



Microsoft Cloud Workshop

Migrate EDW to Azure SQL Data Warehouse

Hands-on lab step-by-step

February 2018

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.

© 2018 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

Migrate EDW to Azure SQL Data Warehouse hands-on lab step-by-step	1
Abstract and learning objectives	1
Overview	1
Solution architecture.....	2
Requirements	2
Before the hands-on lab	3
Task 1: Deploy the source environment.....	3
Task 2: Configure the SQL Server	4
Exercise 1: Configure Azure Services.....	6
Task 1: Create an Azure Storage Account.....	6
Task 2: Create an Azure SQL Data Warehouse.....	8
Task 3: Create Analysis Services	9
Task 4: Pre-load data to SQL Data Warehouse	10
Exercise 2: Data and schema preparation.....	15
Task 1: Validate schema and data	15
Task 2: Prepare Azure SQL Data Warehouse and migrate schema	17
Exercise 3: Migrate the data to Azure SQL Data Warehouse.....	20
Task 1: Exporting the data from your current data warehouse	20
Task 2: Transfer your data to Azure	20
Exercise 4: Investigate table distribution in Azure SQL Data Warehouse	23
Task 1: Create test tables.....	23
Task 2: Create test tables.....	24
Exercise 5: Create an Analysis Services Model	28
Task 1: Configure Analysis Services backup	28
Task 2: Restore Analysis Services backup.....	30
Task 3: Update Analysis Services connections.....	32
Exercise 6: Visualize data with Power BI Desktop.....	34
Task 1: Install Power BI Desktop	34
Task 2: Query data with Power BI Desktop.....	37
After the hands-on Lab	41
Task 1: Cleanup resource groups.....	41

Migrate EDW to Azure SQL Data Warehouse

hands-on lab step-by-step

Abstract and learning objectives

This whiteboard design session will look at the process of migrating an on-premises data warehouse to Azure SQL Data Warehouse. The design session will cover data and schema preparation, data loading, optimizing the data distribution, and building a solution to support ad-hoc queries.

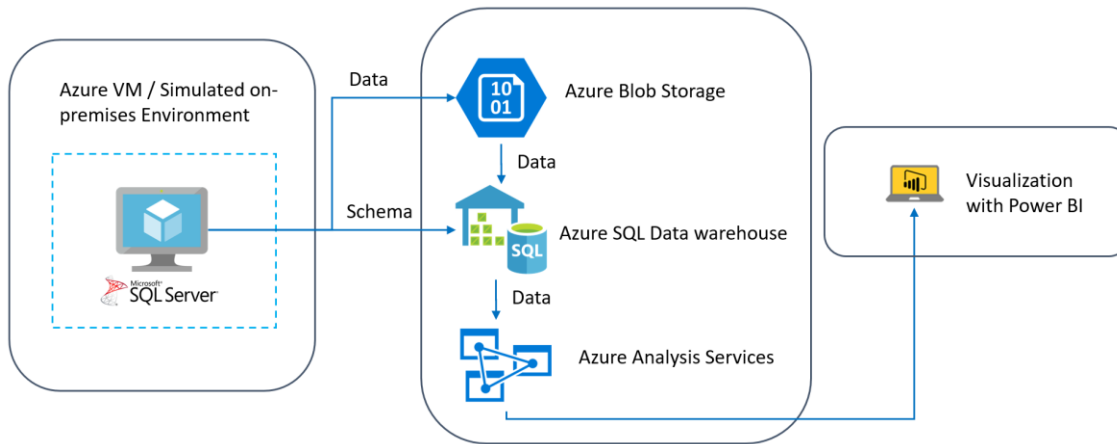
Attendees will learn how to plan a data warehouse migration as well as:

- How to prepare and migrate data warehouse schema and data
- How to configure a BI solution in Azure

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. You will analyze different table distribution methods and their performance impact on performance of the warehouse. You will then configure an Azure Analysis Services data model to allow ad-hoc access to the data via Power BI.

Solution architecture



Requirements

1. Microsoft Azure subscription

Before the hands-on lab

In this exercise, you will deploy the source environment for this lab. The source environment is designed to represent the existing on-premises environment you will migrate to Azure SQL Data Warehouse.

Task 1: Deploy the source environment

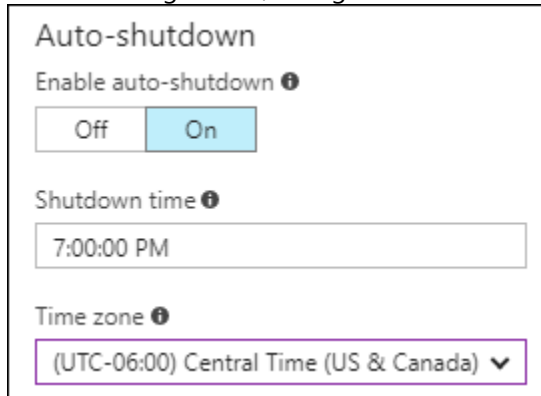
1. Browse to the Azure Portal at <https://portal.azure.com>
2. Click **+Create a resource**, and type **SQL Server 2016** in the search box. Choose the latest service pack version of **Free License: SQL Server 2016 SP1 Developer on Windows Server 2016** from the search results.

NAME	PUBLISHER
SQL Server 2016 SP1 Standard on Windows Server 2016	Microsoft
Free License: SQL Server 2016 SP1 Developer on Windows Server 2016	Microsoft
Free License: SQL Server 2016 SP1 Express on Windows Server 2016	Microsoft

3. Click the **Create** button to begin deployment of the SQL Server
4. On the Basics blade, use the following configurations:
 - Name: **SQLcohoDW**
 - VM disk type: **SSD**
 - User name: **demouser**
 - Password: **Demo@pass123**
 - Subscription: **Your subscription**
 - Resource Group: **Create new → OnPremisesEnvironment**
 - Location: **Choose a location near you**
5. On the size blade, choose **DS2_V2**. If you do not see this option, click **View all**

DS1_V2	DS2_V2	DS3_V2
Standard	Standard	Standard
1 vCPU	2 vCPUs	4 vCPUs
3.5 GB	7 GB	14 GB
2 Data disks	4 Data disks	8 Data disks
3200 Max IOPS	6400 Max IOPS	12800 Max IOPS
7 GB Local SSD	14 GB Local SSD	28 GB Local SSD
Premium disk support	Premium disk support	Premium disk support
Load balancing	Load balancing	Load balancing

6. On the Settings blade, change the Auto-shutdown time zone to reflect your current time zone. Then, click **OK**.



Auto-shutdown

Enable auto-shutdown ⓘ

Off On

Shutdown time ⓘ

7:00:00 PM

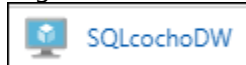
Time zone ⓘ

(UTC-06:00) Central Time (US & Canada) ▼

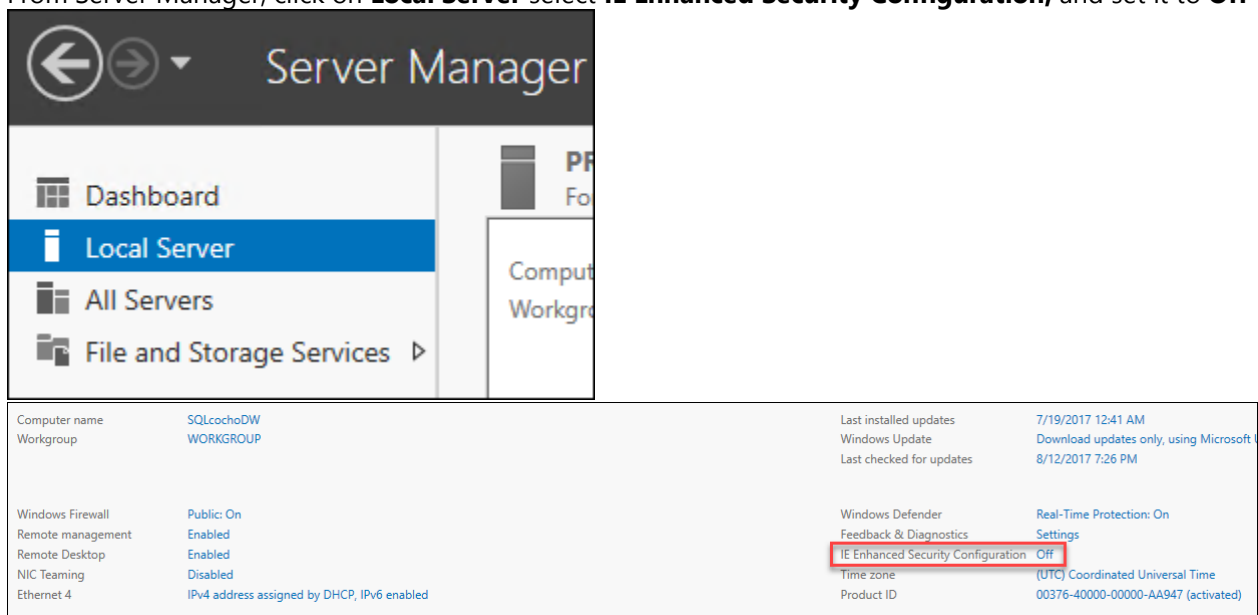
7. On the SQL Server settings blade, accept the defaults, and click **OK**.
8. On the Summary blade, review your settings. Then, click **Create**.

Task 2: Configure the SQL Server

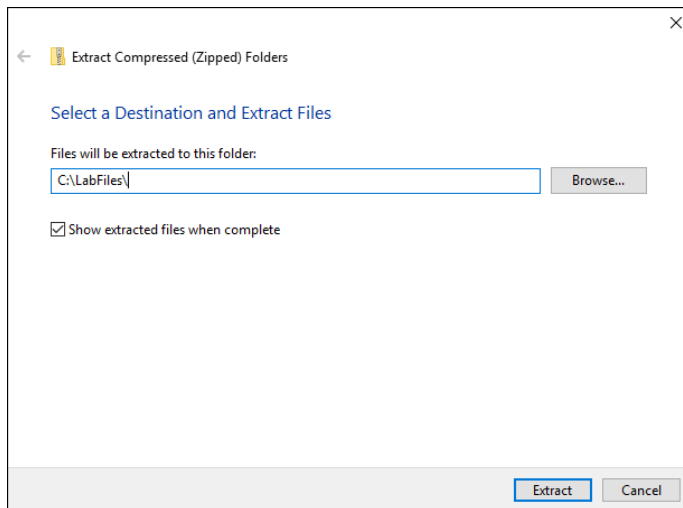
1. Login to the **SQLcochoDW** virtual machine you created in the previous task



2. From Server Manager, click on **Local Server** select **IE Enhanced Security Configuration**, and set it to **Off**



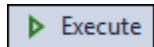
4. Install the Azure PowerShell command-line tools from:
<https://www.microsoft.com/web/handlers/webpi.ashx/getinstaller/WindowsAzurePowerShellGet.3f.3f.3fnew.appids>
5. Open **File Explorer**, and create two new folders called **C:\LabFiles**, **C:\Migration**
6. Download the Student Files from <http://cloudworkshop.blob.core.windows.net/migrate-edw/StudentFiles-11-2017.zip>, and extract the files to **C:\LabFiles**



7. Launch **SQL Server Management Studio**, and connect to the SQLcohoDW instance
8. Open a **New Query** window, and cut/paste the following code into the window

```
RESTORE DATABASE CohoDW FROM DISK = 'C:\LabFiles\CohoDW.bak'  
WITH MOVE 'CohoDW_Data' TO 'F:\Data\CohoDW_Data.mdf'  
      , MOVE 'CohoDW_Log' TO 'F:\Log\CohoDW_Log.ldf'
```

9. Click **Execute** to restore the database



Exercise 1: Configure Azure Services

In this exercise, you will create and configure an Azure Storage Account, Azure SQL Data Warehouse, and Azure Analysis Services. You will also begin pre-loading a large dataset into Azure SQL Data Warehouse which will be used later to evaluate the performance of different table distributions.

Task 1: Create an Azure Storage Account

1. Browse to the Azure Portal and authenticate at <https://portal.azure.com/>
2. Click **+Create a resource** 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: **Storage (general purpose v1)**
 - Performance: **Standard**
 - Replication: **Locally-redundant storage (LRS)**
 - Secure transfer required: **Disabled**
 - Resource group: **Create new - EDWmigrationStor**

- Location: **Location near you**

Create storage account

The cost of your storage account depends on the usage and the options you choose below.
[Learn more](#)

* Name

Deployment model ☒ Resource manager ☐ Classic

Account kind

Performance ☒ Standard ☐ Premium

Replication

* Secure transfer required ☒ Disabled ☐ Enabled

* Subscription

* Resource group ☒ Create new ☐ Use existing

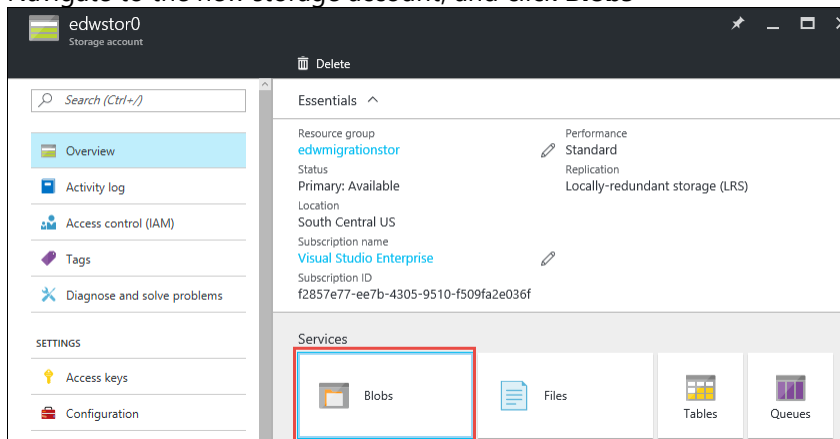
* Location

Virtual networks (Preview)
Configure virtual networks ☐ Disabled ☐ Enabled

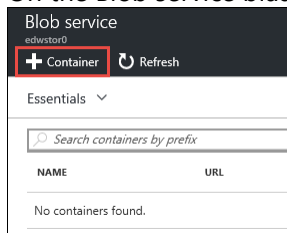
☐ Pin to dashboard

Create [Automation options](#)

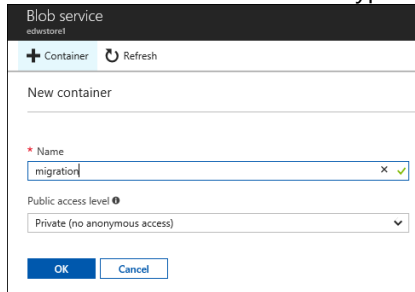
4. Navigate to the new storage account, and click **Blobs**



5. On the Blob service blade, click the **+Container** button



6. On the New container blade type **migration** for the name. Then, click **OK**



Blob service
edwstore1

+ Container Refresh

New container

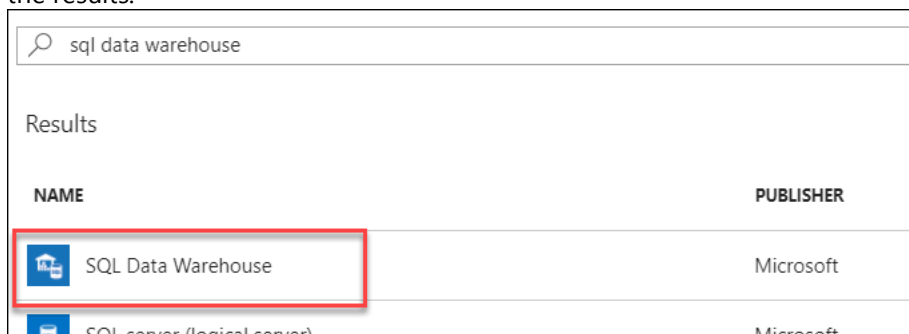
* Name
migration x ✓

Public access level ⓘ
Private (no anonymous access) v

OK Cancel

Task 2: Create an Azure SQL Data Warehouse

1. Click **+Create a resource** and type **SQL Data Warehouse** in the search box. Choose **SQL Data Warehouse** from the results.



sql data warehouse

Results

NAME	PUBLISHER
SQL Data Warehouse	Microsoft
SQL server (logical server)	Microsoft

2. Click **Create** on the SQL Data Warehouse blade. Specify the following information. Then, click the **Server** tile.
- Name: **CohoDW**
 - Resource group: **Create new - CohoDWRG**
 - Select source: **Blank database**
3. 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**

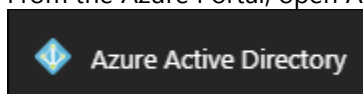
4. Select the Performance tier tile, set the performance to **400 DWU**, and click **Apply**

5. On the SQL Data Warehouse blade, click **Create**

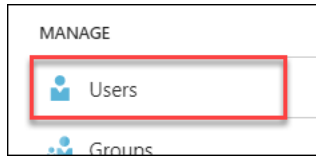
Task 3: Create Analysis Services

The first four steps of this task walk you through creating an organizational account to use as the administrator account for Analysis Services. This account must be an organizational account, it cannot be a Microsoft live account. If you are doing this lab with an existing organizational account you may skip the first four steps of this task and use your organizational account in place of the asadmin account.

1. From the Azure Portal, open Azure Active Directory



- Click **Users** from the menu



- Click the **+New user** button, and create a user using the following configuration

- Name: **asadmin**
- User name: **asadmin@<your-domain>.com**
- Password: Check the **Show password** box, then **copy** the password into Notepad.

The User name setting should be in the form <name>@<your-domain>.com. If you do not know your domain name you can get it by hovering over your login information in the upper right corner of your browser window.

- Click the **Create** button
- Click **+Create a resource** and type **Analysis Services** in the search box. Choose **Analysis Services** from the results

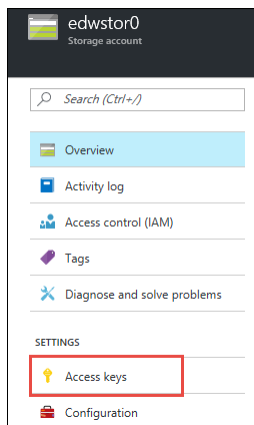


- Click **Create** on the Analysis Services information blade
- Use the following configurations then click **Create**
 - Server name: **Choose a unique name**
 - Subscription: **Choose your subscription**
 - Resource group: **Use existing** → **CohoDWRG**
 - Location: **Choose the same location as your other resources**
 - Pricing tier: **S0 Standard**
 - Administrator: **ASadmin**
 - Backup Storage Settings: **Not configured**
 - Storage key expiration: **Never**

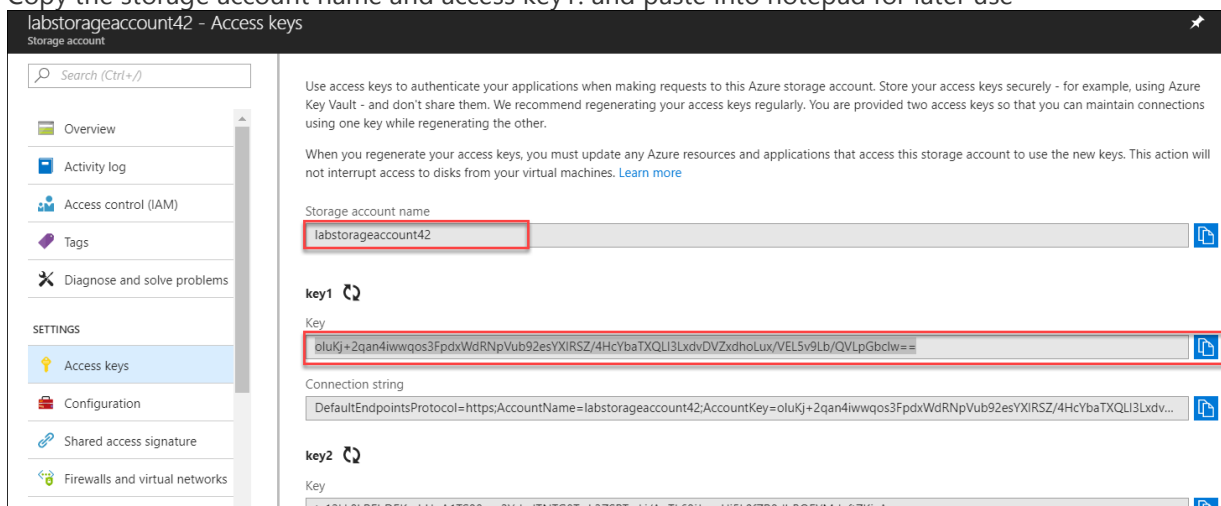
Task 4: Pre-load data to SQL Data Warehouse

- In the Azure Portal navigate to your **EDWmigrationStor** resource group, and click on your storage account

- In the Storage account blade, and under settings, click on **Access keys**



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



- Launch File Explorer and navigate to the **C:\LabFiles** folder
- Right-click the **GenBigTable.ps1** PowerShell script, and choose **Edit** to open the file in **Windows PowerShell ISE**
- In the command window, type **Login-AzureRmAccount**, and hit **Enter**. Then, login with the same credentials you are using to login to the Azure Portal.

```
PS C:\LabFiles> Login-AzureRmAccount

Account           : paul@opsgility.com
SubscriptionName  : Visual Studio Enterprise
SubscriptionId    : 
TenantId         : 
Environment      : AzureCloud

PS C:\LabFiles>
```

- Type **Get-AzureRmSubscription** in the command window. If you see multiple subscriptions, you must set your subscription context by running **Set-AzureRmContext -SubscriptionId '<subscription id you are using for this lab>'**

```
PS C:\LabFiles> Set-AzureRmContext -SubscriptionId '...'

Name           : [paul@opsgility.com, ...]
Account        : paul@opsgility.com
SubscriptionName : Visual Studio Enterprise
SubscriptionId  : 
TenantId       : 
Environment    : AzureCloud

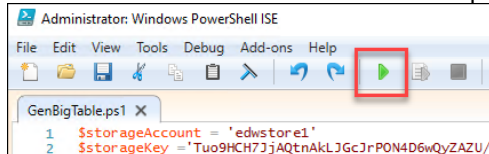
PS C:\LabFiles>
```

8. In the script window, update the **\$storageAccount** and the **\$storageKey** variables with the storage account information that you copied earlier

```

1 $storageAccount = 'edwstore1'
2 $storageKey = 'Tuo9HCH7JjAQtnAkLJGcJrPON4D6wQyZAZU/spMFhb/F3a0rvc40WtZ2/RdbRGKkarQ/7RIM72FL4Ug1S5htgA=='
3
4
5 ### Create the storage context for authenticating the copy
6 $storageContext = New-AzureStorageContext -StorageAccountName $storageAccount -StorageAccountKey $storageKey
7
8 $srcUri = "https://edwstore1.blob.core.windows.net/bigtable/dbo.FactInternetSales.txt"
9 $blobName = "dbo.FactInternetSales.txt"
  
```

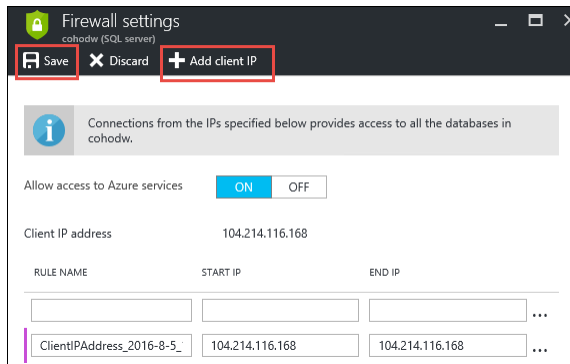
9. Click the **Execute** button to run the script



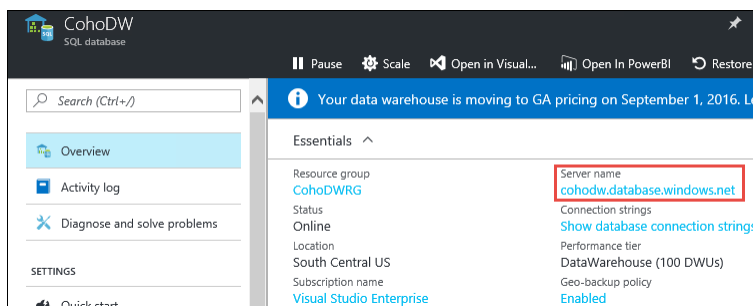
10. The script will take about 10 minutes to run. You may continue with the lab while the script is running but you will not be able to run the table load script until the GenBigTable.ps1 PowerShell script is complete.
11. From the **SQLCohoDW** virtual machine, open **Internet Explorer**, and connect to the **Azure Portal**
12. Navigate to your **CohoDWRG** resource group. Then, click on the cohodw logical SQL Server that hosts your Azure SQL Data Warehouse.

NAME	TYPE	LOCATION	
cohodw	SQL server	South Central US	...
CohoDW	SQL data ware...	South Central US	...

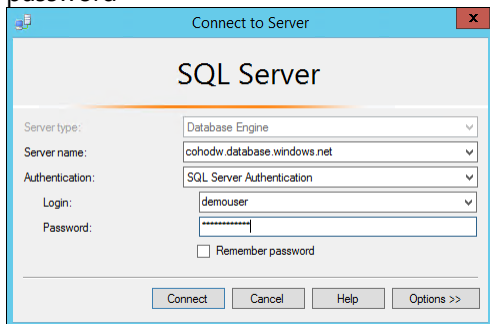
13. In the settings blade, click on **Firewall / Virtual Networks**
14. In the cohodw - Firewall blade, click the **+Add client IP** button. Then, click the **Save** button.



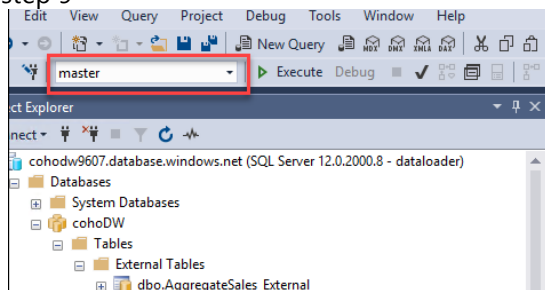
15. Back in the **CohoDWRG** resource group, select the **CohoDW** data warehouse, and copy the server name



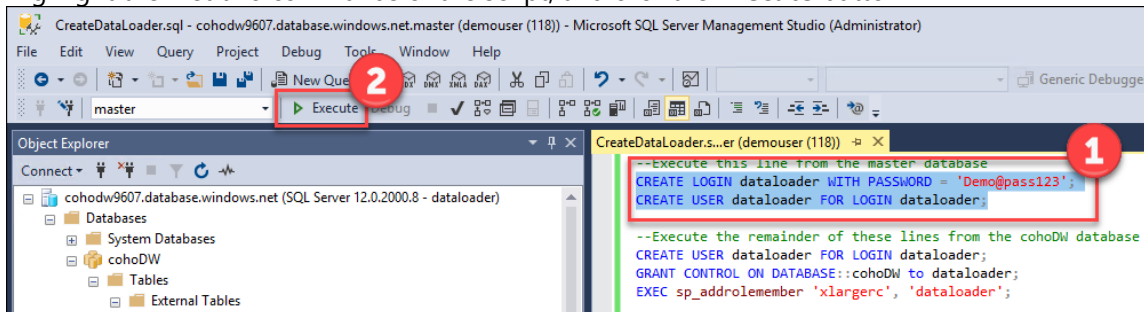
16. From File Explorer, open the **C:\LabFiles\CreateDataLoader.sql** script in SQL Server Management Studio
17. Connect to your Azure SQL Data Warehouse using SQL Server Authentication and the **demouser** account and password



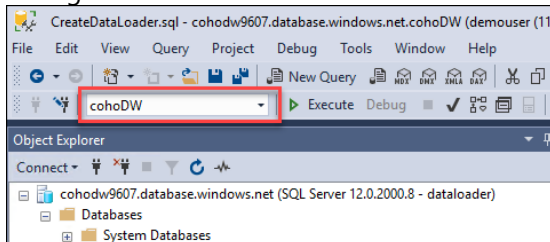
18. Verify you are connected to the **master** database. Do not continue if until your PowerShell script has finished from step 9



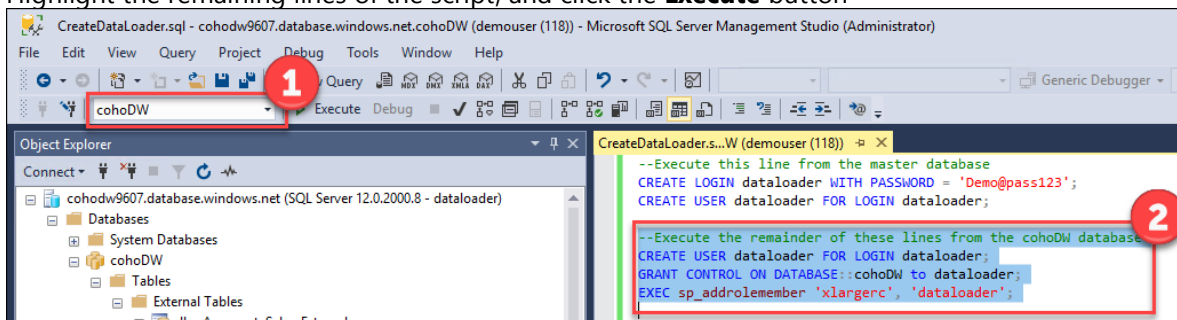
19. Highlight the first two commands of the script, and click the **Execute** button



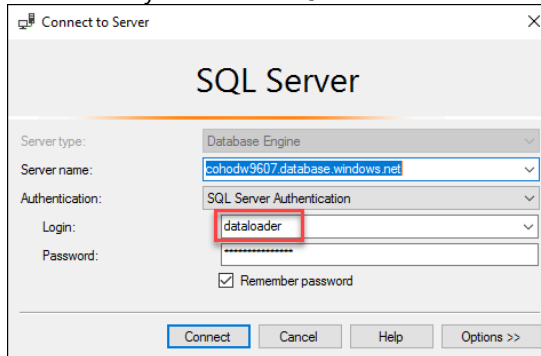
20. Change the database context to **cohoDW**.



21. Highlight the remaining lines of the script, and click the **Execute** button



22. **Close** SQL Server Management Studio
23. From File Explorer, open the **C:\LabFiles\PreLoadBigTable.sql** script in SQL Server Management Studio
24. Connect to your Azure SQL Data Warehouse with the **dataloader** username and password you just created



25. Edit the script by replacing occurrences of **<YourStorageAccountName>** and **<YourStorageAccountKey>** along with the Storage account name and key you copied earlier. There are three in total. Be sure to change all three or you run the risk of having to recreate the data warehouse.

```

CREATE MASTER KEY

CREATE DATABASE SCOPED CREDENTIAL MigrationCredential
WITH IDENTITY = '<YourStorageAccountName>' SECRET = '<YourStorageAccountKey>'

CREATE EXTERNAL DATA SOURCE Migrationstor WITH (TYPE = HADOOP,
LOCATION='wasbs://bigtable@<YourStorageAccountName>.blob.core.windows.net',
CREDENTIAL = MigrationCredential);

CREATE EXTERNAL FILE FORMAT MigrationFiles WITH (FORMAT_TYPE = DelimitedText,
FORMAT_OPTIONS (FIELD_TERMINATOR = '|'));

CREATE EXTERNAL TABLE [dbo].[FactInternetSales_ROUNDROBIN_External]([ProductKey] int not null,[
PRINT N'Begin creating FactInternetSales_ROUNDROBIN ' + RTRIM(CAST(GETDATE() AS nvarchar(30)))

INSERT INTO [dbo].[FactInternetSales_ROUNDROBIN] SELECT [ProductKey],[OrderDateKey],[DueDateKey
PRINT N'Begin creating FactInternetSales_HASHED ' + RTRIM(CAST(GETDATE() AS nvarchar(30))) + N'

CREATE TABLE FactInternetSales_HASHED
WITH (DISTRIBUTION = HASH(ProductKey))
AS
    SELECT *
    FROM FactInternetSales_ROUNDROBIN;

PRINT N'Begin creating statistics on FactInternetSales_ROUNDROBIN ' + RTRIM(CAST(GETDATE() AS n

CREATE STATISTICS stat_FactInternetSales_ROUNDROBIN ON FactInternetSales_ROUNDROBIN (ProductKey

PRINT N'Begin creating statistics on FactInternetSales_HASHED ' + RTRIM(CAST(GETDATE() AS nvarc

CREATE STATISTICS stat_FactInternetSales_HASHED ON FactInternetSales_HASHED (ProductKey, OrderD

```

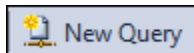
26. Verify you are connected to the **cohoDW** database, and click the **Execute** button to run the script
27. The script will take about 50 minutes to complete. **You may continue with the lab while the script is running.**

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, and make any necessary changes required for a successful migration.

Task 1: Validate schema and data

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

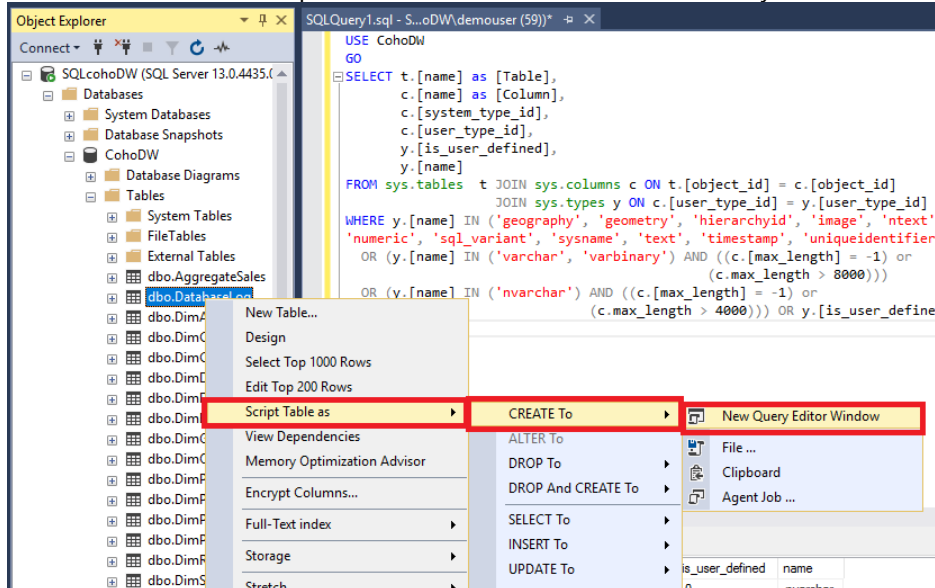


3. Run the following query to check for data incompatibility and potential data length issues

```
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/>

4. 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.



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

```
CREATE TABLE [dbo].[DatabaseLog](
    [DatabaseLogID] [int] NOT NULL,
    [PostTime] [datetime] NOT NULL,
    [DatabaseUser] [nvarchar](128) NOT NULL,
    [Event] [nvarchar](128) NOT NULL,
    [Schema] [nvarchar](128) NULL,
    [Object] [nvarchar](128) NULL,
    [TSQL] [nvarchar](4000) NOT NULL
) ON [PRIMARY]
```

6. 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.

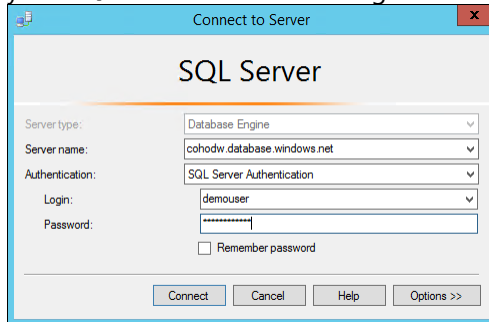
```
SELECT MAX(DATALENGTH([TSQL]))
FROM DatabaseLog
```

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

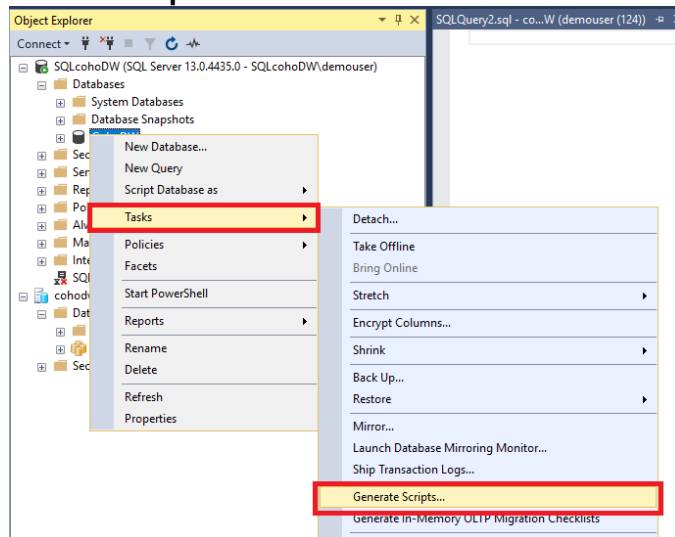
7. Modify the column by executing the following query:
- ```
ALTER TABLE dbo.DatabaseLog ALTER COLUMN [TSQL] nvarchar(2000)
```

## Task 2: Prepare Azure SQL Data Warehouse and migrate schema

1. In SQL Management Studio, click the **connect** button in Object Explorer, chose **Database Engine**, and connect to your SQL Data Warehouse using the **demouser** account and password to verify connectivity

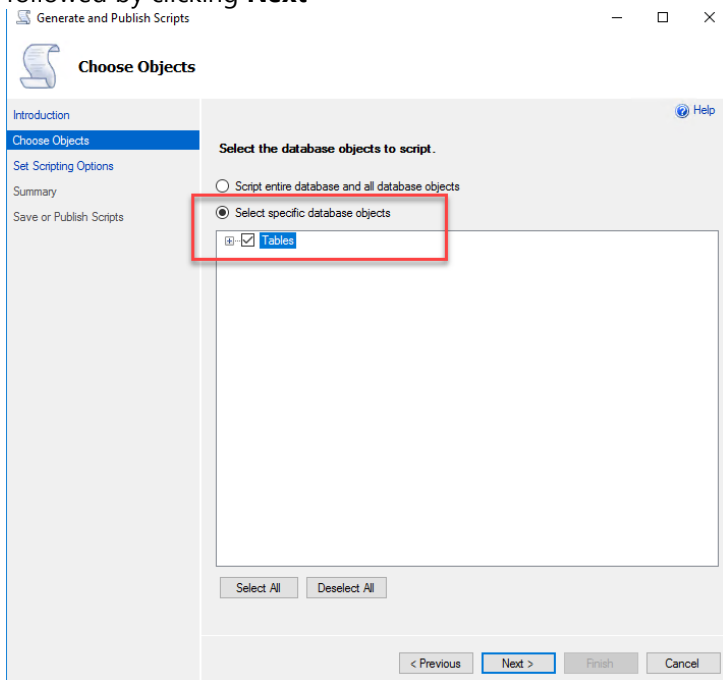


2. On your Azure SQL Data Warehouse, expand **databases**, select the **CohoDW** database followed by selecting the **New Query** button
3. Under your SQLCohoDW instance of SQL Server, right click your local copy of CohoDW, and select **Tasks -> Generate Scripts** to launch the Generate and Publish Scripts wizard

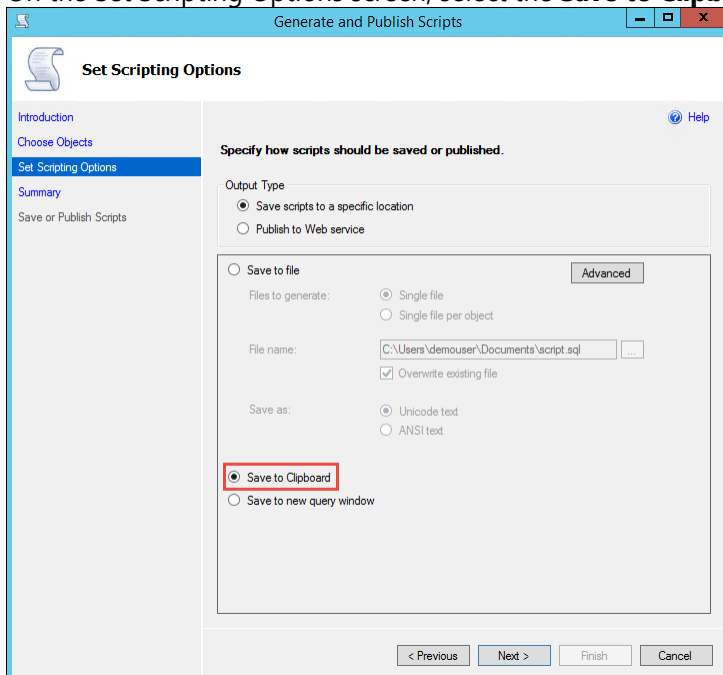


4. Click **Next** on the Introduction screen

5. On the Choose Objects screen, select the **Select specific database objects** radio button, and check **Tables** followed by clicking **Next**



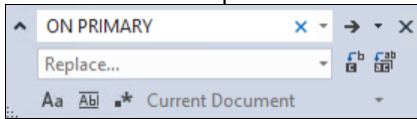
6. On the Set Scripting Options screen, select the **Save to Clipboard** radio button, and click **Next**



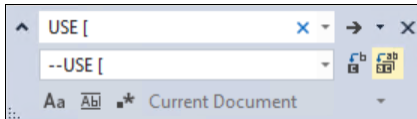
7. Accept the defaults for the remaining screens, and click **Finish**
8. Paste the results into the Query windows connected to your Azure SQL Data Warehouse

9. 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:

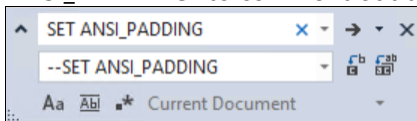
- Execute a Find and Replace on your script to replace all occurrences of "ON [PRIMARY]" with "" to remove them from the script



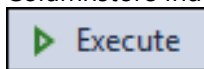
- Execute a Find and Replace on your script to replace all occurrences of "USE [" with "--USE [" to comment out those lines



- Execute a Find and Replace on your script to replace all occurrences of "SET ANSI\_PADDING" with "--SET ANSI\_PADDING" to comment out those lines



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



11. Execute the following query to verify that your tables were created. There should be 38 rows returned

```
SELECT * FROM sys.tables
```

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

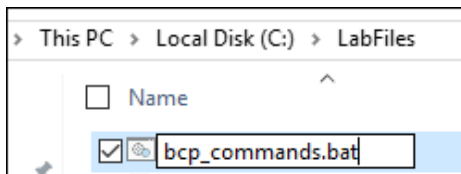
This exercise is focused on migrating the data from your existing data warehouse into SQL Data Warehouse. We will be pulling the data and uploading it to an Azure storage account. We will then import the data via Polybase.

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

1. Connect to your **SQLCohoDW** virtual machine.
2. Open the **C:\LabFiles\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 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 "C:\Migration\dbo.DimScenario.txt" -q -c -C 65001 -t
"| " -r "\n" -S localhost -T
```

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

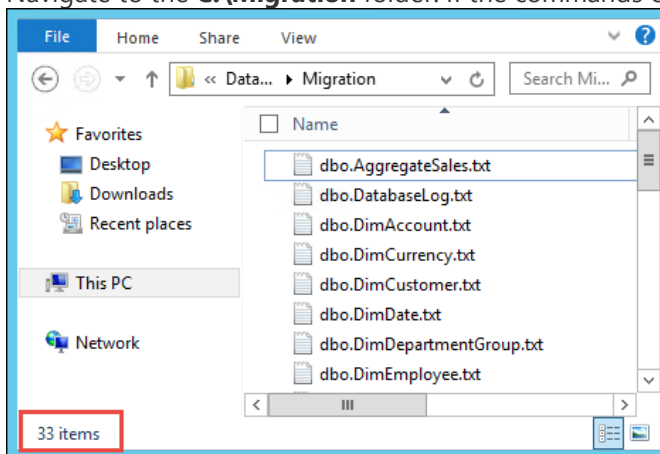


4. Create a new folder named **C:\Migration** on the local drive. This is where the bcp\_commands.bat will save data to.
5. Open a command prompt and execute **C:\LabFiles\bcp\_commands.bat**

```
C:\LabFiles>bcp_command.bat_
```

Note: 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.

6. Navigate to the **C:\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, download and install the latest version of the Microsoft Azure Storage tools from <http://aka.ms/downloadazcopy>.



- Open a command prompt and navigate to the **C:\Program Files (x86)\Microsoft SDKs\Azure\AzCopy** folder.
- Update the following command with your storage account name and key, and execute it to begin copying your data files to Azure (all of the text is a single command).

```
AzCopy /Source:"C:\Migration"
/dest:https://<YourStorageAccount>.blob.core.windows.net/migration
/destkey:<YourStorageAccountKey> /pattern:*.txt /NC:2
```

- Confirm all 33 files were transferred successfully

```
C:\Program Files (x86)\Microsoft SDKs\Azure\AzCopy>AzCopy /
1xPFJxoh8mc57PrFlguFMRcosW28yA= /pattern:*.txt /NC:2
Finished 33 of total 33 file(s).
[2016/09/16 21:47:47] Transfer summary:
Total files transferred: 33
Transfer successfully: 33
Transfer skipped: 0
Transfer failed: 0
Elapsed time: 00.00:00:03
C:\Program Files (x86)\Microsoft SDKs\Azure\AzCopy>_
```

Verify the files are in the correct storage container by navigating to your storage account, clicking on **blobs**, and selecting your **migration** container

| NAME                       | MODIFIED             | BLOB TYPE  | SIZE      |
|----------------------------|----------------------|------------|-----------|
| dbo.AggregateSales.txt     | 9/16/2016 4:47:43 PM | Block blob | 2.24 MB   |
| dbo.DatabaseLog.txt        | 9/16/2016 4:47:43 PM | Block blob | 38.8 KB   |
| dbo.DimAccount.txt         | 9/16/2016 4:47:43 PM | Block blob | 5.64 KB   |
| dbo.DimCurrency.txt        | 9/16/2016 4:47:43 PM | Block blob | 2.21 KB   |
| dbo.DimCustomer.txt        | 9/16/2016 4:47:43 PM | Block blob | 4.48 MB   |
| dbo.DimDate.txt            | 9/16/2016 4:47:43 PM | Block blob | 116.27 KB |
| dbo.DimDepartmentGroup.txt | 9/16/2016 4:47:43 PM | Block blob | 179 B     |

- Open SQL Server Management Studio, and connect to the CohoDW database of your SQL Data Warehouse using the **dataloader** account
- Execute the following to create a database scoped credential you will use to store the access key to the migration storage account

```
CREATE DATABASE SCOPED CREDENTIAL MigrationCredential
WITH IDENTITY = '<YourStorageAccountName>', SECRET = '<YourStorageAccountKey>'
```

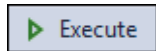
- Create an external data source by executing the following query. The external data source defines the location of your data and the credential used to access it. Again, be sure to replace the values with your own storage name and key.

```
CREATE EXTERNAL DATA SOURCE MigrationStor WITH (TYPE = HADOOP,
LOCATION=
'wasbs://migration@<YourStorageAccountName>.blob.core.windows.net',
CREDENTIAL = MigrationCredential);
```

8. 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 = '|'));
```

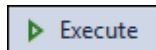
9. Open the **C:\LabFiles\CreateExternalTables.sql** file in SQL Server Management Studio and verify that you are connected to your Azure SQL Data Warehouse **CohoDW** database
10. This file contains all of the external table definitions for our tables and directly leverages the external data source and external file format we created above. Click **Execute** to create the external tables.



11. Run the following code to verify that 33 tables were created (your query should return 34 rows)

```
SELECT * FROM SYS.TABLES WHERE is_external = 1
```

12. From your **SQLCohoDW** virtual machine, open the **C:\LabFiles\LoadData.sql** file in SQL Server Management Studio
13. 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.



14. 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.

## Exercise 4: Investigate table distribution in Azure SQL Data Warehouse

In this exercise, you will compare the performance of round robin and hash table distributions in Azure SQL Data Warehouse.

### Task 1: Create test tables

1. From your SQLcohoDW virtual machine, open SQL Server Management Studio, and connect to the **CohoDW** database of your SQL Data Warehouse using the **dataloader** account
2. Open a new query window, and execute the following T-SQL commands.

These commands will create round robin and hashed versions of the FactInternetSales\_DimProduct tables. It will also create round robin, hashed and replicated versions of the DimDate tables. Note that the FactInternetSales tables were created earlier in the lab.

How are these table distribution types different?

**ROUND\_ROBIN** - This distribution type evenly distributes data across all nodes of the SQL Data Warehouse.

**HASH** - Hash distribution distributes data across the node according to a hash key that you specify. This can help prevent excessive data movement when joining data as data with the same hash will be guaranteed to be on the same node. Hash distribution is ideal for large fact tables.

**REPLICATED** - This distribution creates a copy of the table on every node. Replicated tables are good choice for small dimension tables.

```
CREATE TABLE DimDate_ROUNDROBIN
WITH (DISTRIBUTION = ROUND_ROBIN)
AS
 SELECT *
 FROM DimDate;

CREATE TABLE DimDate_HASHED
WITH (DISTRIBUTION = HASH(DateKey))
AS
 SELECT *
 FROM DimDate;

CREATE TABLE DimDate_REPLICATED
WITH (DISTRIBUTION = REPLICATE)
AS
 SELECT *
 FROM DimDate;

CREATE TABLE DimProduct_ROUNDROBIN
WITH (DISTRIBUTION = ROUND_ROBIN)
AS
 SELECT *
 FROM DimProduct;

CREATE TABLE DimProduct_HASHED
WITH (DISTRIBUTION = HASH(ProductKey))
AS
 SELECT *
 FROM DimProduct;
```

3. Copy/paste the following script into a new query window and execute it to update statistics of our newly created tables

Unlike Azure SQL Database, statistics are not automatically created and maintained in Azure SQL Data Warehouse. It is always a good idea to update statistics manually after a large data load operation.

```
CREATE STATISTICS stat_DimProduct_ROUNDROBIN ON DimProduct_ROUNDROBIN (ProductKey) WITH FULLSCAN;
CREATE STATISTICS stat_DimProduct_HASHED ON DimProduct_HASHED (ProductKey) WITH FULLSCAN;
CREATE STATISTICS stat_DimDate_ROUNDROBIN ON DimDate_ROUNDROBIN (DateKey) WITH FULLSCAN;
CREATE STATISTICS stat_DimDate_HASHED ON DimDate_HASHED (DateKey) WITH FULLSCAN;
CREATE STATISTICS stat_DimDate_REPLICATED ON DimDate_REPLICATED (DateKey) WITH FULLSCAN;
```

## Task 2: Create test tables

1. Open SQL Server Management Studio.
2. Set the connection to use the **CohoDW** database and your **demouser** account and open a New Query window. Copy/paste the following query into the new query window, but do not Execute. This query will be used to investigate the table distribution performance.

```
SELECT
 [distribution_id],
 [sql_spid],
 [pdw_node_id],
 [request_id],
 [dms_step_index],
 [type],
 [start_time],
 [end_time],
 [status],
 'DBCC PDW_SHOWEXECUTIONPLAN ('+CAST([distribution_id] AS
 VARCHAR(MAX))+','+CAST([sql_spid] AS VARCHAR(MAX))+');' as planSQL
FROM sys.dm_pdw_dms_workers
WHERE [status] <> 'StepComplete' and [status] <> 'StepError' and [status] <>
 'StepCancelled'
ORDER BY request_id, [dms_step_index];
```

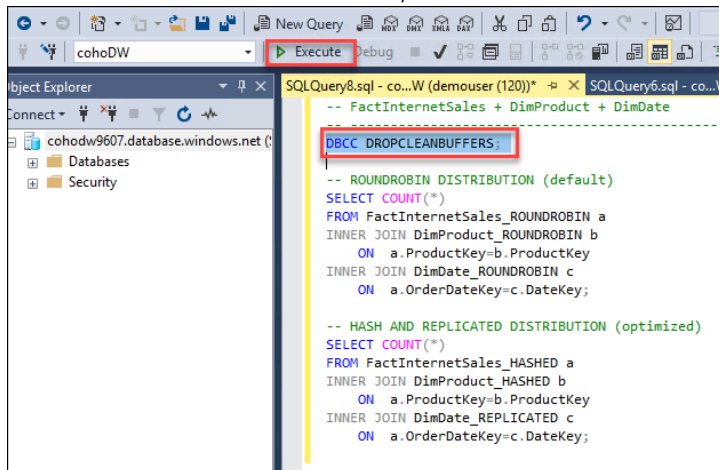
- Open another query window, and copy/paste the following script into it

```
-- FactInternetSales + DimProduct + DimDate
-- -----
DBCC DROPCLEANBUFFERS;

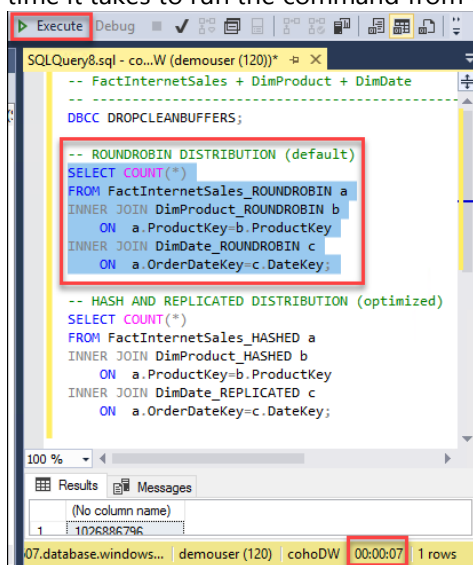
-- ROUNDROBIN DISTRIBUTION (default)
SELECT COUNT(*)
FROM FactInternetSales_ROUNDROBIN a
INNER JOIN DimProduct_ROUNDROBIN b
 ON a.ProductKey=b.ProductKey
INNER JOIN DimDate_ROUNDROBIN c
 ON a.OrderDateKey=c.DateKey;

-- HASH AND REPLICATED DISTRIBUTION (optimized)
SELECT COUNT(*)
FROM FactInternetSales_HASHED a
INNER JOIN DimProduct_HASHED b
 ON a.ProductKey=b.ProductKey
INNER JOIN DimDate_REPLICATED c
 ON a.OrderDateKey=c.DateKey;
```

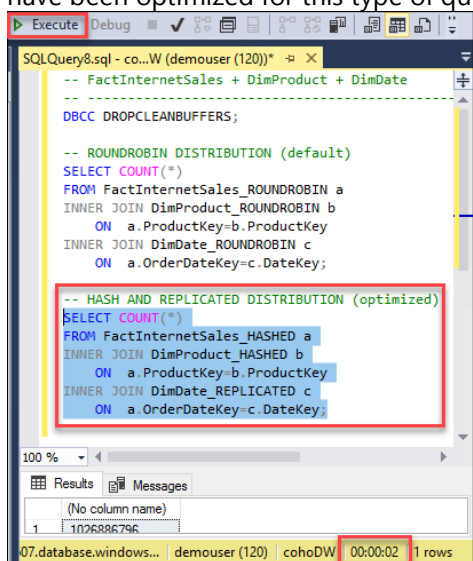
- You can clear the buffer cache by using the DBCC DROPCLEANBUFFERS command. Before each run, you can execute this to prevent a "warm cache" from impacting the performance of the query. Highlight the **DBCC DROPCLEANBUFFERS** command, and click the **Execute** button.



- Execute the **ROUNDROBIN DISTRIBUTION** query by highlighting it, and clicking the Execute button. Note the time it takes to run the command from the lower right of the window.



- Repeat steps 4 and 5 several times, note how long it takes for the query to run. There will be some variability at the scale we have chosen. Higher DWU settings will improve the performance and reduce the variability.
- Repeat the process again using the **HASH AND REPLICATED DISTRIBUTION** script. This script uses tables that have been optimized for this type of query.



- After a few runs, you should see a pattern emerge. The hash and replicated tables will outperform the roundrobin tables. This is because the data to satisfy the joins in the query will always be located on the same node for the hashed tables and the replicated DimDate\_REPLICATED table has been completely replicated to every node. This minimized the relatively expensive data movement operation. The round roundrobin tables have data evenly distributed to all nodes which require a lot of data movement to satisfy the joins in the query resulting in longer run times.

9. To dig a little deeper, open a new query window, and copy/paste the following code, and execute it

```
DBCC DROPLEANBUFFERS;

SELECT COUNT(*)
FROM FactInternetSales_ROUNDROBIN a
INNER JOIN DimProduct_ROUNDROBIN b
 ON a.ProductKey=b.ProductKey
INNER JOIN DimDate_ROUNDROBIN c
 ON a.OrderDateKey=c.DateKey
WHERE a.SalesAmount BETWEEN 0 AND 1000;
```

10. While this query is running, **quickly go back to the first query window we opened and execute the diagnostics query**. If you get no results execute step 9 again and try again.
11. In the query results, note the type of DMS worker. The PARALLEL\_COPY\_READER type indicates data is being sent between nodes in the cluster.

|   | distribution_id | sql_spid | pdw_node_id | request_id | dms_step_index | type                 | start_time              | end_time | status          | planSQL                             |
|---|-----------------|----------|-------------|------------|----------------|----------------------|-------------------------|----------|-----------------|-------------------------------------|
| 1 | 1               | 704      | 169         | QID2252    | 0              | PARALLEL_COPY_READER | 2017-12-01 02:29:15.607 | NULL     | QueryIssueBegin | DBCC PDW_SHOWEXECUTIONPLAN (1,704); |
| 2 | 2               | 713      | 169         | QID2252    | 1              | PARALLEL_COPY_READER | 2017-12-01 02:29:15.607 | NULL     | QueryIssueBegin | DBCC PDW_SHOWEXECUTIONPLAN (2,713); |
| 3 | 3               | 715      | 169         | QID2252    | 2              | PARALLEL_COPY_READER | 2017-12-01 02:29:15.607 | NULL     | QueryIssueBegin | DBCC PDW_SHOWEXECUTIONPLAN (3,715); |
| 4 | 4               | 705      | 169         | QID2252    | 3              | PARALLEL_COPY_READER | 2017-12-01 02:29:15.623 | NULL     | QueryIssueBegin | DBCC PDW_SHOWEXECUTIONPLAN (4,705); |
| 5 | 5               | 708      | 169         | QID2252    | 4              | PARALLEL_COPY_READER | 2017-12-01 02:29:15.623 | NULL     | QueryIssueBegin | DBCC PDW_SHOWEXECUTIONPLAN (5,708); |
| 6 | 6               | 710      | 169         | QID2252    | 5              | PARALLEL_COPY_READER | 2017-12-01 02:29:15.623 | NULL     | QueryIssueBegin | DBCC PDW_SHOWEXECUTIONPLAN (6,710); |
| 7 | 7               | 703      | 169         | QID2252    | 6              | PARALLEL_COPY_READER | 2017-12-01 02:29:15.623 | NULL     | QueryIssueBegin | DBCC PDW_SHOWEXECUTIONPLAN (7,703); |
| 8 | 8               | 706      | 169         | QID2252    | 7              | PARALLEL_COPY_READER | 2017-12-01 02:29:15.623 | NULL     | QueryIssueBegin | DBCC PDW_SHOWEXECUTIONPLAN (8,706); |
| 9 | 9               | 716      | 169         | QID2252    | 8              | PARALLEL_COPY_READER | 2017-12-01 02:29:15.623 | NULL     | QueryIssueBegin | DBCC PDW_SHOWEXECUTIONPLAN (9,716); |

12. **Scale your SQL Data Warehouse back down to 100 to prevent excessive cost.**

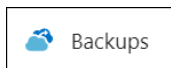
## Exercise 5: Create an Analysis Services Model

Coho has provided you with an existing Analysis Services model for use with the Data Warehouse. They have asked you to use this model to support ad-hoc query access from Power BI.

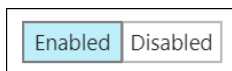
In this exercise, you will configure backup, restore for Analysis Services, and create a tabular model to allow ad-hoc queries from client tools.

### Task 1: Configure Analysis Services backup

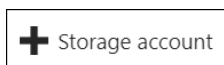
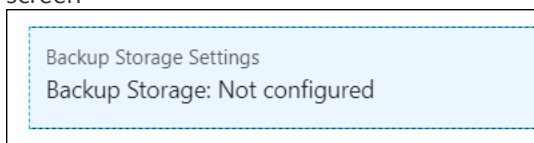
1. Navigate to your Analysis Services instance and click on **Backups** under settings in the menu



2. Set backups to **Enabled**



3. Click the **Backup Storage Settings** tile and then select **+Storage account** from the menu bar at the top of the screen



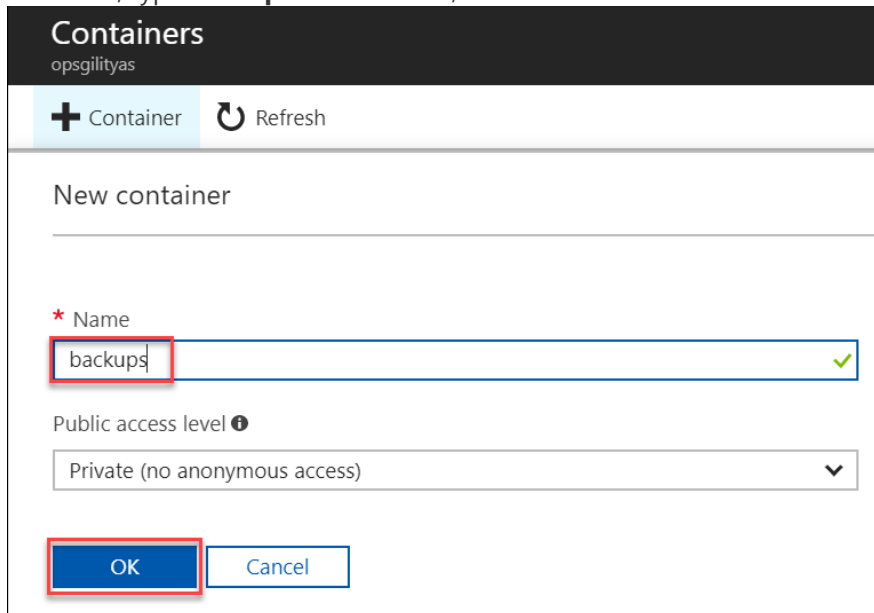
4. On the Create storage account blade, use the following configurations, and click **OK**:
  - Name: **Choose a unique storage account name**
  - Performance: **Standard**
  - Replication: **Locally-redundant storage (LRS)**
  - Location: **The same location you have been using for this lab**

A screenshot of the "Create storage account" blade in the Azure portal. The form includes:

- Name:** A text box containing "cohoas" with a green checkmark and ".core.windows.net" below it.
- Performance:** Two buttons, "Standard" (selected) and "Premium".
- Replication:** A dropdown menu showing "Locally-redundant storage (LRS)".
- Location:** A dropdown menu showing "South Central US".

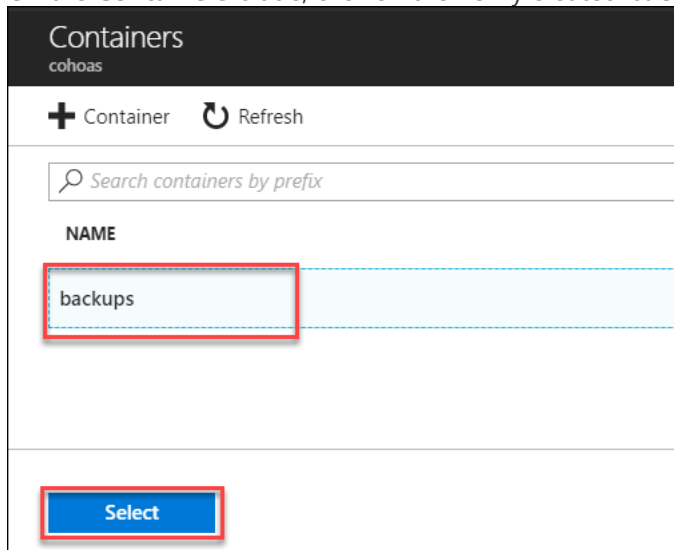


5. Choose the storage account you just created to open the Containers blade. Click **+Container** to create a new container, type **backups** for the name, and click **OK**



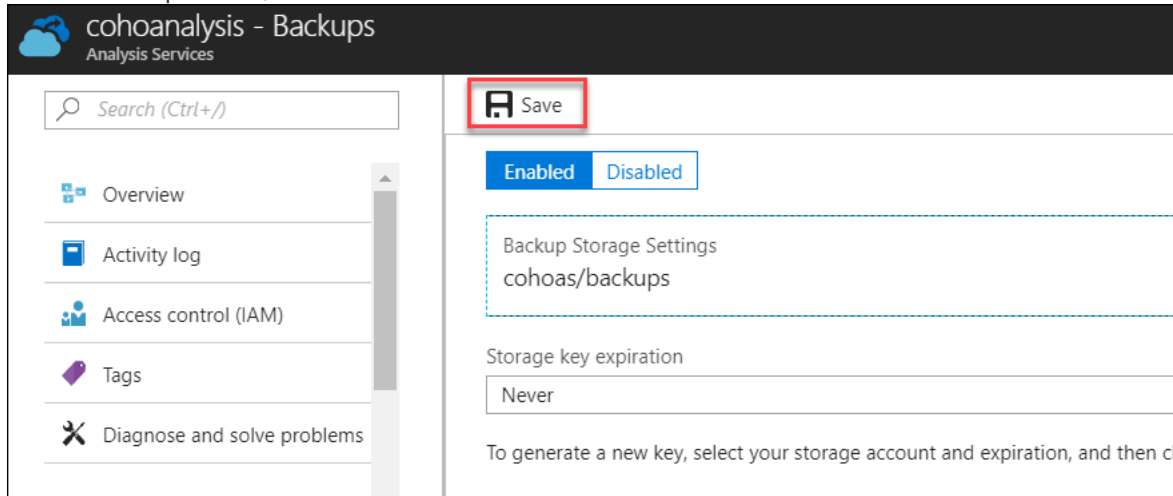
The screenshot shows the 'Containers' blade for the storage account 'opsgilityas'. At the top, there are buttons for '+ Container' and 'Refresh'. Below this is a 'New container' section. It includes a text input field for 'Name' with the value 'backups' and a green checkmark icon to its right. Below the name field is a dropdown menu for 'Public access level' with the selected option 'Private (no anonymous access)'. At the bottom of the form, there are two buttons: 'OK' and 'Cancel'. The 'OK' button is highlighted with a red box.

6. On the **Containers** blade, click on the newly created **backups** Container, then click **Select**



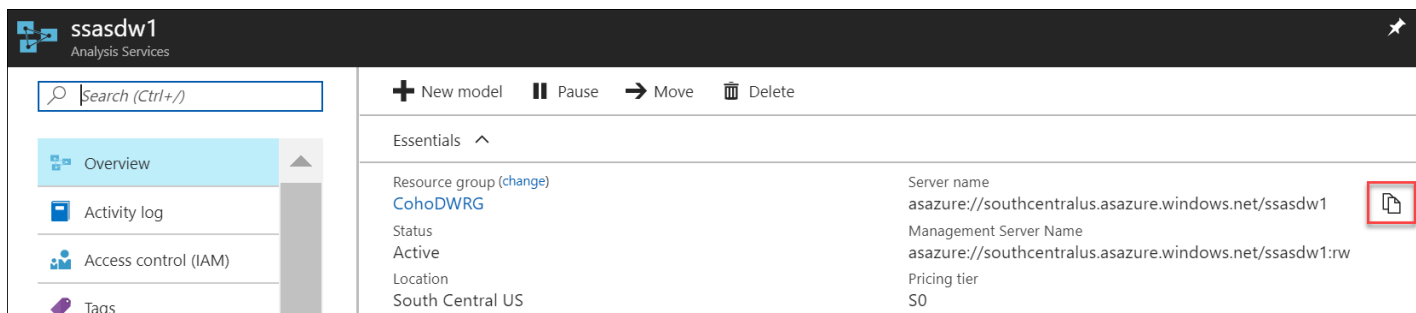
The screenshot shows the 'Containers' blade for the storage account 'cohoas'. It features a search bar with the placeholder text 'Search containers by prefix'. Below the search bar is a table with a single row containing the container name 'backups'. This row is highlighted with a light blue background and a red border. At the bottom of the blade, there is a blue button labeled 'Select', which is also highlighted with a red box.

- On the Backups blade, click the **Save** button

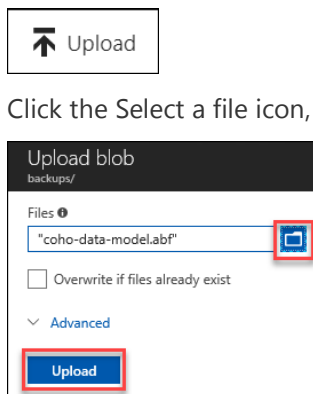


## Task 2: Restore Analysis Services backup

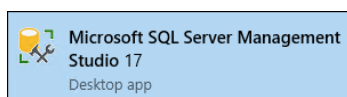
- From the Analysis Services overview blade, hover over the server name and click the copy icon to **copy the Server name**. Save this into notepad for use later in this task.



- In the Azure Portal, navigate to the storage account you just created, click the **Blobs** tile, and open the **backups** container
- Click **Upload**
- Click the Select a file icon, and upload the **C:\LabFiles\coho-data-model.abf** file, and click **Upload**

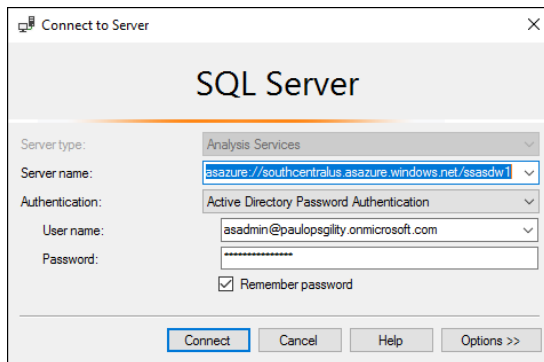


- Login to your SQLcohoDW virtual machine, and open **SQL Server Management Studio**



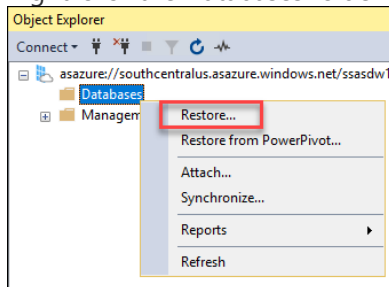
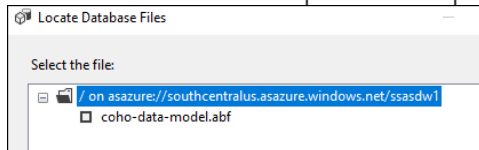
## 6. Connect to your Analysis Server.

- Server Type: **Analysis Server**
- Server name: ***the server name you copied earlier***
- Authentication: **Active Directory Password Authentication**
- User name: [asadmin@<subscription\\_name>.<domain>](#)

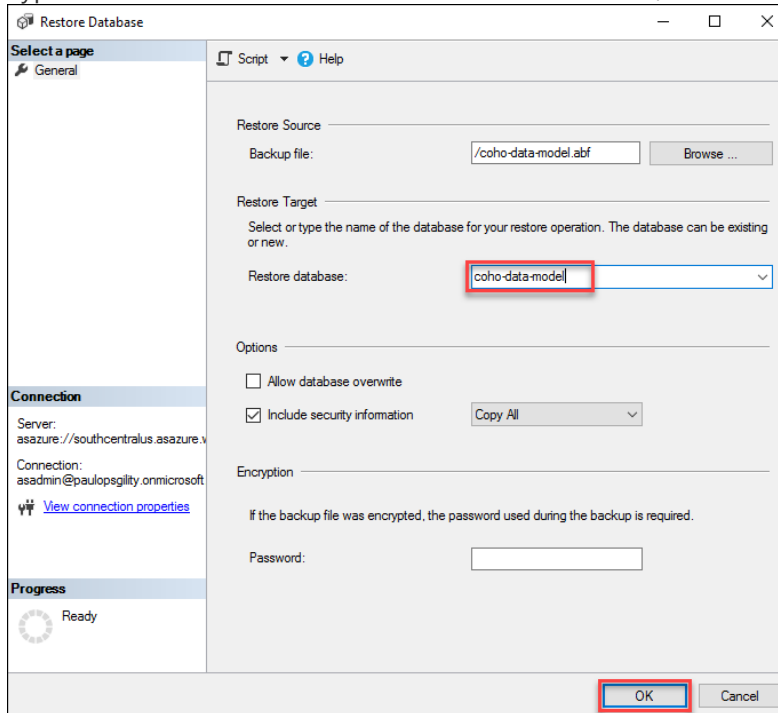


Note that if you are using your own organizational account instead of the one we created earlier in the lab then you will put that in for the user name.

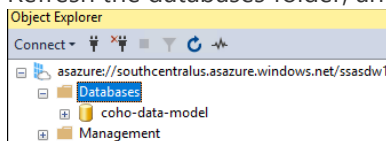
The User name setting should be in the form <name>@<your-domain>. If you do not know your domain name, you can get it by navigating to the Azure Portal and hovering over your login information in the upper right corner of your browser window.

7. Right-click the Databases folder and choose **Restore...**8. Select the backup file by clicking the **Browse** button and selecting the **coho-data-model.abf** file from the storage account. Click **OK** to accept the backup file

9. Type **coho-data-model** into the Restore database field, and click **OK** to restore the database

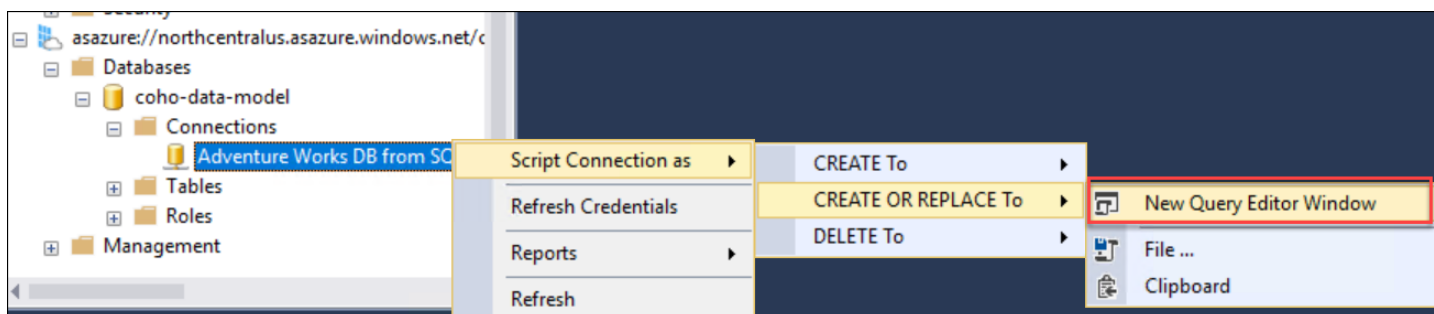


10. Refresh the databases folder, and you should see your coho-data-model now



### Task 3: Update Analysis Services connections

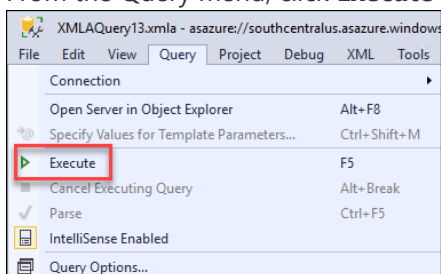
1. From SQL Server Management Studio, expand the **coho-data-model** database, expand **Connections**, right-click **Adventure Works DB from SQL**, script the connection as **CREATE OR REPLACE To** a **New Query Editor Window**



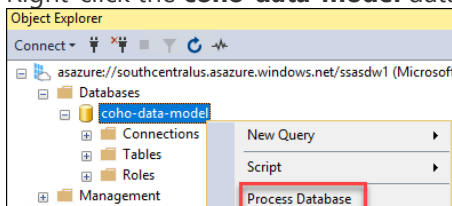
2. Modify the connection string to point to your SQL Data Warehouse

```
{
 "createOrReplace": {
 "object": {
 "database": "coho-data-model",
 "dataSource": "CohoDW"
 },
 "dataSource": {
 "name": "CohoDW",
 "connectionString": "Server=tcp:cohodw9607.database.windows.net,1433;Initial Catalog=cohoDW",
 "impersonationMode": "impersonateCurrentUser",
 "annotations": [
 {
 "name": "ConnectionEditUISource",
 "value": "SqlServer"
 }
]
 }
 }
}
```

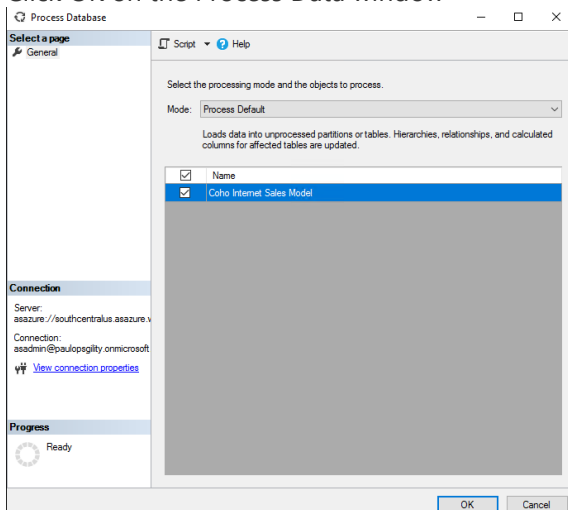
3. From the Query menu, click **Execute** to update the connection



4. Right-click the **coho-data-model** database, and choose **Process Database**



5. Click OK on the Process Data window



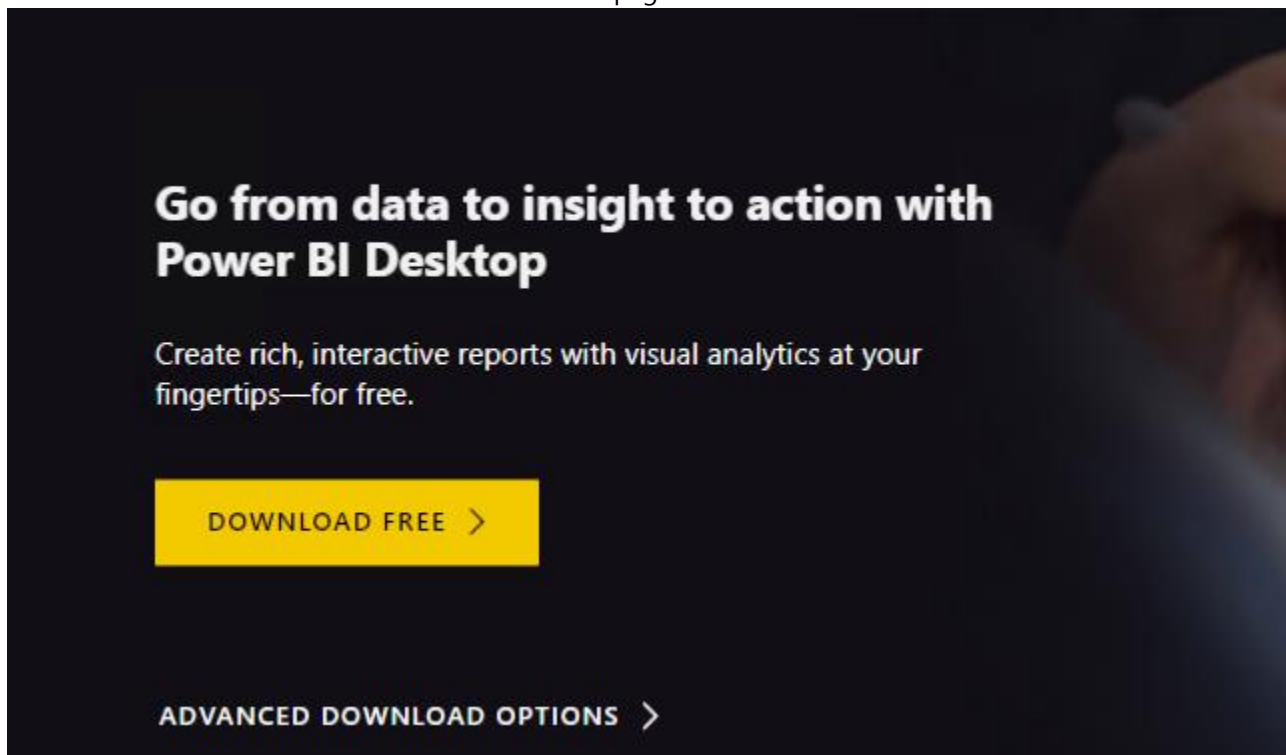
6. Close the Process Data window

## Exercise 6: Visualize data with Power BI Desktop

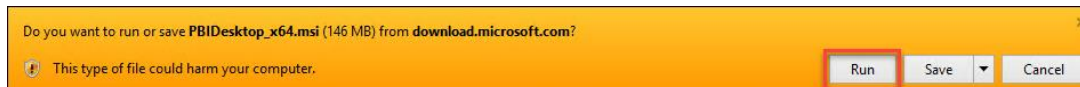
In this exercise, you will setup integration with Power BI Desktop

### Task 1: Install Power BI Desktop

1. Connect to the **SQLcohoDW** virtual machine
2. In a web browser, navigate to the Power BI Desktop download page (<https://powerbi.microsoft.com/en-us/desktop/>)
3. Select the **Download Free** link in the middle of the page



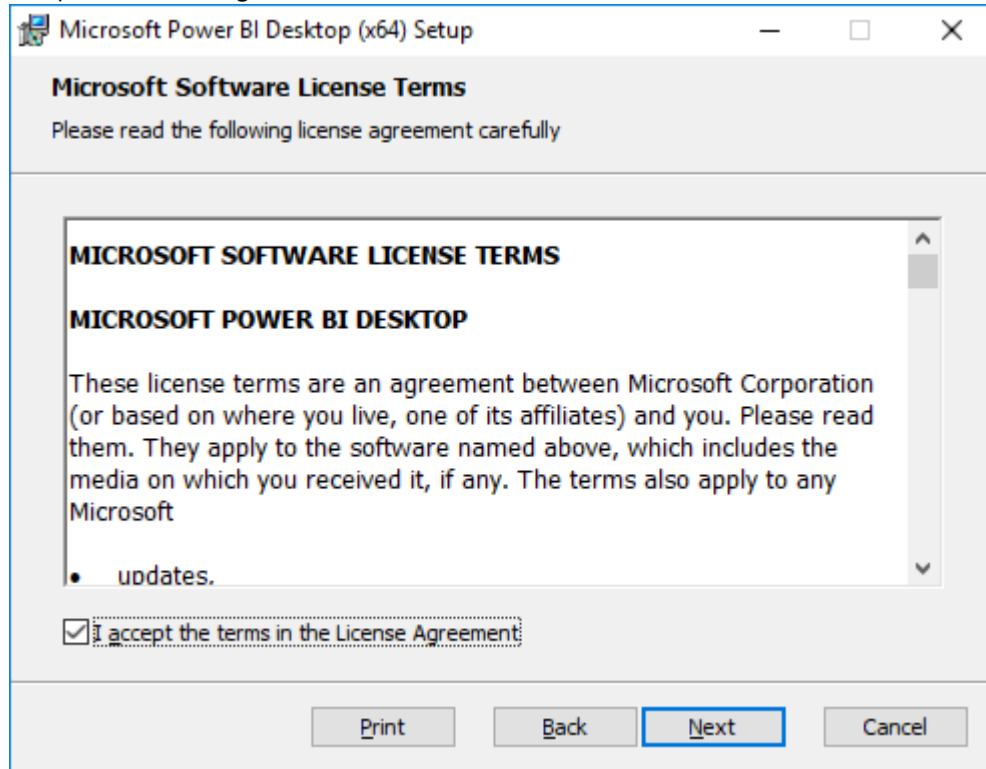
4. Run the installer



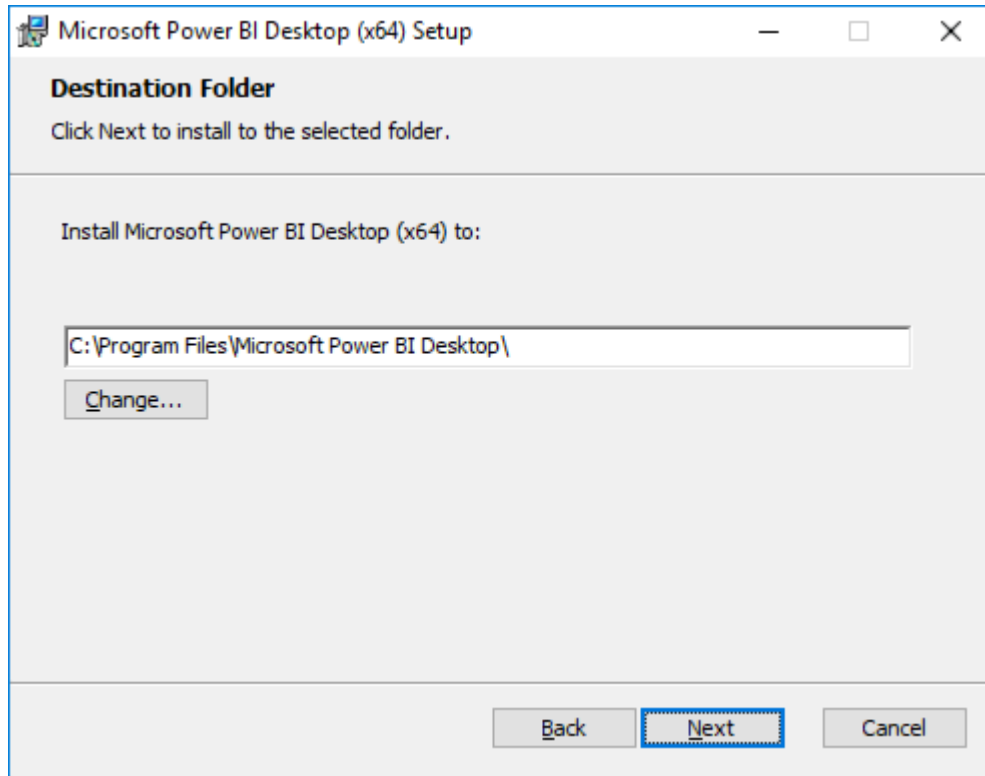
5. Select **Next** on the welcome screen



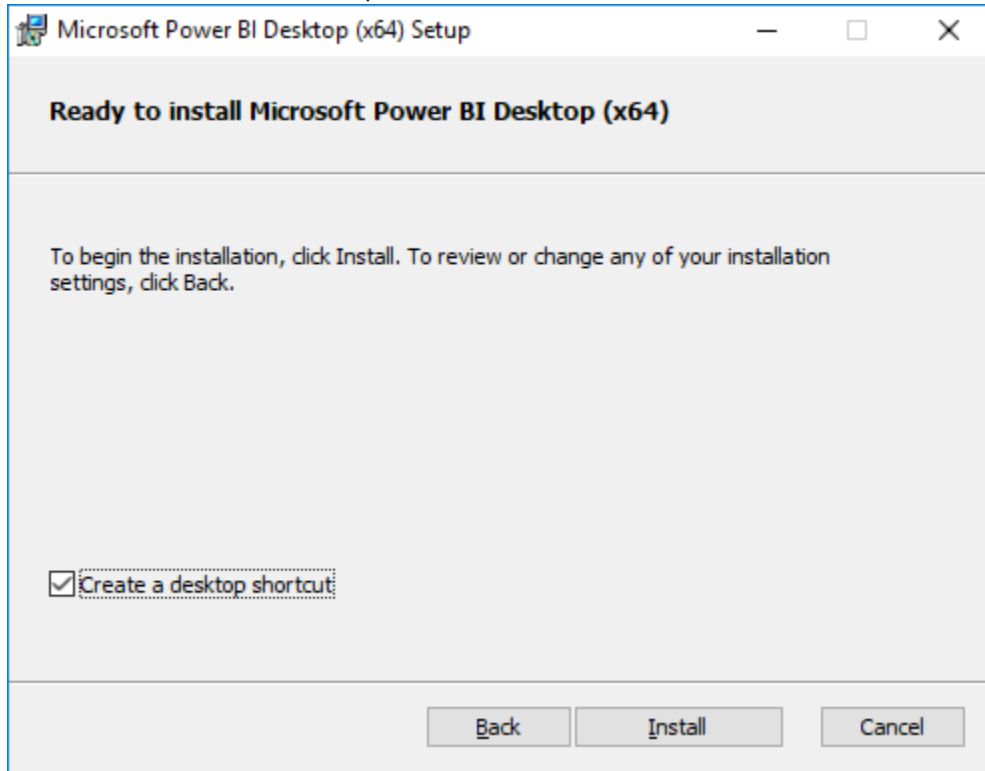
6. Accept the license agreement, and select **Next**



7. Leave the default destination folder, and select **Next**



8. Make sure the Create a desktop shortcut box is checked, and select **Install**.



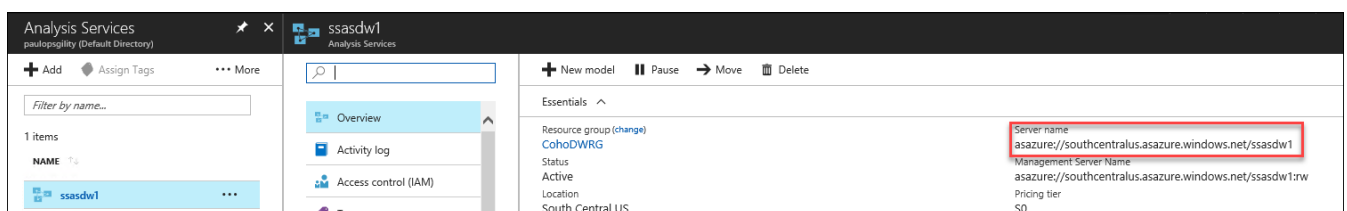


9. Verify that Launch Microsoft Power BI Desktop is checked, and select **Finish**

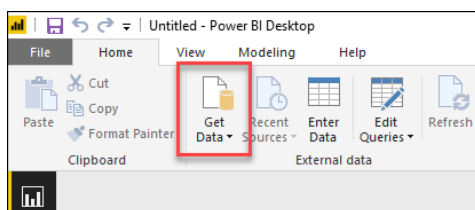


## Task 2: Query data with Power BI Desktop

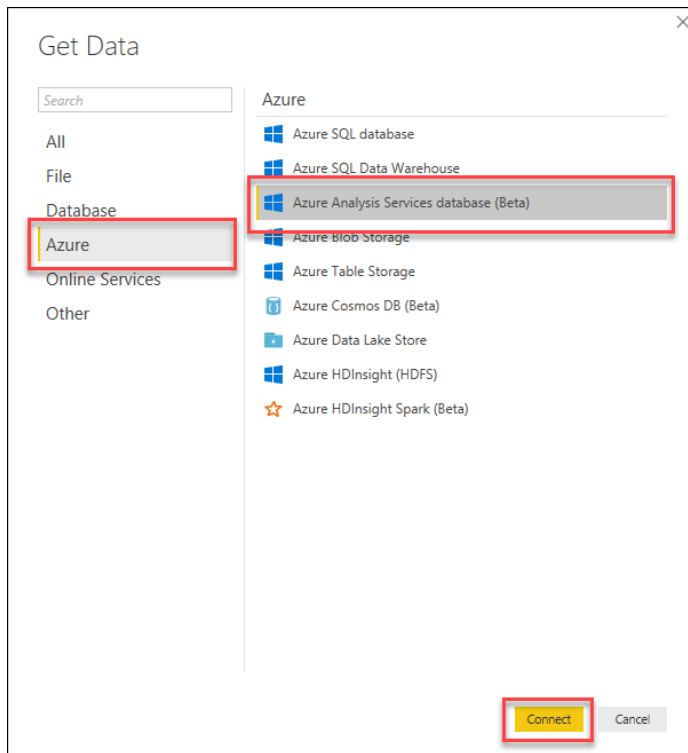
1. Connect to the Azure Portal, and navigate to your Azure Analysis Services
2. Make note of your Analysis Server name to use in your data source configuration later in this task



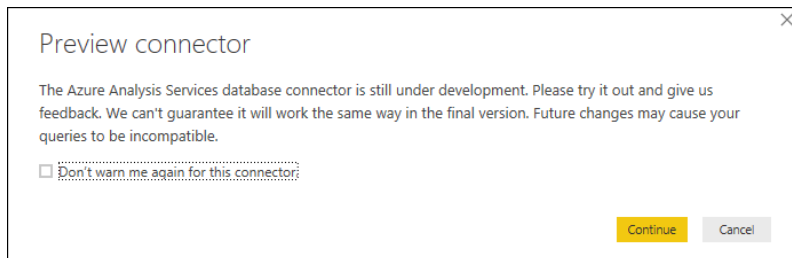
3. From within Power BI, click the **Get Data** button.



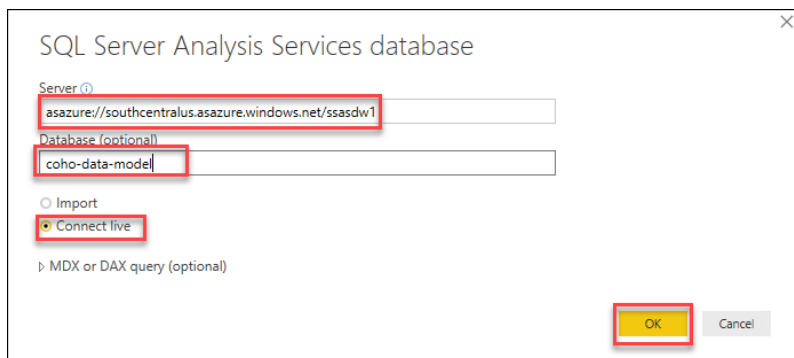
- On the Get Data window, select **Azure** from the list on the left. Then, choose Azure Analysis Services database, and click **Connect**



- If you get a Preview connector warning, click **Continue**

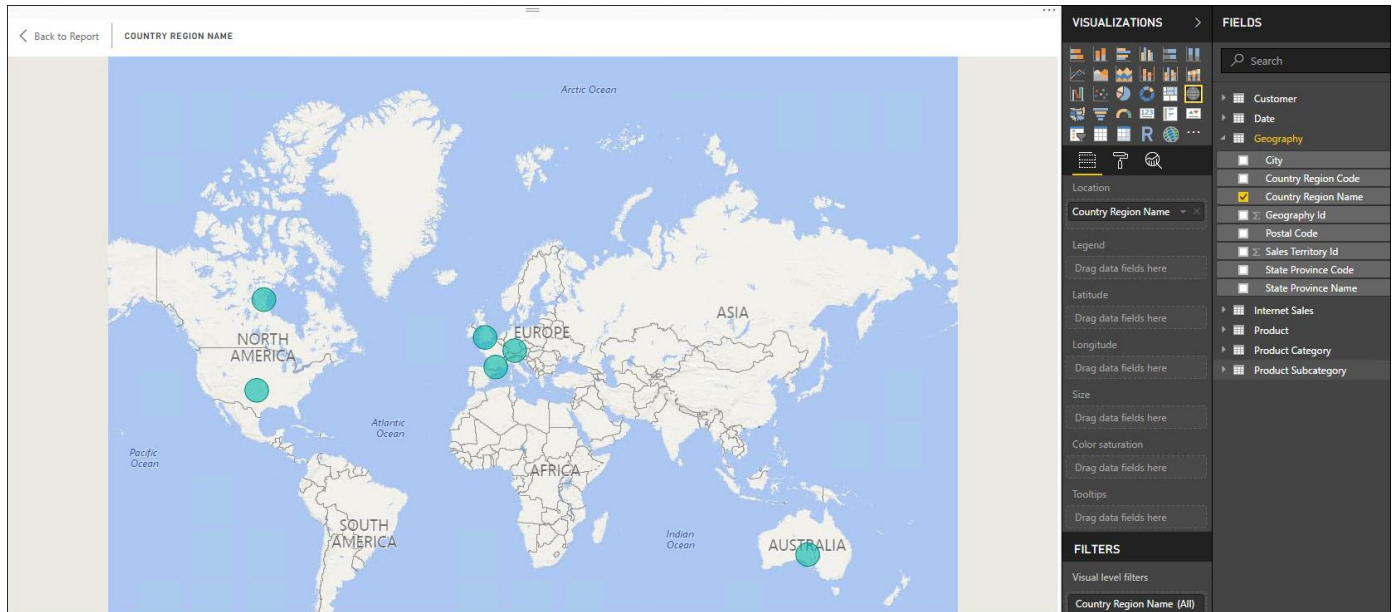


- On the **SQL Server Analysis Services database** screen, provide the name of your Analysis Server service, type, make sure that **Connect live** is selected, and click **OK**

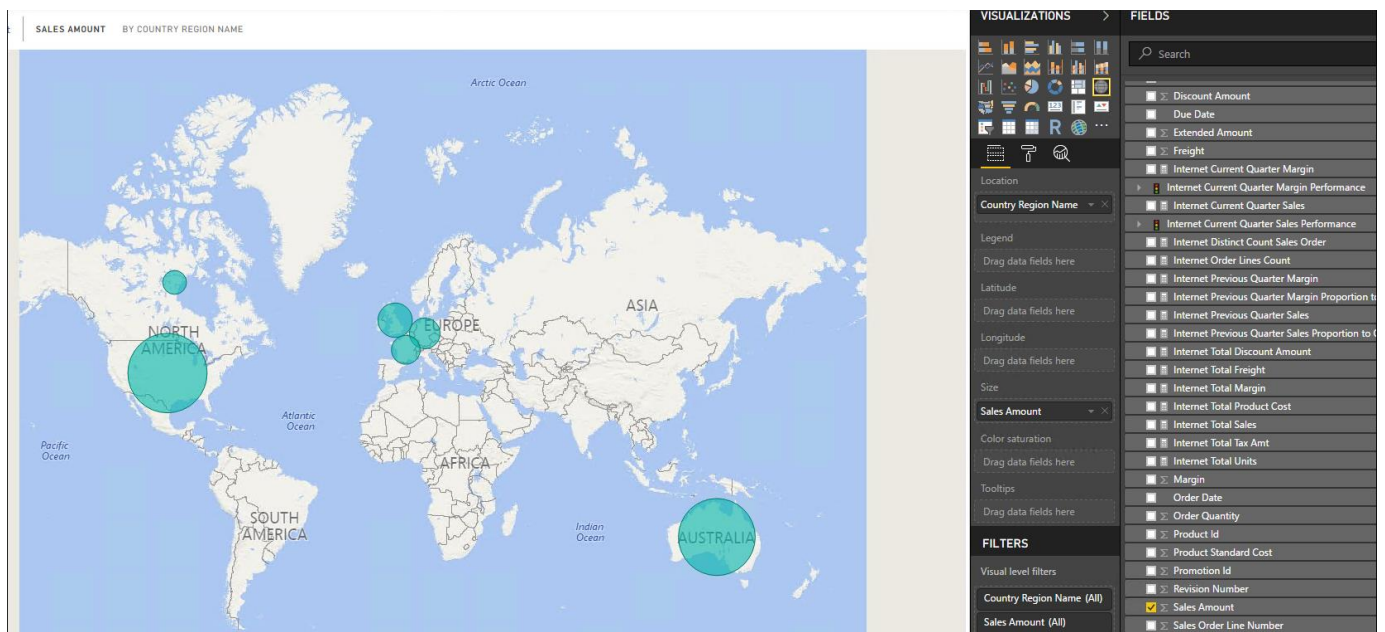


- Login with your Active Directory Azure Portal credentials

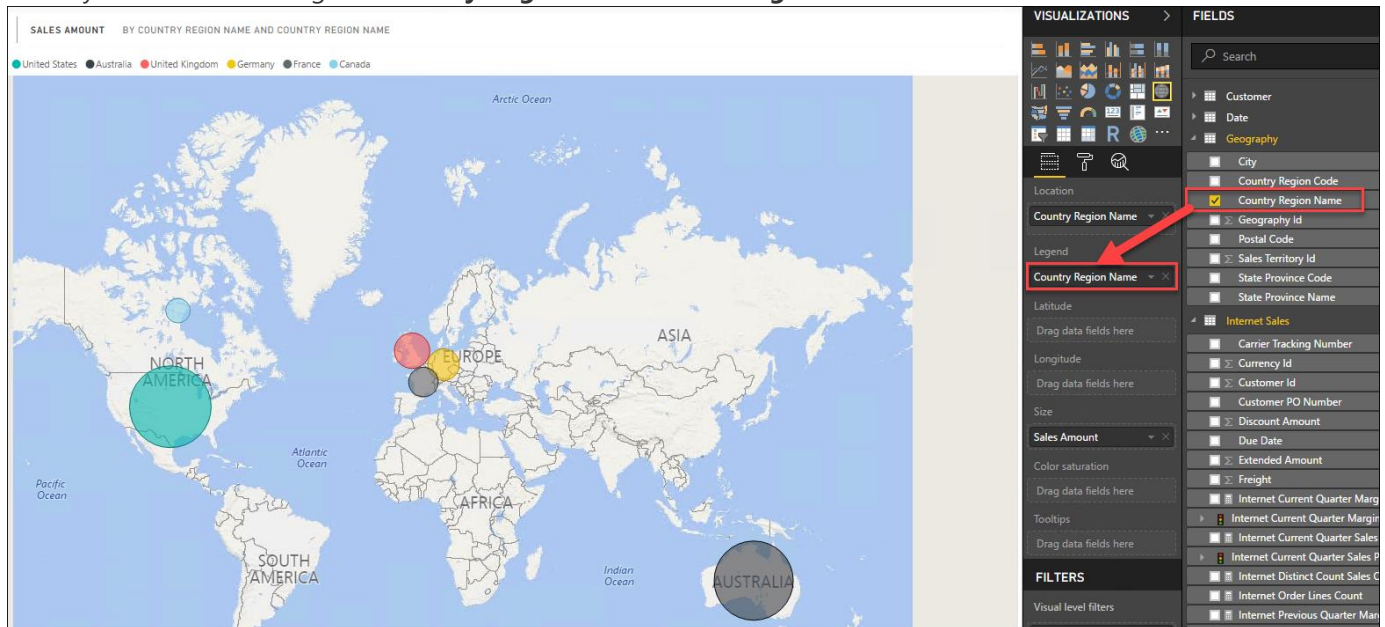
8. In the Fields blade in the dark grey side bar to the right, expand the **Geography**, and check the box next to **Country Region Name**. This will automatically launch the map visualization, because PowerBI is smart enough to understand this is geographic data.



9. The circles that PowerBI adds to the map are simply every country in which Coho had sales. Let's add the sales amount to this to make the map a little more interesting. Add the **Sales Amount** from the **Internet Sales** 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 country.



10. We want to see a little more specific detail around what these circles actually mean, so let's add a legend to identify the countries. Drag the **Country Region Name** under **Legend**.



11. Click the **Save** button in the top left of your screen, name your report **Sales by country**, and click **Save**

## After the hands-on Lab

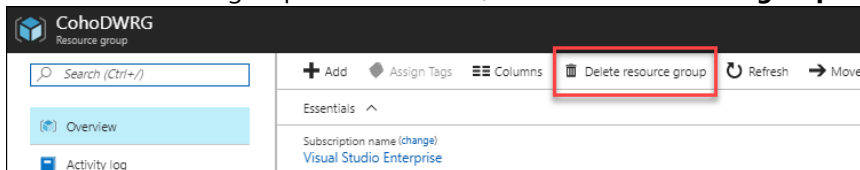
To prevent excessive charges, you should cleanup the resources you have created for this lab.

### Task 1: Cleanup resource groups

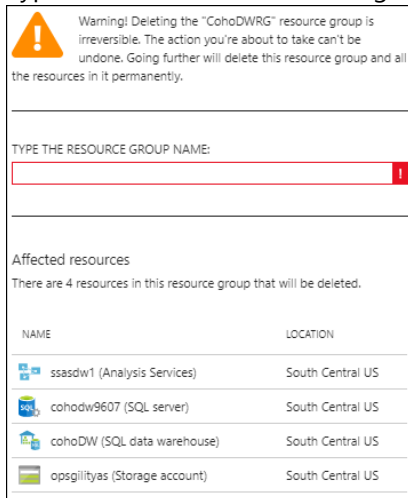
1. From the Azure Portal, navigate to the **CohoDWRG** resource group



2. From the resource group overview blade, click **Delete resource group**



3. Type the name of the resource group to confirm the delete request, and click **delete**



4. Repeat the process to delete the **EDWmigrationStor** and **OnPremEnvironment** resource groups

You should follow all steps provided *after* attending the hands-on lab.