



Azure Developer Series

Migrating a dotnetcore 2-tier application to Azure,
using different architectures and DevOps best practices

Hands-On-Labs step-by-step guides

Prepared by:

Peter De Tender

CEO and Lead Technical Trainer
PDTIT and 007FFFLearning.com

@pdtit

@007FFFLearning

Version: Sept 2019 – 1.0

Information in this document, including URL and other Internet Web site references, is subject to change without notice. Unless otherwise noted, the example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, e-mail address, logo, person, place or event is intended or should be inferred. Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

The names of manufacturers, products, or URLs are provided for informational purposes only and Microsoft makes no representations and warranties, either expressed, implied, or statutory, regarding these manufacturers or the use of the products with any Microsoft technologies. The inclusion of a manufacturer or product does not imply endorsement of Microsoft of the manufacturer or product. Links may be provided to third party sites. Such sites are not under the control of Microsoft and Microsoft is not responsible for the contents of any linked site or any link contained in a linked site, or any changes or updates to such sites. Microsoft is not responsible for webcasting or any other form of transmission received from any linked site. Microsoft is providing these links to you only as a convenience, and the inclusion of any link does not imply endorsement of Microsoft of the site or the products contained therein.

© 2019 Microsoft Corporation. All rights reserved.

Microsoft and the trademarks listed at <https://www.microsoft.com/en-us/legal/intellectualproperty/Trademarks/Usage/General.aspx> are trademarks of the Microsoft group of companies. All other trademarks are property of their respective owners.

Contents

Migrating a dotnetcore 2-tiered application to Azure using different architectures and DevOps best practices - Hands-On-Labs step-by-step	4
Abstract and Learning Objectives	5
Requirements	6
Naming Conventions:.....	6
Azure Subscription:.....	6
Other requirements:	6
Alternative Approach:	6
Final Remarks:.....	7
Lab 3: Deploying an Azure SQL database and migrating from SQL VM	8
What you will learn	8
Time estimate	8
Task 1: Deploying a new SQL Azure Server instance	8
Task 2: Performing a SQL database migration from a SQL Virtual Machine to SQL Azure, using SQL Data Migration Assistant.	10
Task 3 (Optional): Using SQL Management Studio to migrate from SQL VM to SQL Azure Instance	18
Task 4: Validate SQL Azure Database Migration using Azure Portal Query Editor.....	26
Task 5: Defining a hybrid connection from a Web VM to an Azure SQL database	28
Summary	31

Migrating a dotnetcore 2-tiered application to Azure using different architectures and DevOps best practices

- Hands-On-Labs step-by-step

You are part of an organization that is running a dotnetcore e-commerce platform application, using Windows Server infrastructure on-premises today, comprising a WebVM running Windows Server 2012 R2 with Internet Information Server (IIS) and a 2nd SQLVM running Windows Server 2012 R2 and SQL Server 2014.

The business has approved a migration of this business-critical workload to Azure, and you are nominated as the cloud solution architect for this project. No decision has been made yet on what the final architecture should or will look like. Your first task is building a Proof-of-Concept in your Azure environment, to test out the different architectures possible:

- Infrastructure as a Service (IAAS)
- Platform as a Service (PAAS)
- Containers as a Service (CaaS)

At the same time, your CIO wants to make use of this project to switch from a more traditional mode of operations, with barriers between IT sysadmin teams and Developer teams, to a DevOps way of working. Therefore, you are tasked to explore Azure DevOps and determine where CI/CD Pipelines can assist in optimizing the deployment and running operations of this e-commerce platform, especially when deploying updates to the application.

As you are new to the continuous changes in Azure, you want to make sure this process goes as smooth as possible, starting from the assessment to migration to day-to-day operations.

Abstract and Learning Objectives

This workshop enables anyone to learn, understand and build a Proof of Concept, in performing a multi-tiered .Net Core web application using Microsoft SQL Server database, platform migration to Azure public cloud, leveraging on different Azure Infrastructure as a Service, Azure Platform as a Service (PaaS) and Azure Container offerings like Azure Container Instance (ACI) and Azure Kubernetes Services (AKS).

After an introductory module on cloud app migration strategies and patterns, students get introduced to the basics of automating Azure resources deployments using Visual Studio and Azure Resource Manager (ARM) templates. Next, attendees learn about the importance of performing proper assessments, and what tools Microsoft offers to help in this migration preparation phase. Once the application has been deployed on Azure Virtual Machines, students learn about Microsoft SQL database migration to SQL Azure PaaS, as well as deploying and migrating web applications to Azure Web Apps.

After these foundational platform components, the workshop will totally focus on the core concepts and advantages of using containers for running business workloads, based on Docker, Azure Container Registry (ACR), Azure Container Instance (ACI) and WebApps for Containers, as well as how to enable container orchestration and cloud-scale using Azure Kubernetes Service (AKS).

In the last part of the workshop, students get introduced to Azure DevOps, the new Microsoft Application Lifecycle environment, helping in building a CI/CD Pipeline to publish workloads using the DevOps principals and concepts, showing the integration with the rest of the already touched on Azure services like Azure Web Apps and Azure Kubernetes Services (AKS), closing the workshop with a module on overall Azure monitoring and operations and what tools Azure has available to assist your IT teams in this challenge.

The focus of the workshop is having a Hands-On-Labs experience, by going through the following exercises and tasks:

- Deploying a 2-tier Azure Virtual Machine (Webserver and SQL database Server) using ARM-template automation with Visual Studio 2019;
- Publishing a .NET Core e-commerce application to an Azure Web Virtual Machine and SQL DB Virtual Machine;
- Performing a proper assessment of the as-is Web and SQL infrastructure using Microsoft Assessment Tools;
- Migrating a SQL 2014 database to Azure SQL PaaS (Lift & Shift);
- Migrating a .NET Core web application to Azure Web Apps (Lift & Shift);
- Containerizing a .NET Core web application using Docker, and pushing to Azure Container Registry (ACR);
- Running Azure Container Instance (ACI) and WebApp for Containers;
- Deploy and run Azure Azure Kubernetes Services (AKS);

- Deploying Azure DevOps and building a CI/CD Pipeline for the subject e-commerce application;
- Managing and Monitoring Azure Kubernetes Services (AKS);

Requirements

Naming Conventions:

IMPORTANT: Most Azure resources require unique names. Throughout these steps you will see the word “[SUFFIX]” as part of resource names. You should replace this with your initials, guaranteeing those resources get uniquely named.

Azure Subscription:

Participants need a “pay-as-you-go”, MSDN or other paid Azure subscription

- a) In one of the Azure Container Services tasks, you are required to create an Azure AD Service Principal, which typically requires an Azure subscription owner to log in to create this object. If you don't have the owner right in your Azure subscription, you could ask another person to execute this step for you.
- b) The Azure subscription must allow you to run enough cores, used by the baseline Virtual Machines, but also later on in the tasks when deploying the Azure Container Services, where ACS agent and master machines are getting set up. If you follow the instructions as written out in the lab guide, you need 12 cores.
- c) If you run this lab setup in your personal or corporate Azure payable subscription, using the configuration as described in the lab guide, the estimated Azure consumption costs for running the setups during the 2 days of the workshop is \$20.

Other requirements:

Participants need a local client machine, running a recent Operating System, allowing them to:

- browse to <https://portal.azure.com> from a most-recent browser;
- establish a secured Remote Desktop (RDP) session to a lab-jumpVM running Windows Server 2016;

Alternative Approach:

Where the lab scenario assumes all exercises will be performed from within the lab-jumpVM, (since several tools will be installed on the lab-jumpVM or are already installed by default), participants could also execute (most, if not all...) steps from their local client machine.

The following tools are being used throughout the lab exercises:

- Visual Studio 2017 community edition (updated to latest version); this could also be Visual Studio 2019 community edition - latest version
- Docker for Windows (updated to latest version)
- Azure CLI 2.0 (updated to latest version)
- Kubernetes CLI (updated to latest version)
- SimplCommerce Open Source e-commerce platform example (<http://www.simplcommerce.com>)

Make sure you have these tools installed prior to the workshop, if you are not using the lab-jumpVM. You should also have full administrator rights on your machine to execute certain steps within using these tools.

Final Remarks:

VERY IMPORTANT: You should be typing all of the commands as they appear in the guide, except where explicitly stated in this document. Do not try to copy and paste from Word to your command windows or other documents where you are instructed to enter information shown in this document. There can be issues with Copy and Paste from Word or PDF that result in errors, execution of instructions, or creation of file content.

IMPORTANT: Most Azure resources require unique names. Throughout these steps you will see the word "[SUFFIX]" as part of resource names. You should replace this with your initials, guaranteeing those resources get uniquely named.

Lab 3: Deploying an Azure SQL database and migrating from SQL VM

What you will learn

In this lab, you perform a migration from a SQL 2014 database running on the SQLVM, to SQL Azure PaaS, using SQL Server Management Studio (SSMS):

- Deploy a new SQL Azure server instance;
- Authenticate to SSMS on the SQLVM Virtual Machine;
- Run the database migration wizard from within SSMS;
- Verify the successful migration of the SQL database from the VM to Azure;
- Update the connection strings on the WebVM web application to point to the SQL Azure database instead of the on-premises one on SQLVM;
- Optional: migrate the database using Azure Database Migration Assistant

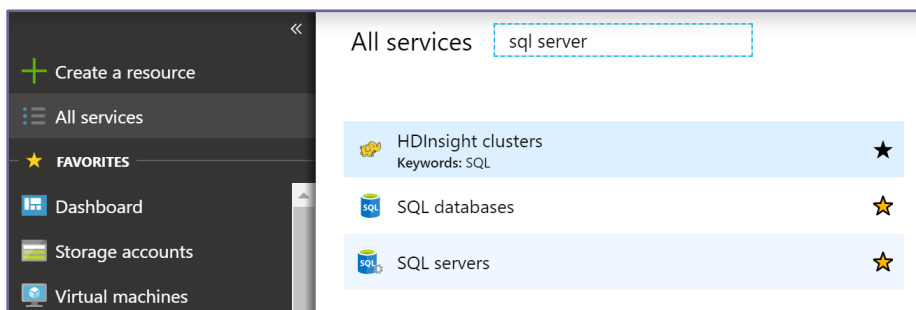
Time estimate

This lab is estimated to take **60min**, assuming your Azure subscription is already available.

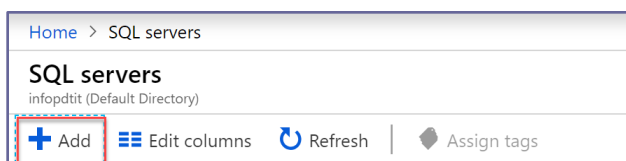
Task 1: Deploying a new SQL Azure Server instance

In this task, you start from deploying a new SQL Azure Server instance from within the Azure Portal, allowing you to migrate a database to it in the next task.

1. From within the **Azure Portal**, Go to “**All Services**”, and enter “**SQL Server**” in the search field. From the list of results, select **SQL Servers**.



2. Click “Create a new SQL Server”, or Click the “**+Add**” button in the top menu. This launches the SQL Server (logical server) deployment blade.



3. **Complete** the different deployment settings as follows:
- Server Name: [suffix]sqlazure[date] e.g. adssqlazure0923 (capitals are not allowed)
 - Server admin login: labadmin
 - Password: [L@BadminPa55w.rd](#) / confirm password: [L@BadminPa55w.rd](#)
 - Subscription: your Azure subscription
 - Resource Group: Create New / [suffix]SQLRG
 - Location: Azure location close to you
 - Allow Azure Services to access server: checked
 - Advanced Threat Protection: Not Now

The screenshot displays the 'SQL Server (logical server only)' configuration page in the Azure portal. The left sidebar shows the 'SQL servers' section with options to 'Add', 'Edit columns', and 'More'. The main area contains the following configuration fields:

- Server name:** adssqlazure0923 (with a green checkmark and '.database.windows.net' suffix)
- Server admin login:** labadmin (with a green checkmark)
- Password:** [masked] (with a green checkmark)
- Confirm password:** [masked] (with a green checkmark)
- Subscription:** Microsoft Azure Sponsorship (dropdown menu)
- Resource group:** (New) ADSSQLRG (dropdown menu with a 'Create new' link below it)
- Location:** East US 2 (dropdown menu)
- Allow Azure services to access server:** ☒ (checkbox with an information icon)
- Advanced Threat Protection:** Start FREE Trial (button) / Not now (button) (with an information icon)

Below the Advanced Threat Protection section, it states: 'FREE trial period of 60 days, and then 12.6495 EUR/server/month.' with a 'Learn more' link.

At the bottom of the configuration area, there is a blue 'Create' button and a link to 'Automation options'.

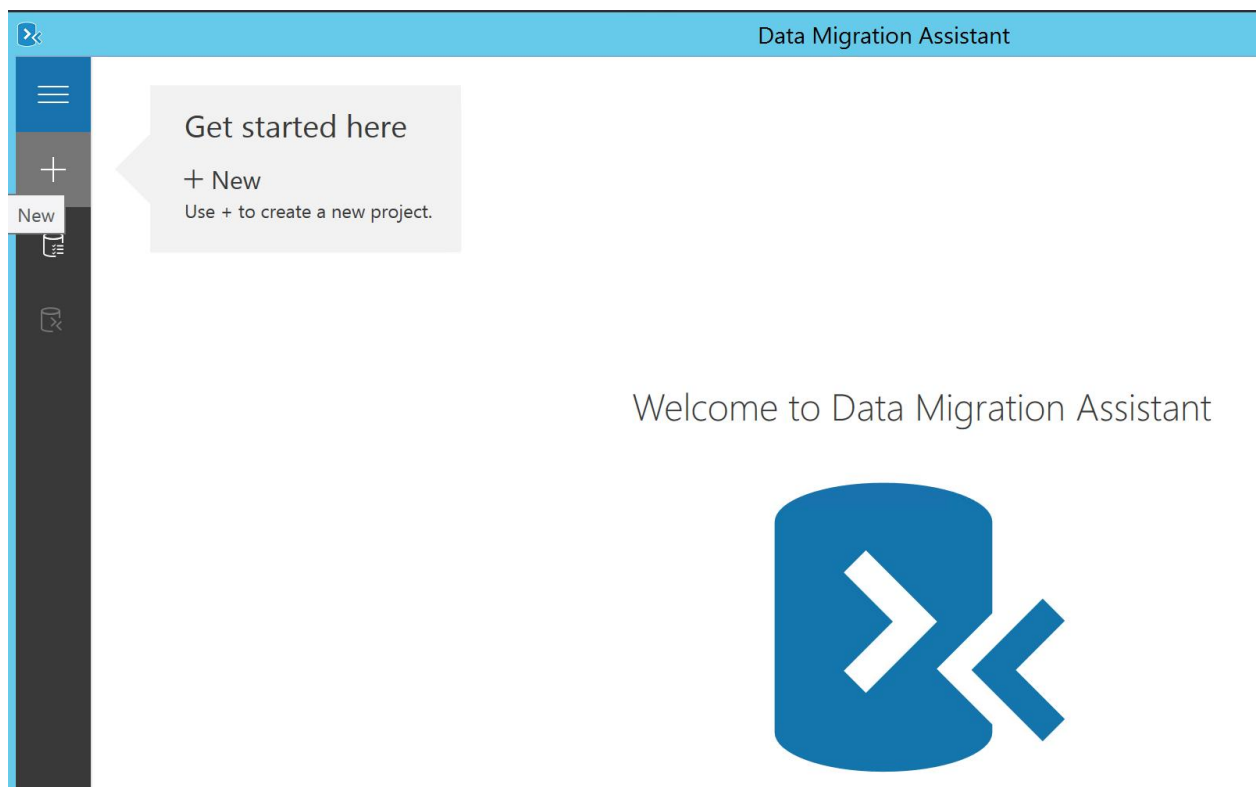
4. **Confirm** the creation of the SQL Azure Server by **pressing the Create** button.
5. **Wait** for the deployment to complete.

6. Once the SQL Azure server has been deployed successfully, continue to Task 2.

Task 2: Performing a SQL database migration from a SQL Virtual Machine to SQL Azure, using SQL Data Migration Assistant.

In this task, you perform a SQL database migration from within a SQL Virtual Machine to SQL Azure. This approach is known as a lift & shift database migration, since no structure or data will be changed during the actual migration. Continuing on the path of the Azure migration tools available, you will use the Azure Data Migration Assistant you used earlier in the assessment phase, to perform the actual migration.

1. **Open an RDP session to the WebVM Virtual Machine** (using the same steps as described in the previous lab).
2. Once you are logged on to the WebVM RDP session, **launch Database Migration Assistant** (from a shortcut on the desktop or Start Menu).



3. Click on "+", to create a new project.

New

Project type

☐ Assessment

☒ Migration

Project name

sqlmig1

Source server type

SQL Server

Target server type

Azure SQL Database

Migration scope

Schema and data

Create

4. Provide the following parameters:

- | | |
|-------------------|--------------------|
| - Project Type | Migration |
| - Project Name | SQLMig1 |
| - Source Server | SQL Server |
| - Target Server | Azure SQL Database |
| - Migration Scope | Schema and Data |

5. Press the **Create** button to start this project.

6. This opens the SQL Migration dashboard; in **Step 1**, complete the following parameters to connect to the source server:

- | | |
|---------------|-------|
| - Server Name | sqlvm |
|---------------|-------|

- Authentication type SQL Server Authentication
- username sa
- Password [L@BadminPa55w.rd](#)
- Encrypt Connection yes
- trust certificate yes

sqlmig1

1 Select source > 2 Select target

Connect to source server

Server name

sqlvm

Authentication type

SQL Server Authentication

SQL Authentication credentials

Username

sa

Password

.....

Connection properties

☒ Encrypt connection

☒ Trust server certificate

Source SQL Server permissions

Credentials used to connect to source SQL Server instance must have CONTROL SERVER permission.

7. This will detect the SimplCommerce SQL database running on the SQL VM. Select it and press Next.

Data Migration Assistant

sqlmig1

1 Select source > 2 Select target > 3 Select objects > 4 Script & deploy schema > 5 Select tables

Server name
sqlvm

Authentication type
SQL Server Authentication

Select a single database from your source server to migrate to Azure SQL Database.
If you skip assessing the databases before migration, DMA will not be able to detect the specific schema. Skip this option if you have already done the assessment and addressed the objects with breaking changes.

Name	Compatibility Level
<input checked="" type="radio"/> SimplCommerce	120

8. In the "Select Target" step, complete the following parameters:

- Server Name servername of the SQL Azure server [suffix]db.database.windows.net
- Authentication SQL Server Authentication
- Username labadmin
- Password L@BadminPa55w.rd
- Encrypt connection yes
- Trust certificate yes

sqlmig1

1 Select source ✓ 2 Select target

Connect to target server

[Create a new Azure SQL Database...](#)

Server name

nopsqlus.database.windows.net

Authentication type

SQL Server Authentication

SQL Authentication credentials

Username

pdtadmin

Password

Connection properties

☒ Encrypt connection

☒ Trust server certificate

Target Azure SQL Database permissions

The principal used to connect must have CONTROL DATABASE permission on the

9. This detects the SQL Azure Database instance you created earlier.

Data Migration Assistant

sqlmig1

1 Select source ✓ 2 Select target 3 Select objects 4 Script & deploy schem 5 Select tables

Server name
nopsqlus.database.windows.net

Authentication type
SQL Server Authentication

SQL Authentication credentials
Username
pdtadmin

Select a single target database from your target Azure SQL Database server. If you intend to migrate Wind

Target external user domain name
e.g. microsoft.com or contoso.com

Name	Compatibility Level
<input type="radio"/> fastnopdock	140
<input type="radio"/> Mon1120	140
<input checked="" type="radio"/> simplcommeredb	140

10. This moves you to the “Select Objects” step. By default, all tables are selected, which is Ok for our scenario.

Data Migration Assistant

sqlmig1

1 Select source ✓ 2 Select target ✓ 3 Select objects 4 Script & deploy schem 5 Select tables 6 Migrate data

Source database
SimplCommerce
sqlvm

Target database
simplcommeredb
nopsqlus.database.windows.net

Assessment issues
No collected objects with blocking issues
No collected objects with other issues

Select the schema objects from your source database that you would like to migrate to Azure SQL Database.

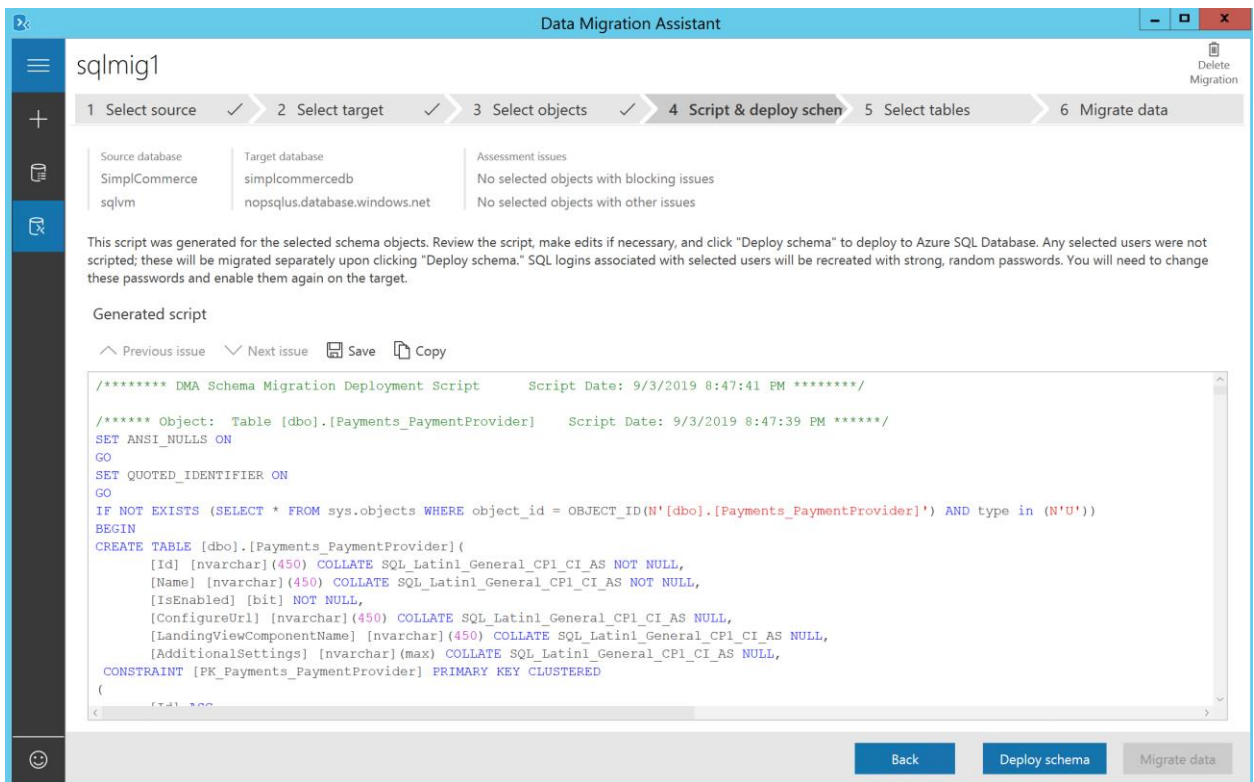
Tables

- ☒ dbo__EFMigrationsHistory
- ☒ dbo.ActivityLog_Activity
- ☒ dbo.ActivityLog_ActivityType
- ☒ dbo.Catalog_Brand
- ☒ dbo.Catalog_Category
- ☒ dbo.Catalog_Product
- ☒ dbo.Catalog_ProductAttribute
- ☒ dbo.Catalog_ProductAttributeGroup
- ☒ dbo.Catalog_ProductAttributeValue
- ☒ dbo.Catalog_ProductCategory
- ☒ dbo.Catalog_ProductLink
- ☒ dbo.Catalog_ProductMedia
- ☒ dbo.Catalog_ProductOption
- ☒ dbo.Catalog_ProductOptionCombination
- ☒ dbo.Catalog_ProductOptionValue
- ☒ dbo.Catalog_ProductPriceHistory
- ☒ dbo.Catalog_ProductTemplate

Select an object to view any issues found for that object

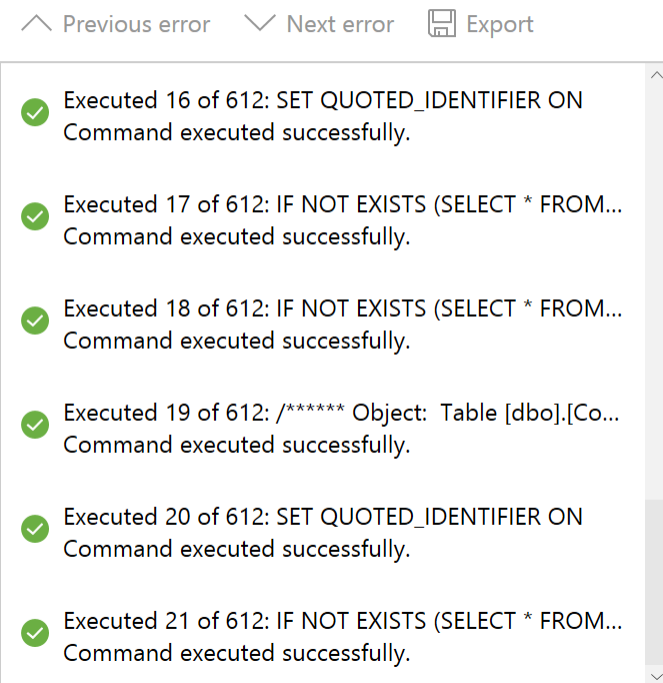
Back Generate SQL script

11. Press “Generate SQL Script” button



12. To run the actual migration, starting with the database schema, press "Deploy Schema".

Deployment results (21 commands executed, 0 err...



13. Wait for this step to complete successfully. This should take only a few minutes.
14. **Lastly**, press the **"Migrate Data"** button to start the actual database content migration. This will first show a list of tables; make sure all tables are selected here to not miss any data.

Selected tables (38/84)

<input type="checkbox"/>	Table name	Row count	Ready to move
<input type="checkbox"/>	[dbo].[ActivityLog_Activity]	4	No: Matching table in target database is not empty
<input type="checkbox"/>	[dbo].[ActivityLog_ActivityType]	1	No: Matching table in target database is not empty
<input type="checkbox"/>	[dbo].[Catalog_Brand]	3	No: Matching table in target database is not empty
<input type="checkbox"/>	[dbo].[Catalog_Category]	6	No: Matching table in target database is not empty
<input type="checkbox"/>	[dbo].[Catalog_Product]	26	No: Matching table in target database is not empty
<input checked="" type="checkbox"/>	[dbo].[Catalog_ProductAttribute]	0	OK
<input checked="" type="checkbox"/>	[dbo].[Catalog_ProductAttributeGroup]	0	OK
<input checked="" type="checkbox"/>	[dbo].[Catalog_ProductAttributeValue]	0	OK

15. And confirm, by pressing the **Start Data Migration** button.

sqlmig1

1 Select source ✓ 2 Select target ✓ 3 Select objects ✓ 4 Script & deploy sct ✓ 5 Select tables ✓ 6 Migrate data

38 Server objects 0 In-progress 38 Successful 0 Warnings 0 Failed

Source database: SimplCommerce sqlvm Target database: simplcommercedb nopsqlus.database.windows.net

▼ Tables (38)

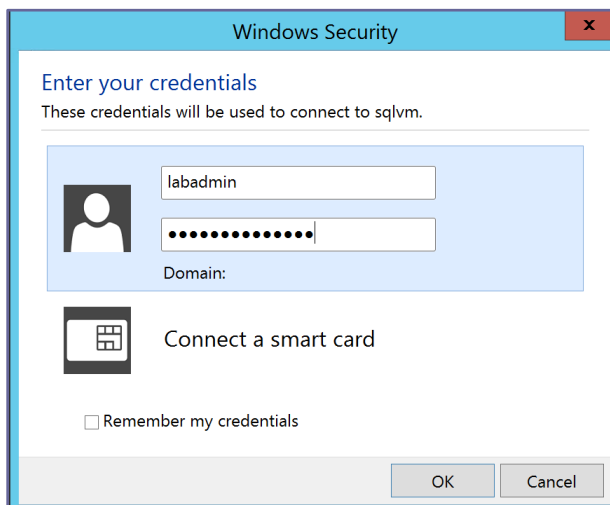
Status	Table name	Migration details
✓	[dbo].[Catalog_ProductAttribute]	Migration successful. Duration: 0 hrs 0 mins 3 secs
✓	[dbo].[Catalog_ProductAttributeGroup]	Migration successful. Duration: 0 hrs 0 mins 2 secs
✓	[dbo].[Catalog_ProductAttributeValue]	Migration successful. Duration: 0 hrs 0 mins 4 secs
✓	[dbo].[Catalog_ProductTemplate]	Migration successful. Duration: 0 hrs 0 mins 2 secs
✓	[dbo].[Catalog_ProductTemplateProductAttribute]	Migration successful. Duration: 0 hrs 0 mins 5 secs
✓	[dbo].[Comments_Comment]	Migration successful. Duration: 0 hrs 0 mins 8 secs
✓	[dbo].[Contacts_Contact]	Migration successful. Duration: 0 hrs 0 mins 3 secs
✓	[dbo].[Contacts_ContactArea]	Migration successful. Duration: 0 hrs 0 mins 3 secs
✓	[dbo].[Core_CustomerGroup]	Migration successful. Duration: 0 hrs 0 mins 4 secs
✓	[dbo].[Core_CustomerGroupUser]	Migration successful. Duration: 0 hrs 0 mins 4 secs
✓	[dbo].[Core_RoleClaim]	Migration successful. Duration: 0 hrs 0 mins 3 secs

Migration in-progress: 0h 0m 55s

16. Wait for this process to complete successfully; this should take about 10 minutes on average.

Task 3 (Optional): Using SQL Management Studio to migrate from SQL VM to SQL Azure Instance

1. If your DBA team is familiar with SQL Server Management Studio, know they can keep using this tool to perform the actual SQL database migration as well. To use this method, open an **RDP Session** to the **WebVM** (labadmin / L@BadminPa55w.rd).
2. Next, from within the RDP session of the WebVM, open a 2nd RDP session to the SQLVM machine (remember, the SQL VM has no public IP address, not making it reachable from the outside).
3. As server name, type "**SQLVM**". (Since both Virtual Machines are in the same Azure Virtual Network and subnet, the netbios name resolution works, relying on Azure DNS). **Press connect.**

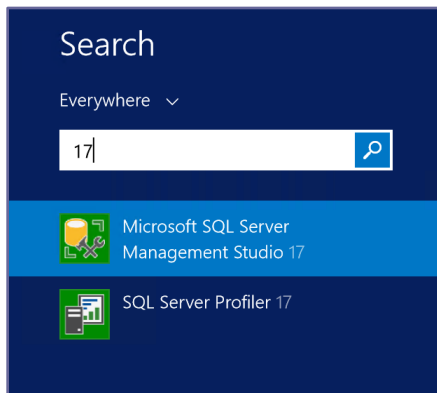


4. Provide the local admin credentials of the SQLVM Virtual Machine:
 - labadmin
 - [L@BadminPa55w.rd](#)

and confirm with **OK**.


5. Once you **are logged on** to the **SQL Server Virtual Machine** (notice the SQL Getting Started shortcut on the desktop), press the **Start** button. **Start typing "17"**; this will resolve several

management tools available on the server. Notice **Microsoft SQL Server Management Studio 17**.




6. Select it to start the **SQL Server Management Studio 17** console.
7. Once opened, you are asked for **server connection information**. This is where you provide the SQL Azure name. You can find this in the **Azure Portal**, by browsing to your **SQL Azure instance**, and selecting its **properties**


[Home](#) > [SQL servers](#) > nopsqlus - Properties





nopsqlus - Properties


SQL server

 Overview


 Activity log


 Access control (IAM)


 Tags


 Diagnose and solve problems


Settings


 Quick start


 Failover groups


 Manage Backups


 Active Directory admin


 SQL databases


 SQL elastic pools


 Deleted databases

 Import/Export history

 DTU quota

 Properties

 Locks

 Export template

Status

Available

Server name

nopsqlus.database.windows.net

Location

Central US

Server admin login

pdtadmin

Active Directory admin

Not configured

Resource group

noprg

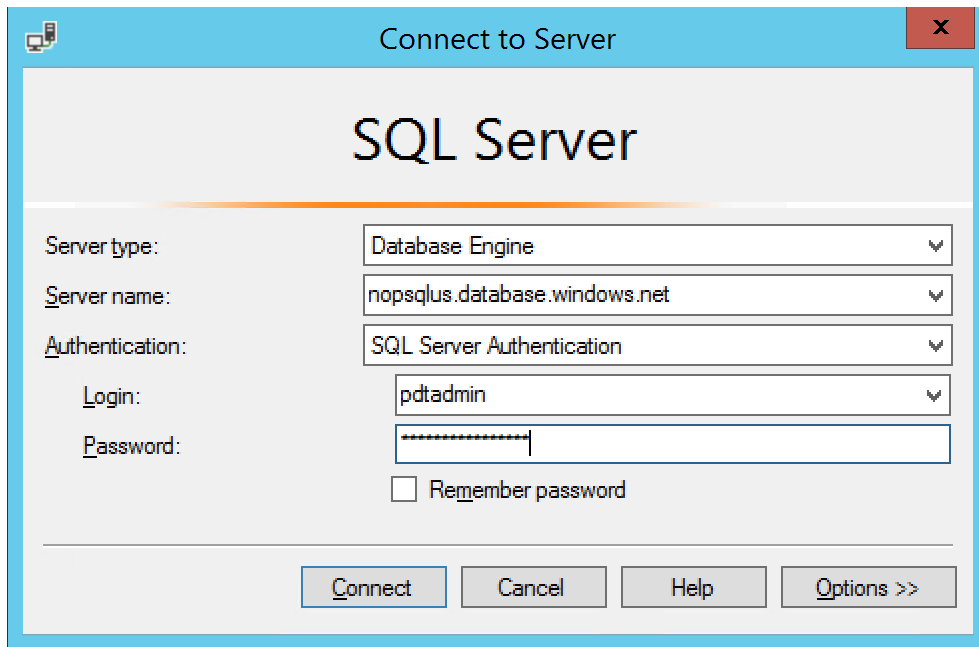
Subscription ID

0a407898-c077-442d-8e17-71420aa82426

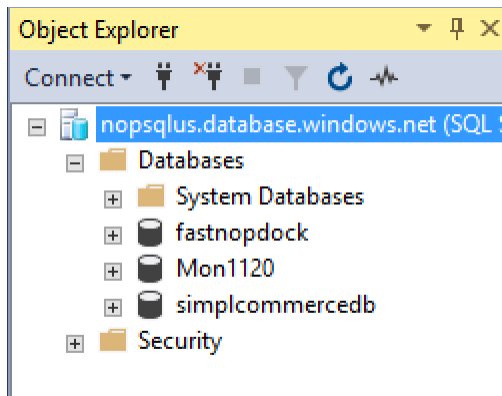
Subscription name

007FFFLearning Labs

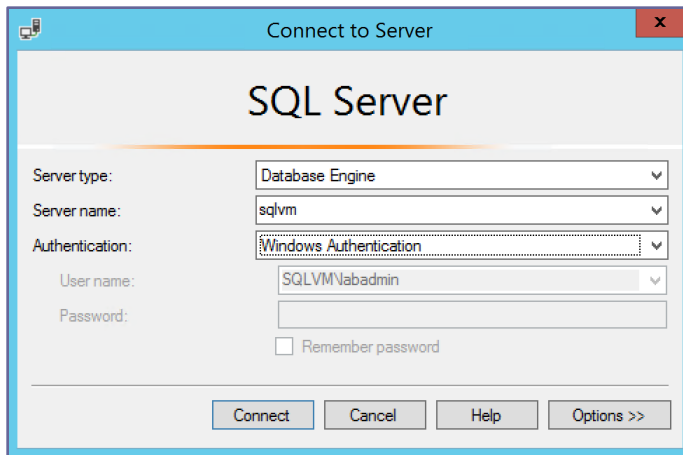
8. **Copy** the SERVER NAME into the **Server Name** field of the SQL connection popup; in the **Authentication** field, change to **SQL Server Authentication**. Provide the Login and Password of the SQL Azure instance account you provided during the deployment of this resource. (labadmin / [L@BadminPa55w.rd](#) would be the instructed ones)



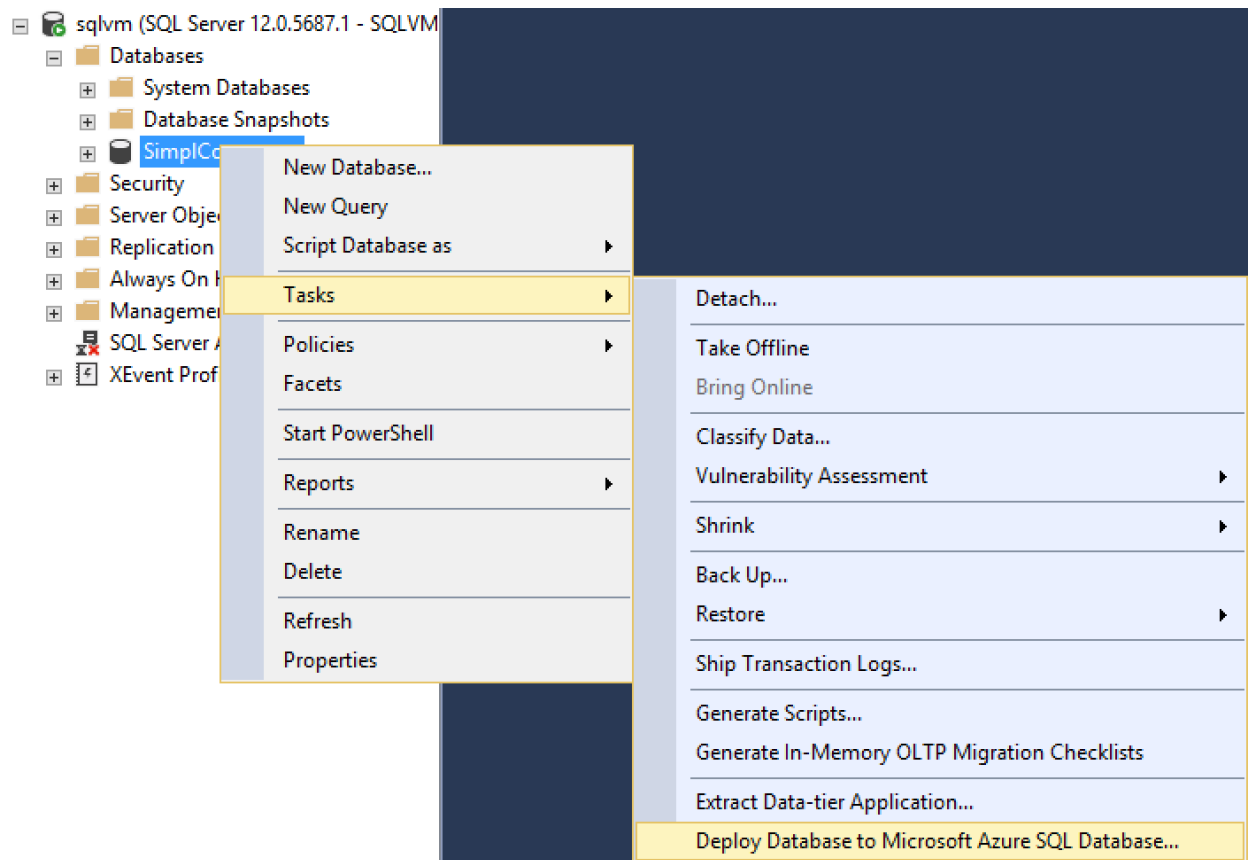
9. Press **Connect** to log on to this SQL Server instance.



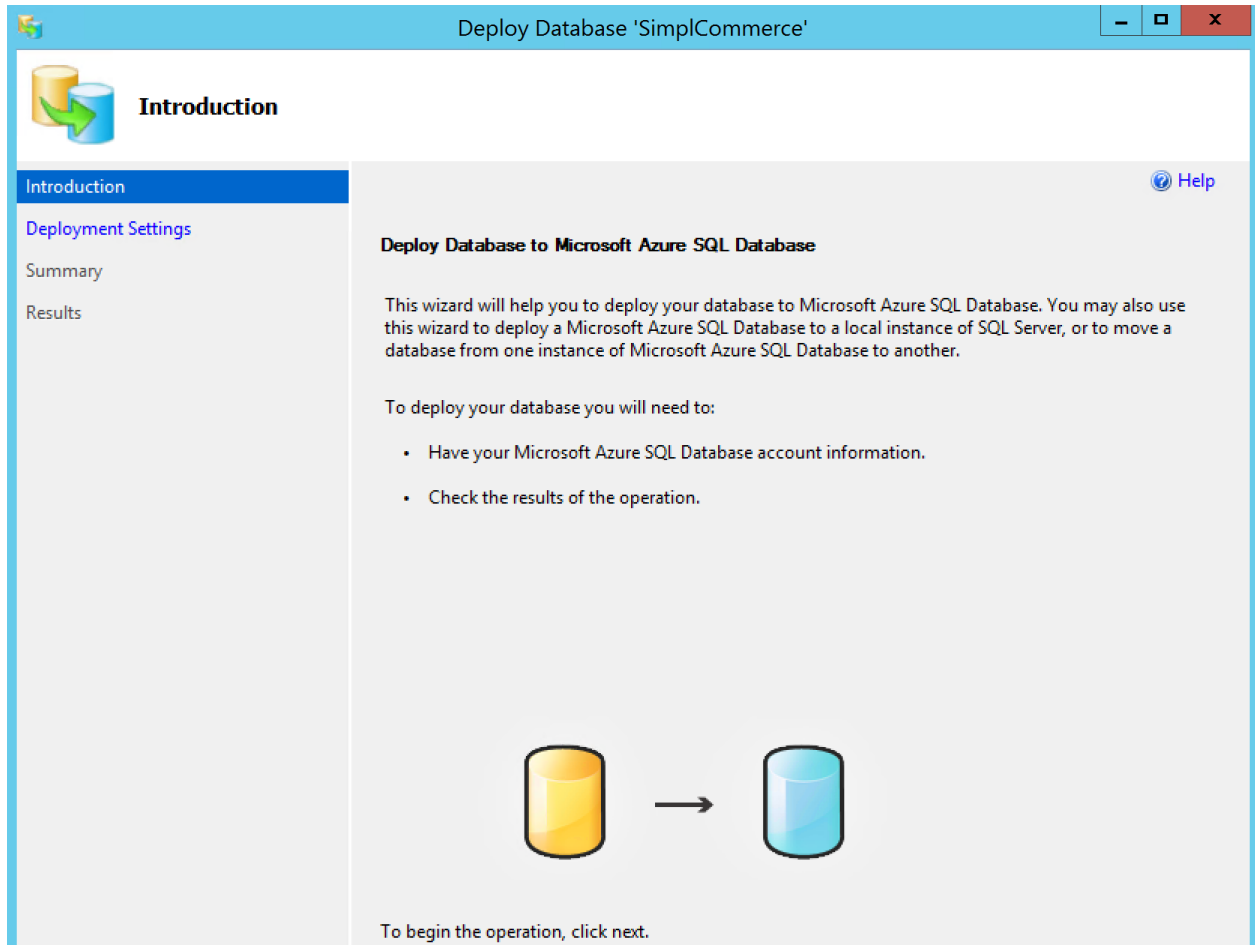
10. In order to have a connection to the SQL VM database instance, we need to add another connection. From the SQL Management Studio console, click **File / Connect Object Explorer**. In the **Connect to server** popup that appears, this time provide the server credentials from the SQLVM:
server name: sqlvm
authentication: **Windows Authentication**



11. Press the **Connect** button.
12. The **Object explorer** shows a successful connection to both databases now. If you open the Databases level, you should see the **SimplCommerce** database.
13. The next step is running the actual migration of the database. Therefore, **select** the **AdventureWorks** database on the local sqlvm, **right-click** it, select **Tasks**, and next, select **Deploy Database to Microsoft SQL Azure Database**.



14. Press the **Next** button when you see the **Introduction** step showing up.



15. In the **Deployment Settings**, provide the **Server Connection** by pressing the **Connect** button.

Provide the following details here:

- Server Connection: <your sql server in Azure>
- New Database name: **SimplCommerce**
- Edition of Microsoft SQL Database: **Basic**
- Max DB size: **2GB**
- Service Objective: **Basic**

Deploy Database 'SimplCommerce'

Deployment Settings

Introduction
Deployment Settings
 Summary
 Results

[Help](#)

Specify Target Connection

Specify the name of the instance of SQL Server or the Microsoft Azure SQL Database server that will host the deployed database, name the new database, and then click Connect to login to the target server.

Server connection:

New database name:

Microsoft Azure SQL Database settings

Edition of Microsoft Azure SQL Database:

Maximum database size (GB):



Service Objective:

Other settings

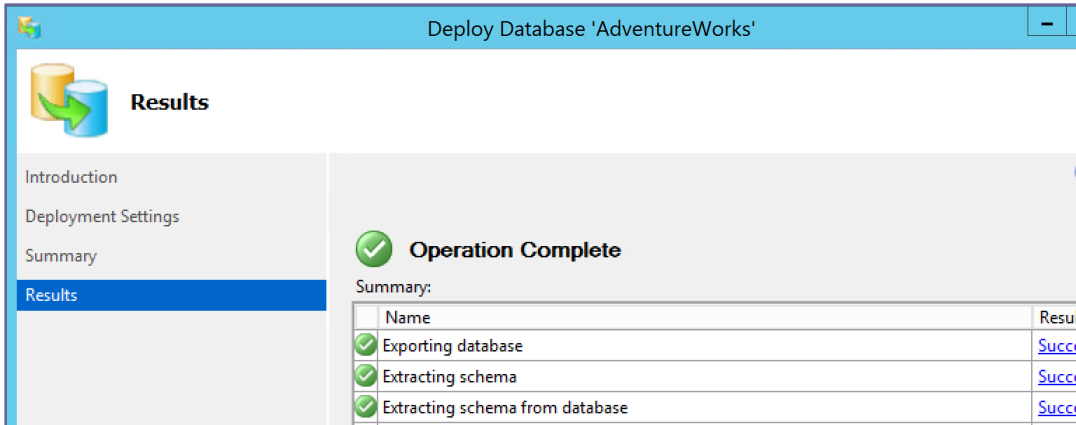
Temporary file name:

16. **Read** through the settings in the summary step. **Press** the **Finish** button to start the actual move process.

Exporting database

Name	Status
 Extracting schema	In Progress
 Extracting schema from database	In Progress

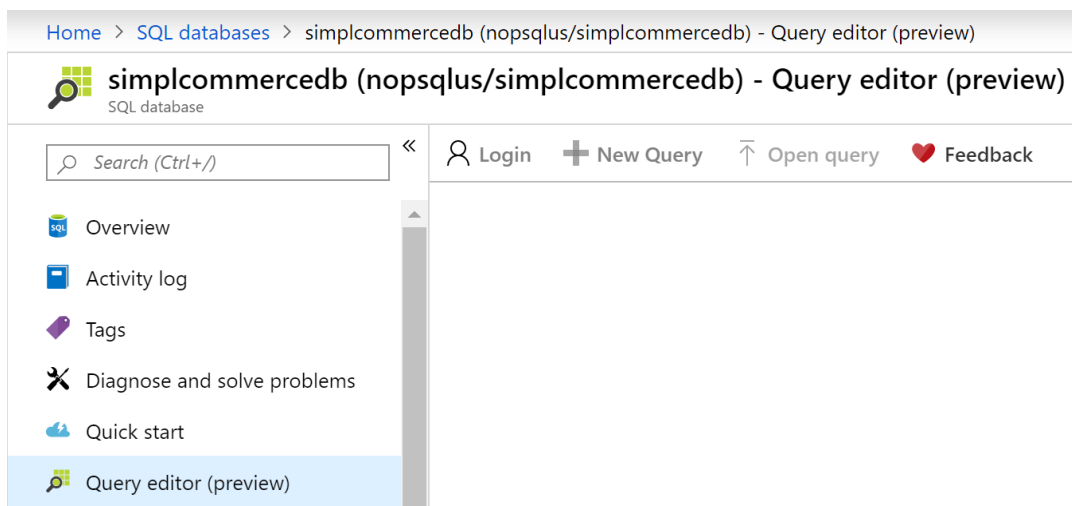
17. **Wait** for this process to complete – this could take about 10minutes.



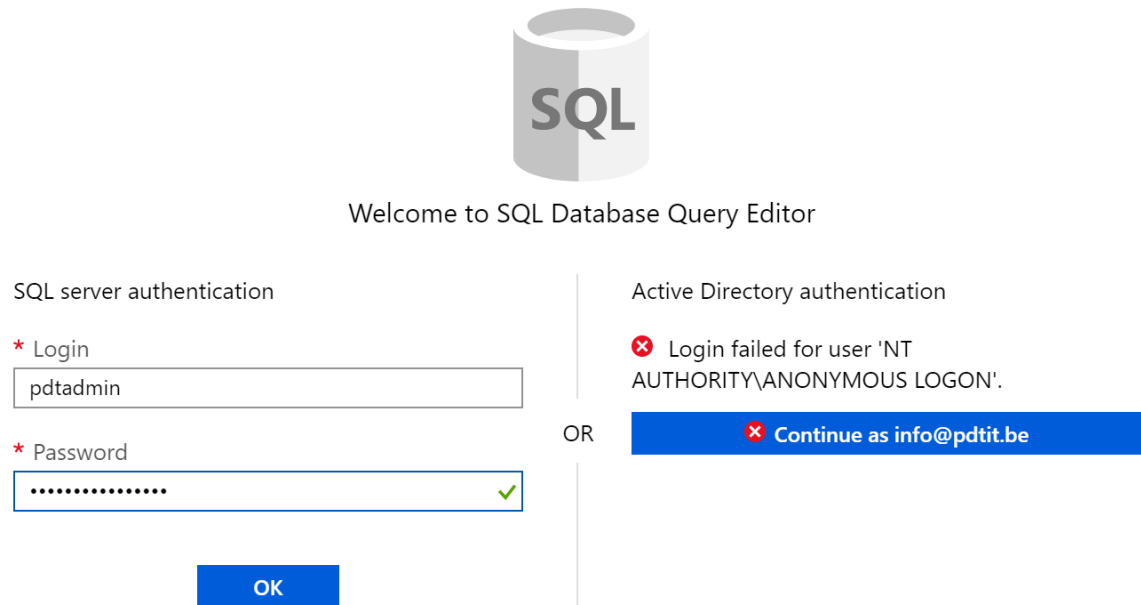
18. Once completed, close the migration window.

Task 4: Validate SQL Azure Database Migration using Azure Portal Query Editor

1. To validate the database is actually migrated successfully, one can go back to the Azure Portal, and use the Query editor, which is currently in preview, to check on the contents of the database and tables. **From the Azure Portal**, navigate to **all services** and **select** SQL databases.
2. From the list of databases, select **SimplCommerceDB (migrated with Azure Data Migration Assistant)**, or **SimplCommerceDB (migrated using SQL Mgmt Studio)**.
3. From the **SimplCommerceDB** detailed blade, select **Query editor (preview)**



4. Notice you are prompted to provide your SQL database credentials, where your Azure admin credentials are not giving you access to the database content.



The image shows the 'Welcome to SQL Database Query Editor' window. At the top is a cylinder icon with 'SQL' on it. Below the title, there are two authentication options. The left option is 'SQL server authentication' with fields for 'Login' (containing 'pdtadmin') and 'Password' (masked with dots and a green checkmark), and an 'OK' button. The right option is 'Active Directory authentication' with a red error message: 'Login failed for user 'NT AUTHORITY\ANONYMOUS LOGON''. Below this is a blue button that says 'Continue as info@pdtit.be'. A vertical line with the word 'OR' separates the two options.

SQL Database Query Editor

SQL server authentication

* Login
pdtadmin

* Password
..... ✓

OK

Active Directory authentication

✗ Login failed for user 'NT AUTHORITY\ANONYMOUS LOGON'.

OR

✗ Continue as info@pdtit.be

5. Provide the labadmin / [L@BadminPa55w.rd](#) credentials, and **press OK**. This opens the query editor window. In the Query editor to the right, enter the following SQL query:

```
select * from dbo.Catalog_Product
```

and **press Run**. This will show you the full list of all Products we have in the database. Which confirms our migration from SQL Management Studio ran successful.

Query 1 x

Run Cancel query Save query Export data as .json Export data as .csv

Export data as .xml

```
1 select * from dbo.catalog_product
```

Results Messages

Search to filter items...

ID	NAME	SLUG	METATITLE
1	Lightweight Jacket	lightweight-jacket	
2	Lightweight Jacket M Black	lightweight-jacket-m-black	
3	Lightweight Jacket M Gray	lightweight-jacket-m-gray	
4	Lightweight Jacket L Black	lightweight-jacket-l-black	
5	Lightweight Jacket L Gray	lightweight-jacket-l-gray	

Task 5: Defining a hybrid connection from a Web VM to an Azure SQL database

- To complete our hybrid cloud migration, we will now update the connection string settings in the appsettings.json file of our Web VM web site application. This information can be retrieved from the SQL database settings in the Azure Portal. **From within the SQL database detailed blade**, browse to **Connection String** under the **settings** section.

Home > SQL databases > simplcommercedb (nopsqlus/simplcommercedb) - Connection strings

simplcommercedb (nopsqlus/simplcommercedb) - Connection strings

ADO.NET JDBC ODBC PHP

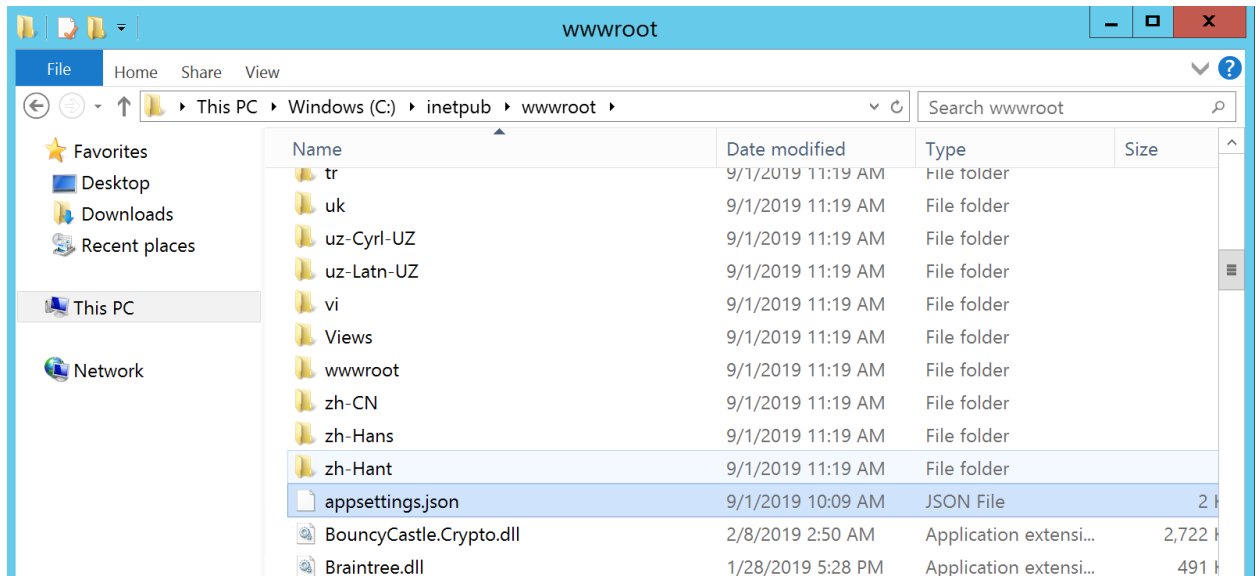
ADO.NET (SQL authentication)

Server=tcp:nopsqlus.database.windows.net,1433;Initial Catalog=simplcommercedb;Persist Security Info=False;User ID=(your_username);Password=(your_password);MultipleActiveResultSets=False;Encrypt=True;TrustServerCertificate=False;Connection Timeout=30;

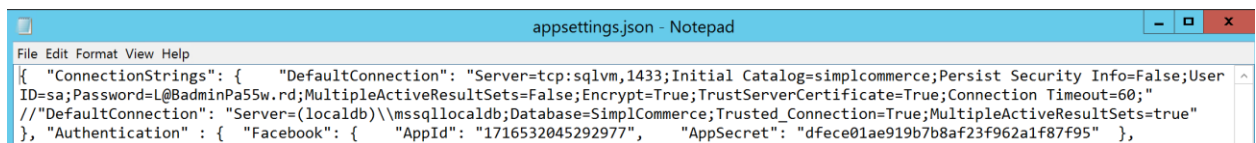
Download ADO.NET driver for SQL server

- Leave this information on screen, as you will need to copy parts of the ADO.NET connection string information into the Web server's web.config file.

3. Go back to the WebVM Virtual Machine Remote Desktop session (or open it again when you already closed the WebVM RDP session)
4. Browse to the IIS web server folder that has the web application content: `c:\inetpub\wwwroot\` and open the file `appsettings.json` with Notepad.



5. Go to the section that starts with "connectionStrings".

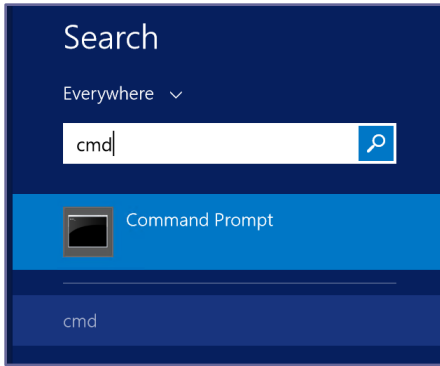


6. Replace the following settings with the parameters from the Connection String information in the Azure Portal:
 - `Server=tcp:sqlvm=>` change the sqlvm to <Azure SQL server name> e.g. `nopsqlus.database.windows.net` in our example

- `Uid=sa =>` change the sa account to `labadmin`

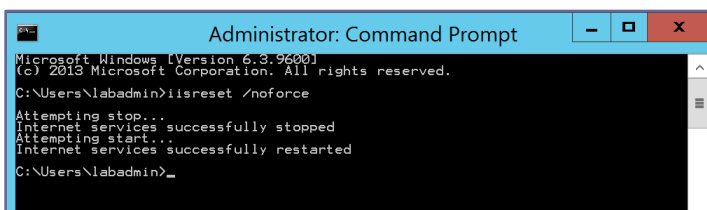
Save the changes to the `appsettings.json` file.

7. From the Start Screen on the WebVM, open a command prompt, by typing "CMD".

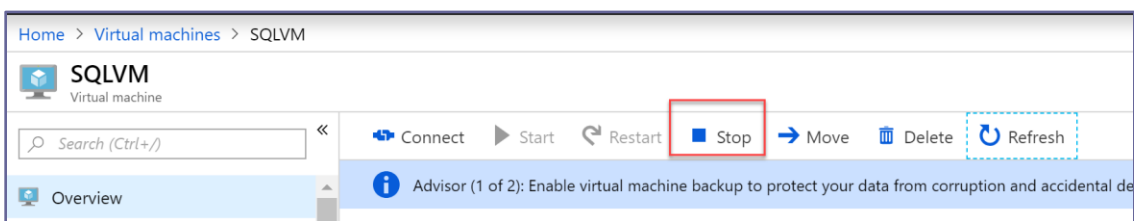


8. In the command prompt, run the following command, to restart the IIS Web Server service.

```
iisreset /noforce
```

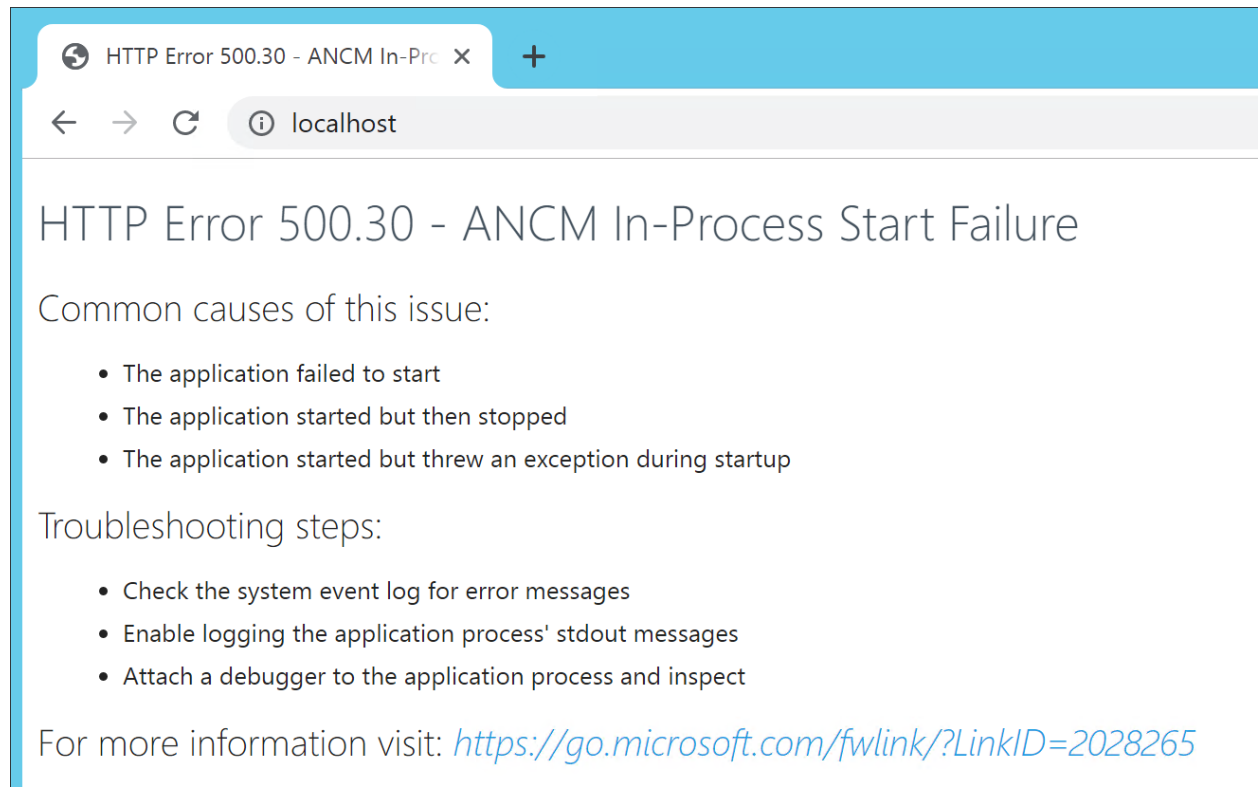


9. To proof that the web application is now connecting to the Azure SQL database, let's shutdown the SQLVM. **From the Azure Portal**, navigate to **Virtual Machines**, and click on the **SQLVM** Virtual Machine.
10. From the **SQLVM** detailed blade, **press** the **STOP** button in the top menu. Wait for the notification message, telling you the VM has shutdown.



11. To test if the web application is now connecting to the Azure SQL database, browse to the web site from within the WebVM's browser, connecting to localhost.
12. The website should load successfully and showing you the product catalog list.
13. If you receive an error message in the browser, similar to below screenshot, it means there is something wrong with the SQL database connection. Verify your settings again the

appsettings.json file, and run IISreset again from the command prompt..



14. This completes this lab.

Summary

In this lab, you learned how to deploy a SQL Azure server resource, as well as how to migrate a SQL database using Azure SQL Data Migration Assistant, and/or the SQL Server Management Studio 17. You updated the IIS web server appsettings.json file and validated the web application is now running in a hybrid setup.