Databases in Azure – Practical Exercises

Overview

This course includes optional exercises where you can try out the techniques demonstrated in the course for yourself. This guide lists the steps for the individual practical exercises.

See the Overview page under Practical Exercises in your course for information about getting started.

Azure Subscription Setup

If you already have a Microsoft Azure subscription, you can skip this section. Otherwise, follow these steps to create a free trial subscription. You will need to provide a valid credit card number for verification, but you will not be charged for Azure services – for more information, see the frequently asked questions on the Azure sign-up page.

1. If you already have a Microsoft account that has not already been used to sign up for a free Azure trial subscription, you're ready to get started. If not, don't worry, just create a new Microsoft account.
2. After you've created a Microsoft account, create your free Microsoft Azure account. You'll need to sign-in with your Microsoft account if you're not already signed in. Then you'll need to:
    - Enter your cellphone number and have Microsoft send you a text message to verify your identity.
    - Enter the code you have been sent to verify it.
    - Provide valid payment details. This is required for verification purposes only – your credit card won't be charged for any services you use during the trial period, and the account is automatically deactivated at the end of the trial period unless you explicitly decide to keep it active.

Lab VM Setup

To ensure consistent experience and eliminate the need for installing SQL Server Management Studio, this course will rely on a lab VM implemented as an Azure VM hosting SQL Server 2016 Express Edition deployed by using an Azure Gallery image. The choice of the Express Edition will also minimize charges associated with running the lab VM in Azure. However, you will notice a definite difference in performance from what you would experience working directly in the

Azure portal to create and work with individual databases. You might want to choose the Azure region with lower VM prices (in general, US prices are lower).
In addition, you should consider stopping the VM whenever you are not actively using it. Use the Azure portal to stop the VM (rather than shutting it down from within the operating system) and ensure that the VM is listed with the **Stopped (Deallocated)** state. The **Stopped** state indicates that the VM is a subject to compute charges, even if its operating system is no longer running.

Create the Lab VM

To deploy the lab VM, follow these steps:

1. Start an Internet browser and browse to the Azure portal at https://portal.azure.com. When prompted, sign in with the credentials that have the Owner role permissions to the subscription.
2. Click **New** in the hub menu on the left-hand side of the portal.
3. On the **New** blade, in the search text box, type **SQL Server 2016 Express**.
4. In the list of results, choose the latest image containing SQL Server 2016 Express (SQL Server 2016 SP1 Express on Windows Server 2016 at the time of authoring of this content).
5. Ensure that the deployment model is set to **Resource Manager**.
6. Configure the following Basic settings for the lab VM:
    - Name: **LabVM**
    - VM disk type: **HDD**
    - User name: **Student**
    - Password: **Pa55w.rd1234**
    - Subscription: the name of your Azure subscription
    - Resource group: create a new resource group named **DBLabRG**
    - Location: choose the Azure region close to your physical location
7. Use **A1 Basic** VM size. If you find performance to be insufficient, change the size to **A1 Standard** of **F1 Standard**.
8. Accept most of the default options on the Settings blade. Disable the Boot diagnostics.
9. On the SQL Server settings blade, specify the following settings:
    - SQL connectivity: **Local (inside VM only)**
    - SQL Authentication: **Disable**
    - Storage configuration: **General**
    - Automated patching: **Disabled**
    - Automated backup: **Disabled**
    - Azure Key Vault integration: **Disabled**
    - R Services (Advanced analytics): **Disabled**
10. Deploy the VM.

Note: Deployment may take about 20 minutes. Wait until it completes before proceeding to the next step.

As part of configuration of the lab VM, you will stop and disable the SQL Server engine (to eliminate its impact on the VM performance) and install Azure PowerShell module.

Configure the lab VM

To configure the lab VM, follow these steps:

1. Connect to the LabVM via Remote Desktop and sign in with the administrative credentials you specified.
2. On the lab VM, in the Server Manager window, click **Tools**, and in the **Tools** menu, click **Services.**
3. In the **Services** console, stop and disable all SQL Server services, starting with SQL Server (MSSQLSERVER).
4. In the Server Manager window, switch to the Local Server page and turn off **IE Enhanced Security Configuration** for Administrators**.**
5. Within the Remote Desktop session, start an Internet browser and browse to https://azure.microsoft.com.
6. On the Microsoft Azure page, click **Resources** and then click **Downloads**.
7. From the Downloads page, download and install Azure PowerShell module.

Deploying and connecting to Azure SQL Databases

Exercise 1 - Deploying Azure SQL Databases

In first exercise, you will deploy an Azure SQL database based on the AdventureWorksLT sample database by using the Azure portal and deploy a blank database by using Azure PowerShell.

Deploying Azure SQL Database by using the Azure portal

1. From the lab VM, navigate to the Azure Portal at https://portal.azure.com and sign in.
2. From the Azure portal, create a new SQL database with the following parameters:
   - Database name: **adventureworks**
   - Subscription: the name of your Azure subscription
   - Resource group: **SQLDBRG**
   - Select source: **Sample**
   - Select sample: **AdventureWorksLT [V12]**
   - Server – create a new server
     - Server name: any valid unique name
     - Server admin: **Student**
     - Password: **Pa55w.rd1234**
     - Location: the same location as the lab VM
     - Create V12 server (Latest update): **Yes**
     - Allow azure services to access server: **Enabled**
   - Want to use SQL elastic pool: **Not now**
   - Pricing tier: **Basic**
   - Pin to dashboard: **enabled**

Deploying Azure SQL Database by using the Azure PowerShell

1. From the lab VM, start Windows PowerShell.
2. In the Windows PowerShell window, run

```
Add-AzureRmAccount
```

3. When prompted, type N to disable data collection and then provide the credentials of the account you use to manage your Azure subscription.
4. Next, in the Windows PowerShell window, run
   ```
   Get-AzureRmLocation | Select-Object Location
   ```
5. Identify the secondary locations other than the one you used to deploy the SQL server in earlier in this exercise.
6. Next, in the Windows PowerShell window, run
   ```
   $pass = ConvertTo-SecureString 'Pa55w.rd1234' -AsPlainText -Force
   $cred = New-Object
   System.Management.Automation.PSCredential('Student',$pass)
   server = New-AzureRmSqlServer -ServerName <unique_name> -
   SqlAdministratorCredentials $cred -Location <secondary_location> -
   ServerVersion '12.0' -ResourceGroupName 'SQLDBRG'
   ```

   Note: If you get prompted for credentials at this point, type in Student as the user name and Pa55w.rd1234 as the password.

   where <unique_name> represents a unique server name and <secondary_location> represents the secondary location you chose in step 5 of this exercise.

7. Next, in the Windows PowerShell window, run
   ```
   New-AzureRmSqlDatabase -ResourceGroupName $server.ResourceGroupName
   -ServerName $server.ServerName -DatabaseName 'newdb'
   ```

Exercise 2 – Configuring server firewall

In this exercise, you will use the Azure portal to configure server firewall to allow SQL Database connectivity from your lab VM. You will then examine the rules by using Azure PowerShell.

Note: If you stop and de-allocate VM afterwards, you might need to update the firewall rules to account for the change of IP address once you bring the VM online again.

Configure server firewall rules by using the Azure portal

1. On the lab VM, from the Azure portal, navigate to the **adventureworks** database blade.
2. On the **adventureworks** database blade, click **Set server firewall**
3. On the Firewall settings blade, identify the public IP address corresponding to your lab VM.
4. On the Firewall settings blade, create a new rule with the following settings:
   - RULE NAME: **AllowVM**
   - START IP: ***XXX.XXX*.0.0**
   - END IP: ***XXX.XXX*.255.255**

   Where ***XXX.XXX*** represents the first two octets of the client IP address.

Identify server firewall rules by using Azure PowerShell

1. From the lab VM, start Windows PowerShell (if it is not already running).
2. If you are starting a new Windows PowerShell session (otherwise, your authentication token you obtained in the previous exercise should still be valid and you can skip this step), then in the Windows PowerShell window, run
   ```
   Add-AzureRmAccount
   ```
   When prompted, provide the credentials of the account you use to manage your Azure subscription.
3. Next, in the Windows PowerShell window, run
   ```
   Get-AzureRmSqlServerFirewallRule -ServerName <your_server_name> –
   ResourceGroupName SQLDBRG
   ```

   where <server_name> represents the name of the server hosting the adventureworks database.

5. Note that there are two rules:
    - The first one (0.0.0.0) corresponds to the **Allow azure services to access server** setting you enabled when creating the server by using the Azure portal
    - The second one represents the rule you created earlier in this exercise

Exercise 3 – Connecting to Azure SQL database

In this exercise, you will connect to the Azure SQL Database by using the SQL Server Management Studio from the lab VM and run a sample query.

Note: If you stopped and de-allocated VM after you configured the server firewall rules, you might need to update the firewall rules to account for the change of IP address once you bring the VM online again, before running this exercise.

Connect to the Azure SQL database server with SQL Server Authentication by using SQL Server Management Studio

1. From the lab VM, start SQL Server 2016 Management Studio (note that the initial load of SQL Server Management Studio takes a few minutes), and in the **Connect to Server** dialog box, specify the following settings, and then click **Connect**:
   - Server type: **Database Engine**
   - Server name: **server_name.database.windows.net** (replace *server_name* with the unique name you specified when creating the first SQL Database server)
   - Authentication: **SQL Server Authentication**
   - Login: **Student**
   - Password: **Pa55w.rd1234**
2. In SQL Server Management Studio, in Object Explorer, under the server name, expand **Databases**, and then verify that the **adventureworks** database is listed.
3. Expand the **aventureworks** database and then click **Views**.
4. Expand Views, right click **SalesLT.vProductandDescription**. In the right-click menu, select **Script View as -> SELECT To -> New Query Editor Window**. This will open a new query editor window with automatically generated T-SQL statement that lists all products and their description.
5. Click **Execute** in the toolbar and note the results in the Results pane.
6. Right-click the server node in the Object Explorer window and disconnect from the serer but leave the SQL Server Management Studio running.

Configuring Azure AD authentication

Exercise 4 - Configuring Azure SQL server Azure AD authentication

In this exercise, you will configure Azure SQL server Azure AD authentication for the first Azure SQL Database server you deployed in the first module. You will start by creating an Azure AD user account that will become the Azure SQL server administrator login. You will also create an additional Azure AD user account that will become a contained user in the adventureworks database. Finally, you will configure the first of the two accounts to become the Azure SQL server admin account.

Note: In general, rather than using individual user accounts, you would typically use groups to streamline administration. In this exercise, we are using individual user accounts for simplicity sake.

Create Azure AD users

1. From the lab VM, navigate to the Azure Portal at https://portal.azure.com
2. Navigate to Azure Active Directory tenant associated with your subscription (this typically would be the tenant named **Default Directory**)

   Note: To complete these steps, you should be a Global Admin of the Azure AD tenant.

3. In the Azure Active Directory blade, click **Domain names** and note the name assigned by default to your Azure Active Directory (the name will have the **.onmicrosoft.com** suffix)
4. From the Azure Active Directory blade, create a new Azure AD user with the following settings:
   - Name: ADSQLAdmin
   - User name: ADSQLAdmin@<your_Azure_AD_domain_name>
   - Profile: Not configured
   - Properties: Default
   - Groups: 0 groups selected
   - Directory role: User
   - Password: click **Show Password** to view the auto-generated password

5. Start another Internet Explorer InPrivate Browsing session and navigate to the Azure Portal at https://portal.azure.com.
6. Sign in with the ADSQLAdmin account and, when prompted, change its password to **Pa55w.rd1234**

> **Note**: The two previous steps are necessary to change the expiring password.

7. Sign out from the Azure portal in the InPrivate Browsing session.
8. Repeat the steps 4 to 7 to create the new Azure AD user named ADSQLUser and change its password.

Configure the Azure AD administrator for Azure SQL server by using the Azure portal

1. In the Azure Portal, navigate to the SQL server hosting the **adventureworks** database.
2. From the blade of the server hosting the **adventureworks** database, navigate to its **Active Directory admin** blade.
3. From the **Active Directory admin** blade, set the admin account to the ADSQLAdmin user (including the domain name you identified earlier in this exercise). Make sure to save your changes.

> **Note**: Note that you can accomplish the same objective by running the **Set-AzureRmSqlServerActiveDirectoryAdministrator** cmdlet.

Exercise 5 - Configuring Azure SQL database Azure AD authentication

In this exercise, you will sign in to the Azure SQL server as the Azure AD-based administrative account by using SQL Server Management Studio, create an Azure SQL database contained user, and connect to that database by using the database contained user. You will also assign db_datareader role to the database contained user to allow reading of database objects.

Connect to the Azure SQL database server as the Azure AD-based administrative account by using SQL Server Management Studio

1. From the lab VM, start SQL Server 2016 Management Studio (if not started), and in the **Connect to Server** dialog box, specify the following settings, and then click **Connect**:
   - Server type: **Database Engine**
   - Server name: **server_name.database.windows.net** (replace *server_name* with the unique name you specified when creating the first SQL Database server)
   - Authentication: **Active Directory Password Authentication**
   - Sign in: the full user name of the ADSQLAdmin account (including the domain name you identified in the previous exercise of this module)
   - Password: **Pa55w.rd1234**
2. In SQL Server Management Studio, in Object Explorer, under the server name, expand **Databases**, and then verify that the **adventureworks** database is listed.
3. Select the **aventureworks** database and then click **Views**.
4. Right click **SalesLT.vProductandDescription**. In the right-click menu, select **Script View as -> SELECT To -> New Query Editor Window**. This will open a new query editor window with automatically generated T-SQL statement that lists all products and their description.
5. Click **Execute** in the toolbar and note the results in the Results pane.

Create a database contained user

1. In SQL Server Management Studio, in Object Explorer, under the server name, expand **Databases**, then select the **aventureworks** database and then click **New Query**.
2. In the new query window, type in the following T-SQL statement

```
CREATE User[ADSQLUser@<domain_name>]
FROM EXTERNAL PROVIDER
```

where <domain_name> represents the domain name you identified in the previous exercise of this module.

3. Execute the query and ensure it completed successfully.
4. Leave the Object Explorer window open and keep the connection to the server open.

Connect as a database contained user to an Azure SQL database

1. In the SQL Server 2016 Management Studio, specify the following settings and click **Connect**:
    - Server type: **Database Engine**
    - Server name: **server_name.database.windows.net** (replace *server_name* with the unique name you used earlier in this exercise)
    - Authentication: **Active Directory Password Authentication**
    - Sign in: the full user name of the ADSQLUser account (including the domain name you identified in the previous exercise of this module)
    - Password: **Pa55w.rd1234**
    - Options: in the **Connect to database** text box, type **adventureworks**

    Note: You must explicitly specify the database to connect to, because this is the only database that the database contained user can connect to (the connection would fail if the user tried to connect to the master database).

2. This will create another node in the Object Explorer window. Note that its label (inside parenthesis, following the server name) indicates that you are connected as the ADSQLUser account.
3. Expand the newly added node, under the server name, expand **Databases**, and then verify that the **adventureworks** database is listed.
4. Select the **aventureworks** database and then click **Views**.
5. Note that you do not see any views.

    At this point, the contained users does not have sufficient permissions to view database objects. In the next task, you will assign the db_datareader role to the user.

    Note: At this point, the contained users does not have sufficient permissions to view database objects. In the next task, you will assign the db_datareader role to the user.

Assign the db_datareader role to the database contained user.

1. In the Object Explorer window, right-click the server node representing the connection with the ADSQLAdmin account.
2. In the right-click menu, click **New Query**.
3. In the new query window, type the following:

```
USE adventureworks
GO
sp_addrolemember db_datareader, [ADSQLUser@<domain_name>]
```

where <domain_name> represents the domain name you identified in the previous exercise of this module.

4. Execute the query and ensure it completed successfully.
5. Switch back to the section of the Object Explorer window displaying the connection with the ADSQLUser account. Right-click the **Views** node and click **Refresh** in the right-click menu.
6. Note that now you can see all views
7. Right click **SalesLT.vProductandDescription**. In the right-click menu, select **Script View as -> SELECT To -> New Query Editor Window**. This will open a new query editor window with automatically generated T-SQL statement that lists all products and their description.
8. Click **Execute** in the toolbar and note the results in the Results pane.

Implementing Azure SQL database business continuity

Exercise 6 - Restoring an Azure SQL database

In this exercise, you will create a new database from a geo-replicated backup and restore a deleted database.

Create a new database from geo-replicated backup

1. From the lab VM, navigate to the Azure Portal at https://portal.azure.com and sign in.
2. From the Azure portal, create a new SQL database with the following parameters:
   - Database name: **newdb_copy**
   - Subscription: the name of your Azure subscription
   - Resource group: Use existing - **SQLDBRG**
   - Select source: **Backup**
   - Backup: **newdb**
   - Server: the first server you created in the first exercise of the first module
   - Want to use SQL elastic pool: **Not now**
   - Pricing tier: **Basic**
   - Pin to dashboard: **disabled**

Note: The restore might take some time. You do not have to wait for it to complete – you can proceed to the next task in this exercise.

Restore a deleted a database

1. In the Hub menu of the Azure Portal, navigate to the **newdb** Azure SQL database
2. From the **newdb** blade, delete the database. Confirm when prompted whether to delete the database permanently.
3. From the Azure portal, create a new SQL database with the following parameters:
   - Database name: **newdb**
   - Subscription: the name of your Azure subscription
   - Resource group: Use existing - **SQLDBRG**

- Select source: **Backup**
- Backup: **newdb**
- Server: the second server you created in the first exercise of the first module, where the newdb database was originally located
- Want to use SQL elastic pool: **Not now**
- Pricing tier: **Basic**
- Pin to dashboard: **disabled**

Note: The restore might take some time. You do not have to wait for the restore to complete - you can proceed to the next task in this exercise.

Note: Note that restoring the deleted database is equivalent to restoring a database from the geo-replicated backup with the same name and to the same server as the original one.

Configure geo-replication

1. In the Azure portal, navigate to the Geo-Replication blade of the adventureworks SQL database.
2. On the **adventureworks - Geo-Replication** blade, create a readable secondary replica with the following settings:
   - Region: use the region to which you deployed the second Azure SQL Database server
   - Database name: **adventureworks**
   - Pricing tier: **Basic**
   - Secondary type: **Readable**
   - Target server: the name of the second Azure SQL Database server you deployed
3. View the graphical representation of the geo-replication on the **Geo-Replication** blade.

Note: The initial geo-replication might take some time. You do not have to wait for it to complete - you can proceed to the next exercise.

Scaling Azure SQL database

Exercise 8 – Scaling Azure SQL Database vertically

In this exercise, you will scale vertically an Azure SQL database by using Azure PowerShell.

Change the pricing tier of an Azure SQL database by using Azure PowerShell

1. From the lab VM, start Windows PowerShell.
2. In the Windows PowerShell window, run
   ```
   Add-AzureRmAccount
   ```
3. When prompted, provide the credentials of the account you use to manage your Azure subscription.
4. Next, in the Windows PowerShell window, run
   ```
   Set-AzureRmSqlDatabase -ServerName <your_server_name> –
   ResourceGroupName SQLDBRG –Edition ‘Standard’ –
   RequestedServiceObjectiveName ‘S1’ –DatabaseName ‘newdb_copy’
   ```
   where <server_name> represents the name of the server hosting the newdb_copy database. This will take about a minute and will display the outcome of the scaling operation. You can use the Azure portal to verify that the pricing tier of the newdb_copy database has changed.

Note: You cannot change the edition of the database which is in the replication relationship, so this command would fail if you attempted to use it to resize the adventureworks database.

You could include the PowerShell cmdlet listed above in an Azure Automation runbook and invoke it (via a webhook) in response to an alert triggered based on a metric representing database utilization level – such as, for example, CPU percentage or DTU used.

5. To change the pricing tier of the database back to its original value, run
   ```
   Set-AzureRmSqlDatabase -ServerName <your_server_name> –
   ResourceGroupName SQLDBRG –Edition ‘Basic’ –DatabaseName
   ‘newdb_copy’
   ```

Note: Changing the pricing tier down to Basic might take some time. You do not have to wait for it to complete before you can proceed to the next exercise.

Exercise 9 – Creating an elastic database pool

In this exercise, you will create an Azure SQL Database elastic database pool.

Create an elastic database pool

1. In the Azure Portal, navigate to the SQL server hosting the secondary replica of the **adventureworks** database.
2. Click **New pool**
3. Create an elastic database pool blade with the following settings:
   - Name: ElasticPool-1
   - Pricing tier: Basic pool
   - Configure pool: 50 eDTU pool, 2 databases (you will need to manually add both adventureworks and newdb databases to the pool)
4. Note the estimated cost.

   In general, you would want to run databases for at least 30 days, to identify their usage patterns. If sufficient usage data is available, the platform will automatically provide recommendations for the optimal pool sizing and database pool assignment.

Note: You can include in the database pool Azure SQL databases which are in the replication relationship, including secondary replicas.

Deploying and working with DocumentDB

Exercise 10 - Deploying NoSQL (DocumentDB)

In this exercise, you will deploy a DocumentDB database by using the Azure portal. In the process, you will become familiar with the DocumentDB hierarchical resource model consisting of DocumentDB accounts, databases, database collections, and documents.

Creating a NoSQL (DocumentDB) account

1. From the lab VM, navigate to the Azure Portal at https://portal.azure.com and sign in.
2. From the New blade of the Azure portal, search for NoSQL (DocumentDB)
3. Create a new NoSQL (DocumentDB) account with the following parameters:
    - ID: a unique account identifier
    - NoSQL API: **DocumentDB**
    - Subscription: the name of your Azure subscription
    - Resource group: Create a new resource group named **DocumentDBRG**
    - Location: the same location as the lab VM

    An account serves as a container for one or more databases. There are DocumentDB settings, such as, for example, geo-replication, key-based security, or default consistency, which apply on the account level.

Creating a NoSQL (DocumentDB) database and a collection by using the Azure portal

1. From the blade of the newly created NoSQL (DocumentDB) database, create a new collection with the following settings:

    . Collection **Id: labCollection1**

    a. Pricing Tier: Standard

    b. Partitioning mode: Single Partition

    > **Note**: Single partition collections support up to 10GB and 10k RU/s throughput. Partitioned collections can support higher throughput, beyond this limit, and scale data and requests automatically across multiple partitions.

    c. Database: create a new database named **labDB**

Note: A DocumentDB database consists of one more collections. You can define a number of settings on per-collection level, including, for example, indexing policy or lifespan of documents (after which, documents in the corresponding collection would be automatically deleted).

Exercise 11 – Managing and querying JSON documents

In this exercise, you will upload sample JSON documents into a collection of the newly created DocumentDB database and run queries targeting their content.

Upload sample JSON documents to a DocumentDB database collection

1. From the lab VM, browse to https://github.com/Azure/azure-documentdb-dotnet/tree/master/samples/searchable-todo/data and download **items.zip** to the Downloads folder on the lab VM.
2. Extract the content of **items.zip**. The file contains a single folder named **items** containing 100 JSON-formatted files.

   Coincidentally, you can upload up to 100 JSON documents in a single step from the Azure portal. Each document must be less than 512 KB. If you need to upload a larger number of documents (or documents of a larger size), you should consider using the Data Migration Tool. For more information, refer to https://docs.microsoft.com/en-us/azure/documentdb/documentdb-import-data.

3. Open one of the files in Notepad and note that a file represents a to-do-task, including such entries as id, title, dueDate, description, isComplete, and tags. You will also find there the entries labeled _rid, _self, _etag, _attachments, and _ts (the properties with the underscore character are system generated and used internally by DocumentDB to distinguish between individual resources or provide such functionality as concurrency control).
4. In the Azure portal, on the DocumentDB account blade, click **Document Explorer**. On the Document Explorer blade, click **Upload**, and, from the **Upload Document** blade, upload the 100 JSON documents you extracted from the **items.zip** file.

Query content of a DocumentDB database collection

1. In the Azure portal, on the DocumentDB account blade, click **Query Explorer**.
2. On the **Query Explorer** blade, note the default, auto-generated query that targets the **labCollection1** in the **labDB** database:

   ```
   SELECT * FROM c
   ```

where c represents the target collection. Click **Run Query**.

Note: The syntax of DocumentDB queries resembles closely T-SQL syntax. However, that they are not equivalent. For example, DocumentDB does not currently implement several T-SQL features, such as wildcard-based searches.

3. Examine the results. Note that they include the listing of records matching those you identified when reviewing one of the JSON files in the previous task of this exercise. Scroll to the left and examine the content of the **Information** section on the Query Explorer blade. The information includes request charge (expressed in request units), round trips, number of individual records in the result set, as well as an indication whether there are any additional records besides these being currently displayed.
4. Modify the query so it takes the following format and re-run it:

```
SELECT * FROM c
WHERE c.id = ‘0’
```

5. This time, the query will return a single record with the value of “id” property matching the WHERE clause. Review the **Information** section again to identify the corresponding request charge and round trips.
6. Now, try rerunning this query but instead of using the id property, use title:

```
SELECT * FROM c
WHERE c.title = ‘Call Reta Murazik Sr.’
```

7. Next, run the following query to return every record (task) which include “One-time” as one of their its tags:

```
SELECT VALUE c
FROM root c JOIN word IN c.tags
WHERE word = “One-time”
```

Queries benefit from the default characteristics of DocumentDB which automatically handles all indexing, including each term that is part of the tag property (you can customize this behavior if needed). For more information, about DocumentDB queries, refer to https://docs.microsoft.com/en-us/azure/documentdb/documentdb-sql-query and https://azure.microsoft.com/en-us/blog/searching-for-text-with-documentdb/.

Deploying and working with Azure SQL Data Warehouse

Exercise 12 - Creating an Azure SQL Data Warehouse

In this exercise, you will create an Azure SQL Data Warehouse database by using the Azure portal. Rather than creating a blank database, you will automatically populate it with data from the sample AdventureWorksDW database.

Create an Azure SQL Data Warehouse by using the Azure portal

1. From the lab VM, navigate to the Azure Portal at https://portal.azure.com
2. Click New in the hub menu, then, on the **New** blade, search for **SQL Data Warehouse**. On the search results blade, click **SQL Data Warehouse** and create a new database with the following settings:
    - Database name: **LabSQLDW**
    - Subscription: the name of your Azure subscription
    - Resource group: create a new resource group named **SQLDWRG**
    - Select source: **Sample**
    - Select sample: **AdventureWorksDW**
    - Server: create a new server with the following settings:
        0. Server name: any valid unique name
        1. Server admin login: Student
        2. Password: Pa55w.rd1234
        3. Location: the same location as the lab VM
        4. Allow azure services to access server: Enabled
    - Collation: **Specified by sample**
    - Performance: 100 DWU
    - Pin to dashboard: Enable the checkbox

Note: Wait for the deployment to complete. This might take about 5 minutes.

Exercise 13 – Configuring the Azure SQL Data Warehouse database server-level firewall

In this exercise, you will use the Azure portal to configure SQL Data Warehouse database server-level firewall to allow direct connectivity from your lab VM.

Note: If you stop and de-allocate VM afterwards, you might need to update the firewall rules to account for the change of IP address once you bring the VM online again.

Configure server firewall rules by using the Azure portal

1. On the lab VM, from the Azure portal, navigate to the **LabSQLDW** database blade.
2. On the **LabSQLDW** database blade, click the **Server name** link. This will open the blade displaying the server properties.
3. On the server blade, click **Firewall**.
4. On the Firewall settings blade, identify the public IP address corresponding to your lab VM.
5. On the Firewall settings blade, create a new rule with the following settings:
    - RULE NAME: **AllowVM**
    - START IP: ***XXX.XXX*.0.0**
    - END IP: ***XXX.XXX*.255.255**

   Where ***XXX.XXX*** represents the first two octets of the client IP address.

   As you likely noticed, this procedure is identical to configuring server-level firewall for Azure SQL Database. Azure SQL Database and Azure SQL Data Warehouse databases exist within confines of a logical SQL Server. You can use the same server to host both types of databases.

Exercise 14 - Connecting to Azure SQL database by using SQL Server Data Tools in Visual Studio 2015

In this exercise, you will install SQL Server Data Tools in Visual Studio 2015 on the lab VM and use it connect to the Azure SQL Data Warehouse database.

Install SQL Server Data Tools in Visual Studio 2015

1. On the lab VM, from the Azure portal, navigate to the **LabSQLDW** database blade.
2. On the **LabSQLDW** database blade, click **Open in Visual Studio**. This will open the **Open in Visual Studio** blade.
3. In the new blade, click **Download SQL Server Data Tools**. This will open the **Download SQL Server Data Tools (SSDT)** Internet Explorer page
4. Click the **Download SQL Server Data Tools (16.5)** link.
5. Click the **Download SQL Server Data Tools** link.
6. When prompted whether to run or save **SSDTSetup.exe**, click **Run**. This will start the **Microsoft SQL Server Data Tools – Visual Studio 2015** wizard.
7. On the **Install tools for these SQL Server features** page, clear the checkboxes next to **SQL Server Analysis Services, SQL Server Reporting Services,** and **SQL Server Integration Services** and click **Next**.
8. Accept the licensing terms and complete the installation.

Note The installation might take about 10 minutes.

Connect to the Azure SQL database server as the Azure AD-based administrative account by using SQL Server Management Studio

1. Switch back in the Azure portal and, on the **Open in Visual Studio** blade, click **Open in Visual Studio**.
2. When prompted, click **Allow**. This will automatically open the SQL Server Data Tools with the Visual Studio 2015 integrated shell.
3. In the **Connect** dialog box, provide the password of the administrative account of the server hosting your Azure SQL Data Warehouse database (**Pa55w.rd1234**) and click **Connect**.

4. In the **SQL Server Object Explorer** window, expand the object hierarchy under the SQL Server node until you get to the **LabSQLDW** database. Note that this allows you to browse through and manage external resources (data sources, and file formats), tables, views, programmability features (stored procedures) and database level security. You can also manage server-level security.
5. Expand the **Views** node and note that it contains several views, including **dbo.AggregatedSales**. You will use this view to create reports in Power BI later in the next exercise of this module.

Exercise 15 - Exploring Azure SQL Data Warehouse by using Power BI

In this exercise, you will connect to the Azure SQL Data Warehouse database by using PowerBI and generate sample reports. To use PowerBI, you need to have a work or school account. To satisfy this requirement, you will create a user account in the Azure AD tenant associated with your Azure subscription and use it to activate a Power BI trial subscription associated with your Azure AD tenant.

Create an Azure AD user account

1. From the lab VM, navigate to the Azure Portal at https://portal.azure.com
2. Navigate to Azure Active Directory tenant associated with your subscription (this typically would be the tenant named **Default Directory**)
3. In the Azure Active Directory blade, click **Domain names** and note the name assigned by default to your Azure Active Directory (the name will have the **.onmicrosoft.com** suffix)
4. From the Azure Active Directory blade, create a new Azure AD user with the following settings:
    - Name: SQLDWAdmin
    - User name: SQLDWAdmin@<your_Azure_AD_domain_name>
    - Profile: Not configured
    - Properties: Default
    - Groups: 0 groups selected
    - Directory role: **Global Administrator**
    - Password: click **Show Password** to view the auto-generated password
5. Start another Internet Explorer InPrivate Browsing session and navigate to the Azure Portal at https://portal.azure.com.
6. Sign in with the SQLDWAdmin account and, when prompted, change its password to **Pa55w.rd1234**

    Note These two previous steps are necessary to change the expiring password.

7. Sign out from the Azure portal in the InPrivate Browsing session.

Activate a free Power BI trial subscription

1. From the lab VM, start an InPrivate Browsing session and navigate to the Azure Portal at https://portal.office.com/admin/default.aspx
2. When prompted to sign in, provide the credentials of the SQLDWAdmin account.
3. In the menu on the left hand side, navigate to **Billing** and click **Subscriptions**.
4. Click **Add subscriptions**.
5. Under Other plans, locate **Power BI (free)** tile, click the ellipsis label and click **Buy now**.
6. On the **Power BI (free)** page, click **Check out now**.
7. On the **Where will you be using this?** page, provide required information and click **Next**.
8. On the **How does this look?** page, click **Next**.
9. On the **How do you want to pay?** Select **Invoice (pay by check or wire transfer)** and click **Place order**.

Connect to the Azure SQL Data Warehouse database by using Power BI

1. On the lab VM, from the Azure portal, navigate to the **LabSQLDW** database blade.
2. On the **LabSQLDW** database blade, click **Open in Power BI**. This will open a new browser window showing the Power BI sign in page (https://powerbi.microsoft.com).
3. Click **Sign in** and specify the credentials of the SQLDWAdmin account.
4. On the **We know you** page, click **Sign in**
5. On the **Almost there** page, click **Start**.
6. On the **Invite more people** page, click **Skip**.
7. When prompted with the message **This content pack is a Power BI Pro feature**, click **Try Pro for free** followed by **Start trial** and **Close**.
8. In the **Azure SQL Data Warehouse** tile, click **Connect**
9. On the first page of the **Connect to Azure SQL Data Warehouse** wizard, type in the following and click **Next**:
   - Server: the name of the server hosting the Azure SQL Database Warehouse database (including the .database.windows.net suffix)
   - Database: **LabSQLDW**
10. On the next page of the wizard, type in the following and click **Sign in**:
    - Username: Student
    - Password: Pa55w.rd1234
11. Click the **Azure SQL Data Warehouse** tile.
12. This will open the database in the Power BI workspace

Create a sample report

1. In the Power BI interface, in the **Fields** pane, locate the **AggregateSales** view.
2. In the list of fields of the **AggregateSales** view, select the checkboxes next to the **PostalCode** and **SalesAmount** fields.
3. Power BI automatically recognizes that the fields reference geographic data and generates a map illustrating the corresponding data. You can use entries in the **View** menu to resize it on the workspace as needed.

4. Next, you will add to the workspace a bar graph representing amount of sales per customer income. To start, in the **Fields** pane, select the **SalesAmount** field.
5. Drag the **Customer Income** entry from the **Fields** pane and drop it onto the **Drag data fields here** section under the **Axis** label in the **Visualizations** pane. Note that this will automatically add a bar graph to the workspace

Note: You might need to resize your workspace to view the new chart.

6. Save the resulting report to the Documents folder to the Power bi workspace as Lab Report. This allows you to maintain list of reports that you want to have readily available for viewing or further editing.

Note: You also have the option (available from the **File** menu) to perform such tasks as printing, publishing to web, or exporting to PowerPoint (in preview at the time of authoring of this content).

Note: Once you have completed all the exercises, consider deleting the DBLabRG, SQLDBRG, DocumentDBRG, and SQLDWRG resource groups to avoid charges associated with the lab VM, Azure SQL databases, Azure DocumentDB, and Azure SQL Data Warehouse.