Big Data Orientation

Lab 2 – Working with a Relational Database in Microsoft Azure

Overview

Azure SQL Database is a cloud service based on the Microsoft SQL Server relational database management system (RDBMS). Application developers can use Azure SQL Database as a relational store for application data, which can be used in big data solutions. In addition to Azure SQL Database, Azure includes a data warehouse service named Azure SQL Data Warehouse, which shares the same core database engine as Azure SQL Database but is optimized for large data workloads, and often provides an analytical data store into which big data processing solutions load the processed data for analysis and reporting.

In this lab, you will provision and work with Azure SQL Database. The tasks you will perform in this exercise can also be performed with Azure SQL Data Warehouse.

Before You Start

To complete this lab, you will need the following:

- A web browser
- A Windows, Linux, or Mac OS X computer

Important: Before you can perform the lab exercises, you must complete the previous lab in this course.

Exercise 1: Working with Azure SQL Database

In this exercise, you will provision a sample database in Azure SQL database, and use Transact-SQL to query the data it contains.

Provision Azure SQL Database

To get started, you must provision Azure SQL Database.

1. In the Microsoft Azure portal, in the menu, click **New**. Then in the **Databases** menu, click **SQL Database**.
2. In the **SQL Database** blade, enter the following settings, and then click **Create**:
    - **Name**: AdventureWorksLT
    - **Subscription**: *Select your Azure subscription*
    - **Resource Group**: *Select the resource group you created previously*

- **Select source**: Sample (AdventureWorksLT)
- **Server:** *Create a new server with the following settings*:
- **Server name**: *Enter a unique name (and make a note of it!)*
- **Server admin login:** *Enter a user name of your choice (and make a note of it!)*
- **Password:** *Enter and confirm a strong password (and make a note of it!)*
- **Location:** *Select any available region*
- **Allow azure services to access server**: Selected
- **Want to use SQL elastic pool?** Not now.
- **Pricing tier:** Basic
- **Collation**: SQL_Latin1_General_CP1_CI_AS
- **Pin to dashboard:** Unselected

3. In the Azure portal, view **Notifications** to verify that deployment has started. Then wait for the SQL database to be deployed (this can take a few minutes.)
4. After the database has been created, browse to your Azure SQL server (not the AdventureWorksLT database) and under **Settings**, click **Properties**.
5. Note the fully qualified name of your server (which should take the form *server*.database.windows.net, where *server* is the server name you specified earlier) and the server admin user name (which should be the login you specified earlier).

Query a Table

A relational database contains tables, each of which contains data. Tables are organized into namespaces called *schemas* – in the case of the **AdventureWorksLT** sample database, most of the tables are defined within a schema named **SalesLT**.

You can query tables using Transact-SQL to retrieve the data they contain.

1. Click **All Resources**, and then click the **AdventureWorksLT** database.
2. On the **AdventureWorksLT** blade, select the **Query Editor** option in the navigation pane. This opens the web-based query editor interface for your Azure SQL Database. Query editor replaces the Data Explorer option, and at the time of this writing is in **Preview**.
3. In the toolbar for the query editor, click **Login**, and then log into your database using SQL Server authentication and entering the login name and password you specified when provisioning the Azure SQL Database server.
4. In the query editor, enter the following Transact-SQL query to retrieve the contents of the **SalesLT.Product** table in the **AdventureWorksLT** database:

```
SELECT * FROM SalesLT.Product;
```

6. Click **Run**, and review the results.

Exercise 2: Loading Data into a Database

In this exercise, you will create a table in the sample database you created previously, and then use Azure Data Factory to copy data from a file in Azure Storage into the new table.

Create a Table

The sample database contains many tables, and you can add your own by using the Transact-SQL CREATE TABLE statement.

1. In the Query pane, replace the existing SELECT statement with the following code:

```
CREATE TABLE SalesLT.ProductReview
( ProductReviewID INTEGER PRIMARY KEY,
  ProductID INTEGER REFERENCES SalesLT.Product(ProductID),
  ReviewerName NVARCHAR(25),
  ReviewDate DATETIME,
  EmailAddress NVARCHAR(50),
  Rating INTEGER,
  Comments NTEXT );
```

2. Click **Run**, and verify that the query succeeds.
3. Replace the CREATE TABLE statement with the following SELECT statement:

```
SELECT * FROM SalesLT.ProductReview;
```

4. Click **Run**, and verify that the query succeeds but returns 0 rows.
5. Close the query editor without saving any changes.

Load Data from a File into the Table

A common task in a big data solution is to transfer data from one store to another. In this case, you will use the Copy Data wizard in the Azure Data Factory service to load the product review data from a text file in Azure Storage into the table you created in Azure SQL Database.

1. In the Microsoft Azure portal, in the menu, click **New**. Then in the **Analytics** menu, click **Data Factory**. Data Factory can also be found within the **Integration** menu.
2. In the **New data factory** blade, enter the following settings, and then click **Create**:
    - **Name**: *Enter a unique name (and make a note of it!)*
    - **Subscription**: *Select your Azure subscription*
    - **Resource Group**: *Select the resource group you created previously*
    - **Version**: 2
    - **Location**: *Select the same region as the Azure SQL Database and server created earlier.*
    - **Pin to dashboard:** Unselected
3. View **Notifications** to verify that deployment has started. Then wait for the data factory to be deployed (this can take a few minutes.)
4. Click **All Resources**, and then click your data factory, and click the **Author & Monitor** tile. This opens a new tab in your browser.
5. On the new browser tab, click the **Copy Data** icon in the **Let's Get Started** section.
6. On the **Properties** page of the Copy Data wizard, enter the following details and then click **Next**:
    - **Task name**: Load Reviews
    - **Task description**: Load review data into Azure SQL Database
    - **Task cadence (or) Task schedule**: Run once now
7. On the **Source data store** page, notice a number of tabs, such as **All**, **Azure**, **Database**, **File,** and more.
8. on the **Connect to a Data Store** tab, select the **Azure**, then click **+ Create new connection**.
9. In the **New Linked Service** dialog, select **Azure Blog Storage**, and click **Continue**.
10. Configure the **Azure Blog Storage Linked Service** as follows, then click **Finish**:
    - **Name**: BlobStorage_LS
    - **Description**: Copying review data to Azure SQL DB
    - **Connect via integration runtime**: AutoResolveIntegrationRuntime

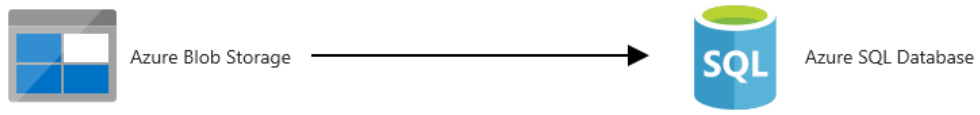
- **Authentication method**: Use account key w / Connection String
- **Account selection method**: From Azure subscription
- **Azure subscription**: *Select the appropriate Azure subscription*
- **Storage account name**: *Select the storage account name created in the first lab.*

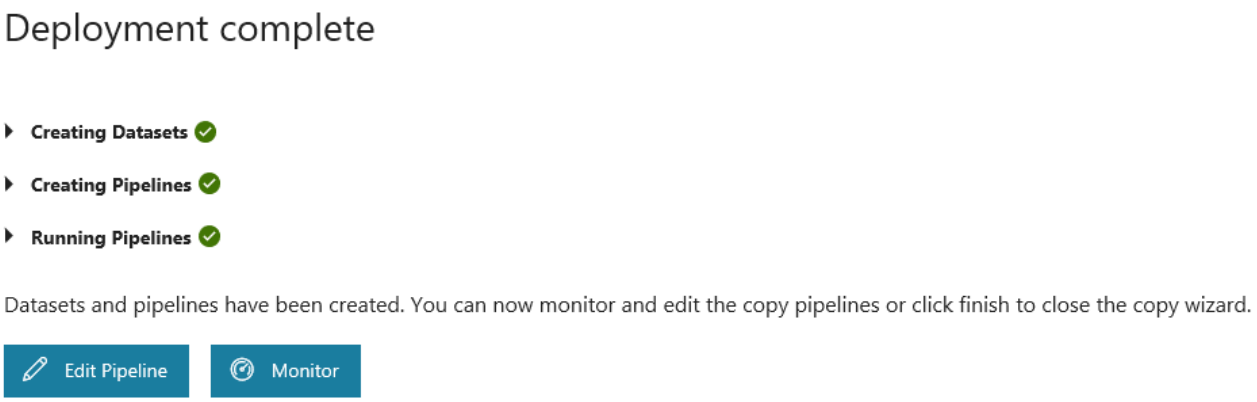
11. On the **Source Data Store** page, click **Next**.
12. On the **Choose the inpt file or folder** page, click **Browse**.
13. Double click on the **bigdata** folder you created in the previous lab, then double click on the **reviews.txt** file to select the file (or select the **reviews.txt** file and click **Choose**), then click **Next**.
14. On the **File format settings** page, wait a few seconds for the data to be read, and then verify the following details, ensuring that the rows of data in the **Preview** section match the table below, and click **Next**:
    - **File format**: text format
    - **Column delimiter**: Tab (\t)
    - **Row delimiter**: Carriage return and line feed (\r\n)
    - **Skip line count**: 0
    - **Column names in first data row**: Selected
15. Select the Advanced option, and configure the following:
    - **Treat empty column value as null**: Selected
16. Click **Next**.
17. On the **Destination data store** page, select the **Azure** option, then click **+ Create new connection.**
18. In the New Linked Service dialog, select **Azure SQL Database**, then click **Continue**.
19. Configure the **Azure SQL database Linked Service** as follows, then click **Finish**:
    - **Name**: SQLDatabase_LS
    - **Description**: Linked service to Azure SQL Database AdventureWorksLT
    - **Connect via Integration runtime**: AutoResolveIntegrationRuntime / Connection String
    - **Account selection method**: From Azure Subscription
    - **Azure subscription**: *Select your subscription*
    - **Server name**: *Select your Azure SQL server*
    - **Database name**: AdventureWorksLT
    - **Authentication type**:
    - **User name**: *The server admin login name you specified when creating the database*
    - **Password**: *The password for your Azure SQL server admin login*
20. On the **Destination data store** page, click **Next**.
21. On the **Table mapping** page, in the **Destination** list, select **[SalesLT].[ProductReview]** and click **Next**.
22. On the **Schema mapping** page, ensure that the following settings are selected, and click **Next**:

Blob path: adf-data/data/	[SalesLT].[ProductReview]	Include this column
ProductReviewID (Int64)	ProductReviewID (Int32)	
ProductID (Int64)	ProductID (Int32)	
ReviewerName (String)	ReviewerName (String)	
ReviewDate (DateTime)	ReviewDate (DateTime)	
EmailAddress (String)	EmailAddress (String)	
Rating (Int64)	Rating (Int32)	

Comments (String)	Comments (String)	

23. Leave the **Azure SQL Database sync properties** values as they are configured.
24. On the **settings** page, expand **Advanced settings** to review the default values. Then click **Next**.
25. On the **Summary** page, review the pipeline as configured, then click **Next**.
26. On the **Deploying** page, wait for the deployment to complete. Once completed, the Copy Data wizard will display the following:





27. Click **Finish**.

Verify that the Data Has Been Copied

The Copy Data wizard should have created a pipeline, and run it to copy the transactions data from your blob store to your Azure SQL Database. Azure Data Factory may take several minutes to schedule the pipeline to run, so wait for around 5 minutes.

1. Click **All Resources**, and then click the **AdventureWorksLT** database.
2. On the **AdventureWorksLT** blade, view the **Data Explorer** page. This opens the web-based query interface for your Azure SQL Database.
3. In the toolbar for the query editor, click **Login**, and then log into your database using SQL Server authentication and entering the login name and password you specified when provisioning the Azure SQL Database server.
4. In the query editor, enter the following Transact-SQL query to retrieve the contents of the **SalesLT.ProductReview** table in the **AdventureWorksLT** database:

```
SELECT * FROM SalesLT.ProductReview;
```

5. Click **Run**, and review the results.