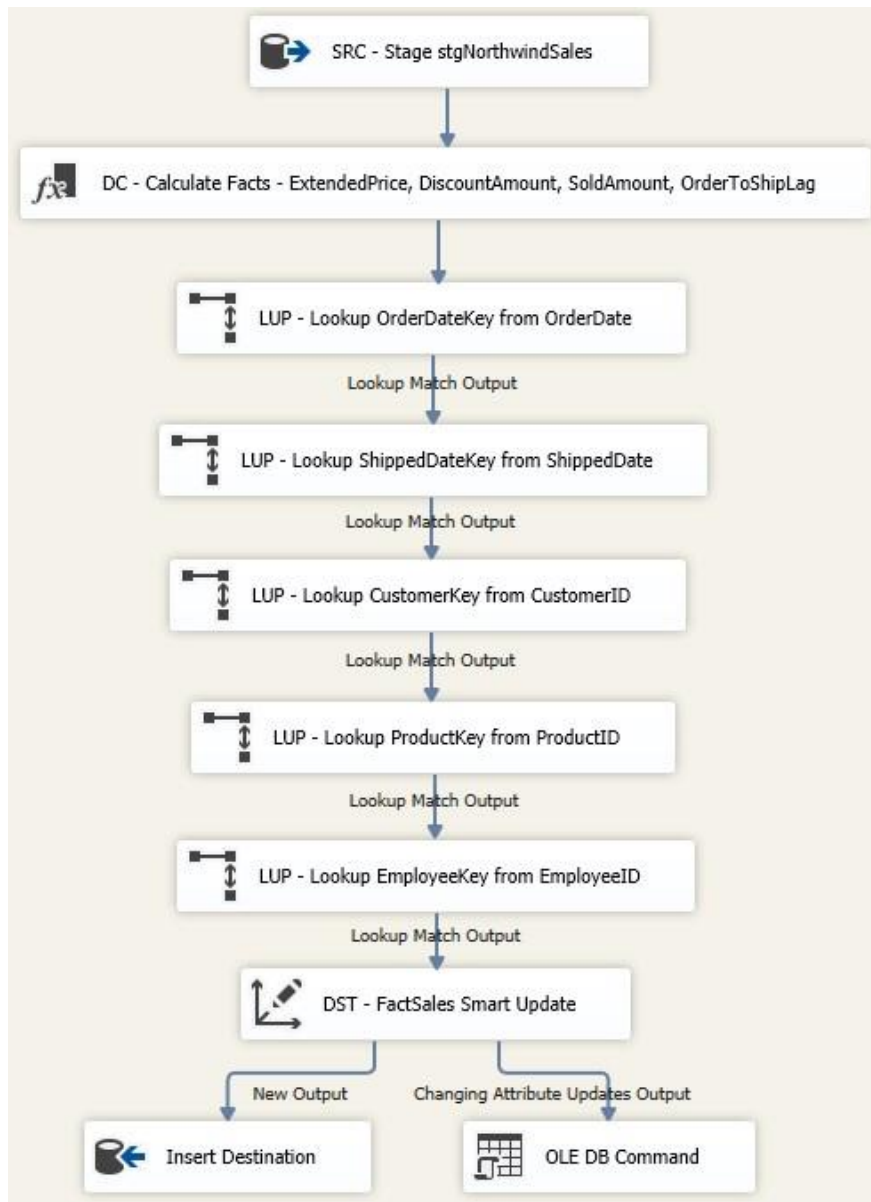
- a. Here's the data flow you'll need to build for **Stage to DimCustomer**:



- b. Here is the data flow you'll need to build for **Stage to DimEmployee**. Notice there is a lookup conversion of HireDate to HireDateKey.

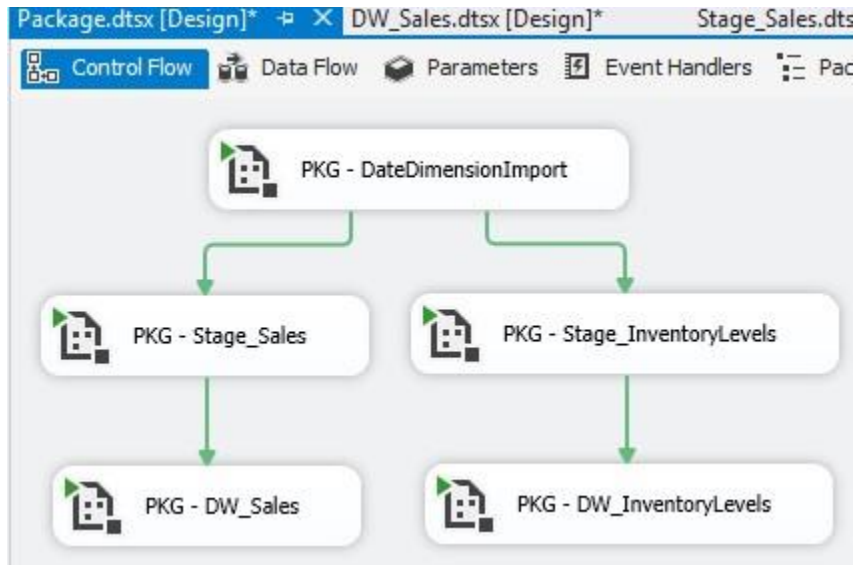


c. Here's the data flow you'll need to build for **Stage to FactSales**:



NOTE: I encourage you to try to figure out the derived columns and other transformations. If you are struggling, refer to **Appendix – Column Transformations**.

2. In the final step, complete **Package.dtsx**—one package to run them all!!!



Appendix – Column Transformations

DC-Replace Nulls With Defaults

| Derived Column Name | Derived Column | Expression | Data Type | Length |
|---------------------|---------------------|--|--------------------------|--------|
| RegionNA | <add as new column> | LEFT(REPLACENULL(Region,"N/A"),15) | Unicode string [DT_WSTR] | 15 |
| PostalCodeUnknown | <add as new column> | LEFT(REPLACENULL(Region,"Unknown"),10) | Unicode string [DT_WSTR] | 10 |

DC - Combine First and Last Names Into 1 Column

| Derived Column Name | Derived Column | Expression | Data Type | Length | Precis |
|---------------------|---------------------|--|--------------------------|--------|--------|
| EmployeeName | <add as new column> | FirstName + " " + LastName | Unicode string [DT_WSTR] | 31 | |
| SupervisorName | <add as new column> | SupervisorFirstName + " " + SupervisorLastName | Unicode string [DT_WSTR] | 31 | |

DC - Calculate Facts - ExtendedPrice, DiscountAmount, SoldAmount, OrderToShipLag

| Derived Column Name | Derived Column | Expression | Data Type | Length |
|-------------------------|---------------------|--|---------------------------------|--------|
| ExtendedPriceAmount | <add as new column> | (DT_NUMERIC,18,4)(Quantity * UnitPrice) | numeric [DT_NUMERIC] | |
| DiscountAmount | <add as new column> | (DT_NUMERIC,18,4)(Quantity * UnitPrice * Discount) | numeric [DT_NUMERIC] | |
| SoldAmount | <add as new column> | (DT_NUMERIC,18,4)(Quantity * UnitPrice * (1 - Discount)) | numeric [DT_NUMERIC] | |
| OrderToShippedLagInDays | <add as new column> | (DT_I2)(DATEDIFF("d",OrderDate,ShippedDate)) | two-byte signed integer [DT_I2] | |

Turning It In

Please turn in a Word document with your name, NetID, and date at the top. Paste a screenshot of your **DW_Sales.dtsx** executing successfully (green checkmarks next to all the tasks) and make sure your name and NetID appear on the screenshot.

The SQL Server will be checked to verify that your data was successfully copied into your **ist722_yournetid_dw** database.