Unit-3

Practical – 1: Extract and Load Product table from AdventureWorksDW of SQL Server Database to Excel file.

Step-1: Start SSDT, using Integration Service Project.

Step-2: Design Data Flow Task from SSIS Toolbox into Control Flow Task, then Double Click on it to open Data Flow.

Step-3: Design ADO NET Source from SSIS Toolbox.

Step-4: Design Excel Destination from SSIS Toolbox.

Step-5: To configure ADO NET Source, double Click on it.

Step-6: Configure ADO.NET connection manager, so that it connect to the Database AdventureWorksDW, then Select Table or view from Data Access Mode option and select DimProduct Table from the list.

Step-7: Connect ADO NET Source to Excel Destination using blue arrow.

Step-8: To configure Excel Destination double click on it.

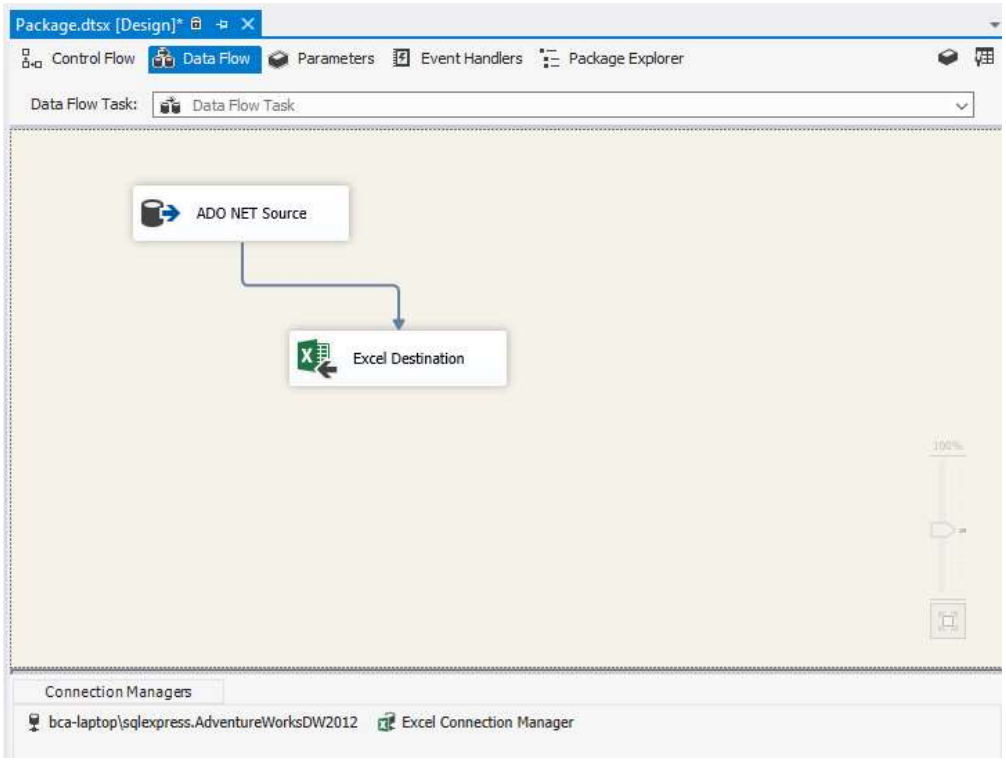
Step-9: Configure Excel Destination, double click on it.

Step-10: Configure Excel connection manager so that it connect to ProductList.xls file.

Step-11: From Data access mode select Table or view, and then Select appropriate sheet from the drop down list.

Step-12: From Mappings match columns.

Step-13: Execute Package to load data into excel file.



Practical – 2: Extract, Transfer and Load Product table from AdventureWorksDW of SQL Server Database to Excel file.

Step-1: Start SSDT, using Integration Service Project.

Step-2: Design Data Flow Task from SSIS Toolbox into Control Flow Task, then Double Click on it to open Data Flow.

Step-3: Design ADO NET Source from SSIS ToolBox.

Step-4: Design Excel Destination form SSIS ToolBox.

Step-5: Configure ADO NET Source by double click on it.

Step-6: Configure ADO.NET connection manager, so that it connect to the Database AdventureWorksDW, then Select Table or view from Data Access Mode option and select DimProduct Table from the list.

Step-7: Design Derived Column from the Common section of SSIS ToolBox.

Step-8: Double click on Derived Column to configure, so that ProductName will be transform to UpperCase.

Step-9: Connect ADO NET Source to Derived Column and Derived Column to Excel Destination.

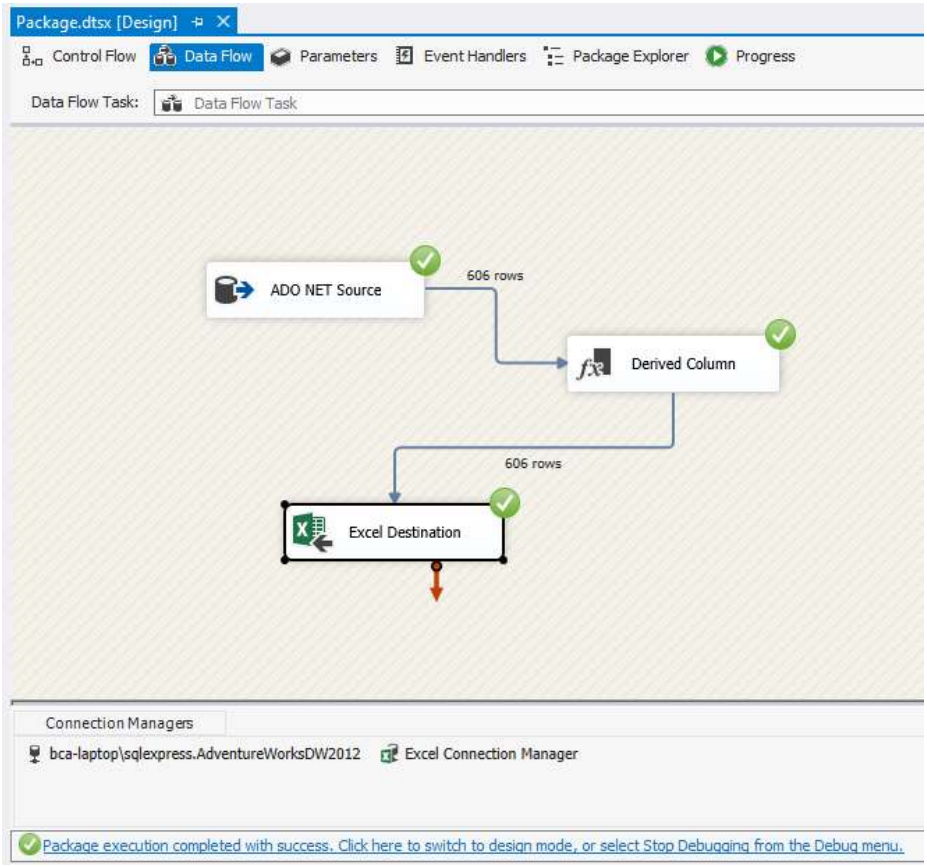
Step-10: Double Click on Excel Destination to configure it.

Step-11: Configure Excel connection manager so that it connect to ProductList.xls file.

Step-12: From Data access mode select Table or view, and then Select appropriate sheet from the drop down list.

Step-13: From Mappings match columns.

Step-14: Execute Package to load data into excel file.



Practical – 3: Extract and Load Employee table from AdventureWorksDW of SQL Server Database to Flat file.

Step-1: Start SSDT, using Integration Service Project.

Step-2: Design Data Flow Task from SSIS Toolbox into Control Flow Task, then Double Click on it to open Data Flow.

Step-3: Design ADO NET Source from SSIS ToolBox.

Step-4: Design Flat File Destination form SSIS ToolBox.

Step-5: Configure ADO NET Source by double click on it.

Step-6: Configure ADO.NET connection manager, so that it connect to the Database AdventureWorksDW, then Select SQL Command Data Access Mode option and give appropriate query to extract rows from DimEmployee Table.

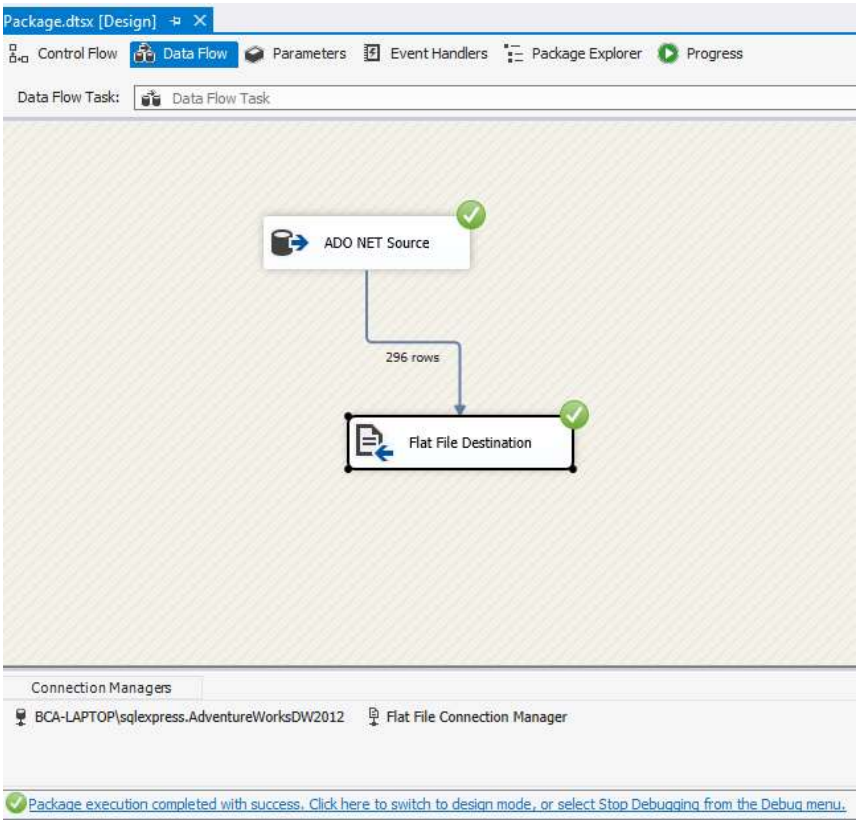
Step-7: Connect ADO NET Source to Flat File Destination.

Step-8: Double Click on Flat File Destination to configure it.

Step-9: Configure Flat File connection manager so that it connect to EmpList.txt file.

Step-10: From Mappings match columns.

Step-11: Execute Package to load data into excel file.



Practical – 4: Extract, Transfter and Load Employee table from AdventureWorksDW of SQL Server Database to Flat file.

Step-1: Start SSDT, using Integration Service Project.

Step-2: Design Data Flow Task from SSIS Toolbox into Control Flow Task, then Double Click on it to open Data Flow.

Step-3: Design ADO NET Source from SSIS ToolBox.

Step-4: Design Flat File Destination form SSIS ToolBox.

Step-5: Configure ADO NET Source by double click on it.

Step-6: Configure ADO.NET connection manager, so that it connect to the Database AdventureWorksDW, then Select Table or view from Data Access Mode option and select DimEmployee Table from the list.

Step-7: Design Derived Column from SSIS Toolbox to Data Flow

Step-8: Double click on Derived Column to configure it. Give two expression as combine FirstName, MiddleName, and LastName as Employee Name, and increase BaseRate to 10%

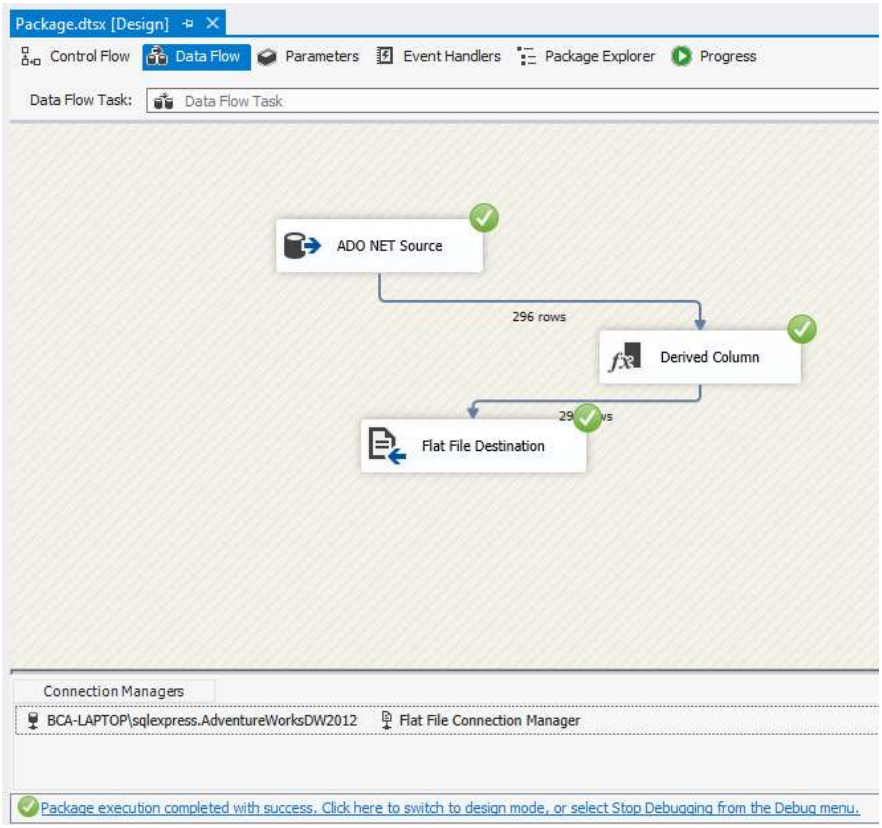
Step-9: Join ADO NET Source to Derived Column and Derived Column to Flat File Destination.

Step-10: Double Click on Flat File Destination to configure it.

Step-11: Configure Flat File connection manager so that it connect to EmpList.txt file.

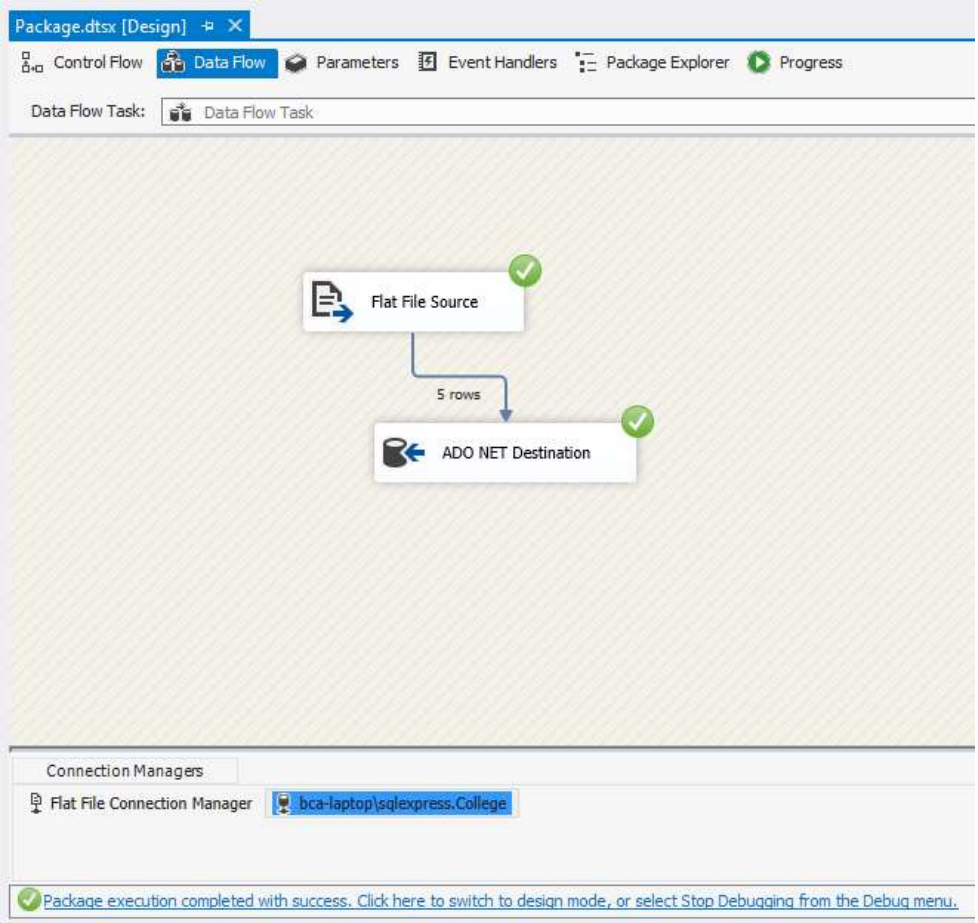
Step-12: From Mappings match columns.

Step-13: Execute Package to load data into excel file.



Practical – 5: Extract and Load Students Data from Flat File to SQL Server DB: College Table: Students

Step-1: Start SSDT, using Integration Service Project.
Step-2: Design Data Flow Task from SSIS Toolbox into Control Flow Task, then Double Click on it to open Data Flow.
Step-3: Design Flat File Destination form SSIS ToolBox.
Step-4: Design ADO NET Source from SSIS ToolBox.
Step-5: Double Click on Flat File Destination to configure it.
Step-6: Configure Flat File connection manager so that it connect to Student.txt file.
Step-7: Configure ADO NET Source by double click on it.
Step-8: Connect ADO NET Source to Flat File Destination.
Step-9: Configure ADO.NET connection manager, so that it connect to the Database College, then Student Table.
Step-10: From Mappings match columns.
Step-11: Execute Package to load data into SQL Server



Practical – 6: Extract and Load Data from Product, ProductCategory, and ProductSubCategory data into Ms-Access Database.

Step-1: Start SSDT, using Integration Service Project.

Step-2: Design Data Flow Task from SSIS Toolbox into Control Flow Task, then Double Click on it to open Data Flow.

Step-3: Design ADO NET Source from SSIS ToolBox.

Step-4: Design ADO NET Destination form SSIS ToolBox.

Step-5: Configure ADO NET Source by double click on it.

Step-6: Configure ADO.NET connection manager, so that it connect to the Database AdventureWorksDW, then Select SQL Command Data Access Mode option and give appropriate query to extract rows from DimProduct, DimProductCateogry, and DimProductSubCategory Table.

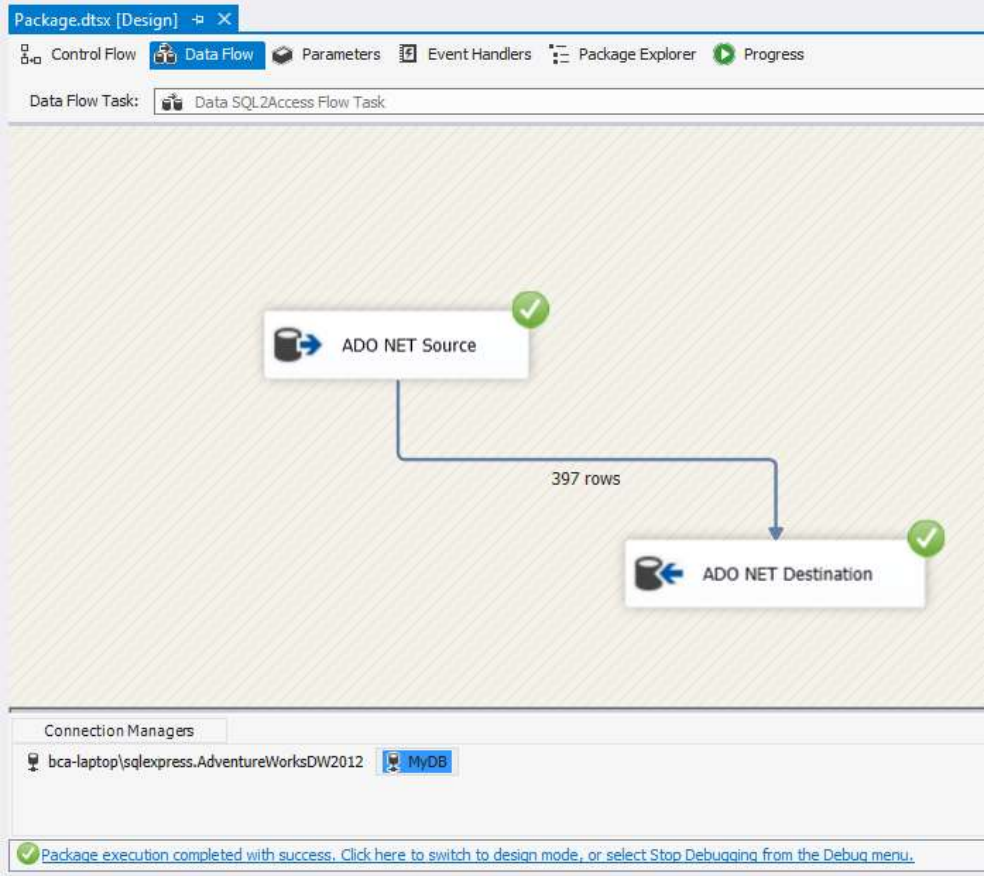
Step-7: Connect ADO NET Destination.

Step-8: Double Click on ADO Net Destination to configure it.

Step-9: Configure ADO Net connection manager so that it connect to MyDB.mdb file, of MS-Access

Step-10: From Mappings match columns.

Step-11: Execute Package to load data into MS-Access.



Unit-4

Practical – 7: Check for Data Quality Issues in adventureWorksDW2012.

In this practice, you analyze part of the AdventureWorksDW2012 sample database and a production system you are familiar with for potential data quality issues. In this theoretical exercise, you check for potential data quality issues in the AdventureWorksDW2012 sample database.

Step-1: Start SSMS and connect to your SQL Server instance. Expand the Databases folder and then the AdventureWorksDW2012 database.

Step-2: Expand the Tables folder. Expand the Columns folder for the following tables:
- DimCustomer
- FactInternetSales
- DimDate
- DimSalesReason
- DimProduct
- DimProductCategory
- DimEmployee
- FactResellerSales
- DimDate

Step-3: Identify which of the columns you have expanded could have problems with completeness.

Step-4: After you have finished with the review of the AdventureWorksDW2012 tables, close SSMS.

Practical – 8: installing Data quality services.

In the following exercises, you will install Data Quality Server and Data Quality Client. This practice assumes that your computer meets the prerequisites for this installation. In the exercise, you use SQL Server Setup to install DQS components.

Step-1: Start SQL Server 2012 Setup.

Step-2: In the SQL Server Installation Center, select the Installation tab (second from the top on the left side).

Step-3: Select the New SQL Server Stand-Alone Installation Or Add Features To Existing Installation link.

Step-4: Wait until the Setup Support Rules have been checked. When the operation is finished, click OK.

Step-5: In the Product Updates window, click Next.

Step-6: Wait while the setup files are installed. The Setup Support Rules page of SQL Server 2012 Setup should appear. Correct any errors and check any warnings. If there are no errors, click Next.

Step-7: On the Installation Type page, select the Add Features To An Existing Instance Of SQL Server 2012 option. Select the instance on which you want to host the DQS databases from the drop-down list of installed instances on your computer. Click Next.

Step-8: On the Feature Selection page, select the check boxes for the Data Quality Services and Data Quality Client options. Then click Next.

Step-9: Wait while the installation rules are checked.

Step-10: On the Disk Space Requirements page, click Next.

Step-11: On the Error Reporting page, clear the only check box and click Next.

Step-12: On the Installation Configuration Rules page, click Next.

Step-13: On the Ready to Install page, click Install.

Step-14: Wait until installation is finished.

Step-15: On the Complete page, click Close.

Step-16: Close the SQL Server Installation window.

Practical – 9: Check Whether an Excel File Exists

This example determines whether the Excel workbook file specified in the ExcelFile variable exists, and then sets the Boolean value of the ExcelFileExists variable to the result. You can use this Boolean value for branching in the workflow of the package.

Step-1: Add a new Script task to the package and change its name to ExcelFileExists.

Step-2: In the Script Task Editor, on the Script tab, click ReadOnlyVariables and enter the property value using one of the following methods:

Type ExcelFile.

-or-

Click the ellipsis (...) button next to the property field, and in the Select variables dialog box, select the ExcelFile variable.

Step-3: Click ReadWriteVariables and enter the property value using one of the following methods:

Type ExcelFileExists.

-or-

Click the ellipsis (...) button next to the property field, and in the Select variables dialog box, select the ExcelFileExists variable.

Step-4: Click Edit Script to open the script editor.

Step-5: Add an Imports statement for the System.IO namespace at the top of the script file.

Step-6: Add the following code.

Code:

```
public class ScriptMain
{
  public void Main()
  {
    string fileToTest;
    fileToTest = Dts.Variables["ExcelFile"].Value.ToString();
    if (File.Exists(fileToTest))
    {
      Dts.Variables["ExcelFileExists"].Value = true;
    }
    else
    {
      Dts.Variables["ExcelFileExists"].Value = false;
    }

    Dts.TaskResult = (int)ScriptResults.Success;
  }
}
```