SQL Moderation Hack – SSIS Migration Lab

V2.4

Contents

PROBLEM STATEMENT

In Lab 1 of this hack, you have migrated 3 databases to Azure for the application Transaction Reporting Application.

Now that the databases for the Transactional Reporting Application have been migrated, there is a set of additional SSIS packages on the LEGACYSQL2008 server that also require migration to the SQL Managed Instance for the central Data Warehouse.

Task: Migrate SSIS from SQL Server 2008r2 to suitable environment, with a successful run of the package, verifying of the data and scheduling of package.

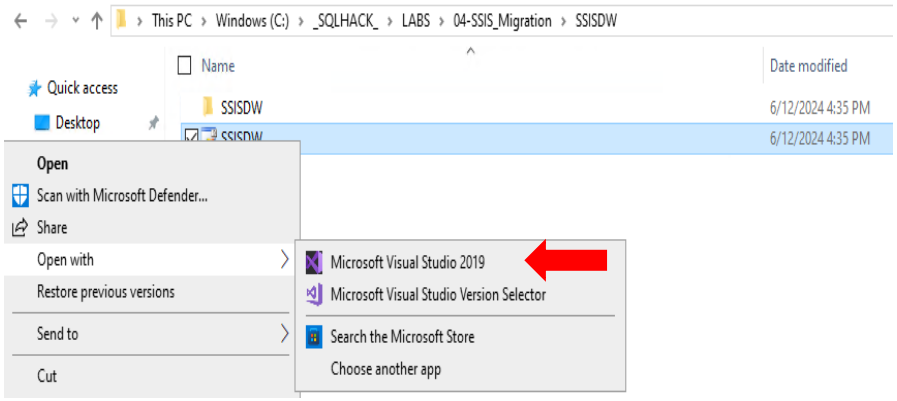
LAB INSTRUCTIONS

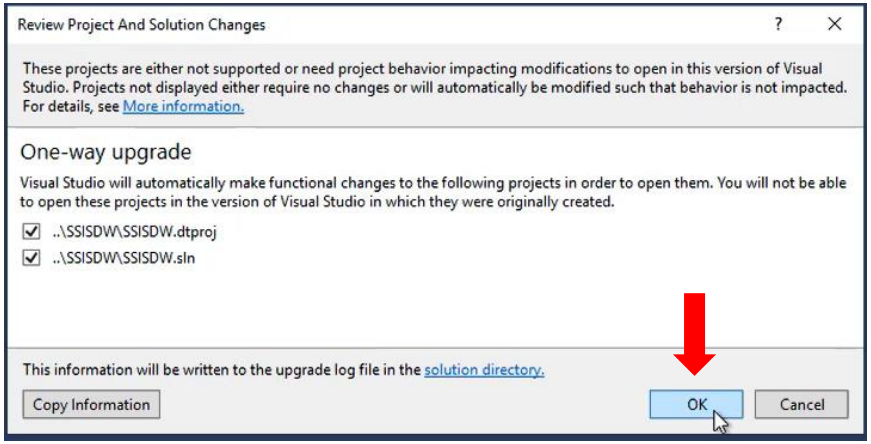
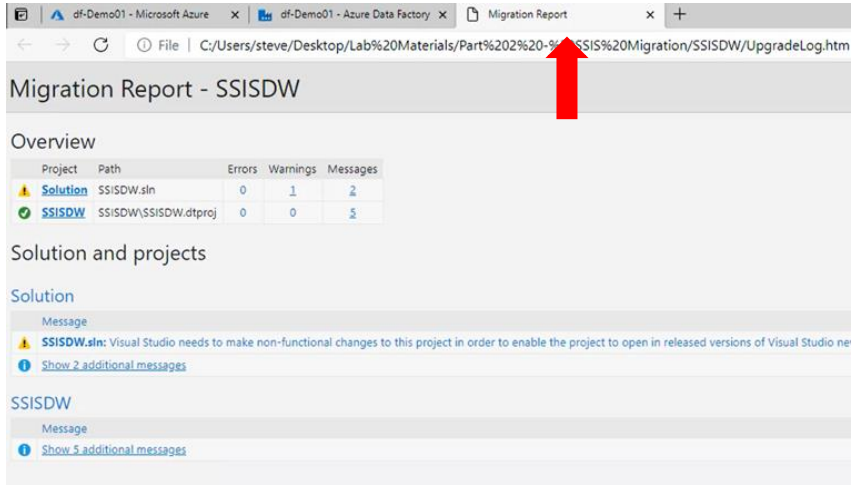
Time: 30 Mins

For Connection Strings and Passwords see sections LAB ENVIROMENT and APPENDIX at the end of this document

Stage 1 – Upgrade Package using the Upgrade Wizard.

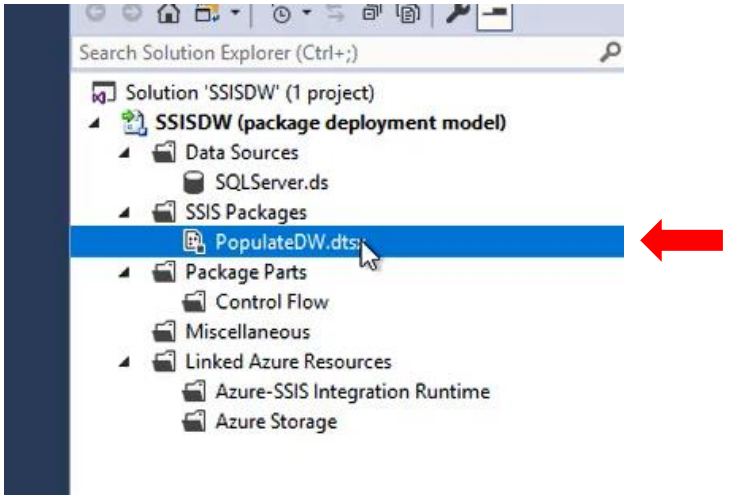
In this section we will be upgrading the Legacy SSIS package so that it can be migrated to Azure.

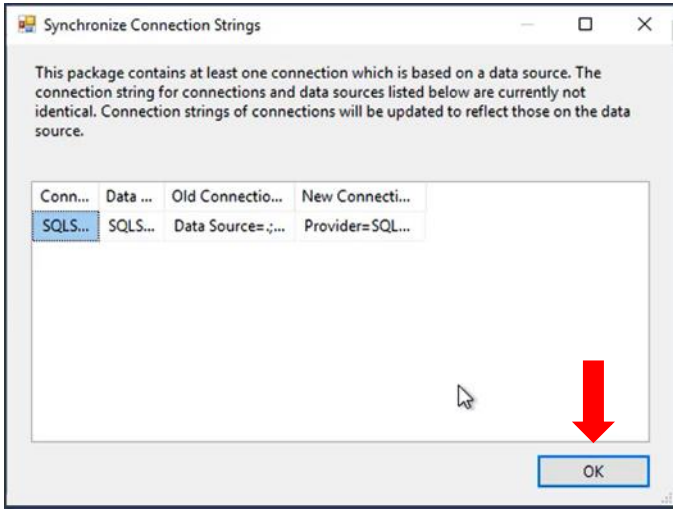
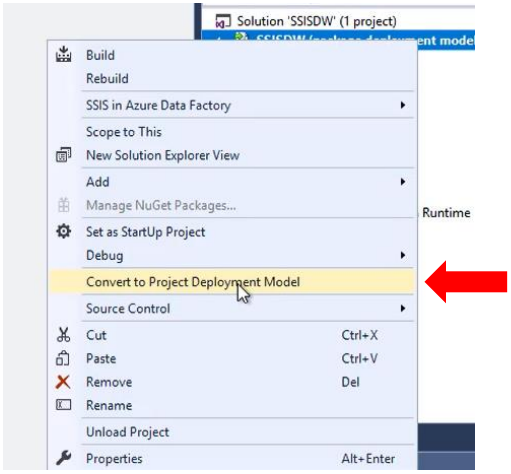
Narrative	Screenshot	Notes
Open the SQL 2008 Solution using Visual Studio 2019. Open the folder: **C:_SQLHACK_\LABS\Part 2 – SSIS Migration\SSISDW** Right click the **SSISDW.sln** solution file Open with **Visual Studio 2019**		You will need to RDP onto the TEAM virtual machine to complete this task. For connection details see **APPENDIX - TEAMXX VM RDP details**

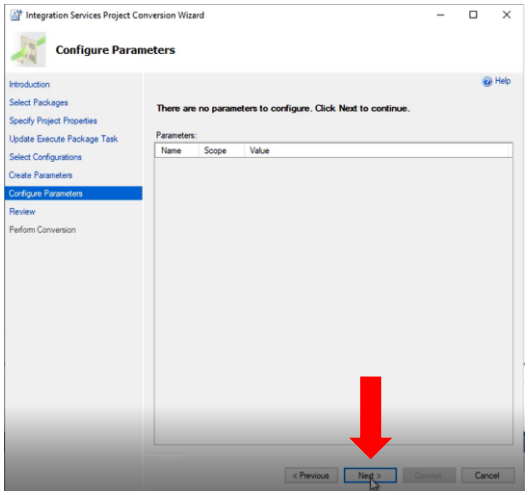
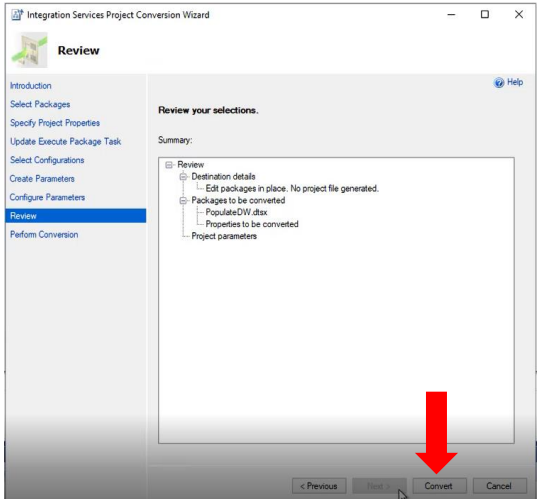
The package upgrade will require confirmation as this is a one-way process. You will be prompted with "**Review Project and Solution Changes**." **Click OK** to acknowledge this is a one-way process.		If you make a mistake there is a BACKUP folder which can be used to copy the sln and project files. Exit the process, replace the files and restart at Step 1.
A **Migration report** will be presented. **Read and close the Migration report** – No further action is required.		If you make a mistake there is a BACKUP folder which can be used to copy the sln and project files. Exit the process, replace the files and restart at Step 1.

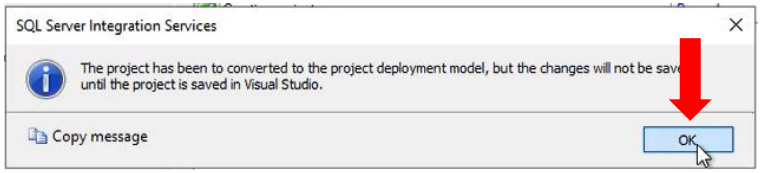
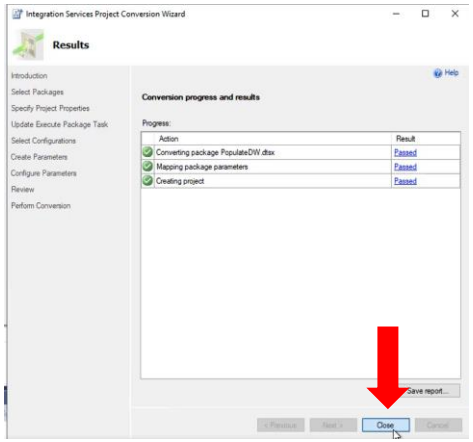
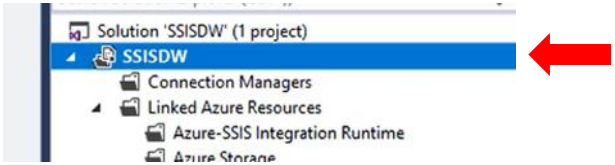
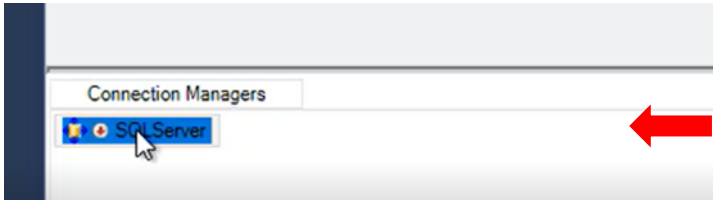
Stage 2 – Convert to Project Deployment mode & update connection string.

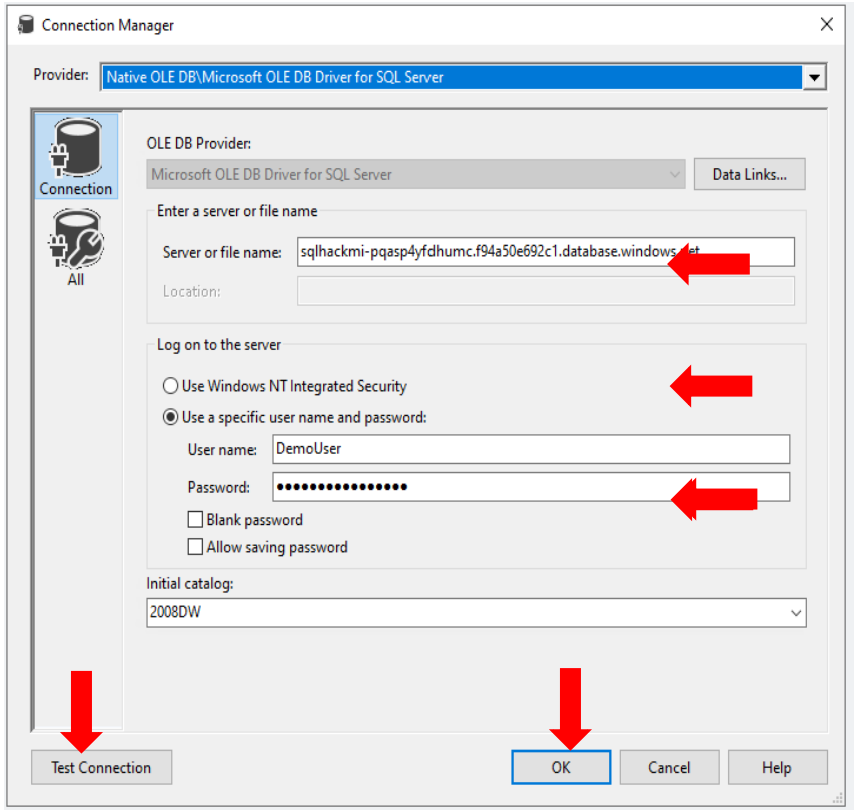
In this section we will be converting the DTSX package into a Project Deployment model and correcting the DTSX package connection strings to use the new SQL Server Managed Instance using Visual Studio 2019.

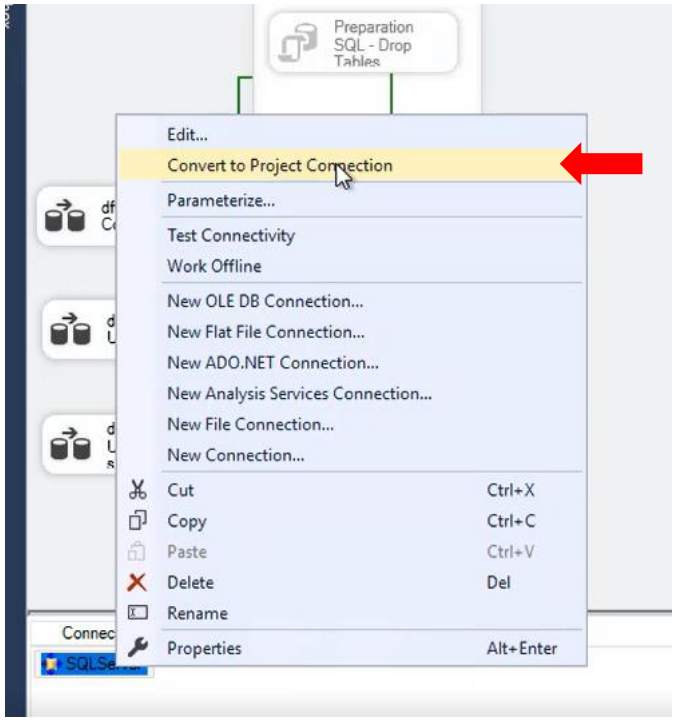
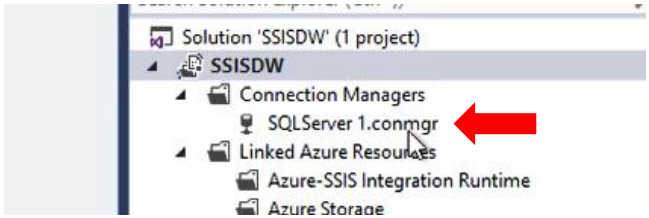
Narrative	Screenshot	Notes
Now the Solution is upgraded, it will be open in Visual Studio 2019. In Solution Explorer: **Double Click PopulateDW.dtsx** to open it.		If Visual Studio 2019 is not open, please confirm Stage 1 has been completed: Open the folder: **C:_SQLHACK_\LABS\Part 2 – SSIS Migration\SSISDW** Right click the **SSISDW.sln** solution file Open with **Visual Studio 2019**

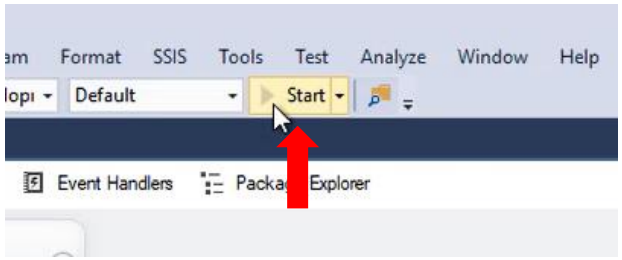
You will be prompted to Synchronise connection strings: **Click OK** to acknowledge connection strings will be updated.		
The SSIS package will require conversion to a Project Deployment Model. • **Right Click SSISDW** (package deployment model) • **Select Convert to Project Deployment Model**		

The Project Conversion Wizard will begin. Accept all the defaults on each page (until the review tab): • **Click Next** • **Click Next** • **Click Next** • **Click Next** • **Click Next** • **Click Next**		
The Project Conversion Wizard is ready. **Click Convert** to confirm your selections and start conversion.		Conversion should take a few moments.

The Project Conversion is complete. **Click Ok** to Acknowledge the SQL Server Integration Services pop-up.		
The Project Conversion is complete. **Click Close** and **Save** the package.		
Verify the Project Conversion has completed successfully. **Check the Package name no longer says, "Package Deployment Model."**		
Correct the Connection Managers. **Double Click** the **SQL Server Connection** in the **Connection Managers**.		

Within the Connection Manager, update to the new connection details. • **Server Name: (See Appendix - Target SQL Server)** • **Authentication: SQL Server Authentication** • **User Name: (See Appendix - Target SQL Server)** • **Password: (See Appendix - Target SQL Server)** • **Select or Enter Database name: 2008DW** Once the settings above are complete. ▪ **Click Test Connection** to test the connection. ▪ **Click OK** to save.		Please See Appendix - Target SQL Server within this document for full details on the connection settings.

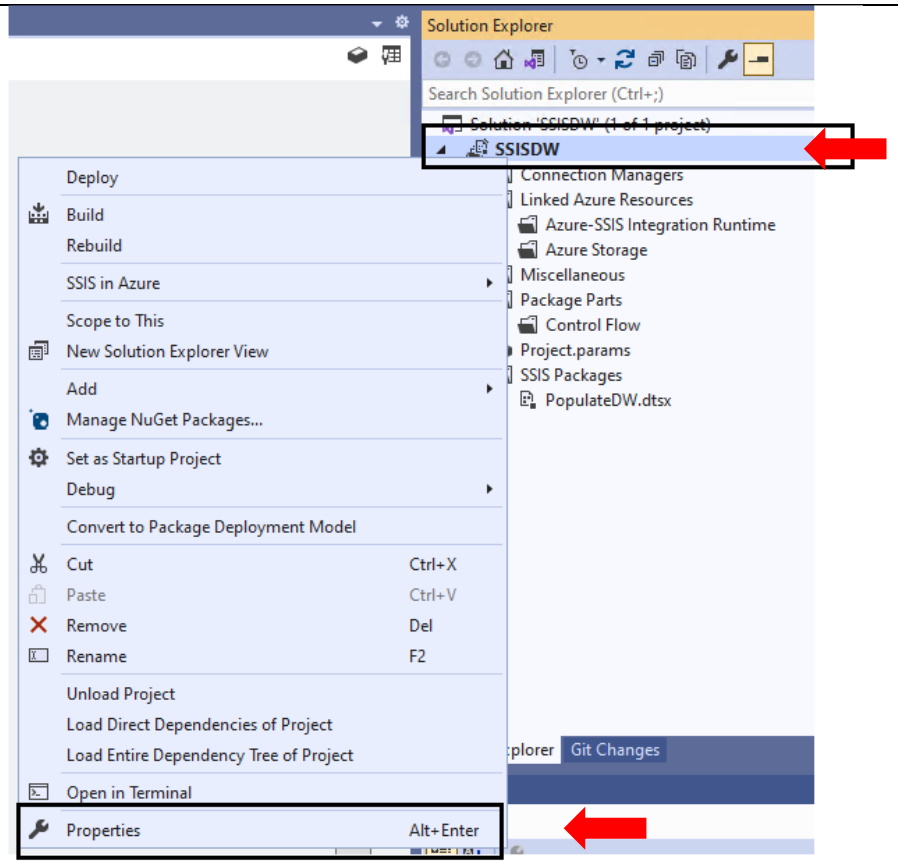
Convert the connection to a project connection. • **Right Click** Connection. • **Click Convert** to Project Connection.		
Verify you now have a project connection. **Check Connection Manager** has the **SQL Server 1.conmgr.**		

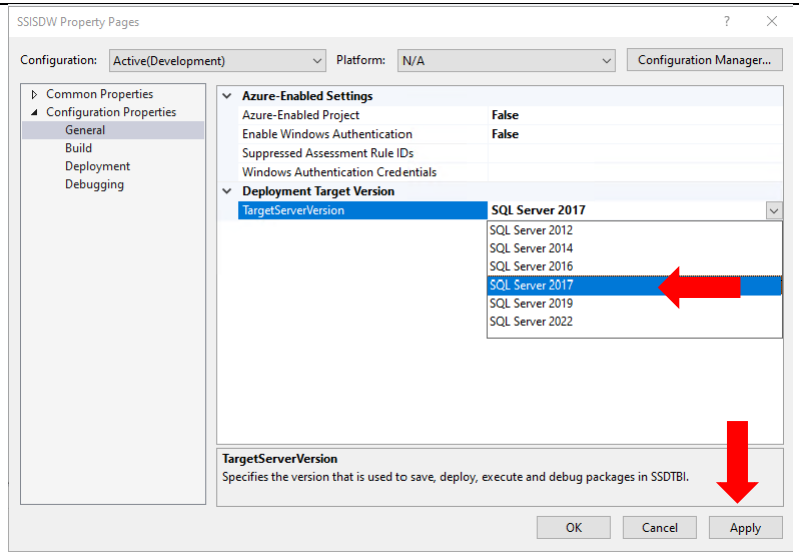
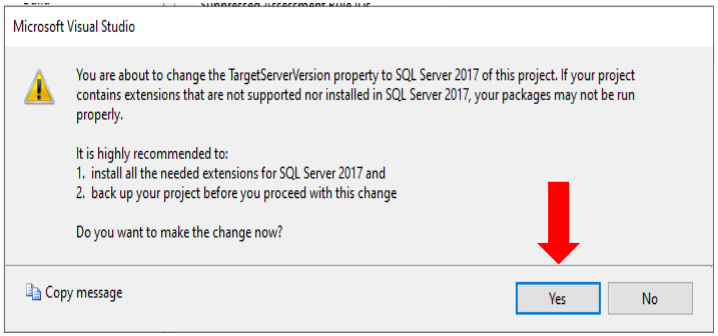
Test the package with the new connection manager. From the **Command bar**, **select Start** to Test the package.		

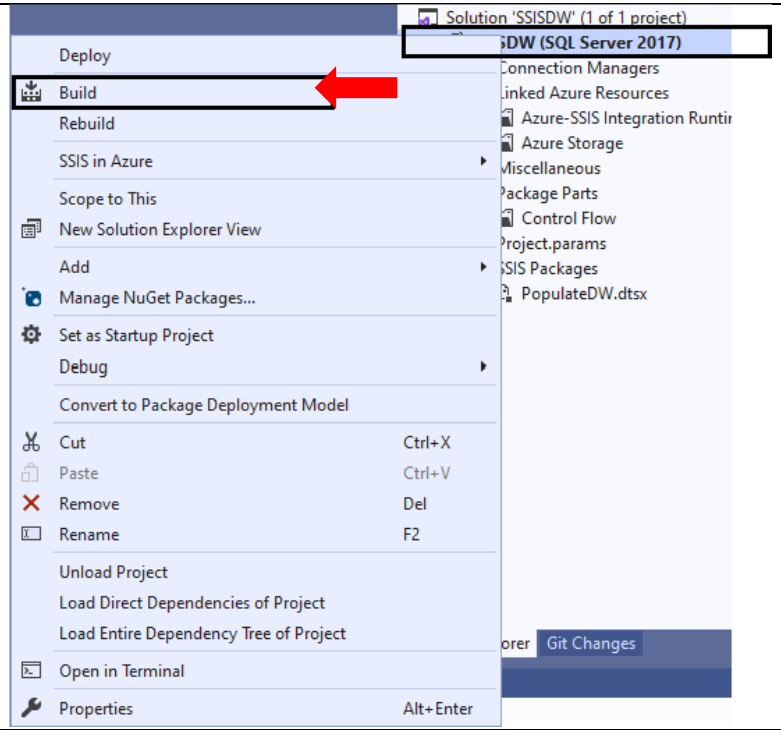
Stage 3 – Deploy Package to the SSISDB on the Managed Instance

In this section we will be deploying the fixed package onto the SSIS integration runtime and SSISDB held within the Managed Instance.

Narrative	Screenshot	Notes

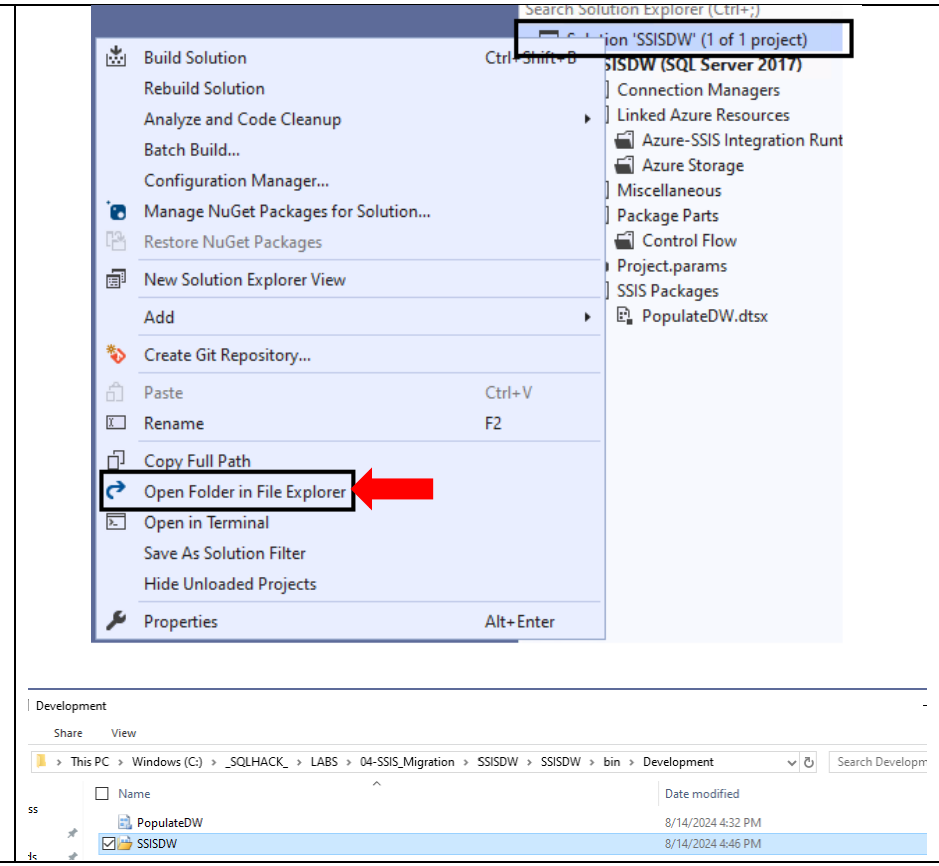
Set the Target Server version to SQL Server 2017 as SQL server 2022 is not yet supported. In Solution Explorer: **Right Click SSISDW package** and **select properties.**		Please ensure you have completed Stage 1 and Stage 2 successfully.

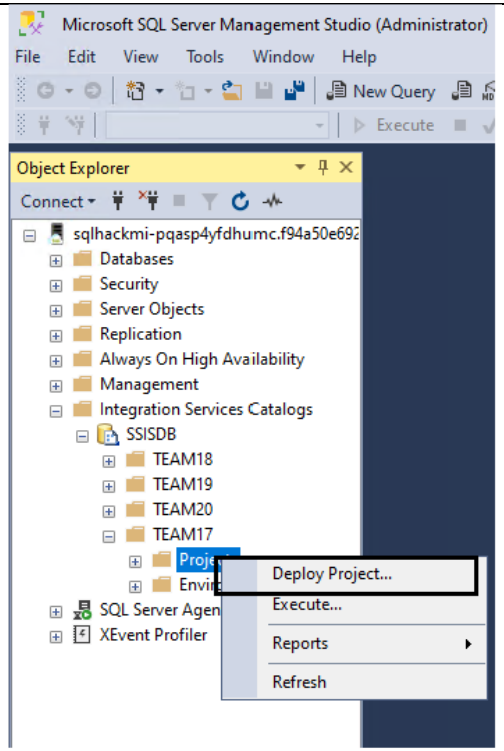
In the SSISDW property page. • **Select "General"** • **Change** "TargetServerVersion" to **SQL Server 2017** • **Click Apply**		
Click Yes to complete setting the SQL Target Version.		

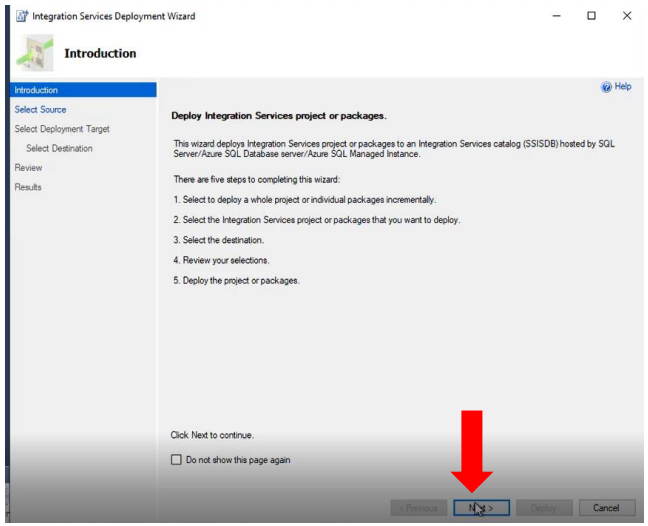
We now need to build the solution before deploy. **Right Click SSISDW package solution and select Build.**		

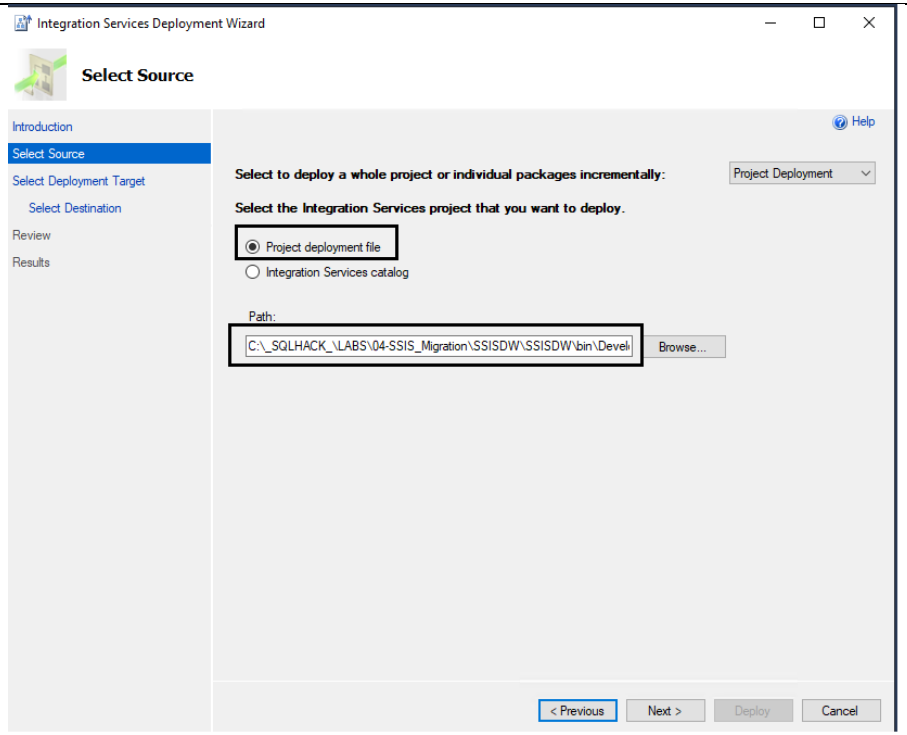
Go to Project solution File Explorer and

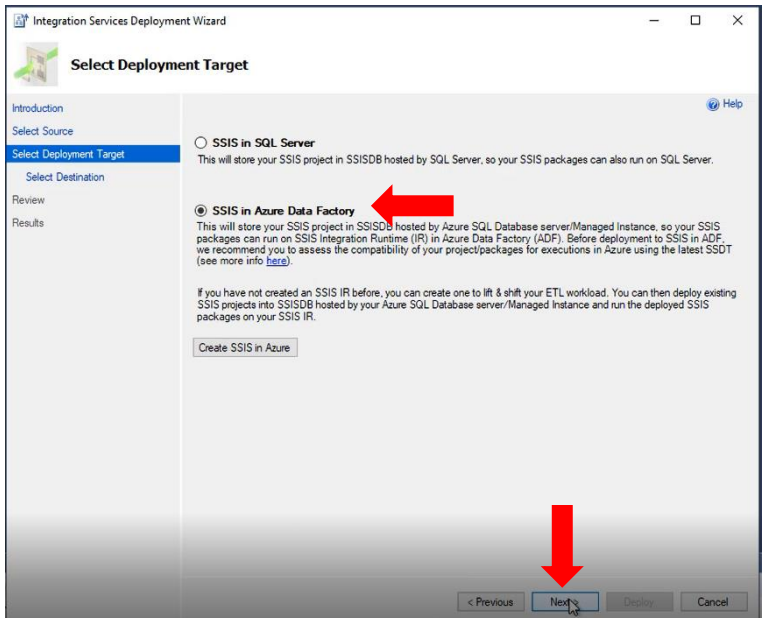
look for File Path as 'C:_SQLHACK_\LABS\04-SSIS_Migration\SSISDW\SSISDW\bin\Development'

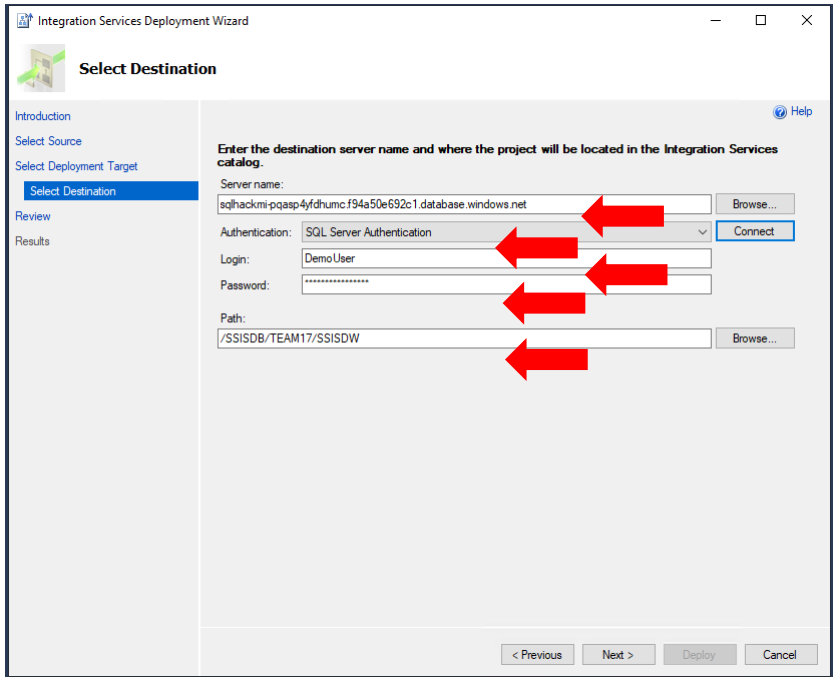


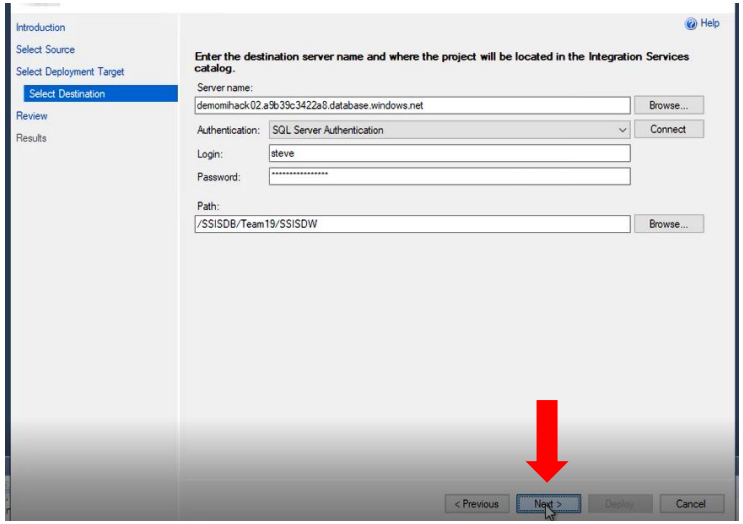
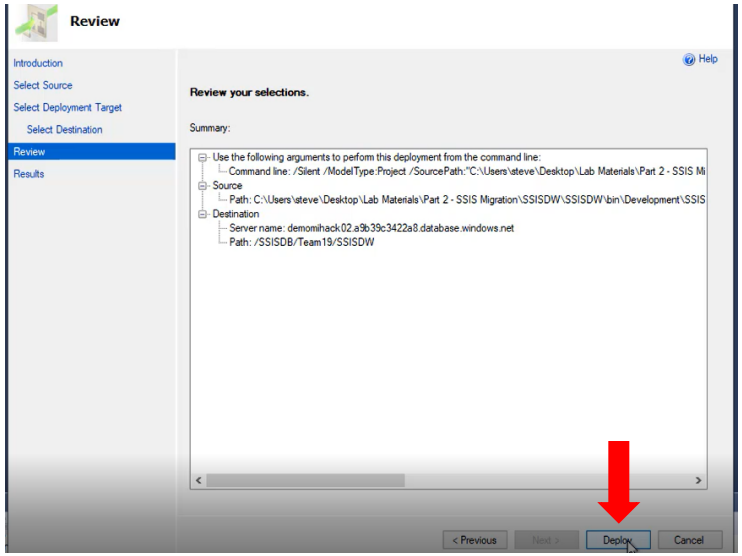
Using SQL Server Management Studio (SSMS), connect to the SQL Server Managed Instance. **Server Name: (See Appendix - Target SQL Server)** • **Authentication: SQL Server Authentication** • **User Name: (See Appendix - Target SQL Server)** • **Password: (See Appendix - Target SQL Server)** In **SSMS, navigate to Integration Service Catalogs SSISB. Create Folder as TEAM folder and Right Click on Projects**		

Integration Services Deployment Wizard will be started. **Click Next** to acknowledge introduction.		

Choose Project Deployment File And provide ISPAC file Path as **'C:_SQLHACK_\LABS\04-SSIS_Migration\SSISDW\SSISDW\bin\Development'** **Click on Next**		

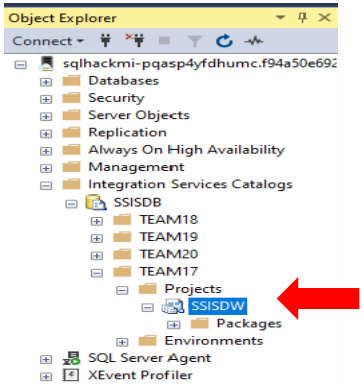
Integration Services Deployment Wizard. • **Select SSIS in Azure Data Factory.** • **Click Next.**		

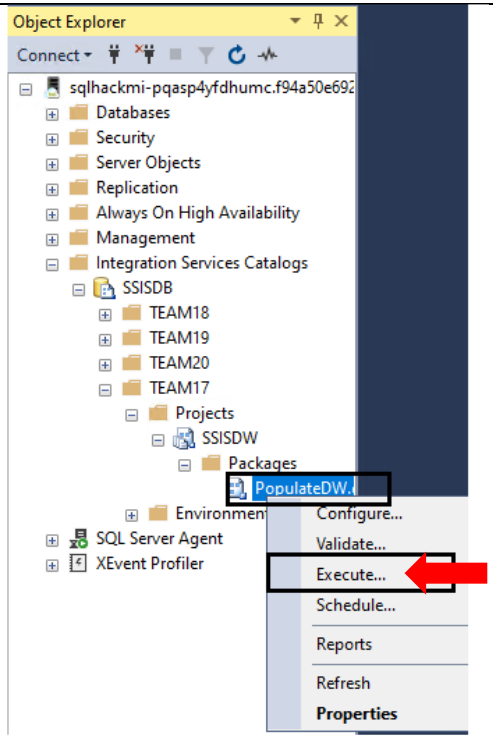
Within the Select Destination tab with the destination details: • **Server Name: (See Appendix - Target SQL Server)** • **Authentication: SQL Server Authentication** • **User Name: (See Appendix - Target SQL Server)** • **Password: (See Appendix - Target SQL Server)** • **Path: Select Browse and Add your TEAM name as a Folder. Example if you are in TEAM 1, enter a folder name of TEAM01.**		Please See Appendix - Target SQL Server within this document for full details on the connection settings.

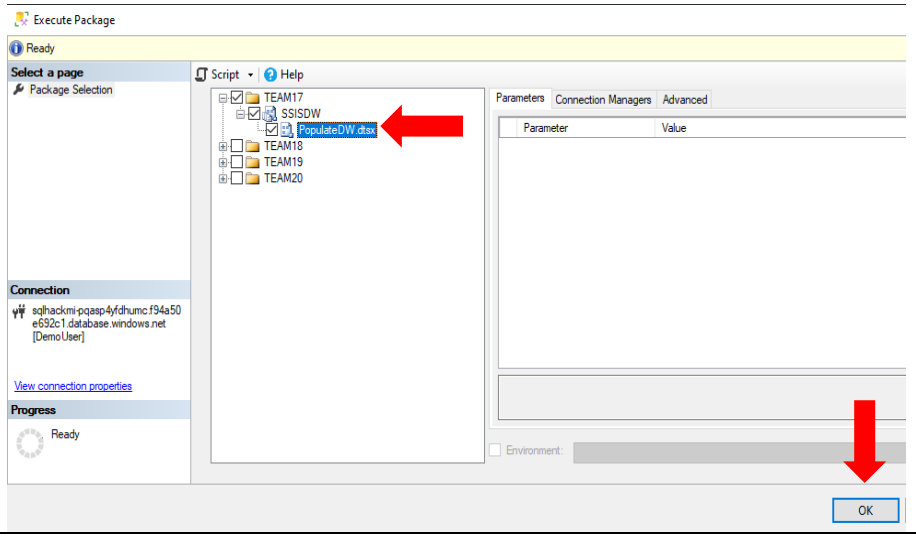
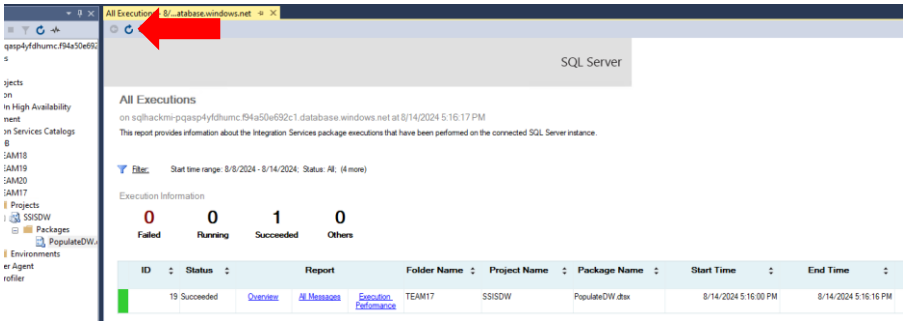
Check the Select Destination details. **Check details** & **click Next** to continue.		
The package is now ready to deploy to the SSISDB on the Managed instance. • **Click Deploy** to begin deployment. • **Click Close** to confirm Successful Deployment.		

Stage 4 – Verify Deployment and test run package.

In this section we will be verifying the package has been deployed successfully to the Managed Instance and running the Package to ensure it is working correctly.

Narrative	Screenshot	Notes
Using SQL Server Management Studio (SSMS), connect to the SQL Server Managed Instance. In **SSMS, navigate to Integration Service Catalogs**: • **Select Projects** • **Select Your TEAM folder** **Verify** the **SSISDW** Package has been deployed.		For connection details please See Appendix - Target SQL Server.

Test the SISS Package migrate to the Managed Instance. Execute the SSIS Package • **Right Click** the **SSISDW** package • Select **Execute**		

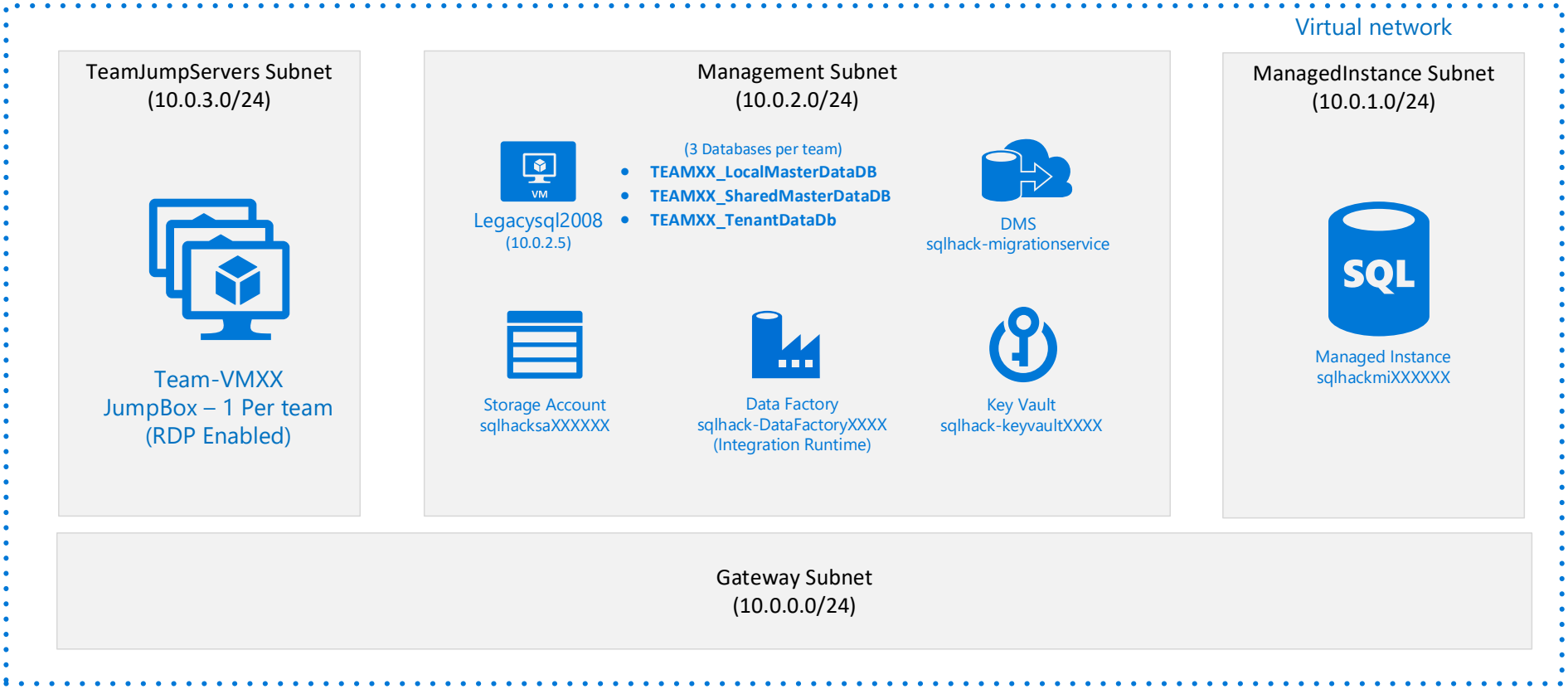
Test the SISS Package migrate to the Managed Instance. Within the Execute Package window: • Ensure the Package **PopulateDW.dtsx is selected.** • **Click OK**		
View the execution report once complete. You be notified that the Selected Packages have been queued to Execute. • **Click Yes** to view the Execution Report. Once the Execution report has loaded: • **Click refresh** until the package has completed. **Congratulations on successfully migrating and upgrading an SSIS package to Azure.**		

Optional Stage 5 – Schedule Package using SQL Server Agent

If you have time, schedule the package to run with a Job using SQL Server Agent

Note: No instructions provided for this task.

LAB ENVIROMENT



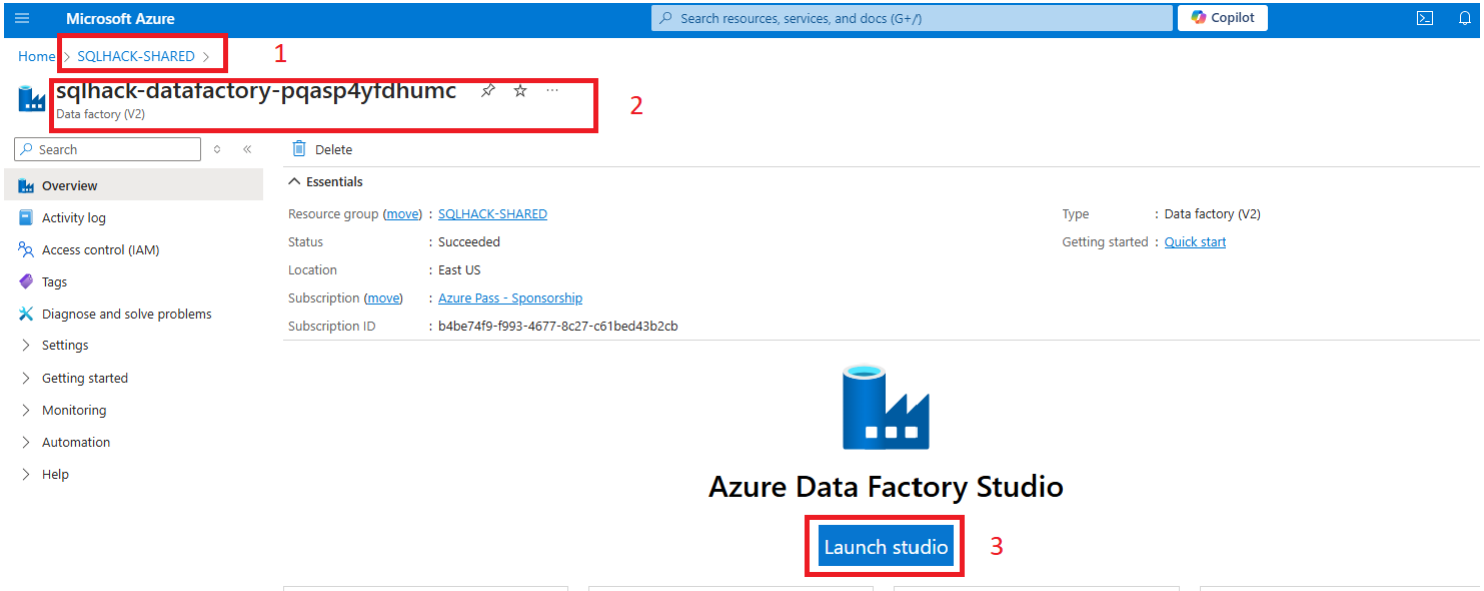
NOTE: There are 20 workshop environments using a SHARED source SQL Server and target Azure SQL Database Managed Instance. Please be respectful of only migrating your teams Databases and Logins.

Optional Stage 6 – How to Deploy Azure SSIS IR Manually

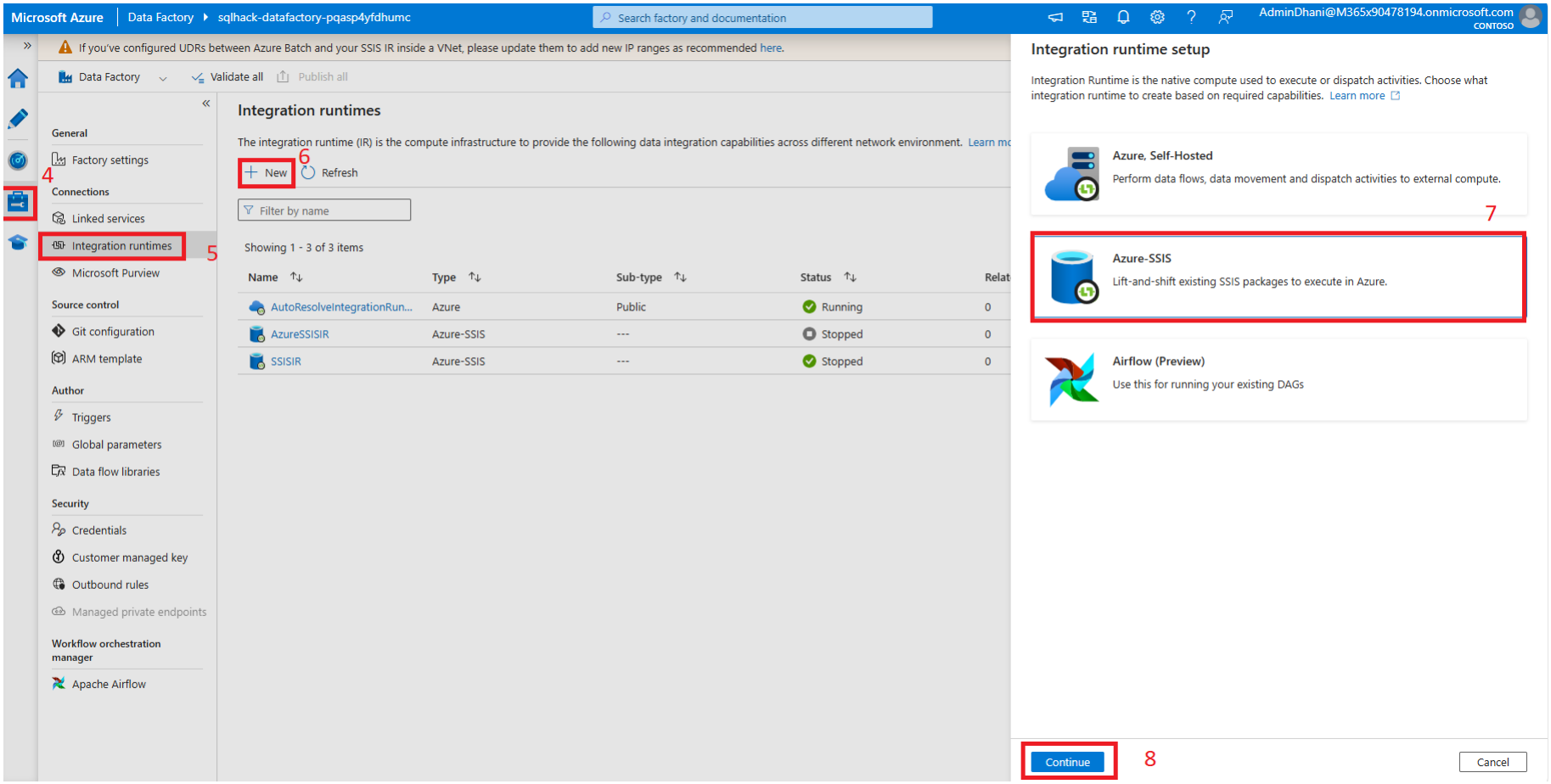
It's currently identified that Azure SSIS IR are not successfully deployed to be use in Lab due to right SQL MI FQDN are not correctly being passed as parameter value in environment deployment script. Deployment script fix in currently back and taken care with next artefacts release. Until then, this step is essential for CSA assigned for deliver to follow following steps and deploy Azure SSIS IR manually if "SSIS Migration topics" in scope for any hack or event.

Please note, without Actively running Azure SSIS IR no participant can deploy their SSIS packages into Azure SQL MI – SSIS catalog. Azure SSIS IR successful deployment can take 15 mins to 60 mins. Hence, its strong recommendation to CSA assigned for delivery should perform the following steps as their mandatory Lab environment setup action items.

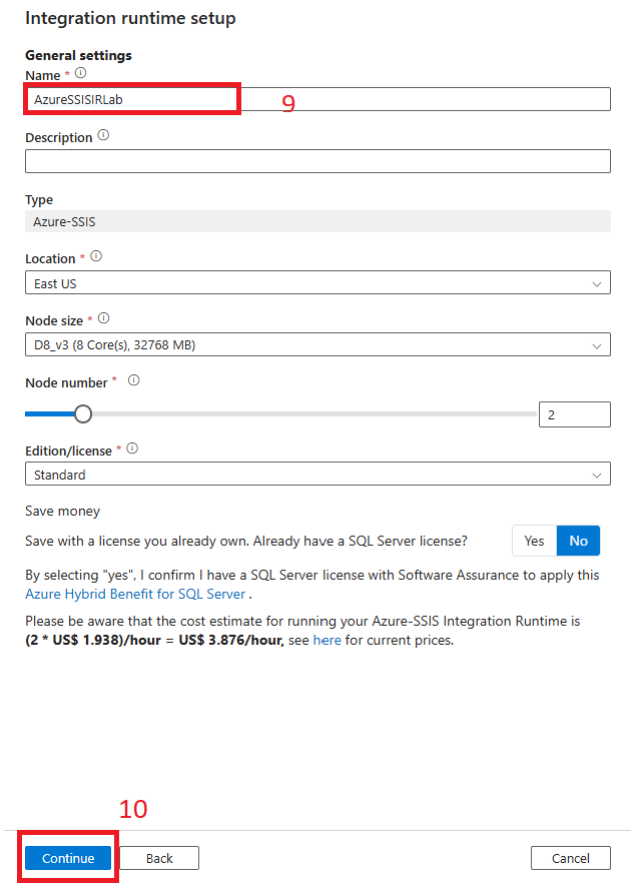
a. **Go to SQLHACK-SHARED → Data Factory Instance → Launch Studio**



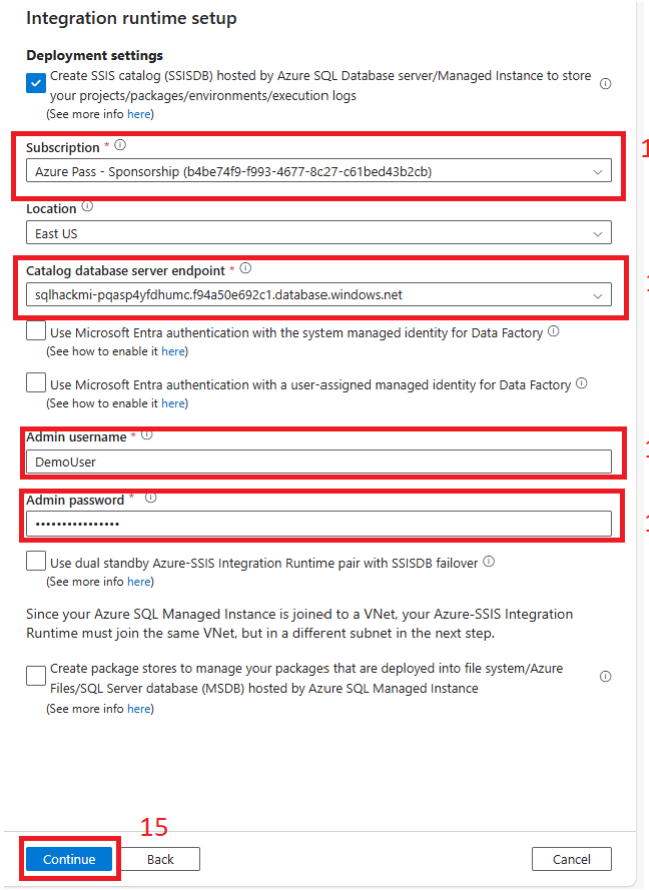
b. **In ADF Studio , Go to Manage Tab → Integration Runtimes →Click on New →Select Azure – SSIS →Click on Continue**



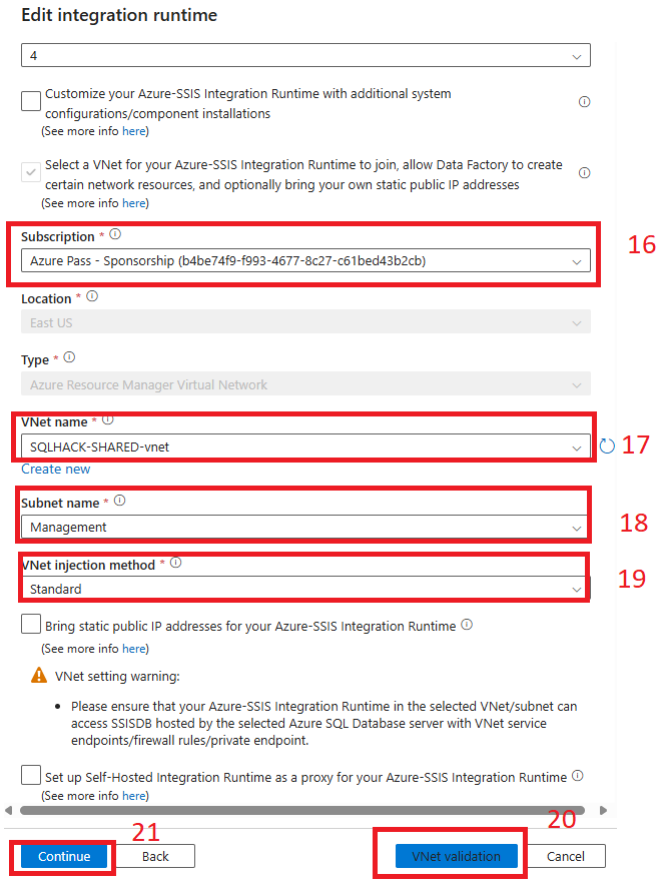
c. **Give some valid , Name → Click on Continue**



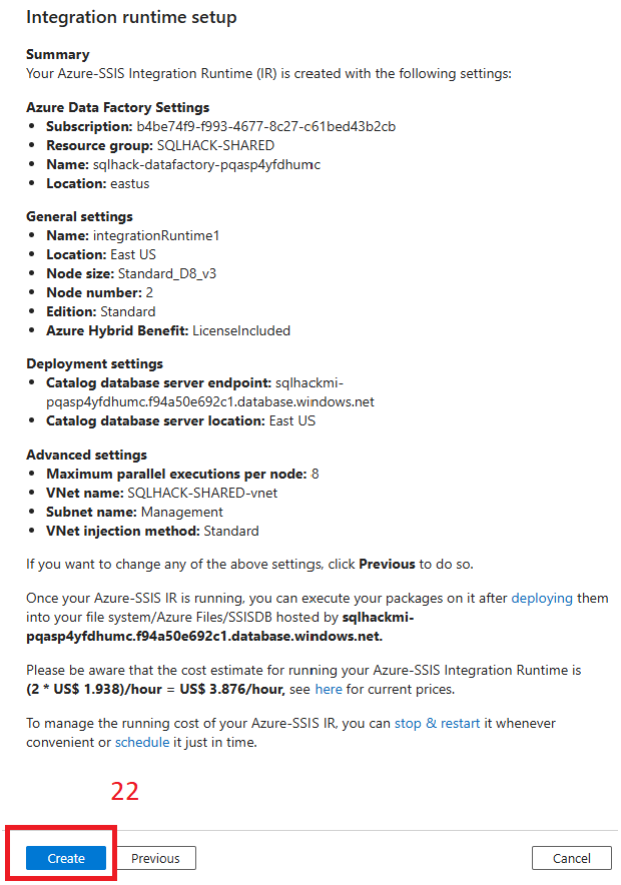
d. **Select Valid Azure Lab Subscription → Select Valid SQL MI Server Name: (See Appendix - Target SQL Server) → User Name: (See Appendix - Target SQL Server) → Password: (See Appendix - Target SQL Server)→ Click on Continue**



e. **Make sure valid Azure Lab scription selected → Select Vnet Name as "SQLHACK-SHARED-vnet"→ Subnet Name as "Management" →Vnet Injection Method as "Standard" → Click on "Vnet validation" → Click on Continue**



f. **And, finally under summary page click on "Create"**



g. **It takes from 15 to 60 minutes to Azure SSIS IR to show as running status.**

APPENDIX

Summary of Logins and Accounts Used

There are several different environments that you need to login/connect to during the labs. Sometimes you will need to login into the same environment with different accounts depending on what you are doing e.g., logging into SQL Server with a standard or sysadmin privileged account.

TEAMXX VM RDP details

Machine IP address (Use for RDP connection)	
Machine Name (Replace XX with Team number)	**vm-TEAMxx**
Win10 Username: (Use for RDP connection)	**Demouser**
Win10 Password: (Use for RDP connection)	**Demo@pass1234567**
Resource Group	**SQLHACK-TEAM-VMs**

Target SQL Server (Azure SQL Managed Instance)

Server Name	***SQL MI FQDN from Azure portal***
Resource Group	**SQLHACK-SHARED**
Sysadmin Login Name: (Use for Migrations)	**DemoUser**
Admin Login Password:	**Demo@pass1234567**