DAT217x

Implementing ETL with SSIS

Module 4 | Lab 3 | ETL with SSIS Data Flows

We are now onto the third lab of the course. This lab focuses on how to create an SSIS ETL package using Data Flow Tasks.

■ **Note:** This lab is standalone and can be completed separately from the previous lab.

Scenario

A local company is looking for consultants to help with the creation of a Business Intelligence solution. Currently you are one of few consultants they are interested in. To get a better understanding of your skill level with Microsoft's SSIS, they have asked you to create a prototype ETL solution using an SSIS Project.

This prototype is based on an example subset of Microsoft's AdventureWorks demonstration database. The IT team has provided both a source and destination database for you to work with.

Previously, they tested your skills with both SQL programing and SSIS, and you showed them how you could use Execute SQL tasks to efficiently create an ETL process. However, they are now asking for more examples of how you could create an ETL process using SSIS Data Flows.

Lab Overview

In this lab, you will create an SSIS project that will perform ETL processing. You will start by reviewing an existing ETL SQL script and then use this script to configure an SSIS package. Before starting this lab, you need to be familiar with the Module 4 | SSIS Data Flows content and have followed the instructions in the Setting up the Lab Environment section at the start of this course.

What You Will Need

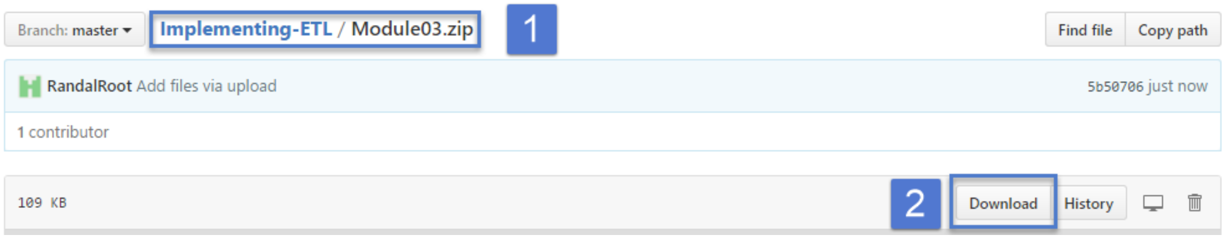
- A personal or virtual computer with the SQL Server 2017 (or 2016, 2014 or 2012) installed on it
- Permissions to download files on the computer
- Permissions to create a new folders and files on the computer
- Permissions to create databases in the SQL Server instance
- Several Course files: https://github.com/MicrosoftLearning/Implementing-ETL (External Link)

Getting Started

To complete the labs, you must download and extract the lab resources.

1. Open a **web browser** and navigate to https://github.com/MicrosoftLearning/Implementing-ETL(This is an external link that opens in a new window.)

2. Download the course resources, by clicking on the ***Implementing-ETL/Module04.zip file***, and then clicking the Download button to save the downloaded file to your computer.



Branch: master ▾ Implementing-ETL / Module03.zip

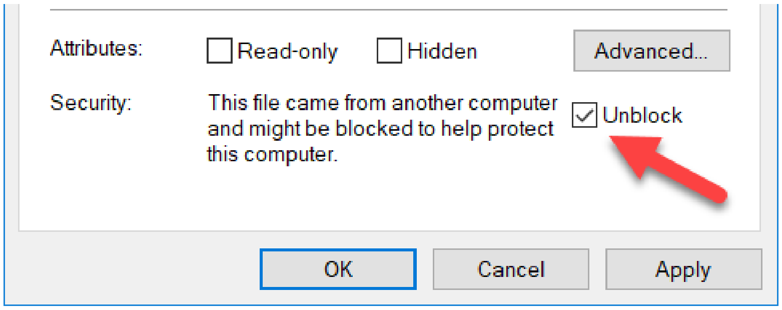
RandalRoot Add files via upload 5b50706 just now

1 contributor

109 KB

3. Locate the **Module04.zip** file, then Right-Click on it and select **Properties** from the context menu.

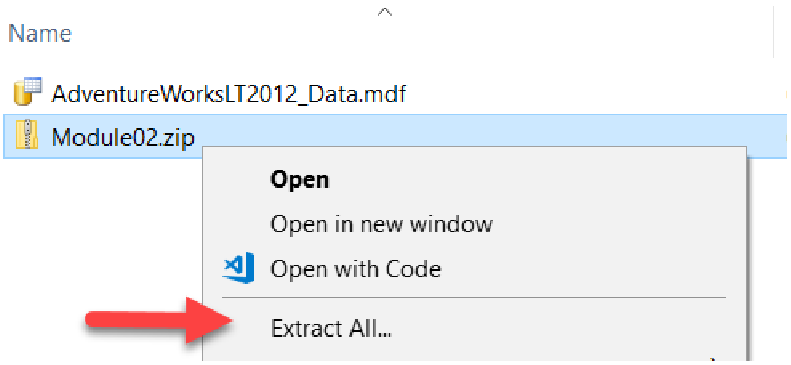
4. In the dialog window, check **Unblock**, then click **OK** to close. (This may be an Unblock button on some versions of Windows.)



Attributes: ☐ Read-only ☐ Hidden

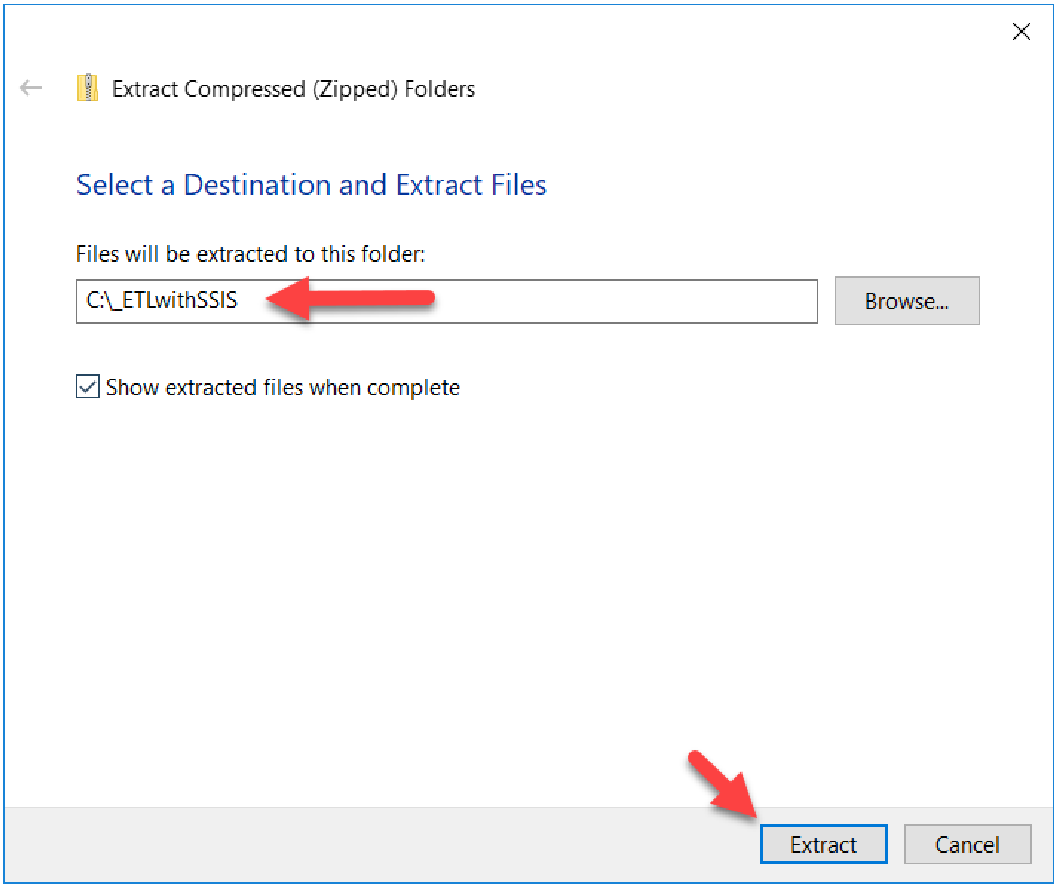
Security: This file came from another computer and might be blocked to help protect this computer. ☑ Unblock

5. Extract the file's content, by right-clicking the **Module04.zip** file, and selecting **Extract All**.



6. In the dialog window, replace the folder **path** to C:_ETLwithSSIS.

■ **Important:** Some lab scripts use absolute file paths, please extract and use the lab resources directly from the C:_ETLwithSSIS path. This lab requires a lot of SQL programming; therefore, the answer code is supplied with this lab in case you need help.



Select a Destination and Extract Files

Files will be extracted to this folder:

C:_ETLwithSSIS

☑ Show extracted files when complete

7. Click Extract. This will create a new **folder** (as needed) and unzipped the **module04.zip** file.

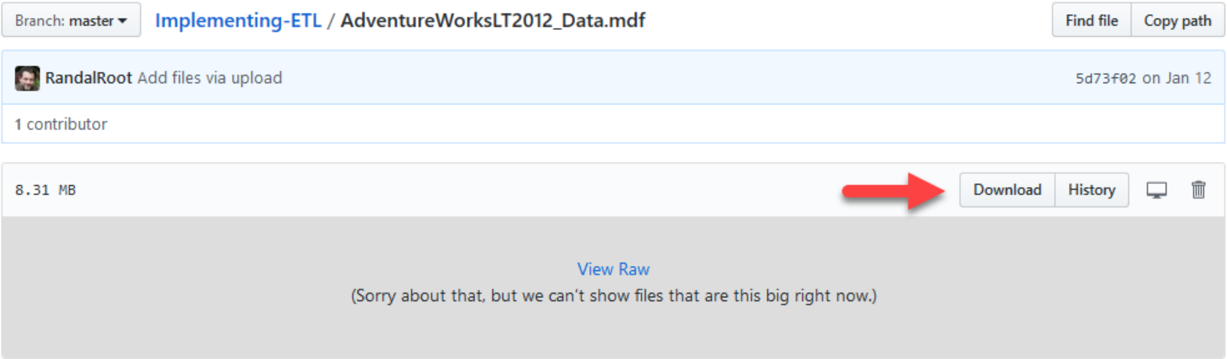
8. Navigate to the C:_ETLwithSSIS **folder**.

■ **Note:** Each lab in this course is standalone and can be completed individually.

Exercise 1 | Attaching the Source Database

Note: If you have already completed this from watching the Creating the Example Databases course video, or in a previous lab, you can skip to Exercise 2.

1. Download and **SQL Server AdventureWorks LT 2012 database file** from **the course GitHub page**. The database file name is **AdvntureWorksLT2012_Data.mdf.** This file can be found at this URL: https://github.com/MicrosoftLearning/Implementing-ETL

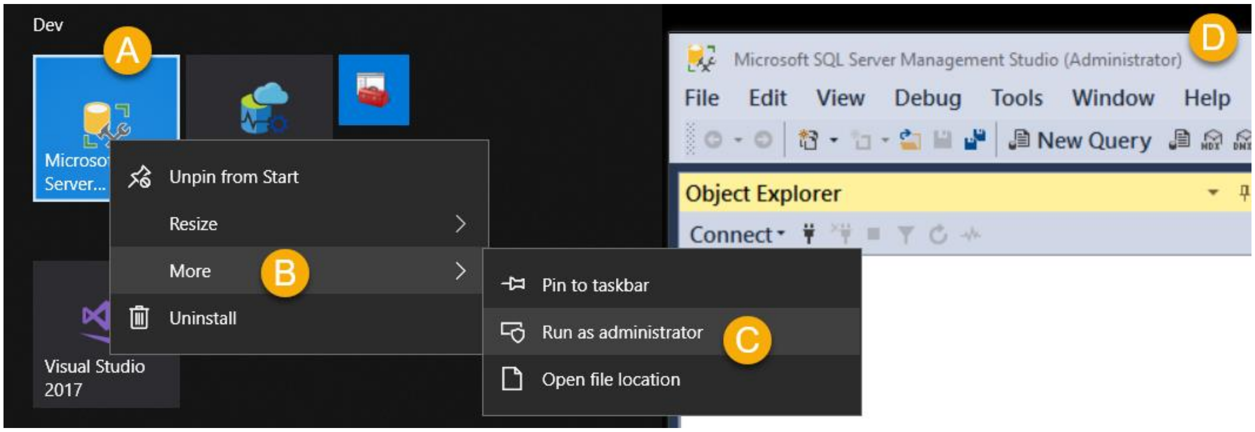

Branch: master
Implementing-ETL / AdventureWorksLT2012_Data.mdf
Find file
Copy path
RandalRoot Add files via upload
5d73f02 on Jan 12
1 contributor
8.31 MB
Download
History
View Raw
(Sorry about that, but we can't show files that are this big right now.)

2. Locate the **AdvntureWorksLT2012_Data.mdf** file, then Right-Click on it and select **Properties** from the context menu.

3. In the dialog window, check **Unblock**, then click **OK** to close. (This may be an Unblock button on some versions of Windows.)

4. Copy the **AdventureWorksLT2012_Data.mdf** file to the **C:_ETLWithSSIS** folder.

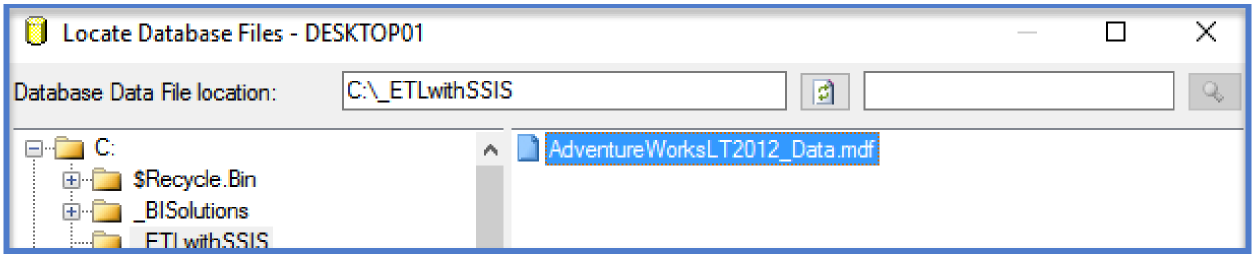
5. Start **SQL Server Management Studio**, using the **Run as Administrator** option, and connect to the **Database Engine** instance.



6. Right-click **Databases** in the **Object Explorer Tree View** window, then click **Attach** in the Context Menu.

7. In the Attach Database Dialog Window, Click the **Add** button.

8. Select the **AdventureWorksLT2012_Data.mdf** database file and click **OK**. If the file is not listed, check the folder to be sure the file is there.



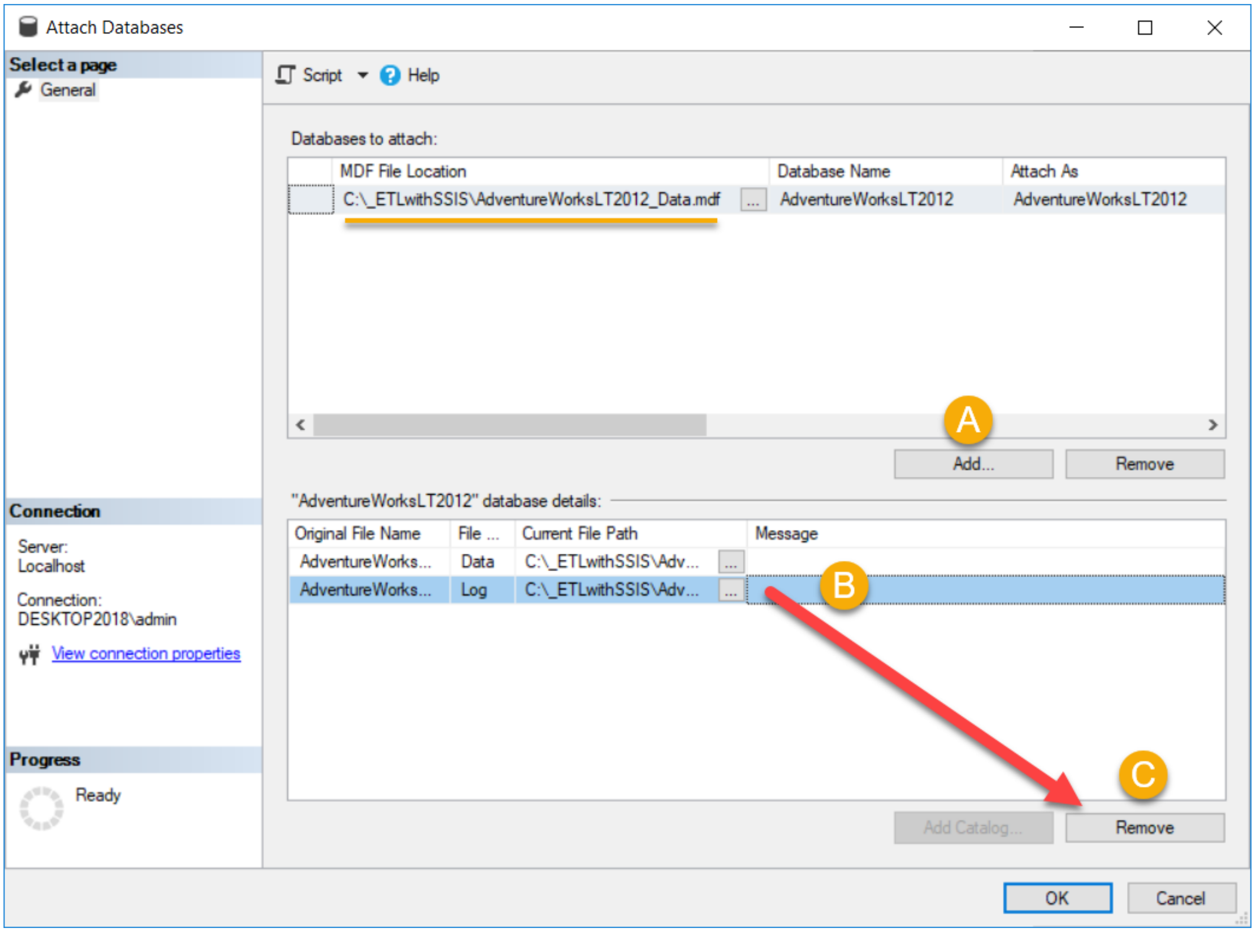
Locate Database Files - DESKTOP01

Database Data File location: C:_ETLwithSSIS

- C:
  - $Recycle.Bin
  - _BISolutions
  - ETLwithSSIS

AdventureWorksLT2012_Data.mdf

9. In database details, Click the **Remove** button to remove the **Log file entry**. The setup program assumes you have a log file, but there is no log file in the sample. A new log file will be created automatically when you attach the database.



Databases to attach:

MDF File Location	Database Name	Attach As
C:_ETLwithSSIS\AdventureWorksLT2012_Data.mdf	AdventureWorksLT2012	AdventureWorksLT2012

"AdventureWorksLT2012" database details:

Original File Name	File ...	Current File Path	Message
AdventureWorks...	Data	C:_ETLwithSSIS\Adv...	
AdventureWorks...	Log	C:_ETLwithSSIS\Adv...	

Connection

Server:
Localhost

Connection:
DESKTOP2018\admin

Progress

Ready

10. Click **OK** to attach just the primary database file.

Exercise 2 | Creating the Destination Database

1. Start **SQL Server Management Studio** and connect to the Database Engine instance.

2. Use the **File > Open > File** menu to open the Open File dialog window.

3. Locate and open the C:_ETLwithSSIS\Module04\Labs\Create the DWAdventureWorksLTLab03 database.sql file.

4. Review the **SQL Code** and noting how the ETL views and stored procedures have been removed.

5. Use the **[Execute]** button on the toolbar to run all the code in the file.

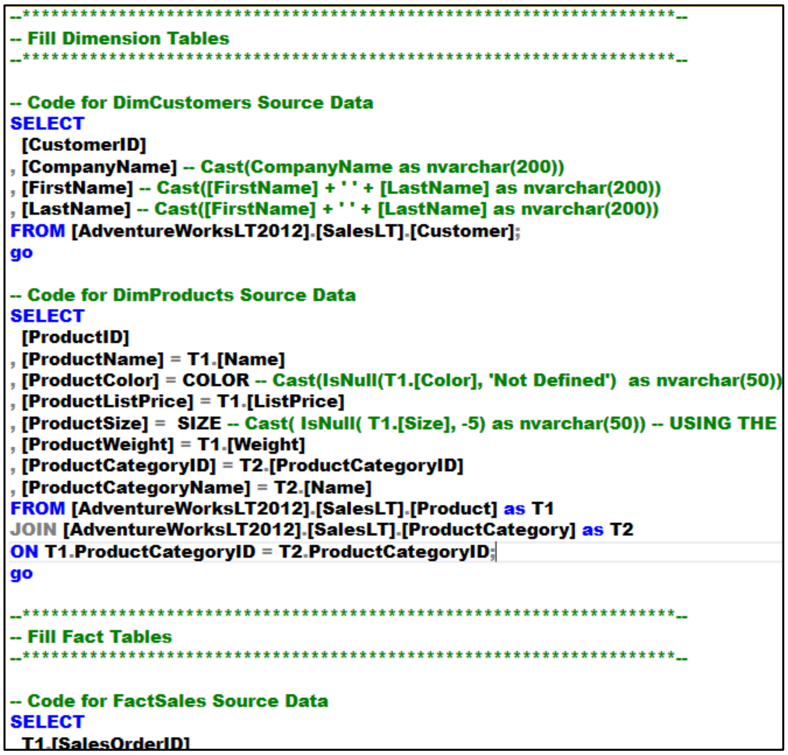
6. Refresh the **Object Explorer** Tree view and verify that the database was created.

Exercise 3 | Review the Existing ETL SQL Script

1. Start **SQL Server Management Studio** and connect to the Database Engine instance.

2. Use the **File > Open > File** Menu to open the **C:_ETLwithSSIS\Module04\Labs\ETL Code for the DWAdventureWorksLTLab03 database.sql** file.

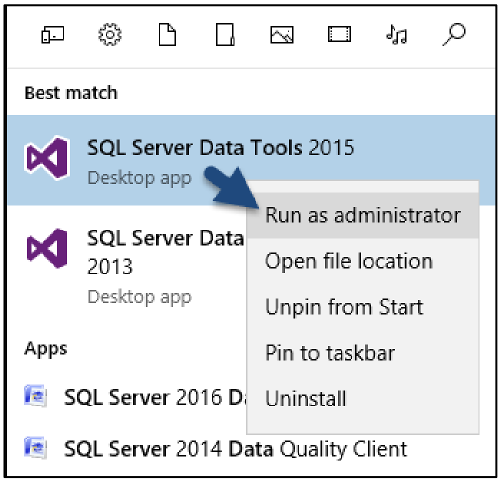
3. Review the **SQL code**, noting how the ETL transformations have been removed, but comments were added to indicate what is needed in your SSIS Data Flows.



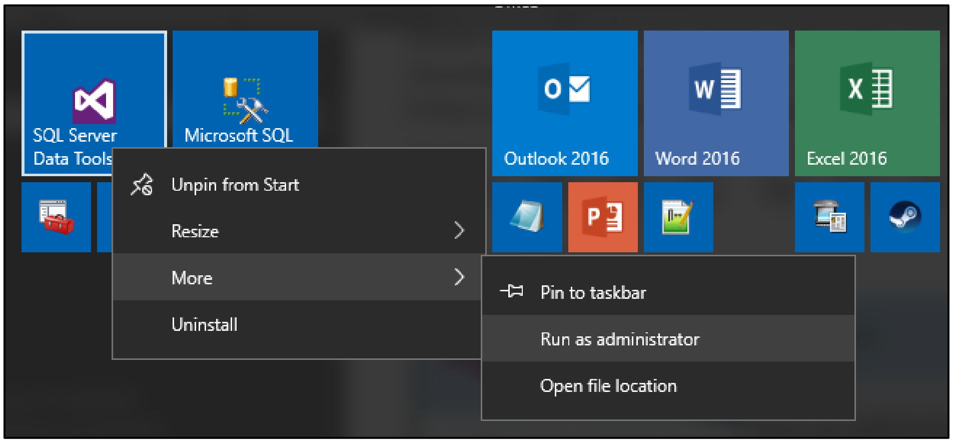
```
--**********************************************************************--
-- Fill Dimension Tables
--**********************************************************************--

-- Code for DimCustomers Source Data
SELECT
 [CustomerID]
, [CompanyName] -- Cast(CompanyName as nvarchar(200))
, [FirstName] -- Cast([FirstName] + ' ' + [LastName] as nvarchar(200))
, [LastName] -- Cast([FirstName] + ' ' + [LastName] as nvarchar(200))
FROM [AdventureWorksLT2012].[SalesLT].[Customer];
go

-- Code for DimProducts Source Data
SELECT
 [ProductID]
, [ProductName] = T1.[Name]
, [ProductColor] = COLOR -- Cast(IsNull(T1.[Color], 'Not Defined')  as nvarchar(50))
, [ProductListPrice] = T1.[ListPrice]
, [ProductSize] =  SIZE -- Cast( IsNull( T1.[Size], -5) as nvarchar(50)) -- USING THE
, [ProductWeight] = T1.[Weight]
, [ProductCategoryID] = T2.[ProductCategoryID]
, [ProductCategoryName] = T2.[Name]
FROM [AdventureWorksLT2012].[SalesLT].[Product] as T1
JOIN [AdventureWorksLT2012].[SalesLT].[ProductCategory] as T2
ON T1.ProductCategoryID = T2.ProductCategoryID;
go

--**********************************************************************--
-- Fill Fact Tables
--**********************************************************************--

-- Code for FactSales Source Data
SELECT
 T1.[SalesOrderID]
```

Exercise 4 | Create an SSIS Project

1. Using the **Window menu** search feature, locate and open the **SQL Server Data Tools 2015** desktop application, by right-clicking its icon and choosing the **Run as administrator** option.



Or if using the **Window Tile** UI option, choosing the **More > Run as administrator** option.

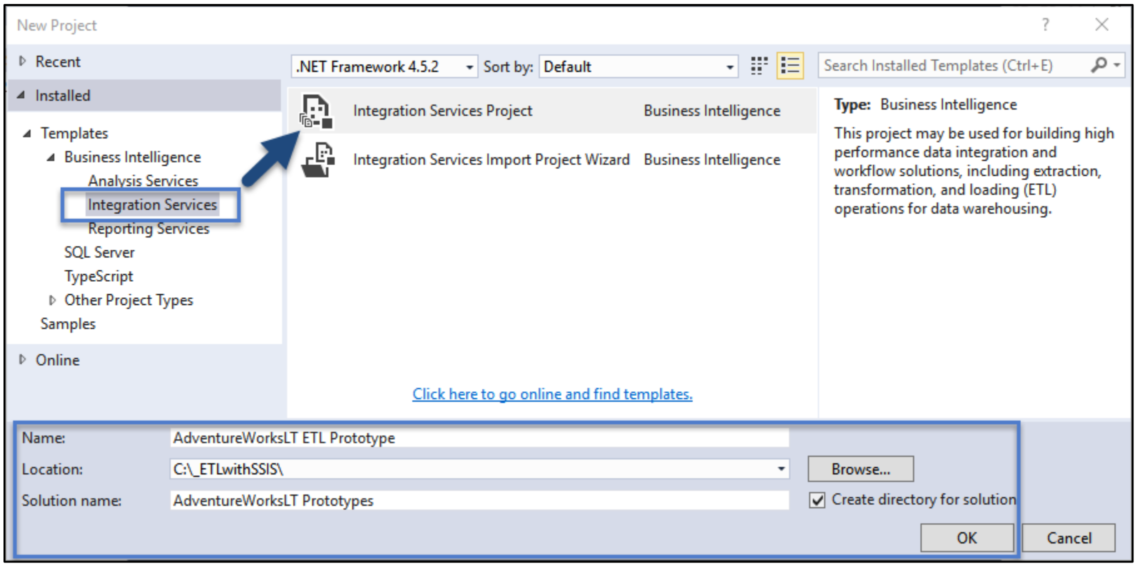


■ **NOTE:** You can use an earlier versions of the SQL Server Data Tools, if that is all you currently have installed, but the images will look a bit different.

2. When **Visual Studio** opens, create a new Integration Services project using **the File > New > Project** to menu item.

   In the **New Project dialog** box, expand the Business Intelligence node under Installed Templates, and select **Integration Services Project** in the Templates pane.

3. In the Name box, change the default name to **AdventureWorksLT ETL Prototype**. Change the default location to the **C:_ETLwithSSIS\Module04** folder. Change the Solution name to **AdventureWorksLT Prototypes**. Then click OK to create the new SSIS project.



Type: Business Intelligence

This project may be used for building high performance data integration and workflow solutions, including extraction, transformation, and loading (ETL) operations for data warehousing.

Name: AdventureWorksLT ETL Prototype

Location: C:_ETLwithSSIS\

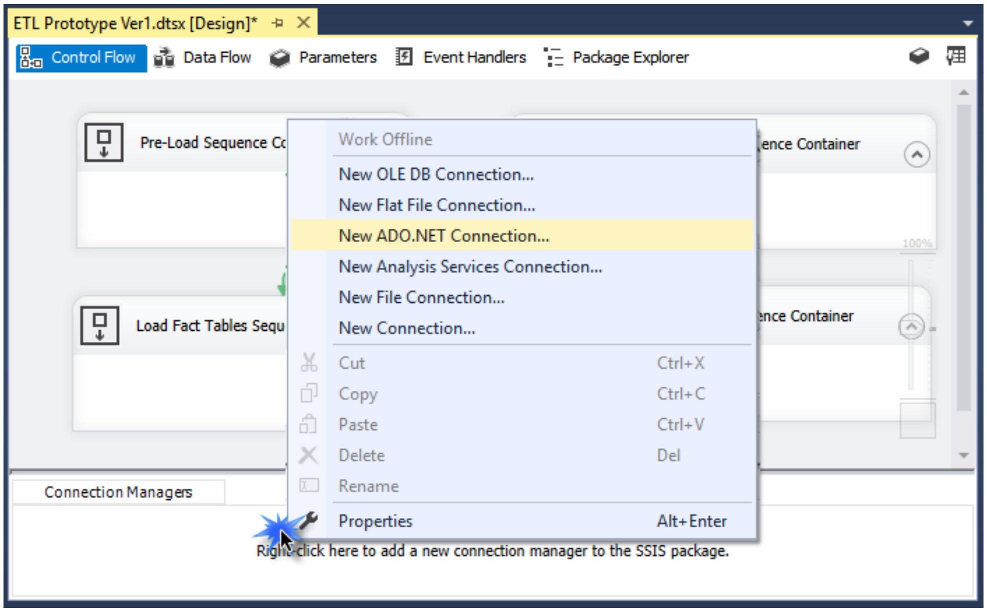
Solution name: AdventureWorksLT Prototypes

☑ Create directory for solution

By default, an empty package, titled **Package.dtsx**, will be created and added to your project under SSIS Packages.

Exercise 5 | Configure an SSIS Package

1. In Solution Explorer window, right-click **Package.dtsx**, click Rename, and rename the default package to **ETL Prototype Ver1 with Data Flows.dtsx**.
2. Add four **Sequence Containers** to the packages designer surface, rename each of them using the following list:
   - Pre-Load Sequence Container
   - Load Dimension Tables Sequence Container
   - Load Fact Tables Sequence Container
   - Post-Load Sequence Container
3. Link them together using **Precedent Constraints** in the same order as the named list. Your package should look like the following image when this step is completed.
4. Add a connection to the **DWAdventureWorksLT2012Lab03** database using an ADO.NET Connection.
5. Right-click anywhere in the **Connection Managers area**, and then click **New ADO.NET Connection**.



Right-click here to add a new connection manager to the SSIS package.

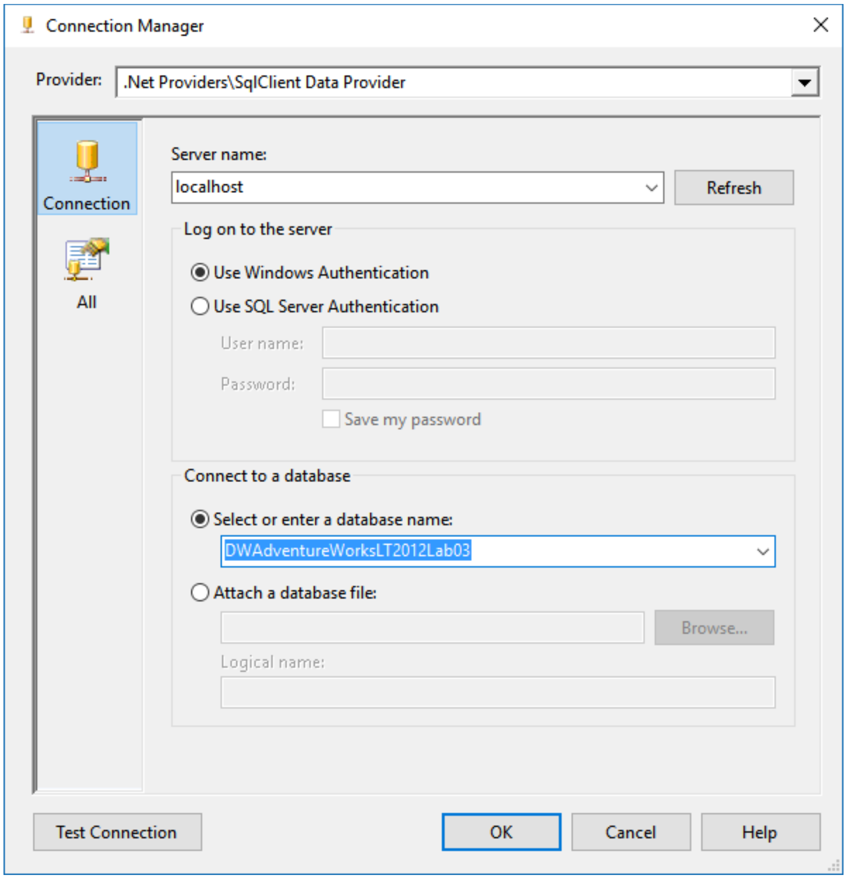
6. In the Configure ADO.NET Connection Manager dialog box, click New.
7. For Server name, enter ***localhost***.

■ **NOTE:** When you specify localhost as the server name, the connection manager connects to the default instance of SQL Server on the local computer. To use a remote instance of SQL Server, replace localhost with the name of the server to which you want to connect. To use a named instance of SQL Server, add a back-slash and the name of the instance (localhost\MyNamedInstance).

8. In the “Log on to the server” group, verify that **Use Windows Authentication** is selected.

9. In the Connect to a database group, in the Select or enter a database name box, type or select **DWAdventureWorksLT2012Lab03**.

10. Click **OK**, again to create the SSIS connection.



Connection Manager

Provider: .Net Providers\SqlClient Data Provider

Server name: localhost

Log on to the server

● Use Windows Authentication

○ Use SQL Server Authentication

User name:

Password:

□ Save my password

Connect to a database

● Select or enter a database name: DWAdventureWorksLT2012Lab03

○ Attach a database file:

Logical name:

Exercise 6 | Configure the Pre-Load Sequence Container

1. From the **SSIS Toolbox**, drag an Execute SQL Task into the **Pre-Load Sequence Container** on Control Flow design surface.

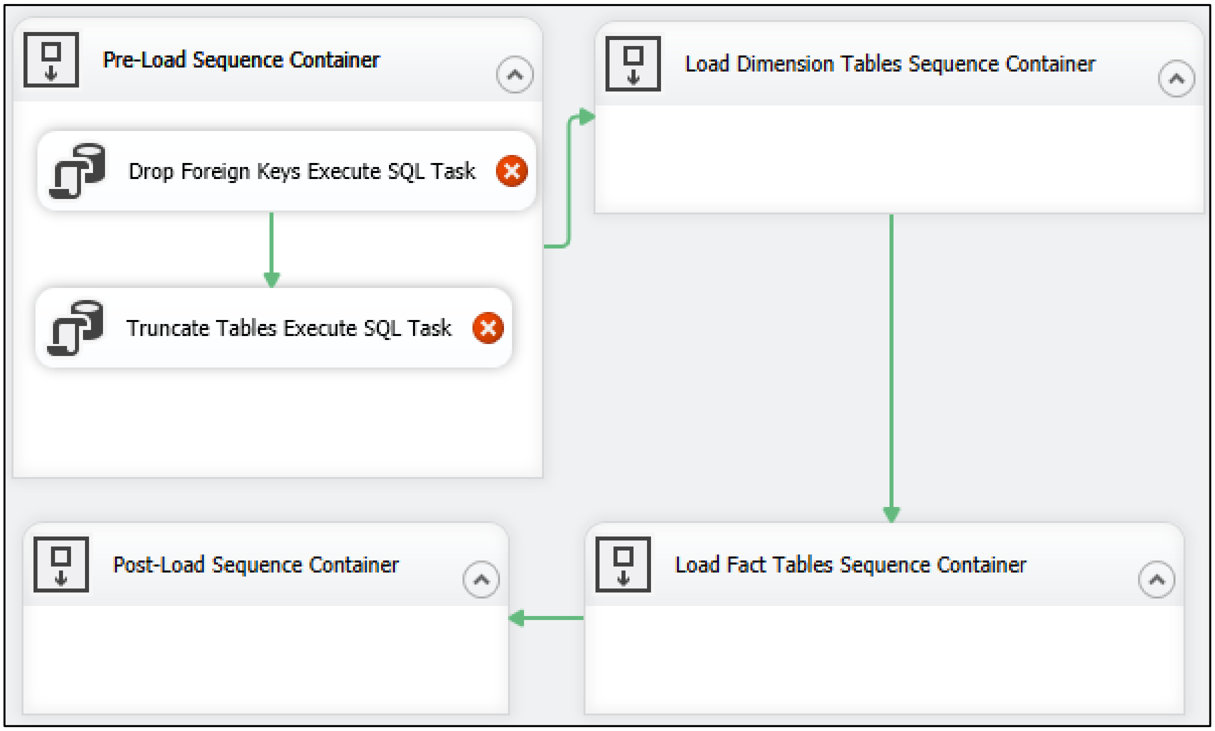
2. Right-click Execute SQL Task and rename it to Drop Foreign Key Constraints Execute SQL Task.

3. From the **SSIS Toolbox**, drag another Execute SQL Task into the **Pre-Load Sequence Container** on Control Flow design surface.

4. Right-click Execute SQL Task and rename it to Truncate Tables Execute SQL Task.

5. Click the **Drop Foreign Key Constraints Execute SQL Task** and drag the green arrow that appears onto the newly added **Truncate Tables Execute SQL Task** to connect the two components.

Completing these steps will make the SSIS Package look *similar* to the following image:


Pre-Load Sequence Container
Drop Foreign Keys Execute SQL Task
Truncate Tables Execute SQL Task
Load Dimension Tables Sequence Container
Post-Load Sequence Container
Load Fact Tables Sequence Container

6. Start **SQL Server Management Studio** and connect to the Database Engine instance.

7. Use the File > Open > File Menu to open the C:_ETLwithSSIS\Module04\Labs\ETL Code for the DWAdventureWorksLTLab03 database.sql file.

8. Locate the **SQL code** that drops all of the foreign key constraints and copy it.

9. Right-click the **Drop Foreign Key Constraints Execute SQL Task** and choose Edit from its context menu.

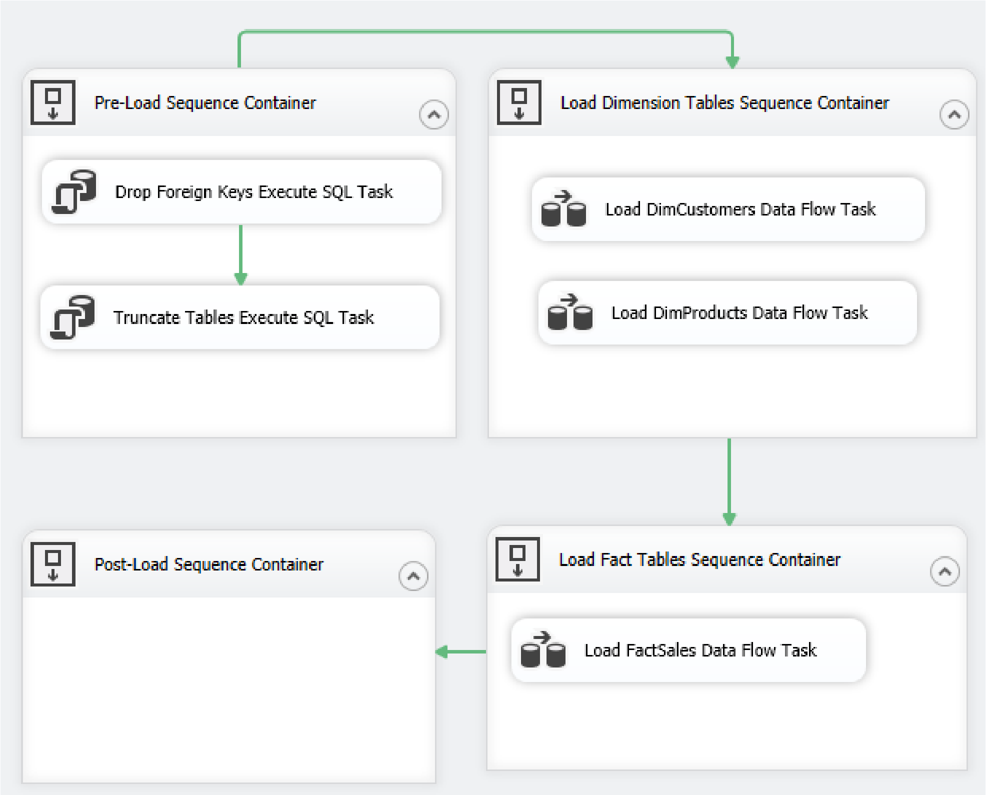
10. Set the ***ConnectionType* property** to the ADO.NET, the ***Connection* property** to the SSIS ADO.NET connection you created earlier, and paste the SQL code you copied into the ***SQLStatement* property**.
11. Close the Execute SQL Task Editor.
12. Locate the **SQL code** that truncates all of the tables and copy it.
13. Right-click the **Truncate Tables Execute SQL Task** and choose Edit from its context menu.
14. Set the ***ConnectionType* property** to the ADO.NET, the ***Connection* property** to the SSIS ADO.NET connection you created earlier, and paste the SQL code you copied into the ***SQLStatement* property**.
15. Right-click the **Pre-Load Sequence Container**, and select **Execute Container** from the context.
16. Verify that all the tasks run successfully or troubleshoot why not. Remember, if you need to change settings in the task, you must first stop the **debugging engine**.

■ **NOTE:** If you keep getting errors, try resetting the database to its normal empty state by running the SQL code in the **C:_ETLwithSSIS\Module04\Labs\Create the DWAdventureWorksLTLab03 database.sql** file. This code re-creates the database and makes it ready for the ETL process.

Exercise 7 | Configure the Load Dimensions Tables and Load Fact Tables Sequence Containers

1. From the **SSIS Toolbox**, drag a Data Flow Task into the **Load Dimension Tables Sequence Container** on Control Flow design surface.
2. Right-click the Data Flow Task and rename it to **Load DimCustomers Data Flow Task**.
3. From the **SSIS Toolbox**, drag another Data Flow Task into the **Load Dimension Tables Sequence Container** on Control Flow design surface.
4. Right-click the Data Flow Task and rename it to **Load DimProducts Data Flow Task**.
5. From the **SSIS Toolbox**, drag a Data Flow Task into the **Load Fact Tables Sequence Container** on Control Flow design surface.
6. Right-click the Data Flow Task and rename it to **Load FactSales Data Flow Task**.

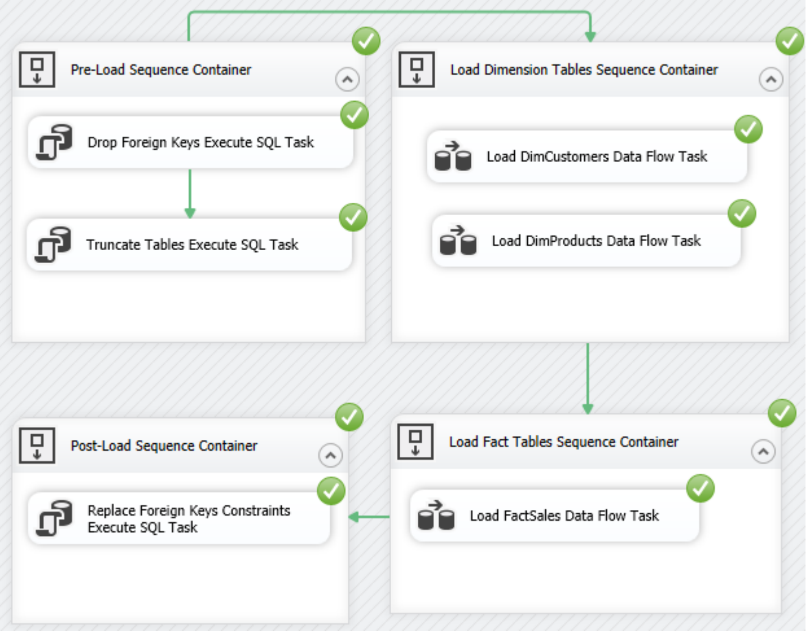
Completing these steps will make the SSIS Package look *similar* to this:


Pre-Load Sequence Container
Drop Foreign Keys Execute SQL Task
Truncate Tables Execute SQL Task
Load Dimension Tables Sequence Container
Load DimCustomers Data Flow Task
Load DimProducts Data Flow Task
Post-Load Sequence Container
Load Fact Tables Sequence Container
Load FactSales Data Flow Task

<span style="color:red">**Note:**</span> **We will configure the Data Flow tasks later in this lab.**

Exercise 8 | Configure the Post-Load Sequence Container

1. From the SSIS Toolbox, drag an Execute SQL Task into the **Post-Load Sequence Container** on Control Flow design surface.
2. Right-click the task and rename it to Replace Foreign Keys Constraints Execute SQL Task.
3. Start **SQL Server Management Studio** and connect to the Database Engine instance.
4. Use the File > Open > File Menu to open the C:_ETLwithSSIS\Module04\Labs\ETL Code for the DWAdventureWorksLTLab03 database.sql file.
5. Locate the SQL code that re-creates all of the foreign key constraints and copy it.
6. Right-click the Replace Foreign Key Constraints Execute SQL Task and choose Edit from its context menu.
7. Set the ***ConnectionType* property** to the ADO.NET, the ***Connection* property** to the SSIS ADO.NET connection you created earlier, and paste the SQL code you copied into the ***SQLStatement* property**.
8. Close the Execute SQL Task Editor.
9. Right-click the **ETL Prototype Ver1 with Data Flows.dtsx** package file and select **Execute Package** from the context menu.
10. Verify that all the tasks have run successfully or troubleshoot any issues. Remember, if you need to change settings in the task, you must first stop the **debugging engine.**


Pre-Load Sequence Container
Drop Foreign Keys Execute SQL Task
Truncate Tables Execute SQL Task
Load Dimension Tables Sequence Container
Load DimCustomers Data Flow Task
Load DimProducts Data Flow Task
Post-Load Sequence Container
Replace Foreign Keys Constraints Execute SQL Task
Load Fact Tables Sequence Container
Load FactSales Data Flow Task

Note: Empty Data Flow tasks run successfully in SSIS even when they do not have data sources, transformations, or destination components.

Exercise 9 | Configure the Load DimCustomers Data Flow Task

1. Right-click the **Load DimCustomers Data Flow Task** and choose **Edit** from its context menu.

2. From the SSIS Toolbox, drag an **ADO NET Source** into the Data Flow design surface.

3. Right-click the source and rename it to **Customers Data ADO NET Source.**

4. Right-click the source and click **Edit** in the context menu**.**

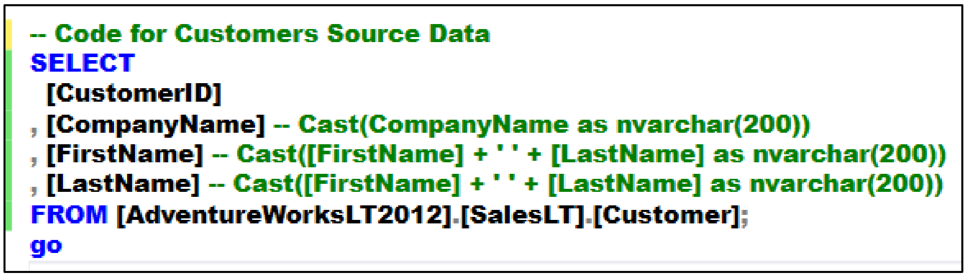
5. Select the ADO.NET connection you created in Exercise 5 in the **ADO.NET connection manager dropdown**.

6. Select the SQL Command option in the **Data access mode dropdown**.

7. Start **SQL Server Management Studio** and connect to the Database Engine instance.

8. Use the File > Open > File Menu to open the C:_ETLwithSSIS\Module04\Labs\ETL Code for the DWAdventureWorksLTLab03 database.sql file.

9. Locate the SQL code that selects data from the **Customers** table and copy it.



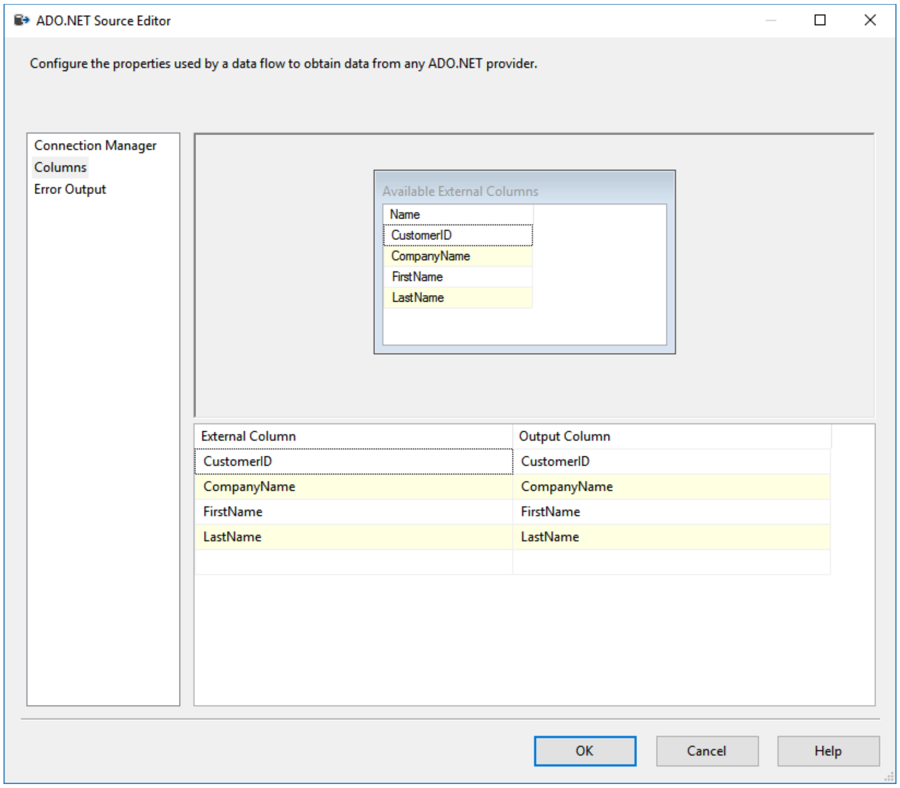
```
-- Code for Customers Source Data
SELECT
 [CustomerID]
, [CompanyName] -- Cast(CompanyName as nvarchar(200))
, [FirstName] -- Cast([FirstName] + ' ' + [LastName] as nvarchar(200))
, [LastName] -- Cast([FirstName] + ' ' + [LastName] as nvarchar(200))
FROM [AdventureWorksLT2012].[SalesLT].[Customer];
go
```

10. Paste the copied code into the **SQL command text textbox**.

Note: Make sure you do not copy the “go” keyword.

11. Click the **Preview** button and verify the code returns data or troubleshoot any issues.

12. Click the **Columns** page in the left-hand navigation area and verify the columns are listed as shown here:



Configure the properties used by a data flow to obtain data from any ADO.NET provider.

Name
CustomerID
CompanyName
FirstName
LastName

External Column	Output Column
CustomerID	CustomerID
CompanyName	CompanyName
FirstName	FirstName
LastName	LastName

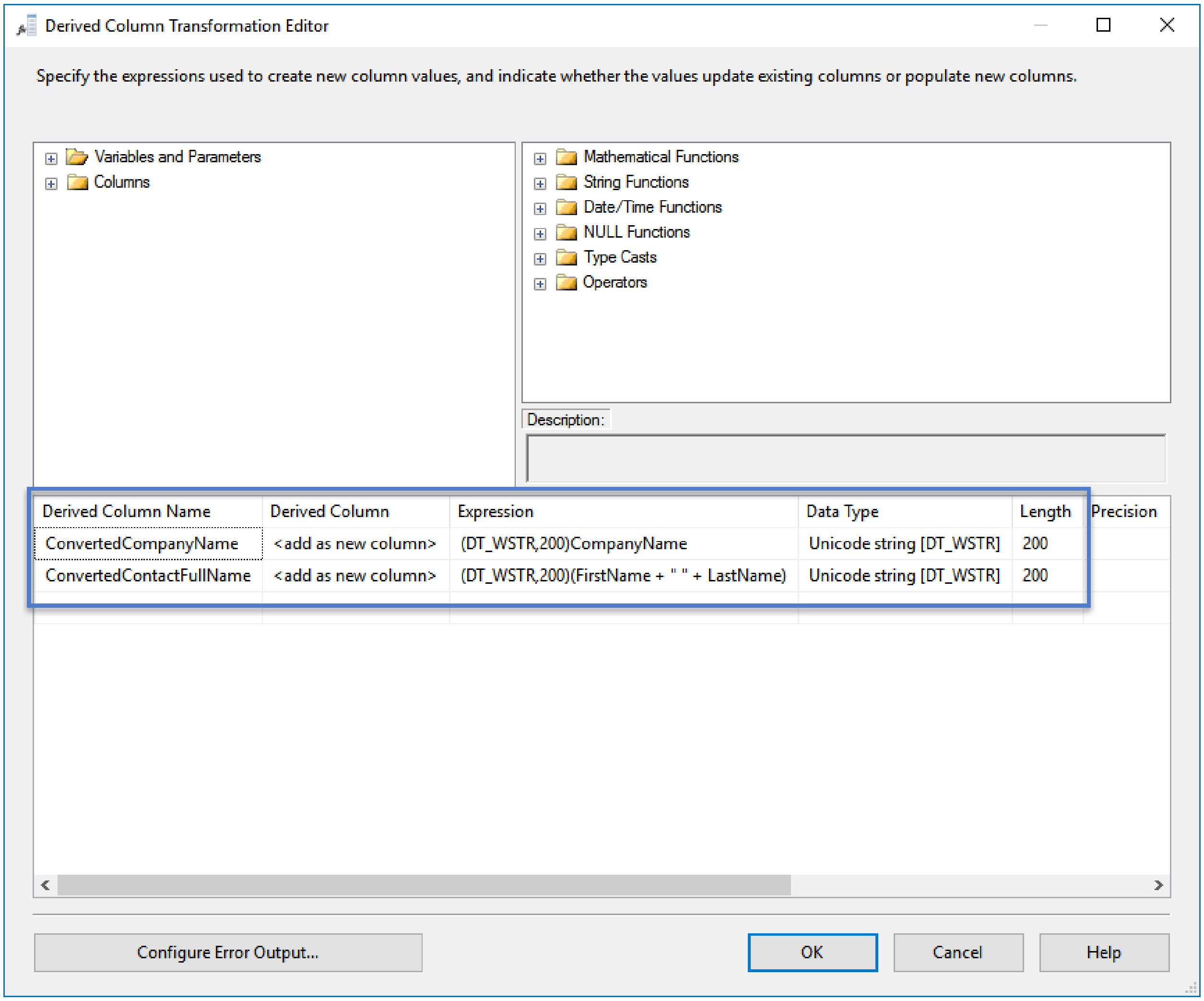
13. Click **OK** to close the dialog window.

14. From the SSIS Toolbox, drag a **Derived Column transformation** into the Data Flow design surface.

15. Right-click the **transformation** and rename it to Derived Column Transformations.

16. Right-click the **transformation** and click Edit in the context menu.

17. In the **Derived Column Transformation Editor**, configure two derived columns as shown here:



Derived Column Transformation Editor
Specify the expressions used to create new column values, and indicate whether the values update existing columns or populate new columns.
Variables and Parameters
Columns
Mathematical Functions
String Functions
Date/Time Functions
NULL Functions
Type Casts
Operators
Description:
Configure Error Output...
OK
Cancel
Help

Derived Column Name	Derived Column	Expression	Data Type	Length	Precision
ConvertedCompanyName	<add as new column>	(DT_WSTR,200)CompanyName	Unicode string [DT_WSTR]	200	
ConvertedContactFullName	<add as new column>	(DT_WSTR,200)(FirstName + " " + LastName)	Unicode string [DT_WSTR]	200	

18. Click **OK** to close the dialog window.

19. From the SSIS Toolbox, drag an **ADO NET Destination** into the Data Flow design surface.

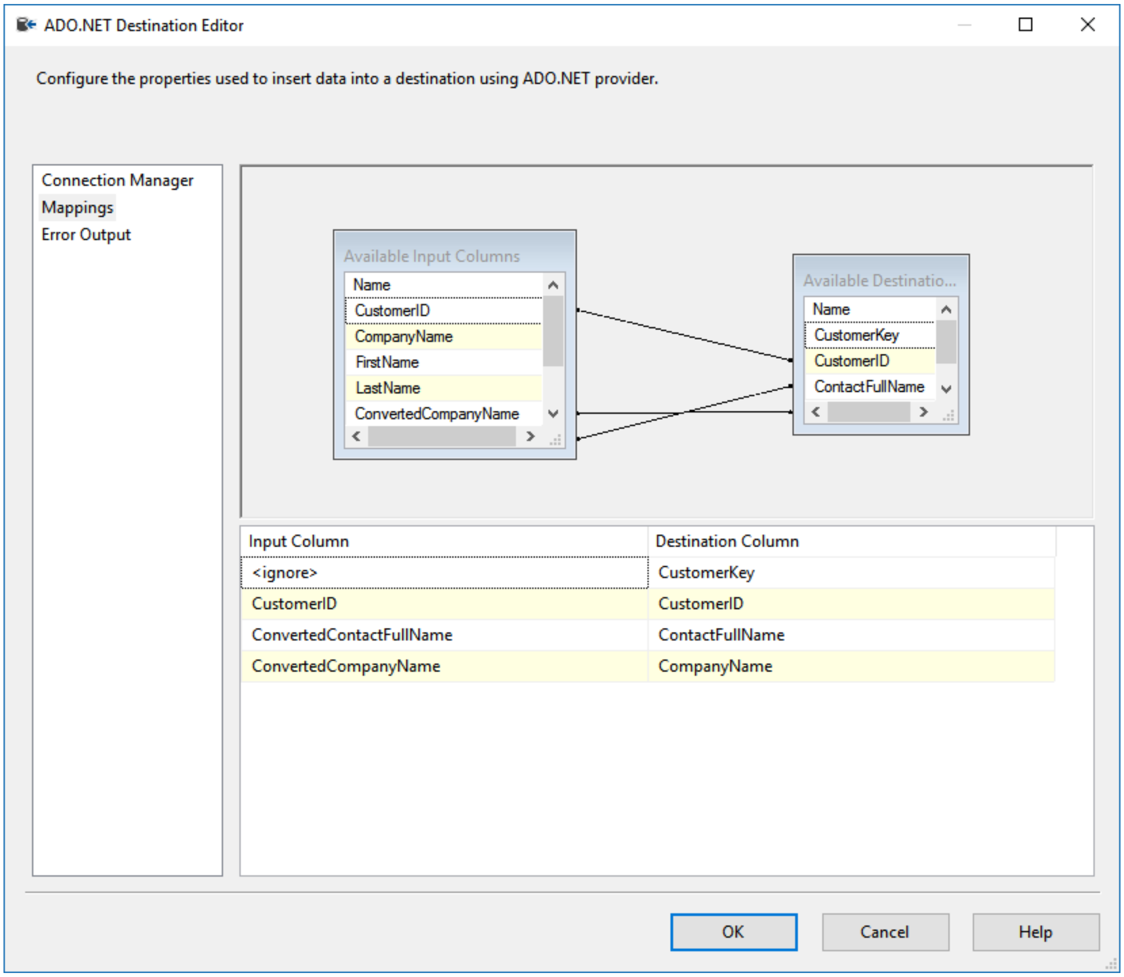
20. Right-click the source and rename it to **DimCustomers ADO NET Destination.**

21. Right-click the source and click **Edit** in the context menu**.**

22. Select the ADO.NET connection you created in Exercise 5 in the **ADO.NET connection manager dropdown**.

23. Select the “dbo”.”DimCustomers” table in the **Use a table or view dropdown**.

24. Click the **Mappings** page in the left-hand navigation area and verify the columns are mapped as shown here:



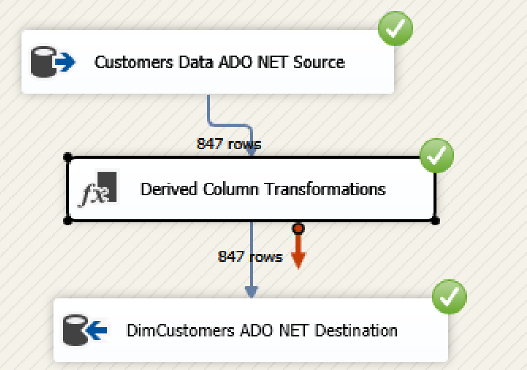
Configure the properties used to insert data into a destination using ADO.NET provider.

Input Column	Destination Column
<ignore>	CustomerKey
CustomerID	CustomerID
ConvertedContactFullName	ContactFullName
ConvertedCompanyName	CompanyName

25. Click **OK** to close the dialog window.

26. Right-click the **ETL Prototype Ver1 with Data Flows.dtsx** package file and select **Execute Package** from the context menu.

27. Verify that all the tasks have run successfully or troubleshoot any issues. Remember, if you need to change settings in the task, you must first stop the **debugging engine.**


Customers Data ADO NET Source
847 rows
Derived Column Transformations
847 rows
DimCustomers ADO NET Destination

Exercise 10 | Configure the Load DimProducts Data Flow Task

1. Choose the **Load DimProducts Data Flow Task** from the Data Flow Task dropdown on the Data Flow tab.

2. From the SSIS Toolbox, drag an **ADO NET Source** into the Data Flow design surface.

3. Right-click the source and rename it to Products and Categories Data ADO NET Source.

4. Right-click the source and click **Edit** in the context menu**.**

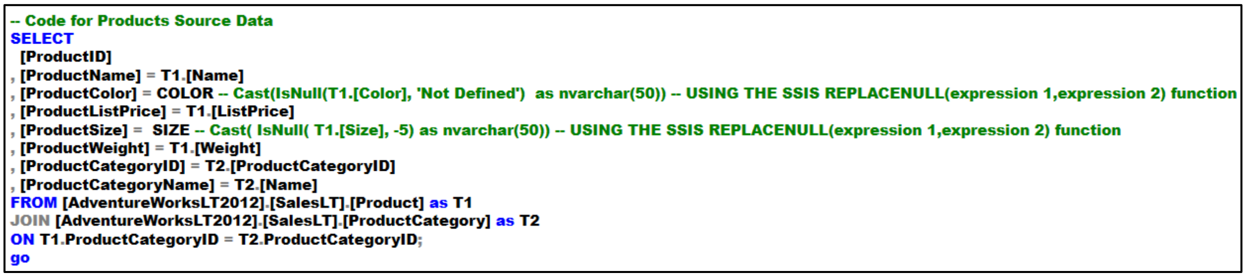
5. Select the ADO.NET connection you created in Exercise 5 in the **ADO.NET connection manager dropdown**.

6. Select the SQL Command option in the **Data access mode dropdown**.

7. Start **SQL Server Management Studio** and connect to the Database Engine instance.

8. Use the File > Open > File Menu to open the C:_ETLwithSSIS\Module04\Labs\ETL Code for the DWAdventureWorksLTLab03 database.sql file.

9. Locate the SQL code that selects data from the **Products and Categories** tables and copy it.



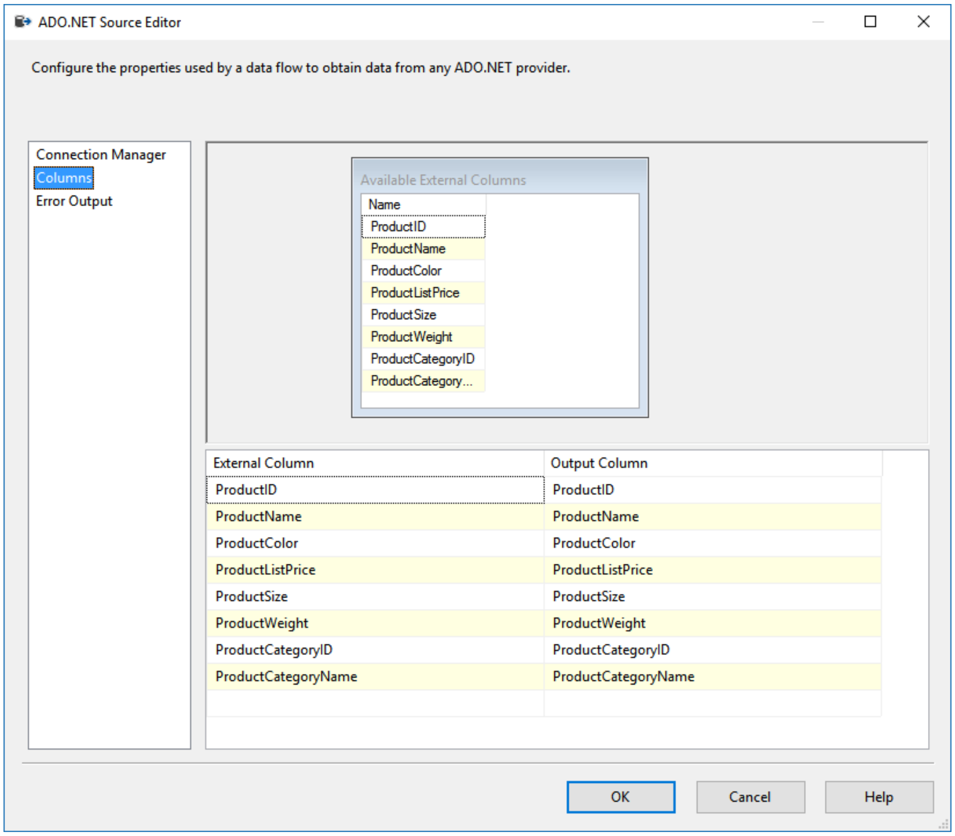
```sql
-- Code for Products Source Data
SELECT
  [ProductID]
, [ProductName] = T1.[Name]
, [ProductColor] = COLOR -- Cast(IsNull(T1.[Color], 'Not Defined')  as nvarchar(50)) -- USING THE SSIS REPLACENULL(expression 1,expression 2) function
, [ProductListPrice] = T1.[ListPrice]
, [ProductSize] =  SIZE -- Cast( IsNull( T1.[Size], -5) as nvarchar(50)) -- USING THE SSIS REPLACENULL(expression 1,expression 2) function
, [ProductWeight] = T1.[Weight]
, [ProductCategoryID] = T2.[ProductCategoryID]
, [ProductCategoryName] = T2.[Name]
FROM [AdventureWorksLT2012].[SalesLT].[Product] as T1
JOIN [AdventureWorksLT2012].[SalesLT].[ProductCategory] as T2
ON T1.ProductCategoryID = T2.ProductCategoryID;
go
```

10. Paste the copied code into the **SQL command text textbox**.

Note: Make sure you do not copy the "go" keyword.

11. Click the **Preview** button and verify the code returns data or troubleshoot any issues.

12. Click the **Columns** page in the left-hand navigation area and verify the columns are listed as shown here:



Configure the properties used by a data flow to obtain data from any ADO.NET provider.

Available External Columns

Name
ProductID
ProductName
ProductColor
ProductListPrice
ProductSize
ProductWeight
ProductCategoryID
ProductCategory...

External Column	Output Column
ProductID	ProductID
ProductName	ProductName
ProductColor	ProductColor
ProductListPrice	ProductListPrice
ProductSize	ProductSize
ProductWeight	ProductWeight
ProductCategoryID	ProductCategoryID
ProductCategoryName	ProductCategoryName

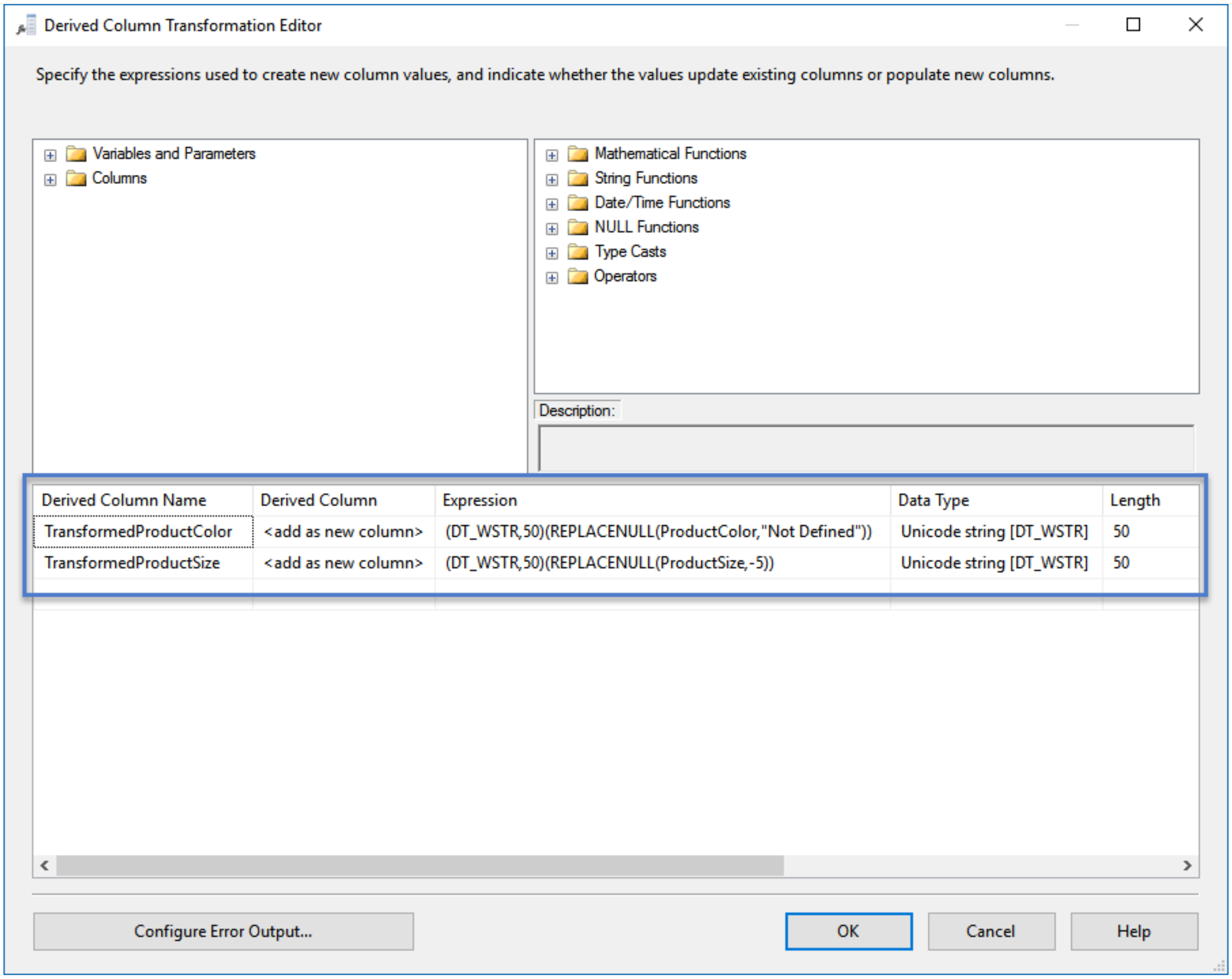
13. Click **OK** to close the dialog window.

14. From the SSIS Toolbox, drag a **Derived Column transformation** into the Data Flow design surface.

15. Right-click the **transformation** and rename it to Derived Column Transformations.

16. Right-click the **transformation** and click Edit in the context menu.

17. In the **Derived Column Transformation Editor**, configure two derived columns as shown here:



Derived Column Transformation Editor

Specify the expressions used to create new column values, and indicate whether the values update existing columns or populate new columns.

Derived Column Name	Derived Column	Expression	Data Type	Length
TransformedProductColor	<add as new column>	(DT_WSTR,50)(REPLACENULL(ProductColor,"Not Defined"))	Unicode string [DT_WSTR]	50
TransformedProductSize	<add as new column>	(DT_WSTR,50)(REPLACENULL(ProductSize,-5))	Unicode string [DT_WSTR]	50

18. Click **OK** to close the dialog window.

19. From the SSIS Toolbox, drag an **ADO NET Destination** into the Data Flow design surface.

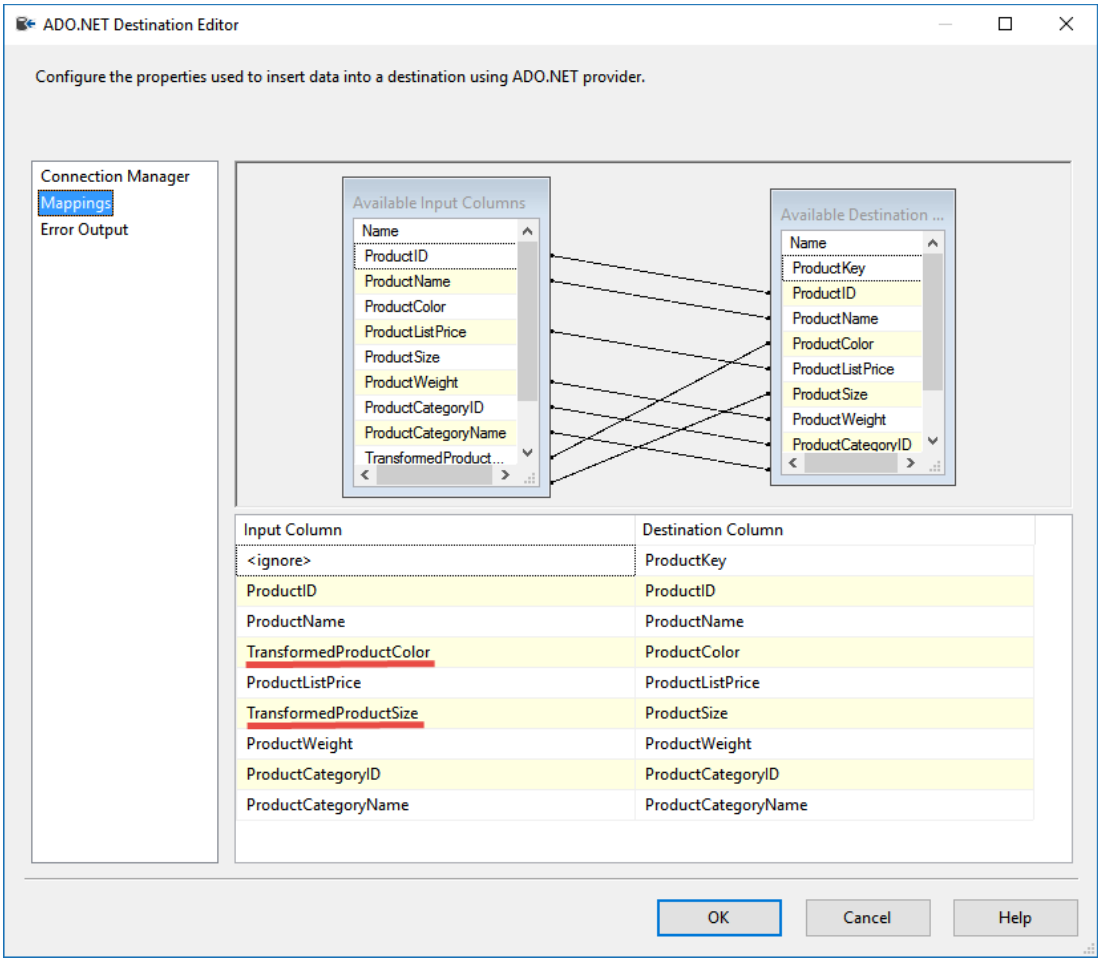
20. Right-click the source and rename it to **DimProducts ADO NET Destination.**

21. Right-click the source and click **Edit** in the context menu**.**

22. Select the ADO.NET connection you created in Exercise 5 in the **ADO.NET connection manager dropdown**.

23. Select the "dbo"."DimProducts" table in the **Use a table or view dropdown**.

24. Click the **Mappings** page in the left-hand navigation area and verify the columns are mapped as shown here:



ADO.NET Destination Editor

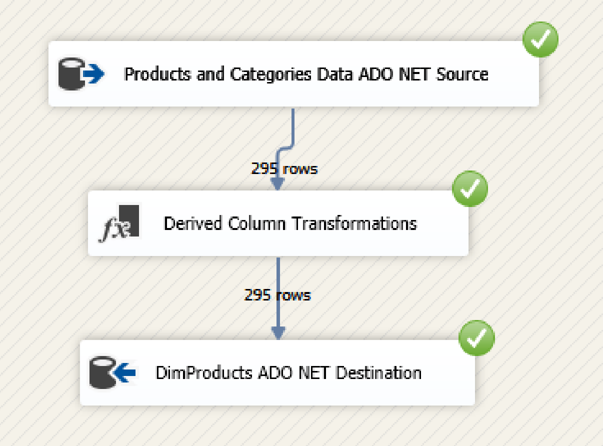
Configure the properties used to insert data into a destination using ADO.NET provider.

Input Column	Destination Column
<ignore>	ProductKey
ProductID	ProductID
ProductName	ProductName
TransformedProductColor	ProductColor
ProductListPrice	ProductListPrice
TransformedProductSize	ProductSize
ProductWeight	ProductWeight
ProductCategoryID	ProductCategoryID
ProductCategoryName	ProductCategoryName

25. Click **OK** to close the dialog window.

26. Right-click the **ETL Prototype Ver1 with Data Flows.dtsx** package file and select **Execute Package** from the context menu.

27. Verify that all the tasks have run successfully or troubleshoot any issues. Remember, if you need to change settings in the task, you must first stop the **debugging engine.**


Products and Categories Data ADO NET Source
295 rows
Derived Column Transformations
295 rows
DimProducts ADO NET Destination

Exercise 11 | Configure the Load Fact Sales Data Flow Task

1. Choose the **Load FactSales Data Flow Task** from the Data Flow Task dropdown on the Data Flow tab.

2. From the SSIS Toolbox, drag an **ADO NET Source** into the Data Flow design surface.

3. Right-click the source and rename it to SalesOrderHeader and SalesOrderDetail Data ADO NET Source.

4. Right-click the source and click **Edit** in the context menu**.**

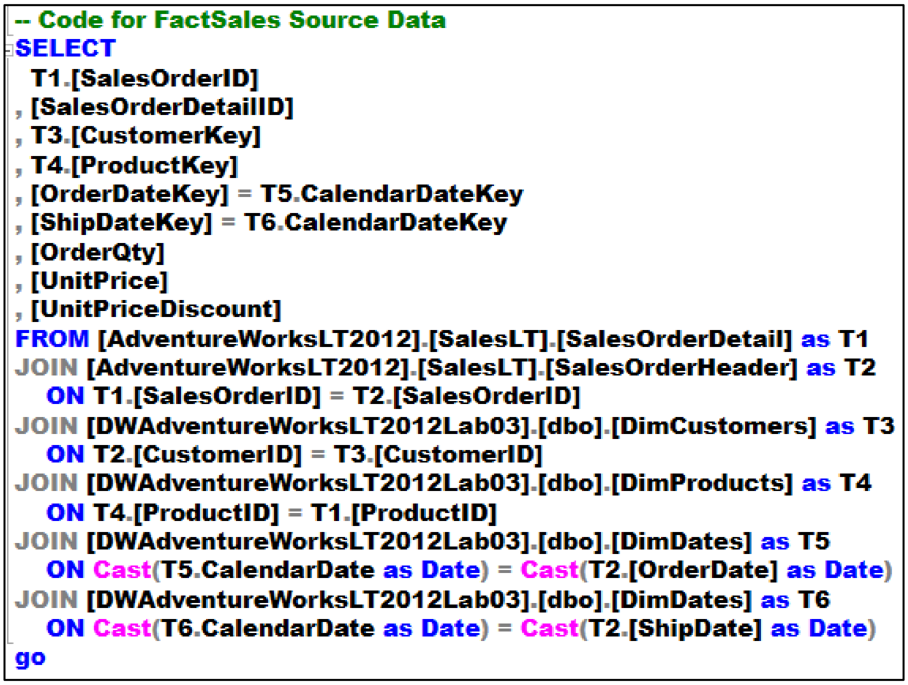
5. Select the ADO.NET connection you created in Exercise 5 in the **ADO.NET connection manager dropdown**.

6. Select the SQL Command option in the **Data access mode dropdown**.

7. Start **SQL Server Management Studio** and connect to the Database Engine instance

8. Use the File > Open > File Menu to open the C:_ETLwithSSIS\Module04\Labs\ETL Code for the DWAdventureWorksLTLab03 database.sql file.

9. Locate the SQL code that selects data from the **SalesOrderHeader and SalesOrderDetail** tables and copy it.



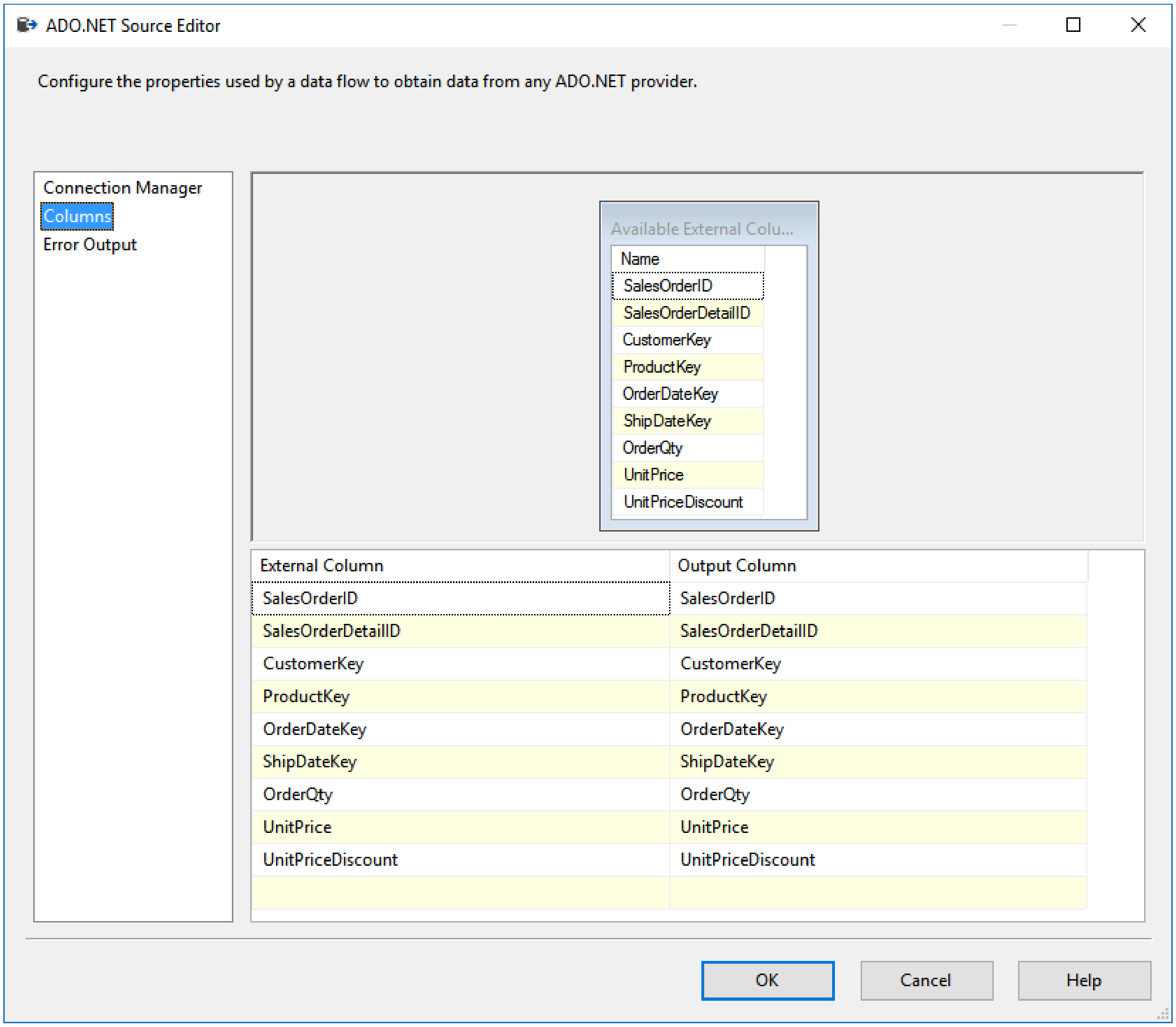
```sql
-- Code for FactSales Source Data
SELECT
  T1.[SalesOrderID]
, [SalesOrderDetailID]
, T3.[CustomerKey]
, T4.[ProductKey]
, [OrderDateKey] = T5.CalendarDateKey
, [ShipDateKey] = T6.CalendarDateKey
, [OrderQty]
, [UnitPrice]
, [UnitPriceDiscount]
FROM [AdventureWorksLT2012].[SalesLT].[SalesOrderDetail] as T1
JOIN [AdventureWorksLT2012].[SalesLT].[SalesOrderHeader] as T2
   ON T1.[SalesOrderID] = T2.[SalesOrderID]
JOIN [DWAdventureWorksLT2012Lab03].[dbo].[DimCustomers] as T3
   ON T2.[CustomerID] = T3.[CustomerID]
JOIN [DWAdventureWorksLT2012Lab03].[dbo].[DimProducts] as T4
   ON T4.[ProductID] = T1.[ProductID]
JOIN [DWAdventureWorksLT2012Lab03].[dbo].[DimDates] as T5
   ON Cast(T5.CalendarDate as Date) = Cast(T2.[OrderDate] as Date)
JOIN [DWAdventureWorksLT2012Lab03].[dbo].[DimDates] as T6
   ON Cast(T6.CalendarDate as Date) = Cast(T2.[ShipDate] as Date)
go
```

10. Paste the copied code into the **SQL command text textbox**.

Note: **Make sure you do not copy the “go” keyword.**

11. Click the **Preview** button and verify the code returns data or troubleshoot any issues.

12. Click the **Columns** page in the left-hand navigation area and verify the columns are listed as shown here:



Configure the properties used by a data flow to obtain data from any ADO.NET provider.

Available External Colu...

Name
SalesOrderID
SalesOrderDetailID
CustomerKey
ProductKey
OrderDateKey
ShipDateKey
OrderQty
UnitPrice
UnitPriceDiscount

External Column	Output Column
SalesOrderID	SalesOrderID
SalesOrderDetailID	SalesOrderDetailID
CustomerKey	CustomerKey
ProductKey	ProductKey
OrderDateKey	OrderDateKey
ShipDateKey	ShipDateKey
OrderQty	OrderQty
UnitPrice	UnitPrice
UnitPriceDiscount	UnitPriceDiscount

13. Click **OK** to close the dialog window.

14. From the SSIS Toolbox, drag an **ADO NET Destination** into the Data Flow design surface.

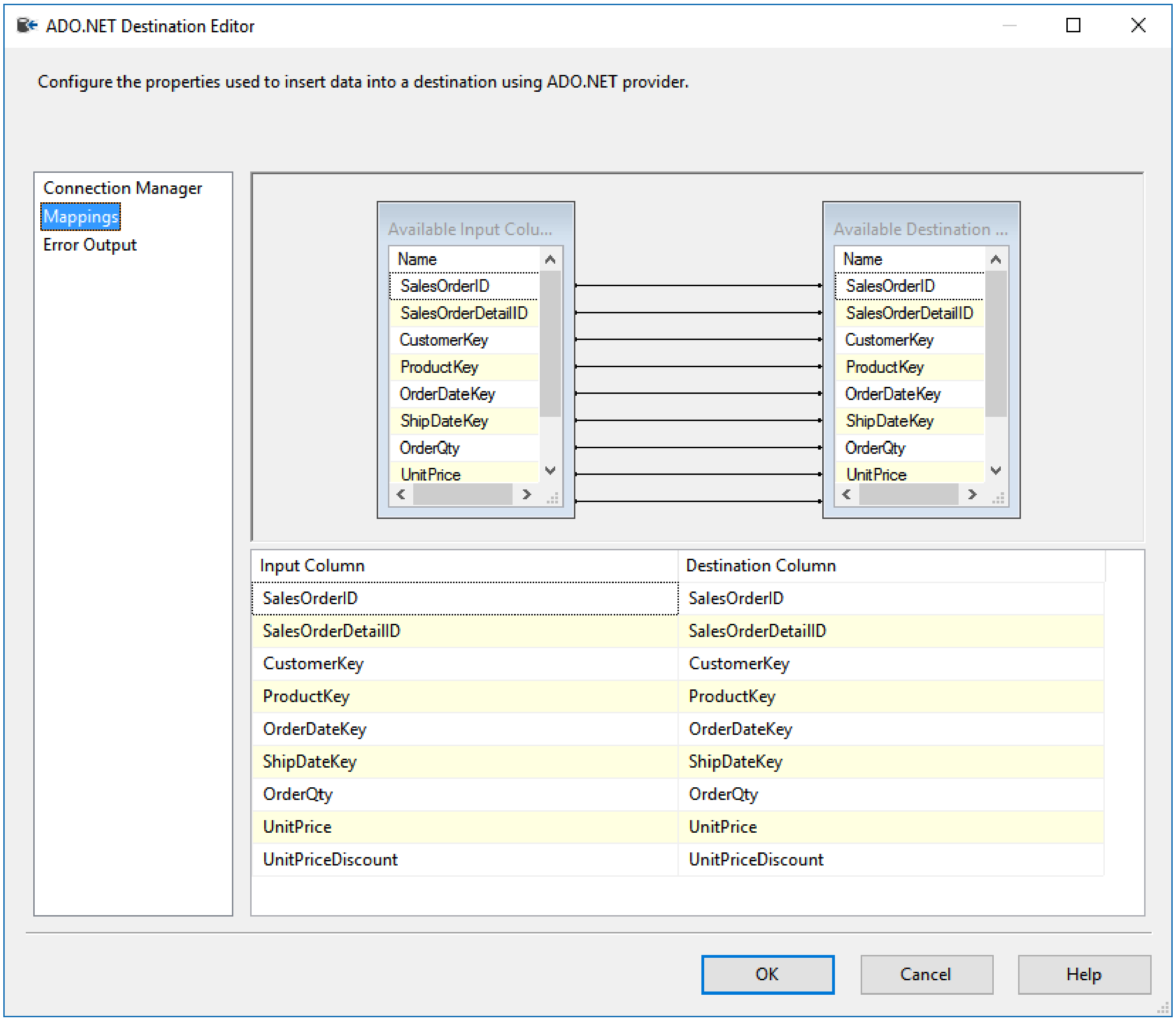
15. Right-click the source and rename it to **FactSales ADO NET Destination.**

16. Right-click the source and click **Edit** in the context menu**.**

17. Select the ADO.NET connection you created in Exercise 5 in the **ADO.NET connection manager dropdown**.

18. Select the "dbo"."FactSales" table in the **Use a table or view dropdown**.

19. Click the **Mappings** page in the left-hand navigation area and verify the columns are mapped as shown here:


ADO.NET Destination Editor

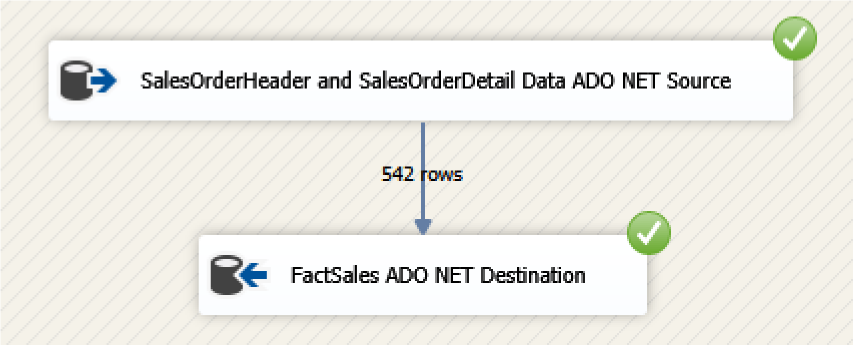
Configure the properties used to insert data into a destination using ADO.NET provider.

Input Column	Destination Column
SalesOrderID	SalesOrderID
SalesOrderDetailID	SalesOrderDetailID
CustomerKey	CustomerKey
ProductKey	ProductKey
OrderDateKey	OrderDateKey
ShipDateKey	ShipDateKey
OrderQty	OrderQty
UnitPrice	UnitPrice
UnitPriceDiscount	UnitPriceDiscount

20. Click **OK** to close the dialog window.

21. Right-click the **ETL Prototype Ver1 with Data Flows.dtsx** package file and select **Execute Package** from the context menu.

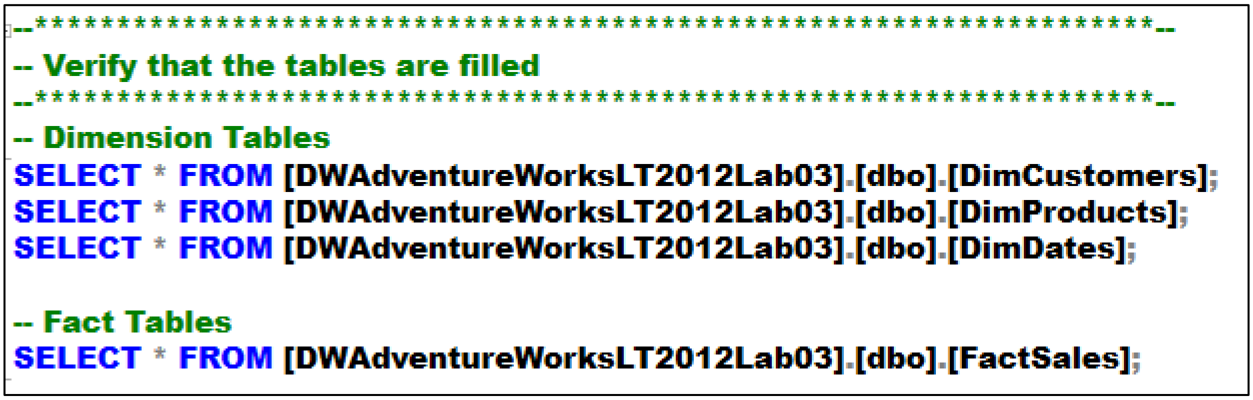
22. Verify that all the tasks have run successfully or troubleshoot any issues. Remember, if you need to change settings in the task, you must first stop the **debugging engine.**


SalesOrderHeader and SalesOrderDetail Data ADO NET Source
542 rows
FactSales ADO NET Destination

Exercise 12 | Verify the tables are filled

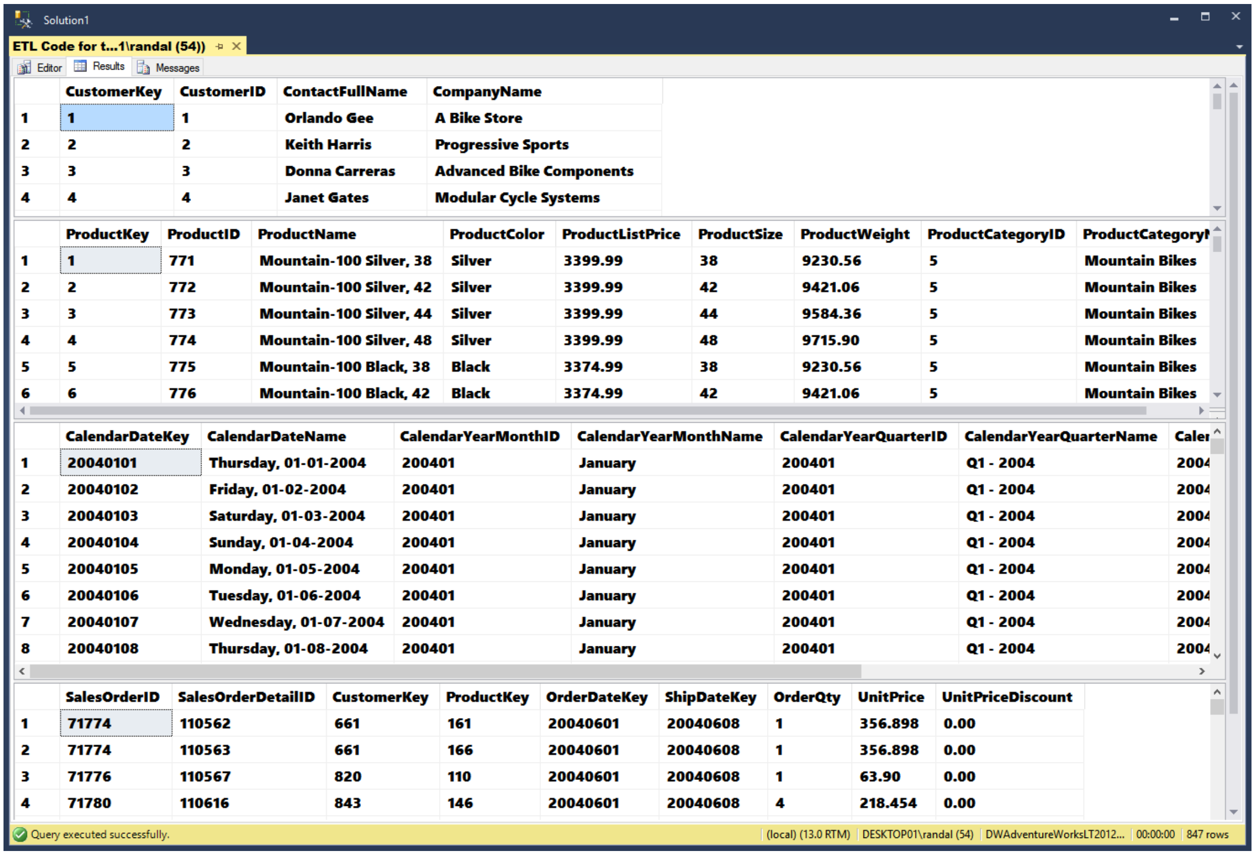
1. Use the File > Open > File Menu to open the C:_ETLwithSSIS\Module04\Labs\ETL Code for the DWAdventureWorksLTLab03 database.sql file.

2. Locate the SQL code that selects data from the **all the dimension and fact tables** and Execute it.



```
--**********************************************************************--
-- Verify that the tables are filled
--**********************************************************************--
-- Dimension Tables
SELECT * FROM [DWAdventureWorksLT2012Lab03].[dbo].[DimCustomers];
SELECT * FROM [DWAdventureWorksLT2012Lab03].[dbo].[DimProducts];
SELECT * FROM [DWAdventureWorksLT2012Lab03].[dbo].[DimDates];

-- Fact Tables
SELECT * FROM [DWAdventureWorksLT2012Lab03].[dbo].[FactSales];
```

3. Verify that all of the tables have data.



	CustomerKey	CustomerID	ContactFullName	CompanyName
1	1	1	Orlando Gee	A Bike Store
2	2	2	Keith Harris	Progressive Sports
3	3	3	Donna Carreras	Advanced Bike Components
4	4	4	Janet Gates	Modular Cycle Systems

	ProductKey	ProductID	ProductName	ProductColor	ProductListPrice	ProductSize	ProductWeight	ProductCategoryID	ProductCategoryN
1	1	771	Mountain-100 Silver, 38	Silver	3399.99	38	9230.56	5	Mountain Bikes
2	2	772	Mountain-100 Silver, 42	Silver	3399.99	42	9421.06	5	Mountain Bikes
3	3	773	Mountain-100 Silver, 44	Silver	3399.99	44	9584.36	5	Mountain Bikes
4	4	774	Mountain-100 Silver, 48	Silver	3399.99	48	9715.90	5	Mountain Bikes
5	5	775	Mountain-100 Black, 38	Black	3374.99	38	9230.56	5	Mountain Bikes
6	6	776	Mountain-100 Black, 42	Black	3374.99	42	9421.06	5	Mountain Bikes

	CalendarDateKey	CalendarDateName	CalendarYearMonthID	CalendarYearMonthName	CalendarYearQuarterID	CalendarYearQuarterName	Calen
1	20040101	Thursday, 01-01-2004	200401	January	200401	Q1 - 2004	2004
2	20040102	Friday, 01-02-2004	200401	January	200401	Q1 - 2004	2004
3	20040103	Saturday, 01-03-2004	200401	January	200401	Q1 - 2004	2004
4	20040104	Sunday, 01-04-2004	200401	January	200401	Q1 - 2004	2004
5	20040105	Monday, 01-05-2004	200401	January	200401	Q1 - 2004	2004
6	20040106	Tuesday, 01-06-2004	200401	January	200401	Q1 - 2004	2004
7	20040107	Wednesday, 01-07-2004	200401	January	200401	Q1 - 2004	2004
8	20040108	Thursday, 01-08-2004	200401	January	200401	Q1 - 2004	2004

	SalesOrderID	SalesOrderDetailID	CustomerKey	ProductKey	OrderDateKey	ShipDateKey	OrderQty	UnitPrice	UnitPriceDiscount
1	71774	110562	661	161	20040601	20040608	1	356.898	0.00
2	71774	110563	661	166	20040601	20040608	1	356.898	0.00
3	71776	110567	820	110	20040601	20040608	1	63.90	0.00
4	71780	110616	843	146	20040601	20040608	4	218.454	0.00

Query executed successfully. | (local) (13.0 RTM) | DESKTOP01\randal (54) | DWAdventureWorksLT2012... | 00:00:00 | 847 rows