Microsoft SQL Server 2012 Data Tools

By Frank Haug

I Suggestions

A Online Resources

Please feel free to use this document, but you should also look at the online help and look at the tutorials. The "MS SQL 2012 Books Online" includes volumes of information that I cannot (and will not) attempt to cover here. The MS SQL Server Data Tools (formerly called the Business Intelligence Development Studio) also has context sensitive help as well as some additional help resources. There are also some samples that come bundled with the Microsoft SQL Server 2012 software (called the tutorials). Lastly, if I mention "BIDS" by mistake, I mean the Data Tools – a simple mistake of using the old name; but there are probably online resources mentioning the "BIDS" that might still be applicable for us here.

B "There Is More Than One Way To Do It" – (TIMTOWTDI)

This is a familiar motto of Perl and Python developers. It is viewed as a good thing. For our purposes here, it applies in several ways. First of all, in your ETL packages, there are certainly a variety of combinations and constituents that you can choose from while you are creating your package. Secondly, in the GUI for the tool, there is often more than one way to achieve the same effect. For example, sometimes the double click will perform the same action as a menu item that you display via a right-click.

While this can certainly be viewed as a "good thing", it can also be frustrating for new users and for documentation authors. I will only describe a single approach here – but feel free to use other GUI tricks or other approaches in your milestone.

C Techniques Used In Your Solution

As far as the Milestone 3 ETL is concerned, you can use ALMOST ANY approach as long as it obtains the correct answer – the only restrictions I have (aside from working only with **your team**) are these:

1. You cannot use your windows login (Windows Authenticated) to connect to the OLTP or Star Schema databases for ETL—so none of the connections inside your M3 Deliverable will use Windows Authentication.

2. You cannot grant additional privileges (access rights) to the four SQL Authenticated accounts. In other words, each database has one and only one SQL Authenticated account that can connect to it and access or modify it – this account is the only account you can use to connect to that database for ETL.

3. All SQL must be performed inside the package (in one or more task) – no external steps or external files required to run your solution against a brand-new "MakeAll.cmd" version of the databases.

In other words, you don't have to use the most efficient, most elegant, or "most-anything" approach as long as it gets the correct answer and does not do anything "illegal" to do it. Remember ALL group-submission project milestones are part of a team project, so you can work with your teammates, but this does not extend to people outside your team! Including work done by others (non-team members) is an academic integrity violation, doing so will result in a minimum of a team-wide zero grades for the milestone, or possibly for the project or course-grade as-a-whole. Otherwise, feel free to create tables, views, stored procedures, etc. in any of the OLTP databases and / or inside the star schema database—provided that you do not violate these rules.

D Overview and Caveats

What follows in an attempt to capture the basic steps needed to perform simple ETL. It does not claim to be the best design, implementation, or approach – merely an illustration. I do not offer a complete definition of anything or a complete list of all the components and nuances – I only supply the minimum necessary for us to work with the tool. Look at the online documentation for further details, or ask questions on the per-team forums under your group page on the blackboard.

Different "Task Types" have different advantages and different limitations. They also have slightly different connections, underlying technology, and interactions. For example, there are three different "flavors" of "Source Connections" and three different "flavors" of "Destination Connections" to choose from. You must not mix and match source and destinations from different types directly – but can mix and match tasks that are using different types internally.

In the section II, we will look at an overview of how the MS SQL Server Data Tools work, and how we will use them in milestone 3—but we are not actually going to create the solution / packages yet. We will start the hands-on portion in section III.

II Data Tools Basics

A The SSIS Elements

A Package consists of one or more Control Flow Element. Control Flow Elements can be categorized as Tasks, Containers, or Precedence Constraints. Tasks and Containers can be connected to each other via precedence constraints ("**On Success**", "**On Failure**", "**On Completion**", etc.) A given Task or Container Element will only be executed after all of its precedence constraints are satisfied. This means that any Task or Container Element without precedence constraints can be executed at any time and in parallel with any other Control Flow Elements that are ready to run. The Container Elements can contain any Control Flow Element and are used to perform Looping, Grouping and other Control Flow activities.

Control Flow Elements are used to:

- Prepare or copy data
- Interact with other processes
- Implement repeating workflow

The Data Flow Task is a special kind of Control Element (a special kind of Task). Editing a Data Flow Task opens a separate tab and reveals a sub diagram that can contain one or more Data Flow Elements. Each Data Flow Element is either a Data Source, a Data Destination, a Data Transformation, or a Data Path. The Data Source Element and Data Destination Element connect to external data sources like an RDBMS (and other types too). Obviously, the Data Source Element is used for extraction while the Data Destination Element is used for Loading or Staging. Data Transformation Elements can be used to implement complex changes in data structure or content after it has been extracted from a Data Source Element and before it is Loaded or Staged by a Data Destination Element. Strictly speaking, we always need a Data Destination Element, almost always need a Data Source Element, but only occasionally need one or more Data Transformation Elements within a typical Data Flow Task.

Data Flow Elements are connected via Data Paths. These look similar to the Precedence Constraints that connect Control Flow Elements in the main package but have entirely different semantics. A Data Path is merely a way to direct the output of one Data Flow Element into the input of another Data Flow Element. Often we can simply connect a Data Source Element's output to a Data Destination's Input. The Data Destination will attempt to define a default mapping based upon column names.

B Database Technology "Flavor" Options

There are three basic flavors of RDBMS technology used to implement Data Source Elements and Data Target Elements: ADO Dot Net, OLE DB, and ODBC. Strictly speaking, the ODBC can also be reused inside of the other two technology flavors. When a Data Source and a Data Destination are used together, they should be based upon a compatible technology. In other words, you should not mix the technologies inside a Data Task most of the time.

1 ADO Dot Net Flavored components

The ADO Dot Net flavored components are implemented using ADO Dot Net Providers for their database connectivity and using ADO Dot Net functions and facilities (threading, garbage collection, connection pooling, etc.) to implement the desired database activities. An ADO Dot Net Provider is usually a wrapper that surrounds an OLE DB provider or an ODBC provider. ADO Dot Net facilities can be a great advantage for more complex transformations (such as non relational sources or destinations, or systems using distributed transactions). Having said that, when ADO Net components are used for simple transformations that contain a large amount of data, there can be significant overhead involved with no additional functionality or features provided. ADO Dot Net components are typically written in C# or VB Dot Net.

ADO Dot Net Sources can be very useful when their output is used by other control flow elements. For example, ADO Dot Net components can return RecordSets and Schema RowSets that the ForEach Control Flow Element can iterate through. In other words, for example, this is how we can request a list of User Defined Tables in a database and then loop through them in order to perform some task for each table (such as deleting all of its rows).

2 OLE DB Flavored components

The OLE DB flavored components use OLE DB Providers (which in turn can be native providers or wrappers that surround an ODBC provider). These are typically the components that we want to use for RDBMS connectivity. In particular, the OLE DB Native provider for MS SQL 2012 is the best connection manager to use for our project in most (if not all) circumstances when it is possible. These are based upon OLE DB, which is to say they rely on COM and DCOM for their implementation. Each component is responsible for its own threading, garbage collection, etc. The provider we use for this is called "The Native Provider" for MS SQL 2012, and it is named "Native OLE DB\SQL Native Client", or simply "SQLNCLI". There is also an older provider called "The OLE DB Provider for SQL Server", this is used for MS SQL 2000, but lacks some of the MS SQL 2012 specific features.

3 ODBC Flavored components

ODBC flavored components are old technology. Sometimes they are necessary due to a lack of available providers. Other times they are necessary to allow for some feature or to avoid some software defect in a different provider. For MS SQL Server 2012, the ODBC provider should only be used inside the ADO Dot Net Provider. **DO NOT USE THIS PROVIDER DIRECTLY**.

C Connection Managers and Connections

A Connection Manager in SSIS is really nothing more than a facility used to build connection strings. You create a new instance of a connection manager and specify all the connectivity details such as the provider type, server name, user name, password, etc. Other components can then use the connection manager to create connections to the specified Data Source or Data Destination as needed.

Connections are **NOT** the same thing as Connection Managers:

- A Connection Manager exists inside of a Data Flow Task and is used by some Data Flow Elements (Data Sources and Data Destinations).
- Other Task types (such as the Execute SQL task or the Execute T-SQL task) also use connections but do **NOT** use Connection Managers.
- A Connection Manager is for a particular flavor (i.e., you need a different connection manager to use a different flavor connection to the same database!)

D Password Protection for the Package

THIS IS VERY IMPORTANT!!!!!

When you first create the ETL Package, open the properties for the solution—to do this, right-click on the name of the solution under the solution explorer. I named my solution Team_00_M3_p1_v01—this usually appears on the right-hand side of the main dialog, but you can move or hide it. If it is not visible, use the view menu to show the solution explorer. When you **right click on the solution**, open the properties as shown in Figure_01.

THIS IS VERY IMPORTANT!!!!!

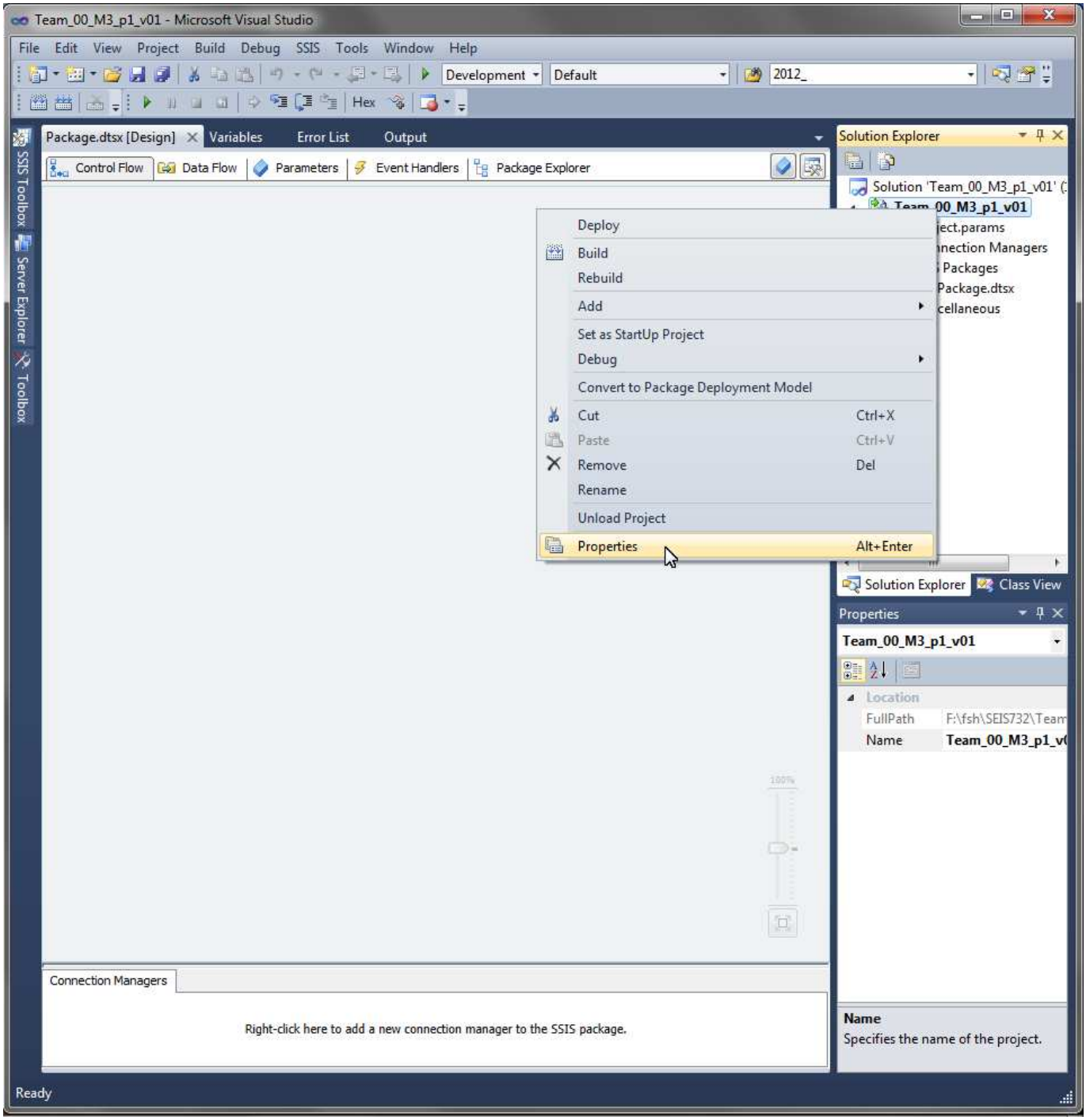

Figure 01 – Viewing / Setting the Solution Properties

Next find the **ProtectionLevel** property under the security category, as shown in Figure_02.

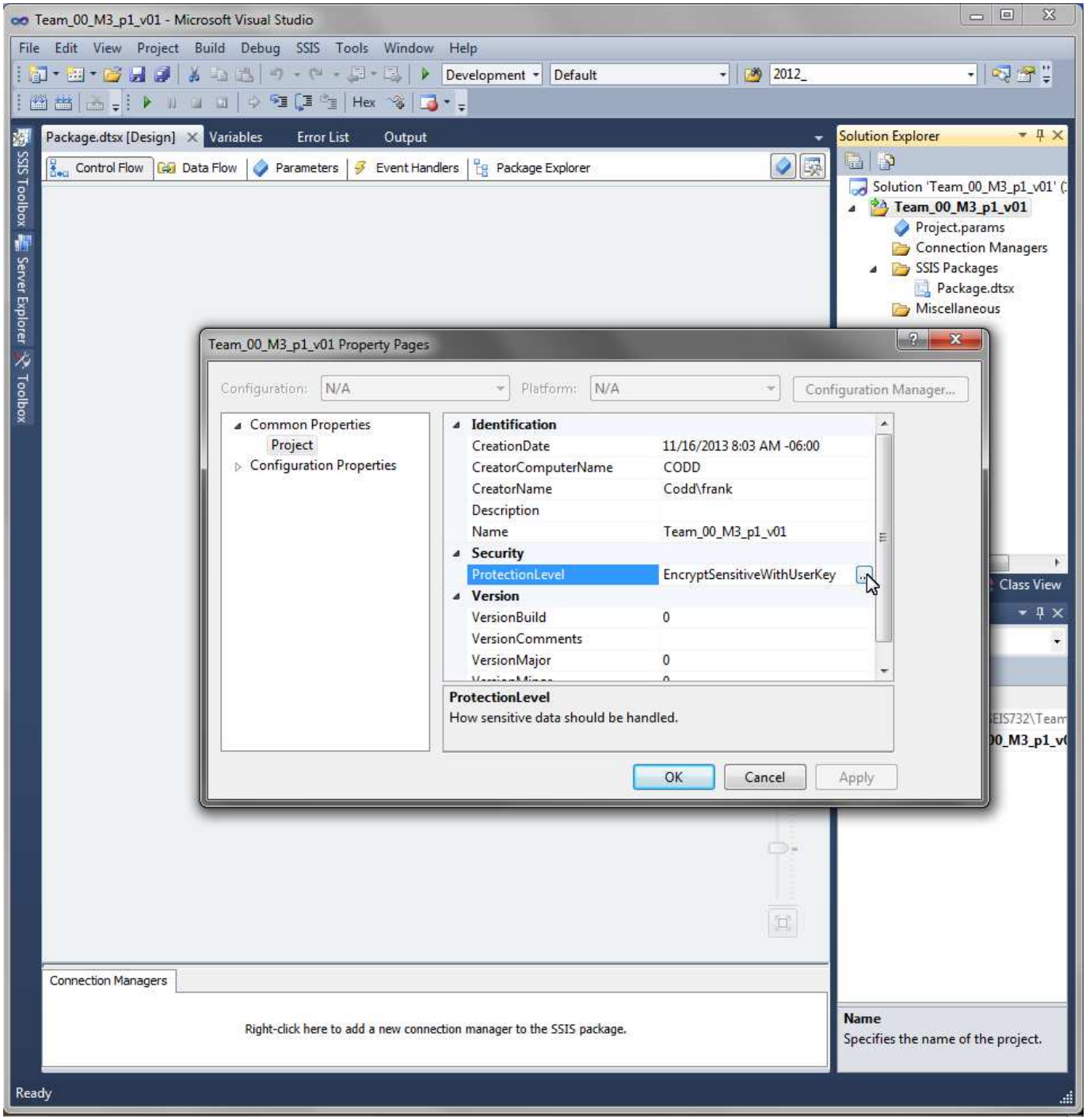

Figure 02 – Viewing / Setting the Solution Encryption Properties

Change the ProtectionLevel from **EncryptSensitiveWithUserKey** to **Encrypt sensitive data with password**, as shown in Figure 03.

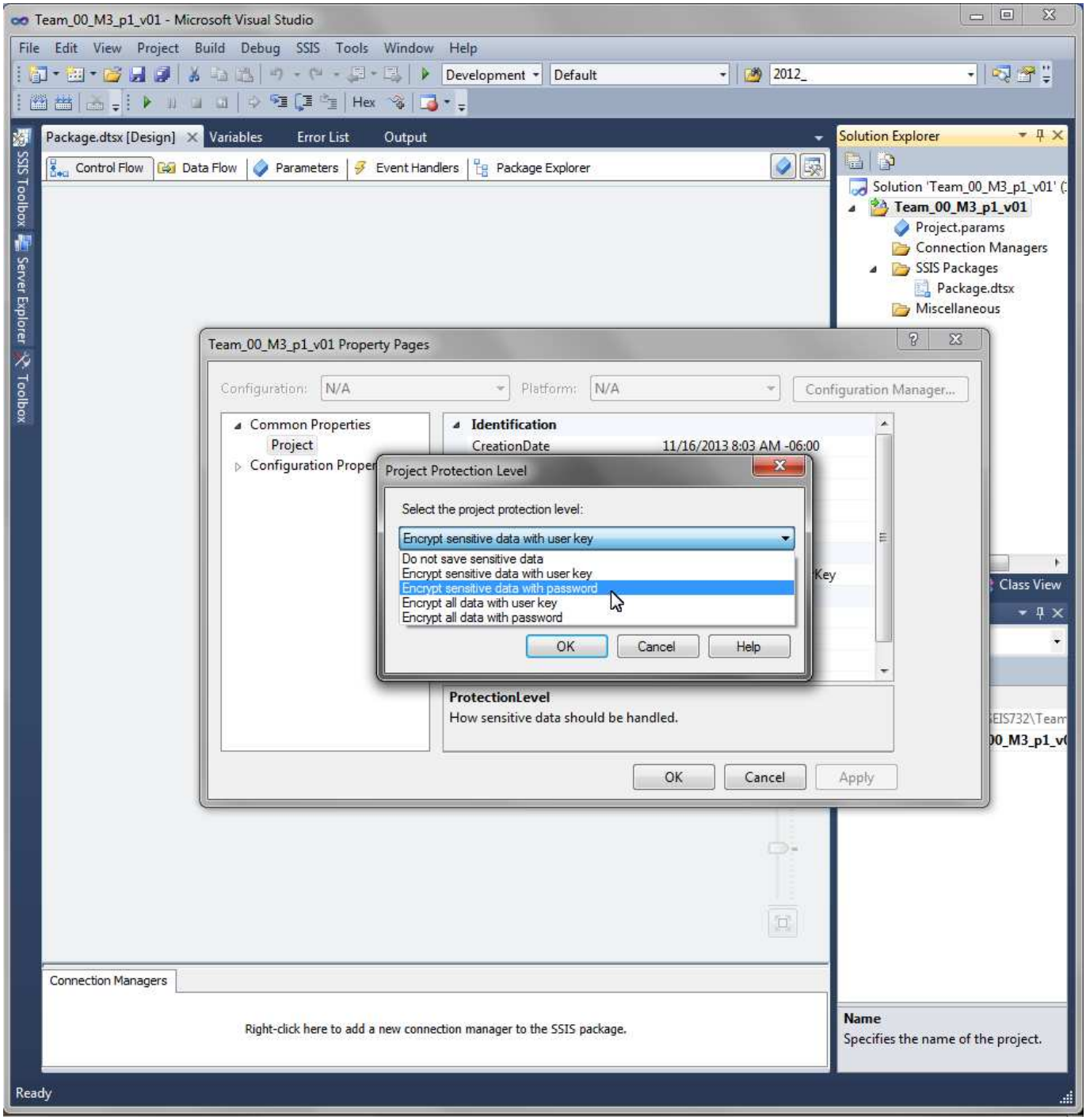

Figure 03 – Changing the Solution Encryption Properties

This will prompt you for a password, which **you need to enter twice**. Set this to the **same password as your data bundle password**, as shown in Figure 04.

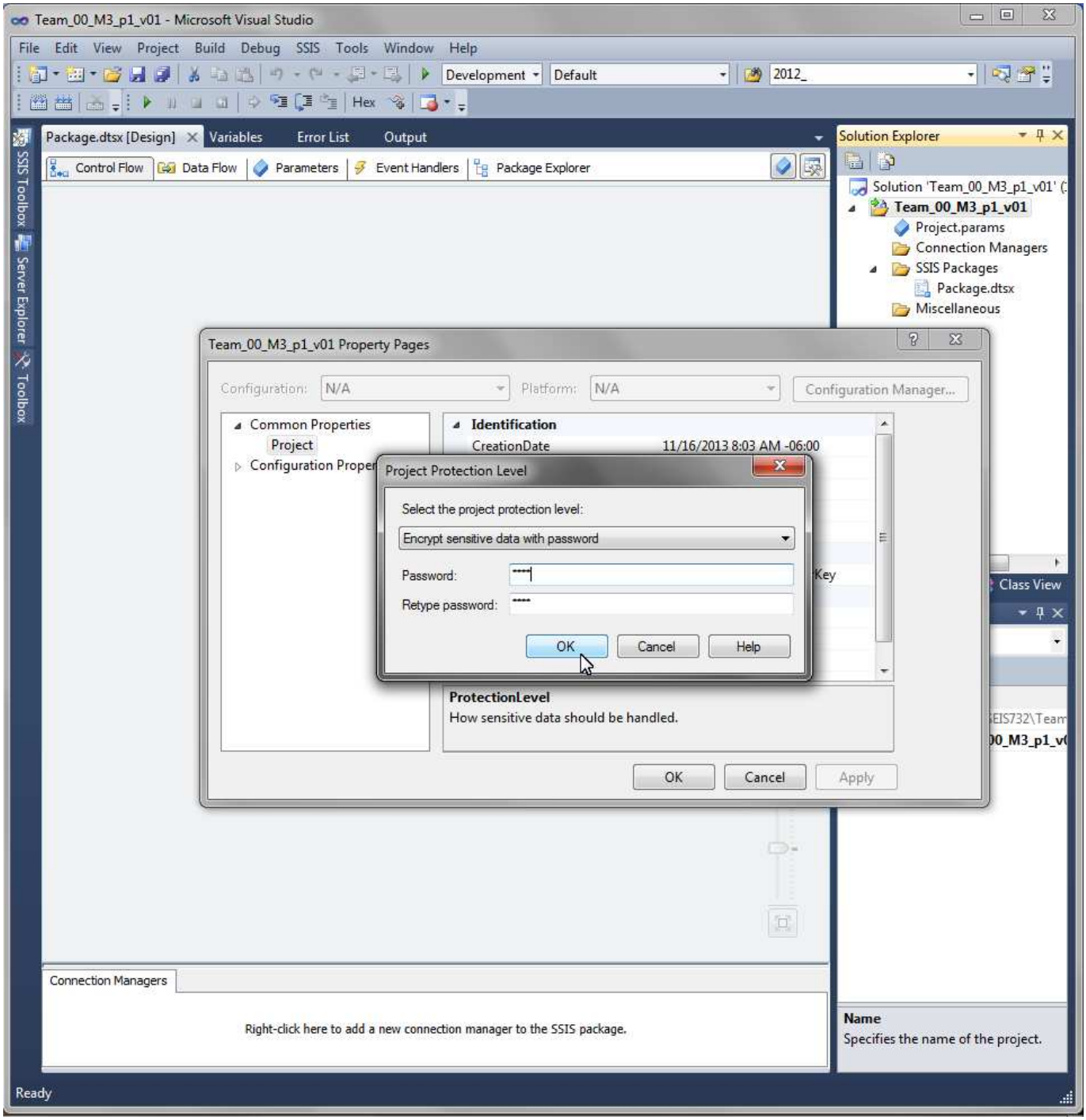

Figure 04 – Setting the Solution Password

This will cause the dialog shown in Figure 05 to appear. Click OK, because we will address it as the next step.

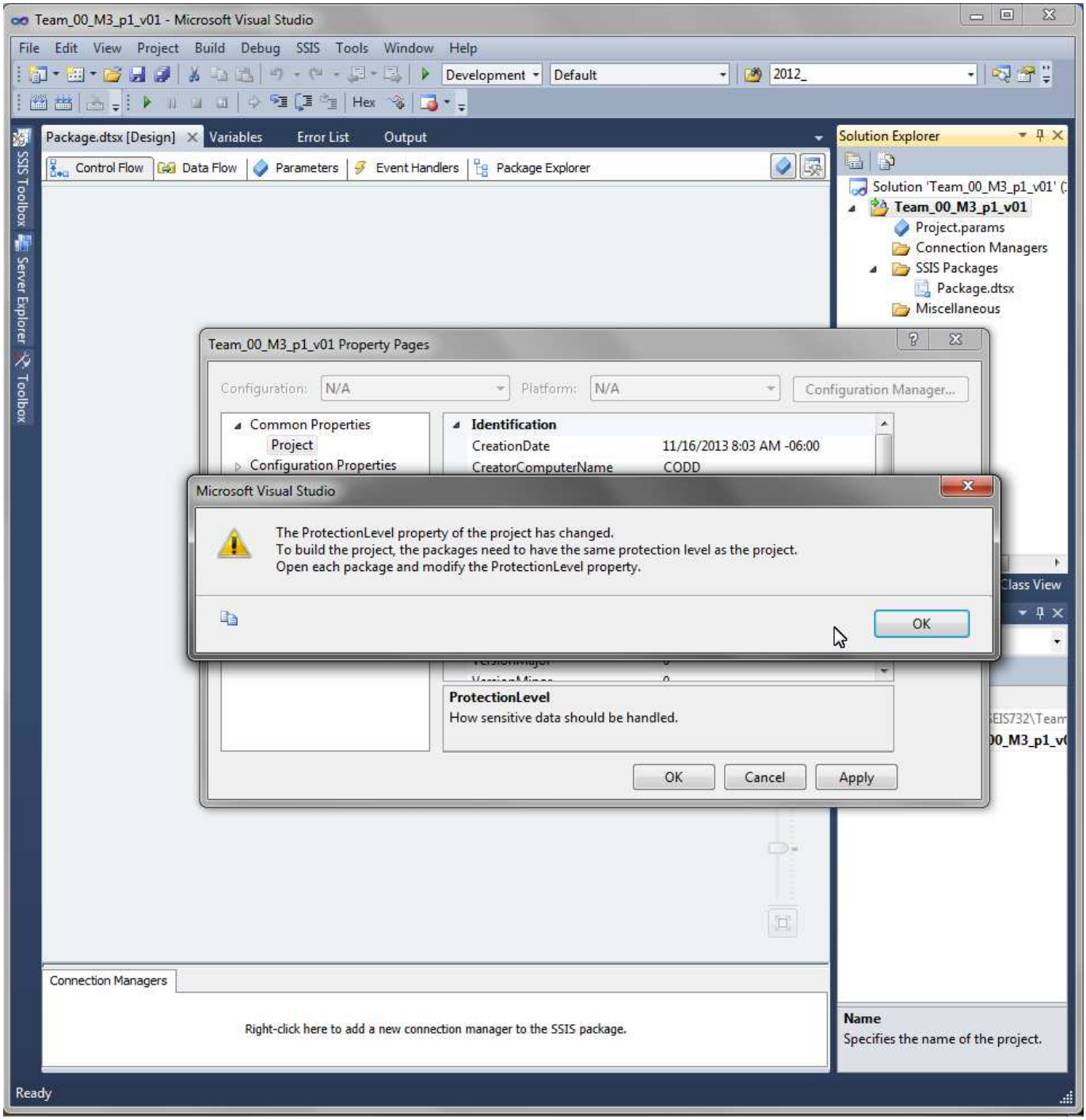

Figure 05 – Solution Password Warning Dialog Message

Now we need to set the protection to the same level for the package—to do this, **right click anywhere on the control flow tab for the package** (the background behind any of the control tasks in the package) and select properties, or simply select the package in the drop-down box at the top of the properties window.

If the properties window is not visible, try pressing (F4) Function Key, or selecting View→Properties Window. The properties widow might be "floating" like it is in Figure 06, it might be docked on one of the edges of the dialog, or it might be hidden. The "F4 / view" action should make it appear somewhere.

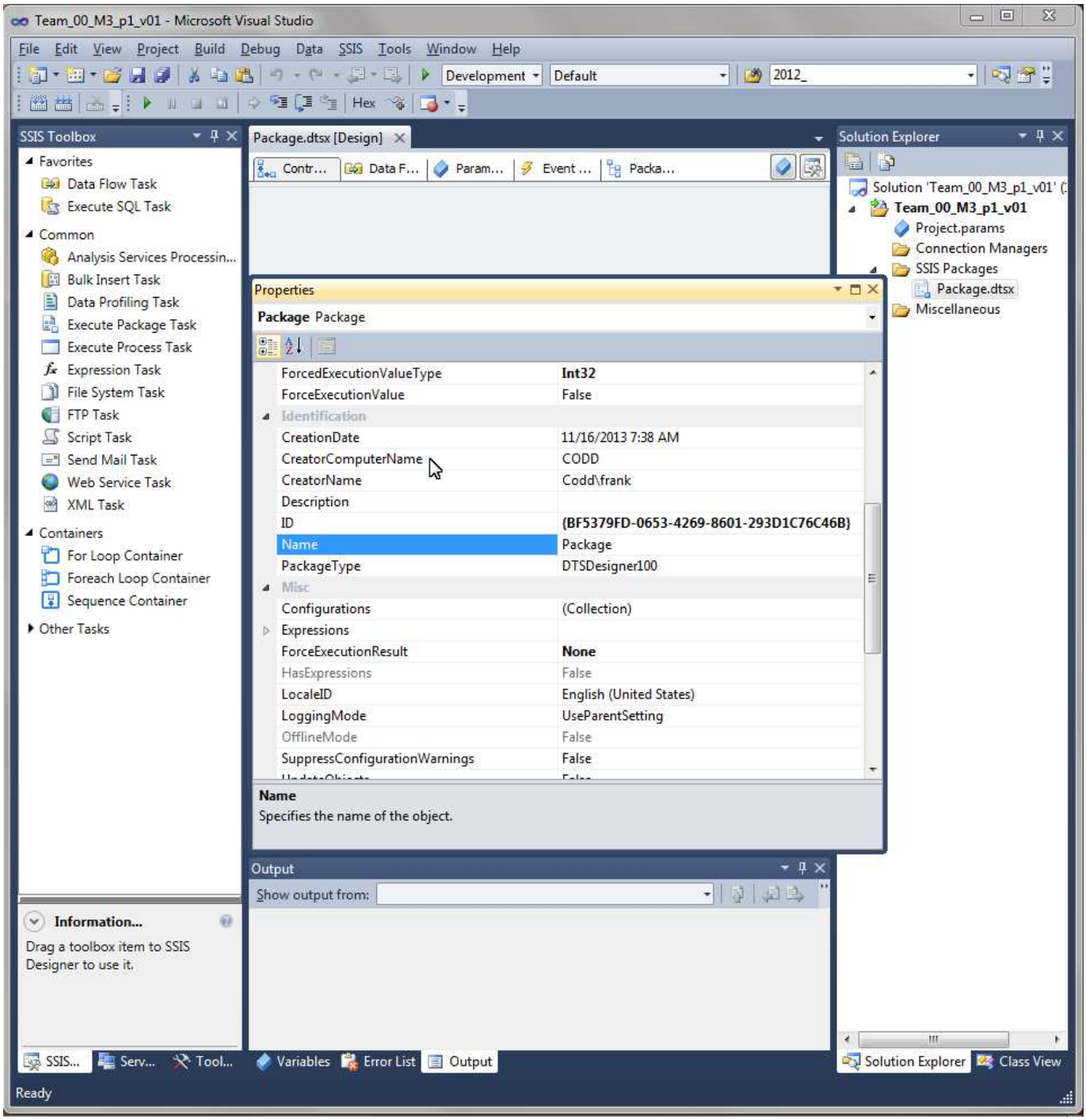

Figure 06 – Viewing / Setting the Package Properties

Next, change the **ProtectionLevel** property from "**EncryptSensitiveWithUserKey**" to "**EncryptSensitiveWithPassword**" (see Figure 07).

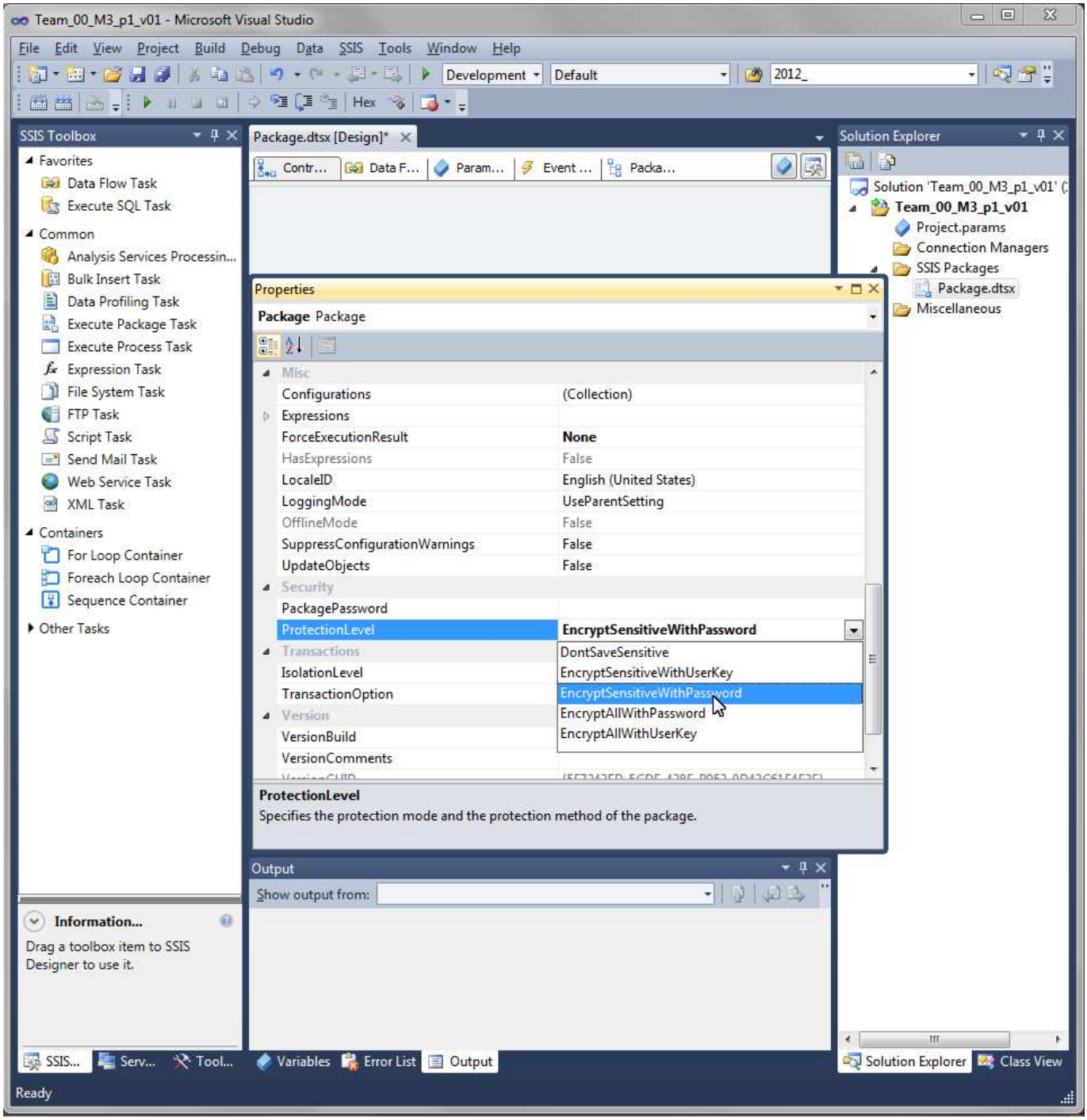

Figure 07 – Viewing / Setting the Package ProtectionLevel Properties

The properties can be listed by category or alphabetically (the two little icons under "Package Package" in the Properties Dialog). I am showing the categorical listing here – clicking on these icons can switch back and forth between these two options. After changing this setting, set the "PackagePassword" property to the data bundle password (**the same as you did for the solution**). This is important because all your team members (**and the instructor** when you turn it the final deliverable) need this password to open your files—make sure you type it in with the correct case, it is CASE-SENSITIVE!!! To enter the value in this step, simply click on the "…" button next to PackagePassword and enter the new password, twice as shown in Figure 08

Do this **RIGHT AWAY** don't do it **AFTER** you have created a Connection Manager or some other tasks because it might be a real headache for you (depending upon the other actions and circumstances) if you do wait… in fact you might want to test this out after you create your first connection manager and first task to **make sure it is working before you go to far**…

In other words, create a project, do this step, then create a task with a connection manager and connection to one of your databases. Next, you simply save the project, zip up the directory and pass the zip file to **another teammate on another computer**. If you want to verify things at this point, you should have your teammates log in using **THEIR account (not yours)**, unzip the project file and then try to open it, run it, and edit it. It should prompt for the password at some point—either when they first open the file or when they attempt to open the package inside it.

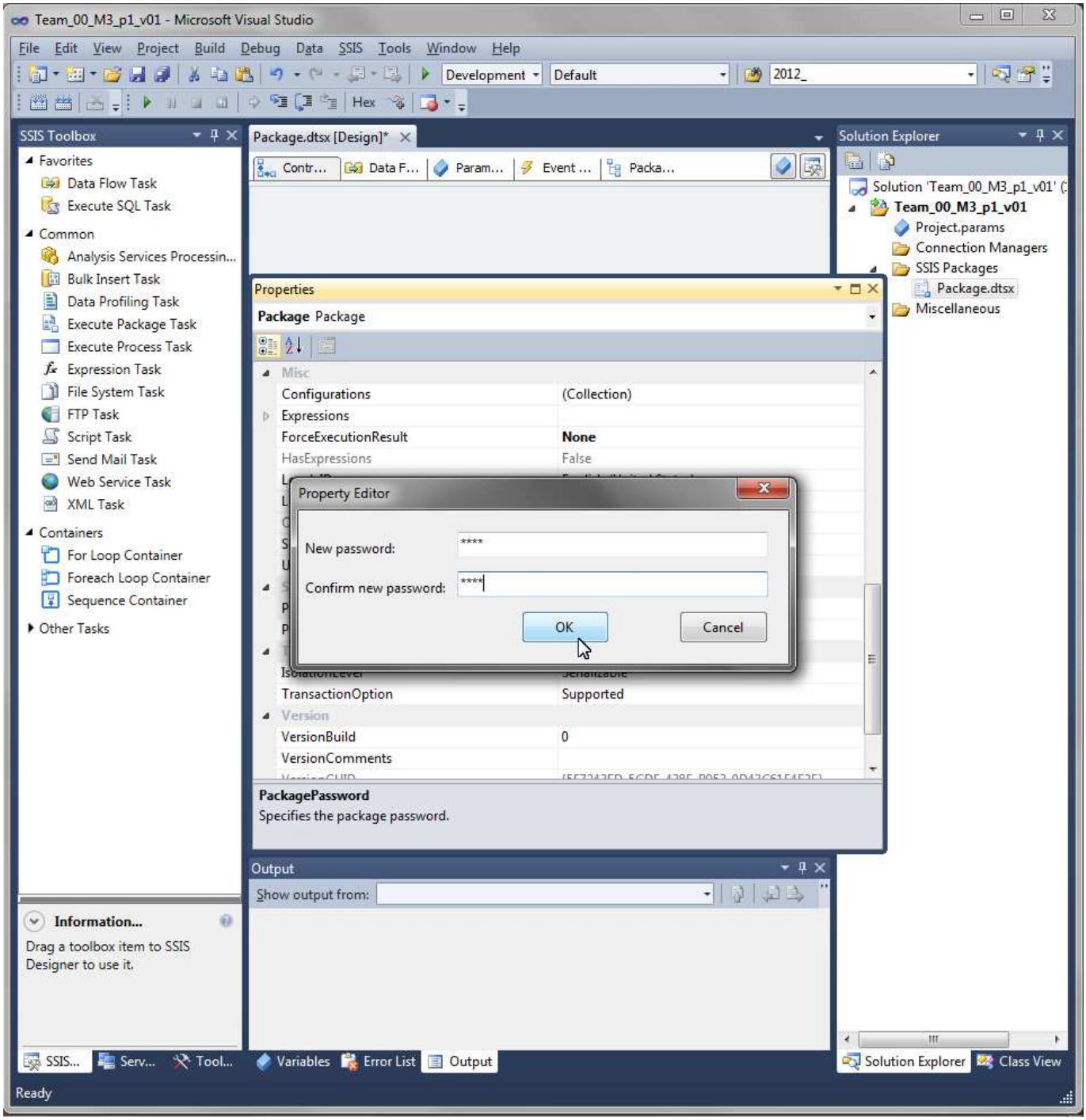

Figure 08 – The "PackagePassword" Property Confirmation dialog

IF YOU DON'T SET THESE UP PROPERLY, ALL DATABASE CONNECTION DETAILS WILL BE LOST WHEN YOU MOVE TO A NEW MACHINE OR USE A DIFFERENT WINDOWS LOGIN!!! IN OTHER WORDS, YOU WILL RECEIVE ZERO POINTS!!!

If you have any problems let me know…

AND MAKE SURE YOU TELL ME THE PASSWORD WHEN YOU SUBMIT YOUR M3!!!!!

IF I CANNOT OPEN YOUR ETL PACKAGE, I CANNOT VERIFY THAT IT WORKS AND YOU WILL NOT EARN ANY POINTS!!!!

E Containers

I strongly suggest that you use containers; in particular, the Sequence container task is very useful. Simply Drag and Drop it into the package or even into another container. This allows you to apply precedence constraints or set properties on the container that affect all of the individual tasks inside it. The tasks inside it execute pretty much the same as they would otherwise except that they can only execute if and when the container can execute.

This is very convenient for debugging and also provides nice encapsulation and abstraction for your package. You can name these containers after the activity being performed—for example, a container used for loading the MSA Dimension can be named something like "Load MSA DIM" (see Figure 09). You can also simply name the container after the DIM it is processing, for example "MSA". You can also resize the container (or any component) and move it around within the package. If you cannot see things easily, you can scroll around in the package and can also change the zoom level.

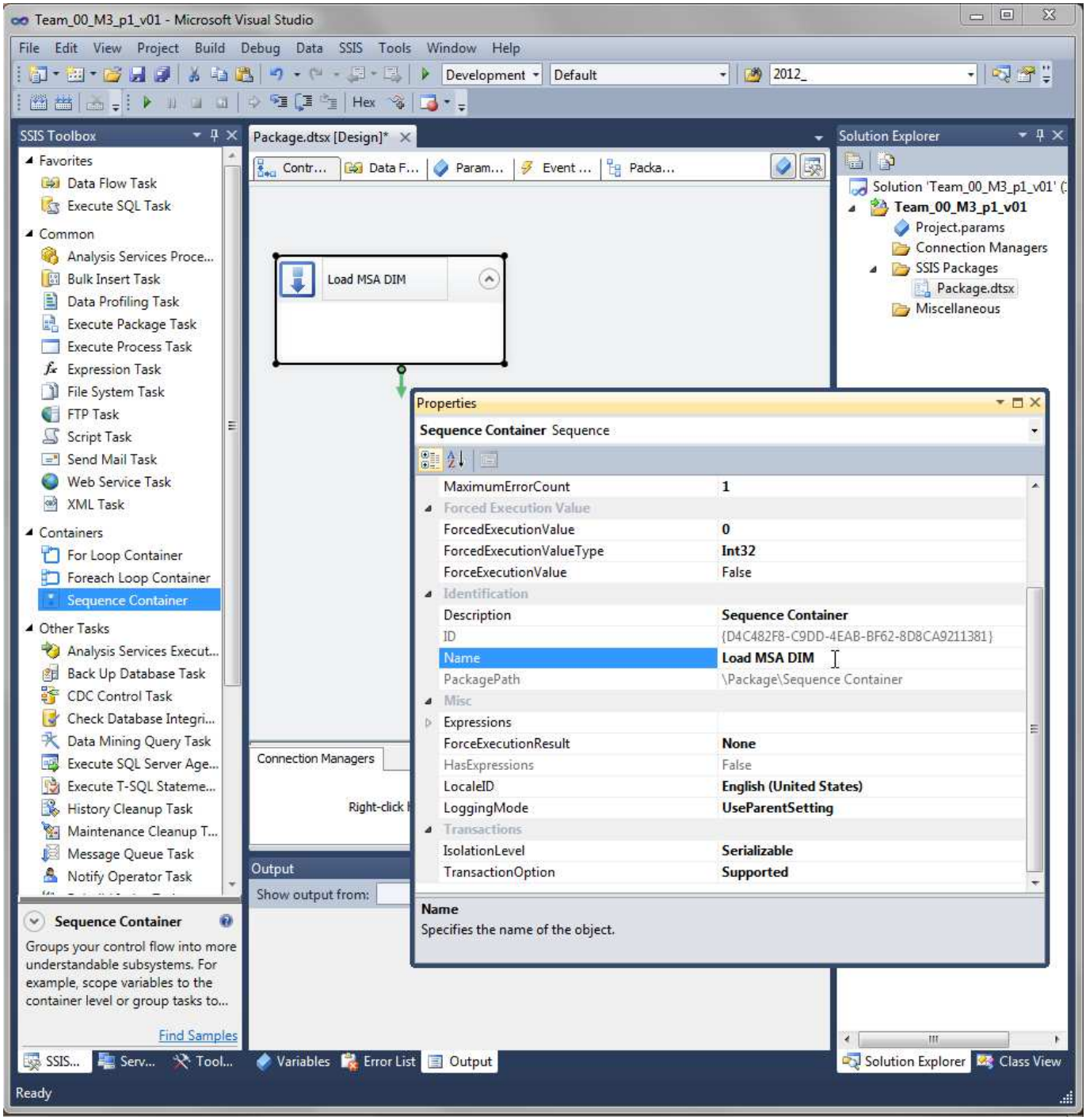

Figure 09 – Sequence Container Control Flow Tasks

F Delaying Validation

If you have a component (task / container / data task / etc.) that needs to directly or indirectly reference a database construct (a table, view, etc.) that **does NOT exist yet or might not exist when the package is first loaded** (for example if you are going to create a staging table, or create a view in some other task before **this** component runs), then you need to delay the validation for the component making the reference. If you don't do this, the package will fail when it is first loaded and the SSIS determines that the referenced construct does not exist. This situation can be quite frustrating since the problem only manifests itself when you move the package to a clean machine where the referenced construct **does not** exist. Once this situation occurs, the environment can do some bad things (such as disconnect the connection manager, reset all details for the referenced construct, delete all of your table and column mappings, clear all the username and password details, etc.)—which can be quite painful to manually restore.

Luckily, there is a property which can be found on the property dialog for the component called "**DelayValidation**"—set this to TRUE by selecting the value from the drop-down list for the property (see Figure 10). You can also double click on the name of this property to toggle the true/false value for it. The component will still be validated eventually, but now the system will wait until the package or component is actually running rather than trying to do this validation during development or when the package / component is first loaded.

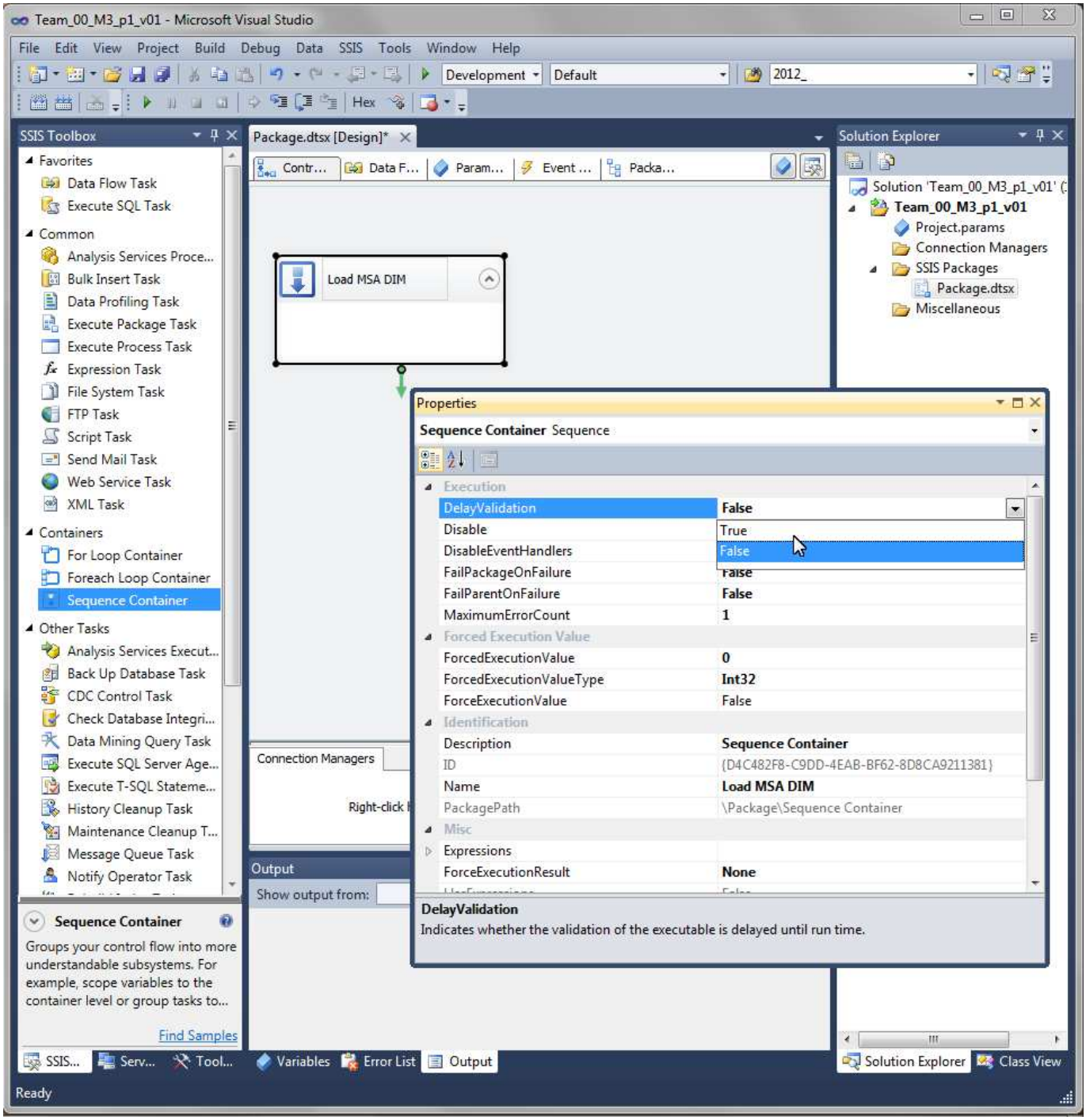

Figure 10 – Viewing /Setting the "DelayValidation" Property for a Sequence Container Task

III Making a Simple Package

A Creating an ETL Solution

Now it is time to start the hands-on portion.

Start out by launching the MS SQL Server Data Tools program. This is found by going to the Start Menu, and selecting "Start → All Programs → Microsoft SQL Server 2012 → SQL Server Data Tools". This will launch the Data Tools, which is a plug-in for Microsoft Visual Studio 2010 / 2012 Dot Net.
Next, select the menu items necessary to create a new ETL SIS project (i.e., if it does not appear in the getting started area, you should select "File→New→Project"). You should then select "Business Intelligence" for the Project Template on the left hand side and "Integration Services Project" for the actual Template. By default, BIDS wants to create a directory for this new project underneath your profile directory, as shown in Figure 11. In other words, usually in some directory underneath:

"<drive>:\Users\<username>\documents\visual studio 2010\Projects".

While you can choose to leave this here if you want to, personally, I find this to be a bad idea. Usually, I create a separate directory, either off the root drive or at least the desktop, and use it as a starting point. For example, "D:\fsh\SEIS732", or "C:\Users\fshaug\Desktop\SEIS732"—I also do **not** recommend using your "U drive" for this on campus!

The U Drive can time out when the system or network is under a heavy load.

A USB thumb drive is also a bad idea—development tools write many files and perform many reads and writes. USB thumb drives are not intended for this type of use and abuse.

One of the reasons I dislike the default directory (and even the desktop or users subdirectory) is the simple fact that it can be destroyed / recreated on login if there is an issue with the domain controller. Another reason I dislike it is that it can be configured to contain spaces (the old default was "<drive>:\Documents and Settings"). The new default has no spaces in the prefix, but the tool has spaces in the last part. Spaces in pathnames make scripting and automation difficult and error-prone. Even though we are not going to need or use any scripting like that in your deliverable, it is a bad practice / habit to get into. Historically speaking, spaces can cause problems for the tools themselves in some circumstances.

The MakeAll.cmd file does not work when spaces are in the directory names it needs to run in.

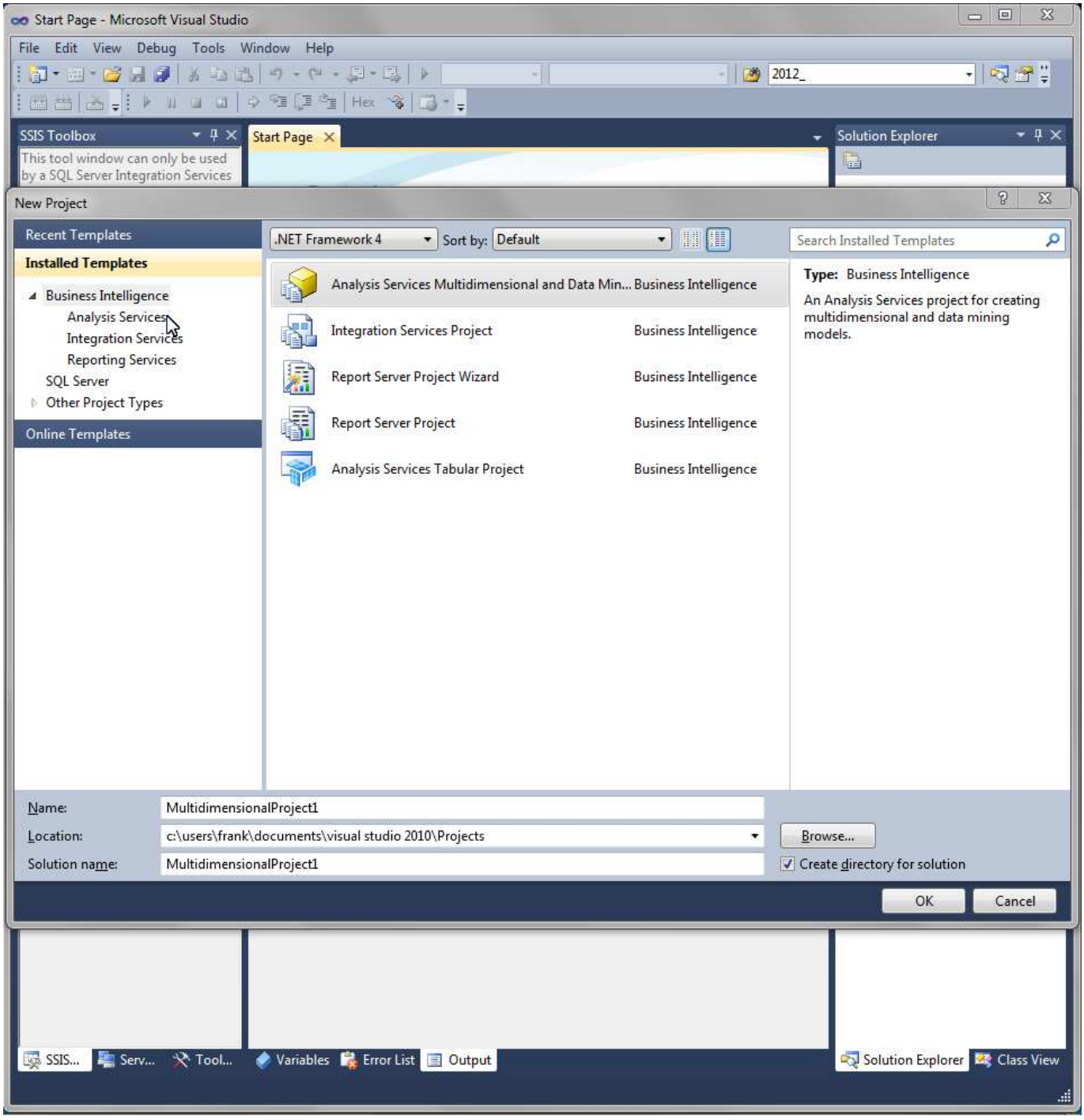

Figure 11 – New Project Dialog

Fill in the other details for the new project. Enter the name for the solution following my naming convention and allow it to create a subdirectory for the new solution as shown in Figure 12.For example, here I entered "**Team_00_M3_p1_v01**" for the solution name (See Figure 12). When you turn in your deliverables for this milestone, zip up this directory (e.g. "**D:\fsh\SEIS732\Team_00_M3_p1_v01**"), which will contain several files.

You and your teammate should submit a single solution with a single package for your final deliverable: You should include your team information as the first part of this name. It is OK to include version information in the file / folder if you like, e.g. "**Team_xx_M3_v01**" versus "**Team_xx_M3_v02**". Obviously, replace the xx with your team number, and increment the v01 as needed.

MAKE SURE THAT DIFFERENT TEAM MEMBERS WORKING ON DIFFERENT PARTS MERGE THEIR WORK INTO THE FINAL SOLUTION AND FINAL PACKAGE.

NB: in the screenshots in this document, I used the "F" drive, and I tried to use "**Team_00_M3_p1_v01**" but if the names are slightly different, please simply ignore these differences in the figures.

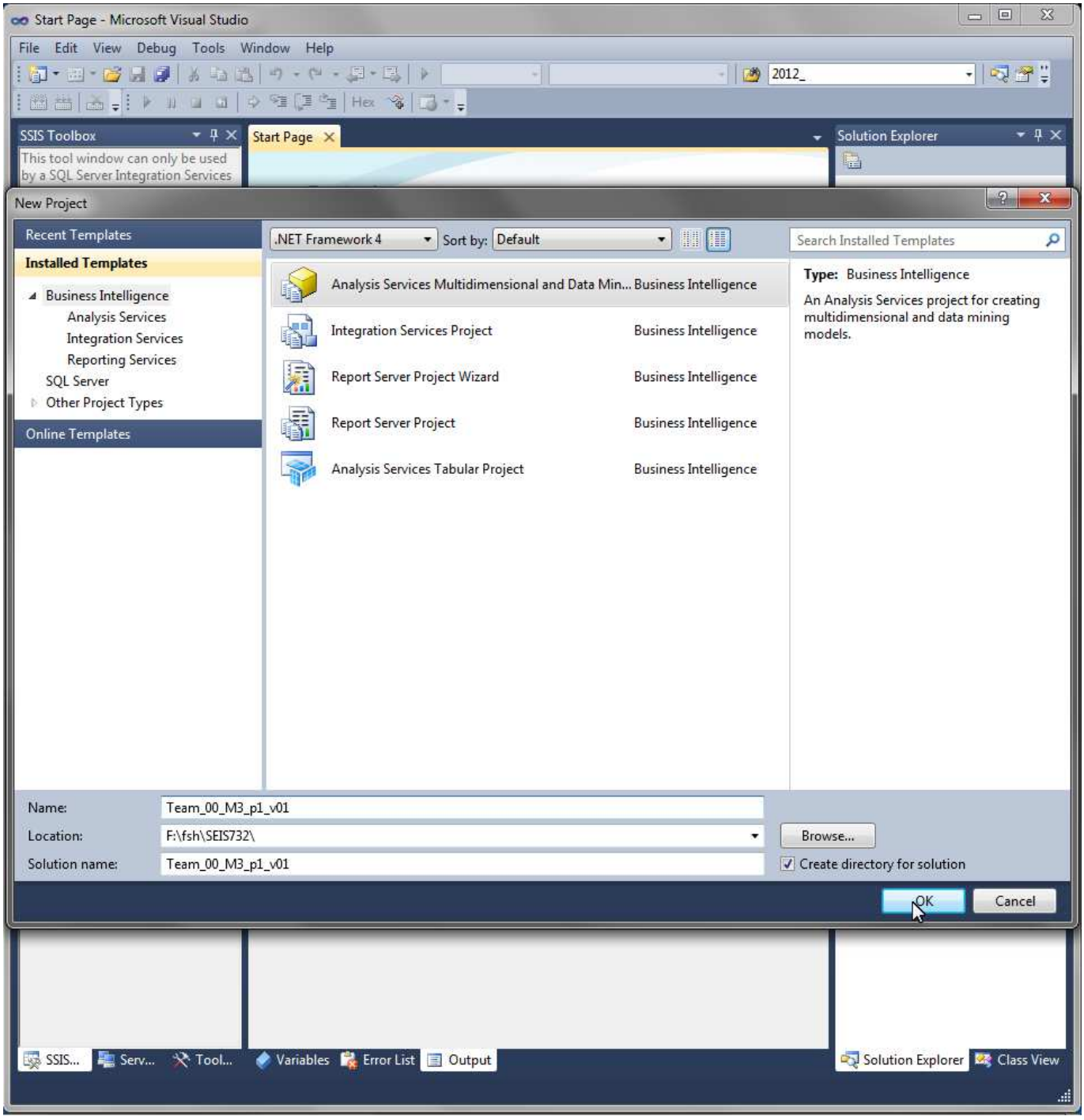

Figure 12 – Filled in New Project Dialog

B Designing the Package

After you click OK in the dialog shown in Figure 12, the package design dialog from Figure 01 will be shown. Your view might be slightly different then the screenshots in this document depending upon how the user interface was configured which is also a per-user setting, so this can vary depending upon who is logged in and what machine you are on. In Figure 01, the Toolbox is "hidden" over on the left hand side of the dialog, in some of the other dialogs; it has been "docked".

This is a good time to set the "ProtectionLevel" and "PackagePassword" properties for the package as shown in Figures 1, 2, 3, and 4!

I suggest that you might want to "dock" the Toolbox bar on the left hand side. To do this, left click the bar and when it opens, left click on the "push pin" at the top of the menu bar (right hand side), which will "permanently dock it", at least until you click on the pin again. You don't have to dock it if you do not want to, just remember to go over there and open it before selecting a component. You can always move the toolbars around, dock them, undock them, and even auto-hide them. If you can't find a toolbar you can always go to the view menu to find it.

C Adding Control Flow Elements

Now you are ready to add control flow elements. For development and testing purposes, it is a good idea to create some task(s) that return the system to a known state before attempting to run any other ETL. In other words, we want to delete all rows from the Star Schema Dimension tables, drop, and recreate any views, stored procedures, or staging tables, etc.

Let's start by creating a sequence container control flow task that we discussed in Figure 09. After you have created the "Load MSA Dim" container, we will add control flow and data flow tasks inside it.

To do this, we can use an "Execute T-SQL Statement" task (which is found on the left hand side Toolbox under the "Maintenance Plan Tasks") or an "Execute SQL" task (which is found on the left hand side Toolbox under the "Control Flow Items"). The difference between these tasks is simply this: an Execute SQL task can only use "standard SQL syntax"—a T-SQL task can use standard syntax as well as MS SQL 2012 specific syntax. For our purposes here, the T-SQL tasks are probably what you want to use most of the time if not all of the time. Remember to set the DelayValidation to true for this as shown here again in Figure 13.

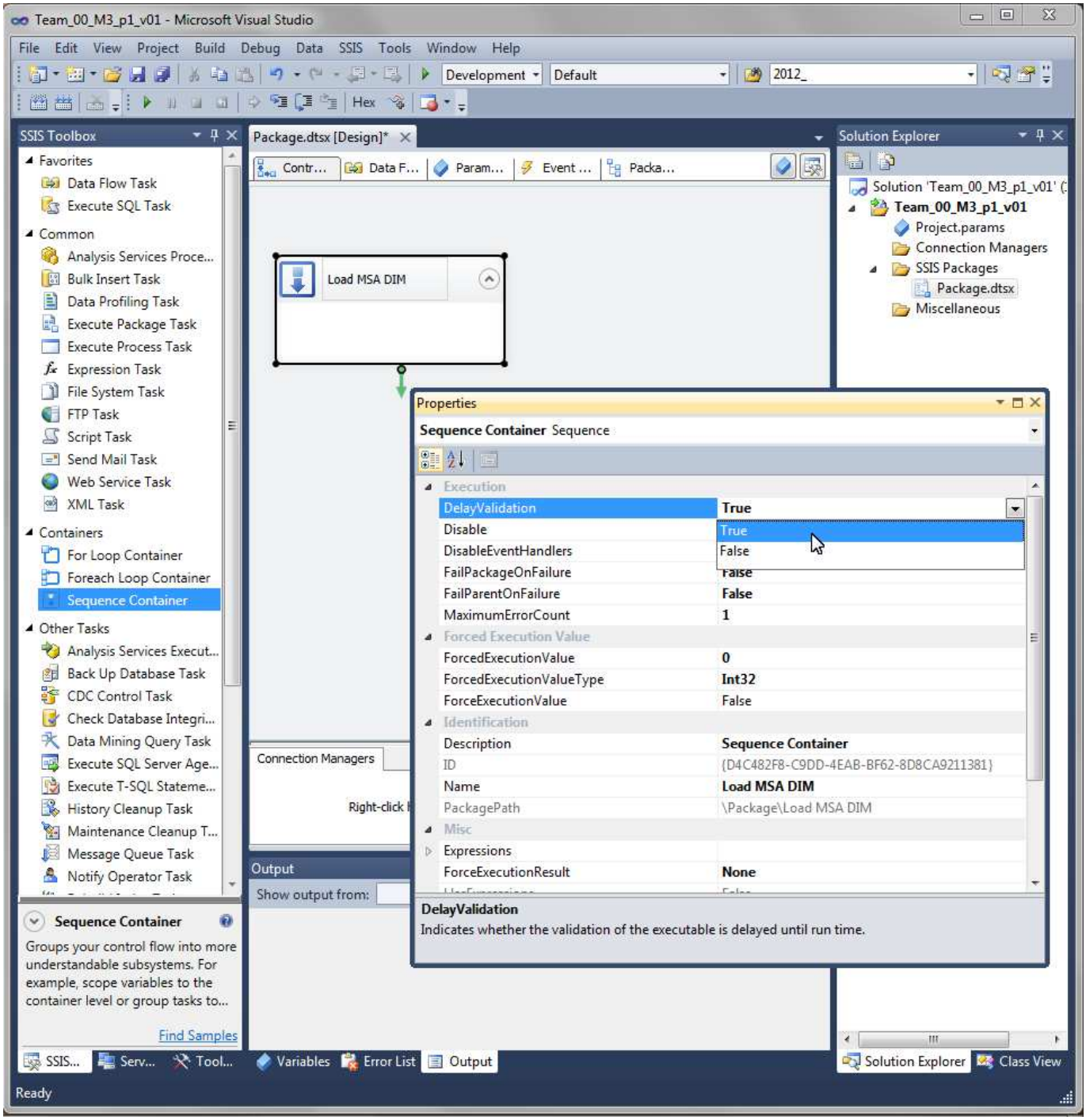

Figure 13 - The Load MSA Dim Sequence Control Flow Task With DelayValidation

Let's create another control flow task inside the sequence task.

1. Drag and drop the desired task type (an "Execute T-SQL Statement" task) into the Load MSA Dim task (if you drop it in the Package workspace area or double click on the desired task type, it will add it to the package workspace, but not the Load MSA Dim container).

   Initially, it will look something like Figure 14.

   We can resize it and move it around inside its container task or the package workspace if we wish.

2. Now, we need to rename the new task, so either right click on the task in the workspace dialog and then select rename or click on the **NAME** twice (but don't double click, or click on the icon).

   You can then enter a more meaningful name (here let's name the task "Delete MSA").

3. This would also be a good time to check / set the "**DelayValidation**" property we discussed earlier, if it is applicable (depending upon what the task type is and the details about the constructs it is using). When in doubt, there are very few disadvantages to delaying the validation for our purposes in this project, so you can usually just set this to True, most of the time. In some circumstances, if you change the definition of a table or view, you might need to toggle this setting for other components that depend on the table or view (turn it back on after making the change, edit the component, and then turn it back off).

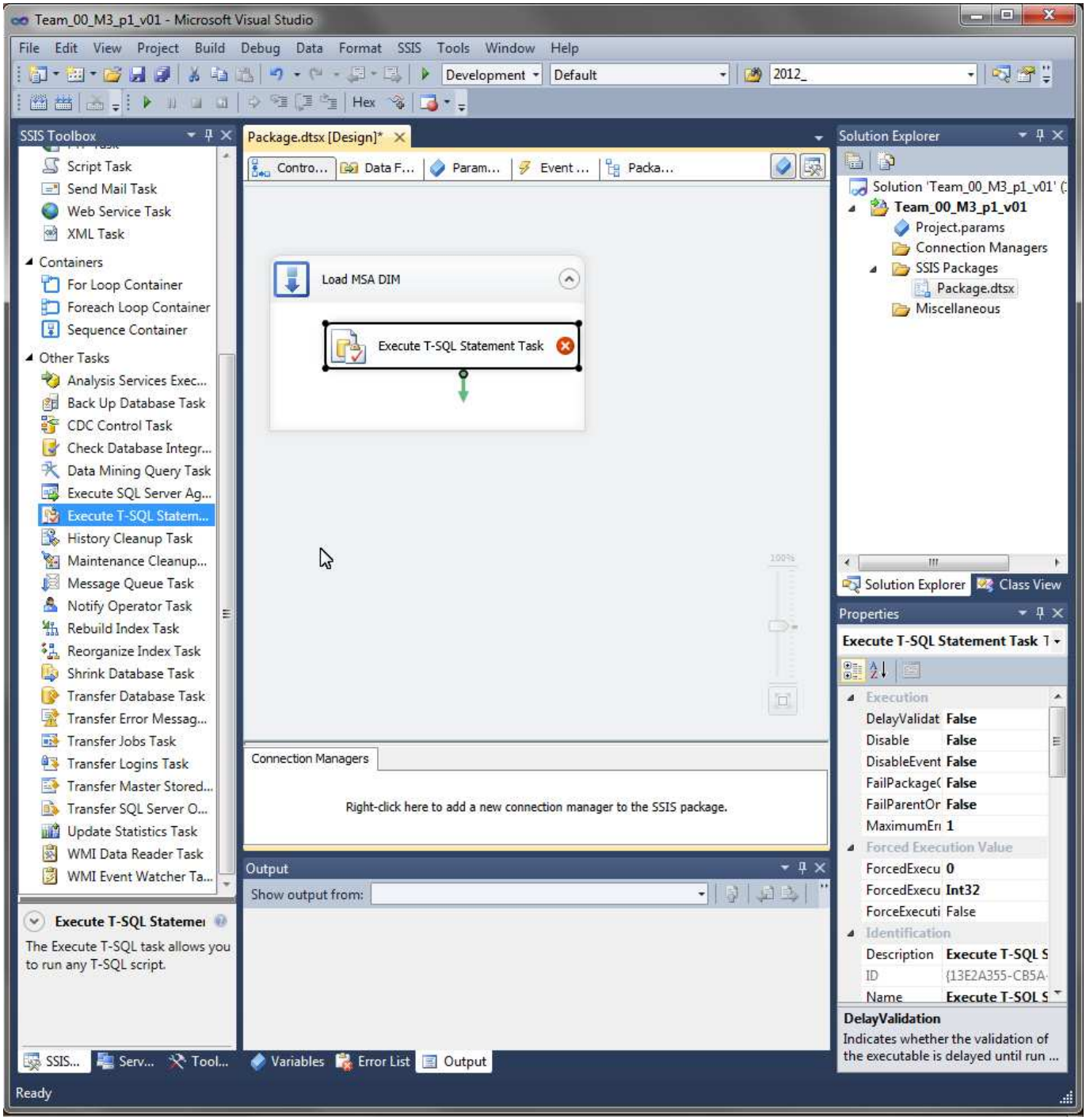

Figure 14 - New T-SQL Inside Load MSA Dim Container

4. Now we need to define the task, so double click the new Execute T-SQL task (or Right click it and select Edit).

   This brings up the edit dialog for the task. Here we can specify the connection details used by the task and enter the T-SQL Statement or Statements (See Figure 15).

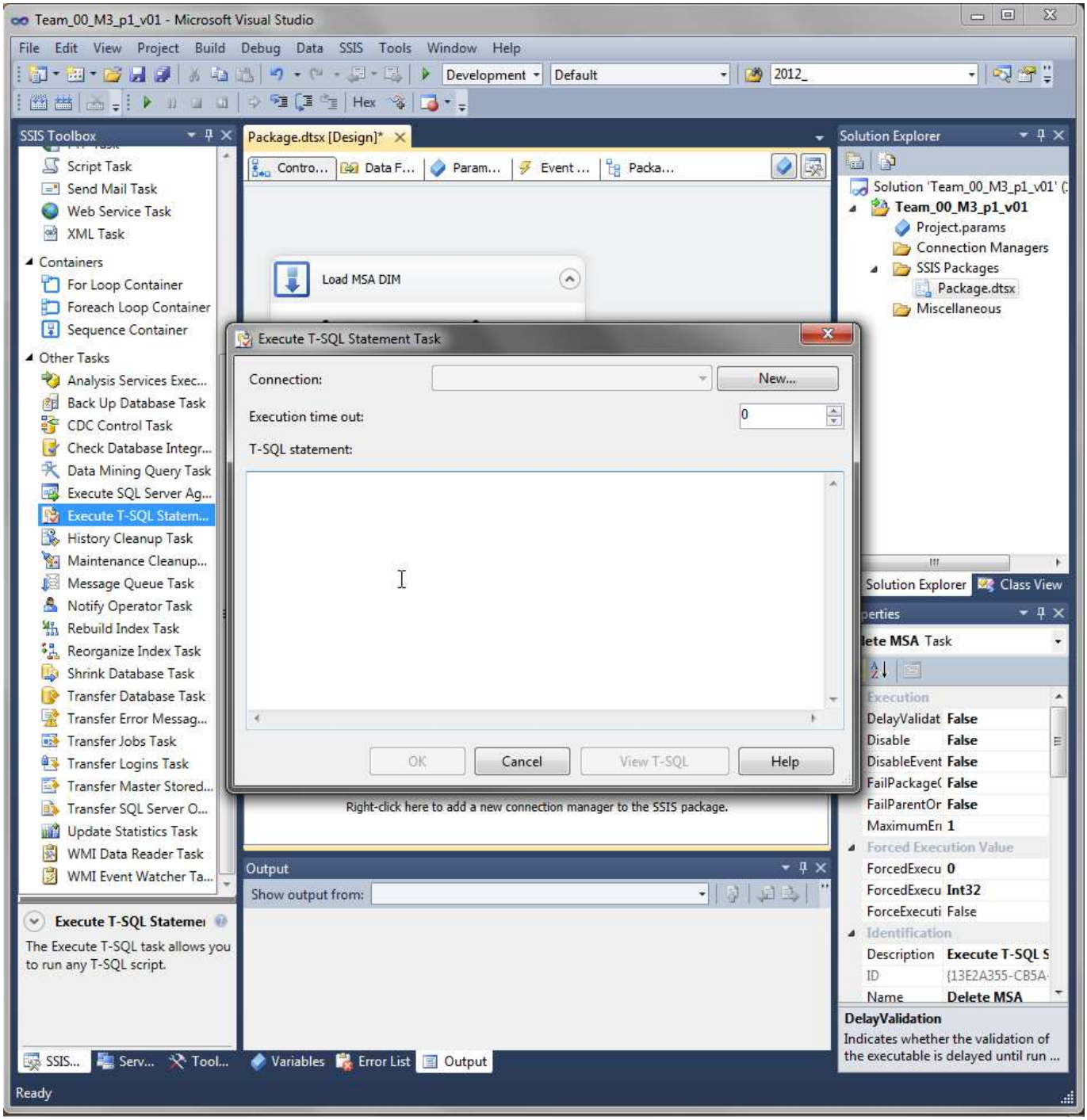

Figure 15 – Empty Execute T-SQL Statement Task Dialog

5. Since this is our first task, there are no connections defined yet.
   To define a connection, click on the "new" button on this dialog, which will bring up another dialog box (See Figure 16).

6. Specify the Connection Name next (here I used "Star_Schema", but you can call it anything you want — I recommend using a name based upon the database, you usually will not need more than one for each database).

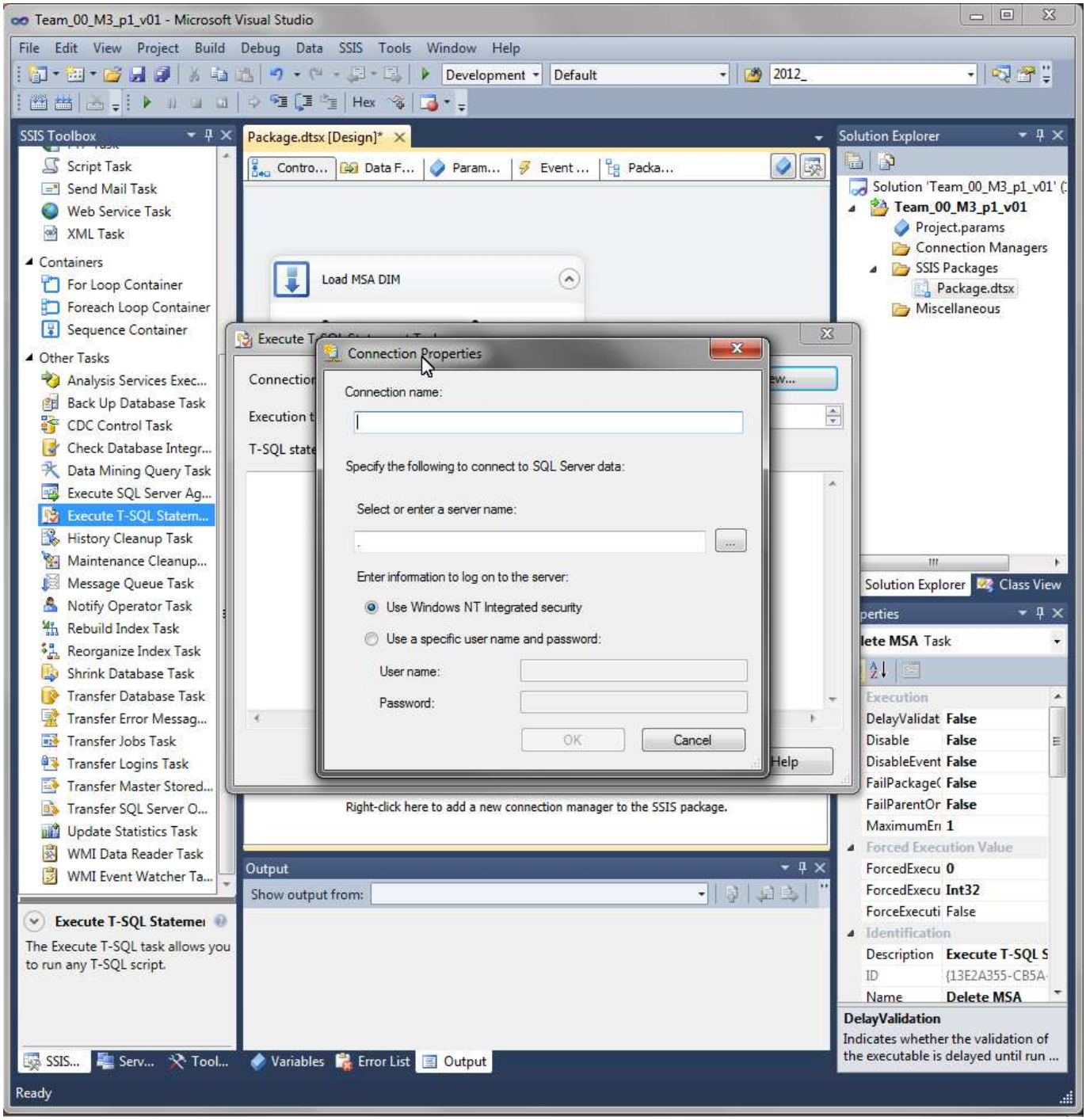

Figure 16 – Empty Connection Properties Dialog

7. Enter the Server Name. If you select the "…" button instead, it can take an enormous amount of time to find the list of available servers (if it doesn't time out). This button will also use the full machine name rather than the "." shortcut—so you are better off typing it directly. Using the "." server name is shorthand for "localhost", but we also need to specify the instance name. For our project we should **ALWAYS** use ".\GPSSQL" (type a period, followed by a backslash, then GPSSQL – don't type the quotes). If you did not install this instance name on your personal MS SQL 2012 installation, you will regret it—fix that NOW.

8. Now enter the username and password details. We need to select "Use a specific username and password" here, and enter the details.

   (NB, the username and password are **CASE-SENSITIVE**).

REMEMBER:
YOU MUST USE .\GPSSQL!
You CANNOT USE TRUSTED SECURITY FOR THIS!
DO NOT USE the "Windows NT Authenticated" option for ANY connections to the four databases in this milestone!
POINTS WILL BE LOST IF YOU USE TRUSTED SECURITY!!

Here, we are going to be deleting all the rows from the MSA dimension table, so we need to connect to the Star Schema database for our team. In this example, I will enter the username as "SEIS732_Team_00_Star_Schema_User" and type the password for my "Team 00" user.

Obviously, you should use your team number instead of 00 and your password.

The password for your team is posted to your team's blackboard group discussion area. See Figure 17 for an example of the connection properties dialog with all the details filled in.

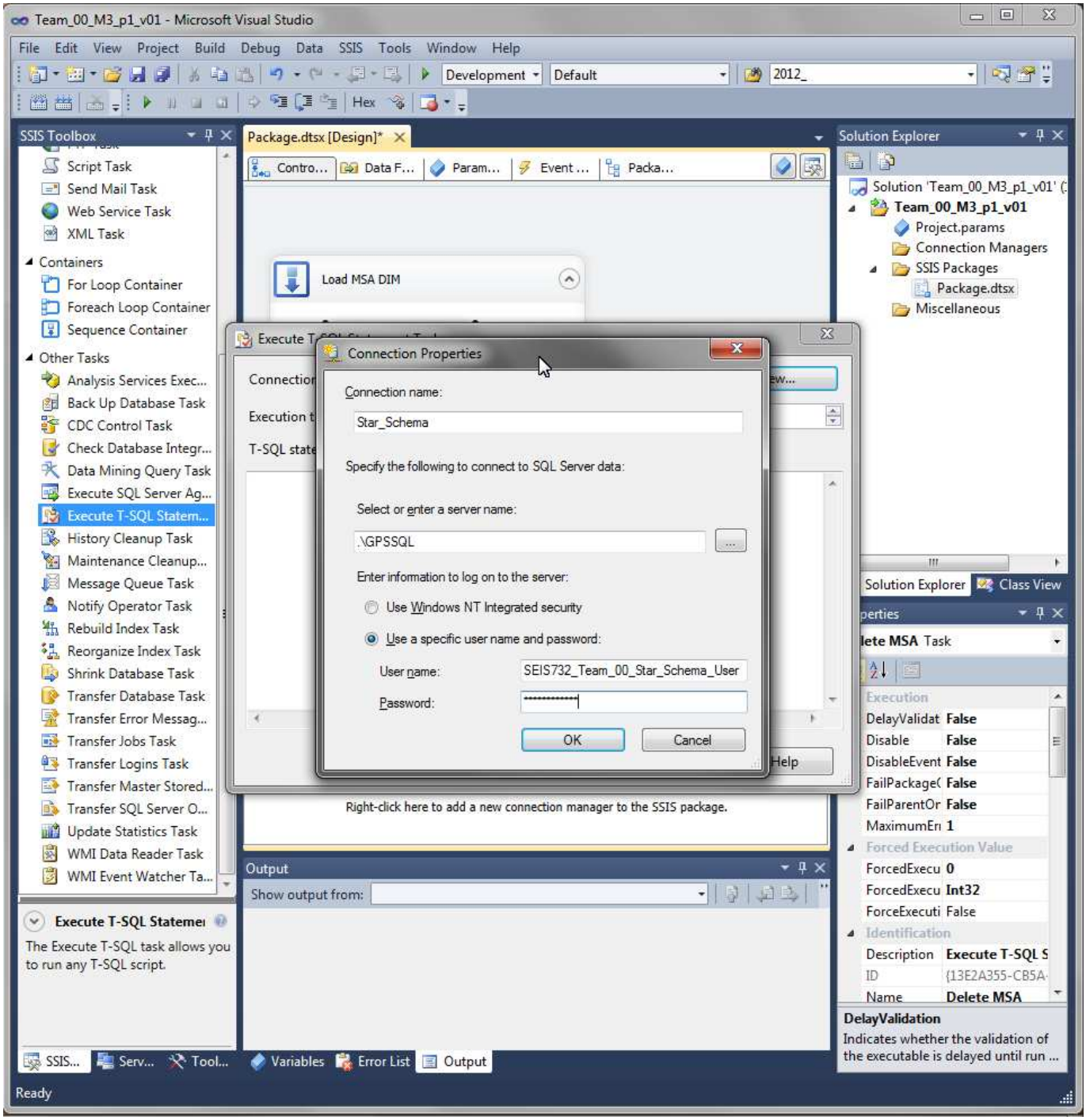

Figure 17 – Filled-In Connection Properties

9. Click OK on the connections property dialog to save these settings and go back to the "Edit Execute T-SQL" dialog box.

10. Now we can continue on our way—once we have created a connection, we can simply select it from the drop-down rather than repeating steps 6-9. In other words, normally, we can skip from step 5 to this step once we have added all the necessary connections via previous dialog box activities.

    Select the correct connection in the drop-down box, and type the SQL statement(s) into the "T-SQL Statement" area. Make sure to put the semicolon on the end of the SQL!

    Here, we want to type "DELETE FROM MSA;"—without the double quotes!

    You can also use cut, copy and paste operations for the SQL in this dialog box—for example you might want to use SQL Server Management studio to build or test your SQL prior to copying it into this dialog box.

    NB, the "DELETE" and "FROM" are case insensitive but all of our DDL names (logins, users, passwords, databases, tables, columns, views, etc.) are **CASE-SENSITIVE!!!**

    See Figure 18 for an example of what this looks like.

11. Click OK to save all these settings and finish editing the task.

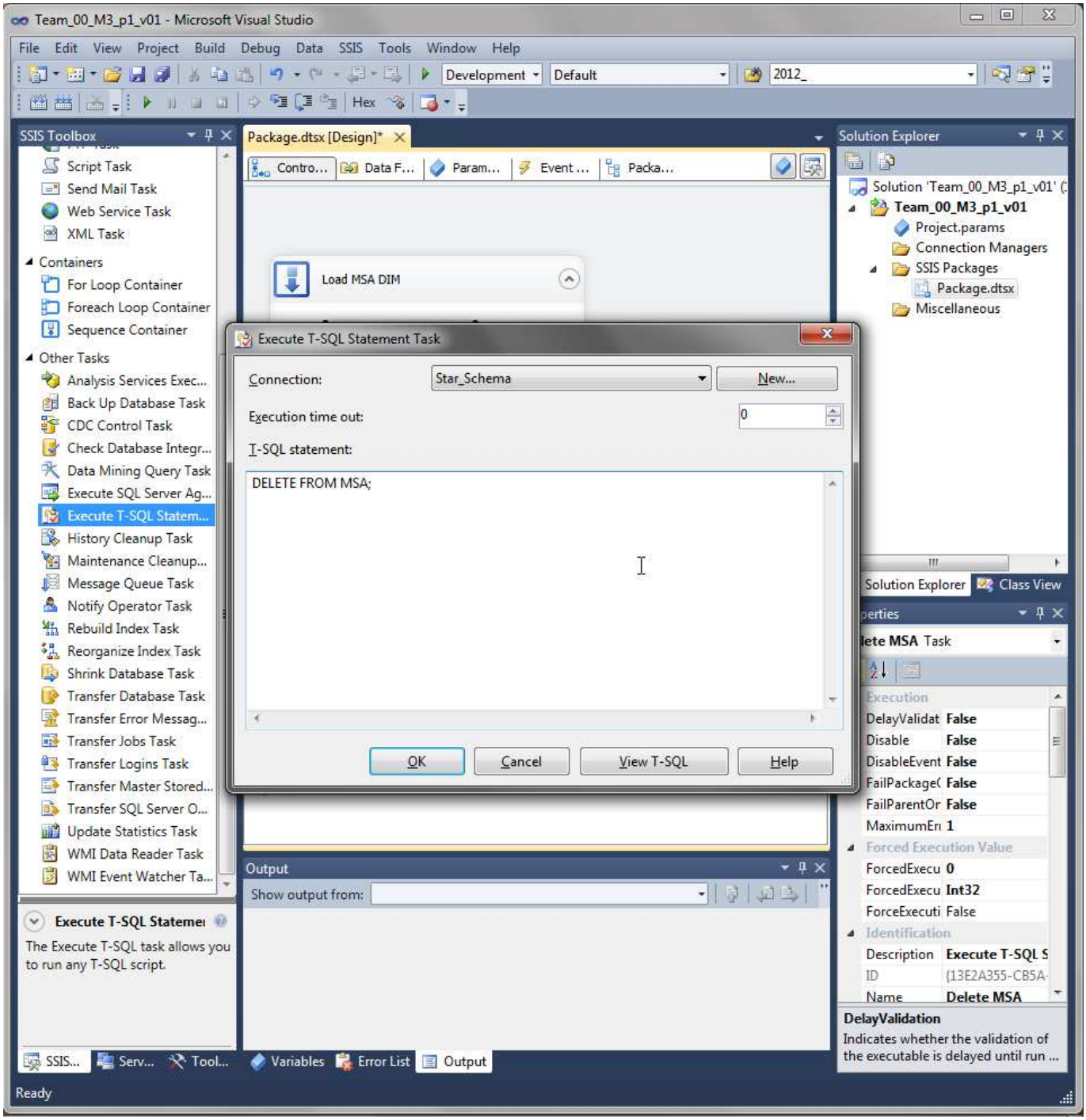

Figure 18 – Completed Execute T-SQL Task Dialog

D Adding a DDL Task and Precedence Constraint

The source for this Dimension is most-likely using SCD type 1 (overwrite) because it is controlled externally (not by out RRV organization)—it is defined by the Census Bureau. However, if we wanted to support SCD type 2 we would need to have a surrogate key for the Metropolitan level (the Micropolitan level is the lowest level and therefore would use the PK from the table for this purpose). Even with the SCD type 1, we should really have a surrogate key for the level, since we cannot control the uniqueness of the source for any of the DATTs. Therefore, because we have included a MBR KEY DATT in our MDM for the Metropolitan level of the HIER, we need to load it somehow. To generate this surrogate key, we can use staging tables or other techniques to automatically generate the SK value for the MBR KEY.

Even though we are **IGNORING SCD** for this milestone, we still need to populate the DATT! In other words, we are not going to try to do and incremental load, and are not going to try to do SCD population here but we need to have unique values for this surrogate key DATT. There are several techniques we can use but the simplest is to mimic the way we do surrogate key generation for the DIM itself. This is done with an identity column in MS SQL, but we can only have one identity column per table so we need to create a staging table for this purpose. We can create this table in any database, but for this example, we will create it in the Sales_Org database. Often, the DBAs for the OLTP systems will not allow this in the real world but of course there is less overhead when we use a staging table that is in the same database as the source—so there is always a tradeoff. In our project, we can create any tables and views we need as long as we use the correct user account so this is not an issue for us here.

WARNING THIS IS NOT NECESSARILY 100% ACCURATE WITH RESPECT TO THE ACTUAL TASK YOU NEED FOR YOUR M3 ETL —IT IS MERELY AN EXAMPLE, BUT SHOULD BE VERY CLOSE!

Let's create another control flow task.

1. Create another Execute T-SQL task in the package workspace and drag it into the "Load MSA DIM" container.

   NB: you can resize and move all of these things to make it easier to see the details and work with the components.

2. Rename the new T-SQL task "Create MSA Staging Tables".

3. Set its connection to the Sales_Org Database (Create a new connection and specify the server, instance, username, and password information appropriately).

4. Enter the following T-SQL into the dialog:
   (you can try it in Management studio first to check for errors, and remember to be careful about case-sensitivity when typing SQL!)

```
if (exists(select name from sysobjects where name = 'Stage_MSA' and type = 'U'))
	begin
		drop table Stage_MSA;
	end;
go

create table Stage_MSA
	(
	MSA_Metropolitan_Area_Key		int IDENTITY(1,1)	NOT NULL,
	MSA_Metropolitan_Area_ID		int			NOT NULL,
	MSA_Metropolitan_Area_Population	int			NOT NULL,
	MSA_Metropolitan_Area_Name		varchar(60)		NOT NULL,
	CONSTRAINT
		PK_Stage_MSA PRIMARY KEY CLUSTERED (MSA_Metropolitan_Area_Key)
	);
go
```

5. After entering the SQL, click on the Ok button.

6. We need to actually create this view before we can proceed—you can do this by executing the "Create MSA Staging Tables" task, or by copying the text into SQL server management studio and connecting to the sales org database as the appropriate user—for this purpose ONLY you can use the windows authentication in SQL server management studio to create this view for **DEVELOPMENT PURPOSES ONLY**.

Here's a brief synopsis of the SQL we just entered:

- Use a select statement to check the data dictionary (sysobjects) for the existence of a user table ('U' — as opposed to the system-defined tables) in this database named Stage_MSA.
- If there is such a table, drop it.
- Regardless of whether we dropped the table or not, create a new table named Stage_MSA.

Notice that we always need to drop the old table before we attempt to create a new one (otherwise it will error and our whole ETL package might halt!). If you use ANY views, stored procedures, temporary / staging tables, etc. you **MUST** create them in the ETL package like this!

This same SQL select can be used to check for views (type='V' for Views), stored procedures (type='P'), and other SQL constructs—look at MS SQL online documentation for more details.

E Adding a Data Flow Task

Now let's add some data flow! Under Control Flow Items on the Toolbox, Drag and Drop a Data Flow Task into the "Load MSA DIM" container in the package workspace. Rename it "Stage MSA". Then click on the "Delete MSA" task and notice the green arrow on the bottom (See Figure 19).

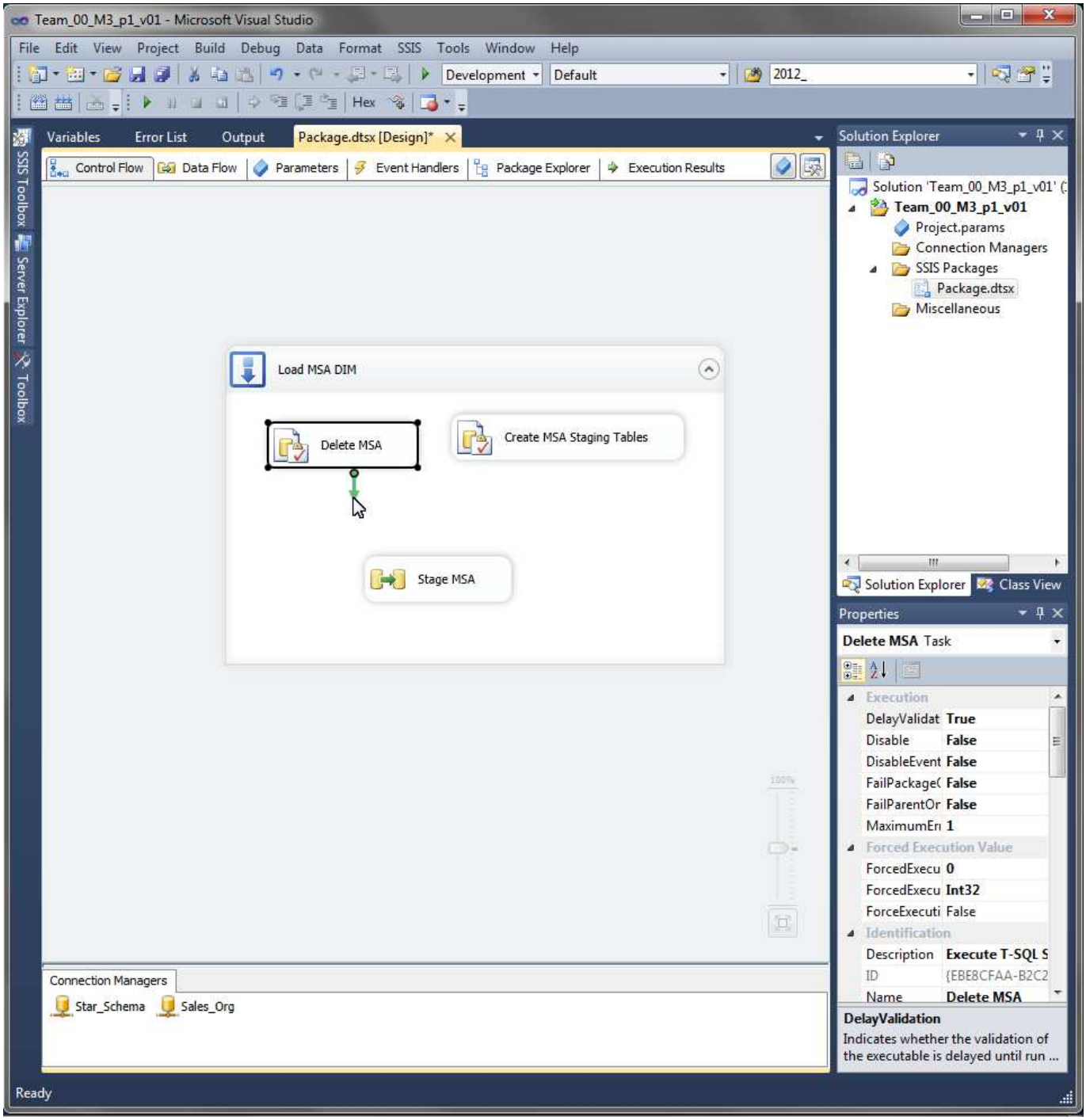

Figure 19 – Creating a Precedence Constraint from the Delete MSA Task

Click on the arrow and drag it into the "Stage MSA" data Flow Task. This is a precedence constraint. Green means "On Success". If you wanted some other type (On Failure or On Completion) you can right click on the Green Arrow and select the other type, as shown in Figure 20, but we want success here.

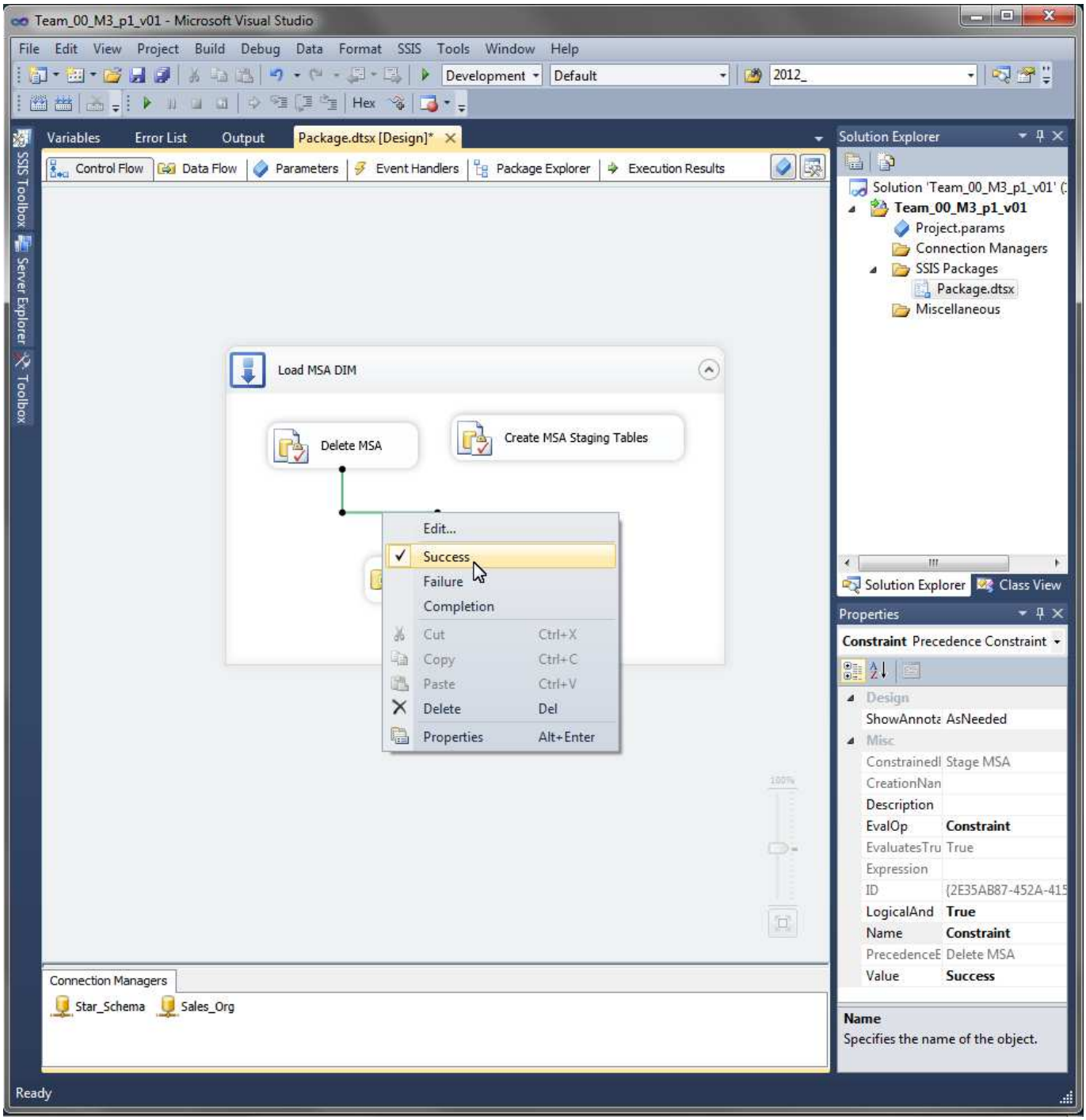

Figure 20 – Edit Precedence Constraint to Stage MSA Data Flow Task

Repeat this process for the "Create MSA Staging Tables" task. When you are done it should look like Figure 21.

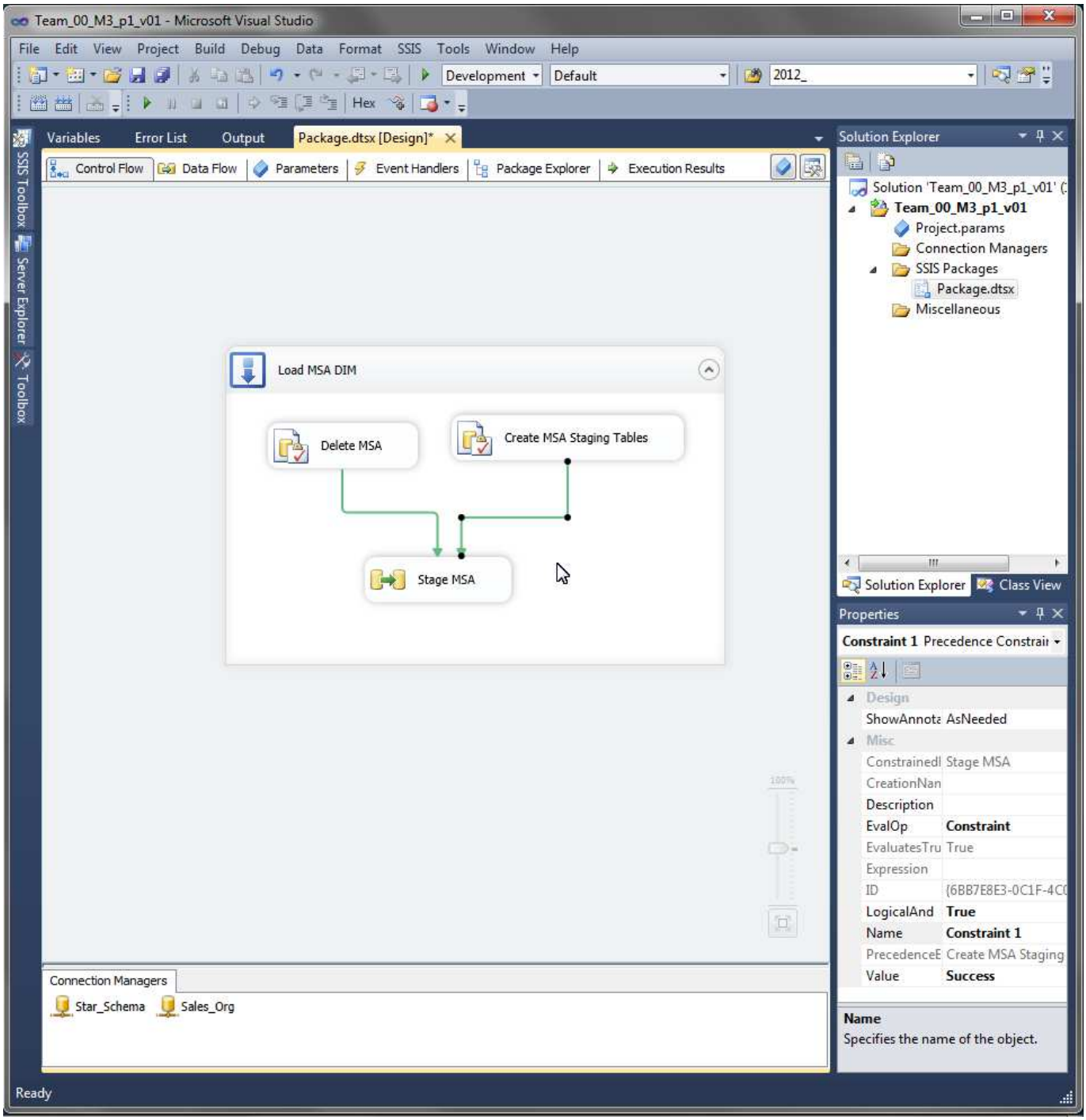

Figure 21 – Completed Precedence Constraints to Stage MSA Data Flow Task

F Designing a Data Flow Task

Now, double click the "Stage MSA" Data flow, or simply select the "Data Flow" tab at the top of the workspace, and make sure "Stage MSA" is selected in the drop-down box at the top after you open the tab. We are now ready to add a data source. From the left hand Toolbox, drag and drop the "OLE DB Source" into the Data Flow workspace. Rename the Data Source "Extract MSA" (see Figure 22).

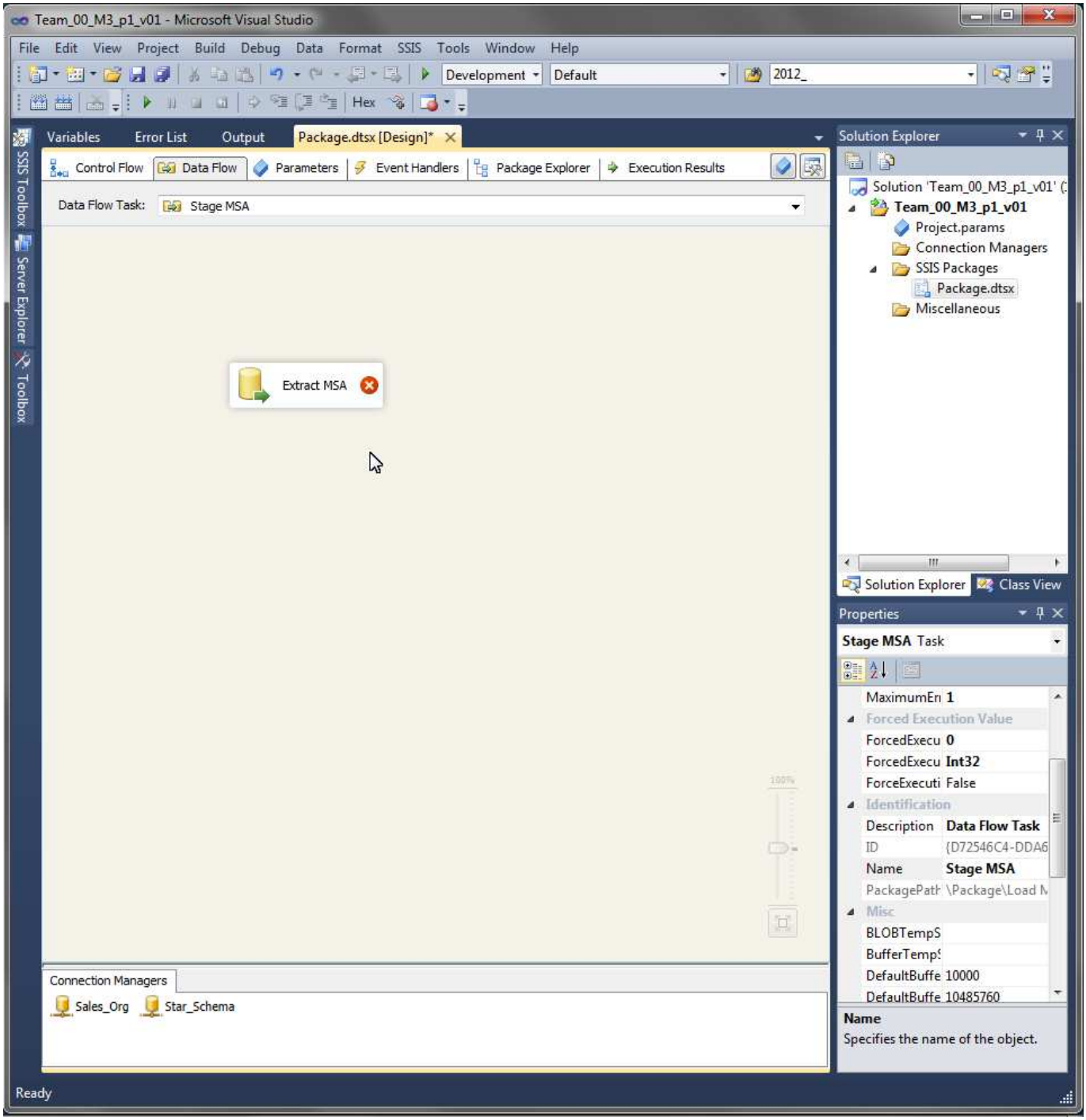

Figure 22 – A New OLE DB Data Source in the Data Flow Editor Workspace

Notice there is a little red **X** in the source – that is because we have some problem with the source (here we have not setup the connection yet, but this can happen if the construct being referenced by the source does not exist—recall that we can set the "DelayValidation" Property). Here, we need to setup the connection. Right-click the data source and select Edit, which will bring up the Editor (See Figure 23).

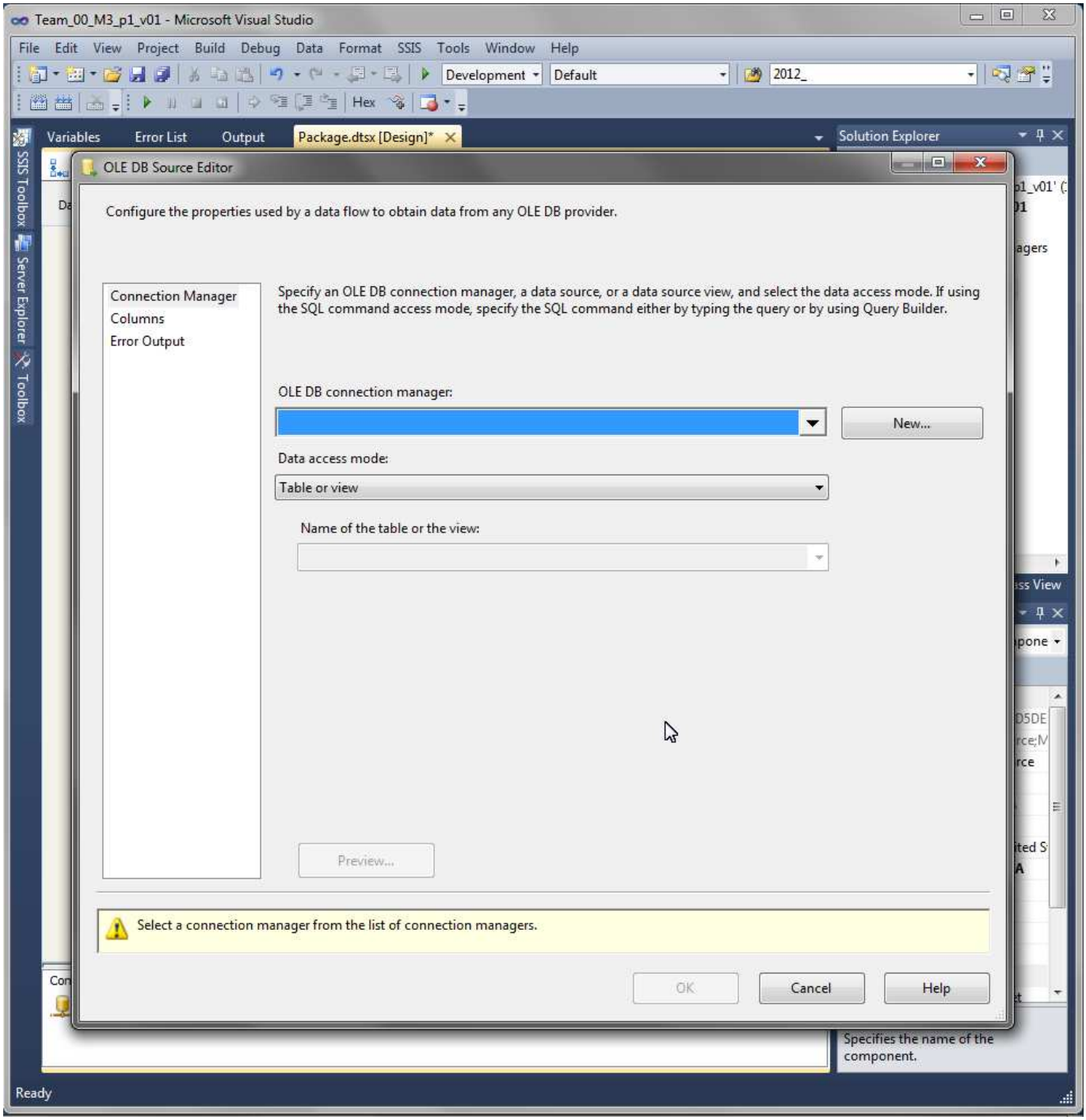

Figure 23 – OLE DB Source Editor

Next, we need to create a new connection manager for this Data Source by clicking the "New" button in this dialog. This will bring up another dialog—with the title "OLE DB Connection Manager" (as shown in Figure 24).

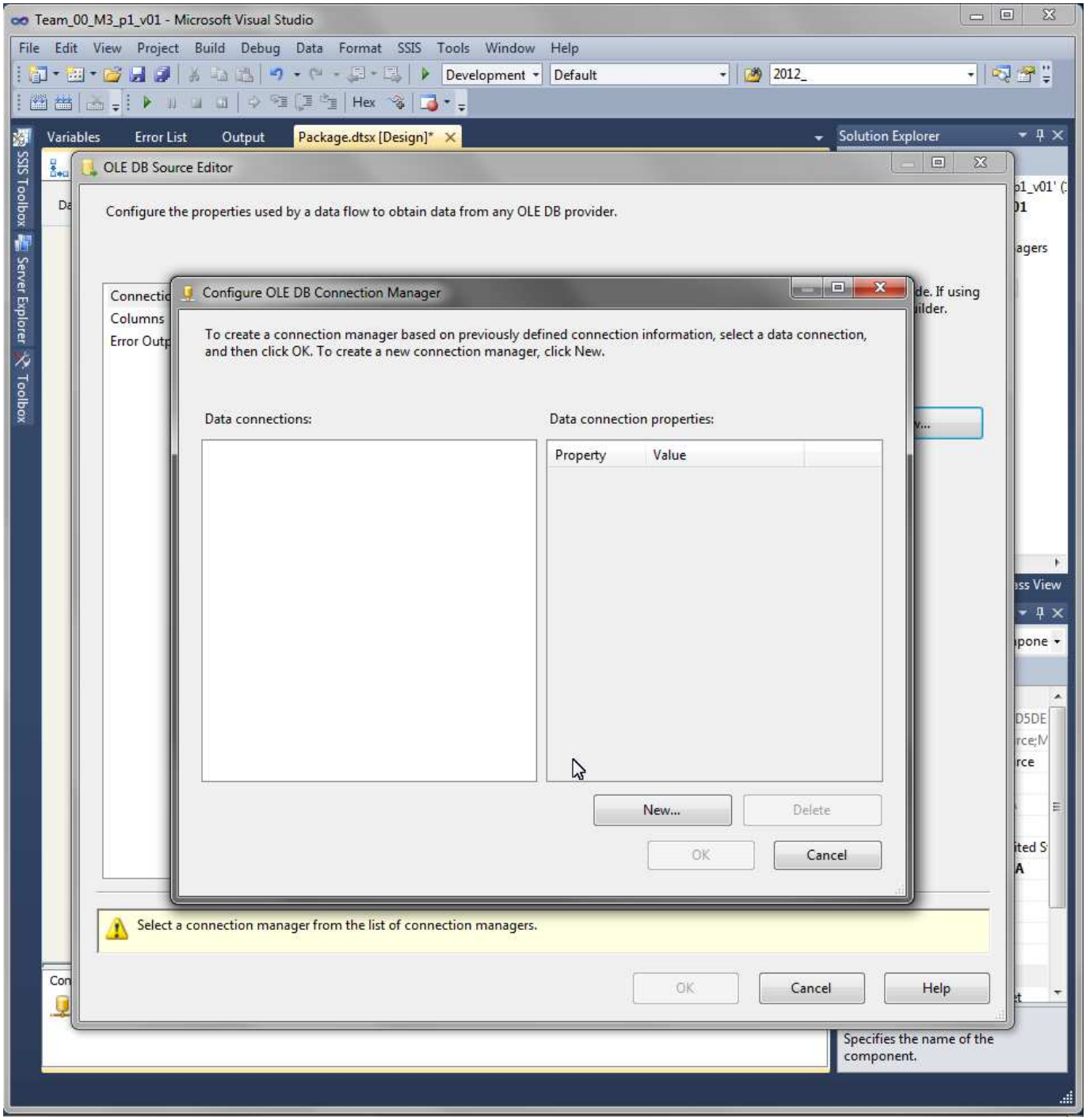

Figure 24 – OLE DB Connection Manager

Click the "New" button in this latest dialog, which will bring up yet another dialog, this one has the title "Connection Manager" (see Figure 25).

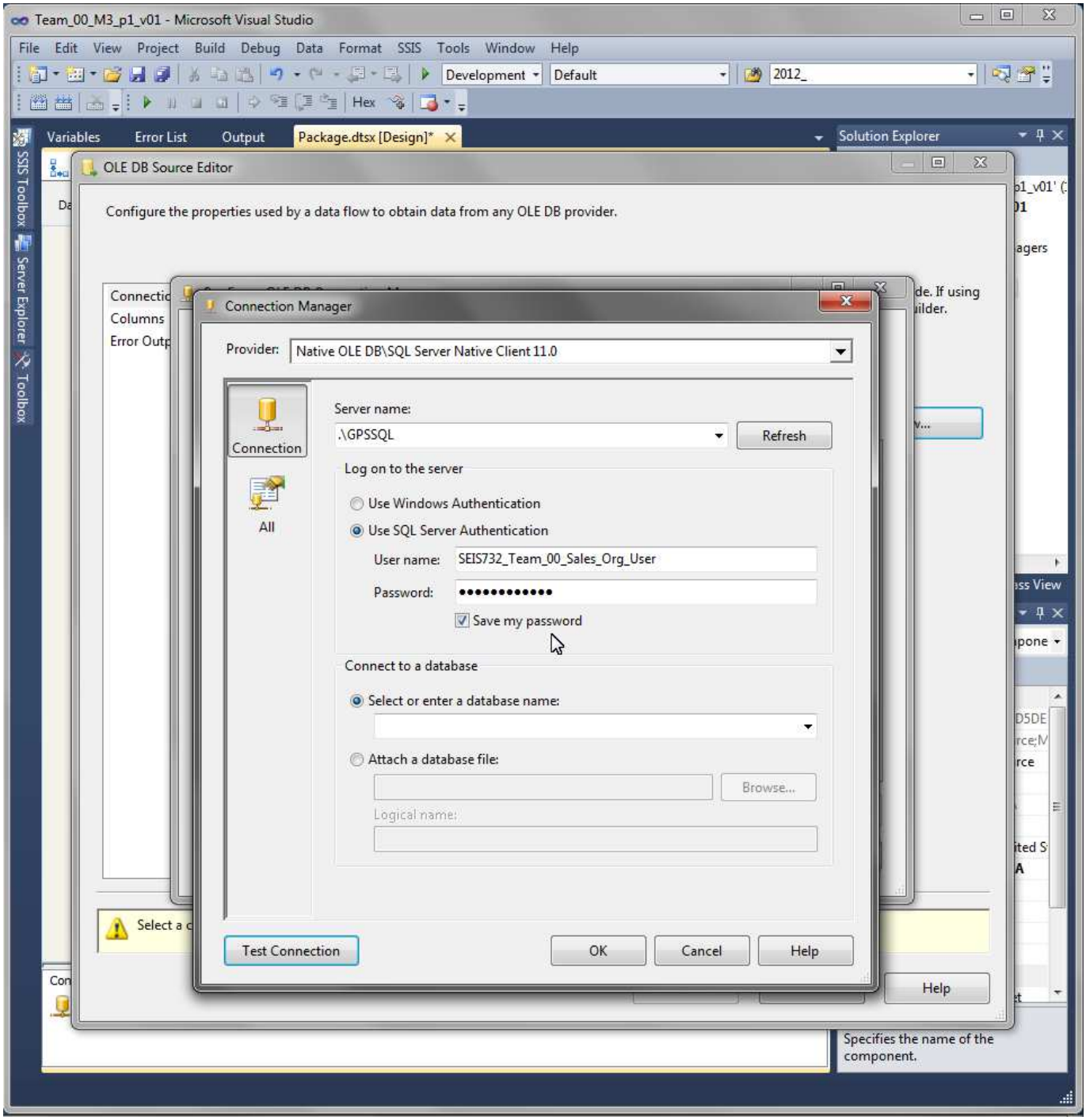

Figure 25 – Connection Manager Dialog

Now, then specify the connection details for the Sales Org Database. Make sure you check the "Save password" check box in the connection manager dialog box, and you should also click the "Test Connection" button before clicking Ok—just to make sure that the information you typed for the user name and password was correct. After that, you can click OK on the various dialogs we had opened—all the way back to the OLE DB Source Dialog (from Figure 23).

Now, you can select new connection in the "OLE DB connection manager" drop-down list.
In order to select the "[dbo].[Metropolitan_Statistical_Area]" in the "Name of the table or view" drop-down box. After you have selected it, click OK (see Figure 26).

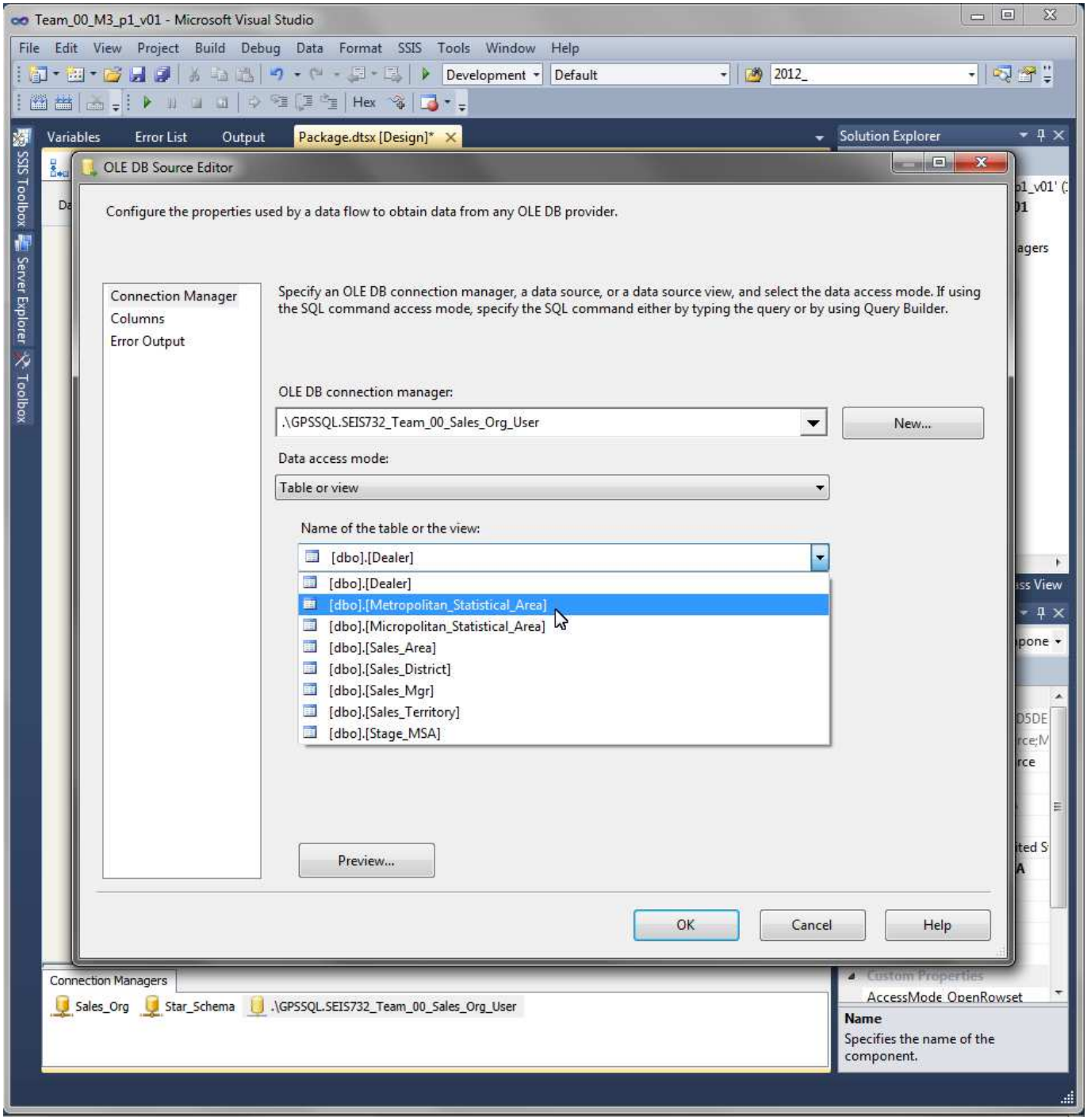

Figure 26 – Selecting the Connection Manager and Source Table in the OLE DB Source Editor

Next, drag an OLE DB Destination into the Data Flow Workspace. Rename it "Load MSA Stage", but do NOT edit the OLE DB Destination yet! If it is not connected it will not have the ability to automatically generate some of the details we need. Using the same technique as the precedence constraint, click on the "Extract MSA" Data Flow source and notice the red and blue arrows. Drag the blue arrow into the "Load MSA Stage" destination. This is a data path. The blue is the data output while the red is the "errors". We do not use the red in this example—and you do not need to use it in your M3.

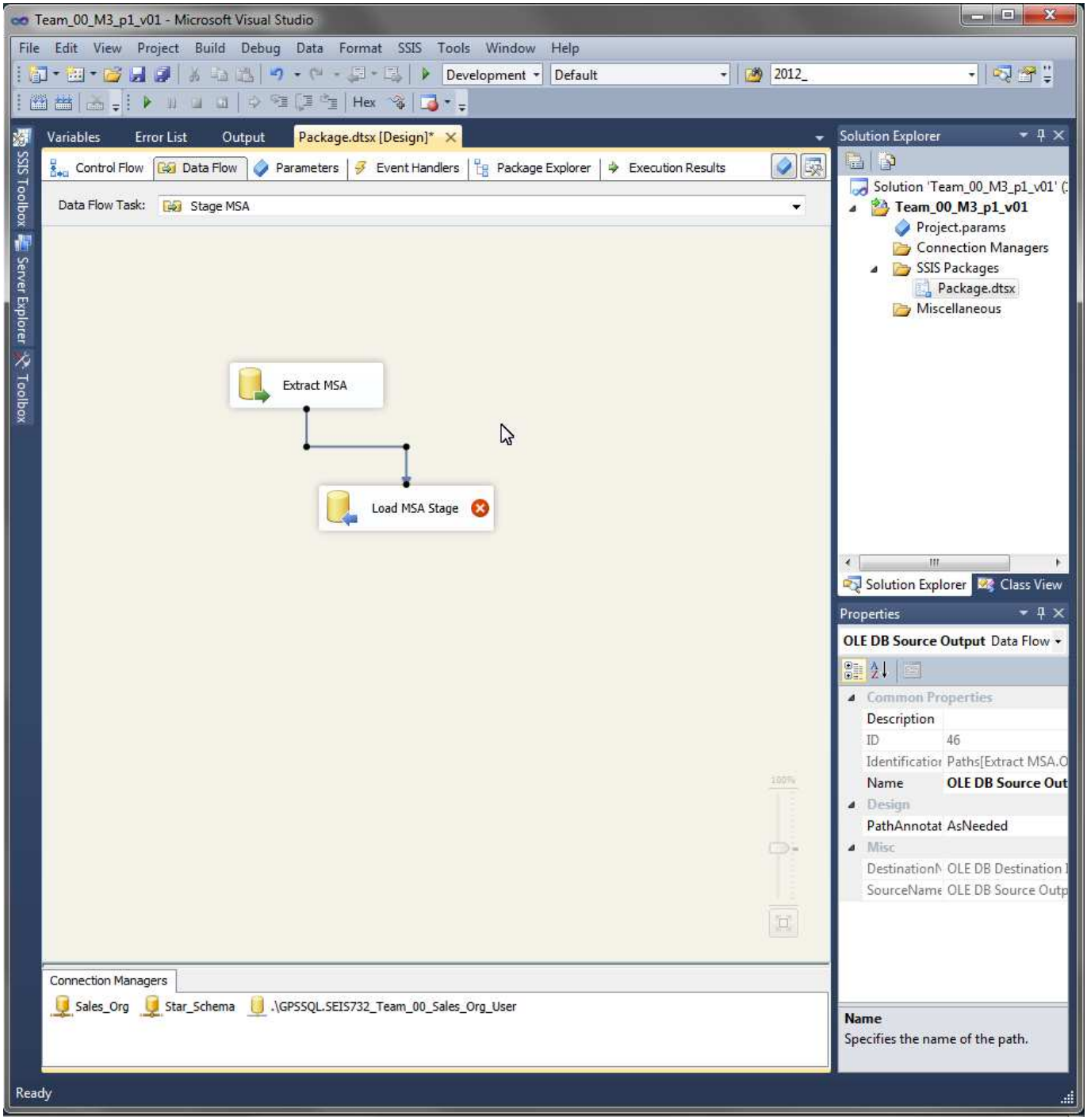

Figure 27 – A Data Path from the Extract MSA source to the Load MSA Stage Destination

Notice the red **X** and the tool-tip message about the Data Flow Destination—again this is expected because we have not set it up yet, which is what we will do next by editing the Data Flow Destination the same way we edited the Data Flow Source (double click or right click and select edit). This will open the editor which is shown in Figure 28.

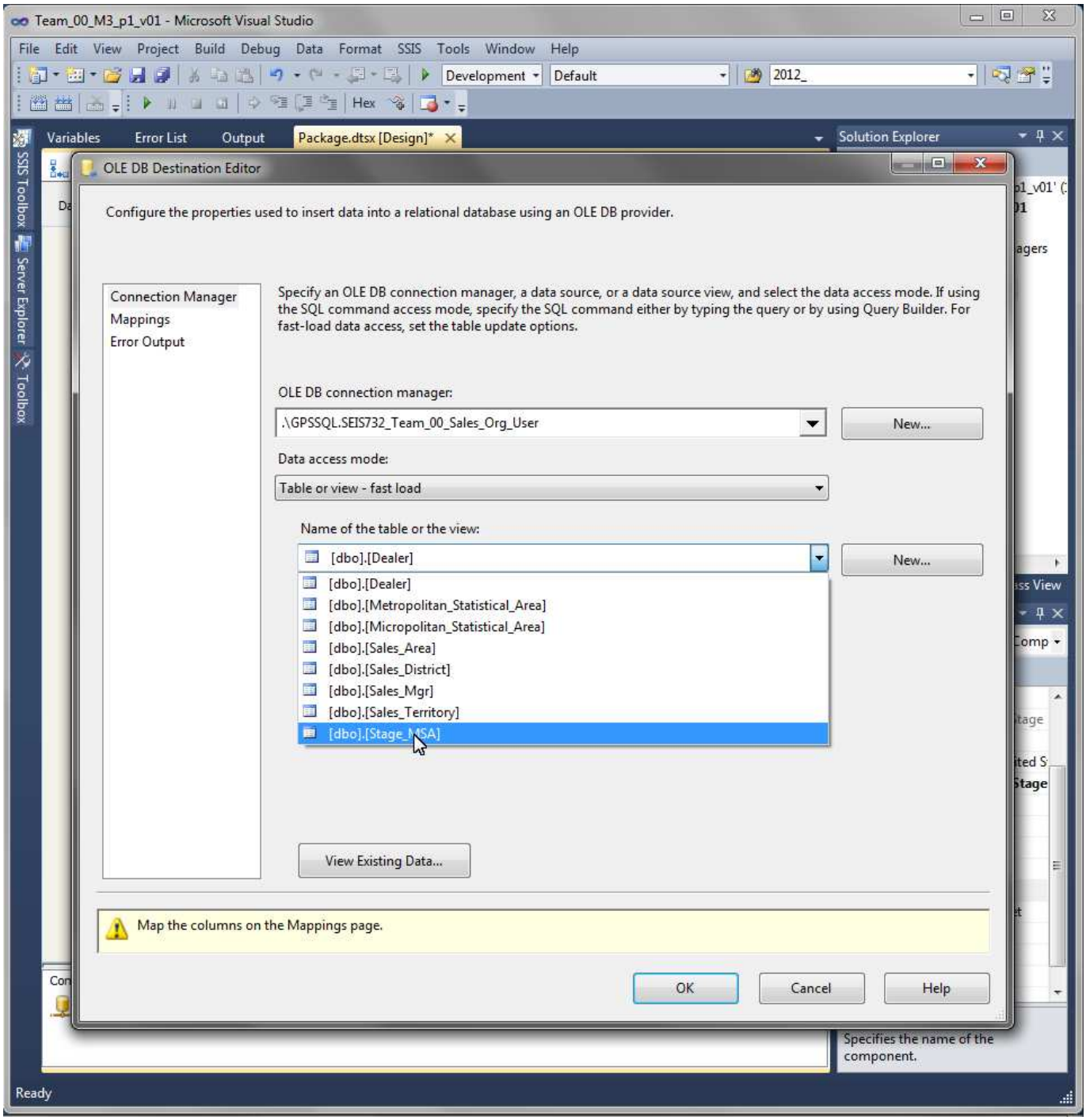

Figure 28 – OLE DB Destination Editor

Once again, we need to create or select a connection (again for the Sales_Org database). After you have done that, select the Data Access mode as either "Table or View – fast load" or "Table or View". And select the [dbo].[Stage_MSA] as the destination table.

The Fast load is only valid when the source and destination are on the same machine, so in the real world we would probably use the plain "table or view" most of the time, but for this milestone, you can use the fast one if you wish (it is the default when they are on the same machine).

Be very careful where you click on all of these dialogs!!!

For example, if you click ANYWHERE in the gray area "near" the check boxes it MIGHT SILENTLY check or uncheck one of the boxes

In other words, anywhere in between Keep Identity and Table Lock might will toggle one of these options!!!!

If the wrong options are selected here, it can be a source of errors and great frustration for your M3 Data Flow Tasks!!!!

Now click on the "Mappings" entry on the left hand side. This will bring up the mapping information (see Figure 29).

By default, the source name column is mapped to the destination column with the same name (which is why creating a view for the source is recommended even when the mappings are trivial). You can change these mappings by clicking on the names in either the Input Column list or the Destination Column list (these are actually drop-down boxes).

In this case, there are no columns with the same name and therefore NOTHING is mapped automatically for us. This is not a problem, we can manually select the columns by dragging the columns into each other in the upper half of the dialog, or by clicking on the drop-down lists in the lower part of the diagram. Here, we want to map "MSA_ID", "MSA_Name", and "MSA_Population" to the corresponding column named "MSA_Metropolitan_Area_ID", etc.

Notice that we ignore the MSA_Metropolitan_Area_Key column in the Destination Column list – skipping it in this list will effectively try to populate it with a NULL value but because this column is the surrogate key for the table, we created it using an identity column so that this will actually automatically generate a new sequential number for each row loaded (provided we did not accidentally click the wrong check box in the previous dialogs).

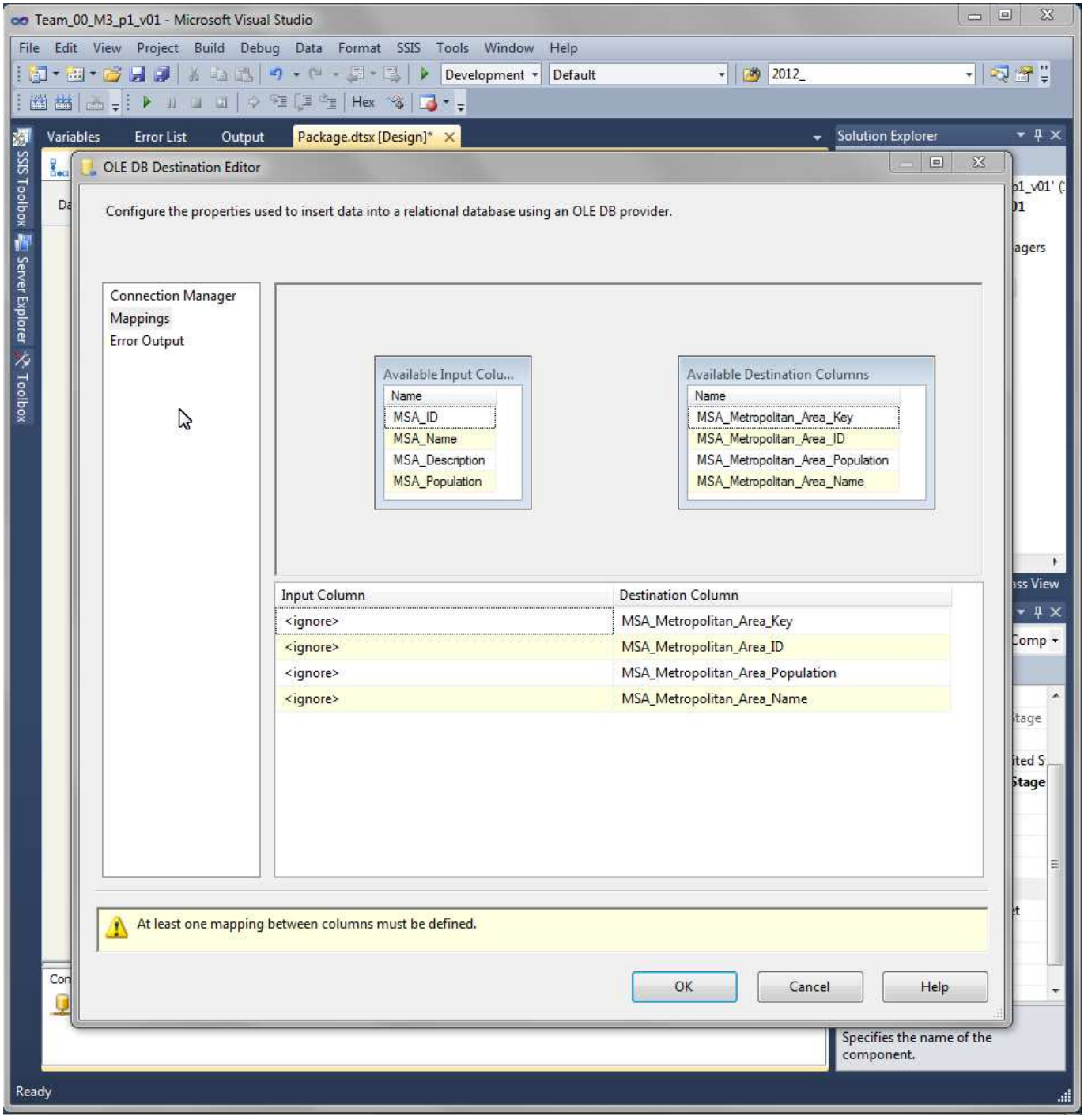

Figure 29 – Mappings in the Data Destination Dialog

As we fill in the mappings, the dialog will update the other part to reflect the changes. For example, if we drag a column on the top part, then the bottom part will change the corresponding drop box value. Contrariwise, if we change the drop-down box value in the bottom of the dialog, the dialog will redraw the lines for the corresponding mapping in the upper part of the dialog. See Figure 30 for the finished mappings.

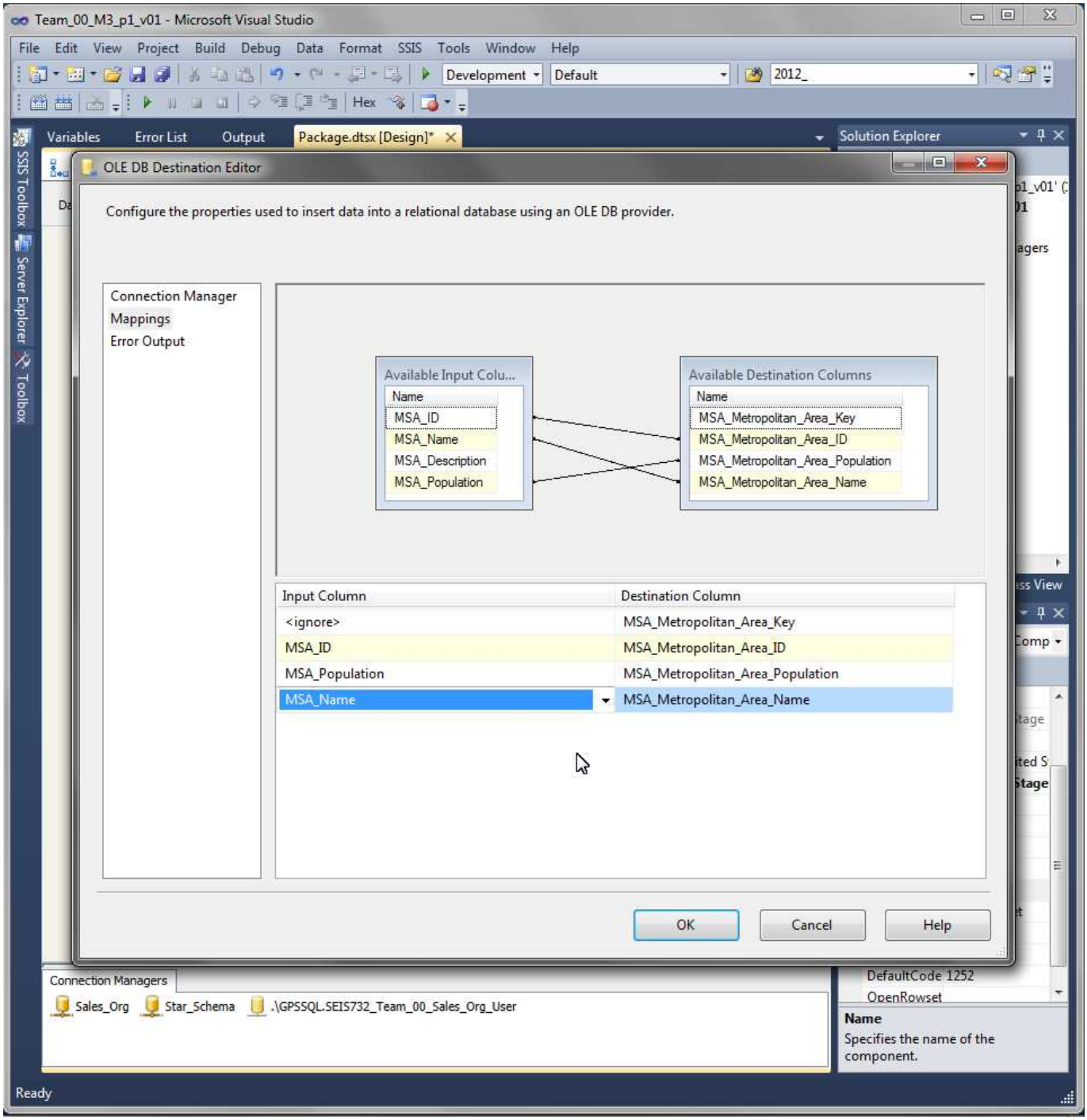

Figure 30 - The Finished Mappings

IV Final Thoughts

A What I did not show

Each of the tasks should have the delay validation property set.

We could create another T-SQL task ("Create MSA Staging View") to create a view that joins the Stage_MSA and the Micropolitan_Statistical_Area table using this text:

```
if (exists(select name from sysobjects where name = 'Extract_MSA' and type = 'V'))
  begin
    drop view Extract_MSA;
  end;
go

create view Extract_MSA
       as
select MICSA.MICSA_ID                       as MSA_Micropolitan_Area_ID,
       MICSA.MICSA_Description              as MSA_Micropolitan_Area_Description,
       MICSA.MICSA_Population               as MSA_Micropolitan_Area_Population,
       MSA.MSA_Metropolitan_Area_Key        as MSA_Metropolitan_Area_Key,
       MSA.MSA_Metropolitan_Area_ID         as MSA_Metropolitan_Area_ID,
       MSA.MSA_Metropolitan_Area_Population as MSA_Metropolitan_Area_Population,
       MSA.MSA_Metropolitan_Area_Name       as MSA_Metropolitan_Area_Name

from   Stage_MSA                             MSA,
       Micropolitan_Statistical_Area         MICSA
where
       MICSA.MSA_ID=MSA.MSA_Metropolitan_Area_ID;
go
```

Connect the "Stage_MSA" Data Flow task to this task to the "Create MSA Staging View" T-SQL task via a precedence constraint. Create another Data Flow task, name it "Load_MSA" and the T-SQL task to it via a precedence constraint. Inside the "Load_MSA", setup the source as the "Extract_MSA" view from the Sales_Org database and the destination as the actual "MSA" DIM Table in the star schema database. Follow the same technique we used before. Notice that this time, the columns will map AUTOMATICALLY to the correct source and target because of the naming we did inside the view!

Lastly, you are required to perform some minimal cross-footing. Using the Instance_Counts.txt file provided with your M3 data bundle, you can estimate what the counts should be for the members in each dimension you load. These numbers can be used to check the load by creating one or more T-SQL tasks and connecting them to the last task in the chain of loading tasks. For example, we can create a T-SQL task named Cross-Foot and connect it to the "Load_MSA" task and place the following text inside it:

```
declare @num_MICSA integer

select @num_MICSA=count(*) from MSA;

if (@num_MICSA<>414)
begin
	raiserror('MSA failed cross footing: ' ,18,1);
end
```

Obviously, this should run on the Star_Schema database. For your each DIM, the "MSA" should be replaced with the DIM table being cross-footed, and the "414" should be replaced with the appropriate count value based upon the OLTP data (and the Instance_Counts.txt file). You can also choose to be less "copy-and-paste" and rename the variables and error message appropriately, but that is up to you.

See Figures 31, 32, and 33 for pictures of what this should look like.

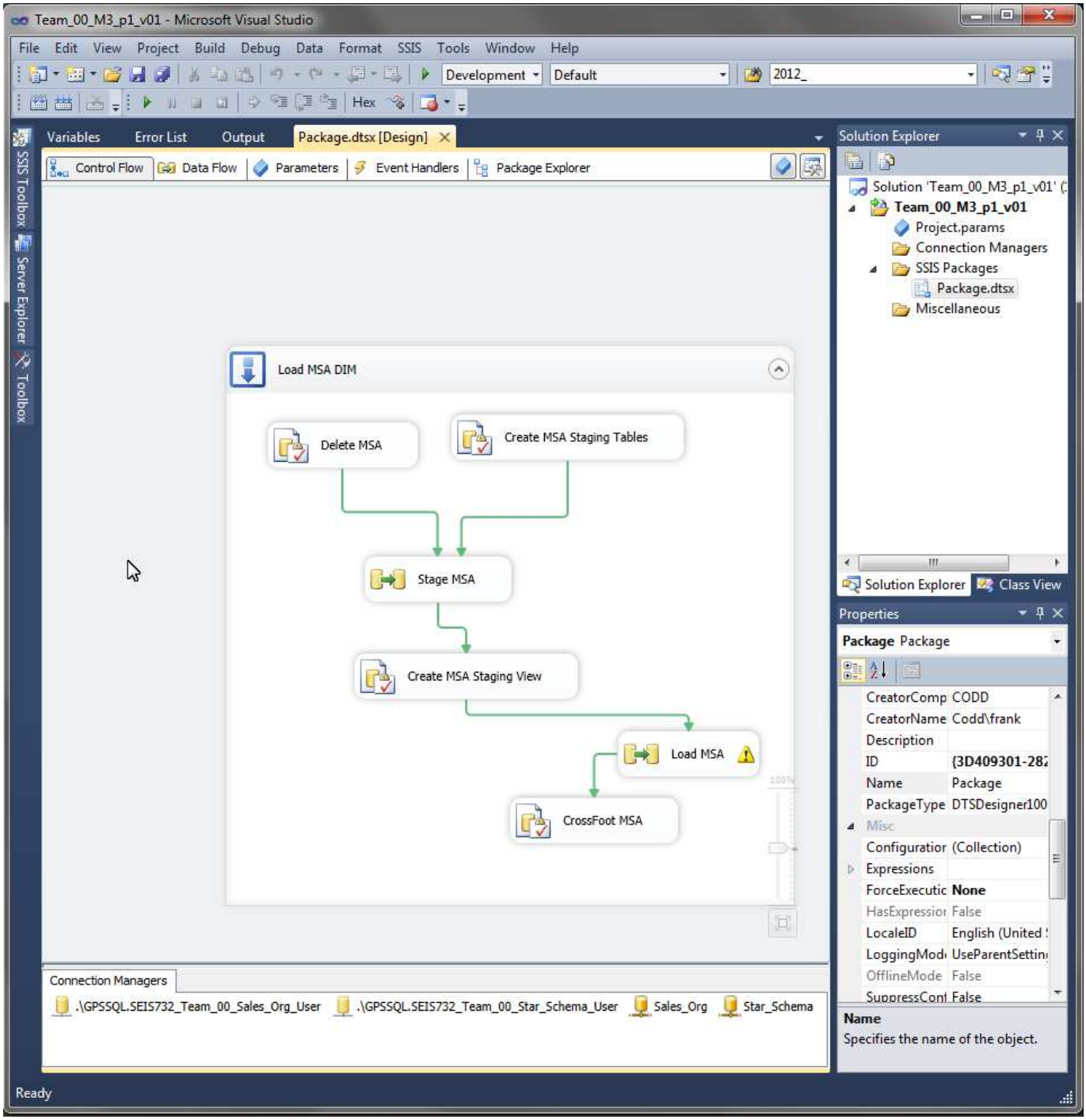

Figure 31 - The Final Set of tasks and Precedence Constraints

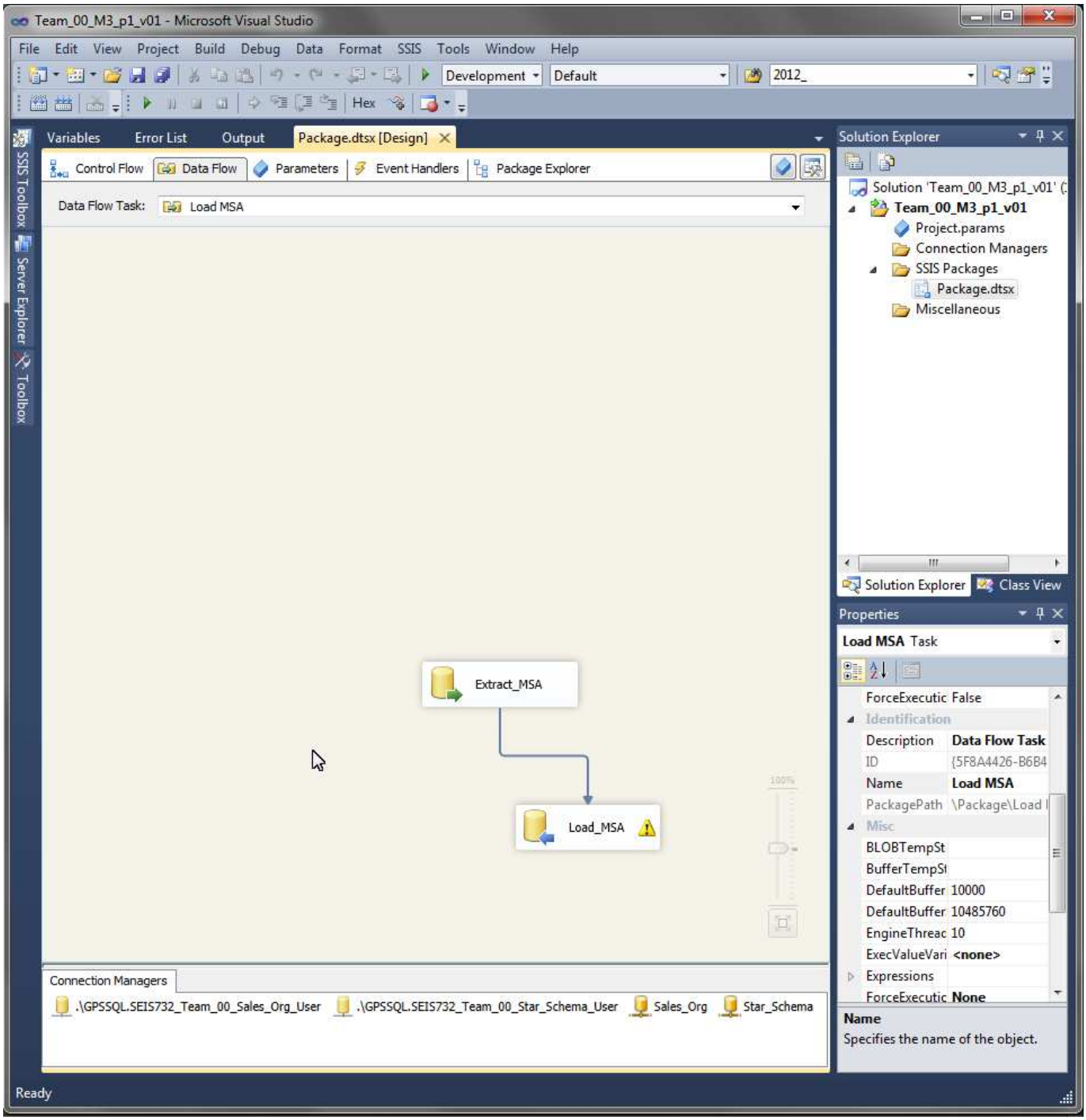

Figure 32 - The Load_MSA Data Flow Task

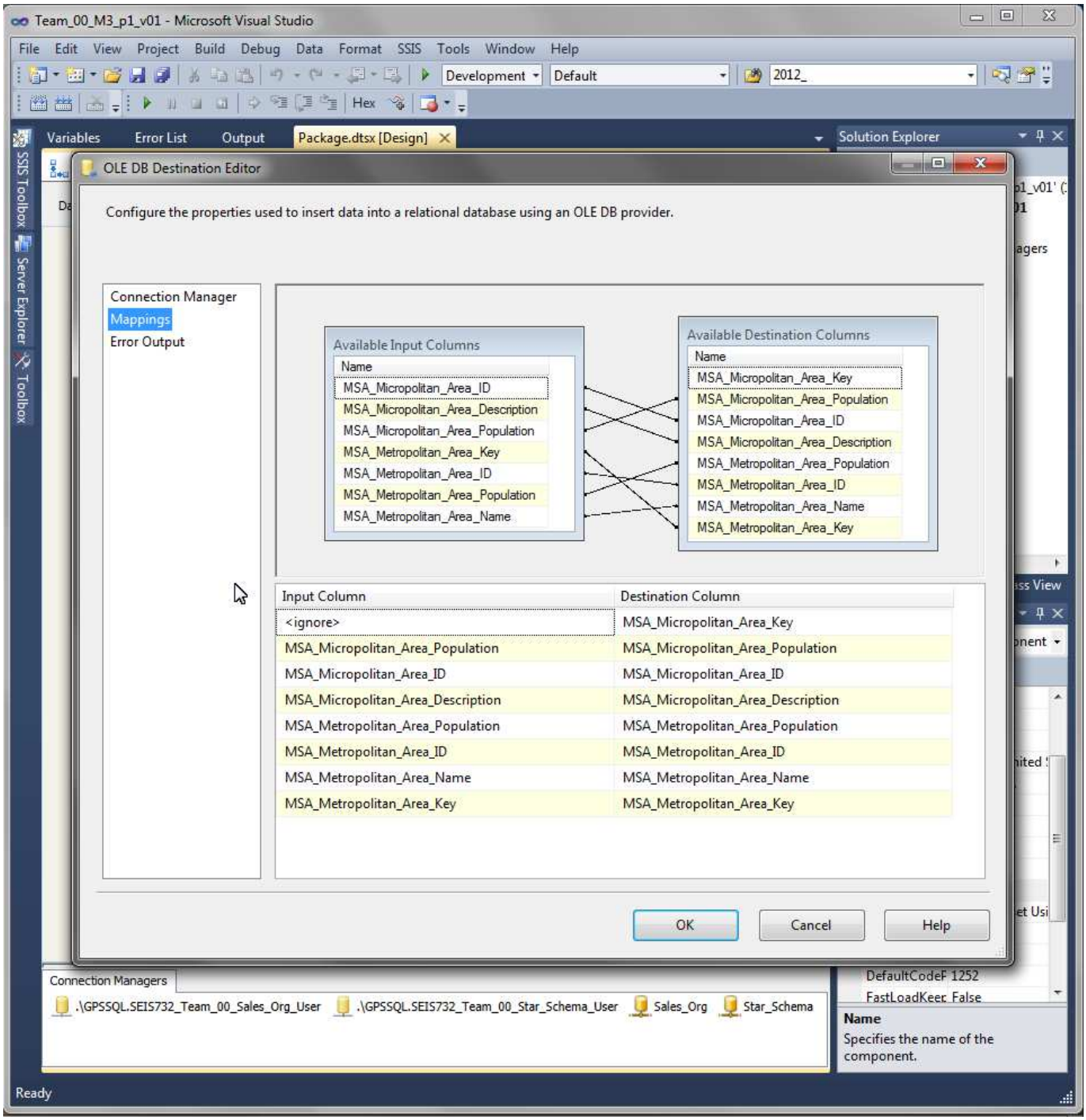

Figure 33 - The Mappings Inside the Load_MSA Data Flow Task

B Other Tips

You can either create separate "DELETE" T-SQL tasks for each dimension table or you could create a single task for all of them. You can create a separate "create view or stored procedure" T-SQL task for each view / stored proc that you have or you can group them together by the database connection used to create them or the DIM being loaded. In general, I recommend using separate tasks grouped within sequence containers for each DIM because you can be more parallel in development and execution.

You can also use a sequence container to hold other containers. You simply drop the new container inside the container and connect the precedence constraints to the container task instead of the individual (contained) tasks.

You can also execute DML in the execute SQL tasks if necessary. You can use more complex transformations. You can use other database technologies if it is appropriate to what you are trying to do.

You can create staging tables and use multiple Data Flows to build up the integrated view of the data—you can also use update statements in the T-SQL tasks to help achieve this integrated data. Sometimes, it may be useful to essentially copy a table (create a staging table with all the desired columns in the star schema database and then load it from one or more production tables as a first step.) You can also use SQL functions (datepart, datename, etc.) as part of this integration process if it helps.

C Some final caveats

Be careful not to deadlock or live-lock yourself.

Be sure to test the solution against a completely "blank" machine before turning it in. In other words, (if possible go to a NEW MACHINE) unzip the CD Bundle, run the MAKEALL command script then unzip your BIDS Project open it and run it against this mew machine. If it fails, or if you need to create a table or a view, etc. then something is MISSING from your M3 and points will be LOST! Do EVERYTHING YOU NEED TO HAVE DONE INSIDE YOUR FINAL PACKAGE!

CHECK YOUR COUNTS… if you "select count(*)" from the DIM tables and/or use select distinct for different DATTS do they match what you would expect? Use the Instance_Counts.txt file in the CD Bundle as a starting point and then use common sense and your understanding of DIM and HIER… For example, if we have less than 10 years of Dates loaded there better not be more than 3,650 rows in the DATE DIM!