

# Elaena Bakman

---

Email: [elaena.bakman@gmail.com](mailto:elaena.bakman@gmail.com)

Twitter: [@Elaena\\_Bakman](https://twitter.com/Elaena_Bakman)

GitHub: <https://github.com/Lenka72/The-Magic-of-SSISDB-Demo>



## The Magic of SSISDB

An abstract graphic consisting of several overlapping, curved, teal-colored shapes that sweep across the left side of the image, creating a sense of movement and depth.

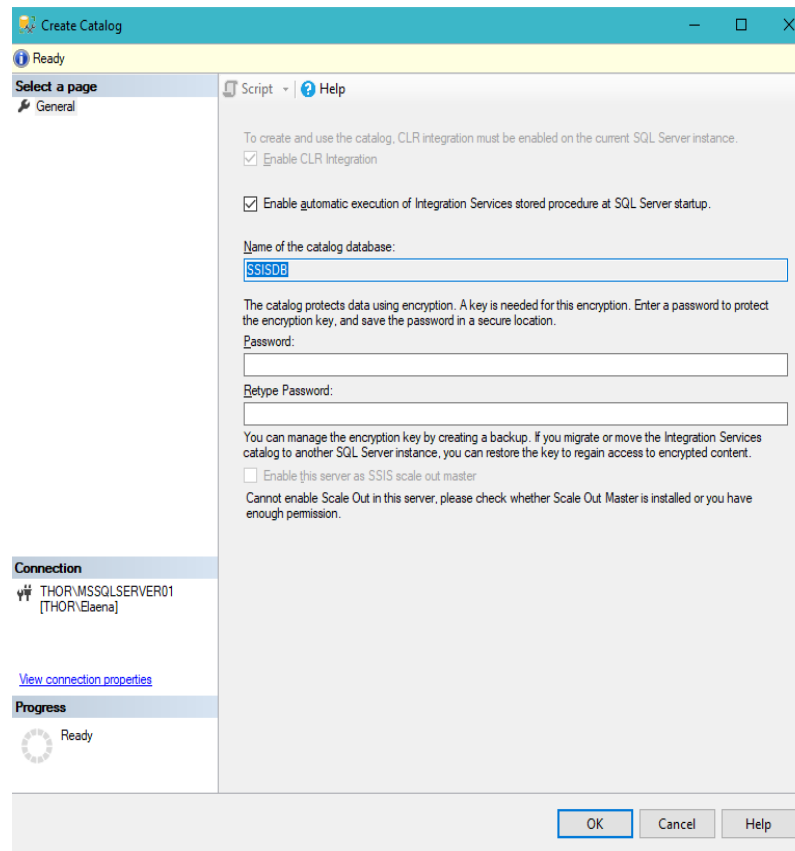
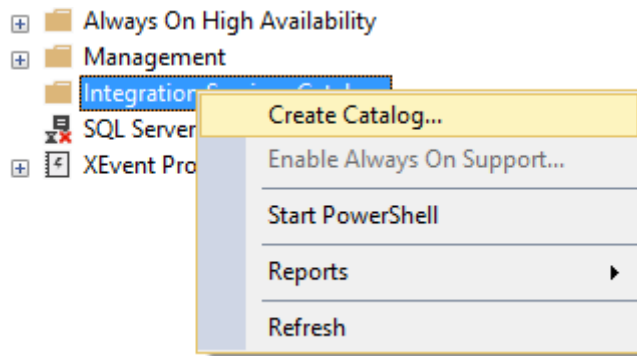
# Creative use of Environment Variables and More

# BACKSTORY



# Integration Services Catalogs

To set up a new Integration Services Catalog...



# SSISDB

SSISDB is the central point for working with the Integration Services Catalogs. There are two schemas in SSISDB:

- Catalog
- Internal

```
SQLQuery1.sql - TH...(THOR\Elaena (61))* -p X
USE SSISDB
GO

SELECT TABLE_SCHEMA
, TABLE_TYPE
FROM INFORMATION_SCHEMA.TABLES
GROUP BY TABLE_SCHEMA
, TABLE_TYPE
ORDER BY TABLE_SCHEMA;
```










100 %

Results Messages

|   | TABLE_SCHEMA | TABLE_TYPE |
|---|--------------|------------|
| 1 | catalog      | VIEW       |
| 2 | internal     | BASE TABLE |
| 3 | internal     | VIEW       |



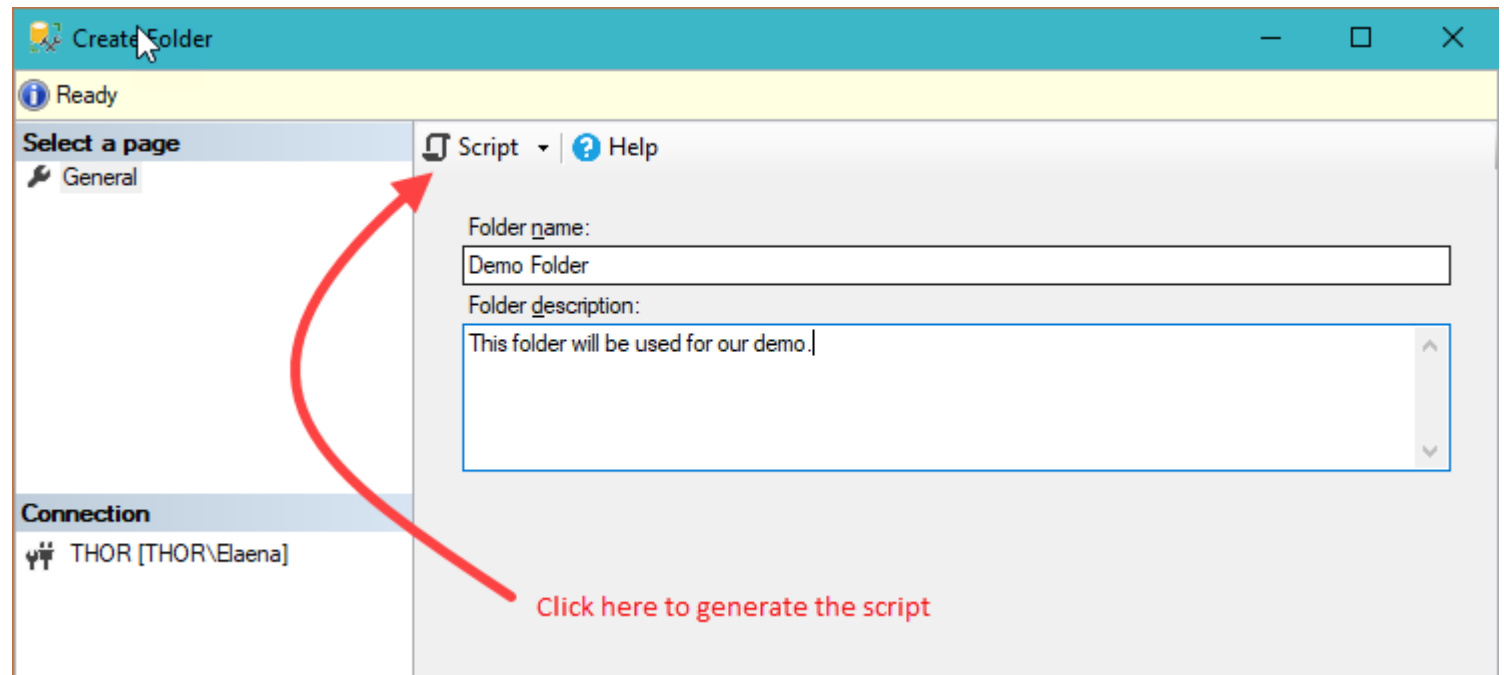
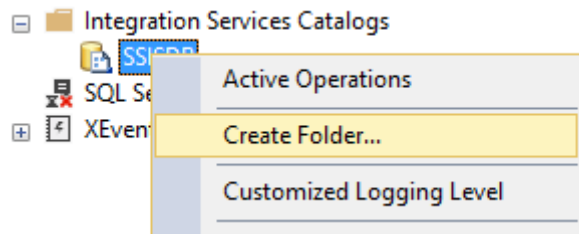
# Catalog Object Identifiers

- [-]  Integration Services Catalogs
  - [-]  SSISDB
    - [-]  Demo Folder
      - [-]  Projects
        - [-]  Demo
          - [-]  Packages
            -  Vitamins.dtsx
    - [-]  Environments
      -  Demo Environment



# Create a Folder (Option 1)

Using Management Studio:



# Create a Folder Scripting

```
DECLARE @folder_name NVARCHAR(128)
        ,@folder_id BIGINT;

SET @folder_name = N'Demo Folder';

IF NOT EXISTS (SELECT 1 FROM catalog.folders WHERE name = @folder_name)
BEGIN
    EXEC SSISDB.catalog.create_folder @folder_name = @folder_name
        ,@folder_id = @folder_id OUTPUT;

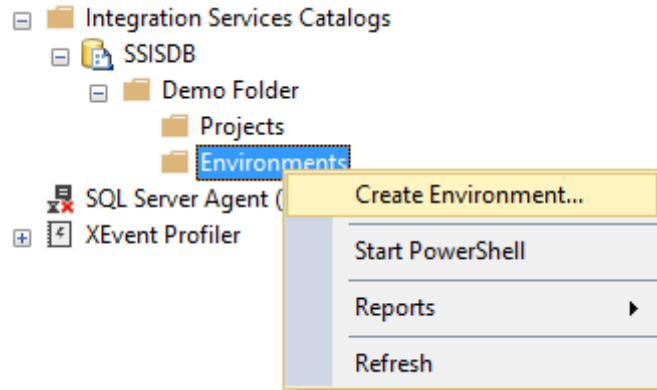
    SELECT @folder_id;

    EXEC SSISDB.catalog.set_folder_description @folder_name = @folder_name
        ,@folder_description = N'This folder will hold the paramters for things you may want to use.';
END;
GO
```





# Create an Environment Using SQL MS



The 'Create Environment' dialog box is shown. It has a title bar with the text 'Create Environment'. Below the title bar is a yellow status bar with an information icon and the text 'Ready'. The main area is divided into two panes. The left pane, titled 'Select a page', contains a 'General' tab. The right pane, titled 'Script' and 'Help', contains the following fields:

- Environment name:** A text box containing 'Demo Environment'.
- Environment description:** A text area containing 'This will be the environment we will be using for our demo.'

At the bottom of the dialog, there is a 'Connection' section with a dropdown menu showing 'THOR [THOR\Elaena]'.



# Create an Environment Using Scripting

```
DECLARE @folder_name NVARCHAR(128)
        ,@environment_name sysname;

SET @folder_name = N'Demo Folder';
SET @environment_name = N'Demo Environment';

IF NOT EXISTS
(SELECT      1
 FROM        catalog.folders F
 INNER JOIN  catalog.environments E
 ON E.folder_id = F.folder_id
 WHERE      F.name = @folder_name
           AND E.name = @environment_name)
BEGIN
    IF EXISTS (SELECT 1 FROM catalog.folders F WHERE F.name = @folder_name)
    BEGIN
        EXEC SSISDB.catalog.create_environment @environment_name = @environment_name
                                                ,@environment_description = N'This will be the environment we will be using for our demo.'
                                                ,@folder_name = @folder_name;
    END;
END;
GO
```

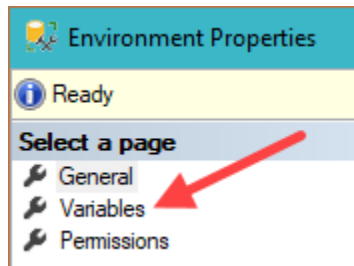


# Add a Variable Using MS

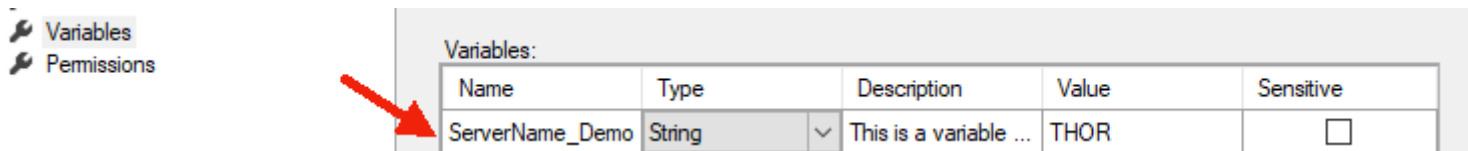
1. Double-click on the environment name:



2. In the Environment Properties window click on "Variables" under "Select a page..."



3. Add the Name, Type, Description and Value for your variable:



# Add a Variable Using Scripting

```
DECLARE @folder_name NVARCHAR(128)
        ,@environment_name sysname
        ,@variable_name NVARCHAR(128)
        ,@var SQL_VARIANT;

SET @folder_name = N'Demo Folder';
SET @environment_name = N'Demo Environment';
SET @variable_name = N'ServerName_Damo';
SET @var = N'THOR';

IF NOT EXISTS
(SELECT      1
 FROM        catalog.folders F
 INNER JOIN  catalog.environments E
 ON E.folder_id = F.folder_id
 INNER JOIN  catalog.environment_variables EV
 ON EV.environment_id = E.environment_id
 WHERE      F.name = @folder_name
           AND E.name = @environment_name)
BEGIN
    EXEC SSISDB.catalog.create_environment_variable @variable_name = @variable_name
                                                ,@sensitive = False
                                                ,@description = N'This is a variable we are going to use for our demo.'
                                                ,@environment_name = @environment_name
                                                ,@folder_name = @folder_name
                                                ,@value = @var
                                                ,@data_type = N'String';
END;
GO
```



# Deploying an ISPAC

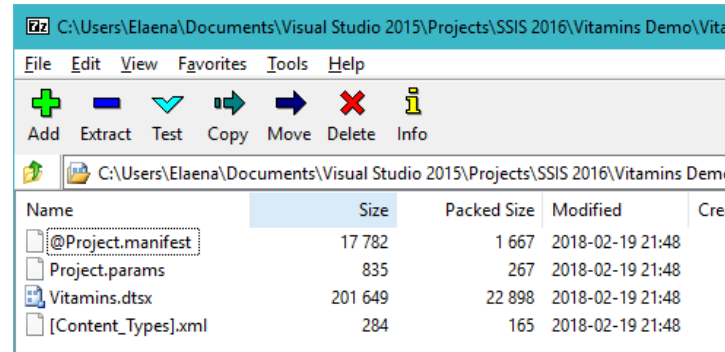
1. Deploy the ISPAC using the catalog.deploy\_project Stored Procedure in SSISDB.

Expected Parameters:

- @folder\_name - The name of the folder where the project is deployed.
- @project\_name - The name of the new or updated project in the folder.
- @projectstream - The binary contents of an Integration Services project deployment file (.ispac extension).
- @operation\_id - Returns the unique identifier for the deployment operation.

2. ISPAC is a ZIP file. You can use 7 zip to view the content. You can download it from:

<http://www.7-zip.org/>




3. The ISPAC holds all packages in the Project. Removing one from the project would remove it from the project upon deployment.



# Deploying an ISPAC

When pulling the value of the @project\_stream variable make sure the location is “visible” to the server:

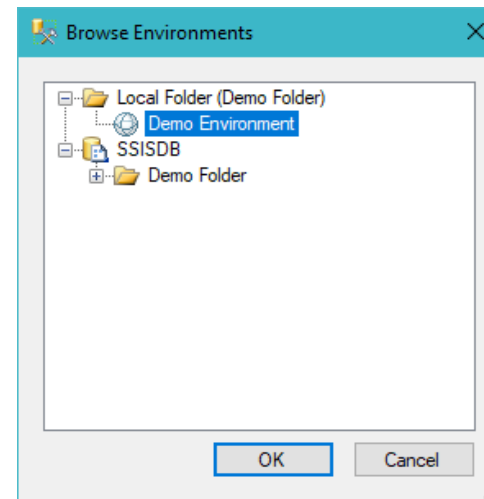
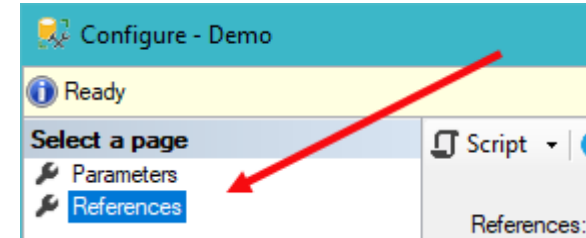
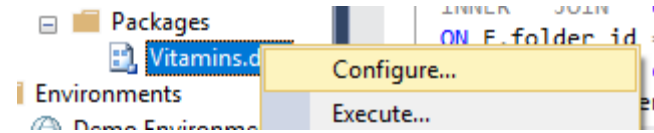
```
SET @ProjectBinary =  
(SELECT *  
FROM  
    OPENROWSET(BULK  
        'C:\Users\Elaena\Documents\Visual Studio 2015\Projects\SSIS 2016\Vitamins Demo\Vitamins Demo\bin\Development\Demo.ispac'  
        ,SINGLE_BLOB)  
    AS BinaryData );  
  
EXEC catalog.deploy_project @folder_name = @FolderName  
    ,@project_name = @ProjectName  
    ,@project_stream = @ProjectBinary  
    ,@operation_id = @OperationId OUT;  
  
GO
```



# Adding an Environment Reference

## Using Management Studio:

1. Right-click on the package name:
2. Click on References:
3. Click the Add button and add the reference:



# Adding an Environment Reference

## Scripting:

```
EXEC catalog.create_environment_reference @environment_name = @environment_name  
                                           ,@reference_id = @reference_id OUTPUT  
                                           ,@project_name = @project_name  
                                           ,@folder_name = @folder_name  
                                           ,@reference_type = R;
```



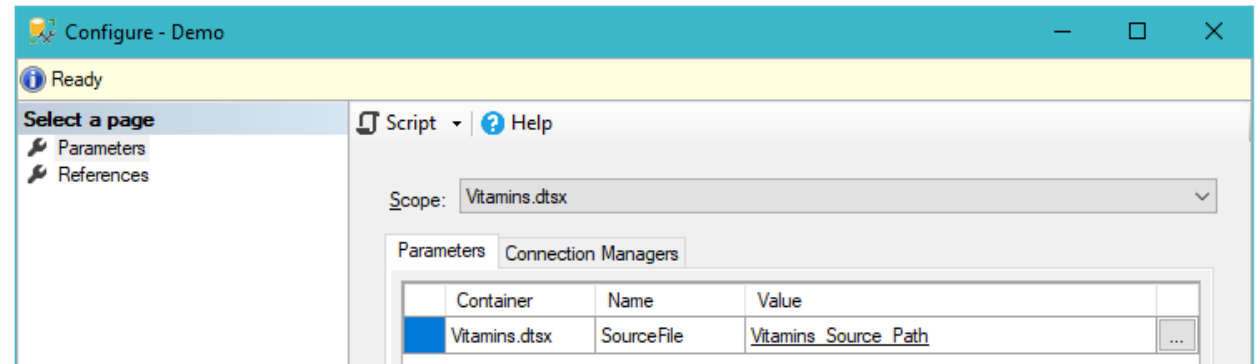


# Set Object Parameter Reference Using MS

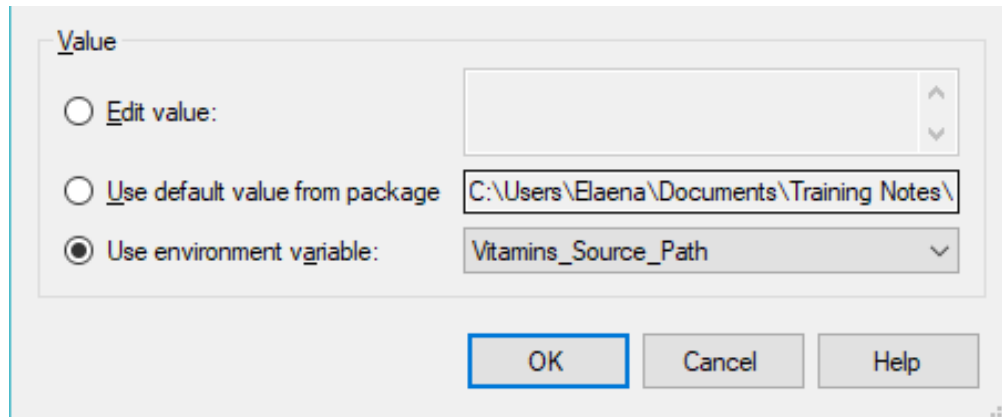
1. Right-click on a package and select the Configure option:



2. In the configure window click on the "..." to configure the object parameters:



3. In the Set Parameter Value view, select the "Use environment variable" option:



# Set Object Parameter Reference Using Scripting

Create a reference between the package or project parameters and the environment variables.

```
EXEC SSISDB.catalog.set_object_parameter_value @object_type = 20
, @parameter_name = N'SQL_Server_Name'
, @object_name = N'Demo'
, @folder_name = N'Demo Folder'
, @project_name = N'Demo'
, @value_type = R
, @parameter_value = N'Vitamins_SQL_Server_Name';
```

GO



# Use Case for Getting the Value of a Variable

Example 1: set a different directory for the source file based on your environment (Development, Test and Production).

```
,@BaseDirectory VARCHAR(256);

SET @ReportDate = EOMONTH(GETDATE(), -1);
SET @DateKey = CONVERT(CHAR(8), @ReportDate, 112);
SET @UserId = dbo.fn_get_process_started_by_userid('Valuation - EBO Anchor Values Set Population');

SET @BaseDirectory = dbo.fn_get_environment_parameter_value('Valuation', 'Valuation', 'BaseDirectory_WLA');;

SET @SourceFile = @BaseDirectory + '\' + CAST(@DateKey AS CHAR(8)) + ' - Monthend\Database\Initial Values Import\S&D Va:

DECLARE @SQL NVARCHAR(MAX);

SET @SQL = '      BULK INSERT #SourcePopulation
FROM          ''' + @SourceFile + '''
WITH
(
    FIRSTROW = 2,
    ROWTERMINATOR = ''\n'',
    FIELDTERMINATOR = ',',''
);

';

EXECUTE (@SQL);
```



# Use Case for Getting the Value of a Variable

Example 2: checking to see if the process is running on the production server and choosing a different action based on that information.

1. A function is used to pull the name of the production server:

```
SET @ProductionServerName = dbo.fn_get_environment_parameter_value('Valuation', 'Valuation', 'ProductionServerName');
```

2. Check the name of the production server against the @@SERVERNAME value. In this case we add a CC on the email to notify the user, the dev team in case of a failure in the production environment only:

```
IF @Status = 'Failed'
BEGIN
    SET @EmailBody = @EmailBody + 'Please try again or contact your developer to address the issue.' + @CrLf;
    SET @EmailBody = @EmailBody + ISNULL(dbo.fn_get_job_failed_error_message(@ProcessName) + @CrLf, '');
    -- add the ProcessId to the failure notification so that it can be tied to a redline ticket if one needs to be created
    SET @EmailBody = @EmailBody + @CrLf + ISNULL('Process Id: ' + LTRIM(CAST(@ProcessId AS VARCHAR(6))) + @CrLf, '');

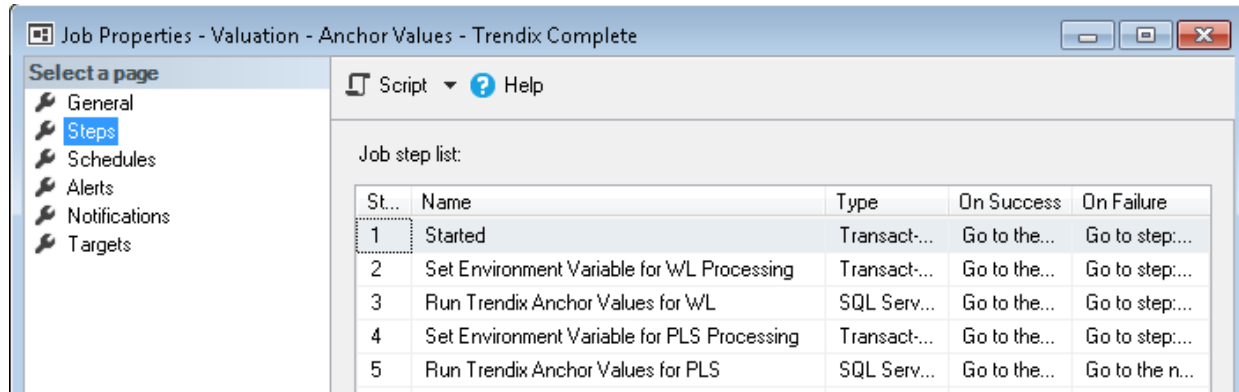
    -- only sent notifications to the valuationsupport@pnmac.com and marcus.vazquez@pnmac.com team when the job fails in production
    IF @@SERVERNAME = @ProductionServerName
        SET @CopyEmail = 'valuationsupport@[REDACTED].com;';
END;

SET @EmailBody = @EmailBody + @CrLf + 'Thanks,' + @CrLf + 'Development Team';
EXEC msdb.dbo.sp_send_dbmail @recipients = @Email, @copy_recipients = @CopyEmail, @subject = @EmailSubject, @body = @EmailBody;
```

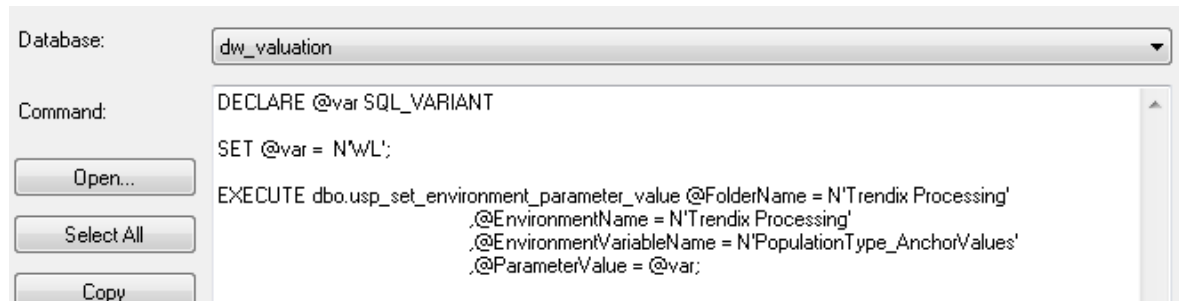


# Use Case for Setting the Value of a Variable

This job runs the same SSIS Package twice but processing two different sets of records by first setting the environment variable to one value, and then to another value (step 2 and 4 set the values):

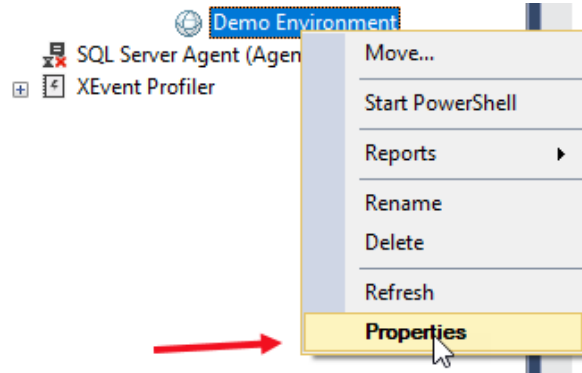


Setting the value:

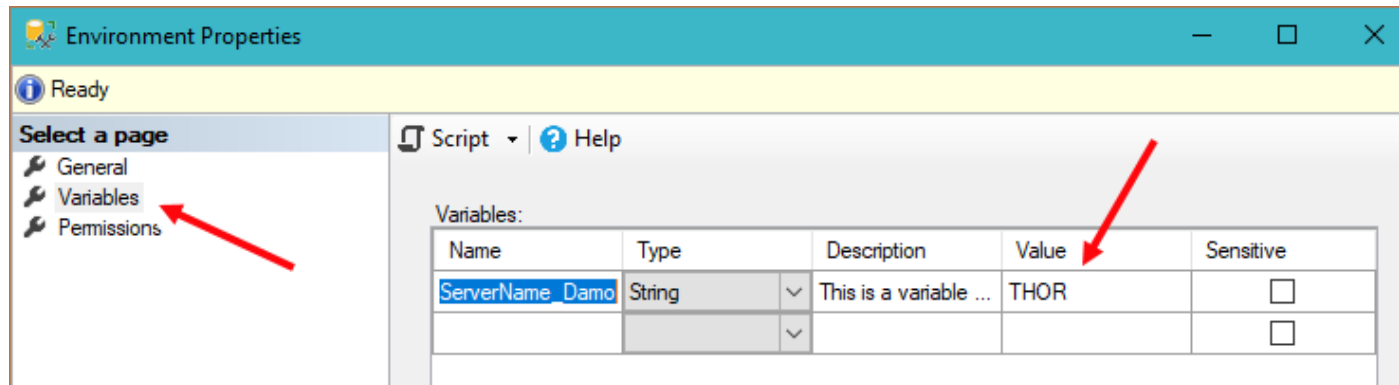


# Set a Variable Value Using MS

1. Right-click on the environment name and go to Properties:




2. Go to the Variables tab and change the value:



# Set a Variable Value Using Scripting

Using the catalog.set\_environment\_variable\_value Stored Procedure in the SSISDB database:

```
CREATE PROCEDURE dbo.usp_set_environment_variable_value (  
    @FolderName NVARCHAR(128)  
    ,@EnvironmentName NVARCHAR(128)  
    ,@EnvironmentVariableName NVARCHAR(128)  
    ,@ParameterValue SQL_VARIANT)  
  
AS  
BEGIN  
    SET NOCOUNT ON;  
  
    EXEC SSISDB.catalog.set_environment_variable_value @folder_name = @FolderName  
    ,@environment_name = @EnvironmentName  
    ,@variable_name = @EnvironmentVariableName  
    ,@value = @ParameterValue;  
  
END;  
GO
```



# Retrieving the Value of a Variable

Create a function to retrieve the value of a variable

```
CREATE FUNCTION dbo.fn_get_environment_variable_value (  
    -- Add the parameters for the function here  
    @FolderName NVARCHAR(128)  
    ,@EnvironmentName NVARCHAR(128)  
    ,@EnvironmentVariableName NVARCHAR(128))  
RETURNS VARCHAR(4000)  
AS  
BEGIN  
    -- Declare the return variable here  
    DECLARE @ParameterValue VARCHAR(4000);  
  
    -- Add the T-SQL statements to compute the return value here  
    SELECT @ParameterValue =  
    (SELECT CONVERT(VARCHAR(256), EV.value) AS BaseDirectory  
    FROM SSISDB.catalog.folders F  
    INNER JOIN SSISDB.catalog.environments E  
    ON F.folder_id = E.folder_id  
    INNER JOIN SSISDB.internal.environment_variables EV  
    ON E.environment_id = EV.environment_id  
    WHERE (F.name = @FolderName)  
    AND (E.name = @EnvironmentName)  
    AND EV.name = @EnvironmentVariableName);  
  
    -- Return the result of the function  
    RETURN @ParameterValue;  
END;  
GO
```





# Retrieving the Value of a Variable

1. Pulling the value into a variable:

--option 1 - setting a variable value

```
DECLARE @MyServerName VARCHAR(25);
```

```
SET @MyServerName =
```

```
(SELECT dbo.fn_get_environment_variable_value('Demo Folder', 'Demo Environment', 'ServerName_Damo'));
```

```
PRINT @MyServerName;
```

```
GO
```

2. Using it in a select statement:

--option 2 - in a SELECT Statement

```
SELECT dbo.fn_get_environment_variable_value('Demo Folder', 'Demo Environment', 'ServerName_Damo');
```

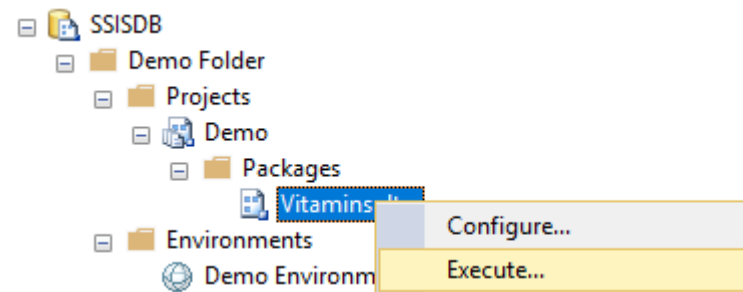


# Executing a Package

1. Execute by right-clicking on the package:

2. Scripting:

```
CREATE PROCEDURE [dbo].[usp_ssis_run_on_demand] (  
    @FolderName NVARCHAR(MAX)  
    ,@ProjectName NVARCHAR(MAX)  
    ,@PackageName NVARCHAR(MAX))  
  
AS  
BEGIN  
    SET NOCOUNT ON;  
  
    DECLARE @ExecutionId BIGINT  
            ,@ReferenceId AS INT;  
  
    SET @ReferenceId =  
    (SELECT      ER.reference_id  
    FROM        SSISDB.catalog.environment_references ER  
    INNER JOIN  SSISDB.catalog.projects P  
    ON P.project_id = ER.project_id  
    WHERE      P.name = @ProjectName);  
  
    EXECUTE SSISDB.catalog.create_execution @folder_name = @FolderName  
        ,@project_name = @ProjectName  
        ,@package_name = @PackageName  
        ,@reference_id = @ReferenceId  
        ,@execution_id = @ExecutionId OUTPUT;  
  
    EXECUTE SSISDB.catalog.start_execution @execution_id = @ExecutionId;  
  
END;  
GO
```



# SSRS Report to View Status and Execute

Knowing the relationship between the Folder, Project, Environment, Environment Variables and other related objects would allow you to recreate the All Executions and All Messages reports in SSRS and make them available to your developers without having to grant them access to the Integration Services Catalog in Production.

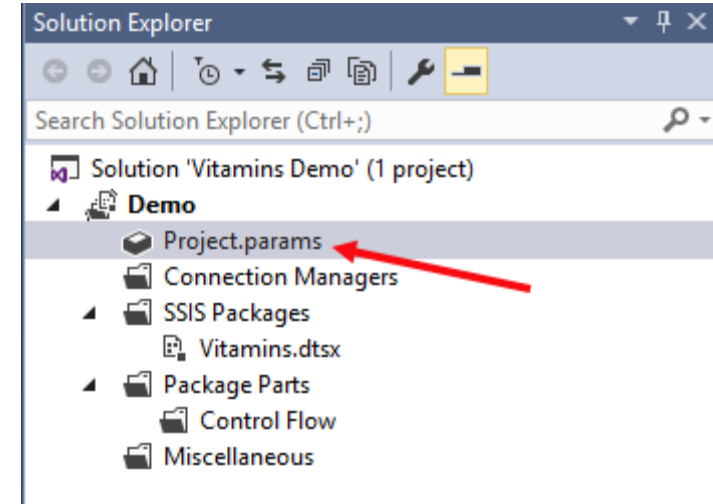
| Folder Name     | Project Name | Package Name      | Status    | Execution Id | Use32Bit At Runtime | Start Date Time     | End Date Time       | Duration | Caller Name                              |
|-----------------|--------------|-------------------|-----------|--------------|---------------------|---------------------|---------------------|----------|--|
| [-] Demo Folder |              |                   |           |              |                     |                     |                     |          |  |
| [-] Demo        |              |                   |           |              |                     |                     |                     |          |  |
|                 |              | [-] Vitamins.dtsx |           |              |                     |                     |                     |          |  |
|                 |              |                   | Failed    | 10*          | False               | 02/25/2018 09:43 PM | 02/25/2018 09:43 PM | 4.993    | MicrosoftAccount\open.sesame@outlook.com |
|                 |              |                   | Failed    | 9*           | False               | 02/25/2018 09:03 PM | 02/25/2018 09:04 PM | 9.326    | MicrosoftAccount\open.sesame@outlook.com |
|                 |              |                   | Succeeded | 8*           | False               | 02/25/2018 01:58 AM | 02/25/2018 01:59 AM | 21.934   | MicrosoftAccount\elaena.bakman@gmail.com |
| Failed:         |              |                   | 2         | Succeeded:   | 1                   |                     |                     |          |  |

| Message Code | Message   | Message Source Name                                | Event Name    | Message Time        | Execution Path   | Package Path   | Message Source Id                      | Environment |
|--------------|---|--|---------------|---------------------|--|--|--|-------------|
| -2147381246  | Vitamins:Warning: SSIS Warning Code DTS_W_MAXIMUMERRORCOUNTREACHED. The Execution method succeeded, but the number of errors raised (1) reached the maximum allowed (1); resulting in failure. This occurs when the number of errors reaches the number specified in MaximumErrorCount. Change the MaximumErrorCount or fix the errors. | Vitamins   | OnWarning     | 02/25/2018 09:04:02 | \Vitamins  | \Package   | {F7A96B02-50DB-4B1E-8B3D-670C10EFAF35} | -           |
|              | Vitamins:Finished, 9:04:02 PM, Elapsed time: 00:00:01.125.  | Vitamins   | OnPostExecute | 02/25/2018 09:04:02 | \Vitamins  | \Package   | {F7A96B02-50DB-4B1E-8B3D-670C10EFAF35} | -           |
|              | Script Task - Show Parameters:Finished, 9:04:02 PM, Elapsed time: 00:00:00.609.   | Script Task - Show Parameters                      | OnPostExecute | 02/25/2018 09:04:02 | \Vitamins\Script Task - Show Parameters                              | \Package\Script Task - Show Parameters   | {89384411-5E14-4F69-8E9A-2B9B78EB6383} | -           |
|              | SQL Task - Check if Staging Tables Exist - onError:Finished, 9:04:02 PM, Elapsed time: 00:00:00.125.  | SQL Task - Check if Staging Tables Exist - onError | OnPostExecute | 02/25/2018 09:04:02 | \Vitamins\OnError\SQL Task - Check if Staging Tables Exist - onError | \Package.EventHandlers[OnError]\SQL Task - Check if Staging Tables Exist - onError | {6be591fb-00ad-4ab4-be1f-ca40a170769e} | -           |
| 1            | Script Task - Show Parameters>Error: Cannot load script for execution.  | Script Task - Show Parameters                      | OnError       | 02/25/2018 09:04:02 | \Vitamins\Script Task - Show Parameters                              | \Package\Script Task - Show Parameters   | {89384411-5E14-4F69-8E9A-2B9B78EB6383} | -           |

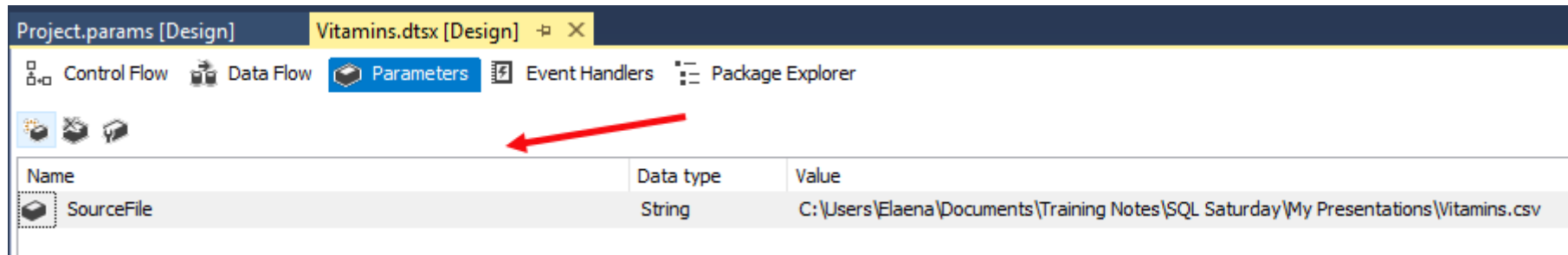


# Project vs. Package Parameters

1. Project level parameters:



2. Package level parameters:



# SSRS Report to View Status and Execute

You can use the SSRS Report to review the parameter mappings:

## SSIS Parameter Value Report

Main Menu

| Folder Name     | Project Name                           | Package Name      | Parameter Name  | Parameter Data Type | Parameter Design Default Value  | Parameter Referenced Variable Name | Environment Variable Name | Environment Variable Description     | Environment Variable Data Type |
|-----------------|--|-------------------|-----------------|---------------------|---|------------------------------------|---------------------------|--------------------------------------|--------------------------------|
| [-] Demo Folder |  |                   |                 |                     |   |                                    |                           |                                      |                                |
|                 | [-] Demo(2/25/2018 11:47:47 PM -08:00) |                   |                 |                     |   |                                    |                           |                                      |                                |
|                 |  | [-] Vitamins.dtsx |                 |                     |   |                                    |                           |                                      |                                |
|                 |  |                   | SourceFile      | String              | C:\Users\Elaena\Documents\Training Notes\SQL Saturday\My Presentations\Vitamins.csv | Vitamins_Source_Path               | Vitamins_Source_Path      | Vitamins Demo Project file path.     | String                         |
|                 |  |                   | SQL_Server_Name | String              | THOR  | Vitamins_SQL_Server_Name           | Vitamins_SQL_Server_Name  | Vitamins Demo Project SQLServer name | String                         |

... and review the parameter scope:

| Environment Variable Description     | Environment Variable Type | Environment Variable Value   | Variable Level |
|--------------------------------------|---------------------------|--|----------------|
| Name of the vitamin to export.       | String                    | Vitamin B12  | Package        |
| Vitamins Demo Project file path.     | String                    | C:\Users\Elaena\Documents\Training Notes\SQL Saturday\My Presentations\Demo Export | Package        |
| Vitamins Demo Project SQLServer name | String                    | THOR   | Project        |



# Permissions

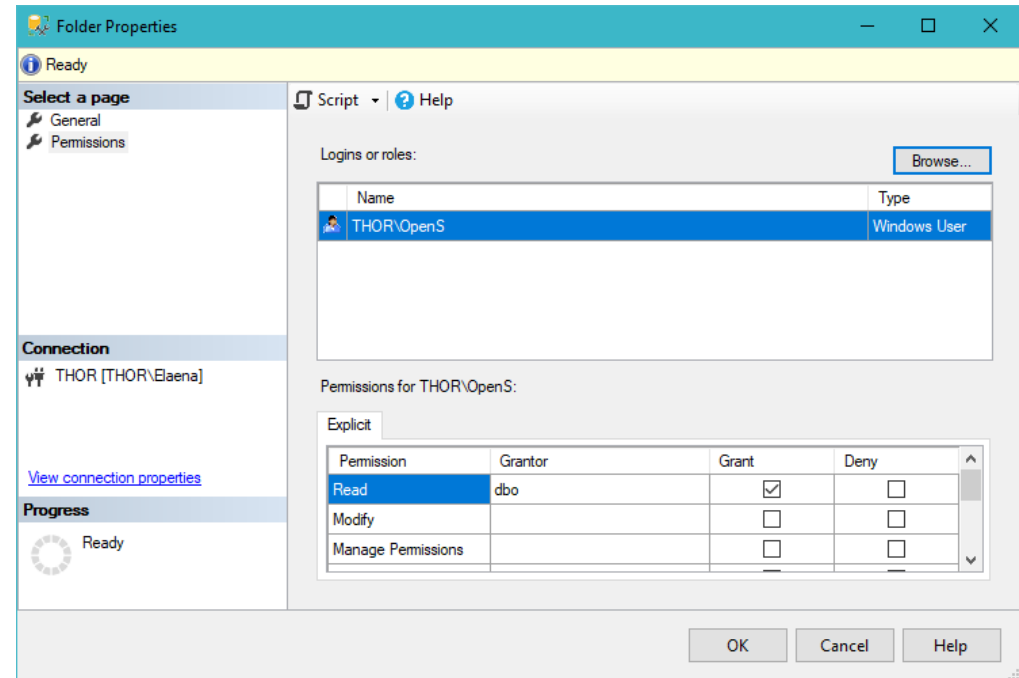
1. Use a Windows Account to for your SSRS (one for each environment). You cannot use a SQL Server Account:

*"The operation cannot be started by an account that uses SQL Server Authentication. Start the operation with an account that uses Windows Authentication.  
A .NET Framework error occurred during execution of user-defined routine or aggregate "start\_execution\_internal":  
System.Data.SqlClient.SqlException: The operation cannot be started by an account that uses SQL Server Authentication. Start the operation with an account that uses Windows Authentication."*

2. Grant Access to SSISDB: role member of ssis\_admin.

3. Grant Access to Folder

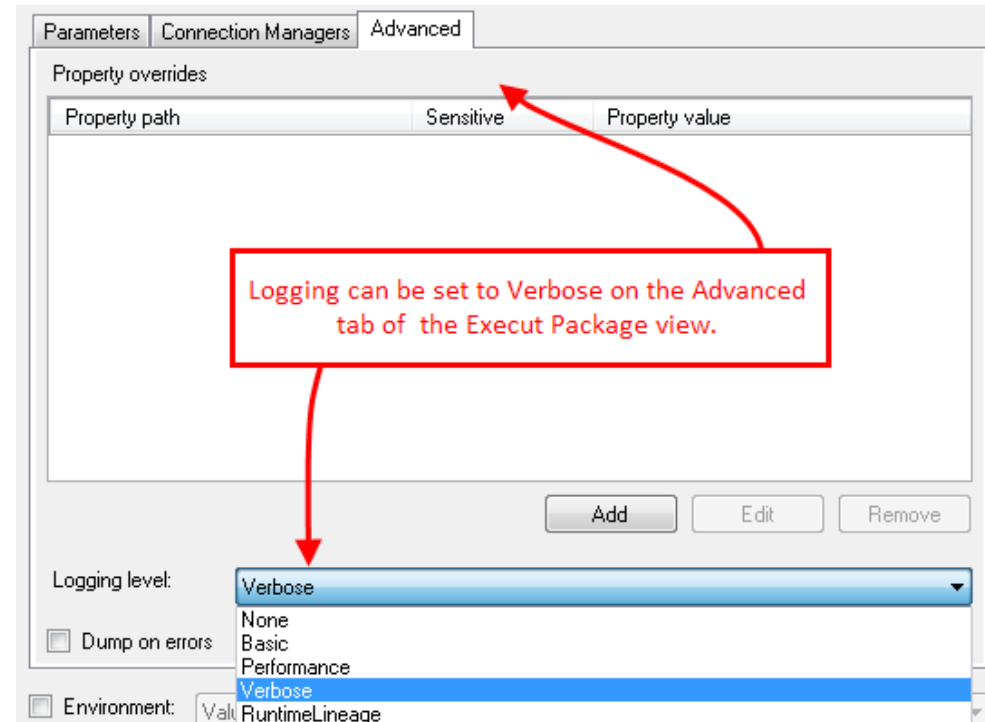
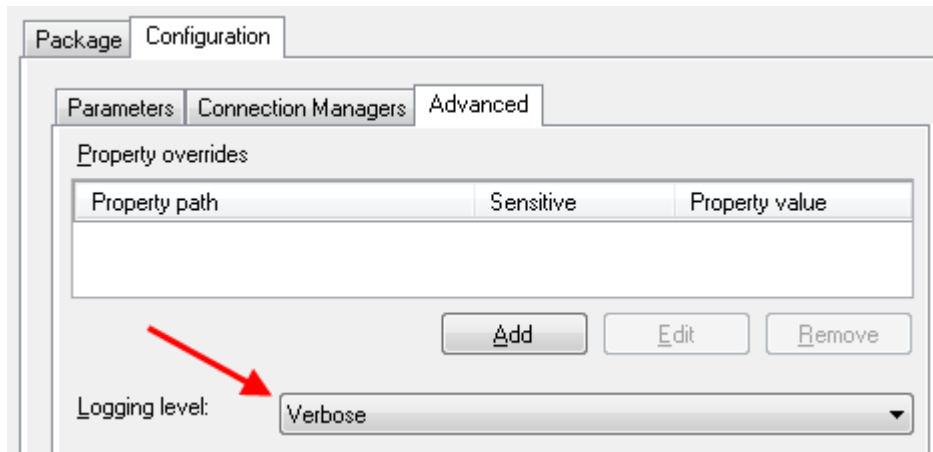
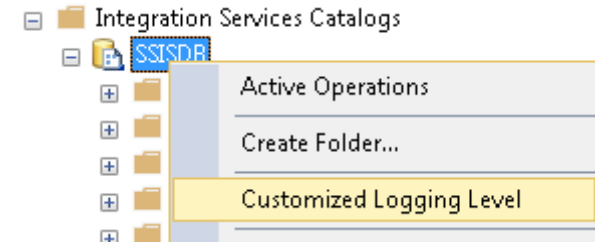
- Read
- Execute Objects
- Read Objects



# Logging

To get row counts enable “Verbose” logging. Logging type can be changed:

1. At the catalog level:
2. When executing a package using Management Studio:
3. Job step Advanced tab under Configuration:



# Questions?





Email: [elaena.bakman@gmail.com](mailto:elaena.bakman@gmail.com)

Twitter: [@Elaena\\_Bakman](https://twitter.com/Elaena_Bakman)

GitHub: <https://github.com/Lenka72/The-Magic-of-SSISDB-Demo>

# Follow me...

