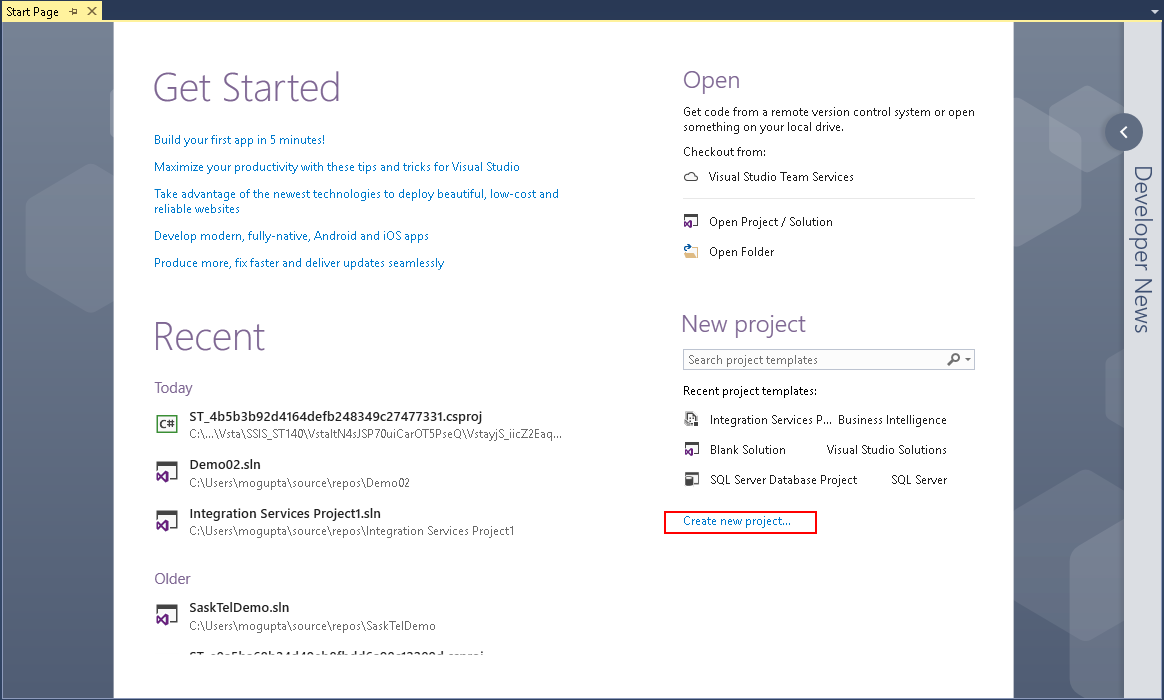
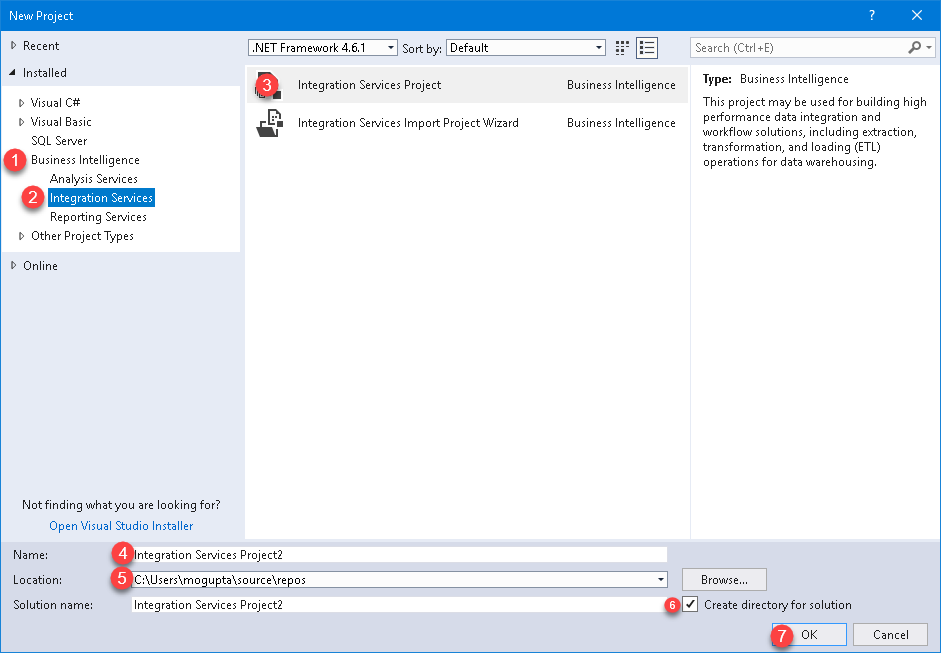
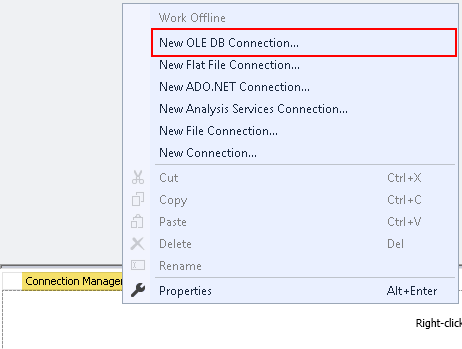
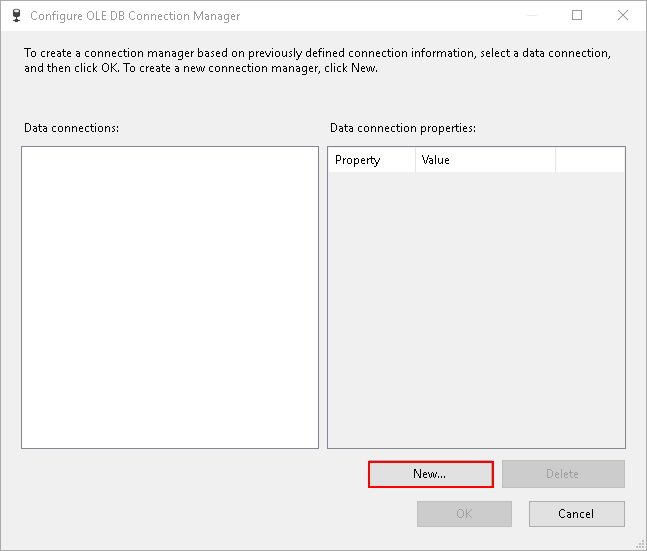
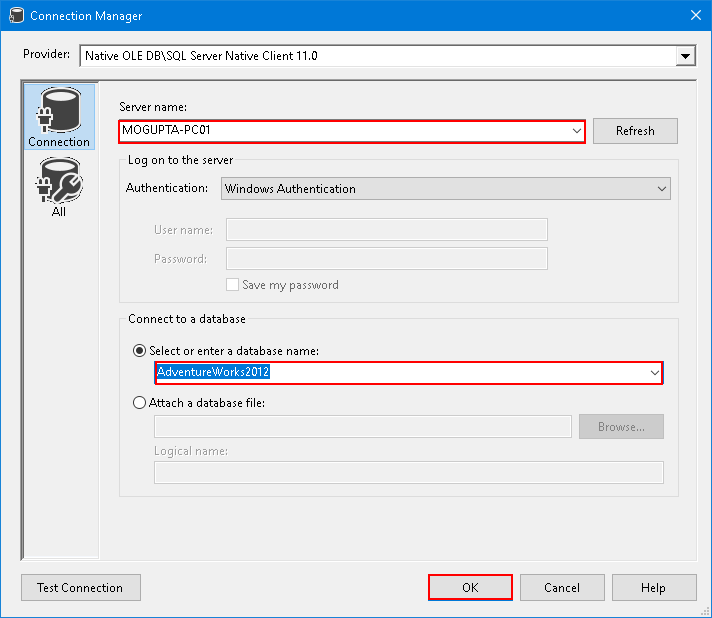
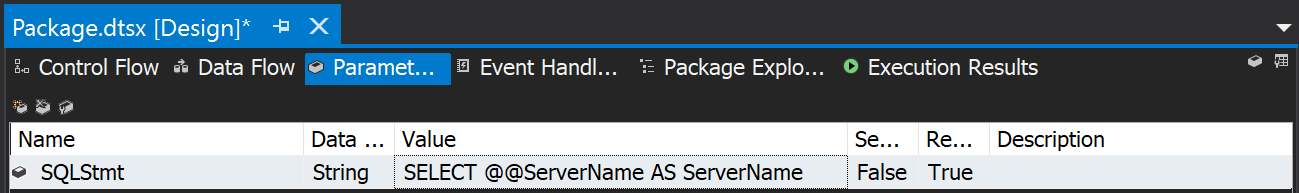
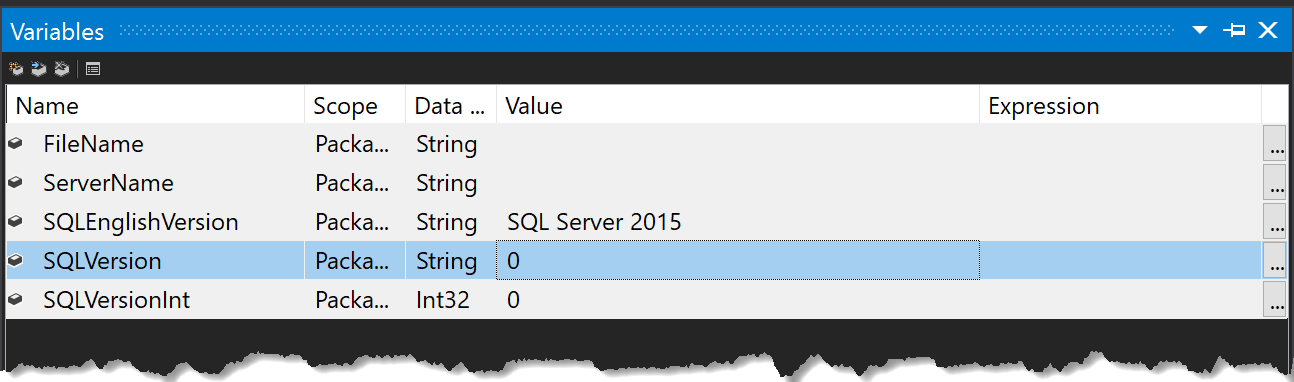
SQL Server Integration Services

# Module 03: CONTROL FLOW: Bring It Together

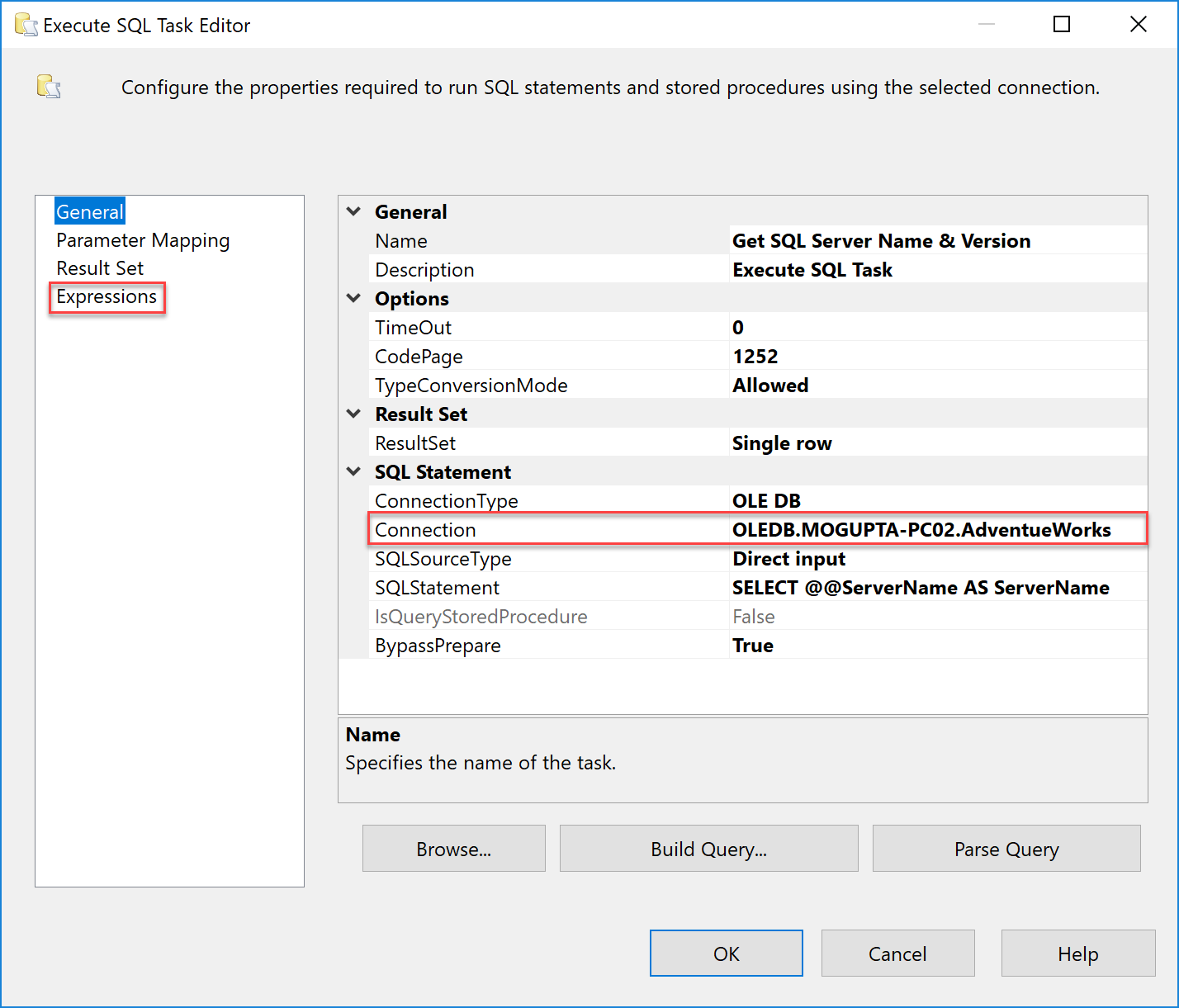
1. Launch SQL Server Data Tools (SSDT), under start menu look for Visual Studio 2017 (SSDT).
2. In the Start Page, click Create new project.  
   
3. In New Project dialog box select Business Intelligence > Integration Services > Integration Services Project. On the bottom enter in project name and location you wish to save the project. Make sure “Create directory for solution” is selected and click OK.



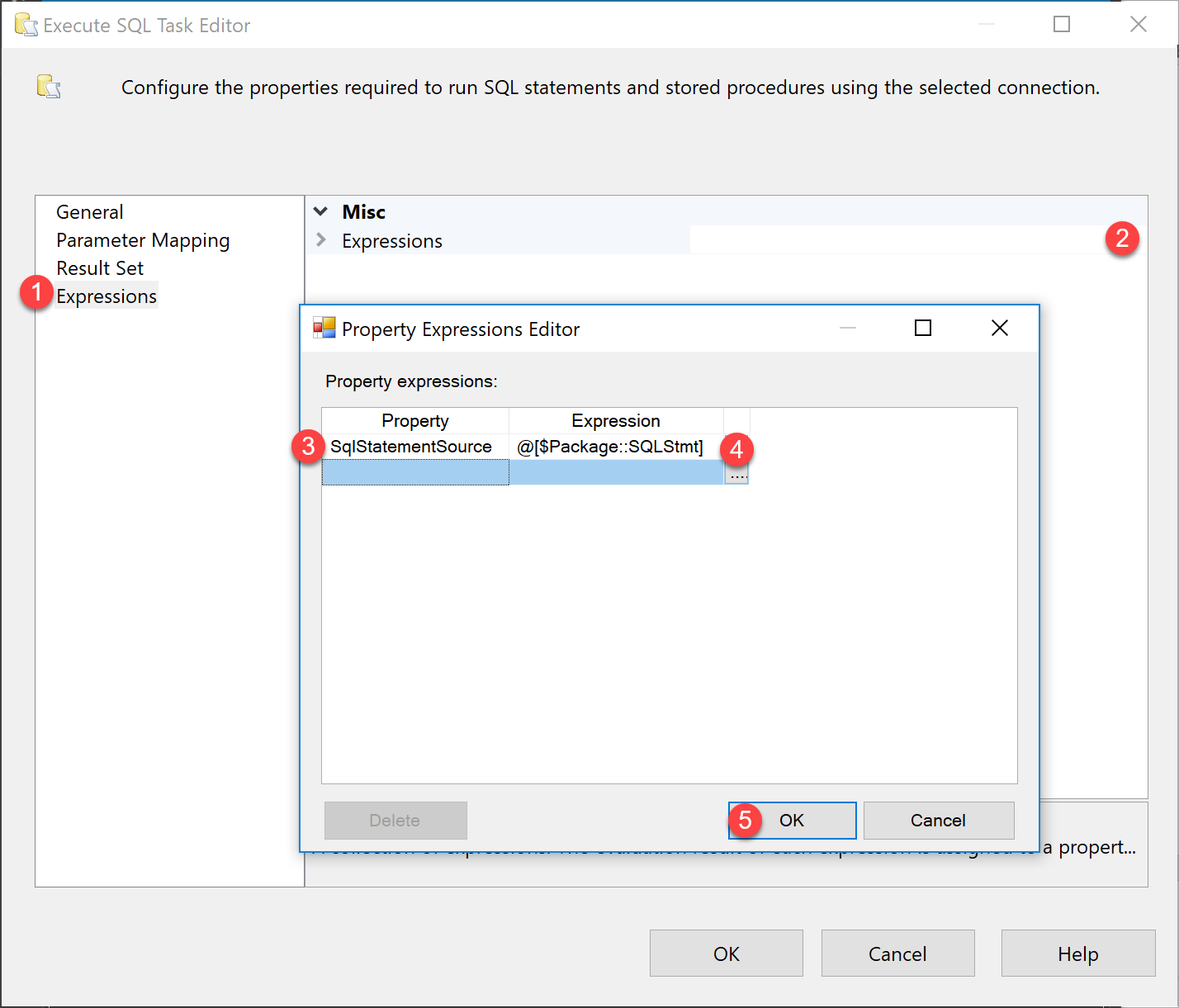
1. Let’s set up a connection manager to our database. In the bottom center pain under Connection Manger, right-click select New Ole-DB Connection.  
   
2. In Configure OLE DB Connect Manager, click New.  
   
3. In connection manager, type the server name, select the database “AdventureWorks2012” and click OK.  
   
4. Click OK in Configure OLE DB Connection Manager. You should see a single connection under Connection Manager.  
     
   *Hint: Rename the connection manager to OLEDB.ServerName.DatabaseName. This will make it easier to identify which driver is being used for the driver.*
5. Setup a Package Parameter assign it value “SELECT @@ServerName AS ServerName”. Set Required to True.  
   
6. Setup variables for the project. Go to SSIS Menu > Variables. Then create following variables in Package Scope and their respective types (Note there is default value for SQLEnglishVersion).

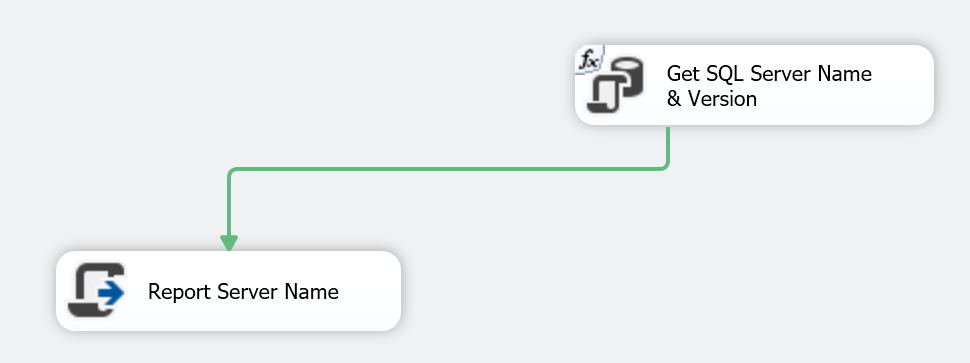


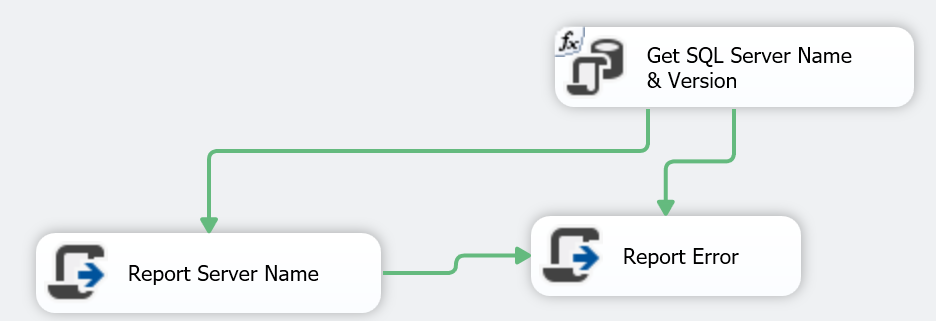
1. Create an Execute SQL Task. Rename it to “Get SQL Server Name & Version”.  
     
   *Hint: You can rename a task by double-clicking and updating the Name attribute. You can also select the Task and press F2 to rename it.*
2. Connect the “Get SQL Server Name & Version” to OLEDB connection created in step #7 and go to Expressions.

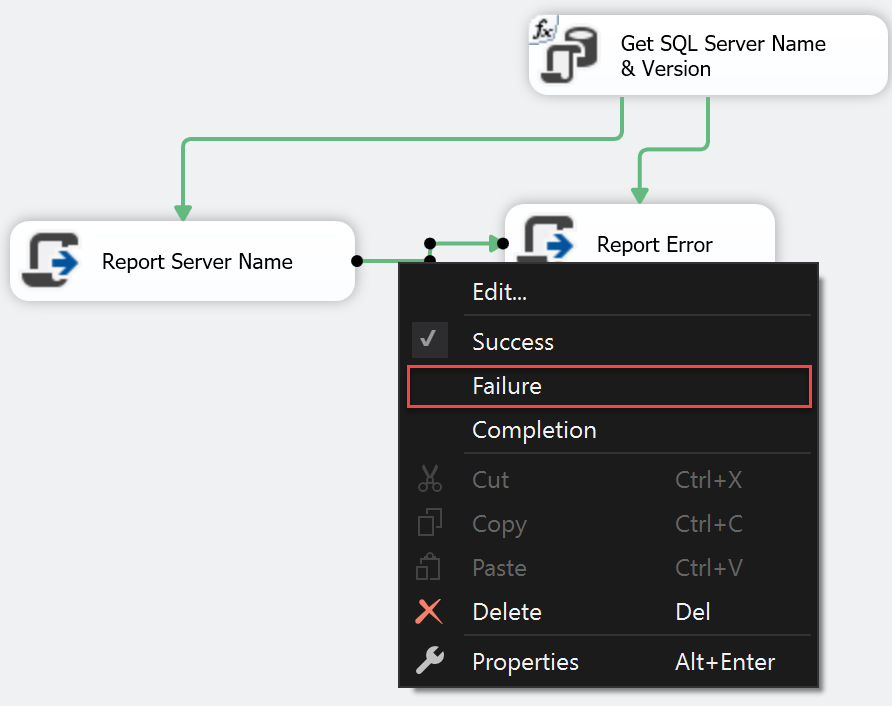


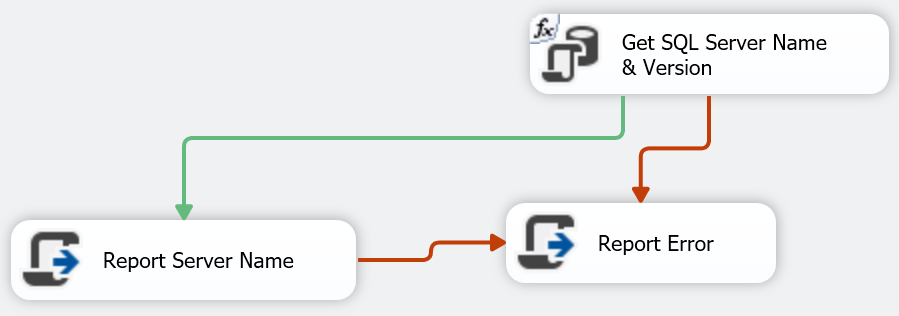
1. Under expressions, build an expression and attach the SqlStatementSource to package parameter, **SQLStmt**.



1. Create a Script Task. Rename it to “Report Server Name”.
2. Connect the “Get SQL Server Name & Version” to “Report Server Name” by dragging the green arrow to task.  
   
3. Create another Script Task. Rename it to “Report Error”.
4. Connect both “Report Server Name” and “Get SQL Server Name & Version” to “Report Error” by dragging the green arrows.

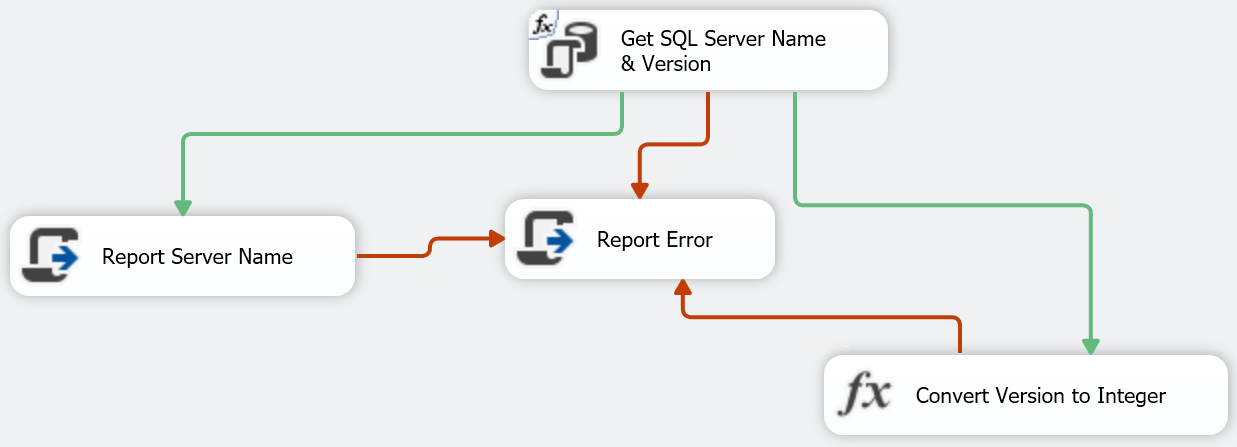
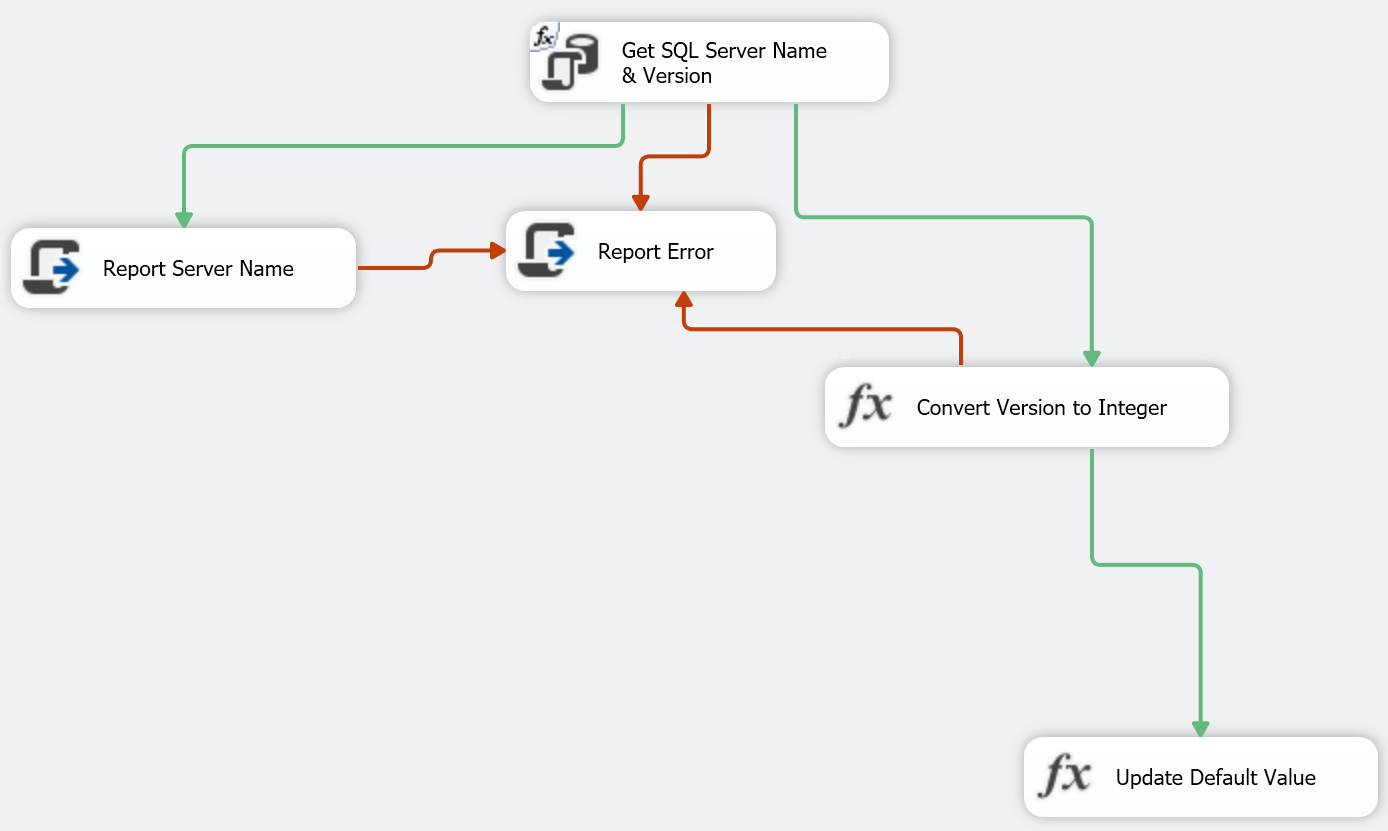


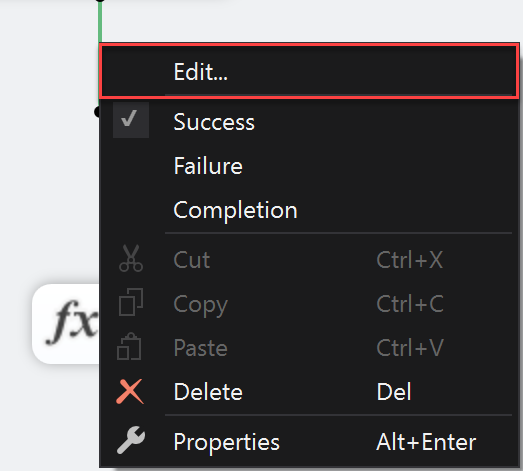
1. However, since we want to report the error on a failure, we will change connections to on failure. Right click on the green line connecting each of the tasks to Report Error and Select Failure. After both lines are updated, it should similar to below.  
     
   



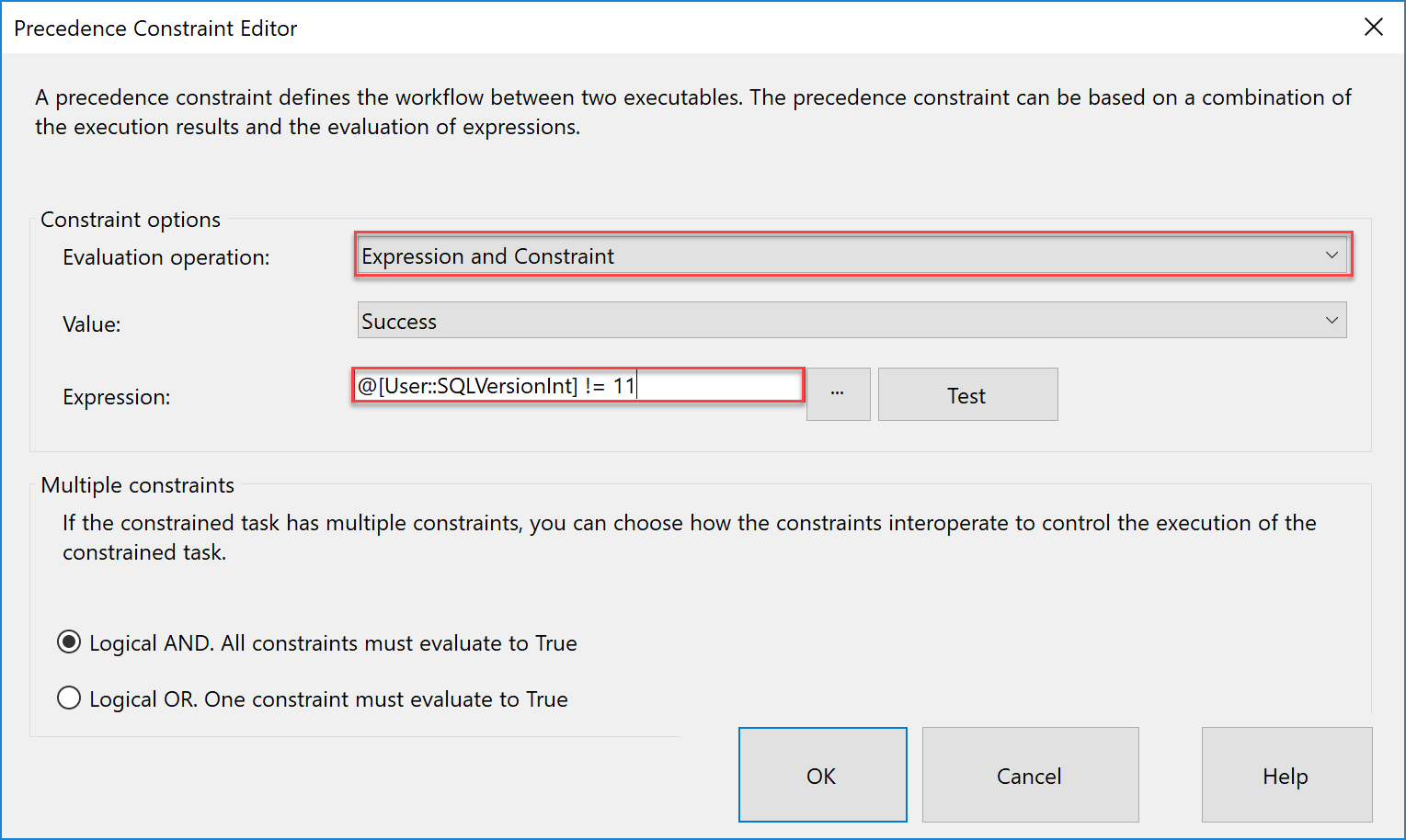
1. Update each Script task with their respect C# Code. Open the script task, assign variable, and type the following script in the Main() function.

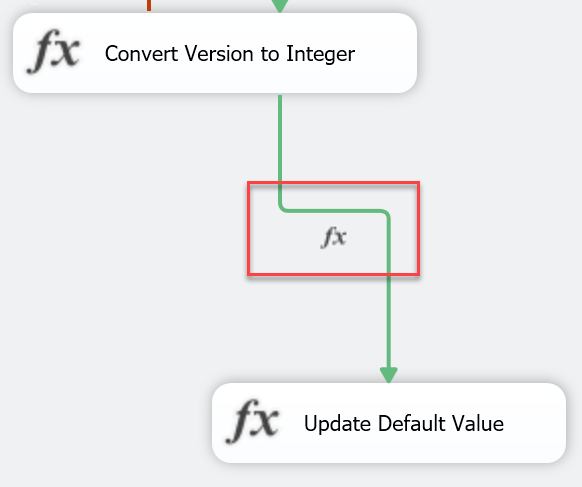
|  |  |  |
| --- | --- | --- |
| Script Task Name | ReadOnlyVariables | Script |
| Report Server Name | User::ServerName | MessageBox.Show("SQL Server Name: " + Dts.Variables["User::ServerName"].Value.ToString(),"Server Name"); |
| Report Error | *Blank-No-Assignment* | MessageBox.Show("Uhmm! I think something went wrong!", "Help!"); |

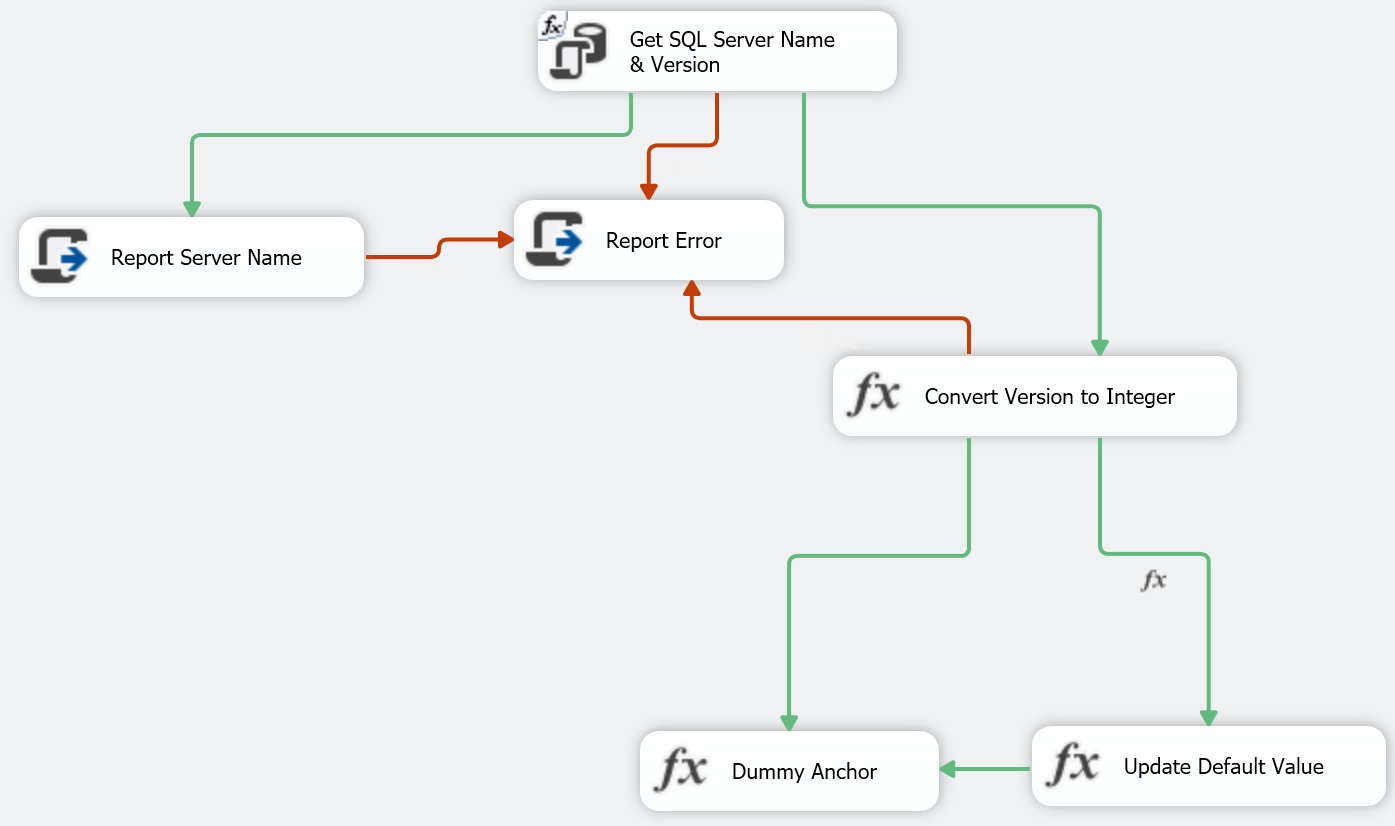
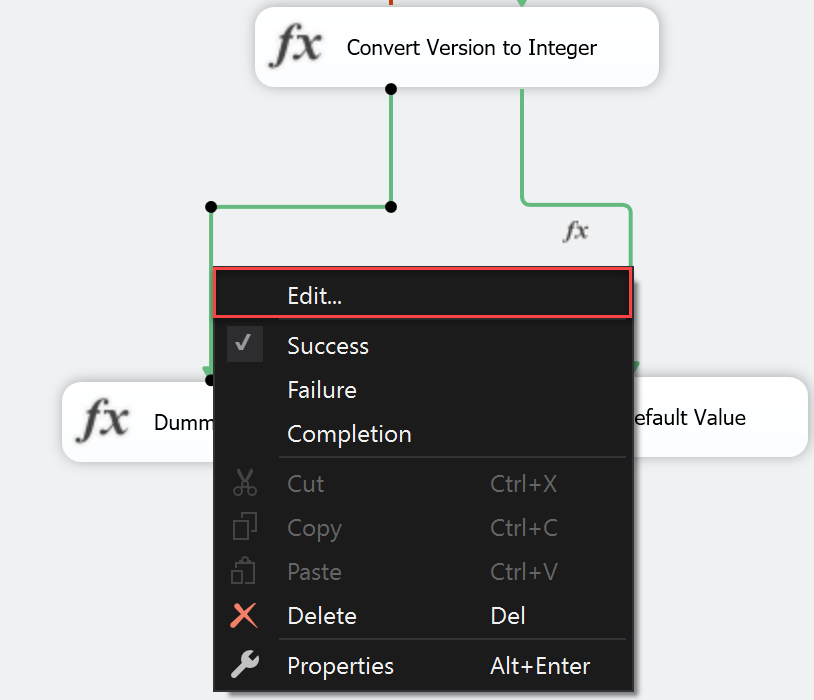
1. Next create an Expression Task, rename it to “Convert Version to Integer” and set the expression to “@[User::SQLVersionInt] = (DT\_I4) @[User::SQLVersion]”.  
   
2. After building the express task, connect, “Get SQL Server Name & Version” to “Convert Version to Integer” with Green arrow (Success) and connect “Convert Version to Integer” to “Report Error” using Red arrow (Failure).  
   
3. Next build another Expression Task, rename it to “Update Default Value”. Set the expression to “@[User::SQLEnglishVersion] = "SQL Server 2017””. Connect the “Update Default Value” to “Get SQL Server Name & Version” using green arrow.  
   
4. We are going to update the expression, for the Update Default Value to make it a bit restrictive. Right click on the green line and select Edit.

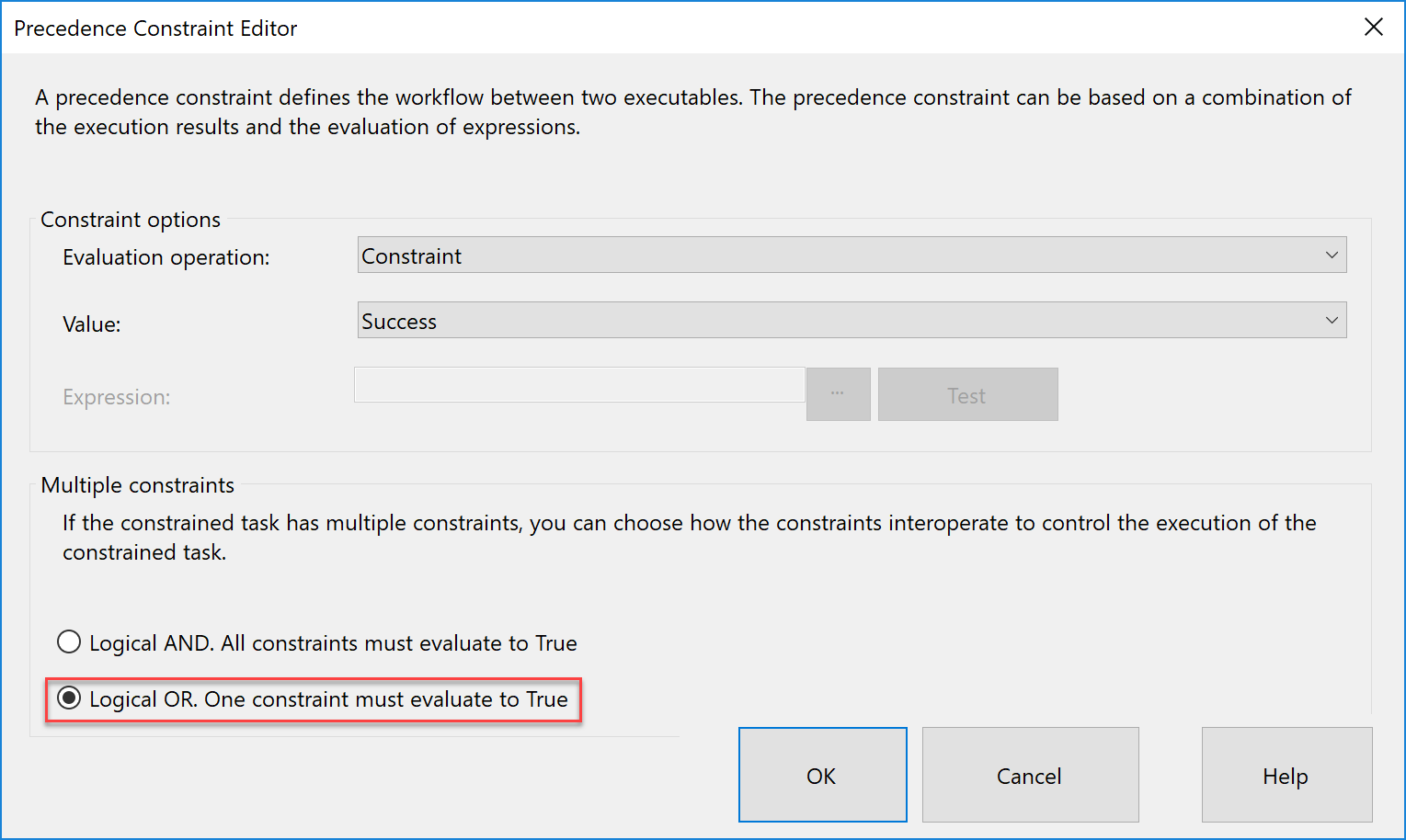


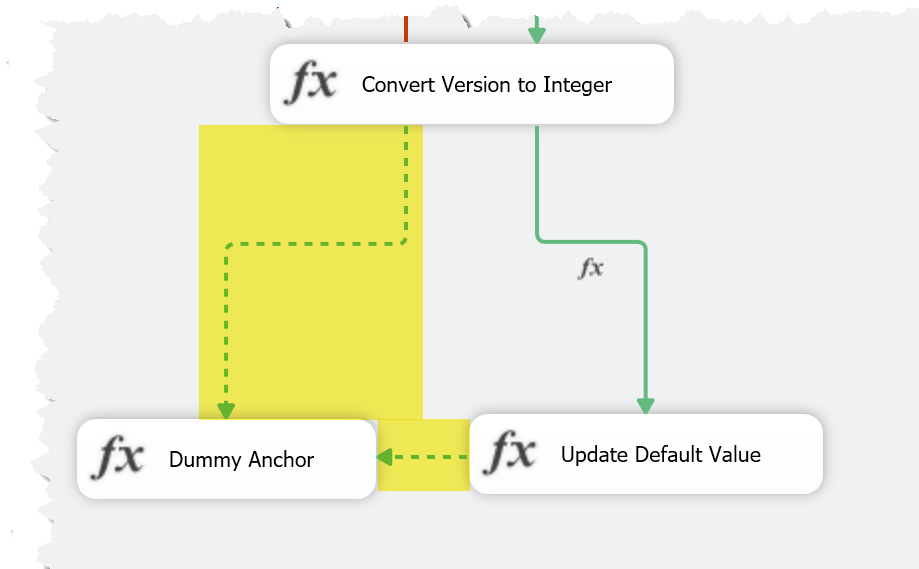
1. Change the Evaluation operation to Expression and Constraint and change the Expression to “@[UserName::SQLVersionInt] != 11”. Note after the precedence constraint is built an “fx” added to the constraint line.

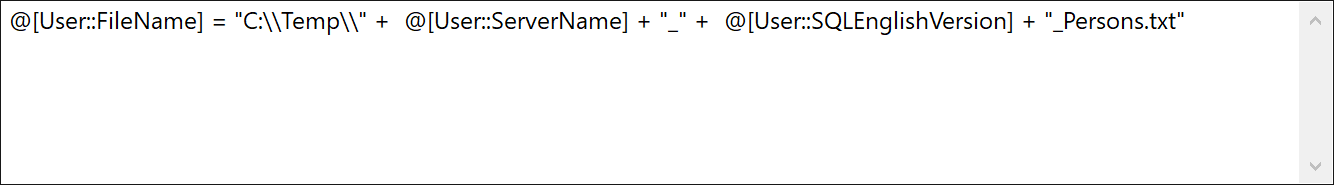
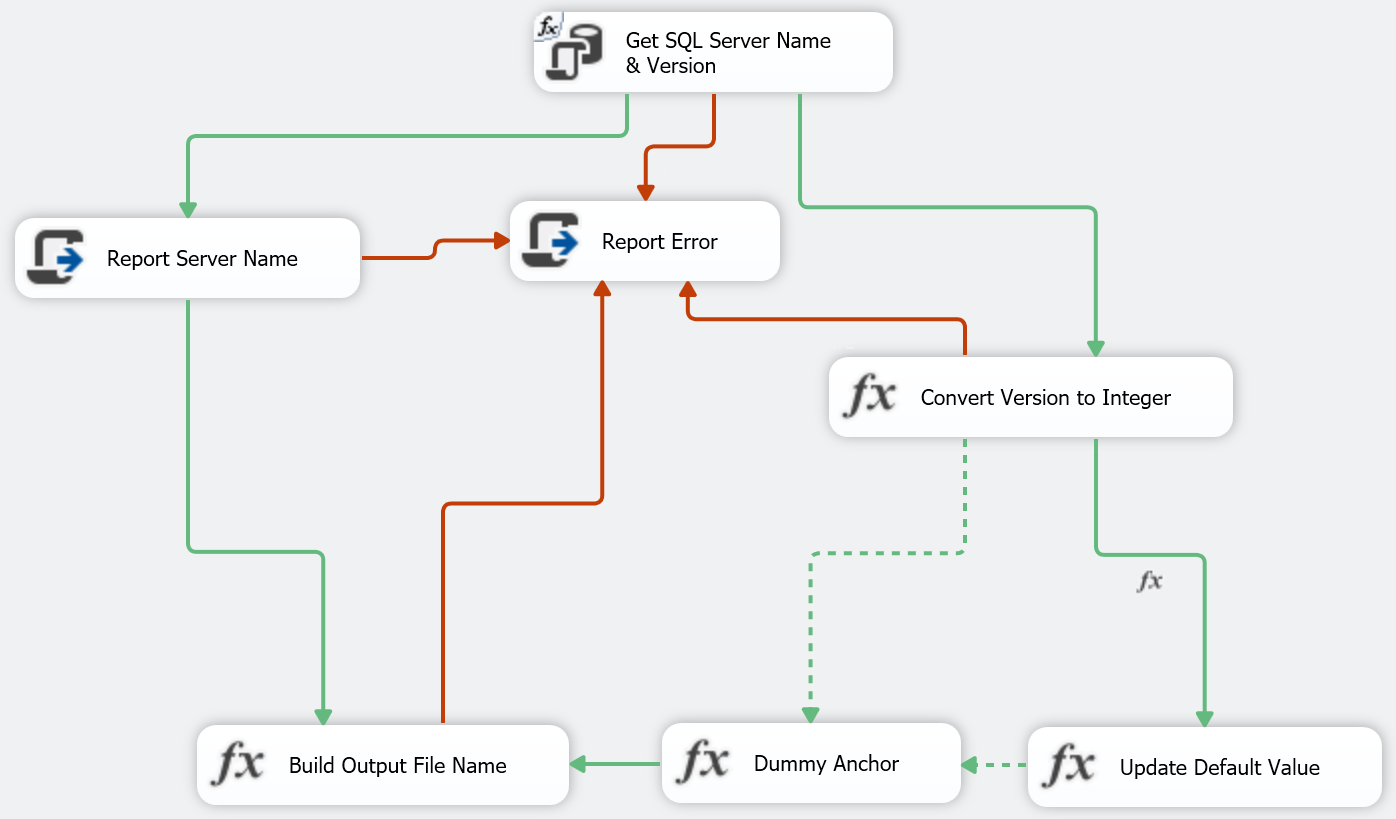
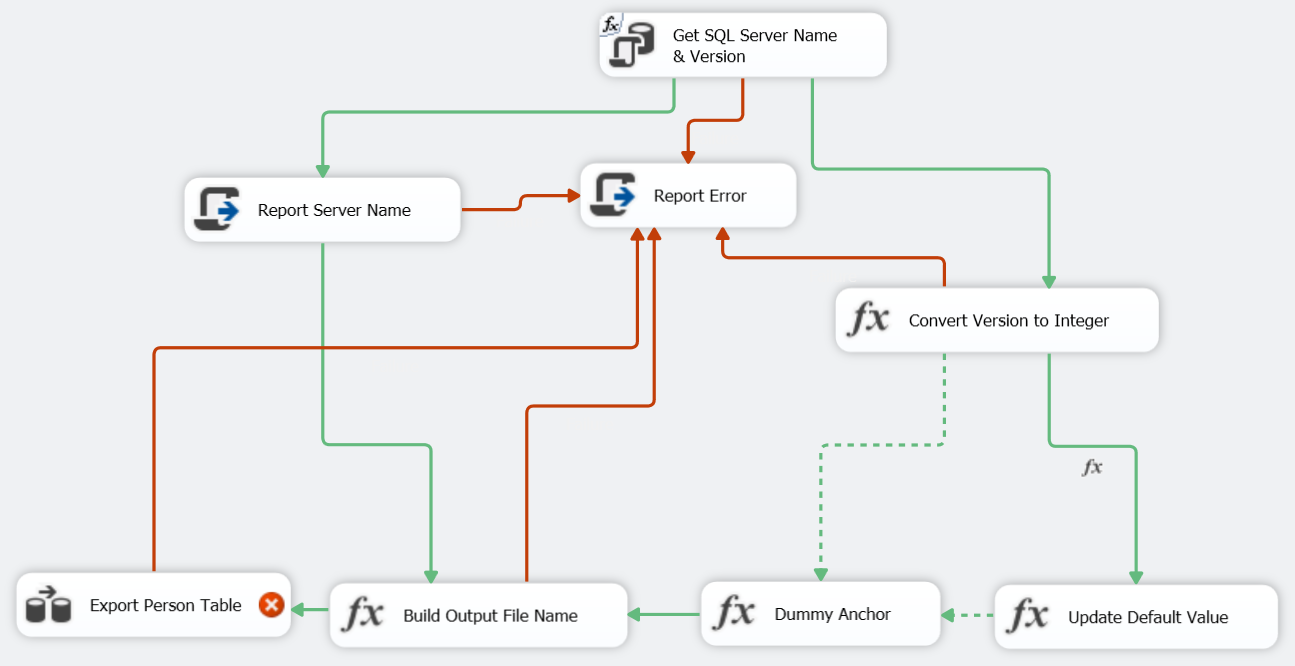
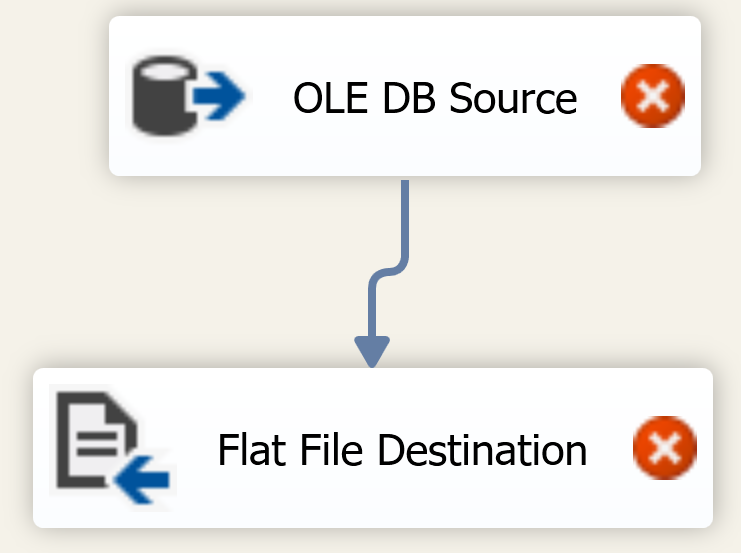
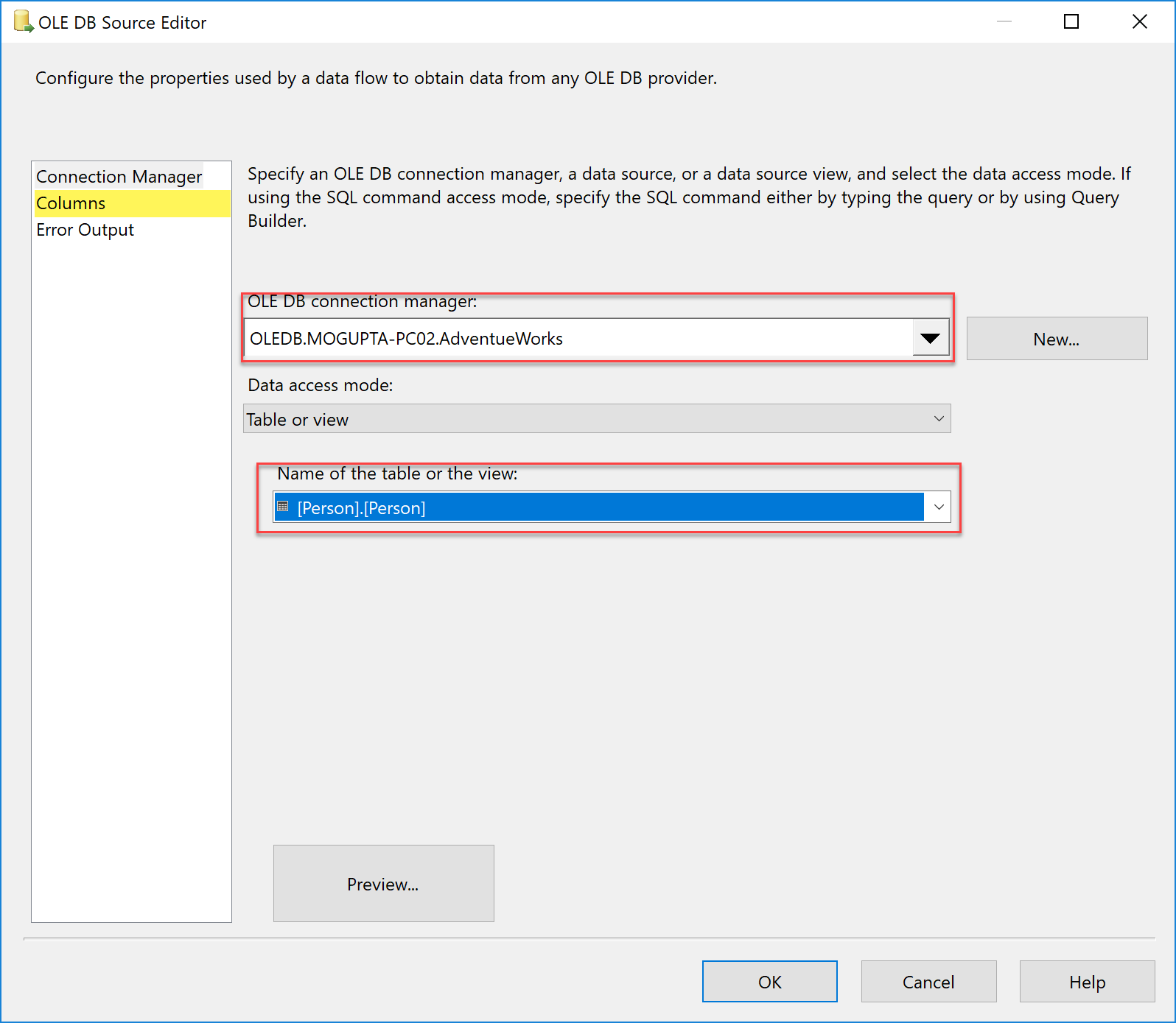
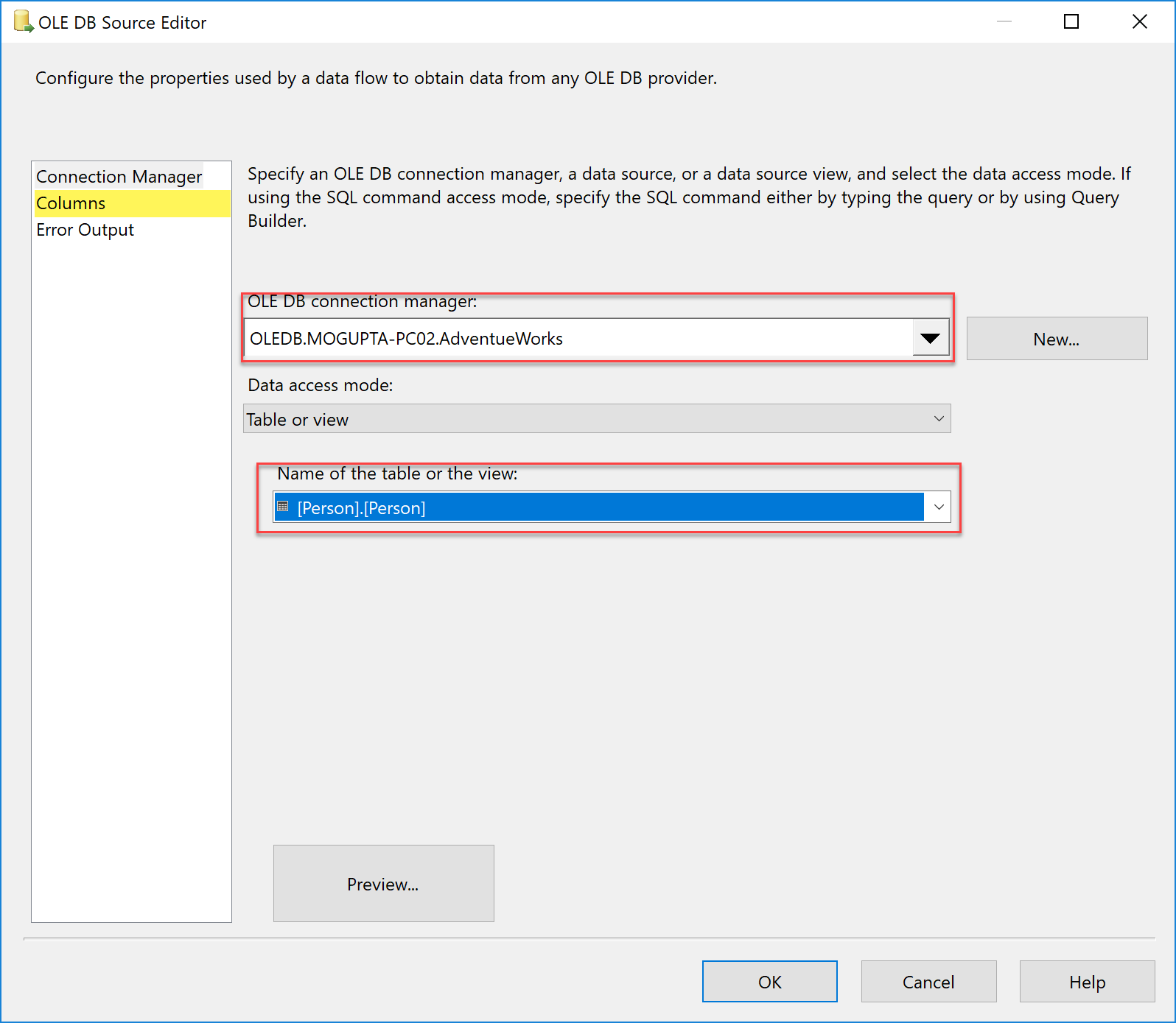


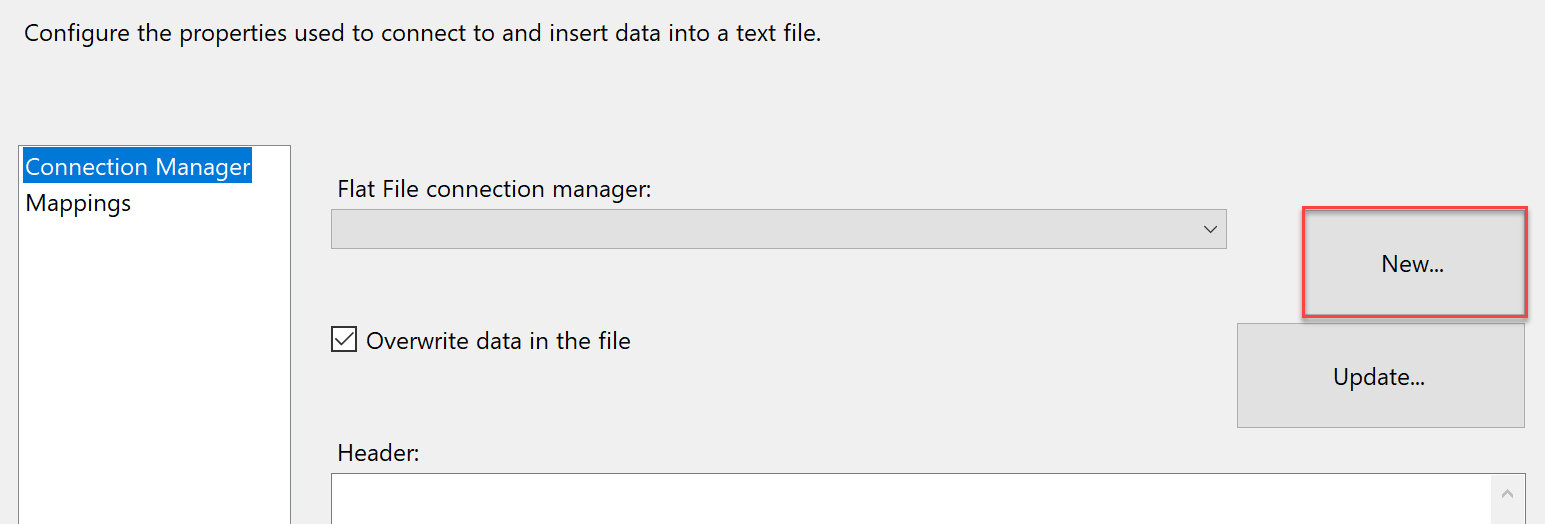


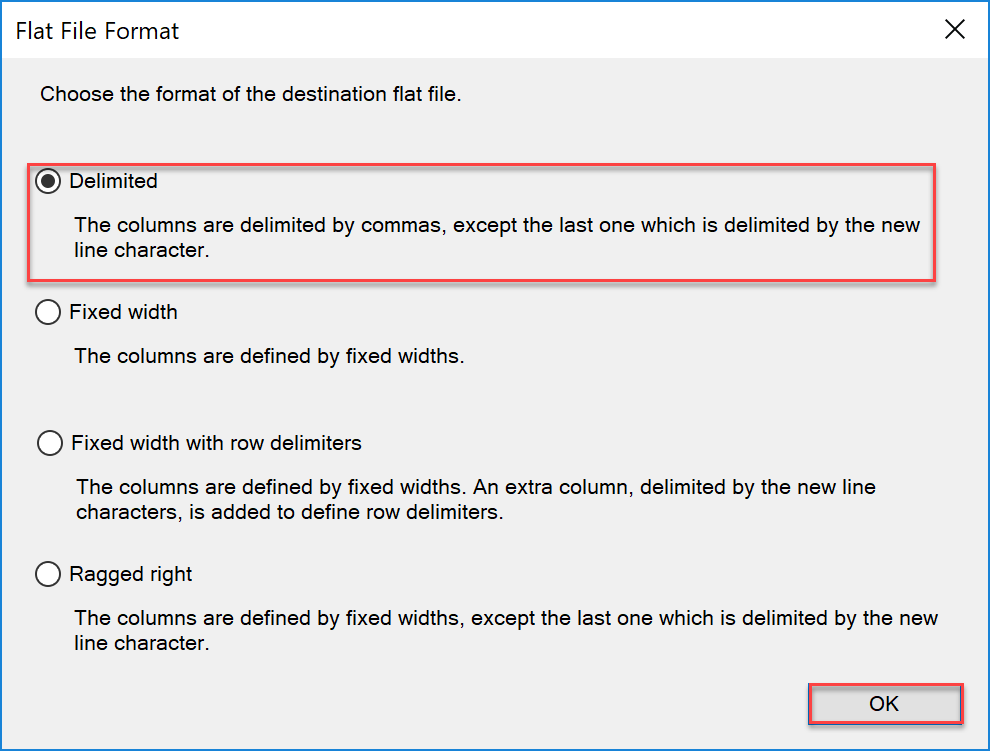
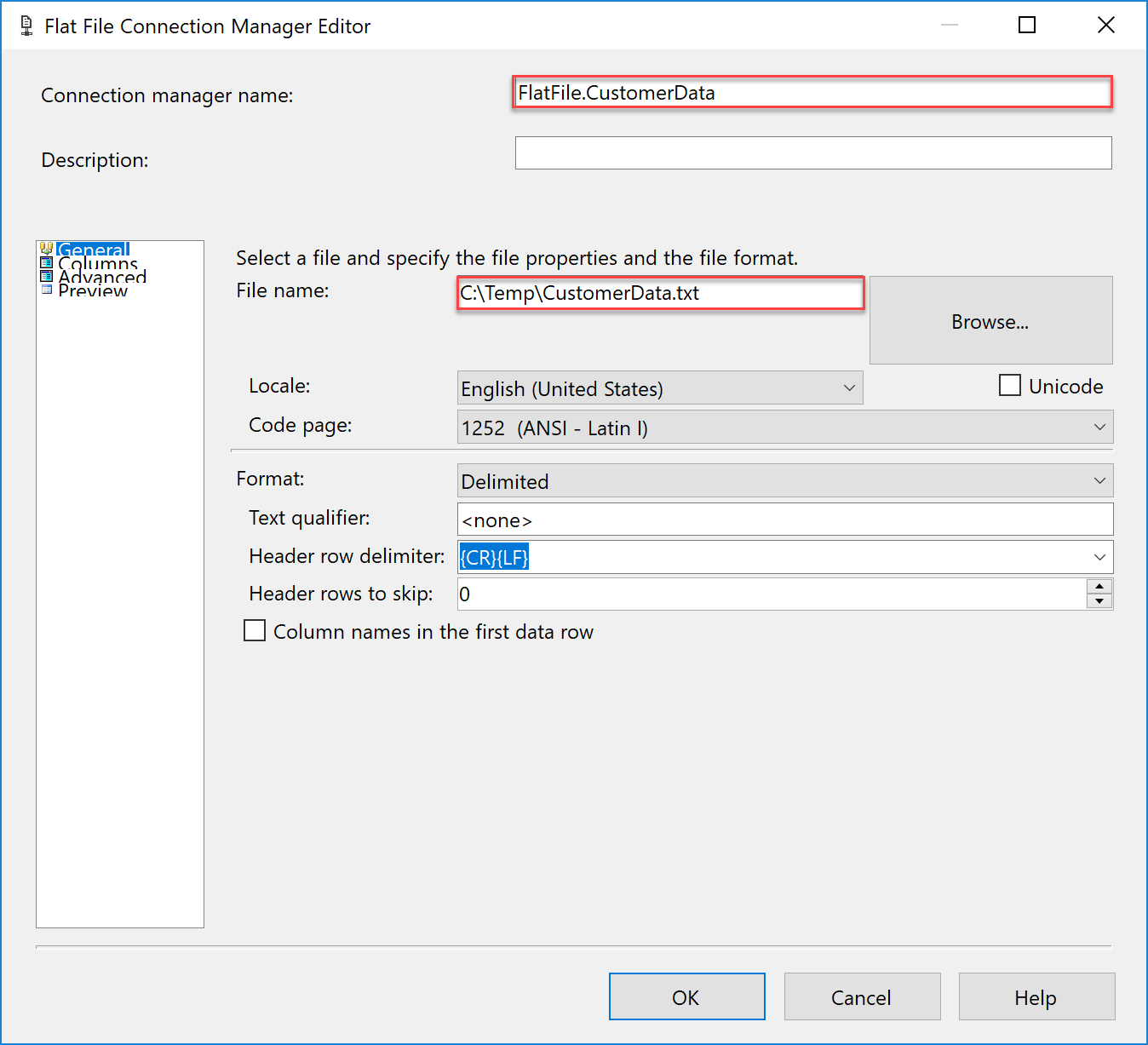
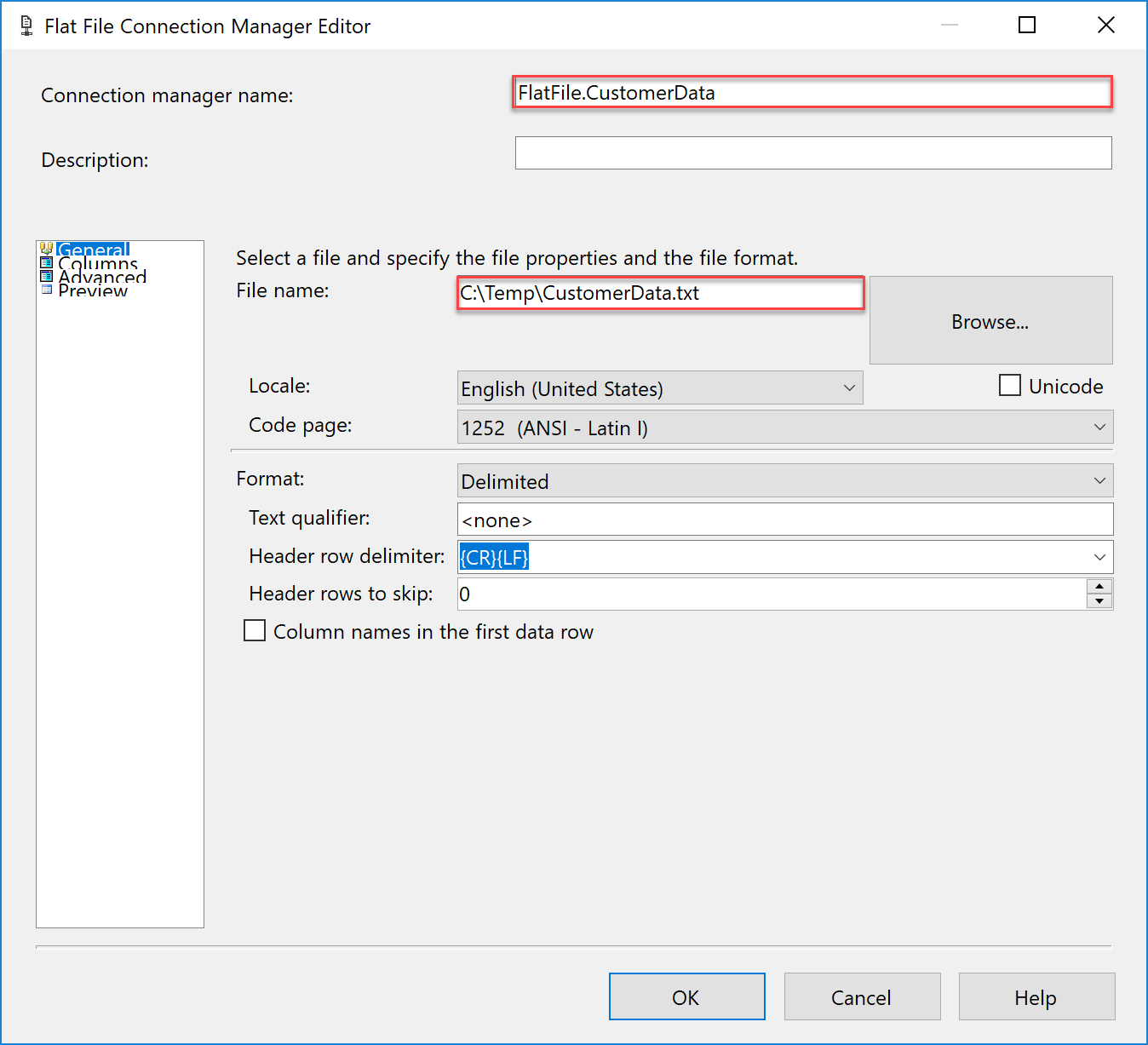
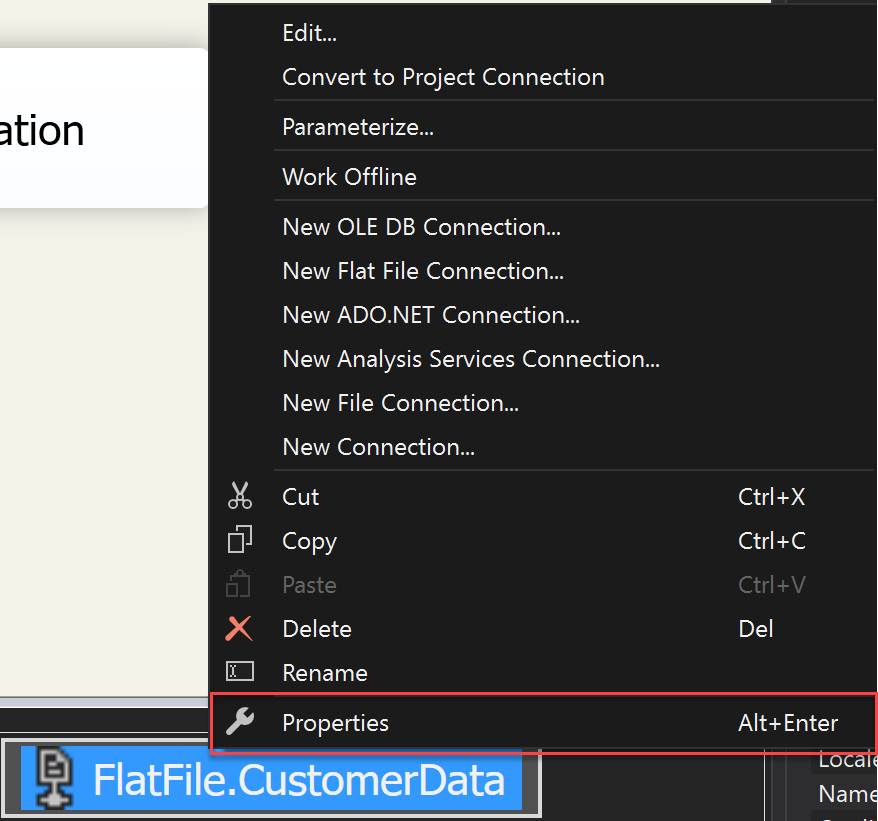
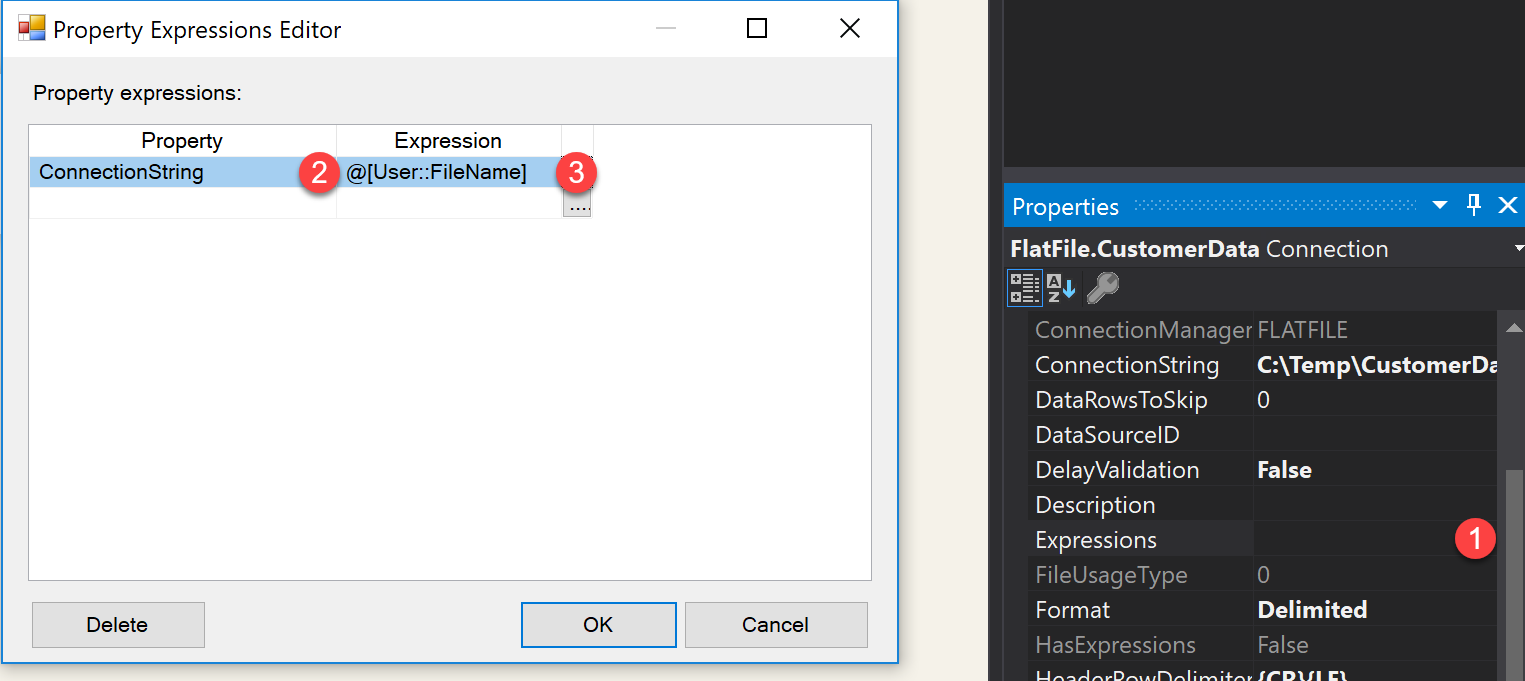
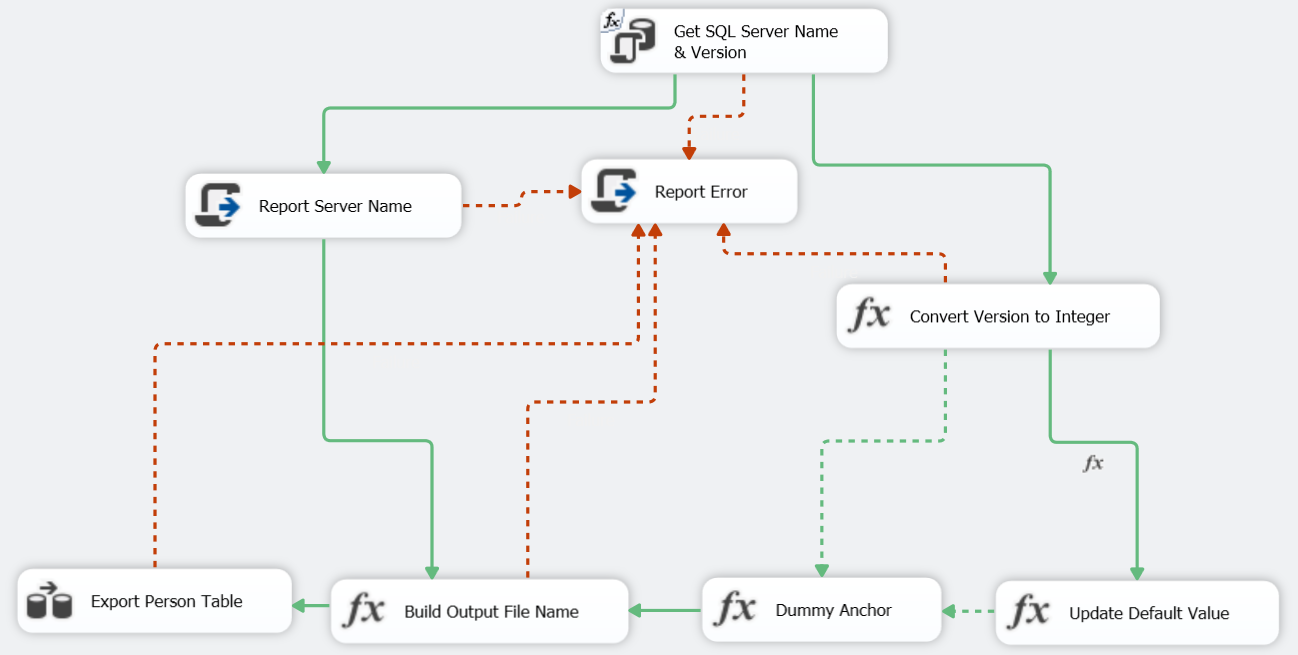
1. Create another Expression Task, rename it to “Dummy Anchor”. Set the expression to “@[User::SQLEnglishVersion] = @[User::SQLEnglishVersion]”. Connect the “Convert Version to Integer” and “Update Default Value” to “Dummy Anchor” task.  
   
2. We cannot have both tasks from “Convert” and “Update” going to Dummy Anchor because of function constraints. Therefore, we need to change the Multiple constraints logical operator from ADD Condition to OR condition. Right click on either of the green-lines connecting to the Dummy Anchor and select edit. In precedence constraint editor, select “Logical OR”. Notice the lines greens line changed from solid to dotted.  
   





1. Create another Expression Task. Rename it “Build Output File Name”. Set the expression to “@[User::FileName] = "C:\\Temp\\" + @[User::ServerName] + "\_" + @[User::SQLEnglishVersion] + "\_Persons.txt"” (Please update the starting path C:\Temp\ to your respective directory. Also note to escape each \ with \\).  
   
2. Connect the “Build Output File Name” task to “Report Error” with Red line. Connect “Dummy Anchor” to “Build Output File Name” with Green line. Connect “Report Server Name” to “Build Output File Name” using Green line.  
   
3. Next add a data flow task and rename it to “Export Person Table”. Connect “Build Output File Name” to “Export Person Table” using Green Line. Connect “Export Person Table” to “Report Error” using Red Line.  
   
4. Double-click on the “Export Person Table” task to configure the data flow.
5. Create two tasks “OLE DB Source” and “Flat File Destination” and connect them.  
   
6. Configure the “OLE DB Source” to connect to our connection string and pull data from Person.Person. After setting those select Columns.  
   
7. Only select columns shown below, we cannot export all columns to CSV because they are not supported types.  
   
8. Next configure the “Flat File Destination”. Create a New Flat File Connection Manager.



1. In Flat File Format, Select Delimited and click OK.  
   
2. Rename the connection manager and set file name, doesn’t have to be valid as we will be changing it via expressions.  
   
3. Before you close the Flat File Destination Editor, click on Mappings to confirm column mapping and click OK.  
   
4. Right click on the FlatFile.CustomerData connection and select properties. This will open properties dialog in bottom right if not already open..  
   
5. Update the expression information for connection string by connecting it to FileName variable.  
   
6. Change the multiple constraint logical condition to OR for Red Lines. So, it looks like below.  
   
7. Execute the package, did it execute successfully? If not why?
8. The parameter value is incorrect, update it to “SELECT @@ServerName AS ServerName, SERVERPROPERTY('ProductMajorVersion') AS SQLVersion”.
9. Execute package again to see the execution path.