# DP 200 - Implementing a Data Platform Solution

# Lab 5 - Working with Relational Data Stores in the Cloud

**Estimated Time**: 75 minutes

**Pre-requisites**: It is assumed that the case study for this lab has already been read. It is assumed that the content and lab for module 1: Azure for the Data Engineer has also been completed

**Lab files**: The files for this lab are located in the Allfiles\Labfiles\Starter\DP-200.5 folder.

## Lab overview

The students will be able to provision an Azure SQL Database and Azure SQL Data Warehouse and be able to issue queries against one of the instances that are created. They will be also be able to integrate SQL Data Warehouse with a number of other Data platform technologies and use PolyBase to load data from one data source into Azure SQL Data Warehouse.

## Lab objectives

After completing this lab, you will be able to:

1. Use Azure SQL Database
2. Describe Azure Data Warehouse
3. Creating and Querying an Azure SQL Data Warehouse
4. Using PolyBase to Load Data into Azure SQL Data Warehouse

## Scenario

You are the senior data engineer at AdventureWorks, and you are working with your team to transition relational database systems from on-premises SQL Servers to relational database located in Azure. You will begin by creating an instance of SQL Server with the company's sample database that will be handed of the junior data engineers to perform some testing of departmental databases.

You will then provision a SQL Data Warehouse and test that the provisioning of the server is successful by testing a sample database with a series of queries. You will then use PolyBase to load dimension tables from Azure Blob and Azure Databricks to test that the integration of these data platform technologies with Azure SQL Data Warehouse.

At the end of this lad, you will have:

1. Use Azure SQL Database
2. Describe Azure Data Warehouse
3. Creating and Querying an Azure SQL Data Warehouse
4. Using PolyBase to Load Data into Azure SQL Data Warehouse

**IMPORTANT**: As you go through this lab, make a note of any issue(s) that you have encountered in any provisioning or configuration tasks and log it in the table in the document located at \Labfiles\DP-200-Issues-Doc.docx. Document the Lab number, note the technology, Describe the issue, and what was the resolution. Save this document as you will refer back to it in a later module.

## Exercise 1: Use Azure SQL Database

Estimated Time: 15 minutes

Individual exercise

The main task for this exercise are as follows:

1. Create and configure a SQL Database instance.

### Task 1: Create and configure a SQL Database instance.

1. In the Azure portal, navigate to the **+ Create a resource** blade.
2. In the New blade, navigate to the **Search the Marketplace** text box, and type the word **SQL Database**. Click **SQL Database** in the list that appears.
3. In the **SQL Database** blade, click **Create**.
4. From the **Create SQL Database** blade, create an Azure SQL Database with the following settings:
   * Subscription: the name of the subscription you are using in this lab
   * Resource group name: **awrgstudxx**, where **xx** are your initials.
   * Database name: **DeptDatabasesxx**, where **xx** are your initials.
   * Server: Create a new server with the following settings and click on **Select**:
     + Server name: **sqlservicexx**, where **xx** are your initials
     + Server admin login: **xxsqladmin**, where **xx** are your initials
     + Password: **Pa55w.rd**
     + Confirm Password: **Pa55w.rd**
     + Location: choose a **location** near to you.
     + Allow Azure services to access server: **checked**
   * Click on **Additional settings** tab, under **Data Source**, click on **Sample**.
   * Leave the remaining settings to their defaults
5. In the **Create SQL Database**\* blade, click **Review + create**.
6. After the validation of the **Create SQL Database**\* blade, click **Create**.

**Note**: The provision will takes approximately 4 minutes.

**Result**: After you completed this exercise, you have an Azure SQL Database instance

## Exercise 2: Describe Azure Data Warehouse

Estimated Time: 15 minutes

Individual exercise

The main tasks for this exercise are as follows:

1. Create and configure a SQL Data Warehouse instance.
2. Configure the Server Firewall

### Task 1: Create and configure a SQL Data Warehouse instance.

1. In the Azure portal, navigate to the **+ Create a resource** blade.
2. In the New blade, navigate to the **Search the Marketplace** text box, and type the word **SQL Data**. Click **SQL Data Warehouse** in the list that appears.
3. In the **SQL Data Warehouse** blade, click **Create**.
4. From the **SQL Data Warehouse** blade, create an Azure SQL Database with the following settings:
   * Database name: **Warehousexx**, where **xx** are your initials.
   * Subscription: the name of the subscription you are using in this lab
   * Resource group name: **awrgstudxx**, where **xx** are your initials.
   * Select Source: **Sample**
   * Select Sample: **AdventureWorksDW**
   * Server: **SQLServicexx**
   * Performance Level: **Gen2 DW200c**
5. In the **SQL Data Warehouse**\* blade, click **Create**.

**Note**: The provision will takes approximately 7 minutes.

### Task 2: Configure the Server Firewall

1. In the Azure portal, in the blade, click **Resource groups**, and then click **awrgstudxx**, and then click on **awdlsstudxx**, where **xx** are your initials
2. Click on **sqlservicexx**, where **xx** are your initials.
3. In the **sqlservicexx** screen, click on **Firewalls and virtual networks**.
4. In the sqlservicexx - Firewalls and virtual networks screen, click on the option **+ Add client IP**, and then click on **Save**.

**Note**: You will receive a message stating that the the server firewall rules have been successfully updated

**Result**: After you completed this exercise, you have created an Azure SQL Data Warehouse instance and configures the server firewall to enable connections against it.

## Exercise 3: Creating an Azure SQL Data Warehouse

Estimated Time: 25 minutes

Individual exercise

The main tasks for this exercise are as follows:

1. Install SQL Server Management Studio and connect to a SQL Data Warehouse instance.
2. Create a SQL Data Warehouse database
3. Create SQL Data Warehouse tables

**Note**: If you are not familiar with Transact-SQL, statements are available for the following labs in the following location **Allfiles\Labfiles\Starter\DP-200.5\SQL DW Files**

### Task 1: Install SQL Server Management Studio and connect to a SQL Data Warehouse instance.

1. In the Azure Portal, in the **sqlservicexx - Firewalls and virtual networks**, in the blade, click on **Properties**
2. Copy the **"Server name"** and paste it into Notepad.
3. Download [SQL Server Management Studio](https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-2017) and install onto your machine
4. On the windows desktop, click on the **Start**, and type **"SQL Server"** and then click on **Microsoft SQL Server Management Studio 17**
5. In the **Connect to Server** dialog box, fill in the following details
   * Server Name: **sqlservicexx.database.windows.net**
   * Authentication: **SQL Server Authentication**
   * Username: **xxsqladmin**
   * Password: **Pa55w.rd**
6. In the **Connect to Server** dialog box, click **Connect**

### Task 2: Create a SQL Data Warehouse database.

1. In **SQL Server Management Studio**, in Object Explorer, right click **sqlservicexx.database.windows.net** and click on **New Query**.
2. In the query window, create a DataWarehouse database named **DWDB**, with a service objective of DW100c (Gen2) and a maximum size of 1024GB.

CREATE DATABASE DWDB COLLATE SQL\_Latin1\_General\_CP1\_CI\_AS

(

EDITION = 'DataWarehouse'

, SERVICE\_OBJECTIVE = 'DW100c'

, MAXSIZE = 1024 GB

);

**Note**: The creation of the database takes approximately 2 minutes.

1. In the **awdlsstudxx** blade, click on **Access keys**, and then click on the copy icon next to the **Storage account name** and paste it into Notepad.

### Task 3: Create SQL Data Warehouse tables.

1. In **SQL Server Management Studio**, in Object Explorer, right click **DWDB** under **sqlservicexx.database.windows.net** and click on **New Query**.
2. Create a table named **dbo.Users** with a **clustered columnstore** index with a distribution of **replicate** with the following columns:

| **column name** | **data type** | **Nullability** |
| --- | --- | --- |
| userId | int | NULL |
| City | nvarchar(100) | NULL |
| Region | nvarchar(100) | NULL |
| Country | nvarchar(100) | NULL |

You can use this script.

CREATE TABLE dbo.Users(

userId int null,

City nvarchar(100) null,

Region nvarchar(100) null,

Country nvarchar(100) null

) WITH (

CLUSTERED COLUMNSTORE INDEX,

DISTRIBUTION = REPLICATE

);

GO

1. In **SQL Server Management Studio**, click on **Execute**.
2. In **SQL Server Management Studio**, in Object Explorer, right click **DWDB** under **sqlservicexx.database.windows.net** and click on **New Query**.
3. Create a table named **dbo.Products** with a **clustered columnstore** index with a distribution of **round robin** with the following columns:

| **column name** | **data type** | **Nullability** |
| --- | --- | --- |
| ProductId | int | NULL |
| EnglishProductName | nvarchar(100) | NULL |
| Color | nvarchar(100) | NULL |
| StandardCost | int | NULL |
| ListPrice | int | NULL |
| Size | nvarchar(100) | NULL |
| Weight | int | NULL |
| DaysToManufacture | int | NULL |
| Class | nvarchar(100) | NULL |
| Style | nvarchar(100) | NULL |

You can use this script:

D

1. In **SQL Server Management Studio**, click on **Execute**.
2. In **SQL Server Management Studio**, in Object Explorer, right click **DWDB** under **sqlservicexx.database.windows.net** and click on **New Query**.
3. Create a table named **dbo.FactSales** with a **clustered columnstore** index with a distribution of **Hash** on the **SalesUnit** with the following columns:

| **column name** | **data type** | **Nullability** |
| --- | --- | --- |
| DateId | int | NULL |
| ProductId | int | NULL |
| UserId | int | NULL |
| UserPreferenceId | int | NULL |
| SalesUnit | int | NULL |

You can use this script:

CREATE TABLE dbo.FactSales(

DateId int null,

ProductId int null,

UserId int null,

UserPreferenceId int null,

SalesUnit int null

) WITH (

CLUSTERED COLUMNSTORE INDEX,

DISTRIBUTION = HASH(SalesUnit)

);

GO

1. In **SQL Server Management Studio**, click on **Execute**.

**Result**: After you completed this exercise, you have installed SQL Server Management Studio to create a data warehouse named DWDB and three tables named Users, Products and FactSales.

## Exercise 4: Using PolyBase to Load Data into Azure SQL Data Warehouse

Estimated Time: 10 minutes

Individual exercise

The main tasks for this exercise are as follows:

1. Collect Azure Blob container and key details
2. Create a dbo.Dates table using PolyBase from Azure Blob
3. Create a dbo.Preferences table using PolyBase from Azure Databricks

### Task 1: Collect Azure Blob account name and key details

1. In the Azure portal, click on **Resource groups** and then click on **awrgstudxx**, and then click on **awsastudxx** where xx are the initials of your name.
2. In the **awsastudxx** screen, click **Access keys**. Click on the icon next to the **Storage account name** and paste it into Notepad.
3. In the **awsastudxx - Access keys** screen, under **key1**, Click on the icon next to the **Key** and paste it into Notepad.

### Task 2: Create a dbo.Dates table using PolyBase from Azure Blob

1. In **SQL Server Management Studio**, in Object Explorer, right click **sqlservicexx.database.windows.net** and click on **New Query**.
2. Create a **master key** against the **DWDB** database.

CREATE MASTER KEY;

1. Create a database scoped credential named **AzureStorageCredential** with the following details:

IDENTITY: **MOCID**

SECRET: **The access key of your storage account**

CREATE DATABASE SCOPED CREDENTIAL AzureStorageCredential

WITH

IDENTITY = 'MOCID',

SECRET = 'Your storage account key'

1. In **SQL Server Management Studio**, highlight both statements and then click on **Execute**.
2. In **SQL Server Management Studio**, in the Query window, type in code that will create an external data source named **AzureStorage** for the Blob storage account and data container created in with a type of **HADOOP** that makes use of the \*\***AzureStorageCredential**

CREATE EXTERNAL DATA SOURCE AzureStorage

WITH (

TYPE = HADOOP,

LOCATION = 'wasbs://data@awsastudxx.blob.core.windows.net',

CREDENTIAL = AzureStorageCredential

);

1. In **SQL Server Management Studio**, in the Query window, type in code that will create an external file format named **TextFile** with a formattype of **DelimitedText** and a filed terminator of **comma**.

CREATE EXTERNAL FILE FORMAT TextFile

WITH (

FORMAT\_TYPE = DelimitedText,

FORMAT\_OPTIONS (FIELD\_TERMINATOR = ',')

);

1. In **SQL Server Management Studio**, highlight the statement and then click on **Execute**.
2. In **SQL Server Management Studio**, in the Query window, type in code that will create an external table named **dbo.DimDate2External** with the **location** as the root file, the Data source as **AzureStorage**, the File\_format of **TextFile** with the following columns:

| column name | data type | Nullability |
| --- | --- | --- |
| Date | datetime2(3) | NULL |
| DateKey | decimal(38, 0) | NULL |
| MonthKey | decimal(38, 0) | NULL |
| Month | nvarchar(100) | NULL |
| Quarter | nvarchar(100) | NULL |
| Year | decimal(38, 0) | NULL |
| Year-Quarter | nvarchar(100) | NULL |
| Year-Month | nvarchar(100) | NULL |
| Year-MonthKey | nvarchar(100) | NULL |
| WeekDayKey | decimal(38, 0) | NULL |
| WeekDay | nvarchar(100) | NULL |
| Day Of Month | decimal(38, 0) | NULL |

You can use this script:

CREATE EXTERNAL TABLE dbo.DimDate2External (

[Date] datetime2(3) NULL,

[DateKey] decimal(38, 0) NULL,

[MonthKey] decimal(38, 0) NULL,

[Month] nvarchar(100) NULL,

[Quarter] nvarchar(100) NULL,

[Year] decimal(38, 0) NULL,

[Year-Quarter] nvarchar(100) NULL,

[Year-Month] nvarchar(100) NULL,

[Year-MonthKey] nvarchar(100) NULL,

[WeekDayKey] decimal(38, 0) NULL,

[WeekDay] nvarchar(100) NULL,

[Day Of Month] decimal(38, 0) NULL

)

WITH (

LOCATION='/',

DATA\_SOURCE=AzureStorage,

FILE\_FORMAT=TextFile

);

1. In **SQL Server Management Studio**, highlight the statement and then click on **Execute**.
2. Test that the table is created by running a select statement against it

SELECT \* FROM dbo.DimDate2External;

1. In **SQL Server Management Studio**, in the Query window, type in a **CTAS** statement that creates a table named **dbo.Dates** with a **columnstore** index and a **distribution** of **round robin** that loads data from the **dbo.DimDate2External** table.

CREATE TABLE dbo.Dates

WITH

(

CLUSTERED COLUMNSTORE INDEX,

DISTRIBUTION = ROUND\_ROBIN

)

AS

SELECT \* FROM [dbo].[DimDate2External];

1. In **SQL Server Management Studio**, highlight the statement and then click on **Execute**.
2. In **SQL Server Management Studio**, in the Query window, type in a query that creates statistics on the **DateKey**, **Quarter** and **Month** column.

CREATE STATISTICS [DateKey] on [Dates] ([DateKey]);

CREATE STATISTICS [Quarter] on [Dates] ([Quarter]);

CREATE STATISTICS [Month] on [Dates] ([Month]);

1. Test that the table is created by running a select statement against it

SELECT \* FROM dbo.Dates;

### Task 3: Create a dbo.Preferences table using PolyBase from Azure Databricks

1. Return back to the Azure Portal and access the Azure Databricks WorkSpace. Click on Clusters on the left. Click on your cluster name **awdbclstudxx**, and click on **Start Cluster**. A message appears confirming if you would like to start your **awdbclstudxx** cluster. Click **Confirm**

**Note**: This will start the cluster and will take approximately 5 minutes to start up. Continue with the next steps

1. In the Azure Databricks blade on the left of Microsoft Edge, click on Under **Workspace**, click on **My Notebook**.
2. In the Notebook, in the cell under **Cmd 2**, click on the **Run** icon and click on **Run Cell**.

**Note** A message will be returned at the bottom of the cell that states "Command took 0.0X seconds -- by person at date, time PM on awdbclstudxx"

1. In the Notebook, in the cell under **Cmd 3**, click on the **Run** icon and click on **Run Cell**.

**Note** A message will be returned at the bottom of the cell that states "Command took 0.0X seconds -- by person at date, time PM on awdbclstudxx"

1. In the Notebook, in the cell under **Cmd 4**, click on the **Run** icon and click on **Run Cell**.

**Note** A message will be returned at the bottom of the cell that states "Command took 0.0X seconds -- by person at date, time PM on awdbclstudxx"

1. In the Notebook, in the cell under **Cmd 5**, click on the **Run** icon and click on **Run Cell**.

**Note** A message will be returned at the bottom of the cell that states "Command took 0.0X seconds -- by person at date, time PM on awdbclstudxx"

1. In the Notebook, in the cell under **Cmd 6**, click on the **Run** icon and click on **Run Cell**.

**Note** A message will be returned at the bottom of the cell that states "Command took 0.0X seconds -- by person at date, time PM on awdbclstudxx"

1. In the Notebook, hover your mouse at the bottom of cell **Cmd 6**, and click on the **+** icon. A new cell will appear named **Cmd 7**.
2. In the Notebook, in the cell **Cmd 7**, copy the following code and paste it into the cell to provide the configuration to access the Azure Storage account from Azure Databricks. replace the blob-storage-account-name, blob-container-name and account-key with the information you saved earlier:

val blobStorage = "<blob-storage-account-name>.blob.core.windows.net"

val blobContainer = "<blob-container-name>"

val blobAccessKey = "<access-key>"

1. In the Notebook, in the cell under **Cmd 7**, click on the **Run** icon and click on **Run Cell**.

**Note** A message will be returned at the bottom of the cell that states "Command took 0.0X seconds -- by person at date, time PM on awdbclstudxx"

1. In the Notebook, hover your mouse at the bottom of cell **Cmd 7**, and click on the **+** icon. A new cell will appear named **Cmd8**.
2. In the Notebook, in the cell **Cmd 8**, copy the following code and paste it into the cell to specify a temporary folder to use while moving data between Azure Databricks and Azure SQL Data Warehouse:

val tempDir = "wasbs://" + blobContainer + "@" + blobStorage +"/tempDirs"

1. In the Notebook, in the cell under **Cmd 8**, click on the **Run** icon and click on **Run Cell**.

**Note** A message will be returned at the bottom of the cell that states "Command took 0.0X seconds -- by person at date, time PM on awdbclstudxx"

1. In the Notebook, hover your mouse at the bottom of cell **Cmd 8**, and click on the **+** icon. A new cell will appear named **Cmd 9**.
2. In the Notebook, in the cell **Cmd 9**, copy the following code and paste it into the cell to store Azure Blob storage access keys in the configuration. This action ensures that you don't have to keep the access key in the notebook in plain text.

val acntInfo = "fs.azure.account.key."+ blobStorage

sc.hadoopConfiguration.set(acntInfo, blobAccessKey)

1. In the Notebook, in the cell under **Cmd 9**, click on the **Run** icon and click on **Run Cell**.

**Note** A message will be returned at the bottom of the cell that states "Command took 0.0X seconds -- by person at date, time PM on awdbclstudxx"

1. In the Notebook, hover your mouse at the bottom of cell **Cmd 9**, and click on the **+** icon. A new cell will appear named **Cmd 10**.
2. In the Notebook, in the cell **Cmd 10**, copy the following code and paste it into the cell to provide the values to connect to the Azure SQL Data Warehouse instance, where **xx** are your initials.

//SQL Data Warehouse related settings

val dwDatabase = "DWDB"

val dwServer = "sqlservicexx.database.windows.net"

val dwUser = "xxsqladmin"

val dwPass = "Pa55w.rd"

val dwJdbcPort = "1433"

val dwJdbcExtraOptions = "encrypt=true;trustServerCertificate=true;hostNameInCertificate=\*.database.windows.net;loginTimeout=30;"

val sqlDwUrl = "jdbc:sqlserver://" + dwServer + ":" + dwJdbcPort + ";database=" + dwDatabase + ";user=" + dwUser+";password=" + dwPass + ";$dwJdbcExtraOptions"

val sqlDwUrlSmall = "jdbc:sqlserver://" + dwServer + ":" + dwJdbcPort + ";database=" + dwDatabase + ";user=" + dwUser+";password=" + dwPass

1. In the Notebook, in the cell under **Cmd 10**, click on the **Run** icon and click on **Run Cell**

**Note** A message will be returned at the bottom of the cell that states "Command took 0.0X seconds -- by person at date, time PM on awdbclstudxx"

1. In the Notebook, hover your mouse at the bottom of cell **Cmd 10**, and click on the **+** icon. A new cell will appear named **Cmd 11**.
2. In the Notebook, in the cell **Cmd 11**, copy the following code and paste it into the cello load the transformed dataframe, renamedColumnsDF, as a table in a SQL data warehouse. This snippet creates a table called Preferences in the SQL database.

spark.conf.set(

"spark.sql.parquet.writeLegacyFormat",

"true")

renamedColumnsDF.write

.format("com.databricks.spark.sqldw")

.option("url", sqlDwUrlSmall)

.option("dbtable", "Preferences")

.option( "forward\_spark\_azure\_storage\_credentials","True")

.option("tempdir", tempDir)

.mode("overwrite")

.save()

1. In the Notebook, in the cell under **Cmd 11**, click on the **Run** icon and click on **Run Cell**

**Note** A message will be returned at the bottom of the cell that states "Command took 0.0X seconds -- by person at date, time PM on awdbclstudxx"

1. Return to SQL Server Management Studio, in **Object Explorer** expand the database **DWDB**, expand **Tables** to confirm that the table **dbo.Preferences** has been created

**Result**: After you completed this exercise, you have used PolyBase to load data into a table named Dates from an Azure Blob Store. You have also populated a table named Preferences from Azure Databricks