Microsoft Cloud Workshops

Migrate EDW to Azure SQL Data Warehouse Hackathon

Lab Guide

February 2017

© 2017 Microsoft Corporation. All rights reserved. This document is confidential and proprietary to Microsoft. Internal use only. This document is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS SUMMARY.

This document is provided "as-is." Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it.

Some examples are for illustration only and are fictitious. No real association is intended or inferred.

## Contents

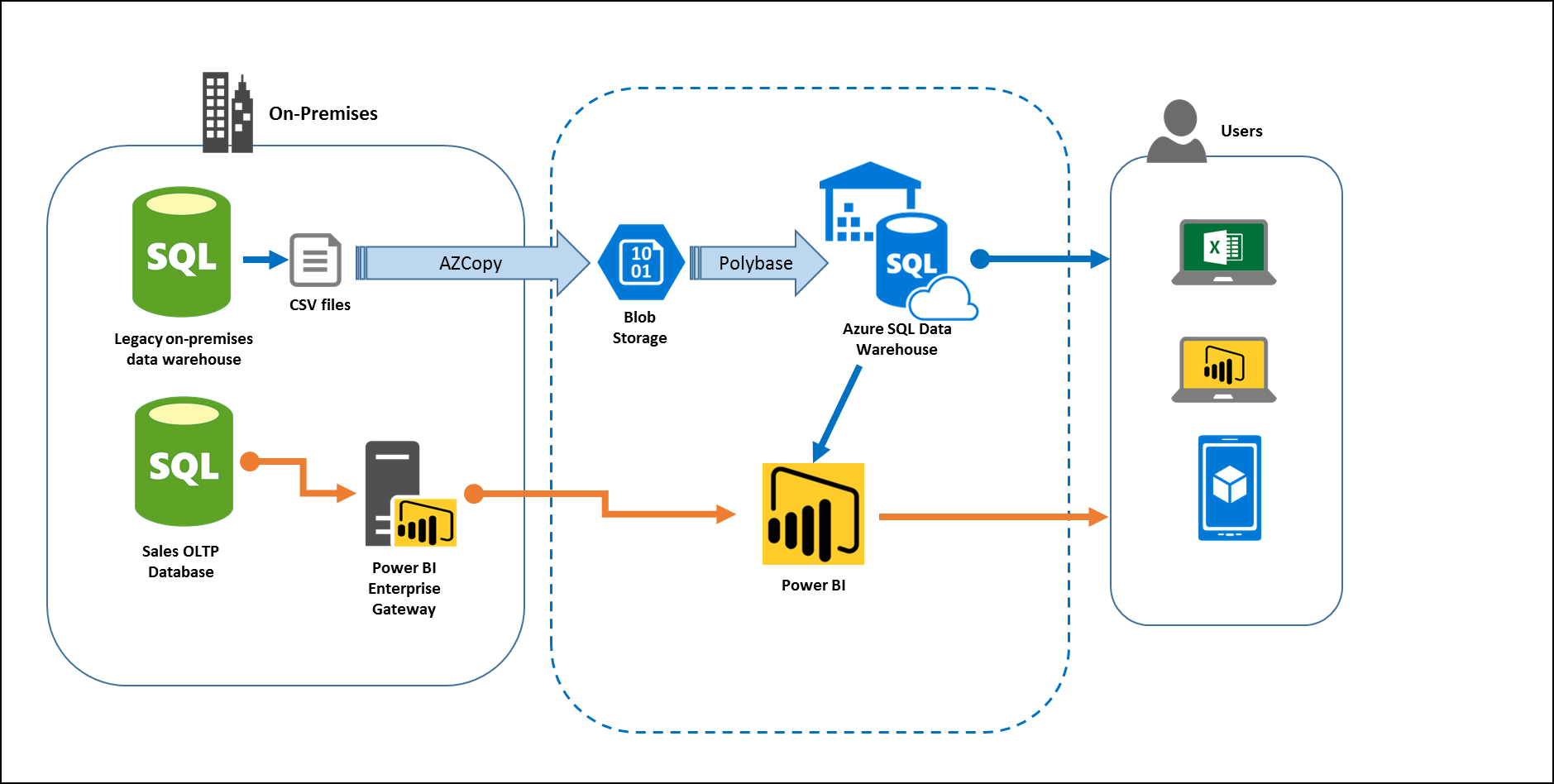
[Exercise 0: Environment setup 3](#_Toc474141703)

[Migrate to Azure SQL Data Warehouse Hackathon 11](#_Toc474141704)

[Migrate to Azure SQL Data Warehouse Hackathon Answers 14](#_Toc474141705)

## Overview

Coho has asked you to migrate an existing on-premises SQL Server data warehouse to Azure SQL Data Warehouse. To build out a viable solution that can replace the existing functionality of the on-premises system you will need to setup and configure an Azure SQL Data Warehouse, validate and migrate the existing data warehouse schema and data to Azure SQL Data Warehouse, and configure Power BI to access both the Azure SQL Data Warehouse and an existing on-premises OLTP sales database via Power BI Gateway.

******

*Data migration process and Power BI configuration*

## Requirements

* Microsoft Azure subscription
* Local machine or a virtual machine configured with:
  + Visual Studio 2015 Community Edition
  + Azure SDK 2.9+ for Visual Studio
    - <https://www.microsoft.com/en-us/download/details.aspx?id=51657>
  + Azure PowerShell (installed as part of Azure SDK)
  + Reboot after installing the Azure SDK

## Lab structure

This lab has two sets of instructions. The first is a high-level set of instructions that is designed for students that have previous experience authoring templates in Azure. The second is a traditional hands-on lab guide that is designed for users that are new to Azure Resource Manger template authoring.

## Help references

|  |  |
| --- | --- |
| Azure SQL Data Warehouse | https://docs.microsoft.com/en-us/azure/sql-data-warehouse/ |
| Migrate your solution to SQL Data Warehouse | https://docs.microsoft.com/en-us/azure/sql-data-warehouse/sql-data-warehouse-overview-migrate |
| Load data into Azure SQL Data Warehouse | https://docs.microsoft.com/en-us/azure/sql-data-warehouse/sql-data-warehouse-overview-load |
| Design decisions and coding techniques for SQL Data Warehouse | https://docs.microsoft.com/en-us/azure/sql-data-warehouse/sql-data-warehouse-overview-develop |
| Leverage other services with SQL Data Warehouse | https://docs.microsoft.com/en-us/azure/sql-data-warehouse/sql-data-warehouse-overview-integrate |
| Azure SQL Data Warehouse with DirectQuery | <https://powerbi.microsoft.com/en-us/documentation/powerbi-azure-sql-data-warehouse-with-direct-connect/> |
| Power BI Gateway - Enterprise in-depth | <https://powerbi.microsoft.com/en-us/documentation/powerbi-gateway-enterprise-indepth/> |

# Exercise 0: Environment setup

## Overview

In this lab, you will create a lab machine if necessary and deploy the source environment for this hackathon.

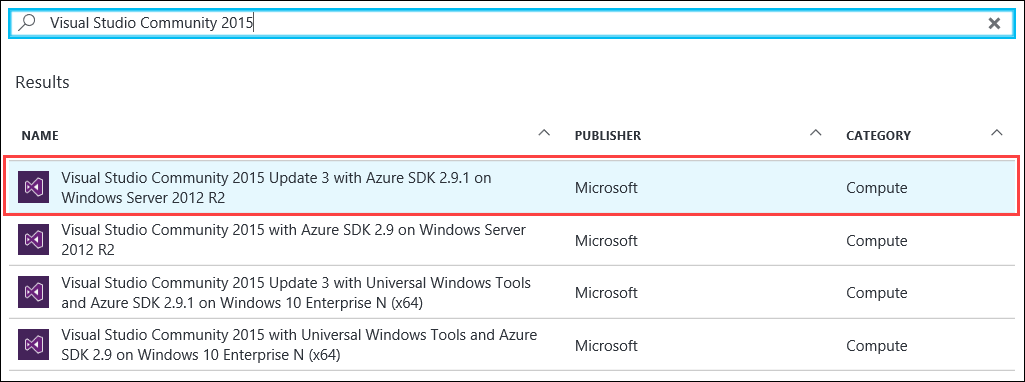
## Prerequisites

* Microsoft Azure subscription: <http://azure.microsoft.com/en-us/pricing/free-trial/>
* Client computer with Windows 7 or later with Visual Studio 2015 - SDK 2.9.1
  + Ensure you reboot after installing the SDK or Azure PowerShell will not work correctly

### Task 1: Setup a Development Environment

If you do not have a machine setup with Visual Studio 2015 Community and Azure SDK 2.9.1 complete this task.

1. Create a virtual machine in Azure using the Visual Studio Community 2015 Update 3 with Azure SDK 2.9 on Windows Server 2012 R2 image.

It is highly recommended to use a DS2\_V2 or D2\_V2 instance size for this VM

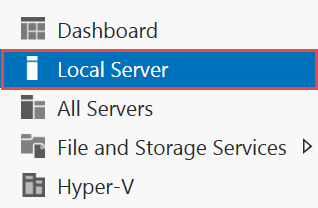
### Task 2: Disable IE Enhanced Security

Note: Sometimes this image has IE ESC disabled, sometimes it does not.

1. On the new VM you just created click the Server Manager icon.



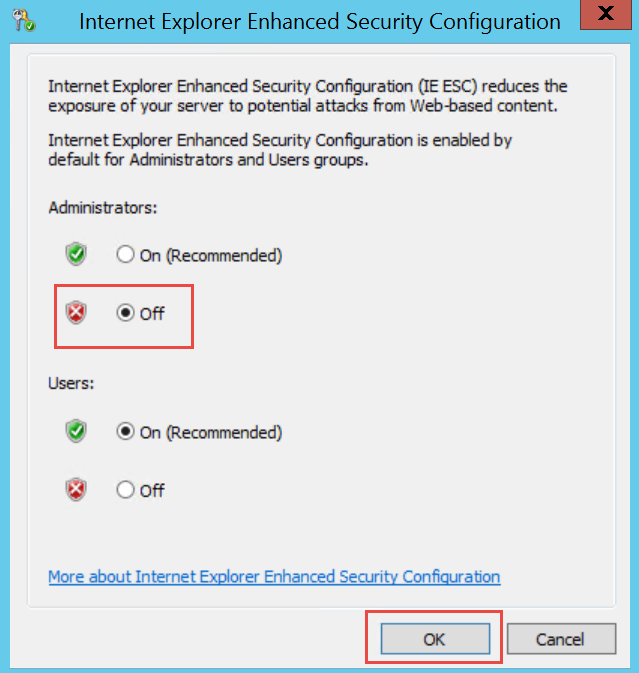
Click Local Server



1. On the right side of the pane, click **On** by IE Enhanced Security Configuration.

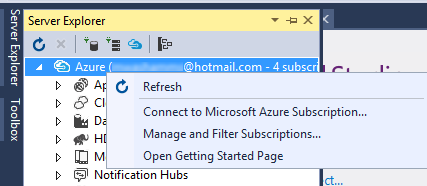


1. Change to **Off** for Administrators and click **OK**.



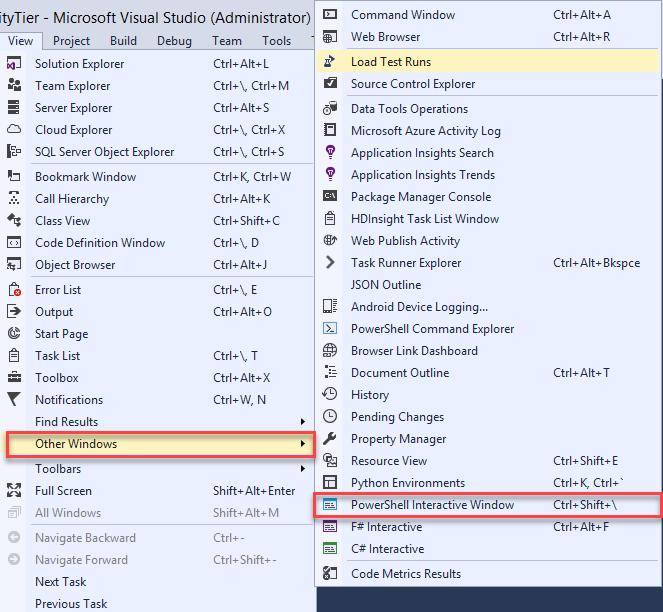
### Task 3: Validate Connectivity to Azure

1. From within the virtual machine, Launch Visual Studio 2015 and validate that you can login with your Microsoft Account when prompted.
2. Validate connectivity to your Azure subscription. Launch Visual Studio, open Server Explorer from the View menu, and ensure that you can connect to your Azure subscription.



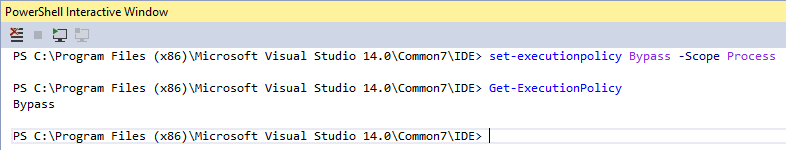
### Task 4: Deploy the ARM template

1. From within your lab machine, create a new folder named **C:\Hackathon**
2. Download the ARM template from <https://opsgilityweb.blob.core.windows.net/cebootcamp-feb-2017/Migating%20EDW%20to%20the%20Cloud%20Hackathon%20-%20Student-Files.zip> and extract to C:\Hackathon.
3. Open the C:\Hackathon\EnvironmentSetup\EnvironmentSetup.sln in Visual Studio
4. Open **PowerShell Interactive Window** within Visual Studio.



1. Run below commands to set PowerShell execution policy to bypass

**set-executionpolicy Bypass -Scope Process**



1. In the Solution Explorer pane, under the EnvironmentSetup solution, right-click the EnvironmentSetup project, navigate to **Deploy -> New**…
2. In the Deploy to Resource Group window, update the following:

* Subscription: **Choose the subscription you are using for your hackathon**
* Resource group: **Create a new resource group called DWEnvironment and select a location that is near you.** Be aware that there is a limit to the number of cores per region available to you. Consider using another region if you already have resources deployed in your nearest region.
* Deployment template: **azuredeploy.json**
* Template parameters file: **azuredeploy.parameters.json**

Click the **Edit Parameters** button.

AdminUserName: **demouser**

AdminPassword: **demo@pass123**

dcipDnsName: **create a unique name**

SQLsalesIPDnasName: **create a unique name**

SQLwarehouseIPDnsName: **create a unique name**

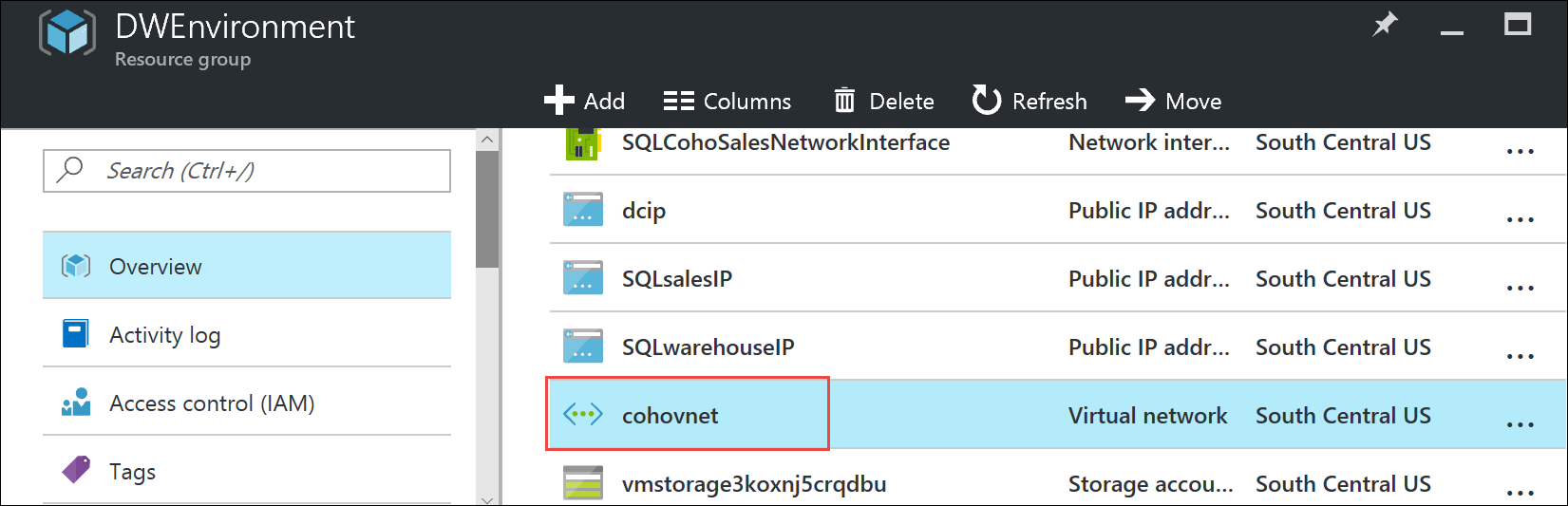
* Artifact storage account: <**Automatically create a storage account>**

|  |  |
| --- | --- |
|  |  |

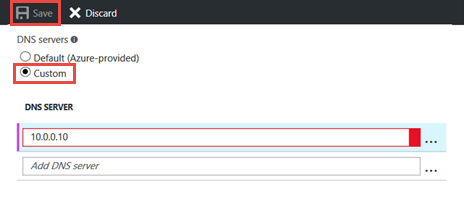
1. Click on **OK**.

### Task 5: Configure DNS and join the domain

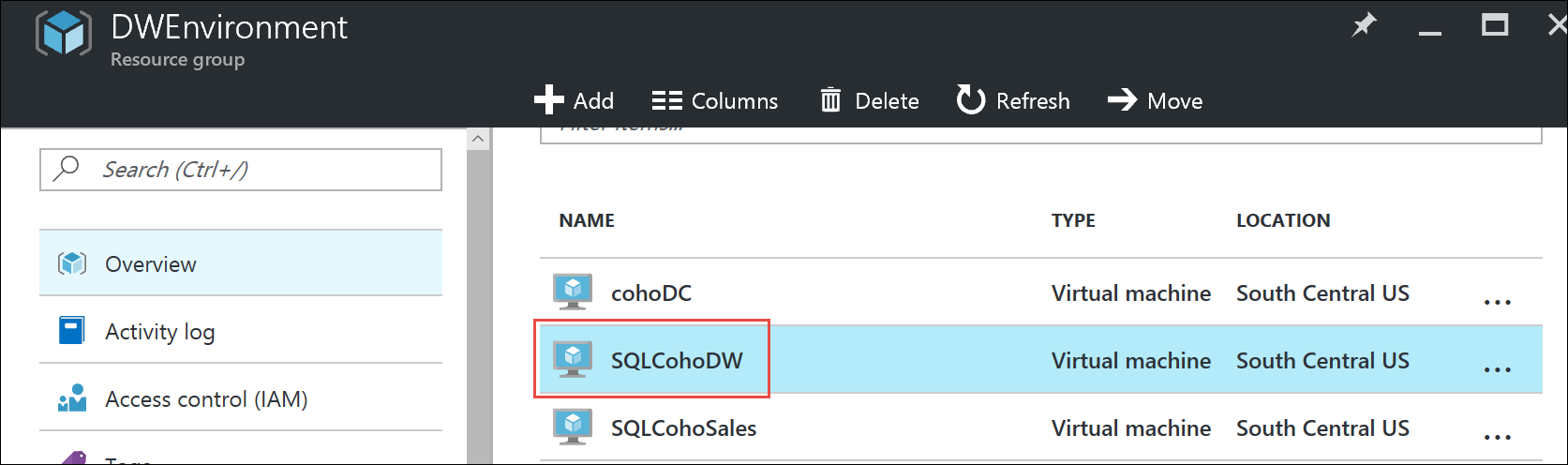
1. Browse to the Azure Portal and authenticate at <https://portal.azure.com/>
2. Navigate to the **DWEnvironment** resource group and select the **cohovnet** virtual network.



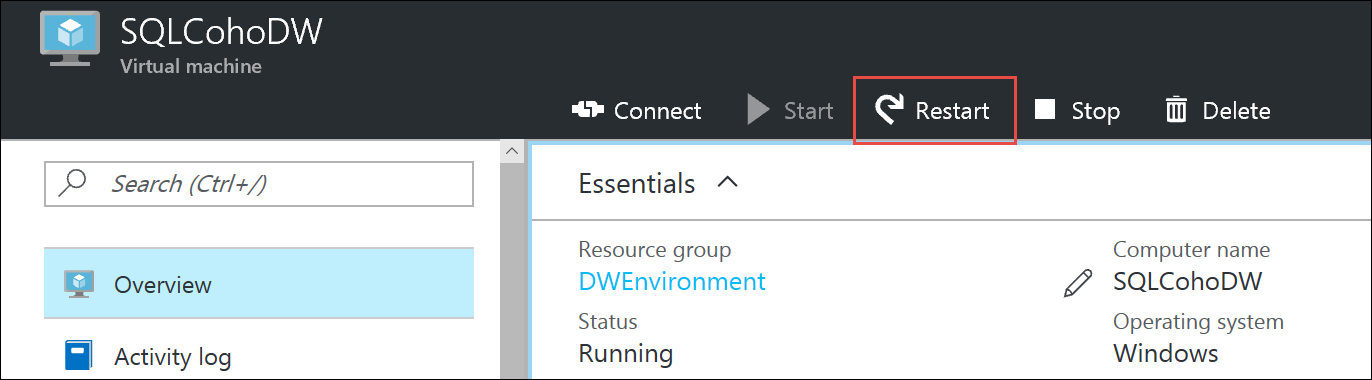
1. On the settings pane select **DNS servers**, then select **Custom DNS** on the DNS servers blade. Type **10.0.0.10** for the Primary DNS server then click the **Save** button.



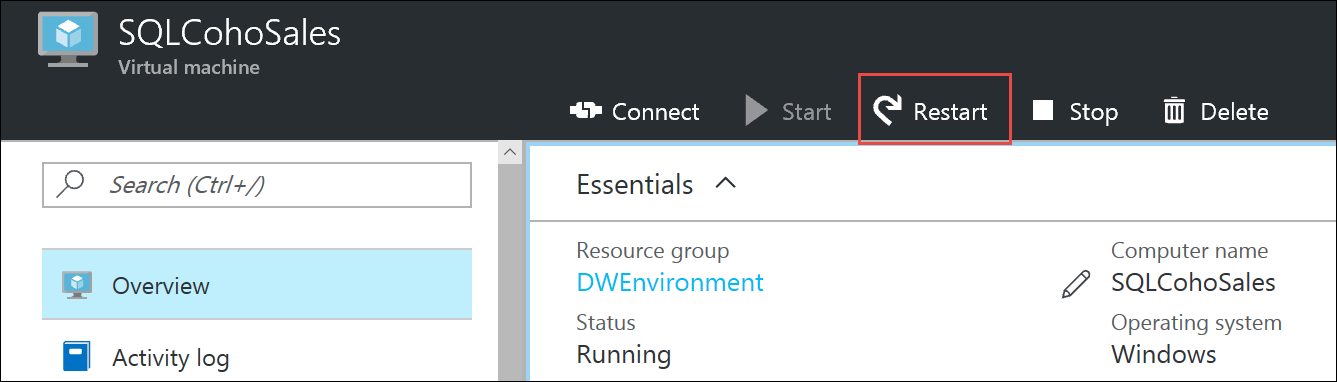
1. Navigate back to your **DWEnvironment** resource group and select the **SQLCohoDW** virtual machine.



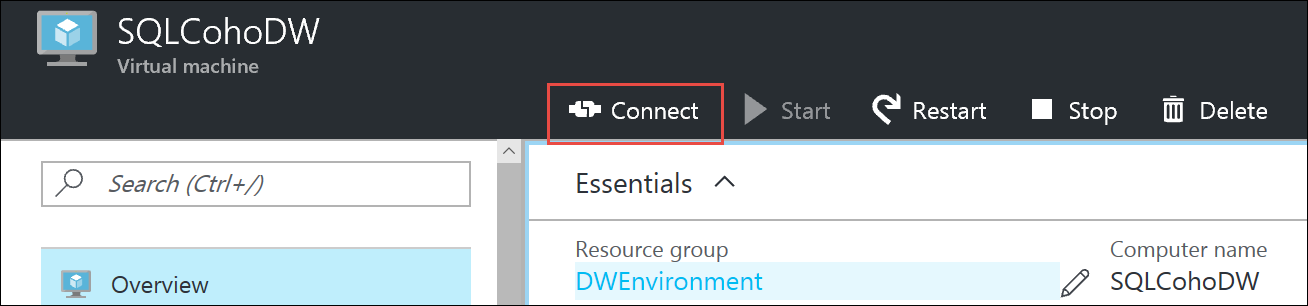
1. On the **SQLCohoDW** virtual machine blade, click the **restart** button, then click **yes** to restart the virtual machine.



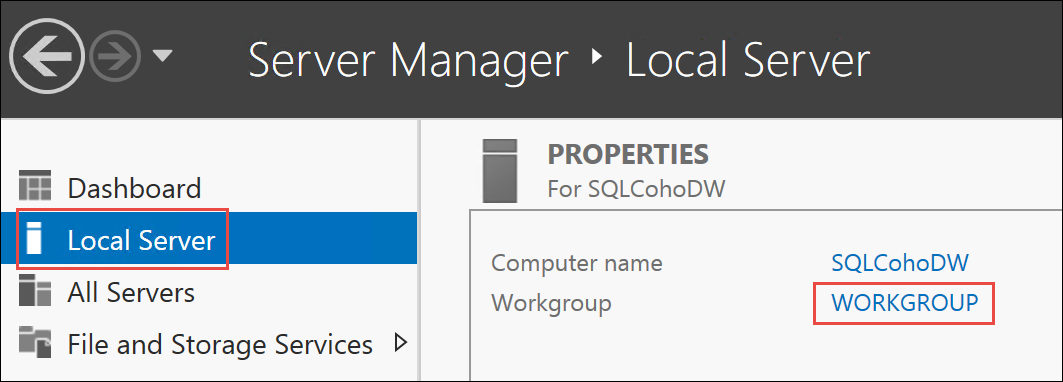
1. Repeat steps 4 and 5 to restart the **SQLCohoSales** virtual machine.



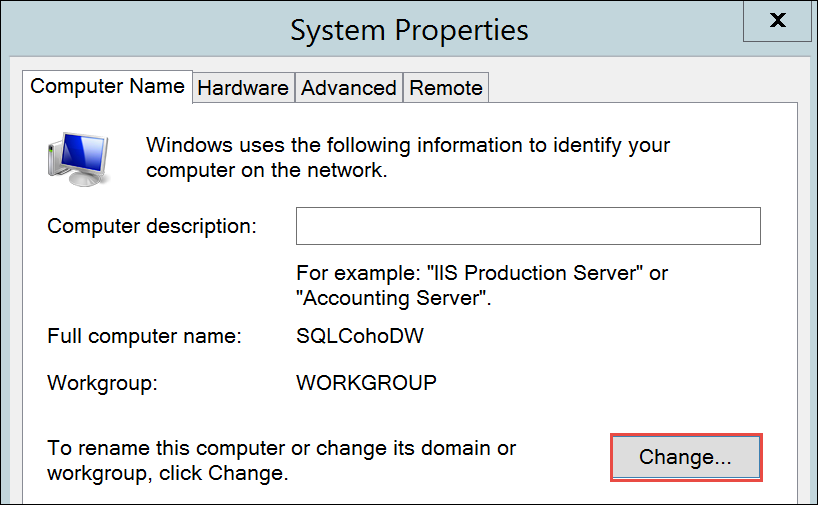
1. Navigate back to the **SQLCohoDW** virtual machine and once **SQLCohoDW** has finished rebooting, connect to it using your **demouser** admin account.



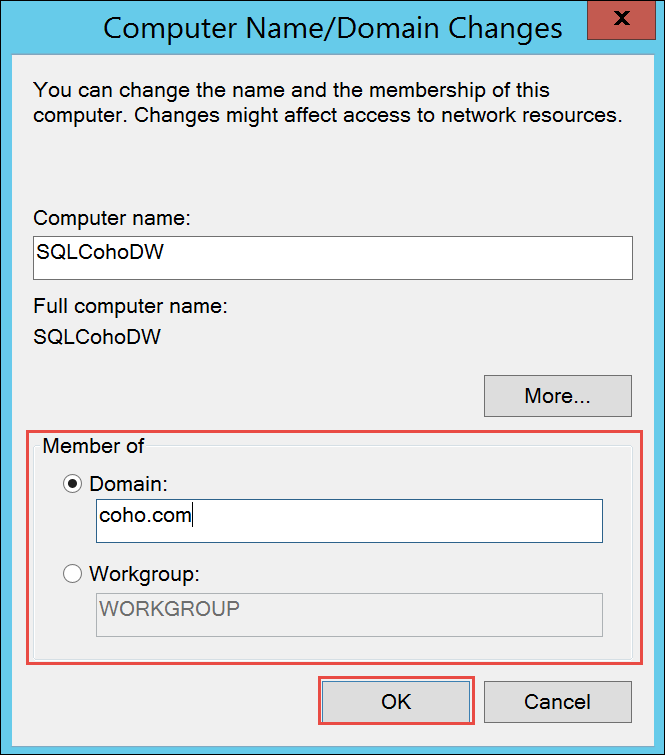
1. Once logged into **SQLCohoDW**, navigate to Server Manager, then click on **Local Server**, then click **WORKGROUP**.



1. On the Computer Name tab, click the **Change** button.



1. Make this virtual machine a member of the **coho.com** domain then click OK.



1. Use **demouser** and **demo@pass123** for the domain administrator account, then click **OK**, **Close**, **Restart** **Now** to reboot the computer.
2. Repeat steps 7 through 11 on the **SQLCohoSales** virtual machine to join **SQLCohoSales** to your domain.

# Migrate to Azure SQL Data Warehouse Hackathon

## Exercise 1: Configure Azure services

Coho has asked you to migrate their current data warehouse environment to Azure SQL Data Warehouse. Before you can begin the migration you need to build out your Azure services you will need to host the environment. You will also need to setup a Power BI Pro account to access on-premises data sources.

### Task 1: Create Azure Services and Accounts

*Tasks to complete*

* Create storage accounts to hold exported data
* Create Azure SQL Datawarehouse
* Create a Power BI Pro account

*Exit criteria*

* Login to the portal, taking note of your endpoints for Azure SQL Data Warehouse and your storage accounts.
* You have a working Power BI Pro account

## Exercise 2: Data and Schema Preparation

Coho is relying on you to migrate the data warehouse to Azure SQL Data Warehouse. One of the most important steps is preparing the data and schema. During this phase you will need to verify compatibility of the schema and data in the CohoDW database hosted on the SQLCohoDW server and make any necessary changes required for a successful migration.

### Task 1: Validate data and schema

#### Tasks to complete

* Validate that there are no compatibility issues with the existing database.
* Identify any architectural changes that need to be made during the migration.

#### Exit criteria

* You have identified that there are no blocking issues with the existing database.
* The schema has been applied without error to the Azure SQL Data Warehouse
* You have identified any architectural changes necessary.

## Exercise 3: Migrate the data to Azure SQL Data Warehouse

The next task is to migrate the data warehouse to Azure SQL Data Warehouse.

### Task 1: Exporting data from the current data warehouse

#### Tasks to complete

* Export the existing data in the correct format using bcp.exe

#### Exit criteria

* Data files for each table exist in the correct format for consumption by Polybase external tables.

### Task 2: Transfer the data to Azure

#### Tasks to complete

* Choose appropriate transfer service to move data from on-premises to Azure Storage.

#### Exit criteria

* All data is uploaded to Azure Storage.

### Task 3: Import the data

#### Tasks to complete

* Create external tables via Polybase to pull data from Azure Storage.
* Import the data into Azure SQL Data Warehouse.

#### Exit criteria

* All data is loaded to Azure SQL Data Warehouse.
* External tables are defined in case a “catch-up” load is needed before the cutover to the new data warehouse.

## Exercise 4: Integrate Azure SQL Data Warehouse and On-Premises Data Sources with Power BI

In this portion of the hackathon, you will setup Power BI to enable self-service BI that will connect to both Azure SQL Data Warehouse and you on-premises OLTP sales database.

Note: This exercise will require a Power BI Pro subscription to complete.

### Task 1: Connect Power BI to Azure SQL Data Warehouse

#### Tasks to complete

* Setup Power BI connectivity to Azure SQL Data Warehouse

#### Exit criteria

* Ability to create visualizations using your Azure SQL Data Warehouse as a data source.

### Task 2: Configure a Power BI Gateway machine to enable on-premises data sources with Power BI

#### Tasks to complete

* Deploy a new virtual machine to host the Power BI Enterprise Gateway
* Install and configure the Power BI Gateway to integrate with an on-premises the SQLCohoSales SQL Server.
* Use Power BI Desktop to create a report that leverages the on-premises SQL Server (SQLCohoSales) and deploy it to the Power BI Service.

#### Exit criteria

* Ability to access the SQLCohoSales SQL Server as a data source from the Power BI Service.

# Migrate to Azure SQL Data Warehouse Hackathon Answers

## Overview

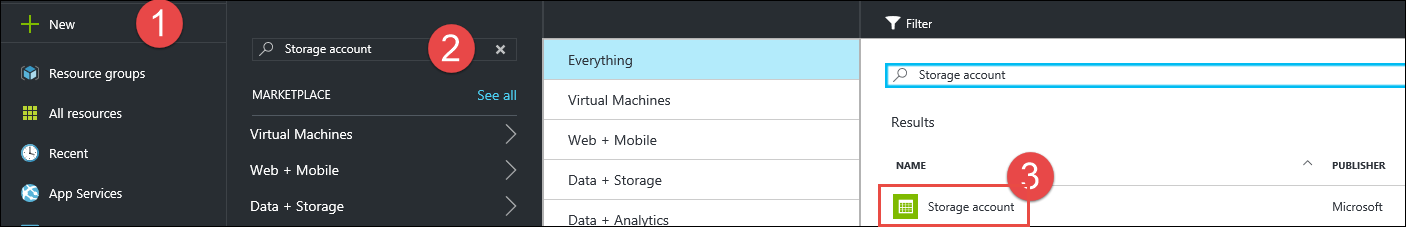
This portion of the lab is designed to help you if you are blocked or have limited experience with database migrations and Azure.

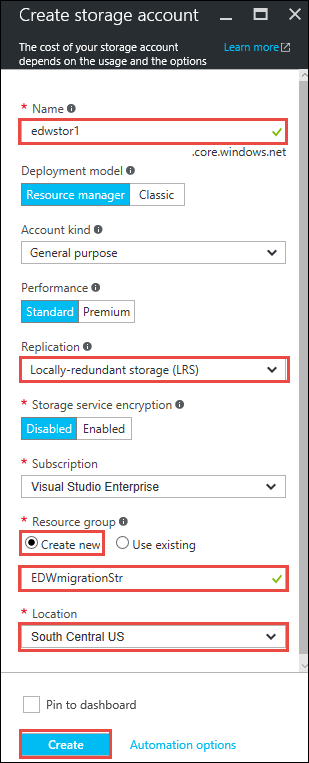
## Exercise 1: Configure Azure services

### Overview

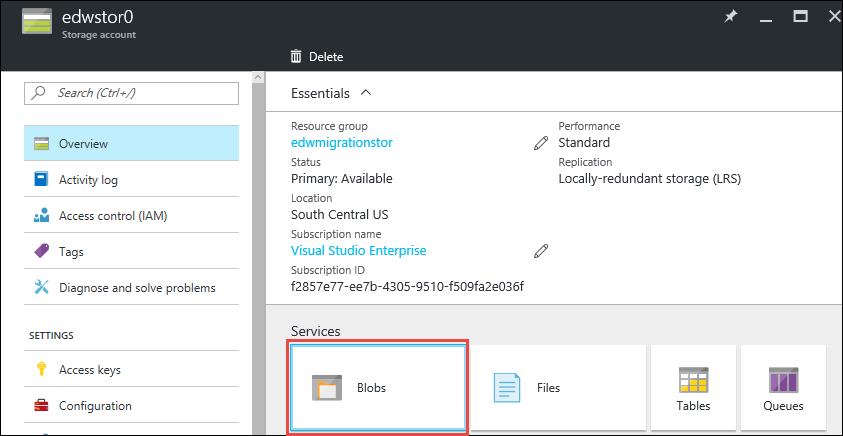
In this exercise, you will create and configure an Azure SQL Data Warehouse, an Azure Storage Account, and a Power BI Account.

### Task1: Create an Azure Storage Account

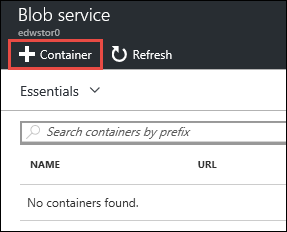
1. Browse to the Azure Portal and authenticate at <https://portal.azure.com/>
2. Click **+New** and type **Storage account** in the search box. Choose **Storage account** from the results.  
   
3. Click **Create** on the Storage account blade. Specify the following information and click **Create**.

* Name: **specify a unique DNS name**
* Deployment model: **Resource manager**
* Account kind: **General purpose**
* Performance: **Standard**
* Replication: **Locally-redundant storage (LRS)**
* Storage encryption: **Disabled**
* Resource group: **Create new** - **EDWmigrationStor**
* Location: **Location near you**  
    
  

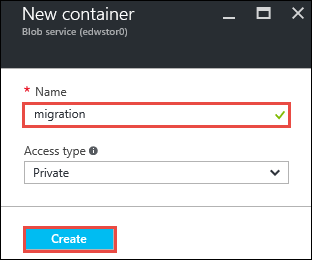
1. Navigate to the new storage account and click **Blobs**.



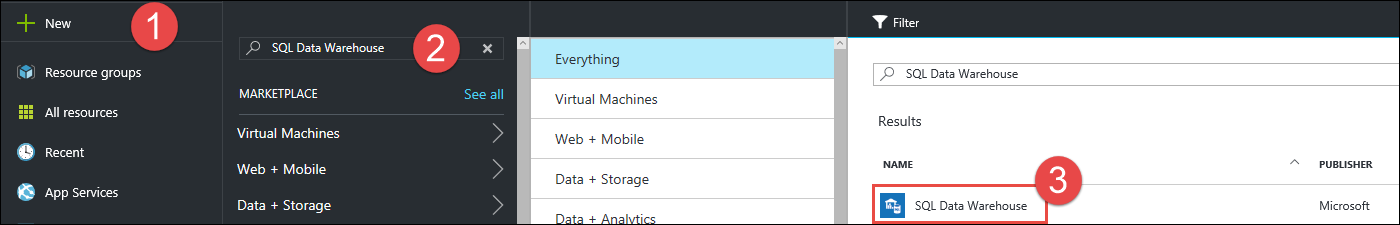
1. On the Blob service blade Click the **+Container** button.



1. On the New container blade type **migration** for the name and then click **Create**.



### Task 2: Create an Azure SQL Data Warehouse

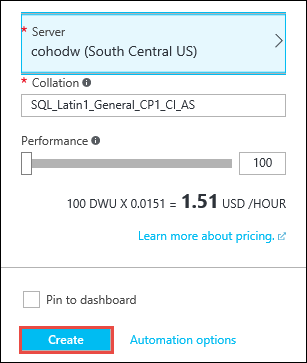
1. Click **+New** and type **SQL Data Warehouse** in the search box. Choose **SQL Data Warehouse** from the results.  
   
2. Click **Create** on the SQL Data Warehouse blade. Specify the following information and then click the **Server** tile.

|  |  |
| --- | --- |
| * Name: **CohoDW** * Resource group: **Create new** - **CohoDWRG** * Performance: **100** |  |

1. On the Server blade, select **Create a new server**, specify the following options and click **Select**.

|  |  |
| --- | --- |
| * Server name: **Choose a unique name** * Server admin login: **demouser** * Password: **demo@pass123** * Location: **Same location as your source** * Allow azure services: **checked** |  |

1. On the SQL Data Warehouse blade click **Create**.

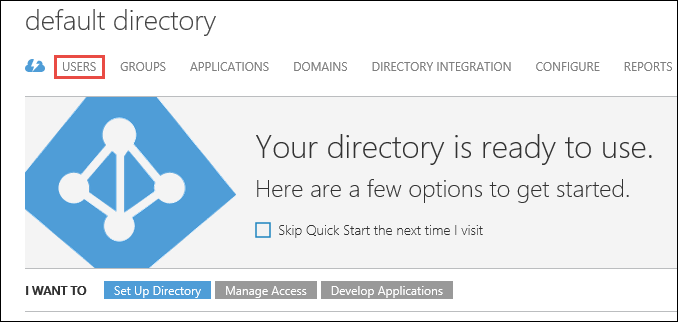


### Task 3: Create a Power BI Pro account

1. Note that these steps describe creating an account that can be enabled for a Power BI Pro trial from within an Azure Pass subscription or MSDN subscription. If you are using an organizational account for this hackathon some of these steps may be unnecessary or impossible depending on your level of access. If your existing account is already enabled for Power BI Pro then you may skip this task.
2. Open a browser and navigate to the Azure Classic Portal at <http://manage.windowsazure.com>
3. Scroll down through the menu along the left side of the browser and click on **Active Directory**.



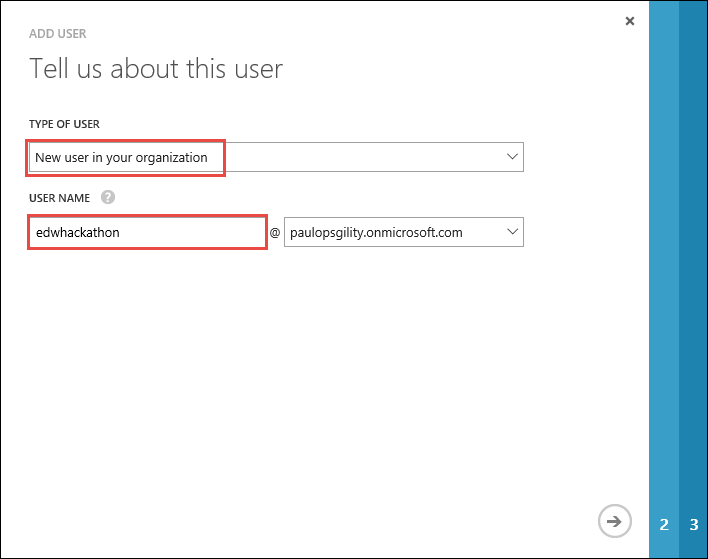
1. If you have multiple subscriptions, you will need to choose the correct Azure Active Directory Tenant. By default, your tenant is called **Default Directory**.
2. Once you have determined the correct AD tenant, click on Users.



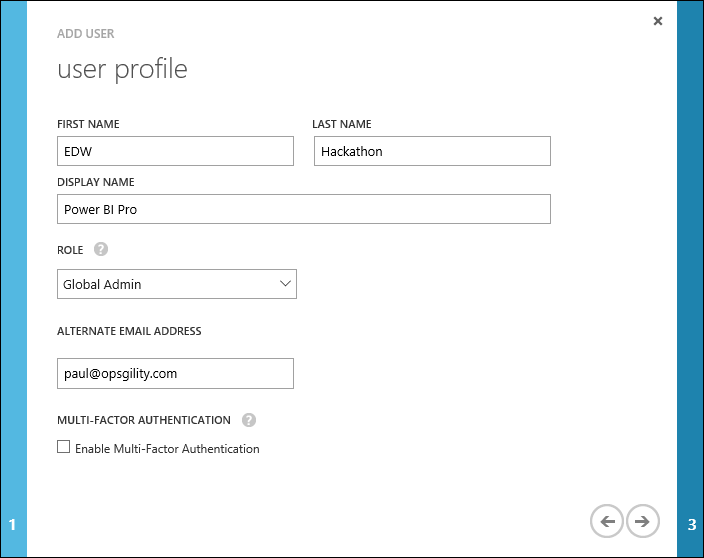
1. At the bottom of the users screen, click on the **Add User** button.



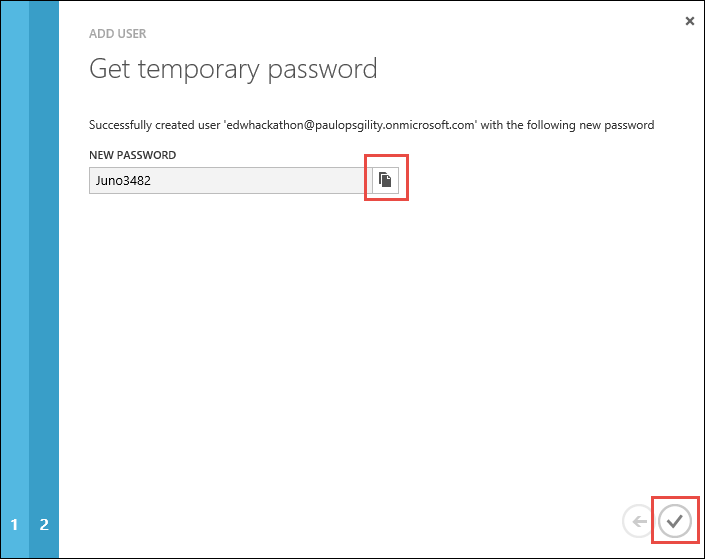
1. Set the type of user to **New user in your organization** and choose a user name for the user. This will be the user name you use to login to Power BI throughout the hackathon.



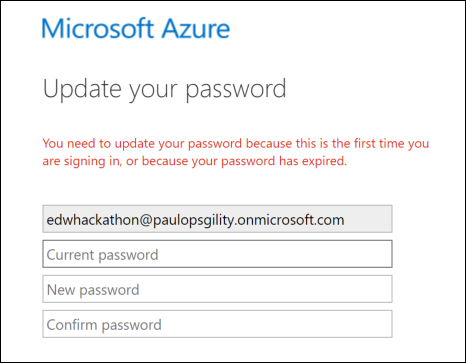
1. Fill in the information on the **user profile** screen. Set the role to **Global Admin**, use your email address for the alternate email address.



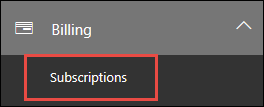
1. On the **Get temporary password** screen click the **Create** button. Once the user has been created, copy the new password and paste it into notepad.exe, also note the full email address of the new user, then click the **Close** button.



1. Open a new **InPrivate** browser and login to <http://portal.azure.com>. You will be prompted to change your password.



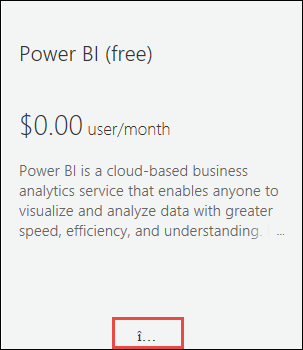
1. After changing your password, navigate to the Office 365 admin center at <https://portal.office.com/admin/default.aspx>
2. On the navigation pane, expand **Billing** and click **Subscriptions**.



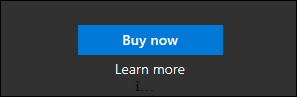
1. Click the **+ Add subscriptions** button.



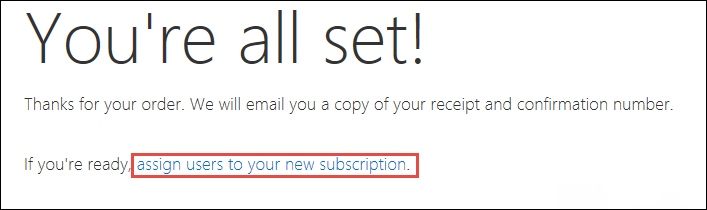
1. Scroll down to Other Plans, and find **Power BI (free)** and hover over the **ellipses** at the bottom.



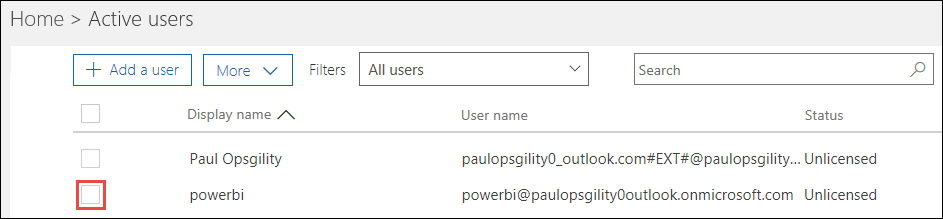
1. Click the **Buy now** button.



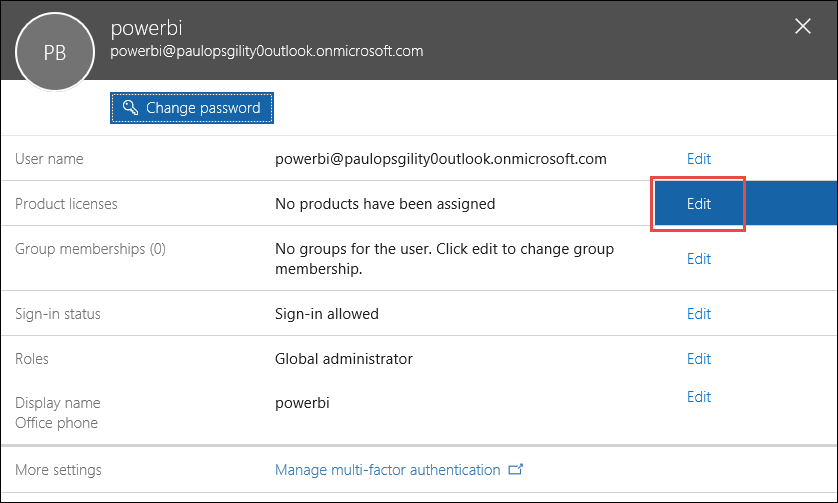
1. Accept the defaults and fill out the remaining billing information. The total should be $0.00 and choose to be invoiced for the bill.
2. Click on **assign users to your new subscription**.



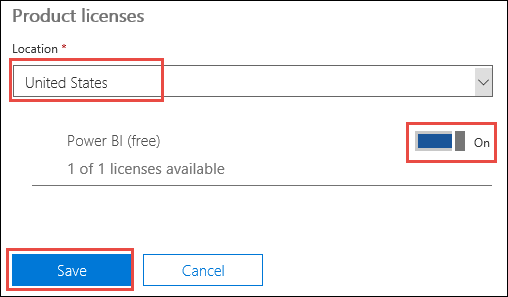
1. Check the box next to the account your created.



1. Click **Edit** Product licenses.



1. Set the **location** to your location, set the Power BI (free) switch to **On**, then click **Save**.



1. Close all of the account editor windows.
2. Navigate to <http://powerbi.com> and click **Sign in**. If you are prompted for additional sign-in then enter your sign-in if necessary. If you are prompted to invite more people, choose **skip**. This should bring you to the Power BI Service. Later in this hackathon you will enable the trial for Power BI Pro.



### Summary

In this exercise you configured your Azure services in preparation for your warehouse environment migration. You configured an Azure Storage account to temporarily store your data before importing, you then created an Azure SQL Data Warehouse which you will use as a migration target and finally you created a Power BI account which you will use to integrate and modernize your data warehouse environment.

## Exercise 2: Data and schema preparation

### Overview

Coho is relying on you to migrate the data warehouse to Azure SQL Data Warehouse. One of the most important steps is preparing the data and schema. During this phase you will need to verify compatibility of the schema and data and make any necessary changes required for a successful migration.

### Task 1: Check compatibility with the Data Warehouse Migration Utility (Preview)

1. In the Azure portal, navigate to your **DWEnvironment** resource group, then connect to the **SQLCohoDW** virtual machine.
2. Download the Data Warehouse Migration Utility (Preview) from http://www.microsoft.com/en-us/download/details.aspx?id=49100 and extract the contents.

Note: If you get an internet explorer message ‘Your current security settings do not allow this file to be downloaded’, Go to Internet explorer Menu 🡪 Internet Options 🡪 Security Tab 🡪 Click on Internet 🡪 Select Custom Level 🡪 Select ‘Enable’ in Downloads\File Download section.

1. Run the Windows Installer file and follow the steps to install the Data Warehouse Migration Utility.
2. Open the Data Warehouse Migration Utility by double-clicking the icon on your desktop.



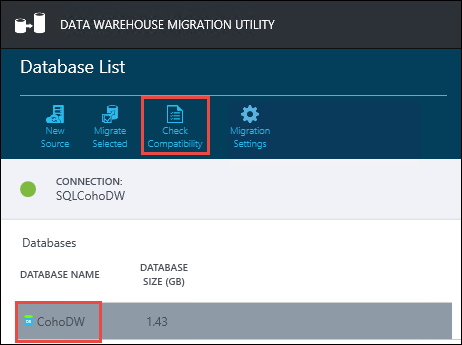
1. The Data Warehouse Migration Utility will open on the source and destination selection window. Review the source and destination. Notice that the Data Warehouse Migration Utility supports migrations from SQL Server and Azure SQL Database. Make sure that Source Type is set to **SQL Server** then click **Next**.



1. On the Migration Source window, change the Server Name to **SQLCohoDW**, set Authentication to Windows, and then click **Connect**.



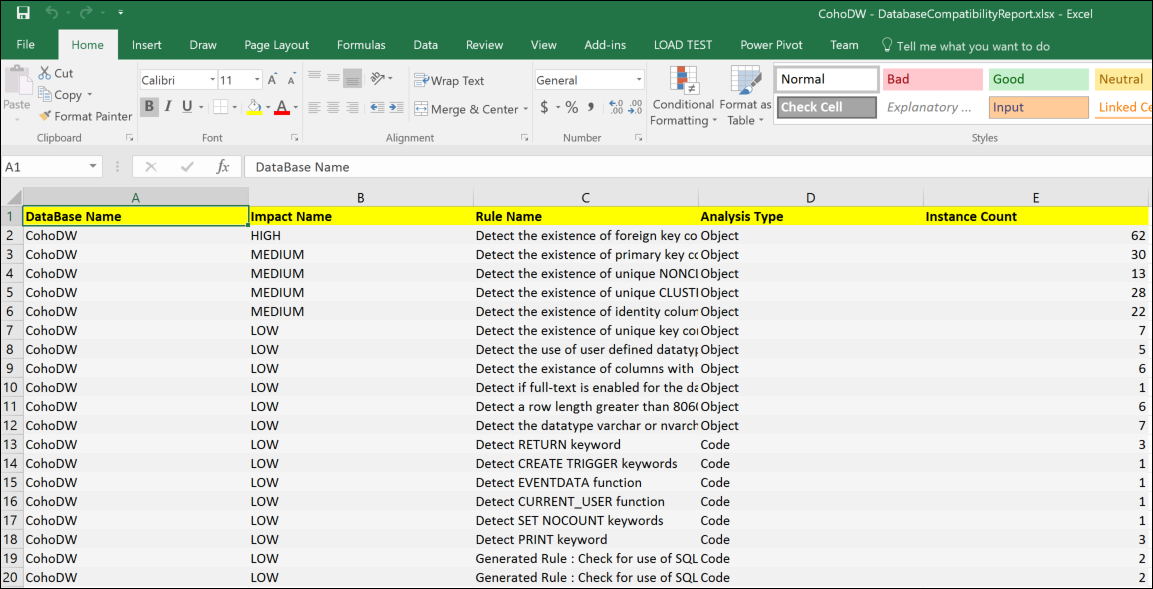
1. Select the **CohoDW** database and then click the Check Compatibility button.



1. Choose an output location for your compatibility report and click the **Save** button. In the File Saved Successfully popup choose **No**.

NOTE: You will not be able to open the compatibility report on the server as it does not have Excel installed.

1. Open the file location that your report was saved (by default it will be in your My Documents folder) and copy and paste the file to your local machine.
2. Open the file on your local machine to review the output (your output may be different than what you see below).



NOTE: The Data Warehouse Migration Utility is currently in preview and the feature-set is incomplete and may have additional issues such as false negatives. It can be used to easily automate the migration of smaller databases but be aware that it does not compress files, move data to Azure storage or use Polybase for import.

### Task 2: Validate schema and data

1. In the Azure portal, navigate to your **DWEnvironment** resource group, then connect to the **SQLCohoDW** virtual machine.
2. Launch SQL Server Management Studio and open a **New Query** window.



1. Run the following query to check for data incompatibility and potential data length issues.

USE CohoDW

GO

SELECT t.[name] as [Table],

c.[name] as [Column],

c.[system\_type\_id],

c.[user\_type\_id],

y.[is\_user\_defined],

y.[name]

FROM sys.tables  t JOIN sys.columns c ON t.[object\_id] = c.[object\_id]

JOIN sys.types y ON c.[user\_type\_id] = y.[user\_type\_id]

WHERE y.[name] IN ('geography', 'geometry', 'hierarchyid', 'image', 'ntext',

'numeric', 'sql\_variant', 'sysname', 'text', 'timestamp', 'uniqueidentifier', 'xml')

OR (y.[name] IN ('varchar', 'varbinary') AND ((c.[max\_length] = -1) or

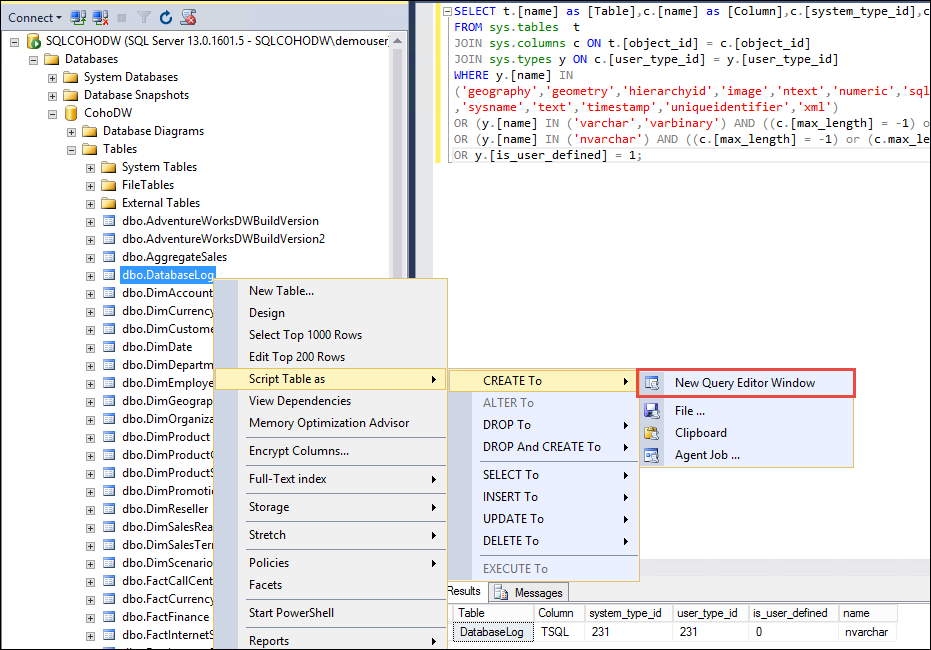
(c.max\_length > 8000)))

OR (y.[name] IN ('nvarchar') AND ((c.[max\_length] = -1) or

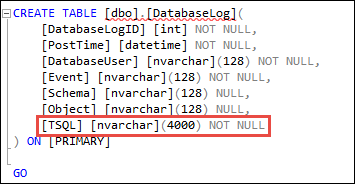
(c.max\_length > 4000))) OR y.[is\_user\_defined] = 1;

NOTE: A full list of incompatible table features and data types can be found in the migration documentation at <https://azure.microsoft.com/en-us/documentation/articles/sql-data-warehouse-overview-migrate/>

1. The output of the query shows the table and column but not the reason for the incompatibility. To gain more insight into the reason you can script the table out by expanding the CohoDW database in Object Explorer, right-click the table, select Script Table as -> CREATE To -> New Query Editor Window.



1. From the script of the table you can see that the ‘TSQL’ column of the ‘DataLog’ table has a data type nvarchar(4000) which is equivalent to 8000 bytes which means that the data may potential exceed the maximum data size.



1. Before we fix this column we must validate that none of the data would be truncated. Check the maximum actual data size with the following query.

SELECT MAX(DATALENGTH([TSQL]))

FROM DatabaseLog

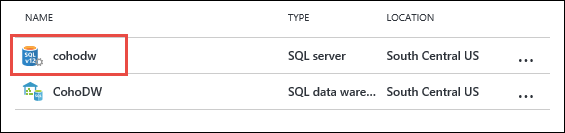
The result is 3034 which means that our longest value is 3034 bytes or 1517 characters, leaving us plenty of space to modify the column with no loss of data.

1. Modify the column by executing the following query:

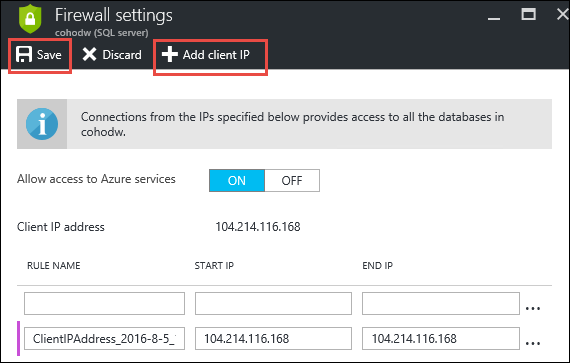
ALTER TABLE dbo.DatabaseLog ALTER COLUMN [TSQL] nvarchar(2000)

### Task 3: Prepare Azure SQL Data Warehouse and migrate schema

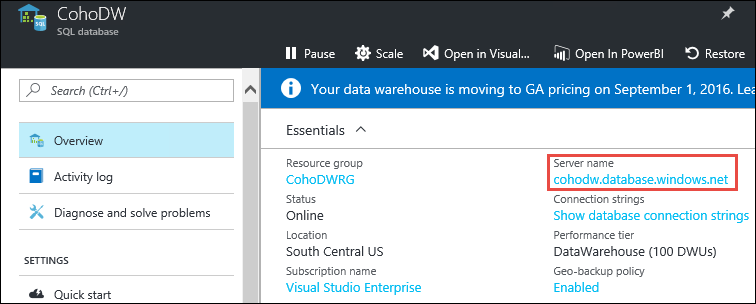
1. In the Azure portal, navigate to your **DWEnvironment** group, then connect to the **SQLCohoDW** virtual machine.
2. Open Internet Explorer and connect to the Azure Portal.
3. Navigate to your **CohoDWRG** resource group, then click on the cohodw logical SQL Server that hosts your Azure SQL Data Warehouse.



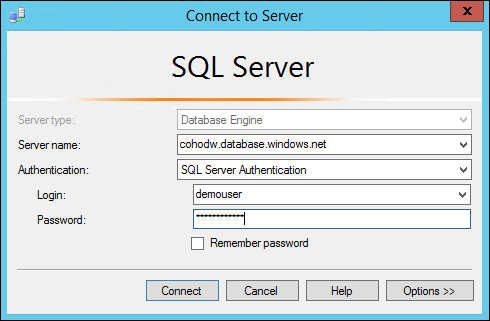
1. In the settings blade, click on **Firewall**.
2. In the cohodw - Firewall blade, click the **+Add client IP** button, then click the **Save** button.



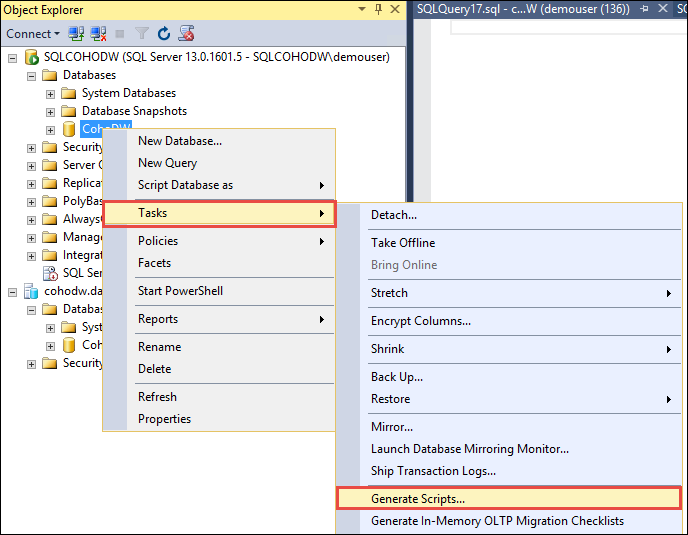
1. Back in the **CohoDWRG** resource group, select the **CohoDW** data warehouse and copy the server name.



1. Open SQL Server Management studio, click the connect button in Object Explorer and connect to your SQL Data Warehouse using the **demouser** account and password to verify connectivity.



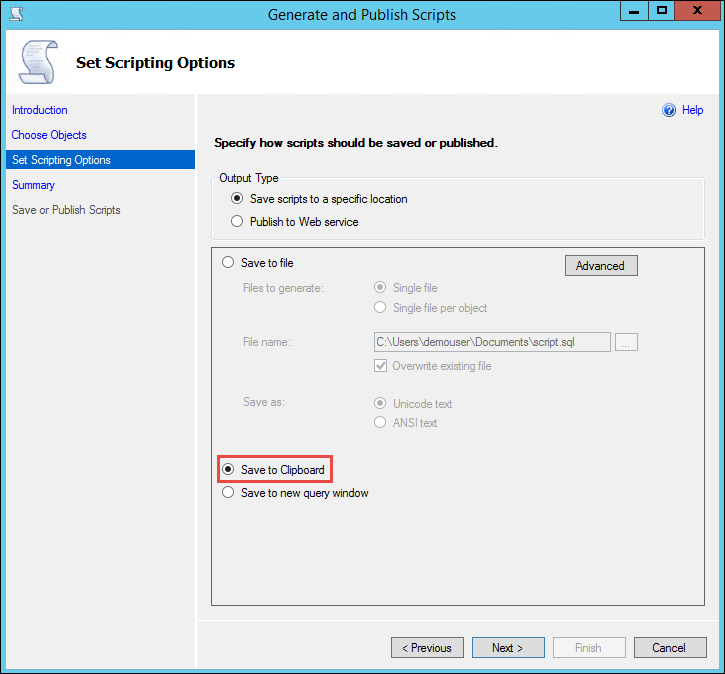
1. Expand your SQL Data Warehouse, expand databases, select the CohoDW database then click the **New Query** button.
2. Connect to the local SQLCohoDW instance of SQL Server, right click your local copy of CohoDW, and select **Tasks** -> **Generate Scripts** to launch the Generate and Publish Scripts wizard.



1. Click Next on the Introduction screen.
2. On the Choose Objects screen, Select the **Select specific database objects** radio button and check **Tables**, then click Next.



1. On the Set Scripting Options screen. Select the **Save to Clipboard** radio button.



1. Accept the defaults for the remaining screens and click **Finish**.
2. Paste the results into the Query windows connected to your Azure SQL Data Warehouse.
3. This script still needs to be modified before it will run correctly in Azure SQL Data Warehouse because some T-SQL syntax is not supported in Azure SQL Data Warehouse. Make the following updates to the script:

* Execute a Find and Replace on your script to replace all occurrences of “ON [PRIMARY]” with “” to remove them from the script.
* Execute a Find and Replace on your scrfipt to replace all occurrences of “USE” with “--USE” to comment out those lines.
* Remove any occurrences of SET ANSI\_PADDING OFF from the script.

.

1. Run the script by clicking the Execute button. This will use the default options to create tables, Clustered Columnstore Indexing and ROUNDROBIN distribution.



1. Execute the following query to verify that your tables were created. There should be 33 rows returned.

SELECT \* FROM sys.tables

### Summary

In this exercise, you prepared and verified your schema with a combination of the Data Warehouse Migration Utility and T-SQL scripts to analyze the schema and data. You then scripted your database, made the necessary changes and applied your script to Azure SQL Data Warehouse.

## Exercise 3: Migrate the data to Azure SQL Data Warehouse

### Overview

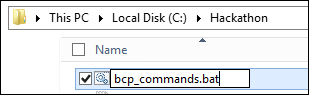
This exercise is focused on migrating the data from your existing data warehouse into SQL Data Warehouse. We will be pulling the data and then uploading it to an Azure storage account. We will then import the data via Polybase.

### Task 1: Exporting data from your current data warehouse.

1. Connect to your **SQLCohoDW** virtual machine.
2. Open the  **C:\Hackathon\bcp\_commands.txt** file. These are the bcp commands for each of the tables you need to migrate. The line below is an example. Notice the bcp commands all use the -C 65001 parameter. This indicates that the output will be in UTF-8 which is required by Polybase. This code page is only an option with bcp.exe that ships with SQL Server 2016 tools. If you are using an older version of bcp you will have an additional step to convert to UTF-8.

bcp "select [ScenarioKey],REPLACE([ScenarioName],'|','||') from [CohoDW].[dbo].[DimScenario]" queryout "F:\Migration/dbo.DimScenario.txt" -q -c -C 65001 -t "|" -r "\n" -S SQLCohoDW -T

1. Close the file after you are done reviewing it. Change the file name to **bcp\_commands.bat**.

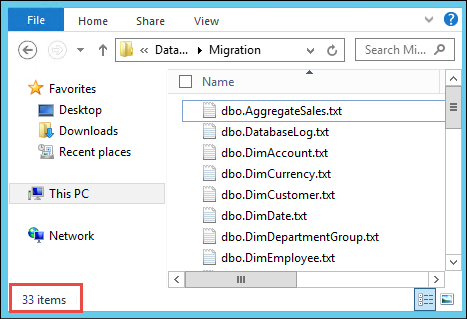


1. Open a command prompt and execute **C:\Hackathon\bcp\_commands.bat**



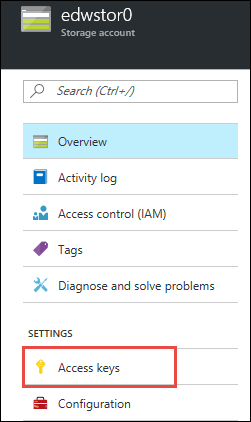
Note that in a production environment you would likely make some effort to parallelize the execution of the various bcp commands. For larger tables, you also might parallelize the export from a single table.

1. Navigate to the **F:\Migration** folder. If the commands completed successfully you will have **33 files**.

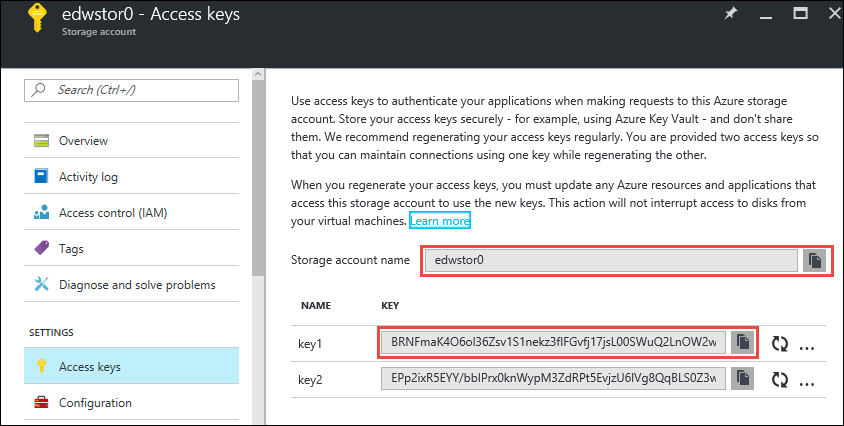


### Task 2: Transfer your data to Azure

1. From your SQLCohoDW virtual machine navigate to you C:\Hackathon folder and double-click **MicrosoftAzureStorageTools.msi** to install AzCopy.
2. In the Azure Portal navigate to your **EDWmigrationStor** resource group and click on your storage account.
3. In the Storage account blade, under settings, click on **Access keys**



1. Copy the storage account name and access key1 and paste into notepad for later use.



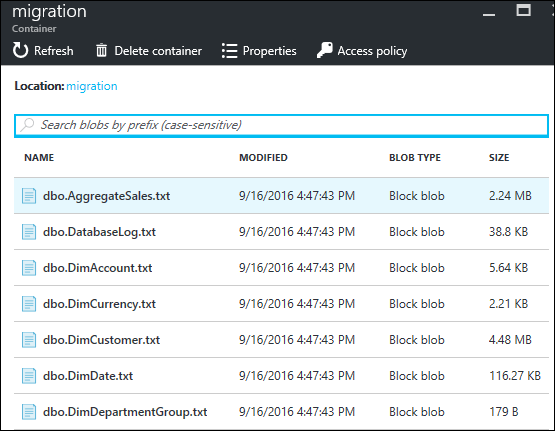
1. Open a command prompt and navigate to the **C:\Program Files (x86)\Microsoft SDKs\Azure\AzCopy** folder.
2. Execute the following command to begin copying your data files to Azure (all of the text is a single command)

AzCopy /Source:"F:\Migration" /Dest:https://<YourStorageAccount>.blob.core.windows.net/migration /DestKey:<YourStorageAccountKey> /pattern:\*.txt /NC:2

1. Confirm that all 33 files were transferred successfully.



And that the files are in the correct storage container by navigating to your storage account, clicking on blobs, then clicking on your container.



1. Open SQL Server Management Studio and connect to your SQL Data Warehouse.
2. Open a new query window and execute the following command to create a master key for your database.

CREATE MASTER KEY

1. Execute the following to create a database scoped credential that you will use to store the access key to the migration storage account.

CREATE DATABASE SCOPED CREDENTIAL MigrationCredential

WITH IDENTITY = '<YourStorageAccountName>' , SECRET = '<YourStorageAccountKey>'

1. Create an external file format by executing the following query. The external file format defines the external storage and its layout.

CREATE EXTERNAL DATA SOURCE MigrationStor WITH (TYPE = HADOOP ,

LOCATION=

'wasbs://<YourStorageContainerName>@<YourStorageAccountName>.blob.core.windows.net',

CREDENTIAL = MigrationCredential);

1. Create an external file format by executing the following query. The external file format defines the external storage and its layout.

CREATE EXTERNAL FILE FORMAT MigrationFiles WITH(FORMAT\_TYPE = DelimitedText,

FORMAT\_OPTIONS (FIELD\_TERMINATOR = '|'));

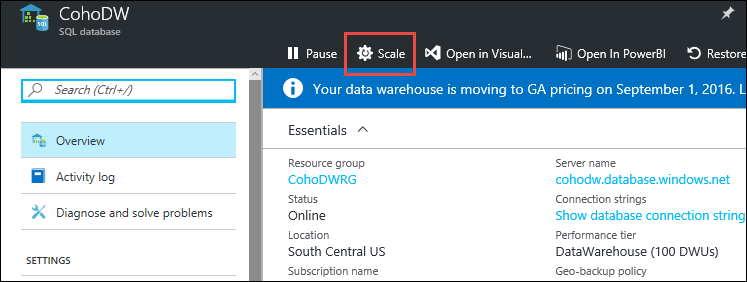
1. Open the C:\Hackathon\CreateExternalTables.sql file in SQL Server Management Studio.
2. This file contains all of the external table definitions for our tables and directly leverages the external data source and external file format we created above. Click Execute to create the external tables.



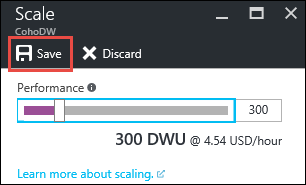
1. Run the following code to verify that 33 tables were created.

SELECT \* FROM SYS.TABLES WHERE is\_external = 1

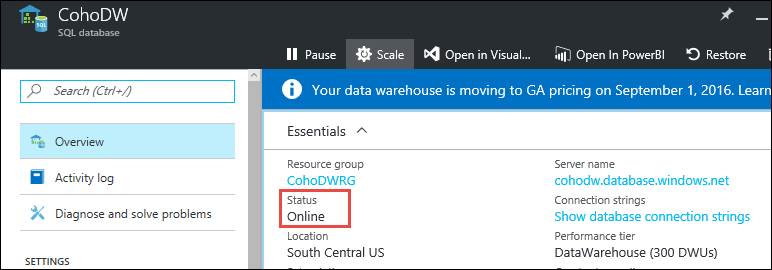
1. We are about to load the warehouse, to maximize performance of the load. To do this we will adjust the performance of the Azure SQL Data Warehouse by adjusting the scale slider in the Azure Portal. Navigate to your CohoDW SQL Data Warehouse and click on the Scale button.



1. In the Scale blade, adjust the Performance slider to 300 and click the Save button.



1. Wait until the scaling process is complete to move on to the next step. You can view the SQL Data Warehouse status in the portal. The status will change from “Scaling” to “Online” when scaling is complete.



1. Open the C:\Hackathon\LoadData.sql file in SQL Server Management Studio.
2. The commands in this file insert data extracted directly from the data files stored in Azure Storage via the external tables we defined in the previous steps. Click execute to begin the data load.



1. After your data is uploaded you can select data from any of the tables to verify success. In production environments, you would go through a much more thorough data validation process.
2. **Scale you SQL Data Warehouse back down to 100 to prevent excessive cost**.

## Exercise 4: Integrate Azure SQL Data Warehouse and On-Premises Data Sources with Power BI

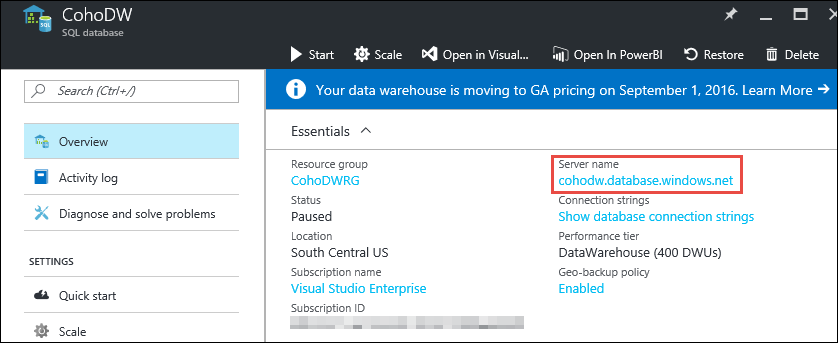
### Overview

In this portion of the exercise, you will setup integration with Power BI.

Note: To connect Power BI dashboards and reports directly to Azure SQL Data Warehouse requires a Power BI Pro subscription. Please see the instructions earlier in this hackathon to configure an account for Power BI and enable a Power BI Pro trial if you do not already have one.

### Task 1: Connect Power BI to Azure SQL Data Warehouse

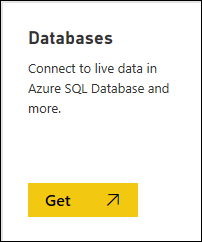
1. Connect to the Azure Portal and navigate to your Azure SQL Data Warehouse.
2. Make note of your server name to use in your data source configuration later in this task.



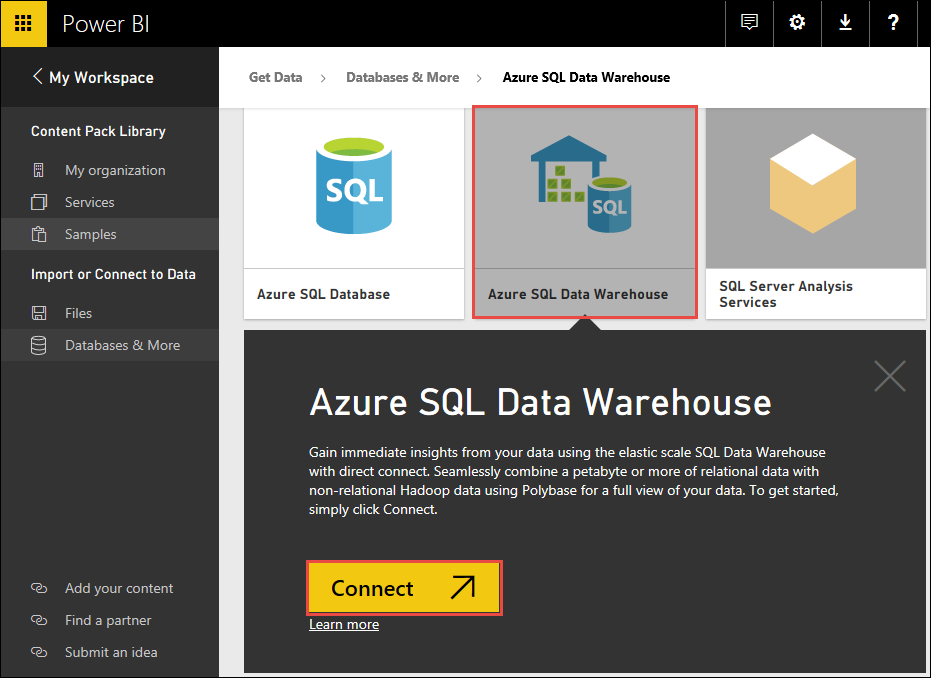
1. Open a web browser and navigate to **powerbi.com** and click **Sign in** in the upper right corner of the screen.



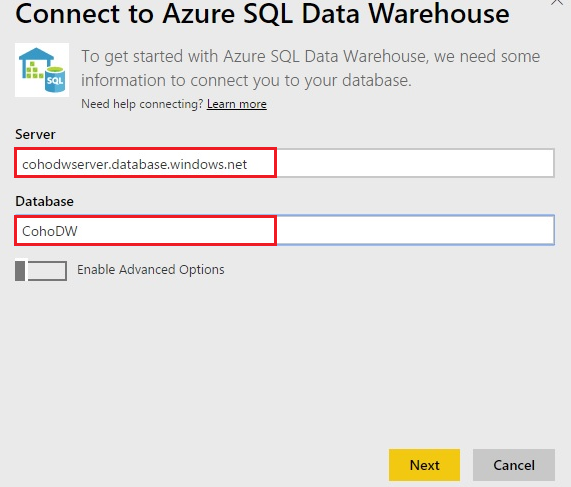
1. Sign in with the organizational account that you are using to access Power Bi Pro.
2. Click on the **Get Databases** button.



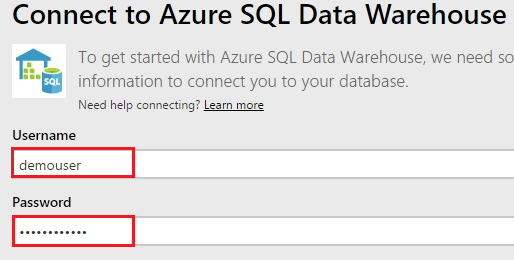
1. On the **Get Data > Databases & More** screen, click on **Azure SQL Data Warehouse** the click **Connect**. If you are presented with a message to indicating that you need Power BI Pro, click the option to enable the trial and a Power BI Pro trial will be started for this account.



1. Supply your Azure Data Warehouse server details:



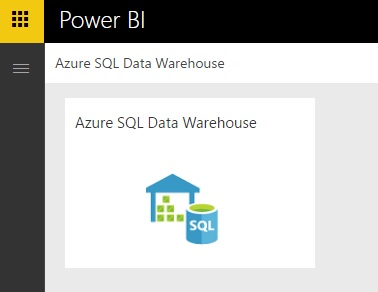
1. Provide the Azure SQL Data Warehouse credentials: **demouser** and **demo@pass123**



1. It will take 1 to 2 minutes to load the data.



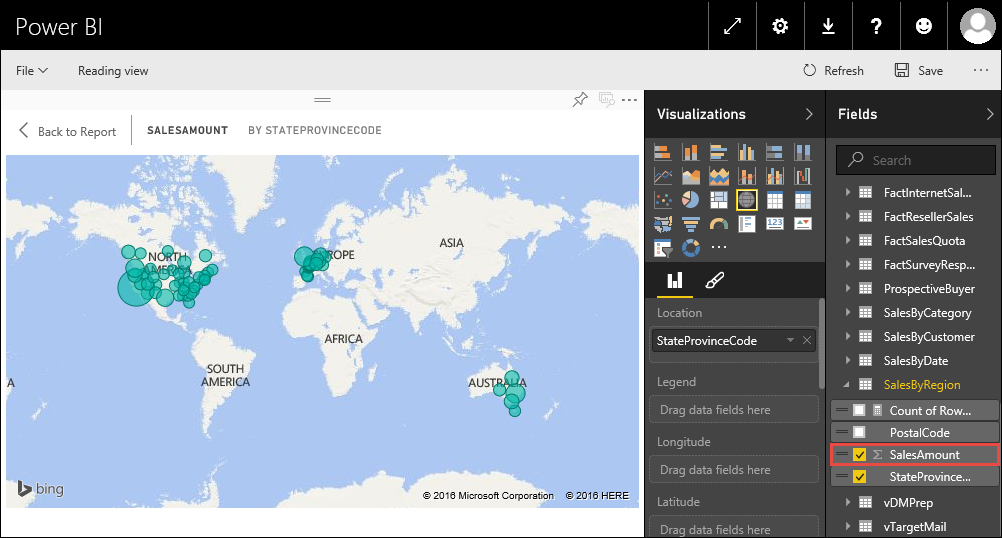
1. Once added SQL Data Warehouse will be shown as below. Click on the tile.



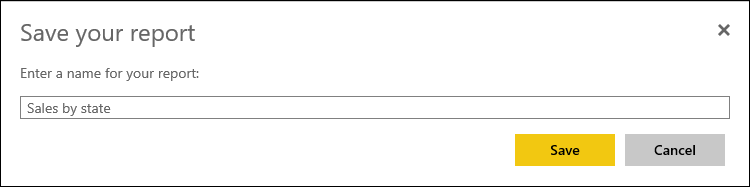
1. In PowerBI, in the Fields blade, expand the **SalesByRegion** table and check the box next to **StateProvinceCode**. This will automatically launch the map visualization because PowerBI is smart enough to understand that this is geographic data.



1. The circles that PowerBI adds to the map are simply every state or province in which Coho had sales. Let’s add the sales amount to this to make the map a little more interesting. Add the **SalesAmount** from the **SalesByRegion** table by putting a check next to it. The circles on the map will change in size to reflect the sum of all sales in that particular State/Province.

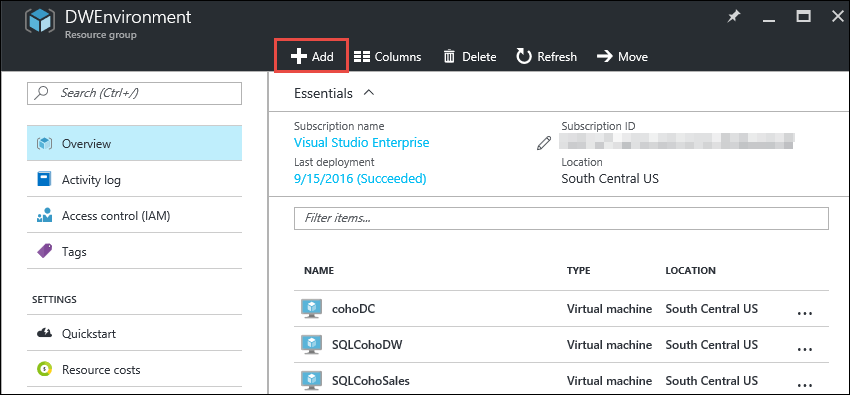


1. Click the **Save** button in the top right of your screen, name your report **Sales by state** and click **Save**.

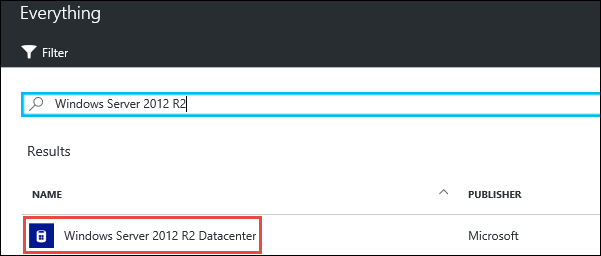


### Task 2: Configure a Power BI Gateway machine to enable on-premises data sources with Power BI.

1. Open the Azure Portal and navigate to your DWEnvironment resource group where you deployed your “on-premises” environment.
2. In the DWEnivronment blade, click the **Add+** button.

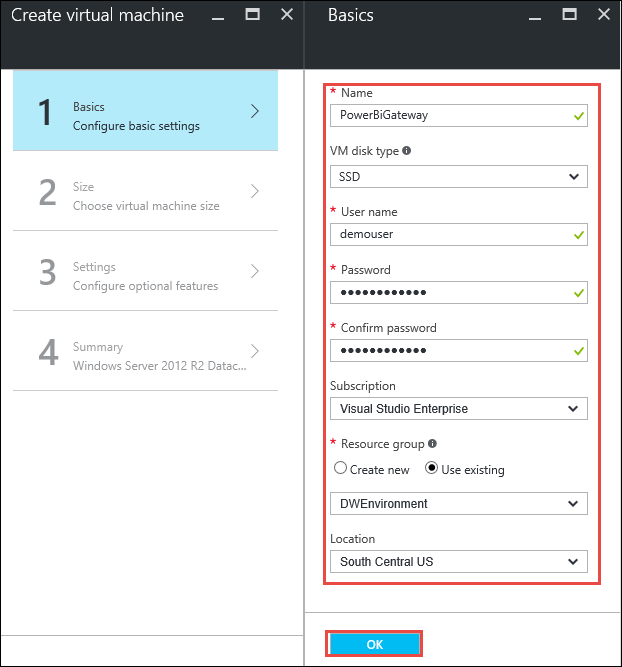


1. In the search field type **Windows Server 2012 R2** and hit enter, then choose **Windows Server 2012 R2 Datacenter** from the results.

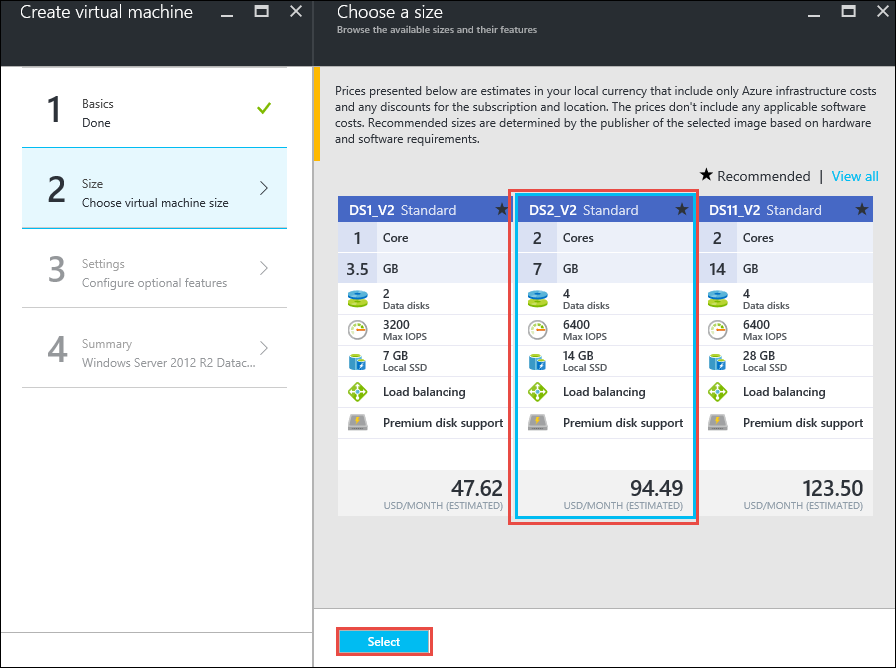


1. On the Windows Server 2012 R2 Datacenter blade click **Create**.
2. On the Create virtual machine Basics blade, fill in the following information and then click **OK**:

* Name: **PowerBiGateway**
* VM disk type: **SSD**
* Username: **demouser**
* Password: **demo@pass123**
* Resource Group: **Use existing - DWEnvironment**
* Location: **Same location you used for this deployment**



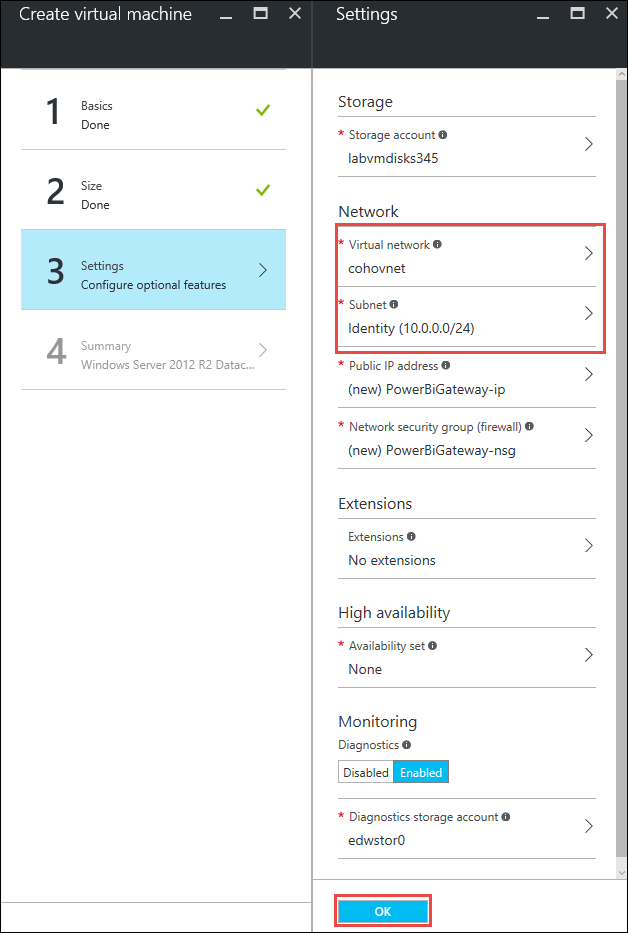
1. On the Create virtual machine, Choose a size blade, select **DS2\_v2** and click **Select**.



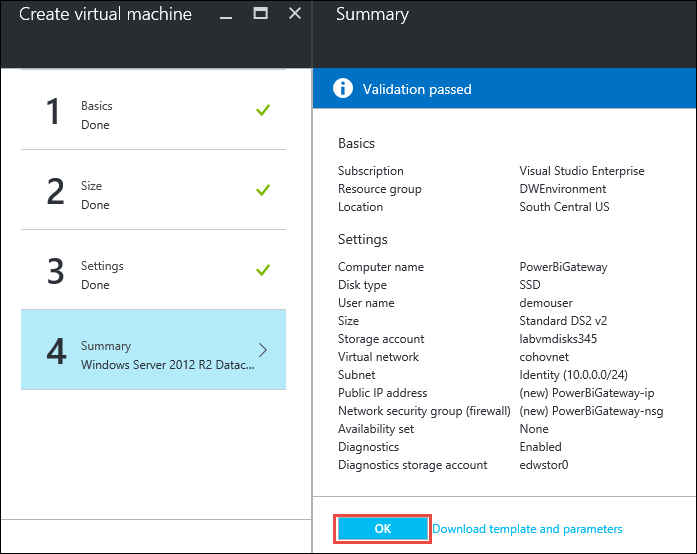
Note: If the DS2\_v2 size is not visible you may need to select View All to see it.

1. On the Create virtual machine Settings blade, configure the following settings, take the defaults for any settings not defined here, then click **OK**:

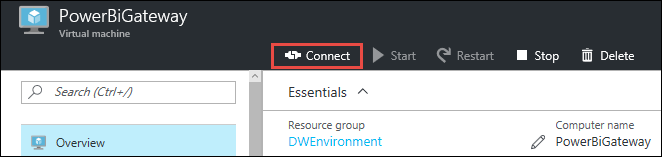
* Virtual network: cohovnet
* Subnet: Identity



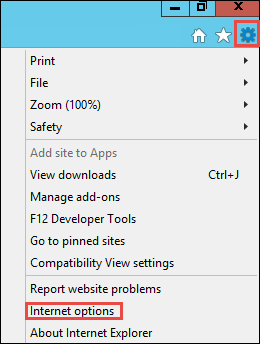
1. On the Create virtual machine Summary blade, click **OK** to begin the deployment of your Power BI Gateway machine.



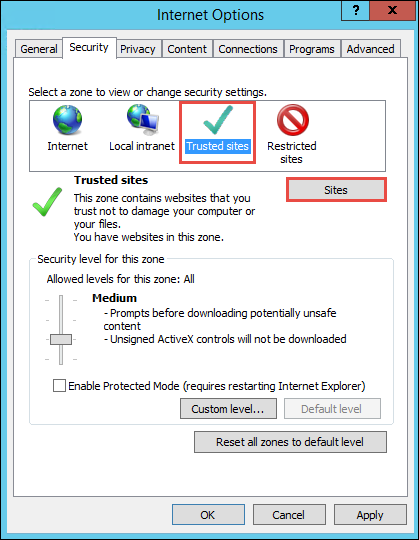
1. Wait for the virtual machine to deploy before proceeding.
2. Once the virtual machine deployment has succeeded, connect to **PowerBiGateway** using your **demouser** admin account.



1. Once logged into **PowerBiGateway**, open Internet Explorer, click **Settings**, then choose **Internet Options**.



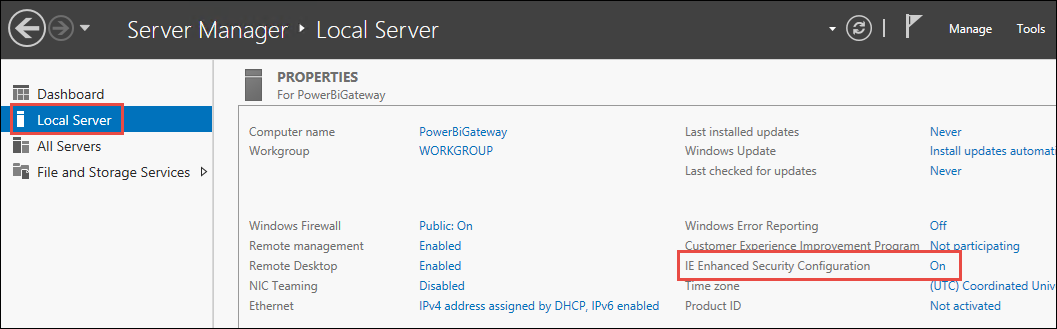
1. Click the Security tab, click on **Trusted sites**, and then click the **Sites** button.



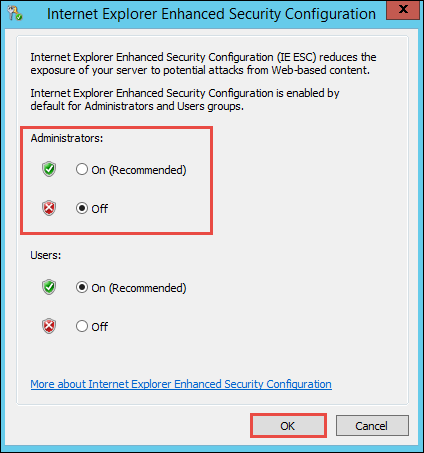
1. On the Trusted sites window, uncheck the **Require server verification (https:) for all sites in this zone** checkbox, then add **microsoft.com** to the trusted site zone, then click **Close.**



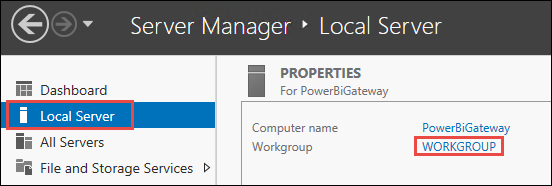
1. Click **OK**, then close Internet Explorer.
2. Navigate to Server Manager, click on **Local Server**, then click on **IE Enhanced Security Configuration**.



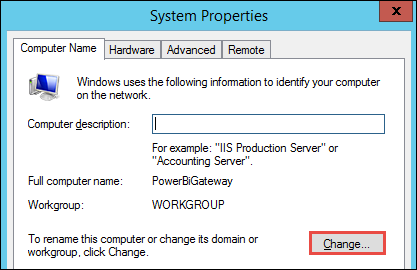
1. Set IE Enhanced Security Configuration to **Off** for Administrators, then click **OK**.



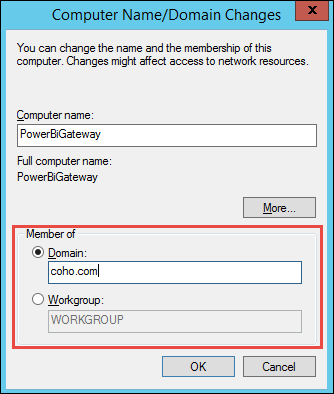
1. From within Server Manager, navigate to **Local Server**, then click **WORKGROUP**.



1. On the Computer Name tab, click the **Change** button.



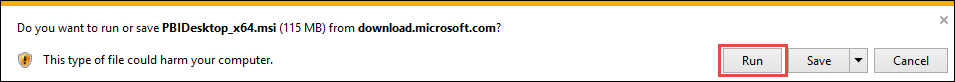
1. Make this virtual machine a member of the **coho.com** domain then click **OK**.



1. Use **demouser** and **demo@pass123** for the domain administrator account, then click **OK**, **Close**, **Restart** **Now** to reboot the computer.
2. After the computer has rebooted, connect to **PowerBiGateway** using your **demouser** admin account.
3. Open Internet Explorer, navigate to <https://powerbi.microsoft.com/en-us/desktop/> and download the Power BI Desktop by clicking the **Download** button.



1. Click Run on the download popup.



1. Click **Next** on the Welcome screen.
2. Accept the license terms and click **Next**.
3. Accept the default destination folder and click **Next**
4. Click **Install** to install Power BI Desktop. Then click **Finish** to complete the install.
5. Open Internet Explorer, navigate to <https://powerbi.microsoft.com/en-us/gateway/> and download the Power BI Gateway by clicking the **Download gateway** button.



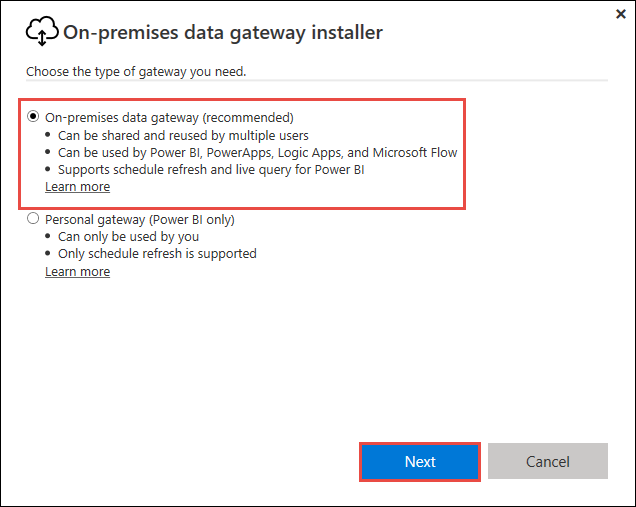
1. Click Run on the download popup.



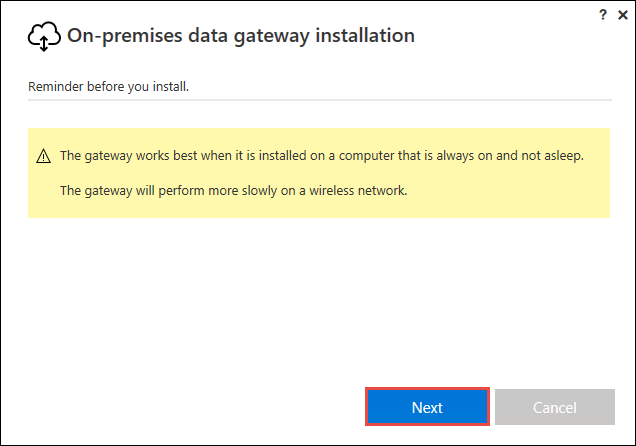
1. Click **Next** on the **Start your on-premises data gateway installation** screen.



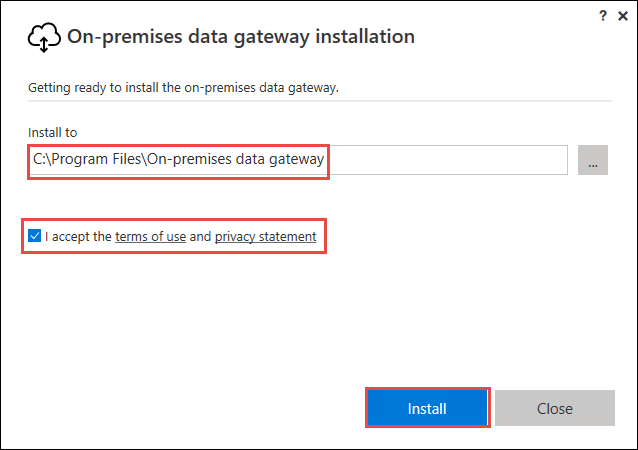
1. Choose **On-premises data gateway** type, then click **Next**.



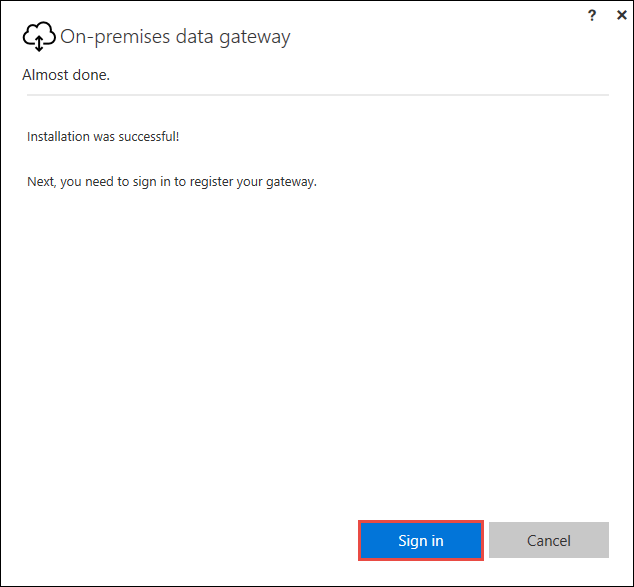
1. Click **Next** on the **Reminder before you install** screen.



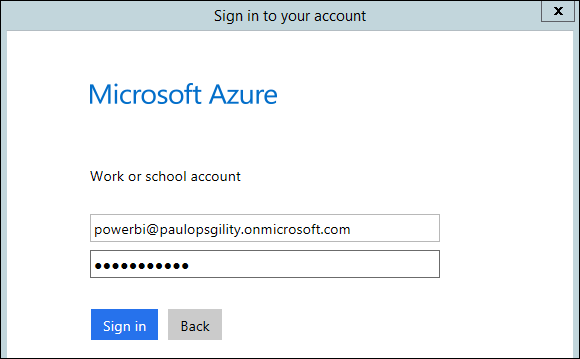
1. Accept the default **install to** location, check the **I accept the terms of use** checkbox, then click **Install**.



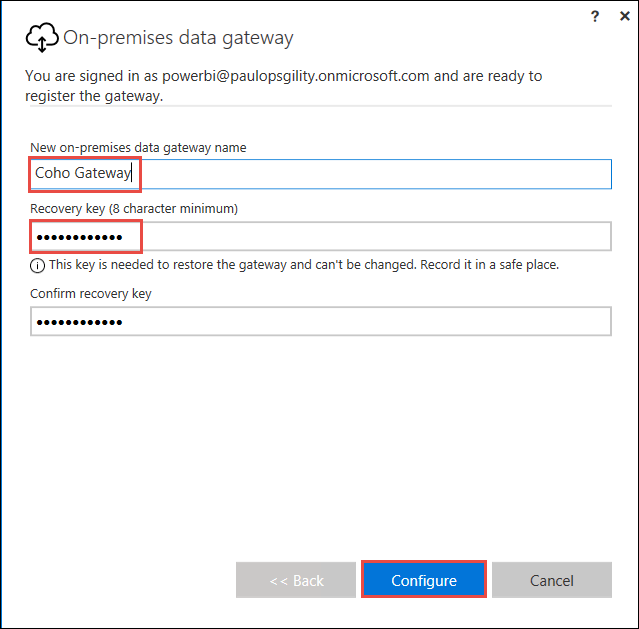
1. Click **Sign in** after successful installation.



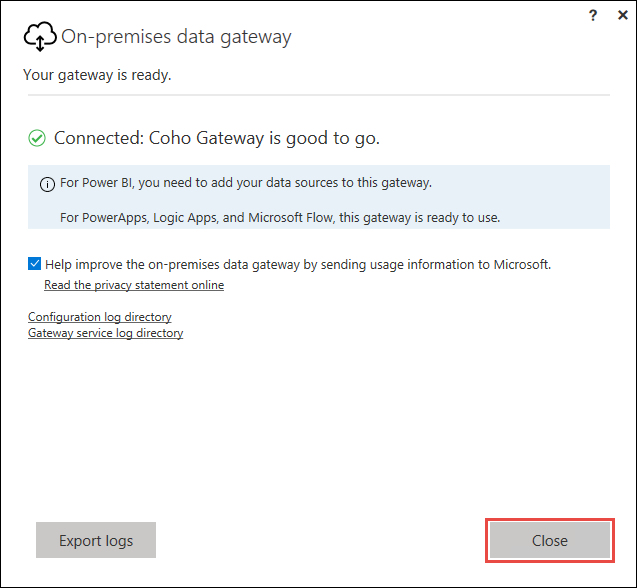
1. Sign in with your Power BI account.



1. Choose a name your gateway, in this lab example we use **Coho Gateway**, use **demo@pass123** for your recovery key, and then click Configure.



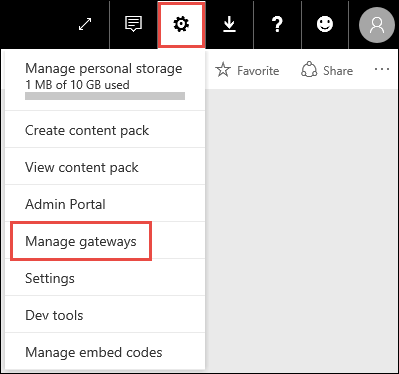
1. After your gateway is configured click **Close**.



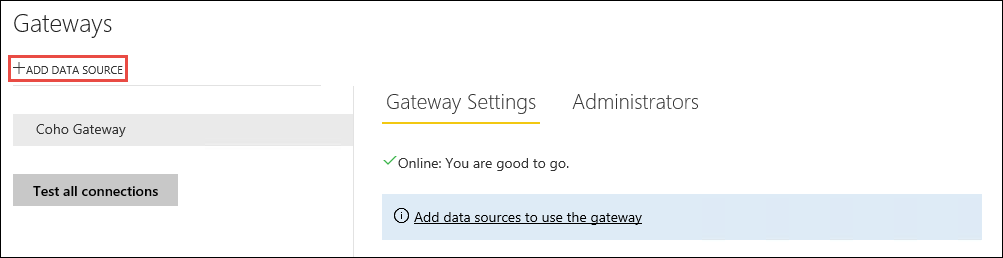
1. Open Internet Explorer and navigate to **powerbi.com** and click **Sign in** in the upper right corner of the screen.



1. Sign in with the organizational account that you are using to access Power Bi Pro.
2. Click the settings icon in the upper right corner then choose **Manage gateways** from the menu.

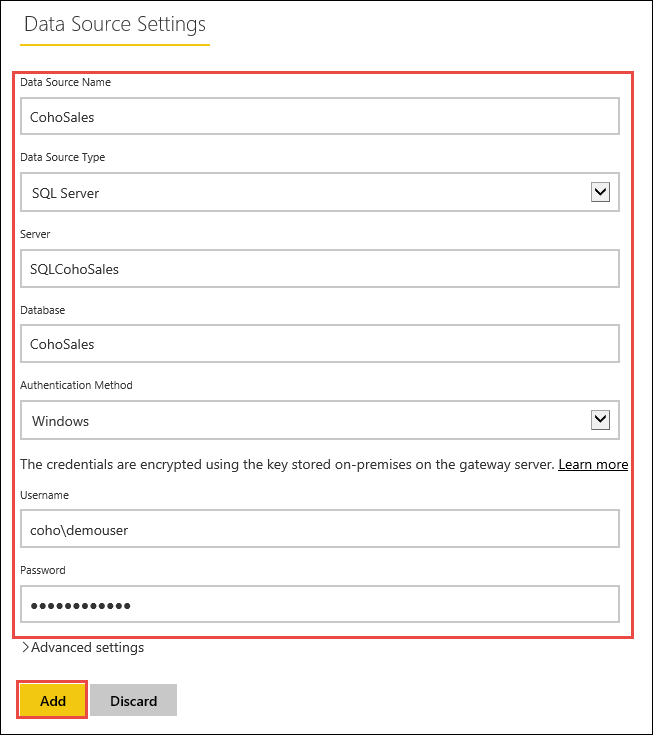


1. Click on **+ADD A DATA SOURCE**.

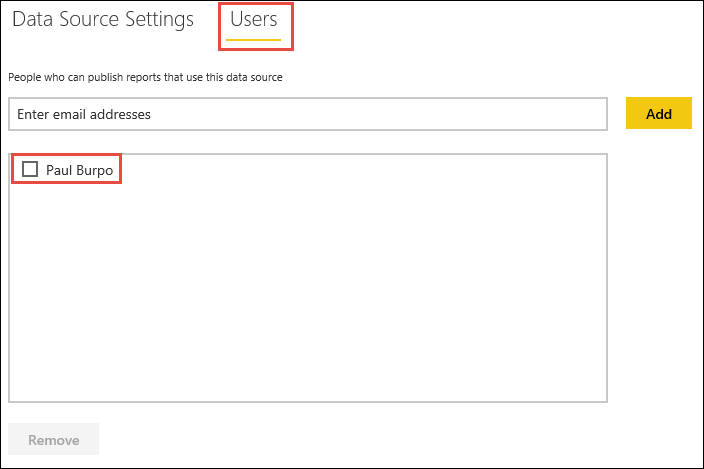


1. Enter the following **Data Source Settings** then click **Add**:

* Data Source Name: **CohoSales**
* Data Source Type: **SQL Server**
* Server: **SQLCohoSales**
* Database: **CohoSales**
* Authentication Method: **Windows**
* Username: **COHO\demouser**
* Password: **demo@pass123**



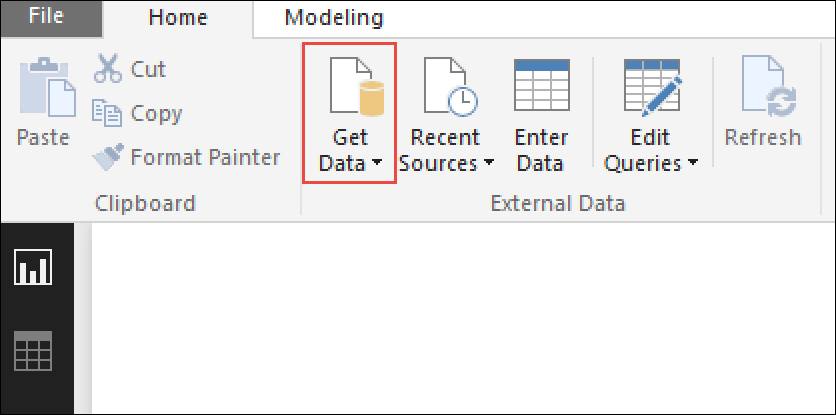
1. Click on the **Users** tab, your Power BI account should automatically be added to the list of users that can publish reports using this data source.



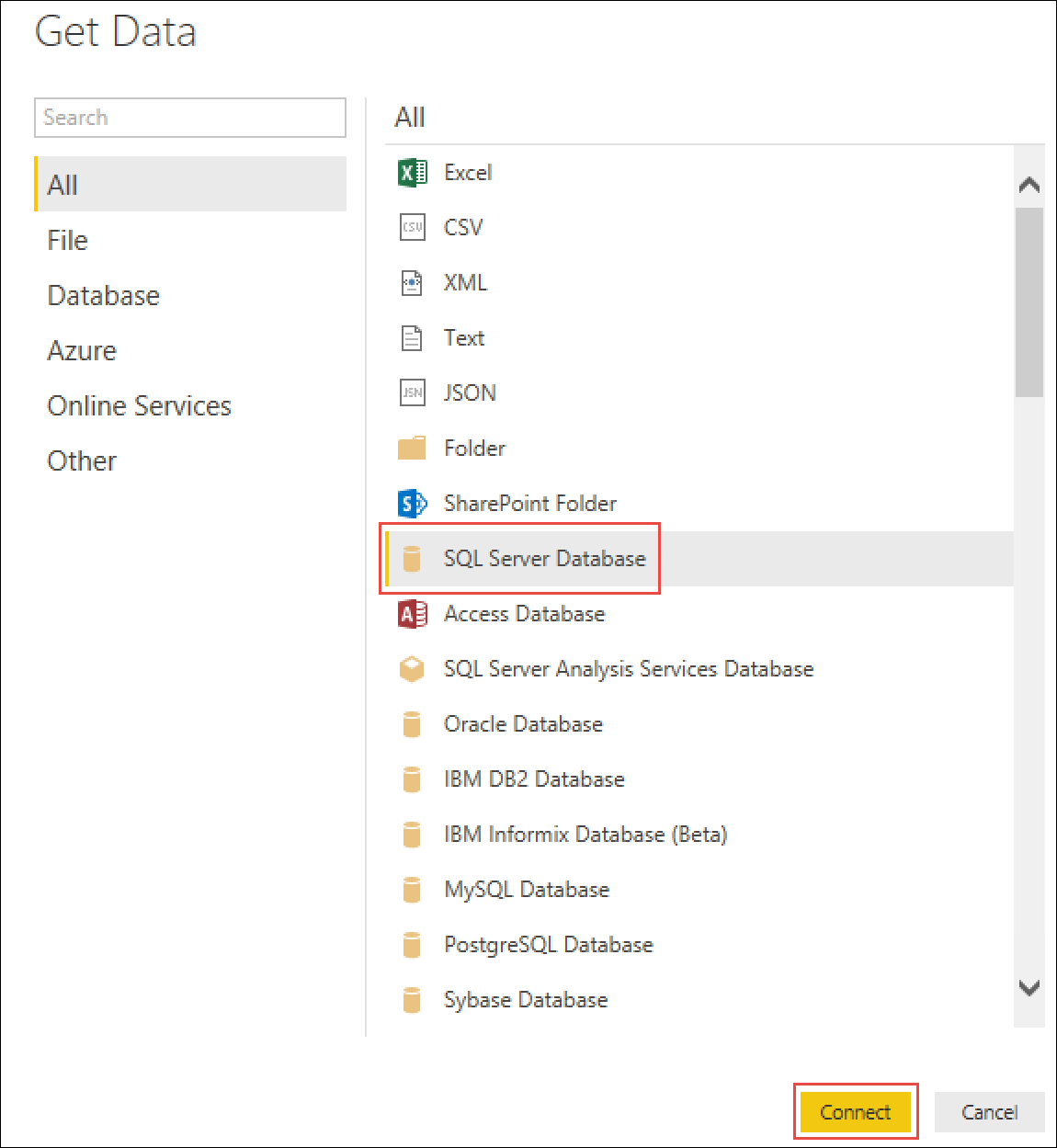
1. Go back to your PowerBiGateway virtual machine and login with the COHO\demouser account.
2. Launch Power BI Desktop and click Sign in.



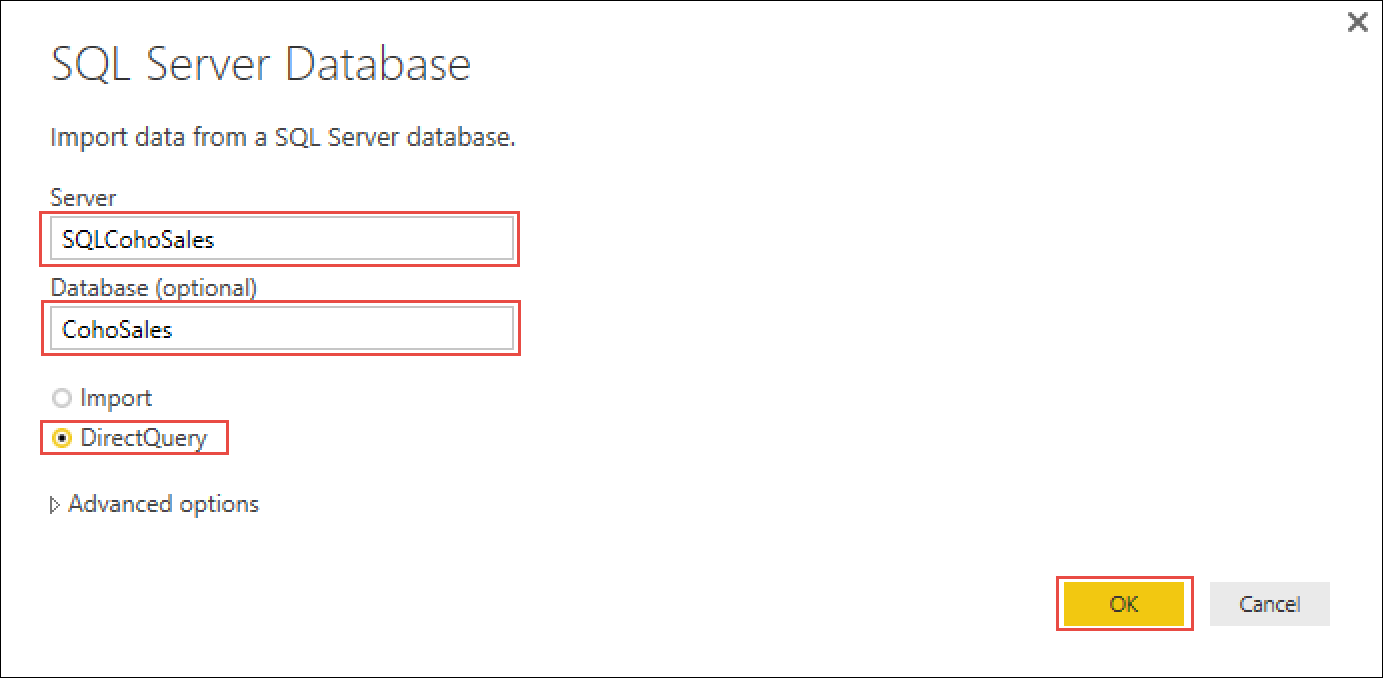
1. Sign in with the account you use with your Power BI Pro subscription.
2. In Power BI Desktop, click on the **Get Data** button.



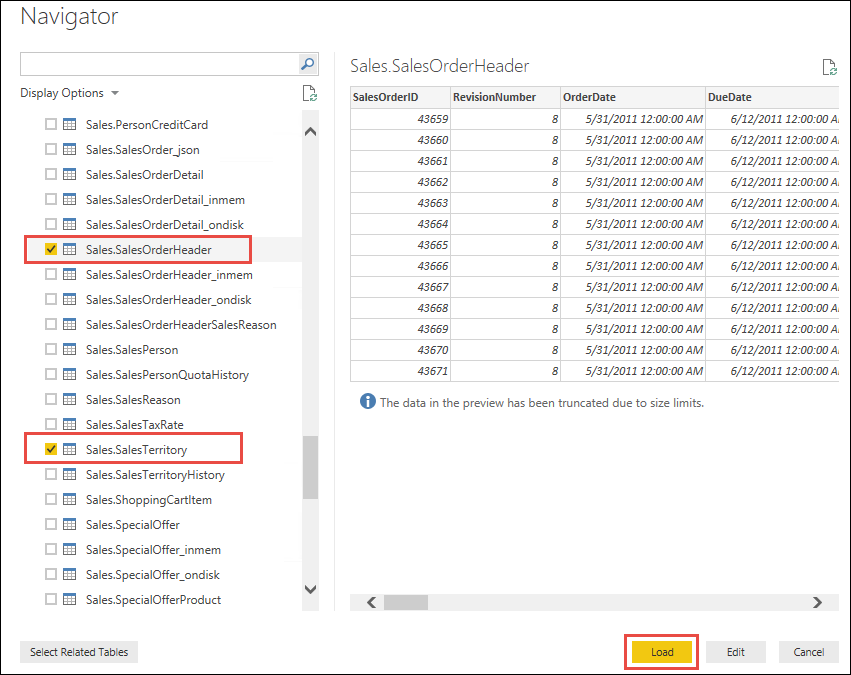
1. From the Get Data windows choose **SQL Server Database** and then click **Connect**.



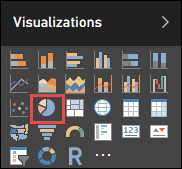
1. On the SQL Server Database window type **SQLCohoSales** for the Server and **CohoSales** for the Database, select the **DirectQuery** radio button, then click **OK**.



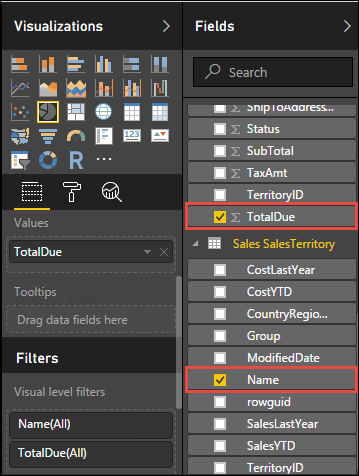
1. On the Navigator screen select the **Sales.SalesOrderHeader** and **Sales.SalesTerritory** tables and then click **Load**.



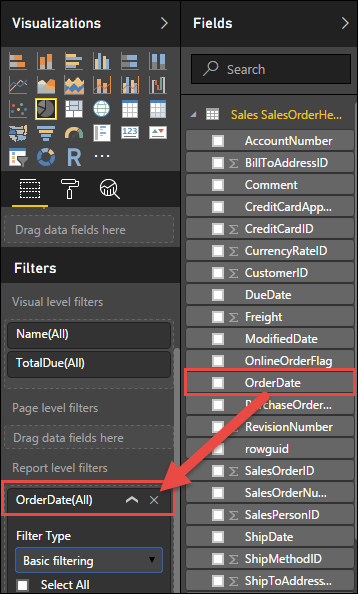
1. On the visualizations blade click the **pie-chart** icon.



1. On the Fields blade, expand the Sales.SalesOrderHeader table and put a check next to **∑ TotalDue**, expand Sales.SalesTerritory and put a check next to **Name**.

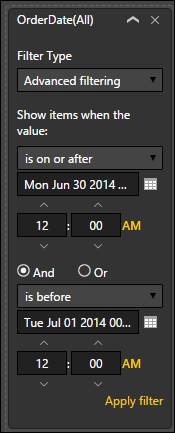


1. Drag the **OrderDate** field from the Sales.SalesOrderHeader table to the Report level filters.



1. Change the Filter Type to **Advanced filtering**, set the following values and then click **Apply filter**:

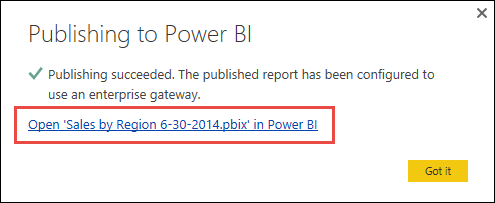
* Show items when the value: **is on or after**
* Date: **2014-06-30**
* Time: **12:00AM**
* And: **is before**
* Date: **2014-07-01**
* Time: **12:00AM**



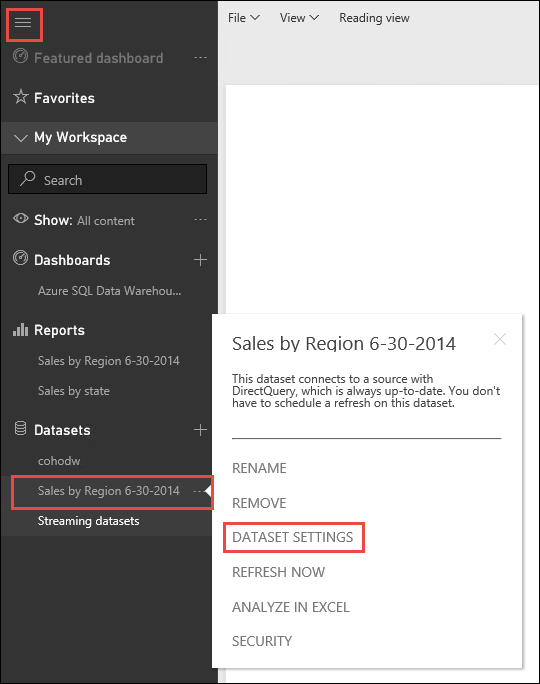
1. Click **save** and save your report to the local machine with the name **Sales by Region 6-30-2014.pbix**
2. Click the Publish button.



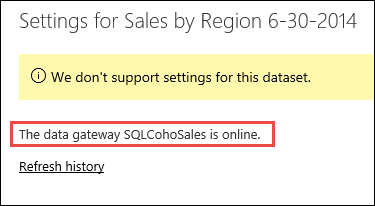
1. After successful publication, click the **Open ‘Sales by Region 6-30-2014.pbix’ in Power BI** link.



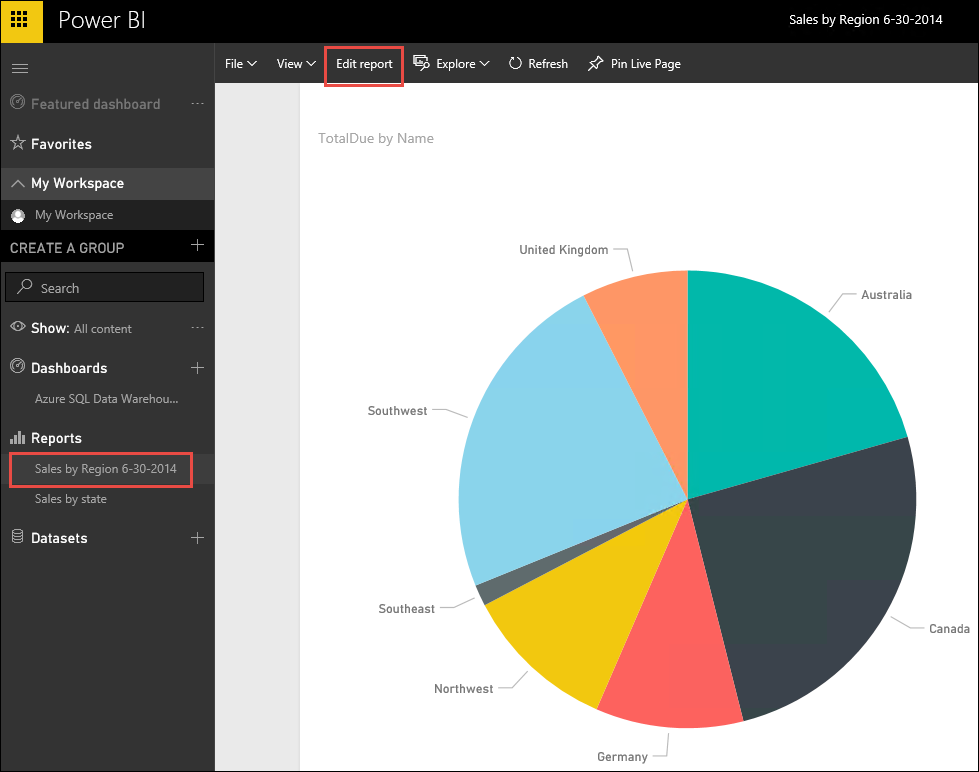
1. Your report will open on the Power BI site. On the Power BI site expand the **menu**, then click on the ellipses next to the **Sales by Region 6-30-2014** dataset and choose Dataset Settings.



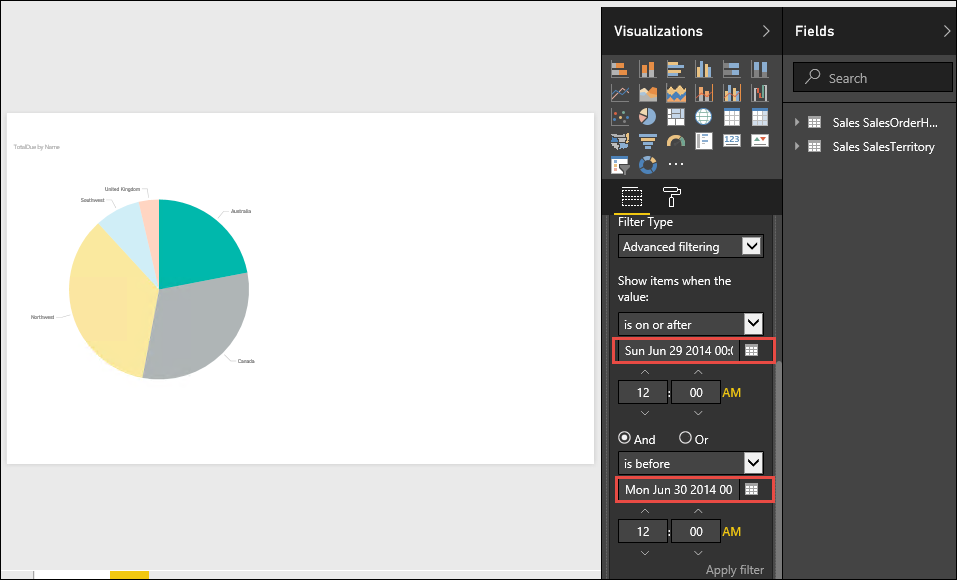
1. Notice that the dataset is using the data gateway that we deployed previously.



1. In the menu, click on the **Sales by Region 6-30-2014** report. Then click **Edit report** at the top of the report.



1. Click on the **OrderDate** filter. And modify both of the date filters to be one day earlier then click **Apply filter**. Notice that the pie chart changes as we modify the report.



### Summary

In this exercise, you have configured Power BI to connect directly to an Azure SQL Data Warehouse. You then deployed an Enterprise Data Gateway to enable access to on-premises data sources. Finally, you used Power BI Desktop to publish a report that leverages the Enterprise Gateway to connect to an on-premises data source.