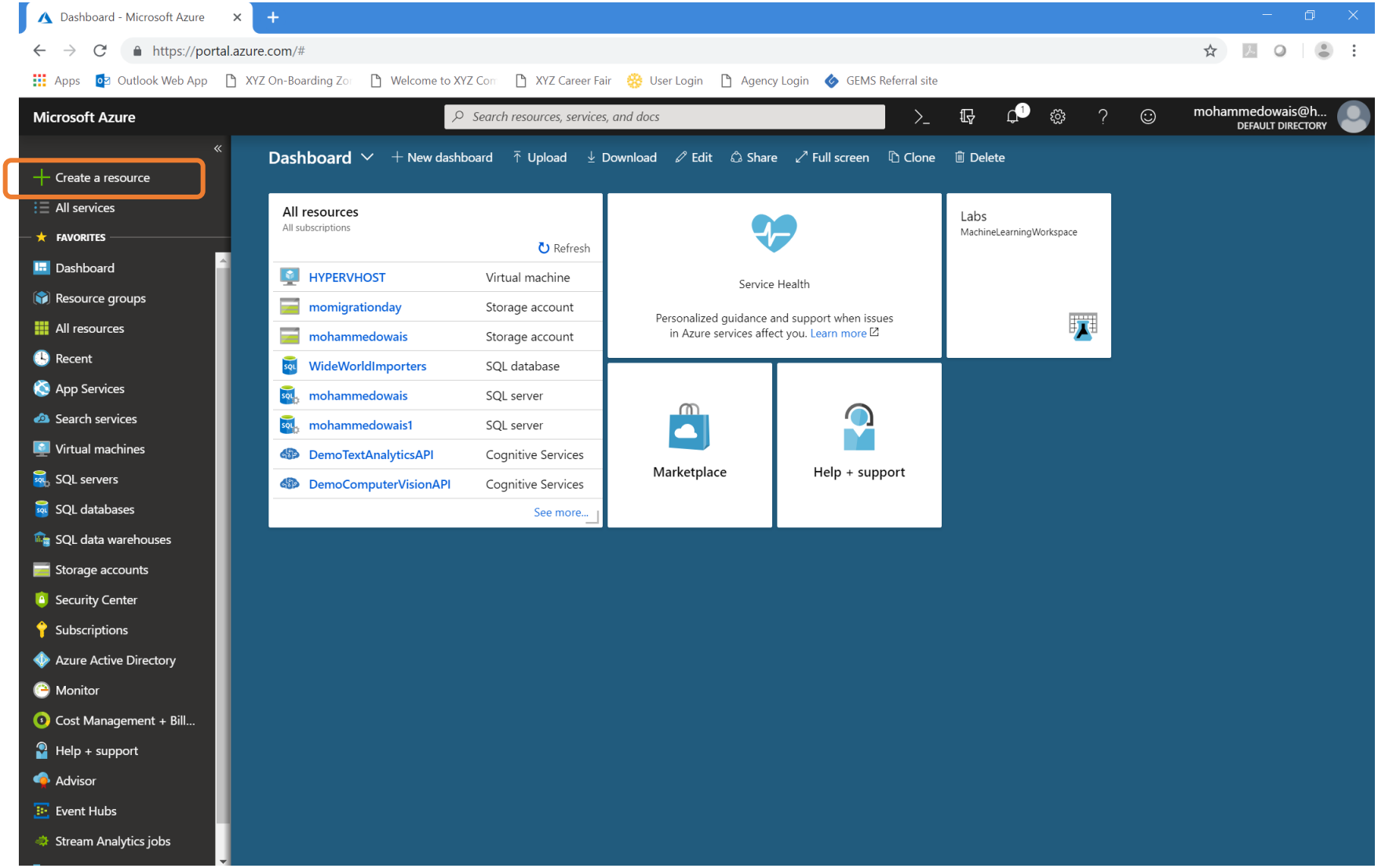
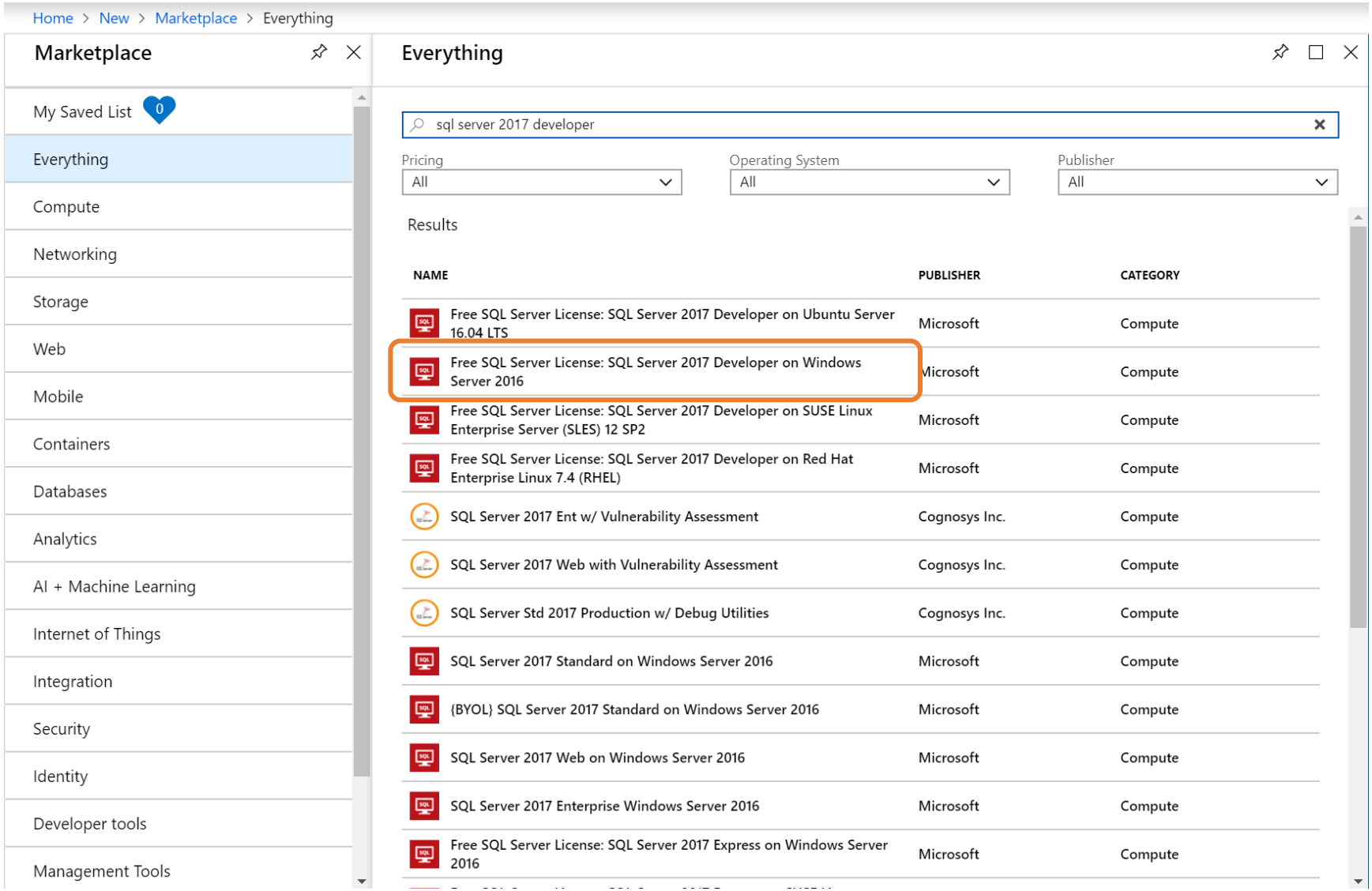
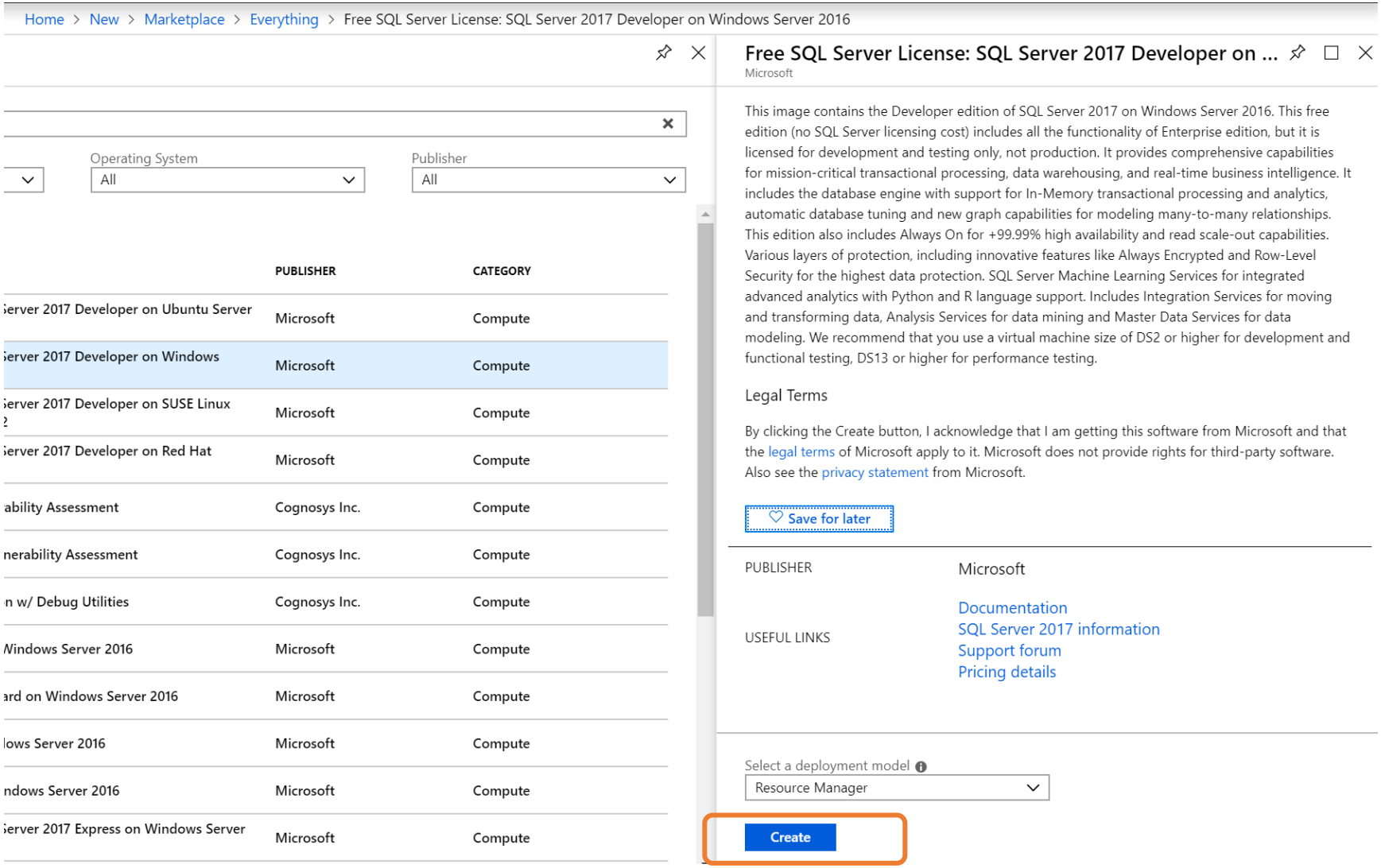
Click on “Create a Resource”



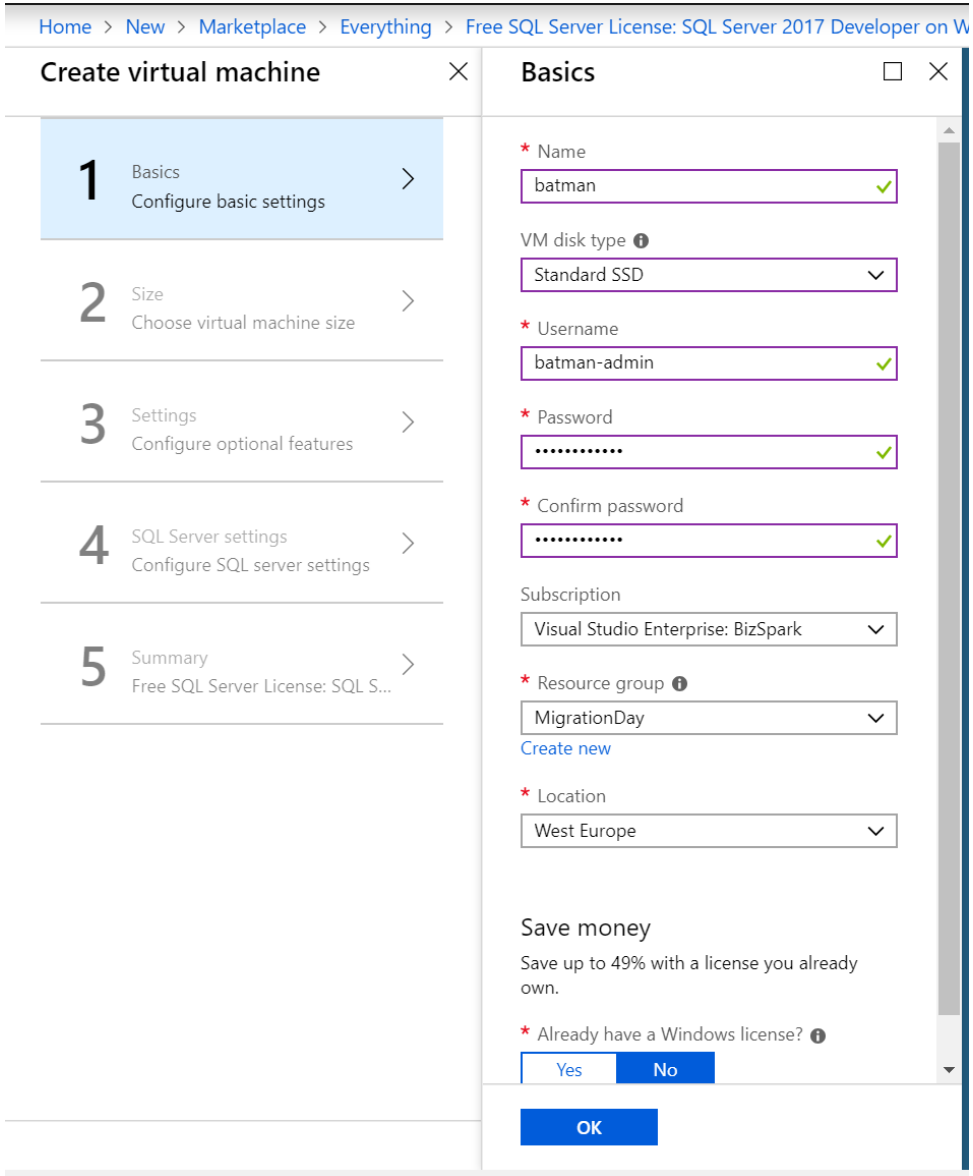
Search for “SQL Server 2017 Developer” and select **SQL Server 2017 Developer on Windows Server 2016** from the list (or on some flavor of linux if you are feeling particularly brave)



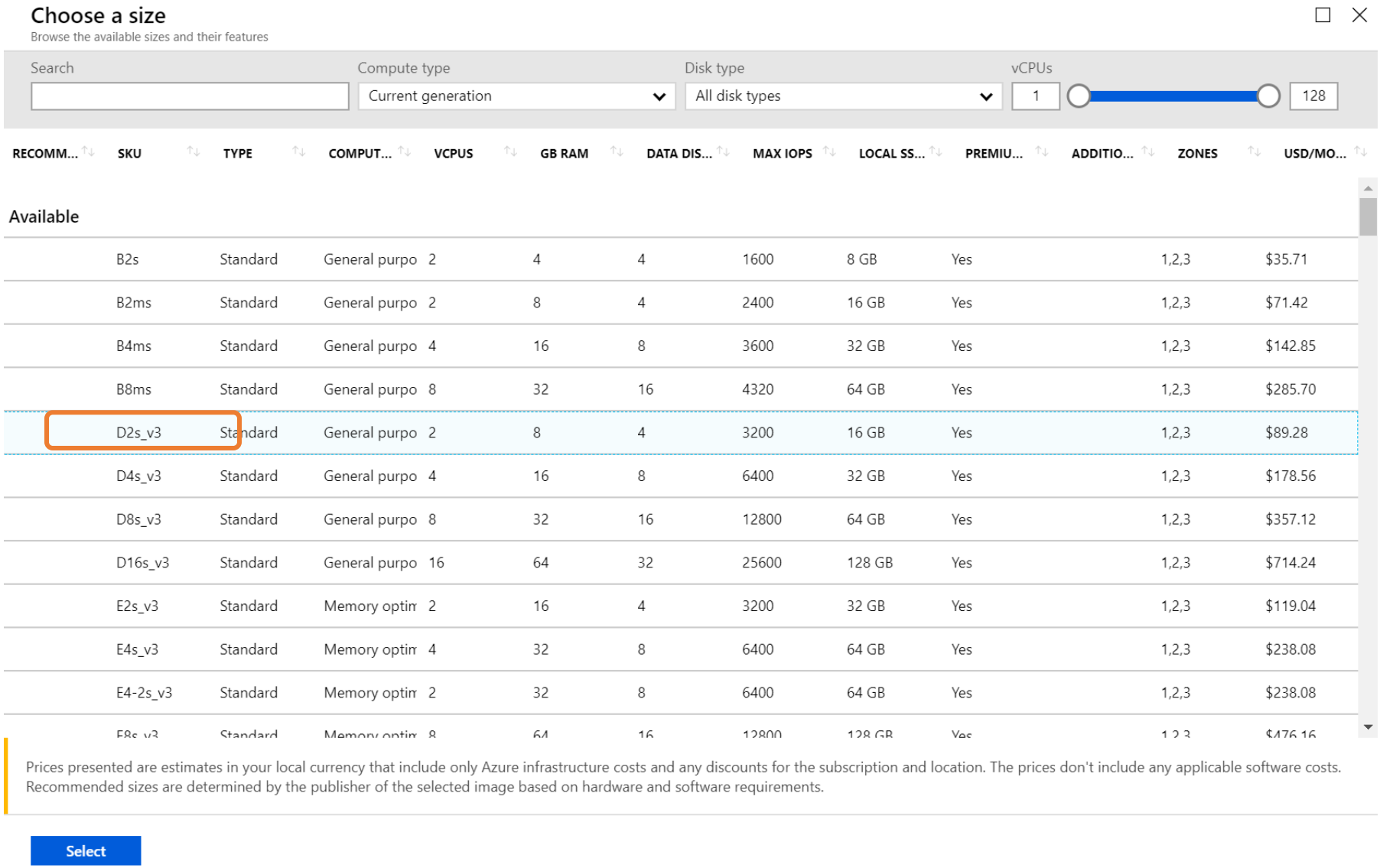
Click on the **Create button**



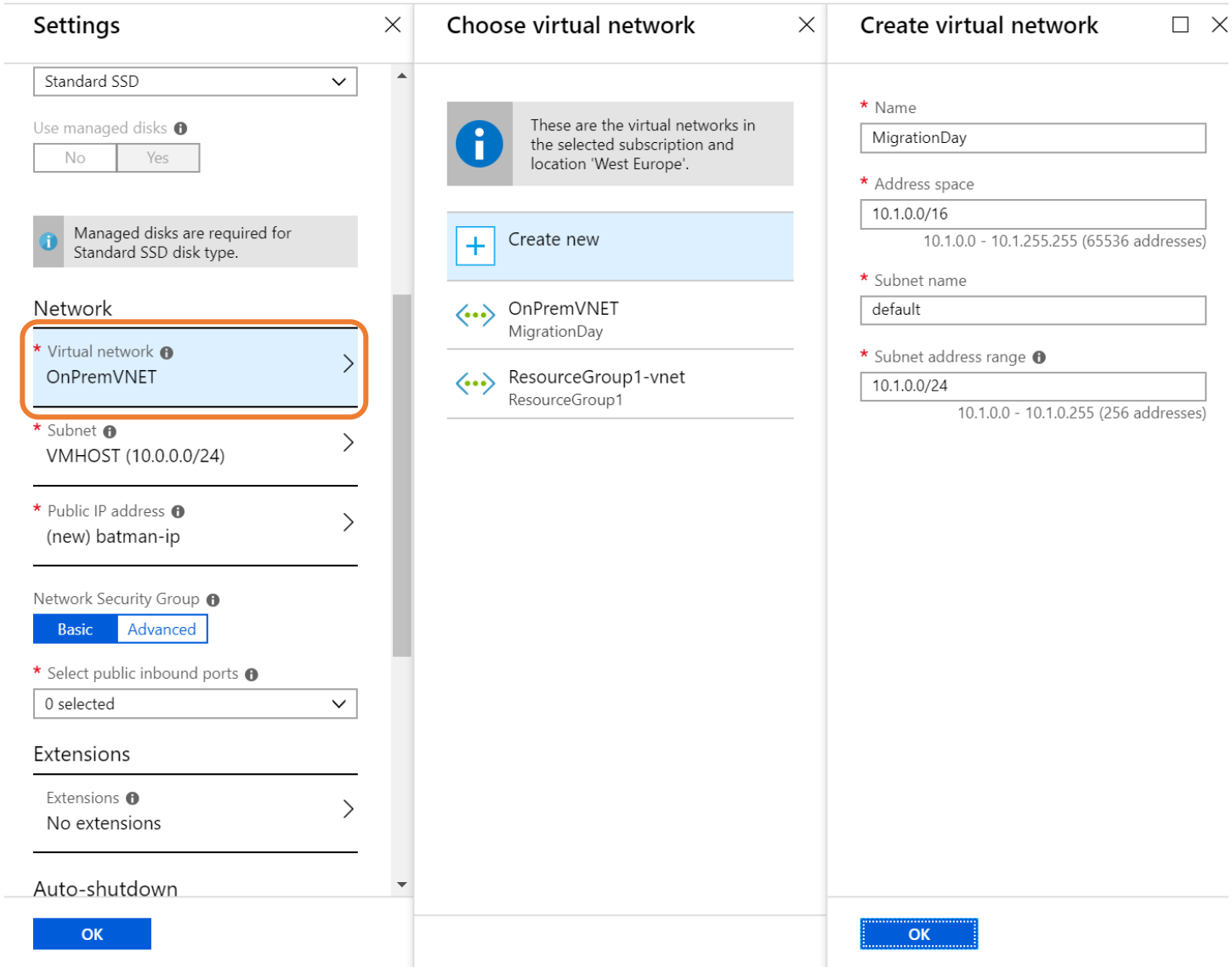
Fill out the form (Screen 1). Select Standard SSD for test servers



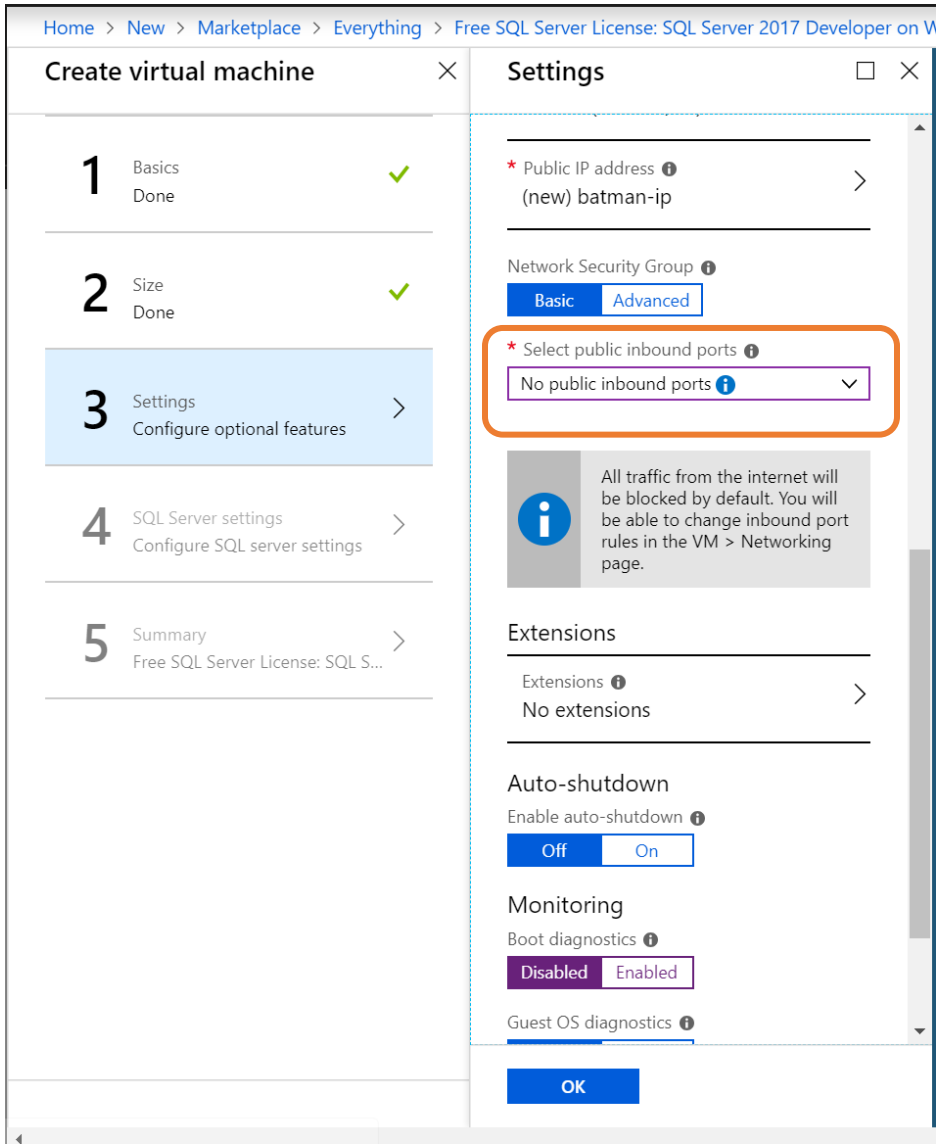
For server size, choose D2S_v3



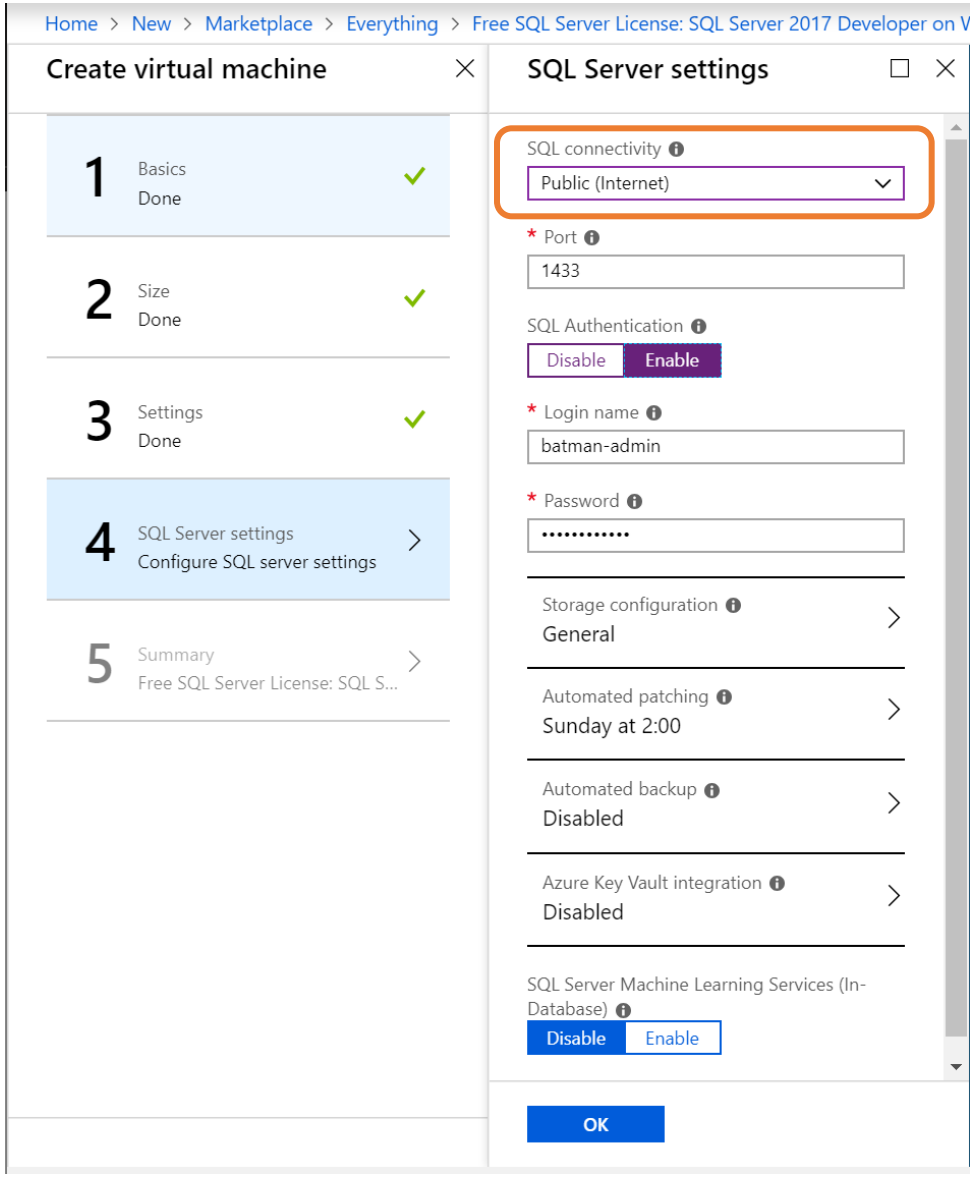
On the settings page, add a new VNET to keep the database server on a separate network



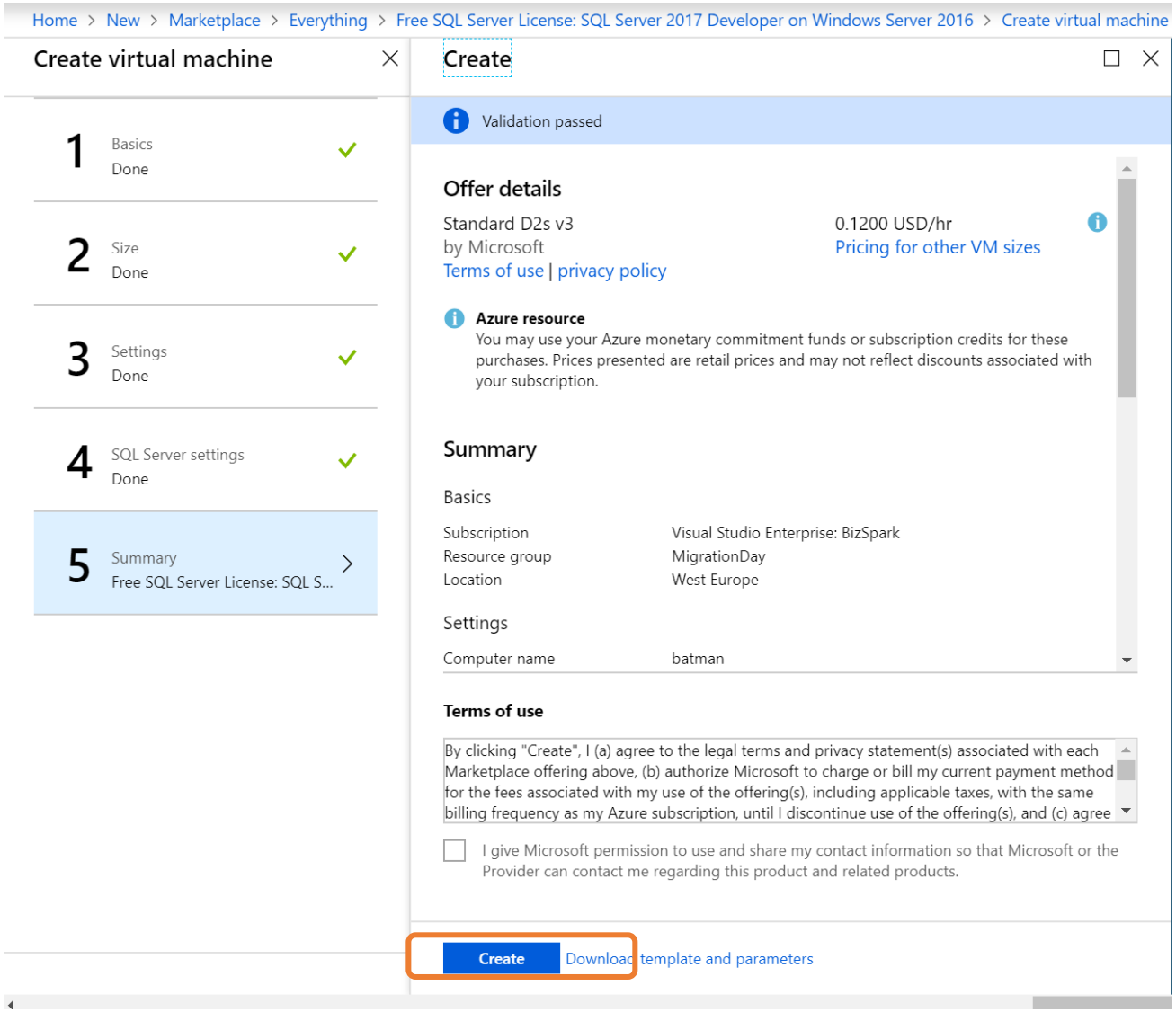
Rest of the settings (disabled **diagnostics** and selected **no public inbound ports**)



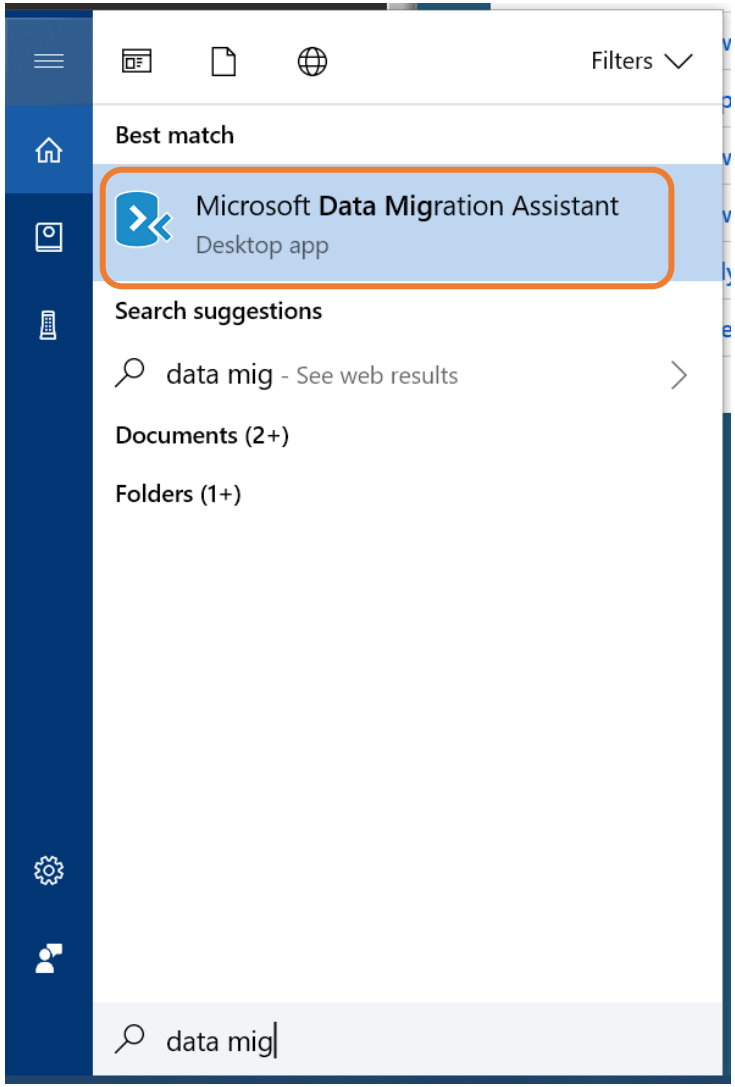
On SQL Server settings page, change **connectivity** to Public (DO NOT USE THIS SETTING IN PRODUCTION!) and enable SQL Authentication



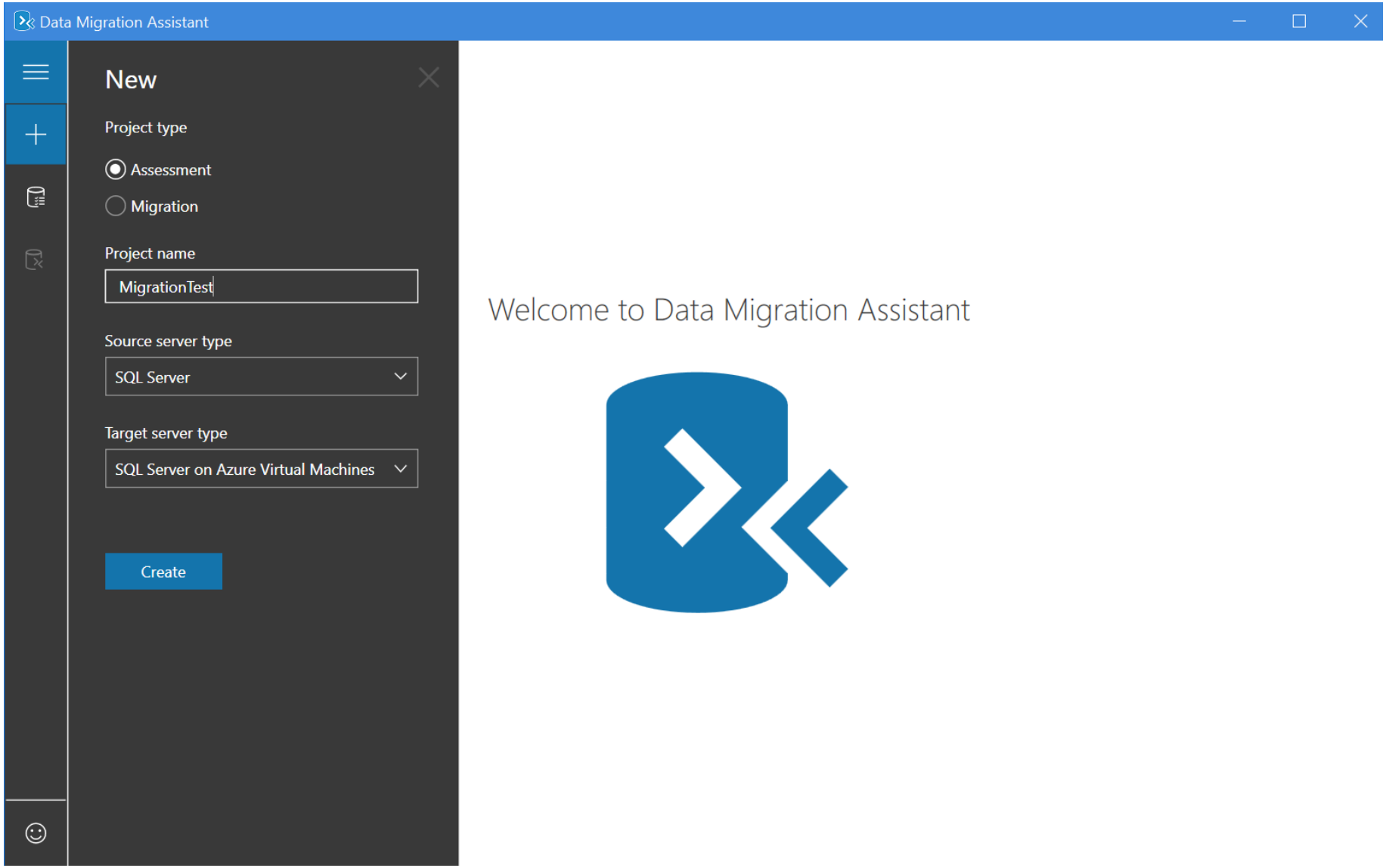
On the summary screen, click on the **Create** button. Takes about 15 minutes to create the server.



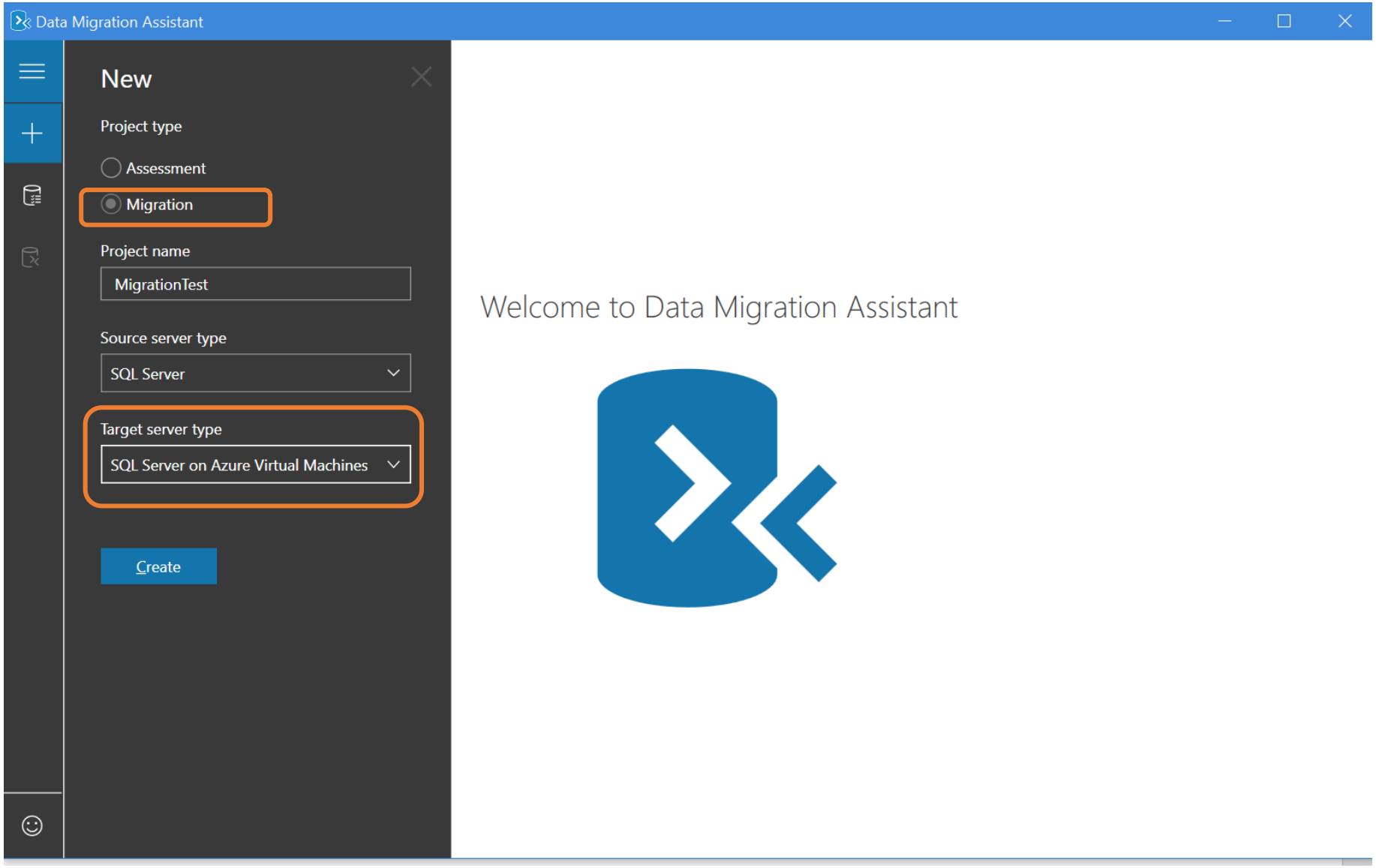
Install the Data Migration Assistant on the the source server (you can also install it on another machine which has access to both servers). Download it from here: https://www.microsoft.com/en-us/download/details.aspx?id=53595 . Open the DMA once it has installed.



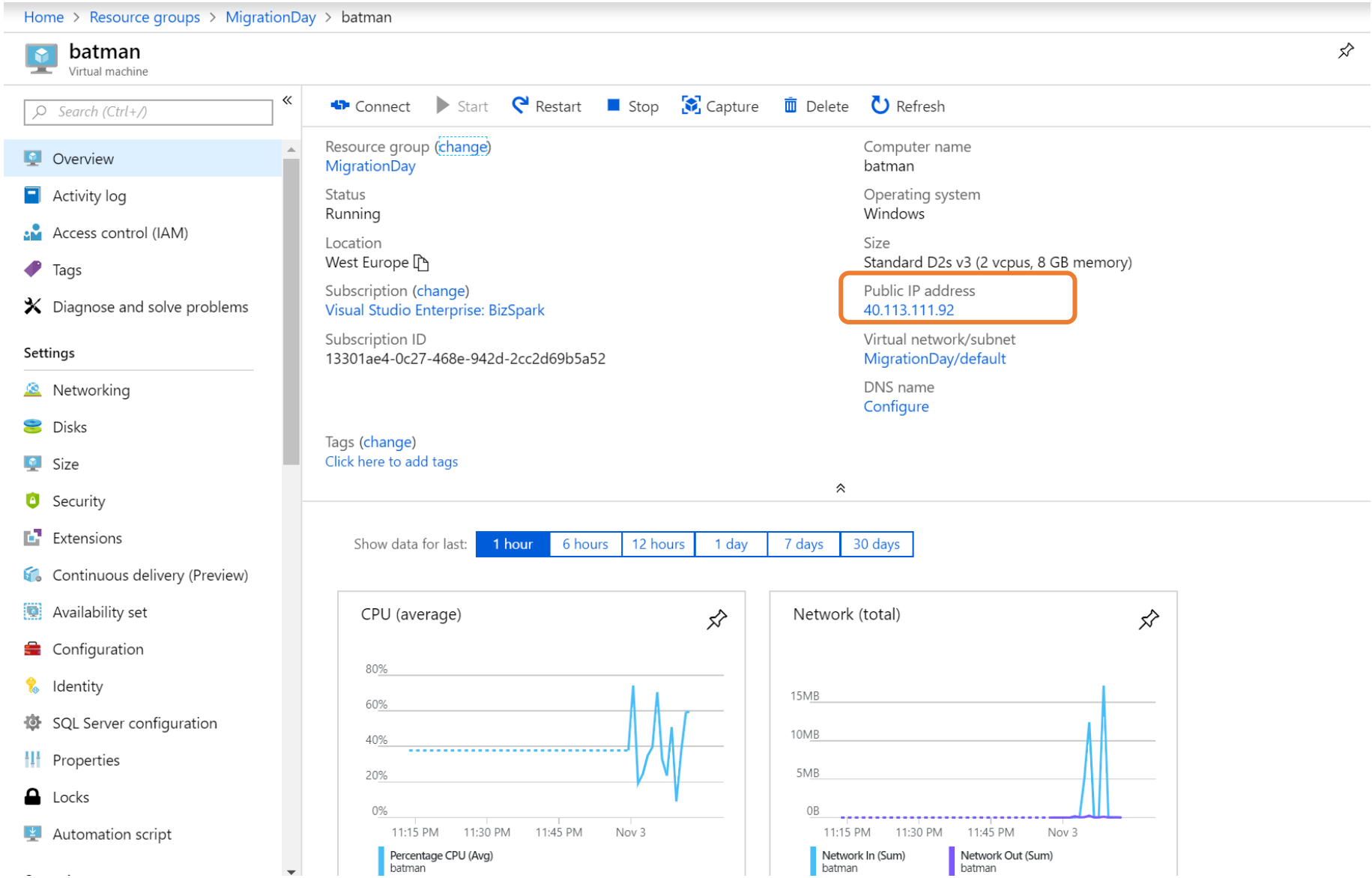
First create an assessment (optional). Make sure you select the correct target server type as SQL Server on Azure Virtual Machine



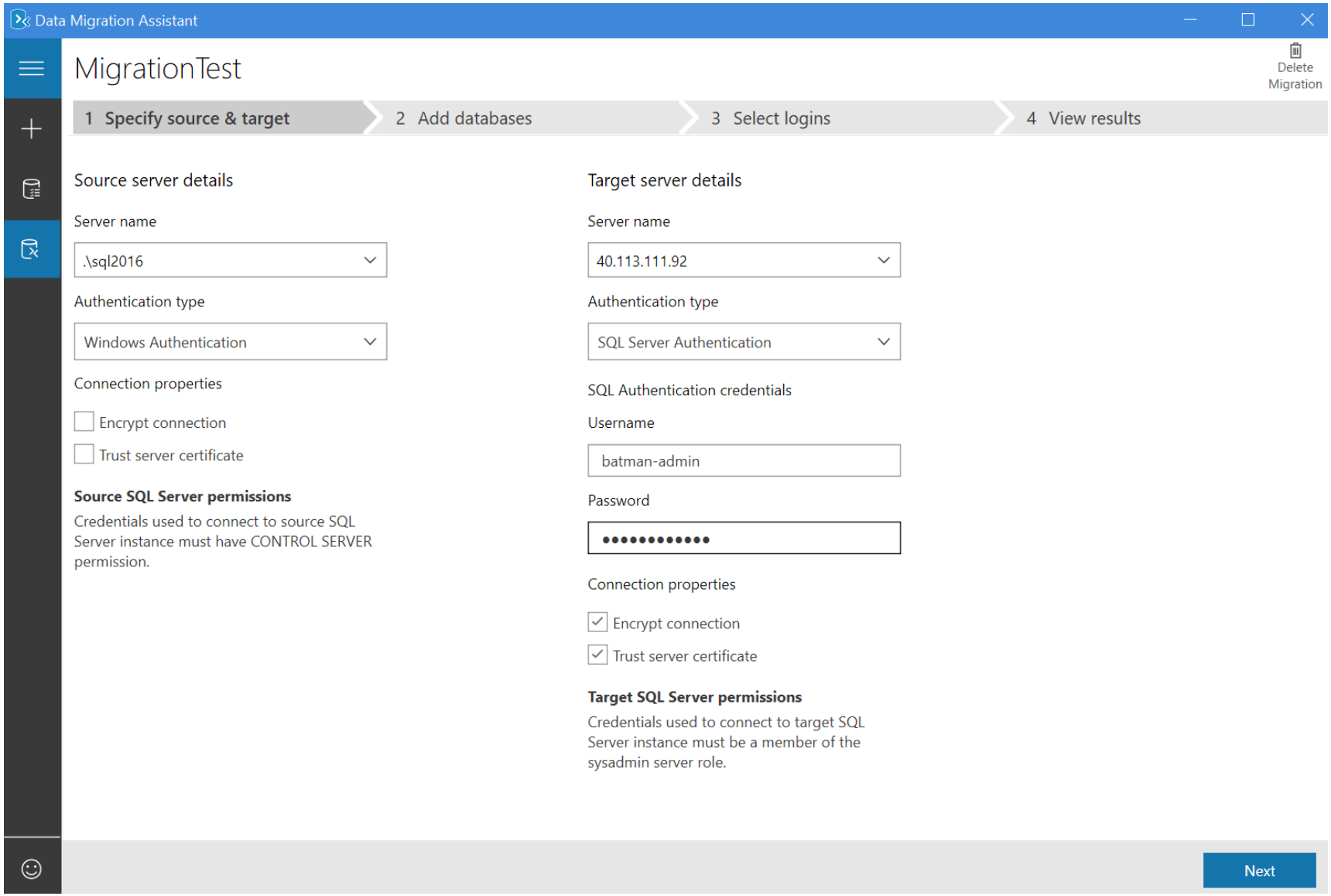
After the assessment has been completed (I have skipped those steps here), create a new Migration



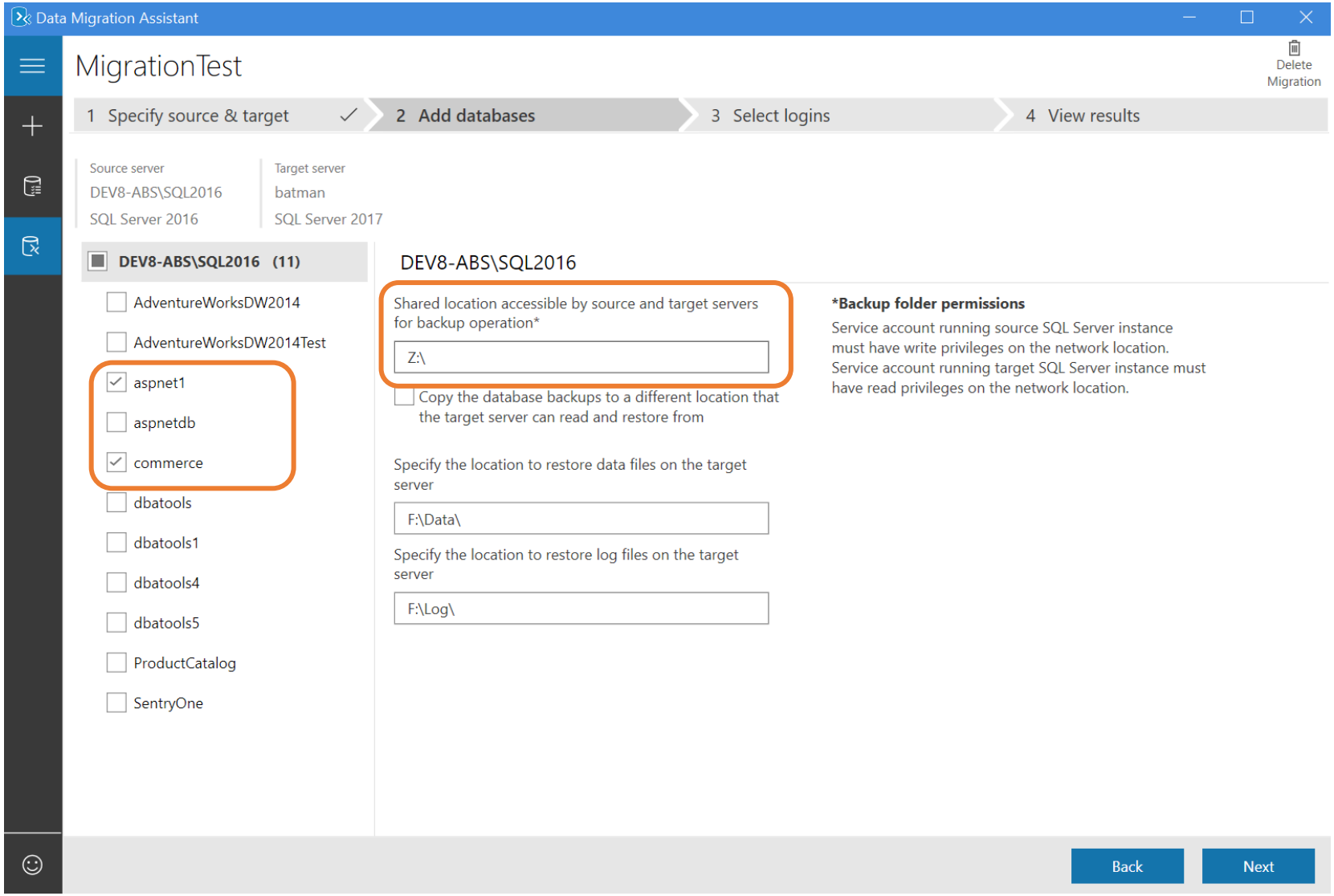
On your Azure portal, navigate to the newly created Virtual Machine to find its IP address



Fill out the source and target server details (make sure both the Encrypt Connection and Trust Server Certificate options are ticked for the target)



Next, we select the databases we want to migrate. The shared location is a little tricky to set up. When you have a production setup, you will probably have a VPN that will allow the Azure VM to communicate with your on-premise network. For our test setup, though, we will need to use something else. I have used a File share on Azure Blob Storage to bridge this gap between the on-prem and cloud servers (see next page for guide), mapped to Z:



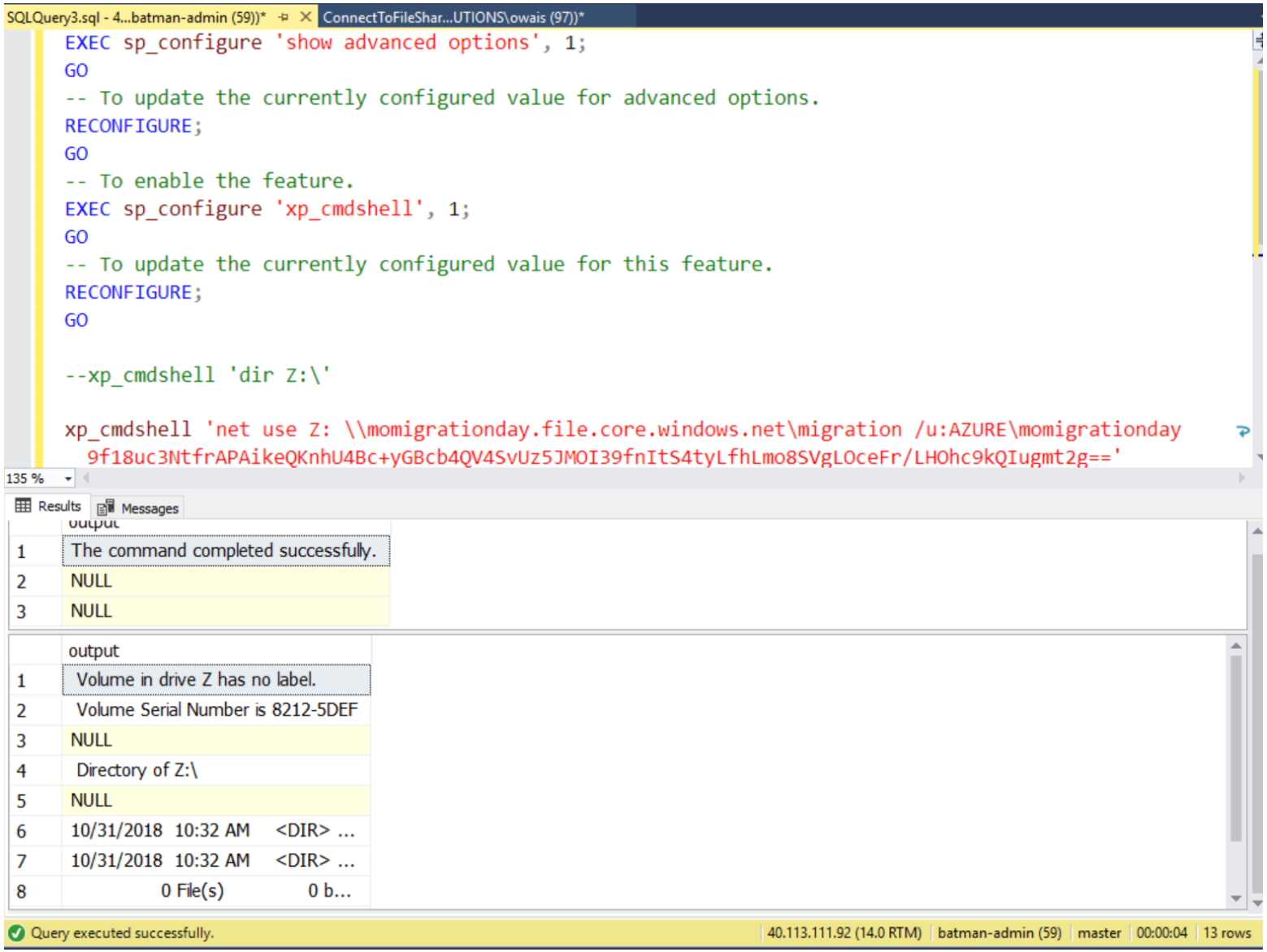
Steps for setting up a file share for both servers to access

Log into the source SQL server using SSMS and run the following script. Note this will connect to my storage account with the given key. I will probably change this key in a few days, so if this script stops working, please create your own file share using the steps defined here.

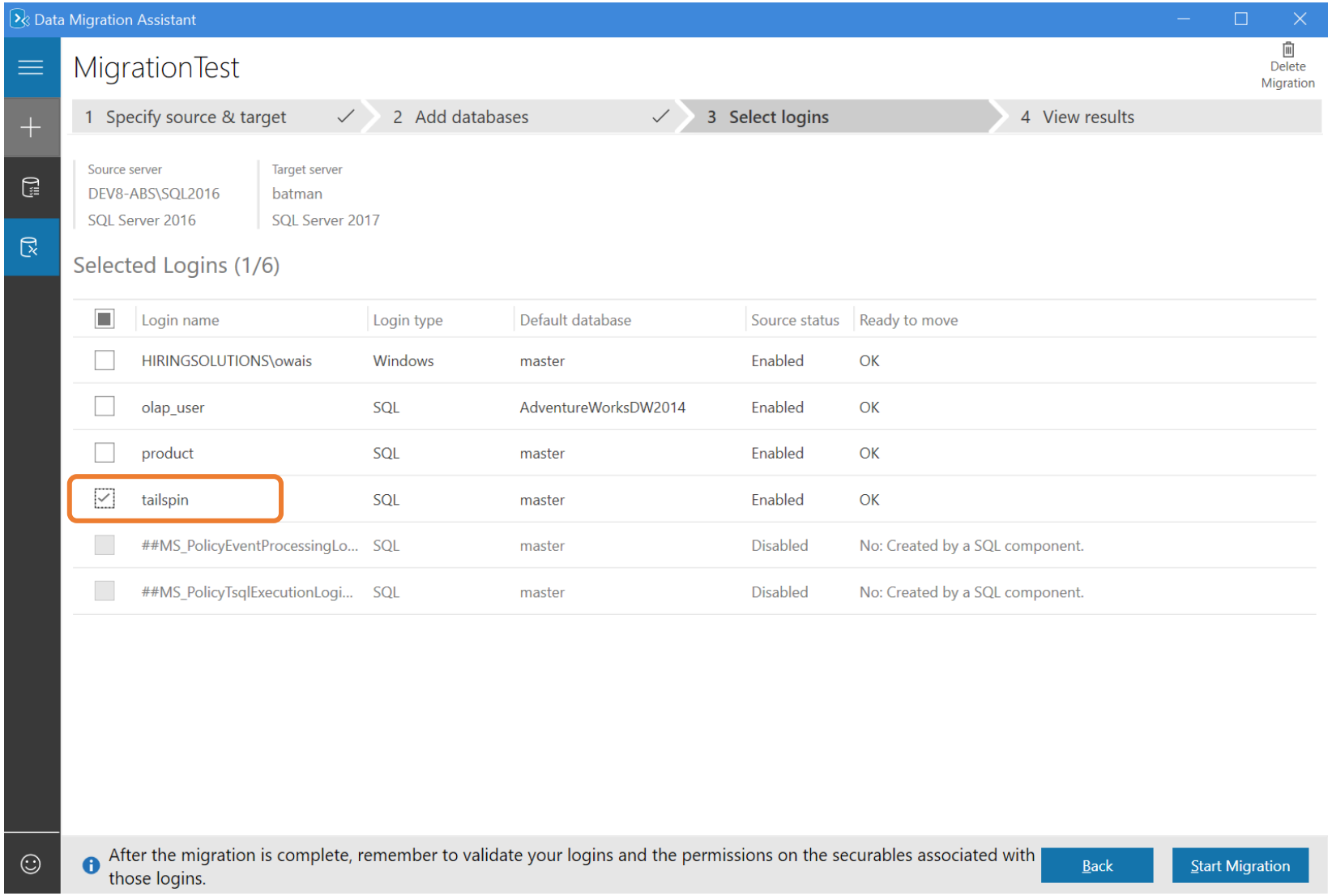
```
EXEC sp_configure 'show advanced options', 1;
GO
-- To update the currently configured value for advanced options.
RECONFIGURE;
GO
-- To enable the feature.
EXEC sp_configure 'xp_cmdshell', 1;
GO
-- To update the currently configured value for this feature.
RECONFIGURE;
GO

xp_cmdshell 'net use Z: \\momigrationday.file.core.windows.net\migration /u:AZURE\momigrationday
9f18uc3NtfrAPAikeQKnhU4Bc+yGBcb4QV4SvUz5JMOI39fnItS4tyLfhLmo8SVgLOceFr/LHOhc9kQIugmt2g=='
GO

xp_cmdshell 'dir Z:\'
GO
```

Now login to the target server using SSMS and run the same script. Now both SQL Server instances have access to a Z: drive that connects to the same network share. This will allow them to exchange the backup files.

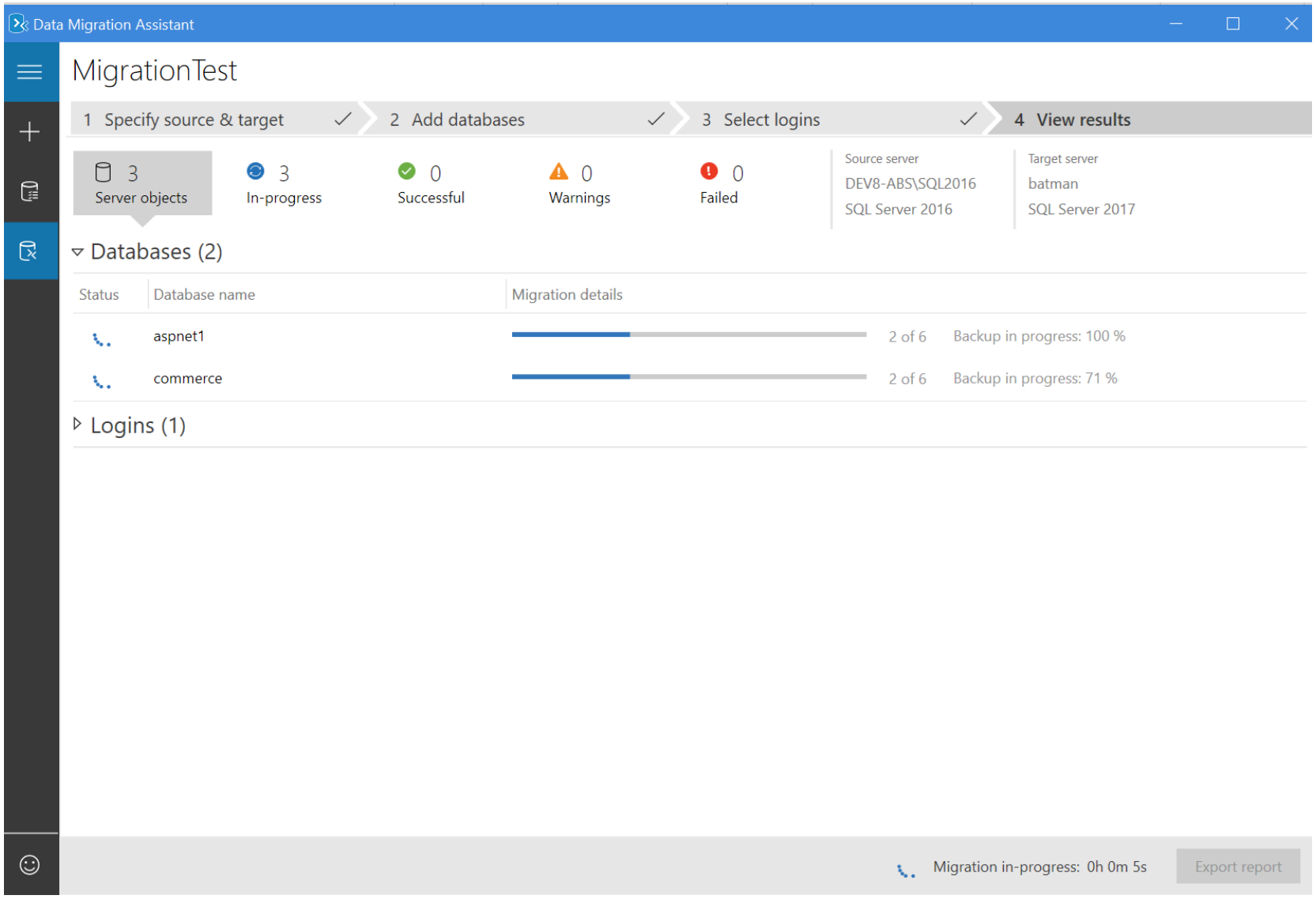
Output of the script confirming that the Z: drive has been mapped



Now back to the Data Migration Assistant, choose the logins you want to migrate (I only want **tailspin**)



Once you click on the Start Migration button, it begins the process.



Depending on your connection speed, this should take 3-5 minutes for our sample database. And we are done!