SQL Server 2005 - SQL Server Integration Services - Part 2

In the first article of this series, we presented the basic concepts relevant to understanding SQL Server 2005 Integration Services, which is the primary mechanism for Extraction, Transformation, and Loading (ETL) functionality available in SQL Server 2005. We also provided a quick overview of tools that allow you to manage such activities as DTS package design, development, and storage, as well as interactive and scheduled execution. In order to gain better familiarity with each of these activities, we will look at them from a more practical perspective, using a number of fairly straightforward examples.

We will start with the creation of a package using SQL Server 2005 Business Intelligence Development Studio. To access it, select the appropriate shortcut from the Start->Programs->Microsoft SQL Server 2005 menu. As we mentioned before, from its interface, you can create solutions consisting of one or several associated projects, (available by selecting New Project from option from the File menu). Since we intend to work specifically with Integration Services functionality, make sure to pick the Integration Services Project template listed in the New Project dialog box. Assign a descriptive name, specify the location of the project files, and choose the desired entry from the Solution dropdown list (out of two available options - "Create new Solution" and "Add to Solution"). When creating a multi-project solution, you might want to mark the "Create Directory for Solution" checkbox.

By default, the project window that you are presented with next consists of four main areas, two of which are dockable windows (although that ultimately depends on your preferences, which you can set using items in the Windows menu or via Options available in the Tools menu):

- Package Design area occupies the majority of the workspace and is divided into four tabs:
  - Control Flow (where you create and manipulate your Control Flow elements, such as containers and tasks, available from the Toolbox area located directly to the left of it)
  - Data Flow (providing the ability to manage data flow elements, which, as we explained in our previous article, determine data movement and its transformations)
  - Event Handlers (allowing defining code to be invoked for every event associated with all executable elements within the package - including the package itself)
  - Package Explorer (offering a comprehensive view of package characteristics, including its variables, executables, precedence constraints, event handlers, connections, and log providers).

    Design area is where majority of the work involved in package development takes place. An additional tab labeled Execution Results outlining individual package execution steps along with relevant statistics is displayed following its execution.

- Connections area - contains list of connections included in the current package. It appears whenever you work within the context of Control Flow and Event Handlers tabs. Connections are also listed in the Connections node of Package Explorer.
- Solution Explorer window - gives you graphical representation of the solution, its projects, Data Sources, Data Source Views and DTS Packages (saved as .dtsx files). From context sensitive menus available in this window

you can manage your solution and its projects (for example, by adding new or existing projects, or by designating the one that will be running on startup, in case of multi-project solutions).
- Properties window - is context sensitive, displaying the properties of an object currently selected in an active area of the Business Intelligence Development Studio. Properties can be ordered alphabetically or according to their category.

Equipped with this knowledge, let's try to create a simple package that will load data from an external data store and store it into a table located in one of the SQL Server 2005 databases. More specifically, in our example, we will obtain a list of environment variables set on the SQL Server computer, which can be accomplished by running the SET command (part of the CMD.EXE Windows command shell environment) and redirect its output to a text file. We will then cleanse the data and store it in a database (we will use for this purpose the tempdb database).

Before we start working directly with Business Intelligence Development Studio, we need to create a batch file that will be used to execute the SET command. We will place it in the "C:\DataBaseJournal\SSIS" folder and save it as SETOUT.CMD (obviously the location and naming are arbitrary). It will contain a single command, which generates a listing of all environment variables in the same format - consisting of the name of environment variables and its value, separated by the equal sign ("=") - and then redirects it to a file SETOUT.TXT residing in the same location:

```
SET > C:\DataBaseJournal\SSIS\SETOUT.TXT
```

Next, create a new Solution containing a single Integration Services Project using the Business Intelligence Development Studio (make sure you select the correct Template from the New Project dialog box). Ensure that the Control Flow tab of the Package Designer area is active and drag the Execute Process Task item (listed under the Control Flow Items category) from the Toolbox onto the Designer workspace. Double-click on it (or select Edit from the context-sensitive menu) and in the Execute Process Task Editor fill out the properties of the Process item. In particular, set the Executable entry to C:\DataBaseJournal\SSIS\SETOUT.CMD and WorkingDirectory to C:\DataBaseJournal\SSIS. You can test the package at this point by pressing the Ctrl+F5 key (the same can be accomplished by selecting Start without Debugging entry in the Debug menu). Check the content of the C:\DataBaseJournal\SSIS\SETOUT.TXT and verify that it contains a listing of all environment variables in the format we described before.

Now that we have the output of the SET command line utility stored in the text file, we want to load its content into a target database. To accomplish this goal, we need a connection to this text file (Flat File Connection) and a corresponding data source (Flat File Source), as well as a connection to tempdb database (OLE DB Connection) and a corresponding data destination (OLE DB Destination). We also require a properly structured table in the target database that will store our data. We can create it prior to running the package, but we can also incorporate this action into it - and this is the approach we will take.

All of this functionality will be enclosed into a single Data Flow task. To create it, drag it from the Toolbox while having the Control Flow area active (note that this also can be done directly from the Data Flow area). Extend the precedence constraint originating from the Execute Process Task (represented by the arrow starting at its border) so that it reaches the newly created Data Flow task (you can organize them within your workspace using Align options available from the Format menu). With default settings, the second task will be triggered as soon as the SET command line utility successfully completes and its output is stored in the SETOUT.TXT. Switch to Data Flow area by selecting its corresponding tab in the designer.

Now it is time to take care of a Flat File connection to our text file and its corresponding data source, as well as the OLE DB connection to the tempdb database and OLE DB data destination. To create a connection to our data source, right click on the area of the Connections area and select "New Flat File Connection..." entry from the context sensitive menu. Point to the C:\DataBaseJournal\SSIS\SETOUT.TXT using the Browse... button next to the File name(s) text box and select Columns entry in the listing of options appearing on the left hand side of the Flat File Connection dialog box. Type in the equal sign ("=") in the Column delimiter text box and click on Refresh link immediately below it. In the Preview display area, you should see the listing of up to 32 rows containing environment variable name in Column 0 and its value in Column 1. Switch to the Column Properties option listed on the left hand side and change the names of the columns to VarName and VarValue, respectively (from Column 0 and Column 1). Increase Output Column Width for the VarValue column from the default 50 to something considerably higher (environment variables in Microsoft Windows XP or later can have up to 8191 characters, on Windows 2000 the maximum length is 2047 characters). You might also want to assign the connection a descriptive name, which would make it self-documenting.

Once the Flat File Connection appears in the Connections area (and the Data Flow tab is selected), add the Flat File Source entry from the Toolbox to your Data Flow Task. Double-clicking on it will automatically bring up Flat File Source Editor with the Flat File Connection listed in the Connection text box. View of data (up to first 200 rows) is available via the Preview button. By selecting Column on the left hand side of the same page, you can control the mapping between the External columns (derived from the connection to SETOUT.TXT) and Ouput columns (which we will use to populate a target table in the tempdb database). Click on OK to confirm the default choices.

To create a connection to the tempdb database, right-click again on the Connections area of the designer and select "New OLE DB Connection" entry from the context-sensitive menu. Next, using the New... command button, bring up the Connection Manager interface and fill out relevant entries, pointing to tempdb of your SQL Server installation. Pick Microsoft OLE DB Provider for SQL Server as the OLE DB Provider, specify the SQL Server computer name, assign proper authentication method (Windows NT Integrated Security is the recommended choice), and chose tempdb as the target database. To complete the process, click twice on the OK button.

Next, create OLE DB Destination by dragging it from the Toolbox to the Data Flow area of the designer. Extend to it the data flow from the Flat File Source component. Note that the rectangle representing the OLE DB Destination task contains a red cross icon - indicating that the connection manager has not been fully configured. . To correct it, double click on the OLE DB Destination, which will display its Editor window. From there, ensure that the previously created connection to the tempdb database is selected in the Connection entry and that Data access mode is set to "Table or view." Click on the New button next to the Name of the table or view drop-down list. This will automatically provide you with the complete CREATE TABLE statement. To account for larger sizes of VarValue entries, change the length of VarValue VARCHAR column to match the value you assigned to VarValue column in the Flat File Connection. Leave the remaining options set to their defaults and click on Mappings entry on the left hand side of the

OLE DB Destination Editor. Accept the existing settings (which maps VarName and VarValue input columns to their respective destination columns) and click on OK.

Finally, you can save and execute the package, which should result in the creation of a two-column table in tempdb, containing pairs of environment variable names and their respective values. You might want to consider assigning appropriate names to each package component, which are more descriptive than their defaults (this can be done from the Properties window within designer interface).

In our next article, we will modify our existing package in order to explore more advanced features available in the SQL Server 2005 Integration Services.