Understanding Synchronous and Asynchronous Transformations

**SQL Server 2016**

[Other Versions](javascript:;)

Applies To: SQL Server 2016 Preview

To understand the difference between a synchronous and an asynchronous transformation in Integration Services, it is easiest to start with an understanding of a synchronous transformation. If a synchronous transformation does not meet your needs, your design might require an asynchronous transformation.

[Synchronous Transformations](javascript:void(0))

A synchronous transformation processes incoming rows and passes them on in the data flow one row at a time. Output is synchronous with input, meaning that it occurs at the same time. Therefore, to process a given row, the transformation does not need information about other rows in the data set. In the actual implementation, rows are grouped into buffers as they pass from one component to the next, but these buffers are transparent to the user, and you can assume that each row is processed separately.

An example of a synchronous transformation is the Data Conversion transformation. For each incoming row, it converts the value in the specified column and sends the row on its way. Each discrete conversion operation is independent of all the other rows in the data set.

In Integration Services scripting and programming, you specify a synchronous transformation by looking up the ID of a component's input and assigning it to the **SynchronousInputID** property of the component's outputs. This tells the data flow engine to process each row from the input and send each row automatically to the specified outputs. If you want every row to go to every output, you do not have to write any additional code to output the data. If you use the **ExclusionGroup** property to specify that rows should only go to one or another of a group of outputs, as in the Conditional Split transformation, you must call the **DirectRow** method to select the appropriate destination for each row. When you have an error output, you must call **DirectErrorRow** to send rows with problems to the error output instead of the default output.

[Asynchronous Transformations](javascript:void(0))

You might decide that your design requires an asynchronous transformation when it is not possible to process each row independently of all other rows. In other words, you cannot pass each row along in the data flow as it is processed, but instead must output data asynchronously, or at a different time, than the input. For example, the following scenarios require an asynchronous transformation:

* The component has to acquire multiple buffers of data before it can perform its processing. An example is the Sort transformation, where the component has to process the complete set of rows in a single operation.
* The component has to combine rows from multiple inputs. An example is the Merge transformation, where the component has to examine multiple rows from each input and then merge them in sorted order.
* There is no one-to-one correspondence between input rows and output rows. An example is the Aggregate transformation, where the component has to add a row to the output to hold the computed aggregate values.

In Integration Services scripting and programming, you specify an asynchronous transformation by assigning a value of 0 to the **SynchronousInputID**property of the component's outputs. . This tells the data flow engine not to send each row automatically to the outputs. Then you must write code to send each row explicitly to the appropriate output by adding it to the new output buffer that is created for the output of an asynchronous transformation.