SYLLABUS

- Introduction to ETL with SSIS
- Introduction to SQL Server Data Tools
- Implementing data flow using SSIS
  - Configuring the Data Flow
    - Sources
    - Transformations
      - Derived Column
      - Conditional Split
      - Destinations
- Introduction to Control Flow
  - SSIS Tasks
- SSIS Containers
  - For Loop Container
  - Foreach Loop Container
    - Foreach File enumerator
    - Foreach Item enumerator
    - Foreach SMO enumerator
    - Foreach Node list enumerator
    - Foreach ADO enumerator
    - Foreach ADO.NET Schema
    - Foreach From Variable enumerator
  - Sequence Container
- Create Dynamic Packages Using Containers

Table of Contents

1. Introduction to ETL with SSIS

Data movement represents an important part of data management. Data is transported from client applications to the data server to be stored, and transported back from the database to the client to be managed and used. In data warehousing, data movement represents a particularly important element, considering the typical requirements of a data warehouse: the need to import data from one or more operational data stores, the need to cleanse and consolidate the data, and the need to transform data, which allows it to be stored and maintained appropriately in the data warehouse.

Microsoft SQL Server 2012 provides a dedicated solution for this particular set of requirements: **SQL Server Integration Services (SSIS).** In contrast to Line of Business (LOB) data management operations, in which individual business entities are processed one at a time in the client application by a human operator, data warehousing (DW) operations are performed against collections of business entities in automated processes. In light of these important differences, SSIS provides the means to perform operations against large quantities of data efficiently and, as much as possible, without any need for human intervention.

Another difference between LOB and DW data management is when the operations are executed; LOB operations are predominantly performed during standard work hours, but DW operations are performed during "maintenance windows": typically, data maintenance in a DW is performed during times of less usage or no usage at all (for example, at night). This is both in order to reduce the impact of resource intensive operations on the LOB system as well as to reduce the impact of data volatility on the DW.

Based on the level of complexity, data movement scenarios can be divided into two groups:

- Simple data movements, where data is moved from the source to the destination "as-is" (unmodified)
- Complex data movements, where the data needs to be transformed before it can be stored, and where additional programmatic logic is required to accommodate the merging of the new and/or modified data, arriving from the source, with existing data, already present at the destination
- In light of this, the SQL Server 2012 tool set provides two distinct approaches to developing data movement processes:
- The SQL Server Import and Export Wizard, which can be used to design (and execute) simple data movements, such as the transfer of data from one database to another
- The SQL Server Data Tools, which boast a complete integrated development environment, providing SQL Server Integration Services (SSIS) developers with the ability to design even the most complex data movement processes

What constitutes a complex data movement? Three distinct elements can be observed in any complex data movement process:

1. The data is extracted from the source (retrieved from the operational data store).

2. The data is transformed (cleansed, converted, reorganized, and restructured) to comply with the destination data model.

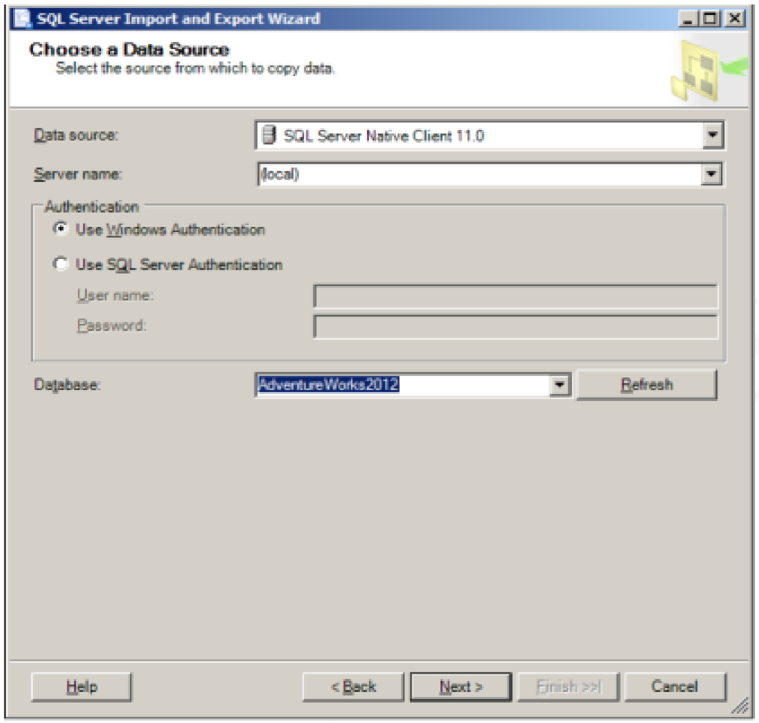
3. The data is loaded into the destination data store (such as a data warehouse).

This process is also known as **extract-transform-load, or ETL**. In simple data movements, however, the transform element is omitted, leaving only two elements: extract and load.

2. Extract Data from a View and Load It into a Table

1. Start the SQL Server Import and Export Wizard: on the Start menu, click All Programs | Microsoft SQL Server 2012. On the Welcome page, click next.

2. To choose the data source, connect to your server, select the appropriate authentication settings, and select the AdventureWorks2012 database, as shown in Figure 1.Then click Next.


SQL Server Import and Export Wizard

Choose a Data Source

Select the source from which to copy data.

Data source: SQL Server Native Client 11.0

Server name: (local)

Authentication

(● Use Windows Authentication)

○ Use SQL Server Authentication

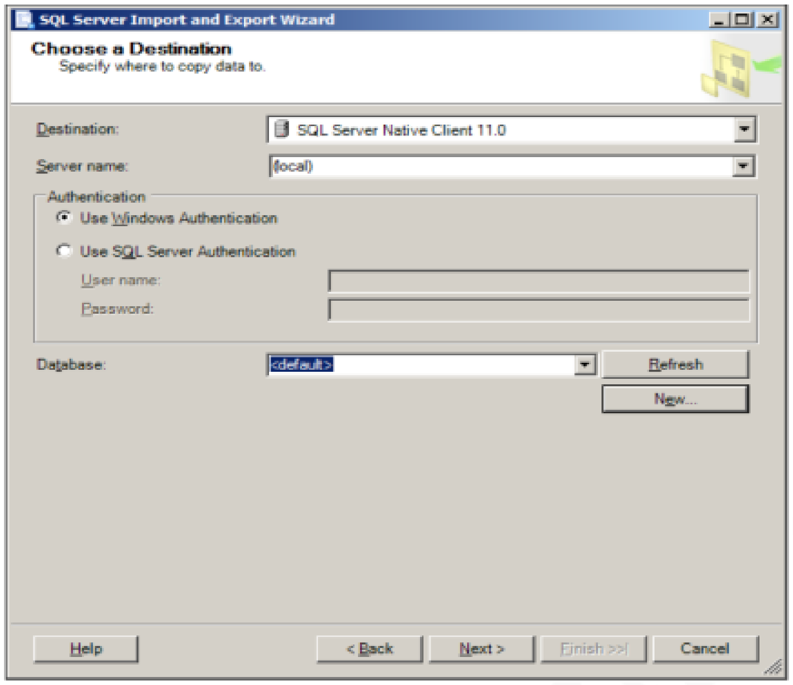
User name:

Password:

Database: AdventureWorks2012

Figure 1 - Choosing a data source.

3. To choose a destination, connect to your server and use the same authentication settings as in the previous step. This is shown in Figure 2. One option is to load the data into an existing database; however, in this exercise, the destination database does not exist. Click New to create it.



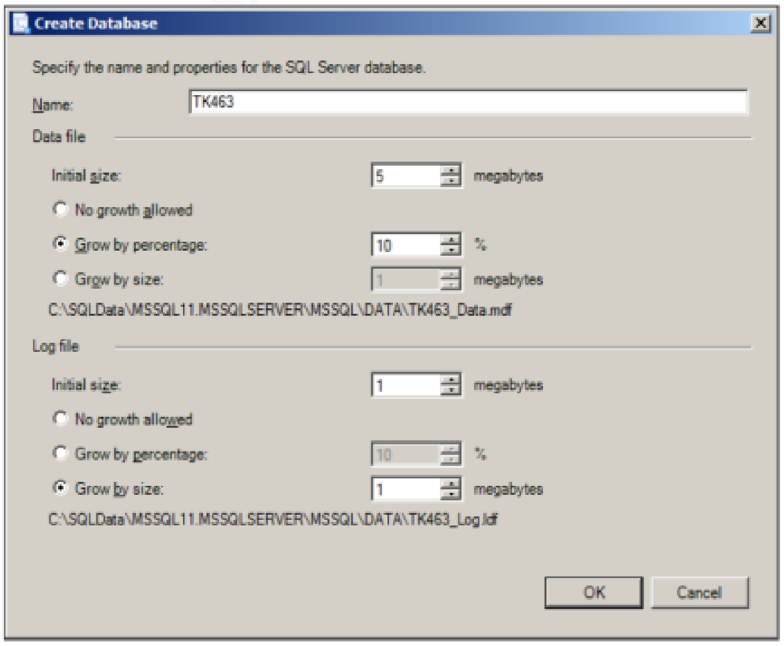
SQL Server Import and Export Wizard

Choose a Destination
Specify where to copy data to.

Destination: SQL Server Native Client 11.0
Server name: (local)
Authentication
Use Windows Authentication [✓]
Use SQL Server Authentication
User name:
Password:
Database: <default>

Figure 2 - Choosing a destination.

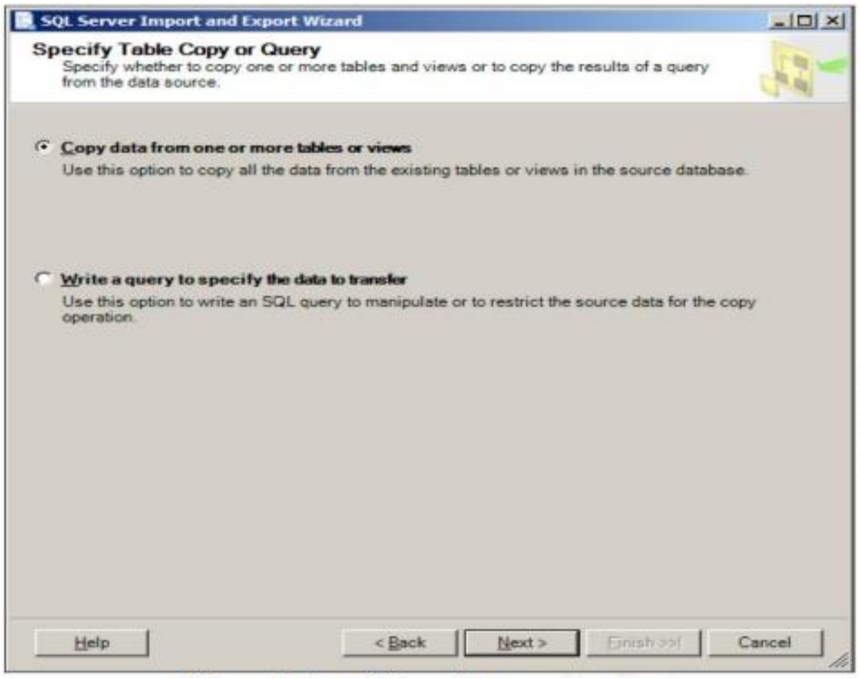
4. To create a new database, provide a name for it (TK 463), as shown in Figure 3. Leave the rest of the settings unchanged. Then click OK, and then Next.
5. On the next page, shown in Figure 4, you need to decide whether you want to extract the data from one or more existing objects of the source database or whether you want to use a single query to extract the data. Select Copy Data from One or More Tables or Views, and then click next.
The first option allows you to select multiple objects, but it does not allow you to restrict the extracted data; the second option, on the other hand, allows you to restrict the extracted data, but it only supports a single result set.



Create Database

Specify the name and properties for the SQL Server database.
Name: TK463
Data file
Initial size: 5 megabytes
No growth allowed
Grow by percentage: 10 % [✓]
Grow by size: 1 megabytes
C:\SQLData\MSSQL11.MSSQLSERVER\MSSQL\DATA\TK463_Data.mdf
Log file
Initial size: 1 megabytes
No growth allowed
Grow by percentage: 10 %
Grow by size: 1 megabytes [✓]
C:\SQLData\MSSQL11.MSSQLSERVER\MSSQL\DATA\TK463_Log.ldf

Figure 3 - Creating the database.


SQL Server Import and Export Wizard

Specify Table Copy or Query

Specify whether to copy one or more tables and views or to copy the results of a query from the data source.

(●) Copy data from one or more tables or views

Use this option to copy all the data from the existing tables or views in the source database.

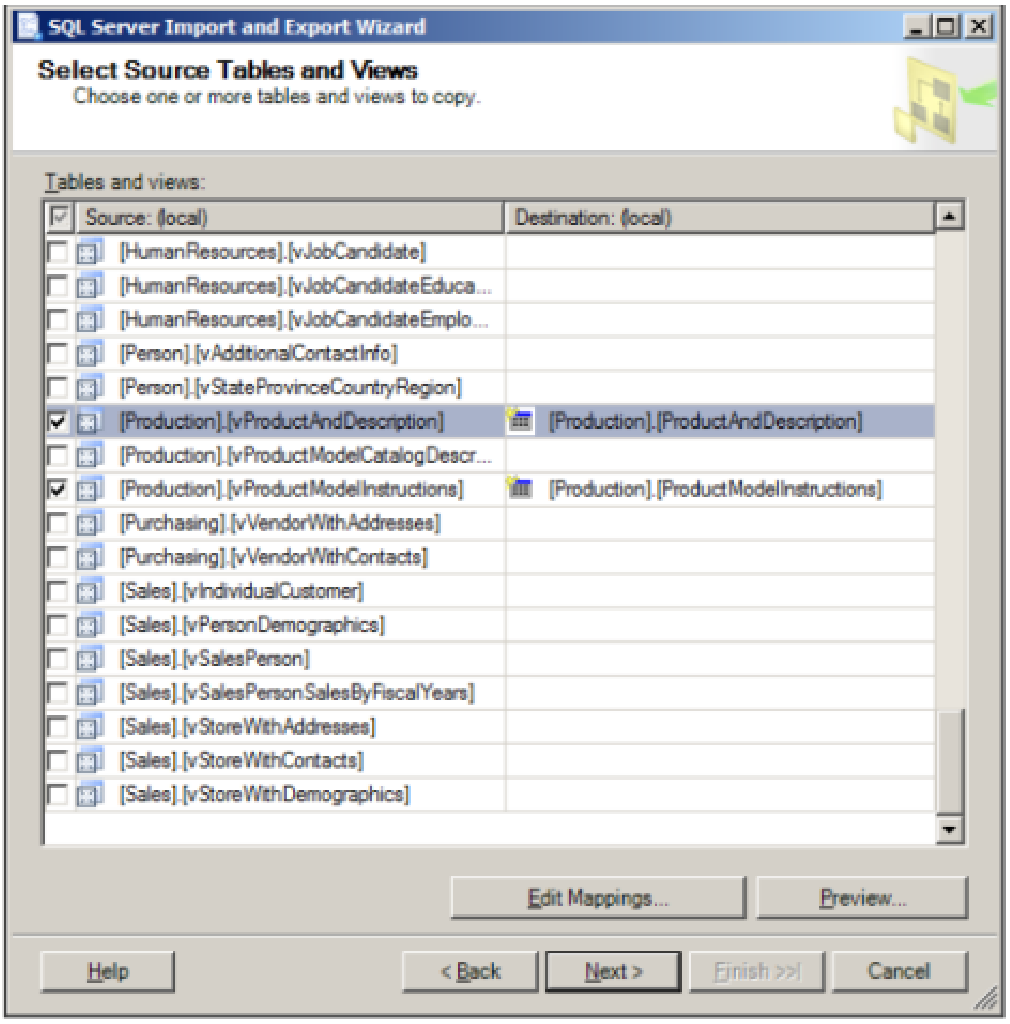
( ) Write a query to specify the data to transfer

Use this option to write an SQL query to manipulate or to restrict the source data for the copy operation.

Figure 4 - Specifying table copy or query.

6. On the next page, you select the objects from which you want to extract the data. In this exercise, you will extract the data from two views and load it into two newly created tables in the destination database.

In the left column of the grid, select the following two source views:
· [Production]. [vProductAndDescription]
· [Production]. [vProductModelInstructions]
In the right column, change the names of the destination tables, as follows:
· [Production]. [ProductAndDescription]
· [Production]. [ProductModelInstructions]
The result is shown in Figure 5.
7. Select the first view, and then click Edit Mappings. As shown in Figure 6, you can see that the data extracted from the view will be inserted into a newly created table.
The definition of the new table is prepared automatically by the wizard and is based on the schema of the source row set. If necessary, you can modify the table definition by clicking Edit SQL. However, in this exercise, you should leave the definition unchanged.



SQL Server Import and Export Wizard

Select Source Tables and Views

Choose one or more tables and views to copy.

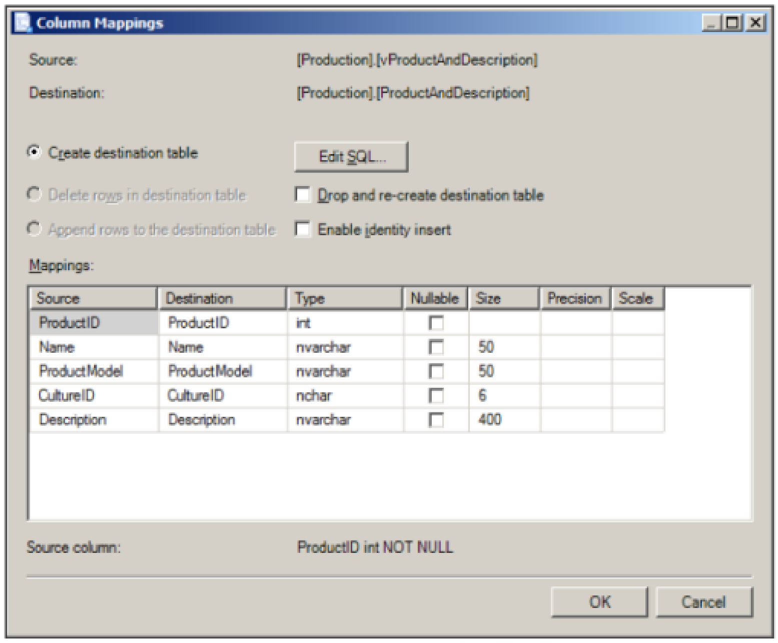
Tables and views:

Source: (local)	Destination: (local)
☐ [HumanResources].[vJobCandidate]	
☐ [HumanResources].[vJobCandidateEduca...	
☐ [HumanResources].[vJobCandidateEmplo...	
☐ [Person].[vAdditionalContactInfo]	
☐ [Person].[vStateProvinceCountryRegion]	
✓ [Production].[vProductAndDescription]	[Production].[ProductAndDescription]
☐ [Production].[vProductModelCatalogDescr...	
✓ [Production].[vProductModelInstructions]	[Production].[ProductModelInstructions]
☐ [Purchasing].[vVendorWithAddresses]	
☐ [Purchasing].[vVendorWithContacts]	
☐ [Sales].[vIndividualCustomer]	
☐ [Sales].[vPersonDemographics]	
☐ [Sales].[vSalesPerson]	
☐ [Sales].[vSalesPersonSalesByFiscalYears]	
☐ [Sales].[vStoreWithAddresses]	
☐ [Sales].[vStoreWithContacts]	
☐ [Sales].[vStoreWithDemographics]	

Figure 5 - Selecting source tables and views.

8. When you are done, click OK to close the Column Mappings window; if you have made any changes to the table definition, click Cancel because no changes are necessary for this exercise. On the Select Source Tables and Views page, click next.

9. On the next page, shown in Figure 7, you can decide whether to run the package, save it for later, or even do both. Make sure the Run Immediately check box is selected, and also select the Save SSIS Package check box. Then select File System as the destination for the newly created package. Under Package Protection Level, select Do Not Save Sensitive Data. Then click next.



Column Mappings

Source: [Production].[vProductAndDescription]

Destination: [Production].[ProductAndDescription]

[✓ Create destination table] Edit SQL...

Delete rows in destination table — Drop and re-create destination table

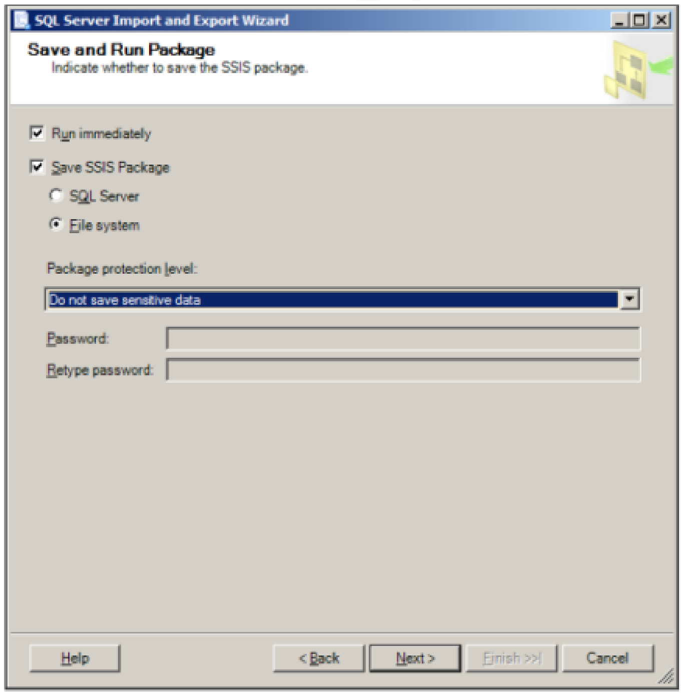
Append rows to the destination table — Enable identity insert

Mappings:

Source	Destination	Type	Nullable	Size	Precision	Scale
ProductID	ProductID	int	☐			
Name	Name	nvarchar	☐	50		
ProductModel	ProductModel	nvarchar	☐	50		
CultureID	CultureID	nchar	☐	6		
Description	Description	nvarchar	☐	400		

Source column: ProductID int NOT NULL

Figure 6 - Column mappings.



SQL Server Import and Export Wizard

Save and Run Package

Indicate whether to save the SSIS package.

[✓ Run immediately]

[✓ Save SSIS Package]

SQL Server [✓ File system]

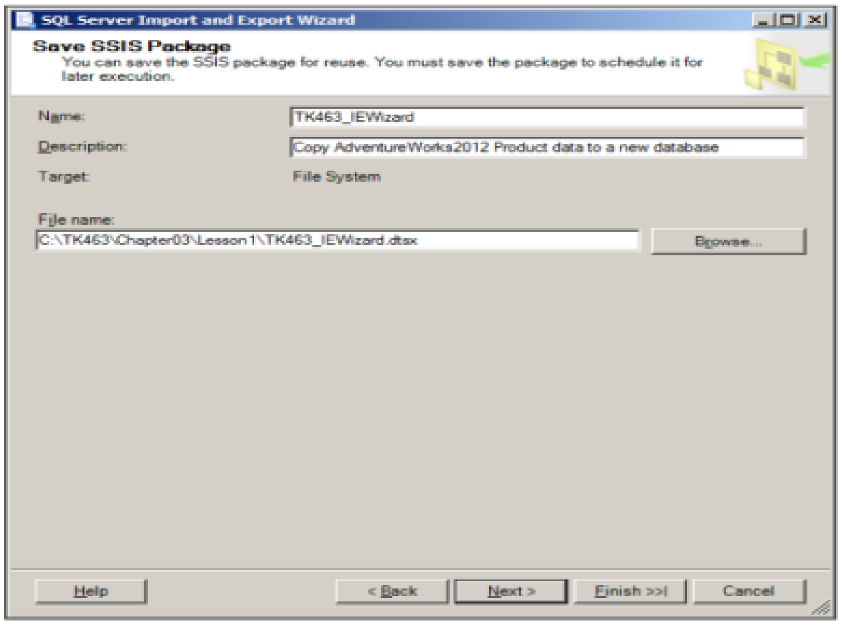
Package protection level: Do not save sensitive data

Password:

Retype password:

Figure 7 - Saving and running the package.

10. On the next page, shown in Figure 8, name your package (TK 463_IE Wizard), provide a description for it if you want (for example, Copy AdventureWorks2012 Product data to a new database), and name the resulting SSIS package file (C:\TK 463\Chapter03\Lesson1\TK 463_IE Wizard.dtsx). When ready, click next.
11. On the next page, shown in Figure 9, you can review the actions that will be performed when the package is executed. When ready, click next.


SQL Server Import and Export Wizard

Save SSIS Package

You can save the SSIS package for reuse. You must save the package to schedule it for later execution.

Name: TK463_IEWizard

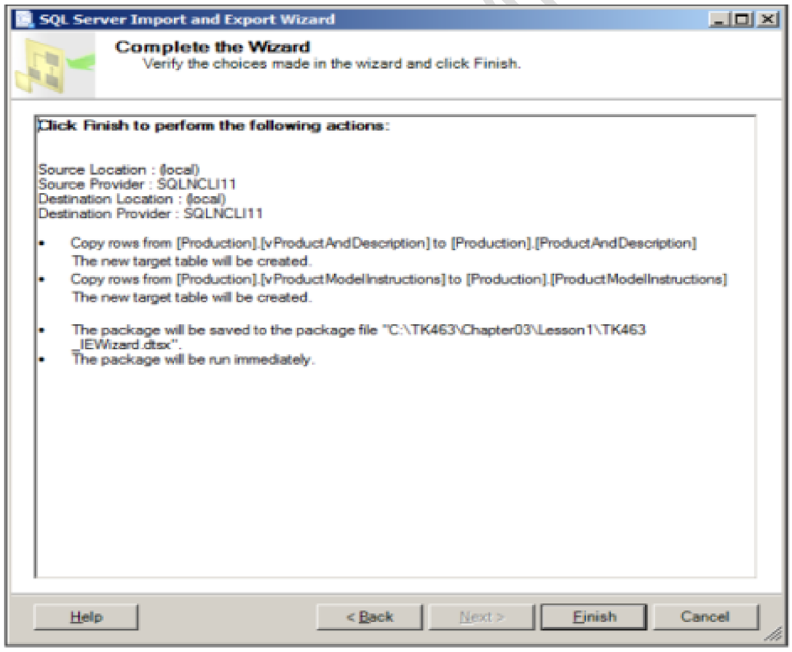
Description: Copy AdventureWorks2012 Product data to a new database

Target: File System

File name: C:\TK463\Chapter03\Lesson1\TK463_IEWizard.dtsx

Figure 8 - Saving the SSIS package.

12. To execute the package, click Finish.
13. A new page appears, as shown in Figure 10, displaying the progress and finally the results of the execution. Close the wizard when you're done.


SQL Server Import and Export Wizard

Complete the Wizard

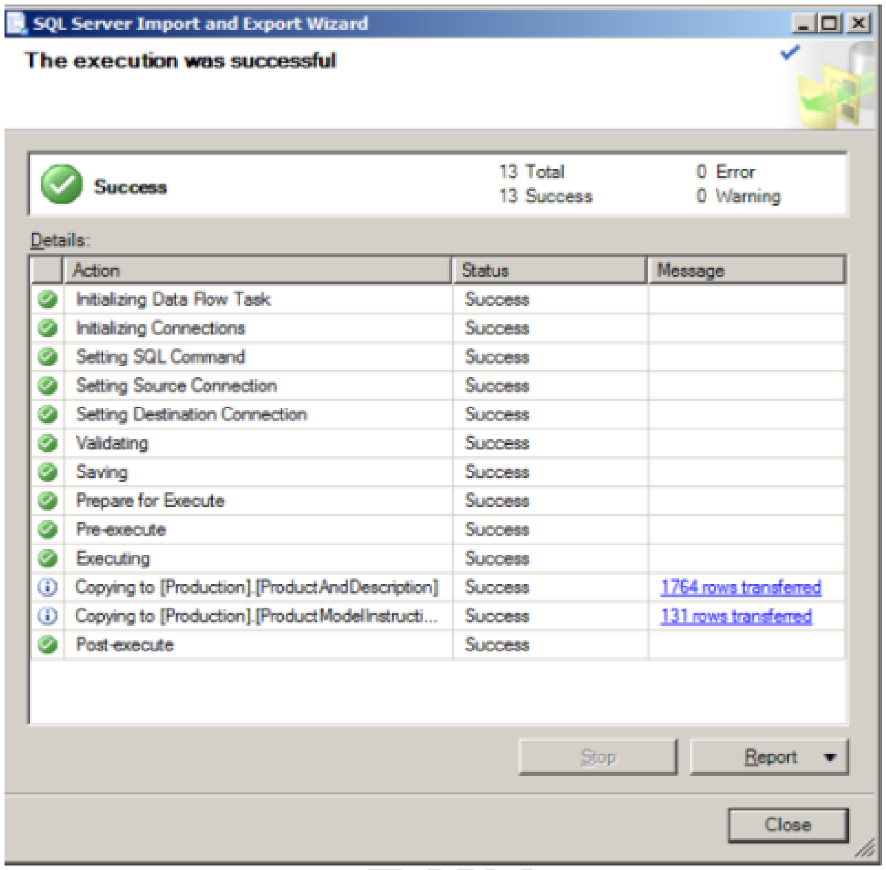
Verify the choices made in the wizard and click Finish.

Click Finish to perform the following actions:

Source Location : (local)
Source Provider : SQLNCLI11
Destination Location : (local)
Destination Provider : SQLNCLI11

- Copy rows from [Production].[vProductAndDescription] to [Production].[ProductAndDescription]
  The new target table will be created.
- Copy rows from [Production].[vProductModelInstructions] to [Production].[ProductModelInstructions]
  The new target table will be created.
- The package will be saved to the package file "C:\TK463\Chapter03\Lesson1\TK463_IEWizard.dtsx".
- The package will be run immediately.

Figure 9 - Completing the wizard.



The execution was successful

Success — 13 Total, 13 Success, 0 Error, 0 Warning

Details:

Action	Status	Message
Initializing Data Flow Task	Success	
Initializing Connections	Success	
Setting SQL Command	Success	
Setting Source Connection	Success	
Setting Destination Connection	Success	
Validating	Success	
Saving	Success	
Prepare for Execute	Success	
Pre-execute	Success	
Executing	Success	
Copying to [Production].[ProductAndDescription]	Success	1764 rows transferred
Copying to [Production].[ProductModelInstructi...	Success	131 rows transferred
Post-execute	Success	

Figure 10 - The execution

3. Introducing Control Flow, Data Flow, and Connection Managers

Before you dive into SSIS development, you should be familiar with three essential elements of every SSIS package:

- Connection managers provide connections to data stores, either as data sources or data destinations. Because the same data store can play the role of the data source as well as the data destination, connection managers allow the connection to be defined once and used many times in the same package (or project).
- Control flow defines both the order of operations and the conditions under which they will be executed. A package can consist of one or more operations, represented by control flow tasks. Execution order is defined by how individual tasks are connected to one another. Tasks that do not follow any preceding task as well as tasks that follow the same preceding task are executed in parallel.
- **Data flow Encapsulates the data movement components—the ETL:**

▪ One or more source components, designating the data stores from which the data will be extracted ▪ One or more destination components, designating the data stores into which the data will be loaded

▪ One or more (optional) transformation components, designating the transformations through which the data will be passed

The role of connection managers is to provide access to data stores, either as data sources, data destinations, or reference data stores. Control flow tasks define the data management operations of the SSIS process, with the data flow tasks providing the core of data warehousing operations—the ETL.

4. Introducing SSDT (SQL Server Data Tools)

SSDT is a special edition of Visual Studio, which is Microsoft's principal integrated development environment. SSDT supports a variety of SQL Server development projects, such as SQL Server Analysis Services Multidimensional and Data Mining projects, Analysis Services Tabular projects, SQL Server Reporting Services Report Server projects, and Integration Services (SSIS) projects. For all of these project types, SSDT provides a complete integrated development environment, customized specifically for each particular project type.

For Integration Services projects, SSDT provides an entire arsenal of data management tasks and components covering pretty much any data warehousing need (a variety of data extraction, transformation, and loading techniques). Nonetheless, in the real world, you could eventually encounter situations for which none of the built-in tools provide the most appropriate solution. Fortunately, the SSIS development model is extensible: the built-in tool set can be extended by adding custom tasks and/or custom components—either provided by third-party vendors or developed by you.

5. Developing SSIS Packages in SSDT

Let's start SSDT integrated development environment (IDE), to create a new SSIS project, and explore the SSIS development tool set.

5.1. Start the SQL Server Data Tools (SSDT): On the Start menu, click either All Programs | Microsoft SQL Server 2012|SQL Server Data Tools or All Programs | Microsoft Visual Studio 2010 | Visual Studio 2010.

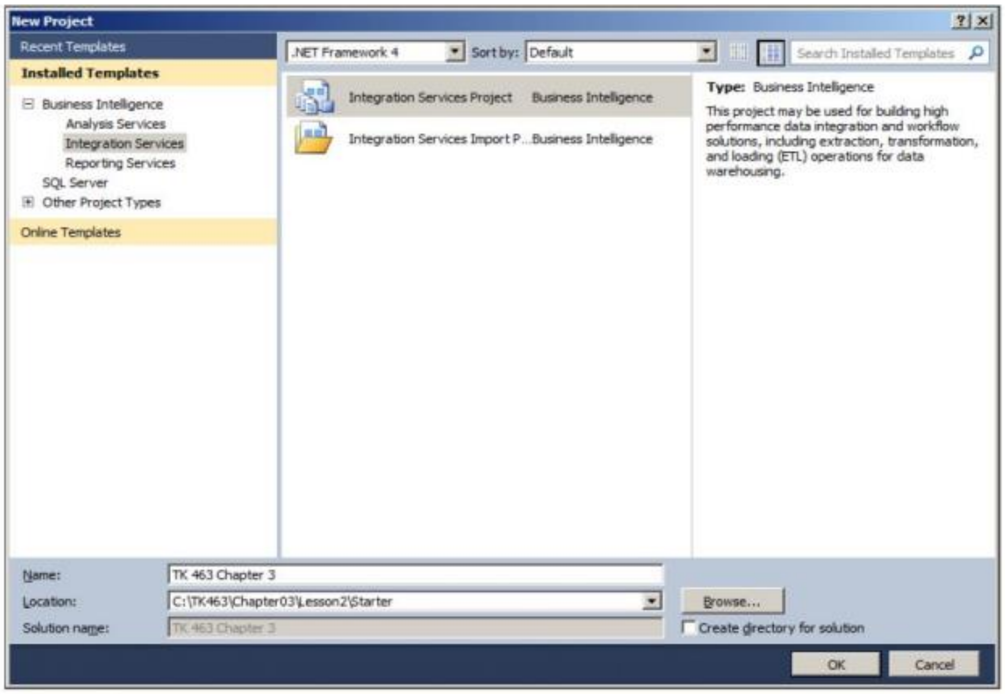
5.2. Create a new project, either by clicking New Project on the Start Page, via the menu by clicking File | New | Project, or by using the Ctrl + Shift + N keyboard shortcut.

5.3. In the New Project window, shown in Figure 11, select the appropriate project template. Under Installed Templates | Business Intelligence | Integration Services, select Integration Services Project.

5.4. At the bottom of the New Project window, provide a name for the project and the location for the project files. Name your project TK 463 Chapter 3, and set the C:\ TK 463\Chapter03\Lesson2\Starter folder as the project location. Also, make sure that the Create Directory for the Solution check box is not selected, because a separate folder for the solution files is not needed. Click OK when ready.

5.5. After the new project and solution have been created, inspect the Solution Explorer pane on the upper-right side of the IDE, as shown in Figure 12. The project you just created should be listed, and it should contain a single SSIS package file named Package.dtsx.

The Solution Explorer pane provides access to solution and project properties and the objects they contain; SSIS projects contain at least one SSIS package. Project-level connection managers and project parameters can be accessed through the Solution Explorer pane.


New Project

Recent Templates
Installed Templates
Business Intelligence
Analysis Services
Integration Services
Reporting Services
SQL Server
Other Project Types
Online Templates

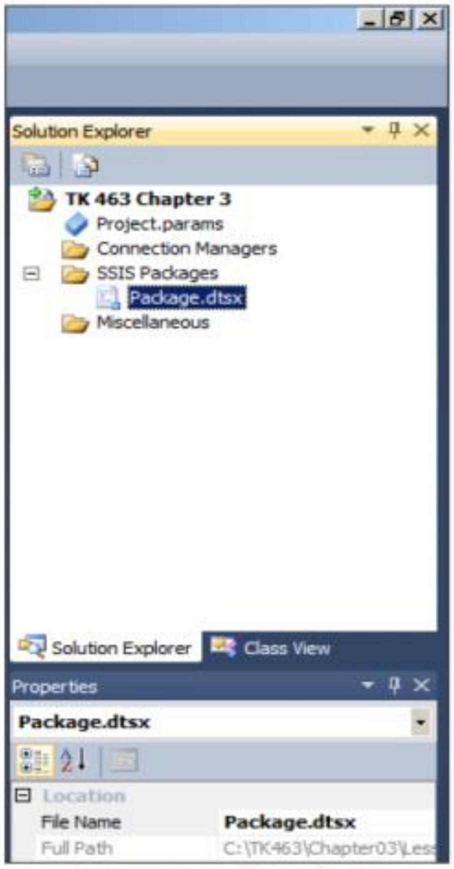
.NET Framework 4 Sort by: Default Search Installed Templates

Integration Services Project Business Intelligence
Integration Services Import P... Business Intelligence

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: TK 463 Chapter 3
Location: C:\TK463\Chapter03\Lesson2\Starter
Solution name: TK 463 Chapter 3
Browse...
Create directory for solution
OK Cancel

Figure 11 - Creating a new project.


Solution Explorer
TK 463 Chapter 3
Project.params
Connection Managers
SSIS Packages
Package.dtsx
Miscellaneous

Solution Explorer Class View

Properties
Package.dtsx
Location
File Name Package.dtsx
Full Path C:\TK463\Chapter03\Les

Figure 12 - Solution Explorer.

6. Implementing Data Flow and Control Flow in SSIS

6.1 What is a control flow?

In SSIS packages, the control flow defines the tasks used in performing data management operations; it determines the order in which these tasks are executed and the conditions of their execution.

6.2 What is a data flow?

In SSIS packages, the data flow is a special control flow task used specifically in data movement operations and data transformations.

7. Edit the SSIS Package Created by the SQL Server Import and Export Wizard

1. Open the TK463_IEWizard.dtsx package by double-clicking it in the Solution Explorer.
2. Review the control flow of the package. It should contain two tasks: an Execute SQL Task named Preparation SQL Task 1, and a data flow task named Data Flow Task 1, as shown in Figure 13.

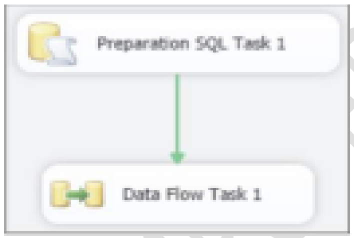
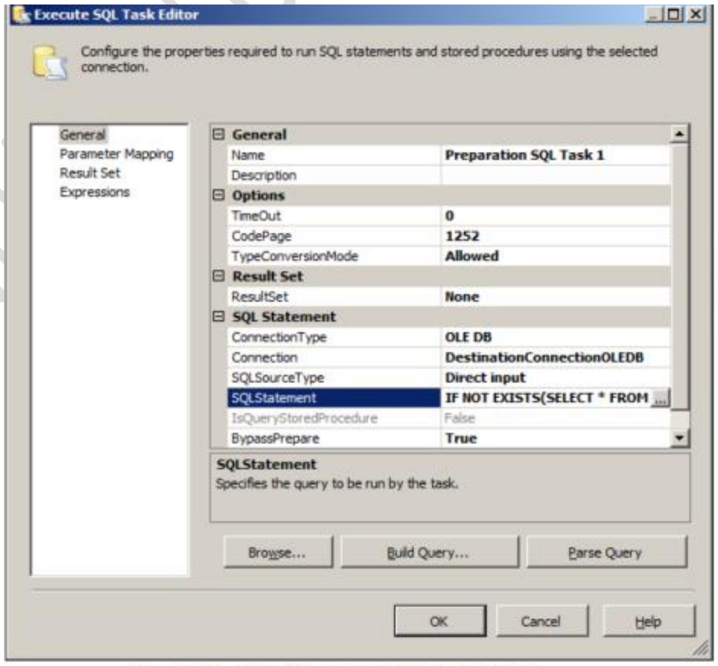


Figure 13 - The control flow of the SSIS package created earlier

3. Double-click (or right-click) Preparation SQL Task 1, and in the shortcut menu select Edit to open the Execute SQL Task Editor. As shown in Figure 14, the editor provides access to the Execute SQL Task's settings used in configuring the operation.



Configure the properties required to run SQL statements and stored procedures using the selected connection.

General	
Name	Preparation SQL Task 1
Description	
Options	
TimeOut	0
CodePage	1252
TypeConversionMode	Allowed
Result Set	
ResultSet	None
SQL Statement	
ConnectionType	OLE DB
Connection	DestinationConnectionOLEDB
SQLSourceType	Direct input
SQLStatement	IF NOT EXISTS(SELECT * FROM
IsQueryStoredProcedure	False
BypassPrepare	True

SQLStatement
Specifies the query to be run by the task.

Figure 14 - The Execute SQL Task Editor.

4. To see the entire definition, click the ellipsis button inside the value box of the SQL Statement property. Resize the script editor dialog box, shown in Figure 15, for better readability, and review the T-SQL script.

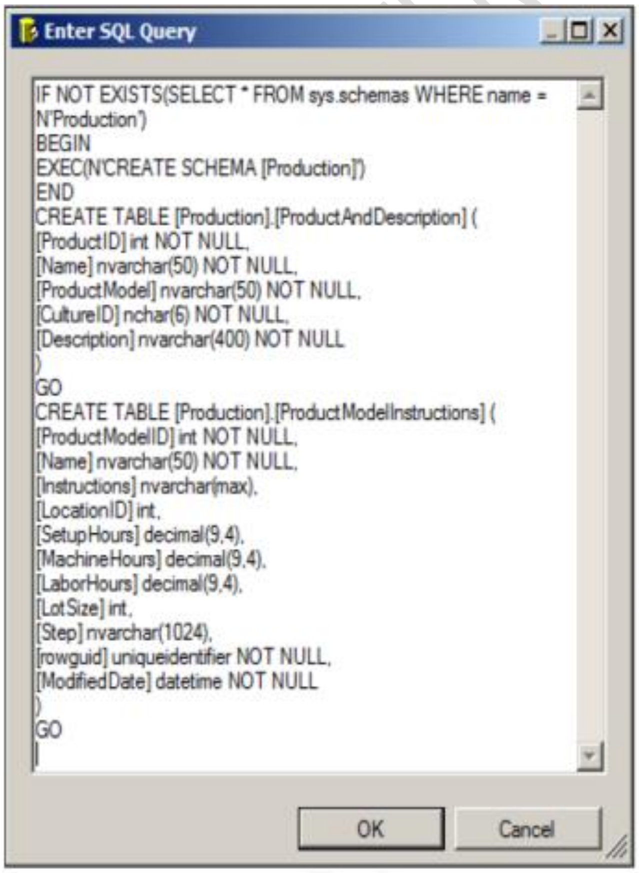
As you can see, the task will attempt to create two tables without checking first to see whether they already exist. The failure will not affect the destination database; it will, however, affect the execution of the SSIS package, causing it to fail.

5. Close the SQL Statement script editor dialog box by clicking Cancel. For the purposes of this exercise, the code does not need to be modified in any way.

6. Close the Execute SQL Task Editor window by clicking Cancel once more.

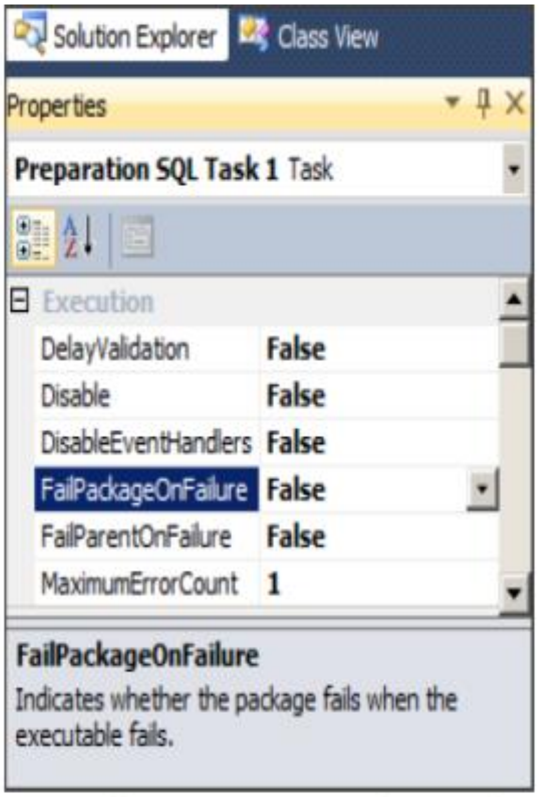
7. Right-click Preparation SQL Task 1 and select Properties on the shortcut menu. In the lower right of the IDE, you can see the Properties pane, displaying additional settings for the selected object—in this case, the Execute SQL Task. Find the Fail Package on Failure setting and make sure its value is False, as shown in Figure 16.

8. Select the precedence constraint (the arrow) leading from Preparation SQL Task 1 to Data Flow Task 1. Press Delete on the keyboard or right-click the constraint and select delete to remove the constraint.



```
IF NOT EXISTS(SELECT * FROM sys.schemas WHERE name =
N'Production')
BEGIN
EXEC(N'CREATE SCHEMA [Production]')
END
CREATE TABLE [Production].[ProductAndDescription] (
[ProductID] int NOT NULL,
[Name] nvarchar(50) NOT NULL,
[ProductModel] nvarchar(50) NOT NULL,
[CultureID] nchar(6) NOT NULL,
[Description] nvarchar(400) NOT NULL
)
GO
CREATE TABLE [Production].[ProductModelInstructions] (
[ProductModelID] int NOT NULL,
[Name] nvarchar(50) NOT NULL,
[Instructions] nvarchar(max),
[LocationID] int,
[SetupHours] decimal(9,4),
[MachineHours] decimal(9,4),
[LaborHours] decimal(9,4),
[LotSize] int,
[Step] nvarchar(1024),
[rowguid] uniqueidentifier NOT NULL,
[ModifiedDate] datetime NOT NULL
)
GO
```

Figure 15 - T-SQL script generated by the Import and Export Wizard.


Solution Explorer | Class View

Properties

Preparation SQL Task 1 Task

Execution

DelayValidation	False
Disable	False
DisableEventHandlers	False
FailPackageOnFailure	False
FailParentOnFailure	False
MaximumErrorCount	1

FailPackageOnFailure
Indicates whether the package fails when the executable fails.

Figure 16 - The Execute SQL Task properties.

9. From the SSIS Toolbox, drag another Execute SQL Task onto the control flow pane.

10. Double-click the newly added task, or right-click it and then select Edit, to open the Execute SQL Task Editor. Configure the task by using the information in Table 3-1.

Property	Value
Name	Preparation SQL Task 2
ConnectionType	OLE DB
Connection	DestinationConnectionOLEDB
SQLSourceType	Direct input

Table 3-1 - Execute SQL Task Properties

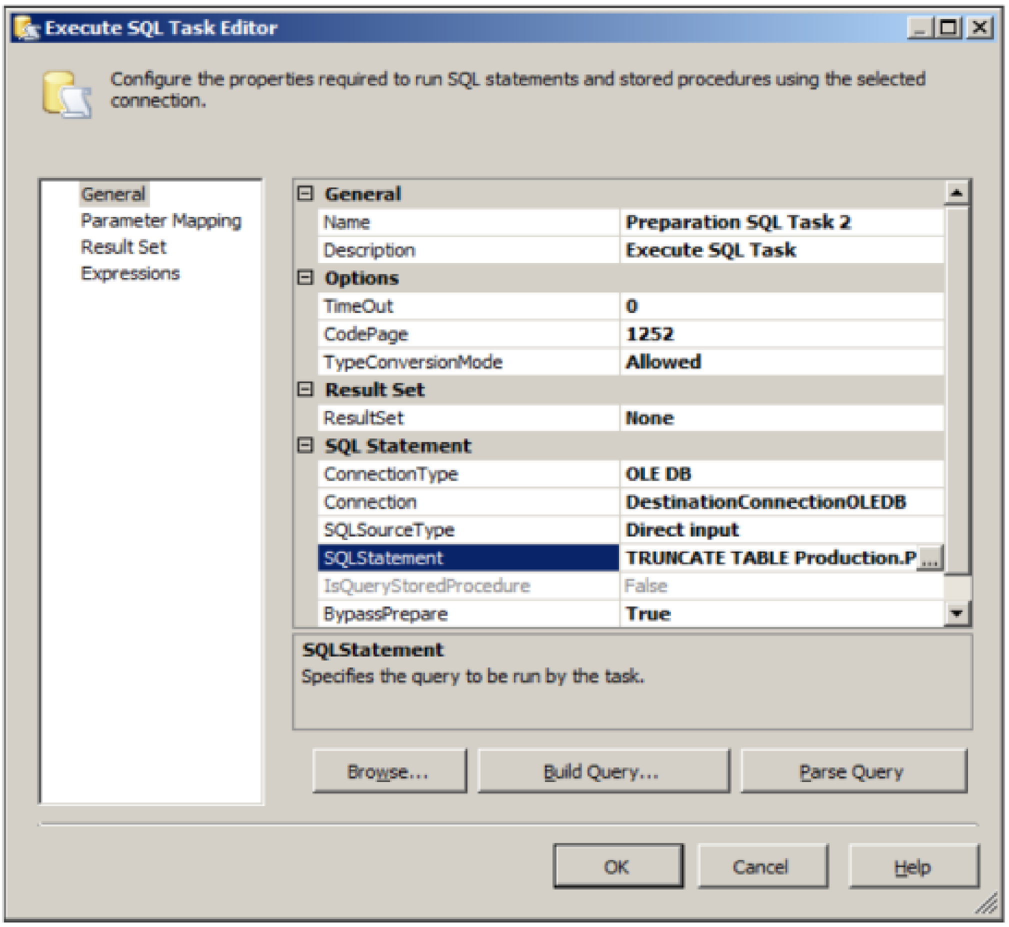
11. Click the ellipsis button inside the value box of the SQLStatement property to edit the SQLStatement, and type in the statements from Listing 3-1.

<u>LISTING 3-1 Truncating Destination Tables</u>

```
TRUNCATE TABLE Production.ProductAndDescription;
TRUNCATE TABLE Production.ProductModelInstructions;
```

Optionally, you can copy and paste the statements from the TK463Chapter03.sql file, located in the C:\TK463\Chapter03\Code folder. Click OK when you are done editing the statements.

12. When you have finished configuring the task as defined in steps 10 and 11, confirm the changes by clicking OK. Figure 17 shows the configured Preparation SQL Task 2.



Configure the properties required to run SQL statements and stored procedures using the selected connection.

General	
Name	Preparation SQL Task 2
Description	Execute SQL Task
Options	
TimeOut	0
CodePage	1252
TypeConversionMode	Allowed
Result Set	
ResultSet	None
SQL Statement	
ConnectionType	OLE DB
Connection	DestinationConnectionOLEDB
SQLSourceType	Direct input
SQLStatement	TRUNCATE TABLE Production.P
IsQueryStoredProcedure	False
BypassPrepare	True

SQLStatement
Specifies the query to be run by the task.

Figure 17 - Preparation SQL Task 2.

13. Select Preparation SQL Task 1. A tiny arrow should appear below it. Drag the arrow over to Preparation SQL Task 2, and then release it to create a precedence constraint between the two tasks, as shown in Figure 18.

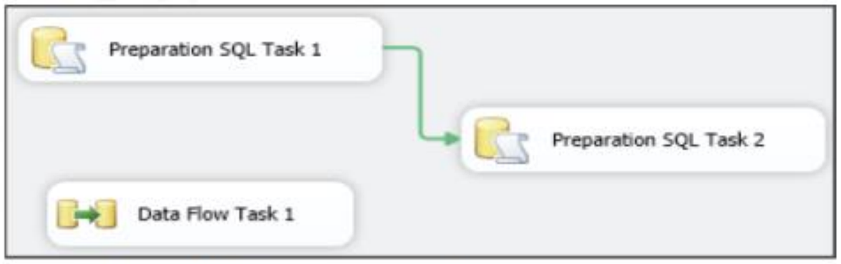
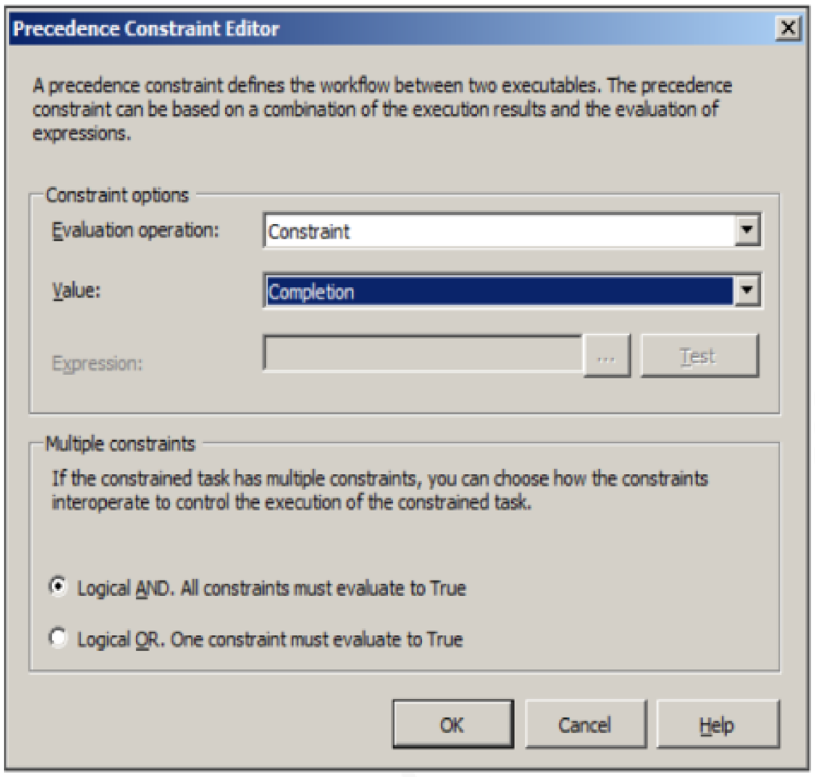


Figure 18 - Creating a precedence constraint.

14. Double-click the precedence constraint you just created, or right-click it and select Edit. In the

Precedence Constraint Editor, shown in Figure 19, you can configure the conditions of the SSIS package execution.


Precedence Constraint Editor

A precedence constraint defines the workflow between two executables. The precedence constraint can be based on a combination of the execution results and the evaluation of expressions.

Constraint options

Evaluation operation: Constraint

Value: Completion

Expression:

Multiple constraints

If the constrained task has multiple constraints, you can choose how the constraints interoperate to control the execution of the constrained task.

[✓ Logical AND. All constraints must evaluate to True]

Logical OR. One constraint must evaluate to True

Figure 19 - The Precedence Constraint Editor.

15. Review the options available for the constraint, make sure that Constraint is selected as the evaluation operation, and then select Completion as the new value, as shown in Figure 19. Confirm the change by clicking OK.

16. Select Preparation SQL Task 2, and connect it to Data Flow Task 1 with a new precedence constraint.

Leave the constraint unchanged. Figure 20, shows the amended control flow.

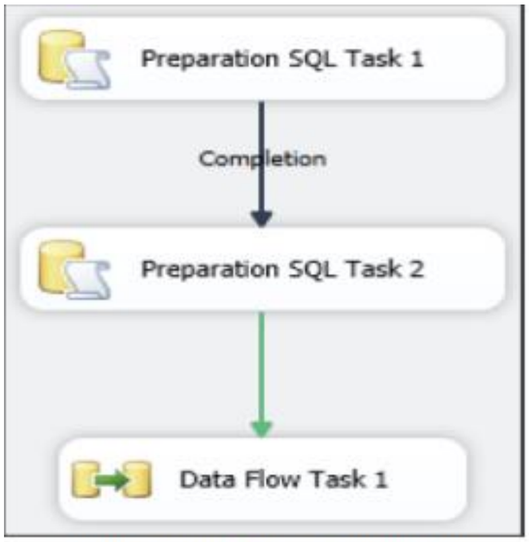


Figure 20 - Modified control flow.

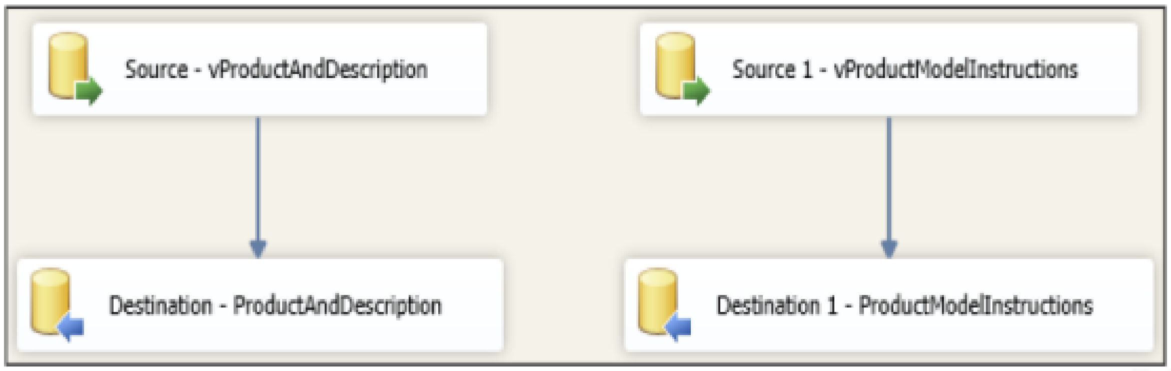

Source - vProductAndDescription
Source 1 - vProductModelInstructions
Destination - ProductAndDescription
Destination 1 - ProductModelInstructions

Figure 21 - The definition of Data Flow Task 1.

17. Double-click Data Flow Task 1, or right-click it and select edit, to view its definition, as shown in Figure 21.
You can observe two data flows, extracting the data from two views in the source database and loading it into two tables in the destination database.

18. Save the SSIS project.

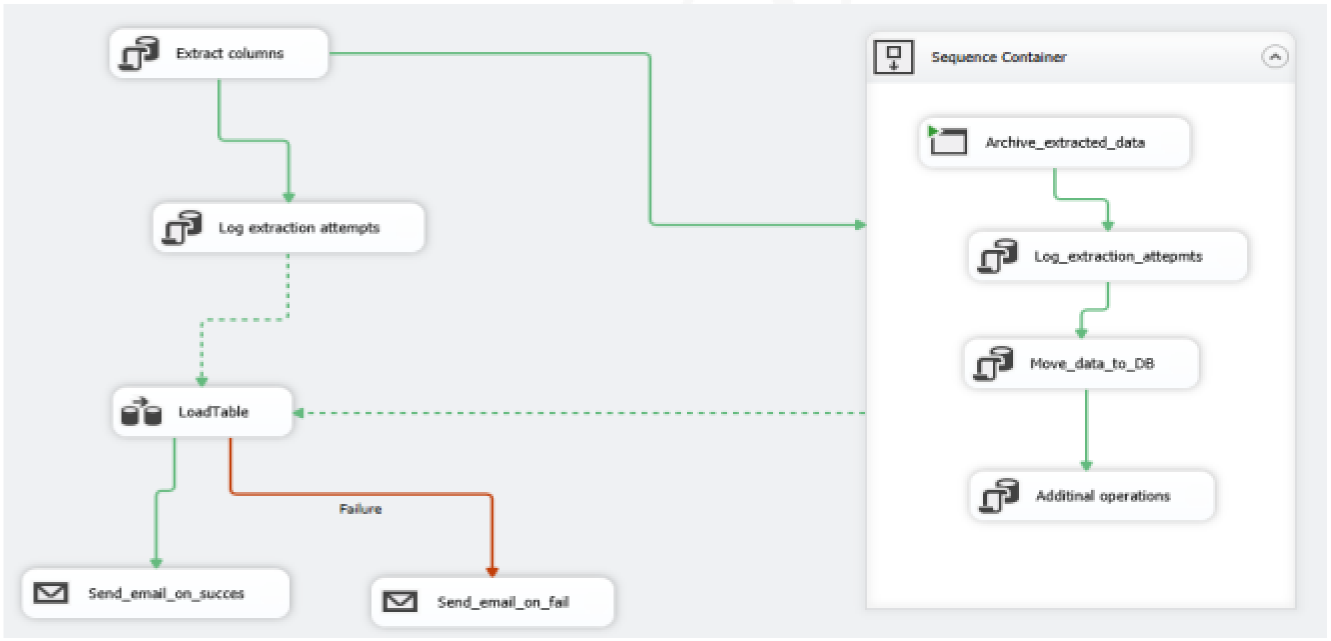
8. SSIS package control flow

Control flow is the SQL Server workflow engine that contains control flow elements. An SSIS package consists of at least one control flow task, and optionally one or more data flows.

There are three types of control flow elements:

- **Containers** – provide structure to package and services tasks. They support repeating control flow tasks, and grouping into meaningful units. A single container can be created inside another container, along with additional tasks. Depending on the type, it could be used to repeat tasks for each element in a collection, repeat tasks until the specified condition is met, or the container can group tasks and other containers in units that must succeed or fail when finished.
- **Control flow tasks** – workflow objects that perform a high level of operations, such as sending an email message, executing a SQL statement, or copying file from a FTP server. If the package contains more than one control flow task, they are connected and sequenced with a precedence constraint. When the control flow task is finished, it either succeeds or fails.
- **Precedence constraints** – connect tasks, executable, and containers inside the control flow, and specify a condition that determines whether the task will run or not. Precedence constraints can be configured by logical AND or logical OR expressions, and succeed or fail. One task can be connected to another with multiple precedence constraints, and for each constraint, a separate condition can be specified

Similar to documenting data flow, documenting a control flow task means to define a descriptive name for the



task, add more about the task in the **Description** field, or to add an annotation. When this information needs to be accessed, the SSIS package needs to be opened. This could be a problem if the package contains numerous control flow tasks with multiple data flow tasks.

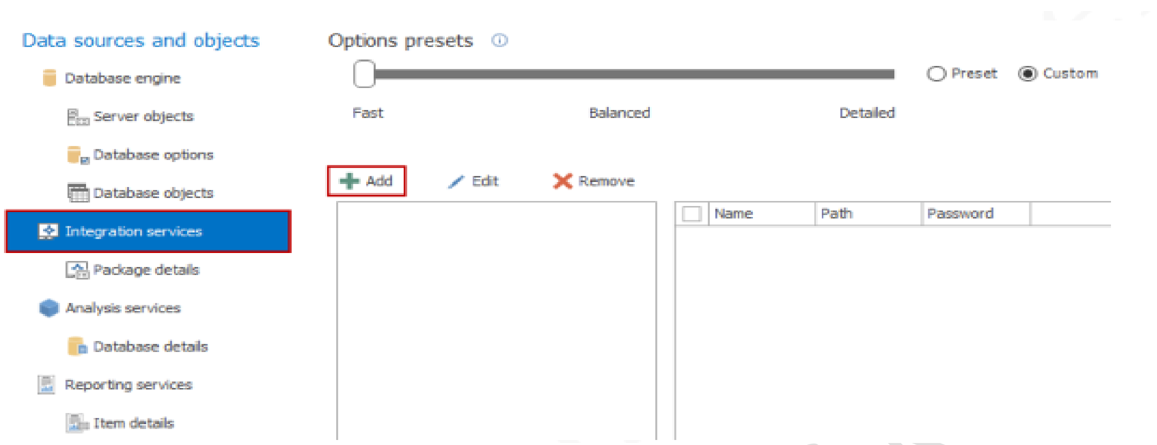
Apex SQL Doc is a documenting tool for SSIS packages, SSRS items, SSAS cubes and SQL databases, Tableau sites and SharePoint farms that generates user-friendly documentation in various formats: CHM, HTML, DOC, DOCX, and PDF.

Documenting SSIS packages includes properties for each SSIS task, both control flow and data flow diagrams. SSIS package sources supported by Apex SQL Doc are SQL Server, file system, and SSIS Package Store. The documenting process can be specified using the Command line Interface and scheduled to run unattended.

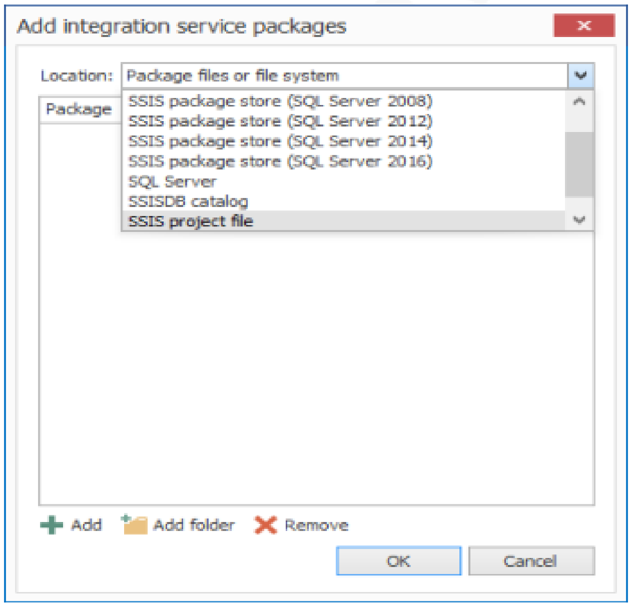
To document a SSIS control flow diagram:

1. Start Apex SQL Doc as an administrator

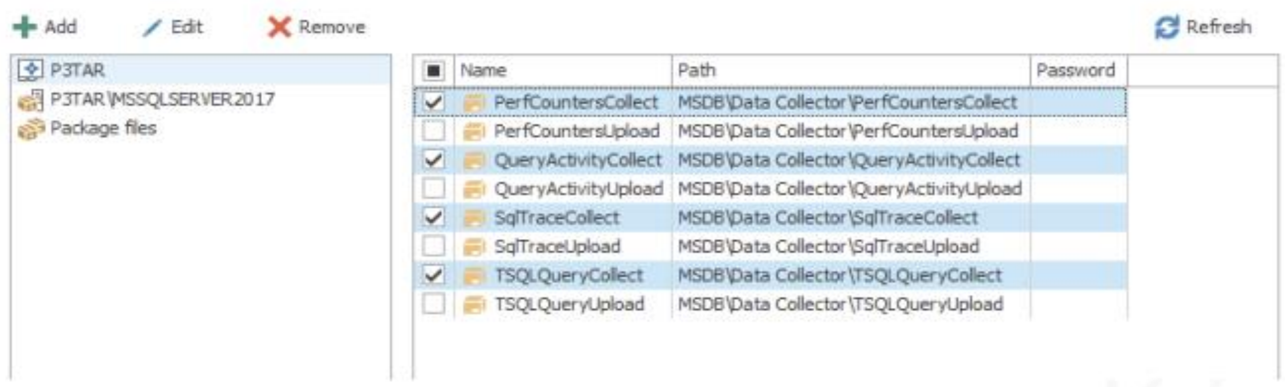
2. Click the **New** button from the Home tab, to start a new project
3. Initiate SSIS package documenting by clicking the **Add** button under the **Integration services** tab:



4. Select the appropriate package source from the **Add integration services packages** dialog, navigate to the package you want to document, and click the **OK** button:

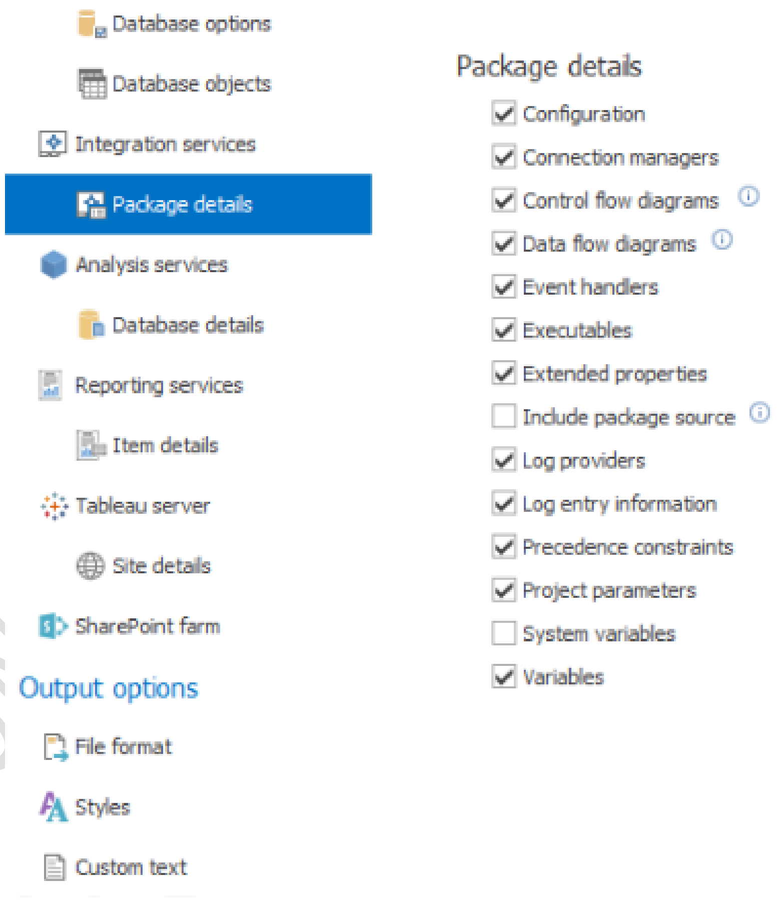

Add integration service packages
Location: Package files or file system
Package
SSIS package store (SQL Server 2008)
SSIS package store (SQL Server 2012)
SSIS package store (SQL Server 2014)
SSIS package store (SQL Server 2016)
SQL Server
SSISDB catalog
SSIS project file
Add
Add folder
Remove
OK
Cancel

When the package is included, it appears in the **Package selection** section:



■	Name	Path	Password
✓	PerfCountersCollect	MSDB\Data Collector\PerfCountersCollect	
	PerfCountersUpload	MSDB\Data Collector\PerfCountersUpload	
✓	QueryActivityCollect	MSDB\Data Collector\QueryActivityCollect	
	QueryActivityUpload	MSDB\Data Collector\QueryActivityUpload	
✓	SqlTraceCollect	MSDB\Data Collector\SqlTraceCollect	
	SqlTraceUpload	MSDB\Data Collector\SqlTraceUpload	
✓	TSQLQueryCollect	MSDB\Data Collector\TSQLQueryCollect	
	TSQLQueryUpload	MSDB\Data Collector\TSQLQueryUpload	

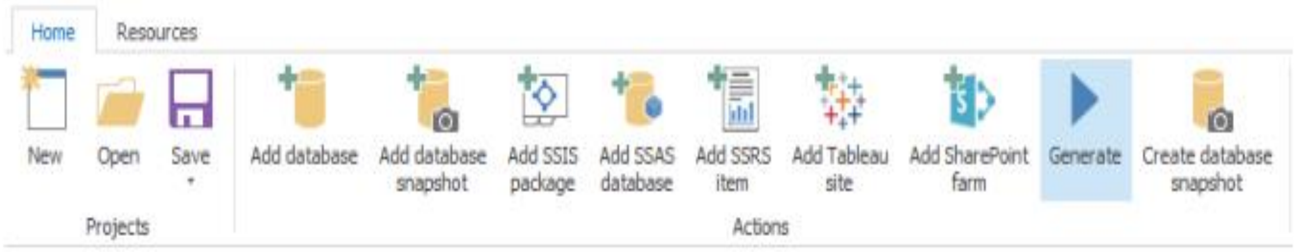
6. The **Package details** section offers filtering package attributes that will be included in the documenting process:



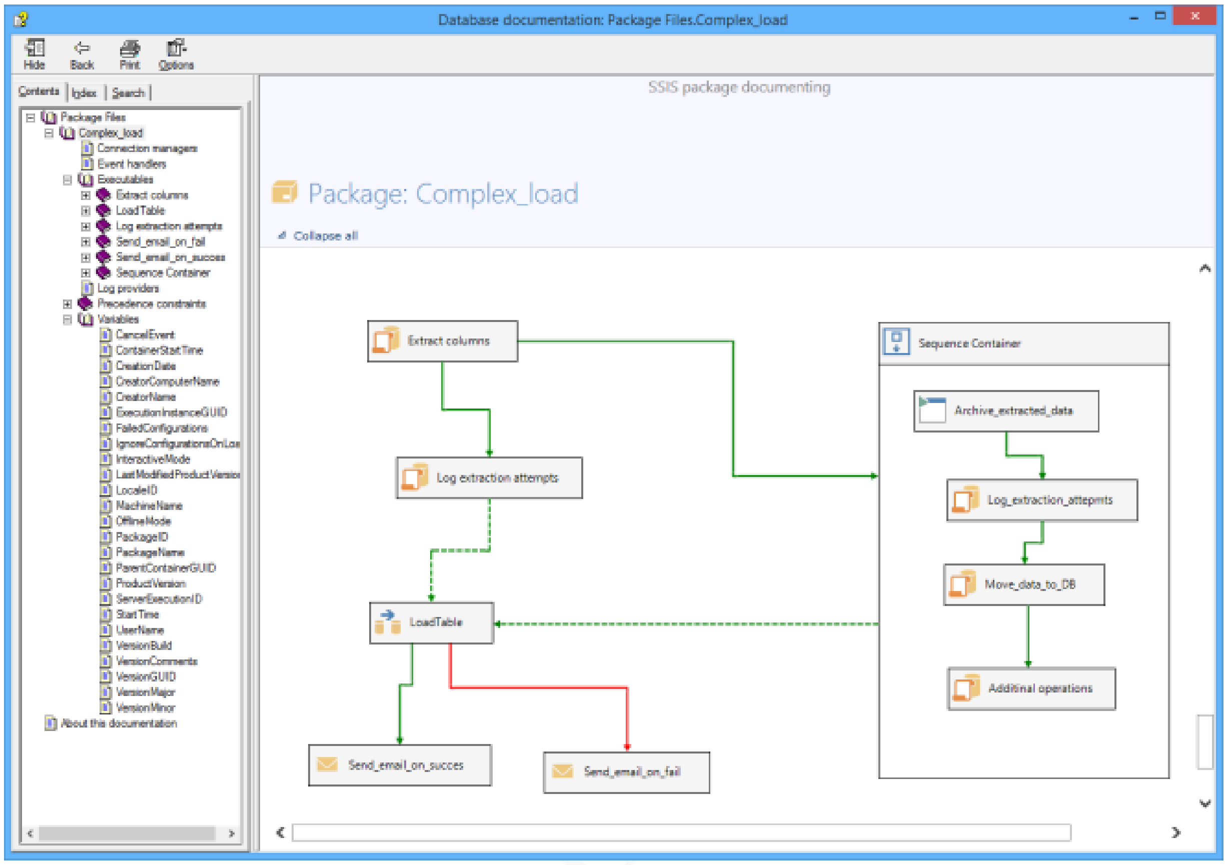
Package details

- [✓] Configuration
- [✓] Connection managers
- [✓] Control flow diagrams
- [✓] Data flow diagrams
- [✓] Event handlers
- [✓] Executables
- [✓] Extended properties
- [ ] Include package source
- [✓] Log providers
- [✓] Log entry information
- [✓] Precedence constraints
- [✓] Project parameters
- [ ] System variables
- [✓] Variables

7. When all of these are set, clicking the **Generate** button from the **Home** tab initiates the documenting process:



8. In the output document, navigate to the SSIS package node, and scroll down the page to see the Control flow diagram:



9. Containers in SSIS

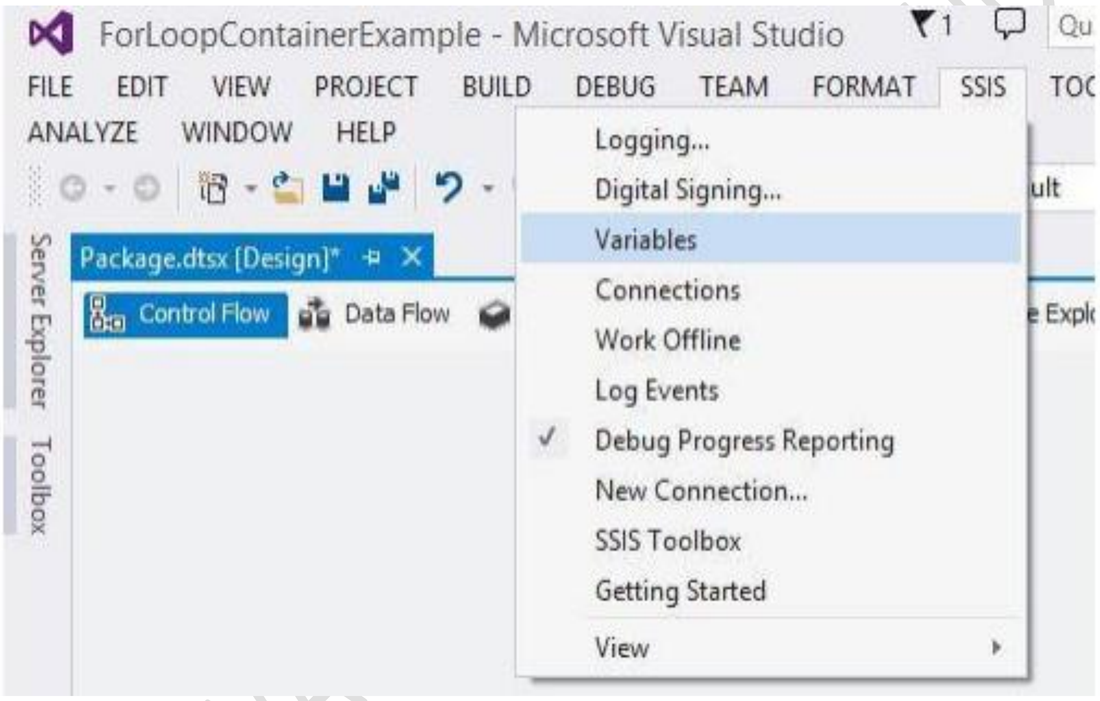
10. For Loop Container

The SQL Server Integration Services (SSIS) For Loop Container will allow us to repeat a task or tasks x number of times as if we had written a "for loop" in most programming languages. This tip was written using SQL Server 2016 Community Technology Preview 2.0’s SQL Server Data Tools in Visual Studio Ultimate 2013. The steps shown here will also work in previous versions of SSIS.

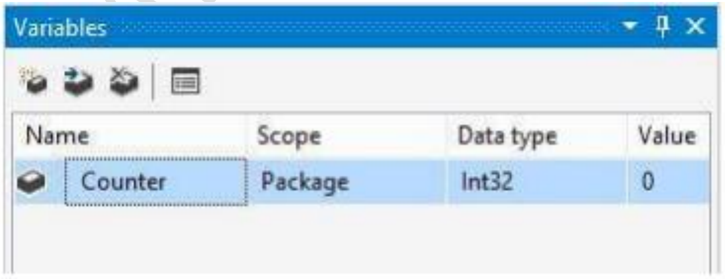
To begin this tip, I used the T-SQL below to create a table that will allow us to watch for changes in our control loop variable.

```
CREATE TABLE [dbo].[tblForLoopExample](
[LoopExampleKey] [int] PRIMARY KEY IDENTITY (1, 1) NOT NULL,
ControlLoopVariableValue int,
[LastUpdated] [datetime]
)
```

In SSIS, we need to create a variable for our counter. Select Variable from the SSIS menu as shown below.

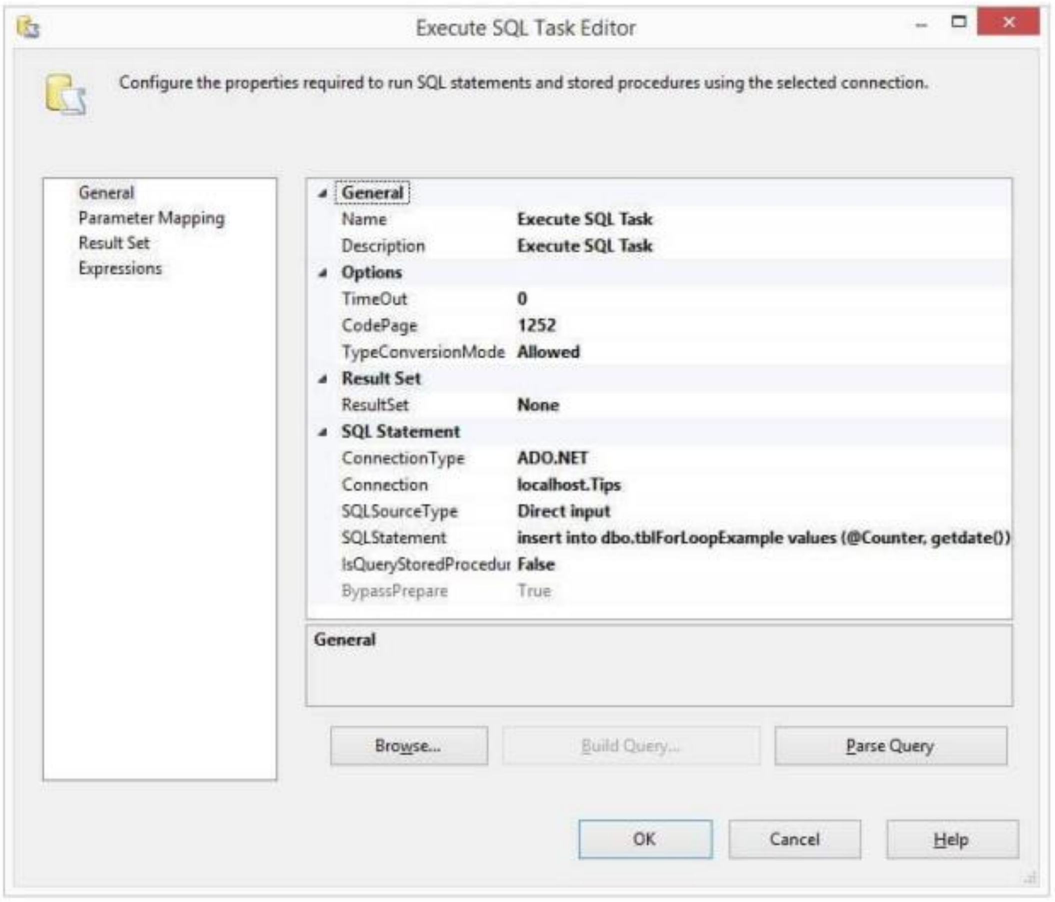


We will add a variable named Counter with the Int32 data type.



Name	Scope	Data type	Value
Counter	Package	Int32	0

In this tip, we will use the For Loop Container to repeat an Execute SQL Task. Drag an Execute SQL Task onto the Control Flow palette and double-click to display the General page of the Execute SQL Task Editor. Select ADO.NET as the Connection Type and create an ADO.NET connection if one does not already exist.

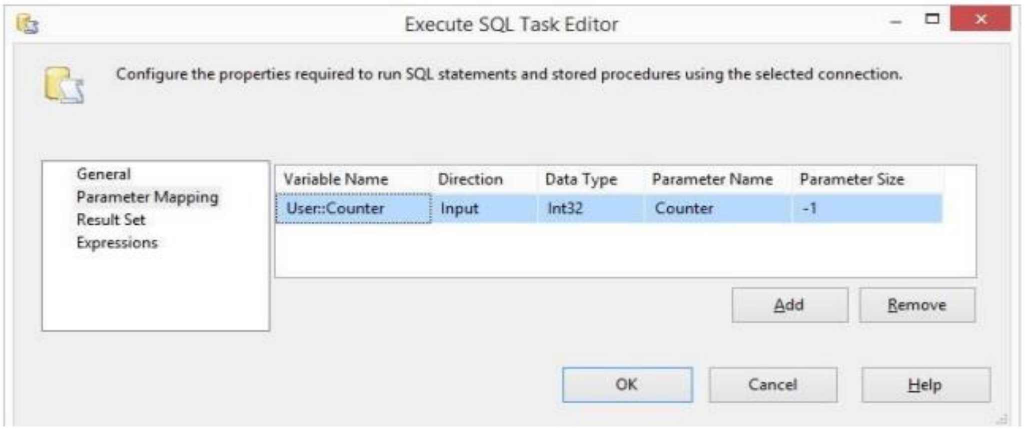


General	
Name	Execute SQL Task
Description	Execute SQL Task
Options	
TimeOut	0
CodePage	1252
TypeConversionMode	Allowed
Result Set	
ResultSet	None
SQL Statement	
ConnectionType	ADO.NET
Connection	localhost.Tips
SQLSourceType	Direct input
SQLStatement	insert into dbo.tblForLoopExample values (@Counter, getdate())
IsQueryStoredProcedur	False
BypassPrepare	True

Then enter the following in the SQL Statement text box:

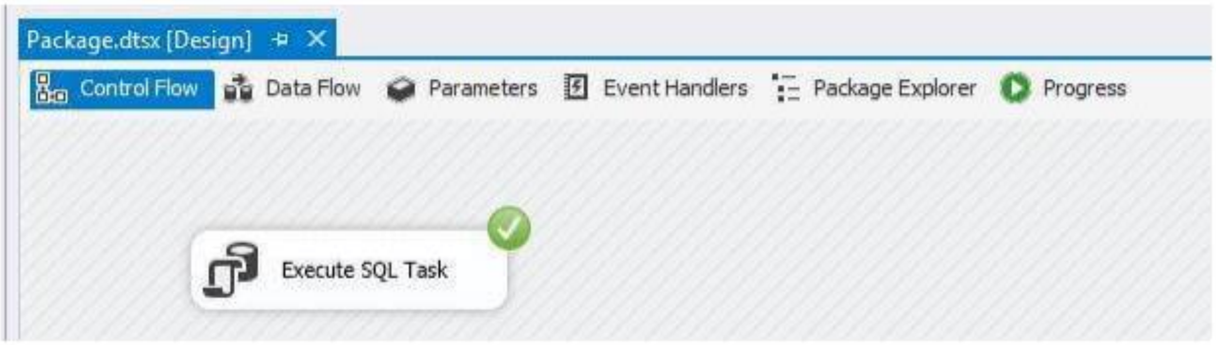
Insert into dbo.tblForLoopExample values (@Counter, getdate())

Click on Parameter Mapping on the left side of the Execute SQL Task Editor to display the Parameter Mapping page of the Execute SQL Task Editor. In the Variable Name column select User::Counter (it should be at or near the bottom of the drop-down list). Select Input as the Direction, Int32 as the Data Type, and enter "Counter" for the Parameter Name. We will leave the Parameter Size at -1. Click on OK when finished.

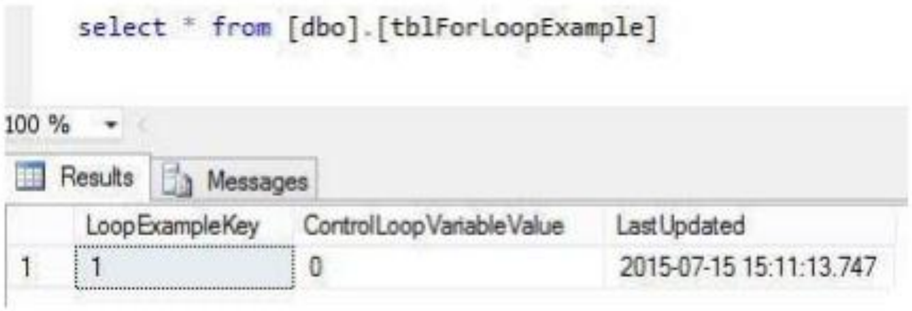

Execute SQL Task Editor
Configure the properties required to run SQL statements and stored procedures using the selected connection.
General
Parameter Mapping
Result Set
Expressions
Add
Remove
OK
Cancel
Help

Variable Name	Direction	Data Type	Parameter Name	Parameter Size
User::Counter	Input	Int32	Counter	-1

Before adding the For Loop Container, it is a good programming practice to make sure the code to be repeated in a loop works correctly before invoking the loop.



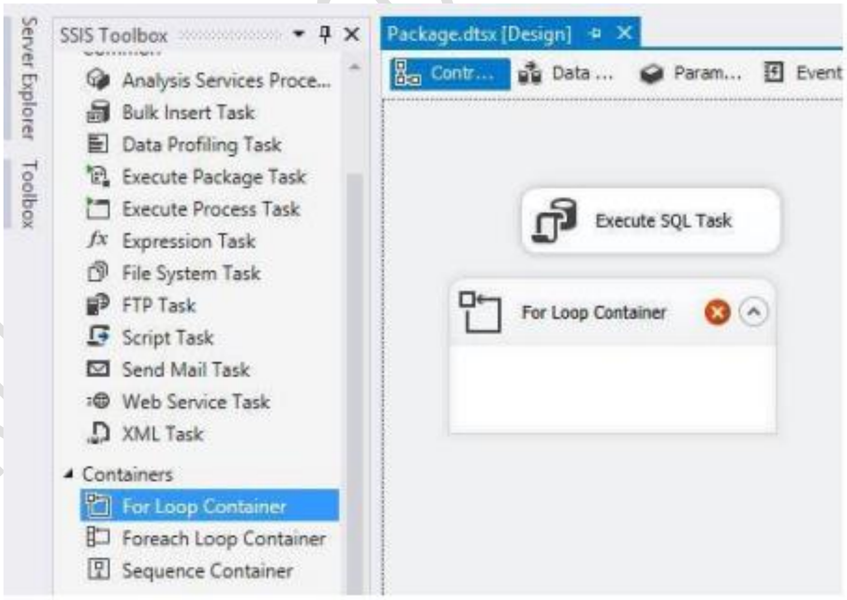
Running a select query on our test table shows that the INSERT statement worked correctly. Remember that the Counter variable is initialized to zero.



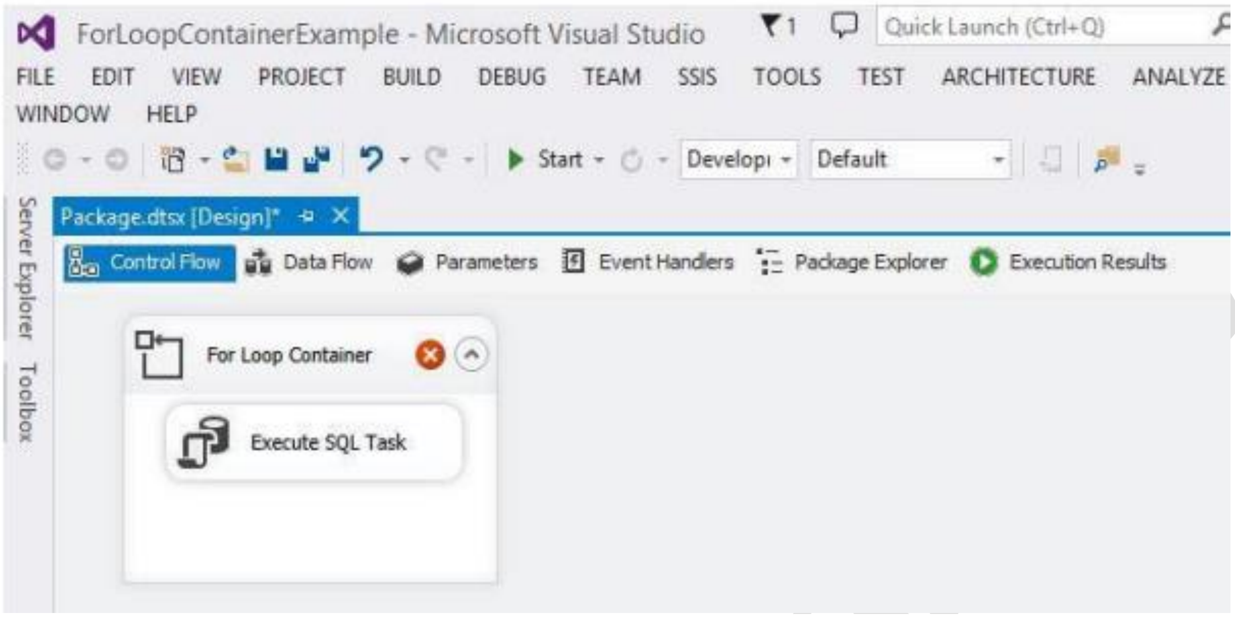
```
select * from [dbo].[tblForLoopExample]
```

	LoopExampleKey	ControlLoopVariableValue	LastUpdated
1	1	0	2015-07-15 15:11:13.747

Now we will drag a For Loop Container from the SSIS Toolbox to the Control Flow palette.

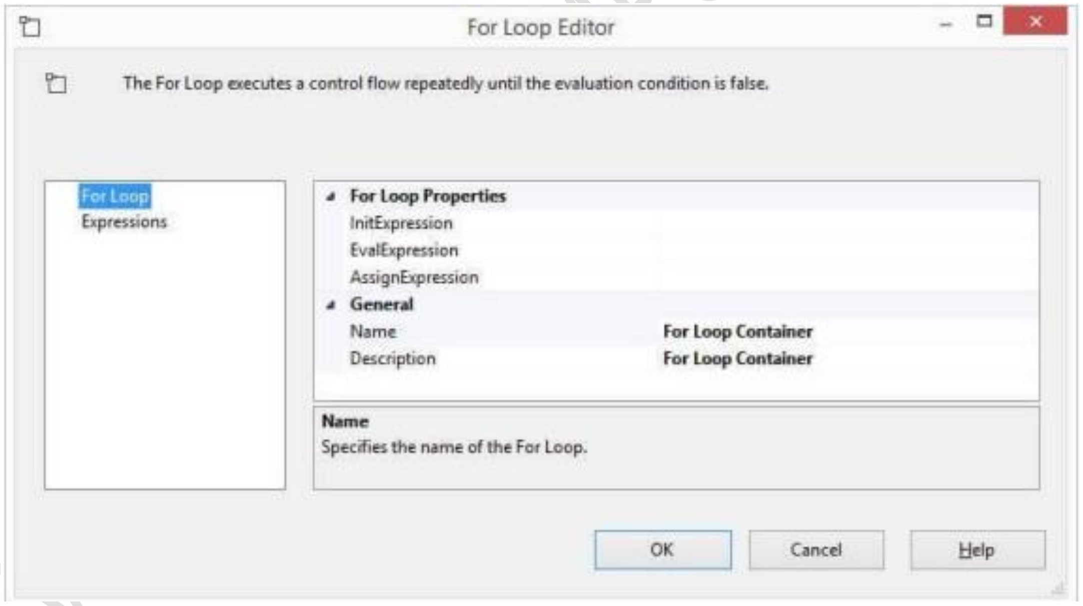


Next, we need to drag the Execute SQL Task onto the For Loop Container.


ForLoopContainerExample - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM SSIS TOOLS TEST ARCHITECTURE ANALYZE WINDOW HELP
Package.dtsx [Design]*
Control Flow
Data Flow
Parameters
Event Handlers
Package Explorer
Execution Results
For Loop Container
Execute SQL Task

The red circle with white X indicates that we need to configure the For Loop Container. Double-click on the For Loop Container to display the For Loop Editor window. We will set values for the InitExpression, EvalExpression and AssignExpression so the Execute SQL Task executes 10 times.

For the Init Expression, we will initialize the Counter value to zero.

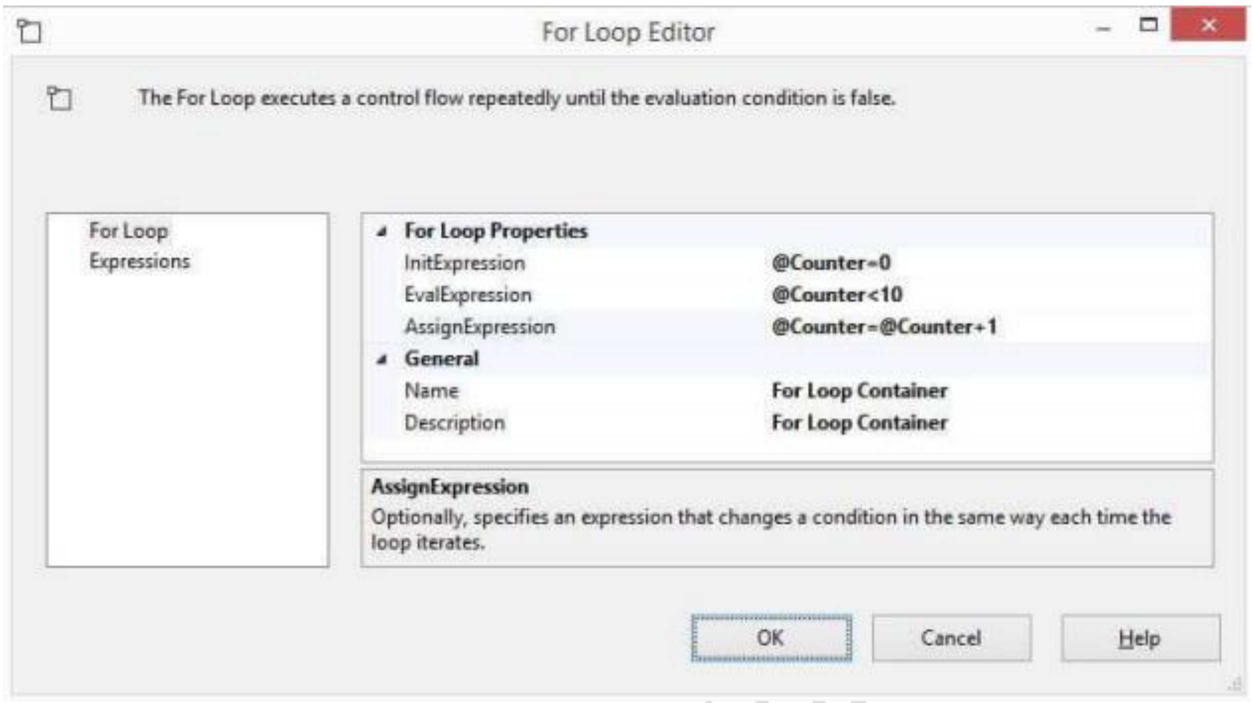

For Loop Editor

The For Loop executes a control flow repeatedly until the evaluation condition is false.

For Loop Properties	
InitExpression	
EvalExpression	
AssignExpression	
General	
Name	For Loop Container
Description	For Loop Container

Name
Specifies the name of the For Loop.

- The EvalExpression needs to evaluate to True or False. For the EvalExpression, we will test to see if the Counter value is less than 10. When the EvalExpression evaluates to False, the for loop will stop.
- The Assign Expression is where we will increment our counter variable. In this tip, we will increment our counter by 1.
- Click on OK to save these configuration changes.

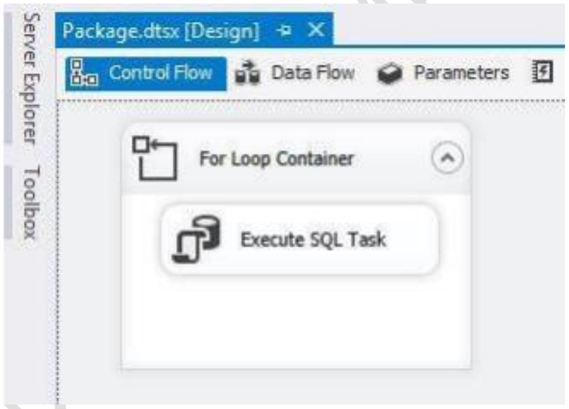

For Loop Editor

The For Loop executes a control flow repeatedly until the evaluation condition is false.

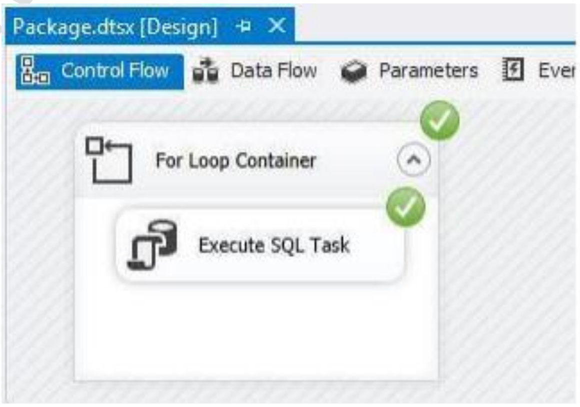
For Loop Properties	
InitExpression	@Counter=0
EvalExpression	@Counter<10
AssignExpression	@Counter=@Counter+1
General	
Name	For Loop Container
Description	For Loop Container

AssignExpression
Optionally, specifies an expression that changes a condition in the same way each time the loop iterates.

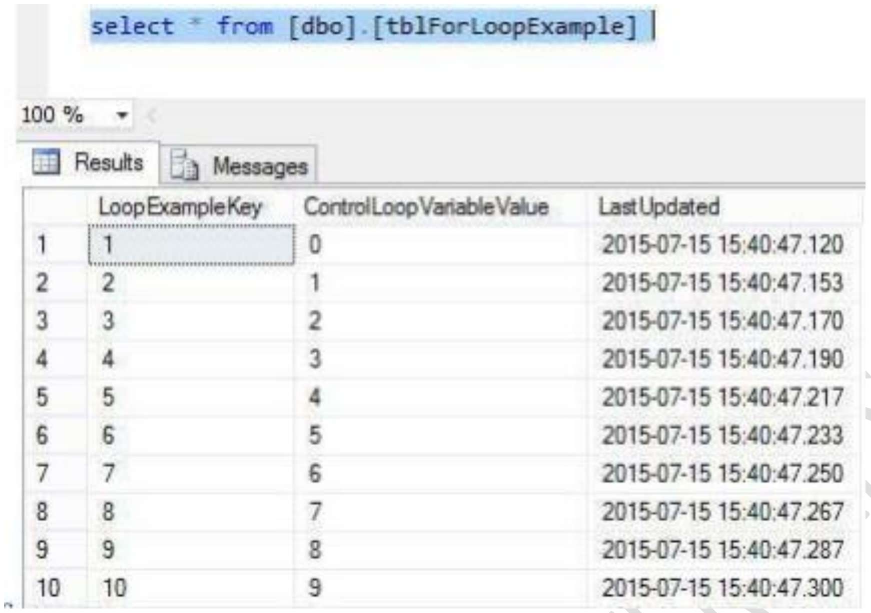
Notice that the red circle with the white X has disappeared. Now it is time to test the For Loop Container. Before testing, I dropped and recreated the test table that was created above.



The package runs successfully.



Querying the table shows that our for loop worked as designed.

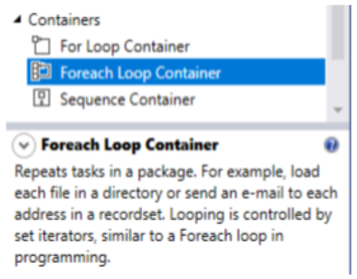


```
select * from [dbo].[tblForLoopExample]
```

	LoopExampleKey	ControlLoopVariableValue	LastUpdated
1	1	0	2015-07-15 15:40:47.120
2	2	1	2015-07-15 15:40:47.153
3	3	2	2015-07-15 15:40:47.170
4	4	3	2015-07-15 15:40:47.190
5	5	4	2015-07-15 15:40:47.217
6	6	5	2015-07-15 15:40:47.233
7	7	6	2015-07-15 15:40:47.250
8	8	7	2015-07-15 15:40:47.267
9	9	8	2015-07-15 15:40:47.287
10	10	9	2015-07-15 15:40:47.300

11. SSIS Foreach Loop Container

The SSIS Foreach Loop Container is more complicated than the For Loop Container since it has many use cases and requires a more complex configuration:



Foreach Loop Container

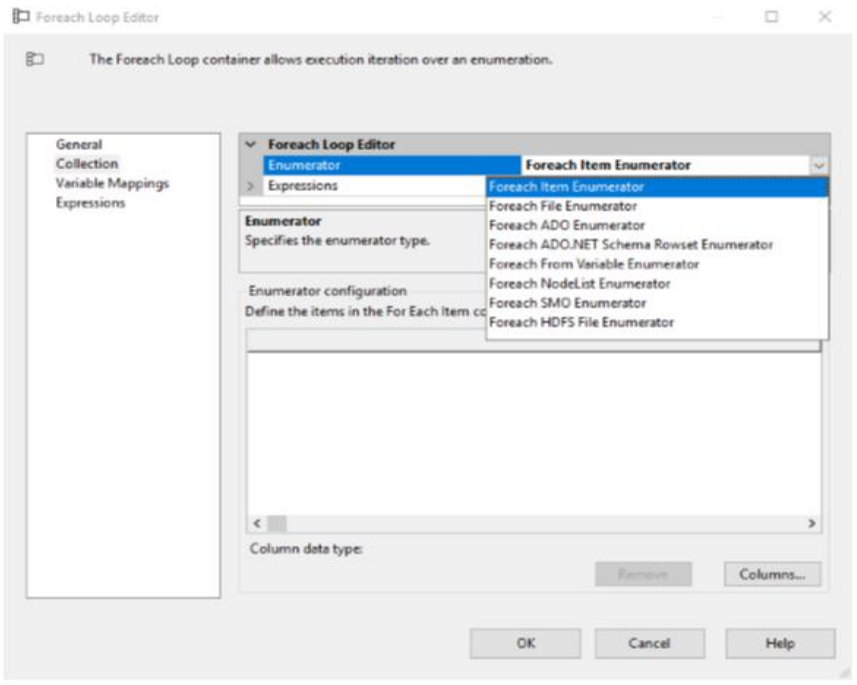
Repeats tasks in a package. For example, load each file in a directory or send an e-mail to each address in a recordset. Looping is controlled by set iterators, similar to a Foreach loop in programming.

Figure 4 – SSIS Foreach Loop Container description from the toolbox

There are different types of enumerators in the SSIS Foreach Loop Container. You can select the enumerator type from the collection tab within the SSIS Foreach Loop Container editor form:

- **Foreach item enumerator:** Loop over a set of items that can be defined manually within the SSIS Foreach Loop Container editor
- **Foreach File enumerator:** Loop over files within a specific directory
- **Foreach ADO enumerator:** Loop over file rows in an ADO Record set
- **Foreach ADO.NET Schema Rowset enumerator:** Loop over schema information from a specific data source (tables in a database)
- **Foreach from Variable enumerator:** Loop over items stored within an SSIS variable of type object (must be enumerable)

- **Foreach NodeList enumerator**: Loop over a result set of an XML Path Language (XPath) expression
- **Foreach SMO enumerator:** Loop over SQL Server Management Objects (SMO) objects, such as available servers
- **Foreach HDFS File enumerator**: Loop over files located within a Hadoop distributed file system directory
- **Foreach Azure Blob:** Loop over blobs in a blob container in Azure Storage
- **Foreach ADLS File:** Loop over files in a directory in Azure Data Lake Store
- **Foreach Data Lake Storage Gen2 File:** Loop over files in a directory in Azure Data Lake Store Gen2


Foreach Loop Editor

The Foreach Loop container allows execution iteration over an enumeration.

General
Collection
Variable Mappings
Expressions

Foreach Loop Editor
Enumerator: Foreach Item Enumerator
Expressions

Foreach Item Enumerator
Foreach File Enumerator
Foreach ADO Enumerator
Foreach ADO.NET Schema Rowset Enumerator
Foreach From Variable Enumerator
Foreach NodeList Enumerator
Foreach SMO Enumerator
Foreach HDFS File Enumerator

Enumerator
Specifies the enumerator type.

Enumerator configuration
Define the items in the For Each Item co

Column data type:

Remove
Columns...

OK
Cancel
Help

Figure 5 – SSIS Foreach Loop Container collection tab page

Each enumerator has its own properties that we must configure.

To catch the current item while looping over a collection, we must add an SSIS package and map this variable to the item within the SSIS Foreach Loop Container variable mappings tab page by specifying the item index within the current row and the variable name.

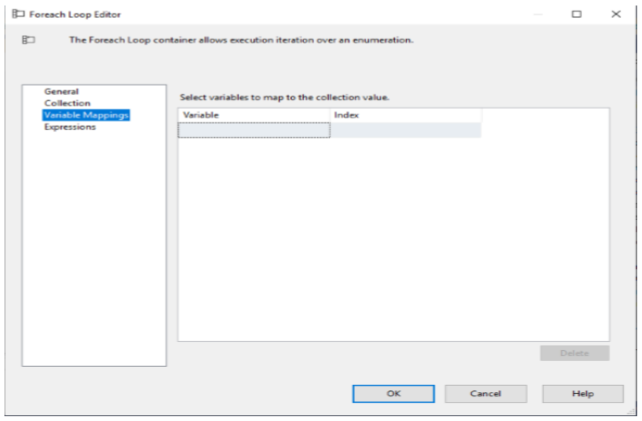

Foreach Loop Editor
The Foreach Loop container allows execution iteration over an enumeration.
General
Collection
Variable Mappings
Expressions
Select variables to map to the collection value.
Variable
Index
Delete
OK
Cancel
Help

Figure 6 – SSIS Foreach Loop Container variable mappings tab page

12. Sequence Container

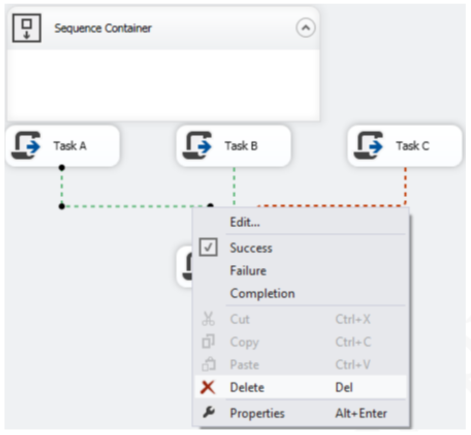
A container is a subset of the SSIS package in the control flow and it can contain zero or more tasks and containers. A sequence container likes a sub package and you can add, edit or remove tasks as you do in the package. It can be configured through its property window. The benefits of using a sequence container are summarized below.

- **One location Management**: Change a property of a sequence container instead of changing the same property of all the tasks in the container.
- **Variable Scope**: Variables defined in the container scope are only accessed by the tasks or containers in the sequence container.
- **Collapsing and Expanding**: Have this ability to group related tasks in a container to be managed easily. · Transaction: Easily to make the related tasks be finished in one transaction.

Now it is time to add a sequence container to the package above. Please follow the steps below.

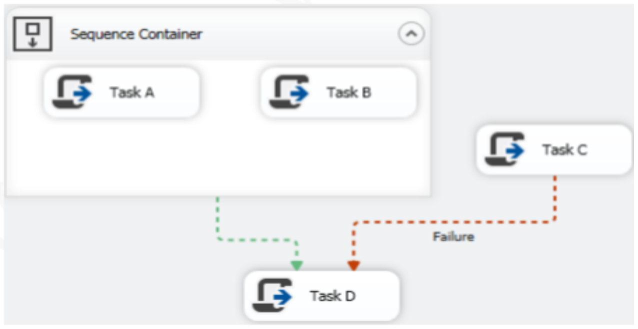
1. Open the package SequenceContainer.dtsx if it is closed. Then drag and drop a sequence container to the control flow of the package.

2. Right click the precedence constraint between task A and D, then choose "Delete" to delete the constraint.



3. Repeat the step 2 again to delete the constraint between task B and D too.

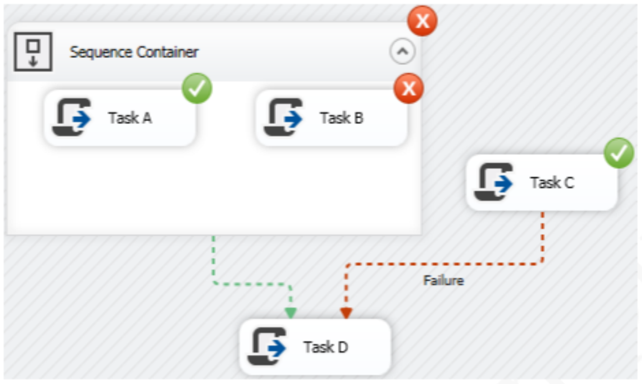
4. Drag and drop the task A and B to the sequence container and link the precedence constraint of the container to the task D.



5. Run the package and you will see the task D will run OK because both task A and B running successfully makes the whole container return success. At last, the logic OR makes the task D execute.

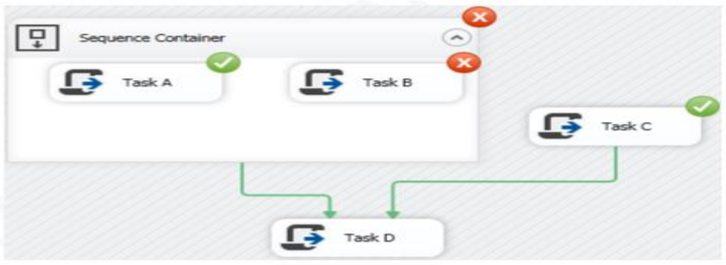
6. Click "Stop Debugging". Then change the code of task B like we did before to make it return failure instead of success.

7. Run the package again.



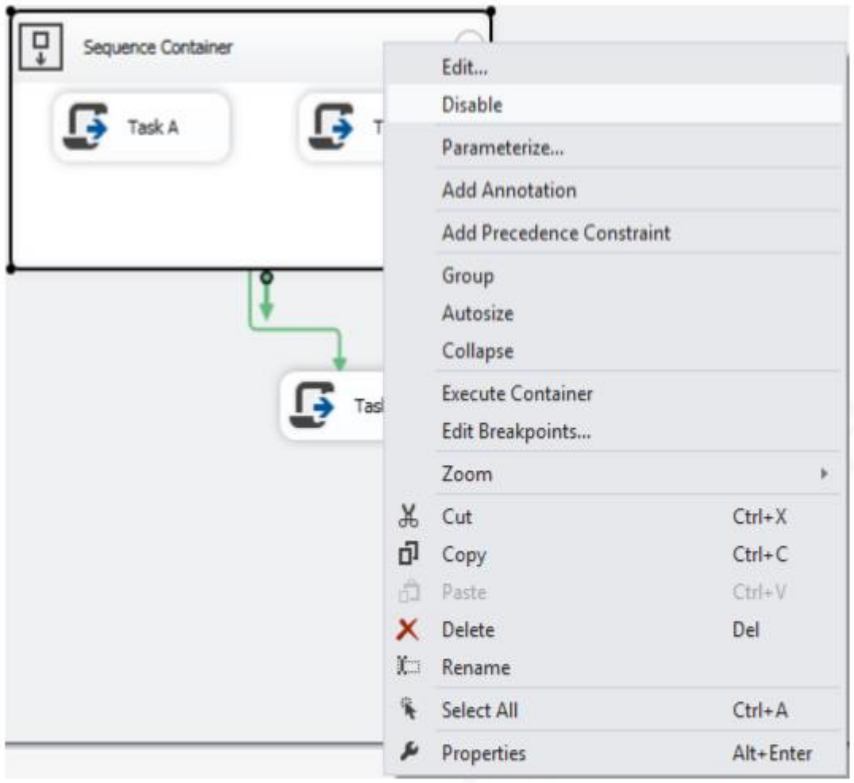
Task B failed running makes the whole sequence container return failure so the task D has never run.

8. Click "Stop Debugging". Then change the constraint between task C and D to Success and logic AND. Run the package again.



The task D still cannot run because of the failure the container returns.

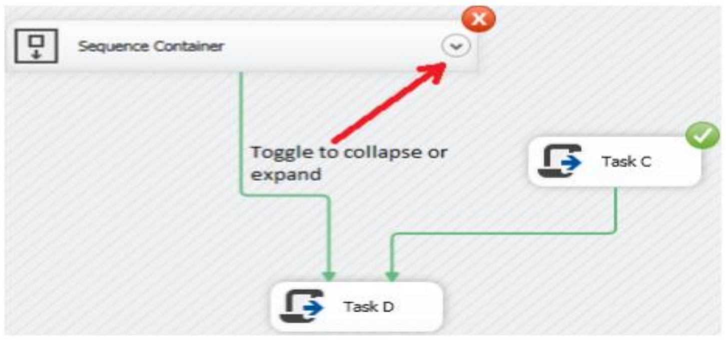
9. Click "Stop Debugging". Then right click the sequence container and choose "Disable" to disable the container.



10. Run the package and you will see the task D runs but the sequence container has never run because it was disabled. Click "Stop Debugging".

11. Right the container and choose "Enable" to enable it. Then click upper arrow at the top right Conner of the container to Collapse it.

12. Run the package.



You can click the arrow to toggle collapsing and expanding the sequence container.

13. Click "Stop Debugging" to terminate running the package.

13. Creating Dynamic Packages

When you are developing Microsoft SQL Server Integration Services (SSIS) packages, it is a good practice to make each task or transformation as dynamic as possible. This enables you to move your packages from one environment to another (for example, from development to a test environment, and then to a production environment) without opening and changing the package. You can also configure your packages to set different properties at run time.

To do this, you can use the new features in SQL Server 2012, such as parameters and project-level connection managers, or you can use the package configurations that were first made available in the package deployment model in earlier versions of SQL Server. This topic discusses both possibilities for designing and configuring your package to dynamically set values at run time. Using dynamic packages eliminates the need to make changes as you move from one environment to the other or to open the project when you want to run your packages using different property or variable values.

14. Parameters

In SQL Server 2012, SSIS introduces parameters. Parameters allow you to assign values to properties within packages at the time of package execution. They can also be used in SSIS expressions—for example, to use an Expression task to set a variable value based on the specific value of the parameter. There are two types of parameters: project parameters, which are created on the project level, and package parameters, which are created at the package level. You can use and set parameters only in projects developed for the project deployment model. When you are using the project deployment model, projects are deployed to the Integration Services catalog.

15. Defining Parameters

You add project or package parameters by using SSDT. Usually you create a project parameter by selecting Project.params in Solution Explorer under your project name, as shown in Figure 22. To add a package parameter, you must first open the package and then select the parameters tab in the package design area.

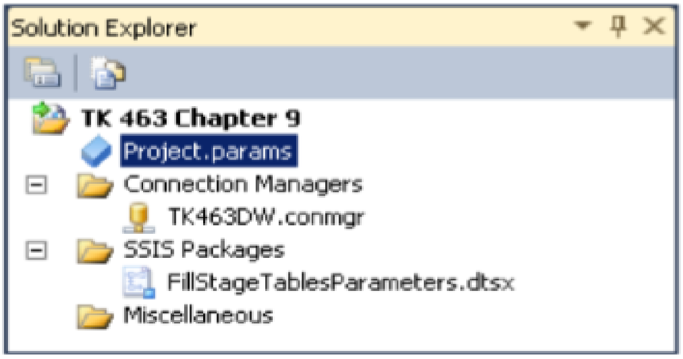
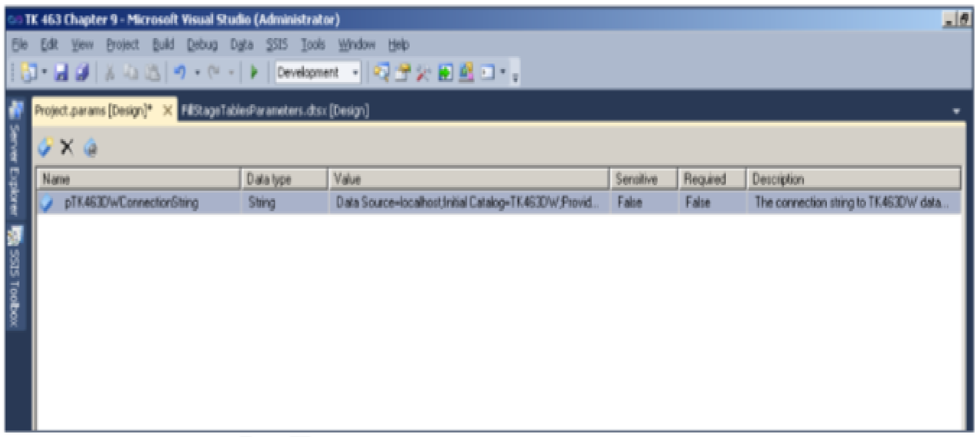


Figure 22 - Solution Explorer with Project.params selected

When you open the package or project parameters window, you can define a new parameter by clicking the Add Parameter icon (the first icon on the left; it looks like a blue cube). Figure 23 shows a parameter window with one parameter called pTK463DWConnectionString.


TK 463 Chapter 9 - Microsoft Visual Studio (Administrator)
File Edit View Project Build Debug Data SSIS Tools Window Help
Development
Project.params [Design]*
FillStageTablesParameters.dtsx [Design]
Server Explorer
SSIS Toolbox

Name	Data type	Value	Sensitive	Required	Description
pTK463DWConnectionString	String	Data Source=localhost;Initial Catalog=TK463DW;Provid...	False	False	The connection string to TK463DW data...

Figure 23 - The parameters window.

When you create a parameter in SSDT, there are several properties to specify:

■■ **Name** the name of the parameter. The first character of the name must be a letter or an underscore.
■■ **Data type** the data type of the parameter.
■■ **Value** The default value for the parameter assigned at design time. This is also known as the design default.
■■ **Sensitive** If the value of this property is true, parameter values are encrypted in the catalog and appear as NULL when viewed with Transact-SQL or SQL Server Management Studio.
■■ **required** requires that a value other than the design default be specified before the package can be executed.
■■ **Description** For maintainability, the description of the parameter. In SSDT, set the parameter description in the Visual Studio Properties window when the parameter is selected in the applicable parameters window.

You can edit parameters in the list in the parameter window, or you can use the Properties window to modify the values of parameter properties. You can delete a parameter by using the Delete Parameter toolbar button. By using the Add Parameters To Configurations toolbar button (the last button on the right), you can specify a value for a parameter for a specific build configuration that is used only when you execute the package in SQL Server Data Tools.

16. Use a Parameter in the Data Flow Task

In this exercise, you create a package-level parameter that will be used to filter source data. You will use this parameter in the data flow task to filter only rows from the source for which the year of the modified date is equal to or greater than the parameter value.

1. If necessary, start SQL Server Data Tools, open any project, and then open the .dtsx package from the previous exercise for editing.
2. Select the Parameters tab and click the Add Parameter button on the toolbar. 3. Name the parameter by setting the Name property to pYear and set the Data Type property to Int16. For the Value property, enter 2002.
4. Click the Data Flow tab and open the Person OLE DB Source adapter.
5. In the OLE DB Source Editor, change the data access mode to SQL Command and enter the following SELECT statement to retrieve the necessary rows and use a parameter placeholder inside the query.

```
SELECT
BusinessEntityID,
PersonType,
NameStyle,
Title,
FirstName,
MiddleName,
LastName,
Suffix,
EmailPromotion,
AdditionalContactInfo,
Demographics,
rowguid,
ModifiedDate
FROM
Person.Person
WHERE
YEAR (ModifiedDate) >=?
```

6. Click the Parameters button to open the Set Query Parameters dialog box.
7. For the Variables property, select the $Package::pYear parameter as the source for the query parameter. Click OK twice to close the window and the OLE DB Source Editor.
8. Execute the package and observe the number of rows displayed in the data flow area. 9. Change the parameter value to 2008 and execute the package. Notice that fewer rows are read from the OLE DB Source adapter in the data flow area.

QUESTION BANK

Q-1: 1 Mark Question

1. Give the full form of SSDT. (July-2021,2019)
2. Give the full form of ETL. (July-2021,2019)(2023)
3. Give the full form of DTS. (July-2021)
4. Give the full form of SSIS. (July-2021,2019)(2023)
5. What is Transformation? (July-2021)
6. Types of extraction. (July-2021)
7. List types of Container(July-2021,2020,2019)
8. What is SSIS Data Flow? (July-2021)
9. In SSIS, Extension of Packages is__. (BKNMU- april-2020)
10. ETL stands for _________. (BKNMU - april-2022)
11. SSIS stands for _________. (BKNMU - april-2022)
12. Write full form of SSDT. (BKNMU - april-2022)
13. SMO stands for_____

Q-2: 3 Mark Question

1. Give basic steps for creating package. (July-2021)
2. Explain ETL in details. (July-2021)
3. Explain metadata. (July-2021)
4. Give the list of components of data flow task. (July-2021)
5. Explain types of loading. (April–2020)
6. Give the brief note on SSIS Data flow Source and Destination. (April–2020)
7. Explain SSTS tasks. (BKNMU - June-2021,2019)
8. Explain Foreach loop container (BKNMU - June-2021,2019)
9. Give basic steps for creating package. (BKNMU - June-2021)
10. Explain Sequence container control. (BKNMU - april-2020)(2023)
11. Explain SSIS task types. (BKNMU - april-2022)(2023)
12. Explain for loop container. (BKNMU - april-2022)
13. What is data flow and control flow- in SSIS? (BKNMU - april-2022)
14. Discuss common sources and destination for the data flow. (BKNMU - april-2022)
15. What is SSIS? Discuss SSIS packages. (2023)
16. Write a note on flat file. (2023)

Q-3: 5 Mark Question

1. Explain Control flow. (July-2021,2019)
2. What is Container? Explain its types and purpose. (July-2021)
3. Explain SSIS Control flow. (April–2020)(2023)
4. Write a note on ETL, process. (BKNMU - June-2021,2019)(2023)
5. Write a step to insert record into table using for loop container control. (BKNMU- april-2020,2019)
6. Explain for loop container control with example. (BKNMU - april-2020)
7. Explain how to creating Dynamic Packages.
8. Explain ETL Process. (BKNMU - april-2022)
9. Explain foreach loop Container. (BKNMU - april-2022)