December 27, 2021

USING SSIS CATALOG ARCHIVE

PRODUCT DOCUMENTATION

Version	Date	Author(s)	Summary
0.1	27 Dec 2021	Andy Leonard	Initial draft

Contents

0 PREFACE

The SQL Server Integration Services (SSIS) Catalog was released with SQL Server 2012. By design, the SSIS Catalog is a *framework*, defined by the author as a collection of functionalities designed to support configuration, execution, and logging.

As noted by Microsoft Data Platform MVPs, authors, and bloggers, the SSIS Catalog database – named SSISDB – may become bloated over time by collecting reasonable log information to support enterprise-scale data engineering / integration workloads. The SSIS Catalog does a good job of managing the workload (with a little help from its friends – like SSIS Framework and SSIS Catalog Compare). Managing database size – especially the size of the logs – is another matter.

Add into this scenario increasing regulatory demands for data retention and reporting over time… and you have a clearer idea why SSIS Catalog logs are important.

In this document, the author covers the following topics:

1. Obtain the SSISDBArchive solution
2. Create the SSISDBArchive Database
3. Deploy the Archive_SSISDB SSIS project
4. Configure the Archive_SSISDB SSIS project
5. Execute the Archive_SSISDB SSIS project
6. Examine the Archive_SSISDB SSIS project controller

Andy

1 Obtaining the SSISDBArchive Solution

To acquire the code for the SSISDBArchive solution, please visit github.com/aleonard763/ArchiveSSISDB. Download the code. An option to download the repository contents as a zip file exists, and the author uses this option regularly:

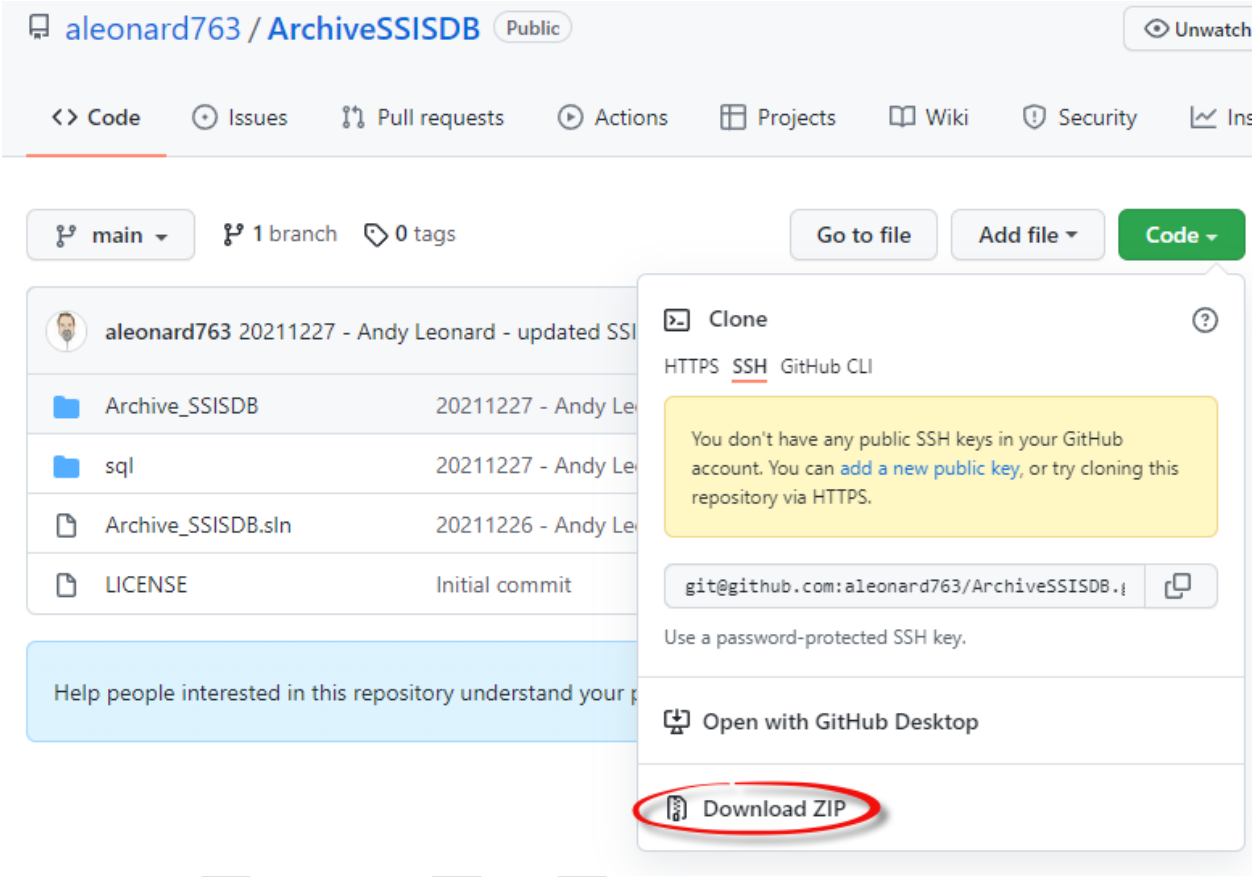

Figure 1

Extract the compressed (zip) file contents.

2 Creating the SSISDBArchive Database

Use SQL Server Management Studio (SSMS) or Azure Data Studio top open the file named Create SSISDBArchive Database.sql found in the sql directory:

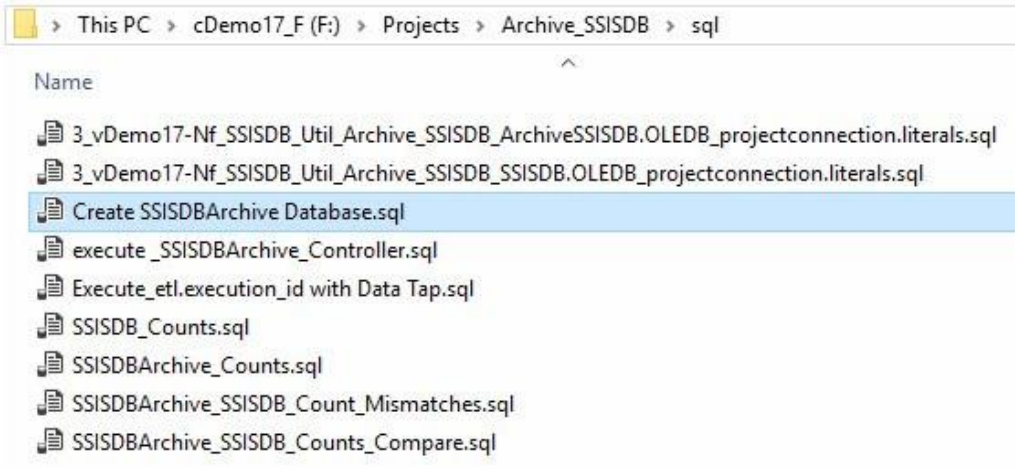

Figure 2

Once open, the Create SSISDBArchive Database.sql appears as shown in Figure 3 (using SSMS):

```
/*

  Script: Create SSISDBArchive Database.sql
  Andy Leonard
  23 Dec 2021

*/

/* Reset NoExec in case it is on */
Set NoExec OFF

Use [master]
go

print 'SSISDBArchive database'
If Not Exists(Select [name]
              From [sys].[databases]
              Where [name] = N'SSISDBArchive')
 begin
  print ' - Creating SSISDBArchive database'
  Create Database SSISDBArchive
  print ' - SSISDBArchive database created'
 end
```

Figure 3

Please check script and environment settings to make certain SSMS is connected to the proper instance of SQL Server.

Always backup SSISDB and other databases regularly.

Execute the Create SSISDBArchive Database.sql script to create the SSISDBArchive database.

The initial execution of the Create SSISDBArchive Database.sql script generate messages similar to those shown in Figure 4:

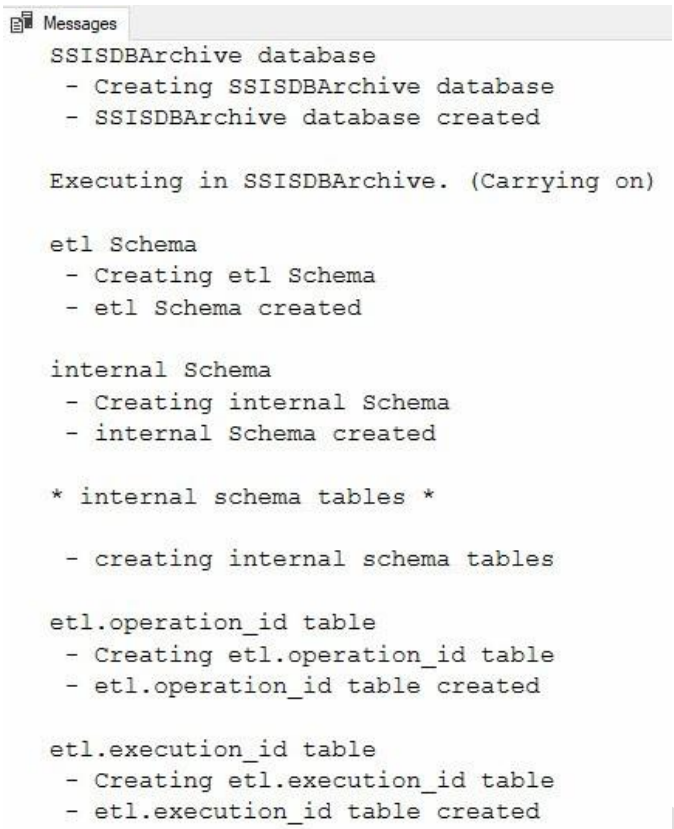

Figure 4

Because the script is re-executable, subsequent executions of the Create SSISDBArchive Database.sql script generate messages similar to those shown in Figure 5:

Messages

```
SSISDBArchive database
 - SSISDBArchive database already exists.

Executing in SSISDBArchive. (Carrying on)

etl Schema
 - etl Schema already exists.

internal Schema
 - internal Schema already exists.

* internal schema tables *

 - creating internal schema tables

etl.operation_id table
 - etl.operation_id table already exists.

etl.execution_id table
 - etl.execution_id table already exists.

etl.executable_id table
 - etl.executable_id table already exists.

etl.validation_id table
 - etl.validation_id table already exists.
```

Figure 5

Refresh the Databases node in SSMS's Object Explorer to view the new database:

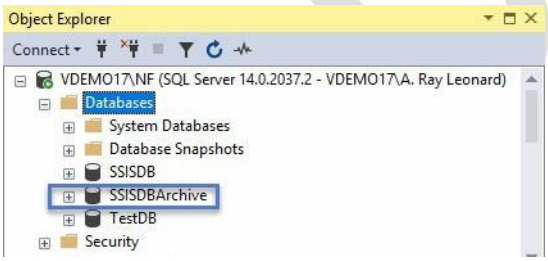

Figure 6

Execute an initial count of SSISDBArchive database table rows using the SSISDBArchive_Counts.sql script included in the sql directory:

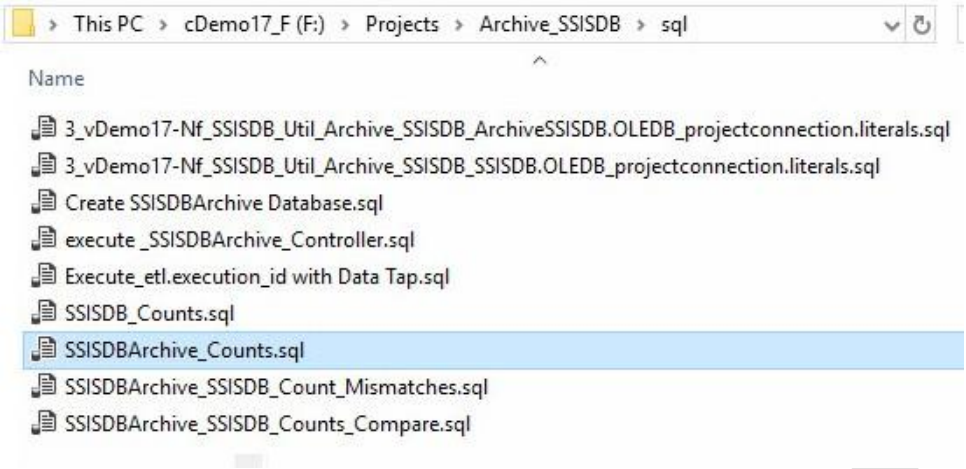

Figure 7

At this time, all tables should exist and contain no rows, as shown in Figure 8:

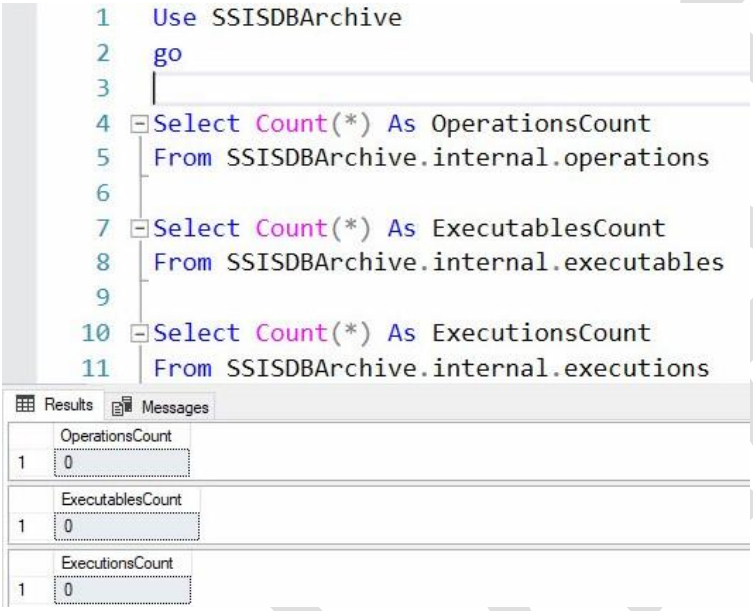

Figure 8

The SSISDBArchive database has been deployed. The next step is to deploy the Archive_SSISDB SSIS project to the SSIS Catalog.

3 Deploying the Archive_SSISDB SSIS Project

Navigate to the solution directory, right-click the Archive_SSISDB.sln file, and then click Open, as shown in Figure 9:

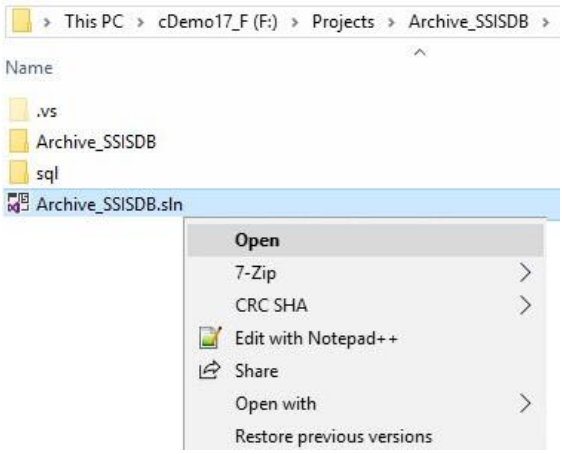

Figure 9

When Visual Studio opens, Solution Explorer should appear as shown in Figure 10:

Figure 10

To begin deploying, right-click on the SSIS project – the item in th green box in the image above – and then click Deploy:

Figure 11

If the Introduction page displays when the Integration Services Deployment Wizard opens, click Next to proceed, as shown in Figure 12:

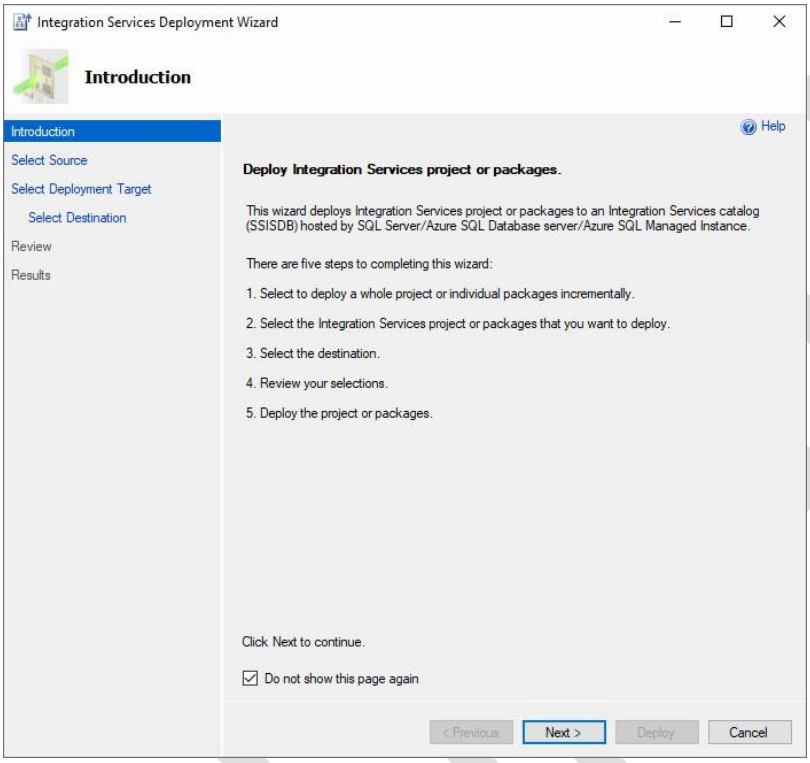

Figure 12

When the Select Deployment Target page displays, select the type of your target database, and then click the Next button to proceed:

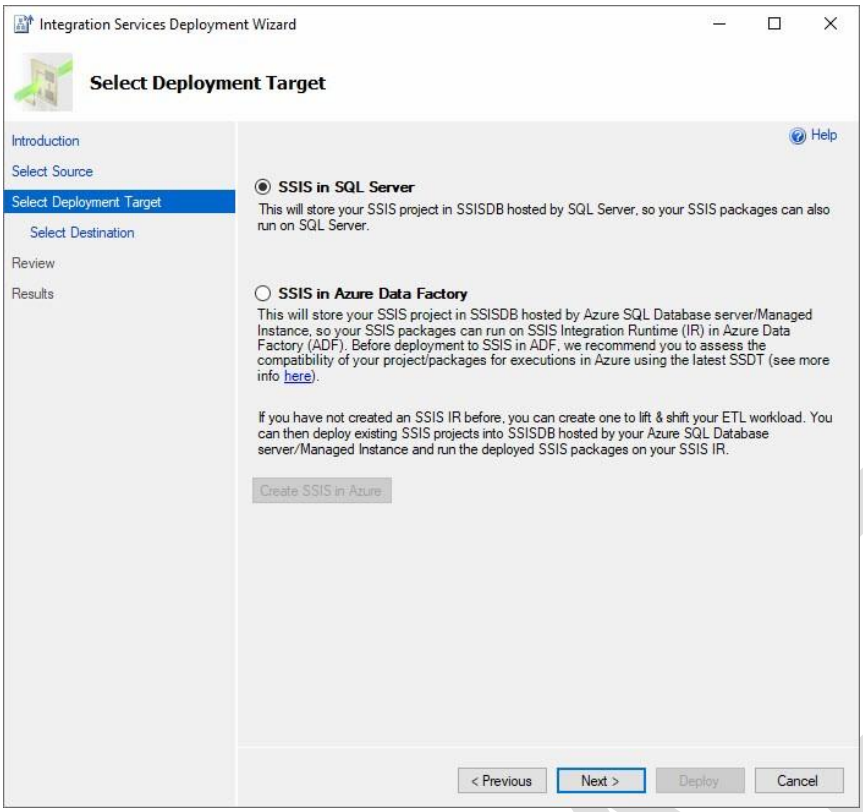

Figure 13

The Select Destination page displays. Enter the name of the target SQL Server instance in the Server Name textbox. Choose an authentication method from the Authentication dropdown. Complete additional authentication properties, if required. Click the Connect button to connect to the target SQL Server instance.

Once connected, the Path property textbox is enabled. An error may display indicating “The folder ‘Util’ was not found in catalog ‘SSISDB’.” as shown in Figure 14:

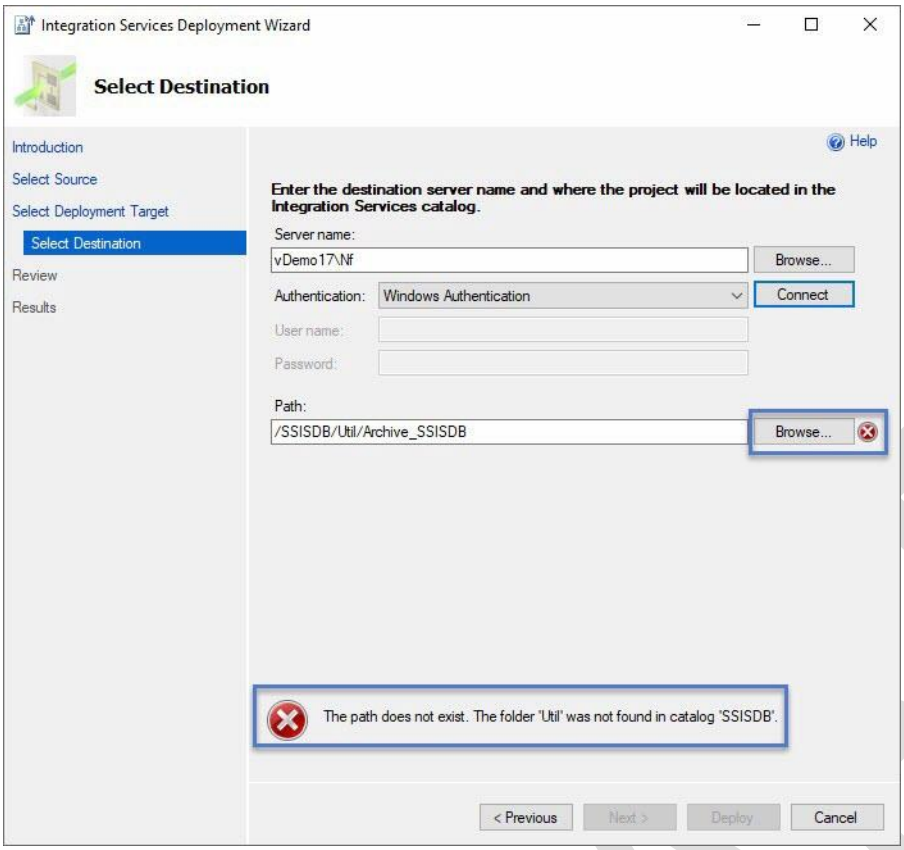

Figure 14

Clear the error by clicking the Browse button shown in Figure 14.

When the Browse for Folder or Project dialog displays, click the “New folder…” button shown in Figure 15:

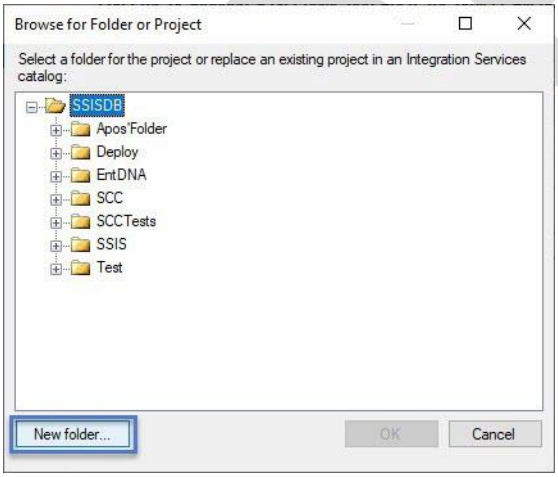

Figure 15

When the Create New Folder dialog displays, create the Util folder as shown in Figure 16:

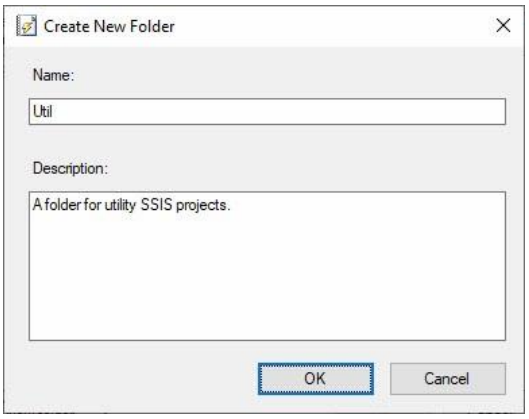

Figure 16

Click the OK button to complete the Util folder creation, and then click the OK button to select the Util folder as the target SSIS Catalog Folder, as shown in Figure 17:

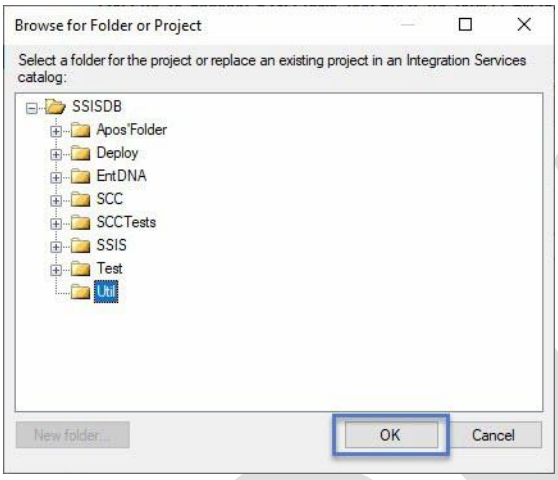

Figure 17

The SSIS project path displays – without error – when you return to the Select Destination page:

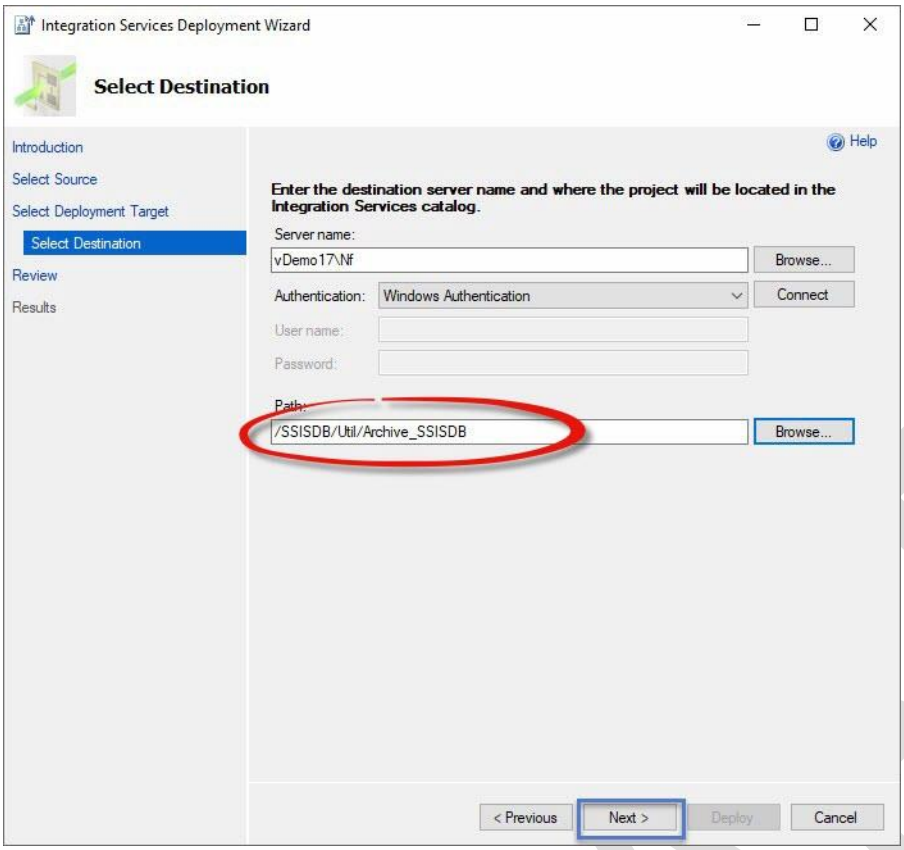

Figure 18

Click the Next button to proceed.

The Review page displays deployment configuration information, similar to that shown in Figure 19:

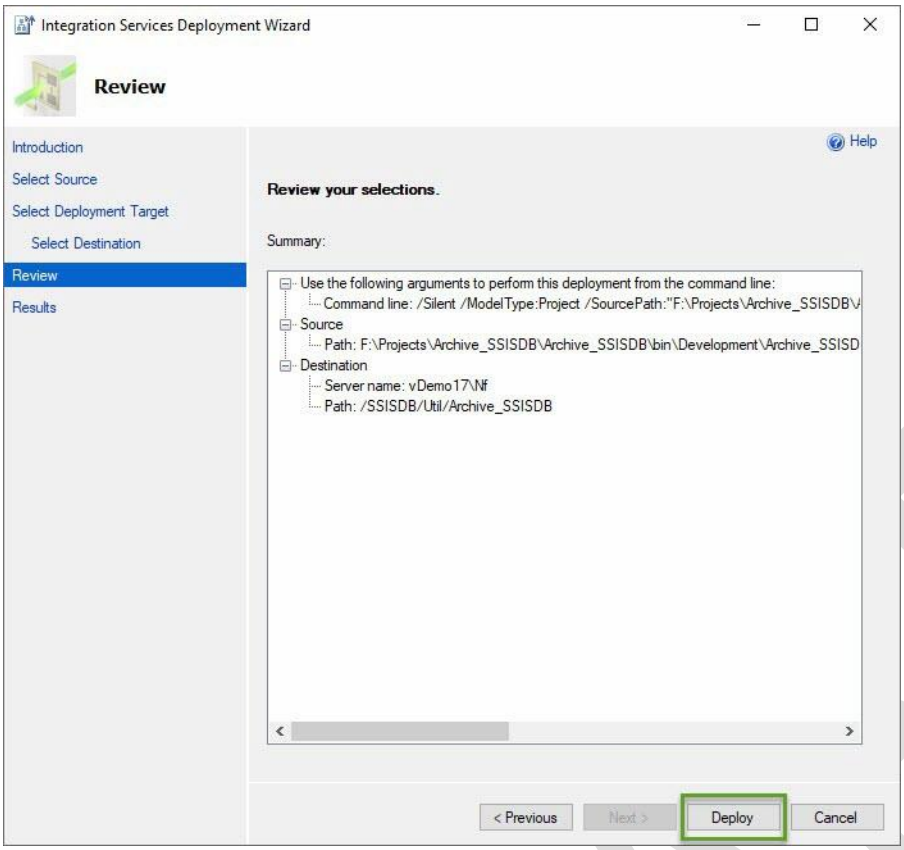

Figure 19

Click the Deploy button to deploy the Archive_SSISDB project to the SSIS Catalog. Once deployed, the Integration Services Deployment Wizard should display the Results page similar to Figure 20:

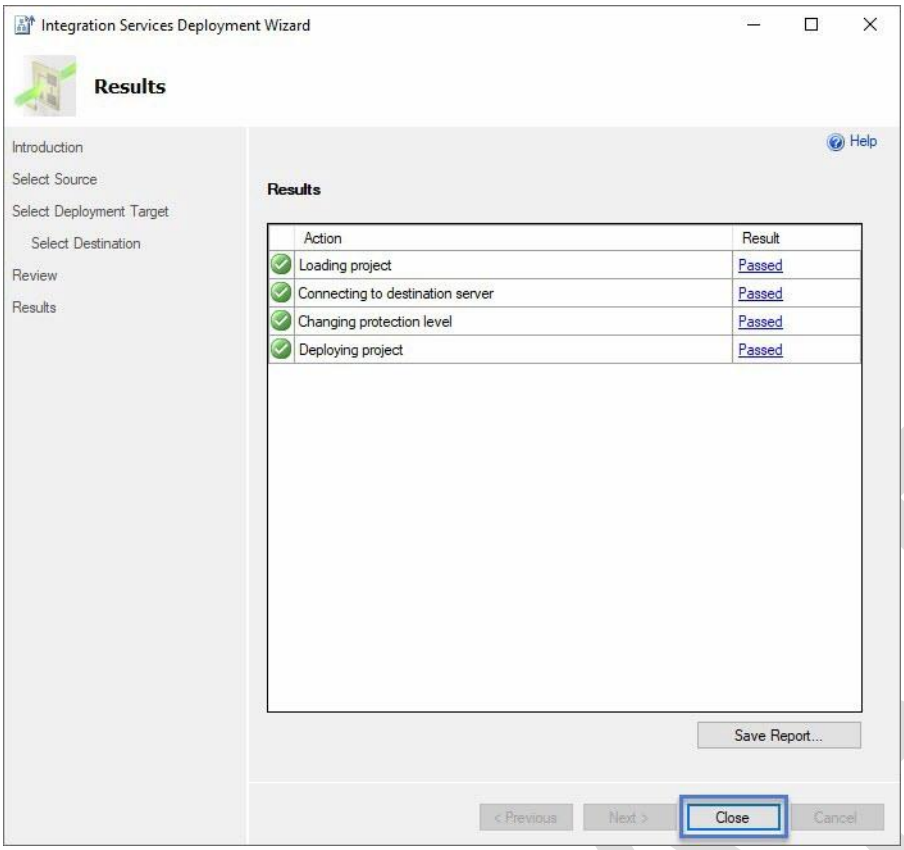

Figure 20

Navigate to the SSMS Object Explorer's Integration Services Catalogs node. Refresh the Catalog node to confirm the Archive_SSISDB project has been deployed, as shown in Figure 21:

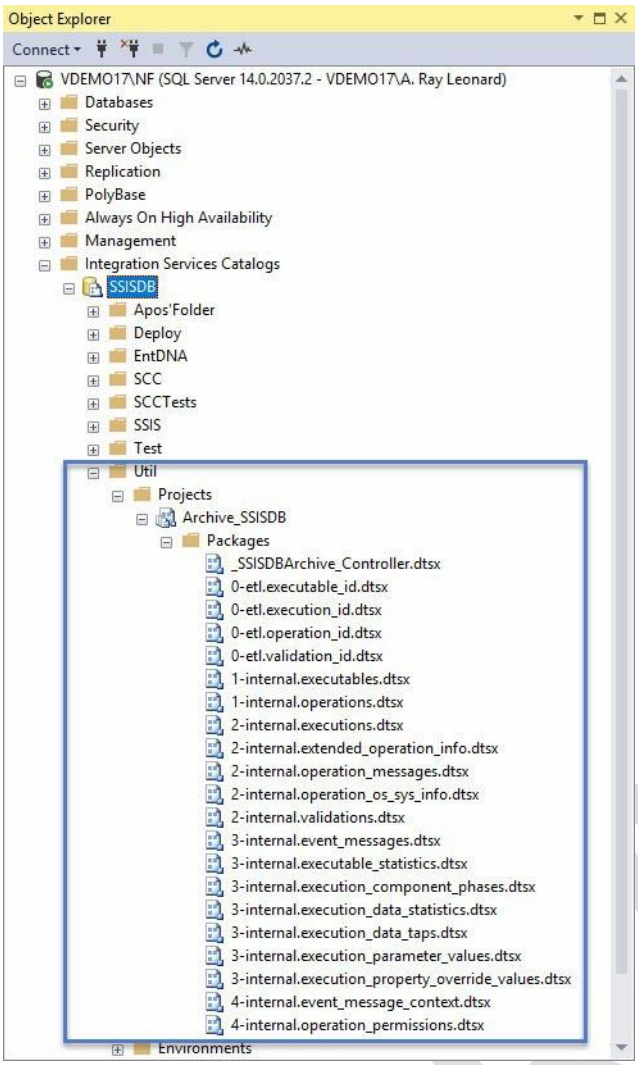

Figure 21

The Archive_SSISDB SSIS project is now deployed. The next step is to configure the Archive_SSISDB project.

4 Configuring the Archive_SSISDB SSIS Project

Here are two ways to update the update the Archive_SSISDB SSIS Project connection manager ConnectionString properties in the SSIS Catalog:

1. Manually update the Archive_SSISDB SSIS project connection manager ConnectionString properties
2. Update the Archive_SSISDB SSIS project connection manager ConnectionString properties via script

Manually Updating the Archive_SSISDB SSIS Project Connection Manager ConnectionString Properties

In SSMS's Object Explorer, expand the Integration Services Catalogs→SSISDB→Util→Projects→Archive_SSISDB node. Right-click "Configure…" as shown in Figure 22:

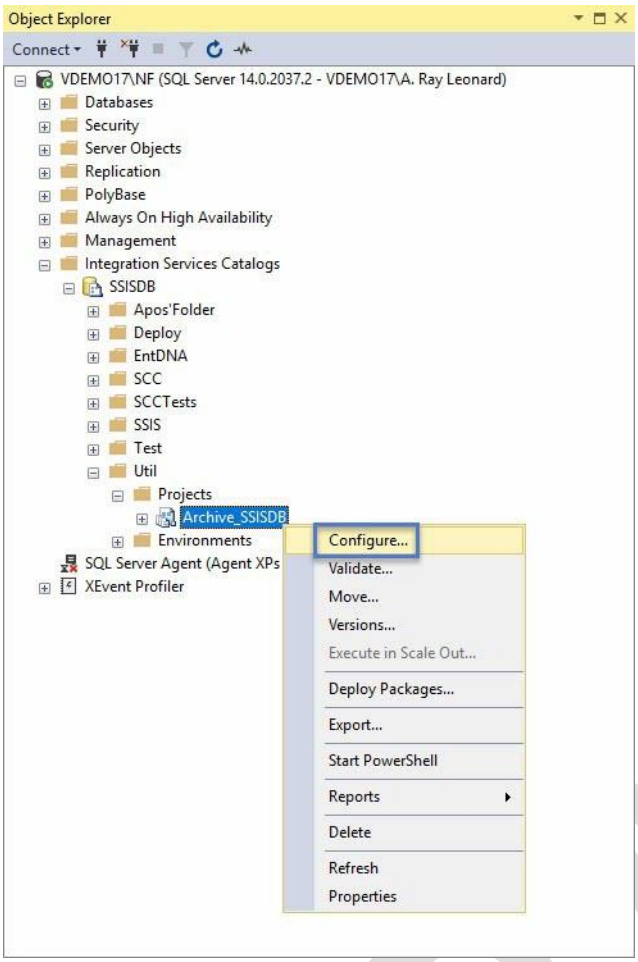

Figure 22

When the Configure – Archive_SSISDB window displays, click the Connection Managers tab. Select the connection manager named "ArchiveSSISDB.OLEDB". The PropertyGrid to the right surfaces configurable properties for the ArchiveSSISDB.OLEDB (project-scoped) connection manager. Click the ellipsis beside the ConnectionString property, as shown in Figure 23:

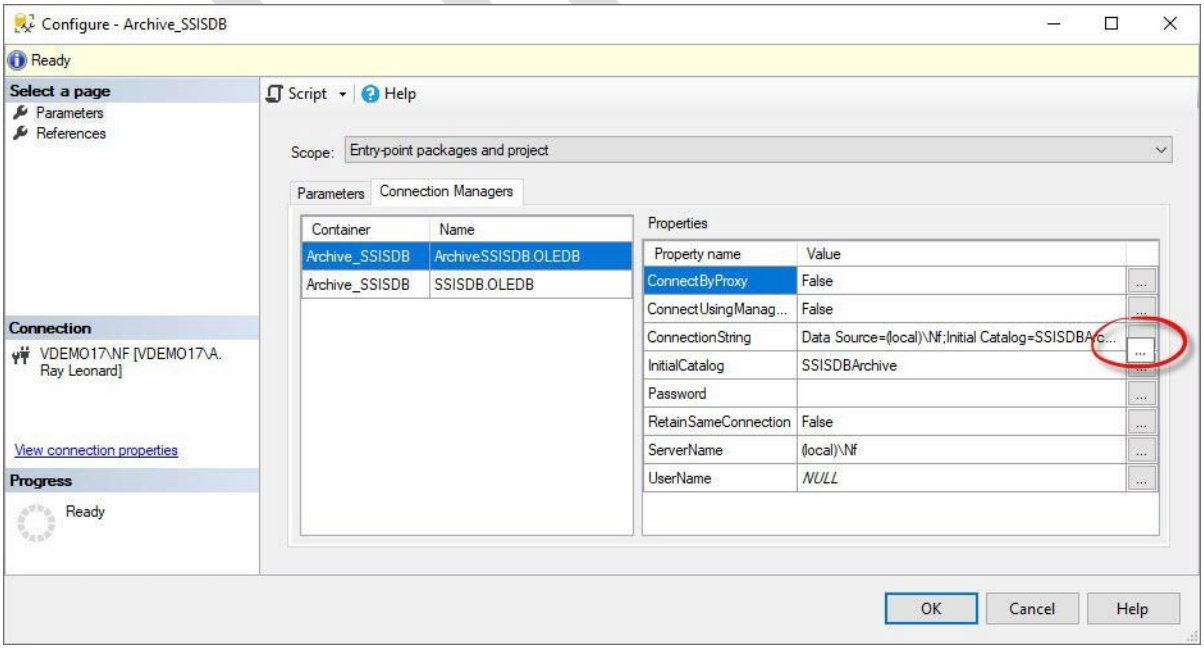

Figure 23

When the Set Parameter Value dialog opens, note the Parameter property displays the name of the parameter being configured:

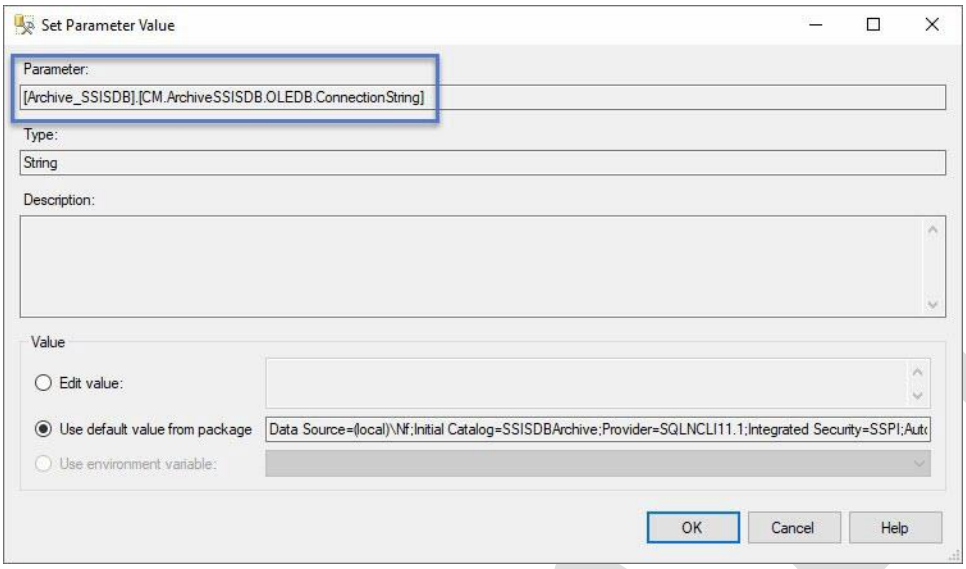

Figure 24

You may copy the default value for the ConnectionString property, click the "Edit value" option, paste the value into the textbox, and then edit the ConnectionString property, as shown in Figure 25:

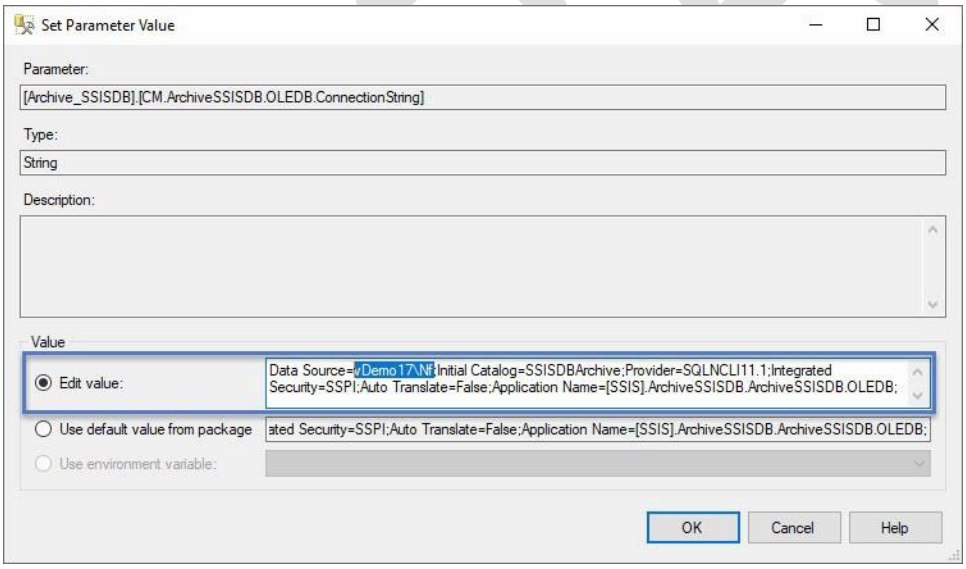

Figure 25

When done, click the OK button to return to the Configure – Archive_SSISDB window.

Note the new ConnectionString appears with bold font – indicating the value is a literal – as shown in Figure 26:

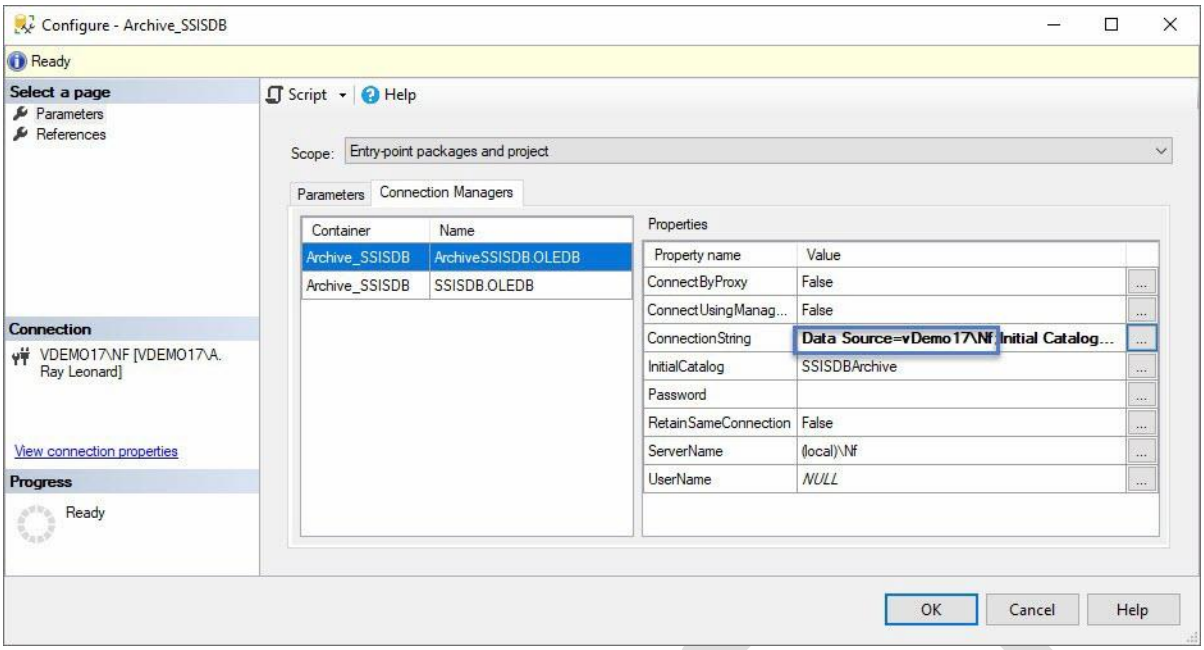

Figure 26

To the left of the Connection Manager properties PropertyGrid, select the connection manager named “SSISDB.OLEDB”. As with the connection manager named “ArchiveSSISDB.OLEDB,” click the ellipsis for the ConnectionString property to open the Set Parameter Value dialog.

Copy the default value for the ConnectionString property, click the “Edit value” option, paste the value into the textbox, and then edit the ConnectionString property, as shown in Figure 27:

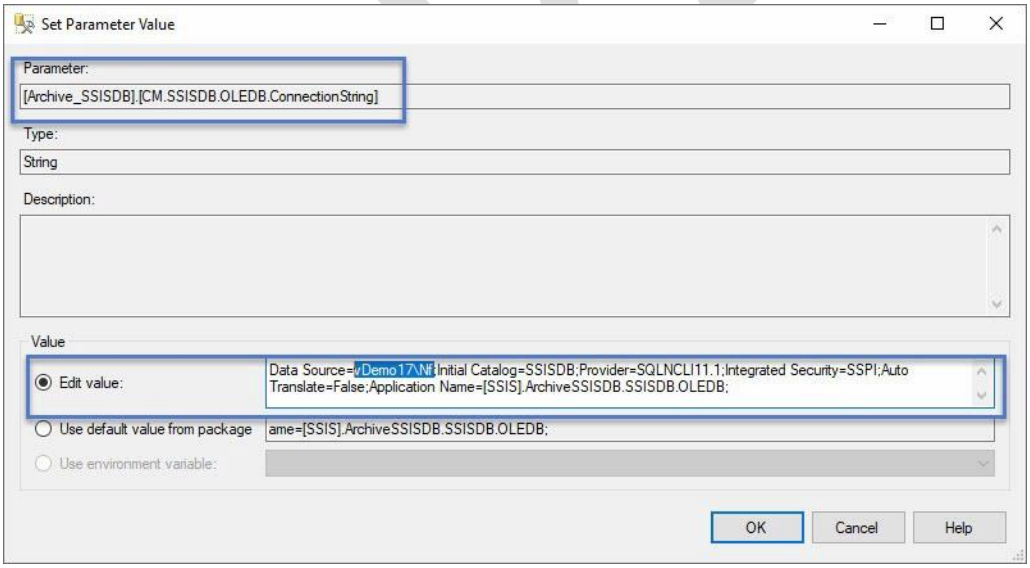

Figure 27

When done, click the OK button to return to the Configure – Archive_SSISDB window.

Note the new ConnectionString appears with bold font – indicating the value is a literal – as shown in Figure 28:

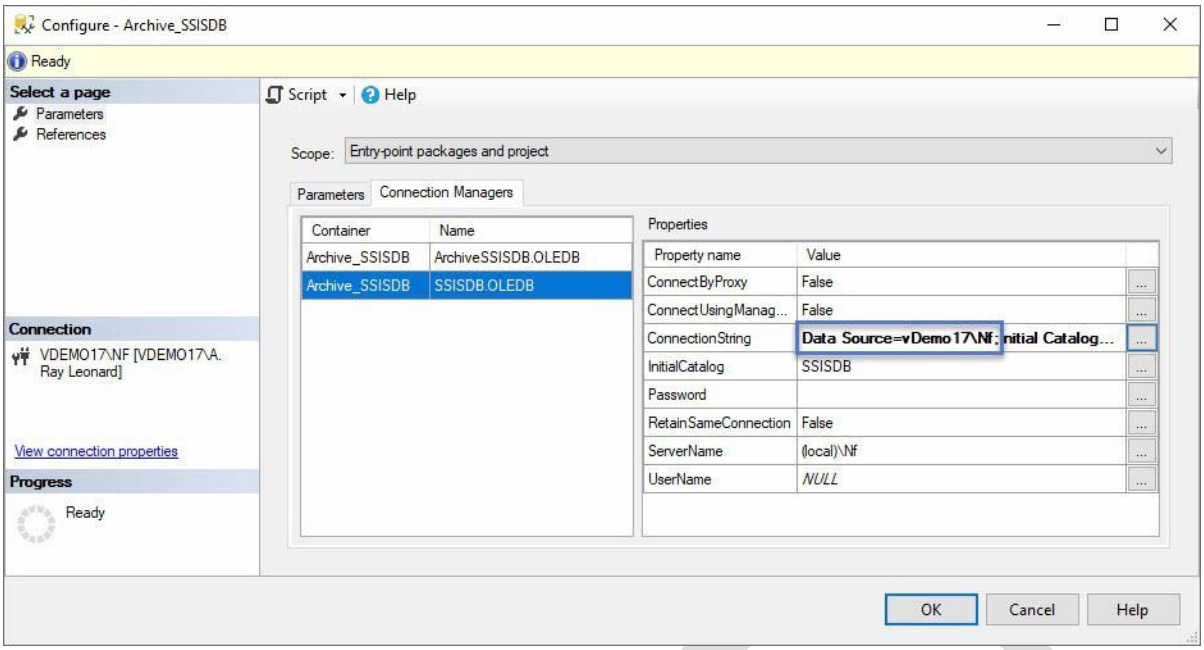

Figure 28

Click the OK button to close the Configure – Archive_SSISDB SSIS project window and apply the configured literals for the connection manager ConnectionString properties.

You may also configure literals for the Archive_SSISDB SSIS project connection manager ConnectionString properties using T-SQL scripts.

Updating the Archive_SSISDB SSIS Project Connection Manager ConnectionString Properties via Scripts

In the sql directory, please find two scripts that will update the Archive_SSISDB SSIS project connection manager ConnectionString properties using SSIS Catalog configured literals. The scripts are named:

- 3_vDemo17-Nf_SSISDB_Util_Archive_SSISDB_SSISDB.OLEDB_projectconnection.literals.sql
- 3_vDemo17-Nf_SSISDB_Util_Archive_SSISDB_ArchiveSSISDB.OLEDB_projectconnection.literals.sql

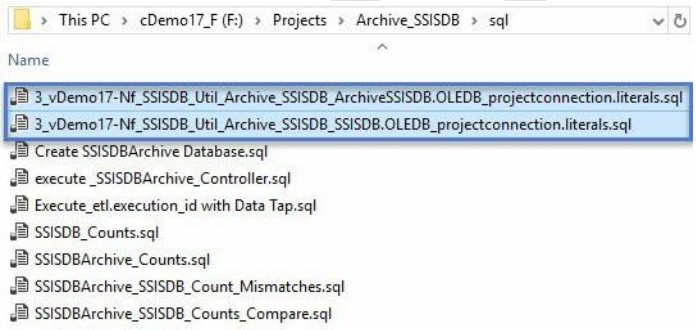

Figure 29

These scripts were generated by SSIS Catalog Compare, which is designed to support enterprise DevOps for SSIS Catalogs that participate in the Data Integration Lifecycle. Learn more at Data Integration Lifecycle Management (DILM) Suite.

You may open these scripts in SSMS and edit the parameter values at the top of each script, as shown in Figure 30:

```
-- PROJECT CONNECTION PARAMETER LITERAL VALUES --

-- Archive_SSISDB\CM.SSISDB.OLEDB.ConnectionString_0 Project Connection Parameter Literal
Declare @ProjectParameter_CM_dot_SSISDB_dot_OLEDB_dot_ConnectionString_0 sql_variant = N'Data Source=vDemo17\Nf;Initial Catalog=SSISDB;Provider

----------------------------

/*
Script Name: F:\_test\20211227\vDemo17-Nf\Util\3_vDemo17-Nf_SSISDB_Util_Archive_SSISDB_SSISDB.OLEDB_projectconnection.literals.sql
Generated From Catalog Instance: vDemo17\Nf
Catalog Name: SSISDB
Folder Name: Util
Project Name: Archive_SSISDB
Project Connection Name: SSISDB.OLEDB
Generated By: VDEMO17\A. Ray Leonard
```

Figure 30

Each script returns messages to indicate the results of checks and stored procedure execution(s), as shown in Figure 31:

```
1  -- PROJECT CONNECTION PARAMETER LITERAL VALUES --
2
3  -- Archive_SSISDB\CM.SSISDB.OLEDB.ConnectionString_0 Project Connection Parameter Literal
4  Declare @ProjectParameter_CM_dot_SSISDB_dot_OLEDB_dot_ConnectionString_0 sql_variant = N'Data Source=vDemo17\Nf;Initial Catalog=SSISDB;Provider
5
6  ----------------------------
Messages
  - Add or Update Util\Archive_SSISDB\CM.SSISDB.OLEDB.ConnectionString Project Connection Parameter Literal
  - Updating Util\Archive_SSISDB\CM.SSISDB.OLEDB.ConnectionString Project Connection Parameter Literal
  - Util\Archive_SSISDB\CM.SSISDB.OLEDB.ConnectionString Project Connection Parameter Literal updated
  - Util\Archive_SSISDB\CM.SSISDB.OLEDB.ConnectionString Project Connection Parameter Value set to: Data Source=vDemo17\Nf;Initial Catalog=SSISDB;Provider=SQLNCLI1
```

Figure 31

Repeat for the script named 3_vDemo17-Nf_SSISDB_Util_Archive_SSISDB_ArchiveSSISDB.OLEDB_projectconnection.literals.sql.

SSIS Catalog configuration for the Archive_SSISDB SSIS project is complete. The next step is to execute the project.

5 Executing the Archive_SSISDB SSIS Project

In SSMS's Object Explorer, expand the Integration Services Catalogs→SSISDB→Util→Projects→Archive_SSISDB→Packages node. Right-click the _SSISDGArchive_Controller.dtsx SSIS package, and then click "Execute…" as shown in Figure 32:

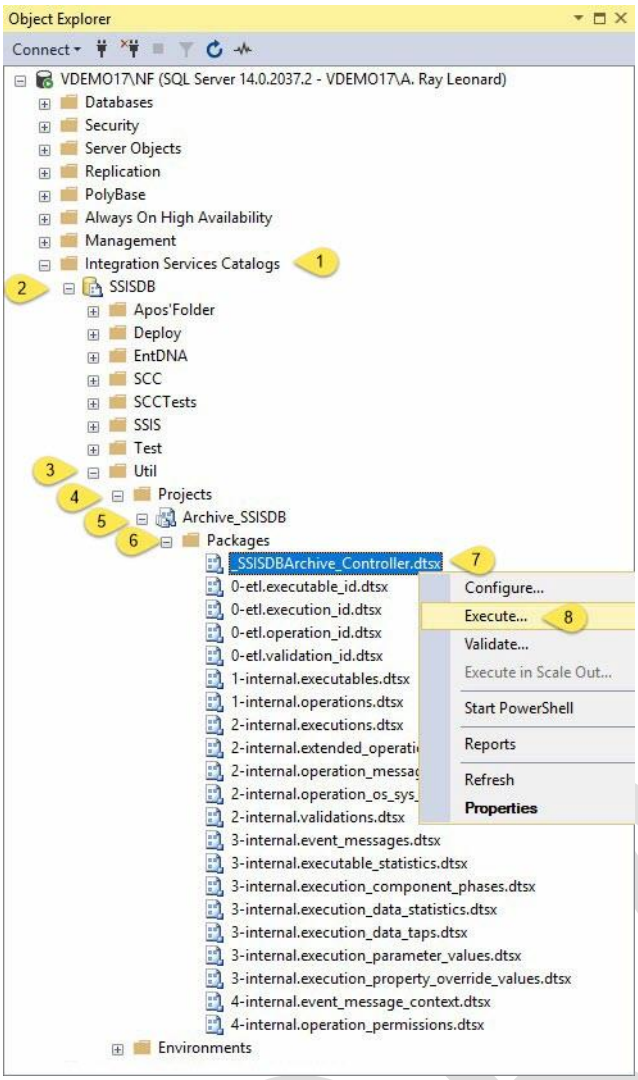

Figure 32

When the Execute Package dialog displays, (optionally) click the Script button to create a script you may use to execute – or schedule execution – of the _SSISDGArchive_Controller.dtsx SSIS package in the future. An example of this script is found in the sql directory; named execute _SSISDBArchive_Controller.sql.

To execute the execute _SSISDBArchive_Controller SSIS package immediately, click the OK button:

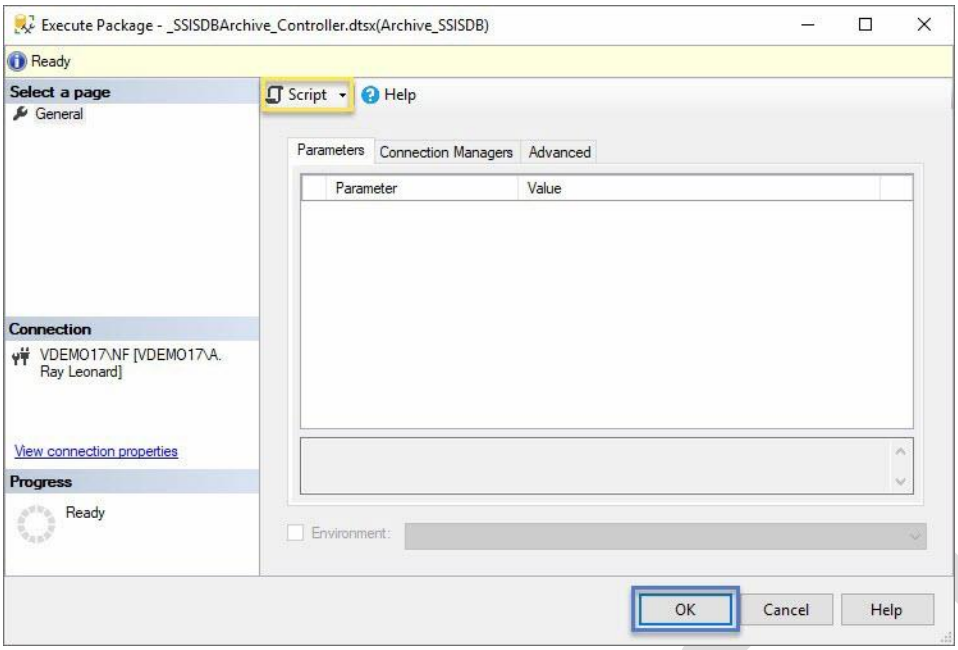

Figure 33

If you click the OK button to execute the controller SSIS package, SSMS prompts you to view the SSIS Catalog Overview Report for this execution, as shown in Figure 34:

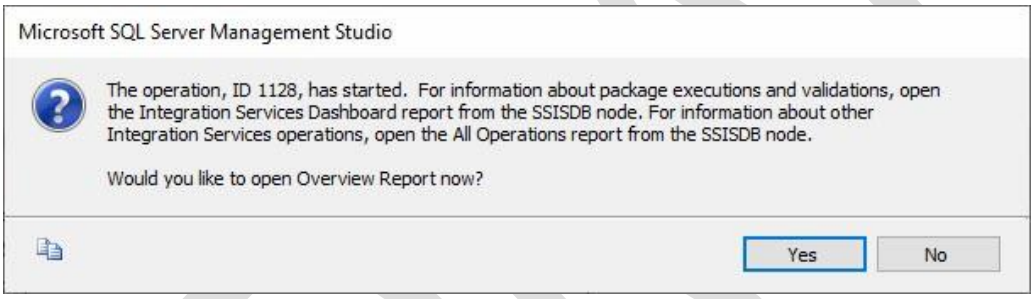

Figure 34

Click the Yes button to view the SSIS Catalog Overview Report for this execution. Once complete, the report shows the status of the execution. If all goes as planned, the execution succeeds, as shown in Figure 35:

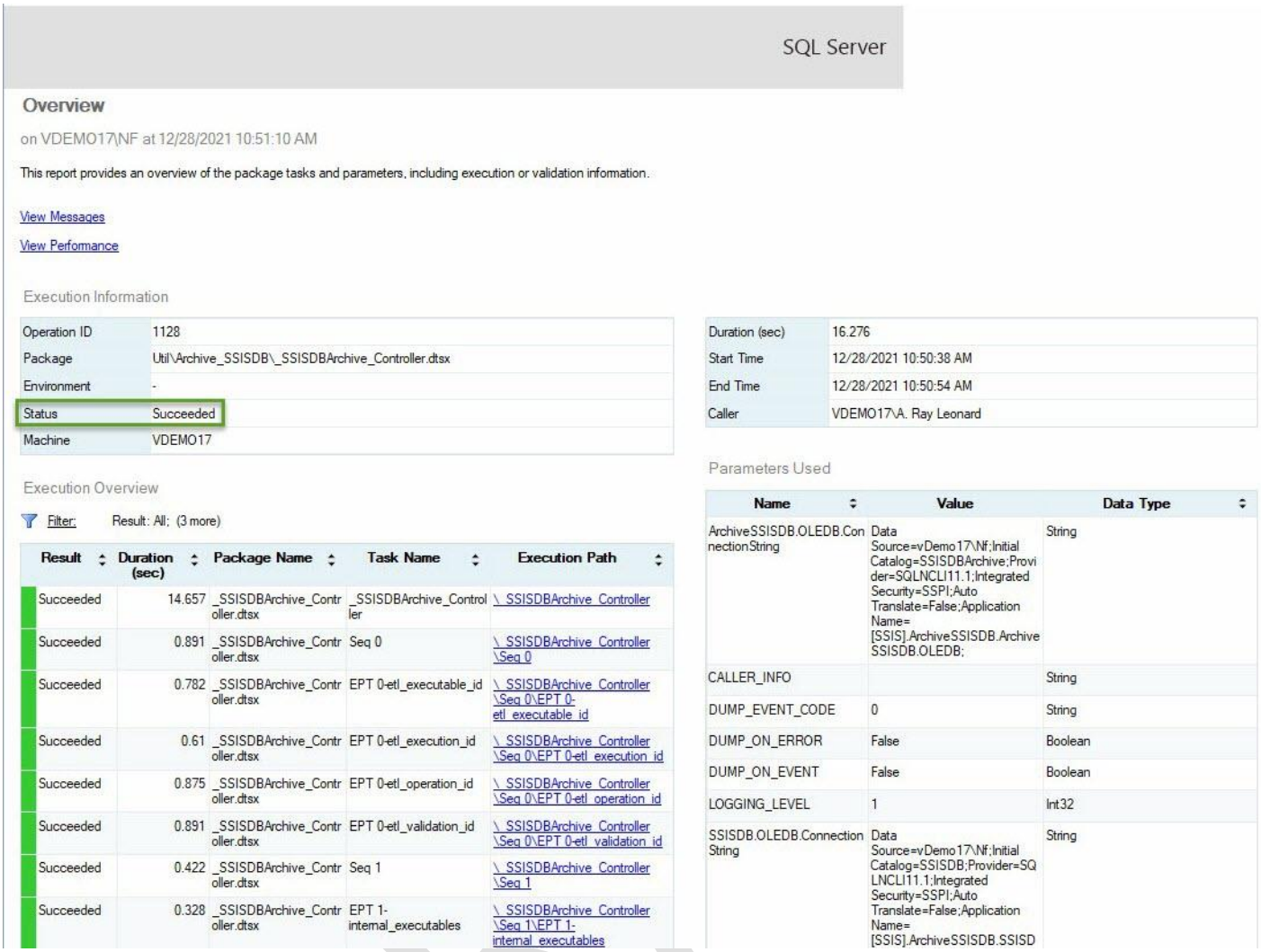

Figure 35

This SSIS design pattern is named "Controller" because it controls the execution order and parallelism of other SSIS packages. The controller design pattern is an application of the "Parent-Child" design pattern. The controller is the parent; each package called by the parent is a "Child" package.

The controller package calls child SSIS packages in an order that respects the referential integrity – maintained by foreign keys – in the SSIS Catalog.

6 Examining the Design of the Archive_SSISDB SSIS Project Controller

The "_SSISDB_Archive_Controller.dtsx" SSIS package is the controller package for the Archive_SSISDB project. The controller contains several Execute package Tasks. The Execute Package tasks reside in Sequence Containers. Each Sequence container represents a "step" in the archive process.

Note the "Seq 0" sequence container calls four child packages:

1. 0-etl.operation_id
2. 0-etl.executable_id

3. 0-etl.execution_id
4. 0-etl.validation_id

The Execute Package tasks in Seq 0 *starts* the four child packages at *nearly* the same time. There are no precedence constraints present. The child packages execute in parallel. Or rather, the child packages *begin* to execute in parallel. Each child package execution completes when its execution is complete.

Because Seq 0 is a container and the precedence constraint between Seq 0 and Seq 1 is configured as "On Success," Seq 1 tasks will not begin executing *until all child packages in Seq 0 complete execution and only if all child packages in Seq 0 succeed*.

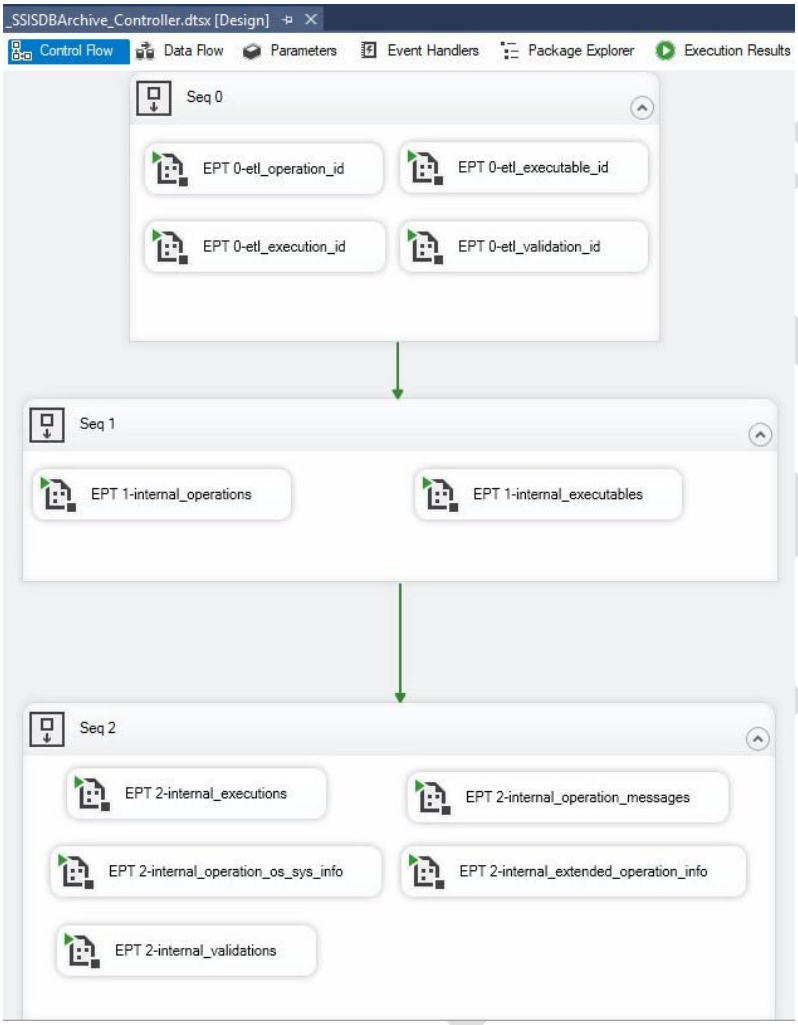

Figure 36

7 Conclusion

The Archive_SSISDB project is designed to help migrate SSIS log data from an SSIS Catalog prior to removing said records from the SSIS Catalog.